μPD7201/7201A
μPD72001
μPD7210
μPD72105

PRODUCT DESCRIPTION
COMMUNICATIONS CONTROLLERS

µPD7201/7201A MULTIPROTOCOL SERIAL COMMUNICATION CONTROLLER
µPD72001 ADVANCED MULTIPROTOCOL SERIAL CONTROLLER
µPD7210 GPIB CONTROLLER
µPD72105 LOCAL AREA NETWORK CONTROLLER
Table of Contents - uPD7201/7201A Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1-1</td>
</tr>
<tr>
<td>2 Pin Description</td>
<td>1-2</td>
</tr>
<tr>
<td>CPU Interface Signals</td>
<td>1-3</td>
</tr>
<tr>
<td>Channel Interface Signals</td>
<td>1-4</td>
</tr>
<tr>
<td>Modem Interface Signals</td>
<td>1-5</td>
</tr>
<tr>
<td>3 Functional Description</td>
<td>1-7</td>
</tr>
<tr>
<td>Receiver Section</td>
<td>1-7</td>
</tr>
<tr>
<td>Transmitter Section</td>
<td>1-8</td>
</tr>
<tr>
<td>CPU Interface Section</td>
<td>1-8</td>
</tr>
<tr>
<td>4 Registers</td>
<td>1-13</td>
</tr>
<tr>
<td>Control Register 0 (CR0)</td>
<td>1-13</td>
</tr>
<tr>
<td>Control Register 1 (CR1)</td>
<td>1-15</td>
</tr>
<tr>
<td>Control Register 2, Channel A (CR2A)</td>
<td>1-16</td>
</tr>
<tr>
<td>Control Register 2, Channel B (CR2B)</td>
<td>1-18</td>
</tr>
<tr>
<td>Control Register 3 (CR3)</td>
<td>1-18</td>
</tr>
<tr>
<td>Control Register 4 (CR4)</td>
<td>1-19</td>
</tr>
<tr>
<td>Control Register 5 (CR5)</td>
<td>1-20</td>
</tr>
<tr>
<td>Control Registers 6 and 7 (CR6 and CR7)</td>
<td>1-21</td>
</tr>
<tr>
<td>Status Register 0 (SR0)</td>
<td>1-22</td>
</tr>
<tr>
<td>Status Register 1 (SR1)</td>
<td>1-24</td>
</tr>
<tr>
<td>Status Register Channel B (SR2B)</td>
<td>1-25</td>
</tr>
<tr>
<td>Status Registers 3 and 4 (SR3 and SR4)</td>
<td>1-26</td>
</tr>
<tr>
<td>Tx Length Register, High Byte and Low Byte (TxLR-H and TxLR-L)</td>
<td>1-26</td>
</tr>
<tr>
<td>5 Operating Modes</td>
<td>1-27</td>
</tr>
<tr>
<td>Asynchronous Mode</td>
<td>1-27</td>
</tr>
<tr>
<td>Synchronous Mode</td>
<td>1-30</td>
</tr>
<tr>
<td>HDLC (SDLC) Mode</td>
<td>1-36</td>
</tr>
<tr>
<td>6 Timing</td>
<td>1-45</td>
</tr>
<tr>
<td>CPU Interface Timing</td>
<td>1-45</td>
</tr>
<tr>
<td>Transmission/Reception Timing</td>
<td>1-51</td>
</tr>
</tbody>
</table>
Table of Contents - uPD7201/7201A Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>System Configurations</td>
</tr>
<tr>
<td></td>
<td>Status polling Mode</td>
</tr>
<tr>
<td></td>
<td>Channels A and B in Interrupt Mode</td>
</tr>
<tr>
<td></td>
<td>Channel A in DMA Mode and Channel B in Interrupt Mode</td>
</tr>
<tr>
<td></td>
<td>Channels A and B in DMA Mode</td>
</tr>
<tr>
<td>8</td>
<td>Electrical Specification</td>
</tr>
<tr>
<td>9</td>
<td>Appendix (uPD7201A and uPD7201 Comparison)</td>
</tr>
<tr>
<td>10</td>
<td>Application Note</td>
</tr>
</tbody>
</table>
# Table of Contents - uPD72001 Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>2-1</td>
</tr>
<tr>
<td>Features</td>
<td>2-1</td>
</tr>
<tr>
<td>Pin Configuration</td>
<td>2-3</td>
</tr>
<tr>
<td>Block Diagram</td>
<td>2-4</td>
</tr>
<tr>
<td>1. Pin Functions</td>
<td>2-5</td>
</tr>
<tr>
<td>1.1 System Interface Pins</td>
<td>2-5</td>
</tr>
<tr>
<td>1.2 Send/Receive Operation Pins</td>
<td>2-11</td>
</tr>
<tr>
<td>2. Configuration</td>
<td>2-15</td>
</tr>
<tr>
<td>2.1 System Clock Section</td>
<td>2-15</td>
</tr>
<tr>
<td>2.2 System Interface Section</td>
<td>2-15</td>
</tr>
<tr>
<td>2.2.1 Interface control circuit</td>
<td>2-16</td>
</tr>
<tr>
<td>2.2.2 Registers</td>
<td>2-16</td>
</tr>
<tr>
<td>2.3 Transmitter/Receiver Section</td>
<td>2-17</td>
</tr>
<tr>
<td>3. Register Configuration</td>
<td>2-18</td>
</tr>
<tr>
<td>3.1 Outline of Registers</td>
<td>2-18</td>
</tr>
<tr>
<td>3.2 Control Registers (CR)</td>
<td>2-28</td>
</tr>
<tr>
<td>3.2.1 Control Register 0 (CR0)</td>
<td>2-30</td>
</tr>
<tr>
<td>3.2.2 Control Register 1 (CR1)</td>
<td>2-34</td>
</tr>
<tr>
<td>3.2.3 Control Register 2A (CR2A)</td>
<td>2-39</td>
</tr>
<tr>
<td>3.2.4 Control Register 2B (CR2B)</td>
<td>2-42</td>
</tr>
<tr>
<td>3.2.5 Control Register 3 (CR3)</td>
<td>2-43</td>
</tr>
<tr>
<td>3.2.6 Control Register 4 (CR4)</td>
<td>2-48</td>
</tr>
<tr>
<td>3.2.7 Control Register 5 (CR5)</td>
<td>2-50</td>
</tr>
<tr>
<td>3.2.8 Control Register 6 (CR6)</td>
<td>2-54</td>
</tr>
<tr>
<td>3.2.9 Control Register 7 (CR7)</td>
<td>2-55</td>
</tr>
<tr>
<td>3.2.10 Control Register 8 (CR8)</td>
<td>2-56</td>
</tr>
<tr>
<td>3.2.11 Control Register 9 (CR9)</td>
<td>2-56</td>
</tr>
<tr>
<td>3.2.12 Control Register 10 (CR10)</td>
<td>2-57</td>
</tr>
<tr>
<td>3.2.13 Control Register 11 (CR11)</td>
<td>2-62</td>
</tr>
<tr>
<td>3.2.14 Control Register 12 (CR12)</td>
<td>2-65</td>
</tr>
<tr>
<td>3.2.15 Control Register 13 (CR13)</td>
<td>2-68</td>
</tr>
<tr>
<td>3.2.16 Control Register 14 (CR14)</td>
<td>2-71</td>
</tr>
<tr>
<td>3.2.17 Control Register 15 (CR15)</td>
<td>2-74</td>
</tr>
</tbody>
</table>
Table of Contents - uPD72001 Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3 Status Registers (SR)</td>
<td>2-77</td>
</tr>
<tr>
<td>3.3.1 Status Register 0 (SRO)</td>
<td>2-77</td>
</tr>
<tr>
<td>3.3.2 Status Register 1 (SR1)</td>
<td>2-82</td>
</tr>
<tr>
<td>3.3.3 Status Register 2B (SR2B)</td>
<td>2-88</td>
</tr>
<tr>
<td>3.3.4 Status Register 3 (SR3)</td>
<td>2-89</td>
</tr>
<tr>
<td>3.3.5 Status Register 4A (SR4A)</td>
<td>2-92</td>
</tr>
<tr>
<td>3.3.6 Status Register 8 (SR8)</td>
<td>2-94</td>
</tr>
<tr>
<td>3.3.7 Status Register 9 (SR9)</td>
<td>2-95</td>
</tr>
<tr>
<td>3.3.8 Status Register 10 (SR10)</td>
<td>2-95</td>
</tr>
<tr>
<td>3.3.9 Status Register 11 (SR11)</td>
<td>2-98</td>
</tr>
<tr>
<td>3.3.10 Status Register 12 (SR12)</td>
<td>2-98</td>
</tr>
<tr>
<td>3.3.11 Status Register 13 (SR13)</td>
<td>2-98</td>
</tr>
<tr>
<td>3.3.12 Status Register 14 (SR14)</td>
<td>2-99</td>
</tr>
<tr>
<td>3.3.13 Status Register 15 (SR15)</td>
<td>2-99</td>
</tr>
<tr>
<td>4. System Configuration Example</td>
<td>2-100</td>
</tr>
<tr>
<td>5. Target Specs</td>
<td>2-101</td>
</tr>
</tbody>
</table>
Table of Contents - uPD7210 Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Introduction</td>
<td>3-1</td>
</tr>
<tr>
<td>1.1 General Description</td>
<td>3-3</td>
</tr>
<tr>
<td>1.2 IEEE STD-448-1978</td>
<td>3-4</td>
</tr>
<tr>
<td>2. Functional Description</td>
<td>3-6</td>
</tr>
<tr>
<td>2.1 Pin Description</td>
<td>3-7</td>
</tr>
<tr>
<td>3. Internal Registers</td>
<td>3-12</td>
</tr>
<tr>
<td>3.1 Data Registers</td>
<td>3-12</td>
</tr>
<tr>
<td>3.2 Interrupt Registers</td>
<td>3-13</td>
</tr>
<tr>
<td>3.2.1 Interrupt Bits</td>
<td>3-13</td>
</tr>
<tr>
<td>3.2.2 Non-Interrupt Bits</td>
<td>3-17</td>
</tr>
<tr>
<td>3.3 Serial Poll Registers</td>
<td>3-17</td>
</tr>
<tr>
<td>3.4 Address Mode Status Registers</td>
<td>3-18</td>
</tr>
<tr>
<td>3.4.1 Selecting T/R2 and T/R3 Pin Function</td>
<td>3-18</td>
</tr>
<tr>
<td>3.4.2 Selecting Address Mode</td>
<td>3-19</td>
</tr>
<tr>
<td>3.5 Address Registers</td>
<td>3-20</td>
</tr>
<tr>
<td>3.6 Command Pass Through Registers</td>
<td>3-22</td>
</tr>
<tr>
<td>3.7 End of String (EOS) Register</td>
<td>3-22</td>
</tr>
<tr>
<td>3.8 Auxiliary Mode Register</td>
<td>3-23</td>
</tr>
<tr>
<td>3.8.1 Auxiliary Commands</td>
<td>3-24</td>
</tr>
<tr>
<td>3.8.2 Internal Counter</td>
<td>3-27</td>
</tr>
<tr>
<td>3.8.3 Auxiliary Register A</td>
<td>3-28</td>
</tr>
<tr>
<td>3.8.4 Auxiliary Register B</td>
<td>3-30</td>
</tr>
<tr>
<td>3.8.5 Auxiliary Register E</td>
<td>3-30</td>
</tr>
<tr>
<td>3.8.6 Parallel Poll Register</td>
<td>3-31</td>
</tr>
</tbody>
</table>
Table of Contents - uPD7210 Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Using the uPD7210</td>
<td>3-32</td>
</tr>
<tr>
<td>4.1 Transmitting Commands</td>
<td>3-32</td>
</tr>
<tr>
<td>4.2 Processing the Undefined Commands</td>
<td>3-32</td>
</tr>
<tr>
<td>4.3 Processing Address Pass Through</td>
<td>3-32</td>
</tr>
<tr>
<td>4.4 Beginning Data Transfer</td>
<td>3-33</td>
</tr>
<tr>
<td>4.5 Transmitting Data</td>
<td>3-33</td>
</tr>
<tr>
<td>4.6 Receiving Data</td>
<td>3-34</td>
</tr>
<tr>
<td>4.6.1 Normal Handshake Mode (A0=A1=0)</td>
<td>3-34</td>
</tr>
<tr>
<td>4.6.2 RFD Holdoff on All Data Mode (A0=1, A1=0)</td>
<td>3-34</td>
</tr>
<tr>
<td>4.6.3 RFD Holdoff on End Mode (A0=0, A1=1)</td>
<td>3-34</td>
</tr>
<tr>
<td>4.6.4 Continuous Mode (A0=A1=1)</td>
<td>3-35</td>
</tr>
<tr>
<td>4.7 Completing Data Block Transfer</td>
<td>3-35</td>
</tr>
<tr>
<td>4.7.1 Placing EOS Byte After Data Block</td>
<td>3-35</td>
</tr>
<tr>
<td>4.7.2 Using the EOI Line</td>
<td>3-35</td>
</tr>
<tr>
<td>4.7.3 Transmission of the EOS Message</td>
<td>3-35</td>
</tr>
<tr>
<td>4.7.4 Transmission of the END Message</td>
<td>3-36</td>
</tr>
<tr>
<td>4.7.5 Detecting the EOS Message</td>
<td>3-36</td>
</tr>
<tr>
<td>4.7.6 Detecting the END Message</td>
<td>3-36</td>
</tr>
<tr>
<td>4.8 Discouting Data Transfer</td>
<td>3-36</td>
</tr>
<tr>
<td>4.8.1 Using the Take Control Asynchronously Command</td>
<td>3-36</td>
</tr>
<tr>
<td>4.8.2 Using the Take Control Synchronously Command</td>
<td>3-37</td>
</tr>
<tr>
<td>4.8.3 Using the Take Control Synchronously on End Command</td>
<td>3-37</td>
</tr>
<tr>
<td>4.9 Serial Polling</td>
<td>3-37</td>
</tr>
<tr>
<td>4.10 Parallel Polling</td>
<td>3-38</td>
</tr>
<tr>
<td>4.11 Parallel Poll Protocol</td>
<td>3-38</td>
</tr>
</tbody>
</table>
# Table of Contents - uPD7210 Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. GPIB-Interface using uPD7210</td>
<td>3-40</td>
</tr>
<tr>
<td>5.1 Hardware</td>
<td>3-40</td>
</tr>
<tr>
<td>5.2 Software</td>
<td>3-43</td>
</tr>
<tr>
<td>6. Electrical Specification</td>
<td>3-49</td>
</tr>
<tr>
<td>7. Appendix</td>
<td>3-55</td>
</tr>
<tr>
<td>8. Application Note</td>
<td>3-73</td>
</tr>
</tbody>
</table>
Table of Contents - uPD72105 Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>4-1</td>
</tr>
<tr>
<td>Features</td>
<td>4-1</td>
</tr>
<tr>
<td>Pin Connection</td>
<td>4-2</td>
</tr>
<tr>
<td>Block Diagram</td>
<td>4-3</td>
</tr>
<tr>
<td>1. Pin Functions</td>
<td>4-4</td>
</tr>
<tr>
<td>2. Internal Configuration</td>
<td>4-9</td>
</tr>
<tr>
<td>2.1 Internal Controller</td>
<td>4-9</td>
</tr>
<tr>
<td>2.2 System Interface</td>
<td>4-9</td>
</tr>
<tr>
<td>2.3 DMA Controller</td>
<td>4-9</td>
</tr>
<tr>
<td>2.4 RxFIFO</td>
<td>4-9</td>
</tr>
<tr>
<td>2.5 TxFIFO</td>
<td>4-9</td>
</tr>
<tr>
<td>2.6 Receiver</td>
<td>4-10</td>
</tr>
<tr>
<td>2.7 Transmitter</td>
<td>4-10</td>
</tr>
<tr>
<td>3. Interfacing with Host System</td>
<td>4-11</td>
</tr>
<tr>
<td>3.1 Control Register</td>
<td>4-11</td>
</tr>
<tr>
<td>3.2 Status Register</td>
<td>4-12</td>
</tr>
<tr>
<td>3.3 Address FIFO</td>
<td>4-12</td>
</tr>
</tbody>
</table>
Table of Contents - uPD72105 Product Description

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Commands</td>
<td>4-13</td>
</tr>
<tr>
<td>4.1 Types of Commands</td>
<td>4-13</td>
</tr>
<tr>
<td>4.2 Command Functions</td>
<td>4-15</td>
</tr>
<tr>
<td>4.2.1 INIT</td>
<td>4-15</td>
</tr>
<tr>
<td>4.2.2 SEND</td>
<td>4-18</td>
</tr>
<tr>
<td>4.2.3 SETUP RCV</td>
<td>4-20</td>
</tr>
<tr>
<td>4.2.4 END RCV</td>
<td>4-22</td>
</tr>
<tr>
<td>4.2.5 WAIT RCV</td>
<td>4-23</td>
</tr>
<tr>
<td>4.2.6 RCV LIST</td>
<td>4-25</td>
</tr>
<tr>
<td>4.2.7 ECHO</td>
<td>4-33</td>
</tr>
<tr>
<td>4.2.8 LOOP BACK</td>
<td>4-34</td>
</tr>
<tr>
<td>4.2.9 INIT MONIT</td>
<td>4-35</td>
</tr>
<tr>
<td>4.2.10 MONIT OFF</td>
<td>4-37</td>
</tr>
<tr>
<td>4.2.11 MONIT ON</td>
<td>4-37</td>
</tr>
<tr>
<td>4.2.12 SET PARM</td>
<td>4-38</td>
</tr>
<tr>
<td>4.2.13 GET PARM</td>
<td>4-41</td>
</tr>
<tr>
<td>4.2.14 NEW CHAIN</td>
<td>4-41</td>
</tr>
<tr>
<td>4.2.15 NEW DEFLT ADDR</td>
<td>4-42</td>
</tr>
<tr>
<td>4.2.16 CLR STAT</td>
<td>4-42</td>
</tr>
<tr>
<td>4.2.17 GET STAT</td>
<td>4-42</td>
</tr>
<tr>
<td>5. Control</td>
<td>4-45</td>
</tr>
<tr>
<td>6. System Configuration Examples</td>
<td>4-49</td>
</tr>
<tr>
<td>7. Target Specs</td>
<td>4-53</td>
</tr>
</tbody>
</table>
µPD7201/7201A
MULTIPROTOCOL SERIAL
COMMUNICATION CONTROLLER
Description

The μPD7201 and μPD7201A Multiprotocol Serial Communication Controllers (MPSCC) are microcomputer peripherals that serve as multi-function peripheral devices capable of supporting a variety of serial data communications. The basic function of the MPSCC is to interface a microcomputer system (CPU) with serial data processing devices such as modems. The μPD7201A is an improved version of the μPD7201. Added features are described in the text, and listed in Appendix A.

The MPSCC controls three protocols:
- Asynchronous (start-stop synchronous)
- Bit-oriented synchronous
- Byte-oriented synchronous

Asynchronous operation provides stop bit lengths of 1, 1 1/2, or 2 bits. It also provides transmit and receive clock rates programmable to 1, 16, 32, and 64 times the input frequency.

Bit-oriented synchronous operations such as HDLC (see note 1) and SDLC (see note 2) include the following features:
- Zero insertion/deletion
- Flag transmission and removal detection
- Fractional character processing
- Secondary address/global address
- End of frame
- Abort transmission detection

Bit-oriented synchronous operation also has a 16-bit transmit length counter and register to control the number of transmit characters (7201A only).

Byte-oriented synchronous operation such as Monosync, External Sync, and Bisync (see note 3) operation includes program-selectable SYNC characters and SYNC character transmission/removal detection.

NOTES:
1. High-level Data Link Control (ISO)
2. Synchronous Data Link Control (IBM)
3. Binary Synchronous Communication (IBM)

Features
- μPD8085/8086 bus compatible
- High-speed transmit/receive operations of 1.1 Mb/s at 5 MHz system clock (7201A) or 880 kb/s at 4 MHz (7201)
- Data buffering
- 5- to 8-bit character length
- Add/checking for odd, even, or no parity
- Cyclic Redundancy Check (CRC) generation/checking (CCITT-0)
- Error checking (parity, framing, Rx overrun, CRC)
- Modern control (two channels, four signals)
- 21/29 control, data, and status registers (7201/7201A)
- Various interrupt functions
- +5 V single power supply
- N-channel MOS
- 40-pin plastic DIP (μPD7201AC, μPD7201C)
- 40-pin ceramic DIP (μPD7201AD, μPD7201D)

Figure 1-1 shows the MPSCC functional block diagram.

![MPSCC Block Diagram](image-url)
Pin Identification

Figure 1-2 shows the pin configuration of the MPSCC. Table 1-1 gives the pin number, signal name, direction, and definition for each signal. The signal names and definitions of pins with two functions are separated by a slash (/).

Figure 1-2. Pin Configuration

Table 1-1. MPSCC Pin Identification

<table>
<thead>
<tr>
<th>No.</th>
<th>Symbol</th>
<th>Direction</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CLK</td>
<td>In</td>
<td>System clock</td>
</tr>
<tr>
<td>2</td>
<td>RESET</td>
<td>In</td>
<td>Reset</td>
</tr>
<tr>
<td>3</td>
<td>DCC A</td>
<td>In</td>
<td>Data carrier detect, channel A</td>
</tr>
<tr>
<td>4</td>
<td>Rx CB</td>
<td>In</td>
<td>Receive clock, channel B</td>
</tr>
<tr>
<td>5</td>
<td>DCC B</td>
<td>In</td>
<td>Data carrier detect, channel B</td>
</tr>
<tr>
<td>6</td>
<td>CTS B</td>
<td>In</td>
<td>Clear to send, channel B</td>
</tr>
<tr>
<td>7</td>
<td>TxB</td>
<td>In</td>
<td>Transmit clock, channel B</td>
</tr>
<tr>
<td>8</td>
<td>TxD B</td>
<td>Out</td>
<td>Transmit data, channel B</td>
</tr>
<tr>
<td>9</td>
<td>Rx DB</td>
<td>In</td>
<td>Receive data, channel B</td>
</tr>
<tr>
<td>10</td>
<td>RTS B/SYN C</td>
<td>In/Out</td>
<td>Request to send, channel B/Synchronization, channel B</td>
</tr>
<tr>
<td>11</td>
<td>W AIT B/DRO TxA</td>
<td>Out</td>
<td>Wait, channel B/DMA request, transmit, channel A</td>
</tr>
<tr>
<td>12-19</td>
<td>D 7-D 0</td>
<td>In/Out</td>
<td>Data bus</td>
</tr>
<tr>
<td>20</td>
<td>GND</td>
<td></td>
<td>Ground potential</td>
</tr>
<tr>
<td>21</td>
<td>WR</td>
<td>In</td>
<td>Write</td>
</tr>
<tr>
<td>22</td>
<td>R D</td>
<td>In</td>
<td>Read</td>
</tr>
<tr>
<td>23</td>
<td>CS</td>
<td>In</td>
<td>Chip select</td>
</tr>
<tr>
<td>24</td>
<td>C / T</td>
<td>In</td>
<td>Control/data</td>
</tr>
<tr>
<td>25</td>
<td>B / A</td>
<td>In</td>
<td>Channel B/Channel A</td>
</tr>
<tr>
<td>26</td>
<td>D TR B/H AI</td>
<td>In/Out</td>
<td>Data terminal ready, channel B/Hold acknowledge input</td>
</tr>
<tr>
<td>27</td>
<td>INT A K</td>
<td>In</td>
<td>Interrupt acknowledge</td>
</tr>
<tr>
<td>28</td>
<td>INT</td>
<td>Out</td>
<td>Interrupt request</td>
</tr>
<tr>
<td>29</td>
<td>PRI/DRO TxB</td>
<td>In/Out</td>
<td>Priority input/DMA request, receive, channel B</td>
</tr>
<tr>
<td>30</td>
<td>PRI/DRO TxA</td>
<td>Out</td>
<td>Priority output/DMA request, transmit, channel B</td>
</tr>
<tr>
<td>31</td>
<td>D TR A/H AO</td>
<td>Out</td>
<td>Data terminal ready, channel A/Hold acknowledge output</td>
</tr>
<tr>
<td>32</td>
<td>W AIT A/DRO TxA</td>
<td>Out</td>
<td>Wait, channel A/DMA request, receive, channel A</td>
</tr>
<tr>
<td>33</td>
<td>SYN C A</td>
<td>In/Out</td>
<td>Synchronization, channel A</td>
</tr>
<tr>
<td>34</td>
<td>Rx DA</td>
<td>In</td>
<td>Receive data, channel A</td>
</tr>
<tr>
<td>35</td>
<td>Rx CA</td>
<td>In</td>
<td>Receive clock, channel A</td>
</tr>
<tr>
<td>36</td>
<td>T x C A</td>
<td>In</td>
<td>Transmit clock, channel A</td>
</tr>
<tr>
<td>37</td>
<td>T x DA</td>
<td>Out</td>
<td>Transmit data, channel A</td>
</tr>
<tr>
<td>38</td>
<td>RTS A</td>
<td>Out</td>
<td>Request to send, channel A</td>
</tr>
<tr>
<td>39</td>
<td>CTS A</td>
<td>In</td>
<td>Clear to send, channel A</td>
</tr>
<tr>
<td>40</td>
<td>VCC</td>
<td></td>
<td>Power supply</td>
</tr>
</tbody>
</table>
The pins of the MPSCC function in three interface categories: CPU, channel, and modem. This section describes the pin functions of each interface category.

Figure 2-1 shows pinouts of the MPSCC by each interface category.

**Figure 2-1. Functional Pinout**

### CPU INTERFACE SIGNALS

The following signals function as the CPU interface:

**RESET (Reset)**

A low-level input to this pin for at least one clock cycle causes the MPSCC to perform a system reset. Since a system reset may cause data loss to control registers, write or rewrite data to the control registers before data transmission or reception.

The state of the MPSCC when the system is reset is:

- Transmitter/receiver: Disabled
- Interrupt/DMA: Disabled
- TxDA/TxDB output: Marking 1s
- Modern control output: 1

See Section 4, D7-D0 (Command Bits) for a comparison of the states of each register and pin when the system is reset and when the channel reset command is issued.

**CLK (System Clock)**

Inputs a single-phase, system clock that is TTL-compatible. The system clock rate must be 4.5 times faster than the data rate.

**D7 - D0 (Data Bus)**

D7-D0 is an 8-bit bidirectional data bus. These tri-state pins connect the MPSCC to a standard CPU such as the μPD8080AF, μPD8085A, or μPD8086. The data bus transmits data, commands, or status signals between the MPSCC and CPU.

**CS (Chip Select)**

Provides access to the Tx or Rx buffer or the status or control register specified by the register pointer, B/A, or C/D pins.

**WR (Write)**

Host sends a low-level signal when data or commands are transmitted from the CPU or memory to the MPSCC.

**RD (Read)**

Host sends a low-level signal when data or commands are transmitted from the CPU or memory to the MPSCC.

**C/D (Control/Data)**

Indicates the type of data on the data bus when a write or read operation is performed. A high level indicates that commands or status data is on the data bus. A low level indicates that transmit or receive data is on the data bus.

**B/A (Channel B/Channel A)**

Indicates the channel to or from which data on the data bus is written or read, when a write or read operation is performed. A high level specifies channel B. A low level specifies channel A.

Table 2-1 shows the different states and functions of the C/D, WR, RD, CS, and B/A signals.

### Table 2-1. Signal Functions

<table>
<thead>
<tr>
<th>C/D</th>
<th>WR</th>
<th>RD</th>
<th>CS</th>
<th>B/A</th>
<th>Channel</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>A</td>
<td>1</td>
<td>Writes transmit data to the Tx buffer</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>A</td>
<td>1</td>
<td>Reads receive data from the Rx buffer</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>A</td>
<td>1</td>
<td>Writes data to the command parameter (CRF-CR0) registers</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>A</td>
<td>1</td>
<td>Reads data from the status vector (SR4-SR0) registers</td>
</tr>
</tbody>
</table>

1-3
INT (Interrupt Request)
Outputs an interrupt request signal. A low-level signal indicates an interrupt within the MPSCC. INT is an open-drain output.

INTACK (Interrupt Acknowledge)
Inputs an interrupt acknowledge signal in response to an interrupt request signal. When several MPSCCs simultaneously request an interrupt, the one with the top priority accepts this signal. The MPSCC may also output an interrupt vector to the data bus, depending on the specified mode. If this pin is not used, it must be held high with a pull-up resistor.

PRI/DRQRxB (Priority Input/DMA Request RxB)
Transmits or receives the PRI and DRQRxB signals. The state of the INT/DMA mode bits (D1 and D0) of the CR2A register specify the function of this pin.
PRI is used when two or more interrupt devices are used to form a daisy chain. A low level indicates that an interrupt device assigned a higher priority is not receiving service by the CPU's interrupt routine. When only one MPSCC is used, this pin must be held low (except when using a μPD7201A in 85-3 Vectored Mode).
DRQRxB is used as a DMA request signal to the DMA controller. The signal is active when data is input to the receive buffer of channel B (Rx Character Available).

PRO/DRQTxB (Priority Output/DMA Request TxB)
The state of the INT/DMA mode bits (D1 and D0) of the CR2A register specify the function of this pin.
PRO is paired with the PRI signal to control interrupt priority by using a daisy chain. When PRI is high, PRO outputs a high-level signal. When PRI is low, the PRO signal goes high when the MPSCC requests an interrupt from the CPU. If the interrupt signal is not output to the CPU, PRO remains low even if PRI is high.
DRQTxB outputs a DMA request signal to the DMA controller and becomes active when the Tx buffer of channel B becomes empty.

WAITA/DRQRxA (Wait A/DMA Request RxA)
The state of the INT/DMA mode bits (D1 and D0) of the CR2A register specify the function of this pin.
WAITA requests the CPU to enter the WAIT state via the CPU's READY pin. Used as an open-drain output.
DRQRxA outputs a DMA request signal to the DMA controller. This signal becomes active when a character is input to the Rx buffer of channel A.

WAITB/DRQTxA (Wait B/DMA Request TxA)
The state of the INT/DMA mode bits (D1 and D0) of the CR2A register specify the function of this pin.
WAITB requests the CPU to enter the WAIT state via the CPU's READY pin. Used as an open-drain output when the WAIT function is used.
DRQTxA outputs a DMA request signal to the DMA controller and becomes active when the Tx buffer of channel A becomes empty.

DTRB/HA1 (Data Terminal Ready B/Hold Acknowledge Input)
The state of INT/DMA mode bits (D1 and D0) of the CR2A register specify the function of this pin.
The level of DTRB changes according to the state of the DTR bit (D7 of the CR5 register) of channel B. When the DTR bit is 1, DTRB becomes 0. When the DTR bit is 0, the DTRB signal becomes 1.
HA1 is used to control the DMA priority with a daisy chain. Normally, HLDA is input to the HA1 pin of the MPSCC with the highest DMA priority. A low-level input to this pin indicates that this particular MPSCC is selected for the DMA operation.

DTRA/HAO (Data Terminal Ready A/Hold Acknowledge Output)
The state of INT/DMA mode bits (D1 and D0) of the CR2A register specify the function of this pin.
The level of DTRA changes according to the state of the DTR bit of channel A (D7 of the CR5 register). When the DTR bit is 1, DTRA is 0. When the DTR bit is 0, DTRA is 1.
HAO is paired with HA1 to control the DMA daisy chain. When HA1 is at a high level, HAO outputs a high. When HA1 is at a low level, HAO outputs a high when the MPSCC requests a DMA operation. This inhibits another MPSCC daisy chained in the lower order from performing the DMA operation. If the DMA request signal is not output from the MPSCC, a low-level signal is output and DMA operations by the lower-order MPSCCs in the daisy chain are enabled.

CHANNEL INTERFACE SIGNALS
The following signals function as the channel interface.
RxDA, RxDB (Receive Data A/B)
Inputs receive data.
TxDA, TxDB (Transmit Data A/B)
Outputs transmit data.
**RxCA, RxCB (Receive Clock A/B)**

Inputs clocks for sampling receive data. Sampling is performed on the leading edge of the RxC signal.

When the asynchronous mode is specified, the clock rate of the TxC signal must be a 1, 16, 32, or 64 multiple of the data rate.

**TxCA, TxCB (Transmit Clock A/B)**

Inputs clocks for transmit data. Data is output at the leading edge of the TxC signal. When the asynchronous mode is specified, the clock rate of the TxC signal must be a 1, 16, 32, or 64 multiple of the data rate.

**SYNCA (Synchronization A)**

The function of this pin depends on the operating protocol of the MPSCC.

**Asynchronous Mode (Input).** If external/synchronous (E/S) interrupts are enabled, an interrupt occurs at the leading or trailing edge of the SYNCA signal (or according to the state of the E/S bit). Inverted data from the SYNCA pin is latched at the SYNCA/Hunt bit (D4 of the SR0 register).

**External Asynchronous Mode (External/Sync) (Input).**

The E/S interrupt operation and function of the E/S bit in this mode is similar to the asynchronous mode.

If the MPSCC is in the hunt phase (Section 4, D4 (Enter Hunt Phase)) when the program specifies the external/sync mode, the MPSCC goes out of the hunt phase at the leading edge of an input SYNCA signal. The MPSCC then starts assembling characters, beginning from the character sampled at the leading edge of the RxC signal input before the SYNCA signal.

When the SYNC character has been detected, the external synchronization circuit resets the SYNCA signal to a low level after two cycles of the RxC signal. The SYNCA signal must be held low until the character synchronization is lost or a new message starts.

**Internal Synchronous Mode (Monosync) (Output).** Outputs a low-level signal each time the SYNC character is detected in received data.

**Internal Synchronous Mode (Bisync) (Output).** Outputs a low-level signal only when the first SYNC character is received after the MPSCC has entered the hunt phase.

**RTSB/SYNCB (Request to Send B/ Synchronization B)**

Transmits or receives the RTSB and SYNCB signals, respectively. The state of the RTSB/SYNCB select bit (D7 of the CR2A register) specifies which function the pin performs.

When the RTSB/SYNCB select bit is 0, the RTSB signal is selected. When the RTSB/SYNCB select bit is 1, the SYNCB signal is selected.

The functions of the RTSB and SYNCB signals are the same as those of the RTSA and SYNCA signals, except that the RTSB and SYNCB signals are applied to channel B.

**MODEM INTERFACE SIGNALS**

The following signals function as the modem interface.

**CTSIA, CTSB (Clear to Send A/B)**

**DCDA, DCDB (Data Carrier Detect A/B)**

The CTS and DCD pins have similar functions. The E/S interrupt is generated (when interrupts are enabled) at the leading or trailing edge of these signals and latched at the E/S bits (D2 and D3) of the SR0 register. The CTS and DCD pins can also be used as a trigger input in the auto enable mode. In this case, when bit D3 of the CR3 is 1, either the transmitter (CTS) or receiver (DCD) is enabled by the presence of a low-level signal. When the MPSCC is not in auto enable mode, these two pins are used as general purpose input pins.

**RTSA (Request to Send A)**

The state of the RTS bit (D1 of the CR5 register) controls this pin. If the RTS bit is reset in the asynchronous mode, a high level will not be output on the RTS pin until all transmit characters are written and the all sent bit (D0 of the SR1 register) is set.

In the synchronous mode, the state of the RTS bit is used as is. That is, when the RTS bit is 0, the RTS pin is 1. When the RTS bit is 1, the RTS pin is 0.
The MPSCC consists of a transmitter/receiver and CPU interface section. The transmitter/receiver section has two full-duplex channels (A and B) and control circuits for communication with other serial data processing devices via a modem. Both channels A and B have a receiver and transmitter and identical structure. Figure 3-1 shows the flow of transmit/receive data in each channel.

The CPU interface section has status and control registers, interrupt and DMA control circuitry, and a data bus buffer.

**RECEIVER SECTION**

Receive (Rx) data input from the RxD terminal is converted into 8-bit data by the Rx shift register and transferred to the Rx data buffer. The operation mode and character length of the data determines the path along which data flows.

**Serial Data Path**

In the asynchronous mode, serial data received by the MPSCC is transmitted to the Rx shift register by two methods. If the character of the received data consists of 7 or 8 bits, it is transferred via a 3-bit buffer. If it consists of 5 or 6 bits, the data is directly input to the Rx shift register.

In the synchronous mode, the serial data is compared with the SYNC character which was input in the 8-bit shift register while the MPSCC is in the hunt phase. If the MPSCC goes out of the hunt phase, or if it is set in the external synchronous mode, serial data is transferred to the 3-bit buffer, bypassing the 8-bit shift register. When the MPSCC is in the hunt phase, input data is compared at the 8-bit shift register with the contents of the CR6 or CR7 register by the Rx SYNC register.

In the monosync mode (8-bit SYNC character), data is compared with the contents of the CR7 register.

In the bisync mode (16-bit SYNC character), the first 8 bits of the input data are compared with the contents of the CR6 register and the last 8 bits with the contents of the CR7 register. If the input data is matched with the contents of the above registers, the MPSCC goes out of the hunt phase. Input data then bypasses the 8-bit shift register as long as this synchronization is maintained. Note that the syn-
chronization may not be properly established when xxH SYNC characters (for example 55H or 33H) are used in the monosync or bisync mode.

In the HDLC mode, input data follows the same data path regardless of whether the MPSCC is in the hunt phase or synchronization is established. Input data is first transferred to the 8-bit shift register and the zeros of the input data are deleted if necessary. The MPSCC then also determines whether the input data contains a flag or abort sequence. The input data is subsequently transferred to the 3-bit buffer, then to the Rx shift register. If the register finds that the first five consecutive bits of the input data are 1s, the MPSCC then performs a function according to the state of the next (6th or 7th) bit(s) as follows:

<table>
<thead>
<tr>
<th>Rx Shift Register Data Bits</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7</td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 1 0</td>
<td>Deletes 0</td>
</tr>
<tr>
<td>1 1 1 1 1 0</td>
<td>Receives flag sequence</td>
</tr>
<tr>
<td>1 1 1 1 1 1</td>
<td>Receives abort sequence</td>
</tr>
</tbody>
</table>

In the 7201A, an abort sequence is only reported after a flag sequence has been detected.

Cyclic Redundancy Check (CRC)

The CRC calculation operation is performed by the Rx CRC calculator and the CRC comparator. Receive data is transferred to the Rx CRC calculator from the Rx shift register via the CRC delay register in the synchronous mode. In the HDLC mode, the receive data is directly transferred to the Rx CRC calculator from the 8-bit shift register.

In the bisync mode, the program must make a decision whether to include each receive character in the CRC calculation. The CRC delay register holds the receive data for 8 bits to compensate for the process. It then transfers the data to the Rx CRC calculator.

In the HDLC mode, flag patterns are not included in the CRC calculation and the calculation is performed on data other than flag patterns. The MPSCC determines whether the CRC calculation is performed on a given character.

Parallel Data Path and Error Display

Serial data is transferred to the Rx buffer (three 8-bit buffers) after it is converted into 8-bit parallel data by the Rx shift register. The Rx buffer has a capacity of three bytes, sufficient capacity for data processing to transfer parallel-converted Rx data into the CPU.

Error data concerning receive data is stored in three 4-bit error registers. Each register indicates parity, overrun, CRC/ framing, and end of frame errors. Each 4-bit error register has error data that corresponds to each byte of the 3-byte Rx buffer. The contents of the error registers can be read from bits $D_7$ to $D_4$ of the SR1 register.

TRANSMITTER SECTION

The 8-bit transmit data sent to the MPSCC is transferred to the Tx buffer via the internal data bus. At the same time, the SYNC character in the synchronous mode or a secondary address and a flag in the HDLC mode are loaded into control register 6 (CR6) and control register 7 (CR7). The contents of the Tx buffer and the CR6 and CR7 registers are then sent to the Tx shift register (3-bit shift control, two 8-bit shift registers and 1-bit delay) and converted to serial data.

Asynchronous Mode

In this mode, asynchronous data in the Tx buffer is transferred to the Tx shift register and converted to serial data. Then parity, start, and stop bits are appended to the serial-converted data. The data is then subsequently transferred to the Tx multiplexer at a specified clock rate for transmittal from the TxD pin.

Synchronous Mode

As in the asynchronous mode, transmit data is sent to the Tx shift register from the Tx buffer. The contents of the CR6 or CR7 registers are sent to the Tx shift register as the SYNC characters, or as a time-fill character for a starting message or when a Tx underrun error occurs. Data output from the Tx shift register is also sent to the Tx multiplexer and to the Tx CRC calculator.

HDLC Mode

The Tx data is sent to the Tx shift register from the Tx buffer, and a flag pattern is sent to the Tx shift register from the Tx buffer when a message starts and ends. The output data from the Tx shift register is sent to the zero insert logic circuit and to the Tx CRC calculator. In the zero insert logic circuit, a zero is inserted after any five consecutive 1s in all fields other than the flags (address, control, frame check).

CPU INTERFACE SECTION

The CPU interface consists of a register section and a control section.

Register Section

The register section consists of eight control registers (CR7-CR0) and five status registers (SR4-SR0) for the 7201A, or three status registers (SR0-SR2) for the 7201. The control registers control the operation mode of the MPSCC, the internal data flow, and retain SYNC characters or flag patterns. The status registers hold status information.

Note that the CR2 register performs different functions depending on the channel to which the register belongs. The CR2 register of channel A (CR2A) is used to determine the system configuration. The CR2 register of channel B
(CR2B) receives an interrupt vector. The interrupt vector written to the CR2B register can be read from the SR2B register. In addition to the CR0-CR7 registers, each channel of the 7201A has a 2-byte transmit length register. Section 4 gives detailed descriptions of the functions of each register.

Control Section

The control section provides an interface between the MPSCC and the main system, and controls data transmission/reception. Data communication between the MPSCC and CPU is performed via an 8-bit data bus consisting of the D7-D0 pins. The type of data transmitted and data transmission directions are controlled by input control signals. Table 3-1 shows how to:

- Select channels using control signals
- Specify a read or write operation
- Select control or status registers

### Table 3-1. MPSCC Control Signals

<table>
<thead>
<tr>
<th>c5</th>
<th>c3</th>
<th>r5</th>
<th>wr</th>
<th>intax</th>
<th>phi</th>
<th>rai</th>
<th>b x</th>
<th>channel</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>A</td>
<td>combination other than the above when c5 = 0. prohibited</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>A</td>
<td>reads receive data from Rx buffer and reads received by DMA operation in DMA-2 mode (7201A only)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>A</td>
<td>writes transmit data to Tx buffer and writes data transmitted by DMA operation in DMA-2 mode (7201A only)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>A</td>
<td>reads data from status vector registers (SR0-SR4)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>A</td>
<td>writes data to command parameter registers (CR0-CR7)</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes: 1. x = Don't care
2. If the RA1 pin is not used for a hold acknowledge input, it will be treated as a don't care pin.

Three data transmission methods exist for communication between the MPSCC and CPU: polling, interrupt, and DMA modes. Each channel (A or B) is set in one of these modes by the contents of the CR2A register. However, each channel's CR1 register determines whether to enable or disable either the interrupt or DMA mode.

The CR2A register also determines the priority of an MPSCC interrupt or DMA request. Table 3-2 shows how the CR2A register specifies the mode (interrupt or DMA), pin function, or priority. Table 3-3 shows the interrupt mode's selection by the CR2A register and the contents of the data bus (interrupt vector) in the interrupt acknowledge cycle.
Table 3-2. CR2A Register Control

<table>
<thead>
<tr>
<th>CR2A</th>
<th>Channel A</th>
<th>Channel B</th>
<th>Pin Function</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>D2</td>
<td>D1</td>
<td>D0</td>
<td>INT/DMA</td>
<td></td>
</tr>
<tr>
<td>0 0 0</td>
<td>INT</td>
<td>INT</td>
<td>WAITA, WAITB</td>
<td>RXA, TXA, RXB, TXB, E/SA, E/SB</td>
</tr>
<tr>
<td>1 0 0</td>
<td>INT</td>
<td>INT</td>
<td>PRI, PRO</td>
<td>RXA, TXA, RXB, TXB, E/SA, E/SB</td>
</tr>
<tr>
<td>x 0 1</td>
<td>DMA</td>
<td>INT</td>
<td>DRQRxA, DRQTxA, PRI, PRO</td>
<td>RXA, RXB, TXA, TXB, E/SA, E/SB</td>
</tr>
<tr>
<td>0 1 0</td>
<td>DMA</td>
<td>DMA</td>
<td>DRQRxB, DRQTxB</td>
<td>RXA, RXB, RXB, TXB, E/SA, E/SB</td>
</tr>
<tr>
<td>1 1 0</td>
<td>DMA</td>
<td>DMA</td>
<td>RxA, TxA, RxB, TxB</td>
<td>RXA, RXB, RXB, TXB, E/SA, E/SB</td>
</tr>
<tr>
<td>x 1 1</td>
<td>DMA</td>
<td>DMA</td>
<td>No priority</td>
<td>RXA, RXB, E/SA, E/SB</td>
</tr>
</tbody>
</table>

Notes: x = Don’t care
* These priorities are for the special receive and external status interrupts which can occur when using dual channel DMA operation.

Table 3-3. Vectored Interrupt Control

<table>
<thead>
<tr>
<th>CR2A</th>
<th>Interrupt Mode</th>
<th>INTA CYCLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>D5</td>
<td>D4</td>
<td>D3</td>
</tr>
<tr>
<td>1 0 0</td>
<td>85-1</td>
<td>0</td>
</tr>
<tr>
<td>1 0 1</td>
<td>85-2</td>
<td>0</td>
</tr>
<tr>
<td>1 1 1</td>
<td>85-3 Note 1</td>
<td>0</td>
</tr>
<tr>
<td>1 1 0</td>
<td>86 Note 2</td>
<td>0</td>
</tr>
</tbody>
</table>

Notes: 1 85-3 mode activates the INT line if an internal interrupt exists whether or not the PRI line is high. Therefore, a daisy chain using the PRI and PRO pins cannot be performed. (This mode is available on the µPD7201A only.)
2 8086 mode issues two INTAR pulses instead of three.

Data Transmission Between MPSCC and CPU

Data transmission between the CPU and MPSCC (or memory and the MPSCC) is performed in either polling, interrupt, or DMA mode.

Polling Mode. When data transmission is performed in the polling mode, the CPU must monitor the status of the MPSCC to learn the timing of the data transmission. The CPU reads status information from the MPSCC’s SR0 and SR1 registers and determines whether:
- a Tx data transmit (MPSCC to CPU) request exists,
- an Rx data receive (CPU to MPSCC) request exists, or
- an error occurred in the MPSCC.

If an Rx character available bit (D0 of the SR0 register) is set, for example, valid Rx data exists in the Rx buffer of the MPSCC, then the CPU must immediately read the data.

Also, the interrupt vector may be used in the SR2B register (vector bits V2-V0) for checking the status of the MPSCC. In this case, channels A and B must be set in the interrupt mode by bits D2 and D1 of the CR2A register, or in a nonvectored mode. The 85 mode is set when bits D2-D0 of the CR2A register are 000, 001, or 011. The 86 mode is set when bits D2-D0 of the CR2A register are 010. The status affects vector bit (D2 of the CR1B register) must also be set. Vector bits V2-V0 of the interrupt vector are modified in the nonvectored 85 mode and V2-V0 in the nonvectored 86 mode.

When operating in the 8080/5 modes, the MPSCC issues an 8080-type CALL CD v (hex) instruction where vv is the contents of control register 2B. Register 2B is modified by the cause of the interrupt if the status affects vector feature is enabled. An MPSCC programmed for 8085 master mode always places the CALL opcode on the data bus whether or not that MPSCC has a pending interrupt request. To avoid problems caused by momentary bus contention, never program more than one device to operate in this mode.
Interrupt Mode. The MPSCC has one interrupt request terminal (INT). When an interrupt source occurs inside the MPSCC, the INT signal becomes active and informs the CPU or interrupt controller of an interrupt request.

In a system using more than one MPSCC, the interrupt process can be performed through a daisy chain using the PRI and PRO pins. If an interrupt source occurs inside the MPSCC, the interrupt vector corresponding to the interrupt source can be generated when the vector mode is specified and the status affects vector bit is set.

Eight types of interrupt vectors, each corresponding to a given interrupt source inside the MPSCC, can be generated by setting the status affects vector bit (D2 of the CR1B register). These interrupt vectors allow direct branching to the process routine for each interrupt source. The MPSCC also has nonvected modes for CPUs that cannot read vectors in an interrupt cycle. In this case, the CPU determines what interrupt source is issuing the request by reading the contents of the status or vector registers of the MPSCC. The CPU will then execute the appropriate process routine.

Three interrupt sources can occur in the MPSCC.

- Receive (Rx)
- Transmit (Tx)
- External/Status (E/S)

The Rx interrupt occurs when the following conditions exist: the MPSCC receives data; the valid (character) data for the read operation to be performed by the CPU is loaded to the Rx buffer; and Rx interrupts are enabled. There are three modes of Rx interrupts.

First Rx Character. In this mode an interrupt occurs only when the first character is received by the MPSCC. In other words, an interrupt is caused by the first character loaded to the Rx buffer after this mode is set. Once an interrupt occurs, subsequent interrupts in this mode can be enabled by issuing the enable interrupt on next Rx character command.

This mode is generally used for data transmission controlled by the software. It is also used for DMA data transmission. This interrupt can be masked by setting bit D6 of the CR2A register to 1 (7201A only).

All Rx Characters. This mode is used for data transmission by using an interrupt each time a character is loaded to the Rx buffer.

Special Rx Conditions. This interrupt is a special case of Rx interrupt and occurs when either the first Rx character interrupt or all Rx character interrupt is specified. Special Rx conditions refer to parity, overrun, framing, or end of frame errors. When the first Rx character interrupt mode is specified, parity errors are not treated as a special Rx condition. When all Rx character interrupt mode is specified, specify whether or not parity errors are included in the special Rx conditions.

The Tx interrupt occurs when the Tx buffer is empty and Tx interrupts are enabled. This interrupt indicates that the MPSCC is requesting transmit data. The interrupt is satisfied when data is written to the Tx buffer or when the reset Tx INT/DMA pending command is issued. The Tx interrupt also occurs when transmission of CRC characters has been completed in the synchronous or HDLC mode. However, immediately after the MPSCC has been reset (RESET signal or channel reset command), the Tx buffer is empty and the Tx interrupt does not occur even if enabled. Note: If bit D0 of the CR1 register is set to 1 (7201A only) in the HDLC mode, the Tx interrupt will occur if the first data is written to the Tx buffer.

The E/S interrupt checks whether the state of the CTS, DCD, or SYNC pin has changed. This interrupt is also used to detect the Tx underrun error, break state (asynchronous mode), abort sequence, or completion of data transmission (HDLC mode).

DMA. The MPSCC has four DMA request pins. When DMA is enabled by D1, D3, and D4 of the CR1 register, a DMA request occurs as follows:

- DRQTx A, DRQTx B: When the Tx buffer becomes empty. However, the DMA request does not occur if the Tx buffer is empty immediately after it has been reset. This does not apply when bit D6 of the CR1 register is 1 (HDLC mode, 7201A only).
- DRQRx A, DRQRx B: When a character exists in the Rx buffer.

Interrupt and DMA. Bits D1 and D0 of the CR2A register specify whether data transmission is performed by means of an interrupt or DMA. An interrupt or DMA is enabled or disabled by bit D1 (Tx interrupt/DMA) and D4, D3 (Rx interrupt/DMA) of the CR1 register. When DMA is specified to a channel, the Tx interrupt of that channel is inhibited. However, the Rx, special Rx condition, and E/S interrupts can still occur when interrupts are enabled. Normally, the first Rx character interrupt is specified as the Rx interrupt mode of the channel that has been specified for DMA operation. This first Rx character interrupt can be masked by setting bit D6 of the CR2A register to 1 (7201A only). See table 3-2 for details on specifying interrupts or DMA. Figure 3-2 shows interrupt/DMA request generation logic.
Figure 3-2. Interrupt/DMA Request Generation Logic

Notes:
1. In All Rx Character mode: Normally active
   In First Rx Character Interrupt Mode: Active until first Rx
   Character interrupt is generated. However, this is masked
   when D5 of the CR2A register = 1 (7201A only).
2. Rx Disable (CR1: D7)
3. Tx INT/DMA Enable (CR1: D7)
Both channels A and B of the MPSCC have eight control registers (CR0-CR7), two transmit length registers (TxLR: 7201A only), and two or four status registers (SR0 and SR1: 7201; and SR0, SR1, SR3, and SR4: 7201A). Channel B also has an additional status register (SR2B).

The control registers select the required functions for the MPSCC's particular application system. These functions include selecting the following:

- CPU Interface mode
- Serial transmit/receive protocol (asynchronous, bisync, or HDLC)
- MPSCC operation mode used with the selected protocol
- Various parameters
- SYNC character codes

The status registers indicate the internal state of the MPSCC, interrupt status or vectors, and the contents of the transmit length counter (7201A only).

This section describes the function of each register; table 4-1 summarizes the function of each register for both channels.

### Table 4-1. Register Functions

<table>
<thead>
<tr>
<th>Register</th>
<th>Channel</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR0</td>
<td>A, B</td>
<td>Selects a status or control register and resets CRC logic and sets commands (reset of the MPSCC interrupt)</td>
</tr>
<tr>
<td>CR1</td>
<td>A, B</td>
<td>Selects the bus interface mode (data transfer mode between CPU and MPSCC)</td>
</tr>
<tr>
<td>CR2A</td>
<td>A</td>
<td>Specifies system configuration</td>
</tr>
<tr>
<td>CR2B</td>
<td>B</td>
<td>Sets the interrupt vectors</td>
</tr>
<tr>
<td>CR3</td>
<td>A, B</td>
<td>Controls data reception operation</td>
</tr>
<tr>
<td>CR4</td>
<td>A, B</td>
<td>Controls common operation of data transmission and reception</td>
</tr>
<tr>
<td>CR5</td>
<td>A, B</td>
<td>Controls data transmission operation</td>
</tr>
<tr>
<td>CR6</td>
<td>A, B</td>
<td>Specifies SYNC character, flags, and secondary address (HDLC mode)</td>
</tr>
<tr>
<td>CR7</td>
<td>A, B</td>
<td>Specifies the length of transmit data up to 16 bits (7201A only)</td>
</tr>
<tr>
<td>TxLR-L</td>
<td>A, B</td>
<td>Specifies the states of E/S bits, interrupts, or buffers</td>
</tr>
<tr>
<td>TxLR-H</td>
<td>A, B</td>
<td>Sets the special Rx condition or residue codes</td>
</tr>
<tr>
<td>SR0</td>
<td>A, B</td>
<td>Indicates contents of a 16-bit Tx length counter (7201A only)</td>
</tr>
<tr>
<td>SR1</td>
<td>A, B</td>
<td>Selects a status or control register and resets CRC logic code</td>
</tr>
<tr>
<td>SR2</td>
<td>B</td>
<td>Enables, interrupts, etc.</td>
</tr>
<tr>
<td>SR3</td>
<td>A, B</td>
<td>Enables, interrupts, etc.</td>
</tr>
<tr>
<td>SR4</td>
<td>A, B</td>
<td>Selects a status or control register and resets CRC logic code</td>
</tr>
</tbody>
</table>

**CONTROL REGISTER 0 (CR0)**

The functions of the CR0 register bits are summarized in table 4-2.

### Table 4-2. CR0 Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Definition</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>CRC1</td>
<td>CRC initialization code</td>
<td>Initializes CRC logic code</td>
</tr>
<tr>
<td>D6</td>
<td>CRC0</td>
<td>CRC initialization code</td>
<td>Initializes CRC logic code</td>
</tr>
<tr>
<td>D5</td>
<td>CMD2</td>
<td>Command bits</td>
<td>Sets commands used to reset the MPSCC.</td>
</tr>
<tr>
<td>D4</td>
<td>CMD1</td>
<td>Selection of control register</td>
<td>Enables, interrupts, etc.</td>
</tr>
<tr>
<td>D3</td>
<td>CMD0</td>
<td>Selection of control register</td>
<td>Enables, interrupts, etc.</td>
</tr>
<tr>
<td>D2</td>
<td>PTR2</td>
<td>Pointer bits</td>
<td>Selects a status or control register and resets CRC logic code</td>
</tr>
<tr>
<td>D1</td>
<td>PTR1</td>
<td>Pointer bits</td>
<td>Enables, interrupts, etc.</td>
</tr>
<tr>
<td>D0</td>
<td>PTR0</td>
<td>Pointer bits</td>
<td>Enables, interrupts, etc.</td>
</tr>
</tbody>
</table>

**D7 and D6 (CRC Initialization Code)**

Table 4-3 shows the bit pattern and related function for the CRC bits.

### Table 4-3. CRC Bit Functions

<table>
<thead>
<tr>
<th>CR1</th>
<th>CR2</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>No operation</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Initializes Rx CRC calculator</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Initializes Tx CRC calculator</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Resets Rx underrun/EOM bit (D0 of the SR0 register) (See Note)</td>
</tr>
</tbody>
</table>

**Note:** Effective when one character is written and Tx is enabled.

**D5-D3 (Command Bits)**

- **D5 D4 D3 (Send Abort)**
  
  No operation is performed. Specified when the CR0 register is used for purposes other than issuing commands.

- **D5 D4 D3 (Send Abort)**
  
  HDLC abort bits (8 bits, all 1s) are transmitted followed by a flag that causes the CRC circuit of the receiver to reset. 8 to 13 bits are transmitted depending on the number of bits (1s) preceding the abort bits. The contents of the Tx (transmit) buffer are lost when this command is issued.

- **D5 D4 D3 (Reset External/Status Interrupt)**
  
  External/status (E/S) interrupt is re-enabled, and status latch operations to E/S bits (D7-D3 of the SR0 register and D0 of SR1 register) can be performed. Once an E/S interrupt occurs, issue this command to allow other E/S inter-
rupts. Also, issue this command to allow status latch operations to the E/S bits.

\[
\begin{array}{ccc}
D_5 & D_4 & D_3 \\
0 & 1 & 1 & (\text{Channel Reset})
\end{array}
\]

Performs almost the same operation on channel A or B as the external reset. When executed to reset a channel, part of the register contents of the channel is lost. Therefore, it is necessary to rewrite data to the register whose contents have been lost. Issue a new command after the lapse of four system clock cycles when this command has been executed.

Tables 4-4 and 4-5 show the states of the registers and pins after a system reset and after execution of the channel reset command.

**Table 4-4. Register State at Reset**

<table>
<thead>
<tr>
<th>Register</th>
<th>Bits</th>
<th>At System Reset</th>
<th>At Channel Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR2A</td>
<td>0-5, 7</td>
<td>0</td>
<td>See table 4-5</td>
</tr>
<tr>
<td>CR0</td>
<td>0-2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CR1</td>
<td>0, 1, 3, 4, 7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CR3</td>
<td>0-7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CR5</td>
<td>1-4, 7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CR6</td>
<td>2, 6</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>SR0</td>
<td>3-5, 7</td>
<td>Undefined</td>
<td></td>
</tr>
<tr>
<td>SR1</td>
<td>4-7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SR3, SR4</td>
<td>0-7</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Table 4-5. Pin State at Reset**

<table>
<thead>
<tr>
<th>Pin</th>
<th>At System Reset</th>
<th>At Channel Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAITA/DRQRxA</td>
<td>WAITA = Hi-Z</td>
<td>DRQRxA = 0</td>
</tr>
<tr>
<td>WAITB/DRQRxB</td>
<td>WAITB = Hi-Z</td>
<td>DRQTxA = 0</td>
</tr>
<tr>
<td>PRI/DRQRxA</td>
<td>PRI = input</td>
<td>DRQRxB = 0</td>
</tr>
<tr>
<td>PRO/DRQTxB</td>
<td>PRO = by PRI</td>
<td>DROQTxB = 0</td>
</tr>
<tr>
<td>DTRB/HAI</td>
<td>DTRB = 1</td>
<td>HA1 = input</td>
</tr>
<tr>
<td>DTRB/HAO</td>
<td>DTRB = 1</td>
<td>HA0 = HAI</td>
</tr>
<tr>
<td>INT</td>
<td>Hi-Z</td>
<td></td>
</tr>
<tr>
<td>RTS/RTSb</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>SYNCB</td>
<td>Input state</td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{array}{ccc}
D_5 & D_4 & D_3 \\
0 & 0 & 0 & (\text{Enable Interrupt on Next Receive Character})
\end{array}
\]

If an interrupt occurs in the first Rx character mode (first character has been received), this command causes another interrupt with the first character after the command is issued. Prepares for new messages after receiving a series of messages.

\[
\begin{array}{ccc}
D_5 & D_4 & D_3 \\
1 & 0 & 1 & (\text{Reset Transmitter Interrupt/DMA Pending})
\end{array}
\]

When the Tx buffer becomes empty, a Tx interrupt/DMA request is generated to request that new data be written to the Tx buffer. In some instances (when a message ends) this request may not be acknowledged. Issue this command to clear the interrupt/DMA request inside the MPSCC.

\[
\begin{array}{ccc}
D_5 & D_4 & D_3 \\
1 & 1 & 0 & (\text{Error Reset})
\end{array}
\]

Resets an error bit and the EOF bits (D4-D7 of the SR1 register). If the special Rx condition interrupt occurs in the first Rx character interrupt mode, the characters loaded to the Rx buffer following the character that caused the interrupt will not be assembled until this command is issued.

\[
\begin{array}{ccc}
D_5 & D_4 & D_3 \\
1 & 1 & 1 & (\text{End of Interrupt})
\end{array}
\]

Resets the in service latch that has the higher priority among the in service latches which are set. Set on channel A only. If there is an interrupt request in service whose process currently is terminated, interrupt request processing will be resumed unless INTR is generated to request an interrupt assigned a higher priority than the interrupt in service. In this manner, the end of interrupt command enables an interrupt with lower priority in the daisy chain. Priority status will be reanalyzed to determine which interrupt takes precedence, if an INTAK sequence occurs from the interrupt with a priority higher than the one in service.

**D2 - D0 (Pointer Bits)**

These bits specify the register to be used for the next read or write operation. Generally, each register is accessed by the MPSCC according to this procedure. The pointer indicates \texttt{000} after an external reset or a channel reset command.

**Pointer Specification.** The pointer bits specify a binary register number used to perform a read or write operation.

**Read/Write Operation.** The register specified as a pointer will be accessed if the read or write operation is performed when C/D is 1.
**Pointer Restoration.** The pointer bits return to 000 after the read or write operation has been performed. Therefore, do not respecify the pointer bits when the CR0 or SR0 register is accessed.

**Example 1 (Setting CR2A).** The pointer bits of the CR0A register specify 2 \((D_2 - D_0 = 010)\). The CR2A register inputs data written during the next write to the MPSCC, for instance, CS = 0, C/D = 1.

**Example 2 (Reading SR3B).** The pointer bits of the CR0B register specify 3 \((D_2 - D_0 = 011)\). The SR3B register outputs the contents of the Tx length register.

The Tx length registers \((TxLR-L, TxLR-H; 7501A only)\) use a special data setting method. The values of these registers are set (without specifying the pointer) by a series of two write cycles that follow writing a 1 to bit D6 of the CR1 register.

**Example 3 (Setting TxLR of Channel A).** The pointer bits of the CR0A register specify 1 \((D_2 - D_0 = 001)\). Bit D6 of the CR1A register is set to 1 and other necessary specifications in this register are then performed. The number of transmit data characters is set in TxLR-L and TxLR-H by the successive write cycles of CRxA.

**CONTROL REGISTER 1 (CR1)**

The functions of the CR1 register bits are summarized in table 4-6.

**Table 4-6. CR1 Bit Functions**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Wait Enable</td>
<td>Controls operation of WAIT pin</td>
</tr>
<tr>
<td>D6</td>
<td>Tx Length Register Set</td>
<td>Instructs data setting of Tx length registers (7201A only)</td>
</tr>
<tr>
<td>D5</td>
<td>Wait on Rx/Tx</td>
<td>Controls operation of WAIT pin</td>
</tr>
<tr>
<td>D4</td>
<td>Rx INT Mode 1</td>
<td>Selects the mode for Rx interrupts</td>
</tr>
<tr>
<td>D3</td>
<td>Rx INT Mode 0</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>Status Affects Vector</td>
<td>Modifies the interrupt vector caused by an interrupt</td>
</tr>
<tr>
<td>D1</td>
<td>Tx INT/DMA Enable</td>
<td>Enables the Tx interrupt/DMA</td>
</tr>
<tr>
<td>D0</td>
<td>E/S Interrupt Enable</td>
<td>Enables the E/S interrupt</td>
</tr>
</tbody>
</table>

**D7 (Wait Enable) and D5 (Wait on Rx/Tx)**

These bits are meaningful only when both channels A and B are set in the interrupt mode \((D_0 = 0)\). When D5 is 1, WAIT becomes active (goes low) when an attempt is made to read receive (Rx) data when the MPSCC is not in the Rx character available state; that is, when the MPSCC has not received data. WAIT becomes inactive (high impedance) when the MPSCC enters the Rx character available state. When D5 is 0, WAIT becomes active when data is written to the MPSCC when the Tx buffer is full. When the Tx buffer becomes empty, WAIT becomes inactive. Table 4-7 shows the state of bits D7 and D5 during receive and transmit operations.

**Table 4-7. Bits D7 and D5 of CR1**

<table>
<thead>
<tr>
<th>Function</th>
<th>D7</th>
<th>D5</th>
<th>Rx/Tx Operation</th>
<th>Condition</th>
<th>WAIT Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>Hi-Z</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>—</td>
<td>Tx buffer</td>
<td>When data is written to the Tx buffer while it is full</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>When the Tx buffer is or becomes empty</td>
<td>Hi-Z</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>—</td>
<td>Rx buffer</td>
<td>When data is read from the Rx buffer when it is empty</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>When the Rx buffer is or becomes full</td>
<td>Hi-Z</td>
</tr>
</tbody>
</table>

Note: x = Don’t care

**D6 (Tx Length Register Set)**

This bit is used in the \(\mu PD7201A\) only. It is used to program the number of transmit data characters in the Tx length registers \((TxLR-L\) and \(TxLR-H)\). The 16-bit value sent to the \(\mu PD7201A\) after this bit is set to 1 will be loaded into the Tx length registers. The first write to the chip \((WR = 0)\) with the C/D line equal to 1 will load the lower eight bits of the Tx length from the data bus into the TxLR-L register. The next command write will load the high-order byte into TxLR-H. When transmission is begun after this mode has been set, the transmission length counter will be incremented each time the INT or DRQTx lines become active. When this counter value equals the value loaded into the TxLR, the Tx interrupt or DRQ request will be masked. When a DMA request or interrupt is masked, the value of the counter will be set to 0.

The Tx length registers can be rewritten by again setting bit D6 of CR1. If the TxLR is rewritten, the interrupt or DMA request masking is reset. Transmission should not be enabled \((D_2 \text{ of CR5 should not be set})\) until the TxLR is programmed.

During HDLC transmission in this mode, a transmission underrun condition will cause the transmission of the CRC bytes if the value in the Tx length counter equals the value in the TxLR. If these values are not equal at the Tx under-run, then an abort sequence is automatically transmitted.

Once this bit has been set, a channel reset command or an external reset must be given to reset this mode. Writing a 0 to D6 in CR1 will not reset it.

**D4 and D3 (Rx Interrupt Modes 0-3)**

These bits determine the mode of an interrupt caused by character reception. In modes 1-3, the special receive condition interrupt occurs. When the DMA mode is selected by
INT DMA mode bits (D0, D1 of the CR2A register), and when the Rx interrupt mode bits are set in any mode other than mode 0, DMA requests are generated on the Rx character available condition. Table 4-8 shows the function of bits D4 and D3 in Rx interrupt mode.

Table 4-8. Bits D4 and D3 of CR1

<table>
<thead>
<tr>
<th>Mode</th>
<th>D4</th>
<th>D3</th>
<th>Interrupt Mode</th>
<th>Parity Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Rx INT DMA disabled</td>
<td>—</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>First character interrupt</td>
<td>Excluded from Special Rx Condition</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>All character interrupt</td>
<td>Included in Special Rx Condition</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>All character interrupt</td>
<td>Excluded from Special Rx Condition</td>
</tr>
</tbody>
</table>

D2 (Status Affects Vector)
This bit is set only at channel B. If a vectored mode (D5 of the CR2A register = 1) is specified, an interrupt vector (vector bits V7 - V0) is output to the data bus in the interrupt acknowledge cycle. The status affects vector bit is used to determine whether the contents of an interrupt vector should be modified according to the interrupt source.

When this bit is 0, the interrupt vector set at the CR2B register is output as it is. When the bit is 1, Vector bits V4-V2 of the interrupt vector set at the CR2B register are modified if 85 mode is specified. In 86 mode, vector bits V2-V0 are modified as shown in table 4-9.

Table 4-9. Modification of Vector Bits V2-V0 or V4-V2

<table>
<thead>
<tr>
<th>Interrupt Cause</th>
<th>Channel</th>
<th>V2</th>
<th>V1</th>
<th>V0</th>
<th>85 mode</th>
<th>V2</th>
<th>V1</th>
<th>V0</th>
<th>86 mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>A</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>B</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>B</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>B</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>B</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

D1 (Tx Interrupt/DMA Enable)
This bit enables the Tx interrupt or DMA. When D1 is set and when the first Tx data byte is written to the MPS CC, an interrupt or DMA takes place each time the Tx buffer becomes empty. However, when bit D5 of the CR1 register is 1, an interrupt/DMA request will be generated each time the Tx buffer becomes empty even if the first Tx data byte is not yet written to the MPS CC (7201A only).

D0 (External/Status Interrupt Enable)
This bit enables the E/S interrupt. The E/S interrupt is enabled when the following occurs:
- The level of an input signal to the CTS, DCD, or SYNC pin changes.
- The start or end of a break condition or abort sequence is detected.
- The transmission of CRC, SYNC, or flag characters starts in the Tx Underrun condition.
- The transmission of a frame has been completed in HDLC mode (7201A only).

CONTROL REGISTER 2, CHANNEL A (CR2A)
The functions of the CR2A register bits are summarized in table 4-10.

Table 4-10. CR2A Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>07</td>
<td>RTSB/SYNC Select</td>
<td>Selects the function of the RTSB/SYNCB pin</td>
</tr>
<tr>
<td>06</td>
<td>Rx INT Mask</td>
<td>Masks an interrupt caused by a first Rx character (7201A only)</td>
</tr>
<tr>
<td>05</td>
<td>Vector Mode</td>
<td>Selects nonvectored or vectored mode</td>
</tr>
<tr>
<td>04</td>
<td>INT Mode</td>
<td>Selects interrupt mode 85-1, 85-2, 85-3, or 86</td>
</tr>
<tr>
<td>03</td>
<td>INT/OMA Mode</td>
<td>Specifies the interrupt or DMA mode of each channel</td>
</tr>
<tr>
<td>02</td>
<td>(RTSB/SYNCB Select)</td>
<td>Specifies the function of the RTSB/SYNCB pin when 0, selects the RTSB function or 1, selects the SYNCB pin.</td>
</tr>
</tbody>
</table>

D7 (RTSB/SYNCB Select)
This bit specifies the function of the RTSB/SYNCB pin. When 0, selects the RTSB function; when 1, selects the SYNCB pin.

D6 (Rx INT Mask)
If bits D4 and D5 of the CR1 register are 0 and 1 (INT on first character mode), an interrupt caused by the first received character is masked when D6 is set (7201A only). While D6 is set, the Rx INT signal will not become active even if the enable INT on next Rx character command is issued. When D6 = 0, no interrupt is masked. This bit must be zero on the μPD7201.

D5 (Vector Mode)
This bit specifies the nonvectored or vectored mode. When 0, the nonvectored mode is selected. Nonvectored mode is used for the CPU that cannot read interrupt vectors in the interrupt acknowledge cycle. When 1, vectored mode is specified. In vectored mode, when the MPS CC responds...
to the state of the interrupt mode bits, PRI and an internal interrupt source, the MPSCC outputs either a CALL code or an interrupt vector to the data bus when the INTAK pin receives an active signal. If the CALL or an interrupt vector is not output, the data bus becomes high impedance.

D4 and D3 (Interrupt Mode)
These bits selects the mode in which the MPSCC responds when it receives the INTAK signal. Also, they determine the vector bits (V7 - V0 of the SR2B register) of an interrupt vector to be modified in status affects vector mode. Table 4-11 shows the states of bits D4 and D3 in the interrupt modes.

Table 4-11. Bits D4 and D3 of CR2A

<table>
<thead>
<tr>
<th>D4</th>
<th>D3</th>
<th>Mode</th>
<th>Interrupt Vector Bits for Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td></td>
<td>85-1</td>
<td>V4 V3 V2</td>
</tr>
<tr>
<td>0 1</td>
<td></td>
<td>85-2</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td></td>
<td>85-3 (7201A only)</td>
<td>V2 V1 V0</td>
</tr>
</tbody>
</table>

D2 (Priority Select)
This bit determines the priority order of TxA (channel A) and RxB (channel B) when an interrupt or DMA occurs inside the MPSCC. The priority of the other interrupt sources, such as an E/S interrupt, is fixed (see table 4-12).

D1 and D0 (INT/DMA Modes 1 and 0)
These bits specify an interrupt or DMA mode for each channel. The pin functions necessary for each system configuration (DMA request generation and recognition performed by the DRQRxR, DRQTxA, or RA pin, and MPSCC daisy chain performed by the HA/RA, PR/L, or PRD pin) is selected and defined by D1 and D0 (see Section 6). When both D1 and D0 = 1, there is no priority for DMA operations (7201A only). The priority of the special Rx condition is the same for each channel.

In Table 4-12, when bit D1 or D0 is 1, channel A is set in the DMA mode, and a DMA request is generated when the MPSCC is in the Rx character available state (DRQRxA specified) or when the Tx buffer is empty (DRQTxA specified). Channel B is set in the interrupt mode. It is also possible that the E/S interrupt or Rx interrupt occurs at channel A. Usually in this case, the INT or first character bit is selected when channel A is set in the Rx interrupt mode. Consequently, the first received character will cause an interrupt. In the Rx interrupt mode, bit D2 can be set to either 0 or 1 without causing any change in interrupt or DMA priority.

Table 4-12. INT/DMA Priorities

<table>
<thead>
<tr>
<th>CR2A</th>
<th>INT/DMA Mode</th>
<th>Pin Function</th>
<th>DTRB/RA</th>
<th>High</th>
<th>Priority</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>D2</td>
<td>D1</td>
<td>D0</td>
<td>Channel A</td>
<td>Channel B</td>
<td>WAITE/DRQRxA</td>
<td>WAITB/DRQTxB</td>
</tr>
<tr>
<td>0 0 0</td>
<td>INT</td>
<td>INT</td>
<td>WAITA</td>
<td>WAITB</td>
<td>PRI</td>
<td>PRO</td>
</tr>
<tr>
<td>1 0 0</td>
<td>INT</td>
<td>INT</td>
<td>RxA</td>
<td>RxB</td>
<td>TxA</td>
<td>TxB</td>
</tr>
<tr>
<td>x 0 1</td>
<td>DMA</td>
<td>INT</td>
<td>DRQRxA</td>
<td>DRQTxA</td>
<td>PRI</td>
<td>PRO</td>
</tr>
<tr>
<td>0 1 0</td>
<td>DMA</td>
<td>DMA</td>
<td>DRQRxA</td>
<td>DRQTxA</td>
<td>DRQRxB</td>
<td>DRQTxB</td>
</tr>
<tr>
<td>1 1 0</td>
<td>DMA</td>
<td>DMA</td>
<td>RxA</td>
<td>RxB</td>
<td>TxA</td>
<td>TxB</td>
</tr>
<tr>
<td>x 1 1</td>
<td>DMA</td>
<td>DMA</td>
<td>RxA*</td>
<td>RxB</td>
<td>TxA</td>
<td>TxB</td>
</tr>
</tbody>
</table>

Table 4-12 shows the states of bits D4 and D3 in the interrupt modes.
CONTROL REGISTER 2, CHANNEL B (CR2B)

The functions of the CR2B register bits are summarized in Table 4-13.

Table 4-13. CR2B Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>V7</td>
<td>Interrupt Vector Set interrupt vectors sent in the interrupt acknowledge cycle. When Status Affects Vector bit = 0, vector bits V7 - V9 are output. When Status Affects Vector bit = 1, Vector Bits V7, V8, and V1, V9 are modified. Vector Bits V4 - V2 are output in 85 mode. Vector bits V2 - V0 are output in 86 mode.</td>
</tr>
<tr>
<td>D6</td>
<td>V6</td>
<td></td>
</tr>
<tr>
<td>D5</td>
<td>V5</td>
<td></td>
</tr>
<tr>
<td>D4</td>
<td>V4</td>
<td></td>
</tr>
<tr>
<td>D3</td>
<td>V3</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>V1</td>
<td></td>
</tr>
<tr>
<td>D0</td>
<td>V0</td>
<td></td>
</tr>
</tbody>
</table>

CONTROL REGISTER 3 (CR3)

The functions of the CR3 register bits are summarized in Table 4-14.

Table 4-14. CR3 Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Rx Bit/Character 1</td>
<td>Define number of bits per serial data character.</td>
</tr>
<tr>
<td>D6</td>
<td>Rx Bit/Character 0</td>
<td></td>
</tr>
<tr>
<td>D5</td>
<td>Auto Enable</td>
<td>Controls transmission/reception operation by input of a CTS or DCD signal.</td>
</tr>
<tr>
<td>D4</td>
<td>Enter Hunt Phase</td>
<td>Sets the Hunt phase.</td>
</tr>
<tr>
<td>D3</td>
<td>Rx CRC Enable</td>
<td>Starts (or restarts) the CRC calculation operation.</td>
</tr>
<tr>
<td>D2</td>
<td>Address Search Mode</td>
<td>Receives a message if the receive address matches either that of CR6 or a global address.</td>
</tr>
<tr>
<td>D1</td>
<td>SYNC CHAR Load Inhibit</td>
<td>Inhibits SYNC characters from being loaded to the Rx buffer.</td>
</tr>
<tr>
<td>D0</td>
<td>Rx Enable</td>
<td>Starts reception operation.</td>
</tr>
</tbody>
</table>

D7 and D6 (Rx Bits/Characters 1 and 0)

These bits define the number of bits per character of serial data to be received. Although the number of bits in a character can be modified during serial data reception, any modification must be done before the character reaches the MPSCC. Table 4-15 shows the states of bits D7 and D6 and how they define the received character.

Table 4-15. Bits D7 and D6 of CR3

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>Bits/Character After Assembly</th>
<th>8-bit Character</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1/P[3][4][3][3]</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1/P[3][4][3][3]</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1/P[3][4][3][3]</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1/P[3][4][3][3]</td>
</tr>
</tbody>
</table>

Notes:
[1] 1/P is 1 when the parity enable bit (D0 of the CR4 register) is 0. Used for parity when the parity enable bit is 1.
[2] x = the unused bit(s) of the received character.
[3] x = the unused bit in the parity disable state (D0 of the CR4 register = 0). Used as a parity bit in the parity enable state (D0 of the CR4 register = 1).
**D₅ (Auto Enable)**

This bit enables or disables data transmission (Tx) or reception (Rx) in response to the modem signal input to either the CTS or DCD pin. When the auto enable bit is 1 and the Tx or Rx enable bit is 1, the transmitter or receiver will be enabled while the CTS or DCD signal is active. The Tx or Rx operation is disabled when the CTS or DCD signal becomes inactive or when the Tx/Rx enable bit is 0. When the auto enable bit is 0, the CTS and DCD inputs are treated only as general-purpose inputs. In other words, input of the two signals has no effect on the Tx or Rx operation; it only causes the status bits (D₅ and D₃ of the SR0 register) to be set.

**D₄ (Enter Hunt Phase)**

This bit directs the MPSCC to enter the hunt phase when the synchronous or HDLC mode is specified. The hunt operation occurs when the MPSCC enters the Rx enable state after the enter hunt phase bit is set. The MPSCC goes out of the hunt phase when it is synchronized and detects the next state.

When the enter hunt phase bit is set in the internal synchronous or HDLC mode, the SYNC/hunt bit (D₄ of the SR0 register) also is set. When the MPSCC is reset, it enters the hunt phase after it has been released from the Rx enable state. For this reason, the enter hunt phase bit need not be set. This bit also enables the MPSCC to reenter the hunt phase if for some reason established character synchronization is lost in the synchronous mode, or if an input frame is unnecessary in the HDLC mode.

**D₃ (Receive CRC Enable)**

This bit directs the start or restart of CRC calculation (1 = start). This bit must be changed before the character following the one involved in the CRC calculation is loaded into the Rx buffer. If there is no character stored in the Rx buffer, check the Rx character available bit to find when to set the bit.

**D₂ (Address Search Mode)**

This bit is valid only in the HDLC mode. When the address search mode bit is set in the HDLC receive mode, a received secondary address (the first character following the flag) is compared with either the secondary reference address in the CR6 register or the global address 11111111. If no match is made, the Rx interrupt is prevented, the message following the received secondary address will not be accepted, and the character will not be assembled. This mode is effective in systems using more than one secondary MPSCC and when a secondary MPSCC selectively receives transmitted messages.

**D₁ (SYNC Character Load Inhibit)**

This bit prevents SYNC characters from being loaded into the Rx buffer. This mode must normally be reset after the MPSCC detects a SYNC character. This prevents the mode from being applied to SYNC characters inserted in a message. In this mode, CRC calculation is also performed on SYNC characters that have not been loaded into the Rx buffer. This bit remains valid even after the MPSCC has exited the hunt phase.

**D₀ (Rx Enable)**

This bit enables start of the receive operation. The Rx enable bit is set after receive parameters are set and the receiver is initialized.

**CONTROL REGISTER 4 (CR4)**

The functions of the CR4 register bits are summarized in table 4-16.
Table 4-16. CR4 Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Clock Rate 1</td>
<td>Specify the clock rate. Must be the data rate times 1, 16, 32, or 64</td>
</tr>
<tr>
<td>D6</td>
<td>Clock Rate 0</td>
<td></td>
</tr>
<tr>
<td>D5</td>
<td>SYNC Mode 1</td>
<td>Select the character synchronization method</td>
</tr>
<tr>
<td>D4</td>
<td>SYNC Mode 0</td>
<td></td>
</tr>
<tr>
<td>D3</td>
<td>Stop Bit 1</td>
<td>Select the transmit stop bit length</td>
</tr>
<tr>
<td>D2</td>
<td>Stop Bit 0</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>Parity Even/Odd</td>
<td>Selects even or odd parity</td>
</tr>
<tr>
<td>D0</td>
<td>Parity Enable</td>
<td>Enables transmission or checks parity bit</td>
</tr>
</tbody>
</table>

D7 and D6 (Clock Rates 1 and 0)

These bits specify the multiple by which the clock rate (Rx and Tx) should be divided to obtain the desired data transmission rate (asynchronous mode). In the synchronous mode, a clock rate of 1 must be used. Table 4-17 shows how the D7 and D6 bits select the clock rate.

Table 4-17. Bits D7 and D6 of CR4

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>Clock Rate (Rx, Tx)</th>
<th>System Clock Rate (CLK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>(Data rate) × 1</td>
<td>(Data Rate) × 4.5 minimum</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>(Data rate) × 16</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>(Data rate) × 32</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>(Data rate) × 64</td>
<td></td>
</tr>
</tbody>
</table>

Note: When clock rate × 1 is specified, bit synchronization must be established externally.

D5 and D4 (SYNC Modes 1 and 0)

These bits select the character synchronization method and the SYNC character length. Set these bits to 00 in the asynchronous mode and 10 in the HDLC mode. Table 4-18 shows how D5 and D4 specify the synchronous mode and character.

Table 4-18. Bits D5 and D4 of CR4

<table>
<thead>
<tr>
<th>D5</th>
<th>D4</th>
<th>Synchronous Mode</th>
<th>Synchronization Character</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Monosync</td>
<td>8-bit SYNC</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Bisync</td>
<td>16-bit SYNC</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>HDLC</td>
<td>Flag pattern (01111110)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>External Sync</td>
<td>None</td>
</tr>
</tbody>
</table>

D3 and D2 (Stop Bits 1 and 0)

These bits specify the stop bit length for transmission in the asynchronous transmit mode. In the receive mode, a stop bit check is performed assuming that the length of a stop bit is 1. Set these bits to 00 in the synchronous mode (including HDLC mode). Table 4-19 shows how bits D3 and D2 specify the stop bit length.

Table 4-19. Bits D3 and D2 of CR4

<table>
<thead>
<tr>
<th>D3</th>
<th>D2</th>
<th>Stop Bit Length (Bit/Char)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Synchronous mode (including HDLC)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1½</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

D1 (Parity Even/Odd)

This bit selects even or odd parity and is valid while the parity enable bit is set. When the parity even/odd bit is 1, even parity is selected. When it is 0, odd parity is selected.

D0 (Parity Enable)

Setting this bit to 1 enables transmission of a parity bit and allows checks of parity error. In the Tx mode, a parity bit is added to the data bits of each character and the character is transmitted. The number of bits making up a character is specified by bits D5 and D4 of the CR5 register. In the Rx mode, the MPSCC performs a parity check, assuming that the data bits for each character are followed by a parity bit. Also, the number of data bits making up a character is determined by bits D7 and D6 of the CR3 register. If a parity error occurs, the error bit is set.

Note that the parity bit is treated as part of a character and transmitted that way to the CPU. The parity bit, however, is not included in a character if the character is already 8 bits long.

CONTROL REGISTER 5 (CR5)

The functions of the CR5 register bits are summarized in Table 4-20.

Table 4-20. CR5 Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Data Terminal Ready</td>
<td>Controls the DTR pin</td>
</tr>
<tr>
<td>D6</td>
<td>Tx Bits/Character 1</td>
<td>Define the number of bits of the serial transmit data character</td>
</tr>
<tr>
<td>D5</td>
<td>Tx Bits/Character 0</td>
<td></td>
</tr>
<tr>
<td>D4</td>
<td>Send Break</td>
<td>Transmits a break sequence</td>
</tr>
<tr>
<td>D3</td>
<td>Tx Enable</td>
<td>Starts the Tx operation</td>
</tr>
<tr>
<td>D2</td>
<td>CRC-16/CCITT</td>
<td>Controls the CRC polynomial</td>
</tr>
<tr>
<td>D1</td>
<td>Request to Send</td>
<td>Controls the RTS pin</td>
</tr>
<tr>
<td>D0</td>
<td>Tx CRC Enable</td>
<td>Enables the CRC calculation or transmission</td>
</tr>
</tbody>
</table>
D7 (Data Terminal Ready)
This bit controls the DTR pin. When the DTRA/HAO or DTRB/HAI pin is used as the DTR pin (D1 and D0 of the CR2A register = 00), the state of this bit is inverted as follows:

<table>
<thead>
<tr>
<th>DTR bit</th>
<th>DTR pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

D6 and D5 (Transmit Bits/Character 1 and 0)
These bits define the number of bits per character of each serial transmit data byte. Bits D7 - D0 are valid transmit data bits. Table 4-21 shows how D6 and D7 specify the number of bits per character and the data byte written to the MPSCC.

Table 4-21. Bits D6 and D5 of CR5

<table>
<thead>
<tr>
<th>D6</th>
<th>D5</th>
<th>Number of bits/character</th>
<th>Data byte written to the MPSCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1 1 1 1 0 0 0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1 1 1 0 0 0 D1 0 D0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1 0 0 0 0 D3 D2 D1 0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td>0 0 0 0 D4 D3 D2 D1 0</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>5</td>
<td>x x D5 D4 D3 D2 D1 0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>6</td>
<td>x D6 D5 D4 D3 D2 D1 0</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>7</td>
<td>D7 D6 D5 D4 D3 D2 D1 0</td>
</tr>
</tbody>
</table>

D4 (Send Break)
When D4 is a 1, the output signal from the TxD pin will be set immediately in the space state (0), and the contents of the Tx buffer will be lost. When 0, the output signal from the TxD pin will return to the mark state (1). However, if this bit remains 1 for a shorter period of time than that required for transmission of one character, the output from the TxD pin will not be in the mark state even after this bit has been reset to 0. In this case, part of the internal transmit data will be transmitted. Therefore, the time during which this bit is set to 1 must be long enough for the specified one character transmission rate.

D3 (Transmit Enable)
When D3 is a 1, it enables the start of data transmission. The auto enable mode (see D5, auto enable in this section) makes it possible to enable the transmitter after the CTS pin becomes active. The transmitter is disabled when the transmit enable bit is 0. This sets the signal output from the TxD pin in the mark state. If the transmit enable bit becomes 0 while data is being transmitted, the TxD pin will return to the mark state as soon as all transmit characters (such as data, SYNC character, and flag) have been transmitted. If a CRC character is transmitted when the transmit enable bit is 0, it will be replaced by either a SYNC character or a flag.

D2 (CRC-16/CCITT)
This bit selects the CRC polynomial to be used by the Tx and Rx CRC calculator as shown in table 4-22.

Table 4-22. CRC Polynomial

<table>
<thead>
<tr>
<th>D2</th>
<th>Polynomial</th>
<th>Operation Mode</th>
<th>Preset Data (CRC: D7, D6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CRC-16 (x16 + x15 + x2 + 1)</td>
<td>Synchronous</td>
<td>All 0s</td>
</tr>
<tr>
<td>0</td>
<td>CCITT-0 (x16 + x12 + x5 + 1)</td>
<td>Synchronous</td>
<td>All 0s</td>
</tr>
<tr>
<td></td>
<td>CCITT-1 (x16 + x12 + x5 + 1)</td>
<td>HDLC</td>
<td>All 1s</td>
</tr>
</tbody>
</table>

D1 (Request to Send)
D1 controls the RTS pin as shown in table 4-23.

Table 4-23. RTS Pin Control

<table>
<thead>
<tr>
<th>Mode</th>
<th>D1</th>
<th>RTS Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asynchronous</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Becomes 1 when all characters have been transmitted and the Tx buffer is empty.</td>
</tr>
<tr>
<td>Synchronous</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>HDLC</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

D0 (Transmit CRC Enable)
D0 enables both CRC calculation and CRC character transmission. Normally, it must be set before the first data (bisync mode) or address field (HDLC mode) is transmitted to the MPSCC. The transmit CRC enable bit becomes valid when a character is transferred to the Tx shift register from the Tx buffer. Therefore, when the CRC calculation is enabled or disabled during message transmission, this bit must be set or reset before the data character involved in the CRC calculation is transmitted.

If the Tx underrun condition occurs when the transmit CRC enable bit is set and when the Tx underrun/EOM bit is reset, a CRC character will be transmitted. If the Tx underrun condition occurs when the transmit CRC enable bit is reset, no CRC character will be transmitted, and a SYNC character or flag will be transmitted instead.

CONTROL REGISTERS 6 AND 7 (CR6 and CR7)
The functions of the CR6 and CR7 register bits are summarized in table 4-24.
### Table 4-24. CR6 and CR7 Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>CR6 Function</th>
<th>CR7 Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Sync Bit 7</td>
<td>Sync Bit 15</td>
</tr>
<tr>
<td>D6</td>
<td>Sync Bit 5</td>
<td>Sync Bit 14</td>
</tr>
<tr>
<td>D5</td>
<td>Sync Bit 5</td>
<td>Sync Bit 13</td>
</tr>
<tr>
<td>D4</td>
<td>Sync Bit 4</td>
<td>Sync Bit 12</td>
</tr>
<tr>
<td>D3</td>
<td>Sync Bit 3</td>
<td>Sync Bit 11</td>
</tr>
<tr>
<td>D2</td>
<td>Sync Bit 2</td>
<td>Sync Bit 10</td>
</tr>
<tr>
<td>D1</td>
<td>Sync Bit 1</td>
<td>Sync Bit 9</td>
</tr>
<tr>
<td>D0</td>
<td>Sync Bit 0</td>
<td>Sync Bit 8</td>
</tr>
</tbody>
</table>

These characters are written to the CR6 and CR7 registers according to the specified mode shown in table 4-25.

### Table 4-25. CR6 and CR7 Characters

<table>
<thead>
<tr>
<th>Mode</th>
<th>CR6 Character</th>
<th>CR7 Character</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monosync</td>
<td>Tx Sync character</td>
<td>Rx Sync character</td>
</tr>
<tr>
<td>Bisync</td>
<td>Sync character 1</td>
<td>Sync character 2</td>
</tr>
<tr>
<td>(First 8 of 16 bits)</td>
<td>(Last 8 of 16 bits)</td>
<td></td>
</tr>
<tr>
<td>EXT Sync</td>
<td>Tx Sync character</td>
<td>Not used</td>
</tr>
<tr>
<td>HDLC</td>
<td>Secondary address</td>
<td>Flag character (01111110)</td>
</tr>
<tr>
<td></td>
<td>(For comparison with an address field)</td>
<td></td>
</tr>
</tbody>
</table>

### STATUS REGISTER 0 (SR0)

The functions of the SR0 register bits are summarized in table 4-26.

### Table 4-26. SR0 Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Break/Abort</td>
<td>Detects the break or abort sequence.</td>
</tr>
<tr>
<td>D6</td>
<td>Tx Underrun/EDM</td>
<td>Detects the Tx Underrun condition.</td>
</tr>
<tr>
<td>D5</td>
<td>E/S bits</td>
<td>Monitors input to the E/S pins.</td>
</tr>
<tr>
<td>D4</td>
<td>CTS</td>
<td>Monitors input to the CTS pin.</td>
</tr>
<tr>
<td>D3</td>
<td>SYNC-Hunt</td>
<td>Monitors input to the SYNC pin or the Hunt phase.</td>
</tr>
<tr>
<td>D2</td>
<td>DCD</td>
<td>Monitors input to the DCD pin.</td>
</tr>
<tr>
<td>D1</td>
<td>Tx Buffer Empty</td>
<td>Checks for data in the Tx buffer.</td>
</tr>
<tr>
<td>D0</td>
<td>Interrupt Pending</td>
<td>Checks for an interrupt whose process has not been completed.</td>
</tr>
<tr>
<td></td>
<td>Rx Character Available</td>
<td>Checks for data in the Rx buffer.</td>
</tr>
</tbody>
</table>

### D7 - D3 (E/S Bits)

These bits indicate (or latch) the states of the modem control pins and data transmission/reception.

The E/S bits latch states in a special manner. When one satisfies a predetermined condition, such as inversion of a bit, each bit latches the respective state at that point. The E/S interrupt then occurs. Once a latch occurs, another latch operation will not be performed until the reset E/S INT command is issued, even if the condition of each bit required to perform another latch operation is satisfied.

**Figure 4-1. E/S Bit Operation**

---

**D7 (Break/Abort)**

This bit indicates whether the MPSCC has received a break sequence in the asynchronous mode or an abort sequence in the HDLC mode. If the MPSCC receives a break or abort sequence during latch operation of the E/S bits, the break/abort bit becomes 1 and latches that state as long as the sequence is received. If the MPSCC does not receive a break or abort sequence, the break/abort bit remains 0. In other words, this bit becomes 1 when a break or abort sequence is detected among receive data and remains 0 when either the MPSCC detects the end of the break or abort sequence or when an E/S interrupt occurs. The reset E/S INT command is normally issued (D7 = 0 or 1) immediately after the E/S interrupt occurs, enabling detection of either the end of the break or abort sequence or of the next break or abort sequence.
After the MPSCC detects the end of a break sequence, a null character (all 0s) received during the break sequence and consisting of one or more bytes will remain in the Rx buffer. Therefore, ignore the first data received immediately after the MPSCC has detected an end of the break sequence.

The break/abort bit is not used in synchronous mode.

D₆ (Transmit Underrun/End of Message)
This bit becomes 1 if the MPSCC is in the Tx underrun condition during latch operation of the E/S bits. If the MPSCC is not in the Tx underrun condition, this bit is 0. However, the transmit underrun/end of message bit also becomes 1 when the MPSCC is reset by either the channel reset command or the RESET signal and can be set to 0 only when the Tx underrun/EOM command is issued by setting bits D₂ and D₆ of the CR0 register to 11. Bit D₆ is set to 0 when the first data before the flag pattern is written into the MPSCC in HDLC mode. However, the Tx underrun/EOM bit will become 1 and the E/S interrupt will occur if the MPSCC enters the Tx underrun condition when CRC character transmission is enabled. Therefore, the E/S interrupt will not occur when the reset Tx underrun/EOM command is issued (Tx underrun/EOM bit changes from 1 to 0). Additionally, latch operation of the other E/S bits will not occur. The Tx underrun/EOM command affects the operation of the MPSCC when it detects the end of a message, just as the Tx buffer empty bit does. See Section 5 for a description of operations in the synchronous and HDLC modes.

D₅ (Clear to Send)
This bit inverts the state of the CTS pin during a latch operation. When the clear to send bit is 1, the CTS pin is 0; when clear to send is 0, the CTS pin is 1. At the leading (0 to 1) or trailing (1 to 0) edge of the input signal to the CTS pin, the clear to send bit becomes 1 or 0 and the E/S interrupt occurs.

D₄ (SYNC/Hunt)
The function of this bit varies depending on the specified operating mode. In the asynchronous or external synchronous modes, the SYNC/hunt bit indicates inversion of the SYNC pin's signal level during latch operation of the E/S bits. That is, if this bit is 1, it indicates the SYNC pin is 0; if the bit is 0, the SYNC pin is 1. At the trailing edge (1 to 0) and leading edge (0 to 1) of the SYNC signal, this bit becomes 1 and 0, respectively. The E/S interrupt occurs whether the bit is 0 or 1.

In the internal synchronous mode (monosync or bisync) or HDLC mode, a 1 indicates that the hunt phase has been entered, and a 0 indicates that a SYNC character has been received. In either case, an E/S interrupt is generated.

D₃ (Data Carrier Detect)
This bit inverts the state of the DCD pin during latch operations. When the D₃ bit is 1, the DCD pin becomes 0; when it is 0, the DCD pin becomes 1.

At the leading edge of the input signal to the DCD pin, the data carrier detect bit becomes 0. At the trailing edge of the signal, the bit becomes 1 and the E/S interrupt occurs. Table 4-27 shows E/S bit operations.

### Table 4-27. E/S Bit Operations

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Mode</th>
<th>Latch Condition of E/S Bits</th>
<th>Status at Read of E/S Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>D₇</td>
<td>Break/Abort</td>
<td>Async</td>
<td>Start of completion of break sequence</td>
<td>1: Break sequence</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Other than break sequence</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HDLC</td>
<td>Start of completion of abort sequence</td>
<td>1: Abort sequence</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Other than abort sequence</td>
</tr>
<tr>
<td>D₆</td>
<td>Tx Underrun/EOM</td>
<td>Tx Underrun state when Tx CRC is enabled</td>
<td>1: Tx Underrun state 0: Other than Tx</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Underrun state</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(Reset Tx</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Underrun/EOM command.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>After issuance: 0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>After reset: 1.</td>
</tr>
<tr>
<td>D₅</td>
<td>CTS</td>
<td>Leading or trailing edge of CTS input</td>
<td>1: CTS = 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: CTS = 1</td>
</tr>
<tr>
<td>D₄</td>
<td>SYNC/Hunt</td>
<td>Async</td>
<td>Leading or trailing edge of SYNC input</td>
<td>1: SYNC = 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXT sync</td>
<td></td>
<td>0: SYNC = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Monosync Hunt phase</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: During hunt</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: During character</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>synchronization</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HDLC</td>
</tr>
<tr>
<td>D₃</td>
<td>DCD</td>
<td>Leading or trailing edge of DCD input</td>
<td>1: DCD = 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: DCD = 1</td>
</tr>
</tbody>
</table>

D₂ (Transmit Buffer Empty)
When this bit is a 1, it indicates that no data exists in the Tx buffer. This is not true while a CRC character is being transmitted in the synchronous or HDLC mode. When 0, it indicates that data exists in the Tx buffer or that a CRC character is being transmitted.

D₁ (Interrupt Pending)
This bit is applicable to only the SRO register of channel A. SROB bit D₁ is always 0. When the interrupt pending bit is 1, it indicates that an interrupt source exists in either channel A or B, or both. This bit is set if an interrupt source exists in the MPSCC whose service is in progress or being terminated (in service latch set). When this bit is 0, it indicates that the processes of all interrupt sources have been completed (all in service latches are reset). If neither channel is in the DMA mode (D₁ and D₀ of the CR2A register = 1), the interrupt pending bit remains 0 as long as the input signal to the PRI pin is 1.
D0 (Receive Character Available)  
When 1, this bit indicates that one or more bytes of receive data are in the Rx buffer. When 0, it indicates that no receive data is in the Rx buffer.

STATUS REGISTER 1 (SR1)  
The functions of register SR1 bits are summarized in table 4-28.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>End of Frame</td>
<td>Indicates reception of flag pattern (usually a closing flag pattern).</td>
</tr>
<tr>
<td>D6</td>
<td>CRC/Framing Error</td>
<td>Indicates CRC or Framing error.</td>
</tr>
<tr>
<td>D5</td>
<td>Rx Overrun Error</td>
<td>Indicates an overrun error.</td>
</tr>
<tr>
<td>D4</td>
<td>Parity Error</td>
<td>Indicates parity error.</td>
</tr>
<tr>
<td>D3</td>
<td>Residue Code 2</td>
<td>Indicates the valid bit range of an information (I) field.</td>
</tr>
<tr>
<td>D2</td>
<td>Residue Code 1</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>Residue Code 0</td>
<td></td>
</tr>
<tr>
<td>D0</td>
<td>All Sent</td>
<td>Indicates completed transmission of all characters in the transmitter.</td>
</tr>
</tbody>
</table>

D7 (End of Frame)  
This bit is valid only in the HDLC mode. It is used to indicate reception of a valid closing flag and the validity of the CRC/framing bit and residue codes. On reception of a valid closing flag, the end of frame bit is set (special Rx condition interrupt occurs) and the error reset command is issued to clear this condition. When the character of the following frame is received and stored to the Rx buffer, the end of frame bit corresponding to the received character is reset. Also, the result of a CRC check (CRC/framing error bit) and residue codes are valid as long as this bit remains set.

D6 (CRC/Framing Error)  
This bit signals the occurrence of a CRC or framing error. If a framing error is detected in the asynchronous mode (0 is detected at the position of the stop bit), the CRC/framing error bit is set to 1, and the special Rx condition interrupt occurs. The framing error is reset when an error-free character is received or when the error reset command is executed.

Because detection of the stop bit is performed in response to the specified character length, the stop bit of the character will never be mistaken for the start bit of the following character even if the parity bit and stop bit of a character are respectively 1 and 0. This is because detection of the start bit of the following character is triggered at the leading edge of the reception line.

In the synchronous or HDLC mode, the CRC/framing error bit indicates the result of a comparison between the contents of the Rx CRC calculator and a received CRC character. As a result, if the received CRC character agrees with the contents of the Rx CRC calculator, the CRC/framing error bit becomes 0, indicating the normal state. If it does not agree, this bit becomes 1, indicating an error. However, the CRC/framing error bit also becomes 0 when the MPSCC is reset or when the error reset command is issued. The special Rx condition interrupt does not occur in the synchronous or HDLC mode.

CRC comparison is done on a character-by-character basis. Because the results of each comparison are indicated by the CRC/framing error bit, the CRC/framing error bit normally indicates the error state (1) during message reception. In other words, the bit remains 1 because the result of the comparison (performed up to the point when the received character is stored to the Rx buffer) is set in this bit, which corresponds to the Rx buffer. For this reason, this bit should be read after a 20-bit interval following completion of the last CRC character transmission (synchronous mode) or when the end of frame bit becomes 1 (HDLC mode).

D5 (Receive Overrun Error)  
The MPSCC sometimes receives more than three characters because the CPU delayed in fetching receive data. If data is written to the Rx buffer that exceeds the buffer's capacity, the receive overrun error bit will be set each time an attempt is made to write a character. When this bit is set, the special Rx condition interrupt occurs when a character has caused the overrun error. This state is latched until the error reset command is issued.

D4 (Parity Error)  
This bit sets when a parity error occurs (parity enable bit set) because the specified odd or even parity does not match. Once the parity error bit is set, the state is retained until the error reset command is issued. When Rx interrupt mode 2 is specified (D4 and D3 of the CR1 register = 1 and 0 respectively), the parity error bit causes the special Rx condition interrupt.

D3 - D1 (Residue Codes 2 through 0)  
Since the data portion of the an HDLC message can consist of a number of bits and not necessarily an integral number of characters, the MPSCC determines when the end of frame flag has been received and the boundary between valid data and the CRC character in the last few data characters read. When the end of frame condition is indicated (D7 of SR1 is 1) and there is a special receive condition interrupt (if enabled), the last bits of the CRC character are in the receive buffer. The residue code for the frame is valid for the byte of SR1 that is associated with that data character.
The meaning of the residue code depends on the number of bits/character specified for the receiver. Table 4-29 shows the residue codes for 8, 7, 6, and 5 bits/character specified for the receiver. The previous character refers to the last character read before the end of frame, and so on.

**D0 (All Sent)**

In the asynchronous transmit mode, this bit is set when all characters have been transmitted and the Tx buffer and Tx shift register become completely empty. No interrupt is caused by a change in the state of the all sent bit. In the synchronous and HDLC modes (D6 of the CR1 register = 0), this bit is always set.

In HDLC mode (D6 of the CR1 register = 1; 7201A only) the level of this bit changes from 0 to 1 after a series of flags have been transmitted causing the E/S interrupt to occur. The host system is notified that the transmission of one frame is complete. Although the level of this bit changes from 1 to 0 on transmission of the CRC character of the next frame, this change does not cause the E/S interrupt.

**STATUS REGISTER CHANNEL B (SR2B)**

The bits in this register indicate an interrupt vector that can be read only through channel B. When the status affects vector is 0, SR2B will indicate the vector which was last programmed in CR2B. If the status affects vector bit is 1 and an interrupt request exists in the MPSCC, the bits of SR2B will be modified according to the interrupt source. The bits modified are V4-V2 (8085 mode) or V2-V0 (8086 mode) of CR2B.

This interrupt vector corresponds to the interrupt source assigned the top priority among the interrupt requests. If no interrupt request exists, vector bits V4-V2 of the CR2B register are set to 1s in 85 mode or V2-V0 are set to 1s in 86 mode. See table 4-30.

### Table 4-30. Vector Bits V4-V0

<table>
<thead>
<tr>
<th>8085 Mode</th>
<th>8086 Mode</th>
<th>V4</th>
<th>V3</th>
<th>V2</th>
<th>V0</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>No Interrupt Pending</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>Channel B Transmitter Buffer Empty</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td>Channel B External/Status Change</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td>Channel B Received Character Available</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td>Channel B Special Receive Condition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td>Channel A Transmitter Buffer Empty</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>Channel A Special Receive Condition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td>Channel A Received Character Available</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>Channel A Special Receive Condition</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
STATUS REGISTERS 3 AND 4 (SR3 and SR4)

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tx Length Counter Bits 7 - 0

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tx Length Counter Bits 15 - 8

These counters are only available on the μPD7201A. They can be used when data transmission is performed in the TxLR set mode (D6 of the CR1 register = 1) to indicate the number of transmit data characters. The values of the counters are cleared when the system is reset, when the contents of the Tx length register high byte (TxLR-H) and low byte (TxLR-L) coincide with those of the SR4 and SR3 registers, or when the Tx length register is set by setting D6 of the CR1 register to 1. However, in HDLC mode, when the TxLR set mode is specified as well (D6 of the CR1 register = 1), the contents of the TxLR-H and TxLR-L registers are compared with those of the SR4 and SR3 registers when a Tx underrun condition exists. If the contents of the counters are not equal to those of the Tx length registers, an abort sequence will be transmitted. When this occurs, the values of the SR4 and SR3 registers will be retained until the system is reset or the TxLR is set again by setting D6 of the CR1 register to 1.

Tx LENGTH REGISTER, HIGH BYTE AND LOW BYTE (TxLR-H and TxLR-L)

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tx Length Counter Bits 15 - 8

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tx Length Counter Bits 7 - 0

These registers are only available on the μPD7201A. They set the number of data characters to be transmitted. They can be set by two successive command write cycles which follow the setting of D6 of the CR1 register to 1. During the first cycle, the command is written to the low-byte register. During the second cycle, the command is written to the high-byte register.

When the system has been externally reset, or when the channel has been reset, transmission must be enabled (D3 of the CR5 register = 1) after the desired number of transmit data characters is sent to these registers. If the number of transmit data coincides with the values of these registers, the Tx INT/DRQ signal will not become active (even if the Tx buffer is empty) until the desired number of transmit data characters is sent to the registers again (D6 of the CR1 register = 1).
The MPSCC has three communication operating modes: asynchronous, synchronous, and HDLC. The contents of control registers (mainly CR3, CR4, and CR5) specify these modes. This section describes the operations performed in each mode.

**ASYNCHRONOUS MODE**

Figure 5-1 shows the format of serial data in the asynchronous mode. When data is transmitted, the MPSCC adds a start and a stop bit, and if necessary a parity bit. The transmit data consists of data bits D0 through DN. Each set of transmit data provided with a start, stop, and parity bit is a "character". When no transmit data exists, the serial circuit and transmit data pin TXD are set in the mark state.

![Figure 5-1. Asynchronous Message Format](image)

The start bit of the receive data indicates the start of the data. The stop bit indicates the end of the data block. When a data block is received, the MPSCC detects the start and stop bits, checks the parity bit, and deletes these bits from the data block. Therefore, the MPSCC transfers only data D0 through DN to the host system.

In asynchronous mode, data transmission or reception is controlled by registers CR0 to CR5. The CR6 and CR7 registers are not used. Registers CR1 and CR2 specify the mode of the host system that controls the MPSCC. In other words, these registers specify whether the host system should use interrupt, DMA, or polling mode to perform data transmission to and from the MPSCC. Register CR3 specifies the receive mode and CR5 specifies the transmit mode. Register CR4 specifies the mode commonly used for both transmission and reception.

Table 5-1 shows the set contents of registers CR3, CR4, and CR5.

**Data Transmission in Asynchronous Mode**

The CR4 register specifies clock mode, number of bits per character, and state of parity bits. The clock mode sets the data transmission rate by dividing the frequency input at TXC by 1, 16, 32, or 64. For example, if 64 is selected as the clock mode and the TXC pin has a clock of 64 kH (15.625 μs clock cycle), the transmission data rate will be 1 kb/s.

Table 5-1. Contents of Registers CR3, CR4, and CR5 in Asynchronous Mode

<table>
<thead>
<tr>
<th>CR3</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rx Bits/CHAR 1</td>
<td>Rx Bits/CHAR 0</td>
<td>Auto Enable</td>
<td>Enter</td>
<td>Host Phase</td>
<td>Rx CRC Enable</td>
<td>Address Search Mode</td>
<td>Load Inhibit</td>
<td>Rx Enable</td>
</tr>
<tr>
<td>00, 5 Bits/CHAR</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Disable</td>
</tr>
<tr>
<td>01, 7 Bits/CHAR</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Enable</td>
</tr>
<tr>
<td>10, 6 Bits/CHAR</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Disable</td>
</tr>
<tr>
<td>11, 8 Bits/CHAR</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Enable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CR4</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock Rate 1</td>
<td>Clock Rate 0</td>
<td>SYNC Mode 1</td>
<td>SYNC Mode 0</td>
<td>Stop Bits 1</td>
<td>Stop Bits 0</td>
<td>Parity Even/Odd</td>
<td>Parity Enable</td>
<td></td>
</tr>
<tr>
<td>00, x1 Clock Mode</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Disable</td>
</tr>
<tr>
<td>01, x16 Clock Mode</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Enable</td>
</tr>
<tr>
<td>10, x32 Clock Mode</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Disable</td>
</tr>
<tr>
<td>11, x64 Clock Mode</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Enable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CR5</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTR</td>
<td>Tx Bits/CHAR 1</td>
<td>Tx Bits/CHAR 0</td>
<td>Send Break</td>
<td>Tx Enable</td>
<td>CRC-16/CCITT</td>
<td>RTS</td>
<td>Tx CRC</td>
<td>Enable</td>
</tr>
<tr>
<td>0 DTR = 1</td>
<td>00, 5 or Less Bits/CHAR</td>
<td>0 Marking</td>
<td>0 Disable</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>RST</td>
<td>1</td>
</tr>
<tr>
<td>1 DTR = 0</td>
<td>1 Marking</td>
<td>1 Space</td>
<td>1 Enable</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>When all Sent</td>
<td>1 RST</td>
</tr>
</tbody>
</table>

**Note:** Set the parameters of register CR4 before those of the CR1, CR3, and CR5 registers.
The CR1 and CR2 registers specify the mode in which data is transmitted from the host system to the MPSCC. After setting the registers and the Tx enable bit (D3 of the CR5 register), the TxD pin outputs the data's start bit. The start bit is output at the trailing edge of TxC at the transmission rate preset when the data is written to the MPSCC.

The data written to the MPSCC from the host system is then output starting from the LSB (least significant bit) of the data specified by bits D2 and D5 of the CR5 register. If data with a specified character length of 7 or 6 bits is transmitted to the MPSCC from the host system, the MPSCC deletes the unused bit(s) from the 8-bit data. Section 4 (Control Register 5) describes data transmitted with fewer than six bits. Parity and stop bits are automatically transmitted to the MPSCC following the data transmitted from the TxD pin.

When the transmit (Tx) buffer becomes empty while the first data byte is being transmitted, the Tx enable bit (D2 of the CR0 register) is set. If the interrupt mode is specified, an interrupt occurs and the MPSCC asks the host system to send the next data. If no transmit data characters remain for transmission, the TxD pin is set in the mark state (1).

Even if the Tx enable bit is set and data is written to the MPSCC from the host system, no data is output from the TxD pin in the auto enable mode (D3 of the CR5 register is 1) unless CTS becomes 0. Data transmission is complete when no further data is transmitted to the MPSCC or when the Tx enable bit (D3 of the CR5 register) is set to 0. The TxD pin is then set in the mark state, and the transmission operation is complete. Note that the TxD pin can also be set in the mark state by turning the CTS pin to 1 when the auto enable mode is specified.

To set the TxD pin in the space state (0), set the send break bit (D2 of the CR5 register). This state will continue until the send break bit (independent of the Tx enable bit) is reset.

If the external status (E/S) interrupt mode is enabled (D0 of the CR1 register is 1), the states of the DCD, CTS, and SYNC pins can be monitored. That is, each time the level of signals input to these pins changes, an interrupt occurs. In this interrupt routine, use the reset E/S INT command (D5, D4, and D3 of the CR0 register are 0, 1, and 0, respectively) to again enable the E/S interrupt that occurs in response to these changes. The RTS pin outputs 1 when the transmission has been completed if the request to send bit (D1 of the CR5 register) is reset. Under this condition, the all sent bit (D0 of the SR1 register) is set.

Reception in Asynchronous Mode

Reception. Before characters are received in the asynchronous mode, set register CR4 to specify the length of transmit characters, data transmission rate, and whether a parity bit(s) is present. Set the character length with bits D7 and D5 of the CR3 register. The received data assembled by the MPSCC is transferred to the host system in the mode specified by the contents of the CR1 and CR2 registers.

After these registers are set and the Rx enable bit of the CR3 register is set, if RxD changes from 1 to 0 and remains 0 after a half-bit period of time, the MPSCC takes the bit as the start bit and assembles the data following it according to the character bit length specified.

If the parity enable bit is set (D4 of the SR1 register), the MPSCC also detects it and checks for a parity error. When the MPSCC detects the stop bit following the parity bit, it finishes assembling the character. If the character length is eight bits and the parity enable bit has been set, the parity bit will not be transferred to the host system. If the length of a character is 7 bits (parity enable bit set) or less, the MPSCC assembles each character into character bit length + parity bit + 1s for the unused bits. For example, if a character is five bits long and the parity enable bit is set, the format of the assembled data is:

\[
\begin{array}{cccccc}
\text{MSB} & \text{LSB} \\
1 & 1 & P & D_4 & D_3 & D_2 & D_1 & D_0
\end{array}
\]

Error Information During Reception. The MPSCC detects three types of errors: parity, framing, and overrun. If the parity bit of each received character does not coincide with the parity bit calculated by the MPSCC, the parity error bit (D4 of the SR1 register) is set. The error status continues until the error reset command is issued.

Normally, the stop bit of a receive character is 1. If 0 is detected at that bit position, the framing error bit (D5 of the SR1 register) will be set. If no error exists in the next character received, the framing error bit is reset. If x1 clock mode is not specified, the RxC input signals are counted after the trailing edge of the start bit has been detected and the MPSCC has determined the sampling period of the received data. If x1 clock mode is specified, however, the synchronization of the RxD and RxC pins must be externally established because the sampling of the received data is performed at the trailing edge of the RxC input signal. Even if a framing error occurs, the bit at the position of the stop bit of the preceding data will not be taken as the bit of the next. However, if a null character is received with a framing error in it, the framing error bit will not be set. Instead, the break bit (D7 of the SR0 register) is set.

An overrun error occurs when data exceeding the capacity of the Rx buffer is written to the MPSCC. This error occurs when the host CPU cannot read the receive data from the Rx buffer on time. Since the capacity of the MPSCC's buffer is three bytes, the Rx overrun error bit (D5 of the SR1 register) is set if the MPSCC receives a character exceeding that number. The host system is then notified of the
oversat. Should an overrun state occur, the fourth received character (or the excess character) will replace the third character in the Rx buffer, causing bit D2 of the SR1 register to be set. An overrun error can be detected by reading the contents of the status register (SR1). This register indicates the state of the third receive character in the Rx buffer. The overrun error status continues until the error reset command is issued.

**Using Receive Data and Received Status.** Each channel of the MPSCC has a 3-byte receive buffer and error status register (SR1) that corresponds to the 3-byte buffer. In addition, the MPSCC has an Rx character available bit (D0 of the SR0 register) and a break/abort bit (D7 of the SR0 register) that are used as receive status bits.

When receive data is processed in the polling mode controlled by the software, the Rx character available bit is set if at least one character is loaded to the receive buffer. By polling this bit, the host system knows if the MPSCC has received a character.

The contents of the SR1 register must be read before characters are read from the receive buffer. A character is then read from the receive buffer to relate the character to the status of that character. In an interrupt mode, an interrupt occurs if the Rx character available bit is set. The contents of the SR1 register and receive data can then be read after this interrupt signal. The receive character must be read out after these contents have been read in the same manner as in the polling mode. If a parity error occurs, overrun error occurs, each will be treated as the special Rx condition interrupt. After an interrupt in first Rx character mode, receive interrupts occur only when the special Rx condition is satisfied, causing an error status to continue until the error reset command is issued. In other words, the SR1 register retains an error status corresponding to the character that caused it until the error reset command is issued, even if the receive character has been read from the buffer. This interrupt mode is effective when performing block transfer by software or DMA processing.

In all Rx character interrupt mode, receive characters must be read after the contents of the SR1 register have been read to detect any errors of the receive character. If a special Rx condition interrupt occurs in all Rx interrupt mode, the SR1 register must be read to fetch receive characters. The error reset command is used to clear an error status. If the special Rx condition interrupt occurs, and the status affects vector bit (D2 of the CR1 register) is set to 1, it can be distinguished from the normal receive interrupts by using interrupt vectors in both the first and the all Rx character modes.

In addition to the interrupts that occur in either the first or the all Rx character mode, an E/S interrupt occurs (if enabled) when a break sequence is detected and the state of the DCD pin changes. A break sequence is the continuation of a space state that exceeds the number of bits of a null character, resulting in a framing error. When a break sequence is detected, an E/S interrupt occurs at the start (trailing edge) and end (leading edge) of the break sequence.

**Asynchronous Mode Sample**

Figure 5-2 shows an example of full-duplex transmit or receive operations in the asynchronous mode, using both channels A and B. Figure 5-3 shows the interrupt process.

![Asynchronous Mode Flowchart](image-url)
SYNCHRONOUS MODE

The MPSCC uses three types of byte-oriented synchronous protocols: monosync, bisync, and external sync.

In monosync protocol, an 8-bit SYNC character in the CR7 register coincides with serially received 8-bit data, and the MPSCC assumes character synchronization has been established. The characters following the 8-bit SYNC character(s) are then treated as received data. A SYNC character consisting of one or more bytes must be added to data at the start of data transmission to establish character synchronization. The SYNC character transmitted is set in the CR6 register.

In bisync protocol, a 16-bit SYNC character in the CR6 and CR7 registers coincides with serially received 16-bit data, and the MPSCC assumes that character synchronization has been established. The characters following the 16-bit SYNC character(s) are treated as receive data. A SYNC character consisting of one word (16 bits) or more must therefore be added to data at the start of data transmission. When the SYNC character is transmitted, the same SYNC character from the CR6 and CR7 registers is used.

In external sync protocol, character synchronization for reception must be externally established. The SYNC pin specifies the character synchronization timing. Detection of a SYNC character is therefore external. The input signal to the SYNC pin becomes 0 after a SYNC character has been detected. The SYNC character set in the CR6 register is sent at the start of a transmission.

Figure 5-4 shows the data format in the three synchronous protocols.

**Figure 5-4. Synchronous SYNC Message Format**

<table>
<thead>
<tr>
<th>External Sync</th>
<th>Monosync</th>
<th>Bisync</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 Bits</td>
<td>8 Bits</td>
<td>Variable Length</td>
</tr>
<tr>
<td>CRC Character</td>
<td>16 Bits</td>
<td></td>
</tr>
</tbody>
</table>
Table 5-2 shows the contents of registers CR3 through CR7 used in the synchronous mode.

### Table 5-2. Contents of Registers CR3-CR7 in Synchronous Mode

<table>
<thead>
<tr>
<th>CR3</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
<th>RX Bits/CHAR 1</th>
<th>RX Bits/CHAR 0</th>
<th>Auto</th>
<th>Enter</th>
<th>Rx CRC</th>
<th>Address</th>
<th>Load</th>
<th>Inhibit</th>
<th>Rx</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Rx Bit CHAR 1</td>
<td>Rx Bit CHAR 0</td>
<td>Enable</td>
<td>Host</td>
<td>Enable</td>
<td>Search</td>
<td>Mode</td>
<td>Inhibit</td>
<td>Enable</td>
</tr>
<tr>
<td>00, 5 Bits/CHAR</td>
<td>00, 7 Bits/CHAR</td>
<td>01, 6 Bits/CHAR</td>
<td>11, 8 Bits/CHAR</td>
<td>0 Disable</td>
<td>0 Nop</td>
<td>0 Disable</td>
<td>0 Nop</td>
<td>0 Disable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 Enable</td>
<td>1 Re-enabled</td>
<td>1 Enable</td>
<td>1 Inhibit</td>
<td>1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CR4</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
<th>Clock Rate 1</th>
<th>Clock Rate 0</th>
<th>Sync Mode 1</th>
<th>Mode 0</th>
<th>Stop</th>
<th>Stop</th>
<th>Even/Odd</th>
<th>Parity</th>
<th>Parity</th>
</tr>
</thead>
<tbody>
<tr>
<td>00, 8-Bit SYNC</td>
<td>01, 16-Bit SYNC</td>
<td>11, EXT SYNC</td>
<td>00 SYNC Mode</td>
<td>0 Odd</td>
<td>0 Disable</td>
<td>1 Even</td>
<td>1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| CR5 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | DTR | Tx Bits/CHAR 1 | Tx Bits/CHAR 0 | Send | Tx | CRC-16 | TxCRC |  
|-----|----|----|----|----|----|----|----|----|----------------|---------------|------|---|--------|--------|  
| 0 DTR = 1 | 00, 5 or Less Bits/CHAR | 0 Marking | 0 Disable | 0 CITT-0 | 0 RTS = 1 | 0 Disable |  
| 1 DTR = 0 | 01, 7 Bits/CHAR | 1 Spacing | 1 Enable | 1 CRC-16 | 1 RTS = 0 | 1 Enable |  

<table>
<thead>
<tr>
<th>CR6</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sync Bit 7</td>
<td>Sync Bit 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MonoSync</td>
<td>BiSync</td>
<td>Ext Sync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tx SYNC Character</td>
<td>SYNC Character Bit 7-0</td>
<td>Tx SYNC Character</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CR7</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sync Bit 15</td>
<td>Sync Bit 15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MonoSync</td>
<td>BiSync</td>
<td>Ext Sync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rx SYNC Character</td>
<td>SYNC Character Bit 15-8</td>
<td>not used</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1-31
Data Transmission in Synchronous Mode

Initialization for Transmission. Before transmitting data in the synchronous mode, initialize the control registers for the transmit operation. The CR4 register specifies whether monosync, bisync, or external sync mode is used. Set the parameters of the CR4 register before the parameters or commands of the CR1, CR3, CR5, and CR7 registers.

Specify the bit length of a transmit character and the CRC polynomial using the CR5 register. Load a SYNC character to the CR6 or CR7 register, or to both.

In the monosync or external sync mode, the contents of the CR6 register is used as the SYNC character. In the bisync mode, the contents of the CR6 and CR7 registers are used as the SYNC character. The CR1 and CR2 registers specify which mode in which data is transferred to the host system.

After initialization of the transmit operation, the contents (SYNC character) of register CR6 (monosync or external sync modes) or CR6 and CR7 registers (bisync mode) are transmitted when the transmitter is enabled. If there are a number of SYNC characters to be transmitted before data transmission, delay writing the first transmit data character to the MPSCC until after transmission of several SYNC characters.

Write the first transmit data to the MPSCC without using an interrupt or DRQ signal. Afterwards, the interrupt or DRQ signal becomes active each time the transmit buffer becomes empty. If auto enable mode is selected, the data transmission operation will start when the transmitter has been enabled and the CTS pin has become active (low level).

In the synchronous mode, specify either CRC-16 or CCITT as the CRC polynomial and reset it to the initial value (0) by the initial Tx CRC calculator command.

The TxD output pin remains in the mark state when the MPSCC is reset or when data transmission is disabled. When the send break bit (D4 of the CR5 register) is set, the TxD (output) pin is immediately set in the space state (0) whether the transmitter is enabled or disabled. If the space state lasts too long, the contents of the transmit buffer and transmit shift register will be lost.

Data Transfer Mode. When transmit data is transferred to the MPSCC by an interrupt, set the parameters of the CR1 and CR2 registers first. The channel from which data is transferred must be set in the interrupt mode by D1 and D0 of the CR2A register. The Tx INT/DMA enable bit (D0 of the CR1 register) must be set. When the first transmit data is written to the MPSCC after the transmitter has been enabled, the Tx interrupt will subsequently occur each time the transmit (Tx) buffer becomes empty. When an interrupt signal is used for data transfer, the data transfer is performed on a byte-by-byte basis until the message is completely transmitted.

When data transfer is by means of DMA, the DRQTx signal is used in place of the interrupt signal. When DMA mode is used, the transmit channel uses the DRQ signal as selected by the CR2A register, and the Tx INT/DMA enable bit in the CR1 register is set. When the transmitter has been enabled and the first transmit byte is written to the MPSCC, the DRQTx signal becomes active each time the Tx buffer becomes empty. The subsequent transmit data characters are written to the MPSCC by means of DMA on a one-by-one basis using the DRQTx signal.

If the MPSCC uses the WAIT pin before a data transfer, the wait on Rx/Tx bit (D5 of the CR1 register) must to set to 0, and the wait enable bit (D7 of the CR1 register) must be set to 1. The MPSCC prepares for the host system to fetch a transmit byte when the host system writes the first transmit byte to the MPSCC after the transmitter has been enabled. The host system then writes the next transmit byte to the MPSCC. The WAIT pin becomes active if the Tx buffer is not empty and inactive when it is empty. This mode is effective when the host system writes transmit data to the MPSCC in response to the software block transfer command.

When data transfer by software polling is used, transmit data is written to the MPSCC under the control of the Tx buffer empty bit (D2 of the SR0 register). The Tx empty bit is polled, and the data transfer operation waits until the bit is set. When the bit is set, the host system writes the transmit data into the MPSCC.

Tx Underrun Condition and CRC Character Transmission. When the Tx buffer is empty, the Tx buffer empty bit is set, causing the interrupt or DRQTx signal to become active. The MPSCC requests that the host system write the next transmit data. Should the host system not respond, in other words, should the Tx buffer remain empty too long, all transmit data held in the shift register in the MPSCC will be transmitted. As a result, no data to be transmitted will remain in the MPSCC. This state is called the Tx underrun condition, and once it occurs, the Tx underrun/EOM bit (D6 of the SR0 register) is set.

Unless the reset Tx underrun/EOM bit command (D7 and D9 of the CR0 register = 1) has been issued immediately before the Tx underrun condition occurs, a SYNC character will be transmitted. If SYNC character(s) are inserted, CRC calculation will not be performed on them, even if the Tx CRC is enabled. The Tx buffer is empty in this case, and writing transmit data to the MPSCC causes it to be followed by the insert SYNC character(s).

If transmission of CRC characters is enabled, and the transmit enable command of the CRC and the reset Tx underrun/EOM command have been issued, a 2-byte CRC character will be transmitted, followed by SYNC character(s). When the Tx underrun condition occurs, the Tx underrun/EOM bit is set by the RESET signal or by the
channel reset command. The bit is reset only by the reset Tx underrun/EOM bit command. When a CRC character is inserted, there is no time requirement for issuing the reset Tx underrun/EOM command. Issue it only immediately before the Tx underrun condition occurs. Normally the command is issued when the host system detects the end of a message. This command can also be issued when the first transmit data is written to the μPD7201A and the Tx underrun condition occurs prematurely.

If no transmit data is sent from the host system to the MPSCC and the Tx underrun condition occurs, a CRC character is transmitted (7201A only). Note, however, that the CRC character will be inserted if the host system delays writing transmit data to the μPD7201A and the Tx underrun condition occurs prematurely.

While the CRC character is transmitted, the Tx buffer empty bit remains in the reset state. This bit is set, however, when a CRC character has been transmitted and writing the first data byte of the next message to the Tx buffer is enabled. Should the interrupt or DMA mode be specified, an interrupt will occur, or the DRQTx signal will become active.

**Using the Tx CRC Enable Bit.** When a protocol that is dependent on characters (bisync) is used, character(s) may sometimes need to be excluded from the CRC calculation. The MPSCC can exclude unnecessary characters by using the Tx CRC enable bit (D0 of the CR5 register). If transmit data is written to the MPSCC when the Tx buffer is empty and the Tx CRC enable bit is set, the CRC calculation of the written transmit data is performed. If the Tx CRC enable bit is reset, and transmit data is written to the MPSCC, CRC calculation will not be performed.

**PAD Character and Transparent Mode.** To confirm that the first and last characters have been correctly transmitted to the modem, the bisync protocol appends PAD character(s) to the first and last character each time a character transmission operation is performed. The character placed in front of the first character of the transmit data is called a “leading PAD character” (55H), and the character placed after the last character is called a “trailing PAD character” (FFH). The MPSCC can append these two PAD characters to transmit characters.

First, 55H (leading PAD) is loaded to the CR6 register and a SYNC character to the CR7 before the transmitter is enabled. When the transmitter is enabled, register CR6’s contents are changed into a SYNC character after the transmission of 55H followed by SYNC characters (55H, SYN, SYN, SYN).

If the trailing PAD character (FFH) is written to the Tx buffer while the MPSCC is transmitting a CRC character, it, instead of a SYNC character, will be transmitted after the CRC character has been transmitted. Monitoring bit D0 of the SR0 register (Tx underrun/EOM bit), determines whether or not the MPSCC is in the CRC transmission operation or when the E/S interrupt occurs in response to the Tx underrun/EOM bit setting.

The bisync protocol employs a transmission procedure dependent on characters. Processing binary data that cannot be regarded as characters is usually difficult. The transparent test mode should be employed with the bisync protocol for transmission of binary data. In this mode, the DLE.SYN pattern is used as the synchronization pattern instead of SYN.SYN. By loading the DLE character to the CR6 register and SYN to CR7, the MPSCC can process the synchronization pattern of the transparent text mode. The CRC calculation of the SYNC character, transmitted from the CR6 and CR7 registers, is not performed. However, if a character whose CRC calculation is not performed is among data to be transmitted, CRC calculation will be enabled by manipulating the Tx CRC enable bit. Because the MPSCC cannot process control characters other than DLE.SYN (such as DLE.DEL) they must be processed by the host system.

**Transmit Operation in Bisync Mode**

Figure 5-5 shows an example of transmit operation in the bisync mode. Data is transferred between the MPSCC and the host system by interrupt. Figure 5-6 shows the interrupt wait routine.

**Figure 5-5. Transmit Operation, Bisync Mode**

```
Initialize

CR0  05H
CR1  57H
CR2  01H
CR3  08H
CR4  14H
CR5  05H
CR6  0D8H
CR7  0D8H

Write Transmit Data

CR0  05H
CR5  0DH

Interrupt Wait Routine

The Channels are Reset and the Initialize Tx CRC Calculator is Specified.

The Pointer is Specified. Bus Interface Mode is Specified.

The Pointer is Specified. Interrupt Vector is Specified.

The Pointer is Specified. Bisync Mode is Specified.

The Pointer is Specified. Character Bit Length and CRC-16 are Specified.

The Pointer is Specified. Disable Auto-Enable Mode.

The Pointer is Specified. SYN Character is Specified.

The Pointer is Specified. SYN Character is Specified.

The Pointer and Reset E/S INT are Specified. The Status Affects Vector Bit is Set. The Tx INT and E/S INT are Enabled.

The Pointer is Specified. The Tx and Tx CRC are Enabled.
```
When the transmitter is enabled after transmit data has been written, two bytes of SYN.SYN are transmitted, followed by transmit data. If a number of SYN characters need to be transmitted, write transmit data after the predetermined period of time since the transmission was enabled.

Reception in Synchronous Mode

Initialization for Reception. Before performing data reception in the synchronous mode, initialize the MPSCC for reception. The CR4 register specifies the monosync, bisync, or external sync mode. The CR3 register specifies the bit length of the receive characters. The CR5 register determines the CRC polynomial. A SYNC character is loaded to the CR6 register in monosync mode and to the CR6 and CR7 registers in bisync mode. There is no need to load a SYNC character in the external sync mode. The CR1 and CR2 registers specify the mode in which receive data transfer takes place between the host system and the MPSCC.

After the Rx enable bit (D₀ of the CR2 register) is set and initialization is complete, the MPSCC starts to assemble received serial data and compares it with the SYNC character loaded in the CR6 and CR7 registers. The synchronization starts when the assembled character coincides with the SYNC characters. Once synchronization is established, the MPSCC exits the hunt phase and starts assembling the characters it receives. If an assembled character equals the SYNC character in the CR6 register when the reception operation starts, it can be inhibited from being loaded into the Rx buffer. This is made possible by setting the SYNC CHAR load inhibit bit (D₁ of the CR3 register). Even after the MPSCC has gone out of the hunt phase, this bit remains valid. It therefore must be reset when loading a SYNC character in received data.

Transfer of Receive Data. When character synchronization has been established, an assembled receive character is transferred to the Rx buffer. The contents of the Rx buffer can be transferred to the host system by an interrupt or software polling.

A first Rx character interrupt occurs only when receive data is loaded to the Rx buffer after reception has been enabled (D₀ of CR2 register is 0). This interrupt can be used to detect when data transfer using DMA should be performed or when data transfer by software should start. When using a protocol that is dependent on characters (bisync), the
process of checking receive characters must be performed in parallel with the process of first Rx character interrupt. For this reason, data transfer by means of DMA is not normally performed. The first Rx character interrupt is also caused only once by the data loaded into the Rx buffer immediately after the enable interrupt on next Rx character command is issued.

When bit D0 of the CR2A register is 1 (7201A only) in this mode, no interrupt is caused by the character received by either of the channels. Also, the Rx interrupt cannot be enabled by the enable interrupt on next Rx character command. The other operations are performed in the same manner as when bit D0 of the CR2A register is 0.

In all Rx character interrupt mode, an interrupt occurs each time a character is loaded into the Rx buffer.

In special Rx condition interrupt mode, if receive data contains an error when the first Rx character or all Rx character interrupt mode is specified, these modes are treated as the special Rx condition interrupt mode. Although two types of errors - parity and Rx overrun - are possible, the special Rx condition interrupt will not occur if a parity error occurs in the first Rx character interrupt mode. When the all Rx character interrupt mode is specified, a parity error may or may not be included in the special Rx condition interrupt.

Data transfer can be performed by polling the Rx character available bit (D0 of the SR0 register) by the software. When bit D0 of the SR0 register is set, it indicates that receive data is loaded into the Rx buffer for the host system to read from the MPSCC. In this case, to maintain the correspondence between the receive data and its status, this data must be read out to the host system after the contents of the SR1 register are read.

The Rx CRC enable bit (D3 of the CR3 register), controls whether the MPSCC performs CRC calculation of a receive character. That is, when the character has been loaded into the Rx buffer manipulation of the Rx CRC enable bit before the next character is loaded into the buffer will determine whether or not to perform the CRC calculation. If the Rx CRC enable bit is set after a character has been loaded into the Rx buffer (Rx interrupt or Rx character available bit is set) and before the next character is loaded, CRC calculation is performed on the first character.

If the Rx CRC enable bit is reset under that condition, the CRC calculation will not be performed. The 3-byte Rx buffer cannot be used to determine whether to perform the CRC calculation on a case-by-case basis.

The CRC/framing error bit (D6 of the SR1 register) indicates the result of a CRC calculation after a 16-bit interval since a character for the CRC calculation was transferred to the Rx buffer from the Rx shift register. This bit indicates the result of the calculation performed before the character was transferred. If the CRC/framing error bit is 0, no error occurred during the calculation. Only the last result of the CRC check is valid, which means results from early in the calculation are meaningless. Normally, if this bit is 0 during data reception, an error is indicated.

The CRC error check is done in this manner when reception is completed. After a 16-bit interval since the second byte of a CRC character was loaded to the Rx buffer, the CRC/framing error bit is monitored. If the bit is 0, no error has occurred. If an error occurs, the bit turns to 1.

The CRC/framing error bit indicates the state of a CRC error after a 20-bit interval since the last CRC character bit was input to the RxD pin. Accordingly, the RxC clock must be input continuously for a 16-bit interval after the second byte of a CRC character has been loaded to the buffer, or for a 20-bit interval after the last CRC character bit has been input to the RxD pin.

**Receive Operation in Bisync Mode**

Figure 5-7 shows the receive operation in the bisync mode. Interrupts transfer data between the MPSCC and host system. Figure 5-8 shows the interrupt routine.

**Figure 5-7. Receive Operation, Bisync Mode**

| Initialize | CR0 - 08H | The Channels are Reset and the Initialize Rx CRC Calculator Bit is Specified. |
| CR0A - 02H | CR0A - 20H | The Pointer is Specified. Bus Interface Mode is Specified as 6085-1. |
| CR0B - 02H | CR0B - 20H | The Pointer is Specified. Interrupt Vector is Specified. |
| CR0 - 14H | CR0 - 10H | The Pointer is Specified. Bisync Mode is Specified. |
| CR0 - 05H | CR0 - 64H | The Pointer is Specified. CRC Polynomial (CRC-16) is Specified. |
| CR0 - 03H | CR0 - C0H | The Pointer is Specified. Bit Length of Receive Characters is Specified. |
| CR0 - 06H | CR0 - 32H | The Pointer is Specified. SYN Character is Specified. |
| CR0 - 07H | CR0 - 32H | The Pointer is Specified. SYN Character is Specified. |
| CR0 - 11H | CR1 - 15H | The Pointer is Specified. All Rx Character Status Affects Vector and E 3 INT are Enabled. |
| CR0 - 03H | CR3 - C0H | The Pointer is Specified. The Rx is Enabled. Lax 1 of SYNC Character is Inhibited. |
| Interrupt | Wait Routine | |
HDLC (SDLC) MODE

The MPSCC can perform HDLC (high-level data link control) and IBM SDLC (synchronous data link control). The HDLC is a bit-oriented protocol different from byte-oriented protocol, such as BSC (binary synchronous communication). Being bit-oriented, HDLC is a protocol flexible in message length and bit pattern. The MPSCC has several features to process messages of different lengths. For details on the HDLC, refer to the CCITT standards.

Figure 5-9 shows the HDLC message format. An HDLC message, or frame, starts with a flag and ends with a flag. The MPSCC transmits and detects the flag to indicate the frame opening and closing. Flags received by MPSCC are not transferred to the CPU as data.

The 8-bit address field in an HDLC frame indicates the secondary address. If the address search mode has been enabled, detection of the secondary address determines whether or not the MPSCC receives a frame. The control field of an HDLC frame is transparent to the MPSCC; the MPSCC receives it merely as data and transfers it to the CPU.

The MPSCC has a Tx CRC calculator and an Rx CRC calculator to check frames. The Tx CRC calculator is initialized by commands (all bits are preset to 1), and the Rx CRC calculator is automatically reset when the MPSCC detects the opening flag of a frame. The MPSCC also has automatic zero insert and delete functions.

Table 5-3 shows the contents of registers CR3-CR7 when the MPSCC is in HDLC mode.
Table 5-3. Contents of Registers CR3-CR7 in HDLC Mode

<table>
<thead>
<tr>
<th>CR3</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rx Bits/CHAR 1</td>
<td>Rx Bits/CHAR 0</td>
<td>Auto</td>
<td>Enter</td>
<td>Mode</td>
<td>Rx CRC</td>
<td>Address</td>
<td>Search</td>
<td>SYNC</td>
</tr>
<tr>
<td>00, 5 Bits/CHAR</td>
<td>0 Disable</td>
<td>0 NOP</td>
<td>0 Disable</td>
<td>0 NOP</td>
<td>0 NOP</td>
<td>0 Enable</td>
<td>0 Disable</td>
<td></td>
</tr>
<tr>
<td>01, 7 Bits/CHAR</td>
<td>0 Enable</td>
<td>1 NOP</td>
<td>0 Enable</td>
<td>1 NOP</td>
<td>1 Available</td>
<td>1 Enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10, 6 Bits/CHAR</td>
<td>0 Enable</td>
<td>0 NOP</td>
<td>0 NOP</td>
<td>0 NOP</td>
<td>0 NOP</td>
<td>0 Enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11, 8 Bits/CHAR</td>
<td>0 Enable</td>
<td>0 NOP</td>
<td>0 NOP</td>
<td>0 NOP</td>
<td>0 NOP</td>
<td>0 Enable</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CR4</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock</td>
<td>Clock</td>
<td>SYNC</td>
<td>SYNC</td>
<td>Mode 0</td>
<td>Bits 1</td>
<td>Bits 0</td>
<td>Even/Odd</td>
<td>Enable</td>
</tr>
<tr>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CR5</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTR</td>
<td>Tx Bits/CHAR 1</td>
<td>Tx Bits/CHAR 0</td>
<td>Send</td>
<td>Tx</td>
<td>CRC-16/CCITT</td>
<td>RTS</td>
<td>Enable</td>
<td></td>
</tr>
<tr>
<td>0 DTR = 1</td>
<td>00, 5 or Less Bits/CHAR</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1 DTR = 0</td>
<td>01, 7 Bits/CHAR</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CR6</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYNC Bit 7 – 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CR7</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYNC Bit 15 – 8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Flag character (01111110)

Data Transmission in HDLC Mode

Before performing data transmission in the HDLC mode, initialize the MPSCC for transmission. The CR4 register specifies the HDLC mode and the CR5 register specifies the bit length of a transmit character and an HDLC polynomial (CCITT-1: X^{16} + X^{12} + X^5 + 1). The CR1 and CR2 registers set the mode in which data is transferred from the host system to the MPSCC. Set bit D3 of the CR5 register to 1 to enable the transmitter.

After the registers have been properly initialized, flags are transmitted from the TxD pin. Flag transmission will continue until the first transmit data byte is written to the MPSCC. Normally after the transmitter has been enabled, writing the first data to the MPSCC is delayed and two or more flags are transmitted so that the synchronization of the reception side is securely established.

Data transfer to the MPSCC is done by means of an interrupt, DMA, or polling operation. When the transmitter is enabled, the interrupt or DRQ signal is not activated by the empty state of the Tx buffer. Therefore, do not use interrupts, DMA, or polling operations to transfer the first transmit data to the MPSCC after the transmitter has been enabled. Note, however, that an interrupt or DMA request is automatically generated when a flag is loaded to the shift register after the transmitter has been enabled, provided that bit D6 of the CR1 register is 1 (7201A only).

If auto enable mode is specified, the transmit operation is enabled when the CTS pin becomes active after the transmitter has been enabled. Of the frame configurations regulated by the HDLC protocol, address, control, and data fields are transparent to the MPSCC. The MPSCC merely transmits these written data without any change. After the transmitter has been enabled, the first data written to the MPSCC is normally treated as an address field. An 8-bit character transferred then from the MPSCC is transmitted sequentially with the MPSCC starting from its least significant bit (LSB) in the number of bits per character specified by D6 and D5 of the CR5 register. Therefore, if a character is made up of seven bits or less, the data byte to be transferred to the MPSCC must accordingly be shifted to the right by the number of unused bit(s) in compliance with the data format described in Section 4, Control Register 5 (CR5).

Change or specify the transmit character length as
The MPSCC transmits CRC character by using the Tx underrun state. When the Tx underrun state occurs, the Tx underrun/EOM bit is 0. CRC characters are transmitted. In the 7201, the Tx underrun/EOM bit must be reset by command. In the 7201A, it is reset by the first bit of the transmission. If the TxLR set bit (D6 of the CR1 register) is 1, CRC or abort sequences will be transmitted according to the value of the Tx length counter. This applies to the 7201A only. The 7201 has no Tx length counter.

The MPSCC has a zero-insertion function. If five consecutive 1s are detected in a frame when data is transmitted using HDLC protocol, a 0 is inserted after the 1s. However, no 0 is inserted after flags or abort sequences.

**Data Transfer Modes.** When the MPSCC transfers data to the host, it uses three modes: polling, interrupt, and DMA.

In polling mode, data transfer is performed without using interrupts or DMA. The CPU writes the transmit data to the MPSCC by checking the state of the Tx buffer empty bit (D2 of the SR0 register). If the bit is 1, the CPU can write transmit data to the MPSCC. If the bit is 0 (Tx buffer is full), the MPSCC is not ready to accept any transmit data and the CPU cannot write the transmit data.

If the Tx buffer is full when WAIT is enabled (D7 of the CR1 register is 1), an attempt to write data to the MPSCC will cause the WAIT signal to become active. The write cycle of the CPU is prolonged by this WAIT signal until the Tx buffer becomes empty. The WAIT function is used to perform data transmission controlled by the software, such as block transmission.

In interrupt mode, data transfer is performed using an interrupt. Bits D1 and D2 of the CR2A register set both channels A and B in the interrupt mode when they are 00. Only channel B is set in the interrupt mode when these bits are 01. If the Tx INT/DMA enable bit is set, the Tx interrupt occurs when the Tx buffer becomes empty. The CPU writes data to the MPSCC using this Tx interrupt. The Tx interrupt is satisfied when data is written to the Tx buffer or when the reset Tx INT/DMA pending command is issued.

When the reset Tx INT/DMA command is issued without the CPU writing data to the MPSCC, the Tx interrupt is temporarily reset, but the Tx buffer remains empty. The MPSCC then enters the Tx underrun state when data in the Tx shift register has been completely transmitted. The Tx interrupt will occur again on completion of CRC character transmission and when the MPSCC requests the CPU to transfer the next message. This interrupt occurs when a flag is internally loaded. When the transmitter is enabled first, the Tx buffer is already empty. Therefore, the Tx interrupt does not occur until the first character has been written to the MPSCC when bit D9 of the CR1 register is 0. When this bit is set to 1, the interrupt occurs each time the Tx buffer becomes empty.

In DMA mode, data transfer is performed using DMA. Bits D1 and D2 of the CR2A register must be set either to 01 (channel A to the DMA mode), to 10 (channel A and B to the DMA mode), or to 11 (channels A and B to the DMA-2 mode, 7201A only).

The DRQTx signal becomes active when the Tx buffer empty bit is 1 and the DMA controller can write transmit data from the CPU to MPSCC. The DRQTx signal does not become active just by enabling the transmission. The DRQTx signal becomes active when the Tx buffer empty bit is set to 1 following the first data written to the MPSCC when D9 of the CR1 register is 0. When bit D9 of the CR1 register is 1 (7201A only), the signal becomes active when a flag is loaded to the Tx shift register after the transmitter has been enabled. Even if the last data of a frame is transferred to the MPSCC, the DRQTx signal becomes active when the Tx buffer becomes empty. Should the DMA controller not respond under this condition, the Tx underrun condition occurs after data currently existing in the shift register has been transmitted.

If necessary, the DRQTx signal can be reset temporarily by using the reset Tx INT/DMA pending command. This command is valid, however, only when the Tx buffer empty bit is set to 1. The DRQTx signal becomes active again when the MPSCC completes transmission of a CRC character and requests the CPU to transfer the next message.

**Tx Underrun/EOM in HDLC Mode.** When all data has been transmitted from the Tx buffer and Tx shift register (Tx underrun condition occurs), the MPSCC closes the frame currently in process and transmits a 2-byte CRC character, followed by a flag.

The Tx underrun condition is indicated by the Tx underrun/EOM bit. This bit is set by an external reset signal input from the RESET pin, by the channel reset command, or when the Tx underrun State occurs. The Tx underrun/EOM bit is automatically reset when the first character is loaded to the Tx shift register. A CRC character is, therefore, transmitted when the Tx underrun state occurs because the Tx underrun/EOM bit is already 0.

After the MPSCC has been reset, the Tx underrun/EOM bit is 1. This prevents transmission of CRC characters when the MPSCC has no data to transmit. Consequently, a flag is
transmitted when the MPSCC is set in the Tx enable state. When data is written to the Tx buffer, the MPSCC starts to transmit the data.

When the Tx underrun condition occurs after the last data of a message has been transmitted, the Tx underrun/EOM bit becomes 1, causing the E/S interrupt to occur and transmission of a CRC character to start. During the CRC character transmission, the Tx underrun/EOM bit remains 1 and the Tx buffer empty bit 0. These states indicate the Tx buffer is filled with CRC. In the 7201, the reset Tx underrun/EOM command must be given before the underrun occurs to assure that CRC bytes are sent. If this bit is not reset, SYNC characters will be transmitted.

When the CRC characters have been completely transmitted, the Tx buffer empty bit becomes 1 again, and transmission of the next message is enabled. If no message is to be transmitted, the transmitter will be disabled, and the transmit operation will terminate.

Abort Sequence Generation. An abort sequence is a series of 8 to 13 “1” bits. A maximum of 13 bits is possible because it is possible for a maximum of 5 consecutive bits to precede an 8-bit sequence.

When an abort sequence is transmitted, characters currently being transmitted and the contents of the Tx buffer are lost. Then, flags follow completion of the abort sequence transmission.

When the MPSCC is reset, the Tx Underrun/EOM bit is 1 and flags are transmitted until a character is written for the first time since the transmitter was enabled. When the next character (secondary address) is written, the Tx underrun/EOM bit resets to 0 (7201A only) and starts the transmission of frames.

If data transfer from the CPU is delayed despite all data transfers having not yet been completed (Tx underrun condition occurs), the frame must be aborted with an abort sequence. This abort sequence can be generated in the MPSCC automatically (7201A only) or by software.

In automatic generation, if the TxLR set bit (D6 of the CR1 register) is set to 1 and the number of transmit bytes is set to the Tx length register as the initial condition, the MPSCC compares the contents of the Tx length register with those of the Tx length counter (SR3 and SR4 registers) to determine whether to transmit an abort sequence. If the number of data characters to be transmitted (set value of the Tx length register) does not agree with the number of data characters which have been transmitted (set value of the Tx length counter) in the Tx underrun condition, the MPSCC will automatically transmit an abort sequence. Since the 7201 does not have Tx length registers and counters, this is only possible for the 7201A.

The abort sequence can also be generated by issuing a send abort command (CR0). The host system, rather than the MPSCC, decides to generate an abort sequence. The Tx underrun/EOM function of the MPSCC makes this decision possible. The 7201 abort sequence must be generated by the host.

If data transfer to the MPSCC is delayed, the Tx underrun condition will occur. If this condition occurs while the CRC characters are being transmitted, then the host system must decide if the occurrence of the Tx underrun state is caused normally. If the condition is abnormal, the host system secures the time required to issue the send abort command.

If a flag, transmitted when the Tx underrun error occurs, erroneously results in a transmit data pattern that matches the results of the Rx CRC calculator + flag, the frame is probably normal. When that happens, the CRC character transmission function prevents the abnormal frame from being seen as normal. The E/S interrupt generally occurs when the Tx underrun/EOM bit is set to 1 from 0. The host system makes the decision based on this interrupt.

The MPSCC performs the following operations in response to a send abort command and the CPU responds to them.

1) The Tx buffer of the MPSCC becomes empty, and the Tx interrupt and DRQTx signals become active.
2) If either the CPU or DMA controller does not respond to operation (1) within a fixed interval, the MPSCC enters the underrun state.
3) The MPSCC then sets the Tx underrun/EOM bit, and the E/S interrupt occurs.
4) The MPSCC starts transmitting CRC characters.
5) The CPU accepts the E/S interrupt described in operation (3), checks the Tx underrun/EOM bit, and decides if the interrupt was caused by completion of a message, as it should have been.
6) If the CPU decides that the interrupt has not been caused by a normal completion of a message, it immediately issues the send abort command (D5, D4, and D3 of the CR0 register are 001).
7) In response to the send abort command, the MPSCC transmits an abort sequence.

The CPU response time for the above sequence is from 22 to 30 Tx clock cycles.

Abort sequences become valid as soon as the send abort command is written to the CR0 register causing loss of the transmit data. The send abort command is automatically reset after transmission of abort.

CRC Calculation in HDLC Mode. In HDLC protocol, the bits between the opening flag of each frame and a CRC character are CRC-calculated using the polynomial X^16 + X^12 + X^5 + 1. The CRC calculation is performed in the
HDLC mode as follows:

1. The HDLC polynomial is selected by bit \( D_0 \) of the CR5 register. This specification must be performed prior to resetting the Tx CRC calculator.

2. The Tx CRC enable bit is set by bit \( D_0 \) of the CR5 register to enable the CRC calculation. This must be performed before transmitting the first data to the MPSCC (address field).

3. The Tx CRC calculator is automatically reset (all bits become 1) at the beginning of each frame when a flag is loaded to the Tx shift register inside the MPSCC (7201A only).

4. The CRC calculation starts when the first data (address field) is transferred to the MPSCC.

5. Upon completion of data transmission to the MPSCC and when the Tx underrun condition occurs, a CRC character is transmitted onto the data line with each bit of the value generated by the Tx CRC calculator inverted.

**Completion of Transmission Operation (7201A only).** The MPSCC can notify the host system that transmission of a frame has been completed (all sent bit is 0) by using the E/S interrupt in HDLC mode. This E/S interrupt occurs when a flag pattern of one byte or more is transmitted from the TxD pin after a CRC character has been transmitted. Inside the MPSCC, however, the interrupt occurs when flag characters of three bytes or more are about to be transmitted.

Because the E/S interrupt caused by the Tx underrun/EOI bit does not indicate completion of a frame transmission, this interrupt is used to detect it.

The E/S interrupt occurs when bit \( D_0 \) of the SR1 register is set to 1. When it occurs, the contents of the SR0 and SR1 registers must be read. Should the transmitter be disabled, the characters held in the Tx buffer will remain unchanged and will not be transmitted.

**Transmit Operation in Tx Length Set Mode (\( D_0 \) of the CR1 Register is 1) (7201A only).** Data transmission using the Tx length register is performed differently from transmission without using it. This mode, in which the desired number of bytes is set to the Tx length register, is especially effective when using both channels A and B in the DMA mode (especially in the both CH.DMA-1 mode when bits \( D_1 \) and \( D_0 \) of the CR2A register are set to 1 and 0). In this mode, the Tx DMA request DRQTx signals automatically become inactive when the number of bytes sent to the Tx length register has been completely transmitted. When bit \( D_0 \) of the CR1 register is set to 1 again, the DRQTx signal becomes active again. If the Tx underrun condition occurs before the number of data transmitted has reached the number set in the Tx length register, the MPSCC will automatically abort the frame. The contents of the SR3 and SR4 registers can confirm whether the frame has been aborted or the transmission has been normally completed. If the frame has been aborted, the contents of the registers indicate the number of data character transmitted up to the point where the abort has occurred - 1. If the contents are 00s, the transmission has completed normally.

To resume transmission after the frame has been automatically aborted, activate the DRQTx signal by setting bit \( D_0 \) of the CR1 register to 1 as if the transmission had been completed normally. The DRQTx signal becomes active after the contents of the Tx length register have been modified.

**Transmit Example in HDLC Mode**

Figure 5-10 shows an example of a transmit operation in HDLC mode. The example assumes data is transmitted using DMA. Figure 5-11 shows the E/S interrupt process.

**Figure 5-10. Transmit Operation, HDLC Mode**

![Diagram](https://example.com/diagram.png)

**Note:** In this example, the DRQTx signal becomes active and requests the first data transmission when CTS becomes low after initialization. Afterward, the DRQTx signal becomes active each time the Tx buffer becomes empty. When the number of data set in the TXLR has been written, subsequent MPSCC DRQTx signals do not become active.

1-40
Reception in HDLC Mode

Receive Operation in HDLC Mode. To receive data in the HDLC mode, perform the following steps:

1. Specify the HDLC mode by setting the necessary parameters in the CR4 register. Do this before setting parameters in other registers.

2. Use the CR5 register to specify the HDLC polynomial (CCITT-1: $X^{16} + X^{12} + X^{5} + 1$).

3. Specify the secondary address and flag character with the CR6 and CR7 registers, respectively.

The MPSCC enters the hunt phase when the enter hunt phase bit (D4 of the CR3 register) is set. This state continues until the first flag (opening flag) is detected after the receiver is enabled. In the HDLC mode, once the MPSCC goes out of the hunt phase and synchronization is established, the synchronization will be maintained until the receiver is disabled. When the MPSCC detects the valid opening flag, it sets the SYNC/hunt bit to 0 and the E/S interrupt occurs. Subsequently, the MPSCC assembles and transfers the serial data input from the RxD pin to the Rx buffer. Data that exists between the opening and closing flags is transparent to the MPSCC. The MPSCC performs no special process on the received data.

The MPSCC has an address search mode function that determines whether the MPSCC should receive the transmitted data. In this mode (D2 of the CR3 register is 1), the MPSCC will only start assembling data when the first data byte (secondary address) following a valid flag matches either the address set in the CR6 register or the global address 11111111. The address search function is used only to assemble 1-byte address data. The host system therefore must decide, when a frame configured of 2 bytes or more of extended address is used, whether the frame should be received. In the event that it is received, the MPSCC transfers the extended address following the first address byte to the CPU if the first address byte coincides with the contents of the CR6 register or with the global address. The CPU, in turn, must check the extended address. Should the CPU decide the frame is unnecessary, it will set the enter hunt phase bit again, temporarily stop reception, and wait for the next frame.

Because the length of the I field is not regulated by HDLC protocol, the MPSCC can modify the transmit character length as desired. The modification must take place while the character whose character length has been changed is being assembled. In other words, the modification must be made at an appropriate time so that it becomes valid before the number of bits specified as the character length is completely assembled. For example, if the address and control field are eight bits long and the I field following them is seven bits long, the modification must be carried out after the control field has been assembled and while the 7-bit character is being assembled.

The abort detection function of the 7201A is valid when the opening flag is detected after reception is enabled. The abort detect function for the 7201 is valid during the hunt phase as well. When eight or more 1s are received (abort sequence), the break/abort bit (D7 of the SR0 register) is set and the E/S interrupt occurs if the interrupt has been enabled. After detecting the abort sequence, the MPSCC automatically enters the hunt phase. If the E/S interrupt occurs because of detection of the abort sequence, the MPSCC will issue the reset E/S INT command. To distinguish whether the detected bit string is an abort sequence or an idle sequence, time is measured from when the reset E/S INT command was issued. If the abort state is reset (zero following the abort sequence is detected), the E/S interrupt occurs. The special Rx condition interrupt occurs, which is caused by the end of frame, when the MPSCC detects the closing flag. Normally, by using this interrupt, the MPSCC detects the completion of frame transmission. If the data length between the opening
and closing flags is seven bits or less, the frame will not be received. If the data length is more than seven bits, the frame will be received. The MPSCC automatically deletes the zero inserted in the data, using its zero deletion function. Flags are not transmitted to the CPU. If the E/S interrupt is enabled, the E/S interrupt will also occur when the state of the DCD pin changes.

**Transfer Modes of Receive Data.** Receive data can be transferred from the MPSCC to the CPU in HDLC mode by polling, interrupt, or DMA.

In polling mode, the CPU reads receive data from the MPSCC as it checks the state of the Rx character available bit (D0 of the SR0 register). The MPSCC turns the Rx character available bit to 1 each time it receives a character and requests the CPU to fetch the receive data out of the MPSCC. If the Rx character available bit is 1, the CPU will read receive data from the MPSCC.

In data transfer by interrupts, data transfer using an interrupt can be in first Rx character interrupt mode or all Rx character interrupt mode.

In the first Rx character interrupt mode (D4 and D3 of the CR1 register are 0), an interrupt is caused by the first received character only. Subsequently, an interrupt will occur only each time the special Rx condition is detected. However, when the enable interrupt on next Rx character command (D5, D4, and D3 of the CR0 register are 1, 0, and 0, respectively) is issued, an interrupt can reoccur when a character is received after the command. In the Rx INT mask mode (D5 of the CR2A register is 1, 7201 A only), an interrupt is not caused by the first received character. The interrupt which is caused by a received character cannot be enabled by issuing the enable interrupt on next Rx character command. This mode can be used to start a polling operation by the software or to block transmission by using the WAIT signal or to start a DMA operation. In the all Rx character interrupt mode (D4 and D3 of the CR1 register are either 10 or 11), an interrupt occurs whenever a character exists in the Rx buffer. Also in this mode, an interrupt provided with an interrupt vector indicating the interrupt, occurs when the special Rx condition is detected (status affects vector is selected). The character immediately before a closing flag (normally the second byte of a CRC character) is read from the MPSCC by the process routine for the special Rx condition interrupt, because of an end of frame.

In data transfer in DMA mode, the first Rx character interrupt mode is selected. The DMA controller usually is enabled by this interrupt process routine, and the DMA operation starts. When an opening flag is detected, the DRQRx signal becomes active each time the MPSCC receives a character and the MPSCC requests the DMA controller to read the receive data. On the other hand, in the first Rx character mode, an Rx interrupt occurs upon reception of the first character. This interrupt can be masked by setting D9 of the CR2A register to 1 (Rx INT mask mode, 7201A only). If the DMA controller is enabled before the first Rx character interrupt occurs, it processes the interrupt before the CPU accepts it. When the receive data is read from the MPSCC to the CPU, the interrupt request signal is reset. When a closing flag is received, the special Rx condition interrupt occurs because of an end of frame, and the MPSCC informs the CPU of the completion of frame reception.

**Special Rx Condition in HDLC Mode.** In the HDLC mode, the special Rx condition interrupt occurs when the Rx Overrun state or end of frame state is detected. When the MPSCC enters either state, an interrupt with a vector indicating the special Rx condition can be generated (status affects vector is selected). The special Rx condition interrupt, however, is not an independent interrupt mode, and either the first Rx character or the all Rx character interrupt mode must have been selected before the conditions necessary to generate this interrupt are established. The Rx overrun status is latched, and the error information read from the MPSCC indicates an error occurred either in the data stored in the Rx buffer or after the error reset command was issued. The special Rx condition interrupt due to an Rx overrun error is generated when the MPSCC reads the data that caused the Rx overrun state.

The special Rx condition interrupt due to the end of frame state is generated when a valid closing flag is received. If the end of frame bit is set, the CRC error bit and residue codes will become valid.

**CRC Calculation of Receive Data.** The MPSCC performs control of the Rx CRC calculator during a receive operation. The Rx CRC calculator is reset by an opening flag (all bits checked are set to 1) and completes the CRC calculation in response to a closing flag. It can also be reset by the reset Rx CRC checker command when bits D7 and D6 of the CR0 register are 0 and 1, respectively.

In the HDLC mode, CRC calculation of all receive data is performed, but no 8-bit delay occurs as in the synchronous mode. Because results of the CRC calculation are inverted when they are transmitted, a special sequence is used for the CRC check of the MPSCC during the receive operation. A 2-byte CRC character is used for the CRC check inside the MPSCC and can be received as normal data. The CRC calculation is complete when the second CRC character has been transferred to the RX buffer. In the 7201A, both bytes of the CRC character can be received as data. In the 7201, only the first CRC byte is received as data correctly.

**Completion of Receive Operation.** When the 7201A receives a closing flag, the end of frame bit (D7 of the SR1 register) is set, and the special Rx condition interrupt occurs. Under this condition, the CRC framing error bit (D7
of the SR1 register) and residue codes (D3, D2 and D1 of the SR1 register) are valid. The residue codes indicate the number of bits on the boundary between a 16-bit CRC character bit and an l field.

The CPU must immediately issue the error reset command (D5, D4, and D3 of the CR0 register are 110) to clear this state after performing necessary processes in the special Rx condition interrupt process routine caused by the end of frame bit.

The MPSCC continues receiving input frames even when it has entered the end of frame state after detecting a closing flag. These receive data characters, however, are not transferred to the last byte of the Rx buffer until the error reset command is issued. This is in case of an INT on first Rx character mode. If no subsequent message is to be received by the MPSCC, the receiver will be disabled to terminate the receive operation.

Figure 5-12 shows a receive operation in HDLC mode. This example assumes that data is transmitted using the DMA mode. Figure 5-13 shows the special Rx condition interrupt process. Figure 5-14 shows the reset E/S interrupt.

**Figure 5-12. Receive Operation, HDLC Mode**

<table>
<thead>
<tr>
<th>CR0</th>
<th>18H</th>
<th>The Channel is Reset.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR0A</td>
<td>02H</td>
<td>Pointer 2 is specified. The Bus Interface Mode (Rx INT Mask, Vectored, Bit-Vectored, and Both ChDMA-2 Mode) is set.</td>
</tr>
<tr>
<td>CR2A</td>
<td>73H</td>
<td>Pointer 2 is specified. The Bus Interface Mode (Rx INT Mask, Vectored, Bit-Vectored, and Both ChDMA-2 Mode) is set.</td>
</tr>
<tr>
<td>CR0B</td>
<td>02H</td>
<td>Pointer 4 is specified. The HDLC Mode is set.</td>
</tr>
<tr>
<td>CR2B</td>
<td>0H</td>
<td>Pointer 5 is specified. CCITT-1 is set.</td>
</tr>
<tr>
<td>CR0</td>
<td>04H</td>
<td>Pointer 6 is specified. The Secondary Address is set.</td>
</tr>
<tr>
<td>CR4</td>
<td>20H</td>
<td>Pointer 7 is specified. Flag is set.</td>
</tr>
<tr>
<td>CR5</td>
<td>05H</td>
<td>Pointer 1 is specified. The Reset E/S INT, INT on First Character Status Affects Vector (Set Only in Channel B), and E/S INT Enable are set.</td>
</tr>
<tr>
<td>CR6</td>
<td>06H</td>
<td>Pointer 3 is specified. Rx Character Length, Auto Enable Mode (Reception is Enabled When DCD is Low and Rx Enable Operation is Performed), Enter Hunt Phase Enable, Address Search Mode, and Rx Enable are set.</td>
</tr>
</tbody>
</table>

**Figure 5-13. Special Rx Condition Interrupt Process**

- **Special Rx Condition Interrupt Process**
- **SR1 is Read**
- **Analysis of Interrupt Source**
- **Interrupt Caused by Change in End of Frame Bit from 0 to 1**
- **Receive Data is Read**
- **Error Process**
- **CRC Error?**
- **Reception Completion Process**
- **Error Reset Command is Issued**
- **End of Interrupt Command is Issued**
- **Return from Special Rx Condition Interrupt Process**

Note: In this example, a receive operation starts when the level of DCD becomes low after initialization. When a valid flag is detected, the first data (other than flags following the valid flag) is compared to the contents of the CR6 register. If this data is the same (address agreement), DRQR becomes active each time data including this address data is received and the transfer of receive data is requested.
Figure 5-14. Reset E/S Interrupt Process

1. Reset E/S INT
2. SR0 is Read.
3. Analysis of Interrupt Source
4. Process Appropriate for Interrupt Source
5. Reset E/S INT Command is Issued.
6. End of Interrupt Command is Issued.
CPU INTERFACE TIMING

CPU interface timing consists of read/write, WAIT, interrupt, and DMA timing.

Read/Write Timing

Figure 6-1 is a diagram of the interface between the CPU and the MPSCC. It also shows timing for reading out data or status information in the MPSCC (read cycle) and for writing commands, parameters, or data to the MPSCC (write cycle). The CPU is an 8085A.

WAIT Timing

Figure 6-2 shows the basic timing chart of a WAIT operation. Both the read cycle in which the CPU fetches data from the MPSCC and the write cycle in which it writes data to the MPSCC can be prolonged by inputting WAITA and WAITB signals to the CPU's READY pin. These two signals are active low.

The WAITA pin also functions as the DRQRxA pin and the WAITB as the DROTxA pin. When D1 and D0 of CR2A register are set to zero, these two pins function as the WAIT pins.

The WAIT pins are open drain pins. Their impedance is high until they become active. Both WAITA and WAITB pins can be wired OR.

Both pins become active low level under the following special conditions:

- D1 and D0 of the CR2A register are both 0s in the WAIT mode.
- D7 of the CR1 register is 1 in the wait enable mode.
- The Tx buffer is full or the Rx buffer is empty and the level of the CS and the C/D pin is 0.

The WAIT pins go to a high level when the Tx buffer becomes empty or the MPSCC is in the Rx character available state, thus enabling the read/write operation.

Interrupt Timing

Vectored Mode. Figure 6-3 shows the basic timing chart of an interrupt in the vectored mode. Figure 6-4 shows a timing chart when more than one interrupt source exists in this mode.

The MPSCC accepts its own internally generated interrupt request when the internal INTAK signal (set at the trailing edge of the first externally input INTAK signal and reset at the leading edge of the second externally input INTAK signal) is inactive. An internal interrupt is disabled when the internal INTAK signal is active.

When an interrupt assigned a priority is enabled, interrupts with lower priority will remain disabled. When an internal interrupt is enabled, the INT pin of the MPSCC goes low and notifies the interrupt controller (for example, the 8259A). At the same time, the PRO pin turns high, and interrupts by an MPSCC connected in the lower order of the daisy chain are disabled.

When the PRI input is high, an output signal from the INT pin will be kept at a high level until the input to the PRI pin becomes low. Because the INT output pin is an open drain pin, the output must be pulled up to turn it to a high level. When the MPSCC enters the interrupt acknowledge cycle, the INTAK signal is input to it from the CPU, and an interrupt request is latched at an in service latch. The MPSCC then analyzes the priority of the interrupt in the time interval between the first and second falling edges of the INTAK signal. As a result, the MPSCC sets the in service latch corresponding to the highest-priority interrupt at the second falling edge of the INTAK signal. As the in service latch is set, the INT signal is reset at a high level.

When an in service latch is set, an interrupt assigned a lower priority will not be enabled, and the in service latches of lower levels also will be reset.

In service latches are reset when the EOI command is issued (D3, D4, and D5 of the CR0 register are all 1s). At this point, the in service latch assigned top priority will be reset. An input signal to the CS pin must be inactive (CS = 1) while the INTAK pin remains at a low level (0).

Even when an in service latch with priority is set, if an interrupt request signal with a higher priority is generated, the interrupt can be enabled during the interrupt enabled period.

Figure 6-4 shows interrupt timings of three interrupts, labeled A, B, and C. Interrupt A has highest priority, interrupt B the second highest, and interrupt C the lowest. After the INT signal is output because of interrupt C, interrupt B will be enabled if it occurs before the INTAK signal is received. When the MPSCC enters the interrupt acknowledge cycle, the in service latch corresponding to interrupt B is set, and the interrupt vector corresponding to interrupt B will be sent to the CPU if the status affects vector mode is specified.

If interrupt A occurs, however, while interrupt B is being serviced (in service latch of interrupt B is 1), it will be enabled instead of B because of its higher priority, and the INT signal will be generated. When the CPU issues the EOI command after servicing interrupt A, the in service latches of interrupt A and interrupt B are both reset. After all three interrupts have been serviced and all interrupt sources in the MPSCC and in service latches have been cleared, the output signal from the PRO pin is reset, and the interrupt of the MPSCC connected in the lower order of the daisy chain will be enabled.
Figure 6-1. Read/Write Timing
The interrupt operation is applicable in the 85-1 and 85-2 interrupt modes (D3 and D4 of the CR2A register are either 00 or 01). However, in the 85-3 interrupt mode (D3 and D4 of the CR2A register are 11, 7201A only) the INT output will become active (low level) if there is an interrupt in the MPSCC even when the PRI input is high.

**Nonvectored Mode.** Figure 6-5 shows an interrupt timing chart in the nonvectored mode. If an interrupt assigned a priority is enabled, no interrupt with a lower priority can be enabled.

When an interrupt is enabled, the INT signal of the MPSCC becomes active, and the MPSCC notifies the CPU. The CPU specifies the SR2B register (pointer two) for the MPSCC to analyze the interrupt source and fetch the interrupt vector.

The MPSCC then sets an in service latch at the falling edge of the RD signal while the SR2B register is being specified and latches the information that an interrupt has been enabled. Once the SR2B register is specified, the MPSCC disables all additional interrupts. It analyzes the priority of the enabled interrupt until the RD signal becomes low, and the in service latch with the highest priority then becomes active. In this manner, specifications of the SR2B register and the RD signal are used in the nonvectored mode to confirm that an interrupt has been enabled by the CPU. This is different from interrupt operation in the vectored mode. Other operations performed in the nonvectored mode, however, are the same as those in the vectored mode.

Also, no subsequent interrupt will be enabled unless the CPU performs a read operation (RD) in the nonvectored mode after the SR2B register has been specified. In the nonvectored mode, the INTAK pin must be kept in the inactive state (INTAK = 1).

**DMA Timing**

**DMA Timing in Both CH.DMA-2 Mode.** (7201A only) The both CH.DMA-2 mode is set by setting D1 and D0 of the CR2A register to 1s. In this mode, DMA priority in the MPSCC is not fixed by a command and DMA process can be performed without using the HA1 or HA0 pins.

An internally generated DMA request is accepted during the low-level period of the CLK signal. However, in this low-level period, data-read and data-write periods are not included. This acceptance of DMA requests is independent of the priority, and an accepted DMA request causes the corresponding pin to become active.

The DMA request is then sent to the DMA controller (µPD8237A, for example). The CS, C/D, and B/A pins of the MPSCC must be controlled using the DACK signal of the DMA controller during each acknowledge period. The DACK signal determines whether an accepted DMA request has been satisfied. The DACK, RD, and WR signals were input during the DMA acknowledge period. Therefore, the DMA controller determines and controls the priority of a DMA operation.

Figures 6-6 and 6-7 show the DMA timing and an example for interfacing the µPD8237A with the MPSCC in this mode.

**Figure 6-5. Interrupt Timing in Nonvectored Mode**
Figure 6-6. DMA Timing, Both CH. DMA-2 Mode

Figure 6-7. Example of MPSCC and μPD8237A Interface
CH.A DMA and Both CH_DMA-1 Modes. When both channels A and B of the MPSCC (D1 and D0 of the CR2A register are 10) or only channel A (D1 and D0 of the CR2A register are 01) is set in the DMA mode, CH.A DMA and both CH_DMA-1 modes are respectively specified. In these modes, the MPSCC enables an internally requested DMA operation while the following occurs:

- The CLK signal is at a low level when the HAI pin is inactive (high level).
- The trailing edge of the RD or WR signal coincides with an active HAI pin.

DMA requests are accepted independent of the priority assigned to each. Whenever one is accepted, each DMA request (DRQ) pin becomes active.

The output DMA requests accepted in this manner are sent to the DMA controller (µPD8257). If either the HALO signal from the MPSCC, connected in the higher order of the daisy chain, or the HLDA signal from the CPU is input to the HAI pin of an MPSCC, that MPSCC enters the DMA cycle and starts the DMA process.

When a DMA request is received, the MPSCC sets the in service latch corresponding to the request at the trailing edge of the input signal to the HAI pin. If more than one in service latch is set, the one with highest priority is serviced first.

Subsequently, if either the RD or WR signal is input while the HAI pin is active, the DMA request being serviced is cleared at the trailing edge of the signal, and another DMA request is the serviced according to priority. When the MPSCC outputs a DMA request and the HAI pin is 0, the CS and C/D signals in the MPSCC are automatically set to 0. The B/A signal becomes 0/1 according to the channel at which a DMA request is being serviced. Therefore, it is not necessary to externally make the CS, C/D and B/A pins active. While the HAI pin is set to 0 in the DMA cycle, the input signal to the CS pin must be inactive (CS = 1). This also applies to the interrupt cycle while the INTAK pin is 0.

When the µPD8257 is the DMA controller, the priority of DMA requests used in the MPCC and those in the µPD8257 must agree. Figures 6-8 and 6-9 show the DMA timing in this mode and an example of an interface between the µPD8257 and MPSCC, respectively.

Figure 6-8. DMA Timing, CH.A DMA and Both CH_DMA-1 Modes

Figure 6-9. Example of MPSCC and µPD8257 Interface
Figures 6-10 through 6-17 show the timing of MPSCC transmission/reception in the asynchronous, synchronous, and HDLC modes. These figures show timings related both to transmit/receive data and interrupt signals.

**Figure 6-10. Transmission in Asynchronous Mode**
Figure 6-11. Reception in Asynchronous Mode

![Diagram showing reception in asynchronous mode with notes 1 and 2.]

Note 1: The special Rx Condition interrupt occurs if a Parity, Framing, or Ovrun error occurs.
Note 2: The x16 clock mode is specified and each character is made up of 5 bits.

Figure 6-12. Transmission in Synchronous Mode

![Diagram showing transmission in synchronous mode with notes.]

Note: Parity enabled.

Figure 6-13. Reception in Synchronous Mode

![Diagram showing reception in synchronous mode with notes.]

1-52
Figure 6-14  Reception in Synchronous Mode (CRC Calculation)

Figure 6-15. Transmission in HDLC Mode (1)

Figure 6-16. Transmission in HDLC Mode (2)
Figure 6-17. Reception in HDLC Mode
This section describes the various μPD7201A MPSCC system configurations.

**STATUS POLLING MODE**

The status polling mode is the simplest system configuration since it does not use the interrupt or DMA request pins. Data transmission between the MPSCC and the main system is by polling the SR0, SR1, and SR2 status registers of the MPSCC. The WAIT pins can be used if needed.

Figure 7-1 shows an example of the status polling mode system configuration during data transmission. If the MPSCC is set in the interrupt mode (both channels A and B) or the nonvectored mode, with the status affects vector mode enabled, the interrupt vector in the SR2B register can analyze interrupt sources.

**CHANNLES A AND B IN INTERRUPT MODE**

In this type of system configuration, interrupts of both channels A and B perform data transmission between the MPSCC and the main system.

**Vectored Mode**

If the MPSCC is set in the vectored mode, an interrupt request signal is input from the MPSCC to the controller (μPD8259A, for example) or directly to the CPU. Set the vectored modes in the MPSCC as shown in table 7-1.

<table>
<thead>
<tr>
<th>CPU</th>
<th>MPSCC 8259A</th>
<th>MPSCC #1</th>
<th>Other than MPSCC #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>8086A</td>
<td>Not provided</td>
<td>85-1 Vectored</td>
<td>85-2 Vectored</td>
</tr>
<tr>
<td></td>
<td>Provided</td>
<td>85-2 Vectored</td>
<td>85-2 Vectored</td>
</tr>
<tr>
<td>8086</td>
<td>Not provided</td>
<td>86 Vectored</td>
<td>86 Vectored</td>
</tr>
<tr>
<td></td>
<td>Provided</td>
<td>86 Vectored</td>
<td>86 Vectored</td>
</tr>
</tbody>
</table>

Figure 7-2 shows a system configuration using the interrupt vectored mode.
Please note that the 85-3 vectored mode differs in its configuration. This mode does not allow a daisy chain using the PRI and PRO pins. However, the output signal from the INT pin becomes active even if the level of the PRI pin is 1. Therefore, the 85-3 vectored mode can be used as a slave of the 8259A. Figure 7-3 shows a system configuration using this mode. Note that 85-3 vectored mode is available for the 7201A only.

**Figure 7-3. Interrupt System (85-3 Vectored Mode)**

Nonvectored Mode

Figure 7-4 shows a system configuration in the nonvectored mode. An output signal from the INT pin of the MPSCC is input either to the RST pin of the μPD8085A (RST5, RST6.6, RST7.5) or to the interrupt controller (such as μPD8259A). When more than two MPSCCs are connected, software must also be used to decide which MPSCC is requesting an interrupt.

**Figure 7-4. Interrupt System (Channels A and B In Interrupt, Nonvectored Mode)**

**CHANNEL A IN DMA MODE AND CHANNEL B IN INTERRUPT MODE**

Figure 7-5 shows a system configuration with channel A set in the DMA mode and channel B in the interrupt mode for data transmission between the MPSCC and the main system. This configuration permits high-speed transmission on channel A and low-speed transmission on channel B. DMA request signals are input to a DMA controller, such as the μPD8257.

If the MPSCC is set in the nonvectored mode, interrupt request signals from the MPSCC are input either to the interrupt controller (such as the μPD8259) or directly to the CPU. In this case, set the MPSCC modes as shown in table 7-2.

**Table 7-2. MPSCC Nonvectored Mode**

<table>
<thead>
<tr>
<th>CPU</th>
<th>MPSCC Mode</th>
<th>Other Than MPSCC #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>8085A</td>
<td>Not provided</td>
<td>85-1 Nonvectored</td>
</tr>
<tr>
<td></td>
<td>Provided</td>
<td>85-2 Nonvectored</td>
</tr>
<tr>
<td>8086</td>
<td>Not provided</td>
<td>86 Nonvectored</td>
</tr>
<tr>
<td></td>
<td>Provided</td>
<td>86 Nonvectored</td>
</tr>
</tbody>
</table>
In the nonvectored mode, interrupt request signals can be input to the RST pin of the μPD8085A.

**Figure 7-5. Interrupt/DMA System (Vectored Mode)**

**Channells A and B in DMA Mode**

It is possible to use two system configurations with both channels set in the DMA mode. The DMA mode used decides which system is configured. In either case, data communication between the MPSCC and the main system is implemented by performing a DMA operation with both channels. In this manner, high-speed data transmission is possible.

The DMA request signal is input to the DMA controller (for example, the μPD8237A). The interrupt signal is generated in response to the E/S, the first Rx character, or the special Rx condition interrupt. Figure 7-6 shows a DMA system configuration.

**Both CH.DMA-2 Mode**

In both CH.DMA-2 mode (7201A only), the DMA process is performed using the DACK signal instead of the DMA controller's HA1 and HAO pins. Figure 7-6 shows a system configuration in this mode. If the configuration uses only one MPSCC, the vectored mode can be used. If so, set the MPSCC vectored modes as shown in table 7-3.

**Table 7-3. Both CH.DMA-2 Mode**

<table>
<thead>
<tr>
<th>CPU</th>
<th>8256A</th>
<th>Mode to be set in MPSCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>8085A</td>
<td>Not provided</td>
<td>85-1 mode</td>
</tr>
<tr>
<td></td>
<td>Provided</td>
<td>85-2, 85-3 modes</td>
</tr>
<tr>
<td>8086</td>
<td>Not provided</td>
<td>86 mode</td>
</tr>
<tr>
<td></td>
<td>Provided</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** If two or more MPSCCs are used, specify the nonvectored mode.

**Figure 7-6. DMA System (Both CH.DMA-2 Mode)**
Both CH.DMA-1 Mode

In both CH.DMA-1 mode, DMA processes are carried out using the HAI and HA0 pins. Both channels can simultaneously perform transmission operations using the Tx length register (Db of the CR1 register = 1).

**Vectored Mode.** Figure 7-7 shows a system configuration in the vectored mode. Use only one MPSCC in this mode. The INTAK signal functions inside the MPSCC as if it were the PRI signal and assigns priority to interrupt sources.

Set the MPSCC vectored modes as shown in table 7-4.

**Table 7-4. Both CH.DMA-1 Mode**

<table>
<thead>
<tr>
<th>CPU</th>
<th>8258A</th>
<th>MPSCC Modes</th>
</tr>
</thead>
<tbody>
<tr>
<td>8085A</td>
<td>Provided</td>
<td>85-1 mode</td>
</tr>
<tr>
<td></td>
<td>Not provided</td>
<td>85-2 mode</td>
</tr>
<tr>
<td>8086</td>
<td>Provided</td>
<td>86 mode</td>
</tr>
<tr>
<td></td>
<td>Not provided</td>
<td></td>
</tr>
</tbody>
</table>

**Nonvectored Mode.** Figure 7-8 shows a system configuration in the nonvectored mode. In this type of system, software must analyze interrupt sources.

*Figure 7-7. DMA System (Channels A and B in DMA, Vectored Mode)*
**μPD7201A Specifications**

### Absolute Maximum Ratings

- **Supply Voltage, \( V_{CC} \):** -0.5V to +7.0V
- **Input Voltage, \( V_i \):** -0.5V to +7.0V
- **Output Voltage, \( V_o \):** -0.5V to +7.0V
- **Storage Temperature, \( T_{STG} \):** -65°C to +125°C
- **Operating Temperature, \( T_{OPT} \):** 0°C to +70°C

*COMMENT:* Exposing the device to stresses above those listed in Absolute Maximum Ratings could cause permanent damage. The device is not meant to be operated under conditions outside the limits described in the operational sections of this specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

### DC Characteristics

- **\( T_a = 0°C \) to +70°C; \( V_{CC} = +5V \pm 10% \)**

### Capacitance

- **\( T_a = 25°C; \ V_{CC} = \text{GND} = 0V \)**

### AC Characteristics

- **\( T_a = 0°C \) to +70°C; \( V_{CC} = +5V \pm 10% \)**

### AC Waveform Measurement Points

- Notes: 1. **RESET:** Must be active for a minimum of one complete CLK cycle.
- 2. In all modes system clock rate must be 4.5 times data rate.
Timing Waveforms

Read Cycle

Write Cycle

INTA Cycle

DMA Cycle

Transmit Data Cycle

Receive Data Cycle

Notes: © INTA signal acts as RD signal.
© PRE and RAI signals act as CS signal.

Other Timing

Clock

Read/Write Cycle
(Software Block Transfer Mode)

Sync Pulse Generation
(External Sync Mode)
Data transmission by using an interrupt or DMA is simplified.  
0: The same as 7201  
1: TxLR set mode  
To realize this mode operation, these two 16-bit registers are additionally provided.  
• TxLR  
This is a transmit length register. Immediately after the setting of bit D₅ of the CR1 register, a desired number of transmit data characters are written to this register.  
• TxLC (SR3 and SR4)  
This is a transmit length counter which indicates the number of transmit data characters.  
The value of the TxLC is incremented by 1 each time the TxINT or DRQTx signal becomes active and when it matches that of the TxLR, a Tx interrupt or DRQ signal, which generates in response to a Tx empty state, is masked. Once set, this mode will not be cleared until the system is reset.

<table>
<thead>
<tr>
<th>μPD7201A</th>
<th>μPD7201</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) TxLR set mode (bit D₅ of CR1 register = 1)</td>
<td></td>
</tr>
<tr>
<td><strong>CR1</strong></td>
<td><strong>CR1</strong></td>
</tr>
<tr>
<td>D₇ D₆ D₅ D₄</td>
<td>D₇ D₆ D₅ D₄</td>
</tr>
<tr>
<td>TxLR Set</td>
<td>0</td>
</tr>
</tbody>
</table>

This bit is always “0”.
## (2) DMA mode (bit D7, D0 of CR2A register = 1, 1)

<table>
<thead>
<tr>
<th>CR2A</th>
<th>CR2A</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>D2</td>
</tr>
<tr>
<td>INT/DM A</td>
<td>INT/DM A</td>
</tr>
<tr>
<td>0 0 Both CH.INT</td>
<td>0 1 CH.A: DMA, CH.B:INT</td>
</tr>
</tbody>
</table>

- **DMA-1 Mode**
  The same as the DMA mode of the 7201.

- **DMA-2 Mode**
  Priority is not determined in the MPSC.
  The DMA controller, in response to a DMA request, determines the priority.
  The DACK signal controls the CS, C/D, B/A, RD, and WR pins of the MPSC. The DTRB/HAI and DTRA/HAO pins respectively function as DTRB and DTRA pins.

## (3) 85-3 Vectored mode (bits D4, D3 of CR2A register = 1, 1)

<table>
<thead>
<tr>
<th>CR2A</th>
<th>CR2A</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>D5</td>
</tr>
<tr>
<td>INT</td>
<td>INT</td>
</tr>
<tr>
<td>0 0 85-1 Vectored</td>
<td>0 1 85-2 Vectored</td>
</tr>
</tbody>
</table>

**85-3 Vectored Mode**
In this mode the INT signal becomes active when an internal interrupt occurs even if the PRI is 1. Therefore, a daisy-chaining system using the PRI and PRO signals cannot be configured. The μPD7201A can be used as a slave of μPD8259A.

DMA-2 mode is not available in the 7201.
### (4) Masking first Rx character interrupt (bit D6 of CR2A register = 1)

<table>
<thead>
<tr>
<th>CR2</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>B7</td>
<td>B6</td>
<td>B5</td>
<td>B4</td>
</tr>
<tr>
<td>Rx INT Mask</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The first Rx character interrupt can be masked.

- **0**: The same as the 7201
- **1**: The first Rx character interrupt is masked when the INT on first Rx character mode is selected. This mode is not cleared by the enable INT on next Rx character command.

<table>
<thead>
<tr>
<th>CR2</th>
</tr>
</thead>
<tbody>
<tr>
<td>B7</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

This bit must be always "0".

### (5) Reception of 2nd CRC character (HDLC mode)

<table>
<thead>
<tr>
<th>Rx buffer (normal data)</th>
<th>Rx buffer (abnormal data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rx Buffer (normal data)</td>
<td>Rx Buffer (erroneous data)</td>
</tr>
</tbody>
</table>

The character received immediately before a flag (normally, CRC #2) is correctly loaded to the Rx buffer.

The character received immediately before a flag (normally, CRC #2) is not correctly loaded to the Rx buffer.

### (6) Tx interrupt/DRQ signal in Tx buffer empty state (HDLC mode)

When TxLR set bit (bit D6 of the CR1 register) is 1, the Tx interrupt/DRQ signal automatically becomes active in response to issuance of the Tx enable command. Therefore, the first data byte can be transferred to the 7201A by using an interrupt or DMA.

The Tx interrupt/DRQ signal does not become active until the first data byte is written into the 7201. Therefore, the first data byte after a channel has been reset must be written from the CPU to the 7201.

### (7) Abort detection before detecting flag (HDLC mode)

Abort sequences are detected only after a flag has been received. Therefore, if a receive line is in the mark state when Rx has been enabled, this will not be detected as an abort sequence.

An abort sequence is detected even before a flag has been received (during hunt phase) when the Rx has been enabled.
<table>
<thead>
<tr>
<th><strong>μPD7201A</strong></th>
<th><strong>μPD7201</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>(8) Detection of frame transmission completion (HDLC mode)</strong></td>
<td><strong>(8) Detection of frame transmission completion (HDLC mode)</strong></td>
</tr>
<tr>
<td>When transmission of a frame has been completed (i.e., when closing flags are transmitted), the all sent bit (bit D9 of the SR1 register) becomes 1 and an E/S interrupt occurs at the same time.</td>
<td>There is no information indicating completion of frame transmission (i.e., transmission of a closing flag).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>(9) Transmission of sequential frames (HDLC mode)</strong></th>
<th><strong>(9) Transmission of sequential frames (HDLC mode)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>The overhead of the CPU is lessened when a series of frames are transmitted by using DMA mode.</td>
<td>When a series of frames are transmitted by using DMA, the CPU is required to perform such things as issuance of the initialize Tx CRC calculator or Tx underrun/EOM command and managing the timing of the command issuance.</td>
</tr>
<tr>
<td>• Initialize Tx CRC calculator function</td>
<td></td>
</tr>
<tr>
<td>The Tx CRC calculator is automatically reset (all the bits are reset to 1's) when a flag is loaded to the Tx shift register.</td>
<td></td>
</tr>
<tr>
<td>• Reset Tx underrun/EOM function</td>
<td></td>
</tr>
<tr>
<td>When the first character is loaded to the MPSCC, the Tx underrun/EOM bit is automatically reset.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>(10) CRC transmission in Tx underrun (HDLC mode)</strong></th>
<th><strong>(10) CRC transmission in Tx underrun (HDLC mode)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>If the Tx underrun condition occurs when the TxLR set bit (bit D6 of the CR1 register) is 1, an abort sequence or CRC is automatically transmitted.</td>
<td>If the Tx underrun condition occurs before all the data in a frame have been transmitted, a send abort command must be issued.</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This application note describes a uPD7201A application for the design of the hardware of software of a system using the uPD7201A.

Detailed explanations of the uPD7201A functions and operation are omitted. For this information, refer to the other parts of the Product Description and to the Data Book.

The first chapter explains uPD7201A operation from the viewpoint of hardware and software designers for reference in examination or discussion of system configuration. Particularly, actual process flow for data transmission and reception is shown to facilitate understanding of system operation. Chapters 4 and 5 provide configuration examples of systems using the uPD7201A and explain the software required for each.

Figure 1.1 Example of configuration of system using MPSC
Chapter 2

µPD7201A Register Configuration

The µPD7201A contains six control registers (CR0 - CR5), two synchronization pattern registers (CR6 and CR7), one transmit length register (TxLR), and four status registers (SR0, SR1, SR3, and SR4) for channel A and for channel B, and a vector register (SR2B) for channel B.

Table 2.1 lists the names and main functions of the registers.

Table 2.2 lists the structure of each register.

For the bit functions and meanings of the registers, refer to the User's Manual.

Selection of the registers is made by combining the B/Ā, C/Ū, and ČS lines. The registers CR1 to CR7 and SR1 to SR4 are selected by using the CR0 pointer bit. Some registers or register bits are contained only in either channel A or channel B. However, if they are set for the channel in which they are contained, they function commonly to both channels.

Figure 2.1 shows register setting examples to operate channels A and B in Asynchronous mode by using interrupts.
Table 2.1 μPD7201A register configuration

<table>
<thead>
<tr>
<th>Classification</th>
<th>Register name</th>
<th>Channel</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control register</td>
<td>CR0</td>
<td>A, B</td>
<td>Register selecting pointer, CRC logic initialization, various control commands.</td>
</tr>
<tr>
<td></td>
<td>CR1</td>
<td>A, B</td>
<td>Interface mode with CPU. Interrupt/DMA control.</td>
</tr>
<tr>
<td></td>
<td>CR2</td>
<td>A</td>
<td>Pin selection, vector mode selection, interrupt/DMA control.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B</td>
<td>Initial value of interrupt vector.</td>
</tr>
<tr>
<td></td>
<td>CR3</td>
<td>A, B</td>
<td>Receive operation control, Auto Enable mode.</td>
</tr>
<tr>
<td></td>
<td>CR4</td>
<td>A, B</td>
<td>Operation protocol mode.</td>
</tr>
<tr>
<td></td>
<td>CR5</td>
<td>A, B</td>
<td>Send operation control, general-purpose output line control.</td>
</tr>
<tr>
<td></td>
<td>CR6</td>
<td>A, B</td>
<td>SYNC character or secondary station address.</td>
</tr>
<tr>
<td></td>
<td>CR7</td>
<td>A, B</td>
<td>SYNC character or flag.</td>
</tr>
<tr>
<td></td>
<td>TxLR</td>
<td>A, B</td>
<td>Setting of the number of send data words (16 bits).</td>
</tr>
<tr>
<td>Status register</td>
<td>SR0</td>
<td>A, B</td>
<td>E/S bit, send and receive buffer status, interrupt state.</td>
</tr>
<tr>
<td></td>
<td>SR1</td>
<td>A, B</td>
<td>Special Rx Condition source, Residue Code, transmitter status.</td>
</tr>
<tr>
<td></td>
<td>SR2</td>
<td>B</td>
<td>Interrupt vector value.</td>
</tr>
<tr>
<td></td>
<td>SR3, SR4</td>
<td>A, B</td>
<td>TxLC value</td>
</tr>
</tbody>
</table>
### Table 2.2 µPD7201A register configuration

<table>
<thead>
<tr>
<th>Bit Register Name</th>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CRC0</strong></td>
<td></td>
<td>00 Null</td>
<td>00 Null</td>
<td>000 Null</td>
<td>(000 CR0,SRO)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>01 Initialize Rx</td>
<td>CRC Calculator</td>
<td>CMD2</td>
<td>CMD1</td>
<td>CMD0</td>
<td>PTR2</td>
<td>PTR1</td>
<td>PTR0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 Initialize Tx</td>
<td>CRC Calculator</td>
<td>001 Send Abort (HDLC)</td>
<td>010 Reset E/S INT</td>
<td>010 CR2,SR2 (CH.B only)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11 Reset Tx underrun/EOM</td>
<td>Bit</td>
<td>010 CR1,SR1</td>
<td>011 CR3,SR3</td>
<td>100 CR4,SR4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CRC1</strong></td>
<td></td>
<td>Wait Enable</td>
<td>TxRx</td>
<td>Wait on Rx/Tx</td>
<td>Rx INT Mode 1</td>
<td>Rx INT Mode 0</td>
<td>Status Affects</td>
<td>TX INT/ DMA</td>
<td>E/S INT Enable</td>
</tr>
<tr>
<td></td>
<td>Wait</td>
<td>00 ENABLE</td>
<td>0 No operation</td>
<td>0 Tx 1 Rx</td>
<td>00 INT/DMA Disable</td>
<td>01 INT on First CHAR.</td>
<td>10 INT on ALL CHAR. (With Parity)</td>
<td>11 INT on ALL CHAR. (No Parity)</td>
<td>0 Fixed Vector</td>
</tr>
<tr>
<td><strong>CR2A</strong></td>
<td></td>
<td>RTSB/ SYNCB Select</td>
<td>RxINT Mask</td>
<td>Vector Mode</td>
<td>INT Mode 1</td>
<td>INT Mode 0</td>
<td>Priority Select</td>
<td>INT/DMA Mode 1</td>
<td>INT/DMA Mode 0</td>
</tr>
<tr>
<td></td>
<td>0 RTSB</td>
<td>0 Don’t Mask</td>
<td>0 Non-Vectored</td>
<td>085-1 Vectored</td>
<td>0185-2 Vectored</td>
<td>1086 Vectored</td>
<td>1185-3 Vectored</td>
<td>0 TxA—RxB</td>
<td>1 RxB—TxA</td>
</tr>
<tr>
<td></td>
<td>1 SYNCB</td>
<td>1 Mask</td>
<td>1 Vectored</td>
<td>085-1 Vectored</td>
<td>0185-2 Vectored</td>
<td>1086 Vectored</td>
<td>1185-3 Vectored</td>
<td>0 Both CH. INT</td>
<td>01 CH. A: DMA</td>
</tr>
</tbody>
</table>

* 0: RxA TxA RxB TxB E/SA E/SB
  1: RxA RxB TxA TxB E/SA E/SB

<table>
<thead>
<tr>
<th>Bit Register Name</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CR2B</strong></td>
<td>V7</td>
<td>V6</td>
<td>V5</td>
<td>V4</td>
<td>V3</td>
<td>V2</td>
<td>V1</td>
<td>V0</td>
</tr>
</tbody>
</table>

1-70
| Regi
name | Bit   | Mode | D7 | D6 | D5 | D4 | D3 | D2 | D1 | DO |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Rx Bits/CHAR</td>
<td>Rx Bits/CHAR</td>
<td>Auto Enable</td>
<td>Enter Hunt Phase</td>
<td>Rx CRC Enable</td>
<td>Address Search Mode</td>
<td>SYNC CHAR Load Inhibit</td>
<td>Rx Enable</td>
</tr>
<tr>
<td>CR3</td>
<td>Async</td>
<td>Clock Rate 1</td>
<td>Clock Rate 0</td>
<td>SYNC Mode 1</td>
<td>SYNC Mode 0</td>
<td>Stop Bits 1</td>
<td>Stop Bits 0</td>
<td>Parity Even/Odd</td>
<td>Parity Enable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>00 5 Bits/CHAR</td>
<td>00 Disable 1 Enable</td>
<td>0 No 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0 Disable 1 Enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>01 7 Bits/CHAR</td>
<td>0 Nop 1 Re-enable</td>
<td>0 Disable 1 Enable</td>
<td>0 Nop 1 Inhibit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11 8 Bits/CHAR</td>
<td>0 Nop 1 Available</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDLC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR4</td>
<td>Async</td>
<td>Clock Mode 1</td>
<td>Clock Mode 0</td>
<td>00 01 1 Stop bit/CHAR</td>
<td>01 1 1/2 Stop bit/CHAR</td>
<td>0 Even 1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00 x 1 Clock Mode</td>
<td>00</td>
<td>01 1 Stop bit/CHAR</td>
<td>0 Even 1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>01 x 16 Clock Mode</td>
<td>10 1 1/2 Stop bit/CHAR</td>
<td>0 Even 1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 x 32 Clock Mode</td>
<td>11 2 Stop bit/CHAR</td>
<td>0 Even 1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11 x 64 Clock Mode</td>
<td>11 2 Stop bit/CHAR</td>
<td>0 Even 1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>00 8 bit SYNC</td>
<td>00 SYNC Mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDLC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 HDLC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR5</td>
<td>Async</td>
<td>DTR</td>
<td>Tx Bits/CHAR</td>
<td>Tx Bits/CHAR</td>
<td>Send Break</td>
<td>Tx Enable</td>
<td>CRC-16/CCIT</td>
<td>RTS</td>
<td>Tx CRC Enable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 DTR=1</td>
<td>00 5 or less Bits/CHAR</td>
<td>00 5 or less Bits/CHAR</td>
<td>0 Marking 1 Spacing</td>
<td>0 Disable 1 Enable</td>
<td>0</td>
<td>0 RTS=1 (when All Sent)</td>
<td>1 RTS=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 DTR=0</td>
<td>01 7 Bits/CHAR</td>
<td>01 7 Bits/CHAR</td>
<td>01 7 Bits/CHAR</td>
<td>01 7 Bits/CHAR</td>
<td>0</td>
<td>0 RTS=1 (when All Sent)</td>
<td>1 RTS=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 6 Bits/CHAR</td>
<td>10 6 Bits/CHAR</td>
<td>10 6 Bits/CHAR</td>
<td>10 6 Bits/CHAR</td>
<td>10 6 Bits/CHAR</td>
<td>1</td>
<td>0 RTS=1 (when All Sent)</td>
<td>1 RTS=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11 8 Bits/CHAR</td>
<td>11 8 Bits/CHAR</td>
<td>11 8 Bits/CHAR</td>
<td>11 8 Bits/CHAR</td>
<td>11 8 Bits/CHAR</td>
<td>1</td>
<td>0 RTS=1 (when All Sent)</td>
<td>1 RTS=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDLC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR6</td>
<td>Sync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>D7</td>
<td>D6</td>
<td>D5</td>
<td>D4</td>
<td>D3</td>
<td>D2</td>
<td>D1</td>
<td>D0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Monosync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bisync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXT sync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDLC</td>
<td>Secondary Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1-71
<table>
<thead>
<tr>
<th>Register name</th>
<th>Bit Mode</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SYNC Bit 15 - 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Async not used</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Monosync Rx SYNC CHAR</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Bisync SYNC CHAR. (Bit 15 - 8)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>EXT sync not used</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HDLC Flag (01111110)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit Register name</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>TxLR-L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tx Length Register bit 7 - 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit Register name</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>TxLR-H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tx Length Register bit 15 - 8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Register name</th>
<th>Bit Mode</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Break/Abort</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Tx Under-run/EOM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CTS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SYNC/ Hunt</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DCD</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Tx Buffer Empty</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>INT Pending</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Rx CHAR Available</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>As SYN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Break Detected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Normal</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 CTS=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 CTS=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 SYNC=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 SYNC=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 DCD=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 DCD=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Tx Buff Empty</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Tx Buff Full</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 INT Pending</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 No INT Pending</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Rx CHAR Available</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Rx Buff. Empty</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>EXT sync</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Tx Under-run</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/EOM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Not Tx Underrun</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Hunt Phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Exit Hunt Phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Tx Buff Empty</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Tx Buff Full</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 INT Pending</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 No INT Pending</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Rx CHAR Available</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Rx Buff. Empty</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HDLC</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Abort Detected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Normal</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Tx Under-run</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/EOM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Not Tx Underrun</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Hunt Phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Exit Hunt Phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Tx Buff Empty</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Tx Buff Full</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 INT Pending</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 No INT Pending</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Rx CHAR Available</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Rx Buff. Empty</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1-72
### Register bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### Register Mode

**Async Mode**

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Sync Mode**

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### SR1 Register Mode

<table>
<thead>
<tr>
<th>End of Frame</th>
<th>CRC/ Framing Error</th>
<th>Rx/ Overrun Error</th>
<th>Parity Error</th>
<th>Residue Code 2</th>
<th>Residue Code 1</th>
<th>Residue Code 0</th>
<th>All Sent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Async</td>
<td>0</td>
<td>1 Framing Error</td>
<td>0 No Error</td>
<td>1 Parity Error</td>
<td>0 No Error</td>
<td>Unknown</td>
<td>1 All Sent</td>
</tr>
<tr>
<td>Sync</td>
<td>1 CRC Error</td>
<td>0 No Error</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Not All Sent</td>
</tr>
<tr>
<td>HDLC</td>
<td>1 End of Frame</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 All Sent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When no residue exists</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Not All Sent</td>
</tr>
</tbody>
</table>

### SR2B Register Mode

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### SR3 Register Mode

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### SR4 Register Mode

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
E STNO ADRS OBJECT M SOURCE STATEMENTS

0001 :*******************************************************************************
0002 : UPD7201 REGISTER SETTING EXAMPLE *
0003 :*******************************************************************************
0004 :
0005 : COMMON REGISTER INITIALIZE
0006 :
0007 0000 3E18 MVI A.18H :CHANNEL RESET
0008 0002 D3F1 OUT CHACT
0009 0004 D3F3 OUT CHBCT
0010 0006 3E02 MVI A.2H :PTR 2
0011 0008 D3F1 OUT CHACT
0012 000A D3F3 OUT CHBCT
0013 000C 3E00 MVI A.CR2A :BUS I/F MODE
0014 000E D3F1 OUT CHACT
0015 0010 3E00 MVI A.CR2B :INTERRUPT VECTOR
0016 0012 D3F3 OUT CHBCT
0017 :
0018 : CH-A OPERATION MODE
0019 :
0020 0014 3E04 STCHA: MVI A.4H :PTR 4A
0021 0016 D3F1 OUT CHACT
0022 0018 3E4F MVI A.CR4A :CH-A OPERATION MODE
0023 001A D3F1 OUT CHACT
0024 001C 3E11 MVI A.11H :PTR 1A.RESET E/S INT
0025 001E D3F1 OUT CHACT
0026 0020 3E13 MVI A.CR1A :INT/DMA MODE
0027 0022 D3F1 OUT CHACT
0028 0024 3E03 MVI A.3H :PTR 3A
0029 0026 D3F1 OUT CHACT
0030 0028 3E41 MVI A.CR3A :RX PARAMETERS
0031 002A D3F1 OUT CHACT
0032 002C 3E05 MVI A.5H :PTR 5A
0033 002E D3F1 OUT CHACT
0034 0030 3E28 MVI A.CR5A :TX PARAMETERS
0035 0032 D3F1 OUT CHACT
0036 :
0037 : CH-B OPERATION MODE
0038 :
0039 0034 3E04 STCHB: MVI A.4H :PTR 4B
0040 0036 D3F3 OUT CHBCT
0041 0038 3E4F MVI A.CR4B :CH-B MODE
0042 003A D3F3 OUT CHBCT
0043 003C 3E11 MVI A.11H :PTR 1B.RESET E/S INT
0044 003E D3F3 OUT CHBCT
0045 0040 3E1F MVI A.CR1B :INT/DMA MODE
0046 0042 D3F3 OUT CHBCT
0047 0044 3E03 MVI A.3H :PTR 3B
0048 0046 D3F3 OUT CHBCT
0049 0048 3E41 MVI A.CR3B :RX PARAMETERS
0050 004A D3F3 OUT CHBCT
0051 004C 3E05 MVI A.5H :PTR 5B
0052 004E D3F3 OUT CHBCT

Figure 2.1 μPD7201A register setting example 1
Figure 2.1 μPD7201A register setting example 2
Chapter 3 deals with the features of the send and receive data transfer method (polling, interrupt, DMA), then explains the \( \mu \text{PD7201A} \) interrupt and DMA operation control method in detail. In addition, the interrupt and DMA functions are explained to facilitate understanding of the \( \mu \text{PD7201A} \) operation.

3.1 Send and Receive Data Transfer Method

The \( \mu \text{PD7201A} \) supports three types of send and receive data transfer methods.* This section explains the differences and method of selection.

* The transfer method mentioned here applies to a write or read of send and receive data between the CPU and the \( \mu \text{PD7201A} \) or between system memory and the \( \mu \text{PD7201A} \).

3.1.1 Transfer under program control

When data is transferred to and from peripheral devices under program control, normally the polling method or interrupt method is used.

- **Polling method:** The CPU continuously checks peripheral device status, and when ready, data is transferred.

- **Interrupt method:** The peripheral device issues an interrupt request to the CPU (or interrupt controller). Data is transferred under control of the interrupt processing program.
(1) Data transfer by polling

The CPU checks theμPD7201A status for send or receive data transfer timing.

Send data write (CPU → μPD7201A) occurs when the Tx Buffer Empty bit (SR0 bit D2) is set to 1.

Receive data read (CPU ← μPD7201A) occurs when the Rx Character Available bit (SR0 bit D0) is set to 1.

When data is received, the error status can be checked by the SR1 error bit.

Normally, these operations are repeated until all send and receive data transfers are terminated.

Figures 3.1 and 3.2 show examples of data transfer programs using polling.
Figure 3.1 Example of send data transfer program using polling

```
LXI H, TXBUF
MVI B, COUNT1

LOOP1: IN SR0
ANI 4
JZ LOOP1
MOV A, M
OUT CHADT
INX H
DCR B
JNZ LOOP1
```

...
Figure 3.2 Example of send data transfer program using polling

START

Set receive data store address

Set the number of receive data bytes

Read SRO

Rx Character Available = "1"?

Y

Read receive data

Store receive data

Update address

Update the number of transfer bytes

N

Is transfer terminated?

Y

END

LXI H, RXBUF
MVI B, COUNT2

LOOP 2: IN SR0
PRC
JNC LOOP2
IN CHADT
MOV M, A
INX H
DCR B
JNZ LOOP2

...
(2) Data transfer by using interrupt

When Tx Buffer Empty or Rx Character Available state occurs, the \texttt{INT} signal goes LOW (active) and an interrupt request is sent to the CPU or interrupt controller. This \textmu PD7201A function can be used to determine the send data write or receive data read timing for the \textmu PD7201A.

Normally, the CPU executes the main program. When an interrupt request occurs, the CPU temporarily stops main program processing and executes the interrupt processing program. The interrupt processing program first saves the current state of each register in the stack area, then performs processing on the peripheral device which requested the interrupt. When processing is terminated, the state of each register saved at the beginning of processing is restored and another interrupt is ready to be accepted, then main program processing is restarted.

Since this method eliminates the need for the CPU to always monitor the \textmu PD7201A status and the CPU needs only to be involved when an interrupt request occurs, CPU overhead is minimized for more programming efficiency.

Figure 3.3 shows an example of a program using an interrupt to send data.

Since the \textmu PD7201A has two interrupt modes, vector mode and nonvector mode, interrupt processing can be performed by the method most suitable for the CPU being used.

The \textmu PD7201A interrupt mode and interrupt operation are explained in 3.2.
Figure 3.3 Example of send data transfer program using interrupts
3.1.2 Transfer using DMA

In a transfer under program control (transfer via interrupt), the time during which the CPU reads instructions and checks peripheral device status is required in addition to the time required for actual data transfer between memory and the peripheral device. Thus, a fairly long time is generally required to transfer one data word.

Thus, if only data transfers between memory and a peripheral device occur, the transfer rate can be dramatically increased over program-controlled transfers. Such direct data transfer between memory and peripheral devices without CPU intervention is called direct memory access (DMA). It is used for high-speed data transfer. Normally, a DMA operation is executed by a DMA controller (such as the µPD8237A-5).

When the µPD7201A is placed in Tx Buffer Empty or Rx Character Available states, the DMA request signal goes HIGH (active) and a DMA transfer request is sent to the DMA controller. There are four DMA request signals: DRQRxA, DRQTxA, DRQRxB, and DRQTxB.

The µPD7201A DMA operation is explained in 3.3.
3.2 μPD7201A Interrupt Operation

3.2.1 μPD7201A interrupt mode

The μPD7201A contains two types of interrupt modes (vector mode and noninterrupt mode) according to the difference in how interrupt vectors are received. Whether the vector contents are automatically changed according to the interrupt source or are fixed can be specified by using CR1B bit D2.

In vector mode, when an interrupt source occurs, the μPD7201A asserts the INT line (LOW) and outputs an interrupt vector in synchronization with the INTAK signal output from the CPU (called an INTAK sequence). In the INTAK sequence, the CS line must be held HIGH. Figure 3.4 shows the interrupt timing in vector mode.

Since an interrupt vector is automatically generated in vector mode, the required processing program should be stored at the jump destination address determined by the μPD7201A for efficient programming.

![Figure 3.4 Interrupt timing in vector mode (85-1 mode)](image-url)
In nonvector mode, when an interrupt source occurs, the μPD7201A asserts the INT line (LOW). When the CPU accepts the interrupt request, it reads the μPD7201A SR2B and determines the interrupt source. (If necessary, it also reads SR0 and SR1 to determine the interrupt source.) In this mode, the INTAK line is not used, but must be pulled up.

Figure 3.5 shows the interrupt timing in nonvector mode.
Figure 3.6 shows an example of an interrupt processing sequence in nonvector mode.

Figure 3.6  Interrupt processing in nonvector mode
Table 3.2 lists the relationship between the control registers related to each \(\mu PD7201A\) interrupt mode and output vectors.

**Table 3.2 \(\mu PD7201A\) interrupt mode**

<table>
<thead>
<tr>
<th>CR2A</th>
<th>CR1B, D2</th>
<th>Interrupt mode</th>
<th>Vector read method</th>
<th>INTA</th>
<th>SR2B</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Non Vectored 85 (Fixed Vector)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Non Vectored 85 (Modified Vector)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Non Vectored 86 (Fixed Vector)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Non Vectored 86 (Modified Vector)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>85-1 Vectored (Fixed Vector)</td>
<td>First time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>85-1 Vectored (Modified Vector)</td>
<td>First time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>85-2 Vectored (Fixed Vector)</td>
<td>First time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>85-2 Vectored (Modified Vector)</td>
<td>First time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>86 Vectored (Fixed Vector)</td>
<td>First time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>86 Vectored (Modified Vector)</td>
<td>First time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>85-3 Vectored (Fixed Vector)</td>
<td>First time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>85-3 Vectored (Modified Vector)</td>
<td>First time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note) Such indication as \(\uparrow\), \(\downarrow\), or \(\Rightarrow\) means that the value is automatically changed according to the interrupt source. In other vectors, the value set in CR2B is output as is.

- : Not used. ○: Used.
3.2.2 μPD7201A interrupt sources

The μPD7201A can detect various interrupt sources so that the CPU can perform appropriate processing. The interrupt sources are roughly classified into the following three types:

a) Transmit interrupt (Tx interrupt)
b) Receive interrupt (Rx interrupt)
c) External/Status interrupt (E/S interrupt)

This section explains each of these interrupt sources, the causes of each, and the interrupt service processing required.

(1) Transmit interrupt

A Tx interrupt occurs when Tx Buffer Empty state (SR0 bit D2 = 1) is set during INT/DMA Enable (CR1 bit D1 = 1). This interrupt indicates that the μPD7201A transmit buffer is empty and the CPU may write send data. In Synchronous or HDLC mode, the Tx interrupt also occurs when CRC character sending is terminated. Note that in the HDLC mode, the Tx interrupt occurrence condition in the TxLR Set mode (CR1 bit D6 = 1) differs from that in other cases.

The Tx interrupt source is cleared when send data is written or a Reset Tx INT/DMA Pending command is executed. However, the Reset Tx INT/DMA Pending command must be issued in the Tx Buffer Empty mode.

When DMA transfer is selected, a Tx interrupt of the channel is disabled internally, thus send data cannot be transferred via a DMA and an interrupt at the same time.
See Figure 3.3 for the send data transfer method using an interrupt.

(2) Receive interrupt

During Rx INT Enable, when received data is transferred to the receive buffer and the Rx Character Available mode is in effect, an Rx interrupt occurs. The mode to enable a receive interrupt is INT On First Character (First Rx INT) or INT on All Character. It is selected by using CR1 bits D4 and D3. First Rx INT in the INT on First Character mode can be masked by setting the Rx INT Mask bit (CR2A bit D6) to 1.

Special receive interrupt is Special Rx Condition interrupt. The Special Rx Condition interrupt occurs when receive data is erroneous (Parity, Overrun, or Framing Error). In HDLC mode, an interrupt also occurs at the end of a frame (End of Frame). The Special Rx Condition interrupt has the same priority level as an Rx interrupt.

Both Special Rx Condition and Rx interrupts do not occur at the same time for the same receive data. When one receive data word causes Special Rx Condition, only Special Rx Condition interrupt occurs, thus the receive data must be read within the processing routine. In the Special Rx Condition interrupt routine, when End-of-Frame occurs in the HDLC mode, the second byte of the CRC character is read.

The Special Rx Condition status is held in SR1, and can be checked by the CPU.
When the Special Rx Condition occurs, the Error Reset command must always be issued.

Table 3.3 lists the receive interrupt mode.

Table 3.4 lists the Special Rx Condition interrupt operation.

Table 3.3 Receive interrupt mode and operation

<table>
<thead>
<tr>
<th>CR1</th>
<th>Receive interrupt mode</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>D4 D3</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>INT/DMA Disable</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>INT on First Character</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>INT on All Character (parity error is used as Special Rx Condition source)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>INT on All Character (parity error is not used as Special Rx Condition source)</td>
</tr>
</tbody>
</table>
Table 3.4 Special Rx Condition interrupt operation

<table>
<thead>
<tr>
<th>Source</th>
<th>Operation mode</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parity Error</td>
<td>Asynchronous Synchronous</td>
<td>When a Parity Error occurs. This bit is latched. It is reset when the Error Reset command is issued.</td>
</tr>
<tr>
<td>(SR1: D4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overrun Error</td>
<td>Asynchronous Synchronous HDLC</td>
<td>When an Overrun Error occurs. This bit is latched. It is reset when Error Reset command is issued.</td>
</tr>
<tr>
<td>(SR1: D5)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Framing Error</td>
<td>Asynchronous</td>
<td>When a Framing Error occurs. This bit is not latched. It is reset when next correct data is received, and also reset when an Error Reset command is issued.</td>
</tr>
<tr>
<td>(SR1: D6)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End of Frame</td>
<td>HDLC</td>
<td>When valid end flag is received. This bit is latched. It is reset when an Error Reset command is issued.</td>
</tr>
<tr>
<td>(SR1: D7)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note (1) When an error status check is made, the status must be first read before receive data is read.

(2) If the First Rx INT mode (CR1 bits D4 and D3 = 0, 1) is selected, when a Special Rx Condition interrupt occurs, the data subsequently input is not transferred to the last stage of the receive buffer until the Error Reset command is issued. (Up to two characters of data received after the Special Rx Condition interrupt occurs are stored in the first and second stages of the receive buffer.)

Figure 3.7 shows an example of a data transfer program using an Rx interrupt. Figure 3.8 shows an example of Special Rx Condition interrupt processing.
Figure 3.7 Example of receive data transfer program using interrupts

```
PUSH PSW
PUSH H
PUSH B
LHLD RXPTR
LDA RXCNT
MOV B, A
IN DATA
MOV M, A
INX H
SHLD RXPTR
DCR B
MOV A, B
STA RXCNT
MVI A, 38H
OUT CTRLA
JZ RXTERM Note)
POP B
POP H
POP PSW
EI
RET

Note)
RXTERM: Receive transfer termination processing
```
Figure 3.8 Example of Special Rx Condition interrupt processing

(3) External/Status interrupt latch operation

The External/Status (E/S) interrupt is used to detect the µPD7201A input status or change in internal state. The E/S interrupt occurs under the conditions listed in Table 3.5 if E/S INT Enable is set (CRI bit D0 = 1).
Table 3.5  E/S interrupt occurrence conditions

<table>
<thead>
<tr>
<th>Source (E/S bit)</th>
<th>Operation mode condition</th>
<th>E/S interrupt occurrence condition</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Asynchronous</td>
<td>Synchro-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>nous</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>HDLC</td>
<td></td>
</tr>
<tr>
<td>CTS</td>
<td>0</td>
<td>0</td>
<td>0 → 1, 1 → 0</td>
</tr>
<tr>
<td>DCD</td>
<td>0</td>
<td>0</td>
<td>0 → 1, 1 → 0</td>
</tr>
<tr>
<td>SYNC/HUNT</td>
<td>0</td>
<td>0</td>
<td>0 → 1, 1 → 0</td>
</tr>
<tr>
<td>Break/Abort</td>
<td>0</td>
<td></td>
<td>0 → 1, 1 → 0</td>
</tr>
<tr>
<td>Tx Underrun/EOM</td>
<td></td>
<td>0</td>
<td>0 → 1</td>
</tr>
<tr>
<td>All Sent</td>
<td></td>
<td></td>
<td>0 → 1</td>
</tr>
</tbody>
</table>

Note) Refer to the User's Manual for occurrence of each stage change.

The points to use the E/S interrupt are explained below:

When the state of any E/S bit changes if E/S INT Enable is set (the occurrence condition listed in Table 3.5 is satisfied and the preceding E/S interrupt processing must be terminated), the state of the other E/S bits is also latched, the values are set in SR0, and and E/S interrupt occurs. Normally, the CPU determines of the source E/S interrupt by comparing the new SR0 contents with the SR0 contents before latching the E/S interrupt handling routine, and executes interrupt service processing accordingly. Note that once an E/S bit is latched, the μPD7201A does not execute a new latch operation even if the E/S bit state changes afterwards unless the Reset E/S INT command is issued. (No E/S interrupt occurs.) Figure 3.9 shows an example operation when the $CTS$ and $DCD$ lines change state.
First, when the CTS line goes LOW, the change of state is latched in SRO bit D5 and an E/S interrupt occurs. The CPU accepts the interrupt and begins interrupt processing (1 in Figure 3.9). Even if DCD goes LOW while the E/S interrupt is being processed, the Reset E/S INT command is not issued for a CTS bit latch operation at the time; therefore, the DCD change of state is ignored (2 in Figure 3.9). When the CTS state change processing continues and the Reset E/S INT command is issued (3 in Figure 3.9), the E/S bit latch operation is enabled again. This means that the DCD change of state is first latched in the DCD bit at the time of 4 in Figure 3.9.

The operation mentioned above also occurs when E/S INT Disable is set. Thus, it becomes difficult to recognize E/S bit change of-state timing if the E/S interrupt is not used. In such a case, the most recent E/S bit state can be known by issuing the Reset E/S INT command to the µPD7201A, then reading SRO.

Figure 3.10 shows an example of E/S interrupt processing.

![Figure 3.9 E/S bit latch operation](image-url)
Figure 3.10 E/S interrupt processing example
Figure 3.11 summarizes the μPD7201A interrupt configuration.

Figure 3.11 μPD7201A interrupt configuration

3.2.3 Precautions on use of interrupts

(1) External circuit must be provided to use the μPD7201A as an interrupt controller μPD8259A-2 slave in the 85-2 or 86 vector mode. If the PRI line is not LOW, the μPD7201A INT signal does not become active even if an interrupt condition occurs. Therefore, the PRI line must normally be held LOW.

However, when the μPD8259A-2 accepts any interrupt other than μPD7201A interrupts, PRI must be driven HIGH in the INTAK sequence to inhibit interrupt vector output from the μPD7201A.
Therefore, an external control circuit must be provided to perform operation control as mentioned above by using the \( \mu P D 8 2 5 9 A-2 \) CAS signal.

To perform such control, although no standards are set, it is necessary to use the setup time (minimum of 280 ns) to change the \( \overline{P R I} \) signal state from LOW to HIGH at the falling edge of the second \( \overline{I N T A K} \) signal, as shown in Figure 3.12 (a).

![Figure 3.12 (a) Relation between \( \overline{P R I} \) and \( \overline{I N T A K} \)]

Figure 3.12 (b) shows an example of an external circuit for reference when the method as mentioned above is used (in the 86 vector mode).

If the 85-3 vector mode is used, the \( \overline{I N T} \) signal goes active when an interrupt source occurs independently of the \( \overline{P R I} \) line state, and the aforementioned external circuit is not needed. The \( \overline{P R I} \) line can be controlled by directly decoding the \( \mu P D 8 2 5 9 A-2 \) CAS signal. However, in this mode, a daisy chain mode system using \( \overline{P R I} \) and \( \overline{P R O} \) cannot be implemented.
(2) To perform interrupt control by using μPD7201A PRI input, obey the following caution:

In the system configuration in which an interrupt request issued from another device is accepted during μPD7201A interrupt processing, issue two EOI commands in the sequence shown in Figure 3.12 (c). In this case, interrupt must be disabled in the host when the EOI command is being issued.

Such processing is not required if the system configuration enables multiinterrupts.
Figure 3.12 (c) Multiinterrupt processing sequence
3.3 μPD7201A DMA Operation

This section explains μPD7201A DMA operation (send DMA and receive DMA) using HDLC operation as an example.

3.3.1 Send DMA

When send data is transferred via DMA, CR2A bits D1 and D0 = 0, 1, 1, 0, or 1, 1 must be selected and CR1 bit D1 must be set to 1. A send interrupt on the channel where DMA transfer is selected will not occur.

Send DMA operation is explained by using HDLC mode as an example. The DMA operation in the HDLC mode varies depending on the value of TxLR (CR1 bit D6), as explained below:

(1) When TxLR Set = 1

This mode is a new function added to the μPD7201A. Figure 3.13 shows the timing chart of send DMA operation when the function is used (only minimum required operation is shown).

In the explanation to follow, Figure 3.13 is used.

In this mode, when Tx Enable is set to 1 after initialization ends (assuming that Auto Enable mode is not entered), the DRQTx signal goes HIGH and the DMA controller is requested to transfer send data (2 in Figure 3.13). As requested, the DMA controller transfers send data. After this, the same operation is repeated. When the DMA controller transfers the last data and stops operation (3 in Figure 3.13), then send data is not transferred. Thus, after the last data is sent, the μPD7201A is placed in Tx
Underrun state and an E/S interrupt occurs (5 in Figure 3.13). The data transferred by the DMA controller is counted each time in the µPD7201A (the result is held in TxLC). When the count reaches the value set in TxLR, then the DRQTx signal does not go active even if Tx Buffer Empty is set (4 in Figure 3.13). This state is continued until TxLR is again set. In addition, a comparison between the TxLR and TxLC values is made in the Tx Underrun state. If they match, CRC is automatically sent; if they do not match, an abort is automatically sent. Therefore, when the Tx Underrun state occurs, software must be used to read the TxLC value (SR3, SR4) and decide whether or not the normal frame end is reached.

When the µPD7201A ends sending at least one end flag, All Sent is set to 1 and an E/S interrupt occurs (6 in Figure 3.13). Normally, after the interrupt occurs, the next processing step (such as successive frame sending or send stopping) is executed.

Figure 3.14 shows an example of control when one frame of send data is transferred in the mode.
Figure 3.13 µPD7201A send DMA operation
(HDLC mode, TxLR Set = 1)

1. Tx Enable is set to 1
2. First DMA request by setting Tx Enable to 1
3. Transfer of last data
4. It does not go HIGH because TxLR = TxLC
5. E/S INT by setting Tx Underrun/EOM to 1
6. E/S INT by setting All Sent to 1

Figure 3.14 (a) Control example of send operation by DMA
(main program)
E/S interrupt processing ... Tx Underrun and All Sent processing

Save registers

Issue Reset E/S INT command

What is source?

All Sent

Tx Underrun

Is normal Tx Underrun?

N

Issue Reset Tx INT/DMA Pending command

Y

Perform send error processing

Disable send

Issue EOI command

Restore registers

EI

RET

Note) Interrupt processing caused by other E/S sources is omitted.

Figure 3.14 (b) Control example of send operation by DMA (part of E/S interrupt processing)
(2) When TxLR Set = 0

In this mode, similar operation to μPD7201A operation is performed except that All Sent E/S interrupt occurs after the frame end. Figure 3.15 shows timing chart in this mode (only minimum required operation is shown).

In the explanation to follow, Figure 3.15 is used.

In the mode, the first send data must be written by the CPU (① in Figure 3.15). After this, when Tx Buffer Empty is set, the DRQTx signal first goes HIGH and the DMA controller is requested to transfer send data (③ in Figure 3.15). Assuming that the total number of send data words in N, the number of data words transferred by DMA equals N - 1.

When the DMA controller transfers the last data to the μPD7201A and stops operation (④ in Figure 3.15), then send data is not transferred. Thus, after the last data is sent, Tx Underrun state is set and an E/S interrupt occurs (⑦ in Figure 3.15).

When Tx Buffer Empty is set after the last data is written, the DRQTx signal goes HIGH (⑤ in Figure 3.15). The signal can be reset by issuing Reset Tx INT/DMA Pending command before CRC character is loaded internally (during Tx Buffer Empty) (⑥ in Figure 3.15). If the Reset Tx INT/DMA Pending command is not issued, the DRQTx signal remains HIGH (dotted-line portion in Figure 3.15), but does not affect the send operation.

Note that if send data is transferred to the μPD7201A in the interval of ⑥, frame end is not normally made.
To send continuous frames, if the next send data is transferred after the \( B \) state is set, the next frame can be sent successively without destroying the preceding frame. However, since the \( \mu PD7201A \) generates an E/S interrupt when All Sent is set to 1, the extra CPU load can be eliminated if the interrupt is waited before necessary processing is performed. In this case, even if send stop (Tx Disable) is set, the DRQTx signal is not reset, but if Tx INT/DMA Disable (CR1 bit D1 = 0) or the Reset Tx INT/DMA Pending command is issued, the signal can be reset.

![Diagram of \( \mu PD7201A \) send DMA operation](image)

1. First send data write
2. Reset Tx Underrun/EOM bit
3. First DRQTx signal by setting Tx Buffer Empty to 1
4. Last data transfer
5. Tx Buffer Empty set after last data is written
6. Reset Tx INT/DMA Pending
7. E/S INT by setting Tx Underrun/EOM to 1

Figure 3.15 \( \mu PD7201A \) send DMA operation
(HDLC mode, TxLR = 0)
3.3.2 Receive DMA

To transfer receive data by using DMA, CR2A bits D1 and D2 is set to 0, 1, 1, 0, or 1, 1 and CR1 bits D4 and D3 is set to 0, 1 (First Rx INT mode). In the First Rx INT mode, an Rx interrupt caused by the first receive data can be masked by setting CR2A bits D6 to 1 (Rx INT Mask mode). This mode is used when an interrupt need not be caused by the first receive data (normal address field) or when the baud rate is fast and it is not timely if the DMA controller is enabled by using the First Rx INT handling routine. (Use of the mode is recommended to initiate a receive DMAS.)

Figure 3.16 shows the timing chart of receive DMA operation in the First Rx INT Mask mode, using the HDLC mode as an example (only minimum required operation is shown).
Rx Character Available (SRO:DO)

End of Frame (SR1:D7)

DMRx signal

INT signal

DMA controller enable/disable

RD by DMA controller

WR by CPU (for µPD7201A)

1. DMA request by setting Rx Character Available to 1
2. Receive data transfer
3. EOF is set to 1 when end flag is detected
4. DMA controller operation stops when the last data is transferred
5. EOF state is cleared

Figure 3.16 µPD7201A receive DMA operation (HLDC, First Rx INT Mask mode)
In the explanation to follow, Figure 3.16 is used.

The µPD7201A receives data after the flag is detected. When Rx Character Available state is set, the DRQRx signal is driven HIGH and a request is made to the DMA controller to transfer receive data (1 in Figure 3.16). As requested, the DMA controller transfers receive data (2 in Figure 3.16). The µPD7201A generates DMA requests for all receive data containing a CRC.

When the µPD7201A detects the end flag, End of Frame is set and Special Rx Condition interrupt occurs (3 in Figure 3.16). This interrupt causes the CPU to detect the receive End of Frame. When the DMA controller transfers the last data (the second byte of the CRC character), operation stops (4 in Figure 3.16). After the End of Frame state occurs, the Error Reset command must be issued to the µPD7201A to clear the state (5 in Figure 3.16). Note the following: If the Error Reset command is not issued, only two bytes of data of the next frame successively input in the Int on First Rx Character mode are stored in the FIFO, and it is not transferred to the last stage of the buffer. Thus, when continuous frames are received (particularly when the end flag is also used for the start flag), limitations are imposed on CPU processing time. (For example, when the end flag is also used for the start flag, it is desirable to issue the Error Reset command within the time of two bytes after End of Frame occurs).

If frame reception is not made after reception of one frame ends, necessary processing such as inhibition of reception is performed.
Figure 3.17 shows a control example when one frame of receive data is transferred via DMA.

Figure 3.17 (a) Control example of receive operation via DMA (main program)

Figure 3.17 (b) Control example of receive operation via DMA (Special Rx Condition interrupt processing)
3.3.3 Caution on use of DMA

There are three μPD7201A DMA modes among which appropriate one can be selected according to the system configuration. The modes differ slightly in use of TxLR Set mode and HAKI pin, as listed in Table 3.6.

Basically, only the combination listed in Table 3.6 can be used. If the μPD7201 is replaced with the μPD7201A in a system using the μPD7201, precautions on one use of μPD7201 must be followed when a change cannot be made to the combination listed in Table 3.6. (For details, refer to the μPD7201 User's Manual or μPD7201 Application Note.)

<table>
<thead>
<tr>
<th>Condition</th>
<th>DMA mode setting</th>
<th>Valid DRQ signal</th>
<th>TxLR Set mode setting</th>
<th>Use of HAKI pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel A</td>
<td>CR2A bits D1 and D0 = 0,1</td>
<td>DRQtxA</td>
<td>As desired</td>
<td>Used</td>
</tr>
<tr>
<td>DMA mode</td>
<td></td>
<td>DRQrxA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Both Ch.</td>
<td>Both Ch. DMA-1</td>
<td>DRQtxA, DRQtxB</td>
<td>1</td>
<td>Used</td>
</tr>
<tr>
<td>channel DMA mode</td>
<td></td>
<td>DRQrxA, DRQrxB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Both Ch.</td>
<td>Both Ch. DMA-2</td>
<td>DRQtxA, DRQtxB</td>
<td>As desired</td>
<td>Not used</td>
</tr>
<tr>
<td>DMA mode</td>
<td></td>
<td>DRQrxA, DRQrxB</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This chapter explains a system using the μPD7201A. The hardware in this example is designed as a means to implement the operation conditions assumed, and serves as a reference for the design of a system using the μPD7201A.

4.1 System Configuration

The μPD7201A is applicable to various serial data communication protocols. Here, assume the following specifications:

Transmission format: Full or half duplex
Applicable protocol: Asynchronous, synchronous or HDLC
Baud rate: 200 Kbps, MAX.

Various methods are designed to implement the specifications. Here, the μPD8085AH, which is a typical eight-bit microprocessor, is used as the CPU. The system configuration is based on this processor.

(1) Memory capacity

16K-byte ROM is used for the system program (monitor program area) and μPD7201A control program area. 8K-byte RAM is used for the data buffer and working area. (The total memory capacity is 24K bytes.)

This memory capacity is set here for convenience. In fact, the memory capacity must be determined in accordance with the system.

(2) Transfer method of send and receive data

Polling, interrupts, and DMA can be used. Here, both interrupts and DMA can be used for both A and B channels.
Generally, the entire system configuration varies depending on what data transfer method is used. It must be determined from an overall viewpoint rather than from only the baud rate.

(3) Interrupt mode

To and simplify processing and make the most efficient use of µPD7201A program control, vector processing in Modified Vector mode and nonvector mode processing can be implemented in hardware.

(4) Others

A baud rate generator is installed. I/O port, etc., can also be installed as required. Table 4.1 lists a system configuration example based on the items mentioned above.

Table 4.1 System configuration example

<table>
<thead>
<tr>
<th>Memory</th>
<th>CPU</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROM</td>
<td>µPD8085AH (3.072 MHz)</td>
<td>System program</td>
</tr>
<tr>
<td></td>
<td></td>
<td>µPD2764x2 (16K bytes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>µPD7201A control program</td>
</tr>
<tr>
<td></td>
<td></td>
<td>System program</td>
</tr>
<tr>
<td></td>
<td></td>
<td>µPD7201A control program</td>
</tr>
<tr>
<td>RAM</td>
<td></td>
<td>Data buffer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Working area</td>
</tr>
<tr>
<td></td>
<td>µPD4016x4 (8K bytes)</td>
<td>Data buffer</td>
</tr>
<tr>
<td>DMA controller</td>
<td></td>
<td>Send and receive data transfer control</td>
</tr>
<tr>
<td></td>
<td>µPD8237A-5 (3.072 MHz)</td>
<td>Send and receive data transfer control</td>
</tr>
<tr>
<td>Counter/timer</td>
<td>µPD8253-2</td>
<td>Baud rate generator</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Timer</td>
</tr>
<tr>
<td>Serial interface</td>
<td>µPD7201A (4.9152 MHz)</td>
<td>Serial data processing</td>
</tr>
</tbody>
</table>
4.2 Circuit Configuration

Figure 4.1 shows an actual circuit design example based on the configuration listed in Table 4.1. The circuit is configured as follows:

(1) Connection of μPD7201A and μPD8237A-5

This connection assumes that Both CH. DMA-2 mode (CR2A:D1, D0 = 1, 1) is used. Since the μPD8237A-5 DACK signal is used to select a μPD7201A DMA request channel (CS, B/A, C/D control), the HAKI pin is not used.

The DMA priority in this circuit is DRQRxA(DRQO) > DRQTxA (DRQ1) > DRQRxB(DRQ2) > DRQTxB(DRQ3) in accordance with μPD8237A-5 DRQ input. The priority can be set by using external circuit as desired. Select it according to the system configuration. The μPD8237A-5 DACK signal is used as active low. (For the μPD8236A-5 use method, refer to the appropriate material, etc.)

(2) Interrupt operation

To enable use of interrupts in the vector mode, the circuit configuration enables the INTAK signal to be used to read interrupt vector. In this mode, the μPD7201A INT signal is connected to the μPD8085AH INTR pin and the INTA signal is input to the μPD7201A INTAK signal.

To use interrupts in the nonvector mode, the μPD7201A INT signal is connected to the μPD8085AH RST7.5 pin and the μPD7201A INTAK signal is pulled up to a high level.

To use interrupts in any mode other than the Both CH. DMA-2 mode (CR2A:D1, D0 = 1, 1), the PRI line must be held LOW (however, Both CH. DMA-1 mode cannot be used).
(3) Serial data send and receive

TXDA, RxDA, TxDB, and RxDB can be used. However, since interfacing with external devices (for example through an as RS-232-C port) is not considered in the circuit, an appropriate interface circuit must be added for connection to external devices.

(4) Send and receive clocks

Send and receive clocks are supplied to TxCA, RxCA, TxCB, and RxCB from µPD8253-2 OUT0 or from external sources. For operation in Asynchronous X1 mode, Synchronous mode, or HDLC mode, receive data must be synchronized with the receive clock.

(5) µPD7201A general-purpose input and output lines

The pins CTSA, DCDA, CTSB, and DCDB, which are always LOW can be used as general-purpose input lines. The functions of the general-purpose input and output pins varies with operation mode. Select the function according to the operation mode.

(6) Address map

Figure 4.2 shows the memory address map. Figure 4.3 shows the I/O address map.

Figure 4.2 Memory map
## Figure 4.3 I/O map

<table>
<thead>
<tr>
<th>I/O address</th>
<th>Contents</th>
<th>I/O device</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0H</td>
<td>Channel A data</td>
<td>μPD7201A area</td>
</tr>
<tr>
<td>C1</td>
<td>Channel A control</td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td>Channel B data</td>
<td></td>
</tr>
<tr>
<td>C3</td>
<td>Channel B control</td>
<td></td>
</tr>
<tr>
<td>C4</td>
<td>Counter #0</td>
<td></td>
</tr>
<tr>
<td>C5</td>
<td>Counter #1</td>
<td>μPD8253-2 area</td>
</tr>
<tr>
<td>C6</td>
<td>Counter #2</td>
<td></td>
</tr>
<tr>
<td>C7</td>
<td>Control</td>
<td></td>
</tr>
<tr>
<td>C8 / CF</td>
<td>Inhibited</td>
<td></td>
</tr>
<tr>
<td>D0</td>
<td>Channel 0 base/current address</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>Channel 0 base/current word count</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>Channel 1 base/current address</td>
<td></td>
</tr>
<tr>
<td>D3</td>
<td>Channel 1 base/current word count</td>
<td></td>
</tr>
<tr>
<td>D4</td>
<td>Channel 2 base/current address</td>
<td></td>
</tr>
<tr>
<td>D5</td>
<td>Channel 2 base/current word count</td>
<td></td>
</tr>
<tr>
<td>D6</td>
<td>Channel 3 base/current address</td>
<td></td>
</tr>
<tr>
<td>D7</td>
<td>Channel 3 base/current word count</td>
<td>μPD8237A-5 area</td>
</tr>
<tr>
<td>D8</td>
<td>Status/command register</td>
<td></td>
</tr>
<tr>
<td>D9</td>
<td>Write request register</td>
<td></td>
</tr>
<tr>
<td>DA</td>
<td>Write single mask register bit</td>
<td></td>
</tr>
<tr>
<td>DB</td>
<td>Write mode register</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>Clear byte pointer flip-flop</td>
<td></td>
</tr>
<tr>
<td>DD</td>
<td>Read temporary register/master clear</td>
<td></td>
</tr>
<tr>
<td>DE</td>
<td>Clear mask register</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>Write all mask register bits</td>
<td></td>
</tr>
</tbody>
</table>
Figure 4.1 (a) μPD7201A application circuit example (reference circuit)
Figure 4.1 (b) μPD7201A application circuit example (reference circuit)
Chapter 5 explains software used to actually operate the hardware designed in Chapter 4. First, memory assignments are made and program allocation is determined, then a specific program example in Synchronous mode and HDLC mode is discussed. Use this example to understand the basic μPD7201A control method.

5.1 Memory Assignment

Memory assignment is discussed before an actual program is prepared. Since the program explained in this chapter is intended to operate in the hardware environment described in Chapter 4, the hardware conditions are considered to when assigning memory, as shown in Figure 5.1.

The regions shaded in Figure 5.1 are mainly used for the system area. The system control program (monitor program) uses the area.

Other regions are used as the μPD7201A control program area.

The initial value of μPD7201A interrupt vector (value set in CR2B) is set to 40H. Thus, the interrupt vector address area is 40H to 50H.

The area stores the jump destination addresses corresponding to various interrupt sources when the μPD7201A is used in vector mode. (See Table 5.1.) Since the μPD8085AH RST7.5 pin is used when the μPD7201A is uses nonvector mode, and the jump destination address becomes 3CH when an interrupt occurs.
Figure 5.1 Memory assignment
Table 5.1 Interrupt vector addresses and interrupt sources

<table>
<thead>
<tr>
<th>Interrupt vector address</th>
<th>μPD7201A interrupt source</th>
</tr>
</thead>
<tbody>
<tr>
<td>40H</td>
<td>Channel B Tx Buffer Empty</td>
</tr>
<tr>
<td>44H</td>
<td>&quot; External Status Change</td>
</tr>
<tr>
<td>48H</td>
<td>&quot; Receive Character Available</td>
</tr>
<tr>
<td>4CH</td>
<td>&quot; Special Rx Condition</td>
</tr>
<tr>
<td>50H</td>
<td>Channel A Tx Buffer Empty</td>
</tr>
<tr>
<td>54H</td>
<td>&quot; External Status Change</td>
</tr>
<tr>
<td>58H</td>
<td>&quot; Receive Character Available</td>
</tr>
<tr>
<td>5CH</td>
<td>&quot; Special Rx Condition</td>
</tr>
</tbody>
</table>
5.2 Operation Example in Synchronous Mode

This program example is to perform send and receive operation on channel A in interrupt mode. In the example program, the receive operation is performed after completion of send operation for reasons of explanation. A program which performs send and receive operation at the same time can also be prepared.

Table 5.2 lists the µPD7201A operation mode in the present example.

The example program assumes that the send operation is to send one data block and receive operation is to receive one data block ending with ETX.
Table 5.2 \( \mu \text{PD7201A} \) operation mode setting (Synchronous operation)

<table>
<thead>
<tr>
<th>Operation contents</th>
<th>Channel A</th>
<th>Channel B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send and receive operation</td>
<td>Send (TxDA)</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Receive (RxDA)</td>
<td></td>
</tr>
<tr>
<td>Send and receive data transfer method (CR2A setting)</td>
<td>Interrupt</td>
<td>Interrupt</td>
</tr>
<tr>
<td>Interrupt sources used</td>
<td>Tx interrupt</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Rx interrupt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Special Rx Condition</td>
<td></td>
</tr>
<tr>
<td></td>
<td>interrupt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>E/S interrupt</td>
<td></td>
</tr>
<tr>
<td>Interrupt mode</td>
<td>Vector mode</td>
<td>Vector mode</td>
</tr>
<tr>
<td>Vector mode (CR2A: D5)</td>
<td>85-1 vector mode</td>
<td>40H</td>
</tr>
<tr>
<td>Interrupt mode (CR2A: D4, D3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status Affects Vector (CR1B: D2)</td>
<td>Modified Vector</td>
<td></td>
</tr>
<tr>
<td>Initial value of interrupt vector (CR2B)</td>
<td></td>
<td>40H</td>
</tr>
<tr>
<td>Synchronous character</td>
<td>&quot;16H&quot; x 2 bytes</td>
<td>-</td>
</tr>
<tr>
<td>Synchronization system</td>
<td>Internal synchronization</td>
<td>-</td>
</tr>
</tbody>
</table>

(1) Send operation

This example assumes that the characters shown in Figure 5.2 are sent. When the characters shown in Figure 5.2 are actually sent in Synchronous mode, normally control characters, etc., are added to normal characters. (For details, refer to the Standard Book, etc.) Figure 5.3 shows the configuration of actually sent characters.
PAD1 and PAD2 in Figure 5.3 are pad characters. Normally, PAD1 sent before send is set to 55H and PAD2 sent following CRC2 is set to FFH. Synchronous characters (SYN) of at least two bytes must be sent as character synchronization code following PAD1. In the example, 55H is set in CR6A as PAD1 and 16H is set in CR7A as SYN at initialization time, then send is enabled. This enables PAD1 and SYN to be automatically sent before characters are sent. Immediately after send is enabled, 16H is rewritten into CR6A and again SYN is set.

This sequence enables PAD1, SYN, SYN, and SYN to be sent. The number of SYN's to be sent can be controlled by reserving the time until the first character is written. All characters other than the first character (STX in the example) are written by using the interrupt handling routine. The CRC characters are automatically sent by the µPD7201A. If PAD2 is written as data when TX underrun occurs, it can be sent following the CRC characters.

CRC calculation is enabled or disabled immediately before characters are written. In the example, STX and PAD2 are excluded from CRC calculation.

CRC calculation is not applied to characters automatically inserted from CR6 and CR7, thus CRC calculation enable/disable control is not required. This applied to the four continuous characters PAD1, SYN, SYN, SYN in the example.

(2) Receive operation

The example assumes that the characters shown in Figure 5.4 are received.
In the receive operation, two continuous bytes of SYNC must first be detected to set character synchronization. Thus, the µPD7201A is first placed in Hunt mode to detect SYN. When character synchronization is set, the µPD7201A generates, receive interrupt for the subsequent characters received. The example program is designed to receive SYN to ETX characters as normal characters. In Synchronous mode, characters excluded from CRC calculation must be checked. This control can be implemented by reading a receive character and checking the character contents, then enabling or disabling CRC calculation.

In this case, CRC calculation enable or disable control must be completed before the next character is transferred to the receive buffer. This can be processed by controlling CRC calculation enable or disable within the time of five bits (during eight bits per character) after a µPD7201A receive interrupt occurs.

In the example, SYN, PAD2, and dummy data are excluded from CRC calculation.

The µPD7201A uses a 16-bit time interval after the last character (second byte of CRC in the example) is transferred to the receive buffer until the CRC calculation result is output. Thus, after the last character is received, on extra two bytes of dummy data are received before the CRC Error bit (SR1A: D6 bit) is checked.

Figures 5.5 to 5.9 show the process flow to perform the operation mentioned above. Figure 5.10 shows actual program listing.
Figure 5.2 Send character configuration
(send starts at the leftmost character)

These characters are contained in CRC calculation.

Figure 5.3 Actually sent characters
(send starts at the leftmost character)

Figure 5.4 Receive character string format
(receive starts at the leftmost character)
... Start at 100H address

... Control flags used in the program are initialized

... Basic operation mode is set

... Send operation is started

... Tx CRC remains disabled

... CR6A is again set to SYN (16H)

... The number of SYNs to be sent can be adjusted by changing the value.

... STX is written

... A check is made to see if all send characters have been written.

... Send operation is terminated

Figure 5.5 Synchronous operation main program 1
Figure 5.5 Synchronous operation main program 2
Figure 5.6 Synchronous operation Tx interrupt processing
Figure 5.7 Synchronous operation Rx interrupt processing
Figure 5.8 Synchronous operation E/S interrupt processing
Figure 5.9 Synchronous operation Special Rx Condition interrupt processing
<table>
<thead>
<tr>
<th>SY:IBOL ADRS</th>
<th>SY:IBOL ADRS</th>
<th>SY:IBOL ADRS</th>
<th>SY:IBOL ADRS</th>
<th>SY:IBOL ADRS</th>
<th>SY:IBOL ADRS</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSCCT 0100</td>
<td>CHACT 00C1</td>
<td>CHADT 00C0</td>
<td>CHBCT 00C3</td>
<td>CHBDT 00C2</td>
<td></td>
</tr>
<tr>
<td>CHNRS 0018</td>
<td>CKCNT 10D7</td>
<td>CKCRC 110E</td>
<td>CKERR 1132</td>
<td>CKES1 105C</td>
<td></td>
</tr>
<tr>
<td>CKESS 1075</td>
<td>CKRDE 109B</td>
<td>CKWRE 1003</td>
<td>CLRFG 0103</td>
<td>CRIA 0013</td>
<td></td>
</tr>
<tr>
<td>CRIB 0004</td>
<td>CR2A 0020</td>
<td>CR2B 0040</td>
<td>CR3A 00C2</td>
<td>CR4A 0010</td>
<td></td>
</tr>
<tr>
<td>CR5A 0044</td>
<td>CR6A 0016</td>
<td>CR7A 0016</td>
<td>CRCEF 5B09</td>
<td>CRCNT 5B08</td>
<td></td>
</tr>
<tr>
<td>CRCSR 10B7</td>
<td>CRCST 1022</td>
<td>CUNDF 1034</td>
<td>DMCNT 5B0C</td>
<td>EDI 0038</td>
<td></td>
</tr>
<tr>
<td>ERDFG 5B07</td>
<td>ERRES 0030</td>
<td>ERVDF 5B08</td>
<td>ESINA 104D</td>
<td>ESKES 0010</td>
<td></td>
</tr>
<tr>
<td>ESR6A 1088</td>
<td>ETX 0003</td>
<td>EWDFG 5B04</td>
<td>EXMD 5B06</td>
<td>EXTHP 1080</td>
<td></td>
</tr>
<tr>
<td>OVREF 5B0A</td>
<td>OVRER 1139</td>
<td>PADI 0055</td>
<td>PAD2 00FF</td>
<td>PEND 0028</td>
<td></td>
</tr>
<tr>
<td>PITCH 00C4</td>
<td>PITCH 00C5</td>
<td>PITCH 00C6</td>
<td>PITCH 00C7</td>
<td>HCRDS 10CC</td>
<td></td>
</tr>
<tr>
<td>RCREN 10C1</td>
<td>RDRCR 10E1</td>
<td>RDAT 10A2</td>
<td>RDMY 10F6</td>
<td>RXBFS 01A3</td>
<td></td>
</tr>
<tr>
<td>RXBIF 4400</td>
<td>RXCRC 0040</td>
<td>RXEOI 1123</td>
<td>RXNA 1094</td>
<td>RXPTR 5B02</td>
<td></td>
</tr>
<tr>
<td>RXTR1 01B8</td>
<td>SPRX 112D</td>
<td>SR0AS 5B0D</td>
<td>SRXEF 111E</td>
<td>STACK 6000</td>
<td></td>
</tr>
<tr>
<td>STCHA 0135</td>
<td>STPIT 010C</td>
<td>STX 0002</td>
<td>STXDT 100A</td>
<td>SWREF 1014</td>
<td></td>
</tr>
<tr>
<td>SYHNT 107A</td>
<td>SYN 0016</td>
<td>SYRCV 01AF</td>
<td>SYSTM 2000</td>
<td>SYXMT 0170</td>
<td></td>
</tr>
<tr>
<td>TDL1Y 0180</td>
<td>TXBFS 016A</td>
<td>TXBIF 4000</td>
<td>TXCRC 0080</td>
<td>TXEOI 1044</td>
<td></td>
</tr>
<tr>
<td>TXIMA 1000</td>
<td>TXPTR 5B00</td>
<td>TXTRM 0193</td>
<td>UNDEM 00C0</td>
<td>UNDES 1061</td>
<td></td>
</tr>
<tr>
<td>UNDFG 5B05</td>
<td>WRDAT 102A</td>
<td>XMTDT 0188</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.10  Synchronous operation program example 1
I STNO ADRS OBJECT II  SOURCE STATEMENTS

0001  :****************************************************=
0002  :   * UPD7201A BSC OPERATION SAMPLE PROGRAM *   
0003  :   TX : TXDA -->                           *
0004  :   RX : RXDA <--                           *
0005  :****************************************************=
0006  :
0007  :***************                            
0008  :* I/O ADDRESS *                          
0009  :***************                            
0110  :
0111  00C0  CHADT  EQU  0C0H    :7201A CH-A DATA
0112  00C1  CHACT  EQU  0C1H    :7201A CH-A CNTRL
0113  00C2  CHBDT  EQU  0C2H    :7201A CH-B DATA
0114  00C3  CHBCT  EQU  0C3H    :7201A CH-B CNTRL
0115  :
0116  00C4  PITC0  EQU  0C4H    :8253 CNT #0
0117  00C5  PITC1  EQU  0C5H    :8253 CNT #1
0118  00C6  PITC2  EQU  0C6H    :8253 CNT #2
0119  00C7  PITCT  EQU  0C7H    :8253 CONTROL
0120  :
0121  2000  SYSTH  EQU  2000H   :SYSTEM PROGRAM
0122  :
0123  :***************                            
0124  :* MPU COMMAND *                           
0125  :***************                            
0126  :
0127  0010  ESRES  EQU  10H     :RESET E/S INT
0128  0018  CHNRS  EQU  18H     :CHANNEL RESET
0129  0028  PEND  EQU  28H      :RES TX PENDING
0130  0030  ERRRES EQU  30H     :ERROR RESET
0131  0038  E01  EQU  38H      :END OF INTERRUPT
0132  :
0133  0040  RXCRC  EQU  40H     :INIT RX CRC CAL
0134  0080  TXCRC  EQU  80H     :INIT TX CRC CAL
0135  00C0  UNDEM  EQU  0C0H    :RESET TX UND/EOM
0136  :
0137  :***************                            
0138  :* :MPSC CONTROL REGISTER PARAMETER *        
0139  :***************                            
0140  :
0141  0013  CR1A  EQU  13H     :RX.TX.E/S INT EN
0142  0020  CR2A  EQU  20H     :INT.VECTORED MODE
0143  00C2  CR3A  EQU  0C2H    :RX 8 BITS
0144  0010  CR4A  EQU  10H     :16 BITS SYN.NO PA
0145  0044  CR5A  EQU  44H     :TX 8 BITS.CRC-16
0146  0016  CR6A  EQU  16H     :SYN #1
0147  0016  CR7A  EQU  16H     :SYN #2
0148  :
0149  0004  CR1B  EQU  04H     :MODIFIED VECTOR
0150  0040  CR2B  EQU  04H     :VECTOR(010***00)
0151  :
0152  :

Figure 5.10  Synchronous operation program example 2
Figure 5.10 Synchronous operation program example 3
Figure 5.10  Synchronous operation program example 4
Figure 5.10 Synchronous operation program example 5
E STNO ADRS OBJECT " SOURCE STATEMENTS

0212 :
0213 01B8 3A085B RXTRM: LDA ERVDF :CHECK RCV END FLG
0214 01BB 0F RRC
0215 01BC D28001 JNC RXTRM
0216 01BF 3E03 MVI A.3H :PTR 3A
0217 01C1 D3C1 OUT CHACT
0218 01C3 3EC0 MVI A.OCOH :RX DISABLE
0219 01C5 D3C1 OUT CHACT
0220 01C7 00 NOP
0221 01C8 00 NOP
0222 01C9 F3 DI
0223 :
0224 01CA C30020 JMP SYSTM
0225 :
0226 :************************************************
0227 ; * INTERRUPT PROCESSING ROUTINE (VECTORED MODE) *
0228 :************************************************
0229 :
0230 01CD ORG 1000H
0231 :
0232 ; CH-A TX INT PROCESS
0233 :
0234 :
0235 :
0236 TXINA:
0237 1000 F5 PUSH PSW
0238 1001 C5 PUSH B
0239 1002 E5 PUSH H
0240 1003 3A045B CKWRE: LDA EWDFG :CHECK WR END FLG
0241 1006 0F RRC
0242 1007 DA3410 JC CUNDF
0243 100A 2A005B STXDT: LHLD TXPTR :GET TX DATA ADRS
0244 100D 7E MOV A.M :GET TX DATA
0245 100E 4F MOV C.A
0246 100F FE03 CCI ETX :LAST TX DATA ?
0247 1011 C22210 JNZ CRCST
0248 1014 3EO1 SWREF: MVI A.1 :SET WR END FLG
0249 1016 32045B STA EWDFG
0250 1019 3EC0 MVI A.UNDEM :RES TX UND/EDM
0251 101B D3C1 OUT CHACT
0252 101D D8C1 IN CHACT :READ SROA
0253 101F 320D5B STA SROAS :NEW SROA STATUS
0254 1022 3E05 CRCST: MVI A.5H :PTR 5A
0255 1024 D3C1 OUT CHACT
0256 1026 3EEF MVI A.0EFH :TX CRC ENABLE
0257 1028 D3C1 OUT CHACT
0258 102A 79 WRDAT: MOV A.C :SET TX DATA
0259 102B D3C0 OUT CHADT
0260 102D 23 INX H :UPDATE ADRS PTR
0261 102E 22005B SHLD TXPTR
0262 1031 C34410 JMP TXEOI
0263 1034 3E28 CUNDF: MVI A.PEND :RESET TX PENDING
0264 1036 D3C1 OUT CHACT

Figure 5.10 Synchronous operation program example 6
E STNO ADRS OBJECT M SOURCE STATEMENTS

0265 1038 3A055B LDA UNDFG :GET UNDRN FLG
0266 103B 0F RRC
0267 103C D24410 JNC TXEO1
0268 103F 3E01 MVI A.I :SET TX END FLG
0269 1041 32065B STA EX4DF
0270 1044 3E38 TXEO1: MVI A.EOI :EOI
0271 1046 D3C1 OUT CHACT
0272 1048 E1 POP H
0273 1049 C1 POP B
0274 104A F1 POP PSW
0275 104B FB EI
0276 104C C9 RET

0277 :----------------------
0278 ; CH-A E/S INT PROCESS
0279 :----------------------
0280 :
0281 : ESINA:
0282 104D F5 PUSH PSW
0283 104E C5 PUSH B
0284 104F 3A0D5B LDA SROAS :GET OLD E/S BITS
0285 1052 47 MOV B.A
0286 1053 DBC1 IN CHACT :READ NEW E/S BITS
0287 1055 4F MOV C.A
0288 1056 320D5B STA SROAS :NEW E/S BITS
0289 1059 A8 XRA B :CHECK CHANGED E/S
0290 105A E650 ANI 50H
0291 105C 07 CKES1: RLC
0292 105D 07 RLC
0293 105E D27510 JNC CKES2
0294 1061 3E05 UNDES: MVI A.5 :PTR 5A
0295 1063 D3C1 OUT CHACT
0296 1065 3EEE MVI A.0EEH :TX CRC DISABLE
0297 1067 D3C1 OUT CHACT
0298 1069 3EFF MVI A.PAD2 :WRITE TPAD
0299 106B D3C0 OUT CHADT
0300 106D 3E01 MVI A.1 :SET UNDRN FLG
0301 106F 32055B STA UNDFG
0302 1072 C38810 JMP ESRTA
0303 1075 07 CKES2: RLC
0304 1076 07 RLC
0305 1077 D28810 JNC ESRTA
0306 107D D28810 JNZ ESRTA
0307 107A 79 SYHNT: MOV A.C :SET NEW E/S BITS
0308 107B E610 ANI 10H :SYNC/HUNT BIT=1? 1
0309 107D C28810 JNZ ESRTA
0310 1080 3E03 EXTHP: MVI A.3H :PTR 3A
0311 1082 D3C1 OUT CHACT
0312 1084 3EC9 MVI A.0C9H :RES SYN LOAD INH
0313 1086 D3C1 OUT CHACT
0314 1088 3E10 ESRTA: MVI A.ESRES :RESET E/S INT
0315 108A D3C1 OUT CHACT
0316 108C 3E38 MVI A.EOI :EOI
0317 108E D3C1 OUT CHACT

Figure 5.10 Synchronous operation program example 7

1-138
Figure 5.10  Synchronous operation program example 8
Figure 5.10 Synchronous operation program example 9
Figure 5.10 Synchronous operation program example 10
E STNO ADRS OBJECT M SOURCE STATEMENTS

0477 ;
0478 5B0D SROAS: DS 1 :SROA STATUS
0479 ;
0480 ;**************
0481 ;* STACK AREA *
0482 ;**************
0483 ;
0484 5B0E ORG 6000H
0485 ;
0486 STACK:
0487 0000 END

Figure 5.10 Synchronous operation program example 11
5.3 Operation Example in HDLC Mode

Send and receive operation via DMA on channel A is described as an operation example in HDLC mode. In the example, the send and receive control programs are independent of each other, but a program which performs send and receive operation at the same time can also be prepared because the μPD7201A enables full-duplex operation.

Each of the example send and receive control programs transfers data of 256 bytes per frame.

Table 5.3 Lists the μPD7201A operation mode in the program example

<table>
<thead>
<tr>
<th>Operation contents</th>
<th>Channel A</th>
<th>Channel B</th>
</tr>
</thead>
</table>
| Send and receive operation | Send (TxDA)  
Receive (RxDA) | -          |
| Send and receive data transfer method (CR2A setting) | DMA       | DMA       |
| Interrupt sources used | Special Rx Condition interrupt  
E/S interrupt | -          |

<table>
<thead>
<tr>
<th>Interrupt setting</th>
<th>Channel A</th>
<th>Channel B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector mode (CR2A: D5)</td>
<td>Nonvector mode</td>
<td></td>
</tr>
<tr>
<td>Interrupt mode (CR2A: D4, D3)</td>
<td>85-1 mode</td>
<td></td>
</tr>
<tr>
<td>Status Affects Vector (CR1B: D2)</td>
<td>Modified Vector</td>
<td></td>
</tr>
<tr>
<td>Initial value of interrupt vector (CR2B)</td>
<td></td>
<td>40H</td>
</tr>
</tbody>
</table>

Table 5.3 μPD7201A operation mode setting (HDLC operation)
5.3.1 Send operation

DMA transfer of data of 256 bytes per frame by using the TxLR set mode (00-FFH) is described as a μPD7201A send operation example. Since nonvector mode is used in interrupt processing, the jump destination address when an interrupt occurs is fixed to address 3CH and a single interrupt handling routine is used.

The interrupt handling routine first reads the SR2B vector register and determines the interrupt source, then executes appropriate processing.

The basic program configuration is explained below:

(1) Main program

The main program sets the transmit buffer, initializes I/O devices (μPD8253-2, μPD8237A-5) and the μPD7201A, manages the send state, etc. After send starts, the program waits for the All Sent flag (which is set by the interrupt handling program) to be set and checks that the flag is set before disabling send.

(2) Interrupt handling program

Since the RST7.5 pin is used as an interrupt input pin, the μPD8085AH jumps to address 3CH when a μPD7201A interrupt request occurs. Using this address, control is passed to the actual interrupt handling routine (starting at address 1000H).
Only E/S interrupts are interrupt sources in the program example (interrupts caused by other sources and disabled). The E/S interrupt sources are Tx Underrun/EOM and All Sent only. Thus, the interrupt handling program checks only these two types of interrupt sources. The program checks to see if any of the states occurs. When the state occurs, it sets the corresponding control flag. The flag is used to manage the µPD7201A send operation state in the main program.

Figure 5.10 shows the send operation processing flow. Figure 5.11 shows actual program examples.
... Start at address 100H

... Data of 00-FFH is set

... Control flags used in the program are initialized.

... Counter 0 is set for use at a rate of 9600 bps.

... Channel 1 is set (number of transfer data words, send data, and transmit buffer address are set).

... Basic operation mode is set.

... DMA operation is enabled

... Wait for all Sent interrupt to occur.

... When TxLR Set mode is used, this command must be issued to set Tx Disable.
Figure 5.10 (b)  HDLC send operation Interrupt handling program
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALSFG</td>
<td>5B01</td>
<td>ALSNT</td>
<td>1027</td>
<td>BADR1</td>
<td>5B04</td>
<td>BCWC1</td>
<td>5B06</td>
<td>CH1ST</td>
<td>012D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHACT</td>
<td>00C1</td>
<td>CHADT</td>
<td>00C0</td>
<td>CHBCT</td>
<td>00C3</td>
<td>CHBDT</td>
<td>00C2</td>
<td>CHNRS</td>
<td>0018</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CKEND</td>
<td>01AA</td>
<td>CKES</td>
<td>100F</td>
<td>CR1A</td>
<td>004B</td>
<td>CR1B</td>
<td>0004</td>
<td>CR2A</td>
<td>0043</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR2B</td>
<td>0040</td>
<td>CR3A</td>
<td>00C0</td>
<td>CR4A</td>
<td>0020</td>
<td>CR5A</td>
<td>0061</td>
<td>CR6A</td>
<td>0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR7A</td>
<td>007E</td>
<td>DM0BC</td>
<td>00D0</td>
<td>DM0CW</td>
<td>00D1</td>
<td>DM1BC</td>
<td>00D2</td>
<td>DM1CW</td>
<td>00D3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DM2BC</td>
<td>00D4</td>
<td>DM2CW</td>
<td>00D5</td>
<td>DM3BC</td>
<td>00D6</td>
<td>DM3CW</td>
<td>00D7</td>
<td>DM3CP</td>
<td>00DC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMCMR</td>
<td>00DE</td>
<td>DMCTR</td>
<td>00D8</td>
<td>DMMCL</td>
<td>00DD</td>
<td>DSMR</td>
<td>00DA</td>
<td>DMWAM</td>
<td>00DF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMMD</td>
<td>00DB</td>
<td>DMWRR</td>
<td>00D9</td>
<td>ENIRX</td>
<td>0020</td>
<td>EOI</td>
<td>0038</td>
<td>EOIA</td>
<td>103A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERRES</td>
<td>0030</td>
<td>ERRTA</td>
<td>1036</td>
<td>ESRES</td>
<td>0010</td>
<td>HDLCT</td>
<td>0100</td>
<td>INTEN</td>
<td>0112</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NVINT</td>
<td>1000</td>
<td>PEND</td>
<td>0028</td>
<td>PITC0</td>
<td>00C4</td>
<td>PITCT</td>
<td>00C7</td>
<td>RXCRC</td>
<td>0040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SABRT</td>
<td>0008</td>
<td>STACK</td>
<td>0600</td>
<td>STCHA</td>
<td>0165</td>
<td>STDMA</td>
<td>0122</td>
<td>STPIIT</td>
<td>0115</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STTXD</td>
<td>0104</td>
<td>SYSTM</td>
<td>2000</td>
<td>TXBUF</td>
<td>4000</td>
<td>TXCRC</td>
<td>0080</td>
<td>TXLRA</td>
<td>5B02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNDEM</td>
<td>00C0</td>
<td>UNDFG</td>
<td>5B00</td>
<td>UNDRN</td>
<td>1019</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.11   HDLC send control program example 1
E STNO ADRS OBJECT M SOURCE STATEMENTS

0001 *
0002 :**************************************************
0003 :* UPD7201A HDLC TX OPERATION SAMPLE PROGRAM *
0004 :* TX : TXDA --> *
0005 :**************************************************
0006 :
0007 :* I/O ADDRESS *
0008 :**************************************************
0009 :
0010 00C0 CHADT EQU 0C0H :7201A CH-A DATA
0011 00C1 CHACT EQU 0C1H :7201A CH-A CONTRO
0012 00C2 CHBDT EQU 0C2H :7201A CH-B DATA
0013 00C3 CHBCT EQU 0C3H :7201A CH-B CONTRO
0014 :
0015 00C4 P1TC0 EQU 0C4H :8253 CNT #0
0016 00C7 P1TC7 EQU 0C7H :8253 CONTROL
0017 :
0018 00D0 DM0BC EQU 0D0H :8237 BASE/CURAD
0019 00D1 DM0CW EQU 0D1H :8237 BASE/CWC
0020 00D2 DM1BC EQU 0D2H :8237 BASE/CURAD
0021 00D3 DM1CW EQU 0D3H :8237 BASE/CWC
0022 00D4 DM2BC EQU 0D4H :8237 BASE/CURAD
0023 00D5 DM2CW EQU 0D5H :8237 BASE/CWC
0024 00D6 DM3BC EQU 0D6H :8237 BASE/CURAD
0025 00D7 DM3CW EQU 0D7H :8237 BASE/CWC
0026 00D8 DMCTR EQU 0D8H :8237 STS/CMD R
0027 00D9 DMWRR EQU 0D9H :8237 WR REQ R
0028 00DA DM5MR EQU 0DAH :8237 WR SIG MRR
0029 00DB DMWMD EQU 0DBH :8237 WR MODE R
0030 00DC DMCBP EQU 0DCH :8237 CL BYTP FF
0031 00DD DMMLCL EQU 0DDH :8237 RDMCL.MSCL
0032 00DE DMCMR EQU 0DEH :8237 CL MSK R
0033 00DF DMWMAM EQU 0DFH :8237 WR ALMS RB
0034 :
0035 2000 SYSTM EQU 2000H :SYSTEM PROGRAM
0036 :
0037 :**************************************************
0038 :* MPSC COMMAND *
0039 :**************************************************
0040 :
0041 0008 SABRT EQU 8H :SEND ABORT
0042 0010 ESR2ES EQU 10H :RESET E/S INT
0043 0018 CHNRS EQU 18H :CHANNEL RESET
0044 0020 EINRX EQU 20H :EN INT NEXT CHAR
0045 0028 PENRT EQU 28H :RES TX PENDING
0046 0030 ERRS EQU 30H :ERROR RESET
0047 0038 EO1 EQU 38H :END OF INTERRUPT
0048 :
0049 0040 RXCRC EQU 40H :INIT RX CRC CAL
0050 0080 TXCRC EQU 80H :INIT TX CRC CAL
0051 00C0 UNDEM EQU 0COH :RESET TX UND/EOM
0052 :

Figure 5.11 HDLC send control program example 2
Figure 5.11 HDLC send control program example 3
E STNO ADRS OBJECT M SOURCE STATEMENTS

0106 ; 8253-2 INITIALIZE
0107 :
0108 0115 210001 STPIT: LXI H.100H :CNT #0.9600 BPS
0109 0118 3E36 MVI A.36H :#0 MODE 2
0110 011A D3C7 OUT PITCT
0111 011C 7D MOV A.L
0112 011D D3C4 OUT PITCO
0113 011F 7C MOV A.H
0114 0120 D3C4 OUT PITCO
0115 :
0116 ; 8237A-5 INITIALIZE
0117 :
0118 0122 AF STDMA: XRA A
0119 0123 D3DD OUT DMMCL :MASTER CLEAR
0120 0125 3E04 MVI A.4H :DMAC DISABLE
0121 0127 D3D8 OUT DMCTR
0122 0129 3EFD MVI A.0FDH :CH-1 ENABLE
0123 012B D3DF OUT DMWAM
0124 :
0125 012D AF CHIST: XRA A :CLEAR BYTE PTR F/
0126 012E D3DC OUT DMCBP
0127 0130 2A045B LHLD BADR1 :CH-1 BADRS
0128 0133 7D MOV A.L :CH-1 BADRS-L
0129 0134 D3D2 OUT DM1BC
0130 0136 7C MOV A.H :CH-1 BADRS-H
0131 0137 D3D2 OUT DM1BC
0132 0139 AF XRA A :CLEAR BYTE PTR F/
0133 013A D3DC OUT DMCBP
0134 013C 2A065B LHLD BCWC1 :CH-1 TRNSFR CNT
0135 013F 7D MOV A.L
0136 0140 D3D3 OUT DM1CW
0137 0142 7C MOV A.H
0138 0143 D3D3 OUT DM1CW
0139 0145 3E49 MVI A.49H :DRQ1 MODE
0140 0147 D3DB OUT DMWMD
0141 :
0142 :
0143 ;*******************************
0144 ; * MPSC INITIALIZE SEQUENCE *
0145 ;*******************************
0146 :
0147 ; COMMON REGISTER INITIALIZE
0148 :
0149 0149 3E18 MVI A.CHNRS :CHANNEL RESET
0150 014B D3C1 OUT CHACT
0151 014D D3C3 OUT CHBCT
0152 014F 3E02 MVI A.2H :PTR 2
0153 0151 D3C1 OUT CHACT
0154 0153 D3C3 OUT CHBCT
0155 0155 3E43 MVI A.CR2A :BUS IF MODE
0156 0157 D3C1 OUT CHACT
0157 0159 3E40 MVI A.CR2B :INERRUPT VECTOR
0158 015B D3C3 OUT CHBCT

Figure 5.11 HDLC send control program example 4

1-151
Figure 5.11 HDLC send control program example 5
Figure 5.11 HDLC send control program example 6
E STNO ADRS OBJECT M SOURCE STATEMENTS

0265  ;******************************
0266  ;* TX DATA BUFFER AREA *
0267  ;******************************
0268  :
0269  1041 ORG 4000H
0270  :
0271  4000 TXBUF: DS 1024
0272  :
0273  ;******************************
0274  ;* STATUS FLAG AREA *
0275  ;******************************
0276  :
0277  4400 ORG 5B00H
0278  :
0279  5B00 UNDFG: DS 1 ;TX UND FLAG
0280  5B01 ALSFG: DS 1 ;ALL SENT FLAG
0281  :
0282  ;******************************
0283  ;* I/O PARAMETER AREA *
0284  ;******************************
0285  :
0286  5B02 0001 TXLRA: DW 100H ;TXLR
0287  :
0288  5B04 0040 BADRI: DW 4000H ;CH-1 DMA ADRS
0289  5B06 FF00 BCWCI: DW 0FFH ;CH-1 WORD COUNT
0290  :
0291  ;******************************
0292  ;* STACK AREA *
0293  ;******************************
0294  :
0295  5B08 ORG 6000H
0296  :
0297  STACK:
0298  0000 END

Figure 5.11  HDLC send control program example 7
5.3.2 Receive operation

DMA transfer of data of 256 bytes per frame (00-FFH) is described as a receive operation example. For the Rx INT mode, the Int on First Rx Character mode (CR1A : D4, D3 = 0, 1) is selected, and the First Rx INT Mask mode (CR2A : D6 = 1) is used for operation. Thus, all receive data (containing CRC characters) is transferred via DMA. Interrupt processing is performed in nonvector mode as in send operation.

The basic program configuration is explained below:

(1) Main program

The main program sets the receive buffer, initializes I/O devices (μPD8253-2, μPD8237A-5) and the μPD7201A, manages the receive state, etc. When receive operation starts, the Enter Hunt bit is set to 1 for flag detection. After flag detection, the program waits for the receive completion flag (which is set by the interrupt handling program when the End of Frame state occurs) to be set, and checks that the flag is set before disabling receive.

(2) Interrupt handling program

The interrupt sources handled by the interrupt handling program are Special Rx Condition interrupts only (interrupts caused by other sources are disabled). However, a check routine to see if the interrupt is an E/S interrupt is contained in the program for convenience.
In interrupt processing, End of Frame, CRC Error, and Overrun Error are checked. When any of these occurs, the control flag indicating the occurrence of the event is set. The control flag is used to manage the μPD7201A receive operation state in the main program.

Figure 5.12 shows the receive operation processing flow. Figure 5.13 shows actual program examples.
Figure 5.12 (a) HDLC receive operation main program
Note 1: This path is not actually passed through in the present program example.

Figure 5.12 (b) HDLC receive operation interrupt handling
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
<th>SYMBOL</th>
<th>ADRS</th>
</tr>
</thead>
<tbody>
<tr>
<td>BADRO</td>
<td>5B06</td>
<td>BCWC0</td>
<td>5B08</td>
<td>CHOST</td>
<td>0131</td>
<td>CHACT</td>
<td>00C1</td>
<td>CHADT</td>
<td>00C0</td>
</tr>
<tr>
<td>CHBCT</td>
<td>00C3</td>
<td>CHBDT</td>
<td>00C2</td>
<td>CHNRS</td>
<td>0018</td>
<td>CR1A</td>
<td>0048</td>
<td>CR1B</td>
<td>0004</td>
</tr>
<tr>
<td>CR2A</td>
<td>0043</td>
<td>CR2B</td>
<td>0040</td>
<td>CR3A</td>
<td>00C8</td>
<td>CR4A</td>
<td>0020</td>
<td>CR5A</td>
<td>0060</td>
</tr>
<tr>
<td>CR6A</td>
<td>0001</td>
<td>CR7A</td>
<td>007E</td>
<td>CRCFA</td>
<td>5B01</td>
<td>CREND</td>
<td>01B7</td>
<td>DM0BC</td>
<td>00D0</td>
</tr>
<tr>
<td>DM0CW</td>
<td>00D1</td>
<td>DM1BC</td>
<td>00D2</td>
<td>DM1CW</td>
<td>00D3</td>
<td>DM2BC</td>
<td>00D4</td>
<td>DM2CW</td>
<td>00D5</td>
</tr>
<tr>
<td>DM3BC</td>
<td>00D6</td>
<td>DM3CW</td>
<td>00D7</td>
<td>DM4BP</td>
<td>00DC</td>
<td>DMCMR</td>
<td>00DE</td>
<td>DMCTR</td>
<td>00D8</td>
</tr>
<tr>
<td>DMMCL</td>
<td>00DD</td>
<td>DMSMR</td>
<td>00DA</td>
<td>DMWAM</td>
<td>00DF</td>
<td>DMWM</td>
<td>00DB</td>
<td>DMWMR</td>
<td>00D9</td>
</tr>
<tr>
<td>ENIRX</td>
<td>0020</td>
<td>EO1</td>
<td>0038</td>
<td>EO1A</td>
<td>105B</td>
<td>ERESA</td>
<td>104F</td>
<td>ERRES</td>
<td>0030</td>
</tr>
<tr>
<td>ESCHA</td>
<td>1016</td>
<td>ESRES</td>
<td>0010</td>
<td>HDLCR</td>
<td>0100</td>
<td>INTEN</td>
<td>0116</td>
<td>NVINT</td>
<td>1000</td>
</tr>
<tr>
<td>OVREA</td>
<td>1038</td>
<td>OVRFA</td>
<td>5B02</td>
<td>PEND</td>
<td>0028</td>
<td>PTCO</td>
<td>00C0</td>
<td>PICT</td>
<td>00C7</td>
</tr>
<tr>
<td>RCVDT</td>
<td>1007</td>
<td>RENDA</td>
<td>5B03</td>
<td>RXBFAC</td>
<td>0104</td>
<td>RXBUF</td>
<td>4400</td>
<td>RXXRC</td>
<td>0040</td>
</tr>
<tr>
<td>RXOPR</td>
<td>01A6</td>
<td>SABRT</td>
<td>0008</td>
<td>SROAS</td>
<td>5B00</td>
<td>SRCDA</td>
<td>1022</td>
<td>STACK</td>
<td>6000</td>
</tr>
<tr>
<td>STCHA</td>
<td>0169</td>
<td>STDMA</td>
<td>0126</td>
<td>STPIT</td>
<td>0119</td>
<td>SYSTM</td>
<td>2000</td>
<td>TXCRC</td>
<td>0080</td>
</tr>
<tr>
<td>TXLRA</td>
<td>5B04</td>
<td>UNDEM</td>
<td>00C0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.13 HDLC receive control program example 1
**UCOM-85 ASSEMBLE LIST**

E STN0 ADRS OBJECT M SOURCE STATEMENTS

Figure 5.13 HDLC receive control program example 2
E STNO ADRS OBJECT M SOURCE STATEMENTS

0053 :******************************************************************************
0054 :* MPSC CONTROL REGISTER PARAMETER *
0055 :******************************************************************************
0056 :
0057 0048 CR1A EQU 48H :TXLR SET
0058 :FIRST RX INT
0059 :TX DMA DISABLE
0060 :E/S INT DISABLE
0061 0043 CR2A EQU 43H :RX INT MASK
0062 :NON-VECTORED
0063 :
0064 :85-1 VECTORED
0065 00C8 CR3A EQU 0C8H :8 BIT/CHAR
0066 :RX CRC ENABLE
0067 0020 CR4A EQU 20H :HDLC MODE
0068 0060 CR5A EQU 60H :8 BIT/CHAR
0069 :
0070 :CCITT-0
0071 0001 CR6A EQU 01H :SECONDARY ADDRESS
0072 007E CR7A EQU 7EH :FLAG
0073 :
0074 0004 CR1B EQU 4H :MODIFIED VECTOR
0075 0040 CR2B EQU 40H :VECTOR(010***00)
0076 :
0077 :******************************************************************************
0078 :* INTERRUPT JUMP TABLE (NON-VECTORED MODE) *
0079 :******************************************************************************
0080 :;
0081 0000 ORG 3CH :RST 7.5 ADRS
0082 :
0083 003C C30010 JMP NVINT :JUMP NON-VCT INT
0084 :
0085 :******************************************************************************
0086 :* PERIPHERAL INITIALIZE SEQUENCE *
0087 :******************************************************************************
0088 :
0089 003F ORG 100H
0090 :
0091 0100 310060 HDLCR: LXI SP.STACK
0092 0103 AF XRA A
0093 0104 210044 RXBFC: LXI H.RXBUF :CLR MPSC RX BUF
0094 0107 77 MOV M.A
0095 0108 2C INR L
0096 0109 C20701 JNZ $-2
0097 010C AF XRA A :CLEAR FLAGS
0098 010D 32015B STA CRCFA
0099 0110 32025B STA OVRFA
0100 0113 32035B STA RENDA
0101 0116 3E1B INTEN: MVI A.1BH :RST 7.5 ENABLE
0102 0118 30 SIM
0103 :
0104 :
0105 :

Figure 5.13 HDLC receive control program example 3
Figure 5.13 HDLC receive control program example 4
E STNO ADRS OBJECT M SOURCE STATEMENTS

0159 0161 3EO1 MVI A.1H :PTR 1B
0160 0163 D3C3 OUT CHBCT
0161 0165 3EO4 MVI A.CRI1B :MODIFIED VECTOR
0162 0167 D3C3 OUT CHBCT
0163 :
0164 : CH-A OPERATION MODE
0165 :
0166 0169 3EO4 STCHA: MVI A.4H :PTR 4A
0167 016B D3C1 OUT CHACT
0168 016D 3E20 MVI A.CR4A :CH-A MODE
0169 016F D3C1 OUT CHACT
0170 0171 3EO5 MVI A.5H :PTR 5A
0171 0173 D3C1 OUT CHACT
0172 0175 3E60 MVI A.CR5A :TX PARAMETER
0173 0177 D3C1 OUT CHACT
0174 0179 3EO6 MVI A.6H :PTR 6A
0175 017B D3C1 OUT CHACT
0176 017D 3EO1 MVI A.CR6A :2ND ADDRESS
0177 017F D3C1 OUT CHACT
0178 0181 3EO7 MVI A.7H :PTR 7A
0179 0183 D3C1 OUT CHACT
0180 0185 3E7E MVI A.CR7A :FLAG
0181 0187 D3C1 OUT CHACT
0182 0189 3E11 MVI A.11H :PTR 1A.RS ES INT
0183 018B D3C1 OUT CHACT
0184 018D 3E48 MVI A.CRI1A :INT/DMA MODE
0185 018F D3C1 OUT CHACT
0186 0191 2A045B LHLID TXLRA :TXLRA
0187 0194 7D MOV A.L :TXLRA-L
0188 0195 D3C1 OUT CHACT
0189 0197 7C MOV A.H :TXLRA-H
0190 0198 D3C1 OUT CHACT
0191 019A 3E03 MVI A.3H :PTR 3A
0192 019C D3C1 OUT CHACT
0193 019E 3EC8 MVI A.CR3A :RX PARAMETER
0194 01AO D3C1 OUT CHACT
0195 01A2 3E10 MVI A.ESRES :RESET E/S INT
0196 01A4 D3C1 OUT CHACT
0197 :
0198 01A6 DBC1 RXOPR: IN CHACT :READ FIRST SROA
0199 01AB 32005B STA SR0AS
0200 01AB AF XRA A
0201 01AC D3D8 OUT DMCTR :DMAC ENABLE
0202 01AE 3E03 MVI A.3H :PTR 3A
0203 01B0 D3C1 OUT CHACT
0204 01B2 3ED9 MVI A.0D9H :RXEN.ENTER HUNT
0205 01B4 D3C1 OUT CHACT
0206 01B6 FB EI
0207 :
0208 01B7 3A035B CREND: LDA RENDA :CHECK RXEND FLG
0209 01BA FEO1 CPI 1H
0210 01BC C2B701 JNZ CREND
0211 :

Figure 5.13 HDLC receive control program example 5
0212 01BF 3E03 MVI A.3H :PTR 3A
0213 01C1 D3C1 OUT CHACT
0214 01C3 3EC0 MVI A.0C0H :RX DISABLE
0215 01C5 D3C1 OUT CHACT
0216 01C7 F3 DI
0217
0218 01C8 C30020 JMP SYSTM
0219
0220
0221 : INERRUPT PROCESSING ROUTINE (NON-VECTORED MODE)
0222
0223 :*************************************************
0224
0225 01CB ORG 1000H
0226
0227 1000 F5 NVINT: PUSH PSW
0228 1001 3E02 MVI A.2H :PTR 2B
0229 1003 D3C3 OUT CHBCT
0230 1005 DBC3 IN CHBCT :READ SR2B
0231 1007 E61C RCVDT: ANI 1CH :MASK
0232 1009 FE1C CPI 1CH :CH-A SRXCN?D?
0233 100B CA2210 JZ SRCDA :IF YES.JUMP
0234 100E FE14 CPI 14H :CH-A E/S CHNG?
0235 1010 CA1610 JZ ESCHA :IF YES.JUMP
0236 1013 C35B10 JMP EO1A :IF OTHS.JP EO1
0237 1016 DBC1 ESCHA: IN CHACT :READ SROA
0238 1018 32005B STA SROAS :STORE NEW SROA
0239 101B 3E10 MVI A.ESRES :RESET E/S INT
0240 101D D3C1 OUT CHACT
0241 101F C35B10 JMP EO1A
0242 1022 3E01 SRCDA: MVI A.1H :PTR 1A
0243 1024 D3C1 OUT CHACT
0244 1026 DBC1 IN CHACT :READ SR1A
0245 1028 07 RLC
0246 1029 D23810 JNC OVREA :IF NOT EOF.JMP
0247 102C 07 RLC :CHECK CRC ERR
0248 102D D24F10 JNC ERESA :IF CRC OK.JUMP
0249 1030 3E01 MVI A.1 :SET CRC ERR FLG
0250 1032 32015B STA CRCFA :STORE CRCCE FLG
0251 1035 C34F10 JMP ERESA
0252
0253 1038 3E03 OVREA: MVI A.3H :PTR 3A
0254 103A D3C1 OUT CHACT
0255 103C 3EC8 MVI A.0C8H :MPSC RX DISABLE
0256 103E D3C1 OUT CHACT
0257 1040 3E04 MVI A.4H :MASK DMAC CH-0
0258 1042 D3DA OUT DMSMR
0259 1044 DBC0 IN CHADT :DAMMY READ
0260 1046 DBC0 IN CHADT
0261 1048 DBC0 IN CHADT
0262 104A 3E01 MVI A.1 :SET OVRNER FLG
0263 104C 32025B STA OVRFA :STORE OVRNER FG
0264

Figure 5.13 HDLC receive control program example 6
Figure 5.13 HDLC receive control program example 7
Chapter 6 shows the state transition for each µPD7201A protocol operation by using figures and tables.

Compare these with the µPD7201A operation described in Chapters 3 to 5.

° Protocols

1) Asynchronous
   Figures 6.1 and 6.2 and Tables 6.1 and 6.2

2) SYNC
   Figures 6.3 and 6.4 and Tables 6.3 and 6.4

3) HDLC
   Figures 6.5 and 6.6 and Tables 6.5 and 6.6
Figure 6.1 Asynchronous send state transition diagram
Table 6.1 Asynchronous send state transition diagram
explanation table

<table>
<thead>
<tr>
<th>No.</th>
<th>State name</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mark sending 1</td>
<td>Initial state. When Tx is disabled and no send data exists, TxD is placed in mark state (1).</td>
</tr>
<tr>
<td>2</td>
<td>Mark sending 2</td>
<td>When Tx is enabled and no send data exists, TxD is placed in mark state (1).</td>
</tr>
<tr>
<td>3</td>
<td>Mark sending 3</td>
<td>When Tx is disabled and send data exists, TxD is placed in mark state (1).</td>
</tr>
<tr>
<td>4</td>
<td>Data sending</td>
<td>Data in Tx shift register is sent.</td>
</tr>
<tr>
<td>5</td>
<td>Break sending</td>
<td>Space (0) is sent and send data is destroyed.</td>
</tr>
</tbody>
</table>
Figure 6.2 Asynchronous receive state transition diagram
Table 6.2 Asynchronous receive state transition diagram
explanation table

<table>
<thead>
<tr>
<th>No.</th>
<th>State name</th>
<th>Operation</th>
</tr>
</thead>
</table>
| 1   | Input ignored            | Initial state.  
The RxD state is ignored.                                                |
| 2   | Wait for character      | A wait for start bit is continued when mark is being received after Rx is enabled. |
| 3   | Character receiving     | After start bit is detected, data is assembled.                            |
| 4   | Break detection         | If the RxD state is one-character mark, break detection is made.            |
Figure 6.3 SYNC send state transition diagram
Table 6.3  SYNC send state transition diagram explanation table

<table>
<thead>
<tr>
<th>No.</th>
<th>State name</th>
<th>Operation</th>
</tr>
</thead>
</table>
| 1   | Mark sending 1      | Initial state.  
|     |                     | In TxD, mark is sent.                                                     |
| 2   | Mark sending 2      | When Tx is disabled and Tx data exists, mark is sent.                     |
| 3   | SYNC sending        | When Tx is enabled and no Tx data exists, SYNC character is automatically sent. |
| 4   | Character sending   | When Tx is enabled and Tx data exists, data is sent.                      |
| 5   | BCC sending         | When CRC is enabled, CRC2 byte is automatically sent.                     |
Figure 6.4  SYNC receive state transition diagram
Table 6.4  SYNC receive state transition diagram explanation table

<table>
<thead>
<tr>
<th>No.</th>
<th>State name</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Input ignored</td>
<td>Initial state. The RxD state is ignored.</td>
</tr>
<tr>
<td>2</td>
<td>Wait for SYNC</td>
<td>After Rx is enabled and Enter Hunt phase command is issued, SYNC character detection state is set. (during internal synchronization detection mode)</td>
</tr>
<tr>
<td>3</td>
<td>Character receiving</td>
<td>Character is assembled every data bit count after synchronization is set.</td>
</tr>
</tbody>
</table>
Figure 6.5 HDLC send state transition diagram (when Tx Length Counter is used and CRC exists)
Table 6.5 HDLC send state transition diagram explanation table

<table>
<thead>
<tr>
<th>No.</th>
<th>State name</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mark sending</td>
<td>Initial state.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In TxD, mark is sent.</td>
</tr>
<tr>
<td>2</td>
<td>Flag sending</td>
<td>Flag is automatically sent.</td>
</tr>
<tr>
<td>3</td>
<td>Character sending</td>
<td>Data is sent. (Zero insertion function is contained.)</td>
</tr>
<tr>
<td>4</td>
<td>FCS sending</td>
<td>16 bits of CRC are automatically sent. (Zero insertion function is contained.)</td>
</tr>
<tr>
<td>5</td>
<td>Abort sending</td>
<td>1 of 8-13 bits is sent.</td>
</tr>
</tbody>
</table>
Figure 6.6 HDLC receive state transition diagram (with no address search)
Table 6.6 HDLC receive state transition diagram
explanation table

<table>
<thead>
<tr>
<th>No.</th>
<th>State name</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Input ignored</td>
<td>Initial state. RxD input is ignored.</td>
</tr>
<tr>
<td>2</td>
<td>Wait for flag</td>
<td>When Rx is enabled and Enter Hunt phase command is issued, start flag detection is started.</td>
</tr>
<tr>
<td>3</td>
<td>Wait for character</td>
<td>When start flag is detected, a wait for data (bit pattern other than flags) is continued.</td>
</tr>
<tr>
<td>4</td>
<td>Character receiving</td>
<td>After data is detected (normal address field), character is assembled every data bit count. (Zero deletion function is contained.)</td>
</tr>
<tr>
<td>4</td>
<td>End/start flag receiving</td>
<td>After A, C, I, FCS field is received, flag (end flag) is received. Character synchronization is made by subsequent flag idle.</td>
</tr>
<tr>
<td>5</td>
<td>Abort detection</td>
<td>Abort is detected by receiving 1 of 7 bit or more after synchronization is set.</td>
</tr>
</tbody>
</table>
μPD72001

ADVANCED MULTIPROTOCOL SERIAL CONTROLLER
INTRODUCTION

The new uPD72001 Advanced Multiprotocol Serial Controller (AMPSC) is a high performance, single chip CMOS controller applicable to a wide range of high speed serial data communication purposes. Located between a general purpose microprocessor and a data communication equipment, the AMPSC provides data conversion according to the predetermined format on two independent serial full-duplex channels. The basic protocols are asynchronous, character oriented (COP) and bit oriented (BOP) as HDLC or SDLC. Data formats NRZ, NRZI and FM can be selected. The maximum transmission speed is 1.6 Mbits/s. Extended DMA and interrupt capabilities, on-chip crystal oscillator, baud rate generators and digital PLL circuit provide an easy hardware implementation. The baud rate generators are independent on transmit and receive and independent on both channels. The internal register structure allows 16 bit transmit data. Various types of error detection assure a highly reliable data transmission. The AMPSC is available now in 52 pin PLCC (uPD72001L) and 40 pin plastic DIP (72001C).

Features
- LSI to high-performance data communications
- Multiprotocol operations:
  - Asynchronous
    - Character-oriented protocol (COP)
    - Bit-oriented protocol (BOP)
- Two full-duplex channels
- Baud rate: DC to 1.6 Mbps
- Transmitter: Double buffer
- Receiver: Quadruple buffer
- Interrupt control capability
- DMA request signal: Transmit and Receive request/channel
- Overrun error detection
- Data formats: NRZ, NRZI, and FM
- Local self test capability
- On-chip crystal oscillator
- On-chip DPLL circuit
- On-chip baud rate generator
- General-purpose input/output pins: 4 pins per channel
- Standby function
- System clock rate: 8 MHz max.
- CMOS technology
- 40-pin DIP
- Single +5 V power supply

2-1
Asynchronous operation
- Character bit length: 5, 6, 7, or 8 bits
- Stop bit length: 1, 1.5, or 2 bits
- Clock rate: x1, x16, x32, or x64
- Parity generation/checking
- Framing error detection
- Break generation/detection

Character-oriented protocol (COP) operation
- Mono-sync, Bi-sync, or External sync mode
- Character bit length: 5, 6, 7, or 8 bits
- Character synchronization: Internal or external
- SYNC character bit length: 6 or 8 bits
- BCS generation/checking (CRC-16, CRC-CCITT)
- Parity generation/checking
- Automatic SYNC character transmission/checking/deletion

Bit-oriented protocol (BOP) operation
- HDLC, SDLC, or SDLC Loop mode
- Flag transmission/detection
- Zero insertion/suppression
- Address field detection
- FCS generation/check (CCITT-1)
- Short frame detection
- Automatic Abort transmission/detection
- Idle detection
- Go-Ahead detection
- Transmit data count control (16-bit transmit length register/counter)
Pin Configuration (Top View)
1. PIN FUNCTIONS

The AMPSC functions are basically divided in two: the system interface function which controls interfacing with a host system, and the data transmit/receive function. This section describes the AMPSC pin functions relating to system interface and send/receive control separately.

The pin functions are closely related to control register (CR) settings which specify device operations. In the following descriptions of pin functions, the related CR settings are referred to wherever needed. Refer to section 3, Register Configurations, as needed.

In the following descriptions, the input/output status of each pin is expressed, as a rule, by "H" (voltage level satisfying \( V_{IH} \) for input, and \( V_{OH} \) for output) and "L" (voltage level satisfying \( V_{IL} \) for input, and \( V_{OL} \) for output).

1.1 System Interface Pins

(1) VDD
Power supply input pin.

(2) GND
Ground pin.

(3) RESET (Reset) --- Input
This pin inputs an external Reset signal which resets the AMPSC. Applying a "L" signal continuously for 2 or more clock cycles (2 tcy) to this pin resets the device (system reset). A system reset causes the transmitter, receiver, and interrupt and DMA functions to be disabled, and the TxD and general-purpose output pins to be set to "H". It also resets (to zeros) all bits of the control register (CR). Therefore, once a system reset is executed, the CR must be set up again. Table 1-1 shows the pin status after system reset, in comparison with the pin status after channel reset (CR0: \( D5, D4, D3 = 0, 1, 1 \)). For the CR and SR status after system reset, see section 3.1, Outline of Registers.

Upon system reset, the AMPSC automatically enters the standby mode, in which power consumption is reduced.
Table 1-1 Pin Status after Reset

<table>
<thead>
<tr>
<th>Pin name</th>
<th>I/O</th>
<th>Pin status</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RESET (system reset)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WR</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RD</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>B/A</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>C/D</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>D7-D0</td>
<td>I/O</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>INT</td>
<td>0</td>
<td>High Impedance</td>
<td></td>
</tr>
<tr>
<td>INTAK</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PRI</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PRO</td>
<td>0</td>
<td>Depends on PRI</td>
<td>Depends on PRI</td>
</tr>
<tr>
<td>DRQTxA</td>
<td>0</td>
<td>&quot;L&quot;</td>
<td>&quot;L&quot;</td>
</tr>
<tr>
<td>DRQRxA</td>
<td>0</td>
<td>&quot;L&quot;</td>
<td>&quot;L&quot;</td>
</tr>
<tr>
<td>DTRxA, DTRxS</td>
<td></td>
<td>Enters DTR function and becomes &quot;H&quot;</td>
<td>Holds status</td>
</tr>
<tr>
<td>TxA,TxB</td>
<td>0</td>
<td>&quot;H&quot;</td>
<td>&quot;H&quot;</td>
</tr>
<tr>
<td>RxDA,RxDB</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>TXCA, TXCB</td>
<td>I/O</td>
<td>Input state</td>
<td>Holds current status</td>
</tr>
<tr>
<td>XI1A/STxCA</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XI1B/STxCB</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>XI2A/SYNCA</td>
<td>I/O</td>
<td>Input state</td>
<td>Input state</td>
</tr>
<tr>
<td>XI2B/SYNCE</td>
<td>I/O</td>
<td>Input state</td>
<td>Input state</td>
</tr>
<tr>
<td>RTS, RSB</td>
<td>0</td>
<td>&quot;H&quot;</td>
<td>&quot;H&quot;</td>
</tr>
<tr>
<td>CTS, CTE</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>DCDA, DCDE</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

(4) **CLK (System Clock) --- Input**
This pin inputs the system clock. The system clock rate must be more than five times as high as the data transfer rate.

(5) **WR (Write) --- Input (active Low)**
This pin inputs a control signal used to write control words or Tx data to the device.

(6) **RD (Read) --- Input (active Low)**
This pin inputs a control signal used to read status or Rx data out of the device.
(7) B/Å (Channel B/Channel A) --- Input
This pin inputs a channel select signal which selects the channel to be accessed for write or read operation. "L" input to this pin selects channel A; "H" input selects channel B.

(8) C/Ð (Control/Data) --- Input
This pin inputs a signal which determines the type of data on the data bus during read or write access to the device.

Table 1-2 shows the status of WR, RD, B/Å, and C/Ð and corresponding operations.

Table 1-2 AMPSC Control Signals Versus Operations

<table>
<thead>
<tr>
<th>WR</th>
<th>RD</th>
<th>B/Å</th>
<th>C/Ð</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>Channel A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H</td>
<td></td>
<td>Writes transmit data to Tx buffer</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>Channel A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H</td>
<td></td>
<td>Reads receive data from Rx buffer</td>
</tr>
<tr>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>Channel A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H</td>
<td></td>
<td>Writes to control register</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>Channel A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H</td>
<td></td>
<td>Reads status register</td>
</tr>
<tr>
<td>H</td>
<td>H</td>
<td>X</td>
<td>X</td>
<td>High-impedance state</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>X</td>
<td>X</td>
<td>Use inhibited</td>
</tr>
</tbody>
</table>

X:Don't Care

(9) D7-D0 (Data Bus) --- Input/Output
These pins constitute a three-state, eight-bit, bidirectional data bus. This bus is connected to the host processor's data bus to transfer control words, status, and transmit/receive data.

(10) INT (Interrupt) --- Output (open drain)
This pin outputs the interrupt request signal. This pin is set to an active "L" if an interrupt condition occurred within the AMPSC. Being an open-drain output, this pin requires a pull-up resistor.
11) **INTAK (Interrupt Acknowledge) --- Input (active Low)**

This pin inputs an acknowledge signal returned in response to an interrupt request. This pin is used to select the Vector mode (CR2A: D7 = 1). When selecting the Non-Vector mode (CR2A: D7 = 0), this pin must be pulled up to "H".

12) **PRI (Priority Input) --- Input (active Low)**

This pin inputs a signal which controls interrupt request generation and interrupt vector output. This pin serves to interrupt generation control during general operation, and to interrupt vector output control during the INTAK sequence. The handling of this pin depends on the interrupt mode.

(a) Vectored mode (D7 of CR2A = 1)

The PRI pin usually controls the generation of interrupts. If Type A-3 or Type B-2 (D5, D4, D3 of CR2A = 0, 1, 0 or 1, 0, 0) interrupt vector output mode is selected, the PRI pin can be set to either "H" or "L". If any other interrupt vector output mode is selected, the PRI pin must be set to "L" to enable the interrupt request. During the INTAK sequence, input of "L" level to the PRI pin causes the device to enable interrupt vector output in any interrupt vector output mode. If the PRI pin is set to "H", interrupt vector output is disabled.

(b) Non-Vectored mode (D7 of CR2A = 0)

Since this mode has no INTAK sequence, the PRI pin serves only for interrupt generation control. If a vector output mode other than Type A-3 and Type B-2 is selected, input of "L" level to the PRI pin enables interrupt generation. If the pin is set to "H", interrupt generation is disabled. When a daisy chain is configured for interrupt, "L" input to this pin indicates that the interrupt from a device with higher priority is not serviced or the device is not in request for interrupt service. Only the AMPSC with "L" applied to this pin can request interrupts.
(13) **PRO (Priority Output) --- Output (active Low)**
This pin is used to configure an interrupt daisy chain. It controls interrupt requests from a device with a lower order of priority. It is usually used with the PRI pin, and is set to the following status depending on the PRI status:
- If PRI = "H", PRO = "H".
- If PRI = "L", PRO = "H" provided an interrupt request is present, and PRO = "L" provided no interrupt request is present.

(14) **DRQTxA (DMA Request TxA) --- Output (active High)**
This is a DMA request output to the DMA controller. This pin is set to "H" when the Tx buffer in transmitter channel A is emptied; The conditions under which this pin is set to "H" differ depending on the status of D2 of CR1:
- CR1A: D2 = 0: DRQTxA is set to "H" when the Tx buffer is empty after the first Tx data was written into the buffer. If remains at "L" when the buffer is empty by reset operation.
- CR1A: D2 = 1: DRQTxA is set to "H" when the Tx buffer is empty.
The DRQTxA is reset when transmit data is written into channel A.

(15) **DRQRxA (DMA Request RxA) --- Output (active High)**
This is a DMA request output to the DMA controller. This pin is set to "H" when the receiver on channel A enters the Rx Character Available state. It is reset when received data is read out of channel A.

(16) **DTRA/DRQTxB (Data Terminal Ready A/DMA Request TxB) --- Output**
This pin serves the following two functions depending on the status of bits D1 and D0 of CR2A.
(a) When CR2A: D1, D0 = 0, 0 or 0, 1
This pin functions as DTRA, which is a general-purpose output usable for modem control or other purposes. The DTRA pin status is as follows:
- When CR5A: D7 = 0: DTRA = "H"
- When CR5A: D7 = 1: DTRA = "L"
(b) When CR2A: D1, D0 = 0, 1
This pin functions as a DROTxB output. Its function is identical to that of the DROTxA pin, with the exception that the former is used on channel B.

(17) DTRB/DRCRxB (Data Terminal Ready B/DMA request RxB) --- Output
This pin serves the following two functions depending on the status of bits D1 and D0 of DR2A.
(a) When D1, D0 of CR2A = 0, 0 or 0, 1
This pin serves as the DTRB output. Its function is identical to that of the DTRA pin, with the exception that the former is used on channel B.
(b) When D1, D0 of CR2A = 1, 0
This pin serves as the DRQRxB output. Its function is identical to that of the DRQRxA pin, with the exception that the former is used on channel B.

(18) CTSA (Clear to Send A), CTSB (Clear to Send B) --- Inputs
These pins are general-purpose inputs usable for modem control or other purposes. A status change on these pins affects E/S bit latch operation. If E/S INT is enabled (D0 of CR1 set to 1), an E/S interrupt occurs. When the Auto Enable mode is selected (D5 of CR3 set to 1), these pins can be used with the Tx Enable bit (D3 of CR5) to control the transmitter status. This is shown in Table 1-3.

<table>
<thead>
<tr>
<th>CTS pin</th>
<th>Tx Enable bit</th>
<th>Transmitter status</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>1</td>
<td>Enable</td>
</tr>
<tr>
<td>H</td>
<td>1</td>
<td>Disable</td>
</tr>
<tr>
<td>H or L</td>
<td>0</td>
<td>Disable</td>
</tr>
</tbody>
</table>
1.2 Pins Relating to Transmit/Receive Operations

(1) **TxDA (Transmit Data A), TxDB (Transmit Data B) --- Outputs**
These are Tx data output pins.
(2) RxDA (Receive Data A), RxDB (Receive Data B) --- Inputs
These are Rx data input pins.

(3) X1IA/STRxCA (Crystal Input 1A/Source of Transmit Receive Clock A)
X1IB/STRxCB (Crystal Input 1B/Source of Transmit Receive Clock B) --- Inputs
The function of these pins depends on the status of D7 of CR15:
(a) When D7 of CR15 is 0
These pins function as STRxC, which inputs a transmit/receive clock or a source clock for the on-chip baud rate generator (BRG) or digital phase locked loop (DPLL).
(b) When D7 of CR15 is 1
These pins function as X1I, which, along with pin XI2, connects with a crystal resonator for TxRx CLK source oscillation.

(4) XI2A/SYNCA (Crystal Input 2A/Synchronization A) --- Inputs,
XI2B/SYNCB (Crystal Input 2B/Synchronization B) Output
The function of these pins depends on the status of D7 of CR15:
(a) When D7 of CR15 is 0
These pins function as SYNCh, which provides the functions as shown in Table 1-6 depending on the setting of CR4.
(b) When D7 of CR15 is 1
These pins function as XI2, which, along with pin XI1, connects with a crystal resonator for TxRx CLK source oscillation.
<table>
<thead>
<tr>
<th>Operation Protocol</th>
<th>Sync detection method</th>
<th>SYNC pin function</th>
<th>CR4</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Async</td>
<td>Input</td>
<td>X</td>
<td>0 1</td>
<td>Functions as a general-purpose input. Status changes at this pin (&quot;H&quot; to &quot;L&quot; or &quot;L&quot; to &quot;H&quot;) affect Sync/Hunt bit (D4 of SR1) latch operation, and cause an E/S interrupt.</td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td>X</td>
<td>0 1</td>
<td>If a SYNC character is detected in the received data, this pin is set to &quot;L&quot; for 1 RXC cycle period.</td>
</tr>
<tr>
<td>COP</td>
<td>Input</td>
<td>0 0</td>
<td>0 0</td>
<td>Accepts character synchronization signal. If this pin is set from &quot;H&quot; to &quot;L&quot;, the device exits the Hunt phase and establishes character synchronization. Character synchronization is maintained while the SYNC input is at &quot;L&quot;. Receive character assembling is initiated at the rising edge of the receive clock which precedes the falling edge of the SYNC input.</td>
</tr>
<tr>
<td>BOP</td>
<td>No function</td>
<td>X</td>
<td>1 0</td>
<td>SYNC pin does not function.</td>
</tr>
</tbody>
</table>

X : Don't Care

Table 1-6SYNC Pin Functions versus CR4 Value (D7 of CR15 = 0)
(5) TRxCA (Transmit Receive Clock A) --- Input,Output
    TRxCB (Transmit Receive Clock B) --- Input,Output
(a) If D2 of CR15 = 0
    These pins function as transmit/receive clock inputs. They input an external transmit/receive clock.
    (Exception): If at least one of the conditions, (D6, D5 of CR15 = 0, 1) or (D4, D3 of CR15 = 0, 1) is satisfied, these pins function as inputs even if D2 of CR15 is set to 1.
(b) When bit D2 of CR15 = 1
    These pins function as outputs. The output clock source is selectable from a crystal oscillator, BRG, DPLL, and transmit clock depending on the setting of D1 and D0 of CR15. However, when the condition given in the (Exception) of the above item (a) is fulfilled, these pins unconditionally function as inputs, in which case the setting of D2, D1, and D0 of CR15 is invalid.
2. CONFIGURATION

The basic function of the AMPSC is to control serial data transmission/reception with other serial data communication devices. The AMPSC contains a flexible architecture to efficiently implement this function.

The internal logics of the AMPSC can be divided into system clock, system interface, and transmit/receive blocks. The system clock section supplies the system clock which is the base used to control the whole operation of the AMPSC's other internal circuits. The system interface block controls interfacing with its host system. The transmit/receive block controls data transmit/receive sequences. Fig. 2-1 shows the internal circuit configuration of the AMPSC.

![Fig. 2-1 AMPSC Internal Circuit Configuration](image)

2.1 System Clock Section
The system clock block generates an internal reference clock based on the system clock applied to the CLK pin. This reference clock is supplied to other internal blocks to synchronize device operations.

2.2 System Interface Block
The system interface block consists of an interface control logic and registers. The interface control logic contains a read/write controller, an interrupt/DMA controller, and a data bus buffer. The registers consist of control and status registers.
2.2.1 Interface control circuit

The interface control circuit controls interfacing between the AMPSC and its host system. Data transfer between a device and host system is controlled by the read/write controller and interrupt/DMA controller via an 8-bit data bus buffer. The type of data on the data bus, channel selection, and direction of data transfer are controlled by control inputs. Table 2-1 shows combinations of control input signals and corresponding selections.

Table 2-1 Control Inputs Versus Selected Functions

<table>
<thead>
<tr>
<th>B/A</th>
<th>C/D</th>
<th>RD</th>
<th>WR</th>
<th>INTAK</th>
<th>PRI</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>L H</td>
<td>L L</td>
<td>H H</td>
<td>X(1)</td>
<td>Channel A Reads receive data Channel B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L H</td>
<td>L H</td>
<td>L L</td>
<td>X</td>
<td>Channel A Writes transmit data Channel B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L H</td>
<td>H L</td>
<td>H H</td>
<td>X</td>
<td>Channel A Reads status register Channel B (SR)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L H</td>
<td>H H</td>
<td>L L</td>
<td>X</td>
<td>Channel A Writes data to control Channel B register (CR)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X X</td>
<td>X H</td>
<td>H L</td>
<td>L L</td>
<td>Interrupt acknowledgment sequence (2)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes: 1. X denotes "Don't care".
2. Data to be output differs depending on vector type.

2.2.2 Registers

The registers within the AMPSC are used to set AMPSC operation or to indicate the device status. Each channel of the AMPSC has 20 control registers (CRs) for setting operation mode and operation control, and 12 types of status registers (SRs) for status indication. Control words are written into these registers by the host processor. The status registers hold device status information. The status of the AMPSC can be determined by reading these registers. For more information about the CRs and SRs, see section 3, Register Configuration.
2.3 Transmitter/Receiver Block
The transmitter/receiver block consists of two independent full-duplex channels, its control circuit and transmit/receive clock circuits. The operation of each transmit/receive channel is selectable from asynchronous, COP (bi-sync, etc.), and BOP (HDLC, SDLC, etc.) protocols, and is controlled by the transmit/receive control circuit. This block also contains a baud rate generator and digital PLL to supply clocks for serial data transmission and reception.
3. REGISTER CONFIGURATION

To implement operations under different communication protocols, the AMPSC contains a large group of registers. This section details the register configuration within the AMPSC and the function of each register.

The descriptions in this section assume that the reader already has some basic knowledge about each communication protocol. For details of communication protocols, refer to the respective standards for protocols. Unless otherwise stated, the following descriptions assume that relevant interrupts are enabled.

3.1 Outline of Registers

The AMPSC's internal registers are divided into control registers (CR) and status registers (SR). Control registers are used to set up the device operation mode or to control device operations. Control words are written into these registers by the host processor. Status registers hold device status information. The host processor can monitor the AMPSC's status by reading these status registers. The AMPSC's internal register configuration is shown in Fig. 3-1. The functions of the registers are outlined in Table 3-1. The initial values of the registers after resetting and the register status in the Standby mode are shown in Table 3-2.
Note: 1. Shared with both channels, though the function of this register differs on each channel.
2. These registers are used in pairs.
3. These registers are made available by setting CR12.
4. These registers do not exist on other channels.
5. The registers listed above are provided on each channel, with some exceptions.

**Fig. 3-1** AMPSC Internal Register Configuration
### Table 3-1 Outline of AMPSC's Internal Register Functions

<table>
<thead>
<tr>
<th>Type</th>
<th>Reg. Name</th>
<th>Channel</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control registers</td>
<td>CR0</td>
<td>A, B</td>
<td>Register selection pointer, control commands, or initialize commands</td>
</tr>
<tr>
<td></td>
<td>CR1</td>
<td>A, B</td>
<td>Interrupt/DMA control, received data transfer mode</td>
</tr>
<tr>
<td></td>
<td>CR2</td>
<td>A</td>
<td>System Interface mode, Interrupt/DMA mode, Vector mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B</td>
<td>Initial value of interrupt vector</td>
</tr>
<tr>
<td></td>
<td>CR3</td>
<td>A, B</td>
<td>Receive control and Auto Enable control</td>
</tr>
<tr>
<td></td>
<td>CR4</td>
<td>A, B</td>
<td>Operation protocol mode, parity control</td>
</tr>
<tr>
<td></td>
<td>CR5</td>
<td>A, B</td>
<td>Transmission control</td>
</tr>
<tr>
<td></td>
<td>CR6</td>
<td>A, B</td>
<td>SYNC character/address</td>
</tr>
<tr>
<td></td>
<td>CR7</td>
<td>A, B</td>
<td>SYNC character/flag pattern</td>
</tr>
<tr>
<td></td>
<td>CR8</td>
<td>A, B</td>
<td>Transmit data count (lower 8 bits) (TxDL-L)</td>
</tr>
<tr>
<td></td>
<td>CR9</td>
<td>A, B</td>
<td>Transmit data count (higher 8 bits) (TxDL-H)</td>
</tr>
<tr>
<td></td>
<td>CR10</td>
<td>A, B</td>
<td>Data format, loop mode control, SYNC character bit length</td>
</tr>
<tr>
<td></td>
<td>CR11</td>
<td>A, B</td>
<td>E/S interrupt control</td>
</tr>
<tr>
<td></td>
<td>CR12</td>
<td>A, B</td>
<td>TRxC pin/DPLL source selection, Tx/RxBRG interrupt control, Tx/RxBRG register control</td>
</tr>
<tr>
<td></td>
<td>RxBRG-L</td>
<td>A, B</td>
<td>RxBRG count value (lower 8 bits)</td>
</tr>
<tr>
<td></td>
<td>RxBRG-H</td>
<td>A, B</td>
<td>RxBRG count value (higher 8 bits)</td>
</tr>
<tr>
<td></td>
<td>TxBRG-L</td>
<td>A, B</td>
<td>TxBRG count value (lower 8 bits)</td>
</tr>
<tr>
<td></td>
<td>TxBRG-H</td>
<td>A, B</td>
<td>TxBRG count value (higher 8 bits)</td>
</tr>
<tr>
<td></td>
<td>CR13</td>
<td>A, B</td>
<td>TxDLC control, standby mode control</td>
</tr>
<tr>
<td></td>
<td>CR14</td>
<td>A, B</td>
<td>DPLL control, test mode control, Tx/Rx BRG control</td>
</tr>
<tr>
<td></td>
<td>CR15</td>
<td>A, B</td>
<td>Crystal oscillator control, Tx/Rx CLK source selection, TRxC pin control</td>
</tr>
<tr>
<td>Status registers</td>
<td>SR0</td>
<td>A, B</td>
<td>Transmit/receive buffer status, Special Rx Condition status</td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
<td>------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>SR1</td>
<td>A, B</td>
<td>E/S bit</td>
<td></td>
</tr>
<tr>
<td>SR2</td>
<td>B</td>
<td>Interrupt vector</td>
<td></td>
</tr>
<tr>
<td>SR3</td>
<td>A, B</td>
<td>Residue Code, Tx/Rx BRG Zero Count</td>
<td></td>
</tr>
<tr>
<td>SR4</td>
<td>A</td>
<td>INT Pending bit</td>
<td></td>
</tr>
<tr>
<td>SR8</td>
<td>A, B</td>
<td>TxDLC-L (lower 8 bits)</td>
<td></td>
</tr>
<tr>
<td>SR9</td>
<td>A, B</td>
<td>TxDLC-H (higher 8 bits)</td>
<td></td>
</tr>
<tr>
<td>SR10</td>
<td>A, B</td>
<td>DPLL CLK Missing status, Loop status</td>
<td></td>
</tr>
<tr>
<td>SR11</td>
<td>A, B</td>
<td>Interrupt Enable status (contents of CR11)</td>
<td></td>
</tr>
<tr>
<td>SR12</td>
<td>A, B</td>
<td>RxBRG count value (lower 8 bits)</td>
<td></td>
</tr>
<tr>
<td>SR13</td>
<td>A, B</td>
<td>RxBRG count value (higher 8 bits)</td>
<td></td>
</tr>
<tr>
<td>SR14</td>
<td>A, B</td>
<td>TxBRG count value (lower 8 bits)</td>
<td></td>
</tr>
<tr>
<td>SR15</td>
<td>A, B</td>
<td>TxBRG count value (higher 8 bits)</td>
<td></td>
</tr>
<tr>
<td>Register name</td>
<td>Bit</td>
<td>D7</td>
<td>D6</td>
</tr>
<tr>
<td>---------------</td>
<td>-----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td><strong>CR0</strong></td>
<td>Function</td>
<td>CRC Control</td>
<td>Command</td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>Channel reset</td>
<td>X</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CR1</strong></td>
<td>Function</td>
<td>Short Frame Error Detect</td>
<td>Overrun Rx INT Mask</td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Channel reset</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>CR2A</strong></td>
<td>Function</td>
<td>Vector Status</td>
<td>Affects Vector</td>
</tr>
<tr>
<td>System reset</td>
<td>0</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>Channel reset</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>CR2B</strong></td>
<td>Function</td>
<td>Initial Vector Value</td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CR3</strong></td>
<td>Function</td>
<td>Rx Character Bit Length</td>
<td>Auto Enter Hunt Phase</td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>Channel reset</td>
<td>-</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>CR4</strong></td>
<td>Function</td>
<td>Clock Rate</td>
<td>Protocol Mode</td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Channel reset</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>CR5</strong></td>
<td>Function</td>
<td>DTR Control</td>
<td>Tx Character Bit Length</td>
</tr>
<tr>
<td>System reset</td>
<td>0</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>Channel reset</td>
<td>0</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
### Table 3-2 (2/5) Register Status (No. 2)

(X: Undefined, -: Not affected)

<table>
<thead>
<tr>
<th>Register name</th>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CR6</strong></td>
<td>Function</td>
<td>SYNC Character/Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CR7</strong></td>
<td>Function</td>
<td>SYNC Character/Flag</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CR8</strong></td>
<td>Function</td>
<td>Tx Data Length L-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CR9</strong></td>
<td>System reset</td>
<td>1</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>1</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CR10</strong></td>
<td>Function</td>
<td>CRC Initialized Condition</td>
<td>Data Format</td>
<td>Auto Tx on Sync/ Tx on Loop</td>
<td>Idle Condition</td>
<td>Tx Condition on Underrun</td>
<td>D4/ Loop Enable</td>
<td>SYNC Character Bits</td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>CR11</strong></td>
<td>Function</td>
<td>Break/ Abort/ GAE IE</td>
<td>Tx Underrun/ EOM IE</td>
<td>CTS IE</td>
<td>Sync/ Hunt IE</td>
<td>DCD IE</td>
<td>All Sent IE</td>
<td>Idle Detect IE</td>
<td>BRG IE</td>
</tr>
<tr>
<td>System reset</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Channel reset</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>CR12</strong></td>
<td>Function</td>
<td>Select for TRxC</td>
<td>Select for DPLL</td>
<td>Tx BRG IE</td>
<td>Rx BRG IE</td>
<td>Tx BRG Register Set</td>
<td>Rx BRG Register Set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3-2 (3/5) Register Status (No. 3)

(X: Undefined, --: Not affected)

<table>
<thead>
<tr>
<th>Register name</th>
<th>Function</th>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rx BRG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Register-L</td>
<td>System reset</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rx BRG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Register-H</td>
<td>System reset</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tx BRG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Register-L</td>
<td>System reset</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tx BRG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Register-H</td>
<td>System reset</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>System reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>System reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CR15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>System reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3-2 (4/5) Register Status (No. 4)

(X: Undefined, -: Not affected)

<table>
<thead>
<tr>
<th>Register name</th>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SR0</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td>End</td>
<td>CRC/</td>
<td>Rx</td>
<td>Parity</td>
<td>Short</td>
<td>Tx</td>
<td>Sending</td>
<td>Rx</td>
<td></td>
</tr>
<tr>
<td></td>
<td>of</td>
<td>Framing</td>
<td>Overrun</td>
<td>Error</td>
<td>Frame</td>
<td>Buffer</td>
<td>Abort</td>
<td>Data</td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>SR1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td>Break/</td>
<td>Tx</td>
<td>CTS</td>
<td>Sync/</td>
<td>DCD</td>
<td>All</td>
<td>Idle</td>
<td>BRG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Abort/</td>
<td>Underrun/</td>
<td></td>
<td>Hunt</td>
<td></td>
<td>Sent</td>
<td></td>
<td>Zéro</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GA Detect</td>
<td>EOM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Count</td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>-</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>SR2B</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SR3</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>SR4A</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td><strong>SR10</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td>One</td>
<td>Two</td>
<td>Sending</td>
<td>Tx</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Clock</td>
<td>Clocks</td>
<td>on Loop</td>
<td>Sync/</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


Table 3-2 (5/5) Register Status (No. 5)

(X: Undefined, -: Not affected)

<table>
<thead>
<tr>
<th>Register name</th>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Function</strong></td>
<td></td>
<td>Break/ Abert IE</td>
<td>Tx Underun/ EOM IE</td>
<td>CTS IE</td>
<td>Sync/ Hunt IE</td>
<td>DCD IE</td>
<td>All Sent IE</td>
<td>Idle Detect IE</td>
<td>BRG IE</td>
</tr>
<tr>
<td><strong>SR11</strong></td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>System reset</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td></td>
<td>Rx BRG Counter Constant L-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SR12</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td></td>
<td>Rx BRG Counter Constant H-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SR13</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td></td>
<td>Tx BRG Counter Constant L-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SR14</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td></td>
<td>Tx BRG Counter Constant H-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SR15</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel reset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The AMPSC's internal registers can be accessed through pins B/\overline{A} and C/\overline{D}. Selection of registers is done by combining pointer bits D0-D2 of CR0 and High Pointer bit D3 of the same register. The register pointer within the AMPSC is automatically cleared when a register is accessed after being selected by the pointer bits. It indicates CR0/SR0 unless the pointer bit value is subsequently specified. The register addressing sequence is shown in Fig. 3-2.

(Example) When setting 20H into CR4A

Fig. 3-2 AMPSC Register Setting Sequence

Exception:
If D1 or D0 of CR12 is set to 1, the subsequent two control word writing operations access the lower then the upper bytes of the TxBRG or RxBRG register. The register pointer is cleared only after these write accesses are completed. This sequence is shown in Fig. 3-3. If both D1 and D0 are zero, this exception is not applied.
Example: Setting the TxBRG register on channel B:

![Diagram of B/A, C/D, WR, Databus, and Register pointer value](image)

Fig. 3-3 TxBRG Register Setting Sequence

3.2 Control Registers (CR)
This section describes control registers used to specify operation modes of the AMPSC. Control register functions are listed in Fig. 3-4 below. Some control registers need not be set depending on the operation modes to be used. Carefully examine the meaning of each bit of the control registers in the desired operation mode.
Fig. 3-4 Control Register Configuration
3.2.1 Control Register 0 (CR0)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
<th>Function</th>
<th>Command</th>
<th>Register Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>0 0</td>
<td>0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>No Operation</td>
<td>No Operation</td>
<td>CR0, SR0</td>
</tr>
<tr>
<td>0 1</td>
<td>0 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Initialize Rx CRC Calculator</td>
<td>High Pointer</td>
<td>CR1, SR1</td>
</tr>
<tr>
<td>1 0</td>
<td>0 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Initialize Tx CRC Calculator</td>
<td>Channel Reset</td>
<td>CR3, SR3</td>
</tr>
<tr>
<td>1 1</td>
<td>1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reset Tx Underrun/EOM Bit</td>
<td>Enable</td>
<td>CR5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Next Rx Character</td>
<td>CR4, SR4A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Interrupt</td>
<td>CR6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CR7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 1 0 Error Reset</td>
<td></td>
<td>CR8, SR8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 1 1 End of Interrupt</td>
<td></td>
<td>CR9, SR9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(Channel A Only)</td>
<td></td>
<td>CR10, SR10</td>
</tr>
</tbody>
</table>

(1) D7, D6 (CRC Control)
These bits are valid when the COP or BOP mode is selected.
They have no meaning in the asynchronous mode.
0 0: No operation.
0 1: Initialize Receive CRC Calculator
This command is used to initialize the Rx CRC calculator. It must be issued before starting data reception. The initial state (value) of the Rx CRC calculator differs depending on the value of D7 of CR10:
. When D7 of CR10 is 0, the initial value is 0.
. When D7 of CR10 is 1, the initial value is 1.
Bit D7 of CR10 must be set before issuing this command.
Exception:
In the BOP mode, the CRC calculator is automatically initialized according to the value of CR10's D7 upon receiving the flag, without the need for issuing this command.

1 0: Initialize Transmit CRC Calculator
This command is used to initialize the TxCRC calculator. It must be issued before starting data transmission. The initial value of the TxCRC calculator differs depending on the value of D7 of CR10:
• When D7 of CR10 is 0, initial value is 0.
• When D7 of CR10 is 1, initial value is 1.
D7 of CR10 must be set before issuing this command.

Exception:
If D7 of CR10 is set to 1 in the BOP mode, the TxCRC calculator is automatically initialized to 1 when a flag is transferred to the transmit shift register within the AMPSC.

1 1: Reset Transmit Underrun/End of Message Bit
This command is used to reset (1 to 0) D6 of SR1 (Tx Underrun/EOM). If Tx data is not written to the transmit buffer in the specified time period after transmit operation was started, the AMPSC enters the Tx Underrun/EOM state. At this point the AMPSC judges whether a CRC is to be sent or not depending on the value of D6 of SR1. Therefore, manipulation of bit D6 of SR1 is required before starting transmission:
• When D6 of SR1 is 0 when a Tx underrun occurs, a CRC is sent.
• When D6 of SR1 is 1 when a Tx underrun occurs, a SYNC character/flag is sent.
If CRC transmission is desired at the time of the Tx underrun, D6 of SR1 must have been set to zero by this command beforehand. This command becomes valid after at least one data byte is written into the AMPSC after transmission is enabled.
Exception:
In the ROP mode, D6 of SP1 is set to zero when the first
Tx data of a frame is written into the AMPSC without
issuing this command.

(2) D5-D3 (Command)
These bits of control register 0 are used to control the
device's status after the device has entered a given
state. The commands are valid only when they are issued.
0 0 0: No operation.
0 0 1: High Pointer
   This command is used with D2-D0 (Register Pointer)
of CR0 when accessing CR8-CR15 or SR8-SR15. For
example, when accessing CR11, D5-D0 of CP0 are set
to 00101.
0 1 0: Reset External/Status Bit Latch
   This command is issued when an E/S bit (each bit of
SRl) latch operation occurred. It allows latching
of a new cause of E/S bit status change. If E/S
interrupt is enabled, an E/S interrupt will occur
when the E/S bit status has changed.
0 1 1: Channel Reset
   This command is used to reset the AMPSC channels.
   It has a function equivalent to a system reset.
   Executing channel reset stops AMPSC operation. To
determine the CR and SR status after channel reset,
see Table 3-2. A control word should not be written
within 3 system clock periods (3 tcy) after the
Channel Reset command was issued.
1 0 0: Enable Next Receive Character Interrupt
   This command is valid only when the First Rx INT
mode (D4, D3 of CR1 = 0, 1) is selected. It is used
when it is desired to guarantee that a Rx interrupt
will occur when the next data character is received
after this command is issued.
Exception:
This command is invalid when the First Rx INT Mask is on (D5 of CR1 = 1), even if the First Rx INT mode is selected.

1 0 1: Reset Transmit Interrupt/DMA Pending
This command is used to clear the AMPSC's Tx interrupt request or Tx DMA request which occurs when the Tx buffer is emptied (D2 of SR0 = 1), without writing Tx data into the device. It is usually used to clear a Tx interrupt or Tx DMA request caused by the Tx Buffer Empty state which occurs after the last Tx data is written into the AMPSC.
This command causes the AMPSC to operate follows:
(a) Asynchronous mode
The device issues a Tx interrupt or Tx DMA request when the Tx buffer is emptied after the next Tx data is written.
(b) COP mode
The device issues a Tx interrupt or Tx DMA request when the internal Tx buffer is emptied after the device has completed CRC transmission in the Tx Underrun state which occurred after this command was issued. If Tx data was written in the Tx Underrun state (e.g., PAD), the device issues a Tx interrupt or Tx DMA request when its Tx buffer is emptied after that data is transmitted following a CRC.
(c) BOP mode
The device issues a Tx interrupt or Tx DMA request when its internal Tx buffer is emptied after a CRC has been transmitted in the Tx Underrun state which occurred after this command was issued.

1 1 0: Error Reset
This command is used to reset the pertinent bits (D3-D7 of SR0) if a Special Rx Condition occurred. If a Special Rx Condition interrupt occurs when the First Rx INT mode is selected, data subsequently received is not transferred to the last stage of the AMPSC's internal Rx buffer but remains in the first and second stages of the buffer.
111: End of Interrupt
This command is used to let the AMPSC recognize the end of an interrupt sequence. It should be issued when an interrupt service for the AMPSC is completed. Execution of this command resets the cause of the interrupt with the highest priority at that point, and enables subsequent interrupt requests.

(3) D2-D0 (Register Pointer)
These bits specify the number of the register within the AMPSC to be accessed. They are reset to 000 when a reset operation is executed or when the AMPSC is accessed after a Register Pointer value is specified. For registers numbered 8 and above, the High Pointer command (D5-D3 = 001) is added to this Register Pointer to access those registers.

3.2.2 Control Register 1 (CR1)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Operation protocol</td>
<td>Short Frame Detect</td>
<td>Overrun Error INT</td>
<td>First Rx INT Mask</td>
<td>Rx INT Mode</td>
<td>First Tx INT</td>
<td>DMA Enable</td>
<td>Tx INT/ DMA Enable</td>
<td>E/S INT Enable</td>
</tr>
<tr>
<td>Async</td>
<td></td>
<td>0 Disable Normal Mode</td>
<td>0 Disable 1 Special Mode</td>
<td>0 0 Rx INT Disable 0 1 First Rx INT 1 0 All Rx INT-1 1 1 All Rx INT-2</td>
<td>0 Disable 0 Disable 1 Enable 0 Disable 0 Disable 0 Disable 0 Disable 0 Disable 0 Disable 0 Disable 0 Disable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COP</td>
<td></td>
<td>0 Enable 1 Enable 1 Enable</td>
<td>0 Enable 1 Enable 1 Enable 1 Enable</td>
<td>0 Enable 1 Enable 1 Enable 1 Enable 1 Enable 1 Enable 1 Enable 1 Enable 1 Enable 1 Enable 1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td></td>
<td>0 Disable 1 Enable</td>
<td>0 Disable 1 Enable</td>
<td>0 Disable 1 Enable</td>
<td>0 Disable 1 Enable</td>
<td>0 Disable 1 Enable</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) D7 (Short Frame Detect; Valid in BOP Mode only)
This bit of Control Register 1 is used to detect short frames (frames less than 32 bits long).
0: Disable
   Short frame detection is disabled.
1: Enable
   Short frame detection is enabled. If a received
   frame is a short frame, D3 of SR0 (Short Frame
   Detect) is set to 1, causing a Special Rx Condition
   interrupt.

(2) D6 (Overrun Error INT)
This bit is used to specify how an overrun error will be
indicated by the Rx Overrun Error bit (D5 of SR0) and
when the Special Rx Condition interrupt occurs.

0: Normal Mode
In this mode, overrun error status is available only
if the received data which caused the overrun error
is transferred to the last stage of the receive
buffer along with the error flag value shifted in
SR0. A Special Rx Condition interrupt occurs at this
time. In this mode the received data which caused an
overrun error corresponds to the information pointed
by the Rx Overrun Error bit. This is illustrated in
Fig. 3-5(a).

1: Special Mode
In this mode overrun information is reflected by the
Rx Overrun Error bit when an overrun error occurred
within the AMPSC. A Special Rx Condition interrupt
also occurs at this time. Therefore, the received
data which caused the overrun error does not
match the information indicated by the Rx
Overrun Error bit. This is illustrated in Fig.
3-5(b).
(a) If $D_6$ of CR1 = 0 (Normal mode)

Parity Error

Rx Overrun Error

CRC/Framing Error

SR0: D6–D4

Data bus buffer side

Final stage

Rx buffer (FIFOx3)

Shift register side

Overrun signal

(b) If $D_6$ of CR1 = 1 (Special mode)

Data bus buffer side

Overrun signal

Shift register side

Fig. 3-5 Rx Overrun Error Bit Set Timing
(3) D5 (First Receive Interrupt Mask)
This is enabled only if the First Rx INT Mode (D4, D3 of CR1 = 0, 1) is selected. It is used to mask Rx interrupt caused by first received data.

0: Disable
The first received data causes an Rx interrupt (normal operation).
1: Enable
The first received data does not cause an Rx interrupt, and the Enable Next Rx Character INT command becomes invalid. However this bit does not affect the Special Rx Condition interrupt.

(4) D4, D3 (Receive Interrupt Mode)
These bits are used to set the Rx Interrupt mode. They specify in what way received data is to be indicated.

0 0: Receive Interrupt Disable
This mode is used to accept received data by using status polling, or to disable receive interrupt requests.

0 1: First Receive Interrupt
This mode is used to accept received data by using DMA. In this mode, an Rx interrupt occurs only by the first data is received after the Rx was enabled after initialization. In this case the first received data refers to:

- Asynchronous mode --- First character received.
- COP mode --- First character, other than SYNC character, received after synchronization was established.
- BOP mode --- First data (generally address data) first received after flag detection. In the Address Search mode it refers to valid address data that follows a flag and matches the value of CR6. These received data items can be read as normal data.

If an Enable Next Rx Character INT command is issued, data received after this command was issued can cause an Rx interrupt.
1 0: All Receive Interrupt-1
This mode is used to indicate received data by using interrupts. An interrupt occurs each time data is loaded into the receive buffer. In this mode a parity error is one of the sources of a Special Rx Condition interrupt.

1 1: All Receive Interrupt-2
Same as the All Rx Interrupt-1, with the exception that a parity error is not indicated by a Special Rx Condition interrupt.

(5) D2 (First Transmit Interrupt/DMA Enable)
This bit determines whether the Tx INT/DMA request is to be activated or not immediately after transmission is enabled. It is valid when Tx INT/DMA mode is enabled (D7 of CR1 = 1).

0: Disable
In this mode no Tx INT/DMA request occurs when transmission is enabled. The first Tx data cannot be written by using a Tx interrupt or DMA request in this mode. Therefore, the first Tx data must be directly written by the host processor immediately after transmission is enabled. Once the first Tx data is written, the Tx INT/DMA request becomes active when the Tx buffer is subsequently emptied.

1: Enable
When transmission is enabled in the AMPSC, the Tx INT/DMA request is activated. In this mode, all Tx data including the first Tx data can be transferred by using an interrupt or DMA request.

(6) D1 (Transmit Interrupt/DMA Enable)
This bit is used to transfer Tx data by using an interrupt or DMA request. It is related to the First Tx INT/DMA Enable bit (D2 of CR1), and affects occurrence of the Tx interrupt and Tx DMA request.

0: Disable
The Tx INT/DMA Request remains inactive if the Tx buffer is emptied. This mode is used to transfer Tx data by using status polling.
1: Enable
The Tx INT/DMA Request becomes activated when the Tx buffer is emptied. The activation timing differs depending on the value of the First Tx INT/DMA Enable bit. This mode is used to transfer Tx data by using an interrupt or DMA request.

(7) D0 (External/Status Interrupt Enable)
This bit determines whether or not an interrupt is to be requested when the E/S bit status has changed. It is related to the settings of the bits of CR11. For more information about the E/S interrupt, refer to subsections 3.2.13, Control Register 11 (CR11), 3.2.14, Control Register 12 (CR12), and 3.3.2, Status Register 1 (SR1).

0: Disable
In this case, no E/S interrupt occurs even if the E/S bit status has changed, regardless of the value of CR11. While no E/S bit latch operation occurs, the E/S bit of SR1 indicates the E/S cause status at the point of status change.

1: Enable
In this case, interrupts caused by each E/S change are enabled according to the bit status on CR11. Changes of the E/S bit are latched.

3.2.3 Control Register 2A (CR2A)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
<th>Vector Mode</th>
<th>Status Affects Vector</th>
<th>Output Vector Type</th>
<th>Priority Select</th>
<th>INT/DMA Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Non-Vectored</td>
<td>0</td>
<td>Fixed</td>
<td>0 0 0 Type A-1</td>
<td>0</td>
<td>RxB</td>
<td>0 Both Channel</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Vectored</td>
<td>1</td>
<td>Vector Status</td>
<td>0 0 1 Type A-2</td>
<td>1</td>
<td>TxA</td>
<td>0 1 Ch-A:DMA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Affects Vector</td>
<td>0 1 0 Type A-3(Slave)</td>
<td></td>
<td>1 RxB</td>
<td>Ch-B:INT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 1 1 Type B-1</td>
<td></td>
<td></td>
<td>1 0 Both Channel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DMA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 1 Use Prohibited</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2-39
(1) D7 (Vector Mode)
This bit determines the treatment of interrupt vectors. The value of this bit affects hardware configuration.

0: Non-Vectored
   Interrupt vectors are obtained by reading the value of SR2B. In this mode the INTAK signal is not used.
1: Vectored
   Interrupt vectors are obtained by using the INTAK signal.

(2) D6 (Status Affects Vector)
This bit determines whether the value of an interrupt vector is to be changed or not depending on the cause of interrupt.

0: Fixed Vector
   The value of the interrupt vector is fixed. The vector value set in CR2B is directly output.
1: Status Affects Vector
   Three bits of a vector value change depending on the cause of the interrupt. The bits that change differ depending on the value of the Output Vector Type bits (D5-D3 of CR2A). The values of the remaining bits set in CR2B are directly output.

(3) D5-D3 (Output Vector Type)
These bits are used to determine interrupt vector output operation and the bits of an interrupt vector which are to be changed if the Status Affects Vector is set by D6 of this register. The following shows the vector bits that are affected:

0 0 0: Type A-1 --- Bits $V_4$, $V_3$, and $V_2$ change.
0 0 1: Type A-2 --- Bits $V_4$, $V_3$, and $V_2$ change.
0 1 0: Type A-3 --- Bits $V_4$, $V_3$, and $V_2$ change.
0 1 1: Type B-1 --- Bits $V_4$, $V_3$, and $V_2$ change.
1 0 0: Type B-2 --- Bits $V_2$, $V_1$, and $V_0$ change.

(4) D2 (Priority Select)
This bit is used to determine the order of priority between Tx request on channel A and Rx request on channel B. The priority level for other requests are fixed. The priority is not applied to DMA transfer.
0: RxA > TxA > RxB > TxB > E/S A > E/S B
1: RxA > RxB > RxA > TxB > E/S A > E/S B

High ← Low

(5) D1–D0 (Interrupt/DMA Mode)
These bits are used to select the Tx/Rx data transfer mode on each channel. While E/S, Rx, and Special Rx Condition interrupts are enabled on the channel for which DMA mode is selected, the Tx interrupt is disabled on that channel.

0 0: Both Channel Interrupt
This mode allows transmit/receive data to be transferred by using interrupts on both channels A and B. An interrupt occurs when the Tx buffer is emptied or the Rx Data Available state is entered.

0 1: Channel A DMA, Channel B Interrupt
This mode allows transfer using DMA for channel A and interrupts for channel B.

1 0: Both Channels DMA
This mode allows transmit/receive data to be transferred by using DMA on both channels.

Table 3-3 shows the values of D2, D1, and D0 of CR2A and corresponding Tx/Rx data transfer modes, pin functions, and priority level.
Table 3.3 CR2A Register Values Versus Operation Modes

<table>
<thead>
<tr>
<th>CR2A</th>
<th>Transfer mode of transmit/receive data</th>
<th>Pin function</th>
<th>Priority</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Channel A</td>
<td>Channel B</td>
<td>DTRA/DRQTxB</td>
<td>DTRB/DRQRxB</td>
</tr>
<tr>
<td>0 0 0</td>
<td>INT</td>
<td>INT</td>
<td>DTRA</td>
<td>DTRB</td>
</tr>
<tr>
<td>1 0 0</td>
<td>INT</td>
<td>INT</td>
<td>DTRA</td>
<td>DTRB</td>
</tr>
<tr>
<td>X 0 1</td>
<td>DMA</td>
<td>INT</td>
<td>DTRA</td>
<td>DTRB</td>
</tr>
<tr>
<td>X 1 0</td>
<td>DMA</td>
<td>DMA</td>
<td>DRQTxB</td>
<td>DRQRxB</td>
</tr>
</tbody>
</table>

*: Rx and E/S interrupts will occur on the channel for which the DMA mode is selected, provided interrupts are enabled. Note X: Don't care.

3.2.4 Control Register 2B (CR2B)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>DO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Initial Vector Value</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contents</td>
<td>V7</td>
<td>V6</td>
<td>V5</td>
<td>V4</td>
<td>V3</td>
<td>V2</td>
<td>V1</td>
<td>V0</td>
</tr>
</tbody>
</table>

(1) D7-D0 (Initial Vector Value)
These bits are used to set the initial value of an interrupt vector.
### 3.2.5 Control Register 3 (CR3)

<table>
<thead>
<tr>
<th>Bit Function</th>
<th>D7-DO</th>
<th>Operation protocol</th>
<th>Rx Character Bit Length</th>
<th>Auto Enter</th>
<th>Rx CRC Enable</th>
<th>Address Search Mode</th>
<th>Rx Character Load Inhibit/Multicast</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Asyc</td>
<td>0 0 5 bits/char.</td>
<td>0 Disable</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0 Disable</td>
</tr>
<tr>
<td></td>
<td>0 1 7 bits/char.</td>
<td>1 Enable</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 Enable</td>
</tr>
<tr>
<td></td>
<td>1 0 6 bits/char.</td>
<td>0 No Operation</td>
<td>1 Enter</td>
<td>0</td>
<td>0</td>
<td>&quot;SYNC Char. Load Inhibit&quot;</td>
<td>0 No Operation</td>
</tr>
<tr>
<td></td>
<td>1 1 8 bits/char.</td>
<td>0 No Operation</td>
<td>1 Enter</td>
<td>0</td>
<td>0</td>
<td>&quot;Multicast&quot;</td>
<td>0 No Operation</td>
</tr>
<tr>
<td></td>
<td>BOP</td>
<td>0 No Operation</td>
<td>0 Disable</td>
<td>0</td>
<td>1 Enable</td>
<td>0 No Operation</td>
<td>1 Enable</td>
</tr>
</tbody>
</table>

1. **D7-D0 (Receive Character Bit Length)**
   These bits determine the number of bits per character when the received serial data is converted into parallel data. The number of bits can be changed while receiving serial data, however, the number should be changed before input of a character which has the new number of bits per character.
   - 0 0: 5 bits/character
   - 0 1: 7 bits/character
   - 1 0: 6 bits/character
   - 1 1: 8 bits/character

2. **D5 (Auto Enable)**
   This bit is used when performing Tx/Rx control using the CTS/DCD pin and when displaying the Tx state using the RTS pin.
   - 0: Disable
   - CTS and DCD pins become input pins. If the state of these pins changes, it will be latched by the corresponding E/S bit (D5 and D3 of SR1) and an E/S interrupt will occur.
In this case, the RTS pin becomes an output pin and its output is controlled by the RTS Control bit (D1 of CR5).

1: Enable
The CTS/DCD pin can be used for Tx/Rx control. For control methods, refer to Tables 1-3 and 1-4 of Chapter1, Pin Functions. Also, in this case, a change of the pin state is latched by the corresponding E/S bit and an E/S interrupt occurs.
The function of RTS pin changes depending on the operation protocol and state of RTS bit (refer to Table 1-5).

(3) D4 (Enter Hunt Phase; Valid in COP or BOP mode)
This bit directs the AMPSC to enter the Hunt Phase.
0: No operation
1: Enter Hunt Phase
   The AMPSC enters the Hunt Phase and starts searching of the SYNC character (COP mode) or flag (BOP mode).
   When this bit is set, its function is retained until synchronization is established. After synchronization is established, this bit is automatically reset to 0 and the detection process stops.

(4) D3 (Receive CRC Enable: valid in COP or BOP mode)
This bit determines whether or not CRC calculation is to be done on received data.
0: Disable
   CRC calculation is not done on received data.
1: Enable
   CRC calculation is done on received data.
(a) COP mode
Since, in this mode, CRC calculation is started after 8-bit times have elapsed from when a character is received, the following control is required:
For example, assume that a character string, \( C_0, C_1, D_2, \ldots \) is received in this order (see Fig. 3.6). \( C_0 \) is first received and loaded into the Rx buffer, followed by \( C_1 \) which is shifted into the shift register in series. Whether or not CRC
calculation is to be done on \( C_0 \) must be decided before
the assembling for \( C \) is completed (before \( C_1 \) is
transferred to the Rx Buffer). If the RxCRC Enable bit
is set to one when \( C_1 \) is transferred to the Rx buffer,
CRC calculation is done on \( C_0 \). If it is zero, CRC
calculation is not done on \( C_0 \). This can be done by
reading \( C_0 \) from the Rx buffer, checking the character
value by the host processor, and then setting the value
of this bit (RxCRC Enable bit). Since this control must
be completed before all bits of \( C \) reach the AMPSC, a
sufficient time interval must be reserved for this
control after the Rx Data Available state (DO of SR0 =
1) is activated by \( C \). Due to this control, received
character buffering (for 3 bytes) using the Rx buffer
is, as a rule, not available in the COP mode.

\[ \text{Fig.3.6 Character Reception} \]

(b) BOP mode
Since, in this mode, all data within a frame are
included in the CRC calculation, the RxCRC Enable
bit should be kept at 1 during data reception.

(5) D2 (Address Search Mode; valid in BOP mode)
This bit determines whether or not the address field
value of a received frame is to be compared with the
value set in CR6.
0: Disable
Address field detection is not done. All frames are
received unconditionally.
1: Enable
Address field detection is done. Operation in Auto Enable mode differs depending on the value of the Multicast bit (D1 of CR3).

(a) When D1 of CR3 = 0
If the address field value of a frame is equal to the value set in CR6 or equal to \textit{\texttt{1111111}} (global address), the frame is received. In any other case, the device doesn't receive the frame and automatically returns to the Flag Detect state.

(b) When D1 of CR3 = 1
The higher 4 bits (D7-D4) of the address field are compared with bits D7-D4 of CR6. The operation in this mode is identical to that in the above case (D1 of CR3 = 0), with the exception that the bits to be compared are different.

In this mode, abort detection is not done unless the address that matches the value set in CR6 is detected. Once the address matches, abort detection is subsequently done unconditionally.

(6) D1 (Sync Character Load Inhibit/Multicast; valid in COP or BOP mode)
The meaning of this bit differs in the COP and BOP modes.

(a) COP mode
This bit serves a Sync Character Load Inhibit function. It determines the treatment of a character (within a received data block) whose data pattern is identical to that of the SYNC character (same as the value of CR6).

0: No Operation
If a character with a data pattern identical to the SYNC character is detected in a received data block, no special processing is done on that character.
1: SYNC Character Load Inhibit
Inhibits the SYNC character (for character synchronization) to be transferred to the Rx buffer as one of the received data items. This bit is used with the Enter Hunt Phase bit (D4 of CR3) before the AMPSC starts a receive operation. The inhibit function remains valid after synchronization is established, so that the character pattern identical to the SYNC character pattern is not transferred to the Rx buffer and is not used for CRC calculation.

(b) BOP mode
This bit is used to enable/disable the Multicast function. It becomes valid when the Address Search mode is selected (D2 of CR3 = 1).
0: No Operation
   The Multicast function remains ineffective.
1: Multicast
   The Multicast function becomes effective. For details of Multicast operations, see the description for the Address Search Mode bit (D2 of CR3).

(7) D0 (Receive Enable)
This bit control receiver operations.
0: Disable
   Disables receiver function.
1: Enable
   Enables the receiver function to start a receive operation. When the Auto Enable mode is selected (D5 of CR3 = 1), the signal applied to the DCD pin also affects the start of a receive operation.
### 3.2.6 Control Register 4 (CR4)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Clock Rate</td>
<td>Protocol Mode</td>
<td>Tx stop Bits</td>
<td>Parity</td>
<td>Parity</td>
<td>Select</td>
<td>Enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td>0 0 X1</td>
<td>Don't Care</td>
<td>0 0 Use Prohibited</td>
<td>0 Odd</td>
<td>0 Disable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 1 X16</td>
<td></td>
<td>0 1 1</td>
<td>1 Even</td>
<td>1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 0 X32</td>
<td></td>
<td>1 0 1.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 1 X64</td>
<td></td>
<td>1 1 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mono- Sync</td>
<td>Don't Care</td>
<td>0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bi-sync</td>
<td></td>
<td>0 1</td>
<td>0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>External Sync</td>
<td>0 0 or 0 1</td>
<td>1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td>Don't Care</td>
<td>1 0</td>
<td>0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) D7, D6 (Clock Rate), D5, D4 (Protocol Mode), D3, D2 (Stop Bits)
Combinations of these six bits are used to specify the operation protocol of the device.

(a) Asynchronous mode
In this mode, set bits D3 and D2 to 0, 1; 1, 0; or 1, 1 (which specify transmit stop bit lengths of 1, 1.5, and 2 bits, respectively). Bits D7 and D6 are used to select the clock rate as a multiple of baud rate.
0 0: x1
0 1: x16
1 0: x32
1 1: x64
When the Asynchronous mode is selected, bits D5 and D4 remain ineffective.

(b) COP mode
Set bits D3 and D2 to 0, 0. Any of the following three modes can be selected with bits D7-D4:
(i) Mono-sync mode
Bits D7 and D6 can be set to any value. Set D5 and D4 to 0, 0.
The SYNC character bit length is selectable from 6 and 8 bits (with D0 of CR10).
(ii) Bi-sync mode
Bits D7 and D6 be set to any value. Set D5 and D4 to 0, 1.
The SYNC character bit length is selectable as 12 or 16 bits (with D0 of CR10).

(iii) External Sync mode
Set bits D7 and D6 to 0, 0 or 0, 1 and D5 and D4 to 1, 1. This mode is used to establish character synchronization using an external synchronization signal. A Low level input to the SYNC pin is used to establish synchronization. For the timing, see the description of the SYNC pin.
This mode is selectable only when the Xtal Select bit (D7 of CR15) is set to zero.

(c) BOP mode
Bits D7 and D6 can be set to any value. Set D5 and D4 to 1, 0 and D3 and D2 to 0, 0. This setting selects the ordinary HDLC/SDLC operation. If the Loop Select bit (D1 of CR10) is set to 1, the Loop mode is selected.

(2) D1 (Parity Select; valid in Async and COP modes)
This bit selects parity type. It is valid only when the Parity Enable bit (D0 of CR4) is set to 1.
0: Odd parity
1: Even parity

(3) D0 (Parity Enable)
This bit enables the device to add a parity bit to Tx data and do a parity check on Rx data.
0: Disable
   Disable the parity function.
1: Enable
   Enable the parity function. The type of parity is selected with the Parity Select bit (D1 of CR4). If the received data length is 7 bits or less (excluding the parity bit), the parity bit can be read as part of the received data.
### 3.2.7 Control Register 5 (CR5)

<table>
<thead>
<tr>
<th>Bit Function</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation protocol</td>
<td>DTR Control</td>
<td>Tx Character Bit Length</td>
<td>Send Break/Abort</td>
<td>Tx CRC</td>
<td>RTS Control</td>
<td>CRC Polynomial</td>
<td>Tx CRC Enable</td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td>0 DTR= &quot;H&quot;</td>
<td>0 0 5 or less bits/char.</td>
<td>&quot;Send Break&quot;</td>
<td>0 Disable</td>
<td>0 RTS= &quot;H&quot;</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 DTR= &quot;L&quot;</td>
<td>0 1 7 bits/char.</td>
<td>0 No Operation</td>
<td>1 Enable</td>
<td>(at All Sent)</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 0 6 bits/char.</td>
<td>1 Send Break</td>
<td>0</td>
<td>1 RTS= &quot;L&quot;</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 1 8 bits/char.</td>
<td>&quot;Send Abort&quot;</td>
<td>0</td>
<td>1 RTS= &quot;L&quot;</td>
<td>0 Disable</td>
<td>1 Enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>COP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. **D7 (DTR Control)**
   This bit controls the DTR pin status.
   - 0: DTR pin = "H"
   - 1: DTR pin = "L"

2. **D6–D5 (Transmit Character Bit Length)**
   These specify the bit count per character in serial Tx data, as follows:
   - 0 0: 5 or less bits/character
   - 0 1: 7 bits/character
   - 1 0: 6 bits/character
   - 1 1: 8 bits/character
   If the bit count per character is 6 or 7 bits, down to the lower 6 or 7 bits of written Tx data become valid, respectively. Any value may be written for the higher 2 or 1 bits. If the bit count per character is 5 bits or less, the parallel data format as shown in Table 3-4 must be used when writing.
Table 3-4 Parallel Data Format for 5 Bits or Less Per Character

<table>
<thead>
<tr>
<th>No.of bits</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>D1</td>
<td>D0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>D2</td>
<td>D1</td>
<td>D0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>D3</td>
<td>D2</td>
<td>D1</td>
<td>D0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>D4</td>
<td>D3</td>
<td>D2</td>
<td>D1</td>
<td>D0</td>
</tr>
</tbody>
</table>

Dn: Effective data bit

(3) D4 (Send Break/Abort)
This bit controls break or abort transmission. The function of this bit depends on the selected operation mode.

(a) Asynchronous or COP mode
This bit provides the Send Break function.
0: No Operation
   If this bit is set to 1 and then reset to zero, the TxD pin is placed in the mark state ("H").
   If the bit remains zero, it causes no operation.
1: Send Break
   The TxD pin is placed in Send Break state (continuous "L" transmission), in which the Tx data in the Tx buffer becomes invalid. This bit remains valid even if transmission is disabled.
   The time interval in which this bit is kept at 1 must be longer than 1-character period.
   If the Tx on Loop is specified (D4 and D1 of CR10 are 1) in the COP mode (D5, D4 of CR4 = 0, 0 or 0, 1 and D3, D2 = 0, 0), this bit, if set to 1, is automatically reset to zero immediately when synchronization is established between the receiver and transmitter (for details, refer to the description of the CR10).

(b) BOP mode
This bit enables/disables the Send Abort function.
0: No Operation
   This bit causes no operation. If it is set to 1, it need no be reset to zero.
1: Send Abort
Starts a send abort sequence (8 consecutive 1s). Up to 13 ones may be sent depending on the preceding serial data pattern. The send abort operation invalidates the Tx data in the Tx buffer. The send abort sequence is followed by flag transmission. This bit is automatically reset to zero when the send abort sequence is completed.

(4) D3 (Transmit Enable)
This bit controls transmitter operations.
0: Disable
Disables transmitter function. If this bit is temporarily set to 1 to start transmission and then reset to zero, the transmitter enters the Mark state after completing the current data transmission. If the bit is reset during CRC character transmission, a SYNC character or flag is sent instead of the CRC character.
If this bit is reset in COP or BOP mode, the Tx Underrun/EOM bit (D6 of SR1) is set.
While the TxD pin is normally placed in the Mark state when the transmitter is disabled, it functions differently during the Echo Loop Test (D3 of CR14 = 1) or BOP Loop mode (D5, D4 of CR4 = 1, 0 and D4, D1 of CR10 = 1,1).
1: Enable
Enables the transmitter function to start transmission. If the Auto Enable mode is selected (D5 of CR3 = 1), the signal applied to the CTS pin affects the transmit operation.

(5) D2 (CRC Polynomial; valid in COP or BOP mode)
This bit selects the polynomial used for CRC calculation. The initial state of the Tx/Rx CRC calculation differs depending on the value of bit D7 of CR10.
(a) COP mode
0: CRC-CCITT
   In this case the polynomial expression is \( X^{16} + X^{12} + X^5 + 1 \).
1: CRC-16
   In this case the polynomial expression is \( X^{16} + X^{12} + X^5 + 1 \).

(b) BOP mode
0: CRC-CCITT
   Only the generation polynomial, \( X^{16} + X^{12} + X^5 + 1 \), is selectable.

(6) D1 (RTS Control)
This bit controls the RTS pin. The function of this bit in Auto Enable mode of Async (D5 of CR3=1) differs from that of the other modes as shown in Table 3-5.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Auto Enable bit</th>
<th>RTS Cont bit</th>
<th>RTS pin status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Async</td>
<td>0</td>
<td>0</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>When 0 from the beginning</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>When set to 1 and then to 0</td>
<td>Remains &quot;L&quot; while All Sent* = 0 and becomes, 1 when changed to &quot;H&quot;</td>
</tr>
<tr>
<td>COP/BOP</td>
<td>X</td>
<td>1</td>
<td>L</td>
</tr>
</tbody>
</table>

Table 3-5 RTS Control Bit and RTS Pin Status

X: Don't Care
*: SR1:D2

(7) D0 (Transmit CRC Enable: valid in COP or BOP mode)
This bit determines whether or not the CRC calculation is to be done on Tx data.
0: Disable
   CRC calculation is not done on Tx data.
1: Enable
   CRC calculation is done on Tx data.
(a) COP mode
When a character written into the Tx buffer is transferred to the Tx shift register, the value of this bit is used to determine whether or not CRC calculation is to be done on that character. This bit must be manipulated before the Tx character is written into the AMPSC.

(b) BOP mode
Since, in this mode, CRC calculation is done on all Tx data, this bit should normally be set to 1 (Tx CRC Enable).
The CRC character is sent if a Tx Underrun/EOM state occurs when TxCRC is enabled, provided the Tx Underrun/EOM bit (D6 of SRI) had been set to zero before. If the Tx Underrun/EOM bit is set to 1 before the Tx Underrun/EOM state occurs, the SYNC character (COP mode) or a flag (BOP mode) will be sent instead of the CRC character.

3.2.8 Control Register 6 (CR6)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation protocol</td>
<td>Function</td>
<td>SYNC Character/Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td>Mono-sync (Tx SYNC)</td>
<td>6</td>
<td>SYNC1</td>
<td>SYNC0</td>
<td>SYNC5</td>
<td>SYNC4</td>
<td>SYNC3</td>
<td>SYNC2</td>
</tr>
<tr>
<td></td>
<td>Bi-sync</td>
<td>12</td>
<td>SYNC3</td>
<td>SYNC2</td>
<td>SYNC1</td>
<td>SYNC0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ext. Sync</td>
<td>6</td>
<td>SYNC7</td>
<td>SYNC6</td>
<td>SYNC5</td>
<td>SYNC4</td>
<td>SYNC3</td>
<td>SYNC2</td>
</tr>
<tr>
<td>COP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ext. Sync</td>
<td>Address Search (Normal)</td>
<td>ADRS7-ADRS0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td>Address Search (Multicast)</td>
<td>ADRS7-ADRS4</td>
<td>Don't Care</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) D7-D0 (SYNC Character/Address; valid in COP or BOP mode)
These bits specify the SYNC character pattern or address value. The value of these bits differs depending on the selected operation mode.
(a) COP mode
In the Mono-sync or External Sync mode, the transmit
SYNC character pattern is set in these bits. In the
Bi-sync mode, a part of the SYNC character pattern
is set in them.

(b) BOP mode
The value set in these bits differs depending on the
Address Search Mode bit (D2 of CR3) and Multicast bit
(D1 of CR3) as shown in Table 3-6.

<table>
<thead>
<tr>
<th>Address Search Mode bit</th>
<th>Multicast bit</th>
<th>CR6 setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 or 1</td>
<td>Invalid</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Address value (ADRS7-ADRS0)</td>
</tr>
</tbody>
</table>
|                         | 1             | Higher 4 bits of address value (ADRS7-ADRS4,
|                         |               | lower 4 bits settings are invalid) |

3.2.9 Control Register 7 (CR7)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Function</td>
<td>SYNC Character/Flag</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Operation protocol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Async</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>COP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mono-sync (Rx SYNC)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>SYNC5 SYNC4 SYNC3 SYNC2 SYNC1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>SYNC7 SYNC6 SYNC5 SYNC4 SYNC3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>SYNC11 SYNC10 SYNC9 SYNC8 SYNC7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>SYNC15 SYNC14 SYNC13 SYNC12 SYNC11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ext. Sync</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BOP</td>
<td>Flag(01111110)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) D7-D0 (SYNC Character/Flag; valid in COP or BOP mode)
These bits specify the SYNC character pattern or flag.
The value to be set in these bits differs depending on
the selected operation mode.
(a) COP mode
In the Mono-sync mode, the receive SYNC character pattern is set in these bits. In the Bi-sync mode, a part of the SYNC character is set in them. These bits are not used in the External Sync mode.

(b) BOP mode
The flag pattern (01111110) is set in these bits.

### 3.2.10 Control Register 8 (CR8)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation protocol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. D7-D0 (Transmit Data Length L-Byte; valid in BOP mode)
   These bits specify the lower byte (bits 7-0) of the Tx data count. This register is used with CR9. It must be set before the Tx Data Length Counter Enable bit (D1 of CR13) and Tx Enable bit (D3 of CR5) are set.

### 3.2.11 Control Register (CR9)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation protocol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. D7-D0 (Transmit Data Length H-Byte; valid in BOP mode)
   These bits specify the higher byte (bits 15-8) of the Tx data count. This register is paired with CR8. Other than the setting of the higher byte, the rules are the same as those for CR8.
### 3.2.12 Control Register 10 (CR10)

<table>
<thead>
<tr>
<th>Bit Function</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation protocol</td>
<td>CRC Initialized Condition</td>
<td>Data Format</td>
<td>Auto Tx on Sync/ Tx on Loop</td>
<td>Idle Condition</td>
<td>Tx Condition on underrun</td>
<td>D4/Loop Enable</td>
<td>SYNC Character Bits</td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>COP</td>
<td>0</td>
<td>All &quot;0&quot;</td>
<td>0</td>
<td>1</td>
<td>NRZ1</td>
<td>0</td>
<td>0</td>
<td>&quot;D4 Enable&quot;</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>All &quot;1&quot;</td>
<td>1</td>
<td>0</td>
<td>FM1</td>
<td>0 Disable</td>
<td>1 Enable</td>
<td>0 Disable</td>
</tr>
<tr>
<td>BOP</td>
<td>0</td>
<td>&quot;Tx on Loop&quot;</td>
<td>0 Flag</td>
<td>1 Mark</td>
<td>0 Normal</td>
<td>1 Abort</td>
<td>&quot;Loop Enable&quot;</td>
<td>0 Disable</td>
</tr>
</tbody>
</table>

1. **D7 (CRC Initialized Condition; valid in COP or BOP mode)**  
   This bit specifies the initial state of the CRC calculation circuit. The CRC calculation circuit is initialized to the state specified by this bit when the Initialize Tx/Rx CRC Calculator command (D7, D6 of CR0 = 1, 0/0, 1) is issued.  
   - 0: All zeros  
     The CRC calculation circuit is initialized into all zeros by the Initialize command.  
   - 1: All ones  
     The CRC calculation circuit is initialized into all 1s by the Initialize command.  

2. **D6-D5 (Data Format)**  
   These bits specify the serial data format. Setting these bits enables the corresponding encoder/decoder.  
   - 0 0: NRZ  
   - 0 1: NRZI  
   - 1 0: FM1  
   - 1 1: FM0  
   If these bits are set to 00 (NRZ), it is possible to decode Manchester encoded data by setting the DPLL mode to FM (D7, D6, D5 of CR14 = 1, 1, 0).
Fig. 3-7 shows the data formats and Table 3-7 shows the operation for each data format.

![Diagram of data formats]

**Fig. 3-7 Data Formats**

<table>
<thead>
<tr>
<th>Format</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>NRZI</td>
<td>All data bits which are zero are inverted into 1s, all data bits which are 1 are not affected.</td>
</tr>
<tr>
<td>FM0</td>
<td>Data value is always inverted at the beginning of a bit cell. If a data bit is zero, it is again inverted at the center of the bit cell; if the data bit is 1, it keeps the same status.</td>
</tr>
<tr>
<td>FM1</td>
<td>Data value is always inverted at the beginning of a bit cell. If a data bit is 1, it is again inverted at the center of the bit cell; if the data bit is zero, it keeps the same status.</td>
</tr>
</tbody>
</table>
Manchester

The first half of a bit cell takes on the same value as a data bit, and is inverted at the center of the bit cell. Therefore, the second half of the bit cell has a value which is an inverse of the data bit. This means that if the data bit is zero, the center of the bit cell rises, while if the data bit is one, the center of the bit cell falls.

(3) D4 (Auto Tx on Sync/Tx on Loop; valid in COP or BOP mode)

The function of this bit differs depending on the operation mode. In the COP mode, it is used to synchronize the receiver with transmitter. In the BOP mode, it is used for SDLC loop operation control. This bit is valid only when the D4/Loop Enable state (D1 of CR10 = 1) is selected.

(a) COP mode

This bit provides the Auto Tx on Sync function to synchronize a receiver operation with a transmitter operation. It must be set before transmitter and receiver are enabled.

0: Disable

Disables the Auto Tx on Sync function (D1 of CR10 = 1).

1: Enable

If this bit is set to 1 along with the D4 Enable bit, the transmitter is disabled, and the receiver enters the Hunt Phase. The receiver subsequently starts searching for the SYNC character set in CR6. When the SYNC character is detected, character synchronization is established, and the transmitter is enabled. Data transmission is thus enabled. Establishment of character synchronization can be acknowledged from the Tx Sync/on Loop bit (D1 of SR10) which is set.

Once synchronization is established after this bit is set to 1, resetting it to zero does not affect synchronization.

(b) BOP mode

This bit enables/disables the Tx on Loop secondary station function. It is used for data transmission during the SDLC loop operation.
0: Disable
Once the AMPSC forms a loop and starts transmission, this bit must be reset to zero. This allows the CRC and flag to be automatically transmitted if a Tx Underrun/EOM state occurred, and the AMPSC to be subsequently placed again in the Loop mode with 1-bit delay. This bit must be reset before CRC transmission is completed after the first Tx data is written into the Tx buffer.

1: Enable
Setting this bit to 1 when the Loop Enable bit (D1 of CR10) is 1 selects the SDLC Loop Operation mode, in which the RxD input is connected to the TxD output within the AMPSC to form a loop. At this point the data input to RxD is transferred directly to TxD without a 1-bit delay and GA (Go-Ahead) pattern (01111111) detection is started. If the GA pattern is detected, a 1-bit delay is inserted, and GA pattern search is continued. While, at this point, the transmitter is still disabled and performs no transmit operation, the receiver can be controlled at the user's option. If the GA pattern is subsequently detected again, the transmitter within the AMPSC is enabled. At this point the GA pattern is automatically transformed into a flag, so that the data in the Tx buffer, if any, may be transmitted following this flag. Once transmission is started, this bit must be reset (see the description regarding Disable).

(4) D3 (Idle Condition; valid in BOP mode)
This bit determines the type of information to be transmitted following two Ending flags or an Abort.

0: Flag
Flags are transmitted.

1: Mark
A mark (consecutive 1s) is transmitted.
(5) D2 (Transmit Condition on Underrun; valid only in BOP mode)
This bit determines the type of information to be transmitted in the event of the Tx Underrun/EOM state.
0: Normal
   If the Tx Underrun/EOM bit (D6 of SRI) is zero, the CRC and flags are consecutively transmitted. If the Tx Underrun/EOM bit is 1 or TxCRC is disabled (D0 of CR5 = 0), only the flags are transmitted.
1: Abort
   Abort is transmitted regardless of the Tx Underrun/EOM bit status. The Abort transmission is followed by a flag transmission.

(6) D1 (D4/Loop Enable; valid in COP or BOP mode)
This bit determines whether the Auto Tx on Sync/Tx on Loop bit (D4 of CR10) is to be validated or not. The function of bit D4 is controlled by the status of this bit. The function of this bit differs depending on the operation mode. It must be set before transmitter and receiver are enabled.
(a) COP mode
   Serves as the D4 Enable function.
   0: Disable
      Disables the Auto Tx on Sync function. Once synchronization is established, this bit does not affect the synchronization if reset to zero.
   1: Enable
      Enables the Auto Tx on Sync function. For the operation in this state, see the description of the Auto Tx on Sync bit.
(b) BOP mode
   Serves as the Loop Enable function. This bit controls the SDLC Loop operation. Used with the Tx on Loop bit (D4 of CR10), this bit controls the loop.
   0: Disable
      Disables the loop function. If this bit is set to zero after the device entered a Loop operation, the TxD output is disconnected from the RxD input to open the SDLC Loop as soon as the GA pattern is detected. The TxD pin is subsequently set to "H".
1: Enable
   Enables the SDLC Loop to be formed. Setting only this bit to 1 connects the RxD input to TxD output within the AMPSC to form a Loop. In this case a 1-bit delay is not inserted, so that the input data to the RxD input is directly output to the next station through TxD. If the Tx on Loop bit is set in this state, the device starts GA pattern detection. For details of this operation, see the description of the Tx on Loop bit.

(7) D0 (SYNC Character Bits; valid in COP mode)
This bit determines the number of bits per SYNC character.
0: 8 bits
   Specifies a SYNC character length of 8 bits. This means that the SYNC character length is 8 bits in the Mono-sync mode, and is 16 bits in the Bi-sync mode.
1: 6 bits
   Specifies a SYNC character length of 6 bits. This means that the SYNC character length is 6 bits in the Mono-sync mode, and is 12 bits in the Bi-sync mode.

3.2.13 Control Register 11 (CR11)
Each bit of CR11 controls E/S interrupt requests which are generated by the E/S interrupt caused in the AMPSC. They are valid if the E/S INT is enabled (D0 of CR1 = 1). For details of the cause of interrupt assigned to each bit, refer to the description of SRI.

(1) D7 (Break/Abort Interrupt Enable)
   (a) Async/COP mode
       This bit serves as the Break IE function.
       0: Disable
          Disables E/S interrupt caused by detection/release of break state.
       1: Enable
          Enables E/S interrupt caused by detection/release of Break state.

   (b) BOP mode
       The function of the bit differs between the Normal BOP mode (HDLC/SDLC) and SDLC Loop mode:
       (i) HDLC/SDLC mode
           This bit serves as the Abort IE function.
           0: Disable
              Disables E/S interrupt caused by Abort detection/release.
           1: Enable
              Enables E/S interrupt caused by Abort detection/release.

       (ii) Loop mode
           This bit provides Abort/GA IE function.
           0: Disable
              Disables the E/S interrupt caused by detection/release of an Abort or GA.
           1: Enable
              Enables the E/S interrupt caused by detection/release of an Abort or GA.

(2) D6 (Transmit Underrun/End of Message Interrupt Enable; valid in COP or BOP mode)
   0: Disable
      Disables E/S interrupt caused by the Tx Underrun/EOM state.
   1: Enable
      Enables E/S interrupt caused by the Tx Underrun/EOM state.

(3) D5 (CTS Interrupt Enable)
   0: Disable
      Disables E/S interrupt caused by CTS pin status changes.
1: Enable
   Enables E/S interrupt caused by CTS pin status changes.

(4) D4 (SYNC/Hunt Interrupt Enable)
0: Disable
   Disables E/S interrupt caused by SYNC/Hunt status changes.
1: Enable
   Enables E/S interrupt caused by SYNC/Hunt status changes.

(5) D3 (DCD Interrupt Enable)
0: Disable
   Disables E/S interrupt caused by DCD pin status changes.
1: Enable
   Enables E/S interrupt caused by DCD pin status changes.

(6) D2 (All Sent Interrupt Enable; valid in Async or BOP mode)
0: Disable
   Disables E/S interrupt caused by All Sent state generation.
1: Enable
   Enables E/S interrupt caused by All Sent state generation.

(7) D1 (Idle Detect Interrupt Enable; valid in BOP mode)
0: Disable
   Disables E/S interrupt caused by Idle detection/release.
1: Enable
   Enables E/S interrupt caused by Idle detection/release.
(8) D0 (BRG Interrupt Enable)
This bit is valid only if Tx/Rx BRG is enabled (D1, D0 of CR14 = "1, 1"). It determines whether or not an E/S interrupt is to be requested when the BRG count reaches zero.

0: Disable
   Disables E/S interrupt even if the BRG count reaches zero.
1: Enable
   Enables E/S interrupt if the BRG count reaches zero.

In relation to this interrupt, see the descriptions of CR12, CR14, SR1, and SR3.

3.2.14 Control Register 12 (CR12)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>BRG Select for TRxC</td>
<td>BRG Select for DPLL</td>
<td>TxBRG IE</td>
<td>RxBRG IE</td>
<td>TxBRG Register Set</td>
<td>RxBRG Register Set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contents</td>
<td>0 RxBRG</td>
<td>0 RxBRG</td>
<td>0 0 Disable</td>
<td>0 Disable</td>
<td>0 No Operation 1 TxBRG Register Set</td>
<td>0 No Operation 1 RxBRG Register Set</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 TxBRG</td>
<td>1 TxBRG</td>
<td>0 1 Enable</td>
<td>1 Enable</td>
<td>1 Operation 1 TxBRG Register Set</td>
<td>1 Operation 1 RxBRG Register Set</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) D7 (BRG Select for TRxC)
This bit selects the source of the BRG clock which is output at the TRxC pin of the device. It is valid only when the TRxC pin is specified for output (D2 of CR15 = 1) and the BRG is selected for the source of the clock at the TRxC pin (D1, D0 of CR15 = 1, 0).

0: RxBRG
   The RxBRG clock is output at the TRxC pin.
1: TxBRG
   The TxBRG clock is output at the TRxC pin.

2-65
(2) D6 (BRG Select for DPLL)
This bit selects the clock source for the AMPSC's on-chip DPLL circuit, from TxBRG and RxERG. It is valid when the BRG clock is selected as the clock for the DPLL circuit (D7, D6, D5 CR14 = 1, 0, 0).
0: RxBRG
The RxBRG clock serves as the clock source for the DPLL.
1: TxBRG
The TxBRG clock serves as the clock source for the DPLL.

(3) D3 (Transmit BRG Interrupt Enable)
This bit determines whether or not an E/S interrupt is to be issued when the TxBRG count value reaches zero. It is valid when the BRG IE bit is set (D0 of CR11 = 1).
0: Disable
No E/S interrupt is issued even if the TxBRG count value reaches zero. The status of the BRG Zero Count bit (D0 of SR1) and TxBRG Zero Count bit (D4 of SR3) also do not change.
1: Enable
An E/S interrupt is issued when the TxBRG count value reaches zero, and the BRG Zero Count and TxBRG Zero Count bits are both set to one.

(4) D2 (Receive BRG Interrupt Enable)
This bit determines whether or not an E/S interrupt is to be issued when the RxBRG count value reaches zero. It is valid when the BRG IE bit is set (D0 of CR11 = 1).
0: Disable
No E/S interrupt occurs even if the RxBRG count value reaches zero. The BRG Zero Count bit and the RxBRG Zero Count bit (D3 of SR3) are both set to one.
1: Enable
An E/S interrupt occurs when the RxBRG count value reaches zero, and the BRG Zero Count and RxBRG Zero Count bits are both set to one.
(5) D1 (Transmit BRG Register Set)
This bit is used to set a count value into the TxBRG register.
0: No operation
   No write cycle for the TxBRG register is provided.
1: TxBRG Register Set
   The first two control data write cycles after this bit is set to one are used to write a count value into the TxBRG register. The lower byte is written in the first write cycle, and the upper byte is written in the second write cycle. This bit is automatically reset when a value is set into the registers.
   Register setting should be done when the TxBRG down counter is not operating, or, in other words, in the TxBRG Disable state (D0 of CR14 = 0). If a register write is attempted while the down counter is operating, the value will not be loaded into the counter until the counter value reaches zero.
   Setting both this bit and bit D0 of CR12 (RxBRG Register Set) at the same time is not allowed.

(6) D0 (Receive BRG Register Set)
This bit is used to set a count value into the RxBRG register.
0: No operation
   No write cycle for the RxBRG register is provided.
1: RxBRG Register Set
   The first two control word write cycles available after this bit is set are used to write a count value into the RxBRG registers. The lower byte is written in the first write cycle, and the upper byte is written in the second write cycle. This bit is automatically reset when a count value is set into the register. Register setting should be done when the RxBRG down counter is not operating, or, in other words, in the RxBRG Disable state (D1 of CR14 = 0). If a write operation is attempted while the down counter is operating, the value will not be loaded.
until the counter value reaches zero.
Setting both this bit and bit D1 of CR12 (TxRRCG Register Set) at the same time is not allowed.

3.2.15 Control Register 13 (CR13)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>TxDLC Enable</td>
<td>Stand-by Mode</td>
<td>Set</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contents</td>
<td>0 Disable</td>
<td>0 No Operation</td>
<td>1 Stand-by Mode</td>
<td>Set</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 Enable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) D1 (Transmit Data Length Counter Enable; valid in BOP mode)
This bit is used for TxDLC control. It enables or disables comparison of the Tx Data Length in CR8 and CR9 with the value of the TxDLC (SR8, SR9).
0: Disable
Disables the TxDLC function and comparison of its contents with the TxDL.
1: Enable
Enables the TxDLC function to start comparison of its contents with the TxDL. After initialization, this bit must be set before the transmitter is enabled (before transmission is started). If this bit is set, the TxDLC is incremented by one each time the TxINT/DMA request is activated. When the value of the TxDLC equals the TxDL value, the subsequent TxINT/DMA requests are masked, which causes counter increment operation to be stopped (counter value is preserved). If it is desired to activate the TxDLC in the next frame, this bit must be set up again (this set-up operation will clear the counter). The resetting of this bit should be done after the current frame is completed (after at
least one Ending flag is output from the TxD pin). If the Tx Underrun/EOM state (D6 of SR1 = 1) occurs when the TxDLC is enabled, the value of the TxDL is compared with that of the TxDLC. If the two values are identical, the current frame will be successfully completed after a CRC and flag are output. If they do not match, an Send Abort operation will be automatically executed, with TxINT/DMA requests masked. In this case the TxDLC value is preserved and the Sending Abort bit (D1 of SR0) is set to indicate that an Abort was automatically sent. The mask on TxINT/DMA requests is cleared by enabling the TxDLC again, which clears the TxDLC.

(2) D0 (Standby Mode Set)
This bit is used to place the AMPSC in the Standby mode.
0: No Operation
This condition results in no operation. The Standby mode, if set, can not be cleared by manipulating this bit but by writing "00H" into CR0.
1: Standby Mode Set
Places the AMPSC in the Standby mode, in which no external clock input other than the system clock (CLK) is input. This bit is automatically reset when the Standby mode is cleared.
For the Status of CRs and SRs in the Standby mode, see Table 3-2; for the pin status in Standby mode, see Table 3-8.
### Table 3-8 Pin Status in Standby Mode

<table>
<thead>
<tr>
<th>Pin name</th>
<th>I/O</th>
<th>Pin status</th>
<th>Pin name</th>
<th>I/O</th>
<th>Pin status</th>
</tr>
</thead>
<tbody>
<tr>
<td>WR</td>
<td>I</td>
<td>Retains the current state</td>
<td>TRxA, TRxB</td>
<td>O</td>
<td>Retains the current state</td>
</tr>
<tr>
<td>RD</td>
<td>I</td>
<td>RxDA, RxDB</td>
<td>TXA, TXB</td>
<td>O</td>
<td>--</td>
</tr>
<tr>
<td>B/A</td>
<td>I</td>
<td>TRxA, TRxB</td>
<td>RXCA, RXCB</td>
<td>I/O</td>
<td>High impedance</td>
</tr>
<tr>
<td>C/D</td>
<td>I</td>
<td>RTxA, RTxB</td>
<td>RXCA, RXCB</td>
<td>I</td>
<td>--</td>
</tr>
<tr>
<td>D7-D0</td>
<td>I/O</td>
<td>High impedance</td>
<td>XI2A/SYNA</td>
<td>I/O</td>
<td>High impedance</td>
</tr>
<tr>
<td>INT</td>
<td>O</td>
<td>Retains the current state</td>
<td>XI2B/SYNB</td>
<td>I/O</td>
<td>High impedance</td>
</tr>
<tr>
<td>INTAK</td>
<td>I</td>
<td></td>
<td>RTS, RTSB</td>
<td>O</td>
<td>Retains the current state</td>
</tr>
<tr>
<td>PRI</td>
<td>I</td>
<td></td>
<td>CTSA, CTSB</td>
<td>I</td>
<td>--</td>
</tr>
<tr>
<td>PR0</td>
<td>O</td>
<td>Depends on PRI</td>
<td>DCDA, DCDB</td>
<td>I</td>
<td>--</td>
</tr>
<tr>
<td>DRQTxA</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRQRxA</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DTRA/DRQTxB</td>
<td>O</td>
<td>Retains the current state</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DTRB/DRQRxB</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

During the Standby mode, the input and input/output pins must be controlled as shown in Table 3-9.

### Table 3-9 Input and Input/Output Pin Control During Standby Mode

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>WR</td>
<td>Fixed to &quot;H&quot;</td>
</tr>
<tr>
<td>RD</td>
<td></td>
</tr>
<tr>
<td>XI2A/SYNA</td>
<td>Fixed to &quot;H&quot; or &quot;L&quot;</td>
</tr>
<tr>
<td>XI2B/SYNB</td>
<td></td>
</tr>
<tr>
<td>CTSA, CTSB</td>
<td></td>
</tr>
<tr>
<td>DCDA, DCDB</td>
<td></td>
</tr>
<tr>
<td>others</td>
<td>Don't Care</td>
</tr>
</tbody>
</table>

The Standby mode is cleared when "00" is written into the control register 0 (WR = "L")
3.2.16 Control Register 14 (CR14)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>DPLL Command</td>
<td>Local Self Test</td>
<td>Echo Loop Test</td>
<td>BRG Source Select</td>
<td>RxBRG Enable</td>
<td>TxBRG Enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contents</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0 0</td>
<td>No Operation</td>
<td>0 No Operation</td>
<td>0 No Operation</td>
<td>0 Xtal/STRxC</td>
<td>0 Disable</td>
<td>0 Disable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0 1</td>
<td>Enter Search</td>
<td>1 Enable</td>
<td>1 Enable</td>
<td>1 System Clock</td>
<td>1 Enable</td>
<td>1 Enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1 0</td>
<td>Reset Missing Clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1 1</td>
<td>DPLL Disable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0 0</td>
<td>DPLL Source BRG Select</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0 1</td>
<td>DPLL Source Xtal/STRxC Select</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 0</td>
<td>FM Mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 1</td>
<td>MRZI Mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) D7, D6, D5 (DPLL Command)
These bits are used to control the DPLL circuit. When they are reset, the DPLL circuit is disabled, clock source input is assigned to the STRxC pin, and NRZI mode selected.

0 0 0: No operation

0 0 1: Enter Search
This command causes the DPLL circuit to start detection of edges in received data. It activates the DPLL circuit to start sampling the Rx data input. The DPLL circuit operation differs depending on the data format used.

0 1 0: Reset Missing Clock
This command is valid only when the FM mode is selected. It is used to reset the Clock Missing bits (D7 and D6 of SR10).

0 1 1: DPLL Disable
This command is used to stop the DPLL circuit operation. It also resets the Clock Missing bits (D7 and D6 of SR10).
1 0 0: DPLL Source BRG Select
This command allows the use of the BRG as the clock source for the DPLL. Whether the TxBRG of the RxBRG is to be used as the clock source is determined by the BRG Select for DPLL bit (D6 of CR12).

1 0 1: DPLL Source Xtal/STRxC Select
This command is issued when a crystal oscillator or a clock applied to the STRxC pin is to be used as a source clock for the DPLL. Selection between the crystal OSC and STRxC input is specified by the Xtal Select bit (D7 of CR15).

1 1 0: FM Mode
This command is issued when received serial data is to be treated in the FM data format.

1 1 1: NRZI Mode
This command is issued when received serial data is to be treated in the NRZI data format.

(2) D4 (Local Self Test)
This bit is used for local self test.
0: No operation
1: Enable
The transmitter output is directly connected to the receiver input within the AMPSC to loop back Tx data to the receiver (Fig. 3-8). External data applied to the RxD pin will be ignored. Once this test mode is selected, transmitter/receiver control from the CTS/DCD inputs becomes unavailable.
(3) D3 (Echo loop Test)
This bit is used to echo back received data directly to the remote sender. It is used for line tests.
0: No operation
1: Enable
The RxD input pin is connected to the TxD pin within the AMPSC so that received data is echoed back to the sender via the TxD pin (Fig. 3-9). In this case the AMPSC's transmitter function is invalidated, with transmitter control by CTS input disabled.

(4) D2 (BRG Source Select)
This bit selects the source clock for the on-chip BRG. The selected clock source is shared for both the TxBRG and the RxBRG.
If bit D7 of CR15 is set to one, the output of the crystal oscillator is used for the source clock for the BRG. If the same bit is set to zero, the clock applied to the STRxC pin of the device is used for the source clock.

1: System Clock
The system clock is used for the source clock for the BRG.

(5) D1 (Receive BRG Enable)
This bit is used for RxBRG control.
0: Disable
Disables the RxBRG, inhibiting count operation.
1: Enable
Enables the RxBRG to start a count operation. It takes 2 count periods after this bit is set before the receive clock is output.

(6) D0 (Transmit BRG Enable)
This bit is used for TxBRG control.
0: Disable
Disables the TxBRG, inhibiting count operation.
1: Enable
Enables the TxBRG to start count operation. It takes 2 count periods after this bit is set before the transmit clock is output.

3.2.17 Control Register 15 (CR15)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Xtal Select</td>
<td>RxCLK Select</td>
<td>TxCLK Select</td>
<td>TRxC Input/Output</td>
<td>Source Select</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contents</td>
<td>0 No</td>
<td>0 0 STRxC Input</td>
<td>0 0 STRxC Input</td>
<td>0 Input</td>
<td>0 0 Xtal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 Xtal</td>
<td>0 1 TRxC Input</td>
<td>0 1 TRxC Input</td>
<td>1 Output</td>
<td>0 1 TxCLK</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 Xtal</td>
<td>1 0 RxBRG Clock</td>
<td>1 0 TxBRG Clock</td>
<td>1 0 BRG Clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 1 DPLL Clock</td>
<td>1 1 DPLL Clock</td>
<td>1 1 DPLL Clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
(1) D7 (Xtal Select)
This bit determines whether the on-chip Xtal oscillator is to be used or not.
0: No Xtal
   The Xtal oscillator remains inoperative, so that no Xtal resonator can be connected to the device.
1: Xtal
   Makes the on-chip Xtal OSC circuit available, so that a Xtal resonator can be connected across pins XI1 and XI2.

(2) D6, D5 (Receive Clock Select)
These bits select the clock source for the receive clock.
0 0: STRxC Input
   The clock applied to the STRxC pin of the device is used as the receive clock.
0 1: TRxC Input
   The clock applied to the TRxC pin is used as the receive clock.
   If this clock is selected, bits D2, D1, and D0 of CR15 become invalid.
1 0: RxBRG Clock
   The output clock of the RxBRG is used as the receive clock.
1 1: DPLL Clock
   The output clock of the DPLL is used as the receive clock.

(3) D4, D3 (Transmit Clock Select)
These bits select the clock source for the transmit clock.
0 0: STRxC Input
   The clock applied to the STRxC pin of the device is used as the transmit clock.
0 1: TRxC Input
The clock applied to the TRxC pin is used as the transmit clock. If this clock is selected, bits D2, D1, and D0 of CR15 become invalid.

1 0: TxBRG Clock
The output clock of the TxBRG is used as the transmit clock.

1 1: DPLL Clock
The output clock of the DPLL is used as the transmit clock.

(4) D2 (TRxC Input/Output)
This bit selects the TRxC pin function from the input and output functions.

0: Input
The TRxC pin functions as an input.

1: Output
The TRxC pin functions as an output. If the pin is specified as an input by D6, D5, D4, and D3 of CR15, the setting of this bit is invalid.

(5) D1, D0 (TRxC Source Select)
These bits select the source of the clock output at the TRxC pin of the device. They are valid only if the TRxC pin is specified for output (D2 of CR15 = 1). If, however, the TRxC pin is specified for input by D6, D5, D4, and D3 of CR15, the setting of these bits are invalid.

0 0: Xtal
The output of the on-chip Xtal oscillator appears at the TRxC pin. This setting is valid only if the Xtal oscillator is used.

0 1: TxCLK
The transmit clock appears at the TRxC pin.

1 0: BRG Clock
The BRG's output clock appears at the TRxC pin. Selection between TxBRG and RxBRG is done with D7 of CR12.
11: DPLL Clock
The output of the DPLL appears at the TRxC pin.

3.3 Status Registers (SR)
This section describes the status registers that are used to indicate the AMPSC's device status. Status registers hold information indicating the current device status of the AMPSC. The host processor can manage AMPSC operations appropriately by reading status information from the SRs.

3.3.1 Status Register 0 (SR0)

<table>
<thead>
<tr>
<th>Bit Function</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>End of Frame</td>
<td>CRC/ Framing Error</td>
<td>Rx Overrun Error</td>
<td>Parity Error</td>
<td>Short Frame Detect</td>
<td>Tx Buffer Empty</td>
<td>Sending Abort</td>
<td>Rx Data Available</td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td>0</td>
<td>0 No Error</td>
<td>0 No Error</td>
<td>0 No Error</td>
<td>0 Tx Buffer Full</td>
<td>0 Tx Buffer Empty</td>
<td>1 Available</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Framing Error&quot;</td>
<td>1 Overrun Error</td>
<td>1 Parity Error</td>
<td>0</td>
<td>1 Tx Buffer Empty</td>
<td>0</td>
<td>0 Not Available</td>
<td></td>
</tr>
<tr>
<td>COP</td>
<td>0</td>
<td>0 No Error</td>
<td>1 Framing Error</td>
<td>&quot;CRC Error&quot;</td>
<td>0</td>
<td>0 Tx Buffer Full</td>
<td>0 Not Sending</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0 No Error</td>
<td>1 CRC Error</td>
<td>1</td>
<td>0</td>
<td>(Including CRC Transmission)</td>
<td>1 Sending Abort</td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td>0 Not EOF</td>
<td>1 EOF</td>
<td>0</td>
<td>0 Not EOF</td>
<td>1</td>
<td>1 Short Frame Detect</td>
<td>0</td>
<td>0 Not Sending</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0 Not EOF</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1 Sending Abort</td>
<td>1 Sending Abort</td>
<td></td>
</tr>
</tbody>
</table>

(1) D7 (End of Frame; valid in BOP mode)
This bit indicates whether reception of a frame is completed or not.
0: Not EOF
Indicates that reception of a frame is not completed yet. This bit, after setting upon completion of one-frame reception, is reset to zero by an Error Reset command or reception of the first data of the next frame (except in the First Rx INT mode).

1: EOF
This bit is set when an Ending flag is received, to indicate the end of a frame reception. The CRC Error bit (D6 of SR0) and Residue Code (D2-D0 of SR3) are validated when this bit is set. The EOF condition causes a Special Rx Condition interrupt.

(2) D6 (CRC/Framing Error)
This bit indicates a CRC error or framing error. The function of this bit differs depending on the operation mode.

(a) Asynchronous mode
This bit indicates a framing error. It is set if a zero is detected at the stop-bit position. It causes a Special Rx Condition interrupt.
This bit is reset by an Error Reset command or reception of a normal data item (containing no framing error).
A framing error does not affect reception of the next data item.

(b) COP or BOP mode
This bit indicates the result of the CRC calculation performed on received data. If set to one, it indicates a CRC error; if remaining at zero, it indicates no CRC error.
The procedure in which this bit is to be read has the following restriction: In the COP mode, read the bit after elapsing 20-bit time from when the last bit of the second CRC byte is input to the RxD pin, or after elapsing 16-bit time from when the second CRC byte is transferred to the Rx buffer. In the BOP mode, read the bit when the End of Frame bit (D7 of SR0) is set to one.
A CRC error does not cause a Special Rx Condition interrupt. This bit is reset by the Error Reset command.
(3) D5 (Receive Overrun Error)

This bit indicates an Rx Overrun error. An Rx Overrun error occurs if 1 data item or more is transferred to the Rx buffer when it is full.

For example, assume that three data items, D0, D1, and D2, are already in the Rx buffer (see Fig. 3-10). When the next data item, D3, is assembled in the shift register and transferred to the Rx buffer, an Overrun error will occur because the buffer location is occupied by data item D2 (see Fig. 3-11).

An Rx Overrun error causes a Special Rx Condition interrupt. The timings at which the Rx Overrun Error bit is set and the resulting Special Rx Condition interrupt occurs differ depending on the setting of the Overrun Error INT bit (D6 of CR1). For more details, see the description of CR1.

The Rx Overrun Error bit is latched, and is reset by the Error Reset command.

![Fig. 3-10 Rx Buffer Full](image1)

![Fig. 3-11 Rx Overrun](image2)
(4) D4 (Parity Error; valid in Asynchronous or COP mode)
This bit indicates the result of parity check. It is valid when parity is enabled. If the received parity bit does not match the predetermined parity (parity error), this bit will be set. The Parity Error bit is latched and is reset by an Error Reset command.
When in the All Receive INT-1 mode (D4, D3 of CR1 = 1, 0), a parity error causes a Special Rx Condition interrupt.

(5) D3 (Short Frame Detect; valid in BOP mode)
This bit is valid when Short Frame Detect Enable is selected (D7 of CR1 = 1). It is set when a short frame (in which data between two flags has less than 32 bits) is received, and is reset by the Error Reset command. Detection of a short frame causes a Special Rx Condition interrupt.

(6) D2 (Transmit Buffer Empty)
This bit indicates AMPSC's Tx buffer status. It gives an indication of when the host system may transfer Tx data to the AMPSC.
0: Tx Buffer Full
   Indicates that the Tx buffer is full with Tx data.
   This bit is also reset to zero when CRC transmission is busy in COP or BOP mode.
   When this bit is at zero, it is not possible correctly to write Tx data into the AMPSC. However, when the PAD character is to be sent following the CRC in the COP mode, the character can be written into the device even if this bit is at zero.
1: Tx Buffer Empty
   Indicates that the Tx buffer is empty. Tx data can be written into the AMPSC in this condition. This bit is, however, reset to zero when CRC transmission is busy in the COP or BOP mode, even if the Tx buffer is empty.
(7) D1 (Sending Abort; valid in BOP mode)
This bit indicates that the AMPSC entered the Sending Abort State. It is set to one when an Abort is sent in any of the following conditions:
(a) The Send Abort bit (D4 of CR5) is set to one.
(b) Tx Underrun occurred when the Tx Condition on Underrun bit (D2 of CR10) was set to one.
(c) Tx Underrun occurred when the TxDLC Enable bit (D1 of CR13) was set to one, and the value of the Tx Data Length (CR8, CR9) did not match that of the Tx Data Length Counter (SR8, SR9).
This bit is latched, and is reset by the Error Reset command. Status changes of this bit do not cause a interrupt.

(8) D0 (Receive Data Available)
This bit indicates whether or not valid receive data exists in the Rx buffer within the AMPSC.
0: Not Available
Indicates that no valid receive data exists in the Rx buffer.
1: Available
Indicates that at least one valid receive data character exists in the Rx buffer.
3.3.2 Status Register 1 (SR1)

<table>
<thead>
<tr>
<th>Bit Function</th>
<th>Operation protocol</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Break/Abort/GA Detect</td>
<td>Break Detect</td>
<td>D7</td>
<td>D6</td>
<td>D5</td>
<td>D4</td>
<td>D3</td>
<td>D2</td>
<td>D1</td>
<td>D0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tx</td>
<td>CTS</td>
<td>Sync/</td>
<td>DCD</td>
<td>All</td>
<td>Idle</td>
<td>BRG</td>
<td>Count</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Status Register 1 consists of E/S bits to indicate the causes of E/S interrupts. The AMPSC uses the E/S bits to indicate its internal status changes or input pin status changes to the host processor.

If E/S INT is enabled (D0 of CR1 = 1) and interrupt by an individual E/S bit is enabled, the changes in the pertinent E/S bit status are latched and cause an E/S interrupt.

If E/S interrupt is disabled, changes in E/S bit status will not be latched, simply indicating the changes in condition.
(1) D7 (Break/Abort/Go Ahead Detect; valid in Async or BOP mode)

The function of this bit differs depending on the operation mode:

(a) Asynchronous mode

This bit provides the Break (character in which the start, stop, and data bits are all zeros) Detect function.

0: Not Detect
   Indicates that no break has occurred.
1: Break Detect
   Indicates that a break was detected.

The changes in the Break Detect bit status are latched to cause an E/S interrupt. Data (null data) received during the Break Detect state is not available. Therefore, the Break condition cannot be read as a character.

(b) BOP mode

The function of this bit differs between the Normal BOP mode (HDLC/SDLC) and SDLC Loop mode:

(i) HDLC/SDLC mode

This bit provides the Abort (7 or more consecutive 1s) Detect function.

0: Not Detect
   Indicates that no Abort has been detected or an Abort condition was cleared.
1: Abort Detect
   Indicates that an Abort was detected.

The changes in the Abort Detect bit are latched to cause an E/S interrupt. The AMPSC's Abort Detect feature becomes valid when a start flag is detected after reception is enabled. When in the Address Search mode (D2 of CR3 = 1), the Abort Detect function remains invalid until an address field with the same value as that present in CR6 is detected. Once an address match has occurred, abort detection is subsequently done unconditionally. If it is
desired to give the same function to subsequent frames, D2 of CP3 must be set to one again.

(ii) Loop mode
This bit serves the Abort/GA (Go-Ahead: LSR 01111111) Detect function.
0: Not Detect
Indicates that no Abort/GA was detected or released (zero received after detection).
1: Abort/GA Detect
Indicates that a Abort/GA was detected.
The changes in the Abort/GA Detect bit status are latched to cause an E/S interrupt.

(2) D6 (Transmit Underrun/End of Message; valid in COP or BOP mode)
This bit indicates the Tx Underrun/EOM state (in which all Tx data has been transmitted and no Tx data remains in the Tx buffer). CRC transmission in the Tx Underrun/EOM state can be controlled by manipulating this bit.
0: Indicates that the device is not in the Tx Underrun/EOM state. If CRC transmission is desired in the Tx Underrun/EOM state, this bit must be reset to zero by the Reset Tx Underrun/EOM bit command (D7, D6 of CR0 = 1, 1). Before issuing this command, transmission must be enabled and at least one Tx data item must be transferred to the AMPSC.
In the BOP mode, however, this bit is automatically reset to zero when the first Tx data is transferred after transmission is enabled.
A status change from one to zero of this bit does not cause an E/S interrupt.
1: Indicates that the device entered the Tx Underrun/EOM state. This bit is reset before the device enters the Tx Underrun/EOM state. If it is set to one by the Tx Underrun/EOM state, the CRC will be transmitted if it is enabled, and an E/S interrupt will occur. This bit is set to one by a reset operation (system or channel reset), Send Abort, or transmission disable, and this change causes an E/S interrupt.

(3) D5 (Clear to Send)
This bit indicates the CTS pin status.
0: CTS pin = "H"
1: CTS pin = "L"
The status changes of this bit are latched to cause an E/S interrupt. However, if the CTS IE bit (D5 of CR11) is zero, this bit only indicates the CTS pin input status and is not latched.

(4) D4 (Sync/Hunt)
This bit indicates the SYNC pin status or AMPSC's synchronization establishment state. Its function differs depending on the operation mode:
(a) Async or External COP mode
This bit indicates the SYNC pin status.
0: SYNC pin = "H"
1: SYNC pin = "L"
The status changes of this bit are latched to cause an E/S interrupt, provided the crystal oscillator is not used (D7 of CR15 = 0).
(b) Internal COP mode or BOP mode
This bit indicates the AMPSC's synchronization establishment state.
0: Exit Hunt Phase
   Indicates that synchronization is established.
1: Hunt Phase
   Indicates the Hunt Phase (sync pattern detection busy or receive operation stop). This bit is set to one by the receiver disable or Enter Hunt Phase (D4 of CR3 = 1) operation, and reset to zero when a sync pattern is detected. The status changes of this bit are latched to cause an E/S interrupt.

(5) D3 (Data Carrier Detect)
   This bit indicates the DCD pin status.
   0: DCD pin = "H"
   1: DCD pin = "L"
   The status changes of this bit are latched to cause an E/S interrupt. However, if the DCD IE bit (D3 of CR11) is zero, this bit only indicates the DCD pin input status and will not be latched.

(6) D2 (All Sent; valid in Async or BOP mode)
   This bit indicates that all Tx data within the AMPSC has been transmitted.
   (a) Asynchronous mode
      0: Not All Sent
         Indicates that Tx data remains in the AMPSC. This bit is always zero when the Tx is disabled.
      1: All Sent
         Indicates that all Tx data within the AMPSC has been transmitted and no data remains in the Tx buffer or Tx shift register.
         This bit is reset to zero when Tx data is transferred to the AMPSC. Zero-to-one transition of this bit is latched to cause an E/S interrupt.

(7) D1 (Idle Detect; valid in BOP mode)
   This bit indicates detection of the Idle state (15 or more consecutive 1s).
   0: Not Idle Detect
      Indicates that no Idle state is detected or the Idle state has been finished.
1: Idle Detect
   Indicates that the Idle state is detected. Status changes of this bit are latched to cause an E/S interrupt.

(8) D0 (BRG Zero Count)
This bit indicates that the BRG count value reached zero.
0: Not Zero
   Indicates that the BRG count value has not reached zero. This bit is always zero if the BRG IE bit (D0 of CR11) is zero.
1: Zero
   Indicates that the BRG count value reached zero. Zero-to-one transition of this bit is latched to cause an E/S interrupt. It is reset by the Reset E/S Bit Latch command.
Which BRG count value out of TxBRG and RxBRG can be determined from D4 and D3 of SR3.
### 3.3.3 Status Register 2B (SR2B)

<table>
<thead>
<tr>
<th>Bit Function Vector Type</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector Value</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type A</td>
<td>V7</td>
<td>V6</td>
<td>V5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 Ch-B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 Ch-A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1 External/Status</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0 Rx: Data Available</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 Special Rx Condition</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type B</td>
<td>V7</td>
<td>V6</td>
<td>V5</td>
<td>V4</td>
<td>V3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 Ch-B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 Ch-A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1 External/Status</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0 Rx: Data Available</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 Special Rx Condition</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Status Register 2B indicates the value of an interrupt vector (cause of interrupt). The vector value indicated by this register differs depending on the value of D6 of CR2A (Status Affects Vector bit).

(a) If D6 of CR2A = 0
The value set in CR2B is directly output.

(b) If D6 of CR2A = 1
The value of SR2B differs depending on the cause of interrupt. The bits of SR2B which are affected by the cause of interrupt depend on the Output Vector Type setting. V4, V3, and V2 are affected for Type A; V2, V1, and V0 are affected for Type B, and all other bits are directly output without being changed.
SR2B is normally used for analyzing the cause of interrupt by the AMPSC in an interrupt service routine in the Non-Vector mode (D7 of CR2A = 0).

When Tx/Rx data is to be transferred by using status polling, the interrupt vector set in SR2B can be treated as status by placing the AMPSC in the Non-Vector mode, Status Affects Vector (D6 of CR2A = 1), and Both Channel INT (D1, D0 of CR2A = 0, 0).

3.3.4 Status Register 3 (SR3)

<table>
<thead>
<tr>
<th>Bit Function Operation protocol</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Async</td>
<td>Unknown</td>
<td>0 Not Zero</td>
<td>0 Not Zero</td>
<td>0 Not Zero</td>
<td>0 Not Zero</td>
<td>0 Not Zero</td>
<td>0 Not Zero</td>
<td>0 Not Zero</td>
</tr>
<tr>
<td>COP</td>
<td>Unknown</td>
<td>1 Zero</td>
<td>1 Zero</td>
<td>1 Zero</td>
<td>1 Zero</td>
<td>1 Zero</td>
<td>1 Zero</td>
<td>1 Zero</td>
</tr>
<tr>
<td>BOP</td>
<td>Unknown</td>
<td>(when there is no remainder)</td>
<td>5 Bits/Char. 1 0 0</td>
<td>6 Bits/Char. 0 0 0</td>
<td>7 Bits/Char. 0 1 1</td>
<td>8 Bits/Char. 0 1 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1) D4 (TxBRG Zero Count)
This bit is meaningful when the TxBRG is enabled (D0 of CR14 = 1).
0: Not Zero
   Indicates that the TxBRG count value does not reach zero.
1: Zero
   Indicates that the TxBRG count value reached zero.
Zero-to-one transition of this bit is latched to cause an E/S interrupt. It is reset by the Reset E/S Bit Latch command. The BRG Zero Count bit (D0 of SR1) is one whenever this bit is set at one. The bit is zero whenever the BRG IE bit (D0 of CR11) is at zero.

(2) D3 (RxBRG Zero Count)
This bit is meaningful when RxBRG is enabled (D1 of CR14 = 1).
0: Not Zero
   Indicates that the RxBRG count value is not zero.
1: Zero
Indicates that the RxBRG count value is zero.
Zero-to-one transition of this bit is latched to cause an E/S interrupt. It is reset by the Reset E/S Bit Latch command. The BRG Zero Count bit (D0 of SRI) is one whenever this bit is set at one. The bit is zero whenever the BRG IF bit (D0 of CR11) is at zero.

(3) D2-D0 (Residue Code; valid in EOF mode)
These bits indicate the valid range of data bits in the information field of a frame at the end of frame reception. The range of valid data bits can be determined by reading these bits when the AMPSC entered the EOF state. Table 3-10 shows the meanings of the Residue Code for different data lengths.

(Example) IF the Residue Code indicates "000" when data of 8-bits is received as shown in Fig. 3-12, the valid I field bits are ...Dj7, Dk0, Dk1.
Table 3-10 Meanings of Residue Code

<table>
<thead>
<tr>
<th>Data length</th>
<th>Residue Code</th>
<th>I field range (effective I field bit)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>D2 D1 DO</td>
<td>nth byte</td>
</tr>
<tr>
<td>5</td>
<td>0 0 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 0 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 1 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 0 0*</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 1 0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0 0 0*</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 0 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 1 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 0 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 0 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 1 0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0 0 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 0 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 1 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 1 1*</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 0 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 0 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 1 0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0 0 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 0 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 1 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0 1 1*</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 0 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 0 1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 1 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1 1 1</td>
<td>0</td>
</tr>
</tbody>
</table>

Di: Indicates valid data bits

o: Indicates all bits (bits 5, 6, 7 and 8) are valid.
-: Indicates all bits (bits 5, 6, 7 and 8) are invalid (denotes CRC).

*: When there is no remainder. Denotes the case when the boundary of the last receive data matches the boundary between the I field and CRC.
Fig. 3-12 Valid Bits in I Field (Residue Code = "000")

3.3.5 Status Register 4A (SR4A)

<table>
<thead>
<tr>
<th>Bit Function</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation protocol</td>
<td></td>
<td></td>
<td>Ch-A</td>
<td>Ch-A</td>
<td>Ch-A</td>
<td>Ch-A</td>
<td>Ch-B</td>
<td>Ch-B</td>
</tr>
<tr>
<td>Async</td>
<td>0 Not Pending</td>
<td>Rx INT Pending</td>
<td>Rx INT Pending</td>
<td>E/S INT Pending</td>
<td>Rx INT Pending</td>
<td>Tx INT Pending</td>
<td>E/S INT Pending</td>
<td></td>
</tr>
<tr>
<td>COP</td>
<td>Pending</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Recieved from the ref

Bits in the enclosed portion become valid in I field.
Each bit of Status Register 4A indicates whether or not a corresponding cause of an interrupt exists within the AMPSC. It is set to one when the pertinent interrupt is pending, being serviced or existing another cause of interrupt with a higher priority. It is set at zero in any other case.

While this register exists only on channel A, its function is shared for both channels.

(1) D7, D6 (Special Rx Condition INT Pending)
These bits indicate that the cause of a Special Rx Condition interrupt exists. Bits D7 and D6 indicate the presence of the cause of a Special Rx Condition interrupt on channel A and channel B, respectively. These bits are set when a Special Rx Condition occurs, even if Rx INT is disabled (D4, D3 of CR1 = 0, 0).

(2) D5 (Channel A Rx INT Pending)
This bit indicates the presence of an Rx interrupt on channel A.

(3) D4 (Channel A Tx INT Pending)
This bit indicates the presence of a Tx interrupt on channel A.

(4) D3 (Channel A E/S INT Pending)
This bit indicates the presence of an E/S interrupt on channel A.

(5) D2 (Channel B Rx INT Pending)
This bit indicates the presence of an Rx interrupt on channel B.

(6) D1 (Channel B Tx INT Pending)
This bit indicates the presence of a Tx interrupt on channel B.
(7) D0 (Channel B E/S INT Pending)
This bit indicates the presence of an E/S interrupt on channel B.

3.3.6 Status Register 8 (SR8)

<table>
<thead>
<tr>
<th>Bit Function Operation protocol</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tx Data Length Counter L-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td>Unknown</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COP</td>
<td>Tx Data Length Counter Bit7-Bit0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Status Register 8 (SR8) is used with SR9 to indicate the Tx interrupt/DMA request activation count. These registers are made valid if the Tx Data Length Counter is enabled (D1 of CR13 = 1).

(1) D7-D0 (Tx Data Length Counter L-Byte; valid in BOP mode)
These bits indicate the lower byte (bits 7-0) of the Tx interrupt request or Tx DMA request count. This register is usually used to judge whether the Tx Underrun/EOM state, if occurred, is legal or not. In this state the contents of SR8 and SR9 will be preserved. If the value of CR8/CR9 does not match with that of SR8/SR9 when the Tx Underrun/EOM state occurred, the AMPSC automatically transmits an Abort. SR8 and SR9 are cleared in the following cases:
1) Reset
2) Tx Data Length Counter Enable bit is set to one.
### 3.3.7 Status Register 9 (SR9)

<table>
<thead>
<tr>
<th>Bit Function Operation protocol</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Tx Data Length Counter H-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Async</td>
<td></td>
<td></td>
<td>Unknown</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOP</td>
<td></td>
<td></td>
<td>Tx Data Length Counter Bit15-Bit8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. D7-D0 (Transmit Data Length Counter H-Byte; valid in BOP mode)
   These bits indicate the upper byte of the Tx interrupt request or Tx DMA request count. This register is used with SR8. The description of this register is identical to that for SR8, with the exception that SR9 indicates the upper byte of count data.

### 3.3.8 Status Register 10 (SR10)

<table>
<thead>
<tr>
<th>Bit Function Operation protocol</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>One Clock Missing</td>
<td>Two Clocks Missing</td>
<td>Sending on Loop</td>
<td></td>
<td></td>
<td>Tx Sync/on Loop</td>
</tr>
<tr>
<td>Async</td>
<td></td>
<td></td>
<td>0 Not Missing</td>
<td>0 Not Missing</td>
<td>Unknown</td>
<td></td>
<td></td>
<td>Unknown</td>
</tr>
<tr>
<td>COP</td>
<td></td>
<td></td>
<td>1 Missing</td>
<td>1 Missing</td>
<td>Unknown</td>
<td></td>
<td></td>
<td>&quot;Tx Sync&quot;</td>
</tr>
<tr>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Unknown</td>
<td></td>
<td></td>
<td>0 Not</td>
</tr>
<tr>
<td>BOP Loop</td>
<td>0 Not</td>
<td></td>
<td>Sending</td>
<td>1 Sending</td>
<td>Unknown</td>
<td></td>
<td></td>
<td>1 Tx Sync</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Unknown</td>
<td></td>
<td></td>
<td>Unknown</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Unknown</td>
<td></td>
<td></td>
<td>Unknown</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Unknown</td>
<td></td>
<td></td>
<td>&quot;on Loop&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 Not</td>
<td></td>
<td></td>
<td>0 Not</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>on Loop</td>
<td></td>
<td></td>
<td>1 on Loop</td>
</tr>
</tbody>
</table>

2-95
(1) D7 (one Clock Missing)
This bit indicates whether or not serial data contains edges. It is valid only when the FM data format is selected and the DPLL is in use.
When FM is used for the data format, an edge (rising or falling edge) occurs within a 1-bit time (at a bit boundary or center of a bit), unlike NRZ or NRZI. The DPLL uses this edge as a reference for clock generation. The DPLL identifies the position of edges in serial data, and generates a clock based on this position. If no edge exists, clock generation by the DPLL may result in failure. To prevent this, the host processor uses this bit to identify whether or not an edge exists within the specified range of time.
The DPLL detects edges at every two bits.
0: Not Missing
   Indicates that edges were detected in serial data.
1: Missing
   Indicates an edge was not detected in serial data. This bit is latched, and is reset by the Reset Missing Clock command (D7, D6, D5 of CR14 = 0, 1, 0) or Enter Search command (D7, D6, D5 of CR14 = 0, 0, 1).

(2) D6 (Two Clocks Missing)
This bit indicates whether edges were detected in serial data. It is valid when FM is selected for the data format and the DPLL is in use. (The basic function of this bit is identical to that of the One Clock Missing bit).
0: Not Missing
   Indicates that edges were detected in serial data. It remains zero if only one edge was not detected.
1: Missing
   Indicates that edges were not detected in serial data two times consecutively.
(3) D4 (Sending on Loop; valid in BOP mode)
This bit indicates that the AMPSC formed an SDLC Loop and is busy for transmission. It is valid when the SDLC Loop mode is selected (D4, D1 of CR10 = 1, 1).
0: Not Sending
   Indicates that the AMPSC is not doing a loop transmission.
1: Sending
   Indicates that the AMPSC is doing a loop transmission.

(4) D1 (Tx Sync/on Loop; valid in COP or BOP mode)
The function of this bit differs depending on the operation mode.
(a) COP mode
   This bit indicates the Tx Sync state.
   0: Not Tx Sync
      Indicates that the receiver is not synchronized with the transmitter. This bit is also reset to zero if the Auto Tx on Sync bit (D4 of CR10) or D4 Enable bit (D1 of CR10) is reset.
   1: Tx Sync
      Indicates that synchronization between the transmitter and receiver is established (SYNC character detection on receiver completed) after both the Auto Tx on Sync and D4 Enable bits were set, and transmission is enabled for the device.
(b) BOP mode
   This bit serves the on Loop function during SDLC Loop operation, and indicates the repeat operation state of the AMSPC.
   0: Not on Loop
      Indicates that no GA pattern (01111111) is detected even if the SDLC Loop is formed (D4, D1 of CR10 = 1, 1). In this state a 1-bit delay is not inserted. This bit is also reset to zero when the device releases the SDLC Loop (D1 of CR10 = 0).
1: on Loop
Indicates that a GA pattern was detected and a
1-bit delay was inserted between the RxD input
and TxD output. This bit is maintained at one
while the SLDC Loop is formed.

3.3.9 Status Register 11 (SR11)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Break/</td>
<td>Tx Underrun/</td>
<td>CTS</td>
<td>Sync/</td>
<td>DCD</td>
<td>All</td>
<td>Idle</td>
<td>BRG</td>
</tr>
<tr>
<td></td>
<td>Abort</td>
<td>EOM</td>
<td>IE</td>
<td>Hunt</td>
<td>IE</td>
<td>Sent</td>
<td>Detect</td>
<td>IE</td>
</tr>
<tr>
<td></td>
<td>IE</td>
<td>IE</td>
<td>IE</td>
<td>IE</td>
<td>IE</td>
<td>IE</td>
<td>IE</td>
<td>IE</td>
</tr>
</tbody>
</table>

This register directly indicates the value set in CR11. The
host processor can use this register to control interrupt
enable for the AMPSC.

3.3.10 Status Register 12 (SR12)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Rx BRG Counter Constant L-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This register indicates the lower 8 bits (bits 7-0) of the
value set in the Rx BRG.

3.3.11 Status Register 13(SR13)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Rx BRG Counter Constant H-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This register indicates the upper 8 bits (bits 15-8) of the
value set in the Rx BRG.
3.3.12 Status Register 14 (SR14)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Tx BRG Counter Constant L-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This register indicates the lower 8 bits (bits 7-0) of the value set in the Tx BRG.

3.3.13 Status Register 15 (SR15)

<table>
<thead>
<tr>
<th>Bit</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Tx BRG Counter Constant H-Byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This register indicates the upper 8 bits (bits 15-8) of the value set in the Tx BRG.
4. SYSTEM CONFIGURATION EXAMPLE

The following figure shows a system configuration example in which the uPD72001 is interfaced with a modem, with both channels placed in the DMA mode.
μPD72001

ADVANCED MULTIPROTOCOL SERIAL CONTROLLER
3. Electrical Specifications

Absolute Maximum Ratings (Ta = 25°C)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>Ratings</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Supply Voltage</td>
<td>( V_{DD} )</td>
<td></td>
<td>-0.5 to +7.0</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>( V_I )</td>
<td></td>
<td>-0.5 to ( V_{DD} +0.5 )</td>
<td>V</td>
</tr>
<tr>
<td>Output Voltage</td>
<td>( V_O )</td>
<td></td>
<td>-0.5 to ( V_{DD} +0.5 )</td>
<td>V</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>( T_{opt} )</td>
<td></td>
<td>-10 to +70</td>
<td>°C</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>( T_{stg} )</td>
<td></td>
<td>-65 to +150</td>
<td>°C</td>
</tr>
</tbody>
</table>

DC Characteristics (Ta = -10 to +70°C, \( V_{DD} = 5V \pm 10\% \))

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>MIN.</th>
<th>TYP.</th>
<th>MAX.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Voltage High</td>
<td>( V_{IH} )</td>
<td>Clock pin</td>
<td>+3.3</td>
<td>( V_{DD} +0.5 )</td>
<td>( V_{DD} +0.5 )</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>( V_{I} )</td>
<td>Other pins</td>
<td>+2.2</td>
<td>( V_{DD} +0.5 )</td>
<td>( V_{DD} +0.5 )</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage Low</td>
<td>( V_{IL} )</td>
<td>Clock pin</td>
<td>-0.5</td>
<td>+0.6</td>
<td>+0.8</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>( V_{I} )</td>
<td>Other pins</td>
<td>-0.5</td>
<td>+0.6</td>
<td>+0.8</td>
<td>V</td>
</tr>
<tr>
<td>Output Voltage High</td>
<td>( V_{OH} )</td>
<td>( I_{O} = -400\mu A )</td>
<td>0.7( V_{DD} )</td>
<td>+0.45 V</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Output Voltage Low</td>
<td>( V_{OL} )</td>
<td>( I_{O} = 2.0mA )</td>
<td>+0.45 V</td>
<td>+0.45 V</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Input Leakage Current High</td>
<td>( I_{LH} )</td>
<td>( V_{I} = V_{DD} )</td>
<td>+10 uA</td>
<td>+10 uA</td>
<td>uA</td>
<td></td>
</tr>
<tr>
<td>Input Leakage Current Low</td>
<td>( I_{LL} )</td>
<td>( V_{I} = 0V )</td>
<td>-10 uA</td>
<td>-10 uA</td>
<td>uA</td>
<td></td>
</tr>
<tr>
<td>Output Leakage Current High</td>
<td>( I_{LOH} )</td>
<td>( V_{O} = V_{DD} )</td>
<td>+10 uA</td>
<td>+10 uA</td>
<td>uA</td>
<td></td>
</tr>
<tr>
<td>Output Leakage Current Low</td>
<td>( I_{LOL} )</td>
<td>( V_{O} = 0V )</td>
<td>-10 uA</td>
<td>-10 uA</td>
<td>uA</td>
<td></td>
</tr>
<tr>
<td>Supply Current</td>
<td>( I_{DD} )</td>
<td>8MHz Operation</td>
<td>20 mA</td>
<td>40 mA</td>
<td>mA</td>
<td></td>
</tr>
</tbody>
</table>

Capacitance (Ta = 25°C, \( V_{DD} = 0V \))

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>MIN.</th>
<th>MAX.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Capacitance</td>
<td>( C_{IN} )</td>
<td>( f = 1MHz )</td>
<td>10</td>
<td>pF</td>
<td></td>
</tr>
<tr>
<td>I/O Capacitance</td>
<td>( C_{IO} )</td>
<td>Unmeasured pins returned to 0V.</td>
<td>20</td>
<td>pF</td>
<td></td>
</tr>
</tbody>
</table>
AC Characteristics (Ta = -10 to +70°C, $V_{CC} = +5V \pm 10\%$)

**System Interface**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>Standard value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock Cycle</td>
<td>$t_{CYK}$</td>
<td></td>
<td>125</td>
<td>2000 ns</td>
</tr>
<tr>
<td>Clock high-level width</td>
<td>$t_{WKH}$</td>
<td></td>
<td>50</td>
<td>1000 ns</td>
</tr>
<tr>
<td>Clock low-level width</td>
<td>$t_{WKL}$</td>
<td></td>
<td>50</td>
<td>1000 ns</td>
</tr>
<tr>
<td>Clock rise time</td>
<td>$t_{KR}$</td>
<td>1.5V to 3.0V</td>
<td>10</td>
<td>ns</td>
</tr>
<tr>
<td>Clock fall time</td>
<td>$t_{KF}$</td>
<td>3.0V to 1.5V</td>
<td>10</td>
<td>ns</td>
</tr>
<tr>
<td>Address setup time to RD↓</td>
<td>$t_{SAR}$</td>
<td></td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>Address hold time from RD↑</td>
<td>$t_{HRA}$</td>
<td></td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>RD pulse width</td>
<td>$t_{WRL}$</td>
<td></td>
<td>150</td>
<td>ns</td>
</tr>
<tr>
<td>Data output delay time from address</td>
<td>$t_{DAD}$</td>
<td></td>
<td>120</td>
<td>ns</td>
</tr>
<tr>
<td>Data output delay time from RD</td>
<td>$t_{DRD}$</td>
<td></td>
<td>120</td>
<td>ns</td>
</tr>
<tr>
<td>Data float delay time from RD</td>
<td>$t_{FRD}$</td>
<td></td>
<td>10</td>
<td>85 ns</td>
</tr>
<tr>
<td>Address setup time to WR↓</td>
<td>$t_{SAW}$</td>
<td></td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>Address hold time from WR↑</td>
<td>$t_{HWA}$</td>
<td></td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>WR pulse width</td>
<td>$t_{WWL}$</td>
<td></td>
<td>150</td>
<td>ns</td>
</tr>
<tr>
<td>Data setup time to WR↑</td>
<td>$t_{SDW}$</td>
<td></td>
<td>120</td>
<td>ns</td>
</tr>
<tr>
<td>Data hold time from WR↑</td>
<td>$t_{HWD}$</td>
<td></td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>RD/ WR recovery time</td>
<td>$t_{RV}$</td>
<td></td>
<td>160</td>
<td>ns</td>
</tr>
</tbody>
</table>
Serial control

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>Standard value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmit/Receive data cycle</td>
<td>$t_{CYD}$</td>
<td></td>
<td>5</td>
<td>$t_{CYK}$</td>
</tr>
<tr>
<td>STRxC, TRxC input clock cycle</td>
<td>$t_{CYC}$</td>
<td></td>
<td>125</td>
<td>ns</td>
</tr>
<tr>
<td>STRxC, TRxC input clock pulse width</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High</td>
<td>$t_{WCH}$</td>
<td></td>
<td>50</td>
<td>ns</td>
</tr>
<tr>
<td>Low</td>
<td>$t_{WCL}$</td>
<td></td>
<td>50</td>
<td>ns</td>
</tr>
<tr>
<td>TxD delay time from STRxC, TRxC</td>
<td>$t_{DTCTD1}$</td>
<td>x 1 mode</td>
<td>100</td>
<td>ns</td>
</tr>
<tr>
<td></td>
<td>$t_{DTCTD2}$</td>
<td>x 16,32,64 mode</td>
<td>300</td>
<td>ns</td>
</tr>
<tr>
<td>RxD setup time to STRxC, TRxC</td>
<td>$t_{SRDRC}$</td>
<td></td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>RxD hold time from STRxC, TRxC</td>
<td>$t_{HCRD}$</td>
<td></td>
<td>140</td>
<td>ns</td>
</tr>
<tr>
<td>INT delay time from TxD</td>
<td>$t_{DTDIQ}$</td>
<td>Tx INT mode</td>
<td>4</td>
<td>6 $t_{CYK}$</td>
</tr>
<tr>
<td></td>
<td>$t_{DTDDQ}$</td>
<td>Tx DMA mode</td>
<td>4</td>
<td>6 $t_{CYK}$</td>
</tr>
<tr>
<td>INT delay time from RxC*</td>
<td>$t_{DRCIQ}$</td>
<td>Rx INT mode</td>
<td>7</td>
<td>11 $-t_{CYK}$</td>
</tr>
<tr>
<td></td>
<td>$t_{DRCDQ}$</td>
<td>Rx DMA mode</td>
<td>7</td>
<td>11 $-t_{CYK}$</td>
</tr>
<tr>
<td>DRQRx delay time from RxC*</td>
<td>$t_{DRDQ}$</td>
<td></td>
<td>120</td>
<td>ns</td>
</tr>
<tr>
<td>DRQTx delay time from WR*</td>
<td>$t_{DWDQ}$</td>
<td></td>
<td>120</td>
<td>ns</td>
</tr>
</tbody>
</table>

*: STRxC or TRxC, which is used for the receive clock.
## Interrupt control

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>Standard value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTAK low-level width</td>
<td>( t_{\text{WIAL}} )</td>
<td></td>
<td>150</td>
<td>ns</td>
</tr>
<tr>
<td>PRO delay time</td>
<td>( t_{\text{DPIPO}} )</td>
<td></td>
<td>50</td>
<td>ns</td>
</tr>
<tr>
<td>PRI setup time to INTAK ( \downarrow )</td>
<td>( t_{\text{SPIIA}} )</td>
<td>When vector output is permitted.</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>PRI hold time from INTAK ( \uparrow )</td>
<td>( t_{\text{HIAPI}} )</td>
<td>When vector output is permitted.</td>
<td>20</td>
<td>ns</td>
</tr>
<tr>
<td>Data output delay time from INTAK</td>
<td>( t_{\text{DIAD}} )</td>
<td></td>
<td>120</td>
<td>ns</td>
</tr>
<tr>
<td>Data float delay time from INTAK</td>
<td>( t_{\text{FIAD}} )</td>
<td></td>
<td>10</td>
<td>85</td>
</tr>
</tbody>
</table>

## Modem control

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>Standard value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTS, DCD, SYNC pulse width</td>
<td>High</td>
<td>( t_{\text{WMH}} )</td>
<td>2</td>
<td>( t_{\text{CYK}} )</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>( t_{\text{WML}} )</td>
<td>2</td>
<td>( t_{\text{CYK}} )</td>
</tr>
<tr>
<td>INT delay time from CTS, DCD, SYNC</td>
<td>( t_{\text{DMIQ}} )</td>
<td></td>
<td>2</td>
<td>( t_{\text{CYK}} )</td>
</tr>
<tr>
<td>SYNC delay time from ( \overline{\text{STRx}} ), ( \overline{\text{TRx}} ) ( \uparrow )</td>
<td>( t_{\text{DTRKSY}} )</td>
<td>COP external synchronization</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>
Crystal oscillation and reset

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>Standard value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>XII Input cycle time</td>
<td>$t_{CYX}$</td>
<td></td>
<td>125</td>
<td>1000</td>
</tr>
<tr>
<td>RESET pulse width</td>
<td>$t_{WRSL}$</td>
<td></td>
<td>2</td>
<td>$t_{CYK}$</td>
</tr>
</tbody>
</table>

Note: At all modes, system clock cycle must be more than five times of data rate.

AC test I/O wave (except clock)

AC test clock input wave

Load circuit for AC test

$C_L$ includes the jig.
Timing Wave

Clock timing

Read cycle timing

Write cycle timing

Read/Write cycle timing (data transfer by software)
Transmit cycle timing

Receive cycle timing

DMA request reset timing
INTAK cycle timing

E/S timing

SYNC input timing (external synchronization mode)

*: SYNCA/B input must be set to 0 at the rising edge of RXC after 2 clock cycles following the last bit of SYNC character.
XII input timing

RESET pulse
μPD7210

GPIB-CONTROLLER
CHAPTER 1
INTRODUCTION

The uPD7210 GPIB-IFC is a general-purpose interface bus interface controller. It interfaces between the interface bus specified in IEEE Std 488-1978 and a microcomputer system under the control of a microprocessor.

Features

- Interface capability that meets IEEE Std 488-1978
- SH1, AH1, T5/TE5, L3/LE3, SR1, RL1, PFL/PP2, DC1, DI1, CI, C2, C3, C4, C5
- Programmable data transfer rate
- Sixteen registers
  - eight read register and eight write registers
  - message transmission and reception, interface function control, and status information
- Address registers
  - detection of MTA, MLA, and MSA
  - two device addresses
- Automatic EOS message detection
- Automatic command processing and undefined command read capability
- DMA capability
- Programmable bus transceiver I/O specification
- 1 to 8MHz clock range
- TTL-compatible
- N-channel MOS
- +5V single power supply
- 40-pin plastic DIP
Names of states and messages are described using the method specified in IEEE Std. 488-1978. Their explanations are omitted. For detailed explanations, refer to the IEEE paper:

IEEE Std. 488-1978 "IEEE Standard Digital Interface for Programmable Instrumentation"

The uPD7210 operates on positive logic; IEEE Std. 488-1978 is based on negative logic. A term or state name in a logic equation equals one when the uPD7210 is in that state, and zero otherwise.

GPIB: an interface bus which meets IEEE Std. 488-1978
Command: Multiline remote interface message
Data: Multiline remote device-dependent message

Figure 1.1 Pin Configuration
1.1 General Description

The µPD7210 is designed to conform to IEEE Std 488-1978. It provides an interface between a microcomputer system and the GPIB. This µPD7210 is connected to a GPIB via non-inverting bus transceivers that meet the electrical specifications in IEEE Std 488-1978.
1.2 IEEE Std 488-1978

This standard outlines a method for simplifying a system by connecting various peripherals to the same bus (GPIB). This requires standardized control signals and data flow to and from each peripheral.

Data transfer using the μPD7210 is carried out in bit-parallel and byte-serial fashion over DIO lines. Transfer timing is controlled by three handshake lines (data byte transfer control lines). In this three-line handshake system, a byte cannot be transferred until one handshake cycle (transfer of the previous byte) is complete. This feature enhances reliable data transfer between peripherals of different speeds.
Figure 1.3 Interface Functions and Bus Configuration
CHAPTER 2
FUNCTIONAL DESCRIPTION

The µPD7210 meets the functional requirements of the following as specified in IEEE Std. 488-1978.

SR1, AR1, T5 or TE5, L3 or LE3,
SR1, RL1, PF1, DC1, DT1, C1, C2, C3, C4, C5

When the controller is active, i.e., CIO=1 (CIO=CIDS+CADS), there are two exceptions.

- When CIC·SRQ=1, the SRQ pin becomes an input and the SRQ message is not transmitted. This SRQ message is detected inside the µPD7210.

- When CIC·PPAS=1, the DIO pins (DIO1 to DIO8) become inputs and the PPR message is not transmitted. You can detect the PPR message by reading the CPT register.

Figure 2.1 IEEE 488-1978: Exception 1
2.1 Pin Description

4 RESET Reset (input, active high)

The uPD7210 goes to an idle state when this pin is high.

3 CLOCK Clock (input)

This is a reference clock input that generates the state change prohibit times ($T_1$, $T_5$, $T_7$, $T_9$) as specified in the IEEE std. 488-1978.

12-19 D0 to D7 Data Bus (input/output)

This is an 8-bit bidirectional data bus which is connected to the data bus of the microcomputer system.

8 CS Chip Select (input, active low)

This enables access to the register selected by RSO-RS2.

9 RD Read (input, active low)

This reads the contents of the read register specified by RSO-RS2 on D0-D7.

10 WR Write (input, active low)

This writes the data in D0-D7 to the write register specified by RSO-RS2.
**Register Select** (input)

This selects one read (or write) register of the eight registers during a read (or write) operation.

**Interrupt Request** (output)

This signal is active when an interrupt request is generated from one of the internal interrupt factors. You can change the active level in software.

- Active high: when $B_3=0$  
  (B3=0 on reset)
- Active low: when $B_3=1$

$B_3$: Bit 3 of auxiliary B register

**DMA Request** (output, active high)

This signal indicates that a request for DMA has been made. It goes low when a DMA acknowledge signal is input.

**DMA Acknowledge** (input, active low)

This signal connects the data bus of the microcomputer system to the data register. When this signal is low, the contents of the Data In register is output into D0-D7 with a read signal, and the data in D0-D7 is output to the Byte Out register with a write signal.

<table>
<thead>
<tr>
<th>CS</th>
<th>RD</th>
<th>WR</th>
<th>DMA ACK</th>
<th>RS0-RS2</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>any</td>
<td>read registers OR-7R</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>any</td>
<td>write to registers OW-7W</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>XXX</td>
<td>read to Data In register (OR)</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>XXX</td>
<td>write to Byte Out register (OW)</td>
</tr>
<tr>
<td>1</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>XXX</td>
<td>register is not selected</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>1</td>
<td>X</td>
<td>XXX</td>
<td>(DO-D7=Hi-Z)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>X</td>
<td>XXX</td>
<td>prohibit</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>XXX</td>
<td>(operation is not guaranteed)</td>
</tr>
</tbody>
</table>
1  T/R1  Transmit/Receive Control 1  (output)

This line is the input and output control signals of the GPIB bus transceivers.

<table>
<thead>
<tr>
<th>T/R1</th>
<th>DIO1-DIO8</th>
<th>TRAV</th>
<th>NRFD</th>
<th>NDAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>output</td>
<td>output</td>
<td>input</td>
<td>input</td>
</tr>
<tr>
<td>0</td>
<td>input</td>
<td>input</td>
<td>output</td>
<td>output</td>
</tr>
</tbody>
</table>

2,5  T/R2, T/R3  Transmit/Receive Control 2,3  (output)

The function of the T/R2 and T/R3 signals is determined by the TRMO and TRMI values of the address mode register as shown below.

<table>
<thead>
<tr>
<th>T/R2</th>
<th>T/R3</th>
<th>TRMO</th>
<th>TRMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDIOE</td>
<td>TRIG</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CIC</td>
<td>TRIG</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>CIC</td>
<td>EDIOE</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>CIC</td>
<td>PE</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

EDIOE=TDACS+SPAS+CIC.CBS
When EDIOE=1, the EDI pin is an output; when EDIOE=0, EDI is an input.

CIC=CIDS+CADS
When CIC=1, ADIN is an output and SRQ is an input. When CIC=0, ADIN is an input and SRQ is an output.

PE=CIC.PFAS
When PE=1, three-state bus transceivers are used in DIO1-DIO8 and the DAV lines. When PE=0, open-collector transceivers are used.
TRIG

When DTAS=1 or when a Trigger auxiliary command is issued, a high pulse is generated.

The following pins are connected to the corresponding lines of the GPIB through non-inverting bus transceivers.

| 28-35 | D10-D18 | Data Input/Output | (input/output) |

This is an 8-bit bidirectional data bus used for transferring remote multiline messages.

| 36 | DAV | Data Valid | (input/output) |

This is a handshake line which indicates that the data on the DIO lines is valid.

| 37 | ERRD | Not Ready for Data | (input/output) |

This is a handshake line which indicates whether the GPIB is prepared to receive messages.

| 38 | EDAC | Not Data Accepted | (input/output) |

This is a handshake line which indicates that the message has been received.

| 26 | MN | Attention | (input/output) |

This is a control line which indicates that data on the DIO lines is an interface message or a device-dependent message.

| 24 | IFD | Interface Clear | (input/output) |

This is a control signal that clears the interface function.
This control line asks the controller for service.

This control line selects remote or local control for a device.

This control line indicates the end of a transfer of multiple bytes or, with ATN, executes a parallel poll.
CHAPTER 3

INTERNAL REGISTERS

The uPD7210 contains 16 registers; eight read registers and eight write registers.

<table>
<thead>
<tr>
<th>R R R R</th>
<th>S S S S</th>
<th>Read Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>D17</td>
<td>D16</td>
</tr>
<tr>
<td>0 0 1</td>
<td>CPT</td>
<td>APT</td>
</tr>
<tr>
<td>0 1 0</td>
<td>INT</td>
<td>SRQI</td>
</tr>
<tr>
<td>0 1 1</td>
<td>S8</td>
<td>PEND</td>
</tr>
<tr>
<td>1 0 0</td>
<td>CIC</td>
<td>AYN</td>
</tr>
<tr>
<td>1 0 1</td>
<td>CPT7</td>
<td>CPT6</td>
</tr>
<tr>
<td>1 1 0</td>
<td>X</td>
<td>DTO</td>
</tr>
<tr>
<td>1 1 1</td>
<td>EOL</td>
<td>DT1</td>
</tr>
</tbody>
</table>

Figure 3.1 Read Registers

<table>
<thead>
<tr>
<th>R R R R</th>
<th>S S S S</th>
<th>Write Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>BO7</td>
<td>BO6</td>
</tr>
<tr>
<td>0 0 1</td>
<td>CPT</td>
<td>APT</td>
</tr>
<tr>
<td>0 1 0</td>
<td>0</td>
<td>SRQI</td>
</tr>
<tr>
<td>0 1 1</td>
<td>S8</td>
<td>raw</td>
</tr>
<tr>
<td>1 0 0</td>
<td>Ion</td>
<td>TRM1</td>
</tr>
<tr>
<td>1 0 1</td>
<td>CNT2</td>
<td>CNT1</td>
</tr>
<tr>
<td>1 1 0</td>
<td>ARS</td>
<td>DT</td>
</tr>
<tr>
<td>1 1 1</td>
<td>EC7</td>
<td>EC8</td>
</tr>
</tbody>
</table>

Figure 3.2 Write Registers

3.1 Data Registers

<table>
<thead>
<tr>
<th>D17</th>
<th>D16</th>
<th>D15</th>
<th>D14</th>
<th>D13</th>
<th>D12</th>
<th>D11</th>
<th>D10</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

These data registers are used for transferring commands and data between the GPIB and the microcomputer system. The Data In register holds data sent from the talker over the GPIB when the uPD7210 is designated as the listener. Data is output over the data bus with a read operation. The contents of the Data In register are held until the next eight bits of data are received.
The Byte Out register holds data or a command written to it by a write operation and sends the data or command to the GPIB. The contents of the Byte Out register are updated at the trailing edge of the write strobe.

### 3.2 Interrupt Registers

<table>
<thead>
<tr>
<th>CPT</th>
<th>APT</th>
<th>DET</th>
<th>END</th>
<th>DEC</th>
<th>ERR</th>
<th>DO</th>
<th>DI</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT</td>
<td>SRQ1</td>
<td>LOK</td>
<td>REM</td>
<td>CO</td>
<td>LOKC</td>
<td>REMC</td>
<td>ADSC</td>
</tr>
<tr>
<td>(1R) Interrupt Status Register 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2R) Interrupt Status Register 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1W) Interrupt Mask Register 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2W) Interrupt Mask Register 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The interrupt registers are composed of interrupt status bits, interrupt mask bits, and other bits not associated with interrupts.

#### 3.2.1 Interrupt Bits

There are thirteen possible interrupt conditions. Each possible interrupt condition has an interrupt status bit and an interrupt mask bit associated with it.

Table 3.1 shows the conditions under which an interrupt status bit is set or reset (bit=1=set). The interrupt mask bit enables or disables the corresponding interrupt condition. When the interrupt mask bit is set, the corresponding interrupt condition is enabled.
<table>
<thead>
<tr>
<th>Bit</th>
<th>Set Conditions</th>
<th>Reset Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPT</td>
<td>([UCG + ACG \cdot (TADS + LADS)] \cdot \text{undefined} \cdot ACDS \cdot B_0 + UDPCF \cdot SCG \cdot ACDS \cdot B_0)</td>
<td>pon + (Read INT Status 1 Reg.)* ↓</td>
</tr>
<tr>
<td>UDPCF</td>
<td>([UCG + ACG \cdot (TADS + LADS)] \cdot \text{undefined} \cdot ACDS \cdot B_0)</td>
<td>([UCG + ACG] \cdot \text{defined} + TAG + LAG \cdot ACDS + B_0 + pon)</td>
</tr>
<tr>
<td>APT</td>
<td>(ADM 1^+ \cdot ADM 0^+ \cdot (TPAS + LPAS) \cdot SCG \cdot ACDS)</td>
<td>pon + (Read INT Status 1 Reg.) ↓</td>
</tr>
<tr>
<td>DET</td>
<td>DTAS</td>
<td></td>
</tr>
<tr>
<td>END</td>
<td>LACS \cdot (EOI + EOS \cdot A_2)^+ \cdot ACDS</td>
<td></td>
</tr>
<tr>
<td>DEC</td>
<td>DCAS</td>
<td></td>
</tr>
<tr>
<td>ERR</td>
<td>SDYS \cdot DAC \cdot RFD + SIDS \cdot (Write Byte Out Reg.)^* + (SDYS \rightarrow SIDS)^*</td>
<td></td>
</tr>
<tr>
<td>DO</td>
<td>(TACS \cdot SGNS)^↑</td>
<td>(Read INT Status 1 Reg.) ↓ + TACS + SGNS</td>
</tr>
<tr>
<td>DI</td>
<td>LACS \cdot ACDS \cdot Continuous Mode</td>
<td>pon + (Read INT Status 1 Reg.) ↓ + (Finish Handshake)^* \cdot (Holdoff Mode)^* + (Read Data in Reg.)^*</td>
</tr>
<tr>
<td>SRQI</td>
<td>(CIC^* \cdot SRQ \cdot RQS \cdot DAV)^↑</td>
<td></td>
</tr>
<tr>
<td>LOKC</td>
<td>LOK^* \uparrow + LOK↓</td>
<td>pon + (Read INT Status 2 Reg.)* ↓</td>
</tr>
<tr>
<td>REMC</td>
<td>REM^* \uparrow + REM↓</td>
<td></td>
</tr>
<tr>
<td>ADSC</td>
<td>((T^* + TA^+ + LA^+ + LA + CIC + CIC + + MJMN^* + MJMN)^↓ \cdot \text{ion} + \text{ion})</td>
<td></td>
</tr>
<tr>
<td>CO</td>
<td>(CACS \cdot SGNS)^↑</td>
<td>(Read INT Status 2 Reg.) ↓ + CACS + SGNS</td>
</tr>
</tbody>
</table>

**Table 3.1 Interrupt Status Bits**

**Notes:**

- \(A_2^*\): auxiliary register A
- \(B_2^*\): auxiliary register B
- \(\text{ADMD},1\): address mode register bits
- \(\text{LOK, REM}\): interrupt status register 2 bits
- \(\text{TA}, \text{LA}, \text{CIC}, \text{MJMN}\): address status register bits
- \(\text{finish handshake})\): Finish Handshake auxiliary command issued
- \(\text{holdoff mode})\): RFD holdoff state
- \(\text{SDYS} \rightarrow \text{SIDS}\): transition from SDYS to SIDS

The INT bit (interrupt status register 2) is the logical OR of the enabled interrupt status bits. When any unmasked interrupt status bit is set, the INT bit=1. This makes the INT pin active.

When the CPU receives an interrupt, it can tell which condition triggered the interrupt by reading the interrupt status register. All the bits of the interrupt status register are cleared after a read. If an interrupt occurs during a read, the interrupt request is held until after the register is cleared, then it is placed in the register.
CPT  Command Pass Through

When \( B_0 = 1 \), this bit indicates that an undefined command has been received over the GPIB or that a secondary command has been received just after an undefined command.

When this bit is set, the DAC message is held and the handshake stops until the Valid auxiliary command is issued. The undefined command can be read from the command pass through register.

The UDPAD also indicates that an undefined primary command has been received.

APT  Address Pass Through

This indicates that the secondary address (which the CPU is required to check in address mode 3) has been received.

When this bit is set, the DAC message is held and the handshake stops until either the Valid or Non-valid auxiliary command is issued. The secondary address can be read from the command pass through register.

DET  Device Trigger

This indicates that the device has been in DTAS. A high pulse is output when T/R3 is used as the TRIG pin.

END  

This indicates that the transfer of a data block is complete. This bit is set when either the END message (EOI) or EOS message (when the contents of the EOS register and the Data In register are the same) is received.

DEC  Device Clear

This indicates that the device is in DCAS.
ERR Error

This bit indicates that the contents of the Byte Out register have been lost. This bit is set when data is sent over the GPIB without a specified listener or when a byte is written to the Byte Out register during SIDS or during the SDYS->SIDS transition.

DO, DI Data Out, Data In

The DO bit indicates a data write request to the Byte Out register.

The DI bit indicates that a data byte has been written to the Data In register from the GPIB and asks the CPU to read the Data In register.

In continuous mode, the DI bit is not set by a write to the Data In register.

When you are not using DMA, you can use the DMAREQ pin as the DO/DI interrupt pin. This causes the DMA0 and DMA1 bits to function as mask bits.

SRQI Service Request Input

This indicates that an SRQ message has been received while the controller is active (CIC=1).

When a service request comes from several devices, the RQS message is detected on the DIO line and the SRQI bit is set again.

LOKC, REMC Lockout Change, Remote Change

LOKC indicates a change in the value of the LOK bit (WLWL+LMWL). REMC indicates a change in the value of the REM bit (RMLS+RLLS).

ADSC Address Status Change

This indicates that a change occurred in one of the four bits (TA, LA, CIC, MNN) of the address status register. You can find the values of these bits by reading the address status register.
This indicates a request that a command be written to the Byte Out register so that it can be transmitted to the GPIB.

### 3.2.2 Non-interrupt Bits

**LOK, REM**  
Lockout, Remote

These indicate the status of the RL interface functions. The LOK bit indicates that the device is in LWLS or RWLS. The REM bit indicates that the device is in REMS or RWLS.

**DMA0, DMAI**  
DMA Output, DMA Input

These bits enable and disable DMA transfers between memory and data registers.

When DMA0=1 and the uPD7210 is in data transmission enable mode, a DMA request that asks for a data byte to be written to the byte out register is generated. When DMAI=1, a DMA request that asks for data to be written from the GPIB to the Data In register is generated.

### 3.3 Serial Poll Registers

<table>
<thead>
<tr>
<th>S8</th>
<th>PEND</th>
<th>S6</th>
<th>S5</th>
<th>S4</th>
<th>S3</th>
<th>S2</th>
<th>S1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(3R) Serial Poll Status Register

<table>
<thead>
<tr>
<th>S8</th>
<th>rsv</th>
<th>S6</th>
<th>S5</th>
<th>S4</th>
<th>S3</th>
<th>S2</th>
<th>S1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(3W) Serial Poll Mode Register

The serial poll mode register holds the status byte (S1-S6, S8; sent over the GPIB) and the local message, rsv.

When the CPU sets rsv=1 (rsv message is issued), the state of the SR interface function becomes SRQS when the controller is not serial polling the device.

When the polling of the controller puts the T/TE interface in SPAS, the contents of the serial poll mode register are sent over the DIO lines as STB and RQS messages. The rsv bit is cleared when the SR interface function is in APRS.
You can read STB in the serial poll mode register from the serial poll status register. The PEND bit is set when RSV=1 and cleared when NRS•RSV=1. You can confirm that a request was accepted and that the STB bit was transmitted by reading the status of the PEND bit.

You can clear all the bits of the serial poll mode register with a reset pulse or with the Chip Reset auxiliary command.

### 3.4 Address Mode Status Registers

<table>
<thead>
<tr>
<th>CIC</th>
<th>ATN</th>
<th>SPRS</th>
<th>LAPS</th>
<th>TPAS</th>
<th>LA</th>
<th>TA</th>
<th>MJMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ton</td>
<td>lon</td>
<td>TRM1</td>
<td>TRMD</td>
<td>0</td>
<td>0</td>
<td>ADM</td>
<td>ADM</td>
</tr>
</tbody>
</table>

(4R) Address Status Register  
(4R) Address Mode Register

The address mode register selects the functions of the T/R2 and T/R3 pins and selects the address mode.

#### 3.4.1 Selecting T/R2 and T/R3 Pin Function

The T/R2 and T/R3 pin functions are selected as follows:

<table>
<thead>
<tr>
<th>TRM1</th>
<th>TRM0</th>
<th>T/R2</th>
<th>T/R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>EIOE</td>
<td>TRIG</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>CIC</td>
<td>TRIG</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>CIC</td>
<td>EIOE</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>CIC</td>
<td>PE</td>
</tr>
</tbody>
</table>

Table 3.2 Pin Function Select
3.4.2 Selecting Address Mode

<table>
<thead>
<tr>
<th>tone</th>
<th>lorn</th>
<th>ADM1</th>
<th>ADM0</th>
<th>Address Mode</th>
<th>Contents of Adr. reg. 0</th>
<th>Contents of Adr. reg. 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>talk only</td>
<td>not used</td>
<td>not used</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>listen only</td>
<td>not used</td>
<td>not used</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Address mode 1</td>
<td>major talk or major listen addr.</td>
<td>minor talk or minor listen address</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Address mode 2</td>
<td>primary addr. talk or listen</td>
<td>secondary address talk or listen</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Address mode 3</td>
<td>primary addr. major talk or major listen</td>
<td>primary address minor talk or minor listen</td>
</tr>
</tbody>
</table>

All other combinations are prohibited.

Table 3.3 Address Mode Select

The µPD7210 automatically detects two types of addresses. These are held in address registers 0 and 1.

□ Address Mode 1

Address mode 1 includes two types of device addresses: major and minor. MTA or MLA reception is indicated when either address equals the received address. The interface function is either T or L.

□ Address Mode 2

Address register 0 holds the primary address and address register 1 holds the secondary address. The interface function is either TE or LE.

□ Address Mode 3

Address mode 3 provides major and minor primary addresses. The CPU must identify the secondary address by reading the command pass through register. The interface function is either TE or LE.

3-19
Talk Only and Listen Only Modes

Address identification is not necessary in these modes. No address register is used.

<table>
<thead>
<tr>
<th>Bit Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATN</td>
<td>ATN</td>
</tr>
<tr>
<td>LPAS</td>
<td>LPAS</td>
</tr>
<tr>
<td>TPAS</td>
<td>TPAS</td>
</tr>
<tr>
<td>CIC</td>
<td>CIDS+CAIS</td>
</tr>
<tr>
<td>LA</td>
<td>LACS+LDAS=LIDS</td>
</tr>
<tr>
<td>TA</td>
<td>TACS+SPAS+TADS=TIDS</td>
</tr>
<tr>
<td>MJMN</td>
<td>Setup: receipt of minor talk or minor listen address; reset: receipt of major talk or major listen address; ADMO=0 or pon=1 or IPC=1</td>
</tr>
<tr>
<td>SPMS</td>
<td>SPMS</td>
</tr>
</tbody>
</table>

Table 3.4 Address Status Register Bits

The ATN bit confirms that the device has entered CSBS after the Go To Standby auxiliary command has been issued.

LPAS and TPAS determine whether the received secondary address is the talk or listen address after an APT interrupt.

CIC, LA, TA, and MJMN are used when you must know their values when an ADSC interrupt is generated.

3.5 Address Registers

(6R) Address Register 0

(7R) Address Register 1

(6W) Address Register 0/1

3-20
You set device address by writing the address to address register 0/1. Each bit of address register 0/1 is as follows:

<table>
<thead>
<tr>
<th>Bit Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARS</td>
<td>Address register 0 selects the low-order bits are written (AD1 to AD5)</td>
</tr>
<tr>
<td></td>
<td>Address register 1</td>
</tr>
<tr>
<td>DT</td>
<td>permitted permits or prohibits the set address (AD1 to AD5) detected as a talk address. This bit corresponds to DT1 or DT0 of the address registers.</td>
</tr>
<tr>
<td></td>
<td>prohibited</td>
</tr>
<tr>
<td>DL</td>
<td>permitted permits or prohibits the set address (AD1 to AD5) detected as a listen address. This bit corresponds to DL1 or DL0 of the address registers.</td>
</tr>
<tr>
<td></td>
<td>prohibited</td>
</tr>
<tr>
<td>AD1-AD5</td>
<td>these bits indicate device and correspond to AD5-0 to AD1-0 and AD5-1 to AD1-1.</td>
</tr>
</tbody>
</table>

Table 3.5 Address Register 0/1 Bits

For example, when the following codes are written into address register 0/1 in address mode 1:

0 0 X A A A A A AAAAA: major talk address
1 0 X B B B B B BBBBB: minor talk address

the device has both the major and minor talk addresses. The uPD7210 operates as if the MTA has been received when the talk address of either AAAAA or BBBBB is received.

You can read the address (AD1-AD5) and the DT and DL bits written in address register 0/1 in either address register 0 or 1, according to the value of the ARS bit. However, the value of bit 7 of address register 0 is unknown. Bit 7 (EOI) indicates the value of the EOI line latched when a data byte is received.
3.6 Command Pass Through Register

![Command Pass Through Register]

The CPU reads the data on the DIO lines through the command pass through register. You can read the following three types of registers from the CPT register:

- **CPT=1, B₀=1**
  This indicates an undefined command (command not defined in IEEE Std. 488-1978) or a secondary command received after an undefined primary command.

- **APT=1, address mode 3**
  This indicates a secondary address.

- **After a parallel poll**
  This indicates the PPR message in the parallel poll. If the PPRn message to be output by the uPD7210 is true during the parallel poll execution in PPAS, the PPR message is latched to the CPT register instead of being output to the DIO line.

In the first two cases, the CPT contains the data on the DIO line. In the last case, the PPR message is latched into the CPT register when CPPS=1 until CIDS=1 or a command byte is sent over the GPIB.

3.7 End of String (EOS) Register

![End of String Register]

This register holds the seven- or eight-bit EOS message byte used by the GPIB to detect the end of a data block transfer. The length of the EOS byte is selected by A₄ (bit 4 of auxiliary register A).
When data is being received with $A_2$ set, end of string is detected when the received data and the contents of EOS are equal. This causes the END bit to be set.

When data is being transmitted with $A_3$ set, the END message is sent at the same time as the transmitted data if the transmitted data and EOS register are equal.

### 3.8 Auxiliary Mode Register

<table>
<thead>
<tr>
<th>CNT2</th>
<th>CNT1</th>
<th>CNT0</th>
<th>COM4</th>
<th>COM3</th>
<th>COM2</th>
<th>COM1</th>
<th>COM0</th>
<th>(SW) Auxiliary mode register</th>
</tr>
</thead>
</table>

A write to this register generates one of the following operations according to the value of the CNT bits (CNT0-2):

- □ a write to the auxiliary register
- □ an auxiliary command is issued
- □ state change prohibit time is set
- □ a write to the parallel poll register

<table>
<thead>
<tr>
<th>CNT 210</th>
<th>COM 43210</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>C0 C3 C2 C1 C0</td>
<td>issues an auxiliary command specified by C0-C4</td>
</tr>
<tr>
<td>001</td>
<td>0 F3 F2 F1 F0</td>
<td>specifies the reference clock frequency and determines T1, T5, T7, and T9</td>
</tr>
<tr>
<td>011</td>
<td>US P3 P2 P1</td>
<td>writes to parallel poll register</td>
</tr>
<tr>
<td>100</td>
<td>A4 A3 A2 A1 A0</td>
<td>writes to auxiliary register A</td>
</tr>
<tr>
<td>101</td>
<td>B4 B3 B2 B1 B0</td>
<td>writes to auxiliary register B</td>
</tr>
<tr>
<td>110</td>
<td>0 0 0 E1 E0</td>
<td>writes to auxiliary register E</td>
</tr>
</tbody>
</table>

**Table 3.6 Auxiliary Register Bits**
### 3.8.1 Auxiliary Commands

You issue the following auxiliary commands by writing $000C_4C_3C_2C_1C_0$ to the auxiliary mode register.

<table>
<thead>
<tr>
<th>$C_4$</th>
<th>$C_3$</th>
<th>$C_2$</th>
<th>$C_1$</th>
<th>$C_0$</th>
<th>Immediate Execute pon</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

A local message "pon" is generated that places the following interface functions into their idle states:
- SIDS, AIDS, TIDS, SPIS, TPIS, LIDS, LPIS, MPRS, LOCs, PPIS, PUCs, CIDS, SRIS, SII S

If you issue this command while a pon local message is already active (by either an external reset pulse or the Chip Reset auxiliary command) the pon local message becomes false.

<table>
<thead>
<tr>
<th>$C_4$</th>
<th>$C_3$</th>
<th>$C_2$</th>
<th>$C_1$</th>
<th>$C_0$</th>
<th>Chip Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

This command performs the same function as an external reset pulse. The uPD7210 is reset to the following state:

- local message pon is set and the interface functions are placed in their idle states;
- all bits of the serial poll mode register are cleared;
- EOI bit is cleared;
- all bits of the auxiliary A, B, and E registers are cleared;
- the Parallel Poll flag and RSC local message are cleared;
- sets $N_F=8$ ($F_3=1$, $F_2=F_1=F_0=0$);
- clears the TRMO bit and the TRML bit;

<table>
<thead>
<tr>
<th>$C_4$</th>
<th>$C_3$</th>
<th>$C_2$</th>
<th>$C_1$</th>
<th>$C_0$</th>
<th>Finish Handshake</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

This command ends the handshake by releasing the RFD message transmission from the holdoff state.

3-24
00100 Trigger

This command generates a high pulse in the TRIG pin (T/R3 pin when TRM1=0). This auxiliary command performs the same function as if the DET bit (interrupt status register 1) were set. The DET bit is not set by this command.

0X101 Return to Local

When C3=0, this command generates the local message "rtl" in the form of pulses. If rtl is already set, this command clears it.

When C3=1, this command sets the local message rtl.

00110 Send BOI

This command sends the END message with the next data byte. It is valid only for TACS.

00111 Non-valid

This command releases the DAC message held off by the address pass through. The uPD7210 is allowed to operate as if an OSA message has been received.

01111 Valid

This command releases the DAC message held off by address pass through and functions as if an NSA message had been received. The DAC message is released at the time of command pass through. DAC is also released if DCAS or DTAS is in holdoff state.

0X001 Set Parallel Poll Flag

This command sets the Parallel Poll Flag to the value of C3. The value of the Parallel Poll flag is used as the local message ist when B4=0; the value of SRQS is used as the ist when B4=1.
Go to Standby

This command sets the local message gts at the time of CACS. gts is cleared when CACS goes low.

Take Control Asynchronously

This generates the local message tca in the form of pulses.

Take Control Synchronously

This command sets the local message tcs. tcs is effective only when CSBS+CSWS=1. tcs is cleared at the leading edge of CACS.

Take Control Synchronously on End

This command sets the local message tcs when the data block transfer end (END=1) is generated at CSBS. tcs is cleared at the leading edge of CACS.

Listen

This command generates the local message ltn in the form of a pulse.

Listen in Continuous Mode

This command generates the local message ltn in the form of a pulse and places the device in continuous mode.

In continuous mode, the local message rdv is issued when ANRS is initiated unless data block transfer end is detected (END=1). When the end is detected, the device is placed in the RFD holdoff state, preventing generation of the rdv message. In continuous mode, the DI bit is not set when a data byte is received. The continuous mode caused by this command is released when the Listen auxiliary command is issued or LIDS is initiated.
11100 Local Unlisten

This command generates the local message lUN in the form of a pulse.

11101 Execute Parallel Poll

This command sets the local message rPP. rPP is cleared when CPRS+CIDS=1. The transition of the C interface function is not guaranteed if the local messages rPP and gTS are issued simultaneously when CACS+STRS+SDNS=1.

1X110 Set/Clear IPC

This command generates the local message rSC and sets IPC to the value of \( C_3 \). In order to meet IEEE Std. 488-1978, you must not issue the Clear IPC command until IPC has been held true for at least 100us.

\[
C_3=1=IPC \quad C_3=0=\overline{IPC}
\]

1X111 Set/Clear REN

This command generates the local message rSC and sets REN to the value of \( C_3 \). In order to meet IEEE Std. 488-1978, you must not issue the Set REN command until REN has been held false for at least 100us.

\[
C_3=1=REN \quad C_3=0=\overline{REN}
\]

10100 Disable System Control

This command clears the local message rSC.

3.8.2 Internal Counter

The internal counter generates the state change prohibit times \((T_1, T_6, T_7, T_9)\) specified in IEEE Std. 488-1978.

\[
T_1 \text{ (low speed)}=T_6=T_7=T_9=2N_f+T_{sync}/fc
\]
\( T_1 \) (high speed) = \( N_F + t_{\text{sync}} \frac{1}{2f_c} \)

Where \( N_F \) = integer represented by binary \( F_0-F_3 \). \( 1 \leq N_F \leq 8 \)

\( f_c = \) reference clock frequency (clock input)

\( 0 \leq t_{\text{sync}} \leq \max \) (reference clock high or low period)

\( T_1 \) (high speed) is used for all bytes following the first byte sent after each false transition of AIN if \( B_2=1 \). You should use this \( T_1 \) if you are using three-state bus drivers on the DIO, DAV, and EOI lines. In other cases, use \( T_1 \) (low speed).

When \( N_F (\text{MHz}) = f_c \), then:

\( T_1 \) (low speed) = \( T_6 = T_7 = T_9 = 2\mu s + t_{\text{sync}} \)

\( T_1 \) (high speed) = \( 500\text{ns} + t_{\text{sync}} \)

When \( N_F (\text{MHz}) < f_c \), IEEE Std. 488-1978 is not satisfied.

\( t_{\text{sync}} \) is a synchronization error greater than zero but less than the larger of the reference clock high and low.

\( 0 \leq t_{\text{sync}} \leq \max \) reference clock high or low period

For a 50\% duty clock:

\( 0 \leq t_{\text{sync}} \leq 1/2 \) reference clock period

### 3.8.3 Auxiliary Register A

You can write to auxiliary register A by writing \( 100A_4 A_3 A_2 A_1 A_0 \) to the auxiliary mode register. The contents of auxiliary register A control the messages (holdoff, EOS/END) associated with data transfer.

<table>
<thead>
<tr>
<th>A_1</th>
<th>A_0</th>
<th>Data Receiving Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>normal handshake mode</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>RDF Holdoff on All data mode</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RDF Holdoff on End mode</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Continuous mode</td>
</tr>
</tbody>
</table>

Table 3.7 Data Receiving Modes
In Normal handshake mode, the local message *rdy* is generated when data is received. When the received data is read from the Data In register, *rdy* is generated in ANRS. This causes the RFD message to be transmitted and the handshake continues.

In RFD Holdoff on All Data mode, RFD is not sent true after data is received until the microprocessor issues the Finish Handshake auxiliary command. Unlike normal handshake mode, this mode does not generate the *rdy* message even if the received data is read through the Data In register (that is, the RFD message is not generated).

In RFD Holdoff on End mode, operation is the same as the previous mode when the end of the data block (EOS or END message) is detected. Handshake holdoff is released by the Finish Handshake auxiliary command.

In continuous mode, the *rdy* message is generated when in ANRS until the end of the data block is detected. A holdoff is generated at the end of a data block. The Finish Handshake command must be issued to release the holdoff. This mode is useful for monitoring the data block transfer without data reception. In this mode, the DI bit is not set by the reception of a data byte.

<table>
<thead>
<tr>
<th>Bit Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>A_2</td>
<td>0 prohibit  \ 1 permit</td>
</tr>
<tr>
<td></td>
<td>permits or prohibits setting the END bit at reception of the EOS message</td>
</tr>
<tr>
<td>A_3</td>
<td>0 prohibit  \ 1 permit</td>
</tr>
<tr>
<td></td>
<td>permits or prohibits automatic transmission of the END message at the same time as the EOS message in TACS</td>
</tr>
<tr>
<td>A_4</td>
<td>0 7-bit EOS  \ 1 8-bit EOS</td>
</tr>
<tr>
<td></td>
<td>selects seven or eight bits as the valid length of the EOS message</td>
</tr>
</tbody>
</table>

Table 3.8 Functions of Auxiliary Register A
3.8.4 Auxiliary Register B

You can write to auxiliary register B by writing 101B\textsubscript{4}B\textsubscript{3}B\textsubscript{2}B\textsubscript{1}B\textsubscript{0} to the auxiliary mode register.

<table>
<thead>
<tr>
<th>Bit Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>B\textsubscript{0}</td>
<td>1 permit permits or prohibits the detection of undefined commands; permits or prohibits the setting of the CPT bits on receipt of an undefined command</td>
</tr>
<tr>
<td></td>
<td>0 prohibit</td>
</tr>
<tr>
<td>B\textsubscript{1}</td>
<td>1 permit permits or prohibits the transmission of the END message in SRI</td>
</tr>
<tr>
<td></td>
<td>0 prohibit</td>
</tr>
<tr>
<td>B\textsubscript{2}</td>
<td>1 T\textsubscript{1} (high spd) sets T\textsubscript{1} (high speed) as T\textsubscript{1} of handshake after transmission of second byte following data transmission</td>
</tr>
<tr>
<td></td>
<td>0 T\textsubscript{1} (low spd) sets T\textsubscript{1} (low speed) as T\textsubscript{1} in all cases</td>
</tr>
<tr>
<td>B\textsubscript{3}</td>
<td>1 INT specifies the active level of the INT pin</td>
</tr>
<tr>
<td></td>
<td>0 INT</td>
</tr>
<tr>
<td>B\textsubscript{4}</td>
<td>1 ist=SRQS SRQS indicates the value of the ist local message (the Parallel Poll flag is ignored) SQR\textsuperscript{S}=ist=1; SQR\textsuperscript{S}=ist=0</td>
</tr>
<tr>
<td></td>
<td>0 ist=Parallel Poll flag the value of the Parallel Poll flag is taken as the ist local message</td>
</tr>
</tbody>
</table>

Table 3.9 Functions of Auxiliary Register B

3.8.5 Auxiliary Register E

You can write to auxiliary register E by writing 110000E\textsubscript{1}E\textsubscript{0} to the auxiliary mode register.

<table>
<thead>
<tr>
<th>Bit Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>E\textsubscript{0}</td>
<td>1 enables enables or disables DAC holdoff by initiating DCAS</td>
</tr>
<tr>
<td></td>
<td>0 disables</td>
</tr>
<tr>
<td>E\textsubscript{1}</td>
<td>1 enables enables or disables DAC holdoff by initiating DTAS</td>
</tr>
<tr>
<td></td>
<td>0 disables</td>
</tr>
</tbody>
</table>

Table 3.10 Functions of Auxiliary Register E
3.8.6 Parallel Poll Register

You can write to the parallel poll register by writing 011USP₂P₁ to the auxiliary mode register.

When you use the subset PP₁ as the PP interface function, you should not write to this register. The parallel poll response (PRₚₑ) is automatically sent out according to the PPE message issued by the GPIB controller. For example, when the values of S and ist are equal, the PRₚₑ message is sent out true according to the specification of P₃P₂P₁ (=N-1).

When you use the subset PP₂, you must write to this register in advance. The U bit implies the local message $\overline{1Pₑ}$. When U=0, S and P₃P₂P₁ mean the same as the bit of the same name in the PPE message and the write operation is the same as the receipt of the PPE message. When U=1, S and P₃P₂P₁ bits do not carry any meaning, but they should be reset to zero.
CHAPTER 4
USING THE μPD7210

4.1 Transmitting Commands

A command is transmitted by writing the code to the Byte Out register when a request for command transmission is received (CO=1). This is repeated when you send several command bytes.

4.2 Processing Undefined Commands

When B₀=1, the DAC message is held false and the CPT bit is set when an undefined command is received. The CPU reads the undefined code via the CPT register. The handshake that stopped is completed when the Valid auxiliary command is issued.

When B₀=0, the handshake is completed, just as when a defined command is received, and the CPT bit remains 0. The received code is ignored.

A secondary command received immediately after an undefined primary command is handled as an undefined command.

4.3 Processing Address Pass Through

The APT bit is set when the secondary address is received. This is the case where a secondary command is received in address mode 3 and LPAS+TPAS=1.

- Address Mode 3 (ton=lon=0 and AIM1=AIM0=1)
  In this mode, the TE and LE interface functions are used as the talker and the listener, respectively. Address register 0 holds the major primary address and address register 1 holds the minor primary address.

- LPAS+TPAS=1
  This condition is satisfied when either the MTA (My Talk Address) or the MLA (My Listen Address) has been received.
When the APT bit is set, the handshake stops with the DAC message held false just as the CPT bit is set. The CPU must then perform the following:

- Determine whether the secondary command just received is a listen, talk, major, or minor address by reading the LPAS, TDAS, and MUMN bits of the address status register.

- Determine whether the secondary command read through the CPT register is my address. If it is my address, the Valid auxiliary command is issued. If it is not my address, the Non-valid auxiliary command is issued.

When the Valid auxiliary command is received, the uPD7210 assumes that the MSA (My Secondary Address) message has been received. When this command is issued, LADS=TDIS=1 if LPAS=1 or TDAS=LIDS=1 if TPAS=1, the DAC message is sent true, and the handshake is finished.

When the Non-valid auxiliary command is received, the uPD7210 assumes that the OSA (Other Secondary Address) message has been received. When this command is issued, TDIS=1 if TPAS=1, the DAC message is sent true, and the handshake is finished.

4.4 Beginning Data Transfer

After specifying the talker and the listener, issue the Go To Standby auxiliary command. Data transfer begins when the last command written to the Byte Out register has been transmitted.

4.5 Transmitting Data

When a data request is received (DO=1), one byte of data is transmitted by a write to the Byte Out register. This process is repeated to send several bytes. The DO bit is cleared when read or when a write is made to the Byte Out register.

You can use DMA for sending or receiving data. When a request to send data is received and DMAO=1, or when a device asks to received data and DMAI=1, a DMA request is generated (DMAREQ=1).
4.6 Receiving Data

When the CPU receives a data receive request, the contents of the Data In register are read. Data is received in the four modes below.

4.6.1 Normal Handshake Mode \((A_0=A_1=0)\)

When the device receives data as a listener, a data receive request \((DI=1)\) is made and the RFD message is sent false. As the receive data is read from the Data In register, the RFD message is sent true, informing the talker that the listener is read for the next data.

4.6.2 RFD Holdoff on All Data Mode \((A_0=1, A_1=0)\)

When the device receives data as a listener, a data receive request \((DI=1)\) is made and the RFD message is sent false, just as in normal handshake mode. However, even if the CPU reads the received data from the Data In register, the RFD message is held false until the Finish Handshake auxiliary command is issued. While the RFD message is being held false, the next data is not received and the contents of the Data In register are not updated. Therefore, the CPU can read the same data several times, if desired.

4.6.3 RFD Holdoff on End Mode \((A_0=0, A_1=1)\)

In this mode, the RFD message is sent and held false when the data is received with the END bit set. Issue the Finish Handshake auxiliary command to send the RFD message true. The function is the same as in normal handshake mode unless the END bit is set.
4.6.4 Continuous Mode \((A_0=A_1=1)\)

In this mode, the RFD message is returned true in response to the DAV message unless the END bit is set. When the END bit is set, operation proceeds as in RFD Holdoff on End mode. This mode is used for data block end detection or interrupt synchronized with the handshake by the tcs local message. When the Listen with Continuous Mode auxiliary command is issued at CACS, operation is the same as in this mode.

4.7 Completing Data Block Transfer

In accordance with IEEE Std. 488-1978, the following two methods are provided for detecting the end of the data block.

4.7.1 Placing EOS Byte After Data Block

The EOS code may be user-defined, but it is not possible to identify the EOS message if a code appearing in the data byte is used. You may not use it if the data byte is a full eight bits. In order to use ASCII code, use the LF code as EOS and the others as data bytes.

4.7.2 Using the EOI Line

In this method, the END message \((EOI=1)\) is sent out with the last byte of the data block.

4.7.3 Transmission of the EOS Message

This transmission works the same as transmission of a data byte. When DO=1, the EOS message can be transmitted by writing the EOS code to the Byte Out register.
4.7.4 Transmission of the END Message

After the Send EDI auxiliary command is issued to the μPD7210, the END message is transmitted with the data byte when the next data byte is written to the Byte Out register.

When \( A_3 = 1 \) (Output EDI on EOS Sent), the END message is sent when the EOS message is sent (based on the contents of the Byte Out register, the EOS register, and the value of \( A_4 \)).

4.7.5 Detecting the EOS Message

When \( A_2 = 1 \) (End on EOS Received), the END bit is set when the EOS message is received. Receipt of the EOS message is determined by the contents of the Data In register, the EOS register, and the value of \( A_4 \).

When the STB (Status Byte) code in the serial poll register and the EOS code agree, this is not evidence of receipt of the EOS message.

4.7.6 Detecting the END Message

The END bit is set when the END message is received in LACS. You can identify the END or EOS message because the value of the EDI line is latched in the EDI bit when data is received and data itself is latched in the Data In register.

4.8 Discontinuing Data Transfer

There are three methods of discontinuing data transfer, as described below.

4.8.1 Using the Take Control Asynchronously Command

When the Take Control Asynchronously command is issued, AVIN is set equal to one and the data transfer stops. You must be careful when using this method to stop data transfer because data on the DIO line might be taken as a command.
4.8.2 Using the Take Control Synchronously Command

When the Take Control Synchronously command is issued, ATN is set equal to one and the data transfer stops at the end of the handshake (ANRS). You must specify the controller as the listener before using this method.

To specify the controller as listener, issue the Listen or Listen with Continuous Mode auxiliary command before issuing the Go to Standby auxiliary command. When you use the Listen command, the listener functions in the mode specified by the $A_0$ and $A_1$ bits. When you use the Listen with Continuous Mode command, the listener functions in continuous mode, whatever the values of $A_0$ and $A_1$.

4.8.3 Using the Take Control Synchronously on End Command

When this command is issued, ATN is set equal to one at the end of the current data transfer and transfer stops.

4.9 Serial Polling

To request service, you must confirm that there is no pending service request (PEND=0). Write the STB into the serial poll mode register with the local message rsv=1. If the device is not in SPAS, the SRQ message is sent true as soon as the rsv message is set. If the device is in SPAS, the SRQ message remains false until the serial polling is complete (SPAS=0). The PEND bit indicates whether a service request is accepted or left pending. It is set when rsv=1 and cleared when the STB is read out by the controller-in-charge (SPAS=0), or when the local message rsv is cleared before SPAS=1.

The STB set to the serial poll mode register is sent out when the STB is asked to send. The STB is sent only once even if the controller does not assert ATN after the first transfer. The END message is sent out if $B_1=1$. 
4.10 Parallel Polling

When the Execute Parallel Poll auxiliary command is issued and the local message rpp is set to 1, the parallel poll is executed as soon as the C interface function is placed in the proper state (CAWS or CACS). The PPR (Parallel Poll Response) is automatically taken in the CPT register and rpp is cleared to 0. The CPU knows that this operation is complete based on the condition of C0=1. The PPR can be obtained by reading the contents of the CPT register, which are held until a command is transmitted or the controller becomes inactive.

4.11 Parallel Poll Protocol

Before a parallel poll is executed, you must specify to which line of DIO1 to DIO8 the one-bit status (ist:individual status) should be output and which polarity should be used. The following two methods are provided for this.

- **Remote Configuration (PPI)**
  In this method, the specifications are made by either PPE or PPD messages sent from the controller.

- **Local Configuration (PP2)**
  In this method, the specifications are made from the device.

Specifications by the CPU are not required in the Remote Configuration. In the Local Configuration, you must write the following values to the auxiliary mode register.

```
0 1 1 U S P3 P2 P1
```

status bit output line (DIO1 to DIO8)
status bit polarity
S=1: in phase
S=0: in reverse phase
response to parallel poll U=1: no response
U=0: response made
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>1</th>
<th>U</th>
<th>S</th>
<th>P₃</th>
<th>P₂</th>
<th>P₁</th>
</tr>
</thead>
</table>

Status bit output line (D1O1 to D108)

Status bit polarity:
- S = 1: In phase
- S = 0: In reverse phase

{ U = 1: No response to parallel poll
  U = 0: Response to parallel poll

**Figure 4.1 Local Configuration for Parallel Poll**
CHAPTER 5

GPIB-INTERFACE USING uPD7210

This chapter describes how to set up a GPIB-Interface using the uPD7210. Complete hardware schematics as well as a software example will be given, to show you how easy an implementation of a GPIB-Interface with the uPD7210 may be.

5.1 Hardware

Figure 1 shows one possible way to implement a minimal uPD8085 system including a GPIB-Interface. As the uPD7210 is directly bus-compatible with the uPD8080,8085,8088 ... processor series, no additional hardware is needed to connect the uPD7210 to the uPD8085.

Figure 1. Minimum 8085 System using uPD7210
The uPD7210 is attached to the GPIB-Bus via appropriate bus drivers, which meet the IEEE-488 bus specifications (Figures 2,3). The uPD7210 is able to operate with almost any IEEE 488 bus drivers, which are controlled by the T/R1, T/R2 and T/R3 lines. These lines are programmed by software depending on the type of the bus drivers used.

Note:
In the case of low-speed data transfer (B2=0), the T/R3 pin can be used as a TRIG output. The PE input of the SN75160 should be set to "0".

Figure 2. Using SN75160 and SN75161 Bus drivers (T/R mode 3: TRM1=TRM0=1)
Note:
In this example, no high-speed data transfer can be made because the bus transceiver are open collector type (set $B_2=0$).

Figure 3. Using MC3448A Bus drivers (T/R mode 2: $TRM1=1, TRM0=0$)
5.2 SOFTWARE

In this software example the uPD7210 operates as CONTROLLER/TALKER on the GPIB-Bus. After specifying a LISTENER the uPD7210 will send a short stream of data to the GPIB-Bus. Figure 4 shows the command and data sequence put on the GPIB-Bus by the uPD7210.

![Diagram of GPIB-Bus communication](image)

Figure 4. Data Transmission with the GPIB-Bus

During initialisation the IFC (interface clear) line is activated to reset all devices attached to the bus. Before starting the data transfer, a TALKER and at least one LISTENER has to be specified by the CONTROLLER. For this reason the CONTROLLER places the command UNLISTEN, the TALKER-ADDRESS (his own one) and a LISTENER-ADDRESS on the bus. As soon as the CONTROLLER releases the ATN line (end of command mode), the TALKER starts the transmission of data (data mode). With the last character transmitted (carriage return) the TALKER activates the EOI (end or identify) line indicating that the data transfer is finished. Now the CONTROLLER again takes the bus control to initiate a new data transfer.

The software listing shows the implementation of the procedure discussed above. This listing is commented extensively and should be self-explanatory.
In this example the uPD7210 works as
CONTROLLER/TALKER and sends the data stream
'0' '1' '2' 'CR' to a LISTENER.

**The following code snippet is a software example for the uPD7210 microcontroller.**

```assembly
; MAIN Routine
MAIN:    JMP XMAIN

; INIT Routine
INIT:    JMP XINIT

; SEND Routine
SEND:    JMP XSEND

; Chip reset
XINIT:   MVI A,02H ;Chip reset
OUT AUXM

; disable interrupt1
MVI A,00H
OUT INTM1

; disable interrupt2
OUT INTM2

; Address mode 1
MVI A,31H ;Address mode 1
OUT ADRM ;T/R mode 3 : 75160/1 drivers

; My address = 0
MVI A,00H ;my address=0
OUT ADR01 ;<address 0 register>

; disable address 1 register
MVI A,0E0H ;disable address 1 register
```

*Example:*

- **INIT**: MVI A,02H ;Chip reset
  - OUT AUXM
- **SEND**: MVI A,00H ;disable interrupt1
  - OUT INTM1
  - OUT INTM2 ;disable interrupt2
- **Address mode 1**: MVI A,31H ;Address mode 1
  - OUT ADRM ;T/R mode 3 : 75160/1 drivers
- **My address = 0**: MVI A,00H ;my address=0
  - OUT ADR01 ;<address 0 register>
- **Disable address 1 register**: MVI A,0E0H ;disable address 1 register
XSEND: MVI A,88H ;continuous mode
OUT AUXM ;output EOI on EOS sent

CALL COCHECK ;CO Bit check (Command Output)
MVI A,3FH ;send UNLISTEN
OUT BOUT

CALL COCHECK ;send TALKER Address
MVI A,40H
OUT BOUT

CALL COCHECK ;send LISTENER Address
MVI A,21H
OUT BOUT

MV! A,1BH ;local message: 10n continuous
OUT AUXM ;(listen)

MOVI A,1AH ;local message: tcs
OUT AUXM ;(take control synchronously on end)

MV! A,10H ;local message: gts
OUT AUXM ;(go to standby)

CALL COCHECK ;DO Bit check (Data Output)
MOV A,B ;send contents of register B
OUT BOUT
; CALL DOCHECK ; send contents of register C
0060 79 MOV A,C
0061 D380 OUT BOUT

; CALL DOCHECK ; send contents of register D
0063 CD7800
0066 7A MOV A,D
0067 D380 OUT BOUT

; CALL DOCHECK ; send contents of register E
0069 CD7800
006C 7B MOV A,E
006D D380 OUT BOUT ; end of SEND-Routine

006F C9 RET

0070 DB82 COCHECK: IN INTM2 ; CO Bit check (Command Output)
0072 E608 ANI 08H
0074 CA7000 JZ COCHECK
0077 C9 RET

; DO Bit check (Data Output)
0078 DB81 DOCHECK: IN INTM1
007A E606 ANI 06H
007C CA7800 JZ DOCHECK
007F E604 ANI 04H
0081 C28500 JNZ ERR
0084 C9 RET

ERR: ; put in your individual ERROR Routine here

; ;
; ; ; MAIN Routine
; ; ============
; ; - initialise uPD7210
; ; - send IFC (interface clear)
; ; - load registers B,C,D,E
; ; - call SEND Routine
; ;
0085 CD0300 XMAIN: CALL INIT ; set IFC (interface clear)
0088 3E1E MVI A,1EH
008A D385 OUT AUXM
008C 0E20 MVI C,20H
008E 0D IFCWAIT: DCR C ; wait 100us
008F C28E00 JNZ IFCWAIT
0092 3E16 MVI A,16H ; clear IFC
0094 D385 OUT AUXM
0096 0630 ; load registers
0098 0E31  MVI  C.31H
009A 1632  MVI  D.32H
009C 1E0D  MVI  E.0DH  :carriage return CR

009E CD0600  CALL SEND  :send contents of the registers

00A1 DB81  ENDCHECK:IN  INTM1  :test END bit
00A3 E610  ANI  10H
00A5 CAA100  JZ  ENDCHECK
00A8 3E03  MVI  A.03  :send finish handshaking command
00AA D385  OUT  AUXM

00AC  END  :end of MAIN routine.
μPD7210

ELECTRICAL SPECIFICATION
### Absolute Maximum Ratings

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply Voltage</td>
<td>VCC</td>
<td>-0.5 ~ +7.0 V</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>V₁</td>
<td>-0.5 ~ +7.0 V</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Output Voltage</td>
<td>V₀</td>
<td>-0.5 ~ +7.0 V</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>T_{opt}</td>
<td>0 ~ +70 °C</td>
<td></td>
<td>°C</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>T_{stg}</td>
<td>-65 ~ +125 °C</td>
<td></td>
<td>°C</td>
</tr>
</tbody>
</table>

### DC Characteristics

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Low Voltage</td>
<td>VIL</td>
<td>±0.5 V</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Input High Voltage</td>
<td>VIH</td>
<td>±2.0 V</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Low Level Output Voltage</td>
<td>V_{OL}</td>
<td>±0.45 V</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>High Level Output Voltage</td>
<td>V_{OH1}</td>
<td>±2.4 V</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>V_{OH2}</td>
<td>±2.4 V</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Input Leakage Current</td>
<td>I_{IL}</td>
<td>-10 µA</td>
<td></td>
<td>µA</td>
</tr>
<tr>
<td>Output Leakage Current</td>
<td>I_{OL}</td>
<td>-10 µA</td>
<td></td>
<td>µA</td>
</tr>
<tr>
<td>Supply Current</td>
<td>ICC</td>
<td>±180 mA</td>
<td></td>
<td>mA</td>
</tr>
</tbody>
</table>

### Capacitance

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Capacitance</td>
<td>C_{IN}</td>
<td>t = 1 MHz</td>
<td>10 pF</td>
<td>pF</td>
</tr>
<tr>
<td>Output Capacitance</td>
<td>C_{OUT}</td>
<td>All Pins Except Pin Under Test Tied to AC Ground</td>
<td>15 pF</td>
<td>pF</td>
</tr>
<tr>
<td>I/O Capacitance</td>
<td>C_{I/O}</td>
<td></td>
<td>20 pF</td>
<td>pF</td>
</tr>
</tbody>
</table>

### AC Characteristics

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Setup to RD</td>
<td>TAR</td>
<td>150 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Address Hold from RD</td>
<td>TRA</td>
<td>200 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>RD Pulse Width</td>
<td>TRS</td>
<td>150 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Data Delay from Address</td>
<td>TAD</td>
<td>200 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Data Delay from RD</td>
<td>TRD</td>
<td>200 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Output Float Delay from RD</td>
<td>TDF</td>
<td>200 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>RD Recovery Time</td>
<td>TRV</td>
<td>200 ns</td>
<td></td>
<td>ns</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Setup to WR</td>
<td>TAW</td>
<td>0 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Address Hold from WR</td>
<td>TRA</td>
<td>0 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>WR Pulse Width</td>
<td>TWR</td>
<td>150 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Data Setup to WR</td>
<td>TDW</td>
<td>150 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Data Hold from WR</td>
<td>TRD</td>
<td>150 ns</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>WR Recovery Time</td>
<td>TRV</td>
<td>150 ns</td>
<td></td>
<td>ns</td>
</tr>
</tbody>
</table>
### AC CHARACTERISTICS

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MIN</td>
<td>TYP</td>
</tr>
<tr>
<td>EOI i = D10</td>
<td>IEOOI</td>
<td>PPSS + PPAS, ATN = True</td>
<td>250 ns</td>
<td></td>
</tr>
<tr>
<td>EOI i = T/R1</td>
<td>IEO11</td>
<td>PPSS + PPAS, ATN = True</td>
<td>155 ns</td>
<td></td>
</tr>
<tr>
<td>EOI i = T/R1</td>
<td>IEO12</td>
<td>PPAS + PPAS, ATN = False</td>
<td>200 ns</td>
<td></td>
</tr>
<tr>
<td>ATN i = NDAC</td>
<td>IATND</td>
<td>AID = ANRS, LIDS</td>
<td>155 ns</td>
<td></td>
</tr>
<tr>
<td>ATN i = T/R1</td>
<td>IATTI</td>
<td>TACS = SPA + TADS, CIDS</td>
<td>155 ns</td>
<td></td>
</tr>
<tr>
<td>ATN i = T/R2</td>
<td>IATT2</td>
<td>TACS = SPA + TADS, CIDS</td>
<td>200 ns</td>
<td></td>
</tr>
<tr>
<td>DAV i = DMAREQ</td>
<td>IDVRQ</td>
<td>ACRS = ACDS, LACD</td>
<td>600 ns</td>
<td></td>
</tr>
<tr>
<td>DAV i = NRF1</td>
<td>IDVR1</td>
<td>ACRS = ACDS</td>
<td>350 ns</td>
<td></td>
</tr>
<tr>
<td>DAV i = NDAC</td>
<td>IDVND1</td>
<td>ACRS = ACDS, AWNS</td>
<td>650 ns</td>
<td></td>
</tr>
<tr>
<td>DAV i = NDAC</td>
<td>IDVND2</td>
<td>ANNS = ANRS</td>
<td>350 ns</td>
<td></td>
</tr>
<tr>
<td>DAV i = NRF1</td>
<td>IDVNR1</td>
<td>ANNS = ANRS, ACRS</td>
<td>350 ns</td>
<td></td>
</tr>
<tr>
<td>RD i = NRF1</td>
<td>IRR1</td>
<td>ANRS = ACDS, LACD, DI reg. selected</td>
<td>500 ns</td>
<td></td>
</tr>
<tr>
<td>NDAC i = DMAREQ</td>
<td>lNDRO</td>
<td>STRS = SWNS = SGNS, TACS</td>
<td>400 ns</td>
<td></td>
</tr>
<tr>
<td>NDAC i = DAV</td>
<td>lNDDV</td>
<td>STRS = SWNS = SGNS</td>
<td>350 ns</td>
<td></td>
</tr>
<tr>
<td>WR i = D10</td>
<td>lNDI</td>
<td>SGNS = SDYS, BO reg. selected</td>
<td>250 ns</td>
<td></td>
</tr>
<tr>
<td>NRF1 i = DAV</td>
<td>lNRF1</td>
<td>SDYS = STRS, T1 = True</td>
<td>350 ns</td>
<td></td>
</tr>
<tr>
<td>WR i = DAV</td>
<td>lNDV</td>
<td>SGNS = SDYS = STRS, BO reg. selected, RFD = True</td>
<td>830 ns</td>
<td>*SYNC</td>
</tr>
</tbody>
</table>

### EXTENDED TEMPERATURE RANGE

\( T_a = 25^\circ C \)

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>RATINGS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply Voltage</td>
<td>VCC</td>
<td></td>
<td>-0.5 ( \sim ) 7.0</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>V1</td>
<td></td>
<td>-0.5 ( \sim ) 7.0</td>
<td>V</td>
</tr>
<tr>
<td>Output Voltage</td>
<td>V0</td>
<td></td>
<td>-0.5 ( \sim ) 7.0</td>
<td>V</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>( T_{opt} )</td>
<td></td>
<td>-40 ( \sim ) +85</td>
<td>( ^\circ C )</td>
</tr>
<tr>
<td>Strage Temperature</td>
<td>( T_{stg} )</td>
<td></td>
<td>-65 ( \sim ) +125</td>
<td>( ^\circ C )</td>
</tr>
</tbody>
</table>

\( T_a = -40^\circ C \ldots +85^\circ C, \ V_{CC} = \pm 5V \pm 10\% \)

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Low Voltage</td>
<td>VIL</td>
<td></td>
<td>-0.5</td>
<td>+0.6</td>
</tr>
<tr>
<td>Input High Voltage</td>
<td>VIH</td>
<td></td>
<td>+2.2</td>
<td>VCC +0.5</td>
</tr>
<tr>
<td>Low Level Output Voltage</td>
<td>VOH1</td>
<td>( \lvert I_OH \rvert = 2mA ) ( 4mA \ : T/R1 ) Pins</td>
<td>+0.45</td>
<td>V</td>
</tr>
<tr>
<td>High Level Output Voltage</td>
<td>VOH2</td>
<td>( \lvert I_{OH} \rvert = +400mA, \ Except ) INT PIn</td>
<td>+2.3</td>
<td>V</td>
</tr>
<tr>
<td>( \lvert I_{OH} \rvert = +50mA \ Int ) PIn</td>
<td></td>
<td>+2.3</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Input Leakage Current</td>
<td>IH</td>
<td>( I_{IN} \sim 0V \sim VCC )</td>
<td>-10</td>
<td>+10</td>
</tr>
<tr>
<td>Output Leakage Current</td>
<td>IL</td>
<td>( I_{OUT} \sim 0.45V \sim VCC )</td>
<td>-10</td>
<td>+10</td>
</tr>
<tr>
<td>Supply Current</td>
<td>ICC</td>
<td></td>
<td>+220</td>
<td>mA</td>
</tr>
</tbody>
</table>

\( T_a = 25^\circ C, VCC = GND = 0V \)

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Capacitance</td>
<td>CIN</td>
<td>All Pins Except Pin Under T = 1 MHz</td>
<td>10</td>
<td>pF</td>
</tr>
<tr>
<td>Output Capacitance</td>
<td>COUT</td>
<td>Test Tied to AC Ground</td>
<td>15</td>
<td>pF</td>
</tr>
<tr>
<td>I/O Capacitance</td>
<td>CIO</td>
<td></td>
<td>20</td>
<td>pF</td>
</tr>
</tbody>
</table>
NEC

μPD7210

T_a = -40°C~+85°C, V_{cc} = 5V ±10 %

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>SYMBOL</th>
<th>TEST CONDITIONS</th>
<th>LIMITS</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Setup to RD</td>
<td>TAR</td>
<td>RS20 CS</td>
<td>MIN</td>
<td>TYP</td>
</tr>
<tr>
<td>Address Hold from RD</td>
<td>TAH</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RD Pulse Width</td>
<td>TRR</td>
<td>170</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Delay from Address</td>
<td>TAD</td>
<td>275</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Delay from RD</td>
<td>TRD</td>
<td>150</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output Float Delay from RD</td>
<td>TDF</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RD Recovery Time</td>
<td>TRV</td>
<td>250</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMAREQ ⊕ Delay from DMAACK</td>
<td>TAKRO</td>
<td>160</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Delay from DMAACK</td>
<td>TAKD</td>
<td>240</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Address Setup to WR</td>
<td>TAW</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Address Hold from WR</td>
<td>TAH</td>
<td>+20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WR Pulse Width</td>
<td>TWW</td>
<td>170</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Setup to WR</td>
<td>TDW</td>
<td>150</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Hold from WR</td>
<td>TNW</td>
<td>+20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WR Recovery Time</td>
<td>TRW</td>
<td>250</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

EOD 1 = DID | TEO11 | PPSS = PPAS, ATN = True | 300 | | | ns |
EOD 1 = T/R 1 | TEO12 | PPSS = PPAS, ATN = True | 202 | | | ns |
AYN ⊕ RDAC 1 | TATND | AIGS = ANRS, LIQS | 202 | | | ns |
ATN ⊕ T/R 1 | TATT1 | TACS = SPAS + TADS, CIDS | 202 | | | ns |
ATN ⊕ T/R 2 | TATT2 | TACS = SPAS + TADS, CIDS | 202 | | | ns |
DAV ⊕ DMAREQ 1 | TIDVRK | ACRS = ACDS, LACS | 720 | | | ns |
DAV ⊕ NRFD 1 | TIDVR1 | ACRS = ACDS | 420 | | | ns |
DAV ⊕ NDAC 1 | TIDVND1 | ACRS = AWNS | 780 | | | ns |
DAV ⊕ NDAC 2 | TIDVND2 | AWNS = ANRS | 420 | | | ns |
DAV ⊕ NRFD 2 | TIDVR2 | AWNS = ANRS | 420 | | | ns |
RD ⊕ NRFD 1 | TRR | ANRS = ACRS | 600 | | | ns |
NRDC 1 = DMAREQ 1 | TINRD | STRS = SWNS + GNS, TACS | 480 | | | ns |
NRDC 1 = DAV 1 | TINDV | STRS = SWNS + GNS | 420 | | | ns |
WR ⊕ DID | TNW | SGNS = SDYS, BO reg. selected | 300 | | | ns |
NRFD 1 = DAV 1 | TNRDV | SDYS = STRS, T1 = True | 420 | | | ns |
WR ⊕ DAV 1 | TNW | SGNS = SDYS + STRS | 860 | *SYNC | | ns |

TRIG Pulse Width | TTRIG | 45 | | | | ns |

CS, RS2-Q

RD

D7-Q

DMAACK

DMAREQ

CS, RS2-Q

WR

D7-Q
APPENDIX A

This Chapter contains all Status Diagrams implemented in the uPD7210.

1. ACCEPTOR HANDSHAKE Function (AH)

<table>
<thead>
<tr>
<th>local messages</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pon</strong></td>
</tr>
<tr>
<td><strong>rdy</strong></td>
</tr>
<tr>
<td><strong>tcs</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>external messages</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ATN</strong></td>
</tr>
<tr>
<td><strong>DAC</strong></td>
</tr>
<tr>
<td><strong>DAV</strong></td>
</tr>
<tr>
<td><strong>RFD</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AH-States</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ACDS</strong></td>
</tr>
<tr>
<td><strong>AIDS</strong></td>
</tr>
<tr>
<td><strong>ANRS</strong></td>
</tr>
<tr>
<td><strong>ACRS</strong></td>
</tr>
<tr>
<td><strong>AWNS</strong></td>
</tr>
</tbody>
</table>
2. SOURCE HANDSHAKE Function (SH)

\[ \alpha = (\text{ATN} \land \overline{\text{CACS}}) \lor \overline{\text{SPAS}} \lor \text{CACS} \lor \overline{\text{TACS}} \lor \text{SPAS} \lor \text{TACS} \lor \text{DAV} \]

(within \( t_2 \))

| local messages | | external messages |
|----------------|------------------|
| pon            | power on         |
| nba            | new byte available |
|                |                   |
| ATN            | Attention         |
| DAC            | Data Accepted     |
| DAV            | Data Valid        |
| RFD            | Ready For Data    |

| SH-States       | |
|-----------------| |
| SDYS            | Source Delay State |
| SGNS            | Source Generator State |
| SIDS            | Source Idle State |
| SIWS            | Source Idle Wait State |
| STRS            | Source Transfer State |
| SWNS            | Source Wait For New Cycle State |
3. TALKER Functions (T,TE)

3.1 TALKER Function

- **TIDS**
  - IFC ∧ (NOT MTA ∧ ACOS)
  - (NUL) (END) (ROS)

- **TADS**
  - (NUL) (END) (ROS)
  - IFC ∧ (NOT MTA ∧ ACOS)
  - (OST [MPL]) ∧ ACOS

- **TACS**
  - DAB or EOS
  - END or END (ROS)
  - ATN (within t₂)

- **SPAS**
  - ATN ∧ SPMS
  - STB END or END
  - ROS or ROS

- **SPMS**
  - ATN (within t₂)
  - APRS APRS

- **SPS**
  - IFC ∧ SPE ∧ ACOS

- **STB**
  - END or END
  - ROS or ROS
  - APRS APRS

- **TIDS**
  - IFC (within t₄)

- **TADS**
  - IFC (within t₄)

- **TACS**
  - IFC (within t₄)

- **SPS**
  - IFC (within t₄)

- **SPMS**
  - IFC (within t₄)
3.2 TALKER EXTENDED Function

Diagram showing the TALKER EXTENDED Function with nodes labeled TIDS, TADS, SPAS, TACS, TPIS, TPAS, and SPIS, and transitions indicated by arrows with conditions such as IFC, MTA, OTA, OSA, ATN, and others involving conditions like (NUL), (END), (ROS), (ACDS), and (TPAS).
### local messages

<table>
<thead>
<tr>
<th>pon</th>
<th>power on</th>
</tr>
</thead>
<tbody>
<tr>
<td>ton</td>
<td>talk only</td>
</tr>
</tbody>
</table>

### external messages

<table>
<thead>
<tr>
<th>ATN</th>
<th>Attention</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC</td>
<td>Interface Clear</td>
</tr>
<tr>
<td>MLA</td>
<td>My Listen Address</td>
</tr>
<tr>
<td>MTA</td>
<td>My Talk Address</td>
</tr>
<tr>
<td>MSA</td>
<td>My Secondary Address</td>
</tr>
<tr>
<td>OTA</td>
<td>Other Talk Address</td>
</tr>
<tr>
<td>OSA</td>
<td>Other Secondary Address</td>
</tr>
<tr>
<td>SPE</td>
<td>Serial Poll Enable</td>
</tr>
<tr>
<td>SPD</td>
<td>Serial Poll Disable</td>
</tr>
<tr>
<td>RQS</td>
<td>Request Service</td>
</tr>
<tr>
<td>PCG</td>
<td>Primary Command Group</td>
</tr>
<tr>
<td>DAB</td>
<td>Data Byte</td>
</tr>
<tr>
<td>END</td>
<td>End</td>
</tr>
<tr>
<td>STB</td>
<td>Status Byte</td>
</tr>
</tbody>
</table>

### T,TE-States

<table>
<thead>
<tr>
<th>TACS</th>
<th>Talker Active State</th>
</tr>
</thead>
<tbody>
<tr>
<td>TADS</td>
<td>Talker Addressed State</td>
</tr>
<tr>
<td>TIDS</td>
<td>Talker Idle State</td>
</tr>
<tr>
<td>TPAS</td>
<td>Talker Primary Addressed State</td>
</tr>
<tr>
<td>TPIS</td>
<td>Talker Primary Idle State</td>
</tr>
<tr>
<td>SPAS</td>
<td>Serial Poll Active State</td>
</tr>
<tr>
<td>SPIS</td>
<td>Serial Poll Idle State</td>
</tr>
<tr>
<td>SPMS</td>
<td>Serial Poll Mode State</td>
</tr>
</tbody>
</table>
4. LISTENER Functions (L, LE)

4.1 LISTENER Function

4.2 LISTENER EXTENDED
**local messages**

<table>
<thead>
<tr>
<th>pon</th>
<th>power on</th>
</tr>
</thead>
<tbody>
<tr>
<td>lon</td>
<td>listen only</td>
</tr>
<tr>
<td>ltn</td>
<td>listen</td>
</tr>
<tr>
<td>lun</td>
<td>local unlisten</td>
</tr>
</tbody>
</table>

**external messages**

<table>
<thead>
<tr>
<th>ATN</th>
<th>Attention</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC</td>
<td>Interface Clear</td>
</tr>
<tr>
<td>UNL</td>
<td>Unlisten</td>
</tr>
<tr>
<td>MLA</td>
<td>My Listen Address</td>
</tr>
<tr>
<td>MTA</td>
<td>My Talk Address</td>
</tr>
<tr>
<td>MSA</td>
<td>My Secondary Address</td>
</tr>
</tbody>
</table>

**L,LE-States**

<table>
<thead>
<tr>
<th>LACS</th>
<th>Listener Active State</th>
</tr>
</thead>
<tbody>
<tr>
<td>LADS</td>
<td>Listener Addressed State</td>
</tr>
<tr>
<td>LIDS</td>
<td>Listener Idle State</td>
</tr>
<tr>
<td>LPAS</td>
<td>Listener Primary Addressed State</td>
</tr>
<tr>
<td>LPIS</td>
<td>Listener Primary Idle State</td>
</tr>
</tbody>
</table>
5. SERVICE REQUEST Function (SR)

<table>
<thead>
<tr>
<th>local messages</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>pon</td>
<td>power on</td>
</tr>
<tr>
<td>rsv</td>
<td>request service</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>external messages</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SRQ</td>
<td>Service Request</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SR-States</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>APRS</td>
<td>Affirmative Poll Response State</td>
</tr>
<tr>
<td>NPRS</td>
<td>Negative Poll Response State</td>
</tr>
<tr>
<td>SRQS</td>
<td>Service Request State</td>
</tr>
</tbody>
</table>

3-64
5. REMOTE/LOCAL Function (RL)

**Local messages**
- **pon**: power on
- **rtl**: return to local

**External messages**
- **REN**: Remote Enable
- **LLO**: Local Lockout
- **GTL**: Go To Local
- **MLA**: My Listen Address
- **MSA**: My Secondary Address

**RL-States**
- **LWLS**: Local With Lockout State
- **LOCS**: Local State
- **REMS**: Remote State
- **RWLS**: Remote With Lockout State
7. PARALLEL POLL Function (PP)

**Local messages**
- **pon** power on
- **ist** individual status
- **lpe** local poll enable

**External messages**
- **ATN** Attention
- **IDY** Identify
- **PPE** Parallel Poll Enable
- **PPD** Parallel Poll Disable
- **PPC** Parallel Poll Configure
- **PPU** Parallel Poll Unconfigure
- **PCG** Primary Command Group
- **PPRn** Parallel Poll Response

**PP-States**
- **PPAS** Parallel Poll Active State
- **PACS** Parallel Poll Addressed To Configure State
- **PPIS** Parallel Poll Idle State
- **PPSS** Parallel Poll Standby State
- **PUCS** Parallel Poll Unaddressed To Configure State
8. DEVICE CLEAR Function (DC)

External messages

<table>
<thead>
<tr>
<th>DCL</th>
<th>Device Clear</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDC</td>
<td>Selected Device Clear</td>
</tr>
</tbody>
</table>

DC-States

| DCAS | Device Clear Active State |
| DCIS | Device Clear Idle State |

9. DEVICE TRIGGER Function

External messages

| GET | Group Execute Trigger |

DT-States

| DTIS | Device Trigger Idle State |
| DTAS | Device Trigger Active State |
10. CONTROLLER Function (C)

![Diagram of controller function (C)]
### Local Messages

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pon</td>
<td>power on</td>
</tr>
<tr>
<td>gts</td>
<td>go to standby</td>
</tr>
<tr>
<td>rpp</td>
<td>request parallel poll</td>
</tr>
<tr>
<td>rsc</td>
<td>request system control</td>
</tr>
<tr>
<td>sic</td>
<td>send interface clear</td>
</tr>
<tr>
<td>sre</td>
<td>send remote enable</td>
</tr>
<tr>
<td>tca</td>
<td>take control asynchronously</td>
</tr>
<tr>
<td>tcs</td>
<td>take control synchronously</td>
</tr>
</tbody>
</table>

### External Messages

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATN</td>
<td>Attention</td>
</tr>
<tr>
<td>IPC</td>
<td>Interface Clear</td>
</tr>
<tr>
<td>TCT</td>
<td>Take Control</td>
</tr>
<tr>
<td>IDN</td>
<td>Identify</td>
</tr>
<tr>
<td>REN</td>
<td>Remote Enable</td>
</tr>
</tbody>
</table>

### C-States

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CACS</td>
<td>Controller Active State</td>
</tr>
<tr>
<td>CNWS</td>
<td>Controller Active Wait State</td>
</tr>
<tr>
<td>CADS</td>
<td>Controller Addressed State</td>
</tr>
<tr>
<td>CIDS</td>
<td>Controller Idle State</td>
</tr>
<tr>
<td>CPFS</td>
<td>Controller Parallel Poll State</td>
</tr>
<tr>
<td>CPWIS</td>
<td>Controller Parallel Poll Wait State</td>
</tr>
<tr>
<td>CSNS</td>
<td>Controller Service Not Requested State</td>
</tr>
<tr>
<td>CSRS</td>
<td>Controller Service Requested State</td>
</tr>
<tr>
<td>CSBS</td>
<td>Controller Standby State</td>
</tr>
<tr>
<td>CSWS</td>
<td>Controller Synchronous Wait State</td>
</tr>
<tr>
<td>CTRS</td>
<td>Controller Transfer State</td>
</tr>
<tr>
<td>SACS</td>
<td>System Control Active State</td>
</tr>
<tr>
<td>SIAS</td>
<td>System Control Interface Clear Active State</td>
</tr>
<tr>
<td>STNS</td>
<td>System Control Interface Clear Not Active State</td>
</tr>
<tr>
<td>SIIS</td>
<td>System Control Interface Clear Idle State</td>
</tr>
<tr>
<td>SNAS</td>
<td>System Control Not Active State</td>
</tr>
<tr>
<td>SRAS</td>
<td>System Control Remote Enable Active State</td>
</tr>
<tr>
<td>SRIS</td>
<td>System Control Remote Enable Idle State</td>
</tr>
<tr>
<td>SRNS</td>
<td>System Control Remote Enable Not Active State</td>
</tr>
</tbody>
</table>
Timing values for State Transitions

\[ T_1 \geq 2\, \text{us} \quad \text{for open collector drivers} \]
\[ \geq 1\,100\,\text{ns} \quad \text{if DIO, DAV, EOI driven by tri-state drivers} \]
\[ \geq 700\,\text{ns} \quad \text{if DIO, DAV, EOI, ATN driven by tri-state drivers} \]
\[ \geq 500\,\text{ns} \quad \text{for all bytes following ATN=false} \]
\[ \geq 350\,\text{ns} \quad \text{for tri-state drivers and short connections} \]
\[ t_2 \leq 200\,\text{ns} \]
\[ t_4 < 100\,\text{us} \]
\[ T_6 \geq 2\,\text{us} \]
\[ T_7 \geq 500\,\text{ns} \]
\[ T_8 \geq 100\,\text{us} \]
\[ T_9 \geq 1.5\,\text{us} \]
\[ \geq 600\,\text{ns} \quad \text{for tri-state drivers} \]
Programming the GPIB Controller
μPD7210

1. Introduction

2. Software Considerations

3. Description of Routines
   3.1 Initialization
   3.2 Interrupt 0
   3.3 Interrupt 1
   3.4 Send Data
   3.5 Receive Data
   3.6 Transfer Data
   3.7 Device Trigger
   3.8 Device Clear
   3.9 Execute Serial Polling
   3.10 Execute Parallel Poling
   3.11 Parallel Poll Configure
   3.12 Interface Clear
   3.13 Local
   3.14 Remote

Appendix
Software Listing
1 Introduction

The µPD7210 allows to implement a fully IEEE-488 (1978) compatible general purpose interface bus (GPIB) with only a minimum amount of hardware. In Europe the IEEE-488 specification is adapted to IEC-625 standard with the same electrical features but different cable and connector. The complete schematic on an IEEE-488 controller board is shown below.

On the other hand the GPIB bus is a very powerful bus by means of communication speed and control protocols. This application note provides a set of routines that allows to handle these protocols. All routines are written in µPD8085 assembly language and can be transferred to any other programming language without problems. Each piece of software is well documented so the user can easily extract those routines which are needed for his application.
2 Software Considerations

Before any software should be written for a GPIB, the user should decide what IEEE functions he desires, which will define what processor routines are necessary. Second, you should develop the necessary interrupt and software protocols to suit your microcomputer system. Lastly, you should write the necessary software. At the end of this application note are program listings with description that should help you to understand what functions are used in this example, and the details of how these functions are implemented. The program flow for the routines included are as follows.

The μPD7210 (Talker/Listener/Controller) interface is activated by calling the INIT routine, thus initializing the μPD7210 and the GPIB. The INT1 routine should be jumped to by any interrupt generated because of changing bus status and polling requests. This routine reads the μPD7210 interrupt status and stores it in memory so that your software can check and respond when available.

The INT0 routine is used to speed data and polling transfers, and should be vectored to by a different interrupt than INT1. The μPD7210 is set up for this type of operation as part of routines for that can use it, such as SEND and RECV.

Now your main program can respond to changing status and initiate desired functions by calling a particular routine. You should note from the routine descriptions that for some routines it is necessary for the main program to initialize its own processor registers with the necessary routine parameters.

3 Description of Routines

The subroutines described below allow you to

- configure your system (i.e., talkers, listeners, remote, local, IFC, etc)
- transfer data between devices or the controller
- service device requests in a serial or parallel format
- finally, trigger synchronous operation of all devices on the bus.

The NEC μPD7210 can make your GPIB interface easier, more versatile, and more powerful than by using first generation controllers, and in most systems, second generation controller also.

ROUTINE LIST

<table>
<thead>
<tr>
<th>Routine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INIT</td>
<td>Initialization</td>
</tr>
<tr>
<td>INTO</td>
<td>Interrupt 0</td>
</tr>
</tbody>
</table>
3.1 Initialization

The operating mode of the μPD7210 is set in this routine. The first function of the routine is to issue a chip reset, thereby disabling the GPIB interface functions. The next operation is to set the operating modes of the μPD7210, which includes interrupt masks, address mode, device address, EOS code. To finalize this routine, the Immediate Execute pon command is issued followed by the system interface clear command.

3.2 Interrupt 0

This routine is initiated by the DMA REQ signal which requests a data transfer when sending data, receiving data, or when executing a serial polling system command. Before executing these system commands, source or destination address of the first data byte is set to the HL register pair (data pointer) of the CPU. In case of the Send Data Command, the BC register pair is used to hold the number of data bytes to be send (data counter).

The subroutine's flow is as follows:

The TA (Talker Active) bit in the Address Status register is checked first to decide on the direction of the data transfer. If it is cleared (data input mode), GPIB data is available in the Data In register and is transferred to the memory location pointed by HL. The data pointer is then incremented preparing for the next byte.

If the TA bit is set (data output mode), the data byte pointed by HL is transferred to the
Byte Out register, sending the data byte onto the GPIB. The data pointer is then incremented and the data counter (BC) is decremented. If the contents of the data counter (BC) becomes 1, a send EOI command issued, thereby alerting the μPD7210 to send END message with the next byte (Last byte). After the last byte is sent (BC = 0), the DMAO bit of the μPD7210 is reset inhibiting any further data transfer. Finally, a subroutine return is implemented.

3.3 Interrupt 1

This routine is initiated by the INT signal, caused by a SRQI (service request interrupt) or the CO (command output request) bit being set. The interrupt status is read from the Interrupt Status 2 register of the μPD7210 and is kept in memory because it is automatically cleared by the read. The interrupt status byte in memory will be read and cleared, bit by bit, by the routine which uses it.

3.4 Send Data

This routine sends data from the microcomputer system to one or more devices via the GPIB using the HL,BC,and DE register pairs of the CPU as data pointer, data counter and device address pointer, respectively. Each device with its device address listed in an address table pointed by the DE register pair is assigned to receive data (a listener). Then all of the data in the data buffer specified by the HL and BC register pairs is transferred.

This routine assumes that device address of the microcomputer system is not involved in the address table. The last device address in the table is assumed to be followed by a delimiter having a value which is greater than 30.

3.5 Receive Data

This routine is used to input data from a device whose device address is in the B register of the CPU. The system is configured such that the microcomputer is the only listener and receives data until an END or EOS (End of String) message is received. The data is stored in the data buffer pointed by the HL register pair. The talker device is assumed not to be the microcomputer system itself.

3.6 Transfer Data

This routine is used to transfer data from a device whose device address is in the B
register to one or more devices whose device address(es) is (are) listed in the address table pointed by the DE register pair. The microcomputer system does not receive the data but lets itself be a listener, so it can detect an END or EOS message from talker. On detection of them, data transfer is terminated. The microcomputer system does not participate in the data transfer itself.

3.7 Device Trigger

This routine issues a GET (Group Execute Trigger) message onto the GPIB after addressing devices listed in the address table pointed by the DE register pair. This routine is intended to start operation in the listed devices.

3.8 Device Clear

This routine issues a SDC (Selected Device Clear) message onto the GPIB after addressing devices listed in the address table pointed by the DE register pair. This routine is intended to clear or initialize these devices. If there is no device address listed, a DCL (Device Clear) message is issued instead of a SDC message, thus clearing all devices.

3.9 Execute Serial Polling

This routine executes serial polling according to the sequence of device addresses in the address table pointed by the DE register pair. One byte of status from each device is received and stored in the data buffer pointed by the HL register pair in the same order as in the address table. The device address of the microcomputer system is assumed not to be involved in the address table.

3.10 Execute Parallel Polling

This routine executes parallel polling and returns one byte of status, Parallel Poll Response, in the A register of the CPU.

3.11 Parallel Poll Configure

This routine configures one or more devices to respond to parallel polling assuming
each device can implement the PP1 interface function. The device address are listed in the address table pointed by the DE register pair and the configuration information is stored in a buffer pointed by the HL register pair in the same order as in the adress table.
The least significant five bits of the configuration byte are assumed to have the same format as PPE (Parallel Poll Enable) or the PPD (Parallel Poll Disable) message defined by the IEEE Standard 488.

3.12 Interface Clear

This routine activates the GPIB's IFC line for 100 microseconds, causing the interface functions of all devices to go to known state. This routine is also executed at the end of the initialization routine.

3.13 Local

This routine issues a GTL (Go To Local) message after addressing the devices listed in the address table pointed by the DE register pair. This causes the addressed devices to go to the local state. If there are no device addresses listed in the address table, the GPIB's REN line is inactivated, letting all of the devices go to local state. This routine is intended to enable the devices to receive local data.

3.14 Remote

This routine activates the GPIB's REN line enabling each device to go to remote state when addressed to listen.

Appendix Software Listing
;*******************************************************************************
;* UTILILITY ROUTINES FOR uPD7210 GPIB CONTROLLER
;*******************************************************************************

; Write Registers
; ===============

E0&REG EQU 00H
INT&M1 EQU 01H
INT&M2 EQU 02H
ADR&MODE EQU 04H
AUX&MODE EQU 05H
ADR&REG EQU 06H
EOS&REG EQU 07H

; Read Registers
; ==============

DH&REG EQU 00H
INT&ST1 EQU 01H
INT&ST2 EQU 02H
ADR&ST EQU 04H
CPT&REG EQU 05H

; GENERAL EQUATES
; ===============

MY&ADR EQU 04H
EOS&CODE EQU 00H
FREQ EQU 08H
AUX&A EQU 00H

; INITIALISATION
; =========

INIT: MVI A,02H ;CHIP RESET
OUT AUX&MODE
XRA A
OUT INT&M1 ;DISABLE INTERRUPTS OTHER THAN SRQ1 & CO
MVI A,58H
OUT INT&M2 ;ENABLE DMAI,DISABLE DMA0
MVI A,31H
OUT ADR&MODE ;SET T/R MODE 3,SET ADDRESS MODE1
LDA MY&ADR
OUT ADR&REG ;SET MY ADDRESS 0-30 TO ADDRESS 0 REGISTER
MVI A,0EOH
OUT ADR&REG ;DISABLE ADDRESS 1 REGISTER
MVI A,EOS&CODE
OUT EOS&REG ;SET EOS CODE
MVI A,FREQ OR 20H ;SET FREQUENCY OF CLOCK INPUT 0-8 MHZ
OUT AUX&MODE
MVI A, AUX&A OR 80H ;SET AUXILIARY MODE A
OUT AUX&MODE ;SENDS EOI IN SPAS, HIGH SPEED T1
MVI A, 0A6H
OUT AUX&MODE
XRA A
STA INT&ST1 ;CLEAR WORKING AREA FOR INTERRUPT STATUS
OUT AUX&MODE ;IMMEDIATE EXECUTE PON
CALL IFC
JMP WAIT&CO ;RETURN ON ENTERING INTO CACS

; INTERRUPT 0

; INTO:

PUSH PSW
IN ADR&ST
ANI 02H ;TA = 1?
JNZ DATA&OUT ;DATA IN
IN DI&REG ;DATA IN
MOV M, A ;STORE GPIB DATA
INX H ;INCREMENT DATA POINTER
RETURN: POP PSW
EI
RET

DATA&OUT:

MOV A, M ;DATA OUT
OUT BO&REG ;LOAD GPIB DATA
INX H ;INCREMENT DATA POINTER
DCX B ;DECREMENT DATA COUNTER
XRA A
ORA B
JNZ RETURN ;RETURN IF (BC) IS GREATER THAN 2
INR A
CMP C
JC RETURN ;RETURN IF (BC) IS GREATER THAN 2
MVI A, 6H ;(BC) = 0 OR 1
OUT AUX&MODE ;SEND EOI WITH THE NEXT BYTE
JZ RETURN ;RETURN IF (BC) = 1
MVI A, 58H ;(BC) = 0
OUT INT&M2 ;DISABLE DMA1 INTERRUPT

LOOP:

IN INT&ST1
ANI 2H ;DO = 1?
JZ LOOP ;WAIT UNTIL HANDSHAKE IS FINISHED
MVI A, 11H ;TCA (TAKE CONTROL ASYNCHRONOUSLY) CODE
OUT AUX&MODE ;ISSUE TCA TO 7210
POP PSW
EI
RET

; INTERRUPT 1

; INTO:

PUSH PSW
PUSH H
IN INT&ST2 ;READ INTERRUPT STATUS 2 REGISTER

3-82
LXI H,INT&ST1
ORA M
MOV M,A ;SET INTERRUPT STATUS BYTE IN MEMORY
POP H
POP FSW
EI
RET

;UTILITY SUBROUTINES
;=====================

;WAIT&CO: PUSH H
LXI H,INT&ST1
MOV A,M ;LOAD INTERRUPT STATUS BYTE
ANI 8H ;CO = 1?
JZ WAIT&CO+1 ;WAIT UNTIL CO BIT IS SET
DI
XRA M
MOV M,A ;CLEAR CO BIT
POP H
EI
RET

;UNLISTEN
;=========

UNLTN: MVI A,3FH ;UNL (UNLISTEN) CODE
OUT BO&REG ;ISSUE UNL ONTO GPIB
JMP WAIT&CO ;RETURN WHEN HANDSHAKE IS FINISHED

;ADDRESS LISTENERS (DEVICE ADDRESS POINTER = DE)
;===============================================

ADR&L: CALL UNLTN ;ISSUE UNL ONTO GPIB
LOOP1: LDAX D ;LOAD DEVICE ADDRESS
CPI 31 ;DELIMITER?
RNC
ORI 20H ;RETURN IF DELIMITER
OUT BO&REG ;ISSUE LISTEN ADDRESS ONTO GPIB
INX D ;INCREMENT LISTEN ADDRESS POINTER
CALL WAIT&CO ;WAIT UNTIL HANDSHAKE IS FINISHED
JMP LOOP1 ;REPEAT

;ADDRESS TALKER (TALK ADDRESS = B)
;=================================

ADR&T: MOV A,B
ADRT1: ORI 40H ;FROM TALK ADDRESS
OUT BO&REG ;ISSUE TALK ADDRESS ONTO GPIB
JMP WAIT&CO ;RETURN WHEN HANDSHAKE IS FINISHED
;SEND DATA (DATA POINTER = HL, DATA COUNTER = BC, DEVICE ADDRESS POINTER = DE)
;==============================================================================
;SEND:
CALL ADR&L ;ADDRESS LISTENERS
IN ADR&REG ;READ MY ADDRESS
ORI 40H ;FROM MY TALK ADDRESS
OUT BO&REG ;ISSUE MTA ONTO GPIB
CALL WAIT&CO ;WAIT UNTIL HANDSHAKE IS FINISHED
MVI A, 78H ;INTERRUPT MASK 2
OUT INT&M2 ;ENABLE DMAO INTERRUPT
MVI A, 10H ;GTS (GO TO STANDBY) CODE
OUT AUX&MODE ;ISSUE GTS TO 7210
JMP WAIT&CO ;RETURN ON DATA CYCLE TERMINATION
;RECIEVE DATA (DATA POINTER = HL, TALK ADDRESS = B)
;==============================================================================
;RECV:
CALL ADR&T ;ADDRESS TALKER
CALL UNLTN ;ISSUE UNL (UNLISTEN) ONTO GPIB
MVI A, 13H ;LTD (LISTEN) CODE
RECV1: OUT AUX&MODE ;ISSUE LTN OR LTNC TO 7210
MVI A, 1AH ;TCSE (TAKE CONTROL SYNCHRONOUSLY ON END) CODE
RECV2: OUT AUX&MODE ;ISSUE TCSE OR TCS TO 7210
MVI A, 10H ;GTS (GO TO STANDBY) CODE
OUT AUX&MODE ;ISSUE GET TO 7210 TO INITIATE DATA CYCLE
JMP WAIT&CO ;RETURN ON DATA CYCLE TERMINATION
;TRANSFER DATA (DEVICE ADDRESS POINTER = DE, TALK TALK ADDRESS = B)
;==============================================================================
;XFER:
CALL ADR&L ;ADDRESS LISTENERS
CALL ADR&T ;ADDRESS TALKER
MVI A, 1BH ;LTNC (LISTEN WITH CONTINUOUS MODE) CODE
JMP RECV1 ;ISSUE LTNC TO 7210, START DATA TRANSFER
;AND RETURN ON DATA TRANSFER TERMINATION
;DEVICE TRIGGER (DEVICE ADDRESS POINTER = DE)
;==============================================================================
;TRIG:
CALL ADR&L ;ADDRESS LISTENERS
MVI A, 8H ;GET (GROUP EXECUTE TRIGGER) CODE
OUT BO&REG ;ISSUE GET ONTO GPIB
JMP WAIT&CO ;RETURN WHEN HANDSHAKE IS FINISHED
;DEVICE CLEAR (DEVICE ADDRESS POINTER = DE)
;==============================================================================
;CLEAR:
LDAX D ;LOAD DEVICE ADDRESS
CPI 31 ;NO DEVICE ADDRESS?
MVI A, 14H ;DCL (DEVICE CLEAR) CODE
JC CLEAR1 ;ADDRESS LISTENERS
CALL ADR&L ;SCL (SELECTED DEVICE CLEAR) CODE
NEC

μPD7210

CLEAR1: OUT BO&REG ; ISSUE DCL OR SDC ONTO GPIB
JMP WAIT&CO ; RETURN WHEN HANDSHAKE IS FINISHED

; EXECUTE SERIAL POLLING (DATA POINTER = HL, DEVICE ADDRESS POINTER = DE)
;=====================================================================

EXSP: MVI A,18H ; SPE (SERIAL POLL ENABLE) CODE
OUT BO&REG ; ISSUE SPE ONTO GPIB
CALL WAIT&CO ; WAIT UNTIL HANDSHAKE IS FINISHED
CALL UNLTN ; ISSUE UNLTN (UNLISTEN) ONTO GPIB
MVI A,13H ; LTN (LISTEN) CODE
OUT AUX&MODE ; ISSUE LTN TO 7210

LOOP2: LDAX D ; LOAD DEVICE ADDRESS
JC CONTN ; CONTINUE IF NOT DELIMITER
MVI A,19H ; SPD (SERIAL POLL DISABLE) CODE
OUT BO&REG ; ISSUE SPD ONTO GPIB
JMP WAIT&CO ; RETURN WHEN HANDSHAKE IS FINISHED

CO:ITN: CALL ADDR&T1 ; ADDRESS TALKER
MVI A,12H ; TCS (TAKE CONTROL SYNCHRONOUSLY) CODE
CALL RECV2 ; ISSUE TCS AND GTS TO 7210
; INITIATING STB TRANSFER
; RETURN ON RECEIPTION OF STB
INX D ; INCREMENT TALK ADDRESS POINTER
JMP LOOP2 ; REPEAT

; EXECUTE PARALLEL POLLING (PARALLEL POLL RESPONSE = A)
;=====================================================================

EXPP: MVI A,1DH ; EPP (EXECUTE PARALLEL POLLING) CODE
OUT AUX&MODE ; ISSUE EPP TO 7210
CALL WAIT&CO ; WAIT UNTIL LPARALLEL POLLING IS COMPLETED
IN CPT&REG ; READ PPR FROM 7210
RET

; PARALLEL POLL CONFIGURE (DEVICE ADDRESS POINTER = DE)
;=====================================================================

PFC: LDAX D ; LOAD DEVICE ADDRESS
CFI 31 ; DELIMITER?
RNC ; RETURN IF DELIMITER
CALL UNLTN ; ISSUE UNLTN (UNLISTEN) ONTO GPIB
LDAX D ; RELOAD DEVICE ADDRESS
ORI 20H ; FROM LISTEN ADDRESS
OUT BO&REG ; ISSUE LISTEN ADDRESS ONTO GPIB
CALL WAIT&CO ; WAIT UNTIL HANDSHAKE IS FINISHED
MVI A,5 ; PFC (PARALLEL POLL CONFIGURE) CODE
OUT BO&REG ; ISSUE PFC ONTO GPIB
CALL WAIT&CO
MOV A,M ; LOAD SECONDARY COMMAND
ORI 60H ; FORM SECONDARY COMMAND
OUT BO&REG ; ISSUE PPE OR PPD ONTO GPIB
INX D ; INCREMENT POINTERS
INX H

3-85
CALL WAIT&CO
JMP PPC ; REPEAT

; INTERFACE CLEAR
; ===============
; IFC:
MVI A,1EH ; SIFC (SET IFC) CODE
OUT AUX&MODE ; ISSUE SIFC TO 7210 ACTIVATING IFC GPIB LINE
CALL WAIT&100 ; WAIT 100 MICROSECONDS
MVI A,16H ; RIFC (RESET IFC) CODE
OUT AUX&MODE ; ISSUE RIFC TO 7210 INACTIVATING IFC
RET

; LOCAL (DEVICE ADDRESS POINTER = DE)
; ===============
;
LOC: LDAX D ; LOAD ADDRESS
   CPI 31 ; NO ADDRESS?
   JC LOC1
   MVI A,17H ; RREN (RESET REN) CODE
   OUT AUX&MODE ; ISSUE RREN TO 7210 INACTIVATING GPIB LINE
   RET

LOC1: CALL ADR&L ; ADDRESS LISTENERS
       MVI A,1 ; GTL (GO TO LOCAL) CODE
       OUT B0&REG ; ISSUE GTL ONTO GPIB
       JMP WAIT&CO ; RETURN WHEN HANDSHAKE IS FINISHED

; REMOTE
; =
;
REM: MVI A,1FH ; SREN (SET REN) CODE
OUT AUX&MODE ; ISSUE SREN TO 7210 ACTIVATING REN GPIB LINE
RET

; WAIT&100: RET ; DUMMY DELAY ROUTINE
; END
μPD72105

LOCAL AREA NETWORK CONTROLLER
INTRODUCTION

The uPD72105 CSMAC (CSMA Controller) is a network controller that supports the Omninet LAN Protocol developed by Corvus Systems Inc. The on-chip DMA capability of the uPD72105 enables the host system to communicate with other nodes in the network by writing the command bytes and reading results directly to and from memory.

Features:
- Supports Omninet LAN Protocol I and II
- Transfer rates: 0.5 to 4 Mbps
  - Omninet LAN Protocol I: 1 Mbps
  - Omninet LAN Protocol II: 4 Mbps
- Command chain function
- On-chip DMA controller
- 8/16-bit system bus compatible
- Memory space: 16M bytes ($2^{24}$)
- Transmit FIFO: 12 bytes
  - Receive FIFO: 20 bytes
- 16/32-bit CRC
- On-chip DPLL (40 MHz clock input)
- LOOP BACK test capability
- 8-MHz system clock (independent from the serial clock)
- CMOS technology
- 48-pin DIP

Note: The use of this device requires a licensing agreement with Corvus Systems Inc.


Pin Connection (Top View)

- **CRQ**: Command Request
- **RxC**: Receive Clock
- **RxD**: Receive Data
- **TxC**: Transmit Clock
- **TxD**: Transmit Data
- **TxEN**: Transmit Enable
- **RESET**: Reset
- **LED**: LED Drive
- **B/W**: Byte/Word
- **TEST**: Test
- **CLK**: Clock
- **A23/D7-A16/D0**: Address Bus,
  **AD15-AD8**, **AD7-A0**: Data Bus
- **AEN**: Address Enable
- **ASTB**: Address Strobe
- **READY**: Ready
- **HLDAK**: Hold Acknowledge
- **HLDRQ**: Hold Request
- **CLRINT**: Clear Interrupt Request
- **INT**: Interrupt Request
- **UBE**: Upper Byte Enable
- **WR**: Write
- **RD**: Read
- **CS**: Chip Select
1. PIN FUNCTIONS

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Number</th>
<th>I/O</th>
<th>Active Level</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDD</td>
<td>36</td>
<td>-</td>
<td>-</td>
<td>+5 power supply pin</td>
</tr>
<tr>
<td>GND</td>
<td>12</td>
<td>-</td>
<td>-</td>
<td>Ground potential (OV) pin</td>
</tr>
<tr>
<td>CLK</td>
<td>11</td>
<td>I</td>
<td>-</td>
<td>Inputs a single-phase system clock.</td>
</tr>
<tr>
<td>TEST</td>
<td>10</td>
<td>I</td>
<td>L</td>
<td>Tests the uPD72105 and is normally held high.</td>
</tr>
<tr>
<td>RESET</td>
<td>7</td>
<td>I</td>
<td>L</td>
<td>Resets the internal circuitry of the chip. The uPD72105 serves as a bus slave after resetting. The minimum pulse width is as 7 clocks.</td>
</tr>
<tr>
<td>CS</td>
<td>38</td>
<td>I</td>
<td>L</td>
<td>Enables RD and WR signals when the chip operates as a bus slave.</td>
</tr>
<tr>
<td>RD</td>
<td>39</td>
<td>I/O</td>
<td>L</td>
<td>Three-state control signal. It acts as an input pin when the chip operates as a bus slave, allowing an external processor to read the contents of the internal registers of the uPD72105. With the uPD72105 operating as a bus master, this pin acts as an output pin allowing the uPD72105 to read the external memory contents.</td>
</tr>
<tr>
<td>WR</td>
<td>40</td>
<td>I/O</td>
<td>L</td>
<td>Three-state control signal. It acts as an input when the chip operates as a bus slave, allowing an external processor to write data to the internal registers of the uPD72105. With the uPD72105 operating as a bus master, this pin acts as an output pin allowing the uPD72105 to write data into external memory.</td>
</tr>
<tr>
<td>Pin Name</td>
<td>Pin Number</td>
<td>I/O</td>
<td>Active Level</td>
<td>Function</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
<td>-----</td>
<td>--------------</td>
<td>----------</td>
</tr>
<tr>
<td>B/W</td>
<td>9</td>
<td>I</td>
<td>L/H</td>
<td>Specifies a data bus to be accessed by the chip operating as a bus master.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>B/W=0 Byte (8-bit) transfers</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>B/W=1 Word (16-bit) transfers</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Do not change the state of this pin after power-up.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>When word access is performed, the lower bits of the data bus access even-numbered addresses</td>
</tr>
<tr>
<td>HLDRQ</td>
<td>44</td>
<td>O</td>
<td>H</td>
<td>Outputs the hold request signal to the host processor. The uPD72105, when performing DMA within the chip, activates this signal to request the right to control the system bus to start operation as a bus master.</td>
</tr>
<tr>
<td>HLDAK</td>
<td>45</td>
<td>I</td>
<td>H</td>
<td>Inputs the hold acknowledge signal from the host processor. When this signal becomes high, the uPD72105 operates as a bus master and starts DMA.</td>
</tr>
<tr>
<td>AEN</td>
<td>48</td>
<td>O</td>
<td>H</td>
<td>Enables a higher address that is latched when the chip operates as a bus master for outputting to the system address bus. This pin is also used to disable the other system bus drivers.</td>
</tr>
<tr>
<td>UBE</td>
<td>41</td>
<td>I/O</td>
<td>L</td>
<td>Three-state control signal. It acts as an input pin and indicates whether valid data are present on the A16/D0-A23/D7 or AD8-AD15 when the chip operates as a bus slave.</td>
</tr>
</tbody>
</table>
The state of this pin changes depending on the state of the B/W pin when the chip operates as a bus master. It is always in a high-impedance state in the byte transfer mode (when B/W=0), whereas it acts as an output pin in the word transfer mode (when B/W=1), indicating whether Valid data is on A16/DO-A23/D7 or AD8-AD15.

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Number</th>
<th>I/O</th>
<th>Active Level</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>UBE</td>
<td>A0</td>
<td>A16/DO-A23/D7</td>
<td>AD8-AD15</td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>o</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>x</td>
<td>o</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>o</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>o</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ASTB 47 O H Strobes the higher address to an external latch.

READY 46 I H Indicates when the memory is in the ready state. This signal is inactivated during the wait cycle in which the width of the RD and WR signals output by the uPD72105 are extended so that slow-speed memories are accessed.
<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Number</th>
<th>I/O</th>
<th>Active Level</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRQ</td>
<td>1</td>
<td>I</td>
<td>H</td>
<td>Signal with which the host system requests the uPD72105 to execute commands. The uPD72105 receives a command at the rising edge of this signal and starts executing it.</td>
</tr>
<tr>
<td>INT</td>
<td>42</td>
<td>O</td>
<td>H</td>
<td>Indicates that the uPD72105 has completed command execution or received a packet.</td>
</tr>
<tr>
<td>CLRINT</td>
<td>43</td>
<td>I</td>
<td>H</td>
<td>Inactivated the INT signal output by the uPD72105. The INT signal is inactivated at the rising edge of this signal.</td>
</tr>
<tr>
<td>A0,A1</td>
<td>37,35</td>
<td>I/O</td>
<td>-</td>
<td>Bidirectional three-state address lines. These pins act as output pins to indicate the lowest 2 bits of a memory address when the chip operates as a bus master. With the chip operating as a bus slave, they act as input pins to indicate the lowest 2 bits of an I/O address with which an external processor accesses the uPD72105.</td>
</tr>
<tr>
<td>A2-A7</td>
<td>34-29</td>
<td>O</td>
<td>-</td>
<td>Three-state address lines. These pins act as output pins to indicate bits 2 to 7 of a memory address when the chip operates as a bus master. They are in a high-impedance state when the chip operates as a bus slave.</td>
</tr>
<tr>
<td>AD8-AD15</td>
<td>28-21</td>
<td>I/O</td>
<td>-</td>
<td>Provide a bidirectional three-state address/data bus. The middle bits (8 to 15) of an address are multiplexed with the higher bits (8 to 15) of data.</td>
</tr>
<tr>
<td>Pin Name</td>
<td>Pin Number</td>
<td>I/O</td>
<td>Active Level</td>
<td>Function</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
<td>-----</td>
<td>--------------</td>
<td>----------</td>
</tr>
<tr>
<td>A16/ D0- D23/ D7</td>
<td>20-13</td>
<td>I/O</td>
<td>-</td>
<td>Provide a bidirectional three-state address/data bus. The higher bits (16 to 23) of an address are multiplexed with the lower bits (0 to 7) of data.</td>
</tr>
<tr>
<td>LED</td>
<td>8</td>
<td>O</td>
<td>L</td>
<td>Indicates the normal uPD72105 operation by an LED connected externally to this pin. It normally outputs a high signal but becomes low immediately after resetting or when the uPD72105 transmits data to the serial bus.</td>
</tr>
<tr>
<td>TxEN</td>
<td>6</td>
<td>O</td>
<td>H</td>
<td>Enables the serial transmitter driver. It becomes high when the uPD72105 transmits data to the serial bus.</td>
</tr>
<tr>
<td>TxC</td>
<td>4</td>
<td>I/O</td>
<td>-</td>
<td>Outputs the transmit clock generated within the uPD72105 when the DPLL mode is selected; otherwise inputs the transmit clock from an external device.</td>
</tr>
<tr>
<td>TxD</td>
<td>5</td>
<td>O</td>
<td>-</td>
<td>Outputs serial transmit data.</td>
</tr>
<tr>
<td>RXC</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td>Inputs the clock for the DPLL when DPLL mode is selected. If the internal DPLL is not selected, it inputs the receive clock, in which case the clock is generated by an external PLL.</td>
</tr>
<tr>
<td>RxD</td>
<td>3</td>
<td>I</td>
<td>-</td>
<td>Inputs serial receive data.</td>
</tr>
</tbody>
</table>
2. INTERNAL CONFIGURATION

The uPD72105 has a transmitter and a receiver that transmits/receives serial data as well as TxFIFO and RxFIFO buffers that respectively hold parallel data for the transmitter and receiver. In addition, it has a system interface to transfer data to and from the host system, a DMA controller to control data transfer, and an internal controller to control operation of each block.

2.1 Internal Controller
The internal controller interprets the commands sent from the host processor and controls the serial section and DMA controller. It also generates results which are to be transmitted to the host system after command execution has been completed.

2.2 System Interface
This is hardware for interfacing with the host system to process the I/O access and DMA transfer initiated by the host system.

2.3 DMA controller
The DMA controller processes the information (such as commands, data, and results) transferred between the host system and uPD72105 on a memory basis, using DMA. The transfer is performed in units of 4-byte blocks. Each time the transmitter or receiver processes the 4-byte data, the DMA controller issues a HLDRQ signal.

2.4 RxFIFO
This is a 20-byte receive buffer.

2.5 TxFIFO
This is a 12-byte transmit buffer.
2.6 Receiver
The receiver receives packets via the RxD pin and stores them in RxFIFO.

2.7 Transmitter
The transmitter transmits the contents of TxFIFO via the TxD pin.
3. INTERFACING WITH HOST SYSTEM

The host system can access three registers in the uPD72105: the control, status, and address FIFO registers. The control and status registers are used to control the uPD72105 without using the CRQ, INT, and CLRINT pins. The address FIFO is used to provide command addresses to the uPD72105.

Table 3.1 Register Selection (x: Don't Care)

<table>
<thead>
<tr>
<th>CS</th>
<th>WR</th>
<th>RD</th>
<th>A1</th>
<th>A0</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Writes to the control register</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>Reads from the status register</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>Specifies the address FIFO</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>Writes to the address FIFO</td>
</tr>
<tr>
<td>All others</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>Use prohibited</td>
</tr>
</tbody>
</table>

3.1 Control Register

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>MIRQ</td>
<td>CIRQ</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

CMDRQ (Command Request):
This bit has the same functions as the CRQ pin. The host system can issue a command request to the uPD72105 by setting this bit to 1. This bit is automatically reset when the command is executed.

CIRQ (Clear Interrupt Request):
This bit has the same functions as the CLRINT pin. The host system can clear the INT signal of the uPD72105 by setting this bit to 1. When the INT signal is cleared, this bit is automatically reset.
3.2 Status Register

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>IRQ</td>
<td>-</td>
<td>-</td>
<td>WNR</td>
<td>-</td>
</tr>
</tbody>
</table>

WNR (Write Not Ready):
This bit indicates that the uPD7210S is not ready when the host system is preparing to write a command address to the address FIFO. The host system must wait for this bit to become 0 before writing the next byte.

IRQ (Interrupt Request):
This bit indicates the status of the INT pin. The host system can detect an interrupt from the uPD7210S by polling this bit.

3.3 Address FIFO
This is a register that receives the memory address to which the host system has written a command and then issues a command request. The uPD7210S then receives a command from the memory area indicated by the address in the address FIFO and executes it.

When the host system issues a command request without writing an address in the address FIFO, the uPD7210S fetches a command from the memory area indicated by the default address (see NEW DEFLT ADDR command) and executes it.
4. COMMANDS

4.1 Types of Commands

The uPD72105 is provided with the following commands:

<table>
<thead>
<tr>
<th>Command name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INIT (Initialize)</td>
<td>Initializes the uPD72105.</td>
</tr>
<tr>
<td>SEND (Send)</td>
<td>Transmits a message to the other nodes through the network.</td>
</tr>
<tr>
<td>SETUP RCV (Setup Receive)</td>
<td>Activates a socket for receive.</td>
</tr>
<tr>
<td>END RCV (End Receive)</td>
<td>Inactivates the set-up socket.</td>
</tr>
<tr>
<td>WAIT RCV (Wait receive)</td>
<td>Directs a particular socket to await reception for a specific period of time.</td>
</tr>
<tr>
<td>RCV LIST (Receive List)</td>
<td>Sets up a list of the receive buffers.</td>
</tr>
<tr>
<td>ECHO (Echo)</td>
<td>Transmits an ECHO packet to the other transporter on the network.</td>
</tr>
<tr>
<td>LOOP BACK (Loop Back)</td>
<td>Transfers data stored in a particular buffer in the node to an other buffer via the uPD72105.</td>
</tr>
<tr>
<td>INIT MONIT (Initialize Monitor)</td>
<td>Starts the network monitor.</td>
</tr>
<tr>
<td>MONIT OFF (Monitor OFF)</td>
<td>Suspends transfer during network monitoring.</td>
</tr>
<tr>
<td>MONIT ON (Monitor ON)</td>
<td>Resumes suspended transfer during network monitoring.</td>
</tr>
<tr>
<td>SET PARM (Set Parameters)</td>
<td>Sets the internal parameters of the uPD72105.</td>
</tr>
<tr>
<td>GET PARM (Get Parameters)</td>
<td>Reads the internal parameters of uPD72105.</td>
</tr>
<tr>
<td>NEW CHAIN (New Chain)</td>
<td>Specifies a command address to execute the command stored there.</td>
</tr>
<tr>
<td>NEW DEFLT ADDR (New Default Address)</td>
<td>Sets the default value of a command address to execute the command stored there.</td>
</tr>
<tr>
<td>Command name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>CLR STAT (clear Statistics)</td>
<td>Clears the Tx/Rx statistics of the uPD72105.</td>
</tr>
<tr>
<td>GET STAT (Get statistics)</td>
<td>Reads the Tx/Rx statistics of the uPD72105.</td>
</tr>
</tbody>
</table>

Note: INIT MONIT, MONIT OFF and MONIT ON commands don't start operation without setting the uPD72105 in a state. The setting way will be informed to the licensees of corvus systems Inc.
4.2 Command Functions
We will now discuss the function of each command in detail. The uPD7210S fetches 12-byte data as a command and processes valid data only. Command codes are even values and when they are changed into odd values by setting to D0 bit to 1, the "command chaining function" of the uPD7210S is enabled. This function allows the uPD7210S to execute a sequence of commands on memory when the host system issues a single command request. When the uPD7210S receives a command whose command code has the D0 bit set to 1, it executes the command then the next one automatically. In this case, however, commands must continuously exist in every 12-byte area in memory. (See Fig. 4-1)

Fig. 4-1 Example of Command Chaining (3 commands)

4.2.1 INIT

The INIT command initializes the uPD7210S.
(1) Command

- **RESULT ADDRESS**
  This parameter specifies an address where the uPD72105 stores the result of a command after it has been executed.

- **HOST**
  This parameter specifies a node number and is a value of 00H to 3FH. The uPD72105 checks if the specified number has already been used by another node and indicates the result by RETURN CODE. When HOST is specified to be FFH, the uPD72105 indicates by RETURN CODE the maximum value of up to 3FH of the node number which is not used in the network (the minimum value of the node number corresponds to the value of FIXED NODES).

- **DMA PARAMETER**
  This parameter specifies the number of ready cycles for the on-chip DMA controller. One ready cycle corresponds to 1 clock cycle.

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>READY CYCLES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **BAUD RATE**
  This parameter specifies the frequency-dividing ratio for the transmit/receive clock when the on-chip DPLL is used. Where the frequency of the transmit/receive clock is x and BAUD RATE=N, the actual baud rate is x/(N+1). When the on-chip DPLL is not used, the value of this parameter has no effect.

- **DPLL PHASE**
  Bit D7 of this parameter specifies whether the on-chip DPLL is used; when it is 0, the on-chip DPLL is used, whereas when bit D7 is 1, the external DPLL is used. Also, when bit D7 is 0, the lower 7 bits are used to specify an operational phase of the DPLL in relation to the data received. These 7 bits are normally set to 02H. When bit D7 is 1, the value of these bits have no effect.
DATA FORMAT
When commands or results are transferred between the host system and the upD72105, this byte specifies which byte of a value (such as an address or data length) comprising 2 or more bytes should be processed first, the lower or higher byte. The default value immediately after resetting is 0; however, the RESULT ADDRESS of the first INIT command itself after reset will be defined by this parameter.

<table>
<thead>
<tr>
<th>2-byte data</th>
<th>DATA FORMAT=0</th>
<th>DATA FORMAT≠0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Form lower to higher byte</td>
<td>From higher to lower byte</td>
</tr>
<tr>
<td>3-byte data</td>
<td>From lower, middle to higher byte</td>
<td>From higher, middle to lower byte</td>
</tr>
</tbody>
</table>

PROTOCOL
This parameter specifies the protocol according to which packets are transmitted. When the value of this parameter is 01H, the Omninet I protocol is specified, when it is 02H, the Omninet II protocol is specified. Setting to the Omninet I protocol, 16-bit CRC is selected. Setting to the Omninet II protocol, 32-bit CRC is selected.

FIXED NODES
This parameter specifies the number of nodes using fixed host addresses. It is valid when HOST=FFH. For example, when HOST=FFH and FIXED NODES=0FH, 0 to 14 are assigned to the fixed addresses. The node number is then determined between 15 to 63.
The fixed host address can be used as an address of common node such as a disk server.
(2) Result

- **RETURN CODE**
  
  **00H to 3FH**: When the RETURN CODE parameter has any of these values, it indicates that initialization has been successfully accomplished and a code is assigned as a node number.
  
  **86H**: This value indicates that the value of the parameter HOST is invalid (40H to FEH).
  
  **89H**: This value indicates that the node number specified by the parameter HOST has already been used by the other nodes.

### 4.2.2 SEND

<table>
<thead>
<tr>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>COMMAND CODE = 40H</td>
</tr>
<tr>
<td>2</td>
<td>RESULT ADDRESS</td>
</tr>
<tr>
<td>3</td>
<td>SOCKET</td>
</tr>
<tr>
<td>4</td>
<td>DATA ADDRESS</td>
</tr>
<tr>
<td>5</td>
<td>DATA LENGTH</td>
</tr>
<tr>
<td>6</td>
<td>CONTROL LENGTH</td>
</tr>
<tr>
<td>7</td>
<td>DESTINATION</td>
</tr>
<tr>
<td>8</td>
<td>RETURN CODE</td>
</tr>
<tr>
<td>9</td>
<td>NOT USED</td>
</tr>
<tr>
<td>10</td>
<td>USER CONTROL</td>
</tr>
<tr>
<td>11</td>
<td>(0-FFH bytes)</td>
</tr>
</tbody>
</table>

The SEND command transmits messages from memory to the other nodes through the network.

(1) Command

- **SOCKET**

  This parameter specifies the socket number of the destination. As a socket number, 80H, 90H, A0H, or B0H can be specified.
DATA ADDRESS
This parameter specifies the first address of memory that holds the data to be transmitted.

DATA LENGTH
This parameter specifies the number of bytes in the data to be transmitted.
- Omnitent I: 0 to 7FFH
- Omnitent II: 0 to 800H

CONTROL LENGTH
This parameter specifies the number of bytes of user control the host system created in the user message. Where the value of SOCKET is 80H or 90H, the number of bytes is 0.

DESTINATION
This parameter specifies the node number of the destination. 00H to 3FH are the regular values, and FFH indicates the broadcast message. All other transporters receive a message addressed DESTINATION=FFH, but do not return acknowledgement.

Result

RETURN CODE
00H : Indicates that transmission has been successfully accomplished without retransmission.
01H to 0AH: Indicates the number of retransmission attempts made before successful transmission was accomplished.
80H : Indicates that retransmission was attempted the maximum allowable number of times but no response was obtained.
81H : Indicates that the transfer data length was longer than the buffer length provided in the destination.
82H : Indicates that the socket has not been set up at the destination.
83H : Indicates that the value of CONTROL LENGTH specified did not agree with the value at the destination.
84H : Indicates that the parameter SOCKET was invalid, i.e., the value of the parameter SOCKET was other than 80H, 90H, A0H, and B0H.

86H : Indicates that the parameter DESTINATION was invalid, i.e., the value of the parameter was other than 00H to 3FH and FFH).

87H : Indicates that the "no empty message" buffer is available for the socket at the destination.

8AH : Indicates that the command was not accepted, because the chip has been in the network monitor mode.

" USER CONTROL 

The user control, which the host system provides in packet information, should be written in addresses higher than (RESULT ADDRESS+4) prior to transmission.

4.2.3 SETUP RCV

The SETUP RCV command readies a socket to receive message packets.
(1) Command
- SOCKET
  This parameter specifies the socket number that enables reception.
- DATA ADDRESS
  This parameter specifies the first address of a buffer in memory in which the received data are to be stored.
- DATA LENGTH
  This parameter specifies the number of bytes in the receive buffer to be provided in the addresses higher than DATA ADDRESS:
  - Omninet I: 0 to 7FFH
  - Omninet II: 0 to 800H
- CONTROL LENGTH
  This parameter specifies the number of bytes in the user control to be received. When the value of SOCKET is 80H or 90H, it is 00H.

(2) Result
- RETURN CODE
  - FEH: Indicates that the receive socket has been successfully set up.
  - 84H: Indicates that the parameter SOCKET was invalid.
  - 85H: Indicates that the socket specified by the parameter SOCKET is being used.
  - 8AH: Indicates that the command was not accepted, because the chip has been in the network monitor mode.
  - 00H: Indicates that the data has been successfully received.

The SETUP RCV command first sets a value to RETURN CODE immediately after the command has been issued to indicate whether the socket has been set up. If the socket has been successfully set up, the command sets the value 00H to RETURN ODE when a message packet is received later. The uPD7210S generates an interrupt whenever it sets a value to RETURN CODE.
When a message packet is received (RETURN CODE=00H), this parameter is set as a result along with LENGTH OF DATA RECEIVED and USER CONTROL to indicate the number of the node that transmitted the packet.

LENGTH OF DATA RECEIVED
This parameter indicates the number of bytes in the user data of the received packet.

USER CONTROL
This area stores the user control of the received packet.

4.2.4 END RCV

The END RCV command causes the socket that has been set up by the SETUP RCV or RCV LIST command to return to the inactive state.

(1) Command
- SOCKET
  Specifies the number of the socket that is to be inactivated.

(2) Result
- RETURN CODE
  84H : Indicates that the parameter SOCKET was invalid.
  00H : Indicates that the socket has been inactivated.
  8AH : Indicates that the command was not accepted, because the chip has been in the network monitor mode.
4.2.5 WAIT RCV

The WAIT RCV command specifies the time during which the socket set up by the SETUP RCV command is to await reception of a message packet.

1. Command
   a. SOCKET
      This parameter specifies the number of the socket for which the wait time for receiving a packet is specified.
   b. TIME
      This parameter specifies the time for receiving a packet.

<table>
<thead>
<tr>
<th>TIME</th>
<th>Wait time [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(system clock: 8 MHz)</td>
</tr>
<tr>
<td>01</td>
<td>1x100</td>
</tr>
<tr>
<td>02</td>
<td>2x100</td>
</tr>
<tr>
<td>03</td>
<td>3x100</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>FF</td>
<td>25x100</td>
</tr>
</tbody>
</table>
Where the value of TIME is 00H, the uPD7210S enters the busy wait state and processes no command until it receives a message packet. With this function, it is possible to suspend execution of a sequence of commands by specifying the command chaining function with a series of commands following the WAIT RCV command, and executing the WAIT RCV command with TIME specified to be 00H.

(2) Result

- **RETURN CODE**

  00H : Indicates that the wait time has been set to the specified socket.
  84H : Indicates that the parameter SOCKET was invalid.
  88H : Indicates that the specified socket has not been set up.
  8AH : Indicates that the command was not accepted, because the chip has been in the network monitor mode.
  92H : Indicates that the RCV LIST command is being executed on the specified socket. The WAIT RCV command cannot be executed on a socket that has been set up by the RCV LIST command.

The subsequent RETURN CODE is set to the address specified by RESULT ADDRESS of the SETUP RCV command that has set up the socket.

  00H : Indicates that a packet has been received within the time specified by TIME.
  90H : Indicates that a packet has not been received within the time specified by TIME (i.e., time-out). In this case, the specified socket becomes inactive.
4.2.6 RCV LIST

The RCV LIST command sets up a list of message buffers (message descriptor) to allow reception by sockets 80H and 90H. Only socket numbers 80H and 90H are supported by this command. The list provided by the host system when this command is executed is shown in Fig. 4-2. The message descriptor consists of a message list indicating the state of the buffer in which the received message is to be stored and control records which manage the buffer.

(1) Command
- **MESSAGE LIST ADDRESS**
  This parameter specifies the first address of the message descriptor.
- **SOCKET**
  This parameter specifies the number of the socket that is set up to receive packets.
- **BUFFER LENGTH**
  This parameter specifies the number of bytes in each message buffer.

(2) Control Record
- **SOCKET STATE**
  Immediately after a command has been executed, RETURN CODE is first set. Afterward, this parameter indicates the presence of a buffer storing unprocessed received messages.
RETURN CODE

FEH : Indicates that the receive list has been set up and that reception is enabled.

84H : Indicates that parameter SOCKET is invalid (i.e., the value of the parameter is other than 80H and 90H).

85H : Indicates that the specified socket is being used.

8AH : Indicates that the command was not accepted because the chip has been in the network monitor mode.

Buffer state

00H : Indicates that the received message has been stored after the last interrupt was generated.

FDH : Indicates that the received messages are stored in all the buffers.

FCH : Indicates that the host system has completed processing of all the messages.

FIRST FREE AND LAST FREE

These parameters indicate the first and last indexes of FREE LIST, respectively. The "FREE LIST" is a message list of an empty message buffer.

FIRST RECEIVED and LAST RECEIVED

These parameters indicate the first and last indexes of RECEIVED LIST, respectively. When the value of LAST RECEIVED is 00H, it indicates that no message has been received yet. When the value of FIRST RECEIVED equals that of FIRST FREE, RECEIVED LIST does not exist. "RECEIVED LIST" is a message list of the buffers storing unprocessed received messages.

(3) Message List

NEXT

This parameter indicates the index of the next message list. When its value is 00H, it indicates the last list.

DATA BUFFER ADDRESS

This parameter indicates the first address of the message buffer each message list supports.
RECORD CODE

FFH : Indicates that the message buffer is empty.
00H : Indicates that the received message is stored in the message buffer.

The host system rewrites FFH to RECORD CODE when a command is issued or when the received message in the message buffer is processed.

SOURCE
This parameter indicates the number of the node that transmitted the received packet. This node number is not set by the host system during command execution. It is written by the uPD72105 after the packet has been received.

ACTUAL LENGTH
This parameter indicates the number of bytes in a received message. It is set by the uPD72105 after the packet has been received.

When the RCV LIST command is issued, the first message list is set up. The uPD72105 then removes the set up of a message list one by one each time receiving a packet, and stores the received message to a buffer according to the list. An interrupt is requested each time a packet is received.

On receiving a packet, the uPD72105 stores a message in the buffer of the message list indicated by FIRST FREE, sets the RECORD CODE in the message list to 00H, and also sets the SOURCE and ACTUAL LENGTH CODE to specific values. It then sets FIRST FREE to an index of the next message list and LAST RECEIVED to that of the current message list. The uPD72105 also sets the SOCKET STATE to 00H and generates an interrupt. Figs. 4-3 and 4-4 show updating on the message descriptor when the first and i-th packets are received, respectively.

If a message is stored in the last buffer (NEXT=00H) of FREE LIST, the uPD72105 sets the SOCKET STATE to FDH and FIRST FREE to 00H.
The host system, on the other hand, checks the SOCKET STATE on acknowledging an interrupt and processes received messages if it is either OOH or FDH. The host system first checks that FIRST RECEIVED does equal FIRST FREE and that FIRST RECEIVED does not equal OOH (RECEIVED LIST is present). It then fetches received message from the buffer indicated by FIRST RECEIVED. After that, the host system rewrites FIRST RECEIVED to the NEXT value in the current message list and sets the NEXT to OOH and the RECORD CODE to FFH. It then checks FIRST FREE further. If FIRST FREE does not equal OOH (FREE LIST is present), the host system sets NEXT and LAST FREE of the message list, indicated by LAST FREE to the index of the current message list. If FIRST FREE equals OOH (FREE LIST is not present), the host system sets FIRST FREE and LAST FREE to the index of the current message list. These procedures mean to add an empty buffer to the end of the FREE LIST. Fig. 4-5 shows updating on the list when the first message is processed after the i-th packet has been received. The host system continues these processes until RECEIVED LIST runs out (FIRST RECEIVED does not equal FIRST FREE) and sets the SOCKET STATE to FCH.
Fig. 4-2 RCV LIST Command List (created by the host system)
Fig. 4-3 Updated List on Receiving the First Packet (by the chip)
Fig. 4-4 Updated List on Receiving the i-th Packet (by the chip)
4-5 Updated list on Processing the First Message
(by the host system)
The ECHO command transmits the ECHO packet to the other nodes on the network. With this command, the host system can learn whether the destination node of the ECHO packet exists on the network.

(1) Command

- DESTINATION
  This parameter specifies the node number of the destination.
- MAX TRY
  This parameter specifies the maximum allowable number of retransmission attempts (0 to 127). It is generally specified as a small value such as 01H or 02H.

(2) Result

- RETURN CODE
  00H  : Indicates that the presence of the destination node on the network was acknowledged by transmitting the first ECHO packet (i.e., acknowledgment packet was received).
  01H to 7FH: Indicates the number of retransmission attempts made before the acknowledgment packet was received.
  80H  : Indicates that the acknowledgment packet was not received after retransmission attempts had been made the number of times specified by MAX TRY.
86H : Indicates that the parameter DESTINATION was invalid (over 40H).
8AH : Indicates that the command was not accepted, because the chip has been in the network monitor mode.

4.2.8 LOOP BACK

The LOOP BACK command transfers the data held by a buffer in memory via the uPD72105 to another buffer in the node. The host system may check the data for changes during transfer by comparing the contents of both the buffers.

(1) Command

° CRC SIZE
This parameter specifies the number of bits in the CRC code. When its value is 02H, the CRC code consists of 16 bits and when it is 04H, 32 bits.

° SOURCE DATA ADDRESS
This parameter specifies the first address of the buffer containing the data to be transferred.

° DATA LENGTH
This parameter specifies the number of bytes of data to be transferred.
(2) Result

* RETURN CODE
  00H : Indicates that the loop back function completed normally.
  8AH : Indicates that the command was not accepted, because the chip has been in the network monitor mode.

* LOOP BACK DATA
These bytes are the data transferred by the loop back function.

* CRC
These bytes are the CRC code generated with transferred data.

4.2.9 INIT MONIT

The INIT MONIT command starts network monitoring to receive all the packets on the network (including those transmitted to the other nodes), and then transfers a packet information and an additional information of each packet to the specified ring buffer. Note that the received packets also remain on the network, not influencing the normal receive operation.
(1) Command

- CRC/INT FLAG
  Bit D0 (CRC FLAG):
  When this bit is 1, the packet information for a packet that caused a CRC error is also transferred. When it is 0, however, the packet information is not transferred.
  Bit D1 (INT FLAG):
  When this bit is 1, an interrupt is generated each time a packet information and an additional information are transferred to the buffer at the last entry in the ring buffer.

- RING BUFFER ADDRESS
  Specifies the first address of the ring buffer in which the packet information and additional information are to be stored.

- ENTRIES
  Specifies the number of entries (buffers) in the ring buffer.

- ENTRY SIZE
  Specifies the number of bytes in each buffer of the ring buffer. The number of bytes from a packet defined by [(ENTRY SIZE)-5 bytes] is transferred to the buffer with the following additional information contained in the remaining 5 bytes:
  1st byte : Indicates the sequence of the packet received during command execution.
  2nd to 4th bytes: Indicates the count value of the internal counter after the start of command execution until reception of the packet.
  5th byte : Indicates whether a CRC error occurred. The value 00H indicates that an error has not occurred, whereas any other value indicates occurrence of an error.
(2) Result

Result:

* RETURN CODE

00H : Indicates that the command has been accepted.

When the μPD72105 accepts an INIT MONIT command, it becomes unable to perform ordinary communications, and can only accept the commands INIT MONIT, MONIT OFF, MONIT ON, and INIT. To release the network monitor mode, the INIT command must be executed.

4.2.10 MONIT OFF

![](image)

The MONIT OFF command suspends DMA transfer to the ring buffer while the μPD72105 is monitoring the network.

(1) Result

* RETURN CODE

00H : Indicates that the command has been accepted.

8AH : Indicates that the command was not accepted, because the chip has not been in the network monitor mode.

4.2.11 MONIT ON

![](image)
The MONIT ON command resumes DMA transfer to the ring buffer suspended by the uPD72105 while it is monitoring the network.

(1) Result

- **RETURN CODE**
  - **00H**: Indicates that the command has been accepted.
  - **8AH**: Indicates that the command was not accepted, because the chip has not been in the network monitor mode.

### 4.2.12 SET PARM

<table>
<thead>
<tr>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>COMMAND = 1EH</td>
</tr>
<tr>
<td>2</td>
<td>RESULT ADDRESS</td>
</tr>
<tr>
<td>3</td>
<td>MAX WAIT</td>
</tr>
<tr>
<td>4</td>
<td>TRAILING FLAGS</td>
</tr>
<tr>
<td>5</td>
<td>MAX RETRY</td>
</tr>
<tr>
<td>6</td>
<td>LEADING FLAGS</td>
</tr>
<tr>
<td>7</td>
<td>PROTOCOL</td>
</tr>
<tr>
<td>8</td>
<td>OVERLAP</td>
</tr>
<tr>
<td>9</td>
<td>RETURN CODE</td>
</tr>
</tbody>
</table>

The SET PARM command modifies the internal parameters of the uPD72105.
(1) Command

- MAX WAIT
  This parameter specifies the maximum random wait time before transmission.
  The following values are acceptable, and the default value is 0FH.

<table>
<thead>
<tr>
<th>MAX WAIT</th>
<th>Maximum wait time (us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>01H</td>
<td>5</td>
</tr>
<tr>
<td>03</td>
<td>15</td>
</tr>
<tr>
<td>07</td>
<td>35</td>
</tr>
<tr>
<td>0F</td>
<td>75</td>
</tr>
<tr>
<td>1F</td>
<td>155</td>
</tr>
<tr>
<td>3F</td>
<td>315</td>
</tr>
<tr>
<td>7F</td>
<td>635</td>
</tr>
<tr>
<td>FF</td>
<td>1275</td>
</tr>
</tbody>
</table>

- TRAILING FLAGS
  The parameter specifies the number of bytes in the TRAILING FLAG of a packet. The default value is 08H.

- MAX RETRY
  This parameter specifies the maximum allowable number of retransmission attempts. The default value is 0AH.

- LEADING FLAGS
  This parameter specifies the number of bytes in the LEADING FLAG of a packet. The default value is 0AH.

- PROTOCOL
  This parameter has the same functions as the parameter PROTOCOL of the INIT command.

- OVERLAP
  This parameter specifies the number of bits that overlap timewise between the TRAILING FLAG of the message packet being received and LEADING FLAG of the acknowledge packet to be transmitted in response to the packet being received.
° TRYINT
This parameter specifies the minimum value of the retransmission intervals.

<table>
<thead>
<tr>
<th>TRYINT</th>
<th>Minimum interval (us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>Use prohibited</td>
</tr>
<tr>
<td>01</td>
<td>3</td>
</tr>
<tr>
<td>02</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>FE</td>
<td>762</td>
</tr>
<tr>
<td>FF</td>
<td>765</td>
</tr>
</tbody>
</table>

° HOST
The parameter HOST of this command has no meaning.
The HOST which was determined by INIT command can not be changed with this parameter.

(2) Result
° RETURN CODE
00H : Indicates that the command has been accepted.
8AH : Indicates that the command was not accepted, because the chip has been in the network monitor mode.
The GET PARM command reads the internal parameters of the uPD72105. To partially modify the internal parameters, the parameters of the SET PARM command can be created by first executing the GET PARAM command and then modifying the result.

(1) Result

- **RETURN CODE**
  - 00H: Indicates that the command has been accepted.
  - 8AH: Indicates that the command was not accepted, because the chip has been in the network monitor mode.

4.2.14 NEW CHAIN

Command

<table>
<thead>
<tr>
<th>Command Code = 04H</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW CHAIN ADDRESS</td>
</tr>
</tbody>
</table>
The NEW CHAIN command executes the command stored in the address specified by NEW CHAIN ADDRESS. The uPD72105 does not generate an interrupt for this command, but does so after execution of the command in the specified address.

4.2.15 NEW DEFLT ADDR

<table>
<thead>
<tr>
<th>Command</th>
<th>No result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>COMMAND CODE = 06H</td>
</tr>
<tr>
<td>2</td>
<td>NEW DEFAULT COMMAND ADDRESS</td>
</tr>
</tbody>
</table>

The NEW DEFLT ADDR command sets a default value of the command address from which the command is fetched when the address FIFO is empty. This command address is 0 immediately after resetting.

On receiving the command, the uPD72105 executes the command specified by NEW DEFAULT COMMAND ADDRESS. The uPD72105 does not generate an interrupt due to this command, but does after execution of the command in the specified address.

4.2.16 CLR STAT

<table>
<thead>
<tr>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>COMMAND CODE = 26H</td>
</tr>
<tr>
<td>2</td>
<td>RESULT ADDRESS</td>
</tr>
<tr>
<td>3</td>
<td>RETURN CODE</td>
</tr>
</tbody>
</table>

The CLR STAT command clears the contents of event counters to initialize the counters. The counters manage the transmit/receive statistics.

1. Result
   - RETURN CODE
     - 00H : Indicates that the command has been accepted.
     - 8AH : Indicates that the command was not accepted, because the chip has been in the network monitor mode.
The GET STAT command reads the contents of event counters to store them in a memory area following to the RETURN CODE.

1. Result
   1. RETURN CODE
      1. RECEIVE ERROR*
      2. RECEIVE OK
      3. SEND ERROR
      4. PACKETS SENT
      5. ABORT
      6. CRC ERROR
      7. OVERRUN ERROR

(1) Result
   ° RETURN CODE
   00H : Indicates that the command has been accepted.
   8AH : Indicates that the command was not accepted, because the chip has been in the network monitor mode.
- **RECEIVE ERROR**
  The RECEIVE ERROR indicates the count of errors which have occurred on receiving packets. The value is the total of ABORT, CRC ERROR and OVERRUN ERROR.
- **RECEIVE OK**
  The RECEIVE OK indicates the count of receiving packets normally.
- **SEND ERROR**
  The SEND ERROR indicates the count of Tx underrun errors which have occurred to send aborts on transmitting packets.
- **PACKETS SENT**
  The PACKETS SENT indicates the count of packet transmissions. It is the total of normal and abnormal transmissions.
- **ABORT**
  The ABORT indicates the count of receiving aborts.
- **CRC ERROR**
  The CRC ERROR indicates the count of CRC errors which have occurred on receiving packets. The polynomials of CRC are as follows.

  16 bits: \( x^{16} + x^{12} + x^5 + 1 \)
  32 bits: \( x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1 \)

  The initial value of CRC is 1. The calculation is done for the packet information. The result of calculation is inverted the bit status, 1 or 0, to transmit the MSB first on transmitting. The result of calculation is compared with \( LSB_{F0B8}^{MSB} \) (for 16-bit) or \( LSB_{DEBB20E3}^{MSB} \) (for 32-bit) on receiving, and the receiving is regarded as CRC error.

<table>
<thead>
<tr>
<th>Leading Flags</th>
<th>Packet Information</th>
<th>CRC</th>
<th>Trailing Flags</th>
</tr>
</thead>
</table>

**CRC calculation**

- **OVERRUN ERROR**
  The OVERRUN ERROR indicates the count of Rx overrun errors which have occurred on receiving packets.
5. CONTROL

The host system needs to have additional data in memory for some commands to issue them to the uPD72105. These data include: a message descriptor and buffer for the RCV LIST command, a buffer for the SETUP RCV command, and transmit data for the SEND command.

A control flowchart of the uPD72105 is shown in Fig. 5-1. Prior to issuing a command request to the uPD72105, the host system constructs a command table in memory and writes FFH as the RETURN CODE in the area of the table indicated by RESULT ADDRESS. Then it writes the first address of the memory on which the command table was created to the address FIFO of the uPD72105.

The command address writing procedure is shown in Fig. 5-2. In writing data to the address FIFO, the host system always checks whether the WNR(D1) bit of the status register is set to 1. Setting the command address is completed by writing 3-byte data.

If a command request is issued without setting the address FIFO, the uPD72105 fetches a command from the default address (see NEW DEFLT ADDR COMMAND) and executes it.

After setting the command address, the host system issues a command request (the CRQ signal is "H" or the CMDRQ(D0) bit in the control register is 1) and then waits for a command completion interrupt to be generated by the uPD72105. When the host system detects the command completion interrupt (the INT signal is "H" or the IRQ(D4) bit of the status register is 1), the host system resets this interrupt (CLRINT signal is "H" or CIRQ(D3) bit of the control register is 1).

In this procedure, the host system completes controlling the uPD72105 and then reads the RETURN CODE stored in RESULT ADDRESS.

However, when a command is to be executed while setting-up a receive socket, an interrupt of command execution completion and that of packet receive may simultaneously be generated. In this case, the host system fails to detect the packet receive
interrupt because the receive interrupt has been also reset after servicing the command completion interrupt. Therefore, the host system may service only the command completion interrupt, and fails to service the receive interrupt. To avoid this, when a command is executed during the setting-up of the receive socket, the process can be controlled as follows: when the uPD72105 generates an interrupt, the RETURN CODE of the received command is first checked and if it is other than FFH or FEH, the receive completion interrupt is serviced. Subsequently, the service routine for the command completion interrupt is executed without waiting for the interrupt to occur. By doing so, depending on the timing of an interrupt, the host system may start servicing the next interrupt immediately before it is generated. This results in no causes of interrupt remaining to be serviced even when an interrupt has been detected. In such cases, the host system must ignore the interrupt.
If command address is omitted, the default address becomes effective.

Fig. 5-1 CSMAC Control Flowchart
Specifying odd,... FIFO
Writing comm.nd addr

Three bytes of higher, middle, and lower bits (order is specified by the INT command)

Fig. 5-2 Command Address Writing
Fig. 6-1 System Configuration Example I (Memory Mapped I/O)
Fig. 6-2 System Configuration Example II
Fig. 6.3  Example for serial interface

Fig. 6.4  NRZI Data Format

Bit call

| Data bit | 1 0 0 0 0 1 0 1 0 1 0 | 1 1 1 |
| NRZI     |  |  |  |  |  |  |  |  |  |  |  |
**Packet Format**

<table>
<thead>
<tr>
<th>Message Packet</th>
<th>Echo Packet</th>
<th>Acknowledgement Packet</th>
<th>Sync Packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leading Flags</td>
<td>Leading Flags</td>
<td>Leading Flags</td>
<td>Leading Flags</td>
</tr>
<tr>
<td>Message Header</td>
<td>Message Header</td>
<td>Message Header</td>
<td>Message Header</td>
</tr>
<tr>
<td>User Control</td>
<td>User Control</td>
<td>User Control</td>
<td>User Control</td>
</tr>
<tr>
<td>User Data</td>
<td>User Data</td>
<td>User Data</td>
<td>User Data</td>
</tr>
<tr>
<td>CRC</td>
<td>CRC</td>
<td>CRC</td>
<td>CRC</td>
</tr>
<tr>
<td>Trailing Flags</td>
<td>Trailing Flags</td>
<td>Trailing Flags</td>
<td>Trailing Flags</td>
</tr>
</tbody>
</table>

<p>| | | | | | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address</td>
<td>Source Address</td>
<td>Validation</td>
<td>Destination Socket</td>
<td>Retry</td>
<td>Parity</td>
<td>User Data Length</td>
<td>User Control Length</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>“A5”</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>“5A”</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination Address</td>
<td>Source Address</td>
<td>Validation</td>
<td>Destination Socket</td>
<td>Retry</td>
<td>Parity</td>
<td>User Data Length</td>
<td>User Control Length</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>“A5”</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>“5A”</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination Address</td>
<td>Source Address</td>
<td>Validation</td>
<td>Destination Socket</td>
<td>Retry</td>
<td>Parity</td>
<td>User Data Length</td>
<td>User Control Length</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>“A5”</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>“5A”</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**I = OMNINET I protocol, II = OMNINET II protocol**
μPD72105
LOCAL AREA NETWORK CONTROLLER

TARGET SPECS
# uPD72105 AC/DC Target spec.

## ABSOLUTE MAXIMUM RATINGS (Ta=25°C)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>Ratings</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Supply Voltage</td>
<td>VDD</td>
<td></td>
<td>-0.5 〜 +7.0</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>VI</td>
<td></td>
<td>-0.5 〜 VDD+0.3</td>
<td>V</td>
</tr>
<tr>
<td>Output Voltage</td>
<td>VO</td>
<td></td>
<td>-0.5 〜 VDD+0.3</td>
<td>V</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>Topt</td>
<td></td>
<td>-40 〜 +85</td>
<td>°C</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>Tstg</td>
<td></td>
<td>-40 〜 +125</td>
<td>°C</td>
</tr>
</tbody>
</table>

## DC CHARACTERISTICS (Ta=-40 〜+85°C, VDD=5V±10%)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>MIN</th>
<th>TYP</th>
<th>MAX</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Low Voltage</td>
<td>VILC</td>
<td>CLK Pin</td>
<td>-0.5</td>
<td>+0.8</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VILR</td>
<td>RXC when using on-chip DPLL</td>
<td>-0.5</td>
<td>+0.4</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VIL</td>
<td>Others</td>
<td>-0.5</td>
<td>+0.8</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Input High Voltage</td>
<td>VIHC</td>
<td>CLK Pin, TEST Pin</td>
<td>+3.3</td>
<td>VDD+0.3</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VIHR</td>
<td>RXC when using on-chip DPLL</td>
<td>+2.4</td>
<td>VDD+0.3</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VIH</td>
<td>Others</td>
<td>+2.2</td>
<td>VDD+0.3</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Output Voltage Low</td>
<td>VOL</td>
<td>IOL=2.5mA</td>
<td></td>
<td>+0.4</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Output Voltage High</td>
<td>VOH</td>
<td>IOH=-400ua</td>
<td>0.7×VDD</td>
<td></td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Supply Current</td>
<td>IDD1</td>
<td>When operating*1</td>
<td>25</td>
<td>40</td>
<td>mA</td>
<td></td>
</tr>
<tr>
<td>Input Leakage Current</td>
<td>ILI</td>
<td>OV≤VIN ≤ VDD</td>
<td>+10</td>
<td>uA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output Leakage Current</td>
<td>ILO</td>
<td>OV≤VOUT ≤ VDD</td>
<td>+10</td>
<td>uA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*1 system clock = 8MHz
DPLL clock = 40MHz

Fix B/W pin to "1" or "0".
### uPD72105

**AC CHARACTERISTIC (Ta=-40 +85°C, VDD=5V±10%)**

**BUS-MASTER MODE**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>MIN</th>
<th>MAX</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLK Cycle Time</td>
<td>TCYK</td>
<td></td>
<td>125</td>
<td>1000</td>
<td>nS</td>
</tr>
<tr>
<td>CLK Active Low</td>
<td>TKKL</td>
<td></td>
<td>50</td>
<td>50</td>
<td>nS</td>
</tr>
<tr>
<td>CLK Active High</td>
<td>TKKH</td>
<td></td>
<td>50</td>
<td>50</td>
<td>nS</td>
</tr>
<tr>
<td>CLK Rise Time</td>
<td>TKR</td>
<td>1.5-3.0V</td>
<td></td>
<td>10</td>
<td>nS</td>
</tr>
<tr>
<td>CLK Fall Time</td>
<td>TKF</td>
<td>3.0-1.5V</td>
<td></td>
<td>10</td>
<td>nS</td>
</tr>
<tr>
<td>HLDRO↑ Delay Time from CLK↓</td>
<td>TDHQH</td>
<td></td>
<td></td>
<td>100</td>
<td>nS</td>
</tr>
<tr>
<td>HLDRO↑ Delay Time from CLK↑</td>
<td>TDQHL</td>
<td></td>
<td></td>
<td>100</td>
<td>nS</td>
</tr>
<tr>
<td>HLDAK Setup Time to CLK↑</td>
<td>TSHA</td>
<td></td>
<td></td>
<td>35</td>
<td>nS</td>
</tr>
<tr>
<td>AEN↑ Delay Time from CLK↓</td>
<td>TDAEH</td>
<td></td>
<td></td>
<td>100</td>
<td>nS</td>
</tr>
<tr>
<td>AEN↑ Delay Time from CLK↑</td>
<td>TDAEL</td>
<td></td>
<td></td>
<td>100</td>
<td>nS</td>
</tr>
<tr>
<td>ASTB↑ Delay Time from CLK↑</td>
<td>TDSTH</td>
<td></td>
<td></td>
<td>70</td>
<td>nS</td>
</tr>
<tr>
<td>ASTB Pulse Width</td>
<td>TSTSTH</td>
<td>TKKH-15</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>ASTB↓ Delay Time from CLK↓</td>
<td>TDSTL</td>
<td></td>
<td></td>
<td>100</td>
<td>nS</td>
</tr>
<tr>
<td>ADR/UBE/RD/WR Delay Time from CLK↑</td>
<td>TDA</td>
<td></td>
<td></td>
<td>100</td>
<td>nS</td>
</tr>
<tr>
<td>ADR/UBE/RD/WR Float Delay from CLK↑</td>
<td>TFA</td>
<td></td>
<td></td>
<td>70</td>
<td>nS</td>
</tr>
<tr>
<td>ADR Setup Time to ASTB↓</td>
<td>TSAST</td>
<td>TKKH-35</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>ADR Hold Time from ASTB↓</td>
<td>THSTA</td>
<td>TKKH-20</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RD↓ Delay Time from ADR Float</td>
<td>TDAR</td>
<td>TKKH-30</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RD↓ Delay Time from CLK↓</td>
<td>TDRL</td>
<td></td>
<td></td>
<td>70</td>
<td>nS</td>
</tr>
<tr>
<td>RD Pulse Width</td>
<td>TRRL2</td>
<td>1.5TCYK-50</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RD↑ Delay Time from CLK↑</td>
<td>TDRH</td>
<td></td>
<td></td>
<td>70</td>
<td>nS</td>
</tr>
<tr>
<td>DATA Setup Time to RD↑</td>
<td>TSDR</td>
<td></td>
<td>70</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>DATA Hold Time from RD↑</td>
<td>THRD</td>
<td></td>
<td>0</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>WR↓ Delay Time from CLK↓</td>
<td>TDWL</td>
<td></td>
<td></td>
<td>70</td>
<td>nS</td>
</tr>
<tr>
<td>WR Pulse Width</td>
<td>TWNL2</td>
<td>1.5TCYK-50</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>WR↑ Delay Time from CLK↑</td>
<td>TDWH</td>
<td></td>
<td></td>
<td>70</td>
<td>nS</td>
</tr>
<tr>
<td>READY Setup Time to CLK↑</td>
<td>TSRY</td>
<td></td>
<td>35</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>READY Hold Time from CLK↑</td>
<td>THRY</td>
<td></td>
<td>20</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>Parameter</td>
<td>Symbol</td>
<td>Test Conditions</td>
<td>MIN</td>
<td>MAX</td>
<td>Units</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------</td>
<td>-----------------</td>
<td>-----</td>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td>CLRINT Pulse Width</td>
<td>TCLKLH</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>INT↑ Delay Time from CLKT</td>
<td>TDIH</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>INT↓ Delay Time from CLRINT↓</td>
<td>TDIL</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>LED↓ Delay Time from CLKT</td>
<td>TDLH</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>LED↑ Delay Time from CLKT</td>
<td>TDLH</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>CRQ Pulse Width</td>
<td>TCRCHR</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
</tbody>
</table>

**BUS-SLAVE MODE**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>MIN</th>
<th>MAX</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>WR Pulse Width</td>
<td>TWRL</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>CS Hold Time from WR↑</td>
<td>THNCS</td>
<td>0</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>ADR/UBE/CS Setup Time to WR↑</td>
<td>TSAN</td>
<td>0</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>ADR/UBE Hold Time from WR↑</td>
<td>THWA</td>
<td>0</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>DATA Setup Time to WR↑</td>
<td>TSDU</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>DATA Hold Time from WR↑</td>
<td>THWD</td>
<td>20</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RD Pulse Width</td>
<td>TRRL</td>
<td>150</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>ADR/CS Setup Time to RD↓</td>
<td>TSAR</td>
<td>35</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>ADR/CS Hold Time from RD↑</td>
<td>THRA</td>
<td>0</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>DATA Delay Time from RD↓</td>
<td>TDRD</td>
<td>120</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>DATA Float Delay from RD↑</td>
<td>TFRD</td>
<td>10</td>
<td>100</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RESET Pulse Width</td>
<td>TRSTL</td>
<td>7TCYK</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>VDD Setup Time to RESET↑</td>
<td>TSVDD</td>
<td>1000</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>1st WR/RD from RESET↑</td>
<td>TSYWR</td>
<td>2TCYK</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>Recovery Time from WR/RD</td>
<td>TRVWR</td>
<td>200</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>High Setup Time to HLDAK↑</td>
<td>TSWR</td>
<td>-20</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>High Hold Time from AEN↓</td>
<td>THWR</td>
<td>100</td>
<td></td>
<td></td>
<td>nS</td>
</tr>
</tbody>
</table>

* 2 Input levels for AC test are
  2.4V (as "1") and
  0.4V (as "0").

Test points are
  2.2V (as "1") and
  0.8V (as "0").
Serial Part

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Conditions</th>
<th>MIN</th>
<th>MAX</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>TXC/RXC Cycle Time</td>
<td>TCYS</td>
<td></td>
<td>250</td>
<td>2000</td>
<td>nS</td>
</tr>
<tr>
<td>TXC/RXC Active Low</td>
<td>TSSL</td>
<td></td>
<td>110</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TXC/RXC Active High</td>
<td>TSSH</td>
<td></td>
<td>110</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TXC/RXC Rise Time</td>
<td>TSR</td>
<td></td>
<td></td>
<td>20</td>
<td>nS</td>
</tr>
<tr>
<td>TXC/RXC Fall Time</td>
<td>TSF</td>
<td></td>
<td></td>
<td>12</td>
<td>nS</td>
</tr>
<tr>
<td>TXD Delay Time from TXC↓</td>
<td>TDTXD</td>
<td></td>
<td>-100</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TXEN↑ Delay Time from TXC↓</td>
<td>TDTEN</td>
<td></td>
<td>100</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TXEN↓ Delay Time from TXC↓</td>
<td>TDTDL</td>
<td></td>
<td>100</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RXD Setup Time to RXC↑</td>
<td>TSAXD</td>
<td></td>
<td>50</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RXD Hold Time from RXC↑</td>
<td>THRXD</td>
<td></td>
<td>70</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RXC Cycle Time *3</td>
<td>TCYH</td>
<td>When using on-chip DPLL</td>
<td>25</td>
<td>200</td>
<td>nS</td>
</tr>
<tr>
<td>RXC Active Low</td>
<td>THHL</td>
<td>When using on-chip DPLL</td>
<td>5</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RXC Active High</td>
<td>THHH</td>
<td>When using on-chip DPLL</td>
<td>5</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RXC Rise Time</td>
<td>THR</td>
<td>When using on-chip DPLL</td>
<td>5</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>RXC Fall Time</td>
<td>THF</td>
<td>When using on-chip DPLL</td>
<td>5</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TX, RX DATA Cycle Time</td>
<td>TCYD</td>
<td>When using on-chip DPLL</td>
<td>250</td>
<td>2000</td>
<td>nS</td>
</tr>
<tr>
<td>RXD Setup Timing</td>
<td>TRX</td>
<td>When using on-chip DPLL</td>
<td>+0.2TCYD</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TXC Active Low</td>
<td>TTTL</td>
<td>When using on-chip DPLL</td>
<td>0.5TCYD-25</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TXC Active High</td>
<td>TTTL</td>
<td>When using on-chip DPLL</td>
<td>0.5TCYD-25</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TXD Change Delay from TXC↓</td>
<td>TDTX1</td>
<td>When using on-chip DPLL</td>
<td>50</td>
<td></td>
<td>nS</td>
</tr>
<tr>
<td>TXD Change Delay from TXC↑</td>
<td>TDTX2</td>
<td>When using on-chip DPLL</td>
<td>0.5TCYD-50</td>
<td></td>
<td>nS</td>
</tr>
</tbody>
</table>

*3  DPLLed clock cycle time should be from 250 ns to 2000 ns.

This is a target spec. and can be changed during development.
AC Timing Test Points

Timing Waveform

CLK Waveform

TXC/RXC

RXC
(When Using on-chip DPLL)

0.4V
0.8V
2.2V
3.0V
1.5V
2.4V
1.5V
0.4V
BUS SLAVE MODE

CS

\( t_{SAW} \)

\( t_{WNL} \)

\( t_{HWC5} \)

\( \overline{WR} \)

ADR/UBE

DB

\( t_{SDW} \)

\( t_{HWD} \)

\( \overline{CS} \)

ADR

\( t_{SAR} \)

\( t_{RRL} \)

\( t_{HRA} \)

\( \overline{RD} \)

DB

\( t_{DRD} \)

\( t_{FRD} \)

V_{DD}

\( t_{SVDD} \)

RESET

\( t_{RSTL} \)

\( \overline{WR} \) or \( \overline{RD} \)

\( t_{SYWR} \)

\( \overline{RD} \)

\( t_{RVWR} \)

\( t_{RVWR} \)

\( \overline{WR} \)

\( t_{RVWR} \)
When not using on-chip DPLL
When using on-chip DPLL
NEC OFFICES

NEC Electronics (Europe) GmbH, Oberrather Str. 4, 4000 Düsseldorf 30, W. Germany, Tel. (02 11) 65 03 01, Telex 8 58 996-0

NEC Electronics (Germany) GmbH, Oberrather Str. 4, 4000 Düsseldorf 30, Tel. (02 11) 65 03 02, Telex 8 58 996-0
- Königstr. 12, 3000 Hannover 1, Tel. (05 11) 31 60 91, Telex 9 230 109
- Arabellastr. 17, 8000 München 2, Tel. (0 89) 92 10 03-0, Telex 5 22 971
- Heilbronner Str. 314, 7000 Stuttgart 30, Tel. (07 11) 89 09 10, Telex 7 252 220

NEC Electronics (BNL) - Boschdijk 187a, NL-5612 HB Eindhoven, Tel. (0 40) 44 58 45, Telex 51 923

NEC Electronics (Scandinavia) - Box 4039, S-18304 Täby, Tel. (08) 73 28 200, Telex 13 839

NEC Electronics (France) S.A., 9, rue Paul Dautier, B. P. 187, F-78142 Velizy Villacoublay Cedex, Tél. (1) 39 46 96 17, Télex 699 499

NEC Electronics (France) S.A., Representacion en Espana, Edificio «La Caixa», Paseo de la Castellana 51, E-28046 Madrid, Tél. (1) 41 94 150, Télex 41 316

- Rome Office, Via Monte Cervialto, 131, I-00139 Rome, Tel. (06) 8 11 12 91, Telex 623 323

NEC Electronics (UK) Ltd., Cygnus House, Sunrise Park Way, Milton Keynes, MK14 6NP, Tel. (09 08) 69 11 33, Telex 777 565
- Dublin Office, 34/35 South William Street, Dublin 2, Ireland, Tel. (00 01) 71 02 00

NEC cannot assume any responsibility for any circuits shown or represent that they are free from patent infringement.
NEC reserves the right to make changes any time without notice.
© by NEC Electronics (Europe) GmbH