MC68020 Minimum System Configuration

As described in this application note, Motorola's MC68020 32-bit microprocessor minimum system configuration can be used for many applications that were formerly in the realm of mainframe computers or microprocessors. These applications need the benefits of the complete 32-bit architecture but with a simpler address and data bus configuration.

DESCRIPTION

This application uses the MC68020 microprocessor in a system having minimum hardware interconnects. The system, which includes an 8-bit data bus, a 24-bit address bus, and as few devices as possible, can upgrade an existing MC68008 design or can be constrained for use in a space-limited environment.

The system uses inexpensive large-scale integration (LSI) devices. In addition to the MC68020, a single byte-wide electrically programmable read-only memory (EPROM) (similar to a 27512-170) and static random-access memory (SRAM) (the Motorola MC68515) are used with the Motorola MC68901 multifunction peripheral (MFP) for system timing and serial communications. Also, medium-scale integration/small-scale integration (MSI/SSI) TTL devices are used for clock generation, high-speed gating, buffering BERR generation, and address decoding. Other common components are required for power supply decoupling, reset generation, and pullups. The schematic diagram is shown in Figure 1 (found at the back of this document), and the list on page 2 shows the inputs, outputs, and logic equations for device U05, a programmable array logic PAL16L8. Table 1 lists the parts for the minimum system configuration.

INPUT/OUTPUT

In this minimum system configuration, the only system I/O required is a serial interface to a terminal or some similar device. This interface uses the USART contained on the MC68901 MFP, and the MFP also generates baud rates for the onboard serial port. The XTAL1 and XTAL2 inputs are connected to a 2.4576-MHz crystal. The delay-only timers C and D in the MFP are configured for prescaling and delay generation for timing a 9600 baud asynchronous communication port. The RS-232-C interface-level generation is accomplished by using Motorola's MC145406, a single 16-pin device providing three RS-232-C line drivers and three RS-232-C line receivers. It provides a very efficient single-device solution for the vast majority of RS-232-C interfacing requirements. Of the standard RS-232-C handshake lines, only RTS is controlled via software, DTR is strapped in the active-high state, and all others are ignored.

Unused inputs are important considerations for any MC68020 system, regardless of configuration. All inputs must be driven to a known level. Several inputs to the MC68020 were not used in this application — signals such as CDIS, BR, BGACK, AVEC, and HALT, all of which are active in a low state. These inputs were pulled to a high level to avoid conflict with functions on the devices that were used.

SYSTEM TIMING GENERATION

The MC68020RC12 microprocessor operates at a clock speed of 12.5 MHz. The simplest way to obtain a clean, symmetrical clock signal is to use the buffered output of a 12.5-MHz oscillator to drive a pair of F04 inverter/buffers, eliminating the use of expensive delay lines or complex timing functions. In addition, critical parameters for worst-case performance can be determined for a guaranteed functional design over worst-case timing constraints. These parameters include clock skew, setup and hold-time conformance, and worst-case signal propagation.

The basic bus cycle of the MC68020 is asynchronous and occurs in three clock periods. Using memory devices listed previously can provide for zero wait-state operation at a 12.5-MHz clock frequency. However, an MC68020 system using an 8-bit data bus would usually not have zero wait-state performance as a major system requirement. Thus, the extra gating required to allow zero wait-state access to the SRAM and EPROM is inconsistent with the minimum system configuration. A simpler approach is to allow a single wait state for memory accesses, using inexpensive 150-170-ns devices. This approach allows
INPUTS, OUTPUTS, AND LOGIC EQUATIONS FOR U05, PAL16L8

Inputs

Pin 1 = A00; Address Bus Bit 0
Pin 2 = A01; Address Bus Bit 1
Pin 3 = A02; Address Bus Bit 2
Pin 4 = A16; Address Bus Bit 16
Pin 5 = A17; Address Bus Bit 17
Pin 6 = A18; Address Bus Bit 18
Pin 7 = A19; Address Bus Bit 19
Pin 8 = A20; Address Bus Bit 20
Pin 9 = A21; Address Bus Bit 21
Pin 10 = A22; Address Bus Bit 22
Pin 11 = A23; Address Bus Bit 23
Pin 12 = FC0; Function Code Bit 0
Pin 13 = FC1; Function Code Bit 1
Pin 14 = FC2; Function Code Bit 2

Outputs

Pin 12 = IACK; IACK cycle output
Pin 17 = MFP; MFP select
Pin 18 = ROM; ROM select
Pin 19 = RAM; RAM select

Logic Equations

IACK = FC0 & FC1 & FC2 & A02 & !A01 & !A00;
MFP = FC0 & !FC1 & A16 & A17 & A18 & A19 & !A20 & A21 & A22 & A23 #
     & !FC0 & FC1 & A16 & A17 & A18 & A19 & !A20 & A21 & A22 & A23;

Table 1. MC68020 Minimum System Configuration Parts List

<table>
<thead>
<tr>
<th>Reference</th>
<th>Part Number</th>
<th>Description</th>
<th>Manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>U01</td>
<td>MC68020RC12</td>
<td>MPU</td>
<td>Motorola</td>
</tr>
<tr>
<td>U02</td>
<td>MCM6064P15</td>
<td>8K x 8 SRAM</td>
<td>Motorola</td>
</tr>
<tr>
<td>U03</td>
<td>27512-170</td>
<td>64K x 8 EPROM</td>
<td>Various</td>
</tr>
<tr>
<td>U04</td>
<td>MC68901P</td>
<td>MFP</td>
<td>Motorola</td>
</tr>
<tr>
<td>U05</td>
<td>PAL16L8</td>
<td>PAL</td>
<td>Various</td>
</tr>
<tr>
<td>U06</td>
<td>MC74F32</td>
<td>Quad 2-in NOR</td>
<td>Motorola</td>
</tr>
<tr>
<td>U07</td>
<td>MC74F00</td>
<td>Quad 2-in NAND</td>
<td>Motorola</td>
</tr>
<tr>
<td>U08</td>
<td>MC74F161</td>
<td>4-Bit Sync Counter</td>
<td>Motorola</td>
</tr>
<tr>
<td>U09</td>
<td>MC145406</td>
<td>Hex RS232 Tx/Rx</td>
<td>Motorola</td>
</tr>
<tr>
<td>U10</td>
<td>MC74F74</td>
<td>Dual D-Flip-Flop</td>
<td>Motorola</td>
</tr>
<tr>
<td>U11</td>
<td>MC74F74</td>
<td>Dual D-Flip-Flop</td>
<td>Motorola</td>
</tr>
<tr>
<td>U12</td>
<td>MC74LS14</td>
<td>Hex Schmitt Inverter</td>
<td>Motorola</td>
</tr>
<tr>
<td>U13</td>
<td>MC74F04</td>
<td>Hex Inverter</td>
<td>Motorola</td>
</tr>
<tr>
<td>Y01</td>
<td>Oscillator</td>
<td>12.5 MHz</td>
<td>Various</td>
</tr>
</tbody>
</table>

NOTE: Capacitors, resistors, a crystal, a diode, and a switch are also required.
DSACK generation using a single MC74F74 and two of the four gates in an MC74F00.

The timing for generation of DSACK0 in this manner is simple. In all bus cycles, the address bus is guaranteed stable within 40 ns of the rising edge of the first clock of the bus cycle, and \( \overline{AS} \) is guaranteed asserted within 40 ns of the falling edge of the same clock. In all read cycles, data is required to meet a 10-ns setup time with respect to the falling edge of the last clock in the cycle, regardless of any wait states. In write cycles, data is guaranteed stable well in advance of the same edge.

Figure 2 shows a RAM/ROM read cycle followed by a RAM write cycle in the minimum system configuration. DSACK generation begins on the falling edge of the second clock of the cycle, and DSACK0 is asserted after the rising edge of the third clock cycle. The cycle completes after the falling edge of the fourth clock cycle.

Address strobe gates the output of the PAL16L8 at U05 to select access to the ROM, RAM, or MFP during an IACK cycle. If the MFP is selected, DSACK is generated by the MFP's DTACK output. If ROM or RAM access is selected, the dual F74 DSACK generation circuit is used per the diagram shown in Figure 3.

Using the previous timing constraints, a simple formula determines the number of wait states, the speed of memory devices required, or the time allowable for decode logic in any 12.5-MHz system:

\[
110 \text{ ns} + 80 \text{ ns} (# \text{ wait states}) = \text{system access time}
\]

or

\[
110 \text{ ns} + 80 \text{ ns} (# \text{ wait states}) = \text{device access + decode time}
\]
In the MC68020, minimum system configuration, one-wait-state operation yields a total system access time of 190 ns. Using 150–170-ns devices, at least 20 ns is available for decoding, well within the performance capability of an MC74F32. The MC74F32 is used to gate address decode with AS to develop chip selects for the memory devices or for higher speed B and D series PALs.

The MC68901 MFP operates at any clock frequency from 1–4 MHz. For simplicity, the 12.5-MHz clock used to operate the MC68020RC12 was divided by four, yielding a 3.125-MHz frequency for operating the MFP. This frequency is also suitable for operating an MC74F161 4-bit counter used to generate BERR as the result of an incomplete bus cycle. The MFP generates DTACK after it has been accessed and adheres to a basic four MFP-clock bus cycle. As such, all read/write accesses to the device complete within 1.28 µs, well within the 5–12-µs nominal timeout of the BERR watchdog.

BASE ADDRESS DECODING

Decoding of base addresses for the directly accessible devices is accomplished using a single 16L8 PAL. The PAL speed required for the minimum system configuration is not critical; only the address lines and functional codes are decoded. Depending on the cycle in process, an active-low output is logically ANDed with AS (also active low) in an MC74F32 to enable the ROM, RAM, MFP, or MFP IACK. It would be possible to eliminate the MC74F32 if a high-speed PAL similar to D-series devices is used.

SOFTWARE

The following software listing (see Figure 4) describes minimum system initialization and routines used to verify the prototype hardware developed in this application note. The routines include a simple memory exerciser program, which first performs a cursory test of RAM memory and then initializes RAM, including the interrupt vector table, with appropriate information. Also included is a minimum initialization of the MC68901 MFP. Actual application software can be added as needed. The software listing in Figure 4 can be used as minimum routines for any system of similar configuration.

SYSTEM EXPANSION

This minimum system configuration can be expanded to a 32-bit data bus configuration by adding three more SRAM and/or EPROM devices. Also add chip select connections to the memory devices, connection to the appropriate address and data bus lines, and expanded hardware in support of 8- and 16-bit accesses over the 32-bit data bus.

For an expanded system, additional I/O requirements can be handled with the unused portions of the MFP. With their highly functional programmability, the six unused ports in the general-purpose I/O can be used for external inputs to allow edge detection, pulse generation, or similar I/O functions. Added circuitry can be limited to external inputs to the device. For other functions, additional address decode logic and the particular I/O are needed.

The required hardware is described in the MC68020 User's Manual.

CONCLUSION

The minimum system configuration can be expanded to larger data paths and can be adapted to many applications requiring the performance of Motorola's MC68020 32-bit microprocessor.
**EOU MFPBAS+$11 INTERRUPT IN-SERVICE B**
**EOU MFPBAS+$13 INTERRUPT MASK A**
**EOU MFPBAS+$15 INTERRUPT MASK B**
**EOU MFPBAS+$17 VECTOR**
**EOU MFPBAS+$19 TIMER A CONTROL**
**EOU MFPBAS+$1B TIMER B CONTROL**
**EOU MFPBAS+$1D TIMER C/D CONTROL**
**EOU MFPBAS+$1F TIMER A DATA**
**EOU MFPBAS+$21 TIMER B DATA**
**EOU MFPBAS+$23 TIMER C DATA**
**EOU MFPBAS+$25 TIMER D DATA**
**EOU MFPBAS+$27 SYNCHRONOUS CHARACTER**
**EOU MFPBAS+$29 USART CONTROL**
**EOU MFPBAS+$2B RECEIVER STATUS**
**EOU MFPBAS+$2D TRANSMITTER STATUS**
**EOU MFPBAS+$2F USART DATA**

---

Program section

The ROM in this application is mapped to the variable ROMBAS. All executable code is resident in ROM.

**START**

**EOU ROMBAS**
**DCL STACK**
**DCL ROMSTART**

**ROMBUF**

**DS.L 32**

**MEMDAT**

**EQU * MEMORI EXERCISER DATA**

**DC.B 55**
**DC.B 5A**
**DC.B 5D**
**DC.B 5F**
**DS.L $100**

**ROMSTART**

**EOU * BEGINNING OF PROGRAM SECTION**
**MOVE.L $RAMBAS,D0**
**MOVECL.L D0,VBR**

*** Memory exerciser ***

This routine performs a cursory check of memory prior to proceeding. An error count is contained in D7 upon completion.

**CLRL D7**
**MOVE.L #3, D3**
**LEA.L RAMBAS,A0**
**LEA.L MEMDAT,A1**

**LOOP0**

**EQU * INIT INNER LOOP COUNTER**
**MOVE.B $1FFF,D0**
**MOVE.B (A1, D3), D2**

**LOOP1**

**EQU * PUT DATA INTO MEMORY**
**MOVE.B D2,(A0,D0)**
**CMP.B (A0,D0),D2**
**BEQ.L LOOPLP_1**
**ADDQ #1,D7**

**LOOP1**

**EQU * TEST ALL OF RAM MEMORY**
**DBRA D0,LOOP1**
**DBRA D3,LOOP0**

---

Figure 4. MC68020 Minimum System Configuration Startup Software (Sheet 2 of 5)
When done with test, memory is to be initialized with NOPs and vector table initialized with address of generic handler. After initialization, D7 will contain the number of errors from the test section.

MEMINIT

```asm
EQU *
LEA L RAMBAS,A0
MOVE L #$SFFE,D0
MOVE L #$S3FE,D2
MOVE W #$NOP,D1
```

POINT AT BASE OF RAM AGAIN
USE AS LOOP COUNTER FOR MEMINIT
POINT AT BOTTOM OF VECTOR TABLE
FILL NON-VECTOR MEMORY WITH NOPs

LOOP2

```asm
EQU *
MOVE.W D1,(A0,00)
SUBQ #2, D0
CMP.L D0,D2
BNE.S LOOP2
```

PUT DATA INTO MEMORY
DECUMENT COUNTER
NOW LOOK FOR BOTTOM OF VECTOR TABLE
CONTINUE UNTIL THERE
ELSE MOVE TO LONG-WORD INIT
AND PUT GENERIC EXCEPTION HANDLER IN REST OF VECTOR TABLE MEMORY

LOOP3

```asm
EQU *
MOVE.L D1,(A0,00)
SUBQ #4, D0
BGE.S LOOP3
```

PUT HANDLER ADDRESS THERE
AND DECREMENT POINTER
FILL REST OF MEMORY

Done with memory check initialization

Now init the MC6801 MFP

```asm
JSR MFPINIT
```

DO SO AS SUBROUTINE FOR ADDED USE AT LATER TIME

TST D7
BEQ NO_ERR
```

NOW CHECK IF ANY ERRORS
IF NONE OUTPUT OK MESSAGE
ELSE OUTPUT ERROR MESSAGE

LEA L ERRMSG,A0
LEA L ERMEND,A1
BRA S INITND
```

POINT AT TOP OF MESSAGE
AND POINT AT END
JUMP TO END OF INIT ROUTINE

NO_ERR

```asm
EQU *
LEA L OKMSG,A0
LEA L ERRMSG-1,A1
```

INIT OK!!!
POINT AT MESSAGE
POINT AT END OF MESSAGE

INITND

```asm
EQU *
JSR SEROUT
```

OUTPUT MESSAGE OVER SERIAL PORT
THEN FALL THRU TO

POLL

```asm
EQU *
BST.B #3,MFPRSR
BNE.S BREAK
BST.B #7,MFPRSR
BEQ.S POLL
```

POLL SERIAL PORT FOR INPUT
CHECK FOR BREAK
IF PRESENT, JUMP TO PROCESS
ELSE CHECK FOR CHARACTER
LOOP IF NO DATA PRESENT, ELSE
DATA PRESENT IN UART RECEIVER
USER INSERT INPUT CHARACTER
PROCESSING ROUTINE HERE

BREAK

```asm
EQU *
```

BREAK DETECT ROUTINE

```asm
JMP POLL
```

USER INSERT BREAK HANDLER HERE
AND RETURN WHEN COMPLETED

Figure 4. MC68020 Minimum System Configuration Startup Software (Sheet 3 of 5)
— See Page 9 for Sheet 4 —
Figure 1. MC68020 Minimu
**SEROUT**

EQU * OUTPUT MESSAGE VIA MC68901 USART
BEGINNING OF MESSAGE POINTED AT BY A0, END BY A1

BTST.B #7,MFPTSR
BEQ.S SEROUT

MOVE.B (A0)+,DO
MOVE.B DO,MFPDDR
CMPA A1,A0
BEQ.S SEROUT

RTS

**GENERIC EXCEPTION HANDLER**

THE GENERIC EXCEPTION HANDLER ACCOMMODATES EXCEPTIONS
THAT OCCUR IN THE MINIMUM SYSTEM VIA RTE. ADD
APPLICATION SAV HERE TO ACCOMMODATE EXCEPTIONS THAT
ARE PROCESSED IN SPECIFIC APPLICATIONS

**EXCHND**

EQU *

RTS

**MFPINIT**

EQU *

MFPINIT MC6801 INITIALIZATION ROUTINE

CLR.L DO
SUBQ #1,DO
MOVE.B DO,MFPDDR
ADDQ #3,DO
MOVE.B DO,MFPDDR

MOVE.B #$11,MFPTCDR
MOVE.B #$88,MFPUCR

**INITIALIZE MFP VECTORS AND HANDLERS**

MOVE.L #MFPVCT,DO
MOVE.B DO,MFPVR
ASL.L #2,DO
MOVE.L DO,A0
MOVE.L #MFP,DO

LOAD INTO MFP
NOW SHIFT LEFT 2
PUT INTO ADDRESS REGISTER
AND INIT APPROPRIATE VECTOR

NOW START Tx, Rx CLOCKS

MOVE B #1,MFPSSR
MOVE B #5,MFPSSR
BSET.B #7,MFPSSIP

START RECEIVER CLOCK
START TRANSMITTER CLOCK
NOW RAISE RTS

RTS

DONE!! RETURN FROM ROUTINE

**MFP EXCEPTION HANDLER ROUTINE**

EQU *

USER INSERT EXCEPTION HANDLER HERE

RTS

**MESSAGES SECTION**

Figure 4. MC68020 Minimum System Configuration Startup Software (Sheet 4 of 5)
Figure 4. MC68020 Minimum System Configuration Startup Software (Sheet 5 of 5)