To: Bob Belleville
Bill Bull
Gary Butts
Mike Clark
Jerome Coonen
Dan Hillman
Larry Kenyon
Burrell Smith

Date: October 29, 1984

From: Peter Ashkin Pete

Subj: Front Desk Bus - rev 3.1

Enclosed is the latest version of the Front Desk Bus (rev 3.1) specification. It is divided into three sections: Preface - which contains the 12 fundamental properties of the bus; Main - which contains the commands which devices on the bus must execute and details on the timing and the modulation of the bus; and the Appendix - which describes the interface between the FDB "modem" and the Macintosh digital subsystem.

There are some things this specification does not contain, there is no description of the connectors nor is there any mention of how the bus should be "used".

Please read this over and feel free to make any changes or improvements. I'm interested in a robust (and useful) specification. I'll contact each of you the first week in November to discuss your comments.

Thanks!!!

Preface - Front Desk Bus

To make the "Front Desk Bus" a more flexible and powerful interface, it should have the following properties:

1. The bus shall be bidirectional. [An input only bus is too restrictive.]

2. Each device on the bus has a unique address. For practical purposed the address range should be 0 - 14. Some of these addresses may be reserved for broadcasting universal messages. [This seems like a sane number of devices, particularly since there exists today only three devices; keyboard, keypad and mouse.]

3. All command transactions shall be eight bits long. All data transactions shall be 16 bits long. [This facilitates the decoding of commands by devices of limited intelligence.]

4. The host shall be the undisputed bus master. [This removes any question of who's controlling the bus.]

5. There shall be a limited number of commands. Commands should be broken into two groups, basic commands (**TALK** and **LISTEN**) which all devices on the bus shall understand; and advanced commands which only intelligent devices (as appropriate) should understand. [This makes the command interpreter, be it hardware or software, simple. It also allows more complex devices to used some of the "fancier" features of the bus.]

6. There shall be only one active talker on the bus at any time, this may be the host or an addressed device. [A device addressed to **TALK** with data to send "untalks" itself after it sends its 16 bits of data or if it has no data to send "untalks" itself immediately and allows the bus to time-out.]

7. The bus protocol must accept devices that talk at different speeds. The host, at a minimum, must be able to listen at various speeds. [This implies that the data on the bus must be "self-clocked". By not rigidly fixing the speed of transmission, the bus does not need to be crystal (etc.) controlled.]

8. There shall be only one active listener on the bus at any time, this may be the host or an addressed device. [A device addressed to **LISTEN** "unlistens" itself after it receives 16 bits of data or if it receives a new command before receiving 16 bits of data.]

9. An interrupt mechanism must be available which circumvents the needs to poll devices that need service. [Since the bus is relatively slow, the interrupt latency time in a polled environment is long. The ability to interrupt the master for service is important.]

10. There shall exist a mechanism that sends a unique signal that puts all devices on the bus into the command (reset) mode. [This is important if for some reason the bus gets "hung".]

11. There should be a minimum number of "time-outs" needed on the bus. The only needed time out should be to time out a non-responsive talker. [Timers are ugly, but waiting for a dead device is uglier. The length of this time-out should be controlled by the host.]

12. Hand-off of the bus from the host to a talker and back again must be made without bus contention. [Contentions hurt output drivers and are noisey. The pullup of the bus if it is actively driven must go tristate when inactive on the bus.]

Commands:

There are two major command groups; basic commands and advanced commands. All devices on the bus shall understand at least one command in the basic group and optionally understand commands in the advanced group.

BASIC Command Group:

There are two commands in this group; **TALK** and **LISTEN**.

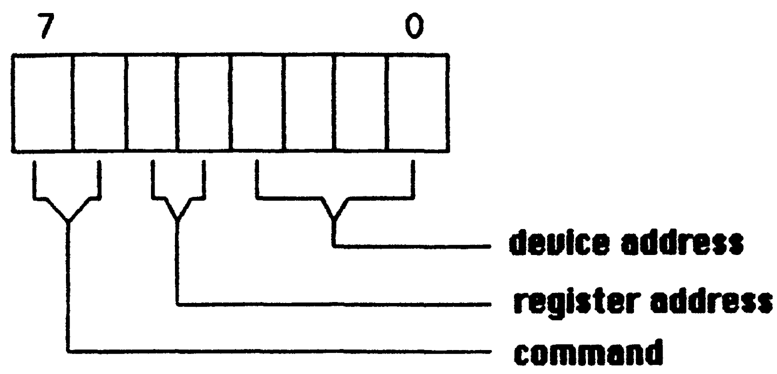
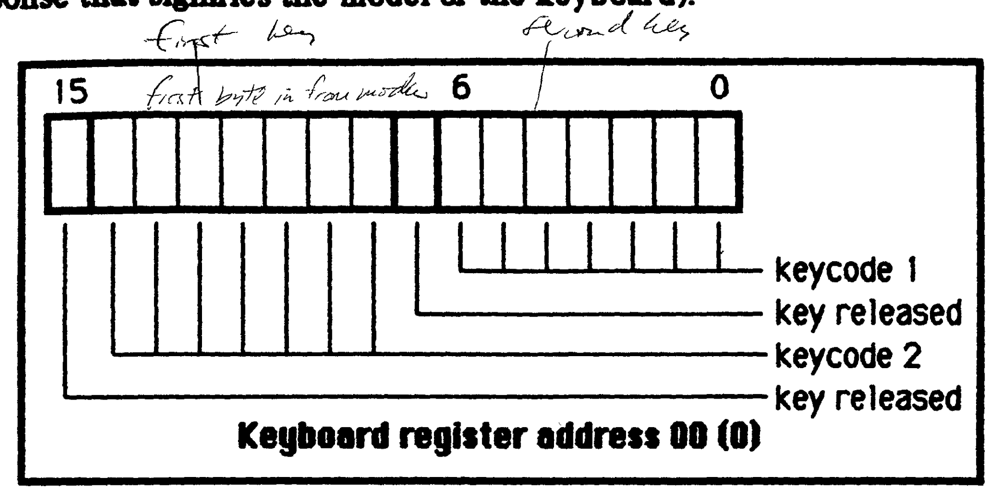


Fig. 1 BASIC Command Format

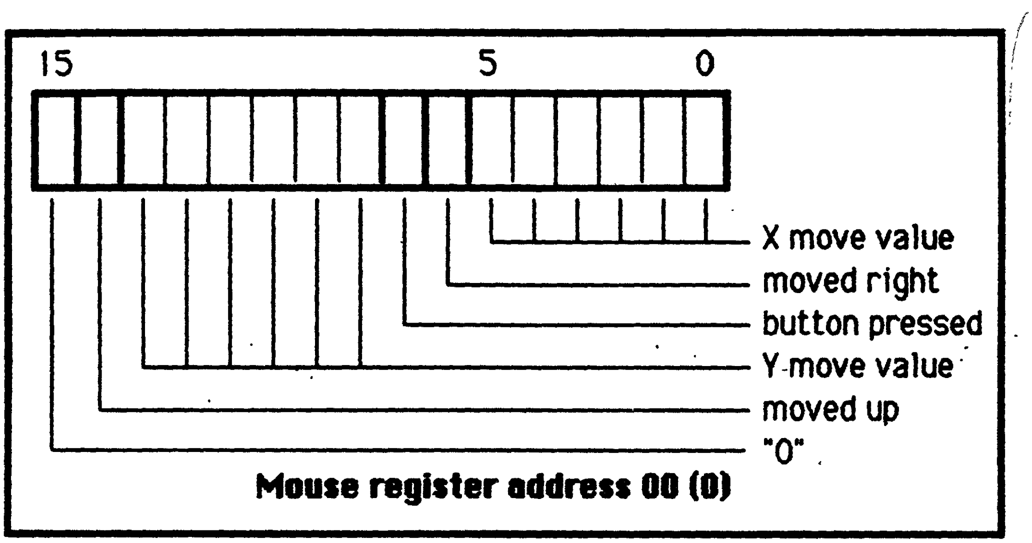
Commands may only be sent by the host. Data may be sent by either the host (**LISTEN**) or the addressed device (**TALK**).

The next two bits form the command: "11" for **TALK** and "10" for **LISTEN**. All devices on the bus must obey at least one of these commands. Keyboards, numeric keypads and mice as a minimum must respond to the "00" **TALK** command. When a device is addressed to **TALK**, it must respond before being timed out by the host. This timeout shall be nominally $2 * T_{cyc}$ after the rising edge of the stop bit of the **TALK** command. The selected device, if it does not timeout, becomes active on the bus, performs its 16 bit data transaction then "untalks" itself and goes inactive on the bus. Thus **TALK** commands transfer only 16 data bits at a time and a new **TALK** command must be issued to transfer additional data.

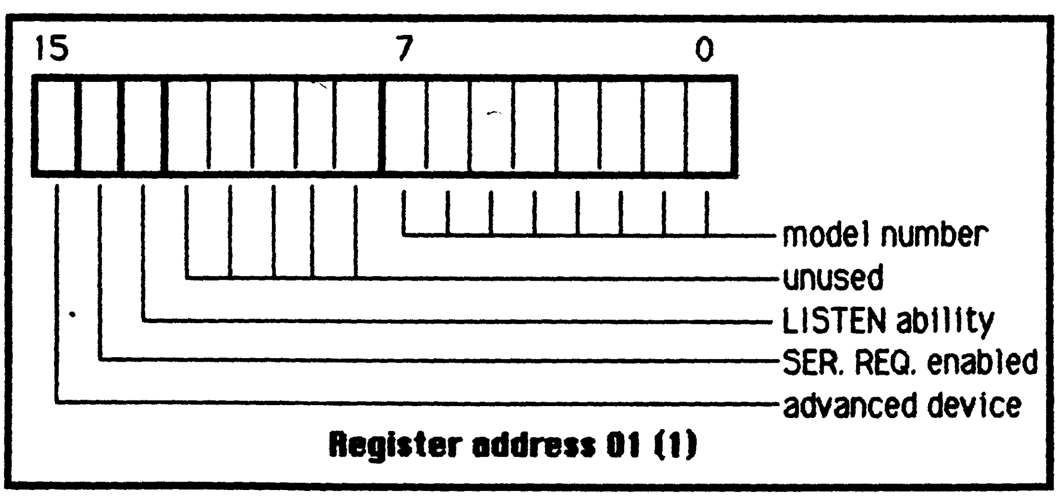
When a device is addressed to **LISTEN**, it is enabled to accept a data transaction from the host. Only a single device at a time can be addressed to **LISTEN**. After the device is addressed, it is enabled to receive the next 16 data bits that are placed on the bus by the host. After the data bits are received, the transaction is complete and the device "unlistens" itself. If a device is addressed to **LISTEN** and it receives another command on the bus before it receives any data, then by definition the transaction is immediately complete and the device "unlistens" itself. The next field is a two bit register address field. This field, which is optional, allows a specific register within an addressed device to be specified. An example of where this might be used is to differentiate a data register (in a keyboard, the specific keystroke) from a status/configuration register (in a keyboard, a response that signifies the model of the keyboard).



Keyboard register address 00 (0)



Mouse register address 00 (0)



Register address 01 (1)

Finally there is a four bit device address field which specifies the address of the selected device. These addresses range from 0 - 14.

Device Table			
Address	**Device type**	**Conforming**	**Advanced**
0011 (3)	Keyboard	yes	yes
1100 (12)	Mouse	yes	yes

ADVANCED Command Group:

There are two commands in this group; **ENABLE (INTERRUPT)** and **DISABLE (INTERRUPT)**. There are also five reserved commands for future expansion.

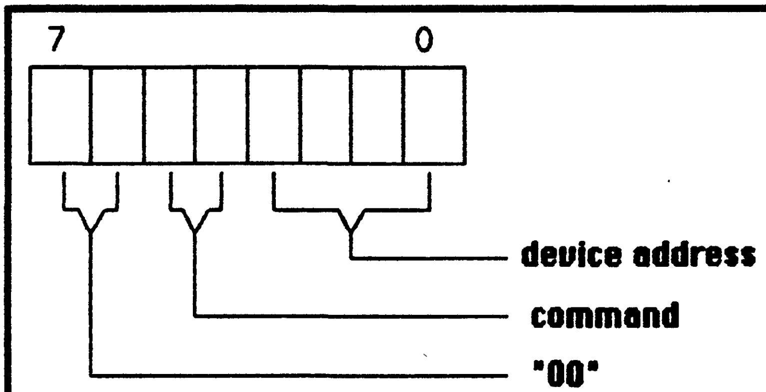


Fig. 2 ADVANCED Command Format

Note that the defined advanced commands have the two most significant bits set to "00".

These commands deal with the ability of devices on the bus to interrupt the host. This is useful in systems where the interrupt response time in a polled system is longer than desired. **ENABLE** allows selected devices to signal an interrupt on the bus, or conversely **DISABLE** selectively inhibits the signalling of an interrupt. When an enabled device signals an interrupt, the host may not know which device has signalled if multiple devices have been enabled.

"00"	**ENABLE**
"01"	**DISABLE**

ENABLE and **DISABLE** require that the address of the desired device be specified. The range is 0 - 14. Address 15 is a reserved address for the

DISABLE command and serves as a global disable. Address 15 is a reserved address for the **ENABLE** command and serves as a global enable.

To allow for future expansion of the command structure, a group of "place holder" **RESERVED** instructions has been defined. These instructions shall be treated as no-ops.

Command Syntax			
76	**54**	**3210**	
00	00	A3 - A0	**ENABLE**
00	01	A3 - A0	**DISABLE**
00	10	XXXX	**SENDRESET** *
00	11	XXXX	**RESERVED**
01	XX	XXXX	**RESERVED**
10	R1 R0	A3 - A0	**LISTEN**
11	R1 R0	A3 - A0	**TALK**
* forces RESET signal on FDB			

MODULATION:

There are two forms of modulation on the bus, **NORMAL** transactions which transmit commands and data, and **SIGNALS** which broadcast global messages such as **RESET**, **SYNCH** and **INTERRUPT**.

NORMAL transactions:

To achieve the goal of the bus being self clocked, a RZ code for modulation has been adopted. This code has several properties that are advantageous to the Front Desk Bus. Among these advantages are: ease of recovery of the clock and the data; always leaves the bus in a known state (without the use of dummy transactions); and has definite "openings" in the waveforms to signal special transactions. Each bit cell boundary is signified by a falling edge on the bus. The period of each bit cell is the time between two falling edges on the bus. The data is encoded as the ratio of low to high time of each bit cell. Thus a "0" is encoded as a bit cell inwhich the low time is <u>greater</u> than the high time. Conversely, a "1" is encoded as a bit cell inwhich the low time is <u>less</u> than the high time.

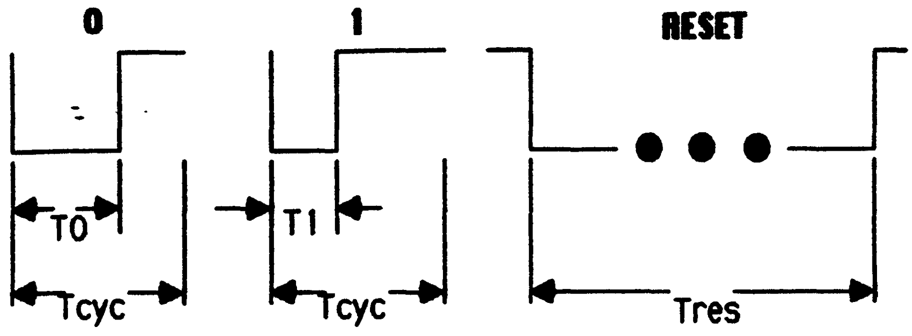


Fig. 3 FDB "0", "1" and RESET Times

To signal the start of a command, a T_{attn} long **attention** pulse is sent. This is followed by a **synch** pulse to give the initial bus timing. The falling edge of the **synch** pulse is used as a timing reference for the first bit of the command or data. To synchronize the stopping of transactions, one "0" **stop** bit is sent. Following the imaginary bit cell boundary after the **stop** bit, the transaction is complete and the host (or talker) releases its active drive of the bus.

As a specific example, a **TALK** command to register 0 of device 6 would be encodes as "11000110". The bus would be modulated with the following:

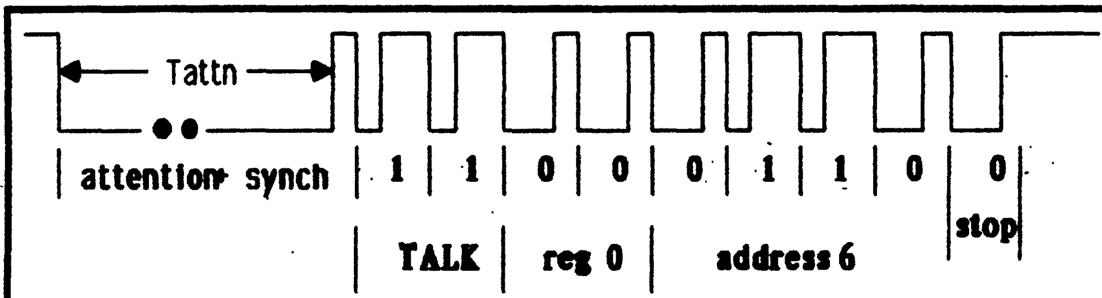


Fig. 4 TALK Command Example

Data transactions on the FDB are all 16 bits in length, with the most significant data bit sent first (bit 15). Data bits are preceeded by a "1" **start** bit and followed by a "0" **stop** bit. There is <u>no</u> timeout on **LISTEN** commands.

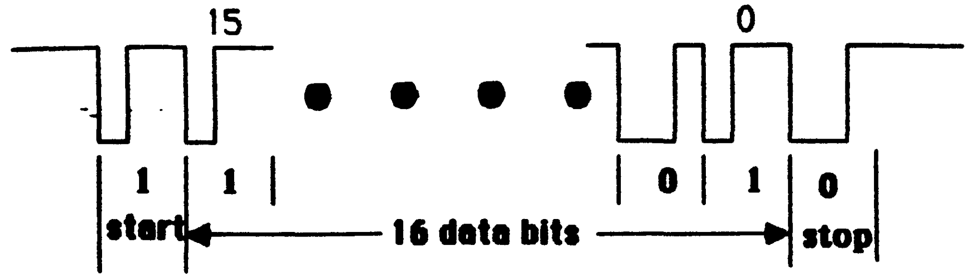


Fig. 5 Data Transaction Example

SIGNALS:

Certain transactions fall under the category of neither commands nor data transactions. These are special transactions which globally broadcast status to devices on the bus. There are three special transactions in this group. **INTERRUPT** is a transaction that devices that have been enabled to interrupt can use to signal the host that they require service. Following any command transaction, an interrupting device can signal by holding the bus low during the low portion of the stop bit of the **NORMAL** transaction. The interrupting device holds the bus low T_{int} beyond the bit cell boundary to signal. Once a device has interrupted, it shall **INTERRUPT** repeatedly until serviced. When the interrupting device is addressed to **TALK**, it shall be considered serviced and not **INTERRUPT** again until it needs to be serviced again.

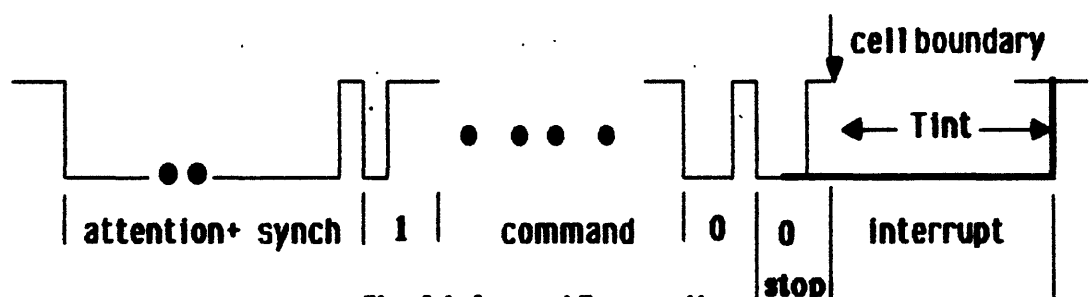


Fig. 6 Interrupt Transaction

The **RESET** signal has the effect that it resets all pending interrupts; it turns the interrupt mode of all devices to **ENABLE**; and in general puts the devices in a mode inwhich they will accept commands. **RESET** issues a break on the bus by holding the bus low for a minimum of T_{res}.

FDB Interface Characteristics						
Symbol	**Parameter**	**Min.**	**Max.**	**Unit**	**Fig.**	**Definition**
T0	"0" low time	30	40	% Tcyc	3	
T1	"1" low time	60	70	% Tcyc	3	
Tattn	ATTENTION signal	560	1040	usec	4	8 * Tcyc
Tcyc	FDB bit cell time	70	130	usec	3	
Tint	INTERRUPT signal	140	260	usec	6	2 * Tcyc
Tres	RESET signal	1.4	2.6	msec	3	20 * Tcyc

Appendix - Macintosh to FDB "modem" interface

The Front Desk Bus "modem" was designed to serve as an interface between the Macintosh and the FDB data line. This modem would handle the bit level modulation/demodulation as well as provide signalling over the line. The hope is to free the Macintosh from the low level protocols and to provide a flexible interrupt driven data transfer mechanism.

The actual Macintosh - modem interface is provided by the 6522A (VIA). The modem is connected to five lines from the VIA's port B. These lines serve to communicate both data and control from the VIA to the modem.

STO: STATE INPUT 0
ST1: STATE INPUT 1. These two input lines to the modem control the transfer of data and commands from the VIA to the modem. There are four possible states:

STATE	ST1	ST0	
S0	0	0	COMMAND
S1	0	1	DATA BYTE 1
S2	1	0	DATA BYTE 2
S3	1	1	NULL (SCLK, DIO and INT = inputs)

SCLK: SHIFT CLOCK. This line is bidirectional in or out. When the modem is in the null state (S3), the SCLK line is configured as an input. The line may be driven by the VIA, but it is ignored by the modem. When in states (S0, S1 and S2), the SCLK line is actively driven by the modem to clock commands (S0) or data (S1 or S2) to/from the modem.

DIO: DATA IN/OUT. This line is birdirectional in or out. When the modem is in the null state (S3), the DIO line is configured as an input. The line may be driven by the VIA, but is ignored by the modem. When in the command state (S0), the DIO line is an input line to the modem and is used to input serial commands. When in data byte states (S1, S2), if the command was a TALK command, then the DIO line is configured as an output line to output serial data received from the FDB. When in data byte states (S1, S2), if the command was a LISTEN command, the the DIO line is configured as an input line to input serial data transmitted to the FDB.

INT: INTERRUPT. This line is an output. It serves two purposes: if it is active (logical 0) at the end of data byte 1 state (S1), then a **TALK TIMEOUT** has occured on the FDB following the last command transaction; if it is active (logical 0) at the end of the data byte 2 state (S2), then a **SERVICE REQUEST** signal was received over the FDB during the last command transaction.

FBDI: **FRONT DESK BUS IN**

FDBO: **FRONT DESK BUS OUT**. Over these lines all devices on the FDB communicate. FDBI is an input and it is the receiver for data sent from devices on the FDB to the modem. FDBO is an open drain output and is the transmitter for data and commands sent from the modem to devices on the FDB. These two lines must be externally connected together. (Note: the FDB is open drain, a 2.2 kohm pullup resistor is required on the modem end of the bus).

OSC: **OSCILLATOR**. This is the clock line for the microprocessor within the FDB modem. 2.46 mHz < fOSC < 4.0 mhz. The bit rate on the FDB is fOSC/280.

PON: **POWER ON RESET**. This is the reset line for the microprocessor within the FDB modem. This line must be held low (logical 0) for a minimum of 10 msec past the power supply rising to >4.5 V.

VCC: **POWER SUPPLY**. This is the power supply pin for the FDB modem. 4.5 V < VCC < 7.0 V. The maximum power supply current is 40 mA.

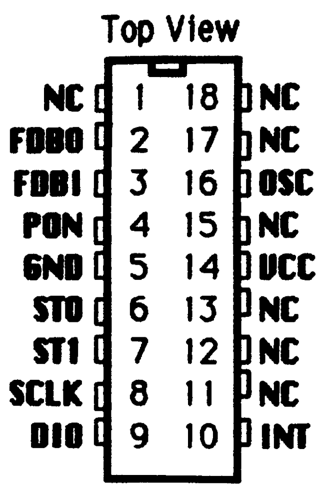
GND: **GROUND**. This is system ground.

Fig. 1 FDB Modem- VIA State Diagram.

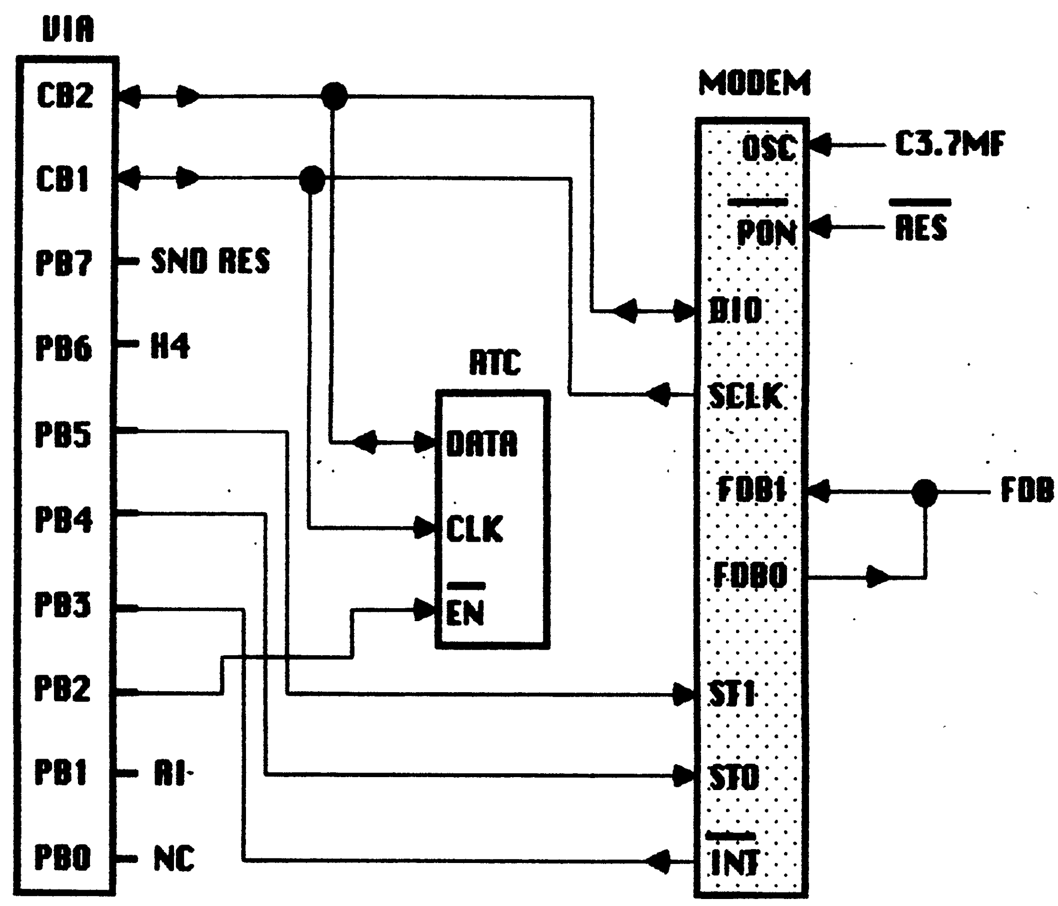
These are the allowed state transitions. All commands may follow the S0 - S1 - S2 sequence. Those commands that have no data or timeout may take the "shortcuts" shown. S3 may be inserted between any two states without affecting the sequence. For example, the sequence S0 - S3 - S1 - S2 is equivalent to S0 - S1 - S2. This is useful when the RTC needs to be serviced while a modem command sequence is in progress. Whenever S0 is entered a new command sequence is initiated. This is useful if a partially executed command needs to be aborted.

Fig. 2 LISTEN and TALK (w/o timeout).

Ts1 is the setup time for a SERVICE REQUEST. It is measured from the falling edge of the eighth SCLK in S2. Th1 is the hold time for a SERVICE REQUEST. It is measured from the change in the ST0 and ST1 lines.



FDB MODEM



Modem - VIA Wiring Diagram

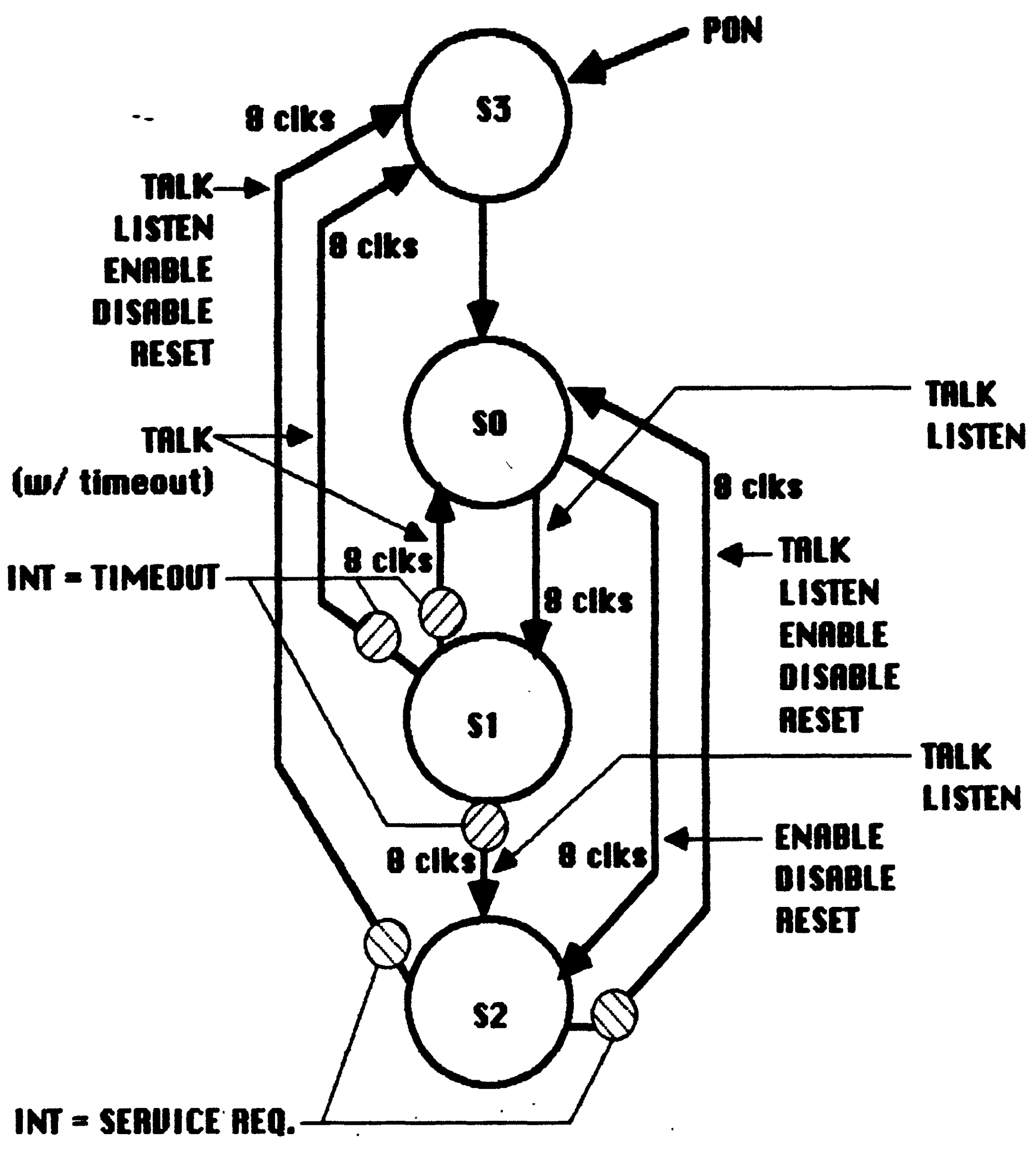


Fig. 1 FDB Modem - VIA State Diagram

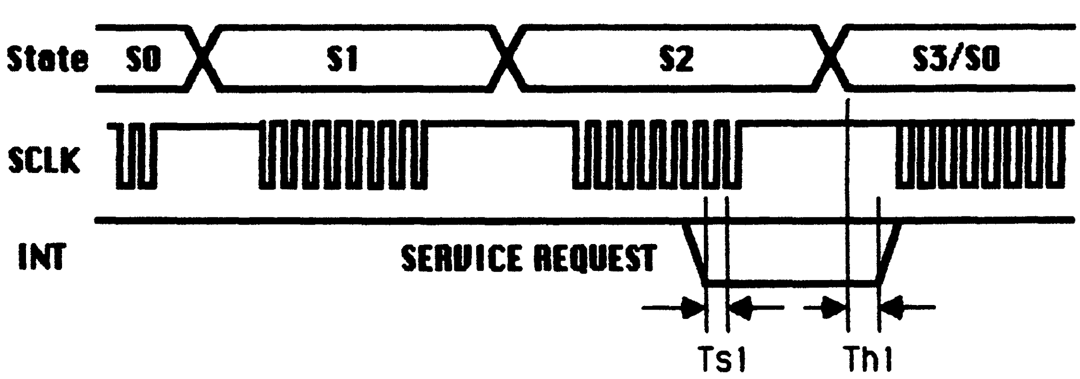


Fig. 2 LISTEN and TALK (w/o timeout)

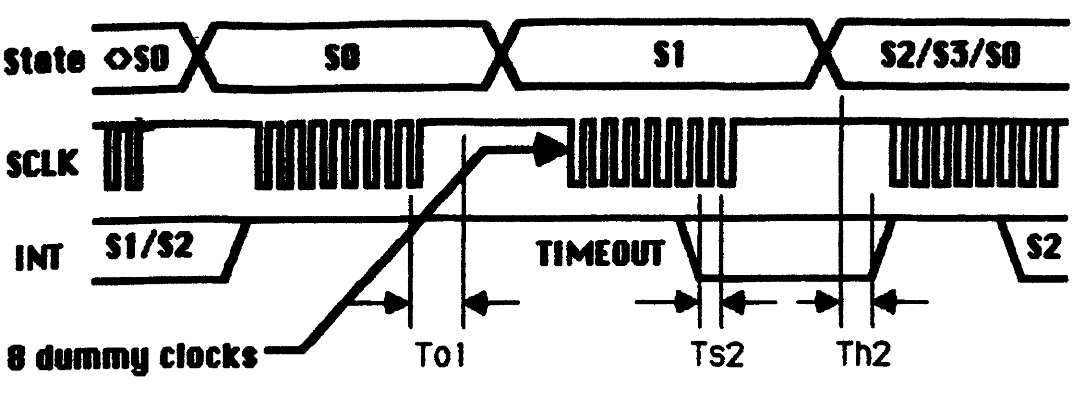


Fig. 3 TALK (w/ timeout)

Fig. 3 TALK (w/timeout).

To1 is the total timeout time on the FDB for a non responsive talker. It is measured from the falling edge of the eighth SCLK in S0. (Actual timeout occurs 260 usec maximum after the rising edge of the stop bit of the TALK command.) Ts2 is the setup time for a TIMEOUT REQUEST. It is measured from the falling edge of the eighth SCLK in S1. Th2 is the hold time for a TIMEOUT REQUEST. It is measured from the change in the ST0 and ST1 lines. Eight dummy SCLKs are generated during S1.

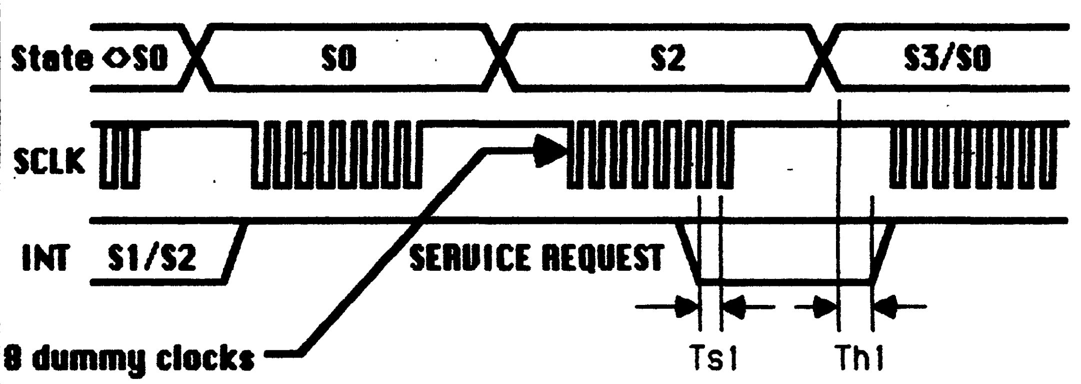


Fig. 4 ENABLE, DISABLE and RESET

Fig. 4 ENABLE, DISABLE and RESET.

ENABLE, DISABLE and RESET commands have no data associated with them. Eight dummy clocks are generated during S2. (If the sequence S0 - S1 - S2 was used, then eight dummy clocks would be generated during S1 as well as during S2.)

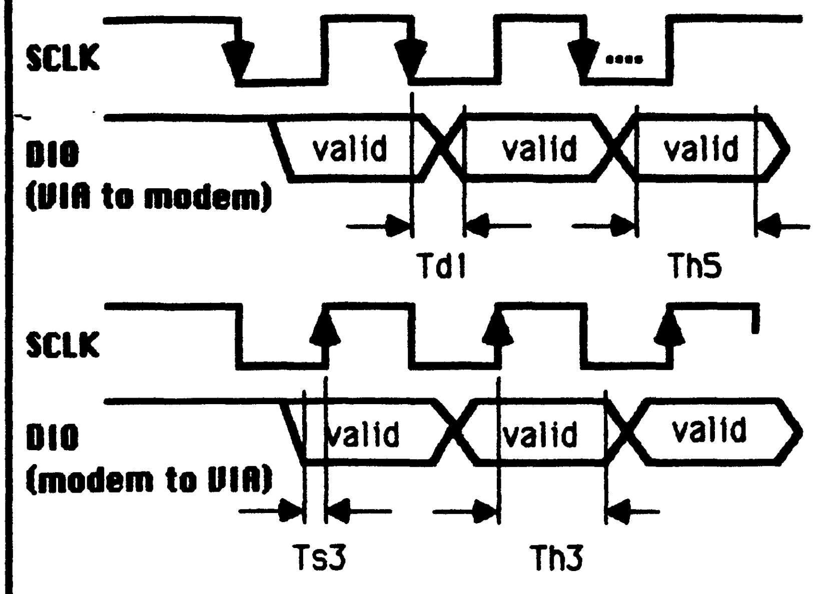


Fig. 5 VIA to modem to VIA

Fig. 5 VIA to modem to VIA

Td1 is the delay time to valid data from the 6522A VIA to the modem. It is measured from the falling edge of SCLK. Th5 is the hold time for valid data from the VIA to the modem following the eighth SCLK. It is measured from the falling edge of the eighth SCLK. Ts3 is the setup time for valid data from the modem to the VIA. It is measured from the rising edge of SCLK. Th3 is the hold time for valid data from the modem to the VIA. It is measured from the rising edge of SCLK.

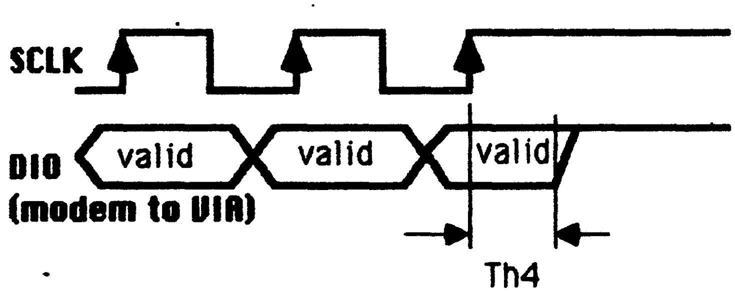


Fig. 6 modem to VIA DIO release time

Fig. 6 modem to VIA DIO release time.

Th4 is the hold time for valid data from the modem to the VIA following the eighth SCLK. It is measured from the rising edge of the eighth SCLK. Following Th4 time, the modem leaves the DIO line in the "1" (high impedance) state.

Modem Interface Characteristics					
Symbol	**Parameter**	**Min.**	**Max.**	**Unit**	**Figure**
Td1	Delay to valid	-	3	usec	5
Th1	Ser. Req. hold	-	60	usec	2,4
Th2	Timeout hold	-	60	usec	3
Th3	Modem data hold	3	-	usec	5
Th4	Modem data hold	-	15	usec	6
Th5	VIA data hold	-	15	usec	5
To1	Timeout delay	-	2.8	msec	3
Ts1	Ser. Req. setup	3	-	usec	2,4
Ts2	Timeout setup	3	-	usec	3
Ts3	Modem data setup	3	-	usec	5

Appendix - Macintosh to FDB "modem" interface

The Front Desk Bus "modem" was designed to serve as an interface between the Macintosh and the FDB data line. This modem would handle the bit level modulation/demodulation as well as provide signalling over the line. The hope is to free the Macintosh from the low level protocols and to provide a flexible interrupt driven data transfer mechanism.

The actual Macintosh - modem interface is provided by the 6522A (VIA). The modem is connected to five lines from the VIA's port B. These lines serve to communicate both data and control from the VIA to the modem.

ST0: **STATE INPUT 0**

ST1: **STATE INPUT 1**. These two input lines to the modem control the transfer of data and commands from the VIA to the modem. There are four possible states:

STATE	ST1	ST0	
S0	0	0	COMMAND
S1	0	1	DATA BYTE 1
S2	1	0	DATA BYTE 2
S3	1	1	NULL (SCLK, DIO and INT = inputs)

SCLK: **SHIFT CLOCK**. This line is bidirectional in or out. When the modem is in the null state (S3), the SCLK line is configured as an input. The line may be driven by the VIA, but it is ignored by the modem. When in states (S0, S1 and S2), the SCLK line is actively driven by the modem to clock commands (S0) or data (S1 or S2) to/from the modem.

DIO: **DATA IN/OUT**. This line is birdirectional in or out. When the modem is in the null state (S3), the DIO line is configured as an input. The line may be driven by the VIA, but is ignored by the modem. When in the command state (S0), the DIO line is an input line to the modem and is used to input serial commands. When in data byte states (S1, S2), if the command was a **TALK** command, then the DIO line is configured as an output line to output serial data received from the FDB. When in data byte states (S1, S2), if the command was a **LISTEN** command, the the DIO line is configured as an input line to input serial data transmitted to the FDB.

INT: **INTERRUPT**. This line is an output. It serves two purposes: if it is active (logical 0) at the end of data byte 1 state (S1), then a **TALK TIMEOUT** has occured on the FDB following the last command transaction; if it is active (logical 0) at the end of the data byte 2 state

(S2), then a **SERVICE REQUEST** signal was received over the FDB during the last command transaction.

FBDI: **FRONT DESK BUS IN**

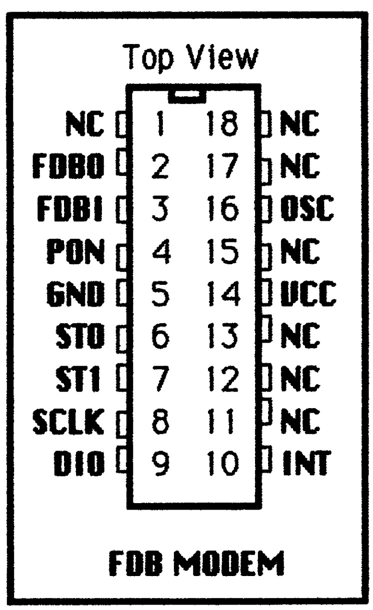
FDBO: **FRONT DESK BUS OUT**. Over these lines all devices on the FDB communicate. FDBI is an input and it is the receiver for data sent from devices on the FDB to the modem. FDBO is an open drain output and is the transmitter for data and commands sent from the modem to devices on the FDB. These two lines must be externally connected together. (Note: the FDB is open drain, a 2.2 kohm pullup resistor is required on the modem end of the bus).

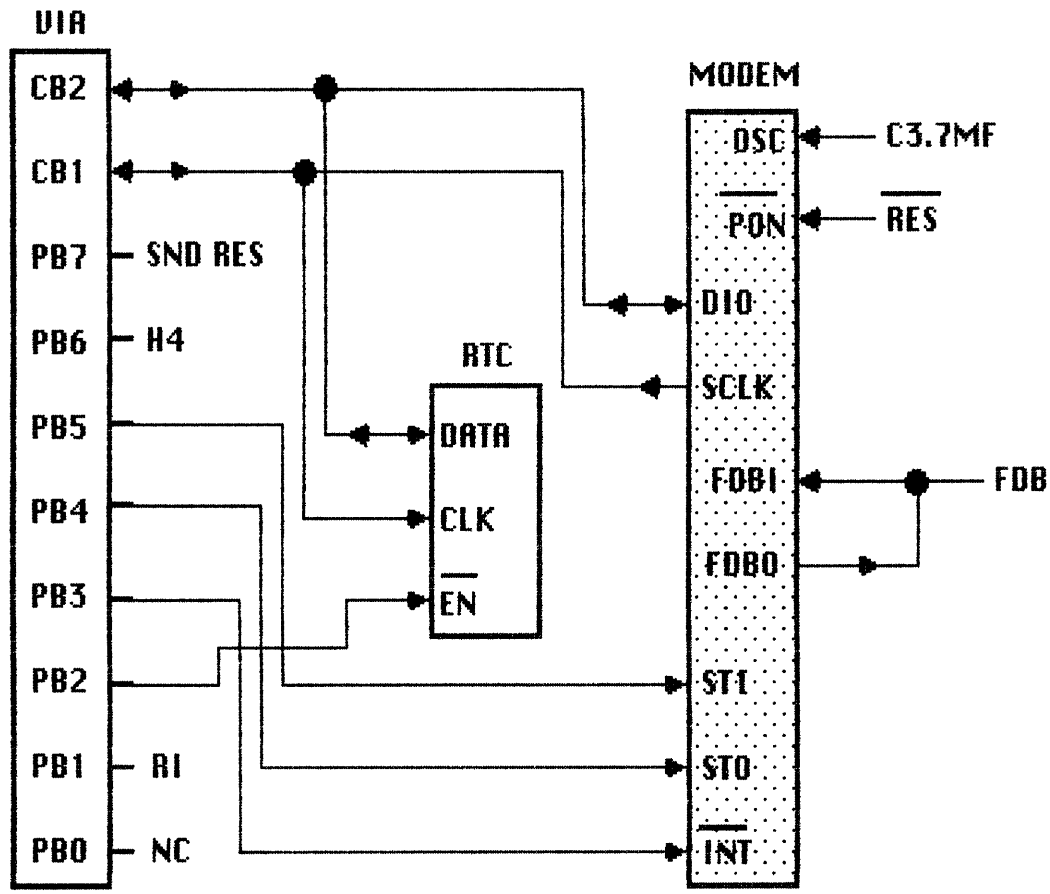
OSC: **OSCILLATOR**. This is the clock line for the microprocessor within the FDB modem. 0.8 mHz $<$ fOSC $<$ 4.0 mhz. The bit rate on the FDB is fOSC/400.

PON: **POWER ON RESET**. This is the reset line for the microprocessor within the FDB modem. This line must be held low (logical 0) for a minimum of 10 msec past the power supply rising to >4.5 V.

VCC: **POWER SUPPLY**. This is the power supply pin for the FDB modem. 4.5 V $<$ VCC $<$ 7.0 V. The maximum power supply current is 40 mA.

GND: **GROUND**. This is system ground.





Modem - VIA Wiring Diagram

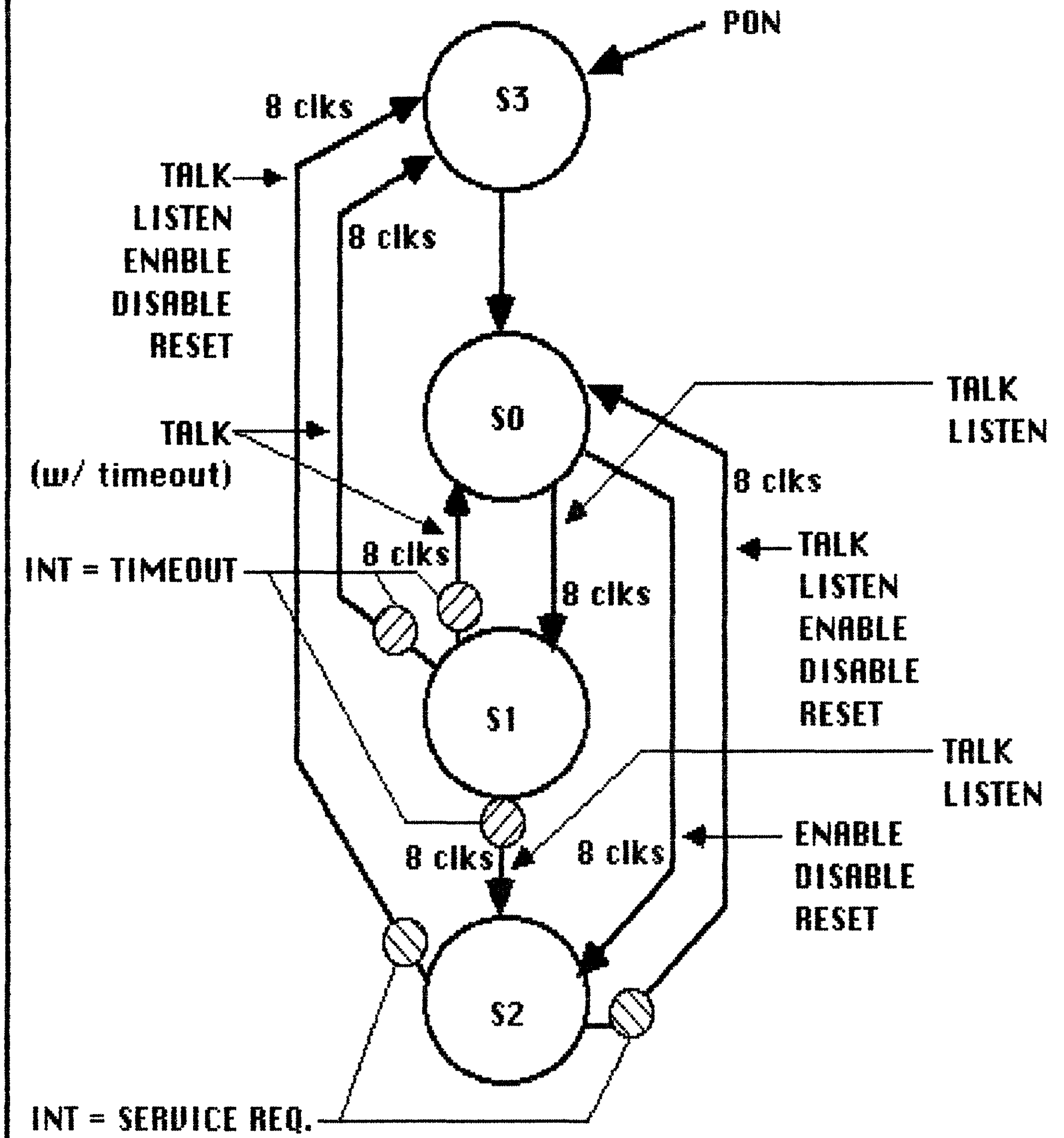


Fig. 1 FDB Modem - VIA State Diagram

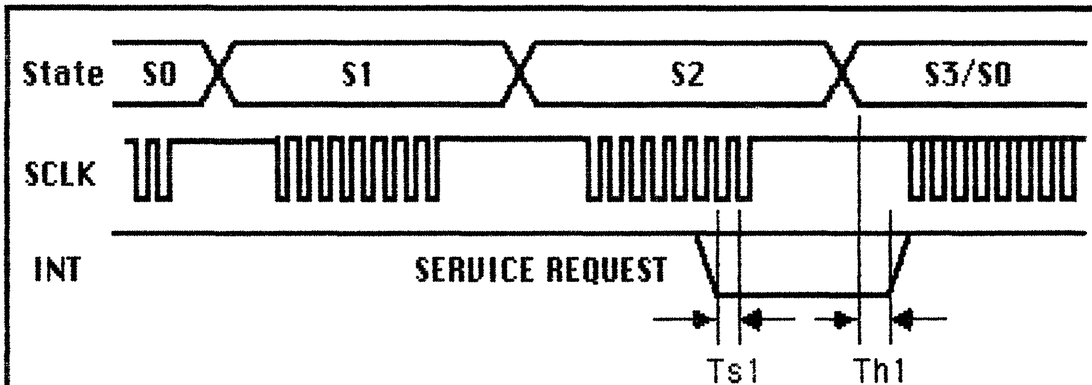


Fig. 2 LISTEN and TALK (w/o timeout)

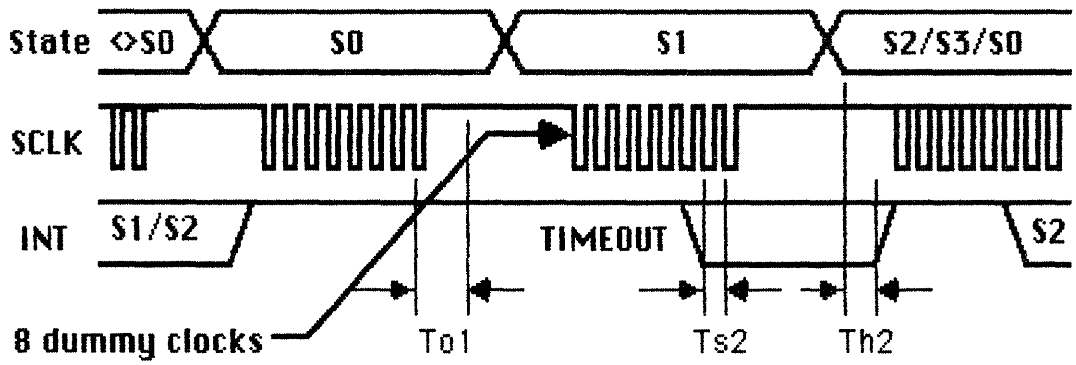


Fig. 3 TALK (w/ timeout)

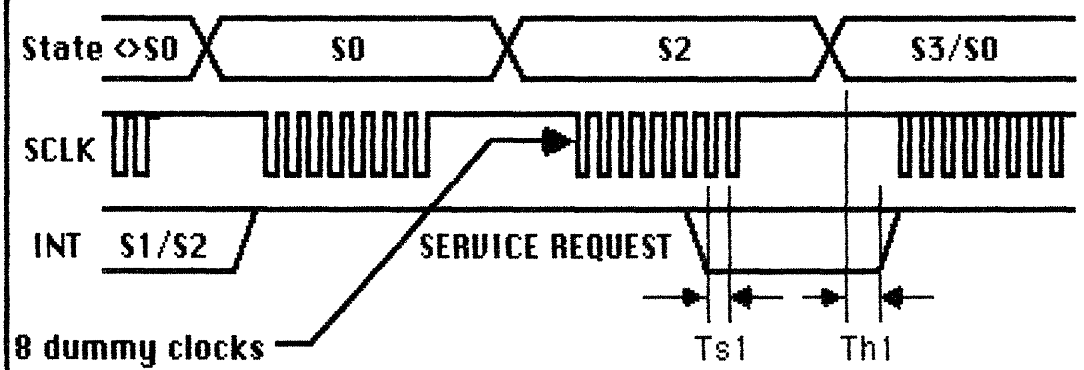


Fig. 4 ENABLE, DISABLE and RESET

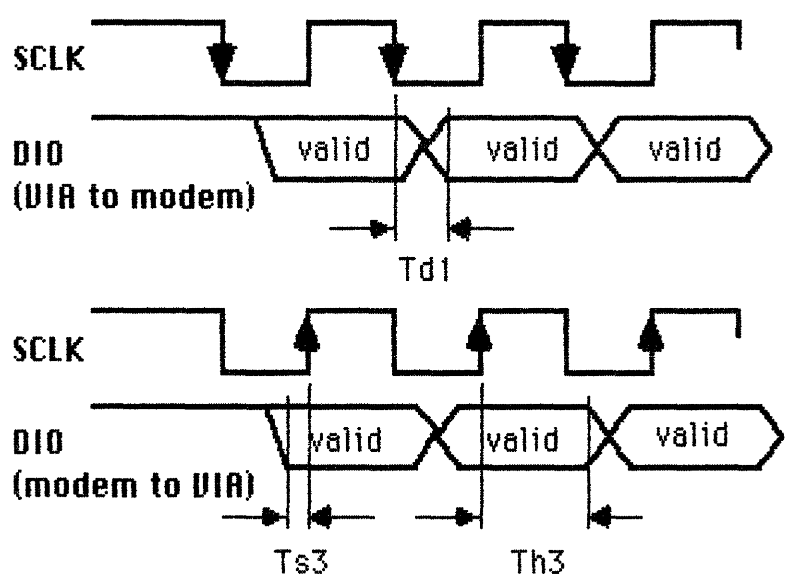


Fig. 5 VIA to modem to VIA

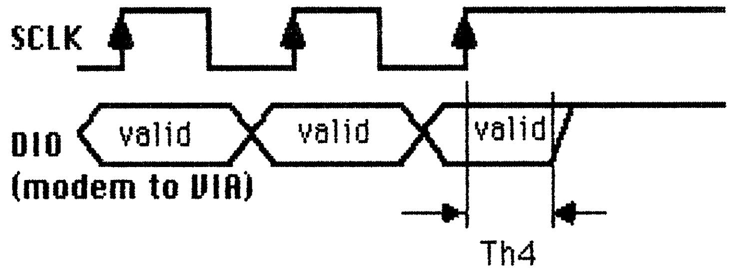


Fig. 6 modem to VIA DIO release time