FUTURE
Domain

# TMC-900 SCSI HOST INTERFACE LSI

## Design and Applications
## Manual

Price: $75.00 US

Document Number: B01-118-S0709 Date: April 20, 1987

by: Future Domain Corporation

TMC-900 Design and Applications Manual

## TABLE OF CONTENTS

# 1.0 Introduction

The following document is a detailed specification of the Future Domain IBM PC/AT Host Interface LSI (TMC-900) integrated circuit with arbitration reselection. The TMC-900 provides an eight bit interface between an IBM PC/XT/AT, called the initiator, and one to seven SCSI devices, called the targets.

## 1.1 Features

The Future Domain TMC-900 was designed to facilitate easy integration of an SCSI interface into an IBM PC or AT system. The TMC-900 is optimized for the Intel 8088/86/286 and 80386 family of microprocessors. In many systems, no glue logic is required to achieve a full SCSI interface. Application notes are available from Future Domain detailing the use of the TMC-900 with chip sets from Chips and Technology, Faraday, and ZyMos.

Features of the TMC-900 are;

- o No host I/O space is required,
- o No host DMA channels are required,
- o No host system memory is required,
- o Use of interrupts is optional.
- o Includes on board 128 byte parameter RAM.
- o Incorporates all SCSI drivers and receivers.
- o Provides full parity generation and checking.
- o 1.5MB/second transfer rate.
- o Performs full arbitration in hardware, no firmware timing loops required.
- o Implemented in 2 micron CMOS for low power

## 2.0 Applicable Documents

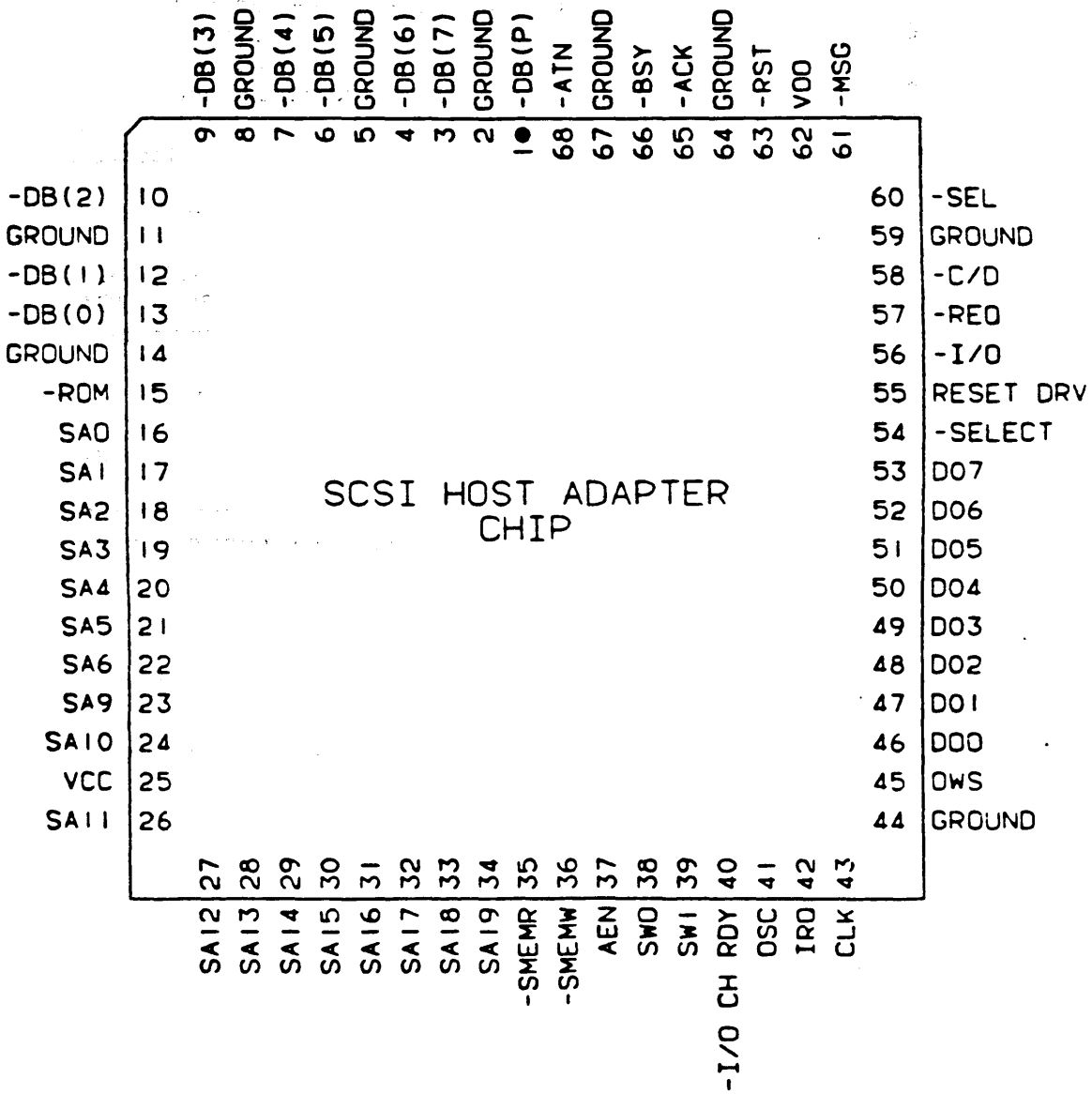
IBM Technical Reference Personal Computer AT, March 1984 (1502243).

IBM Technical Reference Personal Computer XT, April 1983 (1502237).

## 3.0 Device Description

3.1 The TMC-900 is packaged in a JEDEC 68 pin PLCC package. The package is suitable for surface mount, or may be socketed.

3.2 The TMC-900 pinout is shown below.



## 3.3 Pinout Description

| Symbol | Pins | Type | Description |
|---|---|---|---|
| -DB(P) | 1 | I/O | SCSI Parity Bit, 48MA drive |
| -DB(7) - -DB(0) | 3,4,6,7,9 10,12,13 | I/O | SCSI Data Bits, 48MA drive |
| -ROM | 15 | O | External BIOS ROM Select |
| SA0 - SA19 | 16-24,26-34 | I | Address bits |
| -SMEMR | 35 | I | Memory Read Strobe |
| -SMEMW | 36 | I | Memory Write Strobe |
| AEN | 37 | I | Address Enable |
| SW0 | 38 | I | Base Address Select |
| SW1 | 39 | I | Base Address Select |
| I/O CH RDY | 40 | O | I/O Channel Ready |
| OSC | 41 | I | 70ns Clock |
| IRQ | 42 | O | Interrupt Request |
| CLK | 43 | I | Bus Clock |
| 0WS | 45 | O | Zero wait state |
| D00-D07 | 46-53 | I/O | Data Bus, 4MA drive |
| -SELECT | 54 | O | Chip selected |
| RESET DRV | 55 | I | System Bus Reset |
| -I/O | 56 | I | SCSI Data Direction |
| -REQ | 57 | I | SCSI Transfer Request |
| -C/D | 58 | I | SCSI Control/Data |
| -SEL | 60 | I/O | SCSI Select |
| -MSG | 61 | I | SCSI Message |
| -RST | 63 | O | SCSI Reset |
| -ACK | 65 | O | SCSI Transfer Acknowledge |
| -BSY | 66 | I/O | SCSI Busy |
| -ATN | 68 | O | SCSI Attention |
| Vcc | 25,62 | | +5V |
| GND | 2,5,8,11,14,59,64,67 | | Ground |

NOTES:

1. SCSI data and control bits may be connected directly to the SCSI bus (50 pin header). No buffering is required on these signals.

2. Signals with the same name as PC/AT bus require identical characteristics.

3. AEN line may be tied LOW.

4. 0WS line is pulled low by TMC-900 when processor speedup is requested. This line should be an input to processor wait state logic. It may be used to eliminate pre-programmed memory wait states.

5. I/O CH RDY is pulled when wait states are required during a memory cycle.

# 4.0 Functional Description

## 4.1 SCSI Interface

The TMC-900 communicates to the SCSI data bus utilizing the Small Computer System Interface (SCSI) protocol, including Arbitration and Reselection. (See sections 2.0 and 5.0. for the interface description and for pin assignments.) The TMC-900 optionally generates and checks parity on the SCSI data bus.

## 4.2 Host Interface

The TMC-900 utilizes 8 bit memory mapped I/O to communicate with the host. See the 1502243 Technical Reference Personal Computer AT for I/O Channel pin assignments and signal descriptions.

## 4.3 TMC-900 Memory

The TMC-900 contains a 6K x 8 address space for an EPROM, and a 512 x 8 address space for static RAM (128 bytes actually available). Either a 2764 or a 27128 EPROM can be used. The host writes and reads the static RAM and reads the EPROM via the memory mapping specified in section 4.5.

## 4.4 TMC-900 Memory Address Recognition

Two pins, SW0 and SW1, are provided to select the four possible starting addresses for the TMC-900:

0 = PIN OPEN
1 = PIN GROUNDED

| SW0 | SW1 | Starting Address |
|---|---|---|
| 1 | 1 | CA000 |
| 0 | 1 | C8000 |
| 1 | 0 | CE000 |
| 0 | 0 | DE000 |

SW0 and SW1 have internal pullup resistors. the pins may be left open if desired (address=CA000).

## 4.5 Memory Mapped Functions

The TMC-900 performs the following functions: Write RAM, Write Control Register, Write SCSI, Read ROM, Read RAM, Read Status, and Read SCSI.

**Memory Write Address Ranges**

| SW0 | SW1 | Write RAM (512) | Write Control Reg (512) | Write SCSI (1K) |
|---|---|---|---|---|
| 1 | 1 | CB800 - CB9FF | CBA00 - CBBFF | CBC00 - CBFFF |
| 0 | 1 | C9800 - C99FF | C9A00 - C9BFF | C9C00 - C9FFF |
| 1 | 0 | CF800 - CF9FF | CFA00 - CFBFF | CFC00 - CFFFF |
| 0 | 0 | DF800 - DF9FF | DFA00 - DFBFF | DFC00 - DFFFF |

**Memory Read Address Ranges**

| EP1 | EP2 | Read ROM (6K) | Read RAM (512) | Read Status (512) | Read SCSI (1K) |
|---|---|---|---|---|---|
| 1 | 1 | CA000-CB7FF | CB800-CB9FF | CBA00-CBBFF | CBC00-CBFFF |
| 0 | 1 | C8000-C97FF | C9800-C99FF | C9A00-C9BFF | C9C00-C9FFF |
| 1 | 0 | CE000-CF7FF | CF800-CF9FF | CFA00-CFBFF | CFC00-CFFFF |
| 0 | 0 | DE000-DF7FF | DF800-DF9FF | DFA00-DFBFF | DFC00-DFFFF |

### 4.5.1 Write RAM

The 128 x 8 RAM is written with data from the host at the specified address.

### 4.5.2 Write Control Register

The eight bit Control Register receives command information for the SCSI and for the TMC-900 from the host. This register can not be read by the host.

| Control Register Bit | Description |
|---|---|
| 0 | SCSI Reset |
| 1 | SCSI Select |
| 2 | SCSI Busy |
| 3 | SCSI Attention |
| 4 | Start Arbitration |
| 5 | Enable SCSI parity generation |
| 6 | Enable Interrupt on select |
| 7 | Enable SCSI bus drivers |

Bit 0 - SCSI Reset. The SCSI reset line is driven directly off of this signal. Caution must be used when setting this bit, especially in a system using arbitration as this resets all target devices.

Bit 1 - SCSI Select. The SCSI Select line is driven directly off of this control bit. It is the firmware's responsibility to assure that bus arbitration is complete before using this signal.

Bit 2 - SCSI Busy. The SCSI Busy line is driven directly off of this control bit. If the TMC-900's arbitration logic sequencer is used, this bit is not needed for the initial bus arbitration phase. It is needed for reselection.

Bit 3 - SCSI Attention. The SCSI Attention line is driven directly off of this control bit. It is the responsibility of the firmware to recognize the message out phase set up by the target and to output the appropriate message byte.

Bit 4 - Start arbitration. The arbitration sequencer operation is started. See Section 4.7 for further discussion of the arbitration sequencer.

Bit 5 - Enable SCSI Parity. The operation of the parity generator on transmitted data is enabled. Without this bit, no parity is generated on the SCSI parity line. This bit does not impact the detection of parity and the setting of the parity error bit in the input status.

Bit 6 - Enable Interrupt. An interrupt is generated when the select line on the SCSI bus is asserted and this control bit is set. This allows for response to reselection requests and arbitration complete.

Bit 7 - Enable SCSI bus. The initiator in an arbitrated system can only drive the data bus when it is the selected device, or during arbitration in the bus free state. The drivers to the SCSI data bus are enabled based on several conditions. They are the SCSI I/O control line is deasserted, and either this control bit is set or the TMC-900 is arbitrating for the bus.

### 4.5.3 Write SCSI

Eight bits of data from the host are sent to the SCSI data bus. An SCSI Acknowledge is generated by the TMC-900 at the end of the Write SCSI operation if the SCSI Request is asserted.

### 4.5.4 Read ROM

A data byte from the addressed location in the EPROM is returned to the host.

.pa

### 4.5.5 Read RAM

A data byte from the addressed location in the Static RAM is returned to the host.

### 4.5.6 Status Register

The eight bit Status Register returns status information to the IBM PC as follows:

| Status Reg. Bit | Description |
|---|---|
| 0 | SCSI Busy |
| 1 | SCSI Message |
| 2 | SCSI I/O |
| 3 | SCSI Command/Data |
| 4 | SCSI Request |
| 5 | SCSI Select |
| 6 | SCSI Parity Error |
| 7 | Arbitration Complete |

Bits 0-5, SCSI Status. This is a direct reading from the status lines on the SCSI bus.

Bit 6 - SCSI Parity Error. The bus parity is checked during each read operation. If an error is present, this status bit is set and remains set until the next write to the control register.

Bit 7 - Arbitration complete. The arbitration sequencer has arbitrated for the SCSI bus and won. This status is cleared when the Start Arbitration control bit is set to 0. See section 4.7 for a further description of the arbitration sequencer.

### 4.5.7 Read SCSI

A byte of data from the SCSI data bus is returned to the host. A SCSI Acknowledge is returned at the end of the Read SCSI operation if the SCSI request is asserted.

## 4.6 Interrupt Operation

4.6.1 The interrupt is for the purpose of reselection or arbitration complete. When the interrupt control bit is set and the SCSI Select is asserted, the interrupt line is asserted.

4.6.2 The IRQ output of the TMC-900 should be connected to an interrupt input of the 8251 interrupt controller. Jumpers can be provided externally to provide selection of alternate interrupt lines.

4.6.3 The interrupt from the TMC-900 is turned off by deasserting the interrupt enable bit in the control register.

## 4.7 Arbitration Sequencer

4.7.1 The arbitration process is started by asserting the Start Arbitration bit in the control register. When the arbitration process is complete, the arbitration complete bit is set in the status register.

4.7.2 When the start arbitration bit is set in the control register, the arbitration sequencer waits for the bus free phase. When the bus free phase occurs, the SCSI output register is driven onto the bus and the SCSI Busy control line asserted.

4.7.3 There is a delay of 2.4 usec. If during this 2.4 usec delay, the Select line is asserted or another bit is asserted on the bus of a higher priority (0-7 with 7 as highest priority), the arbitration sequencer returns to waiting for a bus free phase.

4.7.4 At the conclusion of the 2.4 usec, the SCSI Select control line is asserted and the arbitration complete status is asserted.

4.7.5 The arbitration sequencer is used as follows:

Output the TMC-900 address bit to the SCSI write port.

Set the Start Arbitration control bit.

Wait for the Arbitration Complete status bit, or wait for the interrupt on SCSI Select.

"OR" the desired target ID bit into the SCSI data output register.

Assert the SCSI Select control line while deasserting the Start Arbitration control bit. The driving of the SCSI Busy bit stops when the Start Arbitration control bit is deasserted.

Poll waiting for the target device to assert SCSI Busy.

## 4.8 Send Message

4.8.1 This is handled by firmware. After the target has been selected and is busy, assert the SCSI Attention line via the control register.

4.8.2 When the SCSI Status lines indicate that the message can be sent to the selected target, deassert the SCSI Attention line and write the message to the SCSI data port. The handshake acknowledge is handled the same as for data transfers.

## 4.9 Reselection

4.9.1 Determine that the SCSI Select line is asserted by either polling the SCSI status register or via interrupt. Read the data bus to see if the TMC-900 is being selected. If it is not, continue to poll the SCSI data bus until the Select line is deasserted. If reselection is expected, continue to poll or wait for an interrupt until the correct address is detected.

4.9.2 When the correct address bit is detected, assert the SCSI Busy and wait for the deassertion of SCSI Select.

4.9.3 Deassert SCSI Busy when SCSI Select is deasserted. Read SCSI status bits to determine what operation is to follow.

## 5.0 Programming

The TMC-900 is available in a turn key configuration. Software is available for the PC/MS-DOS operating system, the Xenix operating system, and Novell Advanced Netware. Utility programs are available for formatting and defect handling. Disk, tape and optical disk software packages are also available.

If you use the existing software packages, you need not program the TMC-900 directly. Future Domain fully supports and licenses these software packages to you. As a system designer, you can be ready to go from the time your first prototype is available.

If you have special requirements not addressed by Future Domain software, you have three options. A ROM BIOS level interface is available that performs basic I/O operations in an IBM PC/AT compatible mode. The ROM BIOS has several extensions to make programming of SCSI peripherals easier. A set of OEM object modules are available. These modules (the OEM SCSI interface) provide a higher level interface to the TMC-900. Finally, you can do direct I/O to the TMC-900 LSI.

## 6.0 ROM BIOS Interface

The ROM BIOS is modeled after the IBM PC/XT/AT ROM BIOS and utilizes the standard Interrupt 13 mass storage vector. As a subset, it contains enough capability to allow booting from devices that can be made IBM media compatible.

In addition to the basic set of commands, extensions are available to provide direct control of SCSI devices for the purposes of control, identification, formatting, extended addressing etc.

The ROM BIOS also maintains information in a private RAM table that may be useful to the writer of device drivers or utilities. This information is also made available through extensions to the INT 13 BIOS call.

## 6.1 ROM BIOS Command Reference

### 6.1.1 SUMMARY LIST OF BIOS COMMANDS - Currently supported by Future Domain TMC-900

| Code | Function |
|---|---|
| 00 | Initialize the System |
| 01 | Request Sense |
| 02 | Disk Read |
| 03 | Disk Write |
| 04 | Disk Verify |
| 05 | Mode Select |
| 06 | Format Unit with passed list |
| 07 | Format Unit, no list |
| 08 | Return Parameters |
| 09 | Not Used |
| 0A | Not Used |
| 0B | Not Used |
| 0C | Disk Seek |
| 0D | Not Used |
| 0E | Not used |
| 0F | Not used |
| 10 | Request Sense |
| 11 | Recalibrate |
| 12 | Drive stop |
| 13 | Request Sense |
| 14 | Adapter Diagnostic |
| 15 | Return DASD |
| 16 | Not Used |
| 17 | Not Used |
| 18 | Identify ROM |
| 19 | Read Drive Capacity |
| 1A | Read Cylinder Capacity |
| 1B | Locate Table |
| 1C | Locate Drive Letter Table |

### 6.1.2 REGISTER USAGE - INPUT TO INT 13 CALL

The following are the general register assignments. Some assignments apply to all calls, and other assignments are dependent on the specific function.

All calls have the following assigned:

AH - Function Number (ie 2=Disk Read)
DL - Drive number (ie 80h for first drive)

The general assignment for other registers is as follows:

ES:BX - Pointer to caller buffer
AL - Number of sectors
CH - Cylinder number, lower 8 bits
CL - Bits 7,6 - high 2 bits of cylinder number
Bits 0-5 - Sector number
DH - Head number

### 6.1.3 ERROR CODE RETURN

The returned error codes are those used by the Int 13h conventions. The recommended mapping to DOS error codes is also given.

| Code | Error | DOS Code | DOS Error |
|---|---|---|---|
| 03h | Write Protect | 00 | Write Protect |
| 20h | Bad Controller | 01 | Unknown Unit |
| FFh | Sense Fail | 02 | Not Ready |
| 80h | Timeout | 02 | Not Ready |
| 01h | Bad Command | 03 | Unknown Command |
| 10h | Bad ECC | 04 | CRC Error |
| BBh | Undefined Error | 05 | Bad Drive Request |
| 40h | Bad Seek | 06 | Seek Error |
| 04h | Record Not Found | 08 | Sector not Found |
| 0Bh | Bad Track | 0A | Write Fault |
| 02h | Bad Address Mark | 0B | Read Fault |
| 07h | Drive Parameter Failure | 0C | General Failure |
| 11h | Data corrected | - | No error |
| 06h | Media changed | 0F | Media changed |

The error codes returned from the SCSI disks are not consistent. As a result, the error codes can not accurately reflect all drives. If more detailed information is required for a specific operation, use the Int 13h code 1Bh to locate the Sense Bytes for further analysis.

If there is an error on any request, one of the above codes is returned in AH/AL and the carry flag is set. If there is no error, AH=AL=0 and the carry flag is cleared.

## 6.2 Description of ROM BIOS Codes

**Code 00 - Initialize the System**

**Input registers - none.**

**Special output registers - none.**

Operation:

The SCSI Reset line is asserted and deasserted followed by a delay of about 1 second. After the delay, each drive attached to the adapter is issued a Recalibrate command, and if an error is returned, the status is read via a Request Sense command. This clears out any error status that may have been caused by the reset.

The status from the request sense command is not returned to the caller as this is done for every disk connected to the adapter.

**Code 01** - Request Sense

Input registers - none.

Special output registers - none.

Operation:

A SCSI Request Sense is issued to the specified Disk. The results of the command are converted to the BIOS Error Code and returned in AH/AL.

**Code 02 - Disk Read**

**Input registers - as defined under general assignment.**

Special output registers - none.

Operation:

Two internal parameters are used to convert the caller's head, sector, and cylinder number to a SCSI logical block number. These parameters are the heads per cylinder and sectors per track numbers. The source of these numbers is discussed under Codes 08 and 19.

If the SCSI bus is not in an idle condition at the start of the command execution, an Initialize System (Code 00) is executed prior to the execution of this command to clear up the condition.

A SCSI Read command (08) is issued to the specified disk. The BIOS then waits for SCSI Request to be asserted. When it is asserted, a check is made to verify that SCSI C/D is not asserted. If C/D is asserted, operation proceeds directly to the Status/Message phase.

Data is transferred in blocks of 512 bytes. After the request is detected, a block of 512 bytes is transferred to the caller's buffer. If the transfer is complete, the operation proceeds to the Status/Message phase, else it again waits for the SCSI Request line to be asserted.

If there is an error reported in the Status/Message phase of the operation, the SCSI Request Sense Command is issued and the results returned as an error in AH/AL.

**Code 03 - Disk Write**

Identical to the Disk Read, Code 02.

**Code 04 - Disk verify**

**Input registers - As defined under general assignment except for the ES:BX. There is no buffer associated with this command.**

Special output registers - none.

Operation:

**This command is executed in the same manner as the Disk Read with the exception that the data read from disk is not stored anywhere. The purpose of this command is to verify that there are no read errors detected.**

**If there is an error reported, the SCSI Request Sense command is issued and the results of the Sense Command returned in AH/AL.**

**Code 05 - Mode Select**

Input registers - ES:BX points to a buffer containing the bytes that are to be output during the data phase of the mode select command. Byte 0 contains the length the buffer of bytes to be output. Bytes 1-n are the data bytes.

Special output registers - none.

Operation:

The purpose of this command is to send the required parameters to the device in preparation for the issuing of the unit format command. The SCSI Mode Select (15h) command is issued to the specified device. The length of the data phase is taken from byte 0 of the caller's buffer. During the data phase of the command, successive bytes are taken from the caller's buffer and sent to the device.

A check is made for SCSI Request asserted, and SCSI C/D deasserted between each byte. If C/D is asserted or the buffer is exhausted, the operation proceeds to the Status/Message phase.

If an error is detected during the Status/Message phase, a SCSI Request Sense is issued and the results returned in AH/AL.

**Code 06** - Format Unit with passed list

**Input registers** - ES:BX points to a buffer containing the bytes of data that are to be output during the data phase of the command. Byte 0 contains the MSB and byte 1 the LSB of the length of the number of bytes to be output during the data phase. Bytes 2-n contain the data that is to be output. If the length is 0, no bytes are output.

DH contains flags that are part of the format command. The specific definition of the flags to the device are device dependent. This byte is or'd with byte 1 of the command before it is output to the device.

AL contains the interleave factor. This is output as byte 4 of the command.

Special output registers - none.

Operation:

This command is normally used to send defect lists to the drive for formatting. By changing the flags and the contents of the buffer, this could be a format cylinder command, format unit command, format with manufacture's defect list, format with passed list, format with spares, etc. The specific operation is dependent on what the specific device manufacture has implemented.

The SCSI Format Unit (04) command is issued to the device. The contents of the caller's DH register are or'd with the LUN to form byte 1 on the command. The caller's AL register is sent as byte 4 for the interleave.

The length of the data phase is taken from bytes 0 and 1 of the caller's buffer. Note that the MSB is in byte 0 and the LSB in byte 1, the same as for values sent over the SCSI bus. During the data phase of the command, successive bytes are taken from the caller's buffer and sent to the device.

A check is made for SCSI Request asserted, and SCSI C/D deasserted between each byte. If C/D is asserted or the buffer is exhausted, the operation proceeds to the Status/Message phase. There is no timeout for this command. Many devices pickup the data bytes such as for defect management as they are needed during the formatting operation. As a result, there is no predetermined timeout interval. If the device hangs, the system must be restarted.

If an error is detected during the Status/Message phase, a SCSI Request Sense is issued and the results returned in AH/AL.

**Code 07** - Format Unit, no list

Input registers - AL contains the interleave which is output as byte 4 of the command.

Special output registers - none.

Operation:

The SCSI Format Unit (04) command is issued to the device. Byte 1 contains only the LUN with no special flags set. Byte 4 is the interleave. After the command is issued, the BIOS waits for the completion of the operation looking for the Status/Message phase of the operation. There is no predetermined length of time to use for a timeout, so if the device hangs, the only method of recovery is to reboot the system.

**Code 08** - Return Parameters

Input registers - none.

Special output registers -

DH- Number of heads-1
DL- Number of drives in HF_NUM
CH- LSB 8 bits of number of cylinders-1.
CL- Number of sectors per track + 2 MSB of cylinders - 1.

Operation:

This function returns the parameters that are used to convert the caller's disk address specification into a logical sector number to be issued to the SCSI device. The ROM BIOS calculates the logical address by:

SECTOR=cyl*(HEADS*SECTS)+head*SECTS+(sect-1)

SECTOR- Logical sector number
HEADS - Number of heads reported for device
SECTS - Sectors per track reported for device
cyl - Caller specified cylinder number
head - Caller specified head number
sect - Caller specified sector number

Note that these values need have no relationship to the actual values on the physical drive for cylinders, heads, and sectors per track. These are only "funny" numbers used for the Int 13h call.

Before the caller makes an Int 13h call to read, write, or verify a logical sector, the reverse of the above algorithm needs to be performed on the desired logical sector number for the setting up of the registers.

This may appear to be a strange way to do things, but it is the "IBM Standard".

**Code 0C - Disk Seek**

Input registers - As defined under general usage with the exception that there is no use of the buffer pointer ES:BX.

Special output registers - none.

Operation:

The caller's parameters are converted to a logical sector address in the same manner as is done in the Disk Read command. A SCSI Seek command is issued. There is probably no need for this command as all SCSI read and writes have implied seeks.

**Code 10 - Request Sense**

This is the same as Code 01.

Code 11 - Recalibrate

Input registers - none.

Special output registers - none.

Operation:

A SCSI Recalibrate (01) command is issued to the specified disk. The resultant status of the command is returned in AH/AL.

**Code 12 - Drive Stop**

Input registers - none.

Special output registers - none.

Operation:

A SCSI stop (1Bh) command is issued to the specified disk. Some manufactures use this command to park the heads in preparation for moving.

**Code 13 - Request Sense**

**This is the same as code 01.**

**Code 14 - Adapter Diagnostic**

**Input registers - none.**

**Special output registers - none.**

Operation:

> **A complete wrap around test is run on the SCSI data path. 256 data patterns are placed on the SCSI data bus and read back for comparison. If there is any difference, an error is returned in AH/AL.**

**Code 15 - Return DASD**

**Input registers - none.**

Special output registers -

**AX- 300h to mean that this is a hard disk.**

**CX:DX - Capacity of the addressed disk. If this is zero, the execution of the SCSI Read Capacity command failed. This usually means that the disk need to be formatted.**

**Code 18** - Identify ROM

Input registers - none.

Special output registers -

AX- 4321h. This code identifies this as a Future Domain ROM BIOS.

BH - Number of Disks being supported by this ROM BIOS.

BL - Index number for this disk.

CH - ROM Firmware version number.

CL - Drive type code.

Operation:

The purpose of this command is to locate and identify the codes that are required for use of the SCSI disks. The Int 13 calls for the SCSI drives are addressed to the hard drives, ie disk 0 is 80h, disk 1 is 81h, . . . Note that this addressing includes internal IBM disks as well as the SCSI disks. This command identifies what the base address is for SCSI disks.

Assume that there is one internal hard disk and four SCSI disks. Addressing this command to hard disk address 82h would result in BH=4 and BL=1. That means that there are 4 SCSI disks and that this is the second disk, ie they are numbered 0, 1, . . . The base address for the SCSI disks is therefore 82h-1=81h. Or, SCSI disk 0 is address 81h, SCSI disk 1 is address 82h, . . .

If this command is issued to an IBM disk, it will return an error code.

**Code 19** - Read Drive Capacity

**Input registers - none.**

Special output registers - Same as for Code 15.

Operation:

A SCSI Read Capacity (25) Command is issued to the specified disk. All of the command bytes are set to zero. The following parameters are then updated in the internal table for the disk. See Code 1Bh for a complete explanation of the internal table.

Total Capacity - This is a double word parameter.

The number of heads, sectors per track, and cylinders are calculated as follows. The number of sectors per track is fixed at 17 because there are some RAM BIOS systems which make that as a fixed assumption and will not work with a number different than this. The number of cylinders can not exceed 1024 as the field is not any larger. Therefore, the total capacity is divided by 1024 and the result divided by 17. The result is incremented by one and becomes the number of heads. The new heads value is multiplied by 17 and divided into the total capacity again with the result becoming the total number of cylinders to use. Note that these numbers will probably not match the physical characteristics of the disk, but it doesn't matter.

**Code 1A** - Read Cylinder Capacity

Input registers - none.

Special output registers - CX:DX is the address of the last logical block of the specified cylinder.

Operation:

The SCSI Read Capacity (25h) Command is issued to the specified disk with the PMI bit set in byte 8 bit 0 of the command sequence. Note that not all disks support this operation. It is sometimes used for incremental formatting of the disk by cylinder and marking defects by cylinder as on the Adaptec controllers.

**Code 1B** - Locate Table

Input registers - none.

Special output registers - ES:BX points to the internal table maintained for the specified disk.

The returned pointer is to a table organized as follows:

| Type | Function |
|---|---|
| Char | Bit 1 is set to cause checking of SCSI bus parity. |
| Int | Total number of cylinders calculated in Read Capacity. |
| Char | Total number of heads calculated in Read Capacity. |
| Char | Sectors per track. |
| Char | SCSI selection codes. These are determined during the ROM boot process. Bits 5-3 are the SCSI address of the disk. There is no LUN support as drives with embedded SCSI controllers only have a single LUN. |
| Char | Ready flag used internally in the ROM. This must be non zero for the disk to be used. |
| Char[4] | The four bytes from the last SCSI Request Sense command that occurred as a result of an error are saved. This information can be used to assist in further defining any errors that occur or for diagnostic printouts. |
| Char[10] | The command bytes sent that caused the SCSI Request Sense to be issued. This is helpful for diagnostic help. |
| Long | Total disk capacity in sectors as returned by the SCSI Read Capacity command. Note that the Read Capacity command actually returns the address of the last sector. This is that number +1. |
| Char[24] | Name of the disk manufacture as returned by the SCSI Inquiry command. |

**Code 1C - Locate Drive Letter Table.**

**Input registers - none.**

**Special output registers - ES:BX points to a table of drive letters assigned by the loadable driver.**

**Operation:**

**This table is filled with the assigned DOS disk drive letters when the Future Domain Loadable driver is executed. The table is then used to specify the drive letter for utilities. It is not necessary to use this feature to make the ROM BIOS operational.**

## 7.0 OEM Interface

These routines allow the user to build their own set of specialized utility or driver programs and communicate directly with their device without having to worry about the low level hardware interface commands, SCSI bus phase issues, and hardware teammates. The routines can be executed from either "C" language or assemble language programs.

There are two routines available. They are as follows:

```
oemreset ();
oemscsi (&oemtable);
```

The first routine, oemreset(), is used to assert and deassert the hard SCSI Reset line on the SCSI bus. This function should only be used in a condition where there is a timeout on the SCSI bus as indicated in the returned status from a call. Using this routine can cause initial error returned status from your device - Consult your device manual for further information on clearing out this status. This operation also requires a recovery period on the part of all devices on the SCSI bus.

The second routine, oemscsi(&oemtable), is the routine used to send commands to devices over the SCSI bus. It then handles the data phase, the status phase, and the message phase of the request. This interface does not support the arbitration reselection features of the host adapters.

The key to the operation of the oemscsi routine is the structured data table which is formatted as follows:

```
struct oemtable
  {
  unsigned char (far *cmdblock)[];
  unsigned char (far *datablock)[];
  long expectlen;
  long actuallen;
  int timeout;
  int blkfactora;
  int blkfactorb;
  unsigned char scsistatus;
  unsigned char scsimsg;
  unsigned char scsibits;
  unsigned char scsiaddress;
  unsigned char scsiparity;
  int adaptererr;
  } ;
```

This is an explanation of the parameters that are in the structure. It is assumed that any routines written in "C" are compiled with the /Zp option to specify packed arrays.

unsigned char (far *cmdblock)[]

Far pointer to a character array that contains the bytes that are to be sent to the device during the command phase of the SCSI bus operations. Data is sent from this array, one byte at a time, until the SCSI bus phase changes from command phase to another phase.

unsigned char (far *datablock)[]

Far pointer to a character array where data from the SCSI bus data phase is either to be written into, or read from. The direction of the operation is determined by checking the direction specified by the SCSI I/O status line during the data phase.

long expectlen

The expected length of the data phase in number of bytes transferred is specified. This is a type "long" variable. This may either be the actual expected length of the transfer, or the size of the buffer into or from which the transfer is to take place. This number should not exceed the length of the actual buffer. If the data phase is longer than the specified buffer size, the extra data is read on an input, or the data phase is filled with unknown data on an output.

long actuallen

The actual length of the transfer in number of bytes transferred is returned to the caller here. This is a type "long" variable. If this exceeds the value of expectlen, the data in excess of expectlen has been discarded and is not valid or predictable. If this number is less than expectlen, then only actuallen bytes of data are valid.

int timeout

There is an expected length of time that a specific operation should take on the SCSI bus. If for some reason the operation exceeds this length of time, there is a problem and some type of error correction needs to take place. This is the timeout interval in seconds to wait for an operation to complete. The maximum interval that can be used is 1800 seconds.

There are two time-outs that are built into the scsioem routine. They are the timeout waiting for the device to respond to the assertion of the SCSI Select line with a SCSI Busy, and the timeout waiting for the SCSI bus free phase. These time-outs are both 3 seconds.

int blkfactora
int blkfactorb

The TMC adapters use a high speed transfer technique that depends on a block of data being transferred in rapid succession with no more than a 20 usec delay between bytes of data. At the end of the block, the software will pause for a maximum of timeout seconds waiting for the device to again assert the SCSI Request line indicating that there is more data available.

Two values are provided for the blocking factor. In most cases, both values will be the same. The routine first transfers a block of blkfactora bytes in length and then transfers a block of blkfactorb bytes in length. For example, in doing a read from a disk system with 512 byte sectors, both values would be set to 512.

Some devices may have a characteristic where the block size may be for example 4096, but only the first 4095 bytes can be transferred in a burst mode, and the last byte needs to be transferred separately. In this example, blkfactora would be 4095, and blkfactorb would be 1.

In commands where the transfer of the information cannot be guaranteed to be blocked, for example in the reading of Request Sense status bytes, then these two values should be set to 1. This assures that the system will wait for any hardware delays.

unsigned char scsistatus

The SCSI status byte returned from the device at the conclusion of the transfer is returned here. The format of this byte will be specified in your device manual.

unsigned char scsimsg

The SCSI message byte returned from the device at the conclusion of the transfer is returned here. This byte is normally zero. If it is not, refer to your device manual for a discussion of its meaning.

unsigned char scsibits

If there is a timeout condition (indicated by the error return status in int adaptererr), the SCSI bus status bits are returned for examination to determine what is causing the hang-up. Note that the oemprint routine prints out the bus status to indicate the condition of these bits. They are as follows:

Bits 7-5 Not Used
Bit 4 - SCSI Request
Bit 3 - SCSI Command/Data
Bit 2 - SCSI I/O
Bit 1 - SCSI Message
Bit 0 - SCSI Busy

unsigned char scsiaddress
This the device's SCSI address. Valid values are 0-7. Check the configuration of your device and your device manual to determine how to set and determine this address.

unsigned char scsiparity
Should parity be checked on the SCSI bus? This is a function of your device and the specific TMC product that is being used. Do not specify parity checking for a device or adapter that does not support parity as each transfer will return an error condition. The summary of adapters in section 1 tells which support parity. Set this =0 for no parity checking and !=0 for parity checking.

int adaptererr
Error status related to the transfer is returned. If there is no error, a 0 is returned. Valid error values are as follows:

| | |
|---|---|
| -1 | Timeout waiting for initial bus free phase |
| -2 | Timeout during Selection phase |
| -3 | Timeout during Command phase |
| -4 | Timeout during Data phase |
| -5 | Timeout during Status phase |
| -6 | Timeout during Message phase |
| -7 | Parity error detected during transfer |
| -16 | Error locating TMC host adapter |
| +1 | Buffer overflow. actuallen > expectlen |
| +2 | Buffer underflow. actuallen < expectlen |

In order to execute a SCSI command, fill in the command bytes, set the pointers to the command array and the buffer, set the expected length, set the blocking factor terms, set the scsiaddress character to the address of your device, set the scsiparity character, and execute the command. An example of the operation is included in the OEM Developers Toolkit. This kit may be purchased from Future Domain.

## 8.0 Direct Interface

You may write software to directly access the control registers and ports of the TMC-900. These ports and registers are described in section four of this manual.

The SCSI data port appears as a range of addresses. It can be read or written to using the REP MOVSB instruction.

Using single byte memory reference instructions to access control ports.

# 9.0 System Interfacing

## 9.1 80286/386 Interface Example

Chips & Technology offer chipsets for both 80286 and 80386 designs. These chipsets allow the designer to implement a PC AT compatible system using a minimum of discrete components. For I/O, the system architect was forced to use expensive bus cards that offer limited performance.

Today, the TMC-900 SCSI LSI can provide either a one, two or three chip I/O solution to the system designer. With the TMC-900, up to six fixed disks, one tape drive, or an optical disk drive can be attached to the system. The TMC-900 offers up to three times the performance of bus based fixed disk controllers.

This application note shows how to interface the TMC-900 to a Chips & Technology based design. It also discusses the design tradeoffs of the various design approaches.

## Single Chip SCSI Solution

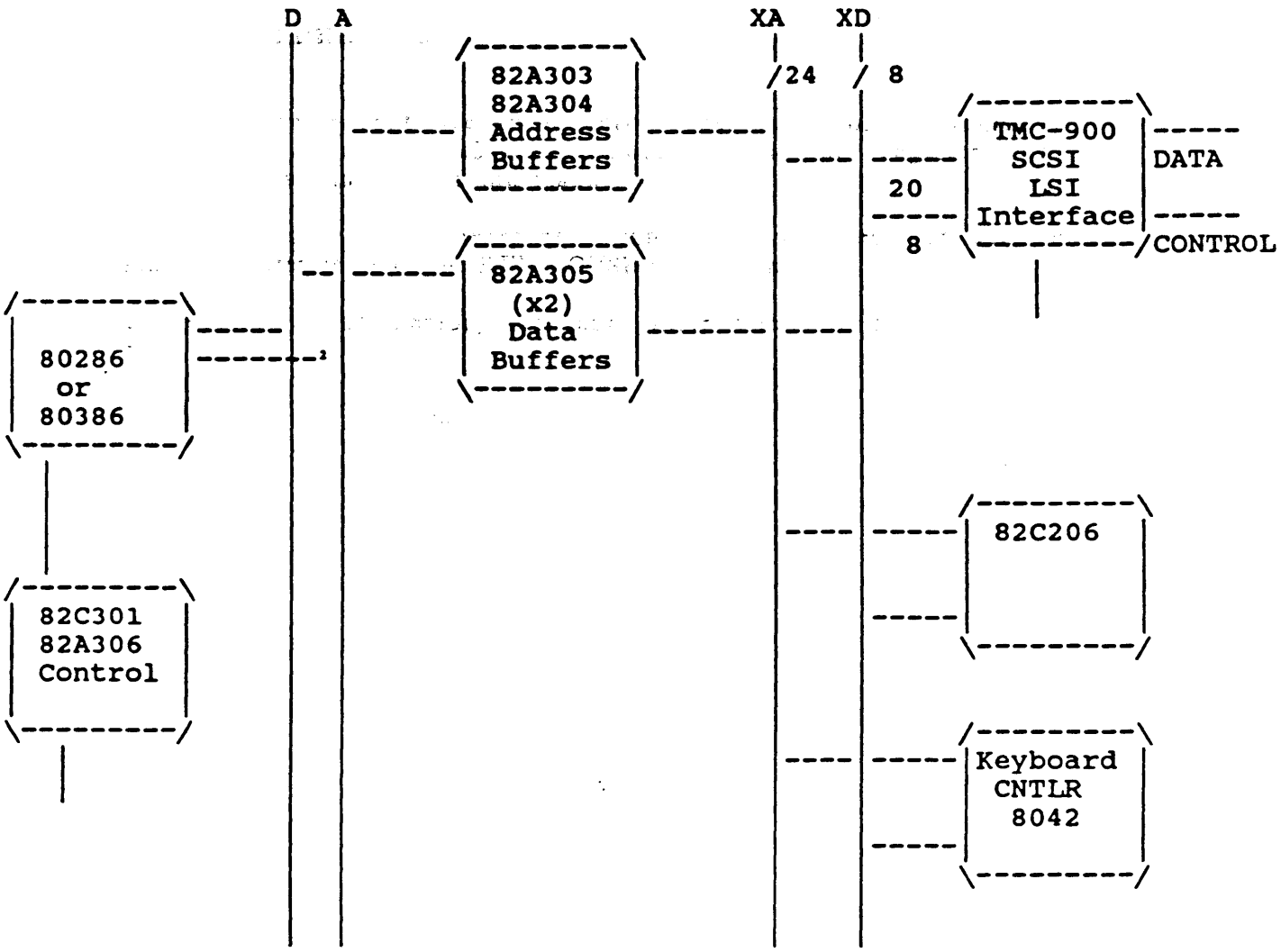



Figure 1

Figure one shows a single chip solution to a system motherboard design. The TMC-900 chip is attached directly to the XA and XD bus of the machine. Additional control line are directly connected. XMEMR, XMEMW are connected from the 82C301 (or 201) to the TMC-900. No glue chips are required.

In order to use the TMC-900 in this configuration, two rules must be followed. First, bus loading on the XD bus must not exceed 4ma. This is generally not a problem, as long as CMOS peripherals are used on this bus. If more than 4ma drive is required, an external 74LS245 buffer is required. The TMC-900 has the proper buffer control logic built right in. Figure 2

shows that type of connection.

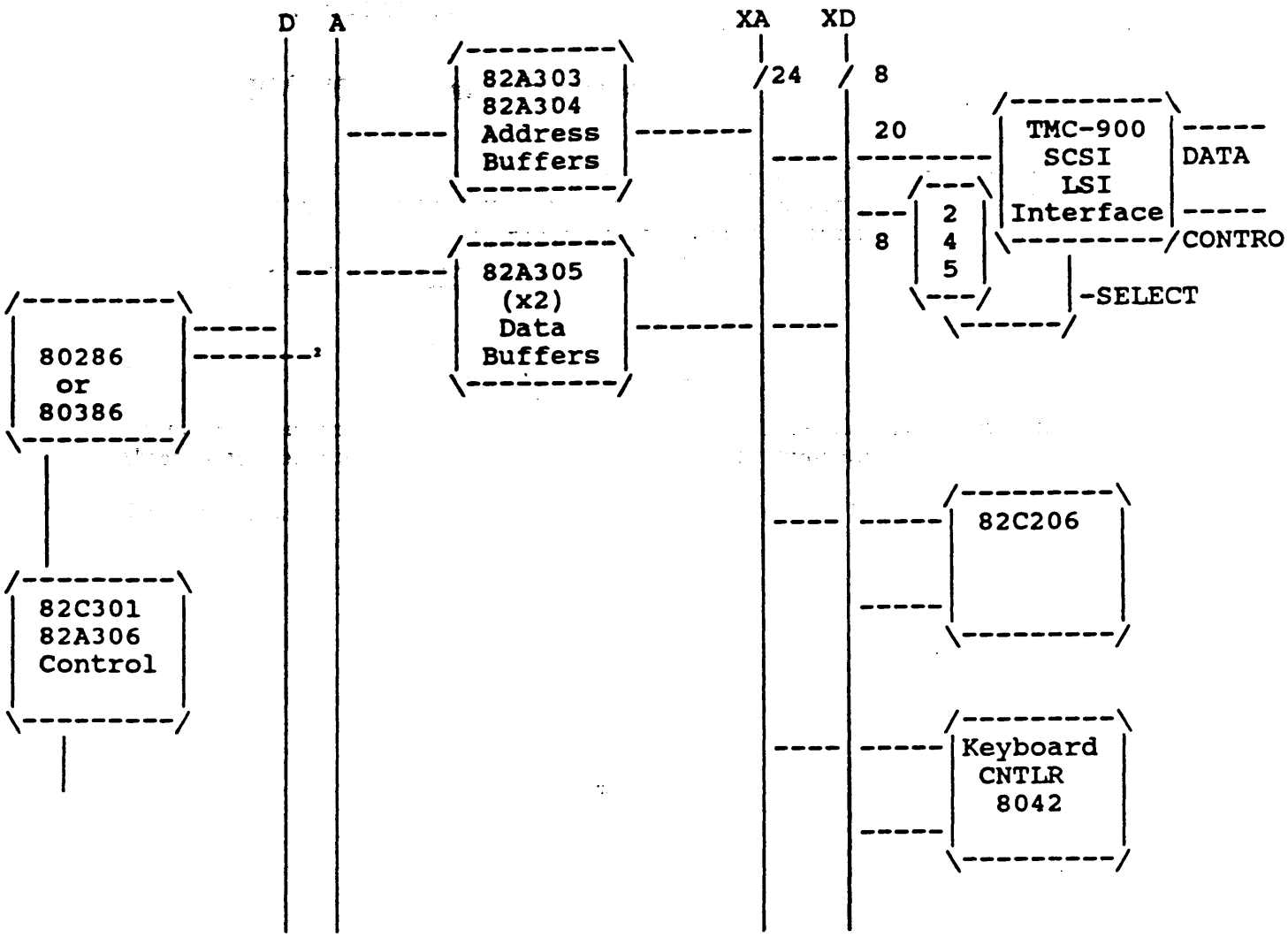



Figure 2

Second, the ROM BIOS must support the TMC-900. Future Domain has licensed a number of BIOS manufacturers to include TMC-900 support. If you have your own BIOS, you may license the proper modules from Future Domain, or write your own.

If you desire to continue to use your existing BIOS, or wish to have the flexibility of a separate SCSI BIOS, the TMC-900 support direct connection of an eight bit BIOS chip. This chip is available from Future Domain. Figure 3 shows the configuration required if you desire to use a independent ROM BIOS.

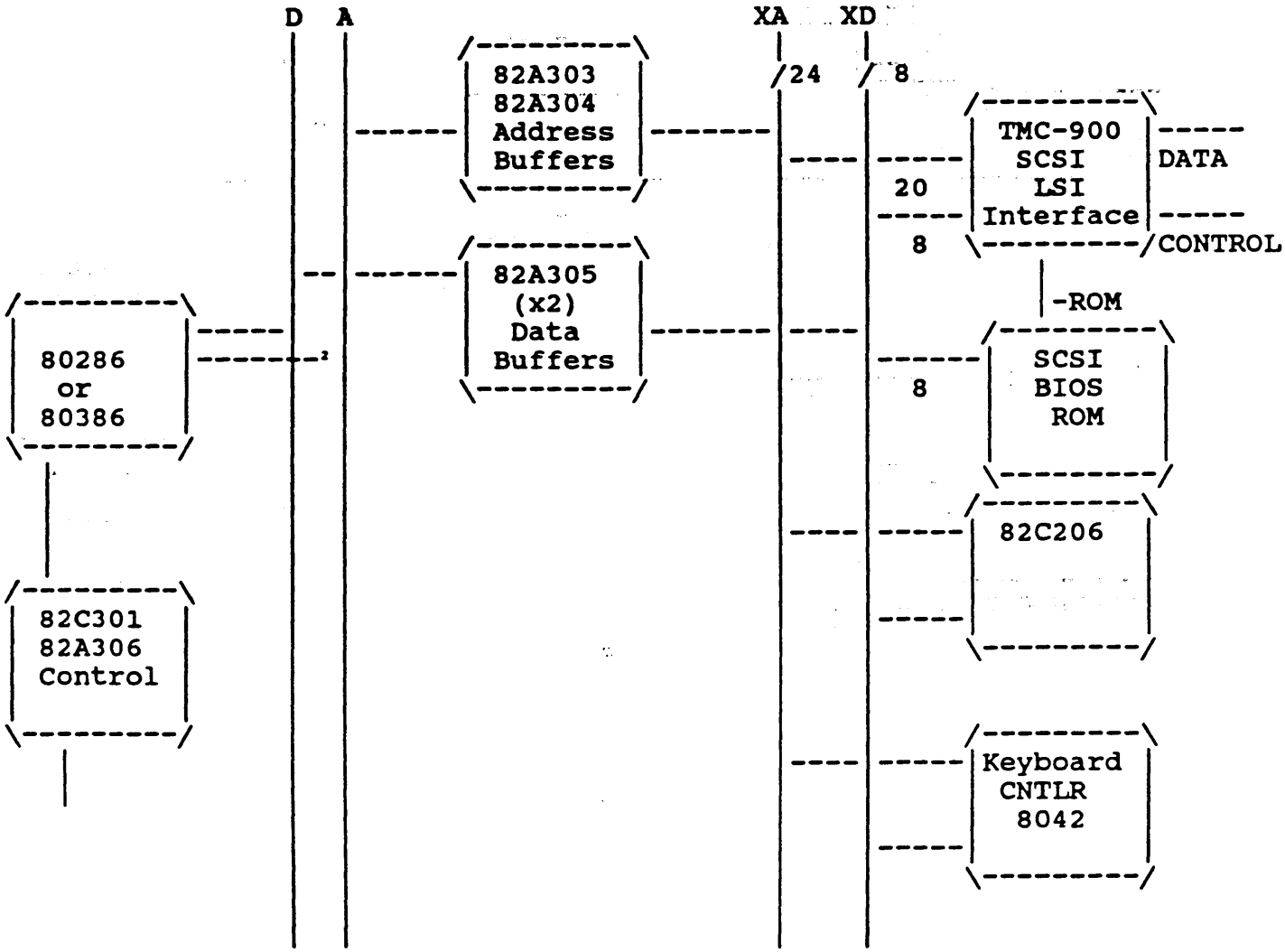
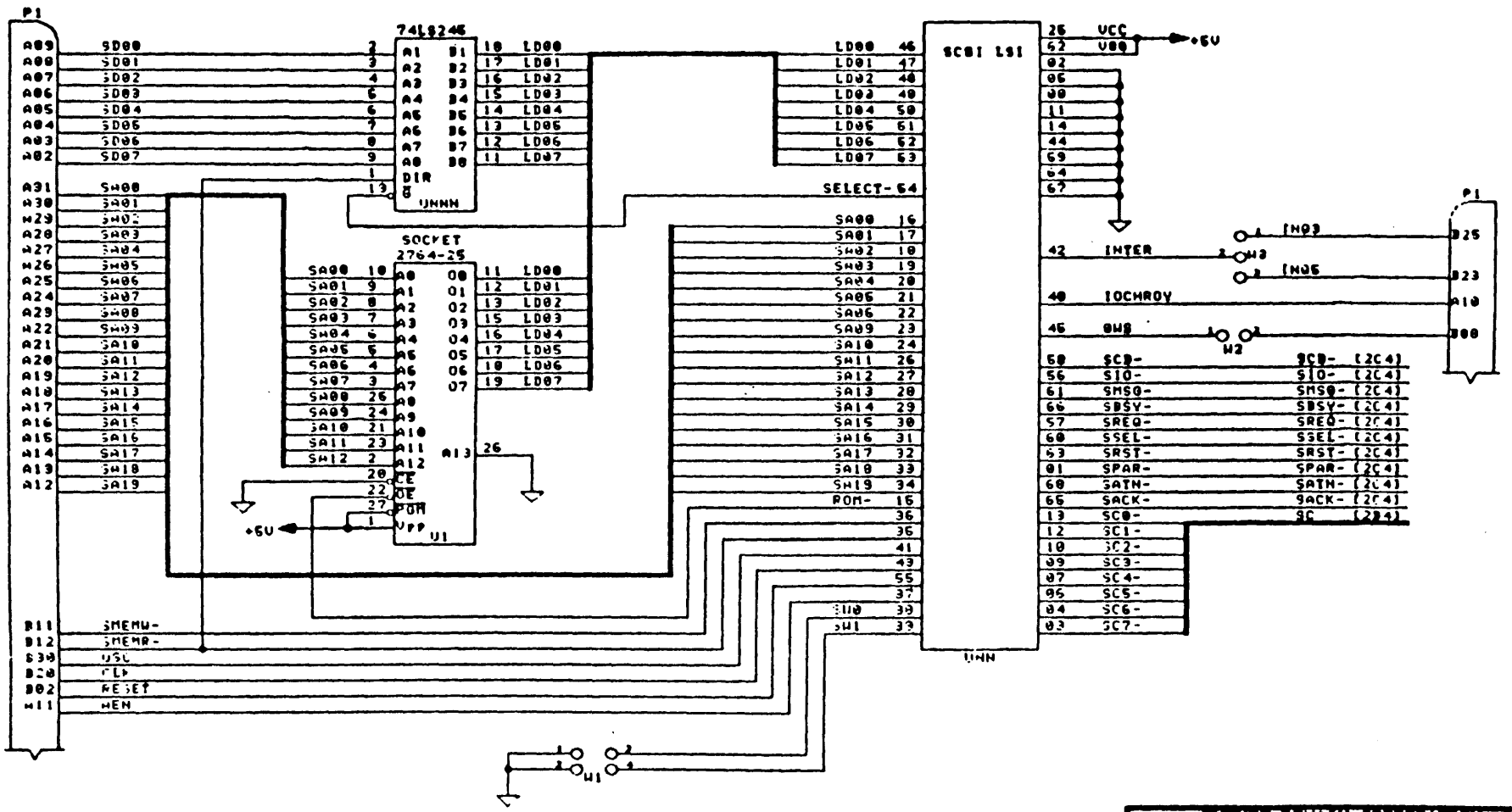


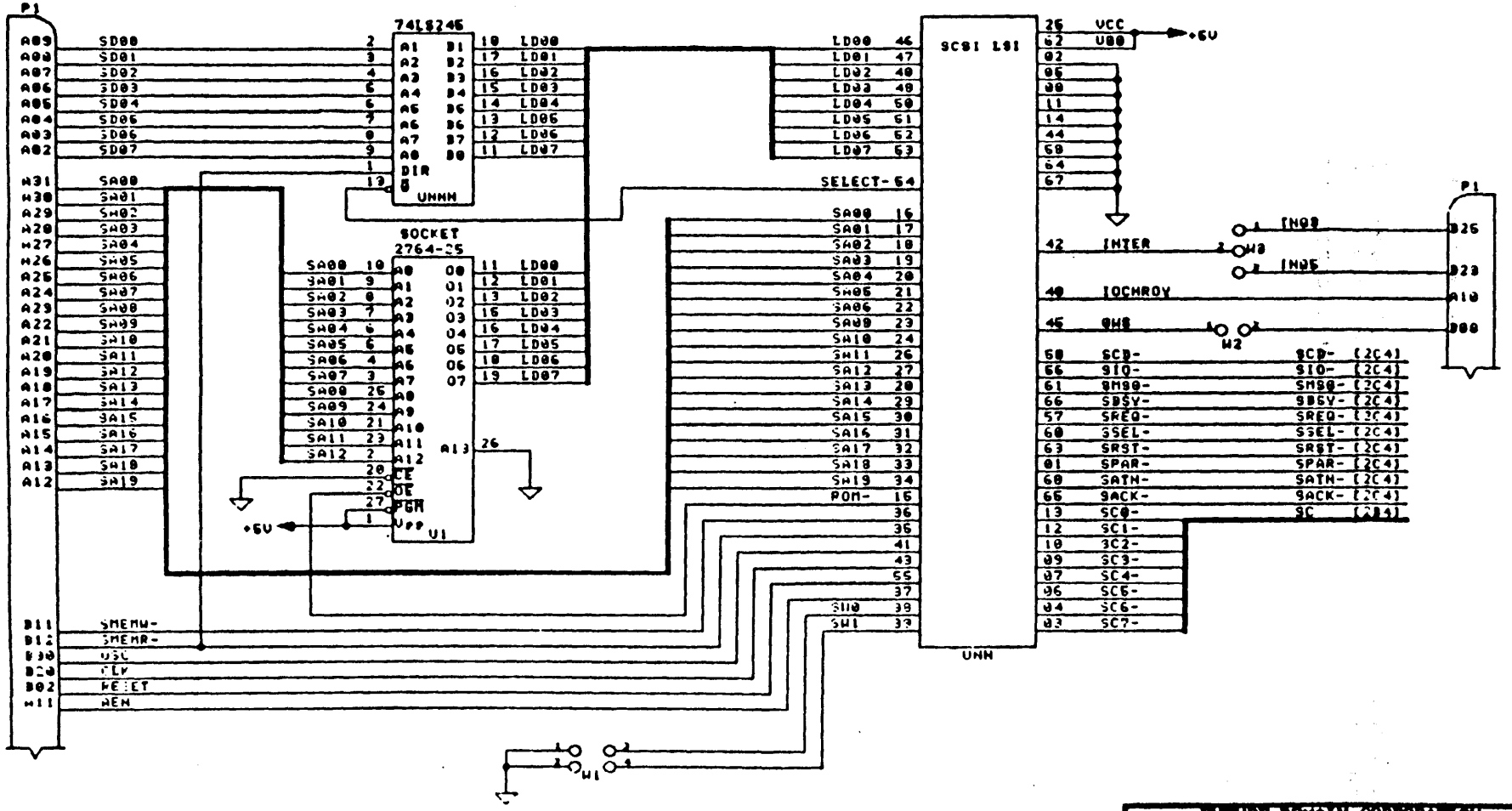

Figure 3

## 9.2 Host Bus Adapter

The following two pages show how to build a simple host bus adapter using the TMC-900. You may purchase artwork from Future Domain for a host adapter like this one.



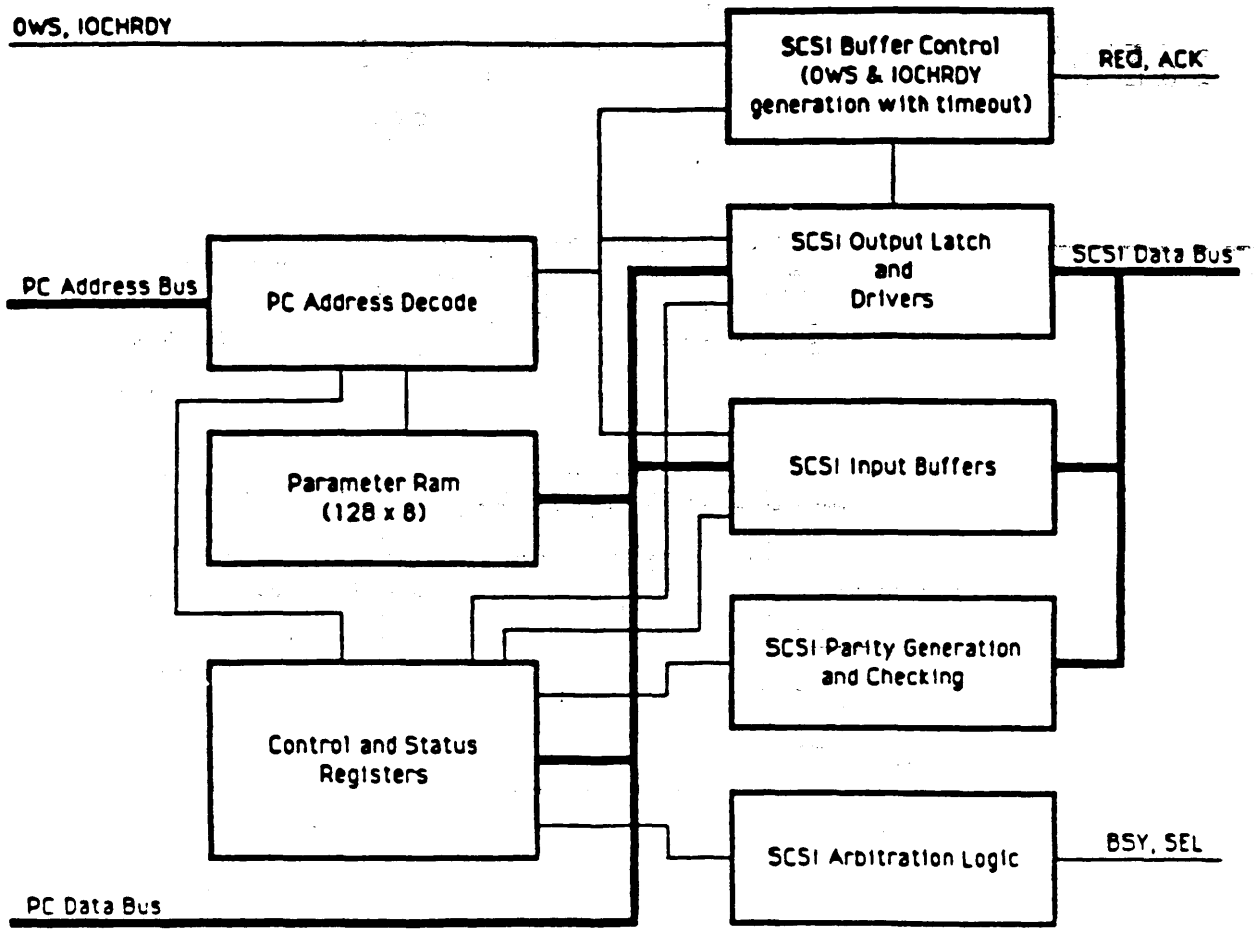

| FUTURE DOMAIN CORPORATION | |
|---|---|
| THIS DOCUMENT CONTAINS PROPRIETARY INFORMATION WHICH SHALL NOT BE REPRODUCED OR TRANSFERRED TO OTHER DOCUMENTS OR DISCLOSED TO OTHERS OR USED FOR MANUFACTURING OR ANY OTHER PURPOSE WITHOUT WRITTEN PERMISSION OF FUTURE DOMAIN CORPORATION | |
| TITLE: TMC-8301 LSI SCSI | |
| DRAWN ROLAND MONTERROSA | DATE 01/29/87 |
| CHECKED | DATE |
| DRAWING No 801-110-100-00 | SHEET 1 OF 2 |




| FUTURE DOMAIN CORPORATION | |
|---|---|
| THIS DOCUMENT CONTAINS PROPRIETARY INFORMATION WHICH SHALL NOT BE REPRODUCED OR TRANSFERRED TO OTHER DOCUMENTS OR DISCLOSED TO OTHERS OR USED FOR MANUFACTURING OR ANY OTHER PURPOSE WITHOUT WRITTEN PERMISSION OF FUTURE DOMAIN CORPORATION | |
| TITLE: TMC-8301 LSI SCSI | |
| DRAWN ROLAND MONTERROSA | DATE 01/29/87 |
| CHECKED | DATE |
| DRAWING NO. B01-110-100-00 | SHEET 1 OF 2 |

# 10.0 Appendix

## 10.1 LSI Block Diagram




PC Host Adapter Block Diagram

**10.2 SCSI Header Pin Assignments**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | GND | 2 | SC0- |
| 3 | GND | 4 | SC1- |
| 5 | GND | 6 | SC2- |
| 7 | GND | 8 | SC3- |
| 9 | GND | 10 | SC4- |
| 11 | GND | 12 | SC5- |
| 13 | GND | 14 | SC6- |
| 15 | GND | 16 | SC7- |
| 17 | GND | 18 | SPAR- |
| 19 | GND | 20 | GND |
| 21 | GND | 22 | GND |
| 23 | GND | 24 | GND |
| 25 | GND | 26 | not used |
| 27 | GND | 28 | GND |
| 29 | GND | 30 | GND |
| 31 | GND | 32 | SATN- |
| 33 | GND | 34 | GND |
| 35 | GND | 36 | SBSY- |
| 37 | GND | 38 | SACK- |
| 39 | GND | 40 | SRST- |
| 41 | GND | 42 | SMSG- |
| 43 | GND | 44 | SSEL- |
| 45 | GND | 46 | SCD- |
| 47 | GND | 48 | SREQ- |
| 49 | GND | 50 | SIO- |