Index

INTRODUCTION TO THE

SYS68K/ISCSI-1

First Edition
October 1986

PART NO. 800114

N O T E

The information in this document has been carefully checked and is believed to be entirely reliable. FORCE COMPUTERS makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. FORCE COMPUTERS reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

FORCE COMPUTERS assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of FORCE COMPUTERS GmbH/Inc.

FORCE COMPUTERS does not convey to the purchaser of the product described herein any license under the patent rights of FORCE COMPUTERS GmbH/Inc. nor the rights of others.

FORCE COMPUTERS Inc.
727 University Avenue
Los Gatos, CA 95030
U.S.A.

Phone : (408) 354 34 10
Telex : 172465
FAX : (408) 395 77 18

FORCE COMPUTERS GmbH
Daimlerstrasse 9
D-8012 Ottobrunn/Munich
West Germany

Phone : (089) 600 91-0
Telex : 524190 forc-d
FAX : (089) 609 77 93

FORCE COMPUTERS FRANCE Sarl
11, rue Casteja
92100 Boulogne
France

Phone : (1) 4620 37 37
Telex : 206 304 forc-f
Fax : (1) 4621 35 19

FORCE Computers UK Ltd.
No. 1 Holly Court
3 Tring Road
Wendover
Buckinghamshire HP22 6NR
England

Phone : (0296) 625456
Telex : 838033
Fax : (0296) 624027

Table of Contents

List of Figures

1.Ø General Information

The SYS68K/ISCSI-1 board is a high performance intelligent SCSIbus controller board which fully supports the SCSI standard.

Up to four floppy drives can be controlled locally, without using the SCSIbus.

The board provides local intelligence with a 68Ø1Ø CPU, 6845Ø DMAC, SCSI Bus Controller (SCSIBC) and Floppy Disk Controller (FDC). A 128Kbyte Dual Ported RAM (DPR) is used to store data and to interface the SYS68K/ISCSI-1 board. Highest throughput is guaranteed by using a 1ØMHz 6845Ø DMAC and the NCR 5386S SCSIBC.

On a single ended SCSIbus, the board works under the powerful firmware (128Kbyte EPROM area) as initiator or as target.

The SYS68K/ISCSI-1's own SCSIbus I.D. is software controlled. Four floppy drives are fully firmware supported, and can also be accessed via the SCSIbus in the target mode.

The SYS68K/ISCSI-1 contains a VMEbus Rev.C/IEEE P1Ø14 compatible interface to communicate to the host CPUs via its 128Kbyte DPR. The access address and the Address Modifier code are jumper selectable. A Bus Interrupter Module - BIM 68153 - is installed on the board to support fully asynchronous operation with the 4 different software programmable interrupt request channels.

The firmware of the SYS68K/ISCSI-1 described in the Firmware User's Manual handles all activities to/from the SCSIbus and the floppy interface. For special applications, the source code of the firmware is optionally available.

1.1 Hardware Features of the SYS68K/ISCSI-1

- 68010 CPU for local control (10MHz)

- 68450 DMA Controller for local transfers (10MHz)

- Dual Ported 128Kbyte 0 wait state static RAM between the VMEbus and the local CPU

- SCSIbus interface built with the NCR 5386S SCSIbus controller. Programmable as an initiator or target

- Transfer rate via the SCSIbus up to 1.5Mbyte/s

- SHUGART compatible floppy interface with the WD1772 FDC. Up to 4 floppy drives can be controlled independent of the SCSIbus

- All I/O signals available on P2 connector

- 4 different interrupt request signals to the VMEbus. Each channel contains a software programmable IRQ level (1 to 7) and vector

- Local parallel interface for controlling and monitoring board functions

- VMEbus Rev.C/IEEE P1014 compatible interface A24:D16, D8

- Watchdog timer controlling correct functions of on-board hard and software

- Status and control LEDs for monitoring local activities

- High level handling firmware for communication, selftest, data caching/hashing and control

1.2 The Hardware Functions

The local CPU reacts on the commands and initialisation parameters within the 128Kbyte DPR. Constant program run times are guaranteed through the special hardware logic providing zero wait state operation from the DPR, independent of the accesses from the VMEbus to the DPR.

The SYS68K/ISCSI-1 consists of self-test functions as well as a hardware watchdog timer which controls the activities of the 68010 CPU running with 10MHz.

User supplied programs can be loaded into the DPR and executed by the local CPU to adapt and extend board functionality.

The local CPU controls the SCSIbus and the floppy disk interface via local interrupts, and communicates to the host CPU via the DPR or via interrupt request to the VMEbus, generated by a Bus Interrupter Module.

The I/O signals to be supported through the NCR 5386S are terminated on the PC board. The terminators can be removed in order to be able to connect more than 3 SCSI devices.

1.2.1 The Local 68010 CPU

A 10MHz 68010 CPU is installed on the SYS68K/ISCSI-1 to control the data traffic between the serial I/O channels and the VMEbus host CPU(s).

Two EPROMs with a maximum capacity of 128Kbyte are installed on the SYS68K/ISCSI-1 to hold the handling firmware. Constant zero wait state operation from the EPROM guarantees maximum CPU throughput and a fixed program run time.

The 128Kbyte Dual Ported RAM is also accessible without the insertion of wait states by using a CPU clock synchronized arbitration mechanism. The accesses from the CPU to the DPR are not delayed if a VMEbus access is pending or being executed.

A local timer, included in the PI/T, is used to interrupt the CPU for task scheduling, command interpretation and execution.

The CPU and all I/O devices can be RESET through a SYSTEM reset via the SYSRESET signal of the VMEbus or by accessing a dedicated location within the DPR reserved for this function.

1.2.2 The SCSIbus Interface

The SYS68K/ISCSI-1 contains an NCR 5386S SCSIbus controller and an NCR 8310 SCSIbus driver/transceiver. These chips provide the following features:

- Support of the ANSI X3T9.2 SCSI standard
- Asynchronous data transfer to 1.5Mbyte per second
- Support of both initiator and target modes
- Parity generation with optional checking
- Support of arbitration
- Control of all bus signals except RESET
- Doubly-buffered data register
- Versatile MPU bus interface
- Memory or I/O mapped MPU interface
- DMA or programmed I/O transfer
- 24-bit internal transfer counter
- Programmable (re)selection timeouts
- Interrupt of MPU on all bus conditions requiring service
- SCSI pass parity optional with checking
- 48mA driver

The NCR 5386S SCSI Bus Controller communicates with the 68010 CPU and the 68450 DMAC as a peripheral device. The SCSI Bus Controller is controlled by reading and writing the internal registers which are addressed via the local address bus.

Since the SCSI Bus Controller interrupts the MPU when it detects an SCSIbus condition that requires servicing, the MPU is free from polling or controlling any of the SCSIbus signals.

For high speed data transfer, the SCSI Bus Controller communicates directly with the Dual Ported RAM via the DMA Controller. In this mode, the data transfer rate will be a maximum of 1.5Mbyte/s. The minimum guaranteed constant data throughput of the ISCSI-1 board is 1.3Mbyte/s, assuming an equivalent SCSI device data transfer rate.

1.2.3 The Floppy Disk Interface

To provide easy connection from the ISCSI-1 board to floppy drives, the ISCSI-1 board includes a floppy disk interface.

The FDC WD1772 is able to control up to four floppy drives (3", 3 1/2" and 5 1/4").

To allow the connection of all SHUGART-compatible floppy disk drives, the Drive Select signals Ø to 3, the Side Select signal and the single/double density signal are software programmable.

1.2.4 The PI/T 68230

A 68230 Parallel Interface and Timer Chip is installed on the SYS68K/ISCSI-1 to control and display the status of all on-board activities. The PI/T is also used to force and monitor the interrupt request lines to the Bus Interrupter Module, which initiates the interrupts to the VMEbus (under control of the host CPU).

One handshake pin is used to interrupt the local CPU if the host CPU accesses a defined location within the DPR. One output signal is used to force the SYSFAIL signal of the VMEbus if an onboard error has been detected or if the board initializes the DPR after RESET or Power up.

The timer, also included in the PI/T, is the time base for the onboard handling firmware and the scheduler for the macro commands.

A watchdog timer, for processor control, is installed on the board to detect software or hardware errors independent from the onboard CPU. For this purpose, one output of the PI/T is used to retrigger the watchdog timer within defined time frames.

If the onboard CPU does not work properly, or if the hardware isn't working correctly, the timer will not be retriggered, and the SYSFAIL signal of the VMEbus will be activated. The host CPU then can initiate a software controlled RESET for the ISCSI, or start other maintenance activities.

The SCSIbus RESET is controlled by the PI/T. One input of the PI/T indicates the state of the SCSIbus RESET, and one output controls the SCSIbus RESET signal.

The FDC 1772 single/double density selection is made via one PI/T output.

1.2.5 The Dual Ported RAM

128Kbyte of Dual Ported Static RAM with 45ns access time are installed on the SYS68K/ISCSI-1 to service all applications requiring fast operations and large amounts of data areas.

The local 68Ø1Ø CPU runs without the insertion of wait states out of the DPR, because a CPU clock synchronised arbitration logic and a full buffered and latched VMEbus interface is installed on the SYS68K/ISCSI-1. Between two CPU access cycles, a VMEbus cycle is serviced and completed. On VMEbus Read cycles, the data pattern is latched, and the internal cycle of the DPR is aborted while the VMEbus cycle is decoupled.

A partition of the DPR is reserved for the local CPU for vector storage, the program counter, and temporary buffers. This partition is used from the VMEbus side for programming the BIM and initiating an interrupt, which will be handled from the onboard CPU, or driving a local RESET.

The access address and the Address Modifier code(s) are jumper selectable in 128Kbyte increments within the standard address range (A24:D16,D8). The access times of the DPR depend on the accesses made by the local CPU while the local 68Ø1Ø has priority over VMEbus accesses.

1.2.6 The VMEbus Interface

A full VMEbus Rev.C/IEEE P1014 compatible interface is installed on the SYS68K/ISCSI-1 to allow an access to the DPR and the Bus Interrupter Module.

The 16-bit data width (D16,D8) of the DPR and the decoding of the standard address range (A24) allows easy installation in all VMEbus environments.

During Power-up and after a RESET has been executed from the local CPU, the SYS68K/ISCSI-1 drives the VMEbus signal SYSFAIL active to signal each board in the VMEbus environment that the board is not ready or has detected a malfunction.

A RESET for the local CPU can be initiated by accessing a dedicated address within the 128Kbyte boundary of the DPR. All local devices as well as the CPU will be reset through this access.

An interrupt to the local CPU can be forced by accessing another location within the DPR, signalling the on-board processor that a command has been given, or that an exception has to be taken.

The Dual Ported RAM can be accessed at least every 640ns because this is the worst case cycle time. The data transfer rate to/from the SYS68K/ISCSI-1 is 3 to 4Mbyte/s including the VMEbus protocol.

	Access Time	Cycle Time
best case	330	400
average	430	500
worst case	560	630

1.2.7 The Bus Interrupter Module

To allow fully asynchronous operation, the SYS68K/ISCSI-1 contains a Bus Interrupter Module - BIM 68153 - providing 4 individually programmable interrupt channels. Each channel is able to force an interrupt request to the VMEbus. For each channel, the IRQ level (1 to 7) as well as the interrupt vector is fully software programmable.

The local CPU forces the requests to the BIM and the host CPU can program the interrupt vector and the level. This allows dynamic change of the interrupt level and vector in multi-processor environments.

1.2.8 The Optional Back Panel

A back panel which can be plugged into the P2 connector of the SYS68K/ISCSI-1 board is optionally available. Included on this board is a 50-pin 2-row connector for the SCSIbus and a 34-pin 2-row connector for the floppy disk interface.

2.0 Specification of the SYS68K/ISCSI-1

Local CPU	68010 with 10MHz clock frequency
EPROM	128Kbyte maximum capacity 0 Wait State operation
Dual Ported RAM	128Kbyte capacity using static RAMs 0 Wait State operation from local CPU 330 ns best case VMEbus access time 430 ns average VMEbus access time 560 ns worst case VMEbus access time
SCSIbus Interface	NCR 53685 providing initiator, target mode asynchronous and synchronous modes asynchronous data rate up to 1.5sbyte/s
Floppy Disk Interface	SHUGART-compatible Up to 4 drives (3", 3 1/2", 5 1/4")
VMEbus Interface	Full Rev. C and IEEE P1014 compatible A24:D16,D8 mode 4 IRQs with SW programmable level (1 to7) and vector Access Address jumper selectable in 128Kbyte increments SYSFAIL* supported
Handling Firmware	in EPROM with macro commands for all I/O channels installed
Power Requirements	+5V : 5.6 A (max) (Power Backplane or +12V : 0.0 A (max) power connection -12V : 0.0 A (max) on P2 necessary)
Operating Temperature	0 to 50 Degrees C
Storage Temperature	-50 to +85 Degrees C (non-operating)
Relative Humidity	0 to 90% (non-condensing)
Dimensions	233 x 160mm 9.2" x 6.3"

3.0 Ordering Information

SYS68K/ISCSI-1 Part No. 300020	Intelligent SCSIbus Controller board including firmware and documentation.
SYS68K/ISCSI-1BPS Part No. 300021	Back panel for the SYS68K/ISCSI-1 board providing SCSIbus connector and floppy drive connector
SYS68K/ISCSI-1/UM Part No. 800114	User's Manual for the SYS68K/ISCSI-1
SYS68K/ISCSI-1/SC Part No. 800022	Source Code of the SYS68K/ISCSI-1 handling firmware, including documentation

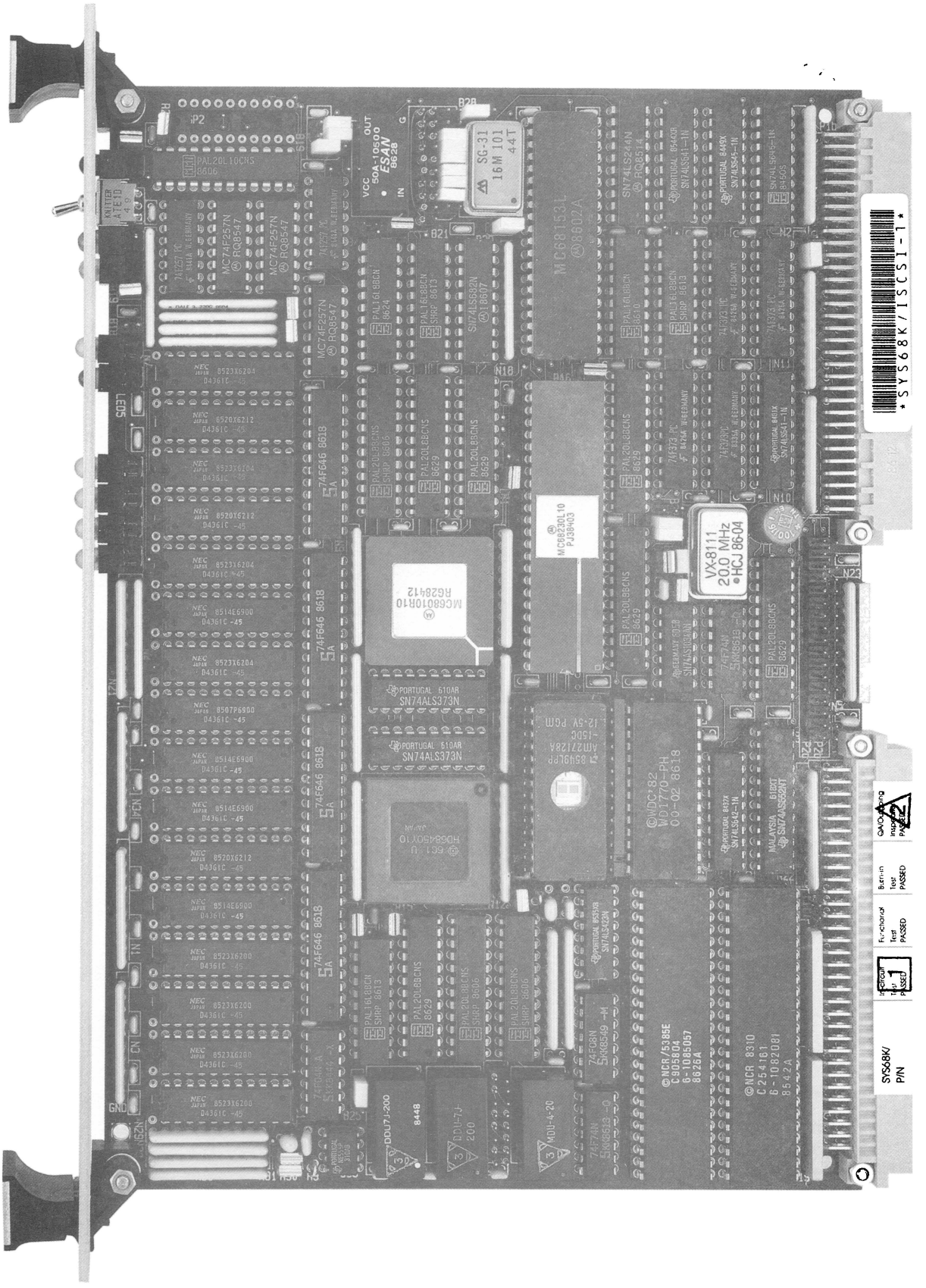
INSTALLATION

Please read the complete installation procedure before the board is installed in a VMEbus environment to avoid malfunctions and component damages.

Table of Contents

List of Tables

Figure 1-1: Photo of the SYS68K/ISCSI-1



1.Ø General Overview

Easy installation of the SYS68K/ISCSI-1 is provided as the board is shipped in a ready-to-operate default configuration. A selftest is executed on reset by the firmware on the board.

Please read the complete installation procedure before mounting the board into a VMEbus backplane.

1.1 The Function Switch Positions

There is a RUN/LOCAL (R/L) toggle switch installed on the front panel.

The two positions of the switch are defined as "UP" and "DOWN". The switch has to be set to "DOWN" for the first installation.

1.2 Connection of I/O Devices

The SCSIbus signals and the floppy disk interface signals are routed to the P2 connector of the SYS68K/ISCSI-1 board. Table 1-1 shows the pin out of the P2 connector. For a detailed description, please refer the to Hardware User's Manual, Chapter 4.8.

If the optional SYS68K/ISCSI-1BPS is used, it must be connector to the P2 connector of the SYS68K/ISCSI-1 board. The pin out of the SCSIbus connector (X1) is listed in Table 1-2 and the pin out of the floppy disk interface connector (X2) is listed in Table 1-3.

Table 1-1: The P2 Pin Assignment

Pin Number	Row A Signal Mnemonic	Row B Signal Mnemonic	Row C Signal Mnemonic
1	DB Ø	VCC	N.C.
2	DB 1	GND	N.C.
3	DB 2	N.C.	Drive Select Ø
4	DB 3	N.C.	Index
5	DB 4	N.C.	Drive Select 1
6	DB 5	N.C.	Drive Select 2
7	DB 6	N.C.	Drive Select 3
8	DB 7	N.C.	Motor On
9	DB P	N.C.	Direction In
1Ø	GND	N.C.	Step
11	GND	N.C.	Write Data
12	GND	GND	Write Gate
13	TERMPWR	VCC	Track ØØØ
14	GND	N.C.	Write Protect
15	GND	N.C.	Read Data
16	ATN	N.C.	Side Select
17	GND	N.C.	N.C.
18	BSY	N.C.	N.C.
19	ACK	N.C.	GND
2Ø	RST	N.C.	GND
21	MSG	N.C.	N.C.
22	SEL	GND	GND
23	C/D	N.C.	GND
24	REQ	N.C.	N.C.
25	I/O	N.C.	N.C.
26	N.C.	N.C.	N.C.
27	GND	N.C.	RESERVED
28	N.C.	N.C.	RESERVED
29	RESERVED	N.C.	RESERVED
3Ø	RESERVED	N.C.	RESERVED
31	RESERVED	GND	RESERVED
32	RESERVED	VCC	RESERVED

Table 1-2: The X1 Pin Assignment

Pin No.	Signal Mnemonic	Pin Number	Signal Mnemonic
2	DB Ø	1	GND
4	DB 1	3	GND
6	DB 2	5	GND
8	DB 3	7	GND
10	DB 4	9	GND
12	DB 5	11	GND
14	DB 6	13	GND
16	DB 7	15	GND
18	DB P	17	GND
20	GND	19	GND
22	GND	21	GND
24	GND	23	GND
26	TERMPWR	25	N.C.
28	GND	27	GND
30	GND	29	GND
32	ATN	31	GND
34	GND	33	GND
36	BSY	35	GND
38	ACK	37	GND
40	RST	39	GND
42	MSG	41	GND
44	SEL	43	GND
46	C/D	45	GND
48	REQ	47	GND
50	I/O	49	GND

Table 1-3: The X2 Pin Assignment

Pin No.	Signal Mnemonic	Pin Number	Signal Mnemonic
2	N.C.	1	GND
4	N.C.	3	GND
6	Drive Select Ø	5	GND
8	Index	7	GND
10	Drive Select 1	9	GND
12	Drive Select 2	11	GND
14	Drive Select 3	13	GND
16	Motor On	15	GND
18	Direction In	17	GND
20	Step	19	GND
22	Write Data	21	GND
24	Write Gate	23	GND
26	Track ØØØ	25	GND
28	Write Protect	27	GND
30	Read Data	29	GND
32	Side Select	31	GND
34	N.C.	33	GND

1.3 Base Address Selection and AM Decoding

The default setup of the SYS68K/ISCSI-1 board is as follows:

Base Address	$A00000
End Address	$A1FFFF

Address Modifiers are set to accept privileged and non-privileged standard data accesses (A24:D16,D8)

If a different setup is required, please refer to the Hardware User's Manual.

1.4 Interrupts

All interrupt levels and vectors are software programmable on the SYS68K/ISCSI-1 so no hardware setup is required.

2.Ø Installation in the Rack

The board is configured to be mounted into a VMEbus rack at any one of the slots 2-21.

A reset generator has to be included in another slot.

Caution:

A) The VMEbus rack has to include a J1 and a J2 backplane.

B) Switch power off before installing the board to avoid electrical damages to the components.

C) The board has the be plugged in and the screws of the front panel must be turned on to guarantee proper installation.

D) No connections are allowed on the P2 backplane on rows A and C (i.e. VMXbus or VSBbus connection).

2.1 Power On

If the board is installed correctly, the following sequence will take place:

1) Green RUN LED lights up.

2) During SYSRESET, HALT LED is red.

3) After SYSRESET goes inactive, the HALT LED turns green and the red SYSFAIL LED turns on. At this time the SYSFAIL signal on the VMEbus is also activated.

4) The onboard firmware starts execution of the onboard selftest, which takes about 3Ø seconds. The status information that is given on the front panel LEDs S1-S4 is described in Chapter 2.2.

5) When the selftest is complete and if no errors have occurred, the SYSFAIL LED turns off and the SYS68K/ISCSI-1 is ready for operation.

If any errors are detected during selftest, the LED S4 turns on and the LEDs S1-S3 show an error code. Please refer to the Software User's Manual for details.

2.2 The SYS68K/ISCSI-1 On-Board Selftest

On power-up or reset, the SYS68K/ISCSI-1 selftest routine is executed in the following manner:

- Test of SCSI controller by write and read back several registers.
- Start SCSI controller self-diagnostic.
- Test the whole local and dual ported memory with read and write bytes, words and long words.
- DMA Controller test with high speed data transfer memory to memory.
- Test the floppy disk controller.

The control of the selftest state and results is provided via the front panel LEDs S1 to S4:

- LED S1 is turned on during the RAM test.
- LED S2 is turned on during the SCSIbus controller test.
- LED S3 is turned on during the floppy disk controller test.
- LED S4 is turned on during the DMA Controller test.

If any error has been found while selftest was active, the LED of the test phase which has generated the error will stay on. After the selftest has been successfully completed, all LEDs are turned off.

Example: After completion of the selftest routine, the LED S3 remains on. This state indicates a hardware error on the WD1772 Floppy Disk Controller.

HARDWARE USER'S MANUAL

Table of Contents

Table of Contents cont'd

List of Figures

List of Tables

1.Ø General Information

The SYS68K/ISCSI-1 is a high performance intelligent SCSIbus controller board which fully supports the SCSI standard.

Up to 4 floppy drives can be controlled locally, without using the SCSIbus.

The board provides local intelligence with a 68Ø1Ø CPU, 6845Ø DMAC, SCSIbus Controller (SCSIBC) and Floppy Disk Controller (FDC). A 128Kbyte Dual Ported RAM (DPR) is used to store data, and to interface the SYS68K/ISCSI-1 board. Highest throughput is guaranteed by using a 1ØMHz 6845Ø DMAC and the NCR 5386S SCSIBC.

On a single ended SCSIbus, the board works under the powerful firmware (128Kbyte EPROM area) as initiator or as target.

The SYS68K/SCSI-1's own SCSIbus I.D. is software controlled.

Four floppy drives are fully firmware supported and can also be accessed via the SCSIbus in the target mode.

The SYS68K/ISCSI-1 contains a VMEbus Rev.C/IEEE P1Ø14 compatible interface to communicate to the host CPUs via its 128Kbyte DPR. The access address and the Address Modifier code are jumper selectable. A Bus Interrupter Module (BIM 68153) is installed on the board to support fully asynchronous operation with the 4 different software programmable interrupt request channels.

The firmware of the SYS68K/ISCSI-1 described in the Firmware User's Manual handles all activities to/from the SCSIbus and the floppy interface. For special applications, the source code of the firmware is optionally available.

Figure 1-1: Photo of the SYS68K/ISCSI-1

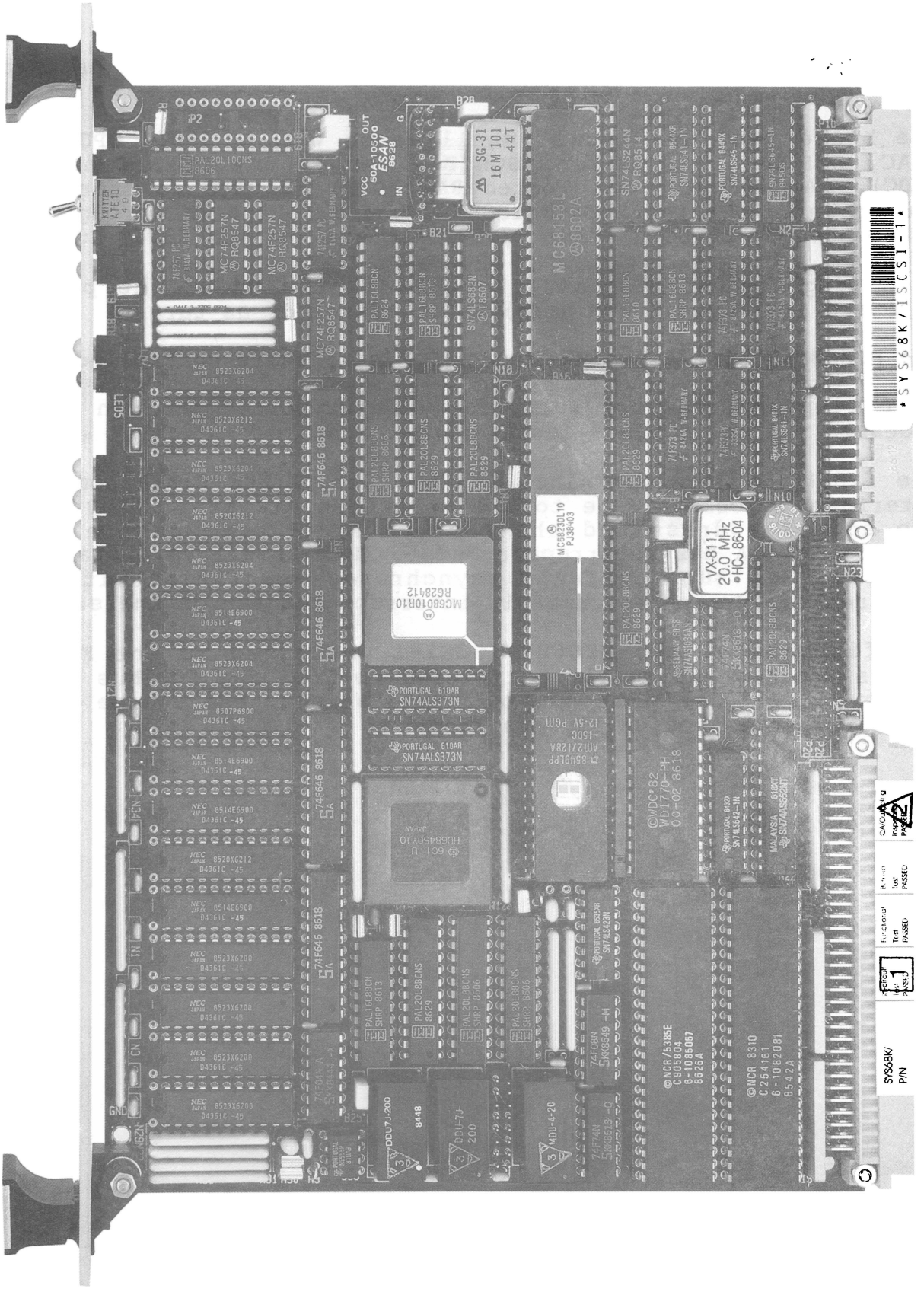
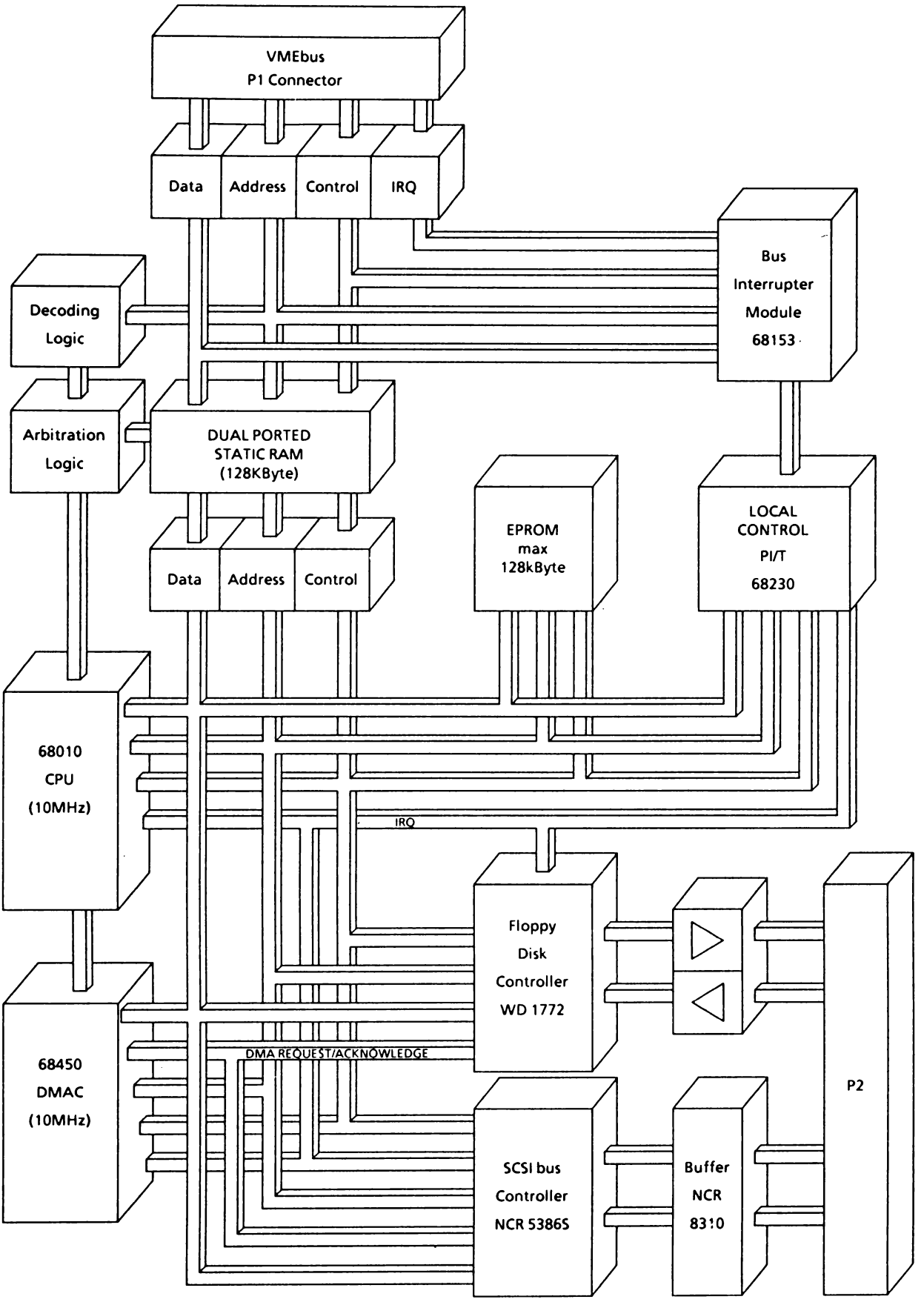


Figure 2-1: Block Diagram of the SYS68K/ISCSI-1



2.0 General Operation

The SYS68K/ISCSI-1 is an intelligent VMEbus controller board providing an SCSIbus interface and a floppy disk interface.

The SYS68K/ISCSI-1 board includes 48mA drivers and receivers for the on-board terminated single ended SCSIbus and SHUGART compatible floppy interface. The buffered I/O signals are routed to the P2 backplane connector.

The NCR 5386S SCSIbus Controller supports the asynchronous and synchronous data transfer via the SCSIbus.

The WD 1772 floppy disk controller/formatter provides control of a maximum of four floppy disk drives (3", 3 1/2", 5 1/4").

The 68010 CPU chip, running at 10MHz, executes local software performing the intelligent I/O functions. The local I/O devices can be directly accessed by the CPU. A 68230 PI/T is installed for local control and timer functions.

The four channel 68450 Direct Memory Access Controller (DMAC) controls the high speed I/O data transfer.

The on-board SRAM is a Dual Ported RAM, which is accessible for the local CPU and DMAC and for the VMEbus master. The Dual Ported RAM (DPR) has 128Kbyte of wait-cycle-free memory space for the local CPU, as well as the medium of interface to the VMEbus master.

The SYS68K/ISCSI-1 board includes the function of an interrupter to the VMEbus. Up to four interrupts can be operated independent from each other.

The SYSFAIL* signal of the VMEbus is supported, and status readout and a restart call are included in the hardware.

The general block diagram of the SYS68K/ISCSI-1 is shown in Figure 2-1.

3.0 Functional Groups of the SYS68K/ISCSI-1

The SYS68K/ISCSI-1 consists of the following functional groups:

- The 68010 CPU with ROM
- The 68454 DMAC
- Local control and decoding
- The SCSIbus interface
- The floppy disk interface
- VMEbus interface and decoding
- Dual Ported RAM (DPR)
- Bus Interrupter Module with status register
- Reset and interrupt trigger call
- The watchdog timer
- Interrupt generator

The detailed description has been organised in two chapters: the local CPU hardware is described in Chapter 4 and the VMEbus interface is described in Chapter 5.

The description of the front panel and the P2 connector is as follows:

3.1 The Front Panel

The SYS68K/ISCSI-1 is delivered with a front panel which also includes lever handles.

There is a RUN/LOCAL switch with LED indicators in the top position. When the switch is down, the board can be accessed on the VMEbus and the green RUN LED indicates this state. When the RUN/LOCAL switch is up, the board cannot be accessed from the VMEbus. In this situation the red LED, labelled LOCAL, lights up. The RUN and LOCAL LEDs can only light up one at a time.

The bi-colour LED with the label HALT will show a green light during the normal operation of the local CPU. It turns to red when the local CPU is in the HALT state and also during the RESET of the board.

The yellow LED, labelled SEL for select, lights up each time the SYS68K/ISCSI-1 board is accessed from the VMEbus during data cycles and interrupt acknowledge cycles of the VMEbus.

The yellow FAIL indicator LED reflects the state of the on-board watch-dog timer and blinks during the high-speed DMAC SCSIbus transfers. If the FAIL LED lights up for more that 2 seconds, this means that the watch-dog timer has run out of time.

The yellow status LEDs labelled, S1 - S4, are controlled by the local PI/T device and are fully software controlled. These LEDs are all turned off at the RESET of the board.

Figure 3-1 shows the front panel of the SYS68K/ISCSI-1 with the switches and LEDs.

3.2 The I/O Connector

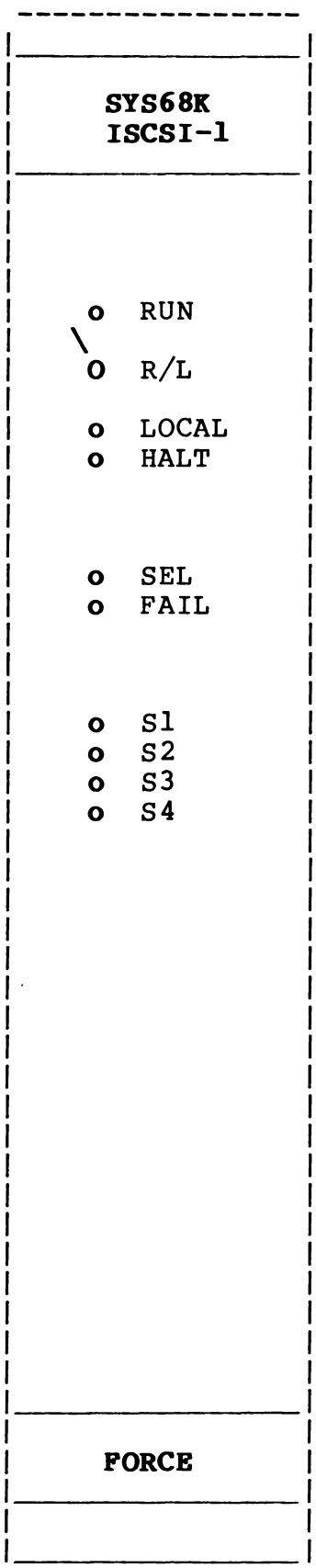
The VMEbus definition of the Double Eurocard form factor includes two 96 pin bus connectors.

The P1 connector is the primary VMEbus interface. The P2 connector carries the extension bus signals and power supply connections via the b row. Rows a and c are for user I/O signals.

The SYS68K/ISCSI-1 board supplies buffered SCSI and floppy disk interface via the P2 connector. There are 25 pins providing the floppy disk I/O signals. 10 pins of the P2 connector are reserved for later applications.

The detailed pin-out is shown in section 4.8.

Figure 3-1 The Front Panel of the SYS68K/ISCSI-1



4.0 The Local CPU Hardware

There is a 68010 microprocessor clocked at 10MHz working on the SYS68K/ISCSI-1.

The local CPU chip has an address and data bus totally independent of the VMEbus.

The local structure includes:

- Address decoding
- Reset driving
- Boot-up support
- Interrupt logic
- Bus error time-out control
- The DMAC (68450)
- The EPROMs
- The PI/T (68230)
- The SCSIbus Controller (5386S)
- The FDC (1772)
- The read back register for floppy disk control
- Access to the Dual Ported RAM

4.1 Implementation of the Local Processor 68010

The local structure is built around the microprocessor 68010 which is fast enough to control the SCSIbus, the floppy disk interface and to handle data conversion and transport commands.

The 68010 works at a clock rate of 10MHz, therefore no wait cycles are needed if the Dual Ported RAM or the EPROM are accessed by the processor. This gives maximum yield of the processors power.

The following subchapters describe:

- Reset and bootup
- Decoding
- Interrupt structure
- Bus error

4.1.1 Reset and Bootup of the Local CPU

The local CPU hardware can be reset in two different ways.

If the SYSRESET* signal of the VMEbus is asserted, then the local CPU as well as the local peripheral devices and the VMEbus interface will be reset.

If a byte Read access is performed to the reset trigger address, then the local CPU and the local peripheral devices will be reset. The reset trigger address is $18Ø1 + board base address. Default is: AØ18Ø1.

For boot-up, upon reset (or restart), the decoding logic is changed, so that the local EPROMs will be selected instead of the RAM. The first Write cycle of the local CPU introduces normal address decoding.

4.1.2 Decoding of the Local Address Space

The local decoding structure is fixed. The address map is as follows:

$ØØØØØØ	-	$Ø1FFFF	read/write	Dual Ported RAM
$C4ØØØØ	-	$C4Ø01F	byte only	SCSIbus controller
$C8ØØØØ	-	$C8ØØFF	read/write	DMAC
$CCØØØØ	-	$CCØØØ7	odd byte only	FDC
$CCØØØ9	-	$CCØØØ9	odd byte only	Control register
$DØØØØØ	-	$DØØØ3F	odd byte only	PIT
$EØØØØØ	-	$E7FFFF	read only	EPROMs

4.1.3 The Local Interrupt Structure

The local interrupt structure is organized as follows:

IRQ Level	IRQ Source	Vector Source if B41 inserted	Vector Source if B41 removed
1	P3 Pin #13	AV1 Autovector	AV1 Autovector
2	DMAC	DMAC	AV2 Autovector
3	SCSIBC	AV3 Autovector	AV3 Autovector
4	FDC	AV4 Autovector	AV4 Autovector
5	PI/T Timer	PI/T Timer Vector Reg.	PI/T Timer Vector Reg.
6	--	--	--
7	PI/T Port	PI/T Port Vector Reg.	PI/T Port Vector Reg.

Default configuration: B41 jumper removed.

The PI/T port interrupt can be used under software control to cause non-maskable (Level 7) interrupts if the watchdog timer elapses and/or if the VMEbus interrupt trigger call occurs.

4.1.4 The Local Bus Error Structure

There is a time-out counter installed to provide for bus error generation in the case of accessing non-decoded memory locations.

4.2 The EPROMs

The SYS68K/ISCSI-1 is delivered together with firmware, stored in two 27128 EPROMs. These EPROMs must have a maximum access time of 150ns. All EPROM access cycles are without wait states.

Other EPROM sizes are configurable by changes in the jumpering at B19.

The two EPROMs are stacked above each other, the lower byte (bits 0-7) in the lower socket, the upper byte (bits 8-15) in the upper socket.

The base address of the EPROMs is $E00000, and only read accesses are allowed.

The first two long locations in the EPROM pair must contain the boot-up information as follows:

E00 000 : Initial Supervisor Stack Pointer
E00 004 : Initial Program Counter

Configuration of the B19 jumper area:

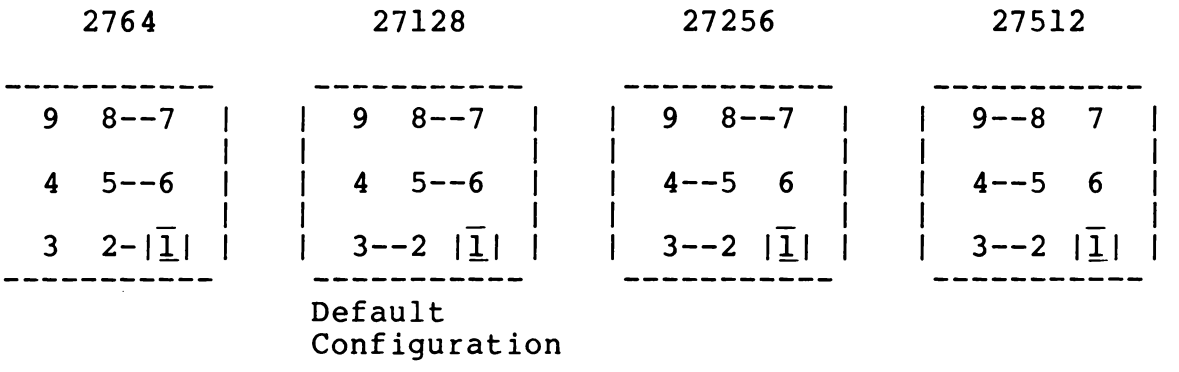
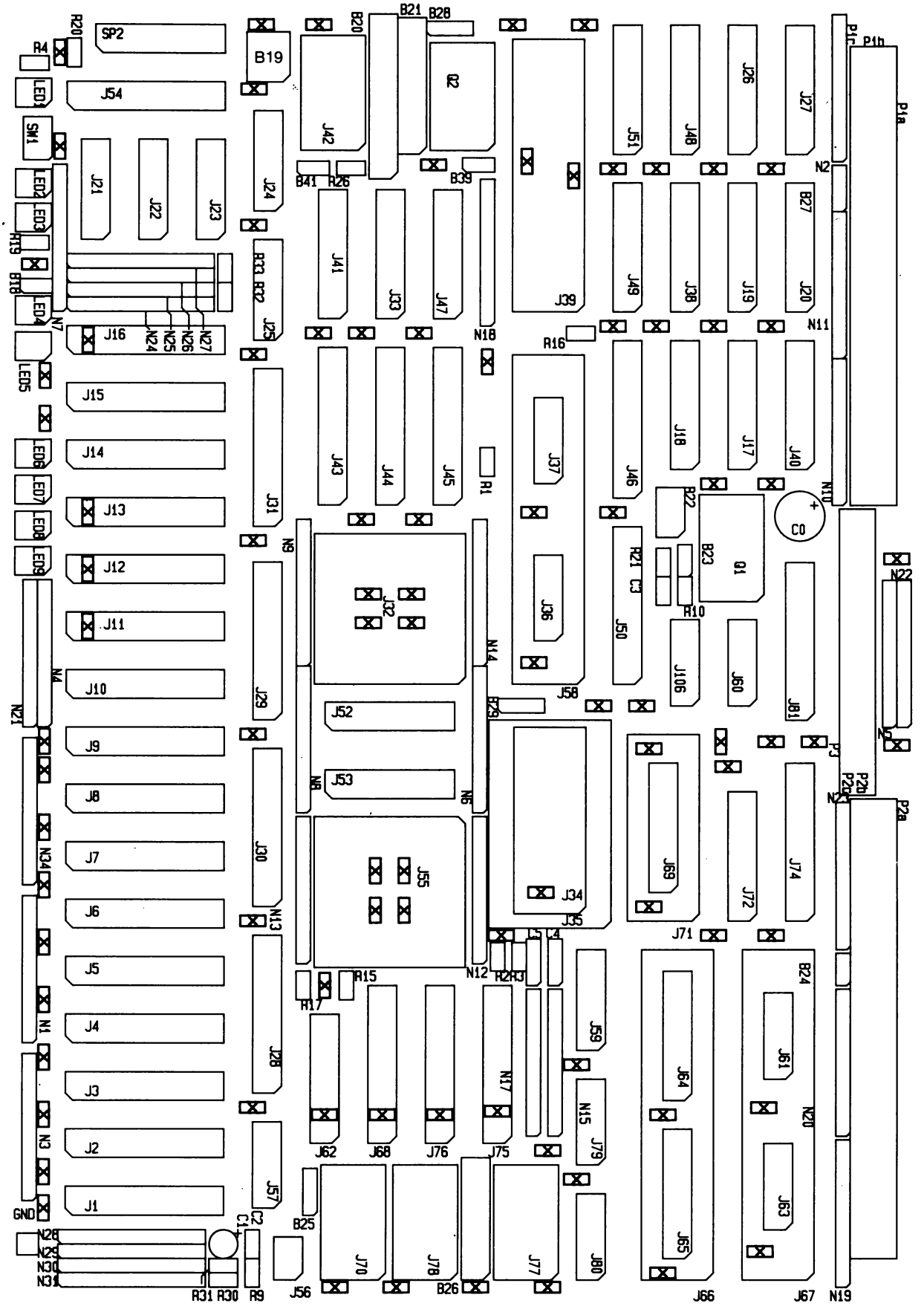


Figure 4-1: Location Diagram of the Jumperfield B19



4.3 The Parallel Interface and Timer 68230 (PI/T)

There is a PI/T device installed for local control on the SYS68K/ISCSI-1.

The PI/T can be accessed only by the local CPU. The address range is $D00001 to $D0003F - odd byte only.

Table 4-1 gives the address map of the PI/T.

The following subchapters describe

- Control of the front panel LEDs
- Control of the SYSFAIL* signal and the watchdog timer
- Control of the VMEbus interrupt requests
- The interrupt trigger input
- Reading the RUN/LOCAL switch
- Reading the B42 jumper selection
- The timer section of the PI/T
- The SCSIbus reset control
- The single or double floppy density selection

The PI/T port interrupt drives level 7 interrupts and the timer is connected to drive level 5 interrupts to the CPU. Both have to be configured (by software) to support interrupt vectors.

Table 4-1: The PI/T Address Map

Base Address : $D00000

Address HEX	Offset	Reset Value	Label	Description
D00001	01	00	PITPGCR	Port General Control Register
D00003	03	00	PITPSRR	Port Service Request Register
D00005	05	00	PITPADDR	Port A Data Direction Register
D00007	07	00	PITPBDDR	Port B Data Direction Register
D00009	09	00	PITPCDDR	Port C Data Direction Register
D0000B	0B	0F	PITPIVR	Port Interrupt Vector Register
D0000D	0D	00	PITPACR	Port A Control Register
D0000F	0F	00	PITPBCR	Port B Control Register
D00011	11	--	PITPADR	Port A Data Register
D00013	13	--	PITPBDR	Port B Data Register
D00015	15	--	PITPAAR	Port A Alternate Register
D00017	17	--	PITPBAR	Port B Alternate Register
D00019	19	--	PITPCDR	Port C Data Register
D0001B	1B	--	PITPSR	Port Status Register
D00021	21	00	PITTCR	Timer Control Register
D00023	23	0F	PITTIVR	Timer Interrupt Vector Reg.
D00025	25	--	PITCPR	Counter Preload Register
D00027	27	--		
D00029	29	--		
D0002B	2B	--		
D0002D	2D	--	PITCNTR	Count Register
D0002F	2F	--		
D00031	31	--		
D00033	33	--		
D00035	35	00	PITTSR	Timer Status Register

4.3.1 Control of the Front Panel LEDs S1-S4

There are four yellow LEDs, labelled S1 to S4, installed on the front panel of the SYS68K/ISCSI-1, that are driven by the PI/T device. These LEDs are fully under control of the local software which can write 1s and Øs to the corresponding PI/T register bits.

LED	Controlled by
S1	Port B Bit Ø
S2	Port B Bit 1
S3	Port B Bit 2
S4	Port B Bit 3

For correct operation, Port B has to be programmed to operate in the bit I/O mode, and bits Ø, 1, 2 and 3 of the Port B data direction register have to be set (1).

When writing to the Port B data register, bits Ø to 3, a zero causes the light to turn on, and a one causes it to turn off.

4.3.2 Control of the SYSFAIL* Signal and the Watchdog Timer

There is a watchdog timer installed on the SYS68K/ISCSI-1. This is a retriggerable multivibrator with a nominal pulse duration of 20 milliseconds. This device is included to give information should a system breakdown occur.

The output state of the watchdog timer is available on the H3 terminal of the PI/T. The watchdog timer trigger input is connected to the PC4 pin of the PI/T. The high to low transition is the trigger.

If the watchdog is not triggered for more than 20 milliseconds, then its output becomes low (0). Triggering at a rate which is high enough (i.e. every 10 milliseconds), means that the output of the watchdog timer can be kept high (1). In case of a system breakdown, this retrigger does not occur any more, and the watchdog timer becomes low (0).

It is possible to use the H3 terminal of the PI/T not only to read, but also to use as an interrupt input (level 7 to the CPU). This gives an option for system recovery as long as the CPU does not go to HALT.

The SYSFAIL* signal of the VMEbus can be used to broadcast the fact that a board is not functioning because of boot-up, selftest or failure. This signal will not be driven by the ISCSI-1 while SYSRESET* is active. As soon as SYSRESET* is released, SYSFAIL will be asserted (driven to low). This situation can only be changed by software controlled action of the local CPU.

Two output pins of the PI/T device control the SYSFAIL* output: PC1 and PC2. After RESET, both pins are driven high due to pull-up resistors. The same is the case with both pins driven high (1) via software control.

If PC1 is programmed to output low logic level (0), then the SYSFAIL* output to the VMEbus is released. With PC1 = low, SYSFAIL* is never active.

If PC1 is high (1), and PC2 is programmed to output low (0), then the SYSFAIL* output will be asserted if the watchdog timer output is low (0). This should be the runtime configuration. As long as the periodical toggle at the PI/T output pin PC4 occurs every 10 milliseconds, the watchdog timer output stays high (1), and SYSFAIL* will not be asserted by ISCSI-1.

There is one more option:

The watchdog timer output can drive an interrupt request to the VMEbus via channel 3 of the BIM device. If pin PC0 of the PI/T is programmed to output low (0), then the watchdog timer output is connected to set a flip-flop to drive an IRQ to channel 3 of the BIM. The flip-flop will be cleared by hardware during the interrupt acknowledge cycle. In this configuration, the PI/T output PA3 is not available for triggering this IRQ channel.

4.3.3 Control of the VMEbus Interrupt Requests

There is a Bus Interrupter Module 68153 (BIM) included in the SYS68K/ISCSI-1. This device is only accessible from the VMEbus (see chapter 6.6). So, the interrupt levels and vectors can only be programmed by VMEbus masters. The ISCSI-1 local software has no influence on this. The local software decides if and on which channel an interrupt request (IRQ) will be emitted. There are four channels to output an IRQ, while the BIM can distribute them to the seven IRQ level lines of the VMEbus.

The interrupt request signal must be released a very short time after it has been acknowledged. No software is fast enough to do that. Therefore, the IRQ outputs are handled in flip-flops that are asserted by software via a toggle from the flip-flop clock. The state of the four IRQ flip-flops can be read back by the local CPU. This is absolutely necessary, because a second IRQ can only be started to the same channel after the first has been acknowledged.

This affords some dynamic software handling by the local CPU. An IRQ asserted can only be cleared by the interrupt acknowledge cycle or by a SYSRESET* from the VMEbus.

The four IRQ channels are associated and controlled by pins of the PI/T device as follows:

	VMEbus Default Address			
BIM Channel	Control Reg Address	Vector Reg Address	Toggle by PI/T	Readback of IRQ Flip-Flop state via PI/T
0	$A00001	$A00009	PA0	PA4
1	$A00003	$A0000B	PA1	PA5
2	$A00005	$A0000D	PA2	PA6
3	$A00007	$A0000F	PA3	PA7

The PI/T pins PA0/PA3 have to be programmed to outputs, PA4/PA7 are inputs. Before writing to the Port A data direction register, $FF has to be written to the Port A data register to avoid an unwanted output pulse.

4.3.4 The Interrupt Trigger Input

There is an option to trigger a level 7 interrupt to the local CPU from the VMEbus. To enable this option, the PI/T port interrupt (vectored) must be initialized by the local software.

The PI/T must be initialized so that a transition (low to high or high to low) on the H1 interrupt trigger input will cause an interrupt request. This interrupt request is of level 7, that is non-maskable. It must be the interrupt handler software that resets the interrupt. The VMEbus counterparts can be informed of the execution via the Dual Ported RAM or interrupt action via the BIM.

4.3.5 Reading the RUN/LOCAL Switch

The RUN/LOCAL switch controls whether or not the ISCSI-1 board can be accessed from the VMEbus. It would be of interest for the local software to distinguish this situation. The status of the RUN/LOCAL switch can be read by the local CPU via PI/T port PB6.

4.3.6 The Timer Section of the PI/T

The timer of the PI/T device is only of interest in the mode of a periodical interrupter. The timer interrupt has to be programmed so that the PI/T will support vectored interrupt acknowledge cycles.

The associated interrupt request level is level 5 of the CPU. This interrupt level is not subject to user configuration.

If the watchdog timer is used, it makes sense for the board to generate timer interrupts in the range of once every 5 to 10 milliseconds. The interrupt handler software should include some proof of correct system operation and finally the watchdog timer has to be triggered if the correct operation is stated.

The PI/T is clocked at a frequency of 8.0MHz.

4.3.7 The SCSIbus Reset Control

The SCSIbus reset is controlled via the PI/T port B bit #4 and #5. Bit #4 is programmed as an output, bit #5 as an input. To generate a reset on the SCSIbus it is necessary to clear ("Ø") bit #4 for at least 25 us and then set ("1") again. Bit #5 indicates the state of the SCSI reset.

4.3.8 The Single or Double Floppy Density Selection

The input of the floppy density selection, which selects either single (FM) or double (MFM) density is directly connected to port B bit #7 (programmed as an output). Bit #7 cleared ("Ø") selects double density, bit #7 set ("1") selects single density.

4.4 The Direct Memory Access Controller 68450 (DMAC)

The SYS68K/ISCSI-1 board contains a 4-channel DMA Controller (68450) with a clock frequency of 10MHz. The 68450 offers a wide variety of programmable transfer options as described in the data sheet in Register 5, Chapter 5.

4.4.1 The DMAC Implementation

The DMAC is an alternative bus master to the CPU, performing read/write cycles on-board.

The base address of the DMAC is $C80000. Byte and word transfers are possible.

The DMAC can interrupt the CPU on IRQ level 2. If jumper B41 is inserted, the DMAC supports a vector which is the contents of one of the eight interrupt vector registers. If jumper B41 is removed, an autovector interrupt is generated. The autovector level is 2.

The DMAC, when master, can access all memory locations accessible to the CPU (except for the DMAC registers).

The DMAC can be requested by the SCSI controller or by the FDC. To provide high speed data transfer on the SCSIbus, the SCSI controller can also be accessed in the "Single Address Mode".

The data sheet of the 68450 DMAC is included in Register 5, Chapter 5.

4.4.2 Addressing of the DMAC

The DMAC can either be programmed in 8 or in 16-bit mode. The data bits D0-D15 can be used for loading the DMAC.

The register model of the DMAC is listed in Table 4-2.

Table 4-2: Register Model of the DMAC

Base Address : $C80000

Address HEX	Offset	Reset Value	Label	Description
				C H A N N E L Ø
C80000	ØØ	Ø1	DMACSR	Channel Status Register
C80001	Ø1	ØØ	DMACER	Channel Error Register (Read)
C80004	Ø4	ØØ	DMADCR	Device Control Register
C80005	Ø5	ØØ	DMAOCR	Operation Control Register
C80006	Ø6	ØØ	DMASCR	Sequence Control Register
C80007	Ø7	ØØ	DMACCR	Channel Control Register
C8000A	ØA	--	DMAMTC	Memory Transfer Counter
C8000B	ØB	--		
C8000C	ØC	--	DMAMAR	Memory Address Register
C8000D	ØD	--		
C8000E	ØE	--		
C8000F	ØF	--		
C80014	14	--	DMADAR	Device Address Register
C80015	15	--		
C80016	16	--		
C80017	17	--		
C8001A	1A	--	DMABTC	Base Transfer Counter
C8001B	1B	--		
C8001C	1C	--	DMABAR	Base Address Register
C8001D	1D	--		
C8001E	1E	--		
C8001F	1F	--		
C80025	25	ØF	DMANIVR	Normal Interrupt Vector
C80027	27	ØF	DMAEIVR	Error Interrupt Vector
C80029	29	Ø7	DMAMFC	Memory Function Codes
C8002D	2D	ØØ	DMACPR	Channel Priority Register
C80031	31	Ø7	DMADFC	Device Function Codes
C80039	39	Ø7	DMABFC	Base Function Codes

cont'd...

Table 4-2 cont'd 68450 DMAC Register Layout

Base Address : $C80000

Address HEX	Offset	Reset Value	Label	Description
				C H A N N E L 1
C80040	40	01	DMACSR1	Channel Status Register
C80041	41	00	DMACER1	Channel Error Register (Read)
C80044	44	00	DMADCR1	Device Control Register
C80045	45	00	DMAOCR1	Operation Control Register
C80046	46	00	DMASCR1	Sequence Control Register
C80047	47	00	DMACCR1	Channel Control Register
C8004A	4A	--	DMAMTC1	Memory Transfer Counter
C8004B	4B	--		
C8004C	4C	--	DMAMAR1	Memory Address Register
C8004D	4D	--		
C8004E	4E	--		
C8004F	4F	--		
C80054	54	--	DMADAR1	Device Address Register
C80055	55	--		
C80056	56	--		
C80057	57	--		
C8005A	5A	--	DMABTC1	Base Transfer Counter
C8005B	5B	--		
C8005C	5C	--	DMABAR1	Base Address Register
C8005D	5D	--		
C8005E	5E	--		
C8005F	5F	--		
C80065	65	0F	DMANIVR1	Normal Interrupt Vector
C80067	67	0F	DMAEIVR1	Error Interrupt Vector
C80069	69	07	DMAMFC1	Memory Function Codes
C8006D	6D	00	DMACPR1	Channel Priority Register
C80071	71	07	DMADFC1	Device Function Codes
C80079	79	07	DMABFC1	Base Function Codes

cont'd...

Table 4-2 cont'd 68450 DMAC Register Layout

Base Address : $C80000

Address HEX	Offset	Reset Value	Label	Description
				C H A N N E L 2
C80080	80	01	DMACSR2	Channel Status Register
C80081	81	00	DMACER2	Channel Error Register (Read)
C80084	84	00	DMADCR2	Device Control Register
C80085	85	00	DMAOCR2	Operation Control Register
C80086	86	00	DMASCR2	Sequence Control Register
C80087	87	00	DMACCR2	Channel Control Register
C8008A	8A	--	DMAMTC2	Memory Transfer Counter
C8008B	8B	--		
C8008C	8C	--	DMAMAR2	Memory Address Register
C8008D	8D	--		
C8008E	8E	--		
C8008F	8F	--		
C80094	94	--	DMADAR2	Device Address Register
C80095	95	--		
C80096	96	--		
C80097	97	--		
C8009A	9A	--	DMABTC2	Base Transfer Counter
C8009B	9B	--		
C8009C	9C	--	DMABAR2	Base Address Register
C8009D	9D	--		
C8009E	9E	--		
C8009F	9F	--		
C800A5	A5	0F	DMANIVR2	Normal Interrupt Vector
C800A7	A7	0F	DMAEIVR2	Error Interrupt Vector
C800A9	A9	07	DMAMFC2	Memory Function Codes
C800AD	AD	00	DMACPR2	Channel Priority Register
C800B1	B1	07	DMADFC2	Device Function Codes
C800B9	B9	07	DMABFC2	Base Function Codes

cont'd...

Table 4-2 cont'd 68450 DMAC Register Layout

Base Address : $C80000

Address HEX	Offset	Reset Value	Label	Description
				C H A N N E L 3
C800C0	C0	01	DMACSR3	Channel Status Register
C800C1	C1	00	DMACER3	Channel Error Register (Read)
C800C4	C4	00	DMADCR3	Device Control Register
C800C5	C5	00	DMAOCR3	Operation Control Register
C800C6	C6	00	DMASCR3	Sequence Control Register
C800C7	C7	00	DMACCR3	Channel Control Register
C800CA	CA	--	DMAMTC3	Memory Transfer Counter
C800CB	CB	--		
C800CC	CC	--	DMAMAR3	Memory Address Register
C800CD	CD	--		
C800CE	CE	--		
C800CF	CF	--		
C800D4	D4	--	DMADAR3	Device Address Register
C800D5	D5	--		
C800D6	D6	--		
C800D7	D7	--		
C800DA	DA	--	DMABTC3	Base Transfer Counter
C800DB	DB	--		
C800DC	DC	--	DMABAR3	Base Address Register
C800DD	DD	--		
C800DE	DE	--		
C800DF	DF	--		
C800E5	E5	0F	DMANIVR3	Normal Interrupt Vector
C800E7	E7	0F	DMAEIVR3	Error Interrupt Vector
C800E9	E9	07	DMAMFC3	Memory Function Codes
C800ED	ED	00	DMACPR3	Channel Priority Register
C800F1	F1	07	DMADFC3	Device Function Codes
C800F9	F9	07	DMABFC3	Base Function Codes
				GLOBAL DEVICE CONTROL
C800FF	FF	00	DMAGCR	DMAC General Control Register

4.4.3 The DMAC Interrupt Scheme

The DMA Controller contains 8 interrupt vector registers which will be read during an interrupt acknowledge cycle of the CPU if jumper B41 is inserted; otherwise, an auto vector is forced.

4.4.4 The DMAC Summary

Default Base Address :	$C80000
Default Access Address :	$C80000 - $C800FF
Access Mode :	Byte and Word Read and Write
Usable Data Bits :	D0-D7, D8-D15
Interrupt Level :	IQ2
Interrupt Vector :	Auto vectors/non-autovector

4.5 The SCSIbus Controller NCR 5386S (SCSIBC)

- The NCR 5386S SCSIbus Controller communicates with the 68Ø1Ø CPU and the 6845Ø DMAC as a peripheral device. The SCSIBC is controlled by reading and writing several internal registers which are addressed via the local address bus.

- Since the SCSIBC interrupts the MPU when it detects a SCSIbus condition that requires servicing, the MPU is free from polling or controlling any of the SCSIbus signals.

- The SCSIBC will be programmed and controlled via the local CPU.

- The SCSIBC Interrupt is connected to the CPU interrupt level 3. The generated interrupt is always an auto vector interrupt.

- For high speed data transfer the SCSIBC communicate directly with the Dual Ported RAM under the DMAC control. In this mode the maximum data transfer rate will be 1.5Mbyte(s).

- The ISCSI-1's own SCSI I.D. is software programmable and is stored in the "Control Register" at the local address $CCØØØ9. For more information refer to section 4.7.

- The SCSIBC is connected to the SCSIbus via the SCSIbus driver/receiver chip NCR 831Ø.

- The data sheet of the SCSIBC is included in Register 5, Chapter 3.

Table 4-3: SCSIBC NCR 5386S Register Address Map

Base Address: $C40000					
Address Hex	Offset	Reset Value	Mode	Label	Description
$C40001	01	--	R/W	SDAT1	Data Register 1
$C40003	03	--	R/W	SCMDR	Command Register
$C40005	05	--	R/W	SCTRLR	Control Register
$C40007	07	--	R/W	SDID	Destination I.D.
$C40009	09	82	R	SASTAT	Auxiliary Status
$C4000B	0B	--	R	SIDR	I.D. Register
$C4000D	0D	01	R	SIDQ	Interrupt Register
$C4000F	0F	--	R	SSID	Source I.D.
$C40011	11	--	R/W	SDAT	Data Register 2
$C40013	13	98	R	SDSTAT	Diagnostic Status
$C40019	19	--	R/W	STCH	Transfer Counter High
$C4001B	1B	--	R/W	STCM	Transfer Counter Mid
$C4001D	1D	--	R/W	STCL	Transfer Counter Low
$C4001F	1F	--	R/W	STST	Reserved for Testability

4.5.1 SCSIbus Controller Summary

Base Address: $C40000

Access Address: $C40001 - $C4001F

Access Mode: Byte
Read and Write

Usable Data bits for programming: D0 - D7

Interrupt Level: IQ Level 3

Interrupt Vector: Auto Vector #3

4.6 The Floppy Disk Controller WD 1772 (FDC)

In order to provide easy connection to floppy drives, the ISCSI-1 board includes a Floppy Disk Controller (FDC)

This FDC, which is able to control up to four floppy drives (3", 3 1/2" or 5 1/4"). The FDC is fully supported by the firmware.

The FDC is programmed and controlled by the on-board CPU. Data transfer is performed via the 68450 DMAC.

The FDC interrupt output is connected to the CPU interrupt level 4. The generated interrupt is an auto vector interrupt.

To select single (FM) or double (MFM) density, bit #7 of the PI/T port B must be cleared or set. Bit #7 cleared ("0") selects double density, bit #7 set ("1") selects single density.

To allow the connection of all SHUGART-compatible floppy disk drives, the Drive Select 0 to 3 signals, the Side Select signal and the single/double density signal are software programmable. The current values of these signals will be written into the Control Register and the PI/T Port B. For more information refer to Chapter 4.7 and Chapter 4.3.8.

The datasheet of the FDC is included in Register 5, Chapter 6.

Table 4-4: The FDC WD 1772 Register Address Map

Base Address : $CC0000					
Address HEX	Offset	Reset Value	Mode	Label	Description
$CC0001	01	7C	R	FSTR	Status Register
$CC0001	01	--	W	FCR	Command Register
$CC0003	03	FF	R/W	FTR	Track Register
$CC0005	05	01	R/W	FSR	Sector Register
$CC0007	07	00	R/W	FDR	Data Register

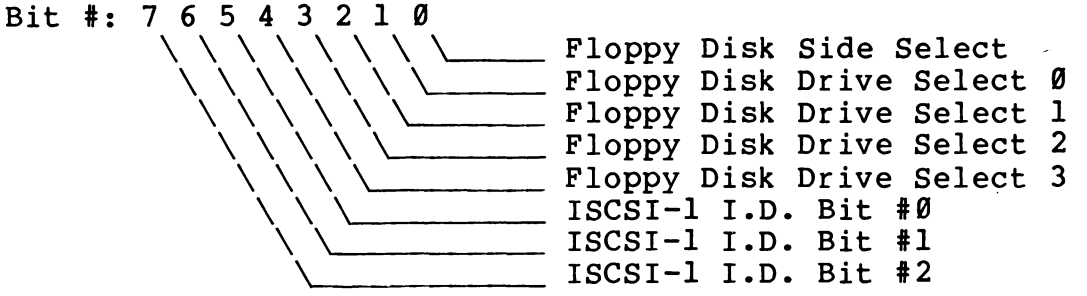
4.6.1 Floppy Disk Controller Summary

Base Address:	$CC0000
Access Address:	$CC0001 - $CC0007
Access Mode:	Byte Read and Write
Usable Data bits for programming:	D0 - D7
Interrupt Level:	IQ Level 4
Interrupt Vector:	Auto Vector #4

4.7 The Control Register

Some I/O and control signals on the board are software programmable, in order to provide easy handling. The on-board Control Register, which handles these signals is designed as a "Read Back Register".

The Control Register bit assignment is shown below:



The control register address is #CC0009.

Access modes available are: Byte
Read and Write

4.8 The SYS68K/ISCSI-1BPS

There is an optional back panel called SYS68K/ISCSI-1BPS available from FORCE Computers.

This board can be plugged into the P2 connector of the ISCSI-1 board. It contains an SCSIbus connector (X1) and a floppy disk interface connector (X2). The pinout of these connectors is shown in Table 4-6 and Table 4-7.

Figure 4-1 shows how the ISCSI-1BPS can be connected to the ISCSI-1.

Table 4-5: The ISCSI-1 P2 Pin Assignment

Pin Number	Row A Signal Mnemonic	Row B Signal Mnemonic	Row C Signal Mnemonic
1	DB Ø	VCC	N.C.
2	DB 1	GND	N.C.
3	DB 2	N.C.	Drive Select Ø
4	DB 3	N.C.	Index
5	DB 4	N.C.	Drive Select 1
6	DB 5	N.C.	Drive Select 2
7	DB 6	N.C.	Drive Select 3
8	DB 7	N.C.	Motor On
9	DB P	N.C.	Direction In
1Ø	GND	N.C.	Step
11	GND	N.C.	Write Data
12	GND	GND	Write Gate
13	TERMPWR	VCC	Track ØØØ
14	GND	N.C.	Write Protect
15	GND	N.C.	Read Data
16	ATN	N.C.	Side Select
17	GND	N.C.	N.C.
18	BSY	N.C.	N.C.
19	ACK	N.C.	GND
2Ø	RST	N.C.	GND
21	MSG	N.C.	N.C.
22	SEL	GND	GND
23	C/D	N.C.	GND
24	REQ	N.C.	N.C.
25	I/O	N.C.	N.C.
26	N.C.	N.C.	N.C.
27	GND	N.C.	RESERVED
28	N.C.	N.C.	RESERVED
29	RESERVED	N.C.	RESERVED
3Ø	RESERVED	N.C.	RESERVED
31	RESERVED	GND	RESERVED
32	RESERVED	VCC	RESERVED

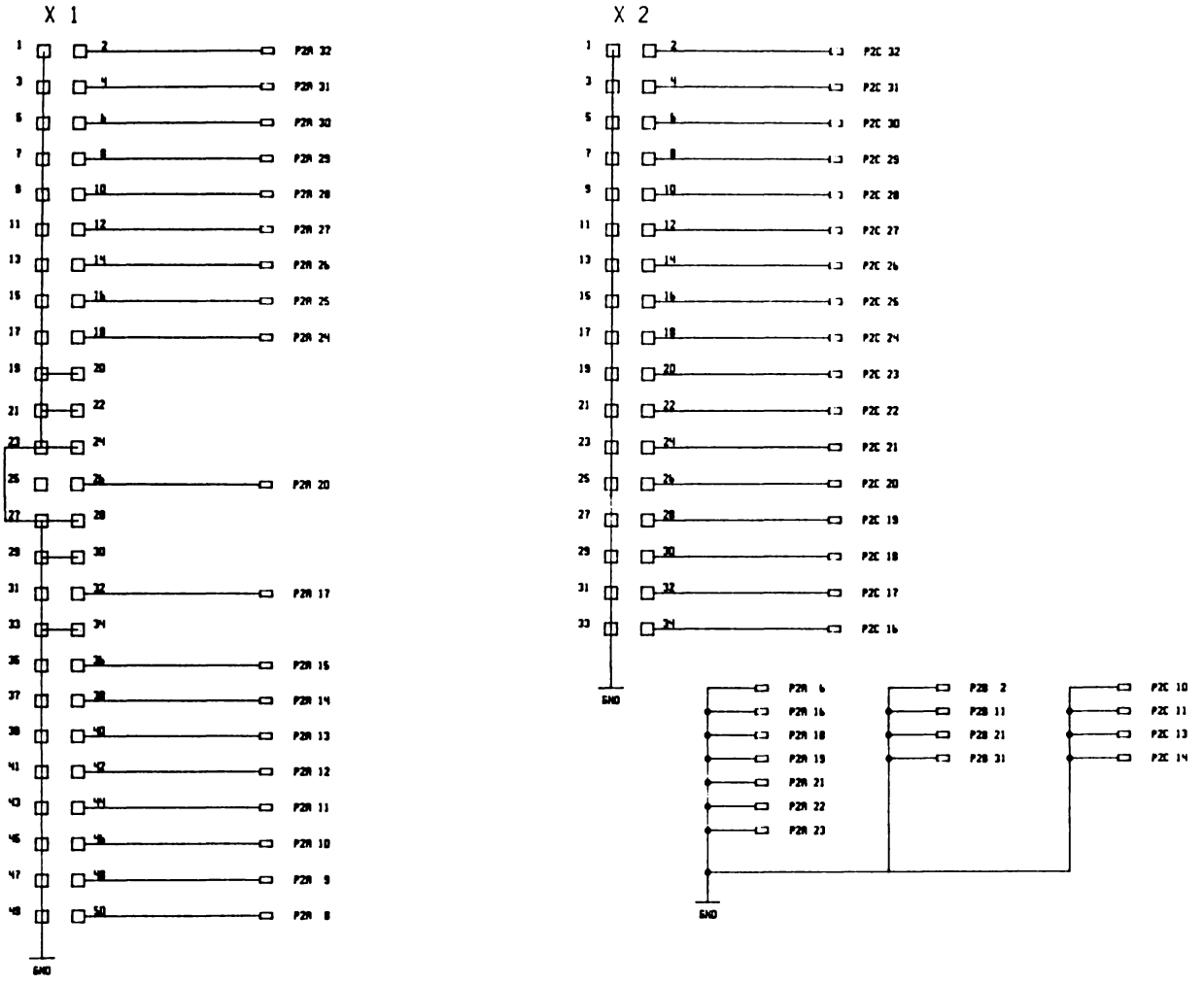
Table 4-6: The ISCSI-1BPS X1 Pin Assignment

Pin No.	Signal Mnemonic	Pin Number	Signal Mnemonic
2	DB Ø	1	GND
4	DB 1	3	GND
6	DB 2	5	GND
8	DB 3	7	GND
1Ø	DB 4	9	GND
12	DB 5	11	GND
14	DB 6	13	GND
16	DB 7	15	GND
18	DB P	17	GND
2Ø	GND	19	GND
22	GND	21	GND
24	GND	23	GND
26	TERMPWR	25	N.C.
28	GND	27	GND
3Ø	GND	29	GND
32	ATN	31	GND
34	GND	33	GND
36	BSY	35	GND
38	ACK	37	GND
4Ø	RST	39	GND
42	MSG	41	GND
44	SEL	43	GND
46	C/D	45	GND
48	REQ	47	GND
5Ø	I/O	49	GND

Table 4-7: The ISCSI-1PBS X2 Pin Assignment

Pin No.	Signal Mnemonic	Pin Number	Signal Mnemonic
2	N.C.	1	GND
4	N.C.	3	GND
6	Drive Select Ø	5	GND
8	Index	7	GND
1Ø	Drive Select 1	9	GND
12	Drive Select 2	11	GND
14	Drive Select 3	13	GND
16	Motor On	15	GND
18	Direction In	17	GND
2Ø	Step	19	GND
22	Write Data	21	GND
24	Write Gate	23	GND
26	Track ØØØ	25	GND
28	Write Protect	27	GND
3Ø	Read Data	29	GND
32	Side Select	31	GND
34	N.C.	33	GND

Figure 4-2: Photo of the SYS68K/ISCSI-1BPS



4.9 The Dual Ported RAM as Local Memory

There is 128Kbyte of static memory installed on the SYS68K/ISCSI-1 board. This RAM is built up with a dual port access structure so that it can be read and written by the local CPU and by the VMEbus master.

There is an access control mechanism installed that decides which side performs a memory cycle (Read or Write). This access control mechanism is controlled by the clock of the local CPU, so that the local CPU is always granted an access cycle in a way that no wait cycles are needed. No wait cycles of the local CPU are granted for the case of the maximum access rate from the VMEbus side. This situation allows local program execution times to be calculated.

4.9.1 Local CPU Access to the DPR

The local CPU access range of the DPR is

$000000 - $01FFFF.

This range can be accessed in all addressing modes of the local CPU.

During the boot-up procedure, only the local EPROMs can be selected and the processor reads start-up information from the EPROM even though during normal operation, the corresponding addresses select the DPR area.

The boot-up procedure's special decoding ends with the first Write cycle of the local CPU; from then on, the DPR decoding range is valid.

The address range

$000000 - $001FFF

cannot be accessed from the VMEbus.

The address range

$002000 - $01FFFF

is Dual Ported Memory.

4.9.2 Local CPU Read-Modify-Write Cycles

The local CPU is allowed to perform Read-Modify-Write operations to the Dual Ported RAM.

This operation is nondivisible for the local software, so it can be used to coordinate for example, between several local interrupt driven tasks.

The Dual Ported RAM is not blocked for external access between the Read and Write cycles of the local processor's Read-Modify-Write operation, nor is the local processor's access blocked between the VMEbus RMW access cycles. So the DPR cannot be used with Read-Modify-Write cycles to synchronize between local and VMEbus processes.

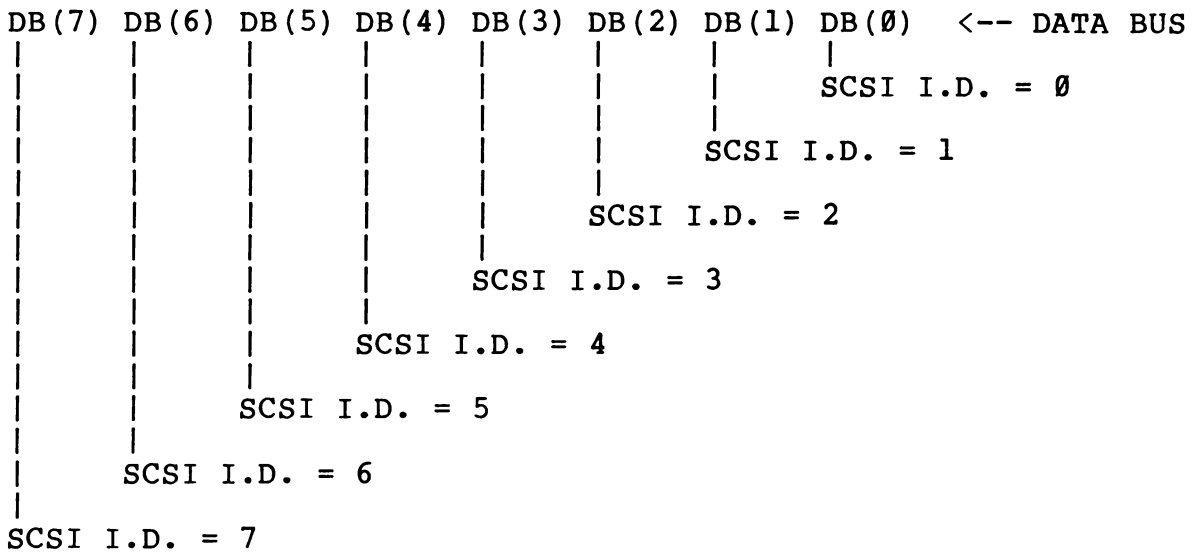
5.0 The SCSIbus Description

5.1 The SCSIbus Configuration

Communication on the SCSIbus is allowed between only two SCSI devices at any given time. There is a maximum of eight SCSI devices. Each SCSI device has an SCSI I.D. bit assigned as shown in Figure 5-1. When two SCSI devices communicate on the SCSIbus, one acts as an initiator, and the target performs the operation. An SCSI device usually has a fixed role as an initiator or target, but some devices may be able to assume either role.

An initiator may address up to eight peripheral devices that are connected to a target. An option allows the addressing of up to 2.048 peripheral devices per target using extended messages. Three sample system configurations are shown in Figure 5-2.

Figure 5-1: SCSI I.D. Bits

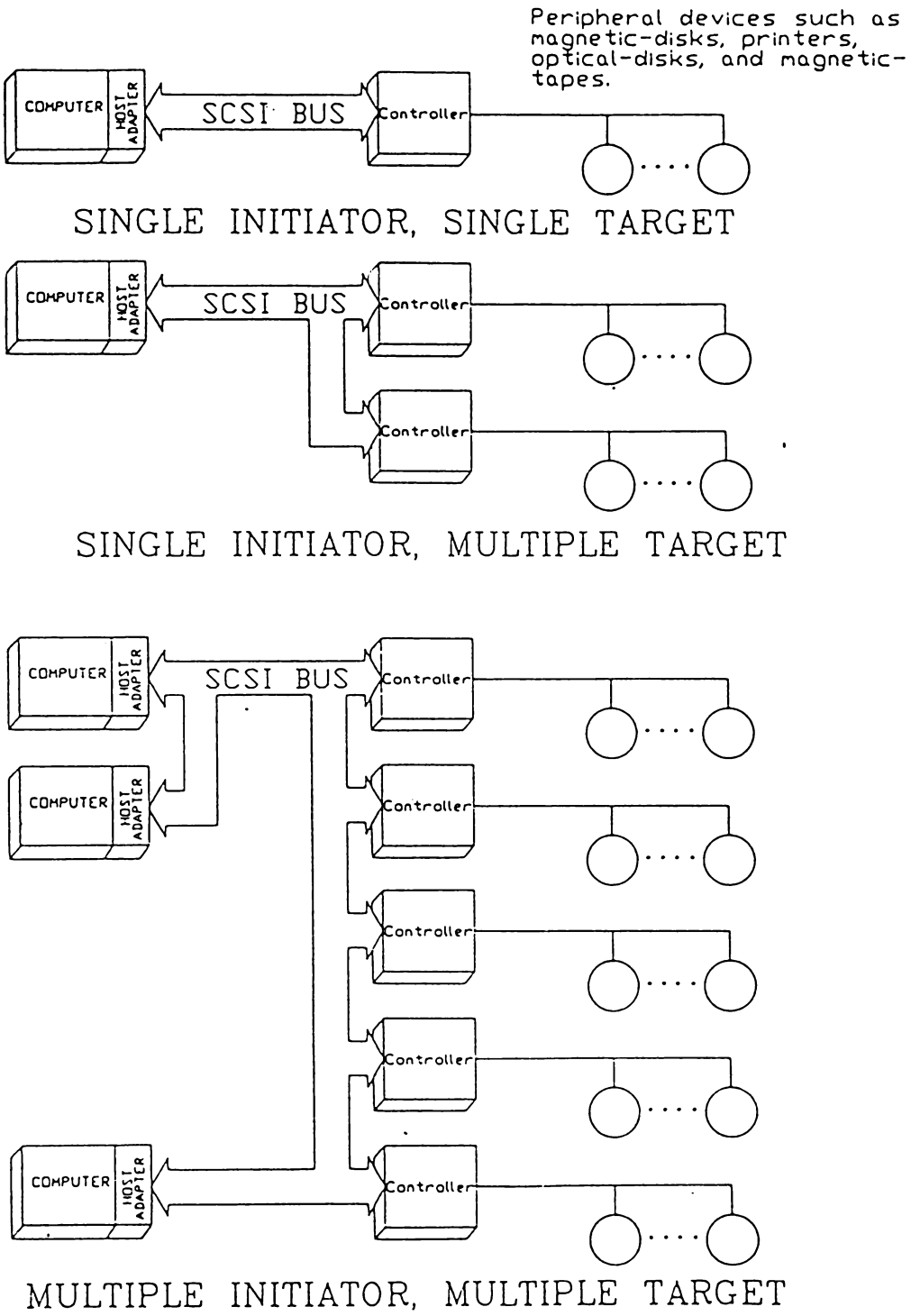


Up to eight SCSI devices can be supported on the SCSIbus. They can be any combination of initiators and targets.

Certain SCSIbus functions are assigned to the initiator and certain SCSIbus functions are assigned to the target. The initiator may arbitrate for the SCSIbus and select a particular target. The target may request the transfer of COMMAND, DATA, STATUS, or other information on the data bus, and in some cases it may arbitrate for the SCSIbus and reselect an initiator for the purpose of continuing an operation.

Information transfers on the data bus are asynchronous and follow a defined REQ/ACK handshake protocol. One byte of information may be transferred with each handshake. A option is defined for synchronous data transfer.

Figure 5-2: Sample SCSI Configurations



There are a total of eighteen signals. Nine are used for control, and nine are used for data. (Data signals include the parity signal option). These signals are described as follows:

BSY (BUSY) — An "OR-tied" signal that indicates that the bus is being used.

SEL (SELECT) — A signal used by an initiator to select a target or by a target to reselect an initiator.

C/D (CONTROL/DATA) — A signal driven by a target that indicates whether CONTROL or DATA information is on the data bus. True indicates CONTROL.

I/O (INPUT/OUTPUT) — A signal driven by a target that control the direction of data movement on the data bus with respect to an initiator. True indicates input to the initiator. This signal is also used to distinguish between SELECTION and RESELECTION phases.

MSG (MESSAGE) — A signal driven by a target during the MESSAGE phase.

REQ (REQUEST) — A signal driven by a target to indicate a request for a REQ/ACK data transfer handshake.

ACK (ACKNOWLEDGE) — A signals driven by an initiator to indicate an acknowledgement for a REQ/ACK data transfer handshake.

ATN (ATTENTION) — A signal driven by an initiator to indicate the ATTENTION condition.

RST (RESET) — An "OR-tied" signal that indicates the RESET condition.

DB(7-Ø,P) (DATA BUS) — Eight data-bit signals, plus a parity-bit signal which together form a data bus. DB(7) is the most significant bit, and has the highest priority during the ARBITRATION phase. Bit number, significance, and priority decrease downward to DB(Ø). A data bit is defined as one when the signal value is true, and is defined as zero when the signal value is false.

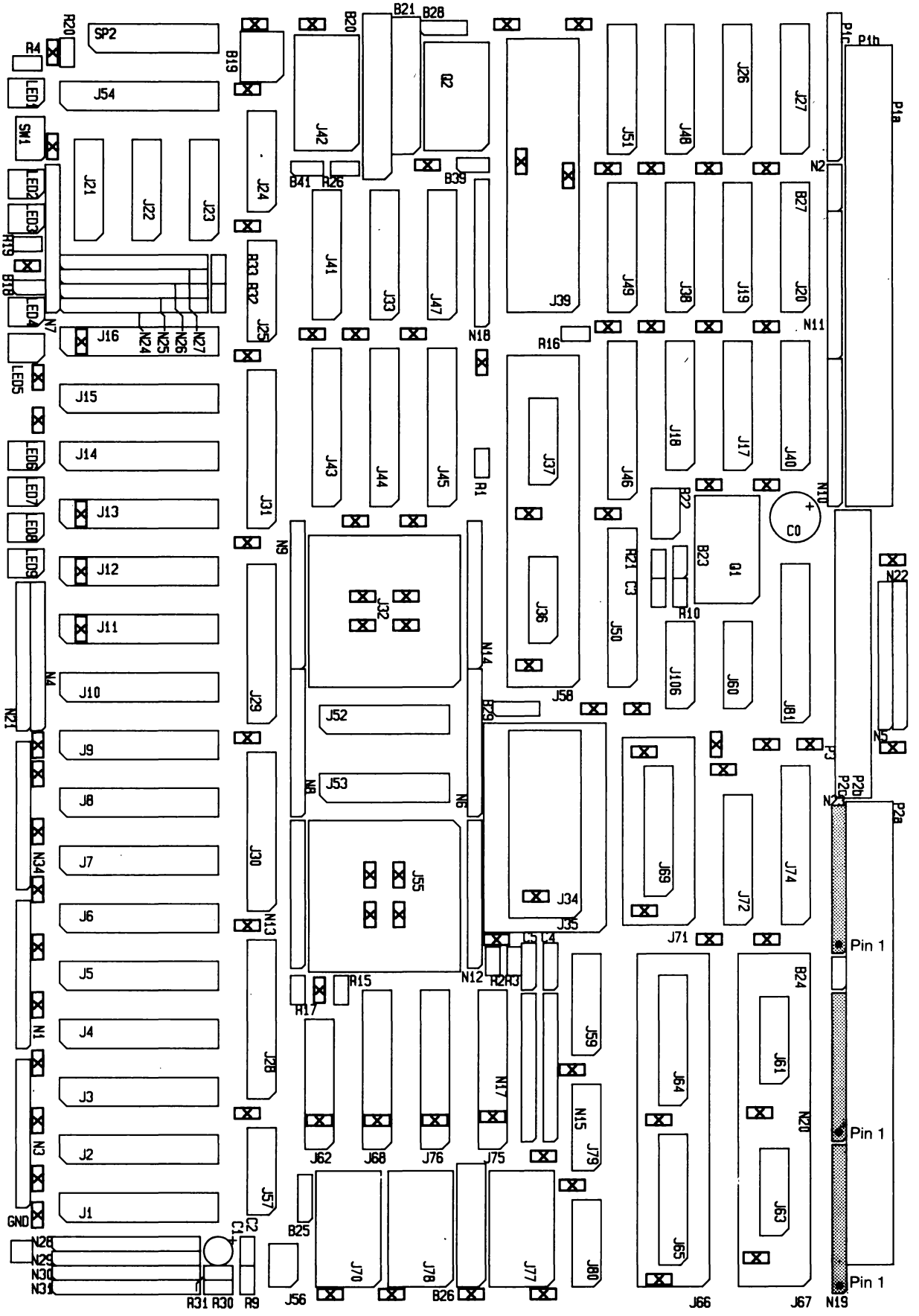
Data parity DP(P) is odd. The use of parity is a system option (i.e. a system is configured so that all SCSI devices on a bus generate parity and have parity detection enabled, or all SCSI devices have parity detection disabled or not implemented). Parity is not valid during the ARBITRATION phase.

5.3 The SCSIbus Signal Termination

Each SCSIbus signal is terminated at the physical start and the physical end of the SCSIbus. Therefore, the terminators must be removable, in order to join one SCSI device to two others.

Figure 5-3 shows the location of the terminators on the SYS68K/ISCSI-1 board. These terminators are plugged into sockets and can be removed if necessary.

Figure 5-3: Location of the SCSIbus Terminators



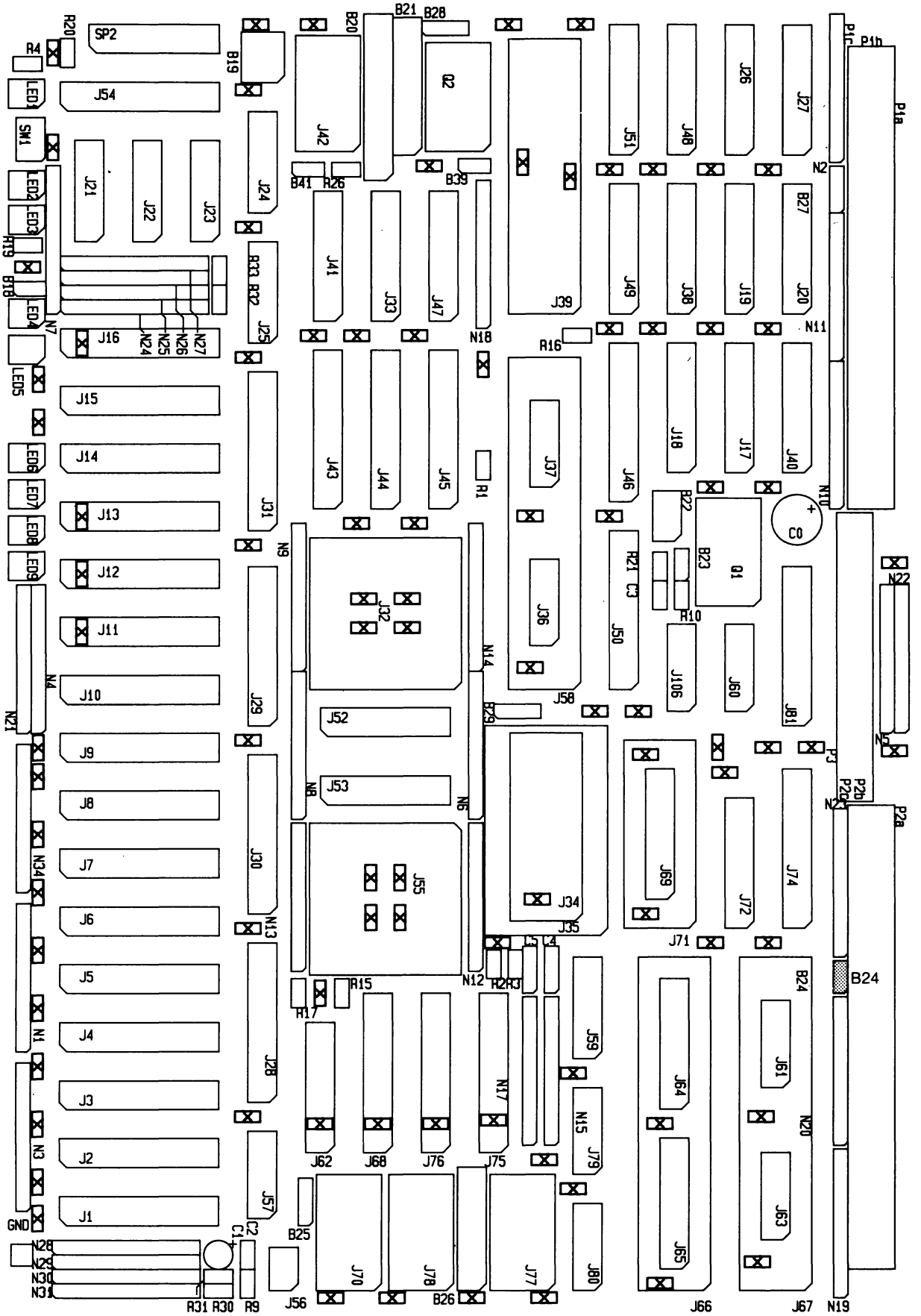
5.4 The SCSIbus Terminator Power

The power for the terminator of any SCSI device can be provided from the SYS68K/ISCSI-1 board, providing the SCSI device supports this optional case.

If jumper B24 is inserted, termination power is delivered from the ISCSI-1 board, and if jumper B24 is not inserted, termination power must be delivered from the SCSI device itself.

The location of the jumper B24 is shown in Figure 5-4.

Figure 5-4: Location of the Jumperfield B24



6.Ø The VMEbus Interface Hardware

The SYS68K/ISCSI-1 performs the functions of a VMEbus slave and interrupter.

The VMEbus Address Modifier decoding as well as the address range decoding are user configurable via jumpers.

The VMEbus interface is installed to decode A1-A23 of the standard address range. A 16-bit wide data bus is supported.

As an interrupter, the SYS68K/ISCSI-1 can be programmed to drive four interrupt request (IRQ) functions. The IRQ level is individually software programmable to any of IRQ1 to IRQ7.

A front panel switch, labelled R/L, allows the VMEbus decoding of the board to be enabled/disabled. A reset and an interrupt trigger call is supported, as is board status readout.

6.1 VMEbus Access to the Board

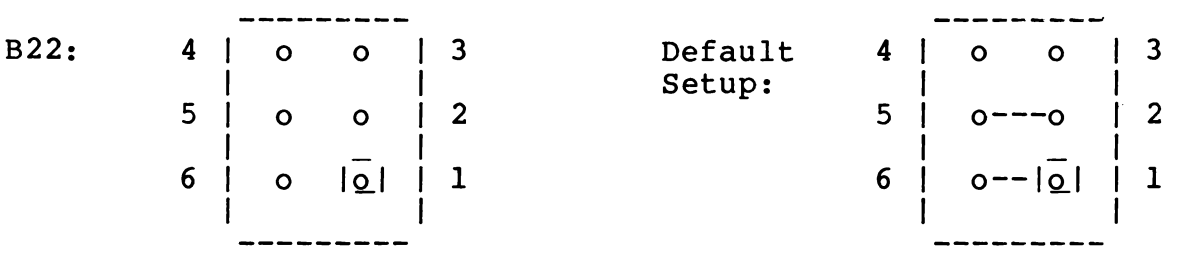
The SYS68K/ISCSI-1 will be accessed from the VMEbus if the current master drives a data transfer cycle under the following conditions:

1) Valid Address Modifier decoding
2) Valid addressing to the board's address range
3) RUN/LOCAL switch (R/L) set to RUN

During each access cycle, the yellow ACCESS LED on the front panel is lit to give optical information.

6.1.1 The Address Modifier Code Selection

The Address Modifier code is defined in the VMEbus specification P1Ø14 of IEEE. The Address Modifier decoding of the board is configured at B22 via jumpers.



More than one condition can be enabled simultaneously. In the default condition with the jumpers 1-6 and 2-5 installed, both supervisory and non-privileged accesses in the standard addressing mode will be recognized as valid (P1Ø14 compatible).

The following selections are installed:

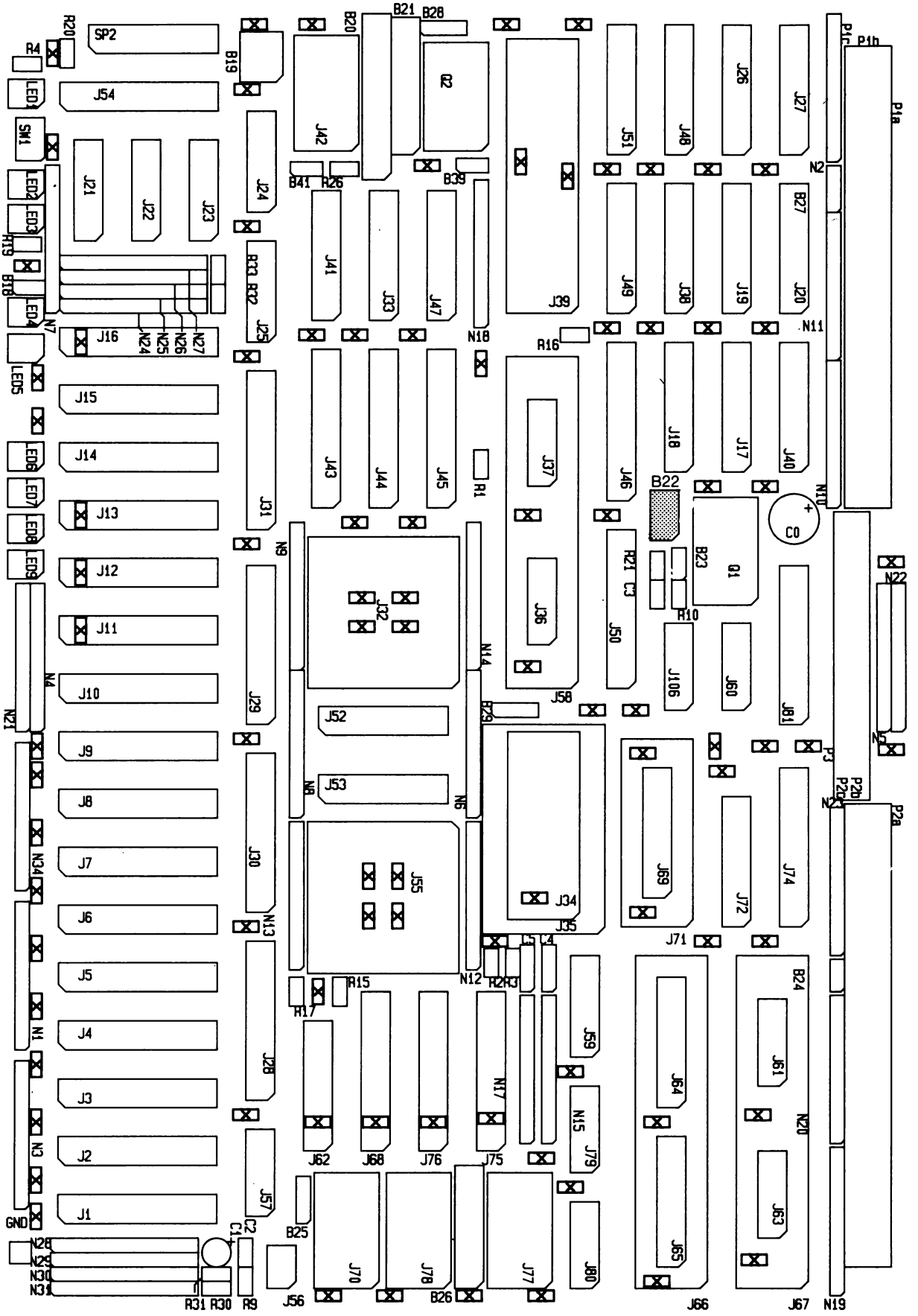
No jumpers : Ignore AM codes (Not P1Ø14 compatible)

1 - 6 : Standard non-privileged program and data access (codes $3A and $39) are enabled

2 - 5 : Standard supervisory program and data access (codes $3E and $3D) are enabled

3 - 4 : Code $3F (not P1Ø14 compatible)

Figure 6-1: Location Diagram of Jumperfield B22

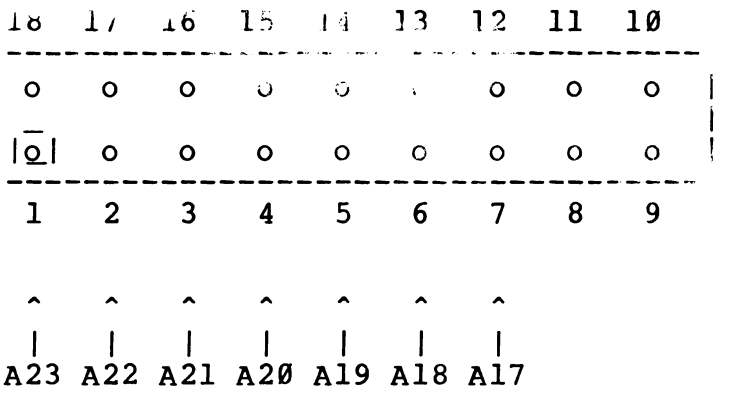


6.1.2 The Board Base Address Selection

The SYS68K/ISCSI-1 board occupies an address range of 128Kbytes of the VMEbus standard address range.

The base address of the board address range can be selected in 128Kbyte steps. The VMEbus address signals A23-A17 are compared to the base address jumper setting at B21. In case of coincidence, the board select condition is met.

The jumperfield B21 is as shown here:



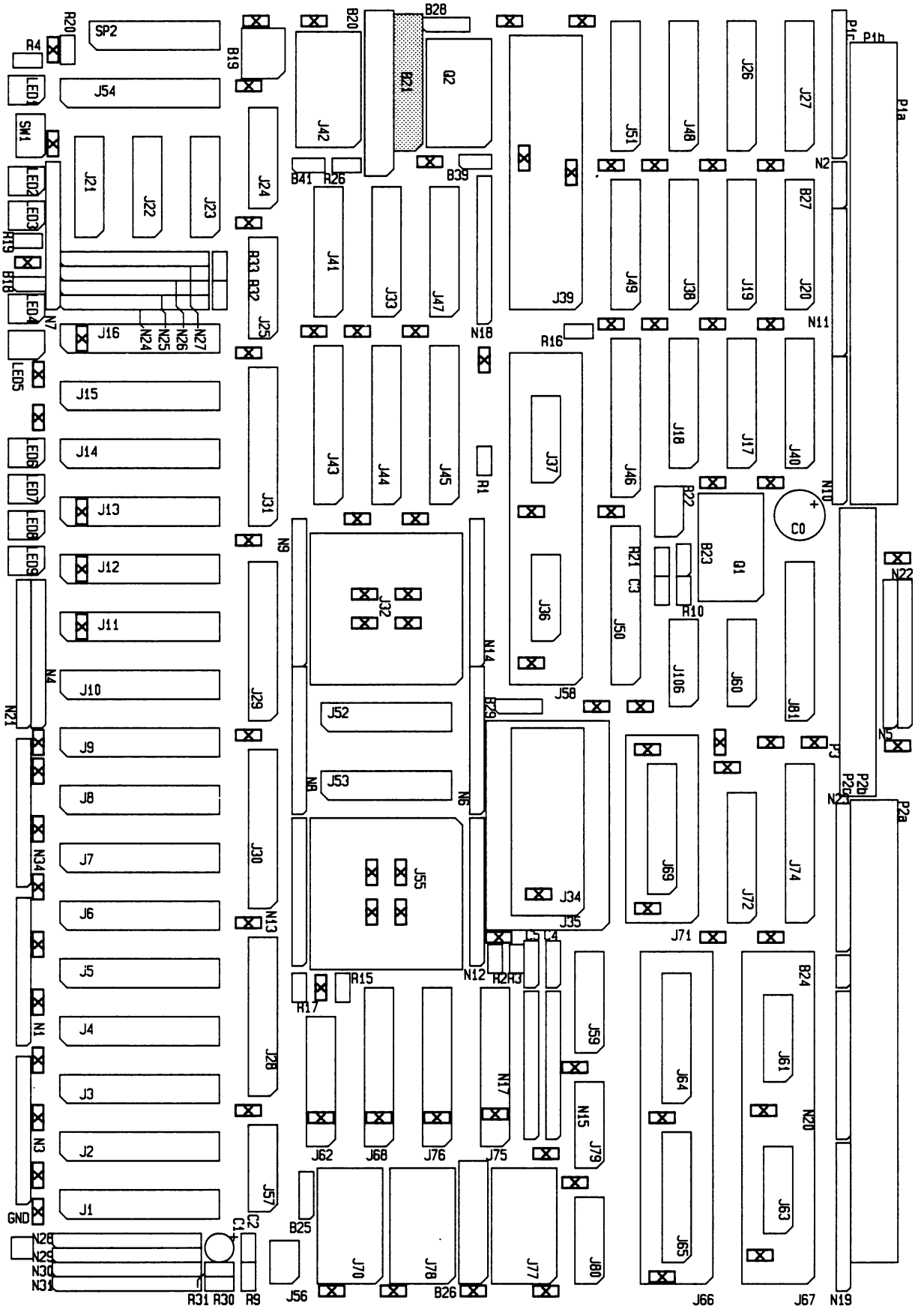
The following jumper connections are allowed:

B21 Connections	Address Signal	Jumper Meaning
1 - 18	A23	in = Ø, out = 1
2 - 17	A22	in = Ø, out = 1
3 - 16	A21	in = Ø, out = 1
4 - 15	A2Ø	in = Ø, out = 1
5 - 14	A19	in = Ø, out = 1
6 - 13	A18	in = Ø, out = 1
7 - 12	A17	in = Ø, out = 1

The default base address of the board is $A00000.

B21 default connections:	$A00000
	A23 = 1
2 - 17	A22 = Ø
4 - 15	A21 = 1
5 - 14	A2Ø = Ø
6 - 13	A19 = Ø
7 - 12	A18 = Ø
	A17 = Ø

Figure 6-2: Location Diagram of Jumperfield B21



6.1.3 The RUN/LOCAL Switch

The R/L switch on the front panel allows VMEbus accesses to the ISCSI-1 board to be enabled/disabled.

If the R/L switch is turned on, then the red LOCAL LED is lit and the board cannot be accessed.

If the R/L switch is turned off, then the green RUN LED is lit and the board can be accessed from the VMEbus. An access occurs if a data cycle is driven with the valid AM code and the selected addressing range is hit.

6.1.4 The Access LED

There is a yellow LED, labelled SEL, mounted on the front panel. If a VMEbus access to the board occurs, then this yellow LED is lit. It is also lit during interrupt acknowledge cycles to the board.

6.2 The Address Map of the VMEbus Address Range

There are five addressable objects that can be accessed by the VMEbus master:

1) The BIM

2) The status byte

3) The interrupt trigger

4) The reset call

5) The Dual Ported RAM

The VMEbus address map is as follows:

Offset to Board Base Address	Default Address	Read Access	Write Access	Access Mode	Register Memory
$000000	$A00000	x	--	Word only	Bit 8 = Local RESET state Bit 9 = Local CPU HALT state Bit 10 = Watchdog Timer State
$000001	$A00001	x	x	Byte	BIM Control 0
$000003	$A00003	x	x	Byte	BIM Control 1
$000005	$A00005	x	x	Byte	BIM Control 2
$000007	$A00007	x	x	Byte	BIM Control 3
$000009	$A00009	x	x	Byte	BIM Vector 0
$00000B	$A0000B	x	x	Byte	BIM Vector 1
$00000D	$A0000D	x	x	Byte	BIM Vector 2
$00000F	$A0000F	x	x	Byte	BIM Vector 3
$001001	$A01001	x	--	Byte	Interrupt trigger call
$001801	$A01801	x	--	Byte	Reset trigger call
$002000 to $01FFFF	$A02000 to $A0FFFF	x x	x x	Byte and Word	Dual Ported RAM

6.3 The Dual Ported RAM (DPR) as VMEbus Memory

The Dual Ported RAM installed on the SYS68K/ISCSI-1 board can be accessed by the local CPU and by a VMEbus master.

The VMEbus access addressing range is by default $A02000-$A1FFFF which is an offset range of $002000-$01FFFF relative to the board base address.

The total size of the RAM is 128Kbyte but the low end 8Kbyte are only local accessible. The actual DPR size is 120Kbyte. The DPR is byte and word mode, read and write accessible for VMEbus masters. Read-Modify-Write accesses are also allowed, but this method can not be used to synchronize between local and VMEbus procedures. The Read-Modify-Write cycles of the local CPU can occur interleaved with the VMEbus access.

6.4 The Reset Trigger Call

A read access from the VMEbus to the byte location with the default address $A01801 that is $001801 offset to the board base address, is reset trigger for the local CPU hardware.

Upon this reset trigger call, the local CPU stops and resets, booting up with the "cold start" in the same way as after the system reset. The call causes a reset pulse on the local side which is long enough for the CPU to reset.

The status of the local RESET signal can be read out via the status register (see Chapter 6.7).

The reset trigger call does not reset the BIM device nor does it reset the interrupt request flip-flops (Chapter 4.3.3).

In a multi-master environment, it is possible that one master triggers reset to the SYS68K/ISCSI-1 and then another master acknowledges an interrupt. Interrupt service routines should begin and end with a check on the SYS68K/ISCSI-1's status register.

6.5 The Interrupt Trigger Call

A non-maskable level 7 interrupt to the local CPU can be triggered by a read access via an addressable location on the SYS68K/ISCSI-1 board. The byte read access has to be addressed by default to $A01001 or to the offset $001001 relative to the board base address. This access does not operate without local software initialization. The trigger signal is routed to the H1 interrupt trigger input of the PI/T device. If software initializes the PI/T to activate H1, to perform IRQ to the CPU and the proper interrupt handling routine is installed, then this interrupt call will be perceived and responded to. The PI/T data sheet includes more details on programming. See also Chapter 4.3.4.

6.6 The Bus Interrupter Module

There is a 68153 Bus Interrupter Module (BIM) installed on the SYS68K/ISCSI-1 board. The device has eight addressable register bytes. The BIM is only accessible from the VMEbus at the address locations which are shown in Chapter 6.2.

The data sheet of the BIM is included in this manual.

If the SYS68K/ISCSI-1 board has to perform interrupt requests to the VMEbus then the corresponding VMEbus master has to program the BIM via register write operations to enable and to drive the IRQ level required. The interrupt vector has to be programmed as well. The four channels of the BIM can be operated by four masters independent of each other.

The description of how the local CPU can drive interrupt requests to the BIM channels is shown in Chapter 4.3.3

There is absolutely no hardware interdependence between the local CPU's interrupt structure and the VMEbus interrupt structure.

All interrupt interfacing has to be built up and needs individual control by local software and by the VMEbus master which needs to be interrupted by the ISCSI-1.

Any sequence that leads the IRQ to the VMEbus has to be executed so that first of all the BIM is programmed by the VMEbus master to the proper vector and interrupt request level. Then the ISCSI-1 software can be triggered to start operations that include a trigger for an interrupt request channel.

Warning: The reset trigger call does not reset interrupt requests pending, only the system reset of the VMEbus can do this. Interrupt request are cleared by the execution of the corresponding interrupt acknowledge cycle.

6.7 The Status Register

There is a status register included in the ISCSI-1 board. The status register is a read only function. If the board base address is accessed in the word mode of addressing, then the data bits 8, 9 and 10 give status information to the ISCSI-1 board.

It is legal to read each of the BIM registers as words together with the status information in bits 8, 9 and 10. The word write access is functionally identical to the corresponding odd byte write to the BIM.

The meaning of the bits on the status register:

Bit 8: If the local CPU's RESET signal is low, then the bit is 0, if not the bit is 1.

This status read is especially important after a reset trigger call, it must be monitored to decide if the board can be operated.

Bit 9: The local CPU's HALT state is monitored via this bit. A 0 means that the CPU's HALT signal is low. HALT is low during the RESET of the CPU, or after a double bus fault condition.

During normal CPU operation, HALT is high and the bit is read as 1. The only way of restarting the board after the CPU is halted is by RESET. This can be a global VME system RESET or a reset trigger call to the ISCSI-1 board.

Bit 10: The state of the watchdog timer on the board can be monitored by this status bit. If the watchdog timer is triggered regularly (e.g. every 10 milliseconds), then this bit is always 1.

If the trigger is not operated (e.g. because of system failure), then the watchdog output turns to low and this bit reads 0.

If the CPU is in the HALT state, then the watchdog timer trigger stops as well. If the watchdog time has elapsed, reporting a malfunction and the local CPU does not go into HALT state, then it is possible to return to the normal operation without resetting the board.

APPENDIX TO THE HARDWARE USER'S MANUAL

A P P E N D I C E S

Appendix A

Specification of the SYS68K/ISCSI-1

Microprocessor	68010, 10MHz
Control	68230 PI/T for local control and timer function
Interrupter	68153 BIM
RAM	128Kbyte Dual Ported RAM, No Wait States during local access
EPROM	128Kbyte (max) 32Kbyte (default) (JEDEC compatible, 150ns) Only local access: zero wait state
SCSIbus Controller	NCR 5386S
Floppy Interface	WD 1772
VMEbus/IEEE P1014	Fully VMEbus compatible slave and interrupter interface
Power requirements	+ 5V/5.6A (max)
Operating temperature	0 to 50 degrees C
Storage temperature	-50 to 85 degrees C
Relative humidity	0 to 95% (non-condensing)
Board dimensions	Double eurocard 234 x 160mm (9.2" x 6.3")

Appendix B

Memory Map of the SYS68K/ISCSI-1

Local CPU Memory Map

$000 000 to $001 FFF	8Kbyte SRAM (Local only)
$002 000 to $01F FFF	120Kbyte SRAM (Dual Ported)
$C00 000 to $CFF FFF	Local Devices
$E00 000 to $E1F FFF	EPROM

VMEbus Memory Map

$A00 000 to $A01 FFF	Control and Status (BIM)
$A02 000 to $A1F FFF	120Kbyte SRAM (Dual Ported)

Appendix C

Address Assignment of the On-Board Devices

Local CPU Devices

Base Address:

$C40 000	SCSIbus Controller
$C80 000	Direct Memory Access Controller
$CC0 000	Floppy Disk Controller
$CC0 009	Control Register
$D00 000	PI/T

VMEbus Devices

Base Address:

$A00 000	BIM and Status Register
$A01 001	Interrupt Trigger
$A01 801	Reset Register

Appendix D

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

ICs

Location	Type	Manufacturer
J1	4361-45	NEC, INMOS
J2	4361-45	NEC, INMOS
J3	4361-45	NEC, INMOS
J4	4361-45	NEC, INMOS
J5	4361-45	NEC, INMOS
J6	4361-45	NEC, INMOS
J7	4361-45	NEC, INMOS
J8	4361-45	NEC, INMOS
J9	4361-45	NEC, INMOS
J10	4361-45	NEC, INMOS
J11	4361-45	NEC, INMOS
J12	4361-45	NEC, INMOS
J13	4361-45	NEC, INMOS
J14	4361-45	NEC, INMOS
J15	4361-45	NEC, INMOS
J16	4361-45	NEC, INMOS
J17	74F373	MOT, FAIR, VALVO
J18	74F373	MOT, FAIR, VALVO
J19	74F373	MOT, FAIR, VALVO
J20	74F373	MOT, FAIR, VALVO

Appendix D

BB Rev. 1
PRN
Date 11/11/86

Component Part List SYS68K/ISCSI-1

ICs

Location	Type	Manufacturer
J21	74F257A	MOT, FAIR, VALVO
J22	74F257A	MOT, FAIR, VALVO
J23	74F257A	MOT, FAIR, VALVO
J24	74F257A	MOT, FAIR, VALVO
J25	74F257A	MOT, FAIR, VALVO
J26	74ALS645-1	TI
J27	74ALS645-1	TI
J28	74F646	MOT, FAIR, VALVO
J29	74F646	MOT, FAIR, VALVO
J30	74F646	MOT, FAIR, VALVO
J31	74F646	MOT, FAIR, VALVO
J32	68010Y10	MOT
J33	PAL16L8B	FORCE
J34	27128-150	NEC, SEEQ
J35	27128-150	NEC, SEEQ
J36	74ALS113	TI
J37	74ALS113	TI
J38	PAL16L8B	FORCE
J39	68153	MOT
J40	74ALS641-1	TI

A p p e n d i x D

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

ICs

Location	Type	Manufacturer
J41	PAL16L8B	FORCE
J42	DDU 50A-10500	ESAN
J43	PAL20L8B	FORCE
J44	PAL20L8B	FORCE
J45	PAL20L8B	FORCE
J46	PAL20L8B	FORCE
J47	74LS682	TI
J48	74ALS641-1	TI
J49	PAL16L8B	FORCE
J50	PAL20L8B	FORCE
J51	74ALS244	TI
J52	74ALS373	TI
J53	74ALS373	TI
J54	PAL20L10	FORCE
J55	68450Y10	HITACHI
J56	555	TI
J57	74F04	MOT, FAIR, VALVO
J58	68230P8	MOT, THOMSON
J59	74LS423	MOT, TI
J60	74F74	MOT, FAIR, VALVO

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

ICs

Location	Type	Manufacturer
J61	74LS164	TI, MOT
J62	PAL16L8B	FORCE
J63	74LS393	TI
J64	74ALS645-1	TI
J65	74ALS645-1	TI
J66	5386S	NCR
J67	8310	NCR
J68	PAL20L8B	FORCE
J69	74ALS645-1	MOT, TI
J70	DDU-7J-200	TELEMETER
J71	WD1772	WESTERN DIGITAL
J72	74LS642-1	TI
J74	74AS652	TI
J75	PAL20L8B	FORCE
J76	PAL20L8B	FORCE
J77	MDU-4-20	TELEMETER
J78	DDU-7J-200	TELEMETER
J79	74F08	MOT, FAIR, VALVO
J80	74F74	MOT, FAIR, VALVO
J81	PAL20L8B	FORCE
J106	74AS1034	MOT, FAIR, VALVO

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

RESISTOR RGU

Location	Type	Manufacturer
R1	1 KOHM	VARIOUS
R2	22 KOHM	VARIOUS
R3	220 KOHM	VARIOUS
R4	47 OHM	VARIOUS
R9	100 KOHM	VARIOUS
R10	3,3 KOHM	VARIOUS
R15	1 KOHM	VARIOUS
R16	1 KOHM	VARIOUS
R17	1 KOHM	VARIOUS
R19	330 OHM	VARIOUS
R20	330 OHM	VARIOUS
R21	47 OHM	VARIOUS
R26	1 KOHM	VARIOUS
R30	3,3 KOHM	VARIOUS
R31	4,7 KOHM	VARIOUS
R32	3,3 KOHM	VARIOUS
R33	4,7 KOHM	VARIOUS

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

RESISTOR NETWORKS

Location	Type	Manufacturer
N1	9*3,3 KOHM	VARIOUS
N2	9*3,3 KOHM	VARIOUS
N3	9*3,3 KOHM	VARIOUS
N4	9*3,3 KOHM	VARIOUS
N5	9*3,3 KOHM	VARIOUS
N6	9*3,3 KOHM	VARIOUS
N7	9*3,3 KOHM	VARIOUS
N8	9*3,3 KOHM	VARIOUS
N9	9*3,3 KOHM	VARIOUS
N10	9*3,3 KOHM	VARIOUS
N11	9*3,3 KOHM	VARIOUS
N12	9*3,3 KOHM	VARIOUS
N13	9*3,3 KOHM	VARIOUS
N14	9*3,3 KOHM	VARIOUS
N15	9*3,3 KOHM	VARIOUS
N17	9*3,3 KOHM	VARIOUS
N18	9*3,3 KOHM	VARIOUS
N19	8*220/330 OHM	VARIOUS
N20	8*220/330 OHM	VARIOUS

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

RESISTORS NETWORKS

Location	Type	Manufacturer
N21	9*330 OHM	VARIOUS
N22	9*330 OHM	VARIOUS
N23	8*220/330 OHM	VARIOUS
N24	5x27 OHM 10 PIN	VARIOUS
N25	5x27 OHM 10 PIN	VARIOUS
N26	5x27 OHM 10 PIN	VARIOUS
N27	5x27 OHM 10 PIN	VARIOUS
N28	9*3,3 KOHM	VARIOUS
N29	9*4,7 KOHM	VARIOUS
N30	9*3,3 KOHM	VARIOUS
N31	9*4,7 KOHM	VARIOUS
N34	9*3,3 KOHM	VARIOUS

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

CAPACITORS

Location	Type	Manufacturer
C0	ELKO 100uF/16V RM 5,08	VARIOUS
C1	10uF 25V TANTAL RM 2,5	VARIOUS
C2	10nF RM 2,5	VARIOUS
C3	1,5nF RM 2,5	VARIOUS
C4	0,047 uF RM 2,5	VARIOUS
C5	100nF RM 2,5	VARIOUS
C10-C99	100nF RM 2,5	VARIOUS

CRYSTALS

Location	Type	Manufacturer
Q1	20MHz QUOM	SE, JAUCH
Q2	16MHz QUOM	SE, JAUCH

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

LEDs

Location	Type	Manufacturer
LD1	550-2206	DIALIGHT
LD2	550-2406	DIALIGHT
LD3	550-3006	DIALIGHT
LD4	550-2306	DIALIGHT
LD5	550-2306	DIALIGHT
LD6	550-2306	DIALIGHT
LD7	550-2306	DIALIGHT
LD8	550-2306	DIALIGHT
LD9	550-2306	DIALIGHT

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

MECHANICAL PARTS

Location	Type	Manufacturer
	PC-BOARD ISCSI-1	VARIOUS
	FRBL ISCSI-1	VARIOUS
SW1	ATE1D	KNITTER SWITCH
B18	EW2	VARIOUS
B19	TW9	VARIOUS
B21	DW18	VARIOUS
B22	DW6	VARIOUS
B23	EW2	VARIOUS
B24	EW2	VARIOUS
B27	EW3	VARIOUS
B28	EW3	VARIOUS
B29	EW3	VARIOUS
B39	EW2	VARIOUS
B41	EW2	VARIOUS

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

MECHANICAL PARTS

Location	Type	Manufacturer
J1-J16	24 PIN SLIM SOCKET	VARIOUS
J32	68 PIN PGA	VARIOUS
J33	20 PIN SOCKET	VARIOUS
J34, J35	56 PIN STACKED DIP SOCKET	VARIOUS
J38	20 PIN SOCKET	VARIOUS
J39	24 PIN SIL SOCKET (2*) 7,5MM HEIGHT	VARIOUS
J41	20 PIN SOCKET	VARIOUS
J43	24 PIN SLIM SOCKET	VARIOUS
J44	24 PIN SLIM SOCKET	VARIOUS
J45	24 PIN SLIM SOCKET	VARIOUS
J46	24 PIN SLIM SOCKET	VARIOUS
J49	20 PIN SOCKET	VARIOUS
J50	24 PIN SLIM SOCKET	VARIOUS
J54	24 PIN SLIM SOCKET	VARIOUS
J55	68 PIN PGA	VARIOUS
J58	24 PIN SIL SOCKET (2*) 7,5mm HEIGHT	VARIOUS

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

MECHANICAL PARTS

Location	Type	Manufacturer
J62	20 PIN SOCKET	VARIOUS
J66	24 PIN SIL SOCKET (2*) 7,5mm HEIGHT	VARIOUS
J67	24 PIN SIL SOCKET (2*) 7,5mm HEIGHT	VARIOUS
J68	24 PIN SLIM SOCKET	VARIOUS
J71	14 PIN SIL SOCKET (2*) 7,5mm HEIGHT	VARIOUS
J75	24 PIN SLIM SOCKET	VARIOUS
J76	24 PIN SLIM SOCKET	VARIOUS
J81	24 PIN SLIM SOCKET	VARIOUS
N19	10 PIN SIL	VARIOUS
N20	10 PIN SIL	VARIOUS
N23	10 PIN SIL	VARIOUS

BB Rev. 1
PRN 100
Date 11/11/86

Component Part List SYS68K/ISCSI-1

MECHANICAL PARTS

Location	Type	Manufacturer
P1	VG MALE CONNECTOR 96 PIN 90 DEGREES 2 x rivet	VARIOUS
P2	VG MALE CONNECTOR 96 PIN 90 DEGREES 2 x rivet	VARIOUS
	JUMPER-A	JUMPER FOR HEADERS (15PCS)
P3	DW34	VARIOUS

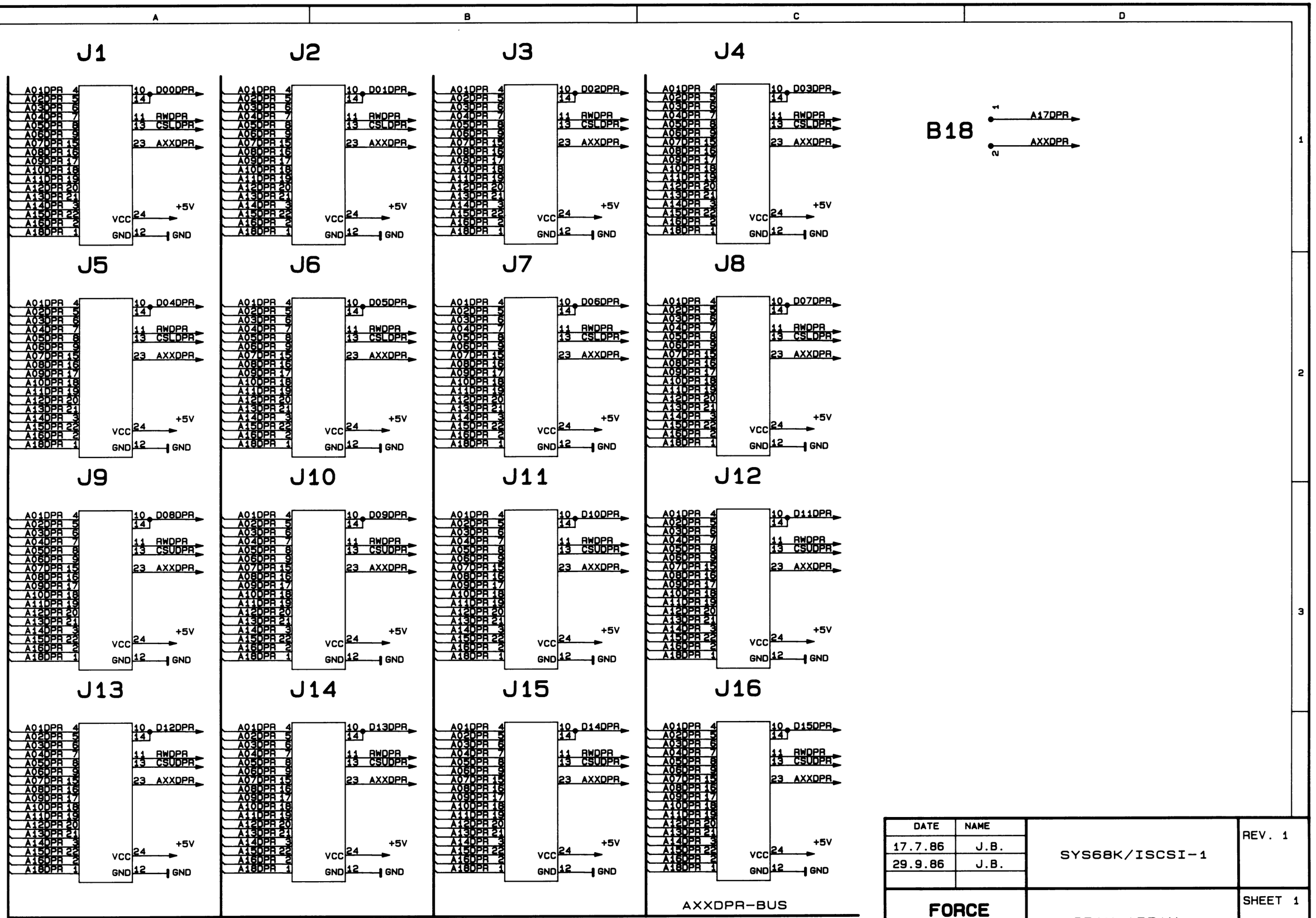
Appendix E

Description of the Jumperfields on the SYS68K/ISCSI-1

Description	Jumper field	Default Connections	Schematics Sheet	See Page
Reserved	B18	-	1	---
EPROM Size Select	B19	2 - 3 5 - 6 7 - 8	4	4-4
Board Base Address Selection	B21	2 - 17 4 - 15 5 - 14 6 - 13 7 - 12	7	6-4
Address Modifier Code Selection	B22	1 - 6 2 - 5	7	6-2
Test Jumper	B23	1 - 2	8	---
Terminator Power	B24	-	10	5-6
VME Reset	B27	1 - 2	7	---
Test Jumper	B28	1 - 2	12	---
Test Jumper	B29	1 - 2 or 2 - 3	11	---
Test Jumper	B39	1 - 2	5	---
Interrupt Configuration	B41	-	6	4-3

Appendix F

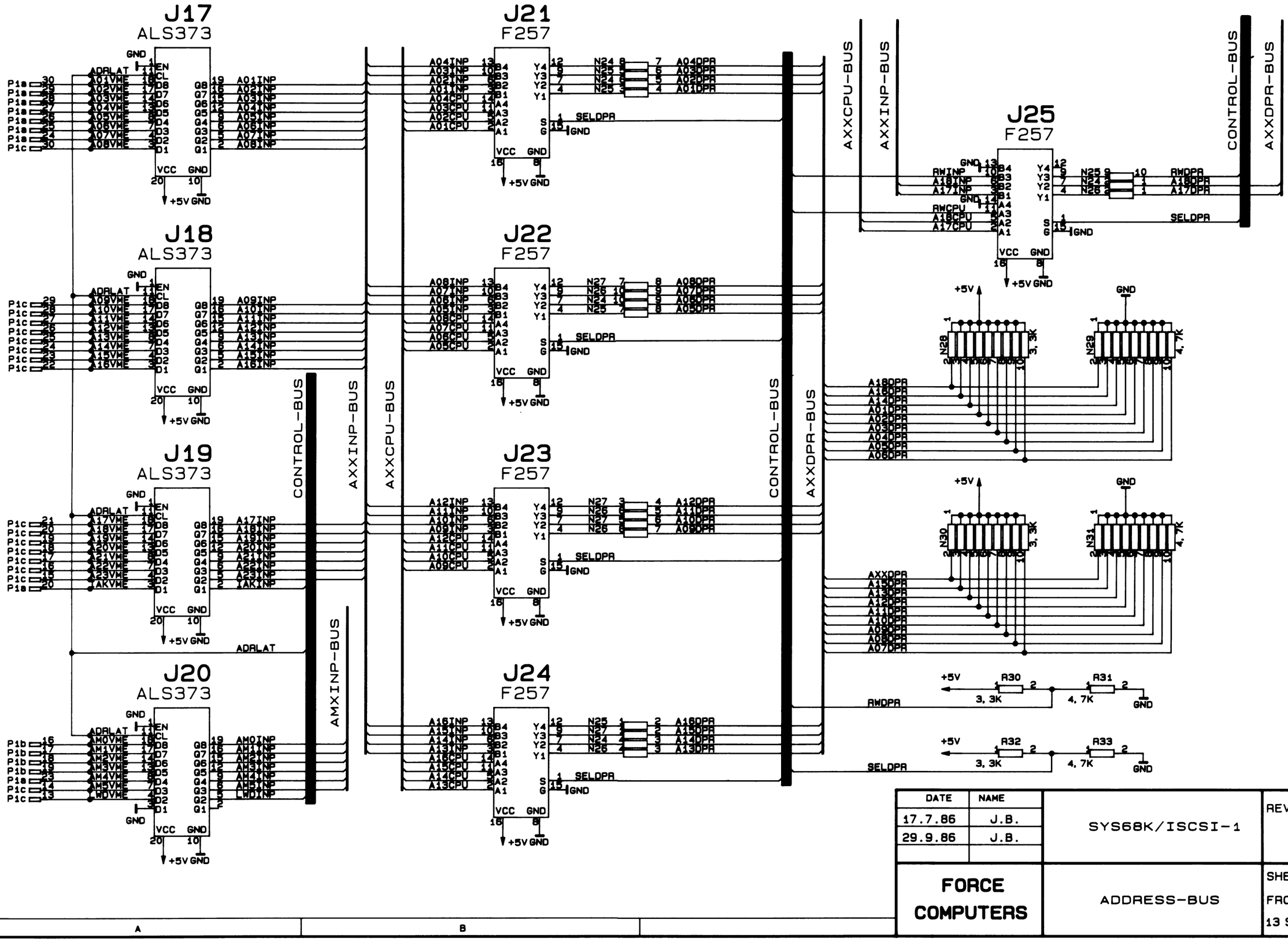
Circuit Schematics of the SYS68K/ISCSI-1



DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		SRAM ARRAY	SHEET 1 FROM 13 SHEETS

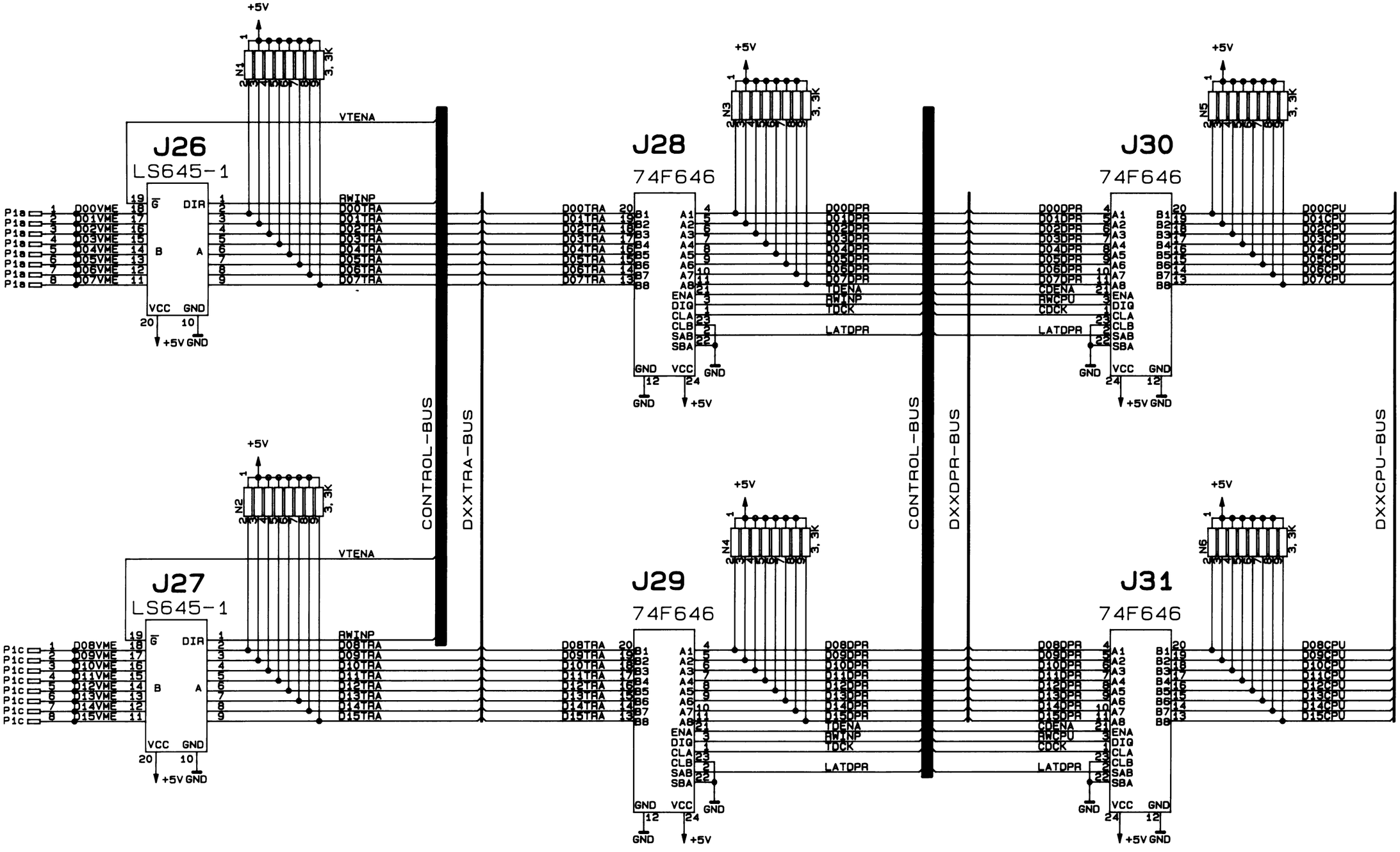
Date : 05.10.1986 Time : 09:39:11

068sh1



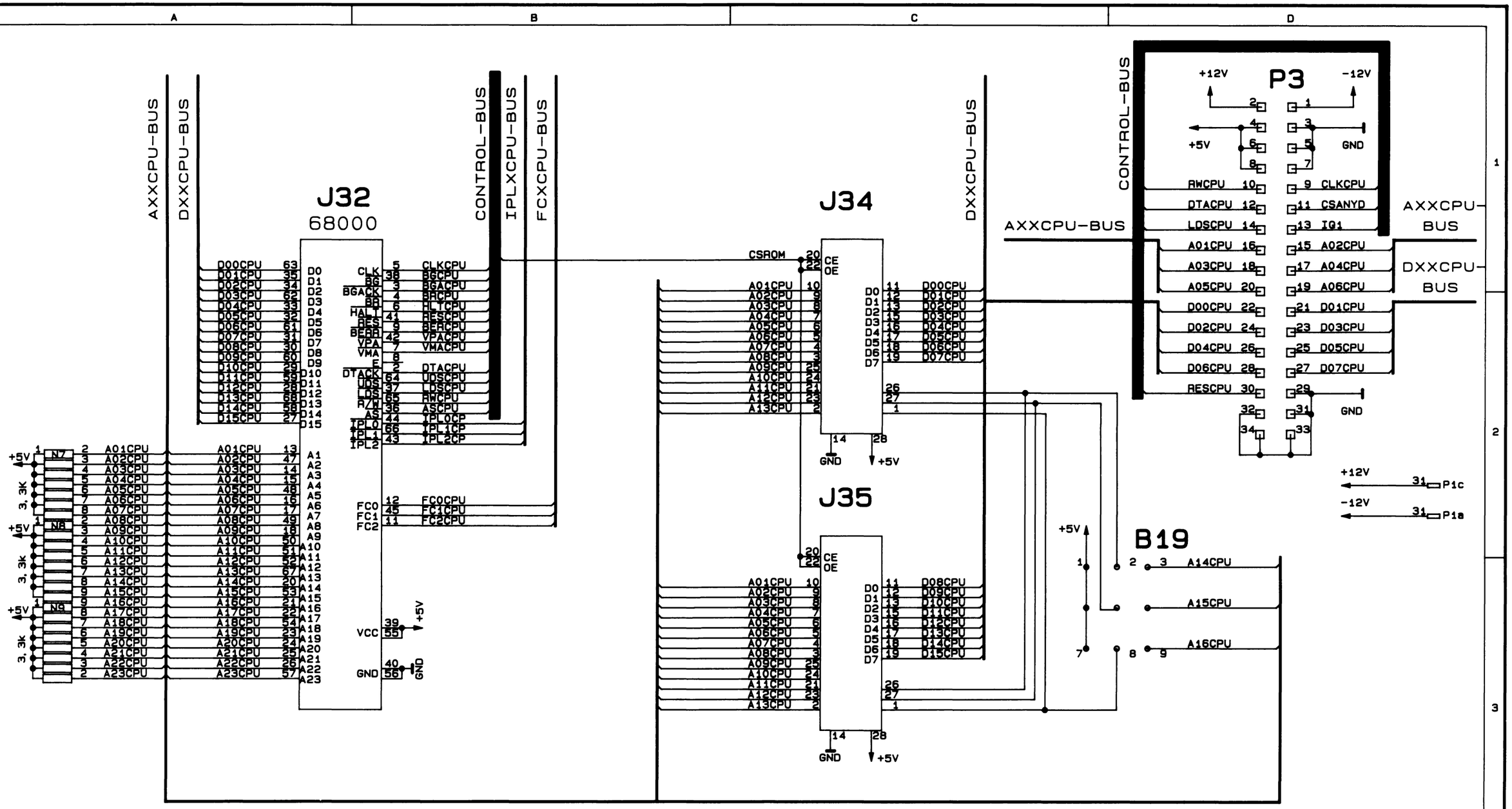
DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		ADDRESS-BUS	SHEET 2 FROM 13 SHEETS

Date : 05.10.1986 Time : 14:09:02



DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		DATA-BUS	SHEET 3 FROM 13 SHEETS

068sh3 Date : 05.10.1986 Time : 10:18:06



DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		LOCAL CPU	SHEET 4 FROM 13 SHEETS

/Date : 05.10.1986 Time : 10:31: 068sh4

INTXBIM-BUS

IA-BUS

INTXCK-BUS

CONTROL-BUS

J36

IA0
INT0CK
INT0BI
LS113

J36

IA1
INT1CK
INT1BI
LS113

J38

PAL16L8B

IA3
IA2
IA1
IA0
RESINP
IATINP
IARINP
RESCPU
VTENA
IAOVME 22 P1a

B39

+5V
N18
3, 3K
LDSINP
FAIL

J37

IA2
INT2CK
INT2BI
LS113

J37

IA3
INT3CK
+5V
R1
1k
INT3BI
LS113

J39

68153

+5V
N11
3, 3K
CLKCPU
CSBIM
RWINP
INT0BI
INT1BI
INT2BI
INT3BI
GND
+5V
ALOBIM
AL1BIM
AEBIM
IAKBIM
IATBIM
IAOBIM
DTABIM
INTAL0
INTAL1
INTAE
IACK
IACKIN
IACKOU
DTACK
IRQ7
IRQ6
IRQ5
IRQ4
IRQ3
IRQ2
IRQ1
D0
D1
D2
D3
D4
D5
D6
D7
A01INP
A02INP
A03INP

+5V
3.3k
N10
IRQ7BI
IRQ6BI
IRQ5BI
IRQ4BI
IRQ3BI
IRQ2BI
IRQ1BI
FAIL

J40

LS641-1

GND
DIR
G
A
B
GND
VCC
GND
+5V
IRQ7VM
IRQ6VM
IRQ5VM
IRQ4VM
IRQ3VM
IRQ2VM
IRQ1VM
FAILVM
P1b
P1c

D00TRA
D01TRA
D02TRA
D03TRA
D04TRA
D05TRA
D06TRA
D07TRA

DXXTRA-BUS

AXXINP-BUS

DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		VME-BUS INTERRUPTER	SHEET 5 FROM 13 SHEETS

AXXCPU-BUS

J43
PAL20L8B

A23CPU 1
A22CPU 2
A21CPU 3
A20CPU 4
A19CPU 5
A18CPU 6
ASCPU 7
BOOT 8
RESCPU 9
UDSCPU 10
LDSCPU 11
RWCPU 13
IAKDET 14
23

22 CSROM
21 CSDPR
20 CSANYD
19 CSPIT
18
17 SCSI
16 CSDMA
15 CSFLOP

12 GND 24 +5V

J81
PAL20L8B

UDSCPU 1
LDSCPU 2
RWCPU 3
BGACPU 4
CLKCPU 5
CKBDEL 6
7
8
9
10
11
13
14
23

22
21
20
19
18
17
16
15

12 GND 24 +5V

J44
PAL20L8B

UDSCPU 1
LDSCPU 2
RWCPU 3
RWINP 4
CSDPR 5
BUSDPR 6
CLKCPU 7
CK1DEL 8
CK2DEL 9
CK3DEL 10
CK4DEL 11
13
CK6DEL 14
DRVDDU 23

22
21
20
19
18
17
16
15

CK5DEL
N27 9 10 SELDPR
CDENA
TDENA

12 GND 24 +5V

J45
PAL20L8B

UDSCPU 1
LDSCPU 2
RWCPU 3
RWINP 4
CDENA 5
TDENA 6
7
8
9
CKHDEL 10
CKZDEL 11
UDSTNP 13
LDSTNP 14
CKXDEL 23

22 CSUDPR
21 CDCK
20 TDCK
19 DTADPR
18
17
16
15 CSLDPR

12 GND 24 +5V

CONTROL-BUS

CONTROL-BUS

J42
DDU7

DRVDDU 1
In Tap1 13
Tap2 3
Tap3 12
Tap4 4
Tap5 11
Tap6 5
Tap7 10
Tap8 6
Tap9 9
Tap10 8
GND 7 VCC 14
GND +5V

B20

1 22 CK1DEL
CK2DEL
CK3DEL
CK4DEL
CKXDEL
CK5DEL
CK6DEL
CK7DEL
CKZDEL
10 13 CKYDEL
11 12 CKHDEL
CKBDEL

J106

CLKCPU 13 12
74AS1034

J41
PAL16L8B

2
3
LDSCPU 4
IAKDET 5
A01CPU 6
A02CPU 7
A03CPU 8
9
11

19
18 IAKPIT
17 IAKTIM
16
15 AK3
14
13
12 VPACPU
1

GND VCC
10 GND 20 +5V

B41

1 2
N10 10 1 3,3K
+5V
GND

J33
PAL16L8B

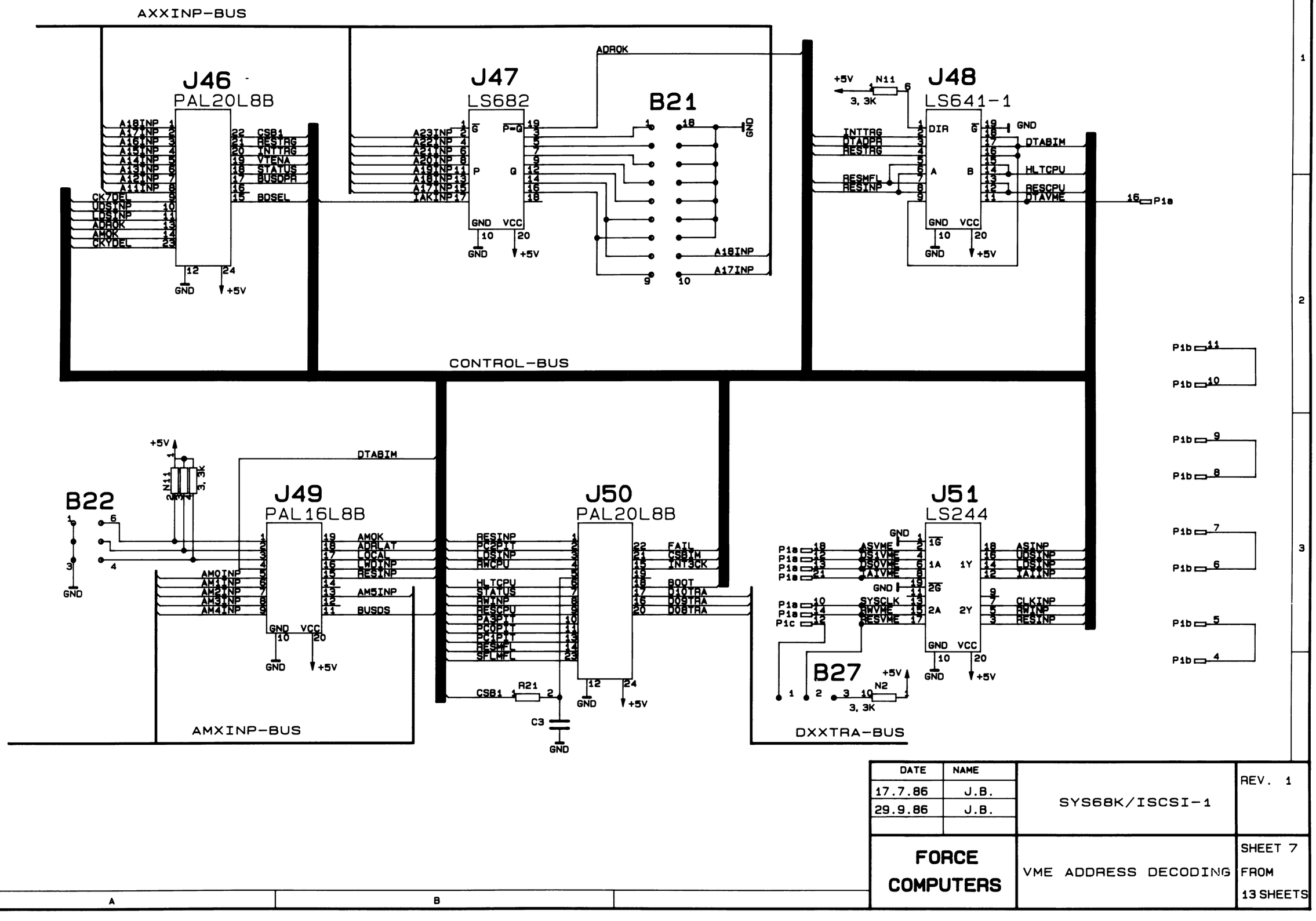
IRQTIM 1
IRQPIT 2
IO1 3
IO2 8
IO3 9
IO4 11
FC0CPU 4
FC1CPU 5
FC2CPU 6

12 IPL0CP
13 IPL1CP
14 IPL2CP
15 IAKDET
16
17
18
7

GND VCC
10 GND 20 +5V

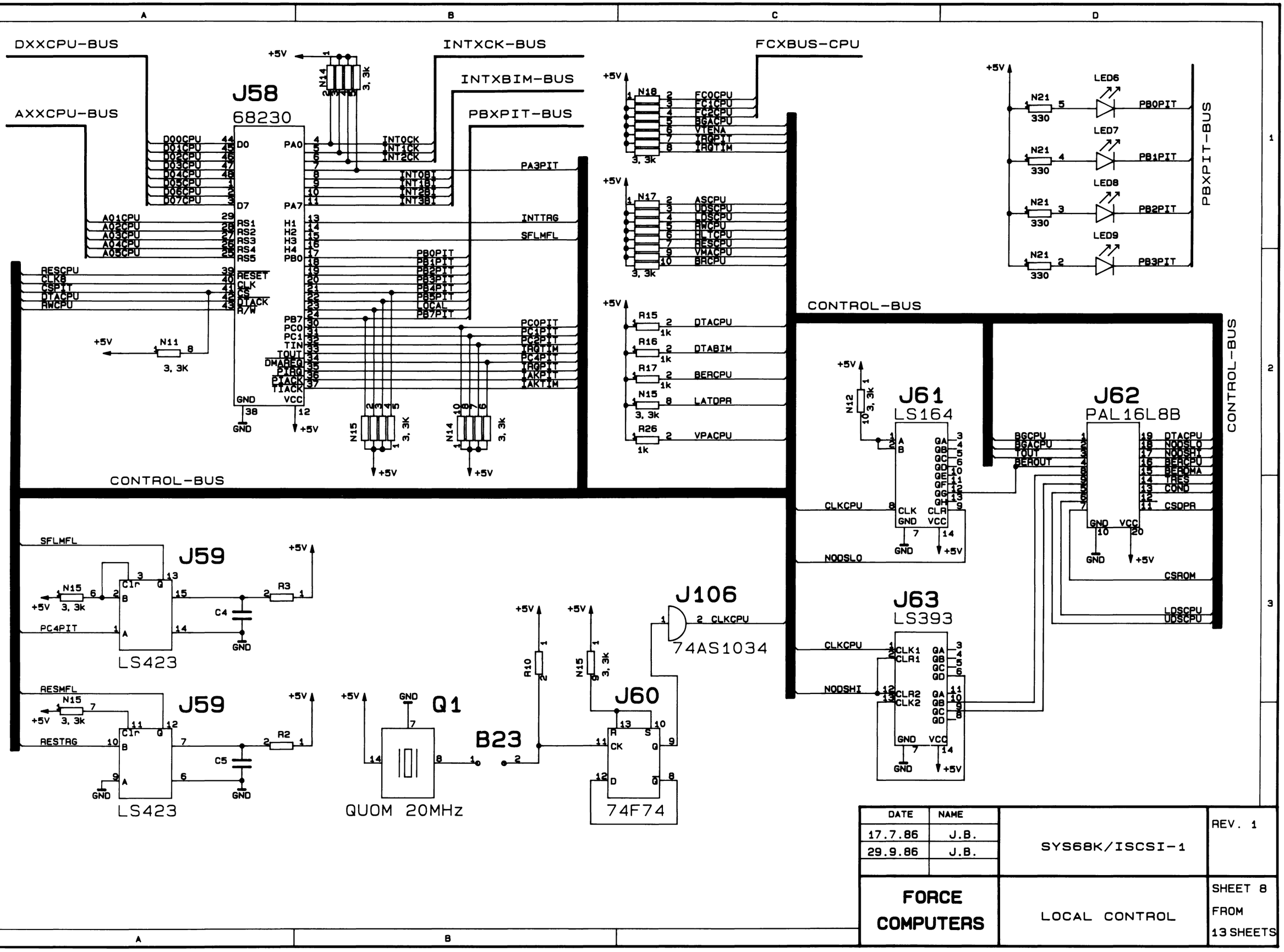
ASCPU

DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		DATA-BUS	SHEET 6 FROM 13 SHEETS

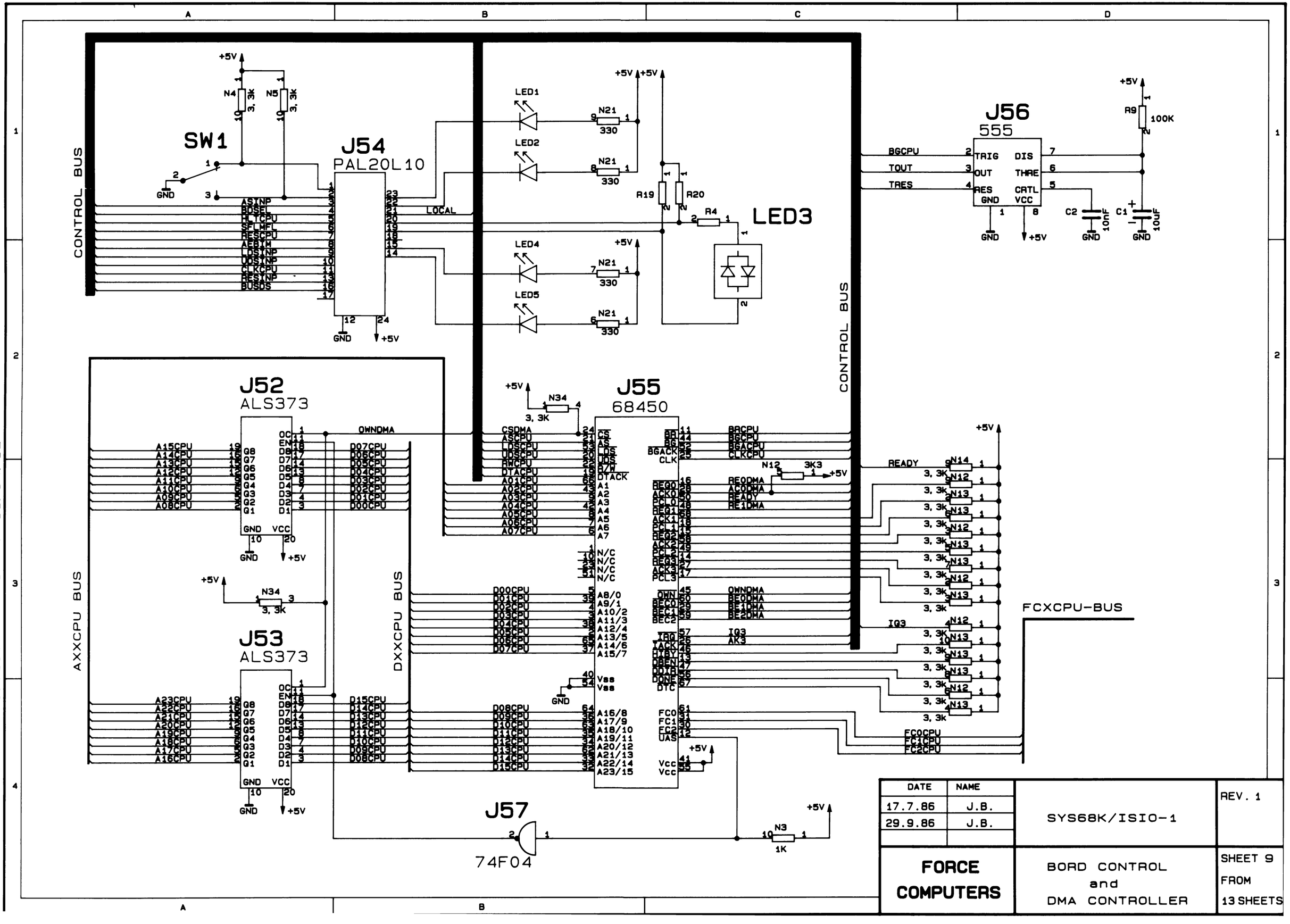


DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		VME ADDRESS DECODING	SHEET 7 FROM 13 SHEETS

068sh7 Date : 05.10.1986 Time : 11:40:32



068sh8 Date : 05.10.1986 Time : 13: 42: 26



DATE	NAME		REV. 1
17.7.86	J.B.	SYS68K/ISIO-1	
29.9.86	J.B.		
FORCE COMPUTERS		BORD CONTROL and DMA CONTROLLER	SHEET 9 FROM 13 SHEETS

Date : 05.10.1986 Time : 11:54:12

68sh9

CONTROL-BUS

J57
74F04
RESCPU
DREQ+
REQDMA
IRQ2

J64
74LS645
DIRSC
D00CPU
D01CPU
D02CPU
D03CPU
D04CPU
D05CPU
D06CPU
D07CPU
GLDS
D00SC
D01SC
D02SC
D03SC
D04SC
D05SC
D06SC
D07SC

J65
74LS645
DIRSC
D08CPU
D09CPU
D10CPU
D11CPU
D12CPU
D13CPU
D14CPU
D15CPU
GUDS
D00SC
D01SC
D02SC
D03SC
D04SC
D05SC
D06SC
D07SC

N34
3k3
+5V
AXXCPU-BUS
DXXCPU-BUS

J66
5386S
CLKCPU
DREQ+
SCDACK
INT+
IDO
ID1
ID2
CSSCSI
RDSCSI
WRSCSI
D00SC
D01SC
D02SC
D03SC
D04SC
D05SC
D06SC
D07SC
A01CPU
A02CPU
A03CPU
A04CPU
P2c 29
P2c 28
P2c 27
P2c 31
P2c 30
P2a 29
P2a 30
P2a 31
P2a 32

J67
8310
P2c 32
PB4PIT
PB5PIT

N19
220
330
GND
+5V

N20
N23
220
330

B24
P2a

DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		SCSI-CONTROLER	SHEET 10 FROM 13 SHEETS

Date : 05.10.1986 Time : 12:33:53
068sh10

CONTROL-BUS

J75
PAL20L8B

ASCPU 1
AS20 2
LDSCPU 3
UDSCPU 4
SCSI 5
ACODMA 6
RWCPU 7
DDREQ 8
RD20 9
RD40 10
11
13
14
23

22 GLDS
21
20 READYA
19 RDSCSI
18 SCDACA
17 DTACPU
16 CSSCSA
15 DIRSC

12 GND
24 +5V

J76
PAL20L8B

ASCPU 1
AS20 2
LDSCPU 3
UDSCPU 4
SCSI 5
ACODMA 6
RWCPU 7
DDREQ 8
WR40 9
WR80 10
WR100 11
WR140 13
WR160 14
23

22
21
20 READYB
19 WRSCSI
18 SCDACB
17 DTACPU
16 CSSCSB
15 GUDS

12 GND
24 +5V

J79
74F08

CSSCSA 1
CSSCSB 2
3 CSSCSI

SCDACA 4
SCDACB 5
6 SCDACK

READYA 9
READYB 10
8 READY

B29
74F08

11
13
12 RESCPU
2
CSSCSI 3

J80 J79

+5V
N17
3k3 1 8

1 Clr PR 4
2 D Q 5 DDREQ
GND
DREQ+ 3 Clk Q 6
74F74

J80

13 Clr PR 10
12 D Q 9
GND
SCDACK 11 Clk Q 8
74F74

J77
MDU-4-20

RDSCSI 1 IN1 OUT1 12 RD20
3 IN2 OUT2 10 RD40
ASCPU 4 IN3 OUT3 9 AS20
5 IN4 OUT4 8
GND 7 VCC 14
GND +5V

B26

WR40 1
WR80
WR100
WR140
WR160
8

16
9

J78
DDU-7J-200

20 Tap1 In 13 1 WRSCSI
40 Tap2 3
60 Tap3 12
80 Tap4 4
100 Tap5 11
120 Tap6 5
140 Tap7 10
150 Tap8 6
180 Tap9 9
200 Tap10 8
VCC 14 GND 7
+5V GND

DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		NCR-CONTROLL-DAL	SHEET 11 FROM 13 SHEETS

Date : 05.10.1986 Time : 13:19:
68sh11

CONTROL-BUS

J68
PAL20L8B
+5V N34 1 6 CSWD
3k3
RESCPU 2
ASCPU 4
LDSCPU 5
RWCPU 8
HLTCPU 10
A03CPU 7
CSWDDE 9
BERRDMA 1
OWNDMA 3
CSFLOP 6
11 13 14 23
20 DTACPU
15 BE0DMA
16 BE1DMA
17 BE2DMA
18 CSWD
21 REGCAB
22 REGGBA
19 BUFEN
AXXCPU-BUS
12 GND
24 +5V

J70
DDU-7J-200
1 In
Tap1 13
Tap2 3
Tap3 12
Tap4 4
Tap5 11
Tap6 5
Tap7 10
Tap8 6
Tap9 9
Tap10 8
CSWDDE
GND VCC
7 GND
14 +5V

J57
74F04
RE1DMA 10
11

J71
WD1772
27 DRQ+ R/W
RWCPU 2
CLK8 18 CLK
CSWD 1 CS
RESCPU 13 MR
FLD0 5 D0
FLD1 6 D1
FLD2 7 D2
FLD3 8 D3
FLD4 9 D4
FLD5 10 D5
FLD6 11 D6
FLD7 12 D7
A0 3
A1 4
A01CPU
A02CPU
GND 14
VCC 15
GND +5V
AXXCPU-BUS
RD 19 FLREAD
TO 23 FLTR00
IP 24 FLINDE
WP 25 FLWRPR
ST 16
DI 17
WD 22
WG 21
MO 20
DDE 26 PB7PIT
IN+ 28

+5V
N22 330
1 2 3 4 5 6 7 8 9

B25
1 2 3
GND

J57
74F04
13 12 IQ4

FLXX-BUS

J72
LS642-1
GND GND
19 G DIR 1
18 2
17 3
16 4
15 5
14 B A 6
13 7
12 8
11 9
VCC GND
20 10
+5V GND
FLREAD 15 P2c
FLTR00 13 P2c
FLINDE 4 P2c
FLWRPR 14 P2c
FLSTEP 10 P2c
FLDIR 5 P2c
FLWD 11 P2c
FLWG 12 P2c
FLMO 8 P2c

J60
+5V N15 10 1
3k3
J106
1 4
Clr PR
2 D Q 5 3 4 CLK8
3 Clk Q 6
74AS1034
74F74
+5V N34 1 8
3k3
CLKINP 3
2 1
B28
Q2
8
+5V 14 7 GND
16MHz

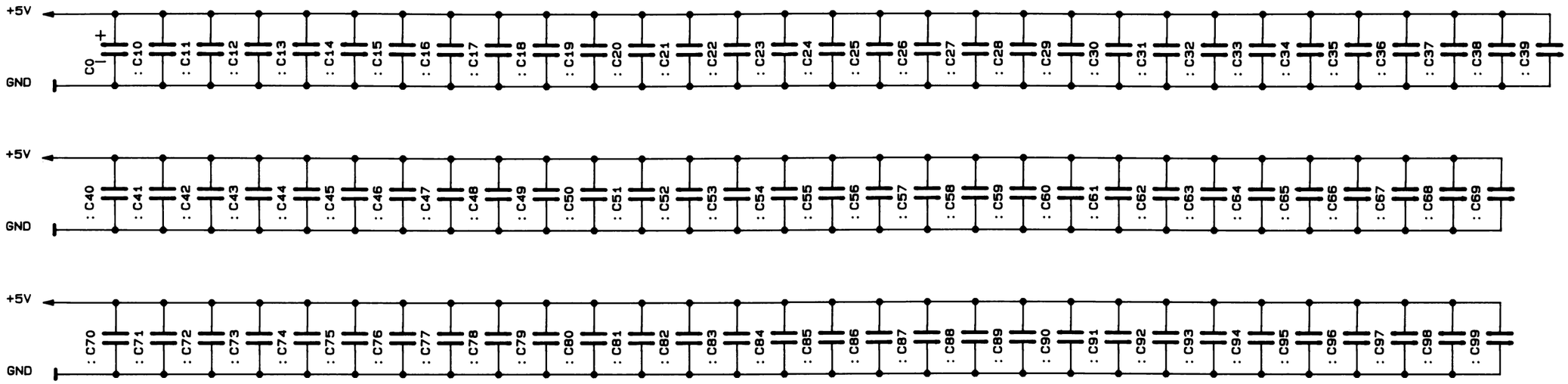
J69
LS245
BUFEN 19 G DIR 1 RWCPU
D00CPU 18 2 FLD0
D01CPU 17 3 FLD1
D02CPU 16 4 FLD2
D03CPU 15 5 FLD3
D04CPU 14 B A 6 FLD4
D05CPU 13 7 FLD5
D06CPU 12 8 FLD6
D07CPU 11 9 FLD7
VCC GND
20 10
+5V GND

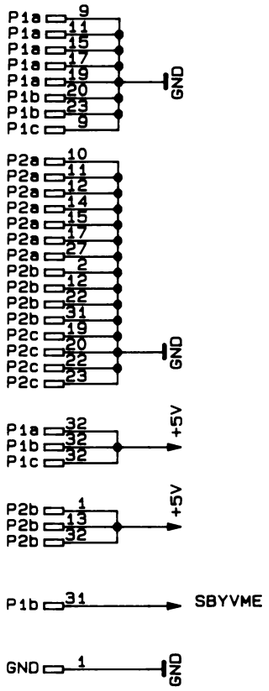
J74
74AS652
D07CPU 11 A8
D06CPU 10 A7
D05CPU 9 A6
D04CPU 8 A5
D03CPU 7 A4
D02CPU 6 A3
D01CPU 5 A2
D00CPU 4 A1
REGGBA 21 GBA
REGCAB 3 GAB
1 CAB
23 CBA
2 SAB
22 SBA
B8 13 ID2
B7 14 ID1
B6 15 ID0
B5 16 FLDS3
B4 17 FLDS2
B3 18 FLDS1
B2 19 FLDS0
B1 20 FLSISE
GND
VCC GND
24 12
+5V GND
N11 10
3k3
N34 7
3k3
+5V
FLDS3 7 P2c
FLDS2 6 P2c
FLDS1 5 P2c
FLDS0 3 P2c
FLSISE 16 P2c

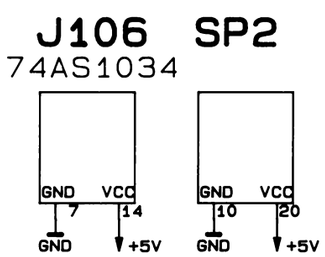
DXXCPU-BUS

DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		SCSI-CONTROLER	SHEET 12 FROM 13 SHEETS

068sh12 Date : 05.10.1986 Time : 13:56:







DATE	NAME		
17.7.86	J.B.	SYS68K/ISCSI-1	REV. 1
29.9.86	J.B.		
FORCE COMPUTERS		POWER	SHEET 13 FROM 13 SHEETS

68sh13 Date : 05.10.1986 Time : 15:25:..

Appendix G

Connector PIN Assignments of the SYS68K/ISCSI-1

Appendix G

Connector Pin Assignments P1

Pin Number	Row A Signal Mnemonic	Row B Signal Mnemonic	Row C Signal Mnemonic
1	D00		D08
2	D01		D09
3	D02		D10
4	D03	BG0IN*	D11
5	D04	BG0OUT*	D12
6	D05	BG1IN*	D13
7	D06	BG1OUT*	D14
8	D07	BG2IN*	D15
9	GND	BG2OUT*	GND
10	SYSCLK	BG3IN*	SYSFAIL*
11	GND	BG3OUT*	
12	DS1*		SYSRESET*
13	DS0*		LWORD*
14	WRITE*		AM5
15	GND		A23
16	DTACK*	AM0	A22
17	GND	AM1	A21
18	AS*	AM2	A20
19	GND	AM3	A19
20	IACK*	GND	A18
21	IACKIN*		A17
22	IACKOUT*		A16
23	AM4	GND	A15
24	A07	IRQ7*	A14
25	A06	IRQ6*	A13
26	A05	IRQ5*	A12
27	A04	IRQ4*	A11
28	A03	IRQ3*	A10
29	A02	IRQ2*	A09
30	A01	IRQ1*	A08
31	-12V		+12V
32	+5V	+5V	+ 5V

Appendix G

Connector Pin Assignments P2

Pin Number	Row A Signal Mnemonic	Row B Signal Mnemonic	Row C Signal Mnemonic
1	DB Ø	VCC	N.C.
2	DB 1	GND	N.C.
3	DB 2	N.C.	Drive Select Ø
4	DB 3	N.C.	Index
5	DB 4	N.C.	Drive Select 1
6	DB 5	N.C.	Drive Select 2
7	DB 6	N.C.	Drive Select 3
8	DB 7	N.C.	Motor On
9	DB P	N.C.	Direction In
1Ø	GND	N.C.	Step
11	GND	N.C.	Write Data
12	GND	GND	Write Gate
13	TERMPWR	VCC	Track ØØØ
14	GND	N.C.	Write Protect
15	GND	N.C.	Read Data
16	ATN	N.C.	Side Select
17	GND	N.C.	N.C.
18	BSY	N.C.	N.C.
19	ACK	N.C.	GND
2Ø	RST	N.C.	GND
21	MSG	N.C.	N.C.
22	SEL	GND	GND
23	C/D	N.C.	GND
24	REQ	N.C.	N.C.
25	I/O	N.C.	N.C.
26	N.C.	N.C.	N.C.
27	GND	N.C.	RESERVED
28	N.C.	N.C.	RESERVED
29	RESERVED	N.C.	RESERVED
3Ø	RESERVED	N.C.	RESERVED
31	RESERVED	GND	RESERVED
32	RESERVED	VCC	RESERVED

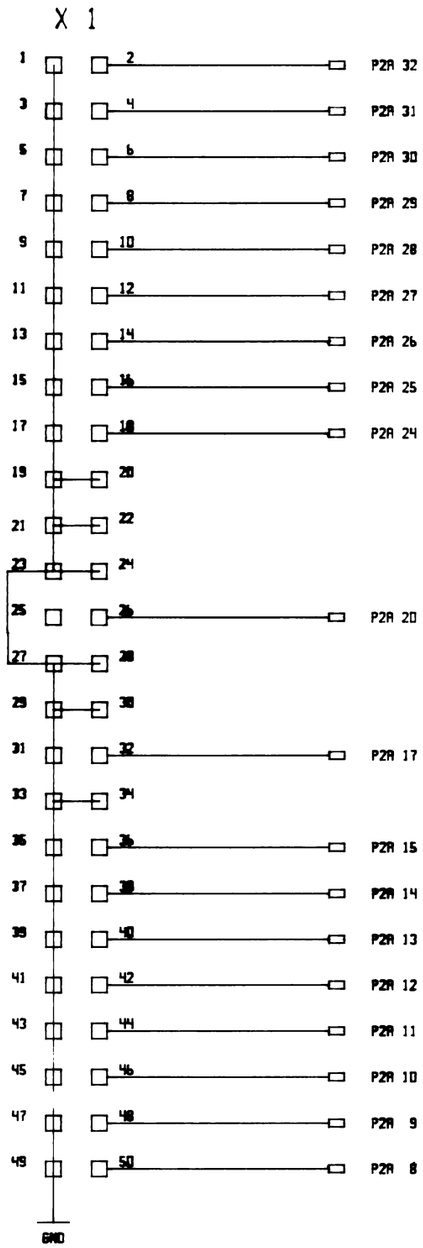
Appendix H

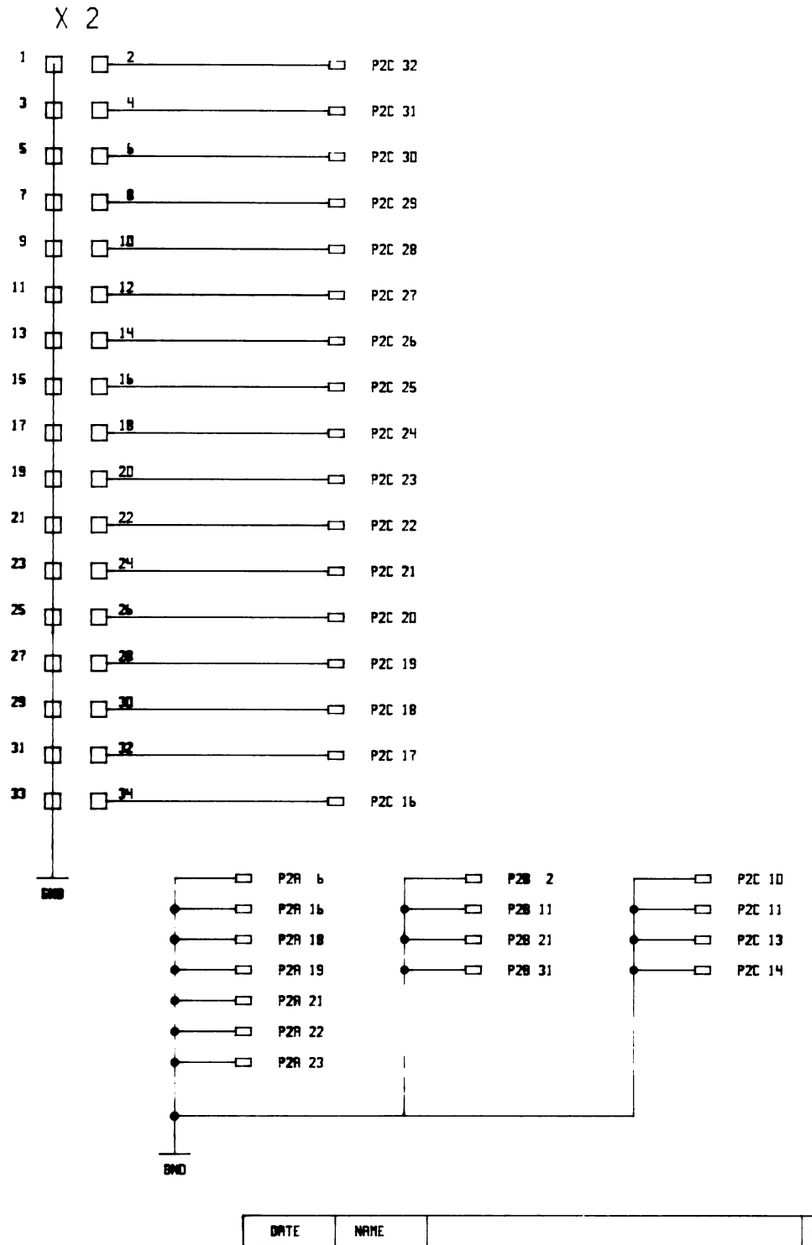
Component Part List of the SYS68K/ISCSI-1BPS

Location	Type	Manufacturer
	PC BOARD ISCSI-1BPS	VARIOUS
P2	VGFE MALE CONNECTOR 96 PIN 90 DEGREES 2 x SCREW 2,5/10 2 x SCREW 2,5/10	VARIOUS
X1	2 ROW 50 PIN	VARIOUS
X2	2 ROW 34 PIN MALE CONNECTOR SHORT ARMS 90 DEGREES 2 x SCREW 2,5/10 2 x MOTHER 2,5	VARIOUS

Appendix I

Circuit Schematics of the SYS68K/ISCSI-1BPS





DATE	NAME		REV. 0
25.7.86	J.B.	SYS68K-ISCSI-1BPS	
FORCE COMPUTERS			SHEET 1 FROM 1 SHEETS

Appendix J

Connector PIN Assignments of the SYS68K/ISCSI-1BPS

The ISCSI-1BPS P2 Pin Assignment

Pin Number	Row A Signal Mnemonic	Row B Signal Mnemonic	Row C Signal Mnemonic
32	DB Ø	VCC	N.C.
31	DB 1	GND	N.C.
3Ø	DB 2	N.C.	Drive Select Ø
29	DB 3	N.C.	Index
28	DB 4	N.C.	Drive Select 1
27	DB 5	N.C.	Drive Select 2
26	DB 6	N.C.	Drive Select 3
25	DB 7	N.C.	Motor On
24	DB P	N.C.	Direction In
23	GND	N.C.	Step
22	GND	N.C.	Write Data
21	GND	GND	Write Gate
2Ø	TERMPWR	VCC	Track ØØØ
19	GND	N.C.	Write Protect
18	GND	N.C.	Read Data
17	ATN	N.C.	Side Select
16	GND	N.C.	N.C.
15	BSY	N.C.	N.C.
14	ACK	N.C.	GND
13	RST	N.C.	GND
12	MSG	N.C.	N.C.
11	SEL	GND	GND
1Ø	C/D	N.C.	GND
9	REQ	N.C.	N.C.
8	I/O	N.C.	N.C.
7	N.C.	N.C.	N.C.
6	GND	N.C.	RESERVED
5	N.C.	N.C.	RESERVED
4	RESERVED	N.C.	RESERVED
3	RESERVED	N.C.	RESERVED
2	RESERVED	GND	RESERVED
1	RESERVED	VCC	RESERVED

Appendix J

The ISCSI-1BPS X1 Pin Assignment

Pin No.	Signal Mnemonic	Pin No.	Signal Mnemonic
2	DB Ø	1	GND
4	DB 1	3	GND
6	DB 2	5	GND
8	DB 3	7	GND
1Ø	DB 4	9	GND
12	DB 5	11	GND
14	DB 6	13	GND
16	DB 7	15	GND
18	DB P	17	GND
2Ø	GND	19	GND
22	GND	21	GND
24	GND	23	GND
26	TERMPWR	25	N.C.
28	GND	27	GND
3Ø	GND	29	GND
32	ATN	31	GND
34	GND	33	GND
36	BSY	35	GND
38	ACK	37	GND
4Ø	RST	39	GND
42	MSG	41	GND
44	SEL	43	GND
46	C/D	45	GND
48	REQ	47	GND
5Ø	I/O	49	GND

Appendix J

The ISCSI-1BPS X2 Pin Assignment

Pin No.	Signal Mnemonic	Pin No.	Signal Mnemonic
2	N.C.	1	GND
4	N.C.	3	GND
6	Drive Select Ø	5	GND
8	Index	7	GND
1Ø	Drive Select 1	9	GND
12	Drive Select 2	11	GND
14	Drive Select 3	13	GND
16	Motor On	15	GND
18	Direction In	17	GND
2Ø	Step	19	GND
22	Write Data	21	GND
24	Write Gate	23	GND
26	Track ØØØ	25	GND
28	Write Protect	27	GND
2Ø	Read Data	29	GND
32	Side Select	31	GND
34	N.C.	33	GND

Appendix K

Glossary of VMEbus Terms (P1Ø14)

A16
A type of module that provides or decodes an address on address line AØ1 through A15.

A24
A type of module that provides or decodes an address on address lines AØ1 through A23.

A32
A type of module that provides or decodes an address on address lines AØ1 through A31.

ARBITRATION
The process of assigning control of the DTB to a REQUESTER.

ADDRESS-ONLY CYCLE
A DTB cycle that consists of an address broadcast, but no data transfer. SLAVES do not acknowledge ADDRESS-ONLY cycles and MASTERS terminate the cycle without waiting for an acknowledgment.

ARBITER
A functional module that accepts bus requests from REQUESTER modules and grants control of the DTB to one REQUESTER at a time.

ARBITRATION BUS
One of the four buses provided by the P1Ø14 backplane This bus allows an ARBITER module and several REQUESTER modules to coordinate use of the DTB.

ARBITRATION CYCLE
An ARBITRATION CYCLE begins when the ARBITER senses a bus request. The ARBITER grants the bus to a REQUESTER, which signals that the DTB is busy. The REQUESTER terminates the cycle by taking away the bus busy signal which causes the ARBITER to sample the bus requests again.

BACKPLANE (P1Ø14)
A printed circuit (PC) board with 96 pin connectors and signal paths that bus the connector pins. Some P1Ø14 systems have a single PC board, called the J1 backplane. It provides the signal paths needed for basic operation. Other P1Ø14 systems also have an optional second PC board called a J2 backplane. It provides the additional 96 pin connectors and signal paths needed for wider data and address transfers. Still others have a single PC board that provides the signal conductors and connectors of both the J1 and J2 backplanes.

BACKPLANE INTERFACE LOGIC

Special interface logic that takes into account the characteristics of the backplane: its signal line impedance, propagation time, termination values, etc. The P1Ø14 specification prescribes certain rules for the design of this logic based on the maximum length of the backplane and its maximum number of board slots.

BLOCK READ CYCLE

A DTB is cycle used to transfer a block of 1 to 256 bytes from a SLAVE to a MASTER. This transfer is done using a string of 1, 2, or 4 byte data transfers. Once the block transfer is started, the MASTER does not release the DTB until all of the bytes have been transferred. It differs from a string of read cycles in that the MASTER broadcasts only one address and address modifier (at the beginning of the cycle). Then the SLAVE increments this address on each transfer so that the data for the next cycle is retrieved from the next higher location.

BLOCK WRITE CYCLE

A DTB cycle used to transfer a block of 1 to 256 bytes from a MASTER to a SLAVE. The block write cycle is very similar to the block read cycle. It uses a string of 1, 2, or 4 byte data transfers and the MASTER does not release the DTB until all of the bytes have been transferred. It differs from a string of write cycles in that the MASTER broadcasts only one address and address modifier (at the beginning of the cycle). Then the SLAVE increments this address on each transfer so that the next transfer is stored on the next higher location.

BOARD

A printed circuit (PC) board, its collection or electronic components, and either one or two 96 pin connectors that can be plugged into P1Ø14 backplane connectors.

BUS TIMER

A functional module that measures how long each data transfer takes on the DTB and terminates the DTB cycle if a transfer takes too long. If the MASTER tries to transfer data to or from a non-existent SLAVE location it might wait forever. The BUS TIMER prevents this by terminating the cycle.

DØ8(Ø)

A SLAVE that sends and receives data 8 bits at a time over DØØ-DØ7,

or

an INTERRUPT HANDLER that receives 8 bit STATUS/IDs over DØØ-DØ7,

or

an INTERRUPTER that sends 8 bit STATUS/IDs over DØØ-DØ7.

D08(E0)

A MASTER that sends or receives data 8 bits at a time over either D00-D07 or D08-D15,

or

A SLAVE that sends and receives data 8 bits at a time over either D00-D07 or D08-D15,

or

an INTERRUPT HANDLER that receives 8 bit STATUS/IDs over D00-D07,

or

an INTERRUPTER that sends 8 bit STATUS/IDs over D00-D07.

D16

A MASTER that sends and receives data 16 bits at a time over D00-D15,

or

A SLAVE that sends and receives data 16 bits at a time over D00-D15,

or

an INTERRUPT HANDLER that receives 16 bit STATUS/IDs over D00-D15,

or

an INTERRUPTER that sends 16 bit STATUS/IDs over D00-D15.

D32

A MASTER that sends and receives data 32 bits at a time over D00-D31,

or

A SLAVE that sends and receives data 32 bits at a time over D00-D31,

or

an INTERRUPT HANDLER that receives 32 bit STATUS/IDs over D00-D31,

or

an INTERRUPTER that sends 32 bit STATUS/IDs over D00-D31.

DAISY-CHAIN
A special type of P1Ø14 signal line that is used to propagate a signal level from board to board, starting with the first slot and ending with the last slot. There are four bus grant daisy-chains and one interrupt acknowledge daisy-chain on the P1Ø14.

DATA TRANSFER BUS
One of the four buses provided by the P1Ø14 backplane. The DATA TRANSFER BUS allows MASTERS to direct the transfer of binary data between themselves and SLAVES. (DATA TRANSFER BUS is often abbreviated to DTB).

DATA TRANSFER BUS CYCLE
A sequence of level transitions on the signal lines of the DTB that result in the transfer of an address or an address and data between a MASTER and a SLAVE. There are seven types of data transfer bus cycles.

DTB
An acronym for DATA TRANSFER BUS.

FUNCTIONAL MODULE
A collection of electronic circuitry that resides on one P1Ø14 board and works together to accomplish a task.

IACK DAISY-CHAIN DRIVER
A functional module which activates the interrupt acknowledge daisy-chain whenever an INTERRUPT HANDLER acknowledges an interrupt request. This daisy-chain ensures that only one INTERRUPTER will respond with its STATUS/ID when more than one has generated an interrupt request.

INTERRUPT ACKNOWLEDGE CYCLE
A DTB cycle, initiated by an INTERRUPT HANDLER that reads a "STATUS/ID" from an INTERRUPTER. An INTERRUPT HANDLER generates this cycle when it detects an interrupt request from an INTERRUPTER and it has control of the DTB.

INTERRUPT BUS
One of the four buses provided by the P1Ø14 backplane. The INTERRUPT BUS allows INTERRUPTER modules to send interrupt requests to INTERRUPT HANDLER modules.

INTERRUPTER
A functional module that generates an interrupt request on the INTERRUPT BUS and then provides STATUS/ID information when the INTERRUPT HANDLER requests it.

INTERRUPT HANDLER
A functional module that detects interrupt requests generated by INTERRUPTERS and responds to those requests by asking for STATUS/ID information.

LOCATION MONITOR

A functional module that monitors data transfers over the DTB in order to detect accesses to the locations it has been assigned to watch. When an access occurs to one of these assigned locations, the LOCATION MONITOR generates an on-board signal.

MASTER

A functional module that initiates DTB cycles in order to transfer data between itself and a SLAVE module.

OBO

A SLAVE that sends and receives data 8 bits at a time over DØØ-DØ7.

POWER MONITOR MODULE

A functional module that monitors the status of the primary power source to the P1Ø14 system and signals when that power has strayed outside the limits required for reliable system operation. Since most systems are powered by an AC source, the power monitor is typically designed to detect drop-out or brown-out conditions on AC lines.

READ CYCLE

A DTB cycle used to transfer 1, 2, or 4 bytes from a SLAVE to a MASTER. The cycle begins when the MASTER broadcasts and address and an address modifier. Each SLAVE captures this address and address modifier, and checks to see of it is to respond to the cycle. If so, it retrieves the data from its internal storage, places it on the data bus and acknowledges the transfer. Then the MASTER terminates the cycle.

READ-MODIFY-WRITE CYCLE

A DTB cycle that is used to both read from, and write to, a SLAVE location without permitting any other MASTER to access that location. This cycle is most useful in multiprocessing systems where certain memory locations are used to control access to certain systems resources. (For example, semaphore locations.)

REQUESTER

A functional module that resides on the same board as a MASTER or INTERRUPT HANDLER and requests use of the DTB whenever its MASTER or INTERRUPT HANDLER needs it.

SERIAL CLOCK DRIVER

A functional module that provides a periodic timing signal that synchronizes operation of the VMSbus. (Although the P1Ø14 specification defines a SERIAL CLOCK DRIVER for use with the VMSbus, and although it reserves two backplane signal lines for use by that bus, the VMSbus protocol is completely independent of the P1Ø14.)

SLAVE

A functional module that detects DTB cycles initiated by a MASTER and, when those cycles specify its participation, transfers data between itself and the MASTER.

SLOT

A position where a board can be inserted into a P1Ø14 backplane. If the P1Ø14 system has both a J1 and a J2 backplane (or a combination J1/J2 backplane) each slot provides a pair of 96 pin connectors. If the system has only a J1 backplane, then each slot provides a single 96 pin connector.

SUBRACK

A rigid framework that provides mechanical support for boards inserted into the backplane, ensuring that the connectors mate properly and that adjacent boards do not contact each other. It also guides the cooling airflow through the system, and ensures that inserted boards do not disengage themselves from the backplane due to vibration or shock.

SYSTEM CLOCK DRIVER

A functional module that provides a 16 MHz timing signal on the UTILITY BUS.

SYSTEM CONTROLLER BOARD

A board which resides in slot 1 of a P1Ø14 backplane and has a SYSTEM CLOCK DRIVER, a DTB ARBITER, an IACK DAISY CHAIN DRIVER, and a BUS TIMER. Some also have a SERIAL CLOCK DRIVER, a POWER MONITOR or both.

UAT

A MASTER that sends or receives data in an unaligned fashion,

or

a SLAVE that sends and receives data in an unaligned fashion.

UTILITY BUS

One of the four buses provided by the P1Ø14 backplane. This bus includes signals that provide periodic timing and coordinate the power-up and power-down of P1Ø14 systems.

WRITE CYCLE

A DTB cycle used to transfer 1, 2, or 4 bytes from a MASTER to a SLAVE. The cycle begins when the MASTER broadcasts an address and address modifier and places data on the DTB. Each SLAVE captures this address and address modifier, and checks to see if it is to respond to the cycle. If so, it stores the data and then acknowledges the transfer. The MASTER then terminates the cycle.

Appendix L

Literature References

Please refer to the following books for further detailed information:

VMEbus Specification Manual (IEEE P1Ø14/D1.2) PRINTEX

Small Computer System Interface Manual (X3TY.2 Rev 17.B)

Appendix M

Product Error Report

DEAR CUSTOMER,

WHILE FORCE COMPUTERS HAS ACHIEVED A VERY HIGH STANDARD OF QUALITY IN OUR PRODUCTS AND DOCUMENTATION, WE CONTINUALLY SEEK SUGGESTIONS FOR IMPROVEMENTS.

WE WOULD APPRECIATE ANY FEEDBACK YOU CARE TO OFFER.

PLEASE USE ATTACHED "PRODUCT ERROR REPORT" FORM FOR YOUR COMMENTS AND RETURN IT TO ONE OF OUR FORCE COMPUTERS OFFICES.

SINCERELY

FORCE COMPUTERS

COPIES OF DATA SHEETS

1. BIM 68153

2. PI/T 68230

3. SCSIBC 5386S

4. SCSIBT/R 8310

5. DMAC 68450

6. FDC 1772

MOTOROLA
SEMICONDUCTORS
P.O. BOX 20912 • PHOENIX, ARIZONA 85036

MC68153

Advance Information

BUS INTERRUPTER MODULE

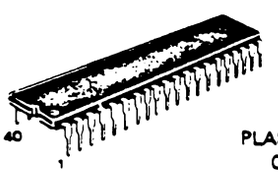
The bipolar LSI MC68153 Bus Interrupter interfaces a microcomputer system bus to multiple slave devices requiring interrupt capabilities. It handles up to 4 independent sources of interrupt requests and is fully programmable.

- VERSAbus/VMEbus Compatible
- MC68000 Compatible
- Handles 4 Independent Interrupt Sources
- 8 Programmable Read/Write Registers
- Programmable Interrupt Request Levels
- Programmable Interrupt Vectors
- Supports Interrupt Acknowledge Daisy Chain
- Control Registers Contain Flag Bits
- Single −5.0 Volt Supply
- Total Power Dissipation = 1.5 W Typical
- Temperature Range of 0°C to 70°C
- Chip Access Time = 200 ns Typical with 16 MHz Clock
- 40-Pin Dual-In-Line Package

TTL

BUS INTERRUPTER MODULE

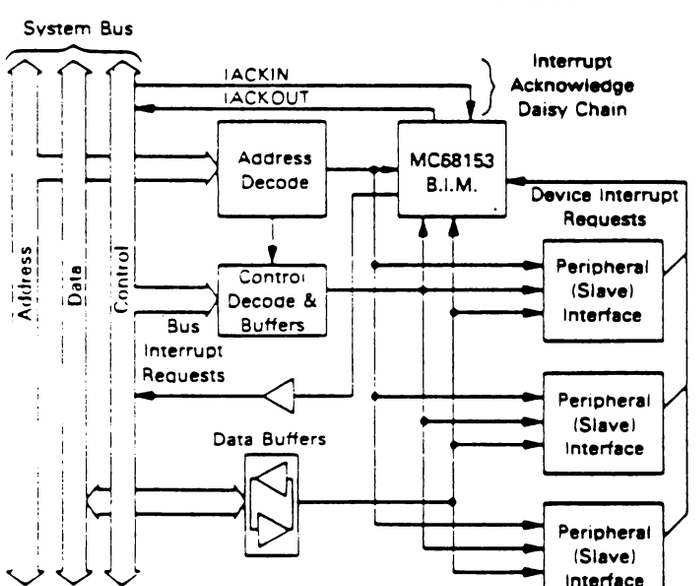
ADVANCED LOW POWER SCHOTTKY



P SUFFIX
PLASTIC PACKAGE
CASE 711-03

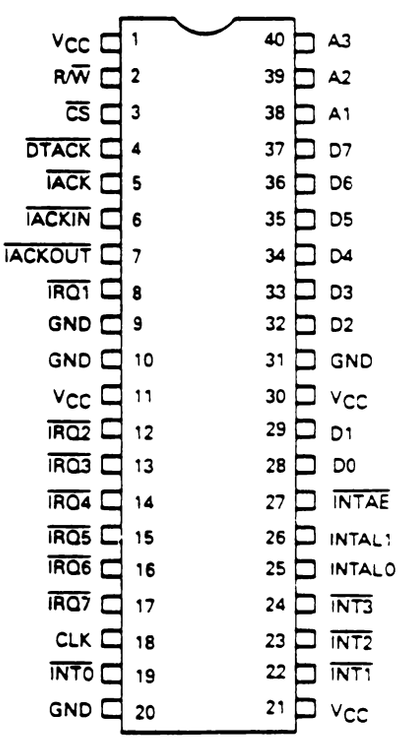
L SUFFIX
CERAMIC PACKAGE
CASE 734-04

FIGURE 1 — MC68153 SYSTEM BLOCK DIAGRAM



VERSAbus is a trademark of Motorola.

PIN ASSIGNMENTS



ADI-1057
(Replacing NP-151)

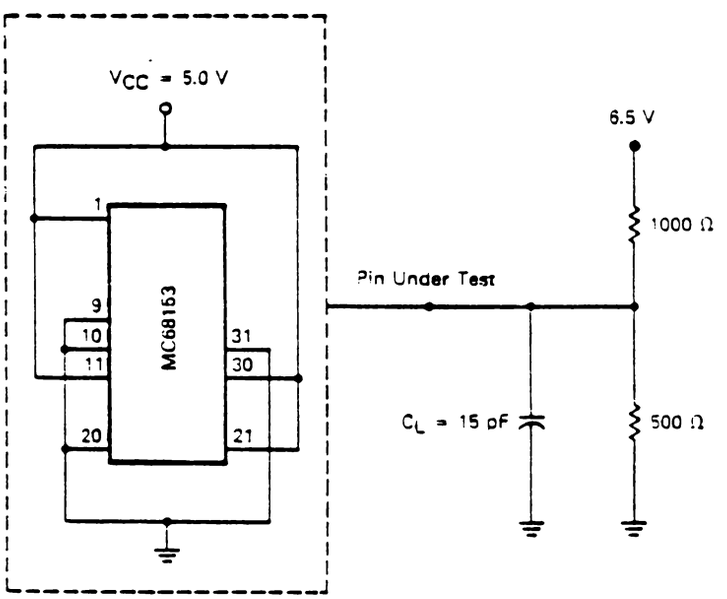
ABSOLUTE MAXIMUM RATINGS (Beyond which useful life may be impaired.)

Parameter	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.5 to −7.0	V
Input Voltage	V_{in}	−0.5 to −7.0	V
Input Current	I_{in}	−30 to −5.0	mA
Output Voltage	V_{out}	−0.5 to −5.5	V
Output Current	I_{OL}	Twice Rated I_{OL}	mA
Storage Temperature	T_{stg}	−65 to −150	°C
Junction Operating Temperature	T_J	−55 to +175	°C

DC ELECTRICAL SPECIFICATIONS (V_{CC} = 5.0 V ±5%, T_A = 0°C to 70°C)

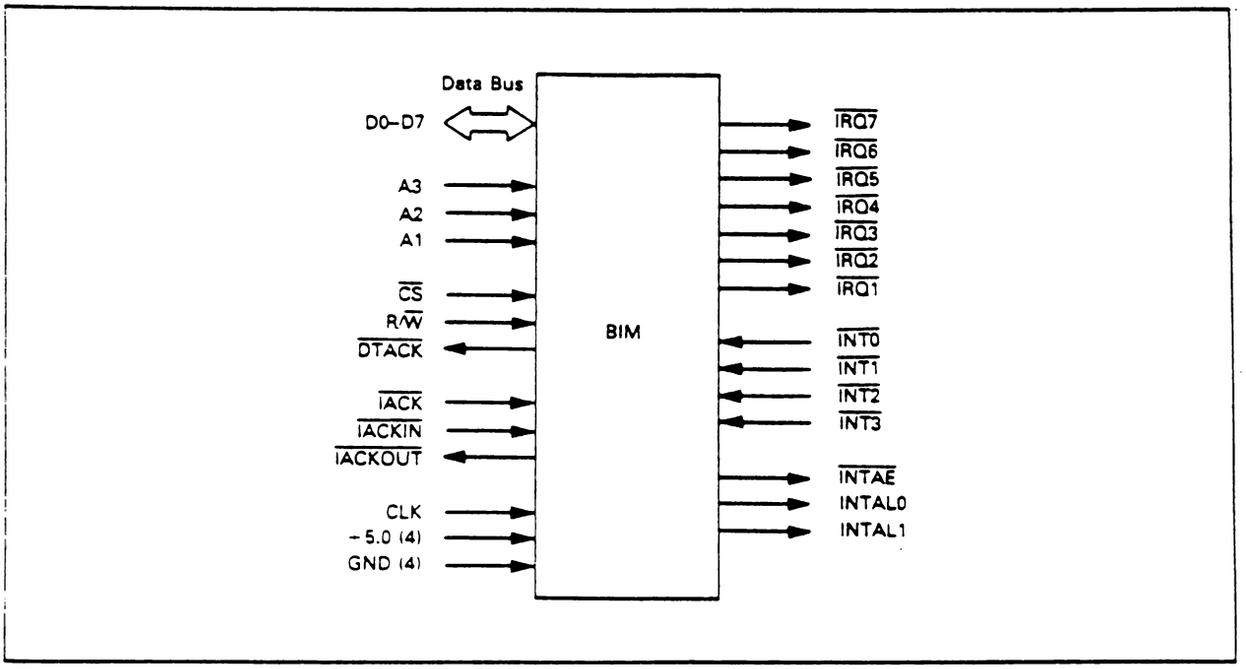
Parameter	Symbol	Min	Max	Unit	Test Conditions
High Level Input Voltage	V_{IH}	2.0	—	V	
Low Level Input Voltage	V_{IL}	—	0.8	V	
Input Clamp Voltage	V_{IK}	—	−1.5	V	V_{CC} = MIN, I_{IN} = −18 mA
High Level Output Voltage(1)	V_{OH}	2.7	—	V	V_{CC} = MIN, I_{OH} = −400 μA
Low Level Output Voltage	V_{OL}	—	0.4	V	V_{CC} = MIN, I_{OL} = 8.0 mA
Output Short Circuit Current(2)	I_{OS}	−15	−130	mA	V_{CC} = MAX, V_{OUT} = 0 V
High Level Input Current	I_{IH}	—	20	μA	V_{CC} = MAX, V_{IN} = 2.7 V
Low Level Input Current	I_{IL}	—	−0.4	mA	V_{CC} = MAX, V_{IN} = 0.4 V
Supply Current	I_{CC}	225	385	mA	V_{CC} = MAX
Output Off Current (High)	I_{OZH}	—	20	μA	V_{CC} = MAX, V_{OUT} = 2.4 V
Output Off Current (Low)	I_{OZL}	—	−20	μA	V_{CC} = MAX, V_{OUT} = 0.4 V

AC TEST CIRCUIT — AC Testing of All Outputs



NOTES:
1. Not applicable to open-collector outputs.
2. Not more than one output should be shorted at a time for longer than one second.
3. $\overline{CS}$ Low to CLK High (Setup Time) of 15 ns Min must be observed.
4. $\overline{IACK}$ Low to CLK High and $\overline{IACKIN}$ Low to CLK High (Setup Times) of 15 ns Min must be observed.
5. See Table 1 for additional performance guidelines.

FIGURE 3 — LOGICAL PIN ASSIGNMENT



INTERRUPT ACKNOWLEDGE SIGNALS — $\overline{IACK}$, $\overline{IACKIN}$, $\overline{IACKOUT}$

These three pins support the interrupt acknowledge cycle. A low level on the $\overline{IACK}$ input indicates an interrupt acknowledge cycle has been initiated. This signal is conditioned externally with Address Strobe and the lower data strobe of an MC68000 type bus. After $\overline{IACK}$ is asserted the BIM compares the interrupt level presented on address lines A1, A2, and A3 with the current levels generated internally and determines if a match exists. Then, if input $\overline{IACKIN}$ is asserted (driven low), the BIM will either complete the interrupt acknowledge cycle if a match exists or assert output $\overline{IACKOUT}$ if *no* match exists.

$\overline{IACKIN}$ and $\overline{IACKOUT}$ form part of a prioritized interrupt acknowledge daisy chain. The daisy chain prioritizes interrupters and guarantees that two or more devices requesting an interrupt on the same level will not respond to the same cycle. The requesting device (or interrupter) must wait until $\overline{IACKIN}$ is asserted and not pass the signal on (assert $\overline{IACKOUT}$) if it is to complete the interrupt acknowledge cycle.

BUS INTERRUPT REQUEST SIGNALS — $\overline{IRQ1}$ – $\overline{IRQ7}$

These open-collector outputs are low when asserted, indicating a bus interrupt is requested at the corresponding level. An open-collector buffer is normally required for sufficient drive when interfacing to a system bus. A pullup resistor is required to maintain $\overline{IRQ1}$ – $\overline{IRQ7}$ high between interrupt requests.

DEVICE INTERRUPT REQUEST SIGNALS — $\overline{INT0}$ – $\overline{INT3}$

$\overline{INT0}$ – $\overline{INT3}$ are active low inputs used to indicate to the BIM that a device wants a bus interrupt.

INTERRUPT ACKNOWLEDGE ENABLE — $\overline{INTAE}$

During an interrupt acknowledge cycle, this output pin is asserted low to indicate that outputs INTAL0 and INTAL1 are valid. These two outputs contain an encoded number (x) corresponding to the interrupt ($\overline{INTx}$) being acknowledged. This feature can be used to signal interrupting devices, which supply their own vector, when to respond to the interrupt acknowledge cycle with the vector and a $\overline{DTACK}$ signal.

INTERRUPT ACKNOWLEDGE LEVEL — INTAL0, INTAL1

These active high outputs contain an encoded number corresponding to the interrupt level being acknowledged. They are valid only when $\overline{INTAE}$ is asserted low.

CLOCK — CLK

The CLK input is used to supply the clock for internal operations of the MC68153.

RESET — $\overline{CS}$, $\overline{IACK}$

Although a reset input is not supplied, an on-board reset is performed if $\overline{CS}$ and $\overline{IACK}$ are asserted simultaneously.

FIGURE 4 — MC68153 REGISTER MODEL

ADDRESS BIT A3	A2	A1	FLAG	FLAG AUTO-CLEAR	EXTERNAL/INTERNAL	INTERRUPT ENABLE	INTERRUPT AUTO-CLEAR	INTERRUPT LEVEL			REGISTER NAME
0	0	0	F	FAC	X/$\overline{IN}$	IRE	IRAC	L2	L1	L0	CONTROL REGISTER 0
0	0	1	F	FAC	X/$\overline{IN}$	IRE	IRAC	L2	L1	L0	CONTROL REGISTER 1
0	1	0	F	FAC	X/$\overline{IN}$	IRE	IRAC	L2	L1	L0	CONTROL REGISTER 2
1	1	1	F	FAC	X/$\overline{IN}$	IRE	IRAC	L2	L1	L0	CONTROL REGISTER 3
1	0	0	V7	V6	V5	V4	V3	V2	V1	V0	VECTOR REGISTER 0
1	0	1	V7	V6	V5	V4	V3	V2	V1	V0	VECTOR REGISTER 1
1	1	0	V7	V6	V5	V4	V3	V2	V1	V0	VECTOR REGISTER 2
1	1	1	V7	V6	V5	V4	V3	V2	V1	V0	VECTOR REGISTER 3
			7	6	5	4	3	2	1	0	REGISTER BIT

REGISTER DESCRIPTION

The MC68153 contains 8 programmable read/write registers. There are four control registers (CR0 – CR3) that govern operation of the device. The other four (VR0 – VR3) are vector registers that contain the vector data used during an interrupt acknowledge cycle. Figure 4 illustrates the device register model.

CONTROL REGISTERS

There is a control register for each interrupt source, i.e., CR0 controls $\overline{INT0}$, CR1 controls $\overline{INT1}$, etc. The control registers are divided into several fields:

1. Interrupt level (L2, L1, L0) — The least significant 3-bit field of the register determines the level at which an interrupt will be generated:

L2	L1	L0	IRQ LEVEL
0	0	0	DISABLED
0	0	1	IRQ1
0	1	0	IRQ2
0	1	1	IRQ3
1	0	0	IRQ4
1	0	1	IRQ5
1	1	0	IRQ6
1	1	1	IRQ7

   A value of zero in the field disables the interrupt.

2. Interrupt Enable (IRE) — This field (Bit 4) must be set (high level) to enable the bus interrupt request associated with the control register. Thus, if the $\overline{INTX}$ line is asserted and IRE is cleared, *no* interrupt request ($\overline{IRQX}$) will be asserted.
3. Interrupt Auto-Clear (IRAC) — If the IRAC is set (Bit 3), IRE (Bit 4) is cleared during an interrupt acknowledge cycle responding to this request. This action of clearing IRE disables the interrupt request. To re-enable the interrupt associated with this register, IRE must be set again by writing to the control register.
4. External/Internal (X/$\overline{IN}$) — Bit 5 of the control register determines the response of the MC68153 during an interrupt acknowledge cycle. If the X/$\overline{IN}$ bit is clear (low level) the BIM will respond with vector data and a $\overline{DTACK}$ signal, i.e., an internal response. If X/$\overline{IN}$ is set, the vector is not supplied and no $\overline{DTACK}$ is given by the BIM, i.e., an external device should respond.
5. Flag (F) — Bit 7 is a flag that can be used in conjunction with the test and set instruction of the MC68000. It can be changed without affecting chip operation. It is useful for processor-to-processor communication and resource allocation.
6. Flag Auto-Clear (FAC) — If FAC (Bit 6) is set, the Flag bit is automatically cleared during an interrupt acknowledge cycle.

VECTOR REGISTERS

Each interrupt input has its own associated vector register. Each register is 8 bits wide and supplies a data byte during its interrupt acknowledge cycle if the associated External/Internal (X/$\overline{IN}$) control register bit is clear. This data can be status, identification, or address information depending on system usage. The information is programmed by the system user.

DEVICE RESET

When the MC68153 is reset, the registers are set to a known condition. The control registers are set to all zeros (low). The vector registers are set to $0F. This value is the MC68000 vector for an uninitialized interrupt vector.

FUNCTIONAL DESCRIPTION

SYSTEM OVERVIEW

The MC68153 is compatible with many system buses, however, it is primarily intended for VMEbus, VERSAbus and MC68000 applications. Figure 5 shows a system configuration similar to VMEbus. In the figure only one system Data Transfer Bus (DTB) master is used. The Priority Interrupt structure provides a means for peripheral slave devices to ask for an interrupt of other processor (DTB master) activity and receive service from the processor. The MC68153 BIM acts as an interface device requesting and responding to interrupt acknowledge cycles for up to 4 independent slaves.

In Figure 5, functional modules are identified as Interrupters and an Interrupt Handler. An Interrupter (such as the MC68153) receives slave requests for an interrupt and handles all interface to the system bus required to ask for and respond to interrupt requests. The Interrupt Handler receives the bus interrupt requests, determines when an interrupt acknowledge will occur and at which level, and finally either performs the interrupt acknowledge (IACK) cycle or tells the DTB master to execute the IACK cycle.

The signal lines in the Priority Interrupt structure include (* — indicates active low):

1. IRQ1*–IRQ7* — seven prioritized interrupt request lines.
2. IACK* — signal line that indicates an interrupt acknowledge cycle is occurring.
3. IACKIN*/IACKOUT* — two signals that form part of a daisy chain that prioritizes interrupters.

In addition Data Transfer Bus control signals are involved in the IACK bus cycle:

1. AS* — the Address Strobe asserted low indicates a valid address is on the bus.
2. DSO* — the lower Data Strobe asserted low indicates a data transfer will occur on bus bits D00–D07.
3. WRITE* — the Read/Write is negated indicating the data is to be read from the Interrupter.
4. A01–A03 — Address lines A01–A03 contain the encoded priority level of the IACK cycle.
5. D00–D07 — Data bus lines D00–D07 are used to pass the interrupt vector from the responding Interrupter to the Interrupt Handler.
6. DTACK* — Data Transfer Acknowledge asserted low signals that the Interrupter has put the vector on the data bus.

FIGURE 5 — SIMPLE VMEbus CONFIGURATION

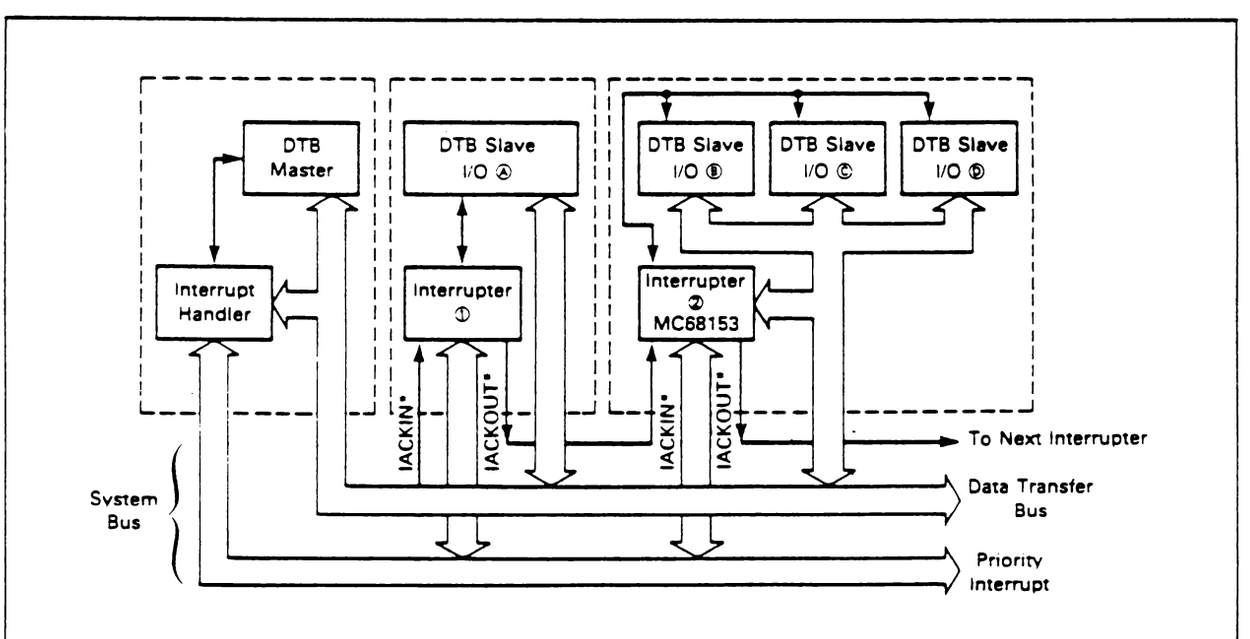
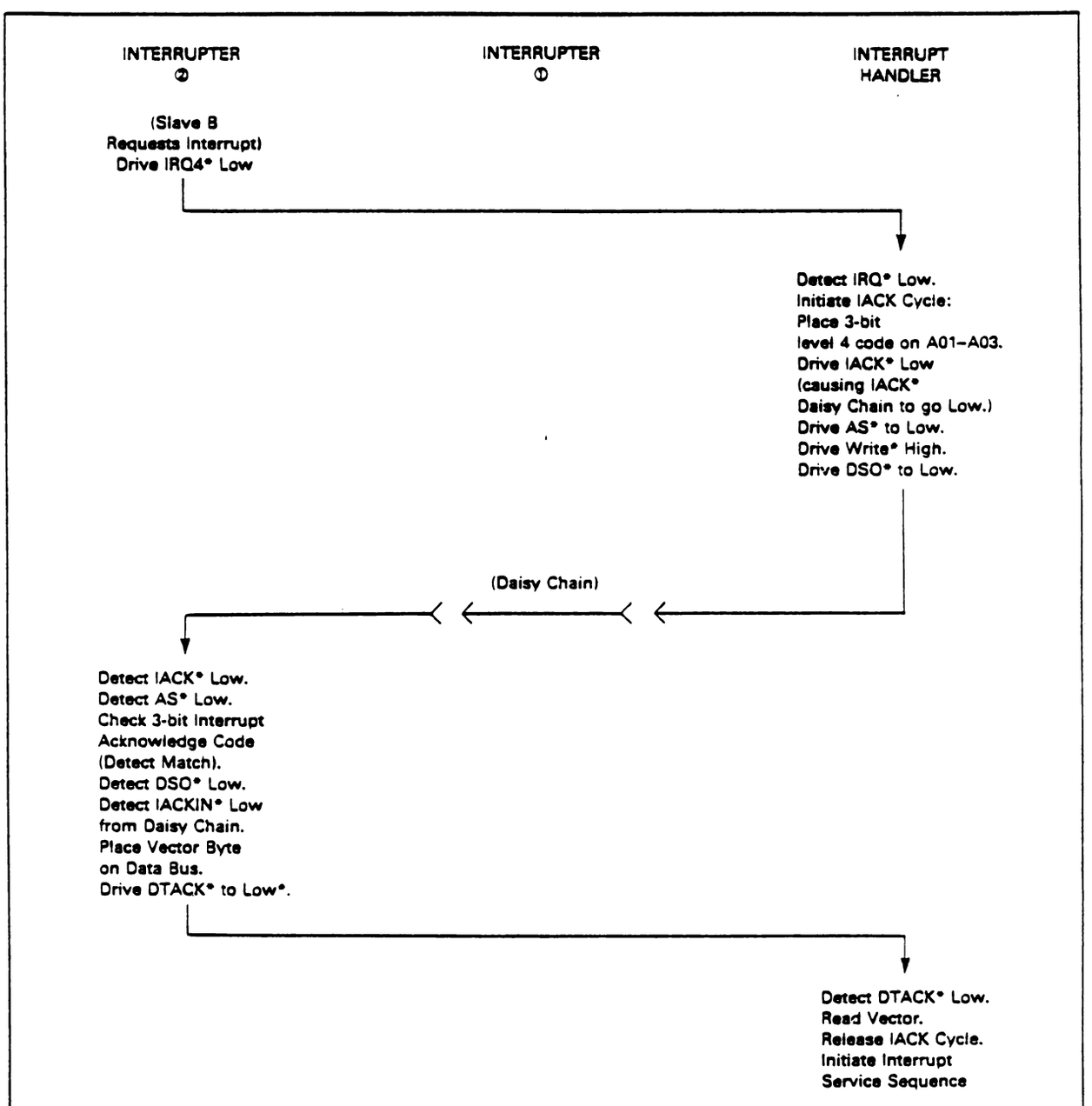


Figure 6 shows a flow diagram of a typical interrupt request and acknowledge operation. Briefly, the sequence of events is first, an Interrupter makes a request, next the Handler responds with an IACK cycle, then the Interrupter passes a vector to the Handler completing the IACK cycle, and finally the Handler uses the vector to determine additional action. Typically, an interrupt service routine is stored in software and the vector points to its starting address.

Note the daisy chain operation. If the IACK level (on A01–A03) does not match the Interrupter's request level or if no request is pending, the Interrupter passes the IACKIN* signal on and asserts IACKOUT*. This sequential action automatically prioritizes Requesters on the same level (first one in line with a request pending gets serviced) and prevents two or more Interrupters from responding simultaneously.

FIGURE 6 — INTERRUPT REQUEST AND ACKNOWLEDGE OPERATION FLOW DIAGRAM



AC ELECTRICAL SPECIFICATIONS (V_{CC} = 5.0 V ±5%, T_A = 0°C to 70°C)

Parameter	Test Number[5]	Max (ns)
CLK High to Data Out Valid (Delay)[3]	1	55
CLK High to $\overline{DTACK}$ Low (Delay)[3]	2	40
$\overline{CS}$ High to $\overline{DTACK}$ High (Delay)	3	35
CLK High to Data Out Valid (Delay)[4]	4	55
CLK High to $\overline{INTAE}$ Low (Delay)[4]	5	40
$\overline{IACK}$ High to Data Out High Impedance (Delay)	6	60
$\overline{IACK}$ High to $\overline{DTACK}$ High (Delay)	7	45
$\overline{CS}$ High to Data Out High (Delay)	8	45
$\overline{CS}$ High to IRQ High (Delay)	9	60
$\overline{IACK}$ High to $\overline{INTAE}$ High (Delay)	10	35

GENERAL DESCRIPTION

The MC68153 Bus Interrupter Module (BIM) is designed to serve as an interrupt requester for peripheral devices in a microcomputer system. Up to 4 independent devices can be interfaced to the system bus by the MC68153. Intended for asynchronous master/slave bus operation, the BIM is compatible with VERSAbus, VMEbus, MC68000 device bus, and other system buses. Figure 1 shows a block diagram of a typical configuration. In this example, three peripheral devices (bus slaves) are connected to the system data bus. Each of these devices could be parallel I/O, serial I/O, or some other function. An interrupt request from any device is routed to the MC68153, and the BIM handles all interface to the system bus. It generates a bus interrupt request as a result of the device interrupt request. When the system interrupt handler or processor responds with an interrupt acknowledge cycle, the MC68153 can answer supplying an interrupt vector and handling all timing.

The functional block diagram of the MC68153 is shown in Figure 2. The device contains circuitry to accept four separate interrupt sources ($\overline{INT0}$ – $\overline{INT3}$). Interface to the system bus includes generation of bus interrupt requests ($\overline{IRQ1}$ – $\overline{IRQ7}$), response to a bus interrupt acknowledge cycle (either supplying a vector or passing on a daisy chain signal), and releasing the bus interrupt request signal at the proper time. The BIM has flexibility provided by eight programmable read/write registers. Four 8-bit vector registers (VR0 – VR3) contain status/address information and supply a byte vector in response to an interrupt acknowledge cycle for the corresponding interrupt source. Four other 8-bit control registers (CR0 – CR3) contain information that oversees operation of the interrupt circuitry. The control information is programmable and includes interrupt request level and interrupt enable and disable. Also contained in the control registers are flag-bits. These flags are useful for task coordination, resource management, and interprocessor communication.

SIGNAL DESCRIPTION

Throughout the data sheet, signals are presented using the terms asserted and negated independent of whether the signal is asserted in the high voltage or low voltage state. Active low signals are denoted by a superscript bar.

BIDIRECTIONAL DATA BUS — D0 – D7

Pins D0 – D7 form an 8-bit bidirectional data bus to/from the system bus. These are active high, 3-state pins.

ADDRESS INPUTS — A1 – A3

These active high inputs serve two functions. One function is to select one of the eight possible registers during a read or write cycle. Secondly, during an interrupt acknowledge A1 – A3 show the level of interrupt being acknowledged, and the BIM uses these to determine if a match exists with an internal level.

CHIP SELECT — $\overline{CS}$

$\overline{CS}$ is an active low input used to select the BIM's registers for the current bus cycle. Address strobe, data strobe, and appropriate address bits must be included in the chip select equation.

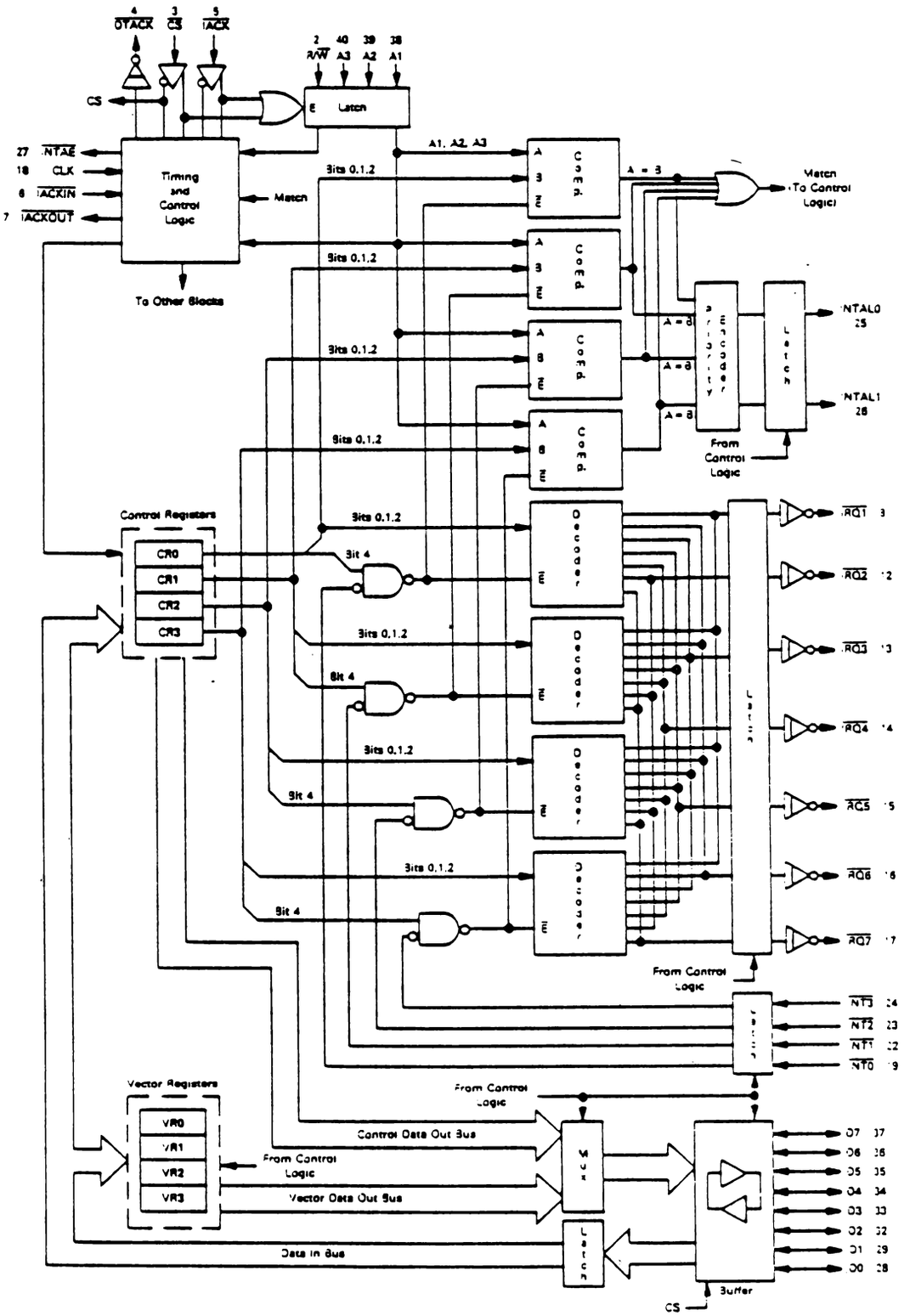
READ/WRITE — R/$\overline{W}$

The R/$\overline{W}$ input is a signal from the system bus used to determine if the current bus cycle is a read (high) or write (low).

DATA TRANSFER ACKNOWLEDGE — $\overline{DTACK}$

$\overline{DTACK}$ is an open-collector, active low output that signals the completion of a read, write, or interrupt acknowledge cycle. During read or interrupt acknowledge cycles, $\overline{DTACK}$ is asserted by the MC68153 after data has been provided on the data bus; during write cycles it is asserted after data has been accepted from the data bus. A pullup resistor is required to maintain $\overline{DTACK}$ high between bus cycles.

FIGURE 2 — MC68153 FUNCTIONAL BLOCK DIAGRAM



This discussion is a very cursory look at the bus operation. For more details including situations with multiple bus masters, the user is directed to the VMEbus Specification MVMEBS or VERSAbus Specification M68KVBS. Also, the MC68153 can be used with other buses having similar interrupt structures.

BIM BUS INTERFACE

Figure 7 shows a simplified block diagram of the MC68153 interface to VERSAbus or VMEbus. Address Decode and Control Decode are dependent on the application and must be designed to guarantee BIM ac specifications. It is possible in most cases that the decode logic can be shared with the slave devices. Buffers are provided where shown to comply with bus loading and drive specifications. It is also possible that buffers can be shared with the slave bus interface.

READ/WRITE OPERATION

All eight BIM registers can be accessed from the system bus in both read and write modes. The BIM has an asynchronous bus interface, primarily designed for MC68000-like buses. The following signals generate read and write cycles: Chip Select ($\overline{CS}$), Read/Write (R/$\overline{W}$), Address Inputs (A1–A3), Data Bus (D0–D7), and Data Transfer Acknowledge ($\overline{DTACK}$). During read and write cycles the internal registers are selected by A1, A2, and A3 in compliance with the Figure 4 Truth Table.

Figure 8 shows the device timing for a read cycle. R/$\overline{W}$ and A1–A3 are latched on the falling edge of $\overline{CS}$ and must meet specified setup and hold times. Chip access time for valid data and $\overline{DTACK}$ are dependent on the clock frequency as shown in the figure.

Figure 9 shows the device timing for a write cycle. R/$\overline{W}$, A1–A3, and D0–D7 are latched on the falling edge of $\overline{CS}$ and must meet specified setup and hold times. Chip access time for $\overline{DTACK}$ is dependent on the clock frequency as shown in the figure.

FIGURE 7 — VMEbus/VERSAbus INTERFACE BLOCK DIAGRAM

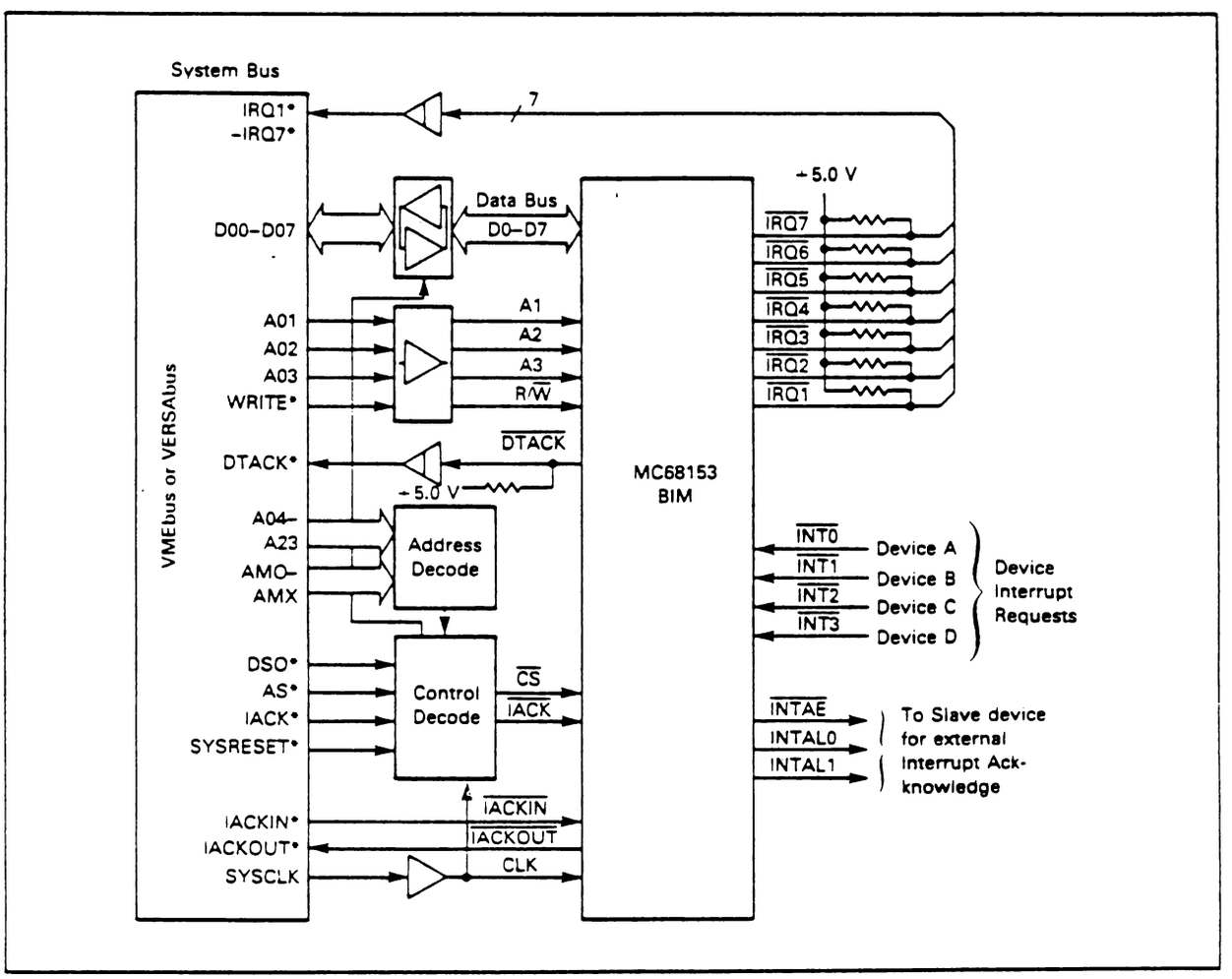


FIGURE 8 — READ CYCLE

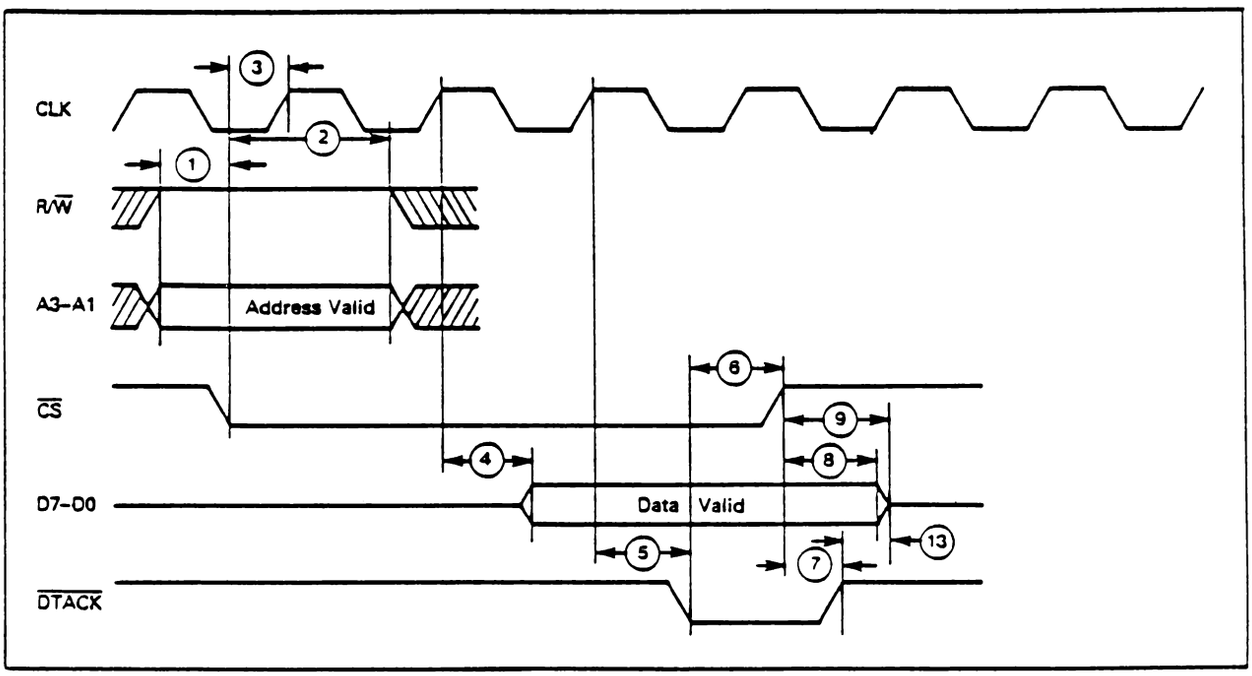
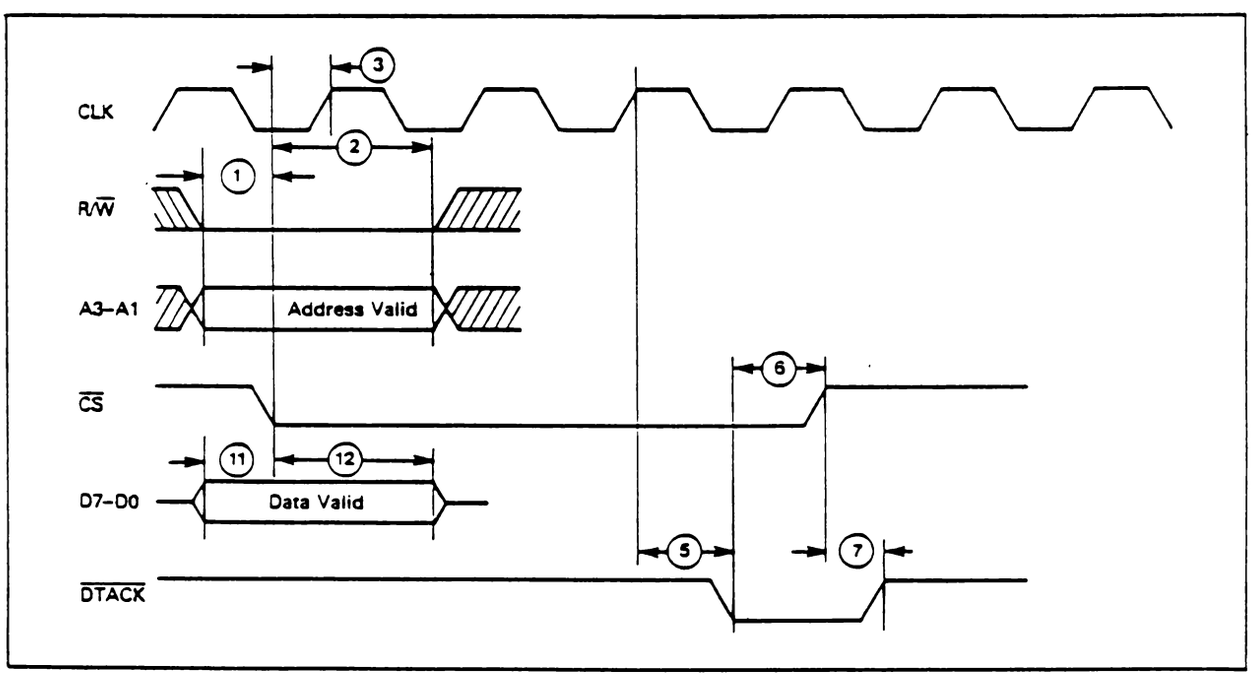


FIGURE 9 — WRITE CYCLE



INTERRUPT REQUESTS

The MC68153 accepts device interrupt requests on inputs $\overline{INT0}$, $\overline{INT1}$, $\overline{INT2}$, and $\overline{INT3}$. Each input is regulated by Bit 4 (IRE) of the associated control register (CR0 controls $\overline{INT0}$, CR1 controls $\overline{INT1}$, etc). If IRE (Interrupt Enable) is set and a device input is asserted, an Interrupt Request open-collector output ($\overline{IRQ1}$–$\overline{IRQ7}$) is asserted. The asserted $\overline{IRQX}$ output is selected by the value programmed in Bits 0, 1, and 2 of the control register (L0, L1, and L2). This 3-bit field determines the interrupt request level as set by software.

Two or more interrupt sources can be programmed to the same request level. That $\overline{IRQX}$ output will remain asserted until multiple interrupt acknowledge cycles respond to all requests.

If the interrupt request level is set to zero, the interrupt is disabled because there is no corresponding IRQ output.

INTERRUPT ACKNOWLEDGE

The response of an Interrupt Handler to a bus interrupt request is an interrupt acknowledge cycle. The IACK cycle is initiated in the MC68153 by receiving $\overline{IACK}$ low. R/$\overline{W}$, A1, A2, A3 are latched, and the interrupt level on line A1–A3 is compared with any interrupt requests pending in the chip. Further activity can be one of four cases:

1. No further action required — This occurs if $\overline{IACKIN}$ is **not** asserted. Asserting $\overline{IACK}$ only starts the BIM activity. If the daisy chain signal never reaches the MC68153 ($\overline{IACKIN}$ is not asserted), another Interrupter has responded to the $\overline{IACK}$ cycle. The cycle will end, the chip $\overline{IACK}$ is negated, and no additional action is required.

2. Pass on the interrupt acknowledge daisy chain — For this case, $\overline{IACKIN}$ input is asserted by the preceding daisy chain Interrupter, and $\overline{IACKOUT}$ output is in turn asserted. The daisy chain signal is passed on when no interrupts are pending on a matching level or when any possible interrupts are disabled. The Interrupt Enable (IRE) bit of a control register can disable any interrupt requests, and in turn, any possible matches.

3. Respond internally — For this case, $\overline{IACKIN}$ is asserted and a match is found. The MC68153 completes the IACK cycle by supplying an interrupt vector from the proper vector register followed by a $\overline{DTACK}$ signal asserted. $\overline{IACKOUT}$ is not asserted because the interrupt acknowledge cycle is completed by this device.

   For the MC68153 to respond in this mode of operation, the EXTERNAL $\overline{INTERNAL}$ control register bit (X/$\overline{IN}$) must be zero. For each source of interrupt request, the associated control register determines the BIM response to an IACK cycle, and the X/$\overline{IN}$ bit sets this response either internally (X/$\overline{IN}$ = 0) or externally (X/$\overline{IN}$ = 1).

4. Respond externally — For the final case, $\overline{IACKIN}$ is also asserted, a match is found and the associated control register has X/$\overline{IN}$ bit set to one. The MC68153 does not assert $\overline{IACKOUT}$ and does assert $\overline{INTAE}$ low. $\overline{INTAE}$ signals that the requesting device must complete the IACK cycle (supplying a vector and $\overline{DTACK}$) and that the 2-bit code contained on outputs INTAL0 and INTAL1 shows which interrupt source is being acknowledged.

These cases are discussed in more detail in the following paragraphs.

Internal Interrupt Acknowledge

For an internal interrupt acknowledge to occur, the following conditions must be met:

1. One or more device interrupt inputs ($\overline{INT0}$–$\overline{INT3}$) has been asserted and corresponding control bit IRE value is one.

2. $\overline{IACK}$ asserted.

3. A match exists between [A3, A2, A1] and the [L2, L1, L0] field of an enabled, requesting control register. If two or more devices are requesting at the same interrupt level, preference is given to the highest number requester, that is, $\overline{INT3}$ has highest priority and $\overline{INT0}$ has lowest.

4. Control register bit X/$\overline{IN}$ of matching interrupt source must be zero.

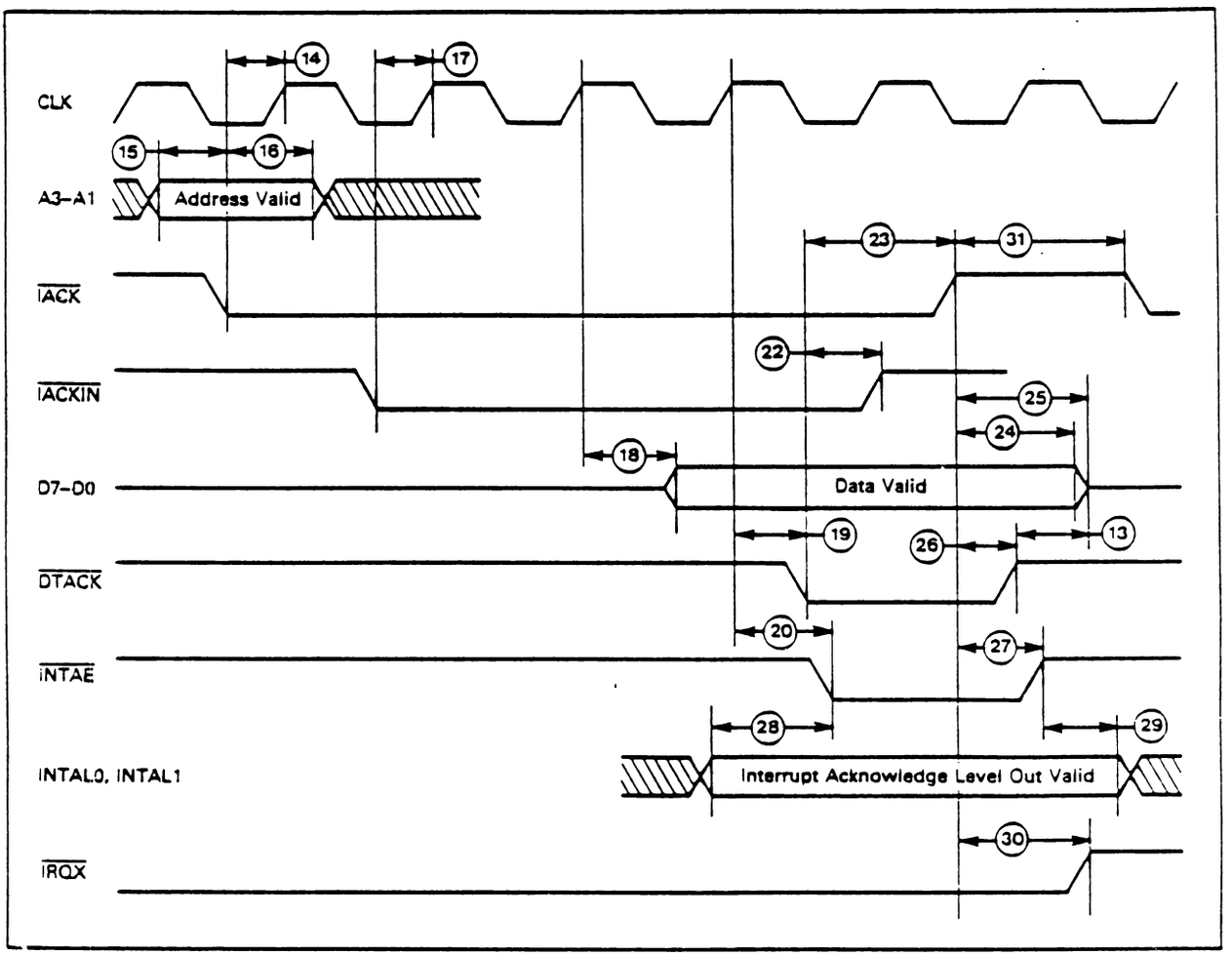
5. $\overline{IACKIN}$ asserted.

The internal interrupt acknowledge cycle timing is shown in Figure 10. The 8-bit interrupt acknowledge vector is presented to the data bus and $\overline{DTACK}$ is asserted. Note also that INTAL0 and INTAL1 are valid and $\overline{INTAE}$ is asserted during this cycle although they would normally not be used. The cycle is terminated (data and $\overline{DTACK}$ released) after $\overline{IACK}$ is negated.

During the IACK cycle, the INTERRUPT AUTO-CLEAR control bit (IRAC) comes into play. If the IRAC = one for the responding interrupt source, the INTERRUPT ENABLE (IRE) bit is automatically cleared during the IACK cycle, thus disabling the associated interrupt input and any $\overline{IRQX}$ output asserted due to this interrupt input. Before another interrupt can be requested from this source, IRE must be set to one by writing to the control register.

Note that $\overline{IACKOUT}$ is not asserted because this device is responding to the IACK and does not pass the daisy chain signal on. Also, new device interrupt requests occurring on $\overline{INT0}$–$\overline{INT3}$ after $\overline{IACK}$ is asserted are locked out to prevent any race conditions on the daisy chain.

FIGURE 10 — INTERRUPT ACKNOWLEDGE CYCLE — INTERNAL VECTOR



External Interrupt Acknowledge

For an external interrupt acknowledge, the same conditions as listed above are met with one exception. Control register bit X/$\overline{IN}$ of matching interrupt source must be set to one. The timing is shown in Figure 11. For this cycle, the interrupt vector and $\overline{DTACK}$ must be supplied by an external device. $\overline{INTAE}$ is asserted indicating that INTAL0 and INTAL1 are valid. The external device can use these signals to enable the vector and $\overline{DTACK}$. The cycle is terminated after $\overline{IACK}$ is negated.

The IRAC control bit acts in the external interrupt acknowledge the same as described for the internal response (see above). Also, $\overline{IACKOUT}$ is not asserted and new device interrupts are disabled for reasons discussed above.

Pass On IACK Daisy Chain

If the MC68153 has no interrupt request pending at the same level as the interrupt acknowledge, the IACK daisy chain signal is passed on to the next device if $\overline{IACKIN}$ is asserted. The following conditions are thus met:

1. $\overline{IACK}$ asserted.
2. No match exists between [A3, A2, A1] and the [L2, L1, L0] field of an enabled, requesting control register.
3. $\overline{IACKIN}$ is asserted.

$\overline{IACKOUT}$ is asserted if these conditions are valid. This output drives $\overline{IACKIN}$ of the next Interrupter on the daisy chain, passing the signal along. Figure 12 shows the timing for this case. $\overline{IACKOUT}$ is negated after $\overline{IACK}$ is negated.

FIGURE 11 — INTERRUPT ACKNOWLEDGE CYCLE — EXTERNAL VECTOR

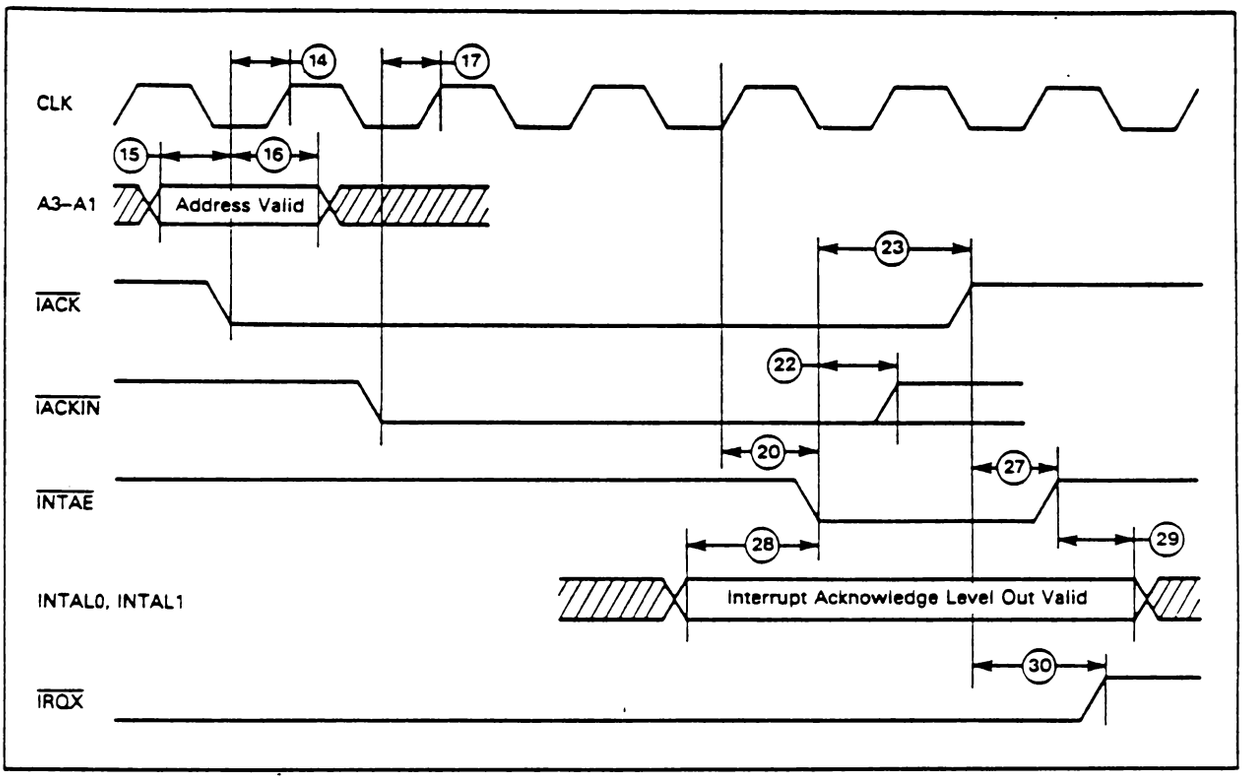
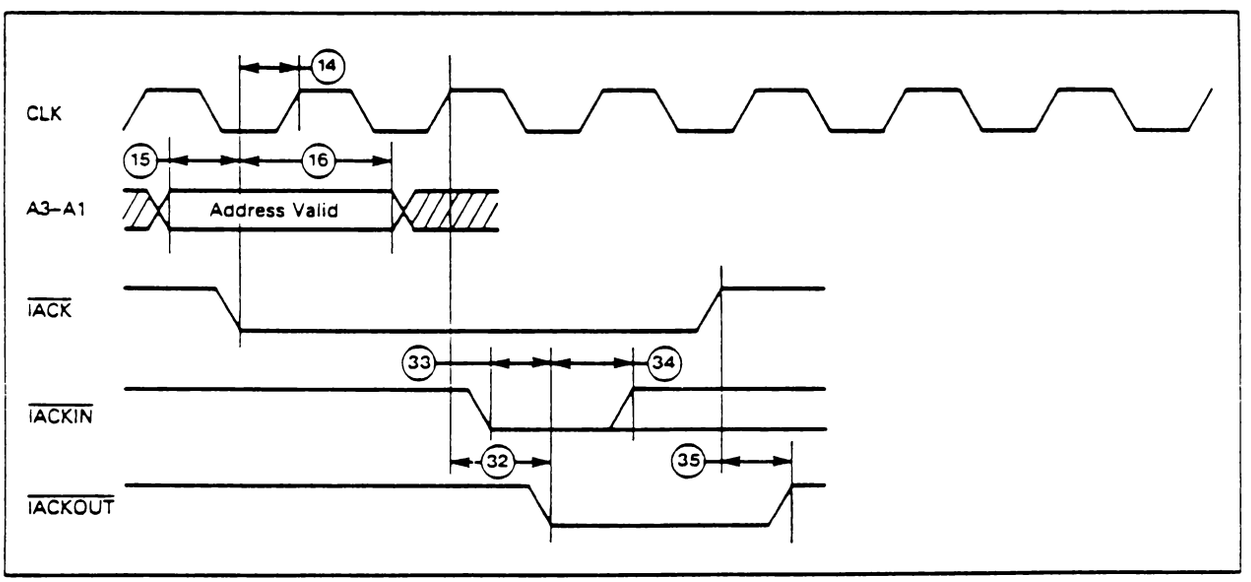


FIGURE 12 — INTERRUPT ACKNOWLEDGE CYCLE — IACKOUT



CONTROL REGISTER FLAGS

Each control register contains a Flag bit (F) and a Flag Auto-Clear bit (FAC). Both bits can be read or altered via a register write without affecting the interrupt operation of the device. The Flag is useful as a status indicator for resource management and as a semaphor in multitasking or multiprocessor systems. Flag (F) is located in bit position 7 and can be used with the MC68000 Test and Set (TAS) instruction.

The Flag Auto-Clear (FAC) is used to manipulate the Flag bit. If the Flag is set to one and the FAC is also one, an interrupt acknowledge cycle to the associated interrupt source clears the Flag bit. This feature is useful in determining the interrupt status and passing messages.

RESET

There is no reset input, however, a chip reset is activated by asserting both $\overline{CS}$ and $\overline{IACK}$ simultaneously (Figure 13). These inputs should be held low for a minimum of two clock cycles for a full reset function. The control registers are reset to all zeroes and the Vector Registers are set to a value of $0F. This vector value is the uninitialized vector for the MC68000. See the MC68000 Users Manual for more details on this vector.

CLOCK

The chip clock is required for internal operation to occur. Typical frequency is 16 MHz in VMEbus and VERSAbus applications derived from the system clock. Any frequency can be used, however, up to 25 MHz (Figure 14).

FIGURE 13 — RESET

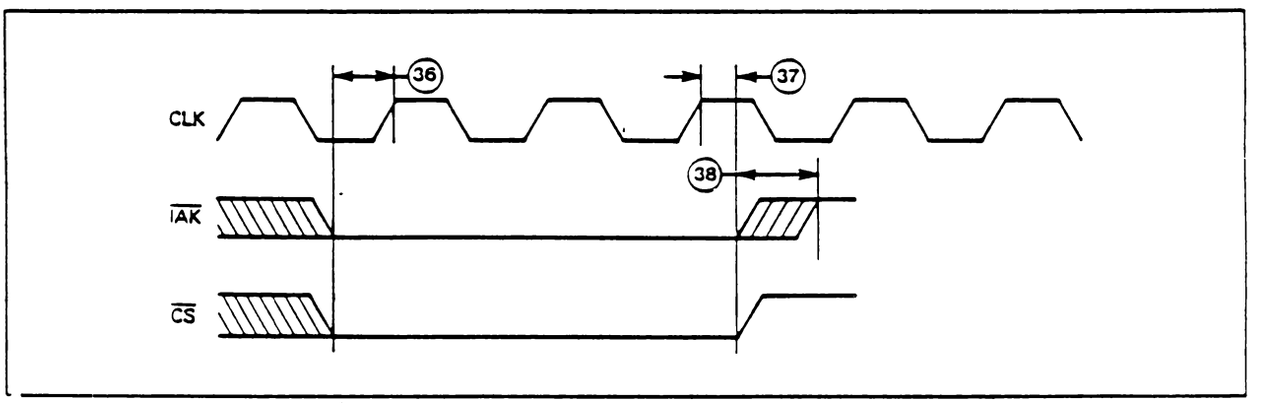


FIGURE 14 — CLOCK WAVEFORM

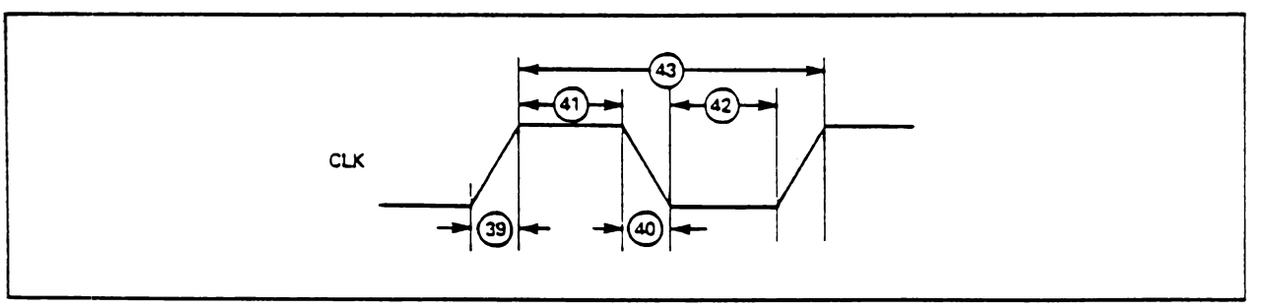


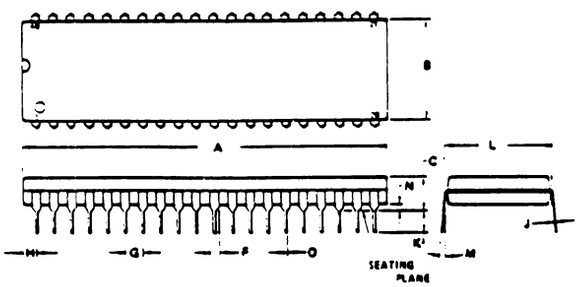
TABLE 1
AC PERFORMANCE SPECIFICATIONS
(V_{CC} = 5.0 V ±5%, T_A = 0°C to 70°C)

Number	Characteristic	Min	Max	Units	Notes
1	R/$\overline{W}$, A1–A3 Valid to $\overline{CS}$ Low (Setup Time)	10	—	ns	
2	$\overline{CS}$ Low to R/$\overline{W}$, A1–A3 Invalid (Hold Time)	5.0	—	ns	
3	$\overline{CS}$ Low to CLK High (Setup Time)	15	—	ns	1
4	CLK High to Data Out Valid (Delay)	—	55	ns	2
5	CLK High to $\overline{DTACK}$ Low (Delay)	—	40	ns	2
6	$\overline{DTACK}$ Low to $\overline{CS}$ High	0		ns	
7	$\overline{CS}$ High to $\overline{DTACK}$ High (Delay)	—	35	ns	10
8	$\overline{CS}$ High to Data Out Invalid (Hold Time)	0	—	ns	
9	$\overline{CS}$ High to Data Out High-Impedance (Hold Time)	—	50	ns	
10	$\overline{CS}$ High to $\overline{CS}$ or $\overline{IACK}$ Low	20	—	ns	
11	Data In Valid to $\overline{CS}$ Low (Setup Time)	10	—	ns	
12	$\overline{CS}$ Low to Data In Invalid (Hold Time)	5.0	—	ns	
13	$\overline{DTACK}$ High to Data Out High-Impedance	—	25	ns	10
14	$\overline{IACK}$ Low to CLK High (Setup Time)	15	—	ns	1
15	A1–A3 Valid to $\overline{IACK}$ Low (Setup Time)	10	—	ns	
16	$\overline{IACK}$ Low to A1–A3 Invalid (Hold Time)	5.0	—	ns	
17	$\overline{IACKIN}$ Low to CLK High (Setup Time)	15	—	ns	1, 8
18	CLK High to Data Out Valid (Delay)	—	55	ns	3
19	CLK High to $\overline{DTACK}$ Low (Delay)	—	40	ns	3
20	CLK High to $\overline{INTAE}$ Low (Delay)	—	40	ns	3
22	$\overline{DTACK}$ Low to $\overline{IACKIN}$ High	0	—	ns	8
23	$\overline{DTACK}$ Low to $\overline{IACK}$ High	0	—	ns	
24	$\overline{IACK}$ High to Data Out Invalid (Hold Time)	0	—	ns	
25	$\overline{IACK}$ High to Data Out High Impedance (Delay)	—	60	ns	
26	$\overline{IACK}$ High to $\overline{DTACK}$ High (Delay)	—	45	ns	10
27	$\overline{IACK}$ High to $\overline{INTAE}$ High (Delay)	—	35	ns	
28	INTAL0, INTAL1 Valid to $\overline{INTAE}$ Low (Setup Time)	1.0	2.0	CLK Per	
29	$\overline{INTAE}$ High to INTAL0, INTAL1 Invalid (Hold Time)	1.0	2.0	CLK Per	
30	$\overline{IACK}$ High to $\overline{IRQx}$ High (Delay)	—	50	ns	7, 10
31	$\overline{IACK}$ High to $\overline{IACK}$ or $\overline{CS}$ Low	20	—	ns	
32	CLK High to $\overline{IACKOUT}$ Low (Delay)	—	40	ns	5
33	$\overline{IACKIN}$ Low to $\overline{IACKOUT}$ Low (Delay)	—	30	ns	4, 8
34	$\overline{IACKOUT}$ Low to $\overline{IACKIN}$, $\overline{IACK}$ High	0	—	ns	8
35	$\overline{IACK}$ High to $\overline{IACKOUT}$ High (Delay)	—	35	ns	
36	$\overline{IACK}$ and $\overline{CS}$ both Low to CLK High (Setup Time)	15	—	ns	9
37	CLK High to $\overline{IACK}$ or $\overline{CS}$ High (Hold Time)	0	—	ns	
38	$\overline{IACK}$ or $\overline{CS}$ High to $\overline{IACK}$ and $\overline{CS}$ High (Skew)	—	1.0	CLK Per	6
39	Clock Rise Time	—	10	ns	
40	Clock Fall Time	—	10	ns	
41	Clock High Time	20	—	ns	
42	Clock Low Time	20	—	ns	
43	Clock Period	40	—	ns	

NOTES:

1. This specification only applies if the VBIM had completed all operations initiated by the previous bus cycle when $\overline{CS}$ or $\overline{IACK}$ was asserted. Following a normal bus cycle, all operations are completed within 2 clock cycles after $\overline{CS}$ or $\overline{IACK}$ have been negated. If $\overline{IACK}$ or $\overline{CS}$ is asserted prior to completion of these operations, the new cycle, and hence, $\overline{DTACK}$ is postponed.
   If the $\overline{IACK}$, $\overline{IACKIN}$ or $\overline{CS}$ setup time is violated, $\overline{DTACK}$ may be asserted as shown, or may be asserted one clock cycle later (i.e. $\overline{IACK}$ will not be recognized until the next rising edge of the clock).
2. Assumes that 3 has been met.
3. Assumes that 14 and 17 have both been met.
4. Assumes that 14 has been met. ($\overline{IACKOUT}$ cannot go low prior to $\overline{IACKIN}$ going low).
5. Assumes that 14 has been met and $\overline{IACKIN}$ has been low for at least the amount of time specified by 33.
6. 38 is the minimum skew between the last moment when both $\overline{IACK}$ and $\overline{CS}$ are asserted to when both are negated, to insure that an access cycle is not unintentionally started.
7. Assumes no other $\overline{INTx}$ input is causing $\overline{IRQx}$ to be driven low.
8. In non-daisy chain systems, $\overline{IACKIN}$ may be tied low.
9. Failure to meet this spec. causes RESET to be ignored for 1 clock period. It is then necessary to keep these signals low for 3 clock periods instead of 2.
10. Delay time is specified from input signal to Open-Collector Output pulled High thru 1.0 kΩ resistor to +6.5 V.

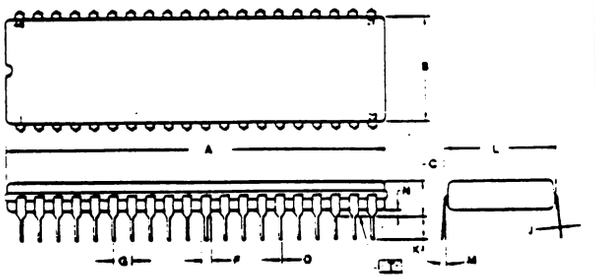
OUTLINE DIMENSIONS



DIM	MILLIMETERS MIN	MAX	INCHES MIN	MAX
A	51.68	52.45	2.035	2.065
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.38	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

NOTES:
1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25 mm (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

CASE 711-03
PLASTIC PACKAGE



DIM	MILLIMETERS MIN	MAX	INCHES MIN	MAX
A	51.31	53.24	2.020	2.096
B	12.70	15.49	0.500	0.610
C	4.06	5.84	0.160	0.220
D	0.38	0.56	0.015	0.022
F	1.27	1.65	0.050	0.065
G	2.54 BSC		0.100 BSC	
J	0.20	0.30	0.008	0.012
K	3.18	4.06	0.125	0.160
L	15.24 BSC		0.600 BSC	
M	5°	15°	5°	15°
N	0.51	1.27	0.020	0.050

NOTES:
1. DIM -A- IS DATUM.
2. POSITIONAL TOLERANCE FOR LEADS: ⌖ Ø 0.25 (0.010) Ⓜ | T | A Ⓜ |
3. | T | IS SEATING PLANE.
4. DIM L TO CENTER OF LEADS WHEN FORMED PARALLEL.
5. DIMENSIONS A AND B INCLUDE MENISCUS.
6. DIMENSIONING AND TOLERANCING PER ANSI Y14.5, 1973

CASE 734-04
CERAMIC PACKAGE

TYPICAL THERMAL CHARACTERISTICS

Package	θJA (Junction to Ambient) Still Air	Junction Temperature Still Air @ 70°C Ambient
L Suffix	40°C/W	147°C
P Suffix	35°C/W	137°C

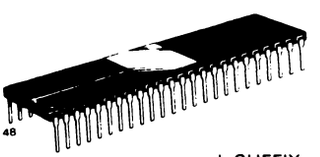
MOTOROLA
SEMICONDUCTORS
3501 ED BLUESTEIN BLVD AUSTIN TEXAS 78721

MC68230L8
MC68230L10

Advance Information

HMOS
(HIGH-DENSITY N-CHANNEL SILICON-GATE)

PARALLEL INTERFACE/TIMER



L SUFFIX
CERAMIC PACKAGE
CASE 740

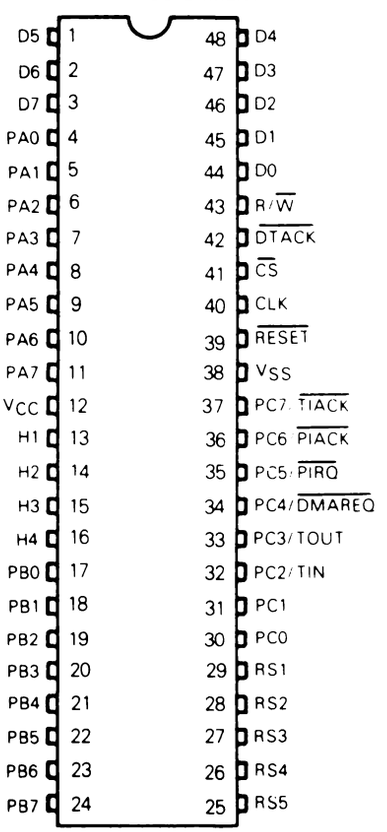
P SUFFIX
PLASTIC PACKAGE
AVAILABLE 2Q82

MC68230 PARALLEL INTERFACE/TIMER

The MC68230 Parallel Interface/Timer provides versatile double buffered parallel interfaces and an operating system oriented timer to MC68000 systems. The parallel interfaces operate in unidirectional or bidirectional modes, either 8 or 16 bits wide. In the unidirectional modes, an associated data direction register determines whether the port pins are inputs or outputs. In the bidirectional modes the data direction registers are ignored and the direction is determined dynamically by the state of four handshake pins. These programmable handshake pins provide an interface flexible enough for connection to a wide variety of low, medium, or high speed peripherals or other computer systems. The PI/T ports allow use of vectored or autovectored interrupts, and also provide a DMA Request pin for connection to the MC68450 Direct Memory Access Controller or a similar circuit. The PI/T timer contains a 24-bit wide counter and a 5-bit prescaler. The timer may be clocked by the system clock (PI/T CLK pin) or by an external clock (TIN pin), and a 5-bit prescaler can be used. It can generate periodic interrupts, a square wave, or a single interrupt after a programmed time period. Also it can be used for elapsed time measurement or as a device watchdog.

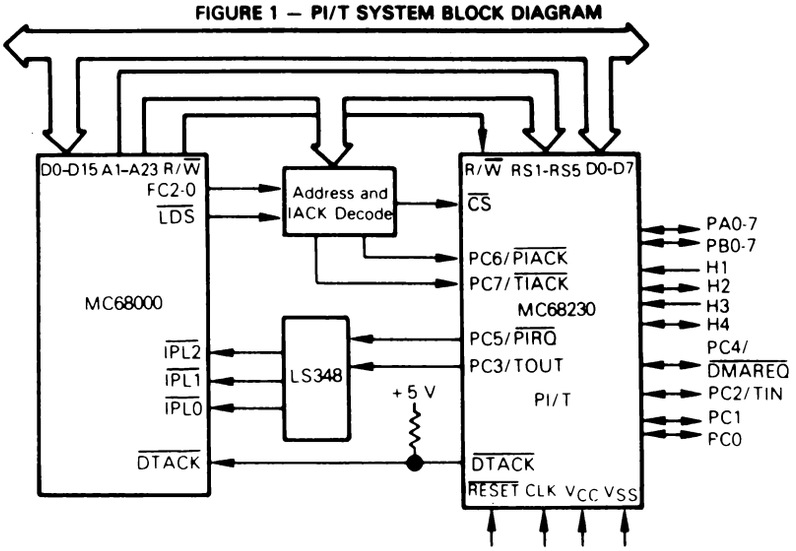
- MC68000 Bus Compatible
- Port Modes Include:
  - Bit I/O
  - Unidirectonal 8-Bit and 16-Bit
  - Bidirectional 8-Bit and 16-Bit
- Selectable Handshaking Options
- 24-Bit Programmable Timer
- Software Programmable Timer Modes
- Contains Interrupt Vector Generation Logic
- Separate Port and Timer Interrupt Service Requests
- Registers are Read/Write and Directly Addressable
- Registers are Addressed for MOVEP (Move Peripheral) and DMAC Compatibility

PIN ASSIGNMENT



This document contains information on a new product. Specifications and information herein are subject to change without notice

FIGURE 1 — PI/T SYSTEM BLOCK DIAGRAM



POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \bullet \theta_{JA}) \quad (1)$$

Where:

T_A = Ambient Temperature, °C

θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W

$P_D = P_{INT} + P_{PORT}$

$P_{INT} = I_{CC} \times V_{CC}$, Watts – Chip Internal Power

P_{PORT} = Port Power Dissipation, Watts – User Determined

For most applications $P_{PORT} \ll P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations 1 and 2 for K gives:

$$K = P_D \bullet (T_A + 273°C) + \theta_{JA} \bullet P_D^2 \quad (3)$$

Where K is a constant pertaining to the particular part. K can be determined from equation 3 by measuring P_D (at equilibrium) for a known T_A. Using this value of K the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

MAXIMUM RATINGS

Characteristics	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature	T_{stg}	−55 to +150	°C

THERMAL CHARACTERISTICS

Characteristics	Symbol	Value	Rating
Thermal Resistance Ceramic	θ_{JA}	50	°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precuations be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

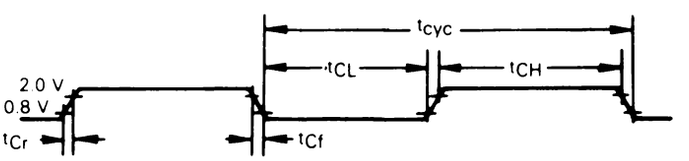
DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, T_A = 0 to 70°C unless otherwise noted)

Characteristics		Symbol	Min	Max	Unit
Input High Voltage	All Inputs	V_{IH}	V_{SS} + 2.0	V_{CC}	V
Input Low Voltage	All Inputs	V_{IL}	V_{SS} − 0.3	V_{SS} + 0.8	V
Input Leakage Current (V_{in} = 0 to 5.25 V)	H1, H3, R/$\overline{W}$, $\overline{RESET}$, CLK, RS1-RS5, $\overline{CS}$	I_{in}	–	10.0	μA
Three-State (Off State) Input Current (V_{in} = 0.4 to 2.4)	$\overline{DTACK}$, PC0-PC7, D0-D7 H2, H4, PA0-PA7, PB0-PB7	I_{TSI}	– −0.1	20 −1.0	μA mA
Output High Voltage (I_{Load} = −400 μA, V_{CC} = min) (I_{Load} = −150 μA, V_{CC} = min) (I_{Load} = −100 μA, V_{CC} = min)	$\overline{DTACK}$, D0-D7 H2, H4, PB0-PB7, PA0-PA7 PC0-PC7	V_{OH}	V_{SS} + 2.4	–	V
Output Low Voltage (I_{Load} = 8.8 mA, V_{CC} = min) (I_{Load} = 5.3 mA, V_{CC} = min) (I_{Load} = 2.4 mA, V_{CC} = min)	PC3/TOUT, PC5/$\overline{PIRQ}$ D0-D7, $\overline{DTACK}$ PA0-PA7, PB0-PB7, H2, H4, PC0-PC2, PC4, PC6, PC7	V_{OL}	–	0.5	V
Internal Power Dissipation (Measured at T_A = 0°C)		P_{INT}	–	500	mW
Input Capacitance (V_{in} = 0, T_A = 25°C, f = 1 MHz)		C_{in}	–	15	pF

CLOCK TIMING (See Figure 2)

Characteristic	Symbol	8 MHz MC68230L8		10 MHz MC68230L10		Unit
		Min	Max	Min	Max	
Frequency of Operation	f	2.0	8.0	2.0	10.0	MHz
Cycle Time	t_{cyc}	125	500	100	500	ns
Clock Pulse Width	t_{CL} t_{CH}	55 55	250 250	45 45	250 250	ns
Clock Rise and Fall Times	t_{Cr} t_{Cf}	– –	10 10	– –	10 10	ns

FIGURE 2 — INPUT CLOCK WAVEFORM



AC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, V_S = 0 Vdc, T_A = 0°C to 70°C)

Number	Characteristic	8 MHz MC68230L8		10 MHz MC68230L10		Unit
		Min	Max	Min	Max	
1	R/$\overline{W}$, RS1-RS5 Valid to $\overline{CS}$ Low (Setup Time)	0	–	0	–	ns
2(10)	$\overline{CS}$ Low to R/$\overline{W}$ and RS1-RS5 Invalid (Hold Time)	100	–	65	–	ns
3(1)	$\overline{CS}$ Low to CLK Low (Setup Time)	30	–	20	–	ns
4(2)	$\overline{CS}$ Low to data Out Valid (Delay)	–	75	–	60	ns
5	RS1-RS5 Valid to Data Out Valid (Delay)	–	140	–	100	ns
6	CLK Low to $\overline{DTACK}$ Low (Read/Write Cycle) (Delay)	0	70	0	60	ns
7(3)	DTACK Low to $\overline{CS}$ High (Hold Time)	0	–	0	–	ns
8	$\overline{CS}$ or $\overline{PIACK}$ or $\overline{TIACK}$ High to Data Out Invalid (Hold Time)	0	–	0	–	ns
9	$\overline{CS}$ or $\overline{PIACK}$ or $\overline{TIACK}$ High to D0-D7 High-Impedance (Delay)	–	50	–	45	ns
10	$\overline{CS}$ or $\overline{PIACK}$ or $\overline{TIACK}$ High to DTACK High (Delay)	–	50	–	30	ns
11	$\overline{CS}$ or $\overline{PIACK}$ or $\overline{TIACK}$ High to $\overline{DTACK}$ High Impedance (Delay)	–	100	–	55	ns
12	Data Invalid to $\overline{CS}$ Low (Setup Time)	0	–	0	–	ns
13	$\overline{CS}$ Low to Data In INvalid (Hold Time)	100	–	65	–	ns
14	Input Data Valid to H1(H3) Asserted (Setup Time)	100	–	60	–	ns
15	H1(H3) Asserted to Input Data Invalid (Hold Time)	20	–	20	–	ns
16	Handshake Input H1(H4) Pulse Width Asserted	40	–	40	–	ns
17	Handshake Input (H1-H4) Pulse Width Negated	40	–	40	–	ns
18	H1(H3) Asserted to H2(H4) Negated (Delay)	–	150	–	120	ns
19	CLK Low to H2(H4) Asserted (Delay)	–	100	–	100	ns
20(4)	H2(H4) Asserted to H1(H3) Asserted	0	–	0	–	ns
21(5)	CLK Low to H2(H4) Pulse Negated (Delay)	–	125	–	125	ns
22(9, 11)	Synchronized H1(H3) to CLK Low on which $\overline{DMAREQ}$ is Asserted (See Figures 13 and 14)	2.5	3.5	2.5	3.5	CLK Per
23	CLK Low $\overline{DMAREQ}$ is Asserted to CLK Low on which $\overline{DMAREQ}$ is Negated	3	3	3	3	CLK Per
24	CLK Low to Output Data Valid (Delay) (Modes 0, 1)	–	150	–	120	ns
25(9, 11)	Synchronized H1(H3) to Output Data Invalid (Modes 0, 1)	1.5	2.5	1.5	2.5	CLK Per
26	H1 Negated to Output Data Valid (Modes 2, 3)	–	70	–	50	ns
27	H1 Asserted to Output Data High Impedance (Modes 2, 3)	0	70	0	70	ns
28	Read Data Valid to $\overline{DTACK}$ Low (Setup Time)	0	–	0	–	ns
29	CLK Low to Data Output Valid (Interrupt Acknowledge Cycle)	–	120	–	100	ns
30(7)	H1(H3) Asserted to CLK High (Setup Time)	50	–	40	–	ns
31	$\overline{PIACK}$ or $\overline{TIACK}$ Low to CLK Low (Setup Time)	50	–	40	–	ns
32(11)	Synchronized $\overline{CS}$ to CLK Low on which $\overline{DMAREQ}$ is Asserted (See Figures 13 and 14)	3	3	3	3	CLK Per
33(9, 11)	Synchronized H1(H3) to CLK Low on which H2(H4) is Asserted	3.5	4.5	3.5	4.5	CLK Per
34	CLK Low to $\overline{DTACK}$ Low (Interrupt Acknowledge Cycle (Delay)	–	75	–	60	ns
35	CLK Low to $\overline{DMAREQ}$ Low (Delay)	0	120	0	100	ns
36	CLK Low to $\overline{DMAREQ}$ High (Delay)	0	120	0	100	ns
–	CLK Low to $\overline{PIRQ}$ Low or High Impedance	–	200	–	150	ns
– (8)	TIN Frequency (External Clock) – Prescaler Used	0	1	0	1	Fclk(Hz)(6)
–	TIN Frequency (External Clock) – Prescaler Not used	0	1/32	0	1/32	Fclk(Hz)(6)
–	TIN Pulse Width High or Low (External Clock)	55	–	45	–	ns
–	TIN Pulse Width Low (Run/Halt Control)	1	–	1	–	CLK
–	CLK Low to TOUT High, Low, or High Impedance	0	200	0	150	ns
–	$\overline{CS}$, $\overline{PIACK}$, or $\overline{TIACK}$ High to $\overline{CS}$, $\overline{PIACK}$, or $\overline{TIACK}$ Low	50	–	30	–	ns

NOTES

1. This specification only applies if the PI/T had completed all operations initiated by the previous bus cycle when $\overline{CS}$ was asserted. Following a normal read or write bus cycle, all operations are complete within three CLKs after the falling edge of the CLK pin on which $\overline{DTACK}$ was asserted. If $\overline{CS}$ is asserted prior to completion of these operations, the new bus cycle, and hence, $\overline{DTACK}$ is postponed.

   If all operations of the previous bus cycle were complete when $\overline{CS}$ was asserted, this specification is made only to insure that $\overline{DTACK}$ is asserted with respect to the falling edge of the CLK pin as shown in the timing diagram, not to guarantee operation of the part. If the $\overline{CS}$ setup time is violated, $\overline{DTACK}$ may be asserted as shown, or may be asserted one clock cycle later.

2. Assuming the RS1-RS5 to Data Valid time has also expired.

3. This specification imposes a lower bound on $\overline{CS}$ low time, guaranteeing that $\overline{CS}$ will be low for at least 1 CLK period.

4. This specification assures recognition of the asserted edge of H1(H3).

5. This specification applies only when a pulsed handshake option is chosen and the pulse is not shortened due to an early asserted edge of H1(H3).

6. CLK refers to the actual frequency of the CLK pin, not the maximum allowable CLK frequency.

7. If the setup time on the rising edge of the clock is violated, H1(H3) may not be recognized until the next rising of the clock.

8. This limit applies to the frequency of the signal at TIN compared to the frequency of the CLK signal during each clock cycle. If any period of the waveform at TIN is smaller than the period of the CLK signal at that instant, then it is likely that the timer circuit will completely ignore one cycle of the TIN signal.

   If these two signals are derived from different sources they will have different instantaneous frequency variations. In this case the frequency applied to the TIN pin must be distinctly less than the frequency at the CLK pin to avoid lost cycles of the TIN signal. With signals derived from different crystal oscillators applied to the TIN and CLK pins with fast rise and fall times, the TIN frequency can approach 80 to 90% of the frequency of the CLK signal without a loss of a cycle of the TIN signal.

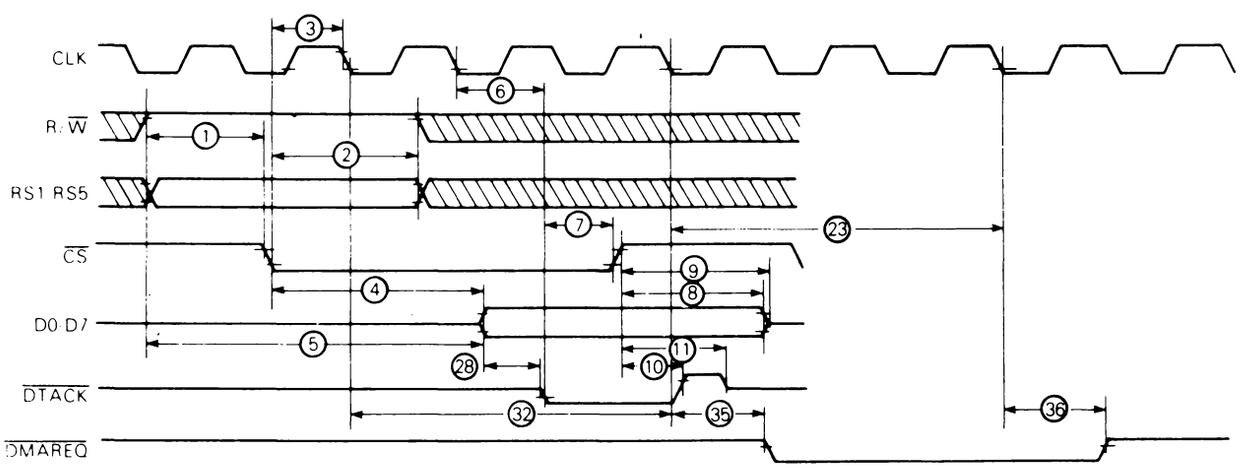
   If these two signals are derived from the same frequency source then the frequency of the signal applied to TIN can be 100% of the frequency at the CLK pin. They may be generated by different buffers from the same signal or one may be an inverted version of the other. The TIN signal may be generated by an 'AND' function of the clock and a control signal.

9. The maximum value is caused by a peripheral access (H1(H3) asserted) and bus access ($\overline{CS}$ asserted) occurring at the same time.

10. See BUS INTERFACE CONNECTION section for exception.

11. Synchronized means that the input signal has been seen by the PI/T on the appropriate edge of the clock (rising edge for H1(H3) and falling edge for $\overline{CS}$). (Refer to the BUS INTERFACE CONNECTION section for the exception concerning $\overline{CS}$.)

FIGURE 3 — BUS READ CYCLE TIMING



NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

FIGURE 4 – BUS WRITE CYCLE TIMING

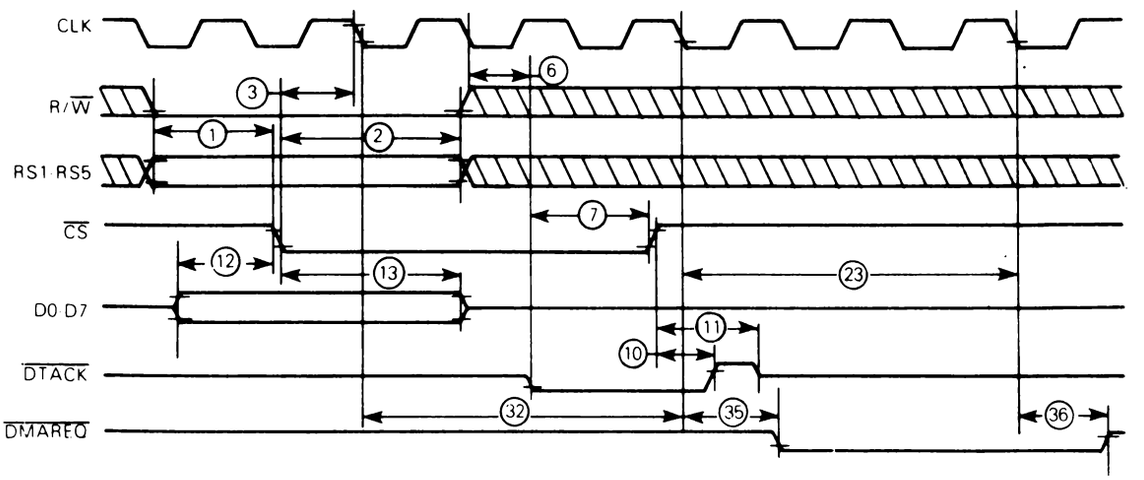
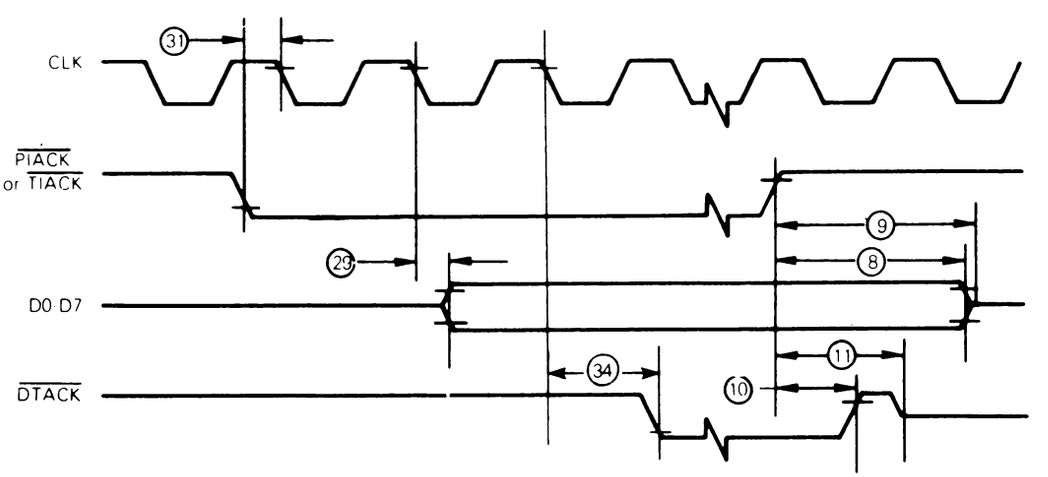
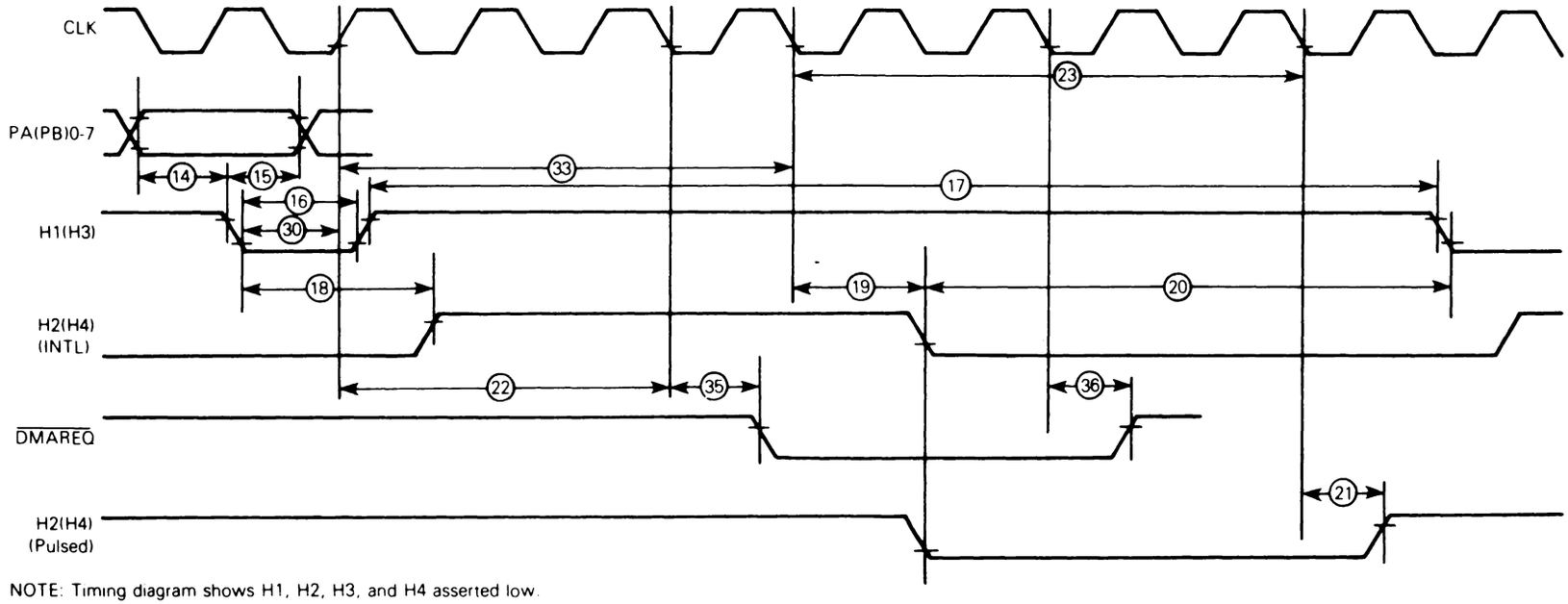


FIGURE 5 – INTERRUPT ACKNOWLEDGE
FUNCTIONAL TIMING DIAGRAM



Note Timing measurements are referenced to and from a low voltage of 0 8 volts and a high voltage of 2 0 volts, unless otherwise noted

FIGURE 6 – PERIPHERAL INTERFACE INPUT TIMING



NOTE: Timing diagram shows H1, H2, H3, and H4 asserted low.

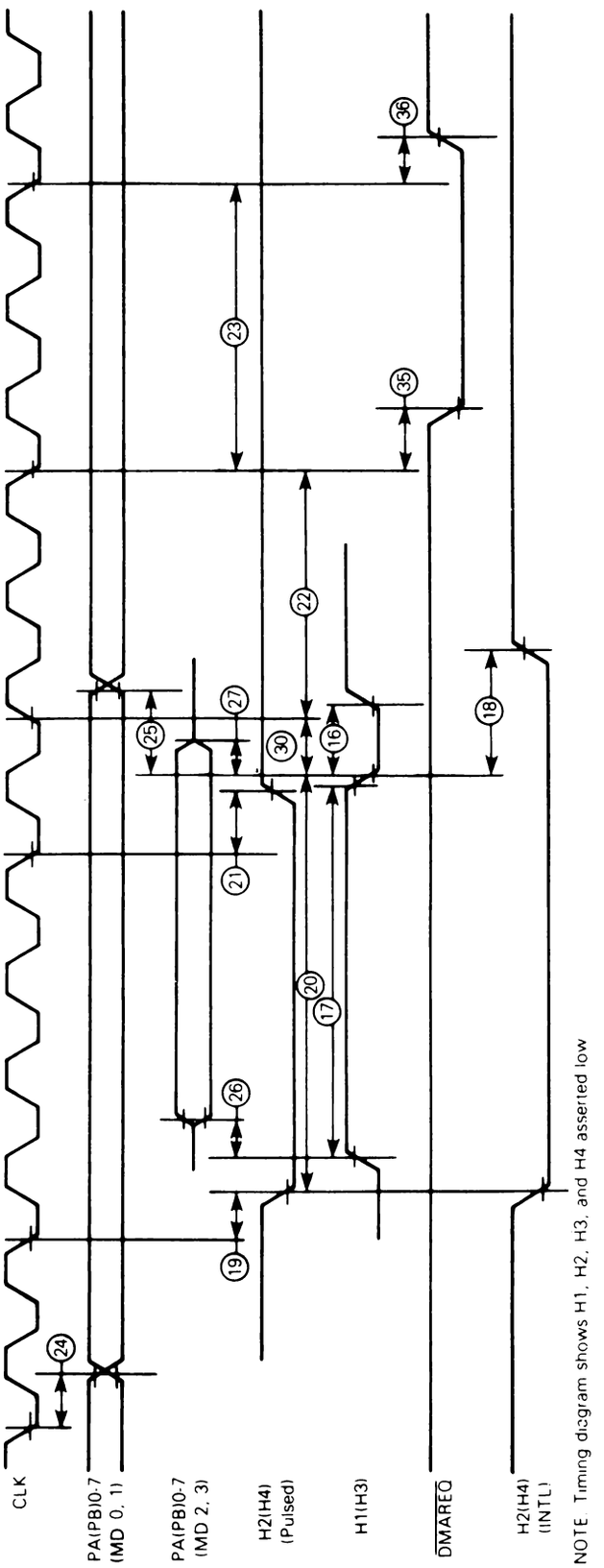


FIGURE 7 – PERIPHERAL INTERFACE OUTPUT TIMING

NOTE: Timing diagram shows H1, H2, H3, and H4 asserted low

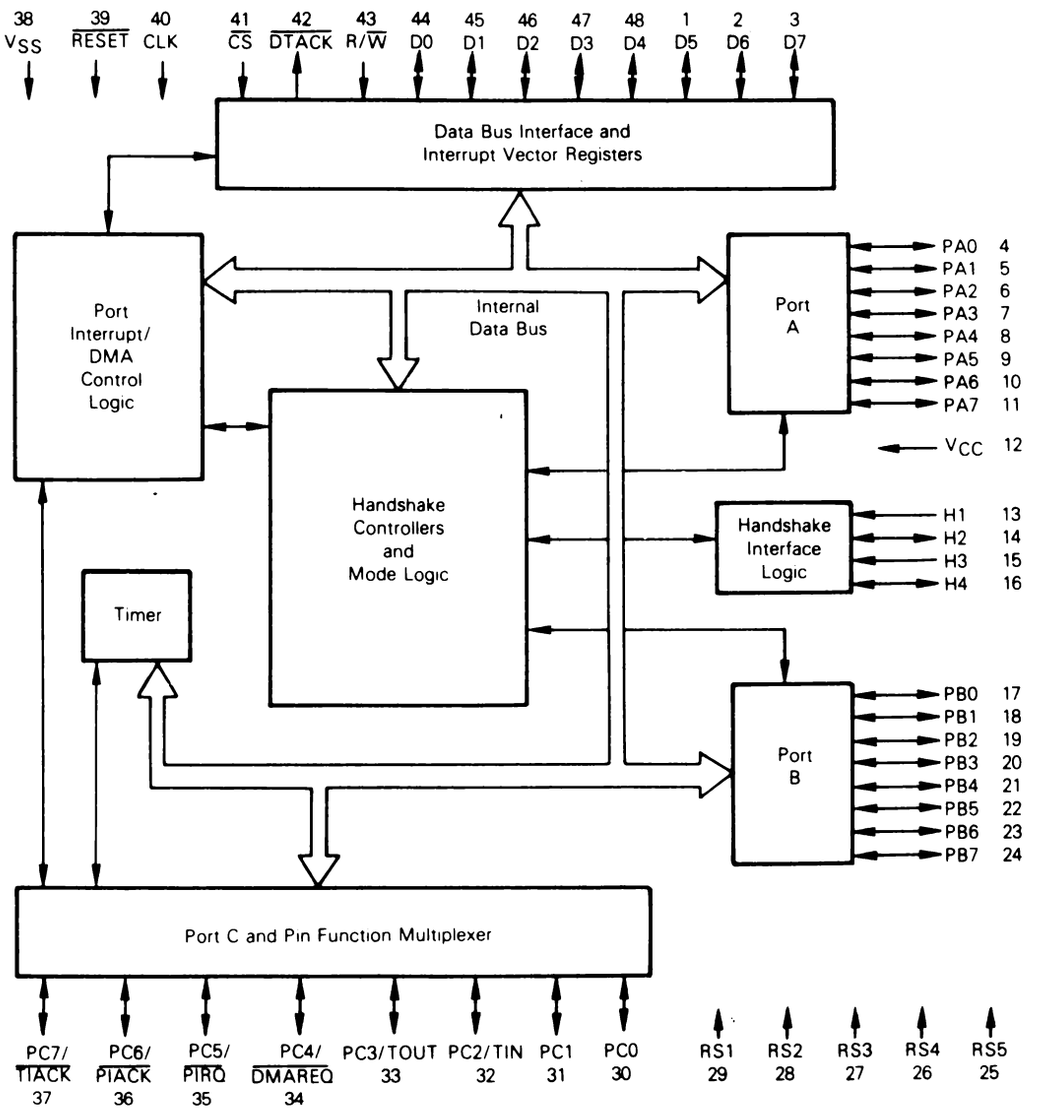
GENERAL DESCRIPTION

The PI/T consists of two logically independent sections: the ports and the timer. The port section consists of Port A (PA0-7), Port B (PB0-7), four handshake pins (H1, H2, H3, and H4), two general I/O pins, and six dual-function pins. The dual-function pins can individually operate as a third port (Port C) or an alternate function related to either Ports A and B, or the timer. The four programmable handshake pins, depending on the mode, can control data transfer to and from the ports, or can be used as interrupt generating inputs, or I/O pins.

The timer consists of a 24-bit counter, optionally clocked by a 5-bit prescaler. Three pins provide complete timer I/O: PC2/TIN, PC3/TOUT, and PC7/$\overline{\text{TIACK}}$. Of course, only the ones needed for the given configuration perform the timer function, while the others remain Port C I/O.

The system bus interface provides for asynchronous transfer of data from the PI/T to a bus master over the data bus (D0-D7). Data transfer acknowledge ($\overline{\text{DTACK}}$), register selects (RS1-RS5), chip select, the read/write line (R/$\overline{\text{W}}$), and Port Interrup Acknowledge ($\overline{\text{PIACK}}$) or Timer Interrupt Acknowledge ($\overline{\text{TIACK}}$) control data transfer between the PI/T and the MC68000.

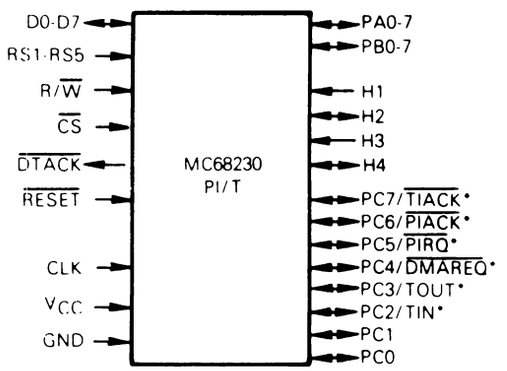
FIGURE 8 — MC68230 BLOCK DIAGRAM



PI/T PIN DESCRIPTION

Throughout the data sheet, signals are presented using the terms active and inactive or asserted and negated independent of whether the signal is active in the high-voltage state or low-voltage state. (The active state of each logic pin is given below.) Active low signals are denoted by a superscript bar. R/$\overline{W}$ indicates a write is active low and a read active high.

FIGURE 9 – LOGICAL PIN ASSIGNMENT



*Individually Programmable Dual-Function Pin

D0-D7 – Bidirectional Data Bus. The data bus pins D0-D7 form an 8-bit bidirectional data bus to/from the MC68000 or other bus master. These pins are active high.

RS1-RS5 – Register Selects. RS1-RS5 are active high high-impedance inputs that determine which of the 25 possible registers is being addressed. They are provided by the MC68000 or other bus master.

R/$\overline{W}$ – Read/Write Input – R/$\overline{W}$ is the high-impedance Read/Write signal from the MC68000 or bus master, indicating whether the current bus cycle is a read (high) or write (low) cycle.

$\overline{CS}$ – Chip Select Input. $\overline{CS}$ is a high-impedance input that selects the PI/T registers for the current bus cycle. Address strobe and the data strobe (upper or lower) of the bus master, along with the appropriate address bits, must be included in the chip select equation. A low level corresponds to an asserted chip select.

$\overline{DTACK}$ – Data Transfer Acknowledge Output. $\overline{DTACK}$ is an active low output that signals the completion of the bus cycle. During read or interrupt acknowledge cycles, $\overline{DTACK}$ is asserted by the MC68230 after data has been provided on the data bus; during write cycles it is asserted after data has been accepted at the data bus. Data transfer acknowledge is compatible with the MC68000 and with other Motorola bus masters such as the MC68450 DMA controller. A holding resistor is required to maintain $\overline{DTACK}$ high between bus cycles.

$\overline{RESET}$ – Reset Input. $\overline{RESET}$ is a high-impedance input used to initialize all PI/T functions. All control and data direction registers are cleared and most internal operations are disabled by the assertion of $\overline{RESET}$ (low).

CLK – Clock Input. The clock pin is a high-impedance TTL-compatible signal with the same specifications as the MC68000. The PI/T contains dynamic logic throughout, and hence this clock must not be gated off at any time. It is not necessary that this clock maintain any particular phase relationship with the MC68000 clock. It may be connected to an independent frequency source (faster or slower) as long as all bus specifications are met.

PA0-PA7 and PB0-PB7 – Port A and Port B. Ports A and B are 8-bit ports that may be concatenated to form a 16-bit port in certain modes. The ports may be controlled in conjunction with the handshake pins H1-H4. For stabilization during system power-up, Ports A and B have internal pullup resistors to V_{CC}. All port pins are active high.

H1-H4 – Handshake pins (I/O depending on the Mode and Submode). Handshake pins H1-H4 are multi-purpose pins that (depending on the operational mode) may provide an interlocked handshake, a pulsed handshake, an interrupt input (independent of data transfers), or simple I/O pins. For stabilization during system power-up, H2 and H4 have internal pullup resistors to V_{CC}. Their sense (active high or low) may be programmed in the Port General Control Register bits 3-0. Independent of the mode, the instantaneous level of the handshake pins can be read from the Port Status Register.

Port C – (PC0-PC7/Alternate function). This port can be used as eight general purpose I/O pins (PC0-PC7) or any combination of six special function pins and two general purpose I/O pins (PC0-PC1). (Each dual function pin can be standard I/O or a special function independent of the other port C pins.) The dual function pins are defined in the following paragraphs. When used as a port C pin, these pins are active high. They may be individually programmed as inputs or outputs by the Port C Data Direction Register.

The alternate functions (TIN, TOUT, and $\overline{TIACK}$) are timer I/O pins. TIN may be used as a rising-edge triggered external clock input or an external run/halt control pin (the timer is in the run state if run/halt is high and in the halt state if run/halt is low). TOUT may provide an active low timer interrupt request output or a general-purpose square-wave output, initially high. $\overline{TIACK}$ is an active low high-impedance input used for timer interrupt acknowledge.

Port A and B functions have an independent pair of active low interrupt request ($\overline{PIRQ}$) and interrupt acknowledge ($\overline{PIACK}$) pins.

The $\overline{DMAREQ}$ (Direct Memory Access Request) pin provides an active low Direct Memory Access Controller (DMAC) request pulse of 3 clock cycles, completely compatible with the MC68450 DMAC.

REGISTER MODEL

A register model that includes the corresponding Register Selects is shown in Table 1.

TABLE 1 — REGISTER MODEL

Register Select Bits 5	4	3	2	1	7	6	5	4	3	2	1	0	
0	0	0	0	0	Port Mode Control		H34 Enable	H12 Enable	H4 Sense	H3 Sense	H2 Sense	H1 Sense	Port General Control Register
0	0	0	0	1	*	SVCRQ Select		Interrupt PFS		Port Interrupt Priority Control			Port Service Request Register
0	0	0	1	0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port A Data Direction Register
0	0	0	1	1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port B Data Direction Register
0	0	1	0	0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port C Data Direction Register
0	0	1	0	1	Interrupt Vector Number						*	*	Port Interrupt Vector Register
0	0	1	1	0	Port A Submode		H2 Control			H2 Int Enable	H1 SVCRQ Enable	H1 Stat Ctrl	Port A Control Register
0	0	1	1	1	Port B Submode		H4 Control			H4 Int Enable	H3 SVCRQ Enable	H3 Stat Ctrl	Port B Control Register
0	1	0	0	0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port A Data Register
0	1	0	0	1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port B Data Register
0	1	0	1	0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port A Alternate Register
0	1	0	1	1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port B Alternate Register
0	1	1	0	0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Port C Data Register
0	1	1	0	1	H4 Level	H3 Level	H2 Level	H1 Level	H4S	H3S	H2S	H1S	Port Status Register
0	1	1	1	0	*	*	*	*	*	*	*	*	(null)
0	1	1	1	1	*	*	*	*	*	*	*	*	(null)
1	0	0	0	0	TOUT/$\overline{\text{TIACK}}$ Control			Z D Ctrl	*	Clock Control		Timer Enable	Timer Control Register
1	0	0	0	1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Timer Interrupt Vector Register
1	0	0	1	0	*	*	*	*	*	*	*	*	(null)
1	0	0	1	1	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Counter Preload Register (High)
1	0	1	0	0	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	(Mid)
1	0	1	0	1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	(Low)
1	0	1	1	0	*	*	*	*	*	*	*	*	(null)
1	0	1	1	1	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Count Register (High)
1	1	0	0	0	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	(Mid)
1	1	0	0	1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	(Low)
1	1	0	1	0	*	*	*	*	*	*	*	ZDS	Timer Status Register
1	1	0	1	1	*	*	*	*	*	*	*	*	(null)
1	1	1	0	0	*	*	*	*	*	*	*	*	(null)
1	1	1	0	1	*	*	*	*	*	*	*	*	(null)
1	1	1	1	0	*	*	*	*	*	*	*	*	(null)
1	1	1	1	1	*	*	*	*	*	*	*	*	(null)

*Unused, read as zero.

PORT CONTROL STRUCTURE

The primary focus of most applications will be on Ports A and B, the handshake pins, the port interrupt pins, and the DMA request pin. They are controlled in the following way: the Port General Control Register contains a 2-bit field that specifies a set of four operation modes. These govern the overall operation of the ports and determine their interrelationships. Some modes require additional information from each port's control register to further define its operation. In each port control register, there is a 2-bit submode field that serves this purpose. Each port mode/submode combination specifies a set of programmable characteristics that fully define the behavior of that port and two of the handshake pins. This structure is summarized in Table 2 and Figure 10.

FIGURE 10 – PORT MODE LAYOUT

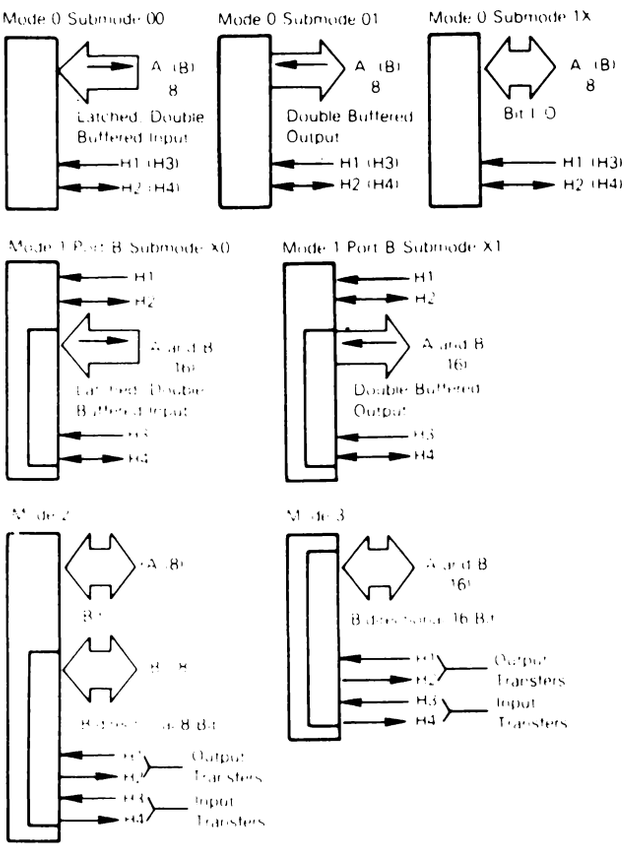


TABLE 2 – PORT MODE CONTROL SUMMARY

Mode 0 (Unidirectional 8-Bit Mode)
- Port A
  - Submode 00 – Double-Buffered Input
    - H1 – Latches input data
    - H2 – Status/interrupt generating input, general-purpose output, or operation with H1 in the interlocked or pulsed input handshake protocols
  - Submode 01 – Double-Buffered Output
    - H1 – Indicates data received by peripheral
    - H2 – Status/interrupt generating input, general-purpose output, or operation with H1 in the interlocked or pulsed output handshake protocols
  - Submode 1X – Bit I/O
    - H1 – Status/interrupt generating input
    - H2 – Status/interrupt generating input or general-purpose output
- Port B, H3 and H4 – Identical to Port A, H1 and H2

Mode 1 (Unidirectional 16-Bit Mode)
- Port A – Double-Buffered Data (Most significant)
  - Submode XX (not used)
    - H1 – Status/interrupt generating input
    - H2 – Status/interrupt generating input or general purpose output
- Port B – Double-Buffered Data (Least significant)
  - Submode X0 – Unidirectional 16-Bit Input
    - H3 – Latches input data
    - H4 – Status/interrupt generating input, general-purpose output, or operation with H3 in the interlocked or pulsed input handshake protocols
  - Submode X1 – Unidirectional 16-Bit Output
    - H3 – Indicates data received by peripheral
    - H4 – Status/interrupt generating input, general-purpose output, or operation with H3 in the interlocked or pulsed output handshake protocols

Mode 2 (Bidirectional 8-Bit Mode)
- Port A – Bit I/O (with no handshaking pins)
  - Submode XX (not used)
- Port B – Bidirectional 8-Bit Data (Double Buffered)
  - Submode XX (not used)
    - H1 – Indicates output data received by peripheral
    - H2 – Operation with H1 in the interlocked or pulsed output handshake protocols
    - H3 – Latches input data
    - H4 – Operation with H3 in the interlocked or pulsed input handshake protocols

Mode 3 (Bidirectional 16-Bit Mode)
- Port A – Double-Buffered Data (Most significant)
  - Submode XX (not used)
- Port B – Double-Buffered Data (Least significant)
  - Submode XX (not used)
- H1 – Indicates output data received by peripheral
- H2 – Operation with H1 in the interlocked or pulsed output handshake protocols
- H3 – Latches input data
- H4 – Operation with H3 in the interlocked or pulsed input handshake protocols

PORT GENERAL INFORMATION AND CONVENTIONS

The following paragraphs introduce concepts that are generally applicable to the PI/T ports independent of the chosen mode and submode. For this reason, no particular port or handshake pins are mentioned; the notation H1 (H3) indicates that, depending on the chosen mode and submode, the statement given may be true for either the H1 or H3 handshake pin.

Unidirectional vs Bidirectional – Figure 10 shows the configuration of Ports A and B and each of the handshake pins in each port mode and submode. In Modes 0 and 1, a data direction register is associated with each of the ports. These registers contain one bit for each port pin to determine whether that pin is an input or an output. Modes 0 and 1 are, thus, called unidirectional modes because each pin assumes a constant direction, changeable only by a reset condition or a programming change. These modes allow double-buffered data transfers in one direction. This direction, determined by the mode and submode definition, is known as the primary direction. Data transfers in the primary direction are controlled by the handshake pins. Data transfers not in the primary direction are generally unrelated, and single or unbuffered data paths exist.

In Modes 2 and 3 there is no concept of primary direction as in Modes 0 and 1. Except for Port A in Mode 2 (Bit I/O), the data direction registers have no effect. These modes are bidirectional, in that the direction of each transfer (always 8 or 16 bits, double-buffered) is determined dynamically by the state of the handshake pins. Thus, for example, data may be transferred out of the ports, followed very shortly by a transfer into the same port pins. Transfers to and from the ports are independent and may occur in any sequence. Since the instantaneous direction is always determined by the external system, a small amount of arbitration logic may be required.

Control of Double-Buffered Data Paths – Generally speaking, the PI/T is a double-buffered device. In the primary direction, double-buffering allows orderly transfers by using the handshake pins in any of several programmable protocols. (When Bit I/O is used, double-buffering is not available and the handshake pins are used as outputs or status/interrupt inputs.)

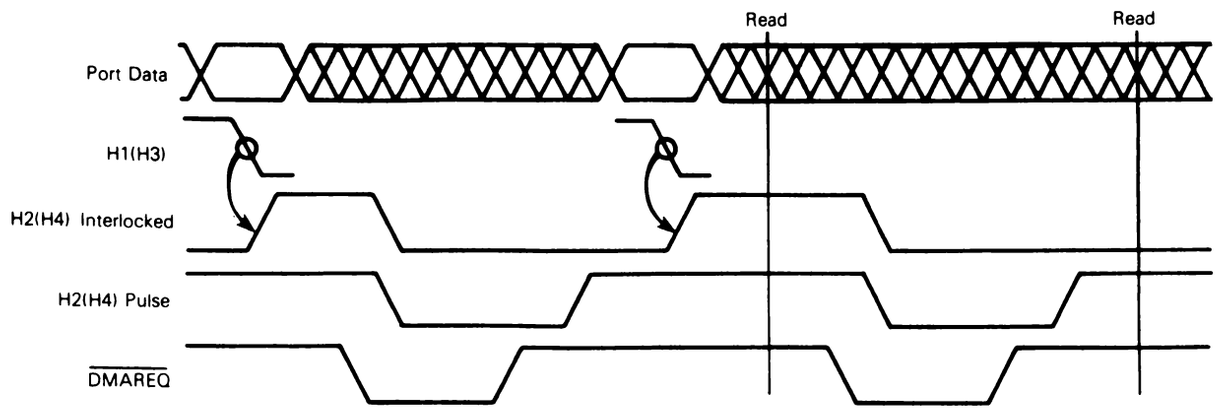
Use of double-buffering is most beneficial in situations where a peripheral device and the computer system are capable of transferring data at roughly the same speed. Double-buffering allows the fetch operation of the data transmitter to be overlapped with the store operation of the data receiver. Thus, throughput measured in bytes or words-per-second may be greatly enhanced. if there is a large mismatch in transfer capability between the computer and the peripheral, little or no benefit is obtained. In these cases there is no penalty in using double-buffering.

Double-Buffered Input Transfers – In all modes, the PI/T supports double-buffered input transfers. Data that meets the port setup and hold times is latched on the asserted edge of H1(H3). H1(H3) is edge-sensitive, and may assume any duty-cycle as long as both high and low minimum times are observed. The PI/T contains a Port Status Register whose H1S(H3S) status bit is set anytime any input data is present in the double-buffered latches that has not been read by the bus master. The action of H2(H4) is programmable; it may indicate whether there is room for more data in the PI/T latches or it may serve other purposes. The following options are available, depending on the mode.

1. H2(H4) may be an edge-sensitive input that is independent of H1(H3) and the transfer of port data. On the asserted edge of H2(H4), the H2S(H4S) status bit is set. It is cleared by the direct method (refer to Direct Method of Resetting Status), the $\overline{\text{RESET}}$ pin being asserted, or when the H12 Enable (H34 Enable) bit of the Port General Control Register is 0.
2. H2(H4) may be a general purpose output pin that is always negated. The H2S(H4S) status bit is always 0.
3. H2(H4) may be a general purpose output pin that is always asserted. The H2S(H4S) status bit is always 0.
4. H2(H4) may be an output pin in the interlocked input handshake protocol. It is asserted when the port input latches are ready to accept new data. It is negated asynchronously following the asserted edge of the H1(H3) input. As soon as the input latches become ready, H2(H4) is again asserted. When the input double-buffered latches are full, H2(H4) remains negated until data is removed. Thus, anytime the H2(H4) output is asserted, new input data may be entered by asserting H1(H3). At other times transitions on H1(H3) are ignored. The H2S(H4S) status bit is always 0. When H12 Enable (H34 Enable) is 0, H2(H4) is held negated.
5. H2(H4) may be an output pin in the pulsed input handshake protocol. It is asserted exactly as in the interlocked input protocol, but never remains asserted longer than 4 clock cycles. Typically, a four clock cycle pulse is generated. But in the case that a subsequent H1(H3) asserted edge occurs before termination of the pulse, H2(H4) is negated asynchronously. Thus, anytime after the leading edge of the H2(H4) pulse, new data may be entered in the PI/T double-buffered input latches. The H2S(H4S) status bit is always 0. When H12 Enable (H34 Enable) is 0, H2(H4) is held negated.

A sample timing diagram is shown in Figure 11. The H2(H4) interlocked and pulsed input handshake protocols are shown. The $\overline{\text{DMAREQ}}$ pin is also shown assuming it is enabled. All handshake pin sense bits are assumed to be 0 (refer to Port General Control Register); thus, the pins are in the low state when asserted. Due to the great similarity between modes, this timing diagram is applicable to all double-buffered input transfers.

FIGURE 11 — DOUBLE-BUFFERED INPUT TRANSFERS

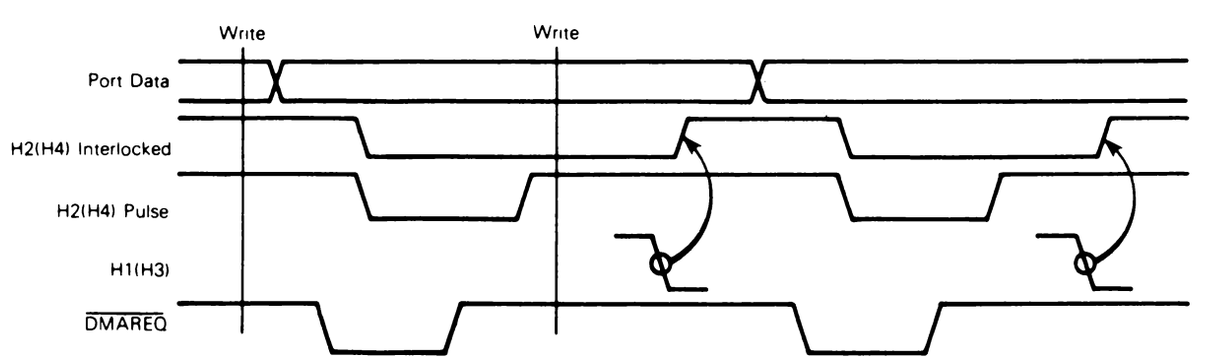


Double-Buffered Output Transfers — The PI/T supports double-buffered output transfers in all modes. Data, written by the bus master to the PI/T, is stored in the port's output latch. The peripheral accepts the data by asserting H1(H3), which causes the next data to be moved to the port's output latch as soon as it is available. The function of H2(H4) is programmable; it may indicate whether new data has been moved to the output latch or it may serve other purposes. The H1S(H3S) status bit may be programmed for two interpretations. Normally the status bit is a 1 when there is at least one latch in the double-buffered data path that can accept new data. After writing one byte/word of data to the ports, an interrupt service routine could check this bit to determine if it could store another byte/word; thus, filling both latches. When the bus master is finished, it is often useful to be able to check whether all of the data has been transferred to the peripheral. The H1S(H3S) Status Control bit of the Port A and B Control Registers provide this flexibility. The programmable options of the H2(H4) pin are given below, depending on the mode.

1. H2(H4) may be an edge-sensitive input pin independent of H1(H3) and the transfer of port data. On the asserted edge of H2(H4), the H2S(H4S) status bit is set. It is reset by the direct method (refer to Direct Method of Resetting Status), the $\overline{\text{RESET}}$ pin being asserted, or when the H12 Enable (H34 Enable) bit of the Port General Control Register is 0.
2. H2(H4) may be a general-purpose output pin that is always negated. The H2S(H4S) status bit is always 0.
3. H2(H4) may be a general-purpose output pin that is always asserted. The H2S(H4S) status bit is always 0.
4. H2(H4) may be an output pin in the interlocked output handshake protocol. H2(H4) is asserted two clock cycles after data is transferred to the double-buffered output latches. The data remains stable and H2(H4) remains asserted until the next asserted edge of the H1(H3) input. At that time, H2(H4) is asynchronously negated. As soon as the next data is available, it is transferred to the output latches. When H2(H4) is negated, asserted transitions on H1(H3) have no effect on the data paths. As is explained later, however, in Modes 2 and 3 they do control the three-state output buffers of the bidirectional port(s). The H2S(H4S) status bit is always 0. When H12 Enable (H34 Enable) is 0, H2(H4) is held negated.
5. H2(H4) may be an output pin in the pulsed output handshake protocol. It is asserted exactly as in the interlocked output protocol above, but never remains asserted longer than four clock cycles. Typically, a four clock pulse is generated. But in the case that a subsequent H1(H3) asserted edge occurs before termination of the pulse, H2(H4) is negated asynchronously shortening the pulse. The H2S(H4S) status bit is always 0. When H12 Enable (H34 Enable) is 0 H2(H4) is held negated.

A sample timing diagram is shown in Figure 12. The H2(H4) interlocked and pulsed output handshake protocols are shown. The $\overline{\text{DMAREQ}}$ pin is also shown assuming it is enabled. All handshake pin sense bits are assumed to be 0, thus, the pins are in the low state when asserted. Due to the great similarity between modes, this timing diagram is applicable to all double-buffered output transfer.

FIGURE 12 — DOUBLE-BUFFERED OUTPUT TRANSFERS



Requesting Bus Master Service – The PI/T has several means of indicating a need for service by a bus master. First, the processor may poll the Port Status Register. It contains a status bit for each handshake pin, plus a level bit that always reflects the instantaneous state of that handshake pin. A status bit is 1 when the PI/T needs servicing, i.e., generally when the bus master needs to read or write data to the ports, or when a handshake pin used as a simple status input has been asserted. The interpretation of these bits is dependent on the chosen mode and submode.

Second, the PI/T may be placed in the processor's interrupt structure. As mentioned previously, the PI/T contains Port A and B Control Registers that configure the handshake pins. Other bits in these registers enable an interrupt associated with each handshake pin. This interrupt is made available through the PC5/$\overline{\text{PIRQ}}$ pin, if the $\overline{\text{PIRQ}}$ function is selected. Three additional conditions are required for $\overline{\text{PIRQ}}$ to be asserted: (1) the handshake pin status bit set, (2) the corresponding interrupt (service request) enable bit is set, (3) and DMA requests are not associated with that data transfer (H1 and H3 only). The conditions from each of the four handshake pins and corresponding status bits are ORed to determine $\overline{\text{PIRQ}}$.

The third method of requesting service is via the PC4/$\overline{\text{DMAREQ}}$ pin. This pin can be associated with double-buffered transfers in each mode. If it is used as a DMA controller request, it can initiate requests to keep the PI/T's input/output double-buffering empty/full as much as possible. It will not overrun the DMA controller. The pin is compatible with the MC68450 Direct Memory Access Controller (DMAC).

Vectored, Prioritized Port Interrupts – Use of MC68000-compatible vectored interrupts with the PI/T requires the $\overline{\text{PIRQ}}$ and $\overline{\text{PIACK}}$ pins. When $\overline{\text{PIACK}}$ is asserted, the PI/T places an 8-bit vector on the data pins D0-D7. Under normal conditions, this vector corresponds to highest priority, enabled, active port interrupt source with which the $\overline{\text{DMAREQ}}$ pin is not currently associated. The most-significant six bits are provided by the Port Interrupt Vector Register (PIVR), with the lower two bits supplied by prioritization logic according to conditions present when $\overline{\text{PIACK}}$ is asserted. It is important to note that the only affect on the PI/T caused by interrupt acknowledge cycles is that the vector is placed on the data bus. Specifically, no registers, data, status, or other internal states of the PI/T are affected by the cycle.

Several conditions may be present when the $\overline{\text{PIACK}}$ input is asserted to the PI/T. These conditions affect the PI/T's response and the termination of the bus cycle. If the PI/T has no interrupt function selected, or is not asserting $\overline{\text{PIRQ}}$, the PI/T will make no response to $\overline{\text{PIACK}}$ ($\overline{\text{DTACK}}$ will not be asserted). If the PI/T is asserting $\overline{\text{PIRQ}}$ when $\overline{\text{PIACK}}$ is received, the PI/T will output the contents of the Port Interrupt Vector Register and the prioritization bits. If the PIVR has not been initialized, $0F will be read from this register. These conditions are summarized in Table 3.

TABLE 3 – RESPONSE TO PORT INTERRUPT ACKNOWLEDGE

Conditions	$\overline{\text{PIRQ}}$ negated OR interrupt request function not selected	$\overline{\text{PIRQ}}$ asserted
PIVR has not been initialized since $\overline{\text{RESET}}$	No response from PI/T. No $\overline{\text{DTACK}}$.	PI/T provides $0F, the Uninitialized Vector.*
PIVR has been initialized since $\overline{\text{RESET}}$	No response from PI/T. No $\overline{\text{DTACK}}$.	PI/T provides PIVR contents with prioritization bits.

*The uninitialized vector is the value returned from an interrupt vector register before it has been initialized.

The vector table entries for the PI/T appear as a contiguous block of four vector numbers whose common upper six bits are programmed in the PIVR. The following table pairs each interrupt source with the 2-bit value provided by the prioritization logic, when interrupt acknowledge is asserted.

H1 source – 00
H2 source – 01
H3 source – 10
H4 source – 11

Autovectored Port Interrupts – Autovecored interrupts use only the $\overline{\text{PIRQ}}$ pin. The operation of the PI/T with vectored and autovectored interrupts is identical except that no vectors are supplied and the PC6/$\overline{\text{PIACK}}$ pin can be used as a Port C pin.

Direct Method of Resetting Status – In certain modes one or more handshake pins can be used as edge-sensitive inputs for sole purpose of setting bits in the Port Status Register. These bits consist of simple flip-flops. They are set (to 1) by the occurrence of the asserted edge of the handshake pin input. Resetting a handshake status bit can be done by writing an 8-bit mask to the Port Status Register. This is called the direct method of resetting. To reset a status bit that is resettable by the direct method, the mask must contain a 1 in the bit position of the Port Status Register corresponding to the desired bit. Other positions must contain 0's. For status bits that are not resettable by the direct method in the chosen mode, the data written to the port status register has no effect. For status bits that are resettable by the direct method in the chosen mode, a 0 in the mask has no effect.

Handshake Pin Sense Control – The PI/T contains exclusive-OR gates to control the sense of each of the handshake pins, whether used as inputs or outputs. Four bits in the Port General Control Register may be programmed to determine whether the pins are asserted in the low or high voltage state. As with other control registers, these bits are reset to 0 when the $\overline{\text{RESET}}$ pin is asserted, defaulting the asserted level to be low.

Enabling Ports A and B – Certain functions involved with double-buffered data transfers, the handshake pins, and the status bits, may be disabled by the external system or by the

programmer during initialization. The Port General Control Register contains two bits, H12 Enable and H34 Enable, which control these functions. These bits are cleared to the o state when the $\overline{\text{RESET}}$ pin is asserted, and the functions are disabled. The functions are the following:

1. Independent of other actions by the bus master or peripheral (via the handshake pins), the PI/T's disabled handshake controller is held to the "empty" state, i.e., no data is present in the double-buffered data path.
2. When any handshake pin is used to set a simple status flip-flop, unrelated to double-buffered transfers, these flip-flops are held reset to 0. (See Table 2.)
3. When H2(H4) is used in an interlocked or pulsed handshake with H1(H3), H2(H4) is held negated, regardless of the chosen mode, submode, and primary direction. Thus, for double-buffered input transfers, the programmer may signal a peripheral when the PI/T is ready to begin transfers by setting the associated handshake enable bit to 1.

The Port A and B Alternate Registers – In addition to the Port A and B Data Registers, the PI/T contains Port A and B Alternate Registers. These registers are read-only, and simply provide the instantaneous level of each port pin. They have no effect on the operation of the handshake pins, double buffered transfers, status bits, or any other aspect of the PI/T, and they are mode/submode independent.

PORT MODES

This section contains information that distinguishes the various port modes and submodes. General characteristics, common to all modes, have been defined previously.

MODE 0 — UNIDIRECTIONAL 8-BIT MODE

In Mode 0, Ports A and B operate independently. Each may be configured in any of its three possible submodes:

Submode 00 – Double-Buffered Input

Submode 01 – Double-Buffered Output

Submode 1X – Bit I/O

Handshake pins H1 and H2 are associated with Port A and configured by programming the Port A Control Register. (The H12 Enable bit of the Port General Control Register enables Port A transfers.) Handshake pins H3 and H4 are associated with Port B and configured by programming the Port B Control Register. (The H34 Enable bit of the Port General Control Register enables Port B transfers.) The Port A and B Data Direction Registers operate in all three submodes. Along with the submode, they affect the data read and written at the associated data register according to Table 4. They also enable the output buffer associated with each port pin. The $\overline{\text{DMAREQ}}$ pin may be associated with either (not both) Port A or Port B, but does not function if the Bit I/O submode is programmed for the chosen port.

TABLE 4 — MODE 0 PORT DATA PATHS

Mode	Read Port A/B Data Register		Write Port A/B Data Register	
	DDR = 0	DDR = 1	DDR = X	
0 Submode 00	FIL, D B	FOL Note 3	FOL, S B	Note 1
0 Submode 01	Pin	FOL Note 3	IOL/FOL, D B	Note 2
0 Submode 1X	Pin	FOL Note 3	FOL, S B	Note 1

Abbreviations

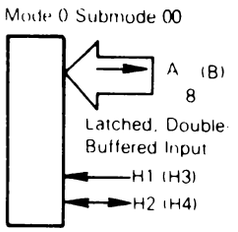
IOL – Initial Output Latch
FOL – Final Output Latch
FIL – Final Input Latch
S B – Single Buffered
D B – Double Buffered
DDR – Data Direction Register

Note 1 Data is latched in the output data registers (final output latch) and will be single buffered at the pin if the DDR is 1. The output buffers will be turned off if the DDR is 0.

Note 2 Data is latched in the double-buffered output data registers. The data in the final output latch will appear on the port pin if the DDR is a 1.

Note 3 The output drivers that connect the final output latch to the pins are turned on.

Port A or B Submode 00 (8-Bit Double-Buffered Input) –

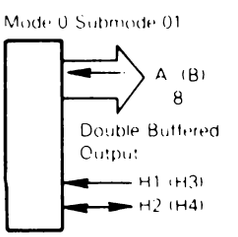


In Mode 0, double-buffered input transfers of up to 8-bits are available by programming Submode 00 in the desired port's control register. The operation of H2 and H4 may be selected by programming the Port A and Port B Control Registers, respectively. All five double-buffered input handshake options, previously mentioned in the Port General Information and Conventions section, are available.

For pins used as outputs, the data path consists of a single latch driving the output buffer. Data written to the port's data register does not affect the operation of any handshake pin, status bit, or any other aspect of the PI/T. Output pins may be used independently of the input transfer. However, read bus cycles to the data register do remove data from the port. Therefore, care should be taken to avoid processor instructions that perform unwanted read cycles.

Refer to PARALLEL PORTS Double-Buffered Input Transfers for a sample timing diagram (Figure 11).

Port A or B Submode 01 (8-Bit Double-Buffered Output) –

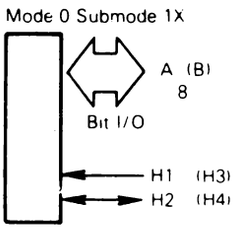


In Mode 0, double-buffered output transfers of up to 8 bits are available by programming submode 01 in the desired port's control register. The operation of H2 and H4 may be selected by programming the Port A and Port B Control Registers, respectively. All five double-buffered output handshake options, previously mentioned in the Port General Information and Conventions section, are available.

For pins used as inputs, data written to the associated data register is double-buffered and passed to the initial or final output latch, as usual, but the output buffer is disabled.

Refer to PARALLEL PORTS Double-Buffered Output Transfers for a sample timing diagram (Figure 12).

Port A or B Submode 1X (Bit I/O) –



In Mode 0, simple Bit I/O is available by programming Submode 1X in the desired port's control register. This submode is intended for applications in which several independent devices must be controlled or monitored. Data written to the associated data register is single-buffered. If the data direction register bit for that pin is a 1 (output), the output buffer is enabled. If it is 0 (input), data written is still latched, but is not available at the pin. Data read from the data register is the instantaneous value of the pin or what was written to the data register, depending on the contents of the data direction register. H1(H3) is an edge-sensitive status input pin only and it controls no data-related function. The H1S(H3S) status bit is set following the asserted edge of the input waveform. It is reset by the direct method, the $\overline{\text{RESET}}$ pin being asserted, or when the H12 Enable (H34 Enable) bit is 0.

H2(H4) can be programmed as a simple status input (identical to H1(H3)), or as an asserted or negated output. The interlocked or pulsed handshake configurations are not available.

MODE 1 — UNIDIRECTIONAL 16-BIT MODE

In Mode 1, Ports A and B are concatenated to form a single 16-bit port. The Port B Submode field controls the configuration of both ports. The possible submodes are:

Port B Submode X0 – Double-Buffered Input
Port B Submode X1 – Double-Buffered Output

Handshake pins H3 and H4, configured by programming the Port B Control Register, are associated with the 16-bit double-buffered transfer. These 16-bit transfers, are enabled by the H34 Enable bit of the Port General Control Register. Handshake pins H1 and H2 may be used as simple status inputs not related to the 16-bit data transfer or H2 may be an output. Enabling of the H1 and H2 handshake pins is done by the H12 Enable bit of the Port General Control Register. The Port A and B Data Direction Registers operate in each submode. Along with the submode, they affect the data read and written at the data register according to Table 5. They also enable the output buffer associated with each port pin. The $\overline{\text{DMAREQ}}$ pin may be associated only with H3.

Mode 1 can provide convenient, high-speed 16-bit transfers. The Port A and B data registers are addressed for compatibility with the MC68000 Move Peripheral (MOVEP) instruction and with the MC68450 DMAC. To take advantage of this, Port A should contain the most-significant byte of data and always be read or written by the bus master first. The interlocked and pulsed handshake protocols are keyed to accesses to the Port B Data Register in Mode 1. If it is accessed last, the 16-bit double-buffered transfers proceed smoothly.

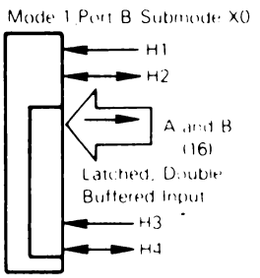
TABLE 5 — MODE 1 PORT DATA PATHS

Mode	Read Port A/B Register		Write Port A/B Register	
	DDR = 0	DDR = 1	DDR = 0	DDR = 1
1, Port B Submode X0	FIL, D.B.	FOL Note 3	FOL, S.B. Note 2	FOL, S.B. Note 2
1, Port B Submode X1	Pin	FOL Note 3	IOL/FOL, D.B., Note 1	IOL/FOL, D.B., Note 1

Note 1: Data written to Port A goes to a temporary latch. When the Port B data register is later written, Port A data is transferred to IOL/FOL.
Note 2: Data is latched in the output data registers (final output latch) and will be single buffered at the pin if the DDR is 1. The output buffers will be turned off if the DDR is 0.
Note 3: The output drivers that connect the final output latch to the pins are turned on.

Abbreviations:
IOL – Initial Output Latch
FOL – Final Output Latch
FIL – Final Input Latch
S.B. – Single Buffered
D.B. – Double Buffered
DDR – Data Direction Register

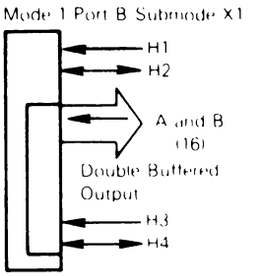
Port B Submode X0 (16-Bit Double-Buffered Input) –



In Mode 1 Port B Submode X0, double-buffered input transfers of up to 16 bits may be obtained. The level of all 16 pins is asynchronously latched with the asserted edge of H3. The processor may check H3S status bit to determine if new data is present. The $\overline{\text{DMAREQ}}$ pin may be used to signal a DMA controller to empty the input buffers. Regardless of the bus master, Port A data should be read first. (Actually, Port A data need not be read at all.) Port B data should be read last. The operation of the internal handshake controller, the H3S bit, and $\overline{\text{DMAREQ}}$ are keyed to the reading of the Port B data register. (The MC68450 DMAC can be programmed to perform the exact transfers needed for compatibility with the PI/T.) H4 may be programmed for all five of the handshake options mentioned in the Port General Information and Conventions section.

For pins used as outputs, the data path consists of a single latch driving the output buffer. Data written to the port's data register does not affect the operation of any handshake pin, status bit, or any other aspect of the PI/T. Thus, output pins may be used independently of the input transfer. However, read bus cycles to the Port B Data Register do remove data, so care should be taken to avoid unwanted read cycles.

Port B Submode X1 (16-Bit Double-Buffered Output) –



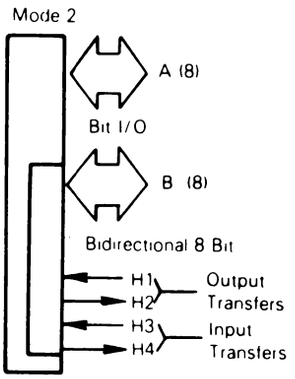
Refer to PARALLEL PORTS Double Buffered Input Transfers for a sample timing diagram (Figure 11).

In Mode 1 Port B Submode X1, double buffered output transfers of up to 16 bits may be obtained. Data is written by the bus master (processor or DMA controller) in two bytes. The first byte (most-significant) is written to the Port A Data Register. It is stored in a temporary latch until the next byte is written to the Port B Data Register. Then all 16 bits are transferred to the final output latches of Ports A and B. Both options for interpretation of the H3S status bit, mentioned in Port General Information and Comments section, are available and apply to the 16-bit port as a whole. The $\overline{\text{DMAREQ}}$ pin may be used to signal a DMA controller to transfer another word to the port output latches. (The MC68450 DMAC can be programmed to perform the exact transfers needed for compatibility with the PI/T.) H4 may be programmed for all five of the handshake options mentioned in the Port General Information and Comments section.

For pins used as inputs, data written to either data register is double-buffered and passed to the initial or final output latch, as usual, but the output buffer is disabled.

Refer to PARALLEL PORTS Double-Buffered Input/Output Transfer for a sample timing diagram (Figure 12).

MODE 2 — BIDIRECTIONAL 8-BIT MODE

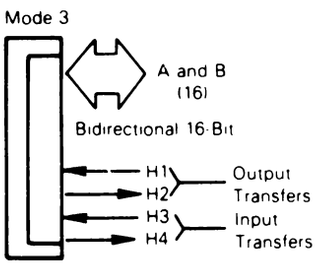


In Mode 2, Port A is used for simple bit I/O with no associated handshake pins. Port B is used for bidirectional 8-bit double-buffered transfers. H1 and H2, enabled by the H12 Enable bit in the Port General Control Register, control output transfers, while H3 and H4, enabled by the Port General Control Register bit H34 Enable, control input transfers. The instantaneous direction of the data is determined by the H1 handshake pin. The Port B Data Direction Register is not used. The Port A and Port B submode fields do not affect PI/T operation in Mode 2.

Double-Buffered I/O (Port B) — The only aspect of bidirectional double-buffered transfers that differs from the unidirectional modes lies in controlling the Port B output buffers. They are controlled by the level of H1. When H1 is negated, the Port B output buffers (all 8) are enabled and the pins drive the bidirectional bus. Generally, H1 is negated in response to an asserted H2, which indicates that new output data is present in the double-buffered latches. Following acceptance of the data, the peripheral asserts H1, disabling the Port B output buffers. Other than controlling the output buffer, H1 is edge-sensitive as in other modes. Input transfers proceed identically to the double-buffered input protocol described in the Port General Information and Conventions Section. In Mode 2, only the interlocked and pulsed handshake pin options are available on H2 and H4. The $\overline{\text{DMAREQ}}$ pin may be associated with either input transfers (H3) or output transfers (H1), but not both. Refer to Table 6 for a summary of the Port B Data Register responses in Mode 2.

Bit I/O (Port A) — Mode 2, Port A performs simple bit I/O with no associated handshake pins. This configuration is intended for applications in which several independent devices must be controlled or monitored. Data written to the Port A data register is single-buffered. If the Port A Data Direction Register bit for that pin is 1 (output), the output buffer is enabled. If it is 0, data written is still latched but not available at the pin. Data read from the data register is either the instantaneous value of the pin or what was written to the data register, depending on the contents of the Port A Data Direction Register. This is summarized in Table 7.

MODE 3 — BIDIRECTIONAL 16-BIT DOUBLE-BUFFERED I/O



In Mode 3, Ports A and B are used for bidirectional 16-bit double-buffered transfers. H1 and H2 control output transfers, while H3 and H4 control input transfers. (H1 and H2 are enabled by the H12 Enable bit while H3 and H4 are enabled by the H34 Enable bit of the Port General Control Register.) The instantaneous direction of the data is determined by the H1 handshake pin, and thus, the data direction registers are not used. The Port A and Port B submode fields do not affect PI/T operation in Mode 3.

The only aspect of bidirectional double-buffered transfers that differs from the unidirectional modes lies in controlling the Port A and B output buffers. They are controlled by the level of H1. When H1 is negated, the output buffers (all 16) are enabled and the pins drive the bidirectional bus. General-

TABLE 6 — MODE 2 PORT B DATA PATHS

Mode	Read Port B Data Register	Write Port B Data Register
2	FIL, D.B.	IOL/FOL, D.B.

Abbreviations:
IOL — Initial Output Latch
FOL — Final Output Latch
FIL — Final Input Latch
D.B. — Double Buffered

TABLE 7 — MODE 2 PORT A DATA PATHS

Mode	Read Port A Data Register		Write Port A Data Register	
	DDR = 0	DDR = 1	DDR = 0	DDR = 1
2	Pin	FOL	FOL	FOL, S.B.

Abbreviations:
S.B. — Single Buffered
FOL — Final Output Latch
DDR — Data Direction Register

ly, H1 is negated in response to an asserted H2, which indicates that new output data is present in the double-buffered latches. Following acceptance of the data, the peripheral asserts H1, disabling the output buffers. Other than controlling the output buffers, H1 is edge-sensitive as in other modes. Input transfers proceed identicaliy to the double-buffered input protocol described in the Port General Information and Conventions section. Port A and B data is latched with the asserted edge of H3. In Mode 3, only the interlocked and pulsed handshake pin options are available to H2 and H4. The $\overline{\text{DMAREQ}}$ pin may be associated with either input transfers (H3) or output transfers (H1), but not both. H2 indicates when new data is available in the Port B (and implicitly Port A) output latches, but unless the buffer is enabled by H1, the data is not driving the pins.

Mode 3 can provide convenient high-speed 16-bit transfers. The Port A and B Data Registers are addressed for compatibility with the MC68000's Move Peripheral (MOVEP) instruction and with the MC68450 DMAC. To take advantage of this, Port A should contain the most-significant data and always be read or written by the bus master first. The interlocked and pulsed handshake protocols are keyed to accesses to the Port B Data Register in Mode 3. If it is accessed last, the 16-bit double-buffered transfer proceed smoothly. Refer to Table 8 for a summary of the Port A and B data paths in Mode 3.

DMA REQUEST OPERATION

The Direct Memory Access Request ($\overline{\text{DMAREQ}}$) pulse (when enabled) is associated with output or input transfers to keep the initial and final output latches full or initial and final input latches empty, respectively. Figures 13 and 14 show all the possible paths in generating DMA requests.

$\overline{\text{DMAREQ}}$ is generated on the bus side of the MC68230 by the synchronized* Chip Select. If the conditions of Figures 13 and 14 are met, an access of the bus (assertion of $\overline{\text{CS}}$) will cause $\overline{\text{DMAREQ}}$ to be asserted 3 PI/T clocks (plus the delay time from the clock edge) after $\overline{\text{CS}}$ is synchronized.* $\overline{\text{DMAREQ}}$ remains asserted 3 clock cycles (plus the delay time from the clock edge) and is then negated.

The $\overline{\text{DMAREQ}}$ pulse associated with a peripheral or port side of the PI/T is caused by the synchronized* H1(H3) input. If the conditions of Figures 13 and 14 are met, a port access (assertion of the H1(H3) input) will cause $\overline{\text{DMAREQ}}$ to be asserted 2.5 PI/T clock cycles (plus the delay time from clock edge) after H1(H3) is sycnhronized.* $\overline{\text{DMAREQ}}$ remains asserted 3 clock cycles (plus the delay time from the clock edge) and is then negated.

TABLE 8 — MODE 3 PORT A AND B DATA PATHS

Mode	Read Port A and B Data Register	Write Port A and B Data Register
3	FIL, D.B.	IOL/FOL, D.B., Note 1
Note 1: Data written to Port A goes to a temporary latch. When the Port B data register is later written, Port A data is transferred to IOL/FOL.		
Abbreviations: IOL — Initial Output Latch FOL — Final Output Latch FIL — Fmal Input Latch		S.B. — Single Buffered D.B. — Double Buffered

FIGURE 13 — $\overline{\text{DMAREQ}}$ ASSOCIATED WITH OUTPUT TRANSFERS
Data in Output Latches

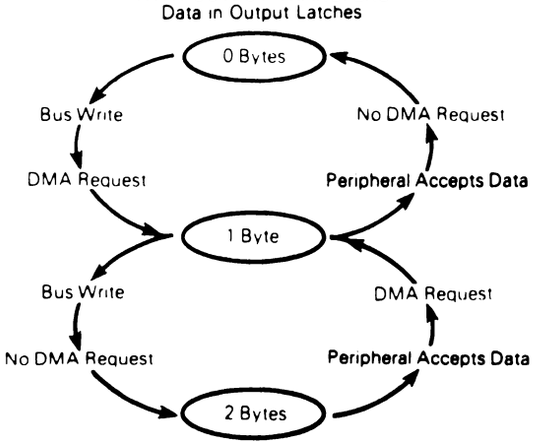
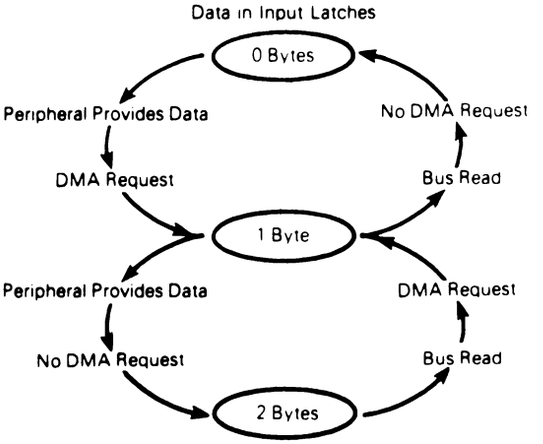


FIGURE 14 — $\overline{\text{DMAREQ}}$ ASSOCIATED WITH INPUT TRANSFERS
Data in Input Latches



*Synchronized means that the input signal has been seen by the PI/T on the appropriate edge of the clock (rising edge for H1(H3) and falling edge for $\overline{\text{CS}}$) (Refer to the BUS INTERFACE CONNECTION section for the exception concerninq $\overline{\text{CS}}$.) If a bus access (assertion of $\overline{\text{CS}}$) and a port access (assertion of H1(H3)) occur at the same time, $\overline{\text{CS}}$ will be recognized without any delay. H1(H3) will be recognized one clock cycle later

TIMER

The MC68230 timer can provide several facilities needed by MC68000 operating systems. It can generate periodic interrupts, a square wave, or a single interrupt after a programmed time period. Also, it can be used for elapsed time measurement or as a device watchdog. This section describes the programmable options available, capabilities, and restrictions that apply to the timer.

The PI/T timer contains a 24-bit synchronous down counter that is loaded from three 8-bit Counter Preload Registers. The 24-bit counter may be clocked by the output of a 5-bit (divide-by-32) prescaler or by an external timer input TIN. If the prescaler is used, it may be clocked by the system clock (CLK pin) or by the TIN external input. The counter signals the occurrence of an event primarily through zero detection. (A zero is when the counter of the 24-bit timer is equal to zero.) This sets the zero detect status (ZDS) bit in the Timer Status Register. It may be checked by the processor or may be used to generate a timer interrupt. The ZDS bit is reset by writing a 1 to the Timer Status Register in that bit position.

The general operation of the timer is flexible and easily programmable. The timer is fully configured and controlled by programming the 8-bit Timer Control Register. It controls: (1) the choice between the Port C operation and the timer operation of three timer pins, (2) whether the counter is loaded from the Counter Preload Register or rolls over when zero detect is reach, (3) the clock input, (4) whether the prescaler is used, and (5) whether the timer is enabled.

RUN/HALT DEFINITION

The overall operation of the timer is described in terms of the run or halt states. The control of the current state is determined by programming the Timer Control Register. When in the halt state, all of the following occur.

1. The prior contents of the counter is not altered and is reliably readable via the Count Registers.
2. The prescaler is forced to $1F whether or not it is used.
3. The ZDS status bit is forced to 0, regardless of the possible zero contents of the 24-bit counter.

The run state is characterized by:

1. The counter is clocked by the source programmed in the Timer Control Register.
2. The counter is not reliably readable.
3. The prescaler is allowed to decrement if programmed for use.
4. The ZDS status bit is set when the 24-bit counter transitions from $000001 to $000000.

TIMER RULES

This section provides a set of rules that allow easy application of the timer.

1. Refer to the Run/Halt Definition above.
2. When the $\overline{\text{RESET}}$ pin is asserted, all bits of the Timer Control Register go to 0, configuring the dual function pins as Port C inputs.
3. The contents of the Counter Preload Registers and counter are not affected by the $\overline{\text{RESET}}$ pin.
4. The Count Registers provide a direct read data path from each portion of the 24-bit counter, but data written to their addresses is ignored. (This results in a normal bus cycle.) These registers are readable at any time, but their contents are never latched. Unreliable data may be read when the timer is in the run state.
5. The Counter Preload Registers are readable and writable at any time and this occurs independently of any timer operation. No protection mechanisms are provided against ill-timed writes.
6. The input frequency to the 24-bit counter from the TIN pin or prescaler output, must be between 0 and the input frequency at CLK pin divided by 32 regardless of the configuration chosen.
7. For configurations in which the prescaler is used (with the CLK pin or TIN pin as an input), the contents of the Counter Preload Register (CPR) is transferred to the counter the first time that the prescaler passes from $00 to $1F (rolls over) after entering the run state. Thereafter, the counter decrements or is loaded from the Counter Preload Register when the prescaler rolls over.
8. For configurations in which the prescaler is not used, the contents of the Counter Preload Registers are transferred to the counter on the first asserted edge of the TIN input after entering the run state. On subsequent asserted edges the counter decrements or is loaded from the Counter Preload Registers.
9. The lowest value allowed in the Counter Preload Register for use with the counter is $000001.

TIMER INTERRUPT ACKNOWLEDGE CYCLES

Several conditions may be present when the timer interrupt acknowledge pin ($\overline{\text{TIACK}}$) is asserted. These conditions affect the PI/T's response and the termination of the bus cycle. (See Table 9.)

TABLE 9 — RESPONSE TO TIMER INTERRUPT ACKNOWLEDGE

PC3/TOUT Function	Response to Asserted $\overline{\text{TIACK}}$
PC3 Port C Pin	No response No $\overline{\text{DTACK}}$
TOUT Square Wave	No response No $\overline{\text{DTACK}}$
TOUT Negated Timer Interrupt Request	No response No $\overline{\text{DTACK}}$
TOUT Asserted Timer Interrupt Request	Timer Interrupt Vector Content is $\overline{\text{DTACK}}$ Asserted

PROGRAMMER'S MODEL

The internal accessible register organization is represented in Table 10. Address space within the address map is reserved for future expansion. Throughout the PI/T data sheet the following conventions are maintained.

1. A read from a reserved location in the map results in a read from the "null register." The null register returns all zeros for data and results in a normal bus cycle. A write to one of these locations results in a normal bus cycle but no write occurs.
2. Unused bits of a defined register are denoted by "•" and are read as zeroes.
3. Bits that are unused in the chosen mode/submode but are used in others, are denoted by "X", and are readable and writeable. Their content, however, is ignored in the chosen mode/submode.
4. All registers are addressable as 8-bit quantities. To facilitate operation with the MOVEP instruction and the DMAC, addresses are ordered such that certain sets of registers may also be accessed as words (2 bytes) or long words (4 bytes).

TABLE 10 — PI/T REGISTER ADDRESSING ASSIGNMENTS

Register	Register Select Bits					Accessible	Affected by Reset	Affected by Read Cycle
	5	4	3	2	1			
Port General Control Register (PGCR)	0	0	0	0	0	R W	Yes	No
Port Service Request Register (PSRR)	0	0	0	0	1	R W	Yes	No
Port A Data Direction Register (PADDR)	0	0	0	1	0	R W	Yes	No
Port B Data Direction Register (PBDDR)	0	0	0	1	1	R W	Yes	No
Port C Data Direction Register (PCDDR)	0	0	1	0	0	R W	Yes	No
Port Interrupt Vector Register (PIVR)	0	0	1	0	1	R W	Yes	No
Port A Control Register (PACR)	0	0	1	1	0	R W	Yes	No
Port B Control Register (PBCR)	0	0	1	1	1	R W	Yes	No
Port A Data Register (PADR)	0	1	0	0	0	R W	No	* *
Port B Data Register (PBDR)	0	1	0	0	1	R W	No	* *
Port A Alternate Register (PAAR)	0	1	0	1	0	R	No	No
Port B Alternate Register (PBAR)	0	1	0	1	1	R	No	No
Port C Data Register (PCDR)	0	1	1	0	0	R W	No	No
Port Status Register (PSR)	0	1	1	0	1	R W*	Yes	No
Timer Control Register (TCR)	1	0	0	0	0	R W	Yes	No
Timer Interrupt Vector Register (TIVR)	1	0	0	0	1	R W	Yes	No
Counter Preload Register High (CPRH)	1	0	0	1	1	R W	No	No
Counter Preload Register Middle (CPRM)	1	0	1	0	0	R W	No	No
Counter Preload Register Low (CPRL)	1	0	1	0	1	R W	No	No
Count Register High (CNTRH)	1	0	1	1	1	R	No	No
Count Register Middle (CNTRM)	1	1	0	0	0	R	No	No
Count Register Low (CNTRL)	1	1	0	0	1	R	No	No
Timer Status Register (TSR)	1	1	0	1	0	R W*	Yes	No

* A write to this register may perform a special status resetting operation.
* * Mode dependent

R = Read
W = Write

Port General Control Register (PGCR) –

7	6	5	4	3	2	1	0
Port Mode Control		H34 Enable	H12 Enable	H4 Sense	H3 Sense	H2 Sense	H1 Sense

The Port General Control Register controls many of the functions that are common to the overall operation of the ports. The PGCR is composed of three major fields: bits 7 and 6 define the operational mode of Ports A and B and affect operation of the handshake pins and status bits; bits 5 and 4 allow a software controlled disabling of particular hardware associated with the handshake pins of each port; and bits 3-0 define the sense of the handshake pins. The PGCR is always readable and writeable.

All bits are reset to 0 when the RESET pin is asserted.

The Port Mode Control field should be altered only when the H12 Enable and H34 Enable bits are 0. Except when Mode 0 is desired, the Port General Control register must be written once to establish the mode, and again to enable the respective operation(s).

PGCR

7	6	Port Mode Control
0	0	Mode 0 (Unidirectional 8-Bit Mode)
0	1	Mode 1 (Unidirectional 16-Bit Mode)
1	0	Mode 2 (Bidirectional 8-Bit Mode)
1	1	Mode 3 (Bidirectional 16-Bit Mode)

PGCR

5	H34 Enable
0	Disabled
1	Enabled

PGCR

4	H12 Enable
0	Disabled
1	Enabled

PGCR

3-0	Handshake Pin Sense
0	The associated pin is at the high-voltage level when negated and at the low-voltage level when asserted.
1	The associated pin is at the low-voltage level when negated and at the high-voltage level when asserted.

Port Service Request Register (PSRR) –

7	6	5	4	3	2	1	0
*	SVCRQ Select		Interrupt PFS		Port Interrupt Priority Control		

The Port Service Request Register controls other functions that are common to the overall operation to the ports. It is composed of four major fields: bit 7 is unused and is always read as 0; bits 6 and 5 define whether interrupt or DMA requests are generated from activity on the H1 and H3 handshake pins; bits 4 and 3 determine whether two dual function pins operate as Port C or port interrupt request/-acknowledge pins; and bits 2, 1, and 0 control the priority among all port interrupt sources. Since bits 2, 1, and 0 affect interrupt operation, it is recommended that they be changed only when the affected interrupt(s) is (are) disabled or known to remain inactive. The PSRR is always readable and writeable.

All bits are reset to 0 when the $\overline{\text{RESET}}$ pin is asserted.

PSRR

6	5	SVCRQ Select
0	X	The PC4/$\overline{\text{DMAREQ}}$ pin carries the PC4 function; DMA is not used.
1	0	The PC4/$\overline{\text{DMAREQ}}$ pin carries the $\overline{\text{DMAREQ}}$ function and is associated with double-buffered transfers controlled by H1. H1 is removed from the PI/T's interrupt structure, and thus, does not cause interrupt requests to be generated. To obtain $\overline{\text{DMAREQ}}$ pulses, Port A Control Register bit 1 (H1 SVCRQ Enable) must be a 1.
1	1	The PC4/$\overline{\text{DMAREQ}}$ pin carries the $\overline{\text{DMAREQ}}$ function and is associated with double-buffered transfers controlled by H3. H3 is removed from the PI/T's interrupt structure, and thus, does not cause interrupt requests to be generated. To obtain $\overline{\text{DMAREQ}}$ pulses, Port B Control Register bit 1 (H3 SVCRQ Enable) must be 1

PSRR

4	3	Interrupt Pin Function Select
0	0	The PC5/$\overline{\text{PIRQ}}$ pin carries the PC5 function. The PC6/$\overline{\text{PIACK}}$ pin carries the PC6 function.
0	1	The PC5/$\overline{\text{PIRQ}}$ pin carries the $\overline{\text{PIRQ}}$ function. The PC6/$\overline{\text{PIACK}}$ pin carries the PC6 function.
1	0	The PC5/$\overline{\text{PIRQ}}$ pin carries the PC5 function. The PC6/$\overline{\text{PIACK}}$ pin carries the $\overline{\text{PIACK}}$ function.
1	1	The PC5/$\overline{\text{PIRQ}}$ pin carries the $\overline{\text{PIRQ}}$ function. The PC6/$\overline{\text{PIACK}}$ pin carries the $\overline{\text{PIACK}}$ function.

Bits 2, 1, and 0 determine port interrupt priority. The priority is shown in descending order left to right.

PSRR 2	1	0	Port Highest	Interrupt	Priority	Control Lowest
0	0	0	H1S	H2S	H3S	H4S
0	0	1	H2S	H1S	H3S	H4S
0	1	0	H1S	H2S	H4S	H3S
0	1	1	H2S	H1S	H4S	H3S
1	0	0	H3S	H4S	H1S	H2S
1	0	1	H3S	H4S	H2S	H1S
1	1	0	H4S	H3S	H1S	H2S
1	1	1	H4S	H3S	H2S	H1S

Port A Data Direction Register (PADDR) – The Port A Data Direction Register determines the direction and buffering characteristics of each of the Port A pins. One bit in the PADDR is assigned to each pin. A 0 indicates that the pin is used as an input, while a 1 indicates it is used as an output. The PADDR is always readable and writeable. This register is ignored in Mode 3.

All bits are reset to the 0 (input) state when the $\overline{\text{RESET}}$ pin is asserted.

Port B Data Direction Register (PBDDR) – The PBDDR is identical to the PADDR for the Port B pins and the Port B Data Register, except that this register is ignored in Modes 2 and 3.

Port C Data Direction Register (PCDDR) – The Port C Data Direction Register specifies whether each dual-function pin that is chosen for Port C operation is an input (0) or an output (1) pin. The PCDDR, along with bits that determine the respective pin's function, also specify the exact hardware to be accessed at the Port C Data Register address. (See the Port C Data Register description for more details.) The PCDDR is an 8-bit register that is readable and writeable at all times. Its operation is independent of the chosen PI/T mode.

These bits are cleared to 0 when the $\overline{\text{RESET}}$ pin is asserted.

Port Interrupt Vector Register (PIVR) –

7	6	5	4	3	2	1	0
Interrupt Vector Number						*	*

The Port Interrupt Vector Register contains the upper order six bits of the four port interrupt vectors. The contents of this register may be read two ways: by an ordinary read cycle, or by a port interrupt acknowledge bus cycle. The exact data read depends on how the cycle was initiated and other factors. Behavior during a port interrupt acknowledge cycle is summarized above in Table 3.

From a normal read cycle (CS), there is never a consequence to reading this register. Following negation of the RESET pin, but prior to writing to the PIVR, a $0F will be read. After writing to the register, the upper 6 bits may be read and the lower 2 bits are forced to 0. No prioritization computation is performed.

Port A Control Register (PACR) –

7	6	5	4	3	2	1	0
Port A Submode		H2 Control			H2 Int. Enable	H1 SVCRQ Enable	H1 Stat. Ctrl.

The Port A Control Register, in conjunction with the programmed mode and the Port B submode, control the operation of Port A and the handshake pins H1 and H2. The Port A Control Register contains five fields: bits 7 and 6 specify the Port A submode; bits 5, 4, and 3 control the operation of the H2 handshake pin and H2S status bit; bit 2 determines whether an interrupt will be generated when the H2S status bit goes to 1, bit 1 determines whether a service request (interrupt request or DMA request) will occur; bit 0 controls the operation of the H1S status bit. The PACR is always readable and writeable.

All bits are cleared to 0 when the RESET pin is asserted.

When the Port A submode field is relevant in a mode/submode definition, it must not be altered unless the H12 Enable bit in the Port General Control Register is 0. (See Table 2.)

The operation of H1 and H2 and their related status bits is given below, for each of the modes specified by Port General Control Register bits 7 and 6. This description is organized such that for each mode/submode all programmable options of each pin and status bit are given.

Bits 2 and 1 carry the same meaning in each mode/submode, and thus are specified only once.

PACR

2	H2 Interrupt Enable
0	The H2 interrupt is disabled.
1	The H2 interrupt is enabled.

PACR

1	H1 SVCRQ Enable
0	The H1 interrupt and DMA request are disabled.
1	The H1 interrupt and DMA request are enabled

PACR Mode 0 Port A Submode 00

PACR

5	4	3	H2 Control
0	X	X	Input pin – status only.
1	0	0	Output pin – always negated.
1	0	1	Output pin – always asserted.
1	1	0	Output pin – interlocked input handshake protocol.
1	1	1	Output pin – pulsed input handshake protocol.

PACR

0	H1 Status Control
X	Not Used

PACR Mode 0 Port a Submode 01

PACR

5	4	3	H2 Control
0	X	X	Input pin – status only.
1	0	0	Output pin – always negated.
1	0	1	Output pin – always asserted.
1	1	0	Output pin – interlocked output handshake protocol.
1	1	1	Output pin – pulsed output handshake protocol.

PACR

0	H1 Status Control
0	The H1S status bit is 1 when either the Port A initial or final output latch can accept new data. It is 0 when both latches are full and cannot accept new data.
1	The H1S status bit is 1 when both of the Port A output latches are empty. It is 0 when at least one latch is full.

PACR Mode 0 Port A Submode 1X

PCR

5	4	3	H2 Control
0	X	X	Input pin – status only.
1	X	0	Output pin – always negated.
1	X	1	Output pin – always asserted.

PACR

0	H1 Status Control
X	Not used.

PACR Mode 1 Port A Submode XX Port B Submode X0

PACR

5	4	3	H2 Control
0	X	X	Input pin – status only.
1	X	0	Output pin – always negated.
1	X	1	Output pin – always asserted

PACR

0	H1 Status Control
X	Not used.

PACR Mode 1 Port A Submode XX Port B Submode X1

PACR

5	4	3	H2 Control
0	X	X	Input pin – status only
1	X	0	Output pin – always negated.
1	X	1	Output pin – always asserted.

PACR

0	H1 Status Control
X	Not used.

PACR Mode 2

PACR

5	4	3	H2 Control
X	X	0	Output pin – interlocked output handshake protocol.
X	X	1	Output pin – pulsed output handshake protocol.

PACR

0	H1 Status Control
0	The H1S status bit is 1 when either the Port B initial or final output latch can accept new data. It is 0 when both latches are full and cannot accept new data.
1	The H1S status bit is 1 when both of the Port B output latches are empty. It is 0 when at least one latch is full.

PACR Mode 3

PACR

5	4	3	H2 Control
X	X	0	Output pin – interlocked output handshake protocol.
X	X	1	Output pin – pulsed output handshake protocol.

PACR

0	H1 Status Control
0	The H1S status bit is 1 when either the initial or final output latch of Port A and B can accept new data. It is 0 when both latches are full and cannot accept new data.
1	The H1S status bit is 1 when both the initial and final output latches of Ports A and B are empty. It is 0 when either the initial or final latch of Ports A and B is full.

Port B Control Register (PBCR) –

7	6	5	4	3	2	1	0
Port B Submode		H4 Control			H4 Int. Enable	H3 SVCRQ Enable	H3 Stat. Ctrl.

The Port B Control Register specifies the operation of Port B and the handshake pins H3 and H4. The Port B control register contains five fields: bits 7 and 6 specify the Port B submode; bits 5, 4, and 3 control the operation of the H4 handshake pin and H4S status bit; bit 2 determines whether an interrupt will be generated when the H4S status bit goes to 1, bit 1 determines whether a service request (interrupt request or DMA request) will occur; bit 0 controls the operation of the H3S status bit. The PACR is always readable and writeable. There is never a consequence to reading the register.

All bits are cleared to 0 when the $\overline{\text{RESET}}$ pin is asserted.

When the Port B submode field is relevant in a mode/submode definition, it must not be altered unless the H34 Enable bit in the Port General Control Register is 0. (See Table 2.)

The operation of H3 and H4 and their related status bits is given below, for each of the modes specified by Port General Control Register bits 7 and 6. This description is organized such that for each mode/submode all programmable options of each pin and status bit are given.

Bits 2 and 1 carry the same meaning in each mode/submode, and thus are specified only once.

PBCR

2	H4 Interrupt Enable
0	The H4 interrupt is disabled.
1	The H4 interrupt is enabled.

PBCR

1	H3 SVCRQ Enable
0	The H3 interrupt and DMA request are disabled.
1	The H3 interrupt and DMA request are enabled.

PBCR Mode 0 Port B Submode 00

PBCR

5	4	3	H4 Control
0	X	X	Input pin – status only.
1	0	0	Output pin – always negated.
1	0	1	Output pin – always asserted.
1	1	0	Output pin – interlocked input handshake protocol.
1	1	1	Output pin – pulsed input handshake protocol.

PBCR

0	H3 Status Control
X	Not used.

PBCR Mode 0 Port B Submode 01

PBCR

5	4	3	H4 Control
0	X	X	Input pin – status only.
1	0	0	Output pin – always negated.
1	0	1	Output pin – always asserted.
1	1	0	Output pin – interlocked output handshake protocol.
1	1	1	Output pin – pulsed output handshake protocol.

PBCR

0	H3 Status Control
0	The H3S status bit is 1 when either the Port B initial or final output latch can accept new data. It is 0 when both latches are full and cannot accept new data.
1	The H3S status bit is 1 when both of the Port B output latches are empty. It is 0 when at least one latch is full.

PBCR Mode 0 Port B Submode 1X

PBCR

5	4	3	H4 Control
0	X	X	Input Pin – status only.
1	X	0	Output pin – always negated.
1	X	1	Output pin – always asserted.

PBCR

0	H3 Status Control
X	Not used.

PBCR Mode 1 Port B Submode X0

5	4	3	H4 Control
0	X	X	Input pin – status only.
1	0	0	Output pin – always negated.
1	0	1	Output pin – always asserted.
1	1	0	Output pin – interlocked input handshake protocol.
1	1	1	Output pin – pulsed input handshake protocol.

PBCR
0 — H3 Status Control
X Not used.

PBCR Mode 1 Port B Submode X1

PBCR
5 4 3 — H4 Control
0 X X Input pin – status only.
1 0 0 Output pin – always negated.
1 0 1 Output pin – always asserted.
1 1 0 Output pin – interlocked output handshake protocol.
1 1 1 Output pin – pulsed output handshake protocol.

PBCR
0 — H3 Status Control
0 The H3S status bit is 1 when either the initial or final output latch of Port A and B can accept new data. It is 0 when both latches are full and cannot accept new data.
1 The H3S status bit is 1 when both the initial and final output latches of Ports A and B are empty. It is 0 when neither the initial or final latch of Ports A and B is full.

PBCR Mode 2

PBCR
5 4 3 — H4 Control
X X 0 Output pin - interlocked input handshake protocol.
X X 1 Output pin - pulsed input handshake protocol.

PBCR
0 — H3 Status Control
X Not used.

PBCR Mode 3

PBCR
5 4 3 — H4 Control
X X 0 Output pin – interlocked input handshake protocol.
X X 1 Output pin – pulsed input handshake protocol.

PBCR
0 — H3 Status Control
X Not used.

Port A Data Register (PADR) – The Port A Data Register is an address for moving data to and from the Port A pins. The Port A Data Direction Register determines whether each pin is an input (0) or an output (1), and is used in configuring the actual data paths. This is mode dependent and is described with the modes above.

This register is readable and writeable at all times. Depending on the chosen mode/submode, reading or writing may affect the double-buffered handshake mechanism. The Port A Data Register is not affected by the assertion of the $\overline{\text{RESET}}$ pin.

Port B Data Register (PBDR) – The Port B Data Register is an address for moving data to and from the Port B pins. The Port B Data Direction Register determines whether each pin is an input (0)or an output (1), and is used in configuring the actual data paths. This is mode dependent and is described with the modes, above.

This register is readable and writeable at all times. Depending on the chosen mode/submode, reading or writing may affect the double-buffered handshake mechanism. The Port B Data Register is not affected by the assertion of the $\overline{\text{RESET}}$ pin.

Port A Alternate Register (PAAR) – The Port A Alternate Register is an alternate address for reading the Port A pins. It is a read-only address and no other PI/T condition is affected. In all modes and the instantaneous pin level is read and no input latching is performed except at the data bus interface (see Bus Interface Connection). Writes to this address are answered with $\overline{\text{DTACK}}$, but the data is ignored.

Port B Alternate Register (PBAR) The Port B Alternate Register is an alternate address for reading the Port B pins. It is a read-only address and no other PI/T condition is affected. In all modes the instantaneous pin level is read and no input latching is performed except at the data bus interface (see Bus Interface Connection). Writes to this address are answered with $\overline{\text{DTACK}}$, but the data is ignored.

Port C Data Register (PCDR) The Port C Data Register is an address for moving data to and from each of the eight Port C/alternate function pins. The exact hardware accessed is determined by the type of bus cycle (read or write) and individual conditions affecting each pin. These conditions are (1) whether the pin is used for the Port C or alternate function, and (2) whether the Port C Data Direction Register indicates the input or output direction. The Port C Data Register is single buffered for output pins and not buffered for input pins. These conditions are summarized in Table 11.

The Port C Data Register is not affected by the assertion of the $\overline{\text{RESET}}$ pin.

The operation of the PCDR is independent of the chosen PI/T mode.

TABLE 11 – PCDR HARDWARE ACCESSES

Read Port C Data Register			
Port C function PCDDR = 0	Port C function PCDDR = 1	Alternate function PCDDR = 0	Alternate function PCDDR = 1
pin	Port C output register	pin	Port C output register
Write Port C Data Register			
Port C Function PCDDR = 0	Port C Function PCDDR = 1	Alternate function PCDDR = 0	Alternate function PCDDR = 1
Port C output register, buffer disabled	Port C output register, buffer enabled	Port C output register	Port C output register

Note that two additional useful benefits result from this structure. First, it is possible to directly read the state of a dual-function pin while used for the non-Port C function. Second, it is possible to generate program controlled transitions on alternate-function pins by switching back to the Port C function, and writing to the PCDR.

This register is readable and writeable at all times.

Port Status Register (PSR) –

7	6	5	4	3	2	1	0
H4 Level	H3 Level	H2 Level	H1 Level	H4S	H3S	H2S	H1S

The Port Status Register contains information about handshake pin activity. Bits 7-4 show the instantaneous level of the respective handshake pin, and is independent of the handshake pin sense bits in the Port General Control Register. Bit 3-0 are the respective status bits referred to throughout this data sheet. Their interpretation depends on the programmed mode/submode of the PI/T. For Bits 3-0 a 1 is the active or asserted state.

Timer Control Register (TCR) –

7	6	5	4	3	2	1	0
TOUT/$\overline{\text{TIACK}}$ Control			Z D Ctrl.	*	Clock Control		Timer Enable

The Timer Control Register (TCR) determines all operations of the timer. Bits 7-5 configure the PC3/TOUT and PC7/$\overline{\text{TIACK}}$ pins for Port C, square wave, vectored interrupt, or autovectored interrupt operation; bit 4 specifies whether the counter receives data from the Counter Preload Register or continues counting when zero detect is reached, bit 3 is unused and is read as 0; bits 2 and 1 configure the path from the CLK and TIN pins to the counter controller; bit 0 enables the timer. This register is readable and writeable at all times.

All bits are cleared to 0 when the $\overline{\text{RESET}}$ pin is asserted.

TCR 7 6 5 — TOUT/$\overline{\text{TIACK}}$ Control

0 0 X The dual-function pins PC3/TOUT and PC7/$\overline{\text{TIACK}}$ carry the Port C function.

0 1 X The dual-function pin PC3/TOUT carries the TOUT function. In the run state it is used as a square wave output and is toggled on zero detect. The TOUT pin is high while in the halt state. The dual-function pin PC7/$\overline{\text{TIACK}}$ carries the PC7 function.

1 0 0 The dual-function pin PC3/TOUT carries the TOUT function. In the run or halt state it is used as a timer interrupt request output. The timer interrupt is disabled; thus, the pin is always three-stated. The dual-function pin PC7/$\overline{\text{TIACK}}$ carries the $\overline{\text{TIACK}}$ function; however, since interrupt request is negated, the PI/T produces no response, i.e., no data or $\overline{\text{DTACK}}$, to an asserted $\overline{\text{TIACK}}$. Refer to Timer Interrupt Cycle section for details. This combination and the 101 state below support vectored timer interrupts.

1 0 1 The dual-function pin PC3/TOUT carries the TOUT function and is used as a timer interrupt request output. The timer interrupt is enabled; thus, the pin is low when the timer ZDS status bit is 1. The dual function pin PC7/$\overline{\text{TIACK}}$ carries the $\overline{\text{TIACK}}$ function and is used as a timer interrupt acknowledge input. Refer to the Timer Interrupt Acknowledge Cycle section for details. This combination and the 100 state above support vectored timer interrupts.

1 1 0 The dual-function pin PC3/TOUT carries the TOUT function. In the run or halt state it is used as a timer interrupt request output. The timer interrupt is disabled; thus, the pin is always three-stated. The dual-function pin PC7/$\overline{\text{TIACK}}$ carries the PC7 function.

1 1 1 The dual-function pin PC3/TOUT carries the TOUT function and is used as a timer interrupt request output. The timer interrupt is enabled; thus, the pin is low when the timer ZDS status bit is 1. The dual-function pin PC7/$\overline{\text{TIACK}}$ carries the PC7 function and autovectored interrupts are supported.

TCR 4 — Zero Detect Control

0 The counter is loaded from the Counter Preload Register on the first clock to the 24-bit counter after zero detect, and resumes counting.

1 The counter rolls over on zero detect, then continues counting.

Bit 3 is unused and is always read as 0.

TCR 2 1 — Clock Control

0 0 The PC2/TIN input pin carries the Port C function and the CLK pin and prescaler are used. The prescaler is decremented on the falling transition of the CLK pin, the 24-bit counter is decremented or loaded from the Counter Preload Registers when the prescaler rolls over from $00 to $1F. The Timer Enable bit determines whether the timer is in the run or halt state.

0 1 The PC2/TIN pin serves as a timer input and the CLK pin and prescaler are used. The prescaler is decremented on the falling transition of the CLK pin, the 24-bit counter is decremented or loaded from the Counter Preload Registers when the prescaler rolls over from $00 to $1F. The timer is in the run state when the Timer Enable bit is 1 and the TIN pin is high, otherwise the timer is in the halt state.

1 0 The PC2/TIN pin serves as a timer input and the prescaler is used. The prescaler is decremented following the rising transition of the TIN pin after syncing with the internal clock. The 24-bit counter is decremented or loaded from the counter preload registers when the prescaler rolls over from $00 to $1F. The Timer Enable bit determines whether the timer is in the run or halt state.

1 1 The PC2/TIN pin serves as a timer input and the prescaler is unused. The 24-bit counter is decremented or loaded from the Counter Preload Registers following the rising edge of the TIN pin after syncing with the internal clock. The Timer Enable bit determines whether the timer is in the run or halt state.

TCR 0	Timer Enable
0	Disabled.
1	Enabled.

Timer Interrupt Vector Register (TIVR) – The timer interrupt vector register contains the 8-bit vector supplied when the timer interrupt acknowledge pin $\overline{TIACK}$ is asserted. The register is readable and writeable at all times, and the same value is always obtained from a normal read cycle and a timer interrupt acknowledge bus cycle ($\overline{TIACK}$). When the $\overline{RESET}$ pin is asserted the value of $0F is automatically loaded into the register. Refer to Timer Interrupt Acknowledge Cycle section for more details.

Counter Preload Register H, M, L (CPRH-L)

7	6	5	4	3	2	1	0	
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	CPRH
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	CPRM
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	CPRL

The Counter Preload Registers are a group of three 8-bit registers used for storing data to be transferred to the counter. Each of the registers is individually addressable, or the group may be accessed with the MOVEP.L or the MOVEP.W instructions. The address one less than the address of CPRH is the null register, and is reserved so that zeros are read in the upper 8 bits of the destination data register when a MOVEP.L is used. Data written to this address is ignored.

The registers are readable and writeable at all times. A read cycle proceeds independently of any transfer to the counter, which may be occuring simultaneously.

To insure proper operation of the PI/T Timer, a value of $000000 may not be stored in the Counter Preload Registers for use with the counter.

The $\overline{RESET}$ pin does not affect the contents of these registers.

Count Register H, M, L (CNTRH-L) –

7	6	5	4	3	2	1	0	
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	CNTRH
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	CNTRM
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	CNTRL

The count registers are a group of three 8-bit addresses at which the counter can be read. The contents of the counter are not latched during a read bus cycle; thus, the data read at these addresses is not guaranteed if the timer is in the run state. (Bits 2, 1, and 0 of the Timer Control Register specify the state.) Write operations to these addresses result in a normal bus cycle but the data is ignored.

Each of the registers is individually addressable, or the group may be accessed with the MOVEP.L or the MOVEP.W instructions. The address one less than the address of CNTRH is the null register, and is reserved so that zeros are read in the upper 8 bits of the destination data register when a MOVEP.L is used. Data written to this address is ignored.

Timer Status Register (TSR) –

7	6	5	4	3	2	1	0
*	*	*	*	*	*	*	ZDS

The Timer Status Register contains one bit from which the zero detect status can be determined. The ZDS status bit (bit 0) is an edge-sensitive flip-flop that is set to 1 when the 24 bit counter decrements from $000001 to $000000. The ZDS status bit is cleared to 0 following the direct clear operation (similar to that of the ports), or when the timer is halted. Note also that when the RESET pin is asserted the timer is disabled, and thus enters the halt state.

This register is always readable without consequence. A write access performs a direct clear operation if bit 0 in the written data is 1. Following that, the ZDS bit is 0.

This register is constructed with a reset dominant S-R flip-flop so that all clearing conditions prevail over the possible zero detect condition.

Bits 7-1 are unused and are read as 0.

TIMER APPLICATIONS SUMMARY

This section outlines programming of the Timer Control Register for several typical examples.

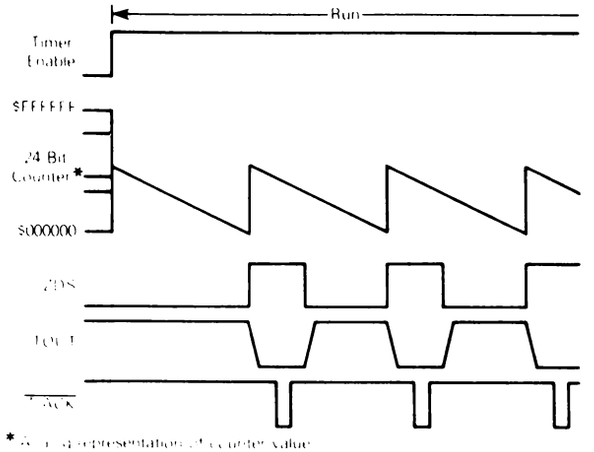
Periodic Interrupt Generator

7	6	5	4	3	2	1	0
TOUT $\overline{TIACK}$ control			Z D Ctrl	*			Timer Enable

In this configuration the timer generates a periodic interrupt. The TOUT pin is connected to the system's interrupt request circuitry and the $\overline{TIACK}$ pin may be used as an interrupt acknowledge input to the timer. The TIN pin may be used as a clock input.

The processor loads the Counter Preload Registers and Timer Control Register, and then enables the timer. When the 24-bit counter passes from $000001 to $000000 the ZDS status bit is set and the TOUT (interrupt request) pin is asserted. At the next clock to the 24 bit counter it is again loaded with the contents of the CPR's, and thereafter decrements. In normal operation, the processor must direct clear the status bit to negate the interrupt request (see Figure 15).

FIGURE 15 — PERIODIC INTERRUPT GENERATOR



* Analog representation of counter value

Square Wave Generator

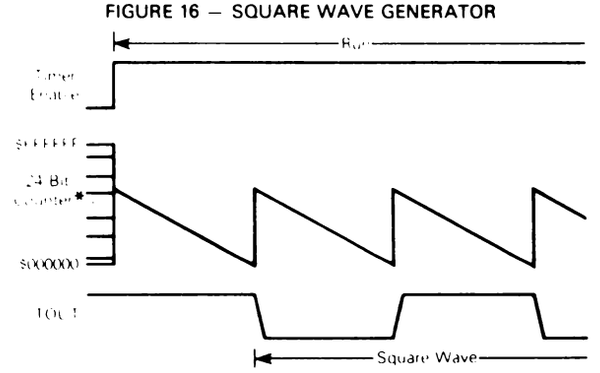
7	6	5	4	3	2	1	0
TOUT/TIACK Control			Z D Ctrl	*	Clock Control		Timer Enable
1	0	X	0	0	00 or 1X		changed

In this configuration the timer produces a square wave at the TOUT pin. The TOUT pin is connected to the user's circuitry and the TIACK pin is not used. The TIN pin may be used as a clock input.

The processor loads the Counter Preload Registers and Timer Control Register, and then enables the timer. When the 24 bit counter passes from $000001 to $000000 the ZDS status bit is set and the TOUT (square wave output) pin is toggled. At the next clock to the 24 bit counter it is again loaded with the contents of the CPRs, and thereafter decrements. In this application there is no need for the processor to direct clear the ZDS status bit, however, it is possible for the processor to sync itself with the square wave by clearing the ZDS status bit, then polling it. The processor may also read the TOUT level at the Port C address.

Note that the PC3/TOUT pin functions as PC3 following the negation of RESET. If used in the square wave configuration a pullup resistor may be required to keep a known level prior to programming. Prior to enabling the timer, TOUT is high (see Figure 16).

FIGURE 16 — SQUARE WAVE GENERATOR



* Analog representation of counter value

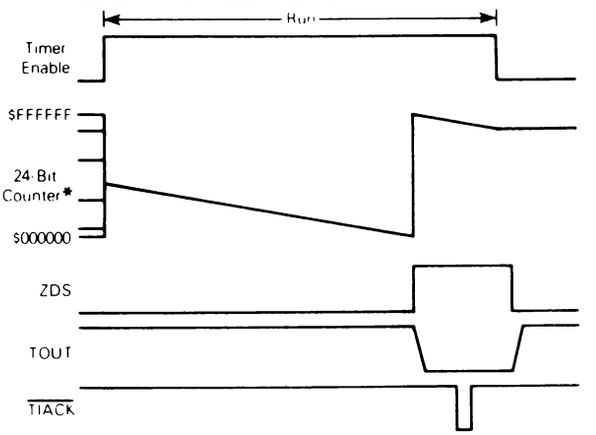
Interrupt After Timeout

7	6	5	4	3	2	1	0
TOUT/TIACK Control			Z D Ctrl	*	Clock Control		Timer En
1	X	1	1	0	00 or 1X		changed

In this configuration the timer generates an interrupt after a programmed time period has expired. The TOUT pin is connected to the system's interrupt request circuitry and the TIACK pin may be an interrupt acknowledge input to the timer. The TIN pin may be used as a clock input.

This configuration is similar to the periodic interrupt generator except that the Zero Detect Control bit is set. This forces the counter roll over after Zero Detect is reached, rather than reloading from the CPRs. When the processor takes the interrupt it can halt the timer and read the counter. This allows the processor to measure the delay time from Zero Detect (interrupt request) to entering the service routine. Accurate knowledge of the interrupt latency may be useful in some applications (see Figure 17).

FIGURE 17 — SINGLE INTERRUPT AFTER TIMEOUT



* Analog representation of counter value

Elapsed Time Measurement

Elapsed time measurement takes several forms, two are described below.

System Clock

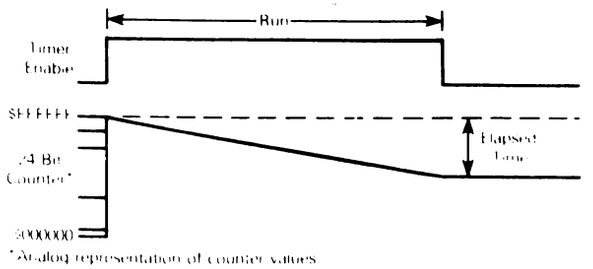
7	6	5	4	3	2	1	0
TOUT/TIACK Control			Z D Ctrl	*	Clock Control		Timer Enable
0	0	X	1	0	0	0	changed

This configuration allows time interval measurement by software. No timer pins are used.

The processor loads the Counter Preload Registers (generally with all 1s) and Timer Control Register, and then enables the timer. The counter decrements until the ending event takes place. When it is desired to read the time interval, the processor must halt the timer, then read the counter.

For applications in which the interval could have exceeded that programmable in this timer, interrupts can be counted to provide the equivalent of additional timer bits. At the end, the timer can be halted and read (see Figure 18).

FIGURE 18 — ELAPSED TIME MEASUREMENT



*Analog representation of counter values

External Clock

7	6	5	4	3	2	1	0
TOUT/$\overline{\text{TIACK}}$ Control			Z/D Ctrl	*	Clock Control		Timer Enable
0	0	X	1	0	1	X	Changed

This configuration allows measurement (counting) of the number of input pulses occurring in an interval in which the counter is enabled. The TIN input pin provides the input pulses. Generally the TOUT and $\overline{\text{TIACK}}$ pins are not used.

This configuration is identical to the Elapsed Time Measurement System Clock configuration except that the TIN pin is used to provide the input frequency. It can be connected to a simple oscillator, and the same methods could be used. Alternately, it could be gated off and on externally and the number of cycles occurring while in the run state can be counted. However, minimum pulse width high and low specifications must be met.

Device Watchdog

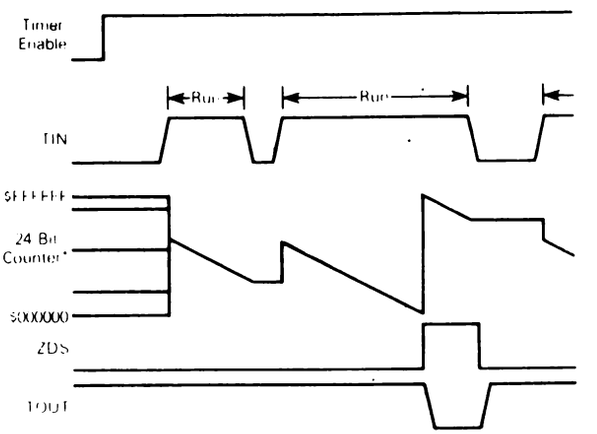
7	6	5	4	3	2	1	0
TOUT/$\overline{\text{TIACK}}$ Control			Z/D Ctrl	*	Clock Control		Timer Enable
1	X	1	1	X	0	1	Changed

This configuration provides the watchdog function needed in many systems. The TIN pin is the timer input whose period at the high (1) level is to be checked. Once allowed by the processor, the TIN input pin controls the run/halt mode. The TOUT pin is connected to external circuitry requiring notification when the TIN pin has been asserted longer than the programmed time. The $\overline{\text{TIACK}}$ pin (interrupt acknowledge) is only needed if the TOUT pin is connected to interrupt circuitry.

The processor loads the Counter Preload Register and Timer Control Register, and then enables the timer. When the TIN input is asserted (1, high) the timer transfers the contents of the Counter Preload Register to the counter and begins counting. If the TIN input is negated before Zero Detect is reached, the TOUT output and the ZDS status bit remain negated. If Zero Detect is reached while the TIN input is still asserted the ZDS status bit is set and the TOUT output is asserted. (The counter rolls over and keeps on counting.)

In either case, when the TIN input is negated the ZDS status bit is 0, the TOUT output is negated, the counting stops, and the prescaler is forced to all 1s (see Figure 19).

FIGURE 19 — DEVICE WATCHDOG



*Analog representation of counter value

BUS INTERFACE CONNECTION

The PI/T has an asynchronous bus interface, primarily designed for use with the MC68000 microprocessor. With care, however, it can be connected to synchronous microprocessor buses. This section completely describes the PI/T's bus interface, and is intended for the asynchronous bus designer unless otherwise mentioned.

In an asynchronous system the PI/T CLK may operate at a significantly different frequency, either higher or lower, than the bus master and other system components, as long as all bus specifications are met. The MC68230 CLK pin has the same specifications as the MC68000 CLK, and must not be gated off at any time.

The following signals generate normal read and write cycles to the PI/T: $\overline{\text{CS}}$ (Chip Select), R/$\overline{\text{W}}$ (Read/Write), RS1-RS5 (five Register Select bits), D0-D7 (the 8-bit bidirectional data bus), and $\overline{\text{DTACK}}$ (Data Transfer Acknowledge). To generate interrupt acknowledge cycles PC6/$\overline{\text{PIACK}}$ or PC7/$\overline{\text{TIACK}}$ is used instead of $\overline{\text{CS}}$, and the Register Select pins are ignored. No combination of the following pins may be asserted simultaneously: $\overline{\text{CS}}$, $\overline{\text{PIACK}}$, or $\overline{\text{TIACK}}$.

READ CYCLES VIA CHIP SELECT

This catagory includes all register reads, except port or timer interrupt acknowledge cycles. When $\overline{\text{CS}}$ is asserted, the Register Select and R/$\overline{\text{W}}$ inputs are latched internallly. They must meet small setup and hold time requirements with respect to the asserted edge of $\overline{\text{CS}}$. (See the AC ELECTRICAL CHARACTERISTICS table.) The PI/T is *not* protected against aborted (shortened) bus cycles generated by an Address Error or Bus Error exception in which it is addressed.

Certain operations triggered by normal read (or write) bus cycles are not complete within the time allotted to the bus cycle. One example is transfers to/from the double-buffered latches that occur as a result of the bus cycle. If the bus master's CLK is significantly faster than the PI/T's the possibility exists that, following the bus cycle, $\overline{\text{CS}}$ can be

negated then re-asserted before completion of these internal operations. In this situation the PI/T does not recognize the re-assertion of $\overline{CS}$ until these operations are complete. Only at that time does it begin the internal sequencing necessary to react to the asserted $\overline{CS}$. Since $\overline{CS}$ also controls the $\overline{DTACK}$ response, this "bus cycle recovery time" can be related to the CLK edge on which $\overline{DTACK}$ is asserted for that cycle. The PI/T will recognize the subsequent assertion of $\overline{CS}$ three (3) CLK periods after the CLK edge on which $\overline{DTACK}$ was previously asserted.

The Register Select and R/$\overline{W}$ inputs pass through an internal latch that is transparent when the PI/T can recognize a new $\overline{CS}$ pulse (see above paragraph). Since the internal data bus of the PI/T is continuously enabled for read transfers, the read access time (to the data bus buffers) begins when the Register Selects are stabilized internally. Also, when the PI/T is ready to begin a new bus cycle, the assertion of $\overline{CS}$ enables the data bus buffers within a short propagation delay. This does not contribute to the overall read access time unless $\overline{CS}$ is asserted significantly after the Register Select and R/$\overline{W}$ inputs are stabilized (as may occur with synchronous bus microprocessors).

In addition to Chip Select's previously mentioned duties, it controls the assertion of $\overline{DTACK}$ and latching of read data at the data bus interface. Except for controlling input latches and enabling the data bus buffers, all of these functions occur only after $\overline{CS}$ has been recognized internally and synchronized with the internal clock. Chip Select is recognized on the falling edge of the CLK if the setup time is met, $\overline{DTACK}$ is asserted (low) on the next falling edge of the CLK. Read data is latched at the PI/T's data bus interface at the same time $\overline{DTACK}$ is asserted. It is stable as long as Chip Select remains asserted independent of other external conditions.

From the above discussion it is clear that if the $\overline{CS}$ setup time prior to the falling edge of the CLK is met, the PI/T can consistently respond to a new read or write bus cycle every four (4) CLK cycles. This fact is especially useful in designing the PI/T's clock in synchronous bus systems not using $\overline{DTACK}$. (An extra CLK period is required in interrupt acknowledge cycles, see Read Cycles via Interrupt Acknowledge.)

In asynchronous bus systems in which the PI/T's CLK differs from that of the bus master, generally there is no way to guarantee that the $\overline{CS}$ setup time with respect to the PI/T CLK is met. Thus, the only way to determine that the PI/T recognized the assertion of $\overline{CS}$ is to wait for the assertion of $\overline{DTACK}$. In this situation, all latched bus inputs to the PI/T must be held stable until $\overline{DTACK}$ is asserted. These include Register Select, R/$\overline{W}$, and write data inputs (see below).

System specifications impose a maximum delay from the trailing (negated) edge of Chip Select to the negated edge of $\overline{DTACK}$. As system speeds increase this becomes more difficult to meet with a simple pullup resistor tied to the $\overline{DTACK}$ line. Therefore, the PI/T provides an internal active pullup device to reduce the rise time, and a level-sensitive circuit that later turns this device off. $\overline{DTACK}$ is negated asynchronously as fast as possible following the rising edge of Chip Select, then three-stated to avoid interference with the next bus cycle.

The system designer must take care that $\overline{DTACK}$ is negated and three-stated quickly enough after each bus cycle to avoid interference with the next one. With the MC68000 this necessitates a relatively fast external path from the data strobe to $\overline{CS}$ going negated.

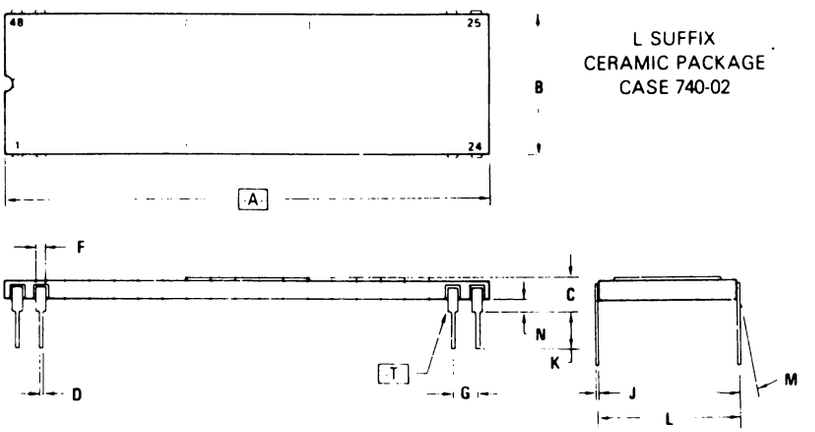
WRITE CYCLES

In many ways write cycles are similar to normal read cycles (see above). On write cycles, data at the D0-D7 pins must meet the same setup specifications as the Register Select and R/$\overline{W}$ lines. Like these signals, write data is latched on the asserted edge of $\overline{CS}$, and must meet small setup and hold time requirements with respect to that edge. The same bus cycle recovery conditions exist as for normal read cycles. No other differences exist.

READ CYCLES VIA INTERRUPT ACKNOWLEDGE

Special internal operations take place on PI/T interrupt acknowledge cycles. The Port Interrupt Vector Register or the Timer Interrupt Vector Register are implicitly addressed by the assertion of PC6/$\overline{PIACK}$ or PC7/$\overline{TIACK}$, respectively. The signals are first synchronized with the falling edge of the CLK. One clock period after they are recognized the data bus buffers are enabled and the vector is driven onto the bus. $\overline{DTACK}$ is asserted after another clock period to allow the vector some setup time prior to $\overline{DTACK}$. $\overline{DTACK}$ is negated, then three-stated as with normal read or write cycle, when $\overline{PIACK}$ or $\overline{TIACK}$ is negated.

PACKAGE DIMENSIONS



L SUFFIX
CERAMIC PACKAGE
CASE 740-02

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	60.35	61.57	2.376	2.424
B	14.63	15.34	0.576	0.604
C	3.05	4.32	0.120	0.160
D	0.381	0.533	0.015	0.021
F	0.762	1.397	0.030	0.055
G	2.54 BSC		0.100 BSC	
J	0.203	0.330	0.008	0.013
K	2.54	4.19	0.100	0.165
L	14.99	15.65	0.590	0.616
M	0°	10°	0°	10°
N	1.016	1.524	0.040	0.060

NOTES
1. DIMENSION A IS DATUM
2. POSTIONAL TOLERANCE FOR LEADS
   ⌖ Ø 0.25 (0.010) M T A M
3. T IS SEATING PLANE.
4. DIMENSION "L" TO CENTER OF LEADS WHEN FORMED PARALLEL
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5, 1973.

NCR 5386 SCSI Protocol Controller Data Sheet

NCR

NCR Microelectronics Division, Colorado Springs

This document contains the latest information available at the time of publication. However, NCR reserves the right to modify the contents of this material at any time. Also, all features, functions and operations described herein may not be marketed by NCR in all parts of the world. Therefore, before using this document, consult your NCR representative or NCR office for the information that is applicable and current.

TABLE OF CONTENTS

SECTION 1
GENERAL DESCRIPTION

The NCR SCSI Protocol Controller (SPC) is designed to accommodate the Small Computer Systems Interface (SCSI) as defined by the ANSI X3T9.2 committee. The SPC operates in both the Initiator and Target roles and can therefore be used in host adapter and control unit designs. This device supports arbitration, including reselection, and is intended to be used in systems that require either open collector or differential pair transceivers.

The NCR 5386 SCSI Protocol Controller communicates with the system microprocessor as a peripheral device. The chip is controlled by reading and writing several internal registers which may be addressed as standard or memory mapped I/O. A 24-bit Transfer Counter and the appropriate handshake signals accomodate large DMA transfers with minimal processor intervention. Since the NCR 5386 interrupts the MPU when it detects a bus condition that requires servicing, the MPU is freed from polling or controlling any of the SCSI bus signals.

Below is a list of important features:

SCSI INTERFACE

*Supports ANSI X3T9.2 SCSI Standard

- Asynchronous data transfers to 2.0 MBPS
- Supports both Initiator and Target roles
- Parity generation with optional checking
- Supports arbitration
- Controls all bus signals except Reset
- Doubly-buffered Data Register

MPU INTERFACE

*Versatile MPU Bus Interface

- Memory or I/O mapped MPU interface
- DMA or programmed I/O transfers
- 24-bit Internal Transfer Counter
- Programmable (Re) Selection Timeouts
- Interrupts MPU on all bus conditions requiring service
- SCSI pass parity optional with checking

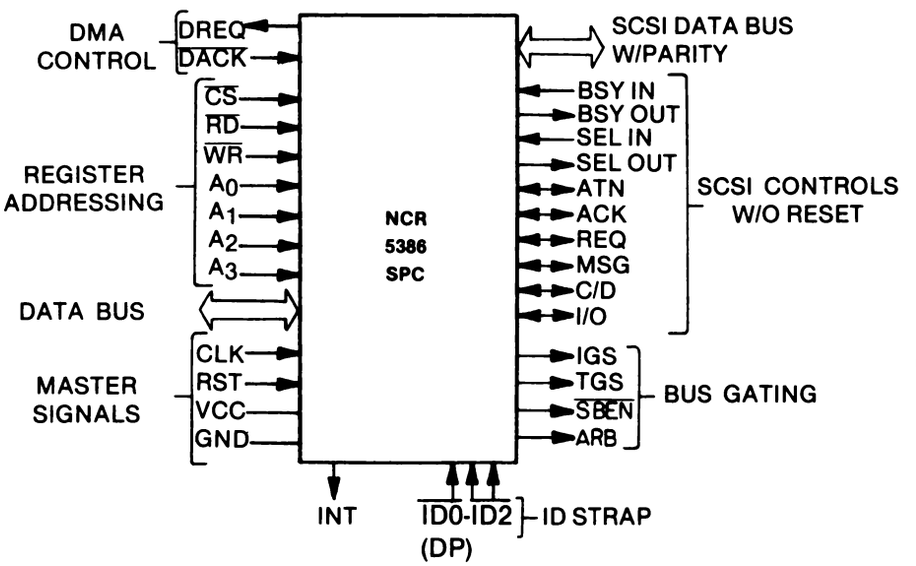


Figure 1.1
Functional Pin Grouping

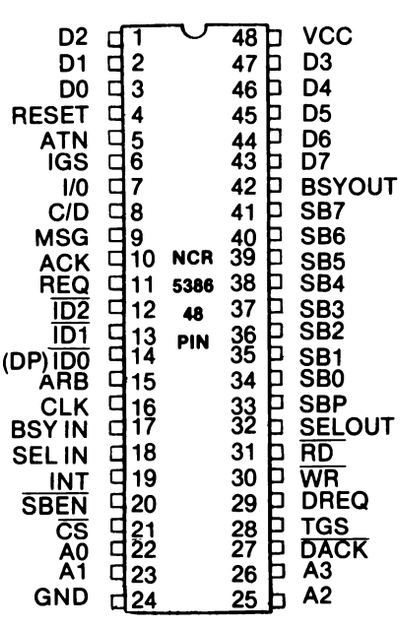


Figure 1.2
Pinout

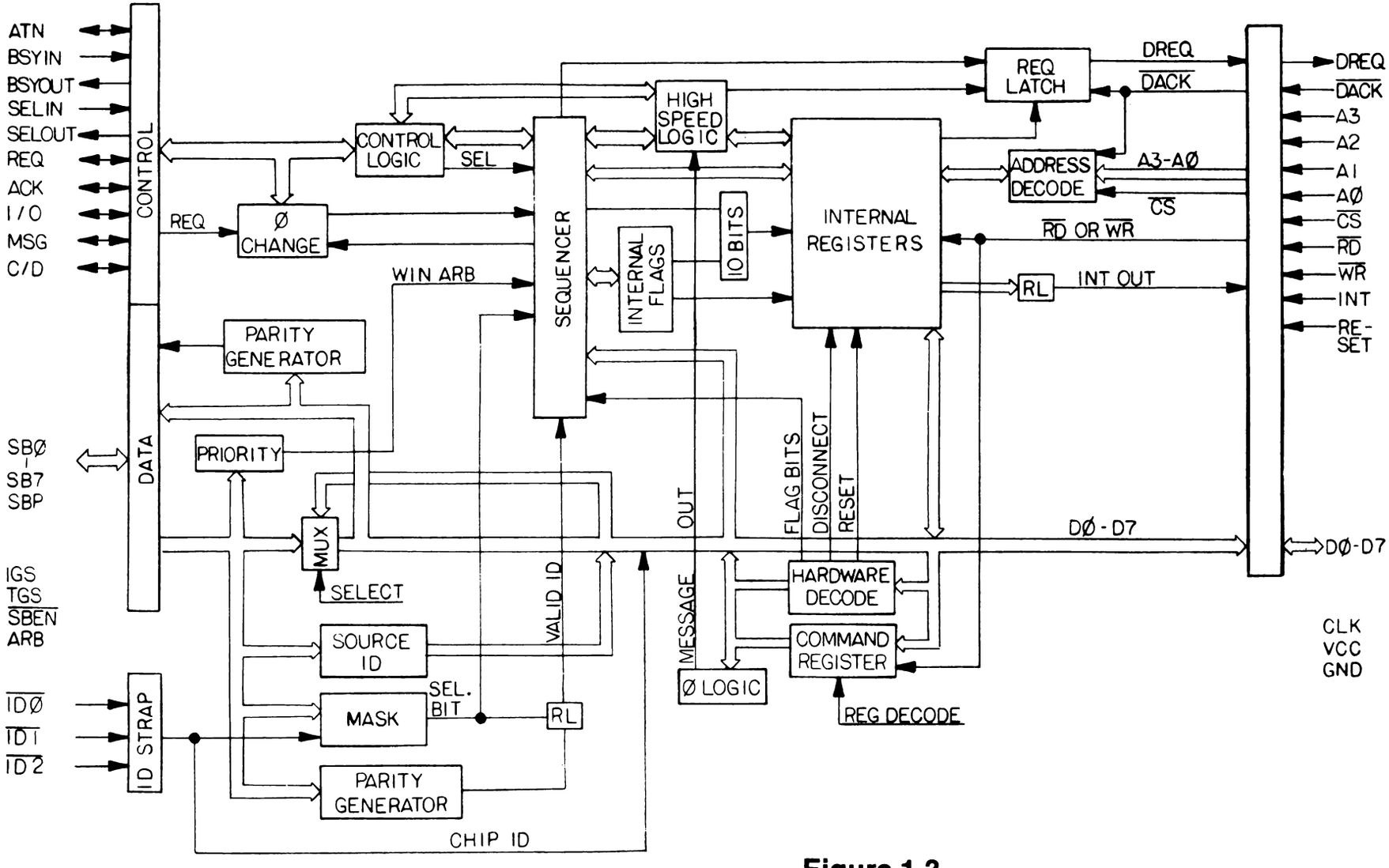


Figure 1.3
NCR 5386 Block Diagram

SECTION 2
PIN DESCRIPTION

2.1 MICROPROCESSOR INTERFACE SIGNALS

CLK	16	Symmetrical square wave signal which generates internal chip timing. Maximum frequency is 10 MHz.
RESET	4	When high (1), this signal forces the chip into a reset state. All current operations are terminated. Internal storage elements are cleared and self-diagnostics are performed.
D0-D7	3-1 47-43	These signals comprise an active high data bus. It is intended that these signals be connected to the microprocessor data bus.
INT	19	This signal is used to interrupt the microprocessor for various bus conditions that require service. INT is set high for request and cleared when the chip is reset or the Interrupt Register is read.
$\overline{WR}$	30	Write pulse (active low) is used to strobe data from the data bus into an internal register which has been selected.
$\overline{RD}$	31	Read pulse (active low) is used to read data from an internal register that has been selected. The contents of the register are strobed onto the data bus.
$\overline{CS}$	21	When low (0), this signal enables reading from or writing to the internal register which has been selected.
A0-A3	22, 23, 25, 26	These signals are used in conjunction with $\overline{CS}$, to address all the internal registers.
DREQ	29	Data request. When high (1), this signal indicates that the internal Data Register has a byte to transfer (inputting from the SCSI bus) or needs a byte to transfer (outputting to the SCSI bus). This signal becomes active only if the DMA mode bit in the Command Register is on. It is cleared when $\overline{DACK}$ becomes active.
$\overline{DACK}$	27	Data acknowledge. When low (0), this signal resets DREQ and selects the Data Register for input or output. $\overline{DACK}$ acts as a chip select for the Data Register when in the DMA mode. $\overline{DACK}$ and $\overline{CS}$ must never be active at the same time.

2.2 SCSI INTERFACE SIGNALS

$\overline{ID0}$-$\overline{ID2}$ (DP)	14-12	These active low signals determine the three-bit code of the SCSI bus ID assigned to the chip. External pullup resistors are required only if tied to switches or straps. (Optionally, pin 14 may be used as the data bus parity signal. In this case, the device ID is programmed by the system processor and pins 12 and 13 are not used. Refer to Section 4.6, "ID Register," for a detailed description.)
SB0-SB7, SBP	34-41, 33	Active high data bus. These signals comprise the SCSI data bus and are intended to be connected to the external SCSI bus transceivers.
BSYIN	17	When high (1), this signal indicates to the chip that the SCSI BSY signal is active.
BSYOUT	42	When high (1), the chip is asserting the BSY signal to the SCSI bus.
SELIN	18	When high (1), this signal indicates to the chip that the SCSI SEL signal is active.
SELOUT	32	When high (1), the chip is asserting the SEL signal to the SCSI bus.
ATN	5	INITIATOR ROLE: The chip asserts this signal when the microprocessor requests the attention condition or a parity error has been detected in a byte received from the SCSI bus. TARGET ROLE: This signal is an input which indicates the state of the ATN signal on the SCSI bus.
ACK	10	INITIATOR ROLE: The chip asserts this signal in response to REQ for a byte transfer on the SCSI bus. TARGET ROLE: This signal is an input which, when active, indicates a response to the REQ signal.
REQ	11	INITIATOR ROLE: This signal is an input which, when active, indicates that the Target is requesting a byte transfer on the SCSI bus. TARGET ROLE: Asserted by the chip to request a byte transfer on the SCSI bus.
MSG, C/D, I/O	9, 8, 7	INITIATOR ROLE: These signals are inputs which indicate the current SCSI bus phase. TARGET ROLE: The chip drives these signals to indicate the current bus phase.
IGS	6	Initiator Group Select. When high (1), this signal indicates to the external SCSI drivers that the chip is controlling in the Initiator role. Its purpose is to enable the external drivers for ATN and ACK. When low (0), ATN and ACK should be received.
TGS	28	Target Group Select. When high (1), this signal indicates to the external SCSI drivers that the chip is controlling in the Target role. Its purpose is to enable the external drivers for REQ, MSG, C/D, and I/O. When low (0), REQ, MSG, C/D, and I/O should be received.

$\overline{\text{SBEN}}$	20	SCSI data Bus Enable. When low (0), this signal directly enables the external SCSI data bus drivers.
ARB	15	Arbitration phase. When high (1), this signal enables the external circuitry to place the ID bit on the SCSI bus for the Arbitration phase.

POWER SIGNALS

VCC	48	+ 5 V input
GND	24	Signal reference input

SECTION 3
ELECTRICAL CHARACTERISTICS PRELIMINARY

OPERATING CONDITIONS

PARAMETER	SYMBOL	MIN	MAX	UNITS
Supply Voltage	V_{DD}	4.75	5.25	VDC
Supply Current	I_{DD}		300	mA
Ambient Temperature	T_A	0	70	°C

INPUT SIGNAL REQUIREMENTS

PARAMETER	CONDITIONS	MIN	MAX	UNITS
High-level Input, V_{IH}		2.0	5.25	VDC
Low-level Input, V_{IL}		-0.3	0.8	VDC
High-level Input Current, I_{IH}	$V_{IH} = 5.25V$		10	μA
Low-level Input Current, I_{IL}	$V_{IL} = 0V$		-10	μA

OUTPUT SIGNAL REQUIREMENTS (Except $\overline{\text{SBEN}}$, IGS, and TGS)

PARAMETER	CONDITIONS	MIN	MAX	
High-level Output Voltage, V_{OH}	$V_{DD} = 4.75V$ @ $I_{OH} = -400\mu A$	2.4	—	VDC
Low-level Output Voltage, V_{OL}	$V_{DD} = 4.75V$ @ $I_{OL} = 2.0mA$	—	0.4	VDC

$\overline{\text{SBEN}}$, IGS, and TGS SIGNALS

PARAMETER	CONDITIONS	MIN	MAX	UNITS
High-level Output Voltage, V_{OH}	$V_{DD} = 4.75V$ @ $I_{OH} = -400\mu A$	2.4	—	VDC
Low-level Output Voltage, V_{OL}	$V_{DD} = 4.75V$ @ $I_{OL} = 4.0mA$	—	0.4	VDC

SECTION 4
INTERNAL REGISTERS

4.0 GENERAL

The NCR SCSI Protocol Controller has a set of internal registers which are used by the microprocessor to direct the operation of the SCSI bus. These registers are read (written) by activating $\overline{CS}$ with an address on A3-A0 and then issuing a $\overline{RD}$/($\overline{WR}$) pulse. They can be made to appear to a microprocessor as standard I/O ports or as memory-mapped I/O ports depending on the external circuitry that controls $\overline{CS}$. The following sections describe the operation of these internal registers.

REGISTER SUMMARY

A3	A2	A1	A0	R/W	REGISTER NAME
0	0	0	0	R/W	Data Register
0	0	0	1	R/W	Command Register
o	0	1	0	R/W	Control Register
0	0	1	1	R/W	Destination ID
0	1	0	0	R	Auxiliary Status
0	1	0	1	R	ID Register
0	1	1	0	R	Interrupt Register
0	1	1	1	R	Source ID
1	0	0	1	R	Diagnostic Status
1	1	0	0	R/W	Transfer Counter (MSB)
1	1	0	1	R/W	Transfer Counter (2nd BYTE)
1	1	1	0	R/W	Transfer Counter (LSB)
1	1	1	1	R/W	Reserved for Testability

4.1 DATA REGISTERS

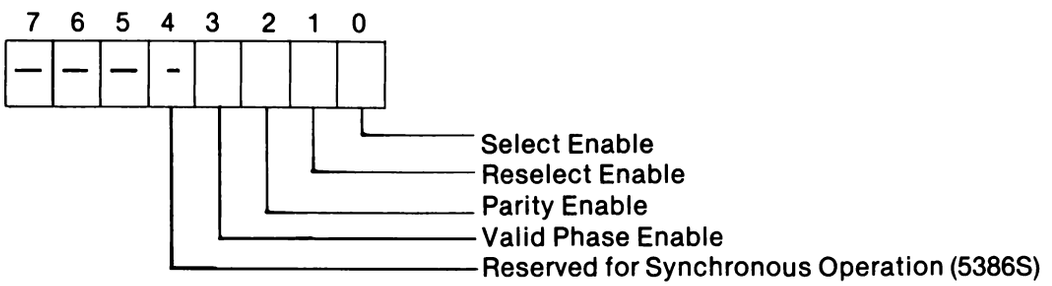
The Data Registers are used to transfer SCSI commands, data, status and message bytes between the microprocessor data bus and the SCSI bus. These are eight-bit registers which are doubly-buffered in order to support maximum throughput. In the non-DMA mode, the microprocessor reads from (writes to) Data Register 1 by activating $\overline{CS}$ with A3-A0 = 0000 and issuing a $\overline{RD}$/($\overline{WR}$) pulse. Two bits have been provided in the Auxiliary Status Register to indicate the status of both Data Register 1 and Data Register 2. In the DMA mode, the DMA logic reads from (writes to) Data Register 1 by responding to DREQ with $\overline{DACK}$ and issuing a $\overline{RD}$/($\overline{WR}$) pulse. The SCSI bus reads from or writes to Data Register 2 when the chip is connected as an Initiator or Target and the bus is in one of the Information Transfer Phases.

4.2 COMMAND REGISTER

The Command Register is an eight-bit register used to give commands to the SCSI chip. The microprocessor can write to (read from) the Command Register by activating $\overline{CS}$ with A3-A0 = 0001 and issuing a $\overline{WR}$/($\overline{RD}$) pulse. Writing to the Command Register causes the chip to execute the command that is written. The Command Register can be read; however, the chip resets the Command Register when the SPC sets an Interrupt. Therefore, one cannot guarantee that the data in the register will be correct after loading an interrupting command or enabling selection or reselection. To be safe, a copy of the last command issued should be stored in the microprocessor's memory. Immediate commands are not stored.

4.3 CONTROL REGISTER

This eight-bit read/write register is used for enabling certain modes of operation for the SCSI Protocol Controller. The microprocessor reads from (writes to) the Control Register by activating $\overline{CS}$ with A3-A0 = 0010 and issuing a $\overline{RD}$ ($\overline{WR}$) pulse.



BIT 7-4 Reserved

BIT 3 Valid Phase Enable — When this bit is a "1", the chip generates an interrupt for phase changes only when REQ becomes active. When bit 3 is a "0", the chip generates interrupts for phase changes that occur even though REQ may not be valid.

BIT 2 Parity Enable — When the parity enable bit is a "1", the chip generates and checks parity on all transfers on the SCSI bus. When the parity enable bit is a "0", the chip generates but does not check parity on bus transfers.

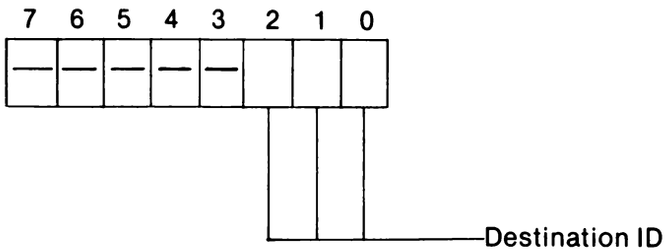
BIT 1 Reselect Enable — When this bit is a "1", the chip will respond to any attempt by a Target to reselect it. When the bit is a "0", the chip will ignore all attempts to reselect it.

BIT 0 Select Enable — When this bit is a "1", the chip will respond to any attempts to select it as a Target. When it is a "0", the chip will ignore all selections.

NOTE: After being reset and completing self-diagnostics, the control register will contain all zeros.

4.4 DESTINATION ID REGISTER

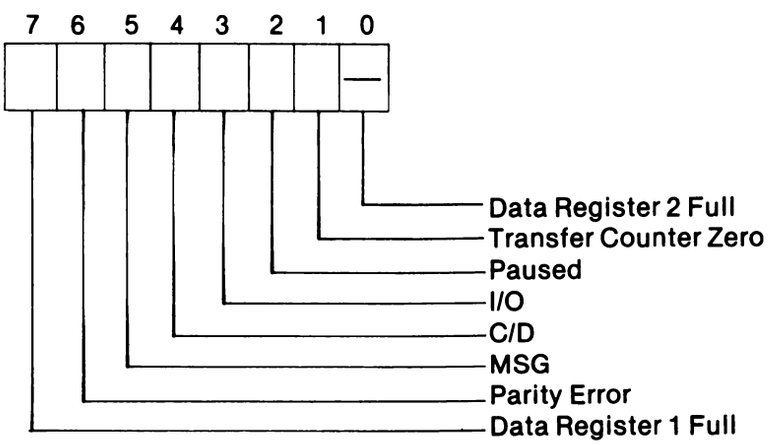
The Destination ID Register is an eight-bit register that is used to program the SCSI bus address of the destination device prior to issuing a Select or Reselect command to the chip. Bits 0-2 specify the address and bits 3-7 are always zeroes. The ID register is written (read) by activating $\overline{CS}$ with A3-A0 equal to "0011" and then pulsing $\overline{WR}$ ($\overline{RD}$).



4.5 AUXILIARY STATUS REGISTER

The Auxiliary Status Register is an eight-bit read-only register. It contains bits which indicate the status of the chip's operational condition. Some of these bits are used to determine the reason for interrupts. Therefore, the Auxiliary Status Register should always be read prior to reading the Interrupt Register when servicing interrupts. After the Interrupt Register is read, the Auxiliary Status Register bits needed to service the interrupt may change.

The Auxiliary Status Register is read by activating $\overline{CS}$ with A3-A0 = 0100 and then pulsing $\overline{RD}$. The individual bits of the Auxiliary Status Register are defined below.



BIT 7 Data Register 1 Full

This bit indicates the status of Data Register 1 and must be monitored by the microprocessor during non-DMA mode commands that use Data Register 1. When the DMA mode bit in the Command Register is off (0) and the command being executed is one of Send, Receive or Transfer Info commands (refer to Section 5.0, COMMANDS), data is transferred to (from) the chip by writing (reading) Data Register 1. Data Register 1 Full is set on (1) when data is written and turned off (0) when data is read. Therefore, Data Register 1 Full should be on before taking data from the chip, and off when sending data to the chip.

The Data Register 1 Full bits is always reset (to 0) at the time an interrupting type command is loaded into the Command Register. Therefore, when issuing such commands, the Command Register should be loaded prior to loading the Data Register 1 and monitoring the Data Register 1 Full flag.

BIT 6 Parity Error

When this bit is one, it indicates that the chip has detected a parity error on a byte of data received across the SCSI bus. It can be set when the chip is executing one of the Receive commands or the Transfer Info command (when the transfer is an input). This bit is reset after the Interrupt Register is read.

In Pass Parity mode, this bit will indicate whether MPU data had a parity error before being sent onto the SCSI bus. This parity error will not generate an interrupt to the MPU, but the receiving SCSI device will indicate an error was detected. The flag is useful in identifying where the error occurred.

BIT 3-5 I/O, C/D, MSG

These bits indicate the status of the SCSI I/O, C/D, and MSG signals at all times. They define the Information Phase type being requested by the Target. These signals are significant when servicing interrupts and the chip is logically connected to the bus in the Initiator role. An interrupt will occur with any phase change. This allows the Initiator to prepare for the next phase of data transfer. These bits are only held while INT is active. The bits are coded as follows:

I/O	C/D	MSG	BUS PHASE
0	0	0	Data Out
0	0	1	Unspecified Info Out
0	1	0	Command
0	1	1	Message Out
1	0	0	Data In
1	0	1	Unspecified Info In
1	1	0	Status
1	1	1	Message In

BIT	2	Paused	When on (1), this bit indicates that the chip has aborted the command being executed in response to the Pause command. It is turned off when the interrupting type command code is loaded into the Command Register.
BIT	1	Transfer Counter Zero	This bit is provided to indicate the status of the 24-bit Transfer Counter. When on (1), it indicates that the Transfer Counter is equal to zero. It is intended to facilitate interrupt servicing. This bit is also set when the Single-byte bit in the Command Register is active.
BIT	0	Data Register 2 Full	Since the 5386 design includes a doubly-buffered data register, this bit is used to indicate if data has been transferred into the second data register. If bit 0 is a "1", Data Register 2 is full; if this bit is a "0", Data Register 2 is empty.

NOTE: The Auxiliary Status Register will contain the following pattern after a Reset and self-diagnostics: 00xxx010.

4.6 ID REGISTER

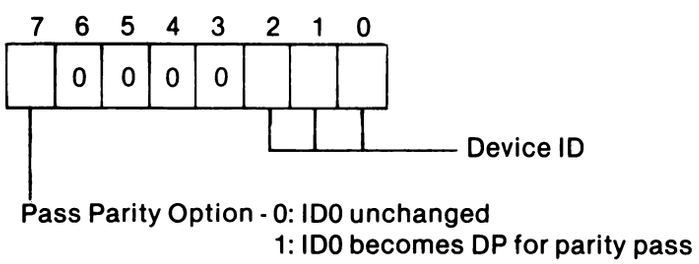
The ID Register operates in two configurations, the "strapped ID" mode or the "programmed ID" mode. If the ID register is written before the Control Register is written, the NCR 5386 assumes the "programmed ID" mode. In the "programmed ID" mode, pin 14 becomes the data bus parity signal (DP) and pins 12 and 13 are not used. If the Control Register is initialized and the ID register has not been written previously, then the device will assume the "strapped ID" mode which is identical to the NCR 5385E operation.

Strapped ID Configuration

The ID Register is an eight-bit read-only register in the "strapped ID" mode, which indicates the logical SCSI bus address occupied by the chip. Bits 0-2 directly reflect the logical inversion of the chip ID input signals ID0-ID2; the ID Register is active high whereas the ID input signals are active low. The ID Register allows the microprocessor to read the chip's SCSI bus address which would normally be strapped in hardware. Bits 3-6 of the ID Register will always be zeros. The ID Register is read by activating CS with A3-A0 equal to 0101 and then pulsing RD.

Programmed ID Configuration

The ID Register is an eight-bit read/write register in the "programmed ID" mode, which is intended to be programmed by the system MPU. Bits 0-2 directly reflect the logical SCSI device ID. Bit 7 indicates if pass parity feature is enabled. In the "programmed ID" mode, pin 14 becomes the parity signal for the data bus interface (DP) only if bit 7 is a "1", and pins 12 and 13 are not used.

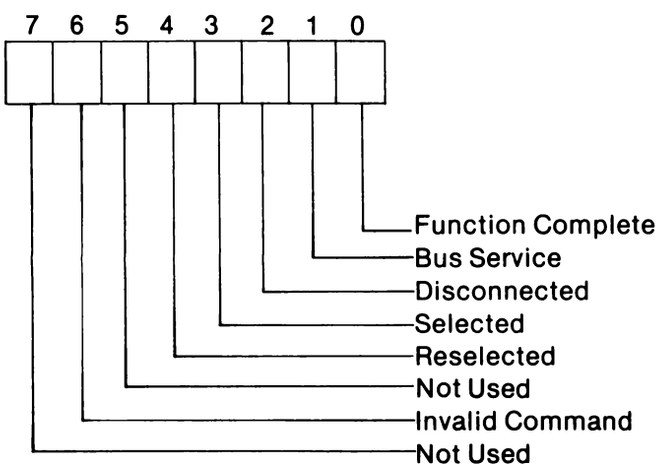


4.7 INTERRUPT REGISTER

The Interrupt Register is an eight-bit read-only register. It is used in conjunction with the Auxiliary Status Register to determine the reason for an interrupt condition. This register is read by activating $\overline{CS}$ with A3-A0 = 0110 and then pulsing $\overline{RD}$. When the Interrupt Register is read, it automatically resets itself (after the read is complete) and enables the chip for a new interrupt condition. Since the Parity Error bit in the Auxiliary Status Register is reset after a read of the Interrupt Register, and since I/O, C/D, and MSG are only held while INT is active, the Auxiliary Status Register should always be read prior to reading the Interrupt Register.

If a Selected or Reselected interrupt occurs after issuing a command that would normally cause an interrupt, the chip will ignore the last command issued. This allows the microprocessor to service the Selected or Reselected interrupt prior to proceeding with the other operation. An example of this situation is when the microprocessor issues a command to select a Target at about the same time another Target reselects the chip. If the chip sees the reselection first, the microprocessor will receive an interrupt for the reselection, and the chip will ignore the Select command, which would now be invalid since the chip is now logically connected on the SCSI bus to another device.

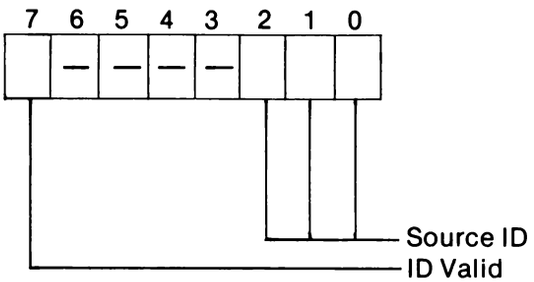
Individual interrupt conditions are described below. (Note: that for all cases, an interrupt condition is on, when the corresponding bit is a one (1), and off when zero (0).)



BIT 7	Not Used	May be either (1) or (0).
BIT 6	Invalid Command	When on (1), this bit indicates that the last command loaded into the Command Register is not valid.
BIT 5	Not Used	(Reserved)
BIT 4	Reselected *	This interrupt will be on (1) when the chip has been reselected by another SCSI device. After setting this interrupt, the chip is logically connected to the bus in an Initiator role and is waiting for the Target to send REQ or disconnect from the bus.
BIT 3	Selected *	This interrupt will be on (1) whenever the chip has been selected by another SCSI device. After setting this interrupt, the chip is logically connected to the bus in the Target role and is waiting for a command to be loaded into the Command Register. * The chip will become selected (reselected) only if the ID data byte put on the SCSI bus during the Selection (Reselection) Phase has good parity and not more than one ID other than the chip's own ID is on.
BIT 2	Disconnected	This interrupt will be set on (1) when the chip is connected to the bus in the Initiator role and the Target disconnects or when the chip is executing a Select or Reselect command and the destination device does not respond before the Transfer Counter times out.
BIT 1	Bus Service	When the chip is logically connected to the bus in the Initiator role, this bit will be set on (1) whenever the Target sends a REQ which the chip cannot automatically handle. This happens when the first REQ for connection is received or when the chip is executing a Transfer Info or Transfer Pad command and either the Transfer Counter is zero or the Target changes the Information Phase type. If the valid phase enable bit is not set, Bus Service interrupt may be set if a phase change occurs before REQ is seen; see "valid phase enable," section 4.3. This early notification will allow the Initiator extra time to prepare for a phase change in some systems. (Note: the chip may Generate Bus Service Interrupts for phases that never request transfers. This is not an error condition, merely transitional status of I/O, C/D, and MSG.) If the chip is logically connected in the Target role, this bit will be set on (1) whenever the Initiator asserts ATN. When indicating ATN the Bus Service interrupt may occur with a Selected interrupt or with a Function Complete interrupt.
BIT 0	Function Complete	When this bit is on (1), it indicates that the last interrupting command has completed. It is the normal successful completion interrupt for Select, Reselect, Send and Receive commands (Refer to Section 5.0, COMMANDS). During any of the Receive commands, it is set on (1) along with the parity error bit as soon as a parity error is detected. A Bus Service Interrupt may also occur simultaneously with the Function Complete if an ATN signal was activated during a Send or a Receive command. The Function Complete interrupt is also generated at the end of a Message In phase for a Transfer Info command. (See TRANSFER INFO command for details.)

4.8 SOURCE ID REGISTER

The Source ID Register is an eight-bit read-only register which contains the three-bit encoded ID of the last device which Selected or Reselected the chip. The following is the format of the Source ID Register.



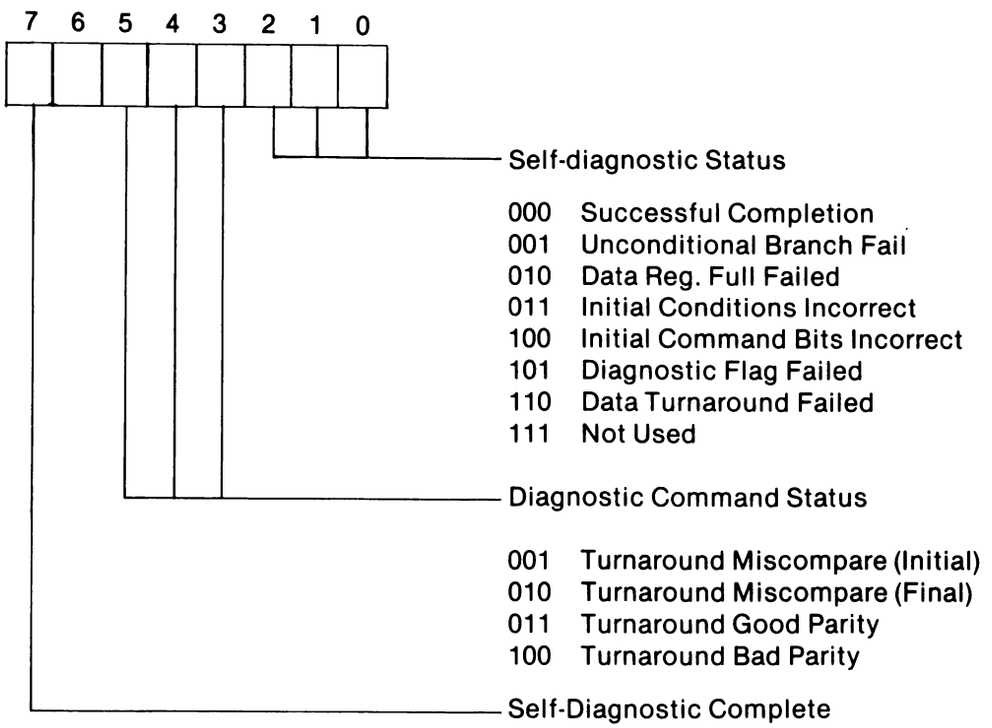
The ID Valid bit indicates that the source device placed its own ID bit on the SCSI bus during the Selection Phase. The SPC chip has encoded the source ID and placed it in bits 2-0. This information remains valid until the chip disconnects from the SCSI bus, at this time the ID Valid bit is reset.

4.9 DATA REGISTER 2

Data Register 2 is an eight-bit read-only register which may contain data that has been sent to the chip but not transferred across the bus. Bit 0 in the Auxiliary Status Register is used to determine if this register is full. The microprocessor can read Data Register 2 by activating $\overline{CS}$ with A3-A0 equal to 1000 and issuing a $\overline{RD}$ pulse.

4.10 DIAGNOSTIC STATUS REGISTER

The Diagnostic Status Register is an eight-bit read-only register which indicates the result of self-diagnostics and the last diagnostic command issued to the chip. The format of the Diagnostic Status Register is shown below.



Bit 7 = 1 indicates that self-diagnostics have been completed. (NOTE: A reset will clear bits 6-3 if possible). After a reset to the chip, the microprocessor should make sure that the Diagnostic Status Register contains the following pattern before attempting any commands: 10000000. This code indicates self-diagnostics are complete and no errors were detected. After a diagnostic command has been executed, bits 6-3 will contain the resulting status, but bit 7 and bits 2-0 are not affected.

The microprocessor may read the Diagnostic Status Register by activating $\overline{CS}$ with A3-A0 = 1001 and issuing a $\overline{RD}$ pulse.

If an error is detected during self-diagnostics, the proper status is loaded into the Diagnostic Status Register and the chip halts until a Reset command or a Reset signal is asserted. Refer to the Self-Diagnostic Status Code Summary for an explanation of the individual codes.

When a diagnostic command is issued to the chip, the chip will attempt to perform the function, load a status into bits 6-3, and initiate a Function Complete Interrupt.

4.10.1 SELF-DIAGNOSTIC STATUS CODE SUMMARY

000 - Successful Completion. The chip executed all self-diagnostics following a reset and detected no errors.

001 - Unconditional Branch Failed. The chip's internal sequencer attempted an unconditional branch and failed to reach the desired location.

010 - Data Register Full Failed. The chip attempted to set and reset the Data Register Full status bit in the Interrupt Register and failed.

011 - Initial Conditions Incorrect. The chip detected one of its internal initial conditions in the wrong state.

100 - Initial Command Bits Incorrect. The chip tested bits 6,4,2,1 and 0 of the Command Register and found at least one was not zero.

101 - Diagnostic Flag Failed. The chip failed in its attempt to set and reset its internal diagnostic flag.

110 - Data Turnaround Failed. During self-diagnostics the chip attempts to flush several bytes of data through its internal data paths. It also attempts to set and reset the Parity Error bit in the Interrupt Status Register. This status indicates that one of these operations failed.

4.11 TRANSFER COUNTER (THREE EIGHT-BIT COUNTERS)

The Transfer Counter is comprised of three, eight-bit register/counters. It is used by the chip for Send, Receive and Transfer commands that require more than a single byte of data to be transferred. It may also be used with Select and Reselect commands to set a timeout for no response. To write to (read from) the Transfer Counter, $\overline{CS}$ is activated with A3-A0 selecting a byte and then pulsing $\overline{WR}$ ($\overline{RD}$). The Transfer Counter is addressed as shown below.

A3	A2	A1	A0	SELECTED BYTE
1	1	0	0	Most Significant Byte
1	1	0	1	Middle Byte
1	1	1	0	Least Significant Byte

For Send, Receive and Transfer commands with single-byte not specified, the Transfer Counter specifies to the chip the maximum number of bytes to be sent or received before interrupting. The Transfer Counter must be loaded prior to issuing the command. When single-byte is specified, the chip neither uses nor alters the Transfer Counter. To facilitate servicing interrupts for commands that use the Transfer Counter, a bit is provided in the Auxiliary Status Register to indicate when the Transfer Counter is zero.

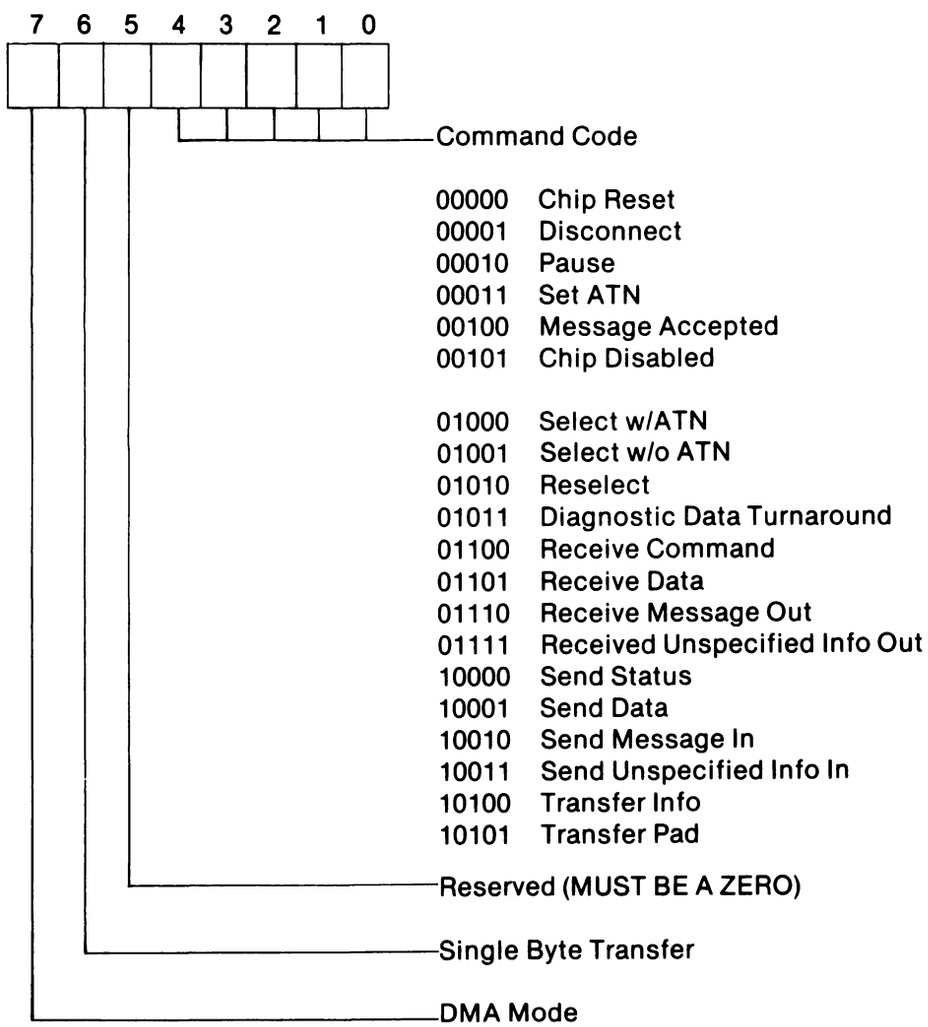
For Select and Reselect commands, the Transfer Counter specifies the number of time intervals (1024 CLK periods) that the chip will wait before automatically aborting the command due to no response (BSY) from the destination device. The Transfer Counter must be loaded prior to issuing the command. If the Transfer Counter is loaded with all zeroes, the timeout logic in the chip will be disabled, and the chip will not automatically abort the command due to no response.

SECTION 5
COMMANDS

This section defines command format, types, codes and operation. Commands are given to the chip by loading the Command Register.

5.1 COMMAND FORMAT

The bits in the Command Register are defined as follows.



BIT	7	DMA Mode	This bit is applicable only for commands that use the Data Register. When this bit is on (1), it indicates that data will be transferred to (from) the Data Register using the DMA signals DREQ and $\overline{\text{DACK}}$. When it is off (0), the microprocessor must monitor the state of the Data Register Full flag in the Auxiliary Status Register. Data is then transferred by using the appropriate input/output command.
BIT	6	Single Byte Transfer	When on (1), this bit indicates that only one byte of data is to be transferred for this command. The Transfer Counter will not be used or altered by the chip. Therefore, for common single byte message and status transfers, the Transfer Counter does not need to be loaded prior to issuing a command with this bit set. When this bit is off (0). the Transfer Counter is used by the chip to determine the length of the transfer for the command.
BIT	5	Reserved	This bit is not used and should always be programmed off (0).
BIT	4-0	Command Code	These bits are used to specify the command to be executed.

5.2 COMMAND TYPES

There are two types of commands; Immediate and Interrupting. All of the Immediate commands, except for Pause, cause immediate results within three clock cycles from the time the Command Register is loaded. The Pause command is explained in a later section (See PAUSE). Interrupting commands do not result in immediate action. Their completion is always flagged by an interrupt.

Command codes 00000-00111 specify Immediate commands. Immediate commands that are listed as reserved, will be ignored if issued to the SPC chip. Command codes 01000-10101 specify Interrupting commands. When one of these codes is loaded into the Command Register, a second Interrupting command code should not be loaded until after the interrupt has occurred for the first command. However, an Immediate type command may be loaded before the interrupt for an Interrupting command occurs. If a reserved Interrupting command code is issued, the chip will respond with an Invalid Command interrupt.

5.3 INVALID COMMANDS

The user of the chip can be in one of three states at any particular time: Disconnected, connected as an Initiator, or connected as a Target. Commands are valid only in specified states. If an invalid Immediate command is issued, the chip will ignore the command. If an Interrupting command is issued in an invalid state, or a reserved Interrupting command code is issued, an Invalid Command interrupt will result. The exceptions are described below:

The microprocessor must never issue any interrupting type command when the chip is not expecting such a command. Unpredictable results will occur in this case. The following is a list of user states in which the chip is not expecting an interrupting command:

1. The chip is currently processing an Interrupting type command and has not yet set the interrupt to signal the completion.
2. The chip is currently processing an Interrupting type command, a Pause command has been issued but the Paused bit in the Auxiliary Status Register has not been set.
3. The chip is connected as an Initiator, but the Target has not yet requested an Information Transfer.
4. The chip has completed a Transfer Info or Transfer Pad command and the Target has not requested additional information or has not changed the Information Phase.

In user states three and four, described above, the microprocessor must wait for a Bus Service, Disconnected, or Function Complete interrupt.

If an interrupting command is illegitimately issued in these states, no interrupt will occur for it, and it is likely that the current function will be altered.

5.4 COMMAND SUMMARY

Below is a summary that lists all commands. In the table the following abbreviations are used.

INT = INTERRUPTING D = DISCONNECTED I = CONNECTED AS AN INITIATOR
IMM = IMMEDIATE T = CONNECTED AS A TARGET

COMMAND CODE	COMMAND	TYPE	VALID STATES
00000	Chip Reset	IMM	D,I,T
00001	Disconnect	IMM	I,T
00010	Paused	IMM	D,T
00011	Set ATN	IMM	I
00100	Message Accepted	IMM	I
00101	Chip Disable	IMM	D,I,T
00110-00111	Reserved	IMM	
01000	Select w/ATN	INT	D
01001	Select w/o ATN	INT	D
01010	Reselect	INT	D
01011	Diagnostic	INT	D
01100	Receive Command	INT	T
01101	Receive Data	INT	T
01110	Receive Message Out	INT	T
01111	Receive Unspecified Info Out	INT	T
10000	Send Status	INT	T
10001	Send Data	INT	T
10010	Send Message In	INT	T
10011	Send Unspecified Info In	INT	T
10100	Transfer Info	INT	I
10101	Transfer Pad	INT	I
10110-11111	Reserved	INT	

5.5 COMMAND DEFINITIONS

5.5.1 CHIP RESET

Chip Reset immediately stops any chip operation and resets all registers, counters, etc. on the chip. It performs the same operation as the hardware "reset" input.

5.5.2 DISCONNECT

Upon receipt of this command, the chip immediately releases all SCSI bus signals and returns to a Disconnected idle state. For the Initiator role, this is the normal method of disconnecting from the bus when a transfer is complete. For the Initiator role, Disconnect may be used to release the bus signals as a result of a timeout condition. In this case, the chip ignores the Target and is left in the Disconnected state. For the Disconnected state, it is not valid to issue a Disconnect command. If issued, the chip will ignore this command.

5.5.3 PAUSE

Pause is an Immediate command that is valid in the Disconnected state or when logically connected to the bus as a Target device. Pause is not valid when connected as an Initiator.

When connected as a Target, the Pause command provides a means of halting a Send or Receive command without having to wait for the transfer to complete. When Pause is issued, it immediately sets a flag in the chip. Within one byte transfer cycle, the chip recognizes the flag, aborts the Send or Receive operation, and then sets the Paused status bit in the Auxiliary Status Register. At this time, the chip is still connected to the bus in the Target role, and it is waiting for another command.

The Pause command stops the Send or Receive command in an orderly manner leaving the Transfer Counter in a valid state that indicates the remaining number of bytes to be transferred. Also no REQ or ACK is asserted on the bus and no data is left in the chip waiting to be transferred. An operation that is paused may be resumed, if desired, simply by reloading the original command into the Command Register. (Note: after issuing the Pause while executing Send or Receive, it is necessary to continue transferring data with the chip (due to double-buffering) until the Paused status bit is set or an interrupt occurs.)

When in the disconnected state, Pause may be issued to abort a Select or Reselect command. After a Select or Reselect command is issued and before an interrupt occurs, a Pause command may be issued to abort the operation. The Pause command immediately sets an internal flag. If the chip has not yet won arbitration, it sets the Paused bit in the Auxiliary Status Register and waits in the disconnected state for another command. If the chip has won arbitration, it releases the bus by dropping the two ID bits with SELOUT on for a minimum of 100μs, checks for no BSYIN, and then releases the bus. After this procedure, it sets the Paused bit in the Auxiliary Status Register and waits for another command in the Disconnected state.

Since Pause is an Immediate command, it does not cause an interrupt. As previously noted, the chip sets the Paused status bit to indicate that is has been executed. If an interrupt-causing event occurs before the chip sees the pause flag set, the chip will set the interrupt. In this case, the Paused status bit is not set by the chip either before or after the interrupt. In all cases, an interrupt-causing event will take precedence over Pause. For example, in the Target role if ATN is on when Pause is issued, a Bus Service interrupt will occur and the Paused status bit will not be set.

If the Pause command is issued when the chip is Disconnected, the Paused status bit will be set by the chip, provided it has not already detected a Selection or Reselection.

5.5.4 SET ATN

The Set ATN command causes ATN to be asserted immediately if the chip is connected as an Initiator. This command is invalid and ignored if issued when the chip is Disconnected or is operating in a Target role. The ATN signal is de-asserted in a Message Out phase when the transfer count becomes zero or one byte has been transferred (in a one-byte transfer command) during the execution of a Transfer Info command.

The chip automatically sets ATN in two cases:

1. If a Select w/ATN command is issued and arbitration is won.
2. If a parity error is detected on an input byte during execution of a Transfer Info command.

5.5.5 MESSAGE ACCEPTED

The Message Accepted command is an Immediate command that is valid only when connected as an Initiator. It is used after a Transfer Info or Pad command (See pages 29, 30 TRANSFER INFO and TRANSFER PAD) to indicate to the chip that ACK can be de-asserted for the last byte.

When an Initiator receives a message, a Transfer command is used. If the transfer is an input (I/O = 1) and the information is a message (MSG = 1, C/D = 1), the chip interrupts after receiving the last byte with a Function Complete interrupt. For this one special case, the chip also leaves ACK asserted on the bus. By interrupting and leaving ACK asserted, the chip gives the microprocessor a chance to interpret the message and set ATN, prior to ACK being de-asserted. This allows the chip to properly request a Message Out phase if the Initiator wants to send a "Reject Message" to the Target.

Message Accepted must always be issued after a Transfer Info for a Message In phase, whether or not Set ATN is issued, in order to have the chip de-assert ACK. If the Initiator wants to reject the message, Set ATN would be issued first followed by Message Accepted. If the message is not to be rejected, only Message Accepted is issued. (Note: until Message Accepted is issued, the Target will not send another REQ since ACK is still asserted.)

5.5.6 CHIP DISABLE

Chip Disable immediately stops all chip operations and logically disconnects it from the circuit. All outputs will be placed in a high impedance state and the chip will not respond to any commands (other than chip reset). The chip will also not respond to any activity on the SCSI bus. The only way to exit this condition is to activate the "reset" input or issue a Reset command.

5.5.7 SELECT w/ATN

This command causes the chip to attempt to select a Target. It may only be used if the SPC is in the Disconnected state. Any attempt to issue this command in another state will result in an Invalid Command interrupt. Before issuing this command, the microprocessor must load the Transfer Counter for a timeout on the Target's response. This value is computed according to the following formula:

Transfer Counter = Desired Timeout/ (1024 x Clock Period)

If the Transfer Counter is loaded with the value zero, the chip will wait indefinitely for a response from the Target being selected.

The microprocessor must also load the Destination ID Register with the three-bit code of the Target to be selected before issuing the Select w/ATN command.

When the chip detects the Select w/ATN command, it begins by attempting to arbitrate for control of the SCSI bus. If, at any time during arbitration the chip becomes selected or reselected, the Select w/ATN is aborted and forgotten and the chip will interrupt with one of the following conditions:

1. Selected
2. Selected and Bus Service
3. Reselected

If arbitration is won, the chip places the SCSI bus in the Selection phase with ATN asserted, and uses the Destination ID Register to identify the desired Target. At the same time, the chip begins a timer based on the value computed above. If the Target does not respond within the timeout period, the chip will disconnect from the bus and interrupt with the Disconnected flag set in the Interrupt Register. (Note: The microprocessor should never monitor the Transfer Counter Zero flag in the Auxiliary Status Register to determine when a timeout has occurred.) If the Target responds within the allotted time, the chip will interrupt with a Function Complete status. Control of the SCSI bus then belongs to the selected Target and after the interrupt status has been read, another interrupt may occur indicating either that the Target has disconnected or is requesting a transfer.

If the timeout is disabled and the Target does not respond, or if arbitration is not won, the only way to abort the Select w/ATN command is to issue the Pause command. After the Pause command is issued, it is still possible that the Function Complete or Disconnect interrupts may occur. This happens if one of the interrupts get set before the chip detects the Pause command, or if the Target responds while the chip is sequencing off the SCSI bus in a timeout condition. If the chip does not set either interrupt, it will set the Paused bit in the Auxiliary Status Register. If the microprocessor detects this bit after issuing the Pause command, then it is assured that the chip aborted the Select w/ATN command and no connection exists.

5.5.8 SELECT w/o ATN

The Select w/o ATN is identical to the Select w/ATN command except that the ATN signal is not asserted during the Selection phase.

5.5.9 RESELECT

This command causes the chip to attempt to reselect an Initiator. It may only be used if the microprocessor is in the Disconnected state. Any attempt to issue this command in another state will result in an Invalid Command interrupt. Before issuing this command, the microprocessor must load the Transfer Counter for a timeout on the Initiator's response. This value is computed according to the following formula:

$$\text{Transfer Counter} = \text{Desired Timeout} / (1024 \times \text{Clock Period})$$

If the Transfer Counter is loaded with the value zero, the chip will wait indefinitely for a response from the Initiator being reselected.

The microprocessor must also load the Destination ID Register with the three-bit code of the Initiator to be reselected before issuing the Reselect command.

When the chip detects the Reselect command, it begins by attempting to arbitrate for control of the SCSI bus. If, at any time during arbitration, the chip becomes selected or reselected, the Reselect is aborted and forgotten and the chip will interrupt with one of the following conditions:

1. Selected
2. Selected and Bus Service
3. Reselected

If arbitration is won, the chip places the SCSI bus in the Reselection phase using the Destination ID Register to identify the desired Initiator. At the same time, the chip begins a timer based on the value computed above. If the Initiator does not respond within the timeout period, the chip will disconnect from the bus and interrupt with the Disconnected flag set in the Interrupt Register. (Note: The microprocessor should never monitor the Transfer Counter Zero flag in the Auxiliary Status Register to determine when a timeout has occurred.) If the Initiator responds within the allotted time, the chip will interrupt with a Function Complete status. The chip (acting as the Target) is then in control of the SCSI bus, and waits for the Interrupt Register to be read by the microprocessor. After it has been read, the chip waits for a command from the microprocessor or ATN from the Initiator. If the ATN occurs, the chip will set the Bus Service interrupt. This interrupt may happen immediately after a command has been issued due to internal timing. In this case, the chip waits for the Interrupt Register to be read and the command is ignored. The chip then waits for a new command.

If the timeout is disabled and the Initiator does not respond, or if arbitration is not won, the only way to abort the Reselect command is to issue the Pause command. After the Pause command is issued, it is still possible that the Function Complete or Disconnected interrupts may occur. This happens if one of the interrupts get set before the chip detects the Pause command, or if the Initiator responds while the chip is sequencing off the SCSI bus in a timeout condition. If the chip does not set either interrupt, it will set the Paused bit in the Auxiliary Status Register. If the microprocessor detects this bit after issuing the Pause command, then it is assured that the chip aborted the Reselect command and no connection exists.

5.5.10 DIAGNOSTIC (DATA TURNAROUND)

This Interrupting command causes the chip to attempt to turn a data byte around through its internal data paths. When the command is loaded into the Command Register the Data Register Full bit is reset. The microprocessor then writes one byte into the Data Register. The chip moves the byte to another register and compares the contents of the Data Register. The byte is then moved to a third register (the SCSI output register) and good parity is generated if bit 6 of the command is off (0); bad parity is generated if bit 6 is on (1). Finally, the chip moves the byte back to the Data Register and compares it with the contents of the second register. Based on these comparisons and parity checking, the chip stores a result into the Diagnostic Status Register and sets the Function Complete interrupt. After reading the Interrupt Register, the microprocessor should make sure the Data Register Full bit is on (1) and read the contents of the Data Register. If the Data Register Full bit is not on (0), then an error has occurred. The following is a list of codes which are loaded into bits 6-3 of the Diagnostic Status Register as a result of this command.

BIT 6543	RESULT
0001	Data Miscompare (INITIAL)
0010	Data Miscompare (FINAL)
0011	Good Parity Detected
0100	Bad Parity Detected

5.5.11 RECEIVE COMMANDS

The Receive commands are Interrupting commands that are valid only when connected as a Target device. They are used by the Target to receive commands, data, and message information from an Initiator.

The Receive commands transfer data; therefore, the Single Byte Transfer and DMA mode bits in the Command Register are valid for these commands. If the Single Byte Transfer bit is off (0), the Transfer Counter must be loaded before a Receive command is issued to the chip. In this case, the chip uses the Transfer Counter to determine the number of bytes to receive.

When a Receive command is issued, the chip immediately resets the Data Register Full bit in the Auxiliary Status Register. The chip then drives the I/O, C/D, and MSG outputs for the proper information phase as follows.

COMMAND NAME	I/O	C/D	MSG
Receive Command	0	1	0
Receive Data	0	0	0
Receive Message Out	0	1	1
Receive Unspecified Info Out	0	0	1

The chip then proceeds to request and receive the specified number of information bytes. The DMA mode bit in the Command Register determines how the chip transfers these bytes from its Data Register to the microprocessor.

When a Receive command is terminated, the chip generates an interrupt. The following two events can cause termination:

1. The operation completes successfully; the Transfer Counter is zero. This event results in a Function Complete interrupt with the Parity Error bit in the Auxiliary Status Register off (0). If the initiator activated ATN during the operation, the Bus Service bit will also be on.
2. A Parity Error occurs. The last byte transferred is the byte that caused the error. This event causes a Function Complete interrupt with the Parity Error bit in the Auxiliary Status Register on (1). If the Initiator activated ATN during the operation, the Bus Service bit will also be on.

After any of the interrupts, the chip is always left in the connected Target state. The Transfer Counter indicates the number of bytes remaining to be transferred (zero if completed successfully, and the Data Register is empty (the last byte received is sent to the microprocessor). Also, ACK and REQ are inactive on the bus.

(Note: if a Bus Service interrupt alone occurs after issuing a Receive command, the Initiator activated ATN before the chip began executing the command. In this case, the command is ignored by the chip.)

A Receive command may be stopped prior to an interrupt causing event by issuing a Pause command. Operation of the Pause command is explained in an earlier section. In the event the Initiator does not respond, or stops responding, the chip is left in a state where it cannot respond to a Pause command. For this case, a Disconnect command can be used to abort the command and the connection. The Disconnect command is explained in an earlier section.

5.5.12 SEND COMMANDS

The Send commands are Interrupting commands that are valid only when connected to the bus in the Target role. They are used by a Target to send status, data, and message information to an Initiator.

The Send commands transfer data, and therefore, the Single Byte Transfer and DMA mode bits in the Command Register are valid for these commands. If the Single Byte Transfer bit is off(0), the Transfer Counter must be loaded before a Send command is issued to the chip. In this case, the chip uses the Transfer Counter to determine the number of bytes to send.

When a Send command is issued, the chip immediately resets the Data Register Full bit in the Auxiliary Status Register. Therefore, the first byte of data for the transfer cannot be put into the Data Register until after a Send command is loaded into the Command Register.

In executing a Send command, the chip drives the I/O, C/D, and MSG outputs for the proper information phase. These lines are logically driven for each Send command as shown below.

COMMAND NAME	I/O	C/D	MSG
Send Status	1	1	0
Send Data	1	0	0
Send Message In	1	1	1
Send Unspecified Info In	1	0	1

After resetting Data Register Full and driving I/O, C/D, and MSG, the chip then proceeds to monitor Data Register Full, take the data from the Data Register, and send it to the Initiator. The DMA mode bit in the Command Register specifies how the data is loaded into the chip.

After interrupting, the chip is left in the connected Target state, and ACK and REQ are inactive on the bus. When the transfer is complete, the chip interrupts with a Function Complete Interrupt. If the Initiator activated ATN during the transfer, a Bus Service bit will also be set by the chip.

(NOTE: if a Bus Service interrupt alone occurs after issuing a Send command, the Initiator activated ATN before the chip began executing the command. In this case, the command is ignored by the chip.)

A Send command may be stopped prior to an interrupt causing event by issuing a Pause command. Operation of the Pause command is explained in an earlier section. In the event the Initiator does not, or stops responding, the chip is left in a state where it cannot respond to a Pause command. For this case, a Disconnect command can be used to abort the command and the connection. The Disconnect command is explained in a earlier section.

5.5.13 TRANSFER INFO

The Transfer Info command is an Interrupting command that is valid only when connected to the bus in the Initiator role. It is used by the Initiator for all information transfers across the SCSI bus.

A Transfer Info command is issued by an Initiator in response to a Bus Service interrupt. The Bus Service interrupt, as explained in a previous section, is received by the connected Initiator upon the following conditions: receiving the first REQ from a Target, a previous command has completed and the Target changes phases, the Target changes phases before termination, or when a previous command has completed and the Target is requesting more information. It is not valid to issue a Transfer Info command without having a Bus Service interrupt, because the Target requests and controls all transfers. The chip will only permit one Transfer Info or Transfer Pad per Bus Service interrupt.

After an Initiator receives a Bus Service interrupt, and prior to issuing a Transfer Info command, the I/O, C/D, and MSG bits from the Auxiliary Status Register (read prior to reading the Interrupt) should be examined to determine the type of information phase and the direction of transfer requested by the Target. The Initiator then prepares for the transfer. If the Single Byte Transfer bit is not going to be set in the Command Register, the Transfer Counter must be loaded prior to issuing the Transfer Info command. This is done in order to specify to the chip the maximum number of bytes to be transferred.

When a Transfer Info is issued, the chip immediately resets the Data Register Full bit in the Auxiliary Status Register. For this reason, the first byte of data for an output operation cannot be loaded into the Data Register until after the command is loaded into the Command Register. The chip then proceeds with the transfer, expecting data to be read from (input), or written to (output), its Data Register as indicated by the DMA Mode bit in the Command Register. The chip automatically detects the direction of the transfer from the I/O bit which is stored in the Auxiliary Status Register.

The chip continues a transfer until an interrupt causing event occurs. The following four events will cause the chip to terminate and interrupt.

1. The maximum number of bytes specified have been transferred and the Target activated REQ or the Information Phase changed. This event results in a Bus Service Interrupt. Either single byte transfer was specified or the Transfer Counter is zero as indicated by a bit in the Auxiliary Status Register. The Target may or may not have changed the information phase type. The I/O, C/D, and MSG bits in the Auxiliary Status Register need to be examined at the time of the interrupt to determine what phase the Target is requesting.

   (NOTE: Due to early notification of the phase change, a phase may be selected spuriously and not transfer any data. The microprocessor should not consider this an error condition. The early notification can be blocked with "valid phase bit".)

2. The Target changes the information phase type before the maximum number of bytes are transferred. This event also causes a Bus Service interrupt. The new information phase may be determined by examining the I/O, C/D, and MSG bits in the Auxiliary Status Register. The Transfer Counter may be read at the time of the interrupt to determine the number of bytes remaining to be transferred. When this interrupt occurs for an output transfer, the chip may take one more byte from the microprocessor than it transfers, because of pre-fetching. However, the Transfer Counter still reflects the number of bytes remaining to be transferred.

3. The Target releases the bus by dropping BSY. This event results in a Disconnected interrupt. Following this interrupt, the chip is no longer in the Initiator role. It now remains in the Disconnected state.

4. The last byte of a Message Input phase has been received. This event results in a Function Complete interrupt. For this case, ACK is left active on the bus to allow the microprocessor to Set ATN for the purpose of rejecting the message. After this interrupt is received and a Set ATN is issued (if desired), a Message Accepted must be issued to turn off ACK for the last byte of the Message In phase.

For input transfers (I/O = 1), the chip checks parity for each byte received if the Parity Enable bit in the Control Register is on. When checking parity and the parity error occurs, the chip activates ATN prior to deactivating ACK for the byte that causes the error. It also turns on the Parity Error bit in the Auxiliary Status Register. The parity error, however, does not result in an interrupt. The chip waits for one of the four events listed above before interrupting. Therefore, the Parity Error bit should be examined when servicing any interrupt after issuing Transfer Info command for an input transfer.

If ATN is asserted by the chip, either because of a parity error or because a SET ATN command is issued, the ATN will remain asserted until the end of the connection, or until a Message Out is transferred. Therefore, during each cycle of a Transfer Info operation for output, the chip checks for a message phase (C/D = 1, MSG = 1) and also either a single byte transfer or the Transfer Counter set at zero. If these conditions exist, the chip turns off ATN prior to activating ACK for the last byte of the message.

As previously stated, a Transfer Info normally terminates with an interrupt. If a Transfer Info command must be aborted, possibly because of a timeout violation, either a Chip Reset or a Disconnect command can be used. It is noted, however, that although these commands will force the chip into a disconnected state, the Target device is left on the bus. A SCSI bus reset, which is not a chip function, is the only way an Initiator can force a Target to disconnect.

5.5.14 TRANSFER PAD

The Transfer Pad command is an Interrupting command that is valid only when connected to the bus as an Initiator. It is similiar to the Transfer Info command except that the data transfer between the chip and the microprocessor bus will be different.

Transfer Pad can be used by an Initiator to continue handshaking with a Target without giving data to, or taking data from, the chip. This may be useful if the Target requests an invalid Information Transfer Phase. The chip operates in the same manner as it does for a Transfer Info command, except that for output transfers it takes only one byte of data from the microprocessor and sends the same byte repeatedly until the transfer terminates. For input transfers, it accepts data from the SCSI bus but does not check parity or send it to the microprocessor. Though data is not exchanged with the microprocessor bus, the Transfer Counter is still used by the chip so that a maximum number of pad bytes can be specified.

Protocol for using a Transfer Pad command is the same as the Transfer Info except that the DMA Mode bit has significance only for output transfers. The Transfer Pad terminates because of the same four events that cause a Transfer Info command to terminate. Also, similar to the Transfer Info command, Chip Reset and Disconnect can be used to abort the command.

SECTION 6
BUS INITIATED FUNCTIONS

6.1 SELECTION

If the Select Enable bit in the Control Register is on, the chip may be selected by another SCSI device to be a TARGET for an I/O operation. Selection occurs in the chip only if all the following conditions exist: SELOUT = 0, BSYIN = 0, SELIN = 1, I/O = 0, the chip's ID bit is asserted by the selecting device on the data bus, no more than one other ID bit (the Initiator's) is asserted on the data bus and data bus parity is good.

When all of these conditions exist, the chip is selected. It then encodes the Initiator's ID and loads it into bits 2-0 of the Source ID Register. The chip also detects whether or not the Initiator asserted its ID during selection, and either sets or resets the ID Valid bit in the Source ID Register.

The chip then asserts BSYOUT, waits for SELIN to turn off, and proceeds to take one of the following actions as a result of being selected:

1. If ATN is not asserted by the Initiator during selection, the chip generates a Selected interrupt indicating that the chip is connected as a Target.
2. If ATN is asserted, the chip simultaneously generates Selected, and Bus Service interrupts, indicating that the chip is connected as a Target and ATN is asserted.

6.2 RESELECTION

If the Reselect Enable bit in the Control Register is on, the chip may be reselected by a SCSI Target device. Reselection occurs only if SELOUT = 0, SELIN = 1, BSYIN = 0, I/O = 1, the chip's ID bit and the Target's ID bit are asserted on the data bus, no other ID bits are asserted, and data bus parity is good.

When all of these conditions exist, the chip is reselected. It then encodes the Target's ID and loads it into the Source ID Register. The chip also sets the ID Valid bit in the Source ID Register.

The chip then asserts BSYOUT and waits for SELIN to be released by the Target. When the chip detects SELIN = 0, it de-asserts BSYOUT and then generates a Reselected interrupt.

Reselection is now complete and the chip is in the connected Initiator state.

SECTION 7
INITIALIZATION

The SCSI device may be initialized by asserting RST for a period of at least 100ns, or by issuing a Chip Reset command to the device. The NCR 5386 will respond to the RST pulse or the Chip Reset command, by immediately disconnecting from the SCSI bus, initializing all storage elements and executing an internal self-diagnostic program. The self-diagnostic is explained in a previous section (See Section 4.10, Diagnostic Status Register). The following table lists the status of all registers after the initialization procedure.

	7	6	5	4	3	2	1	0
Data Register 1	x	x	x	x	x	x	x	x
Command Register	0	0	0	0	0	0	0	0
Control Register	0	0	0	0	0	0	0	0
Destination ID Register	0	0	0	0	0	0	0	0
Auxiliary Status Register	0	0	x	x	x	0	1	0
ID Register	0	0	0	0	0	x	x	x
Interrupt Register	0	0	0	0	0	0	0	0
Source Register	0	0	0	0	0	1	1	1
Data Register 2	x	x	x	x	x	x	x	x
Diagnostic Status Register	1	x	x	x	x	x	x	x
Transfer Counter (MSB)	0	0	0	0	0	0	0	0
Transfer Counter (2nd)	0	0	0	0	0	0	0	0
Transfer Counter (LSB)	0	0	0	0	0	0	0	0

x = Unknown

Table 7.1
Register Initialization

The controlling processor should loop on reading the Diagnostic Status Register until the Self-Diagnostic Complete bit (bit 7) is on (1). This should take approximately 350 clock cycles after reset occurs. If Programmed ID Mode is used, the processor should then check the remaining bits in this register for all zeros (no errors), and then load the ID Register. Following this, the Control Register should be programmed to enable the proper bits for SCSI operation. The SCSI Protocol Controller is now connected to the SCSI bus in a disconnected state. It is ready to receive commands from the controlling processor or respond to (re) selection attempts.

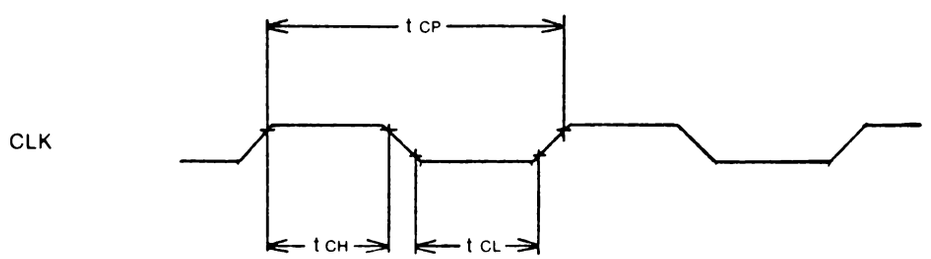
SECTION 8
EXTERNAL CHIP TIMING

Timing requirements must be over the operating temperature (0-70°C) and voltage (4.75 to 5.25V) ranges. Loading for all output signals, except $\overline{SBEN}$, is assumed to be four low-power Schottky inputs, including 50 pF capacitance. Loading for $\overline{SBEN}$ is assumed to be ten low-power Schottky inputs, including 100 pF capacitance.

8.1 MICROPROCESSOR INTERFACE — PRELIMINARY

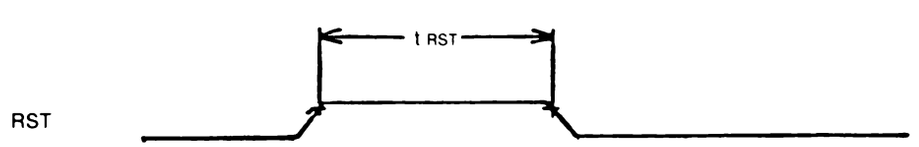
8.1.1 CLK

NAME	DESCRIPTION	MIN	MAX	UNITS
t_{CP}	Clock Period	100	200	ns
t_{CH}	Clock High	.45 t_{CP}	.55 t_{CP}	
t_{CL}	Clock Low	.45 t_{CP}	.55 t_{CP}	



8.1.2 RESET

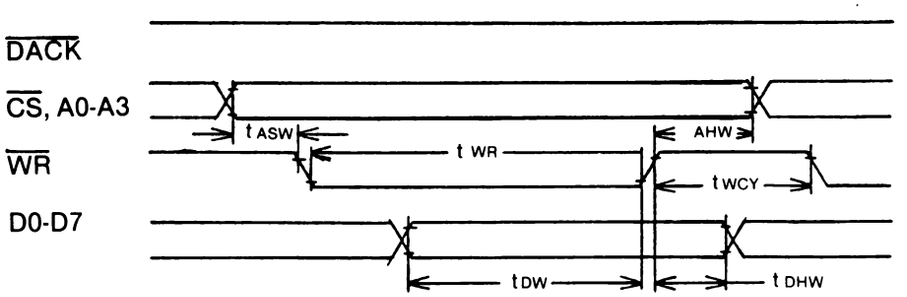
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{RST}	Reset Pulse Width	100			ns



8.1.3 MPU WRITE

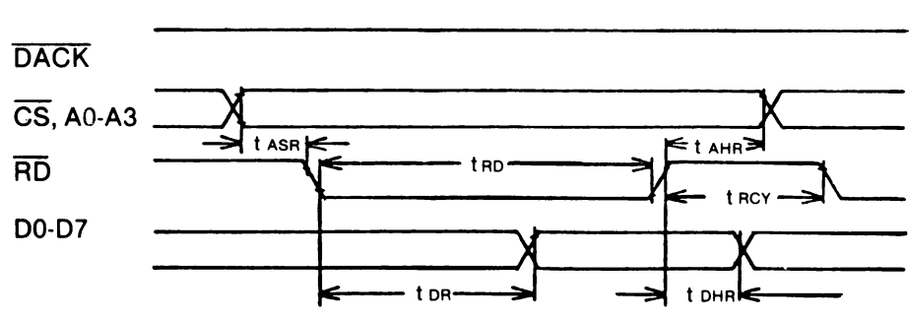
PRELIMINARY

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{ASW}	Address Set-up Time	0			ns
t_{WR}	$\overline{WR}$ Pulse Width	95			ns
t_{DW}	Data-to $\overline{WR}$ High	50			ns
t_{AHW}	Address Hold Time	0			ns
t_{DHW}	Data Hold Time	20			ns
t_{WCY}	$\overline{WR}$ Off to $\overline{WR}$ or $\overline{RD}$ On	125			ns



8.1.4 MPU READ

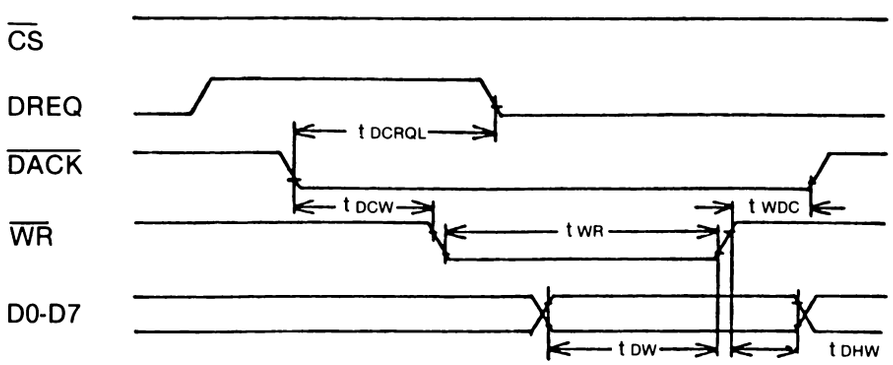
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{ASR}	Address Set-up Time to $\overline{RD}$	0			ns
t_{RD}	$\overline{RD}$ Pulse Width	125			ns
t_{DR}	$\overline{RD}$ to Data			90	ns
t_{AHR}	Address Hold Time	0			ns
t_{DHR}	Data Hold Time	10		65	ns
t_{RCY}	$\overline{RD}$ Off to $\overline{WR}$ or $\overline{RD}$ On	125			ns



8.1.5 DMA WRITE

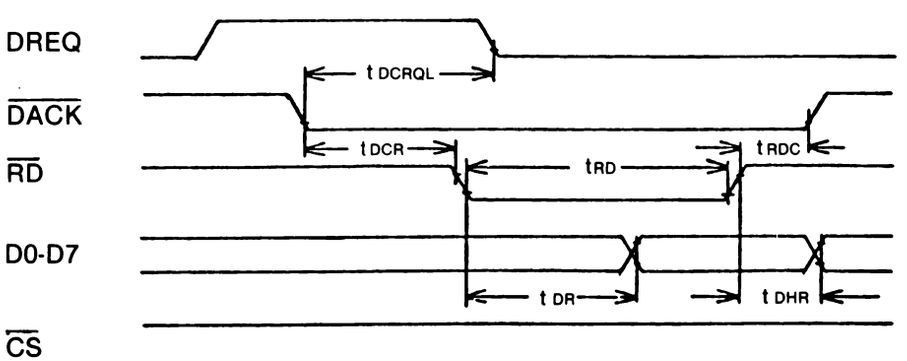
PRELIMINARY

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{DCRQL}	$\overline{DACK}$ to DREQ Low	0		40	ns
t_{DCW}	$\overline{DACK}$ to $\overline{WR}$	0			ns
t_{WR}	$\overline{WR}$ Pulse Width	70			ns
t_{WDC}	$\overline{WR}$ High to $\overline{DACK}$ High	0			ns
t_{DHW}	Data Hold Time	20			ns
t_{DW}	Data to $\overline{WR}$ High	50			ns



8.1.6 DMA READ

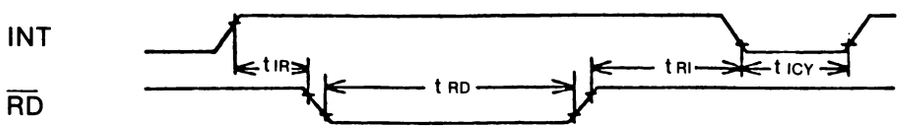
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{DCRQL}	$\overline{DACK}$ to DREQ Low	0		40	ns
t_{DCR}	$\overline{DACK}$ to $\overline{RD}$	0			ns
t_{RD}	$\overline{RD}$ Pulse Width	95			ns
t_{RDC}	$\overline{RD}$ High to $\overline{DACK}$ High	0			ns
t_{DHR}	Data Hold Time	10		65	ns
t_{DR}	$\overline{RD}$ to Data			75	ns



8.1.7 INTERRUPT

PRELIMINARY

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{IR}	INT to $\overline{RD}$	0			ns
t_{RD}	$\overline{RD}$ Pulse Width	95			ns
t_{RI}	$\overline{RD}$ High to INT Low			125	ns
t_{ICY}	INT Off to INT On	125			ns



8.2.1 SELECTION (INITIATOR)

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
^tBF	Bus Free	385			ns
$^tBIA(5)$	BSYIN low to ARB high	1.2		2.6	us
^tSLA	SELOUT high to ARB low & ID bit Disabled	3.2			us
^tBIBO (5)	BSYIN low to BSYOUT high	1.2		2.8	us
^tBCD	Bus Clear Delay			225	ns
^tAD	Arbitration Delay	3.0			us
^tPC	Priority check to SELOUT	0			ns
^tBID (5)	BSYIN low to ID bit high	1.2		2.9	us
^tADV	Arbitration Data Valid to Priority Check	0			ns
^tSI	SELOUT to IGS	2.0			us
^tIDBL	Target ID high to BSYOUT low	1.1			us
^tBOBI	BSYOUT low to BSYIN low	0		400	ns
^tBSL	BSYIN high to SELOUT low	800			ns
^tDID	SBEN active to Bus enabled	150			ns

NOTES:

1. The chip ensures that the bus remains free (BSYIN and SELIN inactive) for ^tBF before attempting arbitration.

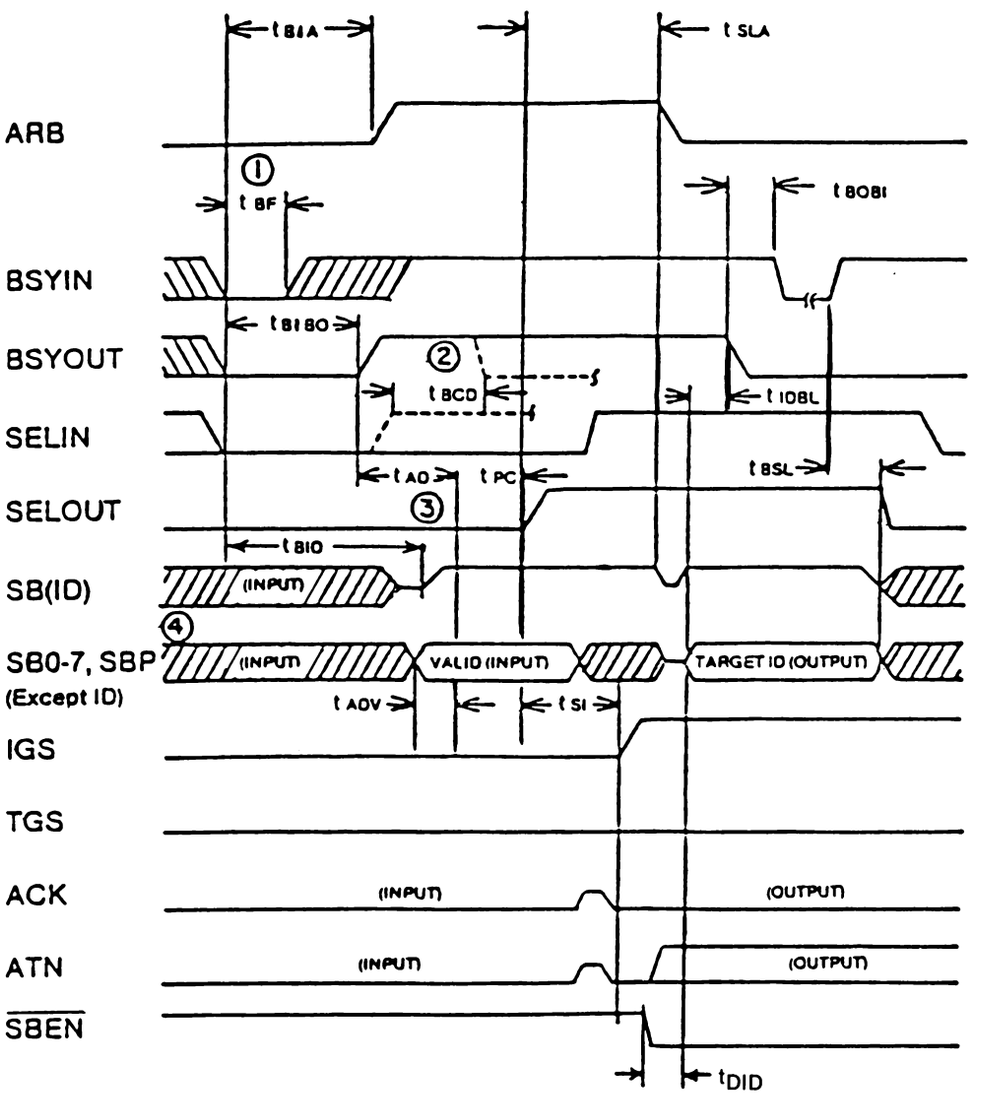
2. If SELIN becomes active at any time during arbitration, the chip must deassert BSYOUT within ^tBCD.

3. The chip waits (^tAD), and then checks to see if arbitration is won (^tPC). The chip then asserts SELOUT if arbitration is won.

4. One of the data bits is assigned as an ID bit by the IDO-ID2 signals. During Bus Free, the chip places all of the data bits, including ID, in a high impedance state. During arbitration the chip enables its ID bit and drives it high, but the remainder of the data bits remain in the high impedance state for reading.

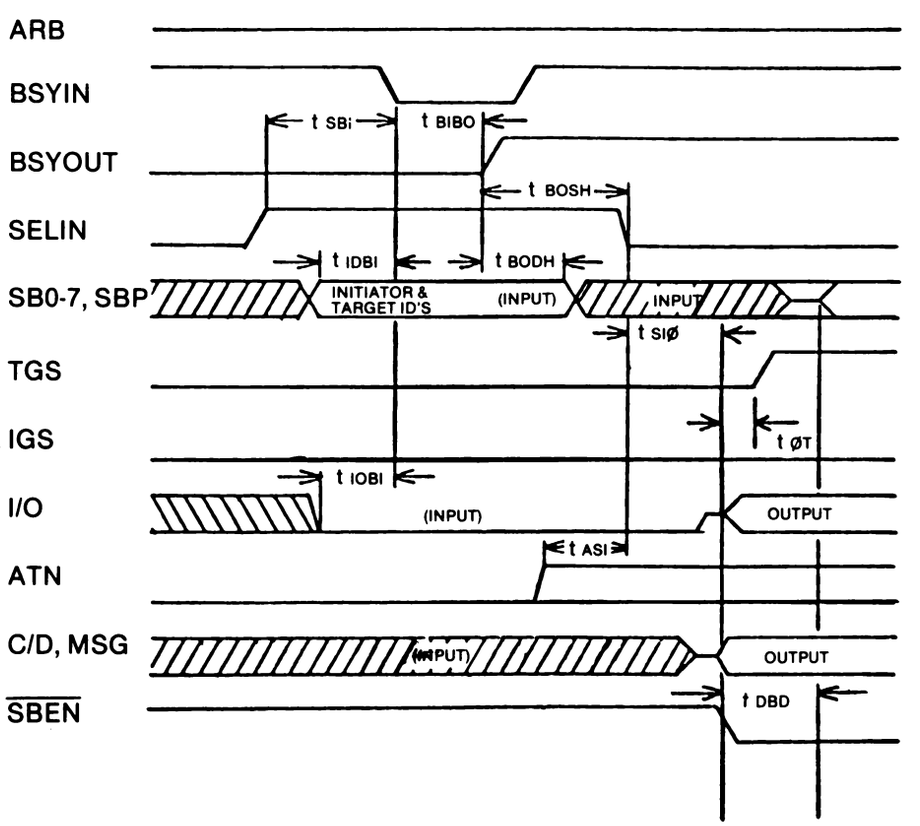
5. To verify these timings in a test environment, the user must allow a minimum of 45 clock cycles after the select command has been issued before the device begins to check for BSYIN low.

8.2.1 SELECTION (INITIATOR)



8.2.2 SELECTION (TARGET) PRELIMINARY

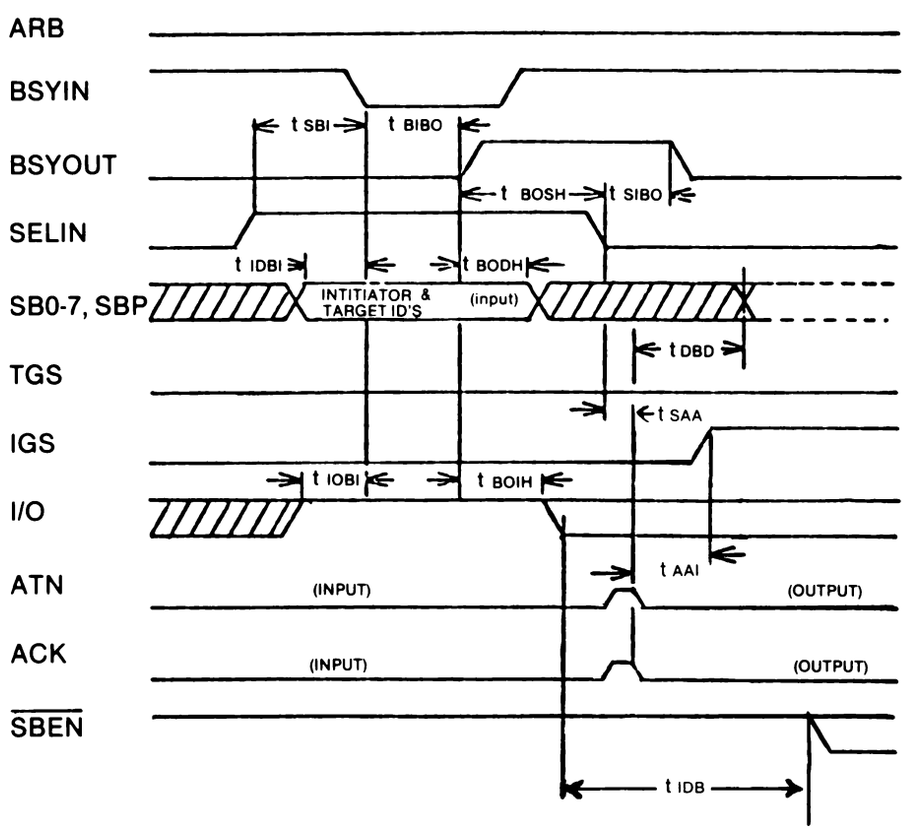
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
T_{SBI}	SELIN high to BSYIN low	50			ns
t_{IDBI}	ID's valid to BSYIN low	0			ns
t_{IOBI}	I/O low to BSYIN low	0			ns
t_{BIBO}	BSYIN low to BSYOUT high	0		2.0	μs
t_{BODH}	BSYOUT high Data Hold	0			ns
t_{BOSH}	BSYOUT high SELIN Hold	0			ns
t_{ASI}	ATN high to SELIN low	0			ns
$t_{SIØ}$	SELIN low to Phase signals Enabled	150			ns
t_{OT}	Phase signals enabled to TGS High	150			ns
t_{DBD}	SBEN low to Data Bus Enabled	150			ns



8.2.3 RESELECTION (INITIATOR)

PRELIMINARY

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{SBI}	SELIN high to BSYIN low	50			ns
t_{IDBI}	ID's valid to BSYIN low	0			ns
t_{IOBI}	I/O high to BSYIN low	0			ns
t_{BIBO}	BSYIN low to BSYOUT high	0		2.0	μs
t_{BODH}	BSYOUT high Data Hold	0			ns
t_{BOSH}	BSYOUT high SELIN Hold	0			ns
t_{BOIH}	BSYOUT high I/O hold	0			ns
t_{SIBO}	SELIN low to BSYOUT low	0			ns
t_{SAA}	SELIN low to ACK & ATN enabled	750			ns
t_{AAI}	ACK & ATN enabled to IGS high	150			ns
t_{IDB}	I/O low to $\overline{SBEN}$ low	0			ns
t_{DBD}	$\overline{SBEN}$ low to Data Bus Enabled	150			ns



8.2.4 RESELECTION (TARGET)

PRELIMINARY

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{BF}	Bus Free	385			ns
t_{BIA} (5)	BSYIN low to ARB high	1.2		2.6	us
t_{SLA}	SELOUT high to ARB low & ID bit Disabled	3.2			us
t_{BIBO} (5)	BSYIN low to BSYOUT high	1.2		2.8	us
t_{BCD}	Bus Clear Delay			225	ns
t_{AD}	Arbitration Delay	3.0			us
t_{PC}	Priority check to SELOUT	0			ns
t_{BID} (5)	BSYIN low to ID bit high	1.2		2.9	us
t_{ADV}	Arbitration Data Valid to Priority Check	0			ns
t_{S0}	SELOUT Phase signals Enabled & SBEN Low	2.4			us
t_{0T}	Phase Signals Enabled to TGS High	150			ns
t_{DID}	$\overline{\text{SBEN}}$ low to Bus Enabled	150			ns
t_{IDBL}	INITIATOR ID high to BSYOUT low	2.7			us
t_{BOBI}	BSYOUT low to BSYIN	0		400	ns
t_{BIBO}	BSYIN high to BSYOUT high	0.7		2.0	us
t_{BSL}	BSYOUT high to SELOUT low	450			ns

NOTES:

1. The chip ensures that the bus remains free (BSYIN and SELIN inactive) for t_{BF} before attempting arbitration.

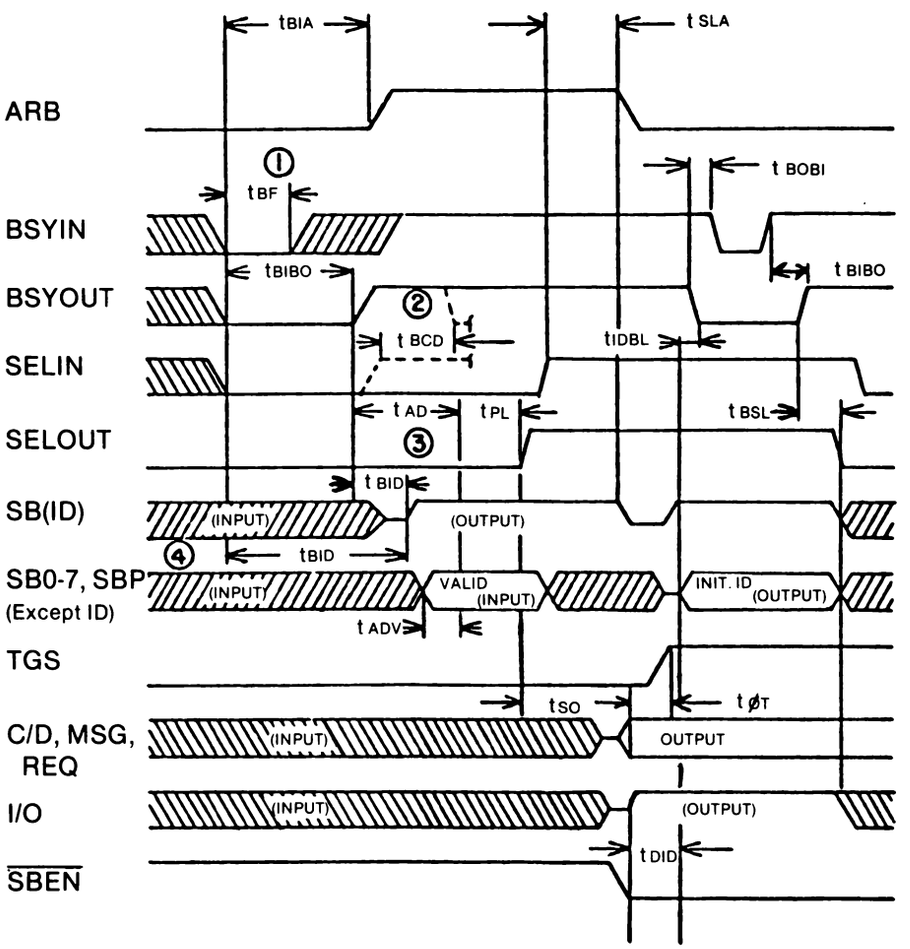
2. If SELIN becomes active at any time during arbitration, the chip must deassert BSYOUT within t_{BCD}.

3. The chip waits (t_{AD}), and then checks to see if arbitration is won (t_{PC}). The chip then asserts SELOUT if arbitration is won.

4. One of the data bits is assigned as an ID bit by the IDO-ID2 signals. During Bus Free, the chip places all of the data bits, including ID, in a high impedance state. During arbitration the chip enables its ID bit and drives it high, but the remainder of the data bits remain in the high impedance state for reading.

5. To verify these timings in a test environment, the user must allow a minimum of 45 clock cycles after the select command has been issued before the device begins to check for BSYIN low.

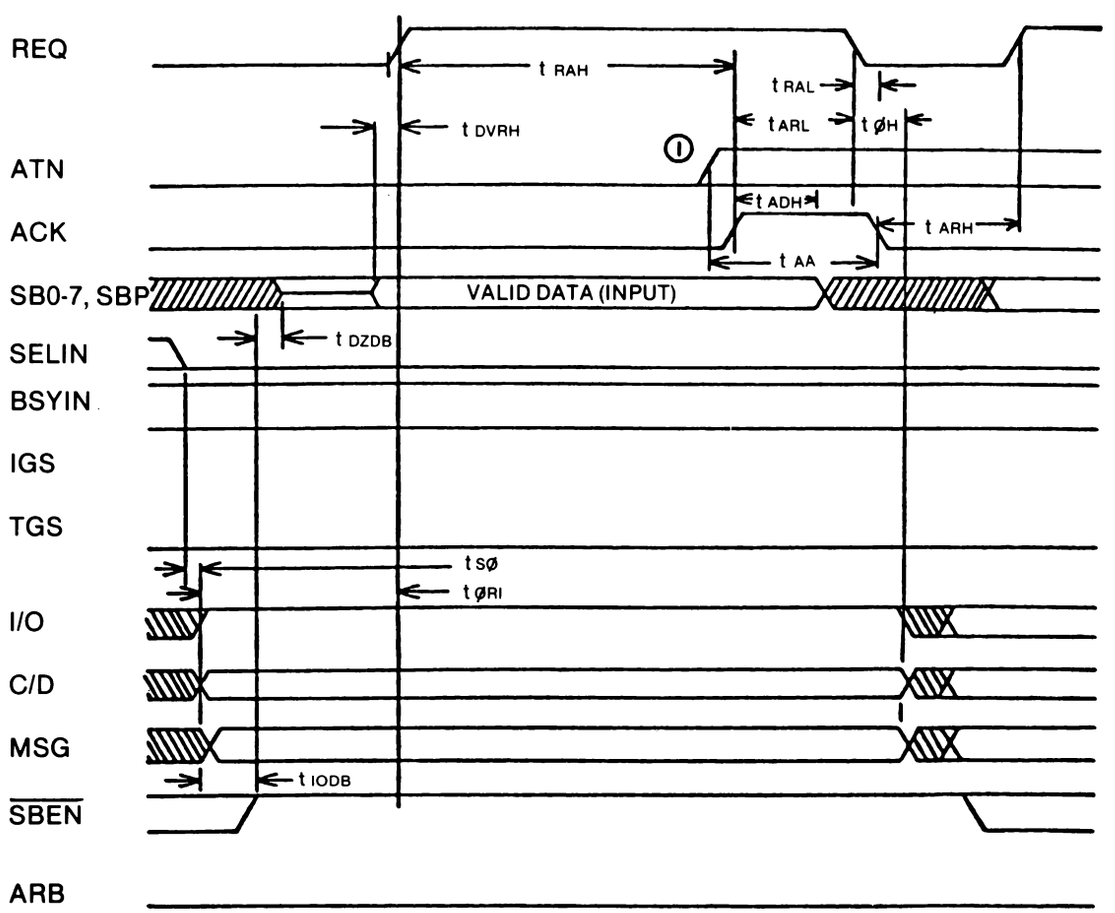
8.2.4 RESELECTION (TARGET)



8.2.5 INFORMATION TRANSFER PHASE INPUT (INITIATOR) PRELIMINARY

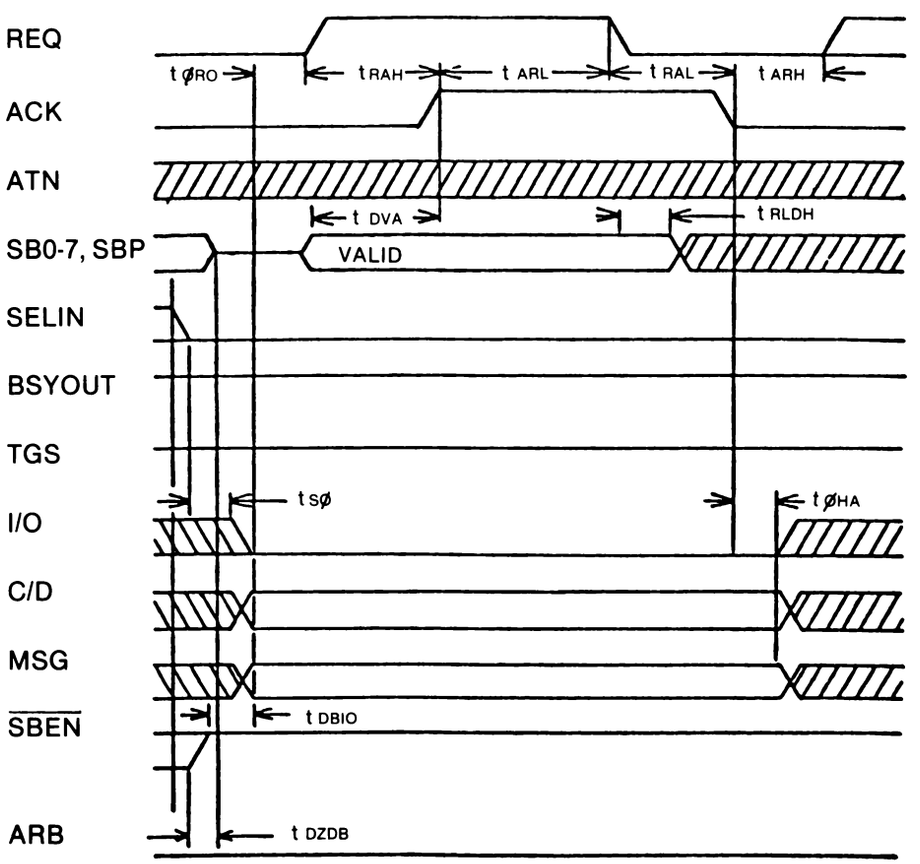
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{DVRH}	Data Valid to REQ high	0			ns
$t_{\phi RI}$	Phase Valid to REQ high	100			ns
t_{RAH}	REQ high to ACK high	0			ns
t_{RAL}	REQ low to ACK low	0			ns
t_{AA}	ATN high to ACK low	100			ns
$t_{S\phi}$	SELIN low to Phase change	0			ns
$t_{\phi H}$	Phase hold from ACK low	20			ns
t_{ADH}	Data hold from ACK high	0			ns
t_{ARL}	ACK high to REQ low	35			ns
t_{IODB}	I/O high to $\overline{SBEN}$ high			50	ns
t_{DZDB}	Data Bus disable from $\overline{SBEN}$ high			10	ns
t_{ARH}	ACK Low to REQ High	35			ns

NOTE 1: If the chip detects a parity error it must assert ATN at least t_{AA} before it de-asserts ACK.



8.2.6 INFORMATION TRANSFER PHASE INPUT (TARGET) PRELIMINARY

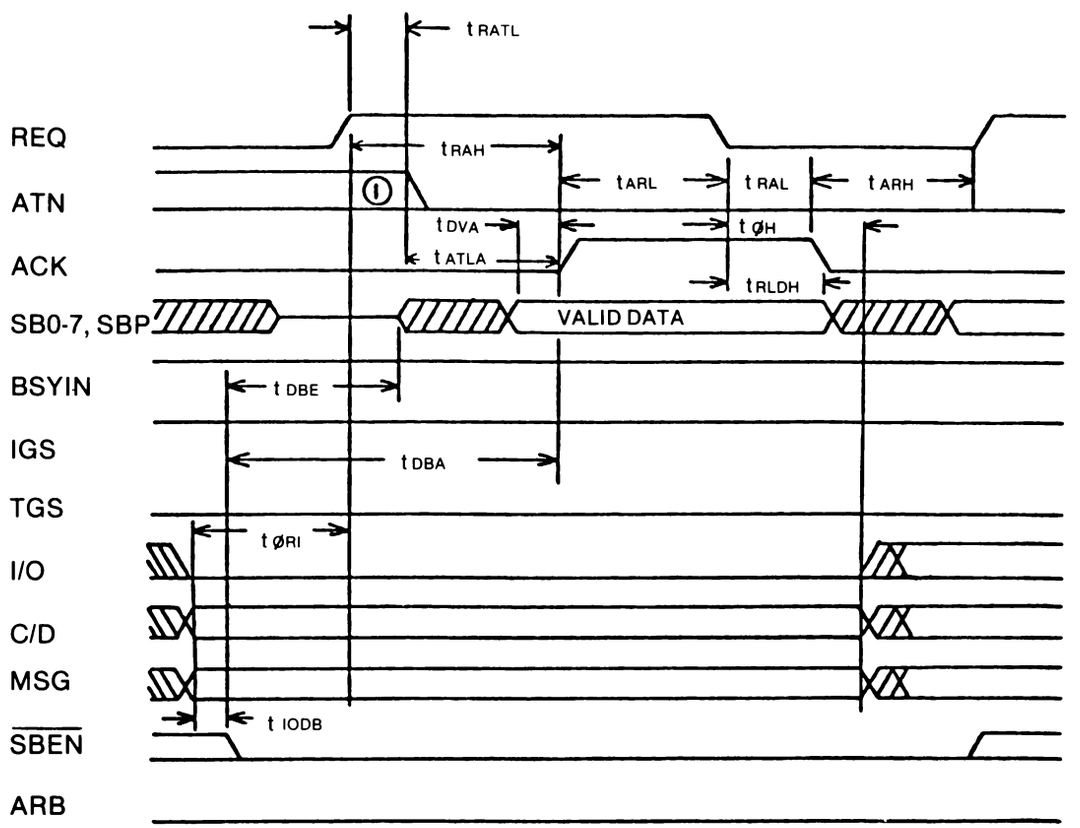
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
$t_{S\emptyset}$	SELIN low to Phase Change	0			ns
$t_{\emptyset RO}$	Phase Change to REQ out	500			ns
t_{RAH}	REQ high to ACK high	35			ns
t_{ARL}	ACK high to REQ low	0			ns
t_{DVA}	Data Valid to ACK high	0			ns
t_{RAL}	REQ low to ACK low	35			ns
t_{ARH}	ACK low to REQ high	0			ns
t_{RLDH}	REQ low Data Hold	0			ns
$t_{\emptyset HA}$	Phase Hold from ACK low	0			ns
t_{DBIO}	SBEN high to I/O low	0			ns
t_{DZDB}	Data Bus disable to $\overline{SBEN}$ high	0			ns



8.2.7 INFORMATION TRANSFER PHASE OUTPUT (INITIATOR) PRELIMINARY

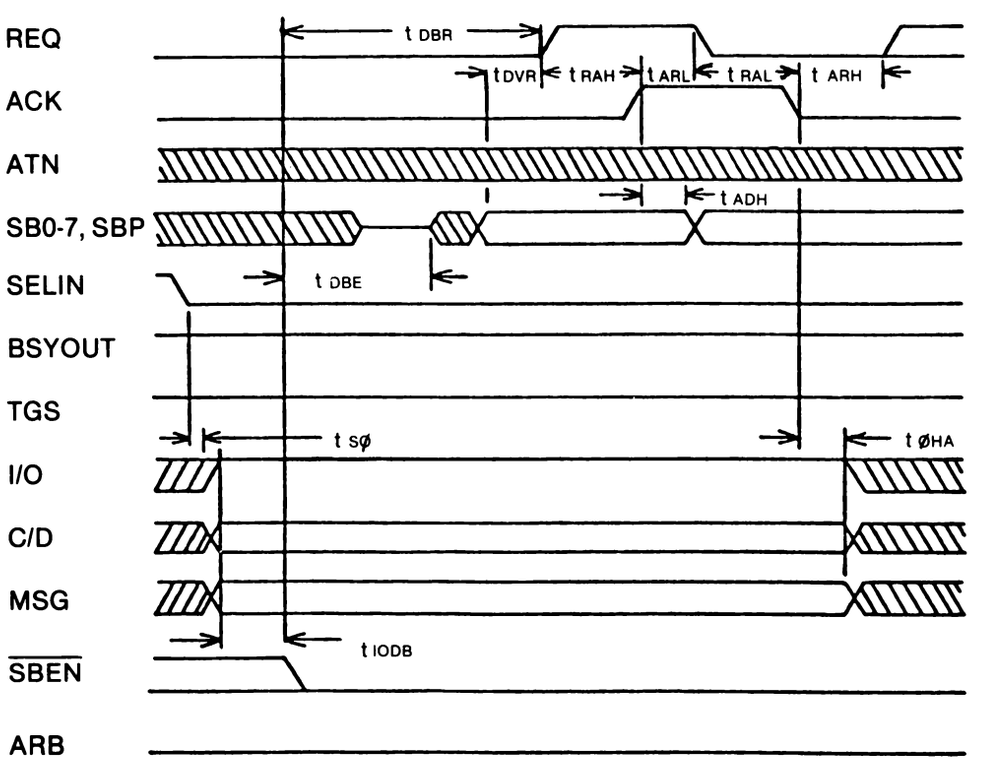
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
$t_{\phi RI}$	Phase Valid to REQ high	100			ns
t_{RAH}	REQ high to ACK high	35			ns
t_{RAL}	REQ low to ACK low	0			ns
t_{DVA}	Data Valid to ACK high	100			ns
t_{RLDH}	REQ low Data hold	0			ns
$t_{\phi H}$	Phase hold from ACK low	20			ns
t_{ARL}	ACK high to REQ low	0			ns
t_{IODB}	I/O low to $\overline{SBEN}$ low	0			ns
t_{DBE}	$\overline{SBEN}$ low to Data Bus Enable	85			ns
t_{DBA}	$\overline{SBEN}$ low to ACK high	185			ns
t_{RATL}	REQ High to ATN low	0			ns
t_{ATLA}	ATN Low to ACK High	25			ns
t_{ARH}	ACK Low to REQ High	35			ns

NOTE 1: ATN is only de-asserted in this manner during the last byte of a Message Out Phase.



8.2.8 INFORMATION TRANSFER PHASE OUTPUT (TARGET) PRELIMINARY

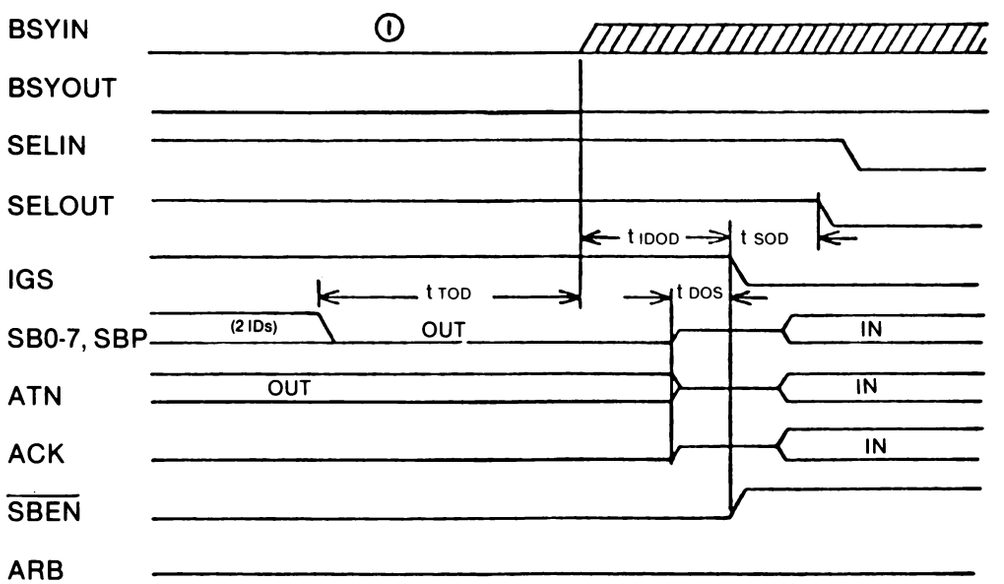
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
$t_{S\emptyset}$	SELIN low to Phase Change	0			ns
t_{IODB}	I/O high to $\overline{SBEN}$ low	500			ns
t_{DBR}	$\overline{SBEN}$ low to REQ out	185			ns
t_{DVA}	Data Valid to REQ high	100			ns
t_{RAH}	REQ high to ACK high	0			ns
t_{ARL}	ACK high to REQ low	0			ns
t_{RAL}	REQ low to ACK low	0			ns
t_{ARH}	ACK low to REQ high	0			ns
$t_{\emptyset HA}$	Phase hold from ACK low	0			ns
t_{ADH}	Data hold from ACK low	0			ns
t_{DBE}	$\overline{SBEN}$ low to Data Bus Enabled	85			ns



8.2.9 BUS RELEASE FROM SELECTION (INITIATOR) PRELIMINARY

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{TOD}	Bus Release Timeout Delay	100			μs
t_{IDOD}	IGS & $\overline{SBEN}$ Turn-off Delay	0			ns
t_{SOD}	SELOUT Turn-off Delay	0			ns
t_{DOS}	Driver Turn-off set-up to IGS & $\overline{SBEN}$ off	0			ns

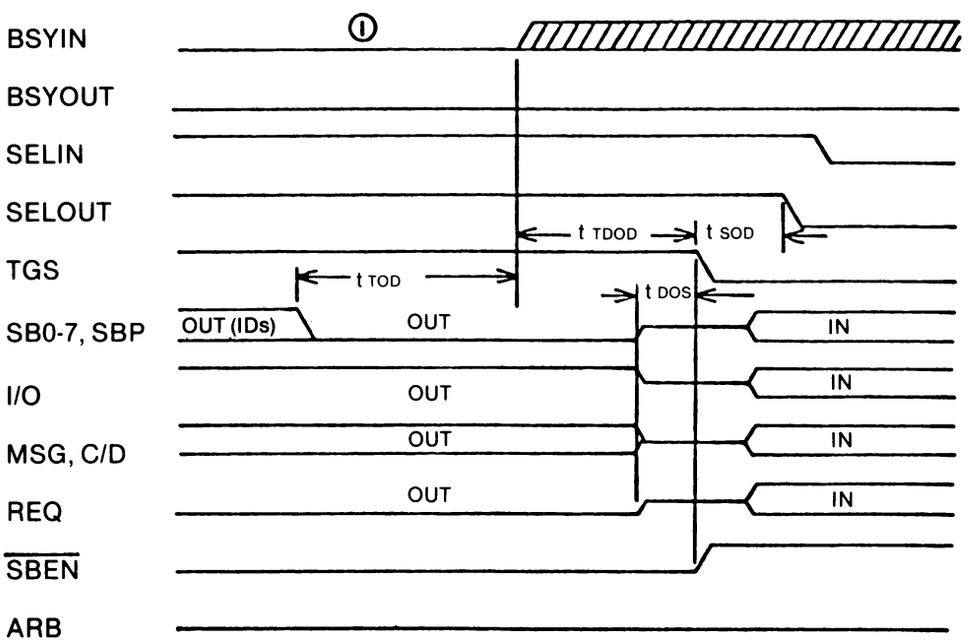
NOTE 1: If the chip detects BSYIN active by the end of the timeout delay, the bus release sequence shall be aborted since selection has been successful.



8.2.10 BUS RELEASE FROM RESELECTION (TARGET) PRELIMINARY

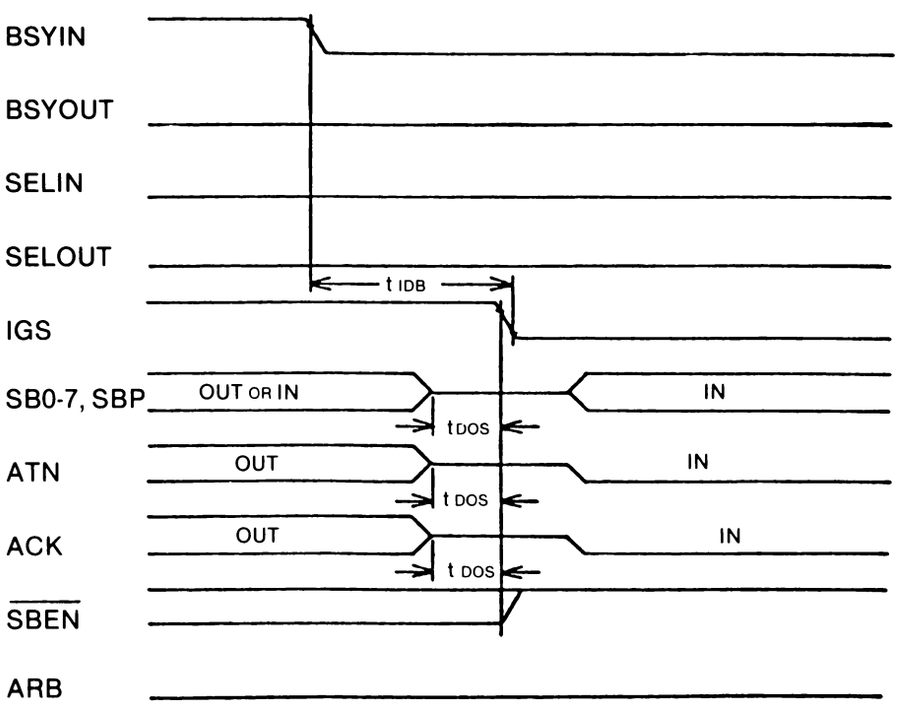
NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{TOD}	Bus Release Timeout Delay	100			*u*s
t_{DOD}	TGS & $\overline{SBEN}$ Turn-off Delay	0			ns
t_{SOD}	SELOUT Turn-off Delay	0			ns
t_{DOS}	Driver Turn-off set-up to TGS & $\overline{SBEN}$ off	0			ns

NOTE 1: If the chip detects BSYIN active by the end of the timeout delay, the bus release sequence shall be aborted since selection has been successful.



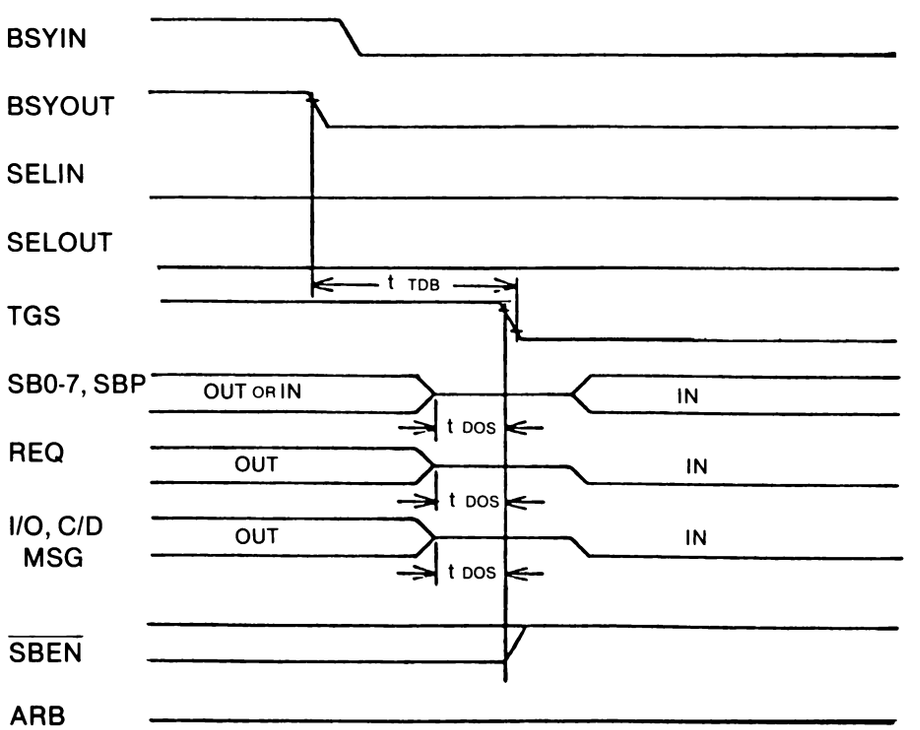
8.2.11 BUS RELEASE FROM INFORMATION PHASE (INITIATOR) PRELIMINARY

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{IDB}	IGS & $\overline{SBEN}$ Turn-off Delay from BSYIN off			225	ns
t_{DOS}	Driver Turn-off set-up to IGS off	0			ns



8.2.12 BUS RELEASE FROM INFORMATION PHASE (TARGET) PRELIMINARY

NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
t_{TDB}	TGS & $\overline{SBEN}$ Turn-off from BSYOUT off			225	ns
t_{DOS}	Driver Turn-off set-up to TGS off	0			ns



NCR 8310 GENERAL PURPOSE DRIVER

RECEIVER CHIP

PRELIMINARY DATA SHEET

NCR

MICROELECTRONICS DIVISION, COLORADO SPRINGS

08/07/85

PRELIMINARY

This document contains the latest information available at the time of publication. However, NCR reserves the right to modify the contents of this material at any time. Also, all features, functions and operations described herein may not be marketed by NCR in all parts of the world. Therefore, before using this document, consult your NCR representative or NCR office for the information that is applicable and current.

TABLE OF CONTENTS

SECTION 1
GENERAL DESCRIPTION

1.0 The NCR 8310 General Purpose Driver/Receiver Chip, a 48-pin NMOS device, is designed as a 48 mA bus transceiver chip for all of the Small Computer System Interface (SCSI) bus signals. It contains high-current single-ended drivers for the SCSI bus. The NCR 8310 is specifically intended to be used with the NCR 5385E and NCR 5386 SCSI Protocol Controller chips and interfaces directly to the above referenced chips. Additionally, it may be used with other interfaces where a general purpose 48 mA driver/receiver chip is required.

Figure 1.1 shows the pinout for the NCR 8310, with signals labeled in parenthesis referenced in Figure 1.2 interfacing directly to the NCR 5385E. Signals not enclosed by parenthesis are referenced to Figure 1.3 and refer to the most general application of the NCR 8310. The prefix "H" indicates "high current." The 48 mA data bus, HD0-HD7/ and microprocessor bus, D0-D7, are controlled by the Data Bus Enable (DBEN/), and port HB0-HB3 and B0-B3 are controlled by PBEN. Similarly, ports A, E, F and G are controlled likewise.

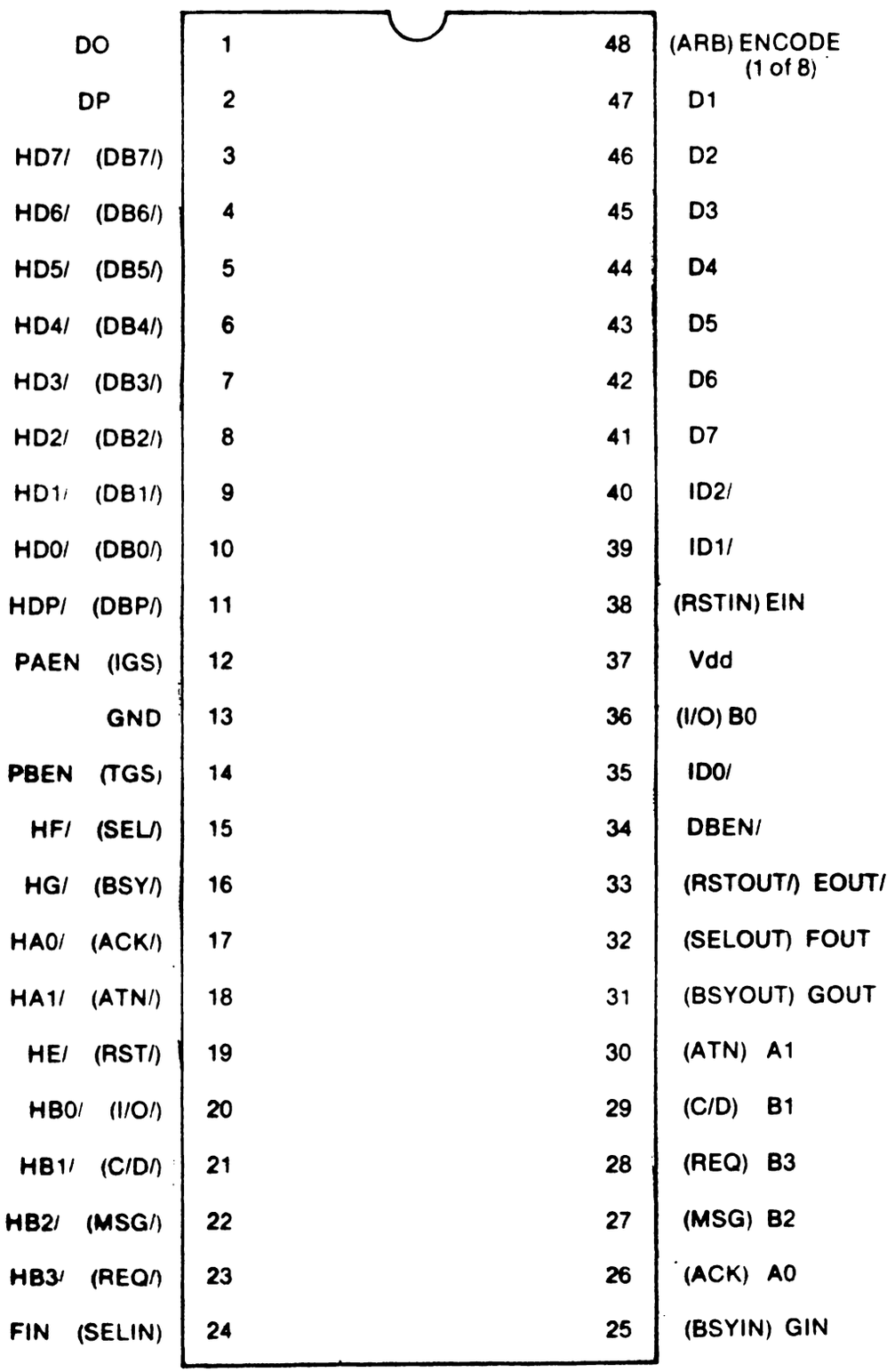


FIGURE 1.1 PINOUT

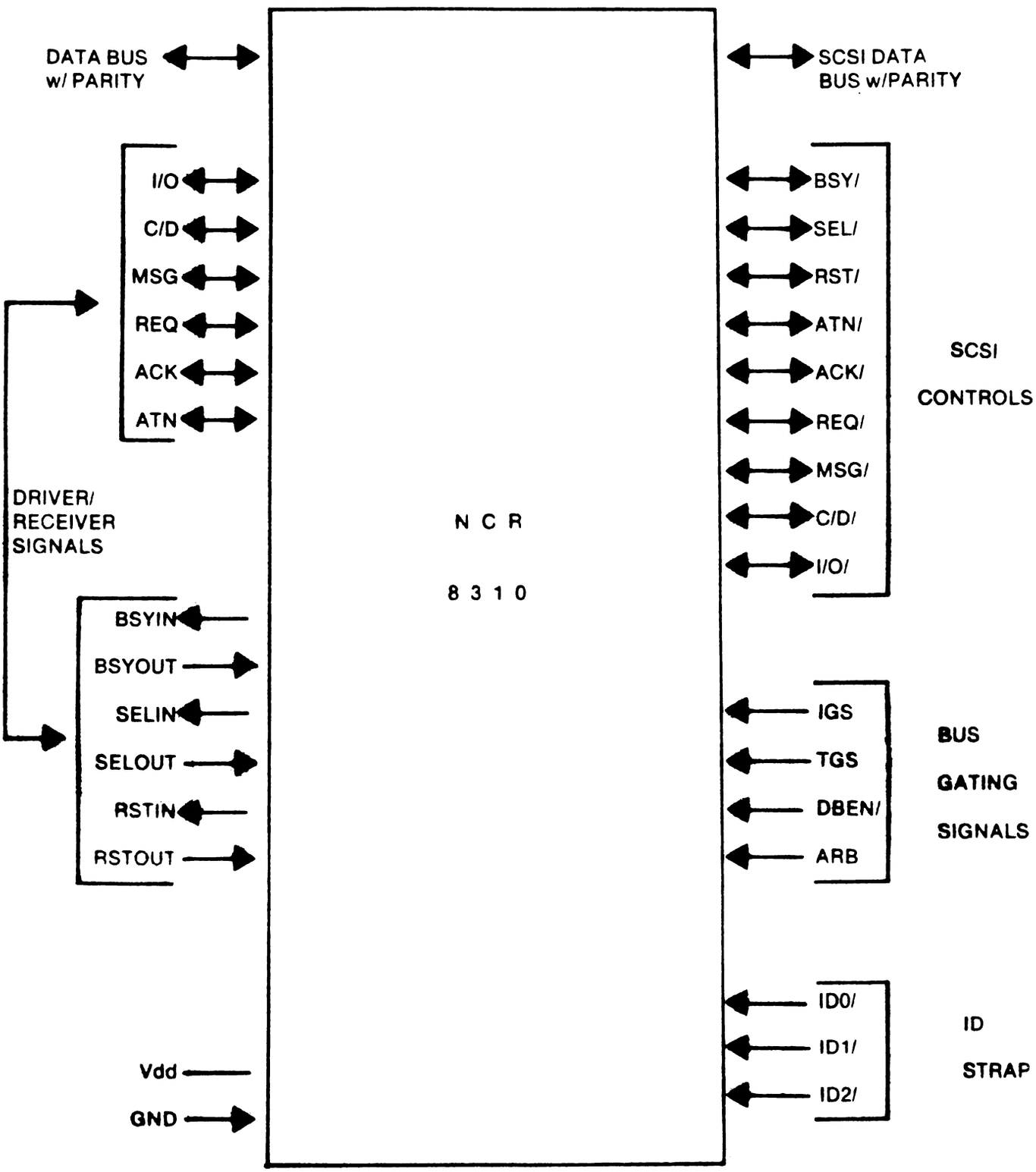


FIGURE 1.2
FUNCTIONAL PIN GROUPING

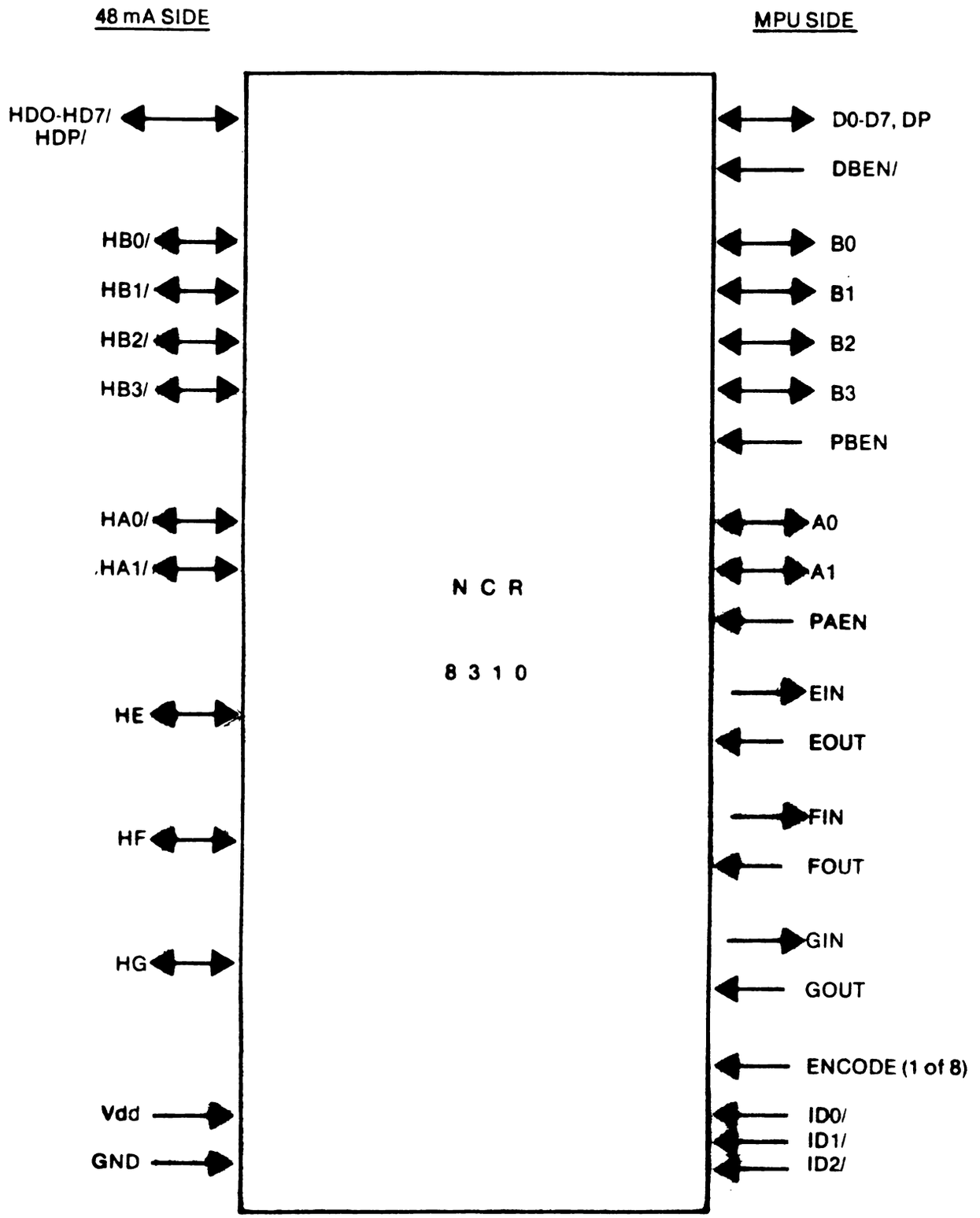


FIGURE 1.3
GENERAL PURPOSE FUNCTIONAL PIN GROUPING

Figure 1.4 shows the interface between the NCR 5385E SCSI Protocol Controller and the NCR 8310 Driver/Receiver Chip. With the implementation of this suggested interface, a two-chip solution is possible.

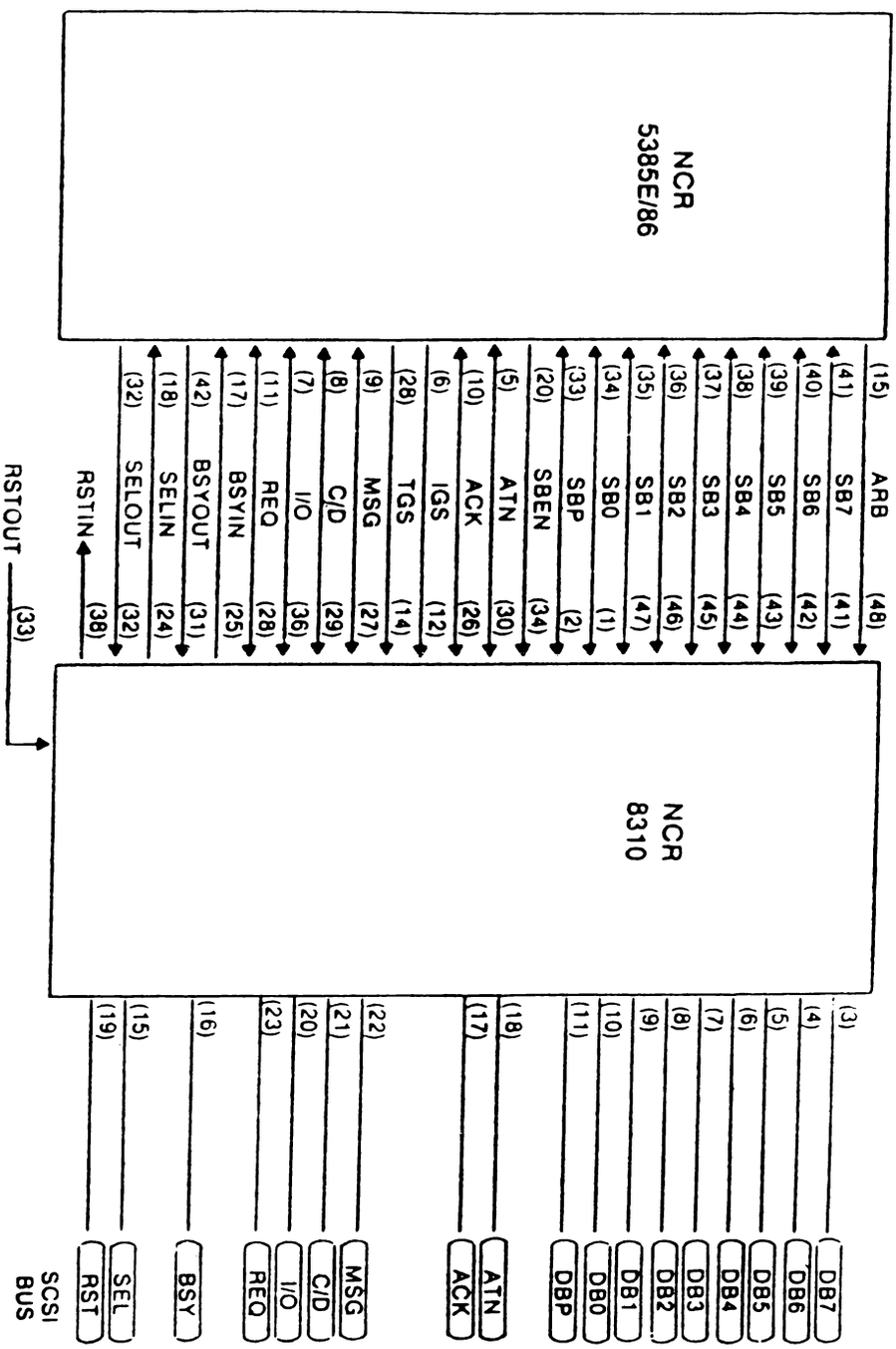


FIGURE 1.4
SINGLE-ENDED INTERFACE USING THE NCR 5385E/5386 SCSI PROTOCOL CONTROLLER AND THE NCR 8310 GENERAL PURPOSE DRIVER/RECEIVER CHIP

Figure 1.5 shows the Equivalent Circuit for the NCR 8310.

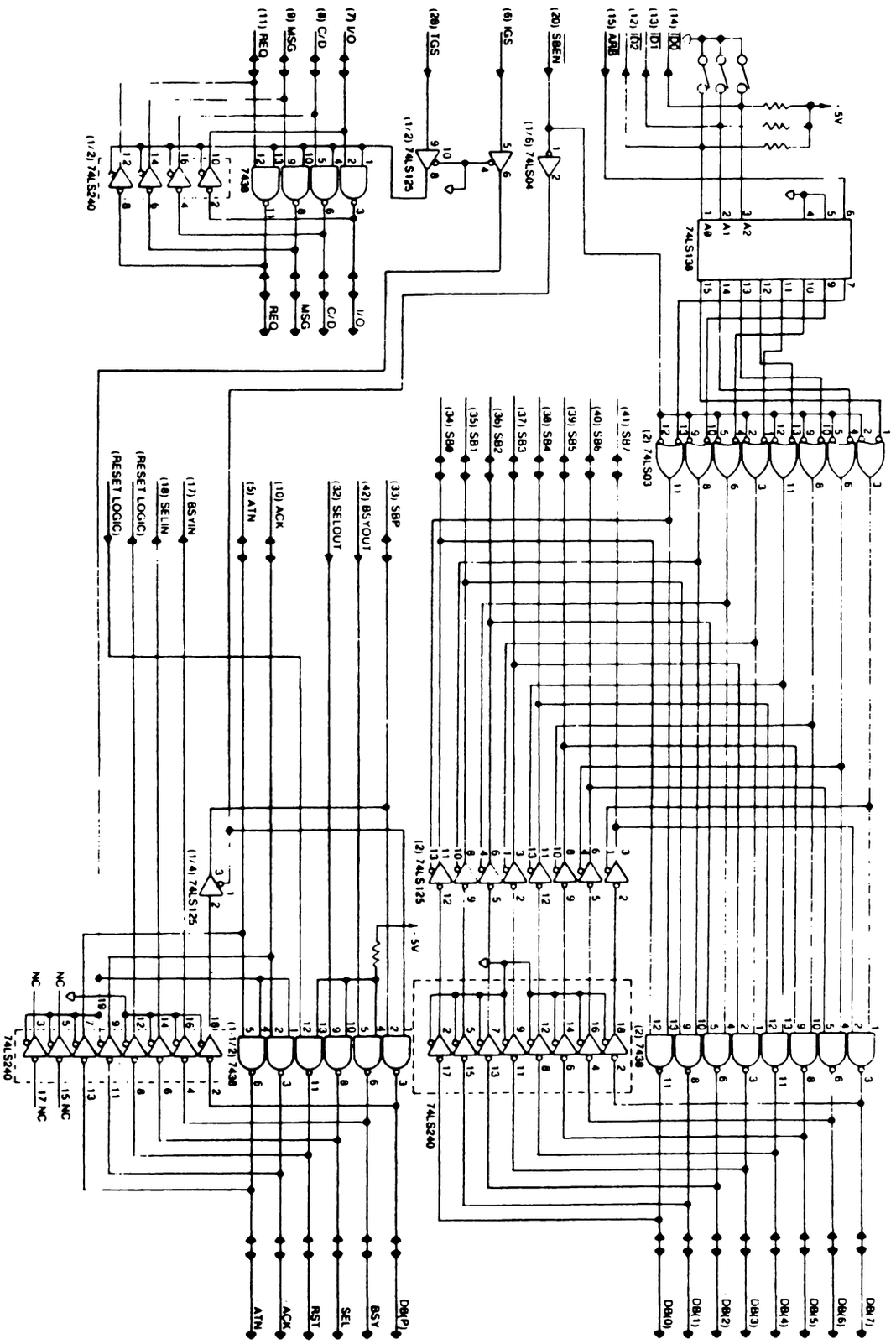


FIGURE 1.5
NCR 8310 GENERAL PURPOSE DRIVER/RECEIVER EQUIVALENT CIRCUIT

SECTION 2
PIN DESCRIPTION

2.1 SCSI INTERFACE SIGNALS

The following signals are all bi-directional, active low, open collector signals. With 48 mA sink capability, all pins interface directly with the SCSI bus.

SIGNAL	PIN	DESCRIPTION
HD0...HD7/ (DB0...DB7/)	10...3	These eight data bits (DB0-DB7/) plus a parity bit (DBP/) form the data bus. DB7/ is the most significant bit and has the highest priority during the Arbitration phase. Data parity is odd. Parity is not valid during Arbitration.
HDP/ (DBP/)	11	
HB0/ (I/O/)	20	I/O/ is a signal driven by a target which controls the direction of data movement on the SCSI bus. True indicates input to the initiator. This signal is also used to distinguish between Selection and Reselection phases.
HB1/ (C/D/)	21	A signal driven by the target, C/D/ indicates Control of Data information is on the data bus. This signal is received by the initiator.
HB2/ (MSG/)	22	MSG/ is a signal driven by the target during the Message phase. This signal is received by the initiator.
HB3/ (REQ/)	23	Driven by a target, REQ/ indicates a request for a REQ/ ACK data transfer handshake. This signal is received by the initiator.
HA0/ (ACK/)	17	Driven by an initiator, ACK/ indicates an acknowledgement for a REQ/ACK data transfer handshake. In the target role, ACK/ is received as a response to the REQ/ signal.
HA1/ (ATN/)	18	Driven by an initiator, ATN/ indicates an attention condition. This signal is received in the target role.
HG/ (BSY/)	16	This signal indicates that the SCSI bus is being used and can be driven by both the initiator and the target device.
HF/ (SEL/)	15	SEL/ is used by an initiator to select a target or by a target to reselect an initiator.
HE/ (RST/)	19	The RST/ signal indicates an SCSI bus RESET condition.

2.2 TRANSCEIVER INTERFACE SIGNALS

SIGNAL	PIN	DESCRIPTION
D0-D7	1,47-41	(I/O) This bidirectional data bus is connected to the NCR 5385E signals SB0-SB7.
DP	2	(I/O) This bidirectional parity signal is connected to the NCR 5385E signal SBP.
DBEN/	34	(INPUT) This signal enables the SCSI bus drivers for DB0-DB7/ and DBP/. It is connected to the NCR 5385E signal SBEN/.
PBEN (TGS)	14	(INPUT) This signal enables the SCSI bus drivers for I/0/, C/D/, MSG/ and REQ/. It is connected to the NCR 5385E signal TGS/.
PAEN (IGS)	12	(INPUT) This signal enables the SCSI bus drivers for ACK/ and ATN/. It is connected to the NCR 5385E signal IGS.
ENCODE (1 of 8) (ARB)	48	(INPUT) This signal enables decode of ID0-ID2/ and asserts the selected SCSI data bus ID. It is connected to the NCR 5385E signal ARB.
ID0-ID2/	35, 39-40	(INPUT) These ID signals are used during arbitration to select the correct DB(n)/ line. They are connected to the NCR 5385E signals ID0-ID2/.
B0 (I/O)	36	(I/O) INPUT/OUTPUT - This pin is used to drive or receive the SCSI signal I/O/. It is connected to the NCR 5385E signal I/P.
B1 (C/D)	29	(I/O) CONTROL/DATA - This pin is used to drive or receive the SCSI signal C/D/. It is connected to the NCR 5385E signal C/D.
B2 (MSG)	27	(I/O) MESSAGE — This pin is used to drive or receive the SCSI signal MSG/. It is connected to the NCR 5385E signal MSG.
B3 (REQ)	28	(I/O) REQUEST - This pin is used to drive or receive the SCSI signal REQ/. It is connected to the NCR 5385E signal REQ.
A0 (ACK)	26	(I/O) ACKNOWLEDGE - This pin controls the SCSI signal ACK/. It is connected to the NCR 5385E signal ACK.
A1 (ATN)	30	(I/O) ATTENTION - This pin controls the SCSI signal ATN/. It is connected to the NCR 5385E signal ATN.
GIN (BSYIN)	25	(OUTPUT) This signal indicates the received state of the SCSI signal BSY/. It is connected to the NCR 5385E signal BSYIN.
GOUT (BSYOUT)	31	(INPUT) This signal drives the SCSI signal BSY/. It is connected to the NCR 5385E signal BSYOUT.
FIN (SELIN)	24	(OUTPUT) This signal indicates the received state of the SCSI signal SEL/. It is connected to the NCR 5385E signal SELIN.
FOUT (SELOUT)	32	(INPUT) This signal drives the SCSI signal SEL/. It is connected to the NCR 5385E signal SELOUT.
EIN (RSTIN)	38	(OUTPUT) This signal indicates the received state of the SCSI signal RST/. It is not connected to the NCR 5385E.
EOUT/ (RSTOUT/)	33	(INPUT) This signal drives the SCSI signal RST/. It is not connected to the NCR 5385E.

2.3 POWER SIGNALS

SIGNAL	PIN	DESCRIPTION
Vdd	37	5 Volts DC
GND	13	Ground

SECTION 3
ELECTRICAL CHARACTERISTICS

3.1 OPERATING CONDITIONS

PARAMETER	SYMBOL	MIN	MAX	UNITS
Supply Voltage	Vdd	4.75	5.25	VDC
Supply Current	Idd		145	mA
Ambient Free Air Temperature	Ta	0	70	degrees C

3.2 INPUT SIGNAL REQUIREMENTS

PARAMETER	CONDITIONS	MIN	MAX	UNITS
High-level Input Voltage Vih		2.0	5.75	VDC
Low-level Input Voltage Vil		-0.3	0.8	VDC
SCSI: High-level Input Current Iih	Vih=5.25V		50	uA
Other Pins: High-level Input Current Iih	Vih=5.25V		10	uA
DP: Low-level Input Current Iil	Vil=0V		-2	mA
SCSI: Low-level Input Current Iil	Vil=0V		-50	uA
All Other Pins: Low-level Input Current Iil	Vil=0V		-10	uA

3.3 OUTPUT SIGNAL REQUIREMENTS

PARAMETER	CONDITIONS	MIN	MAX	UNITS
High-level Output Voltage Voh	Vdd = 4.75V Ioh = 7.0mA	2.4		VDC
Low-level Output Voltage Vol	Vdd = 4.75V Iol = 7.0mA		0.5	VDC
SCSI: Low-level Output Voltage Vol	Vdd = 4.75V Iol = 48.0mA		0.5	VDC

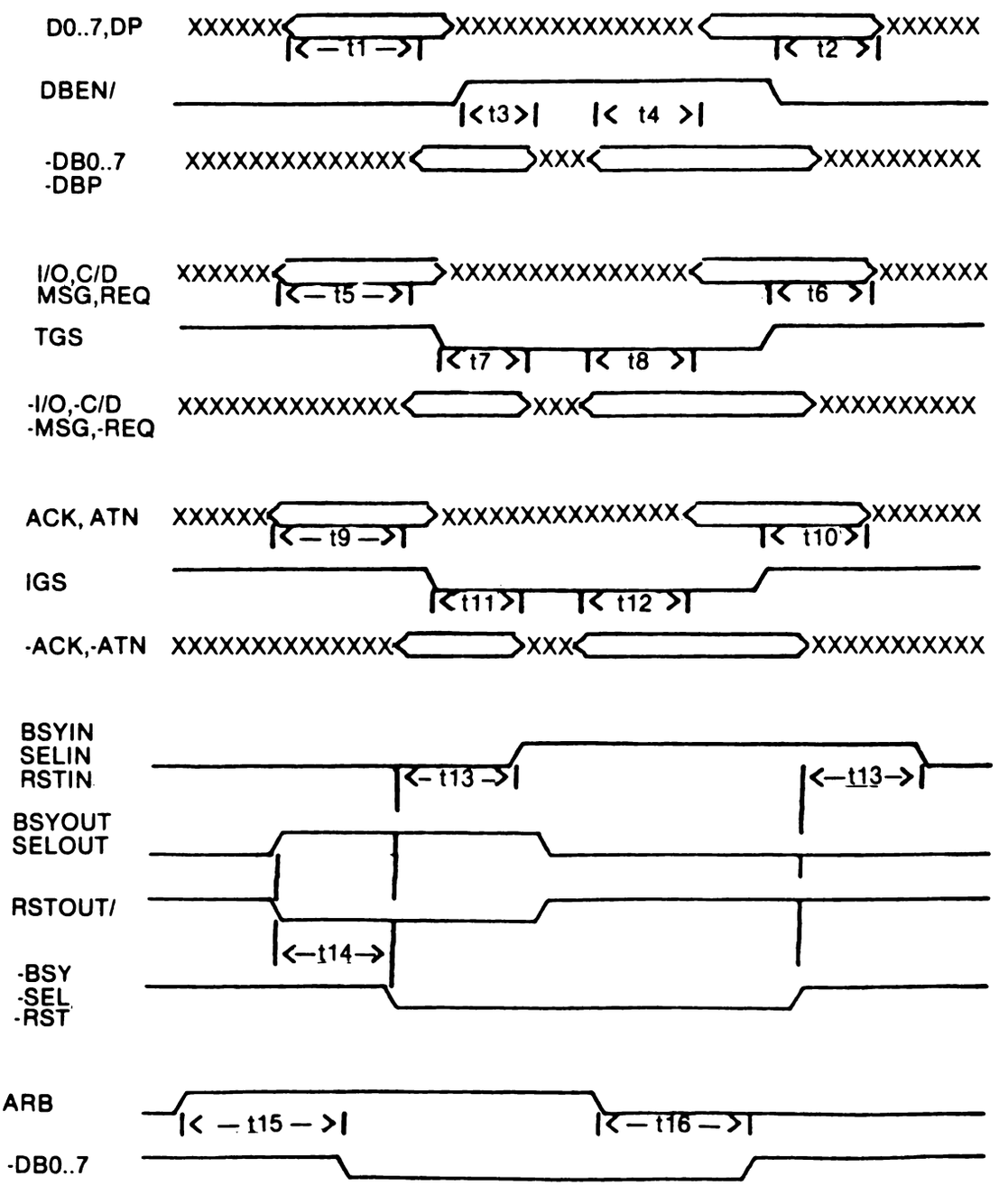
PRELIMINARY

Notice: This is not a final specification.
Some parametric limits are subject to change.

SECTION 4

CHIP TIMING

DRIVER/RECEIVER MODE



NAME	DESCRIPTION	MIN	TYP	MAX	UNITS
T1	Data bus to SCSI bus delay			90	nsec.
T2	DBEN true to Data bus tri-state			90	nsec.
T3	DBEN false to SCSI bus release			100	nsec.
T4	SCSI bus to Data bus delay			60	nsec.
T5	Control signal delay to SCSI bus			70	nsec.
T6	TGS true to input release			50	nsec.
T7	TGS false to SCSI bus release			70	nsec.
T8	SCSI bus receiver delay			50	nsec.
T9	Control signal delay to SCSI bus			70	nsec.
T10	IGS true to input release			50	nsec.
T11	IGS false to SCSI bus release			70	nsec.
T12	SCSI bus receiver delay			50	nsec.
T13	Receiver delay BSY RST SEL			50	nsec.
T14	Driver delay BSY RST SEL			70	nsec.
T15	ARB true to DB(id)/ true			70	nsec.
T16	ARB false to DB(id)/ false			70	nsec.

PRELIMINARY

Notice: This is not a final specification.
Some parametric limits are subject to change.

HD 68450,Y

DMAC (Direct Memory Access Controller)

HITACHI

Microprocessor implemented systems are becoming increasingly complex, particularly with the advent of high-performance 16-bit MPU devices with large memory addressing capability. In order to maintain high throughput, large blocks of data must be moved within these systems in a quick, efficient manner with minimum intervention by the MPU itself.

The HD68450 Direct Memory Access Controller (DMAC) is designed specifically to complement the performance and architectural capabilities of the HD68000 MPU by providing the following features:

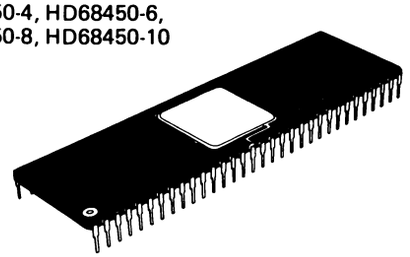
- HMCS68000 Bus Compatible
- 4 independent DMA Channels
- Memory-to-Memory, Memory-to-Device, Device-to-Memory Transfers
- MMU Compatible
- Array-Chained and Linked-Array-Chained Operations
- On-Chip Registers that allow Complete Software Control by the System MPU
- Interface Lines for Requesting, Acknowledging, and Incidental Control of the Peripheral Devices
- Variable System Bus Bandwidth Utilization
- Programmable Channel Prioritization
- 2 Vectored interrupts for each Channel
- Auto-Request and External-Request Transfer Modes
- +5 Volt Operation

The DMAC functions by transferring a series of operands (data) between memory and peripheral device; operand sizes can be byte, word, or long word. A block is a sequence of operations; the number of operands in a block is determined by a transfer count. A single-channel operation may involve the transfer of several blocks of data between memory and device.

■ TYPE OF PRODUCTS

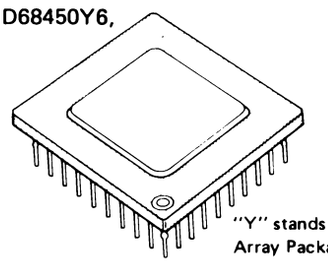
Type No.	Bus Timing	Packaging
HD68450-4	4MHz	DC-64
HD68450-6	6MHz	
HD68450-8	8MHz	
HD68450-10	10MHz	
HD68450Y4	4MHz	PGA-68
HD68450Y6	6MHz	
HD68450Y8	8MHz	
HD68450Y10	10MHz	

HD68450-4, HD68450-6,
HD68450-8, HD68450-10



(DC-64)

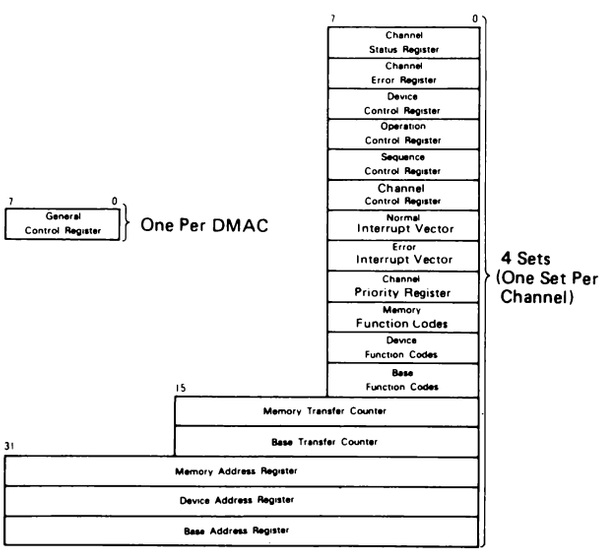
HD68450Y4, HD68450Y6,
HD68450Y8,
HD68450Y10



"Y" stands for Pin Grid Array Package.

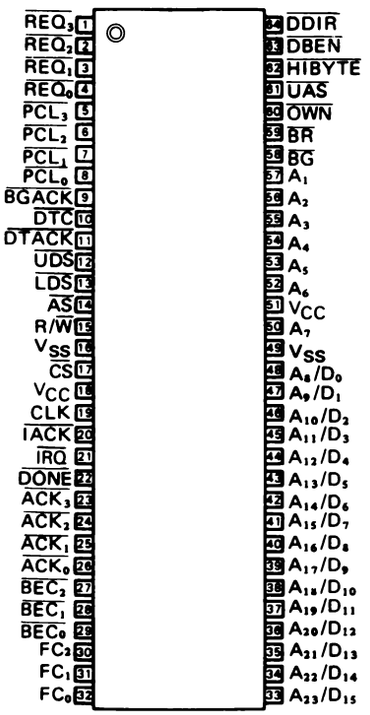
(PGA-68)

■ PROGRAMMING MODEL



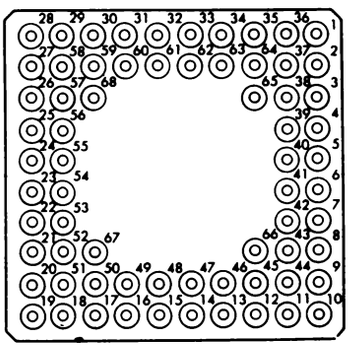
■ PIN ARRANGEMENT

● HD68450



(Top View)

● HD68450Y



(Bottom View)

Pin No.	Function	Pin No.	Function	Pin No.	Function	Pin No.	Function
1	N/C	18	$\overline{PCL_1}$	35	A_{19}/D_{11}	52	$\overline{BGACK}$
2	A_{13}/D_5	19	$\overline{DTACK}$	36	A_{17}/D_9	53	$\overline{LDS}$
3	A_{11}/D_3	20	$\overline{UDS}$	37	A_{15}/D_7	54	V_{SS}
4	A_{10}/D_2	21	$\overline{AS}$	38	A_{12}/D_4	55	V_{CC}
5	A_8/D_0	22	R/$\overline{W}$	39	A_9/D_1	56	$\overline{DONE}$
6	A_7	23	N/C	40	V_{SS}	57	$\overline{IRQ}$
7	A_6	24	$\overline{CS}$	41	V_{CC}	58	$\overline{ACK_2}$
8	A_5	25	CLK	42	A_4	59	$\overline{BEC_2}$
9	A_3	26	$\overline{IACK}$	43	A_2	60	$\overline{BEC_0}$
10	N/C	27	$\overline{ACK_3}$	44	$\overline{BG}$	61	FC_0
11	$\overline{BR}$	28	$\overline{ACK_0}$	45	$\overline{OWN}$	62	A_{21}/D_{13}
12	$\overline{UAS}$	29	$\overline{BEC_1}$	46	$\overline{HIBYTE}$	63	A_{18}/D_{10}
13	$\overline{DBEN}$	30	FC_2	47	$\overline{DDIR}$	64	A_{16}/D_8
14	$\overline{REQ_3}$	31	FC_1	48	$\overline{REQ_1}$	65	A_{14}/D_6
15	$\overline{REQ_2}$	32	A_{23}/D_{15}	49	$\overline{PCL_2}$	66	A_1
16	$\overline{REQ_0}$	33	A_{22}/D_{14}	50	$\overline{PCL_0}$	67	$\overline{DTC}$
17	$\overline{PCL_3}$	34	A_{20}/D_{12}	51	N/C	68	$\overline{ACK_1}$

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}*	-0.3 ~ +7.0	V
Input Voltage	V_{in}*	-0.3 ~ +7.0	V
Operating Temperature Range	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	-55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input Voltage	V_{IH}*	2.0	–	V_{CC}	V
	V_{IL}*	-0.3	–	0.8	V
Operating Temperature	T_{opr}	0	25	70	°C

* With respect to V_{SS} (SYSTEM GND)

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS (V_{CC} = 5V ±5%, V_{SS} = 0V, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage		V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		V_{SS}-0.3	–	0.8	V
Input Leakage Current	$\overline{CS}$, $\overline{IACK}$, $\overline{BG}$, CLK, $\overline{BEC_0}$ ~ $\overline{BEC_2}$, $\overline{REQ_0}$ ~ $\overline{REQ_3}$	I_{in}		–	–	10	μA
Three-State (Off State) Input Current	A_1 ~ A_7, D_0 ~ D_{15}/A_8 ~ A_{23}, $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$, R/$\overline{W}$, $\overline{UAS}$, $\overline{DTACK}$, $\overline{BGACK}$, $\overline{OWN}$, $\overline{DTC}$, $\overline{HIBYTE}$, $\overline{DDIR}$, $\overline{DBEN}$, FC_0 ~ FC_2	I_{TSI}		–	–	10	μA
Open Drain (Off State) Input Current	$\overline{IRQ}$, $\overline{DONE}$	I_{ODI}		–	–	20	μA
Output "High" Voltage	A_1 ~ A_7, D_0 ~ D_{15}/A_8 ~ A_{23}, $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$, R/$\overline{W}$, $\overline{UAS}$, $\overline{DTACK}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{OWN}$, $\overline{DTC}$, $\overline{HIBYTE}$, $\overline{DDIR}$, $\overline{DBEN}$, $\overline{ACK_0}$ ~ $\overline{ACK_3}$, $\overline{PCL_0}$ ~ $\overline{PCL_3}$, FC_0 ~ FC_2	V_{OH}	I_{OH} = -400 μA	2.4	–	–	V
Output "Low" Voltage	A_1 ~ A_7, FC_0 ~ FC_2	V_{OL}	I_{OL} = 3.2 mA	–	–	0.5	V
	D_0 ~ D_{15}/A_8 ~ A_{23}, $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$, R/$\overline{W}$, $\overline{DTACK}$, $\overline{BR}$, $\overline{OWN}$, $\overline{DTC}$, $\overline{HIBYTE}$, $\overline{DDIR}$, $\overline{DBEN}$, $\overline{ACK_0}$ ~ $\overline{ACK_3}$, $\overline{UAS}$, $\overline{PCL_0}$ ~ $\overline{PCL_3}$, $\overline{BGACK}$	V_{OL}	I_{OL} = 5.3 mA	–	–	0.5	
	$\overline{IRQ}$, $\overline{DONE}$	V_{OL}	I_{OL} = 8.9 mA	–	–	0.5	
Power Dissipation		P_D	f = 8 MHz, V_{CC} = 5.0 V, Ta = 25°C	–	1.4	2.0	W
Capacitance		C_{in}	V_{in} = 0V, Ta = 25°C, f = 1 MHz	–	–	15	pF

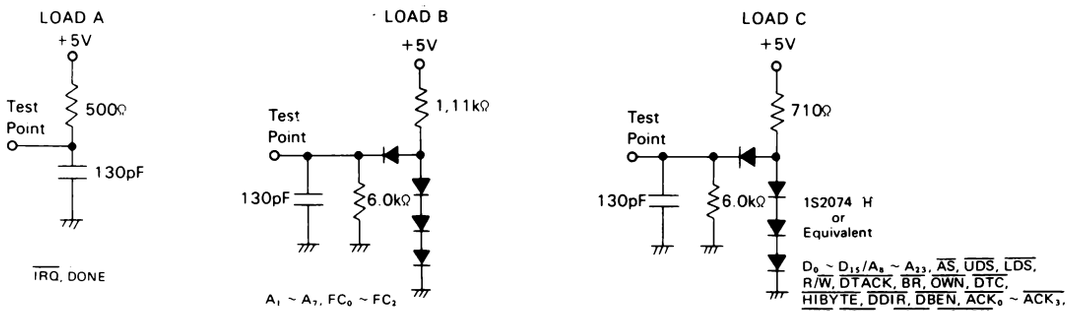


Figure 1 Test Loads

• AC ELECTRICAL SPECIFICATIONS (V_{CC} = 5V ±5%, V_{SS} = 0V, Ta = 0~+70°C)

No.	Item	Symbol	Test Condition	4MHz HD68450-4 HD68450Y4		6MHz HD68450-6 HD68450Y6		8MHz HD68450-8 HD68450Y8		10MHz HD68450-10 HD68450Y10		Unit
				min	max	min	max	min	max	min	max	
	Frequency of Operation	f		2	4	2	6	2	8	2	10	MHz
1	Clock Period	t_{cyc}		250	500	167	500	125	500	100	500	ns
2	Clock Width Low	t_{CL}		115	250	75	250	55	250	45	250	ns
3	Clock Width High	t_{CH}		115	250	75	250	55	250	45	250	ns
4	Clock Fall Time	t_{Cf}		–	10	–	10	–	10	–	10	ns
5	Clock Rise Time	t_{Cr}		–	10	–	10	–	10	–	10	ns
6	Asynchronous Input Setup Time	t_{ASI}		30	–	25	–	20	–	15	–	ns
7	Data in to $\overline{DBEN}$ Low	t_{DIDBL}		0	–	0	–	0	–	0	–	ns
8	$\overline{DTACK}$ Low to Data Invalid	t_{DTLDI}		0	–	0	–	0	–	0	–	ns
9	Address in to $\overline{AS}$ in Low	t_{AIASL}		0	–	0	–	0	–	0	–	ns
10	$\overline{AS}$, $\overline{DS}$ in High to Address in Invalid	t_{SIHAIV}		0	–	0	–	0	–	0	–	ns
10A	$\overline{DS}$ in High to $\overline{CS}$ High	t_{DSHCSH}		–	1.0	–	1.0	–	1.0	–	1.0	clk. per.
11	Clock High to $\overline{DDIR}$ Low	t_{CHDRL}		–	90	–	80	–	70	–	60	ns
12	Clock High to $\overline{DDIR}$ High	t_{CHDRH}		–	90	–	80	–	70	–	60	ns
13	$\overline{DS}$ in High to $\overline{DDIR}$ High Impedance	t_{DSHDRZ}		–	160	–	140	–	120	–	110	ns
14	Clock Low to $\overline{DBEN}$ Low	t_{CLDBL}		–	90	–	80	–	70	–	60	ns
15	Clock Low to $\overline{DBEN}$ High	t_{CLDBH}		–	90	–	80	–	70	–	60	ns
16	$\overline{DS}$ in High to $\overline{DBEN}$ High Impedance	t_{DSHDBZ}		–	160	–	140	–	120	–	110	ns
17	Clock High to Data Out Valid (MPU read)	t_{CHDVM}		–	290	–	230	–	180	–	160	ns
18	$\overline{DS}$ in High to Data Out Invalid	t_{DSHDZn}		0	–	0	–	0	–	0	–	ns
19	$\overline{DS}$ in High to Data High Impedance	t_{DSHDZ}		–	160	–	140	–	120	–	110	ns
20	Clock Low to $\overline{DTACK}$ Low	t_{CLDTL}		–	90	–	80	–	70	–	60	ns
21	DS in High to $\overline{DTACK}$ High	t_{DSHDTH}		–	160	–	130	–	110	–	110	ns
22	$\overline{DTACK}$ Width High	t_{DTH}		10	–	10	–	10	–	10	–	ns
23	$\overline{DS}$ in High to $\overline{DTACK}$ High Impedance	t_{DSHDTZ}		–	220	–	200	–	180	–	160	ns
24	$\overline{DTACK}$ Low to $\overline{DS}$ in High	t_{DTLDSH}		0	–	0	–	0	–	0	–	ns
25	$\overline{REQ}$ Width Low	t_{REQL}		2.0	–	2.0	–	2.0	–	2.0	–	clk. per.
26	$\overline{REQ}$ Low to $\overline{BR}$ Low	t_{RELBRL}		500	–	334	–	250	–	200	–	ns
27	Clock High to $\overline{BR}$ Low	t_{CHBRL}	Fig. 1 ~ Fig. 8	–	90	–	80	–	70	–	60	ns
28	Clock High to $\overline{BR}$ High	t_{CHBRH}		–	90	–	80	–	70	–	60	ns
29	$\overline{BG}$ Low to $\overline{BGACK}$ Low	t_{BGLBL}		4.5	–	4.5	–	4.5	–	4.5	→	clk. per.
30	$\overline{BR}$ Low to MPU Cycle End ($\overline{AS}$ in High)	t_{BRLASH}		0	–	0	–	0	–	0	–	ns
31	MPU Cycle End ($\overline{AS}$ in High) to $\overline{BGACK}$ Low	t_{ASHBL}		4.5	5.5	4.5	5.5	4.5	5.5	4.5	5.5	clk. per.
32	$\overline{REQ}$ Low to $\overline{BGACK}$ Low	t_{REQLBL}		12.0	–	12.0	–	12.0	–	12.0	–	clk. per.
33	Clock High to $\overline{BGACK}$ High	t_{CHBL}		–	90	–	80	–	70	–	60	ns
34	Clock High to $\overline{BGACK}$ High	t_{CHBH}		–	90	–	80	–	70	–	60	ns
35	Clock Low to $\overline{BGACK}$ High Impedance	t_{CLBZ}		–	120	–	100	–	80	–	70	ns
36	Clock High to FC Valid	t_{CHFCV}		–	140	–	120	–	100	–	90	ns
37	Clock High to Address Valid	t_{CHAV}		–	160	–	140	–	120	–	110	ns
38	Clock High to Address/FC/Data High Impedance	t_{CHAZx}		–	140	–	120	–	100	–	100	ns
39	Clock High to Address/FC/Data Invalid	t_{CHAZn}		0	–	0	–	0	–	0	–	ns
40	Clock Low to Address High Impedance	t_{CLAZ}		–	140	–	120	–	100	–	90	ns
41	Clock High to $\overline{UAS}$ Low	t_{CHUL}		–	90	–	80	–	70	–	60	ns
42	Clock High to $\overline{UAS}$ High	t_{CHUH}		–	90	–	80	–	70	–	60	ns
43	Clock Low to $\overline{UAS}$ High Impedance	t_{CLUZ}		–	120	–	100	–	80	–	70	ns
44	$\overline{UAS}$ High to Address Invalid	t_{UHAI}		50	–	40	–	30	–	20	–	ns
45	Clock High to $\overline{AS}$, $\overline{DS}$ Low	t_{CHSL}		–	80	–	70	–	60	–	55	ns
46	Clock Low to $\overline{DS}$ Low (write)	t_{CLDSL}		–	80	–	70	–	60	–	55	ns
47	Clock Low to $\overline{AS}$, $\overline{DS}$ High	t_{CLSH}		–	90	–	80	–	70	–	60	ns
48	Clock Low to $\overline{AS}$, $\overline{DS}$ High Impedance	t_{CLSZ}		–	120	–	100	–	80	–	70	ns
49	$\overline{AS}$ Width Low	t_{ASL}		545	–	350	–	255	–	195	–	ns
50	$\overline{DS}$ Width Low	t_{DSL}		420	–	265	–	190	–	145	–	ns
51	$\overline{AS}$, $\overline{DS}$ Width High	t_{SH}		285	–	180	–	150	–	105	–	ns
52	Address/FC Valid to $\overline{AS}$, $\overline{DS}$ Low	t_{AVSL}		50	–	40	–	30	–	20	–	ns
53	$\overline{AS}$, $\overline{DS}$ High to Address/FC/Data Invalid	t_{SHAZ}		50	–	40	–	30	–	20	–	ns
54	Clock High to R/$\overline{W}$ Low	t_{CHRL}		–	90	–	80	–	70	–	60	ns
55	Clock High to R/$\overline{W}$ High	t_{CHRH}		–	90	–	80	–	70	–	60	ns

(to be continued)

No.	Item	Symbol	Test Condition	4MHz HD68450-4 HD68450Y4		6MHz HD68450-6 HD68450Y6		8MHz HD68450-8 HD68450Y8		10MHz HD68450-10 HD68450Y10		Unit
				min	max	min	max	min	max	min	max	
56	Clock Low to R/$\overline{W}$ High Impedance	t_{CLRZ}	Fig. 1 ~ Fig. 8	–	120	–	100	–	80	–	70	ns
57	Address/FC Valid to R/$\overline{W}$ Low	t_{AVRL}		100	–	40	–	20	–	10	–	ns
58	R/$\overline{W}$ Low to $\overline{DS}$ Low (write)	t_{RLSL}		285	–	170	–	120	–	90	–	ns
59	$\overline{DS}$ High to R/$\overline{W}$ High	t_{SHRH}		60	–	50	–	40	–	20	–	ns
60	Clock Low to $\overline{OWN}$ Low	t_{CLOL}		–	90	–	80	–	70	–	60	ns
61	Clock Low to $\overline{OWN}$ High	t_{CLOH}		–	90	–	80	–	70	–	60	ns
62	Clock High to $\overline{OWN}$ High Impedance	t_{CHOZ}		–	120	–	100	–	80	–	70	ns
63	$\overline{OWN}$ Low to $\overline{BGACK}$ Low	t_{OLBL}		50	–	40	–	30	–	20	–	ns
64	$\overline{BGACK}$ High to $\overline{OWN}$ High	t_{BHOH}		50	–	40	–	30	–	20	–	ns
65	$\overline{OWN}$ Low to $\overline{UAS}$ Low	t_{OLUL}		50	–	40	–	30	–	20	–	ns
66	Clock High to $\overline{ACK}$ Low	t_{CHACL}		–	90	–	80	–	70	–	60	ns
67	Clock Low to $\overline{ACK}$ Low	t_{CLACL}		–	90	–	80	–	70	–	60	ns
68	Clock High to $\overline{ACK}$ High	t_{CHACH}		–	90	–	80	–	70	–	60	ns
69	$\overline{ACK}$ Low to $\overline{DS}$ Low	t_{ACLDSL}		230	–	140	–	100	–	80	–	ns
70	$\overline{DS}$ High to $\overline{ACK}$ High	t_{DSHACH}		50	–	40	–	30	–	20	–	ns
71	Clock High to $\overline{HIBYTE}$ Low	t_{CHHIL}		–	90	–	80	–	70	–	60	ns
72	Clock Low to $\overline{HIBYTE}$ Low	t_{CLHIL}		–	90	–	80	–	70	–	60	ns
73	Clock High to $\overline{HIBYTE}$ High	t_{CHHIH}		–	90	–	80	–	70	–	60	ns
74	Clock Low to $\overline{HIBYTE}$ High Impedance	t_{CLHIZ}		–	120	–	100	–	80	–	70	ns
75	Clock High to $\overline{DTC}$ Low	t_{CHDTL}		–	90	–	80	–	70	–	60	ns
76	Clock High to $\overline{DTC}$ High	t_{CHDTH}		–	90	–	80	–	70	–	60	ns
77	Clock Low to $\overline{DTC}$ High Impedance	t_{CLDTZ}		–	120	–	100	–	80	–	70	ns
78	$\overline{DTC}$ Width Low	t_{DTCL}		230	–	147	–	105	–	80	–	ns
79	$\overline{DTC}$ Low to $\overline{DS}$ High	t_{DTLDH}		95	–	50	–	30	–	20	–	ns
80	Clock High to $\overline{DONE}$ Low	t_{CHDOL}		–	90	–	80	–	70	–	60	ns
81	Clock Low to $\overline{DONE}$ Low	t_{CLDOL}		–	90	–	80	–	70	–	60	ns
82	Clock High to $\overline{DONE}$ High	t_{CHDOH}		–	150	–	140	–	130	–	120	ns
83	Clock Low to $\overline{DDIR}$ High Impedance	t_{CLDRZ}		–	120	–	100	–	80	–	70	ns
84	Clock Low to $\overline{DBEN}$ High Impedance	t_{CLDBZ}		–	120	–	100	–	80	–	70	ns
85	$\overline{DDIR}$ Low to $\overline{DBEN}$ Low	t_{DRLDBL}		50	–	40	–	30	–	20	–	ns
86	$\overline{DBEN}$ High to $\overline{DDIR}$ High	t_{DBHDRH}		50	–	40	–	30	–	20	–	ns
87	$\overline{DBEN}$ Low to Address/Data High Impedance	t_{DBLAZ}		–	17	–	17	–	17	–	17	ns
88	Clock Low to $\overline{PCL}$ Low (1/8 clock)	t_{CLPL}		–	90	–	80	–	70	–	60	ns
89	Clock Low to $\overline{PCL}$ High (1/8 clock)	t_{CLPH}		–	90	–	80	–	70	–	60	ns
90	$\overline{PCL}$ Width Low (1/8 clock)	t_{PCLL}		4.0	–	4.0	–	4.0	–	4.0	–	clk. per.
91	$\overline{DTACK}$ Low to Data In (setup time)	t_{DALDI}		–	320	–	200	–	150	–	115	ns
92	$\overline{DS}$ High to Data Invalid (hold time)	t_{SHDI}		0	–	0	–	0	–	0	–	ns
93	$\overline{DS}$ High to $\overline{DTACK}$ High	t_{SHDAH}		0	240	0	160	0	120	0	90	ns
94	Data Out Valid to $\overline{DS}$ Low	t_{DOSL}		0	–	0	–	0	–	0	–	ns
95	Data In to Clock Low (setup time)	t_{DICL}		30	–	25	–	15	–	15	–	ns
96	$\overline{BEC}$ Low to $\overline{DTACK}$ Low	t_{BECDAL}		50	–	50	–	50	–	50	–	ns
97	$\overline{BEC}$ Width Low	t_{BECL}		2.0	–	2.0	–	2.0	–	2.0	–	clk. per.
98	Clock High to $\overline{IRQ}$ Low	t_{CHIRL}		–	90	–	80	–	70	–	60	ns
99	Clock High to $\overline{IRQ}$ High	t_{CHIRH}		–	150	–	140	–	130	–	120	ns
100	$\overline{READY}$ In to $\overline{DTC}$ Low (Read)	t_{RALDTL}		270	–	180	–	145	–	120	–	ns
101	$\overline{READY}$ In to $\overline{DS}$ Low (Write)	t_{RALDSL}		395	–	240	–	205	–	170	–	ns
102	$\overline{DS}$ High to $\overline{READY}$ High	t_{DSHRAH}		0	240	0	160	0	120	0	90	ns
103	$\overline{DONE}$ In Low to $\overline{DTACK}$ Low	t_{DOLDAL}		50	–	50	–	50	–	50	–	ns
104	$\overline{DS}$ High to $\overline{DONE}$ In High	t_{DSHDOH}		0	240	0	160	0	120	0	90	ns
105	Asynchronous Input Hold Time	t_{ASIH}		15	–	15	–	15	–	15	–	ns

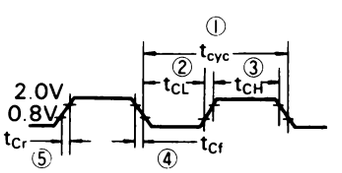
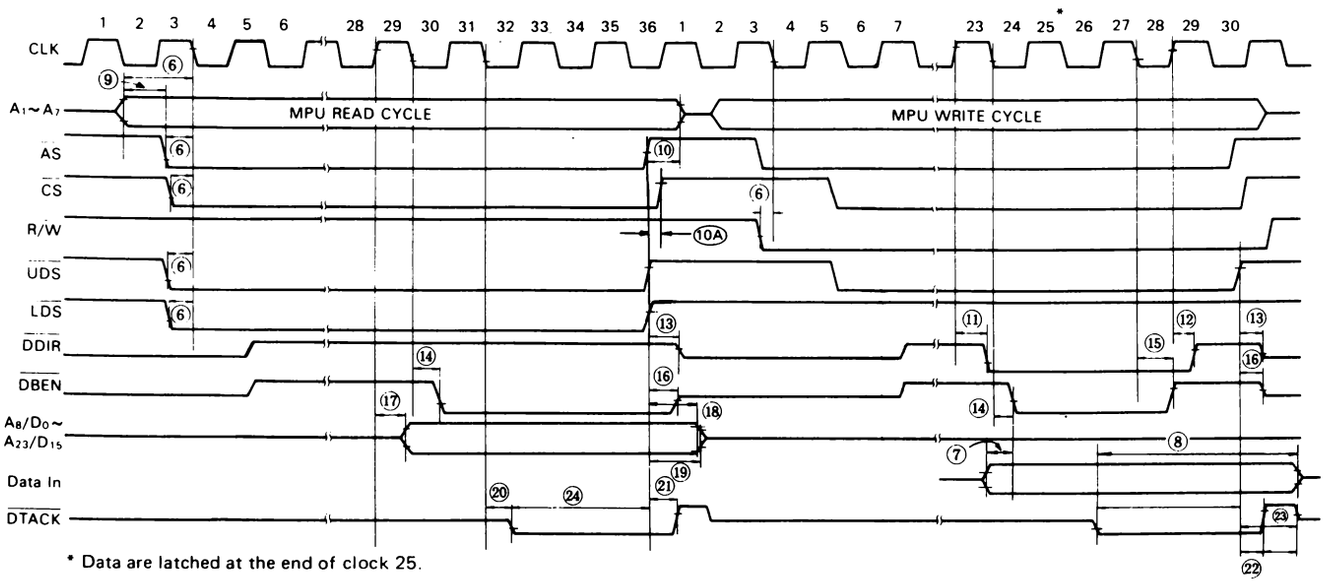
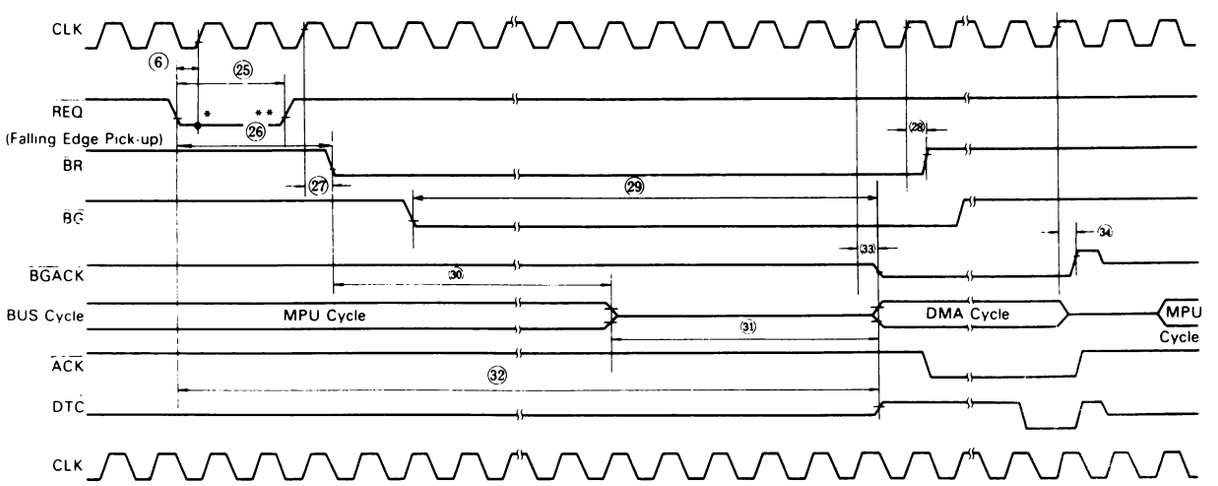


Figure 2 Input Clock Waveform



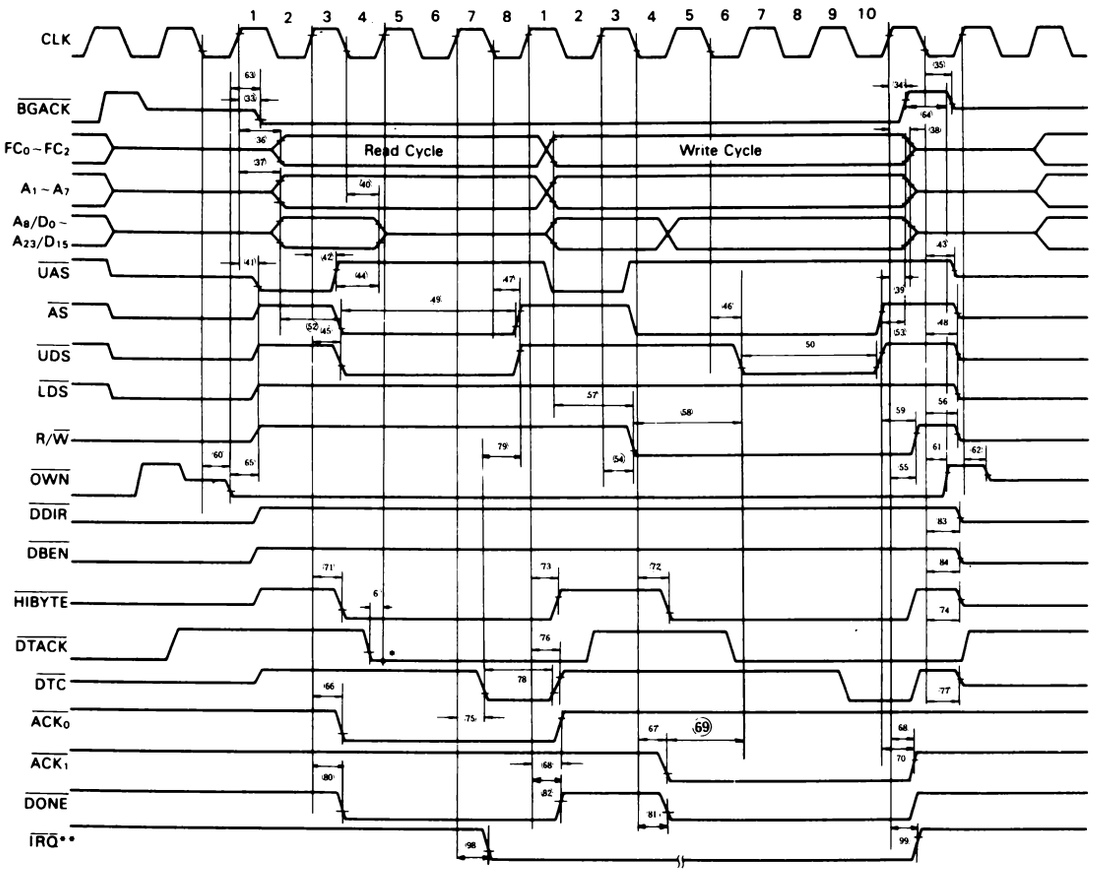
* Data are latched at the end of clock 25.

Figure 3 AC Electrical Waveforms – MPU Read/Write



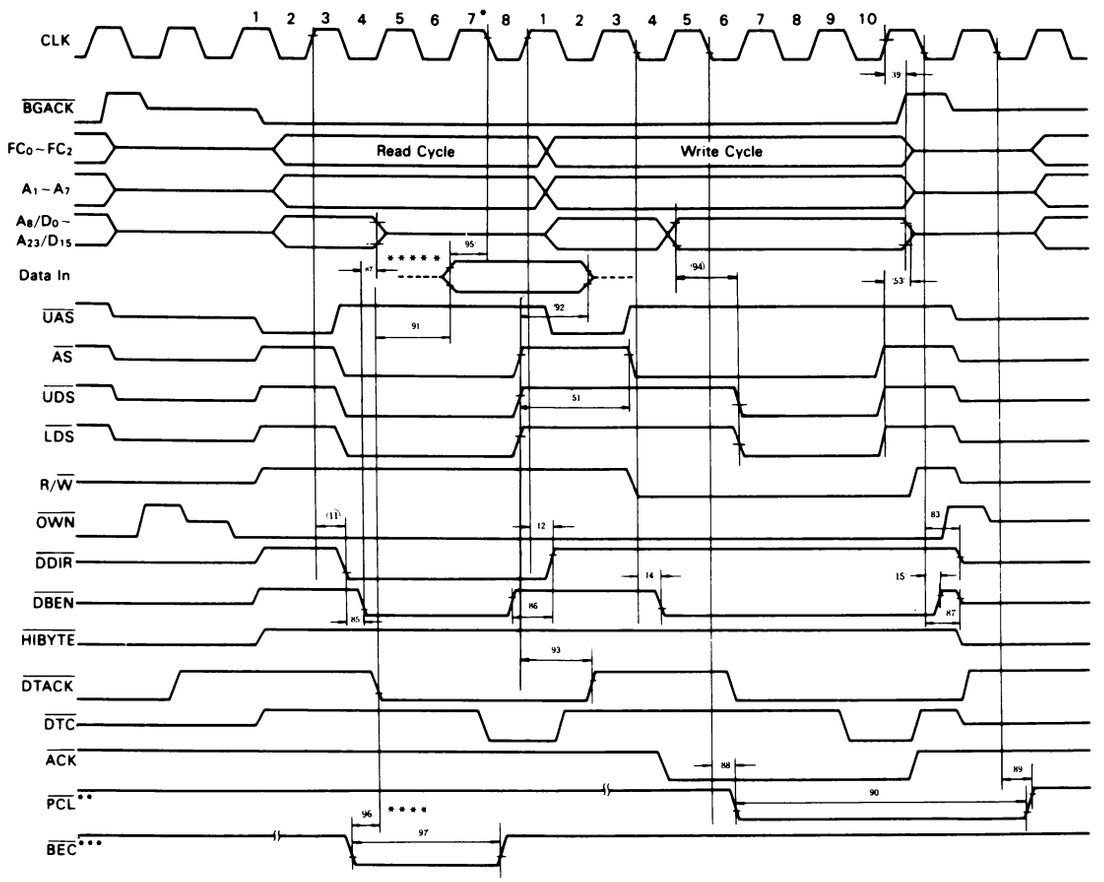
* $\overline{REQ}$ is picked up at the rising edge of CLK in cycle steal and Burst modes.
** $\overline{BR}$ isn't asserted while some $\overline{BEC}$ exception condition exists or DMAC is accessed by MPU.

Figure 4 AC Electrical Waveforms – Bus Arbitration



* $\overline{\text{DTACK}}$ is picked up at the rising edge of CLK. This is different from HD68000.
** This timing is not related to DMA Read/Write (Single Cycle) sequence.

Figure 5 AC Electrical Waveforms — DMA Read/Write (Single Cycle)



* Data are latched at the end of clock 7. This timing is the same as HD68000.
** This timing is not related to DMA Read/Write (Dual Cycle) sequence. This timing is only applicable when 1/8 clock pulse mode is selected.
*** This timing is applicable when a bus exception occures.
**** If #6 is satisfied for both $\overline{\text{DTACK}}$ and $\overline{\text{BEC}}$, #96 may be Ons.
***** If the propagation delay of the external bidirectional buffer LS245 is less than 17nsec, the conflict may occur between the address output of the DMAC and the system data bus. In this case, the output of $\overline{\text{DBEN}}$ must be delayed externally.

Figure 6 AC Electrical Waveforms – DMA Read/Write (Dual Cycle)

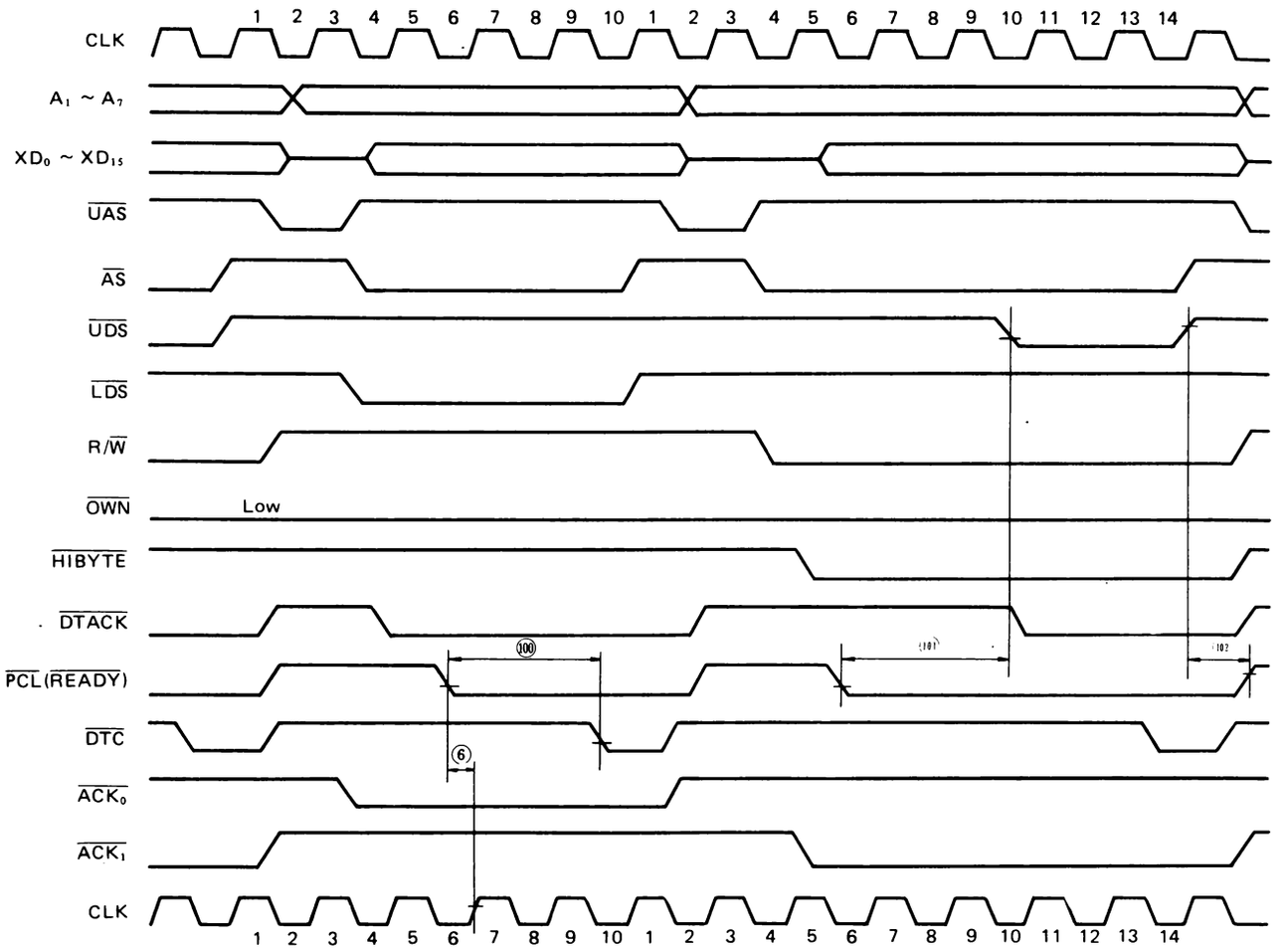
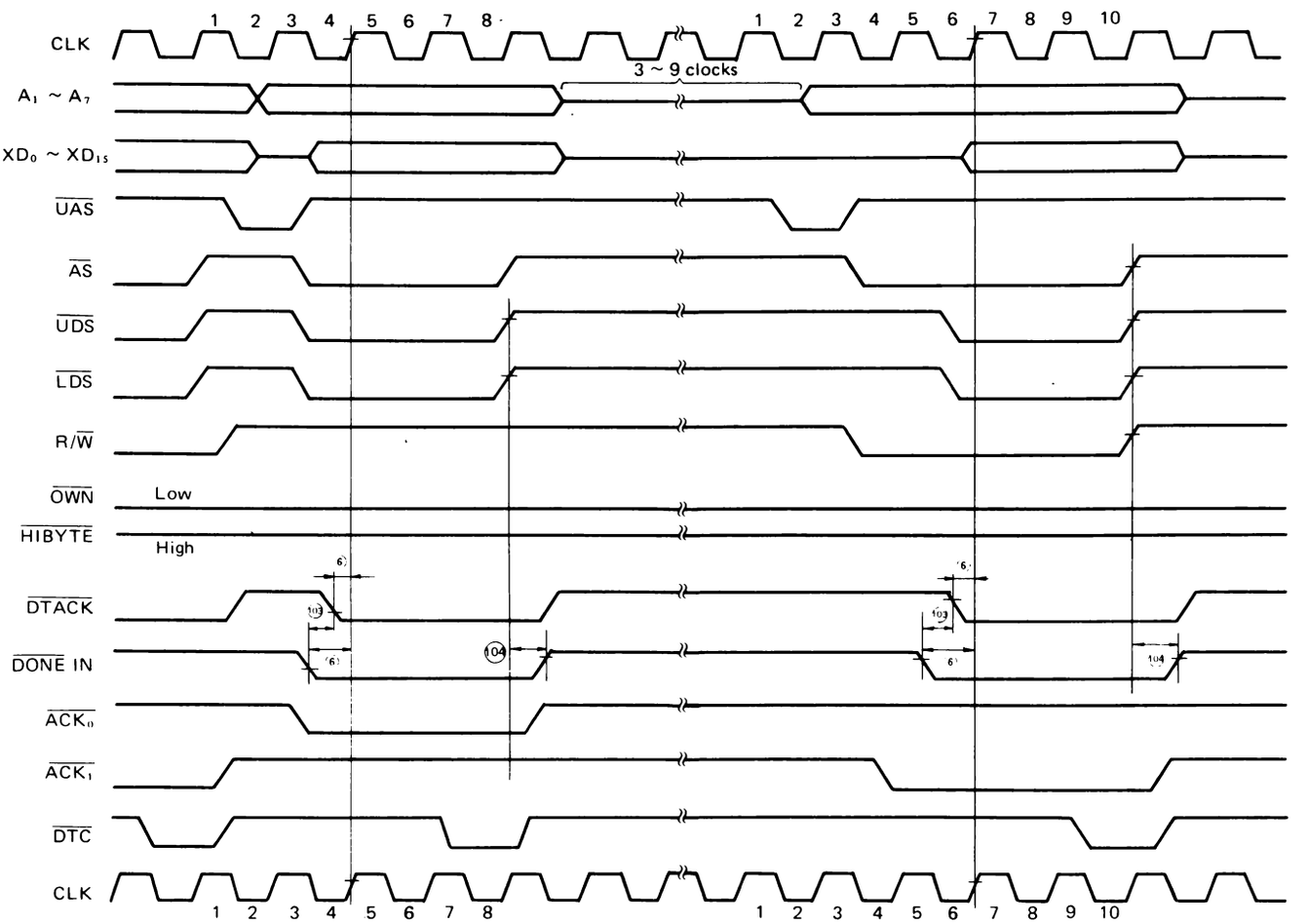


Figure 7 AC Electrical Waveforms — DMA Read/Write (Single Cycle with $\overline{PCL}$)



* If #6 is satisfied for both $\overline{DTACK}$ and $\overline{DONE}$, #103 may be 0ns.

Figure 8 AC Electrical Waveforms – $\overline{DONE}$ Input

(NOTES for Figure 3 through 8)

1) Setup time for the asynchronous inputs $\overline{BG}$, $\overline{BGACK}$, $\overline{CS}$, $\overline{IACK}$, $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$, and R/$\overline{W}$ guarantees their recognition at the next falling edge of the clock. Setup time for $\overline{BEC_0} \sim \overline{BEC_2}$, $\overline{REQ_0} \sim \overline{REQ_3}$, $\overline{PCL_0} \sim \overline{PCL_3}$, $\overline{DTACK}$, and $\overline{DONE}$ guarantees their recognition at the next rising edge of the clock.
2) Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts.
3) These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Refer to other functional descriptions and their related diagrams for device operation.

■ SIGNAL DESCRIPTION

The following section identifies the signals used in the DMAC. In the definitions, "MPU mode" refers to the state when the DMAC is chip selected by MPU. The term "DMA mode" refers to the state when the DMAC assumes ownership of the bus. The DMAC is in the "IDLE mode" at all other times. Moreover, the DMA bus cycle refers to the bus cycle that is executed by the DMAC in the "DMA mode".

NOTE) In this data sheet, the state of the signals is described with these words: active or assert, inactive or negate.

This is done to avoid confusion when dealing with a mixture of "active-low" and "active-high" signals. The term assert or assertion is used to indicate that a signal is active or true independent of whether that voltage is low or high. The term negate or negation is used to indicate that a signal is inactive or false.

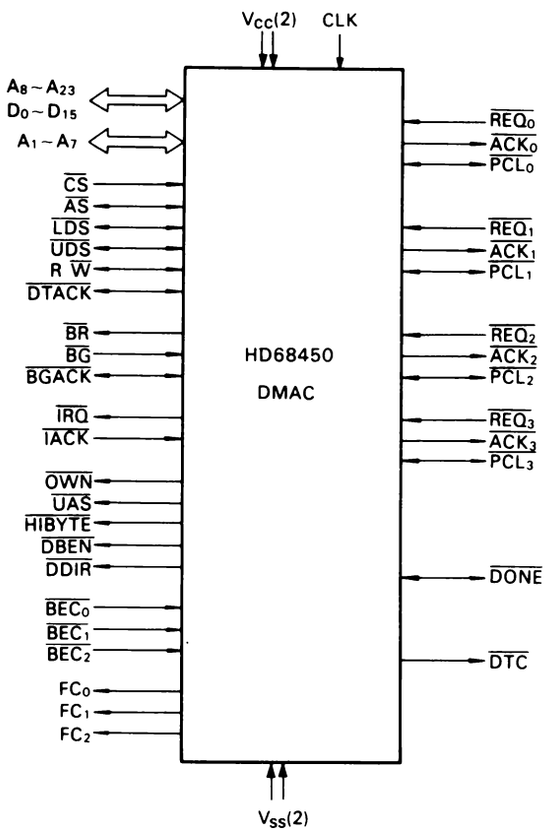


Figure 9 Input and Output Signals

- **Address/Data Bus (A_8/D_0 through A_{23}/D_{15})**

Input/Output Active-high	Three-statable

These lines are time multiplexed for address and data bus. The lines $\overline{DDIR}$, $\overline{DBEN}$, $\overline{UAS}$ and $\overline{OWN}$ are used to control the demultiplexing of the data and address lines externally. Demultiplexing is explained in the later section. The bi-directional data bus is used to transfer data between DMAC, MPU, memory and I/O devices.

Address lines are outputs to address memory and I/O devices.

- **Address Bus (A_1 through A_7)**

Input/Output Active-high	Three-statable

In the MPU mode, the DMAC internal registers are accessed with these lines and $\overline{LDS}$, $\overline{UDS}$. The address map for these registers is shown in Table 1. During a DMA bus cycle, A_1-A_7 are outputs containing the low order address bits of the location being accessed.

- **Function Code (FC_0 through FC_2)**

Output Active-high	Three-statable

These output signals provide the function codes during DMA bus cycles. They are three-stated except in the DMA bus cycles. They are used to control the HMCS68000 memories.

- **Clock (CLK)**

Input

This is the input clock to the HD68450, and should never be terminated at any time. This clock can be different from the MPU clock since HD68450 operates completely asynchronously.

- **Chip Select ($\overline{CS}$)**

Input Active low

This input signal is used to chip select the DMAC in "MPU" mode. If the $\overline{CS}$ input is asserted during a bus cycle which is generated by the DMAC, the DMAC internally terminates the bus cycle and signals an address error. This function protects the DMAC from accessing its own register.

- **Address Strobe ($\overline{AS}$)**

Input/Output Active low	Three-statable

In the "MPU mode", this line is an input indicating valid address input, and during the DMA bus cycle it is an output indicating valid the address output from the DMAC on the address bus.

The DMAC monitors these input lines during bus arbitration to determine the completion of the bus cycle by the MPU or other bus masters.

- **Upper Address Strobe ($\overline{UAS}$)**

Output Active low	Three-statable

This line is an output to latch the upper address lines on the multiplexed data/address lines. It is three-stated except in the "DMA mode".

- **Own ($\overline{OWN}$)**

Output Active low	Three-statable

This line is asserted by the DMAC during DMA mode, and is used to control the output of the address line latch. This line may also be used to control the direction of bi-directional buffers when loads on $\overline{AS}$, $\overline{LDS}$, $\overline{UDS}$, R/$\overline{W}$ and other signals exceed the drive capability. It is three-stated in the "MPU mode" and the "IDLE mode"

- **Data Direction ($\overline{DDIR}$)**

Outputs	Three-statable
Active low	(when data direction is input to the DMAC)
Active high	(when the data direction is output from the DMAC)

This line controls the direction of data through the bidirectional buffer which used to demultiplex the data/address lines. It is three-stated during the "IDLE mode"

- **Data Bus Enable ($\overline{DBEN}$)**

Output	Three-statable
Active low	

This line controls the output enable line of bidirectional buffers on the multiplexed data/address lines. It is a three-stated during the "IDLE mode"

- **High Byte ($\overline{HIBYTE}$)**

Output	Three-statable
Active low	

This line is used when the operand size is byte in the single addressing mode. It is asserted when data is present on the upper eight bits of the data bus. It is used to control the output of bidirectional buffers which connects the upper eight bits of the data bus with the lower eight bits. It is three-stated during the "MPU mode" and the "IDLE mode"

- **Read/Write (R/$\overline{W}$)**

Input/Output	Three-statable
Active low (write)	
Active high (read)	

This line is an input in the "MPU mode" and an output during the "DMA mode". It is three-stated during the "IDLE mode". It is used to control the direction of data flow.

- **Upper Data Strobe ($\overline{UDS}$), Lower Data Strobe ($\overline{LDS}$)**

Input/Output	Three-statable
Active low	

These lines are extensions of the address lines indicating which byte or bytes of data of the addressed word are being addressed. These lines combined corresponds to address line A_0 in table 1.

- **Data Transfer Acknowledge ($\overline{DTACK}$)**

Input/Output	Three-statable
Active low	

In the "MPU mode", this line is an output indicating the completion of Read/Write bus cycle by the MPU.

In the "DMA mode", the DMAC monitors this line to determine when a data transfer has completed. In the event that a bus exception is requested, except for $\overline{HALT}$, prior to or concurrent with $\overline{DTACK}$, the $\overline{DTACK}$ response is ignored and the bus exception is honored. In the "IDLE mode", this signal is three-stated.

- **Bus Exception Controls ($\overline{BEC_0}$ through $\overline{BEC_2}$)**

Input
Active low

These lines provide an encoded signal input indicating an exceptional condition in the DMA bus cycle. See bus exception section for details.

- **Bus Request ($\overline{BR}$)**

Output
Active low

This output line is used to request ownership of the bus by the DMAC.

- **Bus Grant ($\overline{BG}$)**

Input
Active low

This line is used to indicate to the DMAC that it is to be the next bus master. The DMAC cannot assume bus ownership until both $\overline{AS}$ and $\overline{BGACK}$ becomes inactive. Once the DMAC acquires the bus, it does not continue to monitor the $\overline{BG}$ input.

- **Bus Grant Acknowledge ($\overline{BGACK}$)**

Input/Output	Three-statable
Active low	

Bus Grant Acknowledge ($\overline{BGACK}$) is a bidirectional control line. As an output, it is generated by the DMAC to indicate that it is the bus master.

As an input, $\overline{BGACK}$ is monitored by the DMAC, in limited rate auto-request mode, to determine whether or not the current bus master is a DMA device or not. $\overline{BGACK}$ is also monitored during bus arbitration in order to assume bus ownership.

- **Interrupt Request ($\overline{IRQ}$)**

Output	Open drain
Active low	

This line is used to request an interrupt to the MPU.

- **Interrupt Acknowledge ($\overline{IACK}$)**

Input
Active low

This line is an input to the DMAC indicating that the current bus cycle is an interrupt acknowledge cycle by the MPU. The

DMAC responds the interrupt vector of the channel with the highest priority requesting an interrupt. There are two kinds of the interrupt vectors for each channel: normal (NIV) or error (EIV). $\overline{IACK}$ is not serviced if the DMAC has not generated $\overline{IRQ}$.

- **Channel Request ($\overline{REQ_0}$ through $\overline{REQ_3}$)**

Input	
Active low or falling edge	

These lines are the DMA transfer request inputs from the peripheral devices.

These lines are falling edge sensitive inputs when the request mode is cycle steal. They are low-level sensitive when the request mode is burst.

- **Channel Acknowledge ($\overline{ACK_0}$ through $\overline{ACK_3}$)**

Output	
Active low	

These lines indicate to the I/O device requesting a transfer that the request is acknowledged and the transfer is to be performed. These lines may be used as a part of the enable circuit for bus interface to the peripheral.

- **Peripheral Control Line ($\overline{PCL_0}$ through $\overline{PCL_3}$)**

Input/Output	Three-statable
Active low	

The four lines ($\overline{PCL_0} \sim \overline{PCL_3}$) are multi-purpose lines which may be individually programmed to be a START output, an Enable Clock input, a READY input, an ABORT input, a STATUS input, or an INTERRUPT input.

- **Done ($\overline{DONE}$)**

Input/Output	Open Drain
Active low	

As an output, this line is asserted concurrently with the $\overline{ACK_X}$ timing to indicate the last data transfer to the peripheral device. As an input, it allows the peripheral device to request a normal termination of the DMA transfer.

- **Device Transfer Complete ($\overline{DTC}$)**

Output	Three-statable
Active low	

This line is asserted when the DMA bus cycle has terminated normally with no exceptions. It may be used to supply the data latch timing to the peripheral device. In this case, data is valid at the falling edge of $\overline{DTC}$.

■ INTERNAL ORGANIZATION

The DMAC has four independent DMA channels. Each channel has its own set of channel registers. These registers define and control the activity of the DMAC in processing a channel operation.

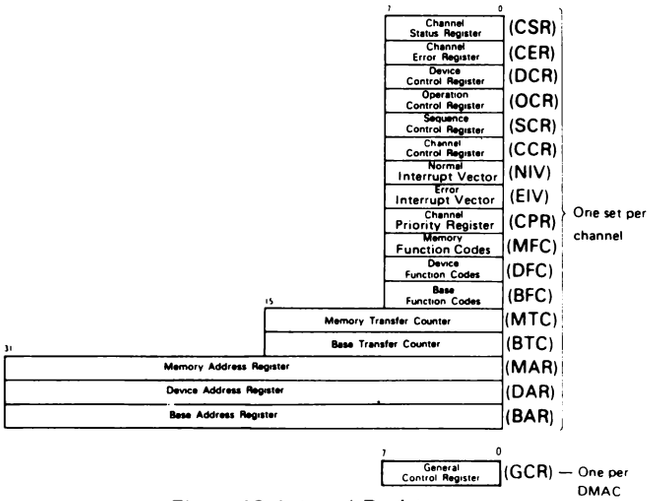


Figure 10 Internal Registers

- **Register Organization**

The internal register addresses are represented in Table 1. Address space not used within the address map is reserved for future expansion. A read from an unused location in the map results in a normal bus cycle with all ones for data. A write to one of these locations results in a normal bus cycle but no write occurs.

Unused bits of the defined registers in Table 1 read as zeros.

Table 1 Internal Register Addressing Assignments

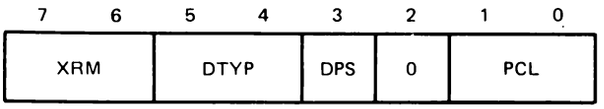
	Address Bits									
Register	7	6	5	4	3	2	1	0	Mode	
Channel Status Register	c	c	0	0	0	0	0	0	R	W*
Channel Error Register	c	c	0	0	0	0	0	1	R	
Device Control Register	c	c	0	0	0	1	0	0	R	W
Operation Control Register	c	c	0	0	0	1	0	1	R	W
Sequence Control Register	c	c	0	0	0	1	1	0	R	W
Channel Control Register	c	c	0	0	0	1	1	1	R	W
Memory Transfer Counter	c	c	0	0	1	0	1	b	R	W
Memory Address Register	c	c	0	0	1	1	s	s	R	W
Device Address Register	c	c	0	1	0	1	s	s	R	W
Base Transfer Counter	c	c	0	1	1	0	1	b	R	W
Base Address Register	c	c	0	1	1	1	s	s	R	W
Normal Interrupt Vector	c	c	1	0	0	1	0	1	R	W
Error Interrupt Vector	c	c	1	0	0	1	1	1	R	W
Channel Priority Register	c	c	1	0	1	1	0	1	R	W
Memory Function Codes	c	c	1	0	1	0	0	1	R	W
Device Function Codes	c	c	1	1	0	0	0	1	R	W
Base Function Codes	c	c	1	1	1	0	0	1	R	W
General Control Register	1	1	1	1	1	1	1	1	R	W

cc:00-Channel #0,01-Channel #1,
10-Channel #2,11-Channel #3,
ss:00-high-order, 01-upper middle,
10-lower middle,11-low-order
b: 0-high-order, 1-low-order
* see Channel Status Register Section

- **Device Control Register (DCR)**

The DCR is a device oriented control register. The XRM bits specifies whether the channel is in burst or cycle steal request mode. The DTYP bits define what type of device is on the channel. If the DTYP bits are programmed to be a HMCS6800 device, the PCL definition is ignored and the $\overline{PCL}$ line is an Enable clock input. If the DTYP bits are programmed to be a device with $\overline{READY}$, the PCL definition is ignored and the $\overline{PCL}$ line is a $\overline{READY}$ input. The DPS bit defines the port size (eight or sixteen bits) of the peripheral device. (A port size is the largest data which the peripheral device can transfer during a DMA bus cycle.) The PCL bits define the function of the $\overline{PCL}$ line. If the DTYP bits are programmed to be HMCS6800 device, or Device with $\overline{ACK}$ and $\overline{READY}$, these definitions are ignored. The XRM

bits are ignored if an auto-request mode (REQG = 00 or 01 in Operation Control Register) is selected.



XRM (EXTERNAL REQUEST MODE)
- 00 Burst Transfer Mode
- 01 (undefined, reserved)
- 10 Cycle Steal Mode without Hold
- 11 Cycle Steal Mode with Hold

DTYP (DEVICE TYPE)
- 00 HD68000 compatible device, explicitly addressed (dual addressing mode)
- 01 HD6800 compatible device, explicitly addressed (dual addressing mode)
- 10 Device with $\overline{ACK}$, implicitly addressed (single addressing mode)
- 11 Device with $\overline{ACK}$ and $\overline{READY}$, implicitly addressed (single addressing mode)

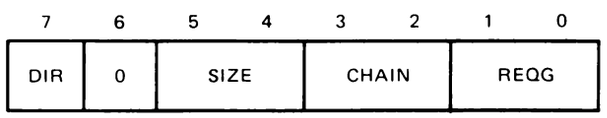
DPS (DEVICE PORT SIZE)
- 0 8 bit port
- 1 16 bit port

PCL (PERIPHERAL CONTROL LINE)
- 00 Status Input
- 01 Status Input with Interrupt
- 10 Start Pulse
- 11 Abort Input

Bit 2 Not Used

- **Operation Control Register (OCR)**

The OCR is an operation control register. The DIR bit defines the direction of the transfer. The SIZE bits define the size of the operand. The CHAIN bits define the type of the CHAIN mode. The REQG bits define how requests for transfers are generated.



DIR (DIRECTION)
- 0 Transfer from memory to device (transfer from MAR address to DAR address)
- 1 Transfer from device to memory (transfer from DAR address to MAR address)

SIZE (OPERAND SIZE)
- 00 Byte (8 bits)
- 01 Word (16 bits)
- 10 Long Word (32 bits)
- 11 (undefined, reserved)

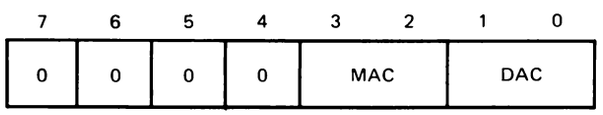
CHAIN (CHAINING OPERATION)
- 00 Chain operation is disabled
- 01 (undefined, reserved)
- 10 Array Chaining
- 11 Linked Array Chaining

REQG (DMA REQUEST GENERATION METHOD)
- 00 Auto-request at transfer rate limited by General Control Register (Limited Rate Auto-Request)
- 01 Auto-request at maximum rate
- 10 $\overline{REQ}$ line requests an operand transfer
- 11 Auto-request the first operand, external request for subsequent operands

Bit 6 Not Used

- **Sequence Control Register (SCR)**

The SCR is used to define the sequencing of memory and device addresses.



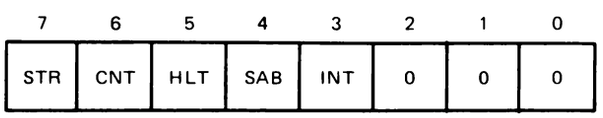
MAC (MEMORY ADDRESS COUNT)
- 00 Memory address register does not count
- 01 Memory address register counts up
- 10 Memory address register counts down
- 11 (undefined, reserved)

DAC (DEVICE ADDRESS COUNT)
- 00 Device address register does not count
- 01 Device address register counts up
- 10 Device address register counts down
- 11 (undefined, reserved)

Bits 7, 6, 5, 4 Not Used

- **Channel Control Register (CCR)**

The CCR is used to start or terminate the operation of a channel. This register also determines if an interrupt request is to be generated. Setting the STR bit causes immediate activation of the channel; the channel will be ready to accept request immediately. The STR and CNT bits of the register cannot be reset by a write to the register. The SAB bit is used to terminate the operation forcedly. Setting the SAB bit will reset STR and CNT. Setting the HLT bit will halt the channel operation, and clearing the HLT bit will resume the operation. Setting start bit must be done by byte access. Otherwise, timing error occurs.



STR (START OPERATION)
- 0 No operation is pending
- 1 Start operation

CNT (CONTINUE OPERATION)
- 0 No continuation is pending
- 1 Continue operation

HLT (HALT OPERATION)
- 0 Operation not halted
- 1 Operation halted

SAB (SOFTWARE ABORT)
- 0 Channel operation not aborted
- 1 Abort channel operation

INT (INTERRUPT ENABLE)
- 0 No interrupts enabled
- 1 Interrupts enabled

Bits 2, 1, 0 Not Used

- **Channel Status Register (CSR)**

The CSR is a register containing the status of the channel.

7	6	5	4	3	2	1	0
COC	BTC	NDT	ERR	ACT	0	PCT	PCS

COC (CHANNEL OPERATION COMPLETE)
0 Channel operation incomplete
1 Channel operation complete
BTC (BLOCK TRANSFER COMPLETE)
0 Block transfer incomplete
1 Block transfer complete
NDT (NORMAL DEVICE TERMINATION)
0 No normal device termination by $\overline{\text{DONE}}$ input
1 Device terminated operation normally by $\overline{\text{DONE}}$ input
ERR (ERROR BIT)
0 No errors
1 Error as coded in CER
ACT (CHANNEL ACTIVE)
0 Channel not active
1 Channel active
PCT ($\overline{\text{PCL}}$ TRANSITION)
0 No $\overline{\text{PCL}}$ transition occurred
1 $\overline{\text{PCL}}$ transition occurred
PCS (THE STATE OF THE $\overline{\text{PCL}}$ INPUT LINE)
0 $\overline{\text{PCL}}$ "Low"
1 $\overline{\text{PCL}}$ "High"
Bit 2 Not Used

- **Channel Error Register (CER)**

The CER is an error condition status register. The ERR bit of CSR indicates if there is an error or not. Bits 0-4 indicate what type of error occurred.

7	6	5	4	3	2	1	0
0	0	0	ERROR CODE				

Error Code
00000 No error
00001 Configuration error
00010 Operation timing error
00101 Address error in MAR
00110 Address error in DAR
00111 Address error in BAR
01001 Bus error in MAR
01010 Bus error in DAR
01011 Bus error in BAR
01101 Count error in MTC
01111 Count error in BTC
10000 External abort
10001 Software abort
Bits 7, 6, 5 Not Used

- **Channel Priority Register (CPR)**

The CPR is used to define the priority level of the channel. Priority level 0 is the highest and priority level 3 is the lowest priority.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	CP	

CP (CHANNEL PRIORITY)
00 Priority level 0
01 Priority level 1
10 Priority level 2
11 Priority level 3
Bit 7 through 2 Not Used

- **General Control Register (GCR)**

The GCR is used to define what portion of the bus cycles is available to the DMAC for limited rate auto-request generation. GCR is also used to specify the hold time for cycle steal mode with hold.

7	6	5	4	3	2	1	0
0	0	0	0	BT		BR	

BT (BURST TIME)
The number of DMA clock cycles per burst that the DMAC allows in the auto-request at a limited rate of transfer is controlled by these two bits. The number is 2(BT+4) (two to the BT+4 power).
BR (BANDWIDTH RATIO)
The amount of the bandwidth utilized by the auto-request at a limited rate transfer is controlled by these two bits. The ratio is 2(BR+1) (two to the BR+1 power).
The hold time for cycle steal mode with hold is defined to be minimum of 1 sample interval and maximum of 2 sample intervals. A sample interval is defined to be 2(BT+BR+5) (two to the BT+BR+5 power) clock cycles.

Bits 7 through 4 Not Used

- **Address Registers (MAR, DAR, BAR)**

Three 32-bit registers are utilized to implement the Memory Address Register, Device Address Register, and the Base Address Register. Only the least significant twenty-four bits are connected to the address output pins. The content of the MAR is outputted when the memory is accessed in single or dual adressing mode. The content of the DAR is outputted when the peripheral device is accessed. The contents of the BAR is outputted when reading chain information from memory in the Array Chaining Mode or the Linked Array Chaining Mode. It is also used to set the top address of the next block transfer in Continue mode.

- **Function Code Registers (MFC, DFC, BFC)**

The DMAC has three function code registers per channel: the Memory Function Code Register (MFC), Device Function Code Register (DFC), and the Base Function Code Register (BFC). The contents of these registers are outputted from FC_0 through FC_2 lines when an address is outputted from MAR, DAR, or BAR, respectively. The BFC is also used to set the MFC for the transfer of the next data block in the Continue mode.

7	6	5	4	3	2	1	0
0	0	0	0	0	FC2	FC1	FC0

Bits 3 through 7 Not Used

- **Transfer Count Registers (MTC, BTC)**

Each channel has two 16-bit counters: the Memory Transfer Counter (MTC) and the Base Transfer Counter (BTC). The MTC

counts the number of transfer words in one block, and is decreased by one for every operand transfer.

The BTC is used to count the number of data blocks in the Array Chaining Mode. BTC is also used to set the number of operands to transfer for the next data block in the Continue Mode.

- **Interrupt Vector Registers (NIV, EIV)**

Each channel has a Normal Interrupt Vector register and an Error Interrupt Vector register.

When an interrupt acknowledge cycle occurs, an interrupt vector is outputted from one of those registers. If the error bit (CSR) is set for the channel with interrupt pending, then content of EIV is outputted, otherwise content of NIV is outputted.

■ OPERATION DESCRIPTION

A DMAC channel operation proceeds in three principal phases. During the initialization phase, the MPU sets the channel control registers, supply the initial address and the number of transfer words, and starts the channel. During the transfer phase, the DMAC accepts requests for data operand transfers, and provides addressing and bus controls for the transfers. The termination phase occurs after the operation is completed.

This section describes DMAC operations. A description of the MPU/DMAC communication is given first. Next, the transfer phase is covered, including how the DMAC recognizes requests and how the DMAC arranges for data transfer. Following this, the initialization phase is described. The termination phase is covered, introducing chaining, error signaling, and bus exceptions. A description of the channel priority scheme rounds out the section.

- **Read/Write of the DMAC Registers by MPU**

The MPU reads and writes the DMAC internal registers and controls the DMA transfer. Figure 11 indicates the timing diagram when the MPU reads the contents of the DMAC register. The MPU outputs A_1-A_{23}, FC_0-FC_2, $\overline{AS}$, R/$\overline{W}$, $\overline{UDS}$, and $\overline{LDS}$, and accesses the DMAC internal register. The specific internal register is selected by A_1-A_7, $\overline{LDS}$ and $\overline{UDS}$. The $\overline{CS}$ and $\overline{IACK}$ lines are generated by the external circuit with A_8-A_{23} and FC_0-FC_2. The DMAC outputs data on the data bus, together with $\overline{DDIR}$, $\overline{DBEN}$ and $\overline{DTACK}$. The $\overline{DDIR}$ and $\overline{DBEN}$ control the bidirectional buffer on the bus and the $\overline{DTACK}$ indicates that the data has been sent or received by the DMAC. Read Cycle is eighteen CLKs. Figure 12 shows the MPU write cycle. Write cycle is fifteen CLKs.

Note the following points.

(1) The clock reference shown in this figure is the DMAC input clock.
(2) The $\overline{DDIR}$ and the $\overline{DBEN}$ are three-stated at the beginning which detects $\overline{CS}$ and the ending of the cycle.
(3) During the MPU read cycle, the $\overline{DTACK}$ is asserted after the data is valid on the system bus.
(4) During the MPU write cycle, the $\overline{DDIR}$ line will be driven low to direct the data buffers toward to DMAC before the buffers are enabled.
(5) During the MPU write cycle, the DMAC will latch the data before asserting $\overline{DTACK}$. Then it will negate $\overline{DBEN}$ and $\overline{DDIR}$ in the proper order.
(6) After the MPU cycle and the $\overline{LDS}$ and the $\overline{UDS}$ are negated by the MPU, the DMAC will put $\overline{DBEN}$, $\overline{DDIR}$ and the address data lines to a high impedance state.
(7) $\overline{DTACK}$ will once go "High" and then to a high impedance state after negating $\overline{LDS}$ and $\overline{UDS}$.

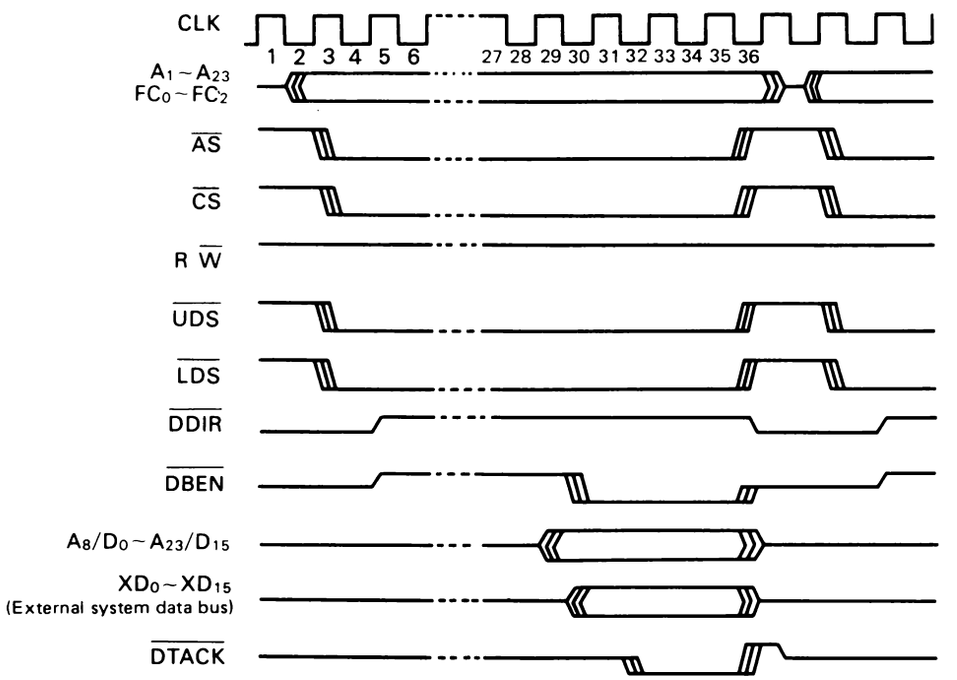


Figure 11 MPU Read from DMAC – Word

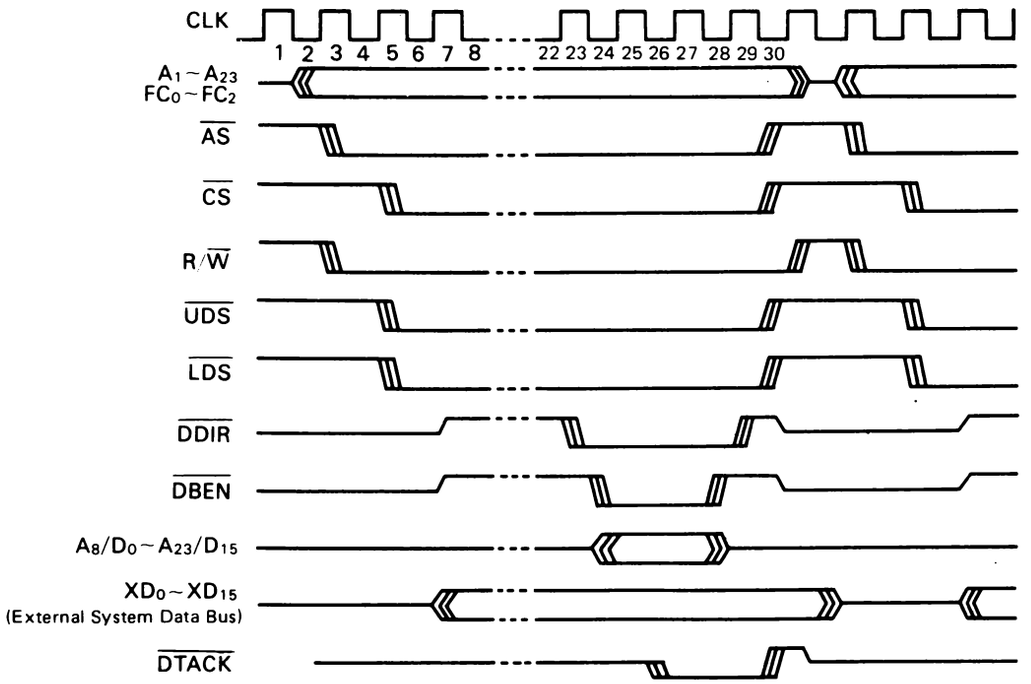
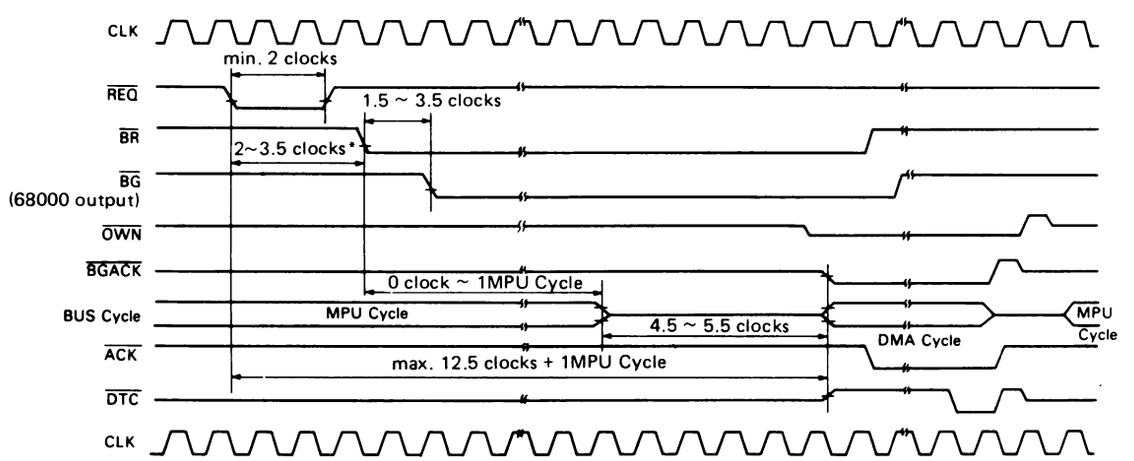


Figure 12 MPU Write to DMAC – Word

- **Bus Arbitration**

The followings are the description of the bus arbitration. The DMAC must obtain the ownership of the bus in order to transfer data. Figure 13 indicates the DMAC bus arbitration timing. It is completely compatible with that of HD68000 MPU. The DMAC asserts the Bus Grant ($\overline{BG}$) to request the bus mastership. The MPU recognizes the request and asserts $\overline{BG}$, then it grants the ownership in the next bus cycle. After the end of the current cycle ($\overline{AS}$ is negated), the MPU relinquishes the bus to the DMAC. The DMAC asserts the bus grant acknowledge ($\overline{BGACK}$) to indicate that it has the bus ownership. A half clock before $\overline{BGACK}$ is asserted, the DMAC asserts $\overline{OWN}$. $\overline{OWN}$ is kept asserted for a half clock after $\overline{BGACK}$ is negated at the end of the DMA cycle. $\overline{BR}$ is negated one clock after $\overline{BGACK}$ is asserted.



* This case assumes that no exception condition exists and DMAC isn't accessed by MPU.

Figure 13 DMAC Bus Arbitration Timing

- **Device/DMAC Communication**

Communication between peripheral devices and the DMAC is accomodated by five signal lines. Each channel has $\overline{REQ}$, $\overline{ACK}$ and $\overline{PCL}$, and the last two lines the $\overline{DONE}$ and $\overline{DTC}$ lines, are shared among the four channels.

(1) Request ($\overline{REQ}$)

The peripheral devices assert $\overline{REQ}$ to request data transfers. See the "Requests" section for details.

(2) Acknowledge ($\overline{ACK}$)

This line is used to implicitly address the device which is transferring the data (This device is not selected by address lines.) It is also asserted when the content of DAR is outputted during memory-to-memory transfer except for the auto-request mode at a limited rate or at the maximum rate.

(3) Peripheral Control Line ($\overline{PCL}$)

The function of this line is quite flexible and is determined by the DCR (Device Control Register).

The DTYP bits of the DCR define what type of device is on the channel. If the DTYP bits are programmed to be a HMCS6800 device, the PCL definition is ignored and the $\overline{PCL}$ line is an Enable clock (E clock) input. If the DTYP bits are programmed to be a device with $\overline{READY}$, the PCL definition as ignored and the $\overline{PCL}$ line is a ready input.

$\overline{PCL}$ As a Status Input

The $\overline{PCL}$ line may be programmed as a status input. The status level of this line can be determined by the PCS bit in the CSR, regardless of the PCL function determined by the DCR. If a negative transition occurs and remains stable for a minimum of two clocks, the PCT bit of the CSR is set. This PCT bit is cleared by resetting the DMAC or the writing "1" to the PCT bit.

$\overline{PCL}$ As an Interrupt

The $\overline{PCL}$ line may be programmed to generate an interrupt on a negative transition. This enables an interrupt which is requested if the PCT bit of the CSR is set. When using this function, it is necessary to reset the PCT bit in the CSR before the PCL bit in the DCR is set to interrupt, in order to avoid assertion of IRQ line at this time.

$\overline{PCL}$ As a Starting Pulse

The $\overline{PCL}$ line may be programmed to output a starting pulse. This active low starting pulse is outputted when a channel is activated, and is "Low" for a period of four clock cycles.

$\overline{PCL}$ As an Abort Input

The $\overline{PCL}$ line may be programmed to be a negative transition above input which terminates an operation by setting the external abort error in CER. It is necessary to reset the PCT bit in the CSR before activating the channel (Setting the ACT bit of CCR) so that the channel operation is not immediately aborted.

$\overline{PCL}$ As an Enable Clock (E Clock) Input

If the DTYP bits are programmed to be a HMCS6800 device, the PCL definition is ignored and the $\overline{PCL}$ line is an Enable Clock input. The Enable clock downtime must be as long as five clock cycles, and must be high for a minimum of three DMAC clock cycles, but need not be synchronous with the DMAC's clock.

$\overline{PCL}$ As a $\overline{READY}$ Input

If the DTYP bits are programmed to be a device with $\overline{READY}$, the PCL definition is ignored and the $\overline{PCL}$ line is a $\overline{READY}$ input. The $\overline{READY}$ is an active low input.

(4) DONE ($\overline{DONE}$)

This line is an active low Input/Output signal with an open drain. It is asserted when the memory transfer count is exhausted in a single block transfer. In the chaining operation, $\overline{DONE}$ is asserted only at the last transfer to the peripheral device of the last data block. In the continue mode, $\overline{DONE}$ is asserted for each data block. It is asserted and negated in coincident with the $\overline{ACK}$ line for the last data transfer to the peripheral device. It is also outputted in coincident with the $\overline{ACK}$ line of the last bus cycle, in which the address is outputted from the DAR, in the memory-to-memory transfer (dual addressing mode) that uses the $\overline{ACK}$ line.

The DMAC also monitors the state of the $\overline{DONE}$ line during the DMA bus cycle. If the device asserts $\overline{DONE}$ during $\overline{ACK}$ active, the DMAC will terminate the operation after the transfer of the current operand. If $\overline{DONE}$ is asserted on the first byte of 2 byte operation or the first word of long word operation, the DMAC does not terminate the operation before the whole operand transfer is completed. If $\overline{DONE}$ is asserted, then the DMAC terminates the operation by clearing the ACT bit of the CSR, and setting the COC and $\overline{NDT}$ bits of the CSR. If both the DMAC and the device assert $\overline{DONE}$, the device termination is not recognized, but the channel operation does terminate. $\overline{DONE}$ is outputted again for the retry exceptions bus cycles.

(5) Data Transfer Complete ($\overline{DTC}$)

$\overline{DTC}$ is an active low signal which is asserted when the actual data transfer is accomplished. It is also asserted in the bus cycle which read a chain information from memory in the Chaining mode. However, if exceptions are generated and the DMA bus cycle terminates, $\overline{DTC}$ is not asserted. $\overline{DTC}$ is asserted one half clock before $\overline{LDS}$ and $\overline{UDS}$ are negated, and negated one half clock after $\overline{LDS}$ and $\overline{UDS}$ are negated.

- **Requests**

Requests may be externally generated by circuitry in the peripheral device, or internally generated by the auto-request mechanism. The REQG bits of the OCR determine these modes. The DMAC also supports an operation in which the DMAC auto-requests the first transfer and then wait for the peripheral device to request the following transfers.

(1) Auto-request Transfers

The auto-request mechanism provides generation of requests within the DMAC. These requests can be generated at either of two rates: maximum-rate and limited-rate. In the former case, the channel always has a request pending.

The limited rate auto-request functions by monitoring the bus utilization.

Limited-rate Auto-request

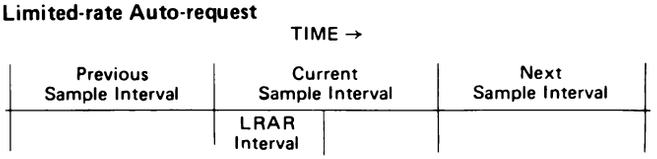


Figure 14 DMAC Sample Intervals

In the limited-rate auto-request the DMAC devides time into equal length sample intervals by counting clock cycles. The end of one sample interval makes the beginning of the next. During a sample interval, the DMAC monitors by means of $\overline{BGACK}$ pin the system bus activity of the DMAC and other bus master devices. At the end of the sample interval, decision is made whether or not to perform the channel's data transfer during the next sample interval. Namely, based on the activity of the DMAC or other bus master devices during the current sample interval, the DMAC allows limited-rate auto-requests for some initial portion of the next sample interval.

The length of the sample interval, and the portion of the sample interval during which limited-rate auto-requests can be

made (the limited-rate auto-request interval) are controlled by the BT and BR bits in the GCR. The length in clock cycles of the limited-rate auto-request interval is $2^{(BT+4)}$ (2 raised to the BT+4 power). For example, if BT equals 2 and the DMA utilization of the bus was low during the previous sample interval, then the DMAC generates the auto-request transfers during the first 64 clock cycles.

The ratio of the length of the sample interval to the length of the limited-rate auto-request interval is controlled by the BR bits. The ratio of the system bus utilization of the MPU to other bus master devices including he DMAC is $2^{(BR+1)}$ (2 raised to the BR+1 power). If the fraction of DMA clock cycles during the sample interval exceeds the programmed utilization level, the DMAC will not allow limited-rate auto-requests during the next sample interval.

For example, if BR equals 3, then at most one out of 16 clock cycles during a sample interval can be used by the DMAC and other bus master devices, and still the DMAC would allow limited rate auto-request during the next sample interval. Therefore, from the viewpoint of long period, the ratio of the system bus utilization of the MPU to I/O devices including the DMAC is about 16:1. The sample interval length is not a direct parameter, but it is equal to $2^{(BT+BR+5)}$ clock cycles. Thus, the sample interval can be programmed between 32 and 2048 clock cycles.

The DMAC uses the $\overline{BGACK}$ to differentiate between the MPU bus cycle and DMAC or other bus master devices. If $\overline{BGACK}$ is active, then the DMAC assumes that the bus is used by a DMAC or other bus master devices. If it is inactive, then the DMAC assumes that it is used by the MPU.

Maximum-rate Auto-request

If the REQG bits in the OCR indicate auto-request at the maximum rate, the DMAC acquires the bus after the start bit is set and keeps it until the data transfer is completed.

If a request is made by another channel of higher priority, the DMAC services that channel and then resumes the auto-request sequence. If two or more channels are set to equal priority level and maximum rate auto-request, then the channels will rotate in a "round robbin" fashion.

If the HMCS68000 compatible device is connected to a channel, the $\overline{ACK}$ line is held inactive during an auto-request operation. Consequently, any channel may be used for the memory-to-memory transfer with the auto-request function in addition to the operation of data transfer between memory and peripheral device with using the $\overline{REQ}$ pin. Refer to Figure 15 for the timing of the memory-to-memory transfer. In this mode, the $\overline{ACK}$, $\overline{HIBYTE}$ and $\overline{DONE}$ outputs are always inactive.

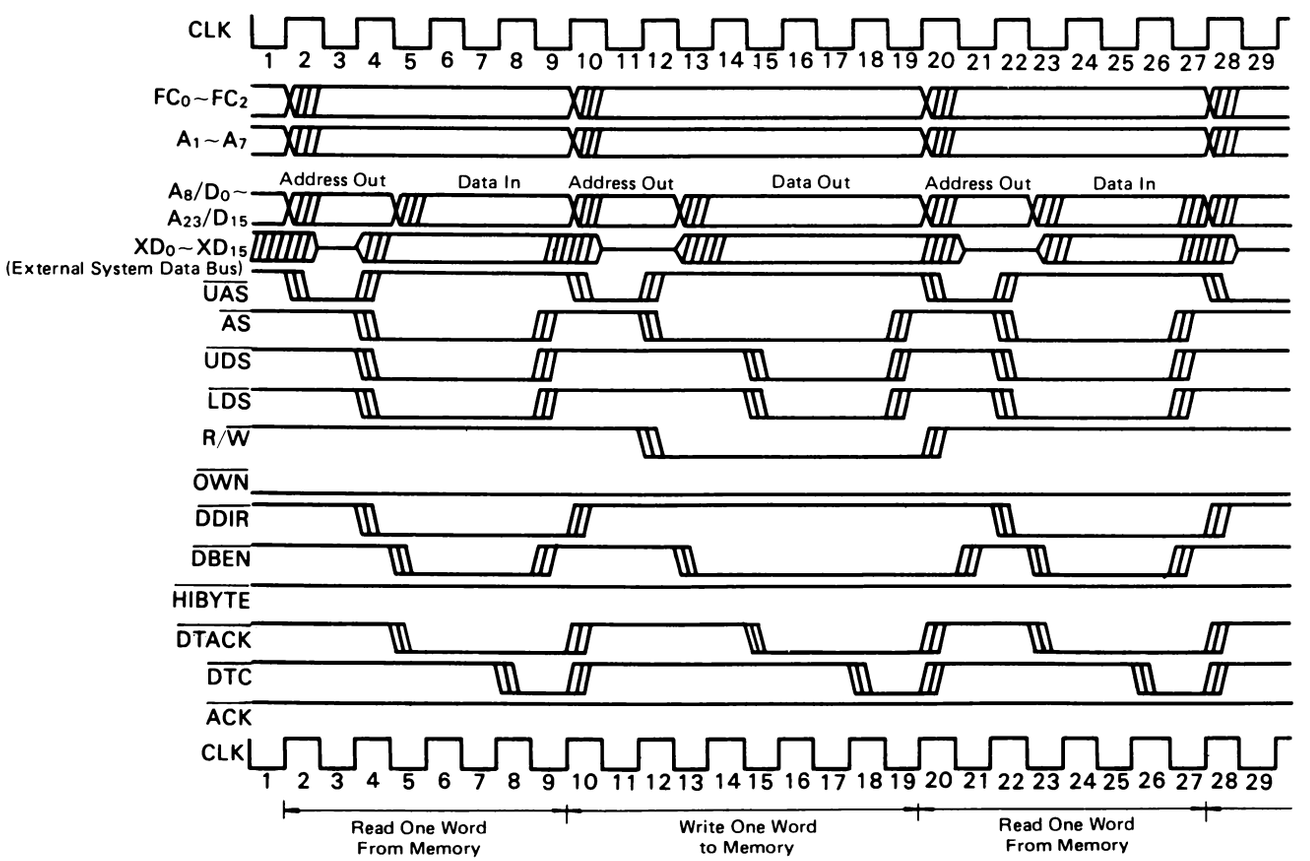


Figure 15 Memory-to-Memory Transfer Read-Write-Read Cycles

(2) External Requests

If the REQG bits of the OCR indicate that the $\overline{REQ}$ line generates requests, the transfer requests are generated externally. The request line associated with each channel allows the device to externally generate requests for DMA transfers. When the device wants an operand transferred, it makes a request by asserting the request line. The external request mode is determined by the XRM bits of the DCR, which allows both burst and cycle steal request modes. The burst request mode allows a channel to request the transfer of multiple operands using consecutive bus cycles. The cycle steal request mode allows a channel to request the transfer of a single operand. The followings are the description of the burst and the cycle steal modes.

Burst Request Recognition

In the burst request mode, the $\overline{REQ}$ line is an active low input. The level sampled at the rising edge of the clock. Once the burst request is asserted, it needs to be held low until the first DMA bus cycle starts in order to insure at least one data transfer operation. In order to stop the burst mode transfer after the current bus cycle, the $\overline{REQ}$ line has to be negated one clock before the $\overline{DTC}$ output clock of this cycle. Refer to Figure 16 or the burst mode timing.

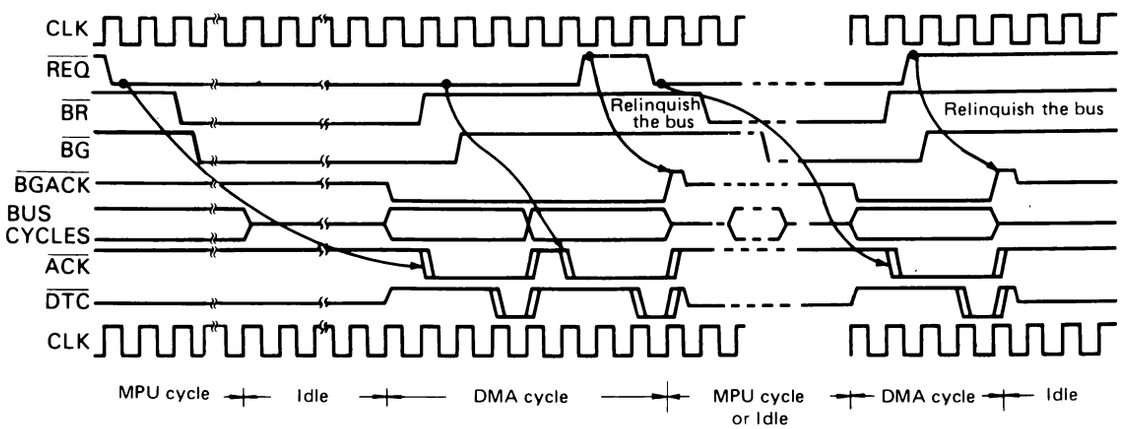


Figure 16 Burst Mode Request Timing
(Only one channel is active)

Cycle Steal Request Recognition

In the cycle steal request mode, the peripheral device requests the DMA transfer by generating an falling edge at the $\overline{REQ}$ line. The $\overline{REQ}$ line needs to be held "low" for at least 2 clock cycles. In the cycle steal mode, if the $\overline{REQ}$ line changes from "High" to "Low" between $\overline{ACK}$ output and one clock before the clock that outputs $\overline{DTC}$, then the next DMA transfer is performed without relinquishing the bus. If the bus is not relinquished, then maximum of 5 idle clocks is inserted between bus cycles. Refer to Figure 17 for the request timing of the cycle steal mode. If the XRM bits specify cycle steal without hold, the DMAC will relinquish the bus. If the XRM bits specify cycle steal with hold, the DMAC will retain ownership. The bus is not given up for arbitration until the channel operation terminates or until the device pauses. The device is determined to have paused if it does not make any requests during the next full sample interval. The sample interval counter is free running and is not reset or modified by this mode of operation. The sample interval counter is the same counter that is used for Limited Rate Auto Request and is programmed via the GCR. Figure 18 shows the request timing in the cycle steal bus hold. If the $\overline{REQ}$ is inputted during the hold time, the $\overline{ACK}$ is outputted after a maximum of 7.5 clock cycles from the picked-up clock. On the cycle steal with hold mode, the DMAC will hold the bus even when the transfer count is exhausted and the last data has been transferred. If DMA transfer is requested from other channels during this period, they are executed normally.

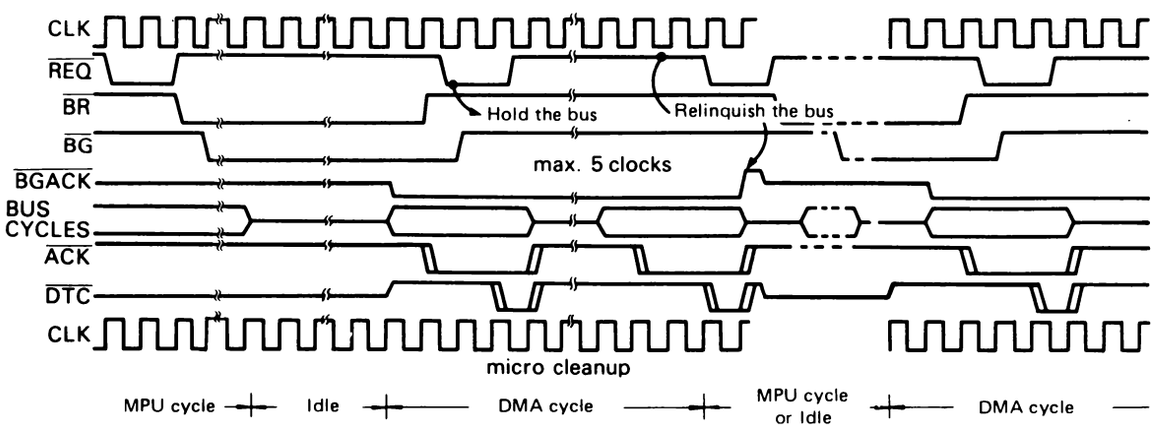


Figure 17 Cycle Steal Mode Request Timing

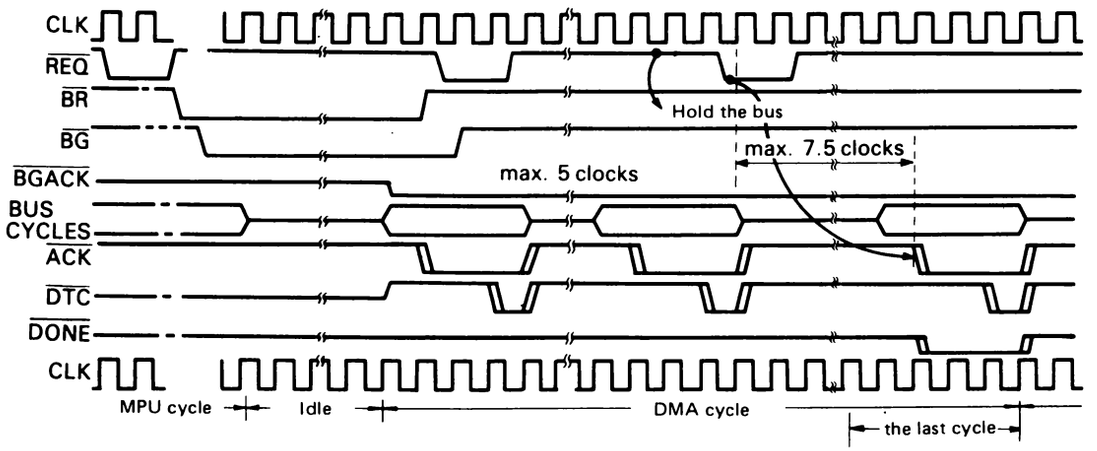


Figure 18 Cycle Steal Bus Hold Mode Request Timing

Request Recognition in Dual-address Transfers

In a following section dual-address transfers are defined. Dual address transfer is an exception to the request recognition rules in the previous paragraphs. In the cycle steal request mode, when there are two or more than transfers between the DMAC and the peripheral device during one operand transfer, the request is not recognized until the last transfer between the DMAC and the I/O device starts.

(3) Mixed Request Generation

A single channel can mix the two request generation methods. By programming the REQG bits of the OCR to "11", when the channel is started, the DMAC auto-requests the first transfer. Subsequent requests are then generated externally by the device. The $\overline{ACK}$ and $\overline{PCL}$ lines perform their normal functions in this operation.

- **Data Transfers**

All DMAC data transfers are assumed to be between memory and the peripheral device. The word "memory" means a 16-bit HMCS68000 bus compatible device. By programming the DCR, the characteristics of the peripheral device may be assigned. Each channel can communicate using any of the following protocols.

DTYP	Device Type	
00	HMCS68000 compatible device	Dual Addressing
01	HMCS6800 compatible device	
10	Device with $\overline{ACK}$	Single Addressing
11	Device with $\overline{ACK}$ and $\overline{READY}$	

(1) Dual Addressing

HMCS68000 and HMCS6800 compatible devices may be explicitly addressed. This means that before the peripheral transfers data, a data register within the device must be addressed. Because the address bus is used to address the peripheral, the data cannot be directly transferred to/from the memory because the memory also requires addressing. Instead, the data is transferred from the source to the DMAC and held in an internal DMAC holding register. A second bus transfer between the DMAC and the destination is then required to complete the operation. Because both the source and destination of the transfer are explicitly addressed, this protocol is called dual-addressed.

HMCS68000 Compatible Device Transfers

In this operation, when a request is received, the bus is obtained and the transfer is completed using the protocol as shown in Figures 19 and 20. Figures 21 through 24 show the transfer timings. Figure 21 and 24 show the operation when the memory is the source and the peripheral device is the destination. Figures 22 and 23 show the transfer in the opposite direction. The peripheral device is a 16-bit device in Figures 21 and 22, and a 8-bit device in Figures 23 and 24.

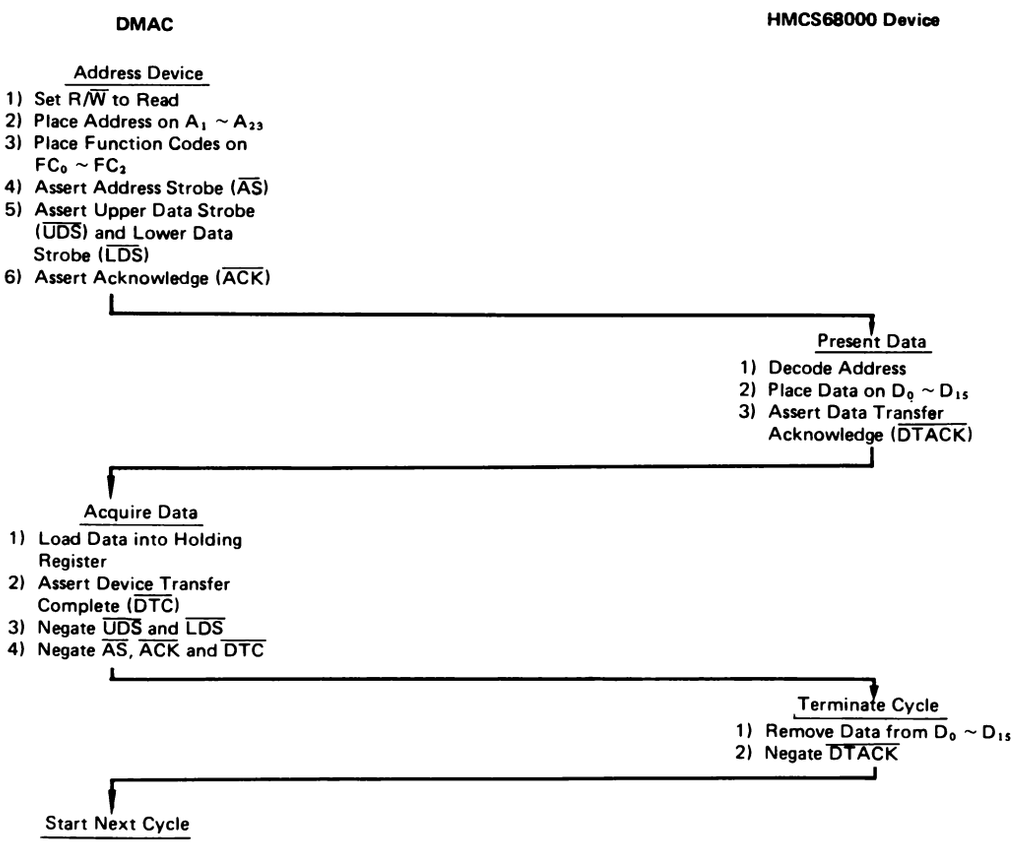


Figure 19 Word Read Cycle Flowchart HMCS68000 Type Device

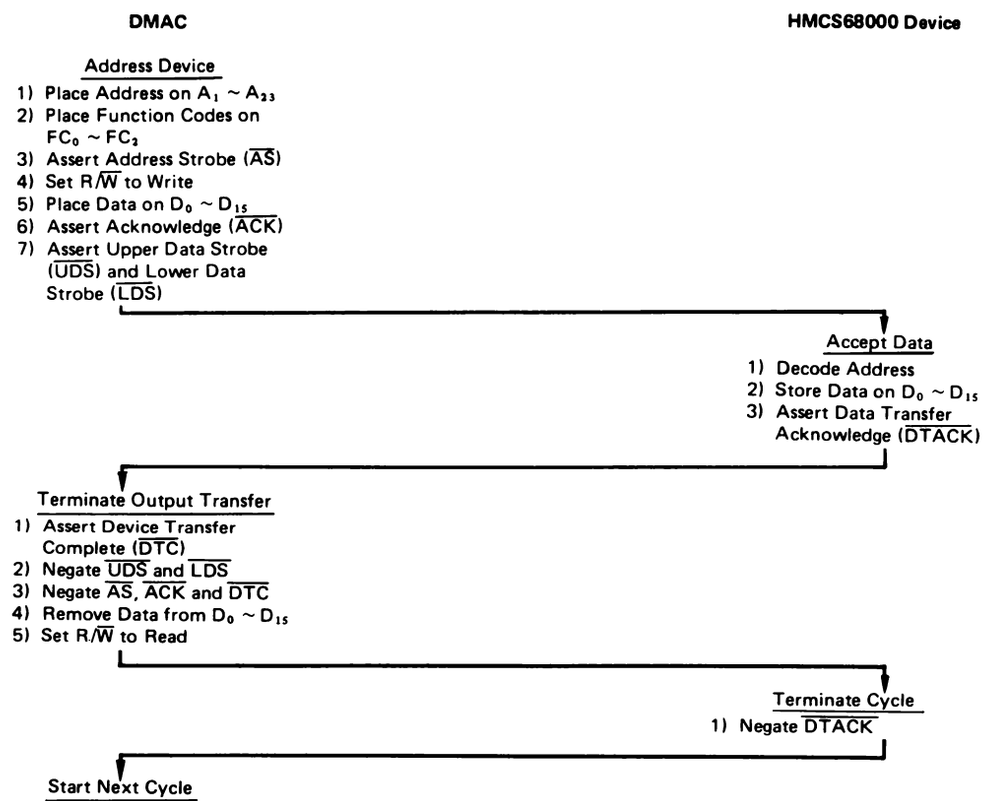


Figure 20 Word Write Cycle Flowchart HMCS68000 Type Device

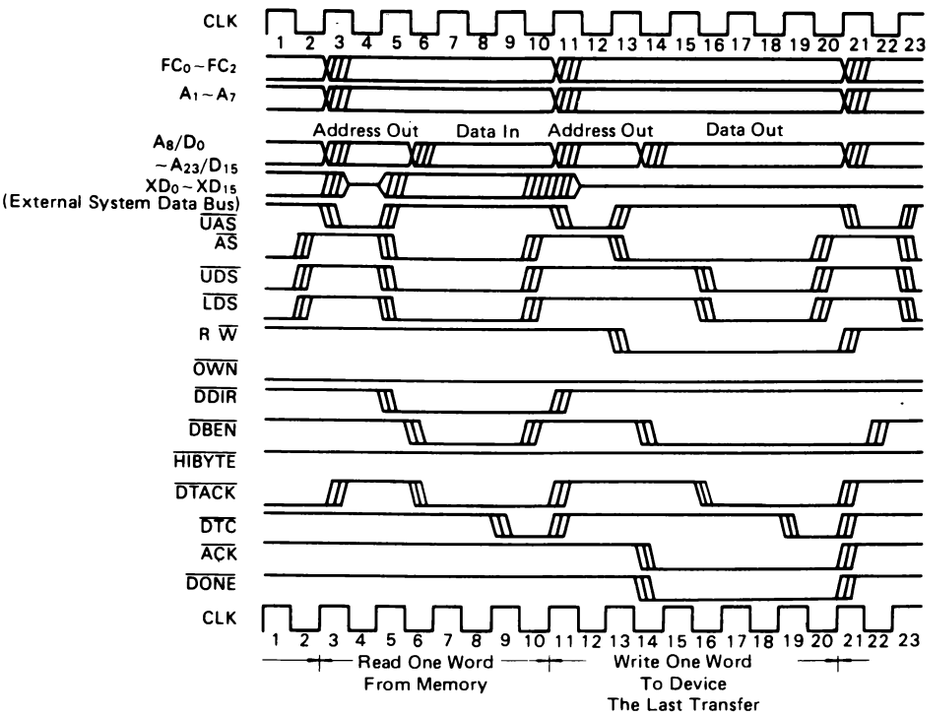


Figure 21 Dual Addressing Mode, Read/Write Cycle, Destination = 16-bit Device, Word Operand

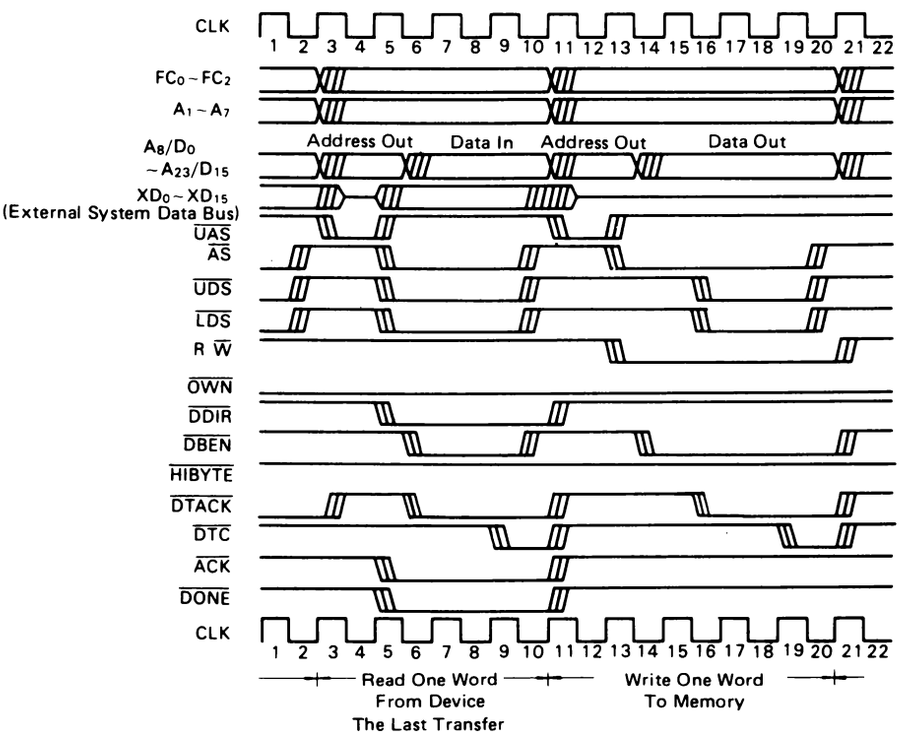


Figure 22 Dual Addressing Mode, Read/Write Cycle, Source = 16-bit Device, Word Operand

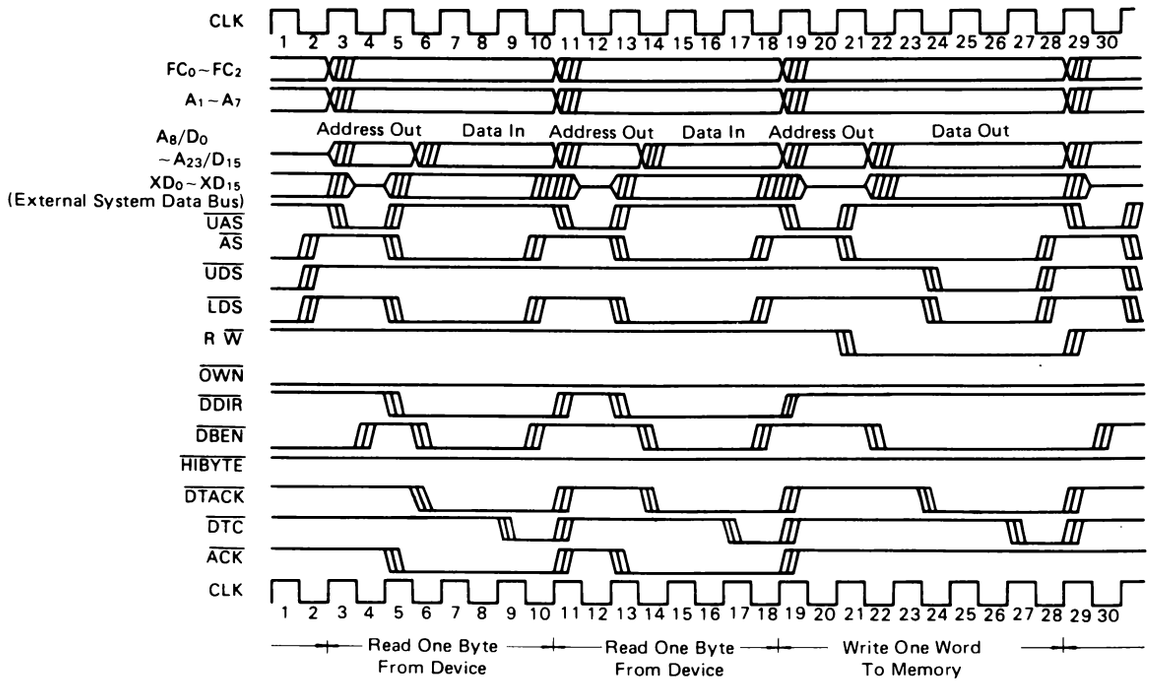


Figure 23 Dual Addressing Mode, Read/Write Cycle
Source = 8-bit Device, Word Operand

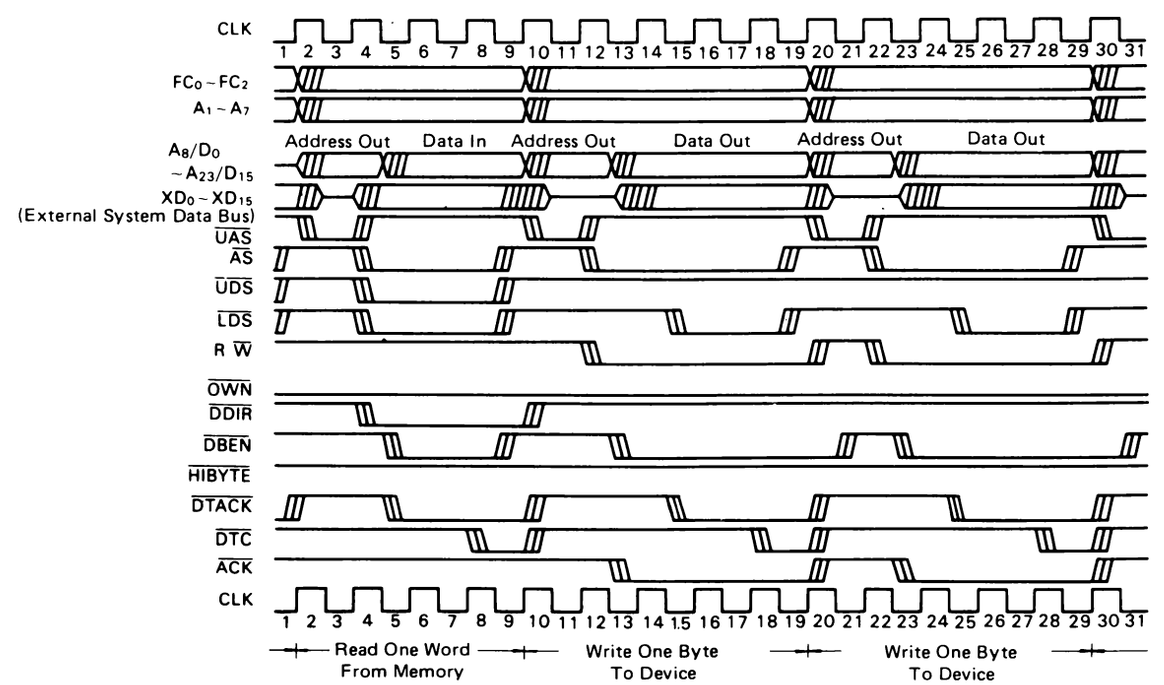


Figure 24 Dual Addressing Mode, Read/Write Cycle,
Destination = 8-bit Device, Word Operand

HMCS6800 Compatible Device Transfers

When a channel is programmed to perform HMCS6800 compatible transfers, the $\overline{PCL}$ line for that channel is defined as an Enable clock input. The DMAC performs data transfers between itself and the peripheral device using the HMCS6800 bus protocol, with the $\overline{ACK}$ output providing the $\overline{VMA}$ (valid memory address) signal. Figure 25 illustrates this protocol. Refer to Figure 26 for the read cycle timing and Figure 27 for the write cycle timing. In Figure 26, the DMAC latches the data at the falling edge of clock 19, so a latch to hold the data is necessary as shown in Figure 47.

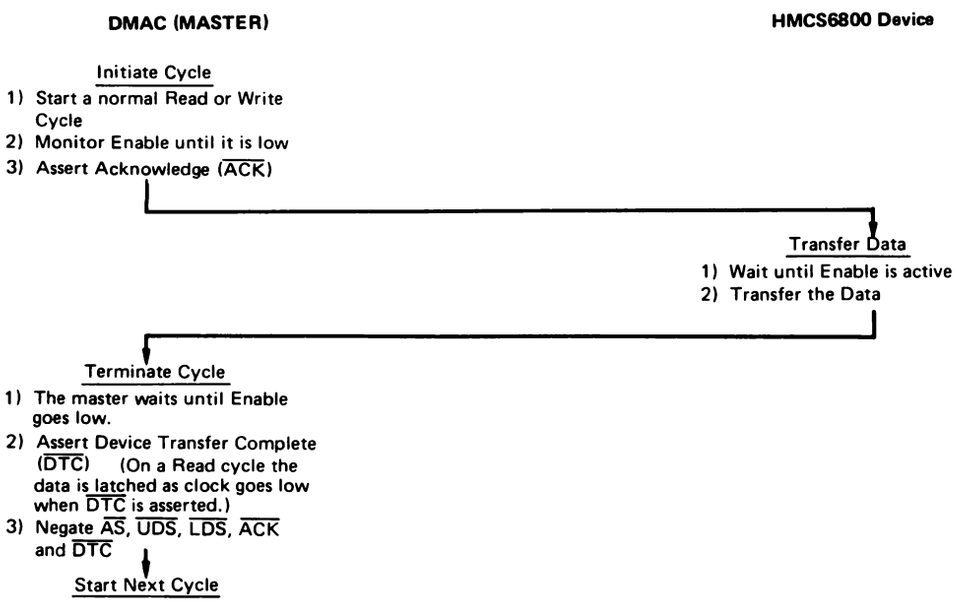


Figure 25 HMCS6800 Cycle Flowchart

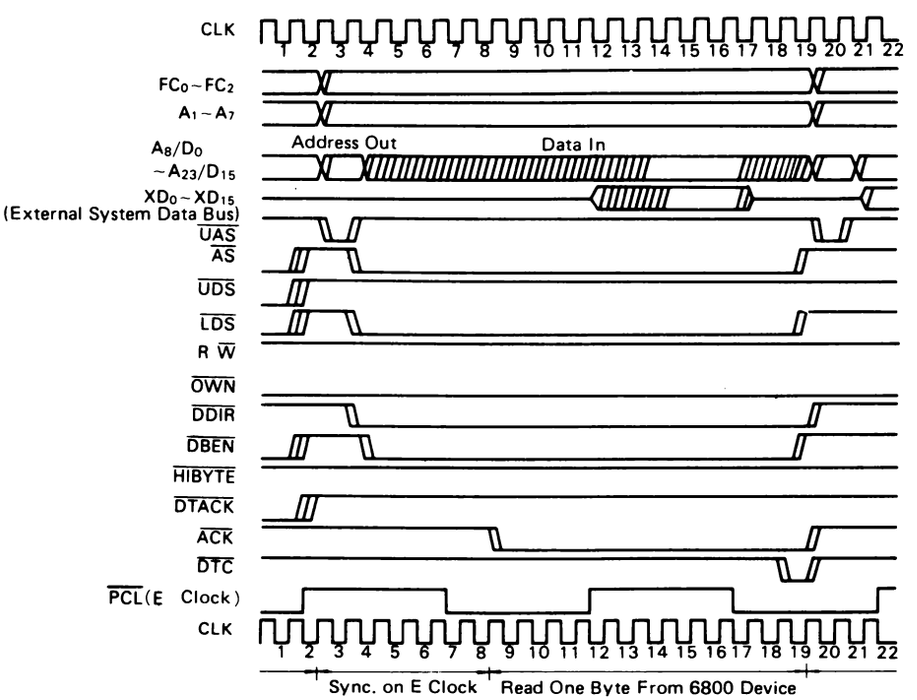


Figure 26 Dual Addressing Mode, HMCS6800 Compatible Device, Read Cycle

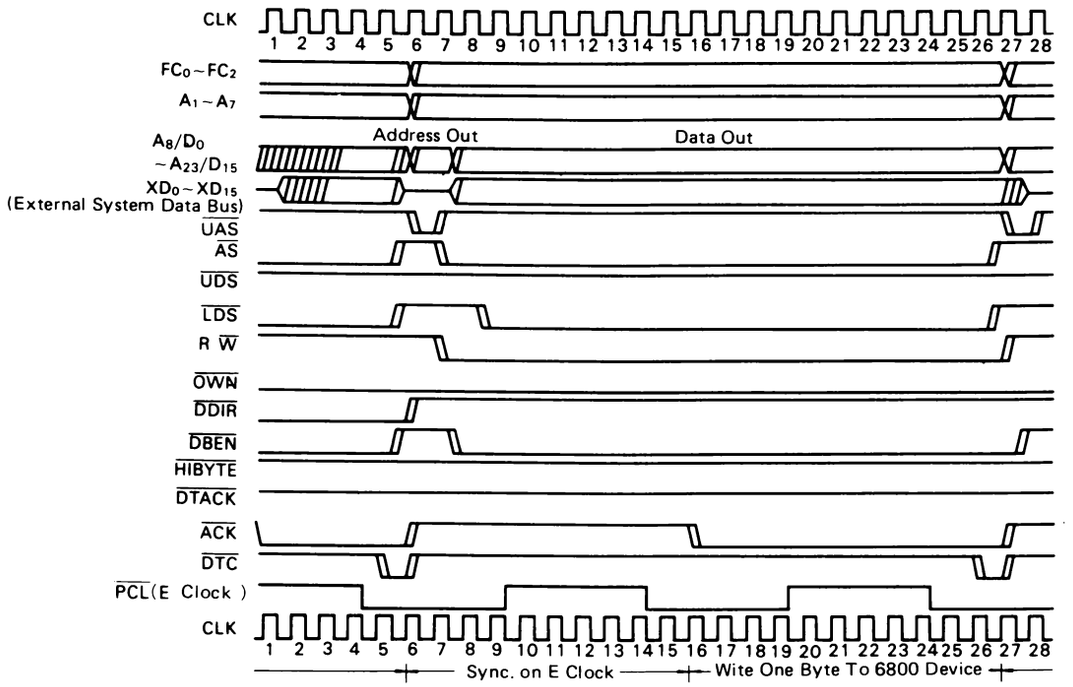


Figure 27 Dual Addressing Mode, HMCS6800 Compatible Device, Write Cycle

(2) Single Addressing Mode

Implicitly addressed devices are peripheral devices selected not by address but by $\overline{ACK}$. They do not require addressing of data register during data transfer. Transfers between memory and these devices are controlled by the request/acknowledge protocol. Such peripherals require only one bus cycle to transfer data, and the DMAC internal holding register is not used. Because only the memory is addressed during a data transfer and a transfer done in only on bus cycle, this protocol is called single-address.

Device with $\overline{ACK}$ Transfers

Under this protocol, the communication between peripheral device and the DMAC is performed with a two signal $\overline{REQ}/\overline{ACK}$ handshake. When a request is generated using the request method programmed in the DMAC's internal control registers, the DMAC obtains the bus and responds with $\overline{ACK}$. The DMAC asserts all the bus control signals required for the memory access. Refer to Figure 28 for the flowchart of the data transfer from memory to the device with $\overline{ACK}$. Figure 29 shows the flowchart of the data transfer from the device with $\overline{ACK}$ to memory. When a request is generated using the request method programmed in the control registers, the DMAC obtains the bus and responds with acknowledge. The DMAC asserts all HMCS68000 bus control signals needed for the transfer. When the DMAC accepts $\overline{DTACK}$ from memory, it asserts $\overline{DTC}$ and informs the peripheral device of the transfer termination. Figure 30 and 31 show the transfer timings of the device with $\overline{ACK}$: the port size for the former figure is 8-bit and the latter is 16-bit respectively.

When the transfer is from memory to a device, data is valid when $\overline{DTACK}$ is asserted and remains valid until the data strobes are negated. The assertion of $\overline{DTC}$ from the DMAC may be used to latch the data.

When the transfer is from device to memory, data must be valid on the HMCS68000 bus before the DMAC asserts the data strobes. The data strobes are asserted one clock period after $\overline{ACK}$ is asserted. When the DMAC obtains the bus and starts a DMA cycle, the tri-state of the $\overline{OWN}$ line is cancelled a half clock earlier than other control lines. If the DMA Cycle terminates and the DMAC relinquishes the bus, all the control signals get tri-stated a half clock before $\overline{OWN}$. The $\overline{DDIR}$ and $\overline{DBEN}$ lines are not asserted in the single addressing mode. Four clocks cycle is the smallest bus cycle for the transfer from memory to device. Five clocks cycle is the smallest bus cycle for the transfer from device to memory. If the device port size is 8-bit, either $\overline{LDS}$ or $\overline{UDS}$ is asserted. In the single adressing mode, A_8-A_{23} are outputted for only one and a half clock from the beginning of the DMA bus cycle. Therefore, A_8 through A_{23} needs to be latched externally just like in the dual addressing mode.

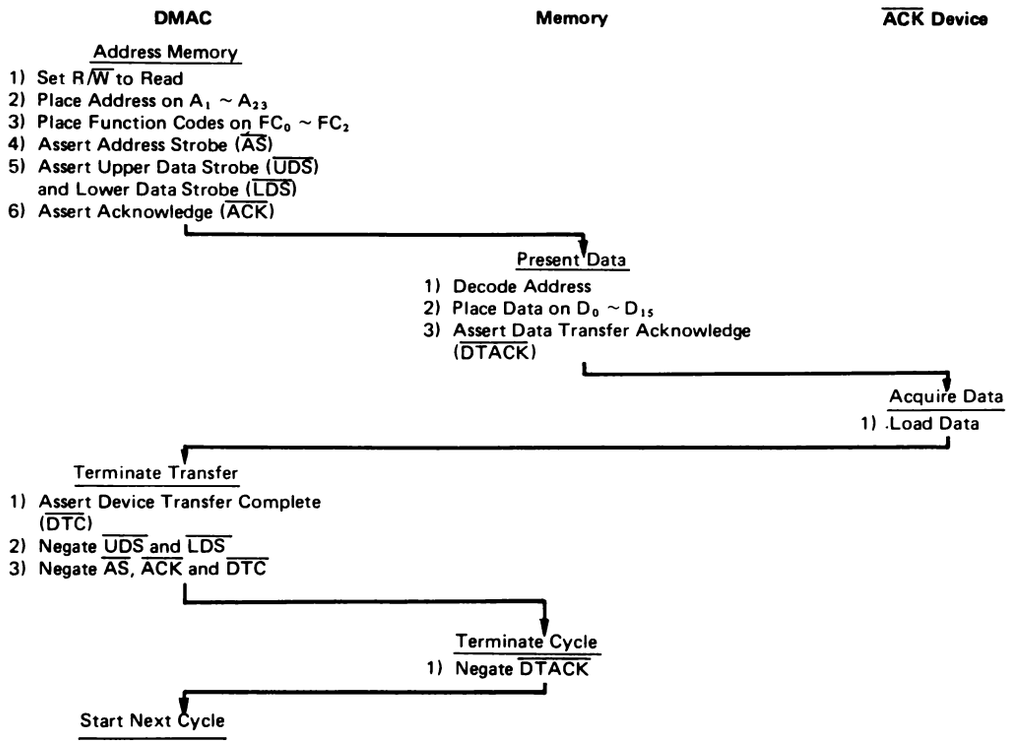


Figure 28 Word from Memory to Device with ACK

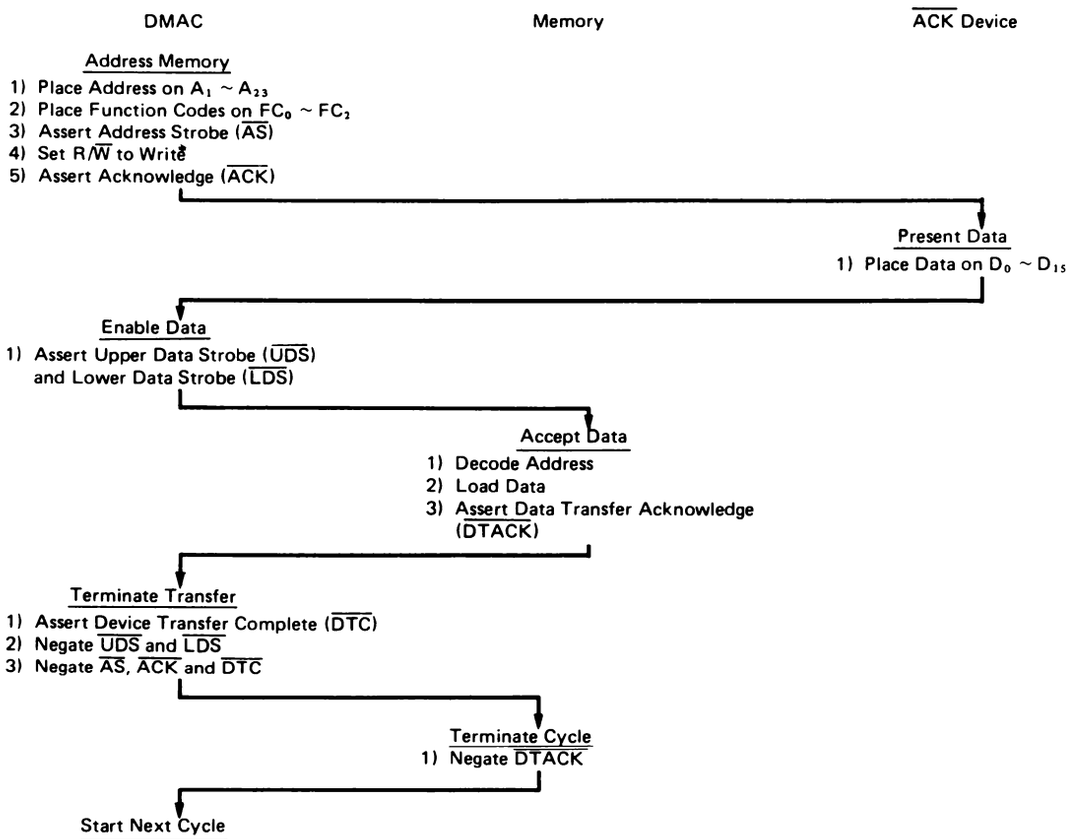


Figure 29 Word from Device with ACK to Memory

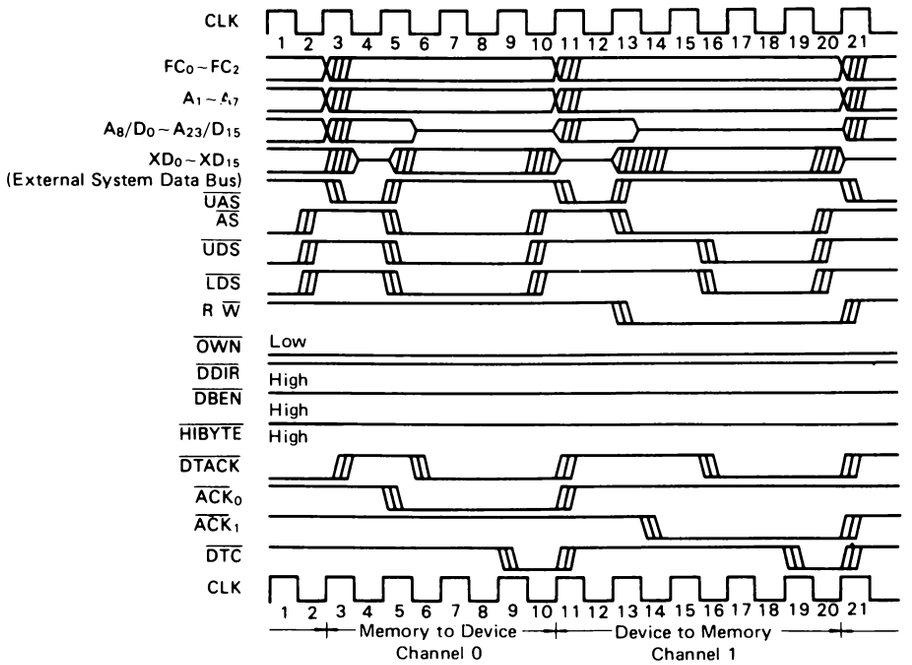


Figure 30 Single Addressing Mode with 16-Bit Devices as Source and Destination (Read-Write Cycles)

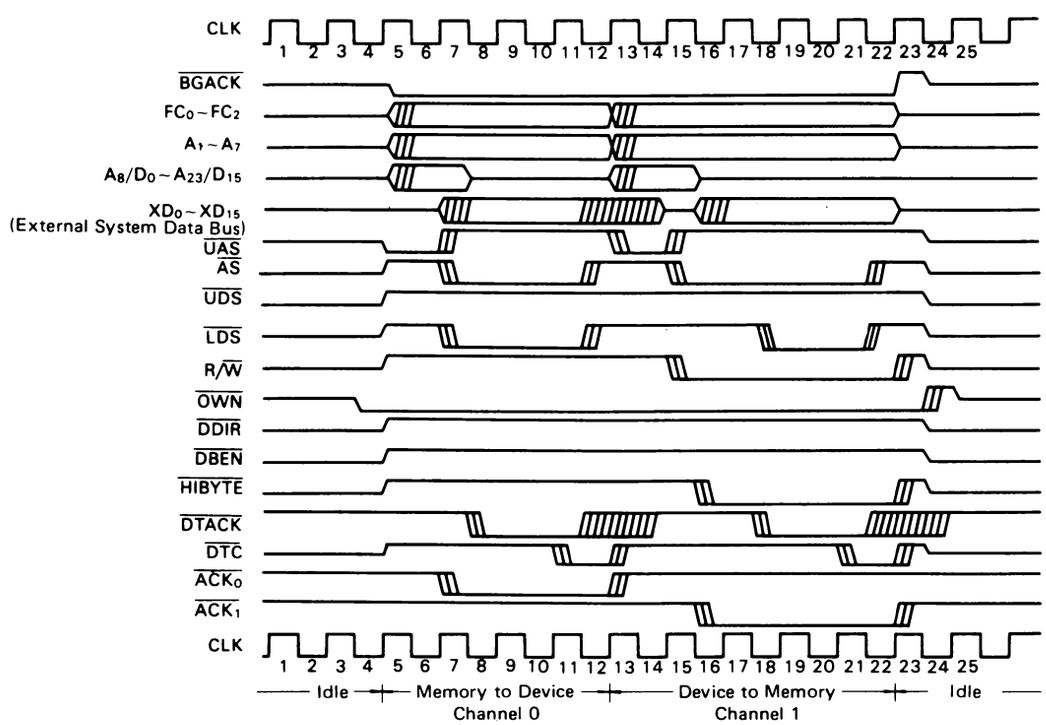


Figure 31 Single Addressing Mode with 8-Bit Device as Source and Destination (Read-Write Cycles)

Device with $\overline{ACK}$ and $\overline{READY}$ Transfers

Under this protocol, the communication between peripheral device and the DMAC is performed using a three signal $\overline{REQ}/\overline{ACK}/\overline{READY}$ handshake. The $\overline{READY}$ input to the DMAC is provided by the $\overline{PCL}$ line. The $\overline{READY}$ line is active low. When a request is generated using the request method programmed in the control registers, the DMAC obtains the bus and asserts $\overline{ACK}$ to notify the device that the transfer is to take place. The DMAC waits for $\overline{READY}$ ($\overline{PCL}$ input), which is a response from the device, in addition to $\overline{DTACK}$ which is a response from memory.

When the DMAC accepts both signals, it terminates the transfer. Refer to Figures 33 and 34 for the flowcharts of the data transfer between memory and the device with $\overline{ACK}$ and $\overline{READY}$. Refer to Figure 35 for the transfer timing of the 8-bit device. When the data transfer is from memory to a device, data is valid from the assertion of $\overline{DTACK}$ to the negation of $\overline{LDS}$ and $\overline{UDS}$. $\overline{DTC}$ is asserted a half clock before $\overline{LDS}$ and $\overline{UDS}$ are negated, so this line may be used for latching the data by the peripheral device. In this case, $\overline{READY}$ ($\overline{PCL}$ input) indicates that the device has received the data. Both $\overline{DTACK}$ and $\overline{READY}$ ($\overline{PCL}$ input) signals are needed for terminating the DMA cycle.

When the data transfer is from the device to memory, data must be valid on the bus before the DMAC asserts $\overline{LDS}$ and $\overline{UDS}$. Therefore, $\overline{READY}$ ($\overline{PCL}$ input) is used as the signal to indicate that the peripheral device has outputted the data on the bus. When the DMAC detects PCL (READY input), then it asserts $\overline{LDS}$ and $\overline{UDS}$. After asserting $\overline{LDS}$ and $\overline{UDS}$, the DMAC terminates the cycle when $\overline{DTACK}$ signal from the memory is detected.

When Array Chain or Link Array Chain is set in Device with $\overline{ACK}$ and $\overline{READY}$ Transfer mode, $\overline{READY}$ input is also necessary during DMA bus cycles for reading the chain information from memory. The circuit as shown in Figure 32 may be used in order to generate $\overline{READY}$ input when reading the chain information from memory.

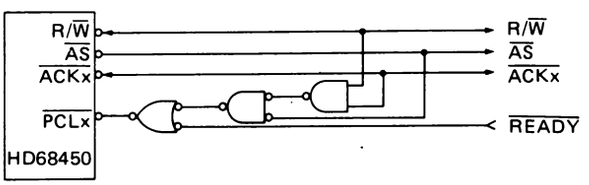


Figure 32 $\overline{READY}$ Circuit When Array or Link Array Chain is set for Device with $\overline{ACK}$ and $\overline{READY}$

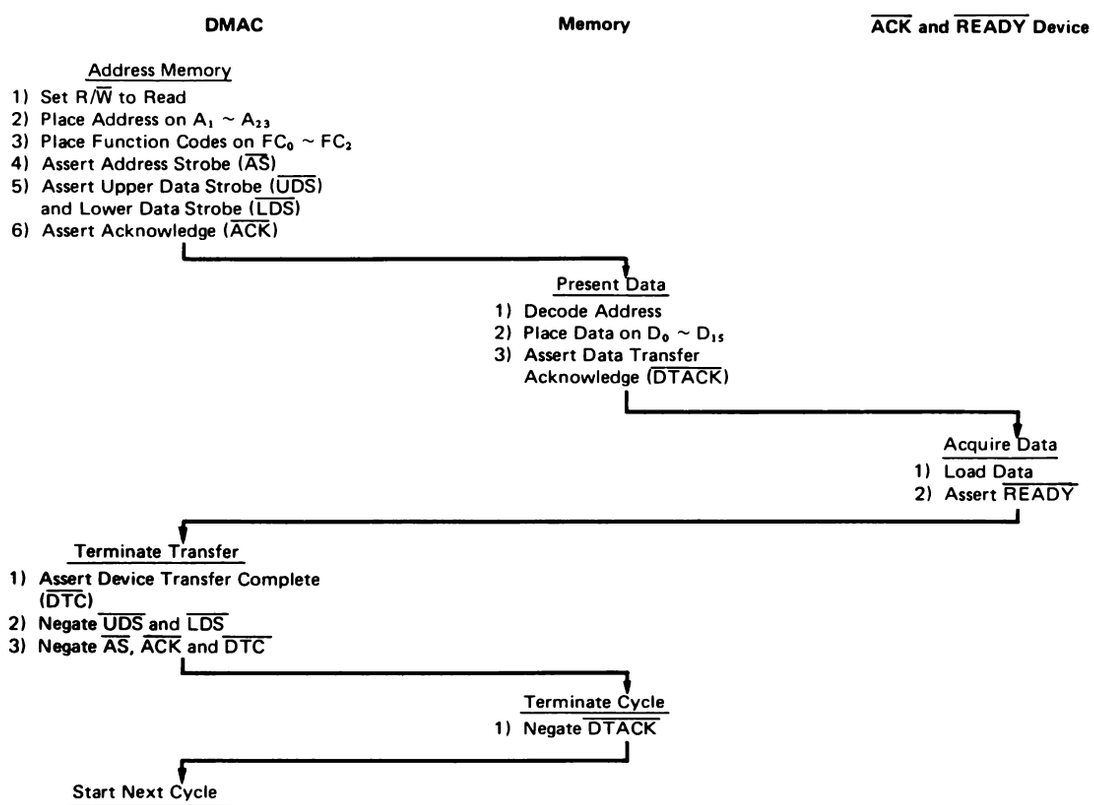


Figure 33 Word from Memory to Device with $\overline{ACK}$ and $\overline{READY}$

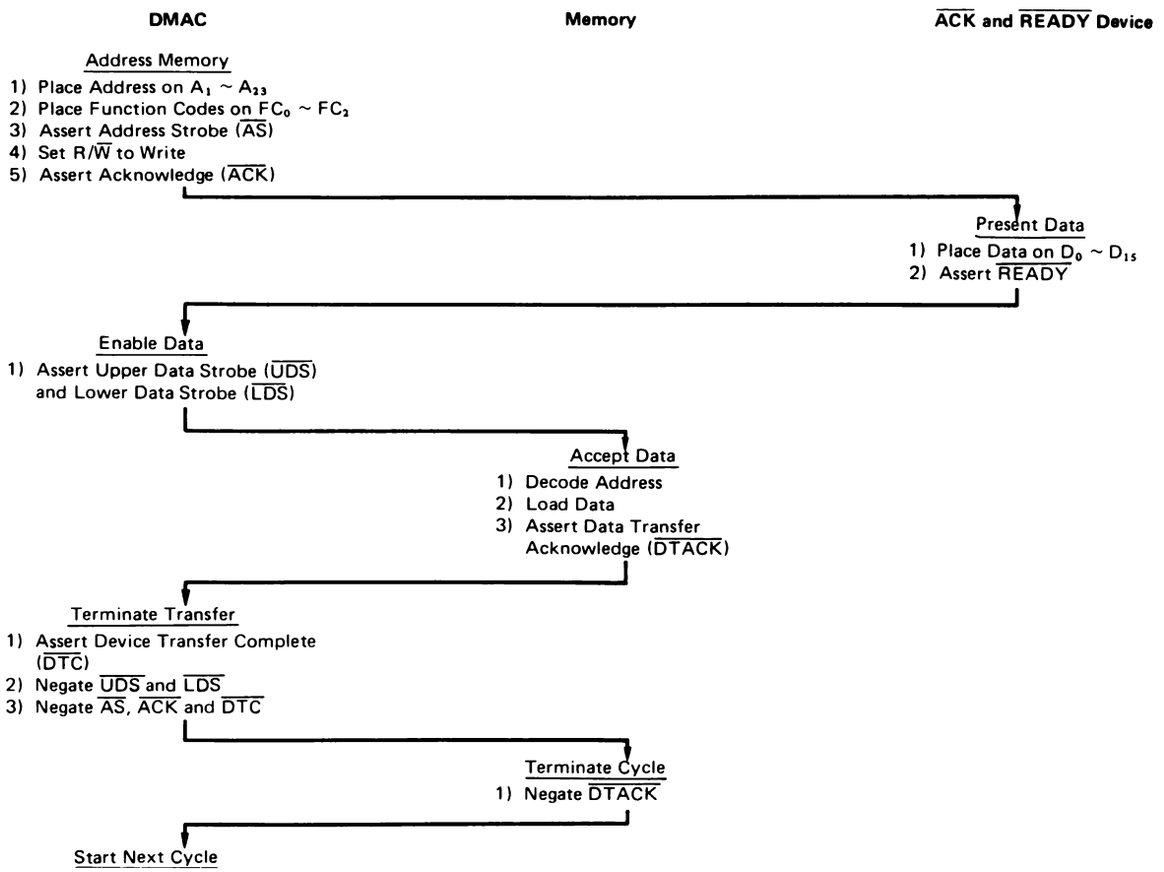


Figure 34 Word from Device with ACK and READY to Memory

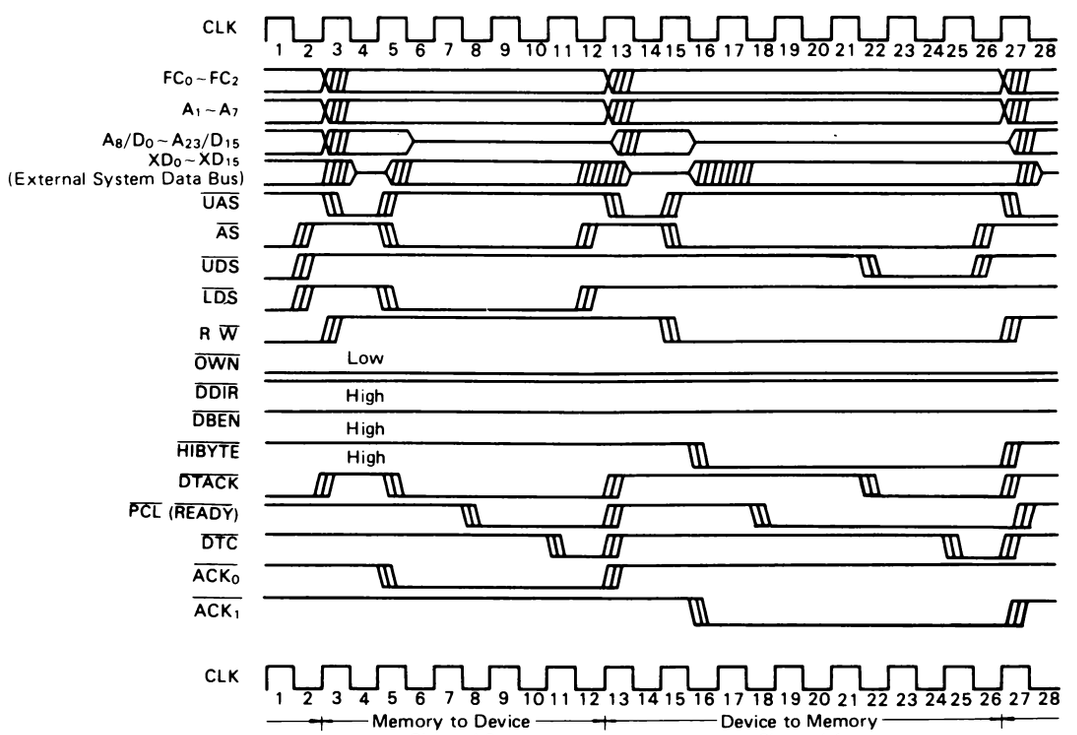


Figure 35 Single Addressing Mode with 8-Bit Devices as Source and Destination with PCL Used as a READY Input (Read-Write Cycles)

Operands and Addressing

Three factors enter into how the actual data is handled: port size, operand size and address sequencing.

Port Size

The DCR is used to program the device port size.

DPS Device Port Size
- 0 8 bit port
- 1 16 bit port

The port size is the number of bits of data which the device can transfer in a single bus cycle. During a DMAC bus cycle, a 16-bit port transfers 16 bits of data on $D_0 \sim D_{15}$, while an 8-bit port transfers 8 bits of data, either on $D_0 \sim D_7$ or on $D_8 \sim D_{15}$. The memory is always assumed to have a port size of 16.

Operand Size

OCR is used to program the operand size.

SIZE Operand Size
- 00 Byte
- 01 Word
- 10 Long word
- 11 (undefined, reserved)

The operand size is the number of bits of data to be transferred to honor a single request. Multiple bus cycles may be required to transfer the operand through the device port. A byte operand consists of 8 bits of data, a word operand consists of 16 bits of data, a long word operand consists of 32 bits of data. The transfer counter counts the number of operands transferred.

Table 2 indicates the combinations supported by the DMAC about the peripheral devices with different port size and operand sizes in the single and dual addressing mode. In the single addressing mode, port size and operand size must be the same. In the dual addressing mode, byte operand cannot be used when the port size is sixteen and the REQG bit is 10 or 11.

Table 2 Operation Combinations

Addressing	Device Type	Port	Operand			REQG bits of OCR
			Byte	Word	Long Word	
Dual	68000, 6800	8	○	○	○	00, 01, 10, 11
Dual	68000, 6800	16	○	○	○	00, 01
Dual	68000, 6800	16	X	○	○	10, 11
Single	with $\overline{ACK}$ or $\overline{ACK}$ & $\overline{READY}$	8	○	X	X	00, 01, 10, 11
		16	X	○	X	00, 01, 10, 11

○ ; enable X ; disable

(3) Address Sequencing

The sequence of addresses generated depends upon the port size, operand size, whether the addresses are to count up, down or not change and whether the transfer is executed in the single addressing mode or the dual addressing mode. The memory address count method and the peripheral device address count method is programmed using the Memory address count (MAC) bit and the Device address count (DAC) bit in the Sequence Control Register (SCR).

(i) Single addressing mode

In the single addressing mode, memory address sequencing is shown in Table 3. If the operand size is byte, the memory address increment is one (1). If the operand size is word, the memory address increment is two (2). If the memory address register does not count, the memory address is unchanged after the transfer.

If the memory address counts up, the increment is added to the memory address; if the memory address counts down, the increment is subtracted from the memory address. The memory address is changed after the operand is transferred.

Table 3 Single Address Sequencing

Port Size	Operand Size	Memory Address Increment		
		+ (increment)	= (unchanged)	- (decrement)
8	Byte	+1	0	-1
16	Word	+2	0	-2

(ii) Dual addressing mode

In the dual addressing mode, the operand size need not match the port size. Thus the transfer of an operand may require several DMA bus cycles. Each DMA bus cycle, between memory and DMAC and between DMAC and the device, is called the operand part and transfers a portion or all of the operand. The addresses of the operand parts are in a linear increasing sequence. The step between the addresses of the operand is two. The size of the operand parts is the minimum of the port size and the operand size. The number of the operand part is the operand size divided by the port size. In the dual addressing mode, memory is regarded as a device whose port size is 16-bits.

If the port size is 16 bits, the operand size is byte, and the request generation method is auto request or auto request at a limited rate, the DMAC packs consecutive transfers. This means that word transfers are made from the associated address with an address increment of two (2). If the initial source address location contains a single byte, the first transfer is a byte transfer to the internal DMAC holding register, and subsequent transfers from the source are word transfers. If the initial destination location contains a single byte, the first transfer is a byte transfer from the internal DMAC holding register, and any remaining byte remains in the holding register. Likewise, if either the final source or destination location contains a single byte, only a byte transfer is done. Packing is not performed if the address does not count; each byte is transferred by a separate access to the same location. The dual address sequencing is shown in Table 4.

Table 4 Dual Address Sequencing

Port Size	Operand Size	Part Size	Operand Part Address	Address Increment		
				+	=	-
8	Byte	Byte	A	+2	0	-2
8	Word	Byte	A, A+2	+4	0	-4
8	Long	Byte	A, A+2, A+4, A+6	+8	0	-8
16	Byte	Pack	A	+P	0	-P
16	Word	Word	A	+2	0	-2
16	Long	Word	A, A+2	+4	0	-4

P = 1 if packing is not done
= 2 if packing is done

Pack = byte if packing is not done
= word if packing is done

An Example of a Dual Address Transfer

This section contains an example of a dual address transfer using Table 4 of Dual-Address Sequencing. The table is reproduced here as Table 5. The transfer mode of this example is the following:

1. Device Port size = 8 bits
2. Operand size = Long Word (32 bits)
3. Memory to Device Transfer
4. Source (Memory) Counts up, Destination (Device) Counts Down
5. Memory Transfer Counter = 2

In this mode, a data transfer from the source (memory) is done according to the 6th row of Table 5, since the port size of the memory is always 16 bits. A data transfer to the destination (device) is done according to the 3rd row of Table 5.
Table 6 shows the data transfer sequence.

The memory map of this example is shown in Table 7. The operand consists of BYTE A through BYTE D in memory of Table 7. Prior to the transfer, MAR and DAR are set to 00000012 and 00000108 respectively. The operand is transferred to the 8 bit port device according to the order of transfer number in Table 6.

Table 5 Dual-Address Sequencing (Table 4)

Row No.	Port Size	Operand Size	Operand Part Size	Operand Part Addresses	Address Increment		
					+	=	-
1	8	BYTE	BYTE	A	+2	0	-2
2	8	WORD	BYTE	A, A+2	+4	0	-4
③	8	LONG	BYTE *4	A, A+2, A+4, A+6 *3 *5 *7 *8	+8	0	-8 *10
4	16	BYTE	PACK (BYTE or WORD)**	A	+P	0	-P
5	16	WORD	WORD	A	+2	0	-2
⑥	16	LONG	WORD *2	A, A+2 *1 *6	+4 *9	0	-4

* Numbers in Table 5 correspond to ones in Table 6 and 7.
** Refer to Address Sequencing on Operand Part Size and PACK.

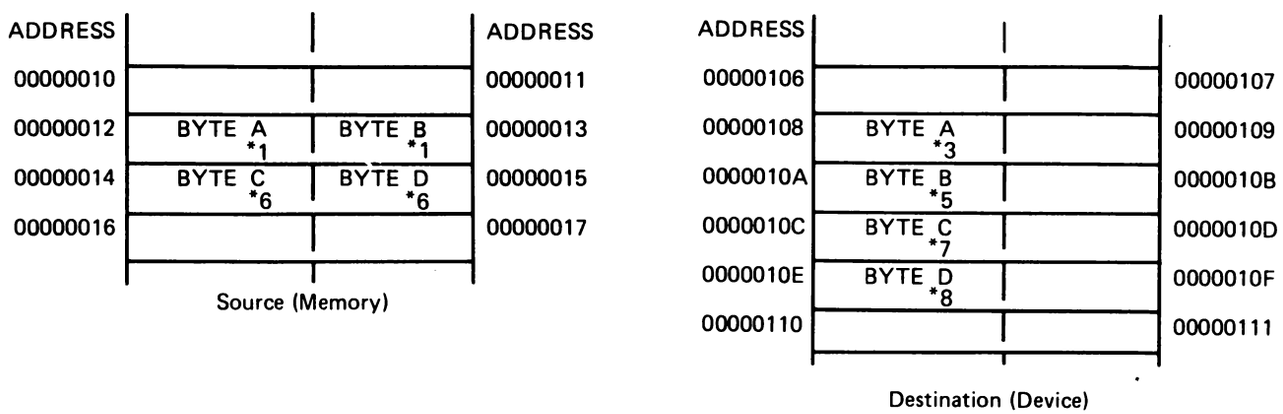
Table 6 An Example of a Data Transfer for One Operand

SRC: Source (Memory), DST Destination (Device), HR: Holding Register (DMAC Internal Reg.)

Transfer No.	Data Transfer	Address Output	Data Size on Bus	DMAC Registers after Transfer		Comment
				MAR	DAR	
0	–	–	–	00000012	00000108	Initial Register Setting
1	SRC → HR	00000012 *1	WORD *2	00000014	00000108	Higher order 16 bits of operand is fetched.
2	HR → DST	00000108 *3	BYTE *4	00000014	0000010A	Higher order 16 bits of operand is transferred.
3	HR → DST	0000010A *5	BYTE *4	00000014	0000010C *10	
4	SRC → HR	00000014 *6	WORD *2	00000016 *9	0000010C	Lower order 16 bits of operand is fetched
5	HR → DST	0000010C *7	BYTE *4	00000016	0000010E	Lower order 16 bits of operand is transferred.
6	HR → DST	0000010E *8	BYTE *4	00000016	00000110 *10	
6'	–	–	–	00000016	00000110	MAR, DAR are pointing the next operand addresses when the transfer is complete.

Mode: Port size = 8, Operand size = Long Word, Memory to Device, Source (Memory) Counts Up, Destination (Device) Counts Down

Table 7 Memory Map for the Example of the Data Transfer



- **Initiation and Control of Channel Operation**

(1) Operation Initiation

To initiate the operation of a channel the STR bit of the CCR is set to start the operation. Setting the STR bit causes the immediate activation of the channel, the channel will be ready to accept requests immediately. The channel initiates the operation by resetting the STR bit and setting the channel active bit in the CSR. Any pending requests are cleared, and the channel is then ready to receive requests for the new operation. If the channel is configured for an illegal operation, the configuration error is signaled, and no channel operation is run. The illegal operations include the selection of any of the options marked "(undefined, reserved)". If the MTC is set to zero in any operation or BTC is set to zero in the array chaining mode, then the count error is signaled and the channel is not activated. The channel cannot be started if any of the ACT, COC, BTC, NDT or ERR bits is set in the CSR. In this case, the channel signals the operation timing error.

(2) Operation Continuation (Continue Mode)

The continue bit (CNT) allows multiple blocks to be transferred in unchained operations. The CNT bit is set in order to continue the current channel operation. If an attempt is made to continue a chained operation, a configuration error is signaled. The base address register and base transfer counter should have been previously initialized.

The continue bit may be set as the channel is started or while the channel is still active. The operation timing error bit is signaled if a continuation is otherwise attempted.

When the memory transfer counter is exhausted and the continue bit of the CCR is set, the DMAC performs a continuation of the channel operation. The base address, base function code, and base transfer count registers are copied into the memory address, memory function code, and memory transfer count registers. The block transfer complete (BTC) bit of the CSR is set, the continue bit is reset, and the channel begins a new block transfer. If the memory transfer counter is loaded with a terminal count, the count error is signaled.

(3) Operation Halting (Halt)

The CCR has a halt bit which allows suspension of the operation of the channel. If this bit is set, a request may still be generated and recognized, but the DMAC does not attempt to acquire the bus or to make transfers for the halted channel. When this bit is reset, the channel resumes operation and services any request that may have been received while the channel was halted. However, in the burst request mode, the transfer request should be kept asserted until the initiation of the first transfer after clearing the halt bit.

(4) Operation Abort by Software (Software Abort)

Setting the software abort bit (SAB) in the CCR allows the current operation of the channel to be aborted. In this case, the ERR bit and the COC bit in the CSR are set and the ACT bit is reset. The error code for the software abort is set in the CER. The SAB bit is designed to be reset if the ERR bit is reset. When the CCR is read, the SAB always reads as zero(0).

(5) Interrupt Enable

The CCR has an interrupt enable bit (INT) which allows the channel to request interrupts. If INT is set, the channel can request interrupts. If it is clear, the channel will not request interrupts.

- **Channel Operation Termination**

As part of the transfer of an operand, the DMAC decrements the memory transfer counter (MTC). If the chaining mode is not used and the CNT bit is not set or the last block is transferred in the chaining mode, the operation of the channel is complete when the last operand transfer is completed and the MTC is zero. The DMAC notifies the peripheral device of the channel completion via the $\overline{\text{DONE}}$ output.

However, in the continue mode, $\overline{\text{DONE}}$ is outputted at the termination of every data block transfer. When the channel operation has been completed, the ACT bit of the CSR is cleared, and the COC bit of the CSR is set.

The occurrence of errors, such as the bus error, during the DMA bus cycle also terminates the channel operation. In this case, the ACT bit in the CSR is cleared, the ERR and the COC bits are set, and at the same time the code corresponding to the error that occurred is set in the CER.

(1) Channel Status Register (CSR)

The channel status register contains the status of the channel. The register, except for ACT and PCS bits, is cleared by writing a one (1) into each bit of the register to be cleared. Those bits positions which contain a zero (0) in the write data remain unaffected. ACT and PCS bits are unaffected by the write operation.

COC

The channel operation complete (COC) bit is set if the channel operation has completed. The COC bit must be cleared in order to start another channel operation. The COC bit is cleared only by writing a one to this bit or resetting the DMAC.

PCS

The peripheral status (PCS) bit reflects the level of the $\overline{\text{PCL}}$ line regardless of its programmed function. If $\overline{\text{PCL}}$ is at "High" level, the PCB bit reads as one. If $\overline{\text{PCL}}$ is at "Low" level, the PCS bit reads as zero. The PCS bit is unaffected by writing to the CSR.

PCT

The peripheral control transition (PCT) bit is set, if a falling edge transition has occurred on the $\overline{\text{PCL}}$ line. (The $\overline{\text{PCL}}$ line must remain at "low" level for at least two clock cycles.) The PCT bit is cleared by writing a one to this bit or resetting the DMAC.

BTC

Block transfer complete (BTC) bit is set when the continue (CNT) bit of CCR is set and the memory transfer counter (MTC) is exhausted. The BTC bit must be cleared before the another continuation is attempted (namely, setting the CNT bit again), otherwise an operation timing error occurs. The BTC bit is cleared by writing a one to this bit or resetting the DMAC.

NDT

Normal device termination (NDT) bit is set when the peripheral device terminates the channel operation by asserting the $\overline{\text{DONE}}$ line while the peripheral device was being acknowledged. The NDT bit is cleared by writing a one to this bit or resetting the DMAC.

ERR

Error (ERR) bit is set if any errors have been signaled. When the ERR bit is set, the code corresponding to the kind of the error that occurred is set in the CER. The ERR bit is cleared by writing a one to this bit or resetting the DMAC.

ACT

The active (ACT) bit is asserted after the STR bit has been set and the channel operation has started. This bit is remains set until the channel operation is terminated. The ACT bit is unaffected by write operations. This bit is cleared by the termination of the channel or resetting the DMAC.

(2) Interrupts

The DMAC can signal the termination of the channel operation by generating an interrupt request. The INT bit of the CCR determines if an interrupt can be generated. The interrupt request is generated by the following condition.

① INT = 1
and
② COC = 1 or BTC = 1 or ERR = 1 or NDT = 1 or PCT = 1
(the $\overline{\text{PCL}}$ line is an interrupt input)

This may be represented as

$$\overline{\text{IRQ}} = \text{INT} \cdot (\text{COC} + \text{BTC} + \text{ERR} + \text{NDT} + \text{PCT}^*)$$

(*$\overline{\text{PCL}}$ line is programmed as an interrupt input.)

When the $\overline{\text{IRQ}}$ line is asserted, changing the INT bit from one to zero to one will cause the $\overline{\text{IRQ}}$ output to change from "low" to "high" to "low" again. The $\overline{\text{IRQ}}$ should be negated by clearing the COC, the BTC, the ERR, the NDT and the PCT bits.

If the DMAC receives $\overline{\text{IACK}}$ from the MPU during asserting the $\overline{\text{IRQ}}$, the DMAC provides an interrupt vector. If multiple channels have interrupt requests, the determination of which channel presents its interrupt vector is made using the same priority scheme defined for the channel operations.

The bus cycle in which the DMAC provides the interrupt vector when receiving an $\overline{\text{IACK}}$ from the MPU is called the interrupt acknowledge cycle. The interrupt vector returned to the MPU comes from either the normal or the error interrupt vector register. The normal interrupt register is used unless the ERR bit of CSR is set, in which case the error interrupt vector register is used. The content of the interrupt vector register is placed on $D_0 \sim D_7$, and $\overline{\text{DTACK}}$ is asserted to indicate that the vector is on the data bus. If a reset occurs, all interrupt vector registers are set to $0F (binary 00001111), the value of the uninitialized interrupt vector. The timing of the interrupt acknowledge cycle is shown in Figure 36. The HD68000 MPU outputs the interrupt level into A_1-A_3 and A_4-A_7 is held "high" during the interrupt acknowledge cycle, but the HD68450 DMAC ignores these signals.

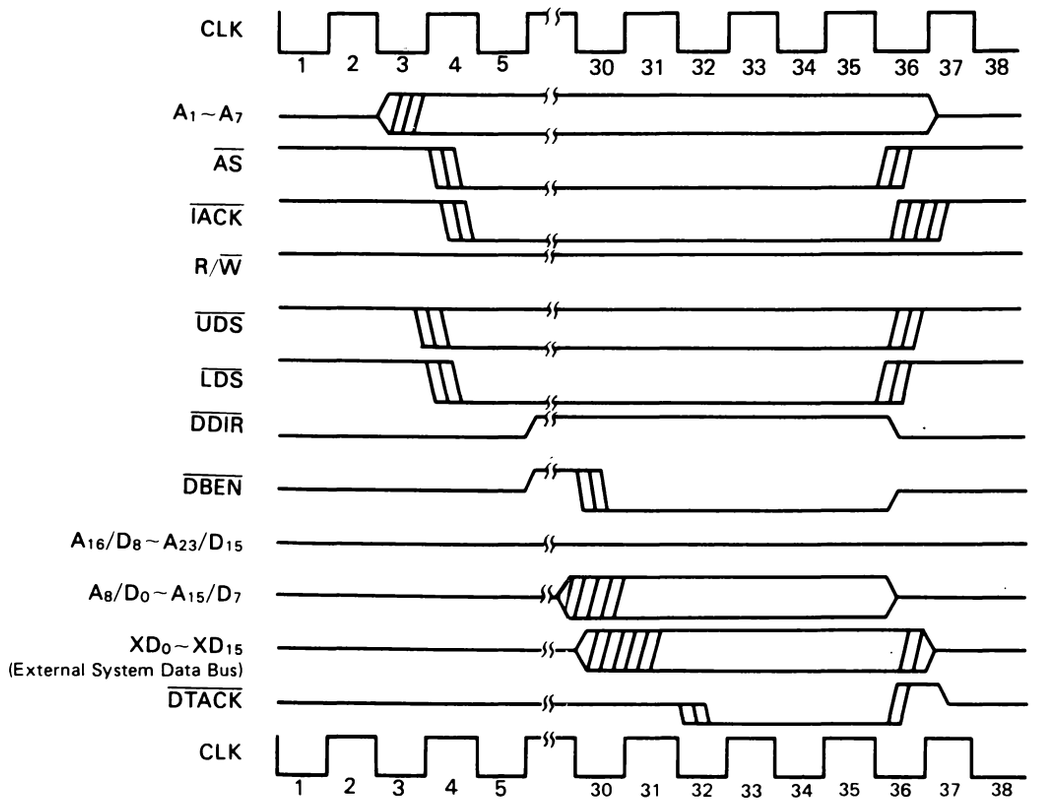


Figure 36 MPU IACK Cycle to DMAC

(3) Multiple Data Block Transfer Operation

When the memory transfer counter (MTC) is exhausted, the channel operation still continues if the channel is set to the array chaining mode or the linked array chaining mode and the chain is not exhausted. The channel operation also containes if the continue bit (CNT) of the CCR is set. The DMAC provides the initialization of the memory address register and the memory transfer counter in these cases so that the DMAC can transfer the multiple blocks.

Continued Operation

The continued operation is described in the Initiation and the Control of the Channel Operation section.

Array Chaining

This type of chaining uses an array in memory consisting of memory addresses and transfer counts. Each entry in the array is six bytes long and, consists of four bytes of address followed by two bytes of transfer count. The beginning address of this array is in the base address register, and the number of entries in the array is in the base transfer counter. Before starting any block transfers, the DMAC fetches the entry currently pointed to by the base address register. The address information is placed in the memory address register, and the count information is placed in the memory transfer counter. As each chaining entry is fetched, the base transfer counter is decremented by one. After the chaining entry is fetched, the base address register is incremented to point the next entry. When the base transfer counter reaches a terminal count of zero, the chain is exhausted, and the entry just fetched determines the last block of the channel operation.

An example of the array chaining mode operation and the memory format for supporting for array chaining is shown in Figure 37. The array must start at an even address, or the entry fetch results is an address error. If a terminal count is loaded into the memory transfer counter or the base transfer counter, the count error is signaled. Since the base registers may be read by the MPU, appropriate error recovery information is available should the DMAC encounter an error anywhere in the chain. Contents of the BFC is outputted as the function code when the DMAC is accessing the memory using the base address register. The value of the function code registers are unchanged in the array chaining operation.

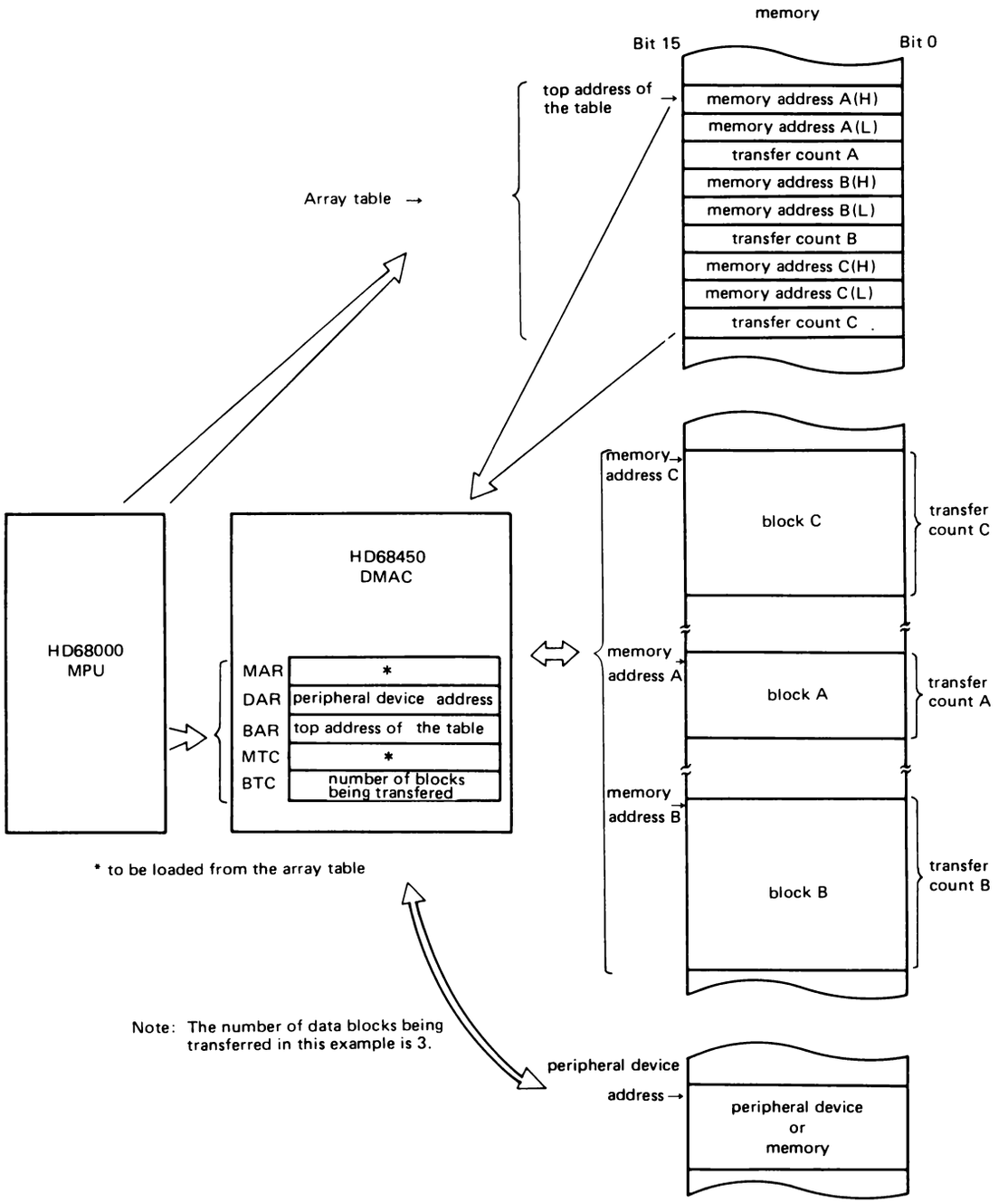


Figure 37 Transfer Example of the Array Chaining Mode

Linked Array Chaining

This type of chaining uses a list in memory consisting of memory address, transfer counts, and link addresses. Each entry in the chain list is ten bytes long, and consists of four bytes of memory address, two bytes of transfer count and four bytes of link address. The address of the first entry in the list is in the base address register, and the base transfer counter is unused. Before starting any block transfers, the DMAC fetches the entry currently pointed to by the base address register. The address information is placed in the memory address register, the count information is placed in the memory transfer counter, and the link address replaces the current contents of the base address register. The channel then begins a new block transfer. As each chaining entry is fetched, the update base address register is examined for the terminal link which has all 32 bits equal to zero. When the new base address is the terminal address, the chain is exhausted, and the entry just fetched determines the last block of the channel operation.

An example of the linked array chaining mode operation and the memory format for supporting it is shown is Figure 38.

In Figure 38, the DMAC transfers data blocks in the order of Block A, Block B, and Block C. In the linked array chaining

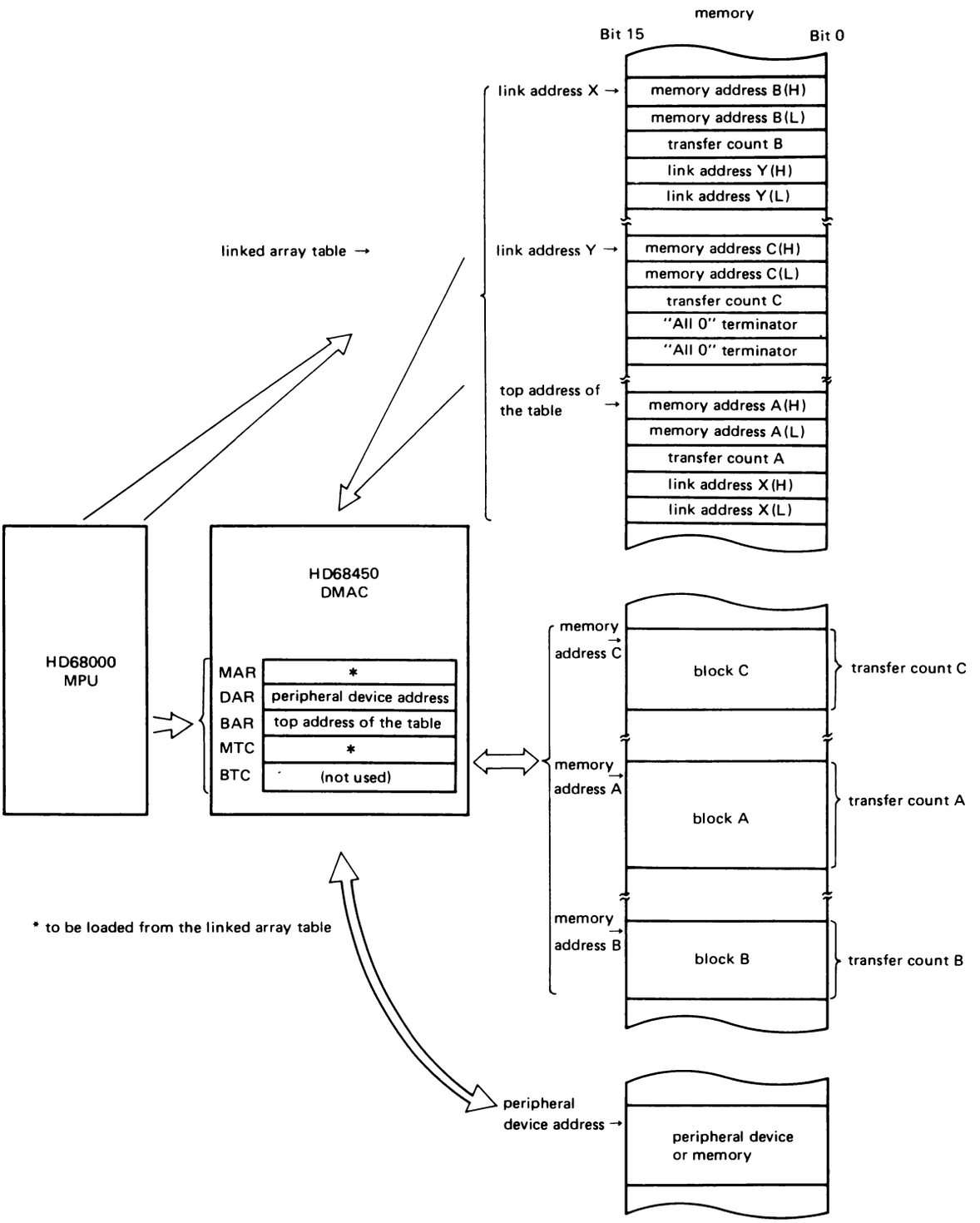


Figure 38 Transfer Example of the Linked Array Chaining Mode

mode, the BTC is not used. When the DMAC refers to the linked array table, the value of the BFC is outputted as the function code. The values of the function code registers are unchanged by the linked array chaining operation.

This type of chaining allows entries to be easily removed or inserted without having to reorganize data within the chain. Since the end of the chain is indicated by a terminal link, the number of entries in the array need not be specified to the DMAC.

The linked array table must start at an even address in the linked array chaining mode. Starting the table at an odd address results in an address error. If "0" is initially loaded to the MTC, the count error is signaled. Because the MPU can read all of the DMAC registers, all necessary error recovery information is available to the operating system.

The comparision of both chaining modes is shown in Table 8.

Table 8 Chaining Mode Address/Count Information

Chaining Mode	Base Address Register	Base Transfer Counter	Completed When
Array Chaining	address of the array table	number of data blocks being transferred	Base Transfer Count = 0
Linked Array Chaining	address of the linked array table	(unused)	Linked Address = 0

(4) Bus Exception Conditions

The DMAC has three lines for inputting bus exception conditions called $\overline{BEC_0}$, $\overline{BEC_1}$, and $\overline{BEC_2}$. The priority encoder can be used to generate these signals externally. These lines are encoded as shown in Table 9.

Table 9

$\overline{BEC_2}$	$\overline{BEC_1}$	$\overline{BEC_0}$	Exception Condition
1	1	1	No exception condition
1	1	0	Halt
1	0	1	Bus error
1	0	0	Retry
0	1	1	Relinquish bus and retry
0	1	0	(undefined, reserved)
0	0	1	(undefined, reserved)
0	0	0	Reset

In order to guarantee, reliable decoding, the DMAC verifies that the incoming code has been statable for two DMAC clock cycles before acting on it. The DMAC picks up $\overline{BEC_0}$-$\overline{BEC_2}$ at the rising edge of the clock. If $\overline{BEC_0}$-$\overline{BEC_2}$ is asserted to the undefined code, the operation of the DMAC does not proceed. For example, when the DMAC is waiting for $\overline{DTACK}$, inputting $\overline{DTACK}$ does not result in the termination of the cycle if $\overline{BEC_0}$-$\overline{BEC_2}$ is asserted to the undefined code. In addition, when the transfer request is received, $\overline{BR}$ is not asserted if the $\overline{BEC_0}$-$\overline{BEC_2}$ is not set to no exception condition.

If exception condition, except for HALT, is inputted during the DMA bus cycle prior to, or in coincidence with $\overline{DTACK}$, the DMAC terminates the current channel operation immediately. Here coincident means meeting the same set up requirements for the same sampling edge of the clock. If a bus exception condition exists, the DMAC does not generate any bus cycles until it is removed. However, the DMAC still recognizes requests.

Halt

The timing diagram of halt is shown in Figure 39. This diagram shows halt being generated during a read cycle from the 68000 compatible device in the dual addressing mode. If the halt exception is asserted during a DMA bus cycle, the DMAC does not terminate the bus cycle immediately. The DMAC waits for the assertion of $\overline{DTACK}$ before terminating the bus cycle so that the bus cycle is completed normally. In the halted state, the DMAC puts all the control signals to high impedance and relinquishes the bus to the MPU. The DMAC does not output the $\overline{BR}$ until halt exception is negated. When halt exception is negated, the DMAC acquires the bus again and proceeds the DMA operation. In order to insure a halt exception operation, the $\overline{BEC}$ lines must be set to halt at least until the assertion of $\overline{DTC}$.

If the DMAC has the bus, but is not executing any bus cycle, the DMAC relinquishes the bus as soon as halt exception is asserted.

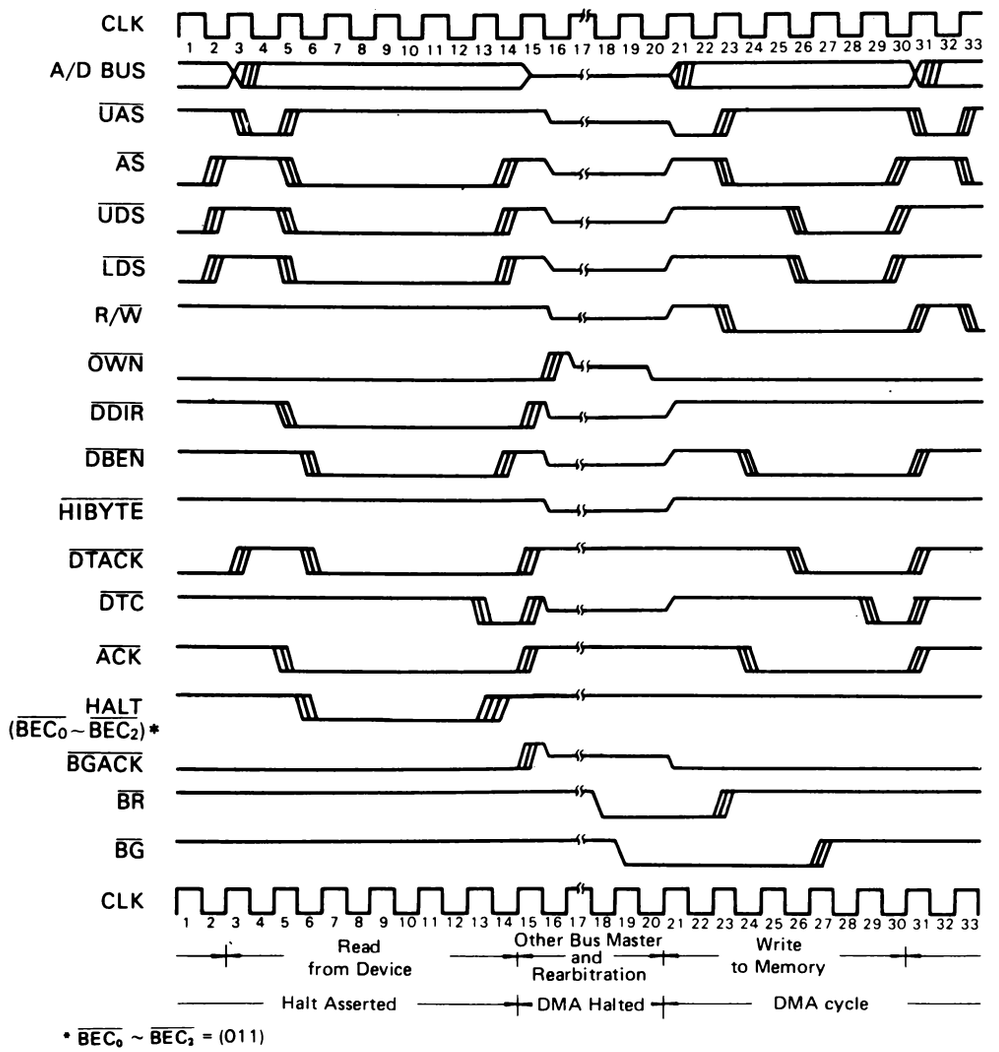
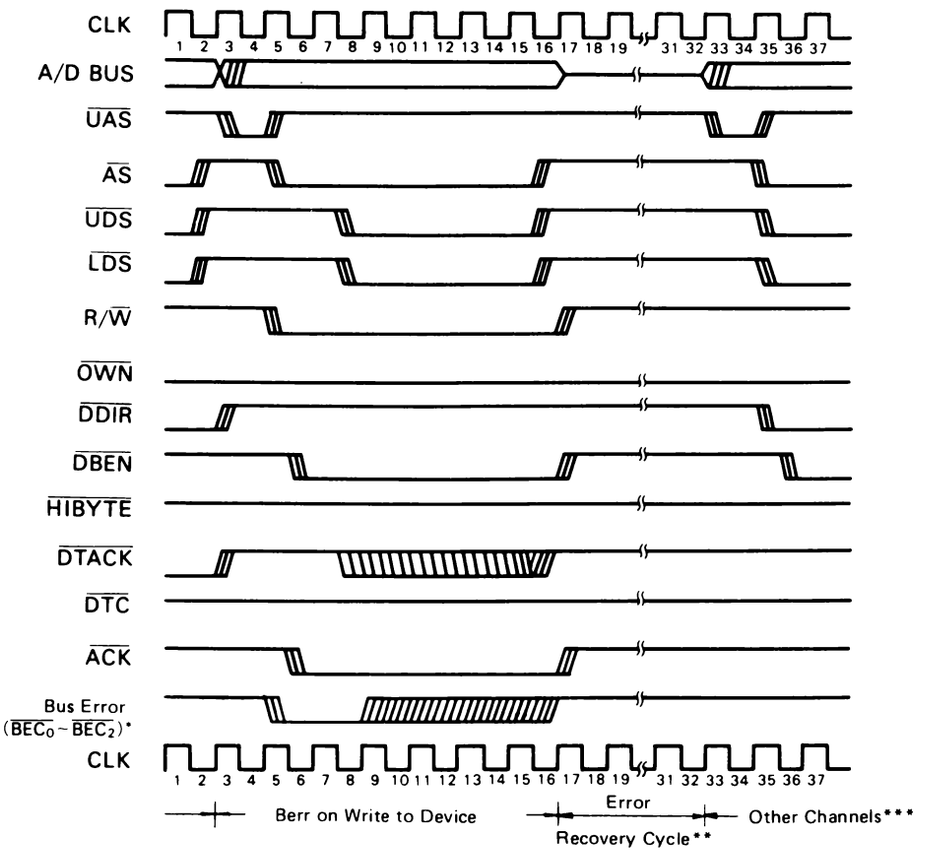


Figure 39 Halt Operation

Bus Error

The bus error exception is generated by external circuitry to indicate the current transfer cannot be successfully completed and is to be aborted. The recognition of this exception during a DMAC bus cycle signals the internal bus error condition for the channel for which the current bus cycle is being run. As soon as the DMAC recognizes the bus error exception, the DMAC immediately terminates the bus cycle and proceeds to the error recovery cycle. In this cycle, the DMAC adjusts the values of the MAR, the DAR, the MTC and the BTC to the values when the bus error exception occurred. 25 clocks are required for the error recovery cycle in the single addressing mode and in the read cycle of the dual addressing mode. 29 clocks are required in the write cycle of the dual addressing mode. If the DMAC does not have any transfer request in the other channels after the error recovery cycle, the DMAC relinquishes the bus.

The diagram of the bus error timing is shown in Figure 40.



* $\overline{BEC_0}$-$\overline{BEC_2}$ = (101)
** In the single addressing mode and in the read cycle of the dual addressing mode: 25 clocks
In the write cycle of the dual addressing mode: 29 clocks
*** The DMAC keeps the bus because the other channels have requests pending. If other channels do not have requests, the DMAC relinquishes the bus after the error recovery cycle.

Figure 40 Bus Error Operation

Retry

The retry exception causes the DMAC to terminate the present operation and retry that operation when retry is removed, and thus will not honor any requests until it is removed. However, the DMAC still recognizes requests. The retry timing is shown in Figure 41.

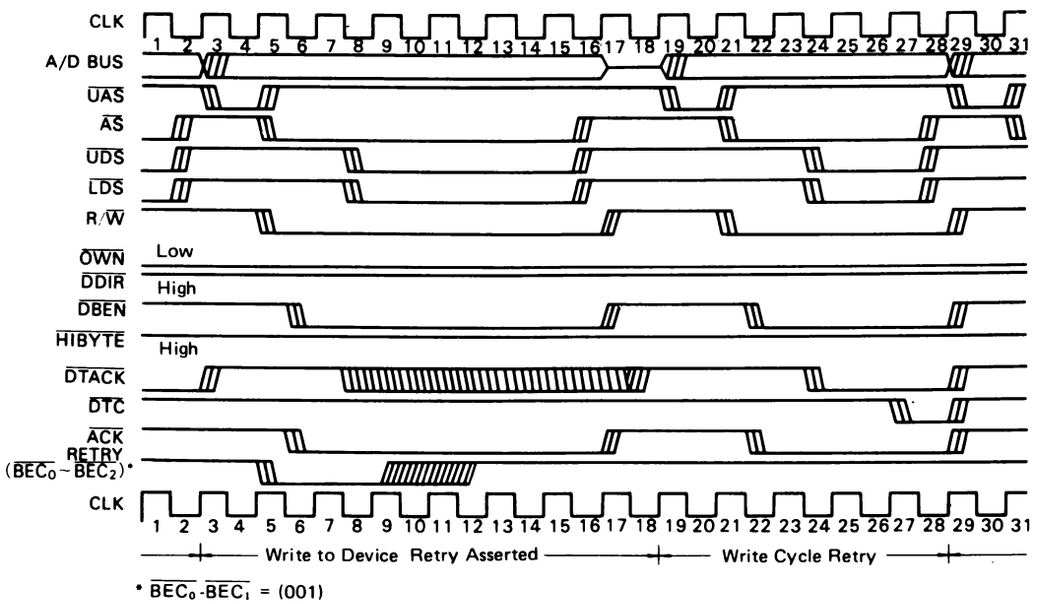


Figure 41 Retry Operation

Relinquish and Retry (R&R)

The relinquish and retry exception causes the DMAC to relinquish the bus and three-state all bus master controls and when the exception is removed, rearbitrate for the bus to retry the previous operation.

The diagram of the relinquish and retry timing is shown in Figure 42.

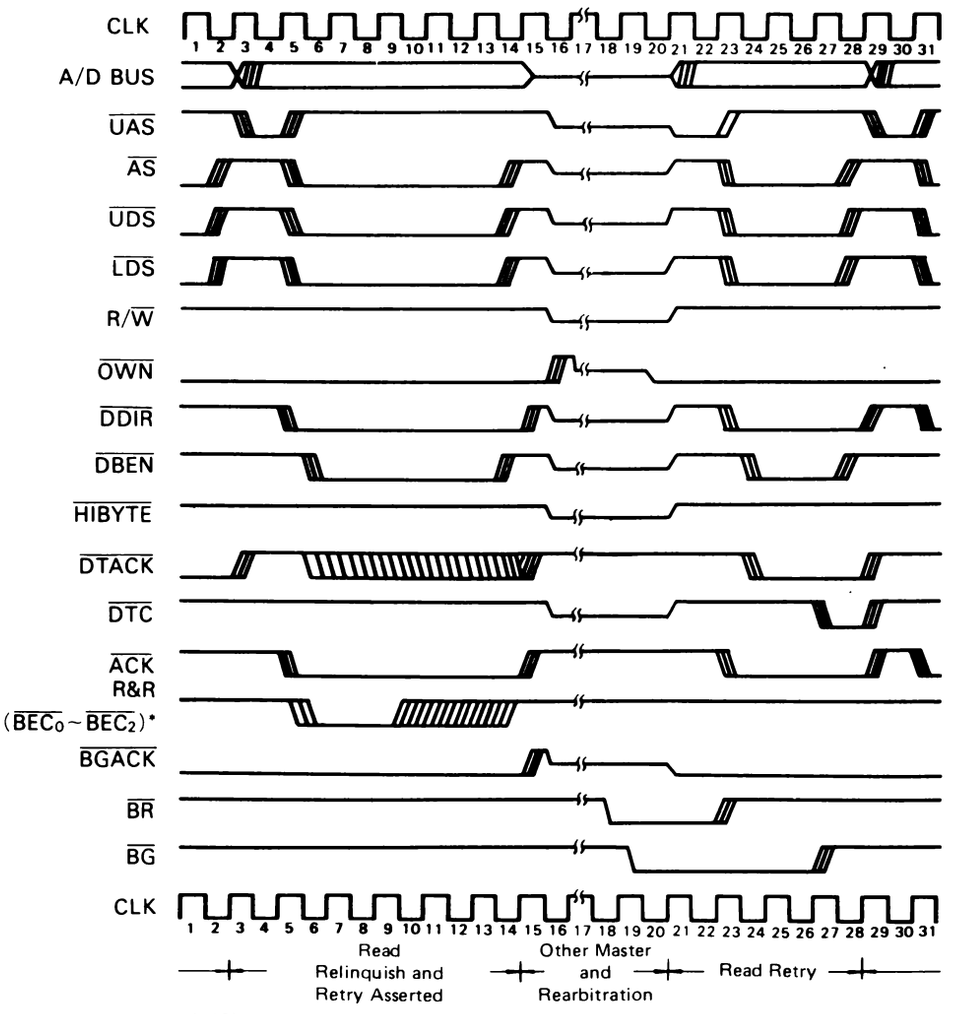


Figure 42 Relinquish and Retry Operation

Reset

The reset provides a means of resetting and initializing the DMAC. If the DMAC is bus master when the reset is asserted, the DMAC relinquishes the bus. Reset clears GCR, DCR, OCR, SCR, CCR, CSR, CPR, and CER for all channels. The NIV and the EIV are all set to $(0F)_{16}$, which is the uninitialized interrupt vector number for the HD68000 MPU. MTC, MAR, DAR, BTC, BAR, MFC, DFC, and BFC are not affected. In order to insure a reset, $\overline{BEC_0} \sim \overline{BEC_2}$ must be kept at "Low" level for at least ten clocks.

(5) Error Conditions

When an error is signaled on a channel, all activity on that channel is stopped. The ACT bit of the CSR is cleared, and the COC bit is set. The ERR bit of the CSR is set, and the error code is indicated in the CER. All pending operations are cleared, so that both the STR and CNT bits of CCR are cleared.

Enumerated below are the error signals and their sources.

(a) Configuration Error – This error occurs if the STR bit is set in the following cases.
  (i) the CNT bit is set at the same time STR bit in the chaining mode.
  (ii) DTYP specifies a single addressing mode, and the device port size is not the same as the operand size.
  (iii) DTYP specifies a dual addressing mode, DPS is 16 bits, SIZE is 8 bits and REQG is "10" or "11".
  (iv) an undefined configuration is set in the registers. The undefined configurations are: XRM = 01, MAC = 11, DAC = 11, CHAIN = 01, and SIZE = 11.

(b) Operation Timing Error – An operation timing error occurs in the following cases:
  (i) when the CNT bit is set after the ACT bit has been set by the DMAC in the chaining mode, or when the STR and the ACT bits are not set.
  (ii) the STR bit is set when ACT, COC, BTC, NDT or ERR is set.
  (iii) an attempt to write to the DCR, OCR, SCR, MAR, DAR, MTC, MFC, or DFC is made when the STR bit or the ACT bit is set.
  (iv) an attempt to set the CNT bit is made when the BTC and the ACT bits are set.

(c) Address Error – An address error occurs in the following cases:
  (i) an odd address is set for word or long word operands.
  (ii) $\overline{CS}$ or $\overline{IACK}$ is asserted during the DMA bus cycle.

(d) Bus Error – Bus error occurs when a bus error excep-

tion is signaled during a DMA bus cycle.

(e) Count Error – A count error occurs in the following cases:
  (i) The STR bit is set when zero is set in the MTC and the MTC and the chaining mode is not used.
  (ii) the STR bit is set when zero is set in BTC for the array chaining mode.
  (iii) zero is loaded from memory to the BTC or the MTC in the chaining modes or the continue mode.

(f) External Abort – External abort occurs if an abort is asserted by the external circuitry when the $\overline{PCL}$ line is configured as an abort input and the STR or the ACT bit is set.

(g) Software abort – Software abort occurs if the SAB bit is set when the STR or the ACT bit is set.

Error Recovery Procedures

If an error occurs during a DMA transfer, appropriate information is available to the operating system (OS) to allow a software failure recovery operation. The operating system must be able to determine how much data was transferred, where the data was transferred to, an what type of error occurred.

The information available to the operating system consists of the present value of the Memory Address, Device Address and Base Address Registers, the Memory Transfer and Base Transfer Counters, the channel status register, the channel error register, and the channel control register. After the successful completion of any transfer, the memory and device address registers points to the location of the next operand to be transferred and the memory transfer counter contains the number of operands yet to be transferred. If an error occurs during a transfer, that transfer has not completed and the registers contain the values they had before the transfer was attempted. If the channel operation uses chaining, the Base Address Register points to the next chain entry to be serviced, unless the termination occurred while attempting to fetch an entry in the chain. In that case, the Base Address Register points to the entry being fetched. However, in the case of external abort, there are cases in which the previous values are not recovered.

Bus Exception Operating Flow

The bus exception operating flow in the case of multiple exception conditions occurring continuously in sequence is shown in Figure 43. Note that the DMAC can receive and execute the next exception condition. For example, if the retry exception occurs, and next the relinquish and retry exception occurs while the DMAC is waiting for the retry condition to be cleared, the DMAC relinquishes the bus and waits for the exception condition to be cleared. If a bus error occurs during this period, the DMAC executes the bus error exception operation.

The flow diagram of the normal operation without exception operation or errors is shown in Figure 44.

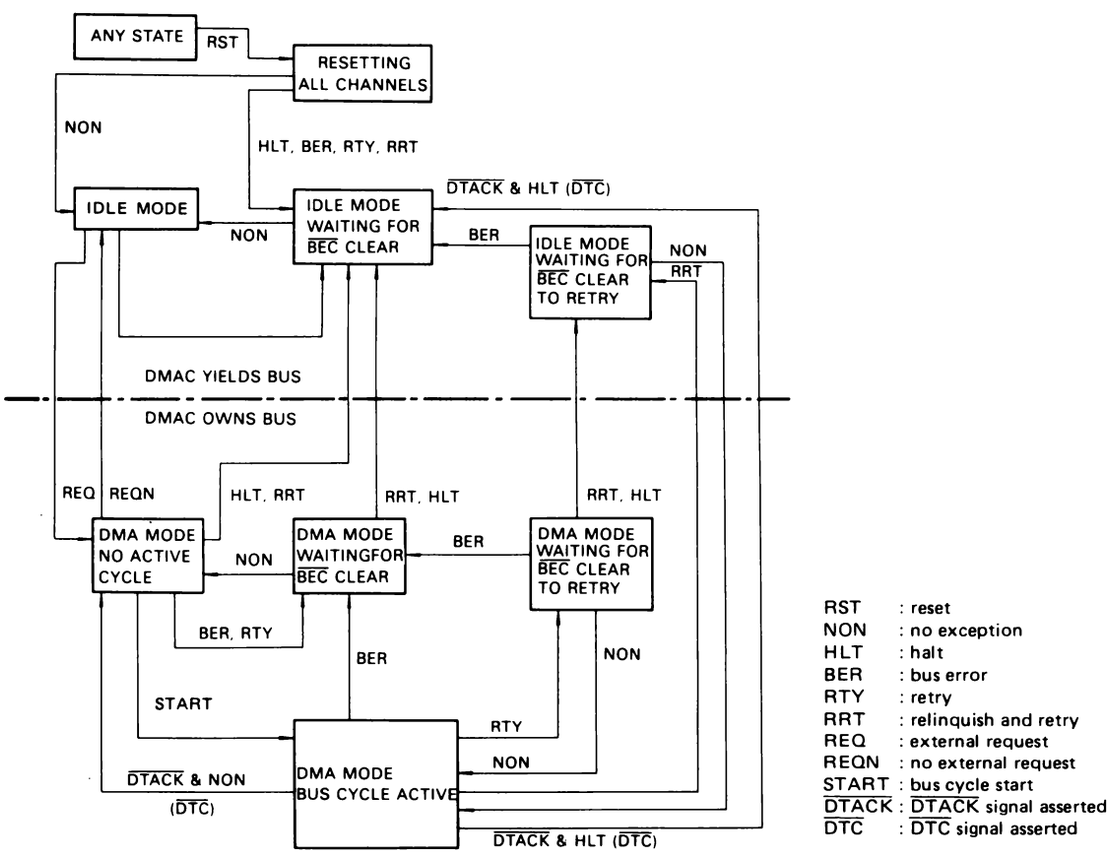


Figure 43 Bus Exception Flow Diagram

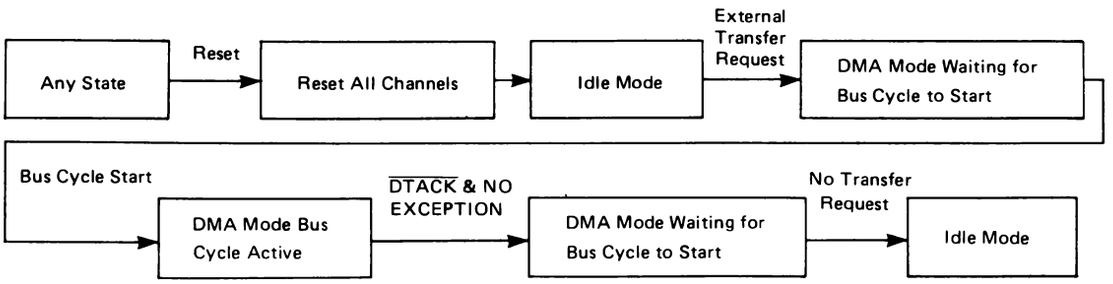


Figure 44 Flow of Normal Operation Without Exception or Error Condition

- **Channel Priorities**

Each channel has a priority level, which is determined by the contents of the Channel Priority Register (CPR). The priority of a channel is a number from 0 to 3, with 0 being the highest priority level. When multiple requests are pending at the DMAC, the channel with the highest priority receives first service. The priority of a channel is independent of the device protocol or the request mechanism for that channel. If there are several requesting channels at the highest priority level, a round-robin resolution is used, that is, as long as these channels continue to have requests, the DMAC does operand transfers in rotation.

Resetting the DMAC puts the priority level of all channels to "0", the highest priority level.

■ APPLICATIONS INFORMATION

Examples of how to interface HD68450 to a HD68000 based system are shown in Figure 45 and Figure 46.

Figure 45 shows an example of how to demultiplex the address/data bus. $\overline{OWN}$ and $\overline{UAS}$ are used to control 74LS373 for latching the address. $\overline{DBEN}$ and $\overline{DDIR}$ are used to control the bi-directional buffer 74LS245.

Figure 46 shows an example of inter-device connection in the HMCS68000 system.

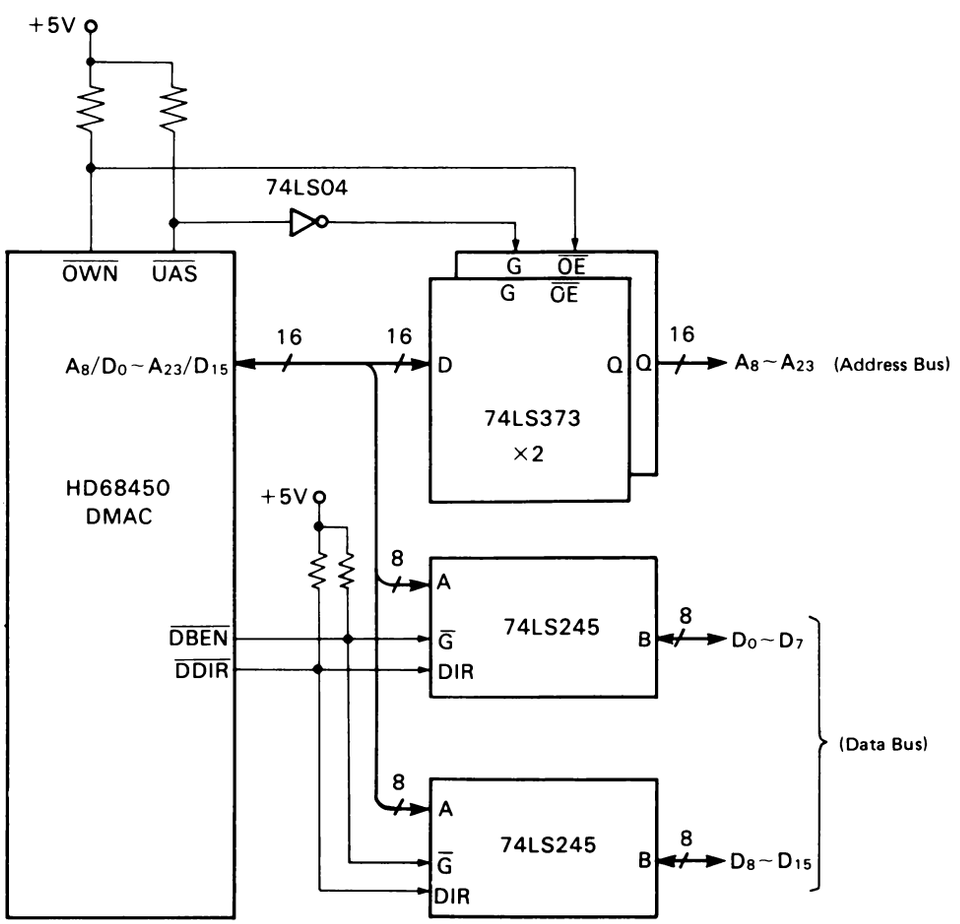
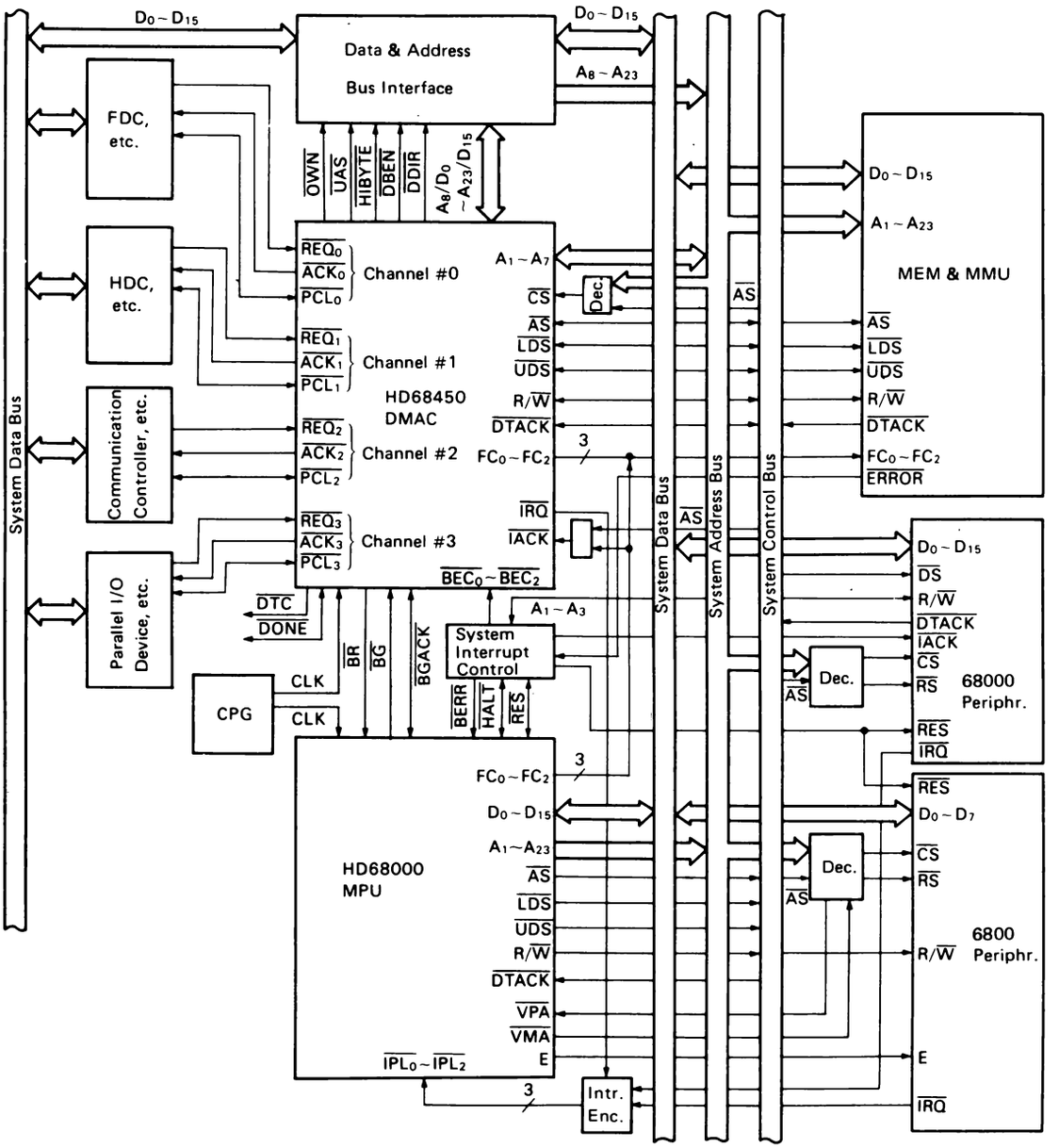


Figure 45 An Example of the Demultiplexed Address Data Bus



The address bus and the system control bus in each device are omitted in this Figure.

Figure 46 An Example of Inter-device Connection in the HMCS68000 System

■ ATTENTION ON USAGE

(1) How to interface various 6800 type peripheral devices to the DMAC based system.

When the DMAC is reading data from the 6800 device, the DMAC latches the data when $\overline{DTC}$ is asserted and not at the falling edge of E clock. The 74LS373 need to be provided externally as shown in Figure 47 so that the data from the 6800 device can be held on the bus for a large period of time until the DMAC can latch the correct data.

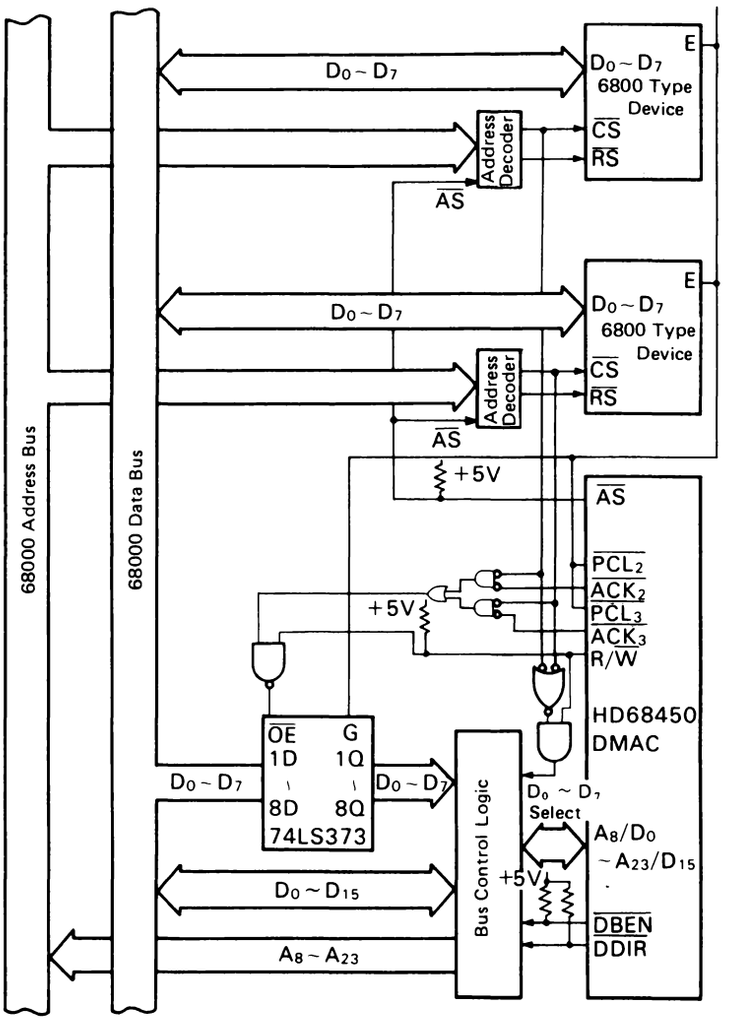


Figure 47 An Example of Connection with 6800 type Peripheral Devices (channel 2 and 3 are used)

(2) When "external abort" is inputted during the $\overline{DONE}$ input cycle

When the transfer direction is from the peripheral device to memory and $\overline{PCL}$ signal is set to the external abort input mode in the dual addressing mode, the external abort will be ignored during the subsequent write cycle from the DMAC's internal holding register to memory if $\overline{DONE}$ is inputted during the read cycle from the peripheral device to the DMAC's internal holding register.

In this case, the channel status register (CSR) and the channel error register (CER) show the normal termination caused by $\overline{DONE}$ Input. The user is able to examine the PCT bit and the ERR bit in order to detect the external abort inputted at the timing described above. If PCT = 1, ERR = 0, and NDT = 1, then an external abort has occurred.

(3) Multiple Errors

The DMAC will log the first error encountered in the channel error resister. If an error is pending in the error register and another error is encountered the second error will not be logged. Even though the second error is not logged in the CER, it will still be recognized internally and the channel will not start.

(4) Relinquish & Retry Exception During Dual Address Mode Operation

When the following two conditions occur simultaneously, incorrect data is outputted by the DMAC at the write cycle immediately following the negation of the relinquish & retry (R&R) exception.

(1) R&R is asserted at the write cycle in the dual address mode.

(2) MPU access to the DMAC's internal register is done after the DMAC relinquishes the bus due to R&R exception.

When the R&R exception occurs during the write cycle of

the dual addressing mode, and the MPU accesses the DMAC's register, then the DMAC's proper operation sequence should be the following (refer to the Fig. 48):

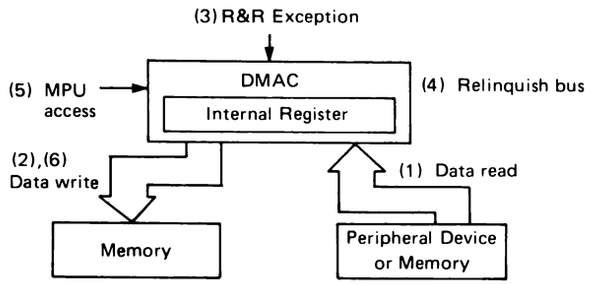


Fig. 48

(1) Data is read by the DMAC during the read cycle
(2) Data read at (1) is outputted at the write cycle
(3) R&R exception is asserted during the write cycle
(4) DMAC relinquishes the bus
(5) MPU accesses the DMAC and it is completed normally
(6) When R&R exception is negated, the DMAC obtains the bus and write cycle is done to output the data read at (1).

But instead, incorrect data is outputted at (6). This is because the data read at (1), which is held internally by the DMAC, is destroyed at (5) when the MPU accesses the DMAC. Avoid occurance of the above condition. For example:

(1) Assert R&R exception only during the read cycle when using dual addressing mode.
(2) If the R&R exception occurred at the write cycle of the dual addressing mode, avoid accessing the DMAC's registers.
(3) Use HALT exception instead of R&R exception to access the DMAC's internal registers.

(5) $\overline{CS}$ Negation Timing

When the $\overline{LDS}$, $\overline{UDS}$ high to $\overline{CS}$ high timing (chip select negation delay) is over 1 clock and the MPU access is long word (32-bit data), then the data stored in the lower word of the register accessed is destroyed and becomes all zeros. In other words, the data in the lower word of the read access is lost and cleared to all zeros. This does not always occur, but on occasional basis due to the asynchronous input timing of the CS signal.

Please observe the timing specification shown in Fig. 49.

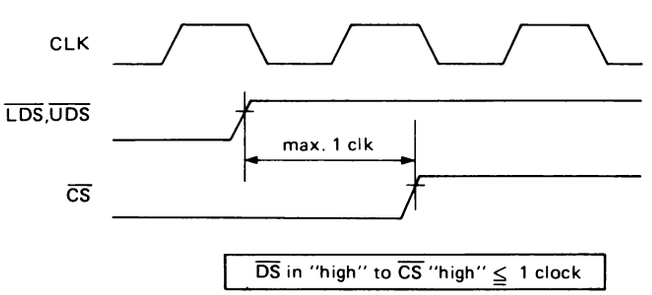


Fig. 49

(6) Unused Function Code (FC_0–FC_2) Lines

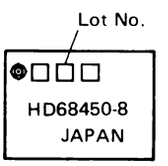
When the FC_0, FC_1, and FC_2 lines are not used, please keep these lines "high" by using a pull-up resistor. If these lines are left unconnected, the HD68450 DMAC may not operate properly.

(7) The use of thick wiring is recommended between Vss of the HD68450 and the ground of the circuit board. When a socket is used to install the DMAC on the board, please make sure that the contact of the Vss pins are made well.

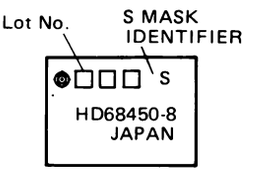
▪ DIFFERENCES AND COMPATIBILITY AMONG THE MASK VERSIONS

• Marking

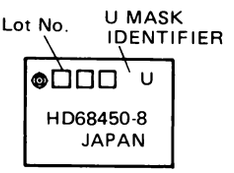
"OLD" MASK VERSION



S MASK VERSION



NEW U MASK VERSION



• The "OLD" Mask

For the "Old" mask version of the HD68450 DMAC, please refer to the following material, which may be obtained from the sales office:

Microcomputer Device/System Technical Information
No. T-026
"Anomalies of the HD68450 DMAC" Aug. 26, 1983

• The "S" Mask

With the S mask version of the HD68450 DMAC, the anomalies listed in the "Microcomputer Device/System Technical Information, No. T-026 – Anomalies of the HD68450 DMAC" (dated Aug. 26, 1983) is fixed with the exception of item #3 – "Extra data transfer in the burst mode." For the description of the S mask's anomaly, please contact the sales office.

The other remaining anomaly in the "S" mask version – one byte of transfer data is left in the DMAC.

When the DMAC is set to dual addressing mode, port size 8 bits, external request mode, and data transfer from peripheral device to memory, the last byte of the transfer may be left inside the DMAC's internal register without being transferred to memory if the transfer is stopped before the transfer count is exhausted. The last byte that is left inside the DMAC becomes unaccessable by the MPU.

In this mode, the DMAC transfers data repeating the following bus cycles:

(1) READ BYTE
(Byte is read from the peripheral device to DMAC)

(2) READ BYTE
(Byte is read from the peripheral device to DMAC)

(3) WRITE WORD
(Word is written to memory from DMAC)

If the transfer is terminated after (1) READ BYTE (see NOTE*), then the byte data that was ready by (1) READ BYTE bus cycle is not written to memory and is left inside the DMAC's internal holding register. The DMAC's internal holding register cannot be accessed by the MPU, so that it becomes "lost".

This will not occur when single addressing mode is used. So, please use the single addressing mode when the transfer needs to be terminated before the transfer is exhausted.

NOTE*: The methods to terminate the transfer operation before the transfer counter becomes zero are (1) assert external about using the $\overline{PCL}$. (2) set the SAB bit to cause software abort.

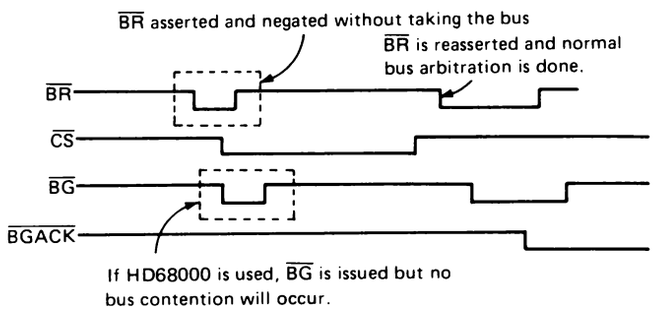
- **The New "U" Mask**

The remaining anomaly of the S mask – "Extra data transfer in the burst mode" is fixed in the U mask. In order to correct the remaining anomaly of the S mask-one byte of transfer data is left in the DMAC, the U mask has a new mode operation that repeats READ BYTE – WRITE BYTE operation in the dual addressing mode, port size 8 bits, external request mode, and data transfer from peripheral device to memory. The new mode uses the SIZE = 11 in the operation control register. In the "OLD" mask and the S mask version, SIZE = 11 causes an configuration error.

- **Bus Arbitration Problem in the S and U mask version**

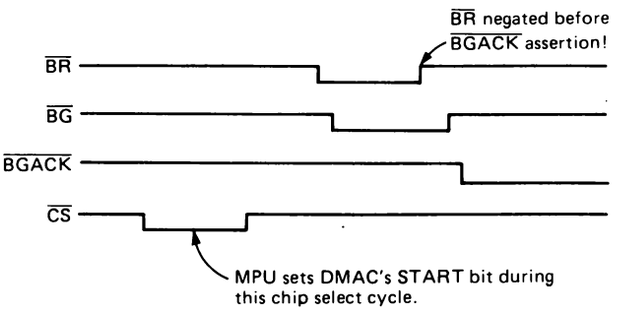
$\overline{BR}$ Negation on MPU's $\overline{CS}$

If the MPU asserts DMAC's $\overline{CS}$ when the DMAC has its $\overline{BR}$ asserted, then the DMAC negates its $\overline{BR}$. This is shown as follows:



$\overline{BR}$ negation before $\overline{BGACK}$ assertion

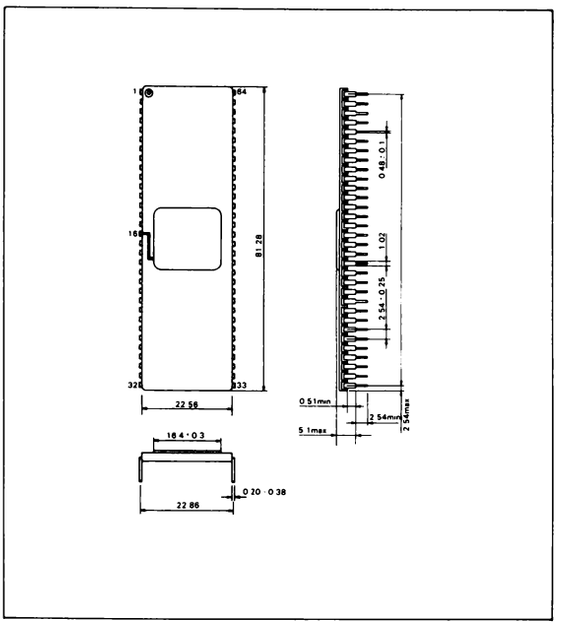
When the MPU sets the START bit of the DMAC, and a different channel is already active in the limited-rate auto-request mode, then the following bus arbitration timing may occur.



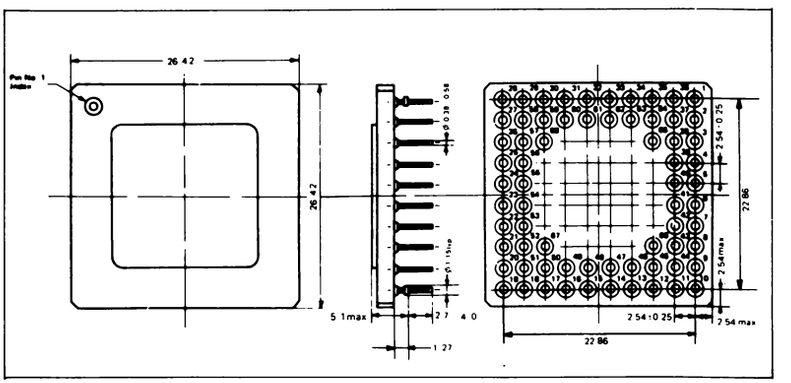
In this timing, the $\overline{BR}$ is negated too early, e.g. before the $\overline{BGACK}$ is asserted. This will cause bus contention between the DMAC and the MPU. For the description of these problems, please contact the sales office.

■ PACKAGE DIMENSIONS (Unit: mm)

● DC-64 (CERAMIC DIP)



● PGA-68 (PIN GRID ARRAY PACKAGE)



14-8 (AD-E0006A,85.2)
10/85/1 rp
Printed in W. Germany

HITACHI ELECTRONIC COMPONENTS EUROPE GMBH

Headquarter: Hans-Pinsel-Str. 10A; D-8013 Haar b. München; Tel.: (089) 46 14-0; Telex: 5-22 593 hitc d; Telefax: (089) 46 31 51

Sales Offices:

North Germany/Benelux		Breslauer Str. 6; D-4040 Neuss 1; Tel.: (0 21 01) 15 00 27-29; Telex: 8-518 039 hiec d; Telefax: (0 21 01) 10 15 13
Central Germany		Fabrikstr. 17; D-7024 Filderstadt 4; Tel.: (07 11) 77 20 11; Teletex: 7 111 410 HITECS; Telefax: (07 11) 7 77 51 16
South Germany		Hans-Pinsel-Str. 10A; D-8013 Haar b. München; Tel.: (089) 46 14-0; Telex: 5-22 593 hitc d; Telefax: (089) 46 31 51
Italy	Milano:	Via. B. Davanzati, 27; I-20158 Milano; Tel.: (02) 3 76 30 24; Telex: 323 377 hitec i; Telefax: (02) 68 37 30
	Roma:	Via Pescosolido, 154; I-00158 Roma; Tel.: (06) 4 51 01 46, -147; Telex: (06) 4 51 01 48
France		Immeuble „Les Gemeaux"; 2, rue Antoine Étex; F-94020 Créteil; Tél.: (1) 43 39 45 00; Télex: 262 246, hitec f; Téléfax: (1) 43 39 84 93

FIRMWARE

USER'S MANUAL

TABLE OF CONTENTS

TABLE OF CONTENTS Cont'd

TABLE OF CONTENTS Cont'd

LIST OF TABLES

LIST OF FIGURES

LIST OF PROGRAMMING EXAMPLES

LIST OF PROGRAMMING EXAMPLES cont'd

1. GENERAL DESCRIPTION OF THE ISCSI-1 FIRMWARE

1.1 INTRODUCTION

The ISCSI-1 Handling Firmware is an EPROM resident package which controls all ISCSI-1 features and supports execution of powerful macro I/O and initialization commands. The Handling Firmware supports the full SCSI standard either as an initiator or as a target and can be initialized to support optional SCSI commands and features.

The ISCSI-1 Handling Firmware consists of:

- I/O initialization and ISCSI-1 selftest routines.

- Modules which support SCSI initiator and target mode.

- Command chaining mode.

- Block buffering and hashing routines.

- Handling for up to four floppy disk drives

- Command execution routines for:
  - SCSIbus control,
  - Board and firmware initialization,
  - System, I/O and device status information,
  - Byte, string and block I/O,
  - Starting and executing user programs,

All programming examples in this manual are written in M680000 Assembler language under PDOS operating system.

1.2 FEATURES OF THE ISCSI-1 HANDLING FIRMWARE

The SCSIbus interface:

- Full support of the SCSI standard, either as initiator or as target.
- Optional SCSI commands can be installed for several logical units.
- Emulation of the SCSI commands COPY, COMPARE and SEARCH.
- High speed data transfer with up to 1,42 Mbytes/second.
- BACKUP command and automatic runtime backup to a secondary unit.
- Support of the RESERVE/RELEASE commands, and of the DISCONNECT/RESELECT operation.
- Automatic handling of REQUEST SENSE.
- Transparent Mode to the SCSI interface for vendor-unique commands or software debugging.

Global ISCSI-1 features:

- Supporting multi processor access via double software interface.
- The interfacing sector size between the host and the ISCSI-1 can be different from the physical sector size of the logical units.
- To upgrade the performance, 16 hashing buffers are installed.
- Up to 5 logical units are under local control and can be accessed via the host interface or the SCSIbus.
    - lun #Ø = the processor unit (CPU and DMAC)
    - lun #1 to #4 = floppy disk drives
- 68ØØØ programs can be loaded into the DPR and executed under local control.

Floppy Disk Drives:

- Supporting FORMAT, FORMAT TRACK, COMPARE, COPY and BACKUP commands.
- Disk parameters and format interleaves are all changeable.

1.3 FUNCTIONAL DESCRIPTION

After executing the startup routine, the firmware main loop is entered. The power-up procedure and the startup selftest are described in sections 1.4 and 1.5 of this manual.
The main routine polls the CMDRAMs for an executable command. When a valid command is found in one of the CMDRAMs the command will be decoded, all parameters are fetched and the command is executed.
On completing command execution, the status and command dependent parameters are returned in the CMDRAM. Optionally an interrupt will be generated.
The ISCSI-1 can work in different modes which are:

- Single command mode,
- Command chaining mode and
- SCSI target mode.

The 'SINGLE COMMAND MODE' works exactly as described above. For more information, please refer to chapters 2 and 3 of this manual.

The 'COMMAND CHAINING MODE' is a powerful feature to build command macros and to relieve the host from disk I/O jobs. It is entered with a special command from the Single Command Mode and requests subsequent sequential CMDRAMs in the command chaining buffer. Each CMDRAM in the chain buffer is handled and executed as in the Single Command Mode. For a detailed description of the Command Chaining Mode, please refer to section 2.3.

The 'SCSI TARGET MODE' can be enabled by an initialization command, and means that the ISCSI-1 controller is selectable from the SCSIbus as TARGET (i.e. SCSIbus slave). This mode is very useful for system interconnection. Please, refer to chapter 3 of this manual for a detailed description of the Target Mode.

To increase the performance of the ISCSI-1 on read/write operations, data buffering and hashing is implemented in the following manner:

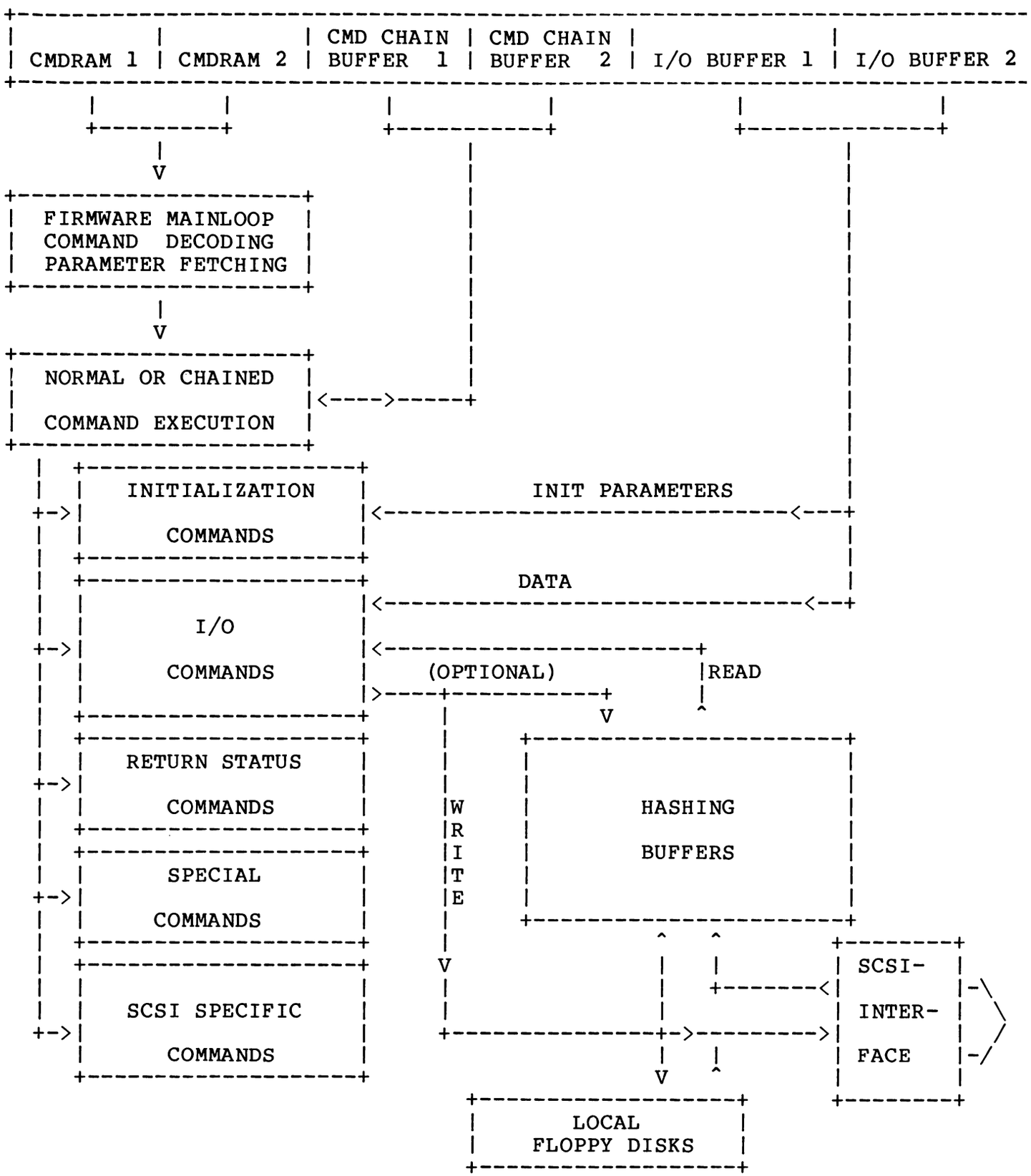
- Hashing is enabled on default for all SCSI devices on any READ - operation. It can be disabled/enabled for each target seperately.

- For the floppy disk drives, hashing is disabled on default but can be enabled for each disk seperately by using the SUNPARM (set units parameters) command.

- For all WRITE - operations, hashing is disabled and can be enabled only GLOBAL for all units.

- On system shut down the FLUSH modified buffers command should be used to make sure that all data is stored to disks!

Figure 1 : Logical Block Diagram of the ISCSI-1 Firmware



1.4 THE POWER-UP PROCEDURE

After power-on or reset, the firmware EPROM is down mapped to address location Ø upward. The CPU fetches the initial stack-pointer and start address and starts the firmware program.

First of all, the ISCSI-1 firmware performs a local selftest which tests the local and dual ported memory, the DMA Controller the SCSI controller and the floppy disk controller.

After selftest the system RAM will be cleared and initialized and the firmware sets all onboard interrupt vectors. The firmware enters the main routine and is now ready to accept commands from the Host.

1.5 THE ISCSI-1 ON-BOARD SELFTEST

On power-up or reset the ISCSI-1 selftest routine is executed in the following manner:

- Testing SCSI controller by write and read back several registers.
- Starting SCSI controller self diagnostic.
- Testing the entire local and dual ported memory with read and write bytes, words and long words.
- DMA Controller test with high speed data transfer memory to memory.
- Testing the floppy disk controller.

The control of the selftest state and result is provided via the front panel LEDs S1 through S4:

- The LED S1 is turned on during the RAM test.
- The LED S2 is turned on during the SCSIbus controller test.
- LED S3 is turned on during the floppy disk controller test.
- LED S4 is turned on during the DMA controller test.

If any error is found while selftest is active, the LED of the testphase which has generated the error would stay illuminated! After the selftest has been successfully completed, all LEDs are turned off.

EXAMPLE:

After completion of the selftest routine the LED S3 was not turned off. This state indicates a hardware problem with the WD1772 Floppy Disk controller.

1.6 ISCSI-1 DEFAULT SETUPS

Global Definitions:	ISCSI-1 SCSIbus ID	7
	Interfacing Sector Size	256 Bytes
	VMEbus Address Offset	Ø
	Data hashing for Write	disabled
SCSIbus Definitions:	Target Mode	disabled
	Special Command Sets	none
	Block size for all log. Units	512 Bytes
	Blocking Mode for Write	disabled
	Auto Backup Mode	disabled
	Auto Reserve/Release Unit	disabled
	Hashing for Read	enabled
Floppy Disk Drives:	Locked/reserved Units	none
	Sides per Disk	2
	Cylinders per Disk	8Ø
	Sectors per Cylinder	32
	Bytes per Sector	256
	Disk Density	double
	Sector Offset (BIAS)	Ø
	Format Pattern	$E5
	Sector Interleave for Format	1
	Floppy Disk Steprate	3 ms
	Hashing for Read	disabled

1.7 ISCSI-1 MEMORY LAYOUT

Table 2 : Memory Layout from Local Sight

========== BOARD BASE ==========		
$0000 - $0400	1 KB	VECTOR AREA
$0400 - $2000	3 KB	FIRMWARE WORK AREA
========== DUAL PORTED MEMORY ==========		
$2000	1 W	HOLDS THE BOARD AND THE FIRMAWARE REVISION CODE
$2100 - $210F	16 B	CMDRAM #1
$2110 - $22FF	496 B	CMD CHAIN AREA #1 = 31 CMDRAMS
$2300 - $2AFF	2 KB	I/O BUFFER #1
$2B00 - $32FF	2 KB	I/O BUFFER #2
$3300 - $34FF	512 B	CMD CHAIN AREA #2 = 31 CMDRAMS
$3500 - $350F	16 B	CMDRAM #2
$3510 - $3FFF	3 KB	RESERVED
$4000 - $1FFFF	112 KB	HASHING BUFFERS (16 BUFFERS, 7KB/BUFFER)
========== END OF RAM ==========		

Table 3 : Memory Layout from System/VMEbus Sight

```
================== ISCSI-1 BASE ADDRESS ========================

A00000 - A00FFF Reading a WORD from an EVEN address such as
                A00000 will return RESET/SYSFAIL status infor-
                mation in the bits 8,9 and 10.
                Reading or writeing a BYTE from an ODD address
                will access the ISCSI-1 BIM registers as de-
                scribed below.

A00001          1 B     BIM control register 0
A00003          1 B     BIM control register 1
A00005          1 B     BIM control register 2
A00007          1 B     BIM control register 3
A00009          1 B     BIM vector register 0
A0000B          1 B     BIM vector register 1
A0000D          1 B     BIM vector register 2
A0000F          1 B     BIM vector register 3
This structure is continued to address A00FFF

A01001-A017FF   1 B     reading performs a local inter-
     rupt (odd addresses only)

A01801-A01FFF   1 B     reading performs a local reset
     (odd addresses only)

========================== BEGIN OF DPR ========================

$A02000                   1 W     HOLDS THE BOARD AND THE FIRMWARE
                                  REVISION CODE

$A02100 - $A0210F        16 B     CMDRAM #1

$A02110 - $A022FF       496 B     CMD CHAIN AREA #1 = 31 CMDRAMS

$A02300 - $A02AFF         2 KB    I/O BUFFER #1

$A02B00 - $A032FF         2 KB    I/O BUFFER #2

$A03300 - $A034FF       512 B     CMD CHAIN AREA #2 = 31 CMDRAMS

$A03500 - $A0350F        16 B     CMDRAM #2

$A03510 - $A03FFF         3 KB    RESERVED

$A04000 - $A01FFFF      112 KB    HASHING BUFFERS
                                  (16 BUFFERS, 7KB/BUFFER)

========================= END OF RAM AREA ======================
```

2. THE ISCSI-1 SOFTWARE INTERFACE

2.1 GENERAL INFORMATION

External software can access and use the ISCSI-1 firmware via DPR located RAM structures called Command RAM Areas (CMDRAM), command chaining buffers and general purpose I/O and program buffers. The CMDRAMs are described in chapter 2.2, command chaining buffers are described in chapter 2.3 and the general perpose I/O buffers in chapter 2.4.
All firmware interface structures exist twice. This is implemented for easy multi processor system support. In addition in each CMDRAM, a word location is reserved for interprocessor communication (access with the M68000 'TAS' instruction).

2.2 THE COMMAND RAM LAYOUT

Each of the two CMDRAMs is defined as a 16 byte DPR located structure. Parameters pass through the CMDRAM on input, status or error returns, and return parameters pass after command completion.

CMDRAM Layout for input:

Offset	Size	Meaning
00	W	ISCSI-1 Command and Target ID
02	W	Command Dependent Parameter
04	L	Pointer Data or Parameter block
08	L	Logical Block address
12	W	Logical Unit
14	W	Reserved for inter-processor communication

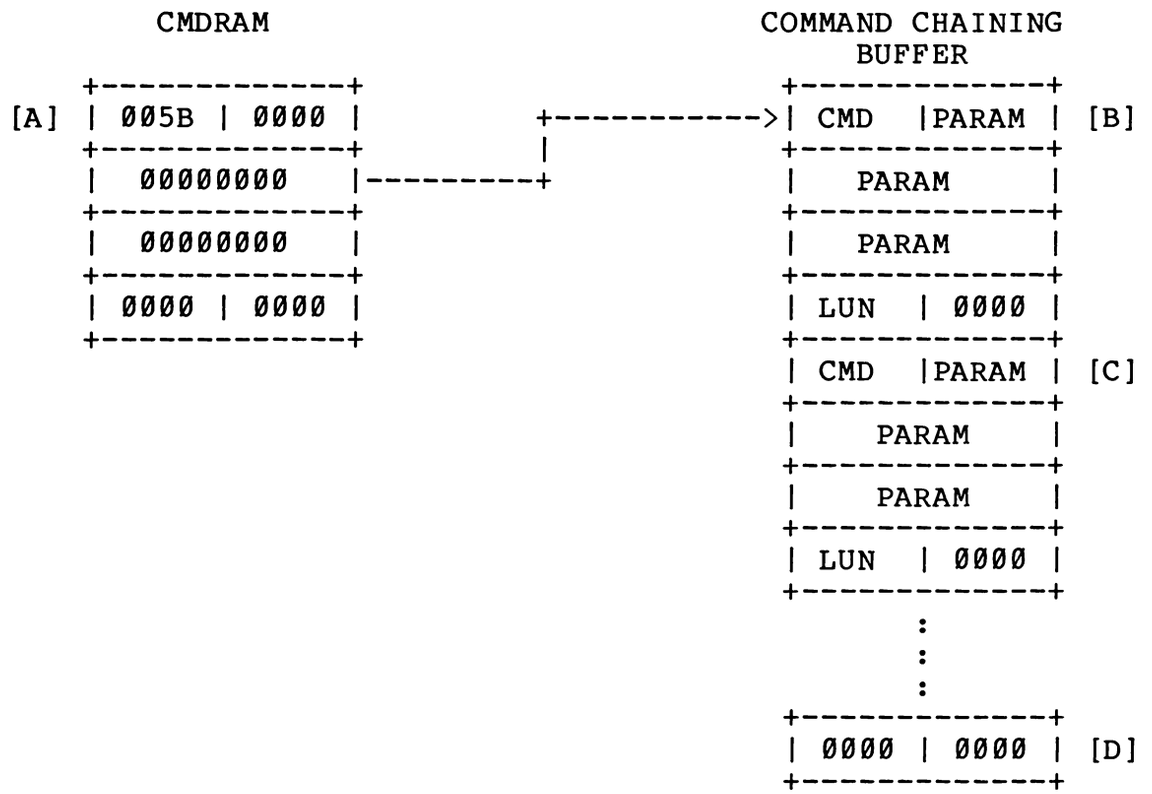
CMDRAM Layout on Command completion:

Offset	Size	Meaning
00	W	Status/Error Return Code
02	W	Extended Error Message or Data after Byte I/O
04	L	Pointer to Data or Return Values
08	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for Inter-Processor Communication

2.3 COMMAND CHAINING

A multiple command execution option is built into the ISCSI-1 firmware. All command RAM blocks that should be executed have to be entered into a command queue. After the last command RAM a long word $ØØ must be entered. This detects the end of the queue and the firmware returns into normal single command mode. The location of the command queue is requested in the command chaining buffer of the corresponding channel. The queue can be longer than the size of the buffer but must begin at the buffer start address. Each one of the commands in the queue is executed in the same way as in the single command mode. The only difference is the return value handling.
Error and completion codes are placed into the current command ram in the queue. In addition for the case that an error occurs, an error code is placed into the root command RAM and a pointer to the error command RAM is returned. After execution of each command in the queue and when the whole sequence has been completed optional interrupts can be generated.

Figure 2 : Command Chaining



[A] - CHAIN command

[B] - First CMDRAM to chain

[C] - Second CMDRAM to chain

[D] - End of CHAIN list

2.4 SPECIAL RAM LOCATIONS

Board type and firmware version number

At the board relative address $2000 (VMEbus $A02000), the board version and the firmware revision are stored in coded form.

```
first byte  : ISCSI version      - 11 = ISCSI-1
                                   12 = ISCSI-2

second byte : firmware revision - 10 = REV 1.0
                                  11 = REV 1.1
                                  and so on
```

Target message info area

This area is located at board relative address $20F0 (VMEbus $A020F0) and contains information about any data that were received from an initiator during the board is selected as a SCSI target. The message area has the following format:

Offset	Size	Meaning
0	W	contains zero if no message was received becomes negative after receiving data
2	W	contains the initiator ID
4	L	pointer to the received data
8	W	number of received 256 byte blocks

3. ISCSI-1 COMMAND DESCRIPTION

3.1 THE COMMAND STRUCTURE AND PHILOSOPHY

The ISCSI-1 commands are structured as general purpose command words which are compatible to other FORCE intelligent I/O controllers (e.g. ISIO-1). This supports easy design and installation of global I/O drivers.

Each command is a word sized code of the following format:

Bit#	Name	Explanation
15	Command/Status Flag	This flag is reset if the code is a command, and set if it is a status/error return code.
14-13		Reserved for future use
12	Interrupt enable	When the bit is set, the ISCSI-1 firmware generates an interrupt after operation complete. The vectors for normal and error interrupts can be set by programming the BIM.
11-9	Target ID	These three bits hold the SCSI-bus Target ID (can be Ø - 7).
8		don't care (only for compatibility with the ISIO-1/2 command set).
7-4	Command Groups	These bits classify up to 16 groups of commands. Only seven of them are implemented: - Initialization commands [Ø] - Byte I/O commands [1] - Block I/O commands [2] - String I/O commands [3] - Status commands [4] - Special commands [5] - SCSI specific commands [6] - Reserved [7] to [15]
3-Ø	Commands per group	A maximum of 16 commands per group is defined.

3.2 COMMAND OVERVIEW

Synonym	Hexcode	Function
SETOFFS	ØXØ3	Set VMEbus address offset
	1XØ3	Same function but complete with interrupt.
RESET	ØXØ9	Reset the firmware to power-on state.
SUNPARM	ØXØA	Install logical unit (SCSI and local floppy disk).
	1XØA	Same function but complete with interrupt.
BLOCKING	ØXØB	Enable/disable blocking for write operations.
	1XØB	Same function but complete with interrupt.
WRPARAL	ØXØC	Enable/disable runtime backup to parallel logical unit.
	1XØC	Same function but complete with interrupt.
MAININIT	ØXØD	Controller main initialization and installation command
	1XØD	Same function but complete with interrupt.
GETCHR	Øx1Ø	Read one byte from a logical block.
	1x1Ø	Same function but complete with interrupt.
PUTCHR	Øx14	Write one byte to a logical block.
	1x14	Same function but complete with interrupt.
GETBLOC	ØX2Ø	Read fix sized blocks/sectors from a logical unit.
	1X2Ø	Same function but complete with interrupt.
PUTBLOC	ØX22	Write fix sized blocks/sectors to a logical unit.
	1X22	Same function but complete with interrupt.
GETSTR	ØX3Ø	Read a delimited string from a logical block.
	1X3Ø	Same function but complete with interrupt.
PUTSTR	ØX32	Write a delimited string to a logical block.
	1X32	Same function but complete with interrupt.
GETCNT	ØX33	Read a counted string from a logical block
	1X33	Same function but complete with interrupt.
PUTCNT	ØX35	Write a counted string to a logical block.
	1X35	Same function but complete with interrupt.

CTLSTAT	0X44	Return controller status information.
	1X44	Same function but complete with interrupt.
CHKTARG	0X45	Check SCSI targets for existence and device type.
	1X45	Same function but complete with interrupt.
GUNPARM	0X46	Return logical units parameter.
	1X46	Same function but complete with interrupt.
EXPROG	0X50	Execute user program in local DPR.
	1X50	Same function but complete with interrupt.
BACKUP	0X55	Backup a logical unit to another.
	1X55	Same function but complete with interrupt.
FLUSH	0X56	Flush all modified buffers.
	1X56	Same function but complete with interrupt.
FORMAT	0X57	Format a logical unit.
	1X57	Same function but complete with interrupt.
FTRACK	0X58	Format a physical track on a floppy disk drive.
	1X58	Same function but complete with interrupt.
COMPARE	0X59	Compare data between two logical units.
	1X59	Same function but completes with interrupt.
LOCKUN	0X5A	Lock local unit against access from SCSI-bus initiator.
	1X5A	Same function but complete with interrupt.
FREEUN	0X5B	Free local unit for SCSIbus access.
	1X5B	Same function but complete with interrupt.
COPY	0X5C	Copy data from one logical unit to another or between logical blocks on the same logical unit.
CHAIN	0X5D	Enter command chaining mode.
	1X5D	Same function but complete with interrupt.
SEARCH	0X5E	Search for data on a logical unit.
	1X5E	Same function but complete with interrupt.
TRSPMOD	0X60	Send a command in transparent mode to SCSIbus.
	1X60	Same function but complete with interrupt.
TARGMOD	0X61	Enable/disable target mode.
	1X61	Same function but complete with interrupt.

SFTIME	ØXØE	Set auto flush time interval.
	1XØE	Same function but complete with interrupt.
TARGINT	ØX63	Enable/disable target message interrupts.
	1X63	Same function but complete with interrupt.

3.3 DETAILED COMMAND DESCRIPTION

Synonym: SETOFFS - Set VMEbus Address Offset

Hex Code: $ØXØ3
$1XØ3

DESCRIPTION:

This command sets an address offset which is always added to all ISCSI's local pointer operations. Normally the pointers passed in the CMDRAM have to be board relative. This means that the default address offset is zero.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command
2	W	Unused/reserved
4	L	Address Offset
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error
8ØØ2, 9ØØ2 = Address error
8ØØ6, 9ØØ6 = Missing or invalid parameter

EXAMPLE:

Set the VMEbus address offset to the system base address of the ISCSI-1 controller.

```
SETOFFS LEA.L   ISCSI+CMDRAM,A0 ;GET COMMAND RAM ADDRESS
        MOVE.L  #ISCSI,4(A0)    ;OFFSET = BASE ADDRESS
        MOVE.W  #$0003,(A0)     ;COMMAND
@001    TST.W   (A0)            ;WAIT UNTIL READY
        BPL.S   @001
        XEXT                    ;EXIT TO SYSTEM
*
        END
```

Synonym: RESET - Reset the Firmware to the Power-on State

Hex Code: $0X09

DESCRIPTION:

This command restarts the firmware and reinitializes the I/O devices as they are after power-on or reset.
During the RESET procedure SYSFAIL is active and can be tested at address BASE+$0000;EVEN.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

NONE

EXAMPLE:

We restart the firmware:

```
RESET   LEA.L   ISCSI+CMDRAM,A0 ;GET CMDRAM ADDRESS
        LEA.L   ISCSI,A1       ;BASE ADDRESS
        MOVE.W  #$0009,(A0)    ;COMMAND
@001    TST.B   (A1)           ;WAIT UNTIL SYSFAIL IS RESETTED
        BPL.S   @001
        XEXT                   ;EXIT TO SYSTEM
*
        END
```

Synonym: SUNPARM - Set Unit Parameters

Hex Code: $0X0A
$1X0A

DESCRIPTION:

This command enables the user to install different SCSIbus devices and to change the parameters of the local floppy disk drives.
Changeable options are:
- SCSI command set,
- SCSI logical units block size
- Hashing enable/disable for each logical unit
- Target device type,
- Floppy disk parameters

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command/Target ID
2	W	Unused/reserved
4	L	Pointer to Parameter Block
8	L	Unused/reserved
12	W	Logical Unit Number
14	W	Reserved for interprocessor communication

PARAMETER BLOCK

The Parameter Block must be located in the I/O buffer area. To install a floppy disk the Target ID must be the own ID of the ISCSI-1 controller (default = 7).

1.) SCSI DEVICE INSTALLATION

Offset	Size	Meaning
00	L	Command set Group 0
04	L	Command set Group 1
08	B	SCSI Device Type
09	B	Logical Block Size 00 = 256 bytes/block 01 = 512 bytes/block 02 = 1024 bytes/block 03 = 2048 bytes/block
10	B	Hashing mode 00 = disable hashing 01 = enable hashing

Command set Installation:

For each SCSI device there exists up to two command groups with 32 opcodes per group. For each of these opcodes is a bit reserved in the parameter block. When the target responds to this command the bit should be set, otherwise it must be cleared.

SCSI Device Types:

$Ø0 - Direct access device (i.e. magnetic disks)
$Ø1 - Sequential access device(i.e. tapes)
$Ø2 - Printer device
$Ø3 - Processor device
$Ø4 - Write once read multiple device
$Ø5 - Read only direct access device

Device block size installation:

The specified block size is the block size used by the logical unit and may differ from the system or interfacing block size.

Hashing mode installation:

The block buffering and hashing can be installed for each target seperately. The default for SCSI devices is hashing enabled.

2.) LOCAL FLOPPY DISK INSTALLATION

Offset	Size	Meaning
Ø0	B	Number of Heads
Ø1	B	Number of Cylinders
Ø2	B	Sectors per Cylinder
Ø3	B	Sector offset to first used sector
Ø4	B	Bytes per sector Ø = 128 bps 1 = 256 bps 2 = 512 bps 3 = 1Ø24 bps
Ø5	B	Disk density Ø = single density 1 = double density
Ø6	B	Format Pattern
Ø7	B	Sector interleave table Ø-7 or new table 8
Ø8	B	Lowest sector number on each physical track (Ø or 1)
Ø9	B	Hashing mode ØØ = disable Ø1 = enable
1Ø	B	Start of sector interleave table when new table is asserted. (The length depends on the number of sectors)

Predefined Sector Interleave Tables:

```
Ø : 1,2,3,4,5,6,7,8,9,1Ø,11,12,13,14,15,16
1 : 1,3,5,7,9,11,13,15,2,4,6,8,1Ø,12,14,16
2 : 1,4,7,1Ø,13,16,2,5,8,11,14,3,6,9,12,15
3 : 1,5,9,13,2,6,1Ø,14,3,7,11,15,4,8,12,16
4 : 1,6,11,16,5,1Ø,15,2,7,12,3,8,13,4,9,14
5 : 1,7,13,2,8,14,3,9,15,4,1Ø,16,5,11,6,12
6 : 1,8,15,2,9,16,3,1Ø,4,11,5,12,6,13,7,14
7 : 1,9,2,1Ø,3,11,4,12,5,13,6,14,7,15,8,16
```

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

```
8ØØØ, 9ØØØ = Successful, no error
8ØØ2, 9ØØ2 = Address error
8ØØ6, 9ØØ6 = Missing or invalid parameter
```

EXAMPLES:

1.) LOCAL FLOPPY DISK INSTALLATION

```
SPARM1  LEA.L   SCSI+CMDRAM,A0  ;GET COMMAND RAM ADDRESS
        LEA.L   SCSI+IOBUF1,A1  ;GET I/O BUFFER #1
        LEA.L   FDCDEF(PC),A2   ;NEW FLOPPY DISK DEFINITIONS
        MOVE.L  #TABLEN,D0      ;GET TABLE LENGTH
@001    MOVE.B  (A2)+,(A1)+     ;TRANSFER DEFINITIONS
        SUBQ.L  #1,D0
        BNE.S   @001
        MOVE.L  #IOBUF1,4(A0)   ;POINTER TO DEFINITIONS
        MOVE.W  #1,12(A0)       ;FLOPPY DISK #1
        MOVE.W  #$0E0A,(A0)     ;COMMAND, ID = 7
@002    TST.W   (A0)            ;WAIT UNTIL READY
        BPL.S   @002
        XEXT                    ;DONE
*
FDCDEF  DC.B    2               ;TWO HEADS
        DC.B    80              ;80 CYLINDERS
        DC.B    8               ;8 SECTORS PER TRACK
        DC.B    16              ;SECTOR OFFSET TO FIRST TRACK
        DC.B    2               ;512 BYTES PER SECTOR
        DC.B    1               ;DOUBLE DENSITY
        DC.B    $E5             ;FORMAT PATTERN
        DC.B    8               ;OWN INTERLEAVE
        DC.B    1               ;FIRST SECTOR NUMBER
        DC.B    1,4,7,2,5,8,3,6 ;SECTOR INTERLEAVE TABLE
TABLEN  EQU     17              ;TABLE LENGTH
        END
```

2.) SCSI TARGET INSTALLATION

```
SPARM2  LEA.L   SCSI+CMDRAM,A0  ;GET COMMAND RAM ADDRESS
        LEA.L   SCSI+IOBUF1,A1  ;GET I/O BUFFER #1
        LEA.L   TARGDEF(PC),A2  ;NEW SCSI TARGET DEFINITIONS
        MOVE.L  #TABLEN,D0      ;GET TABLE LENGTH
@001    MOVE.B  (A2)+,(A1)+     ;TRANSFER DEFINITIONS
        SUBQ.L  #1,D0
        BNE.S   @001
        MOVE.L  #IOBUF1,4(A0)   ;POINTER TO DEFINITIONS
        MOVE.W  #1,12(A0)       ;LUN #1
        MOVE.W  #$040A,(A0)     ;COMMAND, ID = 2
@002    TST.W   (A0)            ;WAIT UNTIL READY
        BPL.S   @002
        XEXT                    ;DONE
*
* DEVICE SUPPORTS ALL COMMANDS
*
TARGDEF DC.L    %11010100101100011111111111111110
        DC.L    0               ;NO GROUP 1 COMMANDS
        DC.B    1               ;SEQUENTIAL ACCESS DEVICE
        DC.B    0               ;BLOCK SIZE = 256 BYTES
TABLEN  EQU     10              ;TABLE LENGTH
        END
```

Synonym: BLOCKING - Enable/disable blocking mode

Hex Code: $000B
$100B

DESCRIPTION:

This command enables or disables (depending on the current state) sector blocking for write operations.
When blocking is disabled (default) then the command enables the blocking mode. If it is enabled, the command causes a 'Flush modified Buffers' operation and blocking is disabled for further write operations.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error

Synonym: WRPARAL - Write to parallel Disks

Hex Code: $0X0C
$1X0C

DESCRIPTION:

This command enables/disables (depending on current state) the automatical runtime backup of a logical unit. When enabled, each write operation goes to the specified logical unit and its parallel logical unit.

Restriction: The logical unit and the parallel logical unit must physically be the same devices, which means they must have the same number of blocks and an identical command set. Both drives should contain the same data when this command is executed.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command/Target ID
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Logical Unit in lower byte, parallel logical Unit in upper byte
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Extended Error Code
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8006, 9006 = Missing or invalid parameter

Synonym: MAININIT - Global ISCSI-1 Initialization

Hex Code: $ØXØD
$1XØD

DESCRIPTION:

The MAININIT command enables installing and changing of global ISCSI-1 parameters and setups.

The items able to be installed are :
- ISCSI-1 own SCSIbus ID, (default = 7)
- The interfacing sector size between the ISCSI-1 and the Host, (default = 256 bytes/sector)
- The Standard String Delimiter for String I/O functions (default = $ØØ).

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command
2	W	Unused/reserved
4	L	Pointer to Parameter Block
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

PARAMETER BLOCK

The Parameter Block must be located in the I/O buffer area.

Offset	Size	Meaning
Ø	B	ISCSI-1 own SCSIbus ID
1	B	Interfacing Sector size Ø = 256 bytes/sector 1 = 512 bytes/sector 2 =1Ø24 bytes/sector 3 =2Ø48 bytes/sector
2	B	Standard String Delimiter

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter

EXAMPLE:

To set the ISCSI-1 ID to 6, the interfacing sector size to 256 bytes and the string delimiter to '$'.

```
MAININIT        LEA.L   SCSI+CMDRAM,A0  ;GET COMMAND RAM ADDRESS
                LEA.L   SCSI+IOBUF1,A1  ;GET I/O BUFFER #1
                MOVE.W  #$0600,(A1)     ;DEFINITIONS
                MOVE.B  #'$',2(A1)
                MOVE.L  #IOBUF1,4(A0)   ;POINTER TO DEFINITIONS
                MOVE.W  #$000D,(A0)     ;COMMAND
@002            TST.W   (A0)            ;WAIT UNTIL READY
                BPL.S   @002
                XEXT                    ;DONE
*
*
                END
```

Synonym: SFTIME - set flush buffers time interval

Hex Code: $ØXØE
$1XØE

DESCRIPTION:

In the buffered write mode the ISCSI-1 firmware flushes out the modified buffers after a specified time out. On default this time out is 15 seconds. The SFTIME command is used to change this interval, or to disable the function.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command
2	W	Unused/reserved
4	L	time out interval (a value of 5ØØ represents one second)
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error

EXAMPLE:

We set the auto flush timeout to 10 seconds.

```
SFTIME  LEA.L   $A02100,A0              ;GET COMMAND RAM
        MOVE.L  #5000,4(A0)             ;10 SECONDS TIMEOUT
        MOVE.W  #$0E,(A0)               ;COMMAND CODE
@001    TST.W   (A0)                    ;WAIT FOR COMPLETION
        BPL.S   @001
        XEXT
        END
```

Synonym: GETCHR - Read one Byte from a logical Block

Hex Code: $0x10
$1x10

DESCRIPTION:

The GETCHR command returns in the CMDRAM a data byte from a logical block. The byte is addressed by the logical block address and an offset from the block base to the byte.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command/Target ID
2	W	Unused/reserved
4	L	Logical Block Address
8	L	Offset from Block base to the Byte
12	W	Logical Unit
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Data byte in the lower byte or extended error message
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message,(refer to Appendix A for detailed Information)

EXAMPLE:

We read the character at position 122 from logical block 2635 on logical unit 1 which is connected to target controller 4. The programming example is designed as a subroutine.

```
                .
                .
                .
        MOVE.W  #4,DØ           ;TARGET ID
        MOVE.W  #1,D1           ;LUN#
        MOVE.L  #2635,D2        ;BLOCK ADDRESS
        MOVE.L  #122,D3         ;CHARACTER POSITION
        BSR.L   GETCHR          ;CALL THE ROUTINE
                .
                .
                .
*
*****************************************************************
*
*       IN :    DØ.W = TARGET ID
*               D1.W = LOGICAL UNIT
*               D2.L = LOGICAL BLOCK ADDRESS
*               D3.L = CHARACTER OFFSET
*
*       OUT:    DØ.B = CHARACTER
*
*
GETCHR  MOVE.L  AØ,-(A7)        ;SAVE AØ
        LEA.L   ISCSI+CMDRAM,AØ ;GET COMMAND RAM POINTER
        MOVE.L  D2,4(AØ)        ;LOGICAL BLOCK
        MOVE.W  D1,12(AØ)       ;LOGICAL UNIT
        MOVE.L  D3,8(AØ)        ;CHARACTER OFFSET
        LSL.W   #8,DØ           ;ADJUST ID
        ORI.W   #$ØØ1Ø,DØ       ;GETCHR COMMAND
        MOVE.W  DØ,(AØ)
@ØØ1    TST.W   (AØ)            ;WAIT UNTIL READY
        BPL.S   @ØØ1
        MOVE.W  2(AØ),DØ        ;GET THE CHARACTER
        MOVE.L  (A7)+,AØ        ;RESTORE REGISTER
        RTS                     ;RETURN TO CALLER
*
*
        END
```

Synonym: PUTCHR - Write one or two Bytes to a logical Block

Hex Code: $ØX14
$1X14

DESCRIPTION:

This command writes up to two data bytes to a logical block. The location where the bytes are written is addressed with the logical block address and an offset to the write position in the logical block. The lower byte of the data is written first the second byte is only written if it is unequal to zero.

NOTE: This command works only when the Blocking Mode is enabled and the accessed block is resident in any hashing buffer!

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command/Target ID
2	W	Data Byte(s)
4	L	Logical Block Address
8	L	Offset from Block Base to write Position
12	W	Logical Unit
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Extended Error Message
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error
8ØØ6, 9ØØ6 = Missing or invalid parameter
8ØØF, 9ØØF = Extended Error Message,(Refer to Appendix A for detailed Information)

EXAMPLE:

We write a byte to position 100 from logical block 2000 on the logical unit 1 which is connected to target controller 4. The programming example is designed as a subroutine.

```
                .
                .
                .
        MOVE.W  #4,D0           ;TARGET ID
        MOVE.W  #1,D1           ;LUN#
        MOVE.L  #2000,D2        ;BLOCK ADDRESS
        MOVE.L  #100,D3         ;CHARACTER POSITION
        MOVE.W  #$00FF,D4       ;DATA
        BSR.L   PUTCHR          ;CALL THE ROUTINE
                .
                .
                .
*
*****************************************************************
*
*       IN :    D0.W = TARGET ID
*               D1.W = LOGICAL UNIT
*               D2.L = LOGICAL BLOCK ADDRESS
*               D3.L = CHARACTER OFFSET
*               D4.W = CHARACTER(S)
*
*       OUT:    D0.L = RETURN CODE
*
PUTCHR  MOVE.L  A0,-(A7)        ;SAVE A0
        LEA.L   ISCSI+CMDRAM,A0 ;GET COMMAND RAM POINTER
        MOVE.L  D2,4(A0)        ;LOGICAL BLOCK
        MOVE.W  D1,12(A0)       ;LOGICAL UNIT
        MOVE.L  D3,8(A0)        ;CHARACTER OFFSET
        MOVE.W  D4,2(A0)        ;DATA BYTE(S)
        LSL.W   #8,D0           ;ADJUST ID
        ORI.W   #$0114,D0       ;PUTCHR COMMAND
        MOVE.W  D0,(A0)
@001    TST.W   (A0)            ;WAIT UNTIL READY
        BPL.S   @001
        MOVE.L  (A0),D0         ;GET RETURN STATUS
        MOVE.L  (A7)+,A0        ;RESTORE REGISTER
        RTS                     ;RETURN TO CALLER
*
*
        END
```

Synonym: GETBLOC - Read logical Blocks from a Unit

Hex Code: $0X20
$1X20

DESCRIPTION:

This function reads a number of blocks from the specified logical unit and returns a pointer to the first byte of the data. The blocks which should be read are requested as contiguous on the medium.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command/Target ID
2	W	Number of Blocks to read (max = 256)
4	L	Block address
8	L	Unused/reserved
12	W	Logical unit number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Number of Blocks which were read
4	L	Pointer to first byte of data
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message (Refer to Appendix A for detailed information)

EXAMPLE:

The program reads 4 logical blocks from the target with the ID = Ø, logical unit number Ø. The first block address is $1ØØ.

```
RDBLOCK MOVEA.L #ISCSI+CMDRAM,AØ        ;COMMAND RAM POINTER
        MOVE.W  #4,2(AØ)                ;NUMBER OF BLOCKS
        MOVE.L  #$1ØØ,4(AØ)             ;BLOCK ADDRESS
        MOVE.W  #Ø,12(AØ)               ;LOGICAL UNIT NUMBER
        MOVE.W  #$ØØ2Ø,(AØ)             ;COMMAND/TARGET ID
@ØØ1    TST.W   (AØ)                    ;READY?
        BPL.S   @ØØ1                    ;N, WAIT
        TST.B   1(AØ)                   ;Y, ERROR?
        BEQ.S   @ØØ2                    ;N
        BRA.S   ERROR                   ;Y
@ØØ2    MOVEA.L 4(AØ),AØ                ;GET DATA POINTER
        ADDA.L  #ISCSI,AØ               ;ADD BASE ADDRESS
        RTS
*
ERROR   .
        .
        .
*
        END
```

Synonym: PUTBLOC - Write logical Blocks to a Unit

Hex Code: $0X22
$1X22

DESCRIPTION:

This function writes a number of contiguous blocks to a logical unit. The blocks must reside in the ISCSI-1 onboard Dual Ported RAM (in the I/O buffers).

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command/Target ID
2	W	Number of blocks to write (max = 256)
4	L	Block address of first block
8	L	Pointer to first byte of data
12	W	Logical unit number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Number of blocks which were written
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communi-

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error message (Refer to Appendix A for detailed information)

EXAMPLE:

The program writes out 4 logical blocks starting with block address $100 to Target 0, logical unit 0.

```
WRBLOCK MOVEA.L #ISCSI+CMDRAM,A0        ;COMMAND RAM POINTER
        MOVE.W  #4,2(A0)                ;NUMBER OF BLOCKS
        MOVE.L  #$100,4(A0)             ;BLOCK ADDRESS
        MOVE.L  #IO_BUF1,8(A0)          ;DATA POINTER
        MOVE.W  #0,12(A0)               ;LOGICAL UNIT NUMBER
*
        LEA.L   BUFFER(PC),A1           ;GET DATA
        MOVEA.L #ISCSI+IO_BUF1,A2       ;GET DPR ADDRESS
        MOVE.L  #512,D0                 ;BLOCK SIZE
@001    MOVE.L  (A1)+,(A2)+             ;TRANSFER DATA
        SUBQ.L  #1,D0
        BNE.S   @001
*
        MOVE.W  #$0022,(A0)             ;COMMAND/ID
@002    TST.W   (A0)                    ;READY?
        BPL.S   @002                    ;N, WAIT
        TST.B   1(A0)                   ;Y, ERROR?
        BEQ.S   @003                    ;N
        BRA.S   ERROR                   ;Y
@003    RTS
*
ERROR   .
        .
        .
*
        END
```

Synonym: GETSTR - Read a delimited String from a logical Block

Hex Code: $0X30
$1X30

DESCRIPTION:

The GETSTR command copies a delimited string from a logical block of a unit to the I/O buffer. When the string delimiter is not found,a maximum string of the size of one block is returned. At offset 2 in the CMDRAM the string delimiter can be specified. If this parameter is set to 0, the standard delimiter (installed with MAININIT, default is $00) is used. The Parameter at offset 4 in CMDRAM holds the offset from the logical block base to the first byte of the requested string.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command/Target ID
2	W	Optional String Delimiter
4	L	First Byte Location in Block
8	L	Logical Block Address
12	W	Logical Unit Number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Extended Error Message
4	L	Pointer to String
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message (Refer to Appendix A for detailed information)

Synonym: PUTSTR - Write a delimited String to a logical Block

Hex Code: $ØX32
$1X32

DESCRIPTION:

The PUTSTR command copies a delimited string from the I/O buffer into a specified block of a logical unit. When the string delimiter is not found, a maximum string of the size of one block is copied.
At offset 2 in the CMDRAM the string delimiter can be specified. If this parameter is set to Ø, the standard delimiter (installed with MAININIT, default is $ØØ) is used. The Parameter at offset 4 in CMDRAM holds the offset from the logical block base to the first byte of the destination area. The string source location is requested at the lowest byte of one of the I/O buffers.

NOTE: This command works only when the Blocking Mode is enabled and the accessed block is resident in any hashing buffer!

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command/Target ID
2	W	Optional string delimiter
4	L	Offset in destination block to the first string location
8	L	Destination block address
12	W	Logical unit number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter

Synonym: GETCNT - Read a counted String from a logical Block

Hex Code: $ØX33
$1X33

DESCRIPTION:

The GETCNT command copies a counted string from a logical block of a unit to the I/O buffer. The string is delimited with the standard string delimiter (default = $ØØ).
The Parameter at offset 4 in CMDRAM holds the offset from the logical block base to the first byte of the requested string.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command/Target ID
2	W	String length
4	L	First Byte Location in Block
8	L	Logical Block Address
12	W	Logical Unit Number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Extended Error Message
4	L	Pointer to String
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error
8ØØ2, 9ØØ2 = Address error
8ØØ6, 9ØØ6 = Missing or invalid parameter

Synonym: PUTCNT - Write a counted String to a logical Block

Hex Code: $ØX35
$1X35

DESCRIPTION:

The PUTCNT command writes a counted string to a logical block. The first byte location of the source string is requested at the beginning of the I/O buffer 1 (or two for the second command RAM).

NOTE: This command works only when the Blocking Mode is enabled and the accessed block is resident in any hashing buffer!

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command/Target ID
2	W	String length
4	L	First Byte location in the Block
8	L	Block address
12	W	Logical unit number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error
8ØØ2, 9ØØ2 = Address error
8ØØ6, 9ØØ6 = Missing or invalid parameter

Synonym: CTLSTAT - Return General Controller Status

Hex Code: $ØX44
$1X44

DESCRIPTION:

This command returns the current status of the ISCSI-1 controller as a block of parameters.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Pointer to the Parameter Block
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

PARAMETER BLOCK

Offset	Size	Meaning
ØØ	B	ISCSI-1 own ID
Ø1	B	Interfacing sector size ØØ = 256 bytes Ø1 = 512 bytes Ø2 = 1Ø24 bytes Ø3 = 2Ø48 bytes
Ø2	B	Standard string delimiter
Ø3	B	SCSI status ØØ = disconnected Ø1 = connected as initiator Ø2 = selected as target

04	B	Target mode 00 = disabled 01 = enabled
05	B	Auto backup mode 00 = disabled 01 = enabled
06	B	Auto reserve/release unit 00 = disabled 01 = enabled
07	B	Data buffering for write 00 = disabled 01 = enabled
08	B	Locked units bits 0-4 - if 0 then unlock if 1 then locked

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter

Synonym: CHKTARG - Check for and test existing Targets

Hex Code: $ØX45
$1X45

DESCRIPTION:

This command scans the SCSIbus for all possible IDs and returns information about the existence and the type of the corresponding devices.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Pointer to return parameters
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

RETURN PARAMETER BLOCK

- The command always returns an eight byte parameter block which contains the ID status information.
- The block starts with the information of ID Ø and ends with ID 7.
- The status information byte can hold one of the following values:
  - $ØØ - direct-access device
  - $Ø1 - sequential-access device
  - $Ø2 - printer device
  - $Ø3 - processor device
  - $Ø4 - write-once read multiple device
  - $Ø5 - read-only direct-access device
  - $7F - device does not exist
  - $8Ø - unknown device type
  - $FF - ISCSI-1 own ID

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error

EXAMPLE:

```
CHECK   XPCL                              ;CR/LF
        XPCL                              ;CR/LF
        XPMC    MSTART                    ;START MESSAGE
        MOVEA.L #ISCSI+CMDRAM,A0          ;CMDRAM
@001    TST.W   (A0)                      ;BUSY?
        BPL.S   @001                      ;Y, WAIT
        MOVE.W  #$45,(A0)                 ;COMMAND
@002    TST.W   (A0)                      ;BUSY?
        BPL.S   @002                      ;Y, WAIT
        MOVEA.L 4(A0),A3                  ;GET POINTER
        ADDA.L  #ISCSI,A3                 ;ADD BOARD BASE
        MOVE.L  #0,D1                     ;COUNTER
@100    MOVE.B  (A3)+,D2                  ;GET DATA
        XPCL                              ;CR/LF
        XCBD                              ;BIN TO DEC_ASCII
        XPEL                              ;WRITE ID
        CMP.B   #$FF,D2                   ;OWN?
        BNE.S   @101                      ;N
        XPMC    MOWNID                    ;Y
        BRA.S   @300
@101    CMP.B   #$80,D2                   ;UNKNOWN DEVICE?
        BNE.S   @102                      ;N
        XPMC    MUNKN                     ;Y
        BRA.S   @300
@102    CMP.B   #$7F,D2                   ;NON EXISTING?
        BNE.S   @103                      ;N
        XPMC    MNEXIS                    ;Y
        BRA.S   @300
@103    CMP.B   #0,D2                     ;MAGNETIC DISK?
        BNE.S   @104                      ;N
        XPMC    MMAGN                     ;Y
        BRA.S   @300
@104    XPMC    MTAPE                     ;SHOULD BE STREAMER TAPE
@300    ADDQ.L  #1,D1                     ;NEXT ID
        CMP.B   #8,D1                     ;DONE?
        BLT.S   @100                      ;N, CONTINUE
        XPCL                              ;CR/LF
        XEXT                              ;EXIT TO PDOS
*
MOWNID  DC.B    '     ISCSI-1',0
MUNKN   DC.B    '     UNKNOWN DEVICE TYPE',0
MMAGN   DC.B    '     MAGNETIC DISK',0
MNEXIS  DC.B    '     NON EXISTING',0
MTAPE   DC.B    '     STREAMER TAPE',0
MSTART  DC.B    'ID    DEVICE',10,13
        DC.B    '--------------------------',0
        EVEN
        END
```

Synonym: GUNPARM - Get Unit Parameters

Hex Code: $ØX46
$1X46

DESCRIPTION:

The command is the invers of the SUNPARM (set units parameters) command and returns the installed values for the specified logical unit.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command/Target ID If the ID is the ISCSI-1 ID, the values of the floppy disk units are returned.
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Logical unit number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Pointer to return parameters
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

RETURN PARAMETER BLOCK

The parameter block format is identical to the format described at the SUNPARM command. Refer to that description for more information.

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error
8ØØ6, 9ØØ6 = Missing or invalid parameter

Synonym: EXPROG - Execute User Program in DPR

Hex Code: $0X50
$1X50

DESCRIPTION:

This command executes a program which must be loaded into the ISCSI-1 DPR area. The program should be located only in the I/O buffers and must complete with a RTS instruction.

NOTE: The program must not access any non-local addresses because the ISCSI-1 CPU can not become a VMEbus master!

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	Unused/reserved
4	L	Program start address
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter

Synonym: BACKUP - Backup a logical Unit

Hex Code: $0055
$1055

DESCRIPTION:

The command processes a backup from one logical unit to another. Before the backup starts, a flush modified buffers is done if the ISCSI-1 is in the blocking mode. The hashing buffers are destroyed because they will be used as data area during the command execution.
For the case of the destination capacity is less than the source capacity, the execution is paused at medium end and the firmware is setting the completion flag in the command RAM but does not clear the command code. In this condition a removeable medium can be changed. The execution continues after the user has reset the completion flag. If the medium end condition was not awaited the user may decide it is an error condition and can cancel the backup by clearing the command code.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command/Target ID
2	W	Unused/reserved
4	L	Source unit low word = logical unit number high word = target ID
8	L	Destination unit high word = logical unit number low word = target ID
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	[extended error message]
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message (refer to Appendix A for detailed information)

Synonym: FLUSH - Flush modified Buffers

Hex Code: $0X56
$1X56

DESCRIPTION:

This command only affects the buffered write mode and is used to save and update all modified buffers to the devices.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	[extended error message]
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message (refer to Appendix A for detailed information)

Synonym: FORMAT - Format a logical Unit

Hex Code: $0X57
$1X57

DESCRIPTION:

This command does a physical format of a logical unit. The installed parameters (SUNPARM) are used to format the unit.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command/Target ID
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Logical unit number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	[extended error message]
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message (refer to Appendix A for detailed information)

Synonym: FTRACK - Format a Track on a Floppy Disk Drive

Hex Code: $ØX58
$1X58

DESCRIPTION:

This command formats a single track on a floppy disk with the parameters given in the input parameter block.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command
2	W	Track number
4	L	Pointer to parameter block
8	L	Unused/reserved
12	W	Floppy disk number
14	W	Reserved for interprocessor communication

PARAMETER BLOCK

Offset	Size	Meaning
ØØ	B	Number of Heads
Ø1	B	Number of Cylinders
Ø2	B	Sectors per Track
Ø3	B	Dummy byte
Ø4	B	Bytes per sector Ø = 128 bps 1 = 256 bps 2 = 512 bps 3 = 1Ø24 bps
Ø5	B	Disk density Ø = single density 1 = double density
Ø6	B	Format Pattern
Ø7	B	Sector interleave table Ø-7 or new table 8
Ø8	B	Lowest sector number on each physical track (Ø or 1)
Ø9	B	Dummy byte
1Ø	B	Start of sector interleave table when new table is asserted. (The length depends on the number of sectors)

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	[extended error message]
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message (refer to Appendix A for detailed information)

Synonym: COMPARE - Compare Data between two logical Units

Hex Code: $0X59
$1X59

DESCRIPTION:

This command compares data between two logical units or between different logical blocks on the same logical unit.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	Unused/reserved
4	L	Pointer to parameter block
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

PARAMETER BLOCK

Offset	Size	Meaning
00	B	First unit - ID
01	B	First unit - logical unit number
02	L	First unit - start block address
06	B	Second unit - ID
07	B	Second unit - logical unit number
08	L	Second unit - start block address
12	W	Number of blocks to compare

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	[extended error message]
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message (refer to Appendix A for detailed information)

Synonym: LOCKUN - Lock local Unit for SCSIbus access

Hex Code: $ØX5A
$1X5A

DESCRIPTION:

The command allows the specified floppy disks to lock for SCSI bus access.
If any Initiator selects one of these locked units a "unit not ready" error will be returned.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command
2	W	Lock unit flags bit# meaning Ø lock all units 1 lock floppy disk 1 2 lock floppy disk 2 3 lock floppy disk 3 4 lock floppy disk 4 When a flag is set (1) the corresponding unit is locked
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error
8ØØ6, 9ØØ6 = Missing or invalid parameter

Synonym: FREEUN - Free local Unit for SCSIbus access

Hex Code: $ØX5B
$1X5B

DESCRIPTION:

The command allows the specified floppy disks to be freed for SCSIbus access.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command
2	W	Unlock unit flags bit# meaning Ø free all units 1 free floppy disk 1 2 free floppy disk 2 3 free floppy disk 3 4 free floppy disk 4 When a flag is set (1) the corresponding unit us unlocked.
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error
8ØØ6, 9ØØ6 = Missing or invalid parameter

Synonym: COPY - Copy data from one logical unit to another

Hex Code: $0X5C
$1X5C

DESCRIPTION:

This command copies data between two logical units or between different logical blocks on the same logical unit.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	Unused/reserved
4	L	Pointer to parameter block
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

PARAMETER BLOCK

Offset	Size	Meaning
00	B	First unit - ID
01	B	First unit - logical unit number
02	L	First unit - start block address
06	B	Second unit - ID
07	B	Second unit - logical unit number
08	L	Second unit - start block address
12	W	Number of blocks to copy

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	[extended error message]
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter
800F, 900F = Extended Error Message (refer to Appendix A for detailed information)

Synonym: CHAIN - Enter command chaining mode

Hex Code: $0X5D
$1X5D

DESCRIPTION:

This command enters the chain mode and starts executing the chain commands which are placed in the current chain buffer. For detailed description of the chain mode refer to section 2.4 of this manual.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	[extended error message]
4	L	Pointer to error command RAM in chain buffer.
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
800F, 900F = Extended Error Message

Synonym SEARCH - Search Data on a Logical Unit

Hex Code: $ØX5E
$1X5E

DESCRIPTION:

This command scans a specified number of blocks on a logical unit for a matching data pattern to the given data pattern. The return parameters are the the block number ond the offset into the block for the first found match.

PARAMETER PASSING

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command/Target ID
2	W	Length of the search pattern
4	L	Pointer to the search pattern
8	L	Unused/reserved
12	W	Logical unit number
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	[extended error message]
4	L	Block address of the pattern match
8	L	Offset into the block
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error
8ØØ2, 9ØØ2 = Address error
8ØØ6, 9ØØ6 = Missing or invalid parameter

Synonym: TRSPMOD - Handle SCSI Command in Transparent Mode

Hex Code: $0X60
$1X60

DESCRIPTION:

The TRSPMOD command transfers a SCSI command from the I/O buffer to the specified Target. The resulting parameters and the SCSI-bus state are returned in CMDRAM. Data and extended messages are returned in the I/O buffer area.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	Data transfer length (bytes)
4	L	Pointer to the Command in Buffer
8	L	Pointer to data area
12	W	Target ID
14	W	Reserved for interprocessor communication

NOTE: A preceding byte which contains the SCSI command length is required at the command block.

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	[extended error message]
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error
8002, 9002 = Address error
8006, 9006 = Missing or invalid parameter

EXAMPLE:

Send the optional command REZERO UNIT to the Target with the ID = Ø in transparent mode.

```
TRSPMOD         LEA.L   SCSI+CMDRAM,AØ  ;GET COMMAND RAM ADDRESS
                LEA.L   SCSI+IOBUF1,A1  ;GET I/O BUFFER #1
                MOVE.L  #IOBUF1,4(AØ)   ;POINTER TO DEFINITIONS
                MOVE.W  #$ØØ,2(AØ)      ;TRANSFER COUNTER = Ø
                MOVE.W  #$ØØ,12(AØ)     ;ID = Ø
                MOVE.B  #$Ø6,(A1)+      ;COMMAND LENGTH
                MOVE.B  #$Ø1,(A1)+      ;LOAD COMMAND BLOCK
                MOVE.B  #$ØØ,(A1)+
                MOVE.L  #$ØØ,(A1)+
                MOVE.W  #$ØØ6Ø,(AØ)     ;ISCSI-1 COMMAND
@ØØ2            TST.W   (AØ)            ;WAIT UNTIL READY
                BPL.S   @ØØ2
*
*
                TST.B   1(AØ)           ;ERROR?
                BEQ.S   @ØØ3            ;N
                BRA.S   ERRHDL          ;ERROR HANDLING
@ØØ3            RTS
*
                END
```

Synonym: TARGMOD - Enable/disable Controller Target Mode

Hex Code: $ØX62
$1X62

DESCRIPTION:

This command enables/disables (depends on the current state) the ISCSI-1 Target mode. This means when the Target mode is enabled, the ISCSI-1 is selectable as a Target via the SCSIbus. The Target mode can be used to connect two Host systems via the SCSIbus or to allow SCSIbus access to the floppy disks.

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
Ø	W	Command/Target ID
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
Ø	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8ØØØ, 9ØØØ = Successful, no error

Synonym: TARGINT - set target interrupt mode

Hex Code: $ØX63
$1X63

DESCRIPTION:

When the SCSI target mode is enabled and a message is received from an initiator, optional an interrupt can be generated. The message info is returned at address $20F0 (board relative) in the following format:

Offset	Size	Meaning
0	W	$0000 if no message was received, negative if a message was received from an initiator
2	W	Initiator ID
4	L	Pointer to the message block
8	W	Number of 256 byte blocks

PARAMETER PASSING:

IN: CMDRAM

Offset	Size	Meaning
0	W	Command
2	W	0 = disable target interrupts 1 = enable target interrupts
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

OUT: CMDRAM

Offset	Size	Meaning
0	W	Status/Error return code
2	W	Unused/reserved
4	L	Unused/reserved
8	L	Unused/reserved
12	W	Unused/reserved
14	W	Reserved for interprocessor communication

POSSIBLE RETURN CODES:

8000, 9000 = Successful, no error

Synonym: ABORT - Abort Command execution

Hex Code: NONE!

DESCRIPTION:

The ABORT function is not a normal ISCSI-1 command but it can be used to abort any command execution. The ABORT function is executed as an interrupt service routine which handles the PI/T port interrupt at level 6. The interrupt can be triggered by reading the address BASE + $1001 from the ISCSI-1. The Firmware terminates any command execution, reset the SCSIbus and restarts the mainloop which polls the command RAMs.

PARAMETER PASSING:

3.5 ISCSI-1 Target Mode Description

3.5.1 Overview

The ISCSI-1 supports a subset of the SCSI Common Command Set which is defined in the ANSI X3T9.2 standardization note. The commands are :

COMMAND	OPCODE
TEST UNIT READY	$ØØ
REQUEST SENSE	$Ø3
FORMAT UNIT	$Ø4
READ/RECEIVE	$Ø8
WRITE/SEND	$ØA
INQUIRY	$12

The ISCSI-1 supports up to five logical units which can be accessed from the SCSIbus:

UNIT#	DEVICE
Ø	ISCSI-1 processor device
1	Floppy disk drive 1
2	Floppy disk drive 2
3	Floppy disk drive 3
4	Floppy disk drive 4

The ISCSI-1 supports intersystem connection using the target message area to inform the host about received data. Optional interrupts can be generated when the data are transferred from the initiator (See the TARGINT command description).
The target message area has the following format:

Address	Size	Meaning
$2ØFØ	W	Contains zero if no data has been received from LUN #Ø. Becomes negative when data are received from the initator.
$2ØF2	W	Contains the initiators ID.
$2ØF4	L	This is a pointer to the received data.
$2ØF8	W	Contains the number of received 256 byte blocks.

3.5.2 Detailed description of the supported CCS commands

TEST UNIT READY

The TEST UNIT READY command returns a status byte which reflects the current state of the target.

COMMAND BLOCK:

7	6	5	4	3	2	1	0	BIT#
0	0	0	0	0	0	0	0	OPCODE
L	U	N	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	

RETURNED STATUS BYTE:

$00 - Good return, unit ready
$02 - Check condition

REQUEST SENSE

The REQUEST SENSE command requests that the target transfer sense data to the initiator.

COMMAND BLOCK:

7	6	5	4	3	2	1	0	BIT#
0	0	0	0	0	0	1	1	OPCODE
L	U	N	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	0	0	0	1	0	0	ALLOCATION LENGTH
0	0	0	0	0	0	0	0	

SENSE DATA:

The REQUEST SENSE command always returns a block of 4 bytes of information.

BYTE	MEANING
00	Sense key, error information
01	Logical block address (MSB)
02	Logical block address
03	Logical block address (LSB)

SENSE KEYS	
$00	NO SENSE.
$01	RECOVERED ERROR
$02	NOT READY
$03	MEDIUM ERROR
$04	HARDWARE ERROR
$05	ILLEGAL REQUEST
$06	not used
$07	not used
$08	not used
$09	not used
$0A	not used
$0B	ABORTED COMMAND
$0C	not used
$0D	VOLUME OVERFLOW
$0E	not used
$0F	reserved

FORMAT UNIT

The FORMAT UNIT command can be used to format one of the logical units of the ISCSI-1. If the LUN number in the command block is 1 through 4, one of the floppy disk drives will be formatted. If the LUN number is Ø, the ISCSI-1 will perform a local reset and a restart of the firmware.

COMMAND BLOCK:

7	6	5	4	3	2	1	Ø	BIT#
Ø	Ø	Ø	Ø	Ø	1	Ø	Ø	OPCODE
L	U	N	Ø	Ø	Ø	Ø	Ø	
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	

READ/RECEIVE

The READ/RECEIVE command requests a data transfer from the target to the initiator. The READ/RECEIVE affects the ISCSI-1 in subsequent ways. If the logical unit number set to zero, the ISCSI-1 memory contents are returned in blocks of 256 bytes, starting with the memory address which is specified as the block address. If the logical unit number is 1 through 4, then the specified blocks/sectors of the corresponding floppy disk drive are returned.

COMMAND BLOCK:

7	6	5	4	3	2	1	0	BIT#
0	0	0	0	1	0	0	0	OPCODE
L	U	N	X	X	X	X	X	BLOCK ADDRESS (MSB)
X	X	X	X	X	X	X	X	BLOCK ADDRESS
X	X	X	X	X	X	X	X	BLOCK ADDRESS (LSB)
X	X	X	X	X	X	X	X	NUMBER OF BLOCKS
0	0	0	0	0	0	0	0	

WRITE/SEND

The WRITE/SEND command requests a data transfer from the initiator to the target. The WRITE/SEND affects the ISCSI-1 in subsequent ways. If the logical unit number set to zero, the data are transfered to the memory address of the ISCSI-1 which is specified with the block address. The block size for this operation is 256 bytes. If the logical unit number is 1 through 4, then the specified blocks/sectors are written to the corresponding floppy disk drive.

COMMAND BLOCK:

7	6	5	4	3	2	1	0	BIT#
0	0	0	0	1	0	1	0	OPCODE
L	U	N	X	X	X	X	X	BLOCK ADDRESS (MSB)
X	X	X	X	X	X	X	X	BLOCK ADDRESS
X	X	X	X	X	X	X	X	BLOCK ADDRESS (LSB)
X	X	X	X	X	X	X	X	NUMBER OF BLOCKS
0	0	0	0	0	0	0	0	

INQUIRY

The INQUIRY command requests that information regarding parameters of the target and its attached peripheral device(s) be sent to the initiator.

COMMAND BLOCK:

7	6	5	4	3	2	1	0	BIT#
0	0	0	1	0	0	1	0	OPCODE
L	U	N	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	1	0	0	0	0	0	ALLOCATION LENGTH
0	0	0	0	0	0	0	0	

INQUIRY DATA:

BYTE	MEANING
00	Peripheral device type
01	not used, always 0
02	not used, always 0
03	reserved
04	additional data length
05 - 31	Vendor unique parameter bytes

4. THE ISCSI-1 INTERRUPT STRUCTURE

On command complete the ISCSI-1 can generate an interrupt if the command interrupt bit is set (refer to section 3.1 in this manual). The interrupt is sent to the VMEbus via the on board BIM. There are four interrupts which are supported:

BIM	Interrupt
Ø	Command successfully completed on channel 1
1	Command completed on channel 1, error was detected
2	Command successfully completed on channel 2
3	Command completed on channel 2, error was detected

The interrupt vectors and levels are programmable by programming the BIM. For detailed information how to program the BIM, please refer to the Hrdware User's Manual.

APPENDIX TO THE FIRMWARE USER'S MANUAL

APPENDIX TO THE ISCSI-1 FIRMWARE USER'S MANUAL

TABLE OF CONTENTS

APPENDIX - A

ISCSI-1 Error Messages Summary

$8001 SCSIbus Parity Error
$8006 Missing or invalid parameter(s)
$8007 Illegal use of an ISCSI-1 Command
$800B Address Error, Invalid Pointer or Address Parameter
$800C An exception was generated during executing a command. The command RAM contains additional informations:

Offset	Size	Meaning
0	W	Error code
2	W	Special format word (68010)
4	L	Program counter (error address)
8	L	Pointer to a system stack copy (This pointer is board relative always!!!)
12	W	Unused/reserved
14	W	reserved for interprocessor communic.

$800E Data compare mismatch
$800F Extended Error Message. The Error Message extension is located in CMDRAM at offset $02.
$8FFF Unknown Command

Extended Error Messages

SCSI ERRORS

$800F0101 ISCSI-1 timeout error
$800F0102 Logical unit is not ready
$800F0103 SCSIbus read error
$800F0104 SCSIbus write error
$800F0105 Specified logical block not found
$800F0106 Target reservation conflict
$800F0107 Format error
$800F0108 Undefined/unknown SCSI command
$800F0109 Target did not respond during selection phase

FLOPPY DISK ERRORS

$800F0302 Floppy disk not ready
$800F0303 FDC read error
$800F0304 FDC write error
$800F0305 Floppy disk record not found
$800F0307 Floppy disk format error

APPENDIX - B

ISCSI-1 Default Setups

Global Definitions:	ISCSI-1 SCSIbus ID	7
	Interfacing Sector Size	256 Bytes
	VMEbus Address Offset	Ø
	Data hashing for write	disabled
SCSIbus Definitions:	Target Mode	disabled
	Special Command Sets	none
	Blocksize for all log. Units	512 Bytes
	Blocking Mode for Write	disabled
	Auto Backup Mode	disabled
	Auto Reserve/Release Unit	disabled
	Hashing for read	enabled
Floppy Disk Drives:	Locked/reserved Units	none
	Sides per Disk	2
	Cylinders per Disk	8Ø
	Sectors per Cylinder	32
	Bytes per Sector	256
	Disk Density	double
	Sector Offset (BIAS)	Ø
	Format Pattern	$E5
	Sector Interleave for Format	1
	Floppy Disk Steprate	3 ms
	Hashing for read	disabled

APPENDIX - C

ISCSI-1 MEMORY LAYOUT FROM LOCAL SIGHT

========== BOARD BASE ==========		
$0000 - $0400	1 KB	VECTOR AREA
$0400 - $2000	3 KB	FIRMWARE WORK AREA
========== DUAL PORTED MEMORY ==========		
$2000	1 W	HOLDS THE BOARD AND THE FIRMAWARE REVISION CODE
$2100 - $210F	16 B	CMDRAM #1
$2110 - $22FF	496 B	CMD CHAIN AREA #1 = 31 CMDRAMS
$2300 - $2AFF	2 KB	I/O BUFFER #1
$2B00 - $32FF	2 KB	I/O BUFFER #2
$3300 - $34FF	512 B	CMD CHAIN AREA #2 = 31 CMDRAMS
$3500 - $350F	16 B	CMDRAM #2
$3510 - $3FFF	3 KB	RESERVED
$4000 - $1FFFF	112 KB	HASHING BUFFERS (16 BUFFERS, 7KB/BUFFER)
========== END OF RAM ==========		

ISCSI-1 MEMORY LAYOUT FROM SYSTEM/VMEBUS SIGHT

```
================== ISCSI-1 BASE ADDRESS ========================

A00000 - A00FFF Reading a WORD from an EVEN address such as
                A00000 will return RESET/SYSFAIL status infor-
                mation in the bits 8,9 and 10.
                Read or write a BYTE at an ODD address will
                access the ISCSI-1 BIM registers as described
                below.

A00001          1 B     BIM control register 0
A00003          1 B     BIM control register 1
A00005          1 B     BIM control register 2
A00007          1 B     BIM control register 3
A00009          1 B     BIM vector register 0
A0000B          1 B     BIM vector register 1
A0000D          1 B     BIM vector register 2
A0000F          1 B     BIM vector register 3
This structure is continued to address A00FFF

A01001-A017FF   1 B     reading  performs a local inter-
     rupt (odd addresses only)

A01801-A01FFF   1 B     reading performs a  local  reset
     (odd addresses only)

========================= BEGIN OF DPR ========================

$A02000                   1 W     HOLDS THE BOARD AND THE FIRMWARE
                                  REVISION CODE

$A02100 - $A0210F        16 B     CMDRAM #1

$A02110 - $A022FF       496 B     CMD CHAIN AREA #1 = 31 CMDRAMS

$A02300 - $A02AFF         2 KB    I/O BUFFER #1

$A02B00 - $A032FF         2 KB    I/O BUFFER #2

$A03300 - $A034FF       512 B     CMD CHAIN AREA #2 = 31 CMDRAMS

$A03500 - $A0350F        16 B     CMDRAM #2

$A03510 - $A03FFF         3 KB    RESERVED

$A04000 - $A01FFFF      112 KB    HASHING BUFFERS
                                  (16 BUFFERS, 7KB/BUFFER)

========================= END OF RAM AREA =====================
```

APPENDIX - D

ISCSI-1 COMMAND SUMMARY

Synonym	Hexcode	Function
SETOFFS	0X03	Set VMEbus address offset
	1X03	Same function but complete with interrupt.
RESET	0X09	Reset the Firmware to power-on state.
SUNPARM	0X0A	Install logical unit (SCSI and local Floppy Disk).
	1X0A	Same function but complete with interrupt.
BLOCKING	0X0B	Enable/disable blocking for write operations.
	1X0B	Same function but complete with interrupt.
WRPARAL	0X0C	Enable/disable runtime backup to parallel logical unit.
	1X0C	Same function but complete with interrupt.
MAININIT	0X0D	Controller main initialization and installation command
	1X0D	Same function but complete with interrupt.
GETCHR	0X10	Read one byte from a logical block.
	1X10	Same function but complete with interrupt.
PUTCHR	0X14	Write one byte to a logical block.
	1X14	Same function but complete with interrupt.
GETBLOC	0X20	Read a fix sized block/sector from a logical unit.
	1X20	Same function but complete with interrupt.
PUTBLOC	0X22	Write a fix sized block/sector to a logical unit.
	1X22	Same function but complete with interrupt.
GETSTR	0X30	Read a delimited string from a logical block.
	1X30	Same function but complete with interrupt.
PUTSTR	0X32	Write a delimited string to a logical block.
	1X32	Same function but complete with interrupt.
GETCNT	0X33	Read a counted string from a logical block
	1X33	Same function but complete with interrupt.

PUTCNT	ØX35	Write a counted string to a logical block.
	1X35	Same function but complete with interrupt.
CTLSTAT	ØX44	Return controller status information.
	1X44	Same function but complete with interrupt.
CHKTARG	ØX45	Check SCSI Targets for existence and device type.
	1X45	Same function but complete with interrupt.
GUNPARM	ØX46	Return logical unit's parameter.
	1X46	Same function but complete with interrupt.
EXPROG	ØX5Ø	Execute User program in local DPR.
	1X5Ø	Same function but complete with interrupt.
BACKUP	ØX55	Backup a logical unit to another.
	1X55	Same function but complete with interrupt.
FLUSH	ØX56	Flush all modified buffers.
	1X56	Same function but complete with interrupt.
FORMAT	ØX57	Format a logical unit.
	1X57	Same function but complete with interrupt.
FTRACK	ØX58	Format a physical track on a local Floppy Disk drive.
	1X58	Same function but complete with interrupt.
COMPARE	ØX59	Compare data between two logical units.
	1X59	Same function but completes with interrupt.
LOCKUN	ØX5A	Lock local unit against access from SCSI-bus initiator.
	1X5A	Same function but complete with interrupt.
FREEUN	ØX5B	Free local unit for SCSIbus access.
	1X5B	Same function but complete with interrupt.
COPY	ØX5C	Copy data from one logical unit to another or between logical blocks on the same logical unit.
CHAIN	ØX5D	Enter command chaining mode.
	1X5D	Same function but complete with interrupt.
TRSPMOD	ØX6Ø	Send a command in transparent mode to SCSIbus.
	1X6Ø	Same function but complete with interrupt.
TARGMOD	ØX61	Enable/disable Target mode.
	1X61	Same function but complete with interrupt.

APPENDIX - E

ISCSI-1 Advanced Programming Examples

All programming examples in this section are written under the assembled with the PDOS macro assembler masm and are executable under PDOS. Appended to each of the programs is an execution example of the program.

The examples are:

- REPORT — This program expects as command line parameter the ID number of an SCSI target and reports the commented device parameters as they are returned by the MODE SENSE command.

- BUS_CHK — This program uses the ISCSI-1 command CHKTARG, check SCSI targets for existence and device type and displays the results as a table.

- IDDUMP — This example is an ISCSI-1 DISK DUMP UTILITY. Look to the listing and the execution example to know how it works.

EXAMPLE 1: REPORT

```
*************************************************************************
***                                                                   ***
***                    ISCSI-1 PROGRAMMING EXAMPLE                    ***
***                                                                   ***
***                 REPORT DISKS CURRENT PARAMETERS                   ***
***                                                                   ***
*************************************************************************
*
        INCLUDE ISCSI:IN
*
        OPT     ALT,NOWARN
*
* FIRST WE GET THE ID FROM THE COMMAND LINE
*
REPORT  XGNP                        ;GET ID FROM COMMAND LINE
        MOVE.L  A1,AØ               ;STORE POINTER
        XCDB                        ;CONVERT TO BINARY
        MOVE.L  D1,D5               ;SAVE IT FOR LATER USE
        XPCL                        ;CR/LF
        XPMC    STRTMSG             ;HEADLINE
        MOVE.L  AØ,A1               ;DISPLAY ID
        XPEL
        XPCL
        BSR     INQU                ;IDENTIFY DRIVE
*
* FIRST TIME MODE SENSE
* IT IS ONLY USED TO GET THE REAL TRANSFER LENGTH
*
        LEA.L   ISCSI+CR_1,AØ       ;CMDRAM
        LEA.L   ISCSI+BUF_1,A1      ;CMD BUFFER
        LEA.L   MODSENS(PC),A2      ;MODE SENSE COMMAND
        MOVE.W  D5,D1               ;TARGET ID
        MOVE.W  #77,D2              ;TRANSFER COUNTER
        BSR     INIT                ;INIT CMDRAM AND COMMAND BUFFER
        BSR     WAIT                ;START AND WAIT UNTIL DONE
*
* LOOK FOR TRANSFER LENGTH AND DO MODE SENSE AGAIN
*
        LEA.L   ISCSI+CR_1,AØ       ;CMDRAM
        LEA.L   ISCSI+BUF_1,A1      ;COMMAND BUFFER
        LEA.L   MODSENS(PC),A2      ;INQUIRY COMMAND
        MOVE.W  D5,D1               ;TARGET ID
        LEA.L   ISCSI+BUF_2,A3      ;DATA BUFFER
        MOVE.B  (A3),D2             ;TRANSFER COUNTER
        MOVE.B  D2,5(A2)            ;SET ALLOCATION LENGTH
        BSR     INIT                ;INIT CMDRAM AND COMMAND BUFFER
        BSR     WAIT                ;START AND WAIT UNTIL DONE
```

EXAMPLE 1 cont'd

```
*
* NOW WE DECODE THE RETURN PARAMETERS
* AND DISPLAY THEM TOGETHER WITH COMMENTS
*
        LEA.L   BUFFER(PC),A1   ;STRING BUFFER
        LEA.L   ISCSI+BUF_2,A2  ;GET DATA BUFFER
        CLR.L   D1
        XPMC    MSG00
        MOVE.B  (A2)+,D1
        MOVE.B  (A2)+,D1
        CMP.B   #0,D1
        BNE.S   @600
        XPMC    DMSG0
        BRA.S   @610
@600    CMP.B   #1,D1
        BNE.S   @601
        XPMC    DMSG1
        BRA.S   @610
@601    XPMC    DMSGU
@610    MOVE.B  (A2)+,D1
        MOVE.B  (A2)+,D1
        TST.B   D1
        BEQ     @100
*
* BLOCK DESCRIPTOR
*
        XPMC    MSG01
        XPMC    MSG02
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        XPMC    MSG03
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        MOVE.B  (A2)+,D1
        CLR.L   D1
        XPMC    MSG04
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        BSR     GETKEY          ;STOP DISPLAY AND WAIT FOR A KEY
```

EXAMPLE 1 cont'd

```
*
* ERROR RECOVERY PARAMETERS = PAGE Ø1
*
@1ØØ    MOVE.B  (A2)+,D1
        ANDI.L  #$1F,D1
        CMP.B   #1,D1
        BNE     @2ØØ
        XPMC    MSGØ5
        XPMC    MSGØ6
        MOVE.B  (A2)+,D1
        MOVE.B  (A2)+,D1
        MOVE.B  #'%',(A1)
        BTST    #7,D1
        BEQ.S   @7ØØ
        MOVE.B  #'1',1(A1)
        BRA.S   @7Ø1
@7ØØ    MOVE.B  #'Ø',1(A1)
@7Ø1    BTST    #6,D1
        BEQ.S   @7Ø2
        MOVE.B  #'1',2(A1)
        BRA.S   @7Ø3
@7Ø2    MOVE.B  #'Ø',2(A1)
@7Ø3    BTST    #5,D1
        BEQ.S   @7Ø4
        MOVE.B  #'1',3(A1)
        BRA.S   @721
@7Ø4    MOVE.B  #'Ø',3(A1)
@721    BTST    #4,D1
        BEQ.S   @7Ø5
        MOVE.B  #'1',4(A1)
        BRA.S   @7Ø6
@7Ø5    MOVE.B  #'Ø',4(A1)
@7Ø6    BTST    #3,D1
        BEQ.S   @7Ø7
        MOVE.B  #'1',5(A1)
        BRA.S   @7Ø8
@7Ø7    MOVE.B  #'Ø',5(A1)
@7Ø8    BTST    #2,D1
        BEQ.S   @7Ø9
        MOVE.B  #'1',6(A1)
        BRA.S   @71Ø
@7Ø9    MOVE.B  #'Ø',6(A1)
@71Ø    BTST    #1,D1
        BEQ.S   @711
        MOVE.B  #'1',7(A1)
        BRA.S   @712
@711    MOVE.B  #'Ø',7(A1)
@712    BTST    #Ø,D1
        BEQ.S   @713
        MOVE.B  #'1',8(A1)
        BRA.S   @714
@713    MOVE.B  #'Ø',8(A1)
```

EXAMPLE 1 cont'

```
@714    MOVE.B  #$Ø,9(A1)
        XPEL
        XPMC    MSGØ7
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        XPMC    MSGØ8
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        XPMC    MSGØ9
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        XPMC    MSG1Ø
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        XPMC    MSG11
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        BSR     GETKEY
*
* DISCONNECT/RECONNECT PARAMETERS = PAGE Ø2
*
@2ØØ    MOVE.B  (A2)+,D1
        ANDI.L  #$1F,D1
        CMP.B   #2,D1
        BNE     @3ØØ
        MOVE.B  (A2)+,D1
        XPMC    MSG12
        XPMC    MSG13
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG14
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG15
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
```

EXAMPLE 1 cont'd

```
        CLR.L   D1
        XPMC    MSG16
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        MOVE.B  (A2)+,D1
        MOVE.B  (A2)+,D1
        BSR     GETKEY
*
* DIRECT ACCESS DEVICE FORMAT PARAMETERS = PAGE 03
*
@300    MOVE.B  (A2)+,D1
        ANDI.L  #$1F,D1
        CMP.B   #3,D1
        BNE     @400
        MOVE.B  (A2)+,D1
        XPMC    MSG17
        XPMC    MSG18
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG19
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG20
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG21
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG22
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
```

EXAMPLE 1 cont'd

```
        CLR.L   D1
        XPMC    MSG23
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG24
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG25
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG26
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG27
        MOVE.B  (A2)+,D1
        MOVE.B  #'%',(A1)
        BTST    #7,D1
        BEQ.S   @800
        MOVE.B  #'1',1(A1)
        BRA.S   @801
@800    MOVE.B  #'0',1(A1)
@801    BTST    #6,D1
        BEQ.S   @802
        MOVE.B  #'1',2(A1)
        BRA.S   @803
@802    MOVE.B  #'0',2(A1)
@803    BTST    #5,D1
        BEQ.S   @804
        MOVE.B  #'1',3(A1)
        BRA.S   @821
@804    MOVE.B  #'0',3(A1)
@821    BTST    #4,D1
        BEQ.S   @805
        MOVE.B  #'1',4(A1)
        BRA.S   @806
@805    MOVE.B  #'0',4(A1)
```

EXAMPLE 1 cont'd

```
@806    BTST    #3,D1
        BEQ.S   @807
        MOVE.B  #'1',5(A1)
        BRA.S   @808
@807    MOVE.B  #'0',5(A1)
@808    BTST    #2,D1
        BEQ.S   @809
        MOVE.B  #'1',6(A1)
        BRA.S   @810
@809    MOVE.B  #'0',6(A1)
@810    BTST    #1,D1
        BEQ.S   @811
        MOVE.B  #'1',7(A1)
        BRA.S   @812
@811    MOVE.B  #'0',7(A1)
@812    BTST    #0,D1
        BEQ.S   @813
        MOVE.B  #'1',8(A1)
        BRA.S   @814
@813    MOVE.B  #'0',8(A1)
@814    MOVE.B  #$0,9(A1)
        XPEL
        CLR.L   D1
        MOVE.B  (A2)+,D1
        MOVE.B  (A2)+,D1
        MOVE.B  (A2)+,D1
        BSR     GETKEY
*
* RIGID DISK DRIVE GEOMETRY PARAMETERS = PAGE 04
*
@400    MOVE.B  (A2)+,D1
        ANDI.L  #$1F,D1
        CMP.B   #4,D1
        BNE     @500
        MOVE.B  (A2)+,D1
        XPMC    MSG28
        XPMC    MSG29
        MOVE.B  (A2)+,D1
        CLR.L   D1
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG30
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
```

EXAMPLE 1 cont'd

```
        XPMC    MSG31
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG32
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG33
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
        XPMC    MSG34
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        LSL.L   #8,D1
        MOVE.B  (A2)+,D1
        XCBX
        XPEL
        CLR.L   D1
*
@500    XEXT
*
************************************
*
* INIT COMMAND BUFFER AND COMMAND RAM
*
INIT    MOVE.L  #7,D0           ;7 BYTES TO COPY
@001    MOVE.B  (A2)+,(A1)+     ;TRANSFER COMMAND
        SUBQ.L  #1,D0
        BNE.S   @001
        MOVE.W  D1,12(A0)       ;SET TARGET ID
        MOVE.W  D2,2(A0)        ;DATA TRANSFER LENGTH
        MOVE.L  #BUF_1,4(A0)    ;SET BUFFER POINTERS
        MOVE.L  #BUF_2,8(A0)
        RTS
```

EXAMPLE 1 cont'd

```
*
************************************
*
* START EXECUTION AND WAIT UNTIL DONE
*
WAIT    MOVE.W  #$60,(A0)       ;TRSPMOD COMMAND
@001    TST.W   (A0)            ;WAIT FOR COMPLETION STATE
        BPL.S   @001
        RTS
*
************************************
*
* WAIT FOR A KEYBOARD INPUT
*
GETKEY  MOVE.L  D0,-(A7)
        XPMC    KEYMSG
        XGCP
        MOVE.L  (A7)+,D0
        RTS
*
************************************
*
* GET INQUIRY MESSAGE FROM TARGET
*
INQU    MOVEM.L D0-A6,-(A7)
        LEA.L   ISCSI+CR_1,A0   ;CMDRAM POINTER
        LEA.L   ISCSI+BUF_1,A1  ;COMMAND BUFFER
        LEA.L   INQUIRY(PC),A2  ;INQUIRY COMMAND
        MOVE.W  D5,D1           ;TARGET ID
        MOVE.W  #50,D2          ;TRANSFER COUNTER
        BSR     INIT            ;INIT CMDRAM AND CMD BUFFER
        BSR     WAIT            ;START AND WAIT UNTIL DONE
        LEA.L   ISCSI+BUF_2,A1  ;GET DATA BUFFER
        ADD.L   #8,A1           ;IGNORE PRECEDING BYTES
        MOVE.B  #0,27(A1)       ;SET 0 DELIMITER
        XPEL                    ;DISPLAY INQUIRY MESSAGE
        MOVEM.L (A7)+,D0-A6
        RTS
*
************************************
*
* DATA AREA
*
* SCSI COMMANDS
*
INQUIRY DC.B    6,$12,0,0,0,50,0        ;INQUIRY COMMAND
MODSENS DC.B    6,$1A,0,$3F,0,77,0      ;MODE SENSE COMMAND
```

EXAMPLE 1 cont'd

```
*
* MESSAGES
*
STRTMSG DC.B    10,13,'DEVICE MODE REPORT, TARGET ID = ',0
MSG00   DC.B    10,13,'DEVICE TYPE = ',0
MSG01   DC.B    10,10,13,'BLOCK DESCRIPTOR',0
MSG02   DC.B    10,13,'  DENSITY CODE.............................',0
MSG03   DC.B    10,13,'  NUMBER OF BLOCKS.........................',0
MSG04   DC.B    10,13,'  BLOCK LENGTH.............................',0
MSG05   DC.B    10,13,'ERROR RECOVERY PARAMETERS',0
MSG06   DC.B    10,13,'  FLAG BYTE...............................',0
MSG07   DC.B    10,13,'  RETRY COUNT.............................',0
MSG08   DC.B    10,13,'  CORRECTION SPAN.........................',0
MSG09   DC.B    10,13,'  HEAD OFFSET COUNT.......................',0
MSG10   DC.B    10,13,'  DATA STROBE OFFSET COUNT................',0
MSG11   DC.B    10,13,'  RECOVERY TIME LIMIT.....................',0
MSG12   DC.B    10,13,'DISCONNECT/RECONNECT PARAMETERS',0
MSG13   DC.B    10,13,'  BUFFER FULL RATIO.......................',0
MSG14   DC.B    10,13,'  BUS INACTIVITY LIMIT....................',0
MSG15   DC.B    10,13,'  DISCONNECT TIME LIMIT...................',0
MSG16   DC.B    10,13,'  CONNECT TIME LIMIT......................',0
MSG17   DC.B    10,13,'DIRECT ACCESS DEVICE PARAMETERS',0
MSG18   DC.B    10,13,'  TRACKS PER ZONE.........................',0
MSG19   DC.B    10,13,'  ALTERNATE SECTORS PER ZONE..............',0
MSG20   DC.B    10,13,'  ALTERNATE TRACKS PER ZONE...............',0
MSG21   DC.B    10,13,'  ALTERNATE TRACKS PER VOLUME.............',0
MSG22   DC.B    10,13,'  SECTORS PER TRACK.......................',0
MSG23   DC.B    10,13,'  DATA BYTES PER SECTOR...................',0
MSG24   DC.B    10,13,'  INTERLEAVE VALUE........................',0
MSG25   DC.B    10,13,'  TRACK SKEW..............................',0
MSG26   DC.B    10,13,'  CYLINDER SKEW...........................',0
MSG27   DC.B    10,13,'  FLAG BYTE...............................',0
MSG28   DC.B    10,13,'RIGID DRIVE GEOMETRY PARAMETERS',0
MSG29   DC.B    10,13,'  MAXIMUM NUMBER OF CYLINDERS.............',0
MSG30   DC.B    10,13,'  MAXIMUM NUMBER OF HEADS.................',0
MSG31   DC.B    10,13,'  STARTING CYLINDER - WRITE PRECOMP.......',0
MSG32   DC.B    10,13,'  STARTING CYLINDER - REDUCED WR CURRENT..',0
MSG33   DC.B    10,13,'  DRIVE STEP RATE.........................',0
MSG34   DC.B    10,13,'  LANDING ZONE CYLINDER...................',0
KEYMSG  DC.B    10,13,'strike any key to continue...',0
DMSG0   DC.B    'MAGNETIC DISK',0
DMSG1   DC.B    'STREAMER TAPE',0
DMSGU   DC.B    'UNKNOWN DEVICE',0
*
        EVEN
*
BUFFER  DS.B    40                      ;STRING BUFFER
*
*
        END
```

EXAMPLE 1 cont'd

PROGRAM RUNNING EXAMPLE:

```
>REPORT 3

DEVICE MODE REPORT, TARGET ID = 3
MICROP  1370            A05
DEVICE TYPE = MAGNETIC DISK
BLOCK DESCRIPTOR
     DENSITY CODE.............................Ø
     NUMBER OF BLOCKS.........................276896
     BLOCK LENGTH.............................512
ERROR RECOVERY PARAMETERS
     FLAG BYTE................................%ØØ1ØØ1ØØ
     RETRY COUNT..............................1Ø
     CORRECTION SPAN..........................11
     HEAD OFFSET COUNT........................Ø
     DATA STROBE OFFSET COUNT.................Ø
     RECOVERY TIME LIMIT......................Ø
DISCONNECT/RECONNECT PARAMETERS
     BUFFER FULL RATIO........................Ø
     BUS INACTIVITY LIMIT.....................5
     DISCONNECT TIME LIMIT....................Ø
     CONNECT TIME LIMIT.......................Ø
DIRECT ACCESS DEVICE PARAMETERS
     TRACKS PER ZONE..........................1
     ALTERNATE SECTORS PER ZONE...............1
     ALTERNATE TRACKS PER ZONE................Ø
     ALTERNATE TRACKS PER VOLUME..............24
     SECTORS PER TRACK........................35
     DATA BYTES PER SECTOR....................512
     INTERLEAVE VALUE.........................Ø
     TRACK SKEW...............................Ø
     CYLINDER SKEW............................Ø
     FLAG BYTE................................%Ø1ØØØØØØ
RIGID DRIVE GEOMETRY PARAMETERS
     MAXIMUM NUMBER OF CYLINDERS..............262152
     MAXIMUM NUMBER OF HEADS..................Ø
     STARTING CYLINDER - WRITE PRECOMP........Ø
     STARTING CYLINDER - REDUCED WR CURRENT...Ø
     DRIVE STEP RATE..........................Ø
     LANDING ZONE CYLINDER....................Ø
```

EXAMPLE 2: BUS CHK

```
*****************************************************************
***                                                           ***
***                  ISCSI-1 PROGRAMMING EXAMPLE              ***
***                                                           ***
***         CHECK TARGETS FOR EXISTENCE AND DEVICE TYPE       ***
***                                                           ***
*****************************************************************
*
        INCLUDE ISCSI:IN
*
        OPT     ALT,NOWARN
*
BUS_CHK XPCL                        ;CR/LF
        XPCL                        ;CR/LF
        XPMC    MSTART              ;START MESSAGE
        MOVEA.L #ISCSI+CR_1,A0      ;CMDRAM
@001    TST.W   (A0)                ;BUSY?
        BPL.S   @001                ;Y, WAIT
        MOVE.W  #$45,(A0)           ;COMMAND
@002    TST.W   (A0)                ;BUSY?
        BPL.S   @002                ;Y, WAIT
*
        MOVEA.L 4(A0),A3            ;GET POINTER
        ADDA.L  #ISCSI,A3           ;ADD BOARD BASE
        MOVE.L  #0,D1               ;COUNTER
@100    MOVE.B  (A3)+,D2            ;GET DATA
        XPCL                        ;CR/LF
        XCBD                        ;BIN TO DEC_ASCII
        XPEL                        ;WRITE ID
        CMP.B   #$FF,D2             ;OWN?
        BNE.S   @101                ;N
        XPMC    MOWNID              ;Y
        BRA.S   @300
@101    CMP.B   #$80,D2             ;UNKNOWN DEVICE?
        BNE.S   @102                ;N
        XPMC    MUNKN               ;Y
        BRA.S   @300
@102    CMP.B   #$7F,D2             ;NON EXISTING?
        BNE.S   @103                ;N
        XPMC    MNEXIS              ;Y
        BRA.S   @300
@103    CMP.B   #$03,D2             ;PROCESSOR DEVICE?
        BNE.S   @105                ;N
        XPMC    MPROC               ;Y
        MOVE.L  D1,D5
        BSR     INQU
        BRA.S   @300
@105    CMP.B   #0,D2               ;MAGNETIC DISK?
        BNE.S   @104                ;N
        XPMC    MMAGN               ;Y
        MOVE.L  D1,D5
```

EXAMPLE 2 cont'd

```
        BSR     INQU
        BSR     CAPACIT
        BRA.S   @300
@104    XPMC    MTAPE           ;SHOULD BE STREAMER TAPE
@300    ADDQ.L  #1,D1           ;NEXT ID
        CMP.B   #8,D1           ;DONE?
        BLT.S   @100            ;N, CONTINUE
*
        XPCL                    ;CR/LF
        XEXT                    ;EXIT TO PDOS
*
INQU    MOVEM.L D0-A6,-(A7)
        LEA.L   ISCSI+CR_1,A0           ;CMDRAM
        LEA.L   ISCSI+BUF_1,A1
        LEA.L   INQUIRY(PC),A2          ;INQUIRY COMMAND
        MOVE.W  D5,D1
        MOVE.W  #50,D2                  ;TRANSFER COUNTER
        BSR     INIT
        BSR     WAIT
*
        LEA.L   ISCSI+BUF_2,A1
        MOVE.B  #0,(A1,D2.W)
        ADD.L   #8,A1
        MOVE.B  #0,27(A1)
        XPMC    BLANK
        XPEL
        BSR     CLEAR
*
        MOVEM.L (A7)+,D0-A6
        RTS
*
*
CAPACIT MOVEM.L D0-A6,-(A7)
        XPMC    MSG03
        LEA.L   ISCSI+CR_1,A0           ;CMDRAM
        LEA.L   ISCSI+BUF_1,A1
        LEA.L   CAPAC(PC),A2            ;CAPACITY COMMAND
        MOVE.W  D5,D1                   ;TARGET ID
        MOVE.W  #8,D2                   ;TRANSFER COUNTER
        BSR     INIT1
        BSR     WAIT1
*
        LEA.L   ISCSI+BUF_2,A0
        LEA.L   BUFFER(PC),A1
        MOVE.L  (A0)+,D3
        MOVE.L  (A0),D4
        MOVE.L  D4,D5
        DIVU    #1000,D3
        ANDI.L  #$0FFFF,D3
        MULU    D4,D3
        DIVU    #1000,D3
        ANDI.L  #$0FFFF,D3
        MOVE.L  D3,D1
```

EXAMPLE 2 cont'd

```
        XCBX
        XPEL
        XPMC    MSGØ2
        MOVE.L  D5,D1
        XPMC    MSGØ1
        LEA.L   BUFF_2(PC),A1
        XCBX
        XPEL
*
        MOVEM.L (A7)+,DØ-A6
        RTS
*
*
INIT1   MOVE.L  #7,DØ
@ØØ1    MOVE.B  (A2)+,(A1)+
        SUBQ.L  #1,DØ
        BNE.S   @ØØ1
        MOVE.W  D1,12(AØ)
        MOVE.W  D2,2(AØ)
        MOVE.L  #BUF_1,4(AØ)
        MOVE.L  #BUF_2,8(AØ)
        RTS
*
*
WAIT1   MOVE.W  #$6Ø,(AØ)
@ØØ1    TST.W   (AØ)
        BPL.S   @ØØ1
        RTS
*
*
CAPAC   DC.B    1Ø,$25,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
*
        EVEN
*
BUFFER  DS.B    4Ø
BUFF_2  DS.B    4Ø
*
MSGØ1   DC.B    1Ø,13,'          BLOCKSIZE = ',Ø
MSGØ4   DC.B    ' BYTES',Ø
MSGØ2   DC.B    ' MBYTE',Ø
MSGØ3   DC.B    1Ø,13,'          CAPACITY = ',Ø
*
        EVEN
*
CLEAR   LEA.L   ISCSI+BUF_2,A4
        MOVE.L  #1Ø,D7
@ØØ1    CLR.L   (A4)+
        SUBQ.L  #1,D7
        BNE.S   @ØØ1
        RTS
```

EXAMPLE 2 cont'd

```
INIT    MOVE.L  #7,D0
@001    MOVE.B  (A2)+,(A1)+
        SUBQ.L  #1,D0
        BNE.S   @001
        MOVE.W  D1,12(A0)
        MOVE.W  D2,2(A0)
        MOVE.L  #BUF_1,4(A0)
        MOVE.L  #BUF_2,8(A0)
        RTS
*
WAIT    MOVE.W  #$60,(A0)
@001    TST.W   (A0)
        BPL.S   @001
        RTS
*
TUNRDY  DC.B    6,0,0,0,0,0,0
INQUIRY DC.B    6,$12,0,0,0,50,0
BLANK   DC.B    10,13,'           ',0
*
*
MOWNID  DC.B    '     ISCSI-1',0
MUNKN   DC.B    '     UNKNOWN DEVICE TYPE',0
MMAGN   DC.B    '     MAGNETIC DISK',0
MNEXIS  DC.B    '     -----',0
MTAPE   DC.B    '     STREAMER TAPE',0
MPROC   DC.B    '     PROCESSOR DEVICE',0
MSTART  DC.B    'ID    DEVICE',10,13
        DC.B    '---------------------------',0
*
        EVEN
        END
```

PROGRAM RUNNING EXAMPLE:

```
>BUS_CHK

ID    DEVICE
---------------------------
0     -----
1     -----
2     MAGNETIC DISK
          QUANTUM Q2802022003-REV00
          CAPACITY = 79 MBYTE
          BLOCKSIZE = 512
3     MAGNETIC DISK
          MICROP  1370            A05
          CAPACITY = 141 MBYTE
          BLOCKSIZE = 512
4     -----
5     -----
6     -----
7     ISCSI-1
```

EXAMPLE 3: IDDUMP

```
*****************************************************************
***                                                           ***
***                  ISCSI-1 PROGRAMMING EXAMPLE              ***
***                                                           ***
***                       DISK DUMP UTILITY                   ***
***                                                           ***
*****************************************************************
*
        INCLUDE ISCSI:IN
*
        OPT     ALT,NOWARN
*
IDDUMP  XCLS                        ;CLEAR SCREEN
        XPMC    SMSG                ;DISPLAY MESSAGE
        XPMC    IMSG
        LEA.L   PBUFF(PC),A1
        XGLB
        BEQ     EXIT
        XCDB
        BLT     EXIT
        MOVE.L  D1,D7
        LSL.W   #1,D7
        LSL.W   #8,D7
        OR.W    #$20,D7
        XPCL
        XPXPMC  LMSG
        LEA.L   PBUFF(PC),A1
        XGLB
        BEQ     @800
        XCDB
        BLT     EXIT
        MOVE.L  D1,D6
        BRA.S   @801
@800    MOVE.L  #0,D6
        XPMC    NULL
@801    XPCL
        MOVE.L  #-1,D5
START   XPMC    PMSG
        LEA.L   PBUFF(PC),A1            ;GET SECTOR #
        XGLB
        BEQ     @000
        XCDB                            ;CONVERT TO BINARY
        BLT     L002
        MOVE.L  D1,D5                   ;SAVE IT FOR LATER USE
        XPCL
        BRA.S   @700
@000    ADD.L   #1,D5
        MOVE.L  D5,D1
        LEA.L   PBUFF(PC),A1
        XCBX
        XPEL
        XPCL
```

EXAMPLE 3 cont'd

```
@700    CMP.W    #$0E20,D7
        BNE.S    @300
        MOVEA.L  #ISCSI+CR_1,A0
@010    MOVE.W   #1,2(A0)
        MOVE.L   D5,4(A0)
        MOVE.L   #$4000,8(A0)
        MOVE.W   D6,12(A0)
        MOVE.W   D7,(A0)
@020    TST.W    (A0)
        BPL.S    @020
*
        TST.B    1(A0)
        BNE      ERROR
*
        MOVEA.L  #ISCSI+CR_1,A0
        MOVEA.L  4(A0),A0
        ADDA.L   #ISCSI,A0
        MOVE.L   #$10,D4
        BSR.L    MDUMP
        BRA.L    START
*
@300    MOVEM.L  D5/D7,-(A7)
        LEA.L    $A02301,A3
        MOVE.B   #$06,(A3)+
        OR.L     #$08000000,D5
        MOVE.L   D5,(A3)+
        MOVE.W   #$0100,(A3)+
        MOVEA.L  #ISCSI+CR_1,A0
        MOVE.L   #$2301,4(A0)
        MOVE.L   #$2400,8(A0)
        MOVE.W   #$200,2(A0)
        LSR.W    #1,D7
        LSR.W    #8,D7
        MOVE.W   D7,12(A0)
        MOVEM.L  (A7)+,D5/D7
        MOVE.W   #$0060,(A0)
*
@301    TST.W    (A0)
        BPL.S    @301
*
        TST.B    1(A0)
        BNE      ERROR
        MOVEA.L  #ISCSI+CR_1,A0
        MOVEA.L  8(A0),A0
        ADDA.L   #ISCSI,A0
        MOVE.L   #$10,D4
        BSR.S    MDUMP
        XPCL
        XPMC     KMSG
        XGCP
        XPCL
        MOVE.L   #$10,D4
        BSR.S    MDUMP
        BRA.L    START
```

EXAMPLE 3 cont'd

```
*
EXIT    XEXT
*
L002    LEA.L   PBUFF(PC),A1
        CMP.B   #'?',(A1)
        BEQ.S   @001
        CMP.B   #'.',(A1)
        BEQ     IDDUMP
        BRA     START
@001    XPCL
        MOVEM.L D1/D7/A1,-(A7)
        LEA.L   PBUFF(PC),A1
        XPMC    IMSG
        LSR.W   #1,D7
        LSR.W   #8,D7
        MOVE.L  D7,D1
        XCBX
        XPEL
        XPMC    LMSG
        MOVE.L  D6,D1
        XCBX
        XPEL
        XPMC    PMSG
        MOVE.L  D5,D1
        XCBX
        XPEL
        XPCL
        MOVEM.L (A7)+,D1/D7/A1
        BRA     START
*
ERROR   XPMC    EMSG
        BRA.S   EXIT
*
**********************************
*
* MEMORY DUMP
*
*       IN : A0.L = MEMORY ADDRESS
*       OUT: 256 BYTES ARE DIPLAYED
*            A0 IS UPDATED
*
MDUMP   MOVEM.L D0-D3/A1,-(A7)
        LEA.L   WBUFF(PC),A1
        MOVE.L  D4,D1
@001    MOVE.W  #$0A0D,(A1)+
        LEA.L   ABUFF(PC),A2
        MOVE.L  #4,D2
        MOVE.L  A0,-(A7)
        BSR.L   HEXAS
        CLR.L   (A7)+
        MOVE.W  #': ',(A1)+
```

EXAMPLE 3 cont'd

```
@002    MOVE.L   (A0),-(A7)
        MOVE.L   (A0),(A2)+
        BSR.L    HEXAS
        CLR.L    (A7)+
        MOVE.W   #'  ',(A1)+
        ADDA.L   #4,A0
        SUBQ.L   #1,D2
        BNE.S    @002
*
        MOVE.L   #16,D2
        LEA.L    ABUFF(PC),A2
@003    MOVE.B   (A2)+,D3
        CMP.B    #$20,D3
        BLT.S    @004
        CMP.B    #$7F,D3
        BGT.S    @004
        MOVE.B   D3,(A1)+
        BRA.S    @005
@004    MOVE.B   #'.',(A1)+
@005    SUBQ.L   #1,D2
        BNE.S    @003
*
        SUBQ.L   #1,D1
        BNE.S    @001
*
        MOVE.L   #$0A0D0000,(A1)+
        LEA.L    WBUFF(PC),A1
        XPEL
        MOVEM.L  (A7)+,D0-D3/A1
        RTS
*
**********************************
*
* TRANSFORM HEX TO ASCII
*
*       IN : 4(A7) = HEX NUMBER
*            A1.L = BUFFER
*
HEXAS   MOVEM.L  D0-D2,-(A7)
        MOVE.L   16(A7),D0                ;GET HEX NUMBER
        MOVE.L   #8,D2
@001    ROL.L    #4,D0
        MOVE.B   D0,D1
        ANDI.L   #$0F,D1
        CMP.B    #$0A,D1
        BLT.S    @002
        ADD.B    #$37,D1
        BRA.S    @003
@002    ADD.B    #$30,D1
@003    MOVE.B   D1,(A1)+
        SUBQ.L   #1,D2
        BNE.S    @001
        MOVEM.L  (A7)+,D0-D2
        RTS
```

EXAMPLE 3 cont'd

```
*
MSG01   DC.B    10,13,'SECTOR DUMP...',10,13,0
PMSG    DC.B    10,13,'BLOCK NUMBER : ',0
SMSG    DC.B    10,13,'ISCSI-1 DISK DUMP UTILITY!'
        DC.B    10,13,'==========================',10,13,0
IMSG    DC.B    10,13,10,'TARGET ID    : ',0
LMSG    DC.B    10,13,'UNIT NUMBER  : ',0
EMSG    DC.B    10,13,10,'***** READ ERROR *****',10,13,0
KMSG    DC.B    'strike any key to continue...',0
NULL    DC.B    '0',0
*
        EVEN
*
WBUFF   DS.B    4000
ABUFF   DS.B    400
PBUFF   DS.B    20
*
        END
```

PROGRAM RUNNING EXAMPLE:

```
>IDDUMP

ISCSI-1 DISK DUMP UTILITY!
==========================

TARGET ID    : 3

UNIT NUMBER  : 0

BLOCK NUMBER : 9

00A02400: 41534D00  00000000  00000000  8004024C  ASM.............L
00A02410: 00000000  00000016  1034AB06  1034AB06  .........4...4..
00A02420: 53554E50  41524D00  53520001  028001DD  SUNPARM.SR......
00A02430: 00000016  00160083  0A16AB57  0F16AB57  ...........W...W
00A02440: 424C4F43  4B494E47  4F420065  2000039B  BLOCKINGOB.e ...
00A02450: 00000001  0001004F  101FAB21  130DAD49  .......O...!...I
00A02460: 424C4F43  4B494E47  53520001  02000125  BLOCKINGSR.....%
00A02470: 0000000B  000B0086  0E07AB1D  1238AD49  ............8.I
00A02480: 42554646  57520000  53520002  02040486  BUFFWR..SR......
00A02490: 00000009  000900C9  0B17AB16  0E26AD47  ............&.G
00A024A0: 43484149  4E000000  4F420065  2004039D  CHAIN...OB.e ...
00A024B0: 00000000  00000066  1020AB21  1020AB21  .......f. .!. .!
00A024C0: 43484149  4E000000  53520001  020402CA  CHAIN...SR......
00A024D0: 00000006  000400A4  1217AB07  0B20AB15  ............. ..
00A024E0: 43484149  4E4D0000  4F420065  2000039E  CHAINM..OB.e ...
00A024F0: 0000000D  000D0007  1020AB21  0E1BAD47  ......... .!...G

strike any key to continue...
```

EXAMPLE 3 cont'd

```
00A02500: 43484149  4E4D0000  53520001  0200029E  CHAINM..SR......
00A02510: 00000013  001200F5  1210AB07  0F30AB23  .............0.#
00A02520: 434D4452  414D0000  53520032  0204006D  CMDRAM..SR.2...m
00A02530: 00000006  00050018  100EAAEF  0A13AB0B  ................
00A02540: 434D4453  45540000  53520032  020002E1  CMDSET..SR.2....
00A02550: 00000013  00130006  0C02AD08  0B20AD46  ............. .F
00A02560: 434D4454  424C0000  494E0003  028400BD  CMDTBL..IN......
00A02570: 00000026  001B00BB  0B11AADA  1008AB57  ...&...........W
00A02580: 434F4D50  4C455400  53520002  02000074  COMPLET.SR.....t
00A02590: 00000009  000900C9  1106AAFF  1108AD54  ...............T
00A025A0: 43544C53  54415400  53520001  020001BF  CTLSTAT.SR......
00A025B0: 00000011  001100F0  0E33AB28  0F13AB28  .........3.(...(
00A025C0: 44454255  47000000  4F420065  200404C3  DEBUG...OB.e ...
00A025D0: 0000001C  001C0081  1101AB21  130BAD49  ...........!...I
00A025E0: 44454654  41420000  53520032  028002B0  DEFTAB..SR.2....
00A025F0: 00000016  001500FB  111FAB1C  1012AB57  ...............W

BLOCK NUMBER : ?

TARGET ID    : 3
UNIT NUMBER  : 0
BLOCK NUMBER : 9

BLOCK NUMBER : .


ISCSI-1 DISK DUMP UTILITY!
==========================

TARGET ID    : 2

UNIT NUMBER  : 0

BLOCK NUMBER : 0

00A02400: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02410: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02420: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02430: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02440: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02450: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02460: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02470: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02480: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02490: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A024A0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A024B0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A024C0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A024D0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A024E0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A024F0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................

strike any key to continue...
```

EXAMPLE 3 cont'd

```
00A02500: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02510: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02520: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02530: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02540: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02550: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02560: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02570: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02580: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A02590: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A025A0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A025B0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A025C0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A025D0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A025E0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................
00A025F0: E5E5E5E5  E5E5E5E5  E5E5E5E5  E5E5E5E5  ................

BLOCK NUMBER : <ESC> (QUITS TO SYSTEM)
```

EXAMPLE 4: CONNECT

```
*****************************************************************
***                                                           ***
***                ISCSI-1 PROGRAMMING EXAMPLE                ***
***                                                           ***
*****************************************************************
***                                                           ***
***     CONNECT TWO ISCSI-1 BOARDS TOGETHER. ONE AS AN        ***
***     INITIATOR, THE OTHER AS A TARGET.                     ***
***                                                           ***
***     WE ASSUME, THE FIRST ISCSI-1 SHOULD BE THE INITIATOR  ***
***     AND HAS THE BASE ADDRESS $A00000.                     ***
***     THE SECOND ISCSI-1 WHICH HAS TO BE THE TARGET HAS THE ***
***     BASE ADDRESS $A80000.                                 ***
***                                                           ***
***     WHEN INITIALIZATION IS DONE WE CAN USE THE OTHER      ***
***     UTILITIES TO ACCESS THE TARGET BOARD.                 ***
***                                                           ***
*****************************************************************
*
        OPT     ALT
*
* FIRST WE INITIALIZE THE INITIATOR BOARD
*
INIT    LEA.L   $A00000,A0              ;GET BASE ADDRESS
*
* SET ADDRESS OFFSET
*
        MOVE.W  #$0003,D3               ;SETOFFS COMMAND
        SWAP    D3                      ;COMMAND IN HIGH WORD
        MOVE.W  #0,D3                   ;FIRST PARAMETER -
        MOVE.L  A0,D2                   ;BASE ADDRESS OFFSET
        MOVEQ.L #0,D1                   ;THIRD PARAMETER
        MOVEQ.L #0,D0                   ;FOURTH PARAMETER
        BSR.L   COMMAND                 ;ISSUE ISCSI-1 COMMAND
        SWAP    D3                      ;GET ERROR CODE
        TST.B   D3                      ;ERROR?
        BNE.L   ERROR                   ;YES
*
* GET ISCSI-1 STATUS
*
        MOVE.W  #$0044,D3               ;CTLSTAT COMMAND
        SWAP    D3                      ;COMMAND IN HIGH WORD
        MOVE.W  #$0,D3                  ;FIRST PARAMETER
        MOVEQ.L #0,D2                   ;SECOND PARAMETER
        MOVEQ.L #0,D1                   ;THIRD PARAMETER
        MOVEQ.L #0,D0                   ;FOURTH PARAMETER
        BSR.L   COMMAND                 ;ISSUE ISCSI-1 COMMAND
        SWAP    D3                      ;GET ERROR CODE
        TST.B   D3                      ;ERROR?
        BNE.L   ERROR                   ;YES
        MOVEA.L D2,A1                   ;GET PARAMETER POINTER
        MOVE.B  (A1),D4                 ;CONTROLLER ID
        TST.B   7(A1)                   ;TEST TARGET MODE
        BEQ.S   @001                    ;DISABLED, CONTINUE
```

```
*
* DISABLE TARGET MODE
*
        MOVE.L  #$00620000,D3           ;TARGMOD COMMAND
        MOVEQ.L #0,D2                   ;SECOND PARAMETER
        MOVEQ.L #0,D1                   ;THIRD PARAMETER
        MOVEQ.L #0,D0                   ;FOURTH PARAMETER
        BSR.L   COMMAND                 ;ISSUE ISCSI-1 COMMAND
        SWAP    D3                      ;GET ERROR CODE
        TST.B   D3                      ;ERROR?
        BNE.L   ERROR                   ;YES
*
* CHECK AND SET ID IF NECESSARY
*
@001    CMP.B   #7,D4                   ;ID = 7?
        BEQ.S   @002                    ;YES, CONTINUE
        MOVE.L  #$07000000,$2300(A0)    ;SET PARAMETERS FOR MAININIT
        MOVE.L  #$000D0000,D3           ;MAININIT COMMAND
        MOVE.L  A0,D2                   ;BASE ADDRESS
        ADD.L   #$2300,D3               ;PARAMETER POINTER
        MOVEQ.L #0,D1                   ;THIRD PARAMETER
        MOVEQ.L #0,D0                   ;FOURTH PARAMETER
        BSR.L   COMMAND                 ;ISSUE ISCSI-1 COMMAND
        SWAP    D3                      ;GET ERROR CODE
        TST.B   D3                      ;ERROR?
        BNE.L   ERROR                   ;YES
*
* NOW WE INITIALZE THE SECOND CONTROLLER WHICH IS THE TARGET
*
@002    LEA.L   $A80000,A0              ;GET BASE ADDRESS
*
* SET ADDRESS OFFSET
*
        MOVE.W  #$0003,D3               ;SETOFFS COMMAND
        SWAP    D3                      ;COMMAND IN HIGH WORD
        MOVE.W  #0,D3                   ;FIRST PARAMETER
        MOVE.L  A0,D2                   ;BASE ADDRESS OFFSET
        MOVEQ.L #0,D1                   ;THIRD PARAMETER
        MOVEQ.L #0,D0                   ;FOURTH PARAMETER
        BSR.L   COMMAND                 ;ISSUE ISCSI-1 COMMAND
        SWAP    D3                      ;GET ERROR CODE
        TST.B   D3                      ;ERROR?
        BNE.L   ERROR                   ;YES
*
* GET ISCSI-1 STATUS
*
        MOVE.W  #$0044,D3               ;CTLSTAT COMMAND
        SWAP    D3                      ;COMMAND IN HIGH WORD
        MOVE.W  #$0,D3                  ;FIRST PARAMETER
        MOVEQ.L #0,D2                   ;SECOND PARAMETER
        MOVEQ.L #0,D1                   ;THIRD PARAMETER
        MOVEQ.L #0,D0                   ;FOURTH PARAMETER
        BSR.L   COMMAND                 ;ISSUE ISCSI-1 COMMAND
        SWAP    D3                      ;GET ERROR CODE
        TST.B   D3                      ;ERROR?
```

```
        BNE.L   ERROR                   ;YES
        MOVEA.L D2,A1                   ;GET PARAMETER POINTER
        MOVE.B  (A1),D4                 ;CONTROLLER ID
        TST.B   7(A1)                   ;TEST TARGET MODE
        BNE.S   @003                    ;ENABLED, CONTINUE
*
* ENABLE TARGET MODE
*
        MOVE.L  #$00620000,D3           ;TARGMOD COMMAND
        MOVEQ.L #0,D2                   ;SECOND PARAMETER
        MOVEQ.L #0,D1                   ;THIRD PARAMETER
        MOVEQ.L #0,D0                   ;FOURTH PARAMETER
        BSR.L   COMMAND                 ;ISSUE ISCSI-1 COMMAND
        SWAP    D3                      ;GET ERROR CODE
        TST.B   D3                      ;ERROR?
        BNE.S   ERROR                   ;YES
*
* CHECK AND SET ID IF NECESSARY
*
@003    CMP.B   #6,D4                   ;ID = 6?
        BEQ.S   @002                    ;YES, CONTINUE
        MOVE.L  #$06000000,$2300(A0)    ;SET PARAMETERS FOR MAININIT
        MOVE.L  #$000D0000,D3           ;MAININIT COMMAND
        MOVE.L  A0,D2                   ;BASE ADDRESS
        ADD.L   #$2300,D3               ;PARAMETER POINTER
        MOVEQ.L #0,D1                   ;THIRD PARAMETER
        MOVEQ.L #0,D0                   ;FOURTH PARAMETER
        BSR.L   COMMAND                 ;ISSUE ISCSI-1 COMMAND
        SWAP    D3                      ;GET ERROR CODE
        TST.B   D3                      ;ERROR?
        BNE.S   ERROR                   ;YES
        XEXT                            ;NO, EXIT TO PDOS
*
* ERROR HANDLING
*
ERROR   XPMC    EMSG                    ;PRINT MESSAGE
        XEXT                            ;EXIT TO PDOS
*
EMSG    DC.B    10,13,'ISCSI-1 ERROR',10,13,0
*
        EVEN
*
* SUBROUTINE
*       ISSUE ISCSI-1 COMMAND
*
COMMAND MOVE.L  A0,-(A7)                ;SAVE BASE ADDRESS
        ADDA.L  #$2110,A0               ;GET COMMAND RAM TOP
        MOVEM.L D0-D3,-(A0)             ;SETUP CMDRAM
@001    TST.W   (A0)                    ;WAIT UNTIL READY
        BPL.S   @001
        MOVEM.L (A0)+,D0-D3             ;GET RETURN VALUES
        MOVE.L  (A7)+,A0                ;RESTORE BASE ADDRESS
        RTS                             ;RETURN TO CALLER
*
*
        END
```

USER NOTES 1

USER NOTES 2

APPLICATIONS

MODIFICATIONS