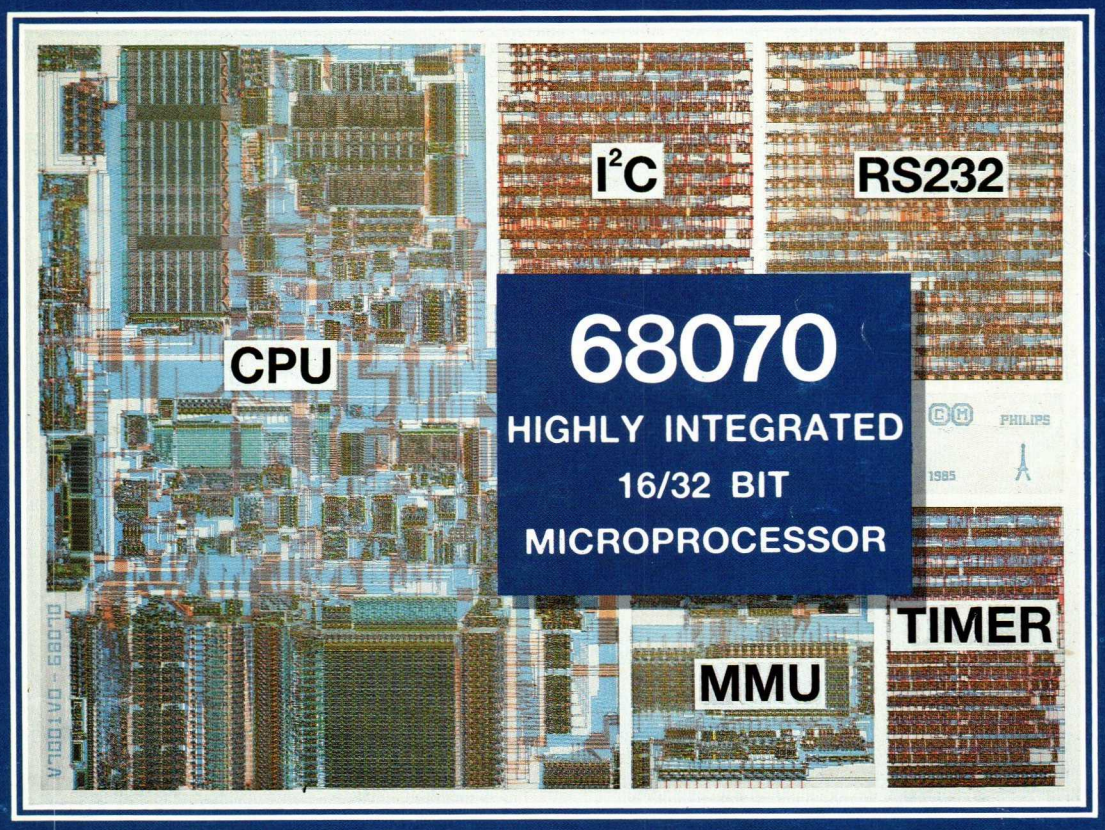
Introducing the 68070



16/32-bit microprocessor

PHILIPS

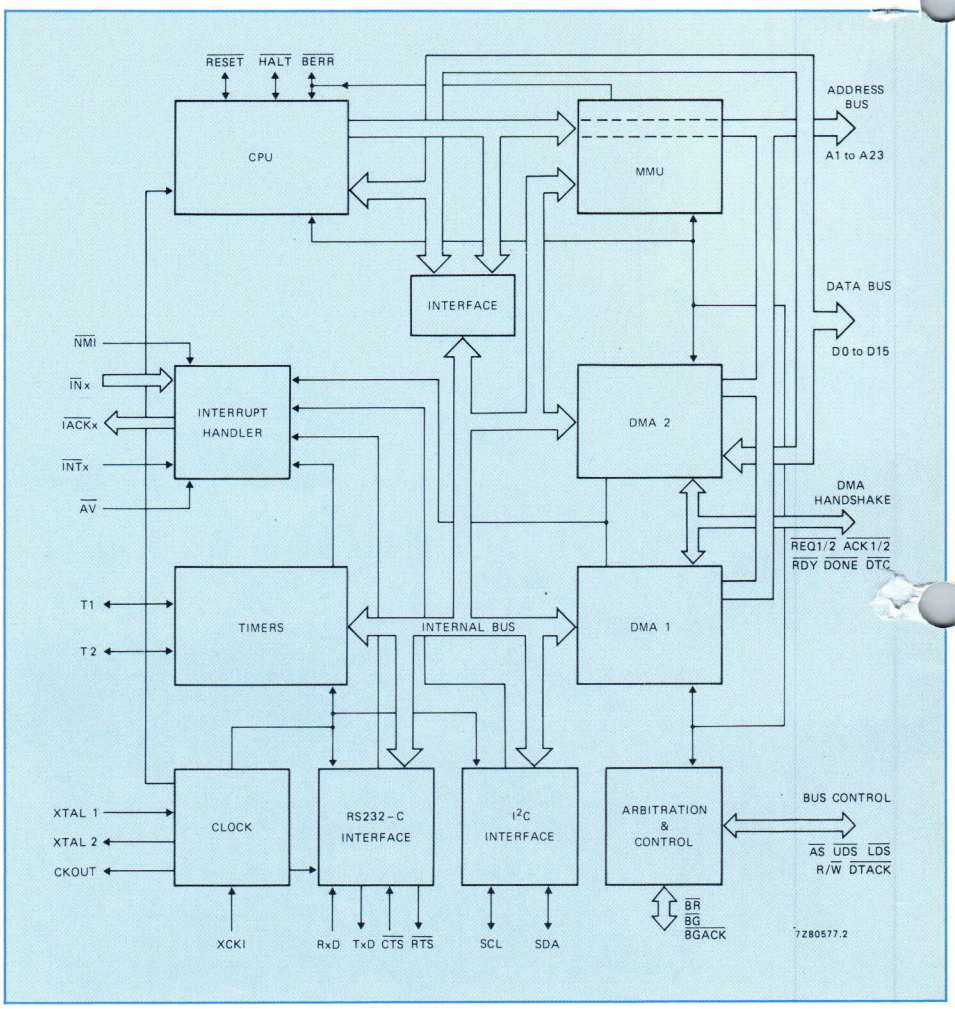
Electronic components and materials

FUNCTIONAL DESCR

Developed from the 68000, the generally accepted standard for mid-range work-stations, the SCC68070 is a highly-integrated microprocessor designed for the more cost-conscious market of home and personal computers, games, intelligent terminals, data communications and high-performance control.

Main Features

- ☐ CHMOS technology
- ☐ Full 68000 software compatibility
- ☐ Enhanced bus error handling
- ☐ 68000-compatible bus interface
- ☐ Same bus timing as 10 MHz 68000
- ☐ 84-pin leaded chip carrier
- ☐ 16 Mbyte addressing range
- ☐ Vectored and auto-vectored interrupts
- ☐ 4 Decoded interrupt levels
- ☐ Decoded interrupt acknowledge
- ☐ 2 programmable latched interrupts
- ☐ Built-in clock generator: 20 MHz (max) crystal
- ☐ Maximum internal clock frequency: 10 MHz
- ☐ On-chip MMU
- ☐ 2-channel DMA controller
- ☐ I²C serial bus interface
- ☐ RS232-C serial interface
- ☐ 16-bit timer
- ☐ Two 16-bit match/count/capture registers



Block diagram of the 68070

Pin Description

Power supply	V_{DD} (×2)	Supply voltage (single 5 V)		BERR (I/O)	Bus error
				HALT (I/O)	Halt
Ground	V_{SS} (×2)			RESET (I/O)	Reset
Clock	XTAL 1, 2 (I)	Crystal input		A1...A23 (O)	Address bus
	CKOUT (O)	Clock out		DO...D15 (I/O)	Data bus
CPU int.cntr.	INT1, INT2 (I)	Edge-triggered interrupts	DMA control	REQ1, REQ2 (I)	DMA request
				ACK1, ACK2 (O)	DMA request acknowledge
	IN2, IN4, IN5 (I)	Decoded interrupt			
	NMI (I)	Non-maskable interrupt		RDY (I)	Device ready
	IACK2...7 (O)	Interrupt acknowledge		DTC (O)	Device transfer complete
	AV (I)	Auto vectored interrupts			
CPU, control	AS (O)	Address strobe		DONE (I/O)	Done
	LDS (O)	Lower data strobe	I²C	SCL (I/O)	Serial clock
	UDS (O)	Upper data strobe		SDA (I/O)	Serial data
	R/W (O)	Read/write	RS232-C	RxD (I)	Receive data
	DTACK (I)	Data transfer acknowledge		TxD (O)	Transmit data
				RTS (O)	Request to send
	BR (I)	Bus request		CTS (I)	Clear to send
	BG (O)	Bus grant		XCKI (I)	External clock input
	BGACK (I/O)	Bus grant acknowledge	Capture timer	T1, T2 (I/O)	Timer output or event input

PTION OF THE 68070

The 68070 is divided into several functional units:

a) central processing unit (CPU)
b) memory management unit (MMU)
c) DMA channels
d) serial bus (I²C)
e) serial interface (RS232-C)
f) capture timer

a) Central processing unit (CPU)

The 68070 CPU has a full 32-bit architecture and with the same instruction set, programming model and internal resources, the 68070 maintains full software compatibility with the 68000.

Four of its 7 interrupt levels (2, 4, 5 and 7) are provided as decoded external interrupts with separate acknowledge outputs. A further 2 latched interrupts can be programmed to a desired priority. The on-chip priority-programmable interrupts are accessed using a separate vector table.

CPU recovery from bus errors is similar to that of the 68010. When a bus errors occurs, the CPU will push its internal content onto a stack prior to executing the exception routine.

b) Memory management unit (MMU)

The MMU is used to translate logical (virtual) to physical addresses and to provide segment protection against illegal access. The MMU has 8 descriptors in hardware for segment base, segment length and attributes. The MMU can handle 8 segments of up to 2 Mbytes or 128 segments of up to 128 Kbytes. Segment protection is ensured by checking the supervisor, read, write or execute permission. A stack attribute supports stack handling. When illegal access to a protected area occurs, the MMU initiates a bus error exception routine on the CPU. The MMU can be disabled by the CPU via a control register.

c) DMA channels

The 68070 has two independent DMA channels with fixed priority. The functional signals are compatible with the 68430, 68440 and 68450. The transfer (format: byte or word) is from memory to memory (channel 2) or memory to I/O-device and vice versa (channel 1 and 2). Maximum transfer rate is 1,6 million per second in either single-cycle (cycle stealing) or burst mode, with typical latency times of less than 2,5 μs.

d) Serial bus (I²C)

The I²C (inter-IC) bus consists of two serial lines: one carrying the clock signal and the other, the serial data. More than one device in a system may request to be bus master and after arbitration, one of the devices is recognized as such. The 68070 can operate as a receiver or transmitter in either master or slave mode. Data is transferred in bytes (excluding start and stop bits) at rates up to 100 kbits/s. The I²C interfaces directly with other I²C devices (84XX family, 84CXX family and I²C-bus IC examples on back cover).

e) Serial interface (RS232-C)

The RS232-C serial interface is a universal asynchronous data communication controller to connect to a printer, terminal or tape unit, for example. It features full duplex as well as auto echo-mode. Two baud rate generators are programmable to one of eight transmit or receive rates (75, 150, 300, 1200, 2400, 4800, 9600 and 19600 baud), or to divide an external clock for data communication.

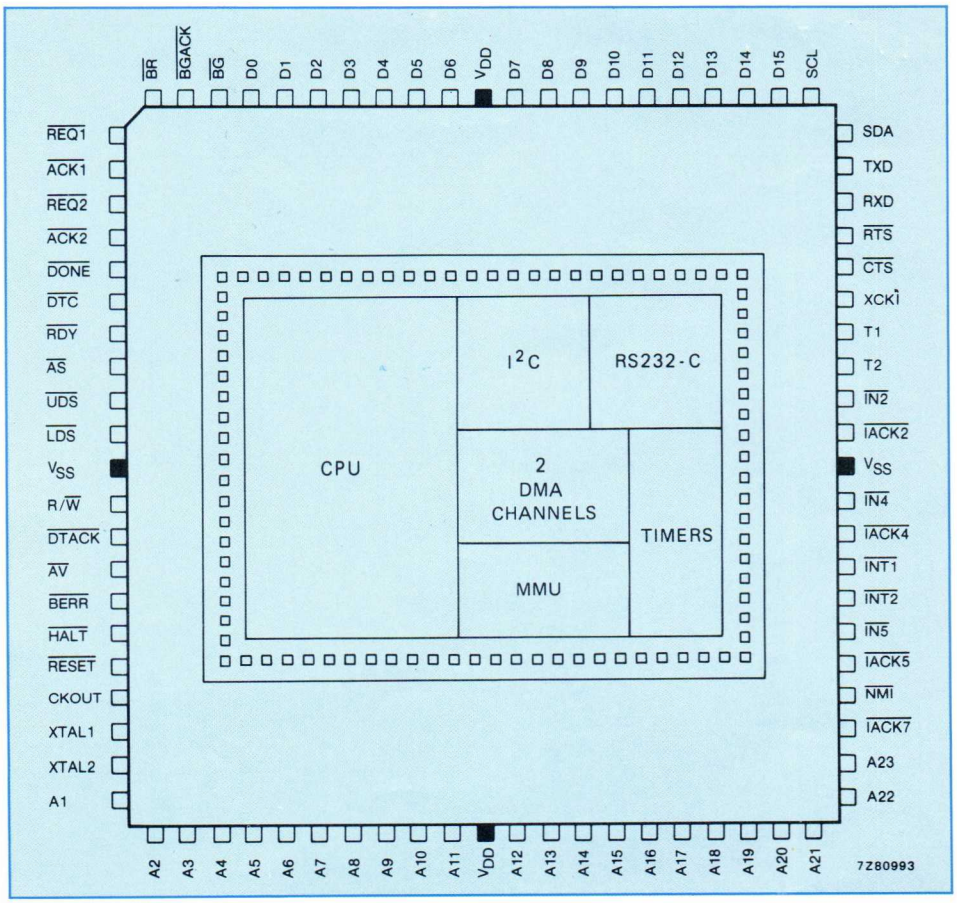
f) Capture timer

The 68070 has 3 independent 16-bit timers:

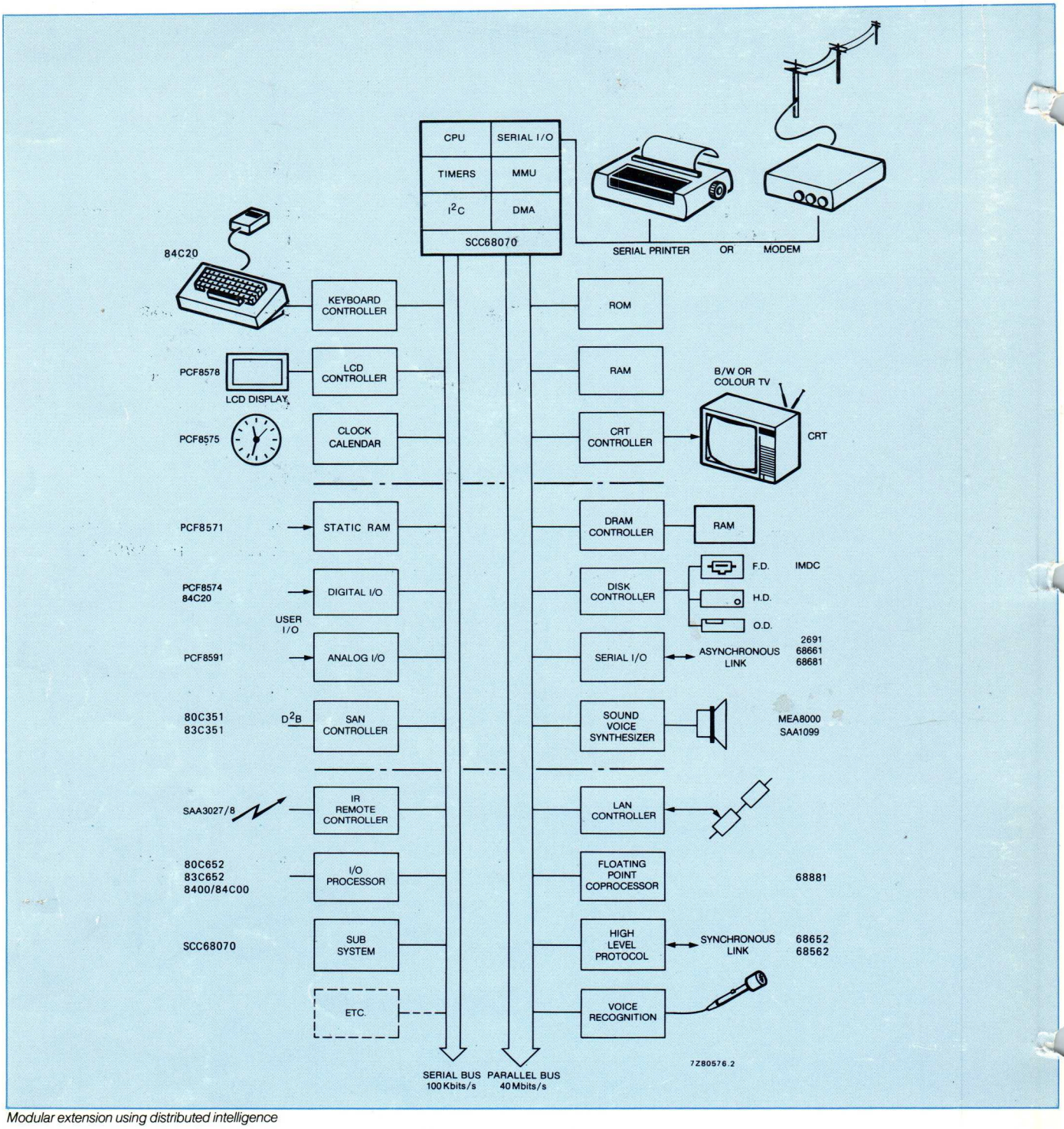
- one continuous 16-bit timer as a reference clock, 10 μs resolution, with automatic reload on overflow
- two programmable capture timers. Each capture timer features a 16-bit register/counter that can be used in the following modes:

  1) **Pulse generator,** to change output state when a match occurs between the reference counter and the match register.
  2) **Event counter,** to count external events into a count register.
  3) **Capture timer,** to store the reference timer value in the capture register when an external event occurs.

  An external event can be a HIGH/LOW, LOW/HIGH or both.



Pinning diagram of the 68070



Modular extension using distributed intelligence

MORE INFORMATION FROM:

Austria: ÖSTERREICHISCHE PHILIPS BAUELEMENTE INDUSTRIE G.m.b.H., WIEN, Tel. 62 91 11.
Belgium: N.V. PHILIPS & MBLE ASSOCIATED, 9 rue du Pavillon, B-1030 BRUXELLES, Tel. (02) 242 74 00.
Denmark: MINIWATT A/S, COPENHAGEN S, Tel. (01) 54 11 33.
Finland: OY PHILIPS AB, Elcoma Division, HELSINKI, Tel. 1 72 71.
France: R.T.C. LA RADIOTECHNIQUE-COMPELEC, PARIS, Tel. 338 80-00.
Germany (Fed. Republic): VALVO, UB Bauelemente der Philips G.m.b.H., HAMBURG, Tel. (040) 3296-0.
Greece: PHILIPS S.A. HELLENIQUE, Elcoma Division, 52, Av. Syngrou, ATHENS, Tel. 9215111.
Ireland: PHILIPS ELECTRICAL (IRELAND) LTD., DUBLIN, Tel. 69 33 55.
Italy: PHILIPS S.p.A., Sezione Elcoma, MILANO, Tel. 2-6752.1.
Netherlands: PHILIPS NEDERLAND, Marktgroep Elonco, Postbus 90050, 5600 PB EINDHOVEN, Tel. (040) 79 33 33.
Norway: NORSK A/S PHILIPS, Electronica Dept., OSLO, Tel. 68 02 00.
Portugal: PHILIPS PORTUGUESA S.A.R.L., LISBOA Codex, Tel. 68 31 21.
Spain: MINIWATT S.A., BARCELONA, Tel. 301 63 12.
Sweden: PHILIPS KOMPONENTER A.B., STOCKHOLM, Tel. 08/7821000.
Switzerland: PHILIPS A.G., Elcoma Dept., ZÜRICH, Tel. 01-442 2 11.
Turkey: TÜRK PHILIPS TICARET A.S., Elcoma Dept., P.K.504, 80074 ISTANBUL, Tel. 435 9 10.
United Kingdom: MULLARD LTD., LONDON, Tel. 01-580 6633.

Printed in The Netherlands

Date of release: 1 - 86

9398 331 30011

16-BIT MICROPROCESSOR

GENERAL DESCRIPTION

The 68070 is a highly integrated 16/32-bit central processing unit for use in a large variety of applications and is fully software compatible with the 68000. Integrating standard as well as advanced peripheral functions on the 68070 (housed in an 84-pin package), dramatically reduces system cost.

This document gives an overview of the basic functions, internal structure, and d.c. and a.c. characteristics. For further detail on the features and operation of the 68070, refer to the "User manual PCB68070".

FEATURES

- **CHMOS technology**
- **32-bit internal structure**
- **Enhanced bus error handling**
- **84-pin package**
- **4 decoded interrupt inputs**
- **2 programmable interrupt inputs**
- **Decoded interrupt acknowledge**
- **Built-in clock generator: max. 20 MHz crystal**
- **On-chip MMU, supporting virtual memory**
- **2-channel DMA controller**
- **I^2C serial bus interface**
- **RS232-C serial interface**
- **16-bit timer/counter**
- **Two 16-bit match/count/capture registers**

- **Full 68000 software compatibility**
- **68000-compatible bus interface (10 MHz)**
- **56 powerful instruction types**
- **5 basic data types**
- **16 Mbyte addressing range**
- **14 addressing modes**
- **Memory mapped I/O**
- **Vectored and auto-vectored interrupts**
- **7 interrupt levels**
- **Maximum internal clock frequency: 10 MHz**

The internal architecture of the 68070 is built around a bus interconnecting the CPU and the various on-chip peripheral functions. Each function has several dedicated connections to the external circuitry. The 68070 includes powerful programmable interrupt processing circuitry for interrupts generated by internal and external sources. An on-chip clock generator provides a 10 MHz clock signal for CPU and peripheral interfaces.

If enabled, the on-chip MMU takes care of address translation and memory protection. Two DMA channels increase data throughput and I^2C-bus interface allows easy and low-cost addition of peripherals (master and slave devices). The 68070 also includes an RS232-C interface. A built-in Timer/counter with two independently programmable MATCH/COUNT/CAPTURE registers means that the 68070 can be programmed with two of the following options simultaneously:

- pulse generator;
- external event counter;
- reference timer.

ORDERING INFORMATION

type number	temperature range	clock frequency	package
PCB68070 WP	0 to 70 °C	10 MHz	84-pin PLCC

9 091 10142

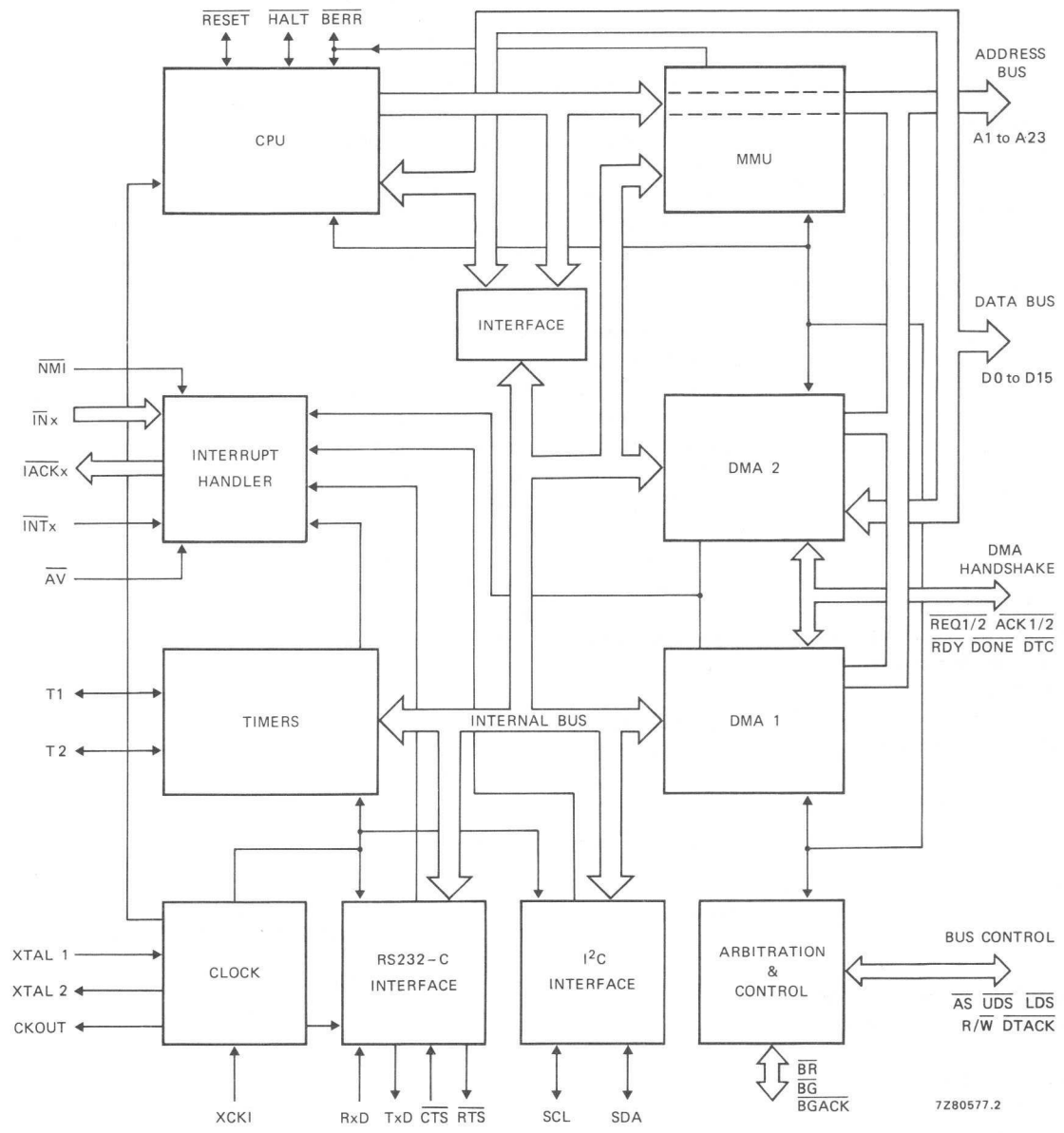
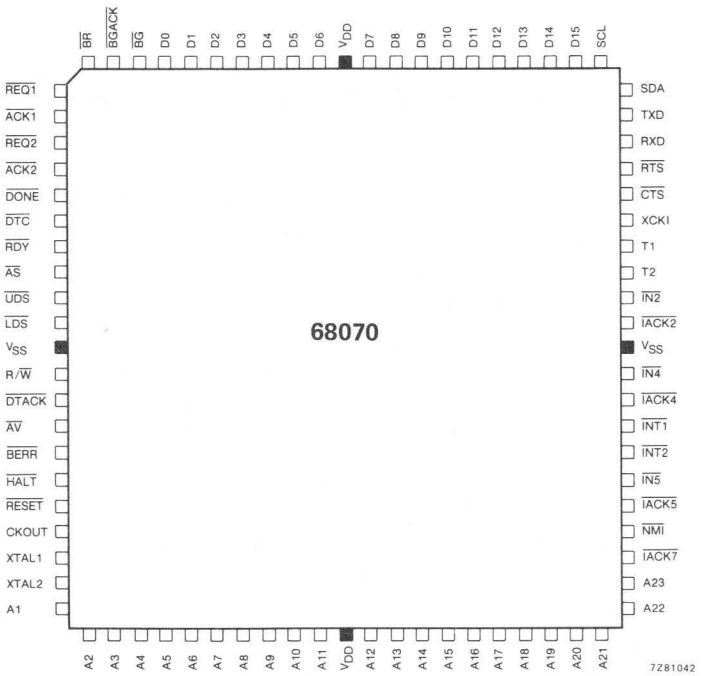


Fig. 1 Block diagram of the 68070.

DEVELOPMENT DATA

SIGNAL DESCRIPTION



NOTATION:
O – Output
I – Input
I/O – Bidirectional
TS – 3-state
OD – Open-Drain
AH – Active-HIGH
AL – Active-LOW

Fig. 2 Pinning.

MNEMONIC	FUNCTION
A1 to A23 O, TS, AH	Address bus for direct addressing 16 Mbytes of memory.
D0 to D15 I/O, TS, AH	16-bit wide bidirectional data bus.
$\overline{AS}$ O, TS, AL	Address Strobe; indicates a valid address on the bus.
$\overline{LDS}$ O, TS, AL	Lower Data Strobe; indicates that: – For a WRITE cycle, the data is valid on the lower half of the data bus (D0 to D7). – For a READ cycle, the data is to be placed on the lower half of the bus (D0 to D7).
$\overline{UDS}$ O, TS, AL	Upper Data Strobe; indicates that: – For a WRITE cycle, the data is valid on the upper half of the data bus (D8 to D15). – For a READ cycle, the data is to be placed on the upper half of the bus (D8 to D15).
R/$\overline{W}$ O, TS	Read (active-HIGH)/Write (active-LOW); controls the direction of data flow.
$\overline{DTACK}$ I, AL	Data Transfer Acknowledge – Asserted by the peripheral during CPU or DMA bus cycles when data is either received from or placed on the bus. If not asserted puntually, it causes the CPU or DMA controller to insert wait states.

MNEMONIC	FUNCTION
$\overline{BR}$ I, AL	Bus Request – Asserted by wire-ORed external DMA devices that request bus ownership.
$\overline{BG}$ O, AL	Bus Grant Output – A daisy chain output that is asserted by the 68070 when the bus is granted by the CPU and the DMA does not have a bus request pending.
$\overline{BGACK}$ I/O, OD, AL	Bus Grant Acknowledge – Asserted by any DMA device (internal or external) that has control of the bus. As long as this line is held LOW externally, the 68070 will hold the bus signals in the high impedance state. When $\overline{BGACK}$ is released, the 68070 will have access to the bus. Note that interrupts cannot be serviced while $\overline{BGACK}$ is held LOW.
$\overline{RESET}$ I/O, OD, AL	Bidirectional Reset – If asserted externally together with the $\overline{HALT}$ line, it will cause the processor to enter the Reset state. It is driven LOW by the processor when the $\overline{RESET}$ instruction resets external hardware.
$\overline{HALT}$ I/O, OD, AL	Bidirectional Halt – If asserted externally together with $\overline{RESET}$, it causes the 68070 to enter the Reset state. If asserted alone, it will cause the CPU or DMA controller to stop after completion of the current bus cycle. If $\overline{HALT}$ and $\overline{BERR}$ are asserted together, the 68070 will complete the current bus cycle, stop operation, and place all 3-state lines in the high-impedance state until $\overline{HALT}$ and $\overline{BERR}$ have been released, and then it will rerun the same bus cycle. $\overline{BERR}$ should be released before $\overline{HALT}$. As long as $\overline{HALT}$ is held LOW, all control signals are inactive and all 3-state lines are placed in the high-impedance state. When the processor has stopped executing instructions (e.g. after a double bus fault), the processor drives this line LOW.
$\overline{BERR}$ I/O, OD, AL	Bus Error – If this line is asserted during a bus cycle, it indicates that there was a fault in the bus access. If asserted together with $\overline{HALT}$, the same bus cycle will be rerun after both $\overline{HALT}$ and $\overline{BERR}$ have been released. If $\overline{BERR}$ is asserted alone, the 68070 will start bus-error exception processing. $\overline{BERR}$ is driven LOW by the 68070 when the on-chip MMU indicates a bus error.
$\overline{INT1}$, $\overline{INT2}$ I, AL	Latched interrupt inputs – A LOW level of $\geqslant$ 1 clock pulse will be stored as a pending interrupt request. Priority levels can be programmed on-chip.
$\overline{IN2}$, $\overline{IN4}$, $\overline{IN5}$ I, AL	Decoded interrupt priority inputs – IN2 has the lowest and IN5 has the highest priority except for $\overline{NMI}$.
$\overline{NMI}$ I, AL	Non-maskable interrupt (level 7) – While the other interrupts may be masked (disabled), this interrupt is always enabled.
$\overline{IACK2}$, $\overline{IACK4}$ $\overline{IACK5}$, $\overline{IACK7}$ O, AL	Decoded interrupt acknowledge – Asserted during an interrupt acknowledge sequence to indicate to a peripheral that its interrupt request is being serviced.
$\overline{AV}$ I, AL	Auto vectored interrupts – If held LOW during the interrupt acknowledge sequence, the processor calculates the appropriate vector from a fixed vector table. If kept HIGH, the peripheral must provide an 8-bit offset for vector acquisition.
V_{DD}	Supply voltage +4,5 V to +5,5 V
V_{SS}	Ground
XTAL1, XTAL2 I	External crystal inputs – XTAL1 can be used as a clock-input if an external clock generator is used (V_{IH} = 3,5 V). The crystal or external clock frequency is divided by 2 to obtain the internal clock and CKOUT signals.
CKOUT O	Clock out – This is the reference from the internal clock system: maximum frequency 10 MHz.
$\overline{REQ1}$, $\overline{REQ2}$ I, AL	DMA Request (active-LOW) – These are inputs from I/O devices requesting service from the DMA controller and causes it to request control of the bus. In burst mode, the inputs are level sensitive and the DMA controller releases the bus after $\overline{REQx}$ is negated and the current DMA cycle is completed. In cycle-stealing mode, $\overline{REQx}$ inputs are triggered by a negative pulse. This pulse must occur at least one clock cycle before $\overline{DTC}$ is asserted to ensure continuous transfer.
$\overline{ACK1}$, $\overline{ACK2}$ O, AL	DMA Request Acknowledge (active-LOW) – $\overline{ACKx}$ is asserted by the DMA controller to indicate that it has acquired the bus and the requested device bus cycle is now beginning. It is asserted at the beginning of every device bus cycle together with $\overline{AS}$, and is negated at the end of every device bus cycle.

MNEMONIC	FUNCTION
$\overline{RDY}$ I, AL	Device Ready (active-LOW) – The requesting device asserts $\overline{RDY}$ to indicate to the DMA controller that valid data has either been stored or put on the bus. If $\overline{RDY}$ is negated, it indicates that the data has neither been stored nor put on the bus, causing the DMA controller to insert wait states. $\overline{RDY}$ can be held LOW permanently if the device is fast enough, indicating that the device is always ready and so no wait states are required. $\overline{RDY}$ is not monitored by channel 2 in the two address modes.
$\overline{DTC}$ O, OD, AL	Device Transfer Complete (active-LOW) – In DMA mode, the $\overline{DTC}$ is asserted by the DMA controller to indicate to the device that the requested data transfer is complete. On a write-to-memory operation, it indicates that the data provided by the device has been stored successfully. On a read-from-memory operation, it indicates to the device that the data from memory is present on the data bus and should be latched.
$\overline{DONE}$ I/O, OD, AL	Done (active-LOW, open drain) – With $\overline{DONE}$ as an output, the DMA controller asserts it simultaneously with the $\overline{ACK}$x output to indicate to the device that the transfer count is zero and therefore, the DMA controller's operation is complete. If, as an input, $\overline{DONE}$ is asserted by the device before the transfer count reaches zero, it causes the DMA controller to abort operation and generate an interrupt request (if the interrupts are enabled).
SCL I/O, OD	Serial clock – SCL is the clock signal pin for I^2C-bus operation. It is either driven by the 68070 when the I^2C interface is in the master mode, or is the clock input if the I^2C interface is in the slave mode.
SDA I/O, OD	Serial data – SDA is the data I/O line for I^2C-bus operation.
T1, T2 I/O, TS	Timers 1 and 2 – These are I/O signals for the capture timers of channels 1 and 2 respectively. They can be programmed as either outputs for pulses or inputs for count cycles and events.
RXD I	Receive Data (TTL-level) – RXD is data input for the RS232-C interface.
TXD O	Transmit Data (TTL-level) – TXD is data output for the RS232-C interface.
$\overline{RTS}$ O, AL	Request To Send (TTL-level) – This output of the RS232-C interface indicates that the transmitter is about to transmit data on the TXD line.
$\overline{CTS}$ I, AL	Clear To Send (TTL-level) – This input to the RS232-C interface indicates that the receiving device is ready. $\overline{RTS}$ and $\overline{CTS}$ can be connected together if no control lines are needed.
XCKI I	External clock (TTL-level) – XCKI is the clock input for the RS232-C interface. The signal is used to generate special baud rates. When a clock frequency other than the 20 MHz is used by the 68070, an external clock can be connected to this input to generate the baud rates and transfer clocks for the RS232-C and I^2C interfaces.
XCKI I	External clock (TTL-level) – XCKI is the clock input for the RS232-C interface. The signal is used to generate special baud rates. When a clock frequency other than the 20 MHz is used by the 68070, an external clock can be connected to this input to generate the baud rates and transfer clocks for the RS232-C and I^2C interfaces.

DEVELOPMENT DATA

ON-CHIP ADDRESSES

All memory locations on the peripheral side of the on-chip interface can only be accessed in SUPERVISOR mode; i.e. the S-bit in the Processor Status Word (PSW) is set to 1. All registers on the peripheral side of the interface are memory mapped separately from the 68070's 16 Mbyte memory map of the outside world. The on-chip address space which is decoded by the two MSBs (A31 and A30) of the 32-bit internal address and the S-bit of the processor status word are explained in Table 1.

Table 1 A31, A30 and S-bit decoding

S	A31	A30	
X	0	0	external
X	0	1	external
1	1	0	internal only
0	1	0	external
X	1	1	external

The address map of all on-chip peripherals is given in Table 2.

Table 2 On-Chip Addresses

0	0	0	0	0	0	0	0	to	7	F	F	F	F	F	F	F	off-chip
8	0	0	0	0	0	0	0	to	8	0	0	0	0	F	F	F	on-chip, reserved
8	0	0	0	1	0	0	0										$\overline{\text{INT}}$x priority
8	0	0	0	1	0	0	1	to	8	0	0	0	2	0	0	0	on-chip, reserved
8	0	0	0	2	0	0	1	to	8	0	0	0	2	0	0	9	I^2C interface
8	0	0	0	2	0	0	A	to	8	0	0	0	2	0	1	0	on-chip, reserved
8	0	0	0	2	0	1	1	to	8	0	0	0	2	0	1	F	RS232-C interface
8	0	0	0	2	0	2	0	to	8	0	0	0	2	0	2	9	Timer
8	0	0	0	2	0	2	A	to	8	0	0	0	2	0	4	4	on-chip, reserved
8	0	0	0	2	0	4	5										PICR1
8	0	0	0	2	0	4	6										on-chip, reserved
8	0	0	0	2	0	4	7										PICR2
8	0	0	0	2	0	4	8	to	8	0	0	0	3	F	F	F	on-chip, reserved
8	0	0	0	4	0	0	0	to	8	0	0	0	4	0	6	D	DMA Controller
8	0	0	0	4	0	6	E	to	8	0	0	0	7	F	F	F	on-chip, reserved
8	0	0	0	8	0	0	0	to	8	0	0	0	8	0	7	F	MMU
8	0	0	0	8	0	8	0	to	B	F	F	F	F	F	F	F	on-chip, reserved
C	0	0	0	0	0	0	0	to	F	F	F	F	F	F	F	F	off-chip

BUS TIMING

Bus cycles in the 68070 are similar to those of a 68000 running on a 10 MHz clock frequency. However, if the $\overline{\text{DTACK}}$ signal is not asserted by the time the 68070 is ready to transmit or receive data, it will insert wait cycles. Upper and lower data strobes ($\overline{\text{UDS}}$ and $\overline{\text{LDS}}$) are asserted independently with respect to the type of transfer (low byte – $\overline{\text{LDS}}$ asserted, high byte – $\overline{\text{UDS}}$ asserted, and word – both strobes asserted).

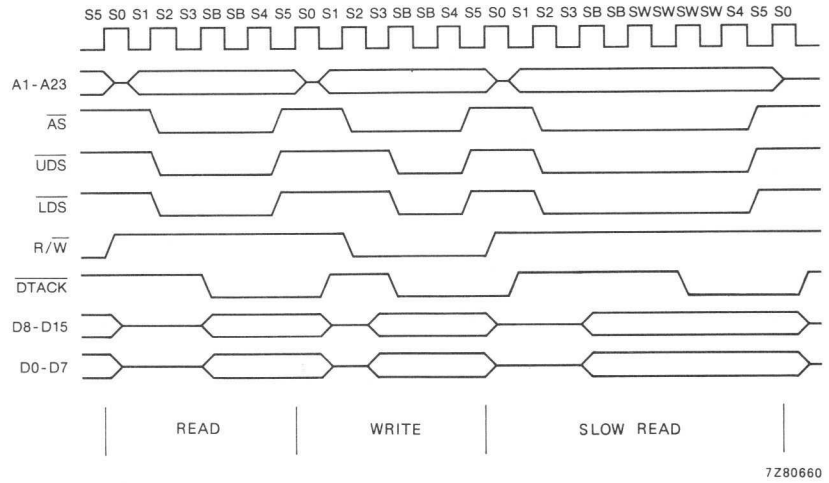


Fig. 3 Read, write and slow read cycle timing.

CPU FUNCTIONAL DESCRIPTION

GENERAL

The CPU of the 68070 is software compatible with the 68000, so all programs written for the 68000 will run on the 68070 unchanged. However, for certain applications the following differences between the processors should be noted:

- Differences exist in the bus-error exception processing since the 68070 can provide full bus-error recovery.
- The timing is different because of the 68070's new architecture and technology. Although the bus timing is similar to a 10 MHz 68000, instruction execution timing is completely different (For execution timing, see the end of this section, Tables 7 to 19).

PROGRAMMING MODEL AND DATA ORGANIZATION

The programming model is identical to that of the 68000 as shown in Fig. 4 with seventeen 32-bit registers, a 32-bit Program Counter and a 16-bit Status Register. The first eight registers (D0 - D7) are used as data registers for byte, word and long-word operations. The second group of registers (A0 - A6) and the system stack pointer (A7) can be used as software stack pointers and base address registers. In addition, these registers can be used for word and long-word address operations.

All seventeen registers can be used as index registers.

Figure 4 shows the programming model and Figure 5 illustrates the status register.

The 68070 supports bit, 8, 16 and 32-bit integers, and BCD data, together with 32-bit addresses. Each data type is arranged in the memory as shown in Figure 6.

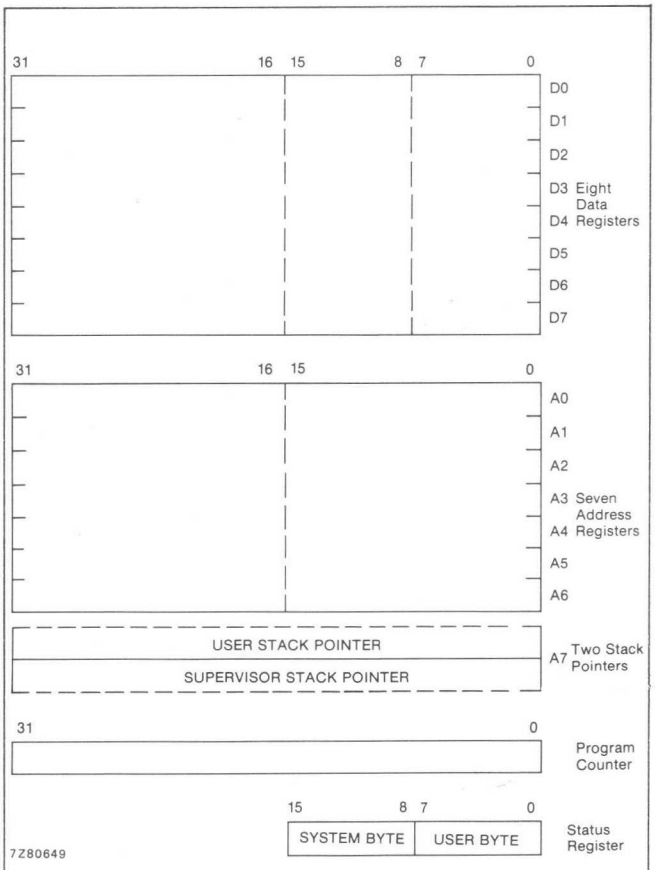


Fig. 4 Programming model.

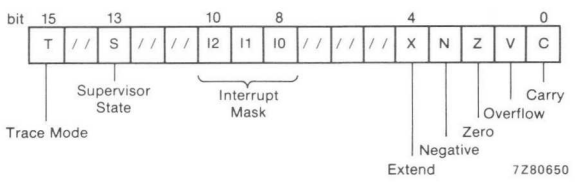


Fig. 5 Status Register.

PHILIPS

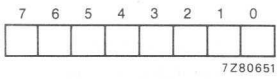


Fig. 6(a) Bit data (1 byte = 8 bits).

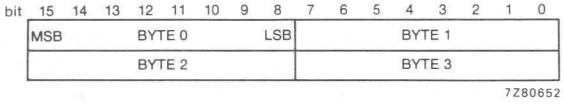


Fig. 6(b) Integer data (1 byte = 8 bits).

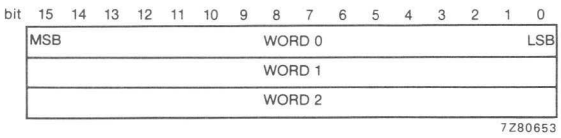


Fig. 6(c) Word data (16 bits).

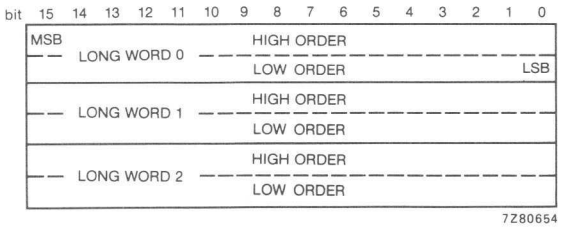


Fig. 6(d) Long-word data (32 bits).

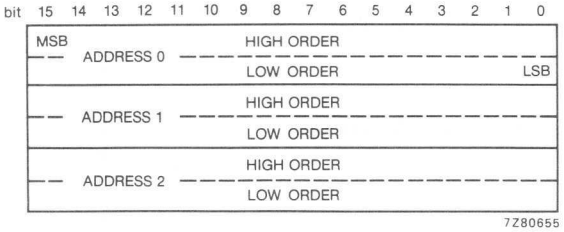


Fig. 6(e) Addresses (1 address = 32 bits).

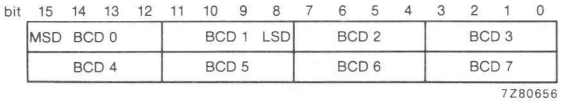


Fig. 6(f) BCD data (2 BCD digits = 1 byte).

INTERNAL AND EXTERNAL OPERATION

The 68070 operates with a maximum internal clock frequency of 10 MHz and a minimum of 4 MHz. Each clock cycle is divided into 2 states. A non-access machine cycle has 3 clock cycles or 6 states (S0 to S5) and a bus cycle (SB) normally consists of 3 clock cycles plus 1 bus cycle (8 states). When $\overline{\text{DTACK}}$ is not asserted, indicating that data has not been received or not put on the bus, wait states (SW) are inserted in multiples of 2.

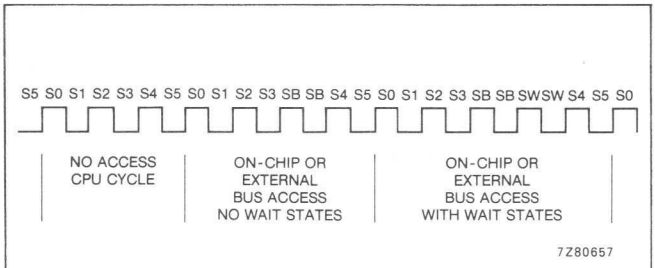


Fig. 7 Machine cycles.

BUS ARBITRATION

Because a DMA controller is integrated on the 68070 as a possible bus master, the bus arbitration needs a priority protocol. This is done by a daisy-chain using the $\overline{\text{BG}}$ of the CPU such that channel 1 of the DMA controller has highest priority, followed by channel 2 and then the external devices. The CPU grants bus acquisition and therefore has the lowest priority. Once the DMA controller has submitted the internal bus grant to an outside master, it will not interrupt the line until BUS GRANT ACKNOWLEDGE ($\overline{\text{BGACK}}$) has been negated. If the DMA controller has a DMA request pending, it will acquire the bus as soon as the external device has negated $\overline{\text{BGACK}}$. In this case, it will not submit $\overline{\text{BG}}$ to an outside master even if the prospective master's $\overline{\text{BR}}$ signal had been asserted before the DMA controller's pending request.

The 68070 bus arbitration also works like the 68008's two-signal mode, using $\overline{\text{BG}}$ and $\overline{\text{BR}}$ only. $\overline{\text{BG}}$ will be asserted in response to $\overline{\text{BR}}$ after either the processor has finished the bus cycle or the DMA controller has completed its task. $\overline{\text{BG}}$ will stay asserted until either $\overline{\text{BGACK}}$ is asserted or $\overline{\text{BR}}$ is negated.

PROCESSING STATES AND EXCEPTION PROCESSING

The CPU is always in one of three processing states: normal, exception, or halted. The normal processing state is that associated with instruction execution; the memory references are to fetch instructions and operands, and to store results. A special case of the normal state is the stoppped state which the processor enters when a STOP instruction is executed. In this state, no further memory references are made by the CPU.

The exception processing state is associated with interrupts, trap instructions, tracing and other exceptional conditions. The exception may be generated internally by an instruction or by an unusual condition arising during the execution of an instruction. Externally, exception processing can be forced by an interrupt, by bus error, or by a reset.

The processor can work in the 'user' or 'supervisor' state determined by the state of the S-bit in the status register. Accesses to the on-chip peripherals must be in the supervisor state.

All exception processing is performed in the supervisor state once the current content of the Status Register has been copied. The interrupt vector number is then determined, and the PC value and the copy of the Status Register are saved on the supervisor stack using the SSP. Finally, the interrupt vector content is fetched and loaded into the PC.

Exception vectors

Exception vectors are memory locations from which the CPU fetches the address of a routine that will handle that exception. All exception vectors are 2 words in length (see Fig. 8) except for the reset vector which is made up of 4 words. All exception vectors are contained in the supervisor data space. When the reset vector is fetched after a RESET, the MMU is disabled and the reset vector is located at physical address 0.

A vector number is an 8-bit number that, when multiplied by 4, gives the address of an exception vector. Vector numbers are generated internally or externally depending on the cause of the exception. In the case of interrupts, during the interrupt acknowledge bus cycle, an external peripheral may send the CPU an 8-bit vector number (see Fig. 9) on the data bus lines D0 to D7. The CPU translates the vector number into the full 24-bit address as shown in Fig. 10. The memory layout for the exception vectors is given in Table 4.

Multiple exceptions

Because more than one exception could occur simultaneously, exceptions are grouped in order of priority (group 0, 1 and 2). Within each group the priority is as shown in Table 3, uppermost with the highest priority.

Table 3 Exception grouping and priority

GROUP	EXCEPTION	PROCESSING
0	RESET ADDRESS ERROR BUS ERROR	Exception processing begins at the next minor cycle.
1	TRACE INTERRUPT ILLEGAL PRIVILEGE	Exception processing begins before the next instruction.
2	TRAP, TRAPV, CHK ZERO DIVIDE FORMAT ERROR	Exception processing is started through normal instruction execution.

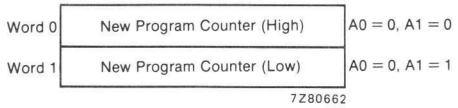


Fig. 8 Exception vector format.

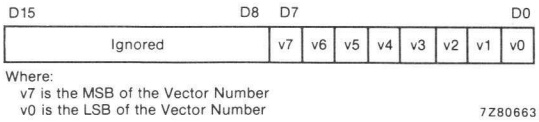


Fig. 9 External peripheral vector format.

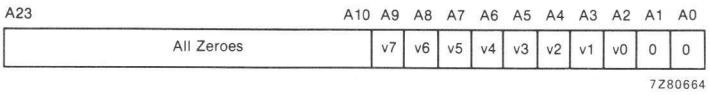


Fig. 10 Address translated from 8-bit vector number.

DEVELOPMENT DATA

Table 4 Exception vector assignment

Vector Number(s)	DEC	HEX	Assignment
0	0	000	Reset: initial SSP
–	4	004	Reset: initial PC
2	8	008	Bus error
3	12	00C	Address error
4	16	010	Illegal instruction
5	20	014	Zero divide
6	24	018	CHK instruction
7	28	01C	TRAPV instruction
8	32	020	Privilege violation
9	36	024	Trace
10	40	028	Line 1010 emulator
11	44	02C	Line 1111 emulator
12*	48	030	(Unassigned, reserved)
13*	52	034	(Unassigned, reserved)
14	56	038	Format error
15	60	03C	Uninitialized interrupt vector
16 – 23*	64	040	(Unassigned, reserved)
	95	05F	–
24	96	060	Spurious interrupt
25	100	064	Level 1 interrupt autovector
26	104	068	Level 2 interrupt autovector
27	108	06C	Level 3 interrupt autovector
28	112	070	Level 4 interrupt autovector
29	116	074	Level 5 interrupt autovector
30	120	078	Level 6 interrupt autovector
31	124	07C	Level 7 interrupt autovector
32 – 47	128	080	TRAP instruction vectors
	191	0BF	
48 – 56*	192	0C0	(Unassigned, reserved)
	227	0E3	
57	228	0E4	Level 1 on-chip interrupt autovector
58	232	0E8	Level 2 on-chip interrupt autovector
59	236	0EC	Level 3 on-chip interrupt autovector
60	240	0F0	Level 4 on-chip interrupt autovector
61	244	0F4	Level 5 on-chip interrupt autovector
62	248	0F8	Level 6 on-chip interrupt autovector
63	252	0FC	Level 7 on-chip interrupt autovector
64 – 255	256	100	User interrupt vectors

* Vectors 12, 13, 16 to 23, and 48 to 56 are reserved for future enhancements. No user peripheral devices should be assigned to these numbers.

RESET OPERATION

When the CPU executes a RESET instruction, the $\overline{\text{RESET}}$ signal is driven LOW for 146 clock cyles to reset internal and external peripherals. In this case the CPU itself is not affected, but all on-chip peripherals are reset. When both the $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ signals are driven LOW by an external device, the CPU and on-chip peripherals are reset. The CPU responds by reading the reset vector table entry (vector number zero, address 000000 HEX) and loads it into the Supervisor Stack Pointer (SSP). Vector table entry number one (at address 000004 HEX) is read next and loaded into the Program Counter (PC). The CPU then initializes the Status Register (SR) to an interrupt level of seven and instruction execution is started (see Fig. 11).

All 3-state output signals are placed in the high-impedance state for as long as the $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ signals are externally driven. When the $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ signals are released, after start-up time the CPU will execute 4 read cycles to load the SSP High, SSP Low, PC High, and PC Low. Then the first instruction is fetched and executed. The $\overline{\text{HALT}}$ signal must be driven LOW at the same time as the $\overline{\text{RESET}}$ signal. The 68070 will only start to read the stack pointer and the initial Program Counter after both signals have been released. $\overline{\text{RESET}}$ should not be released after $\overline{\text{HALT}}$.

When the V_{DD} is initially applied to the 68070, an external RESET must be applied to the $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ pins for at least 100 ms.

BUS ERROR PROCESSING

Like the 68000, the 68070 uses the BUS ERROR ($\overline{\text{BERR}}$) and the $\overline{\text{HALT}}$ signals to distinguish between two bus-error handling routines. If both the $\overline{\text{BERR}}$ and the $\overline{\text{HALT}}$ signals are asserted, the 68070 will rerun the last bus cycle as soon as both the $\overline{\text{HALT}}$ and $\overline{\text{BERR}}$ lines are released. This is valid for both the CPU and the DMA controller. The $\overline{\text{BERR}}$ should be negated before $\overline{\text{HALT}}$ (see Fig. 12).

If just the $\overline{\text{BERR}}$ signal is asserted, bus error exception processing is entered.

If the DMA controller is the affected master, it will:

- stop the service after the current bus cycle is terminated,
- release the bus,
- set the $\overline{\text{BERR}}$ bit in its status word, and
- send an interrupt if the INTERRUPT ENABLE bit in its status word was set.

The address counter reflects the address of the faulty bus cycle, while the transfer counter indicates the number of successful transfers.

If the CPU is the bus master, it will enter the bus error exception processing after the current bus cycle is terminated and $\overline{\text{BERR}}$ has been released. Since the architecture of the 68070 differs from the 68000, it handles bus errors in a different manner. Unlike the 68000, the 68070 allows full recovery from bus errors.

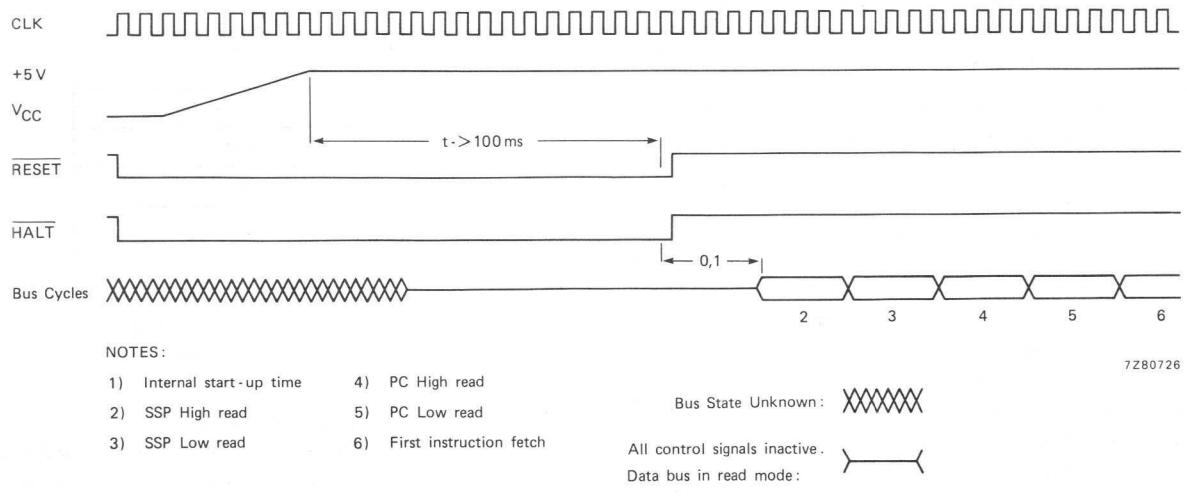


Fig. 11 Reset timing.

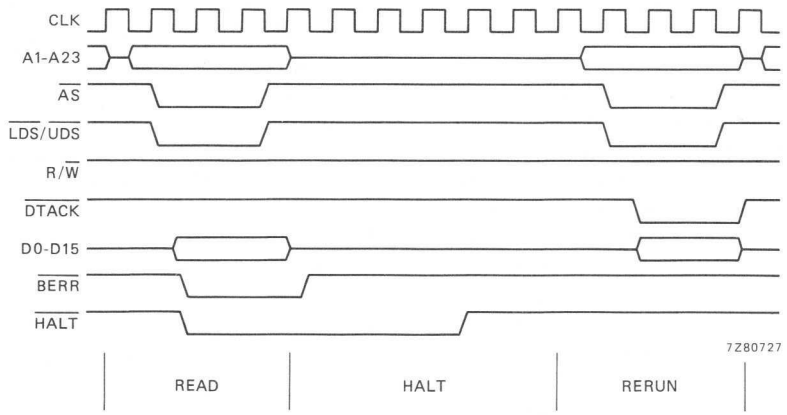


Fig. 12 Rerun bus cycle timing.

The procedure follows the usual sequence of steps:

- the status register is copied,
- the supervisor state is entered,
- the trace state is turned off, and
- the vector number is generated to refer to the bus error vector.

To save more of the context, additional information is saved in the supervisor stack as follows:

- The program counter and a copy of the status registers is saved.
- A format word containing a special bit configuration for the 68070 long stack format, and the vector number of the exception, in this case the bus error exception vector, is stacked.
- Besides other internal information, the processor saves the address which was being accessed by the aborted bus cycle.
- Specific information about the access is either saved or is retrievable from stacked information.
- A special status word is saved to determine the state of the (internal) function codes, the source of the bus error (MMU or External), and whether the error occurred during a read or write cycle, or in the first or second access of a long word operation. A RERUN bit in this special status word has to be set to suppress a retry of the faulty bus cycle on Return from Exception (RTE).
- The instruction registers as well as the temporary registers are saved.

68070 STACK FORMAT

The stack format for exception processing is similar to the 68010 (rather than the 68000) although the information stored is not the same, due to the different architecture.

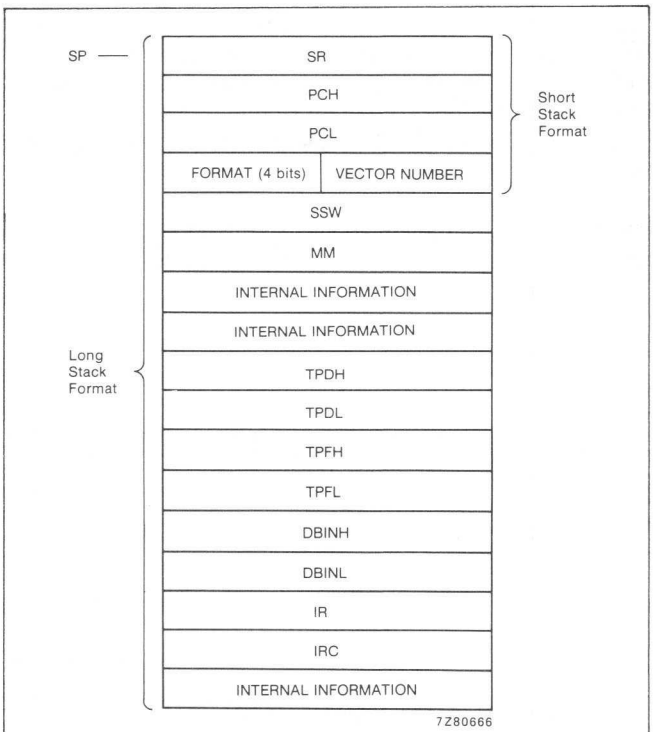


Fig. 13 Stack format.

where:

SR	–	Status Register.
PCH/PCL	–	Program Counter High/Low Word.
Format	–	Indicating either a short stack format (only the first four words), or the long 68070 format for bus and address error exceptions.
Vector Number	–	The vector number of the exception in the vector table; e.g. 2 for a bus error, and 3 for an address error.
SSW	–	Special Status Word.

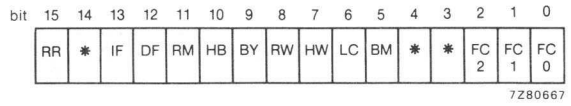


Fig. 14 Special status word.

RR – Rerun. By default, this bit is set to 0. If set 1, the CPU will not rerun the faulty bus cycle on return from exception (RTE).

* – Undefined (reserved).

IF – The faulty cycle was an instruction fetch.

DF – The faulty cycle was a data fetch.

RM – The error occurred during a read-modify-write cycle.

HB – High Byte.

BY – The faulty cycle was a byte transfer.

RW – Read/write cycle.

HW – High Word.

LC – The faulty cycle was during a long-word access.

BM – The bus error was caused by the on-chip MMU.

FC 2, 1, 0 – These bits hold the internal function codes during the faulty bus cycle. The function codes are the same as for the 68000 and affect the status of the CPU during the faulty bus cycle as follows:

FC2	FC1	FC0	
0	0	0	reserved
0	0	1	user data
0	1	0	user program
0	1	1	reserved
1	0	0	reserved
1	0	1	supervisor program
1	1	0	supervisor program
1	1	1	interrupt acknowledge

MM – Current Move Multiple Mask.

TDPH/TDPL – In case of a faulty write cycle, the data can be found here.

TPFH/TPFL – The address used during the faulty bus cycle.

DBINH/DBINL – Data that has been read prior to the faulty cycle can in some cases be found here.

IR – Holds the present instruction executed.

IRC – Holds either the present instruction executed or the prefetched instruction.

To handle this format, the 68070 differs from the 68000 in that:

- the stack format is changed
- the minimum number of words put into or restored from the stack is 4 (68010 compatible; not 3 as for the 68000)
- the RTE instruction decides (with aid of the 4 format bits) whether more information has to be restored, as follows:

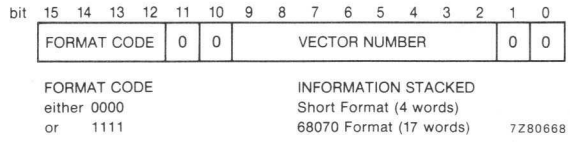


Fig. 15 Vector number and format code.

The 68070 long format is used for bus error and address error exceptions, all other exceptions use the short format

- if another format code other than one of the two listed above is detected during the restore action, a FORMAT ERROR occurs.

If the user want to finish the instruction in which the bus or address error occurred, the 68070 format must be used on RTE. If no changes to the stack are required during exception processing, the stack format is transparent to the user.

INTERRUPT PROCESSING

The 68070 interrupt handling follows the same basic rules as the 68000. However, the following changes have been made to simplify system development:

- the IPL signals have been replaced by decoded interrupt signals $\overline{IN2}$, $\overline{IN4}$, $\overline{IN5}$, and $\overline{NMI}$ (interrupt level 7)
- each of the interrupt inputs has a separate acknowledge signal $\overline{IACK2}$, $\overline{IACK4}$, $\overline{IACK5}$ and $\overline{IACK7}$
- two latched interrupt inputs ($\overline{INT1}$ and $\overline{INT2}$) have programmable priority levels. They have no interrupt acknowledge signal and are always served by auto-vectoring with the vectors located in the on-chip vector table
- interrupt priority levels IPL1, IPL3 and IPL6 are not available externally, unless programmed into $\overline{INT1}$ or $\overline{INT2}$
- if autovectoring is desired, the AUTOVECTOR request signal ($\overline{AV}$) must be asserted during the interrupt acknowledge routine
- to ensure being recognized, an interrupt signal $\overline{IN2}$, $\overline{IN4}$, $\overline{IN5}$ and $\overline{NMI}$ must stay asserted until it is acknowledged by its $\overline{IACK}$x signal.

DEVELOPMENT DATA

If the priority of the interrupt pending is greater than the current processor priority, then:

- the exception processing sequence is started
- a copy of the status register is saved
- the privilege level is set to supervisor state
- tracing is supressed
- the priority level of the processor is set to that of the interrupt being acknowledged.

The processor then gets the vector number from the interrupting device, classifies it as an interrupt acknowledge and displays the interrupt level number being acknowledged on the address bus.

During the interrupt acknowledge cycle, $\overline{LDS}$ is not asserted (as is the case with the 68000). This is done to simplify the address decoding circuitry of the memory. Acknowledge cycle decoding (by the interrupting device) is done using the $\overline{IACK}$x signal instead.

If autovectoring is requested by the external logic, the processor generates a vector number internally that corresponds to the interrupt level number. Then, if a bus error is indicated by the external logic, the interrupt is treated as spurious and the vector number that was generated will refer to the spurious interrupt vector.

Priority level 7 cannot be inhibited by the interrupt priority mask (Non-maskable interrupt – $\overline{NMI}$).

If external and on-chip peripherals are programmed to the same interrupt priority level, an on-chip daisy-chain defines the priority as follows:

external interrupts INx	highest priority
INT1	
INT2	
TIMER	
RS232 RxD	
RS232 TxD	
I^2C	
DMA CH1	
DMA CH2	lowest priority

The two latched interrupt inputs $\overline{INT1}$ and $\overline{INT2}$ have a common Latched Interrupt priority level Register (LIR).

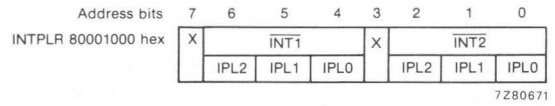


Fig. 16 Latched Interrupt priority level Register (LIR).

CLOCK CIRCUITRY

The clock signals required by each part of the 68070 are generated from the same Master Oscillator running at a frequency of 19,6608 MHz. Dividing this frequency by two gives the clock signals for the CPU, MMU and DMA. Dividing the Master Oscillator signal by four provides the clock signals for the RS232-C and I^2C interfaces, and dividing it by 192 gives the clock for the Timer.

When the Master Oscillator frequency differs from 19,6608 MHz, it is possible to clock the RS232-C interface by switching to a 4,9125 MHz signal supplied to the XCKI pin. Only the RS232-C can operate from the XCKI pin. The Timer will have a period 192 times as long as the Master Oscillator period. As the speed for the I^2C interface is programmable, the interface will have to be programmed with a different division factor.

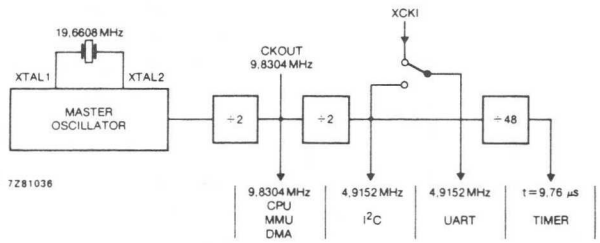


Fig. 17 Clock circuitry.

INSTRUCTION SET AND ADDRESSING MODES

The 68070 is completely code compatible with the 68000, which means that programs developed for the 68000 will run on the 68070. This applies to both source and object code.

The instruction set was designed to minimize the number of mnemonics that the programmer has to remember. Tables 5 and 6 give an overview of the instruction set and of the different addressing modes.

Table 5 Instruction set

Mnemonic	Description	Operation	Condition Codes				
			X	N	Z	V	C
ABCD	Add Decimal with Extend	(Destination)$_{10}$ + (Source)$_{10}$ → Destination	•	U	•	U	•
ADD	Add Binary	(Destination) + (Source) → Destination	•	•	•	•	•
ADDA	Add Address	(Destination) + (Source) → Destination	–	–	–	–	–
ADDI	Add Immediate	(Destination) + Immediate Data → Destination	•	•	•	•	•
ADDQ	Add Quick	(Destination) + Immediate Data → Destination	•	•	•	•	•
ADDX	Add Extended	(Destination) + (Source) + X → Destination	•	•	•	•	•
AND	AND Logical	(Destination) Λ (Source) → Destination	–	•	•	0	0
ANDI	AND Immediate	(Destination) Λ Immediate Data → Destination	–	•	•	0	0
ASL, ASR	Arithmetic Shift	(Destination) Shifted by <count> → Destination	•	•	•	•	•
B_{CC}	Branch Conditionally	If $_{CC}$ then PC + d → PC	–	–	–	–	–
BCHG	Test a Bit and Change	~(<bit number>) OF Destination → Z ~(<bit number>) OF Destination → <bit number> OF Destination	–	–	•	–	–
BCLR	Test a Bit and Clear	~(<bit number>) OF Destination → Z 0 → <bit number> → OF Destination	–	–	•	–	–
BRA	Branch Always	PC + d → PC	–	–	–	–	–
BSET	Test a Bit and Set	~(<bit number>) OF Destination → Z 1 → <bit number> OF Destination	–	–	•	–	–
BSR	Branch to Subroutine	PC → SP@ – ; PC + d → PC	–	–	–	–	–
BTST	Test a Bit	~(<bit number>) OF Destination → Z	–	–	•	–	–
CHK	Check Register against Bounds	If Dn <0 or Dn> (<ea>) then TRAP	–	•	U	U	U
CLR	Clear an Operand	0 → Destination	–	0	1	0	0
CMP	Compare	(Destination) – (Source)	–	•	•	•	•
CMPA	Compare Address	(Destination) – (Source)	–	•	•	•	•
CMPI	Compare Immediate	(Destination) – Immediate Data	–	•	•	•	•
CMPM	Compare Memory	(Destination) – (Source)	–	•	•	•	•
DB_{CC}	Test Condition, Decrement and Branch	If ~ $_{CC}$ then Dn – 1 → Dn; if Dn ≠ – 1 then PC + d → PC	–	–	–	–	–
DIVS	Signed Divide	(Destination)/(Source) → Destination	–	•	•	•	0
DIVU	Unsigned Divide	(Destination)/(Source) → Destination	–	•	•	•	0
EOR	Exclusive OR Logical	(Destination) ⊕ (Source) → Destination	–	•	•	0	0
EORI	Exclusive OR Immediate	(Destination) ⊕ Immediate Data → Destination	–	•	•	0	0
EXG	Exchange Register	Rx ↔ Ry	–	–	–	–	–
EXT	Sign Extend	(Destination) Sign-extended → Destination	–	•	•	0	0
JMP	Jump	Destination → PC	–	–	–	–	–
JSR	Jump to Subroutine	PC → SP@ – ; Destination → PC	–	–	–	–	–
LEA	Load Effective Address	Destination → An	–	–	–	–	–
LINK	Link and Allocate	An → SP@ – ; SP → An; SP + d → SP	–	–	–	–	–
LSL, LSR	Logical Shift	(Destination) Shifted by <count> → Destination	•	•	•	0	•
MOVE	Move Data from Source to Destination	(Source) → Destination	–	•	•	0	0
MOVE to CCR	Move to Condition Code	(Source) → CCR	•	•	•	•	•
MOVE to SR	Move to the Status Register	(Source) → SR	•	•	•	•	•

* affected 0 cleared U defined
– unaffected 1 set

Table 5 Instruction set (continued)

Mnemonic	Description	Operation	Condition Codes				
			X	N	Z	V	C
MOVE from SR	Move from the Status Register	SR → Destination	–	–	–	–	–
MOVE USP	Move User Stack Pointer	USP → An; An → USP	–	–	–	–	–
MOVEA	Move Address	(Source) → Destination	–	–	–	–	–
MOVEM	Move Multiple Registers	Registers → Destination (Source) → Registers	–	–	–	–	–
MOVEP	Move Peripheral Data	(Source) → Destination	–	–	–	–	–
MOVEQ	Move Quick	Immediate Data → Destination	–	*	*	0	0
MULS	Signed Multiply	(Destination)*(Source) → Destination	–	*	*	0	0
MULU	Unsigned Multiply	(Destination)*(Source) → Destination	–	*	*	0	0
NBCD	Negate Decimal with Extend	$0-(\text{Destination})_{10}-X \rightarrow$ Destination	*	U	*	U	*
NEG	Negate	0 – (Destination) → Destination	*	*	*	*	*
NEGX	Negate with Extend	0 – (Destination) – X → Destination	*	*	*	*	*
NOP	No Operation	–	–	–	–	–	–
NOT	Logical Complement	~ (Destination) → Destination	–	*	*	0	0
OR	Inclusive OR Logical	(Destination) v (Source) → Destination	–	*	*	0	0
ORI	Inclusive OR Immediate	(Destination) v Immediate Data → Destination	–	*	*	0	0
PEA	Push Effective Address	Destination → SP@ –	–	–	–	–	–
RESET	Reset External Devices	–	–	–	–	–	–
ROL, ROR	Rotate (Without Extend)	(Destination) Rotated by <count> → Destination	–	*	*	0	*
ROXL, ROXR	Rotate with Extend	(Destination) Rotated by <count> → Destination	*	*	*	0	*
RTE	Return from Exception	SP@ + → SR; SP@ + → PC	*	*	*	*	*
RTR	Return and Restore Condition Codes	SP@ + → CC; SP@ + → PC	*	*	*	*	*
RTS	Return from Subroutine	SP@ + → PC	–	–	–	–	–
SBCD	Subtract Decimal with Extend	$(\text{Destination})_{10}-(\text{Source})_{10}-X \rightarrow$ Destination	*	U	*	U	*
S_{CC}	Set According to Condition	If CC then 1's → Destination else 0's → Destination	–	–	–	–	–
STOP	Load Status Register and Stop	Immediate Data → SR; STOP	*	*	*	*	*
SUB	Subtract Binary	(Destination) – (Source) → Destination	*	*	*	*	*
SUBA	Subtract Address	(Destination) – (Source) → Destination	–	–	–	–	–
SUBI	Subtract Immediate	(Destination) – Immediate Data → Destination	*	*	*	*	*
SUBQ	Subtract Quick	(Destination) – Immediate Data → Destination	*	*	*	*	*
SUBX	Subtract with Extend	(Destination) – (Source) – X → Destination	*	*	*	*	*
SWAP	Swap Register Halves	Register [31:16] ↔ Register [15:0]	–	*	*	0	0
TAS	Test and Set an Operand	(Destination) Tested → CC; 1 → [7] OF Destination	–	*	*	0	0
TRAP	Trap	PC → SSP@ – ; SR → SSP@ – ; (Vector) → PC	–	–	–	–	–
TRAPV	Trap on Overflow	If V then TRAP	–	–	–	–	–
TST	Test an Operand	(Destination) Tested → CC	–	*	*	0	0
UNLK	Unlink	An → SP; SP@ + → An	–	–	–	–	–

[ ] = bit number

* affected 0 cleared U defined
– unaffected 1 set

Table 6 Data addressing modes

Mode	Generation
Register Direct Addressing	
Data Register Direct	EA = Dn
Address Register Direct	EA = An
Absolute Data Addressing	
Absolute Short	EA = (Next Word)
Absolute Long	EA = (Next Two Words)
Program Counter Relative Addressing	
Relative with Offset	EA = (PC) + d_{16}
Relative with Index and Offset	EA = (PC) + (Xn) + d_8
Register Indirect Addressing	
Register Indirect	EA = (An)
Postincrement Register Indirect	EA = (An), An ← An + N
Predecrement Register Indirect	An ← An − N, EA = (An)
Register Indirect with Offset	EA = (An) + d_{16}
Indexed Register Indirect with Offset	EA = (An) + (Xn) + d_8
Immediate Data Addressing	
Immediate	DATA = Next Word(s)
Quick Immediate	Inherent Data
Implied Addressing	
Implied Register	EA = SR, USP, SSP, PC, SP

Notes

EA = Effective address

An = Address register

Dn = Data register

Xn = Address or Data register used as index register

N = 1 for bytes, 2 for words, and 4 for long words

← = Replaces

SR = Status register

PC = Program counter

( ) = Contents of

d_8 = 8-bit offset (displacement)

d_{16} = 16-bit offset (displacement)

SP = Stack pointer

SSP = System stack pointer

USP = User stack pointer

DEVELOPMENT DATA

INSTRUCTION TIMING

Table 7 Effective address calculation times

Addressing Mode		.B .W	.L
Rn	Data or Address Register Direct	0 (0/0)	0 (0/0)
(An)	Address Register Indirect	4 (1/0)	8 (2/0)
(An)+	Address Register Indirect postincrem.	4 (1/0)	8 (2/0)
−(An)	Address Register Indirect predecrem.	7 (1/0)	11 (2/0)
d(An)	Address Register Indirect Displacem.	11 (2/0)	15 (3/0)
d(An,Xi)	Address Register Indirect with Index	14 (2/0)	18 (3/0)
xxx.S	Absolute Short	8 (2/0)	12 (3/0)
xxx.L	Absolute Long	12 (3/0)	16 (4/0)
d(PC)	Program Counter with Displacement	11 (2/0)	15 (3/0)
d(PC,Xi)	Program Counter with Index	14 (2/0)	18 (3/0)
#xxx	Immediate	4 (1/0)	8 (2/0)

Table 8 MOVE Byte and Move Word instruction clock periods

	Rn	(An)	(An)+	−(An)	d(An)	d(An,iX)	xxx.S	xxx.L
Rn	7 (1/0)	11 (1/1)	11 (1/1)	14 (1/1)	18 (2/1)	21 (2/1)	15 (2/1)	19 (3/1)
(An)	11 (2/0)	15 (2/1)	15 (2/1)	18 (2/1)	22 (3/1)	25 (3/1)	19 (3/1)	23 (4/1)
(An)+	11 (2/0)	15 (2/1)	15 (2/1)	18 (2/1)	22 (3/1)	25 (3/1)	19 (3/1)	23 (4/1)
−(An)	14 (2/0)	18 (2/1)	18 (2/1)	21 (2/1)	25 (3/1)	28 (3/1)	22 (3/1)	26 (4/1)
d(An)	18 (3/0)	22 (3/1)	22 (3/1)	25 (3/1)	29 (4/1)	32 (4/1)	26 (4/1)	30 (5/1)
d(An,Xi)	21 (3/0)	25 (3/1)	25 (3/1)	28 (3/1)	32 (4/1)	35 (4/1)	29 (4/1)	33 (5/1)
xxx.S	15 (3/0)	19 (3/1)	19 (3/1)	22 (3/1)	26 (4/1)	29 (4/1)	23 (4/1)	27 (5/1)
xxx.L	19 (4/0)	23 (4/1)	23 (4/1)	26 (4/1)	30 (5/1)	33 (5/1)	27 (5/1)	31 (6/1)
d(PC)	18 (3/0)	22 (3/1)	22 (3/1)	25 (3/1)	29 (4/1)	32 (4/1)	26 (4/1)	30 (5/1)
d(PC,Xi)	21 (3/0)	25 (3/1)	25 (3/1)	28 (3/1)	32 (4/1)	35 (4/1)	29 (4/1)	33 (5/1)
#xxx	11 (2/0)	15 (2/1)	15 (2/1)	18 (2/1)	22 (3/1)	25 (3/1)	19 (3/1)	23 (4/1)

Table 9 MOVE Long instruction clock periods

	Rn	(An)	(An)+	−(An)	d(An)	d(An,iX)	xxx.S	xxx.L
Rn	7 (1/0)	15 (1/2)	15 (1/2)	18 (1/2)	22 (2/2)	25 (2/2)	19 (2/2)	23 (3/2)
(An)	15 (1/0)	23 (3/2)	23 (3/2)	26 (3/2)	30 (4/2)	33 (4/2)	27 (4/2)	31 (5/2)
(An)+	15 (3/0)	23 (3/2)	23 (3/2)	26 (3/2)	30 (4/2)	33 (4/2)	27 (4/2)	31 (5/2)
−(An)	18 (3/0)	26 (3/2)	26 (3/2)	29 (3/2)	33 (4/2)	36 (4/2)	30 (4/2)	34 (5/2)
d(An)	22 (4/0)	30 (4/2)	30 (4/2)	33 (4/2)	37 (5/2)	40 (5/2)	34 (5/2)	38 (6/2)
d(An,Xi)	25 (4/0)	33 (4/2)	33 (4/2)	36 (4/2)	40 (5/2)	43 (5/2)	37 (5/2)	41 (6/2)
xxx.S	19 (4/0)	27 (4/2)	27 (4/2)	30 (4/2)	34 (5/2)	37 (5/2)	31 (5/2)	35 (6/2)
xxx.L	23 (5/0)	31 (5/2)	31 (5/2)	34 (5/2)	38 (6/2)	41 (6/2)	35 (6/2)	39 (7/2)
d(PC)	22 (4/0)	30 (4/2)	30 (4/2)	33 (4/2)	37 (5/2)	40 (5/2)	34 (5/2)	38 (6/2)
d(PC,Xi)	25 (4/0)	33 (4/2)	33 (4/2)	36 (4/2)	40 (5/2)	43 (5/2)	37 (5/2)	41 (6/2)
#xxx	15 (3/0)	23 (3/2)	23 (3/2)	26 (3/2)	30 (4/2)	33 (4/2)	27 (4/2)	31 (5/2)

Table 10 Standard instruction clock periods

	Size	op<ea>,An	op<ea>,Dn	op<ea>,<M>
ADD	.B .W	7+ (1/0)	7+ (1/0)	11+ (1/1)
	.L	7+ (1/0)	7+ (1/0)	15+ (1/2)
AND	.B .W	–	7+ (1/0)	11+ (1/1)
	.L	–	7+ (1/0)	15+ (1/2)
CMP	.B .W	7+ (1/0)	7+ (1/0)	–
	.L	7+ (1/0)	7+ (1/0)	–
DIVS	–	–	169+** (1/0)	–
DIVU	–	–	130+* (1/0)	–
EOR	.B .W	–	7+ (1/0)	11+ (1/1)
	.L	–	7+ (1/0)	15+ (1/2)
MULS	–	–	76+* (1/0)	–
MULU	–	–	76+* (1/0)	–
OR	.B .W	–	7+ (1/0)	11+ (1/1)
	.L	–	7+ (1/0)	15+ (1/2)
SUB	.B .W	7+ (1/0)	7+ (1/0)	11+ (1/1)
	.L	7+ (1/0)	7+ (1/0)	15+ (1/2)

\+ add effective address calculation time
* the duration of the instruction is constant
** indicates maximum value

Table 11 Immediate instruction clock periods

	Size	op<im>,Dn	op<im>,An	op<im>,<M>
ADDI	.B .W	14 (2/0)	–	18+ (2/1)
	.L	18 (3/0)	–	26+ (3/2)
ADDQ	.B .W	7 (1/0)	7 (1/0)	11+ (1/1)
	.L	7 (1/0)	7 (1/0)	15+ (1/2)
ANDI	.B .W	14 (2/0)	–	18+ (2/1)
	.L	18 (3/0)	–	24+ (3/2)
CMPI	.B .W	14 (2/0)	–	14+ (2/0)
	.L	18 (3/0)	–	18+ (3/0)
EORI	.B .W	14 (2/0)	–	18+ (2/1)
	.L	18 (3/0)	–	26+ (3/2)
MOVEQ	.L	7 (1/0)	–	–
ORI	.B .W	14 (2/0)	–	18+ (2/1)
	.L	18 (3/0)	–	26+ (3/2)
SUBI	.B .W	14 (2/0)	–	18+ (2/1)
	.L	18 (3/0)	–	26+ (3/2)
SUBQ	.B .W	7 (1/0)	7 (1/0)	11+ (1/1)
	.L	7 (1/0)	7 (1/0)	15+ (1/2)

+ add effective address calculation time

DEVELOPMENT DATA

Table 12 Single operand instruction clock periods

Instruction	Size	Register	Memory
CLR	Byte, Word	7 (1/0)	11 (1/1)+*
	Long	7 (1/0)	15 (1/2)+**
NBCD	Byte	10 (1/0)	14 (1/1)+
NEG	Byte, Word	7 (1/0)	11 (1/1)+
	Long	7 (1/0)	15 (1/2)+
NEGX	Byte, Word	7 (1/0)	11 (1/1)+
	Long	7 (1/0)	15 (1/2)+
NOT	Byte, Word	7 (1/0)	11 (1/1)+
	Long	7 (1/0)	15 (1/2)+
Scc	Byte, Word	13 (1/0)	17 (1/1)+
	Long	13 (1/0)	14 (1/1)+
TAS	Byte	10 (1/0)	15 (1/1)+*
TST	Byte, Word	7 (1/0)	7 (1/0)+
	Long	7 (1/0)	7 (1/0)+

\+ add effective address calculation time
* subtract one read cycle (–4 (1/0)) from effective address calculation
** subtract two read cycles (–8 (2/0)) from effective address calculation

Table 13 Shift/rotate instruction clock periods

Instruction	Size	Register	Memory
ASR, ASL	Byte, Word	13+3n (1/0)	14 (1/1)+
	Long	13+3n (1/0)	14 (1/1)+
LSR, LSL	Byte, Word	13+3n (1/0)	14 (1/1)+
	Long	13+3n (1/0)	14 (1/1)+
ROR, ROL	Byte, Word	13+3n (1/0)	14 (1/1)+
	Long	13+3n (1/0)	14 (1/1)+
ROXR, ROXL	Byte, Word	13+3n (1/0)	14 (1/1)+
	Long	13+3n (1/0)	14 (1/1)+

+ add effective address calculation time

Table 14 Bit manipulation instruction clock periods

Instruction	Size	Dynamic		Static	
		Register	Memory	Register	Memory
BCHG	Byte		14 (1/1)+		21 (2/1)+
	Long	10 (1/0)		17 (2/0)	
BCLR	Byte		14 (1/1)+		21 (2/1)+
	Long	10 (1/0)		17 (2/0)	
BSET	Byte		14 (1/1)+		21 (2/1)+
	Long	10 (1/0)		17 (2/0)	
BTST	Byte		7 (1/0)+		14 (2/0)+
	Long	7 (1/0)		14 (2/0)	

\+ add effective address calculation time
* indicates maximum value

Table 15 Conditional instruction clock periods

	Displ.	Trap or Branch Taken		Trap or Branch Not taken	
Bcc	.B	13	(1/0)	13	(1/0)
	.L	14	(2/0)	14	(2/0)
BRA	.B	13	(1/0)	–	
	.L	14	(2/0)	–	
BSR	.B	17	(1/2)	–	
	.L	22	(2/2)	–	
DBcc	cc True	–		14	(2/0)
	cc False	17	(2/0)	17	(3/2)

\+ add effective address calculation time
* indicates maximum value

Table 16 JMP, JSR, LEA, PEA, MOVEM instruction clock periods

	Size	(An)	(An)+	−(An)	d(An)	d(An,Xi)	xxx.S	xxx.L	d(PC)	d(PC,Xi)
JMP	–	7	–	–	14	17	14	18	14	17
	–	1/0	–	–	2/0	2/0	2/0	3/0	2/0	2/0
JSR	–	18	–	–	25	28	25	29	25	28
	–	1/2	–	–	2/2	2/2	2/2	2/2	2/2	2/2
LEA	–	7	–	–	14	17	14	18	14	17
	–	1/0	–	–	2/0	2/0	2/0	3/0	2/0	2/0
PEA	–	18	–	–	25	28	25	29	25	28
	–	1/2	–	–	2/2	2/2	2/2	3/2	2/2	2/2
MOVEM	.W	26+7(n)	2+n/0	–	30+7n	33+7n	30+7n	34+7n	30+7n	33+7n
				–	3+n/0	3+n/0	3+n/0	4+n/0	3+n/0	3+n/0
M → R	.L	26+11n	2+2n/0	–	30+11n	33+11n	30+11n	34+11n	30+11n	33+11n
				–	3+2n/0	3+2n/0	3+2n/0	4+2n/0	3+2n/0	3+2n/0
MOVEM	.W	23+7n	–	23+7n	27+7n	30+7n	27+7n	31+7n	–	–
		2/n	–	2/n	3/n	3/n	3/n	4/n	–	–
R → M	.W	23+11n	–	23+11n	27+11n	30+11n	27+11n	31+11n	–	–
		2/2n	–	2/2n	3/2n	3/2n	3/2n	4/2n	–	–

Table 17 Multi-precision instruction clock periods

Instruction	Size	op Dn, Dn	op M, M
ADDX	Byte, Word	7 (1/0)	28 (3/1)
	Long	7 (1/0)	40 (5/2)
CMPM	Byte, Word	–	18 (3/0)
	Long	–	26 (5/0)
SUBX	Byte, Word	7 (1/0)	28 (3/1)
	Long	7 (1/0)	40 (5/2)
ABCD	Byte	10 (1/0)	31 (3/1)
SBCD	Byte	10 (1/0)	31 (3/1)

DEVELOPMENT DATA

Table 18 Miscellaneous instruction clock periods

Instruction	Size	Register	Memory	Register to Memory	Memory to Register
MOVE from SR	–	7 (1/0)	11 (1/1)+	–	–
MOVE to CCR	–	10 (2/0)	10 (2/0)	–	–
MOVE to SR	–	10 (2/0)	10 (2/0)	–	–
MOVEP	Word	–	–	25 (2/2)	22 (4/0)
	Long	–	–	39 (2/4)	36 (6/0)
EXG	–	13 (1/0)	–	–	–
EXT	Word	7 (1/0)	–	–	–
	Long	7 (1/0)	–	–	–
LINK	–	25 (2/2)	–	–	–
MOVE from USP	–	7 (1/0)	–	–	–
MOVE to USP	–	7 (1/0)	–	–	–
NOP	–	7 (1/0)	–	–	–
RESET	–	154 (1/0)	–	–	–
RTE short format long format: no rerun with rerun rerun of TAS	–	 39 (5/0) 140 (18/0) 146 (18/0) 151 (19/1)	–	–	–
RTR	–	22 (4/0)	–	–	–
RTS	–	15 (3/0)	–	–	–
STOP	–	13 (0/0)	–	–	–
SWAP	–	7 (1/0)	–	–	–
UNLK	–	15 (3/0)	–	–	–

+ add effective address calculation time

Table 19 Exception processing clock periods

Exception	Number of clock periods
Address error	158 (3/17)
Bus error	158 (3/17)
Interrupt	65 (4/4)
Illegal instruction	55 (3/4)
Privileged instruction	55 (3/4)
Trace	55 (3/4)

* The interrupt acknowledge bus cycles is assumed to take four external clock periods.

68070 ON-CHIP MMU

The 68070 has an on-chip Memory Management Unit (MMU) that, if enabled, supports virtual memory, multitasking, task protection and dynamic stack allocation. The 68070's MMU is a subset of the Memory Access Controllers in the 68000 family (68910 and 68920) and is therefore software compatible with the contiguous segment mode of these devices.

SEGMENTATION

The MMU divides the memory into segments of multiples of 1 Kbyte (blocks). Two modes are possible:

Mode	Number of segments	Maximum segment length
1	8	2048 blocks = 2 Mbytes
2	128	128 blocks = 128 Kbytes

Memory protection is assigned on a segment to segment basis.

For address translation, the logical addresses can be split into 3 (see Fig. 18):

- segment number,
- displacement and
- offset.

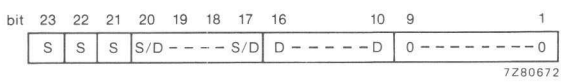


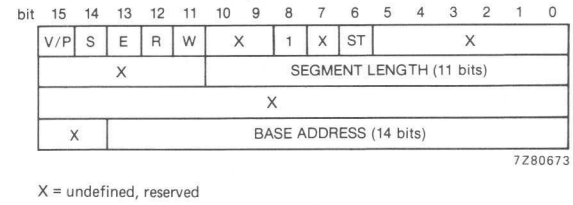
Fig. 18 Logical address format.

where:

S = the segment number
S/D = either the segment number or a displacement, as defined by the MMU Control Register (MCR)
D = displacement
O = offset

SEGMENT DESCRIPTORS

Every segment is described by a segment descriptor stored in the segment descriptor table in main memory. Each descriptor contains a segment address field, a segment protection field (with protection attributes) and the segment length. However, for ease of access, up to 8 descriptors can be stored in the on-chip descriptor RAM. The format of the descriptors in main memory is shown in Fig. 19.



X = undefined, reserved

Fig. 19 Format of descriptors in main memory.

where:

Base Address: the 14 MSBs of the start address of this segment.

Note: All bits indicated as "undefined, reserved" must be programmed to zeros, except bit 8 of the first word which must be set to 1 to be compatible with the MACs in the 68000 family (68910 and 68920).

V/P (Valid/Present) – may be used by the loading software to determine whether the segment is valid and present.
S (Supervisor) – supervisor permission is required to access this segment.
E (Execute) – instruction fetches may be performed with this segment.
R (Read) – read operations may be performed with this segment.
W (Write) – write operation may be performed with this segment.
ST (Stack) – stack segments grow from high to low addresses.
Segment Length: the 11 bits define the length of a segment by the number of 1 Kbyte blocks. (If the SN bit of the MMU Control Register (MCR) is set, placing the MMU in mode 2, only 7 MSBs define the length of the segment.)

The format of the descriptor in the on-chip descriptor RAM is shown in Fig. 20 (four consecutive words for each descriptor).

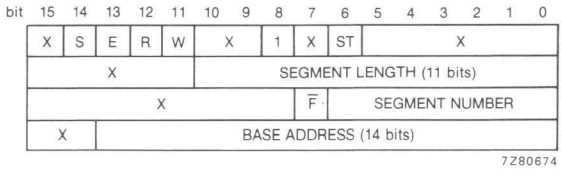


Fig. 20 Descriptor format in RAM.

$\overline{F}$ = FLUSH: If this bit is equal to zero it invalidates the descriptor. Therefore, it must be set to one when writing a new valid segment number. The $\overline{F}$ bit should be set to one after the remainder of the segment descriptor has been loaded into the MMU.

X = undefined, reserved; will return zeros when read by the CPU.

The low-order byte of the third word contains the segment number that is loaded into the fully associative CAM.

MMU CONTROL AND STATUS REGISTERS

The format of the MMU's control (MCR) and the status (MSR) registers are shown in Fig. 21. The mnemonics used in Fig. 20 are defined in Table 20.

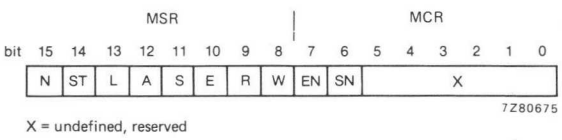


Fig. 21 MMU Status and Control Registers (MSR and MCR).

The registers can be accessed as either one word or two separate bytes. MCR is a read/write register, while MSR is read-only and a write to MSR will result in a normal bus cycle with no effect.

Table 20 Control and status register bits

	bit number	symbol	description		
MSR	15	N	Not-present	:	'1' addressed segment has no descriptor in the MMU (or $\overline{F}$ = '0')
	14	ST	Stack segment	:	if set the segment grows from the highest to the lowest address
	13	L	Length violation	:	if set, a length violation has occurred
	12	A	Access error	:	if set, an attribute violation has occurred
	11	S	Supervisor bit		Reflects the descriptor attributes when an access error occurred
	10	E	Execute bit		
	9	R	Read bit		
	8	W	Write bit		
MCR	7	EN	Enable bit	:	'1' enables the MMU (address translation and protection)
	6	SN	Number of segments	:	'0' = mode 1; '1' = mode 2

ADDRESS MAP OF THE MMU

The internal addresses of the MMU's Control Register (MCR), Status Register (MSR), and Descriptor RAM are as follows:

Base address = 80008000 (HEX)

A6	A5	A4	A3	A2	A1	A0	
0	0	0	0	0	0	0	MSR
0	0	0	0	0	0	1	MCR
1	C	C	C	0	0	0/1	ATTR
1	C	C	C	0	1	0/1	SEG LENGTH
1	C	C	C	1	0	0	UNDEFINED, RESERVED
1	C	C	C	1	0	1	SEG NUMBER
1	C	C	C	1	1	0/1	BASE ADDR

Notation:
MSR = MMU Status Register
MCR = MMU Control Register
SEG = Segment
ATTR = Attributes
CCC = 000 to 111 (i.e. Segment Descriptors 0 to 7)

All internal MMU addresses are mapped in the 68070's on-chip address space (also see Tables 1 and 2) and are only accessible in supervisor mode. On-chip addresses are not processed by the MMU.

OPERATION

The MMU provides protection and virtual memory support. When the MMU is enabled, it only influences addresses sent by the CPU and adds two states (or one clock cycle) at the beginning of each external bus cycle of the CPU. However, during the vector acquisition of an interrupt acknowledge cycle, the MMU does not process the address coming from the CPU; i.e. logical address equals physical address.

If an error occurs within the enabled MMU (length or attribute violation, or segment not-present), the MMU inhibits data strobes ($\overline{UDS}$, $\overline{LDS}$) and asserts a bus error signal to both the CPU and the external world ($\overline{BERR}$). The address strobe ($\overline{AS}$) will still be asserted to indicate bus occupation to other possible bus masters. Then, the CPU will start bus error exception processing with the stack operation.

DMA CONTROLLER

GENERAL

The 68070 has on-chip, a two-channel DMA controller that handles byte or word operands and devices with port sizes of 8 or 16-bits. The channels can be programmed to transfer data in cycle-stealing (single cycle) or burst (a block of successive cycles) mode. Channel 1 always uses single addressing, while channel 2 can operate with single or dual addresses (memory to memory).

Typical latency times are less than 2,5 μs and the maximum transfer rate (using single addressing) is 1,6 million transfer/s.

The 68070's DMA controller is a subset of the existing DMA controllers in the 68000 family (68430, 68440, and 68450) and is therefore software compatible with these devices.

BUS ARBITRATION AND PRIORITY RESOLUTION

The 68070 contains three possible bus masters, the DMA channels 1 and 2 and the CPU, where channel 1 has the highest priority, and the CPU the lowest. There can also be external bus masters which have a priority lower than channel 2 but higher than the CPU (which has the lowest priority in the system). The priority is defined by connecting a daisy-chain to the internal bus grant signal of the CPU, through the internal bus grant signals of channel 1 and channel 2, and then to the external devices.

When a valid transfer request is received from a device, the DMA controller will arbitrate for and acquire the bus. The DMA controller indicates to the CPU that it wants to become bus master by generating an internal bus request signal, and the CPU responds by sending an internal bus grant signal, daisy-chained through the DMA controller. In this case, it is not offered to the external devices because the DMA controller has a request pending. If $\overline{BG}$ has already been offered to the external devices when a DMA request arrives, the DMA controller waits for $\overline{BGACK}$ to be negated.

If the CPU is the current bus master, it will finish its current bus cycle and then give the bus to the DMA controller. When the DMA controller has received the internal bus grant signal, it waits for the external signals, Address Strobe ($\overline{AS}$) and Data Transfer Acknowledge ($\overline{DTACK}$), to become inactive before assuming bus ownership.

If the $\overline{BR}$ signal is asserted by an external device when the CPU is not the bus master (i.e. $\overline{BGACK}$ is asserted) no $\overline{BG}$ signal will be sent until the CPU has regained the bus (i.e. $\overline{BGACK}$ is negated).

DEVICE/DMA CONTROLLER COMMUNICATION

The following five signal lines let the peripheral devices and the DMA controller communicate (see the pin description for further details):

- Request ($\overline{REQx}$). The device makes a request for service by asserting the $\overline{REQx}$ line. Either burst or cycle-stealing mode can be used.
- Acknowledge ($\overline{ACKx}$). The channel asserts the acknowledge line (which implicitly addresses the device making the request) during transfers to and from the device.
- Ready ($\overline{RDY}$). $\overline{RDY}$ is an active-LOW input which is asserted by the requesting device in single-address mode. Bit 8 in the Channel Status Register reflects the current state of $\overline{RDY}$.
- Device Transfer Complete ($\overline{DTC}$). $\overline{DTC}$ is an active-LOW output asserted by the DMA controller during device bus-cycles to indicate that the cycle has been completed successfully.
- Done ($\overline{DONE}$). $\overline{DONE}$ is a bidirectional active-LOW signal. As an output, it indicates to the device that the memory transfer count is exhausted. As an input, it indicates that the operation will terminate after current operand has been transferred (also see **Termination phase**).

DEVELOPMENT DATA

DMA CONTROL AND STATUS REGISTERS

Table 21 DMA controller address map

Base Address 80004000 (HEX) Address bits[1], [2] 7	6	5	4	3	2	1	0	Acronym	Register Name	Mode	Affected by RESET
c	c	0	0	0	0	0	0	CSR	Channel Status Register	R/W[3]	Yes
c	c	0	0	0	0	0	1	CER	Channel Error Register	R	Yes
c	c	0	0	0	0	1	0		Reserved		
c	c	0	0	0	0	1	1		Reserved		
c	c	0	0	0	1	0	0	DCR	Device Control Register	R/W	Yes
c	c	0	0	0	1	0	1	OCR	Operation Control Register	R/W	Yes
c	c	0	0	0	1	1	0	SCR	Sequence Control Register	R/W[4]	Yes
c	c	0	0	0	1	1	1	CCR	Channel Control Register	R/W	Yes
c	c	0	0	1	0	0	0		Reserved		
c	c	0	0	1	0	0	1		Reserved		
c	c	0	0	1	0	1	0	MTCH	Memory Transfer Counter High	R/W	No
c	c	0	0	1	0	1	1	MTCL	Memory Transfer Counter Low	R/W	No
c	c	0	0	1	1	0	0	MACH	Memory Address Counter High	R/W[4]	No
c	c	0	0	1	1	0	1	MACMH	Memory Address Counter Middle High	R/W	No
c	c	0	0	1	1	1	0	MACML	Memory Address Counter Middle Low	R/W	No
c	c	0	0	1	1	1	1	MACL	Memory Address Counter Low	R/W	No
c	c	0	1	0	0	d	d		Reserved		
c	c	0	1	0	1	0	0	DACH	Device Address Counter High	R/W[4], [5]	No
c	c	0	1	0	1	0	1	DACMH	Device Address Counter Middle High	R/W[5]	No
c	c	0	1	0	1	1	0	DACML	Device Address Counter Middle Low	R/W[5]	No
c	c	0	1	0	1	1	1	DACL	Device Address Counter Low	R/W[5]	No
c	c	0	1	1	d	d	d		Reserved		
c	c	1	0	0	d	d	d		Reserved		
c	c	1	0	1	0	d	d		Reserved		
c	c	1	0	1	1	0	0		Reserved		
c	c	1	0	1	1	0	1	CPR	Channel Priority Register	R/W[4]	No
c	c	1	0	1	1	1	0		Reserved		
c	c	1	0	1	1	1	1		Reserved		
c	c	1	1	d	d	d	d		Reserved		

Notes: [1]) 'cc' = 00 for channel 1, 'cc' = 01 for channel 2, 'cc' = 10 or 11 reserved.
[2]) 'd' designates don't care.
[3]) A write to this register may perform a status reset operation.
[4]) This is a dummy register present only to provide compatibility with other 68000 family DMA controllers. A write to this register has no effect on the DMA controller.
[5]) Channel 2 only.

To be compatible with other 68000 family DMA controllers, the control and status bits are mapped into bit positions equivalent to the register map of the other devices. Bits which are used in the other devices but not in the DMA controller are assigned default values. If upward compatibility with the other devices is required, the programmer should use these default values when writing control words into the registers although they have no effect in the DMA controller. When a register is read, the default value is returned regardless of the value used when the register was programmed. The default value is indicated by '(x)' in unused bit positions in the register formats that follow.

REGISTER FORMATS

Device control register (DCR)

CHANNEL 1

	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 09	bit 08
DCR	External Request Mode 0 = Burst 1 = Cycle Steal	Not Used (0)	Device Type ACK/RDY Device (1)	(1)	Not Used (*)	Not Used (0)	Not Used (0)	Not Used (0)

* Must be 0 if OCR 5:4 = 00 (SIZE), otherwise 1. When read, the value of this bit is (OCR 5.OR.OCR 4).

CHANNEL 2

	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 09	bit 08
DCR	External Request Mode 0 = Burst 1 = Cycle Steal	Not Used (0)	Device Type 00 = 68000 Device 01 = Reserved 10 = Reserved 11 = ACK/RDY Device		Device Size * 0 = 8 bit Port 1 = 16 bit Port	Not Used (0)	Not Used (0)	Not Used (0)

7Z80680

* This bit is programmable only when DCR 13:12 = 00 (68000 device). For DCR 13:12 = 11 ($\overline{ACKx}/\overline{RDY}$ Device) the function is identical to channel 1.

Fig. 22 Device Control Registers (DCR).

DEVELOPMENT DATA

Device Type (DCR 13:12)

These bits determine how the device is addressed, as follows:

CHANNEL 1.

11 The device connected is implicitly addressed by the 5 device control signals with handshaking using $\overline{\text{ACKx}}$ and $\overline{\text{RDY}}$.

CHANNEL 2.

00 The device is explicitly addressed by the Device Address Counter (DAC) via the 68070 bus interface. Transfers are made in two bus cycles and the data is stored in the CPU between bus cycles.

11 The connected device is implicitly addressed by the 5 device control signals with handshaking using $\overline{\text{ACKx}}$ and $\overline{\text{RDY}}$.

Device Size (DCR 11, Channel 2 only)

This bit functions only with explicitly addressed devices, DCR 13:12 = 00.

0 The device port size is 8-bit, and the device reads/writes only using the low-order half of the bus (D0-D7) or the high-order (D8-D15) half depending on bit A0 of the Device Address Counter (DAC). Depending on which part of the bus is used either $\overline{\text{LDS}}$ for the lower-order part or $\overline{\text{UDS}}$ for the upper-order part is asserted. During byte size transfers (OCR 5:4 = 00) the half of the data bus used for read/write to the memory depends on the state of A0 of the Memory Address Counter (MAC).

During word size transfers (OCR 5:4 = 01), read/write operations to the device take place in two successive bytes; the DAC will either be unchanged or incremented by two per byte, depending on SCR 9:8 (see Fig. 4.3).

1 The device port size is 16-bit, and the device is accessed as a normal memory location.

Operation control register (OCR)

CHANNEL 1

	bit 07	bit 06	bit 05 bit 04	bit 03	bit 02	bit 01	bit 00
OCR	Direction 0 = Mem. to Dev. 1 = Dev. to Mem.	Not Used (0)	Operand Size 00 = Byte 01 = Word (16-bit) 10 = Reserved 11 = Reserved	Not Used (0)	Not Used (0)	Not Used (1)	Not Used (0)

CHANNEL 2

	bit 07	bit 06	bit 05 bit 04	bit 03	bit 02	bit 01	bit 00
OCR	Direction 0 = Mem. to Dev. 1 = Dev. to Mem.	Not Used (0)	Operand Size 00 = Byte 01 = Word (16-bit) 10 = Reserved 11 = Reserved	Not Used (0)	Not Used (0)	Not Used (1)	Not Used (0)

7Z80681

Fig. 23 Operation Control Registers (OCR).

Operand size (OCR 5:4)

The values of these bits determines whether $\overline{UDS}$, $\overline{LDS}$, or both are generated during the transfer cycle and what value by which the address counters MAC and DAC (AC) are incremented each transfer cycle.

Sequence control register (SCR)

CHANNEL 1

	bit 15	bit 14	bit 13	bit 12	bit 11 bit 10	bit 09 bit 08
SCR	Not Used (0)	Not Used (0)	Not Used (0)	Not Used (0)	MAC Count Up (0) (1)	DAC Not Used (0) (0)

CHANNEL 2

	bit 15	bit 14	bit 13	bit 12	bit 11 bit 10	bit 09 bit 08
SCR	Not Used (0)	Not Used (0)	Not Used (0)	Not Used (0)	MAC 00 = No Change 01 = Count Up 10 = Reserved 11 = Reserved	DAC 00 = No Change 01 = Count Up 10 = Reserved 11 = Reserved

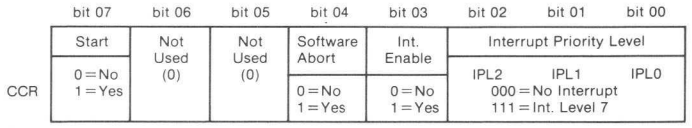
7Z80682

Fig. 24 Sequence Control Registers (SCR).

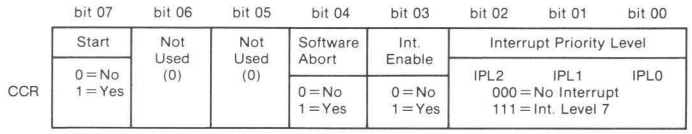
DEVELOPMENT DATA

Channel control register (CCR)

CHANNEL 1



CHANNEL 2



7Z80683

Fig. 25 Channel Control Registers (CCR).

Start Operation (CCR 7)

0 No Start pending.

1 Start Operation. The start bit is set to initiate channel operation.

Software Abort (CCR 4)

0 Do Not Abort.

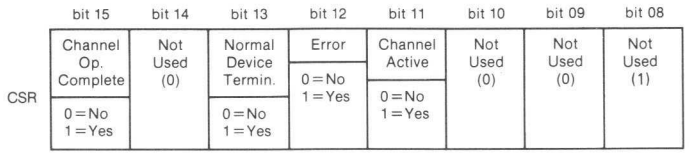
1 Abort Operation. Setting this bit terminates the current operation and puts it into the Idle state, then:

- The COC bit (CSR 15) and ERR bit (CSR 12) are set.
- The Channel Active bit (CSR 11) in the CSR is reset.
- An Abort Error condition is signalled in the CER.

Setting this bit causes a pending start to be reset. When reading CCR, this bit always appears as a zero.

Channel status register (CSR)

CHANNEL 1



CHANNEL 2

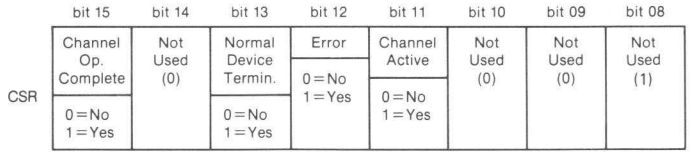


Fig. 26 Channel Status Registers (CSR).

Reading this register gives the status of the channel. The COC bit (CSR 15), NDT bit (CSR 13), and ERR bit (CSR 12) can be cleared by writing a '1' to the appropriate bit positions of the register. Writing a '0' to these bit positions has no effect.

Channel error register (CER)

CHANNEL 1 and 2

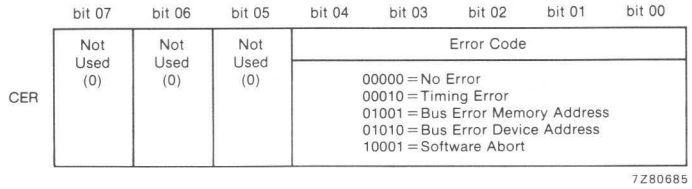


Fig. 27 Channel Error Registers (CER).

DEVELOPMENT DATA

Error Code (CER 4:0)

This field indicates the source of the error when the ERR bit (CSR 12) is set. Clearing bit 12 of the CSR clears the contents of the Channel Error Register.
The error codes are:

00000 No Error.

00010 Timing Error. An attempt has been made to start the channel before all the bits of the CSR had been cleared.

01001 Bus Error memory side. A bus error ($\overline{BERR}$ asserted without $\overline{HALT}$) occurred during the cycle with MAC presenting the address.

01010 Bus Error device side. A bus error ($\overline{BERR}$ asserted without $\overline{HALT}$) occurred during the cycle with DAC presenting the address (channel 2 only).

10001 Software Abort. The channel operation was terminated by a software Abort command (CCR 4).

Channel priority register (CPR)

CHANNEL 1

	bit 07	bit 06	bit 05	bit 04	bit 03	bit 02	bit 01	bit 00
CPR	Not Used (0)	Not Used (0)	Not Used (0)	Not Used (0)	Not Used (0)	Not Used (0)	Priority (0)	(0)

CHANNEL 2

	bit 07	bit 06	bit 05	bit 04	bit 03	bit 02	bit 01	bit 00
CPR	Not Used (0)	Not Used (0)	Not Used (0)	Not Used (0)	Not Used (0)	Not Used (0)	Priority (0)	(1)

7Z80686

Fig. 28 Channel Priority Registers (CPR).

Memory address counter (MACH, MACMH, MACML, MACL)

The 32-bit memory address counter defines a memory location as the first source operand to be transferred or its destination, depending on the direction of the transfer.

Only the least significant 24 bits of the counter (MACMH, MACML, and MACL) are implemented in the DMA controller.

Device address counter (DACH, DACMH, DACML, DACL) – Channel 2 only

The 32-bit device address counter defines a device location as the source of the first operand to be transferred or its destination, depending on the direction of the transfer.

Only the least significant 24 bits of the counter (DACMH, DACML, and DACL) are implemented in the DMA controller.

Memory transfer counter (MTCH, MTCL)

The 16-bit memory transfer counter defines the number of operands to be transferred by the channel.

OPERATION

The operation of the DMA channels can be divided into these phases:

- Initialization,
- Transfer, and
- Termination.

Initialization

After programming the Channel Control Registers (CCR), the memory and device address counters, and the memory transfer counter, the CPU sets the START bit (CCR 7). The channel initializes the operation by clearing any pending requests, clearing the START bit, and setting the Channel Active bit in the Channel Status Register (CSR). The channel is then ready to receive valid requests for an operation.

Transfer phase

The actual transfer of data between the device and the memory occurs during this phase in one of two ways. In single address mode, the transfers occur during a single bus cycle, while in double address mode, each transfer is performed with a read and a write bus cycle.

Termination phase

The termination phase of the block transfer occurs under the following conditions.

Count Termination

As a part of operand transfer, the channel decrements the Memory Transfer Counter (MTC). When this counter has been decremented to zero, it indicates that this is the last operand transfer and that the channel operation is complete. $\overline{\text{DONE}}$ is asserted, CSR 11 is cleared and CSR 15 is set.

Device Termination

The channel monitors the state of the $\overline{\text{DONE}}$ line while acknowledging a device transfer request. If the device asserts $\overline{\text{DONE}}$, the channel will terminate the operation after transferring the current operand. CSR 11 is cleared, and CSR 13 and CSR 15 are set.

Software Abort

The Software Abort bit (CCR 4) allows the CPU to abort the current channel operation (see description of CCR 4).

Bus error treatment

If both the $\overline{\text{BERR}}$ and $\overline{\text{HALT}}$ signals are asserted during a DMA controller cycle, the DMA controller will enter the RERUN state.

If only the $\overline{\text{BERR}}$ signal is asserted during a DMA controller cycle, the channel stops DMA controller operation, releases the bus, sets the ERR (CSR 12) and COC (CSR 15) bits in the CSR, clears the Channel Active bit (CSR 11) in the CSR, and sets the error code in the CER to indicate a bus error.

Reset

Via RESET, external sources and CPU programs can reset and initialize the DMA controller. If the DMA controller is the bus master when reset is detected, it releases the bus and resets all the bits of the status registers (except CSR 8) to zero.

Interrupts

If the Interrupt Enable bit (CCR 3) is set, the channel will send an interrupt when it terminates an operation (COC bit set – CSR 15). The priority level of the interrupt is given by the IPL bits in the CCR. During the interrupt acknowledge cycle, the channel requests an autovector and thus the IPL bits correspond directly to the vector used.

THE 68070'S PERIPHERAL INTERRUPT CONTROL

The I^2C and RS232-C serial interfaces, and the Timer, use a common set of Peripheral Interrupt Control Registers PICR. These registers are memory mapped on the on-chip bus and communicate with the CPU of the 68070.

Address 80002045 (HEX)

ADDRESS								bit	7	6	5	4	3	2	1	0
A6	A5	A4	A3	A2	A1	A0	*			I^2C				Timer		
1	0	0	0	1	0	1		PICR1	X	IPL 2	IPL 1	IPL 0	X	IPL 2	IPL 1	IPL 0

Address 80002047 (HEX)

ADDRESS								bit	7	6	5	4	3	2	1	0
A6	A5	A4	A3	A2	A1	A0	*			RS232-C RxD				RS232-C TxD		
1	0	0	0	1	1	1		PICR2	X	IPL 2	IPL 1	IPL 0	X	IPL 2	IPL 1	IPL 0

7Z80713

* Note: all locations with A0 = 0 are reserved and undefined.

I^2C	I^2C serial bus interface.
RS232	RS232-C serial interface.
RxD	RS232-C receiver.
TxD	RS232-C transmitter.
Timer	Timer functions.
IPL	Interrupt priority level of interrupts requested by the I^2C, RS232, and the Timer. IPL2 is the MSB, and IPL0 is the LSB. All values are positive true, IPL0 – IPL2 = 111 represents the highest priority level 7. IPL0 – IPL2 = 000 will inhibit interrupts.
X	undefined, reserved.

Fig. 29 Peripheral Interrupt Control Registers (PICR).

Initially and after RESET, all PICRx bits are cleared to zero.

THE I²C SERIAL BUS INTERFACE

The 68070 has an on-chip interface to the inter-IC (I^2C) bus. The I^2C-bus can be used in a master or slave mode, and can be connected to up to 128 different peripheral ICs, each with a unique device address. Maximum transmission speed is 100 kbits/s.

Communication with the bus is via two dedicated pins, serial clock SCL, serial clock and SDA-serial data.
The interface can generate interrupts with priority programmed to one of 7 levels. Interrupts from I^2C-bus slave peripherals are handled via two general-purpose latched interrupt request lines of programmable priority. A complete data transfer on the I^2C-bus is shown in Fig. 30.

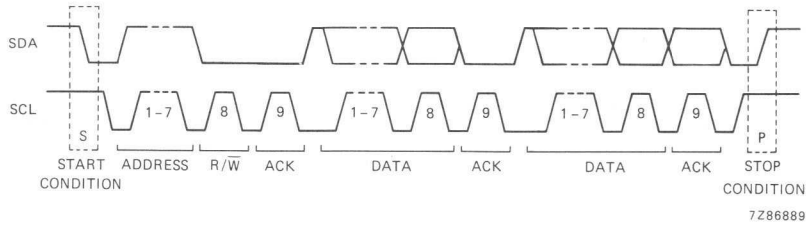


Fig. 30 A complete data transfer on the I^2C bus.

OPERATING MODES

The CPU can operate in the following modes with the serial I^2C-bus:

- master transmitter (MTX).
- master receiver (MRX).
- slave transmitter (STX).
- slave receiver (SRX).

THE 68070'S I²C-BUS I/O REGISTERS

The communication between the 68070's CPU and I^2C-bus interface is via a set of registers and an interrupt request facility. The data and information controlling the operation of the interface is stored in the following four registers (that are fully transparent and memory mapped to the CPU):

Base address: 80002001 (HEX)

A3	A2	A1	A0*	
0	0	0	1	I^2C Data Register (IDR)
0	0	1	1	I^2C Address Register (IAR)
0	1	0	1	I^2C Status Register (ISR)
0	1	1	1	I^2C Control Register (ICR)
1	0	0	1	I^2C Clock Control register (ICC)

* Note: All locations with A0* = 0 are undefined/reserved.

I²C Data Register (IDR)

The data register IDR performs the conversion between the serial and parallel data formats. Data to be transmitted is loaded into IDR by the CPU and is shifted out serially (MSB first), and data received on the serial bus is shifted into IDR (MSB first).

I²C Address Register (IAR)

The address register holds the slave address allocated to the device in its 7 MSB's. It is only written to by the CPU and remains unchanged until rewritten. The LSB is undefined/reserved and should be set to zero.

A6	A5	A4	A3	A2	A1	A0	0

Fig. 31 I²C Address Register (IAR).

I²C Status Register (ISR)

Status register ISR contains all the information concerning the status of the I²C-bus interface, and all its bits can be written and read by the CPU. The functions of the status bits illustrated in Fig. 32 are given below:

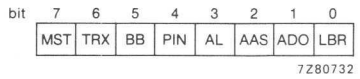


Fig. 32 I²C Status Register (ISR).

MST = Master
If this bit is 1, the I²C-bus interface is in the master mode and it generates clock pulses on SCL for transmission/reception timing of serial data. If the MST bit is 0, the I²C-bus interface is in the slave mode and the clock pulses are received from the master on SCL.

TRX = Transmitter
If this bit is 1, the I²C-bus interface is in the transmitter mode and data in the IDR register is shifted out onto the data line SDA, synchronized with the clock pulses on SCL. If this bit is 0, the I²C-bus interface is in the receiver mode and data on the data line SDA is shifted into the IDR synchronized with the clock pulses on SCL.

BB = Bus Busy
This bit indicates the state of the serial bus; if 0, the bus is free and if 1, the bus is busy.

PIN = Pending Interrupt Not
The PIN bit is set to 0 every time an I²C-bus interrupt is requested.

AL = Arbitration Lost
The AL bit generally indicates the detection of an error.

AAS = Addressed As Slave
AAS is set to 1 when the address comparator recognizes either its own slave address or the general call address (8 zeros). It is reset to 0 when the CPU sets the PIN bit to 1.

ADO = Address Zero
ADO is set to 1 if the address comparator detects the general call address (8 zeros).

LRB = Last Received Bit
The LRB position in ISR of the transmitter contains the receiver acknowledge. When it is 0, the reception of the transmission has been acknowledged.

I²C Control Register (ICR)

Some additional functions of the interface can be controlled with the flags in the control register ICR as shown in Fig. 33.

Fig. 33 I²C Control Register (ICR).

SEL = Selected
The SEL bit is set together with the AAS bit in ISR and remains at 1 during the whole transfer. It is reset when a STOP or repeated START condition is detected.

ESO = Enable I²C-bus
If the ESO bit is set, the I²C-bus interface is enabled and when it is reset the I²C-bus is disabled. Only the CPU can alter the ESO bit.

ACKS = Acknowledge
- ACKS = 1 Reception will be acknowledged by a '0' bit.
- ACKS = 0 Reception will not be acknowledged, a '1' bit is sent.

I²C Clock Control register (ICC)

7	6	5	4	3	2	1	0
X	X	X	CK4	CK3	CK2	CK1	CK0

X = undefined/reserved

Fig. 34 I²C Clock Control register (ICC).

By programming the 5 LSBs of the clock control register, the frequency of SCL and SDA can be adapted to the needs of the I²C-bus or the 68070's system clock. After initialization or RESET, bits CK4 to CK0 are cleared to zero.

Table 22 I^2C-bus interface divisors

CK4 – CK0 (HEX)	divisor	approximate SCL frequency (kHz) (crystal or clock input frequency = 19,66 MHz)
0	illegal	–
1	78	126,025*
2	90	109,222*
3	102	96,372
4	126	78,015
5	150	65,533
6	174	56,494
7	198	49,464
8	246	39,959
9	294	33,435
A	342	28,742
B	390	25,205
C	486	20,266
D	582	16,890
E	678	14,498
F	774	12,700
10	966	10,175
11	1158	8,488
12	1350	7,281
13	1542	6,374
14	1926	5,103
15	2310	4,255
16	2694	3,648
17	3078	3,193
18	3846	2,555
19	4614	2,130
1A	5382	1,826
1B	6150	1,598
1C	7686	1,278
1D	9222	1,065
1E	10758	0,913
1F	12294	0,799

* Note: Maximum bus clock frequency in I^2C systems is 100 kbits/s.

DEVELOPMENT DATA

RS232-C SERIAL INTERFACE

The RS232-C interface is a universal asynchronous data communication controller that interfaces directly with the CPU and can be used in either polled or interrupt driven modes. It accepts programmed instructions from the CPU while supporting asynchronous serial data communication in either full or half-duplex mode. The interface then converts data received from the CPU into a serial form for transmission, and simultaneously, it can receive serial data and convert it to parallel data as input to the CPU. Two baud rate generators can be programmed to generate transmit/receive baud rates by either using the 68070's system clock, or accepting an external clock for special baud rates.

FUNCTIONAL BLOCKS

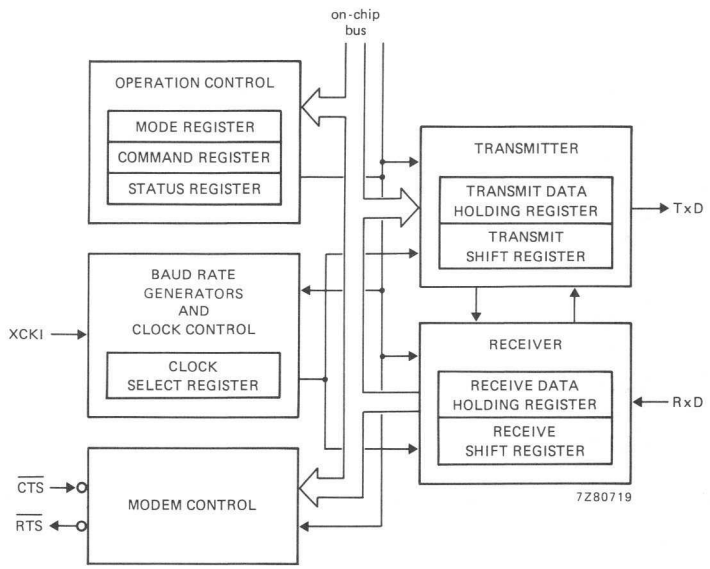


Fig. 36 RS232 Mode Register (RMR).

PROGRAMMING

Before initiating data communication, the RS232 operation mode must be programmed by writing to the mode, clock select and command registers. The RS232-C interface can be re-configured at any time during program execution.

Table 23 RS232-C register addressing

Base Address = 80002011

A3	A2	A1	A0*	Function
0	0	0	1	Receive Holding Register (RHR)
0	0	1	1	Transmit Holding Register (THR)
0	1	0	1	RS232 Status Register (RSR)
0	1	1	1	Undefined, reserved
1	0	0	1	RS232 Mode Register (RMR)
1	0	1	1	Undefined, reserved
1	1	0	1	RS232 Command Register (RCR)
1	1	1	1	Clock Select Register (CLS)

* Note, all locations with A0 = 0 are undefined, reserved.

bit 07 bit 06	bit 05	bit 04	bit 03	bit 02	bit 01	bit 00
Channel mode	**Not used**	**$\overline{CTS}$ enable TxD**	**Parity control**	**Parity type**	**Stop bit length**	**Character length**
00 = normal 01 = auto echo 10 = local loopback 11 = remote loopback	(0)	0 = no $\overline{CTS}$ control 1 = $\overline{CTS}$ control TxD	0 = inhibited 1 = enabled	0 = odd 1 = even	0 = one stop bit 1 = two stop bits	0 = 7 bits 1 = 8 bits

Fig. 36 RS232 Mode Register (RMR).

Mode register (RMR)

RMR 0 selects the character length (7 or 8 bits). This doesn't include the parity bit (if programmed) or the START/STOP bits.

RMR 1 selects the number of STOP bits; either 1 or 2.

RMR 2 selects either odd or even parity when parity has been enabled by RMR 3.

RMR 3 controls the parity generation and when enabled, a parity bit is added to the transmitted character and the receiver performs a parity check on the incoming data.

RMR 4 determines if the $\overline{CTS}$ line controls the operation of the transmitter.

RMR 7:6 = 00 is the normal mode, with transmitter and receiver operating independently.
RMR 7:6 = 01 places the channel in auto echo mode, which automatically retransmits the received data.
RMR 7:6 = 10 selects local loopback mode.
RMR 7:6 = 11 selects the remote loopback mode.

All bits of the mode register RMR 7:0 are cleared by a $\overline{RESET}$ signal or a RESET instruction issued by the CPU.

DEVELOPMENT DATA

Clock select register

	bit 07	bit 06	bit 05	bit 04	bit 03	bit 02	bit 01	bit 00
	Receiver Clock Select				**Not Used**	**Transmitter Clock Select**		
CLS	**Clock Source**			**Divisor**				**Divisor**
	0 = Internal (default after RESET) 1 = External	000 = 001 = 010 = 011 = 100 = 101 = 110 = 111 =	75 Baud 150 300 1200 2400 4800 9600 19200	65536 32768 16384 4096 2048 1024 512 256		000 = 001 = 010 = 011 = 100 = 101 = 110 = 111 =	75 Baud 150 300 1200 2400 4800 9600 19200	65536 32768 16384 4096 2048 1024 512 256

Fig. 37 Clock Select Register (CLS).

The CLS register allows selection of the clock source and the baud rate for receiver and transmitter; as shown above. The baud rates given above are generated when either a 19,66 MHz clock is used as the 68070's system clock (source = internal) or a 4,9152 MHz clock is applied to XCKI (source = external). Other frequencies will give a different set of baud rates. Note that when using the 68070's internal clock, it is pre-divided by 4.

CLS 7 selects the clock source for the receiver and transmitter baud rates. After RESET, the clock source is the on-chip clock and with a 19,66 MHz crystal, the listed baud rates are available. An external clock source (XCKI) is selected if CLS 7 = 1. The maximum frequency that can be applied to XCKI is 5 MHz.

All bits of the clock select register are cleared by a $\overline{\text{RESET}}$ signal, or by a RESET instruction issued by the CPU.

Command register

bit 07	bit 06	bit 05	bit 04	bit 03	bit 02	bit 01	bit 00

Command field	TxD control	RxD control
See text for detail	01 = enable 10 = disable 00 = illegal 11 = illegal	01 = enable 10 = disable 00 = illegal 11 = illegal

Fig. 38 RS232 Command Register (RCR).

RCR is used to write commands to the UART.

RCR 7:4 = Miscellaneous commands

The encoded value of this field may be used to specify a single command as follows:

0000 No command
0001 No command
0010 Reset receiver — resets the receiver as if a hardware reset had been applied. The receiver is disabled and the FIFO flushed.
0011 Reset transmitter — resets the transmitter as if a hardware reset had been applied.
0100 Reset error status — clears the received break, parity error, framing error, and overrun error bits in the status register RSR 7:4.
0101 No command
0110 Start break — forces the TxD output LOW (spacing).
0111 Stop break — The TxD line will go HIGH (marking) whithin two bit times. TxD will remain HIGH for one bit time before the next character, if any, is transmitted.
1XXX No command

All bits of the command register RCR 7:0 are cleared by a $\overline{\text{RESET}}$ signal, or by a RESET instruction issued by the CPU.

Status Register

bit 07	bit 06	bit 05	bit 04	bit 03	bit 02	bit 01	bit 00
Break	**Frame**	**Parity**	**Overrun**	**TxDEMT**	**TxDRDY**	**Not used**	**RxDRDY**
0 = normal 1 = break cond. received	0 = normal 1 = frame error	0 = normal 1 = parity error	0 = normal 1 = overrun error	0 = normal 1 = TxD shift register empty	0 = TxD register busy 1 = TxD register empty	(0)	0 = RxD register empty 1 = data in RxD register

TxDEMT = Transmitter Empty
TxDRDY = Transmitter Ready
RxDRDY = Receiver Ready

Fig. 39 RS232 Status Register (RSR).

Data defining the receiver and transmitter conditions and the modem/data set status are contained in the status register.

All status bits RSR 7:0 are cleared by a $\overline{\text{RESET}}$ signal, or a RESET instruction issued by the CPU.

DEVELOPMENT DATA

TIMER

The 68070's Timer comprises a 16-bit reference timer with an auto-reload register, and two identical (independently function-programmable) 16-bit registers. Clock period of the reference timer, and hence the maximum resolution, is 9,6 μs. Two programmable I/O lines provide the necessary connection to external circuitry.

Three modes can be selected:

- match or pulse-generator mode which changes the output state when there is a match between the reference and register values.
- count mode which counts external events that occur at the T1 (T2) input.
- capture mode which stores the reference timer value in a capture register when an external event occurs at the T1 (T2) input.

Any transition on the inputs to T1 or T2 can be programmed as an external event.

TIMER PROGRAMMING

The address map of the Timer is shown below:

Base address = 80002020 (HEX)

A3	A2	A1	A0	REGISTER	
0	0	0	0	Timer Status Register	TSR
0	0	0	1	Timer Control Register	TCR
0	0	1	0	Reload Register HIGH	RRH
0	0	1	1	Reload Register LOW	RRL
0	1	0	0	Timer 0 HIGH	T0H
0	1	0	1	Timer 0 LOW	T0L
0	1	1	0	Timer 1 HIGH	T1H
0	1	1	1	Timer 1 LOW	T1L
1	0	0	0	Timer 2 HIGH	T2H
1	0	0	1	Timer 2 LOW	T2L

The CPU can read from and write to all the Timer registers and they can be accessed "on the fly".

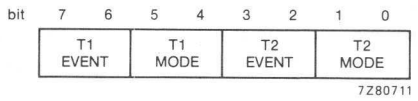


Fig. 40 Timer Control Register (TCR).

EVENT – Control for external events monitored to trigger a function in Timer registers T1 or T2.

00 – Input inhibited.
01 – LOW-to-HIGH transitions will be monitored.
10 – HIGH-to-LOW transitions will be monitored.
11 – Any transition will be monitored.

MODE – Control for the function of the Timer registers T1 or T2.

00 – Timer inhibited.
01 – Match Mode. A match between the reference timer T0 with the respective Timer register will reset output T1 or T2. Each overflow of T0 will set output T1 or T2. The I/O port of the Timer is automatically switched to output mode.
10 – Capture Mode. When an external event occurs (as described above), the contents of the reference timer T0 will be stored in the Timer register T1 or T2.
11 – Event Counter Mode. When an external event occurs the Timer register T1 or T2 will be incremented.

TIMER STATUS REGISTER

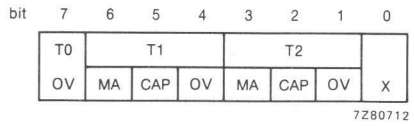


Fig. 41 Timer Status Register (TSR).

The Timer Status Register (TSR) indicates which timer and what specific event occured that caused an interrupt (if enabled) and can be read or written by the CPU. After being read, each bit of this register should be reset by software as an acknowledge of the read because the status bits are automatically set but are not reset by the Timer.

OV – Overflow. The Timer counts from FFFF (HEX) to 0000 (HEX). This bit will be set by timers T1 and T2 in event-counter mode only.
MA – Match. A match between the value stored in Timer registers T1 or T2 and the value of the continuous timer T0 occured (in match mode).
CAP – Capture. When an external event occurs the current value of the continuous timer (T0) is stored into timer T1 or T2 (in capture mode).
X – Undefined, reserved.

REFERENCE TIMER

The reference timer (T0) will increment by 1 (starting from the value initially loaded into T0H and T0L). Using a clock frequency of 19,66 MHz for the 68070's crystal, the 68070 will increment every 9,6 μs (or 192 clock cycles). When T0 reaches FFFF, the OV flag in the status register is set and the reload register (RR) is loaded into T0. T0 will then start incrementing again until the next overflow occurs.

ELECTRICAL SPECIFICATION

ABSOLUTE MAXIMUM RATINGS[1]

parameter	range	unit
Supply voltage	−0,3 to +7,0	V
Input voltage[3]	−0,3 to +7,0	V
Operating temperature[2]	0 to +70	°C
Storage temperature	−55 to +150	°C

Notes

1. Stresses above those listed in the Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only and do not mean that the device will operate at these or other conditions above those given in the operation section.
2. For operating at elevated temperatures, the device must be derated based on a 150 °C maximum junction temperature.
3. This product includes circuitry specifically designed to protect its internal devices from excessive static charge. Nevertheless, it is recommended that conventional precautions be taken to avoid applying any voltage above the rated maxima.
4. Parameters are valid over specified temperature range.
5. All voltages are measured with ground as reference (GND). For testing, all input signals swing between 0,4 and 2,4 V with a transition time of t.b.f. maximum. All time measurements are made with input voltages of 0,8 and 2,0 V and output voltages of 0,8 and 2,0 V as appriopriate.
6. On clock input XTAL1 when an external clock is used.
7. All timing measurements have CKOUT as a reference for both internal oscillator and external clock input modes. The device has been designed to be used with a 19,6608 MHz crystal, but the minimum crystal frequency specified is 8 MHz. All timing measurements except number 1 are specified at 19,6608 MHz.
8. Actual value depends on clock period.
9. If #47 is satisfied for both DTACK and BERR, #48 can be 0 ns.
10. After V_{DD} has been applied for 100 ms.
11. If the asynchronous setup time (#47) requirements are met, the DTACK LOW-to-data setup time (#31) requirements can be ignored. The data must only satisfy the data-in to clock-LOW setup time (#27) for the following cycle.

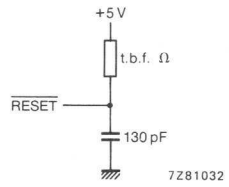


Fig. 42 Reset timing.

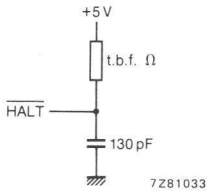


Fig. 43 Halt test load.

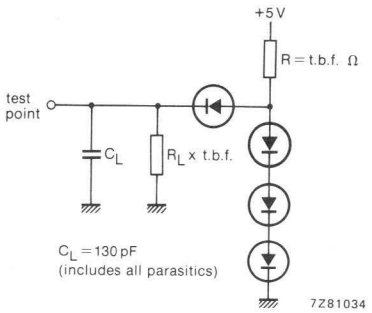


Fig. 44 Test loads.

All timing diagrams should only be referred to 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 the functional description and related diagrams for device operation.

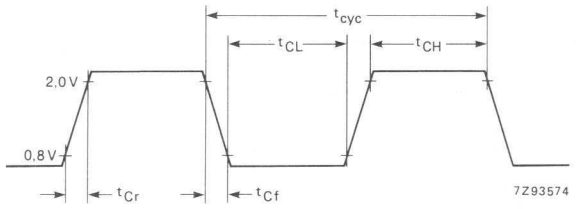


Fig. 45 Input clock timing (XTAL1) if external clock is used.

DC CHARACTERISTICS

V_{DD} = 5,0 V ± 5%, V_{SS} = 0 V; T_{amb} = 0 to 70 °C (see Figures 42 to 44)[4,5]

parameter		test conditions	limits		unit
			min.	max.	
V_{IH}	Input voltage HIGH all inputs except XTAL1 and XTAL2 XTAL1		 2,0 2,5	 V_{DD} V_{DD}	 V V
V_{IL}	Input voltage LOW		V_{SS}−0,3	0,8	V
I_{in}	Input leakage current $\overline{RTS}$, $\overline{CTS}$, XCKI, $\overline{BR}$, $\overline{DTACK}$, $\overline{INT1}$, $\overline{INT2}$, $\overline{REQ1}$, $\overline{REQ2}$, $\overline{RDY}$, $\overline{IN2}$, $\overline{IN4}$, $\overline{IN5}$, $\overline{NMI}$, and $\overline{AV}$ XTAL1	5,25 V		t.b.f. t.b.f.	µA µA
I_{TSI}	Three-state (off-state) input current A1-A23, D0-D15, $\overline{AS}$, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, T1, and T2	2,4/0,4 V		t.b.f.	µA
I_{ODI}	Open-drain (off-state) input current $\overline{BGACK}$, $\overline{RESET}$, $\overline{HALT}$, $\overline{BERR}$, $\overline{DTC}$, $\overline{DONE}$, SCL, and SDA			t.b.f.	µA
V_{OH}	Output voltage HIGH CKOUT A1-A23, D0-D15, $\overline{AS}$, $\overline{BG}$, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, T1, T2, TXD, $\overline{RTS}$, $\overline{ACK1}$, $\overline{ACK2}$, and $\overline{IACK2,4,5,7}$	I_{OH} = t.b.f. 	 0,8V_{DD} 2,4	 	 V V
V_{OL}	Output voltage LOW $\overline{HALT}$, $\overline{BERR}$, $\overline{IACK2,4,5,7}$, A1-A23, $\overline{BG}$, $\overline{BGACK}$, $\overline{ACK1}$, $\overline{ACK2}$, $\overline{RESET}$, SDA, SCL, T1, T2, TXD, $\overline{RTS}$, $\overline{AS}$, D0-D15, $\overline{LDS}$, R/$\overline{W}$, $\overline{UDS}$, $\overline{DTC}$, $\overline{DONE}$ CKOUT	 I_{OL} = 3,2 mA I_{OL} = 3,2 mA	 	 0,5 0,45	 V V
P_D	Power dissipation	Clock frequency = 20 MHz		t.b.f.	mW
C_{in}	Capacitance	V_{in} = 0 V, T_{amb} = 25 °C frequency = 1 MHz		t.b.f.	pF

AC CHARACTERISTICS

CRYSTAL FREQUENCY = 19,6608 MHz, V_{DD} = 5 V, T_{amb} = 0 to 70 °C, C_{load} on CKOUT = 50 pF (see Fig. 46 to 49).

number	parameter	symbol	tentative limits		unit
			min.	max.	
1	Crystal or input (XTAL1) clock period	t_{CYC}	50	125	ns
1A	XTAL HIGH to CKOUT HIGH or LOW		t.b.f.	t.b.f.	ns
2	Clock out, LOW level	t_{COL}	25		ns
3	Clock out, HIGH level	t_{COH}	25		ns
4	Clock out fall-time	t_{COf}			ns
5	Clock out rise-time	t_{COr}			ns
6	Clock LOW to address	t_{CLAV}		50	ns
7	Clock HIGH to address data, high-impedance (maximum)	t_{CHAZx}	15		ns
8	Clock HIGH to address /FC valid (minimum)	t_{CHAZn}	0		ns
9	Clock HIGH to AS, DS LOW (maximum)	t_{CHSLx}		45	ns
10	Clock HIGH to AS, DS LOW (minimum)	t_{CHSLn}	0		ns
11[8]	Address to AS, DS (read) LOW/AS write	t_{AVSL}	t.b.f.		ns
12	Clock LOW to AS, DS HIGH	t_{CLSH}		45	ns
13[8]	AS, DS HIGH to address invalid	t_{SHAZ}	t.b.f.		ns
14[8]	AS, DS LOW level (read)/AS (write)	t_{SL}	200		ns
14A[8]	DS LOW level (write)		100		ns
15	AS, DS HIGH level	t_{SH}	100		ns
16	Clock HIGH to AS, DS high-impedance	t_{CHSZ}		t.b.f.	ns
17[8]	AS, DS HIGH to R/W HIGH	t_{SHRH}	t.b.f.		ns
18	Clock HIGH to R/W HIGH (maximum)	t_{CHRHx}		t.b.f.	ns
19	Clock HIGH to R/W HIGH (minimum)	t_{CHRHn}	0		ns
20	Clock HIGH to R/W LOW	t_{CHRL}		t.b.f.	ns
21[8]	Address valid to R/W LOW	t_{AVRL}	0		ns
22[8]	R/W LOW to DS LOW (write)	t_{RLSL}		t.b.f.	ns
23	Clock LOW to data-out valid	t_{CLDO}		50	ns
25[8]	DS HIGH to data-out invalid	t_{SHDO}	t.b.f.		ns
26[8]	Data-out valid to DS LOW (write)	t_{DOSL}	t.b.f.		ns
27[11]	Data-in to clock LOW (setup time)	t_{DICL}	40		ns
28[8]	AS, DS HIGH to DTACK, RDY HIGH	t_{SHDAH}	0	150	ns
29	DS HIGH to data invalid (hold time)	t_{SHDI}	0		ns
30	AS, DS HIGH to BERR HIGH	t_{SHBEH}	0		ns
31[8]	DTACK LOW to data in (setup time)	t_{DALDI}		t.b.f.	ns
32	HALT and RESET input transition time	t_{RHrf}	0	t.b.f.	ns
33	Clock HIGH to BG LOW	t_{CHGL}		45	ns
34	Clock HIGH to BG HIGH	t_{CHGH}		55	ns

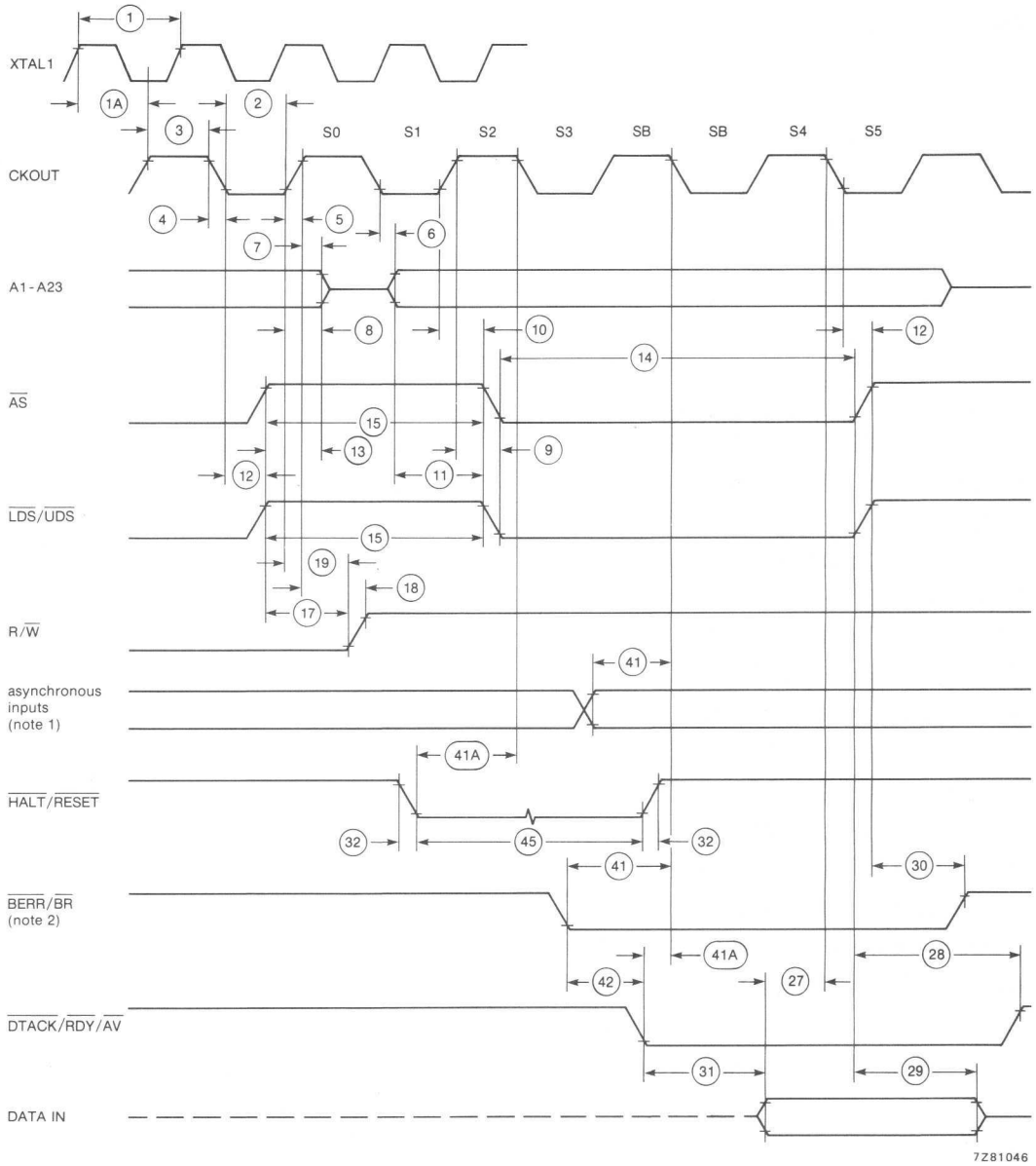
See Notes on page 52

AC CHARACTERISTICS (cont.)

CRYSTAL FREQUENCY = 19,6608 MHz, V_{DD} = 5 V, T_{amb} = 0 to 70 °C, C_{load} on CKOUT = 50 pF

number	parameter	symbol	tentative limits		unit
			min.	max.	
35	BR LOW to BG LOW	t_{BRLGL}	1,5	3,0	c.p.
35	BR LOW to BG LOW (Fig. 49)	t_{BRLGL}	1,5	3,5	c.p.
36	BR HIGH to BG HIGH	t_{BRHGH}	1,5	3,0	c.p.
37	BGACK LOW to BG HIGH	t_{GALGH}	1,5	3,0	c.p.
38	BG LOW to bus high-impedance (AS HIGH)	t_{GLZ}	15	ns	
39	BG HIGH level	t_{GH}	1,5		c.p.
40	BGACK width	t_{BGL}	1,5		c.p.
41[11]	Asynchronous input setup time	t_{ASI}	35		ns
41A	Asynchronous input setup time for DTACK, AV, BERR, HALT, RDY	t_{ASDT}	35		ns
42	BERR LOW to DTACK LOW	t_{BELDAL}	0		ns
43	Data hold from clock HIGH	t_{CHDO}	0		ns
44	R/W to data bus impedance change	t_{RLDO}	t.b.f.		ns
45	HALT/RESET pulse width	t_{HRPW}	10		c.p.
46	REQ setup before CKOUT LOW		35		ns
47	ACKx LOW from CKOUT HIGH		0	45	ns
48	REQx hold after CKOUT LOW		10		ns
49	DTC LOW from CKOUT HIGH			45	ns
50	AS, LDS, UDS HIGH from DTC LOW		t.b.f.		
51	ACK HIGH from CKOUT HIGH			55	ns
52	DTC non-active from CKOUT HIGH			45	ns
53	DONE (output) LOW from CKOUT HIGH			45	ns
54	DONE (output) non-active from CKOUT HIGH			45	ns
56	DONE (input) setup LOW before CKOUT LOW		35		ns
57	DONE (input) hold LOW after CKOUT HIGH		10		ns
58	REQ LOW to BGACK (output) LOW		2,5		c.p.

See Notes on page 52



Notes

1. Setup time for the asynchronous inputs and $\overline{AV}$ guarantees their recognition at the next falling edge of the clock.
2. $\overline{BR}$ need fall at this time only to ensure being recognized at the end of this bus cycle.

Fig. 46 Read cycle timing.

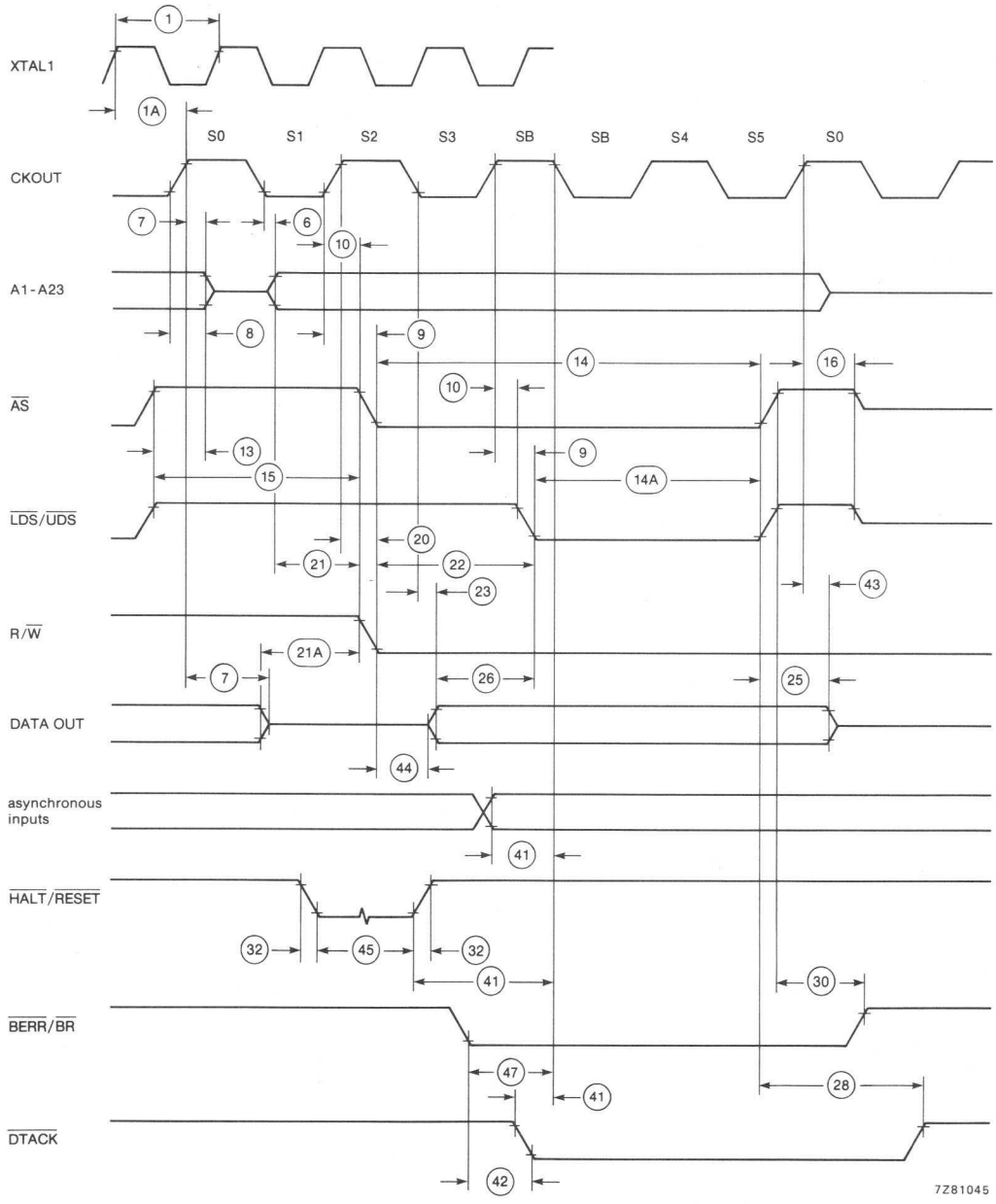


Fig. 47 Write cycle timing.

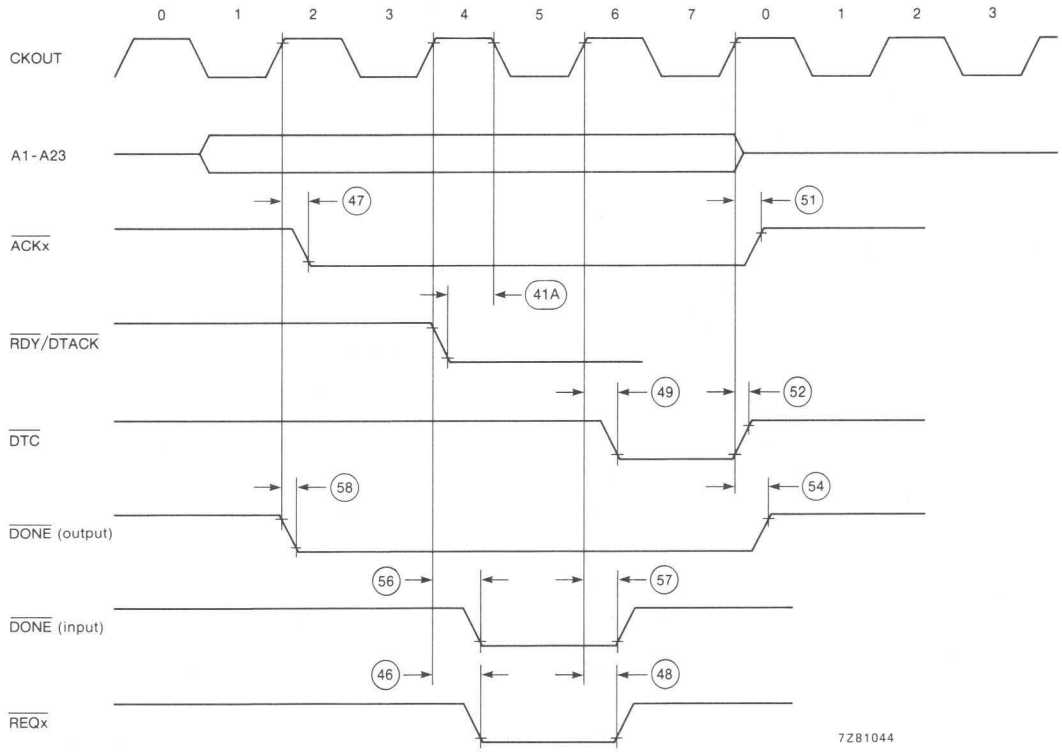
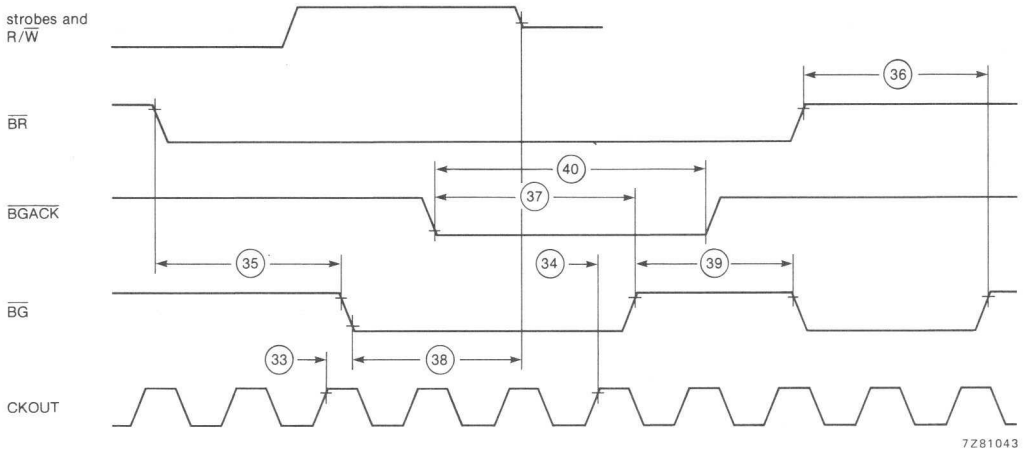


Fig. 48 DMA timing.



Notes

1. Setup time for the asynchronous inputs $\overline{BERR}$, $\overline{BR}$, $\overline{BGACK}$, $\overline{DTACK}$, $\overline{IN2}$, $\overline{IN4}$, $\overline{IN5}$, $\overline{NMI}$, $\overline{INT1}$, $\overline{INT2}$, $\overline{AV}$, $\overline{REQ1}$, $\overline{REQ2}$, $\overline{DONE}$, $\overline{RDY}$ and DATA guarantees their recognition.

Fig. 49 AC electrical timing – bus arbitration.

CLOCK TIMING (see Fig. 45)

parameter	symbol	min.	max.	unit
Frequency of operation	F	8	20	MHz
Cycle time	t_{cyc}	50	125	ns
Clock pulse width	t_{CL}	15	80	ns
	t_{CH}	15	80	ns
Rise and fall times	t_{Cr}	–	t.b.f.	ns
	t_{Cf}	–	t.b.f.	ns

POWER CONSIDERATIONS

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

$$T_j = T_{amb} + (P_D \times R_{THJA}) \quad (1)$$

where:

T_{amb} = ambient temperature, °C
R_{THJA} = package thermal resistance, junction-to-ambient, °C/W

$$P_D = P_{INT} + P_{I/O} \quad (2)$$

P_{INT} = $I_{CC} \times V_{CC}$ W = chip internal power
$P_{I/O}$ = power dissipation on input and output pins (determined by the user)

For most applications $P_{I/O} \ll P_{INT}$ and can be neglected. An approximate relationship between P_D and T_j (if $P_{I/O}$ is neglected) is:

$$P_D = K + (T_j + 273)$$

Solving equations (1) and (2) for K gives:

$$K = P_O(T_{amb} + 273) + R_{THJA}P_D^2 \quad (3)$$

Where K is a constant pertaining to a particulat part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_{amb}. 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_{amb}.

I^2C INTERFACE TIMING

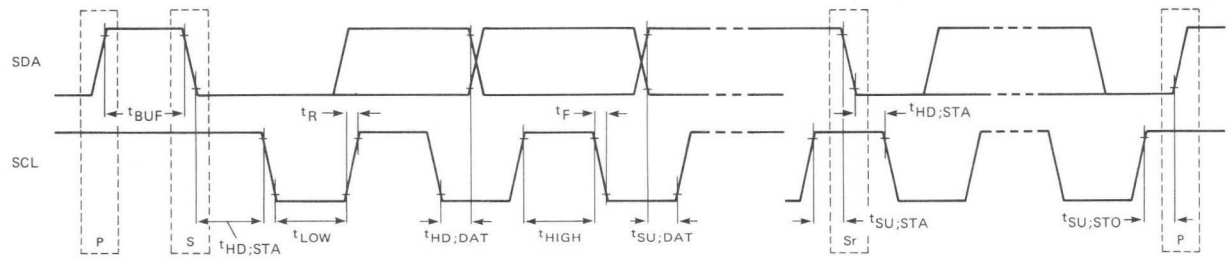


Fig. 50 I^2C interface timing.

parameter	symbol	min.	max.	units
SCL clock frequency	f_{SCL}	0	128*	kHz
Time the bus must be free before a new transmission can start	t_{BUF}	t.b.f.	–	μs
Hold time START condition. After this period the first clock pulse is generated	$t_{HD;STA}$	t.b.f.	–	μs
The LOW period of the clock	t_{LOW}	3,0	–	μs
The HIGH period of the clock	t_{HIGH}	3,0	–	μs
Set-up time for START condition (only relevant for a repeated start condition).	$t_{SU;STA}$	t.b.f.	–	μs
Hold time DATA for I^2C devices	$t_{HD;DAT}$	0	–	μs
Set-up time DATA	$t_{SU;DAT}$	t.b.f.	–	ns
Rise time of both SDA and SCL lines	t_R	–	t.b.f.	μs
Fall time of both SDA and SCL lines	t_F	–	t.b.f.	ns
Set-up time for STOP condition	$t_{SU;DAT}$	t.b.f.	–	ns
All values referred to V_{IH} and V_{IL} levels.				

* Note: Maximum bus clock frequency in I^2C systems is 100 kHz

RS232-C INTERFACE TIMING

parameter	symbol	min.	typ.	max.	unit
Transmitter timing (Fig. 52)					
TXD output delay from XCKI LOW	t_{XTXD}			t.b.f.	ns
TXD output delay from CKOUT LOW	t_{CTXD}			t.b.f.	ns
Receiver timing (Fig. 53)					
RXD data setup time to XCKI HIGH	t_{XRXS}	t.b.f.			ns
RXD data hold time from XCKI HIGH	t_{XRXH}	t.b.f.			ns
RXD data setup time to CKOUT HIGH	t_{CRXS}	t.b.f.			ns
RXD data hold time from CKOUT HIGH	t_{CRXH}	t.b.f.			ns
XCKI frequency of operation	F_{XCKI}	t.b.f.	4,9152	5	MHz
XCKI cycle time	t_{XI}	200		t.b.f.	ns
XCKI pulse width	t_{XH}	60		140	ns
	t_{XL}	60		140	ns
XCKI rise and fall times	t_{Xr}			t.b.f.	ns
	t_{Xf}			t.b.f.	ns

* This frequency gives accurate baud rate generation.

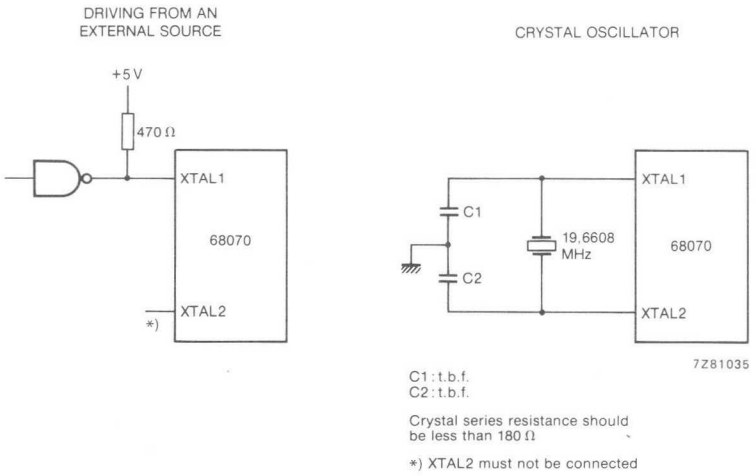


Fig. 51 Clock circuitry.

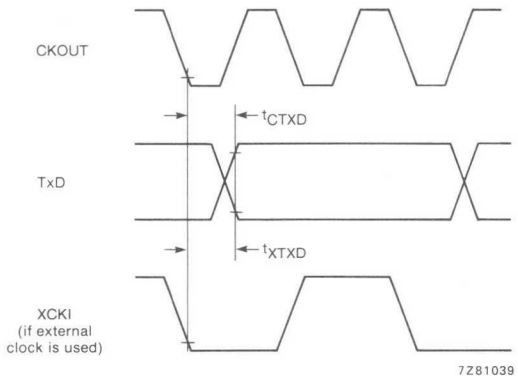


Fig. 52 Transmit timing.

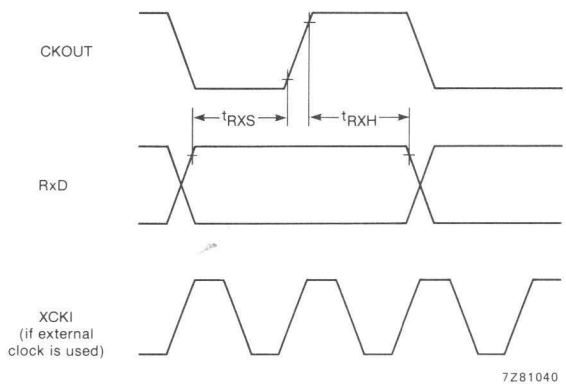


Fig. 53 Receive timing.

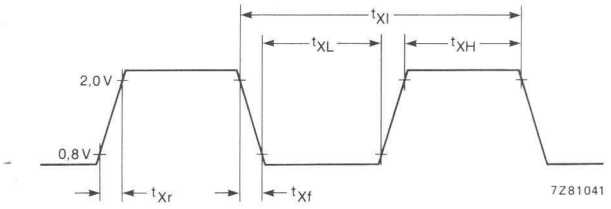


Fig. 54 XCKI input timing.

TIMER SPECIFICATION

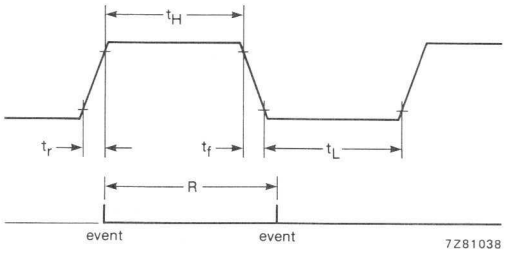


Fig. 55 Input timing to T1 or T2.

parameter	symbol	min.	max.	unit
T1 or T2 pulse width	t_H	700		ns
	t_L	700		ns
T1 or T2 rise and fall times	t_r		t.b.f.	ns
	t_f		t.b.f.	ns
Resolution (the time between two events to be taken into account)	R	10		μs

* T1 and T2 input signals must be held HIGH or LOW longer than t_H or t_L to be latched at the input to the Timer. Events must be separated by more than 10 μs, the resolution of the Timer.

DEVELOPMENT DATA

SUMMARY OF ON-CHIP ADDRESSES

HEX address			symbol	register
8000 0000	to	8000 0FFF	–	reserved
8000 1000			LIR	Latched Interrupt priority Register
8000 1001	to	8000 2000	–	reserved
8000 2001			IDR	I^2C Data Register
8000 2002			–	reserved
8000 2003			IAR	I^2C Address Register
8000 2004			–	reserved
8000 2005			ISR	I^2C Status Register
8000 2006			–	reserved
8000 2007			ICR	I^2C Control Register
8000 2008			–	reserved
8000 2009			ICC	I^2C Clock Control Register
8000 200A	to	8000 2010	–	reserved
8000 2011			RHR	Receive Holding Register RS232-C
8000 2012			–	reserved
8000 2013			THR	Transmit Holding Register RS232-C
8000 2014			–	reserved
8000 2015			RSR	RS232-C Status Register
8000 2016	to	8000 2018	–	reserved
8000 2019			RMR	RS232-C Mode Register
8000 201A	to	8000 201C	–	reserved
8000 201D			RCR	RS232-C Command Register
8000 201E			–	reserved
8000 201F			CLS	Clock Select Register RS232-C
8000 2020			TSR	Timer Status Register
8000 2021			TCR	Timer Control Register
8000 2022			RRH	
8000 2023			RRL	
8000 2024			T0H	Timer 0 High
8000 2025			T0L	Timer 0 Low
8000 2026			T1H	Timer 1 High
8000 2027			T1L	Timer 1 Low
8000 2028			T2H	Timer 2 High
8000 2029			T2L	Timer 2 Low
8000 202A	to	8000 2044	–	reserved
8000 2045			PICR1	Peripheral Interrupt Control Register 1
8000 2046			–	reserved
8000 2047			PICR2	Peripheral Interrupt Control Register 2
8000 2048	to	8000 3FFF	–	reserved
8000 4000			CSR	Channel Status Register Channel 1
8000 4001			CER	Channel Error Register Channel 1
8000 4002	to	8000 4003	–	reserved
8000 4004			DCR	Device Control Register Channel 1
8000 4005			OCR	Operation Control Register Channel 1

SUMMARY OF ON-CHIP ADDRESSES (cont.)

HEX address			symbol	register
8000 4006			SCR	Sequence Control Register Channel 1
8000 4007			CCR	Channel Control Register Channel 1
8000 4008	to	8000 4009	–	reserved
8000 400A			MTCH	Memory Transfer Counter High Channel 1
8000 400B			MTCL	Memory Transfer Counter Low Channel 1
8000 400C			MACH	Memory Address Counter High Channel 1
8000 400D			MACMH	Memory Address Counter Middle High Channel 1
8000 400E			MACML	Memory Address Counter Middle Low Channel 1
8000 400F			MACL	Memory Address Counter Low Channel 1
8000 4010	to	8000 402C	–	reserved
8000 402D			CPR	Channel Priority Register Channel 1
8000 402E	to	8000 403F	–	reserved
8000 4040			CSR	Channel Status Register Channel 2
8000 4041			CER	Channel Error Register Channel 2
8000 4042	to	8000 4043	–	reserved
8000 4044			DCR	Device Control Register Channel 2
8000 4045			OCR	Operation Control Register Channel 2
8000 4046			SCR	Sequence Control Register Channel 2
8000 4047			CCR	Channel Control Register Channel 2
8000 4048	to	8000 4049	–	reserved
8000 404A			MTCH	Memory Transfer Counter High Channel 2
8000 404B			MTCL	Memory Transfer Counter Low Channel 2
8000 404C			MACH	Memory Address Counter High Channel 2
8000 404D			MACMH	Memory Address Counter Middle High Channel 2
8000 404E			MACML	Memory Address Counter Middle Low Channel 2
8000 404F			MACL	Memory Address Counter Low Channel 2
8000 4050	to	8000 4053	–	reserved
8000 4054			DACH	Device Address Counter High
8000 4055			DACMH	Device Address Counter Middle High
8000 4056			DACML	Device Address Counter Middle Low
8000 4057			DACL	Device Address Counter Low
8000 4058	to	8000 406C	–	reserved
8000 406D			CPR	Channel Priority Register Channel 2
8000 406E	to	8000 7FFF	–	reserved
8000 8000			MSR	MMU Status Register
8000 8001			MCR	MMU Control Register
8000 8002	to	8000 803F	–	reserved
8000 8040			SAH	Segment Attributes High, Descriptor 0
8000 8041			SAL	Segment Attributes Low, Descriptor 0
8000 8042			SLH	Segment Length High, Descriptor 0
8000 8043			SLL	Segment Length Low, Descriptor 0
8000 8044			–	reserved
8000 8045			SNR	Segment Number, Descriptor 0
8000 8046			SBH	Segment Base Address High, Descriptor 0
8000 8047			SBL	Segment Base Address Low, Descriptor 0

SUMMARY OF ON-CHIP ADDRESSES (cont.)

HEX address	symbol	register
8000 8048	SAH	Segment Attributes High, Descriptor 1
8000 8049	SAL	Segment Attributes Low, Descriptor 1
8000 804A	SLH	Segment Length High, Descriptor 1
8000 804B	SLL	Segment Length Low, Descriptor 1
8000 804C	–	reserved
8000 804D	SNR	Segment Number, Descriptor 1
8000 804E	SBH	Segment Base Address High, Descriptor 1
8000 804F	SBL	Segment Base Address Low, Descriptor 1
8000 8050	SAH	Segment Attributes High, Descriptor 2
8000 8051	SAL	Segment Attributes Low, Descriptor 2
8000 8052	SLH	Segment Length High, Descriptor 2
8000 8053	SLL	Segment Length Low, Descriptor 2
8000 8054	–	reserved
8000 8055	SNR	Segment Number, Descriptor 2
8000 8056	SBH	Segment Base Address High, Descriptor 2
8000 8057	SBL	Segment Base Address Low, Descriptor 2
8000 8058	SAH	Segment Attributes High, Descriptor 3
8000 8059	SAL	Segment Attributes Low, Descriptor 3
8000 805A	SLH	Segment Length High, Descriptor 3
8000 805B	SLL	Segment Length Low, Descriptor 3
8000 805C	–	reserved
8000 805D	SNR	Segment Number, Descriptor 3
8000 805E	SBH	Segment Base Address High, Descriptor 3
8000 805F	SBL	Segment Base Address Low, Descriptor 3
8000 8060	SAH	Segment Attributes High, Descriptor 4
8000 8061	SAL	Segment Attributes Low, Descriptor 4
8000 8062	SLH	Segment Length High, Descriptor 4
8000 8063	SLL	Segment Length Low, Descriptor 4
8000 8064	–	reserved
8000 8065	SNR	Segment Number, Descriptor 4
8000 8066	SBH	Segment Base Address High, Descriptor 4
8000 8067	SBL	Segment Base Address Low, Descriptor 4
8000 8068	SAH	Segment Attributes High, Descriptor 5
8000 8069	SAL	Segment Attributes Low, Descriptor 5
8000 806A	SLH	Segment Length High, Descriptor 5
8000 806B	SLL	Segment Length Low, Descriptor 5
8000 806C	–	reserved
8000 806D	SNR	Segment Number, Descriptor 5
8000 806E	SBH	Segment Base Address High, Descriptor 5
8000 806F	SBL	Segment Base Address Low, Descriptor 5

SUMMARY OF ON-CHIP ADDRESSES (cont.)

parameter	symbol	register
8000 8070	SAH	Segment Attributes High, Descriptor 6
8000 8071	SAL	Segment Attributes Low, Descriptor 6
8000 8072	SLH	Segment Length High, Descriptor 6
8000 8073	SLL	Segment Length Low, Descriptor 6
8000 8074	–	reserved
8000 8075	SNR	Segment Number, Descriptor 6
8000 8076	SBH	Segment Base Address High, Descriptor 6
8000 8077	SBL	Segment Base Address Low, Descriptor 6
8000 8078	SAH	Segment Attributes High, Descriptor 7
8000 8079	SAL	Segment Attributes Low, Descriptor 7
8000 807A	SLH	Segment Length High, Descriptor 7
8000 807B	SLL	Segment Length Low, Descriptor 7
8000 807C	–	reserved
8000 807D	SNR	Segment Number, Descriptor 7
8000 807E	SBH	Segment Base Address High, Descriptor 7
8000 807F	SBL	Segment Base Address Low, Descriptor 7
8000 8080 to BFFF FFFF	–	reserved

DEVELOPMENT DATA

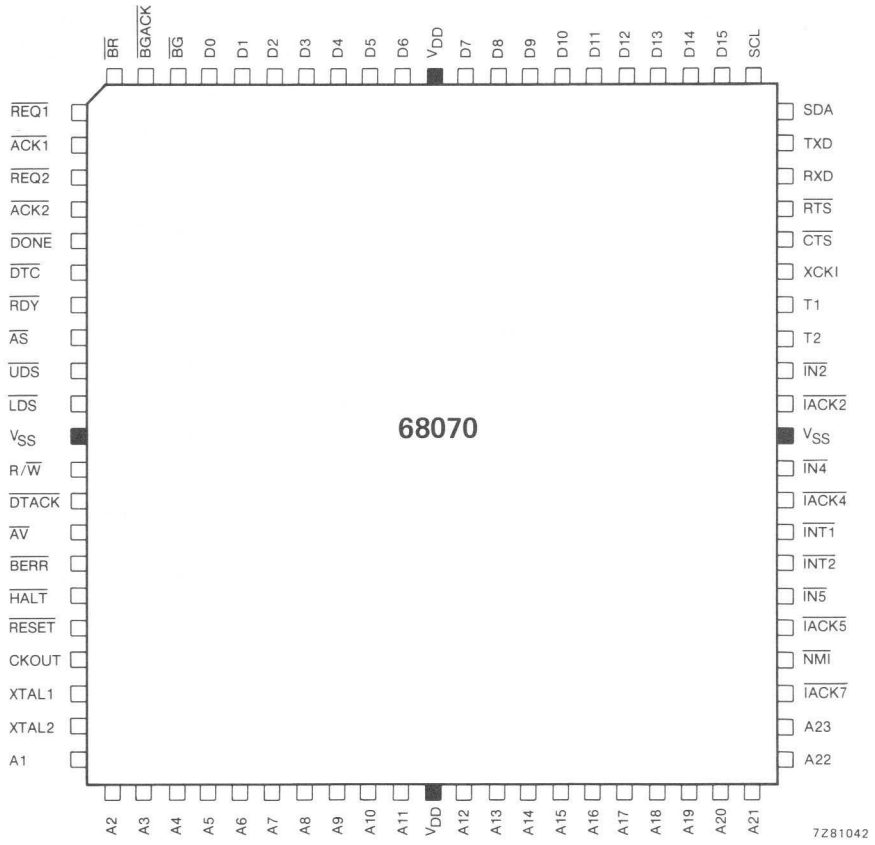
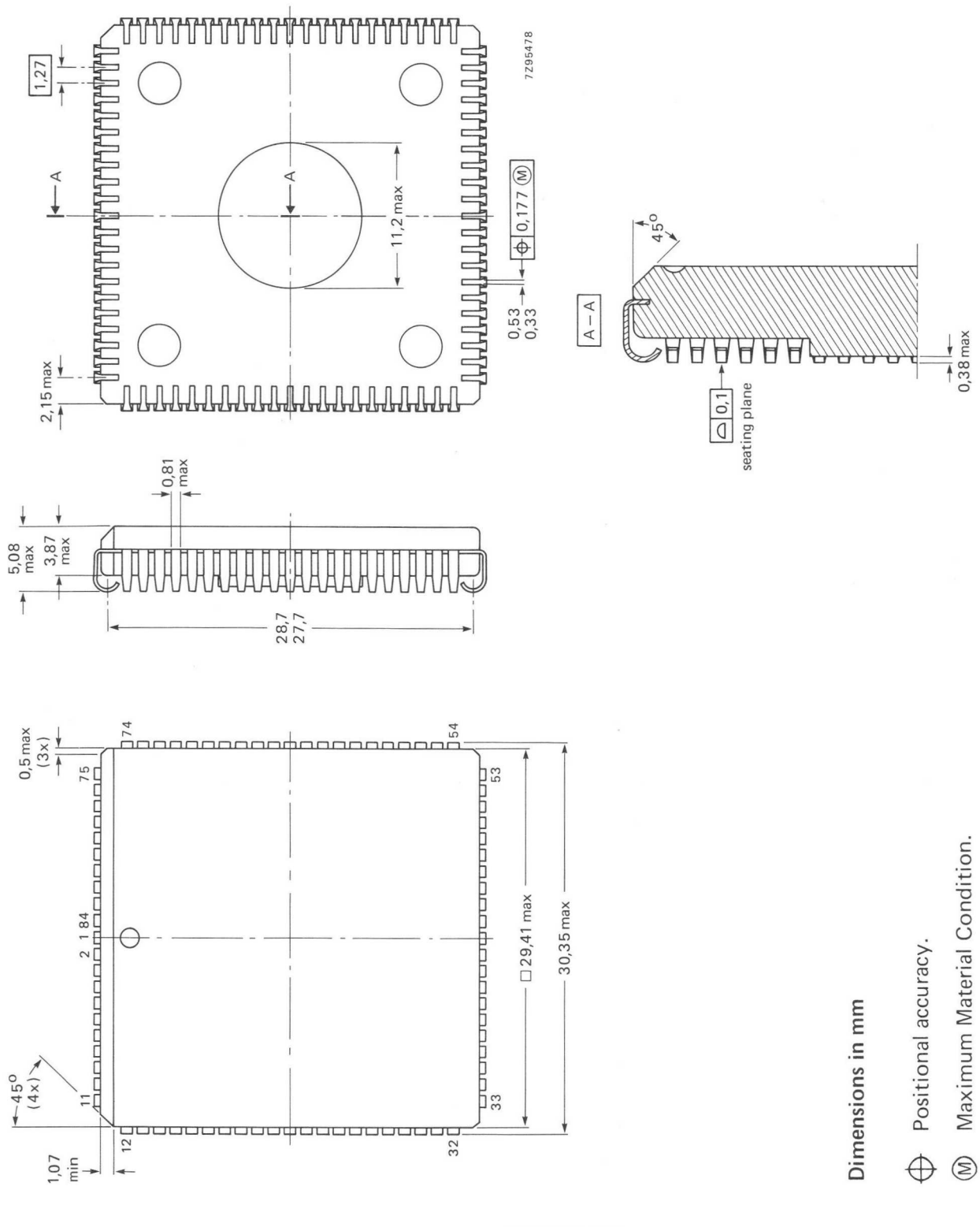


Fig. 56 Pinning.

DEVELOPMENT DATA

84-LEAD PLASTIC LEADED CHIP-CARRIER (PLCC); SOT-189A



Dimensions in mm

⊕ Positional accuracy.

Ⓜ Maximum Material Condition.

October '87

Anomaly sheet for SCC 68070 (V70003 V3S version)

Marking: ....
....

Relevant development datasheet. April 1987.

CPU: All instructions are working correctly, with the exception of the TAS instruction when recovering from a bus error: normally the RR bit in the Special Status Word on the system stack (long stack format during bus or address error) can be negated to prevent a faulty bus cycle from re-running. the TAS instruction error means that this bit is ignored and the faulty bus cycle will be re-run regardless of the setting of the RR bit.

MMU: o.k.

DMA: If channel 1 and 2 are working simultaneausly, then under special circumstances, the DONEN signal may not be generated, resulting in a continuatoin of transfers. This can occur if between the two last transfers of one channel there are one or more transfers on the other channel (in cycle steal as well as in burst mode).

Example:
CHx.D means transfer on channel x with DONEN generated.

```
It works if:      CH1 OR CH2.CH1.CH1.D.CH2.........
              OR  CH1 OR CH2.CH2.CH2.D.CH1.........
It fails if:      CH1.CH2.CH1.D.CH2.....  FAILED ON CHANNEL 1
              OR  CH1.CH1.CH2.D.CH1.....  FAILED ON CHANNEL 2
```

UART: o.k.

I²C: o.k.

TIMER:
If writing to any timer register on the cycle that the free-running reference timer (T0)increments, then the write operation is faulty.
To avoid this problem, read the value stored in the register, compare it to the desired value and, if wrong, rewrite the register with a period diferent to a multiple of 24 CKOUT cycles.

Reliability:
This version of the 68070 must be used for engeneering purposes only. The standard reliability figures for our products cannot be applied to this device.