MC68705P3

Technical Summary
8-Bit EPROM Microcomputer Unit

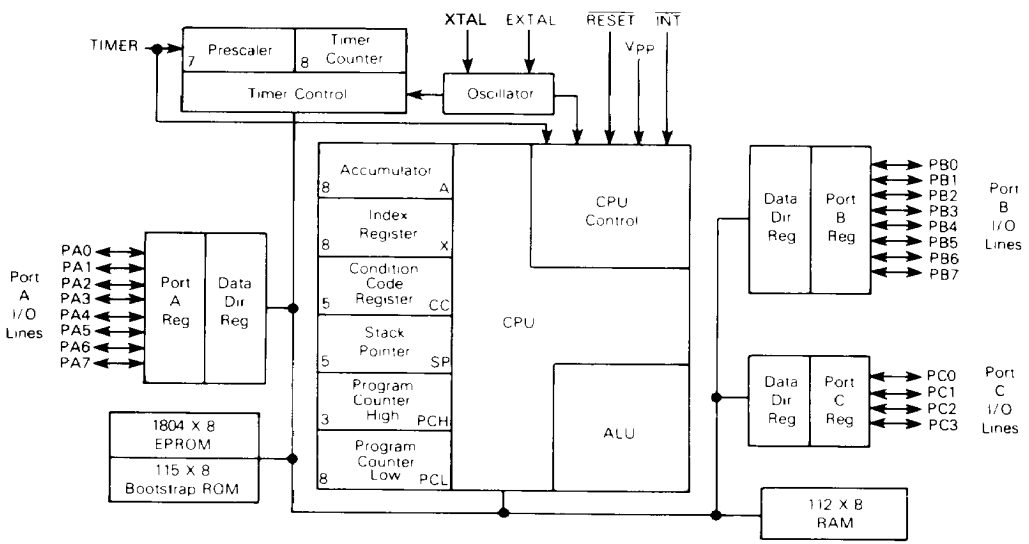
The MC68705P3 (High-Density NMOS) Microcomputer Unit (MCU) is an EPROM member of the MC6805 Family of microcomputers. The user programmable EPROM allows program changes and lower volume applications. This low cost MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Bootstrap program in ROM
- 1804 Bytes EPROM
- 112 Bytes RAM
- 20 TTL/CMOS Compatible Bidirectional I/O Lines

3

BLOCK DIAGRAM



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

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

V_{PP}

This pin is used when programming the EPROM. In normal operation, this pin is connected to V_{CC}.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made by the CLK bit in the mask option register.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges

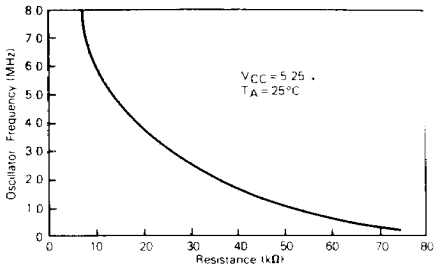


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option only

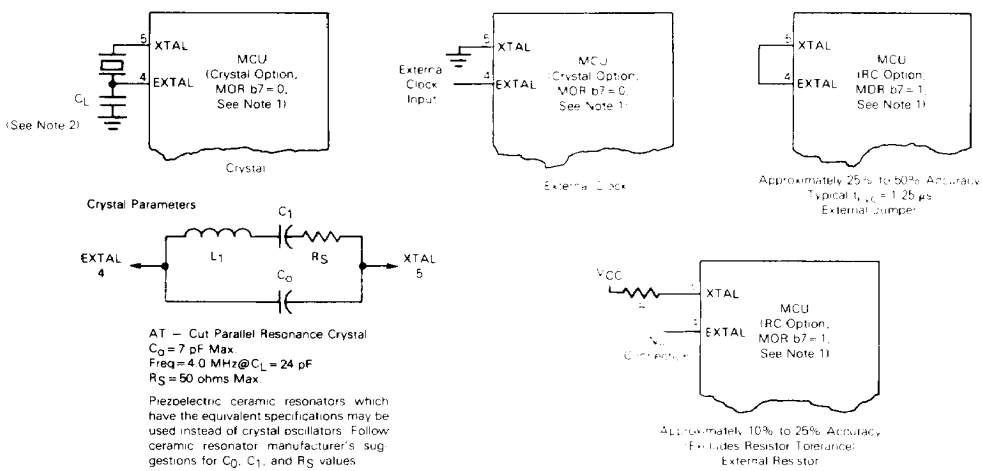
are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register.

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a



NOTES:

1. When the TIMER input pin is in the V_{IHTP} range (in the bootstrap EPROM programming mode), the crystal option is forced. When the TIMER input is at or below V_{CC}, the clock generator option is determined by bit 7 of the mask option register (CLK).
2. The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

higher voltage level used to initiate the bootstrap program.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low. Refer to **RESETS** section for more detail.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC3)

These 20 lines are arranged into two 8-bit ports (A and B) and one 4-bit port (C). All lines are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

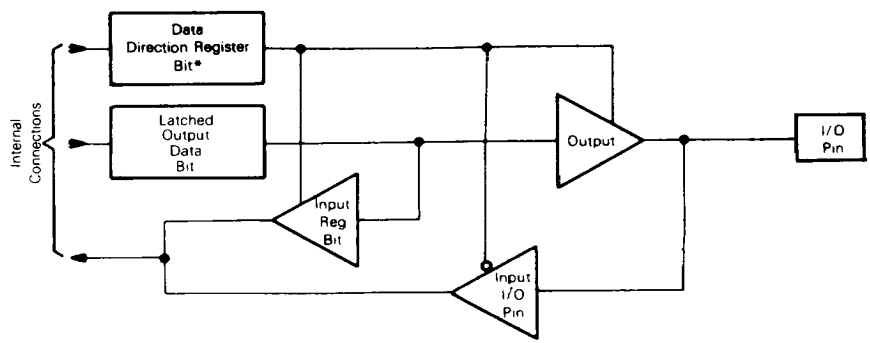
Any port pin is programmable as either input or output under software control of the corresponding write-only data direction register (DDR); DDRs always read "1". The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic 1 for output and a logic 0 for input. On reset, all the DDRs are initialized to a logic 0 state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (0) and also the latched output when the DDR is an output (1). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

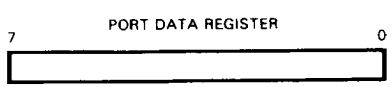
Table 1. I/O Pin Functions

Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

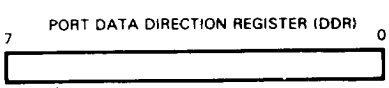
**Ports A (with CMOS drive disabled), B, and C are three state ports. Port A has optional internal pullup devices to provide CMOS drive capability. See Electrical Characteristic tables for complete information.



*DDR is a write-only register and reads as all "1s".



Port A Addr = $000
Port B Addr = $001
Port C Addr = $002 (Bits 0→3)



(1) Write Only; reads as all "1s"
(2) 1 = Output; 0 = Input. Cleared to 0 by reset.
(3) Port A Addr = $004
Port B Addr = $005
Port C Addr = $006 (Bits 0→3)

Figure 3. Typical Port I/O Circuitry and Register Configuration

MEMORY

The MCU is capable of addressing 2048 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user EPROM, bootstrap ROM, RAM, a mask option register (MOR), a program control register, and I/O. The interrupt vectors are located from $7F8 to $7FF. The bootstrap is a mask-programmed ROM that allows the MCU to program its own EPROM.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

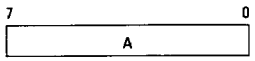
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

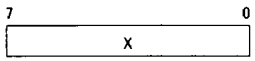
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



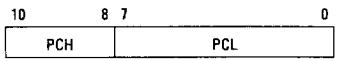
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



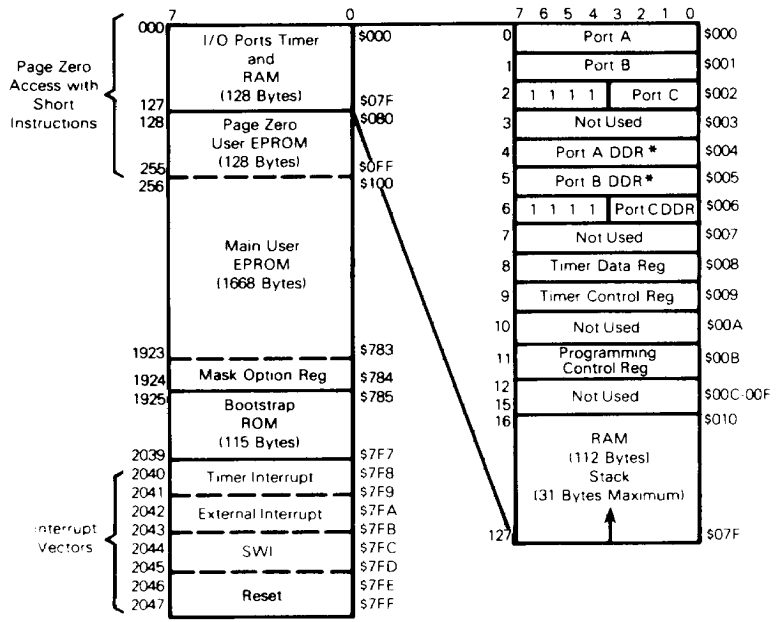
PROGRAM COUNTER (PC)

The program counter is an 11-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

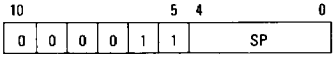
The stack pointer is an 11-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.



Caution Data Direction Registers (DDRs) are write-only, they read as $FF

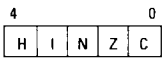
Figure 4. Memory Map

The six most-significant bits of the stack pointer are permanently set at 000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic 1).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the $\overline{\text{RESET}}$ line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 5) typically provides sufficient delay.

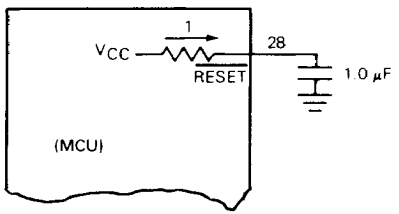


Figure 5. Power-up $\overline{\text{RESET}}$ Delay Circuit

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

INTERRUPTS

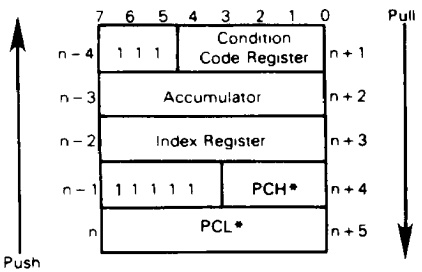
The MCU can be interrupted three different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, or (3) using the software interrupt instruction (SWI).

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and then normal processing resumes. The stacking order is shown in Figure 6.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.



*For subroutine calls, only PCH and PCL are stacked.

Figure 6. Interrupt Stacking Order

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack, and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$. Clearing the I bit enables the external interrupt. The following paragraphs describe two typical external interrupt circuits.

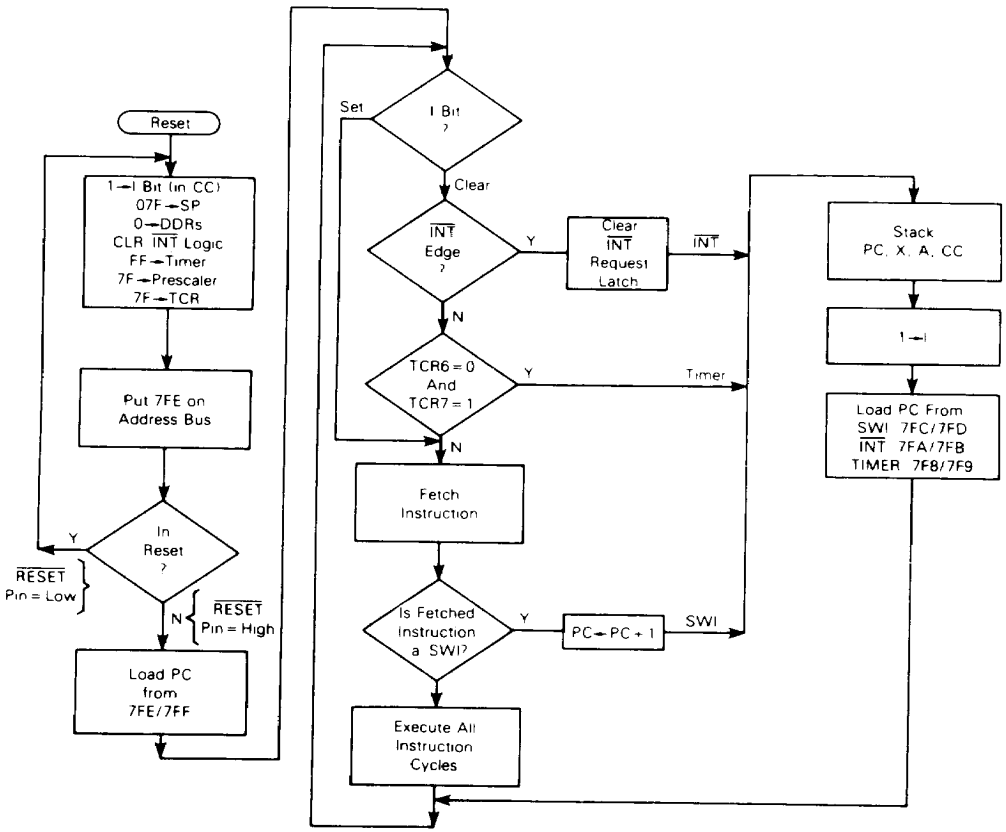


Figure 7. Reset and Interrupt Processing Flowchart

Zero-Crossing

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 8a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and thereby provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Fibure 8b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. Various timer sources are made via the timer control register (TCR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 9 for timer block diagram.

Timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared, and TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by 1) saving the present CPU state on the stack, 2) fetching the timer interrupt vector,

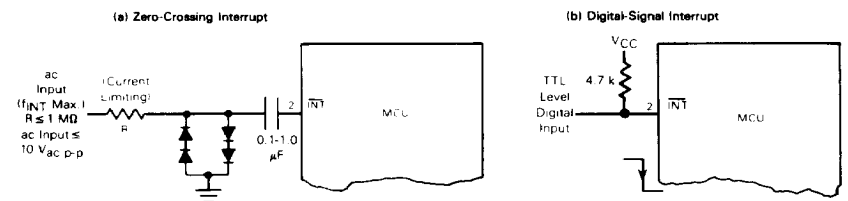


Figure 8. Typical Interrupt Circuits

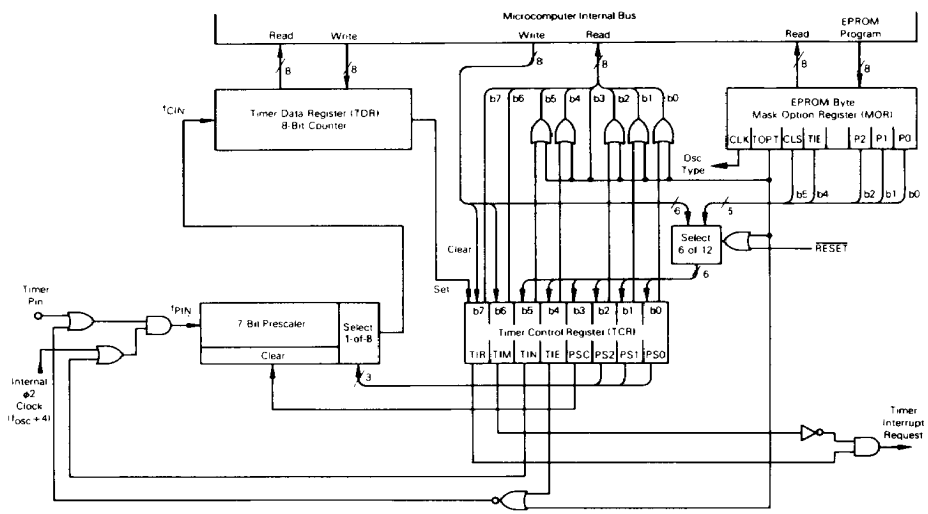


Figure 9. Timer Block Diagram

and 3) executing the interrupt routine. Timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic 1; however, TCR bit 3 always reads as a logic 0 to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. TDR is unaffected by reset.

SOFTWARE CONTROLLED MODE

The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TIE and TIN). The following paragraphs describe the different modes.

Timer Input Mode 1

When TIE and TIN are both programmed to zero, the timer input is from the internal clock (phase 2) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement.

Timer Input Mode 2

When TIE = 1 and TIN = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TIE = 0 and TIN = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TIE and TIN are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts.

MOR CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to logic 1. The timer circuits are the same as described in **SOFTWARE CONTROLLED MODE**. The logic levels of TCR bits 0, 1, 2, and 5 are determined during EPROM programming by the same bits in the MOR. Therefore, bits 0, 1, 2, and 5 in the MOR control the prescaler division and the timer clock selection. TIE (bit 4) and PSC (bit 3) in the TCR are set to a logic 1 when in the MOR controlled mode. TIM (bit 6) and TIR (bit 7) are controlled by the counter and software.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. All bits are read/write except bit 3. When the MOR TOPT = 1, then bits 5, 2, 1, and 0 in the TCR take on the corresponding bits of the MOR during reset.

7	6	5	4	3	2	1	0
TIR	TIM	1	1	1	1	1	1
RESET:							
0	1		U	U			

TCR with MOR TOPT = 1 (MC6805P2/P6 Emulation)

7	6	5	4	3	2	1	0
TIR	TIM	TIN	TIE	PSC	PS2	PS1	PS0
RESET:							
0	1						

TCR with MOR TOPT = 0 (Software Programmable Timer)

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt.
1 = Interrupt inhibited
0 = Interrupt enabled

TIN — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected ($f_{OSC}/4$)

TIE — TIMER External Enable
Used to enable external TIMER pin
1 = Enables external timer pin
0 = Disables external timer pin

PSC — Prescaler Clear
Write only bit. Writing a 1 to this bit resets the prescaler to zero. A read of this location always indicates a zero.

PS2, PS1, PS0 — Prescaler Select Bits
Decoded to select one of eight outputs of the prescaler

PS2	PS1	PS0	Prescaler Division
0	0	0	1 (Bypass Prescaler)
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

MASK OPTION REGISTER (MOR)

The MOR is implemented in EPROM and contains all zeros prior to programming. This register is not affected by reset. The MOR bits are described in the following paragraphs.

7	6	5	4	3	2	1	0
CLK	TOPT	CLS	TIE		P2	P1	P0

CLK — Clock (oscillator type)
1 = Resistor Capacitor (RC)
0 = Crystal

TOPT — Timer Option
1 = MC6805P2/P6 type timer/prescaler. All bits except 6 and 7 of the TCR are invisible to the user. Bits 5, 2, 1, and 0 of the MOR determine the equivalent MC6805P2/P6 mask options.
0 = All TCR bits are implemented as a software programmable timer. The state of MOR bits 5, 4, 2, 1, and 0 sets the initial values of their respective TCR bits.

CLS — Timer/Prescaler Clock Source
1 = External TIMER pin
0 = Internal clock

TIE — Timer External Enable
Not used if TOPT = 1. Sets the initial value of TIE in the TCR if TOPT = 0.
1 = Not used
0 = Sets initial value of TIE in the TCR

P2, P1, P0
The logical levels of these bits, when decoded, select one of eight outputs on the timer prescaler.

P2	P1	P0	Prescaler Division
0	0	0	1 (Bypass Prescaler)
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

PROGRAMMING CONTROL REGISTER (PCR)

The PCR is an 8-bit register which provides the necessary control bits to program the EPROM. The bootstrap program manipulates the PCR when programming, so the user need not be concerned with PCR in most applications.

7	6	5	4	3	2	1	0
1	1	1	1	1	$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$
RESET:							
U	U	U	U	U	U	1	1

$\overline{PLE}$ — Programming Latch Enable
Controls address and data being latched into the EPROM. Set during reset, but may be cleared anytime.
1 = Read EPROM
0 = Latch address and data on EPROM

$\overline{PGE}$ — Program Enable
Enables programming of EPROM. Must be set when changing the address and data. Set during reset.
1 = Inhibit EPROM programming
0 = Enable EPROM programming (if $\overline{PLE}$ is low)

$\overline{VPON}$ — Vpp On
A read-only bit that indicates high voltage at the Vpp pin. When set to "1", disconnects $\overline{PGE}$ and $\overline{PLE}$ from the chip.
1 = No high voltage on Vpp pin
0 = High voltage on Vpp pin

NOTE

$\overline{VPON}$ being "0" does not indicate that the Vpp level is correct for programming. It is used as a safety interlock for the user in the normal operating mode.

$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$	Programming Conditions
0	0	0	Programming Mode (Program EPROM Byte)
1	0	0	PGE and PLE Disabled from System
0	1	0	Programming Disabled (Latch Address and Data in EPROM)
1	1	0	PGE and PLE Disabled from System
0	0	1	Invalid State: PGE = 0 if PLE = 0
1	0	1	Invalid State: PGE = 0 if PLE = 0
0	1	1	"High Voltage" on Vpp
1	1	1	PGE and PLE Disabled from System (Operating Mode)

EPROM PROGRAMMING

PROGRAMMING

The MCU bootstrap program can be used to program the MCU EPROM.

A 2764 UV EPROM must first be programmed with the same information that is to be transferred to the MCU EPROM. Refer to application note, *MC68705P3/R3/U3 8-bit EPROM Microcomputer Programming Module* (AN-857 Rev 2) for a schematic diagram and instructions on programming the MCU EPROM.

EMULATION

The MC68705P3 emulates the MC6805P2 and MC6805P6 "exactly." The MC6805P2/P6 mask features are implemented in the mask-option register (MOR) EPROM byte on the MC68705P3. A few minor exceptions to the exactness of emulation are listed below:

1. The MC68705P2/P6 "future ROM" area is implemented in the MC68705P3, and these 704 bytes must

be left unprogrammed to accurately simulate the MC6805P2/P6. The MC6805P2/P6 read all "0s" from this area.

2. The reserved ROM areas in the MC6805P2/P6 and the MC68705P3 have different data stored in them. This data is subject to change without notice. The MC6805P2/P6 use the reserved ROM for the self-check feature, and the MC68705P3 uses this area for the bootstrap program.
3. The MC6805P2/P6 read all "1s" in its 48-byte "future RAM" area. This RAM is not implemented in the MC6805P2/P6 mask ROM versions but is implemented in the MC68705P3.
4. The V_{PP} line (pin 6) in the MC68705P3 must be tied to V_{CC} for normal operation. In the MC6805P2 P6, pin 6 is the NUM pin and is grounded in normal operation.
5. The LVI feature is not available in the MC68705P3. Processing differences are not presently compatible with proper design of this feature in the EPROM version.

The operation of all other circuitry has been exactly duplicated or designed to function identically in both devices including interrupts, timer, data ports, and data direction registers (DDRs). A design goal has been to provide the user with a safe, inexpensive way to verify a program and system design before committing to a factory programmed ROM.

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

— Continued —

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)

— Continued —

Function	Mnemonic
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

3

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

OPCODE MAP SUMMARY

Table 2 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code coversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true.

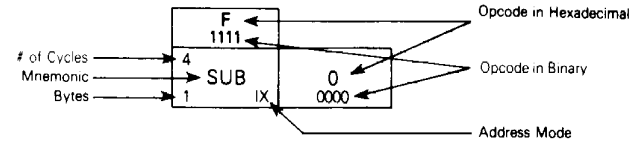
Table 2. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 4 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 4 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 4 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 4 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 4 BSC	4 BCC 2 REL	6 LSR 2 DIR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 4 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 4 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 4 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		4 STA 2 DIR	5 STA 3 EXT	6 STA 3 IX2	5 STA 2 IX1	4 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 4 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 4 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 4 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 4 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 4 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		4 JMP 2 DIR	5 JMP 3 EXT	6 JMP 3 IX2	5 JMP 2 IX1	4 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 4 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	8 JSR 2 DIR	9 JSR 3 EXT	8 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 4 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 4 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this 2-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this 3-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single 2-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write only registers (registers at $004, $005, and $006). A read operation on these registers always returns "1". Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single 3-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltages EPROM Programming Voltage (V_{PP} Pin) All Others	V_{PP} V_{in}	−0.3 to +22.0 −0.3 to +7.0	V V
TIMER Pin (Normal Mode) TIMER Pin (Bootstrap Programming Mode) All Others	V_{in} V_{in}	−0.3 to +15.0 −0.3 to +7.0	V V
Operating Temperature Range	T_A	T_L to T_H 0 to +70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Cerdip	T_J	150	°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip	θ_{JA}	60	°C/W

POWER CONSIDERATIONS

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

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

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts – Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins – User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

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

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

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

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

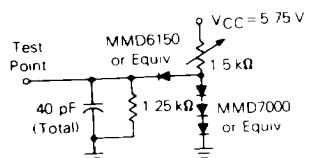


Figure 10. TTL Equivalent Test Load (Port B)

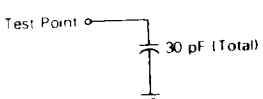


Figure 11. CMOS Equivalent Test Load (Port A)

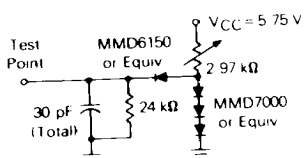


Figure 12. TTL Equivalent Test Load (Ports A and C)

ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC}	
(V_{CC} < 4.75)		V_{CC} −0.5	—	V_{CC}	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	**	V_{CC}	
(V_{CC} < 4.75)		V_{CC} −0.5	**	V_{CC}	
All Other		2.0	—	V_{CC}	
Input High Voltage (TIMER Pin)	V_{IH}				V
Timer Mode		2.0	—	V_{CC}	
Bootstrap Programming Mode		9.0	12.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		−0.3	—	0.8	
$\overline{INT}$		−0.3	**	1.5	
All Other		−0.3	—	0.8	
Internal Power Dissipation (No Port Loading, V_{CC} = 5.25 V, T_A = 0°C)	P_{INT}	—	450	TBD	mW
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other		—	10	—	
$\overline{INT}$ Zero-Crossing Voltage, through a Capacitor	V_{INT}	2.0	—	4.0	V_{acp-p}
$\overline{RESET}$ Hysteresis Voltage					V
Out of Reset Voltage	V_{IRES+}	2.1	—	4.0	
Into Reset Voltage	V_{IRES-}	0.8	—	2.0	
Programming Voltage (V_{PP} Pin)	V_{PP}*				V
Programming EPROM		20.0	21.0	22.0	
Operating Mode		4.0	V_{CC}	5.75	
Input Current	I_{in}				μA
TIMER (V_{in} = 0.4 V)		—	—	20	
$\overline{INT}$ (V_{in} = 0.4 V)		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option)		—	—	10	
(V_{in} = 0.4 V Crystal Option)		—	—	−1600	
$\overline{RESET}$ (V_{in} = 0.8 V)		−4.0	—	−40	
(External Capacitor Changing Current)					

*V_{PP} is pin 6 on the MC68705P3 and is connected to V_{CC} in the normal operating mode. In the MC6805P2, pin 6 is NUM and is connected to V_{SS} in the normal operating mode. The user must allow for this difference when emulating the MC6805P2 ROM-based MCU.

**Due to internal biasing, this input (when not used) floats to approximately 2.0 V.

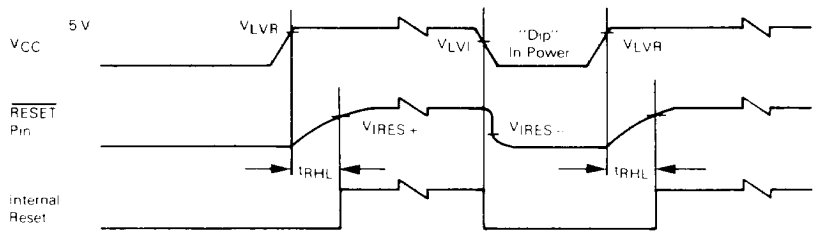


Figure 13. Power and Reset Timing

PORT DC ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 µA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 µA	V_{OH}	V_{CC} −10	—	—	V
Input High Voltage, I_{Load} = −300 µA (Max)	V_{IH}	2.0	—	V_{CC} +0.7	V
Input Low Voltage, I_{Load} = −500 µA (Max)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	µA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	µA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	(V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 µA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	20	µA
Port C					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 µA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	20	µA

SWITCHING CHARACTERISTICS (V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency Normal	f_{osc}	 0.4	 —	 4.2	MHz
Instruction Cycle Time ($4/f_{osc}$)	t_{cyc}	0.950	—	10	µs
$\overline{INT}$ or Timer Pulse Width (See Interrupt Section)	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Cap = 1.0 µF)	t_{RHL}	100	—	—	ms
$\overline{INT}$ Zero Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock Duty Cycle (EXTAL)	—	40	50	60	%

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 20° to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage (Vpp Pin)	V_{PP}	20.0	21.0	22.0	V
Vpp Supply Current V_{PP} = 5.25 V V_{PP} = 21.0 V	I_{PP}	 — —	 — —	 8 30	mA
Programming Oscillator Frequency	f_{oscp}	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (TIMER Pin) I_{in} = 100 µA Max	V_{IHTP}	9.0	12.0	15.0	V

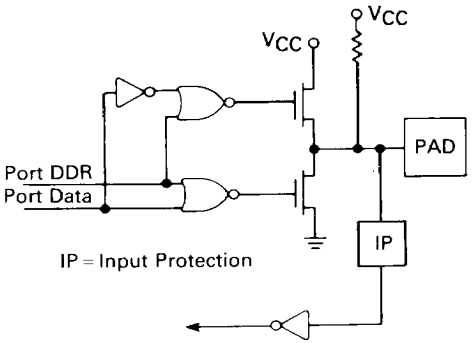


Figure 14. Port A Logic Diagram

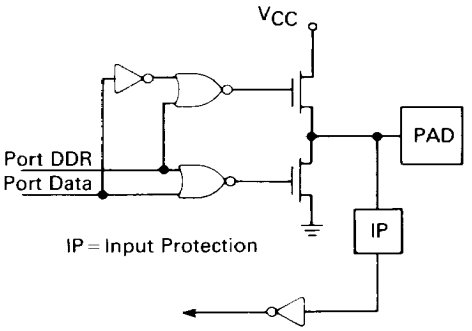


Figure 15. Port B and Port C Logic Diagram

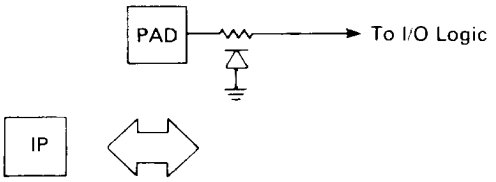


Figure 16. Typical Input Protection

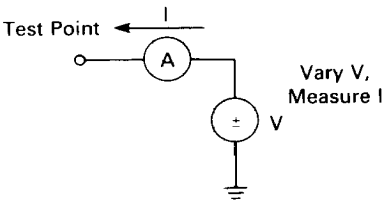


Figure 17. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC68705P3.

Table 3. Generic Information

Package Type	Internal Clock Frequency (MHz)	Temperature	Order Number
Cerdip (S Suffix)	1.0	0° to 70°C	MC68705P3S
Cerdip (S Suffix)	1.0	−40° to +85°C	MC68705P3CS

MECHANICAL DATA

PIN ASSIGNMENTS