Grokking the MCS-48 System

Version 3.0, September 2006

Preface

This document came into existence as the effort to collect and maintain the information about the MCS-48 microcontroller family I have access to.

The collected paperwork consists of several copied sheets from Intel databooks that age over time, slowly fading away. In addition, the resources in electronic form for this microcontroller family were never complete nor exhaustive. As nowadays, not everyone is lucky enough to have access to these old, big databooks, this document intends to transport all the knowledge necessary to work with these grandfathers.

Please note that this document does not claim correctness of the contained information. Nor should you ever use it as the only reference for building critical applications.

Five separate sources contributed to this document:

- Two different versions of the "MCS-48™ Microcomputer User's Manual"
  The chapter "The Expanded MCS-48™ System" was found in a 1978's copy of the User's Manual while the chapter "The Single Component MCS®-48 System" was probably part of a later revision of the User's Manual.
- Das 8039, 8048, 8748 Brevier
  H. Weidner, Braunschweig
  The first part of this booklet is a translation of "The Single Component MCS®-48 System" into the German language while the second part gives a very detailed description of the family's instruction set.
- HMOS Single-Component 8-Bit Microcomputer
  Intel Corporation, August 1982, Order Number 210677-001
  A 9-page datasheet with an excellent instruction set table.
- Some scanned pages with details about the 8243, apparently from an old Intel databook.
- A set of copied pages containing the 8245 product specification. Pages 6 and 22 are missing.

September 2006

Arnim Läuger
mailto: arnim.laeuger <at> gmx.net

HMOS Single-Component 8-Bit Microcomputer

- **High Performance HMOS**
- **Interval Timer/Event Counter**
- **Two Single Level Interrupts**
- **Single 5-Volt Supply**
- **Over 96 Instructions; 90% Single Byte**
- **Reduced Power Consumption**
- **Compatible with 8080/8085 Peripherals**
- **Easily Expandable Memory and I/O**
- **Up to 1.36 µs Instruction Cycle**
  All Instructions 1 or 2 Cycles

The Intel MCS®-48 family are totally self-sufficient, 8-bit parallel computers fabricated on single silicon chips using Intel's advanced N-channel silicon gate HMOS process.

The family contains 27 I/O lines, an 8-bit timer/counter and on-board oscillator/clock circuits. For systems that require extra capability, the family can be expanded using MCS®-80/MCS®-85 peripherals.

To minimize development problems and provide maximum flexibility, a logically and functionally pin-compatible version of the ROM devices with UV-erasable user-programmable EPROM program memory is available with minor differences.

These microcomputers are designed to be efficient controllers as well as arithmetic processors. They have extensive bit handling capability as well as facilities for both binary and BCD arithmetic. Efficient use of program memory results from an instruction set consisting mostly of single byte instructions and no instructions over 2 bytes in length.

Device	Internal Memory		RAM Standby
8050AH	4K x 8 ROM	256 x 8 RAM	yes
8049H	2K x 8 ROM	128 x 8 RAM	yes
8048H	1K x 8 ROM	64 x 8 RAM	yes
8040AHL	none	256 x 8 RAM	yes
8039HL	none	128 x 8 RAM	yes
8035HL	none	64 x 8 RAM	yes
8749H	2K x 8 EPROM	128 x 8 RAM	yes
8748H	1K x 8 EPROM	64 x 8 RAM	yes

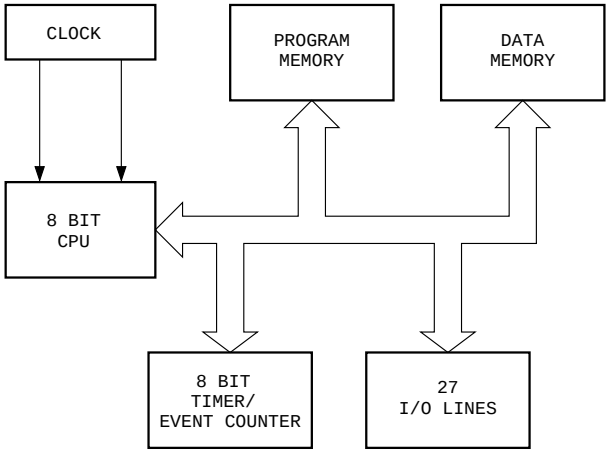


Figure 1.
Block Diagram

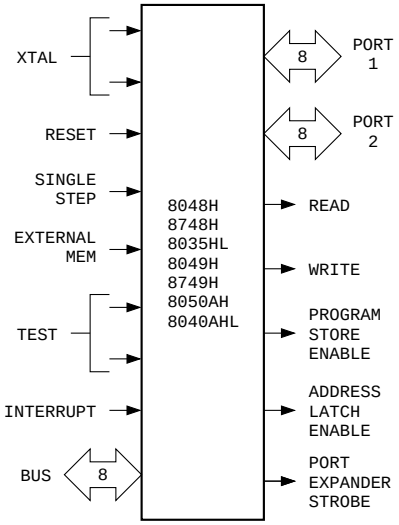


Figure 2.
Logic Symbol

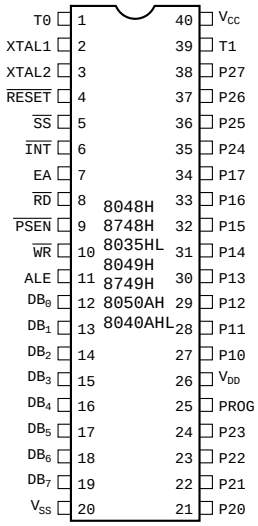


Figure 3.
Pin Configuration

THE SINGLE COMPONENT MCS®-48 SYSTEM

1.0 INTRODUCTION

Sections 2 through 5 describe in detail the functional characteristics of the 8748H and 8749H EPROM, 8048AH/8049AH/8050AH ROM and 8035AHL/8039AHL/8040AHL CPU single component microcomputers. Unless otherwise noted, details within these sections apply to all versions. This chapter is limited to those functions useful in single-chip implementations of the MCS®-48. The Chapter on the Expanded MCS®-48 System discusses functions which allow expansion of program memory, data memory and input output capability.

2.0 ARCHITECTURE

The following sections break the MCS-48 Family into functional blocks and describe each in detail. The following description will use the 8048AH as the representative product for the family. See Figure 4.

2.1 Arithmetic Section

The arithmetic section of the processor contains the basic data manipulation functions of the 8048AH and can be divided into the following blocks:

- Arithmetic Logic Unit (ALU)
- Accumulator
- Carry Flag
- Instruction Decoder

In a typical operation data stored in the accumulator is combined in the ALU with data from another source on the internal bus (such as a register or I/O port) and the result is stored in the accumulator or another register.

The following is more detailed description of the function of each block.

INSTRUCTION DECODER

The operation code (op code) portion of each program instruction is stored in the Instruction Decoder and is converted to outputs which control the function of each of the blocks of the Arithmetic Section. These lines control the source of data and the destination register as well as the function performed in the ALU.

ARITHMETIC LOGIC UNIT

The ALU accepts 8-bit data words from one or two sources and generates an 8-bit result under control of the Instruction Decoder. The ALU can perform the following functions:

- Add With or Without Carry
- AND, OR, Exclusive OR
- Increment/Decrement
- Bit Complement
- Rotate Left, Right
- Swap Nibbles
- BCD Decimal Adjust

If the operation performed by the ALU results in a value represented by more than 8 bits (overflow of most significant bit), the Carry Flag is set in the Program Status Word.

ACCUMULATOR

The accumulator is the single most important data register in the processor, being one of the sources of input to the ALU and often the destination of the result of operations performed by the ALU. Data to and from I/O ports and memory also normally passes through the accumulator.

2.2 Program Memory

Resident program memory consists of 1024, 2048 or 4096 words eight bit wide which are addressed by the program counter. In the 8748H and the 8749H this memory is user programmable and erasable EPROM; in the 8048AH/8049AH/8050AH the memory is ROM which is mask programmable at the factory. The 8035AHL/8039AHL/8040AHL has no internal program memory and is used with external memory devices. Program code is completely interchangeable among the various versions. To access the upper 2K of program memory in the 8050AH and other MCS-48 devices, a select memory bank and a JUMP or CALL instruction must be executed to cross the 2K boundary.

There are three locations in Program Memory of special importance as shown in Figure 5.

LOCATION 0
Activating the Reset line of the processor causes the first instruction to be fetched from location 0.

LOCATION 3
Activating the Interrupt input line of the processor (if interrupt is enabled) causes a jump to subroutine at location 3.

LOCATION 7
A timer/counter interrupt resulting from timer counter overflow (if enabled) causes a jump to subroutine at location 7.

Therefore, the first instruction to be executed after initialization is stored in location 0, the first word of an external interrupt service subroutine is stored in location 3 and the first word of a timer/counter service routines is stored in location 7. Program memory can be used to

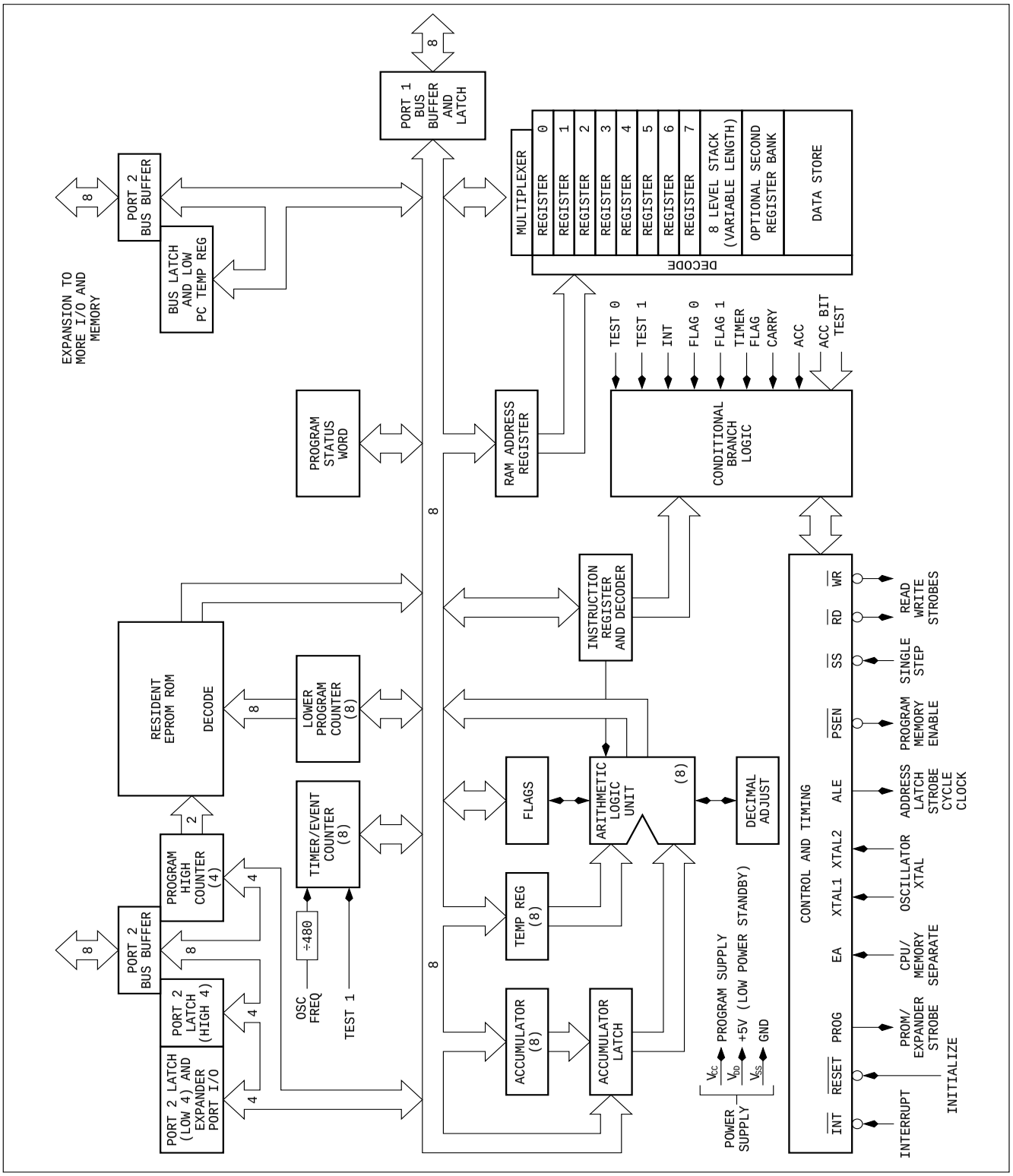


Figure 4. 8748H/8048H/8749AH/8050AH Block Diagram

store constants as well as program instructions. Instructions such as MOVP and MOVP3 allow easy access to data "lookup" tables.

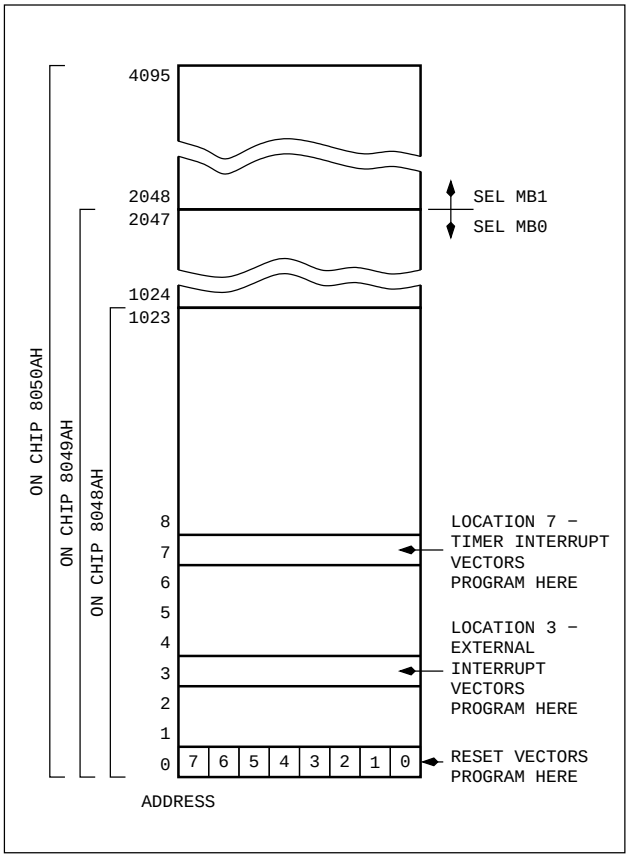


Figure 5. Program Memory Map

2.3 Data Memory

Resident data memory is organized as 64, 128 or 256 by 8-bits wide in the 8048AH, 8049AH and 8058AH. All locations are indirectly addressable through either of two RAM Pointer Registers which reside at address 0 and 1 of the register array. In addition, as shown in Figure 6, the first 8 locations (0-7) of th array are designated as working registers and are directly addressable by several instructions. Since these registers are more easily addressed, they are usually used to store frequently accessed intermediate results. The DJNZ instruction makes very efficient use of the working registers as program loop counters by allowing the programmer to decrement and test the register in a single instruction.

By executing a Register Bank Switch instruction (SEL RB), RAM locations 24-31 are designated as the working registers in place of locations 0-7 and are then directly addressable. This second bank of working registers may be used as an extension of the first bank or reserved for use during interrupt service subroutines allowing the registers of Bank 0 used in the main program to be instantly "saved" by a Bank Switch. Note that if this second bank is not used, locations 24-31 are still addressable as general purpose RAM. Since the two RAM pointer Registers R0 and R1 are a part of the working register array, bank switch effectively creates two more pointer registers (R0/ and R1/) which can be used with R0 and R1 to easily access up to four separate working areas in RAM at one time. RAM locations 8-23 also serve a dual role in that they contain the program counter stack as explained in Section 2.6. These locations are addressed by the Stack Pointer during subroutine calls as well as by RAM Pointer Registers R0 and R1. If the level of subroutine nesting is less than 8, all stack registers are not required and can be used as general purpose RAM locations. Each level of subroutine nesting not used provides the user with two additional RAM locations.

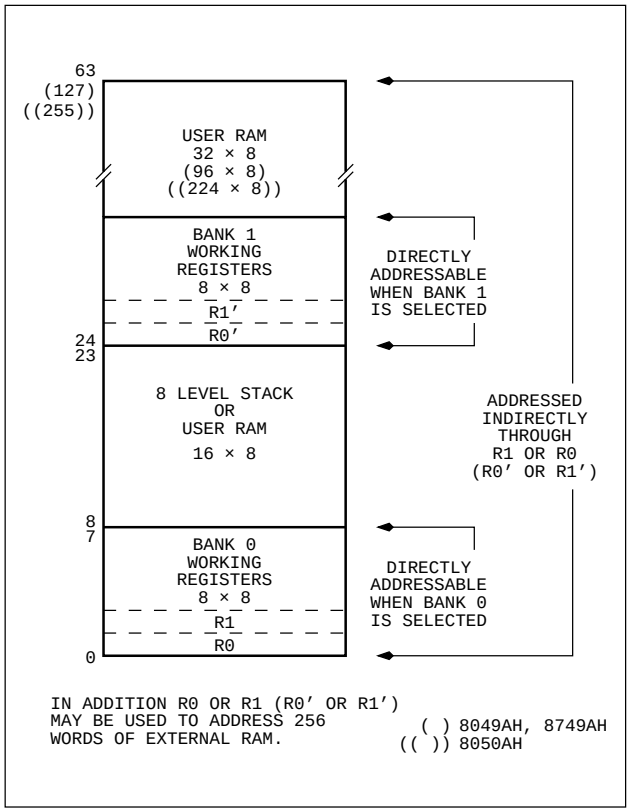
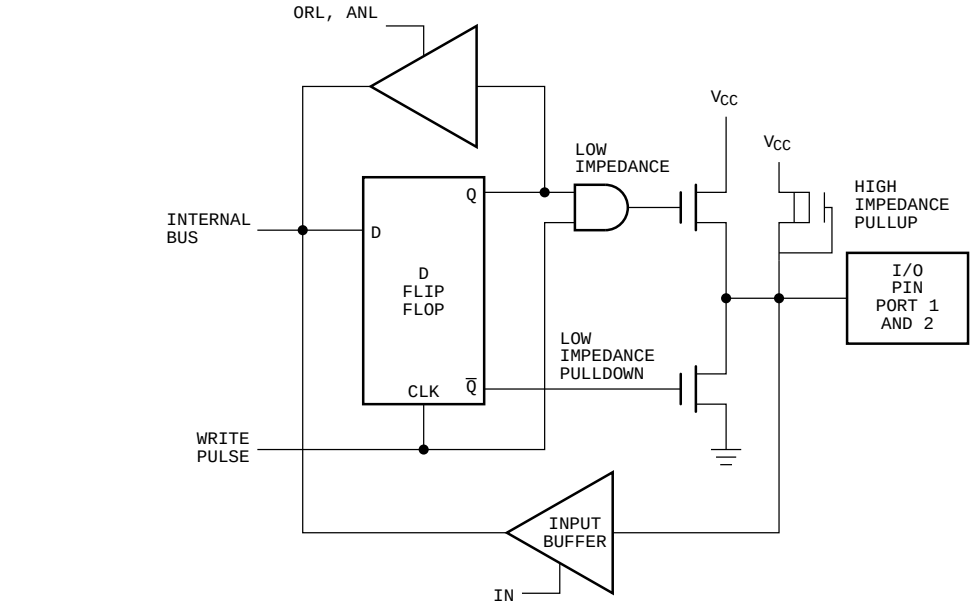
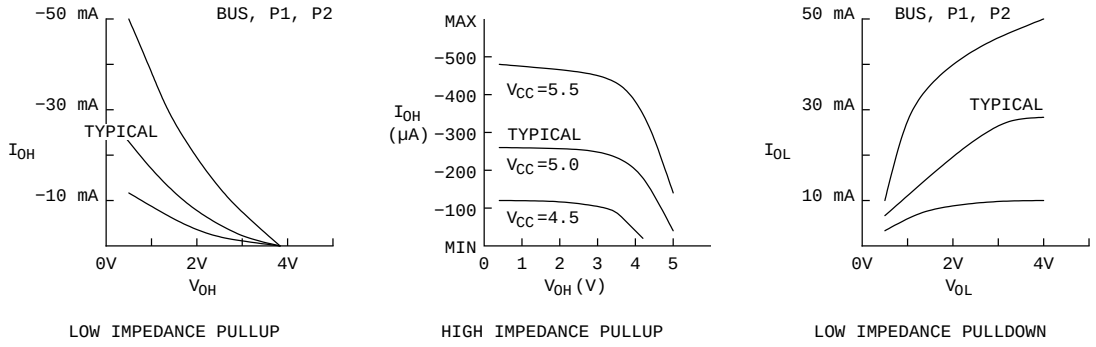


Figure 6. Data Memory Map





These graphs are for informational purpose only and are not guaranteed minimums or maximums.

Figure 7. "Quasi-bidirectional" Port Structure

2.4 Input/Output

The 8048AH has 27 lines which can be used for input or output functions. These lines are grouped as 3 ports of 8 lines each which serve as either inputs, outputs or bidirectional ports and 3 "test" inputs which can alter program sequences when tested by conditional jump instructions.

PORTS 1 AND 2

Ports 1 and 2 are each 8 bits wide and have identical characteristics. Data written to these ports is statically latched and remains unchanged until rewritten. As input ports these lines are non-latching, i. e., inputs must be present until read by an input instruction. Inputs are fully TTL compatible and outputs will drive one standard TTL load.

The lines of ports 1 and 2 are called quasi-bidirectional because of a special output circuit structure which allows each line to serve as an input and output or both even though outputs are statically latched. Figure 7 shows the circuit configuration in detail. Each line is continuously pulled to V_{CC} through a resistive device of relatively high impedance.

This pullup is sufficient to provide the source current for a TTL high level yet can be pulled low by a standard TTL gate thus allowing the same pin to be used for both input and output. To provide fast switching times in a "0" to "1" transition a relatively low impedance device is switched in momentarily ($\approx$ 1/5 of a machine cycle) whenever a "1" is written to the line. When a "0" is written to the line, a low impedance device overcomes the light pullup and provides TTL current sinking capability. Since the pulldown transistor is a low impedance device a "1" must first be written to any line which is to be used as an input. Reset initializes all lines to the high impedance "1" state.

It is important to note that the ORL and the ANL are read/write operations. When executed, the µC "reads" the port, modifies the data according to the instruction, then "writes" the data back to the port. The "writing" (essentially an OUTL instruction) enables the low impedance pull-up momentarily again even if the data was unchanged from a "1". This specifically applies to configurations that have inputs and outputs mixed together on the same port. See also section 8 in the Expanded MCS-48 System chapter.

BUS

Bus is also an 8-bit port which is a true bidirectional port with associated input and output strobes. If the bidirectional feature is not needed, Bus can serve as either a statically latched output port or non-latching input port. Input and output lines on this port cannot be mixed however.

As a static port, data is written and latched using the OUTL instruction and inputted using the INS instruction. The INS and OUTL instructions generate pulses on the corresponding RD' and WR' output strobe lines; however, in the static port mode they are generally not used. As a bidirectional port the MOVX instructions are used to read and write the port. A write to the port generates a pulse on the WR' output line and output data is valid at the trailing edge of WR'. A read of the port generates a pulse on the RD' output line and input data must be valid at the trailing edge of RD'. When not being written or read, the BUS lines are in a high impedance state. See also sections 7 and 8 in the Expanded MCS-48 System chapter.

2.5 Test and INT Inputs

Three pins serve as inputs and are testable with the conditional jump instructions. These are T0, T1 and INT'. These pins allow inputs to cause program branches without the necessity to load an input port into the accumulator. The T0, T1 and INT' pins have other possible functions as well. See the pin description in Section 3.

2.6 Program Counter and Stack

The Program Counter is an independent counter while the Program Counter Stack is implemented using pairs of registers in the Data Memory Area. Only 10, 11 or 12 bits of the Program Counter are used to address the 1024, 2048 or 4096 words of on-board program memory of the 8048AH, 8049AH or 8050AH, while the most significant bits can be used for external Program Memory fetches. See Figure 8. The Program Counter is initialized to zero by activating the Reset line.

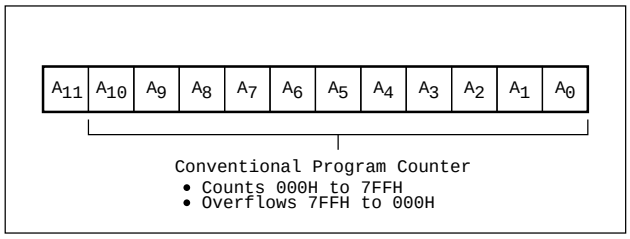


Figure 8. Program Counter

An interrupt or CALL to a subroutine causes the contents of the program counter to be stored in one of the 8 register pairs of the Program Counter Stack as shown in Figure 9. The pair to be used is determined by a 3-bit Stack Pointer which is part of the Program Status Word (PSW).

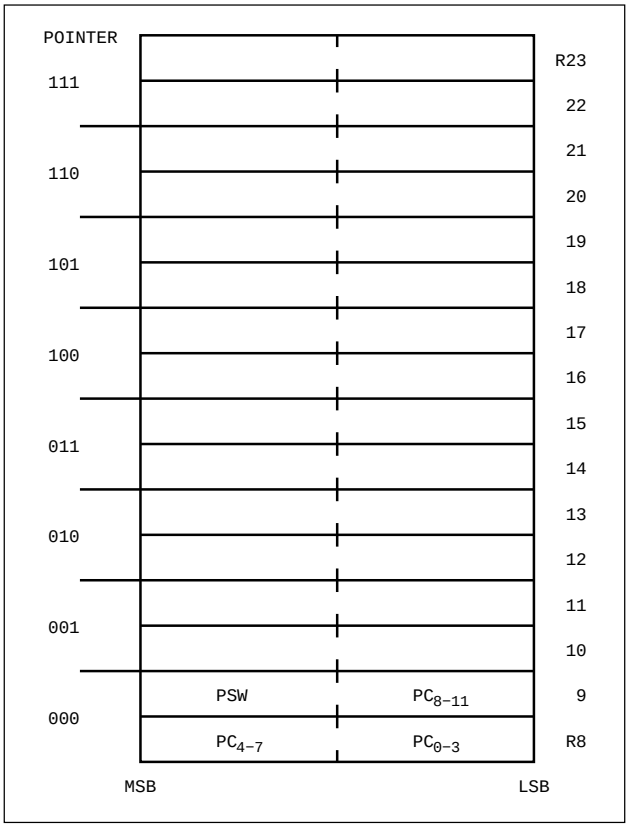


Figure 9. Program Counter Stack

Data RAM locations 8-23 are available as stack registers and are used to store the Program Counter and 4 bits of PSW as shown in Figure 9. The Stack Pointer when initialized to 000 points to RAM location 8 and 9. The first subroutine jump or interrupt results in the program counter contents being transferred to locations 8 and 9 of the RAM array. The stack pointer is then incremented by one to point to locations 10 and 11 in anticipation of another CALL. Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur, the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The end of a subroutine, which is signaled by a return instruction (RET or RETR), causes the Stack Pointer to be decremented and the contents of the resulting register pair to be transferred to the Program Counter.

2.7 Program Status Word

An 8-bit status word which can be loaded to and from the accumulator exists called the Program Status Word (PSW). Figure 10 shows the information available in the word. The Program Status Word is actually a collection of flip-flops throughout the machine which can be read or written as a whole. The ability to write the PSW allows for easy restoration of machine status after a power down sequence.

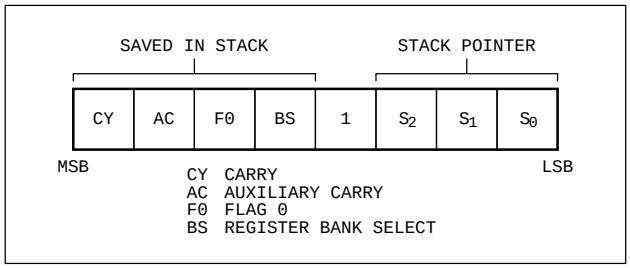


Figure 10. Program Status Word (PSW)

The upper four bits of PSW are stored in the Program Counter Stack with every call to subroutine or interrupt vector and are optionally restored upon return with the RETR instruction. The RET return instruction does not update PSW.

The PSW bit definitions are as follows:

Bits 0-2: Stack Pointer bits (S_0, S_1, S_2)

Bit 3: Not used ("1" level when read)

Bit 4: Working Register Bank Switch Bit (BS)
0 = Bank 0
1 = Bank 1

Bit 5: Flag 0 bit (F0) user controlled flag which can be complemented or cleared and tested with the conditional jump instruction JF0.

Bit 6: Auxiliary Carry (AC) carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A.

Bit 7: Carry (CY) carry flag which indicates that the previous operation has resulted in overflow of the accumulator.

2.8 Conditional Branch Logic

The conditional branch logic within the processor enables several conditions internal and external to the processor to be tested by the user program. By using the conditional jump instruction the conditions that are listed in Table 1 can affect a change in the sequence of the program execution.

Table 1

Device Testable	Jump Condition (Jump On)	
Accumulator	All zeros	not all zeros
Accumulator Bit	---	1
Carry Flag	0	1
User Flags (F0, F1)	---	1
Timer Overflow Flag	---	1
Test Inputs (T0, T1)	0	1
Interrupt Input (INT)	0	---

2.9 Interrupt

An interrupt sequence is initiated by applying a low "0" level input to the INT' pin. Interrupt is level triggered and active low to allow "WIRE ORing" of several interrupt sources at the input pin. Figure 12 shows the interrupt logic of the 8048AH. The interrupt line is sampled every instruction cycle and when detected causes a "call to subroutine" at location 3 in program memory as soon as all cycles of the current instruction are complete. On 2-cycle instructions the interrupt line is sampled on the 2nd cycle only. INT' must be held low for at least 3 machine cycles to ensure proper interrupt operations. As in any CALL to subroutine, the Program Counter and Program Status word is saved in the stack. For a description of this operation see the previous section, Program Counter and Stack. Program Memory location 3 usually contains an unconditional jump to an interrupt service subroutine elsewhere in program memory. The end of an interrupt service subroutine is signaled by the execution of a Return and Restore Status instruction RETR. The interrupt system is single level in that once an interrupt is detected all further interrupt requests are ignored until execution of an RETR reenables the interrupt input logic. This occurs at the beginning of the second cycle of the RETR instruction. This sequence holds true also for an internal interrupt generated by timer overflow. If an internal timer/counter generated interrupt and an external source will be recognized at the same time, the external source will be recognized. See the following Timer/Counter section for a description of timer interrupt. If needed, a second external interrupt can be created by enabling the timer/counter interrupt, loading FFH in the Counter (one less than terminal count) and enabling the event counter mode. A "1" to "0" transition on the T1 input will then cause an interrupt vector to location 7.

INTERRUPT TIMING

The interrupt input may be enabled or disabled under Program Control using the EN I and DIS I instructions. Interrupts are disabled by Reset and remain so until enabled by the user's program. An interrupt request must be removed before the RETR instruction is executed upon return from the service subroutine otherwise the processor will re-enter the service routine immediately. Many peripheral devices prevent this situation by resetting their interrupt request line whenever the processor accesses (Reads or Writes) the peripheral's data buffer register. If the interrupt device does not require access by the processor, one output line of the 8048AH may be designated as an "interrupt acknowledge" which is activated by the service subroutine to reset the interrupt request. The INT' pin may also be tested using the conditional jump instruction JNI. This instruction may be used to detect the presence of a pending interrupt before interrupts are enabled. If interrupt is left disabled, INT' may be used as another test input like T0 and T1.

2.10 Timer/Counter

The 8048AH contains a counter to aid the user in counting external events and generating accurate time delays without placing a burden on the processor for these functions. In both modes the counter operation is the same, the only difference being the source of the input to the counter. The timer/event counter is show in Figure 11.

COUNTER

The 8-bit binary counter is presettable and readable with the MOV instructions which transfer the contents of the accumulator to the counter and vice versa. The counter content may be affected by Reset and should be initialized by software. The counter is stopped by a Reset or STOP TCNT instruction and remains stopped until started as a timer by a STRT T instruction or as an event counter by a STRT CNT instruction. Once started, the counter will increment to this maximum count (FF) and overflow to zero continuing its count until stopped by a STOP TCNT instruction or Reset.

The increment from maximum count to zero (overflow) results in the setting of an overflow flag flip-flop and in the generation of an interrupt request. The state of the overflow flag is testable with the conditional jump instruction JTF. The flag is reset by executing a JTF or by Reset. The interrupt request is stored in a latch and then ORed with the external interrupt input INT. The timer interrupt may be enabled or disabled independently of external interrupt by the EN TCNTI and DIS TCNTI instructions. If enabled, the counter overflow will cause a subroutine call to location 7 where the timer or counter service routine may be stored.

If timer and external interrupts occur simultaneously, the external interrupt will be recognized and the Call will be to location 3. Since the timer interrupt is latched it will remain pending until the external device is serviced and

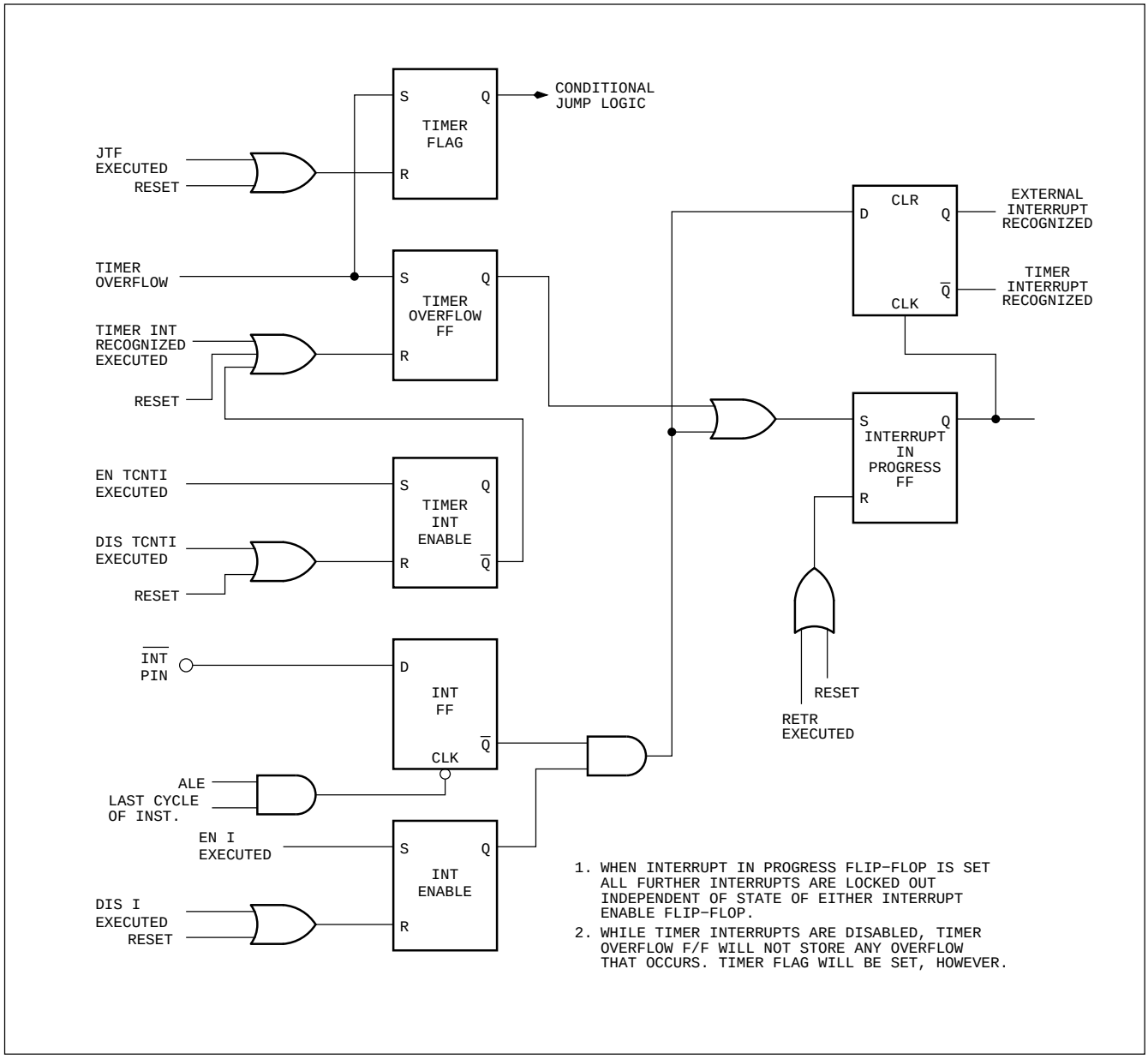


Figure 12. Interrupt Logic

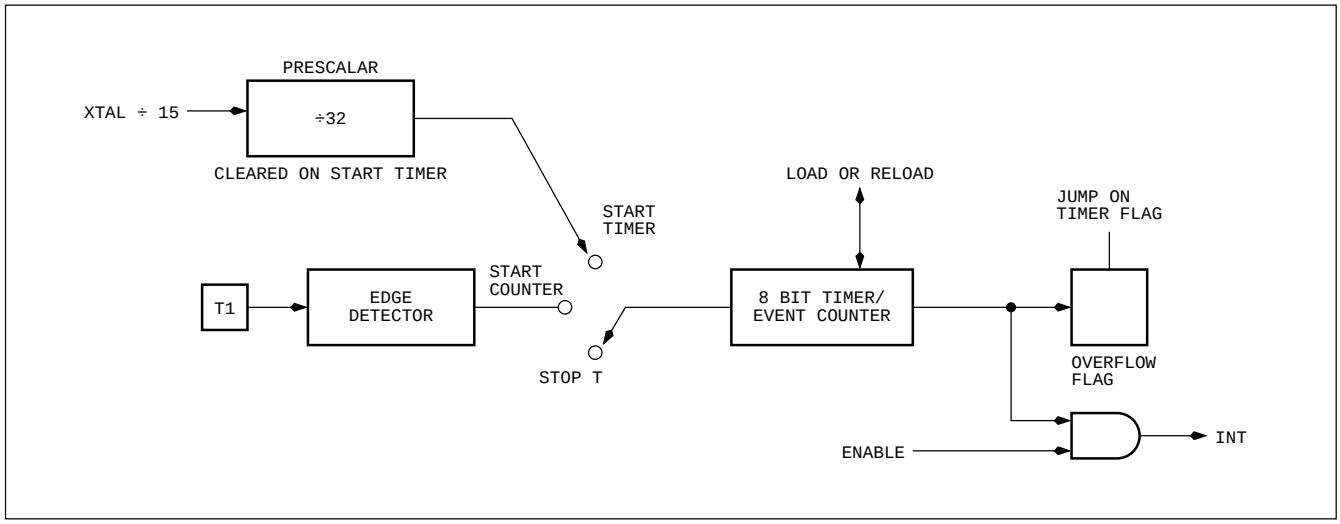


Figure 11. Timer/Event Counter

immediately be recognized upon return from the service routine. The pending timer interrupt is reset by the Call to location 7 or may be removed by executing a DIS TCNTI instruction.

AS AN EVENT COUNTER

Execution of a START CNT instruction connects the T1 input pin to the counter input and enables the counter. The T1 input is sampled at the beginning of state 3 or in later MCS-48 devices in state time 4. Subsequent high to low transitions on T1 will cause the counter to increment. T1 must be held low for at least 1 machine cycle to ensure it won't be missed. The maximum rate at which the counter may be incremented is once per three instruction cycles (every 5.7 µsec when using an 8 MHz crystal) --- there is no minimum frequency. T1 input must remain high for at least 1/5 machine cycle after each transition.

AS A TIMER

Execution of a START T instruction connects an internal clock to the counter input and enables the counter. The internal clock is derived by passing the basic machine cycle clock through a ÷ 32 prescaler. The prescaler is reset during the START T instruction. The resulting clock increments the counter every 32 machine cycles. Various delays from 1 to 256 counts can be obtained by presetting the counter and detecting overflow. Times longer than 256 counts may be achieved by accumulating multiple overflows in a register under software control. For time resolution less than 1 count an external clock can be applied to the T1 input and the counter operated in the event counter mode. ALE divided by 3 or more can serve as this external clock. Very small delays or "fine tuning" of larger delays can be easily accomplished by software delay loops.

Often a serial link is desirable in an MCS-48 family member. Table 2 lists the timer counts and cycles needed for a specific baud rate given a crystal frequency.

2.11 Clock and Timing Circuits

Timing generation for the 8048AH is completely self contained with the exception of a frequency reference which can be XTAL, ceramic resonator or external clock source. The Clock and Timing circuitry can be divided into the following functional blocks.

OSCILLATOR

The on-board oscillator is a high gain parallel resonant circuit with a frequency range of 1 to 11 MHz. The X1 external pin is the input to the amplifier stage while X2 is the output. A crystal or ceramic resonator connected between X1 and X2 provides the feedback and phase shift required for oscillation. If an accurate frequency reference is not required, ceramic resonator may be used in place of the crystal.

For accurate clocking, a crystal should be used. An externally generated clock may also be applied to X1—X2 as the frequency source. See the data sheet for more information.

Table 2. Baud Rate Generation

Frequency (MHz)		T_{CY}	T0 Prr(1/5 T_{CY})	Timer Prescaler (32 T_{CY})
	4 6 8 11	3.75 µs 2.50 µs 1.88 µs 1.36 µs	750 ns 500 ns 375 ns 275 ns	120 µs 80 µs 60.2 µs 43.5 µs
Baud Rate	4 MHz Timer Counts + Instr. Cycles	6 MHz Timer Counts + Instr. Cycles	8 MHz Timer Counts + Instr. Cycles	11 MHz Timer Counts + Instr. Cycles
110	75 + 24 Cycles .01% Error	113 + 20 Cycles .01% Error	151 + 3 Cycles .01% Error	208 + 28 Cycles .01% Error
300	27 + 24 Cycles .1% Error	41 + 21 Cycles .03% Error	55 + 13 Cycles .01% Error	76 + 18 Cycles .04% Error
1200	6 + 30 Cycles .1% Error	10 + 13 Cycles .1% Error	12 + 27 Cycles .06% Error	19 + 4 Cycles .12% Error
1800	4 + 20 Cycles .1% Error	6 + 30 Cycles .1% Error	9 + 7 Cycles .17% Error	12 + 24 Cycles .12% Error
2400	3 + 15 Cycles .1% Error	5 + 6 Cycles .4% Error	6 + 24 Cycles .29% Error	9 + 18 Cycles .12% Error
4800	1 + 23 Cycles 1.0% Error	2 + 19 Cycles .4% Error	3 + 14 Cycles .74% Error	4 + 25 Cycles .12% Error

STATE COUNTER

The output of the oscillator is divided by 3 in the State Counter to create a clock which defines the state times of the machine (CLK). CLK can be made available on the external pin T0 by executing an ENT0 CLK instruction. The output of CLK on T0 is disabled by Reset of the processor.

CYCLE COUNTER

CLK is then divided by 5 in the Cycle Counter to provide a clock which defines a machine cycle consisting of 5 machine states as shown in Figure 13. Figure 14 shows the different internal operations as divided into the machine states. This clock is called Address Latch Enable (ALE) because of its function in MCS-48 systems with external memory. It is provided continuously on the ALE output pin.

2.12 Reset

The reset input provides a means for initialization for the processor. This Schmitt-trigger input has an internal pullup device which in combination with an external 1 µF capacitor provides an internal reset pulse of sufficient length to guarantee all circuitry is reset, as shown in Figure 15. If the reset pulse is generated externally, the RESET pin must be held low for at least 10 milliseconds after the power supply is within tolerance. Only 5 machine cycles (6.8 µs @ 11 MHz) are required if power is already on and the oscillator has stabilized. ALE and PSEN (if EA = 1) are active while in Reset.

Reset performs the following functions:

1) Sets program counter to zero.
2) Sets stack pointer to zero.
3) Selects register bank 0.
4) Selects memory bank 0.
5) Sets BUS to high impedance state (except when EA = 5 V).
6) Sets Ports 1 and 2 to input mode.
7) Disables interrupts (timer and external).
8) Stops timer.
9) Clears timer flag.
10) Clears F0 and F1.
11) Disables clock output from T0.

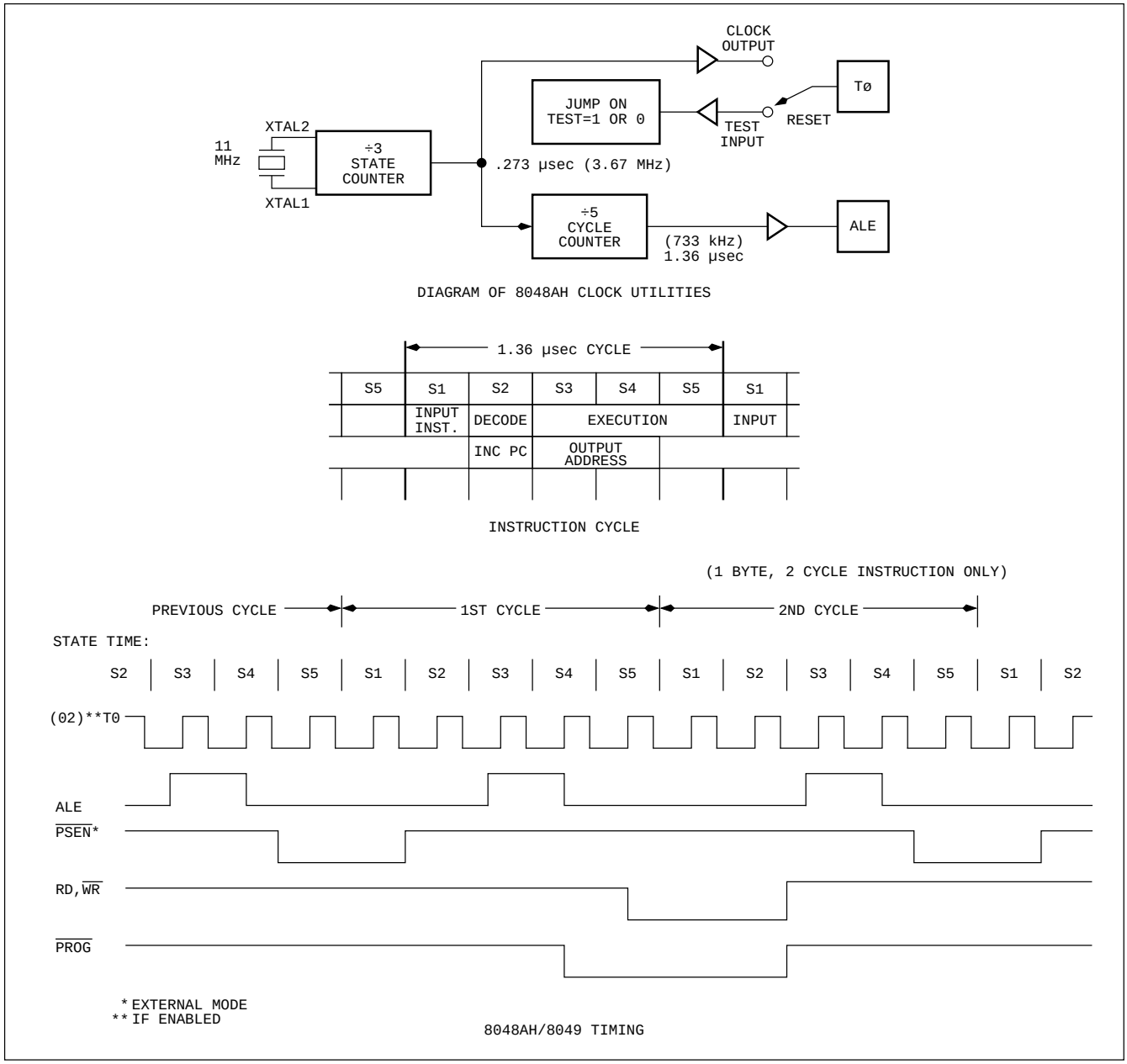


Figure 13. MCS-48 Timing Generation and Cycle Timing

2.13 Single-Step

This feature, as pictured in Figure 16, provides the user with a debug capability in that the processor can be stepped through the program one instruction at a time. While stopped, the address of the next instruction to be fetched is available concurrently on BUS and the lower half of Port 2. The user can therefore follow the program through each of the instruction steps. A timing diagram, showing the interaction between output ALE and input SS', is shown. The BUS buffer contents are lost during single step; however, a latch may be added to reestablish the lost I/O capability if needed. Data is valid at the leading edge of ALE.

	CYCLE 1					CYCLE 2				
INSTRUCTION	S1	S2	S3	S4	S5	S1	S2	S3	S4	S5
IN A,P	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	*INCREMENT TIMER	---	---	READ PORT	---	* ---	---
OUTL P,A	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	*INCREMENT TIMER	OUTPUT TO PORT	---	---	---	* ---	---
ANL P,=DATA	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	*INCREMENT TIMER	READ PORT	FETCH IMMEDIATE DATA	---	INCREMENT PROGRAM COUNTER	*OUTPUT TO PORT	---
ORL P,=DATA	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	*INCREMENT TIMER	READ PORT	FETCH IMMEDIATE DATA	---	INCREMENT PROGRAM COUNTER	*OUTPUT TO PORT	---
INS A,BUS	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	INCREMENT TIMER	---	---	READ PORT	---	* ---	---
OUTL BUS,A	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	INCREMENT TIMER	OUTPUT TO PORT	---	---	---	* ---	---
ANL BUS,=DATA	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	*INCREMENT TIMER	READ PORT	FETCH IMMEDIATE DATA	---	INCREMENT PROGRAM COUNTER	*OUTPUT TO PORT	---
ORL BUS,=DATA	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	*INCREMENT TIMER	READ PORT	FETCH IMMEDIATE DATA	---	INCREMENT PROGRAM COUNTER	*OUTPUT TO PORT	---
MOVX @R,A	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	OUTPUT RAM ADDRESS	INCREMENT TIMER	OUTPUT DATA TO RAM	---	---	---	* ---	---
MOVX A,@R	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	OUTPUT RAM ADDRESS	INCREMENT TIMER	---	---	READ DATA	---	* ---	---
MOVD A,Pi	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	OUTPUT OPCODE/ADDRESS	INCREMENT TIMER	---	---	READ P2 LOWER	---	* ---	---
MOVD Pi,A	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	OUTPUT OPCODE/ADDRESS	INCREMENT TIMER	OUTPUT DATA TO P2 LOWER	---	---	---	* ---	---
ANLD P,A	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	OUTPUT OPCODE/ADDRESS	INCREMENT TIMER	OUTPUT DATA	---	---	---	* ---	---
ORLD P,A	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	OUTPUT OPCODE/ADDRESS	INCREMENT TIMER	OUTPUT DATA	---	---	---	* ---	---
J(CONDITIONAL)	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	SAMPLE CONDITION	*INCREMENT SAMPLE	---	FETCH IMMEDIATE DATA	---	UPDATE PROGRAM COUNTER	* ---	---
STRT T STRT CNT	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	* ---	START COUNTER					
STOP TCNT	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	* ---	START COUNTER					
EN I	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	* ENABLE INTERRUPT	---					
DIS I	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER	---	* DISABLE INTERRUPT	---					
ENT0 CLK	FETCH INSTRUCTION	INCREMENT PROGRAM COUNTER		* ENABLE CLOCK	---					

* VALID INSTRUCTION ADDRESSES ARE OUTPUT AT THIS TIME IF EXTERNAL PROGRAM MEMORY IS BEING ACCESSED.

(1) IN LATER MCS-48 DEVICES T1 IS SAMPLED IN S4.

Figure 14. 8048AH/8049AH Instruction Timing Diagram

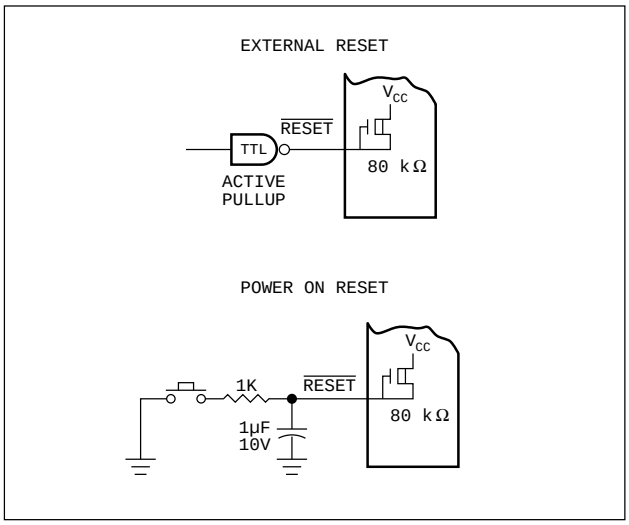


Figure 15.

TIMING

The 8048AH operates in a single-step mode as follows:

1) The processor is requested to stop by applying a low level on SS'.

2) The processor responds by stopping during the address fetch portion of the next instruction. If a double cycle instruction is in progress when the single step command is received, both cycles will be completed before stopping.

3) The processor acknowledges it has entered the stopped state by raising ALE high. In this state (which can be maintained indefinitely) the address of the next instruction to be fetched is present on BUS and the lower half of port 2.

4) SS' is then raised high to bring the processor out of the stopped mode allowing it to fetch the next instruction. The exit from stop is indicated by the processor brining ALE low.

5) To stop the processor at the next instruction, SS' must be brought low again soon after ALE goes low. If SS' is left high the processor remains in a "Run" mode.

A diagram for implementing the single-step function of the 8748Ah is show in Figure 16. D-type flip-flop with preset and clear is used to generate SS'. In the run mode SS' is held high by keeping the flip-flop preset (preset has precedence over the clear input). To enter single step, preset is removed allowing ALE to bring SS' low via the clear input. ALE should be buffered since the clear input of an SN7474 is the equivalent of 3 TTL loads. The processor is now in the stopped state. The next instruction is initiated by clocking a "1" into the flip-flop. This "1" will not appear on SS' unless ALE is high removing clear from the flip-flop. In response to SS' going high the processor begins an instruction fetch which brings ALE low resetting SS' through the clear input and causing the processor to again enter the stopped state.

2.14 Power Down Mode (8048AH, 8049AH, 8050AH, 8039AHL, 8035AHL, 8040AHL)

Extra circuitry has been added to the 8048AH/8049AH/8050AH ROM version to allow power to be removed from all but the data RAM array for low power standby operation. In the power down mode the contents of data RAM can be maintained while drawing typically 10% to 15% of normal operating power requirements.

V_{CC} serves as the 5V supply pin for the bulk of circuitry while the V_{DD} pin supplies only the RAM array. In normal operation both pins are at 5V while in standby, V_{CC} is at ground and V_{DD} is maintained at its standard value. Applying Reset to the processor through the RESET' pin inhibits any access to the RAM by the processor and guarantees the RAM cannot be inadvertently altered as power is removed from V_{CC}.

A typical power down sequence (Figure 17) occurs as follows:

1) Imminent power supply failure is detected by user defined circuitry. Signal must be early enough to allow 8048AH to save all necessary data before V_{CC} falls below normal operating limits.

2) Power fail signal is used to interrupt processor and vector it to a power fail service routine.

3) Power fail routine saves all important data and machine status in the internal data RAM array. Routine may also initiate transfer of backup supply to the V_{DD} pin and indicate to external circuitry that power fail routine is complete.

4) Reset is applied to guarantee data will not be altered as the power supply falls out of limits. Reset must be held low until V_{CC} is at ground level.

Recovery from the Power Down mode can occur as any other power-on sequence with an external capacitor on the Reset input providing the necessary delay. See the previous section on Reset.

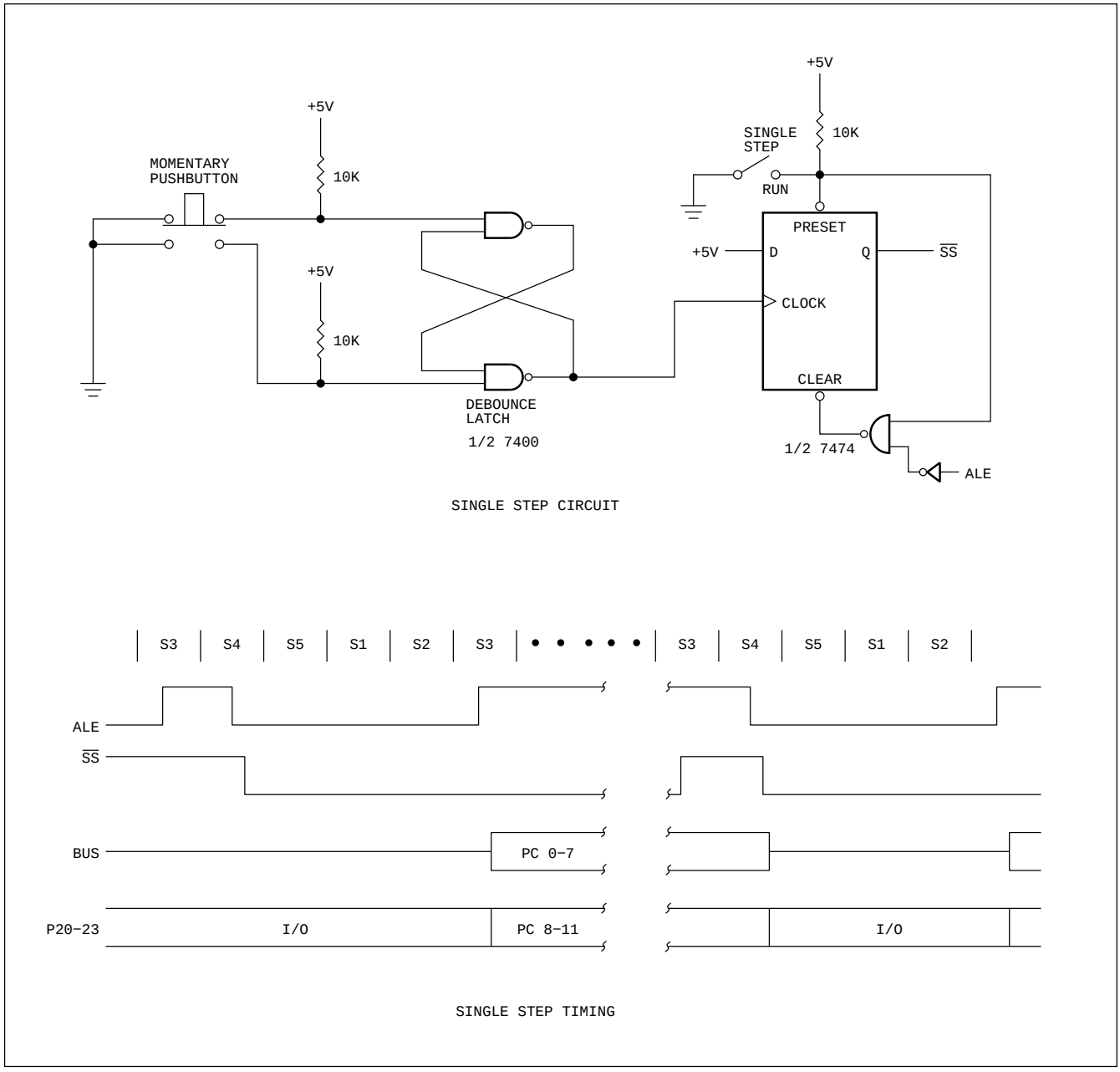


Figure 16. Single Step Operation

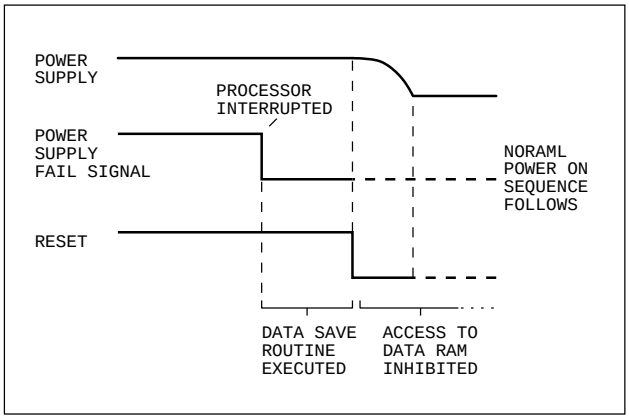


Figure 17. Power Down Sequence

2.15 External Access Mode

Normally, the first 1K (8048AH), 2K (8049AH) or 4K (8050AH) words of program memory are automatically fetched from internal ROM or EPROM. The EA input pin however allows the user to efficiently disable internal program memory by forcing all program memory fetches to reference external memory. The following chapter explains how access to external program memory is accomplished.

The External Access mode is very useful in system test and debug because it allows the user to disable his internal applications program and substitute an external program of his choice --- a diagnostic routine for instance. In addition, the data sheet shows how internal program memory can be read externally, independent of the processor. A "1" level on EA initiates the external access mode. For proper operation, Reset should be applied while the EA input is changed.

2.16 Sync Mode

The 8048AH, 8049AH, 8050AH has incorporated a new SYNC mode. The Sync mode is provided to ease the design of multiple controller circuits by allowing the designer to force the device into known phase and state time. The SYNC mode may also be utilized by automatic test equipment (ATE) for quick, easy and efficient synchronizing between the tester and the DUT (device under test).

SYNC mode is enabled when SS' pin is raised to high voltage level of +12 volts. To begin synchronization, T0 is raised to 5 volts at least four clock cycles after SS'. T0 must be high for at least four X1 clock cycles to fully reset the prescaler and time state generators. T0 may then be brought down with the rising edge of X1. Two clock cycles later, with the rising edge of X1, the device enters into Time State 1, Phase 1. SS' is then brought down to 5 volts 4 clocks later after T0. RESET' is allowed to go high 5 t_{CY} (75 clocks) later for normal execution of code. See Figure 18.

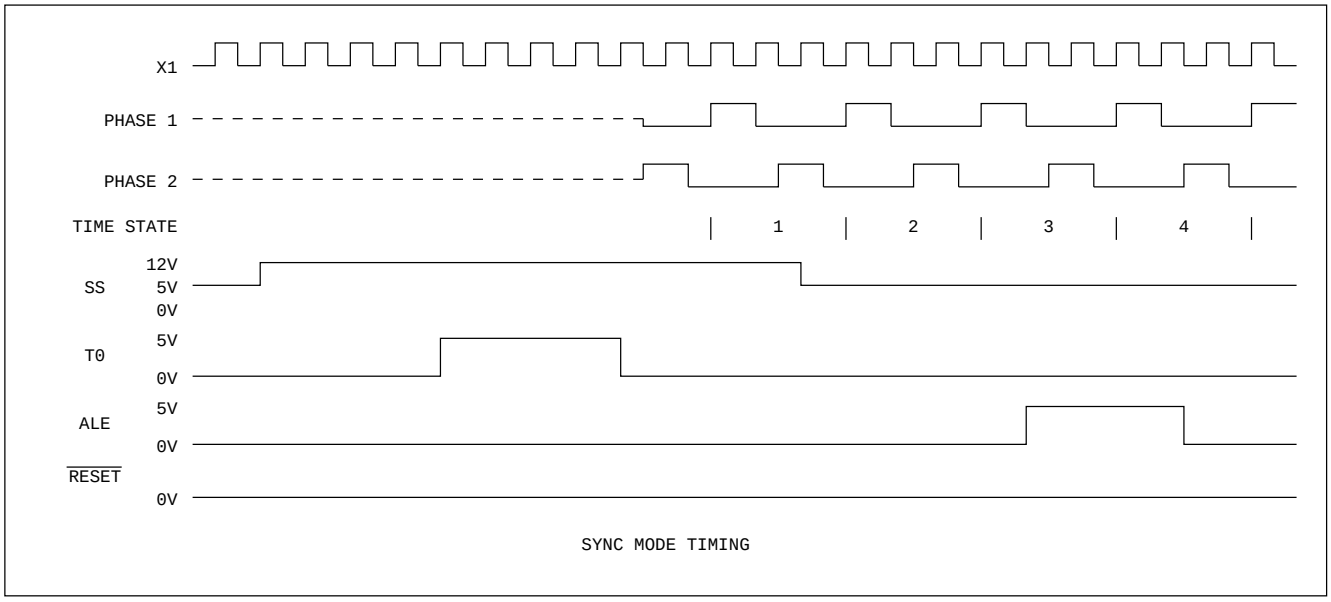


Figure 18. Sync Mode Timing

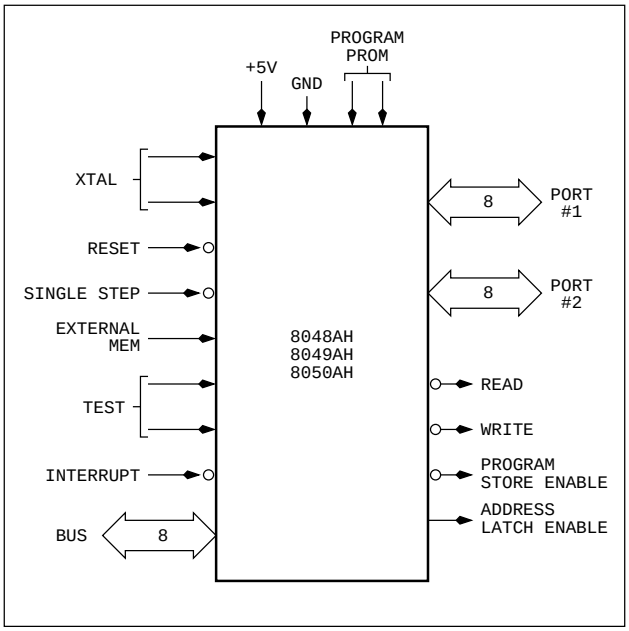


Figure 19. 8048AH and 8049AH Logic Symbol

3.0 PIN DESCRIPTION

The MCS-48 processors are packaged in 40 pin Dual In-Line Packages (DIP's). Table 3 is a summary of the functions of each pin. Figure 19 is the logic symbol for the 8048AH product family. Where it exists, the second paragraph describes each pin's function in an expanded MCS-48 system. Unless otherwise specified, each input is TTL compatible and each output will drive one standard TTL load.

Table 3. Pin Description

Designation	Pin Number*	Function
V_{SS}	20	Circuit GND potential
V_{DD}	26	Programming power supply; 21V during program for the 8748H/8749H; +5V during operation for both ROM and EPROM. Low power standby pin in 8048AH and 8049AH/8050AH ROM versions.
V_{CC}	40	Main power supply; +5V during operation and during 8748H and 8749H programming.
PROG	25	Program pulse; +18V input pin during 8748H/8749H programming. Output strobe for 8243 I/O expander.
P10-P17 (Port 1)	27-34	8-bit quasi-bidirectional port. (Internal Pullup ≈ 50 kΩ)
P20-27 (Port 2)	21-24 35-38	8-bit quasi-bidirectional port. (Internal Pullup ≈ 50 kΩ) P20-23 contain the four high order program counter bits during an external program memory fetch and serve as a 4-bit O/O expander bus for 8243.
D0-D7 (BUS)	12-19	True bidirectional port which can be written or read synchronously using the RD', WR' strobes. The port can also be statically latched. Contains the 8 low order program counter bits during an external program memory fetch, and receives the addressed instruction under the control of PSEN'. Also contains the address and data during an external RAM data store instruction, under control of ALE, RD' and WR'.
T0	1	Input pin testable using the conditional transfer instructions JT0 and JNT0. T0 can be designated as a clock output using ENT0 CLK instruction. T0 is also used during programming and sync mode.
T1	39	Input pin testable using the JT1 and JNT1 instructions. Can be designated the event counter input using the STRT CNT instruction. (See Section 2.10).
INT'	6	Interrupt input. Initiates an interrupt if interrupt is enabled. Interrupt is disabled after a reset. (Active low) Interrupt must remain low for at least 3 machine cycles to ensure proper operation.
RD'	8	Output strobe activated during a BUS read. Can be used to enable data onto the BUS from external device. (Active low)
RESET'	4	Input which is used to initialize the processor. Also used during EPROM programming and verification. (Active low) (Internal pullup ≈ 50 kΩ)
WR'	10	Output strobe during a BUS write. (Active low) Used as a write strobe to external data memory.
ALE	11	Address Latch Enable. This signal occurs once during each cycle and is useful as a clock output. The negative edge of ALE strobes address into external data and program memory.
PSEN'	9	Program Store Enable. This output occurs only during a fetch to external program memory. (Active low)
SS'	5	Single step input can be used in conjunction with ALE to "single step" the processor through each instruction. (Active low) (Internal Pullup ≈ 300 kΩ) +12V for sync modes (See Section 2.16).

Table 3. Pin Description (Continued)

Designation	Pin Number*	Function
EA	7	External Access input which forces all program memory fetches to reference external memory. Useful for emulation and debug and essential for testing and program verification. (Active high) +12V for 8048AH/8049AH/8050AH program verification and +18V for 8748H/8749H program verification (Internal Pullup ≈ 10 MΩ on 8048AH/8049AH/ 8035AHL/8039AHL/8050AH/8040AHL)
XTAL1	2	One side of crystal input for internal oscillator. Also input for external source.
XTAL2	3	Other side of crystal/external source input.

*Unless otherwise stated, inputs do not have internal pullup resistors. 8048AH, 8748H, 8049AH, 8050AH, 8040AHL

4.0 PROGRAMMING, VERIFYING AND ERASING EPROM

The internal Program Memory of the 8748H and the 8749H may be erased and reprogrammed by the user as explained in the following sections. See also the 8748H and 8749H data sheets.

4.1 Programming/Verification

In brief, the programming process consists of: applying the program mode, applying an address, latching the address, applying data and applying a programming pulse. This programming algorithm applies to both the 8748H and 8749H. Each word is programmed completely before moving on to the next and is followed by a verification step. The following is a list of the pins used for programming and a description of their functions:

Pin	Function
XTAL1	Clock Input (3 to 4 MHz)
Reset	Initialization and Address Latching
Test 0	Selection of Program (0 V) or Verify (5 V) Mode
EA	Activation of Program/Verify Modes
BUS	Address and Data Input. Data Output During Verify
P20-1	Address Input for 8748H
P20-2	Address Input for 8749H
V_{DD}	Programming Power Supply
PROG	Program Pulse Input
P10-P11	Tied to ground (8749H only)

4.1.1 8748H and 8749H ERASURE CHARACTERISTICS

The erasure characteristics of the 8748H and 8749H are such that erasure begins to occur when exposed to light with wavelengths shorter than approximately 4000 Angstroms (Å). It should be noted that the sunlight and certain types of fluorescent lamps have wavelengths in the 3000-4000 Å range. Data show that constant exposure to room level fluorescent lighting could erase the typical 8748H and 8749H in approximately 3 years while it would take approximately 1 week to cause erasure when exposed to direct sunlight. If the 8748H or 8749H is to be exposed to these types of lighting conditions for extended periods of time, opaque labels should be placed over the 8748H window to prevent unintentional erasure.

When erased, bits of the 8748H and 8749H Program Memory are in the logic "0" state.

The recommended erasure procedure for the 8748H and 8749H is exposure to shortwave ultraviolet light which has a wavelength of 2537 Angstroms (Å). The integrated dose (i. e., UV intensity × exposure time) for erasure should be a minimum of 15 Wsec/cm^2 power rating. The 8748H and 8749H should be placed within one inch from the lamp tubes during erasure. Some lamps have a filter in their tubes and this filter should be removed before erasure.

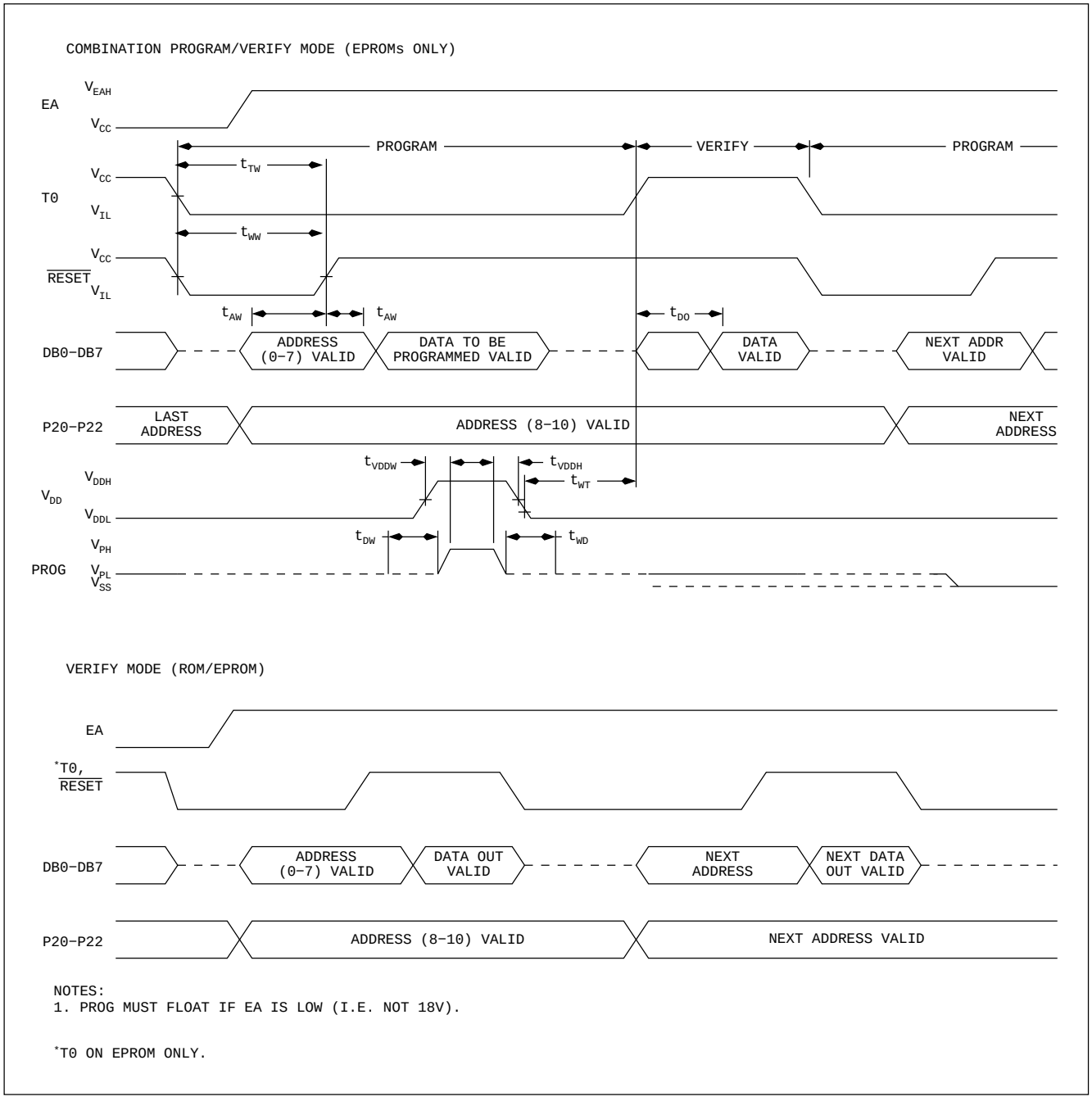


Figure 20. Program/Verify Sequence for 8749H/8748H

Table 4. A.C. Specification for Programming 8748H/8749H

Symbol	Parameter	Min	Max	Unit	Test Conditions
t_{AW}	Address Setup Time to RESET' ↑	$4t_{CY}$			
t_{WA}	Address Hold Time to RESET' ↑	$4t_{CY}$			
t_{DW}	Data in Setup Time to PROG ↑	$4t_{CY}$			
t_{WD}	Data in Hold Time after PROG ↓	$4t_{CY}$			
t_{PH}	RESET' Hold Time to Verify	$4t_{CY}$			
t_{VDDW}	V_{DD} Hold Time before PROG ↑	0	1.0	ms	
t_{VDDH}	V_{DD} Hold Time after PROG ↓	0	1.0	ms	
t_{PW}	Program Pulse Width	50	60	ms	
t_{TW}	TEST 0 Setup Time for Program Mode	$4t_{CY}$			
t_{WT}	TEST 0 Hold Time after Program Mode	$4t_{CY}$			
t_{DO}	TEST 0 to data Out Delay		$4t_{CY}$		
t_{WW}	RESET' Pulse Width to Latch Address	$4t_{CY}$			
t_r, t_f	V_{DD} and PROG Rise and Fall Times	0.5	100	µs	
t_{CY}	CPU Operation Cycle Time	3.75	5	µs	
t_{RE}	RESET' Setup Time before EA ↑	$4t_{CY}$			

T_A = 25 °C; V_{CC} = 5 V ±5%; V_{DD} = 21 V ±5%

NOTE:
If TEST 0 is high, t_{DO} can be triggered by RESET' ↑.

Table 5. D.C. Specification for Programming 8748H/8749H

Symbol	Parameter	Min	Max	Unit	Test Conditions
V_{DDH}	V_{DD} Program Voltage High Level	20.5	21.5	V	
V_{DDL}	V_{DD} Voltage Low Level	4.75	5.25	V	
V_{PH}	PROG Program Voltage High Level	17.5	18.5	V	
V_{PL}	PROG Voltage Low Level	4.0	V_{CC}	V	
V_{EAH}	EA Program or Verify Voltage High Level	17.5	18.5	V	
I_{DD}	V_{DD} High Voltage Supply Current		20.0	mA	
I_{PROG}	PROG High Voltage Supply Current		1.0	mA	
I_{EA}	EA High Voltage Supply Current		1.0	mA	

T_A = 25 °C; V_{CC} = 5 V ±5%; V_{DD} = 21 V ±5%

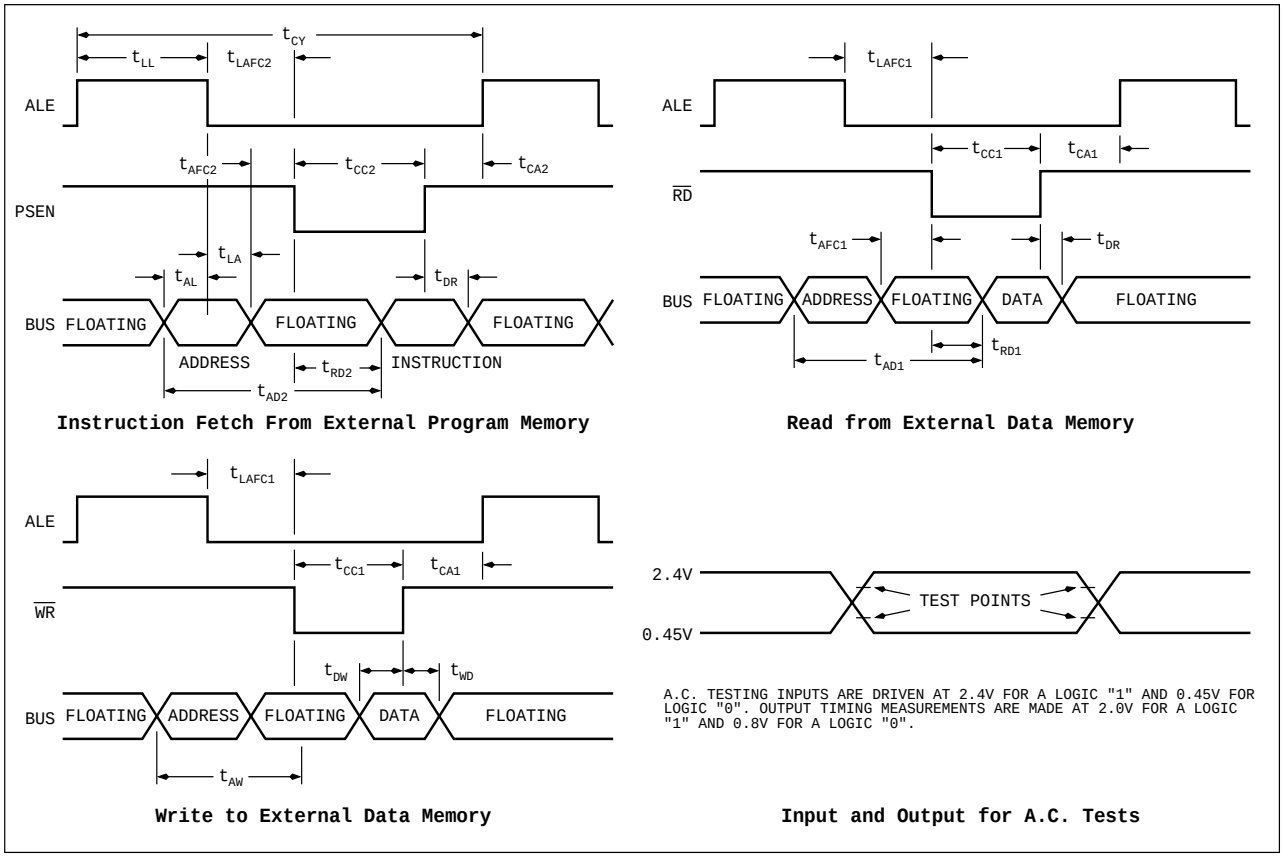


Figure 21. External Access Timing Waveforms

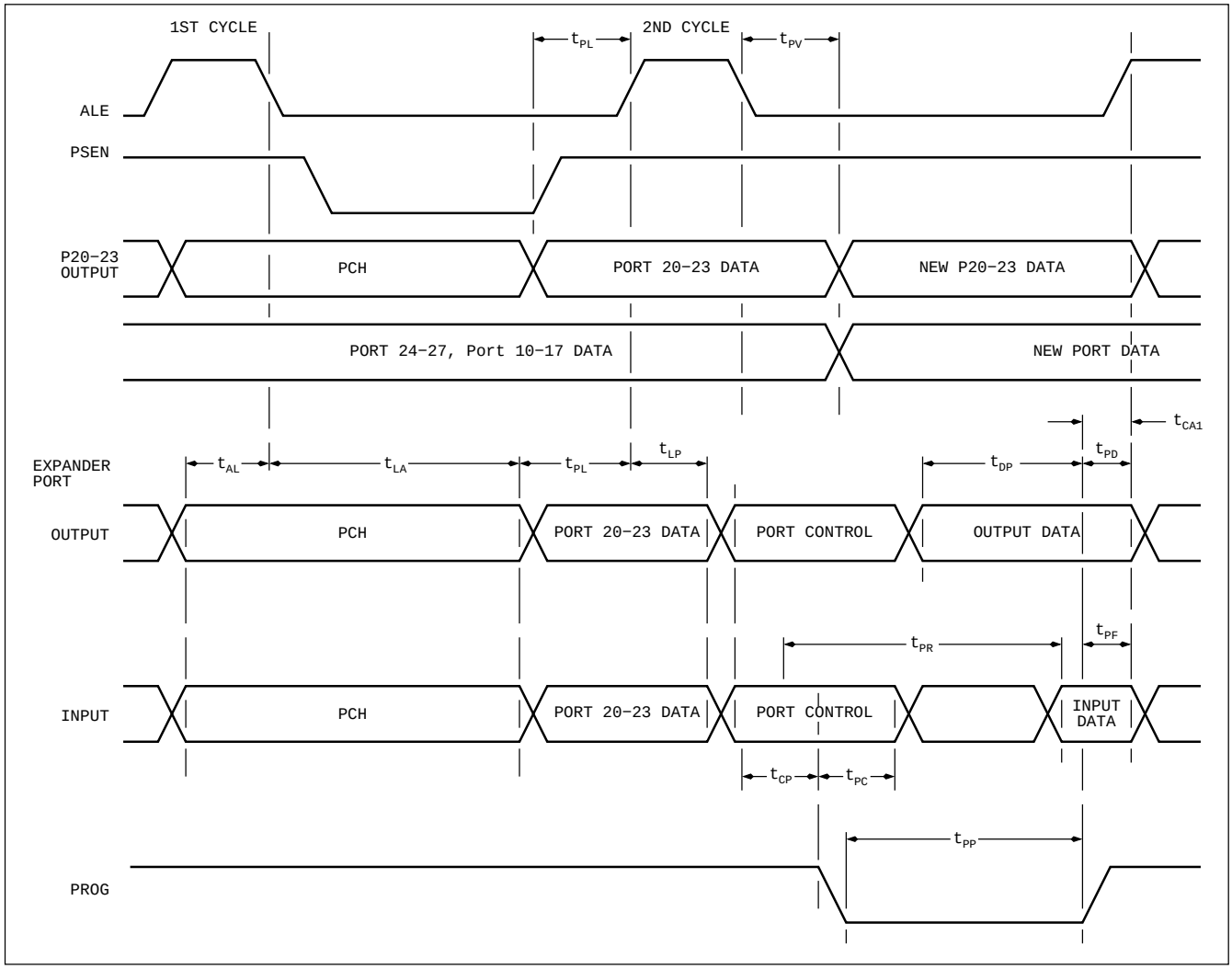


Figure 22. Port 1 / Port 2 Timing Waveforms

A.C. CHARACTERISTICS (T_A = 0 °C to 70 °C; V_{CC} = V_{DD} = 5 V ± 10%; V_{SS} = 0 V)

Symbol	Parameter	f (t_{CY}) (Note 4)	11 MHz		Unit	Conditions (Note 1)
			Min	Max		
t_{CY}	Cycle Time	15/F(XTAL)	1.36	15.0	us	(Note 3)
t_{LL}	ALE Pulse Width	7/30 t_{CY} - 170	150		ns	
t_{AL}	Addr Setup to ALE	2/15 t_{CY} - 110	70		ns	(Note 2)
t_{LA}	Addr Hold from ALE	1/15 t_{CY} - 40	50		ns	
t_{CC1}	Control Pulse Width (RD', WR')	1/2 t_{CY} - 200	480		ns	
t_{CC2}	Control Pulse Width (PSEN')	2/5 t_{CY} - 200	350		ns	
t_{DW}	Data Setup before WR'	13/30 t_{CY} - 200	390		ns	
t_{WD}	Data Hold after WR'	1/15 t_{CY} - 50	40		ns	
t_{DR}	Data Hold (RD', PSEN')	1/10 t_{CY} - 30	0	110	ns	
t_{RD1}	RD' to Data In	11/30 t_{CY} - 170		330	ns	
t_{RD2}	PSEN' to Data In	4/15 t_{CY} - 170		190	ns	
t_{AW}	Addr Setup to WR'	1/3 t_{CY} - 150	300		ns	
t_{AD1}	Addr Setup to Data (RD')	7/10 t_{CY} - 220		730	ns	
t_{AD2}	Addr Setup to Data (PSEN')	1/2 t_{CY} - 220		460	ns	
t_{AFC1}	Addr Float to RD', WR'	2/15 t_{CY} - 40	140		ns	(Note 2)
t_{AFC2}	Addr Float to PSEN'	1/30 t_{CY} - 40	10		ns	(Note 2)
t_{LAFC1}	ALE to Control (RD', WR')	1/5 t_{CY} - 75	200		ns	
t_{LAFC2}	ALE to Control (PSEN')	1/10 t_{CY} - 75	60		ns	
t_{CA1}	Control to ALE (RD', WR', PROG')	1/15 t_{CY} - 40	50		ns	
t_{CA2}	Control to ALE (PSEN')	4/15 t_{CY} - 40	320		ns	
t_{CP}	Port Control Setup to PROG'	2/15 t_{CY} - 80	100		ns	
t_{PC}	Port Control Hold to PROG'	4/15 t_{CY} - 200	160		ns	
t_{PR}	PROG' to P2 Input Valid	17/30 t_{CY} - 120		650	ns	
t_{PF}	Input Data Hold from PROG'	1/10 t_{CY}	0	140	ns	
t_{DP}	Output Data Setup	2/5 t_{CY} - 150	400		ns	
t_{PD}	Output Data Hold	1/10 t_{CY} - 50	90		ns	
t_{PP}	PROG' Pulse Width	7/10 t_{CY} - 250	700		ns	
t_{PL}	Port 2 I/O Setup to ALE	4/15 t_{CY} - 200	160		ns	
t_{LP}	Port 2 I/O Hold to ALE	1/30 t_{CY} - 30	15		ns	
t_{PV}	Port Output from ALE	3/10 t_{CY} + 100		510	ns	
t_{0PRR}	T0 Rep Rate	3/15 t_{CY}	270		ns	

Notes:

1. Control output CL = 80 pF
BUS Outputs CL = 150 pF

2. BUS High Impedance
Load 20 pF

3. f(t_{CY}) assumes 50% duty cycle on X1 and X2.

Mnemonic	Function	Description	Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Cycles	Bytes	Flags C	AC	F0	F1
		Data Moves														
MOV A, = data	(A) ← data	Move immediate the specified data into the Accumulator.	0 d_7	0 d_6	1 d_5	0 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2				
MOV A, Rr	(A) ← (Rr); r = 0 – 7	Move the contents of the designated register into the Accumulator.	1	1	1	1	1	r	r	r	1	1				
MOV A, @ Rr	(A) ← ((Rr)); r = 0 – 1	Move indirect the contents of data memory location into the Accumulator.	1	1	1	1	0	0	0	r	1	1				
MOV A, PSW	(A) ← (PSW)	Move contents of the Program Status Word into the Accumulator.	1	1	0	0	0	1	1	1	1	1				
MOV Rr, = data	(Rr) ← data; r = 0 – 7	Move immediate the specified data into the designated register.	1 d_7	0 d_6	1 d_5	1 d_4	1 d_3	r d_2	r d_1	r d_0	2	2				
MOV Rr, A	(Rr) ← (A); r = 0 – 7	Move Accumulator contents into the designated register.	1	0	1	0	1	r	r	r	1	1				
MOV @ Rr, A	((Rr)) ← (A); r = 0 – 1	Move indirect Accumulator contents into data memory location.	1	0	1	0	0	0	0	r	1	1				
MOV @ Rr, = data	((Rr)) ← data; r = 0 – 1	Move immediate the specified data into data memory.	1 d_7	0 d_6	1 d_5	1 d_4	0 d_3	0 d_2	0 d_1	r d_0	2	2				
MOV PSW, A	(PSW) ← (A)	Move contents of Accumulator into the Program Status Word.	1	1	0	1	0	1	1	1	1	1				
MOVP A, @ A	(PC 0 – 7) ← (A) (A) ← ((PC))	Move data in the current page into the Accumulator.	1	0	1	0	0	0	1	1	2	1				
MOVP3 A, @ A	(PC 0 – 7) ← (A) (PC 8 – 10) ← 011 (A) ← ((PC))	Move Program data in Page 3 into the Accumulator.	1	1	1	0	0	0	1	1	2	1				
MOVX A, @ Rr	(A) ← ((Rr)); r = 0 – 1	Move indirect the contents of external data memory into the Accumulator.	1	0	0	0	0	0	0	r	2	1				
MOVX @ Rr, A	(Rr) ← ((A)); r = 0 – 1	Move indirect the contents of the Accumulator into external data memory.	1	0	0	1	0	0	0	r	2	1				
XCH A, Rr	(A) ⇄ (Rr); r = 0 – 7	Exchange the Accumulator and designated register's contents.	0	0	1	0	1	r	r	r	1	1				
XCH A, @ Rr	(A) ⇄ ((Rr)); r = 0 – 1	Exchange indirect contents of Accumulator and location in data memory.	0	0	1	0	0	0	0	r	1	1				
XCHD A, @ Rr	(A 0 – 3) ⇄ ((Rr)(0 – 3)) r = 0 – 1	Exchange indirect b-bit contents of Accumulator and data memory.	0	0	1	1	0	0	0	r	1	1				
		Flags														
CPL C	(C) ← NOT (C)	Complement content of carry bit.	1	0	1	0	0	1	1	1	1	1	•			
CPL F0	(F0) ← NOT (F0)	Complement content of Flag F0.	1	0	0	1	0	1	0	1	1	1			•	
CPL F1	(F1) ← NOT (F1)	Complement content of Flag F1.	1	0	1	1	0	1	0	1	1	1				•
CLR C	(C) ← 0	Clear content of carry bit to 0.	1	0	0	1	0	1	1	1	1	1	•			
CLR F0	(F0) ← 0	Clear content of Flag 0 to 0.	1	0	0	0	0	1	0	1	1	1			•	
CLR F1	(F1) ← 0	Clear content of Flag 1 to 0.	1	0	1	0	0	1	0	1	1	1				•

Mnemonic	Function	Description	Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Cycles	Bytes	Flags C	AC	F0	F1
		Input/Output														
ANL BUS, = data	(BUS) ← (BUS) AND data	Logical and immediate specified data with contents on BUS.	1 d_7	0 d_6	0 d_5	1 d_4	1 d_3	0 d_2	0 d_1	0 d_0	2	2				
ANL Pp, = data	(Pp) ← (Pp) AND data p = 1 – 2	Logical and immediate specified data with designated port (1 – 2).	1 d_7	0 d_6	0 d_5	1 d_4	1 d_3	0 d_2	p d_1	p d_0	2	2				
ANLD Pp, A	(Pp) ← (Pp) AND (A 0 – 3) p = 4 – 7	Logical and contents of Accumulator with designated port (4 – 7).	1	0	0	1	1	1	p	p	2	1				
IN A, Pp	(A) ← (Pp); p = 1 – 2	Input data from designated port (1 – 2) into Accumulator.	0	0	0	0	1	0	p	p	2	1				
INS A, BUS	(A) ← (BUS)	Input strobed BUS data into Accumulator.	0	0	0	0	1	0	0	0	2	1				
MOVD A ,Pp	(A 0 – 3) ← (Pp); p = 4 – 7 (A 4 – 7) ← 0	Move contents of designated port (4 – 7) into Accumulator.	0	0	0	0	1	1	p	p	2	1				
MOVD Pp, A	(Pp) ← A (0 – 3); p = 4 – 7	Move contents of Accumulator to designated port (4 – 7).	0	0	1	1	1	1	p	p	2	1				
ORL BUS, = data	(BUS) ← (BUS) OR data	Logical or immediate specified data with contents of BUS.	1 d_7	0 d_6	0 d_5	0 d_4	1 d_3	0 d_2	0 d_1	0 d_0	2	2				
ORLD Pp, A	(Pp) ← (Pp) OR (A 0 – 3) p = 4 – 7	Logical or contents of Accumulator with designated port (4 – 7).	1	0	0	0	1	1	p	p	1	1				
ORL Pp, = data	(Pp) ← (Pp) OR data p = 1 – 2	Logical or immediate specified data with designated port (1 – 2).	1 d_7	0 d_6	0 d_5	0 d_4	1 d_3	0 d_2	p d_1	p d_0	2	2				
OUTL BUS, A ⑤	(BUS) ← A	Output contents of Accumulator onto BUS.	0	0	0	0	0	0	1	0	2	1				
OUTL Pp, A	(Pp) ← (A); p = 1 – 2	Output contents of Accumulator to designated port (1 – 2).	0	0	1	1	1	0	p	p	1	1				
		Registers														
DEC Rr	(Rr) ← (Rr) – 1; r = 0 – 7	Decrement by 1 contents of designated register.	1	1	0	0	1	r	r	r	1	1				
INC Rr	(Rr) ← (Rr) + 1; r = 0 – 7	Increment by 1 contents of designated register.	0	0	0	1	1	r	r	r	1	1				
INC @ Rr	((Rr)) ← ((Rr)) + 1; r = 0 – 1	Increment indirect by 1 the contents of data memory location.	0	0	0	1	0	0	0	r	1	1				
		Subroutine														
CALL addr	((SP)) ← (PC), (PSW 4 – 7) (SP) ← (SP) + 1 (PC 8 – 10) ← addr 8 – 10 (PC 0 – 7) ← addr 0 – 7 (PC 11) ← DBF	Call designated Subroutine.	a_{10} a_7	a_9 a_6	a_8 a_5	1 a_4	0 a_3	1 a_2	0 a_1	0 a_0	2	2				
RET	(SP) ← (SP) – 1 (PC) ← ((SP))	Return from Subroutine without restoring Program Status Word.	1	0	0	0	0	0	1	1	2	1				
RETR	(SP) ← (SP) – 1 (PC) ← ((SP)) (PSW 4 – 7) ← ((SP))	Return from Subroutine restoring Program Status Word.	1	0	0	1	0	0	1	1	2	1				
		Timer/Counter														
EN TCNTI		Enable internal interrupt Flag for Timer/Counter output.	0	0	1	0	0	1	0	1	1	1				
DIS TCNTI		Disable internal interrupt Flag for Timer Counter output.	0	0	1	1	0	1	0	1	1	1				
MOV A, T	(A) ← (T)	Move contents of Timer/Counter into Accumulator.	0	1	0	0	0	0	1	0	1	1				
MOV T, A	(T) ← (A)	Move contents of Accumulator into Timer/Counter.	0	1	1	0	0	0	1	0	1	1				
STOP TCNT		Stop Count for Event Counter.	0	1	1	0	0	1	0	1	1	1				
STRT CNT		Start Count for Event Counter.	0	1	0	0	0	1	0	1	1	1				
STRT T		Start Count for Timer.	0	1	0	1	0	1	0	1	1	1				

Mnemonic	Function	Description	Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Cycles	Bytes	Flags C	AC	F0	F1
		Miscellaneous														
NOP		No Operation performed.	0	0	0	0	0	0	0	0	1	1				
		Accumulator														
ADD A, = data	(A) ← (A) + data	Add immediate the specified Data to the Accumulator.	0 d_7	0 d_6	0 d_5	0 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2	•	•		
ADD A, Rr	(A) ← (A) + (Rr) r = 0 – 7	Add contents of designated register to the Accumulator.	0	1	1	0	1	r	r	r	1	1	•	•		
ADD A, @ Rr	(A) ← (A) + ((Rr)) r = 0 – 1	Add indirect the contents of the data memory location to the Accumulator.	0	1	1	0	0	0	0	r	1	1	•	•		
ADDC A, = data	(A) ← (A) + (C) + data	Add immediate with carry the specified data to the Accumulator.	0 d_7	0 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2	•	•		
ADDC A, Rr	(A) ← (A) + (C) + (Rr) r = 0 – 7	Add with carry the contents of the designated register to the Accumulator.	0	1	1	1	1	r	r	r	1	1	•	•		
ADDC A, @ Rr	(A) ← (A) + (C) + ((Rr)) r = 0 – 1	Add indirect with carry the contents of data memory location to the Accumulator.	0	1	1	1	0	0	0	r	1	1	•	•		
ANL A, = data	(A) ← (A) AND data	Logical and specified immediate data with Accumulator.	0 d_7	1 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2				
ANL A, Rr	(A) ← (A) AND (Rr) r = 0 – 7	Logical and contents of designated register with Accumulator.	0	1	0	1	1	r	r	r	1	1				
ANL A, @ Rr	(A) ← (A) AND ((Rr)) r = 0 – 1	Logical and indirect the contents of data memory with Accumulator.	0	1	0	1	0	0	0	r	1	1				
CPL A	(A) ← NOT (A)	Complement the contents of the Accumulator.	0	0	1	1	0	1	1	1	1	1				
CLR A	(A) ← 0	CLEAR the contents of the Accumulator.	0	0	1	0	0	1	1	1	1	1				
DA A		DECIMAL ADJUST the contents of the Accumulator.	0	1	0	1	0	1	1	1	1	1	•			
DEC A	(A) ← (A) – 1	Decrement by 1 the Accumulator's contents.	0	0	0	0	0	1	1	1	1	1				
INC A	(A) ← (A) + 1	Increment by 1 the Accumulator's contents.	0	0	0	1	0	1	1	1	1	1				
ORL A, = data	(A) ← (A) OR data	Logical or specified immediate data with Accumulator.	0 d_7	1 d_6	0 d_5	0 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2				
ORL A, Rr	(A) ← (A) OR (Rr) r = 0 – 7	Logical or contents of designated register with Accumulator.	0	1	0	0	1	r	r	r	1	1				
ORL A, @ Rr	(A) ← (A) OR ((Rr)) r = 0 – 1	Logical or indirect the contents of data memory with Accumulator.	0	1	0	0	0	0	0	r	1	1				
RL A	(AN + 1) ← (AN); N = 0 – 6 (A0) ← (A7)	Rotate Accumulator left by 1-bit without carry.	1	1	1	0	0	1	1	1	1	1				
RLC A	(AN + 1) ← (AN); N = 0 – 6 (A0) ← (C) (C) ← (A7)	Rotate Accumulator left by 1-bit through carry.	1	1	1	1	0	1	1	1	1	1	•			
RR A	(AN) ← (AN + 1); N = 0 – 6 (A7) ← (A0)	Rotate Accumulator right by 1-bit without carry.	0	1	1	1	0	1	1	1	1	1				
RRC A	(AN) ← (AN + 1); N = 0 – 6 (A7) ← (C) (C) ← (A0)	Rotate Accumulator right by 1-bit through carry.	0	1	1	0	0	1	1	1	1	1	•			
SWAP A	(A4 – 7) ⇄ (A0 – 3)	Swap the 2 4-bit nibbles in the Accumulator.	0	1	0	0	0	1	1	1	1	1				
XRL A, = data	(A) ← (A) XOR data	Logical xor specified immediate data with Accumulator.	1 d_7	1 d_6	0 d_5	1 d_4	0 d_3	0 d_2	1 d_1	1 d_0	2	2				
XRL A, Rr	(A) ← (A) XOR (Rr) r = 0 – 7	Logical xor contents of designated register with Accumulator.	1	1	0	1	1	r	r	r	1	1				
XRL A, @ Rr	(A) ← (A) XOR ((Rr)) r = 0 – 1	Logical xor indirect the contents of data memory with Accumulator.	1	1	0	1	0	0	0	r	1	1				

Mnemonic	Function	Description	Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Cycles	Bytes	Flags C	AC	F0	F1
		Branch														
DJNZ Rr, addr	(Rr) ← (Rr) – 1; r = 0 – 7 if (Rr) ≠ 0: (PcC0 – 7) ← addr	Decrement the specified register and test contents.	1 a_7	1 a_6	1 a_5	0 a_4	1 a_3	r a_2	r a_1	r a_0	2	2				
JBb addr	(PC 0 – 7) ← addr if Bb = 1 (PC) ← (PC) + 2 if Bb = 0	Jump to specified address if Accumulator bit is set.	b_2 a_7	b_1 a_6	b_0 a_5	1 a_4	0 a_3	0 a_2	1 a_1	0 a_0	2	2				
JC addr	(PC 0 – 7) ← addr if C = 1 (PC) ← (PC) + 2 if C = 0	Jump to specified address if carry flag is set.	1 a_7	1 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JF0 addr	(PC 0 – 7) ← addr if F0 = 1 (PC) ← (PC) + 2 if F0 = 0	Jump to specified address if Flag F0 is set.	1 a_7	0 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JF1 addr	(PC 0 – 7) ← addr if F01= 1 (PC) ← (PC) + 2 if F1 = 0	Jump to specified address if Flag F1 is set.	0 a_7	1 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JMP addr	(PC 8 – 10) ← addr 8 – 10 (PC 0 -7) ← add 0 – 7 (PC 11) ← DBF	Direct Jump to specified address within the 2K address block.	a_{10} a_7	a_9 a_6	a_8 a_5	0 a_4	0 a_3	1 a_2	0 a_1	0 a_0	2	2				
JMPP @ A	(PC 0 – 7) ← ((A))	Jump indirect to specified address within address page.	1	0	1	1	0	0	1	1	2	1				
JNC addr	(PC 0 – 7) ← addr if C = 0 (PC) ← (PC) if C = 1	Jump to specified address if carry flag is low.	1 a_7	1 a_6	1 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JNI addr	(PC 0 – 7) ← addr if I = 0 (PC) ← (PC) if I = 1	Jump to specified address if interrupt is low.	1 a_7	0 a_6	0 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JNT0 addr	(PC 0 – 7) ← addr if T0 = 0 (PC) ← (PC) if T0 = 1	Jump to specified address if Test 0 is low.	0 a_7	0 a_6	1 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JNT1 addr	(PC 0 – 7) ← addr if T1 = 0 (PC) ← (PC) if T1 = 1	Jump to specified address if Test 1 is low.	0 a_7	1 a_6	0 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JNZ addr	(PC 0 – 7) ← addr if A ≠ 0 (PC) ← (PC) if A = 0	Jump to specified address if Accumulator is non-zero.	1 a_7	0 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JTF addr	(PC 0 – 7) ← addr if TF = 1 (PC) ← (PC) if TF = 0	Jump to specified address if Timer Flag is set to 1.	0 a_7	0 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JT0 addr	(PC 0 – 7) ← addr if T0 = 1 (PC) ← (PC) if T0 = 0	Jump to specified address if Test 0 is a 1.	0 a_7	0 a_6	1 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JT1 addr	(PC 0 – 7) ← addr if T1 = 1 (PC) ← (PC) if T1 = 0	Jump to specified address if Test 1 is a 1.	0 a_7	1 a_6	0 a_5	1 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
JZ addr	(PC 0 – 7) ← addr if A = 0 (PC) ← (PC) if A ≠ 0	Jump to specified address if Accumulator is 0.	1 a_7	1 a_6	0 a_5	0 a_4	0 a_3	1 a_2	1 a_1	0 a_0	2	2				
		Control														
EN I		Enable the external Interrupt input.	0	0	0	0	0	1	0	1	1	1				
DIS I		Disable the external Interrupt input.	0	0	0	1	0	1	0	1	1	1				
ENT0 CLK		Enable the Clock Output pin T0.	0	1	1	1	0	1	0	1	1	1				
SEL MB0	(DBF) ← 0	Select Bank 0 (locations 0 – 2047) of Program Memory.	1	1	1	0	0	1	0	1	1	1				
SEL MB1	(DBF) ← 1	Select Bank 1 (locations 2048 – 4095) of Program Memory.	1	1	1	1	0	1	0	1	1	1				
SEL RB0	(BS) ← 0	Select Bank 0 (locations 0 – 7) of Data Memory.	1	1	0	0	0	1	0	1	1	1				
SEL RB1	(BS) ← 1	Select Bank 1 (locations 24 – 31) of Data Memory.	1	1	0	1	0	1	0	1	1	1				

Notes: ① Instruction Code Designators r and p form the binary representation of the Registers and Ports involved.
② The dot under the appropriate flag bit indicates that its content is subject to change by the instruction it appears in.
③ References to the address and data are specified in bytes 2 and or 1 of the instruction.
④ Numerical Subscritions appearing in the FUNCTION column reference the specific bits affected.
⑤ When the Bus is written to with an OUTL instruction, the Bus remains an Output Port until either device is reset or a MOVX instruction is executed.

5.0 INSTRUCTION SET

The MCS-48 instruction set is extensive for a machine of its size and has been tailored to be straightforward and very efficient in its use of program memory. All instructions are either one or two bytes in length and over 70% are only one byte long. Also, all instructions execute in either one or two cycles (2.5 µsec or 5.0 µsec when using a 6 MHz XTAL) and over 50% of all instructions execute in a single cycle. Double cycle instructions include all immediate instructions and all I/O instructions.

The MCS-48 microcomputers have been designed to efficiently handle arithmetic operations in both binary and BCD as well as to efficiently handle the single bit operations required in control applications. Special instructions have also been included to simplify loop counters, table lookup routines and N-way branch routines.

Data Transfers

The 8-bit accumulator is the central point for all data transfers within the 8048. Data can be transferred between the 8 registers of each working register bank and the accumulator directly, i.e. The source or destination register is specified by the instruction. The remaining locations of the internal RAM array are referred to as Data Memory and are addressed indirectly via an address stored in either R0 or R1 of the active working register bank. R0 and R1 are also used to indirectly address external data memory when it is present. Transfers to and from internal RAM require one cycle while transfers to external RAM require two. Constants stored in Program Memory can be loaded directly to the accumulator and to the 8 working registers. Data can also be transferred directly between the on-board timer/counter or the accumulator and the Program Status Word (PSW). Writing to the PSW alters machine status accordingly and provides a means of restoring status after an interrupt or of altering the stack pointer if necessary.

Accumulator Operations

Immediate data, data memory or the working registers can be added with or without carry to the accumulator. These sources can also be ANDed, ORed or Exclusive ORed to the accumulator. Data may be moved to or from the accumulator and working registers or data memory. The two values can also be exchanged in a single operation.

In addition, the lower 4 bits of the accumulator can be exchanged with the lower 4 bits of any of the internal RAM locations. This instruction, along with an instruction which swaps the upper and lower 4-bit halves of the accumulator, provides easy handling of 4-bit quantities, including BCD numbers. To facilitate BCD arithmetic, a Decimal Adjust instruction is included. This instruction is used to correct the result of the binary addition of two two-digit BCD numbers. Performing a decimal adjust on the result in the accumulator produces the required BCD result.

Finally, the accumulator can be: incremented, decremented, cleared or complemented and can be rotated left or right 1 bit at a time with or without carry.

Although there is no subtract instruction in the 8048, this operation can be easily implemented with three single-byte single-cycle instructions.

A value may be subtracted from the accumulator with the result in the accumulator by:

Complement Accumulator	CPL A
Add register to Accumulator	ADD A, Rr
Complement Accumulator	CPL A

Register Operations

The working registers can be accessed via the accumulator as explained above or can be loaded immediate with constraints from program memory. In addition, they can be incremented or decremented or used as loop counters using the decrement and skip, if not zero instruction, as explained under branch instructions.

All Data Memory including working registers can be accessed with indirect instructions via R0 and R1 and can be incremented.

Flags

There are four user accessible flags in the 8048: Carry, Auxiliary Carry, F0 and F1. Carry indicates overflow of the accumulator and Auxiliary Carry is used to indicate overflow between BCD digits and is used during decimal adjust operation. Both Carry and Auxiliary Carry are accessible as part of the Program Status word and are stored on the stack during subroutines. F0 and F1 are undedicated general purpose flags to be used as the programmer desires. Both flags can be cleared or complemented and tested by conditional jump instructions. F0 is also accessible via the Program Status word and is stored on the stack with the carry flags.

Branch Instructions

The unconditional jump instruction is two bytes and allows jumps anywhere in the first 2K words of program memory. Jumps to the second 2K of memory (4K words are directly addressable) are made by first executing a select memory bank instruction then executing the jump instruction. The 2K boundary can only be crossed via a jump or subroutine call instruction, i.e. The bank switch does not occur until a jump is executed. Once a memory bank has been

selected all subsequent jumps will be to the selected bank until another select memory bank instruction is executed. A subroutine in the opposite bank can be accessed by a select memory bank instruction followed by a call instruction. Upon completion of the subroutine execution will automatically return to the original bank; however, unless the original bank is reselected, the next jump instruction encountered will again transfer execution to the opposite bank.

Conditional jumps can test the following inputs and machine status:

T0 Input pin
T1 Input pin
INT' Input pin
Accumulator Zero
Any bit of Accumulator
Carry Flag
F0 Flag
F1 Flag

Conditional jumps allow a branch to any address within the current page (256 words) of execution. The conditions tested are the instantaneous values at the time the conditional jump is executed. For instance, the jump on accumulator zero instruction tests the accumulator itself not an intermediate zero flag.

The decrement register and skip if not zero instruction combines a decrement and a branch instruction to create an instruction very useful in implementing a loop counter. This instruction can designate any one of the 8 working registers as a counter and can effect a branch to any address within the current page of execution.

A single byte indirect jump instruction allows the program to be vectored to any one of several locations based on the contents of the accumulator. The contents of the accumulator points to a location in program memory which contains the jump address. The 8-bit jump address refers to the current page of execution. This instruction could be useful, for instance, to vector to any one of several routines based on an ASCII character which has been loaded in the accumulator. In this way ASCII key inputs can be used to initiate various routines.

Subroutines

Subroutines are entered by executing a call instruction. Calls can be made like unconditional jumps to any address in a 2K word bank and jumps across the 2K boundary are executed in the same manner. Two separate return instructions determine whether or not status (upper 4 bits of PSW) is restored upon return from the subroutine.

The return and restore status instruction also signals the end of an interrupt service routine if one has been in progress.

Timer Instructions

The 8-bit on board timer/counter can be loaded or read via the accumulator while the counter is stopped or while counting. The counter can be started as a timer with an internal clock source or as an event counter or timer with an external clock applied to the T1 input pin. The instruction executed determines which clock source is used. A single instruction stops the counter whether it is operating with an internal or an external clock source. In addition, two instructions allow the timer interrupt to be enabled or disabled.

Control Instructions

Two instructions allow the external interrupt source to be enabled or disabled. Interrupts are initially disabled and are automatically disabled while an interrupt service routine is in progress and re-enabled afterwards.

There are four memory bank select instructions, two to designate the active working register bank and two to control program memory banks. The operation of the memory bank switch is explained in section 6.1.2. The working register bank switch instructions allow the programmer to immediately substitute a second 8 register working register bank for the one in use. This effectively provides 16 working registers or it can be used as a means of quickly saving the contents of the registers in response to an interrupt. The user has the option to switch or not to switch banks on interrupt. However, if the banks are switched, the original bank will be automatically restored upon execution of a return and restore status instruction at the end of the interrupt service routine.

A special instruction enables an internal clock, which is the XTAL frequency divided by three, to be output on pin T0. This clock can be used as a general purpose clock in the user system. This instruction should be used only to initialize the system since the clock output can be disabled only by application of system reset.

Input/Output Instructions

Ports 1 and 2 are 8-bit static I/O ports which can be loaded to and from the accumulator. Outputs are statically latched but inputs are not latched and must be read while inputs are present. In addition, immediate data from program memory can be ANDed or ORed directly to Port 1 and Port 2 with the result remaining on the port. This allows “masks” stored in program memory to selectively set or reset individual bits of the I/O ports. Ports 1 and 2

are configured to allow input an a given pin by first writing a “1” out to th pin.

An 8-bit port called BUS can also be accessed via the accumulator and can have statically latched outputs as well. It too can have immediate data ANDed or ORed directly to its outputs, however, unlike Ports 1 and 2, all eight lines of BUS must be treated as either input or output at any one time. In addition to being a static port, BUS can be used as a true synchronous bi-directional port using the Move External instructions used to access external data memory. When these instructions are executed a corresponding READ or WRITE pulse is generated and data is valid only at this time. When data is not being transferred BUS is in a high impedance state.

The basic three on board I/O ports can be expanded via a 4-bit expander bus using half of Port 2. I/O expander devices on this bus consist of four 4-bit ports which are addressed as ports 4 through 7. These ports have their own AND and OR instructions like the on board ports as well as move instructions to transfer data in and out. The expander AND and OR instructions, however, combine the contents of accumulator with the selected port rather than immediate data as is done with the on board ports.

I/O devices can also be added externally using the BUS port as the expansion bus. In this case the I/O ports become “memory mapped”, i.e. They are addressed in the same way as external data memory and exist in the external data memory address space addressed by pointer register R0 or R1.

5.1 Instruction Set Description

The following pages describe the MCS-48 instruction set in detail. The instruction set is first summarized with instructions grouped functionally. This summary page is followed by a detailed description listed alphabetically by mnemonic opcode.

The alphabetic listing includes the following information:

Mnemonic
Machine Code
Verbal Description
Symbolic Description

The machine code is represented with the most significant bit (7) to the left and two byte instructions are represented with the first byte on the left.

ADD A, Rr Add Register Contents to Accumulator

(A) ← (A) + (Rr) r = 0 – 7

The contents of register Rr is added to the Accumulator. The Carry flag is modified according to the result of the operation.

0	1	1	0	1	r	r	r

ADD A, @ Rr Add Data Memory Contents to Accumulator

(A) ← (A) + ((Rr)) r = 0 – 1

The contents of the Data Memory location addressed by bits 0 – 5 (0 – 6 for 8039/8049) of register Rr is added to the Accumulator. The Carry flag is modified according to the result of the operation.

0	1	1	0	0	0	0	r

ADD A, data Add Immediate Data to Accumulator

(A) ← (A) + data 2 cycles

The contents byte that follows the opcode is added to the Accumulator. The Carry flag is modified according to the result of the operation.

0	0	0	0	0	0	1	1
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

ADDC A, Rr Add Carry and Register Contents to Accumulator

(A) ← (A) + (C) + (Rr) r = 0 – 7

The Carry flag is added to the Accumulator and then cleared. Afterwards, the contents of register Rr is added to the Accumulator. The Carry flag is modified according to the result of the operation.

0	1	1	1	1	r	r	r

ADDC A, @ Rr Add Carry and Data Memory Contents to Accumulator

(A) ← (A) + (C) + ((Rr)) r = 0 – 1

The Carry flag is added to the Accumulator and then cleared. Afterwards, the contents of the Data Memory location addressed by bits 0 – 5 (0 – 6 for 8039/8049) of register Rr is added to the Accumulator. The Carry flag is modified according to the result of the operation.

0	1	1	1	0	0	0	r

ADDC A, data Add Carry and Immediate Data to Accumulator

(A) ← (A) + (C) + data 2 cycles

The Carry flag is added to the Accumulator and then cleared. Afterwards, the contents byte that follows the opcode is added to the Accumulator. The Carry flag is modified according to the result of the operation.

0	0	0	1	0	0	1	1
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

ANL A, Rr Logical AND Accumulator with Register Mask

(A) ← (A) ∧ (Rr) r = 0 – 7

The contents of the Accumulator is logically (bitwise) anded with the contents of register Rr.

0	1	0	1	1	r	r	r

ANL A, @ Rr

Logical AND Accumulator with Memory Mask

(A) ← (A) ∧ ((Rr)) r = 0 – 1

The contents of the Accumulator is logically (bitwise) anded with the contents of the Data Memory location addressed by bits 0 – 5 (0 – 6 for 8039/8049) of register Rr.

0	1	0	1	0	0	0	r

ANL A, data

Logical AND Accumulator with Immediate Mask

(A) ← (A) ∧ data 2 cycles

The contents of the Accumulator is logically (bitwise) anded with the contents of the byte that follows the opcode.

0	1	0	1	0	0	1	1
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

ANL BUS, data

Logical AND BUS with Immediate Mask

(BUS) ← (BUS) ∧ data 2 cycles

BUS is read and its contents is logically (bitwise) anded with the contents of the byte that follows the opcode. This instruction requires the instruction OUTL BUS, A to be executed in advance.

Note: This instruction is not available for the 8021/8022. It must not be used with the 8035/8039 as BUS is used for external Program Memory access.

1	0	0	1	1	0	0	0
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

ANL Pp, data

Logical AND Port 1 – 2 with Immediate Mask

(Pp) ← (Pp) ∧ data p = 1 – 2 2 cycles

Port p is read and its contents is logically (bitwise) anded with the contents of the byte that follows the opcode.

Note: This instruction is not available for the 8021/8022.

1	0	0	1	1	0	p	p
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

ANLD Pp, A

Logical AND Port 4 – 7 with Accumulator Mask

(Pp) ← (Pp) ∧ A(0 – 3) p = 4 – 7 2 cycles

Port p is read and its bits 0 – 3 are logically (bitwise) anded with bits 0 - 3 of the Accumulator.

1	0	0	1	1	1	p	p

p	1 – 0	Port
0	0	4
0	1	5
1	0	6
1	1	7

CALL addr

Subroutine Call

((SP)) ← (PC), (PSW 4 – 7) 2 cycles
(SP) ← (SP) + 1
(PC 8 – 10) ← addr 8 – 10
(PC 0 – 7) ← addr 0 – 7
(PC 11) ← DBF

The Program Counter and bits 4 to 7 of the PSW are stored on the Program Stack. The Program Stack Counter (PSW 0 – 2) is updated. Program execution continues at the Program Memory location addressed by addr. Bit 11 of the Program Counter is initialized with DBF (determined by last SEL MBx instruction).

A CALL instruction cannot begin at Program Memory locations 2046-2047 or 4094-4095.

The program execution resumes with the next instruction after the CALL.

a_{10}	a_9	a_8	1	0	1	0	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

CLR A

Clear Accumulator

(A) ← 0

The contents of the Accumulator is cleared to 0.

0	0	1	0	0	1	1	1

CLR C

Clear Carry Bit

(C) ← 0

The Carry bit is cleared to 0.

The Carry bit can set by ADD, ADDC, RL C, CPL C, RRC an DAA.

1	0	0	1	0	1	1	1

CLR F0

Clear Flag 0

(F0) ← 0

Flag 0 is cleared to 0.

Note: This instruction is not available for the 8021/8022.

1	0	0	0	0	1	0	1

CLR F1

Clear Flag 1

(F1) ← 0

Flag 1 is cleared to 0.

Note: This instruction is not available for the 8021/8022.

1	0	1	0	0	1	0	1

CPL A

Complement Accumulator

(A) ← ¬ (A)

The contents of the Accumulator is inverted (one's complement).

0	0	1	1	0	1	1	1

CPL C

Complement Carry Bit

(C) ← ¬ (C)

The contents of the Carry bit is inverted.

1	0	1	0	0	1	1	1

CPL F0

Complement Flag 0

(F0) ← ¬ (F0)

The contents of the Flag 0 is inverted.

Note: This instruction is not available for the 8021/8022.

1	0	0	1	0	1	0	1

CPL F1

Complement Flag 1

(F1) ← ¬ (F1)

The contents of the Flag 1 is inverted.

Note: This instruction is not available for the 8021/8022.

1	0	1	1	0	1	0	1

DA A

Decimal Adjust Accumulator

The 8-bit value in the Accumulator is modified resulting in two BCD-encoded values.

1) If the value of bits 0 – 3 is greater than 9 or AC is set then 6 is added to the contents of the Accumulator.

2) If the value of bits 4 – 7 is now greater than 9 or C is set then 6 is added to the most significant nibble.

Example: The Accumulator contains 10011011.

C	AC	7			4	3			0	
0	0	1	0	0	1	1	0	1	1	
						0	1	1	0	Add 6 to bits 0 – 7
0	0	1	0	1	0	0	0	0	1	
		0	1	1	0					Add 6 to bits 4 – 7
1	0	0	0	0	0	0	0	0	1	Overflow to Carry

0	1	0	1	0	1	1	1

DEC A

Decrement Accumulator

(A) ← (A) – 1

The contents of the Accumulator is decremented by 1.

0	0	0	0	0	1	1	1

DEC Rr

Decrement Register

(Rr) ← (Rr) – 1 r = 0 – 1

The contents of register Rr is decremented by 1.

Note: This instruction is not available for the 8021/8022.

1	1	0	0	1	r	r	r

DIS I

Disable External Interrupt

Disable external interrupt requests.

Note: This instruction is not available for the 8021.

0	0	0	1	0	1	0	1

DIS TCNTI

Disable Timer/Counter Interrupt

Disable Timer/Counter interrupt request. A pending interrupt request is cleared. Overflow of the counter does not trigger an interrupt, although the counter continues its operation and sets the T flag eventually.

Note: This instruction is not available for the 8021.

0	0	1	1	0	1	0	1

DJNZ Rr, addr

Decrement Register and Test

(Rr) ← (Rr) – 1 r = 0 – 7 2 cycles
(PC 0 – 7) ← addr; if Rr ≠ 0
(PC) ← (PC) + 2; if Rr = 0

The contents of register Rr is decremented by 1 and tested for zero. If all bits are 0 then program execution is continued with the next instruction. If not all bits are 0 then a jump to the given address is done.

The address must be an 8-bit value, i. e. the jump will be within the current page (256 bytes). If the DJNZ instruction is at location 255 of a page then the jump destination will be located in the following page.

1	1	1	0	1	r	r	r
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

EN I

Enable External Interrupt

Enable external interrupt request.

Note: This instruction is not available for the 8021.

0	0	0	0	0	1	0	1

EN TCNTI

Enable Timer/Counter Interrupt

Enable Timer/Counter interrupts request.

Note: This instruction is not available for the 8021.

0	0	1	0	0	1	0	1

ENT0 CLK

Enable Clock Output

Enables the clock output at T0 pin. A reset deactivates this function.

Note: This instruction is not available for the 8021/8022.

0	1	1	1	0	1	0	1

IN A, Pp

Input Port or Data to Accumulator

(A) ← (Pp) p = 1 – 2 2 cycles

Port p is read and its contents is stored in the Accumulator.

For the 8021 the instruction IN A, P2 triggers storing of P 20 – 23 to the Accumulator bits 0 – 3 (bits 4 – 7 are cleared to 0).

0	0	0	0	1	0	p	p

INC A

Increment Accumulator

(A) ← (A) + 1

The contents of the Accumulator is incremented by 1.

0	0	0	1	0	1	1	1

INC Rr

Increment register

(Rr) ← (Rr) + 1 r = 0 – 7

The contents of register Rr is incremented by 1.

0	0	0	1	1	r	r	r

INC @ Rr

Increment Data Memory Location

((Rr)) ← ((Rr)) + 1 r = 0 – 1

the contents of the Data Memory location addressed by bits 0 – 5 (0 – 6 for 8039/8049) of register Rr is incremented by 1.

0	0	0	1	0	0	0	r

IN A, P0

Input of Port 0 Data to Accumulator

Equivalent to INS A, BUS but not RD' strobe is generated.
Note: This instruction is only available for the 8021/8022.

INS A, BUS

Strobed Input of BUS Data to Accumulator

(A) ← (BUS) 2 cycles

BUS is read with a low-strobe on RD' and it contents is transferred to the Accumulator.

Note: This instruction must not be used with the 8035/8039 as BUS is used for external Program Memory access.

0	0	0	0	1	0	0	0

JBb addr

Jump if Accumulator Bit is Set

(PC 0 – 7) ← addr; if Bb = 1 b = 0 – 7 2 cycles
(PC) ← (PC) + 2; if Bb = 0

If Accumulator bit b is set then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is not set then program execution continues with the next instruction.

Note: This instruction is not available for the 8021/8022.

b_2	b_1	b_0	1	0	0	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JC addr

Jump if Carry is Set

(PC 0 – 7) ← addr; if C = 1 2 cycles
(PC) ← (PC) + 2; if C = 0

If Carry is set then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is not set then program execution continues with the next instruction.

1	1	1	1	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JF0 addr

Jump if Flag 0 is Set

(PC 0 – 7) ← addr; if F0 = 1 2 cycles
(PC) ← (PC) + 2; if F0 = 0

If Flag 0 is set then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is not set then program execution continues with the next instruction.

Note: This instruction is not available for the 8021/8022.

1	0	1	1	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JF1 addr

Jump if Flag 1 is Set

(PC 0 – 7) ← addr; if F1 = 1 2 cycles
(PC) ← (PC) + 2; if F1 = 0

If Flag 1 is set then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is not set then program execution continues with the next instruction.

Note: This instruction is not available for the 8021/8022.

0	1	1	1	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JMP addr

Direct Jump

(PC 8 – 10) ← addr 8 – 10 2 cycles
(PC 0 – 7) ← addr 0 – 7
(PC 11) ← DBF

Program execution continues at the Program Memory location addressed by addr. Bit 11 of the Program Counter is initialized with DBF (determined by last SEL MBx instruction).

a_{10}	a_9	a_8	0	0	1	0	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JMPP @ A

Indirect Jump Within Page

(PC 0 – 7) ← ((A)) 2 cycles

The contents of the Program Memory location addressed by the Accumulator is used as the new page offset of the Program Counter.

1	0	1	1	0	0	1	1

JNC addr

Jump if Carry is Not Set

(PC 0 – 7) ← addr; if C = 0 2 cycles
(PC) ← (PC) + 2; if C = 1

If Carry is not set then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is set then program execution continues with the next instruction.

1	1	1	0	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JNI addr

Jump if Interrupt Input is Low

(PC 0 – 7) ← addr; if I = 0 2 cycles
(PC) ← (PC) + 2; if I = 1

If the external interrupt input pin is low then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is high then program execution continues with the next instruction.

Note: This instruction is not available for the 8021/8022.

1	0	0	0	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JNT0 addr

Jump if Test 0 is Low

(PC 0 – 7) ← addr; if T0 = 0 2 cycles
(PC) ← (PC) + 2; if T0 = 1

If Test 0 is low then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is high then program execution continues with the next instruction.

Note: This instruction is not available for the 8021/8022.

0	0	1	0	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JNT1 addr

Jump if Test 1 is Low

(PC 0 – 7) ← addr; if T1 = 0 2 cycles
(PC) ← (PC) + 2; if T1 = 1

If Test 1 is low then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is high then program execution continues with the next instruction.

0	1	0	0	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JNZ addr

Jump if Accumulator is Not Zero

(PC 0 – 7) ← addr; if Accu ≠ 0 2 cycles
(PC) ← (PC) + 2; if Accu = 0

If the Accumulator contents is not all zero then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is all zero then program execution continues with the next instruction.

1	0	0	1	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JTF addr

Jump if Timer/Counter Flag is Set

(PC 0 – 7) ← addr; if T = 1 2 cycles
(PC) ← (PC) + 2; if T = 0

If the Timer/Counter flag is set then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is not set then program execution continues with the next instruction.

This instruction clears the Timer/Counter flag after it has been read.

0	0	0	1	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JT0 addr

Jump if Test 0 is High

(PC 0 – 7) ← addr; if T0 = 1 2 cycles
(PC) ← (PC) + 2; if T0 = 0

If Test 0 is high then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is low then program execution continues with the next instruction.

Note: This instruction is not available for the 8021/8022.

0	0	1	1	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JT1 addr

Jump if Test 1 is High

(PC 0 – 7) ← addr; if T1 = 1 2 cycles
(PC) ← (PC) + 2; if T1 = 0

If Test 1 is high then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is low then program execution continues with the next instruction.

0	1	0	1	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

JZ addr

Jump if Accumulator is Zero

(PC 0 – 7) ← addr; if Accu = 0 2 cycles
(PC) ← (PC) + 2; if Accu ≠ 0

If the Accumulator contents is all zero then a jump is executed to the Program Memory location addressed by the byte that follows the opcode. If it is not all zero then program execution continues with the next instruction.

1	1	0	0	0	1	1	0
a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

MOV A, data

Move Immediate Data to Accumulator

(A) ← data 2 cycles

The Accumulator is loaded with the contents of the Program Memory location that follows the opcode .

0	0	1	0	0	0	1	1
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

MOV A, PSW

Move PSW Contents to Accumulator

(A) ← (PSW)

The contents of the PSW is copied into the Accumulator.

Note: This instruction is not available for the 8021/8022.

1	1	0	0	0	1	1	1

MOV A, Rr

Move Register Contents to Accumulator

(A) ← (Rr) r = 0 – 7

The contents of register Rr is copied into the Accumulator.

1	1	1	1	1	r	r	r

MOV A, @ Rr

Move Data Memory Contents to Accumulator

(A) ← ((Rr)) r = 0 – 1

The contents of the Data Memory location addressed by bits 0 – 5 (0 – 6 for the 8039/8049) is copied into the Accumulator.

1	1	1	1	0	0	0	r

MOV A, T

Move Timer/Counter Contents to Accumulator

(A) ← (T)

The contents of the Timer/Counter register is copied into the Accumulator.

0	1	0	0	0	0	1	0

MOV PSW, A

Move Accumulator Contents to PSW

(PSW) ← (A)

The contents of the Accumulator is copied into the PSW (all bits).

Note: This instruction is not available for the 8021/8022.

1	1	0	1	0	1	1	1

MOV Rr, A

Move Accumulator Contents to Register

(Rr) ← (A) r = 0 – 7

The contents of the Accumulator is copied into register Rr.

1	0	1	0	1	r	r	r

MOV Rr, data

Move Immediate Data into Register

(Rr) ← (A) r = 0 – 7 2 cycles

The contents of the Program Memory location that follows the opcode is copied into register Rr.

1	0	1	1	1	r	r	r
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

MOV @ Rr, A

Move Accumulator Contents to Data Memory

((Rr)) ← (A) r = 0 – 1

The contents of the Accumulator is copied into the Data Memory location addressed by bits 0 – 5 (0 – 6 for the 8039/8049).

1	0	1	0	0	0	0	r

MOV @ Rr, data

Move Immediate Data to Data Memory

((Rr)) ← data r = 0 – 1 2 cycles

The byte that follows the opcode is copied into the Data Memory location addressed by bits 0 – 5 (0 – 6 for 8039/8049) of register Rr.

1	0	1	1	0	0	0	r
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

MOV T, A

Move Accumulator Contents to Timer/Counter

(T) ← (A)

The contents of the Accumulator is copied into the Timer/Counter register.

0	1	1	0	0	0	1	0

MOVD A, Pp

Move Port 4 – 7 Data to Accumulator

(A 0 – 3) ← (Pp) p = 4 – 7 2 cycles
(A 4 – 7) ← 0

Port p is read and its bits 0 – 3 are copied into the Accumulator. Bits 4 – 7 of the Accumulator are cleared to 0.

See the ANLD Pp, A instruction for coding of the p field.

0	0	0	0	1	1	p	p

MOVD Pp, A

Move Accumulator Data to Port 4 – 7

(Pp) ← (A 0 – 3) p = 4 – 7 2 cycles

Bits 0 – 3 of the Accumulator are copied to port p.

See the ANLD Pp, A instruction for coding of the p field.

0	0	1	1	1	1	p	p

MOVP A, @ A

Move Current Page Data to Accumulator

(PC 0 – 7) ← (A) 2 cycles
(A) ← ((PC))

The contents of the Program Memory location addressed by the Accumulator is copied into the Accumulator. Only bits 0 – 7 of the Program Counter are affected. Therefore, the instruction is restricted to the current page. The Program Counter is restored after the operation.

Note: If the instruction is located at address 255 of the current page then the next page is addressed with the Accumulator as offset.

1	0	1	0	0	0	1	1

MOVP3 A, @ A

Move Page 3 Data to Accumulator

(PC 0 – 7) ← (A) 2 cycles
(PC 8 – 11) ← 0011
(A) ← ((PC))

The contents of the Program Memory location addressed by the Accumulator in page 3 is copied into the Accumulator. The Program Counter is restored after the operation.

1	1	1	0	0	0	1	1

MOVX A, @ Rr

Move External Data Memory Contents to Accumulator

(A) ← ((Rr)) r = 0 – 1 2 cycles

The contents of the external Data Memory location addressed by register Rr is copied into the Accumulator.

Note: This instruction is not available for the 8021/8022.

1	0	0	0	0	0	0	r

MOVX @ Rr, A

Move Accumulator Contents to External Data Memory

((Rr)) ← (A) r = 0 – 1 2 cycles

The contents of the Accumulator is copied to the external Data Memory location addressed by register Rr.

Note: This instruction is not available for the 8021/8022.

1	0	0	1	0	0	0	r

NOP

No Operation

No operation is executed. Program execution continues with the next instruction.

0	0	0	0	0	0	0	0

ORL A, Rr

Logical OR Accumulator With Register Mask

(A) ← (A) ∨ (Rr) r = 0 – 7

The contents of the Accumulator is logically (bitwise) ored with the contents of register Rr.

0	1	0	0	1	r	r	r

ORL A, @ Rr

Logical OR Accumulator With Memory Mask

(A) ← (A) ∨ ((Rr)) r = 0 – 1

The contents of the Accumulator is logically (bitwise) ored with the contents of the Data Memory location addressed by bits 0 – 5 (0 – 6 for the 8039/8049).

0	1	0	0	0	0	0	r

ORL A, data

Logical OR Accumulator With Immediate Mask

(A) ← (A) ∨ data 2 cycles

The contents of the Accumulator is logically (bitwise) ored with the contents of the Program Memory location following the opcode.

0	1	0	0	0	0	1	1
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

ORL BUS, data

Logical OR BUS with Immediate Mask

(BUS) ← (BUS) ∨ data 2 cycles

BUS is read and its contents is logically (bitwise) ored with the contents of the byte that follows the opcode. This instruction requires the instruction OUTL BUS, A to be executed in advance.

Note: This instruction is not available for the 8021/8022. It must not be used with the 8035/8039 as BUS is used for external Program Memory access.

1	0	0	0	1	0	0	0
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

ORL Pp, data

Logical OR Port 1 – 2 With Immediate Mask

(Pp) ← (Pp) ∨ data p = 1 – 2 2 cycles

Port p is read and its contents is logically (bitwise) ored with the contents of the byte that follows the opcode.

Note: This instruction is not available for the 8021/8022.

1	0	0	0	1	0	p	p
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

ORLD Pp, A

Logical OR Port 4 – 7 With Accumulator Mask

(Pp) ← (Pp) ∨ A(0 – 3) p = 4 – 7 2 cycles

Port p is read and its bits 0 – 3 are logically (bitwise) ored with bits 0 - 3 of the Accumulator.

See the ANLD Pp, A instruction for coding of the p field.

Note: This instruction is not available for the 8021/8022.

1	0	0	0	1	1	p	p

OUTL P0, A

Output Accumulator Data to Port 0

Note: This instruction is only available for the 8021/8022.

1	0	0	1	0	0	0	0

OUTL BUS, A

Output Accumulator Data to BUS

(BUS) ← (A) 2 cycles

The contents of the Accumulator is put on the BUS with a low-strobe on WR'. The BUS lines maintain their contents until the next OUTL BUS, A instruction. All other instructions related to BUS (except INS A, BUS) destroy the BUS state. This also includes the MOVX instructions.
Note: This instruction is not available for the 8021/8022.

0	0	0	0	0	0	1	0

OUTL Pp, A

Output Accumulator Data to Port 1 – 2

(Pp) ← (A) p = 1 – 2 2 cycles

The contents of the Accumulator is put on port p. The port lines maintain their contents until they are modified by another instruction related to port Pp.

0	0	1	1	1	0	p	p

RAD

Move Conversion Result Register to Accumulator

2 cycles

The contents of the A/D result register is copied into the Accumulator.

Note: This instruction is only available for the 8022.

1	0	0	0	0	0	0	0

RET

Return Without PSW Restore

(SP) ← (SP) – 1 2 cycles
(PC) ← ((SP))

The Program Stack Counter is decremented by 1. Using this new address, the Program Counter is restored from the Program Stack.

Bits 4 – 7 of the PSW are not restored.

1	0	0	0	0	0	1	1

RETI

Return from Interrupt

(SP) ← (SP) – 1 2 cycles
(PC) ← ((SP)) + 1

The Program Stack Counter is decremented by 1. Using this new address, the Program Counter is restored from the Program Stack.

The interrupt logic enables a new interrupt request.

Note: This instruction is only available for the 8022.

1	0	0	1	0	0	1	1

RETR

Return With PSW Restore

(SP) ← (SP) – 1 2 cycles
(PC) ← ((SP))
(PSW 4 – 7) ← ((SP))

The Program Stack Counter is decremented by 1. Using this new address, the Program Counter is restored from the Program Stack. In addition, bits 4 – 7 of the PSW are restored from the Program Stack.

This instruction should only be used to return from an interrupt service subroutine. Moreover, it should not be used for call/returns within the interrupt service routine as it clears the interrupt flag and thus enables a new interrupt request to be processed by the system (interrupt nesting).

Note: This instruction is not available for the 8021/8022.

1	0	0	1	0	0	1	1

RL A

Rotate Left Without Carry

(A n+1) ← (A n); n = 0 – 6
(A 0) ← (A 7)

The contents of the Accumulator is rotated left by one bit position.

1	1	1	0	0	1	1	1

RLC A

Rotate Left With Carry

(A n+1) ← (A n); n = 0 – 6
(A 0) ← (C)
(C) ← (A 7)

The contents of the Accumulator is rotated left through Carry by one bit position.

1	1	1	1	0	1	1	1

RR A

Rotate Right Without Carry

(A n) ← (A n+1); n = 0 – 6
(A 7) ← (A 0)

The contents of the Accumulator is rotated right by one bit position.

0	1	1	1	0	1	1	1

RRC A

Rotate Right With Carry

(A n) ← (A n+1); n = 0 – 6
(A 7) ← (C)
(C) ← (A 0)

The contents of the Accumulator is rotated right through Carry by one bit position.

0	1	1	0	0	1	1	1

SEL AN0

Select Analog Input 0

Input 0 of the A/D converter is selected and a new conversion is started. A running conversion is aborted and restarted.

Note: This instruction is only available for the 8022.

1	0	0	1	0	1	0	1

SEL AN1

Select Analog Input 1

Input 1 of the A/D converter is selected and a new conversion is started. A running conversion is aborted and restarted.

Note: This instruction is only available for the 8022.

1	0	0	0	0	1	0	1

SEL MB0

Select Memory Bank 0

(DBF) ← 0

MBF is set to 0, selecting bank 0 of the Program Memory. The following JMP and CALL instructions will target the locations 0 – 2047 of the Program Memory.

Note: This instruction is not available for the 8021/8022.

1	1	1	0	0	1	0	1

SEL MB1

Select Memory Bank 1

(DBF) ← 1

MBF is set to 1, selecting bank 1 of the Program Memory. The following JMP and CALL instructions will target the locations 2048 – 4095 of the Program Memory.

Note: This instruction is not available for the 8021/8022.

1	1	1	1	0	1	0	1

SEL RB0

Select Register Bank 0

(BS) ← 0

Bit 4 of PSW is set to 0. Registers 0 – 7 are mapped to Data Memory locations 0 – 7. This is the recommended setting for normal program execution.

Note: This instruction is not available for the 8021/8022.

1	1	0	0	0	1	0	1

SEL RB1

Select Register Bank 1

(BS) ← 1

Bit 4 of PSW is set to 1. Register s0 – 7 are mapped to Data Memory locations 24 – 31.

This is the recommended setting for interrupt service subroutines as Data Memory locations 0 – 7 are not modified. The RETR instruction at the end of the interrupt subroutine will restore BS (bit 4 of PSW).

Note: This instruction is not available for the 8021/8022.

1	1	0	1	0	1	0	1

STOP TCNT

Stop Timer/Counter

The Timer/Counter is stopped.

0	1	1	0	0	1	0	1

STRT CNT

Start Event Counter

T1 is declared as input for the Event Counter. Every high-to-low transition at T1 increments the counter register by 1.

0	1	0	0	0	1	0	1

STRT T

Start Timer

The internal clock is declared as input for the Timer Counter. After 32 machine cycles, the counter register is incremented by 1. This instruction resets the prescaler to 0 by does not change the counter register.

0	1	0	1	0	1	0	1

SWAP A

Swap Nibbles Within Accumulator

(A 4 – 7) ⇄ (A 0 – 3)

Bits 0 – 3 and 4 – 7 of the Accumulator are swapped.

0	1	0	0	0	1	1	1

XCH A, Rr

Exchange Accumulator and Register Contents

(A) ⇄ (Rr) r = 0 – 7

The contents of the Accumulator is exchanged with the contents of register Rr.

0	0	1	0	1	r	r	r

XCH A, @ Rr

Exchange Accumulator and Data Memory Contents

(A) ⇄ ((Rr)) r = 0 – 1

The contents of the Accumulator is exchanged with the contents of the Data Memory location addressed by bits 0 – 5 (0 – 6 for the 8039/8049).

0	0	1	0	0	0	0	r

XCHD A, @ Rr

Exchange Accumulator and Data Memory 4-Bit Data

(A 0 – 3) ⇄ ((Rr)) r = 0 – 1

The bits 0 – 3 of the Accumulator are exchanged with bits 0 – 3 of the Data Memory location addressed by bits 0 – 5 (0 – 6 for the 8039/8049).

0	0	1	1	0	0	0	r

XRL A, Rr

Logical XOR Accumulator With Register Mask

(A) ← (A) ⊕ (Rr) r = 0 – 7

The contents of the Accumulator is logically (bitwise) xored with the contents of register Rr.

1	1	0	1	1	r	r	r

XRL A, @ Rr

Logical XOR Accumulator With Memory Mask

(A) ← (A) ⊕ ((Rr)) r = 0 – 1

The contents of the Accumulator is logically (bitwise) xored with the contents of the Data Memory location addressed by bits 0 – 5 (0 – 6 for the 8039/8049).

1	1	0	1	0	0	0	r

XRL A, data

Logical XOR Accumulator With Immediate Mask

(A) ← (A) ⊕ data 2 cycles

The contents of the Accumulator is logically (bitwise) xored with the contents of the Program Memory location following the opcode.

1	1	0	1	0	0	1	1
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

THE EXPANDED MCS-48™ SYSTEM

6.0 Summary

If the capabilities resident on the single-chip 8048/8049, 8748 or 8045/8039 are not sufficient for your system requirements, special on-board circuitry allows the addition of a wide variety external memory, I/O or special peripherals you may require. The processor can be directly and simply expanded in the following areas:

- Program Memory to 4K words
- Data Memory to 320 words (384 words with 8049)
- I/O by unlimited amount
- Special Functions using 8080/8085 peripherals

By using bank switching techniques maximum capability is essentially unlimited. Bank switching is discussed later in the chapter. Expansion is accomplished in two ways:

1. Expander I/O --- A special I/O Expander circuit the 8243 provides for the addition of four 4-bit Input/Output ports with the sacrifice of only the lower half (4 bits) of port 2 for inter-device communication. Multiple 8243s may be added to this 4-bit bus by generating the required “chip select” lines.

2. Standard 8085 Bus --- One port of the 8048 is like the 8 bit bidirectional data bus of the 8085 microcomputer system allowing interface to the numerous standard memories and peripherals of the MCS-80/85 microcomputer family.

MCS-48 systems can be configured using either both of these expansion features to optimize system capabilities to the application. Both expander devices and standard memories and peripherals can be added virtually any number and combination required.

6.1 Expansion of Program Memory

Program Memory is expanded beyond the resident 1K or 2K words by using the 8058 BUS feature of the MCS-48. All program memory fetches from addresses less than 1024 (2048) occur internally with no external signals being generated (except ALE which is always present). At address 1024 the 8048 automatically initiates external program memory fetches.

6.1.1 Instruction Fetch Cycle (External)

For all instruction fetches from addresses of 1024 (2048) or greater the following will occur:

1. The contents of the 12 bit program counter will be output on BUS and the lower half of port 2.

2. Address Latch Enable (ALE) will indicate the time at which address is valid. The trailing edge of ALE is used to latch the address externally.

3. Program Store Enable (PSEN') indicates that an external instruction fetch is in progress and serves to enable the external memory device.

4. BUS reverts to input (floating) mode and the processor accepts its 8 bit contents as an instruction word.

All instruction fetches including internal addresses can be forced to be external by activating the EA pin of the 8048/8049. The 8035/8039 processors without program memory always operate in the external program memory mode (EA = 5 V).

6.1.2 Extended Program Memory Addressing (Beyond 2K)

For programs of 2K words or less, the 8048/8049 addresses program memory in the conventional manner. Addresses beyond 2047 can be reached by executing a program memory bank switch instruction (SEL MB0, SEL MB1) followed by a branch instruction (JMP or CALL). The bank switch feature extends the range of branch instructions beyond their normal 2K range and at the same time prevents the user from inadvertently crossing the 2K boundary.

Program Memory Bank Switch

The switching of 2K program memory banks is accomplished by directly setting or resetting the most significant bit of the program counter (bit 11). Bit 11 is not altered by normal incrementing of the program counter but is loaded with the contents of a special flip-flop each time a JMP or CALL instruction is executed. This special flip-flop is set by executing an SEL MB1 and reset by SEL MB0. Therefore, the SEL MB instruction may be executed at any time prior to the actual bank switch which occurs during the next branch instruction encountered. Since all twelve bits of the program counter including bit (11) are stored in the stack when a CALL is executed, the user may jump to subroutines across the 2K boundary and the proper bank will be restored upon return. However, the bank switch flip-flop will not be altered on return.

Interrupt Routines

Interrupts always vector the program counter to location 3 or 7 in the first 2K bank and bit 11 of the program counter is held at “0” during the interrupt service routine. The end of the service routine is signaled by the execution of an RETR instruction. Interrupt service routines should therefore be contained entirely in the lower 2K words of program memory. The execution of a SEL MB0 or SEL MB1 instruction within an interrupt routine is not recommended since it will not alter PC11 while in the routine, but will change the internal flip-flop.

6.1.3 Restoring I/O Port Information

Although the lower half of Port 2 is used to output the four most significant bits of address during an external program memory fetch, the I/O information is still output during certain portions of each machine cycle. I/O information is always present on Port 2 lower at the rising edge of ALE and can be sampled or latched at this time.

6.2 Expansion of Data Memory

Data Memory is expanded beyond the resident 64 words by using the 8085 type bus feature of the MCS-48.

6.2.1 Read/Write Cycles

All address and data is transferred over the 8 lines of BUS. A read or write cycle occurs as follows:

1. The contents of register R0 or R1 is outputted on BUS.
2. Address Latch Enable (ALE) indicates address is valid. The trailing edge of ALE is used to latch the address externally.
3. A read (RD') or write (WR') pulse on the corresponding output pins of the 8048 indicates the type of data memory access in progress. Output data is valid at the trailing edge of WR' and input data must be valid at the trailing edge of RD'.
4. Data (8-bits) is transferred in or out over BUS.

6.2.2 Addressing External Data Memory

External Data Memory is accessed with its own tw-cycle move instructions MOVX A, @R and MOVX @R, A which transfer 8 bits of data between the accumulator and the external memory location addressed by the contents of one of the RAM Pointer Registers R0 or R1. This allows 256 locations to be addressed in addition to the resident locations. Additional pages may be added by “bank switching” with extra output lines of the 8048.

6.3 Expansion of Input/Output

There are four possible modes of I/O expansion with the 8048: one using a special low cost expander, the 8243; another using standard MCS-80/85 I/O devices; and a third using the combination memory/I/O expander devices the 8155, 8355 and 8755. It is also possible to expand using standard TTL devices.

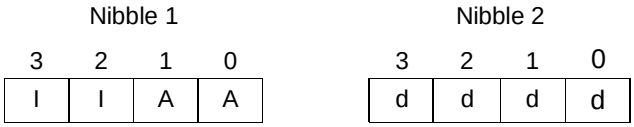
6.3.1 I/O Expander Device

The most efficient means of I/O expansion for small systems is the 8243 I/O Expander Device which requires only 4 port lines (lower half of Port 2) for communication with the 8048. The 8243 contains four 4-bit I/O ports which serve as extension of the on chip I/O and are addressed as ports #4 – 7. The following operations may be performed on these ports:

1. Transfer Accumulator to Port.
2. Transfer Port to Accumulator.
3. AND Accumulator to Port.
4. OR Accumulator to Port.

A 4-bit transfer from a port to the lower half of the Accumulator sets the most significant four bits to zero. All communication between the 8048 and the 8243 occurs over Port 2 lower (P20 – P23) with timing provided by an output pulse on the PROG pin of the processor. Each transfer consists of two 4-bit nibbles:

The first containing the “op code” and port address and the second containing the actual 4 bits of data.



Instruction Code		Port Address	
II		AA	
00	Read	00 --	Port #4
01	Write	01 --	Port #5
10	OR	10 --	Port #5
11	AND	11 --	Port #6

A high to low transition of the PROG line indicates that address is present while a low to high transition indicates the presence of data. Additional 8243s may be added to the four bit bus and chip selected using additional output lines from the 8048/8748.

I/O Port Characteristics

Each of the four 4-bit ports of the 8243 can serve as either input or output and can provide high drive capability in both the high and low state.

6.3.2 I/O Expansion with Standard Peripherals

Standard MCS-80/85 type I/O devices may be added to the MCS-48 using the same bus and timing used for Data Memory expansion. I/O devices reside on the Data Memory bus and in the data memory address space and

are accessed with the same MOVX instructions. See the previous section on data memory expansion for a description of timing. The following are a few of the Standard MCS-80 devices which are very useful in MCS-48 systems.

8214 Priority Interrupt Encoder

8251 Serial Communications Interface

8255 General Purpose Programmable I/O

8279 Keyboard/Display Interface

8253 Interval Timer

6.3.3 Combination Memory and I/O Expanders

As mentioned in the sections on program and data memory expansion the 8355/8755 and 8155 expanders also contain I/O capability.

8355/8755: These two parts are ROM and EPROM equivalents and therefore contain the same I/O structure. I/O consists of two 8-bit ports which normally reside in the external data memory address space and are accessed with MOVX instructions. Associated with each port is an 8-bit Data Direction Register which defines each bit in the port as either an input or an output. The data direction registers are directly addressable thereby allowing the user to define under software control each individual bit of the ports as either input or output. All outputs are statically latched and double buffered. Inputs are not latched.

8155/8156: I/O on the 8155/8156 is configured as two 8-bit programmable I/O ports and one 6-bit programmable port. These three registers and a Control/Status register are accessible as external data memory with the MOVX instructions. The contents of the control register determines the mode of the three ports. The ports can be programmed as input or output with or without associated handshake communication lines. In the handshake mode, lines of the six-bit port become input and output strobes for the two 8-bit ports. See the data sheet below for details. Also included in the 8155 is a 14-bit programmable timer. The clock input to the timer and the timer overflow output are available on external pins. The timer can be programmed to stop on terminal count or to continuously reload itself. A square wave or pulse output on terminal count can also be specified.

6.4 Memory Bank Switching

Certain systems may require more than the 4K words of program memory which are directly addressable by the program counter or more than the 256 data memory and I/O locations directly addressable by the pointer registers R0 and R1. These systems can be achieved using “bank switching” techniques. Bank switching is merely the selection of various blocks or “banks” of memory using dedicated output port lines from the processor. In the case of the 8048 program memory is selected in blocks of 4K words at a time while data memory and I/O are enabled 256 words at a time.

The most important consideration in implementing two or more banks is the software required to cross the bank boundaries. Each crossing of the boundary requires that the processor first write a control bit to an output port before accessing memory or I/O in the new bank. If program memory is being switched, programs should be organized to keep boundary crossing to a minimum. Jumping to subroutines across the boundary should be avoided when possible since the programmer must keep track of which bank to return to after completion of the subroutine. If these subroutines are to be nested and accessed from either bank, a software “stack” should be implemented to save the bank switch bit just as if were another bit of the program counter.

From a hardware standpoint bank switching is very straight-forward and involves only the connection of an I/O line or lines as bank enable signals. These enables are ANDed with normal memory and I/O chip select signals to activate the proper bank.

6.5 Control Signal Summary

The following table summarizes the instructions which activate the various control outputs of the MCS-48 processor.

CONTROL SIGNAL	WHEN ACTIVE
RD'	DURING MOVX A, @R OR INS BUS
WR'	DURING MOVX @R, A OR OUTL BUS
ALE	EVERY MACHINE CYCLE
PSEN'	DURING FETCH OF EXTERNAL PROGRAM MEMORY (INSTRUCTION OR IMMEDIATE DATA)
PROG	DURING MOVD A, P ANLD P, A MOVD P, A ORLD P, A

During all other instructions these outputs are driven to the inactive state.

6.6 Port Characteristics

BUS Port Operations

The BUS port can operate in three different modes: as a latched I/O port, as a bi-directional bus port, or as a pro-

gram memory address output when external memory is used. The BUS port lines are either active high, active low or high impedance (floating). The latched mode (INS, OUTL) is intended for use in the single chip configuration where BUS is not being used as an expander port. OUTL and MOVX instructions can be mixed if necessary. However, a previously latched output will be destroyed by executing a MOVX instruction and BUS will be left in the high impedance state. OUTL should never be used in a system with external program memory, since latching BUS can cause the next instruction, if external, to be fetched improperly.

Port 2 Operations

The lower half of Port 2 can be used in three different ways: as a quasi, bi-directional static port, as an 8243 expander port and to address external program memory. In all cases outputs are driven low by an active device and driven high momentarily by an active device and held high by a 50 KΩ resistor to +5 V.

The port may contain latched I/O data prior to its use in another mode without affecting operation of either. If lower Port 2 (P20-3) is used to output address for an external program memory fetch the I/O information previously latched will be automatically removed temporarily while address is present then restored when the fetch is complete. However, if lower Port 2 is used to communicate with an 8243, previously latched I/O information will be removed and not restored. After an input from the 8243 P(20-3) will be left in the input mode (floating). After an output to the 8243 P(20-3) will contain the value written, ANDed or ORed to the 8243 port.

8243

MCS-48® INPUT/OUTPUT EXPANDER

- **Low Cost**
- **Simple Interface to MCS-48® Microcomputers**
- **Four 4-Bit I/O Ports**
- **AND and OR Directly to Ports**
- **24-Pin DIP**
- **Single 5V Supply**
- **High Output Drive**
- **Direct Extension of Resident 8048 I/O Ports**

The Intel® 8243 is an input/output expander designed specifically to provide a low cost means of I/O expansion for the MCS-48 family of single chip microcomputers. Fabricated in 5 volts NMOS, the 8243 combines low cost, single supply voltage and high drive capability.

The 8243 consists of four 4-bit bidirectional static I/O ports and one 4-bit port which serves as an interface to the MCS-48 microcomputers. The 4-bit interface requires that only 4 I/O lines of the 8048 be used for I/O expansion, and also allows multiple 8243's to be added to the same bus.

The I/O ports of the 8243 serve as a direct extension of the resident I/O facilities of the MCS-48 microcomputers and are accessed by their own MOV, ANL and ORL instructions.

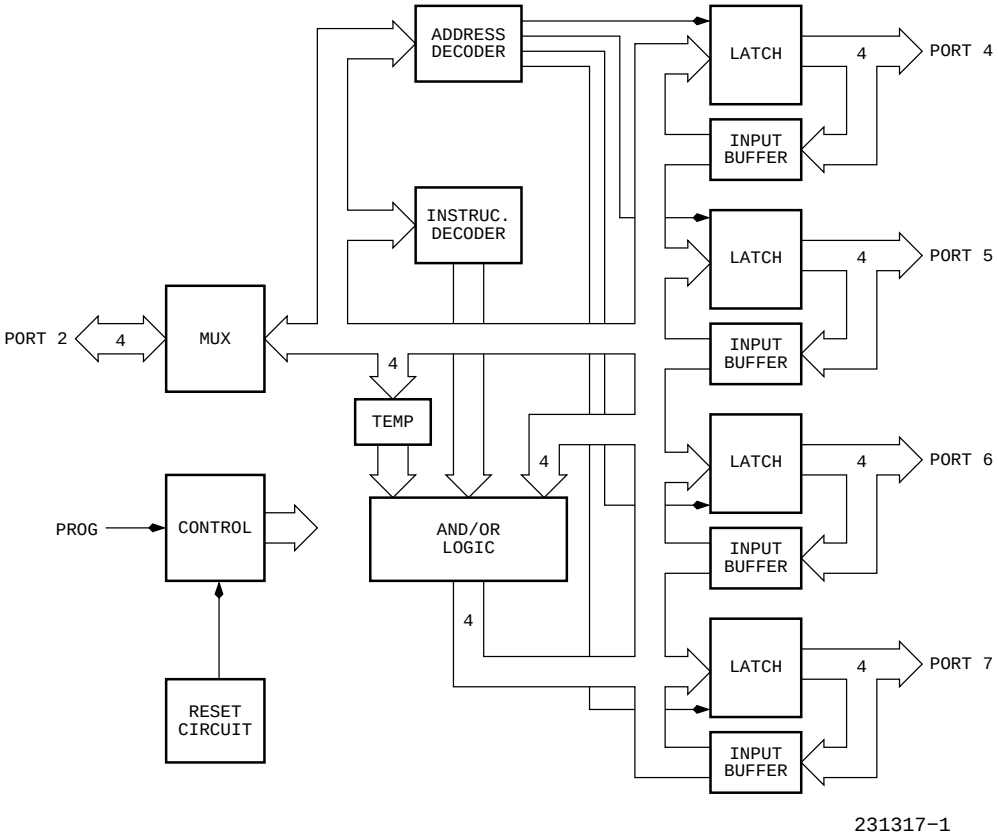


Figure 23. 8243 Block Diagram

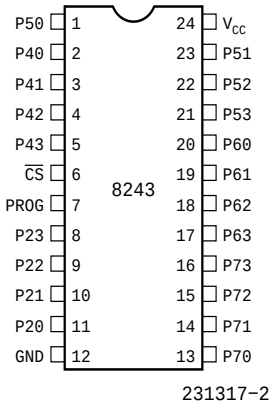


Figure 24. 8243 Pin Configuration

Table 6. Pin Description

Symbol	Pin No.	Function
PROG	7	**CLOCK INPUT.** A high to low transition on PROG signifies that address and control are available on P20 – P23, and a low to high transition signifies that data is available on P20 – P23.
CS'	6	**CHIP SELECT INPUT.** A high on CS inhibits any change of output or internal status.
P20 – P23	11 – 8	Four (4) bit bi-directional port contains the address and control bits on a high to low transition of PROG. During a low to high transition, contains the data for a selected output port if a write operation, or the data from a selected port before the low to high transition if a read operation.
GND	12	0 volt supply.
P40 – P43 P50 – P53 P60 – P63 P70 – P73	2 – 5 1,23 – 21 20 – 17 13 – 16	Four (4) bit bi-directional I/O ports. May be programmed to be input (during read), low impedance latched output (after write), or a tristate (after read). Data on pins P20 – P23 may be directly written, ANDed or ORed with previous data.
V_{CC}	24	+5 volt supply.

FUNCTIONAL DESCRIPTION

General Operation

The 8243 contains four 4-bit I/O ports which serve as an extension of the on-chip I/O and are addressed as ports 4 – 7. The following operations may be performed on these ports:

- Transfer Accumulator to Port
- Transfer Port to Accumulator
- AND Accumulator to Port
- OR Accumulator to Port

All communication between the 8048 and the 8243 occurs over Port 2 (P20 – P23) with timing provided by an output pulse on the PROG pin of the processor. Each transfer consists of two 4-bit nibbles.

The first containing the “op code” and port address and the second containing the actual 4-bits of data. A high to low transition of the PROG line indicates that address is present while a low to high transition indicates the presence of data. Additional 8243's may be added to the 4-bit bus and chip selected using additional output lines from the 8048/8748/8035.

Power On Initialization

Initial application of power to the device forces input/ output ports 4, 5, 6 and 7 to the tr-state and port 2 to the input mode. The PROG pin may be either high or low when power is applied. The first high to low transition of PROG causes device to exit power on mode. The power on sequence is initiated if V_{CC} drops below 1V.

		Address			Instruction
P21	P20	Code	P23	P22	Code
0	0	Port 4	0	0	Read
0	1	Port 5	0	1	Write
1	0	Port 6	1	0	ORLD
1	1	Port 7	1	1	ANLD

Write Modes

The device has three write modes. MOVD Pi, A directly writes new data into the selected port and old data is lost. ORLD Pi, A takes new data, OR's it with the old data and then writes it to the port. ANLD Pi, A takes new data, AND's it with the old data and then writes it to the port. Operation code and port address are latched from the input port 2 on the high to low transition of the PROG pin. On the low to high transition of PROG, data on port 2 is transferred to the logic block of the specified output port.

After the logic manipulation is performed, the data is latched and output. The old data remains latched until new valid outputs are entered.

Read Mode

The device has one read mode. The operation code and port address are latched from the input port 2 on the high to low transition of the PROG pin. As soon as the read operation and port address are decoded, the appropriate outputs are tri-stated and the input buffers switched on. The read operation is terminated by a low to high transition of the PROG pin. The port (4, 5, 6 or 7) that was selected is switched to the tri-state mode while port 2 is returned to the input mode.

Normally, a port will be in an output (write mode) or input (read mode). If modes are changed during operation, the first read following a write should be ignored; all following reads are valid. This is to allow the external driver on the port to settle after the first read instruction removes the low impedance drive from the 8243 output. A read of any port will leave that port in a high impedance state.

ASOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias ..... 0 °C to 70 °C

Storage Temperature ..... -65 °C to +150 °C

Voltage on Any Pin
With Respect to Ground ..... -0.5 V to +7 V

Power Dissipation ..... 1 W

** Notice: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods affect device reliability.*

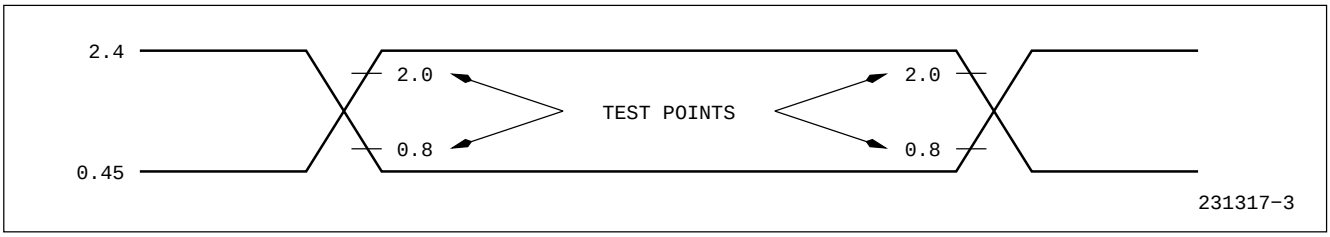
D.C. CHARACTERISTICS T_A = 0 °C to 70 °C, V_{CC} = 5 V ±10%

Symbol	Parameter	Min	Typ	Max	Units	Test Conditions
V_{IL}	Input Low Voltage	-0.5		0.8	V	
V_{IH}	Input High Voltage	0.2		V_{CC} + 0.5	V	
V_{OL1}	Output Low Voltage Ports 4 – 7			0.45	V	I_{OL} = 4.5 mA*
V_{OL2}	Output Low Voltage Port 7			1	V	I_{OL} = 20 mA
V_{OH1}	Output High Voltage Ports 4 – 7	2.4			V	I_{OH} = 240 µA
I_{L1}	Input Leakage Ports 4 – 7	-10		20	µA	V_{IN} = V_{CC} to 0 V
I_{L2}	Input Leakage Port 2, CS, PROG	-10		10	µA	
V_{OL3}	Output Low Voltage Port 2			0.45	V	I_{OL} = 0.6 mA
I_{CC}	V_{CC} Supply Current		10	20	mA	
V_{OL2}	Output Voltage Port 2	2.4			V	I_{OH} = 100 µA
I_{OL}	Sum of all I_{OL} from 16 Outputs			72	mA	4.5 mA Each Pin

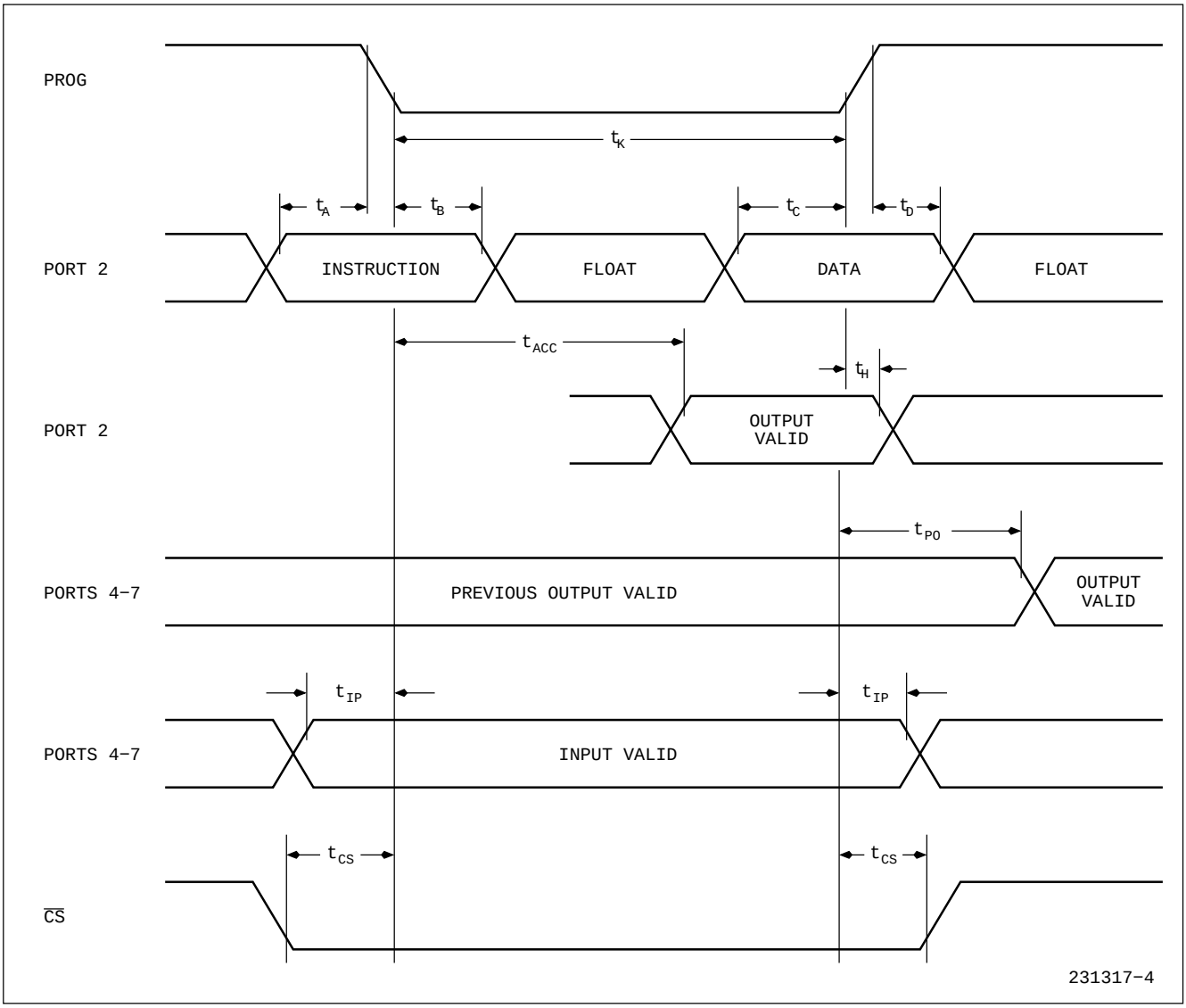
* See following graph for additional sink current capability.

A.C. CHARACTERISTICS T_A = 0 °C to 70 °C, V_{CC} = 5 V ±10%

Symbol	Parameter	Min	Max	Units	Test Conditions
t_A	Code Valid Before PROG	100		ns	80 pF Load
t_B	Code Valid After PROG	60		ns	20 pF Load
t_C	Data Valid Before PROG	200		ns	80 pF Load
t_D	Data Valid After PROG	20		ns	20 pF Load
t_H	Floating After PROG	0	150	ns	20 pF Load
t_K	PROG Negative Pulse Width	700		ns	
t_{CS}	CS Valid Before / After PROG	50		ns	
t_{PO}	Ports 4 – 7 Valid After PROG		700	ns	100 pF Load
t_{LP1}	Ports 4 – 7 Valid Before / After PROG	100		ns	
t_{ACC}	Port 2 Valid After PROG		650	ns	80 pF Load


2.4
0.45
2.0
0.8
TEST POINTS
2.0
0.8
231317-3

WAVEFORMS


PROG
t_K
t_A
t_B
t_C
t_D
PORT 2
INSTRUCTION
FLOAT
DATA
FLOAT
t_{ACC}
t_H
PORT 2
OUTPUT VALID
t_{PO}
PORTS 4-7
PREVIOUS OUTPUT VALID
OUTPUT VALID
t_{IP}
t_{IP}
PORTS 4-7
INPUT VALID
t_{CS}
t_{CS}
$\overline{CS}$
231317-4

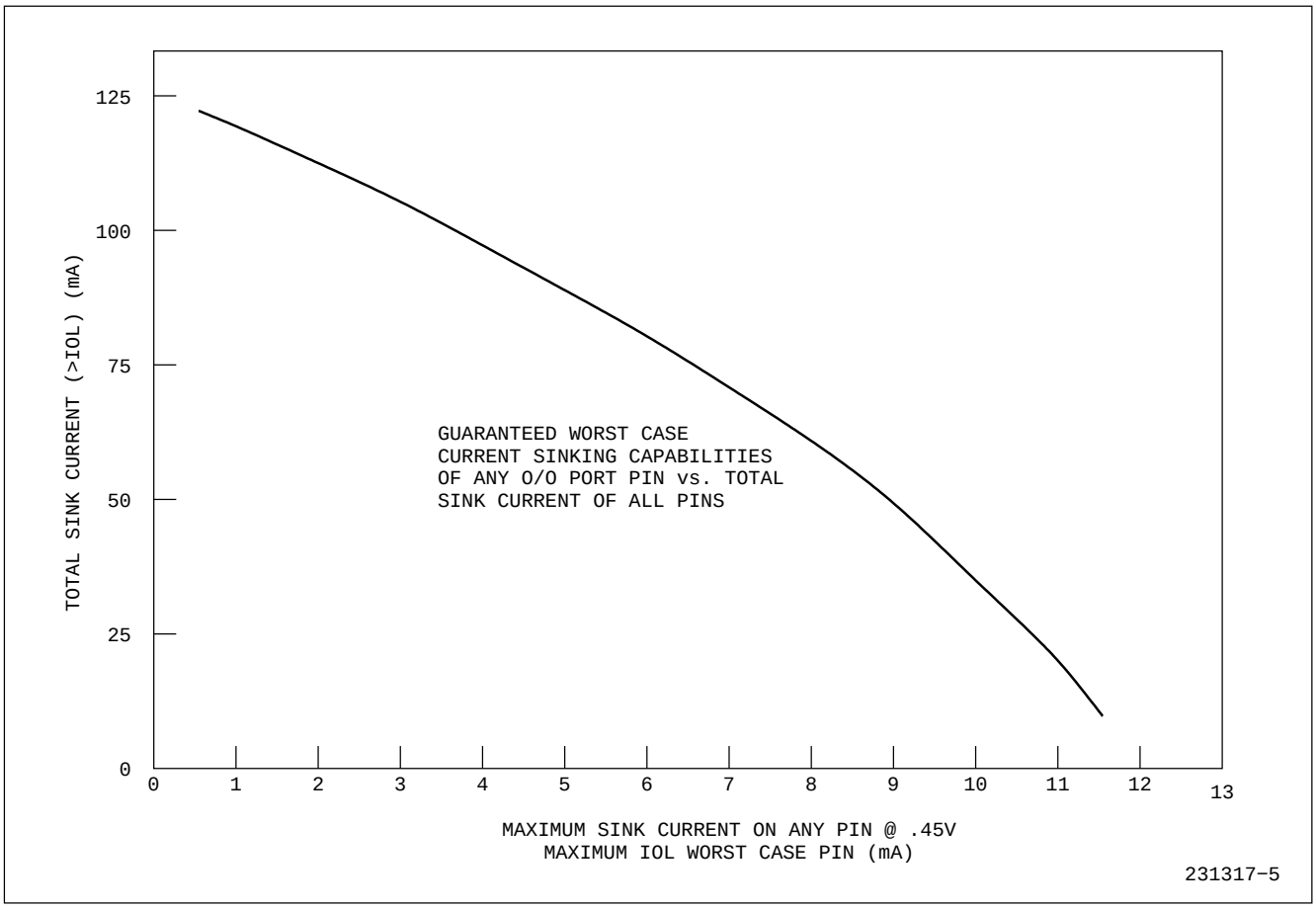


Figure 25.

Sink Capability

The 8243 can sink 5 mA @ 0.45 V on each of its 16 I/O lines simultaneously. If, however, all lines are not sinking simultaneously or all lines are not fully loaded, the drive capability of any individual line increases as is shown by the accompanying curve.

For example, if only 5 of the 16 lines are to sink current at one time, the curve shows that each of those 5 lines is capable of sinking 9 mA @ 0.45 V (if any lines are to sink 9 mA the total I_{OL} must not exceed 45 mA or five 9 mA loads).

Example: How many pins can drive 5 TTL loads (1.6 mA) assuming remaining pins are unloaded?
I_{OL} = 5 x 1.6 mA = 8 mA
$\in I_{OL}$ = 60 mA from curve
pins = 60 mA ÷ 8 mA/pin = 7.5 = 7
In this case, 7 lines can sink can sink 8 mA for a total of 56 mA. This leaves 4 mA sink current capability which can be divided in any way among the remaining 8 i/O lines of the 8243.

Example: This example shows how the use of the 20 mA sink capability of Port 7 affects the sinking capability of the other I/O lines.
An 8243 will drive the following loads simultaneously.
2 loads – 20 mA @ 1 V (port 7 only)
8 loads – 4 mA @ 0.45 V
6 loads – 3.2 mA @ 0.45 V
is this within the specified limits?

$\in I_{OL}$ = (2 x 20) + (8 x 4) + (6 x 3.2) = 91.2 mA. From the curve: for I_{OL} = 4 mA, $\in I_{OL} \approx$ 93 mA. Since 91.2 mA < 93 mA the loads are within the specified limits.

Although the 20 mA @ 1 V loads are used in calculating $\in I_{OL}$, it is the largest current required @ 0.45 V which determines the maximum allowable $\in I_{OL}$.

NOTE:

A 10 kΩ to 50 kΩ pullup resistor to +5 V should be added to 8243 outputs when driving to 5 V CMOS directly.

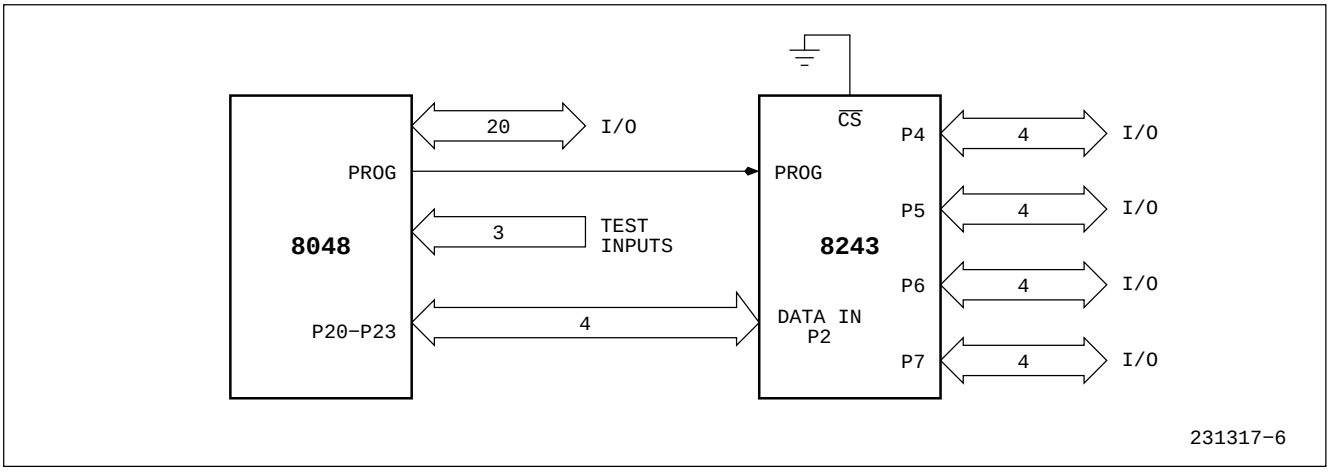


Figure 26. Expander Interface

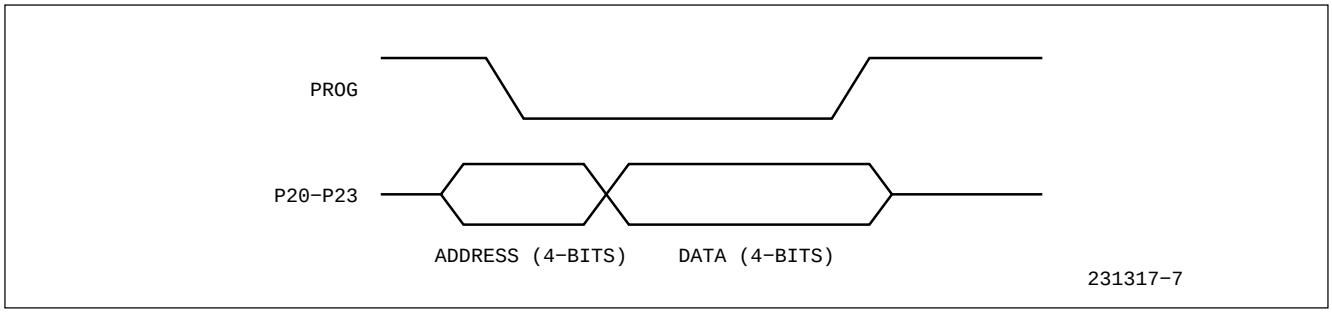


Figure 27. Output Expander Timing

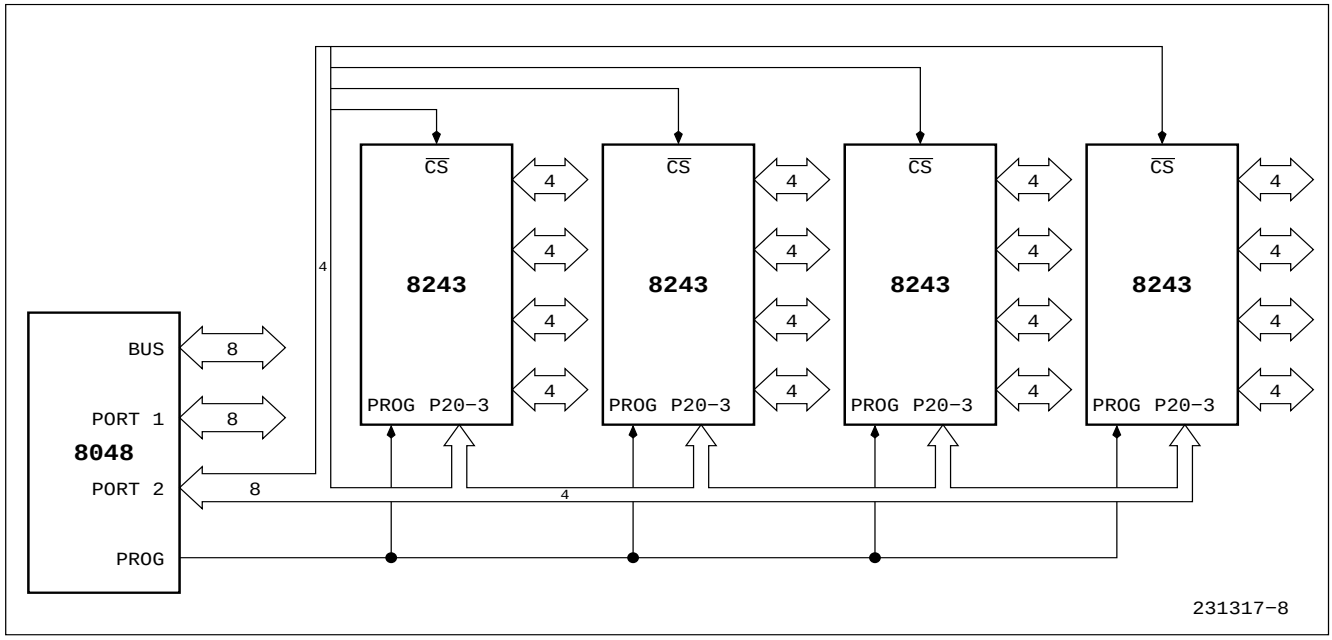


Figure 28. Using Multiple 8243's

PRODUCT: 8245 **REV:** 0 **DATE:** 1/20/78

The 8245 is a general purpose graphics display device that operates in conjunction with raster scan type displays. Its primary purpose is to provide a means for generating and moving objects on a TV screen for use in the customer game market. However, its generality and flexibility makes it suitable for use also in teaching machines, animation displays, simulation trainers, and etc.

This device is a peripheral that communicates over the data and address bus of the 8048/8748. Although other microprocessors may be used with it, these particular devices provide the greatest capability for the low system cost.

FEATURES

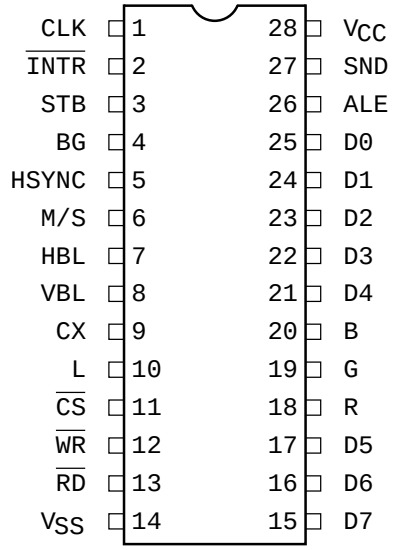
- Single 5V supply
- 28 pin CerDip or plastic DIP package.
- 8048/8748/8085 Compatible.
- Partial color TV sync generator, CCIR standards.
- Provides shapes that are mask programmable in internal ROM.
- Accommodate up to 32 object locations on the display simultaneously.
- Devices may be multiplexed to provide greater than 32 object locations on the display.
- All movement of objects displayed is under software control in the microprocessor.
- Display objects that collide return status and location information to the microprocessor.
- Provides Red, Green, Blue, Luminance, and Sound outputs.

* Note: Signals that are asserted when the variable is low voltage are designated by a $\overline{\ }$, eg. $\overline{CS}$ is active low.

SYMBOL	I/O	
D0-D7	I/O	Data/Address lines to/from 8048/8749.
$\overline{CS}$	I	Chip Select enables writing to or reading from the addressed Functional block within the device.
ALE	I	Address Latch Enable allows the contents of the multiplexed address/data bus to be interpreted as an address.
$\overline{WR}$	I	Write Strobe causes the bus data to be written into the previously selected memory element.
$\overline{RD}$	I	Read Strobe allows status and counter information to be read from the device.
$\overline{IRQ}$	I	Interrupt request to the microprocessor, set Low for request and cleared when the status register is read.
HSYNC	O	Horizontal sync.
VBL	I/O	Vertical blanking identifies the period during which the display is blank while the CRT beam is in vertical retrace.
HBL	O	Horizontal blanking identifies the period during which the display is blank while the CRT beam is in horizontal retrace.
M/S	I	Master/Slave designates a device to be either a master or a slave unit. A master, so designated, feeds VBL and HBL to itself from its internal sync generator and also sends VBL and HBL out to the slave device if one exists. A slave, so designated receives VBL and HBL for its internal synchronization.

CLK	I	The clock input operates at a fixed frequency of 3.54 MHz. The duty cycle shall be 50% with 5% deviation.
R	O	The Red output is a chroma signal representing objects that are to be displayed in a read color.
G	O	The Green output is a chroma signal representing objects that are to be displayed in a green color.
B	O	The Blue output is a chroma signal representing objects that are to be displayed in a read color.
L	O	The Luminance output represents the ORed result of active patterns in the minor system, the major system, and the grid (if set grid bright is active)
BG	O	The Burst Gate defines the duration of the 4.43 MHz color reference signal required for generation of the composite color signal in external analog circuitry.
SND	O	The Sound output provides an audio driving signal to the external sound modulator.
STB	I	The position strobe input.
CX	I	Chip Expander. If there are 2 8245's in a system, LUM #2 is connected to CX #1 and LUM #1 is connected to CX #2. This allows status or overlaps between objects on different chips to be read by the CPU.
VCC	I	+5V supply.
VSS	I	Gnd.

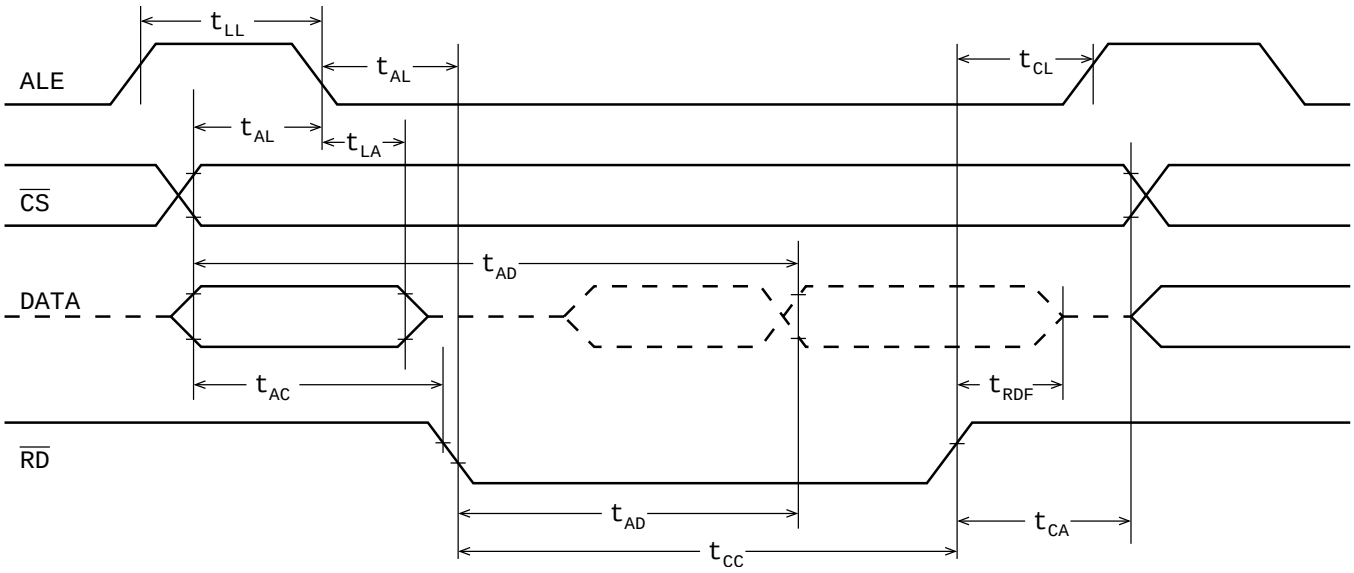
8245 PINOUT



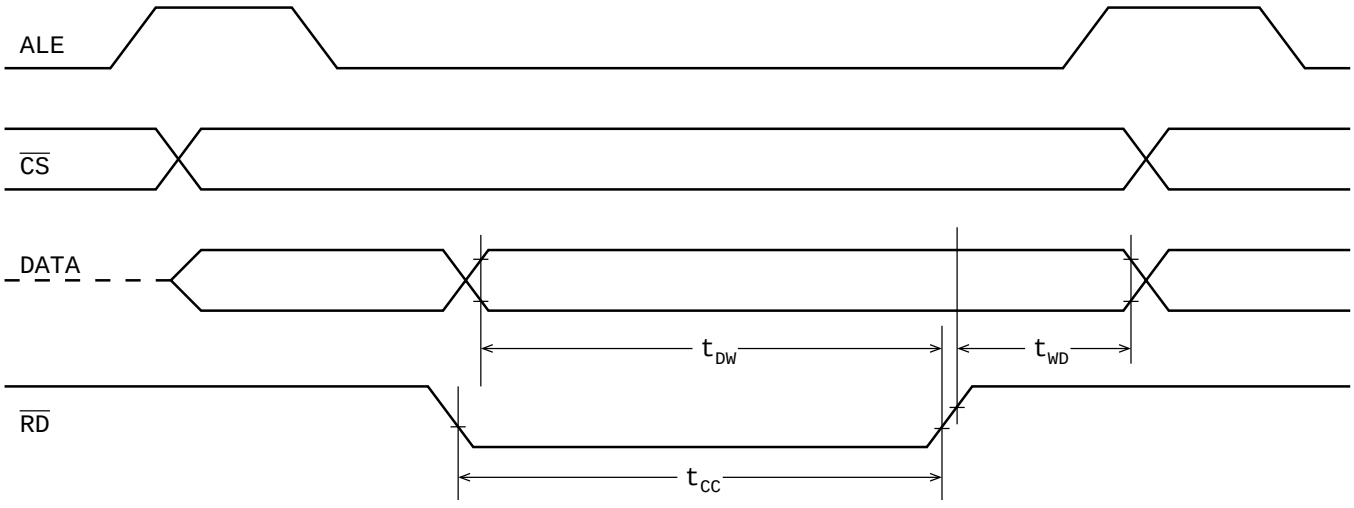
SUMMARY

The 8085 timings are more restrictive than the 8048 timings. Although the initial game product will match an 8245 with an 8048, it is desirable to design the 8245 to work with both the 8048 and the 8085. This will allow upwards compatibility with the more powerful CPU and very probably will extend the product life of the 8245. The following timings should allow the 8245 to work in either system.

READ CYCLE:



WRITE CYCLE:



References: “Standard Peripheral Timing for 8085 Bus”, April 20, 1976; 8048/8748/8035 <cropped> datasheet, September 1976

SYMBOL	DESCRIPTION	MIN	MAX	UNITS
tAL	Address valid before T.E. of ALE	50		NSEC
tLA	Address hold time after ALE	100		NSEC
tLL	ALE width	100		NSEC
tAC	Address valid to L.E. of control	150		NSEC
tLC	T.E. of ALE to L.E. of control	100		NSEC
tAD	Address valid to valid data out		400	NSEC
tRD	Data out delay from RD		150	NSEC
tRDF	Data bus float after RD	10	75	NSEC
tCC	Width of control	250		NSEC
tDW	Data in valid to T.E. of WR	150		NSEC
tWD	Data valid after T.E. of WR	0		NSEC
tCL	T.E. of control to L.E. of ALE	20		NSEC
tRV	T.E. of control to L.E. of next control	300		NSEC
tCA	Address hold after control	0		NSEC

NOTE: The 8245 will ignore the information on the data lines except for the cycles when $\overline{RD}$ or $\overline{WR}$ are active.

8245 ELECTRICAL SPECIFICATION

D.C. CHARACTERISTICS

T_A = 0°C to 70°C; V_{CC} = +5V ±5%; V_{SS} = 0V

SYMBOL	PARAMETER	MIN.	TYP.	MAX.	UNIT	CONDITIONS
V_{IL}*	Input low voltage	V_{SS}-0.5		0.8	V	
V_{IH}*	Input high voltage	2.0		V_{CC}	V	
V_{OL}	Output low voltage			0.45	V	I_{OL} = 1.6mA
V_{OH}	Output high voltage	2.4			V	I_{OH} = -200µA
I_{IL}	Input leakage			± 20	µA	(V_{SS} + 0.45) ≤ V_{IN} ≤ V_{CC}
I_{CC}	V_{CC} current drain			200	mA	

A.C. CHARACTERISTICS

T_A = 0°C to 70°C; V_{CC} = +5V ±5%; V_{SS} = 0V

Timing measurements are made at the following reference voltages unless otherwise noted:

Input “1” = 2.0V, “0” = 0.8V
Output “1” = 2.0V, “0” = 0.8V

Output loading consists of one TTL load and 50pF total external capacitance except the system bus which is loaded by one TTL load and 100pF.

* Except CLK input which has V_{IL} = 0.5 MAX V_{IH} = 4.0 MIN and rise/fall time Rise and fall times will be 200 NSEC or less. Rise and fall times will be measured

from 0.8V to 2.0V. RGB outputs occur within 25 nsec, of each other, luminance output will occur 0 to 150 nsec, after RGB.

FUNCTIONAL SPECIFICATION

The 8245 is organized as a group of subfunction blocks that communicate via an internal bus with the I/O port. Most of the subfunctions are individually addressable for the transfer of information with the controlling microprocessor. These blocks may be categorized functionally as follows:

1. Major display system
2. Minor display system
3. Grid display system
4. Sound system
5. Status and control circuits
6. Sync generator

The use of both major and minor display systems provides hardware parallelism to circumvent the problem of concurrent objects. In general, the single major system is used to display fixed objects, while the plurality of minor systems similarly handles moving objects. In exceptional cases nonstrategic moving objects may be placed in the major system but this should be avoided where accommodation is provided by the minor systems. All objects in the major system are composed of 8 X 7 bit arrays,* while all objects in the minor system are composed of 8 X 8 bit arrays. Larger objects are produced by concatenation of the basic arrays. All major system objects start on even lines.

The grid display system places a segment programmable grid in the background of the display. Any grid segment may be either inserted or deleted programmatically to produce a variety of arrays such as checkerboards, racetracks, mazes, and etc. An additional feature allows vertical segments to be expanded horizontally and thus provide illuminated square and rectangular areas.

The sound system contains both a random noise generator as well as a programmed sound section. The resulting signals may be combined digitally to produce special effects such as gun shot sounds.

The status and control circuits provide a message transfer mechanism between the 8245 and the microprocessor. Control messages sent from the microprocessor are utilized within the control circuits. These messages determine the types of status messages to be returned and also define certain key conditions associated with the displayed objects. The status messages returned to the microprocessor from the 8245 provide information relative to the display that is used by the microprocessor for input to the program.

* An 8 X 7 array is composed of 8 horizontal dots and (7 X 2) horizontal lines.

- missing -

The first group uses four CAM locations to provide starting points for multiple objects. Each group CAM location contains an additional 2-bit counter and thereby is able to point to four LSS* locations. Thus, the first group controls the placement and selection of 16 objects. If the fourth pattern is truncated all four objects will be truncated to the height of the fourth.

The second group within the major system provides for the placement of game obstacles that are either fixed in location or may be movable within certain restrictions. As movable objects, they should not be utilized as strategic elements such as balls, bullets, race cars, etc. However, they do provide slow moving obstacles such as covered wagons or other vehicles. In addition, these objects, when moving, should be prevented from overlapping any other objects in the major system as no means for identification by the microprocessor is available. This nonoverlapping function is achieved by proper programming. Within this group there is a one to one correspondence between a CAM location and a single displayed object. Since there are 12 CAM locations in the second group, there are also 12 objects that may be placed.

The portion of the total CAM array that constitutes the major system points to a total of 28 storage locations in the LSS. The information stored in the LSS represents the location address of the associated pattern in the pattern ROM. Rather than store the starting address of the desired pattern in the LSS, a two's complement displacement is stored. This expedient allows a simple hardware mechanism for sequencing through the consecutive addresses of ROM patterns as they are encountered. The displacement represents the difference between the starting address of the object pattern in ROM and the scanning line number in the raster display. This may be simply stated as follows:

$$LSS_N = \text{ROM object Address} - \left[\frac{(\text{VertiCam Data})_N}{2}\right]$$

N = Object Number

A single 9 bit adder is sufficient for accommodating all address sequencing for the major system. The LSS must be able to store 9 bit displacements. The sequence of events that takes place for the placement of an object pattern is as follows: As a match occurs between the contents of the beam location counter and the address stored in any particular CAM cell, a pointer enables an output from the particular associated line number, obtained from the line counter, and results in the address of the desired row of the object pattern in ROM. If the full 8 X 7 pattern is desired, then the first match of the CAM causes the above described procedure to produce the starting address of the object pattern in ROM. As horizontal matches occur on successive scan lines, a consequential incrementation of the ROM address occurs. An end of pattern address is detected on every eighth address and terminates the presentation of each particular object. There is no restriction as to whether a full 8 X 7 or any portion of the pattern in the vertical dimension may be presented as a displayed object. This flexibility allows the programmer to use fractions of object patterns if it is desired to do so.

*LSS = Linear Select Store

In addition to the 9 bit displacement in each location in the LSS, there is provision for the storage of 3 color bits. These bits designate the primary colors red, green, and blue. By the activation of more than one color bit simultaneously, various hues are also produced.

The following table presents a summary of the specifications for the major system:

Group	No. of CAM Loc.	No. of Bits in Vert. CAM	No. of Bits in Hor. CAM	No. of Simul Objects CAM loc.	No. of Simul Objects Group	No. of Selectable Objects *	Disp. Bits in LSS	Attribute Bits in LSS	Object Size
1	4	7	8	4	16	64	9	3	8 dots wide 14 lines high
2	12	7	8	1	12	64	9	3	8 dots wide 14 lines high

* There are 64 objects total, any of which can be selected by either Group.

MINOR DISPLAY SYSTEM

As a first order objective all strategic objects are selected and displayed by the minor display system. In some games it may be useful to put fixed objects in the minor system and there is no restriction that prevents this. There are four replicated blocks within the minor system. Each block is autonomous in function and provides for the placement of a single object. Each of these objects may collide with each other or with objects in the major system. In either event, the minor object is readily identifiable by the microprocessor so that immediate responsive action may be taken.

Minor system objects are locatable by a portion of the overall CAM array, just as in the major system. However, the similarity of the two systems ends in the signal path beyond the CAM array. Each of the four CAM locations in the minor system is dedicated and points to a singular block of object pattern bits located in RAM storage. In contrast, the CAM locations in the major system can point to any of the 64 objects stored in the pattern ROM. In addition, the mechanism for sequencing through the rows of the pattern RAM's is different than that used in the major system. In place of a singular adder as used in the major system, each minor system contains its individual three bit counter. This counter is updated at the beginning of each horizontal scan line. The decoding of the counter points to the proper location in the pattern RAM. Thus, each dot row in an object is presented as the RAM locations are sequenced. The RAM locations are loadable from the internal bus by a Write operation of the microprocessor. The RAM has the capacity to store eight bytes for each object thereby allowing an 8 X 8 object presentation. Associated with each minor system is an Attribute Register whose contents are arranged as follows:

7	6	5	4	3	2	1	0
X	X	B	G	R	D	S	X_9

The Bits in this register are defined as follows:

Bit 0 – X_9 is the ninth Bit in the horizontal address of the beam location. This bit allows the beam location to be resolved to 140ns increments.

Bit 1 – The S or smoothing Bit allows a displacement of either the odd or even count horizontal sweep lines in order to provide an improved appearance of objects that visually rotate on the screen.

Bit 2 – The D or duration Bit determines whether an object will be presented in normal size or if its x and y dimensions will be increased by a factor of two.

Bit 3 – The R Bit specifies whether the object contains a red component of color display.

Bit 4 – The G Bit specifies whether the object contains a green component of color display.

Bit 5 – The B Bit specifies whether the object contains a blue component of color display.

Bits 6 and 7 – These bits are unspecified and exert no control.

The definition of the delay of dot rows within an object depends on Bits 0, 1, and 2 as shown in the following table:

Bit 2 D	Bit 1 S	Bit 0 X_9	Even Line Delay (ns)	Odd Line Delay (ns)
0	0	0	0	0
0	0	1	140	140
0	1	0	140	0
0	1	1	0	140
1	0	0	0	0
1	0	1	280	280
1	1	0	280	0
1	1	1	0	280

GRID DISPLAY SYSTEM

The grid display consists of an array of nine enclosed areas horizontally and eight areas vertically. Each line segment between the nodes of the array is individually controllable so that it may be presented or be inhibited.

A full array, consisting of all segments present, is created by combining nine complete horizontal display bars with ten complete vertical display bars. Each horizontal bar consists of nine concatenated bar segments, while a vertical bar consists of eight concatenated bar segments. Each horizontal bar on the TV screen is composed of three consecutive horizontal scan lines, while adjacent bars are spaced by 21 horizontal scan lines. A vertical bar is made up of a column of dot groups. Each dot group is programmable to consist of either two or sixteen clock intervals (3.54Mhz) in width*. A conventional grid utilizes two clock intervals while large area clock arrays, such as checkerboards, utilize sixteen clock intervals. The spacing between adjacent vertical bars is fourteen clock intervals. Thus, wide vertical bar segments that are adjacent, appear to be continuous displayed areas. The grid is centered vertically on the TV screen by allowing the first or top horizontal bar to start on the 24th horizontal scan line relative to the end of vertical blanking (VBL). Similarly, horizontal centering is accomplished by allowing the first or left-most vertical bar to start on the 10th clock cycle from the end of horizontal blanking (HBL).

A programmable feature allows the grid to be augmented by the addition of a dot matrix. In this case the dots appear at a physical placement on the TV screen, where otherwise the intersection of the horizontal and vertical bars would appear. The dots are composed of three horizontal scan line segments that have a width equivalent to two clock cycles. Since, a full array of dots is always presented, no segment programming requirement exists for dot arrays, although segments and dots can simultaneously be displayed.

An additional programmable feature allows the grid display, or any of its previously described subsets, to be either presented or inhibited on the TV screen.

The upper left hand corner of the grid has program coordinates (Y,X) = (13H,0BH). The observed position on the screen will be (Y,X) = (18H,18H).

* Horizontal displacement on the TV screen is directly proportional to time.

SOUND SYSTEM

The sound system generates a duty cycle modulated square wave from which an audio signal is extracted by means of an external low pass filter. The control of the duty cycle is effected by information that is transferred from the microprocessor to the 8245. This information consists of triple byte groups that determine the audio frequency and an accompanying 4 bits that determine volume.

The triple byte groups are loaded into three-eight bit shift registers located on the 8245. Each byte in the group is loaded sequentially into its respective register during a load interval. All three bytes are loaded in between consecutive shift clock pulses. The concatenation of the three registers results in a 24 bit string that is shifted out by this shift clock. The resulting serial pattern of ones and zeroes contains a fundamental band of frequency components that lie in the audio range. This particular signal is further "chopped" by a higher frequency that is a multiple of the shift clock. By duty cycle modulation of this "chopping" signal, the amplitude of the audio component is varied. There are four control bits that are used to control the audio level. These bits are loaded into a four bit down counter that is shifted by the high frequency shift clock. The resulting output is ANDED with the output from the three concatenated shift registers to produce the composite audio output. In addition to the four volume control bits, three other control bits are used to augment the overall operation of the sound system. A noise enable bit enables a feedback path in the output eight bit shift register, in the 25 bit shift path to produce the noise component. Simultaneously, the noise is added to the audio component that is progressing down the shift register.

The shift frequency for the register may be varied between two values by another control bit. This expedient allows low audio frequencies to be produced with fewer refresh cycles from the microprocessor than for high frequencies thus, reducing the load on the processor.

For the reproduction of certain audio tones that are subharmonically related to the shift clock, the need for microprocessor refresh is totally eliminated by recirculation of the 24 bit shift path.

The format of the sound control word is described below:

7	6	5	4	3	2	1	0
EN	X	S	N	V_3	V_2	V_1	V_0

Bits 0 – 3 – Volume Bits, collectively as a 4 bit word, these bits define the output audio level.

Bit – 4 – Noise Enable; controls noise generation and mixing with the audio signal. Bit 4 = 1, noise on; Bit = 0, noise off.

SOUND SYSTEM

Bit 5 – Shift Frequency; determines frequency of shift clock.
Bit 5 = 1, f = 3933Hz. Bit 5 = 0, f = 983Hz.

$$\left(3933\,\text{Hz} = \frac{H}{4} = \frac{15{,}734}{4}\,;\ 983\,\text{Hz} = \frac{H}{16} = \frac{15{,}634}{16}\right)$$

Bit 7 – Enable sound; 0 = No sound, 1 = sound.

For those modes of operation requiring sound refresh data from the microprocessor, an interrupt is generated each time that the 24 sound bits have been shifted through the three eight bit shift registers. A 5 bit counter set to module 24 counts shift clocks and determines when the interrupt should occur.

The sound shift registers, volume counter and sound control word register are all individually addressed by the microprocessor for the purpose of loading data. The address of these elements is shown under the topic of “Address Structure”.

CONTROL AND STATUS

The control over various operational parameters on the chip is effected by the bits in the control word that is written into the control register by the microprocessor. The bits in the control word are defined as follows:

Bit 0 – Enable Horizontal Interrupt, generates an interrupt 20 µs in advance of the occurrence of horizontal blanking. This advance notice to the microprocessor allows a sufficient interval for the reading of the status information so that appropriate control can be exerted during the horizontal blanking interval.

Bit 1* - Forced Position Strobe, allows the freezing of the beam location information in the X-Y position registers so that the microprocessor can locate the beam at any time. Bit 1 = 1 strobes beam location to X-Y registers. Bit 1 = 0 disables strobe internal, but external strobe through position strobe Pin can still take place.

Bit 2 – Enable Sound Interrupt, allows an interrupt to be generated whenever the sound register needs new data. Bit 2 = 1 enables sound interrupt, Bit 2 = 0 disables sound interrupt.

Bit 3 – Enable Grid, allows two luminance levels of the grid. If the bit is a 0, the luminance signal is inhibited during grid information intervals. If the Bit is a 1, the luminance signal is active during grid information intervals.

Bit 4 – Enable External Overlap Interrupt, allows an interrupt upon detection of an overlap, when more than one 8245 exists in a system.

Bit 5 – Enable Display, allows objects to be displayed. Bit 5 = 1 enables display; Bit 5 = 0 disables objects. The purpose of this bit is to allow the µP to write new data to the 8245 when the display is part way through a display field. It is the responsibility of the software to disable the display only when there are no active patterns being displayed.

Bit 6 – Dot Enable, allows the presentation of a dot array in addition to the grid array. The dots appear at the intersection of the horizontal and vertical lines in the grid format. Bit 6 = 1 enables dots; bit 6 = 0 enables normal grid.

Bit 7 – Grid Segment Width, allows the selection of narrow or wide vertical segments in the grid. Bit 7 = 1 enables wide segments; bit 7 = 0 enables narrow grid.

The color of the grid and background is determined by the data stored in the Color Latch. Also by setting the Enable Grid bit to '0' the grid can be turned off and the grid RAM can be used for other data storage. The bits in the Color Latch are defined as follows:

*The “or” of STB and Force Position Strobe will cause X-Y reg to follow the BLC. A falling edge on the OR output will freeze the BLC. The reg will remain frozen until after the x-register is read. This strobe will not cause false data to be loaded into latch (synchronized on 0 edge, when BLC is not changing).

Bit 0 – Grid Color Blue
Bit 1 – Grid Color Green
Bit 2 – Grid Color Red
Bit 3 – Background Color Blue
Bit 4 – Background Color Green
Bit 5 – Background Color Red
Bit 6 – Set Grid Bright

CONTROL AND STATUS

The Enable Overlap register allows for selectable masking of overlaps. When a bit in the Enable Overlap register is a '0' the overlap of that object with any other object will not set the bits for the other objects in the Overlap Status register. The bit pattern is as follows:

Bit 0 – Minor System 0
Bit 1 – Minor System 1
Bit 2 – Minor System 2
Bit 3 – Minor System 3
Bit 4 – Vertical Grip
Bit 5 – Horizontal Grid and dots
Bit 6 – External Chip
Bit 7 – Major System

The Overlap Status register stores the coincidences as they occur on the screen. Whenever two or more objects are simultaneously displayed the bits for both objects are set unless the Enable Overlap register has those bits masked. The External Chip overlap corresponds to the Signal on the 'CX' Pin. The bit pattern is the same as that of the Enable Overlap Register above. This register is reset when read.

The Control Status Word is used to determine the chip status and interrupt sources. The bit pattern is as follows:

Bit 0 – Horizontal Status – Starts 20 µs before Horizontal blank starts. Ends 5 µs before Horizontal blank ends.

Bit 1 – Position Strobe Status – Status of X-Y Register strobe '1' = Follow Beam Location Ctr, '0' = latched. (See note on page 14)

Bit 2 – Sound Needs Service – Sound register empty

Bit 3 – Vertical Status = Vertical Blanking

Bit 4 – N/C

Bit 5 – N/C

Bit 6 – External Chip Overlap Interrupt – Set when an overlap occurs with signal on 'CX' Pin.

Bit 7 – Major System Overlap – Set when the chip attempts to load major system shift register if shift register already has one. The status register bits 2, 6 and the interrupt flip flop are cleared by reading the status reg. The overlap status register is cleared when read.

SYNC GENERATOR

The sync generator operates as a non-addressable autonomous circuit block within the 8245. As such, it provides a source for synchronizing signals both for internal use by the 8245 circuitry and for transmission to external circuitry. Externally the signal becomes processed with the color, luminance, and sound signals and ultimately results in synchronization of the associated TV receiver.

The manner in which the signals are utilized is determined by the mode in which the 8245 operates in a particular configuration. In a small system, utilizing a single 8245, it is operated in the Master Mode. The signals from the sync generator drive both the display circuitry on the 8245 and also exit the chip, via appropriate pins, to drive the external circuitry. The external logic and the 8245 sync logic together generate sync signals compatible with CCI standards.

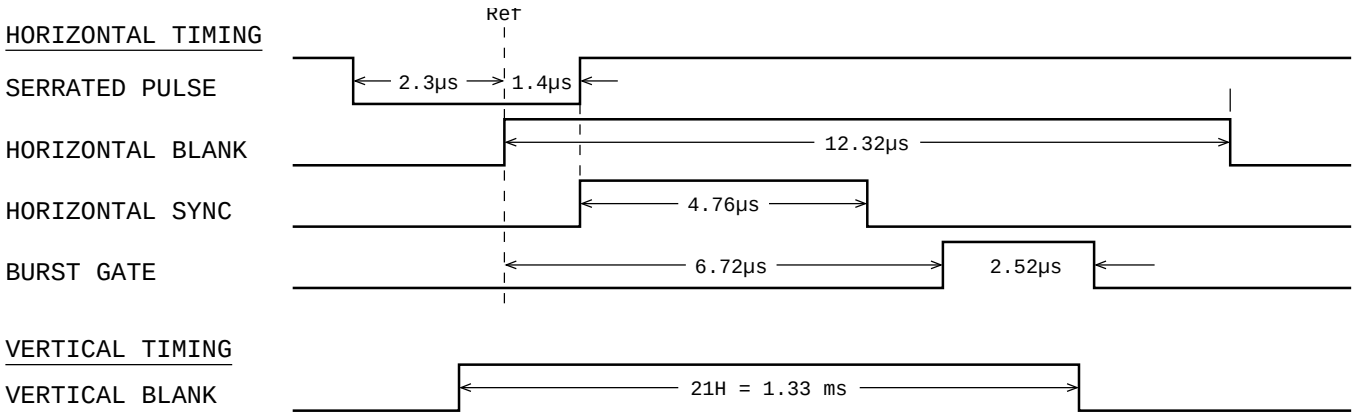
In larger system configurations, where the need for more than one 8245 exists, a single 8245 is designated the Master, as previously described. In addition, one or more 8244's are used as Slave Mode devices by connecting their M/S pins to Vss. The sync generator on a Slave Mode device is free to run but the output is not utilized outside the chip.

In Master Mode operation the sync generator signals used internally by the 8245 are horizontal blanking (HBL) and vertical blanking (VBL). Correspondingly, these two signals provide outputs along with horizontal sync (HSYNC) and color burst gate (BG). In Slave Mode operation the 8244 received only HBL and VBL from a Master Mode 8245.

The sync generator in conjunction with external logic provides non-interlaced synchronizing signals. It operates from a frequency of 3.54 Mhz. This clocking signal is divided by a factor of 227.0 in order to obtain the horizontal line frequency of 15,625 Hz.

In conjunction with external circuitry the horizontal line frequency is divided by a factor of 313 or 312 to produce the vertical sync frequency of 49.92 Hz. In the following timing diagrams, those signals shown under Horizontal Timing are reproduced at the 15,625 Hz rate, while the signals shown under Vertical Timing are reproduced at the 49.92 Hz or 50.08 Hz rate. Horizontal Blanking Burst Gate, and Horizontal Sync are generated during all portions of the vertical timing sequence. Vertical Interrupts are not generated within the 8245, although the vertical status bit is set during Vertical Blanking.

SYNC GENERATOR



ADDRESS STRUCTURE

The subfunction blocks within the 8244 may be individually addressed for the writing and in some cases, the reading of data. The addressing structure of these blocks is shown below:

CAM AND LINEAR SELECT STORE

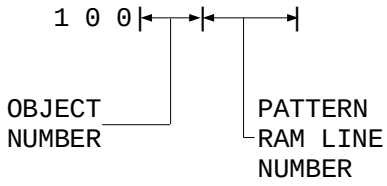
ADDRESS		SUBFUNCTION
Bit: 7 6 5 4 3 2 1 0		
0 | Object	0 0	LINE CAM (Y CAM)
0 | Number	0 1	DOT CAM (X CAM)
0 |	1 0	LSS BITS 0-7
0 |	1 1	LSS BITS 8-11

Note: For the Minor System LSS Bits 0-7 are attribute bits stored in the Attribute register.

ADDRESS STRUCTURE

MINOR SYSTEM PATTERN RAM

ADDRESS	SUBFUNCTION
Bit: 7 6 5 4 3 2 1 0	PATTERN RAM



MISCELLANEOUS REGISTERS

ADDRESS	SUBFUNCTION
Bit: 7 6 5 4 3 2 1 0	
1 0 1 X 0 0 0 0	CONTROL
0 0 0 1	CONTROL STATUS
0 0 1 0	OVERLAP STATUS AND OVERLAP ENABLE
0 0 1 1	COLOR LATCH
0 1 0 0	Y REGISTER
0 1 0 1	X REGISTER
0 1 1 0	NA
0 1 1 1	SOUND 0
1 0 0 0	SOUND 1
1 0 0 1	SOUND 2
1 0 1 0	SOUND STATUS

GRID

ADDRESS	SUBFUNCTION
Bit: 7 6 5 4 3 2 1 0	
1 1 0 0 | |	HORIZONTAL SEGMENTS 0 to 7,
1 1 0 1 | column |	HORIZONTAL SEGMENTS 8
1 1 1 0 | number |	VERTICAL SEGMENTS

READ AND WRITE CAPABILITY:

READ/WRITE: ALL CAM
ALL LINEAR STORE EXCEPT MINOR SYSTEM
GRID RAM
MINOR SYSTEM PATTERN RAM
CONTROL REG.
SOUND STATUS REG.

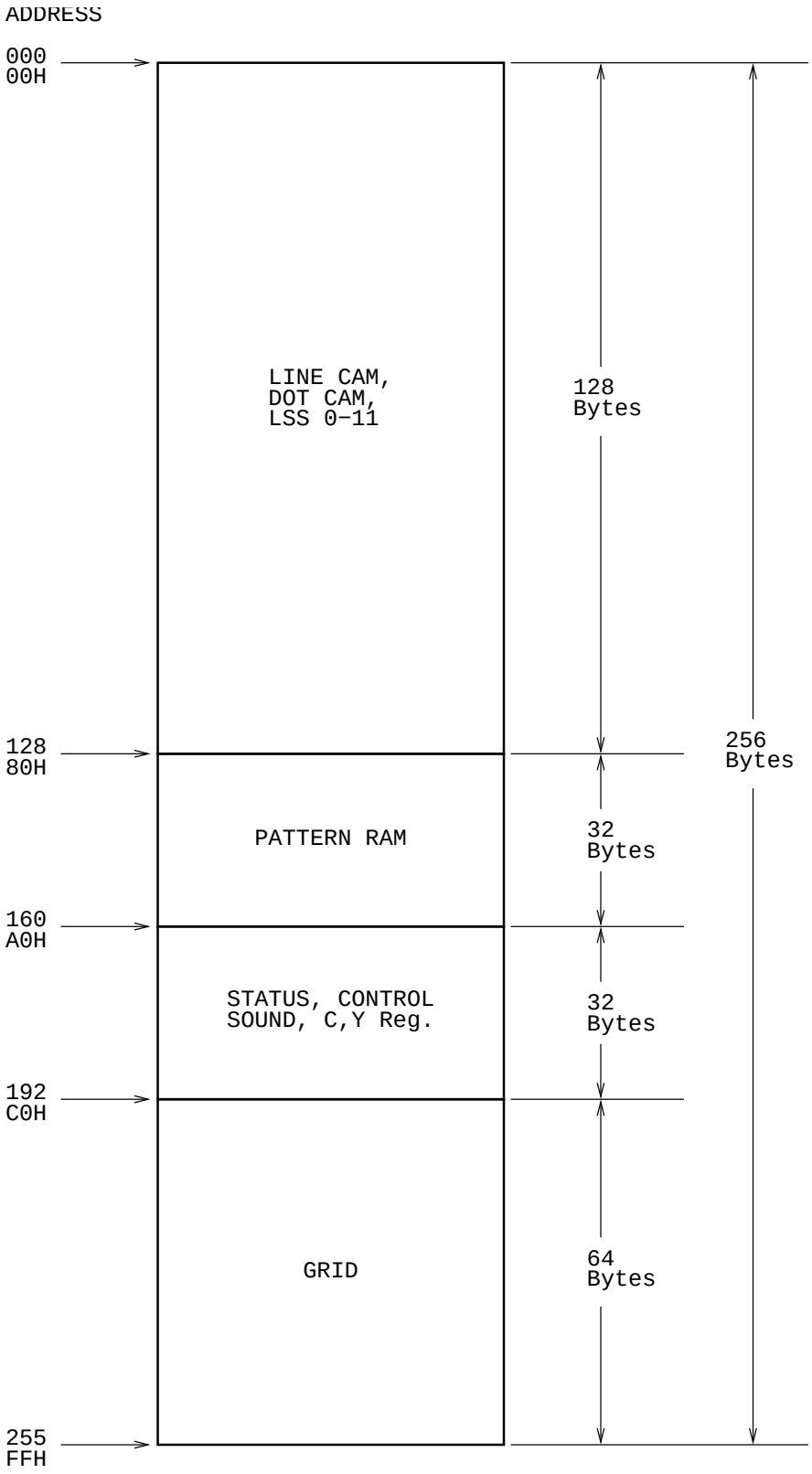
ADDRESS STRUCTURE

READ ONLY: X-REG
Y-REG
OVERLAP STATUS REG
OVERLAP STATUS REG

WRITE ONLY: MINOR SYSTEM LINEAR STORE (ATTRIBUTE REG)
COLOR LATCH
ENABLE OVERLAP
SOUND REGS 0, 1, 2

ADDRESS STRUCTURE

The addressable function block structure may be shown by means of an overview address map as follows:



REGISTER ADDRESS

		CONTROL	CONTROL STATUS	OVERLAP STATUS ENABLE OVERLAP	Y REGISTER	X REGISTER
		A0	A1	A2	A4	A5
	7	Grid Width 1 = Wide	Major with Major	Major with minor or grid	MSB	MSB
BITS	6	Dot Enable	Ext. Chip Overlap*	External Chip		
	5	Enable Overlay	————	Horiz Grid		
	4	Enable Ext. Overlap	————	Vert Grid		
	3	Enable Grid	Vert. Status (V.B)	Minor Sys 3		
ENABLE	2	Enable Sound Int.	Sound Needs Service*	Minor Sys 2		
GRID	1	Forced Pos. Strobe*	Position Strobe Status	Minor Sys 1		
	0	Enable Horiz. Int.	Horiz. Status*	Minor Sys 0	LSB	LSB

* Explained Below

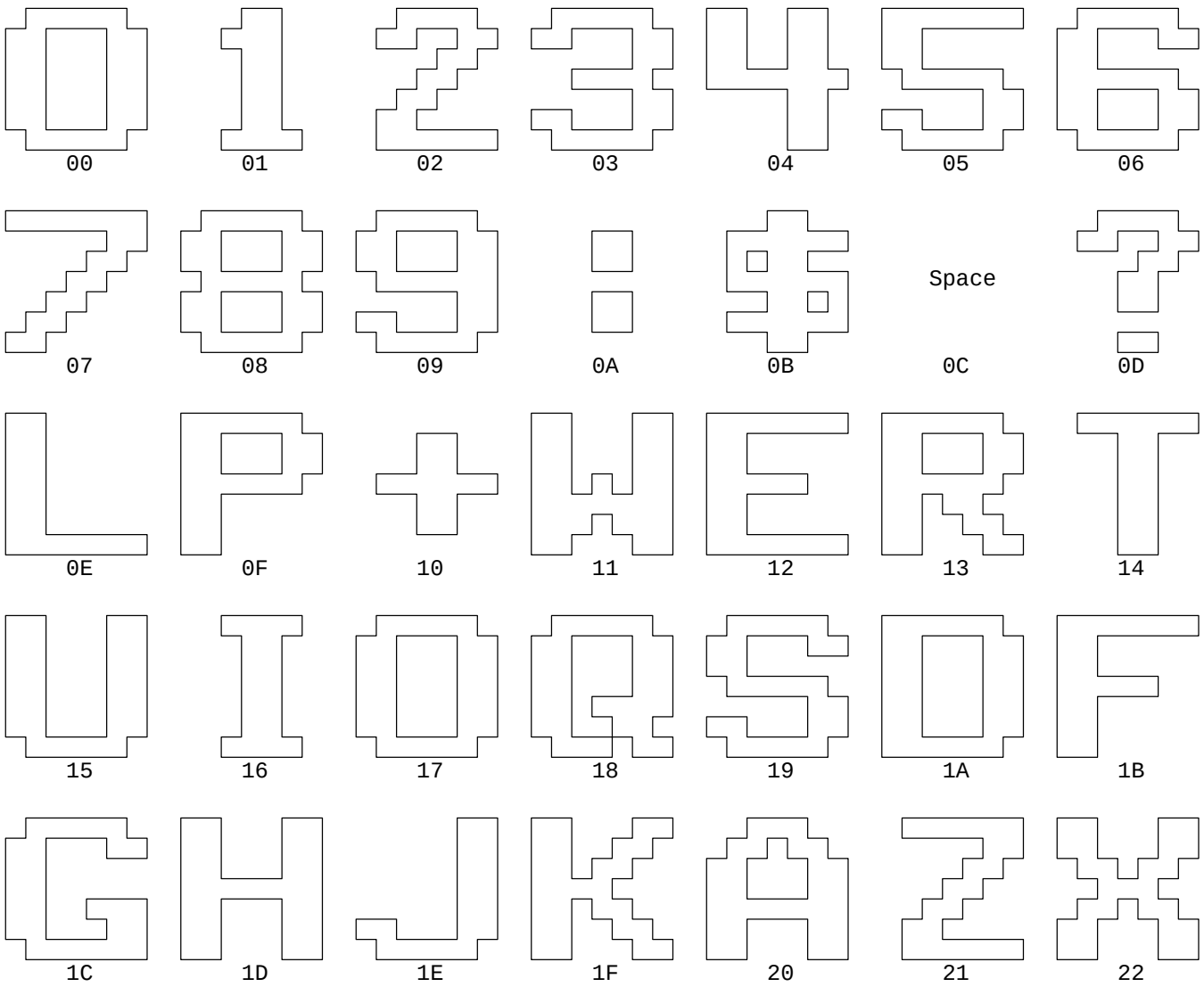
Forces Position Strobe: Bit = 1 Strobes beam location to X − Y registers. Bit = 0 Disables strobe.

External Chip Overlap: Set when an overlap occurs with signal on CX Pin.

Sound Needs Service: Sound register has been shifted out.

Position Strobe Status: Ored of strobe has been shifted out.

Horizontal Status: Starts 20 usec. before leading edge of horizontal blanking and ends 5 usec before lagging edge of Horizontal blanking.



Rom Character Set Final Version

10/13/7?

- missing -