CDP68HC05D2

HCMOS Microcomputer

Introduction

General

The CDP68HC05D2 Microcomputer Unit (MCU) belongs to the CDP6805 Family of Microcomputers This 8-bit MCU contains on-chip oscillator CPU, RAM, ROM, I/O, and Timer The fully static design allows operation at frequencies down to DC, further reducing its already low-power consumption. It is a low-power processor designed for low-end to mid-range applications in the telecommunications, consumer, automotive, and industrial markets where very low power consumption constitutes an important factor.

The CDP68HC05D2 is supplied in a 40-lead hermetic dual-in-line side brazed ceramic package (D suffix), a 40-lead dual-in-line plastic package (E suffix), and a 44-lead Plastic Chip Carrier (Q suffix).

Specific Features

- *Typical power:*
  Operating, 25 mW
  WAIT, 7.5 mW
  STOP, 5 μW
- *Fully static operation*
- *96 bytes of on-chip RAM*
- *2176 bytes of on-chip ROM*
- *31 I/O lines*
- *12 programmable open-drain output lines*
- *On-chip oscillator for Timer*
- *2.1 MHz internal operating frequency*
- *Internal 16-bit timer*
- *Serial Peripheral Interface (SPI)*
- *External ($\overline{IRQ}$), timer, Port B, and Serial Interrupts*
- *Self check mode*
- *Single 2.5 to 6 volt supply (2-V data retention mode)*
- *RC or crystal on-chip oscillator*
- *8x8 multiply instruction*
- *True bit manipulation*
- *Indexed addressing for tables*
- *Memory mapped I/O*

Functional Pin Descriptions

V_{DD} and V_{SS}

Power is supplied to the MCU using these two pins V_{DD} is power and V_{SS} is ground.

N.C.

The pin labelled N.C. should be left disconnected.

$\overline{IRQ}$ (Maskable Interrupt Request)

IRQ is a programmable option which provides two different choices of interrupt triggering sensitivity. These options are. 1) negative edge-sensitive triggering only, or 2) both negative edge-sensitive and level-sensitive triggering. In the latter case, either type of input to the $\overline{IRQ}$ pin will produce the interrupt. The MCU completes the current instruction before it responds to the interrupt request. When the $\overline{IRQ}$ pin goes low for at least one t_{ILIH}, a logic one is latched internally to signify that an interrupt has been requested. When the MCU completes its current instruction, the interrupt latch is tested. If the interrupt latch contains a logic one, and the interrupt mask bit (I bit) in the condition code register is clear, the MCU then begins the interrupt sequence. If the option is selected to include level-sensitive triggering, then the $\overline{IRQ}$ input requires an external resistor to V_{DD} for "wire-OR" operation. See the INTERRUPTS section for more detail.

$\overline{RESET}$

The $\overline{RESET}$ input is not required for startup but can be used to reset the MCU internal state and provide an orderly software startup procedure. Refer to the RESETs section for a detailed description

3

TSM-204A

CDP68HC05D2

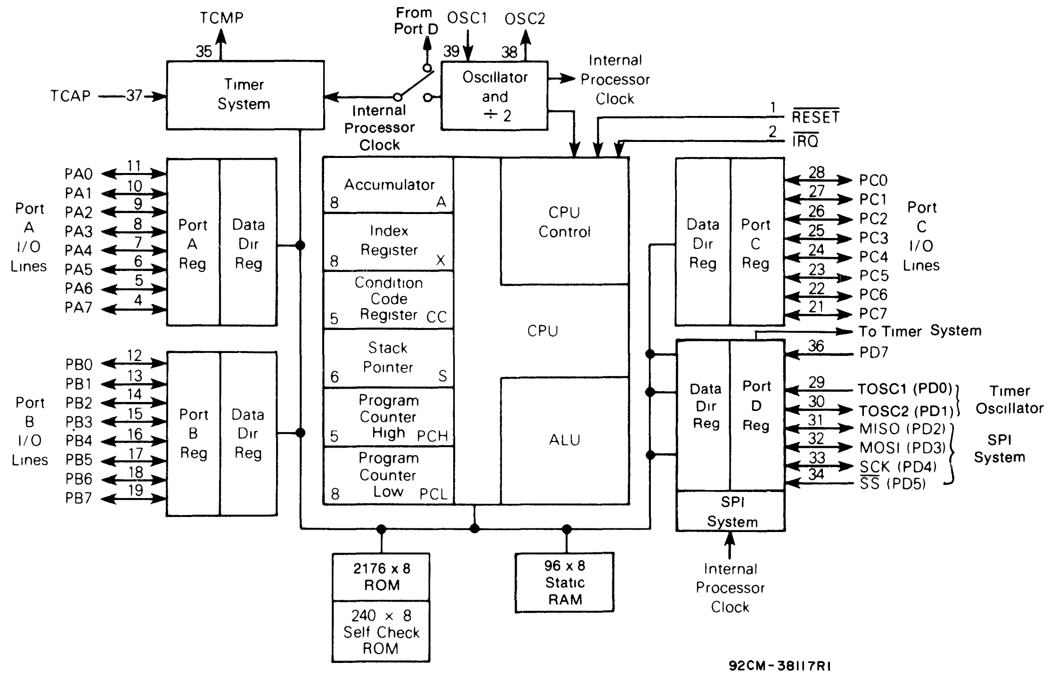


Fig. 1 — CDP68HC05D2 CMOS microcomputer block diagram.

TCAP
The TCAP input controls the input capture feature for the on-chip programmable timer system Refer to the INPUT CAPTURE REGISTER section for additional information

TCMP
The TCMP pin (35) provides an output for the output compare feature of the on-chip timer system. Refer to the OUTPUT COMPARE REGISTER section for additional information.

OSC1, OSC2
The CDP68HC05D2 can be configured to accept either a crystal input or an RC network to control the internal oscillator. This option is mask selectable. The internal clocks are derived by a divide-by-two of the internal oscillator frequency (f_{OSC}).

CRYSTAL.
The circuit shown in Fig. 2(b) is recommended when using a crystal. The internal oscillator is designed to interface with an AT-cut parallel resonant quartz crystal resonator in the frequency range specified for f_{OSC} in the control timing charts. Use of an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to the Electrical Characteristics Table.

CERAMIC RESONATOR
A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Fig. 2(b) is recommended when using a ceramic resonator. Fig. 2(a) lists the recommended capacitance and feedback resistance values. The manufacturer of the particular ceramic resonator being considered should be consulted for specific information.

RC.
If the RC oscillator option is selected, then a resistor is connected to the oscillator pins as shown in Fig. 2(d).

EXTERNAL CLOCK.
An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Fig. 2(e). An external clock may be used with either the RC or crystal oscillator option. The t_{OXOV} or t_{ILCH} specifications do not apply when using an external clock input. The equivalent specification of the external clock should be used in lieu of t_{OXOV} or T_{ILCH}.

PA0-PA7
These eight I/O input comprise port A. The state of any pin is software programmable and all port A lines are configured as input during power-on or reset. These lines are open-drain software programmable. Refer to INPUT/OUTPUT PROGRAMMABLE paragraph below for a detailed description of I/O programming.

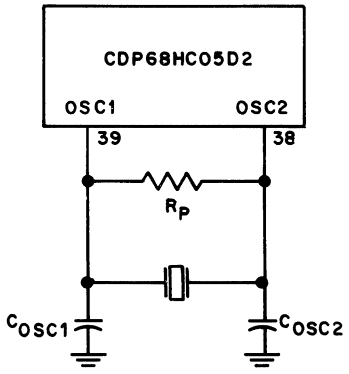
Crystal

	2 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15-40	15-30	pF
C_{OSC2}	15-30	15-25	pF
R_P	10	10	MΩ
Q	30	40	K

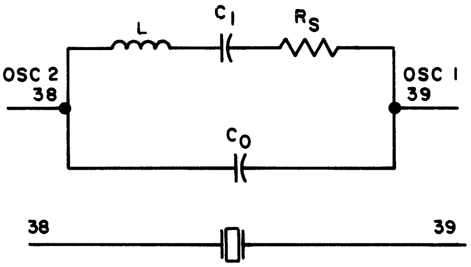
Ceramic Resonator

	2-4 MHz	Units
R_S (typical)	10	Ω
C_0	40	pF
C_1	4.3	pF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1-10	MΩ
Q	1250	—

(a) Crystal/Ceramic Resonator Parameters

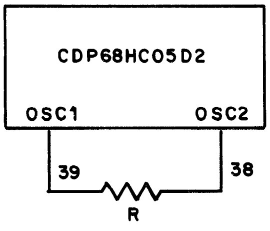


(b) Crystal Oscillator Connections

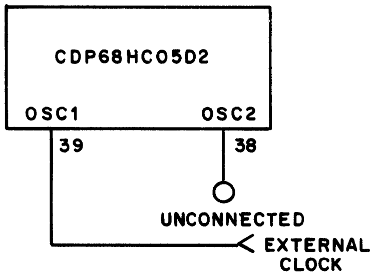


(c) Equivalent Crystal Circuit

3



(d) RC Oscillator Connections



(e) External Clock Source Connections

92CS-39366

Fig. 2 — Oscillator Connections

PB0-PB7
These eight lines comprise port B. The state of any pin is software programmable and all port B lines are configured as input during power-on or reset. These lines may be configured to generate interrupts. Refer to port B interrupt section. Refer to INPUT/OUTPUT PROGRAMMING paragraph below for a detailed description of I/O programming

PC0-PC7
These eight lines comprise port C. The state of any pin is software programmable and all port C lines are configured as input during power-on or reset. Refer to INPUT/OUTPUT PROGRAMMING paragraph below for a detailed description of I/O programming.

PD0-PD5, PD7
These seven lines comprise Port D. Four pins (PD2-PD5) are individually programmable as either inputs or outputs. PD7 is always an input line. PD0-PD5 lines are set as inputs on power-on or reset. The enabled Timer and SPI special functions listed below affect the pins on this port. PD0-PD1 (referred to as TOSC1, TOSC2) are used to control the oscillator for the timer in the external clock mode. If the external clock mode is not used, these pins are configured as inputs only. See sections EXTERNAL TIMER OSCILLATOR and SPECIAL PURPOSE PORT. MOSI is the SPI Serial Data Output (in Master Mode) MISO is the SPI Serial Data Input (in Master Mode). SCK is the clock for the SPI (configured as output in the Master Mode). SS is the Slave Select input for the SPI.

Note: It is recommended that all unused inputs (except OSC2) and I/O ports configured as inputs be tied to an appropriate logic level (e.g. either V_{DD} or V_{SS}).

Parallel I/O

The I/O register section is found in the first 32 bytes of memory and includes the following.

- Three programmable parallel ports (Ports A, B, and C).
- One port (Port D) with three input lines and four programmable lines which share its external pins with Serial Peripheral Interface (SPI) and Timer functions.

The general memory arrangement for each system has a control register, followed by a status register, followed by a data register. A CPU read of any undefined/unused bits will obtain a value of "0". The register assignment may be found in Table II.

Input/Output Programming

Parallel Ports
Ports A, B, and C may be programmed as an input or an output under software control. The direction of the pins is determined by the state of the corresponding bit in the port data direction register (DDR). Each 8-bit port has an associated 8-bit data direction register. Any port A, port B, or port C pin is configured as an output if its corresponding DDR bit is set to a logic one. A pin is configured as an input if its corresponding DDR bit is cleared to a logic zero. At power-on or reset all DDRs are cleared, which configure all port A, B, and C pins as inputs. The data direction registers are capable of being written to or read by the processor. Refer to Fig. 3 and Table I. During the programmed output state, a read of the data register actually reads the value of the output data latch and not the I/O pin.

As an option for Port A, the eight Port A outputs (PA0-PA7) can be programmed to be open drain outputs when bit 0 in the Special Port Control/Status register is set and their DDR bits are set. Also, the setting of the "Wired-OR" Mode (WOM) bit in the SPI Control Register will cause Port D lines 2-5 (when programmed as outputs) to be open drain

SPECIAL PURPOSE PORT
Port D contains four individually programmable bi-directional lines (PD2-PD5) and three input lines (PD0, PD1, and PD7). The direction of the four bi-directional lines is determined by the state of the data direction register (DDR). Each of these four lines has an associated DDR bit. The validity of a port bit is determined by whether the SPI system and external timer oscillator are enabled or disabled. When the SPI system is disabled, lines PD2-PD5 behave as normal I/O lines and the corresponding DDR bits determine whether the lines are inputs or outputs. Lines PD0 and PD1 are inputs when the external timer oscillator is not used. However, once the external timer oscillator has been enabled, PD1 will become an output-only line until the processor is reset.

A write to bits 0, 1, 6, and 7 of the Port D Data Direction Register will have no effect. A read of DDR bits 0, 1, 6, and 7 will always return zeros.

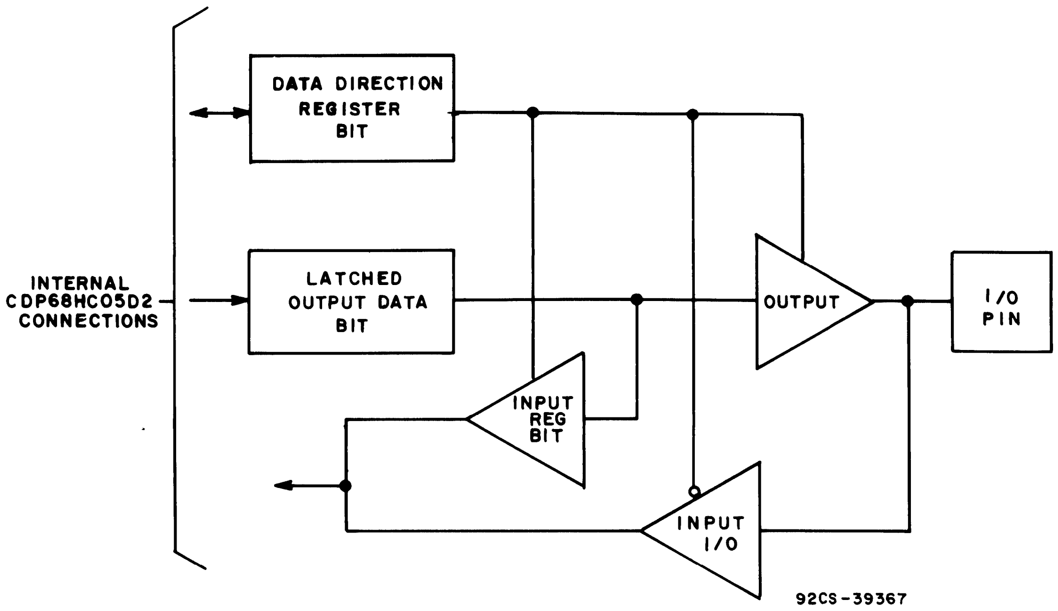
Note: When using the Serial Peripheral Interface (SPI), bit 5 of Port D is dedicated as the Slave Select (SS) input when the SPI system is enabled In SPI Slave Mode, DDR bit 5 has no meaning or effect In SPI Master Mode, DDR bit 5 determines whether Port D bit 5 is an error detect input to the SPI (DDR bit clear) or a general purpose output line (DDR bit set)

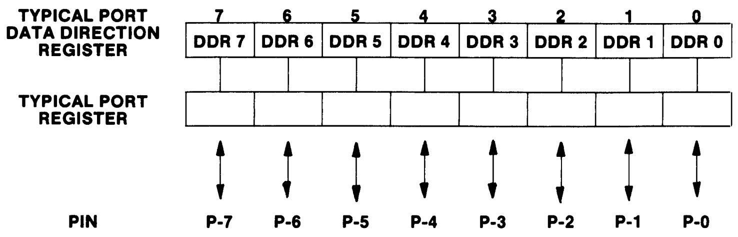
For bits 2, 3, and 4 (MISO, MOSI, and SCK), if the SPI is enabled and expects the bit to be an input, it will be an input regardless of the state of the DDR bit. If the SPI is enabled and expects the bit to be an output, it will be an output ONLY if the DDR bit is set.

Memory
The CDP68HC05D2 has a total address space of 8192 bytes. The address map is shown in Fig. 4. The CDP68HC05D2 has implemented 2550 bytes of the address locations.

The first 256 bytes of memory (page zero) is comprised of the I/O port locations, timer locations, 128 bytes of ROM and 96 bytes of RAM. The next 2048 bytes comprise the user ROM. The 16 highest address bytes contain the reset and interrupt vectors.

The stack pointer is used to address data stored on the stack. Data is stored on the stack during interrupts and subroutine calls. At power-up, the stack pointer is set to $00FF and it is decremented as data is pushed on the stack. When data is removed from the stack, the stack pointer is incremented. A maximum of 64 bytes of RAM is available for stack usage. Since most programs use only a small part of the allocated stack locations for interrupts and/or subroutine stacking purposes, the unused bytes are usable for program data storage. See Fig. 4 for details on stacking order.





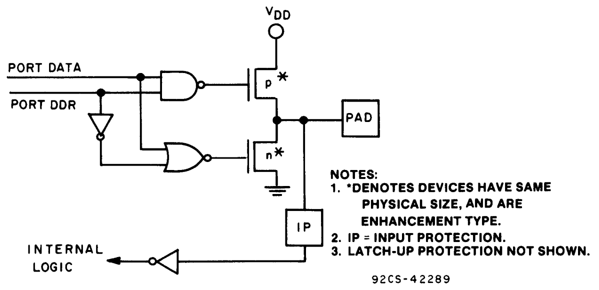

VDD
PORT DATA
PORT DDR
p*
n*
PAD
IP
INTERNAL LOGIC
NOTES:
1. *DENOTES DEVICES HAVE SAME PHYSICAL SIZE, AND ARE ENHANCEMENT TYPE.
2. IP = INPUT PROTECTION.
3. LATCH-UP PROTECTION NOT SHOWN.
92CS-42289

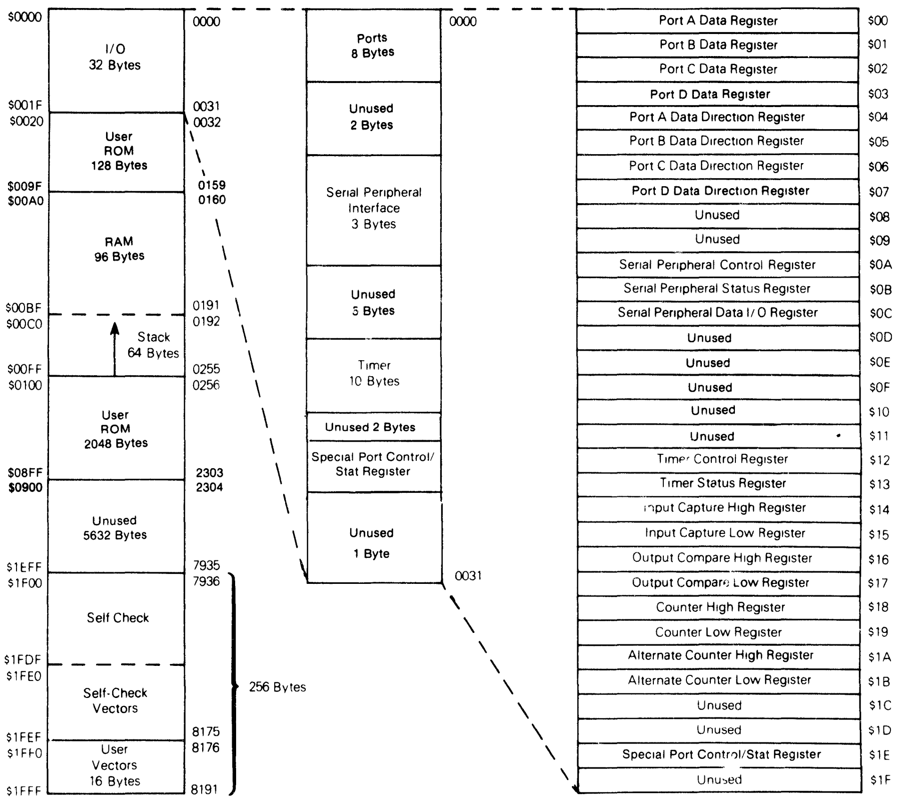
Fig. 3 - Typical Parallel Port I/O Circuitry

Table I - I/O Pin Functions

R/$\overline{W}$*	DDR	I/O Pin Function
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R/$\overline{W}$ is an internal signal.

CDP68HC05D2



92CS-38118R2

Fig. 4 - Address Map

Table II — CDP68HC05D2 I/O Registers

ADDRESS $0000-$001F	DATA 7	6	5	4	3	2	1	0
00 Port A Data								
01 Port B Data								
02 Port C Data								
03 Port D Data		*						
04 Port A DDR								
05 Port B DDR								
06 Port C DDR								
07 Port D DDR	—	—					—	—
08 Unused	—	—	—	—	—	—	—	—
09 Unused	—	—	—	—	—	—	—	—
0A SPI Control	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0
0B SPI Status	SPIF	WCOL	—	MODF	—	—	—	—
0C SPI Data								
0D Unused	—	—	—	—	—	—	—	—
0E Unused	—	—	—	—	—	—	—	—
OF Unused	—	—	—	—	—	—	—	—
10 Unused	—	—	—	—	—	—	—	—
11 Unused	—	—	—	—	—	—	—	—
12 Timer Control	ICIE	OCIE	TOIE	EOE	ECC	—	IEDG	OLVL
13 Timer Status	ICF	OCF	TOF	—	—	—	—	—
14 Capture High								
15 Capture Low								
16 Compare High								
17 Compare Low								
18 Counter High								
19 Counter Low								
1A Dual TM High								
1B Dual TM Low								
1C Unused	—	—	—	—	—	—	—	—
1D Unused	—	—	—	—	—	—	—	—
1E Special Port Cntl/STAT	PBIF	—	—	—	—	DLY	PBIE	PAOD
1F Unused	—	—	—	—	—	—	—	—

* = dedicated as TCMP output
— = unused bits

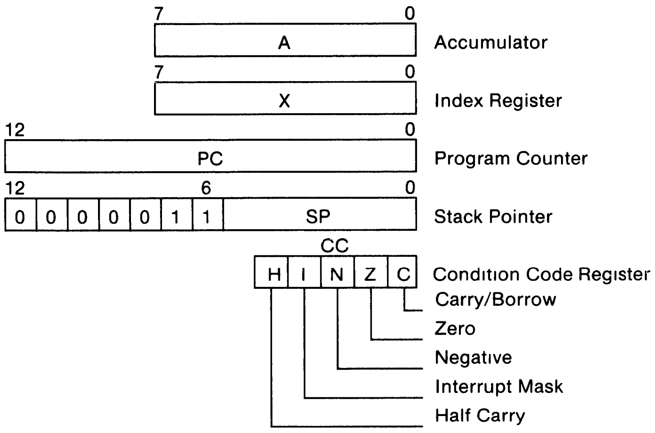
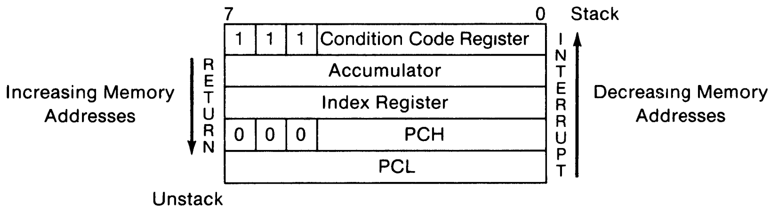


Fig. 5 - Programming model.


7
0
Stack
1 1 1 Condition Code Register
Accumulator
Index Register
0 0 0 PCH
PCL
Increasing Memory Addresses
RETURN
INTERRUPT
Decreasing Memory Addresses
Unstack

Note: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc Pulling from the stack is in the reverse order

Fig. 6 - Stacking order.

CPU Registers

The CDP68HC05D2 CPU contains five registers, as shown in the programming model of Fig. 5. The interrupt stacking order is shown in Fig. 6.

Accumulator (A)

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

Index Register (X)

The x register is an 8-bit register which is used during the indexed modes of addressing. It provides an 8-bit value which is used to create an effective address. The index register is also used for data manipulations with the read-modify-write type of instructions and as a temporary storage register when not performing addressing operations.

Program Counter (PC)

The program counter is a 13-bit register that contains the address of the next instruction to be executed by the processor.

Stack Pointer (SP)

The stack pointer is a 13-bit register containing the address of the next free locations on the push-down/pop-up stack. When accessing memory; the seven most significant bits are permanently configured to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0. The stack area of RAM is used to store the return address on subroutine calls and the machine state during interrupts During external or power-on reset, and during a reset stack pointer (RSP), instruction, the stack pointer is set to its upper limit ($00FF) Nested interrupt and/or subroutines may use up to 64 (decimal) locations. When the 64 locations are exceeded, the stack pointer wraps around and points to its upper limit ($00FF), losing the previously stored information. A subroutine call occupies two RAM bytes on the stack, while an interrupt uses five RAM bytes.

Condition Code Register (CC)

The condition code register is a 5-bit register which indicates the results of the instruction just executed as well as the state of the processor These bits can be individually tested by a program and specified action taken as a result of their state. Each bit is explained in the following paragraphs.

HALF CARRY BIT (H).

The H bit is set to a one when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instruction. The H bit is useful in binary-coded decimal subroutines.

INTERRUPT MASK BIT (I).

When the I bit is set, all interrupts are disabled. Clearing this bit enables the interrupts. If an external interrupt occurs while the I bit is set, the interrupt is latched and is processed after the I bit is next cleared; therefore, no interrupts are lost because of the I bit being set An internal interrupt can be lost if it is cleared while the I bit is set (refer to PROGRAMMABLE TIMER, SERIAL PERIPHERAL INTERFACE, and PORT B INTERRUPT sections for more information.

NEGATIVE (N).

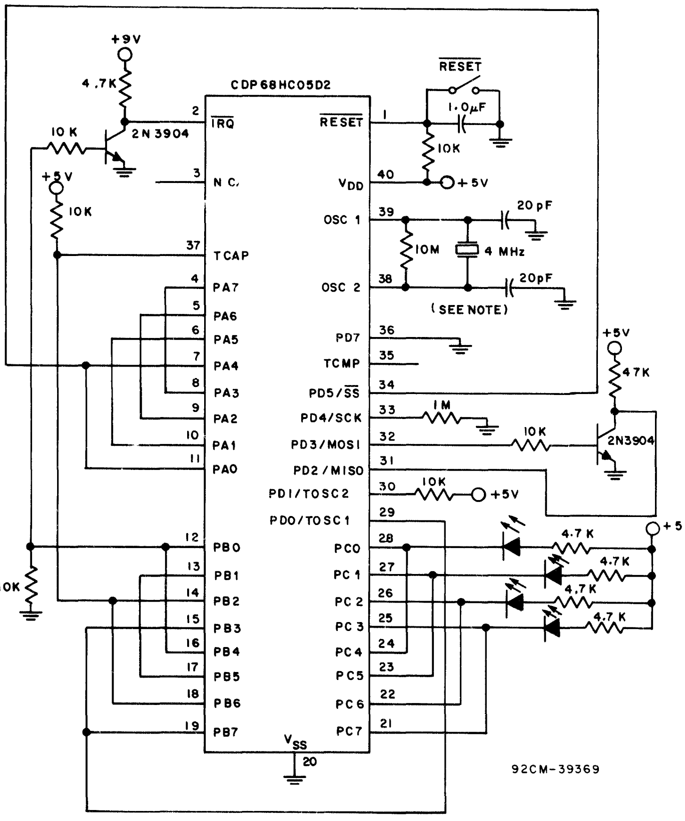
When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation is negative (bit 7 in the result is a logic one).

ZERO (Z).

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

CARRY/BORROW (C).

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



NOTE
THE RC OSCILLATOR OPTION MAY ALSO BE USED IN THIS CIRCUIT

Fig. 7 - Self-Check Circuit Schematic Diagram

Self-Check

The CDP68HC05D2 contains in mask ROM address locations \$1F00 to \$1FEF, a program designed to check the part's integrity with a minimum of support hardware The self-check capability of the CDP68HC05D2 MCU provides an internal check to determine if the device is functional. Self-check is performed using the circuit shown in the schematic diagram of Fig. 7. As shown in the diagram, port C pins PC0-PC3 are monitored (light-emitting diodes are shown but other devices could be used) for the self-check results The self-check mode is entered by applying a 9Vdc input (through a 4 7 kilohm resistor) to the $\overline{\text{IRQ}}$ pin (2), a 5Vdc input (through a 10-kilohm resistor) to the TCAP pin (37), a 5Vdc input (through a 10K resistor) to Port B, bit 2 (pin 14), and then depressing the reset switch to execute a reset After reset, the following six tests are performed automatically·

I/O — Functionally exercises ports A, B, and C
RAM — Counter test for each RAM byte
Timer — Tracks counter register and checks OCF flag
ROM — Exclusive OR with odd ones parity result
SPI — Transmission test with check for SPIF, WCOL, and MODF flags
INTERRUPTS — Tests external, timer, Port B and SPI interrupts

Self-check results (using LEDs as monitors) are shown in Table III. The following subroutines are available to user programs and do not require any external hardware.

Table III. Self-Check Results

PC3	PC2	PC1	PC0	Remarks
1	0	0	1	Bad I/O
1	0	1	0	Bad RAM
1	0	1	1	Bad Timer
1	1	0	0	Bad Port D and/or Timer Oscillator
1	1	0	1	Bad ROM
1	1	1	0	Bad SPI
1	1	1	1	Bad Interrupts or $\overline{\text{IRQ}}$ Request
Flashing				Good Device
All Others				Bad Device, Bad Port C, etc.

0 indicates LED on; 1 indicates LED is off

TIMER TEST SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. This subroutine is called at location \$1F0E. The output compare register is first set to the current timer state. Because the timer is free-running and has only a divide-by-four prescaler, each timer count cannot be tested. The test reads the timer once every 10 counts (40 cycles) and checks for correct counting. The test tracks the counter until the timer wraps around, triggering the output compare flag in the timer status register. RAM locations \$00A0 and \$00A1 are overwritten. Upon return to the user's program, X=40 If the test passed, A=0.

ROM CHECKSUM SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. This subroutine is called at location \$1F93 with RAM location \$00A3 equal to \$01 and A = 0. A short routine is set up and executed in RAM to compute a checksum of the entire ROM pattern. Upon return to the user's program, X=0 If the test passed, A=0 RAM locations \$00A0 through \$00A3 are overwritten

RESETS

The CDP68HC05D2 has two reset modes· an active low external reset pin ($\overline{\text{RESET}}$) and a power-on reset function, refer to Fig. 8.

$\overline{\text{RESET}}$ Pin

The $\overline{\text{RESET}}$ input pin is used to reset the MCU to provide an orderly software startup procedure When using the external reset mode, the $\overline{\text{RESET}}$ pin must stay low for a minimum of one and one-half t_{cyc}. The $\overline{\text{RESET}}$ pin contains an internal Schmitt Trigger as part of its input to improve noise immunity

Power-On-Reset

The power-on reset occurs when a positive transition is detected on V_{DD}. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drops in the power supply voltage There is no provision for power-down reset. The power-on circuitry provides for a delay from the time that the oscillator becomes active upon power-up or when exiting the STOP mode.

Associated with the mask programmable CPU oscillator option in the D2 is a mask option for controlling the timeout which occurs at power-on or when exiting the STOP mode. The user has a mask option of selecting a 4064 t_{cyc} delay (meant for use with the crystal oscillator option) or a 2 cycle timeout permitting faster startups with the RC oscillator mask option or external oscillator.

To permit use of an external oscillator with crystal mask option and a two cycle delay when exiting from STOP, bit 2 (DLY) of the Special Port Control/Status Register (memory location \$001E), when set, will override the 4064 cycle mask-programmable delay and force a two cycle timeout Since this bit is reset at power-on, the power-on delay will remain as mask-programmed.

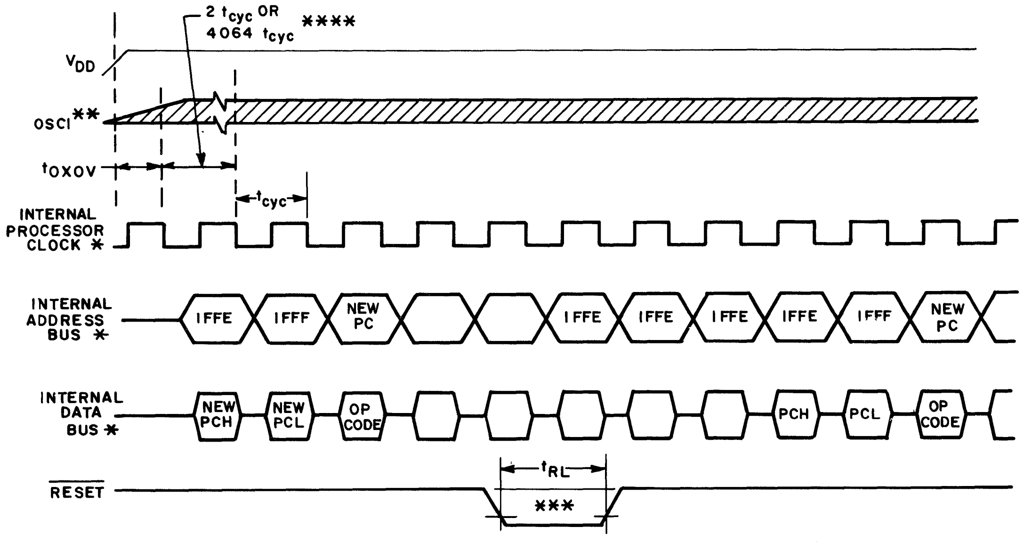
If the external $\overline{\text{RESET}}$ pin is low at the end of the delay timeout, the processor remains in the reset condition until the $\overline{\text{RESET}}$ goes high. Table IV shows the actions of the two resets on internal circuits, but not necessarily in order of occurrence.

Interrupts

Systems often require that normal processing be interrupted so that some external event may be serviced. The CDP68HC05D2 may be interrupted by one of five different methods: either one of four maskable hardware interrupts ($\overline{\text{IRQ}}$, SPI, PBINT, or Timer) and one non-maskable software interrupt (SWI). Interrupts such as Timer and SPI have several flags which will cause the interrupt. Generally, interrupt flags are located in read-only status registers, while their equivalent enable bits are located in associated control registers If the enable bit is a logic zero it blocks the interrupt from occurring but does not inhibit the flag from being set Reset clears all enable bits to preclude interrupts during the reset procedure

The general sequence for clearing an interrupt is a software sequence of first accessing the status register while the interrupt flag is set, followed by a read or write of an associated register When any of these interrupts occur, and if the enable bit is a logic one, normal processing is suspended at the end of the current instruction execution. Interrupts cause the processor registers to be saved on the stack (see Fig. 6) and the interrupt mask (I bit) set to prevent

CDP68HC05D2



* INTERNAL TIMING SIGNAL AND BUS INFORMATION NOT AVAILABLE EXTERNALLY.
** OSC1 LINE IS NOT MEANT TO REPRESENT FREQUENCY. IT IS ONLY USED TO REPRESENT TIME.
*** THE NEXT RISING EDGE OF THE INTERNAL PROCESSOR CLOCK FOLLOWING THE RISING EDGE OF $\overline{\text{RESET}}$ INITIATES THE RESET SEQUENCE.
**** DELAY IS MASK PROGRAMMABLE.

Fig. 8 - Power-On Reset and $\overline{\text{RESET}}$

Table IV. Reset Action on Internal Circuit

Condition
Timer Prescaler reset to zero state
Timer counter configured to $FFFC
Timer output compare (TCMP) bit reset to zero
All timer interrupt enable bits cleared (ICIE, OCIE, and TOIE) to disable timer interrupts. The OLVL timer bit is also cleared by reset.
All data direction registers cleared to zero (input)
Configure stack pointer to $00FF
Force internal address bus to restart vector ($1FFE-$1FFF)
Set I bit in condition code register to a logic one
Clear STOP latch*
Clear external interrupt latch
Clear WAIT latch
Disable SPI (serial output enable control bit SPE=0). Other SPI bits cleared by reset include: SPIE, MSTR, SPIF, WCOL, and MODF.
Clear serial interrupt enable bit
Place SPI system in slave mode (MSTR=0)
Timer oscillator disabled and 3-stated
CPU oscillator connected to timer
Reset Port B interrupt enable
DWOM bit reset
PAOD bit reset
Reset DLY bit in special control/status register

*Indicates that timeout still occurs with $\overline{\text{RESET}}$ pin additional interrupts. The appropriate interrupt vector then points to the starting address of the interrupt service routine (refer to Fig. 4 for vector location). Upon completion of the interrupt service routine, the RTI instruction (which is normally a part of the service routine) causes the register contents to be recovered from the stack followed by a return to normal processing. The stack order is shown in Fig. 6.

Note: The interrupt mask bit (I bit) will be cleared upon returning from the interrupt if and only if the corresponding bit stored in the stack is zero The priority of the various interrupts is as follows (highest priority to lowest priority·

RESET → * → EXT INT → TIMER → SPI → Port B

*is any instruction or the SWI service routine

A discussion of interrupts, plus a table listing vector addresses for all interrupts including reset, in the CDP68HC05D2 is provided in Table V

Table V. Vector Address for Interrupts and Reset

Register	Flag Name	Interrupts	CPU Interrupt	Vector Address
N/A	N/A	Reset	$\overline{\text{RESET}}$	\$1FFE-\$1FFF
N/A	N/A	Software	$\overline{\text{SWI}}$	\$1FFC-\$1FFD
N/A	N/A	External Interrupt	$\overline{\text{IRQ}}$	\$1FFA-\$1FFB
Timer Status	ICF	Input Capture	TIMER	\$1FF8-\$1FF9
	OCF	Output Compare		
	TOF	Timer Overflow		
SPI Status	SPIF	Transfer Complete	SPI	\$1FF4-\$1FF5
	MODF	Mode Fault		
Special Port c/s	PBIF	Port B	PB	\$1FF2-\$1FF3

Hardware Controlled Interrupt Sequence

The following three functions ($\overline{\text{RESET}}$, STOP, and WAIT) are not in the strictest sense an interrupt; however, they are acted upon in a similar manner. Flowcharts for hardware interrupts are shown in Fig. 9, and for STOP and WAIT are provided in Fig. 10. A discussion is provided below:

- A low input on the $\overline{\text{RESET}}$ input pin causes the program to vector to its starting address which is specified by the contents of memory locations \$1FFE and \$1FFF. The I bit in the condition code register is also set. Much of the MCU is configured to a known state during this type of reset as previously described in the RESET paragraph.
- STOP — The STOP instruction causes the oscillator to be turned off and the processor to "sleep" until an external interrupt ($\overline{\text{IRQ}}$), Port B interrupt, Timer interrupt (if using an external timer clock), or $\overline{\text{RESET}}$ occurs.
- WAIT — The WAIT instruction causes all processor clocks to stop, but leaves the Timer and SPI clocks running. This "rest" state of the processor can be cleared by reset, an external interrupt ($\overline{\text{IRQ}}$), Timer interrupt, SPI interrupt, or Port B interrupt. There are no special wait vectors for these individual interrupts.

Software Interrupt (SWI)

The software interrupt is an executable instruction. The action of the SWI instruction is similar to the hardware interrupts. The SWI is executed regardless of the state of the interrupt mask (I bit) in the condition code register. The interrupt service routine address is specified by the contents of memory location \$1FFC and \$1FFD.

External Interrupt

If the interrupt mask (I bit) of the condition code register has been cleared and the external interrupt pin ($\overline{\text{IRQ}}$) has gone low, then the external interrupt is recognized. When the interrupt is recognized, the current state of the CPU is pushed onto the stack and the I bit is set. This masks further interrupts until the present one is serviced. The interrupt service routine address is specified by the content of memory location \$1FFA and \$1FFB. Either a level-sensitive and negative edge-sensitive trigger, or a negative edge-sensitive only trigger are available as a mask option. Fig. 11 shows both a functional and mode timing diagram for the interrupt line. The timing diagram shows two different treatments of the interrupt line ($\overline{\text{IRQ}}$) to the processor. The first method shows single pulses on the interrupt line spaced far enough apart to be serviced. The minimum time between pulses is a function of the number of cycles required to execute the interrupt service routine plus 21 cycles. Once a pulse occurs, the next pulse should not occur until the MCU software has exited the routine (an RTI occurs). The second configuration shows several interrupt lines "wire-ORed" to form the interrupts at the processor. Thus, if after servicing one interrupt the interrupt line remains low, then the next interrupt is recognized.

Note: The internal interrupt latch is cleared in the first part of the service routine, therefore, one (and only one) external interrupt pulse could be latched during t_{ILIL} and serviced as soon as the I bit is cleared

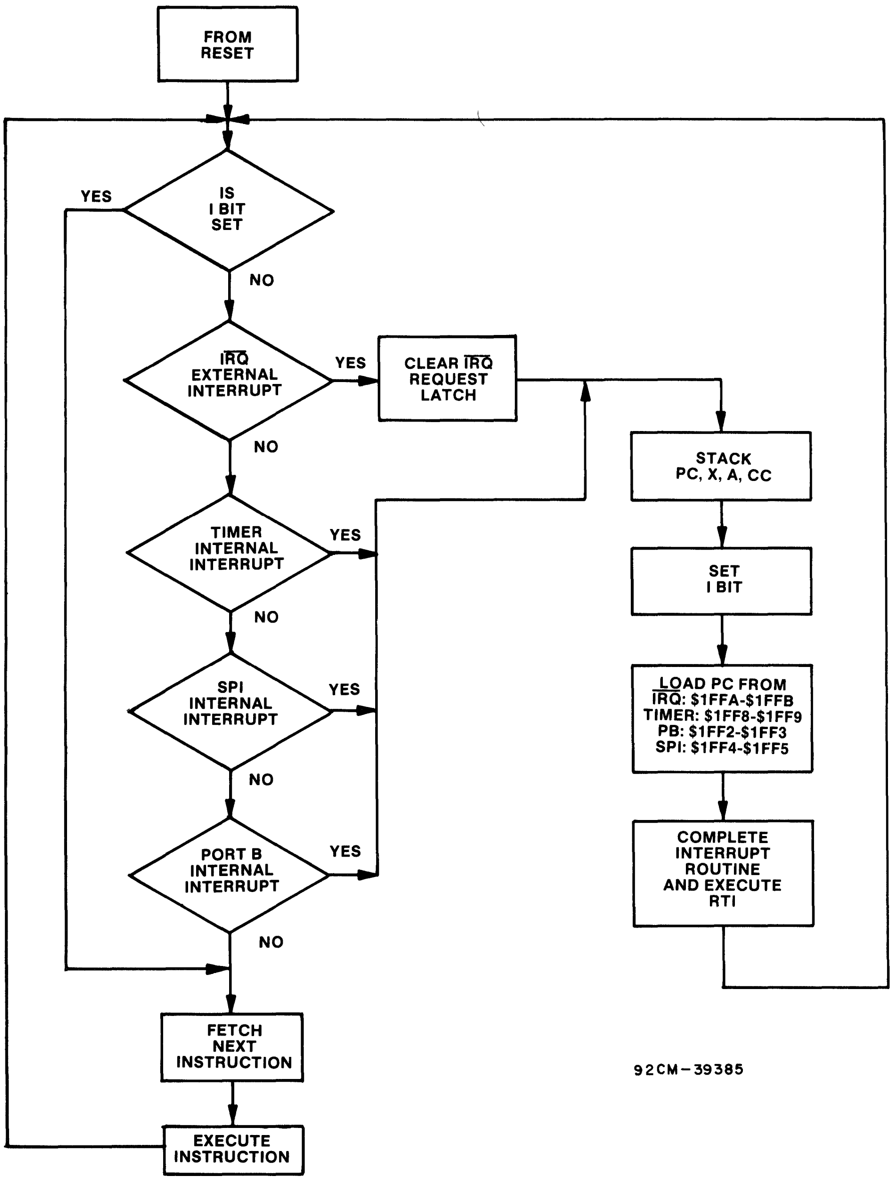


Fig. 9 - Hardware Interrupt Flowchart

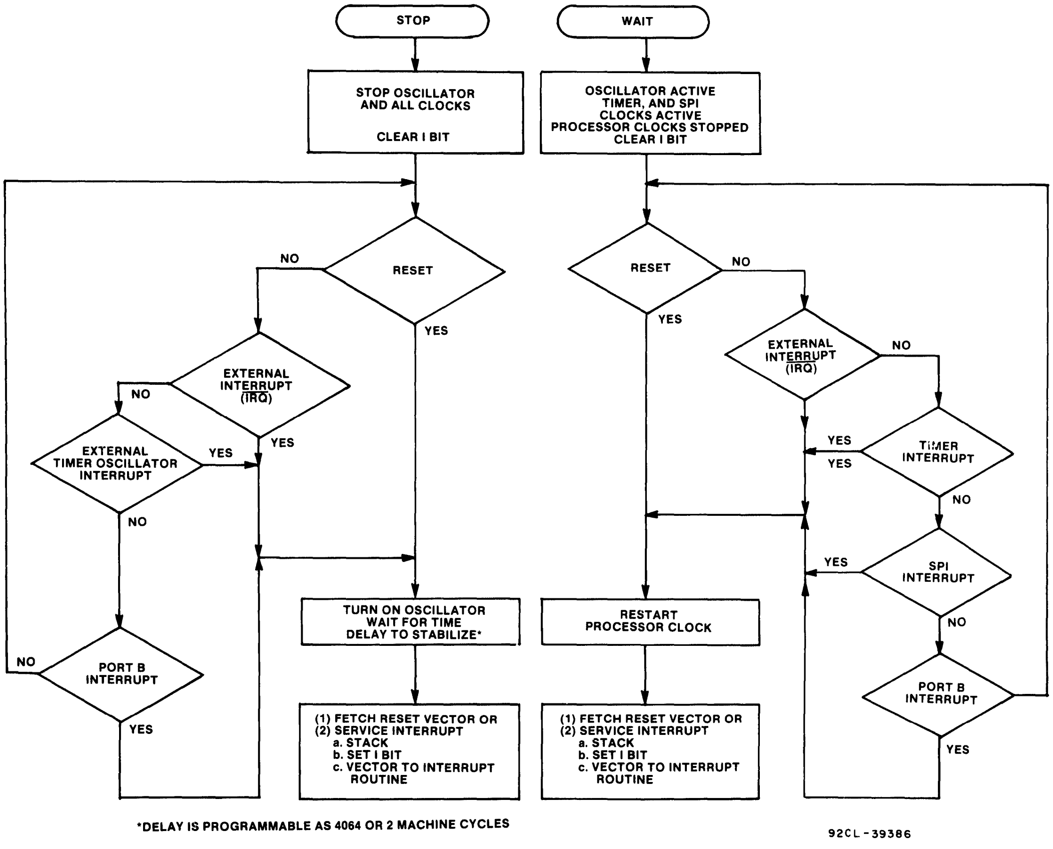

STOP
STOP OSCILLATOR AND ALL CLOCKS
CLEAR I BIT
RESET
NO
YES
EXTERNAL INTERRUPT ($\overline{IRQ}$)
EXTERNAL TIMER OSCILLATOR INTERRUPT
PORT B INTERRUPT
TURN ON OSCILLATOR WAIT FOR TIME DELAY TO STABILIZE*
(1) FETCH RESET VECTOR OR (2) SERVICE INTERRUPT a. STACK b. SET I BIT c. VECTOR TO INTERRUPT ROUTINE
*DELAY IS PROGRAMMABLE AS 4064 OR 2 MACHINE CYCLES
WAIT
OSCILLATOR ACTIVE TIMER, AND SPI CLOCKS ACTIVE PROCESSOR CLOCKS STOPPED CLEAR I BIT
TIMER INTERRUPT
SPI INTERRUPT
RESTART PROCESSOR CLOCK
92CL-39386

Fig. 10 - STOP/WAIT Flowcharts

Timer Interrupt

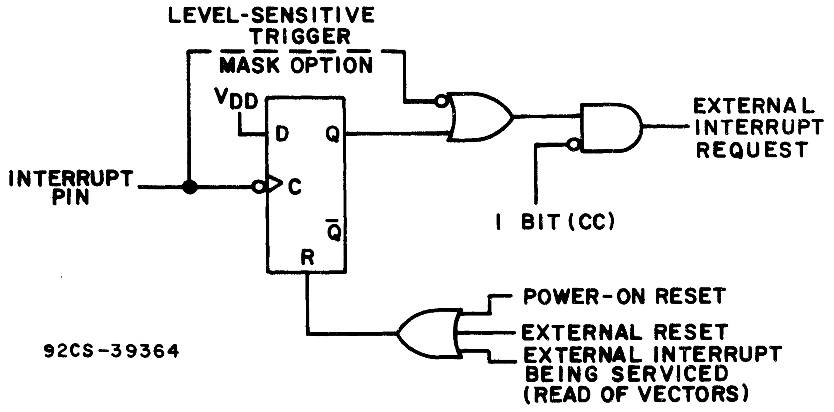
There are three different timer interrupt flags that will cause a timer interrupt whenever they are set and enabled. These three interrupt flags are found in the three most significant bits of the timer status register (TSR, location $13) and all three will vector to the same interrupt service routine ($1FF8-$1FF9). The three timer interrupt conditions are timer overflow, output compare, and input capture.

All interrupt flags have corresponding enable bits (ICIE, OCIE, and TOIE) in the timer control register (TCR, location $12). Reset clears all enable bits, thus preventing an interrupt from occurring during the reset period. The actual processor interrupt is generated only if the I bit in the condition code register is also cleared. When the interrupt is recognized, the current machine state is pushed onto the stack and I bit is set. This masks further interrupts until the present one is serviced. The interrupt service routine address is specified by the contents of memory location $1FF8 and $1FF9. The general sequence for clearing an interrupt is a software sequence of accessing the status register while the flag is set, followed by a read or write of an associated register. Refer to the PROGRAMMABLE TIMER section for additional information about the timer circuitry.

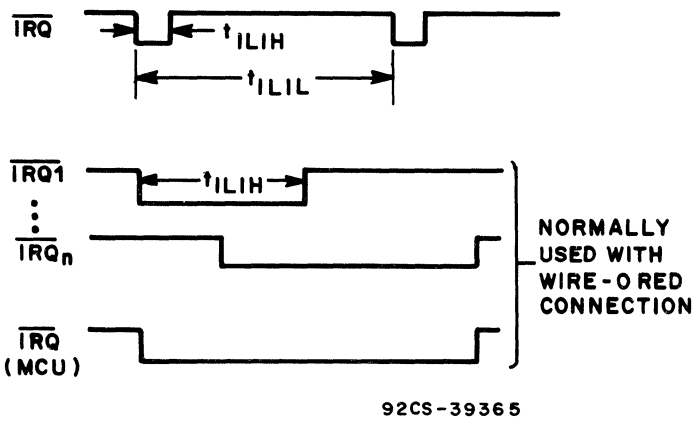
Serial Peripheral Interface (SPI) Interrupts

An interrupt in the serial peripheral interface (SPI) occurs when one of the interrupt flag bits in the serial peripheral status register (Location $0B) is set, provided the I bit in the condition code register is clear and the enable bit in the serial peripheral control register (location $0A) is enabled When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the condition code register is set. This masks further interrupts until the present one is serviced. The SPI interrupt causes the program counter to vector to memory location $1FF4 and $1FF5 which contains the starting address of the interrupt

CDP68HC05D2


LEVEL-SENSITIVE TRIGGER MASK OPTION
VDD
D
Q
C
Q̄
R
INTERRUPT PIN
EXTERNAL INTERRUPT REQUEST
I BIT (CC)
POWER-ON RESET
EXTERNAL RESET
EXTERNAL INTERRUPT BEING SERVICED (READ OF VECTORS)
92CS-39364

(a) Interrupt Function Diagram



Edge-Sensitive Trigger Condition
The minimum pulse width (t_{ILIH}) is either 125 ns (V_{DD} = 5 V) or 250 ns (V_{DD} = 3 V). The period t_{ILIL} should not be less than the number of t_{cyc} cycles it takes to execute the interrupt service routine plus 21 t_{cyc} cycles.

Level-Sensitive Trigger Condition
If after servicing an interrupt the IRQ remains low, then the next interrupt is recognized.

(b) Interrupt Mode Diagram

Fig. 11 - External Interrupt

service routine. Software in the serial peripheral interrupt service routine must determine the priority and cause of the SPI interrupt by examining the interrupt flag bits located in the SPI status register. The general sequence for clearing an interrupt is a software sequence of accessing the status register while the flag is set, followed by a read or write of an associated register. Refer to SERIAL PERIPHERAL INTERFACE section for a description of the SPI system and its interrupts.

Port B Interrupt

A Port B interrupt will occur when any one of the eight port lines (PB0-PB7) is pulled to a low level, provided the interrupt mask bit of the condition code register is clear and the enable bit (Bit 1) in the Special Port control register (Memory location $001E) is enabled. Before enabling Port B interrupts, PB0 through PB7 should be programmed as inputs, i.e., their corresponding DDR bits must be 0.

A Port B interrupt will set the Port B interrupt flag (PBIF) located in the Special Port Control/Status register (bit 7), cause the current state of the machine to be pushed onto the stack, and set the I-bit in the condition code register. This masks further interrupts until the present one is serviced. The Port B interrupt causes the Program Counter to vector to memory locations $1FF2 and $1FF3 which contain the starting address of the interrupt service routine. To clear a Port B interrupt, the user must read the Special Port Control/Status register followed by a read of Port B.

The purpose of this interrupt is to provide easy use of the PB0-PB7 lines as sensor inputs, such as in keyboard scanning. For systems where the keyboard response is not interrupt driven, this interrupt can be disabled. Programming any of these lines as outputs inhibits them from generating an interrupt.

Port B interrupts will cause an exit from the stop mode provided that the Port B interrupt enable bit is set. Port B interrupt vector is located at $1FF2, $1FF3.

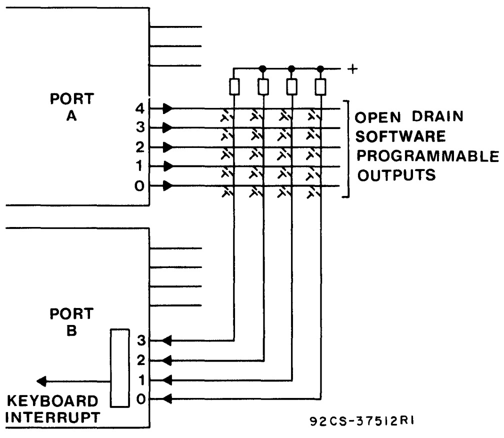


Fig. 12 - Keyboard interface.

STOP Instruction

The STOP instruction places the CDP68HC05D2 in its lowest power consumption mode. In the STOP mode the intenal oscillator is turned off, causing all internal processing to be halted; refer to Fig. 10. During the STOP mode, the I bit in the condition code register is cleared to enable external interrupts. All other registers and memory remain unaltered and all input/output lines remain unchanged. This continues until an external interrupt ($\overline{IRQ}$), port B interrupt, external timer oscillator interrupt, or reset is sensed, at which time the internal oscillator is turned on. These interrupts cause the program counter to vector to their respective interrupt vector locations ($1FFA and $1FFB, $1FF2 and $1FF3, $1FF8 and $1FF9, and $1FFE and $1FFF, respectively) which contain the starting addresses of the interrupt service routines.

WAIT Instruction

The WAIT instruction places the CDP68HC05D2 in a low power consumption mode, but the WAIT mode consumes somewhat more power than the stop MODE In the WAIT mode, the internal clock remains active, and all CPU processing is stopped; however, the programmable timer and serial peripheral interface systems remain active. Refer to Fig. 10. During the WAIT mode, the I bit in the condition code register is cleared to enable all interrupts. All other registers and memory remain unaltered and all parallel input/output lines remain unchanged. This continues until any interrupt or reset is sensed. At this time the program counter vectors to the memory location ($1FF2 through $1FFF) which contains the starting address of the interrupt or reset service routine

Data Retention Mode

The contents of RAM and CPU registers are retained at supply voltages as low as 2 Vdc. This is referred to as the data retention mode, where the data is held, but the device is not guaranteed to operate.

PROGRAMMABLE TIMER

The programmable timer, which is preceded by a fixed divide-by-four prescaler, can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from several microseconds to many seconds. A block diagram of the timer is shown in Fig. 15 and timing diagrams are shown in Figs. 16 through 19.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

Note: The I bit in the condition code register should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur. This prevents interrupts from occurring between the time that the high and low bytes are accessed

The programmable timer capabilities are provided by using the following ten addressable 8-bit registers (note the high and low represent the significance of the byte). A description of each register is provided in the following pages.

Timer Control Register (TCR) location $12,
Timer Status Register (TSR) location $13,
Input Capture High Register location $14,
Input Capture Low Register location $15,
Output Compare High Register location $16,
Output Compare Low Register location $17,
Counter High Register location $18,
Counter Low Register location $19,
Alternate Counter High Register location $1A, and
Alternate Counter Low Register location $1B.

External Timer Oscillator

In addition to clocking the CDP68HC05D2's internal 16-bit timer with the CPU clock, a separate oscillator circuit may

CDP68HC05D2

be used by connecting an RC or crystal circuit to pins 29 and 30 (TOSC1 and TOSC2). The circuits shown in Figs. 13(b) and 13(c) are recommended when using a crystal. This oscillator is designed to interface with an AT-cut parallel resonant quartz crystal resonator in the frequency range specified for f_{tosc} in the Control Timing Tables at the end of this specification. See Fig. 13(a) for the RC circuit.

When not using the external timer oscillator feature these pins function as input lines. However, once the external timer oscillator has been enabled, PD1 will become an output only line until the processor is reset.

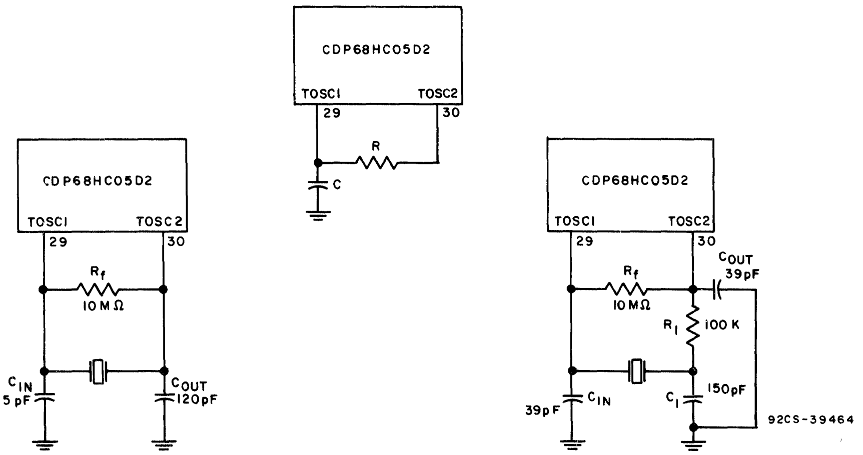
The EOE (External Oscillator Enable bit 4) and ECC (External Clock Connect bit 3) bits in the Timer Control Register control the external timer oscillator. If bit 3 (ECC) in the timer control register is set, the internal clock input to the timer is disabled and the clock to the timer is connected to the external timer oscillator This clock can be either a crystal or RC oscillator. Since this mode of operation permits the timer to continue running when the CPU is in the stop mode, timer interrupts, if enabled, will still occur and can be used to exit from the stop mode. Fig. 14 shows the timer oscillator controls. The frequency of the external oscillator must be less than one-quarter the CPU oscillator frequency.

The procedures for using this circuit are:

- Crystal Oscillator Operation — First set the EOE bit to start the crystal oscillating. When oscillation has stabilized, the ECC bit can be set to begin clocking the timer with the external timer oscillator. This time delay may vary depending upon crystal frequency and manufacturer.
- RC Oscillator Operation — When it is desired to clock the timer from an RC timer oscillator, set both the EOE and the ECC bits at the same time in order to keep power consumption minimal.
- No external timer oscillator being used — If the EOE bit is never set, the oscillator will remain in its high impedance state allowing its pins to be used as PD0 and PD1 input lines. In this case, these pins function as normal inputs and should not be left floating.
- Timer Oscillator used for event counting — Set both the EOE and ECC bits and drive the timer oscillator input pin with the event signal which is to be counted. If EOE remains reset and only ECC is set, the event signal can be connected to the timer oscillator output pin, and the input can be used as a Port D input line.

Fig. 13 - External Timer Oscillator Connections

(a) RC Oscillator Connections



(b) Crystal Oscillator connections for crystal speeds above approx. 400 KHz. The C_{in} and C_{out} values may vary depending upon crystal manufacturer.

(c) Crystal Oscillator connections for crystal speeds below approx. 400 KHz. The C_{in}, C_1 and R_1 values shown work well for most 32.768 KHz crystals; however, sizes may vary depending upon crystal frequency manufacturer.

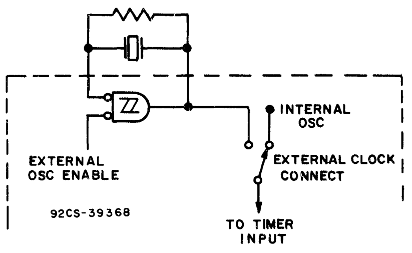


Fig. 14 - External Timer Oscillator Controls

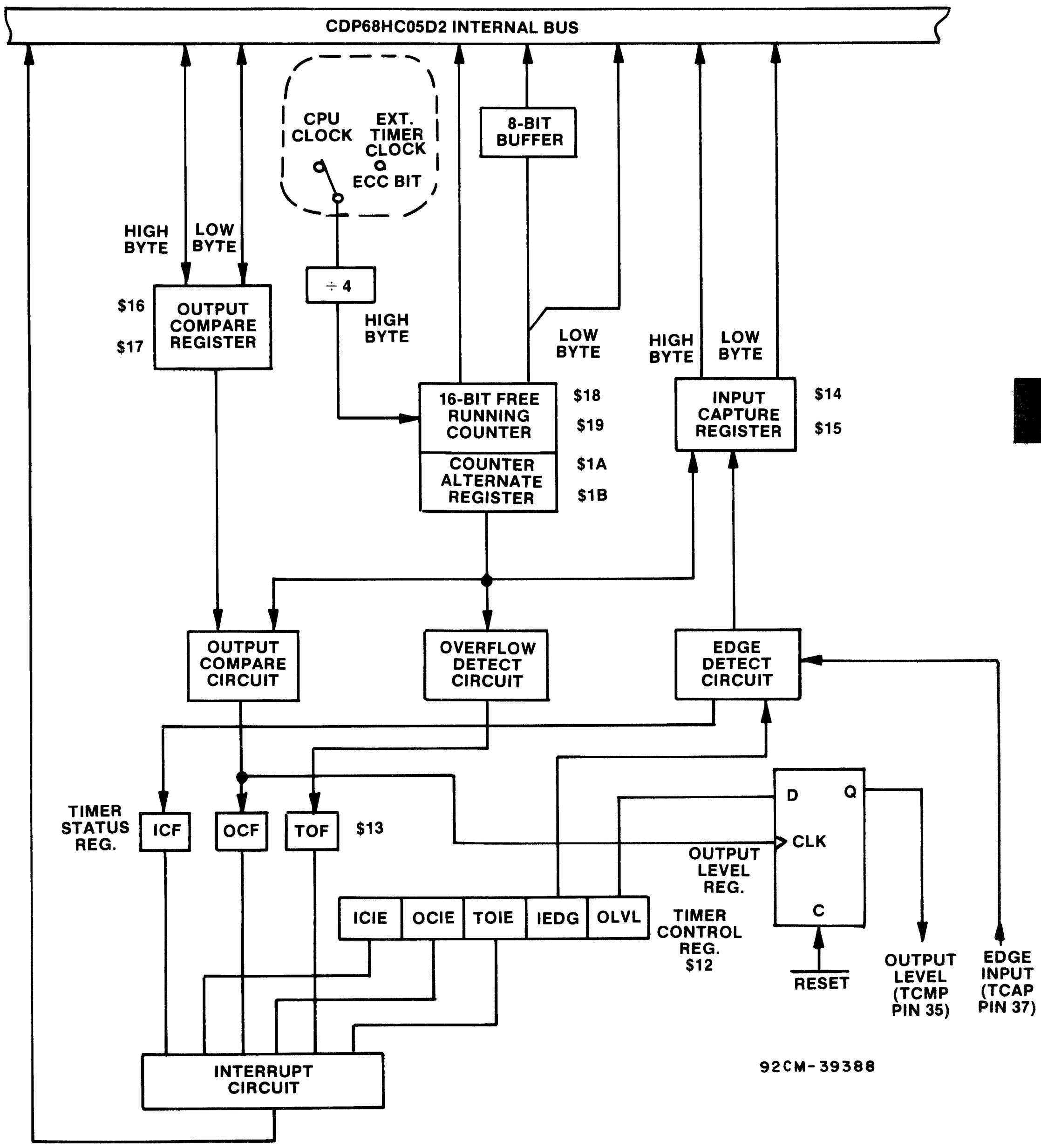
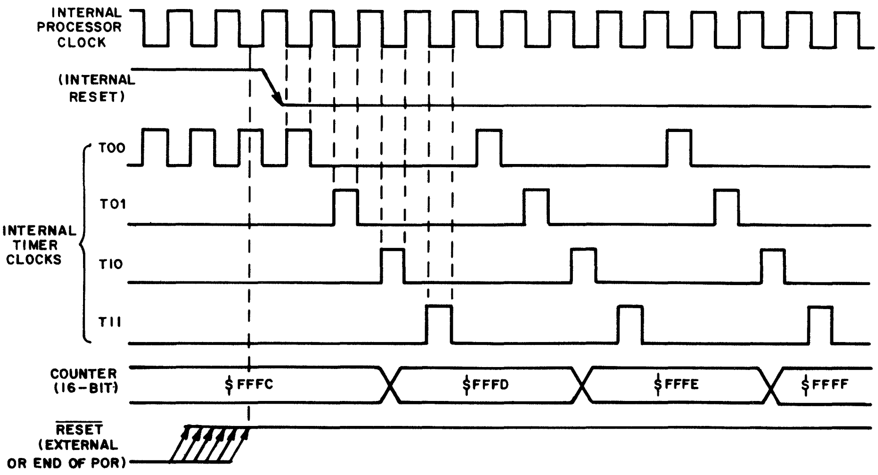


Fig. 15 - Programmable Timer Block Diagram

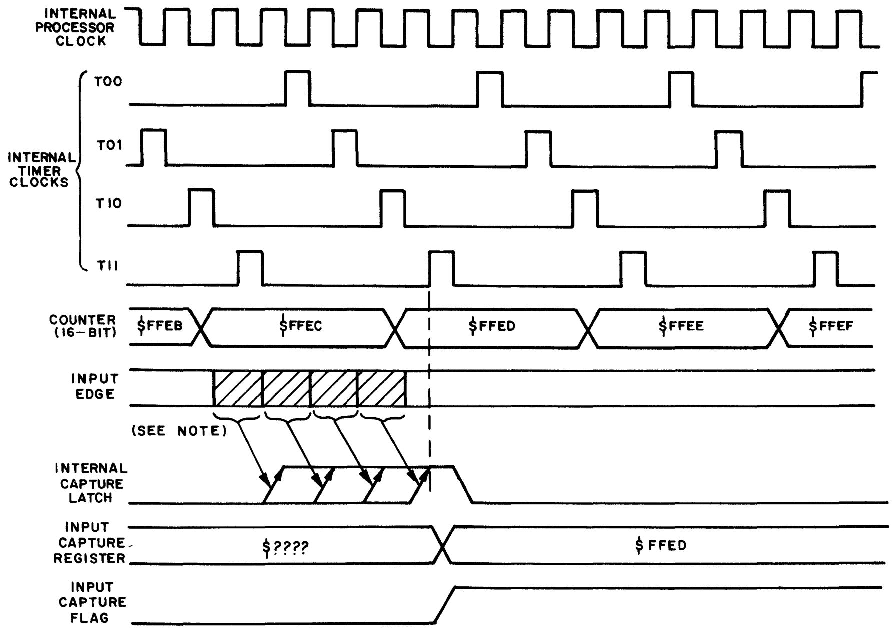
3



NOTE:
THE COUNTER REGISTER AND TIMER CONTROL REGISTER ARE THE ONLY ONES AFFECTED BY $\overline{\text{RESET}}$.

92CM-39380

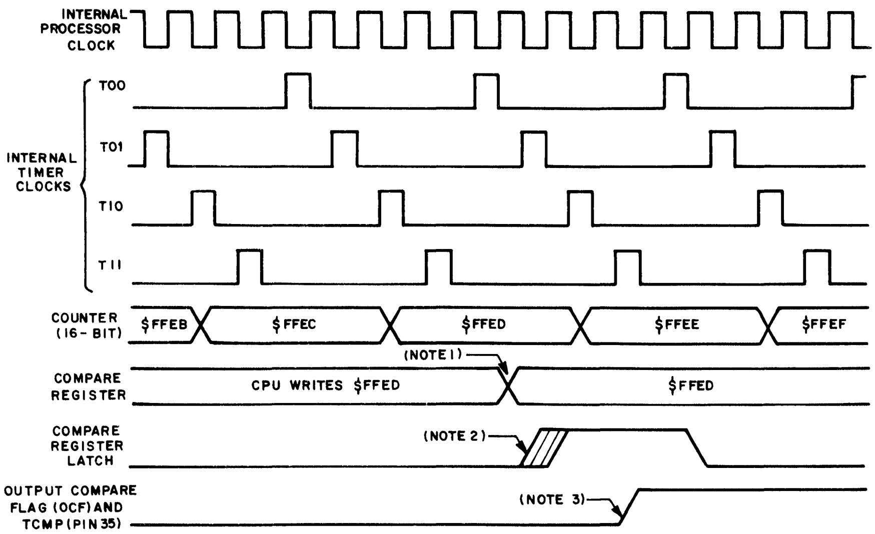
Fig. 16 - Timer State Timing Diagram For Reset



NOTE:
IF THE INPUT EDGE OCCURS IN THE SHADED AREA FROM ONE TIMER STATE T10 TO THE OTHER TIMER STATE T10, THE INPUT CAPTURE FLAG IS SET DURING THE NEXT STATE T11

92CM-39381

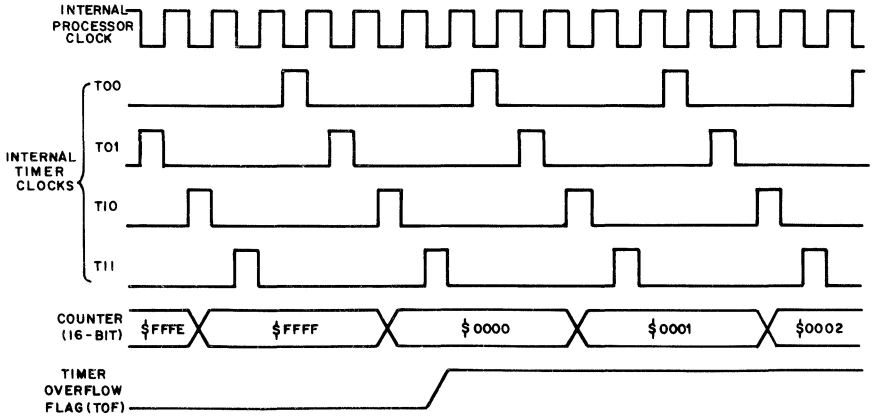
Fig. 17 - Timer State Timing Diagram For Input Capture



NOTES:
1 THE CPU WRITE TO THE COMPARE REGISTER MAY TAKE PLACE AT ANY TIME, BUT A COMPARE ONLY OCCURS AT TIMER STATE T01 THUS, A 4-CYCLE DIFFERENCE MAY EXIST BETWEEN THE WRITE TO THE COMPARE REGISTER AND THE ACTUAL COMPARE
2. INTERNAL COMPARE TAKES PLACE DURING TIMER STATE T01
3. OCF IS SET AT THE TIMER STATE T11 WHICH FOLLOWS THE COMPARISON MATCH ($FFED IN THIS EXAMPLE)

92CM-39378

Fig. 18 - Timer State Timing Diagram For Output Compare



NOTE·
THE TOF BIT IS SET AT TIMER STATE T11 (TRANSITION OF COUNTER FROM $FFFF TO $0000). IT IS CLEARED BY A READ OF THE TIMER STATUS REGISTER DURING THE INTERNAL PROCESSOR CLOCK HIGH TIME FOLLOWED BY A READ OF THE COUNTER LOW REGISTER.

92CM-39379

Fig. 19 - Timer State Diagram For Timer Overflow

Counter

The key element in the programmable timer is a 16-bit free-running counter, or counter register, preceded by a prescaler which divides the internal processor clock by four The prescaler gives the timer a resolution of 2.0 microseconds if the internal processor clock is 2.0 MHz. The counter is clocked to increasing values during the low portion of the internal processor clock. Software can read the counter at any time without affecting its value

The double-byte free-running counter can be read from either of two locations $18-$19 (called counter register at this location), or $1A-$1B (counter alternate register at this location) A read sequence containing only a read of the least significant byte of the free-running counter ($19, $1B) will receive the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte ($18, $1A) it causes the least significant byte ($19, $1B) to be transferred to a buffer. This buffer value remains fixed after the first most significant byte "read" even if the user reads the most significant byte several times. This buffer is accessed when reading the free-running counter or counter alternate register least significant byte ($19 or $1B), and thus completes a read sequence of the total counter value. Note that in reading either the free-running counter or counter alternate register, if the most significant byte is read, the least significant byte must also be read in order to complete the sequence.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on-reset (POR), the counter is also configured to $FFFC and begins running after the oscillator startup delay. Because the free-running counter is 16 bits preceded by a fixed divide-by-four prescaler, the value in the free-running counter repeats every 262,144 MPU internal processor clock cycles. When the counter rolls over from $FFFF to $0000, the timer overflow flag (TOF) bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

Output Compare Register

The output compare register is a 16-bit register, which is made up of two 8-bit registers at locations $16 (most significant byte) and $17 (least significant byte). The output compare register can be used for several purposes, such as, controlling an output waveform or indicating when a period of time has elapsed The output compare register is unique in that all bits are readable and writeable and are not altered by the timer hardware Reset does not affect the contents of this register and if the compare function is not utilized, the two bytes of the output compare register can be used as storage locations.

The contents of the output compare register are compared with the contents of the free-running counter once during every four internal processor clocks. If a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output level (OLVL) bit is clocked (by the output compare circuit pulse) to an output level register. The values in the output compare register and the output level bit should be changed after each successful comparison in order to control an output waveform or establish a new elapsed timeout An interrupt can also accompany a successful output compare provided the corresponding interrupt enable bit, OCIE, is set.

After a processor write cycle to the output compare register containing the most significant byte ($16), the output compare function is inhibited until the least significant byte ($17) is also written. The user must write both bytes (locations) if the most significant byte is written first A write made only to the least significant byte ($17) will not inhibit the compare function The free-running counter is updated every four internal processor clock cycles due to the internal prescaler. The minimum time required to update the output compare register is a function of the software program rather than the internal program

A processor write may be made to either byte of the output compare register without affecting the other byte. The output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.

Because neither the output compare flag (OCF bit) nor output compare register is affected by reset, care must be exercised when initializing the output compare function with software. The following procedure is recommended:

(1) Write the high byte of the output compare register to inhibit further compares until the low byte is written.
(2) Read the timer status register to arm the OCF if it is already set.
(3) Write the output compare register low byte to enable the output compare function with the flag clear.

The advantage of this procedure is to prevent the OCF bit from being set between the time it is read and the write to the output compare register. A software example is shown below.

```
B7  16  STA  OCMPHI   INHIBIT OUTPUT COMPARE
B6  13  LDA  TSTAT    ARM OCF BIT IF SET
BF  17  STX  OCMPLD   READY FOR NEXT COMPARE
```

Input Capture Register

The two 8-bit registers which make up the 16-bit input capture register are read-only and are used to latch the value of the free-running counter after a defined transition is sensed by the corresponding input capture edge detector. The level transition which triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal processor clock preceding the external **transition (refer to timing diagram shown in Fig. 17). This** delay is required for external synchronization. Resolution is affected by the prescaler allowing the timer to only increment every four internal processor clock cycles

The free-running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is set or clear. The input capture register always contains the free-running counter value which corresponds to the most recent input capture.

After a read of the most significant byte of the input capture register ($14), counter transfer is inhibited until the least significant byte ($15) of the input capture register is also read. This characteristic forces the minimum pulse period attainable to be determined by the time used in the capture software routine and its interaction with the main program. A polling routine using instructions such as BRSET, BRA, LDA, STA, INCX, CMPX, and BEG might take 34 machine cycles to complete The free-running counter increments every four internal processor clock cycles due to the prescaler. A read of the least significant byte ($15) of the input capture register does not inhibit the free-running counter transfer. Again, minimum pulse periods are ones which allow software to read the least significant byte ($15) and perform the needed operations. There is no conflict between the read of the input capture register and the free-running counter since they occur on opposite edges of the internal processor clock.

Timer Control Register (TCR)

The timer control register (TCR, location $12) is an 8-bit read/write register which contains seven control bits. Three of these bits control interrupts associated with each of the three flag bits found in the timer status register (discussed below). The other four bits control: 1) which edge is significant to the input capture edge detector (i.e., negative or positive), 2) the next value to be clocked to the output level register in response to a successful output compare, 3) the source of the timer clock, and 4) whether the external timer oscillator is enabled. The timer control register and the free-running counter are the only sections of the timer affected by reset. The TCMP pin is forced low during external reset and stays low until a valid compare changes it to a high. The timer control register is illustrated below followed by a definition of each bit.

7	6	5	4	3	2	1	0	
ICIE	OCIE	TOIE	EOE	ECC	0	IEDG	OLVL	$12

B7, ICIE	If the input capture interrupt enable (ICIE) bit is set, a timer interrupt is enabled when the ICF status flag (in the timer status register) is set. If the ICIE bit is clear, the interrupt is inhibited. The ICIE bit is cleared by reset.
B6, OCIE	If the output compare interrupt enable (OCIE) bit is set, a timer interrupt is enabled whenever the OCF status flag is set. If the OCIE bit is clear, the interrupt is inhibited. The OCIE bit is cleared by reset.
B5, TOIE	If the timer overflow interrupt enable (TOIE) bit is set, a timer interrupt is enabled whenever the TOF status flag (in the timer status register) is set If the TOIE bit is clear, the interrupt is inhibited. The TOIE bit is cleared by reset.
B4, EOE	External Oscillator Enable — If set, the external timer oscillator is enabled. If it is then cleared, the inverter between pins 29 and 30 is prevented from switching and cannot be used in a crystal or RC oscillator. This bit is cleared by reset which configures both TOSC1 and TOSC2 as inputs.
B3, ECC	If the external clock connect (ECC) is set, the internal clock input to the timer is disabled and the timer oscillator is connected to the input to the timer It is cleared by reset. Accuracy of the timer count is not guaranteed while this bit is switched.
B1, IEDG	The value of the input edge (IEDG) bit determines which level transition on pin 37 will trigger a free-running counter transfer to the input capture register. Reset clears the IEDG bit. 0 = negative edge 1 = positive edge
B0, OLVL	The value of the output level (OLVL) bit is clocked into the output level register by the next successful output compare and will appear at pin 35. This bit and the output level register are cleared by reset. 0 = low output 1 = high output

Timer Status Register (TSR)

The timer status register (TSR) is an 8-bit register of which the three most significant bits contain read-only status information. These three bits indicate the following:

1. A proper transition has taken place at pin 37 with an accompanying transfer of the free-running counter contents to the intput capture register,
2. A match has been found between the free-running counter and the output compare register, and
3. A free-running counter transition from $FFFF to $0000 has been sensed (timer overflow)

The timer status register is illustrated below followed by a definition of each bit. Refer to timing diagrams shown in **Fig. 16, 17, and 18 for timing relationship to the timer status register bits.**

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	0	0	0	0	0	$13

B7, ICF	The input capture flag (ICF) is set when a proper edge has been sensed by the input capture edge detector. It is cleared by a processor read of the timer status register (with ICF set) followed by reading the low byte ($15) of the input capture register. Reset does not affect the input compare flag.
B6, OCF	The output compare flag (OCF) is set when the output compare register contents matches the contents of the free-running counter. The OCF is cleared by reading the timer status register (with the OCF set) and then writing to the low byte ($17) of the output compare register. Reset does not affect the output compare flag.
B5, TOF	The timer overflow flag (TOF) bit is set by a transition of the free-running counter from $FFFF to $0000 It is cleared by reading the timer status register (with TOF set) followed by a read of the free-running counter least significant byte ($19). Reset does not affect the TOF bit.

Reading the timer status register satisfies the first condition required to clear any status bits which happened to be set during the access. The only remaining step is to provide an access of the register which is associated with the status bit. Typically, this presents no problem for the input capture and output compare functions.

A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if: 1) the timer status register is read when TOF is set, and 2) the least significant byte of the free-running counter is read but not for the purpose of servicing the flag. The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this

3

alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

During STOP and WAIT instructions, the programmable timer functions as follows if using the CPU clock: during the wait mode, the timer continues to operate normally and may generate an interrupt to trigger the CPU out of the wait state; during the stop mode, the timer holds at its current state, retaining all data, and resumes operation from this point when an external interrupt is received. If using an external timer oscillator the timer will continue to count and generate interrupts.

Serial Peripheral Interface (SPI)

The Serial Peripheral Interface (SPI) is a four wire synchronous serial communication system with separate wires for input data, output data, clock and slave select. A master MCU, which produces the clocking signal, initiates the exchange of data bytes with a slave MCU or peripheral device such as an LCD display driver or an A/D converter. A diagram of the control, status, and data registers may be found in the section labelled "Registers". The SPI system registers are found at addresses $000A-$000C. The SPI output drivers may be switched off to allow the user access to external pins for use as parallel inputs to Port D. Upon power-up or reset the SPI output drivers will be initialized in the off state. The serial system enable bit which controls the output drivers and other functional inhibits is the SPE bit found in the serial control register.

Fig. 20 illustrates two different system configurations Fig. 20a represents a system of five different MCUs in which there are one master and four slaves (0, 1, 2, 3). In this system four basic lines (signals) are required for the MOSI (master out, slave in), MISO (master in, slave out), SCK (serial clock), and $\overline{SS}$ (slave select) lines. **Fig. 20**b represents a system of three MCUs in which each MCU is capable of being a master or a slave. The SPI interface is well-suited for multiprocessor communications.

Features

- Full duplex, three-wire synchronous transfers
- Master or slave operation
- 1.05 MHz (maximum) master bit frequency
- 2.1 MHz (maximum) slave bit frequency
- Four programmable master bit rates
- Programmable clock polarity and phase
- End of transmission interrupt flag
- Write collision flag protection
- Master-Master mode fault protection capability

Signal Description

The four basic signals (MOSI, MISO, SCK, and $\overline{SS}$) discussed above are described in the following paragraphs. Each signal function is described for both the master and slave mode.

Master Out Slave In (MOSI)

The MOSI pin is configured as a data output in a master (mode) device and as a data input in a slave (mode) device. In this manner data is transferred serially from a master to a slave on this line; most significant bit first, least significant bit last. The timing diagrams of **Fig. 21** summarize the SPI timing diagram and show the relationship between data and clock (SCK). As shown in **Fig. 21** four possible timing relationships may be chosen by using control bits CPOL and CPHA. The master device always allows data to be applied on the MOSI line a half-cycle before the clock edge (SCK) in order for the slave device to latch the data.

Note: Both the slave device(s) and a master device must be programmed to similar timing modes for proper data transfer

When the master device transmits data to a second (slave) device via the MOSI line, the slave device responds by sending data to the master device via the MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal (one which is provided by the master device). Thus, the byte transmitted is replaced by the byte received and eliminates the need for separate transmit-empty and receiver-full status bits. A single status bit (SPIF) is used to signify that the I/O operation is complete.

Configuration of the MOSI pin is a function of the MSTR bit in the serial peripheral control register (SPCR, loction $0A). Setting the MSTR bit will place the device in the Master mode and cause the MOSI pin to be an output.

Note: The Port D Data Direction Register bit 3 must be set for the MOSI pin to transfer data in the Master mode

Master In Slave Out (MISO)

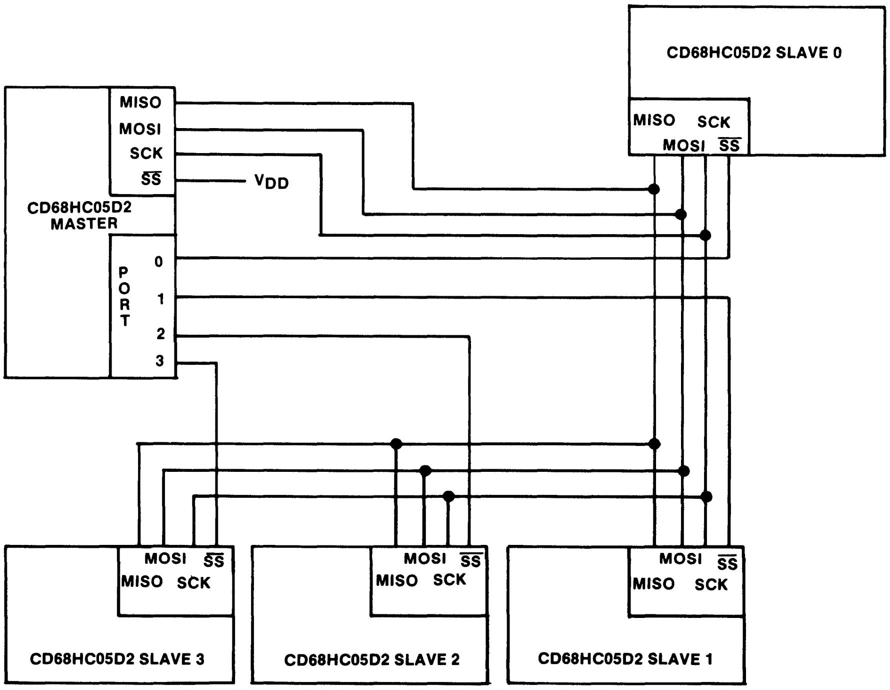
The MISO pin is configured as an input in a master (mode) device and as an output in a slave (mode) device In this manner data is transferred serially from a slave to a master on this line; most significant bit first, least significant bit last. The MISO pin of a slave device is placed in the high-impedance state if it is not selected by the master; i.e., its $\overline{SS}$ pin is a logic one. The timing diagram of **Fig. 21** shows the relationship between data and clock (SCK) As shown in **Fig. 21**, four possible timing relationships may be chosen by using control bits CPOL and CPHA. The master device always allows data to be applied on the MOSI line a half-cycle before the clock edge (SCK) in order for the slave device to latch the data.

Note: The slave device (s) and a master device must be programmed to similar timing modes for proper data transfer

When the master device transmits data to a slave device via the MOSI line, the slave device responds by sending data to the master device via the MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal (one which is provided by the master device). Thus, the byte transmitted is replaced by the byte received and eliminates the need for separate transmit-empty and receiver-full status bits. A single status bit (SPIF) in the serial peripheral status register (SPSR, location $0B) is used to signify that the I/O operation is complete.

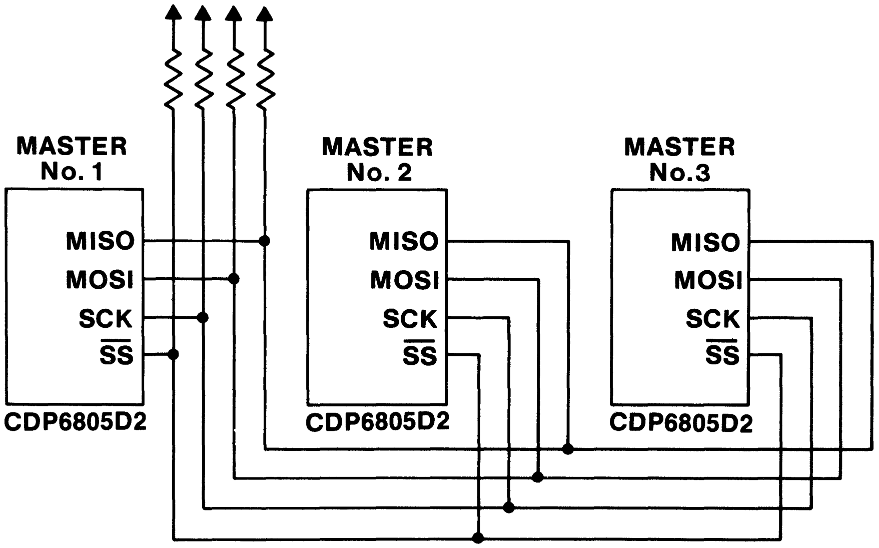
In the master device, the MSTR control bit in the serial peripheral control register (SPCR, location $0A) is set to a logic one (by the program) to allow the master device to receive data on its MISO pin. In the slave device, its MISO pin is enabled by the logic level of the $\overline{SS}$ pin; i.e., if $\overline{SS}$=1 then the MISO pin is placed in the high-impedance state, whereas, if $\overline{SS}$=0 the MISO pin is an output for the slave device

Note: The Port D Data Direction Register bit 2 must be set for the MISO pin to transfer data in the slave mode



(a) Single Master, Four Slaves

92CM-39384



(b) Multimaster System

92CS-37494

Fig. 20 - Master-Slave System Configuration

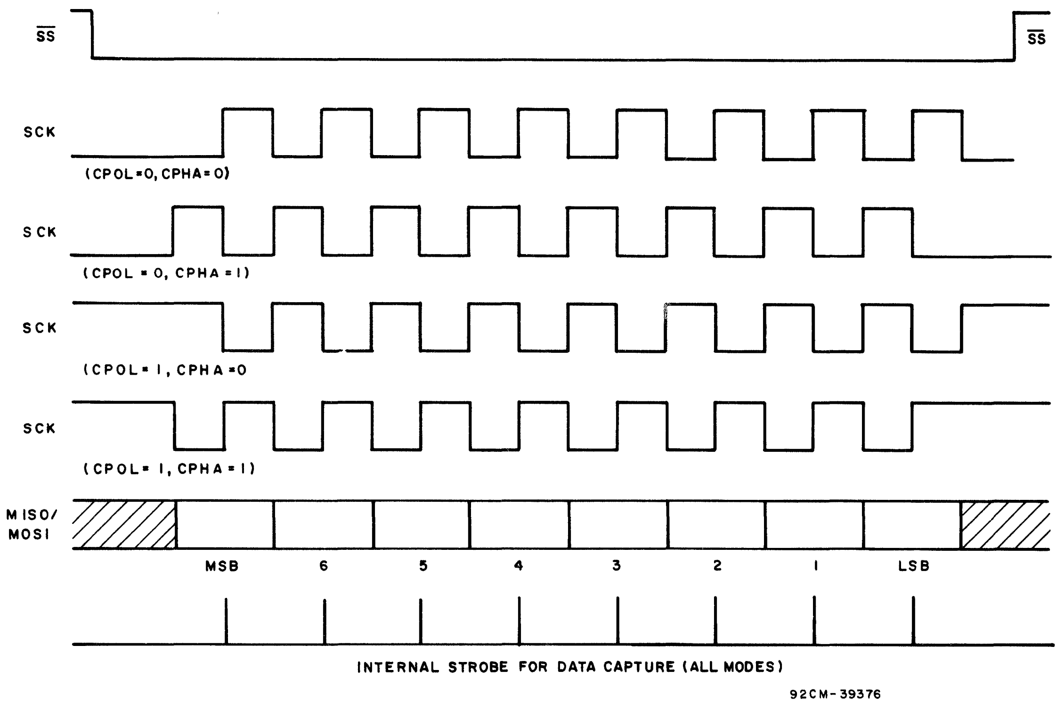


Fig. 21 - Data Clock Timing Diagram

Slave Select ($\overline{SS}$)

In the slave mode the slave select ($\overline{SS}$) pin is an input (PD5, pin 34), which receives an active low signal that is generated by the master device to enable slave device(s) to accept data. To ensure that data will be accepted by a slave device, the $\overline{SS}$ signal line must be a logic low prior to occurrence of SCK (system clock) and must remain low until after the last **(eighth) SCK cycle. Fig. 21 illustrates the relationship be**tween SCK and the data for two different level combinations of CPHA, when $\overline{SS}$ is pulled low. These are: 1) with CPHA=1 of 0, the first bit of data is applied to the MISO line for transfer, and 2) when CPHA = 0 the slave device is prevented from writing to its data register. Refer to the WCOL status flag in the serial peripheral status register (location $0B) description for further information on the effects that the SS input and CPHA control bit have on the I/O data register. A high level SS signal forces the MISO (master in, slave out) line to the high-impedance state. Also, SCK and the MOSI (master out, slave in) line are ignored by a slave device when its $\overline{SS}$ signal is high.

When a device is a master, it monitors its $\overline{SS}$ signal for a logic low, provided that Port D bit 5 is cleared. See Note. The master device will become a slave device any time its $\overline{SS}$ signal is detected low. This ensures that there is only one master controlling the $\overline{SS}$ line for a particular system. When the $\overline{SS}$ line is detected low, it clears the MSTR control bit (serial peripheral control register, location $0A). Also, control bit SPE in the serial peripheral control register is cleared which causes the serial peripheral interface (SPI) to be disabled (port D SPI pins become inputs). The MODF flag bit in the serial peripheral status register (location $0B) is also set to indicate to the master device that another device is attempting to become a master. Two devices attempting to be outputs are normally the result of a software error; however, a system could be configured which would contain a default master which would automatically "take over" and restart the system.

Note: In the master mode Port D DDR bit 5 determines whether Port D bit 5 ($\overline{SS}$) is an error detect input to the SPI (DDR bit 5 clear) or a general-purpose output line (DDR bit 5 set), that can be used to strobe the $\overline{SS}$ lines of slaves

Serial Clock (SCK)

The serial clock is used to synchronize the movement of data both in and out of the device through its MOSI and MISO pins. The master and slave devices are capable of exchanging a data byte of information during a sequence of eight clock pulses. Since the SCK is generated by the master device, the SCK line becomes an input on all slave devices and synchronizes slave data transfer. The type of clock and its relationship to data are controlled by the CPOL and CPHA bits in the serial peripheral control register **(location $0A) discussed below. Refer to Fig. 21 for timing.**

The master device generates the SCK through a circuit driven by the internal processor clock. Two bits (SPR0 and SPR1) in the serial peripheral control register (location $0A) of the master device select the clock rate. The master device uses the SCK to latch incoming slave device data on the MISO line and shifts out data to the slave on the MOSI line. Both master and slave devices must be operated in the same timing mode as controlled by the CPOL and CPHA bit in the serial peripheral control register. In the slave device, SPR0 and SPR1 have no effect on the operation of the Serial Peripheral Interface. Timing is shown in Fig. 21.

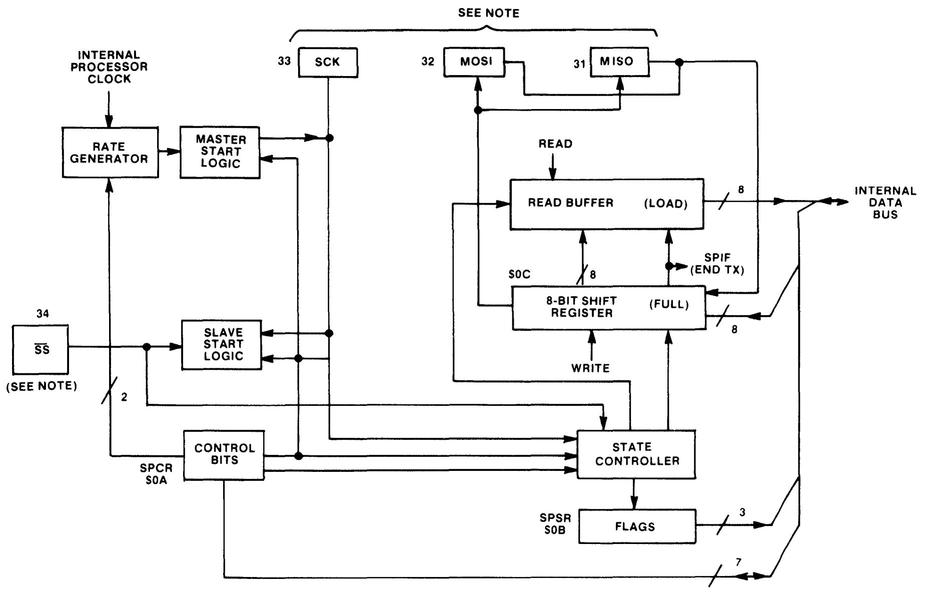
Note: The Port D Data Direction Register bit 4 must be set for the SCK pin to generate (output) a SCK signal

Functional Description

A block diagram of the serial peripheral interface (SPI) is shown in Fig. 22. In a master configuration the master start logic receives an input from the CPU (in the form of a write to the SPI rate generator) and originates the system clock (SCK) based on the internal processor clock. This clock is also used internally to control the state controller as well as the 8-bit shift register. As a master device, data is parallel loaded into the 8-bit shift register (from the internal bus) during a write cycle and then shifted out serially to the MOSI pin for application to the slave device(s). During a read cycle, data is applied serially from a slave device via the MISO pin to the 8-bit shift register After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer and then is made available to the internal data bus during a CPU read cycle

In a slave configuration, the slave start logic receives a logic low (from a master device) at the $\overline{SS}$ pin and a system clock input (from the same master device) at the SCK pin Thus, the slave is synchronized with the master Data from the master is received serially at the slave MOSI pin and loads the 8-bit shift register After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer and then is made available to the internal data bus during a CPU read cycle. During a write cycle, data is parallel loaded into the 8-bit shift register from the internal data bus and then shifted out serially to the MISO pin for application to the master device

Fig. 23 illustrates the MOSI, MISO, and SCK master-slave interconnections. Note that in Fig. 23 the master $\overline{SS}$ pin is tied to a logic high and the slave $\overline{SS}$ pin is a logic low. Fig 21a provides a larger system connection for these same pins. Note that in Fig. 20(a), all $\overline{SS}$ pins are connected to a port pin of a master/slave device. In this case any of the devices can be a slave.



NOTES
THE $\overline{SS}$, SCK, MOSI, AND MISO ARE EXTERNAL PINS WHICH PROVIDE THE FOLLOWING FUNCTIONS
(a) MOSI-PROVIDES SERIAL OUTPUT TO SLAVE UNIT(S) WHEN DEVICE IS CONFIGURED AS A MASTER RECEIVES SERIAL INPUT FROM MASTER UNIT WHEN DEVICE IS CONFIGURED AS A SLAVE UNIT
(b) MISO-RECEIVES SERIAL INPUT FROM SLAVE UNIT(S) WHEN DEVICE IS CONFIGURED AS A MASTER PROVIDES SERIAL OUTPUT TO MASTER WHEN DEVICE IS CONFIGURED AS A SLAVE UNIT
(c) SCK -PROVIDES SYSTEM CLOCK WHEN DEVICE IS CONFIGURED AS A MASTER UNIT RECEIVES SYSTEM CLOCK WHEN DEVICE IS CONFIGURED AS A SLAVE UNIT
(d) $\overline{SS}$ -PROVIDES A LOGIC LOW TO SELECT A SLAVE DEVICE FOR A TRANSFER WITH A MASTER DEVICE

92CM-39390

Fig. 22 - Serial Peripheral Interface Block Diagram

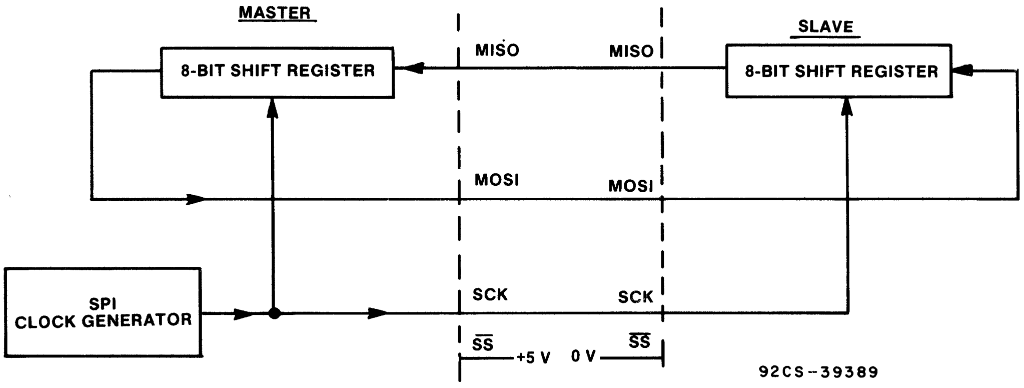


Fig. 23 - Serial Peripheral Interface Master-Slave Interconnection

Registers

There are three registers in the serial parallel interface which provide control, status, and data storage functions These registers, which include the serial peripheral control register (SPCR, location $0A), serial peripheral status register (SPSR, location $0B), and serial peripheral data I/O register (SPDR, location $0C) are described below.

Note: In addition, the Port D Data Direction Register (DDR) must be properly configured See note in the section labelled "Input/Output Programming-Special-Purpose Port"

Serial Peripheral Control Register (SPCR)

7	6	5	4	3	2	1	0	
SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	$0A

The serial peripheral control register bits are defined as follows.

B7, SPIE When the serial peripheral interrupt enable bit is high, it allows the occurrence of a processor interrupt, and forces the proper vector to be loaded into the program counter if the serial peripheral status register flag bit (SPIF and/or MODF) is set to a logic one. It does not inhibit the setting of a status bit. The SPIE bit is cleared by reset.

B6, SPE When the serial peripheral output enable control bit is set, all output drive is applied to the external pins and the system is enabled. When the SPE bit is set, it enables the SPI system by connecting it to the external pins thus allowing it to interface with the external SPI bus. The pins that are defined as output depend on which mode (master or slave) the device is in Because the SPE bit is cleared by reset, the SPI system is not connected to the external pins upon reset.

B5, DWOM The Port D Wire-OR Mode bit controls the output buffers for Port D bits 2 through 5 If DWOM=1, the four Port D output buffers behave as open-drain outputs. If DWOM=0, the four Port D output buffers operate as normal CMOS outputs. DWOM is cleared by reset.

B4, MSTR The master bit determines whether the device is a master or a slave. If the MSTR bit is a logic zero it indicates a slave device and a logic one denotes a master device. If the master mode is selected, the function of the SCK pin changes from an input to an output and the function of the MISO and MOSI pins are reversed. This allows the user to wire device pins MISO to MISO, and MOSI to MOSI and SCK to SCK without incident. The MSTR bit is cleared by reset, therefore, the device is always placed in the slave mode during reset.

B3, CPOL The clock polarity bit controls the normal or steady state value of the clock when data is not being transferred. The CPOL bit affects both the master and slave modes It must be used in conjunction with the clock phase control bit (CPHA) to produce the wanted clock-data relationship between a master and a slave device. When the CPOL bit is a logic zero, it produces a steady state low value at the SCK pin of the master device If the CPOL bit is a logic one, a high value is produced at the SCK pin of the master device when data is not being transferred. The CPOL bit is not affected by **reset. Refer to Fig. 21.**

B2, CPHA The clock phase bit controls the relationship between the data on the MISO and MOSI pins and the clock produced or received at the SCK pin. This control has effect in both the master and slave modes. It must be used in conjunction with the clock polarity control bit (CPOL) to produce the wanted clock-data relation. The CPHA bit in general selects the clock edge which captures data and allows it to change states. It has its greatest impact on the first bit transmitted (MSB) in that it does or does not allow a clock transition before the first data capture edge. The CPHA bit is not affected by **reset. Refer to Fig. 21.**

B1, SPR1
B0, SPR0 These two serial peripheral rate bits select one of four baud rates to be used as SCK if the device is a master; however, they have no effect in the slave mode. The slave device is

capable of shifting data in and out at a maximum rate which is equal to the CPU clock (maximum = 2.1 MHz). A rate table is given below for the generation of the SCK from the master The SPR1 and SPR0 bits are not affected by reset

SPR1	SPR0	Internal Processor Clock Divide By
0	0	2 •
0	1	4
1	0	16
1	1	32

Serial Peripheral Status Register (SPSR)

7	6	5	4	3	2	1	0	
SPIF	WCOL	—	MODF	—	—	—	—	$0B

The status flags which generate a serial peripheral interface (SPI) interrupt will not be blocked by the SPIE control bit in the serial peripheral control register; however, the interrupt will be blocked The WCOL bit does not cause an interrupt. The serial peripheral status register bits are defined as follows:

B7, SPIF The serial peripheral data transfer flag bit notifies the user that a data transfer between the device and an external device has been completed. With the completion of the data transfer, SPIF is set, and if SPIE is set, a serial peripheral interrupt (SPI) is generated. During the clock cycle that SPIF is being set, a copy of the received data byte in the shift register is moved to a buffer When the data register is read, it is the buffer that is read. During an overrun condition, when the master device has sent several bytes of data and the slave device has not responded to the first SPIF, only the first byte sent is contained in the receiver buffer and all other bytes are lost.

The transfer of data is initiated by the master device writing its serial peripheral data register.

Clearing the SPIF bit is accomplished by a software sequence of accessing the serial peripheral status register while SPIF is set and followed by a write to or a read of the serial peripheral data register. While SPIF is set, all writes to the serial peripheral data register are inhibited until the proper clearing sequence is followed. This occurs in the master device. In the slave device, SPIF can be cleared (using a similar sequence) during a second transmission, however, it must be cleared before the second SPIF in order to prevent an overrun condition The SPIF bit is cleared by reset

B6, WCOL The function of the write collision status bit is to notify the user that an attempt was made to write the serial peripheral data register while a data transfer was taking place with an external device The transfer continues uninterrupted; therefore, a write will be unsuccessful. A "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the MCU operation. If a "write collision" occurs, WCOL is set but no SPI interrupt is generated. The WCOL bit is a status flag only.

Clearing the WCOL bit is accomplished by a software sequence of accessing the serial peripheral status register while WCOL is set, followed by 1) a read of the serial peripheral data register prior to the SPIF bit being set, or 2) a read or write of the serial peripheral data register after the SPIF bit is set. A write to the serial peripheral data register (SPDR) prior to the SPIF bit being set, will result in generation of another WCOL status flag. Both the SPIF and WCOL bits will be cleared in the same sequence. If a second transfer has started while trying to clear (the previously set) SPIF and WCOL bits with a clearing sequence containing a write to the serial peripheral data register, only the SPIF bit will be cleared.

A collision of a write to the serial peripheral data register while an external data transfer is taking place can occur in both the master mode and the slave mode, although with the proper programming the master device should have sufficient information to preclude this collision.

Collision in the master device is defined as a write of the serial peripheral data register while the internal rate clock (SCK) is in the process of transfer. The signal on the SS pin is always high on the master device.

A collision in a slave device is defined in two separate modes. One problem arises in a slave device when the CPHA control bit is a logic zero. When CPHA is a logic zero, data is latched with the occurrence of the first clock transition. The slave device does not have any way of knowing when that transition will occur; therefore, the slave device collision occurs when it attempts to write the serial peripheral data register after its $\overline{SS}$ pin has been pulled low. The $\overline{SS}$ pin of the slave device freezes the data in its serial peripheral data register and does not allow it to be altered if the CPHA bit is a logic zero. The master device must raise the $\overline{SS}$ pin of the slave device high between each byte it transfers to the slave device

The second collision mode is defined for the state of CPHA control bit being a logic one. With the CPHA bit set, the slave device will be receiving a clock (SCK) edge prior to the latch of the first data transfer. This first clock edge will freeze the data in the slave device I/O register and allow the MSB onto the external MISO pin of the slave device. The $\overline{SS}$ pin low state enables the slave device but the drive onto the MISO pin does not take place until the first data transfer clock edge. The WCOL bit will only be set if the I/O register is accessed while a transfer is taking place. By definition of the second collision mode, a master device might hold a slave device $\overline{SS}$ pin low during a transfer of several bytes of data without a problem.

A special case of WCOL occurs in the slave device. This happens when the master device starts a transfer sequence (an edge of SCK for CPHA=1; or an active $\overline{SS}$ transition for CPHA=0) at the same time the slave device CPU is writing to its serial peripheral interface data register. In this case it is assumed that the data byte written (in the slave device serial peripheral interface) is lost and the contents of the slave device read buffer become the byte that is transferred. Because the master device receives back the last byte transmitted, the master device can detect that a fatal WCOL occurred.

Because the slave device is operating asynchronously with the master device, the WCOL bit may be used as an indicator of a collision occurrence. This helps alleviate the user from a strict real-time programming effort The WCOL bit is cleared by reset.

Bit 4 MODF The function of the mode fault flag (MODF) is defined for the master mode device. If the device is a slave device, the MODF bit will be prevented from toggling from a logic zero to a logic one; however, this does not prevent the device from being in the slave mode with the MODF bit set The MODF bit is normally a logic zero and is set only when the master device has its $\overline{SS}$ pin pulled low. Toggling the MODF bit to a logic one affects the internal serial peripheral interface (SPI) system in the following ways.

1. MODF is set and SPI interrupt is generated if SPIE=1
2. The SPE bit is forced to a logic zero This blocks all output drive from the device, disabled the SPI system.
3 .The MSTR bit is forced to a logic zero, thus forcing the device into the slave mode.

Clearing the MODF is accomplished by a software sequence of accessing the serial peripheral status register while MODF is set followed by a write to the serial peripheral control register. Control bits SPE and MSTR may be restored to their original set state during this clearing sequence or after the MODF bit has been cleared. Hardware does not allow the user to set the SPE and MSTR bit while MODF is a logic one unless it is during the proper clearing sequence. The MODF flag bit indicates that there might have been a multi-master conflict for system control and allows a proper exit from system operation to a reset or default system state. The MODF bit is cleared by reset.

Serial Peripheral Data I/O Register (SPDR)

7	6	5	4	3	2	1	0
Serial Peripheral Data I/O Register							

The serial peripheral data I/O register is used to transmit and receive data on the serial bus. Only a write to this register will initiate transmission/reception of another byte and this will only occur in the master device. A slave device writing to its data I/O register will not initiate a transmission. At the completion of transmitting a byte of data, the SPIF status bit is set in both the master and slave devices. A write or read of the serial peripheral data I/O register, after accessing the serial peripheral status register with SPIF set, will clear SPIF.

During the clock cycle that the SPIF bit is being set, a copy of the received data byte in the shift register is being moved to a buffer. When the user reads the serial peripheral data I/O register, the buffer is actually being read. During an overrun condition, when the master device has sent several bytes of data and the slave device has not internally responded to clear the first SPIF, only the first byte is contained in the receive buffer of the slave device; all others are lost The user may read the buffer at any time The first SPIF must be cleared by the time a second transfer of data from the shift register to the read buffer is initiated or an overrun condition will exist.

A write to the serial peripheral data I/O register is not buffered and places data directly into the shift register for transmission

The ability to access the serial peripheral data I/O register is limited when a transmission is taking place. It is important to read the discussion defining the WCOL and SPIF status bits to understand the limits on using the serial peripheral data I/O register.

Serial Peripheral Interface (SPI) System Considerations

There are two types of SPI systems: single master system and multi-master systems. **Figure 20** illustrates both of these systems and a discussion of each is provided below.

Figure 20 a illustrates how a typical single master system may be configured, using a CDP6805 CMOS Family device as the master and four CDP6805 CMOS Family devices as slaves As shown, the MOSI, MISO, and SCK pins are all wired to equivalent pins on each of the five devices. The master device generates the SCK clock, the slave devices all receive it. Because the CDP6805 CMOS master device is the bus master, it internally controls the function of its MOSI and MISO lines, thus writing data to the slave devices on the MOSI and reading data from the slave devices on the MISO lines. The master device selects the individual slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the slave devices. A slave device is selected when the master device pulls its $\overline{SS}$ pin low. The $\overline{SS}$ pins are pulled high during reset because the master device ports will be forced to be inputs at that time, thus disabling the slave devices. Notice that the slave devices do not have to be enabled in a mutually exclusive fashion except to prevent bus contention on the MISO line. For example, three slave devices enabled for a transfer are permissible if only one has the capability of being read by the master. An example of this is a write to several display drivers to clear a display with a single I/O operation. To ensure that proper data transmission is occurring between the master device and a slave device, the master device may have the slave device respond with a previously received data byte (this data byte could be inverted or at least be a byte that is different from the last one sent by the master device). The master device will always receive the previous byte back from the slave device if all MISO and MOSI lines are connected and the slave has not written to its data I/O register. Other transmission security methods might be defined using ports for handshake lines or data bytes with command fields.

A multi-master system may also be configured by the user. A system of this type is shown in **Figure 20b**. An exchange of master control could be implemented by an exchange of code messages through the serial peripheral interface system. The major device control that plays a part in this system is the MSTR bit in the serial peripheral control register and the MODF bit in the serial peripheral status register. Note that the DWOM bit would also be set to prevent bus contention. For additional information on this configuration and SPI in general, refer to RCA Application Note ICAN 7264 entitled "Versatile Serial Protocol for a Microcomputer-Peripheral Interface."

Effects of Stop and Wait Modes on the Timer and Serial System

The STOP and WAIT instructions have different effects on the programmable timer and serial peripheral interface (SPI) system These different effects are discussed separately below

Stop Mode

When the processor executes the STOP instruction, the internal oscillator is turned off This halts all internal CPU processing and the serial peripheral interface The programmable timer will only continue to count if an external timer oscillator is used. The only way for the MCU to "wake up" from the stop mode is by receipt of an external interrupt (logic low on $\overline{\text{IRQ}}$ pin), an external timer oscillator interrupt, a Port B interrupt or by the detection of a reset (logic low on $\overline{\text{RESET}}$ pin or a power-on reset). The effects of the stop mode on each of the MCU systems (Timer and SPI) are described separately.

Timer During Stop Mode

When the MCU enters the STOP mode, the timer will continue to count and generate interrupts if using an external timer oscillator. If using the CPU clock to clock the timer, the timer counter stops counting (the internal processor clock is stopped) and remains at that particular count value until the stop mode is exited by an interrupt (if exited by reset the counter is forced to $FFFC). If the stop mode is exited by an external low on the $\overline{\text{IRQ}}$ pin, then the counter resumes from its stopped value as if nothing had happened. Another feature of the programmable timer, in the stop mode, is that if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuitry is armed This action does not set any timer flags or "wake up" the MCU, but when the MCU does "wake up" there will be an active input capture flag (and data) from that first valid edge which occurred during the stop mode If the stop mode is exited by an external reset (logic low on $\overline{\text{RESET}}$ pin), then no such input capture flag or data action takes place even if there was a valid input capture edge (at the TCAP pin) during the MCU stop mode

SPI During Stop Mode

When the MCU enters the stop mode, the baud rate generator which drives the SPI shuts down. This essentially stops all master mode SPI operation, thus the master SPI is unable to transmit or receive any data If the STOP instruction is executed during an SPI transfer, that transfer is halted until the MCU exits the stop mode (provided it is an exit resulting from a logic low on the $\overline{\text{IRQ}}$ pin) If the stop mode is exited by a reset, then the appropriate control/status bits are cleared and the SPI is disabled If the device is in the slave mode when the STOP instruction is executed, the slave SPI will still operate. It can still accept data and clock information in addition to transmitting its own data back to a master device.

At the end of a possible transmission with a slave SPI in the STOP mode, no flags are set until a logic low $\overline{\text{IRQ}}$ input results in an MCU "wake up". Caution should be observed when operating the SPI (as a slave) during the stop mode because none of the protection circuitry (write collision, mode fault, etc.) is active.

It should also be noted that when the MCU enters the stop mode all enabled output drivers (TDO, TCMP, MISO, MOSI, and SCK ports) remain active and any sourcing currents from these outputs will be part of the total supply current required by the device

Wait Mode

When the MCU enters the wait mode, the CPU clock is halted All CPU action is suspended; however, the timer and SPI systems remain active. In fact an interrupt from the timer or SPI (in addition to a logic low on the $\overline{\text{IRQ}}$ or $\overline{\text{RESET}}$ pins or a Port B interrupt, if enabled) causes the processor to exit the wait mode. Since the three systems mentioned above operate as they do in the normal mode, only a general discussion of the wait mode is provided below.

The wait mode power consumption depends on how many systems are active. The power consumption will be highest when all the systems (timer, TCMP and SPI) are active. The power consumption will be the least when the SPI system is disabled (timer operation cannot be disabled in the wait mode). If a non-reset exit from the wait mode is performed (i.e., timer overflow interrupt exit), the state of the remaining systems will be unchanged. If a reset exit from the wait mode is performed all the systems revert to the disabled reset state.

Instruction Set

The MCU has a set of 62 basic instructions. They can be divided into five different types: register/memory, read/modify/write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

All of the instructions used in the CDP6805 CMOS Family are used in the CDP68HC05D2 MCU, plus an additional one; the multiply (MUL) instruction. This instruction allows for unsigned multiplication of the contents of the accumulator (A) and the index register (X). The high order product is then stored in the index register and the low order product is stored in the accumulator. A detailed definition of the MUL instruction is shown below.

Operation:	X A ← X*A
Description:	Multiplies the eight bits in the index register by the eight bits in the accumulator to obtain a 16-bit unsigned number in the concatenated accumulator and index register.
Condition Codes:	H: Cleared I: Not affected N: Not affected Z: Not affected C: Cleared
Source Form(s):	MUL

Addressing Mode	Cycles	Bytes	Opcode
Inherent	11	1	$42

Register/Memory Instructions

Most of these instructions use two operands. The first operand is either the accumulator or the index register. The second operand is obtained from memory using one of the addressing modes. The operand for the jump unconditional (JMP) and jump to subroutine (JSR) instructions is the program counter Refer to Table VI.

Ready-Modify-Write Instructions

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

Table VI — Register/Memory Instructions

		Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
Function	Mnem	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	OP Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	3	C6	3	4	F6	1	3	E6	2	4	D6	3	5
Load X from Memory	LDX	AE	2	2	BE	2	3	CE	3	4	FE	1	3	EE	2	4	DE	3	5
Store A in Memory	STA	—	—	—	B7	2	4	C7	3	5	F7	1	4	E7	2	5	D7	3	6
Store X in Memory	STX	—	—	—	BF	2	4	CF	3	5	FF	1	4	EF	2	5	DF	3	6
Add Memory to A	ADD	AB	2	2	BB	2	3	CB	3	4	FB	1	3	EB	2	4	DB	3	5
Add Memory and Carry to A	ADC	A9	2	2	B9	2	3	C9	3	4	F9	1	3	E9	2	4	D9	3	5
Subtract Memory	SUB	A0	2	2	B0	2	3	C0	3	4	F0	1	3	E0	2	4	D0	3	5
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	3	C2	3	4	F2	1	3	E2	2	4	D2	3	5
AND Memory to A	AND	A4	2	2	B4	2	3	C4	3	4	F4	1	3	E4	2	4	D4	3	5
OR Memory with A	ORA	AA	2	2	BA	2	3	CA	3	4	FA	1	3	EA	2	4	DA	3	5
Exclusive OR Memory with A	EOR	AB	2	2	B8	2	3	C8	3	4	F8	1	3	E8	2	4	D8	3	5
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	3	C1	3	4	F1	1	3	E1	2	4	D1	3	5
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	3	C3	3	4	F3	1	3	E3	2	4	D3	3	5
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	3	C5	3	4	F5	1	3	E5	2	4	D5	3	5
Jump Unconditional	JMP	—	—	—	BC	2	2	CC	3	3	FC	1	2	EC	2	3	DC	3	4
Jump to Subroutine	JSR	—	—	—	BD	2	5	CD	3	6	FD	1	5	ED	2	6	DD	3	7

Table VII — Read-Modify-Write Instructions

		Addressing Modes														
		Inherent (A)			Inherent (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
Function	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	3	5C	1	3	3C	2	5	7C	1	5	6C	2	6
Decrement	DEC	4A	1	3	5A	1	3	3A	2	5	7A	1	5	6A	2	6
Clear	CLR	4F	1	3	5F	1	3	3F	2	5	7F	1	5	6F	2	6
Complement	COM	43	1	3	53	1	3	33	2	5	73	1	5	63	2	6
Negate (2's Complement)	NEG	40	1	3	50	1	3	30	2	5	70	1	5	60	2	6
Rotate Left Thru Carry	ROL	49	1	3	59	1	3	39	2	5	79	1	5	69	2	6
Rotate Right Thru Carry	ROR	46	1	3	56	1	3	36	2	5	76	1	5	66	2	6
Logical Shift Left	LSL	48	1	3	58	1	3	38	2	5	78		5	68	2	6
Logical Shift Right	LSR	44	1	3	54	1	3	34	2	5	74	1	5	64	2	6
Arithmetic Shift Right	ASR	47	1	3	57	1	3	37	2	5	77	1	5	67	2	6
Test for Negative or Zero	TST	4D	1	3	5D	1	3	3D	2	4	7D	1	4	6D	2	5
Multiply	MUL	42	1	11	—	—	—	—	—	—	—	—	—	—	—	—

Branch Instructions

Most branch instructions test the state of the condition code register and, if certain criteria are met, a branch is executed. This adds an offset between -127 and +128 to the current program counter. Refer to Table VIII.

Table VIII — Branch Instructions

		Relative Addressing Mode		
Function	Mnemonic	Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	3
Branch Never	BRN	21	2	3
Branch IFF Higher	BHI	22	2	3
Branch IFF Lower or Same	BLS	23	2	3
Branch IFF Carry Clear	BCC	24	2	3
(Branch IFF Higher or Same)	(BHS)	24	2	3
Branch IFF Carry Set	BCS	25	2	3
(Branch IFF Lower)	(BLO)	25	2	3
Branch IFF Not Equal	BNE	26	2	3
Branch IFF Equal	BEQ	27	2	3
Branch IFF Half Carry Clear	BHCC	28	2	3
Branch IFF Half Carry Set	BHCS	29	2	3
Branch IFF Plus	BPL	2A	2	3
Branch IFF Minus	BMI	2B	2	3
Branch IFF Interrupt Mask Bit is Clear	BMC	2C	2	3
Branch IFF Interrupt Mask Bit is Set	BMS	2D	2	3
Branch IFF Interrupt Line is Low	BIL	2E	2	3
Branch IFF Interrupt Line is High	BIH	2F	2	3
Branch to Subroutine	BSR	AD	2	6

Bit Manipulation Instructions

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space except for ROM, port D data location ($03) bits 0, 1, 6, 7, serial peripheral status register ($0B), timer status register ($13), and timer input capture register ($14, $15). All port registers, DDRs, timer, serial system, on-chip RAM, and 128 bytes of ROM reside in the first 256 bytes (pages zero). An additional feature allows the software to test and branch on the state of any bit within the first 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is automatically placed in the carry bit of the condition code register. Refer to Table IX.

Table XI — Bit Manipulation Instructions

		Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
Function	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IFF Bit n is Set	BRSET n (n=0...7)	—	—	—	2•n	3	5
Branch IFF Bit n is Clear	BRCLR n (n=0...7)	—	—	—	01 + 2•n	3	5
Set Bit n	BSET n (n=0...7)	10 + 2•n	2	5	—	—	—
Clear Bit n	BCLR n (n=0...7)	11 + 2•n	2	5	—	—	—

Control Instructions

These instructions are register reference instructions and are used to control processor operation during a program execution. Refer to Table X.

Table X — Control Instructions

Function	Mnemonic	Inherent Op Code	Inherent # Bytes	Inherent # Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	10
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2
Stop	STOP	8E	1	2
Wait	WAIT	8F	1	2

Alphabetical Listing

The complete instruction set is given in alphabetical order in Table XI.

Opcode Map

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

Addressing Modes

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code to all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short absolute (direct) and long absolute (extended) addressing are also included. One and two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory. Table XII shows the addressing modes for each instruction, with the effects each instruction has on the condition code register.

The term "effective address" (EA) is used in describing the various addressing modes, and is defined as the byte address to or from which the argument for an instruction is fetched or stored. The ten addressing modes of the processor are described below. Parentheses are used to indicate "contents of" the location or register referred to; e.g., (PC) indicates the contents of the location pointed to by the PC. An arrow indicates "is replaced by", and a colon indicates concatenation of two bytes.

Table XI — Instruction Set

	Addressing Modes										Condition Codes				
Mnemonic	Inherent	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		X	X	X		X	X	X			A	●	A	A	A
ADD		X	X	X		X	X	X			A	●	A	A	A
AND		X	X	X		X	X	X			●	●	A	A	●
ASL	X		X			X	X				●	●	A	A	A
ASR	X		X			X	X				●	●	A	A	A
BCC					X						●	●	●	●	●
BCLR									X		●	●	●	●	●
BCS					X						●	●	●	●	●
BEQ					X						●	●	●	●	●
BHCC					X						●	●	●	●	●
BHCS					X						●	●	●	●	●
BHI					X						●	●	●	●	●
BHS					X						●	●	●	●	●
BIH					X						●	●	●	●	●
BIL					X						●	●	●	●	●
BIT		X	X	X		X	X	X			●	●	A	A	●
BLO					X						●	●	●	●	●
BLS					X						●	●	●	●	●
BMC					X						●	●	●	●	●
BMI					X						●	●	●	●	●
BMS					X						●	●	●	●	●
BNE					X						●	●	●	●	●
BPL					X						●	●	●	●	●
BRA					X						●	●	●	●	●
BRN					X						●	●	●	●	●
BRCLR										X	●	●	●	●	A
BRSET										X	●	●	●	●	A
BSET									X		●	●	●	●	●
BSR					X						●	●	●	●	●
CLC	X										●	●	●	●	0
CLI	X										●	0	●	●	●
CLR	X		X			X	X				●	●	0	1	●
CMP		X	X	X		X	X	X			●	●	A	A	A
COM	X		X			X	X				●	●	A	A	1
CPX		X	X	X		X	X	X			●	●	A	A	A
DEC	X		X			X	X				●	●	A	A	●
EOR		X	X	X		X	X	X			●	●	A	A	●
INC	X		X			X	X				●	●	A	A	●
JMP			X	X		X	X	X			●	●	●	●	●
JSR			X	X		X	X	X			●	●	●	●	●
LDA		X	X	X		X	X	X			●	●	A	A	●
LDX		X	X	X		X	X	X			●	●	A	A	●
LSL	X		X			X	X				●	●	A	A	A
LSR	X		X			X	X				●	●	0	A	A
MUL	X										0	●	●	●	0
NEG	X		X			X	X				●	●	A	A	A
NOP	X										●	●	●	●	●
ORA		X	X	X		X	X	X			●	●	A	A	●
ROL	X		X			X	X				●	●	A	A	A
ROR	X		X			X	X				●	●	A	A	A
RSP	X										●	●	●	●	●
RTI	X										?	?	?	?	?
RTS	X										●	●	●	●	●
SBC		X	X	X		X	X	X			●	●	A	A	A
SEC	X										●	●	●	●	1
SEI	X										●	1	●	●	●
STA			X	X		X	X	X			●	●	A	A	●
STOP	X										●	0	●	●	●
STX			X	X		X	X	X			●	●	A	A	●
SUB		X	X	X		X	X	X			●	●	A	A	A
SWI	X										●	1	●	●	●
TAX	X										●	●	●	●	●
TST	X		X			X	X				●	●	A	A	●
TXA	X										●	●	●	●	●
WAIT	X										●	0	●	●	●

Condition Code Symbols:

H	Half Carry (From Bit 3)	Z	Zero	●	Not Affected	0	Cleared
I	Interrupt Mask	C	Carry/Borrow	?	Load CC Register From Stack	1	Set
N	Negate (Sign Bit)	A	Test and Set if True Cleared Otherwise				

3

CDP68HC05D2

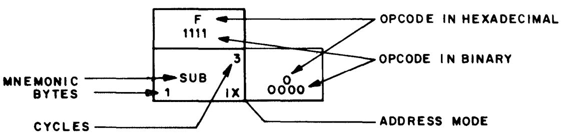
Table XII — CDP68HC05D2 HCMOS Instruction Set Opcode Map

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

Abbreviations for Address Modes

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

LEGEND



Inherent
In inherent instructions, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator, and no other arguments, are included in this mode.

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

EA = PC +1; PC ← PC + 2

Direct
In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two byte instruction. This includes all on-chip RAM and I/O registers, and 128 bytes of on-chip ROM. Direct addressing is efficient in both memory and time

EA = (PC +1); PC ← PC + 2
Address Bus High ← 0; Address Bus Low ← (PC+1)

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

EA = (PC +1):(PC + 2); PC ← PC + 3
Address Bus High ← (PC + 1); Address Bus Low ← (PC+2)

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

EA = X; PC ← PC + 1
Address Bus High ← 0; Address Bus Low ← X

Indexed, 8-Bit Offset

Here the EA is obtained by adding the contents of the byte following the opcode to that of the index register; therefore, the operand is located anywhere within the lowest 511 memory locations. For example, this mode of addressing is useful for selecting the mth element in a n element table. All instructions are two bytes. The content of the index register (X) is not changed. The content of (PC+1) is an unsigned 8-bit integer. One byte offset indexing permits look-up tables to be easily accessed in either RAM or ROM.

$$EA = X + (PC + 1);\ PC \leftarrow PC + 2$$
$$\text{Address Bus High} \leftarrow K;\ \text{Address Bus Low} \leftarrow X + (PC + 1)$$

where;

K = The carry from the addition of X + (PC +1)

Indexed, 16-Bit Offset

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed 8-bit offset, except that this three byte instruction allows tables to be anywhere in memory (e.g., jump tables in ROM). The content of the index register is not changed

$$EA = X + [(PC + 1):(PC + 2))];\ PC \leftarrow PC + 3$$
$$\text{Address Bus High} \leftarrow (PC + 1) + K;$$
$$\text{Address Bus Low} \leftarrow X + (PC + 2)$$

where:

K = The carry from the addition of X + (PC + 2)

Relative

Relative addressing is used only in branch instructions. In relative addressing, the content of the 8-bit signed byte following the opcode (the offset) is added to the PC if and only if the branch condition is true. Otherwise, control proceeds to the next instruction. The span of relative addressing is limited to the range of -126 to +129 bytes from the branch instruction opcode location.

$$EA = PC + 2 + (PC + 1);\ PC \leftarrow EA \text{ if branch taken;}$$
$$\text{otherwise, } EA = PC \leftarrow PC + 2$$

Bit Set/Clear

Direct addressing and bit addressing are combined in instructions which set and clear individual memory and I/O bits. In the bit set and clear instructions, the byte is specified as a direct address in the location following the opcode. The first 256 addressable locations are thus accessed The bit to be modified within that byte is specified in the first three bits of the opcode. The bit set and clear instructions occupy two bytes, one for the opcode (including the bit number) and the other to address the byte which contains the bit of interest.

$$EA = (PC + 1);\ PC \leftarrow PC + 2$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow (PC + 1)$$

Bit Test and Branch

Bit test and branch is a combination of direct addressing, bit set/clear addressing, and relative addressing. The actual bit to be tested, within the byte, is specified within the low order nibble of the opcode. The address of the data byte to be tested is located via a direct address in the location following the opcode byte (EA1). The signed relative 8-bit offset is in the third byte (EA2) and is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any bit in the first 256 locations of memory.

$$EA1 = (PC + 1)$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow (PC + 1)$$
$$EA2 = PC + 3 + (PC + 2),\ PC \leftarrow EA2 \text{ if branch taken;}$$
$$\text{otherwise, } PC \leftarrow PC + 3$$

CDP68HC05D2

Device Characteristics

MAXIMUM RATINGS *(Voltages Referenced to V_{SS})*

Ratings	Symbol	Value	Unit
Supply Voltage	V_{DD}	-0.5 to +7.0	V
Input Voltage	V_{in}	V_{SS} -0.5 to V_{DD} +0.5	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range	T_A	-40 to +125	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C

THERMAL CHARACTERISTICS

Characteristics	Symbol	Value	Unit
Thermal Resistance			
Ceramic	θJA	50	°C/W
Plastic		100	
Plastic Chip Carrier		70	

This device contains circuitry to protect the inputs against damage due to high static voltages of electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in} \text{ or } V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs except OSC2 are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

V_{DD} = 4.5 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD6	3.26 kΩ	2.38 kΩ	50 pF
PD1-PD4	1.9 kΩ	2.26 kΩ	200 pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD6	10.91 kΩ	6.32 kΩ	50 pF
PD1-PD4	6 kΩ	6 kΩ	200 pF

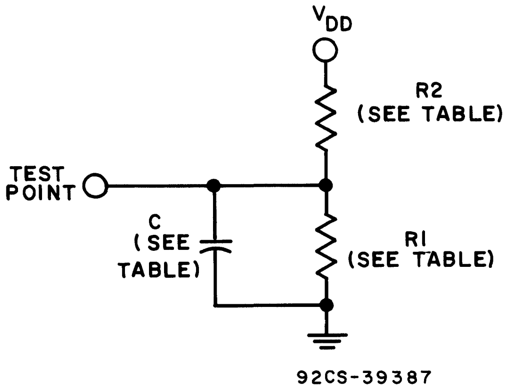


Fig. 24 - Equivalent Test Load

Power Considerations

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

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

Where·

T_A = Ambient Temperature, °C

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

$P_D = P_{INT} + P_{I/O}$

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

$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

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

An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected is:

$$P_D = K + (T_J + 273°C) \quad (2)$$

Solving equations 1 and 2 for K gives:

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

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

DC ELECTRICAL CHARACTERISTICS *(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = -40°C to +125°C unless otherwise noted)*

Characteristic	Symbol	Limits Min	Limits Typ	Limits Max	Unit
Output Voltage, $I_{LOAD} \leq$ 10 0 μA	V_{OL} V_{OH}	— V_{DD}-0.1	— —	0 1 —	V V
Output High Voltage (I_{Load} = 0.8 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (I_{Load} = 1.6 mA) PD1-PD4	 V_{OH} V_{OH}	 V_{DD}-0 8 V_{DD}-0 8	 — —	 — —	 V V
Output Low Voltage (I_{Load} = 1 6 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD2-PD5, TCMP	 V_{OL}	 —	 —	 0 4	 V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	 V_{IH}	 0.7 x V_{DD}	 —	 V_{DD}	 V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	 V_{IL}	 V_{SS}	 —	 0.2 x V_{DD}	 V
Total Supply Current (C_L = 50 pF on Ports, no dc Loads, t_{cyc} = 500 ns, (V_{IL} = 0.2 V, V_{IH} = V_{DD} - 0.2V) No external timer oscillator RUN WAIT (See Note) STOP (See Note)	 I_{DD} I_{DD} I_{DD}	 — — —	 4 1 5 1 0	 TBD TBD TBD	 mA mA μA
Total Supply Current (C_L = 50 pF on Ports, no dc Loads, t_{cyc} = 500 ns, (V_{IL} = 0.2 V, V_{IN} = V_{DD} - 0 2V) 32 768 KHz external timer crystal oscillator for circuit as shown in Fig. 13(c). RUN WAIT (See Note) STOP (See Note)	 I_{DD} I_{DD} I_{DD}	 — — —	 4.5 2.0 500	 TBD TBD TBD	 mA mA μA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD5	 I_{IL}	 —	 —	 ±10	 μA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0, PD7	 I_{in}	 —	 —	 ±1	 μA
Capacitance Ports (as input or output) $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	 pF pF

NOTE: Measured under the following conditions:
1. All ports are configured as input, V_{IL} = 0.2 V, V_{IH} = V_{DD} - 0 2 V.
2. No load on TCMP, C_L = 20 pF on OSC2
3. OSC1 is a square wave with V_{IL} = 0 2 V, V_{IH} = V_{DD} - 0.2 V
4. SPE = 0

CDP68HC05D2

DC ELECTRICAL CHARACTERISTICS *(V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = -40°C to +125°C unless otherwise noted)*

Characteristic	Symbol	Limits Min	Limits Typ	Limits Max	Unit
Output Voltage, $I_{LOAD} \le 10.0\ \mu A$	V_{OL} V_{OH}	— V_{DD}-0 1	— —	0 1 —	V V
Output High Voltage (I_{Load} = 0 2 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP, PD5 (I_{Load} = 0 4 mA) PD1-PD4	 V_{OH} V_{OH}	 V_{DD}-0.3 V_{DD}-0.3	 — —	 — —	 V V
Output Low Voltage (I_{Load} = 0.4 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD2-PD5, TCMP	V_{OL}	—	—	0 3	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	0 7 x V_{DD}	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	0.2 x V_{DD}	V
Total Supply Current (C_L = 50 pF on Ports, no dc Loads, t_{cyc} = 1000 ns, (V_{IL} = 0.2 V, V_{IH} = V_{DD} - 0 2V) No external timer oscillator RUN WAIT (See Note) STOP (See Note)	 I_{DD} I_{DD} I_{DD}	 — — —	 1 4 500 1	 TBD TBD TBD	 mA µA µA
Total Supply Current (C_L = 50 pF on Ports, no dc Loads, t_{cyc} = 1000 ns, (V_{IL} = 0.2 V, V_{IH} = V_{DD} - 0 2V) 32 768 KHz external timer crystal oscillator circuit as shown in **Fig. 13(c)**. RUN WAIT (See Note) STOP (See Note)	 I_{DD} I_{DD} I_{DD}	 — — —	 1.5 600 100	 TBD TBD TBD	 mA µA µA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD5	I_{IL}	—	—	±10	µA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0, PD7	I_{in}	—	—	±1	µA
Capacitance Ports (as input or output) $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	 pF pF

NOTE· Measured under the following conditions:

1. All ports are configured as input, V_{IL} = 0.2 V, V_{IH} = V_{DD} - 0 2 V.
2. No load on TCMP, C_L = 20 pF on OSC2.
3. OSC1 is a square wave with V_{IL} = 0 2 V, V_{IH} = V_{DD} - 0.2 V
4. SPE = 0

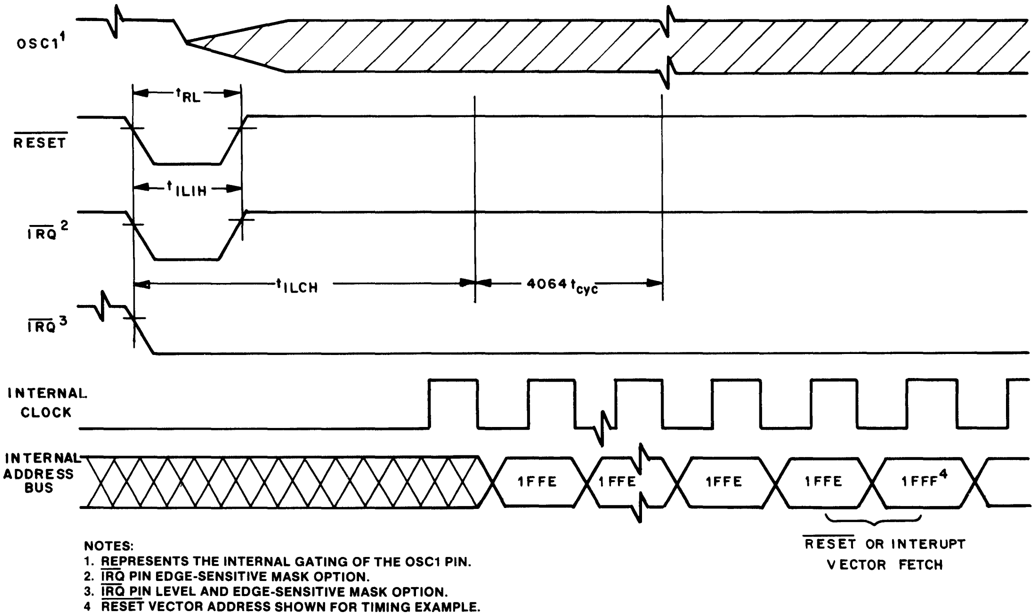
CONTROL TIMING *(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = -40°C to +125°C)*

Characteristic	Symbol	Limits		Unit
		Min	Max	
Frequency of Operation				
Crystal Option	f_{osc}	—	4.2	MHz
External Clock Option	f_{osc}	dc	4 2	MHz
Internal Operating Frequency				
Crystal (f_{osc} ÷ 2)	f_{op}	—	2.1	MHz
External Clock (f_{osc} ÷ 2)	f_{op}	dc	2.1	MHz
Cycle Time **(See Figu[illegible])**	t_{cyc}	480	—	ns
Crystal Oscillator Startup Time **(See Figure 8)**	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) **(See Figure 25)**	t_{ILCH}	—	100	ms
RESET Pulse Width **(See Figure 9)**	t_{RL}	1.5	—	t_{cyc}
Timer				
Resolution**	t_{RESL}	4.0	—	t_{cyc}
Input Capture Pulse Width **(See Figure 26)**	t_{TH}, t_{TL}	125	—	ns
Input Capture Pulse Period **(See Figure 26)**	t_{TLTL}	***	—	t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) **(See Figure 11)**	t_{ILIH}	125	—	ns
Interrupt Pulse Period **(See Figure 11)**	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns
External Timer Oscillator frequency of operation	f_{tosc}	—	f_{osc} ÷ 4	f_{osc}

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}



NOTES:
1. REPRESENTS THE INTERNAL GATING OF THE OSC1 PIN.
2. IRQ PIN EDGE-SENSITIVE MASK OPTION.
3. IRQ PIN LEVEL AND EDGE-SENSITIVE MASK OPTION.
4 RESET VECTOR ADDRESS SHOWN FOR TIMING EXAMPLE.

92CM-39375

Fig. 25 - Stop Recovery Timing Diagram

CDP68HC05D2

CONTROL TIMING (V_{DD} *= 3.0 Vdc ± 10%,* V_{SS} *= 0 Vdc,* T_A *= -40°C to +125°C)*

Characteristic	Symbol	Limits Min	Limits Max	Unit
Frequency of Operation Crystal Option External Clock Option	 f_{osc} f_{osc}	 — dc	 2.0 2.0	 MHz MHz
Internal Operating Frequency Crystal (f_{osc} ÷ 2) External Clock (f_{osc} ÷ 2)	 f_{op} f_{op}	 — dc	 1.0 1.0	 MHz MHz
Cycle Time **(See Figure 8)**	t_{cyc}	1000	—	ns
Crystal Oscillator Startup Time **(See Figure 8)**	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) **(See Figure 25)**	t_{ILCH}	—	100	ms
RESET Pulse Width - Excluding Power-Up **(See Figure 8)**	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width **(See Figure 26)** Input Capture Pulse Period **(See Figure 26)**	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 250 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) **(See Figure 11)**	t_{ILIH}	250	—	ns
Interrupt Pulse Period **(See Figure 11)**	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns
External timer oscillator frequency of operation	f_{tosc}	—	f_{osc} ÷ 4	f_{osc}

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

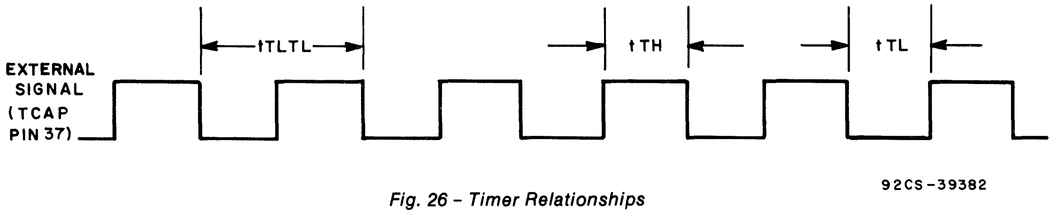


Fig. 26 - Timer Relationships

SERIAL PERIPHERAL INTERFACE (SPI) TIMING (Figure 29)
(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = -40°C to +125°C)

Num.	Characteristic	Symbol	Limits Min	Limits Max	Unit
	Operating Frequency				
	Master	$f_{op(m)}$	dc	0.5	f_{op}***
	Slave	$f_{op(s)}$	dc	2.1	MHz
1	Cycle Time				
	Master	$t_{cyc(m)}$	2.0	—	t_{cyc}
	Slave	$t_{cyc(s)}$	480	—	ns
2	Enable Lead Time				
	Master	$t_{lead(m)}$	*	—	
	Slave	$t_{lead(S)}$	240	—	ns
3	Enable Lag Time				
	Master	$t_{lag(m)}$	*	—	
	Slave	$t_{lag(S)}$	240	—	ns
4	Clock (SCK) High Time				
	Master	$t_{w(SCKH)m}$	340	—	ns
	Slave	$t_{w(SCKH)s}$	190	—	ns
5	Clock (SCK) Low Time				
	Master	$t_{w(SCKL)m}$	340	—	ns
	Slave	$t_{w(SCKL)s}$	190	—	ns
6	Data Setup Time (Inputs)				
	Master	$t_{su(m)}$	100	—	ns
	Slave	$t_{su(s)}$	100	—	ns
7	Data Hold Time (Inputs)				
	Master	$t_{h(m)}$	100	—	ns
	Slave	$t_{h(s)}$	100	—	ns
8	Access Time (Time to data active from high impedance state)				
	Slave	t_a	0	120	ns
9	Disable Time (Hold Time to High-Impedance State)				
	Slave	t_{dis}	—	240	ns
10	Data Valid				
	Master (Before Capture Edge)	$t_{v(m)}$	0.25	—	$t_{cyc(m)}$
	Slave (After Enable Edge)**	$t_{v(s)}$	—	240	ns
11	Data Hold Time (Outputs)				
	Master (After Capture Edge)	$t_{ho(m)}$	0.25	—	$t_{cyc(m)}$
	Slave (After Enable Edge)	$t_{ho(s)}$	0	—	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF)				
	SPI Outputs (SCK, MOSI, MISO)	t_{rm}	—	100	ns
	SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	t_{rs}	—	2.0	μs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200 pF)				
	SPI Outputs (SCK, MOSI, MISO)	t_{fm}	—	100	ns
	SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	t_{fs}	—	2.0	μs

*Signal production depends on software
**Assumes 200 pF load on all SPI pins
***Note that the unit this specification uses is f_{op} (internal operating frequency), not MHz! In the master mode the SPI bus is capable of running at one-half of the device's internal operating frequency, therefore 1.05 MHz maximum.

CDP68HC05D2

SERIAL PERIPHERAL INTERFACE (SPI) TIMING (Figure 29)

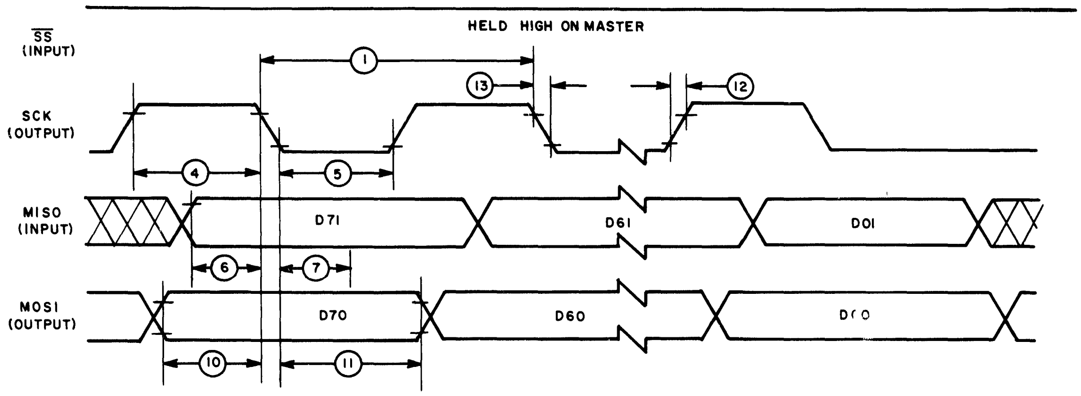
($V_{DD} = 3.3$ Vdc ± 10%, $V_{SS} = 0$ Vdc, $T_A = -40°C$ to $+125°C$)

Num.	Characteristic	Symbol	Limits Min	Limits Max	Unit
	Operating Frequency				
	Master	$f_{op(m)}$	dc	0.5	f_{op}***
	Slave	$f_{op(s)}$	dc	1.0	MHz
1	Cycle Time				
	Master	$t_{cyc(m)}$	2.0	—	t_{cyc}
	Slave	$t_{cyc(s)}$	1.0	—	μs
2	Enable Lead Time				
	Master	$t_{lead(m)}$	*	—	
	Slave	$t_{lead(S)}$	500	—	ns
3	Enable Lag Time				
	Master	$t_{lag(m)}$	*	—	
	Slave	$t_{lag(S)}$	500	—	ns
4	Clock (SCK) High Time				
	Master	$t_{w(SCKH)m}$	720	—	μs
	Slave	$t_{w(SCKH)s}$	400	—	ns
5	Clock (SCK) Low Time				
	Master	$t_{w(SCKL)m}$	720	—	μs
	Slave	$t_{w(SCKL)s}$	400	—	ns
6	Data Setup Time (Inputs)				
	Master	$t_{su(m)}$	200	—	ns
	Slave	$t_{su(s)}$	200	—	ns
7	Data Hold Time (Inputs)				
	Master	$t_{h(m)}$	200	—	ns
	Slave	$t_{h(s)}$	200	—	ns
8	Access Time (Time to data active from high impedance state)				
	Slave	t_a	0	250	ns
9	Disable Time (Hold Time to High-Impedance State)				
	Slave	t_{dis}	—	500	ns
10	Data Valid				
	Master (Before Capture Edge)	$t_{v(m)}$	0.25	—	$t_{cyc(m)}$
	Slave (After Enable Edge)**	$t_{v(s)}$	—	500	ns
11	Data Hold Time (Outputs)				
	Master (After Capture Edge)	$t_{ho(m)}$	0.25	—	$t_{cyc(m)}$
	Slave (After Enable Edge)	$t_{ho(s)}$	0	—	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, $C_L = 200$ pF)				
	SPI Outputs (SCK, MOSI, MISO)	t_{rm}	—	200	ns
	SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	t_{rs}	—	2.0	μs
13	Fall Time (70% V_{DD} to 20% V_{DD}, $C_L = 200$ pF)				
	SPI Outputs (SCK, MOSI, MISO)	t_{fm}	—	200	ns
	SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	t_{fs}	—	2.0	μs

*Signal production depends on software

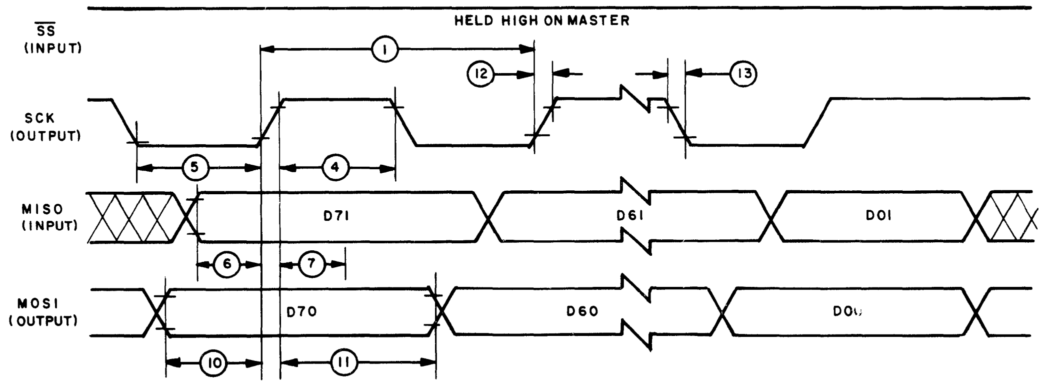
**Assumes 200 pF load on all SPI pins.

***Note that the unit this specification uses is f_{op} (internal operating frequency), not MHz! In the master mode the SPI bus is capable of running at one-half of the device's internal operating frequency, therefore 0.5 MHz maximum.



(a) SPI Master Timing CPOL = 0, CPHA = 1

92CM - 39372

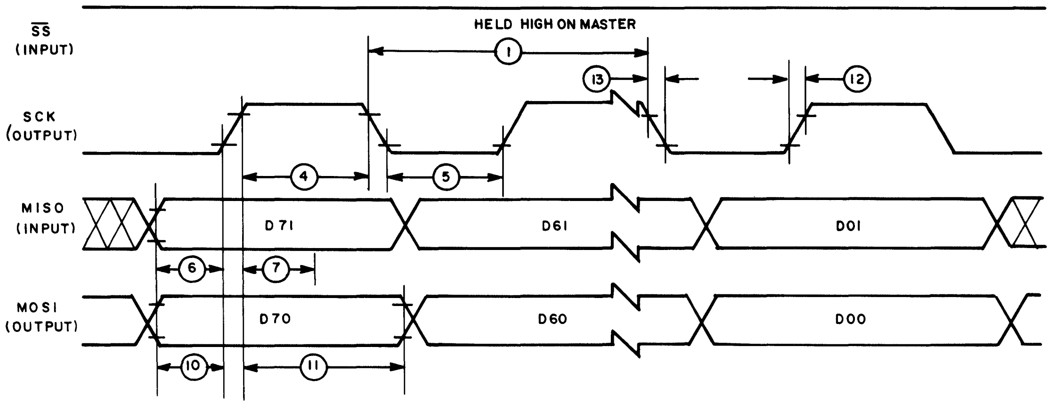


(b) SPI Master Timing CPOL = 1, CPHA = 1

92CM-39372

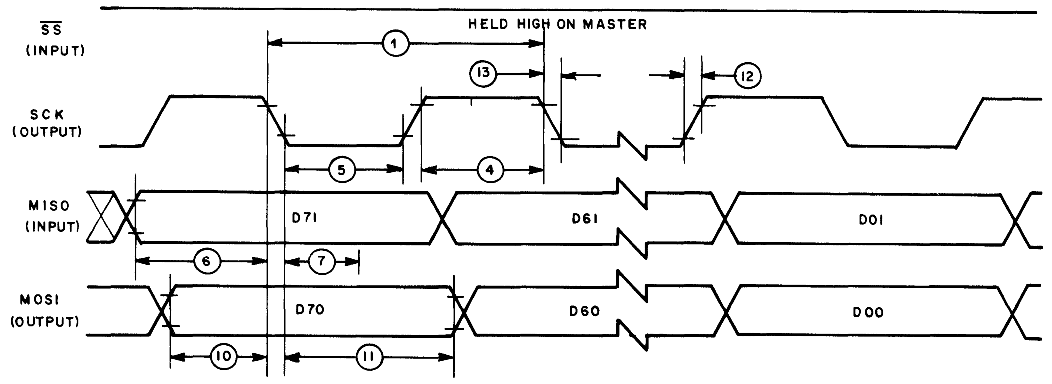
NOTE: MEASUREMENT POINTS ARE V_{OL}, V_{OH}, V_{IL}, V_{IH}

Fig. 27 - Timing Diagrams



(c) SPI Master Timing CPOL = 0, CPHA = 0

92CM-39372

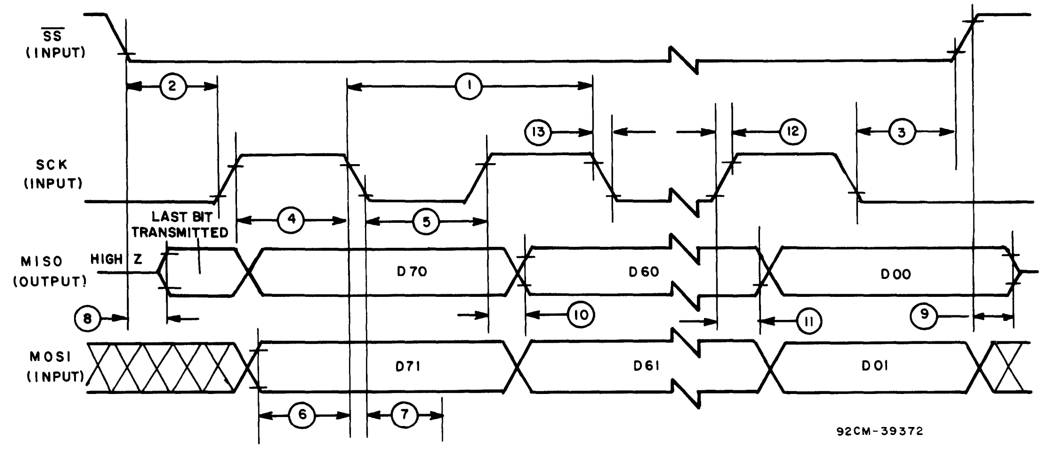


(d) SPI Master Timing CPOL = 1, CPHA = 0

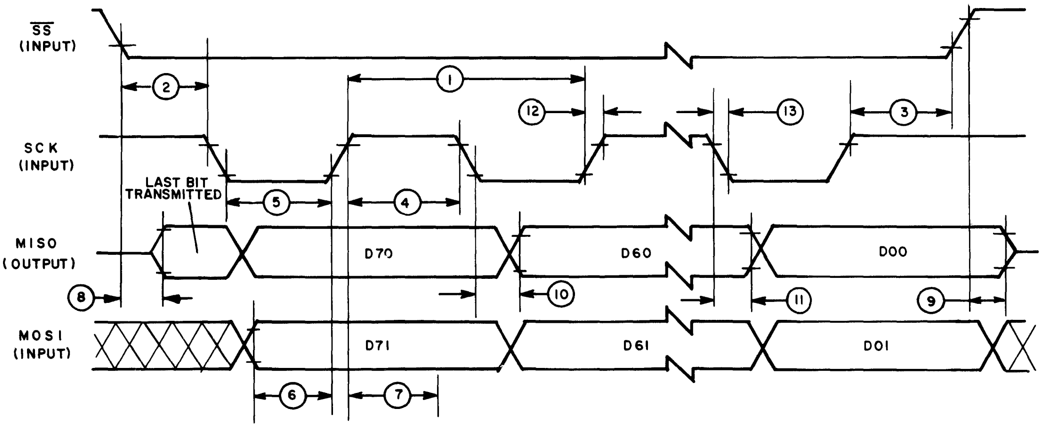
92CM-39372

NOTE: MEASUREMENT POINTS ARE V_{OL}, V_{OH}, V_{IL} AND V_{IH}

Fig. 27 - Timing Diagrams (Continued)



(e) SPI Slave Timing CPOL = 0, CPHA = 1

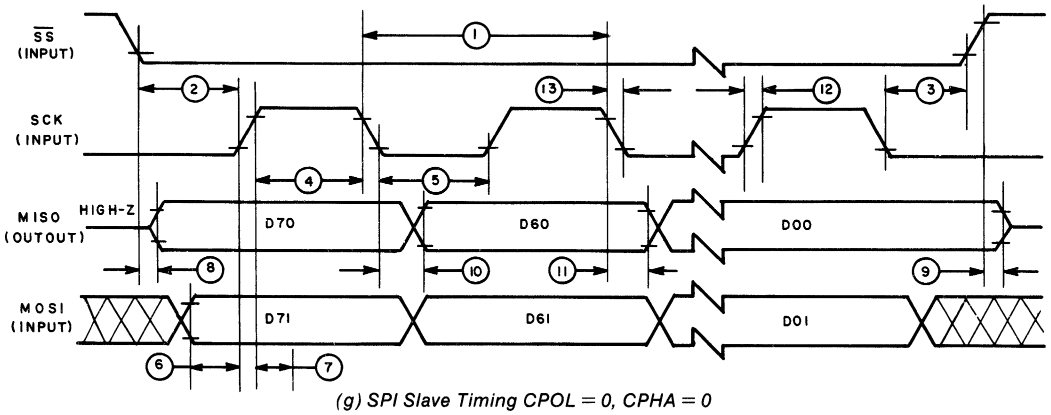


(f) SPI Slave Timing CPOL = 1, CPHA = 1

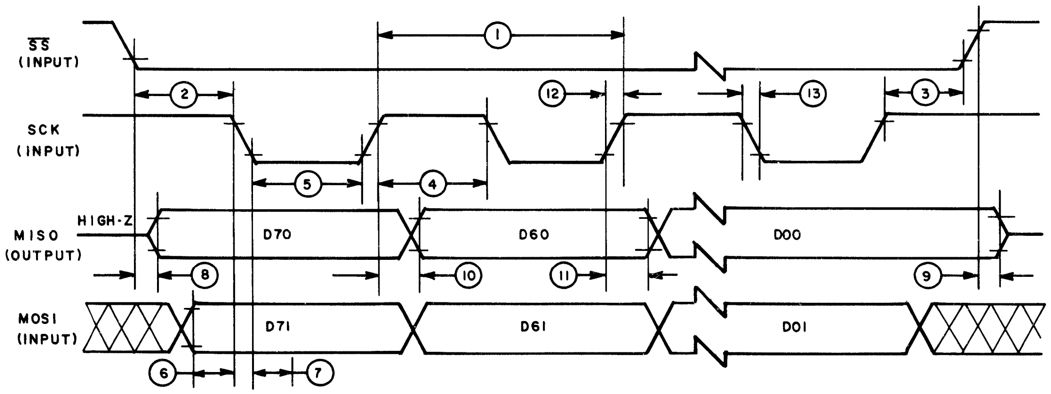
92CM-39372

NOTE: MEASUREMENT POINTS ARE V_{OL}, V_{OH}, V_{IL}, AND V_{IH}.

Fig. 27 - Timing Diagrams (Continued)



(g) SPI Slave Timing CPOL = 0, CPHA = 0



(h) SPI Slave Timing CPOL = 1, CPHA = 0

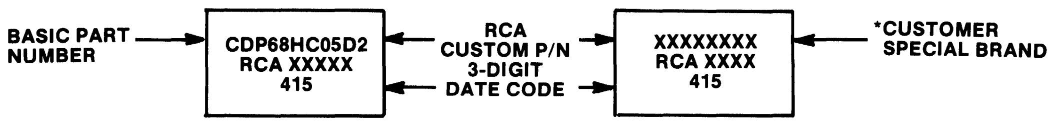
NOTE: MEASUREMENT POINTS ARE V_{OL}, V_{OH}, V_{IL} AND V_{IH}

92CM-39372

Fig. 27 - Timing Diagrams (Concluded)

Branding:

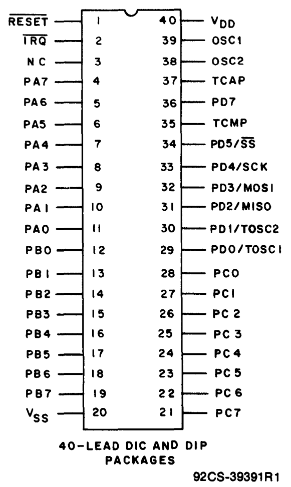
The packages (DIC, DIP, or PCC) in which the RCA custom Microcomputers are supplied are branded with both the basic type number and an RCA custom part number. Please refer to both numbers when discussing a custom Microcomputer order with RCA representatives. RCA can accommodate special requirements of customers. The standard format is as follows:

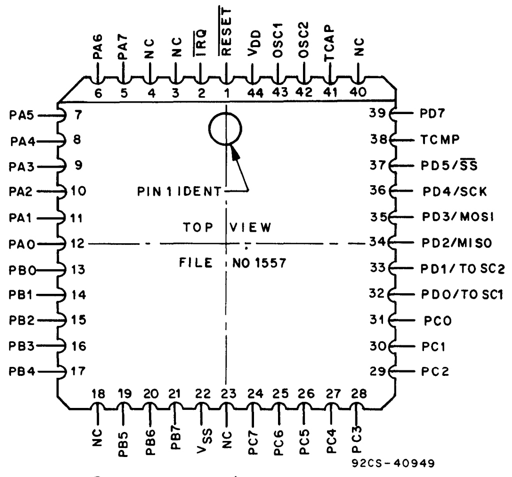


*CUSTOMER SPECIAL BRAND (UP TO 10 CHARACTERS FOR DIC. 13 CHARACTERS FOR DIP).

Mechanical Data

Terminal Assignments





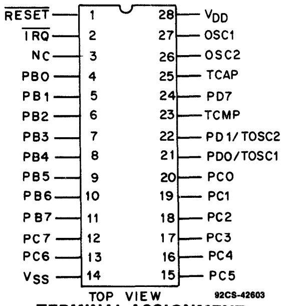
N C = no connection

D Suffix - 40-Lead Dual-In-Line Side-Brazed Ceramic Package
E Suffix - 40-Lead Dual-In-Line Plastic Package

Q Suffix - 44-Lead Plastic Chip-Carrier Package

CDP68HC05D2A

Product Preview



TERMINAL ASSIGNMENT

CMOS High-Performance Silicon-Gate 8-Bit Microcomputer

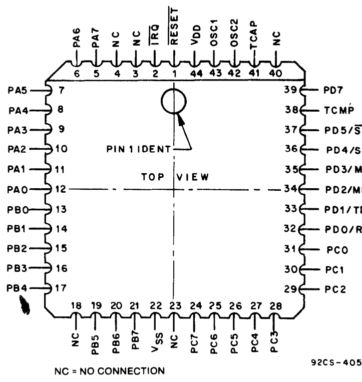
Features:

- *Typical power:*
  Operating, 25 mW
  WAIT, 7.5 mW
  STOP, 5 μW
- *Fully static operation*
- *96 bytes of on-chip RAM*
- *2176 bytes of on-chip ROM*
- *16 I/O and 3-input lines*
- *2.1-MHz internal operating frequency*
- *Internal 16-bit timer*
- *Separate external timer oscillator*
- *External (IRQ), timer, and Port B interrupts*
- *Self-check mode*
- *Single 2.5 to 6-volt supply*
- *RC or crystal on-chip oscillator*
- *8 x 8 multiply instruction*
- *True bit manipulation*
- *Indexed addressing for tables*
- *Memory mapped I/O*

The CDP68HC05D2A Microcomputer Unit (MCU) is a 28-pin version of the 40-pin CDP68HC05D2. In order to accomplish the lower pin count, Port A and lines 2 to 5 of Port D (the SPI bus) are removed in the CDP68HC05D2A, resulting in 12 fewer I/O lines. All other features and functions are identical to those of CDP68HC05D2. Refer to GE publication TSM-204A, "Technical Specifications for the RCA HCMOS Microcomputer CDP68HC05D2." This 8-bit MCU contains on-chip oscillator CPU, RAM, ROM, I/O, and Timer. The fully static design allows operation at frequencies down to DC, further reducing its already low-power consumption. It is a low-power processor designed for low-end to mid-range applications in the telecommunications, consumer, automotive, and industrial markets where low cost and very low power consumption constitute important factors.

The CDP68HC05D2A is supplied in a 28-lead dual-in-line plastic package (E suffix) and a 28-lead plastic chip-carrier package (Q suffix).

TERMINAL ASSIGNMENT



28-Lead Plastic Chip-Carrier Package (Q Suffix)

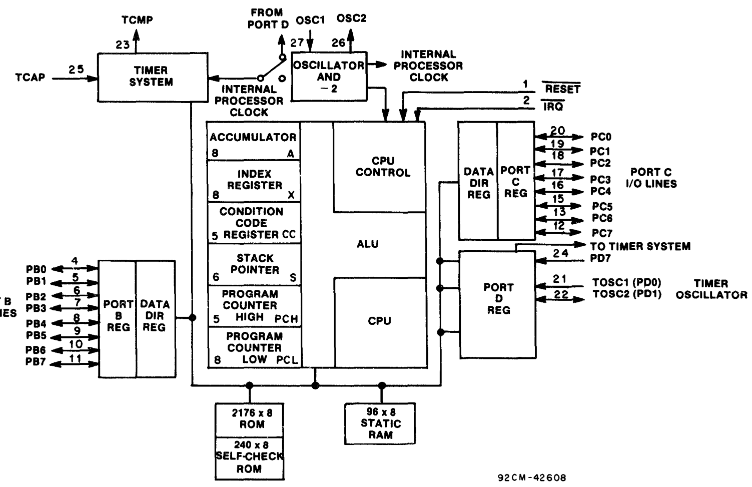


Fig. 1 - CDP68HC05D2A CMOS microcomputer block diagram.

File Number **2100**