HIP7030A2

ADVANCE INFORMATION

August 1996

J1850 8-Bit 68HC05 Microcontroller

Features

- **Fully Supports VPW Specifications of SAE J1850 Standard for Class B Data Communications Network Interface**
- **On-Chip Memory**
- **176 Bytes of RAM**
- **2110 Bytes of User ROM**
- **13 Bidirectional I/O Lines**
- **16-Bit Timer with Capture and Compare Registers**
- **Serial Peripheral Interface (SPI) System**
- **Watchdog Timer and Slow Clock Detect**
- **10MHz Operating Frequency (5.0MHz Internal Bus Frequency) at 5V**
- **Built-In-Test Bootstrap Mode with 242 Bytes of ROM**
- **Two Channel Analog Comparator**
- **On-Chip Oscillator Amplifier**
- **8-Bit CPU Architecture**
- **Power-Saving STOP, WAIT and Data Retention Modes**
- **Full -40^{O}C to 125^{O}C Operating Range**
- **Single 3.0V to 6.0V Supply**
- **28 Lead Dual-In-Line and Small Outline Plastic Packages**

Software Features

- **Standard 68HC05 Instruction Set**
- **True Bit Manipulation**
- **Addressing Modes Include Indexed Addressing**
  - **Memory Mapped I/O**

Ordering Information

PART NUMBER	TEMP. RANGE (^{O}C)	PACKAGE	PKG. NO.
HIP7030A2P	-40 to 125	28 Lead Plastic DIP	M28.3
HIP7030A2M	-40 to 125	28 Lead Plastic SOIC (W)	E28.6

Description

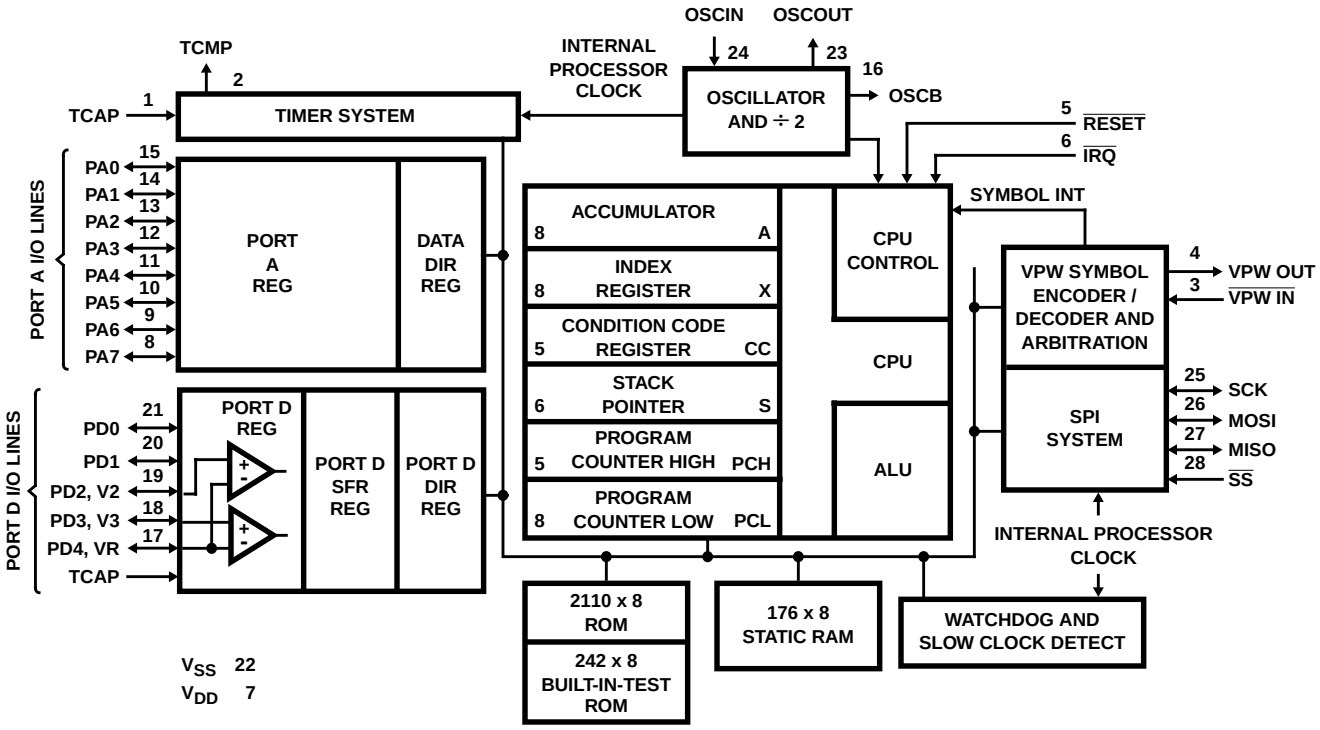
The HIP7030A2 HCMOS Microcomputer is a member of the CDP68HC05 family of low-cost single-chip microcomputers. The integrated hardware functions provide the system designer with a complete set of building blocks for implementing a “Class B” multiplexed communications network interface, which fully conforms to the VPW Multiplexed Wiring protocol specified in SAE Recommended Practice J1850. This 8-bit microcomputer unit (MCU) contains an on-chip oscillator, CPU, 176 bytes of RAM, 2110 bytes of user ROM, 13 I/O lines, a J1850 Variable Pulse Width Symbol Encoder/Decoder (VPW SENDEC) system, a Serial Peripheral Interface (SPI) system, a two channel analog Comparator, a Watchdog Timer, a Slow Clock Detect, and a 16-bit Timer. The static HCMOS design allows operation at input frequencies up to 10MHz (5MHz internal clock).

Table of Contents

http://www.intersil.com or 407-727-9207 | Copyright © Intersil Corporation 1999

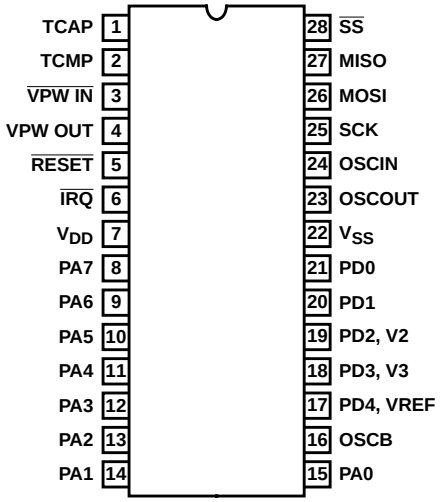
File Number **3646.2**

Block Diagram



Pinout

HIP7030A2
(PDIP, SOIC)
TOP VIEW



Absolute Maximum Ratings

Supply Voltage (V_{DD}) . . . . . -0.3V to +7.0V
Input or Output Voltage
  Pins with V_{DD} Diode . . . . . -0.3V to V_{DD} +0.3V
  Pins without V_{DD} Diode . . . . . -0.3V to +10V
Current Drain Per Pin, I (Excluding V_{DD} and V_{SS}) 25mA
Lead Temperature (Soldering 10s) . . . . . +265^{o}C
ESD Classification . . . . . Class 2
Gate Count . . . . . 9000 Gates

Thermal Information

Thermal Resistance (Typical) θ_{JA}
  Plastic DIP Package . . . . . 60^{o}C/W
  Plastic SOIC Package . . . . . 75^{o}C/W
Maximum Package Power Dissipation at +125^{o}C
  DIP Package 415mW
  SOIC Package 325mW
Operating Temperature Range (T_A) . . . . . -40^{o}C to +125^{o}C
Storage Temperature Range (T_{STG}) . . . . . -65^{o}C to +150^{o}C
Junction Temperature . . . . . +150^{o}C

CAUTION: Stresses above those listed in "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress only rating and operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

Operating Conditions

Operating Voltage Range . . . . . +3.0V to +5.5V
Operating Temperature Range . . . . . -40^{o}C to 125^{o}C
Input Low Voltage . . . . . 0V to +0.8V
Input High Voltage . . . . . (0.8•V_{DD}) to V_{DD}
Input Rise and Fall Time
  CMOS Inputs . . . . . 100ns Max.
  CMOS Schmitt Inputs . . . . . Unlimited

DC Electrical Specifications

V_{DD} = 5V_{DC} ±10%, V_{SS} = 0V_{DC}, T_A = -40^{o}C to +125^{o}C Unless Otherwise Specified

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
No Load Output Voltage	V_{OL}	I_{LOAD} < ±10µA	-	-	0.1	V
	V_{OH}		V_{DD} -0.1	-	-	V
Output High Voltage: PA0-7, PD0-4, VPWOUT, TCMP	V_{OH}	I_{LOAD} = -0.8mA	V_{DD} -0.8	V_{DD} -0.4	-	V
Output High Voltage: OSCOUT	V_{OH}	I_{LOAD} = -0.08mA	V_{DD} -0.8	V_{DD} -0.4	-	V
Output High Voltage: MISO, MOSI, SCK, OSCB	V_{OH}	I_{LOAD} = -1.6mA	V_{DD} -0.8	V_{DD} -0.4	-	V
Output Low Voltage: OSCOUT	V_{OL}	I_{LOAD} = 0.17mA	-	0.2	0.4	V
Output Low Voltage: MISO, MOSI, SCK, OSCB	V_{OL}	I_{LOAD} = 1.6mA	-	0.2	0.4	V
Input High Voltage: PA0-7, PD0-4, MISO, MOSI, $\overline{SS}$, SCK	V_{IH}		0.7•V_{DD}	-	V_{DD}	V
Input High Voltage: $\overline{RESET}$, $\overline{IRQ}$, TCAP, $\overline{VPWIN}$, OSCIN	V_{IH}		0.8•V_{DD}	-	V_{DD}	V
Input Low Voltage: PA0-7, PD0-4, MISO, MOSI, $\overline{SS}$, SCK	V_{IL}		V_{SS}	-	0.3•V_{DD}	V
Input Low Voltage: $\overline{RESET}$, $\overline{IRQ}$, TCAP, $\overline{VPWIN}$, OSCIN	V_{IL}		V_{SS}	-	0.2•V_{DD}	V
Input Hysteresis Voltage: $\overline{RESET}$, $\overline{IRQ}$, TCAP, $\overline{VPWIN}$, OSCIN	V_{HYS}		0.1•V_{DD}	1.0	0.5•V_{DD}	V
Supply Current						
RUN	I_{RUN}	f_{OSC} = 10MHz External Square Wave	-	8	18	mA
WAIT (Note 2)	I_{WAIT}		-	3.2	10	mA
STOP (Notes 2, 3)	I_{STOP}	T_A = 25^{o}C	-	2	50	µA
		T_A = -40^{o}C to 125^{o}C	-	10	250	µA
I/O Ports Hi-Z Leakage Current: PA0-7, PD0-4, MISO, MOSI, SCK	I_{IL}		-10	±0.01	+10	µA
Input Current: $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSCIN, $\overline{VPWIN}$, $\overline{SS}$	I_{IN}		-1	.001	+1	µA

DC Electrical Specifications

V_{DD} = 5V_{DC} ±10%, V_{SS} = 0V_{DC}, T_A = -40°C to +125°C Unless Otherwise Specified **(Continued)**

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Capacitance: (Note 4)	C_{OUT}		-	-	12	pF
	C_{IN}		-	-	8	pF
Powerdown Input Voltage: $\overline{RESET}$, $\overline{IRQ}$, $\overline{VPWIN}$, OSCIN	V_{INPD}	V_{DD} = 0	-0.3	-	7	V
Comparator: Input Voltage: V_2, V_3, V_{REF}	V_{IN}		V_{SS}-0.2	-	V_{DD} +0.02	V
Input Current: V_2, V_3, V_{REF}	I_{IN}		-1	-	+1	µA
Offset Voltage	V_{OFF}		-	20	-	mV
Response	t_R		-	2	-	µs

NOTES:

1. 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}<(V_{IN} or V_{OUT})<V_{DD}. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).
2. WAIT, STOP I_{DD}: All ports configured as inputs, V_{IL} = 0.2V and V_{IH} = V_{DD} - 0.2V.
3. STOP I_{DD} measured with OSCIN = V_{SS}, no feedback resistor connected.
4. Includes Ports used as Input/Output Pins, Ports used as Input only Pins; Ports used as Output only Pins.

Control Timing

V_{DD} = 5V_{DC} ±10%, V_{SS} = 0V_{DC}, T_A = -40°C to 125°C Unless Otherwise Specified

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Frequency Of Operation Crystal Option	f_{OSC}		1	-	10	MHz
External Clock Option	f_{OSC}		1	-	10	MHz
Internal Operating Frequency Crystal (f_{OSC} +2)	f_{OP}		0.5	-	5	MHz
External Clock (f_{OSC} +2)	f_{OP}		0.5	-	5	MHz
Cycle Time	t_{CYC}		200	-	-	ns
Crystal Oscillator Start-up Time for AT-cut Crystal	t_{OXOV}		-	-	100	ms
Stop Recovery Start-up Time (AT-cut Crystal Oscillator)	t_{ILCH}		-	-	100	ms
$\overline{RESET}$ Pulse Width	t_{RL}		1.5	-	-	t_{CYC}
Timer Resolution (Note 1)	t_{RES}		4	-	-	t_{CYC}
Input Capture Pulse Width	t_{TH}, t_{TL}		50	-	-	ns
Input Capture Pulse Period	t_{TL}, t_{TL}		(Note 2)	-	-	t_{CYC}
Interrupt Pulse Width Low (Edge-Triggered)	t_{ILIH}		50	-	-	ns
OSC1 Pulse Width	t_{OH}, t_{OL}		35	-	-	ns
Slow Clock Detect Frequency Range	f_{SLOW}		20	50	200	KHz

NOTES:

1. 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.
2. 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}.

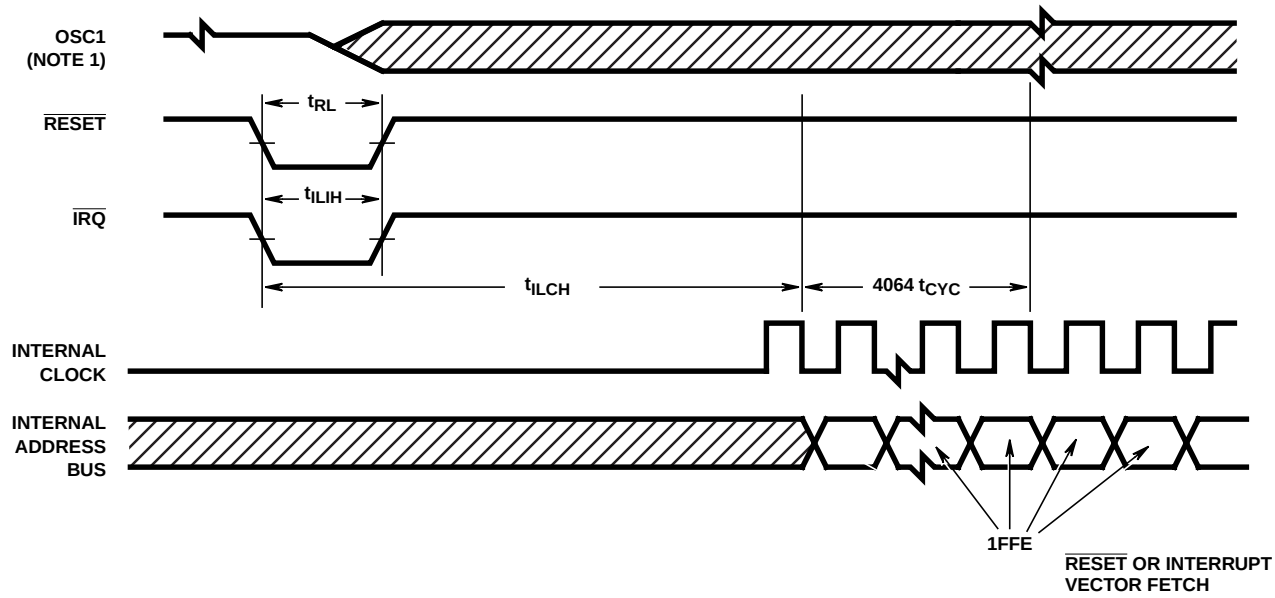
Serial Peripheral Interface (SPI) Timing (See Figure 3) $V_{DD} = 5V_{DC} \pm 10\%$, $V_{SS} = 0V_{DC}$, $T_A = -40^oC$ to 125^oC Unless Otherwise Specified

NUMBER	PARAMETER	SYMBOL	MIN	MAX	UNITS
	Operating Frequency Master	$f_{OP(M)}$	0.03	0.5	f_{OP} (Note 3)
	Slave	$f_{OP(S)}$	DC	5	MHz
1	Cycle Time Master	$t_{CYC(M)}$	2	-	t_{CYC}
	Slave	$t_{CYC(S)}$	200	-	ns
2	Enable Lead Time Master	$t_{LEAD(M)}$	(Note 1)	-	-
	Slave	$t_{LEAD(S)}$	50	-	ns
3	Enable Lag Time Master	$t_{LAG(M)}$	(Note 1)	-	-
	Slave	$t_{LAG(S)}$	50	-	ns
4	Clock (SCK) High Time Master	$t_{W(SCKH)M}$	200	-	ns
	Slave	$t_{W(SCKH)S}$	50	-	ns
5	Clock (SCK) Low Time Master	$t_{W(SCKL)M}$	200	-	ns
	Slave	$t_{W(SCKL)S}$	50	-	ns
6	Data Setup Time (Inputs) Master	$t_{SU(M)}$	50	-	ns
	Slave	$t_{SU(S)}$	50	-	ns
7	Data Hold Time (Inputs) Master	$t_{H(M)}$	50	-	ns
	Slave	$t_{H(S)}$	50	-	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}	-	200	ns
10	Data Valid Time Master (Before Capture Edge)	$t_{V(M)}$	0.25	-	$t_{CYC(M)}$
	Slave (After Enable Edge) (Note 2)	$t_{V(S)}$	-	150	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 (V_{DD} = 20% to 70%, C_L = 100pF) SPI Outputs (SCK, MOSI, MISO)	$t_{R(M)}$	-	50	ns
	SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	$t_{R(S)}$	-	2	µs
13	Fall Time (V_{DD} = 20% to 70%, C_L = 100pF) SPI Outputs (SCK, MOSI, MISO)	$t_{F(M)}$	-	50	ns
	SPI Inputs (SCK, MOSI,MISO, $\overline{SS}$)	$t_{F(S)}$	-	2	µs

NOTES:

1. Signal Production depends on software.
2. Assumes 200pF load on all SPI pins.
3. Note that the units 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 devices's internal operating frequency, therefore, 2.5MHz maximum.

Control Timing Diagrams



NOTE:

1. Represents the internal gating of the OSC1 pin.

FIGURE 1. STOP RECOVERY TIMING DIAGRAM

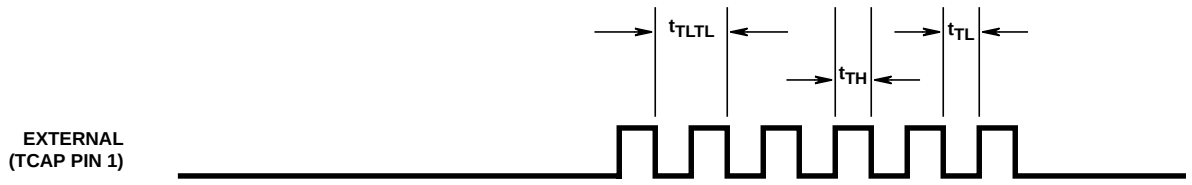


FIGURE 2.

Serial Peripheral Interface (SPI) Timing Diagrams

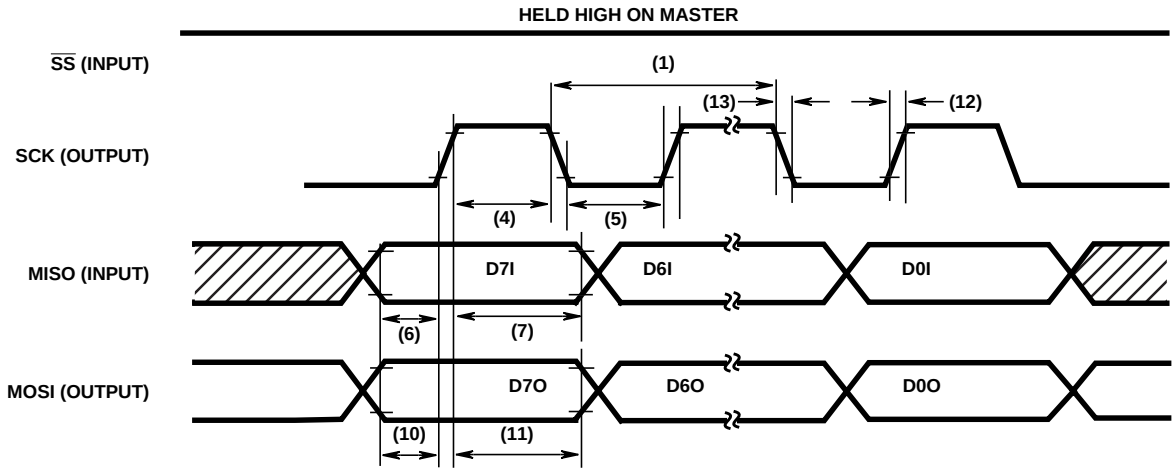


FIGURE 3A. SPI MASTER TIMING CPOL = 0, CPHA = 1

Serial Peripheral Interface (SPI) Timing Diagrams (Continued)

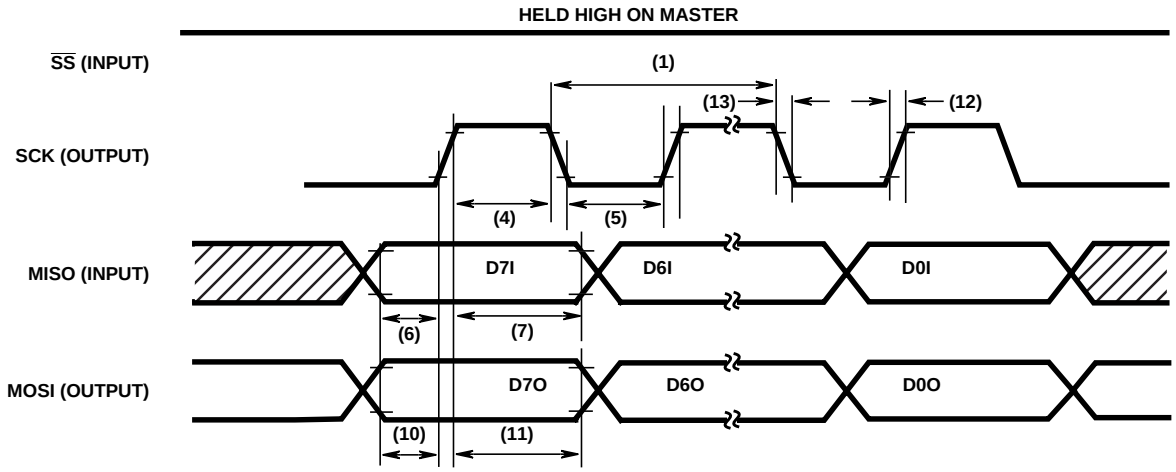


FIGURE 3B. SPI MASTER TIMING CPOL = 1, CPHA = 1

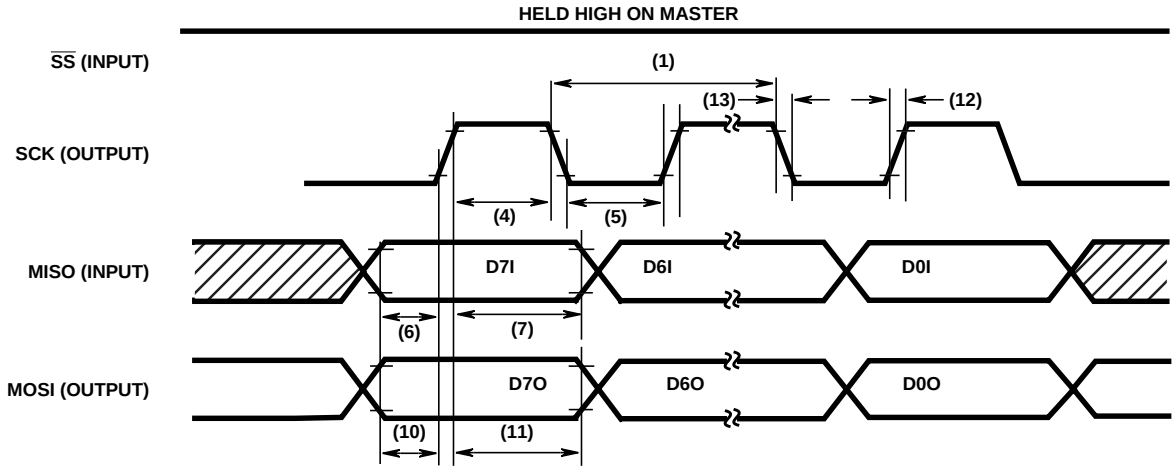
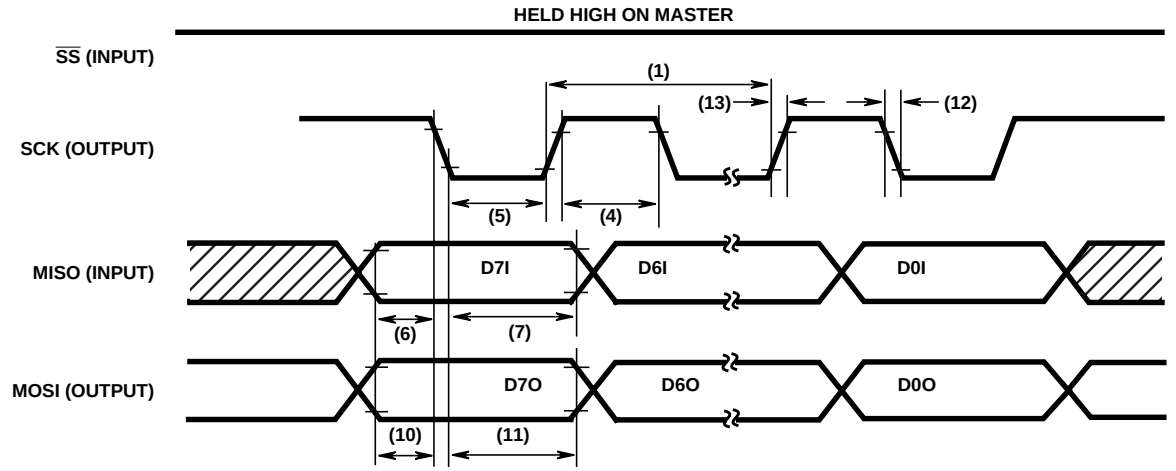


FIGURE 3C. SPI MASTER TIMING CPOL = 0, CPHA = 0



NOTE:

1. Measurement points are V_{OL}, V_{OH}, V_{IL} and V_{IH}.

FIGURE 3D. SPI MASTER TIMING CPOL = 1, CPHA = 0

Serial Peripheral Interface (SPI) Timing Diagrams (Continued)

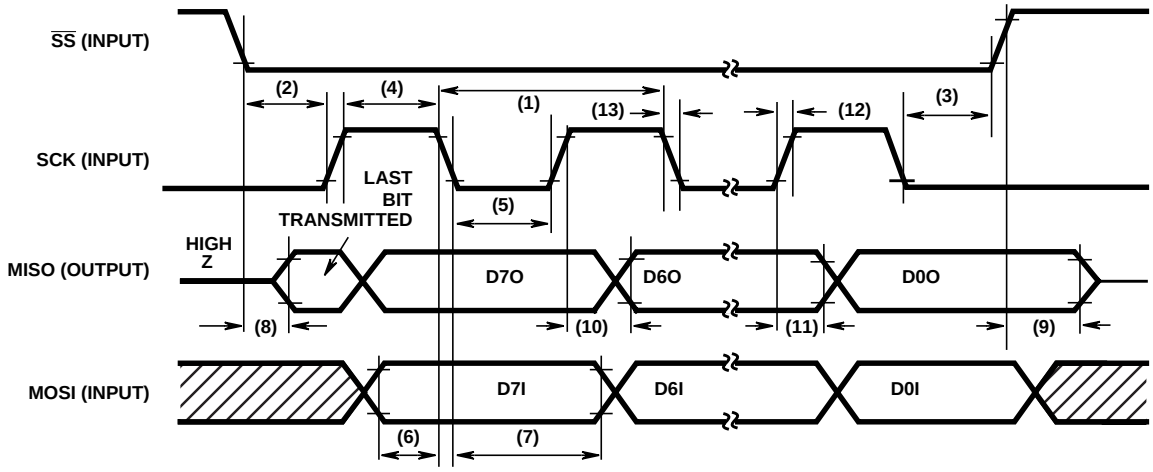
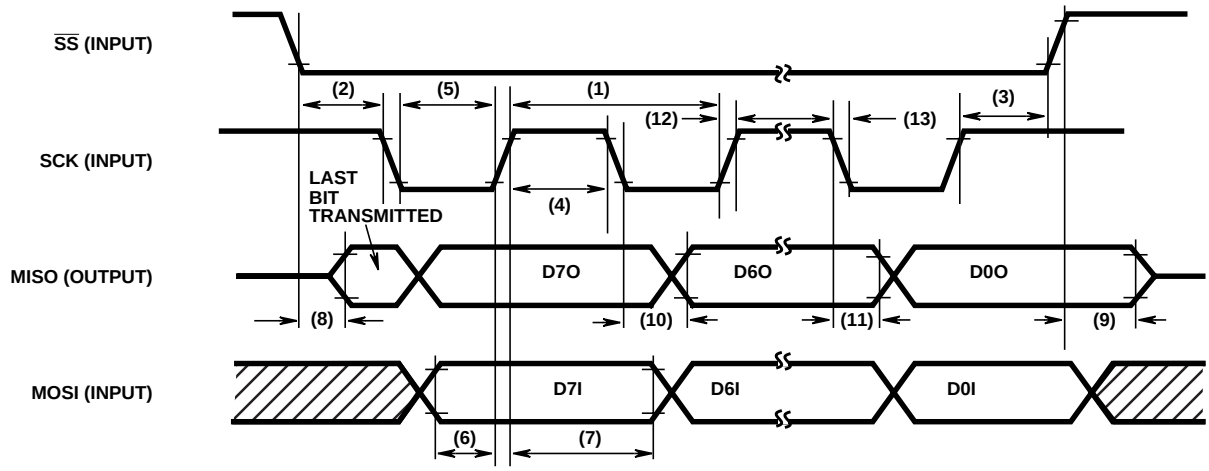


FIGURE 3E. SPI SLAVE TIMING CPOL = 0, CPHA = 1



NOTE:

1. Measurement points are V_{OL}, V_{OH}, V_{IL} and V_{IH}.

FIGURE 3F. SPI SLAVE TIMING CPOL = 1, CPHA = 1

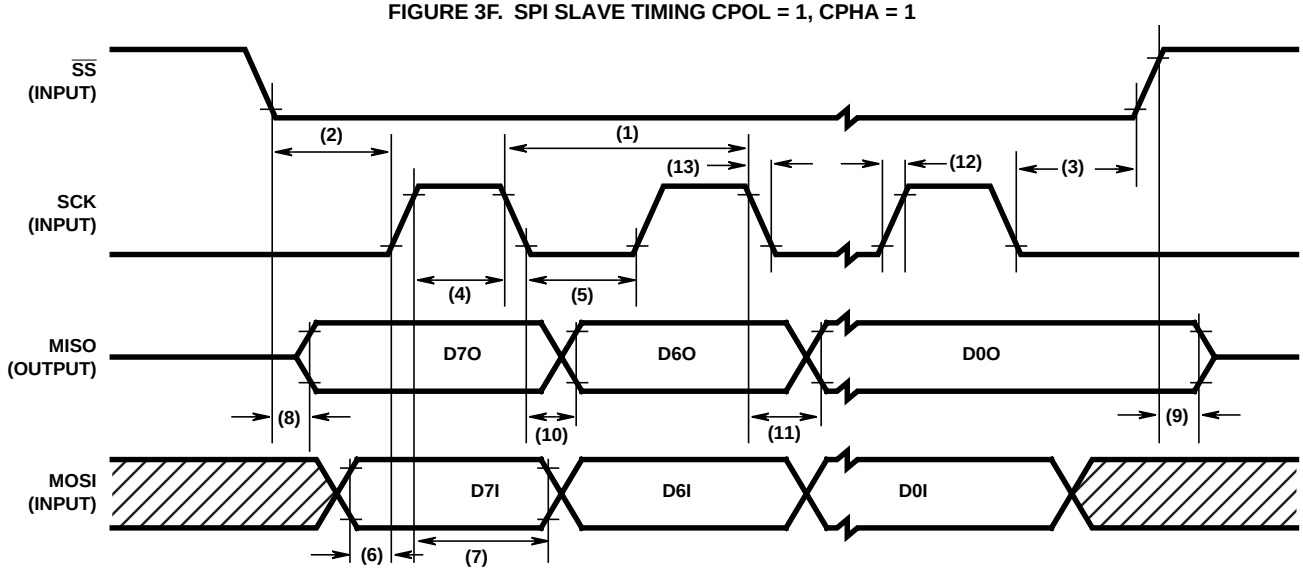
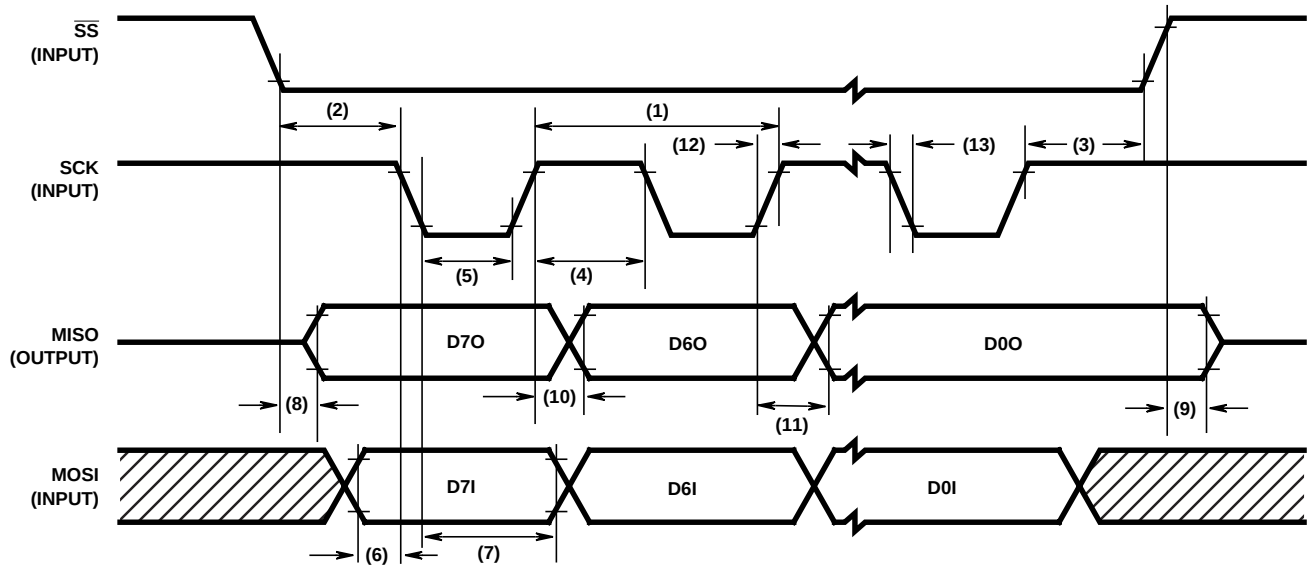


FIGURE 3G. SPI SLAVE TIMING CPOL = 0, CPHA = 0

Serial Peripheral Interface (SPI) Timing Diagrams (Continued)



NOTE:

1. Measurement points are V_{OL}, V_{OH}, V_{IL} and V_{IH}.

FIGURE 3H. SPI SLAVE TIMING CPOL = 1, CPHA = 0

Functional Pin Description

This section provides a description of each of the 28 pins of the HIP7030A2 MCU.

V_{DD} and V_{SS} (Power)

Power is supplied to the MCU using these two pins. V_{DD} is connected to the positive supply and V_{SS} is connected to the negative supply.

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

The $\overline{IRQ}$ pin is negative edge-sensitive triggering. A high to low transition on the $\overline{IRQ}$ pin will produce an interrupt.

In the event of an interrupt request, the MCU always completes the current instruction before it responds to the request. An internal mask can be used to inhibit the MCU from responding to IRQ interrupts.

An IRQ interrupt is generated if the $\overline{IRQ}$ pin is pulled low for at least one t_{ILIH}. The occurrence of the low going pulse is registered in a flip-flop and the IRQ interrupt will be recognized even if the $\overline{IRQ}$ pin has returned to a high state before the interrupt can be serviced.

Once the edge-sensitive flip-flop is cleared, (it is automatically cleared at the start of the interrupt service routine) the interrupt request is removed until the $\overline{IRQ}$ pin returns to a high level and once again goes low.

See *INTERRUPTS* for more details concerning IRQ interrupts.

$\overline{RESET}$ (Master Reset - Input)

The HIP7030A2 contains an integrated Power-On Reset (POR) circuit and the $\overline{RESET}$ input is therefore not required for start-up. It can be used to reset the MCU internal state and provides for an orderly re-start of the software after initial power-up. Refer to *Resets* for a detailed description of POR and $\overline{RESET}$.

TCAP (Timer Capture - Input)

The TCAP input controls the input capture feature for the on-chip programmable timer system. The TCAP input is also used as the strobe signal to the Port D strobed outputs. Refer to *Input Capture Register* and *PD0, PD1 Strobed Output Mode* for additional information.

TCMP (Timer Compare - Output)

The TCMP pin provides an output for the output compare feature of the on-chip timer system. Refer to *Output Compare Register* for additional information.

OSCIN (Oscillator Input - Input), OSCOUT (Oscillator Output - Output), OSCB (Oscillator Buffered Output - Output)

OSCIN is the input and OSCOUT is the output of an inverter/amplifier which can be used to build either a quartz crystal or ceramic resonator based clock oscillator. Alternatively, the OSCIN input can be driven from any external clock source which satisfies the CMOS Schmitt trigger input level requirements of the OSCIN pin. See *Electrical Specifications* for input level specification.

OSCB is a squared, buffered version of the OSCIN signal, available for driving one external CMOS load.

The fundamental internal clock is derived by a divide-by-two of the external oscillator frequency (f_{OSC}). All other internal clocks are also derived from the external frequency. These clocks include the input to the 16-bit Timer, the Serial Clock (SCK), and the VPW Symbol Encoder/Decoder (SENDEC).

Quartz Crystal

The circuit shown in Figure 4A is recommended when using a quartz crystal. The internal oscillator is designed to interface with an AT-cut parallel resonant quartz crystal in the frequency range specified for f_{OSC} in *Electrical Specifications*. Figure 4B lists the recommended capacitance and feedback resistance values. 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 OSCIN and OSCOUT pins to minimize output distortion and start-up stabilization time.

Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost sensitive applications. The circuit in Figure 4A is recommended when using a ceramic resonator. Figure 4C lists the recommended capacitance and feedback resistance values. The manufacturer of the particular ceramic resonator being considered should be consulted for specific information.

External Clock

If an external clock is used, it should be applied to the OSCIN input with the OSCOUT output not connected, as shown in Figure 4E. The t_{OXOV} or t_{ILCH} specifications do not apply when using an external clock input. The equivalent specification of the external clock source should be used in lieu of t_{OXOV} or t_{ILCH}.

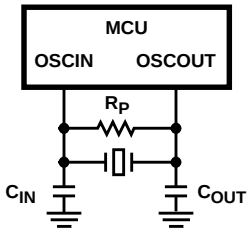


FIGURE 4A. CRYSTAL/RESONATOR CIRCUIT CONNECTIONS

	2MHz	4MHz	8MHz	10MHz	UNITS
R_S (Max)	400	75	50	30	Ω
C_0	5	7	5	5	pF
C_1	0.008	0.012	0.015	0.018	µF
C_{IN}	15-40	15-30	12-30	12-30	pF
C_{OUT}	15-30	15-25	12-25	12-25	pF
R_P	1-10	1-10	1-10	1-10	MΩ
Q	30	30	30	30	K

FIGURE 4B. QUARTZ CRYSTAL PARAMETERS

	10MHz	UNITS
R_S (Typical)	5.5	Ω
C_0	35	pF
C_1	5	pF
C_{IN}	22	pF
C_{OUT}	22	pF
R_P	1-5	MΩ
Q	500	-

NOTES:

1. When no power is applied to the HIP7030A2, the OSCIN, $\overline{IRQ}$, $\overline{RESET}$, and $\overline{VPWIN}$ pins can have up to $9V_{DC}$ applied with no side effects.
2. When power is applied to the HIP7030A2, it is recommended that all unused inputs, except Port A and Port D I/O lines configured as outputs, be tied to an appropriate logic level (i.e., either V_{DD} or V_{SS}).

FIGURE 4C. CERAMIC RESONATOR PARAMETERS

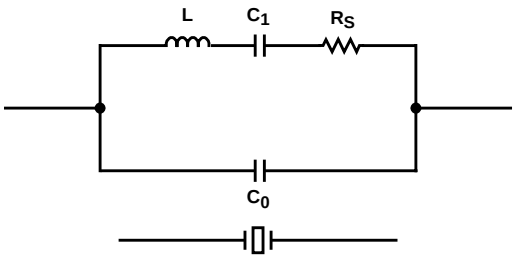


FIGURE 4D. CRYSTAL/RESONATOR EQUIVALENT CIRCUIT

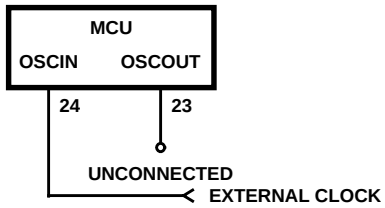


FIGURE 4E. EXTERNAL CLOCK SOURCE CONNECTIONS

PA0-PA7 (Port A - Input/Output)

These eight I/O lines comprise Port A. The mode (i.e., input or output) of each pin is software programmable. All Port A I/Os are configured as inputs during a POR, COP, or external reset. Refer to *Port A* under *Port A and D I/O Lines* for a detailed description of programming the Port A I/O lines.

PD0-PD4 (Port D - Input/Output)

These five I/O lines comprise Port D. As with PA0-PA7, the mode (i.e., input or output) of each pin is software programmable. In addition, a Special Function Register (SFRD) allows configuring PD0 and PD1 as “strobed” outputs, and/or PD2, PD3, and PD4 as inputs to an on-chip analog comparator.

All Port D I/Os are configured as inputs during a POR, COP, or external reset. Refer to *PD0-PD4 Special Function I/O Lines* under *Port A and D I/O Lines* for a detailed description of programming the Port D I/O lines.

VPWOUT (Variable Pulse Width Out - Output),
$\overline{\text{VPWIN}}$ (Variable Pulse Width In - Input)

These two lines are used to interface to the J1850 bus transceiver.

VPWOUT is the pulse width modulated output of the SENDEC encoder block.

$\overline{\text{VPWIN}}$ is the inverted input to the SENDEC decoder block.

See *VPW Symbol Encoder/Decoder (SENDEC)* for a detailed description of the J1850 interface pins.

MISO (Master-in/Slave-out - Input/Output),
MOSI (Master-out/Slave-in - Input/Output),
SCK (Serial Clock - Input/Output),
$\overline{\text{SS}}$ (Slave Select - Input)

These four lines constitute the Serial Peripheral Interface (SPI) communications port. The MCU can be configured as a SPI “master” or as a SPI “slave”. In master mode MOSI and SCK function as outputs and MISO functions as an input. In slave mode MOSI and SCK are inputs and MISO is an output. $\overline{\text{SS}}$ is always an input.

Serial data words are transmitted and received over the MISO/MOSI lines synchronously with the SCK clock stream. The word size is fixed at 8-bits. Single buffering is used which results in an inherent inter-byte delay. The master device always provides the synchronizing clock.

A low on the $\overline{\text{SS}}$ line causes the MCU to immediately assume the role of slave, regardless of it's current mode. This allows multi-master systems to be constructed with appropriate arbitration protocols.

See the detailed discussion of the SPI interface under *Serial Peripheral Interface (SPI)*.

Integrated Hardware I/O Functions

PORT A

Each of the Parallel Port pins of Port A may be individually programmed as an input or an output under software control. The direction of each pin is determined by the state of the corresponding bit in the Port A Data Direction Register (DDRA, location $04).

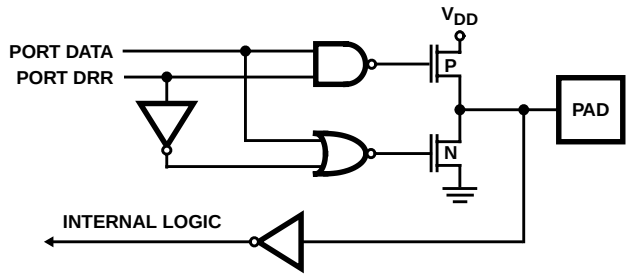


FIGURE 5A. PORT A I/O PAD CIRCUITRY

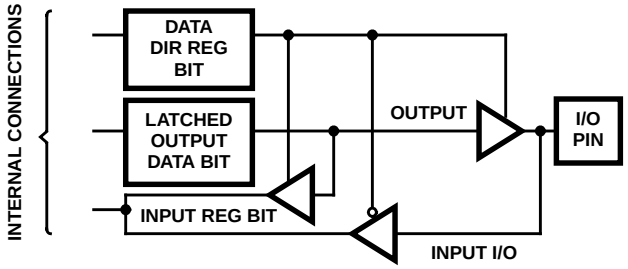


FIGURE 5B. PORT A FUNCTIONAL BLOCK DIAGRAM

7	6	5	4	3	2	1	0
DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0

PORT A DATA DIRECTION REGISTER (DDRA, LOCATION $04)

Any Port A 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. Any reset will clear all DDR bits, which configures all Port A and D pins as inputs. The data direction register is capable of being written to or read by the processor. Refer to Figure 5 and Table 1.

7	6	5	4	3	2	1	0
A7	A6	A5	A4	A3	A2	A1	A0

PORT A DATA REGISTER (PORTA, LOCATION $00)

Port A is an 8-bit wide read-write data register. Regardless of the state of the DDRA bits, all Port A data latches are modified with each write to Port A. When Port A is read, the value read for bits programmed as outputs, is the contents of the data latch, not the pin. The value read for bits programmed as inputs is the value on the pin.

TABLE 1. PORT A TRUTH TABLE

(NOTE 1) R/W	DDR	I/O PIN FUNCTION
W	0	The I/O pin is in input mode. Data is written into the output data latch
W	1	Data is written into the output data latch and simultaneously output to the I/O pin.
R	0	The state of the I/O pin is read.
R	1	The I/O pin is in output mode. The output data latch is read.

NOTE:

1. R/W is an internal signal which equals R when reading the Port Data Register and equals W when writing the Port Data Register.

PD0-PD4 SPECIAL FUNCTION I/O LINES

These five lines comprise Port D. The five lines can be individually programmed to provide input or output capabilities similar to the eight Port A lines. Additionally, each of the lines

can be programmed to provide special capabilities, beyond the standard digital input and output functions. The PD0-PD4 I/O lines are controlled via three read/write registers.

7	6	5	4	3	2	1	0
0	0	0	DD4	DD3	DD2	DD1	DD0

PORT D DATA DIRECTION REGISTER (DDRD, LOCATION $07)

DDRD contains five data direction bits, DD0-DD4, which control whether the associated I/O line behaves as an Input or as an Output. Setting a data direction bit causes the related I/O line to be configured as an output, while clearing the bit causes the line to be configured as an input. When configured as an output, the I/O line is actively driven by the HIP7030A2. When configured as an input, the I/O line appears as a high impedance input and should be driven by external circuitry.

DDRD bits D0-D4 are cleared by RESET.

7	6	5	4	3	2	1	0
CMP3	CMP2	0	D4	D3	D2	D1	D0

PORT D DATA REGISTER (PORTD, LOCATION $03)

PortD is an 8-bit wide register with 5 read/write data bits and 2 read-only bits. When writing to PortD, bits 5-7 are ignored. All other bits (D0-D4) are stored in latches until they are explicitly modified with a subsequent write (or read-modify-write) instruction. The utilization of bits D0-D4 is dependent on the value in the associated DDRD bit. If a line is programmed as an input, the value read in PortD (D0-D4) is the logic level present on the external I/O line. If a line is programmed as an output, the value read in PortD (D0-D4) is the value last written to the same bit in PortD and that value is forced onto the corresponding I/O line. The PortD CMP2 and CMP3 read-only input bits indicate the results of the last analog comparisons (see *PD2, PD3, PD4 Analog Comparator Inputs* for details on CMP2 and CMP3). PortD bit 5 is always read as a 0.

PortD is not affected by RESET.

7	6	5	4	3	2	1	0
0	0	CMPE	0	0	0	STE1	STE0

PORT D SPECIAL FUNCTION REGISTER (SFRD, LOCATION $08)

SFRD is an 8-bit wide register with 3 read/write control bits. The Strobe Enable 0 and 1-bits (STE0 and STE1) and used to configure PD0 and PD1 as strobed outputs. STE0 and STE1 only affect PD0 and PD1 when they are programmed as outputs by setting the corresponding bits in DDRD. See *PD0, PD1 Strobed Outputs* for a detailed explanation. The Comparator Enable bit (CMPE) controls the HIP7030A2's auto-zeroing, analog comparator (see *PD2, PD3, PD4 Analog Comparator Inputs* for details on CMPE). SFRD bits 2, 3, 4, 6 and 7 are always read as a 0.

SFRD bit CMPE, is cleared by RESET. All other SFRD bits are unaffected by RESET.

PD0, PD1 Strobed Output Mode

(DD0/DD1) is set, setting the STE0/1-bit configures the PD0/1 output in strobed mode. Clearing the STE0/1-bit causes the PD0/1 output to function identically to a PortA line in output mode. If the DDRD direction bit is clear, the associated line functions as an input and the state of the STE bit has no effect. When programmed as strobed outputs, data written to Port D Data Register bits 0 and 1 will appear on the external PD0 and PD1 pins synchronously with a low to high transition on the TCAP pin. This same transition on TCAP can be programmed to generate an interrupt to the processor. See *Programmable Timer* for details on using the interrupt capabilities of the TCAP pin. The strobed output mode of PD0 and PD1, coupled with the interrupt capability of TCAP, provides a mechanism for synchronously passing two bits of data between the HIP7030A2 and an external, asynchronous device.

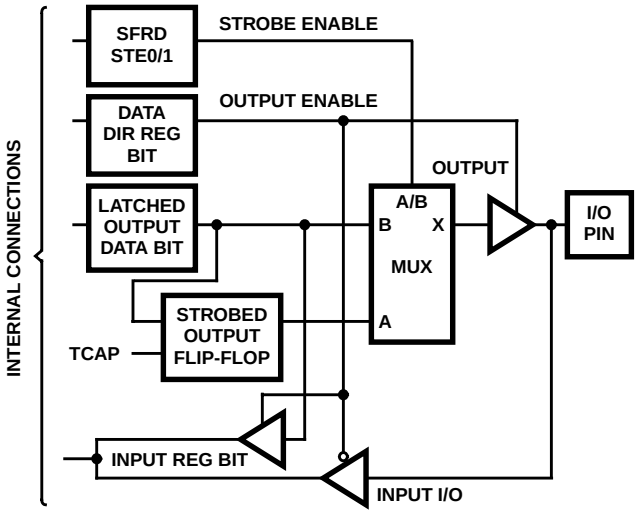


FIGURE 6. STROBED OUTPUT BLOCK DIAGRAM (PD0, PD)

STE0 and STE1 are not affected by RESET.

TABLE 2. PORT D STROBED OUTPUTS TRUTH TABLE

(NOTE 1) R/W	DDR	STE	I/O PIN FUNCTION
W	0	X	The I/O pin is in input mode. Data is written into the output data latch.
W	1	0	Data is written into the output data latch and simultaneously output to the I/O pin.
W	1	1	Data is written into the output data latch and transferred to the I/O pin on the next TCAP low to high transition.
R	0	X	The state of the I/O pin is read.
R	1	0	The I/O pin is in standard output mode. The output data latch is read.
R	1	1	The I/O pin is in strobed output mode. The output data latch is read.

NOTE:

1. R/W is an internal signal which equals R when reading the Port Data Register and equals W when writing the Port Data Register.

PD2, PD3, PD4 Analog Comparator Input Mode

When the CMPE bit is low in SFRD PD2, PD3, and PD4 behave as standard bidirectional I/O pins. Each of these three pins can be programmed as an input pin by setting the associated DDR bit low. Setting the DDR bit high configures the pin as an output. When CMPE is set high the three pins are connected to the appropriate comparator inputs and the contents of the DDRD doesn't affect comparator operation. While it is possible to perform comparisons of the pins when they are in the output mode (DDR bits are set high) the comparator result of comparing two equal digital values is not predictable. The comparator is intended for comparing analog input values, in which case the DDR bits must be set low to configure the pins as inputs. When CMPE is high, all of the associated PortD digital inputs (bits D4, D3, and D2 of PortD) are forced to 0, to conserve power.

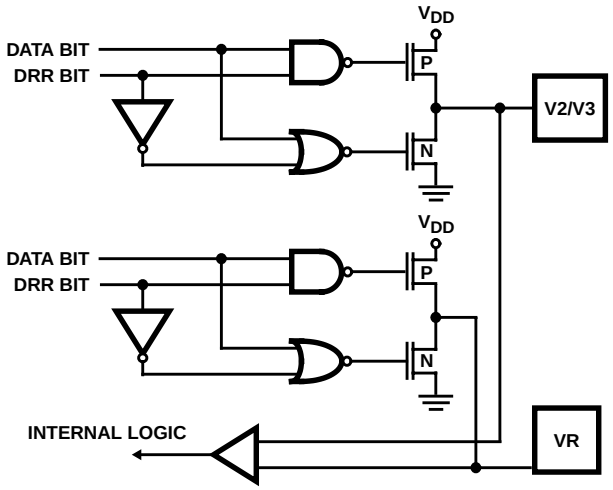


FIGURE 7. ANALOG INPUT I/O PINS

The circuitry of the clocked comparator consists of a differential amplifier with requisite current sources, auto-zero storage elements, and multiplexing switches. It is convenient to view it as a conventional differential comparator to which PD4 is connected as a "reference" at the negative input and PD2 and PD3 are connected via a multiplexer at the positive input. The comparator is enabled be setting the CMPE bit (bit 5) of SFRD and disabled by clearing CMPE. To conserve power the comparator should be disabled when not in use. RESET clears the CMPE bit. The three analog inputs function properly with inputs from -0.3V to V_{DD} +0.3V.

In order to use the comparator, PD4 and either (or both) PD2 or(and) PD3 should be selected as inputs via the DD2, DD3, and DD4 bits in DDRD. The results of the last comparison are available each time the PortD register is read. The CMP2 bit of PortD is set if PD2 was greater than PD4 during the last comparison and cleared otherwise. Similarly, the CMP3 bit of PortD is set if PD3 was greater than PD4 during the last comparison and cleared otherwise. CMP2 and CMP3 are not affected by RESET.

The HIP7030A2 includes a hardware sequencer to control the auto-zero function and input multiplexer of the comparator. Each complete compare cycle consists of a series of:

1. Auto Zero
2. Compare V2, write results to CMP2
3. Auto Zero
4. Compare V3, write results to CMP3

The hardware sequencer is enabled via the CMPE bit. Once enabled the compare cycling is performed continuously at a 1MHz step rate until CMPE is set low. A complete cycle takes 4µs. It follows that, at any given time, the results read in CMP2 or CMP3 of the PortD Data Register can be, at most, 4µs old.

The auto-zero operation involves charging a pair of bias capacitors. The charging time depends on the source impedance of the analog inputs, the relative voltages of V2 and V3, and the slew rate of all three input voltages. Incomplete charging of the capacitors will affect the accuracy of the comparator. The comparator is intended to perform favorably with input impedances up to 10kΩ and moderate slew rates.

J1850 Bus Interface

The VPW Symbol Encoder/Decoder (SENDEC) block provides the design with all the features needed to send and receive properly timed messages on a J1850 Class B Multiplexed Bus. Refer to *VPW Symbol Encoder/Decoder (SENDEC)* for detailed documentation on the use of the SENDEC.

16-Bit Timer

The integrated 16-bit Timer includes both capture and compare features. External events can be timed, pulses generated, and periodic interrupts programmed. A sophisticated set of control and status registers allows interrupt or polled operation. For a detailed guide to the operation of the Timer refer to *Programmable Timer*.

Serial Peripheral Interface (SPI)

The serial peripheral interface (SPI) is a synchronous serial interface with separate input, output, and clock lines. The SPI uses the MISO (serial data input/output), MOSI (serial data output/input), SCK (serial clock), and SS (slave select) pins. Refer to *Serial Peripheral Interface* for a detailed discussion of the SPI system.

Memory Organization

The HIP7030A2 MCU is capable of addressing 8192 bytes of memory and I/O registers with its program counter. The MCU has implemented 2520 bytes of these locations as shown in Figure 8. The first 256 bytes of memory (page zero) include: 24 bytes of I/O features such as data ports, the port DDRs, Timer, serial peripheral interface (SPI), and J1850 VPW Registers; 48 bytes of user ROM, and 176 bytes of RAM. The next 2048 bytes complete the user ROM. The Built-In-Test ROM (228 bytes) and Built-In-Test vectors (14 bytes) are contained in memory locations $1F00 through $1FF1. The 14 highest address bytes contain the user defined reset and the interrupt vectors. Eight bytes of the lowest 32 memory locations are unused and the 176 bytes of user RAM include up to 64 bytes for the stack. 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.

CPU Registers

The CPU contains five registers, as shown in the programming model of Figure 9. The interrupt stacking order is shown in Figure 10.

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 pushdown/popup stack. When accessing memory, the most significant bits are permanently configured to 0000011. These 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), thus, losing the previously stored information. A subroutine call occupies two RAM bytes on the stack, while an interrupt uses five RAM bytes.

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.

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 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 Communications Interface, and Serial Peripheral Interface 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.

Built-In-Test (BIT)

The BIT test routines utilize the SPI interface of the HIP7030A2 to provide an efficient method to test devices, both at the component and board level. The BIT routines are invoked by resetting the HIP7030A2 while applying $9V_{DC}$ (through a 4.7kΩ resistor) to the $\overline{IRQ}$ pin and $5V_{DC}$ to the TCAP pin. After reset, the HIP7030A2 will begin executing the BIT code stored at locations $1F00-$1FF1. The COP system remains active during BIT. When the BIT program begins, the SPI is configured in the master mode. SPI transfers are therefore controlled by the HIP7030A2. The tester paces the transfers by driving the $\overline{IRQ}$ line low to initiate a SPI transfer. When the transfer is complete, the tester raises $\overline{IRQ}$ (to 5-$9V_{DC}$) to signal successful transfer and to prepare for the next transfer. A convenient means of driving the $\overline{IRQ}$ pin is to connect it to $9V_{DC}$ through a 4.7kΩ resistor and to drive it with an open-collector/collector device such as the collector of an NPN device with its emitter grounded.

Following reset, the HIP7030A2 waits for a command to be received from the tester by monitoring the $\overline{IRQ}$ line and the SPIF flag in the SSR. $\overline{SS}$ should normally be held high throughout the BIT procedure. If $\overline{SS}$ is low following reset, the BIT routine will immediately branch to location $5D and begin executing the program stored there.

There are five BIT functions which are accessible via the SPI. Each is selected by sending the associated command number to the HIP7030A2. Following completion of each command (except Command $00), the HIP7030A2 waits for another command. Commands outside of the range $00-$04 will be ignored.

Download (Command $00):

Download takes 175 bytes from the SPDR and writes them to RAM beginning at $50 and ending and $FE. After receiving the 175th byte, the program begins executing at location $5D. The 13 locations $50-$5C are used as a link table to

allow testing of the interrupt functions. The link locations are: SPI Interrupt $50-51; Timer Interrupt $52-53; IRQ Interrupt $54-55; SED Interrupt $56-57; COP Interrupt $58-59; SWI Interrupt $5A-5C. If any entries in the link table are not required for the downloaded program, they may be used for variables or subroutines. Interrupts are disabled when execution begins. If interrupts are enabled via a CLI instruction, handling of all interrupts which are generated (including the NEW interrupt) is required.

Location $5D is always the start location.

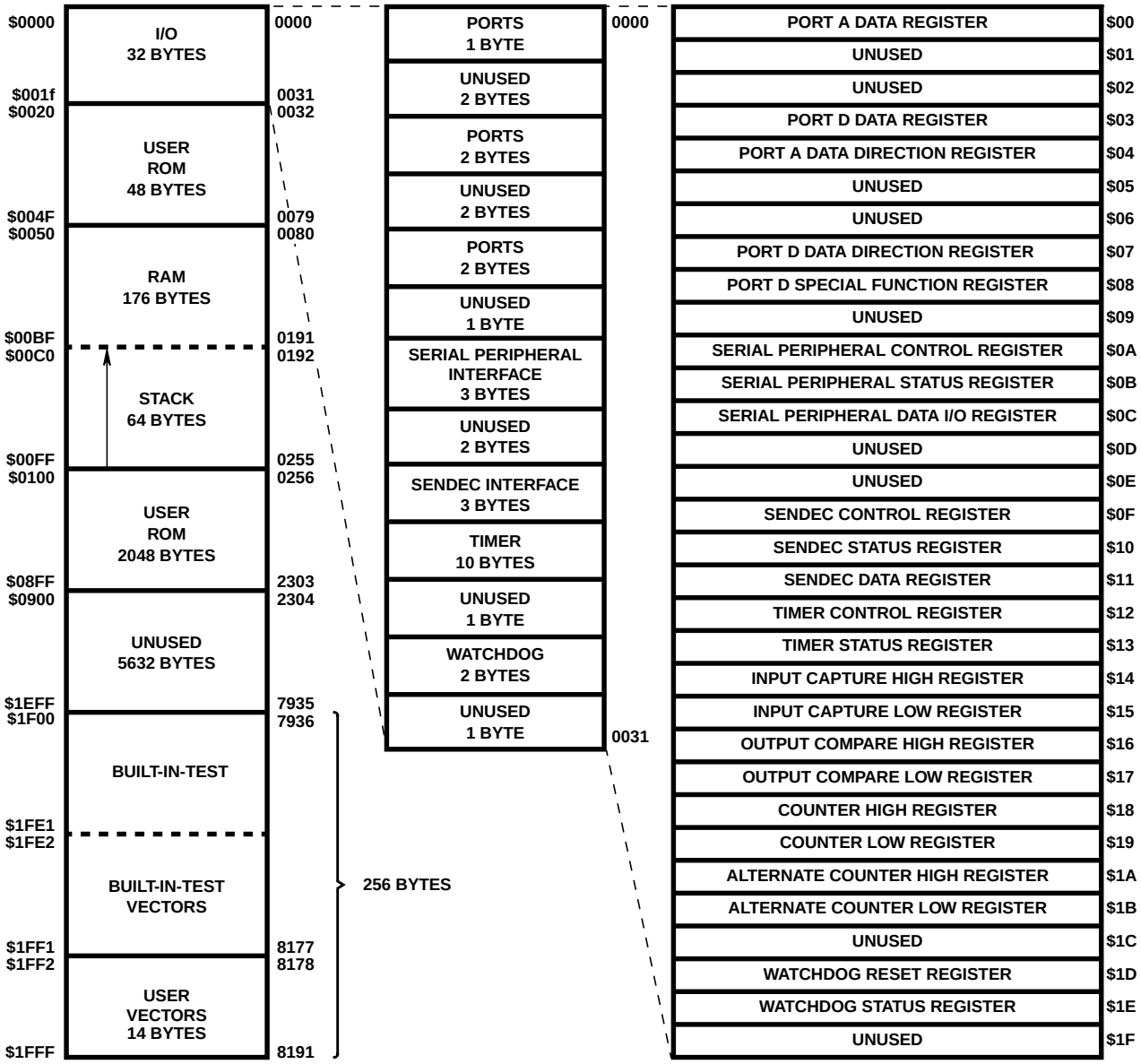


FIGURE 8. MEMORY MAP OF THE HIP7030A2

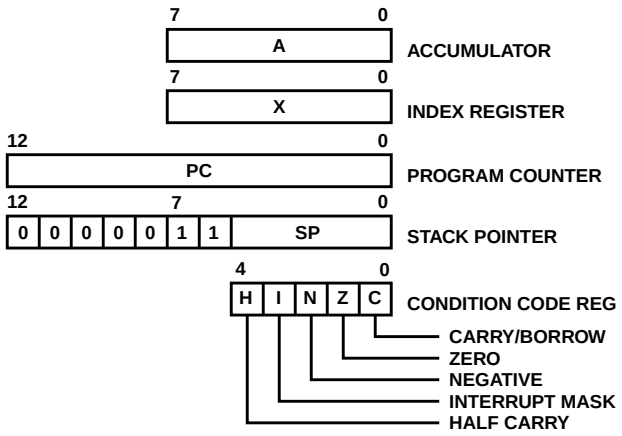


FIGURE 9. CPU REGISTER SET

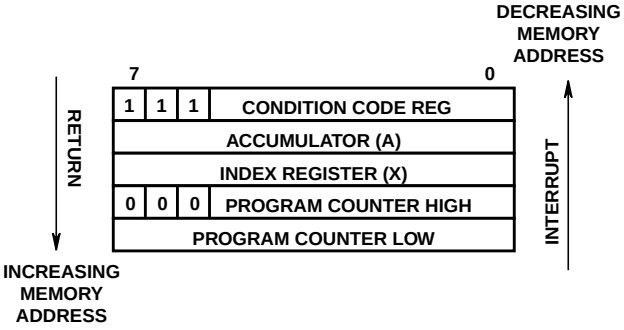


FIGURE 10. STACKING ORDER DURING INTERRUPTS

ROM Dump (Command $01):

ROM Dump reads the entire ROM contents and transfers them one byte at a time via the SPI register. The entire ROM must be read from start to end. Locations $20-$4F, $100-$8FF, and $1F00-$1FFF are read in lowest to highest address sequence. Following the complete ROM contents, a two byte checksum is transferred.

Read Page 0 (Command $02):

Allows reading of any page 0 byte (i.e., $00-$FF). The command ($02) is sent, followed by the address, followed by the data transfer. The data sent is ignored, while the return data is the contents of the specified address. The three byte sequence must be repeated for each transfer.

Write Page 0 (Command $03):

Allows writing of any non-ROM, Page 0 byte (i.e., $00-$1F or $50-$FF). The command ($03) is sent, followed by the address, followed by the data transfer. The data is written to the location, while the return data is the contents of the specified address. The three byte sequence must be repeated for each transfer. The procedure will work for ROM locations identically to all other locations, except the write will be supressed.

Read Mask ID (Command $04):

Transfers the three character custom mask ID assigned to each customer pattern. The command ($04) is sent, followed by three data transfers The data sent is ignored, while the return data is a three character ASCII sequence which uniquely identifies each customer's mask ROM pattern.

Resets

The MCU has three reset modes: an active low external reset pin ($\overline{\text{RESET}}$), a power-on reset function, and a Computer Operating Properly (COP) reset function.

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

The $\overline{\text{RESET}}$ input pin is used to reset the MCU to provide an orderly software start-up 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 a power-down reset. The power-on circuitry provides for a 4064 t_{CYC} delay from the time that the oscillator becomes active to allow for stabilization (see Figure 11). If the external $\overline{\text{RESET}}$ pin is low at the end of the 4064 t_{CYC} time out, the processor remains in the reset condition until $\overline{\text{RESET}}$ goes high. Table 3 shows the actions of the two resets on internal circuits, but not necessarily in order of occurrence (X indicates that the condition occurs for the particular reset).

COP Reset

The COP reset is generated by either of two events: 1) the Watchdog Timer reaches its maximum value prior to being cleared, or 2) the Slow Clock Detect circuitry doesn't detect a transition on the OSCIN pin during a period of approximately 2µs. The COP reset is identical to an external $\overline{\text{RESET}}$ pin reset, except the Program Counter is loaded with the address at $1FFA-$1FFB instead of the address at $1FFE-$1FFF and the Watchdog Flag (bit 0) is set in the Watchdog Status Register (WSR, location $1E). COP Resets are discussed under Interrupts.

Interrupts

Systems often require that normal processing be interrupted so that some external event may be serviced. The HIP7030A2 may be interrupted by one of six different methods: either one of four maskable hardware interrupts ($\overline{\text{IRQ}}$, SPI, SENDEC, or Timer), one non-maskable Watchdog/Slow Clock Detect interrupt, and one non-maskable software interrupt (SWI). Interrupts such as Timer, SPI, and SENDEC have several flags which will cause the interrupt. Generally, interrupt flags are located in read-only status register, whereas, their equivalent enable bits are located in associated control registers. The interrupt flags and enable bits are never contained in the same register. 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 Figure 12) and the interrupt mask (I-bit) set to prevent additional interrupts. The appropriate interrupt vector then points to the starting address of the interrupt service routine (refer to Table 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 Figure 12.

NOTE: The interrupt mask bit (I-bit) will be cleared if and only if the corresponding bit stored in the stack is zero. A discussion of interrupts, plus a table listing vector addresses for all interrupts including RESET, in the MCU is provided in Table 4.

Hardware Controlled Interrupt Sequence

The following three functions (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 Figure 13, and for STOP and WAIT are provided in Figure 14. A discussion is provided below.

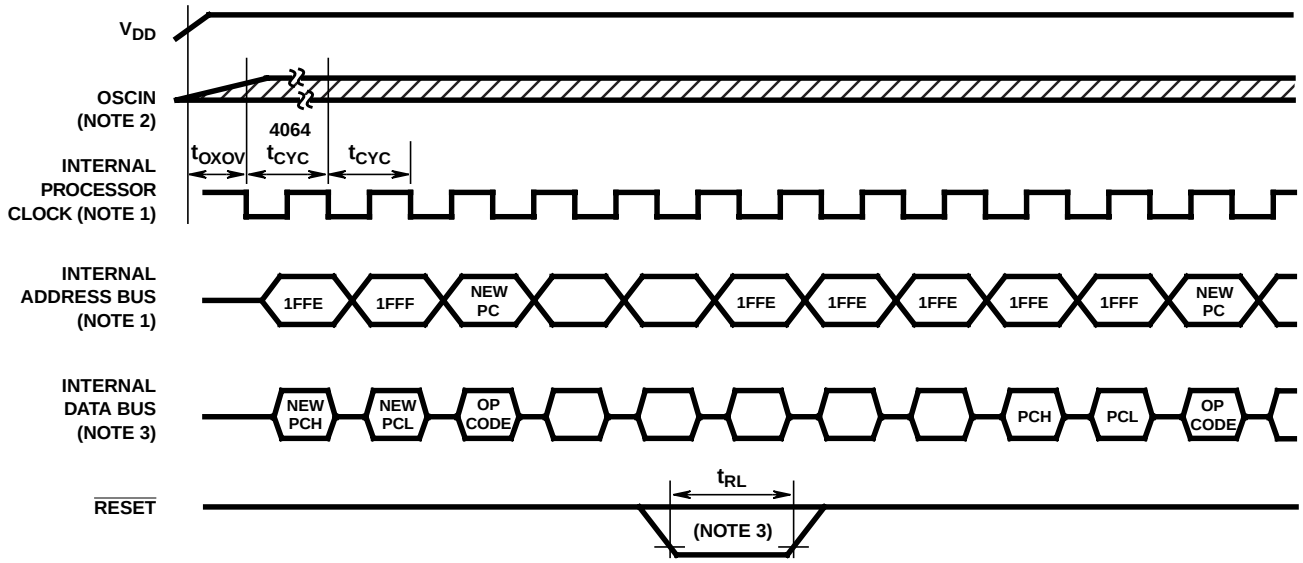
(a) **RESET** - A low input on the $\overline{\text{RESET}}$ input pin causes the program to vector to its starting address which is specified by the contents of memory locations $1FFE and $1FFF. The I bit in the condition code register is also set. Much of the MCU is configured to a known state during this type of reset as previously described in RESETS paragraph.

TABLE 3. $\overline{\text{RESET}}$ PIN, COP, AND POR ACTIONS ON INTERNAL CIRCUITRY

CONDITION	$\overline{\text{RESET}}$ PIN/ COP RESET	POWER-ON RESET
Timer Prescaler Reset to Zero State	X	X
Timer Counter Configure to $FFFC	X	X
Timer Output Compare (TCMP) Bit Reset to Zero	X	X
All Timer Interrupt Enable Bits Cleared (ICIE, OCIE, and TOIE) to Disable Timer Interrupts	X	X
Timer Output Level (OLVL) Bit is Cleared	X	X
Port A and Port D Data Direction Registers (DDRA and DDRD) Cleared to Zero, Placing all Port Pins in Input Mode	X	X
Port D Special Function Register Bit CMPE is Cleared Disabling Comparator	X	X
Set Stack Pointer (SP) to $00FF	X	X
Force Internal Address to RESET Vector ($1FFE)	X	X
Set I-Bit in Condition Code Register (CC) to 1; Disabling all Maskable Interrupts	X	X
Clear STOP Latch	X	X
Clear WAIT Latch	X	X
Reset Oscillator Stabilization Delay to 4064	(Note 1)	X
Clear External Interrupt (Irq) Flip-flop	X	X
VPWOUT Set Low (Passive State)	X	X
Slow Clock Detect Circuitry Reset	(Note 1)	X
Watchdog Timer Reset to Zero State	X	X
Watchdog Flag (WDF) Cleared/Set in Watchdog Status Register (WSR)	(Note 2)	(Note 2)
Watchdog Timer Interrupt Latch Cleared	X	X
Serial Peripheral Interface (SPI) Control Bits SPIE, MSTR, SPIF, WCOL, and MODF Cleared; Disabling SPI Interrupts and Setting to Slave Mode	X	X

NOTES:

1. Only if MCU is in STOP state; if NDEL is set in the SENDEC Control Register a delay of 128 is used for $\overline{\text{RESET}}$/COP.
2. WDF is cleared by POR and is set by a Watchdog Reset. $\overline{\text{RESET}}$ has no effect on WDF.



NOTES:

1. Internal signal and bus information is not available externally.
2. OSCIN is not meant to represent frequency. It is only meant to represent time.
3. The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

FIGURE 11. POWER-ON RESET AND $\overline{\text{RESET}}$

TABLE 4. VECTOR ADDRESSES FOR INTERRUPTS AND RESETS

REGISTER	FLAG NAME	INTERRUPTS SOURCE	INTERRUPT NAME	VECTOR ADDRESS
N/A	N/A	RESET	RESET	$1FFE-$1FFF
N/A	N/A	Software	SWI	$1FFC-$1FFD
N/A	N/A	Watchdog Timeout	COP	$1FFA-$1FFB
		Slow Clock Detect		
SENDEC Status (SEDSR)	TX	Transition Detected	SENDEC	$1FF8-$1FF9
	BRK	Break Detected		
	NEW	IFS or SOF		
	NECHO	Echo Failure		
N/A	N/A	External Interrupt	IRQ	$1FF6-$1FF7
TIMER Status (TSR)	ICF	Input Capture	TIMER	$1FF4-$1FF5
	OCF	Output Compare		
	TOF	Timer Overflow		
SPI Status (SPSR)	SPIF	Transfer Complete	SPI	$1FF2-$1FF3
	MODF	Mode Fault		

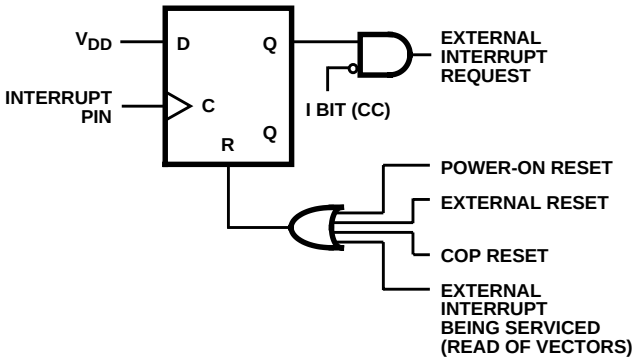


FIGURE 12A. INTERRUPT FUNCTION DIAGRAM

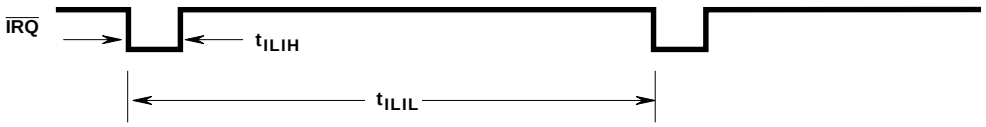


FIGURE 12B. INTERRUPT TIMING DIAGRAM

FIGURE 12. EXTERNAL INTERRUPT

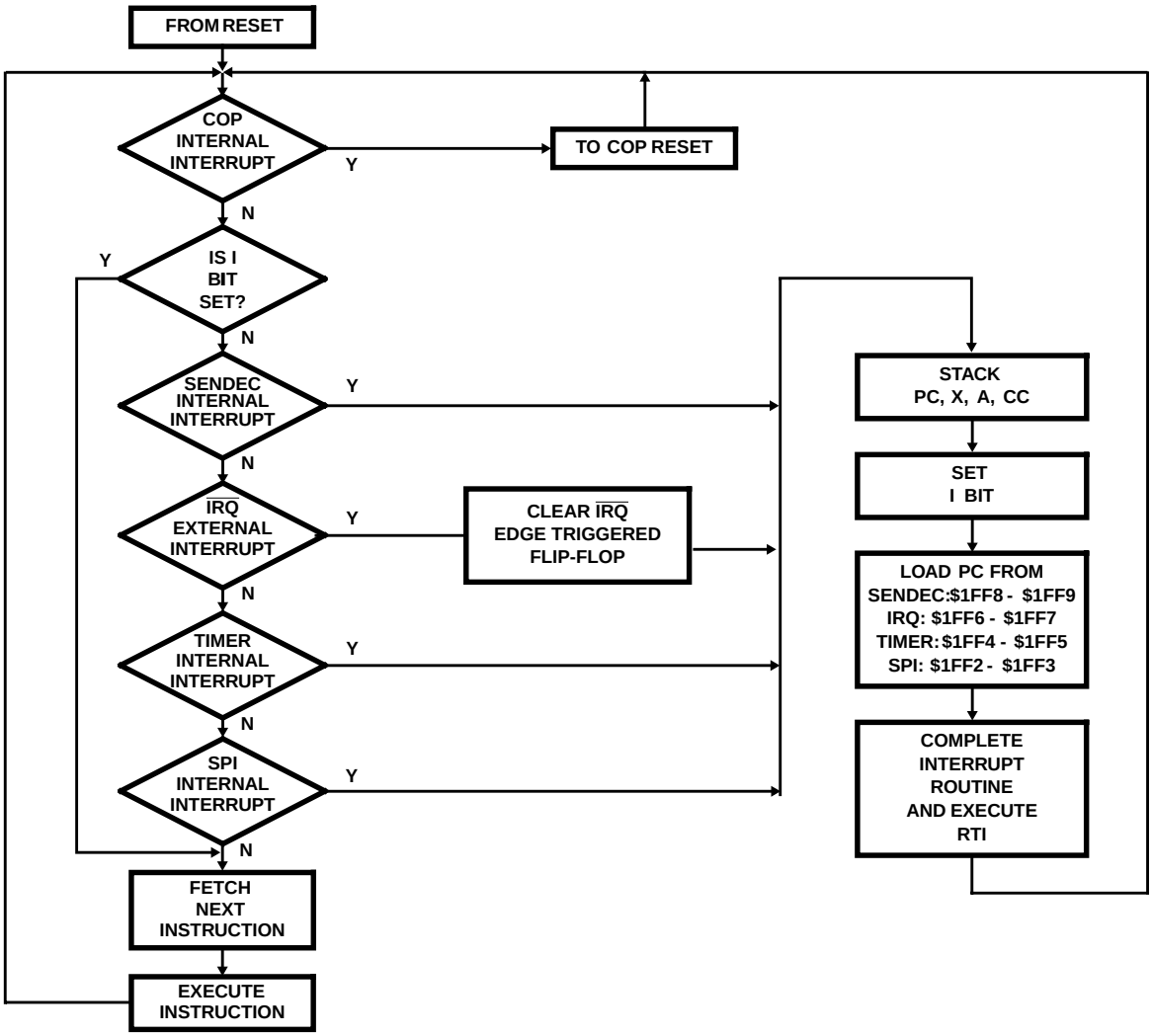


FIGURE 13. HARDWARE INTERRUPT FLOW DIAGRAM

(b) **STOP** - The STOP instruction causes the oscillator to be turned off and the processor to “sleep” until an external interrupt (IRQ) or reset occurs. The VPWOUT pin is forced to a low level, the SPI is set to slave mode, and all other pins remain at their previously set levels.

(c) **WAIT** - The WAIT instruction causes all processor clocks to stop, but leaves the Timer, the Watchdog, SENDEC, and SPI clocks running. This “rest” state of the processor can be cleared by RESET, Slow Clock Detect, an external interrupt (IRQ), Timer interrupt, SPI interrupt, or SENDEC interrupt. There are no special “wait mode” vectors for these 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.

COP Interrupt/Reset

The Computer Operating Properly (COP) system consists of two functions: 1) the Watchdog Timer, and 2) the Slow Clock Detect Circuit. These interrupts cause a complete system restart and the effect is identical in every respect to an externally generated $\overline{\text{RESET}}$ pin reset, except the restart address is taken from locations $1FFA and $1FFB and the Watchdog Flag (bit-0) is set in the Watchdog Status Register (WSR, location $1E) if the reset was the result of a Watchdog Timer overflow. Starting the PC with an address different than the standard RESET address allows the system to provide an appropriate response to the situation.

Because the CPU is reset during a COP Interrupt, the COP service routine must not exit with an RTI. Instead the routine should branch to subsequent code.

Since the most likely cause of a Slow Clock Detect is a malfunctioning oscillator, a COP Interrupt caused by a Slow Clock Detect will generally reset the MCU per Table 3 and remain in the RESET state.

SENDEC Interrupt

The VPW Symbol Encoder/Decoder (SENDEC) system has four different interrupt flags that will cause a SENDEC interrupt whenever they are set and enabled. These four interrupt flags are found in the SENDEC status register (SEDSR, location $10) and all will vector to the same interrupt service routine ($1FF8-$1FF9). One of the interrupt flags (TX - transition) has a corresponding enable bit in the SENDEC control register (SEDCR, location $0F). RESET clears the enable bit, thus, preventing a TX interrupt from occurring during the reset time period. The other three interrupts (BRK - break, NECHO - no echo, and NEW - new frame) are nonmaskable at the SENDEC level. The processor responds to all SENDEC interrupts 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 VPW Symbol Encoder/Decoder (SENDEC) for additional information about the SENDEC interrupts.

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 an 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 contents of memory location $1FF6 and $1FF7. Figure 12 shows both a functional and mode timing diagram for the interrupt line. The timing diagram shows treatment of the interrupt line (IRQ) for generating periodic interrupts to the processor. The figure 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 (for interrupt call and return delays). Once a pulse occurs, the next pulse should not occur until the MCU software has exited the routine (an RTI occurs).

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

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 ($1FF4-$1FF5). 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 time 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 $1FF4 and $1FF5. 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 Programmable Timer 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 $1FF2 and $1FF3 which contain the starting address of the 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 for a description of the SPI system and its interrupts.

Low Power Modes

STOP Instruction

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode the internal oscillator is turned off, causing all internal processing to be halted; refer to Figure 14. During the STOP mode, the I-bit in the condition code register is automatically cleared to enable external and SENDEC interrupts. All other registers and memory remain unaltered and all input/output lines remain unchanged, except the VPWOUT line which is forced to a passive (low) state and the SPI Master bit (MSTR) in the SPI Control Register (SPCR) which is cleared placing the SPI pins into Slave mode. This mode persists until an external interrupt (IRQ), a low on the $\overline{\text{VPWIN}}$ pin, or a low on $\overline{\text{RESET}}$ is sensed, at which time the internal oscillator is turned on. The interrupt or reset causes the program counter to vector to the starting address of the interrupt or reset service routine respectively.

WAIT Instruction

The WAIT instruction places the MCU 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, serial peripheral interface, Watchdog Timer, and SENDEC systems remain active. Refer to Figure 14. 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.

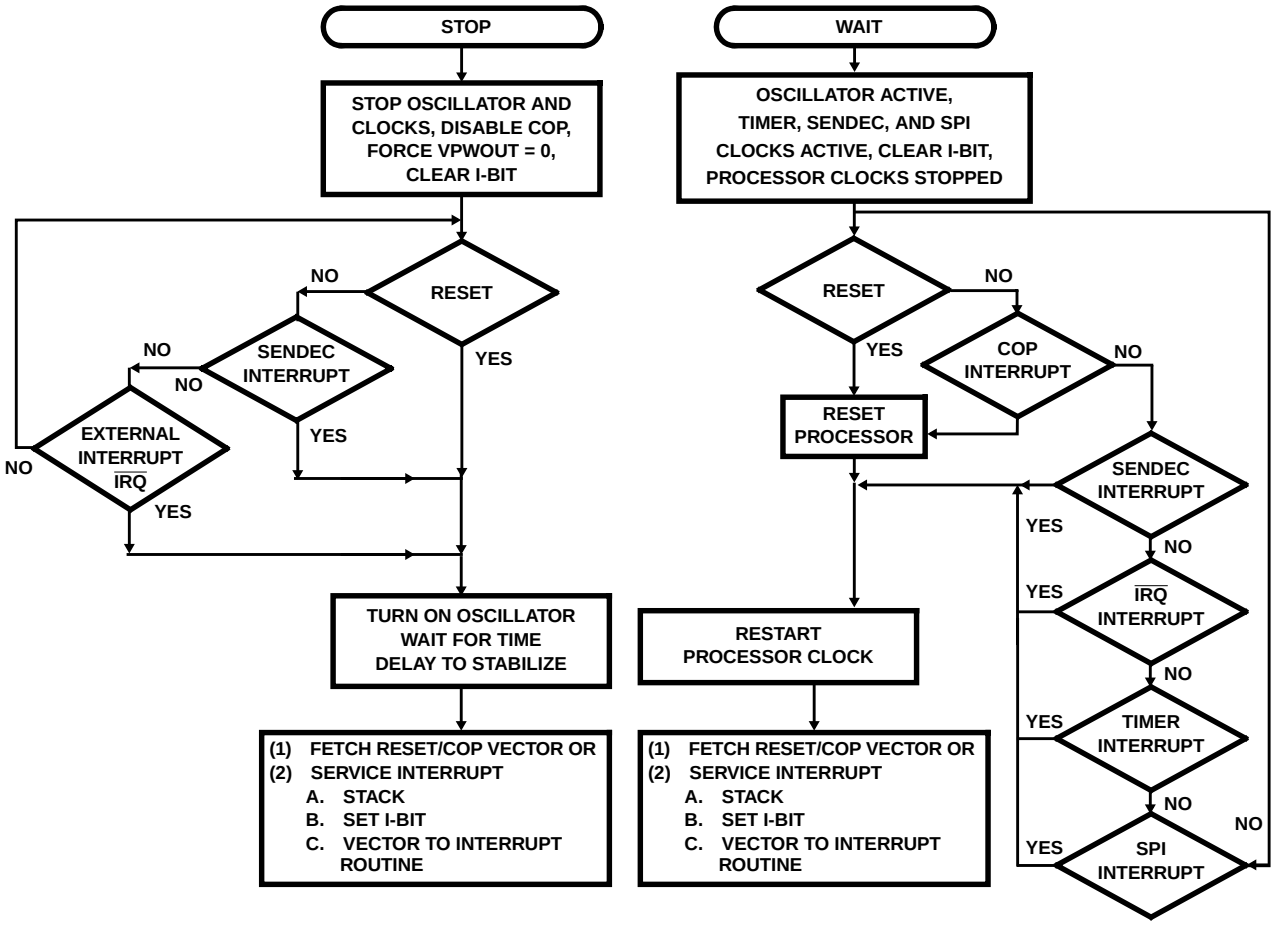


FIGURE 14. STOP AND WAIT FLOW DIAGRAM

Data Retention Mode

The contents of RAM and CPU registers are retained at supply voltages as low as $2V_{DC}$. This is referred to as the DATA RETENTION mode, where the data is held, but the device is not guaranteed to operate.

Programmable Timer

Introduction

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 Figure 15 and timing diagrams are shown in Figure 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.

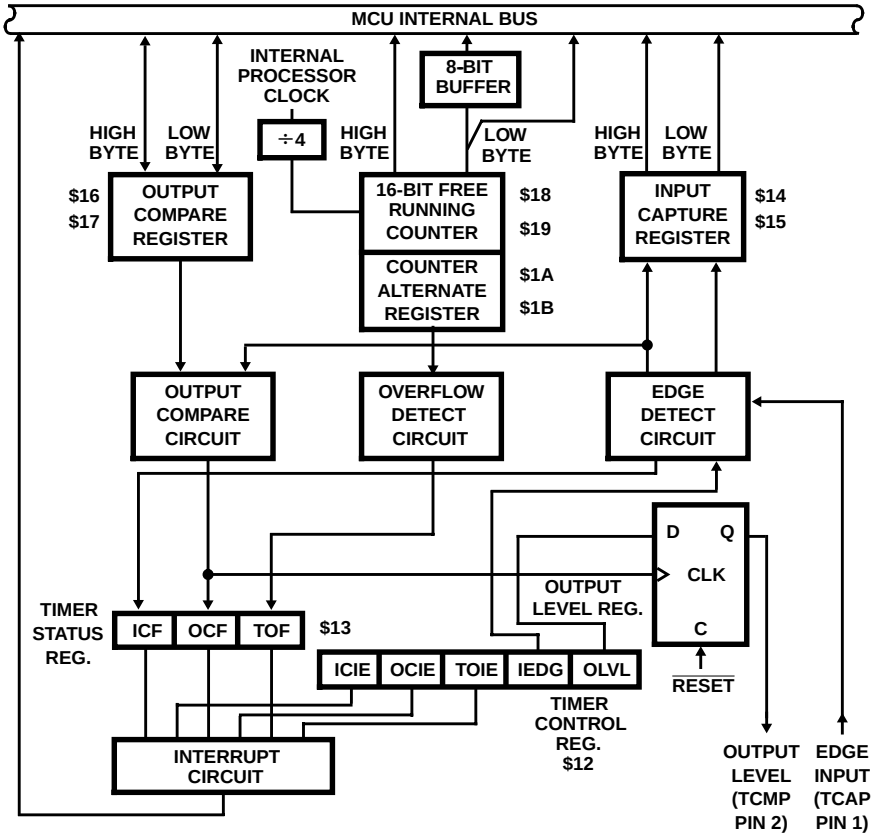
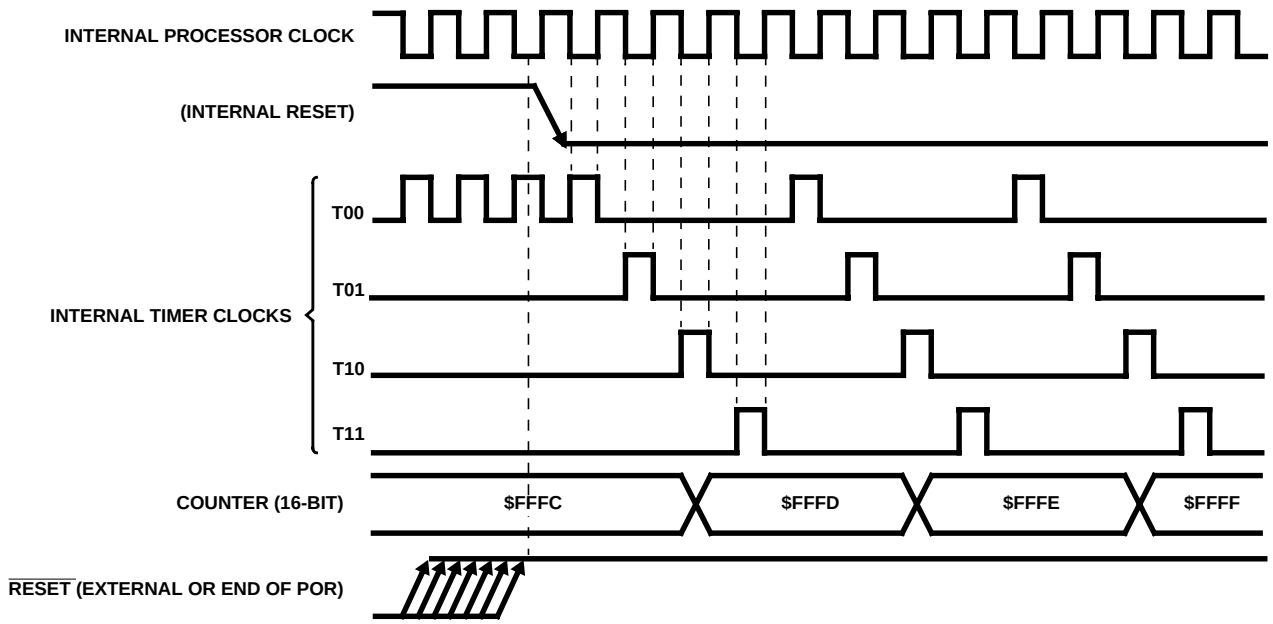


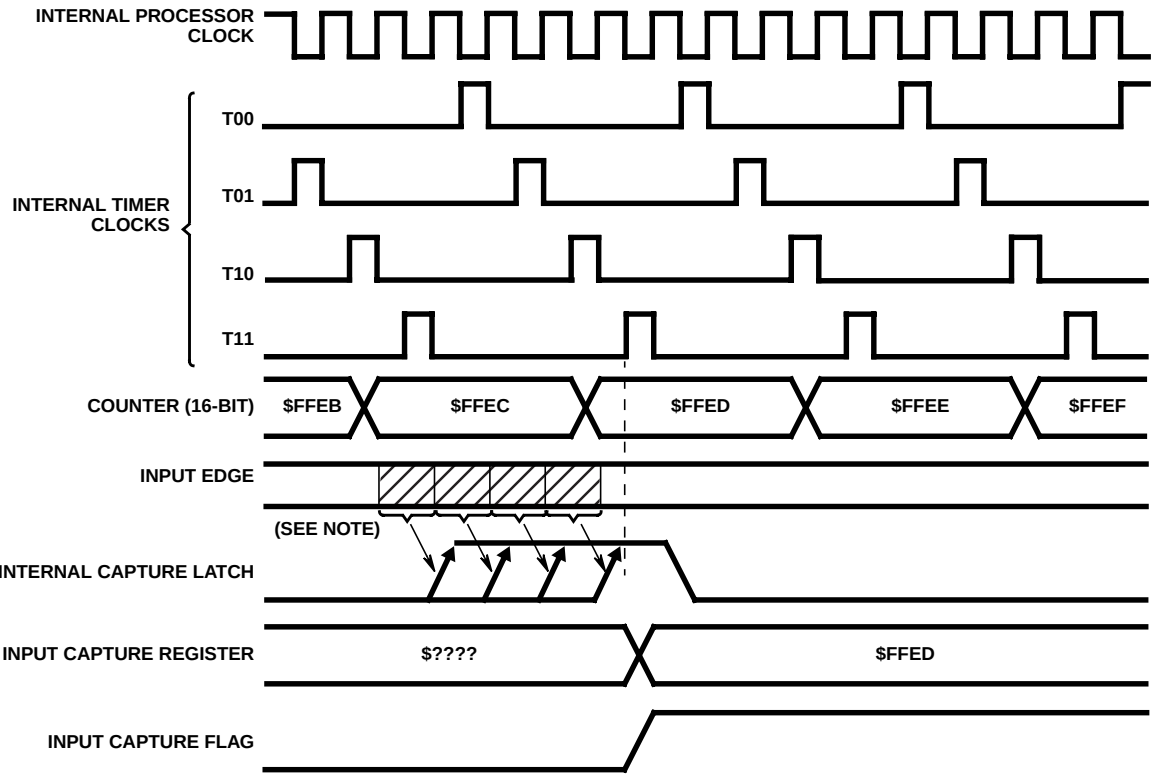
FIGURE 15. PROGRAMMABLE TIMER BLOCK DIAGRAM



NOTE:

1. The Counter Register and the Timer Control Register are the only ones affected by reset.

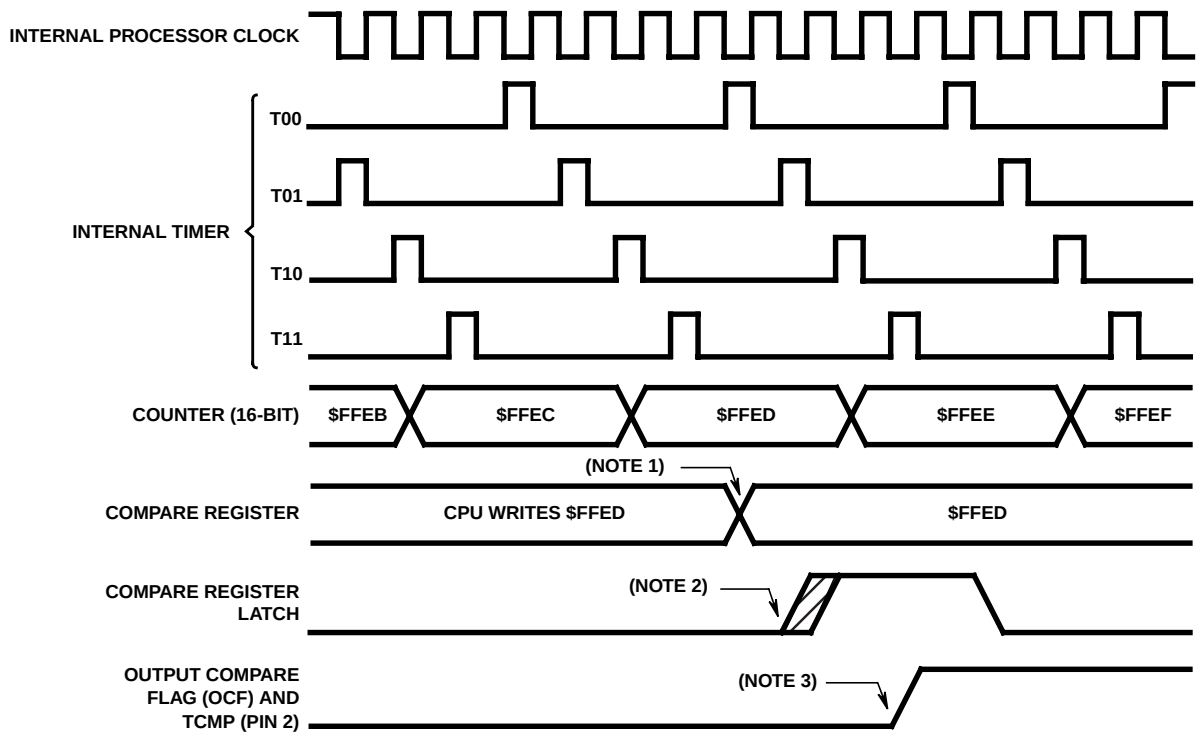
FIGURE 16. TIMER STATE DIAGRAM FOR RESET



NOTE:

1. The Counter Register and the Timer Control Register are the only ones affected by reset.

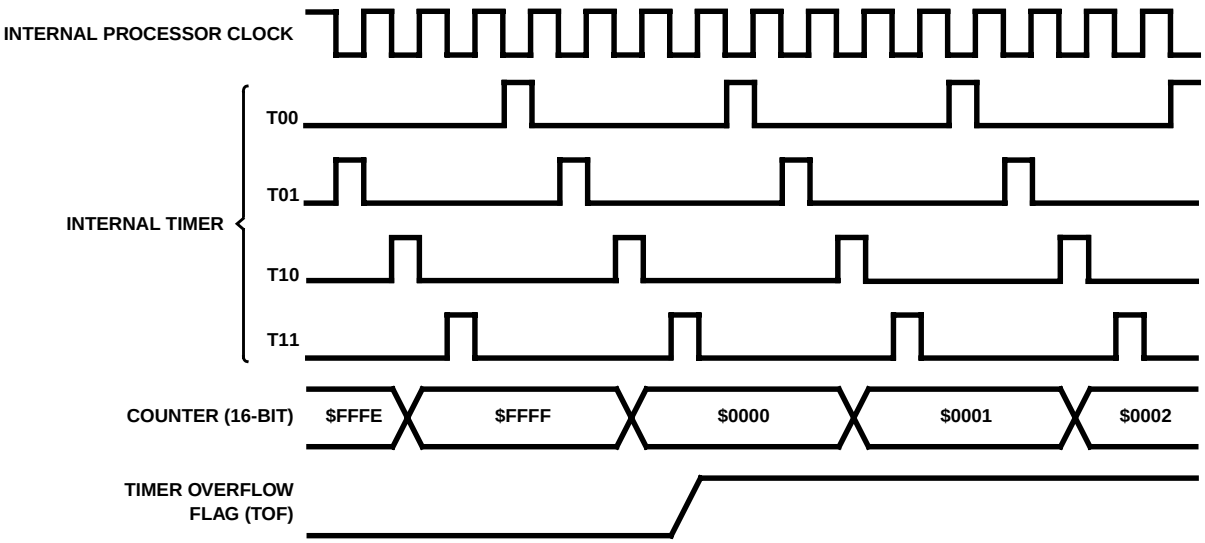
FIGURE 17. TIMER STATE DIAGRAM FOR INPUT CAPTURE



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).

FIGURE 18. TIMER STATE DIAGRAM FOR OUTPUT COMPARE



NOTE:

1. The TOF bit is set at timer state T11 (transition of the 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

FIGURE 19. TIMER STATE DIAGRAM FOR TIMER OVERFLOW

The I-1bit 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 below.

Timer Control Register (TCR) locations $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.

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 800ns if the internal processor clock is 5.0MHz. 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 of only the least significant byte (LSB) of the free running counter ($19, $1B) retrieves the current count value. If a read of the free running counter first addresses the most significant byte ($18, $1A) the least significant byte is 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 LSB of the free running counter or counter alternate register ($19, $1B), 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) or a COP reset, the counter is also configured to $FFFC and begins running after the oscillator start-up 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 writable 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 hardware.

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 the 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.

```
B716  STA OCMPHI;   INHIBIT OUTPUT COMPARE
B613  LDA TSTAT;    ARM OCF BIT IF SET
BF17  STX OCMPLO;   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). The selected edge

is also fed to the Port D strobed output pins (see Port D Strobed Output Mode). 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 Figure 17). This delay is required for internal synchronization. Resolution is affected by the prescaler allowing the timer to only increment every four internal processor clock cycles.

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. 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 needed operations. There is no conflict between the read of the input capture register and the free running counter transfer 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 five 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 two bits control: 1) which edge is significant to the capture edge detector (i.e., negative or positive), and 2) the next value to be clocked to the output level register in response to a successful output compare. 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	0	0	0	IEDG	OLVL

TCR (LOCATION $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.

B1, IEDG The value of the input edge (IEDG) bit determines which level transition on pin 1 will trigger a free running counter transfer to the input capture register. Reset does not affect 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 2. 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 the TCAP pin with an accompanying transfer of the free running counter contents to the input 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 Figures 13, 14, and 15 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

TSR (LOCATION $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 access of the timer status register (with ICF set) followed by accessing 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 match the contents of the free running counter. The OCF is cleared by accessing the timer status register (with OCF set) and then accessing 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 accessing the timer status register (with TOF set) followed by an access of the free running counter least significant byte ($19). Reset does not affect the TOF bit.

Accessing the timer status register satisfies the first condition required to clear any status bits which happen 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 or written 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 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: 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.

Serial Peripherial Interface (SPI)

Introduction

The serial peripheral interface (SPI) is an interface built into the MCU which allows several MCUs, or one MCU plus peripheral devices, to be interconnected within a single “black box” or on the same printed circuit board. In a serial peripheral interface (SPI), separate wires (signals) are required for data and clock. In the SPI format, the clock is not included in the data stream and must be furnished as a separate signal. An SPI system may be configured as one containing one master MCU and several slave MCUs, or in a system in which an MCU is capable of being either a master or a slave.

Figure 20 illustrates a typical multi-computer system configuration. Figure 20 represents a system of five different MCUs in which there are one master and four slave (0, 1, 2, 3). In this system four basic line (signals) are required for the MOSI (master out slave in), MISO (master in slave out), SCK serial clock, and $\overline{SS}$ (slave select) lines.

Features

- Full Duplex, Three-wire Synchronous Transfers
- Master or Slave Operation
- Master Bit Frequency - 2.5MHz Maximum
- Slave Bit Frequency - 5.0MHz Maximum
- 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, $\overline{SS}$) discussed above are described in the following paragraphs. Each signal function is described for both the master and slave mode.

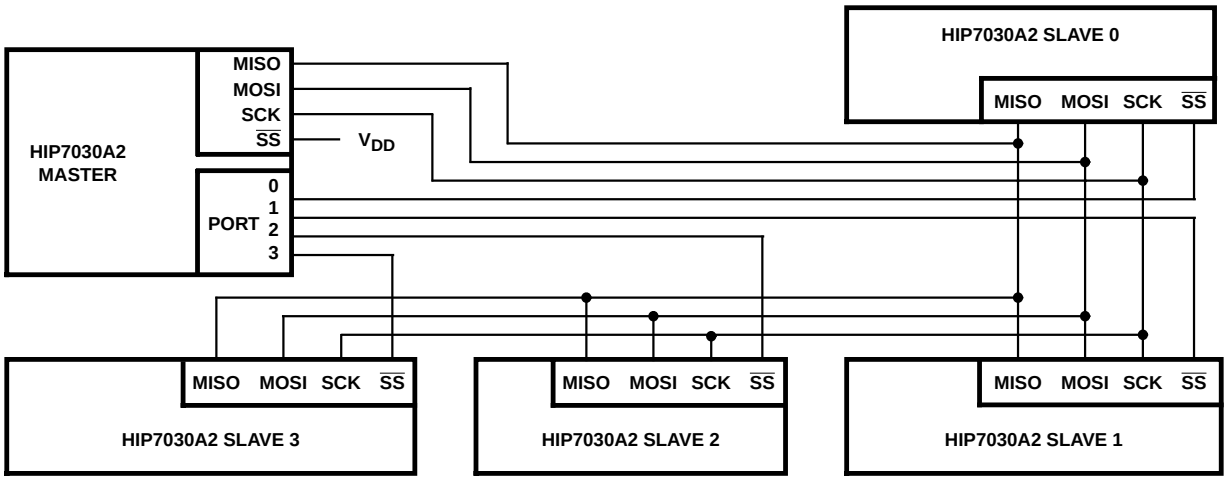


FIGURE 20. MASTER-SLAVE SYSTEM CONFIGURATION (SINGLE MASTER, FOUR SLAVES)

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 Figure 21 summarize the SPI timing and show the relationship between data and clock (SCK). As shown in Figure 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, location $0A). When a device is operating as a master, the MOSI pin is an output because the program in firmware sets the MSTR bit to a logic one.

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 SS pin is a logic one. The timing diagram of Figure 21 shows the relationship between data and clock (SCK). As shown in Figure 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 enable by the logic level of the SS pin; i.e., if SS = 1 then the MISO pin is placed in the high-impedance state, whereas, if SS = 0 the MISO pin is an output for the slave device.

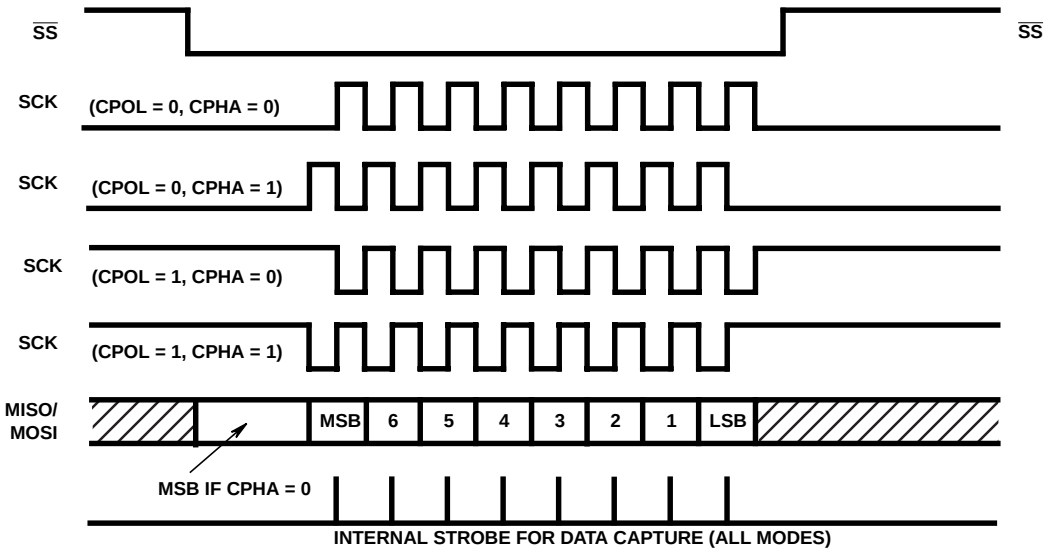


FIGURE 21. DATA CLOCK TIMING DIAGRAM

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. The SCK is generated by the master device, is an input on all slave devices, and synchronizes master/slave data transfers. The type of clock and its relationship to data are controlled by the CPOL and CPHA bits in the Serial Peripheral Control Register (SPCR, location $0A) discussed below. Refer to Figure 21 for timing.

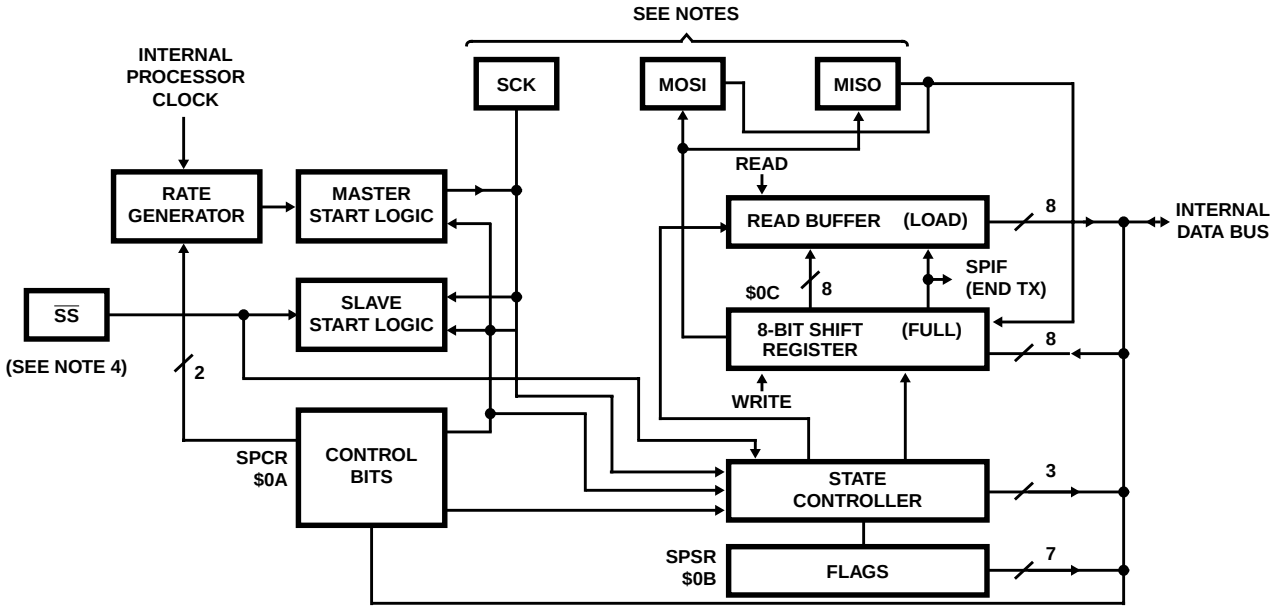
The master device generates the SCK through a circuit driven by the internal processor clock. Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. The master device uses the SCK to latch incoming slave device data on the MISO line and shifts out data to the slave device on the MOSI line. Both master and slave devices must be operated in the same timing mode as controlled by the CPOL and CPHA bits in the SPCR. In slave devices, SPR0, SPR1 have no effect on the operation of SPI. Timing is shown in Figure 21.

Slave Select ($\overline{SS}$)

The slave select ($\overline{SS}$) pin is a fixed input, which receives an active low signal to enable slave device(s) to transfer data. A high level $\overline{SS}$ signal forces the MISO line to the high-impedance state. Also, SCK and MOSI are ignored by a slave device when its $\overline{SS}$ signal is high. The $\overline{SS}$ signal must be driven low prior to the first SCK and must remain low throughout a transfer. The $\overline{SS}$ input on a Master must be held high at all times (see description of MODF under **Serial Peripheral Status Register** for more details).

As shown in Figure 21, with CPHA = 0, the first bit of data must be applied to the MISO line prior to the first transition of the SCK. In this case, $\overline{SS}$ going low is used to provide the first clock edge of a transfer. A device is prevented from writing to its SPI data register while $\overline{SS}$ is low and CPHA = 0 (see description of WCOL under **Serial Peripheral Status Register** for more details). **These facts require that $\overline{SS}$ go high between SPI data transfers whenever CPHA = 0**.

When CPHA = 1, the $\overline{SS}$ of a slave can be held low throughout a series of SPI transfers and in a single slave system can even be permanently wired low.



NOTES: The $\overline{SS}$, SCK, MOSI, and MISO are external pins which provide the following functions:

1. 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.
2. MISO - Provides serial input from slave unit(s) when device is configured as a master. Receives serial output to master unit when device is configured as a slave unit.
3. SCK - Provides system clock when device is configured as a master. Receives system clock when device is configured as a slave unit.
4. $\overline{SS}$ - Provides a logic low to select device for a transfer with the master.

FIGURE 22. SERIAL PERIPHERAL INTERFACE (SPI) BLOCK DIAGRAM

When a device is a master, it constantly monitors its $\overline{SS}$ signal input for a logic low. The master device will become a slave device any time its $\overline{SS}$ signal input 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. 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.

Functional Description

A block diagram of the serial peripheral interface (SPI) is shown in Figure 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, 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.

Figure 23 illustrates the MOSI, MISO, and SCK master-slave interconnections. Note that in Figure 23 the master $\overline{SS}$ pin is tied to a logic high and the slave $\overline{SS}$ pin is a logic low. Figure 20 provides a larger system connection for these same pins. Note that in Figure 20, 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.

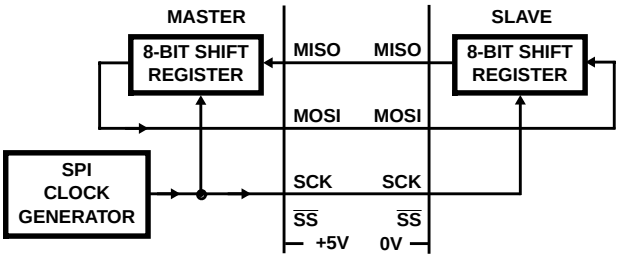


FIGURE 23. SERIAL PERIPHERAL INTERFACE (SPI) MASTER-SLAVE INTERCONNECTION

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.

Serial Peripheral Control Register (SPCR)

The serial peripheral control register bits are defined as follows:

7	6	5	4	3	2	1	0
SPIE	SPE	-	MSTR	CPOL	CPHA	SPR1	SPR0

SPCR (LOCATION $0A)

B7, SPIE When the serial peripheral interrupt enable 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 MODE) 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. When SPE is low, all pins appear as inputs to the external system. Because the SPE bit is cleared by RESET, the SPI system is not connected to the external pins upon 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 Figure 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 Figure 21.

B1, SPR1; B0, SPR0

These two serial peripheral rate bits select one of four baud rates to 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. 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)

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

7	6	5	4	3	2	1	0
SPIF	WCOL	0	MODF	0	0	0	0

SPSR (LOCATION $0B)

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 serial peripheral status register is read. 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 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 $\overline{SS}$ 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 the 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 on 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 becomes 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.

Since 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.

B4, MODF The function of the mode fault flag 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, disables 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 bit SPE and MSTR may be restored to their original set state during this cleared 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							

SPDR (LOCATION $0C)

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 bit 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 a single master system and a discussion of both is provided below.

Figure 20 illustrates how a typical single master system may be configured, using a CDP68HC05 family device as the master and four CDP68HC05 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 device all receive it. Since the CDP68HC05 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 since the master device ports will be forced to be inputs at that time, thus, disabling the slave devices. Note 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 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. An exchange of master control could be implemented using a handshake method through the I/O ports or 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.

J1850 VPW Messaging

This section provides an introduction to J1850 multiplexed communications. It is assumed that the user is or will become familiar with the appropriate documents published by the Society of Automotive Engineering (SAE). The following discussion is not comprehensive.

Overview

The SAE J1850 Standard (Note 1) (J1850) establishes the requirements for communications on a Class B multiplexed wiring network for automotive applications. The J1850 document details the requirements in a three layer description which separately specifies the characteristics of the *physical layer*, the *data link layer*, and the *application layer*. There are several options within each layer which allows vehicle manufacturers to customize the network while still maintaining a level of universality.

NOTE:

1. **SAE J1850 Standard, Class B Data Communication Network Interface**, May 1994, Society of Automotive Engineers Inc.

The hardware of the Intersil HIP7030A2 provides features which facilitate implementation of the 10.4Kbps Variable Pulse Width Modulated (VPW) physical layer option of J1850. In combination with a bus transceiver, such as the Intersil J1850 Bus Transceiver HIP7020, and appropriate software algorithms the HIP7030A2 circuitry enables the designer to completely implement a 10.4Kbps VPW Class B Communications Network Interface per J1850. Features of such an implementation include:

- Single Wire 10.4Kbps Communications
- Message Buffering and Message Filtering
- Bit-by-Bit Bus Arbitration
- Industry Standard Protocol
- Message Acknowledgment (“In-Frame Response”) Capabilities
- Exceptionally Tolerant of Clock Skew, System Noise, and Ground Offsets
- Meets CARB and EPA Diagnostic Requirements
- Supports up to 32 Nodes
- Low Error Rates
- Excellent EMC Levels (when interfaced via Intersil J1850 Bus Transceiver HIP7020)

In addition to the standard J1850 features, the HIP7030A2 hardware provides a high speed mode, (intended for receive only use) which can significantly enhance vehicle maintenance capabilities. The high speed mode provides a 41.6Kbps communications path to any node built with the HIP7030A2.

Anatomy of a J1850 VPW Message

All messages in a J1850 VPW system are sent along a single wire, shared bus. At any given moment the bus can be in either of two states: *active* (high) or *passive* (low). Multiple nodes are connected to the bus as a “wired-OR” network in which the bus is high if *any* one (or more) node is generating an active output. The bus is only low when *no* nodes are generating active outputs. It follows that, when no communications are taking place the bus will rest in the passive state. A message begins when the bus is first driven to the high state. Each succeeding state transition (i.e., a change from active to passive or passive to active) transfers one bit of information (*symbol*) until the message is complete and the bus once again rests at the passive state. The interpretation of each symbol in the message is dependent on its duration (and state), hence, the descriptor Variable Pulse Width (VPW).

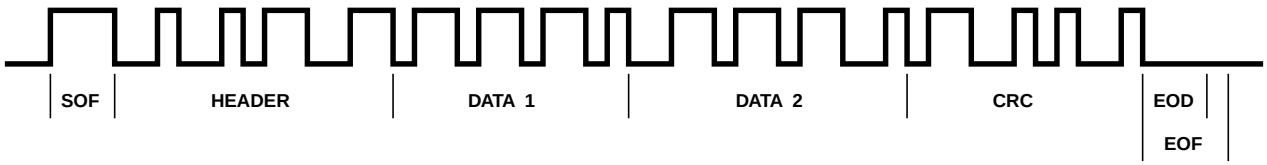


FIGURE 24. TYPICAL J1850 VPW MESSAGE FRAME

Each message has a beginning and an end, the span of which encompasses the entire *message* or *frame* (refer to Figure 24). A frame consists of an active *start of frame* (SOF) symbol and a passive *end of frame* (EOF) symbol sandwiched around a series of byte sized (8-bit) groups of symbols. The first byte of the frame contents is always a *header* byte, followed by possibly additional header bytes, followed by one or more *data* bytes, followed by an integrity check byte (*CRC* byte), followed by a passive *end of data* (EOD) symbol, followed by possibly one or more *in-frame-response* (IFR) bytes. To keep waiting times low, messages are limited to 12 bytes total (including header, data, check, and IFR bytes). All message bytes are transmitted most significant bit (MSB) first.

VPW Symbol Definitions

Within the J1850 scheme, symbols are defined in terms of both duration and state (passive or active). The duration is measured as the time between successive transitions. There is one transition per symbol and one symbol per transition. The end of one symbol marks the beginning of the next. Since the bus is passive when a message begins and must return to that same state when the message completes, all frames have an even number of transitions and hence an even number of symbols.

There are unique definitions for data bit symbols (all the symbols which occur within the header, data, and check bytes) and protocol symbols (including SOF, EOD, and EOF). The duration of each symbol is expressed in terms of VPW Timing Pulses (TV values). Table 5 summarizes the TV definitions. Each TV is specified in terms of a *nominal* (or ideal) duration and a *minimum* and *maximum* duration. The span between the minimum and maximum limits accommodates system noise sources such as node to node clock skew, ground offsets, clock jitter, and electromechanical noise. There are no dead zones between the maximum of one TV and the minimum of the next.

TABLE 5. J1850 TV DEFINITIONS

TV ID	DURATION (ALL TIMES IN MICROSECONDS)		
	MINIMUM	NOMINAL	MAXIMUM
Illegal	0	NA	≤34
TV1	>34	64	≤96
TV2	>96	128	≤163
TV3	>163	200	≤239
TV4	>239	280	NA
TV5	>239	300	NA
TV6	>280	300	NA

The terms *short* and *long* are often used to refer to pulses of duration TV1 and TV2 respectively.

VPW is a non-return-to-zero (NRZ) protocol in which each transition represents a complete bit of information. Accordingly, a *0* data bit will sometimes be transmitted as a passive pulse and sometimes as an active pulse. Similarly, a *1* data bit will sometimes be transmitted as a passive pulse and sometimes as an active pulse. In order to accommodate arbitration (see **Bus Arbitration**) a *long active* pulse represents a 0 data bit and a *short active* pulse represents a 1 data bit. Complementing this fact, a *short passive* pulse represents a 0 and a *long passive* pulse represents a 1. Starting from a transition to the active state, a 0 data bit will maintain the active level longer than a 1. Similarly, starting from a transition to the passive state, a 0 data bit will return to the active level quicker than a 1. These facts give rise to the dominance of 0's over 1's on the J1850 bus as depicted in Figure 25. See **Bus Arbitration** for additional details.

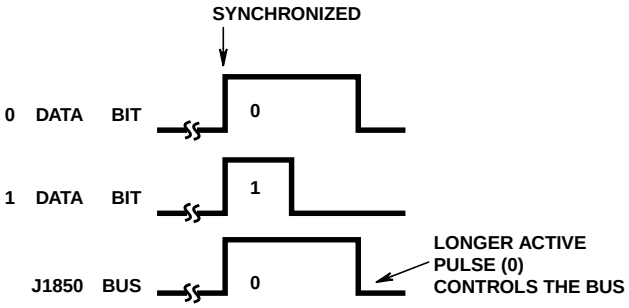


FIGURE 25A. DOMINANCE OF ACTIVE 0 DATA BIT

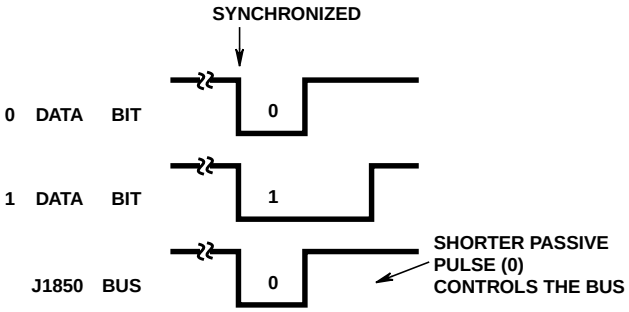


FIGURE 25B. DOMINANCE OF PASSIVE 0 DATA BIT

Table 6 summarizes the complete set of symbol definitions based on duration and state.

TABLE 6. J1850 SYMBOL DEFINITIONS

SYMBOL	DEFINITION
0 Data	Passive TV1 or Active TV2
1 Data	Active TV1 or Passive TV2
SOF (Start of Frame)	Active TV3
EOD (End of Data)	Passive TV3
EOF (End of Frame)	Passive TV4
IFS (Inter-Frame Separation)	Passive TV6
IDLE (Idle Bus)	Passive >TV6 nom
NB (Normalization Bit)	Active TV1 or Active TV2
BRK (Break)	Active TV5

In Frame Response (IFR)

The distinction between two of the passive symbols, EOD and EOF, is subtle but important (refer to Figure 26). The EOD (TV3) interval signifies that the originator of the message is done broadcasting and any nodes which have been requested to respond (i.e., to acknowledge receipt of the message) can now do so. The EOD interval begins when the transmitting node has completed sending the eighth bit of the check byte. The transmitter simply releases the bus and allows it to revert to a passive state. In the course of normal messaging, no node can seize the bus until an EOD time has been detected. Once an EOD has elapsed, any nodes which are scheduled to produce an IFR will arbitrate for control of the bus (see **Bus Arbitration**) and respond appropriately. If no responses are forthcoming the bus remains in the passive state until an EOF (TV4) interval has elapsed. After the EOF has been generated, the frame is considered closed and the next communications on the bus will represent a totally new message.

IFRs can consist of multiple bytes from a single respondent, one byte from a single respondent, or one byte from multiple respondents. In all cases the first response byte must be preceded by a *normalization bit (NB)* which serves as a *start of response* symbol and places the bus in an active state so that following the IFR byte(s) the bus will be left in the passive state.

The NB symbol is by definition active, but can be either TV1 or TV2 in duration. The long variety (TV2) signifies the IFR contains a CRC byte. The short variety (TV1) precedes an IFR without CRC.

Message Types

Messages are classified into one of four *Types* according to whether the message has an IFR and what kind of IFR it is. The definitions are:

- Type 0 - No IFR
- Type 1 - One byte IFR from a single respondent (no CRC byte)
- Type 2 - One byte IFRs from multiple respondents (no CRC byte)
- Type 3 - Multiple byte IFR from a single respondent (CRC appended)

Bus Arbitration

The nature of multiplexed communications leads to contention issues when two or more nodes attempt to transmit on the bus simultaneously. Within J1850 VPW systems, messages are assigned varying levels of priority which allows implementation of an arbitration scheme to resolve potential contentions. The specified arbitration is performed on a symbol by symbol basis throughout the duration of every message.

Arbitration begins with the rising edge of the SOF pulse. No node should attempt to issue an SOF until an Idle bus has been detected (i.e., an *Inter-Frame Separation (IFS)* symbol with a period of TV6 has been received). If multiple nodes are ready to access the bus and are all waiting for an IFS to elapse, invariable skews in timing components will cause one arbitrary node to detect the Idle condition before all others and start transmission first. For this reason, all nodes waiting for an IFS will consider an IFS to have occurred if either:

- An IFS nominal period has elapsed
  or,
- An EOF minimum period has elapsed and a rising edge has been detected

Arbitrating devices will all be synchronized during the SOF. Beginning with the first data bit and continuing to the EOF, every transmitting device is responsible for verifying that the symbol it sent was the symbol which appeared on the bus. Each transition, every transmitting node must decode the symbol, verify the received symbol matches the one sent, and begin timing of the next symbol. Since timing of the next symbol begins with the last transition detected on the bus, all transmitters are re-synchronized each symbol. When the received symbol doesn't match the symbol sent, a conflict (*bit collision*) occurs. Any device detecting a collision will assume it has lost arbitration and immediately relinquish the bus. Typically, after losing arbitration, a device will attempt re-transmission of the message when the bus once again becomes Idle.

The definition of 1 and 0 data bits (see Table 6 and discussion under **VPW Symbol Definitions**) leads to 0's having priority over 1's in this arbitration scheme. Header bytes are generally assigned such that arbitration is completed before the first data byte is transmitted. Because of the dominance of 0-bits and the MSB first bit order, a header with the hexadecimal value $00 will have highest priority, then $01, $02, $03, etc. An example of two nodes arbitrating for control of the bus is shown in Figure 27.

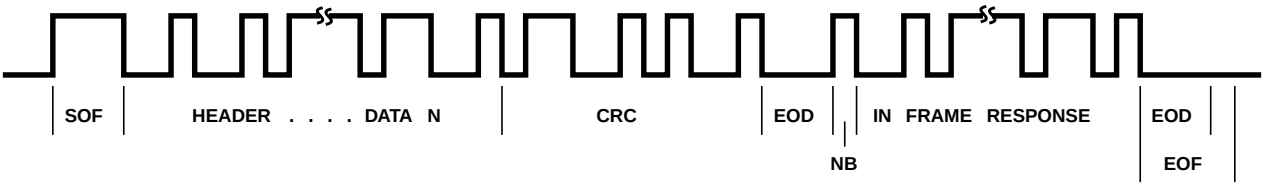


FIGURE 26. J1850 MESSAGE WITH IN-FRAME-RESPONSE

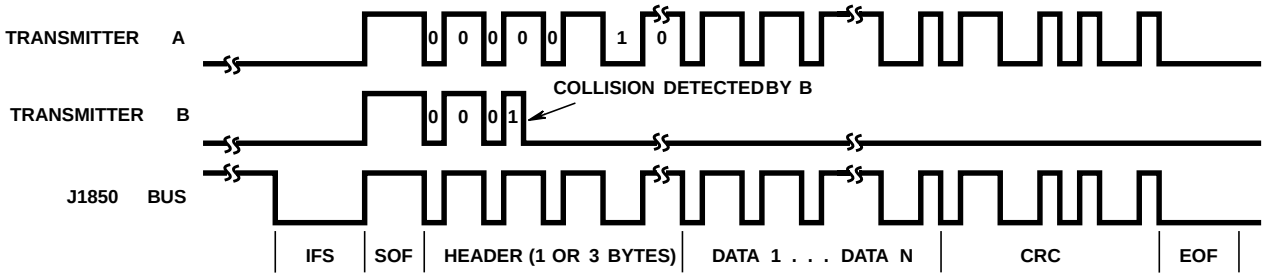


FIGURE 27. TWO NODES ARBITRATING FOR CONTROL OF J1850 BUS

Arbitration also takes place during the IFR portion of a message, if more than one node is attempting to generate a response. Arbitration begins with the NB symbol, which follows the EOD and precedes the first IFR byte.

For Type 1 and Type 3 messages only the respondent which successfully arbitrates for control of the bus produces an IFR. All other respondents abort their IFRs.

For Type 2 messages, all respondents which lose arbitration must count symbols and re-attempt transmission at the end of each byte. Each node, which successfully responds, eliminates itself from the subsequent arbitration until all nodes have responded. This arbitration scheme limits each respondent to a single byte during a Type 2 IFR.

Break

To force a message to be aborted before EOF is reached, a break (BRK) symbol can be transmitted by any node. The BRK symbol is an active pulse of duration TV5. Reception of a break causes all nodes to reset to a *ready-to-receive* state and to re-arbitrate for control following an IFS.

Variable Pulse Width Symbol Encoder Decoder (SENDEC)

Overview Of SENDEC Operation

The SENDEC hardware integrated in the HIP7030A2 facilitates generation and reception of J1850 messages on a symbol by symbol basis. Symbols are output from the SENDEC, as a digital signal, on the VPWOUT pin and input, as a digital signal, on the $\overline{\text{VPWIN}}$ pin. These two lines must be connected through a bus transceiver (such as the Intersil J1850 Bus Transceiver HIP7020) to the single wire J1850 bus. The transceiver is responsible for generating and receiving waveforms consistent with the physical layer specifications of J1850. In addition, the transceiver is responsible for providing isolation from bus transients.

Every symbol sent out on the VPWOUT is in effect inverted and echoed back on the $\overline{\text{VPWIN}}$ pin after some finite delay through the transceiver. In actuality, only long active symbols are guaranteed to be echoed unchanged. If the transmitted symbol is passive and another node is simultaneously sending an active symbol, the active symbol will dominate and pull the bus to a high level.

The SENDEC circuitry includes a 3-bit digital filter which effectively filters out noise pulses less than 7µs in duration.

Communications between the CPU and the SENDEC are via three registers mapped into Page 0 of the MCUs memory space.

When transmitting symbols, the desired symbol is specified by writing an appropriate code to the SENDEC Data Register (SEDDR). Timing of each symbol is calculated from the last transition on the $\overline{\text{VPWIN}}$ line. Each write to the SEDDR, which occurs within 34µs of the last received transition, will enable the VPWOUT pin and the SENDEC automatically produces a transition on the VPWOUT pin after the proper delay (the seven microseconds added by the digital filter and a 17µs delay through the bus transceiver are compensated for). The VPWOUT pin remains active until 34µs after the last received transition. Failure to write a new symbol during the 34µs window causes the VPWOUT pin to go low until the next valid write or until the Force Start of Frame (FSOF) bit is set in the SENDEC Data Register (SEDDR).

Decoding of received symbols is automatically performed by the SENDEC. The decoded symbol is valid until the next transition occurs. The value can be read via the SEDDR.

Generally the SENDEC is programmed to interrupt the MCU with each transition on the $\overline{\text{VPWIN}}$ pin. When the SENDEC is receiving a message, the interrupt signals that a new symbol has been received and appropriate actions must be taken to read and process the symbol. When the SENDEC is transmitting, the transition interrupt signifies that the reflected symbol has been received, and it is time to start

timing the next symbol. The reflected symbol should be read and compared to the previously sent one. If the reflected symbol doesn't match the symbol sent, a collision has occurred and the software must cease transmissions until the next idle period. If there was no collision, the new symbol must be immediately (within one TV1 minimum time) written to the SEDDR (the SEDDR is not buffered).

In addition to features already discussed, the SENDEC includes, noise detection, idle bus detection, a clock prescaler, an echo fail detector, a wake-up facility, and a high speed receive mode. Symbol timing is based on the main MCU oscillator. The programmable prescaler allows proper SENDEC operation with 10MHz, 8MHz, or 4MHz oscillators. The high speed receive mode is a J1850 extension which allows maintenance equipment to transmit messages at 4X the normal 10.4Kbps rate.

Software algorithms can be employed to implement message buffering and filtering, CRC generation and detection, IFR handling, and other needed features to create a complete J1850 VPW node. See the *Applications* section for typical algorithms.

SENDEC Registers

The SENDEC register set consists of the read-write SENDEC Data Register (SEDDR), the read-write SENDEC Control Register (SEDCR), and the read-only SENDEC Status Register (SEDSR). A detailed description of the operation of each follows:

SENDEC Control Register (SEDCR)

The SENDEC Control Register (SEDCR, location $0F) is an 8-bit read/write register which contains five control bits. One of the bits controls interrupts which are associated with a flag bit in the SENDEC Status Register (discussed following). Three bits control the clock prescaler and the high speed 4X mode of the SENDEC. The final bit doesn't directly control the SENDEC, rather it controls the start-up delay following exit from the STOP mode of the processor. The bit assignments are illustrated below, followed by a detailed description of each bit.

7	6	5	4	3	2	1	0
TXIE	-	-	NDEL	-	4X	PRE1	PRE0

SEDCR (LOCATION $0F)

B7, TXIE If the transition interrupt enable (TXIE) bit is set, the MCU will receive a SENDEC interrupt on the occurrence of each transition on the $\overline{\text{VPWIN}}$ line.

If TXIE is low the TX interrupts are inhibited but the associated flags in the SENDEC Status Register (SEDSR) are still set (see discussion following).

TXIE is cleared by RESET.

B4, NDEL When set, the No Delay (NDEL) bit suppresses the 4064 t_{CYC} delay which is normally introduced when exiting from the STOP mode via an interrupt. Instead of the 4064 t_{CYC} delay a 96 t_{CYC} delay is introduced. NDEL is intended for applications where the clock source to the HIP7030A2 continues to run when the device enters STOP mode or when a ceramic resonator based oscillator is used. NDEL should not be used when the HIP7030A2 is driven by a quartz crystal based oscillator.

NDEL is cleared by RESET and POR.

B2, 4X When set, the 4X bit causes the SENDEC symbol timing to be accelerated by a factor of four. Due to fixed delays in the loop back from VPWOUT to $\overline{\text{VPWIN}}$, the 4X mode is only useful for receiving symbols. 4X mode is intended for high speed data linking between the HIP7030A2 and maintenance or test equipment which has capability to send at the accelerated rate.

Writing to the 4X bit is inhibited except when the NEW bit in the SENDEC Status Register (SEDSR) is set.

Once modified, the new value of 4X doesn't take effect until the next transition on the $\overline{\text{VPWIN}}$ pin.

Receipt of a Break symbol on the $\overline{\text{VPWIN}}$ line will automatically clear 4X.

RESET and POR clear the 4X bit.

B1, PRE1; B2, PRE0

PRE1 and PRE0 control the SENDEC clock prescaler. The SENDEC circuit requires a fundamental clock of 1MHz. To generate the 1MHz frequency, while allowing a choice of MCU oscillator frequencies, the PRE1 and PRE0 bits must be set to match the OSCIN frequency.

TABLE 7. SENDEC PRESCALER BIT SELECTION

PRE1	PRE0	OSCIN FREQUENCY (MHz)
0	0	4
0	1	8
1	0	10
1	1	12

Following RESET a window of 4 instructions is allowed for setting the PRE1 and PRE0 bits. Writes to these bits after the fourth instruction have no effect on their values.

Table 7 gives the proper settings of PRE1 and PRE0 for various frequencies.

RESET and POR force PRE1 to a 1 and PRE0 to a 0, selecting the 10MHz mode.

SENDEC Status Register (SEDSR)

The SENDEC Status Register (SEDSR, location $10) is an 8-bit read-only register which contains seven status bits. One of the bits is a flag bit which correspond to the interrupt control bit in the SENDEC Control Register (discussed earlier). Three other bits provide error status information.

Another two bits provide an indication of special symbols (Break, IFS) occurring on the bus. The final bit indicates the transmit status of the SENDEC. The bit assignments are illustrated below, followed by a detailed description of each bit.

7	6	5	4	3	2	1	0
TX	BRK	NEW	NOIZ	OVR	TALK	NECHO	-

SEDSR (LOCATION $10)

B7, TX The transition (TX) flag bit indicates that a transition has occurred on the VPWIN line. The line is first filtered through the SENDECs 3-bit digital filter to reject noise.

Once set the TX flag will interrupt the MCU if the TXIE bit in the SEDCR is set and the I bit in the condition code register is clear. TX is cleared by a sequence of first reading the SEDSR followed by reading or writing the SENDEC Data Register (SEDDR).

Note that both TX and NEW will be set on the leading edge of an SOF. See description of the NEW flag below.

TX is cleared by RESET.

B6, BRK The break (BRK) bit indicates that a break symbol has been detected on the $\overline{\text{VPWIN}}$ line. BRK is set at the end of the break symbol, on the active to passive transition.

Once set the BRK flag will interrupt the MCU if the I-bit is cleared in the condition code register. BRK is cleared by a sequence of first reading the SEDSR followed by a read or write of the SEDCR.

BRK is cleared by RESET.

B5, NEW The new frame (NEW) flag indicates that one of two possible events has been detected on the J1850 bus:

The bus has been passive for at least an IFS nominal symbol time (i.e., the bus is Idle)

or, A transition has occurred on the bus following an EOF minimum (i.e., another node has started a new message).

NEW is set when either of these events is detected. In the case of a transition following an EOF, the TX bit is also set.

When NEW goes from a 0 to a 1, a SENDEC interrupt will be generated if the I-bit is cleared in the CC register. The NEW interrupt can be cleared under software control by reading the SEDSR followed by reading or writing the SEDCR. This only removes the source of the interrupt and does not clear the NEW bit.

The NEW flag cannot be cleared by software. It is automatically cleared 128 (nominal) microseconds into the next (or current - if NEW was set by a transition following EOF) symbol. This is normally during the SOF of a new message. If the symbol is less than 128µs in duration (an illegal SOF symbol), the NEW flag is cleared on the active to passive transition.

Polling NEW provides a convenient means for software to determine that transmission of a new message can be commenced.

NEW is cleared by all resets.

B4, NOIZ The noise (NOIZ) flag indicates that a symbol shorter than a legal TV1 has been received. NOIZ is cleared by a sequence of first reading the SEDSR followed by reading or writing the SEDCR.

NOIZ is cleared by RESET.

B3, OVR The overrun (OVR) flag is set if TALK is set in the SEDSR and a minimum short symbol time (34µs) has elapsed since the last transition and no write to the SEDDR has taken place. An overrun condition is a serious error and the user should treat it as such. When OVR is set it automatically forces the VPWOUT pin to a low level. OVR is cleared by a sequence of first reading the SEDSR followed by reading or writing the SEDCR.

Setting of OVR is inhibited while NEW is true in the SEDSR.

OVR is cleared by RESET.

B2, TALK The transmit (TALK) flag is set if the HIP7030A2 is actively transmitting symbols via the SENDEC. TALK is set by writing a non-zero to the SEDDR (see *SENDEC Data Register* for details).

The TALK bit is cleared by writing a $00 to the SEDDR, when NECHO is set, or when OVR is set.

TALK is cleared by RESET.

B1, NECHO

The No Echo Received (NECHO) flag is set if, during the process of transmitting a symbol, the expected echo of the symbol is not received. This event will cause the VPWOUT pin to be forced to a 0 level. Setting of NECHO automatically clears the TALK bit.

The time required to detect an echo failure is dependent on many factors. The minimum time to detect a failure is 105µs (26µs in 4X mode) and the maximum time to detect a failure is 512µs.

When NECHO goes from a low to a high level, a SENDEC interrupt will be generated if the I-bit is cleared in the CC register. NECHO must go low then high again to generate another interrupt.

NECHO is cleared by a sequence of first reading the SEDSR followed by reading or writing the SENDEC Data Register (SEDDR).

NECHO is cleared by all resets.

SENDEC Data Register

The SENDEC Data Register (SEDDR, location $11) is an 8-bit read/write register which contains one write-only bit, three read/write bits, and four read-only bits. The write only bit triggers SOF symbols required to initiate new transmissions, the three read/write bits are used to specify transmitted symbol durations, and the four read only bits uniquely identify the received J1850 symbol. Reading the SEDDR at anytime provides the received symbol which resulted from the last transition of VPWIn. When writing data to the SEDDR, the value represents the duration of the symbol currently being transmitted. The bit assignments are illustrated below, followed by a detailed description.

7	6	5	4	3	2	1	0
FSOF	S2	S1	S0	LEV	R2	R1	R0

SEDDR (LOCATION $11)

B7, FSOF Writing a 1 to the Force Start of Frame (FSOF) bit while simultaneously writing a non-zero value to S2-0, causes the VPWOUT to immediately go active (high level). The low to high transition will eventually be reflected on the $\overline{\text{VPWIN}}$ line causing a TX interrupt. Upon receipt of the TX interrupt an SOF symbol (S2-0 = 3) must be written to the SEDDR to time the high SOF.

Setting the FSOF bit can only be done when the NEW flag is set in the SENDEC Status Register (NEW is set when the J1850 bus is idle or during the first portion of an SOF symbol).

FSOF is a write only bit. Reading FSOF always returns a 0.

B6, S2; B5, S1; B4, S0

When writing to the SEDDR, the three bits (S2-0) determine the transmitted symbol as shown in Table 8. During a write to the SEDDR the S2-0-bits are ignored except in three specific situations:

The NEW flag is high in the SEDSR)

or, A transition has been received on the $\overline{\text{VPWIN}}$ pin, from the bus, within the past 34µs

or,

S2-0-bits = 0

In the first two cases, each write to the SEDDR will produce one properly timed symbol on the VPWOUT pin. The completion of the symbol is reported to the controller, *not* at the end of the transmitted symbol, but at the end of the symbol echoed back via the $\overline{\text{VPWIN}}$ input. Writing the FSOF bit, in conjunction with S2-0 = 3, produces the initial transition for the SOF symbol. All timing for a message begins with the receipt of that transition.

Writing a $00, at anytime, immediately disables transmissions (forcing the VPWOUT pin low) and clears the TALK bit in the SEDSR. This is the preferred method to end transmissions.

RESET doesn't affect S2-S0

TABLE 8. S2-S1 SYMBOL ENCODING

S2	S1	S0	TRANSMIT SYMBOL
0	0	0	Disable Transmit
0	0	1	TV1
0	1	0	TV2
0	1	1	TV3
1	0	0	TV1
1	0	1	TV1
1	1	0	TV1
1	1	1	TV1

B3, LEV; B2, R2; B1, R1; B0, R0

These four bits uniquely identify all symbols received via the $\overline{\text{VPWIN}}$ pin. The symbol decoding map is shown in Table 9. R2-0 represent the duration of the symbol and LEV represents the level of the symbol (active or passive)

These bits are only updated upon detection of a bus transition and therefore reflect the last symbol received. An exception to this is for an Idle bus. When an Idle has been detected the values in R2-R0 and LEV are immediately updated - no bus transition is necessary.

R2-R0 = 101 with LEV = 0 indicates that the bus is currently Idle.

Note that R2-0 combinations of 110 and 111 will not be produced by the SENDEC. A value of 101 represents all durations equal to and beyond an IFS/IDLE (for the passive case) and a BREAK (for the active case).

RESET does not affect LEV or R2-R0.

When a transition is detected on $\overline{\text{VPWIN}}$, the received symbol is decoded and made available for reading via the SEDDR. The TX bit is set in the SEDSR and, if TXIE is high in the SEDCR, an interrupt will be generated. Once the transition is detected the next symbol begins timing out. A new symbol must be written to the SEDDR, before the minimum transmit time for a short symbol has elapsed (34µs). Failure to write to the SEDDR in time will result in the OVR bit being set and transmission aborted. This is a safety precaution to prevent "streaming" messages.

The control routines should verify that the symbol sent matches the symbol received. A mismatch indicates the device has lost control of the bus. It is up to the user code to handle the collision, in terms of disabling the SENDEC, requeueing of the message, filtering the incoming message, etc.

TABLE 9. R2-R0 AND LEV SYMBOL DECODING

R2	R1	R0	LEV	RECEIVE SYMBOL
0	0	0	0	Passive Noise
0	0	0	1	Active Noise
0	0	1	0	TV1 Passive
0	0	1	1	TV1 Active
0	1	0	0	TV2 Passive
0	1	0	1	TV2 Active
0	1	1	0	EOD
0	1	1	1	SOF
1	0	0	0	EOF
1	0	0	1	BREAK
1	0	1	0	IFS/IDLE
1	0	1	1	BREAK
1	1	0	0	-
1	1	0	1	-
1	1	1	0	-
1	1	1	1	

In the receive mode (i.e., no writes to the SEDDR) the controller typically responds to the TX interrupts and reads the incoming symbols as they become available, performing necessary real-time operations such as filtering messages, computing and verifying CRCs, and issuing IFRs.

Computer Operating Properly (COP) System

Introduction

The Computer Operating Properly (COP) system is comprised of two basic circuit components. One is a free running watchdog timer which, left unattended, generates a periodic MCU reset. The second is a Slow Clock Detect circuit which constantly monitors the OSCIN line for activity. A lack of activity on OSCIN will generate a reset.

Both circuits are capable of generating a COP interrupt which forces an MCU reset and restarts operation at the vector specified by the contents of location $1FFA, $1FFB. Because the COP interrupt behaves as a reset, the stack pointer is cleared and exiting the COP interrupt software handler must be done via a jump instruction as opposed to an RTI or RTS.

The Watchdog Status Register (WSR, location $1E) contains a flag (Watchdog Flag - WDF, bit 0) which is set whenever a Watchdog Timer overflow interrupt occurs. The WDF bit is cleared by a POR or a write to the Watchdog Reset Register (WRR, location $1D) with bit 0 = 0. The WDF can be used to distinguish the type of COP reset (Watchdog timeout vs. Slow Clock Detect) which has occurred.

Following are the details of each of the two circuit.

Slow Clock Detect Circuit

The Slow Clock Detect Circuit consists of a reset-able timer element. The timer is constructed with integrated resistive and capacitive components. Each positive transition on the OSCIN line reinitializes the timer. In the absence of frequent enough transitions on the input, the timer will eventually reach a preset limit at which point the MCU will be reset via a COP interrupt.

When the frequency has dropped below the preset threshold a COP reset will take place. A COP reset is identical to a POR or $\overline{\text{RESET}}$ pin reset, except the restart vector is the COP Vector. Following the COP reset the HIP7030A2 is held reset until the start-up timeout of 4064 clocks has been reached. During the 4064 clocks the Slow Clock Detect circuit is inhibited. If at the end of the 4064 clocks the frequency remains below the threshold, a COP reset will immediately take place again.

The primary purpose of this circuit is to force the HIP7030A2 off of the J1850 and SPI busses should the oscillator circuit fail. Due to variability of integrated resistors and capacitors there is a non-critical spread in the timeout specification of approximately 10:1. Maximum threshold is 200kHz. Refer to f_{SLOW} in *Electrical Specifications* for details.

There is no means to disable the Slow Clock Detect. RESET resets the Slow Clock Detect circuit and holds it reset until the start-up timeout of 4064 clocks has been reached and the $\overline{\text{RESET}}$ pin has gone high.

Watchdog Timer

The Watchdog Timer is a free-running 21 stage counter which divides the OSCIN input by 2,097,152. The Timer is software reset-able, and must be constantly reset before the terminal count is reached. Failure to reset the Watchdog Timer, in due time, results in a forced MCU RESET via a COP interrupt.

7	6	5	4	3	2	1	0
Watchdog Reset Register							

WRR (LOCATION $1D)

Resetting the Watchdog Timer requires two distinct operations. A write of the value $55 to the Watchdog Reset Register (WRR, location $1D) must be followed by a write of the value $AA to the WRR. There is no limit on the time between the writes, other than both must take place before the Watchdog Timer has reached its limit. Typically the two writes are placed in distinct sections of code, which can only be reached by proper flow through the software.

Each time that a write is made to the WRR with bit 0 = 0, the Watchdog Flag in the WSR is cleared. This will happen as a normal side effect of clear the Watchdog Timer via the $55, $AA sequence. The Watchdog flag is also cleared by POR. RESET and Slow Clock Detect do not affect the WDF. It is set by a Watchdog Timer overflow and can be used to distinguish a Slow Clock Detect reset from the Watchdog reset, both of which share the COP reset vector ($1FFA,$1FFB).

7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	WDF

WSR (LOCATION $1E)

Watchdog timeout periods for various OSCIN frequencies are given in Table 10.

There is no mechanism to disable the Watchdog Timer. RESET clears the Watchdog Timer to its initial value.

TABLE 10. WATCHDOG TIMEOUTS FOR COMMON OSCIN FREQUENCIES

WATCHDOG TIMEOUT	OSCIN FREQUENCY (MHz)
175ms	12
210ms	10
262ms	8
524ms	4

Effects of Stop and Wait Modes on the Timer, COP, and Serial Systems

Introduction

The STOP and WAIT instructions have different effects on the programmable timer, VPW Symbol Encoder/Decoder (SENDEC), and serial peripheral interface (SPI) systems. These 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 including the operation of the programmable timer, serial communications interface, and serial peripheral interface. The only way for the MCU to “wake up” from the STOP mode is by receipt of an external interrupt (logic low on IRQ pin), a negative edge on the $\overline{\text{VPWIN}}$ pin, or by the detection of a RESET (logic low on $\overline{\text{RESET}}$ pin or a power-on reset). Execution will resume at the instruction immediately following the STOP instruction that caused the HIP7030A2 to enter the STOP mode.

Normally a start-up delay of 4064 t_{CYC} is inserted after exiting from STOP before fetching the first instruction. This delay is intended to guarantee stability of a crystal clock source. If it is known that the clock source will be stable prior to exiting STOP, then the NDEL bit in the SEDCR can be set prior to executing the STOP instruction. Setting NDEL has the effect of shortening the start-up delay to 96 t_{CYC}.

The effects of the STOP mode on each of the MCU systems (COP, Timer, SENDEC, and SPI) are described separately in the following sections.

COP During STOP Mode

When the MCU enters the STOP mode, the Watchdog Timer and the Slow Clock Detect circuits are both inhibited.

Timer During STOP Mode

When the MCU enters the STOP mode, the timer counter stops counting (the internal processor 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.

SENDEC During STOP Mode

When the MCU enters the STOP mode, the absence of any internal clocks causes all SENDEC functions, except Wake Up to cease. If the SENDEC was currently being used to transmit a symbol, that symbol is truncated and the VPWOUT is forced to a low (passive) state. For proper operation, a STOP instruction should not be executed except when the bus is idle.

Normally all transitions are first filtered through the SENDECs 3-bit digital filter. When in STOP mode the 3-bit filter is bypassed and any passive to active transition (high to low) on $\overline{\text{VPWIN}}$ will cause a SENDEC interrupt which will, in turn, cause the processor to exit the STOP mode.

Upon exiting the STOP mode the processor will execute a SENDEC interrupt. The setting of the TX bit in the SEDSR does *not* bypass the 7µs filter and as such the TX bit will *not* be set when first awakening from STOP. If the NDEL bit has been set prior to entering STOP, software should delay 8µs and check TX. If at that time TX has not been set, the assumption can be made that a noise pulse caused the wakeup and the STOP mode can be reentered. When NDEL is not employed monitoring of TX must continue for several hundred microseconds, as a complete message could have transpired during the oscillator start-up time.

During handling of a SENDEC interrupt following STOP, the SEDSR must be read at least one time to remove the source of the interrupt.

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. To ensure the SPI bus remains free for transfers, the MSTR bit in the SPCR is cleared, configuring the SPI pins in slave mode. If the STOP instruction is executed during an SPI transfer, in which the HIP7030A2 was the master, that transfer is aborted. 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 an $\overline{\text{IRQ}}$ or SENDEC interrupt 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.

Wait Mode

When the MCU enters the WAIT mode, the CPU clock is halted. All CPU action is suspended; however, the timer, SENDEC, and SPI systems remain active. In fact an interrupt from the timer, SENDEC, or SPI (in addition to a logic low on the $\overline{IRQ}$ or $\overline{RESET}$ pins) 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.

Note that the Slow Clock Detect and Watchdog Timer circuitry continues to function during WAIT. It is requisite upon the designed to ensure that the CPU is removed from WAIT (via an external or TIMER or SENDEC interrupt) frequently enough to prevent a Watchdog Timer overflow.

The WAIT mode power consumption depends on how many systems are active. The power consumption will be highest when all the systems (timer, TCMP, SENDEC, and SPI) are active. The power consumption will be the least when the SENDEC and SPI systems are 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.

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 11.

Read-Modify-Write Instructions

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to Table 12.

TABLE 11. REGISTER/MEMORY INSTRUCTIONS

FUNCTION	MNEM	ADDRESSING MODES																	
		IMMEDIATE			DIRECT			EXTENDED			INDEXED (NO OFFSET)			INDEXED (8-BIT OFFSET)			INDEXED (16-BIT OFFSET)		
		OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. 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	A8	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	2	CD	3	3	FD	1	5	ED	2	6	DD	3	7

TABLE 12. READ-MODIFY-WRITE INSTRUCTIONS

FUNCTION	MNEM	ADDRESSING MODES														
		INHERENT (A)			INHERENT (X)			DIRECT			INDEXED (NO OFFSET)			INDEXED (8-BIT OFFSET)		
		OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. CYCLES	OP CODE	NO. BYTES	NO. 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	1	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 13.

TABLE 13. BRANCH INSTRUCTIONS

FUNCTION	MNEM	RELATIVE ADDRESSING MODE		
		OP CODE	NO. BYTES	NO. 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), serial peripheral status register ($0B), serial communications status register (10), timer status register ($13), and timer input capture register ($14 - $15). All port registers, port DDRs, timer, two serial systems, on-chip RAM, and 48 bytes of ROM reside in the first 256 bytes (page 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 14.

TABLE 14A. BIT SET/CLEAR INSTRUCTIONS

FUNCTION	MNEM	OP CODE	NO. BYTES	NO. CYCLES
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

TABLE 14B. BIT TEST AND BRANCH INSTRUCTIONS

FUNCTION	MNEM	OP CODE	NO. BYTES	NO. 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

Control Instructions

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to Table 15.

TABLE 15. CONTROL INSTRUCTIONS

FUNCTION	MNEM	INHERENT		
		OP CODE	NO. BYTES	NO. 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 16.

Opcode Map

Table 17 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 17 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.

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 most on-chip RAM and all I/O registers. 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 (S) 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 ← PC + 2

Address Bus High ← K; Address Bus Low ← 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 ← PC + 3

Address Bus High ← (PC + 1) + K

Address Bus Low ← X + (PC + 2)

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

Relative

Relative addressing is only used 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 ← EA if branch taken;

otherwise, EA = PC ← 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 ← PC + 2

Address Bus High ← 0; Address Bus Low ← (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)

Address Bus High Q 0; Address Bus Low ← (PC + 1)

EA2 = PC + 3 + (PC + 2); PC ← EA2 if branch taken;

otherwise, PC ← PC + 3.

Power Considerations

The average chip-junction temperature, T_J, in ^oC can be obtained from:

$$T_J = T_A + (PD \bullet \theta_{JA}) \quad \text{(EQ. 1)}$$

Where: T_A = Ambient Temperature, ^oC

θ_{JA} = Package Thermal Resistance Junction-to-Ambient, $^oC/W$

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

$P_{INT} = I_{CC} \bullet V_{CC}$, 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.

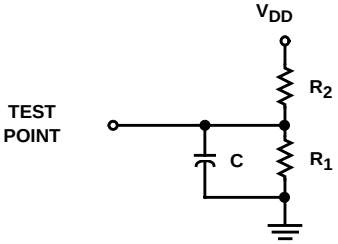
An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

$$P_D = K / (T_J + 273^oC) \quad \text{(EQ. 2)}$$

Solving equations 1 and 2 for K gives:

$$K = P_D \bullet (T_A + 273^oC) + \theta_{JA} \bullet P_D^2 \quad \text{(EQ. 3)}$$

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 know T_A. Using this value of K, the values of P_D and T_J can be obtained by solving Equation 1 and Equation 2 iteratively for any value of T_A.



EQUIVALENT TEST LOAD (SEE TABLE FOR VALUES OF R_1 AND R_2)

PINS	R_1	R_2	C
V_{DD} = 4.5V PA0 - PA7, PD0 - PD4	3.26kΩ	2.38kΩ	50pF
MISO, MOSI, SCK	1.9kΩ	2.2kΩ	200pF

TABLE 16. INSTRUCTION SET

	ADDRESSING MODES										CONDITION CODES				
MNEM	INHERENT	IMMEDIATE	DIRECT	EXTENDED	RELATIVE	INDEXED (NO OFFSET)	INDEXED (8 BITS)	INDEXED (16 BITS)	BIT SET/CLEAR	BIT TEST AND BRANCH	H	I	N	Z	C
ADC		X	X	X		X	X	X			Λ	•	Λ	Λ	Λ
ADD		X	X	X		X	X	X			Λ	•	Λ	Λ	Λ
AND		X	X	X		X	X	X			•	•	Λ	•	Λ
ASL	X		X			X	X				•	•	Λ	Λ	Λ
ASR	X		X			X	X				•	•	Λ	Λ	Λ
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			•	•	Λ	Λ	•
BLO					X						•	•	•	•	•
BLS					X						•	•	•	•	•
BMC					X						•	•	•	•	•
BMI					X						•	•	•	•	•
BMS					X						•	•	•	•	•
BNE					X						•	•	•	•	•
BPL					X						•	•	•	•	•
BRA					X						•	•	•	•	•
BRN					X						•	•	•	•	•
BRCLR										X	•	•	•	•	Λ
BRSET										X	•	•	•	•	Λ
BSET									X		•	•	•	•	•
BSR					X						•	•	•	•	•
CLC	X										•	•	•	•	0
CLI	X										•	0	•	•	•
CLR	X		X			X	X				•	•	0	1	•
CMP		X	X	X		X	X	X			•	•	Λ	Λ	Λ
COM	X		X			X	X				•	•	Λ	Λ	1
CPX		X	X	X		X	X	X			•	•	Λ	Λ	
DEC	X		X			X	X				•	•	Λ	Λ	•

TABLE 16. INSTRUCTION SET (Continued)

MNEM	ADDRESSING MODES										CONDITION CODES				
	INHERENT	IMMEDIATE	DIRECT	EXTENDED	RELATIVE	INDEXED (NO OFFSET)	INDEXED (8 BITS)	INDEXED (16 BITS)	BIT SET/CLEAR	BIT TEST AND BRANCH	H	I	N	Z	C
EOR		X	X	X		X	X	X			•	•	Λ	Λ	•
INC	X		X								•	•	Λ	Λ	•
JMP			X	X		X	X	X			•	•	•	•	•
JSR			X	X		X	X	X			•	•	•	•	•
LDA		X	X	X		X	X	X			•	•	Λ	Λ	•
LDX		X	X	X		X	X	X			•	•	Λ	Λ	•
LSL	X		X			X	X				•	•	Λ	Λ	Λ
LSR	X		X			X	X				•	•	0	Λ	Λ
MUL	X										0	•	•	•	0
NEG	X		X			X	X				•	•	Λ	Λ	Λ
NOP	X										•	•	•	•	•
ORA		X	X	X		X	X	X			•	•	Λ	Λ	•
ROL	X		X			X	X				•	•	Λ	Λ	Λ
ROR	X		X			X	X				•	•	Λ	Λ	Λ
RSP	X										•	•	•	•	•
RTI	X										?	?	?	?	?
RTS	X										•	•	•	•	•
SBC		X	X	X		X	X	X			•	•	Λ	Λ	Λ
SEC	X										•	•	•	•	1
SEI	X										•	1	•	•	•
STA			X	X		X	X	X			•	•	Λ	Λ	•
STOP	X										•	0	•	•	•
STX			X	X		X	X	X			•	•	Λ	Λ	•
SUB		X	X	X		X	X	X			•	•	Λ	Λ	Λ
SWI	X										•	1	•	•	•
TAX	X										•	•	•	•	•
TST	X		X			X	X				•	•	Λ	Λ	•
TXA	X										•	•	•	•	•
WAIT	X										•	0	•	•	•

Condition Code Symbols:

H = Half Carry (from Bit 3)

I = Interrupt Mask

N = Negate (Sign Bit)

Z = Zero 0 = Cleared

Λ = Test and Set if True Cleared Otherwise

• = Not Affected

? = Load CC Register From Stack

C = Carry/Borrow 1 = Set

Bit 7 6 5 4 3 2 1 0

TABLE 17. 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	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	HI / LOW
0	BRSET0 3 BTB	BSET0 2 BSC	BRA 2 REL	NEG 2 DIR	NEGA 1 INH	NEGX 1 INH	NEG 2 IX1	NEG 1 IX	RTI 1 INH		SUB 2 IMM	SUB 2 DIR	SUB 3 EXT	SUB 3 IX2	SUB 2 IX1	SUB 1 IX	0
1	BRCLR0 3 BTB	BCLR0 2 BSC	BRN 2 REL						RTS 1 INH		CMP 2 IMM	CMP 2 DIR	CMP 3 EXT	CMP 3 IX2	CMP 2 IX1	CMP 1 IX	1
2	BRSET1 3 BTB	BSET1 2 BSC	BHI 2 REL		MUL 1 INH						SBC 2 IMM	SBC 2 DIR	SBC 3 EXT	SBC 3 IX2	SBC 2 IX1	SBC 1 IX	2
3	BRCLR1 3 BTB	BCLR1 2 BSC	BLS 2 REL	COM 2 DIR	COMA 1 INH	COMX 1 INH	COM 2 IX1	COM 1 IX	SWI 1 INH		CPX 2 IMM	CPX 2 DIR	CPX 3 EXT	CPX 3 IX2	CPX 2 IX1	CPX 1 IX	3
4	BRSET2 3 BTB	BSET2 2 BSC	BCC 2 REL	LSR 2 DIR	LSRA 1 INH	LSRX 1 INH	LSR 2 IX1	LSR 1 IX			AND 2 IMM	AND 2 DIR	AND 3 EXT	AND 3 IX2	AND 2 IX1	AND 1 IX	4
5	BRCLR2 3 BTB	BCLR2 2 BSC	BCS 2 REL								BIT 2 IMM	BIT 2 DIR	BIT 3 EXT	BIT 3 IX2	BIT 2 IX1	BIT 1 IX	5
6	BRSET3 3 BTB	BSET3 2 BSC	BNE 2 REL	ROR 2 DIR	RORA 1 INH	RORX 1 INH	ROR 2 IX1	ROR 1 IX			LDA 2 IMM	LDA 2 DIR	LDA 3 EXT	LDA 3 IX2	LDA 2 IX1	LDA 1 IX	6
7	BRCLR3 3 BTB	BCLR3 2 BSC	BEQ 2 REL	ASR 2 DIR	ASRA 1 INH	ASRX 1 INH	ASR 2 IX1	ASR 1 IX		TAX 1 INH		STA 2 DIR	STA 3 EXT	STA 3 IX2	STA 2 IX1	STA 1 IX	7
8	BRSET4 3 BTB	BSET4 2 BSC	BHCC 2 REL	LSL 2 DIR	LSLA 1 INH	LSLX 1 INH	LSL 2 IX1	LSL 1 IX		CLC 1 INH	EOR 2 IMM	EOR 2 DIR	EOR 3 EXT	EOR 3 IX2	EOR 2 IX1	EOR 1 IX	8
9	BRCLR4 3 BTB	BCLR4 2 BSC	BHCS 2 REL	ROL 2 DIR	ROLA 1 INH	ROLX 1 INH	ROL 2 IX1	ROL 1 IX		SEC 1 INH	ADC 2 IMM	ADC 2 DIR	ADC 3 EXT	ADC 3 IX2	ADC 2 IX1	ADC 1 IX	9
A	BRSET5 3 BTB	BSET5 2 BSC	BPL 2 REL	DEC 2 DIR	DECA 1 INH	DECX 1 INH	DEC 2 IX1	DEC 1 IX		CLI 1 INH	ORA 2 IMM	ORA 2 DIR	ORA 3 EXT	ORA 3 IX2	ORA 2 IX1	ORA 1 IX	A
B	BRCLR5 3 BTB	BCLR5 2 BSC	BMI 2 REL							SEI 1 INH	ADD 2 IMM	ADD 2 DIR	ADD 3 EXT	ADD 3 IX2	ADD 2 IX1	ADD 1 IX	B
C	BRSET6 3 BTB	BSET6 2 BSC	BMC 2 REL	INC 2 DIR	INCA 1 INH	INCX 1 INH	INC 2 IX1	INC 1 IX		RSP 1 INH		JMP 2 DIR	JMP 3 EXT	JMP 3 IX2	JMP 2 IX1	JMP 1 IX	C

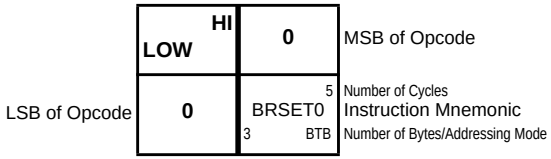
TABLE 17. INSTRUCTION SET OPCODE MAP (Continued)

	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	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	HI / LOW
D	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
E	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
F	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

INH = Inherent
IMM = Immediate
DIR = Direct
EXT = Extended

REL = Relative
BSC = Bit Set/Clear
BTB = Bit Test and Branch
BTB = Bit Test and Branch

IX = Indexed, No Offset
IX1 = Indexed, 8-Bit Offset
IX2 = Indexed, 16-Bit Offset



$0000	I/O	I/O	I/O	I/O	I/O	I/O	I/O	I/O	PORT A
	8	9	10	11	12	13	14	15	Pin Numbers
	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	Pin Name
$0001	UNUSED								
$0002	UNUSED								
$0003	CMP3	CMP2	0	I/O	I/O	I/O	I/O	I/O	PORT D
	18	19	-	17	18	19	20	21	Pin Numbers
	V3>VREF	V2>VREF	-	PD4/VREF	PD3/V2	PD2/V1	PD1	PD0	Pin Name
$0004	I/O	I/O	I/O	I/O	I/O	I/O	I/O	I/O	DDRA
$0005	UNUSED								
$0006	UNUSED								
$0007	0	0	0	I/O	I/O	I/O	I/O	I/O	DDRD
$0008	0	0	CMPE	0	0	0	STE1	STE0	SFRD
$0009	UNUSED								
$000A	SPIE	SPE	-	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
$000B	SPIF	WCOL	0	MODF	0	0	0	0	SPSR
$000C	SPI DATA REGISTER								SPDR
$000D	UNUSED								
$000E	UNUSED								
$000F	TXIE	-	-	NDEL	-	4X	PRE1	PRE0	SEDCR
$0010	TX	BRK	NEW	NOIZ	OVR	TALK	NECHO	0	SEDSR
$0011	FSOF	S2	S1	S0	LEV	R2	R1	R0	SEDDR
$0012	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL	TCR
$0013	ICF	OCF	TOF	0	0	0	0	0	TSR
$0014	Bit 15							Bit 8	CAPHI
$0015	Bit 7							Bit 0	CAPLO
$0016	Bit 15							Bit 8	CMPHI
$0017	Bit 7							Bit 0	CMPLO
$0018	Bit 15							Bit 8	CNTHI
$0019	Bit 7							Bit 0	CNTLO
$001A	Bit 15							Bit 8	ALTHI
$001B	Bit 7							Bit 0	ALTLO
$001C	UNUSED								
$001D	$55/$AA								WRR
$001E	0	0	0	0	0	0	1	WDF	WSR
$001F	RESERVED								RESERVED

I/O, CONTROL, STATUS, AND DATA REGISTER DEFINITIONS

Ordering Information Sheet

A. Package Type (select one):

☐ 28 Ld Dual-In-Line Plastic (E)

☐ 28 Ld SO (M)

B. Choose from the following microcomputer option. A manufacturing mask will be generated from this information. Refer to data sheet or data book instructions for submitting data for ROM patterns.

OSCB - Buffered Oscillator Output (select one)

☐ Enabled

☐ Disabled

C. Customer Company _____

Address _____

City _____

Phone (____) _____ Extension _____

Contact Person _____

Customer Part Number _____

D. Pattern Media (S-Record Formatted File Should Be Used - Unspecified locations are filled with 0's)

Floppy Disk: ☐ $3^1/_2$" ☐ $5^1/_4$" MODEM Upload: ☐ S-Record Filename _____

Medium if other than above † _____

Signature _____ Title _____

† The HIP7030A2 requires 8K of data

Date _____

ROM Ordering Instructions

The HIP7030A2 family of microcontrollers contains mask programmed ROMs. The contents of these ROMs are personalized to meet a customer's code requirements during manufacturing of the ICs. The code is programmed via photomasking techniques. Semiconductor manufacturing is a batch process, and all microcontrollers manufactured in a given lot (a batch) will contain identical ROM code.

Intersil generates a customer's ROM mask from an ASCII representation of the desired ROM contents together with other specific information. The preceding page contains a sheet which can be used to provide the required information when ordering a masked ROM microcontroller.

Data Format Options

The ROM data can be submitted in various formats. The following list summarizes the principal formats which Intersil will accept. The list is in order of preference, with S-Record formatted data files being the preferred format.

- **S-Record Formatted Hex Data File via modem upload**
- **S-Record Formatted Hex Data File via e-mail**
- **S-Record Formatted Hex Data File on floppy disk**
- **6805 Assembly Language Source File on floppy disk**
- **Contents of a 27XX type EPROM/EEPROM**

Regardless of the medium used to transfer the data, contents of all of the User ROM regions of the memory map of the particular microcontroller should be specified. This includes any Page 0 User ROM and User Reset/Interrupt Vectors. Data should not be specified for the Self Check ROM space of a device. All unused locations should either not be specified (S-Record and source files) or specified as $00 (EPROM/EEPROM). Any unspecified locations will be filled with $00 by Intersil.

Procedure for Submitting Data

When submitting data via a physical medium such as a floppy disk or EPROM, the "Ordering Information Sheet" on the preceding page must be completed and submitted with the data.

When utilizing the Intersil Customer Pattern Retrieval System (modem upload) the customer will be prompted for the same information as that specified on the "Ordering Information Sheet".

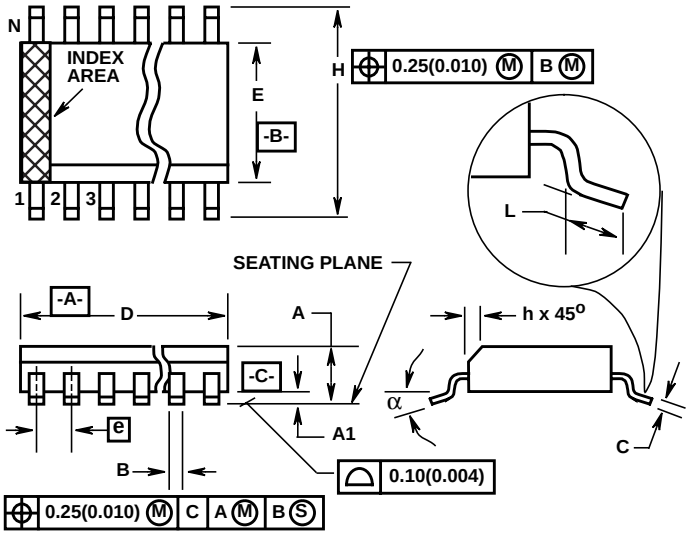
If the data is submitted via e-mail, the message should include the same information as that specified on the "Ordering Information Sheet".

Intersil Customer Pattern Retrieval System

To access the Intersil Customer Pattern Retrieval System, you must first obtain an account ID and password from your Intersil sales representative. The system is accessed by dialing 1-908-685-6541. It is presently set to run with baud rates up to 2400 baud, with 8 data bits, 1 stop bit, and no parity bit. The data transfer is done using text mode Kermit transfers.

Check the Intersil Corporate Internet Site, http://www.intersil.com, for the latest information on the Intersil Customer Pattern Retrieval System.

Small Outline Plastic Packages (SOIC)



M28.3 (JEDEC MS-013-AE ISSUE C)
28 LEAD WIDE BODY SMALL OUTLINE PLASTIC PACKAGE

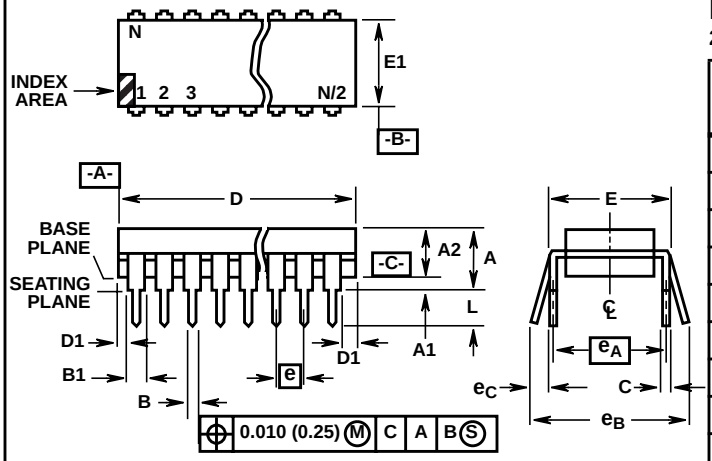
SYMBOL	INCHES		MILLIMETERS		NOTES
	MIN	MAX	MIN	MAX	
A	0.0926	0.1043	2.35	2.65	-
A1	0.0040	0.0118	0.10	0.30	-
B	0.013	0.0200	0.33	0.51	9
C	0.0091	0.0125	0.23	0.32	-
D	0.6969	0.7125	17.70	18.10	3
E	0.2914	0.2992	7.40	7.60	4
e	0.05 BSC		1.27 BSC		-
H	0.394	0.419	10.00	10.65	-
h	0.01	0.029	0.25	0.75	5
L	0.016	0.050	0.40	1.27	6
N	28		28		7
α	0^o	8^o	0^o	8^o	-

Rev. 0 12/93

NOTES:

1. Symbols are defined in the “MO Series Symbol List” in Section 2.2 of Publication Number 95.
2. Dimensioning and tolerancing per ANSI Y14.5M-1982.
3. Dimension “D” does not include mold flash, protrusions or gate burrs. Mold flash, protrusion and gate burrs shall not exceed 0.15mm (0.006 inch) per side.
4. Dimension “E” does not include interlead flash or protrusions. Interlead flash and protrusions shall not exceed 0.25mm (0.010 inch) per side.
5. The chamfer on the body is optional. If it is not present, a visual index feature must be located within the crosshatched area.
6. “L” is the length of terminal for soldering to a substrate.
7. “N” is the number of terminal positions.
8. Terminal numbers are shown for reference only.
9. The lead width “B”, as measured 0.36mm (0.014 inch) or greater above the seating plane, shall not exceed a maximum value of 0.61mm (0.024 inch)
10. Controlling dimension: MILLIMETER. Converted inch dimensions are not necessarily exact.

Dual-In-Line Plastic Packages (PDIP)



E28.6 (JEDEC MS-011-AB ISSUE B)
28 LEAD DUAL-IN-LINE PLASTIC PACKAGE

SYMBOL	INCHES		MILLIMETERS		NOTES
	MIN	MAX	MIN	MAX	
A	-	0.250	-	6.35	4
A1	0.015	-	0.39	-	4
A2	0.125	0.195	3.18	4.95	-
B	0.014	0.022	0.356	0.558	-
B1	0.030	0.070	0.77	1.77	8
C	0.008	0.015	0.204	0.381	-
D	1.380	1.565	35.1	39.7	5
D1	0.005	-	0.13	-	5
E	0.600	0.625	15.24	15.87	6
E1	0.485	0.580	12.32	14.73	5
e	0.100 BSC		2.54 BSC		-
e_A	0.600 BSC		15.24 BSC		6
e_B	-	0.700	-	17.78	7
L	0.115	0.200	2.93	5.08	4
N	28		28		9

Rev. 0 12/93

NOTES:

1. Controlling Dimensions: INCH. In case of conflict between English and Metric dimensions, the inch dimensions control.
2. Dimensioning and tolerancing per ANSI Y14.5M-1982.
3. Symbols are defined in the “MO Series Symbol List” in Section 2.2 of Publication No. 95.
4. Dimensions A, A1 and L are measured with the package seated in JEDEC seating plane gauge GS-3.
5. D, D1, and E1 dimensions do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.010 inch (0.25mm).
6. E and e_A are measured with the leads constrained to be perpendicular to datum -C-.
7. e_B and e_C are measured at the lead tips with the leads unconstrained. e_C must be zero or greater.
8. B1 maximum dimensions do not include dambar protrusions. Dambar protrusions shall not exceed 0.010 inch (0.25mm).
9. N is the maximum number of terminal positions.
10. Corner leads (1, N, N/2 and N/2 + 1) for E8.3, E16.3, E18.3, E28.3, E42.6 will have a B1 dimension of 0.030 - 0.045 inch (0.76 - 1.14mm).

Sales Office Headquarters

NORTH AMERICA
Intersil Corporation
P. O. Box 883, Mail Stop 53-204
Melbourne, FL 32902
TEL: (407) 724-7000
FAX: (407) 724-7240

EUROPE
Intersil SA
Mercure Center
100, Rue de la Fusee
1130 Brussels, Belgium
TEL: (32) 2.724.2111
FAX: (32) 2.724.22.05

ASIA
Intersil (Taiwan) Ltd.
Taiwan Limited
7F-6, No. 101 Fu Hsing North Road
Taipei, Taiwan
Republic of China
TEL: (886) 2 2716 9310
FAX: (886) 2 2715 3029