National Semiconductor

August 2000

COP912C
8-Bit Microcontroller

General Description

Note: COP8SA devices are instruction set and pinout compatible supersets of the COP912C devices, and are replacements for these in new designs when possible.

The COP912C ROM based microcontrollers are integrated COP8(tm) Base core devices with smaller memory (768 bytes), and fewer on-board features. These single-chip CMOS devices are suited for lower-functionality applications where system cost is of prime consideration. Pin and software compatible (different Vcc range) 4k/32k OTP versions are available (COP87LxxCJ/RJ Family). Erasable windowed versions are available for use with a range of COP8(tm) software and hardware development tools.

Family features include an 8-bit memory mapped architecture, 10MHz CKI with 2.5us(912C) or 2us(912CH) instruction cycle, one multi-function 16-bit timer/counter with PWM, MICROWIRE/PLUS(tm) serial I/O, power saving HALT mode, three clock modes, high current outputs, software selectable I/O options, multi-volt operation and 20 pin packages.

Devices included in this datasheet are:

Device	Memory (bytes)	RAM (bytes)	I/O Pins	Packages	Temperature	Comments
COP912C	768 ROM	64	16	20 DIP/SOIC	0 to +70˚C	2.3v - 4.0v
COP912CH	768ROM	64	16	20 DIP/SOIC	0 to +70˚C	4.0v - 5.5v

Key Features

- Lowest cost COP8 microcontroller
- 16-bit multi-function timer supporting
  - — PWM mode
  - — External event counter mode
  - — Input capture mode
- 768 bytes of ROM
- 64 bytes of RAM

I/O Features

- Memory mapped I/O
- Software selectable I/O options (TRI-STATE® Output, Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- Schmitt trigger inputs on Port G
- MICROWIRE/PLUS™ Serial I/O
- Packages: 20 DIP/SO with 16 I/O pins

CPU/Instruction Set Features

- Instruction cycle time of 2 µs for COP912CH and 2.5 µs for COP912C
- Three multi-sourced interrupts servicing
  - — External Interrupt with selectable edge
  - — Timer interrupt
  - — Software interrupt
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Register Indirect Memory Pointers (B, X)

Fully Static CMOS

- Low current drain (typically < 1 µA)
- Single supply operation: 2.3V to 4.0V or 4.0V to 5.5V
- Temperature range: 0˚C to +70˚C

Development Support

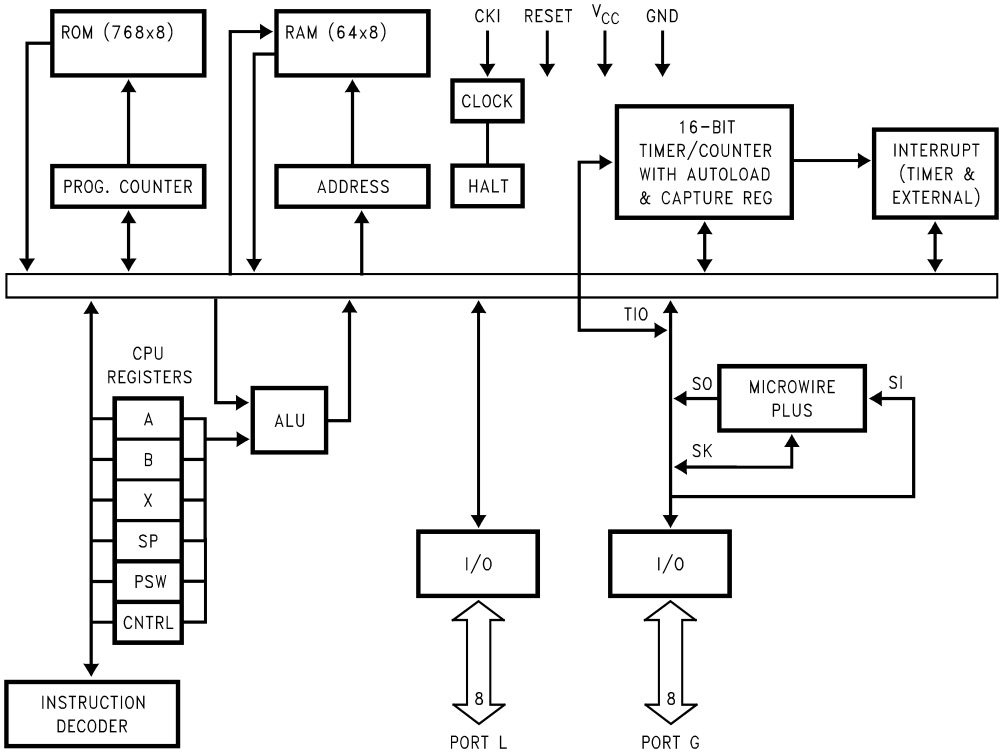
- Emulation and OTP devices
- Real time emulation and full program debug offered by MetaLink Development System

Applications

- Electronic keys and switches
- Remote Control
- Timers
- Alarms
- Small industrial control units
- Low cost slave controllers
- Temperature meters
- Small domestic appliances
- Toys and games

TRI-STATE® is a registered trademark of National Semiconductor Corporation.
COP8™, MICROWIRE/PLUS™, WATCHDOG™ and MICROWIRE™ are trademarks of National Semiconductor Corporation.
PC® is a registered trademark of International Business Machines Corp.
iceMaster™ is a trademark of MetaLink Corporation.

Block Diagram



Absolute Maximum Ratings (Note 1)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Supply Voltage (V_{CC})	6.0V
Voltage at Any Pin	−0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	80 mA
Total Current out of GND Pin (Sink)	80 mA
Storage Temperature Range	−65˚C to +150˚C

Note 1: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

COP912C/COP912CH; 0˚C ≤ T_A ≤ +70˚C unless other specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage					
912C		2.3		4.0	V
912CH		4.0		5.5	V
Power Supply Ripple 1 (Note 2)	Peak to Peak			0.1 V_{CC}	V
Supply Current (Note 3)					
CKI = 4 MHz	V_{CC} = 5.5V, tc = 2.5 µs			6.0	mA
CKI = 4 MHz	V_{CC} = 4.0V, tc = 2.5 µs			2.5	mA
HALT Current	V_{CC} = 5.5V, CKI = 0 MHz		<1	8	µA
INPUT LEVELS (V_{IH}, V_{IL})					
Reset, CKI:					
Logic High		0.9 V_{CC}			V
Logic Low				0.1 V_{CC}	V
All Other Inputs					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage/TRI-STATE Leakage	V_{CC} = 5.5V	−2		+2	µA
Input Pullup Current	V_{CC} = 5.5V			250	µA
G-Port Hysteresis			0.05 V_{CC}	0.35 V_{CC}	V
Output Current Levels					
Source (Push-Pull Mode)	V_{CC} = 4.0V, V_{OH} = 3.8V	0.4			mA
	V_{CC} = 2.3V, V_{OH} = 1.8V	0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.0V, V_{OL} = 1.0V	4.0			mA
	V_{CC} = 2.3V, V_{OL} = 0.4V	0.7			mA
Allowable Sink/Source Current Per Pin				3	mA
Input Capacitance (Note 4)				7	pF
Load Capacitance on D2 (Note 4)				1000	pF

Note 2: Rate of voltage change must be less then 0.5 V/ms.

Note 3: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 4: Characterized, not tested.

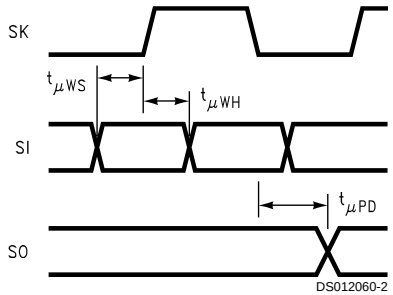
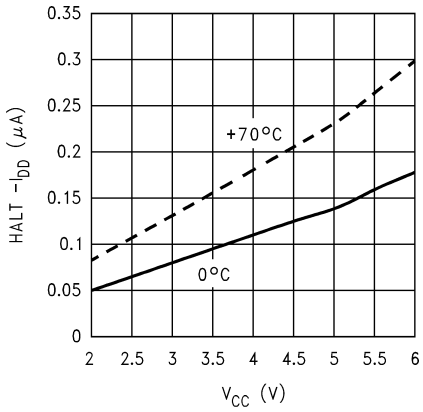


FIGURE 1. MICROWIRE/PLUS Timing

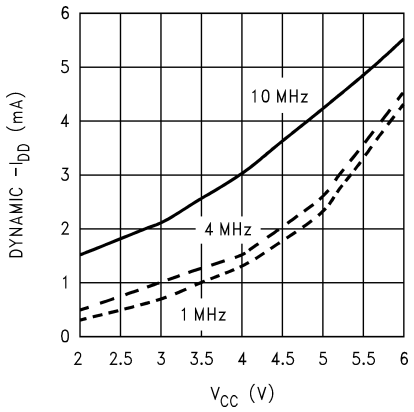
Typical Performance Characteristics

Halt — I_{DD}



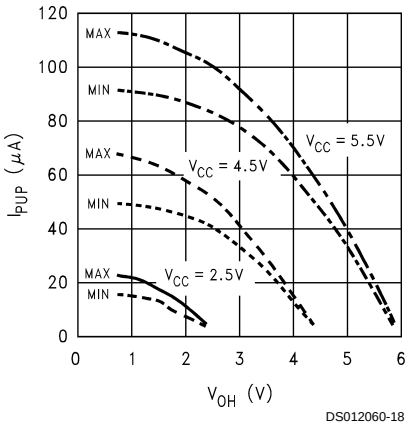
DS012060-16

Dynamic — I_{DD} (Crystal Clock Option)



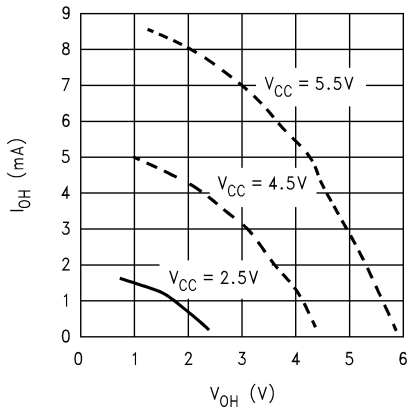
DS012060-17

Port L/G Weak Pull-Up Source Current



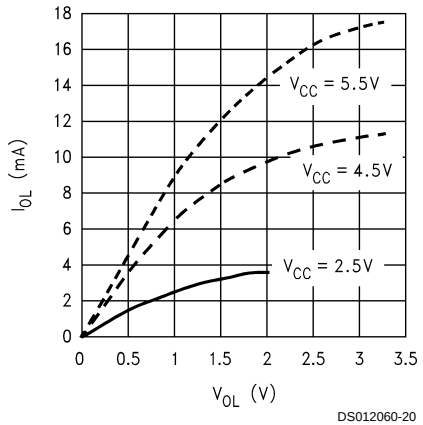
DS012060-18

Port L/G Push-Pull Source Current



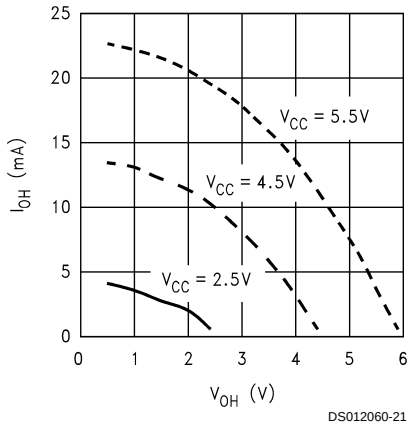
DS012060-19

Port L/G Push-Pull Sink Current



DS012060-20

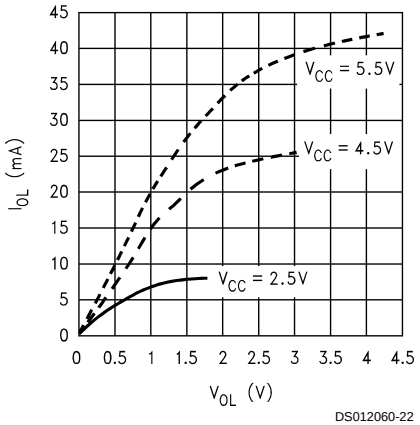
Port D Source Current



DS012060-21

Typical Performance Characteristics (Continued)

Port D Sink Current



DS012060-22

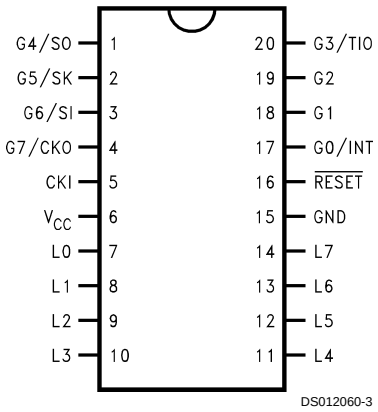
AC Electrical Characteristics

COP912C/COP912CH; 0˚C ≤ T_A ≤ +70˚C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
INSTRUCTION CYCLE TIME (tc)					
Crystal/Resonator	4.0V ≤ V_{CC} ≤ 5.5V	2		DC	µs
	2.3V ≤ V_{CC} < 4.0V	2.5		DC	µs
R/C Oscillator	4.0V ≤ V_{CC} ≤ 5.5V	3		DC	µs
	2.3V ≤ V_{CC} < 4.0V	7.5		DC	µs
Inputs					
t_{Setup}	4.0V ≤ V_{CC} ≤ 5.5V	200			ns
	2.3V ≤ V_{CC} < 4.0V	500			ns
t_{Hold}	4.0V ≤ V_{CC} ≤ 5.5V	60			ns
	2.3V ≤ V_{CC} < 4.0V	150			ns
Output Propagation Delay	R_L = 2.2 kΩ, C_L = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	4.0V ≤ V_{CC} ≤ 5.5V			0.7	µs
	2.3V ≤ V_{CC} < 4.0V			1.75	µs
All Others	4.0V ≤ V_{CC} ≤ 5.5V			1	µs
	2.3V ≤ V_{CC} < 4.0V			5	µs
Input Pulse Width					
Interrupt Input High Time		1 tc			
Interrupt Input Low Time		1 tc			
Timer Input High Time		1 tc			
Timer Input Low Time		1 tc			
MICROWIRE Setup Time ($t_{\mu WS}$)		20			ns
MICROWIRE Hold Time ($t_{\mu WH}$)		56			ns
MICROWIRE Output				220	ns
Propagation Delay ($t_{\mu PD}$)					
Reset Pulse Width		1.0			µs

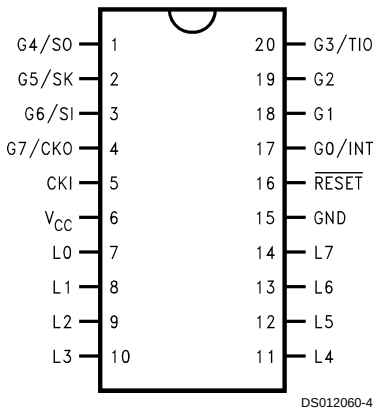
COP912C/COP912CH Pinout

20 DIP



Top View
Order Number COP912C-XXX/N, COP912CH-XXX/N

20 SO Wide



Top View
Order Number COP912C-XXX/WM, COP912CH-XXX/WM

FIGURE 2. COP912C/COP912CH Pinout

Pin Description

V_{CC} and **GND** are the power supply pins.

CKI is the clock input. This can come from an external source, a R/C generated oscillator or a crystal (in conjunction with CKO). See Oscillator description.

$\overline{\text{RESET}}$ is the master reset input. See Reset description.

PORT L is an 8-bit I/O port.

There are two registers associated to configure the L port: a data register and a configuration register. Therefore, each L I/O bit can be individually configured under software control as shown below:

Port L Config.	Port L Data	PORT L Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Three data memory address locations are allocated for this port, one each for data register [00D0], configuration register [00D1] and the input pins [00D2].

PORT G is an 8-bit port with 6 I/O pins (G0–G5) and 2 input pins (G6, G7).

All eight G-pins have Schmitt Triggers on the inputs.

There are two registers associated to configure the G port: a data register and a configuration register. Therefore each G port bit can be individually configured under software control as shown below:

Port G Config.	Port G Data	PORT G Setup
0	0	Hi-Z Input (TRI-STATE)
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

Three data memory address locations are allocated for this port, one for data register [00D4], one for configuration register [00D5] and one for the input pins [00D6]. Since G6 and G7 are Hi-Z input only pins, any attempt by the user to configure them as outputs by writing a one to the configuration register will be disregarded. Reading the G6 and G7 configuration bits will return zeroes. Note that the chip will be placed in the Halt mode by writing a “1” to the G7 data bit.

Six pins of Port G have alternate features:

G0 INTR (an external interrupt)

G3 TIO (timer/counter input/output)

G4 SO (MICROWIRE serial data output)

G5 SK (MICROWIRE clock I/O)

G6 SI (MICROWIRE serial data input)

G7 CKO crystal oscillator output (selected by mask option) or HALT restart input/general purpose input (if clock option is R/C- or external clock)

Pins G1 and G2 currently do not have any alternate functions.

The selection of alternate Port G functions are done through registers PSW [00EF] to enable external interrupt and CNTRL [00EE] to select TIO and MICROWIRE operations.

Functional Description

The internal architecture is shown in the block diagram. Data paths are illustrated in simplified form to depict how the various logic elements communicate with each other in implementing the instruction set of the device.

ALU AND CPU REGISTERS

The ALU can do an 8-bit addition, subtraction, logical or shift operations in one cycle time. There are five CPU registers:

A is the 8-bit Accumulator register

PC is the 15-bit Program Counter register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is the 8-bit address register and can be auto incremented or decremented

X is the 8-bit alternate address register and can be auto incremented or decremented.

Functional Description (Continued)

SP is the 8-bit stack pointer which points to the subroutine stack (in RAM).

B, X and SP registers are mapped into the on chip RAM. The B and X registers are used to address the on chip RAM. The SP register is used to address the stack in RAM during subroutine calls and returns. The SP must be preset by software upon initialization.

MEMORY

The memory is separated into two memory spaces: program and data.

PROGRAM MEMORY

Program memory consists of 768 x 8 ROM. These bytes of ROM may be instructions or constant data. The memory is addressed by the 15-bit program counter (PC). There are no "pages" of ROM, the PC counts all 15 bits. ROM can be indirectly read by the LAID instruction for table lookup.

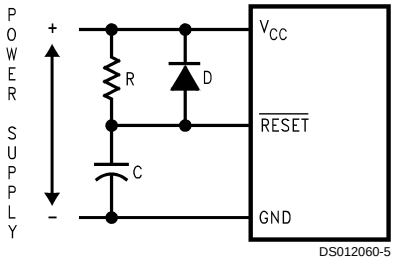
DATA MEMORY

The data memory address space includes on chip RAM, I/O and registers. Data memory is addressed directly by the instruction or indirectly through B, X and SP registers. The device has 64 bytes of RAM. Sixteen bytes of RAM are mapped as "registers", these can be loaded immediately, decremented and tested. Three specific registers: X, B, and SP are mapped into this space, the other registers are available for general usage.

Any bit of data memory can be directly set, reset or tested. I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested.

RESET

The RESET input pin when pulled low initializes the microcontroller. Upon initialization, the ports L and G are placed in the TRI-STATE mode. The PC, PSW and CNTRL registers are cleared. The data and configuration registers for ports L and G are cleared. The external RC network shown in *Figure 3* should be used to ensure that the RESET pin is held low until the power supply to the chip stabilizes.



RC > 5 x POWER SUPPLY RISE TIME

FIGURE 3. Recommended Reset Circuit

OSCILLATOR CIRCUITS

The device can be driven by a clock input which can be between DC and 5 MHz.

CRYSTAL OSCILLATOR

By selecting CKO as a clock output, CKI and CKO can be connected to create a crystal controlled oscillator. *Table 1* shows the component values required for various standard crystal values.

R/C OSCILLATOR

By selecting CKI as a single pin oscillator, CKI can make an R/C oscillator. CKO is available as a general purpose input and/or HALT control. *Table 2* shows variation in the oscillator frequencies as functions of the component (R and C) value.

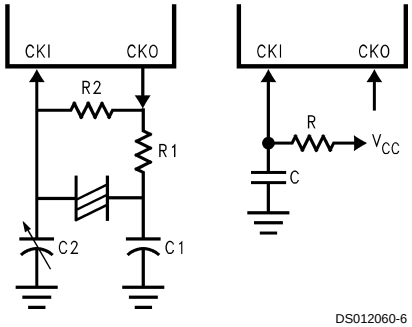


FIGURE 4. Clock Oscillator Configurations

TABLE 1. Crystal Oscillator Configuration

R1 (kΩ)	R2 (mΩ)	C1 (pF)	C2 (pF)	CKI Freq. (MHz)
0	1	30	30–36	5
0	1	30	30–36	4
5.6	1	200	100–150	0.455

TABLE 2. RC Oscillator Configuration (Part-to-Part Variation, T_A = 25˚C)

R (kΩ)	C (pF)	CKI Freq. (MHz)	Intr. Cycle (µs)
3.3	82	2.2 to 2.7	3.7 to 4.6
5.6	100	1.1 to 1.3	7.4 to 9
6.8	100	0.9 to 1.1	8.8 to 10.8

Note 5: 3k ≤ R ≤ 200 kΩ, 50 pF ≤ C ≤ 200 pF.

HALT MODE

The device is a fully static device. The device enters the HALT mode by writing a one to the G7 bit of the G data register. Once in the HALT mode, the internal circuitry does not receive any clock signal and is therefore frozen in the exact state it was in when halted. In this mode the chip will only draw leakage current.

The device supports two different ways of exiting the HALT mode. The first method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO is a dedicated output), and so may be used either with an RC clock configuration (or an external clock configuration). The second method of exiting the HALT mode is to pull the RESET low.

Note: To allow clock resynchronization, it is necessary to program two NOP's immediately after the device comes out of the HALT mode. The user must program two NOP's following the "enter HALT mode" (set G7 data bit) instruction.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., A/D converters, display driv-

Functional Description (Continued)

ers, EEPROMS etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). *Figure 5* shows a block diagram of the MICROWIRE logic.

The shift clock can be derived from either the internal source or from an external source. Operating the MICROWIRE arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE mode. To use the MICROWIRE, the MSEL bit in the CNTRL register is set to one. The SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register.

The following table details the different clock rates that may be selected.

SK Divide Clock Rates

SL1	SL0	SK
0	0	2 x tc
0	1	4 x tc
1	x	8 x tc
Where tc is the instruction cycle clock.		

MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 5* shows how two microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangement.

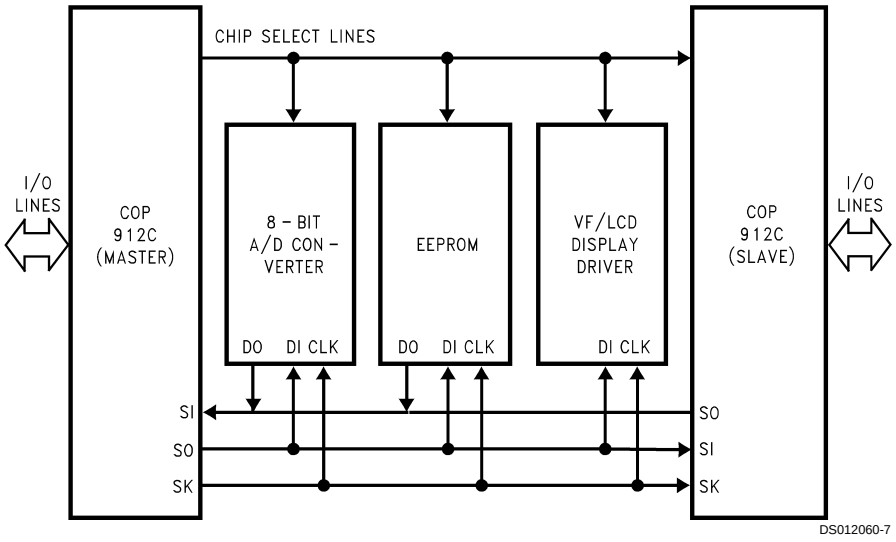


FIGURE 5. MICROWIRE/PLUS Application

WARNING: The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

Table 3 summarizes the settings required to enter the Master/Slave modes of operations.

The table assumes that the control flag MSEL is set.

TABLE 3. MICROWIRE/PLUS G Port Configuration

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Pin	G5 Pin	G6 Pin	Operation
1	1	SO	Int. SK	SI	MICROWIRE Master
0	1	TRI-STATE	Int. SK	SI	MICROWIRE Master
1	0	SO	Ext. SK	SI	MICROWIRE Slave
0	0	TRI-STATE	Ext. SK	SI	MICROWIRE Slave

MICROWIRE/PLUS MASTER MODE OPERATION

In MICROWIRE/PLUS Master mode operation, the SK shift clock is generated internally. The MSEL bit in the CNTRL register must be set to allow the SK and SO functions onto the G5 and G4 pins. The G5 and G4 pins must also be selected as outputs by setting the appropriate bits in the Port G configuration register. The MICROWIRE Master mode always initiates all data exchanges. The MSEL bit in the CNTRL register is set to enable MICROWIRE/PLUS. G4 and G5 are selected as output.

Functional Description (Continued)

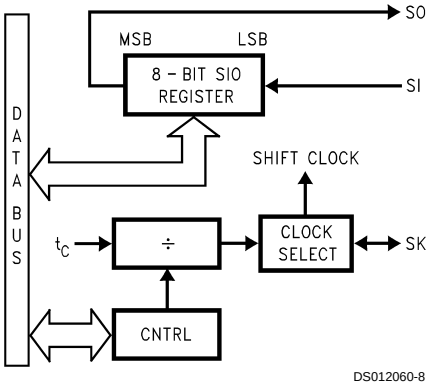


FIGURE 6. MICROWIRE/PLUS Block Diagram

MICROWIRE/PLUS SLAVE MODE

In MICROWIRE/PLUS Slave mode operation, the SK shift clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G port. The SK pin must be selected as an input and the SO pin as an output by resetting and setting their respective bits in the G port configuration register.

The user must set the BUSY flag immediately upon entering the slave mode. This will ensure that all data bits sent by the master will be shifted in properly. After eight clock pulses, the BUSY flag will be cleared and the sequence may be repeated.

Note: In the Slave mode the SIO register does not stop shifting even after the busy flag goes low. Since SK is an external output, the SIO register stops shifting only when SK is turned off by the master.

Note: Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO register to be narrow. When the BUSY flag is set, the MICROWIRE logic becomes active with the internal SIO shift clock enabled. If SK is high in slave mode, this will cause the internal shift clock to go from low in standby mode to high in active mode. This generates a rising edge, and causes one bit to be shifted into the SIO register from the SI input. For safety, the BUSY flag should only be set when the input SK clock is low.

Note: The SIO register must be loaded only when the SK shift clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register.

Timer/Counter

The device has an on board 16-bit timer/counter (organized as two 8-bit registers) with an associated 16-bit autoreload/capture register (also organized as two 8-bit registers). Both are read/write registers.

The timer has three modes of operation:

PWM (PULSE WIDTH MODULATION) MODE

The timer counts down at the instruction cycle rate (2 µs max). When the timer count underflows, the value in the autoreload register is copied into the timer. Consequently, the timer is programmable to divide by any value from 1 to 65536. Bit 5 of the timer CNTRL register selects the timer underflow to toggle the G3 output. This allows the user to generate a square wave output or a pulse-width-modulated output. The timer underflow can also be enabled to interrupt the processor. The timer PWM mode is shown in *Figure 7*.

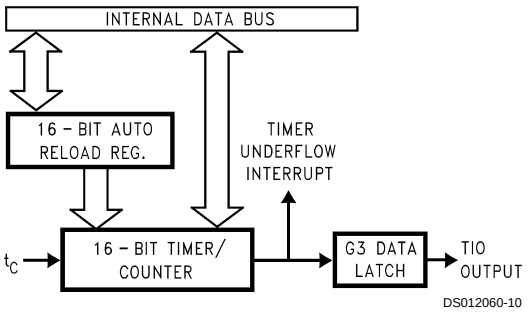


FIGURE 7. Timer in PWM Mode

EXTERNAL EVENT COUNTER MODE

In this mode, the timer becomes a 16-bit external event counter, clocked from an input signal applied to the G3 input. The maximum frequency for this G3 input clock is 250 kHz (half of the 0.5 MHz instruction cycle clock). When the external event counter underflows, the value in the autoreload register is copied into the timer. This timer underflow may also be used to generate an interrupt. Bit 5 of the CNTRL register is used to select whether the external event counter clocks on positive or negative edges from the G3 input. Consequently, half cycles of an external input signal could be counted. The External Event counter mode is shown in *Figure 8*.

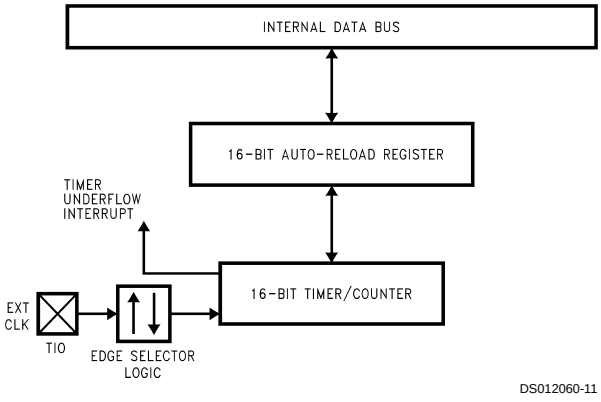


FIGURE 8. Timer in External Event Mode

INPUT CAPTURE MODE

In this mode, the timer counts down at the instruction clock rate. When an external edge occurs on pin G3, the value in the timer is copied into the capture register. Consequently, the time of an external edge on the G3 pin is “captured”. Bit 5 of the CNTRL register is used to select the polarity of the external edge. This external edge capture can also be programmed to generate an interrupt. The duration of an input signal can be computed by capturing the time of the leading edge, saving this captured value, changing the capture edge, capturing the time of the trailing edge, and then subtracting this trailing edge time from the earlier leading edge time. The Input Capture mode is shown in *Figure 9*.

Timer/Counter (Continued)

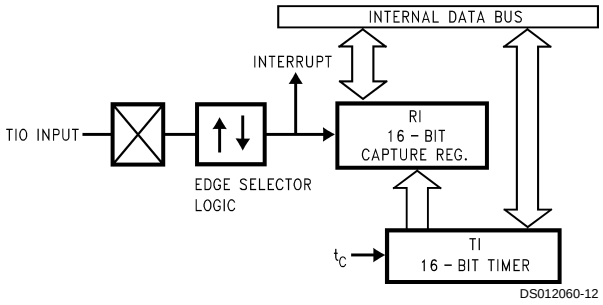


FIGURE 9. Timer in Input Capture Mode

Table 4 below details the TIMER modes of operation and their associated interrupts. Bit 4 of CNTRL is used to start and stop the timer/counter. Bits 5, 6 and 7 of the CNTRL register select the timer modes. The ENTI (Enable Timer Interrupt) and TPND (Timer Interrupt Pending) bits in the PSW register are used to control the timer interrupts.

Care must be taken when reading from and writing to the timer and its associated autoreload/capture register. The timer and autoreload/capture register are both 16-bit, but they are read from and written to one byte at a time. It is recommended that the timer be stopped before writing a new value into it. The timer may be read "on the fly" without stopping it if suitable precautions are taken. One method of reading the timer "on the fly" is to read the upper byte of the timer first, and then read the lower byte. If the most significant bit of the lower byte is then tested and found to be high, then the upper byte of the timer should be read again and this new value used.

TABLE 4. Timer Modes and Control Bits

CNTRL Bits			Operation Mode	Timer Interrupt	Timer Counts On
7	6	5			
0	0	0	External Event Counter with Autoreload Register	Timer Underflow	TIO Positive Edge
0	0	1	External Event Counter with Autoreload Register	Timer Underflow	TIO Negative Edge
0	1	0	Not Allowed	Not Allowed	Not Allowed
0	1	1	Not Allowed	Not Allowed	Not Allowed
1	0	0	Timer with Autoreload Register	Timer Underflow	tc
1	0	1	Timer with Autoreload Register and Toggle TIO Out	Timer Underflow	tc
1	1	0	Timer with Capture Register	TIO Positive Edge	tc
1	1	1	Timer with Capture Register	TIO Negative Edge	tc

TIMER APPLICATION EXAMPLE

The timer has an autoreload register that allows any frequency to be programmed in the timer PWM mode. The timer underflow can be programmed to toggle output bit G3, and may also be programmed to generate a timer interrupt. Consequently, a fully programmable PWM output may be easily generated.

The timer counts down and when it underflows, the value from the autoreload register is copied into the timer. The CNTRL register is programmed to both toggle the G3 output and generate a timer interrupt when the timer underflows. Following each timer interrupt, the user's program alternately loads the values of the "on" time and the "off" time into the timer autoreload register. Consequently, a pulse-width-modulated (PWM) output waveform is generated to a resolution of one instruction cycle time. This PWM application example is shown in *Figure 10*.

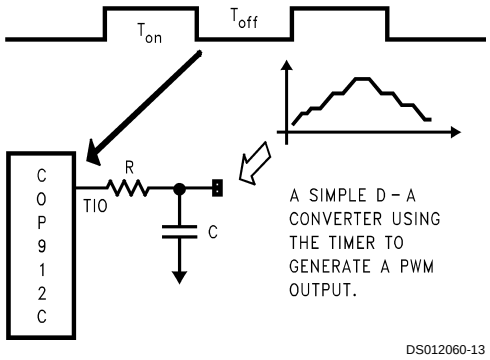


FIGURE 10. Timer Based PWM Application

Interrupts

There are three interrupt sources:

1. A maskable interrupt on external G0 input positive or negative edge sensitive under software control
2. A maskable interrupt on timer underflow or timer capture
3. A non-maskable software/error interrupt on opcode zero. The GIE (global interrupt enable) bit enables the interrupt function. This is used in conjunction with ENI and ENTI to select one or both of the interrupt sources. This bit is reset when interrupt is acknowledged.

ENI and ENTI bits select external and timer interrupt respectively. Thus the user can select either or both sources to interrupt the microcontroller when GIE is enabled. IEDG se-

Interrupts (Continued)

lects the external interrupt edge (1 = rising edge, 0 = falling edge). The user can get an interrupt on both rising and falling edges by toggling the state of IEDG bit after each interrupt.

IPND and TPND bits signal which interrupt is pending. After interrupt is acknowledged, the user can check these two bits to determine which interrupt is pending. The user can prioritize the interrupt and clear the pending bit that corresponds to the interrupt being serviced. The user can also enable GIE at this point for nesting interrupts. Two things have to be kept in mind when using the software interrupt. The first is that executing a simple RET instruction will take the program control back to the software interrupt instruction itself. In other words, the program will be stuck in an infinite loop. To avoid the infinite loop, the software interrupt service routine should end with a RETSK instruction or with a JMP instruction. The second thing to keep in mind is that unlike the other interrupt sources, the software interrupt does not reset the GIE bit. This means that the device can be interrupted by other interrupt sources while servicing the software interrupt.

Interrupts push the PC to the stack, reset the GIE bit to disable further interrupts and branch to address 00FF. The RETI instruction will pop the stack to PC and set the GIE bit to enable further interrupts. The user should use the RETI or the RET instruction when returning from a hardware (maskable) interrupt subroutine. The user should use the RETSK instruction when returning from a software interrupt subroutine to avoid an infinite loop situation.

The software interrupt is a special kind of non-maskable interrupt which occurs when the INTR instruction (opcode 00 used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped. When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

Hardware and Software interrupts are treated differently. The software interrupt is not gated by the GIE bit. However, it has the lowest arbitration ranking. Also the fact that all interrupts vector to the same address 00FF Hex means that a software interrupt happening at the same time as a hardware interrupt will be missed.

Note: There is always the possibility of an interrupt occurring during an instruction which is attempting to reset the GIE bit or any other interrupt enable bit. If this occurs when a single cycle instruction is being used to reset the interrupt enable bit, the interrupt enable bit will be reset but an interrupt may still occur. This is because interrupt processing is started at the same time as the interrupt bit is being reset. To avoid this scenario, the user should always use a two, three, or four cycle instruction to reset interrupt enable bits.

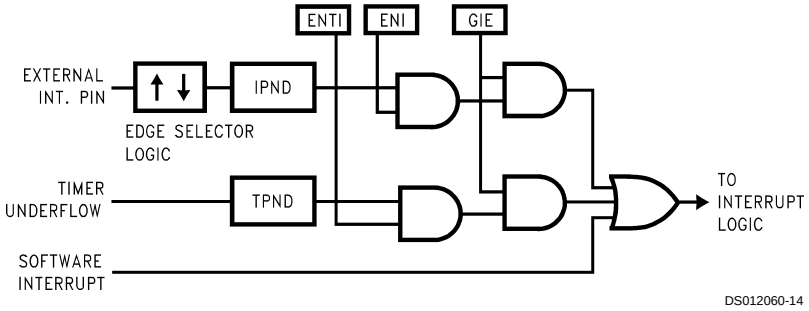


FIGURE 11. Interrupt Block Diagram

DETECTION OF ILLEGAL CONDITIONS

Reading of undefined ROM gets zeroes. The opcode for software interrupt is zero. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signalling that an illegal condition has occurred.

Note: A software interrupt is acted upon only when a timer or external interrupt is not pending as hardware interrupts have priority over software interrupt. In addition, the Global Interrupt bit is not set when a software interrupt is being serviced thereby opening the door for the hardware interrupts to occur. The subroutine stack grows down for each call and grows up for each return. If the stack pointer is initialized to 2F Hex, then if there are more returns than calls, the stack pointer will point to addresses 30 and 31 (which are undefined RAM). Undefined RAM is read as all 1's, thus, the program will return to address FFFF. This is a undefined ROM location and the instruction fetched will generate a software interrupt signalling an illegal condition. The device can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

Illegal conditions may occur from coding errors, "brown out" voltage drops, static, supply noise, etc. When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to RESET but might not clear the RAM). Examination of the stack can help in identifying the source of the error. For example, upon a software interrupt, if the SP = 30, 31 it implies that the stack was over "POP"ed (with the SP=2F hex initially). If the SP contains a legal value (less than or equal to the initialized SP value), then the value in the PC gives a clue as to where in the user program an attempt to access an illegal (an address over 300 Hex) was made. The opcode returned in this case is 00 which is a software interrupt.

The detection of illegal conditions is illustrated with an example:

0043 CLRA

0044 RC

0045 JMP 04FF

0046 NOP

When the device is executing this program, it seemingly "locks-up" having executed a software interrupt. To debug this condition, the user takes a look at the SP and the contents of the stack. The SP has a legal value and the contents of the stack are 04FF. The perceptive user immediately realizes that an illegal ROM location (04FF) was accessed and the opcode returned (00) was a software interrupt. Another way to decode this is to run a trace and follow the sequence of steps that ended in a software interrupt. The damaging jump statement is changed.

Control Registers

CNTRL REGISTER (ADDRESS X'00EE)

The Timer and MICROWIRE control register contains the following bits:

SL1 and SL0 Select the MICROWIRE clock divide-by (00 = 2, 01 = 4, 1x = 8)

IEDG External interrupt edge polarity select

MSEL Selects G5 and G4 as MICROWIRE signals SK and SO respectively

TRUN Used to start and stop the timer/counter (1 = run, 0 = stop)

TC1 Timer Mode Control Bit

TC2 Timer Mode Control Bit

TC3 Timer Mode Control Bit

7							0
TC1	TC2	TC3	TRUN	MSEL	IEDG	SL1	SL0

PSW REGISTER (ADDRESS X'00EF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)

ENI External interrupt enable

BUSY MICROWIRE busy shifting flag

IPND External interrupt pending

ENTI Timer interrupt enable

TPND Timer interrupt pending (timer underflow or capture edge)

C Carry Flip/flop

HC Half carry Flip/flop

7							0
HC	C	TPND	ENTI	IPND	BUSY	ENI	GIE

The Half-Carry bit is also effected by all the instructions that effect the Carry flag. The flag values depend upon the instruction. For example, after executing the ADC instruction the values of the Carry and the Half-Carry flag depend upon the operands involved. However, instructions like SET C and RESET C will set and clear both the carry flags. *Table 5* lists out the instructions that effect the HC and the C flags.

TABLE 5. Instructions Effecting HC and C Flags

Instr.	HC Flag	C Flag
ADC	Depends on Operands	Depends on Operands
SUBC	Depends on Operands	Depends on Operands
SETC	Set	Set
RESET C	Set	Set
RRC	Depends on Operands	Depends on Operands

MEMORY MAP

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

TABLE 6. Memory Map

Address	Contents
00 to 2F	On-chip RAM Bytes (48 Bytes)
30 to 7F	Unused RAM Address Space (Reads as all ones)
80 to BF	Expansion Space for On-Chip EERAM (Reads Undefined Data)
C0 to CF	Expansion Space for I/O and Registers
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (read only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (read only)
D7	Reserved
D8 to DB	Reserved
DC to DF	Reserved
E0 to EF	On-Chip Functions and Registers
E0 to E7	Reserved for Future Parts
E8	Reserved
E9	MICROWIRE Shift Register
EA	Timer Lower Byte
EB	Timer Upper Byte
EC	Timer Autoreload Register Lower Byte
ED	Timer Autoreload Register Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FF	On-Chip RAM Mapped as Registers (16 Bytes)
FC	X Register
FD	SP Register
FE	B Register

Reading other unused memory locations will return undefined data.

Addressing Modes

The device has ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode for the chip. The operand is the data memory addressed by the **B** or **X** pointer.

Register Indirect With Auto Post Increment Or Decrement

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the **B** or **X** pointer. This is a register indirect mode that automatically post increments or post decrements the **B** or **X** pointer after executing the instruction.

Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

Immediate

The instruction contains an 8-bit immediate field as the operand.

Short Immediate

Addressing Modes (Continued)

This addressing mode issued with the LD B,# instruction, where the immediate # is less than 16. The instruction contains a 4-bit immediate field as the operand.

Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

TRANSFER OF CONTROL ADDRESSING MODES

Relative

This mode is used for the JP instruction with the instruction field being added to the program counter to produce the next instruction address. JP has a range from −31 to +32 to allow a one byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "blocks" or "pages" when using JP since all 15 bits of the PC are used.

Absolute

This mode is used with the JMP and JSR instructions with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

Absolute Long

This mode is used with the JMPL and JSRL instructions with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location in the entire 32k program memory space.

Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serves as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Instruction Set

REGISTER AND SYMBOL DEFINITIONS

Registers

A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
SP	8-Bit Stack Pointer Register
S	8-Bit Data Segment Address Register
PC	15-Bit Program Counter Register
PU	Upper 7 Bits of PC
PL	Lower 8 Bits of PC
C	1-Bit of PSW Register for Carry
HC	1-Bit of PSW Register for Half Carry
GIE	1-Bit of PSW Register for Global Interrupt Enable

Symbols

[B]	Memory Indirectly Addressed by B Register
[X]	Memory Indirectly Addressed by X Register
MD	Direct Addressed Memory
Mem	Direct Addressed Memory, or B
MemI	Direct Addressed Memory, B, or Immediate Data
Imm	8-Bit Immediate Data
Reg	Register Memory: Addresses F0 to FF (Includes B, X, and SP)
Bit	Bit Number (0 to 7)
←	Loaded with
↔	Exchanged with

TABLE 7. Instruction Set

Instr		Function	Register Operation
ADD	A, MemI	Add	A←A + MemI
ADC	A, MemI	Add with Carry	A←A + MemI + C, C←Carry
SUBC	A, MemI	Subtract with Carry	A←A − MemI + C, C←Carry
AND	A, MemI	Logical AND	A←A and MemI
OR	A, MemI	Logical OR	A←A or MemI
XOR	A, MemI	Logical Exclusive-OR	A←A xor MemI
IFEQ	A, MemI	IF Equal	Compare A and MemI, Do Next if A = MemI
IFGT	A, MemI	IF Greater than	Compare A and MemI, Do Next if A > MemI
IFBNE	#	IF B not Equal	Do Next If Lower 4 Bits of B not = Imm
DRSZ	Reg	Decrement Reg, Skip if Zero	Reg←Reg - 1, Skip if Reg Goes to Zero
SBIT	#, Mem	Set Bit	1 to Mem.Bit (Bit = 0 to 7 Immediate)
RBIT	#, Mem	Reset Bit	0 to Mem.Bit (Bit = 0 to 7 Immediate)
IFBIT	#, Mem	If Bit	If Mem.Bit is True, Do Next Instruction
X	A, Mem	Exchange A with Memory	A↔Mem
LD	A, MemI	Load A with Memory	A←MemI
LD	Mem, Imm	Load Direct Memory Immed.	Mem←Imm
LD	Reg, Imm	Load Register Memory Immed.	Reg←Imm
X	A, [B±]	Exchange A with Memory [B]	A↔[B] (B←B±1)
X	A, [X±]	Exchange A with Memory [X]	A↔[X] (X←X±1)

Instruction Set (Continued)

TABLE 7. Instruction Set (Continued)

Instr		Function	Register Operation
LD	A, [B±]	Load A with Memory [B]	A←[B] (B←B±1)
LD	A, [X±]	Load A with Memory [X]	A←[X] (X←X±1)
LD	[B±], Imm	Load Memory Immediate	[B]←Imm (B←B±1)
CLRA		Clear A	A←0
INC		Increment A	A←A + 1
DEC		Decrement A	A←A − 1
LAID	A	Load A Indirect from ROM	A←ROM(PU, A)
DCOR	A	Decimal Correct A	A←BCD Correction (follows ADC, SUBC)
RRC		Rotate Right Through Carry	C→A7→…→A0→C
SWAP	A	Swap Nibbles of A	A7…A4↔A3…A0
SC	A	Set C	C←1
RC	A	Reset C	C←0
IFC		If C	If C is True, do Next Instruction
IFNC		If Not C	If C is not True, do Next Instruction
JMPL		Jump Absolute Long	PC←ii (ii = 15 Bits, 0k to 32k)
JMP		Jump Absolute	PC11…PC0←i (i = 12 Bits)
			PC15…PC12 Remain Unchanged
JP		Jump Relative Short	PC←PC + r (r is −31 to +32, not 1)
JSRL	Addr.	Jump Subroutine Long	[SP]←PL, [SP−1]←PU, SP−2, PC←ii
JSR	Addr.	Jump Subroutine	[SP]←PL, [SP−1]←PU, SP−2, PC11..PC0←ii
JID	Disp.	Jump Indirect	PL←ROM(PU, A)
RET	Addr.	Return from Subroutine	SP+2, PL←[SP], PU←[SP−1]
RETSK	Addr.	Return and Skip	SP+2, PL←[SP], PU←[SP−1],
			Skip next Instr.
RETI		Return from Interrupt	SP+2, PL←[SP], PU←[SP−1], GIE←1
INTR		Generate an Interrupt	[SP]←PL, [SP−1]←PU, SP−2, PC←0FF
NOP		No Operation	PC←PC+1

- Most instructions are single byte (with immediate addressing mode instructions requiring two bytes).
- Most single byte instructions take one cycle time to execute.
- Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

The following tables show the number of bytes and cycles for each instruction in the format byte/cycle.

Arithmetic and Logic Instructions (Bytes/Cycles)

Instr	[B]	Direct	Immediate
ADD	1/1	3/4	
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2

Instr	[B]	Direct	Immediate
IFGT	1/1	3/4	2/2
IFBNE	1/1		2/2
DRSZ	1/1	1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

Instructions Using A and C (Bytes/Cycles)

Instr	Bytes/Cycles
CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCOR	1/1
RRCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1

Instruction Set (Continued)

Instructions Using A and C (Bytes/Cycles) (Continued)

Instr	Bytes/Cycles
IFC	1/1
IFNC	1/1

Transfer of Control Instructions (Bytes/Cycles)

Instr	Bytes/Cycles
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

Memory Transfer Instructions (Bytes/Cycles)

Instr	Register Indirect		Direct	Immed.	Register Indirect Auto Incr and Decr	
	[B]	[X]			[B+, B−]	[X+, X−]
X A,[a]	1/1		2/3		1/2	
LD A,*	1/1		2/3		1/2	
LD B,Imm		1/3		2/2		1/3
LD B,Imm		1/3		1/1[b]		1/3
LD Mem,Imm	2/2		3/3	2/3[c]	2/2	
LD Reg,Imm			2/3			

a. Memory location addressed by B or X directly
b. IF B < 16
c. IF B > 15

Instruction Set (Continued)

UPPER NIBBLE BITS 7–4																	LOWER NIBBLE BITS 3–0
F	**E**	**D**	**C**	**B**	**A**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
JP–15	JP–31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADCA, 3I	ADCA, (B)	IFBIT 0, (B)	*	LD B, 0F	IFBNE 0	JSR 0000–00FF	JMP 0000–00FF	JP+17	INTR	0	
JP–14	JP–30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBCA, #i	SUBCA, (B)	IFBIT 1, (B)	*	LD B, 0E	IFBNE 1	JSR 0100–01FF	JMP 0100–01FF	JP+18	JP+2	1	
JP–13	JP–29	LD 0F2, #i	DRSZ 0F2	XA, (X+)	XA, (X+)	IFEQA, #i	IFEQ, #i	IFBIT A, (B)	*	LD B, 0D	IFBNE 2	JSR 0200–02FF	JMP 0200–02FF	JP+19	JP+3	2	
JP–12	JP–28	LD 0F3, #i	DRSZ 0F3	XA, (X–)	XA, (B–)	IFGT A,#i	IFGT A,(B)	IFBIT 3,(B)	*	LD B, 0C	IFBNE 3	JSR 0300–03FF	JMP 0300–03FF	JP+20	JP+4	3	
JP–11	JP–27	LD 0F4, #i	DRSZ 0F4	*	LAID	ADD A,#i	ADD A,(B)	IFBIT 4,(B)	CLRA	LD B, 0B	IFBNE 4	JSR 0400–04FF	JMP 0400–04FF	JP+21	JP+5	4	
JP–10	JP–26	LD 0F5, #i	DRSZ 0F5	*	JID	AND A,#i	AND A,(B)	IFBIT 5,(B)	SWAPA	LD B, 0A	IFBNE 5	JSR 0500–05FF	JMP 0500–05FF	JP+22	JP+6	5	
JP–9	JP–25	LD 0F6, #i	DRSZ 0F6	X A,(X)	X A,(B)	XOR A,#i	XOR A,(B)	IFBIT 6,(B)	DCORA	LD B, 9	IFBNE 6	JSR 0600–06FF	JMP 0600–06FF	JP+23	JP+7	6	
JP–8	JP–24	LD 0F7, #i	DRSZ 0F7	*	*	OR A,#i	OR A,(B)	IFBIT 7,(B)	*	LD B, 8	IFBNE 7	JSR 0700–07FF	JMP 0700–07FF	JP+24	JP+8	7	
JP–7	JP–23	LD 0F8, #i	DRSZ 0F8	NOP	*	LD A,#i	IFC	SBIT 0,(B)	RBIT 0,(B)	LD B, 7	IFBNE 8	JSR 0800–08FF	JMP 0800–08FF	JP+25	JP+9	8	
JP–6	JP–22	LD 0F9, #i	DRSZ 0F9	*	*	*	IFNC	SBIT 1,(B)	RBIT 1,(B)	LD B, 6	IFBNE 9	JSR 0900–09FF	JMP 0900–09FF	JP+26	JP+10	9	
JP–5	JP–21	LD 0FA, #i	DRSZ 0FA	LD A,(X+)	LD A,(B+)	LD (B+),#i	INCA	SBIT 2,(B)	RBIT 2,(B)	LD B, 5	IFBNE 0A	JSR 0A00–0AFF	JMP 0A00–0AFF	JP+27	JP+11	A	
JP–4	JP–20	LD 0FB, #i	DRSZ 0FB	LD A,(X–)	LD A,(B–)	LD (B–),#i	DECA	SBIT 3,(B)	RBIT 3,(B)	LD B, 4	IFBNE 0B	JSR 0B00–0BFF	JMP 0B00–0BFF	JP+28	JP+12	B	
JP–3	JP–19	LD 0FC, #i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	*	SBIT 4,(B)	RBIT 4,(B)	LD B, 3	IFBNE 0C	JSR 0C00–0CFF	JMP 0C00–0CFF	JP+29	JP+13	C	
JP–2	JP–18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	SBIT 5,(B)	RBIT 5,(B)	LD B, 2	IFBNE 0D	JSR 0D00–0DFF	JMP 0D00–0DFF	JP+30	JP+14	D	
JP–1	JP–17	LD 0FE, #i	DRSZ 0FE	LD A,(X)	LD A,(B)	LD B,#i	RET	SBIT 6,(B)	RBIT 6,(B)	LD B, 1	IFBNE 0E	JSR 0E00–0EFF	JMP 0E00–0EFF	JP+31	JP+15	E	
JP–0	JP–16	LD 0FF, #i	DRSZ 0FF	*	*	*	RETI	SBIT 7,(B)	RBIT 7,(B)	LD B, 0	IFBNE 0F	JSR 0F00–0FFF	JMP 0F00–0FFF	JP+32	JP+16	F	

Option List

The mask programmable options are listed out below. The options are programmed at the same time as the ROM pattern to provide the user with hardware flexibility to use a variety of oscillator configuration.

OPTION 1: CKI INPUT

= 1 Crystal (CKI/10) CKO for crystal configuration

= 2 NA

= 3 R/C (CKI/10) CKO available as G7 input

OPTION 2: BONDING

= 1 NA

= 2 NA

= 3 20 pin DIP package

= 4 20 pin SO package

= 5 NA

The following option information is to be sent to National along with the EPROM.

Option Data

Option 1 Value__is: CKI Input

Option 2 Value__is: COP Bonding

COP8 Tools Overview

National is engaged with an international community of independent 3rd party vendors who provide hardware and software development tool support. Through National's interaction and guidance, these tools cooperate to form a choice of tools that fits each developer's needs.

This section provides a summary of the tool and development kits currently available. Up-to-date information, selection guides, free tools, demos, updates, and purchase information can be obtained at our web site at: www.national.com/cop8.

SUMMARY OF TOOLS

COP8 Evaluation Software and Reference Designs

- **COP8–NSEVAL:** Software Evaluation package for Windows. A fully integrated evaluation environment for COP8. Includes WCOP8 IDE evaluation version (Integrated Development Environment), COP8-NSASM (Full COP8 Assembler), COP8-MLSIM (COP8 Instruction Level Simulator), COP8C Compiler Demo, DriveWay™ COP8 Device-Driver-Builder Demo, Manuals, Applications Software, and other COP8 technical information.
- **COP8–REF-xx:** Reference Designs for COP8 Families. Realtime hardware environment with a variety of functions for demonstrating the various capabilities and features of specific COP8 device families. Run Win 95 demo reference software and exercise specific device capabilities.

  Includes PCB with pre-programmed COP8, 9v battery for stand-alone operation, assembly listing, full applications source code, BOM, and schematics.

  (Add COP8-NSEVAL and an OTP programmer to implement your own software ideas in Assembly Code.)

COP8 Starter Kits and Hardware Target Solutions

- **COP8-EVAL-xxx:** A variety of Multifunction Evaluation, Design Test, and Target Boards for COP8 Families. Realtime target design environments with a selection of peripherals and features including multi I/O, LCD display, keyboard, A/D, D/A, EEPROM, USART, LEDs, and bread-board area. Quickly design, test, and implement a custom target system (some target boards are standalone, and ready for mounting into a standard enclosure), or just evaluate and test your code. Includes COP8-NSDEV with IDE and Assembler, software routines, reference designs, and source code (no p/s).

COP8 Software Development Languages and Integrated Environments

- **COP8-NSDEV:** National's COP8 Software Development package for Windows on CD. A fully Integrated Development Environment for COP8. Includes a fully licensed WCOP8 IDE, COP8-NSASM. Plus Manuals, Applications Software, and other COP8 technical information.
- **COP8C:** ByteCraft - C Cross-Compiler and Code Development System. Includes BCLIDE (Integrated Development Environment) for Win32, editor, optimizing C Cross-Compiler, macro cross assembler, BC-Linker, and MetaLinktools support. (DOS/SUN versions available; Compiler is linkable under WCOP8 IDE; Compatible with DriveWay COP8)
- **EWCOP8, EWCOP8-M, EWCOP8-BL:** IAR - ANSI C-Compiler and Embedded Workbench. (M version includes MetaLink debugger support) (BL version: 4k code limit; no FP). A fully integrated Win32 IDE, ANSI C-Compiler, macro assembler, editor, linker, librarian, and C-Spy high-level simulator/debugger.

COP8 Development Productivity Tools

- **DriveWay-COP8:** Aisys Corporation - COP8 Peripherals Code Generation tool. Automatically generates tested and documented C or Assembly source code modules containing I/O drivers and interrupt handlers for each on-chip peripheral. Application specific code can be inserted for customization using the integrated editor. (Compatible with COP8-NSASM, COP8C, and WCOP8 IDE.)
- **COP8-UTILS:** COP8 assembly code examples, device drivers, and utilities to speed up code development. (Included with COP8-NSDEV and COP8-NSEVAL.)
- **WCOP8 IDE:** KKD - COP8 IDE (Integrated Development Environment). Supports COP8C, COP8-NSASM, COP8-MLSIM, DriveWay COP8, and MetaLink debugger under a common Windows Project Management environment. Code development, debug, and emulation tools can be launched from a single project window framework. (Included in COP8-NSDEV and COP8-NSEVAL.)

COP8 Hardware Debug Tools

- **COP8xx-DM:** Metalink COP8 Debug Module for non-flash COP8 Families. Windows based development and real-time in-circuit emulation tool, with 100 frame trace, 32k s/w breaks, Enhanced User Interface, MetaLinkDebugger, and COP8 OTP Programmer with sockets. Includes COP8-NSDEV, power supply, DIP and/or SMD emulation cables and adapters.

COP8 Tools Overview (Continued)

- **IM-COP8:** MetaLink iceMASTER® for non-flash COP8 devices. Windows based, full featured real-time in-circuit emulator, with 4k trace, 32k s/w breaks, and MetaLink-Windows Debugger. Includes COP8-NSDEV and power supply. Package-specific probes and surface mount adaptors are ordered separately. (Add COP8-PM and adapters for OTP programming.)

COP8 Development and OTP Programming Tools

- **COP8-PM:** COP8 Development Programming Module. Windows programming tool for COP8 OTP Families. Includes 40 DIP programming socket, control software, RS232 cable, and power supply. (SMD and 87Lxx programming adapters are extra.)
- **Development:** Metalink's Debug Module includes development device programming capability for COP8 devices. Many other third-party programmers are approved for development and engineering use.
- **Production:** Third-party programmers and automatic handling equipment cover needs from engineering prototype and pilot production, to full production environments.
- **Factory Programming:** Factory programming available for high-volume requirements.

WHERE TO GET TOOLS

Tools are ordered directly from the following vendors. Please go to the vendor's web site for current listings of distributors.

Vendor	Home Office	Electronic Sites	Other Main Offices
Aisys	U.S.A.: Santa Clara, CA 1-408-327-8820 fax: 1-408-327-8830	www.aisysinc.com info@aisysinc.com	Distributors
Byte Craft	U.S.A. 1-519-888-6911 fax: 1-519-746-6751	www.bytecraft.com info@bytecraft.com	Distributors
IAR	Sweden: Uppsala +46 18 16 78 00 fax: +46 18 16 78 38	www.iar.se info@iar.se info@iar.com info@iarsys.co.uk info@iar.de	U.S.A.: San Francisco 1-415-765-5500 fax: 1-415-765-5503 U.K.: London +44 171 924 33 34 fax: +44 171 924 53 41 Germany: Munich +49 89 470 6022 fax: +49 89 470 956
ICU	Sweden: Polygonvaegen +46 8 630 11 20 fax: +46 8 630 11 70	www.icu.se support@icu.se support@icu.ch	Switzeland: Hoehe +41 34 497 28 20 fax: +41 34 497 28 21
KKD	Denmark:	www.kkd.dk	
MetaLink	U.S.A.: Chandler, AZ 1-800-638-2423 fax: 1-602-926-1198	www.metaice.com sales@metaice.com support@metaice.com bbs: 1-602-962-0013 www.metalink.de	Germany: Kirchseeon 80-91-5696-0 fax: 80-91-2386 islanger@metalink.de Distributors Worldwide
National	U.S.A.: Santa Clara, CA 1-800-272-9959 fax: 1-800-737-7018	www.national.com/cop8 support@nsc.com europe.support@nsc.com	Europe: +49 (0) 180 530 8585 fax: +49 (0) 180 530 8586 Distributors Worldwide

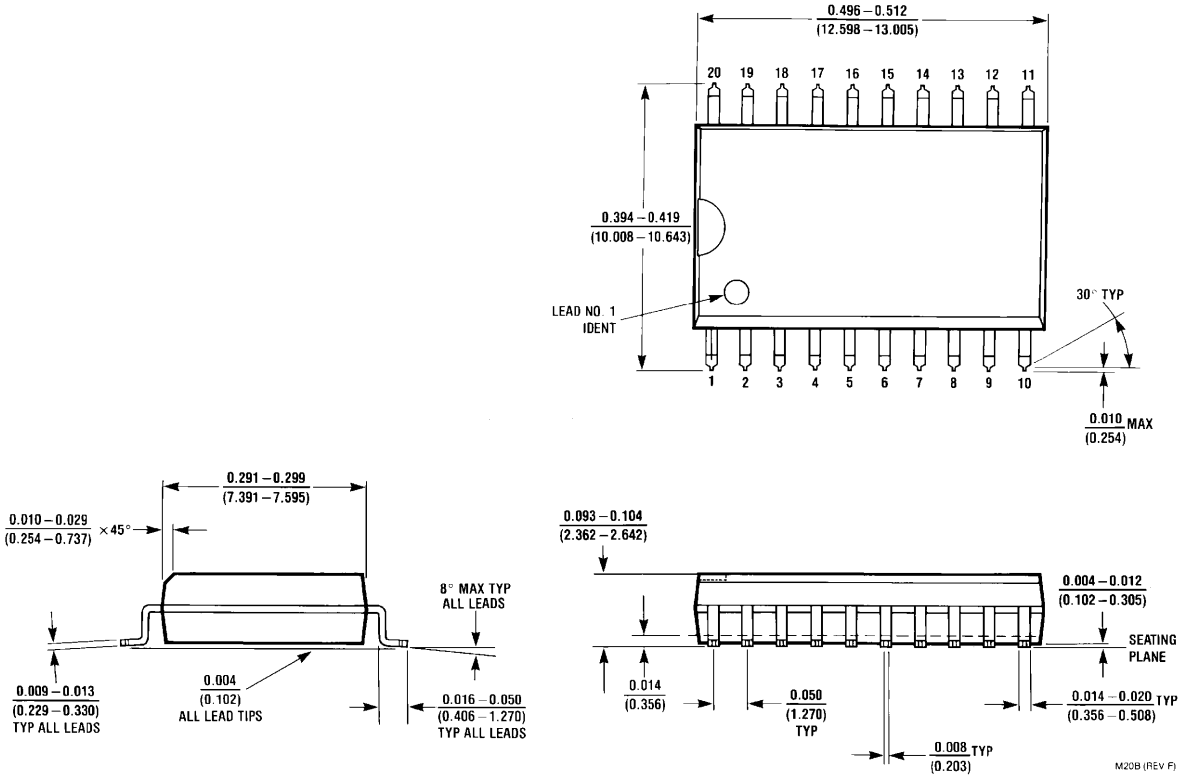
The following companies have approved COP8 programmers in a variety of configurations. Contact your local office or distributor. You can link to their web sites and get the latest listing of approved programmers from National's COP8 OTP Support page at: www.national.com/cop8.

Advantech; Dataman; EE tools; Minato; BP Microsystems; Data I/O; Hi-Lo Systems; ICE Technology; Lloyd Research; Logical Devices; MQP; Needhams; Phyton; SMS; Stag Programmers; System General; Tribal Microsystems; Xeltek.

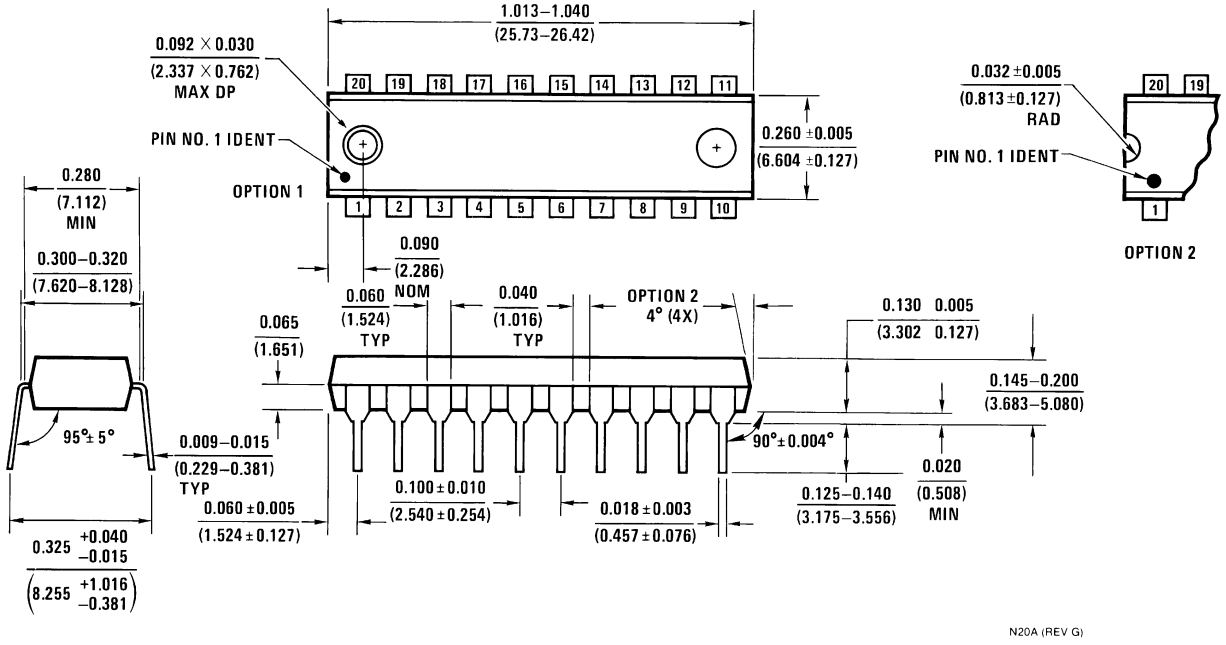
CUSTOMER SUPPORT

Complete product information and technical support is available from National's customer response centers, and from our on-line COP8 customer support sites.

Physical Dimensions inches (millimeters) unless otherwise noted



20-Lead Molded Small Outline Package (M)
Order Number COP912C-XXX/WM, COP912CH-XXX/WM
NS Package Number M20B



20-Lead Molded Dual-In-Line Package (N)
Order Number COP912C-XXX/N, COP912CH-XXX/N
NS Package Number N20A

Notes

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
Americas
Tel: 1-800-272-9959
Fax: 1-800-737-7018
Email: support@nsc.com
www.national.com

National Semiconductor Europe
Fax: +49 (0) 180-530 85 86
Email: europe.support@nsc.com
Deutsch Tel: +49 (0) 69 9508 6208
English Tel: +44 (0) 870 24 0 2171
Français Tel: +33 (0) 1 41 91 8790

National Semiconductor Asia Pacific Customer Response Group
Tel: 65-2544466
Fax: 65-2504466
Email: ap.support@nsc.com

National Semiconductor Japan Ltd.
Tel: 81-3-5639-7560
Fax: 81-3-5639-7507

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and National reserves the right at any time without notice to change said circuitry and specifications.