National Semiconductor

January 1998

COP888GD/COP988GD 8-Bit Microcontroller with A/D Converter

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M^{2}CMOS™ process technology. The COP888GD is a member of this expandable 8-bit core processor family of microcontrollers. (continued)

Key Features

- 8-channel A/D converter with prescaler and both differential and single ended modes
- Three 16-bit timers, each with two 16-bit registers supporting
  - — Processor Independent PWM mode
  - — External Event counter mode
  - — Input Capture mode
- Quiet design (low radiated emissions)
- 16k bytes on-board ROM
- 256 bytes on-board RAM

Additional Peripheral Features

- Idle Timer
- Multi-Input Wake-Up (MIWU) with optional interrupts (8)
- WATCHDOG and clock monitor logic
- MICROWIRE/PLUS™ serial I/O

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 ports G and L
- Package: 44 PLCC with 40 I/O Pins

CPU/Instruction Set Features

- 1 μs instruction cycle time
- Twelve multi-source vectored interrupts servicing
  - — External Interrupt
  - — Idle Timer T0
  - — Three Timers (Each with 2 Interrupts)
  - — MICROWIRE/PLUS
  - — Multi-Input Wake Up
  - — Software Trap
  - — Default VIS (default interrupt)
- Versatile and easy to use instruction set
- 8-bit Stack Pointer (SP) - stack in RAM
- Two 8-bit Register Indirect Data Memory Pointers (B and X)

Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Low current drain (typically < 1 μA)
- Single supply operation: 2.5V to 5.5V
- Temperature ranges:0˚C to + 70˚C, –40˚C to +85˚C

Development Support

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

Block Diagram

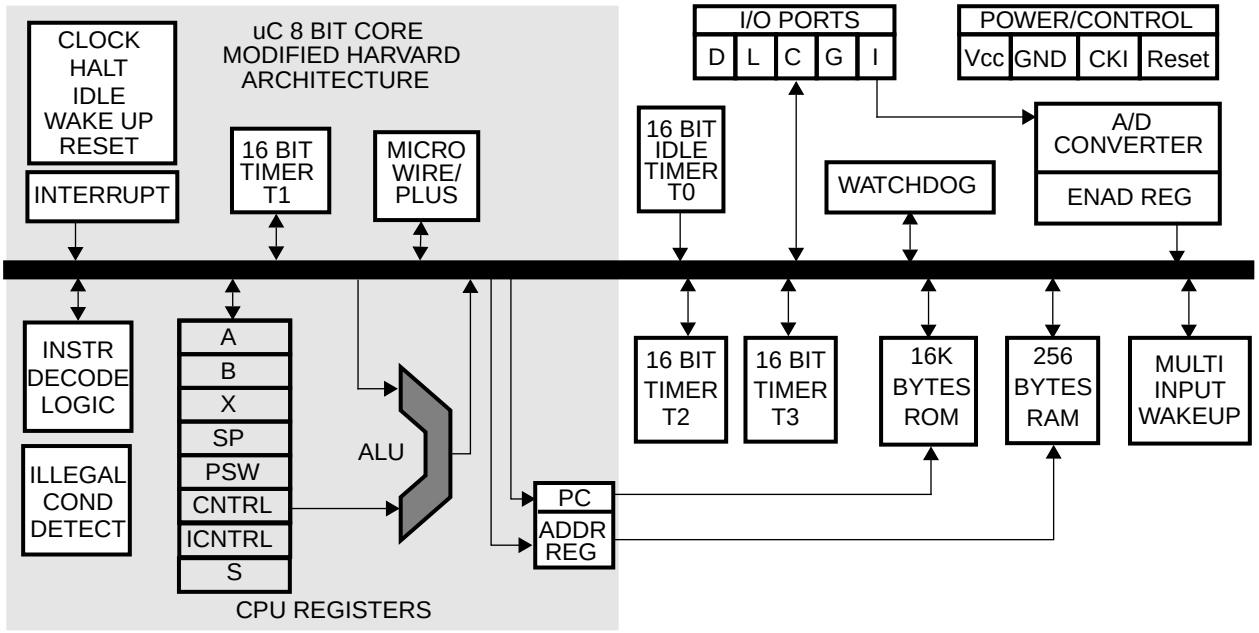


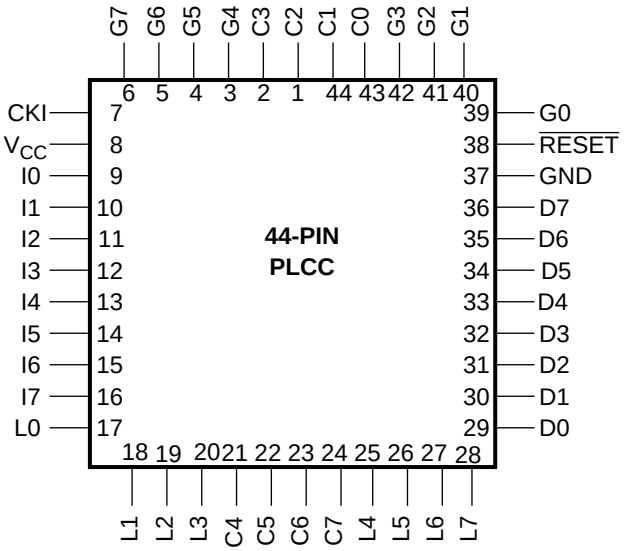
Figure 1. COP888GD Block Diagram

General Description (Continued)

It is a fully static part, fabricated using double-metal silicon gate microCMOS technology. Features include an 8-bit memory mapped architecture, MICROWIRE serial I/O, three 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), 8 channel analog to digital convertor, and two power saving modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced interrupt capability may also be used independent of the HALT or IDLE modes. The device includes an IDLE Timer with 5 selectable interrupt periods which can be used to wake the device from the IDLE mode. Each I/O pin has software selectable configurations. The devices operate over a voltage range of 2.5V to 5.5V. High throughput is achieved with an efficient, regular instruction set operating at a maximum rate of 1 µs per instruction. Low radiated emissions are achieved by gradual turn-on output drivers and internal I_{CC} smoothing filters on the chip logic and crystal oscillator.

Connection Diagrams

Plastic Lead Chip Carrier



Top View

Order Number COP888GD-XXX/V, or COP988GD-XXX/V,
See NS Package Number V44A

Figure 2. Connection Diagrams

Connection Diagrams (Continued)

Pinouts for 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	44-Pin PLCC
L0	I/O	MIWU		17
L1	I/O	MIWU		18
L2	I/O	MIWU		19
L3	I/O	MIWU		20
L4	I/O	MIWU	T2A	25
L5	I/O	MIWU	T2B	26
L6	I/O	MIWU	T3A	27
L7	I/O	MIWU	T3B	28
G0	I/O	INT		39
G1	WDOUT			40
G2	I/O	T1B		41
G3	I/O	T1A		42
G4	I/O	SO		3
G5	I/O	SK		4
G6	I	SI		5
G7	I/CKO	HALT Restart		6
D0	O			29
D1	O			30
D2	O			31
D3	O			32
D4	O			33
D5	O			34
D6	O			35
D7	O			36
I0	I	ACH0		9
I1	I	ACH1		10
I2	I	ACH2		11
I3	I	ACH3		12
I4	I	ACH4		13
I5	I	ACH5		14
I6	I	ACH6		15
I7	I	ACH7		16
C0	I/O			43
C1	I/O			44
C2	I/O			1
C3	I/O			2
C4	I/O			21
C5	I/O			22
C6	I/O			23
C7	I/O			24
V_{CC}				8
GND				37
CKI				7
$\overline{\text{RESET}}$				38

Absolute Maximum Ratings

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

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	–0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	–65˚C to +140˚C

Note: *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 **COP988GD:** $0^{\circ}C \le T_A \le +70^{\circ}C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage					
COP988GD		2.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 µs			16	mA
CKI = 4 MHz	V_{CC} = 4.5V, t_c = 2.5µs			6	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<5	8	µA
	V_{CC} = 4.5V, CKI = 0 MHz		<3	5	µA
IDLE Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, tc = 1µs			1.2	mA
CKI = 4 MHz	V_{CC} = 5.5V, tc = 2.5 µs			1.0	mA
Input Levels					
RESET, CKI					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs (L0-L7, G0-G6, C0-C7, I0-I7)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	–1		+1	µA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	–40		–250	µA
G and L Port Input Hysteresis (Note 6)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	–0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	–0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	–10		–100	µA
	V_{CC} = 2.5V, V_{OH} = 1.8V	–2.5		–33	µA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	–0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	–0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	–1		+1	µA
Allowable Sink/Source					
Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current	Room Temp.			±100	mA
without Latchup (Note 4,6)					
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics **COP988GD:** $0^{\circ}C \le T_A \le +70^{\circ}C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal, Resonator	$4.5V \le V_{CC} \le 5.5V$	1		DC	µs
	$2.5V \le V_{CC} < 4.5V$	2.5		DC	µs
R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	3		DC	µs
	$2.5V \le V_{CC} < 4.5V$	7.5		DC	µs
CKI Clock Duty Cycle (Note 6)	f_r = Max	40		60	%
Rise Time (Note 6)	f_r = 10 MHz Ext Clock			5	ns
Fall Time (Note 6)	f_r = 10 MHz Ext Clock			5	ns
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
	$2.5V \le V_{CC} < 4.5V$	500			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
	$2.5V \le V_{CC} < 4.5V$	150			ns
Output Propagation Delay	R_L = 2.2k, C_L = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	µs
	$2.5V \le V_{CC} < 4.5V$			1.75	µs
All Others	$4.5V \le V_{CC} \le 5.5V$			1	µs
	$2.5V \le V_{CC} < 4.5V$			2.5	µs
MICROWIRE™ Setup Time (t_{UWS}) (Note 5)		20			ns
MICROWIRE Hold Time (t_{UWH}) (Note 5)		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width (Note 6)					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			µs

t_c = Instruction cycle time

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs programmed low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L, C, G0, and G2-G5 programmed as low outputs and not driving a load; all inputs tied to V_{CC}; clock monitor disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14 Volts. **WARNING: Voltages in excess of 14 volts will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.

Absolute Maximum Ratings

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

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	–0.3V to V_{CC} +0.3V
Total Current into V_{CC} Pin (Source)	100 mA
Total Current out of GND Pin (Sink)	110 mA
Storage Temperature Range	–65˚C to +140˚C

Note: *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 **COP888GD:** –40˚C ≤ T_A ≤ +85˚C unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		5.5	V
Power Supply Ripple (Note 1)	Peak-to-Peak			0.1 V_{CC}	V
Supply Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 µs			16	mA
CKI = 4 MHz	V_{CC} = 4.5V, t_c = 2.5 µs			6	mA
HALT Current (Note 3)	V_{CC} = 5.5V, CKI = 0 MHz		<3	10	µA
	V_{CC} = 4.5V, CKI = 0 MHz			7.5	µA
IDLE Current (Note 2)					
CKI = 10 MHz	V_{CC} = 5.5V, t_c = 1 µs			1.2	mA
CKI = 4 MHz	V_{CC} = 5.5V, t_c = 2.5 µs			1.0	mA
Input Levels					
$\overline{\text{RESET}}$, CKI					
Logic High		0.8 V_{CC}			V
Logic Low				0.2 V_{CC}	V
All Other Inputs (L0-L7, G0-G6, C0-C7, I0-I7)					
Logic High		0.7 V_{CC}			V
Logic Low				0.2 V_{CC}	V
Hi-Z Input Leakage	V_{CC} = 5.5V	–2		+2	µA
Input Pullup Current	V_{CC} = 5.5V, V_{IN} = 0V	–40		–250	µA
G and L Port Input Hysteresis (Note 6)				0.35 V_{CC}	V
Output Current Levels					
D Outputs					
Source	V_{CC} = 4.5V, V_{OH} = 3.3V	–0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	–0.2			mA
Sink	V_{CC} = 4.5V, V_{OL} = 1.0V	10			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	V_{CC} = 4.5V, V_{OH} = 2.7V	–10		–100	µA
	V_{CC} = 2.5V, V_{OH} = 1.8V	–2.5		–33	µA
Source (Push-Pull Mode)	V_{CC} = 4.5V, V_{OH} = 3.3V	–0.4			mA
	V_{CC} = 2.5V, V_{OH} = 1.8V	–0.2			mA
Sink (Push-Pull Mode)	V_{CC} = 4.5V, V_{OL} = 0.4V	1.6			mA
	V_{CC} = 2.5V, V_{OL} = 0.4V	0.7			mA
TRI-STATE Leakage	V_{CC} = 5.5V	–2		+2	µA
Allowable Sink/Source Current per Pin					
D Outputs (Sink)				15	mA
All others				3	mA
Maximum Input Current without Latchup (Note 4, 6)	Room Temp.			±100	mA
RAM Retention Voltage, Vr	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics **COP888GD:** $-40^{\circ}C \le T_A \le +85^{\circ}C$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (tc)					
Crystal, Resonator	$4.5V \le V_{CC} \le 5.5V$	1		DC	µs
	$2.5V \le V_{CC} < 4.5V$	2.5		DC	µs
R/C Oscillator	$4.5V \le V_{CC} \le 5.5V$	3		DC	µs
	$2.5V \le V_{CC} < 4.5V$	7.5		DC	µs
CKI Clock Duty Cycle (Note 6)	f_r = Max	40		60	%
Rise Time (Note 6)	f_r = 10 MHz Ext Clock			5	ns
Fall Time (Note 6)	f_r = 10 MHz Ext Clock			5	ns
Inputs					
t_{SETUP}	$4.5V \le V_{CC} \le 5.5V$	200			ns
	$2.5V \le V_{CC} < 4.5V$	500			ns
t_{HOLD}	$4.5V \le V_{CC} \le 5.5V$	60			ns
	$2.5V \le V_{CC} < 4.5V$	150			ns
Output Propagation Delay	R_L = 2.2k, C_L = 100 pF				
t_{PD1}, t_{PD0}					
SO, SK	$4.5V \le V_{CC} \le 5.5V$			0.7	µs
	$2.5V \le V_{CC} < 4.5V$			1.75	µs
All Others	$4.5V \le V_{CC} \le 5.5V$			1	µs
	$2.5V \le V_{CC} < 4.5V$			2.5	µs
MICROWIRE™ Setup Time (t_{UWS})		20			ns
MICROWIRE Hold Time (t_{UWH})		56			ns
MICROWIRE Output Propagation Delay (t_{UPD})				220	ns
Input Pulse Width					
Interrupt Input High Time		1			t_c
Interrupt Input Low Time		1			t_c
Timer Input High Time		1			t_c
Timer Input Low Time		1			t_c
Reset Pulse Width		1			µs

t_c = Instruction cycle time

Note 1: Maximum rate of voltage change must be < 0.5 V/ms.

Note 2: Supply and IDLE currents are measured with CKI driven with a square wave Oscillator, CKO driven 180° out of phase with CKI, inputs connected to V_{CC} and outputs programmed low but not connected to a load.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I_{DD} HALT is done with device neither sourcing nor sinking current; with L, C, G0, and G2-G5 programmed as low outputs and not driving a load; all inputs tied to V_{CC}; clock monitor disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

Note 4: Pins G6 and $\overline{\text{RESET}}$ are designed with a high voltage input network. These pins allow input voltages > V_{CC} and the pins will have sink current to V_{CC} when biased at voltages > V_{CC} (the pins do not have source current when biased at a voltage below V_{CC}). The effective resistance to V_{CC} is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to < 14 Volts. **WARNING: Voltages in excess of 14 volts will cause damage to the pins. This warning excludes ESD transients.**

Note 5: The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

Note 6: Parameter characterized but not tested.

A/D Converter Specifications V_{CC} = 5V ±10%, (V_{SS} – 0.050V) ≤ Any Input ≤ (V_{CC} + 0.050V)

Parameter	Conditions	Min	Typ	Max	Units
Resolution				8	Bits
Absolute Accuracy				1	LSB
Non-Linearity	Deviation from the Best Straight Line			±1	LSB
Differential Non-Linearity				±1	LSB
Common Mode Input Range (Note 9)		GND		V_{CC}	V
DC Common Mode Error				±1/2	LSB
Off Channel Leakage Current			1	2	µA
On Channel Leakage Current			1	2	µA
A/D Clock Frequency (Note 8)		0.1		1.67	MHz
Conversion Time (Note 7)			17		A/D Clock Cycles
Internal Reference Resistance Turn-on Time (Note 10)				1	µs

Note 7: Conversion Time includes 7 A/D clock cycle sample and hold time.
Note 8: See Prescaler description.
Note 9: For $V_{IN}(-)>=V_{IN}(+)$ the digital output code will be 0000 0000. Two on-chip diodes are tied to each analog input. The diodes will forward conduct for analog input voltages below ground or above the V_{CC} supply. Be careful, during testing at low V_{CC} levels (4.5V), as high level analog inputs (5V) can cause this input diode to conduct—especially at elevated temperatures, and cause errors for analog inputs near full-scale. The spec allows 50 mV forward bias of either diode. This means that as long as the analog V_{IN} does not exceed the supply voltage by more than 50 mV, the output code will be correct. To achieve an absolute 0 V_{DC} to 5 V_{DC} input voltage range will therefore require a minimum supply voltage of 4.950 V_{DC} over temperature variations, initial tolerance and loading.
Note 10: Time for internal reference resistance to turn on and settle after coming out of HALT or IDLE mode.

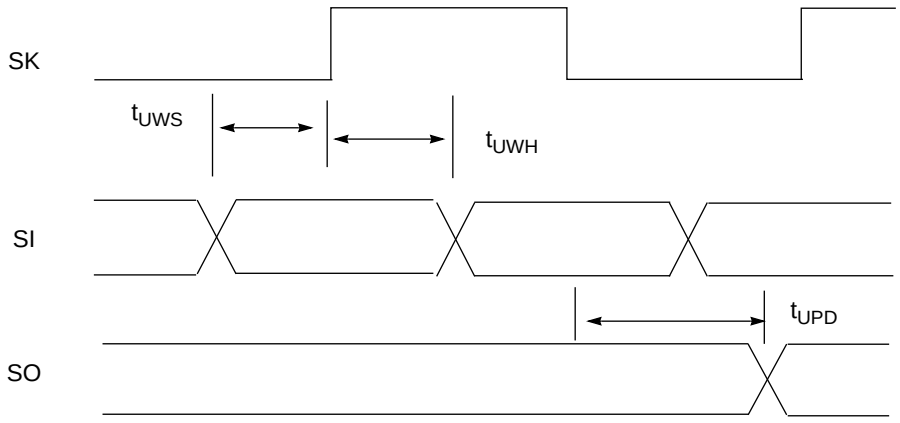


Figure 3. MICROWIRE/PLUS Timing

Pin Descriptions

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

V_{CC} and GND are the reference voltage pins for the on-board A/D converter.

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

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

The device contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input (Schmitt trigger inputs on ports G and L), output or TRI-STATE under program control. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) Figure 1 shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

CONFIGURATION Register	DATA Register	Port Set-Up
0	0	Hi-Z Input (TRI-STATE) Output
0	1	Input with Weak Pull-Up
1	0	Push-Pull Zero Output
1	1	Push-Pull One Output

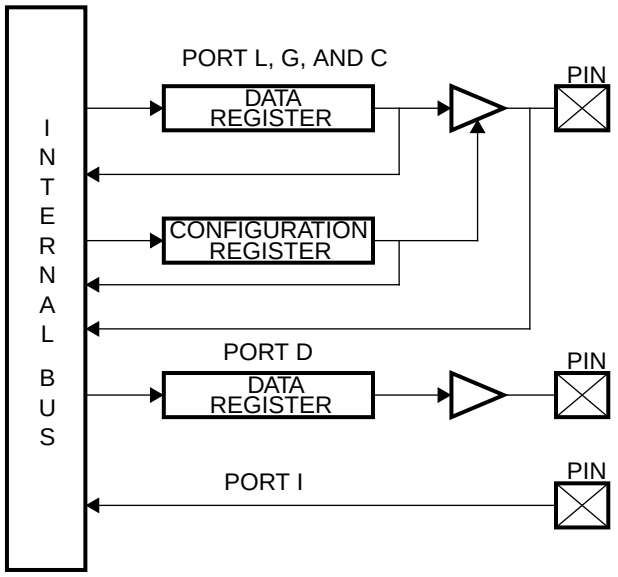


Figure 1. I/O Port Configurations

PORT L is an 8-bit I/O port. All L-pins have Schmitt triggers on the inputs.

Port L supports Multi-Input Wakeup (MIWU) on all eight pins. L4, L5, L6 and L7 are used for the timer input functions T2A, T2B, T3A and T3B.

Port L has the following alternate features:

L0 MIWU
L1 MIWU
L2 MIWU
L3 MIWU
L4 MIWU or T2A
L5 MIWU or T2B
L6 MIWU or T3A
L7 MIWU or T3B

Port G is an 8-bit port with 5 I/O pins (G0, G2--G5), an input pin (G6), and two dedicated output pins (G1 and G7). Pins G0 and G2--G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin, but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin or general purpose input (R/C clock configuration), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a “1” to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a “1” to bit 6 of the Port G Data Register.

Writing a “1” to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

	Config Reg.	Data Reg.
G7	CLKDLY	HALT
G6	Alternate SK	IDLE

Port G has the following alternate features:

G0 INTR (External Interrupt Input)
G2 T1B (Timer T1 Capture Input)
G3 T1A (Timer T1 I/O)
G4 SO (MICROWIRE Serial Data Output)
G5 SK (MICROWIRE Serial Clock)
G6 SI (MICROWIRE Serial Data Input)

Port G has the following dedicated functions:

G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output
G7 CKO Oscillator dedicated output or general purpose input

Port C is an 8-bit I/O port.

Port I is an 8-bit Hi-Z input port, and also provides the analog inputs to the A/D converter. If unterminated, Port I pins will draw power only when addressed.

Port D is an 8-bit output port that is preset high when RESET goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

Note: Care must be exercised with the D2 pin operation. At $\overline{\text{RESET}}$, the external loads on this pin must ensure that the output voltages stay above 0.9 V_{CC} to prevent the chip from entering special modes. Also, keep the external loading on D2 to less than 1000 pF.

Functional Description

The architecture of the device is modified Harvard architecture. With the Harvard architecture, the program memory (ROM) is separated from the data memory (RAM). Both ROM and RAM have their own separate address space with separate address buses. The architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can perform an 8-bit addition, subtraction, logical or shift operation in one instruction (tc) 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 an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

PROGRAM MEMORY

The program memory consists of 16,384 bytes of ROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the devices vector to program memory location 0FF Hex.

DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, SP pointers and S register.

The data memory consists of 256 bytes of RAM. Sixteen bytes of RAM are mapped as “registers” at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, SP, B and S are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All 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. The accumulator (A) bits can also be directly and individually tested.

Note: RAM contents are undefined upon power-up.

Data Memory Segment RAM Extension

Data memory address 0FF is used as a memory mapped location for the Data Segment Address Register (S).

The data store memory is either addressed directly by a single byte address within the instruction, or indirectly relative to the reference of the B, X, or SP pointers (each contains a single-byte address). This single-byte address allows an addressing range of 256 locations from 00 to FF hex. The upper bit of this single-byte address divides the data store memory into two separate sections as outlined previously. With the exception of the RAM register memory from address locations 00F0 to 00FF, all RAM memory is memory mapped with the upper bit of the single-byte address being equal to zero. This allows the upper bit of the single-byte address to determine whether or not the base address range (from 0000 to 00FF) is extended. If this upper bit equals one (representing address range 0080 to 00FF), then address extension does not take place. Alternatively, if this upper bit equals zero, then the data segment extension register S is used to extend the base address range (from 0000 to 007F) from XX00 to XX7F, where XX represents the 8 bits from the S register. Thus the 128-byte data segment extensions are located from addresses 0100 to 017F for data segment 1, 0200 to 027F for data segment 2, etc., up to FF00 to FF7F for data segment 255. The base address range from 0000 to 007F represents data segment 0.

Figure 2 illustrates how the S register data memory extension is used in extending the lower half of the base address range (00 to 7F hex) into 256 data segments of 128 bytes each, with a total addressing range of 32 kbytes from XX00 to XX7F. This organization allows a total of 256 data segments of 128 bytes each with an additional upper base segment of 128 bytes. Furthermore, all addressing modes are available for all data segments. The S register must be changed under program control to move from one data segment (128 bytes) to another. However, the upper base segment (containing the 16 memory registers, I/O registers, control registers, etc.) is always available regardless of the contents of the S register, since the upper base segment (address range 0080 to 00FF) is independent of data segment extension.

The instructions that utilize the stack pointer (SP) always reference the stack as part of the base segment (Segment 0), regardless of the contents of the S register. The S register is not changed by these instructions. Consequently, the stack

(used with subroutine linkage and interrupts) is always located in the base segment. The stack pointer will be initialized to point at data memory location 006F as a result of reset.

The 128 bytes of RAM contained in the base segment are split between the lower and upper base segments. The first 112 bytes of RAM are resident from address 0000 to 006F in the lower base segment, while the remaining 16 bytes of RAM represent the 16 data memory registers located at addresses 00F0 to 00FF of the upper base segment. No RAM is located at the upper sixteen addresses (0070 to 007F) of the lower base segment.

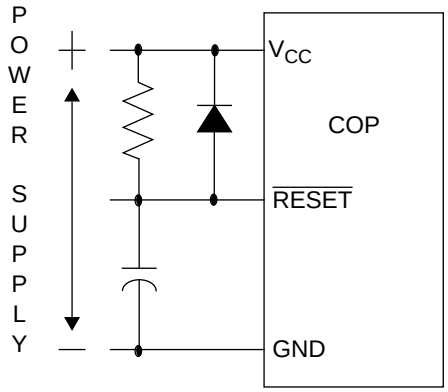
Additional RAM beyond these initial 128 bytes, however, will always be memory mapped in groups of 128 bytes (or less) at the data segment address extensions (XX00 to XX7F) of the lower base segment. The additional 128 bytes of RAM are memory mapped at address locations 0100 to 017F hex.

Reset

The $\overline{\text{RESET}}$ input when pulled low initializes the microcontroller. Initialization will occur whenever the $\overline{\text{RESET}}$ input is pulled low. Upon initialization, the data and configuration registers for Ports L, G, and C are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is initialized high with $\overline{\text{RESET}}$. The PC, PSW, CNTRL, ITMR, ENAD, ICNTRL, T2CNTRL and T3CNTRL control registers are cleared. The Multi-Input Wakeup registers WKEN, WKEDG, and WKPND are cleared. The Stack Pointer, SP, is initialized to 06F Hex.

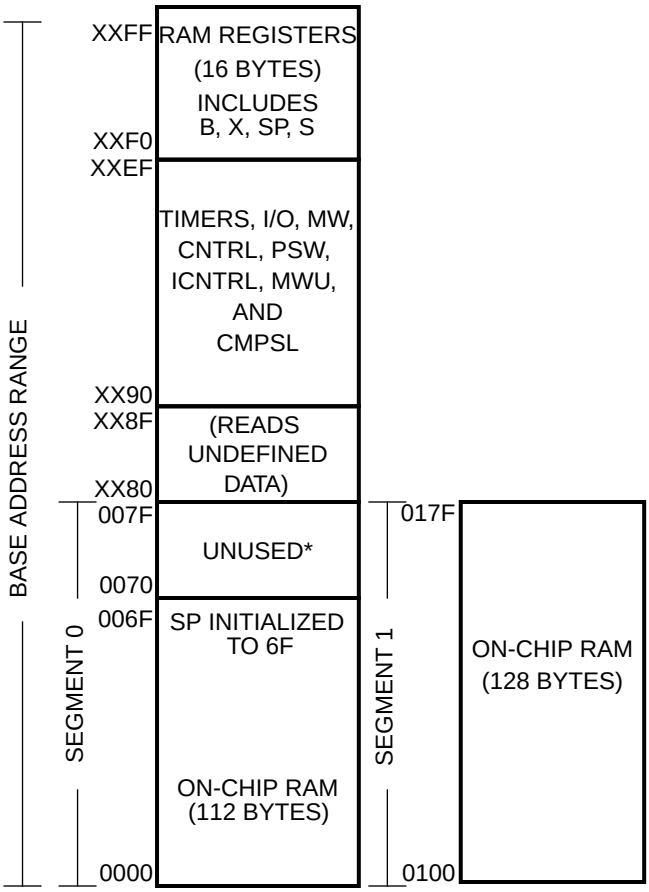
The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, and with both the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor detector circuits are inhibited during reset. The WATCHDOG service window bits are initialized to the maximum WATCHDOG service window of 64k tc clock cycles. The Clock Monitor bit is initialized high, and will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16–32 tc clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

The external RC network shown in Figure 3 should be used to ensure that the $\overline{\text{RESET}}$ pin is held low until the power supply to the chip stabilizes.



RC > 5 x POWER SUPPLY RISE TIME

Figure 3. Recommended Reset Circuit



* READS AS ALL ONES

Figure 2. RAM Organization

Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Note: External clocks with frequencies above about 4 MHz require the user to drive the CKO (G7) pin with a signal 180 degrees out of phase with CKI.

Figure 4 shows the Crystal and R/C diagrams.

CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

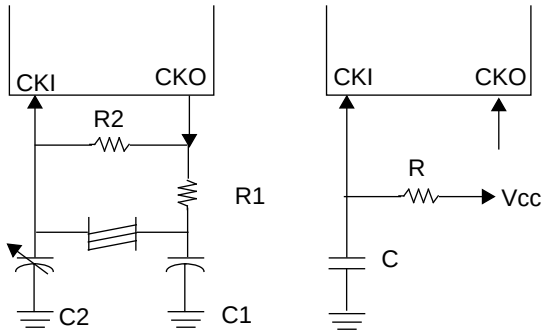


Figure 4. Crystal and R/C Oscillator Diagrams

Table 1 shows the component values required for various standard crystal values.

Table 1 Crystal Oscillator Configuration, $T_A = 25^{\circ}C$

R1 (kΩ)	R2 (MΩ)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30--36	10	V_{CC} = 5V
0	1	30	30--36	4	V_{CC} = 5V
0	1	200	100--150	0.455	V_{CC} = 5V

R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

Table 2 shows the variation in the oscillator frequencies as functions of the component (R and C) values.

Table 2 R/C Oscillator Configuration, $T_A = 25^{\circ}C$

R (kΩ)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.2 to 2.7	3.7 to 4.6	V_{CC} = 5V
5.6	100	1.1 to 1.3	7.4 to 9.0	V_{CC} = 5V
6.8	100	0.9 to 1.1	8.8 to 10.8	V_{CC} = 5V
Note: 3k ≤ R ≤ 200k 50 pF ≤ C ≤ 200 pF				

Current Drain

The total current drain of the chip depends on:

1. Oscillator operation mode—I1
2. Internal switching current—I2
3. Internal leakage current—I3
4. Output source current—I4
5. DC current caused by external input not at V_{CC} or GND—I5
6. DC reference current contribution from the A/D converter—I6
7. Clock Monitor current when enabled—I7

Thus the total current drain, It, is given as

$$It = I1 + I2 + I3 + I4 + I5 + I6 + I7$$

To reduce the total current drain, each of the above components must be minimum.

The chip will draw more current as the CKI input frequency increases up to the maximum 10 MHz value. Operating with a crystal network will draw more current than an external square-wave. Switching current, governed by the equation, can be reduced by lowering voltage and frequency. Leakage current can be reduced by lowering voltage and temperature. The other two items can be reduced by carefully designing the end-user's system.

$$I2 = C \times V \times f$$

where C = equivalent capacitance of the chip
V = operating voltage
f = CKI frequency

Control Registers

CNTRL Register (Address X'0xEE)

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

SL1 & SL0 Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)
IEDG External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)
MSEL Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively
T1C0 Timer T1 Start/Stop control in timer Timer T1 Underflow Interrupt Pending Flag in timer mode 3
T1C1 Timer T1 mode control bit
T1C2 Timer T1 mode control bit
T1C3 Timer T1 mode control bit

T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0
Bit 7							Bit 0

PSW Register (Address X'0xEF)

The PSW register contains the following select bits:

GIE Global interrupt enable (enables interrupts)
EXEN Enable external interrupt
BUSY MICROWIRE/PLUS busy shifting flag
EXPND External interrupt pending
T1ENA Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge
T1PNDA Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3)
C Carry Flag
HC Half Carry Flag

HC	C	T1PNDA	T1ENA	EXPND	BUSY	EXEN	GIE
Bit 7							Bit 0

The Half-Carry bit is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and RC (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and RC instructions, ADC, SUBC, RRC and RLC instructions affect the carry and Half Carry flags.

ICNTRL Register (Address X'0xE8)

The ICNTRL register contains the following bits:

T1ENB Timer T1 Interrupt Enable for T1B Input capture edge
T1PNDB Timer T1 Interrupt Pending Flag for T1B capture edge
∝WEN Enable MICROWIRE/PLUS interrupt
µWPND MICROWIRE/PLUS interrupt pending
T0EN Timer T0 Interrupt Enable (Bit 12 toggle)
T0PND Timer T0 Interrupt pending
LPEN L Port Interrupt Enable (Multi-Input Wakeup/Interrupt)
Bit 7 could be used as a flag

Unused	LPEN	T0PND	T0EN	µWPND	µWEN	T1PNDB	T1ENB

Bit 7 Bit 0

T2CNTRL Register (Address X'0xC6)

The T2CNTRL register contains the following bits:

T2ENB Timer T2 Interrupt Enable for T2B Input capture edge
T2PNDB Timer T2 Interrupt Pending Flag for T2B capture edge
T2ENA Timer T2 Interrupt Enable for Timer Underflow or T2A Input capture edge
T2PNDA Timer T2 Interrupt Pending Flag (Autoreload RA in mode 1, T2 Underflow in mode 2, T2A capture edge in mode 3)
T2C0 Timer T2 Start/Stop control in timer modes 1 and 2 Timer T2 Underflow Interrupt Pending Flag in timer mode 3
T2C1 Timer T2 mode control bit
T2C2 Timer T2 mode control bit
T2C3 Timer T2 mode control bit

T2C3	T2C2	T2C1	T2C0	T2PNDA	T2ENA	T2PNDB	T2ENB

Bit 7 Bit 0

T3CNTRL Register (Address X'0xB6)

The T3CNTRL register contains the following bits:

T3ENB Timer T3 Interrupt Enable for T3B Input capture edge
T3PNDB Timer T3 Interrupt Pending Flag for T3B capture edge
T3ENA Timer T3 Interrupt Enable for Timer Underflow or T3A Input capture edge
T3PNDA Timer T3 Interrupt Pending Flag (Autoreload RA in mode 1, T3 Underflow in mode 2, T3A capture edge in mode 3)
T3C0 Timer T3 Start/Stop control in timer modes 1 and 2 Timer T3 Underflow Interrupt Pending Flag in timer mode 3
T3C1 Timer T3 mode control bit
T3C2 Timer T3 mode control bit
T3C3 Timer T3 mode control bit

T3C3	T3C2	T3C1	T3C0	T3PNDA	T3ENA	T3PNDB	T3ENB

Bit 7 Bit 0

Timers

The device contains a very versatile set of timers (T0, T1, T2 and T3). All timers and associated autoreload/capture registers power up containing random data.

TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock, tc. The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

Figure 5 is a functional block diagram showing the structure of the IDLE Timer and its associated interrupt logic.

Bits 11 through 15 of the ITMR register can be selected for triggering the IDLE Timer interrupt. Each time the selected bit underflows (every 4k, 8k, 16k, 32k or 64k instruction cycles), the IDLE Timer interrupt pending bit T0PND is set, thus generating an interrupt (if enabled), and bit 6 of the Port G data register is reset, thus causing an exit from the IDLE mode if the device is in that mode.

In order for an interrupt to be generated, the IDLE Timer interrupt enable bit T0EN must be set, and the GIE (Global Interrupt Enable) bit must also be set. The T0PND flag and T0EN bit are bits 5 and 4 of the ICNTRL register, respectively. The interrupt can be used for any purpose. Typically, it is used to perform a task upon exit from the IDLE mode. For more information on the IDLE mode, refer to the Power Save Modes section.

The Idle Timer period is selected by bits 0-2 of the ITMR register Bits 3-7 of the ITMR Register are reserved and should not be used as software flags.

Table 3 Idle Timer Window Length

ITSEL2	ITSEL1	ITSEL0	Idle Timer Period (Instruction Cycles)
0	0	0	4,096
0	0	1	8,192
0	1	0	16,384
0	1	1	32,768
1	X	X	65,536

The ITMR register is cleared on Reset and the Idle Timer period is reset to 4,096 instruction cycles.

ITMR Register (Address X'0xCF)

Reserved	ITSEL2	ITSEL1	ITSEL0

Bit 7 Bit 0

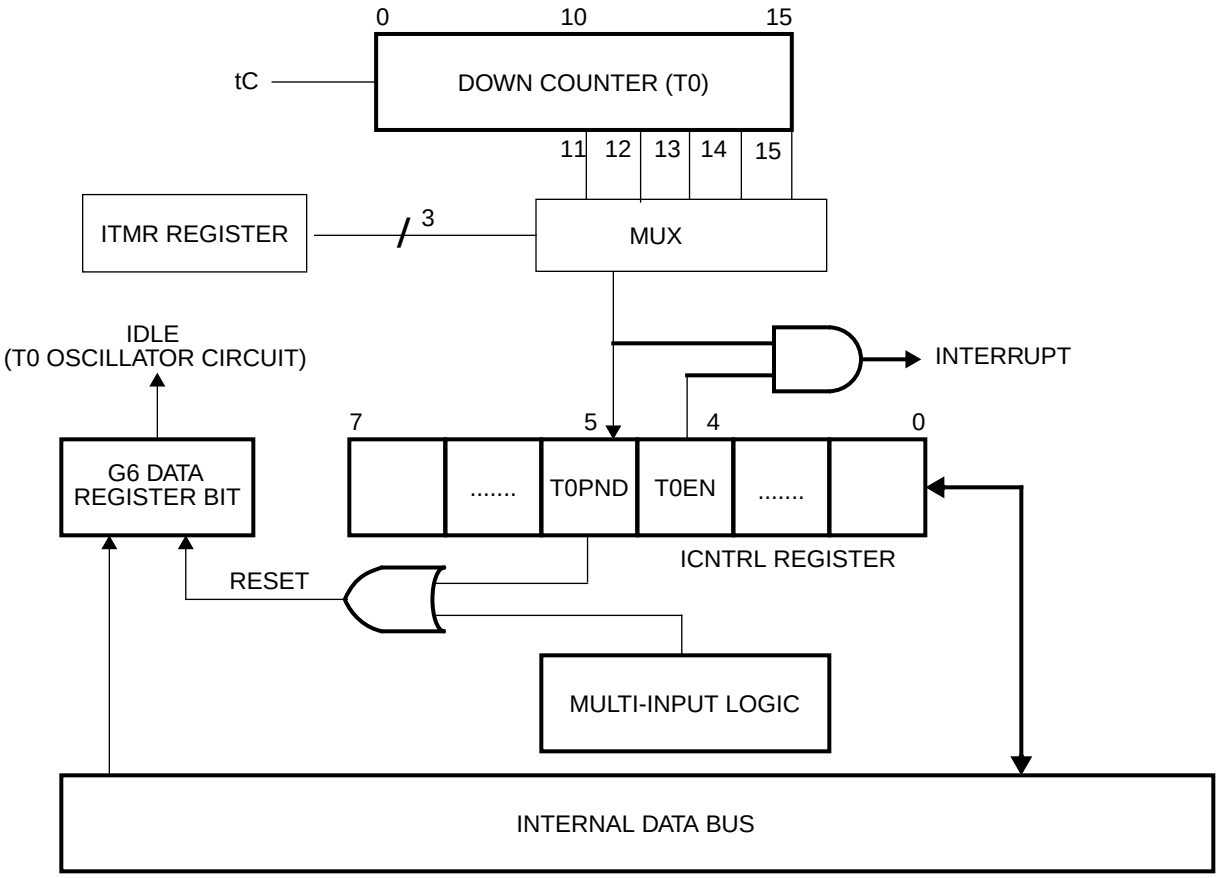


Figure 5. Functional Block Diagram for Idle Timer T0

Any time the IDLE Timer period is changed there is the possibility of generating a spurious IDLE Timer interrupt by setting the T0PND bit. The user is advised to disable IDLE Timer interrupts prior to changing the value of the ITSEL bits of the ITMR Register and then clear the T0PND bit before attempting to synchronize operation to the IDLE Timer.

TIMER T1, TIMER T2, AND TIMER T3

The device has a set of three powerful timer/counter blocks, T1, T2, and T3. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the three timer blocks, T1, T2 and T3 are identical, all comments are equally applicable to either of the three timer blocks.

Each timer block consists of a 16-bit timer, Tx, and two supporting 16-bit autoreload/capture registers, RxA and RxB. Each timer block has two pins associated with it, TxA and TxB. The pin TxA supports I/O required by the timer block, while the pin TxB is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits TxC3, TxC2, and TxC1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention.

The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

In this mode the timer Tx counts down at a fixed rate of tc. Upon every underflow the timer is alternately reloaded with the contents of supporting registers, RxA and RxB. The very first underflow of the timer causes the timer to reload from the register RxA. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register RxB.

The Tx Timer control bits, TxC3, TxC2 and TxC1 set up the timer for PWM mode operation.

Figure 6 shows a block diagram of the timer in PWM mode.

The underflows can be programmed to toggle the TxA output pin. The underflows can also be programmed to generate interrupts.

Underflows from the timer are alternately latched into two pending flags, TxPNDA and TxPNDB. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the

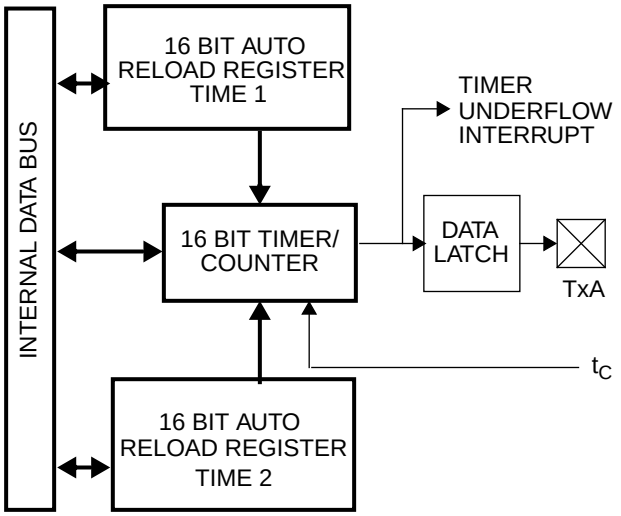


Figure 6. Timer in PWM Mode

timer underflow to be enabled or disabled. Setting the timer enable flag TxENA will cause an interrupt when a timer underflow causes the RxA register to be reloaded into the timer. Setting the timer enable flag TxENB will cause an interrupt when a timer underflow causes the RxB register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode described above. The main difference is that the timer, Tx, is clocked by the input signal from the TxA pin. The Tx timer control bits, TxC3, TxC2 and TxC1 allow the timer to be clocked either on a positive or negative edge from the TxA pin. Underflows from the timer are latched into the TxPNDA pending flag. Setting the TxENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin TxB can be used as an independent positive edge sensitive interrupt input if the TxENB control flag is set. The occurrence of a positive edge on the TxB input pin is latched into the TxPNDB flag.

Figure 7 shows a block diagram of the timer in External Event Counter mode.

Note: The PWM output is not available in this mode since the TxA pin is being used as the counter input clock.

Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, Tx, in the input capture mode.

In this mode, the timer Tx is constantly running at the fixed t_c rate. The two registers, RxA and RxB, act as capture registers. Each register acts in conjunction with a pin. The register RxA acts in conjunction with the TxA pin and the register RxB acts in conjunction with the TxB pin.

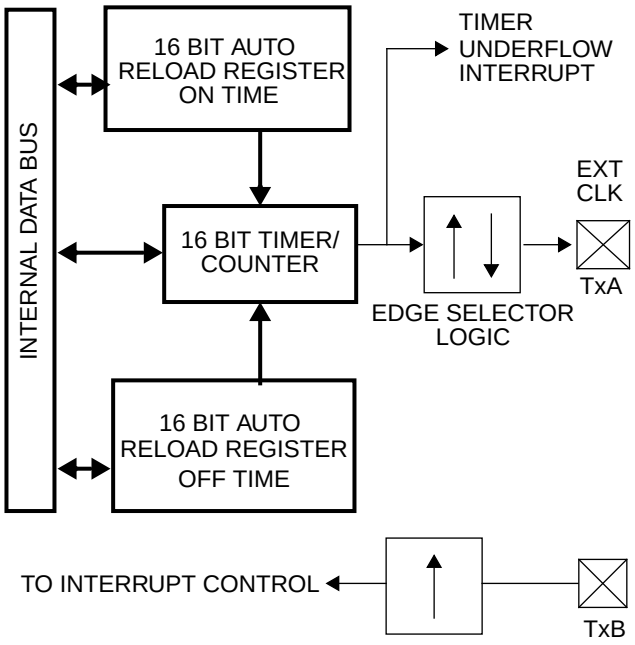


Figure 7. Timer in External Event Counter Mode

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, TxC3, TxC2 and TxC1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the TxA and TxB pins will be respectively latched into the pending flags, TxPNDA and TxPNDB. The control flag TxENA allows the interrupt on TxA to be either enabled or disabled. Setting the TxENA flag enables interrupts to be generated when the selected trigger condition occurs on the TxA pin. Similarly, the flag TxENB controls the interrupts from the TxB pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer TxC0 pending flag (the TxC0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxC0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the TxENA control flag. When a TxA interrupt occurs in the Input Capture mode, the user must check both whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 8 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The timers T1, T2 and T3 have identical control structures. The control bits and their functions are summarized below.

TxC0 Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop
Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture)

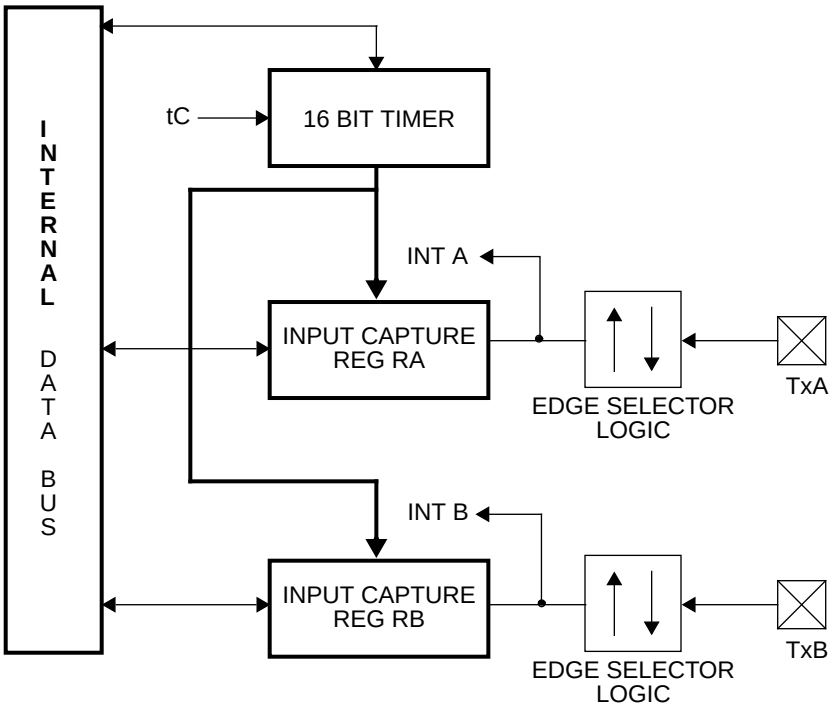


Figure 8. Timer in Input Capture Mode

TxPNDA Timer Interrupt Pending Flag
TxPNDB Timer Interrupt Pending Flag
TxENA Timer Interrupt Enable Flag
TxENB Timer Interrupt Enable Flag
1 = Timer Interrupt Enabled
0 = Timer Interrupt Disabled

TxC3 Timer mode control
TxC2 Timer mode control
TxC1 Timer mode control

The timer mode control bits (TxC3, TxC2 and TxC1) are detailed below:

TxC3	TxC2	TxC1	Timer Mode	Interrupt A Source	Interrupt B Source	Timer Counts On
0	0	0	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Pos. Edge
0	0	1	MODE 2 (External Event Counter)	Timer Underflow	Pos. TxB Edge	TxA Neg. Edge
1	0	1	MODE 1 (PWM) TxA Toggle	Autoreload RA	Autoreload RB	t_C
1	0	0	MODE 1 (PWM) No TxA Toggle	Autoreload RA	Autoreload RB	t_C
0	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Pos. Edge	Pos. TxA Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	0	MODE 3 (Capture) Captures: TxA Pos. Edge TxB Neg. Edge	Pos. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C
0	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Pos. Edge	Neg. TxB Edge or Timer Underflow	Pos. TxB Edge	t_C
1	1	1	MODE 3 (Capture) Captures: TxA Neg. Edge TxB Neg. Edge	Neg. TxA Edge or Timer Underflow	Neg. TxB Edge	t_C

Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

HALT MODE

The device is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, timers, and A/D converter, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry if enabled remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage (V_{CC}) may be decreased to Vr (Vr = 2.0V) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the L port. The second 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 becomes a dedicated output), and so may be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the RESET pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the tc instruction cycle clock. The tc clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The start-up time-out from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect).

The WATCHDOG detector circuit is inhibited during the HALT mode. However, the clock monitor circuit if enabled remains active during HALT mode in order to ensure a clock monitor error if the device inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The device is placed in the IDLE mode by writing a "1" to the IDLE flag (G6 data bit). In this mode, all activity, except the associated on-board oscillator circuitry, the WATCHDOG logic, the clock monitor and the IDLE Timer T0, is stopped. The power supply requirements of the microcontroller in this mode of operation are typically around 30% of normal power requirement of the microcontroller.

As with the HALT mode, the device can be returned to normal operation with a reset, or with a Multi-Input Wakeup from the L Port.

The microcontroller may also be awakened from the IDLE mode after a selectable amount of time up to 65,536 instruction cycles, or 65.536 milliseconds with a 1 MHz instruction clock frequency.

The IDLE timer period is selectable from one of five values, 4k, 8k, 16k, 32k or 64k instruction cycles. Selection of this value is made through the ITMR register.

The user has the option of being interrupted with an underflow of the selected bit of the IDLE Timer T0. This condition is latched into the T0PND pending flag. The interrupt can be enabled or disabled via the T0EN control bit. Setting the T0EN flag enables the interrupt and vice versa.

The user can enter the IDLE mode with the Timer T0 interrupt enabled. In this case, when the T0PND bit gets set, the device will first execute the Timer T0 interrupt service routine and then return to the instruction following the "Enter Idle Mode" instruction.

Alternatively, the user can enter the IDLE mode with the IDLE Timer T0 interrupt disabled. In this case, the device will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

The IDLE timer cannot be started or stopped under software control, and it is not memory mapped, so it cannot be read or written by the software. Its state upon Reset is unknown. Therefore, if the device is put into the IDLE mode at an arbitrary time, it will stay in the IDLE mode for somewhere between 1 and the selected number of instruction cycles.

Upon reset the ITMR register is cleared and selects the 4,096 instruction cycle tap of the Idle Timer.

Note: It is necessary to program two NOP instructions following both the set HALT mode and set IDLE mode instructions. These NOP instructions are necessary to allow clock resynchronization following the HALT or IDLE modes.

For more information on the IDLE Timer and its associated interrupt, see the description in the Timers Section.

Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 8 edge selectable external interrupts.

Figure 9 shows the Multi-Input Wakeup logic.The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the Reg: WKEN. The Reg: WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the Reg: WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a pseudo Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT  5, WKEN
SBIT  5, WKEDG
RBIT  5, WKPND
SBIT  5, WKEN
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid inherited pseudo wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user has the responsibility of clearing the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

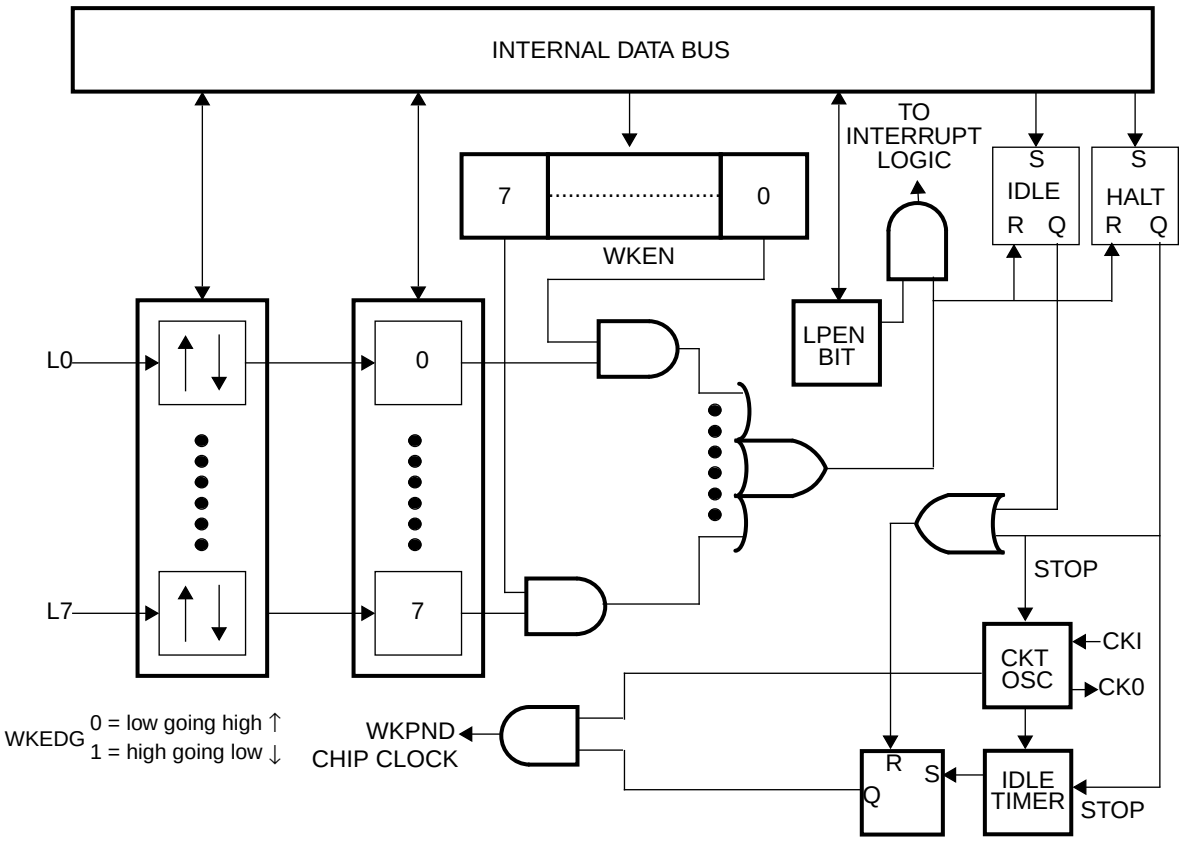


Figure 9. Multi-Input Wake Up Logic

A/D Converter

The device contains an 8-channel, multiplexed input, successive approximation, Analog-to-Digital convertor. The device's VCC and GND pins are used for voltage reference.

OPERATING MODES

The A/D convertor supports ratiometric measurements. It supports both Single Ended and Differential modes of operation.

Four specific analog channel selection modes are supported. These are as follows:

Allow any specific channel to be selected at one time. The A/D convertor performs the specific conversion requested and stops.

Allow any specific channel to be scanned continuously. In other words, the user specifies the channel and the A/D convertor scans it continuously. At any arbitrary time the user can immediately read the result of the last conversion. The user must wait for only the first conversion to complete.

Allow any differential channel pair to be selected at one time. The A/D convertor performs the specific differential conversion requested and stops.

Allow any differential channel pair to be scanned continuously. In other words, the user specifies the differential channel pair and the A/D convertor scans it continuously. At any arbitrary time the user can immediately read the result of the last differential conversion. The user must wait for only the first conversion to complete.

The A/D convertor is supported by two memory mapped registers, the result register and the mode control register. When the device is reset, the mode control register (ENAD) is cleared, the A/D is powered down and the A/D result register has unknown data.

A/D Control Register

The ENAD control register contains 3 bits for channel selection, 2 bits for prescaler selection, 2 bits for mode selection and a Busy bit. An A/D conversion is initiated by setting the ADBSY bit in the ENAD control register. The result of the conversion is available to the user in the A/D result register, ADRSLT, when ADBSY is cleared by the hardware on completion of the conversion.

ENAD (Address 0xCB)

CHANNEL SELECT			MODE SELECT		PRESCALER SELECT		BUSY
ADCH2	ADCH1	ADCH0	ADMOD1	ADMOD0	PSC1	PSC0	ADBSY

Bit 7 Bit 0

CHANNEL SELECT

This 3-bit field selects one of eight channels to be the V_{IN+}. The mode selection determines the V_{IN-} input.

Single Ended mode:

Bit 7	Bit 6	Bit 5	Channel No.
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Differential mode:

Bit 7	Bit 6	Bit 5	Channel Pairs (+. -)
0	0	0	0, 1
0	0	1	1, 0
0	1	0	2, 3
0	1	1	3, 2
1	0	0	4, 5
1	0	1	5, 4
1	1	0	6, 7
1	1	1	7, 6

MODE SELECT

This 2-bit field is used to select the mode of operation (single conversion, continuous conversions, differential, single ended) as shown in the following table.

Bit 4	Bit 3	Mode
0	0	Single Ended mode, single conversion
0	1	Single Ended mode, continuous scan of a single channel into the result register
1	0	Differential mode, single conversion
1	1	Differential mode, continuous scan of a channel pair into the result register

PRESCALER SELECT

This 2-bit field is used to select one of the four prescaler clocks for the A/D converter. The following table shows the various prescaler options.

A/D Convertor Clock Prescale

Bit 2	Bit 1	Clock Select
0	0	Divide by 2
0	1	Divide by 4
1	0	Divide by 6
1	1	Divide by 12

BUSY BIT

The ADBSY bit of the ENAD register is used to control starting and stopping of the A/D conversion. When ADBSY is cleared, the prescale logic is disabled and the A/D clock is turned off. Setting the ADBSY bit starts the A/D clock and initiates a conversion based on the mode select value currently in the ENAD register. Normal completion of an A/D conversion clears the ADBSY bit and turns off the A/D convertor.

The ADBSY bit remains a one during continuous conversion. The user can stop continuous conversion by writing a zero to the ADBSY bit.

If the user wishes to restart a conversion which is already in progress, this can be accomplished only by writing a zero to the ADBSY bit to stop the current conversion and then by writing a one to ADBSY to start a new conversion. This can be done in two consecutive instructions.

ADC Operation

The A/D convertor interface works as follows. Setting the ADBSY bit in the A/D control register ENAD initiates an A/D conversion. The conversion sequence starts at the beginning of the write to ENAD operation which sets ADBSY, thus powering up the A/D. At the first falling edge of the convertor clock following the write operation, the sample signal turns on for seven clock cycles. If the A/D is in single conversion mode, the conversion complete signal from the A/D will generate a power down for the A/D convertor and will clear the ADBSY bit in the ENAD register at the next instruction cycle boundary. If the A/D is in continuous mode, the conversion complete signal will restart the conversion sequence by deselecting the A/D for one convertor clock cycle before starting the next sample. The A/D 8-bit result is immediately loaded into the A/D result register (ADRSLT) upon completion. Internal logic prevents transient data (resulting from the A/D writing a new result over an old one) being read from ADRSLT.

Inadvertent changes to the ENAD register during conversion are prevented by the control logic of the A/D. Any attempt to write any bit of the ENAD Register except ADBSY, while ADBSY is a one, is ignored. ADBSY must be cleared either by completion of an A/D conversion or by the user before the prescaler, conversion mode or channel select values can be changed. After stopping the current conversion, the user can load different values for the prescaler, conversion mode or channel select and start a new conversion in one instruction.

It is important for the user to realize that, when used in differential mode, only the positive input to the A/D converter is sampled and held. The negative input is constantly connected and should be held stable for the duration of the conversion. Failure to maintain a stable negative input will result in incorrect conversion.

PRESCALER

The A/D Convertor (A/D) contains a prescaler option that allows four different clock selections. The A/D clock frequency is equal to CKI divided by the prescaler value. Note that the prescaler value must be chosen such that the A/D clock falls within the specified range. The maximum A/D frequency is 1.67 MHz. This equates to a 600 ns A/D clock cycle.

The A/D convertor takes 17 A/D clock cycles to complete a conversion. Thus the minimum A/D conversion time for the device is 10.2 μs when a prescaler of 6 has been selected. The 17 A/D clock cycles needed for conversion consist of 1 cycle at the beginning for reset, 7 cycles for sampling, 8 cycles for converting, and 1 cycle for loading the result into the A/D result register (ADRSLT). This A/D result register is a read-only register. The user cannot write into ADRSLT.

The ADBSY flag provides an A/D clock inhibit function, which saves power by powering down the A/D when it is not in use.

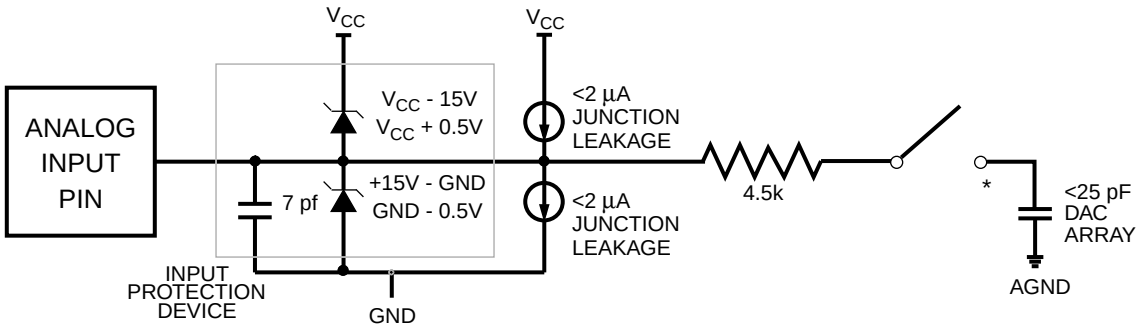
Note: The A/D convertor is also powered down when the device is in either the HALT or IDLE modes. If the A/D is running when the device enters the HALT or IDLE modes, the A/D powers down and then restarts the conversion with a corrupted sampled voltage (and thus an invalid result) when the device comes out of the HALT or IDLE modes.

Analog Input and Source Resistance Considerations

Figure 10 shows the A/D pin model in single ended mode. The differential mode has a similar A/D pin model. The leads to the analog inputs should be kept as short as possible. Both noise and digital clock coupling to an A/D input can cause conversion errors. The clock lead should be kept away from the analog input line to reduce coupling. The A/D channel input pins do not have any internal output driver circuitry connected to them because this circuitry would load the analog input signals due to output buffer leakage current.

Source impedances greater than 3 kΩ on the analog input lines will adversely affect the internal RC charging time during input sampling. As shown in Figure 10, the analog switch to the DAC array is closed only during the 7 A/D cycle sample time. Large source impedances on the analog inputs may result in the DAC array not being charged to the correct voltage levels, causing scale errors.

If large source resistance is necessary, the recommended solution is to slow down the A/D clock speed in proportion to the source resistance. The A/D convertor may be operated at the maximum speed for R_S less than 3 kΩ . For R_S greater than 3 kΩ , A/D clock speed needs to be reduced. For example, with R_S = 6 kΩ , the A/D convertor may be operated at half the maximum speed. A/D convertor clock speed may be slowed down by either increasing the A/D prescaler divide-by or decreasing the CKI clock frequency. The A/D minimum clock speed is 100 kHz.



*The analog switch is closed only during the sample time.

Figure 10. A/D Pin Model (Single Ended Mode)

Interrupts

Introduction

The device supports fourteen vectored interrupts. Interrupt sources include Timer 1, Timer 2, Timer 3, Timer T0, Port L Wakeup, Software Trap, MICROWIRE/PLUS, UART and External Input.

All interrupts force a branch to location 00FF Hex in program memory. The VIS instruction may be used to vector to the appropriate service routine from location 00FF Hex.

The Software trap has the highest priority while the default VIS has the lowest priority.

Each of the 13 maskable inputs has a fixed arbitration ranking and vector.

Figure 11 shows the Interrupt block diagram.

Maskable Interrupts

All interrupts other than the Software Trap are maskable. Each maskable interrupt has an associated enable bit and pending flag bit. The pending bit is set to 1 when the interrupt condition occurs. The state of the interrupt enable bit, combined with the GIE bit determines whether an active pending flag actually triggers an interrupt. All of the maskable interrupt pending and enable bits are contained in mapped control registers, and thus can be controlled by the software.

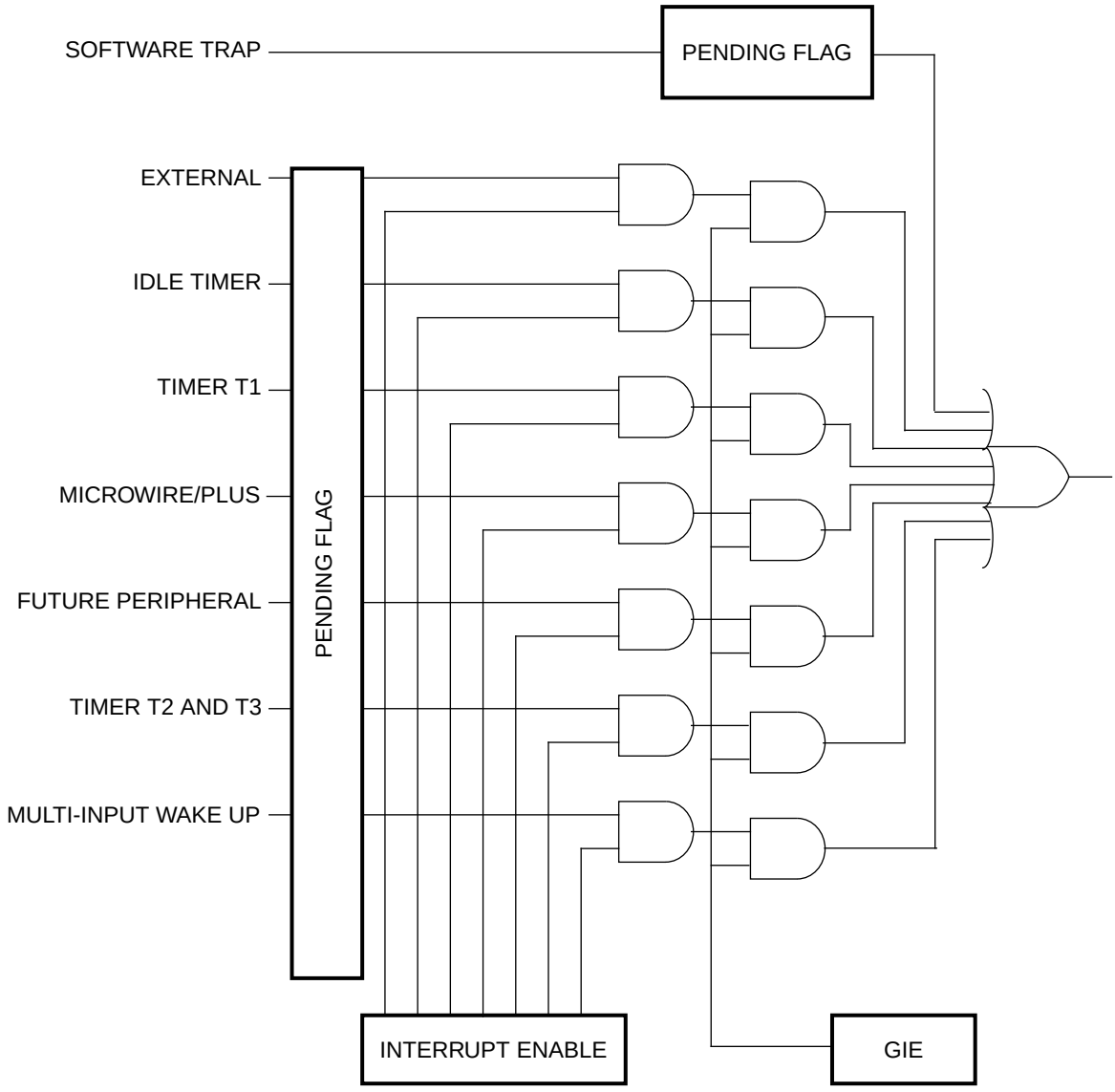


Figure 11. COP888GD Interrupt Block Diagram

A maskable interrupt condition triggers an interrupt under the following conditions:

1. The enable bit associated with that interrupt is set.
2. The GIE bit is set.
3. The device is not processing a non-maskable interrupt. (If a non-maskable interrupt is being serviced, a maskable interrupt must wait until that service routine is completed.)

An interrupt is triggered only when all of these conditions are met at the beginning of an instruction. If different maskable interrupts meet these conditions simultaneously, the highest-priority interrupt will be serviced first, and the other pending interrupts must wait.

Upon Reset, all pending bits, individual enable bits, and the GIE bit are reset to zero. Thus, a maskable interrupt condition cannot trigger an interrupt until the program enables it by setting both the GIE bit and the individual enable bit. When enabling an interrupt, the user should consider whether or not a previously activated (set) pending bit should be acknowledged. If, at the time an interrupt is enabled, any previous occurrences of the interrupt should be ignored, the associated pending bit must be reset to zero prior to enabling the interrupt. Otherwise, the interrupt may be simply enabled; if the pending bit is already set, it will immediately trigger an interrupt. A maskable interrupt is active if its associated enable and pending bits are set.

An interrupt is an asychronous event which may occur before, during, or after an instruction cycle. Any interrupt which occurs during the execution of an instruction is not acknowledged until the start of the next normally executed instruction is to be skipped, the skip is performed before the pending interrupt is acknowledged.

At the start of interrupt acknowledgment, the following actions occur:

1. The GIE bit is automatically reset to zero, preventing any subsequent maskable interrupt from interrupting the current service routine.This feature prevents one maskable interrupt from interrupting another one being serviced.
2. The address of the instruction about to be executed is pushed onto the stack.
3. The program counter(PC) is loaded with 00FF Hex, causing a jump to that program memory location.

The device requires seven instruction cycles to perform the actions listed above.

If the user wishes to allow nested interrupts, the interrupts service routine may set the GIE bit to 1 by writing to the PSW register, and thus allow other maskable interrupts to interrupt the current service routine. If nested interrupts are allowed, caution must be exercised. The user must write the program in such a was as to prevent stack overflow, loss of saved context information, and other unwanted conditions.

The interrupt service routine stored at location 00FF Hex should use the VIS instruction to determine the cause of the interrupt, and jump to the interrupt handling routine corresponding to the highest priority enabled and active interrupt. Alternately, the user may choose to poll all interrupt pending and enable bits to determine the source(s) of the interrupt. If more than one interrupt is active, the user's program must decide which interrupt to service.

Within a specific interrupt service routine, the associated pending bit should be cleared. This is typically done as early as possible in the service routine in order to avoid missing the next occurrence of the same type of interrupt event. Thus, if the same event occurs a second time, even while the first occurrence is still being serviced, the second occurrence will be serviced immediately upon return from the current interrupt routine.

An interrupt service routine typically ends with an RETI instruction. This instruction set the GIE bit back to 1, pops the address stored on the stack, and restores that address to the program counter. Program execution then proceeds with the next instruction that would have been executed had there been no interrupt. If there are any valid interrupts pending, the highest-priority interrupt is serviced immediately upon return from the previous interrupt.

VIS Instruction

The general interrupt service routine, which starts at address 00FF Hex, must be capable of handling all types of interrupts. The VIS instruction, together with an interrupt vector table, directs the device to the specific interrupt handling routine based on the cause of the interrupt.

VIS is a single-byte instruction, typically used at the very beginning of the general interrupt service routine at address 00FF Hex, or shortly after that point, just after the code used for context switching. The VIS instruction determines which enabled and pending interrupt has the highest priority, and causes an indirect jump to the address corresponding to that interrupt source. The jump addresses (vectors) for all possible interrupts sources are stored in a vector table.

The vector table may be as long as 32 bytes (maximum of 16 vectors) and resides at the top of the 256-byte block containing the VIS instruction. However, if the VIS instruction is at the very top of a 256-byte block (such as at 00FF Hex), the vector table resides at the top of the next 256-byte block. Thus, if the VIS instruction is located somewhere between 00FF and 01DF Hex (the usual case), the vector table is located between addresses 01E0 and 01FF Hex. If the VIS instruction is located between 01FF and 02DF Hex, then the vector table is located between addresses 02E0 and 02FF Hex, and so on.

Each vector is 15 bits long and points to the beginning of a specific interrupt service routine somewhere in the 32-Kbyte memory space. Each vector occupies two bytes of the vector table, with the higher-order byte at the lower address. The vectors are arranged in order of interrupt priority. The vector of the maskable interrupt with the lowest rank is located to 0yE0 (higher-order byte) and 0yE1 (lower-order byte). The next priority interrupt is located at 0yE2 and 0yE3, and so forth in increasing rank. The Software Trap has the highest rand and its vector is always located at 0yFE and 0yFF. The number of interrupts which can become active defines the size of the table.

Table 4 shows the types of interrupts, the interrupt arbitration ranking, and the locations of the corresponding vectors in the vector table.

The vector table should be filled by the user with the memory locations of the specific interrupt service routines. For example, if the Software Trap routine is located at 0310 Hex, then

the vector location 0yFE and -0yFF should contain the data 03 and 10 Hex, respectively. When a Software Trap interrupt occurs and the VIS instruction is executed, the program jumps to the address specified in the vector table.

The interrupt sources in the vector table are listed in order of rank, from highest to lowest priority. If two or more enabled and pending interrupts are detected at the same time, the one with the highest priority is serviced first. Upon return from the interrupt service routine, the next highest-level pending interrupt is serviced.

If the VIS instruction is executed, but no interrupts are enabled and pending, the lowest-priority interrupt vector is used, and a jump is made to the corresponding address in the vector table. This is an unusual occurrence and may be the result of an error. It can legitimately result from a change in the enable bits or pending flags prior to the execution of the VIS instruction, such as executing a single cycle instruction which clears an enable flag at the same time that the pending flag is set. It can also result, however, from inadvertent execution of the VIS command outside of the context of an interrupt.

The default VIS interrupt vector can be useful for applications in which time critical interrupts can occur during the servicing of another interrupt. Rather than restoring the program context (A, B, X, etc.) and executing the RETI instruction, an interrupt service routine can be terminated by returning to the VIS instruction. In this case, interrupts will be serviced in turn until no further interrupts are pending and the default VIS routine is started. After testing the GIE bit to ensure that ex4ecution is not erroneous, the routine should restore the program context and execute the RETI to return to the interrupted program.

This technique can save up to fifty instruction cycles (t_c), or more, (50 s at 10 MHz oscillator) of latency for pending interrupts with a penalty of fewer than ten instruction cycles if no further interrupts are pending.

To ensure reliable operation, the user should always use the VIS instruction to determine the source of an interrupt. Although it is possible to poll the pending bits to detect the source of an interrupt, this practice is not recommended. The use of polling allows the standard arbitration ranking to be altered, but the reliability of the interrupt system is compromised. The polling routine must individually test the enable and pending bits of each maskable interrupt. If a Software Trap interrupt should occur, it will be serviced last, even though it should have the highest priority. Under certain conditions, a Software Trap could be triggered but not serviced, resulting in an inadvertent “locking out” of all maskable interrupts by the Software Trap pending flag. Problems such as this can be avoided by using VIS instruction.

Table 4 Interrupt Vector Table

ARBITRATION RANKING	SOURCE	DESCRIPTION	VECTOR* ADDRESS (Hi-Low Byte)
(1) Highest	Software	INTR Instruction	0yFE - 0yFF
(2)	Reserved		0yFC - 0yFD
(3)	External	Pin G0 Edge	0yFA - 0yFB
(4)	Timer T0	Underflow	0yF8 - 0yF9
(5)	Timer T1	T1A/Underflow	0yF6 - 0yF7
(6)	Timer T1	T1B	0yF4 - 0yF5
(7)	MICROWIRE/PLUS	Busy Low	0yF2 - 0yF3
(8)	Reserved		0yF0 - 0yF1
(9)	Reserved		0yEE - 0yEF
(10)	Reserved		0yEC - 0yED
(11)	Timer T2	T2A/Underflow	0yEA - 0yEB
(12)	Timer T2	T2B	0yE8 - 0yE9
(13)	Timer T3	T3A/Underflow	0yE6 - 0yE7
(14)	Timer T3	T3B	0yE4 - 0yE5
(15)	Port L/Wakeup	Port L Edge	0yE2 - 0yE3
(16) Lowest	Default	VIS instr. Execution without Any Interrupts	0yE0 - 0yE1

* The location of the vector table depends on the location of the VIS instruction. Vector addresses shown in the table assume a VIS instruction between 00FF and 01DF Hex

VIS Execution

When the VIS instruction is executed it activates the arbitration logic. The arbitration logic generates an even number between E0 and FE (E0, E2, E4, E6 etc...) depending on which active interrupt has the highest arbitration ranking at the time of the 1st cycle of VIS is executed. For example, if the software trap interrupt is active, FE is generated. If the external interrupt is active and the software trap interrupt is not, then FA is generated and so forth. If the only active interrupt is software trap, than E0 is generated. This number replaces the lower byte of the PC. The upper byte of the PC remains unchanged. The new PC is therefore pointing to the vector of the active interrupt with the highest arbitration ranking. This vector is read from program memory and placed into the PC which is now pointed to the 1st instruction of the service routine of the active interrupt with the highest arbitration ranking.

Figure 12 illustrates the different steps performed by the VIS instruction. Figure 13 shows a flowchart for the VIS instruction.

The non-maskable interrupt pending flag is cleared by the RPND (Reset Non-Maskable Pending Bit) instruction (under certain conditions) and upon RESET.

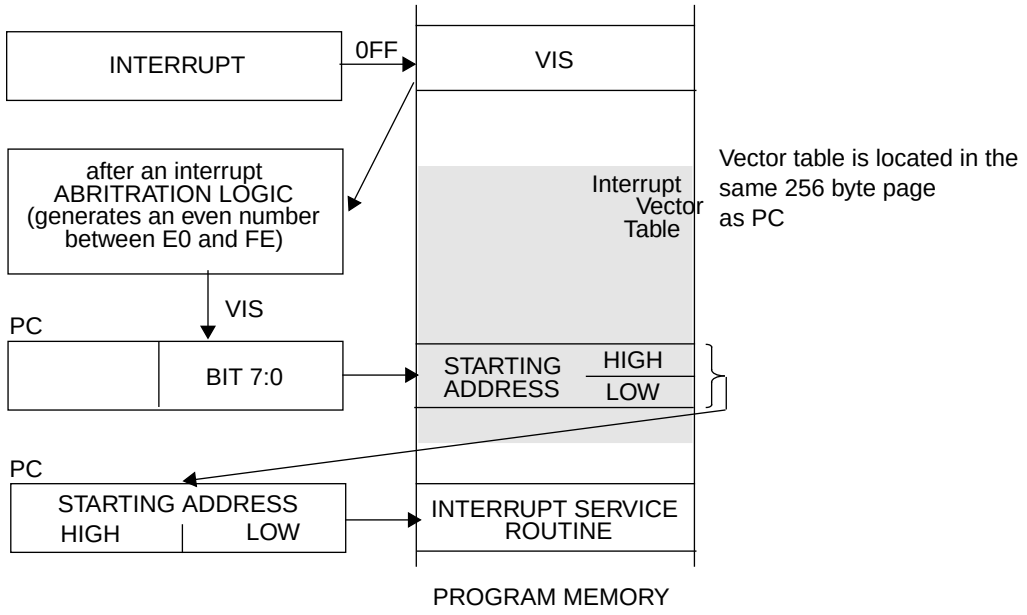


Figure 12. VIS Operation

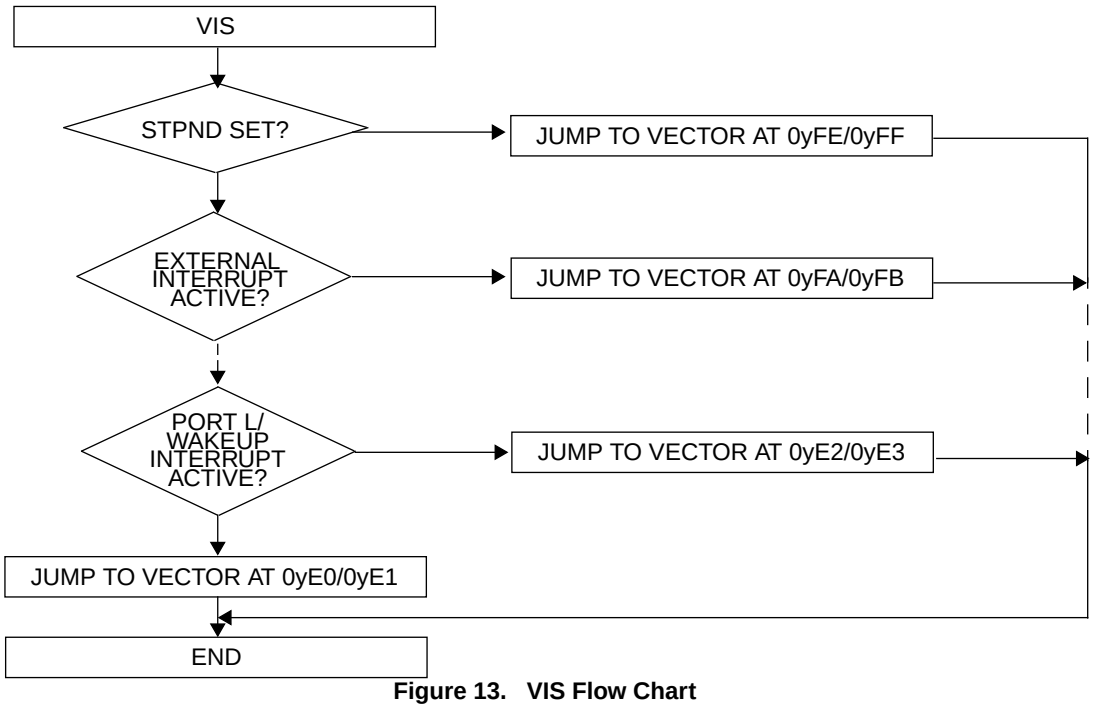


Figure 13. VIS Flow Chart

Programming Example: External Interrupt

```
                PSW             =00EF
                CNTRL           =00EE
                RBIT            0,PORTGC
                RBIT            0,PORTGD        ; G0 pin configured Hi-Z
                SBIT            IEDG, CNTRL     ; Ext interrupt polarity: falling edge
                SBIT            GIE, PSW        ; Set the GIE bit
                SBIT            EXEN, PSW       ; Enable the external interrupt
WAIT:           JP              WAIT            ; Wait for external interrupt
                .
                .
                .
                .=0FF                           ; The interrupt causes a
                VIS                             ; branch to address 0FF
                                                ; The VIS causes a branch to
                                                ; interrupt vector table
                .
                .
                .
                .=01FA                          ; Vector table (within 256 byte
                .ADDRW SERVICE                  ; of VIS inst.) containing the ext
                                                ; interrupt service routine
                .
                .
                .
SERVICE:                                        ; Interrupt Service Routine
                RBIT, EXPND, PSW                ; Reset ext interrupt pend. bit
                .
                .
                .
                RET                             ; Return, set the GIE bit
```

Non-maskable Interrupt

Pending Flag

There is a pending flag bit associated with the non-maskable interrupt, called STPND. This pending flag is not memory-mapped and cannot be accessed directly by the software.

The pending flag is reset to zero when a device Reset occurs. When the non-maskable interrupt occurs, the associated pending bit is set to 1. The interrupt service routine should contain an RPND instruction to reset the pending flag to zero. The RPND instruction always resets the STPND flag

Software Trap

The Software Trap is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from program memory and placed in the instruction register. This can happen in a variety of ways, usually because of an error condition. Some examples of causes are listed below.

If the program counter incorrectly points to a memory location beyond the available program memory space, the non-existent or unused memory location returns zeros which is interpreted as the INTR instruction.

If the stack is popped beyond the allowed limit (address 02F or 06F Hex), a Software Trap is triggered.

A Software Trap can be triggered by a temporary hardware condition such as a brownout or power supply glitch.

The Software Trap has the highest priority of all interrupts. When a Software Trap occurs, the STPND bit is set. The GIE bit is not affected and the pending bit (not accessible by the user) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. Nothing can interrupt a Software Trap service routine except for another Software Trap. The STPND can be reset only by the RPND instruction or a chip Reset.

The Software Trap indicates an unusual or unknown error condition. Generally, returning to normal execution at the point where the Software Trap occurred cannot be done reliably. Therefore, the Software Trap service routine should re-initialize the stack pointer and perform a recovery procedure that re-starts the software at some known point, similar to a device Reset, but not necessarily performing all the same functions as a device Reset. The routine must also execute the RPND instruction to reset the STPND flag. Otherwise, all other interrupts will be locked out. To the extent possible, the interrupt routine should record or indicate the context of the device so that the cause of the Software Trap can be determined.

If the user wishes to return to normal execution from the point at which the Software Trap was triggered, the user must first execute RPND, followed by RETSK rather tan RETI or RET. This is because the return address stored on the stack is the address of the INTR instruction that triggered the interrupt. The program must skip that instruction in order to proceed with the next one. Otherwise, an infinite loop of Software Traps and returns will occur.

Programming a return to normal execution requires careful consideration. If the Software Trap routine is interrupted by another Software Trap, the RPND instruction in the service routine for the second Software Trap will reset the STPND flag; upon return to the first Software Trap routine, the STPND flag will have the wrong state. This will allow maskable interrupts to be acknowledged during the servicing of the first Software Trap. To avoid problems such as this, the user program should contain the Software Trap routine to perform a recovery procedure rather than a return to normal execution.

Under normal conditions, the STPND flag is reset by a RPND instruction in the Software Trap service routine. If a programming error or hardware condition (brownout, power supply glitch, etc.) sets the STPND flag without providing a way for it to be cleared, all other interrupts will be locked out. To alleviate this condition, the user can use extra RPND instructions in the main program and in the Watchdog service routine (if present). There is no harm in executing extra RPND instructions in these parts of the program.

Port L Interrupts

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

Interrupt Summary

The device uses the following types of interrupts, listed below in order of priority:

1. The Software Trap non-maskable interrupt, triggered by the INTR (00 opcode) instruction. The Software Trap is acknowledged immediately. This interrupt service routine can be interrupted only by another Software Trap. The Software Trap should end with two RPND instructions followed by a re-start procedure.
2. Maskable interrupts, triggered by an on-chip peripheral block or an external device connected to the device. Under ordinary conditions, a maskable interrupt will not interrupt any other interrupt routine in progress. A maskable interrupt routine in progress can be interrupted by the non-maskable interrupt request. A maskable interrupt routine should end with an RETI instruction or should return to execute the VIS instruction. This is particularly useful when exiting long interrupt service routines if the time between interrupts is short. In this case the RETI instruction would only be executed when the default VIS routine is reached.

WATCHDOG/Clock Monitor

The devices contain a user selectable WATCHDOG and clock monitor. The following section is applicable only if WATCHDOG feature has been selected in the ECON register. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs.

The WATCHDOG logic contains two separate service windows. While the user programmable upper window selects the Watchdog service time, the lower widow provides protection against an infinite program loop that contains the watchdog service instruction.

The COP8SGR7 devices provide the added feature of a software trap that provides protection against stack overpops and addressing locations outside valid user program space.

The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table 5 shows the WDSVR register.

Table 5 WATCHDOG Service Register

Window Select		Key Data					Clock Monitor
X	X	0	1	1	0	0	Y
7	6	5	4	3	2	1	0

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table 6 shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

Table 6 WATCHDOG Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2048–8k t_c Cycles
0	1	2048–16k t_c Cycles
1	0	2048–32k t_c Cycles
1	1	2048–64k t_c Cycles

Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ($1/t_c$) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

WATCHDOG/Clock Monitor Operation

The WATCHDOG is enabled by bit 2 of the ECON register. When this ECON bit is 0, the WATCHDOG is enabled and pin G1 becomes the WATCHDOG output with a weak pullup.

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table 7 shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin has a weak pullup in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16 tc–32 t_c cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low.The WATCHDOG service window will restart when the WDOUT pin goes high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will go high.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will go high following 16 t_c–32 t_c clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_c$ > 10 kHz—No clock rejection.

$1/t_c$ < 10 Hz—Guaranteed clock rejection.

WATCHDOG And Clock Monitor Summary

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having he maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.
- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator option selected, or with the single-pin R/C oscillator option selected and the CLKDLY bit set, the WATCHDOG service window will be set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.
- The IDLE timer T0 is not initialized with external RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the T0PND flag. The T0PND flag is set whenever the twelfth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the T0PND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is 00. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4... etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

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

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 that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPND instruction.

Table 7 WATCHDOG Service Actions

Key Data	Window Data	Clock Monitor	Action
Match	Match	Match	Valid Service: Restart Service Window
Don't Care	Mismatch	Don't Care	Error: Generate WATCHDOG Output
Mismatch	Don't Care	Don't Care	Error: Generate WATCHDOG Output
Don't Care	Don't Care	Mismatch	Error: Generate WATCHDOG Output

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 drivers, E2PROMs 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 14 shows a block diagram of the MICROWIRE/PLUS logic.

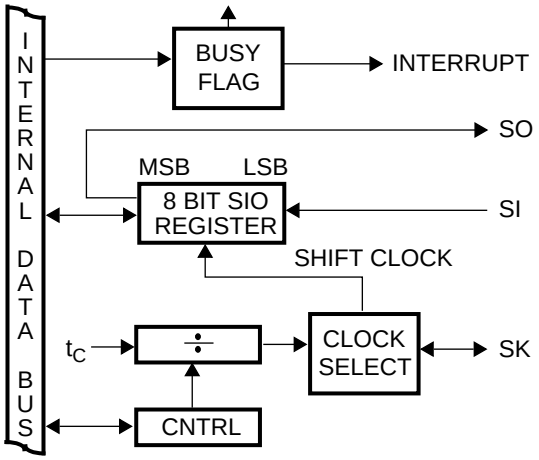


Figure 14. MICROWIRE/PLUS Block Diagram

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

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table 8 details the different clock rates that may be selected.

Table 8 MICROWIRE/PLUS Master Mode Clock Selection

SL1	SL0	SK
0	0	2 X t_c
0	1	4 X t_c
1	x	8 X t_c

Where t_c 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. If enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. Figure 15 shows how two microcontroller devices and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.

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. SK clock is normally low when not shifting.

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.

MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table 9 summarizes the bit settings required for Master mode of operation.

MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK 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 is selected as an output pin by setting and resetting the appropriate bit in the Port G configuration register. Table V summarizes the settings required to enter the Slave mode of operation.

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 properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock in the normal mode. In the alternate SK phase mode the SIO register is shifted on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

Table 9 MICROWIRE/PLUS Mode Settings

G4 (SO) Config. Bit	G5 (SK) Config. Bit	G4 Fun.	G5 Fun.	Operation
1	1	SO	Int. SK	MICROWIRE/PLUS Master
0	1	TRI-STATE	Int. SK	MICROWIRE/PLUS Master
1	0	SO	Ext. SK	MICROWIRE/PLUS Slave
0	0	TRI-STATE	Ext. SK	MICROWIRE/PLUS Slave
This table assumes that the control flag MSEL is set.				

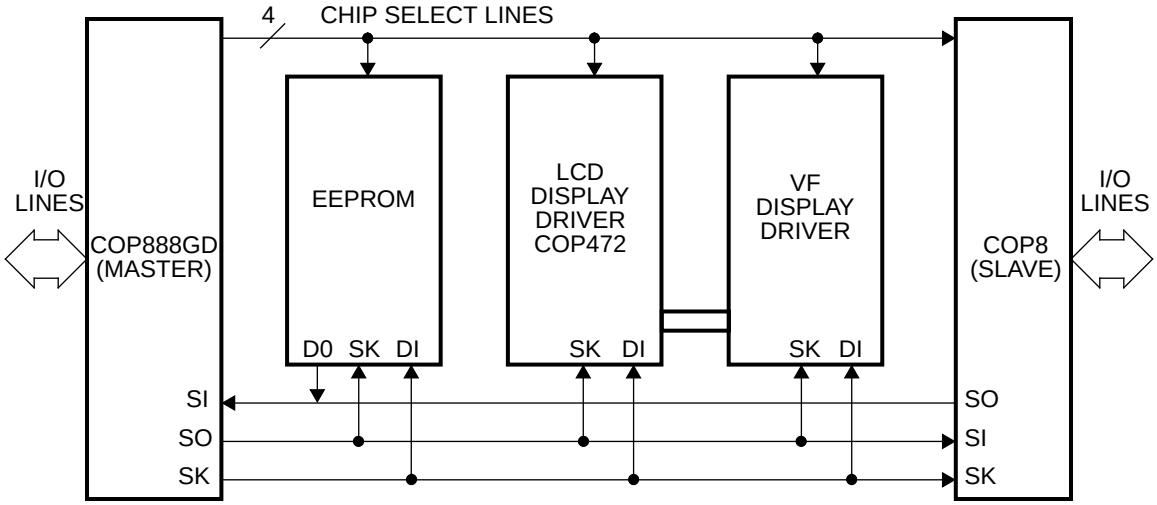


Figure 15. MICROWIRE/PLUS Application

Memory Map

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

Address S/ADD REG	Contents
0000 to 006F	On-Chip RAM bytes (112 bytes)
0070 to 007F	Unused RAM Address Space (Reads As All Ones)
xx80 to xxAF	Unused RAM Address Space (Reads Undefined Data)
xxB0	Timer T3 Lower Byte
XXB1	Timer T3 Upper Byte
xxB2	Timer T3 Autoload Register T3RA Lower Byte
xxB3	Timer T3 Autoload Register T3RA Upper Byte
xxB4	Timer T3 Autoload Register T3RB Lower Byte
xxB5	Timer T3 Autoload Register T3RB Upper Byte
xxB6	Timer T3 Control Register
xxB7 to xxBF	Reserved
xxC0	Timer T2 Lower Byte
xxC1	Timer T2 Upper Byte
xxC2	Timer T2 Autoload Register T2RA Lower Byte
xxC3	Timer T2 Autoload Register T2RA Upper Byte
xxC4	Timer T2 Autoload Register T2RB Lower Byte
xxC5	Timer T2 Autoload Register T2RB Upper Byte
xxC6	Timer T2 Control Register
xxC7	WATCHDOG Service Register (Reg:WDSVR)
xxC8	MIWU Edge Select Register (Reg:WKEDG)
xxC9	MIWU Enable Register (Reg:WKEN)
xxCA	MIWU Pending Register (Reg:WKPND)
xxCB	A/D Convertor Control Register (Reg: ENAD)
xxCC	A/D Convertor Result Register (Reg: ADRSLT)
xxCD to xxCE	Reserved
xxCF	IDLE Timer Control Register (Reg: ITMR)
xxD0	Port L Data Register
xxD1	Port L Configuration Register
xxD2	Port L Input Pins (Read Only)
xxD3	Reserved for Port L
xxD4	Port G Data Register
xxD5	Port G Configuration Register
xxD6	Port G Input Pins (Read Only)
xxD7	Port I Input Pins (Read Only)
xxD8	Port C Data Register
xxD9	Port C Configuration Register
xxDA	Port C Input Pins (Read Only)
xxDB	Reserved for Port C
xxDC	Port D
xxDD to DF	Reserved for Port D
xxE0 to xxE5	Reserved
xxE6	Timer T1 Autoload Register T1RB Lower Byte
xxE7	Timer T1 Autoload Register T1RB Upper Byte
xxE8	ICNTRL Register
xxE9	MICROWIRE/PLUS Shift Register
xxEA	Timer T1 Lower Byte
xxEB	Timer T1 Upper Byte
xxEC	Timer T1 Autoload Register T1RA Lower Byte
xxED	Timer T1 Autoload Register T1RA Upper Byte
xxEE	CNTRL Control Register
xxEF	PSW Register
xxF0 to FB	On-Chip RAM Mapped as Registers
xxFC	X Register
xxFD	SP Register
xxFE	B Register
xxFF	S Register
0100–017F	On-Chip 128 RAM Bytes

Reading memory locations 0070H–007FH (Segment 0) will return all ones. Reading unused memory locations 0080H–00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 2, Segment 3, ... etc.) will return all ones.

Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

See the BYTES and CYCLES per INSTRUCTION table for details.

Bytes and Cycles per Instruction

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

Arithmetic and Logic Instructions

	[B]	Direct	Immed.
ADD	1/1	3/4	2/2
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
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	

RPND	1/1

Instructions Using A & C

CLRA	1/1
INCA	1/1
DECA	1/1
LAID	1/3
DCORA	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

Transfer of Control Instructions

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

Memory Transfer Instructions

	Register Indirect		Direct	Immed.	Register Indirect Auto Incr & Decr		
	[B]	[X]			[B+, B-]	[X+, X-]	
X A,*	1/1	1/3	2/3		1/2	1/3	
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3	
LD B,Imm				1/1			(If B < 16)
LD B,Imm				2/2			(If B > 15)
LD Mem,Imm	2/2		3/3		2/2		
LD Reg,Imm			2/3				
IFEQ MD,Imm			3/3				

* => Memory location addressed by B or X or directly.

Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the “normal” addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

Register Indirect (with auto post increment or decrement of pointer)

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

This addressing mode is used with the Load B Immediate instruction. 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 get the new program location. JP has a range from –31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no “pages” when using JP, since all 15 bits of 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 current 4k 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 serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

Note: The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

Instruction Set

Register and Symbol Definition

Registers	
A	8-Bit Accumulator Register
B	8-Bit Address Register
X	8-Bit Address Register
S	8-Bit Segment Register
SP	8-Bit Stack Pointer 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
VU	Interrupt Vector Upper Byte
VL	Interrupt Vector Lower Byte
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]
Meml	Direct Addressed Memory or [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

INSTRUCTION SET

ADD	A,Meml	ADD	A ← A + Meml
ADC	A,Meml	ADD with Carry	A ← A + Meml + C, C ← Carry, HC ← Half Carry
SUBC	A,Meml	Subtract with Carry	A ← A - MemI + C, C ← Carry, HC ← Half Carry
AND	A,Meml	Logical AND	A ← A and $\overline{\text{Meml}}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if (A and Imm) = 0
OR	A,Meml	Logical OR	A ← A or Meml
XOR	A,Meml	Logical EXclusive OR	A ← A xor Meml
IFEQ	MD,Imm	IF EQual	Compare MD and Imm, Do next if MD = Imm
IFEQ	A,Meml	IF EQual	Compare A and Meml, Do next if A = Meml
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if A Meml
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if A > Meml
IFBNE	#	If B Not Equal	Do next if lower 4 bits of B Imm
DRSZ	Reg	Decrement Reg., Skip if Zero	Reg ← Reg - 1, Skip if Reg = 0
SBIT	#,Mem	Set BIT	1 to bit, Mem (bit = 0 to 7 immediate)
RBIT	#,Mem	Reset BIT	0 to bit, Mem
IFBIT	#,Mem	IF BIT	If bit in A or Mem is true do next instruction
RPND		Reset PeNDing Flag	Reset Software Interrupt Pending Flag
X	A,Mem	EXchange A with Memory	A ↔ Mem
X	A,[X]	EXchange A with Memory [X]	A ↔ [X]
LD	A,Meml	LoaD A with Memory	A ← Meml
LD	A,[X]	LoaD A with Memory [X]	A ← [X]
LD	B,Imm	LoaD B with Immed.	B ← Imm
LD	Mem,Imm	LoaD 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)
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 [B] Immed.	[B] ← Imm, (B ← B 1)
CLR	A	CLeaR A	A ← 0
INC	A	INCrement A	A ← A + 1
DEC	A	DECrementA	A ← A - 1
LAID		Load A InDirect from ROM	A ← ROM (PU,A)
DCOR	A	Decimal CORrect A	A ← BCD correction of A (follows ADC, SUBC)
RRC	A	Rotate A Right thru C	C → A7 → ... → A0 → C
RLC	A	Rotate A Left thru C	C ← A7 ←... ← A0 ← C
SWAP	A	SWAP nibbles of A	A7...A4 ↔ A3...A0
SC		Set C	C ← 1, HC ← 1
RC		Reset C	C ← 0, HC ← 0
IFC		IF C	IF C is true, do next instruction
IFNC		IF Not C	If C is not true, do next instruction
POP	A	POP the stack into A	SP ← SP + 1, A ← [SP]
PUSH	A	PUSH A onto the stack	[SP] ← A, SP ← SP - 1
VIS		Vector to Interrupt Service Routine	PU ← [VU], PL←[VL]
JMPL	Addr.	Jump absolute Long	PC ← ii (ii = 15 bits, 0 to 32k)
JMP	Addr.	Jump absolute	PC9...0 ← i (i = 12 bits)
JP	Disp.	Jump relative short	PC ← PC + r (r is -31 to +32, except 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, PC9...0 ← i
JID		Jump InDirect	PL←ROM (PU,A)
RET		RETurn from subroutine	SP + 2, PL← [SP], PU← [SP-1]
RETSK		RETurn and SKip	SP + 2, PL← [SP],PU← [SP-1]
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

Opcode TableMask Options

UPPER NIBBLE																LOWER NIBBLE
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	ADC A,#i	ADC A,[B]	IFBIT 0,[B]	ANDSZ A,#i	LD B,#0F	IFBNE 0	JSR x000-x0FF	JMP x000-x0FF	JP+17	INTR	0
JP-14	JP-30	LD 0F1,#i	DRSZ 0F1	*	SC	SUBC A,#i	SUBC A,[B]	IFBIT 1,[B]	*	LD B,#0E	IFBNE 1	JSR x100-x1FF	JMP x100-x1FF	JP+18	JP + 2	1
JP-13	JP-29	LD 0F2,#i	DRSZ 0F2	X A,[X+]	X A,[B+]	IFEQ A,#i	IFEQ A,[B]	IFBIT 2,[B]	*	LD B,#0D	IFBNE 2	JSR x200-x2FF	JMP x200-x2FF	JP+19	jp + 3	2
JP-12	JP-28	LD 0F3,#i	DRSZ 0F3	X A,[X-]	X A,[B-]	IFGT A,#i	IFGT A,[B]	IFBIT 3,[B]	*	LD B,#0C	IFBNE 3	JSR x300-x3FF	JMP x300-x3FF	JP+20	JP + 4	3
JP-11	JP-27	LD 0F4,#i	DRSZ 0F4	VIS	LAID	ADD A,#i	ADD A,[B]	IFBIT 4,[B]	CLRA	LD B,#0B	IFBNE 4	JSR x400-x4FF	JMP x400-x4FF	JP+21	JP = 5	4
JP-10	JP-26	LD 0F5,#i	DRSZ 0F5	RPND	JID	AND A,#i	AND A,[B]	IFBIT 5,[B]	SWAPA	LD B,#0A	IFBNE 5	JSR x500-x5FF	JMP x500-x5FF	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,#09	IFBNE 6	JSR x600-x6FF	JMP x600-x6FF	JP+23	JP + 7	6
JP-8	JP-24	LD 0F7,#i	DRSZ 0F7	*	*	OR A,#i	OR A,[B]	IFBIT 7,[B]	PUSHA	LD B,#08	IFBNE 7	JSR x700-x7FF	JMP x700-x7FF	JP+24	JP + 8	7
JP-7	JP-23	LD 0F8,#i	DRSZ 0F8	NOP	RLCA	LD A,#i	IFC	SBIT 0,[B]	RBIT 0,[B]	LD B,#07	IFBNE 8	JSR x800-x8FF	JMP x800-x8FF	JP+25	JP + 9	8
JP-6	JP-22	LD 0F9,#i	DRSZ 0F9	IFNE A,[B]	IFEQ Md,#i	IFNE A,#i	IFNC	SBIT 1,[B]	RBIT 1,[B]	LD B,#06	IFBNE 9	JSR x900-x9FF	JMP x900-x9FF	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,#05	IFBNE 0A	JSR xA00-xAFF	JMP xA00-xAFF	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,#04	IFBNE 0B	JSR xB00-xBFF	JMP xB00-xBFF	JP+28	JP + 12	B
JP-3	JP-19	LD 0FC,#i	DRSZ 0FC	LD Md,#i	JMPL	X A,Md	POPA	SBIT 4,[B]	RBIT 4,[B]	LD B,#03	IFBNE 0C	JSR xC00-xCFF	JMP xC00-xCFF	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,#02	IFBNE 0D	JSR xD00-xDFF	JMP xD00-xDFF	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,#01	IFBNE 0E	JSR xE00-xEFF	JMP xE00-xEFF	JP+31	JP + 15	E
JP-0	JP-16	LD 0FF,#i	DRSZ 0FF	*	*	LD B,#i	RETI	SBIT 7,[B]	RBIT 7,[B]	LD B,#00	IFBNE 0F	JSR xF00-xFFF	JMP xF00-xFFF	JP+32	JP + 16	F

where,

i is the immediate data
Md is a directly addressed memory location
* is an unused opcode

The opcode 60 Hex is also the opcode for IFBIT #i,A

The mask programmable options are shown below. The options are programmed at the same time as the ROM pattern submission.

```
OPTION 1: CLOCK CONFIGURATION
    = 1   Crystal Oscillator (CKI/10)
          G7 (CKO) is clock generator output
          to crystal/resonator CKI is the
          clock input
    = 2   Single-pin RC controlled oscillator
          (CKI/10)
          G7 is available as a HALT restart
          and/or general purpose input
OPTION 2: HALT
    = 1   Enable HALT mode
    = 2   Disable HALT mode
OPTION 3: BONDING OPTIONS
    = 1   44-Pin PLCC
```

Development Support

Overview

National is engaged with an international community of independent 3rd party vendors who provide hardware and software development tool support. Each vendor brings an individual expertise to create a precision development tool. Through National's interaction and guidance, these tools cooperate to form a usable tool suite that fits the developer's needs.

This section provides a summary of the tool suite. Tools are added and improved continuously. By utilizing the on-line World Wide Web for information, every user can keep current. Search at www.cop8.com.

Following sections will provide a feature summary of tools available at publication, order information for each tool, and finally provide the URL link paths to enable every user to remain current.

Summary of Tools by Category

- Project Definition and Maintenance Tools:
  - — Aisys Corporation's DriveWay-COP8 is a Windows based project development tool that will automatically generate documented C or Assembly source code modules containing customized I/O drivers, interrupt handlers and main program shell to meet application requirements. Application specific code can be inserted using the integrated editor. Language and emulation tools can be launched directly from DriveWay-COP8.
  - — K&K Development's WCOP8 IDE is a Windows based integrated development tool that can be used to define and edit source language modules associated to the project. Code language development and emulation tools can be launched directly from the project window framework.
- Language Tools:
  - — COP8 Assembly Language Development Software Kit provides DOS based macro assembler, linker, libraries and utility functions for COP8 software cross development on a PC platform.
  - — Bytecraft's COP8C is a C language Software Development Kit that provides a DOS based IDE, editor, C compiler, linker and utility functions for COP8 software cross development on a PC platform.
- COP8 Emulation & Simulation Tools:
  - — MetaLink's iceMASTER™: IM-COP8/400 -- Full featured in-circuit emulation system for all COP8 products. A full set of COP888GD package specific probes and surface mount adaptors are available to emulate the COP888GD part family directly on application hardware.
  - — COP8 Debug Module: COP8-DM/888GD -- Moderate cost in-circuit emulation and development programming unit.
  - — COP8 Evaluation & Programming Unit: EPU-COP888GG-1/2 -- A low cost In-circuit simulation and development programming unit, not fully compatable to the COP888GD.
  - — COP8 Instruction Level Simulator -- a non-hardware platform for evaluating and testing software algorithms.
- OTP/EPROM Programmer Support: Covering needs from engineering prototype, pilot production to full production environments.
- In-factory Programming Support: Covering high volume production OTP programming.

DriveWay-COP8 Device Driver Design Environment (3DE).

Aisys Corporation is a company dedicated to Window based development tools that are intuitive, easy to use and remove much of the burden to acquire detail component knowledge in order to create working, production worthy, control firmware for an application. DriveWay-COP8 provides a quick and dependable method to develop efficient software functions that correctly initialize the COP8 hardware and provide callable functions to access and control the hardware. Documented and pretested source code modules are generated automatically after the user makes a modest set of selections from the menus presented. DriveWay-COP8 has a built in Knowledge Base that provides the processor and peripheral definition details to the software that enables the first time generation of working code. Included with the code generation functionality is an extensive on-line help facility that provides the guidance necessary in making choices and also provides an interactive context linked copy of the COP888EG data sheet for on-line reference. Not only does the software generate end-application code modules but there are options to generate test driver software as well. The mysteries of proper initialization to obtain the exact functional behavior are avoided. In a short interactive session, functional prototyping of control software can be accomplished quickly and accurately.

Starting from a design concept or block diagram sketch, a firmware engineer makes simple choices:

- Select the COP888EG model and package, EG is not fully compatible.
- Define the oscillator frequency and other configuration parameters, such as WATCHDOG enable, to configure the core chip elements.
- Using the interactive data sheet's (IDS) chip block diagram, select the I/O port and alternate peripheral func-

tions to be enabled and to define application functional pin usage.

- For each function, select and define the parameters for correct initialization. The IDS may be consulted for a definition of parameters, and provides guidance to make good choices. As an example, a PWM timer's characteristics are defined by its frequency and duty cycle in user friendly units. The hardware setup values are then generated automatically based on the oscillator frequency. Change the global oscillator frequency and all of the code objects will have the parameters adjusted correctly without requiring redefinition. The function to initialize the timer will be generated automatically and linked to the initialization start-up code.
- Select language tool: Assembly or C. DriveWay-COP8 will generate proper syntax for this selection.
- Select or disable optional test code generation.
- At click of the mouse button, generate include files and source code files.
- Display the source lines in the integrated EDIT window and add the application specific code statements to link input stimulus to output actions. Code lines edited within well marked 'save' points are preserved from generation to generation within the project.
- The entire application, or early-on application segment, is ready for compile, link and test using the language and emulation tools.
- Generated code is well documented, automatically as well. A TEXT file documenting the project is likewise automatically generated. All generated code has been extensively pre-tested by Aisys on COP8 emulation tools.

DriveWay-COP8 Order Information

Additional information about Aisys is available on the WEB and DriveWay-COP8 may be ordered directly from Aisys Corporation. Demo software is available for download.

Aisys Corporation	
Phone	**URL/email**
Santa Clara: 408-327-8820 Boston: 617-270-7430 Israel: 972-3-9226863	www.aisys.co.il/Products/cop8.html info@aisys.co.il

WCOP8 IDE Integrated Development Environment

WCOP8 IDE is a Windows based COP8 Integrated Development Environment that serves as an interface to the COP8 Language and Emulation development tools. The edit function for code and documentation development is integrated. Also available is a "Project Wizard" that provides a simple means to link the development tools into a project file, which contains pointers to source files, and locates/defines the development tool usage. The Build and Make functions are automatically invoked by the point and click method. Compiler / assembler errors are linked to the editor for correction. The emulation tools are also invoked by the simple point and click method for start-up. The DOS environment used by these tools is transparent.

WCOP8 IDE Order Information

Additional information about K&K Development is available on the WEB. User documentation and a zero cost 30 day evaluation copy of WCOP8 IDE may be downloaded directly from the WEB. Current product release status, product and ordering information are all available at the one URL.

K&K Development	
URL	**email**
www.kkd.dk	kkd@dk-online.dk

COP8 Assembler/linker Software Development Tool Kit

National offers a relocatable COP8 macro cross assembler, linker, librarian and utility software development tool kit. Features are summarized as follows:

- Basic and Feature Family instruction set by "device" type.
- Nested macro capability.
- Extensive set of assembler directives.
- Supported on PC/DOS platform.
- Generates National standard COFF output files.
- Integrated Linker & Librarian.
- Integrated utilities to generate ROM code file outputs.
- DUMPCOFF utility.

This product is integrated with MetaLink tools as a development kit, fully supported by the MetaLink debugger. The assembler is likewise supported by DriveWay-COP8 and WCOP8 IDE. The assembler may be downloaded from the WEB or may be may be ordered separately. It is bundled with MetaLink products at no additional cost.

COP8 Assembler/Linker Order Information

Assembler SDK:	
COP8-DEV-IBMA	Assembler SDK on installable 3.5" PC®/DOS Floppy Disk Drive format. Periodic upgrades and most recent version is available on National's Web site.
URL for latest information.	www.national.com/cop8

COP8 C Compiler Software Development Tool

A C Compiler is developed and marketed by Byte Craft Limited. The COP8C compiler is a fully integrated development tool specifically designed to support the compact embedded configuration of the COP8 family of products.

Features are summarized as follows:

- DOS based IDE with editor for on-line development
- COP8 header include files and libraries
- Byte Craft linker; links objects C source or Assembly source, compiled by COP8C
- ANSI C with some restrictions and extensions that optimize development for the COP8 embedded application.
- BITS data type extension. Register declaration #pragma with direct bit level definitions.
- C language support for interrupt routines.
- Expert system, rule based code generation and optimization.
- Performs consistency checks against the architectural definitions of the target COP8 device.
- Generates program memory code.
- Global optimization of linked code.

- Symbolic debug load format fully source level supported by the MetaLink debugger software
- Tested source generated by Aisys' DriveWay-COP8.

COP8C Order Information

Byte Craft Limited	
Phone	**URL**
Canada: 519-888-6911	www.bytecraft.com/cop8c.html

IceMASTER (IM) In-circuit Emulation

The iceMASTER IM-COP8/400 is a PC DOS based full feature in-circuit emulation tool developed and marketed by MetaLink Corporation to support the whole COP8 family of products. National and it's Authorized Distributors are resale vendors for these products.

See Figure 16 for configuration.

The iceMASTER IM-COP8/400 with its device specific COP8 Probe and provides a rich feature set for developing, testing and maintaining product:

- Real-time in-circuit emulation; full 2.5-5.5V operation range, full DC-10 MHz clock. Chip options are programmable or jumper selectable.
- Direct connection to application board by package compatible socket or surface mount assembly.
- Full 32K byte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated on the probe as necessary.
- Full 4k frame synchronous trace memory. Address, instruction, and eight unspecified, circuit connectable trace lines. Display can be HLL source (e.g., C source), assembly or mixed.
- A full 64k hardware configurable break, trace on, trace off control, and pass count increment events.
- Tool set integrated interactive symbolic debugger - supports both assembler (COFF) and C Compiler (.COD) linked object formats.
- Real time performance profiling analysis; selectable bucket definition.
- Watch windows, content updated automatically at each execution break.
- Instruction by instruction memory/register changes displayed on source window when in single step operation.
- Single base unit & debugger software reconfigurable to support the entire COP8 family; only the probe personality needs to change. Debugger software is processor customized, and reconfigured from a master model file.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- On-line HELP.
- Includes a copy of COP8-DEV-IBMA assembler & linker SDK.

IceMASTER Order Information

MetaLink Corporation	
Phone	**URL/email**
Chandler, AZ 800-638-2423 602-926-0797	www.metaice.com sales@metaice.com
National's NSID	**Description**
IM-COP8/400-1	iceMASTER base unit, 110V Power Supply
IM-COP8/400-2	iceMASTER base unit, 220V Power Supply
iceMASTER Probe Card, COP888GD	
MHW-888GD44PWPC	44 PLCC, 2.5 – 5.5V

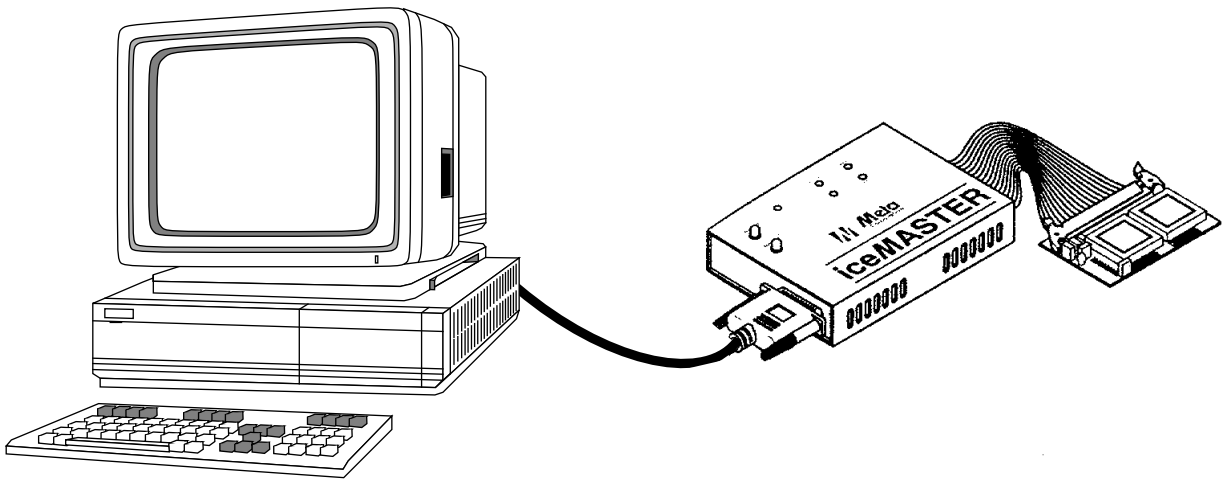


Figure 16. COP8 IceMASTER Environment

IceMASTER Debug Module (DM)

The iceMASTER Debug Module is a combination in-circuit emulation and COP8 OTP/EPROM programming tool which is bundled with MetaLink's PC based, DOS compatible debugger software and National's assembler SDK. The DM products are marketed by MetaLink Corporation to support the COP8 family of products. National and it's Authorized Distributors are resale vendors for these products.

See Figure 17 for configuration.

The iceMASTER Debug Module is a moderate cost development tool. It has the capability of in-circuit emulation for a specific COP8 microcontroller and in addition serves as a programming tool for COP8 OTP and EPROM product families. Summary of features is as follows:

- Real-time in-circuit emulation; full operating voltage range operation, full DC-10MHz clock.
- All processor I/O pins can be cabled to an application development board with package compatible cable to socket and surface mount assembly.
- Full 32 kbyte of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- 100 frames of synchronous trace memory. The display can be HLL source (C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- configured break points; uses INTR instruction which is modestly intrusive.
- software - only supported features are selectable.
- Tool set integrated interactive symbolic debugger - supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification.
- Programming hardware supports all SOIC, DIP and PLCC packages.
- Program menu and algorithm supports full product line of programmable OTP and EPROM COP8 products. Program data is taken directly from the overlay RAM.
- Includes wall mount power supply
- Includes a copy of COP8-DEV-IBMA assembler & linker SDK.

Debug Module Order Information

MetaLink Corporation	
Phone	**URL/email**
Chandler, AZ 800-638-2423 602-926-0797	www.metaice.com sales@metaice.com
National's NSID	**Description**
COP8-DM/888GD	Debug Module
Cable Assembles, requires one for emulation. *The flagged items are included, 1 each, with a DM; others, as required, need to be ordered separately.	
*DM-COP8/44P	44 PLCC

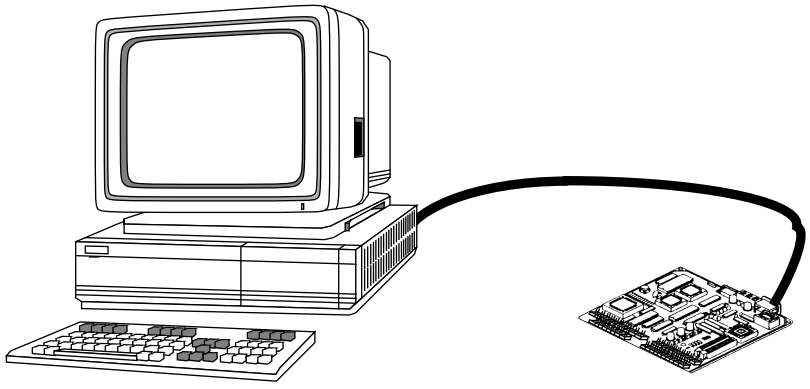


Figure 17. COP8-DM Environment

IceMASTER Evaluation Programming Unit (EPU)

The iceMASTER EPU-COP888GG is a PC based in-circuit hardware simulation tool bundled with the MetaLink DOS based debugger software to support the COP888xG products for evaluation and limited development.The GD is not pin function compatible.

See Figure 18 for configuration.

The simulation capability is a very low cost means of evaluating the general COP888xG architecture. In addition, the EPU has programming capability, with added adapters, for programming the whole COP888xG product family of OTP and EPROM products. The product includes the following features:

- Non-real-time in-circuit simulation. Program overlay memory is PC resident; instructions are downloaded over RS-232 as executed. Approximate performance is 20KHz.
- Includes a 40 pin DIP cable adapter for direct connection to a target with 40-pin DIP socket. Other target packages are not supported. All processor I/O pins are cabled to the application development environment.
- **Full 32 kbyte** of loadable programming space that overlays (replaces) the on-chip ROM or EPROM. On-chip RAM and I/O blocks are used directly or recreated as necessary.
- On-chip timer and watch-dog execution are NOT synchronized to the instruction simulation.
- 100 frames of synchronous trace memory. The display can be HLL source (e.g., C source), assembly or mixed. The most recent history prior to a break is available in the trace memory.
- Up to eight software configured break points; uses INTR instruction which is modestly intrusive.
- Common look-feel debugger software across all MetaLink products - only supported features are selectable.
- Tool set integrated interactive symbolic debugger - supports both assembler (COFF) and C Compiler (.COD) SDK linked object formats.
- Instruction by instruction memory/register changes displayed when in single step operation.
- Processor specific symbolic display of registers and bit level assignments, configured from master model file.
- Halt/Idle mode notification. Restart requires special handling.
- Programming menu supports full product line of programmable OTP and EPROM COP8 products. Only a 40 ZIF socket is available on the EPU unit. Adapters are available for other part package configurations.
- Includes wall mount power supply.
- Includes a copy of COP8-DEV-IBMA assembler, linker SDK.

EPU Order-Information

MetaLink Corporation	
Phone	**URL/email**
Chandler, AZ 800-638-2423 602-926-0797	www.metaice.com sales@metaice.com
National's NSID	**Description**
EPU-COP888GG-1/2	Evaluation Programming Unit with debugger and programmer control software with 40 pin ZIF programming socket. Programs 40-pin DIP COP87Lxxx.
Optional Programming Adapters	
COP8SA-PGMA	44 PLCC, 28, 20 and 16 SOIC, and 28, and 20 DIP

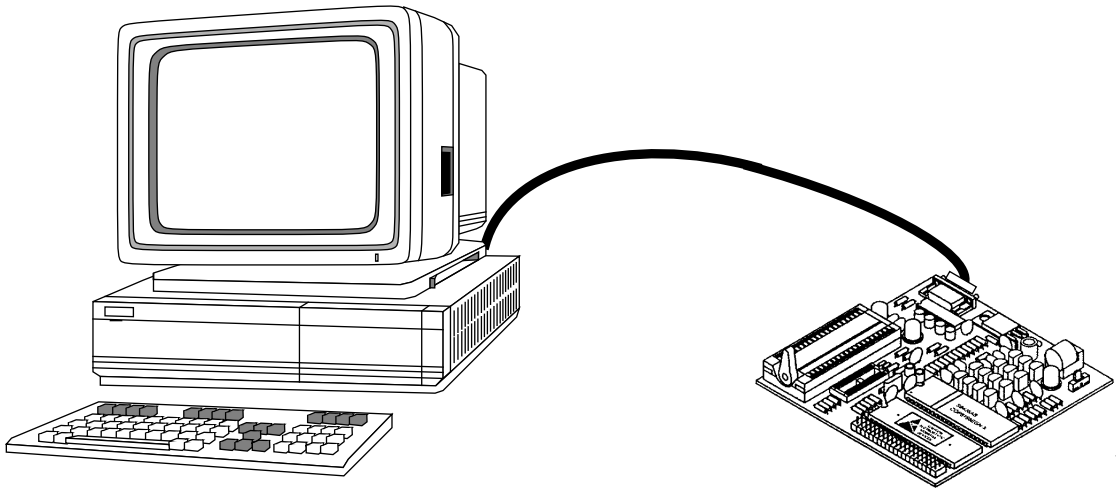


Figure 18. EPU-COP8 Tool Environment

iceMASTER Debugger with Software Simulation

This is a software debugger tool that uses the common iceMASTER PC DOS based debugger to provide instruction level debugging. Setup and configuration are common to all MetaLink tools. The simulator is instruction level only and does not simulation I/O or interrupt behavior. Available at no charge from National's COP8 web site:

www.cop8.com

Industry Wide OTP / EPROM Programming Support

Programming support, in addition to the MetaLink development tools, is provided by a full range of independent approved vendors to meet the needs from the engineering laboratory to full production. Updated information can be obtained from our web site at:

www.cop8.com

Approved List:

Manufacturer URL/email	Home Office Location Telephone/Fax/bbs
BP Microsystems (USA) www.bpmicro.com sales@bpmicro.com	USA: Houston, TX 800-225-2102 713-688-4600 Fax : 713-688-0920 bbs: 713-688-9283
Data I/O (USA) www.data-io.com sales@data-io.com techhelp@data-io.com	USA: Redmon, WA 800-426-1045 206-881-6444 Fax: 206-882-1043 bbs: 206-882-3211
HI - LO (Taiwan) www.hilosystems.com.tw hilosys@fit.ivnet.com.tw	TAI: Taipei 2-764-0215 Fax: 2-756-6403 bbs: 2-769-0881
ICE Technology (UK) www.icetech.com sales@icetech.com	UK: Penistone, S. York 0-1226-767404 Fax: 0-1226-370434 Faxback:0-1226-761844 bbs: 0-1226-761181
MetaLink (USA) www.metaice.com sales@metaice.com	USA: Chandler, AZ 800-638-2423 602-926-0797 Fax: 602-693-0681
Needhams (USA) www.needhams.com support@needhams.com	USA: Sacramento, CA 916-924-8037 Fax: 916-924-8065
SMS (Germany) www.sms-sprint.com info@sms-sprint.com	Fax: 7522-9728-88
Stag (UK) www.stagdev.com stagdev@henge.com	Fax: 01707-371503
System General (Taiwan) www.sg.com.tw sales@sg.com.tw	TAI: Taipei 2-917-3005 Fax: 2-911-1283 bbs: 2-918-1076
Xeltek (USA) www.xeltec.com info@xeltec.com	USA: Sunnyvale, CA 408-524-1929 Fax: 408-245-7084 bbs: 408-245-7082

Available Literature

For more information, please see the COP8 Feature Family User's Manual, Literature Number 620897, and National's Family of 8-bit Microcontrollers COP8 Selection Guide, Literature Number 630006. Literature can be obtained from National's web site at:

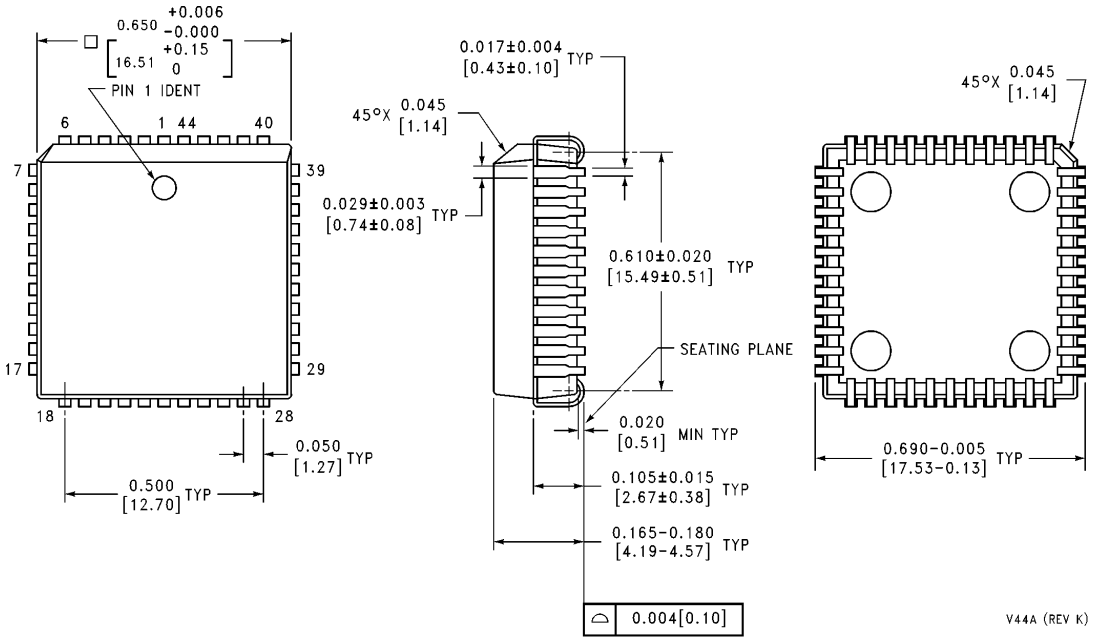
www.cop8.com

Customer Response Center

Complete product information and technical support is available from National's customer response centers.

CANADA/U.S.:	Tel:	(800) 272-9959
	email:	support@tevm2.nsc.com
EUROPE:	email:	europe.support@nsc.com
	Deutsch Tel:	+49 (0) 180-530 85 85
	English Tel:	+49 (0) 180-532 78 32
JAPAN:	Tel:	+81-043-299-2309
S.E. ASIA:	Singapore Tel:	(+65) 254-4466
	email:	sea.support@nsc.com
AUSTRALIA:	Tel:	(+61) 3-9558-9999
INDIA:	Tel:	(+91) 80-559-9467

Physical Diamensions inches (millimeters) unless otherwise noted



Plastic Leaded Chip Carrier (V)
Order Number COP888GD-XXX/V or COP988GD-XXX/V
See NS Package Number V44A

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 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
Tel: 1-800-272-9959
Fax: 1-800-737-7018
Email: support@nsc.com

National Semiconductor Europe
Fax: (+49) 0-180-530 85 86
Email: europe.support@nsc.com
Deutsch Tel: (+49) 0-180-530 85 85
English Tel: (+49) 0-180-532 78 32

National Semiconductor Asia Pacific Customer Response Group
Tel: 65-254-4466
Fax: 65-250-4466
Email: sea.support@nsc.com

National Semiconductor Japan Ltd.
Tel: 81-3-5620-6175
Fax: 81-3-5620-6179

www.national.com