

COP888CL Single-Chip microCMOS Microcontroller

General Description

The COP888 family of microcontrollers uses an 8-bit single chip core architecture fabricated with National Semiconductor's M²CMOS™ process technology. The COP888CL is a member of this expandable 8-bit core processor family of microcontrollers. (Continued)

Features

- Low cost 8-bit microcontroller
- Fully static CMOS, with low current drain
- Two power saving modes: HALT and IDLE
- 1 μ s instruction cycle time
- 4096 bytes on-board ROM
- 128 bytes on-board RAM
- Single supply operation: 2.5V–6V
- MICROWIRE/PLUSTM serial I/O
- WatchDog and Clock Monitor logic
- Idle Timer
- Multi-Input Wakeup (MIWU) with optional interrupts (8)
- Ten multi-source vectored interrupts servicing
 - External Interrupt
 - Idle Timer T0
 - Timers TA, TB (Each with 2 Interrupts)
 - MICROWIRE/PLUS
 - Multi-Input Wake Up
 - Software Trap
 - Default VIS
- Two 16-bit timers, each with two 16-bit registers supporting:
 - Processor Independent PWM mode
 - External Event counter mode
 - Input Capture mode
- 8-bit Stack Pointer SP (stack in RAM)
- Two 8-bit Register Indirect Data Memory Pointers (B and X)
- Versatile instruction set
- True bit manipulation
- Memory mapped I/O
- BCD arithmetic instructions
- Package: 44 PCC or 40 N or 28 N or 28 PCC
 - 44 PCC with 39 I/O pins
 - 40 N with 33 I/O pins
 - 28 PCC or 28 N, each with 23 I/O pins
- 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
- Temperature ranges: –40°C to +85°C, –55°C to +125°C
- ROMless mode for accurate emulation and external program memory capability
- Single chip COP888CLP piggy back emulation device
- Real time emulation and full program debug offered by National's Development Systems

Block Diagram

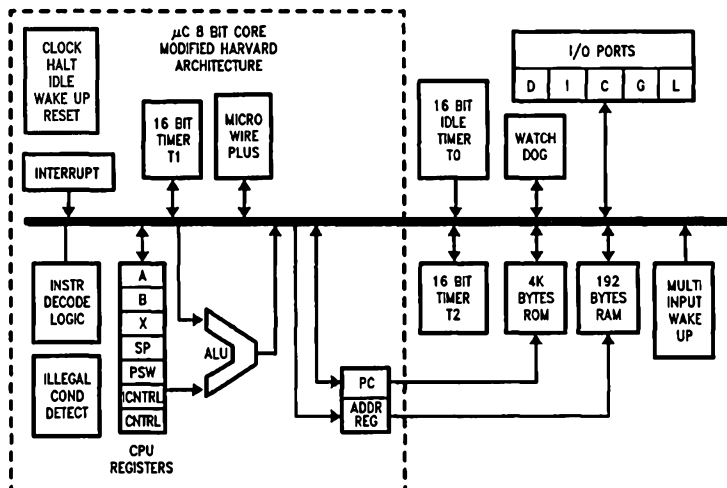


FIGURE 1. COP888CL Block Diagram

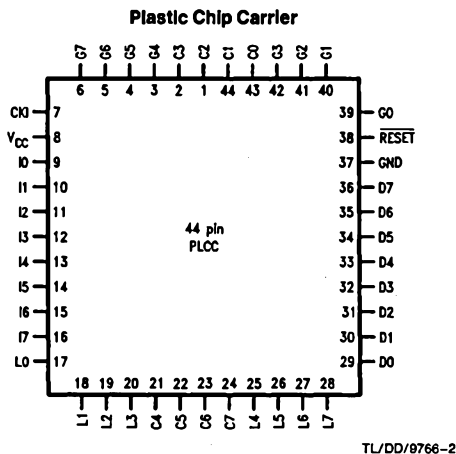
TL/DD/9766–1

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/PLUS serial I/O, two 16-bit timer/counters supporting three modes (Processor Independent PWM generation, External Event counter, and Input Capture mode capabilities), and two power savings modes (HALT and IDLE), both with a multi-sourced wakeup/interrupt capability. This multi-sourced in-

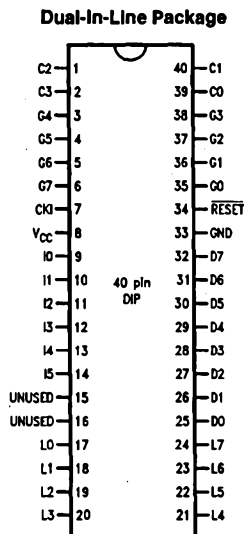
terrupt capability may also be used independent of the HALT or IDLE modes. Each I/O pin has software selectable configurations. The COP888CL operates over a voltage range of 2.5V to 6V. High throughput is achieved with an efficient, regular instruction set operating at a maximum of 1 μ s per instruction rate. The COP888CL may be operated in the ROMless mode to provide for accurate emulation and for applications requiring external program memory.

Connection Diagrams



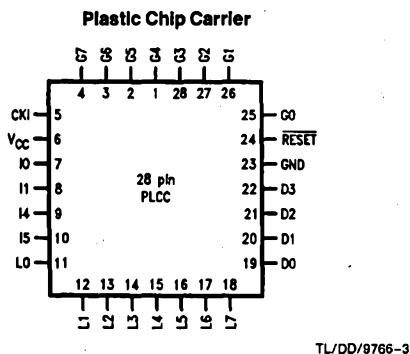
Top View

Order Number COP888CL-XXX/V
See NS Plastic Chip Package Number V44A



Top View

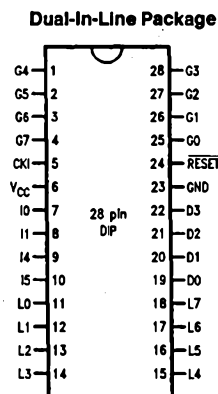
Order Number COP888CL-XXX/N
See NS Molded Package Number N40A



Top View

Order Number COP884CL-XXX/V
See NS Plastic Chip Package Number V28A

*Note: The pins labeled unused must be connected to GND.



Top View

Order Number COP884CL-XXX/N
See NS Molded Package Number N28A

FIGURE 2. COP888CL Connection Diagrams

Connection Diagrams (Continued)

COP888CL Pinouts for 28-, 40- and 44-Pin Packages

Port	Type	Alt. Fun	Alt. Fun	28-Pin Pack.	40-Pin Pack.	44-Pin Pack.
L0	I/O	MIWU	T2A T2B	11	17	17
L1	I/O	MIWU		12	18	18
L2	I/O	MIWU		13	19	19
L3	I/O	MIWU		14	20	20
L4	I/O	MIWU		15	21	25
L5	I/O	MIWU		16	22	26
L6	I/O	MIWU		17	23	27
L7	I/O	MIWU		18	24	28
G0	I/O	INT		25	35	39
G1	WDOUT			26	36	40
G2	I/O	T1B		27	37	41
G3	I/O	T1A		28	38	42
G4	I/O	SO		1	3	3
G5	I/O	SK		2	4	4
G6	I	SI		3	5	5
G7	I/CKO	HALT RESTART		4	6	6
D0	O	ROM DATA*		19	25	29
D1	O	PCL*		20	26	30
D2	O	EMUL*		21	27	31
D3	O	PCU*		22	28	32
I0	I			7	9	9
I1	I			8	10	10
I2	I				11	11
I3	I				12	12
I4	I			9	13	13
I5	I			10	14	14
I6	I					15
I7	I					16
D4	O	S CLOCK*			29	33
D5	O	HALTSEL*			30	34
D6	O	LOAD*			31	35
D7	O	D DATA*			32	36
C0	I/O				39	43
C1	I/O				40	44
C2	I/O				1	1
C3	I/O				2	2
C4	I/O					21
C5	I/O					22
C6	I/O					23
C7	I/O					24
Unused**					16	
Unused**					15	
V _{CC}				6	8	8
GND				23	33	37
CKI				5	7	7
RESET				24	34	38

* = Only in the ROMless Mode

** = On the 40-pin package Pins 15 and 16 must be connected to GND.

Absolute Maximum Ratings

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

Supply Voltage (V_{CC})	7V
Voltage at Any Pin	$-0.3V$ to $V_{CC} + 0.3V$
ESD Susceptibility (Note 4)	2000V
Total Current into V_{CC} Pin (Source)	100 mA

Total Current out of GND Pin (Sink) 110 mA
Storage Temperature Range -65°C to $+140^{\circ}\text{C}$

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 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Operating Voltage		2.5		6	V
Power Supply Ripple (Note 1)	Peak-to-Peak			$0.1 V_{CC}$	V
Supply Current (Note 2)					
CKI = 10 MHz	$V_{CC} = 6V, t_c = 1 \mu s$			15	mA
CKI = 4 MHz	$V_{CC} = 2.5V, t_c = 2.5 \mu s$			2	mA
HALT Current (Note 3)	$V_{CC} = 6V, \text{CKI} = 0 \text{ MHz}$		<1		μA
IDLE Current					
CKI = 10 MHz	$V_{CC} = 6V, t_c = 1 \mu s$			5	mA
CKI = 4 MHz	$V_{CC} = 2.5V, t_c = 2.5 \mu s$			0.6	mA
Input Levels					
RESET					
Logic High		$0.8 V_{CC}$			V
Logic Low				$0.2 V_{CC}$	V
CKI (External and Crystal Osc. Modes)					
Logic High		$0.7 V_{CC}$			V
Logic Low				$0.2 V_{CC}$	V
All Other Inputs					
Logic High		$0.7 V_{CC}$			V
Logic Low				$0.2 V_{CC}$	V
Hi-Z Input Leakage	$V_{CC} = 6V$	-2		+2	μA
Input Pullup Current	$V_{CC} = 6V$	40		250	μA
G and L Port Input Hysteresis			$0.05 V_{CC}$		V
Output Current Levels					
D Outputs					
Source	$V_{CC} = 4V, V_{OH} = 3.3V$	0.4			mA
	$V_{CC} = 2.5V, V_{OH} = 1.8V$	0.2			mA
Sink	$V_{CC} = 4V, V_{OL} = 1V$	10			mA
	$V_{CC} = 2.5V, V_{OL} = 0.4V$	2.0			mA
All Others					
Source (Weak Pull-Up Mode)	$V_{CC} = 4V, V_{OH} = 2.7V$	10		100	μA
	$V_{CC} = 2.5V, V_{OH} = 1.8V$	2.5		33	μA
Source (Push-Pull Mode)	$V_{CC} = 4V, V_{OH} = 3.3V$	0.4			mA
	$V_{CC} = 2.5V, V_{OH} = 1.8V$	0.2			mA
Sink (Push-Pull Mode)	$V_{CC} = 4V, V_{OL} = 0.4V$	1.6			mA
	$V_{CC} = 2.5V, V_{OL} = 0.4V$	0.7			mA
TRI-STATE Leakage		-2		+2	μA

Note 1: Rate of voltage change must be less than 0.5 V/ms.

Note 2: Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

Note 3: The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Test conditions: All inputs tied to V_{CC} , L and G ports in the TRI-STATE mode and tied to ground, all outputs low and tied to ground. The clock monitor is disabled.

Note 4: Human body model, 100 pF through 1500 Ω .

DC Electrical Characteristics $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ unless otherwise specified (Continued)

Parameter	Conditions	Min	Typ	Max	Units
Allowable Sink/Source Current per Pin D Outputs (Sink) All others				15 3	mA mA
Maximum Input Current without Latchup (Note 6)	$T_A = 25^{\circ}\text{C}$			± 100	mA
RAM Retention Voltage, V_r	500 ns Rise and Fall Time (Min)	2			V
Input Capacitance				7	pF
Load Capacitance on D2				1000	pF

AC Electrical Characteristics $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ unless otherwise specified

Parameter	Conditions	Min	Typ	Max	Units
Instruction Cycle Time (t_c) Crystal or Resonator	$4\text{V} \leq V_{CC} \leq 6\text{V}$	1		DC	μs
	$2.5\text{V} \leq V_{CC} < 4\text{V}$	2.5		DC	μs
R/C Oscillator	$4\text{V} \leq V_{CC} \leq 6\text{V}$	3		DC	μs
	$2.5\text{V} \leq V_{CC} < 4\text{V}$	7.5		DC	μs
CKI Clock Duty Cycle (Note 5)	$f_r = \text{Max}$	40		60	%
Rise Time (Note 5)	$f_r = 10\text{ MHz Ext Clock}$			5	ns
Fall Time (Note 5)	$f_r = 10\text{ MHz Ext Clock}$			5	ns
Inputs t_{SETUP}	$4\text{V} \leq V_{CC} \leq 6\text{V}$	200			ns
	$2.5\text{V} \leq V_{CC} < 4\text{V}$	500			ns
t_{HOLD}	$4\text{V} \leq V_{CC} \leq 6\text{V}$	60			ns
	$2.5\text{V} \leq V_{CC} < 4\text{V}$	150			ns
Output Propagation Delay $t_{\text{PD1}}, t_{\text{PD0}}$ SO, SK	$R_L = 2.2\text{k}, C_L = 100\text{ pF}$				
	$4\text{V} \leq V_{CC} \leq 6\text{V}$			0.7	μs
	$2.5\text{V} \leq V_{CC} < 4\text{V}$			1.75	μs
All Others	$4\text{V} \leq V_{CC} \leq 6\text{V}$			1	μs
	$2.5\text{V} \leq V_{CC} < 4\text{V}$			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

Note 5: Parameter sample but not 100% tested.

Note 8: Except Pin G7: -80 mA to $+100\text{ mA}$ (sampled but not 100% tested).

AC Electrical Characteristics (Continued)

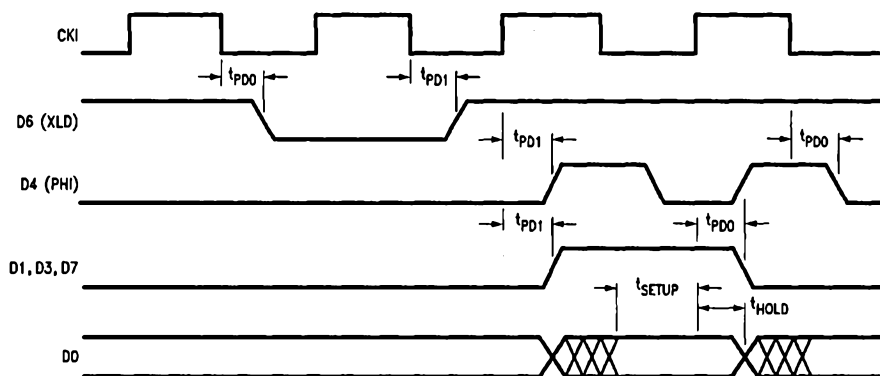


FIGURE 2a. AC Timing Diagrams in ROMless Mode

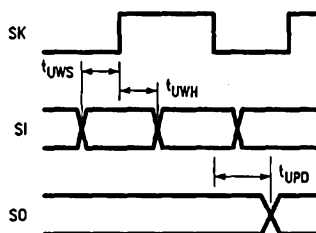


FIGURE 2b. MICROWIRE/PLUS Timing

Pin Descriptions

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

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.

RESET is the master reset input. See Reset Description section.

The COP888CL contains three bidirectional 8-bit I/O ports (C, G and L), where each individual bit may be independently configured as an input, 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 COP888CL memory map for the various

addresses associated with the I/O ports.) Figure 3 shows the I/O port configurations for the COP888CL. 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

Pin Descriptions (Continued)

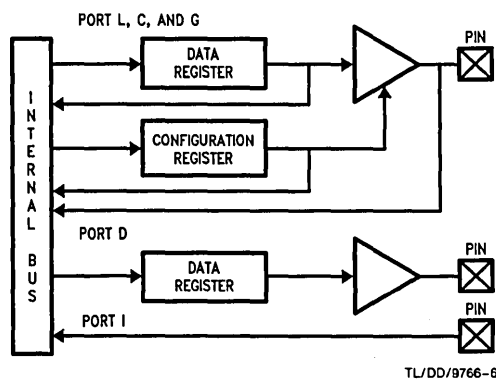


FIGURE 3. I/O Port Configurations

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 and L5 are used for the timer input functions T2A and T2B.

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
- L7 MIWU

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. 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 I is an 8-bit Hi-Z input port. The 40-pin device does not have a full complement of Port I pins. Pins 15 and 16 on this package must be connected to GND.

The 28-pin device has four I pins (I0, I1, I4, I5). The user should pay attention when reading port I to the fact that I4 and I5 are in bit positions 4 and 5 rather than 2 and 3.

The unavailable pins (I4–I7) are not terminated i.e., they are floating. A read operation for these unterminated pins will return unpredictable values. The user must ensure that the software takes into account by either masking or restricting the accesses to bit operations. The 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 together in order to get a higher drive.

Functional Description

The architecture of the COP888CL is modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The COP888CL architecture, though based on Harvard architecture, permits transfer of data from ROM to RAM.

CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction (t_c) 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

Program memory for the COP888CL consists of 4096 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 COP888CL 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 and SP pointers.

The COP888CL has 128 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, and B are memory mapped into this space at address locations 0FC to 0FE Hex respectively, with the other registers (other than reserved register 0FF) being available for general usage.

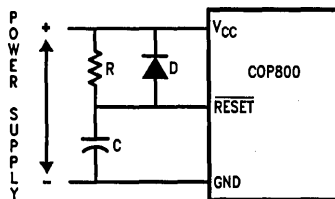
The instruction set of the COP888CL permits any bit in memory to be set, reset or tested. All I/O and registers on the COP888CL (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

Reset

The RESET input when pulled low initializes the microcontroller. Initialization will occur whenever the 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 RESET. The PC, PSW, CNTRL, ICNTRL, and T2CNTRL 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 COP888CL 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 t_c 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 t_c 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 4 should be used to ensure that the RESET pin is held low until the power supply to the chip stabilizes.



$$RC > 5 \times \text{Power Supply Rise Time}$$

TL/DD/8766-7

FIGURE 4. Recommended Reset Circuit

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 ($1/t_c$).

Figure 5 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.

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

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 pin.

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

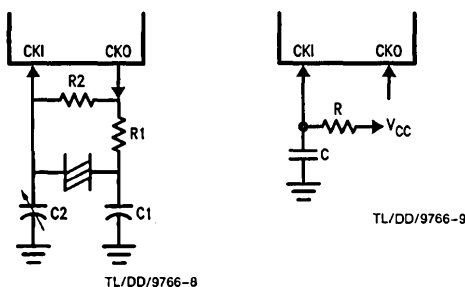


FIGURE 5. Crystal and R/C Oscillator Diagrams
TABLE A. Crystal Oscillator Configuration, $T_A = 25^\circ\text{C}$

R1 (k Ω)	R2 (M Ω)	C1 (pF)	C2 (pF)	CKI Freq (MHz)	Conditions
0	1	30	30–36	10	$V_{CC} = 5V$
0	1	30	30–36	4	$V_{CC} = 2.5V$
0	1	200	100–150	0.455	$V_{CC} = 2.5V$

TABLE B. RC Oscillator Configuration, $T_A = 25^\circ\text{C}$

R (k Ω)	C (pF)	CKI Freq (MHz)	Instr. Cycle (μs)	Conditions
3.3	82	2.8 to 2.2	3.6 to 4.5	$V_{CC} = 5V$
5.6	100	1.5 to 1.1	6.7 to 9	$V_{CC} = 5V$
6.8	100	1.1 to 0.8	9 to 12.5	$V_{CC} = 2.5V$

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. Clock Monitor current when enabled—I6

Thus the total current drain, I_t , is given as

$$I_t = I_1 + I_2 + I_3 + I_4 + I_5 + I_6$$

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 below, 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.

$$I_2 = C \times V \times f$$

where C = equivalent capacitance of the chip

V = operating voltage

f = CKI frequency

Some sample current drain values at $V_{CC} = 5V$ are:

CKI (MHz)	Inst. Cycle (μs)	I_t (mA)
10	1	15
3.58	2.8	5.4
2	5	3
0.3	33	0.45
0 (HALT)		0.005

Control Registers

CNTRL Register (Address X'00EE)

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 modes 1 and 2
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'00EF)

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

Control Registers (Continued)

T1PND 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	T1PND	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'00E8)

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

μWEN 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'00C6)

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

T2PND 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	T2PND	T2ENA	T2PNDB	T2ENB
------	------	------	------	-------	-------	--------	-------

Bit 7

Bit 0

ROMless Mode

The COP888CL can address up to 32 kbytes of address space. If at power up, D2 is held at ground, the COP888CL executes from external memory. Port D is used to interface to external program memory. The address comes out in a serial fashion and the data from the external program memory is read back in a serial fashion. The Port D pins perform the following functions.

D0 Shifts in ROM data

D1 Shifts out lower eight bits of PC

D2 Places the μC in the ROMless mode if grounded at reset

D3 Shifts out upper eight bits of PC

D4 Data Shift Clock

D5 HALT Mask Option select pin

(D5 = 0) for HALT enable, D5 = 1 for HALT disable)

D6 Load Clock

D7 Shifts out recreated Port D data

The most significant bit of the data to come out on the D3 pin is a status signal. It is used by the MOLE development system. This "lost" output port (D0–D7) can be accurately reconstructed with external components as shown in *Figure 6*, providing an accurate emulation.

The 44-pin and 40-pin versions of the COP888CL have a full complement of the D Port pins and can be used in the ROMless mode.

The 28-pin part cannot be used for emulation since it does not have the full complement of 8 D Port pins necessary for entering the ROMless mode.

Note that in the ROMless mode the D Port is recreated one full CK1 clock cycle behind the normal port timings.

Note: Standard parts used in the ROMless mode will operate only at a reduced frequency (to be defined).

The COP888CL device has a spare D pin (D5) in the ROMless mode since only seven pins are required for emulation and recreation. This pin D5 is used in the ROMless mode to enable or disable the HALT mask option feature.

Figure 6 shows the COP888CL ROMless Mode Schematic.

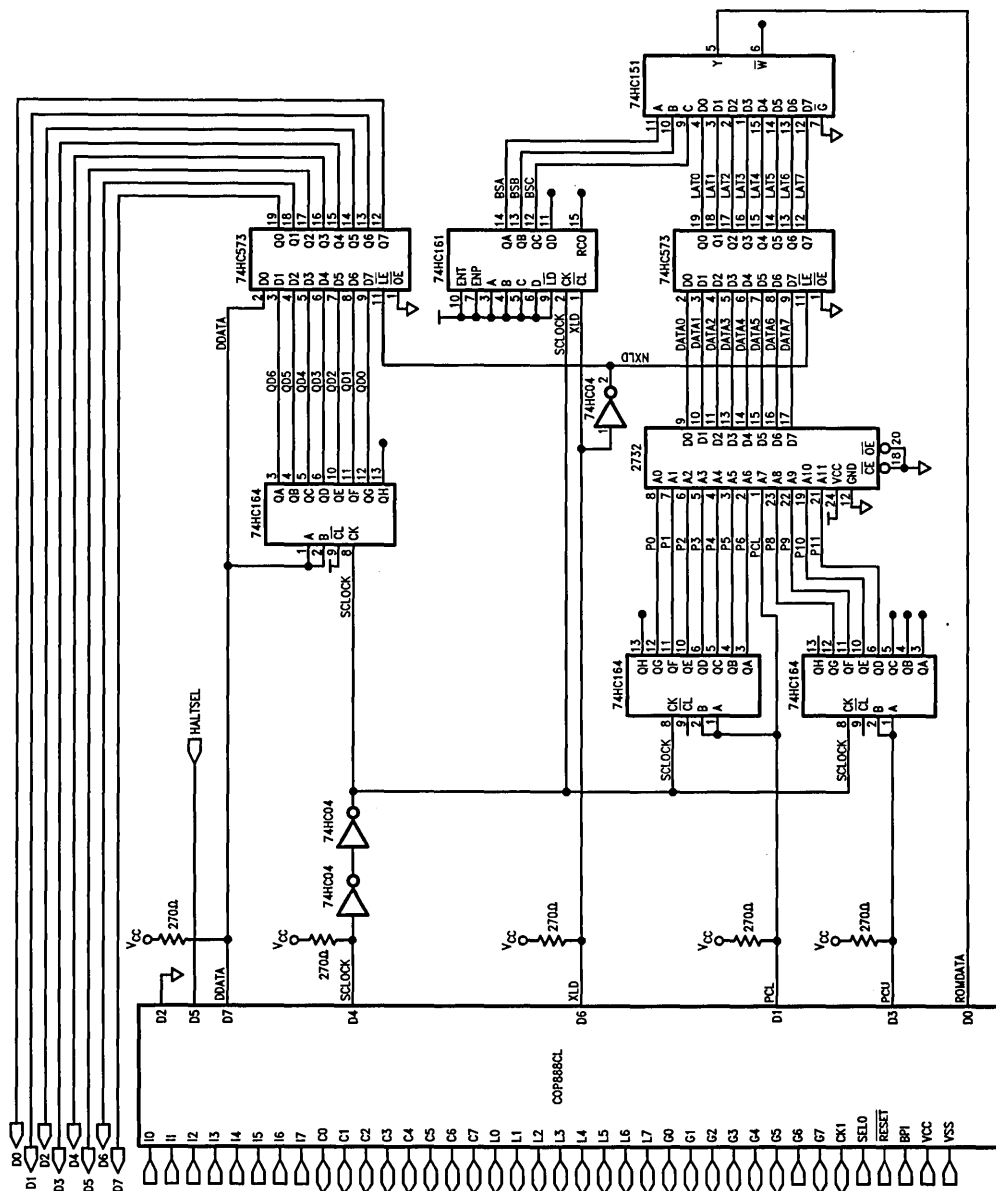


FIGURE 6. COP888CL ROMless Mode Schematic

Timers

The COP888CL contains a very versatile set of timers (T0, T1, T2). All timers and associated autoreload/capture registers power up containing random data.

Figure 7 shows a block diagram for the timers on the COP888CL.

TIMER T0 (IDLE TIMER)

The COP888CL 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, t_c . 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

The IDLE Timer T0 can generate an interrupt when the thirteenth bit toggles. This toggle is latched into the TOPND pending flag, and will occur every 4 ms at the maximum clock frequency ($t_c = 1 \mu s$). A control flag T0EN allows the interrupt from the thirteenth bit of Timer T0 to be enabled or disabled. Setting T0EN will enable the interrupt, while resetting it will disable the interrupt.

TIMER T1 AND TIMER T2

The COP888CL has a set of two powerful timer/counter blocks, T1 and T2. The associated features and functioning of a timer block are described by referring to the timer block Tx. Since the two timer blocks, T1 and T2, are identical, all comments are equally applicable to either timer block.

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 COP888CL to easily perform all timer functions with minimal software

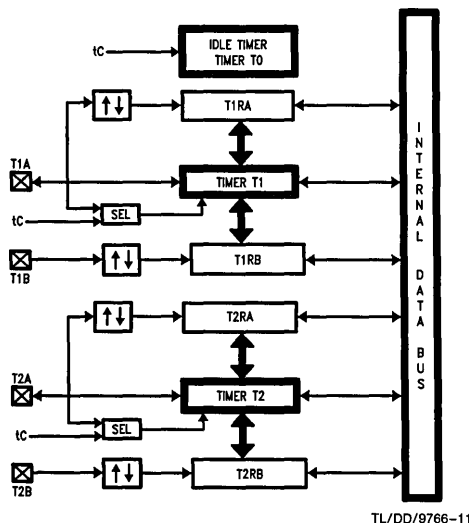


FIGURE 7. Timers for the COP888CL

overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

The control bits Tx3, Tx2, and Tx1 allow selection of the different modes of operation.

Mode 1. Processor Independent PWM Mode

As the name suggests, this mode allows the COP888CL 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 t_c . 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, Tx3, Tx2 and Tx1 set up the timer for PWM mode operation.

Figure 8 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, TxPND A and TxPND B. The user must reset these pending flags under software control. Two control enable flags, TxENA and TxENB, allow the interrupts from the 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.

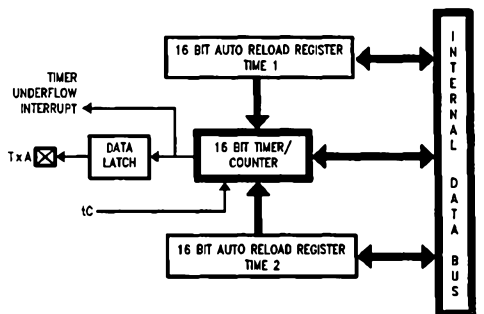


FIGURE 8. Timer in PWM Mode

TL/DD/9766-13

Timers (Continued)

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 TxPND 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 9 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.

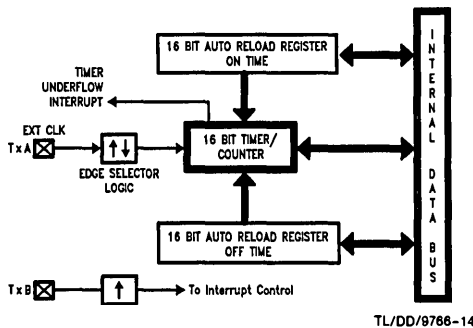


FIGURE 9. Timer In External Event Counter Mode

Mode 3. Input Capture Mode

The COP888CL 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.

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, TxPND 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 TxCO pending flag (the TxCO control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the TxCO 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 the TxPND and TxCO pending flags in order to determine whether a TxA input capture or a timer underflow (or both) caused the interrupt.

Figure 10 shows a block diagram of the timer in Input Capture mode.

TIMER CONTROL FLAGS

The timers T1 and T2 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
TxC1	Timer mode control
TxC2	Timer mode control
TxC3	Timer mode control
TxPND	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

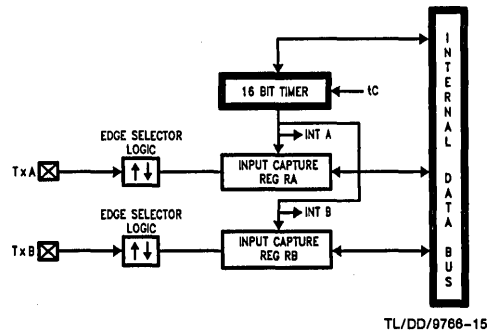


FIGURE 10. Timer In Input Capture Mode

Timers (Continued)

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 COP888CL 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 COP888CL is placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock, timers, are stopped. The WatchDog logic on the COP888CL is disabled during the HALT mode. However, the clock monitor circuitry, if enabled, remains active. In the HALT mode, the power requirements of the COP888CL are minimal and the applied voltage (V_{CC}) may be decreased to V_r ($V_r = 2.0V$) without altering the state of the machine.

The COP888CL 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 t_c instruction cycle clock. The t_c 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 startup timeout 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.

Power Save Modes (Continued)

The COP888CL 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 COP888CL will enter and exit the HALT mode as described above. With the HALT disable mask option, the COP888CL 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 COP888CL inadvertently enters the HALT mode as a result of a runaway program or power glitch.

IDLE MODE

The COP888CL 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 COP888CL can be returned to normal operation with a reset, or with a Multi-Input Wake-up from the L Port. Alternately, the microcontroller resumes normal operation from the IDLE mode when the thirteenth bit (representing 4.096 ms at internal clock frequency of 1 MHz, $t_c = 1 \mu s$) of the IDLE Timer toggles.

This toggle condition of the thirteenth bit of the IDLE Timer T0 is latched into the T0PND pending flag.

The user has the option of being interrupted with a transition on the twelfth bit of the IDLE Timer T0. 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 COP888CL 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 COP888CL will resume normal operation with the instruction immediately following the "Enter IDLE Mode" instruction.

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.

Multi-Input Wakeup

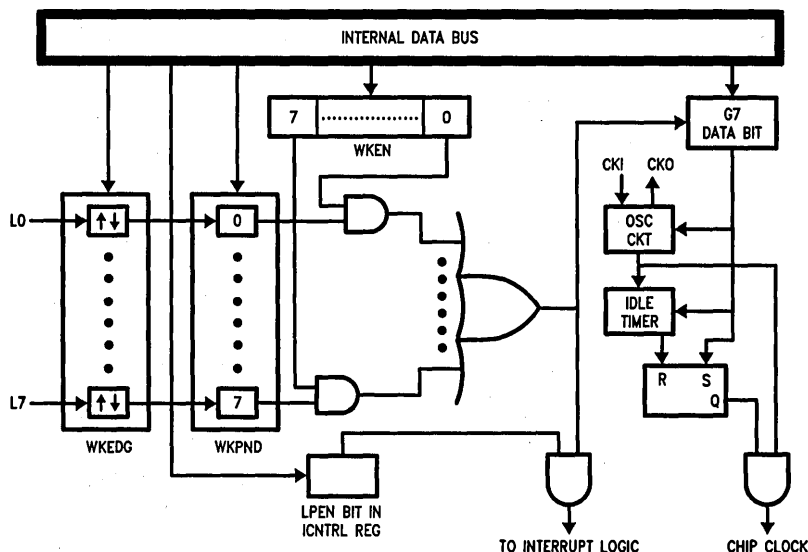


FIGURE 11. Multi-Input Wake Up Logic

TL/DD/9768-16

The Multi-Input Wakeup feature is used to return (wakeup) the COP888CL 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 11 shows the Multi-Input Wakeup logic for the COP888CL microcontroller.

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 COP888CL 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 COP888CL 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.

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

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

Multi-Input Wakeup (Continued)

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 COP888CL 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 COP888CL 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 COP888CL will first execute the interrupt service routine and then revert to normal operation.

The Wakeup signal will not start the chip running immediately since crystal oscillators or ceramic resonators have a finite start up time. The IDLE Timer (T0) generates a fixed delay to ensure that the oscillator has indeed stabilized before allowing the COP888CL to execute instructions. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry and the IDLE Timer T0 are enabled. The IDLE Timer is loaded with a value of 256 and is clocked from the t_c instruction cycle clock. The t_c clock is derived by dividing down the oscillator clock by a factor of 10. A Schmitt trigger

following the CKI on-chip inverter 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 startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If the RC clock option is used, the fixed delay is under software control. A control flag, CLKDLY, in the G7 configuration bit allows the clock start up delay to be optionally inserted. Setting CLKDLY flag high will cause clock start up delay to be inserted and resetting it will exclude the clock start up delay. The CLKDLY flag is cleared during reset, so the clock start up delay is not present following reset with the RC clock options.

Interrupts

The COP888CL supports a vectored interrupt scheme. It supports a total of ten interrupt sources. The following table lists all the possible COP888CL interrupt sources, their arbitration ranking and the memory locations reserved for the interrupt vector for each source.

Two bytes of program memory space are reserved for each interrupt source. All interrupt sources except the software interrupt are maskable. Each of the maskable interrupts have an Enable bit and a Pending bit. A maskable interrupt is active if its associated enable and pending bits are set. If GIE = 1 and an interrupt is active, then the processor will be interrupted as soon as it is ready to start executing an

Arbltration Ranking	Source	Description	Vector Address Hi-Low Byte
(1) Highest	Software	INTR Instruction	0yFE–0yFF
	Reserved	for Future Use	0yFC–0yFD
(2)	External	Pin G0 Edge	0yFA–0yFB
(3)	Timer T0	Underflow	0yF8–0yF9
(4)	Timer T1	T1A/Underflow	0yF6–0yF7
(5)	Timer T1	T1B	0yF4–0yF5
(6)	MICROWIRE/PLUS	BUSY Goes Low	0yF2–0yF3
	Reserved	for Future Use	0yF0–0yF1
	Reserved	for UART	0yEE–0yEF
	Reserved	for UART	0yEC–0yED
(7)	Timer T2	T2A/Underflow	0yEA–0yEB
(8)	Timer T2	T2B	0yE8–0yE9
	Reserved	for Future Use	0yE6–0yE7
	Reserved	for Future Use	0yE4–0yE5
(9)	Port L/Wakeup	Port L Edge	0yE2–0yE3
(10) Lowest	Default	VIS Instr. Execution without Any Interrupts	0yE0–0yE1

y is VIS page, y \neq 0.

Interrupts (Continued)

instruction except if the above conditions happen during the Software Trap service routine. This exception is described in the Software Trap sub-section.

The interruption process is accomplished with the INTR instruction (opcode 00), which is jammed inside the Instruction Register and replaces the opcode about to be executed. The following steps are performed for every interrupt:

1. The GIE (Global Interrupt Enable) bit is reset.
2. The address of the instruction about to be executed is pushed into the stack.
3. The PC (Program Counter) branches to address 00FF. This procedure takes 7 t_c cycles to execute.

At this time, since $GIE = 0$, other maskable interrupts are disabled. The user is now free to do whatever context switching is required by saving the context of the machine in the stack with PUSH instructions. The user would then program a VIS (Vector Interrupt Select) instruction in order to branch to the interrupt service routine of the highest priority interrupt enabled and pending at the time of the VIS. Note that this is not necessarily the interrupt that caused the branch to address location 00FF Hex prior to the context switching.

Thus, if an interrupt with a higher rank than the one which caused the interruption becomes active before the decision of which interrupt to service is made by the VIS, then the interrupt with the higher rank will override any lower ones and will be acknowledged. The lower priority interrupt(s) are still pending, however, and will cause another interrupt immediately following the completion of the interrupt service routine associated with the higher priority interrupt just serviced. This lower priority interrupt will occur immediately following the RETI (Return from Interrupt) instruction at the end of the interrupt service routine just completed.

Inside the interrupt service routine, the associated pending bit has to be cleared by software. The RETI (Return from

Interrupt) instruction at the end of the interrupt service routine will set the GIE (Global Interrupt Enable) bit, allowing the processor to be interrupted again if another interrupt is active and pending.

The VIS instruction looks at all the active interrupts at the time it is executed and performs an indirect jump to the beginning of the service routine of the one with the highest rank.

The addresses of the different interrupt service routines, called vectors, are chosen by the user and stored in ROM in a table starting at 01E0 (assuming that VIS is located between 00FF and 01DF). The vectors are 15-bit wide and therefore occupy 2 ROM locations.

VIS and the vector table must be located in the same 256-byte block (0y00 to 0yFF) except if VIS is located at the last address of a block. In this case, the table must be in the next block. The vector table cannot be inserted in the first 256-byte block.

The vector of the maskable interrupt with the lowest rank is located at 0yE0 (Hi-Order byte) and 0yE1 (Lo-Order byte) and so forth in increasing rank number. The vector of the maskable interrupt with the highest rank is located at 0yFA (Hi-Order byte) and 0yFB (Lo-Order byte).

The Software Trap has the highest rank and its vector is located at 0yFE and 0yFF.

If, by accident, a VIS gets executed and no interrupt is active, then the PC (Program Counter) will branch to a vector located at 0yE0–0yE1. This vector can point to the Software Trap (ST) interrupt service routine, or to another special service routine as desired.

Figure 12 shows the COP888CL Interrupt block diagram.

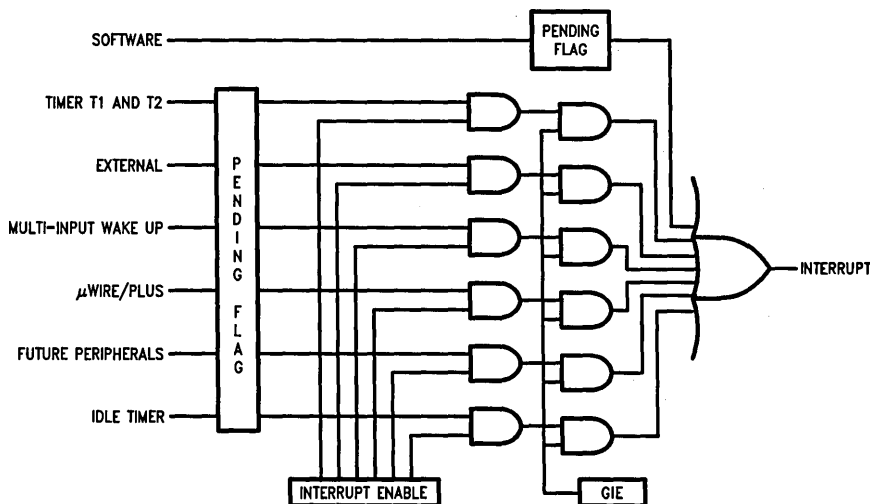


FIGURE 12. COP888CL Interrupt Block Diagram

TL/DD/9766-18

Interrupts (Continued)

SOFTWARE TRAP

The Software Trap (ST) is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from ROM and placed inside the instruction register. This may happen when the PC is pointing beyond the available ROM address space or when the stack is over-popped.

When an ST occurs, the user can re-initialize the stack pointer and do a recovery procedure (similar to reset, but not necessarily containing all of the same initialization procedures) before restarting.

The occurrence of an ST is latched into the ST pending bit. The GIE bit is not affected and the ST 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. The RPND instruction is used to clear the software interrupt pending bit. This pending bit is also cleared on reset.

The ST has the highest rank among all interrupts.

Nothing (except another ST) can interrupt an ST being serviced.

WatchDog

The COP888CL contains a WatchDog and clock monitor. The WatchDog is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or "runaway" programs. 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 I shows the WDSVR register.

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 II 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 I. WatchDog Service Register (WDSVR)

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

TABLE II. WatchDog Service Window Select

WDSVR Bit 7	WDSVR Bit 6	Service Window (Lower-Upper Limits)
0	0	2k-8k t_c Cycles
0	1	2k-16k t_c Cycles
1	0	2k-32k t_c Cycles
1	1	2k-64k t_c Cycles

Clock Monitor

The Clock Monitor aboard the COP888CL 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 Operation

The WatchDog and Clock Monitor are disabled during reset. The COP888CL 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 III 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 user may service the WatchDog as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WatchDog service.

The WatchDog has an output pin associated with it. This is the WDOOUT pin, on pin 1 of the port G. WDOOUT is active low. The WDOOUT pin is in the high impedance state in the inactive state. Upon triggering the WatchDog, the logic will pull the WDOOUT (G1) pin low for an additional $16 t_c$ – $32 t_c$ cycles after the signal level on WDOOUT pin goes below the lower Schmitt trigger threshold. After this delay, the COP888CL will stop forcing the WDOOUT output low.

The WatchDog service window will restart when the WDOOUT pin goes high. It is recommended that the user tie the WDOOUT pin back to V_{CC} through a resistor in order to pull WDOOUT high.

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

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 enter the high impedance TRI-STATE mode following $16 t_c$ – $32 t_c$ clock cycles. The Clock Monitor generates a continual Clock Moni-

WatchDog Operation (Continued)

TABLE III. 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

TABLE IV. MICROWIRE/PLUS
Master Mode Clock Select

SL1	SL0	SK
0	0	$2 \times t_c$
0	1	$4 \times t_c$
1	x	$8 \times t_c$

Where t_c is the instruction cycle clock

tor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_c > 10 \text{ kHz}$ —No clock rejection.

$1/t_c < 10 \text{ Hz}$ —Guaranteed clock rejection.

Detection of Illegal Conditions

The COP888CL 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 zero. 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 Hex 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:

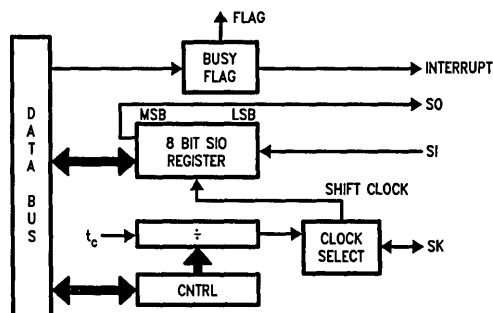
- Executing from undefined ROM
- Over "POP"ing the stack by having more returns than calls.

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.

MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the COP888CL 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 13 shows a block diagram of the MICROWIRE logic.

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 arrangement with an external shift clock is called the Slave mode of operation.



TL/DD/8766-20

FIGURE 13. MICROWIRE/PLUS Block Diagram

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 IV details the different clock rates that may be selected.

MICROWIRE/PLUS (Continued)

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 COP888CL may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. *Figure 14* shows how two COP888CL microcontrollers 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. The 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 by the COP888CI. 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 V 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 MSE 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 COP888CL 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 alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out 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 V

This table assumes that the control flag MSEL is set.

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

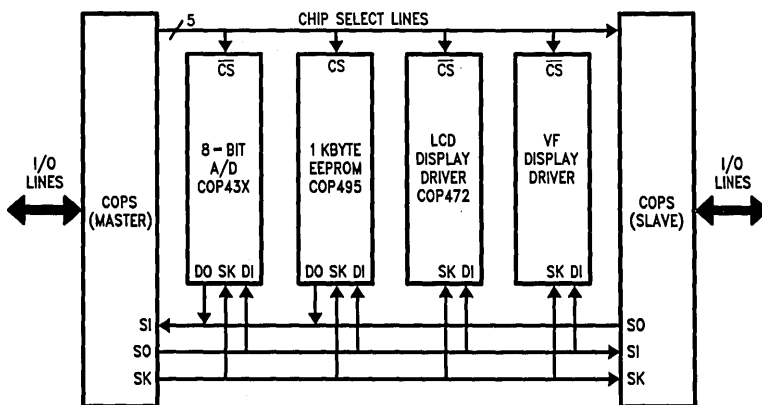


FIGURE 14. MICROWIRE/PLUS Application

TL/DD/9768-21

Memory Map

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

Address	Contents
00 to 6F	On-Chip RAM bytes
70 to BF	Unused RAM Address Space
C0	Timer T2 Lower Byte
C1	Timer T2 Upper Byte
C2	Timer T2 Autoload Register T2RA Lower Byte
C3	Timer T2 Autoload Register T2RA Upper Byte
C4	Timer T2 Autoload Register T2RB Lower Byte
C5	Timer T2 Autoload Register T2RB Upper Byte
C6	Timer T2 Control Register
C7	WatchDog Service Register (Reg:WDSVR)
C8	MIWU Edge Select Register (Reg:WKEDG)
C9	MIWU Enable Register (Reg:WKEN)
CA	MIWU Pending Register (Reg:WKPND)
CB	Reserved
CC	Reserved
CD to CF	Reserved
D0	Port L Data Register
D1	Port L Configuration Register
D2	Port L Input Pins (Read Only)
D3	Reserved for Port L
D4	Port G Data Register
D5	Port G Configuration Register
D6	Port G Input Pins (Read Only)
D7	Port I Input Pins (Read Only)
D8	Port C Data Register
D9	Port C Configuration Register
DA	Port C Input Pins (Read Only)
DB	Reserved for Port C
DC	Port D Data Register
DD to DF	Reserved for Port D
E0 to E5	Reserved
E6	Timer T1 Autoload Register T1RB Lower Byte
E7	Timer T1 Autoload Register T1RB Upper Byte
E8	ICNTRL Register
E9	MICROWIRE Shift Register
EA	Timer T1 Lower Byte
EB	Timer T1 Upper Byte
EC	Timer T1 Autoload Register T1RA Lower Byte
ED	Timer T1 Autoload Register T1RA Upper Byte
EE	CNTRL Control Register
EF	PSW Register
F0 to FB	On-Chip RAM Mapped as Registers
FC	X Register
FD	SP Register
FE	B Register
FF	Reserved

Reading memory locations 70-7F Hex will return all ones. Reading other unused memory locations will return undefined data.

Addressing Modes

The COP888CL has ten addressing modes, six for operand addressing and four for transfer of control.

OPERAND ADDRESSING MODES

Register Indirect

This is the "normal" addressing mode for the COP888CL. 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 LBI 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
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 (Continued)

INSTRUCTION SET

ADD	A,Meml	ADD	$A \leftarrow A + \text{Meml}$
ADC	A,Meml	ADD with Carry	$A \leftarrow A + \text{Meml} + C, C \leftarrow \text{Carry}$ $HC \leftarrow \text{Half Carry}$
SUBC	A,Meml	Subtract with Carry	$A \leftarrow A - \text{Meml} + C, C \leftarrow \text{Carry}$ $HC \leftarrow \text{Half Carry}$
AND	A,Meml	Logical AND	$A \leftarrow A \text{ and Meml}$
ANDSZ	A,Imm	Logical AND Immed., Skip if Zero	Skip next if $(A \text{ and Imm}) = 0$
OR	A,Meml	Logical OR	$A \leftarrow A \text{ or Meml}$
XOR	A,Meml	Logical EXclusive OR	$A \leftarrow A \text{ xor Meml}$
IFEQ	MD,Imm	IF EQUAL	Compare MD and Imm, Do next if $MD = \text{Imm}$
IFEQ	A,Meml	IF EQUAL	Compare A and Meml, Do next if $A = \text{Meml}$
IFNE	A,Meml	IF Not Equal	Compare A and Meml, Do next if $A \neq \text{Meml}$
IFGT	A,Meml	IF Greater Than	Compare A and Meml, Do next if $A > \text{Meml}$
IFBNE	#	If B Not Equal	Do next if lower 4 bits of $B \neq \text{Imm}$
DRSZ	Reg	Decrement Reg., Skip if Zero	$\text{Reg} \leftarrow \text{Reg} - 1$, Skip if $\text{Reg} = 0$
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	$A \leftrightarrow \text{Mem}$
X	A,[X]	EXchange A with Memory [X]	$A \leftrightarrow [X]$
LD	A,Meml	LoAD A with Memory	$A \leftarrow \text{Meml}$
LD	A,[X]	LoAD A with Memory [X]	$A \leftarrow [X]$
LD	B,Imm	LoAD B with Immed.	$B \leftarrow \text{Imm}$
LD	Mem,Imm	LoAD Memory Immed	$\text{Mem} \leftarrow \text{Imm}$
LD	Reg,Imm	LoAD Register Memory Immed.	$\text{Reg} \leftarrow \text{Imm}$
X	A, [B \pm]	EXchange A with Memory [B]	$A \leftrightarrow [B], (B \leftarrow B \pm 1)$
X	A, [X \pm]	EXchange A with Memory [X]	$A \leftrightarrow [X], (X \leftarrow X \pm 1)$
LD	A, [B \pm]	LoAD A with Memory [B]	$A \leftarrow [B], (B \leftarrow B \pm 1)$
LD	A, [X \pm]	LoAD A with Memory [X]	$A \leftarrow [X], (X \leftarrow X \pm 1)$
LD	[B \pm],Imm	LoAD Memory [B] Immed.	$[B] \leftarrow \text{Imm}, (B \leftarrow B \pm 1)$
CLR	A	CLear A	$A \leftarrow 0$
INC	A	INCRement A	$A \leftarrow A + 1$
DEC	A	DECrement A	$A \leftarrow A - 1$
LAID		LoAD A INDirect from ROM	$A \leftarrow \text{ROM (PU,A)}$
DCOR	A	Decimal CORrect A	$A \leftarrow \text{BCD correction of A (follows ADC, SUBC)}$
RRC	A	Rotate A Right thru C	$C \leftrightarrow A7 \leftrightarrow \dots \leftrightarrow A0 \leftrightarrow C$
RLC	A	Rotate A Left thru C	$C \leftarrow A7 \leftarrow \dots \leftarrow A0 \leftarrow C$
SWAP	A	SWAP nibbles of A	$A7 \dots A4 \leftrightarrow A3 \dots A0$
SC		Set C	$C \leftarrow 1, HC \leftarrow 1$
RC		Reset C	$C \leftarrow 0, HC \leftarrow 0$
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	$SP \leftarrow SP + 1, A \leftarrow [SP]$
PUSH	A	PUSH A onto the stack	$[SP] \leftarrow A, SP \leftarrow SP - 1$
VIS		Vector to Interrupt Service Routine	$PU \leftarrow [VU], PL \leftarrow [VL]$
JMPL	Addr.	Jump absolute Long	$PC \leftarrow ii \text{ (} ii = 15 \text{ bits, } 0 \text{ to } 32k)$
JMP	Addr.	Jump absolute	$PC9 \dots 0 \leftarrow i \text{ (} i = 12 \text{ bits)}$
JP	Disp.	Jump relative short	$PC \leftarrow PC + r \text{ (} r \text{ is } -31 \text{ to } +32, \text{ except } 1)$
JSRL	Addr.	Jump SubRoutine Long	$[SP] \leftarrow PL, [SP-1] \leftarrow PU, SP-2, PC \leftarrow ii$
JSR	Addr	Jump SubRoutine	$[SP] \leftarrow PL, [SP-1] \leftarrow PU, SP-2, PC9 \dots 0 \leftarrow i$
JID		Jump INDirect	$PL \leftarrow \text{ROM (PU,A)}$
RET		RETurn from subroutine	$SP+2, PL \leftarrow [SP], PU \leftarrow [SP-1]$
RETSK		RETurn and Skip	$SP+2, PL \leftarrow [SP], PU \leftarrow [SP-1]$
RETI		RETurn from Interrupt	$SP+2, PL \leftarrow [SP], PU \leftarrow [SP-1], GIE \leftarrow 1$
INTR		Generate an Interrupt	$[SP] \leftarrow PL, [SP-1] \leftarrow PU, SP-2, PC \leftarrow 0FF$
NOP		No Operation	$PC \leftarrow PC + 1$

Instruction Execution Time

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

Most single byte instructions take one cycle time (1 μ s at 10 MHz) 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 (a cycle is 1 μ s at 10 MHz).

	[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
IFNE	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
DCOR	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-]
XA,*	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		
LD B, Imm				2/2		
LD Mem, Imm	2/2		3/3		2/2	
LD Reg, Imm			2/3			
IFEQ MD, Imm			3/3			

(IF B < 16)

(IF B > 15)

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

COP888CL Opcode Table

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

F	E	D	C	B	A	9	8	
JP -15	JP -31	LD 0F0, # i	DRSZ 0F0	RRCA	RC	ADC A, #i	ADC A,[B]	0
JP -14	JP -30	LD 0F1, # i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A,[B]	1
JP -13	JP -29	LD 0F2, # i	DRSZ 0F2	X A, [X+]	X A,[B+]	IFEQ A, #i	IFEQ A,[B]	2
JP -12	JP -28	LD 0F3, # i	DRSZ 0F3	X A, [X-]	X A,[B-]	IFGT A, #i	IFGT A,[B]	3
JP -11	JP -27	LD 0F4, # i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A,[B]	4
JP -10	JP -26	LD 0F5, # i	DRSZ 0F5	RPND	JID	AND A, #i	AND A,[B]	5
JP -9	JP -25	LD 0F6, # i	DRSZ 0F6	X A,[X]	X A,[B]	XOR A, #i	XOR A,[B]	6
JP -8	JP -24	LD 0F7, # i	DRSZ 0F7	*	*	OR A, #i	OR A,[B]	7
JP -7	JP -23	LD 0F8, # i	DRSZ 0F8	NOP	RLCA	LDA, #i	IFC	8
JP -6	JP -22	LD 0F9, # i	DRSZ 0F9	IFNE A,[B]	IFEQ Md, #i	IFNE A, #i	IFNC	9
JP -5	JP -21	LD 0FA, # i	DRSZ 0FA	LD A,[X+]	LD A,[B+]	LD [B+], #i	INCA	A
JP -4	JP -20	LD 0FB, # i	DRSZ 0FB	LD A,[X-]	LD A,[B-]	LD [B-], #i	DECA	B
JP -3	JP -19	LD 0FC, # i	DRSZ 0FC	LD Md, #i	JMPL	X A,Md	POPA	C
JP -2	JP -18	LD 0FD, # i	DRSZ 0FD	DIR	JSRL	LD A,Md	RETSK	D
JP -1	JP -17	LD 0FE, # i	DRSZ 0FE	LD A,[X]	LD A,[B]	LD [B], #i	RET	E
JP -0	JP -16	LD 0FF, # i	DRSZ 0FF	*	*	LDB, #i	RETI	F

COP888CL Opcode Table (Continued)

Upper Nibble Along X-Axis

Lower Nibble Along Y-Axis

7	6	5	4	3	2	1	0	
IFBIT 0,[B]	ANDSZ A, #i	LD B, #0F	IFBNE 0	JSR x000-x0FF	JMP x000-x0FF	JP + 17	INTR	0
IFBIT 1,[B]	*	LD B, #0E	IFBNE 1	JSR x100-x1FF	JMP x100-x1FF	JP + 18	JP + 2	1
IFBIT 2,[B]	*	LD B, #0D	IFBNE 2	JSR x200-x2FF	JMP x200-x2FF	JP + 19	JP + 3	2
IFBIT 3,[B]	*	LD B, #0C	IFBNE 3	JSR x300-x3FF	JMP x300-x3FF	JP + 20	JP + 4	3
IFBIT 4,[B]	CLRA	LD B, #0B	IFBNE 4	JSR x400-x4FF	JMP x400-x4FF	JP + 21	JP + 5	4
IFBIT 5,[B]	SWAPA	LD B, #0A	IFBNE 5	JSR x500-x5FF	JMP x500-x5FF	JP + 22	JP + 6	5
IFBIT 6,[B]	DCORA	LD B, #09	IFBNE 6	JSR x600-x6FF	JMP x600-x6FF	JP + 23	JP + 7	6
IFBIT 7,[B]	PUSHA	LD B, #08	IFBNE 7	JSR x700-x7FF	JMP x700-x7FF	JP + 24	JP + 8	7
SBIT 0,[B]	RBIT 0,[B]	LD B, #07	IFBNE 8	JSR x800-x8FF	JMP x800-x8FF	JP + 25	JP + 9	8
SBIT 1,[B]	RBIT 1,[B]	LD B, #06	IFBNE 9	JSR x900-x9FF	JMP x900-x9FF	JP + 26	JP + 10	9
SBIT 2,[B]	RBIT 2,[B]	LD B, #05	IFBNE 0A	JSR xA00-xAFF	JMP xA00-xAFF	JP + 27	JP + 11	A
SBIT 3,[B]	RBIT 3,[B]	LD B, #04	IFBNE 0B	JSR xB00-xBFF	JMP xB00-xBFF	JP + 28	JP + 12	B
SBIT 4,[B]	RBIT 4,[B]	LD B, #03	IFBNE 0C	JSR xC00-xCFF	JMP xC00-xCFF	JP + 29	JP + 13	C
SBIT 5,[B]	RBIT 5,[B]	LD B, #02	IFBNE 0D	JSR xD00-xDFF	JMP xD00-xDFF	JP + 30	JP + 14	D
SBIT 6,[B]	RBIT 6,[B]	LD B, #01	IFBNE 0E	JSR xE00-xEFF	JMP xE00-xEFF	JP + 31	JP + 15	E
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

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

Mask Options

The COP888CL 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 (CK0) 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: COP888CL BONDING

- = 1 44-Pin PCC
- = 2 40-Pin DIP
- = 3 28-Pin PCC
- = 4 28-Pin DIP

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 (if clock option-1 has been selected). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ($1/t_c$).

Development Support

MOLE™ DEVELOPMENT SYSTEM

The MOLE (Microcomputer On Line Emulator) is a low cost development system and emulator for all microcontroller products. These include COPs™ microcontrollers and the HPC family of products. The MOLE consists of a BRAIN Board, Personality Board and optional host software.

The purpose of the MOLE is to provide the user with a tool to write and assemble code, emulate code for the target microcontroller and assist in both software and hardware debugging of the system.

It is a self contained computer with its own firmware which provides for all system operation, emulation control, communication, PROM programming and diagnostic operations.

It contains three serial ports to optionally connect to a terminal, a host system, a printer or a modem, or to connect to other MOLEs in a multi-MOLE environment.

MOLE can be used in either a stand alone mode or in conjunction with a selected host system using PC-DOS communicating via a RS-232 port.

How to Order

To order a complete development package, select the section for the microcontroller to be developed and order the parts listed.

Development Tools Selection Table

Microcontroller	Order Part Number	Description	Includes	Manual Number
COP888	MOLE-BRAIN	Brain Board	Brain Board Users Manual	420408188-001
	MOLE-COP8-PB2	Personality Board	COP888 Personality Board Users Manual	420420084-001
	MOLE-COP8-IBM	Assembler Software for IBM	COP800 Software Users Manual and Software Disk PC-DOS Communications Software Users Manual	424410527-001 420040416-001
	420411060-001	Programmer's Manual		420411060-001

Development Support (Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications group. The Dial-A-Helper is an Electronic Bulletin Board Information System and additionally provides the capability of remotely accessing the MOLE development system at a customer site.

Information System

The Dial-A-Helper system provides access to an automated information storage and retrieval system that may be accessed over standard dial-up telephone lines 24 hours a day. The system capabilities include a MESSAGE SECTION (electronic mail) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities could be found. The minimum requirement for accessing the Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

Order P/N: MOLE-DIAL-A-HLP

Information System Package Contents

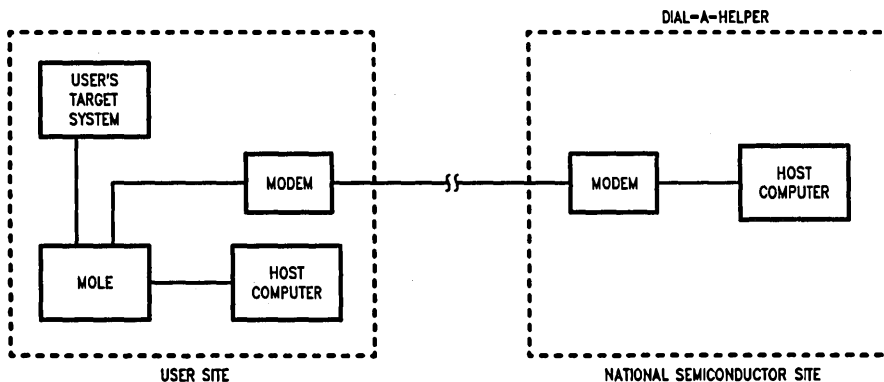
Dial-A-Helper User Manual P/N

Public Domain Communications Software

Factory Applications Support

Dial-A-Helper also provides immediate factory applications support. If a user is having difficulty in operating a MOLE, he can leave messages on our electronic bulletin board, which we will respond to, or under extraordinary circumstances he can arrange for us to actually take control of his system via modem for debugging purposes.

Voice: (408) 721-5582
 Modem: (408) 739-1162
 Baud: 300 or 1200 baud
 Set-up: Length: 8-Bit
 Parity: None
 Stop Bit 1
 Operation: 24 Hours, 7 Days



TL/DD/9766-24