

Computer Operating Properly - COP Reset

The *Computer Operating Properly* function is a *watchdog timer*. It can reset the microcontroller if the program gets lost.

A COP, or watchdog, system is a vital part of computers used in embedded applications. The system must have some way to recover from unexpected errors that may occur. Power surges or programming errors may cause the program "to get lost" and to lose control of the system. This could be disastrous and so the watchdog timer is included to help the program recover. When in operation, the program is responsible for pulsing the COP at specific intervals. This is accomplished by choosing a place in the program to pulse the watchdog timer regularly. Then, if the program fails to do this, the COP automatically provides a hardware RESET* to begin the processing again.¹

The COP failure interrupt vector is at \$FFFA:FFFB.

For the M68HCS12 operating in normal modes, the COP is disabled when the CPU is reset. The COP timeout period is controlled by the *CR2:CR1:CR0* bits in the *COPCTL* control register, and the COP is enabled by writing a non-zero value to these bits. *CR2:CR1:CR0* may be programmed to give a COP time-out period ranging from 1.024 ms to 1.049 seconds when the *OSCCLK* is 16 MHz. After the COP timer has started, the program must write first \$55 followed by \$AA to the *ARMCOP* - *Arm COP Register* before the COP times out. During each COP time-out period, this sequence (\$55 followed by \$AA) must be written. Other instructions can be executed between the \$55 and the \$AA but both must be completed in the time-out period to avoid a COP reset. When a COP timeout occurs, or if the program writes anything other than \$55 or \$AA to the *ARMCOP* register, a COP RESET is generated and the program restarts at the program location given by the COP failure vector. See Example 1.

A *windowed* COP operation is available. When the *WCOP* bit in the *COPTL* register is set, writes to the *ARMCOP* register to clear the COP timer must occur in the last 25% of the selected time-out period. A premature write will immediately reset the microcontroller.

¹ A particularly good article on watchdog timers can be found at <http://www.ganssle.com/watchdogs.htm>

COPCTL – Base + \$003C – COP Control Register

	Bit 7	6	5	4	3	2	1	0
Read:	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
Write:								
Reset:	0	0	0	0	0	0	0	0
= reserved, unimplemented or cannot be written to.								
WCOP								
Windowed COP Mode Bit.								
Read: Anytime. Write: Once in user mode, anytime in special mode.								
When set, a write to the ARMCOP register must occur in the last 25% of the selected period. A write during the first 75% will reset the microcontroller. As long as all write occur during this window, \$55 can be written as often as desired. Once \$AA is written after the \$55, the time-out logic restarts and the user must wait until the next window before writing to ARMCOP again.								
0 = Normal COP operation. (Default)								
1 = Windowed COP operation.								
RSBCK								
COP and RTI stop in Active BDM mode bit.								
Read: Anytime. Write: Once.								
0 = Allows the COP and real-time interrupt to keep running in active background debugger mode. (Default)								
1 = Stops the COP and RTI counter whenever the microcontroller is in active BDM mode.								
CR2, CR1, CR0								
COP Watchdog Timer Rate Select.								
Read: Anytime. Write: Once in user mode, anytime in special mode.								
See Table 1.								

Table 1 COP Watchdog Rates

CR2	CR1	CR0	OSSCLK cycles to time-out	Time to time-out with 16 MHz Clock
0	0	0	COP disabled	
0	0	1	2^{14}	1.024 ms.
0	1	0	2^{16}	4.096 ms
0	1	1	2^{18}	16.384 ms
1	0	0	2^{20}	65.536 ms
1	0	1	2^{22}	262.144 ms
1	1	0	2^{23}	524.288 ms
1	1	1	2^{24}	1.049 s

Note: The very first COP timeout is only half of the nominal COP timeout period. This is because the counter starts at 0000 and the timeout becomes active when the MSB goes high. The period is from a rising edge to the next rising edge.

ARMCOP – Base + \$003F – Arm Cop Register

	Bit 7	6	5	4	3	2	1	0
Read:	0	0	0	0	0	0	0	0
Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0
<p>Read: Always reads \$00 Write: Anytime When the COP is disabled (CR[2:0] = "000") writing to this register has no effect. When the COP is enabled by setting CR[2:0] to non-zero, the following applies: Writing any value other than \$55 or \$AA causes a COP reset. To restart the COP time-out period you must write \$55 followed by a write of \$AA. Other instructions may be executed between these writes but the sequence \$55 \$AA must be completed prior to the COP end of time-out period to avoid a COP reset. Sequences of \$55 or \$AA writes are allowed. When the WCOP bit is set, \$55 and \$AA must be done in the last 25% of the selected time -out period; writing any value in the first 75% of the selected period will cause a COP reset.</p>								

Example 1 COP Initialization

Show a segment of code that initializes the COP timeout period to 1.049 seconds and then, in the main processing loop, resets the COP timer.

Solution:

Metrowerks HC12-Assembler

(c) COPYRIGHT METROWERKS 1987-2003

```

Rel. Loc   Obj. code Source line
-----
1           ; Example program to initialize and start
2           ; the COP and then to reset it in the main
3           ; control loop.
4           XDEF COP_Restart
5           0000 0000 BASE: EQU 0 ; Registers base address
6           0000 003C COPCTL: EQU BASE+$003C ; COP Control reg
7           0000 003F ARMCOP: EQU BASE+$003F ; COP Arm register
8           0000 0007 SEC_1 EQU %00000111 ; 1.049 sec timeout
9
10          Entry:
11          MyCode: SECTION
12          ; All I/O initialization
13          ;
14          ; Set up the COP
15 000000 8607          ldaa #SEC_1
16 000002 5A3C          staa COPCTL
17          ;
18
19          loop:
20          ; Main processing loop
21          ; Blah, blah, blah
22          ; Now reset the COP before it times out
23 000004 180B 5500          movb #$55,ARMCOP
24          000008 3F
25 000009 180B AA00          movb #$AA,ARMCOP
26          00000D 3F
27          ; Continue with the main processing loop
28          bra loop
29

```

```
28           ; COP restart code
29           ; This code is executed when the COP reset
30           ; occurs.
31 COP_Restart:
32 000010 06xx xx           jmp     Entry
```
