Tutorial Introduction

PURPOSE

- To explain how to configure and use the Computer Operating Properly (COP) Module in common applications

OBJECTIVES:

- Describe the uses and features of the COP Module.
- Identify the steps to configure and service the COP Module.
- Describe methods for protecting the system from runaway software.

CONTENT:

- 9 pages
- 1 question

LEARNING TIME:

- 10 minutes

PREREQUESITE:

- Training modules: 68HC08 CPU, Resets and Interrupts

Welcome to this tutorial on the 68HC08 Computer Operating Properly (COP) Module. This tutorial describes the features and configuration of the COP. Please note that on subsequent pages, you will find reference buttons in the upper right of the content frame that access additional content.

Upon completion of this tutorial, you'll be able to configure and service the COP Module. You'll also be able to describe methods for protecting the system from runaway software.

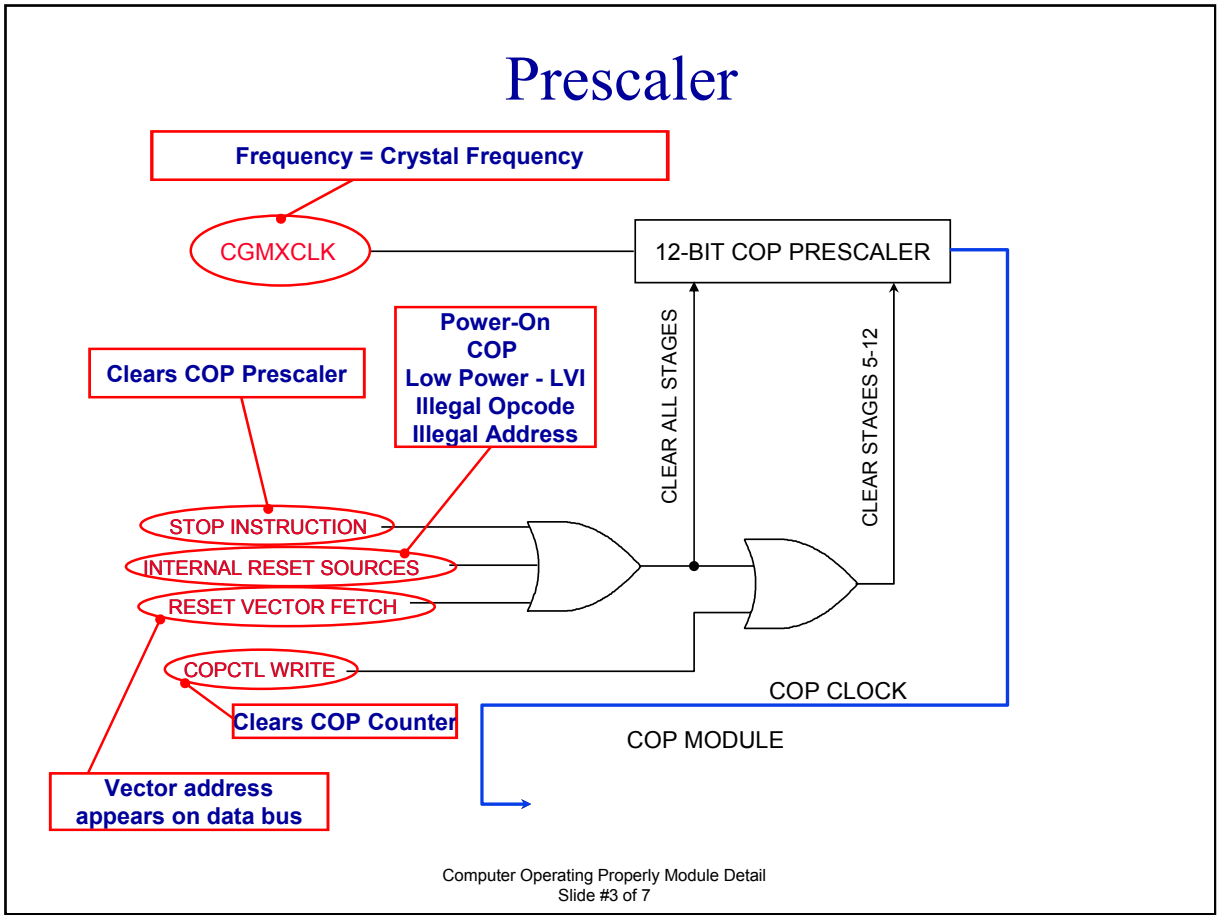
The recommended prerequisites for this tutorial are the 68HC08 CPU and the Resets and Interrupts training modules. Click the Forward arrow when you're ready to begin the tutorial.

Uses and Features

- Free-running counter cleared by user code.
- Provides system protection.
- Allows the CPU to recover to a known state after unexpected events:
  - Runaway software
  - Software processing errors
- Includes a selectable timeout period.
- COP resets if not serviced within the timeout period.

Let's begin this tutorial with an overview of the uses and features of the COP. The COP timer, also known as a watchdog timer, is based on a free-running counter that can be cleared by user code. The COP allows the CPU to recover from unexpected events such as runaway software and software processing errors.

To use the COP, simply enable it and select the desired timeout period. Most importantly, you must remember to always service it before the timeout period expires. If the COP timer is enabled, and it isn't serviced within the timeout period, the COP will reset the CPU.



Next, let's review the different signals and blocks associated with the COP Module beginning with the COP prescaler.

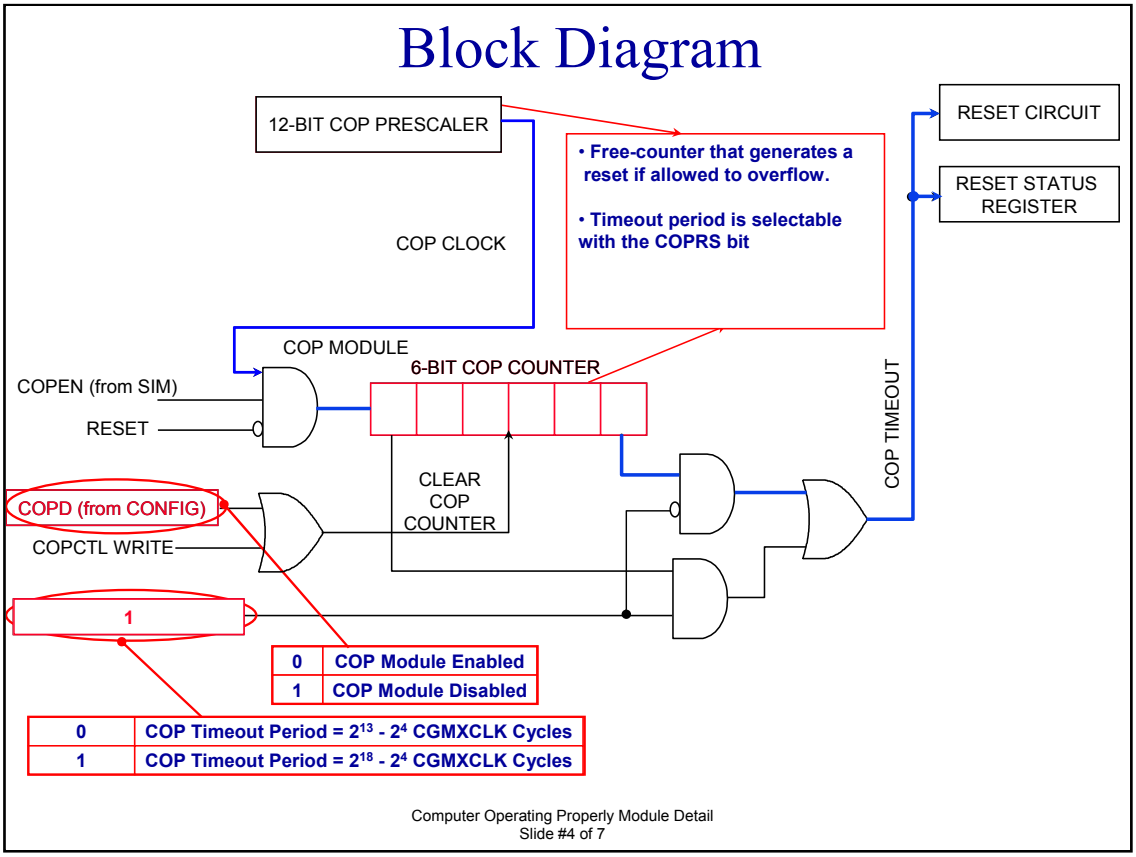
The CGMXCLK is the crystal oscillator output signal. The CGMXCLK frequency is equal to the crystal frequency.

The STOP instruction automatically clears the COP prescaler. You can prevent inadvertently turning off the COP with a STOP instruction by disabling the STOP instruction. When the STOP instruction is disabled, executing a STOP instruction results in an illegal opcode reset.

Internal resets, such as power-on, COP, low-power LVI, illegal opcode, and illegal address will clear the COP prescaler and the COP counter.

A reset vector fetch occurs when the vector address appears on the data bus. A reset vector fetch clears the COP prescaler.

Writing any value to the COP control register, COPCTL, clears the COP counter and clears bits 12 through 5 of the prescaler.



The figure shows the remaining signals and blocks associated with the COP Module

The COP is enabled out of reset. If COP operation is not desired, it should immediately be disabled by setting the COP disable bit , COPD, in the configuration register. The configuration register is a write-once register to protect against runaway code that may inadvertently disable the COP. Once the CONFIG register is written with the COP enabled, the COP cannot be disabled until the device is reset.

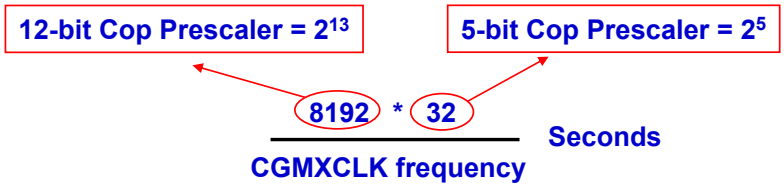
The COP rate select bit , COPRS, in the configuration register selects one of two timeout periods. Setting the COPRS bit to 1 selects a COP timeout period of 2^{13} - 2^4 CGMXCLK cycles. With the COPRS bit set to 1, a 32.768-kHz crystal gives a COP timeout period of 250 ms. Setting the COPRS bit to 0 selects a COP timeout period of 2^{18} - 2^4 CGMXCLK cycles.

The COP counter is a free-running 6-bit counter preceded by a 12-bit prescaler counter. If not cleared by software before the timeout period expires, the COP counter overflows and generates an asynchronous reset. To prevent a COP reset, write any value to the COP control register, COPCTL, before an overflow occurs. This clears the COP counter and stages 12 through 5 of the prescaler.

Reset Rate Example

Refresh rate depends on CGMXCLK frequency

- COP must be serviced within:



- Example: 4 MHz external crystal (CGMXCLK = 4 MHz)

$$\frac{8192 * 32}{4000000} \approx 0.065536 \text{ seconds}$$

or 65.53 ms

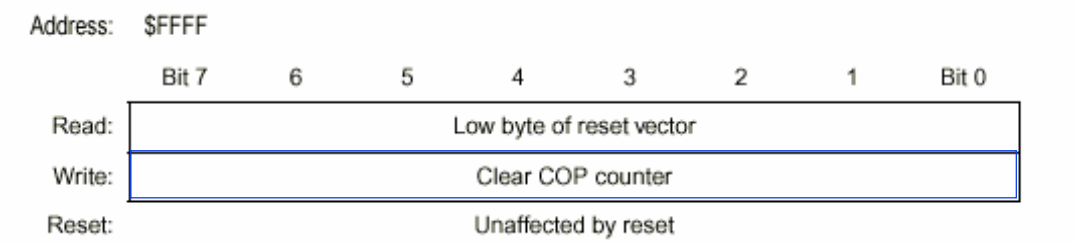
Computer Operating Properly Module Detail
Slide #5 of 7

Let's look at a specific example of how to calculate the COP timeout period. To calculate the COP timeout period in seconds, multiply the values of the 12-bit COP prescaler and the 6-bit COP counter and divide the result by the CHMXCLK frequency.

For this example, we'll set the COPRS bit to 1. This selects a COP timeout period of $2^{18} - 2^4$ CGMXCLK cycles. With the COPRS bit set to 1, the 12-bit COP prescaler will count 2^{13} cycles, or 8192. The 6-bit COP counter will count 2^5 cycles, or 32.

The CGMXCLK frequency is the same as the crystal frequency. For this example, let's assume a crystal frequency of 4 MHz. Multiplying 8192 by 32 and dividing the result by 4 MHz gives a timeout period of 65.53 ms. This represents the period in seconds that the COP must be serviced to avoid a COP reset.

COP Control Register (COPCTL)



- Clears COP counter
- Clears bits 12 through 4 of the SIM Counter
- Prevents reset
- Starts new timeout period

The COP control register, COPCTL, is located at address $FFFF and overlaps with the CPU reset vector.

Reading location $FFFF returns the low byte of the CPU reset vector.

Writing any value to address $FFFF clears the COP counter and the SIM Counter. This prevents a COP reset and starts a new timeout period.

System Protection Methods

- Use one instruction, or one section of code, to refresh the COP.
  - Include refresh instructions in main code loop.
  - Avoid using in subroutines.
  - Avoid using in timer or interrupt routines.
- Service COP within the minimum timeout period.
  - See the technical data book for more information.

Next, let's review some methods you can use to protect the system from runaway software.

To protect the system from runaway software, you should use one instruction or one section of code to refresh the COP. In a runaway situation, the software may get caught in a loop of code that refreshes the COP. If this happens, the COP will be refreshed when the code is not executing correctly. The device will not reset in this case.

In some cases, you can't avoid using multiple refresh instructions throughout the code. One example is when there is a possibility that the refresh may not occur within the timeout period, even when the code is executing correctly. If it is necessary to use multiple refresh instructions, try to keep the number of instructions to a minimum.

The refresh instructions should be in the main code loop. When the refresh instructions are in the main code loop and the device operations are correctly monitored, we can be confident that the COP is being refreshed correctly.

Avoid using refresh instructions in subroutines. However unlikely it may seem, the subroutine might be called by runaway code, especially if the watchdog subroutine address is in the stack RAM. Also, avoid using refresh instructions in a timer or other interrupt routines. Although it may appear convenient to refresh the COP in a timer interrupt routine, it is not good practice and should be avoided if possible.

There may be instances when interrupts are the only possibility to refresh the COP. One example of this is when the device is put into wait mode to reduce power consumption. In this case, one work-around you can use is another routine that sets bits in RAM when these routines are operating correctly. These bits can then be tested in the interrupt routine providing some level of confidence of correct operation. If the bits are incorrect, the necessary recovery action may be carried out.

You must always consider the minimum timeout period for the device. If you do not service the COP within the minimum timeout period, the device will be reset if the COP is enabled. The minimum timeout period is provided in the device technical data book.

Question

Which clock source is used as the input to the COP counter?

a) Internal output of the PLL
b) Crystal frequency
c) Bus frequency
d) User-selectable

Let's complete this tutorial with a question to check your understanding of the material. Which clock source is used as the input to the COP counter?

Answer: The crystal frequency, CGMXCLK, is the source for the COP counter.

Tutorial Completion

- COP Uses and Features
- COP Configuration and Servicing
- System Protection Methods

In this tutorial, you've learned about the features of the 68HC08 Computer Operating Properly Module and how to configure the device using the system configuration register. You've also learned how to service the COP timer using methods that protect the system from runaway code.