

NEED A WATCHDOG FOR YOUR MICRO? CHOOSE ONE THAT COMES ON A LEASH!

By: Elnaz Shayesteh

Abstract: Watchdog timers increase the reliability of microcontroller-based systems. However, they are not infallible. In this design solution, we review the function and operation of watchdog timer circuits, explore their limitations, and propose a family of watchdog ICs that help overcome these shortcomings.

Introduction

You wake in the middle of the night to the sound of your loyal dog, Sam, barking loudly. You lie there listening for a moment, before arriving at the usual conclusion that he was most likely disturbed by the noise of a random wild animal passing through your backyard, on its nightly prow. Indeed, a couple of minutes later, the barking stops, and you feel comforted as you begin to drift off once again. Suddenly you jolt awake—has the dog stopped barking because he’s gone back to sleep or, just maybe, he’s been silenced by an intruder? Eventually, your curiosity gets the better of you and as you go downstairs, you berate yourself for having watched too many crime thrillers. After assuring yourself that all is in order, you slip back under the covers and resume your trip to the land of nod. Watchdogs are great for security, but while the bark of a dog can be reassuring, it can also be worrying, meaning that just occasionally, you need to do a manual check.

When it comes to monitoring unusual activity microprocessor watchdog circuits, like their canine equivalent (**Figure 1**), are reliable and trustworthy, but unlike our four-legged friends, are not intelligent. As microcontrollers find their way into an increasingly diverse range of applications, the performance of the watchdog circuit, once considered trivial (and to some extent taken for granted) must be re-assessed. In this design solution, we quickly review the function and operation of the watchdog timer circuit. We discuss their limitations and the serious implications this can have for some applications before presenting a family of watchdog timer ICs that provides a failsafe to address these shortcomings while also conferring several other benefits.



Figure 1. Dog keeping watch for unusual activity.

Watchdog Timers

Micros executing critical or safety-related functions demand a high level of supervision to ensure that faults can be properly detected and corrected. A critical function can be defined as one for which downtime cannot be tolerated, and (in many cases) one for which repair is very costly. Such functions are found in almost every segment of the micros market: patient-monitoring systems, process-control plants, and safety-related automotive applications, to name a few. A micro is often subjected to power-supply transients, electromagnetic interference (EMI), and electrostatic discharge (ESD). These can cause it to execute erroneous instructions. To prevent this, a watchdog timer is a useful peripheral that can help catch and reset a micro that has gone "out of control." A watchdog timer is a simple countdown timer which is used to reset a micro after a specified interval of time. In a properly operating system, software will periodically restart the watchdog timer. Once restarted, the watchdog begins timing another predetermined interval. If the micro is functioning correctly, software will restart the watchdog timer before it times out. If the watchdog timer times out, it resets the micro. If the system software has been correctly designed and there has been no hardware failure, the reset will place the system into a known good state and begin to operate properly again.

Description of Operation

Figure 2 shows a simple watchdog circuit arrangement. The micro is programmed to send pulses to a watchdog timer IC at specified intervals. If the watchdog timer input (WDI) is not toggled within that time, it sends a pulse to the micro warning that a fault has occurred. The warning signal can be a reset to the micro, or a narrow pulse fed to the micro's non-maskable interrupt (NMI) port. A fault can be caused either by a code-execution error or an error in the timing circuit that generates the WDI pulse.

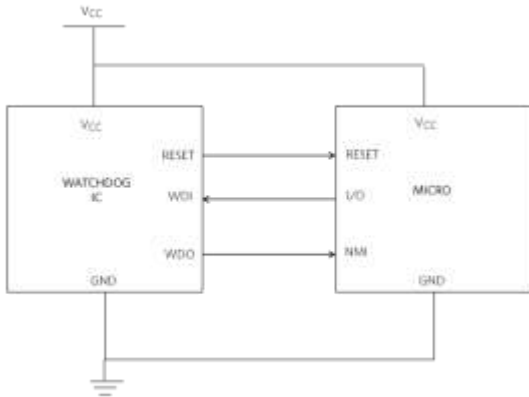


Figure 2. Typical watchdog timer circuit.

A simple timing diagram for this arrangement is shown in **Figure 3** below.

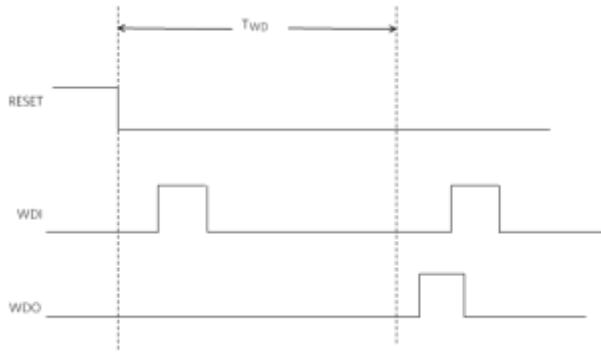


Figure 3. WDO toggles if WDI does not toggle within the watchdog window.

Some watchdog timers provide extra security by warning the micro not only when WDI toggles too late, but also if it toggles too soon. The difference between the upper- and lower-time intervals is called a “watchdog window.” As the micro executes a routine, it typically sets a flag at some point in the code. At preset intervals, it responds to an interrupt service routine (ISR) which notes, among other things, whether the watchdog flag has been set and if so, then it sends a pulse to WDI. If not, the micro is assumed to have hung up; perhaps executing an endless loop. A fault in the micro could also mean that the ISR was executed too often.

Not Bad ≠ Good

Regardless of how capable a watchdog timer might be, there are certain failures that cannot be corrected by a reset. For instance, a watchdog timer cannot prevent or detect corrupted data memory. Unless corrupted data affects program flow, watchdog timeout will not occur. Also, it should be noted that a watchdog timer cannot detect a fault instantaneously. By definition, the watchdog timer must reach the end of its timeout interval before it resets the processor. Clearly, in a time-critical application such as a continuous glucose monitor or an insulin pump where delay cannot be tolerated, it is imperative that the watchdog timer circuit has an override facility which can be activated immediately if the micro becomes stuck in an infinite loop or begins to execute corrupted data.

Watchdog on a Leash

The [MAX16152](#) and [MAX16153](#) (**Figure 4**) address these concerns by providing a manual reset (MR) input to allow an external pushbutton or logic signal to initiate a reset pulse. RST and WDO are provided as open-drain outputs.

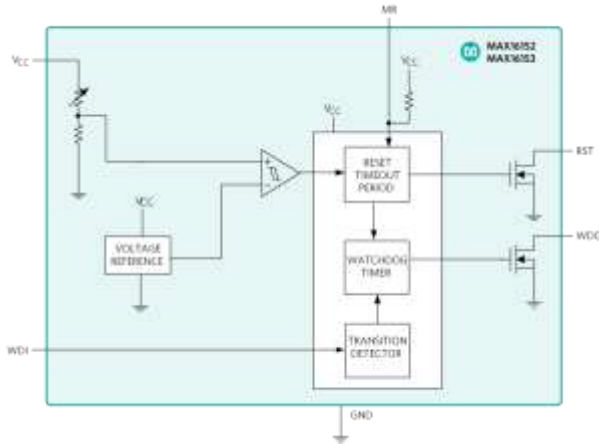


Figure 4. Functional diagram for the MAX16152/MAX16153.

An added advantage of these ICs is that they can also function as ultra-low-current (400nA typical) supervisory circuits to monitor a single system supply voltage. The reset output is asserted whenever the V_{CC} supply voltage is greater than the minimum operating voltage, but less than the reset threshold. After the supply voltage rises above the reset threshold, the reset output remains asserted for the reset timeout period, and then de-asserts. Reset voltage thresholds are available from 1.50V to 5.0V in approximately 100mV increments. The MAX16152 is offered in a tiny 0.86mm x 1.27mm 6-bump WLP while the MAX16153 is available in a 6-pin SOT23 package.

For less critical applications (e.g. personal computing devices) that do not require a manual reset, two additional family variants, the [MAX16154](#) (6-bump WLP) and the [MAX16155](#) (6-pin SOT23) instead provide a logic input (WD_EN) pin that allows watchdog functionality to be disabled when the system microcontroller is in “sleep” mode or is not executing code. This can be used to prevent the watchdog IC from sending an interrupt (i.e. waking) to the microcontroller unnecessarily during these times.

Summary

In this design solution, we have reviewed the function and operation of watchdog timer circuits and explored their limitations. Memory errors and infinite loops are two microcontroller errors that a watchdog timer cannot always detect and reset. For applications where this is unacceptable, it is important to use a watchdog timer that can be manually reset. We have presented a family of nanoPower watchdog timer ICs that provide this option, along with the added benefit of acting as a supply voltage supervisor. Very low current consumption and tiny package size make these ICs ideal for multiple battery-powered applications including portable computing, metering, and medical wearables.

Related Parts

[MAX16152](#) nanoPower Supervisor and Watchdog Timer

[MAX16153](#) nanoPower Supervisor and Watchdog Timer

[MAX16154](#) nanoPower Supervisor and Watchdog Timer

[MAX16155](#) nanoPower Supervisor and Watchdog Timer
