

Re: 8051 BIT

Source: <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2004-05/1292.html>

From: Jack Klein (jackklein_at_spamcop.net)

Date: 05/18/04

Date: Tue, 18 May 2004 00:26:14 -0500

On 17 May 2004 21:42:41 -0700, ffled@yahoo.com (Stu Cazzo) wrote in comp.arch.embedded:

- > *Hi,*
- > *Looking for what's out there in terms of built-in self-test*
- > *C language code for an 8051 product. Code snippet pointers*
- > *or even ideas on what would constitute a good BIT test*
- > *for an 8051 product. Already have in mind, a RAM test.*
- > *Any ideas on what else we can test for boot?*

Are you talking about internal or external RAM? You can write a test for the internal RAM, but the chances of a processor being damaged in just such a way that some of the internal RAM is damaged but the core still operates well enough to execute the test code and detect the error is probably vanishingly small.

In any case, my basic sequence for POST in embedded systems has been something like this for more than twenty years:

1. Disable all interrupts, so the rest of the POST runs continuously.
2. Set all outputs to their default, fail-safe condition. If your product controls motors, lasers, high voltages, etc., and the reset was caused by a power glitch or watchdog, the first thing you need to do is make sure all outputs are safely in an off condition.
3. Validate the firmware image, either the entire thing or, if there is some boot down loader and application, then at least the boot loader. Some form of checksum or CRC algorithm should match a precomputed value stored at a specific place in the image.
4. Test RAM (internal and/or external).
5. Initialize important peripherals, on-chip and off, and test them as far as is possible. This would include initializing the timer you will use for periodic interrupts and verify that it and the interrupt controller deliver interrupts properly. Other peripherals, for example external UARTs, often provide loop-back or other test modes

that you can exercise.

If any critical test fails, your processor should fall back to a fail-safe mode with interrupts disabled and use whatever means are available to signal the user and/or service technician what the problem is.

--

Jack Klein

Home: <http://JK-Technology.Com>

FAQs for

comp.lang.c <http://www.eskimo.com/~scs/C-faq/top.html>

comp.lang.c++ <http://www.parashift.com/c++-faq-lite/>

alt.comp.lang.learn.c-c++

<http://www.contrib.andre>