

Abstract

Power-on intermittent failures have many causes and fewer good methods to test them. But test we must! This article describes one tool for testing this class of failures.

The Problem

Power-on intermittent failures are nightmares for designers of electronic circuits and systems. These are the failures that occasionally occur at power up. They may occur as infrequently as 1 in 10,000 power-ups. There is really no limit to the way a power-on intermittent can manifest itself. It can be as benign as falsely failing a Power-On System Test or as serious as hanging a system with a loss of all its data. These failures are time consuming to detect, frequently hard to find and because of their intermittent nature difficult to verify when the correct fix has been made.

What causes them? In a word, bad design. Table 1 lists just a few of the design problems which can cause power-on intermittents. Since none of us ever create bad designs, they must primarily come from other sources. Most electronic systems today consist of a wide variety of components (VLSI chips, circuit boards, sensors, power supplies, and software). Some are designed in-house and others are purchased off-the-shelf. More and more systems today are using a wide range of complex off-the-shelf hardware (microcontrollers, FPGA, etc) and software components (graphics libraries, printer libraries, serial communication software, and operating systems). As the components we use become more complex, the potential for power-on intermittents increases dramatically. In just the last two years, we have come across power-on problems with:

- Chips (Cirrus GD6215 Video Controller, Xilinx FPGA)¹
- Off-the-Shelf Software²
- In-house software and circuitry (well, we hardly ever create bad designs)

What are the effects of these problems? For a variety of reasons, very few designs today are tested across 1000s of power cycles before they reach manufacturing. As a result, these problems are rarely found before either time of manufacture or by time of delivery. The cost to repair any design defect is well documented to be exponential with respect to time (see Figure 1). This results in costly re-work and in the potential loss of future sales.

The Solution

The solution is really quite simple. **Test. Test. Test.** Every design of even moderate complexity should be tested across thousands of power cycles. If the design uses AC power, these cycles should vary the phase angle at which turn-on occurs. The On and the Off times need to be varied

and a means of determining system health should be provided.

The problem is that most designers do not do this. Although expensive programmable power sources, controlled for example by the HP-IB bus, would readily do the trick³, this type of equipment is rarely available to the designer and takes too long to set-up and use..

MICROTOOLS' *Poc-it* (Power-On Cycler - Intermittent Tester) provides the means for both cycling AC and DC power, but also provides a means for detecting system health (see Photo 1).

AC Power Cycling - *Poc-it* provides a 10 amp standard 120 VAC three prong receptacle with On and Off times programmable from 0.01 seconds to 100 minutes. Since the power is switched asynchronously to the line phase, a random phasing is introduced into the test. *Poc-it* provides a simple means of reading out the number of AC cycles performed. Figure 2 shows a typical set-up for testing a switching power supply. AC power is provided by *Poc-it* to the power supply. As a means for determining whether or not the power supply becomes operational after power up, either the actual voltage or a power valid signal can be monitored. If the power supply has a 5 VDC output or a power valid signal, a 5 Volt input on *Poc-it* can be used. For other outputs, the optically isolated 10-30 Volt input can be used.

Figure 3 shows a typical set-up for testing a complete system. Again, switched AC power is provided by *Poc-it*. A signal that becomes valid after the unit has powered up can be used as an indication of system health. For example, once a unit has successfully powered up, it may begin scanning the keyboard. In this case, the keyboard strobe can be used as the system health indicator. On a PC based system, the LPT1 strobe signal could be used. On other systems where no such health output is provided, one could be temporarily added for purposes of verification.

AC/DC Power Cycling - Figure 4 shows how you would set up *Poc-it* to test a circuit or circuit board. In this case, board power will be switched by *Poc-it* via a set of banana jacks on the front panel. System health can be monitored as described above. A 10 amp relay is provided for switching both DC and AC. On and Off times are also programmable from 0.01 seconds to 100 minutes.

System Health Monitoring - Three digital inputs are provided for monitoring system health: two 5 VDC high speed inputs and one optically isolated 10-30 VDC input. *Poc-it* provides a means for delaying the monitoring of this system health input to allow it to be indeterminate during some period of the power-on cycle.

Basic Operation - *Poc-it* set-up and operation is simple and quick. Once the On and the Off times are programmed, the input sample delay time is set and the test sequence is started. The test can be monitored while in process. After several thousand cycles, the test may be stopped. The number of power cycles should be equal to the number of times the System Health input was

toggled. If these numbers do not agree, a problem has occurred.

Summary - We have a motto at MICROTOOLS:

If it's not tested, it doesn't work.

Designing complex systems has confirmed the truth of this motto for me. I am embarrassed to admit how often obvious designs don't work when I test them. One significant area is in power-on testing. No matter whether you use **Poc-it** or some other solution, test your designs over 1000s of turn-ons. Seek out those failures before they seek you out.

Table 1
Design Problems that can Cause Power-On Intermittents

Hardware Problems	Software Problems
Improper hardware initialization	Incorrect hardware initialization
Temperature sensitive race conditions	Incorrect software initialization
Vibration sensitive components	Unprotected interrupt windows
Component sensitive race conditions	Power-On System Test problems
Voltage or phase sensitive conditions	Hardware/Software timing problems
Noisy or noise susceptible power circuitry	Uninitialized data regions
Intermittents	

Figure 1 Relationships between defect detection and time⁴

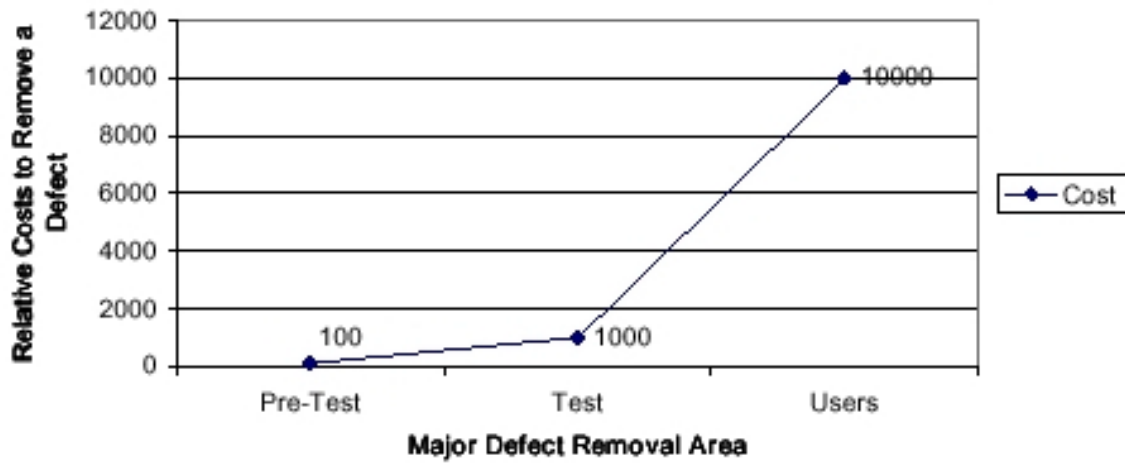


Photo 1 *Poc-it* (Power-On Cycler - Intermittent Tester)



Notes

1. "It sometimes takes a microcontroller to boot up a FPGA", Scott Rosenthal Personal Engineering, November 1994.
2. Greenleaf's Comm+ Communications library came with three very subtle bugs that manifested themselves at power-up. Two of them provided interrupt windows of opportunity for floating hardware lines to hang the system.
3. For example, HP 6800 Series ac Power Sources and HP 6600 Series dc Power Sources can be programmed from a PC over the HP-IB bus to perform AC and DC power cycling. The problem is both the cost (greater than \$2,000 per unit) and ease of use (i.e. programming the PC to set up the power cycling). This solution is typically used in a manufacturing acceptance test environment.
4. Glenford J. Meyers Software Reliability, Principles and Practices, John Wiley and Sons, New York 1976