Don't trust your tamper detection circuitry, it may be dumb?

Mohit Arora - July 26, 2012

Often products use anti-tamper switches in a variety of applications to be able to protect the product from being tampered with. For example, for an electricity meter, an external tamper condition may include breaking the meter case, chemical injection or even burning the meter.

All of these result in changing the electrical characteristics of the components, thereby recording less or no energy usage. One may want to open the meter case to change the settings or even remove the backup battery so that the meter will reset when the main power goes off.

Anti-tamper switches can be placed on the casing of the meter to trigger a tamper event when the casing is opened.

Detecting Tampers External to the System

There can be several attacks that a system may have to face from the external world. These may include damage to the casing of the system and alteration of certain signals, etc. These intrusions can be monitored by anti-tamper switches in the system. Since these anti-tamper switches need to be monitored at all the time, they need to be powered by a battery (RTC) supply.

These anti-tamper switches can be prone to noise and can cause false tamper conditions. Thus it is important to filter out these noises to prevent incorrect triggering.

![Figure 1: External Tamper Detection](image)

The tamper event should be one of the sources of interrupt to the CPU. The CPU, on the event of tamper, can take necessary actions like erasing any secure information, generating system reset, storing the tamper event in EEPROM or battery backed registers, and finally clearing the interrupt flag. The CPU response to a tamper event is generally application specific.

For a typical application, including the electricity meter here in this example, it is important to note that once a tamper signal is asserted, it should not be cleared unless both the main (VDD) as well as battery supply (VBAT) is removed. When the supply is reconnected, the tamper condition should be the default condition and should only be reset by code within the processor. For example, in an electricity meter, this is normally done during meter calibration.
An inherent disadvantage of using open or passive anti-tamper switches (Figure 1) is that with the passage of time these switches tend to get oxidized (since there is no passage of current until the tamper event, and these switches are otherwise in contact with oxygen, thus getting oxidized or corroded). And when a tamper occurs these switches remain open due to the oxidation and thus a tamper event may never be indicated to the system.

A simple workaround could be to use a normally closed switch, which ensures a constant passage of current to avoid it getting oxidized. However it suffers from the same fact that there is no feedback path to ensure that the anti-tamper switch is healthy and working as expected. This is overcome by the active tamper detection technique described next.

**Active Tamper Detection**

Active tamper detection introduces a feedback loop providing a more advanced method of monitoring external tampers and also ensuring an extended life of anti-tamper switches. Unlike passive tampers that are input only, an active tamper mechanism includes a pair of one or more input/output switches.

A chip outputs a known sequence (fixed or generated by a linear feedback shift register) on the output anti-tamper switch while monitoring the input tamper switches for the same sequence (as shown in Figure 2). As long as the sequence matches, no tamper is indicated. When the sequence skips a value or is incorrect, either due to an external tamper event or fault in the switch, a tamper event is activated.

![Figure 2: Active Tamper Detection](image)

A major benefit from the technique is that it allows integrators/product manufacturers to create security zones on the board such that a single trace from the TAMPER_OUT to TAMPER_IN covers several components on the board so that any manipulation (for example, removing a component and replacing it with a faulty component on the board) breaks the path and creates a tamper condition.

In addition, the pattern push out time (time between the individual bits through the TAMPER_OUT) should be no more than 500 ms to 1000 ms to ensure bad guys cannot intercept and inject their own pattern (very unlikely) through human intervention attacks.