Understanding the evolving landscape of SoC vulnerabilities and analog threats

SoC attacks call for analog and digital defenses

Version 1.00 • July 2020

Table of Contents

Security

The vulnerability landscape

Many SoC integrators are only too aware that security enforced only by software is highly vulnerable to attack. All that a hacker needs to do is find a way to replace key parts of the bootloader or the low-level firmware to compromise other software in the system used to support secure access.

The easiest attacks we see are those conducted remotely over a network, with so many examples of zero-day attacks due to open root access accounts being left unsecured (the Exploiteers' 20 Devices in 45 minutes video gives some examples). When identified, these can be patched remotely with software upgrades. However, there are increasing numbers of examples that criminals have found lucrative which involve physical access, and for which the impact cannot be mitigated by software upgrade.

An extreme example of this that has been deployed in real-world attacks is the spate of "jackpotting" attacks on automated teller machines (ATMs) that hit the EU and US over the past three years. In these attacks, thieves use physical access to replace legitimate hard drives containing the core ATM software with their own version of the operating system and applications, or introduce spyware devices that plug into an unprotected USB port. Others have infiltrated the banks' core networks to plant malware on the targeted machines.

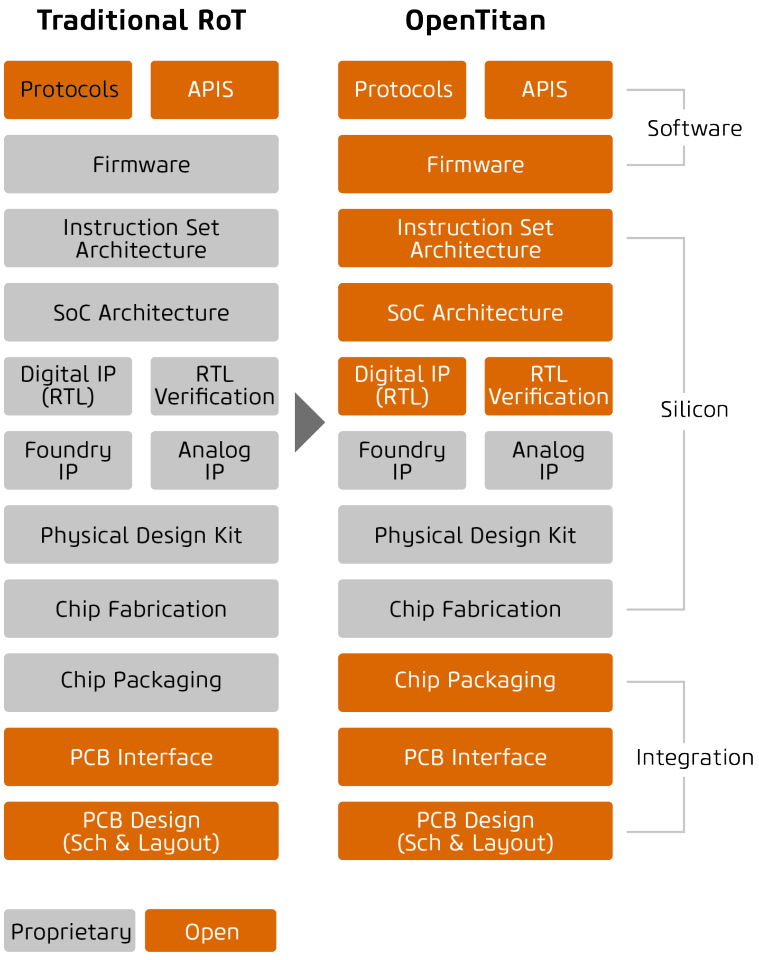


Figure 1: OpenTitan introduces a security focused hardware module into the core machine.

By introducing a security-focused hardware module into the core machine design, for example the OpenTitan project, manufacturers can prevent many remote and local attacks. The hardware module provides a root of trust, a module on which the system should always be able to rely (see Figure 2). This module supports functions such as a secure or measured boot, which ensures only firmware hashed against a known signature or security certificate is allowed to run. Any modifications that are applied in the absence of the code-signing processes will fail the hash test applied by the root of trust and will be terminated by the bootloader.

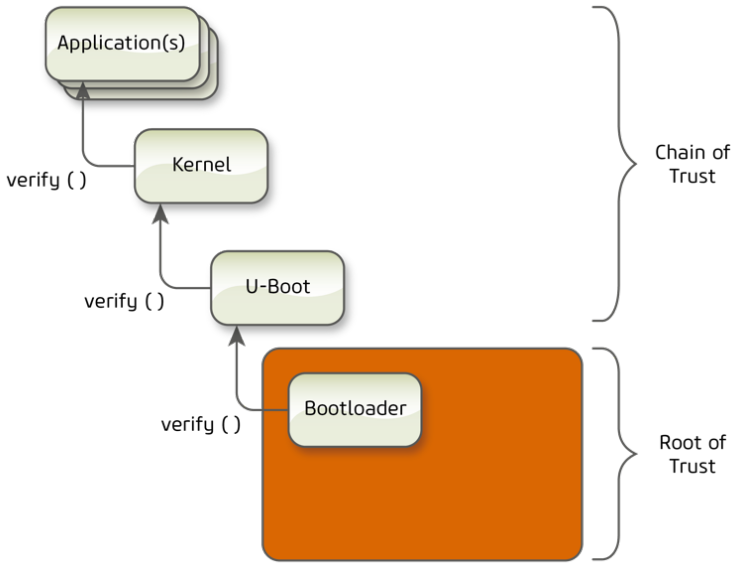


Figure 2: the root-of-trust model.

A protected cryptocontroller is used to implement the root of trust. This can act as a secure enclave used to perform any sensitive operations that need to be protected against intrusion or spying. But even the root of trust and similar secure enclaves are vulnerable to attack if the implementation does not take account of the techniques attackers can deploy to subvert its protections. These attacks often exploit the gap between the theoretical security offered by encryption protocols and behavior of the hardware and software functions that implement them.

Side channel attacks

Very often, the way in which algorithms are implemented provide hackers with important clues not just to how they operate but the sensitive data, such as private keys, they work with. The degree to which a user of a system can acquire this information and use it in an attack depends not just on the protections employed but the degree of access that the attacker has to the target system and the degree to which the secure enclave is isolated from less sensitive circuitry.

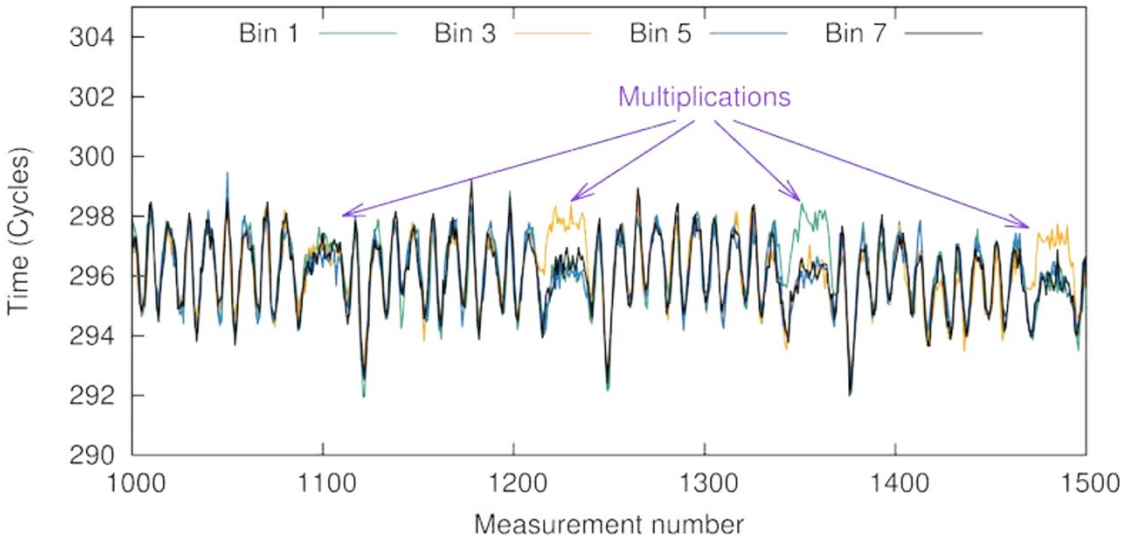


Figure 3: a cache side channel attack observes small variations in the execution timings of operations due to cache contention. Combining this information with knowledge of the underlying algorithm (in this case RSA) allows the attacker to deduce the key values (or at least dramatically reduces the search space for a trial-and-error attack). The figure highlights the multiplication portions of the process; the distinctive negative spikes are squaring and modular reduction.

Caches, execution pipelines, electromagnetic (EM) emissions and instantaneous changes in voltage and current on power rails all provide clues as to what a target is doing. Whereas EM and power signatures call for local access, cache and pipeline behavior can be tracked remotely as shown in proof-of-concept attacks such as Prime+Probe, created by researchers at the Weizmann Institute of Science, as well as the more heavily publicized Meltdown and Spectre techniques. These are more likely to be used on cloud servers where the adversary has little chance of penetrating the data center's physical security.

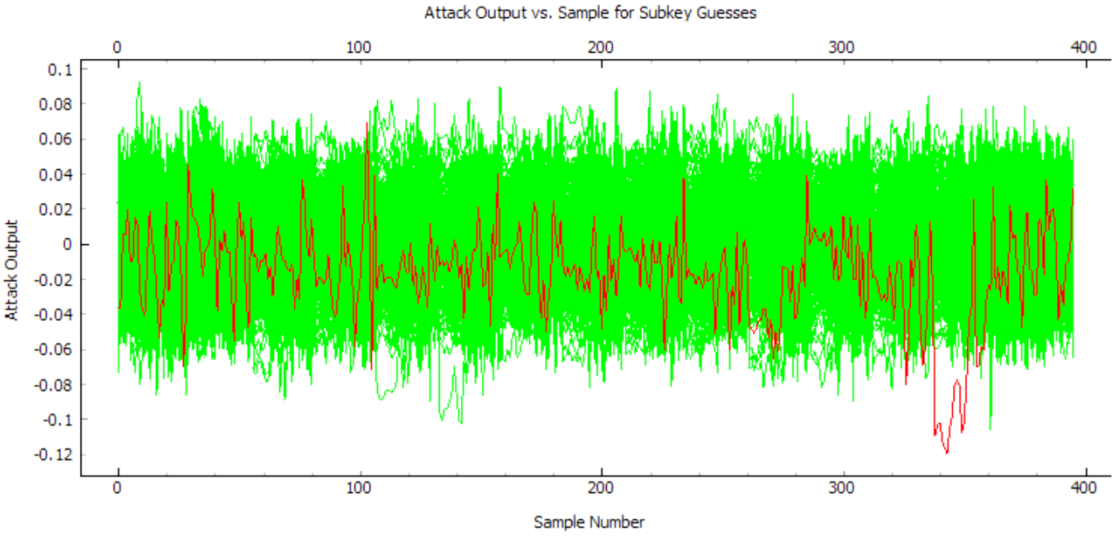


Figure 4: Attack output vs sample number for subkey guesses. The leakage that lets an attacker deduce each key byte is frequently isolated to specific groups of calculations and is often identified by changes in correlation. In this example, a large change in correlation in the region around 350 samples shows the results of a correctly guessed key byte.

But compute is expanding at a far greater rate outside the data center than within, and these devices have increasing access to our secrets and our personal information. The rollout of services such as 5G anticipates strong growth in the use of edge servers and smart IoT devices that are able to run AI and other more advanced algorithms with far lower latency issues than systems that rely heavily on remote cloud computing. They are being joined by increasingly autonomous vehicles and robotics systems that can be regarded as edge servers, as well as enticing targets for hackers.

These devices are far more vulnerable to intrusive attacks that are able to subvert hardware-enforced secure enclaves. The forms of attack vary by a great deal and become increasingly difficult to counter as the degree of control that an attacker can exert on the target increases, ranging from the execution of many repeated operations on a target crypto core to analyze its internal operation, through manipulations of voltage, clock and temperature, all the way to decapping the device and inserting probes that interfere with normal operation. Although these latter attacks are generally restricted to systems that offer a high value to the instigator, recent work from chip.fail shows that with less than $150 of lab kit, a wide variety of IoT devices can be hacked.

The changing nature of attacks

In the past, outside highly targeted attacks used by state actors on systems with a military or national-defense role, the main focus of attacks that require physical presence was on consumer-facing, financially sensitive devices: smartcards, point-of-sale terminals, and pay-TV decoders. Cracking the protection on these devices was lucrative enough for criminal gangs to invest in sophisticated tools to understand how they work and obtain enough information to either make clones for sale or to compromise the protection enough to siphon off money directly.

However, the increasing integration of edge devices into large-scale distributed systems provides motivated groups with an increasingly large attack space. In order to maximize their potential for success, they will use multiple attack types. Sometimes, this is a matter of trying different approaches until one works. But, thanks to the use of more advanced statistical tools and machine learning, malicious users are combining information from multiple sources in order to reverse-engineer a target and increase its vulnerability to the final attack.

A simple example of the use of machine learning in attacks is found in side-channel analysis: an approach that has been used to try to obtain the private keys employed by an embedded cryptocore. The attacks rely on the way in which emissions from the circuitry, often seen as electromagnetic interference or as supply-rail fluctuations, reveal information on the data being processed. By collecting a large number of traces – 10,000 or more – statistical tools can be used to locate samples that provide the strongest clues. Countermeasures such as noise injection or random masks applied to the key bytes at important points can hide these telltale transitions though they may fail to disguise all the operations.

Using even higher numbers of traces and providing them as training data to a deep-learning pipeline, a number of groups have shown that it is possible to obtain the private key in the presence of countermeasures. One issue that attackers face in developing side-channel attacks is that of repeatability. There are many variables involved in collecting accurate traces that will yield information. Training across multiple models and across a range of devices through techniques such as ensemble learning helps reduce variability in the predictions and increase the probability of deriving keys accurately.

Using more intrusive methods, attackers may increase their probability of deriving information about a system or even gaining a higher degree of control that allows them to read out sensitive information directly.

Though they are only practical in situations where the attacker has full physical access to the target over a long period of time, these techniques can be extremely difficult to fight because they focus on elements of circuit design that are generally assumed to be well behaved in a system that passes its manufacturing tests.

Depending on the overall system architecture, this information may lead to the ability to compromise an entire network of devices if a private key is exposed, or access to a privileged account is gained.

What can physical attacks do?

A common aim for users of intrusive physical attacks is to take direct control of the target by performing some kind of privilege escalation. A glitch inserted at the right moment and on a particular part of the SoC may force it to skip a normally mandatory authentication check. An example of this, shown recently by LevelDown security, was through the following code sequence:

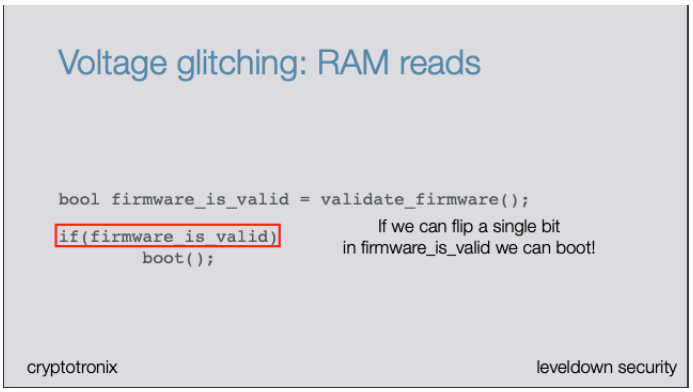


Figure 5: The ability to flip a single bit at the right time, achieved through monitoring of the clock whilst glitching the supply, causes the microprocessor to discard a failed firmware validation step and proceed to boot anyway (https://chip.fail)

An older practice with microprocessors was to use illegal opcodes to try to find states that may trigger execution units in unexpected ways and potentially access target memory locations or corrupt the pipeline. At minimum, illegal opcodes often trigger exceptions that may lead to internal information being exposed to the hacker on the memory bus.

Some illegal instruction attacks have not just exposed bugs in pipeline-controlled state machines. The Project:Rosenbridge "God mode" attack on the VIA C3 processor managed to activate hidden processor cores inside the device that were supposed to remain inaccessible to end users. Once active, they could run arbitrary code without checks by the operating system managing the legitimately active cores and, in doing so, gain root privileges.

Although many processor cores now routinely check for valid instruction opcodes and operations, other vectors often remain. By targeting the clock and power rails, an adversary can potentially force the execution of illegal operations even with valid code by taking advantage of an artificial mismatch between the speed of combinatorial logic and clock signals that latch the data they produce into registers. As almost all SoCs are designed according to a synchronous methodology, a production-grade device is assumed to have combinatorial-logic paths that, when provided with data from a source register on the start of a clock cycle will deposit a correct, updated value into its relevant destination registers by the end of that cycle (See Figure 5).

The propagation delay is dependent on a number of factors. Some are fixed, such as intra-die process variations. However, temperature and supply rail voltage have a strong influence on switching speed and both are potentially under the control of an attacker with physical access to the device.

Temperature attacks

The response of a device to changes in temperature can be complex. Traditionally, transistors tend to switch faster at a given supply voltage as they are cooled. However, this relationship has become more complex with the use of processes that deploy transistors with multiple threshold voltages. Whereas devices with low threshold voltages, which are generally designed for performance, tend to speed up when run cooler, those with higher thresholds can slow down, a phenomenon known as temperature inversion dependence. Though multi-corner simulation provides the information design teams need to compensate for these effects at design time, these temperature dependencies can lead to unexpected results when a circuit is operated beyond the temperature range used for verification.

Many attacks will heat the SoC to beyond its normal range in order to trigger errors but some will cool the device to well below freezing in order to trigger other faults. Another benefit to attackers from rapid cooling of a target is to prevent the contents of volatile memory arrays from being erased if the system is halted. A number of proof-of-concept attacks have been published since the effect was demonstrated

by researchers at the University of Cambridge in 2002. One recent attack froze the target to –110°C using liquid nitrogen to extract private keys from a physically unclonable function (PUF) array built from an on-chip SRAM. The target was halted during boot, just after the key was generated by the PUF and stored in on-chip memory, and then probed using heat or laser stimulation.

Excessive heat can also be used to take advantage of abnormal data remanence. In this case, the memory contents are imprinted by degrading the transistors through ageing effects such as negative bias temperature instability. High-temperature attacks can be used to induce memory errors through hours of extensive heating as well as mistakes in combinatorial logic trees.

Voltage attacks

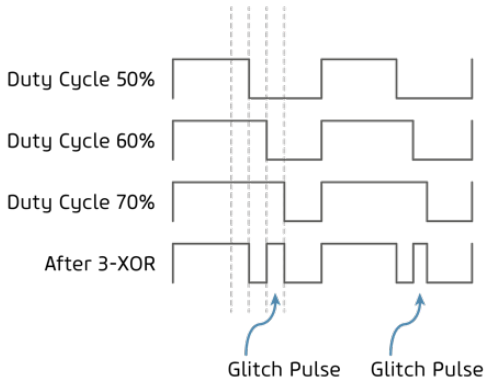
The effect of supply-rail changes on circuit behavior are generally easier for implementors to predict. A reduction pushes the transistor closer to its threshold voltage, which in turn slows down the switching speed. If pushed too far, critical paths will fail to complete before the next clock cycle begins. Logic fed by a victim flip-flop or register array is more likely to contain incorrect data.

Attackers will experiment with voltage levels to see how the victim device reacts up to and including temporary brownout conditions. As many complex SoCs require multiple power rails, and often reset generators have a slow response and low sensitivity to brown-out, an adversary can focus on those that are more likely to yield results. The processor core rails are often chosen as they can yield better control over datapath operations. With modern methods and access to PCB data from public filings, it is simple for attackers to cut the PCB traces and introduce their own voltage controllers. Many remove decoupling capacitors to improve the resolution of any voltage and current changes they introduce. As a result the victim device can easily be subjected to very short (100ns) brown-out conditions, which at the crucial time can flip the key bit resulting in the necessary result.

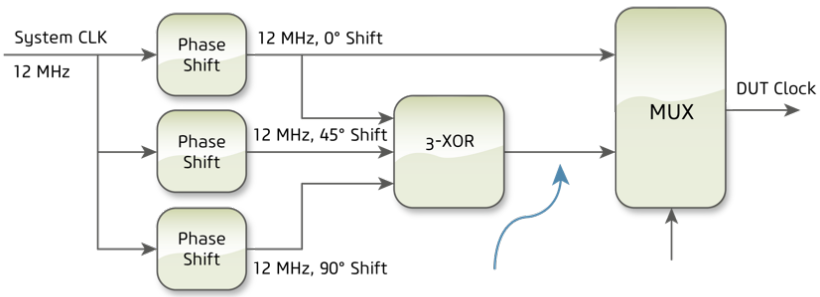
Clock attacks

Another option for the attacker is to modify device clocks directly if external clock sources are used. Bootloaders often operate direct from an external crystal, so that PLL settings can be tuned after production. Clock glitching or inserting false transitions will be interpreted as a short-term increase in clock frequency, causing registers to try to latch data early from the combinatorial logic chain that feeds them.

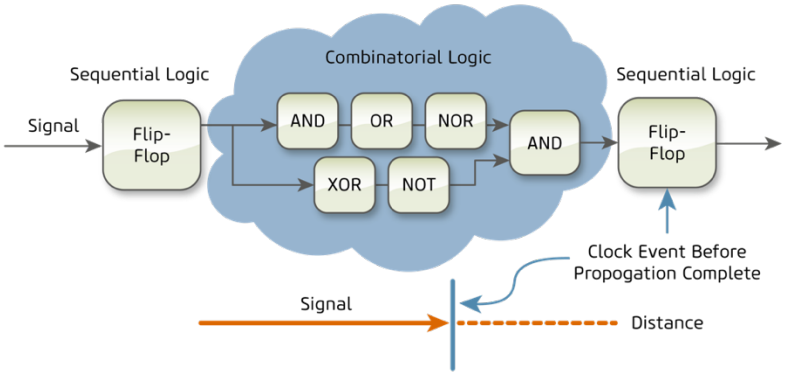
Figure 6: clock glitching is a common invasive attack, which can be implemented in a number of simple ways



6(a) Poly-PWM combines (by XOR-ing) three waveforms at a single frequency and with fixed phase, but variable duty cycle



6(b) The polyphase technique employs waveforms with common frequency and duty cycle, but shifted phase



6(c) Where sequential and combinatorial logic are combined, putting the set-up and hold time of a flip-flop out of spec is another approach. Here a clock event arrives at the second flip-flop before propagation through the combinatorial logic is complete

If fed by an external source, clock glitch attacks are relatively cheap to carry out and are even featured modes in off-the-shelf kits such as the ChipWhisperer. This uses a pair of phase shifters coupled with enable controls to selectively insert very fast clock transitions into the clock signal that feeds the target. As well as corrupting data values if arithmetic operations are being performed by the target's execution pipeline, glitch attacks are known to force faults such as skipped instructions and errors in instruction decoding, which causes incorrect instructions to be run on the next full cycle.

Even if the adversary is unable to force the wrong program path to be taken, data corruption that the adversary detects on a memory or system bus may be helpful: they can use small changes in the instruction and data feed to determine the behavior of a password-checking or encryption routine and so build a picture of what a successful input would be.

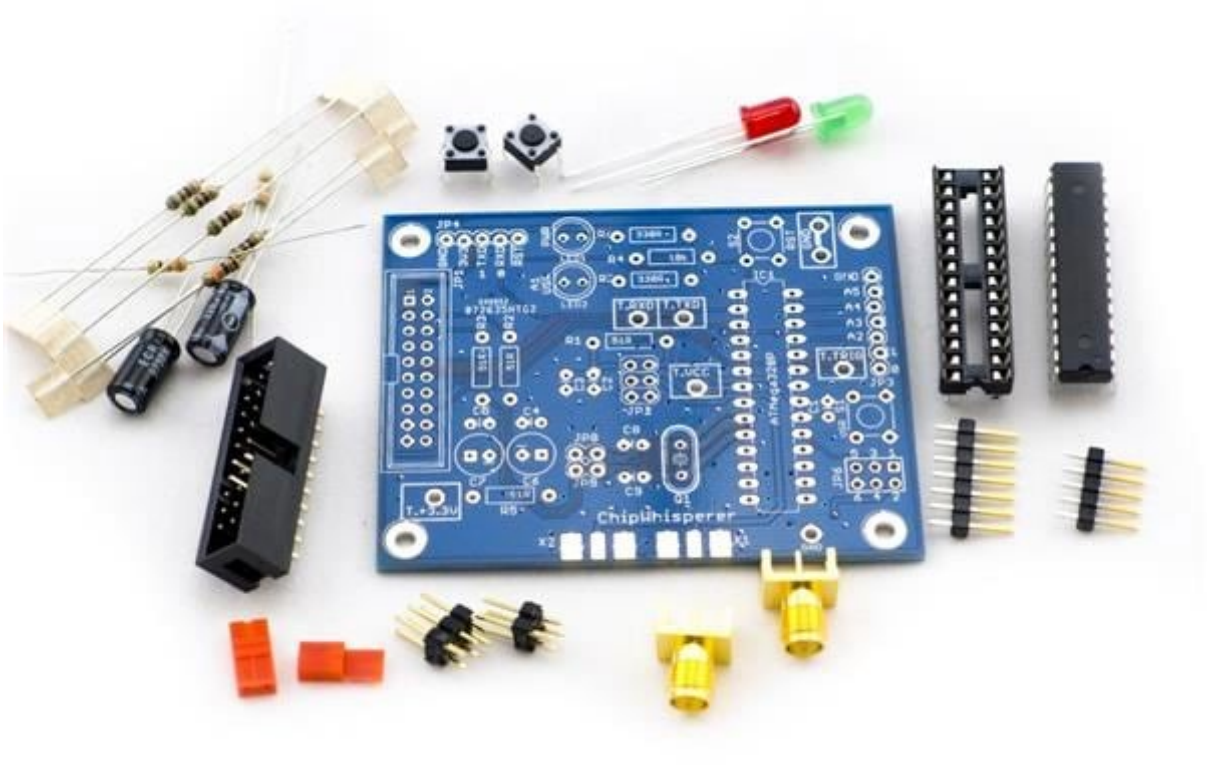


Figure 7: Off-the shelf kits such as ChipWhisperer Lite enable clock glitching (photo: Mouser).

A clock attack described by researchers at CEA-Leti in 2010 focused on the key used by an AES algorithm implemented on a field-programmable gate array (FPGA). The attack used a series of deliberate faults that created setup and hold errors in the encryption logic to force mistakes into the ciphertext that were then compared with the correctly encrypted message. Many of the errors altered a single bit in target key bytes. Successive rounds of clock-induced errors combined with targeted guesses yielded the target key bytes.

More sophisticated attacks

Other forms of fault injection attacks can be performed via electromagnetic (EM) radiation and lasers. Some of them have questionable value to attackers, though they can be implemented using comparatively low-end equipment. The ChipShouter is an off-the-shelf device that generates short-duration, high-intensity EM pulses that induce stray currents within the target device. However, in experiments a number of research teams have noted that, unless very strong pulses are used, EM pulses need to be timed to coincide with clock edges to generate repeatable faults.

Laser-induced fault injection using the infrared, visible or ultraviolet spectrum, provides greater targeting ability for an adversary though it involves far higher skill levels as it calls for at least partial decapping of the IC package without damaging the device itself. Similar to EM in the RF range, short pulses excite logic gates and can force errors.

All inputs are at risk

Some attacks on signals are far less obvious. An example was a proof-of-concept attack demonstrated in 2018. Researchers on a team with members working at the University of Michigan and Zhejiang University found it was possible to attack the controllers inside hard-disk drives through their vibration sensors.

Vibration sensors are used to protect the read heads from damaging the surface of the magnetic platters underneath – telling the controller to retract the heads if they detect a sudden shock that might otherwise force them to collide with the disk's surface. Fed with strong low-frequency pulses, the controller could misinterpret the incoming vibration readings with the result that the heads would damage the magnetic coating and destroy the data it contains. In other cases, the tactic worked as a form of denial-of-service (DoS) attack. The constant retracting and redeployment of the heads under the heavy, low-frequency vibration would reduce read-write rates to a crawl. In extreme cases, this caused timeouts and resets in the host system.

Rather than attack sensor inputs directly, some hacks take advantage of the distributed nature of many systems. A motor vehicle, for example, relies on a network of sensor modules feeding into an array of electronic control units (ECUs) that work together to power and steer the overall system. Such systems are vulnerable to attacks where a compromised module or one that has been inserted as a Trojan into the network can generate false signals. Traditionally, vehicle networks such as the Controller Area Network (CAN) were designed on the assumption that the network elements could be trusted. The protocol itself has no mechanism for checking the authenticity of messages passed along the bus. As a result, the protocol is vulnerable to a wide range of attacks that do not just include DoS but replay and false-frame injection that may be used to confuse or crash sensor and ECU modules. A combination of physical and remote accesses is enough to compromise the system, demonstrating the complex relationship between vulnerabilities that may make systems far less secure than they seem.

Countermeasure options

There are many possible countermeasures that implementors can decide to incorporate into their designs but they need to be assembled in a coherent manner with a clear system-level perspective. But traditional countermeasures have been focused on specific attack vectors. For example, microcontrollers designed for secure devices often employ not just hardware elements that are intended to confound side-channel attacks but also circuitry to react to supply-rail and clock glitches. However, it is notable that many IoT-grade microcontrollers are missing this protection, and yet the access to information can be just as damaging to the user.

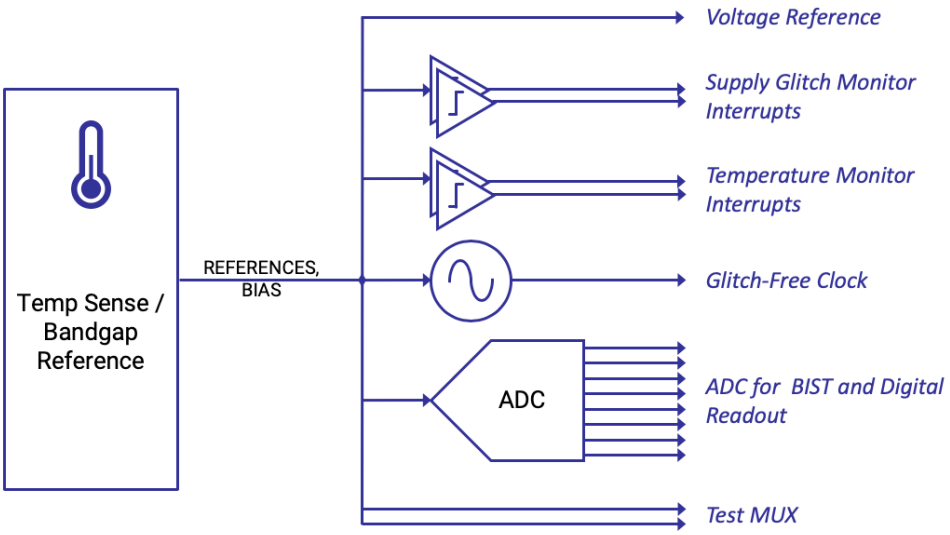


Figure 8: Agile Analog's monitors can protect against common attacks.

Some of these techniques are not even practical on higher-performance SoCs. For example, a common defense against clock-glitch attacks is to use an internally generated source which external clocks can be monitored against for consistency. This can prevent the majority of glitch-style attacks.

A similar option is employed in secure microcontrollers to handle power glitches. Conventionally, a brownout detector based on comparator circuitry watches for sudden droops in voltage and will force a reset to restart the device in order to enter protection state, zero out any sensitive data that may be cached in RAM and prevent any pipeline operations from proceeding. When the supply rails return to a valid state for a sufficient length of time, the brownout detector releases the reset pin.

When the supplied voltage rises past the threshold for a sufficient length of time, the controller releases the reset pin and the device is allowed to complete a reset, allowing execution to start from a known state and disrupting attempts by the attacker to alter program execution.

Forcing a reset on every detection of a power anomaly is acceptable in a smartcard or TV decoder. There is rarely an issue if the system restarts and terminates operation other than one of false positives interrupting legitimate use. However, in many circumstances more nuanced responses are advisable. It may make sense to terminate operations within the parts of the die that are considered sensitive but interruptible while others continue. But other reactions might be needed. For example, the SoC may need to switch to a mode that preserves safe operation and employ on-chip protection modes that are too power hungry to use in all situations but which can ameliorate some of the effects of intrusive attacks.

Operation from a lower-performance internal clock may be possible if the external clock is attacked. Similarly, direct access to core-voltage rails might be disabled with all power supplied through a primary in-package power-management unit, perhaps at a lower overall level that supports slower operation without disabling the entire SoC. In cases where interruption cannot be tolerated unless the SoC simply cannot function correctly the device may record the actions and report them over the network at the earliest opportunity.

Another reason for supporting more-nuanced responses to suspected attacks is the issue of false positives. In high-reliability devices built for operation in extreme environments, the practical difference in ambient temperature that is considered normal versus the heating or cooling used by attacker may be very little. However, high-resolution measurements from dedicated temperature sensors can provide the detailed information a controller may need to be able to tell a deliberate attack using rapid changes rather than typical environmental deviations.

Defense in depth

As so many reactions to intrusive attacks are conceivable, the key to handling them is an infrastructure that is secure, adaptive to different forms of intrusion and responsive. An important requirement for such an infrastructure is that it can operate independently of the main system logic so that it can react to events while target subsystems come under attack.

UltraSoC's solution embeds transaction-aware hardware monitors into the digital infrastructure of an SoC, all interconnected using a message-based architecture. Into that framework it is possible to integrate a wide range of system-control, debug and security-monitoring cores. Examples for digital and remote intrusion attempts are the Bus Sentinel and CAN Sentinel hardware modules. These can identify and, if required for the implementation, immediately block suspicious communications within the chip and on CAN ports. The company's partnership with Agile Analog extends the possibilities for intrusion monitoring to the physical, analog domain. Agile Analog's portfolio of monitors can provide continuous checks on clock, voltage and temperature within the SoC.

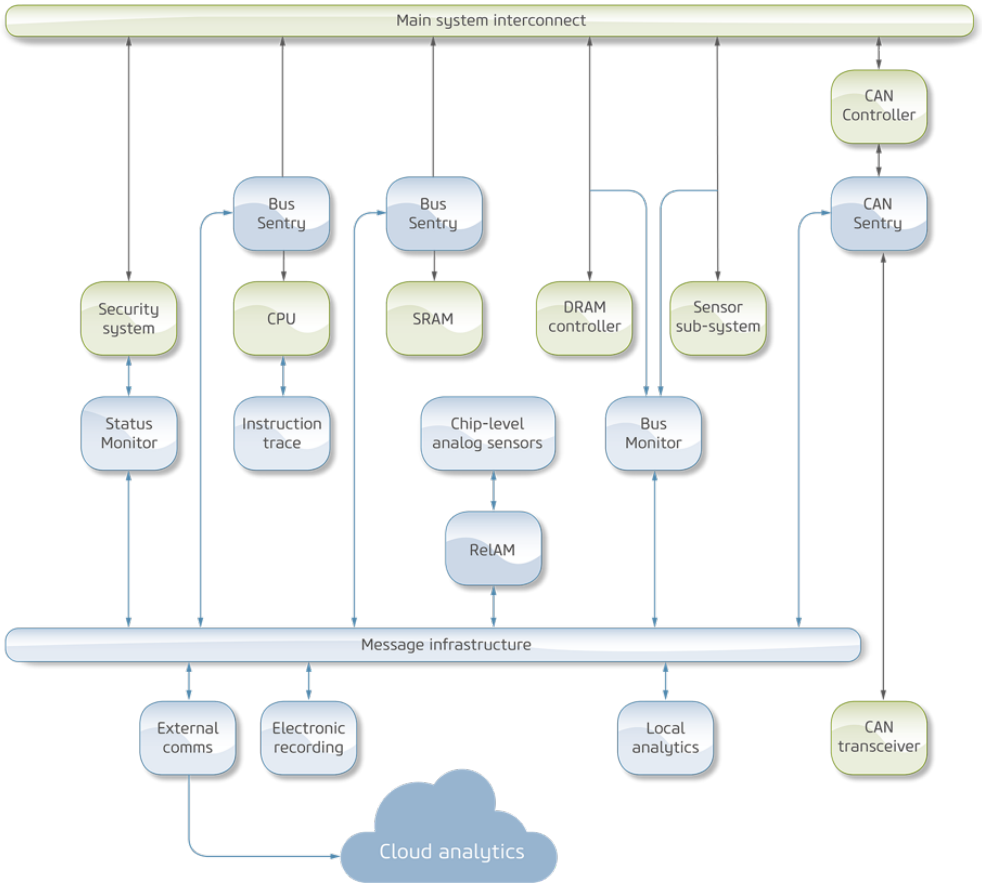


Figure 9: a complete on-chip security solution can detect and mitigate many threats.

By tying the monitors into a cross-chip infrastructure, SoC integrators can not only react to specific intrusions but combine information from multiple sources as well as locally stored history to tune the response as necessary. Because adversaries are increasingly turning to the use of blended attacks, which may involve the profiling of numerous victim devices before attacking their actual target, the ability to trace and report anomalies will be vital.

Using a cross-chip architecture also supports the ability to tune power consumption and processing overhead based on the threat level. For example, monitoring some analog channels, such as multiple power rails, on a continuous basis may be too expensive in terms of energy. Instead, the system may use a low-resolution sensor to detect brownout events or increases in noise on a high-importance target to activate a wider range of sensors to determine whether the event is likely to be malicious or the consequence of a system problem. Once it has enough information, the system controller can trigger countermeasures or issue a warning to a higher-level management system.

The infrastructure can readily be tempered for the use-cases of the SoC. The choice over which sensors and countermeasures to use will depend on factors such as accessibility and the value of the target to a hacker. In data center systems, an unlikely possibility is for an insider to place some kind of data-extraction or interference device close to a server blade. But even then, the attacker cannot easily be certain of which blades will be used by a given workload at any time given the ease with which applications and data can be moved around a data center using Kubernetes and similar dev-ops tools. As a result, such systems can adequately be protected by a combination of physical access security and countermeasures that focus on the behavior of software workloads, as well as container deployment procedures that isolate workloads handling sensitive data from other tasks.

Conversely, systems that go into factories, vehicles and the home need far more physical-security protections and analog-sensing modalities built into them to protect against not just direct intrusion but profiling studies that may aid a later attack that is conducted using remote access through, for example, the RF module in an infotainment system.

As malicious adversaries have proven with their attacks on real-world systems, information is power. With a full-chip security infrastructure that incorporates smart controllers and both analog and digital sensor modules, SoC integrators can use their access to real-time information to turn the tables.