Demo: Analog Trojan Design, Fabrication, and Detection**

Yumin Hou*, Hu He*, Kaveh Shamsi†, Yier Jin†, Dong Wu* and Huaqiang Wu*
*Department of Microelectronics, Tsinghua University
†Departement of Electrical and Comptuer Engineering, University of Florida
yier.jin@ece.ufl.edu
**The hardware demo is based on a HOST 2018 paper titled "R2D2: Runtime Reassurance and Detection of A2 Trojan"

***Abstract*—This demo will show a fabricated analog hardware Trojan detection method. This design mainly targets on A2-like Trojans which are triggered by a successive toggling events. The principle of the detection method is to guard a set of concerned signals, and initiate a hardware interrupt request when abnormal toggling events occur in these guarded signals. We design a processor based on ARMv7-A&R ISA, and insert an analog Trojan into the processor. The detection mechanism is also implemented in this processor. We also fabricate a proof-of-concept chip in the SMIC 130 nm Mixed-Signal 1P7M process and demonstrate that the analog Trojan works on the ARM processor, and the detection method is effective in detecting the analog Trojan.**

I. HARDWARE ARCHITECTURE

A. The ARM-compatible Processor

We propose a fused microarchitecture based on the ARMv7-A&R ISA. This ARM processor is named Merlin. Using michroarchitectural techniques, Merlin expands the DSP capabilities of the ARM processor. Merlin supports most traditional ARM instructions, but does not support some ISA extensions, such as Thumb, ThumbEE, Jazelle, Floating-point, and Advanced SIMD. We realize 181 ARM instructions in total, which is enough to run common benchmarks, such as DhryStone, CoreMark, DSPStone, and EEMBC telecom. There are 7 execution modes defined in ARMv7-A&R ISA, while Merlin works only under user mode. Merlin adopts a fused microarchitecture integrating in-order superscalar and VLIW. Normally, Merlin works under dual-issue in-order superscalar mode. It can be switched to 6-issue VLIW mode when the task is compute intensive. Mode switch can be performed through software. Merlin can be used as an MCU, or a DSP under different application scenarios. Merlin has 16 KB of on-chip L1 instruction Cache, with a 256-bit wide port, and 32 KB dual-port data memory, with each port being 64-bits wide. Merlin has 6 functional units, consisting of 2 arithmetic units (A), 2 multiply units (M), and 2 load/store units (D).

An SoC is designed, where Merlin is used as an MCU. The chip diagram is shown in Figure 1. On this chip, Merlin is integrated with DMA, ROM, SRAM, four 128KB embedded ReRAM, and a variety of peripheral I/O. The Dhrystone performance of Merlin is 1.9 DMIPS/MHz, which is comparable to ARM Cortex-A8 processors.

B. A2-like Analog Trojan in Merlin

A2 is a small analog circuit, which can be inserted into an already placed and routed design. It reads a digital pulse signal (the trigger input), and triggers the payload when the pulse signal has toggled with a high frequently for a certain period of time. The trigger input is connected to a signal that can be toggled with high frequency through a special code snipper running on the processor. The attacker insures that the trigger signal has a much lower toggle rate during typical workloads. This makes detecting the hardware Trojan difficult through testing, not to mention that there exists many low toggling frequency bits in modern processors.

When inserting the analog Trojan trigger circuit into the Merlin processor, we should first select a viable trigger input. The trigger input should have low toggling rate in common cases. It should be controllable through software, so that the trigger code can make it toggle at high frequency to launch the attack.

CPSR (Current Program Status Register) is a software reachable register. The definition of CPSR is shown in Figure 2. We select the CPSR_J bit as the trigger input. Since Merlin does not support ISA extensions, this bit has no function. We use R0 as the attack payload. Once the attack is triggered, the value store in R0 will be modified. We generate the trigger input by frequently writing 0 and 1 to CPSR_J alternatively. When the Trojan is triggered, it changes the value stored in R0 from 0 to 1 so that we can observe the change through a register read. We also attach the trigger output signal

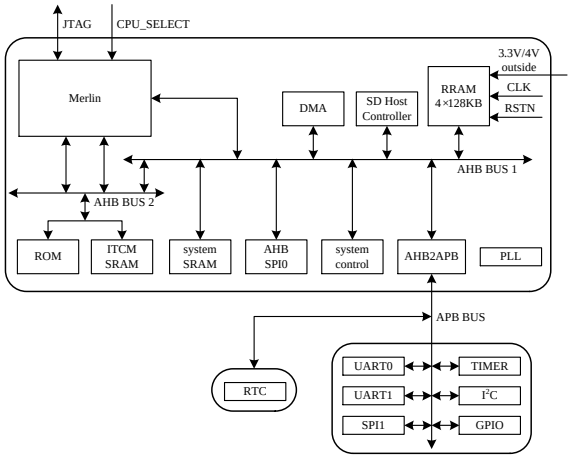

Figure 1: The SoC chip diagram