DVFS covert-channels in Zynq Ultrascale+ SoC-FPGAs

CryptArchi

30th May 2022, Porquerolles France.

Lilian BOSSUET and Carlos LARA

cnrs
UNIVERSITÉ DE LYON
UNIVERSITÉ JEAN MONNET SAINT-ÉTIENNE
LABORATOIRE HUBERT CURIEN
UMR • CNRS • 5516 • SAINT-ETIENNE
PROJET FINANCÉ PAR L'ANR - PROJECT FUNDED BY THE ANR
ANR
ARCHI-SEC
CryptArchi

In this talk:

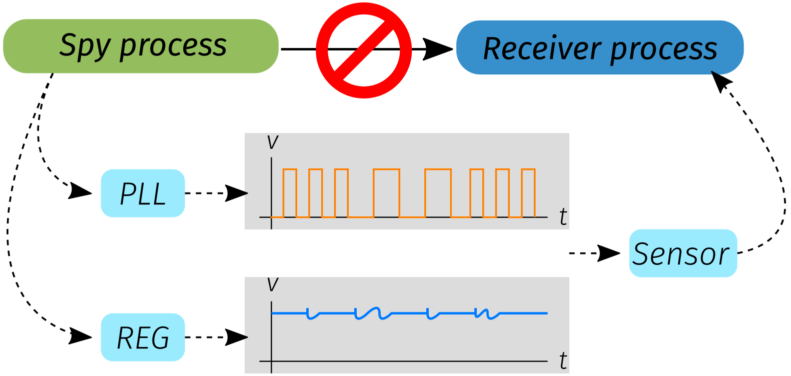
- Covert channels
- The Zynq Ultrascale+ SoC-FPGA
- Frequency and Voltage modulation strategies
- Covert channels between the RPU and the PL
- Covert channels between the APU and the PL
- Covert channels between the APU and the RPU

In this talk:

- Covert channels
- The Zynq Ultrascale+ SoC-FPGA
- Frequency and Voltage modulation strategies
- Covert channels between the RPU and the PL
- Covert channels between the APU and the PL
- Covert channels between the APU and the RPU

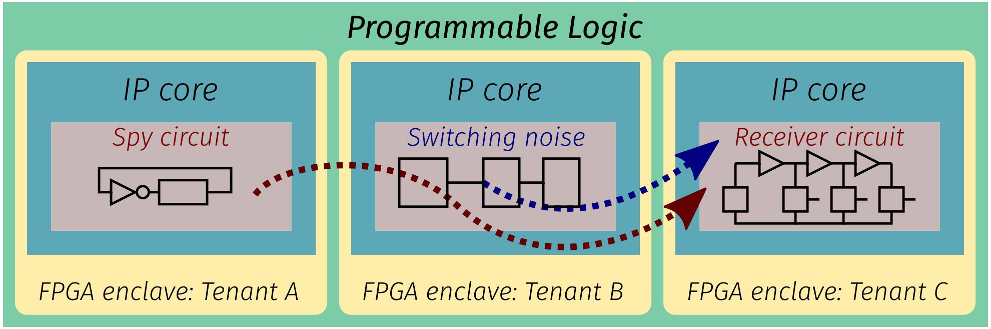
Covert channels

"A Covert Channels has the goal of enabling the flow of information between two or more parties which are barred from communicating by an overseer party."



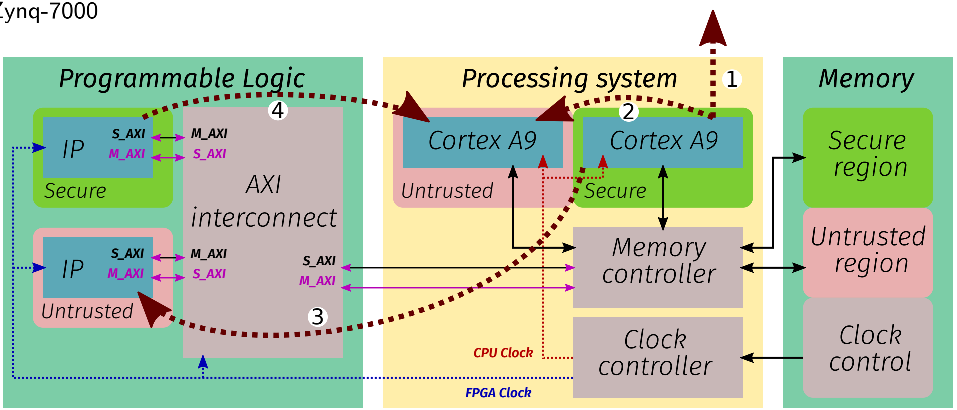
Covert channels - State of the Art

- D. R. E. Gnad, C. D. K. Nguyen, S. H. Gillani, M. B. Tahoori, Voltage-based Covert Channels using FPGAs, in: *Cryptology ePrint Archive*, Report 2019/1394, `https://ia.cr/2019/1394` (2019).
- Zynq-7000



Covert channels - State of the Art

- E. M. Benhani, L. Bossuet, DVFS as a Security Failure of TrustZone-enabled Heterogeneous SoC, in: *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2018, pp. 489–492. `doi:10.1109/ICECS.2018.8618038`.
- Zynq-7000

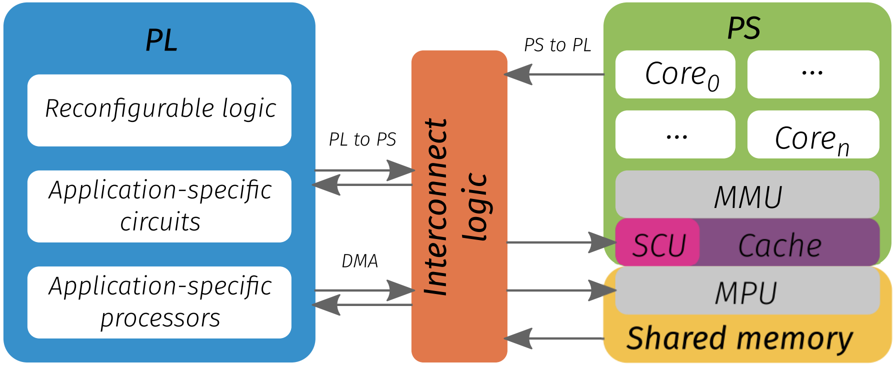


In this talk:

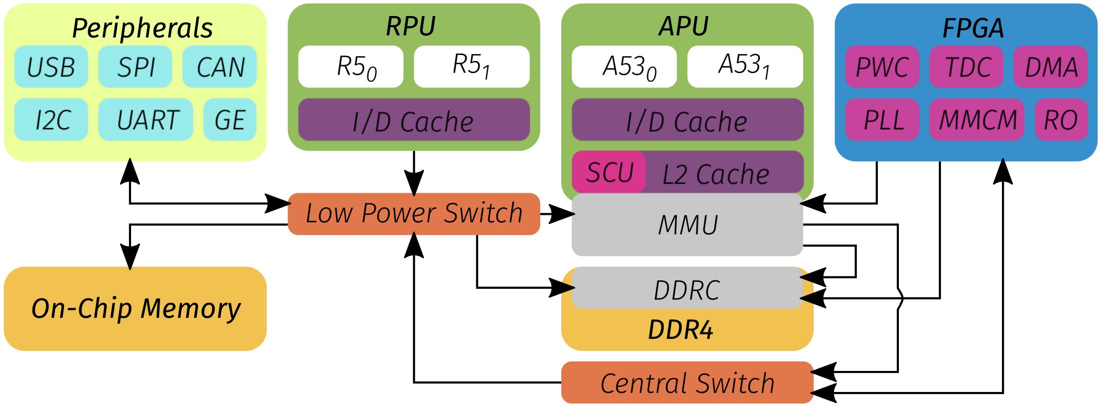
- Covert channels
- The Zynq Ultrascale+ SoC-FPGA
- Frequency and Voltage modulation strategies
- Covert channels between the RPU and the PL
- Covert channels between the APU and the PL
- Covert channels between the APU and the RPU

General architecture of SoCs

"A System on a Chip (SoC) is an heterogeneous platform, constituted by the creation of processors and hardware accelerators in the same die."



The Zynq Ultrascale+ SoC-FPGA

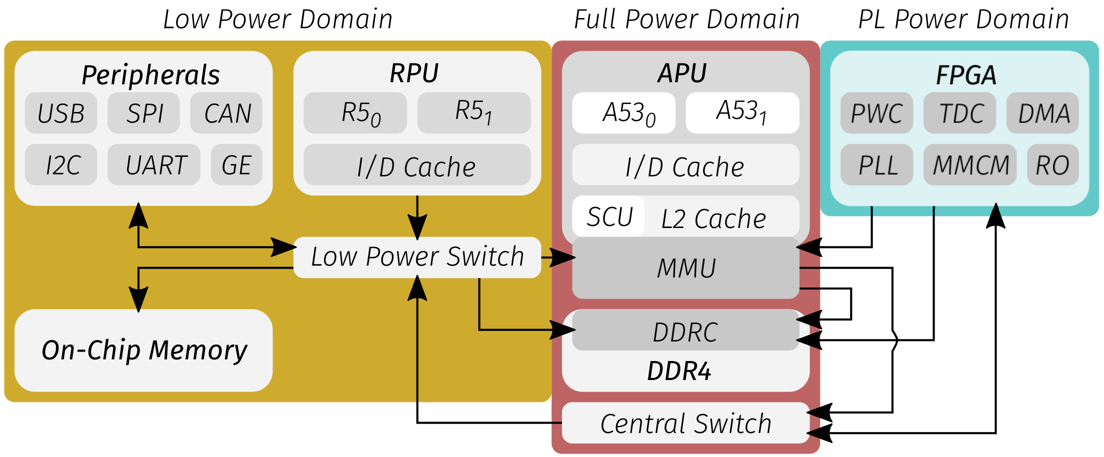


* Arrows indicate who acts as the master of the transaction.

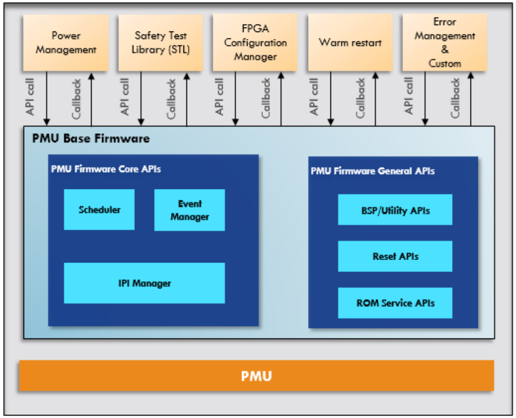
In this talk:

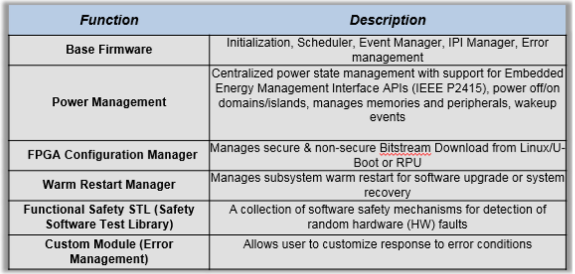
- Covert channels
- The Zynq Ultrascale+ SoC-FPGA
- Frequency and Voltage modulation strategies
- Covert channels between the RPU and the PL
- Covert channels between the APU and the PL
- Covert channels between the APU and the RPU

The Zynq Ultrascale+ Power Domains



Platform Management Overview

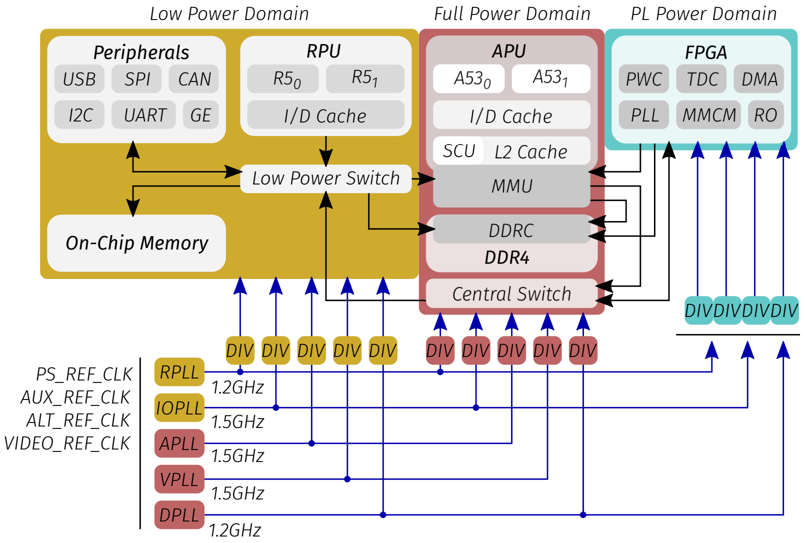




Function	*Description*
Base Firmware	Initialization, Scheduler, Event Manager, IPI Manager, Error management
Power Management	Centralized power state management with support for Embedded Energy Management Interface APIs (IEEE P2415), power off/on domains/islands, manages memories and peripherals, wakeup events
FPGA Configuration Manager	Manages secure & non-secure Bitstream Download from Linux/U-Boot or RPU
Warm Restart Manager	Manages subsystem warm restart for software upgrade or system recovery
Functional Safety STL (Safety Software Test Library)	A collection of software safety mechanisms for detection of random hardware (HW) faults
Custom Module (Error Management)	Allows user to customize response to error conditions

- **Provide centralized access to system resources**
- **Modular PMU SW architecture**
  - Build & Initialize only what is needed
- **Provide resources to be used by all modules**
  - PMU Firmware Core APIs, PMU Firmware General APIs

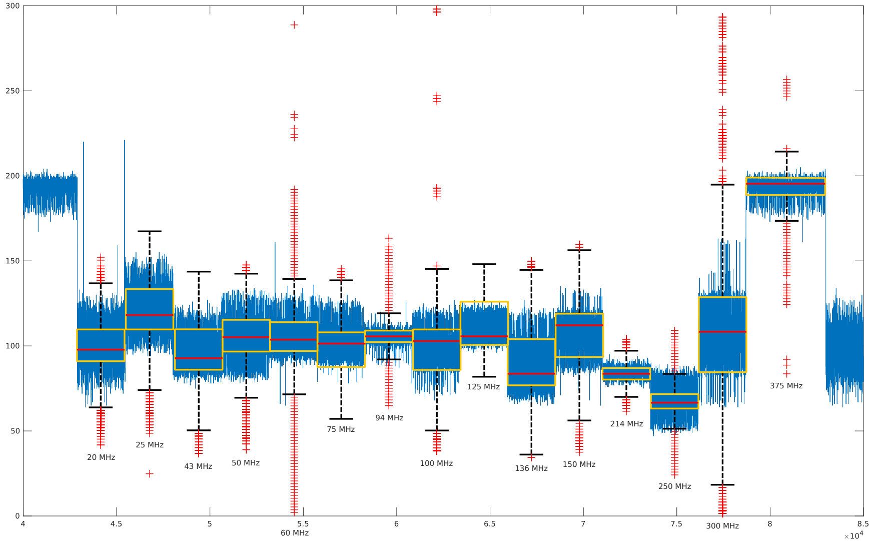
The Zynq Ultrascale+ Clock Tree



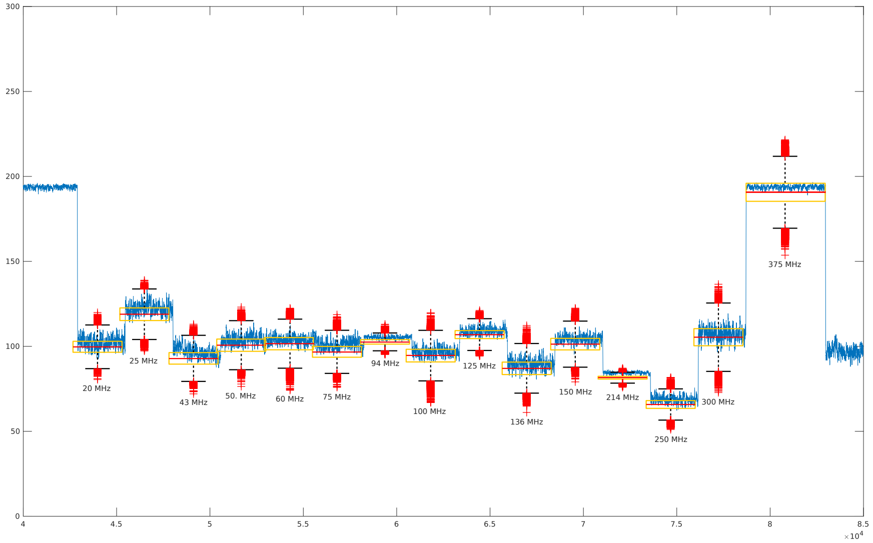
To modify the divider registers

- Bare metal: Simply perform read/write operations over the address.
- Linux:
  - Employ the *busybox* utility to *mmap* `/dev/mem`
  - Employ the userspace drivers: `cpufreq`, `fclk0`, `fclk1`, `fclk2`, `fclk3`

To characterize the channel (RO counts retrieved from the PS)



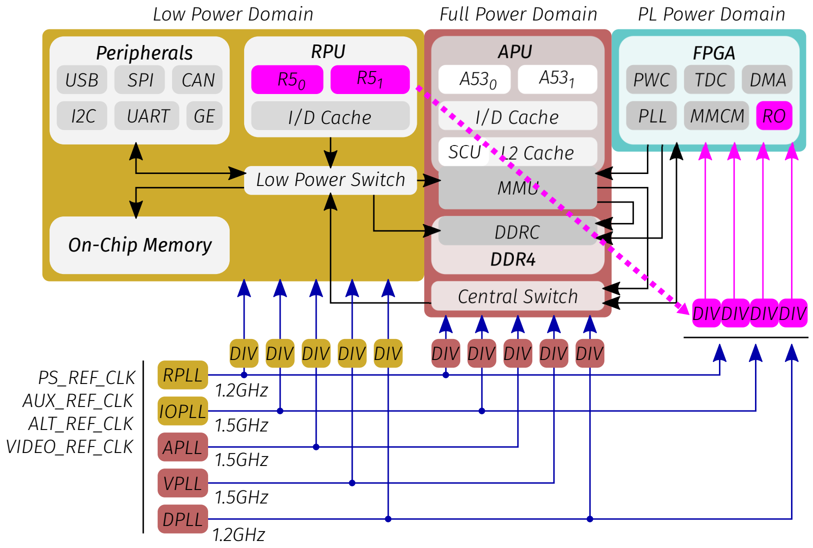
To characterize the channel: moving average $n = 4$



In this talk:

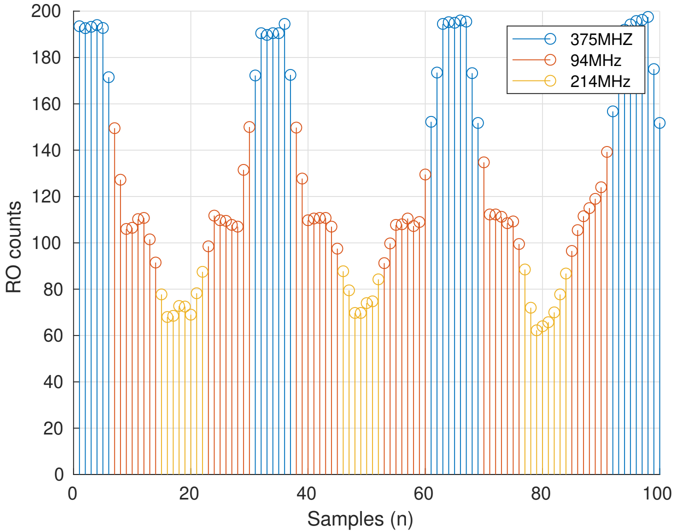
- Covert channels
- The Zynq Ultrascale+ SoC-FPGA
- Frequency and Voltage modulation strategies
- Covert channels between the RPU and the PL
- Covert channels between the APU and the PL
- Covert channels between the APU and the RPU

Attack #1: RPU to PL

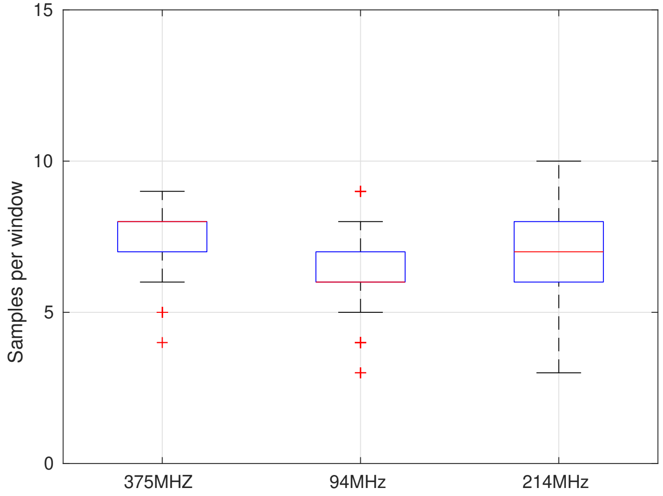


Experiments: $f_1 = 375\ MHz, f_2 = 214\ MHz, f_3 = 94\ MHz$

```
for byte in message do
    for bit in byte do
        if bit then
            fclk ← $f_1$
        else
            fclk ← $f_2$
        end if
        fclk ← $f_3$
    end for
end for
```



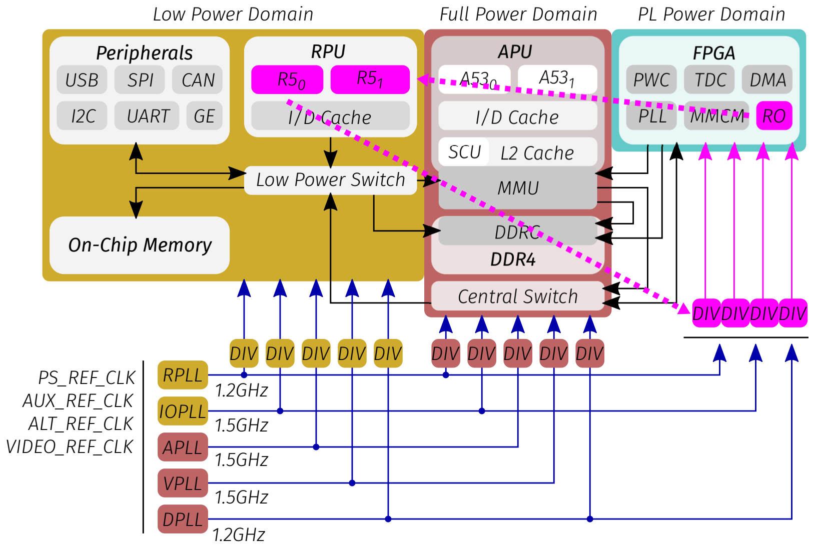
Experiments: zero errors over 12 KB



Results

- Sender: Bare metal on Cortex-R5F@533 MHz
- Transmission delay (RPU to PL, per bit): 6.82 us
- Transmission rate: 17.899 KBps
- Receiver: PL, 16 ROs, 8-bit output
- Sampling rate (PL): 333 MSps
- Transmission delay (PL to RPU, per sample): 620 ns
- Sampling rate (RPU): 1.613 MSps $\rightarrow$ RPU to RPU: 17.899 KBps

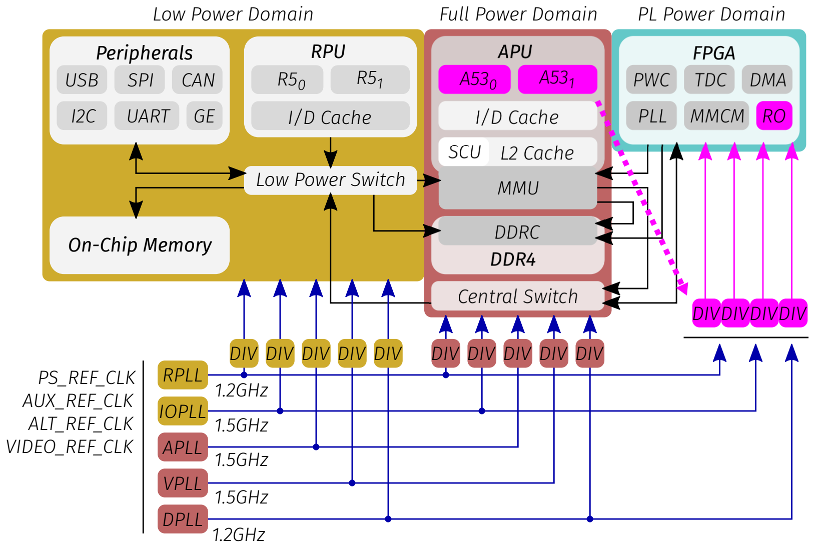
Attack #2: RPU to RPU



In this talk:

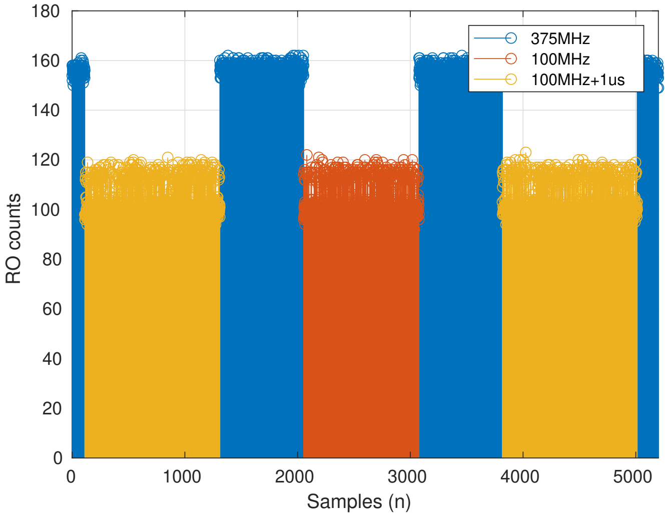
- Covert channels
- The Zynq Ultrascale+ SoC-FPGA
- Frequency and Voltage modulation strategies
- Covert channels between the RPU and the PL
- Covert channels between the APU and the PL
- Covert channels between the APU and the RPU

Attack #3: APU to PL

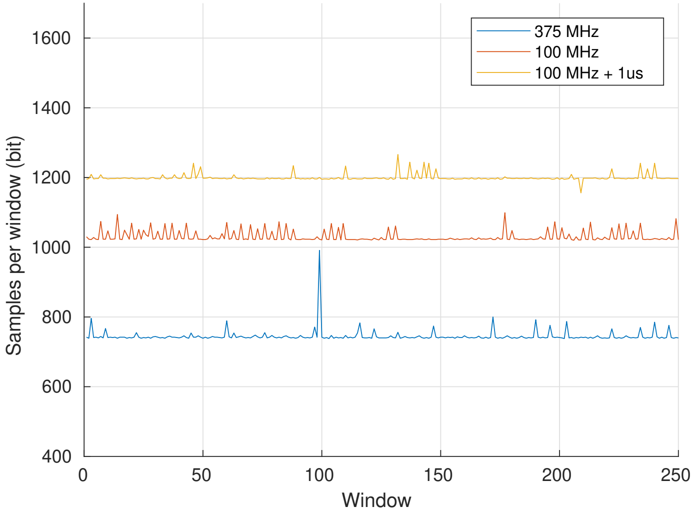


Experiments: $f_1 = 100\ MHz, f_2 = 375\ MHz$

```
for byte in message do
    for bit in byte do
        if bit then
            fclk ← $f_1$
            usleep(1)
        else
            fclk ← $f_1$
        end if
        fclk ← $f_2$
    end for
end for
```



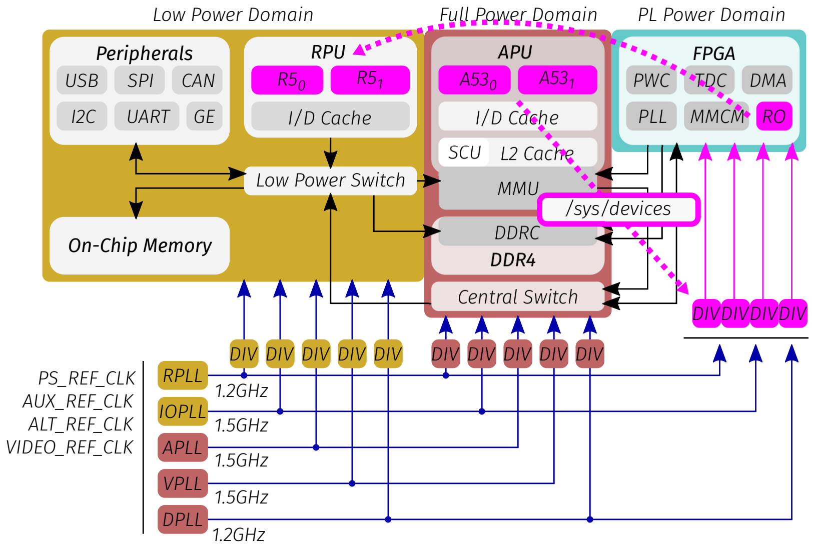
Experiments: zero errors over 64 B



Results

- Sender: Linux on Cortex-A53@1.3 GHz
- Transmission delay (APU to PL, per bit): 676 μs (627 μs, 676 μs)
- Transmission rate: 185 Bps
- Receiver: PL, 16 ROs, 8-bit output
- Sampling rate (PL): 333 MSps
- Transmission delay (PL to RPU, per window): 620 ns
- Sampling rate (PS): 1.613 MSps $\rightarrow$ APU to RPU: 185 Bps

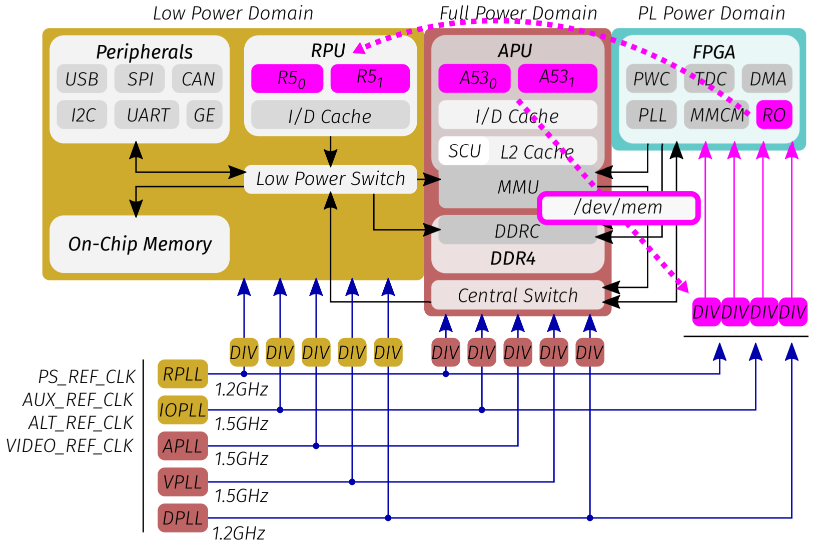
Attack #4: APU to RPU, using the API



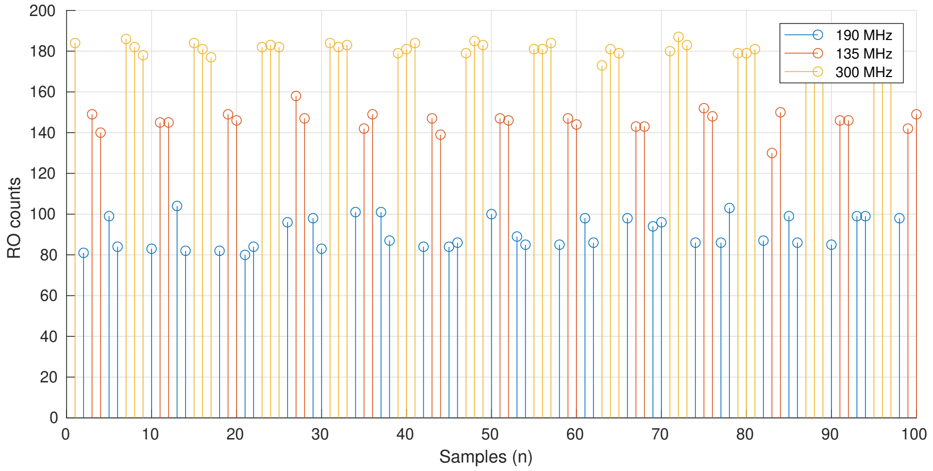
In this talk:

- Covert channels
- The Zynq Ultrascale+ SoC-FPGA
- Frequency and Voltage modulation strategies
- Covert channels between the RPU and the PL
- Covert channels between the APU and the PL
- Covert channels between the APU and the RPU

Attack #4: APU to RPU, using /dev/mem



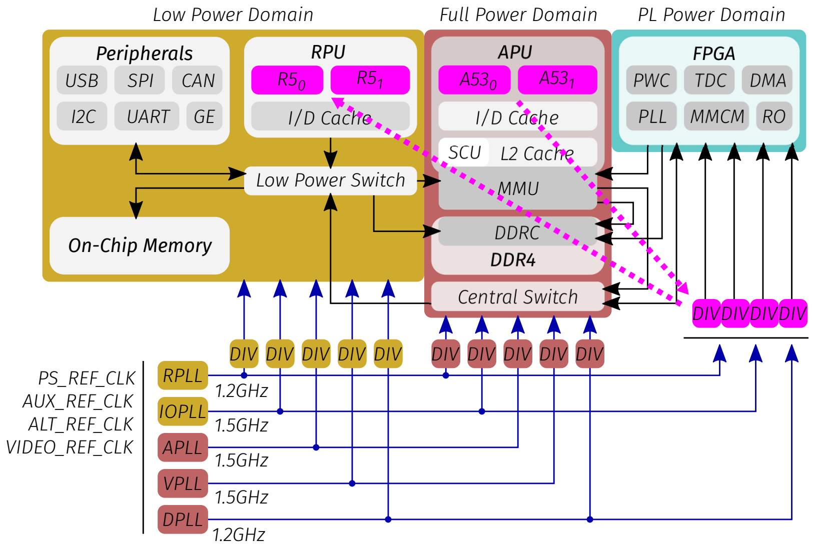
Experiments: $f_1 = 190\ MHz, f_2 = 135\ MHz, f_3 = 300\ MHz$



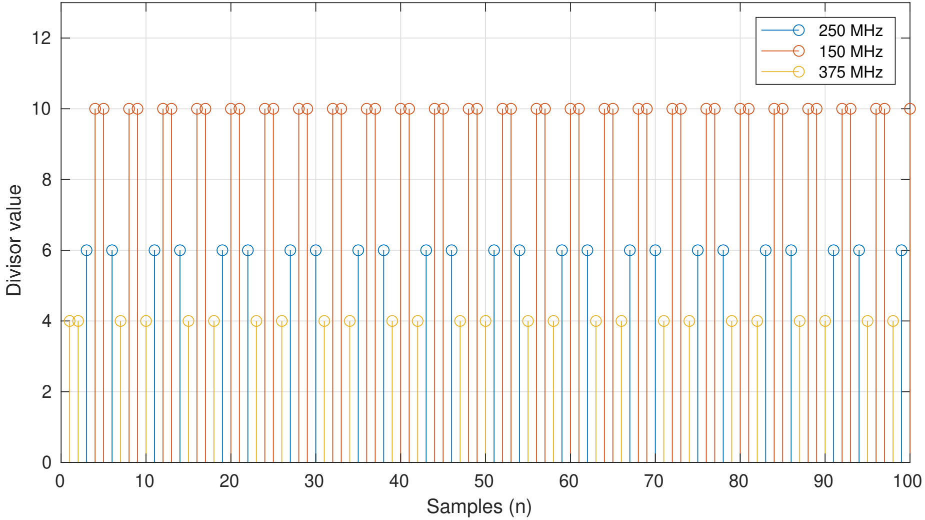
Results

- Sender: Linux on Cortex-A53@1.3 GHz
- Transmission delay (APU to REG, per bit): $< 1\ \mu$s
- Transmission rate: $\sim$122 KBps
- Receiver: Bare metal on Cortex-R5F@533 MHz
- Sampling rate (PL): 620 ns
- Sampling rate (RPU): $\sim$1.613 MSps $\rightarrow$ APU to RPU: 122 KBps

Attack #4: APU to RPU, register-register



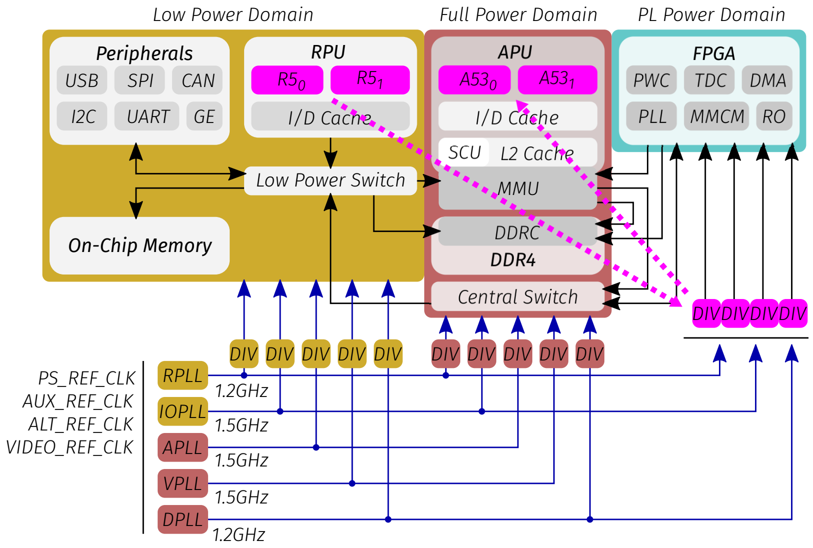
Experiments: $f_1 = 375\ MHz, f_2 = 250\ MHz, f_3 = 150\ MHz$



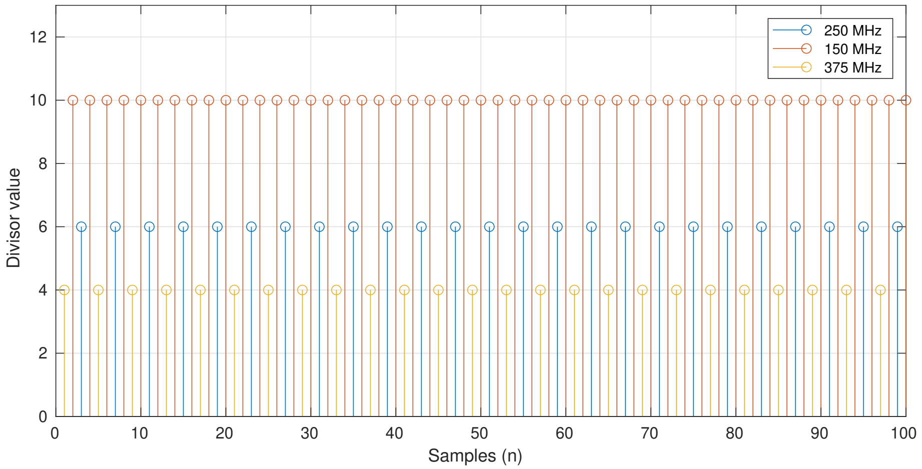
Results

- Sender: Linux on Cortex-A53@1.3 GHz
- Transmission delay (APU to REG, per bit): $< 1\ \mu$s
- Transmission rate: $\sim$122 KBps
- Receiver: Bare metal on Cortex-R5F@533 MHz
- Sampling rate (REG): 497 ns
- Sampling rate (RPU): $\sim$2 MSps $\rightarrow$ APU to RPU: 122 KBps

Attack #5: RPU to APU, register-register



Experiments: $f_1 = 375\ MHz, f_2 = 250\ MHz, f_3 = 150\ MHz$



Results

- Sender: Bare metal on Cortex-R5F@533 MHz
- Transmission delay (RPU to REG, per bit): 457 ns $\times$ 2 $=$ 914 ns
- Transmission rate: 133.556 KBps
- Receiver: Linux on Cortex-A53@1.3 GHz
- Sampling rate (REG): $\sim$243 ns
- Sampling rate (APU): $>$4 MSps $\rightarrow$ RPU to RPU: $\sim$133 KBps

In this talk:

- Covert channels
- The Zynq Ultrascale+ SoC-FPGA
- Frequency and Voltage modulation strategies
- Covert channels between the RPU and the PL
- Covert channels between the APU and the PL
- Covert channels between the APU and the RPU
- Final remarks

To modify the divider registers

- Bare metal: Read/write operations over the address.
  - Advantages: High efficiency, unlimited potential
  - Disadvantages: High scrutiny and auditing, requires precise knowledge of the architecture
- Linux:
  - Employ the *busybox* utility to *mmap* `/dev/mem`
    - Similar to a bare metal scenario.
  - Employ the userspace drivers: `cpufreq`, `fclk0`, `fclk1`, `fclk2`, `fclk3`
    - Advantages:
      - Can affect a wide range of architectures
      - Does not require precise knowledge of the system
      - Difficult to detect
    - Disadvantages: Low efficiency

Acknowledgments

The authors acknowledge the support of the French Agence Nationale de la Recherche (ANR), under grant ANR-19-CE39-0008 (project ARCHI-SEC).

ARCHI-SEC

https://archi-sec.telecom-paristech.fr/

UNIVERSITÉ DE LYON

UNIVERSITÉ JEAN MONNET SAINT-ÉTIENNE

Laboratoire Hubert Curien

Lilian BOSSUET

Carlos Andres LARA-NINO