Shaheen: An Open, Secure, and Scalable RV64 SoC for Autonomous Nano-UAVs

University of Bologna


luca.valente@unibo.it

PULP Platform
Open Source Hardware, the way it should be!
Autonomous Nano-UAVs

• **Versatility, safety, and cost-effet:**
  - small and agile
  - ideal for accessing hard-to-reach areas or tight spaces (inspection/maintenance)
  - relatively inexpensive to produce and operate

• **Requirements for future generation of nano-UAVs:**
  - Run increasingly complex multi-tasking workloads with large memory footprint
  - Within a few hundred mW power budget
  - Support for virtualization and secure operations in uncontrolled/hostile scenarios
Shaheen: an Open, Secure, and Scalable RV64 SoC for Autonomous Nano-UAVs

- 9mm² SoC in 22nm FDSOI technology with:
  - A RV64 Linux-capable CPU enhanced with
    - Hypervisor support
    - Timing-channels mitigation
  - An energy efficient programmable multi-core accelerator (PMCA) based on 8 RV32 cores with ML and DSP extensions
  - Up to 512MB of low-power off-chip main memory
  - Logic locking on key IPs within the architecture
  - 200mW power envelope
Let’s dive in!
RV64 and custom RV32: the best of both worlds

• Different cores serve different parts of the target application

• Host:
  • On top of the hypervisor:
    • Attitude control (RTOS-based)
    • Linux-based legacy software such as wireless network stack.

• PMCA:
  • The PMCA runs the CNN-based pose estimation task [1] fed by a low-resolution front-looking camera.

Shaheen’s heterogeneous HW-SW stack

**HOST**

- L2 SPM
- Mem
- AXI interconnect
- HyperRAM
- Mem ctrl
- HyperRAMs
- I/O
- SENSORS

**Programmable MultiCore Accelerator**

- L1 SPM
- Mem
- Interconnect
- DMA
- RV 32
- I$
- RV 32
Timing-channel mitigation

- The 64-bit core implements the temporal fence instruction “fence.t”[2]:
  - capability of clearing vulnerable microarchitectural states
  - enables a history-independent context-switch latency
  - low implementation effort (<1%)
  - low performance impact
  - negligible hardware costs

Timing-channel mitigation: prime and probe attacks

- **Prime and probe attacks:**
  - The spy brings the target HW into a known state (*prime*)
  - The OS switches to an applications containing a Trojan, accessing a subset of the HW resources to encode a secret
  - The execution switches back to the spy, which *probes* the execution time, correlated with the encoded secret.

Execution time depending on the encoded secret, without and with fence.t [2]
• Overview:
  • **1MB+256kB** of scratchpad memory
  • **200mW** (120mW Host domain+ 80mW PMCA)
  • RV-32 cluster’s cores aggressively optimized for FP-DSP and integer QNN inference [3]
  • The cluster can deliver up to:
    • **7.9GFLOp/s** on **16-bit** FP data
    • up to **90GOp/s** on 2-bit integer data @1.2TOPs/s/W (high-throughput mode)
    • up to **50GOp/s** on 2-bit integer data @**1.8TOPs/s/W** (energy-efficient mode)

Physical implementation details: logic locking

- **Logic Locking:**
  - Consists in modifying a hardware IP to add a new input (“logic locking key”) to be applied to unlock the original IP functionality. Without the proper logic locking key, the chip is non-functional [4].
  - Between the interconnect and the memory controller
  - Between the interconnect and the PMCA

---

Advancing the SoA

• Overview:
  • Match best in class (AI-IoT) SW performance
  • Only SoC for autonomous UAVs (within 200mW) with Hypervisor+Linux support
  • Advanced security features

<table>
<thead>
<tr>
<th>Target board</th>
<th>Technology</th>
<th>Die Size</th>
<th>CPU</th>
<th>Supported OS</th>
<th>Host-compute FP support</th>
<th>Security Features</th>
<th>Peak SW Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>PiXhawk</td>
<td>40nm</td>
<td>-</td>
<td>Cortex M7</td>
<td>RTOS</td>
<td>SP-FPU, DP-FPU</td>
<td>Crypto/hash accelerators</td>
<td>240MFLOp/s(FP32)</td>
</tr>
<tr>
<td>Crazyflie</td>
<td>90nm</td>
<td>-</td>
<td>Cortex M4</td>
<td>RTOS</td>
<td>-</td>
<td>-</td>
<td>72MFLOp/s(8b)</td>
</tr>
<tr>
<td>AIDeck</td>
<td>55nm</td>
<td>10mm2</td>
<td>9x RISCY</td>
<td>RTOS</td>
<td>SP-FPU</td>
<td>-</td>
<td>6 MFLOp/s(8b)</td>
</tr>
<tr>
<td>AIDeck</td>
<td>22nm FDSOI</td>
<td>12mm2</td>
<td>10x RISCY-NN</td>
<td>RTOS</td>
<td>SP-FPU</td>
<td>-</td>
<td>7GFLOp/s(FP16)</td>
</tr>
<tr>
<td>AIDeck</td>
<td>22nm FDSOI</td>
<td>9mm2</td>
<td>9x RISCY-XNN</td>
<td>RTOS</td>
<td>SP-FPU</td>
<td>-</td>
<td>15,6GFLOp/s(8b)</td>
</tr>
<tr>
<td>AIDeck / PiXhawk</td>
<td>22nm FDSOI</td>
<td>9mm2</td>
<td></td>
<td></td>
<td></td>
<td>Side-channel protection, Logic Locking, IOPMP</td>
<td>3,12GFLOPs(FP32)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>85GFLOp/s(2b)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7.9GFLOp/s(FP16)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>90 GFLOp/s(2b)</td>
</tr>
</tbody>
</table>
