Report 03.12.20

03/12/2020

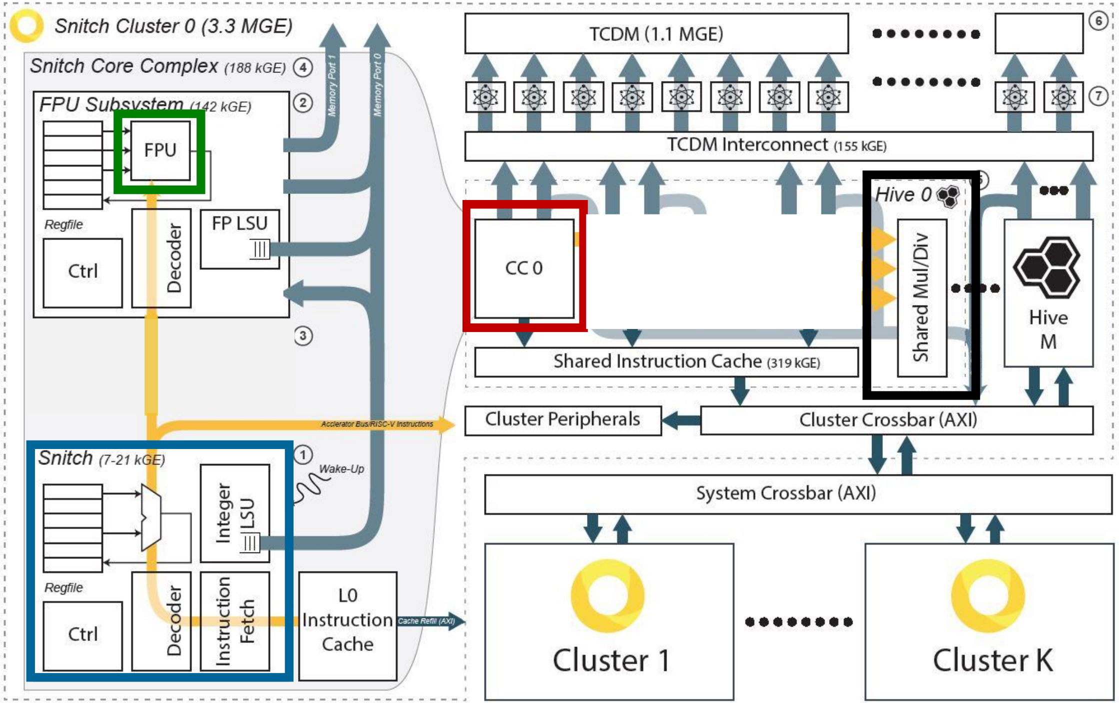
Matteo Perotti
Luca Bertaccini
Pasquale Davide Schiavone
Stefan Mach

Professor Luca Benini
Integrated Systems Laboratory
ETH Zürich

Summary

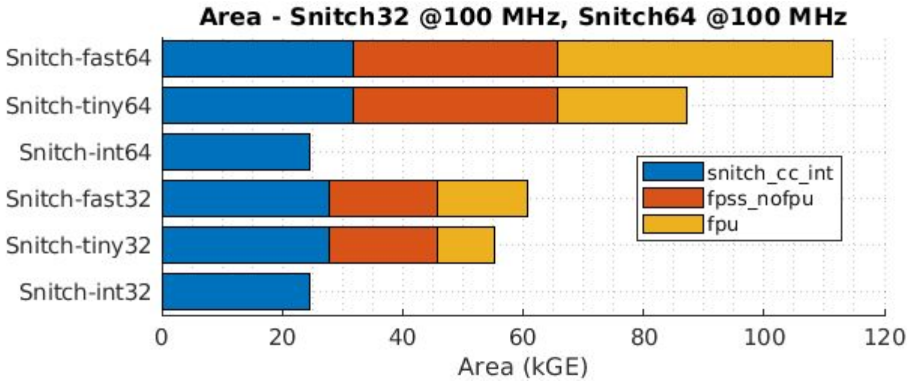
- libgcc performance:
    - __addsf3
    - __mulsf3
    - __adddf3
    - __muldf3
- Zfinx implementation (Snitch+FP32_TinyFPU)

Snitch - MulDiv

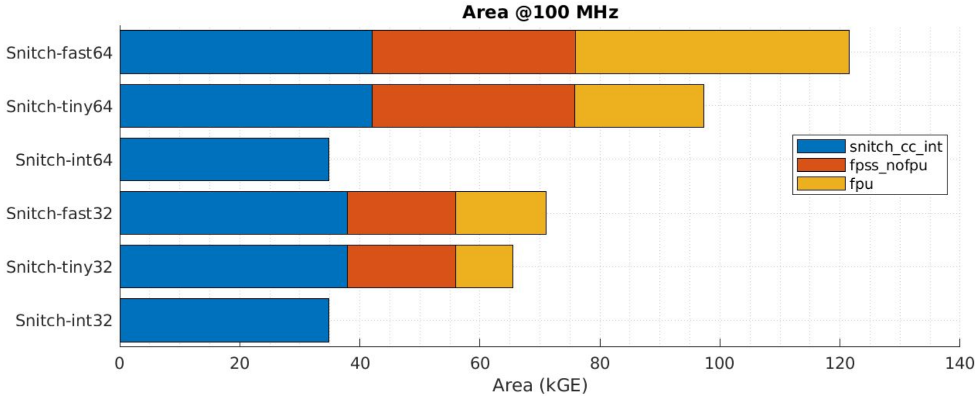


Results including MulDiv

1. Results Snitch_cc



2. Results Snitch_cc + MulDiv



Results including MulDiv

- Same performance

- Area (muldiv unit occupies around 50% of the integer core area)
  - Snitch-tiny is **2.8x** (DP) and **1.9x** (SP) larger than Snitch-int
  - Without muldiv, **3.5x** (DP) and **2.25x** (SP)

  - Zfinx Snitch-tiny32 is **1.4x** larger than Snitch-int
  - Without muldiv, **1.7x** larger than Snitch-int

  - Snitch-tiny **20%** (DP) and **7.8x** (SP) smaller than Snitch-fast
  - Without muldiv, **22%** (DP) and **9%** (SP)

- Power
  - Snitch-tiny consumes up to **50%** (DP) and **37%** (SP) less than Snitch-fast (instead of **47%** (DP) and **30%** (SP))

libgcc - Performance

- 10000 tests with random inputs in the interval [-10000, 10000]

	libgcc (Snitch)	libgcc (Snitch - larger cache)	libgcc (cv32e40p)*
__addsf3	**253**	105	80
__mulsf3	**193**	126	120
__adddf3	**284**	150	101
__muldf3	**454**	238	219

- **Area overhead** for larger cache higher than FP64 TinyPFU

*different input values - interval (0,1)

Results new cache configuration

- **Performance**:
  - Maximum speed-up is **9.5x** (DP and SP)
  - Instead of **18.5x** (DP) and **15.5x** (SP)

- **Area**
  - Snitch-cc not affected
  - Overhead higher than a FP64 TinyFPU

- **Power** increases for Snitch-int

- **Energy efficiency** improvement (Snitch-tiny vs. Snitch-int):
  - **6.2x** (DP) and **7.4x** (SP)
  - Instead of **8x** (DP) and **9.9x** (SP)

Zfinx implementation

- Added **one read port** to the INT register file (fmadd)

- Created new FP_SS:
  - without **LSU**
  - without **FP register file**

- **Accelerator interface** used as for the shared MULDIV unit

Zfinx implementation

- Using **PULP compiler**, I ran a **simple program** composed by:
  - 2 fmadd.s, 1 fadd.s, 1 fmul.s
- Very **simple program** to prevent the compiler from using non-supported PULP instructions
- **Further tests** with **Zfinx compiler** to validate the design

Next Steps

- **Complete Zfinx** implementation of **Snitch + 32-bit TinyFPU**
- Test using **Zfinx compiler**
- **RISC-V Summit**

RISC-V Summit

- **Presentation**: Tuesday, 8 December 2020 11:00am - 11:20am PST (Pacific Standard Time, GMT-8)

https://tmt.knect365.com/risc-v-summit/agenda/1/#system-architectures_a-tiny-risc-v-floating-point-unit_11-00am

- **Presentation**: 8:00pm - 8:20pm (Zurich Time)
- **Live Q&A Forum with Speakers Room A**: 9:50pm - 10:30pm (Zurich Time)

DAC paper

- **Paper submitted**
- Work **presented** to the RISC-V Code Size reduction sub-committee
- Further improvements -> student projects
- **Clean** the **repository** for the release,
- Evaluate a more liberal license

Library + Code Size reduction instructions

- **Memory** operations are **not used** -> no benefits from optimizing them
- **clz**, branch against immediate, shift+logic, c.not, c.neg
- 4 Byte **LUI** is usually **enough** (sign mask, implicit 1, inf/qNaN patterns)