Centrifuge: Evaluating full-system HLS-generated heterogeneous-accelerator SoCs using FPGA-Acceleration

Qijing Jenny Huang, Christopher Yarp, Sagar Karandikar, Nathan Pemberton, Benjamin Brock, Liang Ma, Guohao Dai, Robert Quitt, Krste Asanović, John Wawrzynek

jennyhuang@nvidia.com
Motivation

- What is a good methodology for designing SoCs with many accelerators?

- High end-to-end performance
- Low NRE costs
- Short time-to-market
1. Workload Characterization
2. Accelerator Modeling
   - Analytical
   - Transaction-based
3. RTL Development
4. Emulation and Verification
5. Chip Tapeout
6. Software Development

- Proposal 1: Native Simulation
  Run full-stack software with target SoC simulation

- Proposal 2: Rapid Prototyping
  Combine 2 and 3 with one high-level abstraction

- Proposal 3: Agile Development
  Improve software stack concurrently with hardware development
Background

Projects incorporating AI or RISC-V processors in design in 2020

* Figure From Wilson Research Group
Centrifuge Design Flow

Centrifuge is a unified accelerator design flow that generates the interfaces and design automation scripts that leverages existing tools:

- **Injects:**
  - High-level Algorithmic Description in C/C++
  - Accelerator Configuration
- **Produces:**
  - Full Functional Accelerator SoC
  - Corresponding Software Stack
  - Tooling Scripts
Centrifuge Tool Flow

Step 1: Hotspot Detection on FireSim

Step 2: SoC Generation with HLS

Step 3: SW Primitives Generation

Step 4: End-to-end Evaluation
Step 1: Hotspot Detection on FireSim

### Inputs

**Input Programs**
- func1
- func2
- func3

### Base SoC

### Transformed SoC

### Profiled Programs
- func1 (90 %)
- func2 (8 %)
- func3 (1 %)

...
Step 2: SoC Generation with HLS

User Defines

**HLS C Func Pragma**

```c
void func1 (){
  for(i=0;i<n;i++)
    #pragma pipeline II=N
  for(j=0;j<m;j++)
    #pragma unroll M }
```

**SoC Definition**

"RoCC":{
  "custom0":{
    "pgm": "pgm0",
    "func": "func1"},
  "TLL2":[
    {"pgm":"vadd_tl",
     "func": "vadd",
     "addr":"0x20000"},
    {"pgm":"aes_tl",
     "func": "encrypt",
     "addr":"0x30000"}]
}

Auto-generated

**HW Interface**

1. RoCC
2. TileLink
3. Network-Attached

**Accel SoC**

Generated RTL
Step 3: SW Primitives Generation

User Defines

Runtime Env

```
"name": "pgm0",
"base": "linux.json",
"workdir": "/",
"files": ["pgm0"],
"command": "/pgm0"
```

SoC Definition

```
"RoCC": {
    "custom0": {
        "pgm": "pgm0",
        "func": "func1"},
    "TLL2": {
        "pgm": "vadd_tl",
        "func": "vadd",
        "addr": "0x20000"},
        "pgm": "aes_tl",
        "func": "encrypt",
        "addr": "0x30000"},
```

Auto-generated

SW Primitives

- RoCC Insn
- MMIO Wrapper
- Linux Wrapper

SW Compiler

- RISC-V LLVM/GCC

Accel Programs

- func1_accel
- func2
- func3
- ...

func1_accel
func2
func3
...
Step 4: End-to-end Evaluation

Generated

**Accel Programs**

- `func1_accel`
- `func2`
- `func3`
  ...

**Accel SoC**

**Transformed SoC**

**Accelerated Programs**

- `func1_accel` (8%)
- `func2` (91%)
- `func3` (1%)
  ...

Repeat from Step 1
Centrifuge Design Flow

Centrifuge offers the user:
1. Full-system evaluation
2. Fast development and verification cycle for both HW/SW
3. Large design space
   a. Hardware Integration
   b. Architectural Design Variation
   c. Software Integration
In Summary

- We present a methodology and flow, Centrifuge, that can rapidly generate and evaluate heterogeneous SoCs by combining an HLS toolchain with the open-source FireSim FPGA-accelerated simulation platform.

Thanks!
Access to code: https://github.com/hqjenny/centrifuge
Email: jennyhuang@nvidia.com