BeiHang Short Course, Part 6: CoRAM FPGA “Architecture”

James C. Hoe
Department of ECE
Carnegie Mellon University

Collaborators: Eric S. Chung (MSR), Gabriel Weisz, Michael Papamichael, and Ken Mai

Moore’s Law for FPGA

10TFLOPS


Stratix 10
Why doesn’t everybody want to compute with FPGAs?

“Traditionally, FPGAs have been the bastard step-brother of ASICs...”

Proceedings of ISFPGA 2004

• FPGAs today NOT designed for computing
  – tools and languages difficult to use
  – users exposed to device and platform details
  – no application portability

Outline

• Motivations

• CoRAM Architecture

• Current work and evaluations

• Conclusions
Review of FPGA Anatomy

Programmable lookup tables (LUT) and flip-flops (FF) aka “soft logic” or “fabric”

Computing on FPGA Today

User responsible for:
- application and data
Computing on FPGA Today

User responsible for:
- application and data
- platform I/O and memory interfaces
- data distribution
- memory control logic

The CoRAM FPGA
The CoRAM FPGA

- Hard logic for data distribution (NoC)
- "Software"-managed memory hierarchy

CoRAM FPGA Architectural View

- Hard logic for data distribution (NoC)
- "Software"-managed memory hierarchy
- Simple, portable abstraction for user
FPGA as First-Class, Equal Partner

Simple Example

ctrlthread() {
    cohandle c0 = get_sram("c0");
    c_coram_write(c0, sram_address, global_address, size);
    c_fifo_write(1);
}

module verilog_top(...) {
    coram c0 (/*ports*/);
    cofifo f0 (/*ports*/);
    ...
    endmodule
Control Actions

- Edge Memory
- Single CoRAM
  - `coram_write(coh coram, void *offset, void *memaddr, int bytes);`
- Edge Memory
- 2 CoRAMs (concatenated)
  - `collective_write(coh coram, void *offset, void *memaddr, int bytes);`
- Src CoRAM
- Dst CoRAM
  - `coram_copy(coh src, coh dst, void *srcoffset, void *dstoffset, int bytes);`
- Edge Memory
- 2 CoRAMs (interleaved)
  - `collective_write(coh coram, void *offset, void *memaddr, int bytes);`

Design Case Study: MMMult

MMM in hardware
- N compute engines (1 per row of matrix)
- custom data network

C=AB

A SRAM

B

C SRAM

C SRAM

A/B SRAMs

Single Compute Engine

Network

Packet Generator

Address Generator

Packet Format

Single Compute Engine

Network

Packet Generator

Address Generator

Packet Format

OP PE_ID DATA
### MMMult with CoRAM

**Control Thread Program**

```c
void ctrl_thread() {
    for (j = 0; j < N; j += NB)
        for (i = 0; i < N; i += NB)
            for (k = 0; k < N; k += NB) {
                c_fifo_read(...);
                for (m = 0; m < NB; m++) {
                    c_collective_write(
                        ramsA,
                        m*NB,
                        A + i*N+k + m*N,
                        NB*dsz);
                }
                c_fifo_write(...);
            }
}
```

### Related Work in Abstracting

- **BORPH** [So, UCB PhD Thesis 2007]
  - support HW kernels with fixed interface/infrastructure
  - manage HW kernels as processes
  - communication between HW and SW processes
- **VirtualRC** [Kirchgessner, et al., ISFPGA 2012]
  - “virtual machine” for FPGA platforms
  - fixed SW API to interact with HW kernels
- **LEAP** [Adler, et al., ISFPGA 2011]
  - large, linear memory abstraction of local backing store
  - communicating with SW and remote memory
Outline

• Motivations
• CoRAM FPGA Architecture
• Development and evaluations
• Conclusions

Beneath the Abstraction

CoRAM FPGA

J. C. Hoe, CMU/ECE/CALCM, © 2014, BHSC-16-s18
CoRAM Cluster Design

Data response packet arrival in single clock

Data arrives sequentially from main memory, all accesses assumed 4-byte word-aligned

Cluster Resources

FPGA logic area for 32-way Cluster, 16B memory datapath, W=4

21% of V6-LX760 (if every BRAM used as CoRAM)
RTL Prototyping on FPGA

Outline

• Motivations

• CoRAM FPGA Architecture

• Development and evaluations

• Conclusions
BTW, we got it wrong......

- Architecture design is an exercise in compromise
  i.e., enough but not too much
- Do we really need an FPGA “architecture” just for computing?

CoRAM “Architecture” Revisited

- Must have hardened memory interface and NoC
- I like the decoupled computation paradigm
- Do I need to harden CoRAM?
  - fixed set of commands
  - insolate logic to one-side
Recap the Last Hour

- Future must look beyond general purpose
  - FPGAs offer impressive performance and efficiency
  - but need to be easy-to-use and portable
- The CoRAM Architecture/API
  - standard, scalable memory abstraction for FPGA
  - simplifies application development and portability
- Software is easy; hardware is hard?
  - C libraries and system calls is a big part of this impression
  - CPUs and GPUs are not much easier if to get the last ounce of performance