**FPGA Soft Core Processors**

**James Brakefield**

What is a soft core processor (uP)

What is a FPGA

IO pins

Clock network

D flip-flops

Lookup tables

RAM/ROM

Inferred

Vendor IP

Vendor primitives

Multiplier/DSP ALU

DDR memory controllers

FPGA tools

VHDL/Verilog/System Verilog

High level synthesis

Vendor tool flow

FPGA processors

Built-in processor(s)

Off-the-shelf soft core uP

DIY

State machine alternatives

Why soft core processors

Useful skill set

Term project

Architectural exploration

Hard Real-Time embedded

The problem: Complexity, complexity & complexity

The complex FPGA

The complexity of VHDL/Verilog

The complexity of the processor

**Soft Processor Basics:**

**Devise an instruction set**

**Minimal initial implementation**

**Single pipeline stage**

**Minimum set of instructions**

**Simulate/debug**

**Tracking LUT count and Fmax**

**Performance metrics**

Beyond the basic soft processor

Adding instructions

Inserting a data path

Adding pipeline stages

Adding modalities

*Advanced soft processor features*

*Adding floating-point*

*Adding external RAM and caches*

*Writing an assembler*

*Implementing a compiler*

*Adding peripherals*

1. **Soft Core Processor definition**
   1. Micro-processor implemented with FPGA resources. Usually coded in VHDL or Verilog.
   2. Why soft core processors
      1. Useful skill set
      2. Term project
      3. Architectural exploration
      4. Hard Real-Time embedded
2. **FPGA definition**
   1. IC with digital circuit with its resources connected by programmable/configurable wiring
   2. **Resources**
      1. **IO pins**
      2. **Clock network** & PLL/DLLs
      3. **D flip-flops**
      4. **Lookup tables**: three to six inputs and one or two outputs, may have carry chain
      5. **RAM**
         1. LUT RAM single port, dual port, another use for the LUT table
         2. Small block RAM 32x18, 32x20, 64x18
         3. Regular block RAM 128x36 to 1024x36
         4. Large block RAM 2Kx72, 4Kx72, 4Kx144
         5. DDR interface DDR, DDR2, DDR3, DDR4, QDR
      6. Multiplier/DSP ALU Considerable variety
      7. SERDES Gigabit serial, not needed here
      8. ARM processor Cortex M3, A9, A53, M6
   3. Vendors: Actel(Microsemi), Altera(Intel), Cypess PSOC, Lattice Semi + Silicon Blue, Xilinx
   4. Evaluation/Development boards: LEDs, Switches, Push buttons, VGA, Ethernet, IO pins
3. **Spartan-6 chip resources (xc6lx9-3csg324)**
   1. Avnet Spartan-6 FPGA LX9 Micro-Board
   2. 4 LEDs, 4 switches, 1 push button, 2 PMOD (16 IO total), Ethernet jack, 64MB DRAM
   3. (200) IO, (11,440) Dff, (5,720) 6LUTs, (64) 8K block ram, (16) 18x18 MUL/DSP, (2) PLL, (0) SERDES
4. **FPGA soft core processor definition**
   1. Microprocessor implemented using FPGA resources
   2. ROIS24\_24uP Instruction set/Programmer’s model:
      1. 64 24-bit registers
      2. 24 bit instruction with 6-bit op-code and three 6-bit register designators
      3. 24-bit by 1024 word block RAM main memory
      4. IO ports via D flip-flops and input multiplexor
   3. **Instruction set:**
      1. D: destination register, R: source register, S: 2nd source register, sN: 6-bit immediate/offset, sNN: 12-bit immediate/displacement, sNNN: 18-bit immediate, C: condition
      2. Three operand (DRS): add, add with carry, subtract, subtract with carry, AND, OR, XOR
      3. Two operand and immediate (DRsN): add, adc, and, or, xor
      4. Load/store (DRS, DRsN): load, store, load immediate
      5. Branch/Call (DsNN, DRS, DRsN): call
      6. Conditional (CsNN, CRsN): jump conditional, branch relative conditional
      7. IO (DRsN): in, out
      8. Prefix (sNNN): load 18-bit immediate into prefix register
      9. Register zero always reads as zero
5. **FPGA Tools**
   1. VHDL/Verilog/System Verilog
   2. High level synthesis
   3. Vendor tool flow
      1. User constraints
      2. Compile
      3. Simulate
      4. Place & Route
      5. Download
6. **FPGA processors**
   1. Built-in processor(s) ARM Cortex M3, 2xA9, 4xA53/2xR5, 2xPowerPC
   2. Off-the-shelf soft core processors NIOS2, MicroBlaze, opencores.org
   3. DIY Anything you want
   4. State machine alternatives Typically anything up to 20+ states
7. **The difficulties: Complexity, complexity, complexity**
   1. The complex FPGA Each FPGA family is different & getting more complex
   2. The complexity of VHDL/Verilog Must code with FPGA primitives in mind
   3. The complexity of the processor Large instruction sets & high performance
8. **Soft Processor Basics:**
   1. Devise an instruction set
   2. Minimal initial implementation
      1. Single pipeline stage
      2. Minimum set of instructions
      3. Short program
         1. Can start with case statement version
         2. Then move to Block RAM initialization
      4. Simulate/debug
   3. Tracking LUT count and Fmax
      1. Performance metrics/goals
9. **Beyond the Basic Soft Processor**
   1. Adding instructions Customization, trade-off studies
   2. Inserting a data path Normal practice: bit LUT count reduction & better Fmax
   3. Adding pipeline stages Common, optimum is short pipes
   4. Adding modalities Addressing modes, bit flags
10. **Advanced Soft Processor Features**
    1. Adding floating-point Typically takes 2K or more LUTs
    2. Adding external RAM and caches Go together, DDR interfaces available
    3. Writing an assembler Subroutine for each kind of instruction
    4. Implementing a compiler
    5. Adding peripherals Many available off-the-shelf
    6. Barrel processors Pipeline contains one stage of each thread
    7. Multiple dispatch Simple approach: multiple register sets