CS250: DISCUSSION #4

Brian Zimmer

9/22/2011

OVERVIEW

- Logistics
- Debugging
- Lab 2 Hints
- ALU

LOGISTICS

- Lab #2 is due on Monday, Oct. 3rd at 1pm (pushed 1 week)
- Make sure your work is both committed AND pushed to github

DEBUGGING

- General idea: Debug with the C++ emulator only
- Various methods of debugging
  - View waveforms
  - Text tracing in testbench
- Various references
  - ISA simulator output
  - Instruction by instruction

GENERATION WITH CHISEL

Inline

- ```
  if(isBranchPrediction){
      something := val
  }
  ```

- ```
  if(isBranchPrediction){
      val something = val
  }
  ```

IO Bundles

```
val ctrl  = new IoCtrlToDpath().flip();
val ctrl_bp = new IoCtrlToDpathBP().flip();
```

IO

```
if(isBranchPrediction){
  c.io.ctrl_bp  <> d.io.ctrl_bp;
  c.io.dpath_bp  <> d.io.dpath_bp;
}
```

Components

```
var btb:DpathBTB = null;
if(isBranchPrediction){
  btb = new DpathBTB;
  btb.io.current_pc4 := ...
}
```

DESIGN SPACE GENERATION

- gen_design_space.py
- Creates symbolic link trees to mimic project directory
- Within each design point, run as if normal project

ALU OPTIMIZATION

- Pure behavioral:
  - Inefficient
- Need to support:
  - ADD, SUB, SLT, SLTU, SLTI, SLTIU
  - SL, SR, SRA
  - AND, OR, NOR
- Eg. SUB xc, xa, xb
  - xc = xa + (~xb + 1) (use the carry input)
- Draw a single ALU, and mux its inputs

Cell	Module	Parameters	Contained Operations
add_x_10_1	DW01_add	width=32	add_10
sub_x_10_2	DW01_sub	width=32	sub_10
lt_x_10_3	DW_cmp	width=32	lt_10
lt_x_10_4	DW_cmp	width=32	lt_10_2
ash_10_5	DW_leftsh	A_width=32 SH_width=5	sll_10
ashr_10_6	DW_rightsh	A_width=32 SH_width=5	srl_10
ashr_10_7	DW_rightsh	A_width=32 SH_width=5	sra_10
lt_x_10_8	DW_cmp	width=32	lt_10_3
lt_x_10_9	DW_cmp	width=32	lt_10_4

SLT

- Algorithm: To test in1 < in0, do in1 - in0
  - If the MSB is 1 (then answer is negative and slt is true)
  - If the MSB is 0 (then the answer is positive and slt is false)
- What about overflow? eg. -4 < 3
  - -4 - 3 = 100+100+0011=1**0**10 MSB says slt false, but we know that it is true. Rule: Flip answer if overflow
  - What is overflow? eg. 3 < -4 has overflow but is correct
    - Answer: Operands (of a and ~b) same sign but result is different sign