Part 1b Computer Design

Lecture 6: CLARVI
Our RISC-V Implementation

Prof. Simon Moore

UNIVERSITY OF CAMBRIDGE
Computer Laboratory
Computer Architecture Group

Processing an instruction

- Steps to process one instruction:
  - Instruction fetch
  - Decode
  - Register fetch } often combined
  - Branch (optional) – combined with decode or execute
  - Execute (ALU operation)
  - Memory access (optional)
  - Write-back any result to destination register

Instruction Fetch

- Program counter (PC) used as an address to do a memory access
- Takes time to do the memory access
- Result placed in an instruction register (IR) or similar
- PC=PC+4
  - gets PC ready to fetch the next instruction

Decode & Register Fetch

- Decoding expands the instruction into a more usable state, e.g.:
  - 32-bit sign extended immediate
  - is the instruction a branch/jump?
  - does the instruction do memory access (load/store)?
  - what arithmetic to do (e.g. add, sub, or, and, xor, etc.)
    - lots of instructions end up being add, e.g. address calculation for load or store instructions
  - what are the source registers and the destination register
    - easy to figure out on RISC-V
- Register fetch
  - look up the source registers

Branch

- Branch (optional, may be done with decode or execute)
  - PC = PC + signExtend(immediate)
  - return address may be stored in a register

Execute, Memory Access, Write-back

The following are usually performed as separate steps:

- Execute
  - performs an integer or logical operation using an arithmetic logic unit (ALU)
  - pretty easy, but this is what does all the work!
- Memory access (optional)
  - use the address calculated in Execute to either:
    - load data (8-bits, 16-bits or 32-bits)
    - store data (8-bits, 16-bits or 32-bits)
- Write-back
  - write any result to Rd, the destination register

Example Instructions

- Add immediate: addi x5, x6, 8
- Action: x5=x6+8
- It is an I-type instruction:

12-bits	5-bits	3-bits	5-bits	7-bits
imm[11:0]	rs1	funct3	rd	opcode
8	6	3'b000	5	7'b0010011

- Load word: lw x5,8(x6)
- Action: x5=mem[x6+8]
- It is also an I-type instruction:

12-bits	5-bits	3-bits	5-bits	7-bits
imm[11:0]	rs1	funct3	rd	opcode
8	6	3'b010	5	7'b0000011