RISC-V Processor Design 🚀

Building Tiny Vedas

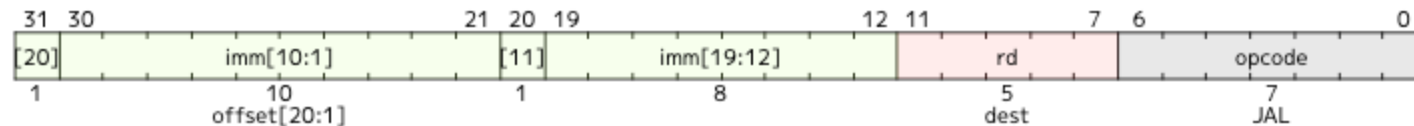Marco Spaziani Brunella, PhD

Lecture 10

Agenda

- Branches

- Pipeline Flushing

- Branch Penalty

Branches

- Instructions used to implement conditional code execution: if, else, while, for, etc.
- Branches can be unconditional or conditional

Unconditional Branches in RISC-V

- JAL and JALR operation, they involve also a reg file side effect generation
    - JAL -> pc += sext(offset), rd = pc + 4
    - JALR -> rd = pc + 4, pc = rs1 + sext(offset)

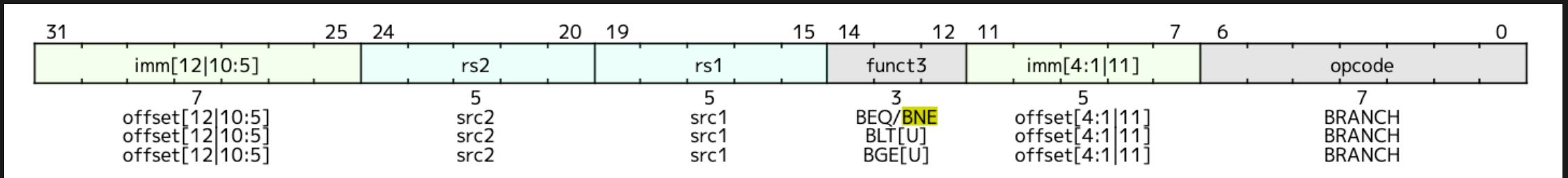| 31 30		21 20 19		12 11	7 6	0
[20]	imm[10:1]	[11]	imm[19:12]	rd	opcode	
1	10	1	8	5	7	
	offset[20:1]			dest	JAL	

The indirect jump instruction JALR (jump and link register) uses the I-type encoding. The target address is obtained by adding the sign-extended 12-bit I-immediate to the register *rs1*, then setting the least-significant bit of the result to zero. The address of the instruction following the jump (pc+4) is written to register *rd*. Register x0 can be used as the destination if the result is not required.

31	20 19	15 14	12 11	7 6	0
imm[11:0]	rs1	funct3	rd	opcode	
12	5	3	5	7	
offset[11:0]	base	0	dest	JALR	
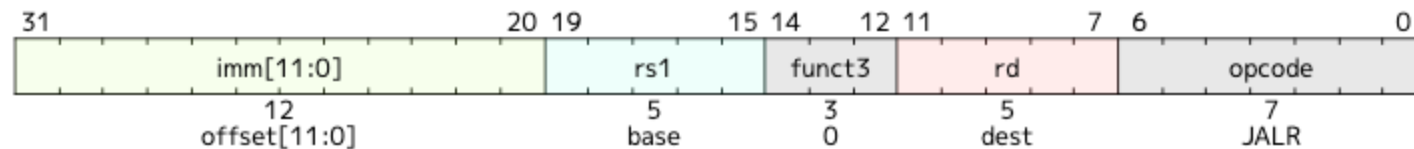
Conditional Branches in RISC-V

- BEQ, BNE, BLT, BGE, BLTU, BGEU

- All of them follow the same pattern:
  - pc += sext(offset) if cond is true
  - pc += 4 if cond is false

31	25	24	20	19	15	14	12	11	7	6	0
imm[12|10:5]		rs2		rs1		funct3		imm[4:1|11]		opcode	
7		5		5		3		5		7	
offset[12|10:5]		src2		src1		BEQ/BNE		offset[4:1|11]		BRANCH	
offset[12|10:5]		src2		src1		BLT[U]		offset[4:1|11]		BRANCH	
offset[12|10:5]		src2		src1		BGE[U]		offset[4:1|11]		BRANCH	

Implementing Branching in the Pipeline

- We fetch instructions sequentially by default --> Assume the branch is never taken

- Whenever we change the pc, it means that the instructions we fetched are now invalid

- We need to flush the pipeline: get rid of the instructions we fetched and get new ones

Branch Penalty

- Once we take a branch and flush the pipeline, we need to wait for the pipeline to be filled again
- We thus incur a penalty in taking a branch, as every time we take a branch we need to fill again the pipeline
- The branch penalty, measured in clock cycles, is the same as the pipeline stages

Improvements (not covered in this course)

- Branch preditcion: Based on how many times a branch has been taken in the past, we can predict if it will be taken or not

- Different ways of doing this, with a varying grade of complexity and prediction performance:
  - 2-bit branch preditctor
  - TAGE predictor