CS152 Section 5 Worksheet

Q1. Microprogramming (2020 MT1 Q2)

The `SWITCH` instruction has the following format:

```
SWITCH rs1, rs2, imm
```

The operands consist of two source registers and one **B-type** immediate:

`rs1`: Zero-based index to select a branch table entry

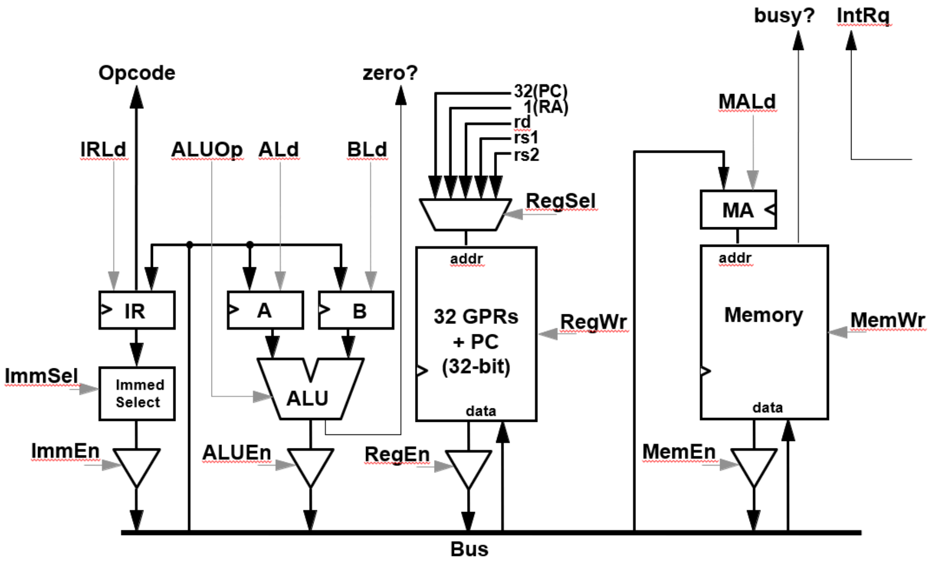
`rs2`: Pointer to a branch table in memory

`imm`: Limit, the index of the last table entry ($N - 1$)

The *table* operand (`rs2`) points to an array in memory with N word-sized entries, each holding a branch target address:

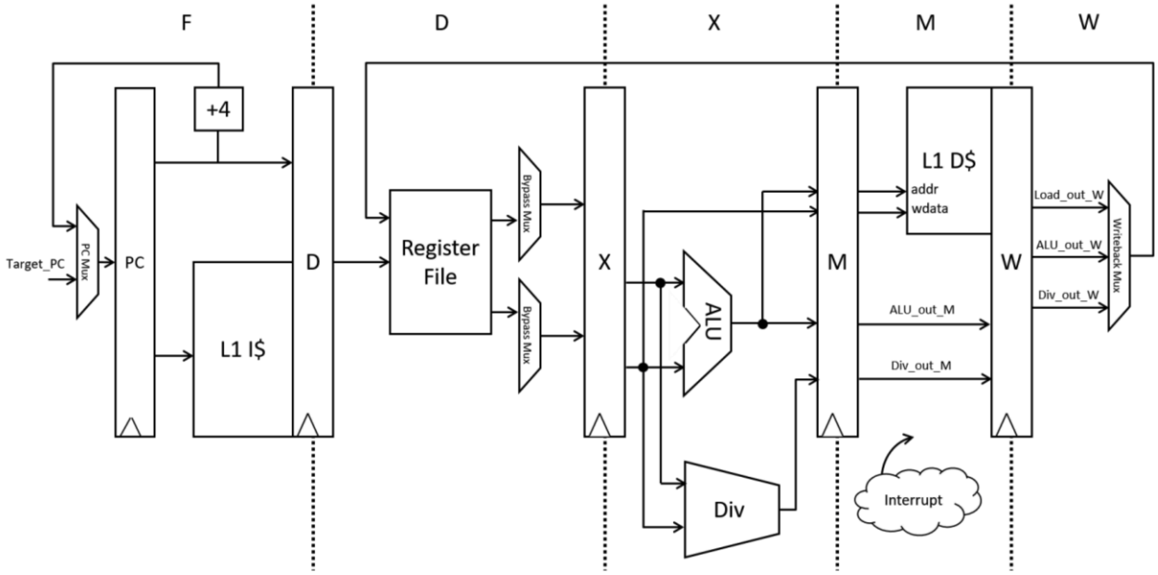
Address	Content
`table + 0`	`target_0`
`table + 4`	`target_1`
`table + 8`	`target_2`
…	…
`table + (4×limit)`	`target_last`

The *index* (`rs1`) is compared with *limit* (`imm`) to check that it is within the table range. If *index* ≤ *limit*, then the processor branches to the address stored in the *table[index]* entry. Otherwise, if *index* > *limit*, no branch is taken, and execution continues at PC + 4 as usual.



State	Pseudocode	ld IR	RegSel	RegWr	enReg	ld A	ld B	ALUOp	en ALU	ld MA	Mem Wr	en Mem	Imm Sel	en Imm	μ Br	Next State
FETCH0	MA := PC; A := PC	*	PC	0	1	1	*	*	0	1	0	0	*	0	N	*
	IR := Mem	1	*	0	0	0	*	*	0	0	0	1	*	0	S	*
	PC := A + 4	0	PC	1	0	0	*	INC_A_4	1	*	0	0	*	0	D	*
...																
NOP0	μBr to FETCH0	*	*	0	0	*	*	*	0	*	0	0	*	0	J	FETCH0
SWITCH0:																

Q2. Pipelining (2020 MT1 Q3)



a) What is the latency of a divide operation when the iterative divider produces 2 bits per cycle until it outputs a full 32-bit result?

b) What is the occupancy of a divide operation in cycles?

c) Note that the `div` instruction in RISC-V cannot raise a data-dependent exception. To avoid pipeline stalls while a multi-cycle divide operation is in progress, the pipeline control logic allows subsequent instructions that do not depend on the divide result to be issued and completed before the divide has completed.

What additional hazards might be caused by `div` instructions, aside from the structural hazard on the divider itself? If any, describe how they could be resolved using an interlock.