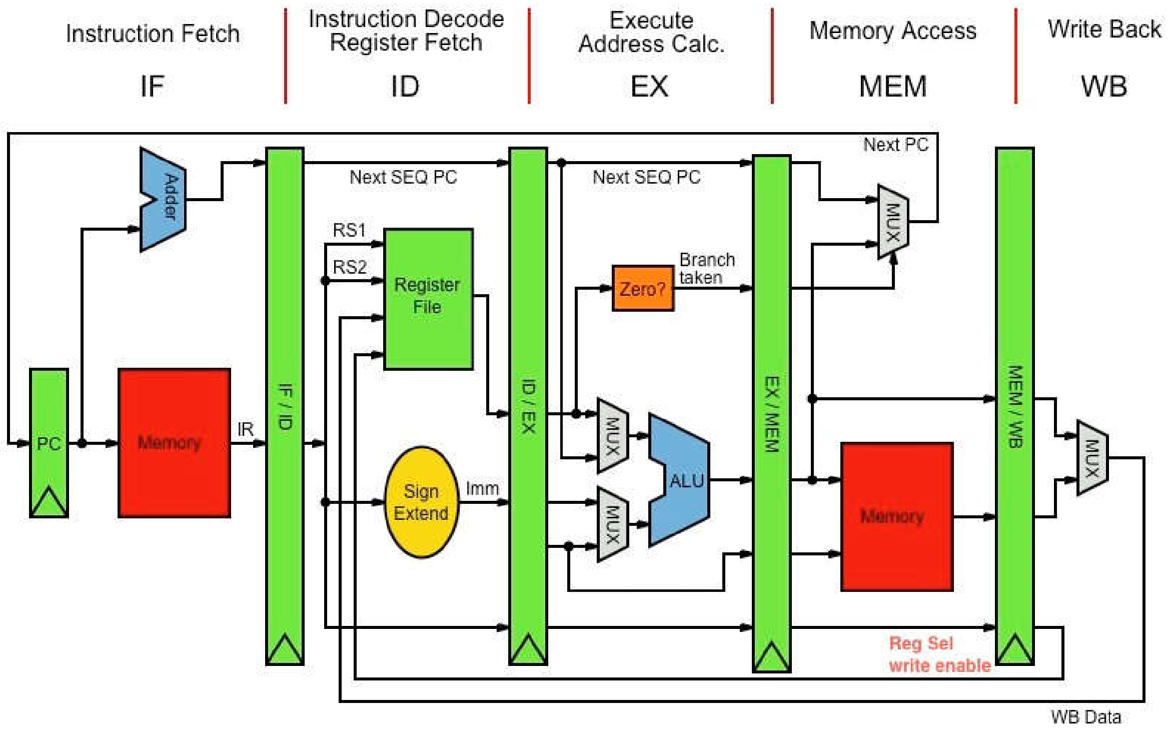
A naïve RISC-V CPU

January 4, 2020
Wu Runzhe

Now let's talk about the general design

- A standard 5-stage pipeline with data forwarding



Instruction Cache

- No I-cache?
  - 5~6 cycles to fetch an instruction
  - So pipeline what?
  - Fake pipeline

Instruction Cache

- A direct mapping I-cache
  - 1,024 bytes
  - 256 instructions
- Supports consecutive hits
  - one instruction per cycle
- A victim cache
  - Sounds great
  - But actually useless

Data Cache

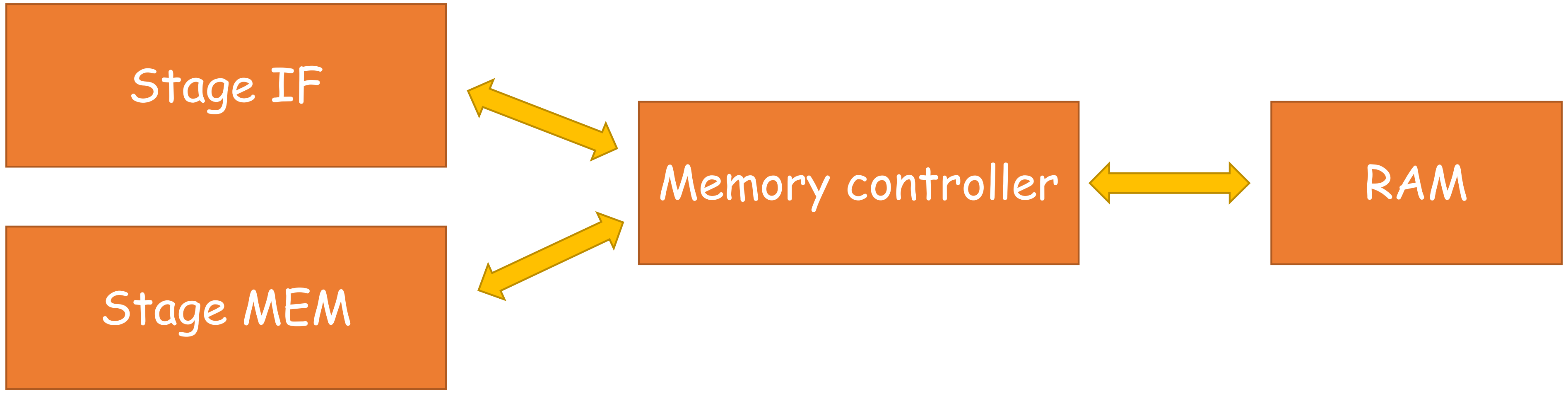
- A direct mapping D-cache
  - 64 bytes
  - Why so small
    - useless

Branch Prediction

- Tournament Predictor
  - Global Predictor & Local Predictor
  - A selector
- Branch Target Buffer
- Predict at stage IF
- Correct
  - No stall
- Incorrect
  - Recover in stage EX by stalling 1 cycle

Memory controller

- To resolve structural hazard
- Stage MEM priority over stage IF

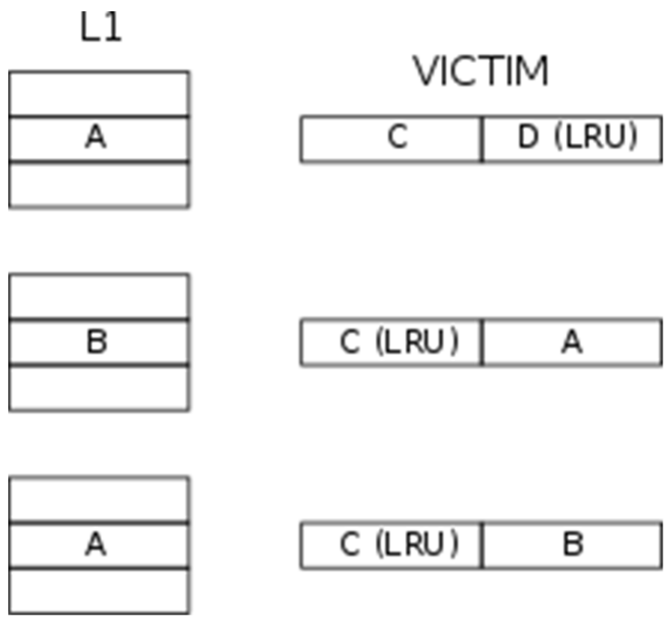


Performance

- 130 MHz, all test cases passed
- pi.c: 0.996875 s
- When higher than 130 MHz, Bulgarian.c failed
  - Because of the bad uart buffer?

Now let's focus on the details

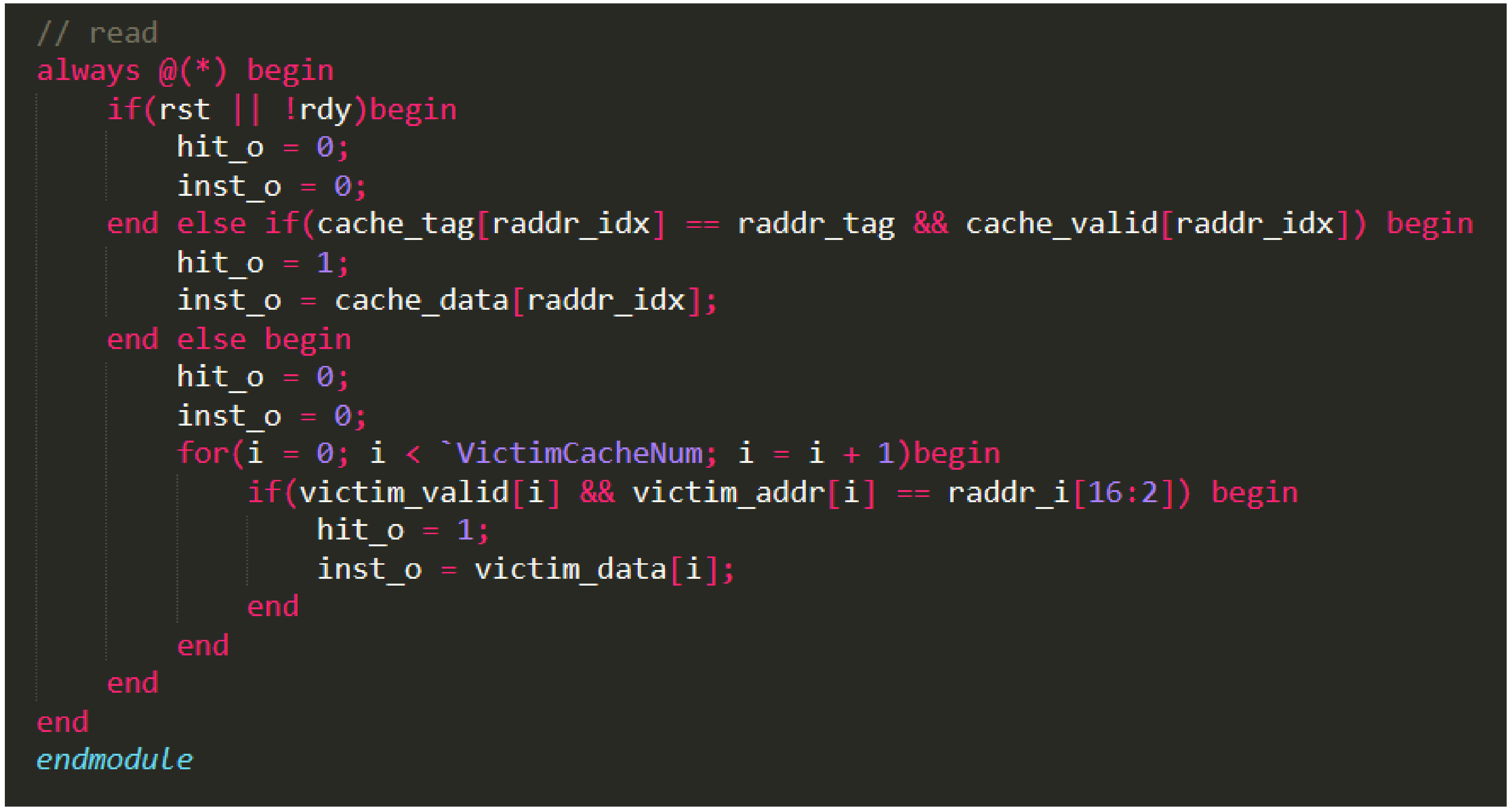
Victim cache



Victim cache

- A small and fully associative cache that stores all the blocks evicted from some cache
- I don't know why but it does little to reduce miss rate
- Because I-cache is large enough
- Speed up ≈ 1

Victim cache



```
// read
always @(*) begin
    if(rst || !rdy)begin
        hit_o = 0;
        inst_o = 0;
    end else if(cache_tag[raddr_idx] == raddr_tag && cache_valid[raddr_idx]) begin
        hit_o = 1;
        inst_o = cache_data[raddr_idx];
    end else begin
        hit_o = 0;
        inst_o = 0;
        for(i = 0; i < `VictimCacheNum; i = i + 1)begin
            if(victim_valid[i] && victim_addr[i] == raddr_i[16:2]) begin
                hit_o = 1;
                inst_o = victim_data[i];
            end
        end
    end
end
endmodule
```

1/4/20

Tournament Predictor

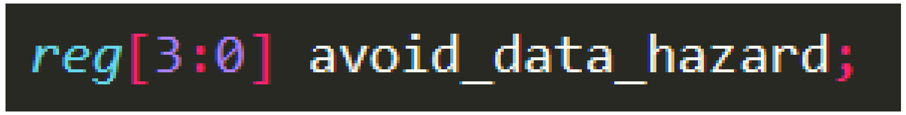
- 2 predictors: 1 based on global information and 1 based on local information, and combine with a selector
- Global Predictor 4K entries, indexed by the history of the last 12 branches; each entry in the global predictor is a standard 2-bit saturating counter
- Local Predictor 32 entries, indexed by the address of the branch instruction; each entry is a standard 2-bit saturating counter

Tournament Predictor

- Selector
  - A 2-bit saturating counter
    - 0 stands for global predictor
    - 1 stands for local predictor
- When a prediction produced by some predictor is correct, we should attach more importance to it.
- Otherwise we should incline to the other predictor.

Pipelined <-> Not pipelined

- About debugging
- A register named "avoid_data_hazard" to stall some specific cycles after an instruction is fetched in stage IF
- "avoid_data_hazard" = 0
  - Pipelined
- "avoid_data_hazard" > 9
  - Not pipelined



```
reg[3:0] avoid_data_hazard;
```

Thank You!