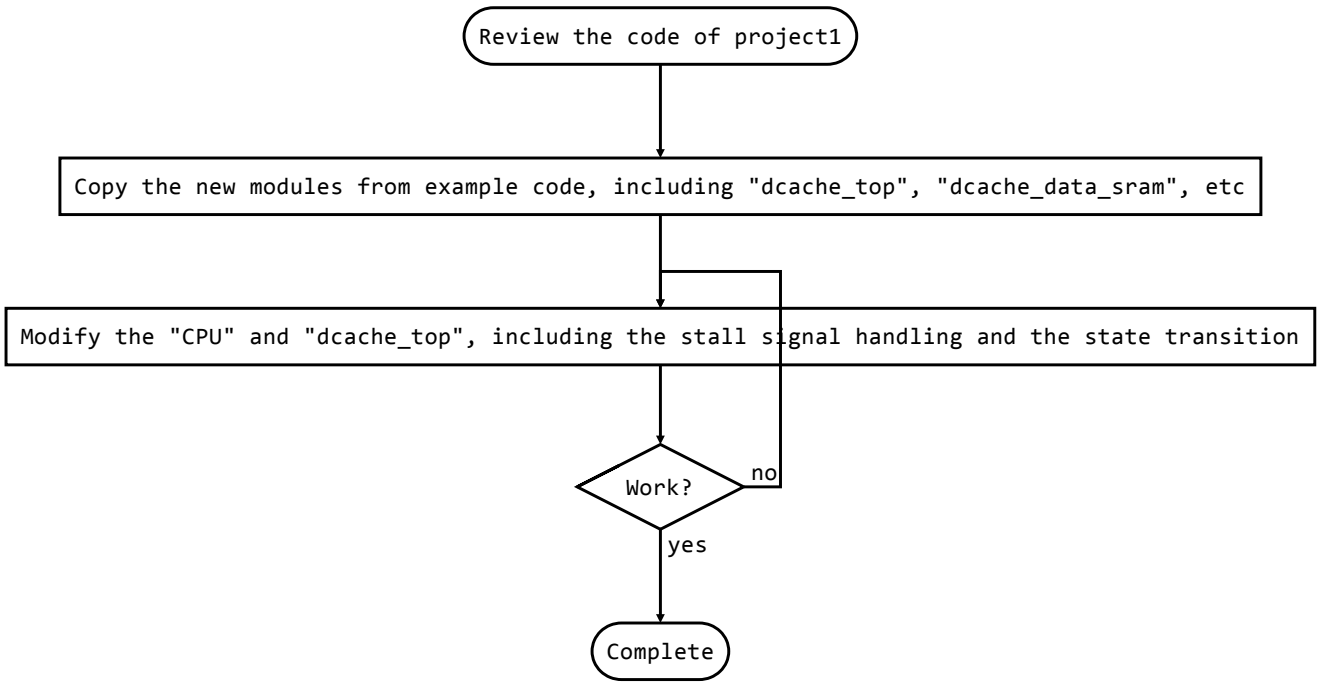
Computer Architecture Project2 Report

Members amd Team Work

> Link the sub-modules of the CPU
> Handle the stall signal from cache
> — B03902007 鄭德馨
>
> Setup the read/write data of the cache controller
> Report
> — B03902065 陳奕先
>
> Implement the state transition of the cache controller
> Overall debug
> — B03902015 簡瑋德

Project Implementation



Cache Controller Implementation

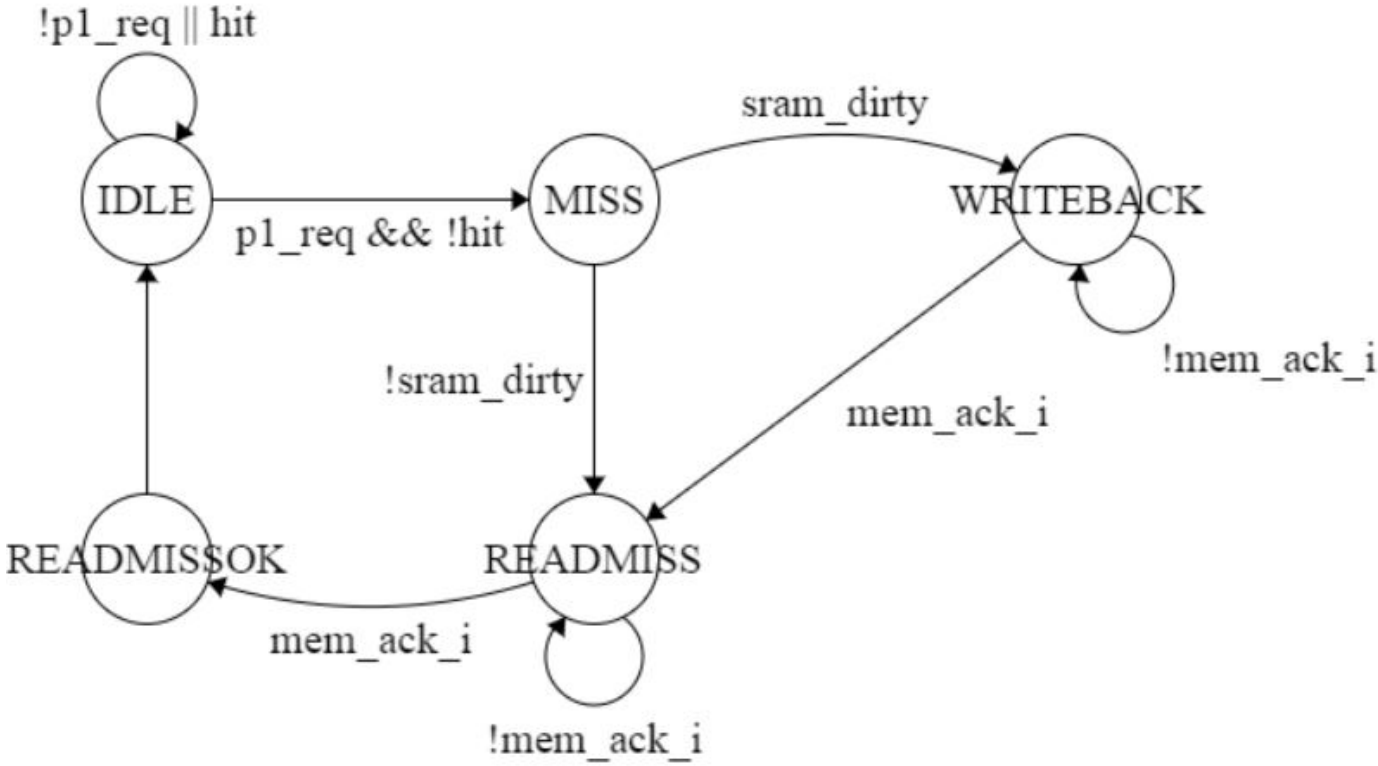
- We should compare the tags to check whether there's a read miss or not

```
assign  {hit, r_hit_data} =
    (p1_req && sram_valid && p1_tag == sram_cache_tag[21:0]) ?
    {1'b1, sram_cache_data} : {1'b0, 256'b0};
```

- Setup the 32-bit output data, given the 256-bit cache data and the offset

```
always@(p1_offset or r_hit_data) begin
        case(p1_offset)
                5'h00:
                        p1_data <= r_hit_data[31:0];
                5'h04:
                        p1_data <= r_hit_data[63:32];
                5'h08:
                        p1_data <= r_hit_data[95:64];
                5'h0c:
                        p1_data <= r_hit_data[127:96];
                5'h10:
                        p1_data <= r_hit_data[159:128];
                5'h14:
                        p1_data <= r_hit_data[191:160];
                5'h18:
                        p1_data <= r_hit_data[223:192];
                5'h1c:
                        p1_data <= r_hit_data[255:224];
                default
                        p1_data <= 32'd0;
        endcase
end
```

- Setup the 256-bit data for sram in the same way, given the original data and the 32-bit modified part
- Modify the signals of the cache and define the reactions of the state transitions



From	To	Actions
STATE_IDLE	*STATE_IDLE*	None
STATE_IDLE	*STATE_MISS*	None
STATE_MISS	*STATE_WRITREBACK*	*mem_enable*=1 *mem_wirte*=1 *write_back*=1
STATE_MISS	*STATE_READMISS*	*mem_enable*=1 *mem_wirte*=0
STATE_READMISS	*STATE_READMISSOK*	*mem_enable*=0 *cache_we*=1
STATE_READMISS	*STATE_READMISS*	None
STATE_READMISSOK	*STATE_IDLE*	*cache_we*=0
STATE_WRITREBACK	*STATE_READMISS*	*mem_write*=0 *wirte_back*=0
STATE_WRITREBACK	*STATE_WRITREBACK*	None

Problems and Solutions

- Problem: Hard to Debug, since there're so many ports and modules
- Solution: Besides unit-tests, we print the values of the ports and pipeline registers by modifying the testbench

Testbench Details

- Line 81: `Data_Memory.memory[0] = 256'h5 // n = 5 for example;`
- Line 95 to 106:

```
95  if(counter == 150) begin       // store cache to memory
96      $fdisplay(outfile, "Flush Cache! \n");
97      for(i=0; i<32; i=i+1) begin
98          tag = CPU.dcache.dcache_tag_sram.memory[i];
99          index = i;
100         address = {tag[21:0], index};
101         Data_Memory.memory[address] = CPU.dcache.dcache_data_sram.memory[i];
102     end
103 end
104 if(counter > 150) begin        // stop
105     $stop;
106 end
```