ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ

CẤU TRÚC MÁY TÍNH – EE3043

BÁO CÁO THÍ NGHIỆM – LỚP L01 – HK232

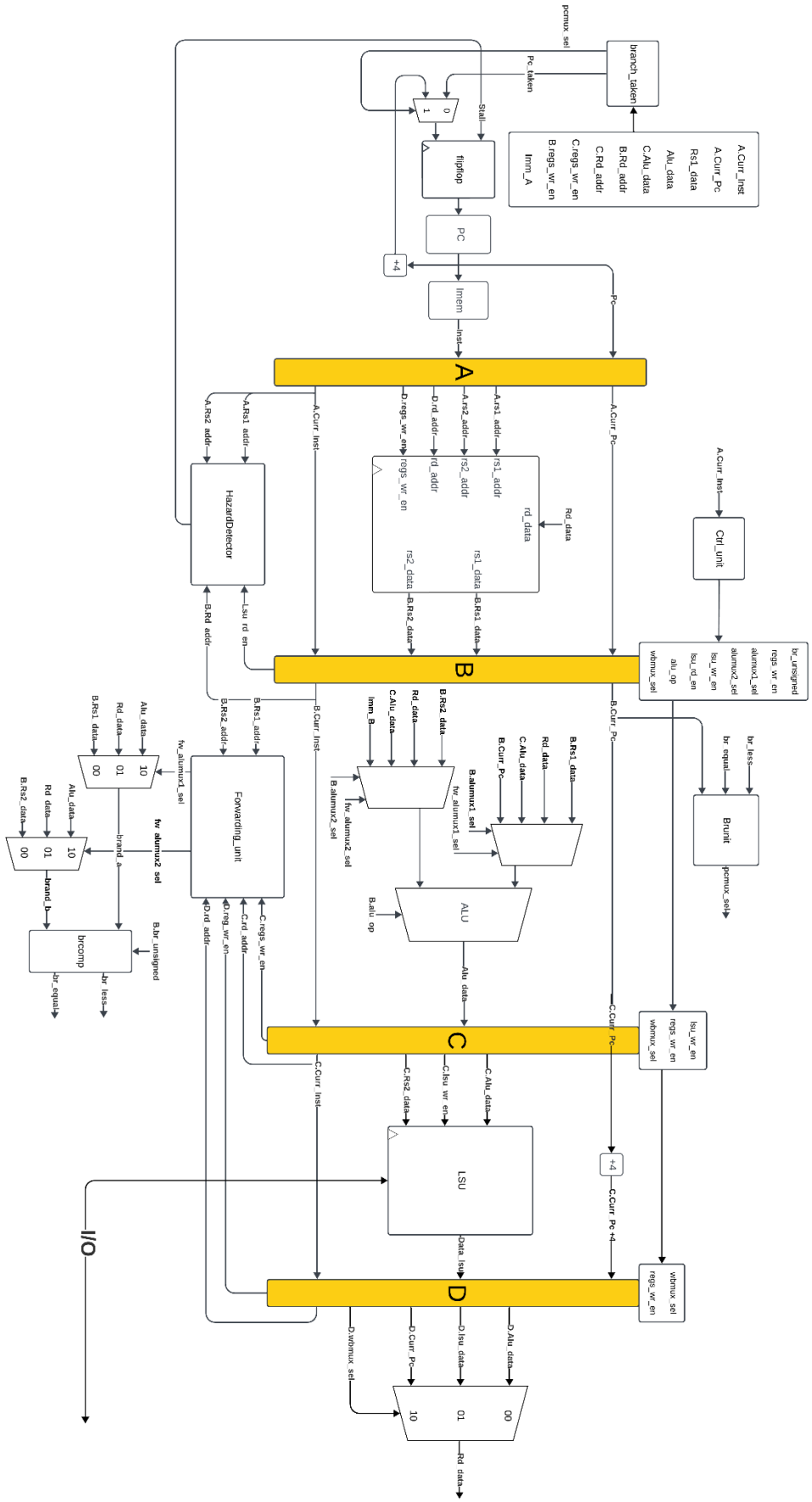
Milestone3: Design of a Pipelined Processors

NHÓM 9: HUỲNH PHƯỚC TOÀN – 2012230
NGUYỄN ĐỨC CHÍNH – 2012739

THÀNH PHỐ HỒ CHÍ MINH, THÁNG 05/2024

MỤC LỤC

1. SƠ ĐỒ KHỐI TỔNG QUÁT



2. MỘT SỐ MODULE MỚI

a. BRANCH_TAKEN:

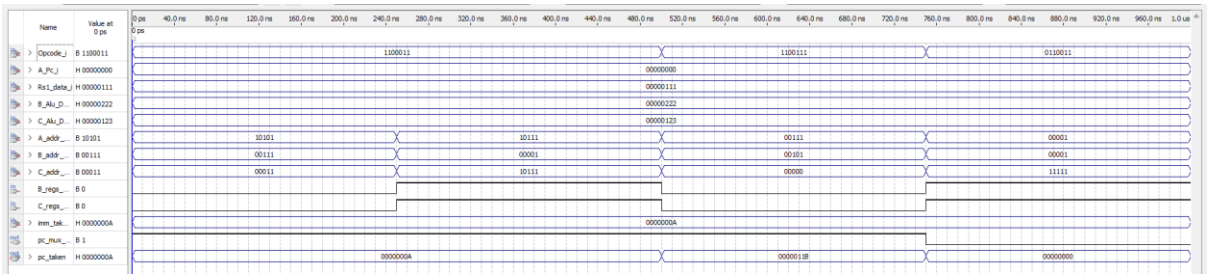
- Tóm gọn code (chi tiết xin mời xem tại github: https://github.com/tlatonf/venmac/tree/20240511_milestone3/milestone3

```
  assign data = ((B_regs_wr_en_i) && (B_addr_Rd_i != 0) && (B_addr_Rd_i
== A_addr_Rs1_i)) ? B_Alu_Data_i:((C_regs_wr_en_i) && (C_addr_Rd_i != 0)
&& (C_addr_Rd_i == A_addr_Rs1_i)) ? C_Alu_Data_i : Rs1_data_i;

  assign pcmux_sel_i = (Opcode_i == 7'b1100011 || Opcode_i == 7'b1100111
|| Opcode_i == 7'b1101111) ? 1'b1 : 1'b0;

  assign pc_taken = (Opcode_i == 7'b1100011 || Opcode_i == 7'b1101111) ?
(A_Pc_i + imm_taken_i) :  (Opcode_i == 7'b1100111) ? (data + imm_taken_i)
: 14'b0;
```

- Verify: Dùng phương pháp waveform, truyền vào các giá trị OPCODE, A.Pc, Rs1_data, B.Alu_data,… sau đó kiểm tra thủ công xem module có thực hiện đúng yêu cầu thiết kế hay không.



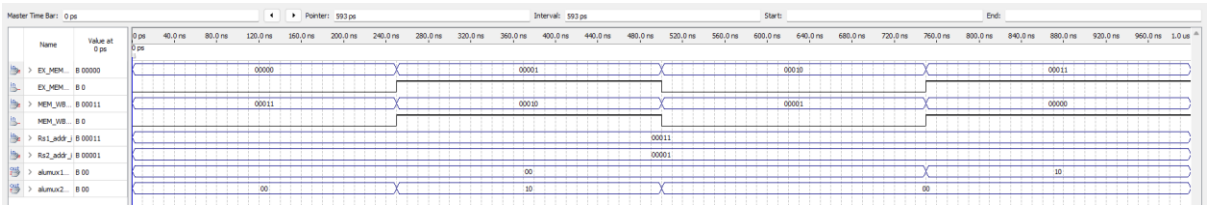
b. FORWARDING_UNIT:

- Tóm gọn code (chi tiết xin mời xem tại github: https://github.com/tlatonf/venmac/tree/20240511_milestone3/milestone3

```
    assign alumux1_sel_o = ((EX_MEM_Regs_wr_en_i) && (EX_MEM_Rd_i != 0) &&
(EX_MEM_Rd_i  ==  Rs1_addr_i))  ?  2'b10  :  ((MEM_WB_Regs_wr_en_i)  &&
(MEM_WB_Rd_i != 0) && (MEM_WB_Rd_i == Rs1_addr_i)) ? 2'b01 : 2'b00;

    assign alumux2_sel_o = ((EX_MEM_Regs_wr_en_i) && (EX_MEM_Rd_i != 0) &&
(EX_MEM_Rd_i  ==  Rs2_addr_i))  ?  2'b10  :  ((MEM_WB_Regs_wr_en_i)  &&
(MEM_WB_Rd_i != 0) && (MEM_WB_Rd_i == Rs2_addr_i)) ? 2'b01 : 2'b00;
```

- Verify: Dùng phương pháp waveform, cố định giá trị Rs1_addr và Rs2_addr, nhập ngẫu nhiên giá trị của EX_MEM_Rd, MEM_WB_Rd, EX_MEM_regs_wr và MEM_WB_regs_wr. Thực hiện mô phỏng ta thấy giá trị alumux1_sel và alumux2_sel có kết quả đúng với yêu cầu thiết kế.

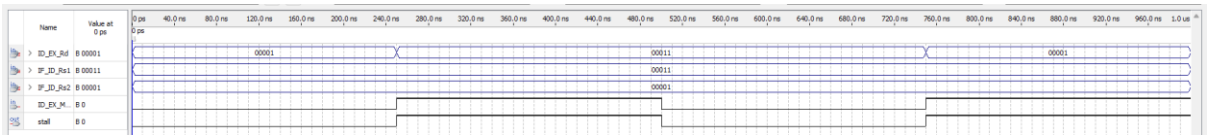


c. HAZARD DETECTOR:

- Code chi tiết:
  https://github.com/tlatonf/venmac/tree/20240511_milestone3/milestone3

```
    assign stall = (ID_EX_lsu_rd) ? ((ID_EX_Rd == IF_ID_Rs1) || (ID_EX_Rd
== IF_ID_Rs2)) : 1'b0;
```

- Verify: Dùng phương pháp waveform, cố định giá trị IF_ID_Rs1, và IF_ID_Rs2, cho giá trị của ID_EX_Rd và ID_EX_Memread ngẫu nhiên để kiểm tra. Thực hiện mô phỏng ta thu được giá trị của biến Stall.



d. DATA_TRANSFER:

- Cơ sở thiết kế: tạo các giá trị lưu trữ của các Stage bằng cách sử dụng :
  typedef struct packed{
  …
  } Stage
- Như vậy, các giá trị của Stage sẽ được lưu trữ và sử dụng trong các Stage tiếp theo.
- Tóm gọn code (chi tiết xin mời xem tại github:
  https://github.com/tlatonf/venmac/tree/20240511_milestone3/milestone3

3. MỘT SỐ VẤN ĐỀ GẶP PHẢI

Vấn đề 1: Gặp lỗi *DEFINE LANG_MAP DLHSYN*

- Nhóm tụi em chưa tìm hiểu được đây là lỗi gì, nó không thường xuyên xảy ra, tuy nhiên tụi em mất rất nhiều thời gian để xác định nguyên nhân của nó.
- Giải pháp: Tụi em xoá tất cả file output từ Xcelium, sau đó chạy lại thì thành công.

Vấn đề 2: Gặp hơn 500 cảnh báo do gặp kí tự đặt biệt trong file

- Lí giải: Do nhóm em code bên môi trường windows, sau đó mới tiến hành chạy mô phỏng ở linux. Mà kí tự kết thúc một dòng ở windows là CR LF, còn bên linux chỉ là LF. Đây là một kiến thức khá cơ bản, nhưng phải rất lâu sâu thì nhóm mới nhận ra sai lầm của mình.
- Giải pháp: Thực hiện việc chuyển đổi từ DOS sang UNIX khi copy file từ windows sang linux. Thực hiện bằng lệnh có sẵn trong linux:

```
find . -type f -print0 | xargs -0 dos2unix
```

```
Analysis summary :

 Errors    : (39)
  CBPAHI (37)     VERCAS (2)

 Warnings  : (570)
  BADSYS (3)      BBXSIG (1)      BITUNS (1)      BLKSQB (34)
  CBYNAM (24)     CONSBS (2)      CTLCHR (1)      FFWASR (63)
  IMPDTC (32)     IMPTYP (2)      INDXOP (2)      INIUSP (3)
  INTTOB (10)     IPRTEX (1)      LCVARN (63)     LRGOPR (8)
  MAXLEN (40)     MICAWS (1)      MRSTDT (1)      NBCOMB (3)
  NBGEND (5)      NCASEX (1)      NEFLOP (34)     NOBLKN (7)
  NUMSUF (14)     OLDALW (3)      PADMSB (1)      POOBID (2)
  RDBFAS (14)     REVROP (1)      SENCMW (1)      SEPLIN (2)
  STYVAL (51)     SYNPRT (45)     TDOPKG (4)      TRUNCZ (17)
  UCCONN (4)      ULCMPE (6)      URDREG (1)      USEPRT (37)
  VARUAW (22)     VLGMEM (3)

 Notes     : (307)
  ALOWID (8)      CLKINF (1)      DECLIN (5)      FFASRT (269)
  IDLENG (5)      NUMDFF (1)      PRTCNT (18)

Analysis complete.
```

Vấn đề 3: Không tìm được nguyên nhân và cách sửa lỗi *halstruct: CBPAHI Combinatorial path crossing multiple units drives.*

- Khi checksyntax bằng Xcelium thì nhóm em vẫn còn 33 lỗi chưa được giải quyết (không báo lỗi khi compile bằng quartus). Tất cả đều báo là *Combinatorial path crossing multiple units drives* (bao gồm 1 lỗi do pc, và 32 lỗi do 32 thanh ghi regfile).
- Tụi em đã thử nhiều cách nhưng vẫn không sửa được lỗi này.

HẾT !