Computer Architecture Project 2025

Cairo University
Faculty Of Engineering
Computer Department

Name	ID
مصطفى محمد عبداللاه محمد	91240765
مسلم احمد محمد اسماعيل سيد	91241062
مريم سامح يسن عبدالمطلب	91240739
مريم محمد رمضان محمد	91240747

Delivered to

Dr. Mostafa Ghouneem

Eng. Ahmed Hashish

Eng. Mohammad Abdallah

Hazards

- *Instruction fetching while another instruction using memory:*

  The Hazard Detection Unit will get the MemRead/MemWrite signal from EX/MEM register, and it will set the PC Stall, set the IF/ID.Flush and that will lead to a NOP then fetching the right instruction.
  (Stall PC & Flush IF/ID)

- *Immediate fetching while another instruction using memory:*

  The Hazard Detection Unit will get the MemRead/MemWrite signal from EX/MEM register and the IMM.Flush of the currently decoded instruction then it will set the PC Stall, set IF/ID.Stall, set the ID/EX.Flush and that so after PC will take the right immediate value.
  (Stall PC & Stall IF/ID & Flush ID/EX)

- *Instruction Decoding while another instruction in WB:*

  The Register File will write in Falling Edge and read in Rising Edge.

- *Swap:*

  The Hazard Detection Unit gets Control Unit Swap, EX/MEM.Swap to process the swap instruction that will take 3 cycles.
  (Stall PC & Stall IF/ID for 2 cycles)

- *Forwarding:*

  The Forward unit takes EX/MEM.WB the Rs1, Rs2 from ID/EX and Rd from EX/MEM to perform ALU to ALU forwarding when Rs1 or Rs2 = EX/MEM.Rd.
  It also takes MEM/WB.WB, and MEM/WB.Rd to perform Memory to ALU forwarding when RS1 or Rs2 = MEM/WB.Rd.
  (Forwarding to ALU or StoreData)

- *Load-Use:*

  The Hazard Detection Unit gets IF/ID.Rs1, IF/ID.Rs2, ID/EX.Rd, MemRead, SP+ve and when Rs1 or Rs2 = Rd and MemRead or SP+ve = 1, it sets IF/ID.Stall and PC-Stall and ID/EX.Flush.

  (Stall PC & Stall IF/ID & Flush ID/EX)

Reg Sizes

Pipeline Registers:

1. IF/ID + IMM (46 bits)
    - IF/ID -> 14 bits (instruction)
    - IMM -> 32 bits
2. ID/EX + Flags (92 bits)
    - Swap / OUT / Call / Imm / Buff / INC / WB (control signals) -> 1 bit
    - FEnable + SetC / SP / M (control signals) -> 2 bits
    - EX -> 3bits
    - ID/EX -> 73 bits
        - Rd + Rs1 + Rs2 Addresses -> 3*3bits
        - Data1 + Data2 -> 2*32bits
    - Flags -> 3 bits
3. EX/MEM (75 bits)
    - Swap / OUT / Call / WB (control signals) -> 1 bit
    - SP / M (control signals) -> 2 bits
    - EX/MEM -> 67 bits
        - Rd Address -> 3bits
        - ALUData + WriteData -> 2*32bits
4. MEM/WB (69 bits)
    - WB / OUT (control signals) -> 1bit
    - MEM / WB -> 67 bits
        - Rd Address -> 3bits
        - ALUData + WriteData -> 2*32bits

Non-Pipeline Registers:

1. Flags | Prev Pc+1 (21 bits)
    - Flags -> 3 bits
    - Prev PC+1 -> 18 bits
2. SP/PC (18 bits)
3. NZC (flags) (3 bits)

Hazards

- *Instruction fetching while another instruction using memory:*

  The Hazard Detection Unit will get the MemRead/MemWrite signal from EX/MEM register, and it will set the PC Stall, set the IF/ID.Flush and that will lead to a NOP then fetching the right instruction.
  (Stall PC & Flush IF/ID)

- *Immediate fetching while another instruction using memory:*

  The Hazard Detection Unit will get the MemRead/MemWrite signal from EX/MEM register and the IMM.Flush of the currently decoded instruction then it will set the PC Stall, set IF/ID.Stall, set the ID/EX.Flush and that so after PC will take the right immediate value.
  (Stall PC & Stall IF/ID & Flush ID/EX)

- *Instruction Decoding while another instruction in WB:*

  The Register File will write in Falling Edge and read in Rising Edge.

- *Swap:*

  The Hazard Detection Unit gets Control Unit Swap, EX/MEM.Swap to process the swap instruction that will take 3 cycles.
  (Stall PC & Stall IF/ID for 2 cycles)

- *Forwarding:*

  The Forward unit takes EX/MEM.WB the Rs1, Rs2 from ID/EX and Rd from EX/MEM to perform ALU to ALU forwarding when Rs1 or Rs2 = EX/MEM.Rd.
  It also takes MEM/WB.WB, and MEM/WB.Rd to perform Memory to ALU forwarding when RS1 or Rs2 = MEM/WB.Rd.
  (Forwarding to ALU or StoreData)

- *Load-Use:*

  The Hazard Detection Unit gets IF/ID.Rs1, IF/ID.Rs2, ID/EX.Rd, MemRead, SP+ve and when Rs1 or Rs2 = Rd and MemRead or SP+ve = 1, it sets IF/ID.Stall and PC-Stall and ID/EX.Flush.

  (Stall PC & Stall IF/ID & Flush ID/EX)

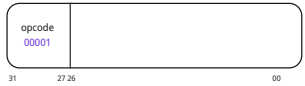
Changes In Design

- Changed Flag Enable to 3 bits (one bit for each flag) instead of 1 bit.
- Added FlagReset 3 bits (one bit for each flag)
- Changed WriteBack to 2 bits (MemToReg & RegWrite)
- Added ID/EX.SP to the Hazard Detection Unit

Type 1

NOP

HLT



HLT, IMMFlush

SETC

FlagEnable , Carry

NOT Rdst

Reg write , INC/NOT ,Buff,
ALUSignal(NOT=011)

INC Rdst

Reg write, INC/NOT ,imm,
ALUSignal(ADD=000)

OUT Rdst

OUT, Buffer,
ALUSignal(ADD=000)

IN Rdst

Reg write, I N, Buffer,
ALUSignal(ADD=000)

Type 2

MOV Rs,Rd

Reg write =1 , Buffer =1 ,imm=0 ,memToReg=0
ALUSignal(ADD=000)

SWAP Rs,Rd

Reg write =1 for 3 cycles, swap=1 for 3 cycles , op stall=1 for 2 cycles, Buffer =0 ,imm=0,memToReg=0
ALUSignal(XOR=100)

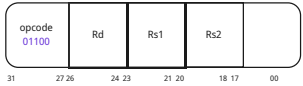
ADD Rd,Rs1,Rs2

Reg write =1 , Buffer =0 ,imm=0,memToReg=0, FlagEnable=1
ALUSignal(ADD=000)

SUB Rd,Rs1,Rs2

Reg write =1 , Buffer =0 ,imm=0,memToReg=0 , FlagEnable=1
ALUSignal(SUB=001)

AND Rd,Rs1,Rs2



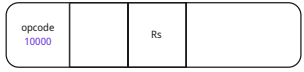
Reg write =1 , Buffer =0 ,imm=0,memToReg=0 , FlagEnable=1
ALUSignal(AND=010)

IADD Rd,Rs,Imm

Reg write =1 , Buffer =0 ,imm=1 ,memToReg=0
,IMM-Flush=1,Inc=0, FlagEnable =1
ALUSignal(ADD=000)

Type 3

PUSH Rs

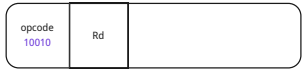


StackPointer-ve, Buffer
ALUSignal(ADD: 00)

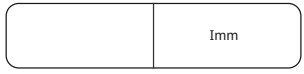
POP Rd

RegWrite, StackPointer+ve, MemToReg

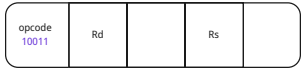
LDM Rd, Imm



IMMFlush,IMM,Buff,RegWrite
ALUSignal(ADD: 00)



LDD Rd, Offset(Rs)



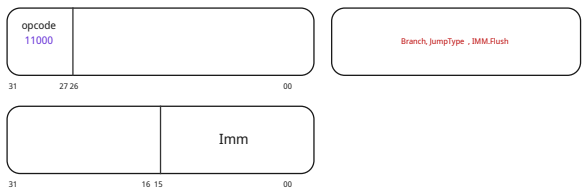
IMMFlush,IMM , MemRead , MemToReg , RegWrite
ALUSignal(ADD: 00)

STD Rs1, Offset(Rs2)

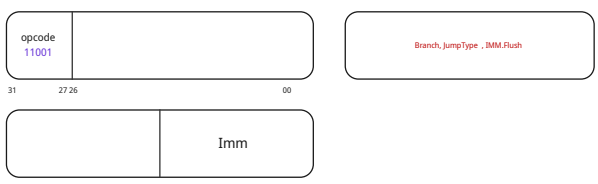
IMMFlush,IMM, MemWrite
ALUSignal(ADD: 00)

Type 4

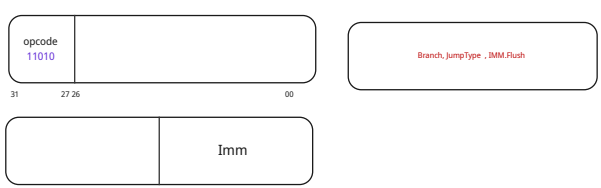
JZ Imm



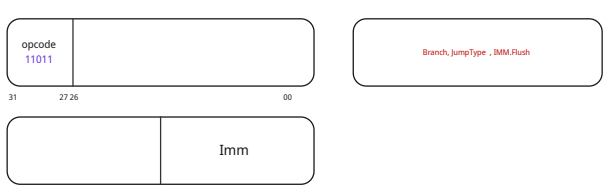
JN Imm



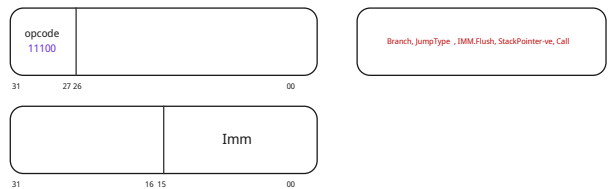
JC Imm



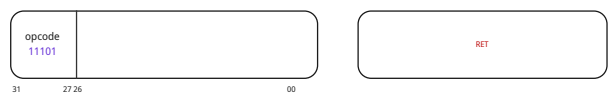
JMP Imm



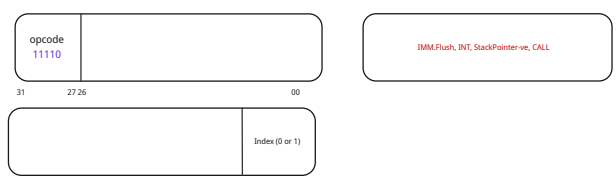
CALL Imm



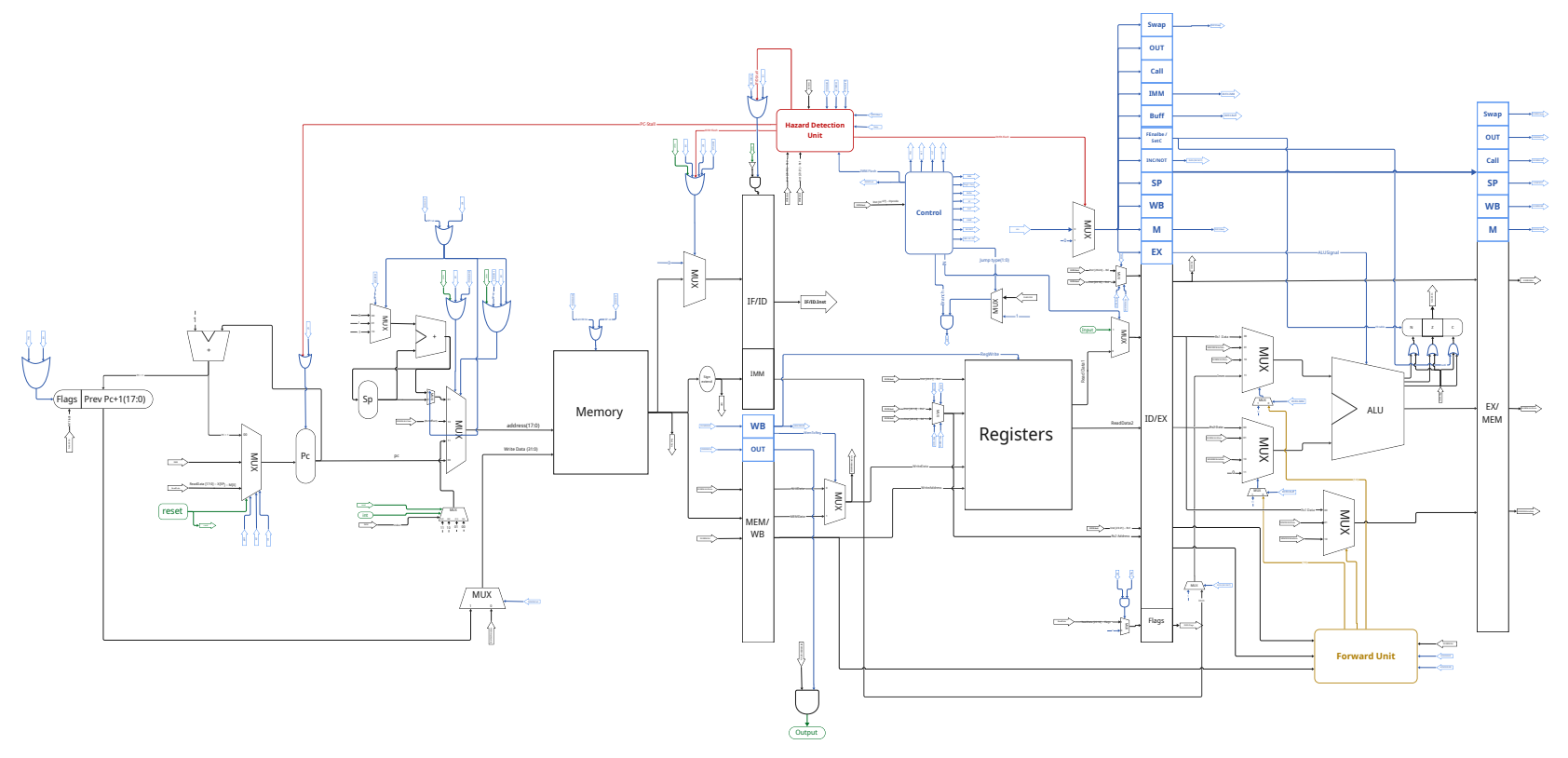
RET

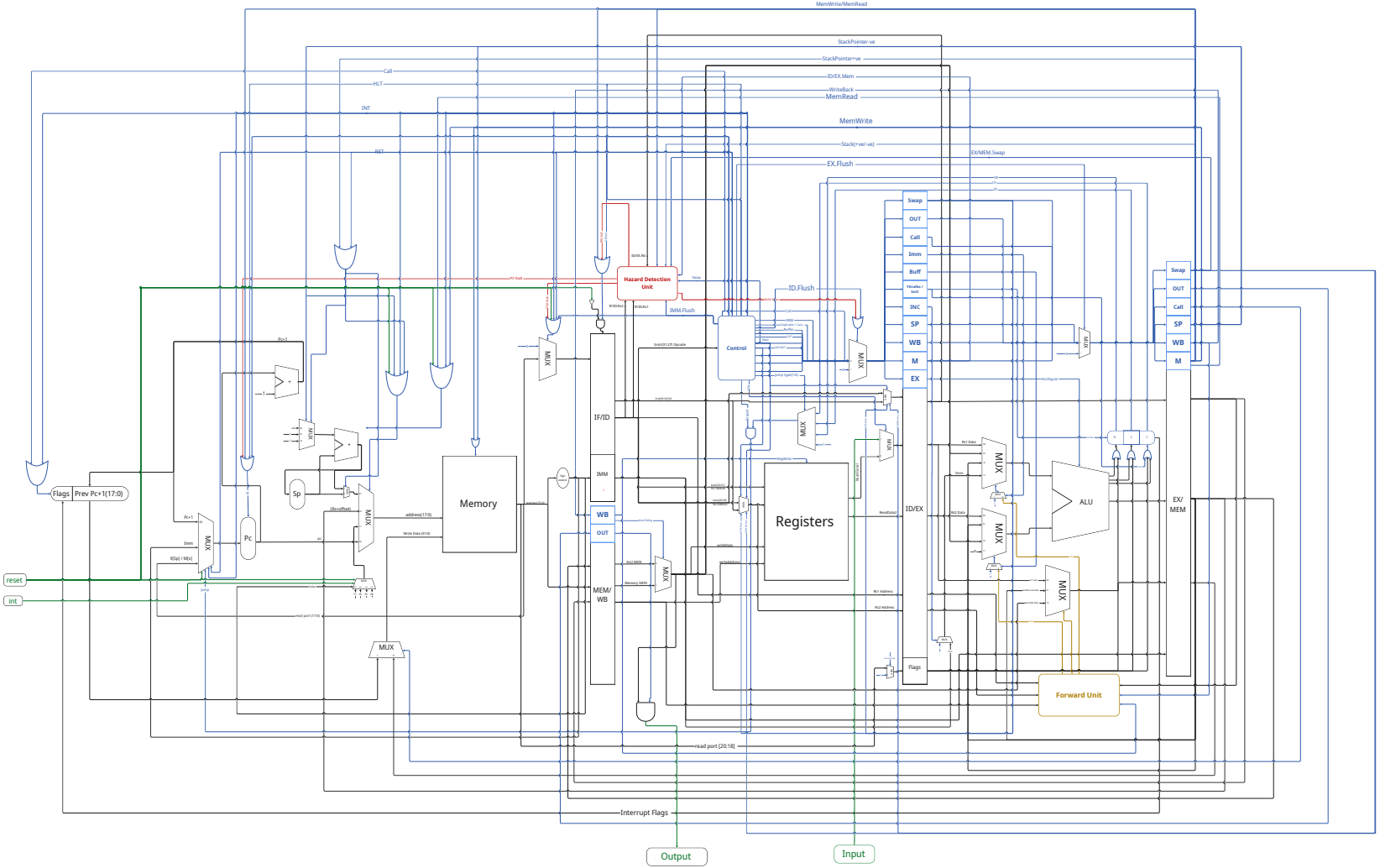


INT index



RTI


Hazard Detection Unit
Control
Memory
Registers
ALU
Forward Unit
IF/ID
IMM
WB
OUT
MEM/ WB
ID/EX
EX/ MEM
Swap
OUT
Call
IMM
Buff
INC/NOT
SP
WB
M
EX
Flags
Prev Pc+1(17:0)
Pc
Sp
reset
MUX
Output


MemWrite/MemRead
StackPointer-ve
StackPointer+ve
ID/EX.Mem
WriteBack
MemRead
MemWrite
Stack(+ve/-ve)
EX.Flush
EX/MEM.Swap
Call
HLT
INT
RET
Hazard Detection Unit
ID.Flush
IMM.Flush
Control
Swap
OUT
Call
Imm
Buff
INC
SP
WB
M
EX
IF/ID
IMM
WB
OUT
MEM/ WB
ID/EX
EX/ MEM
Registers
ALU
Memory
Sp
Pc
Flags
Prev Pc+1(17:0)
reset
int
Forward Unit
Interrupt Flags
Output
Input