# PIPELINE IMPLEMENTATION OF A PROCESSOR

## RISC Processor (MPIS 32)

- 32-bit General Purpose Registers (GPRs), **R0** to **R31**
  - ❖ **R0** is constant; cannot be written.
- Special Purpose 32-bit register, **Program Counter (PC)**
- Memory Word size is 32-bits

## Instruction Sets:

**1. Load and Store**

```
LW    R2, 124(R8)       //R2 = Mem[R8+124]

SW    R5, -10(R25)     //Mem [R25 – 10] = R5
```

**2. Arithmetic and Logic Instructions** (Only Register Operands)

```
ADD    R1, R2, R3       //R1 = R2 + R3
ADD    R1, R2, R0       //R1 = R2 + 0 (Moving)
SUB, AND, OR, MUL, SLT (Set Less Than)
SLT       R5, R11, R12   //IF R11 < R12, R5 = 1; else R5 = 0
```

**3. Arithmetic and Logic Instructions** (Immediate Operand)

```
ADDI   R1, R2, 25        //R1 = R2 + 25
SUBI, SLTI
```

**4. Branch Instructions**

BEQZ R1, Loop //Branch to loop if R1 = 0

BNEQZ R1, Label //Branch to label if R5! = 0

**5. JUMP Instruction**

J Loop //Branch to Loop unconditionally

**6. Miscellaneous Instructions**

HLT //Halt execution

| Instructions | Op-Code |
|---|---|
| ADD | 000000 |
| SUB | 000001 |
| AND | 000010 |
| OR | 000011 |
| SLT | 000100 |
| MUL | 000101 |
| HLT | 111111 |

## R-Type Instruction Encoding

| 31 26 | 25 21 | 20 16 | 15 11 | 10 6 | 5 0 |
|---|---|---|---|---|---|
| opcode | rs | rt | rd | 0 | 0 |

## Example

**SUB R5, R12, R25**

| | | | | |
|---|---|---|---|---|
| 000001 | 01100 | 11001 | 00101 | 00000 000000 |
| SUB | R12 | R25 | R5 | Unused |

= 05992800 (in HEX)

# I-Type Instruction Encoding

| 31 26 | 25 21 | 20 16 | 15 0 |
|---|---|---|---|
| **opcode** | **rs** | **rt** | **Immediate Data** |

## Examples

**LW R20, 84(R9)**

| | | | |
|---|---|---|---|
| 001000 | 01001 | 10100 | 0000000001010100 |
| LW | R9 | R20 | Offset |

= 21340054 (in HEX)

**BEQZ R25, Label**

| | | | |
|---|---|---|---|
| 001110 | 11001 | 00000 | yyyyyyyyyyyyyyyy |
| LW | R25 | unused | Offset |

= 3b20YYYY (in HEX)

| Instructions | Op-Code |
|---|---|
| LW | 001000 |
| SW | 001001 |
| ADDI | 001010 |
| SUBI | 001011 |
| SLTI | 001100 |
| BNEQZ | 001101 |
| BEQZ | 001110 |

## J-Type Instruction Encoding

| 31 26 | 25 0 |
|---|---|
| opcode | Immediate Data |

| Instruction | Op-Code |
|---|---|
| J | 010000 |

**//J-type Instructions are not Implemented Here**

# MPIS 32 Instruction Cycle

I. IF : Instruction Fetch

II. ID : Instruction Decode / Resistor Fetch

III. EX : Execution/Effective Address Calculation

IV. MEM : Memory Access/Branch Completion

V. WB : Register Write-back

## ADD R2, R5, R10

| | | |
|---|---|---|
| **IF** | IR | ← Mem [PC] ; |
| | NPC | ← PC + 1 ; |
| **ID** | A | ← Reg[rs] ; |
| | B | ← Reg[rt] ; |
| **EX** | ALUOut | ← A + B ; |
| **MEM** | PC | ← NPC ; |
| **WB** | Reg[rd] | ← ALUOut ; |

## ADDI R2, R5, 110

| | | |
|---|---|---|
| **IF** | IR | ← Mem [PC] ; |
| | NPC | ← PC + 1 ; |
| **ID** | A | ← Reg[rs] ; |
| | Imm | ← $(IR_{15})^{16}$ ## $IR_{15..0}$ |
| **EX** | ALUOut | ← A + Imm ; |
| **MEM** | PC | ← NPC ; |
| **WB** | Reg[rt] | ← ALUOut ; |

## LW R2, 200(R6)

| | | | |
|---|---|---|---|
| IF | IR | ← | Mem [PC] ; |
| | NPC | ← | PC + 1 ; |
| ID | A | ← | Reg[rs] ; |
| | Imm | ← | $(IR_{15})^{16}$ ## $IR_{15..0}$ |
| EX | ALUOut | ← | A + Imm; |
| MEM | PC | ← | NPC ; |
| | LMD | ← | Mem[ALUOut] ; |
| WB | Reg[rt] | ← | LMD ; |

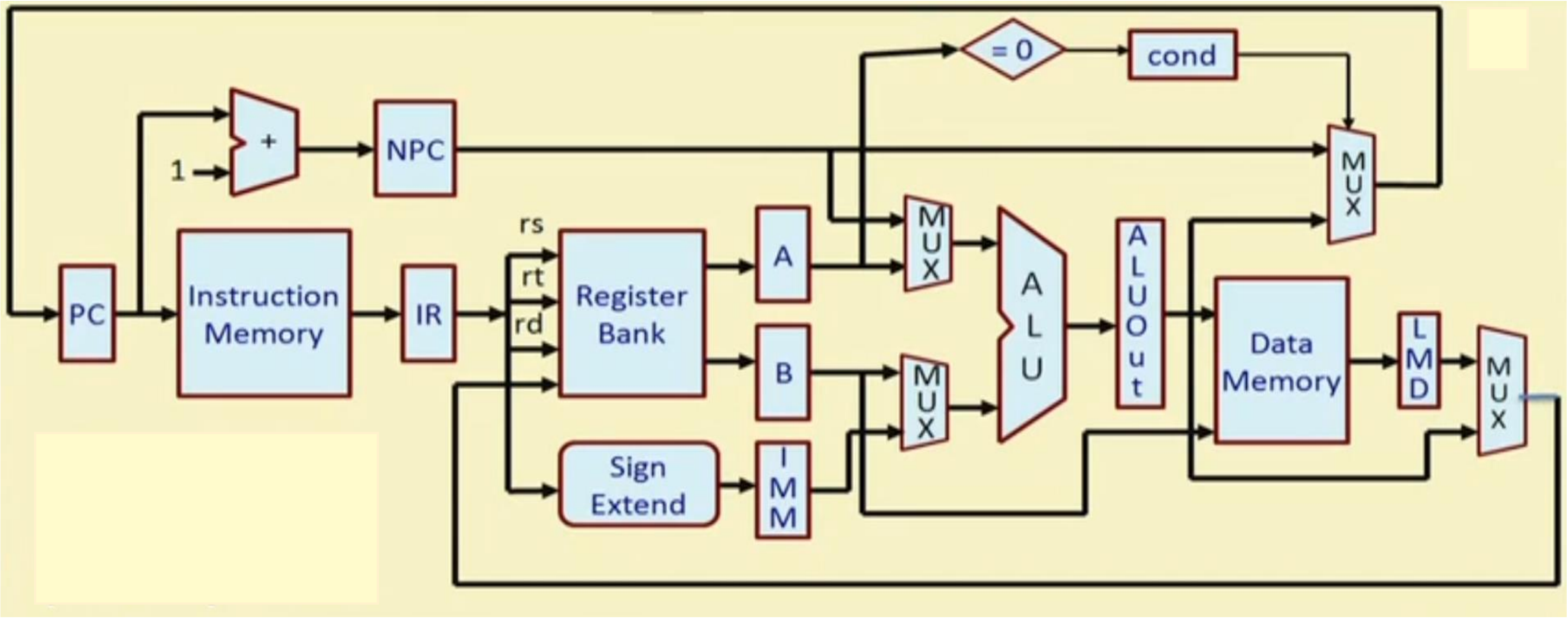
## SW R2, R5, 110

| | | | |
|---|---|---|---|
| IF | IR | ← | Mem [PC] ; |
| | NPC | ← | PC + 1 ; |
| ID | A | ← | Reg[rs] ; |
| | B | ← | Reg[rt] ; |
| | Imm | ← | $(IR_{15})^{16}$ ## $IR_{15..0}$ |
| EX | ALUOut | ← | A + Imm ; |
| MEM | PC | ← | NPC ; |
| | Mem[ALUOut] | ← | B ; |
| WB | | ------ | |

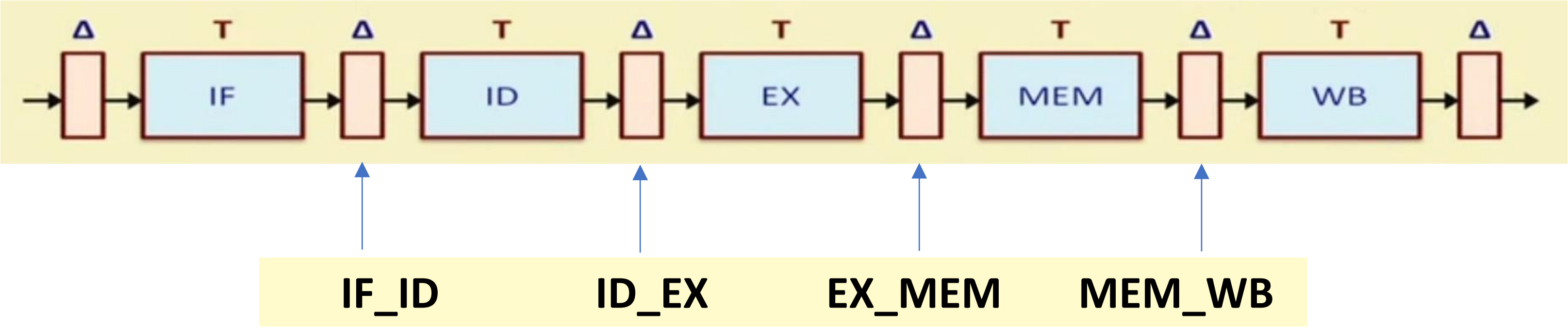
# BEQZ  R3, Label

| | | | |
|---|---|---|---|
| IF | IR | ← | Mem [PC] ; |
| | NPC | ← | PC + 1 ; |
| ID | A | ← | Reg[rs] ; |
| | Imm | ← | $(IR_{15})^{16}$ ## $IR_{15..0}$ |
| EX | ALUOut | ← | NPC + Imm; |
| | cond | ← | (A = = 0) ; |
| MEM | PC | ← | NPC ; |
| | If(cond) PC | ← | ALUOut ; |
| WB | | | ------ |

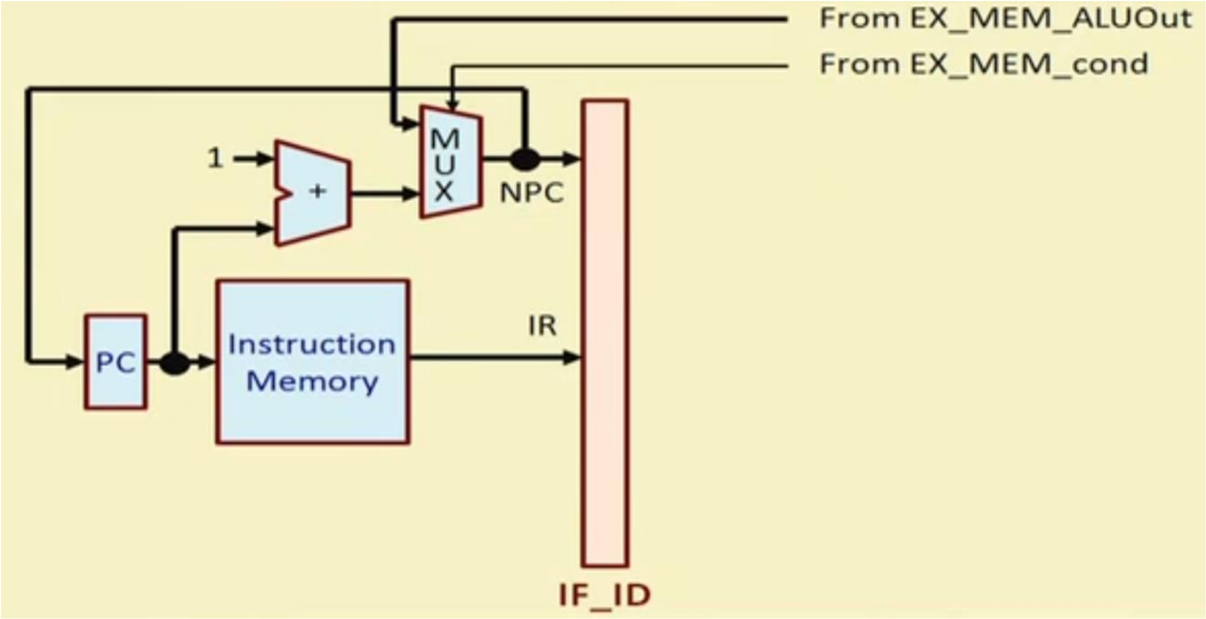
# Data Path (MIPS32 Non-Pipelined Design)



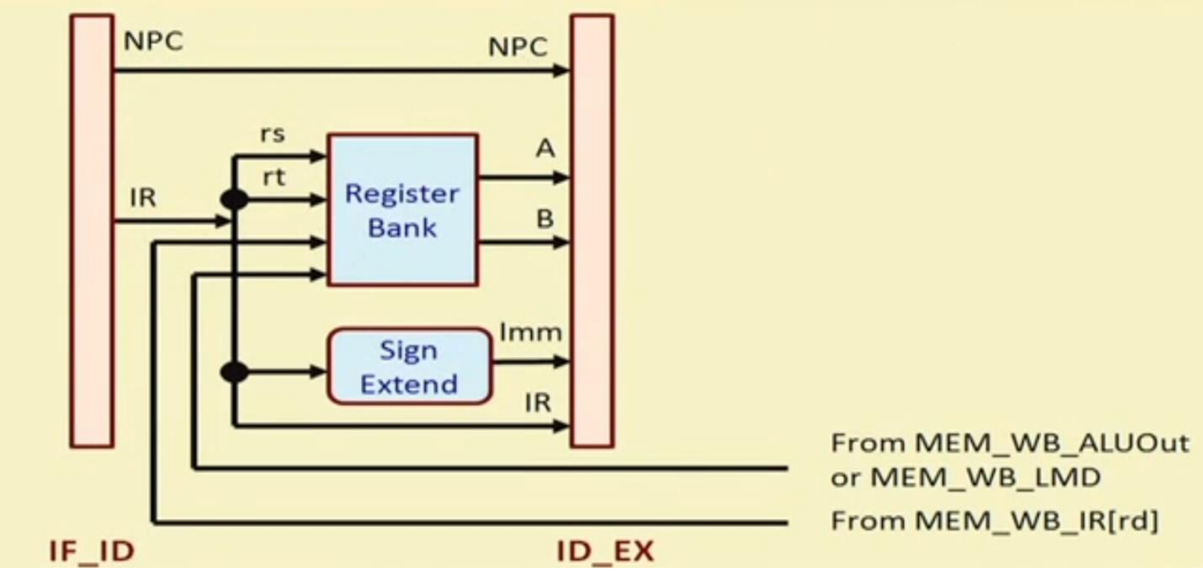
# Data Path (MIPS32 Pipelined Design)



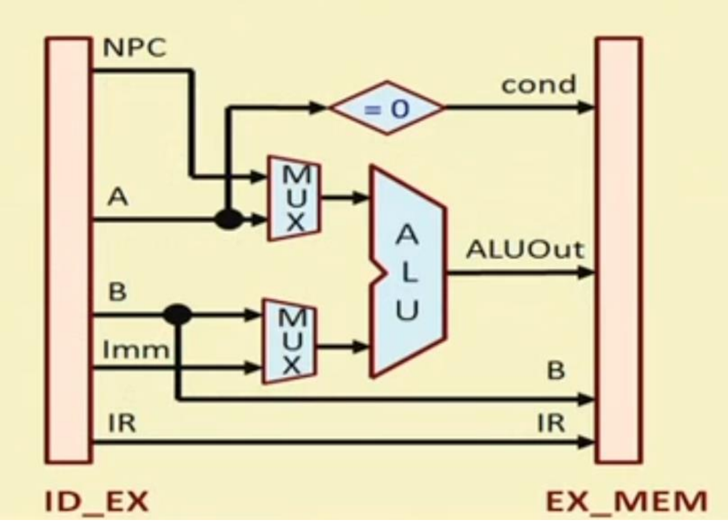
## IF Stage



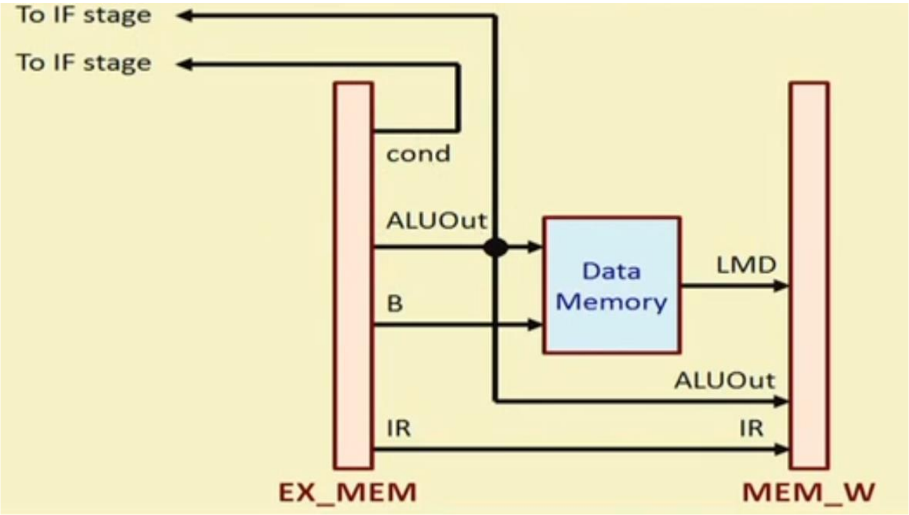
# ID Stage



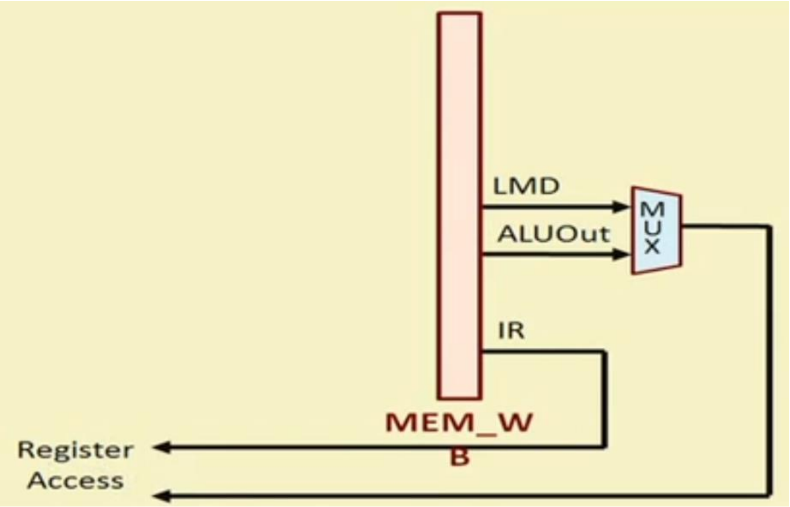
# EX Stage



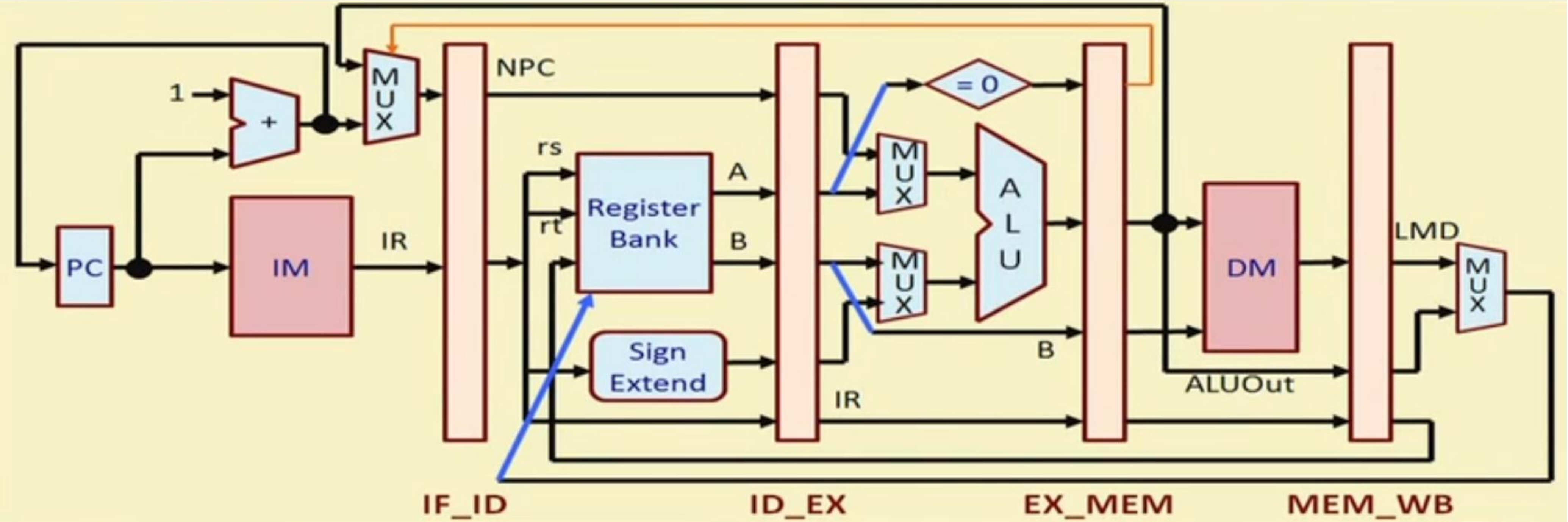
# MEM Stage



# WB Stage



# Data Path (MIPS32 Pipelined Design)


1
+
MUX
NPC
PC
IM
IR
rs
rt
Register Bank
A
B
Sign Extend
IR
MUX
MUX
= 0
A L U
B
DM
ALUOut
LMD
MUX
IF_ID
ID_EX
EX_MEM
MEM_WB