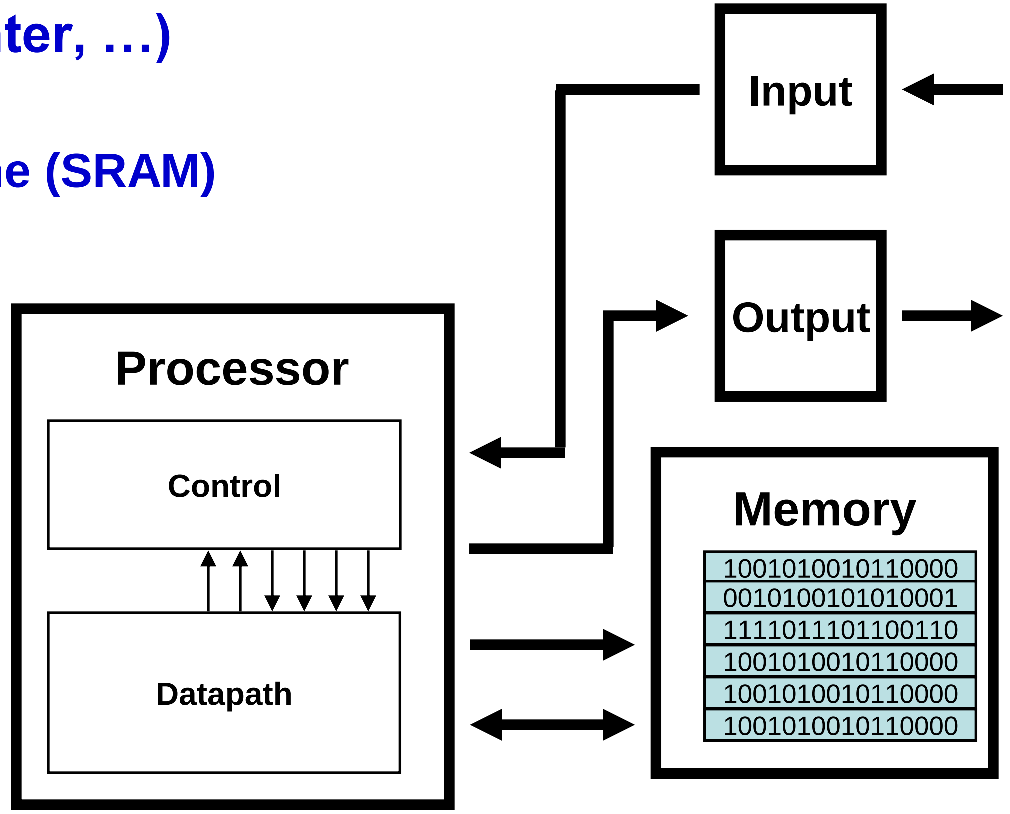
# The Five Classic Components of a Computer

- Input (mouse, keyboard, …)
- Output (display, printer, …)
- Memory
  - main (DRAM), cache (SRAM)
  - secondary (disk, CD, DVD, …)
- Datapath
- Control

} Processor (CPU)



## CPU Operations

Fetch a word from Memory

Store a word into memory

Reg Transfers

Performing an ALU function

## A CISC PROCESSOR

Larger instructions with variable formats
(16-64 bits/ instruction)

Larger Addressing Modes (12- 24)

Few Registers

Most Microcoded with control Memory

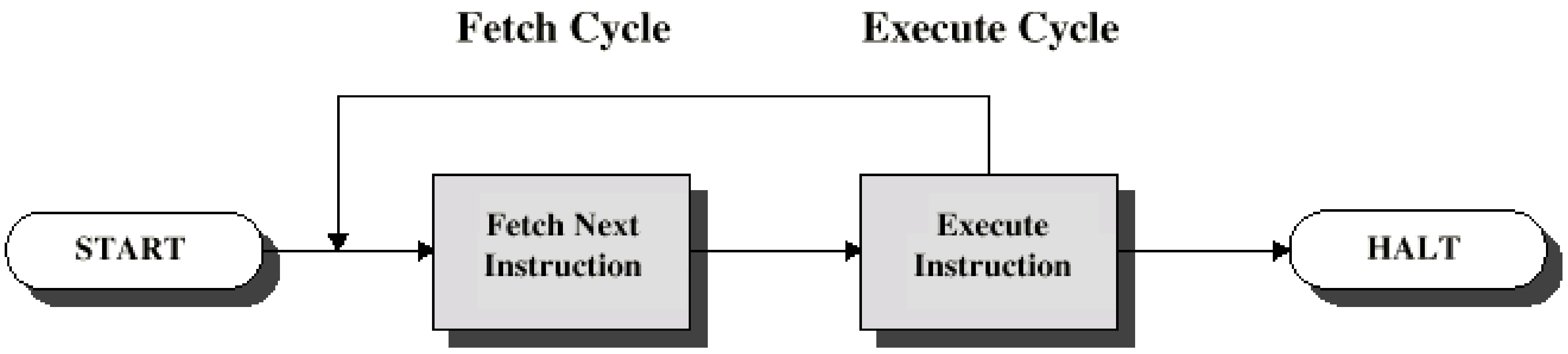
Close to high level language

## Reduced Instruction Set Computer

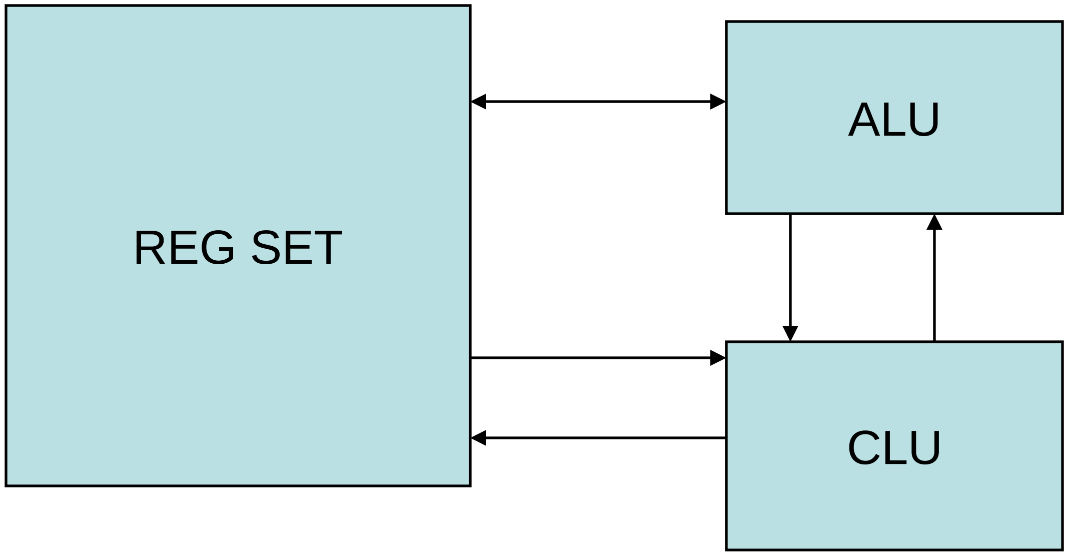
LOAD- STORE Architecture
Fewer Addressing Modes
Fixed Length Instructions
More Registers
Designed for Pipeline Efficiency
Hardwired Control Unit

# Instruction Cycle

- Two steps:
  - Fetch
  - Execute

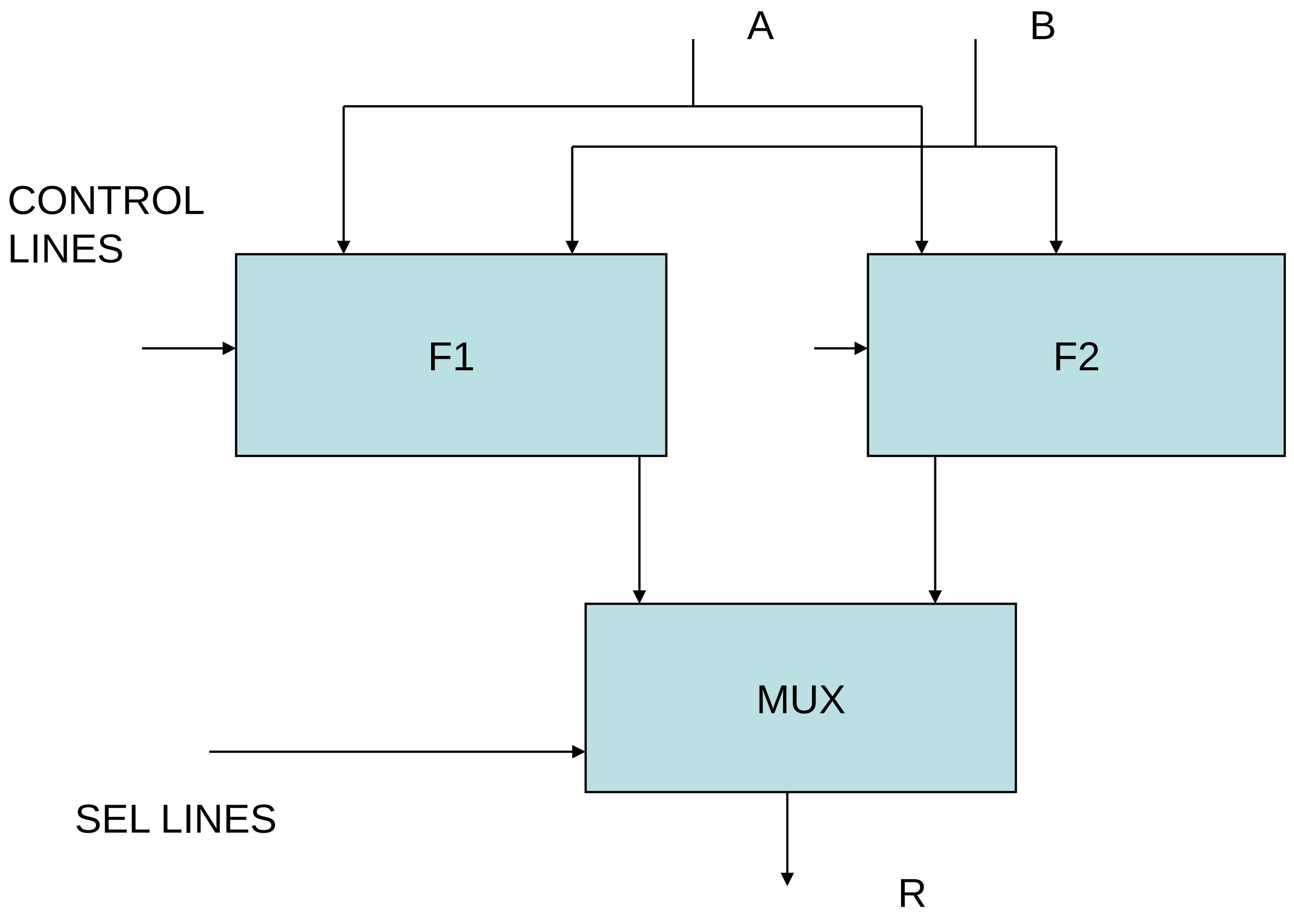


## Functional Blocks in a Processor



# CENTRAL PROCESSING UNIT

## ARITHMETIC LOGIC UNIT



## Instruction set Summary

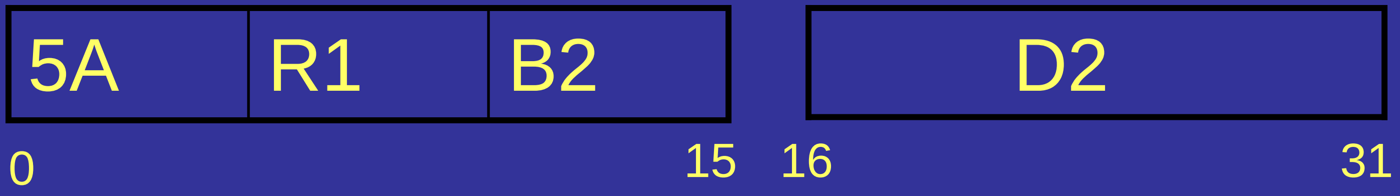
- Instruction Formats
- Operations
- Addressing Modes
- Programmers Registers

## A TYPICAL INSTRUCTION

- **ADD  R1, D2(B2)**

**(R1, B2 - Registers ), D2- displacement**

**(R1) + (Memory) $\rightarrow$ (R1) ; (B2) + D2 $\rightarrow$ Memory**



## Steps in Execution of an Instruction

CPU fetches instruction from Main Memory

CPU Decodes the Instruction Op-code

Depending on Op-code
- Fetches another operand
- Execute instruction via register to register transfer
- Write the results in M
- Write the results in I/O

Repeat steps

## Steps for executing instruction

- Fetch the first instruction half-word

- Find ADD control sequence

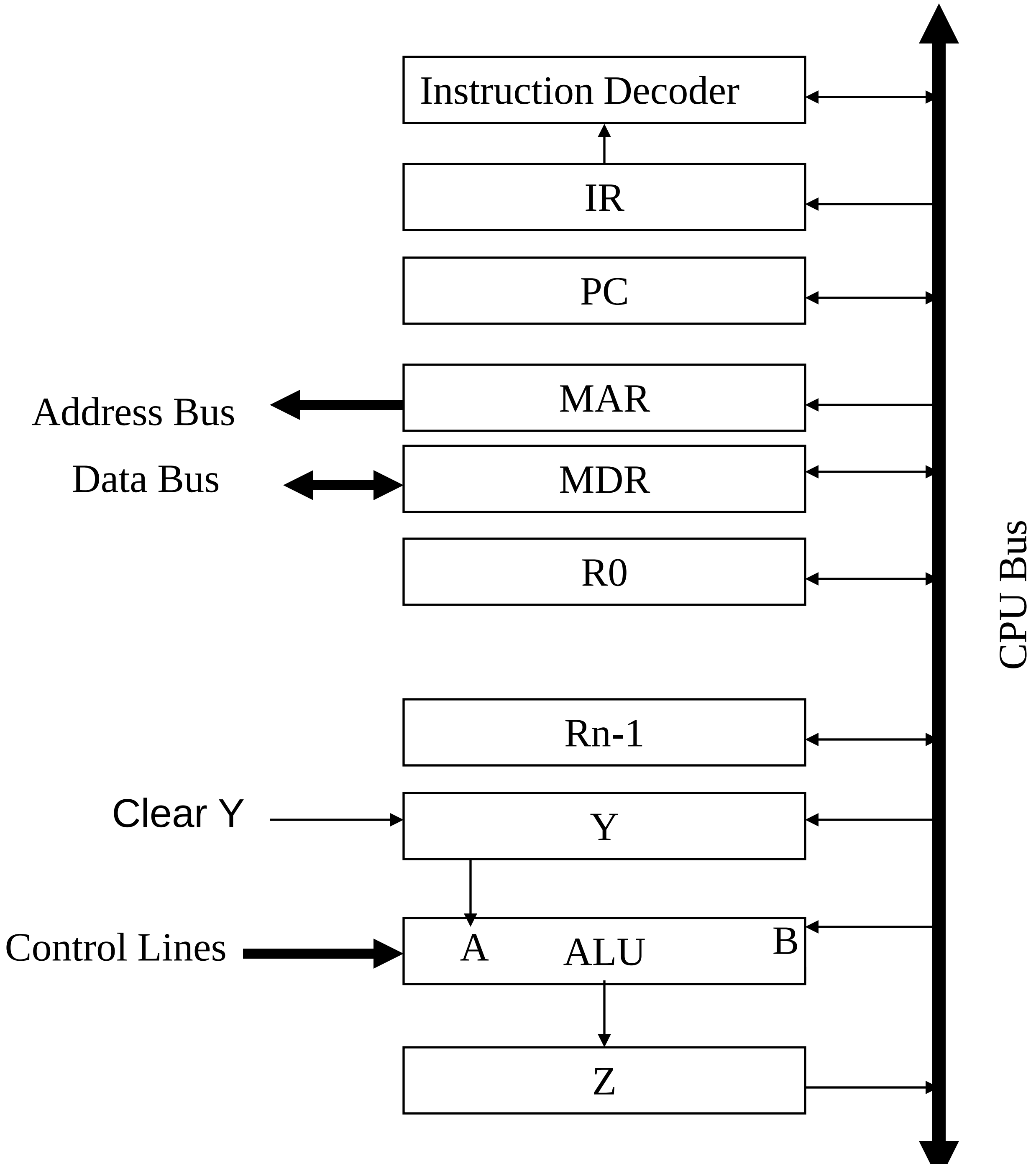
- Fetch the remaining instruction half-word

- Calculate the operand address

- Fetch the operand

- Add

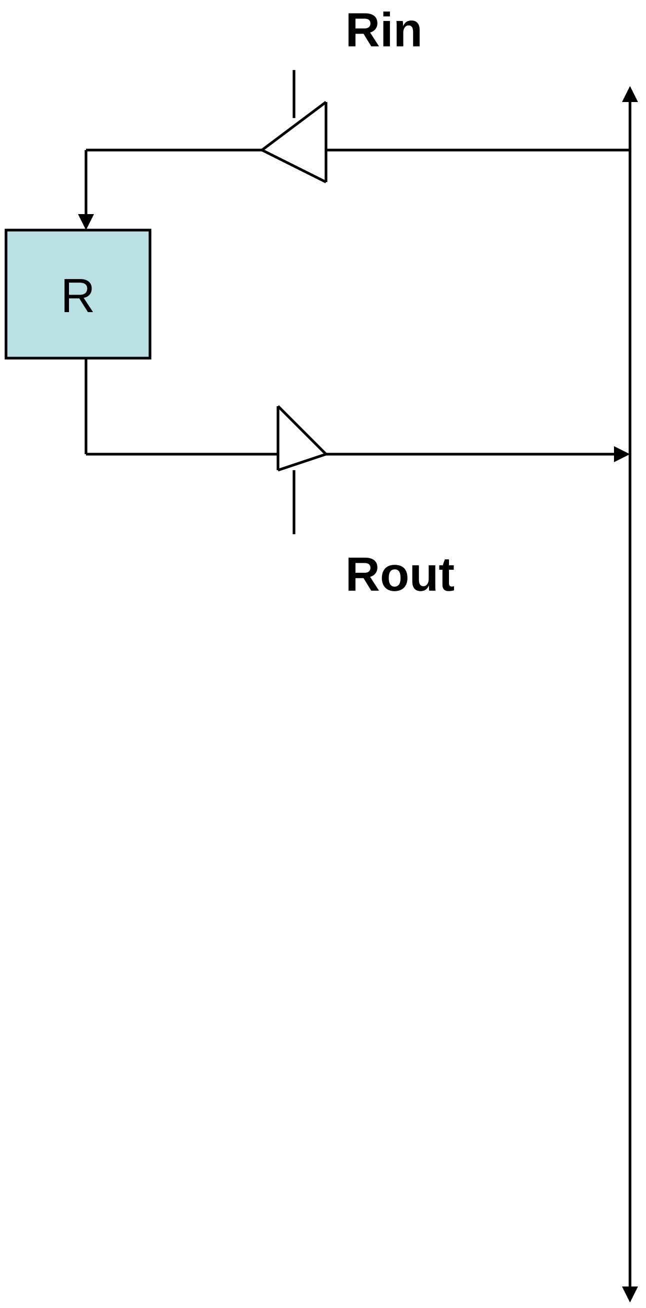
- Store the result

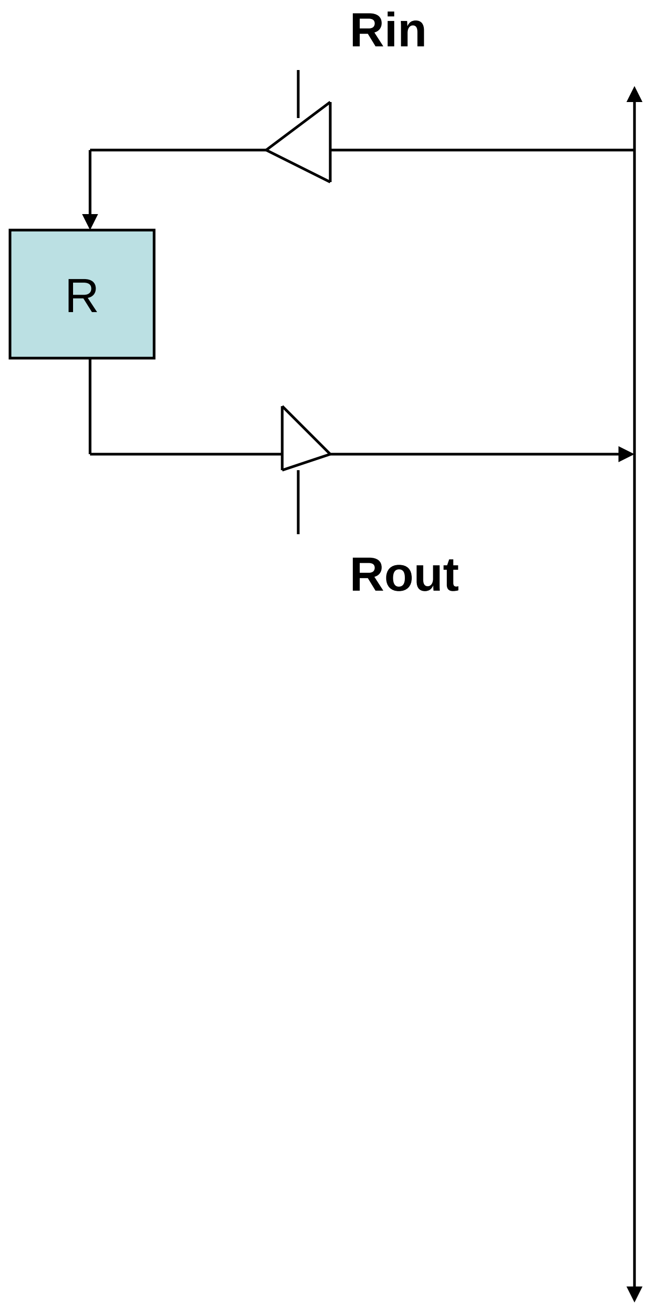

Instruction Decoder
IR
PC
Address Bus
MAR
Data Bus
MDR
R0
Rn-1
Clear Y
Y
Control Lines
A
ALU
B
Z
CPU Bus

## Register Transfer:

**$R2 \leftarrow R1$**

To enable data transfer between various
Blocks connected to common bus provide
Input output gating.


Rin
R
Rout



Control Signals for

$R1 \longleftarrow R2$

$R_{2OUT}$ , $R_{1in}$

## Example Microinstructions:

Open/.Close a gate from Reg to a bus

Transfer data along a bus

Send timing signals

Test bits within a register

## Fetching a word from memory:

i. MAR $\leftarrow$ (R1)

ii. Read Signal

iii. Wait for Memory-function-complete (MFC) signal

iv. R2 $\leftarrow$(MDR)

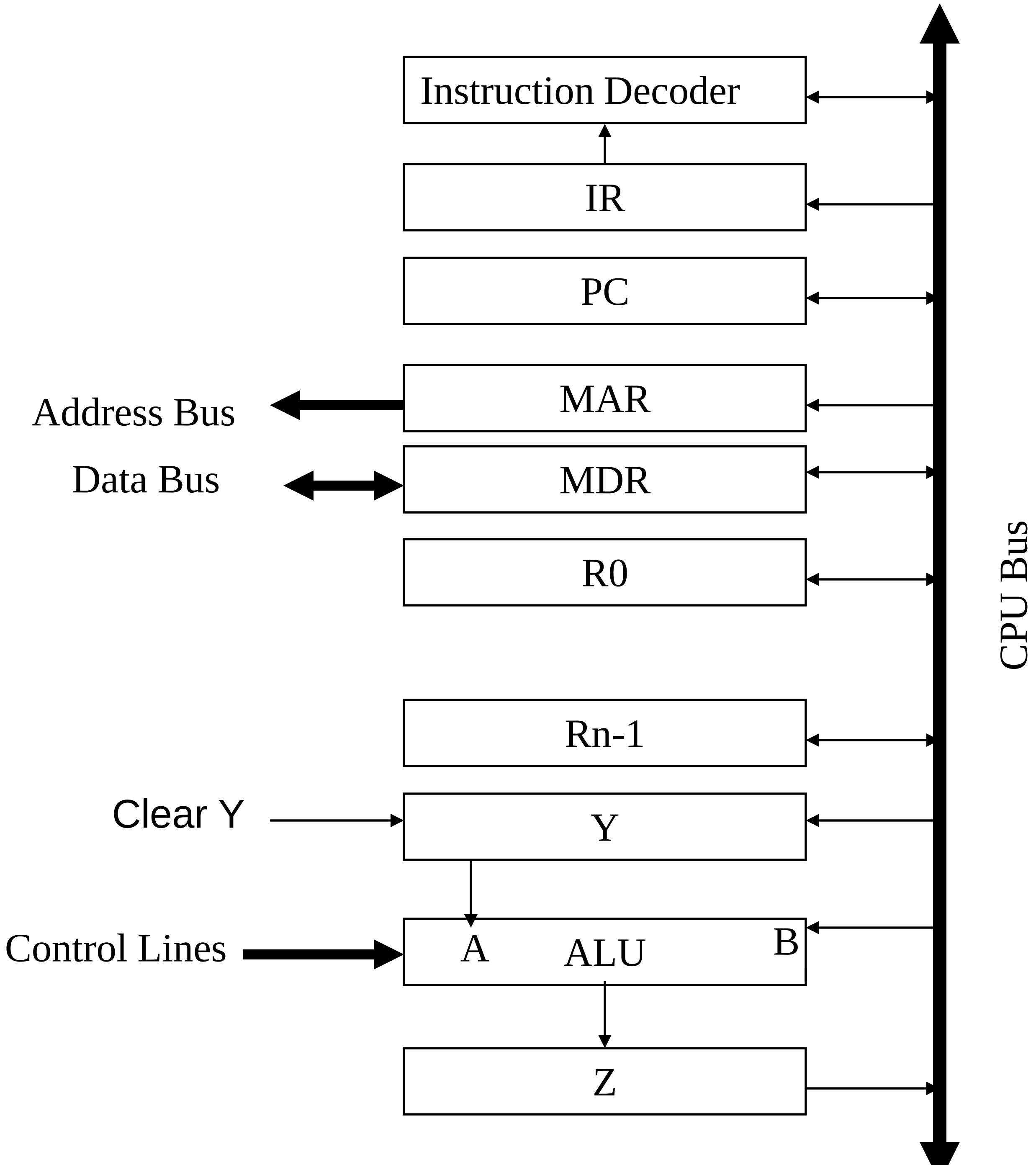
## Storing a word into Memory:

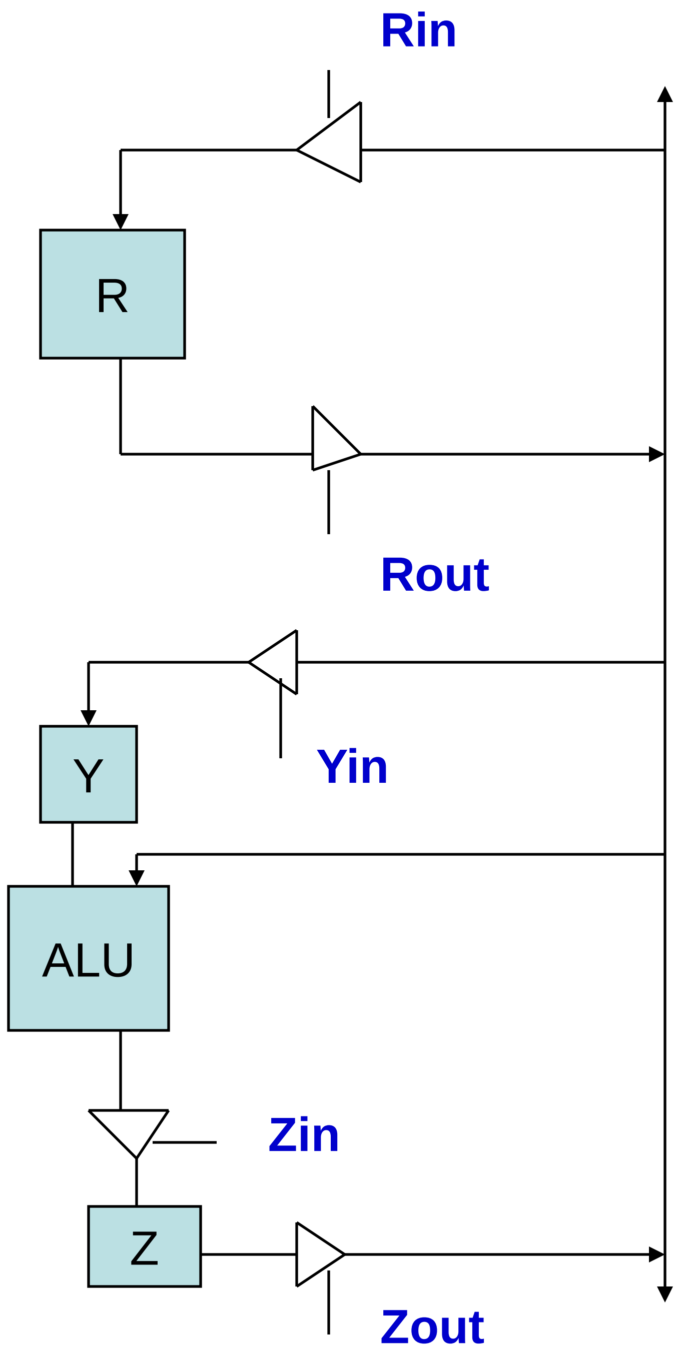
i. MAR $\leftarrow$(R1)

ii. MDR $\leftarrow$(R2)

iii. Memory write signal

iv. Wait for MFC


Instruction Decoder
IR
PC
Address Bus
MAR
Data Bus
MDR
R0
CPU Bus
Rn-1
Clear Y
Y
Control Lines
A
ALU
B
Z



Control Signals for

$R1 \longleftarrow R2$

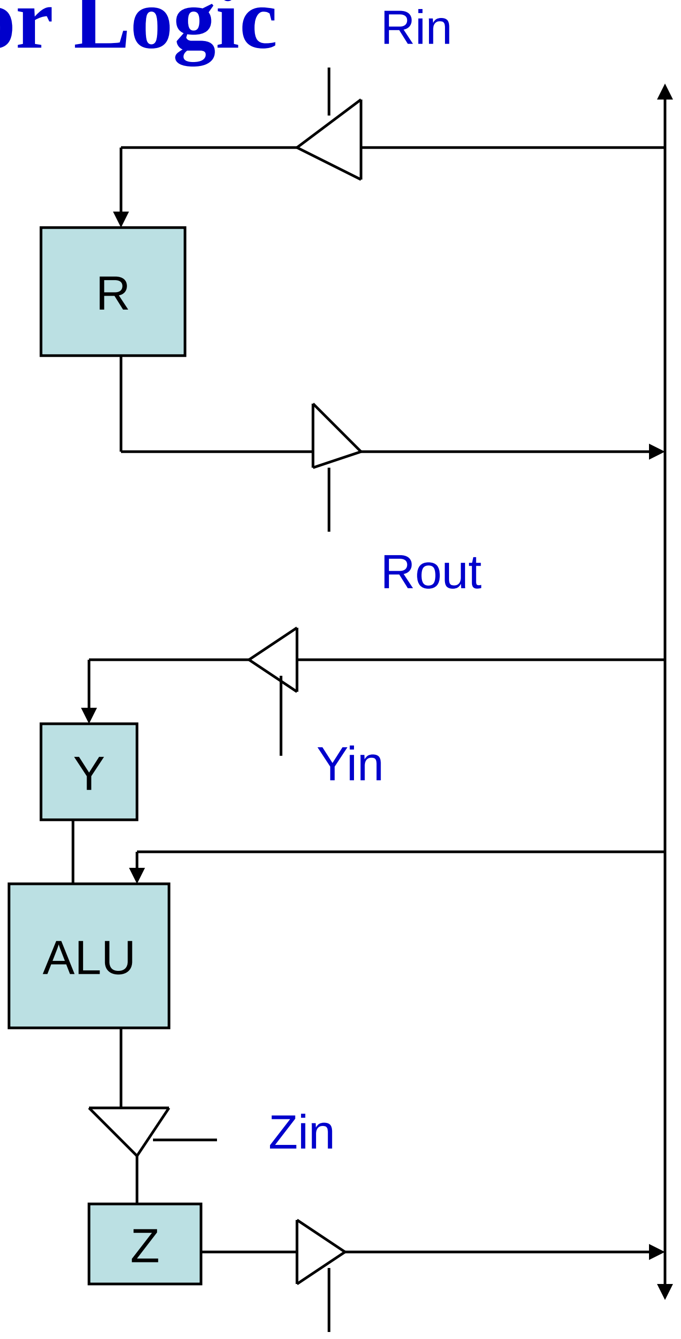
$R_{2OUT}$ , $R_{1in}$

## Performing an Arithmetic or Logic Operation:

i. $R1_{out}$, $Y_{in}$

ii. $R2_{out}$, Add, $Z_{in}$

iii. $Z_{out}$, $R3_{in}$



## Ex: Add contents of a memory location to register R1

(Address of Memory location is in part of the instruction )

## Ex: Add contents of a memory location to register R1

| Step | RTL | Control Sequence |
|---|---|---|
| T1 | MAR$\leftarrow$PC; PC $\leftarrow$PC+1 | PC$_{out}$, MAR$_{in}$, Clear Y, Set Carry$_{in}$ of ALU, ADD, Z$_{in}$, READ |

## Ex: Add contents of a memory location to register R1

| Step | RTL | Control Sequence |
|---|---|---|
| T1 | MAR$\leftarrow$PC; PC $\leftarrow$PC+1 | $PC_{out}$, $MAR_{in}$, Clear Y, Set $Carry_{in}$ of ALU, ADD, $Z_{in}$, READ |
| T2 | Wait | $Z_{out}$, $PC_{in}$, Wait for MFC |
| T3 | IR $\leftarrow$MDR | $MDR_{out}$, $IR_{in}$ |

**- Instruction Fetch**

# Ex: Add contents of a memory location to register R1

| Step | RTL | Control Sequence |
|---|---|---|
| T4 | MAR $\leftarrow$IR | Addr-field of IR$_{out}$, MAR$_{in}$, READ |
| T5 | Y $\leftarrow$R1 | R1$_{out}$, Y$_{in}$, Wait for MFC |
| T6 | Z $\leftarrow$Y + M[MAR] | MDR$_{out}$, ADD, Z$_{in}$ |
| T7 | R1 $\leftarrow$Z | Z$_{out}$, R1$_{in}$, END |

## Ex: Branch by an offset x

| Step | RTL | Control Sequence |
|---|---|---|
| T1 | MAR←PC; PC ←PC+1 | $PC_{out}$, $MAR_{in}$, Clear Y, Set $Carry_{in}$ of ALU, ADD, $Z_{in}$, READ |
| T2 | Wait | $Z_{out}$, $PC_{in}$, Wait for MFC |
| T3 | IR ←MDR | $MDR_{out}$, $IR_{in}$ |
| T4 | Y ←PC | $PC_{out}$, $Y_{in}$ |
| T5 | Z ←Y + [x of IR] | ADD, $Z_{in}$ Addr-field of $IR_{out}$ |
| T6 | PC ←Z | $Z_{out}$, $PC_{in}$, END |