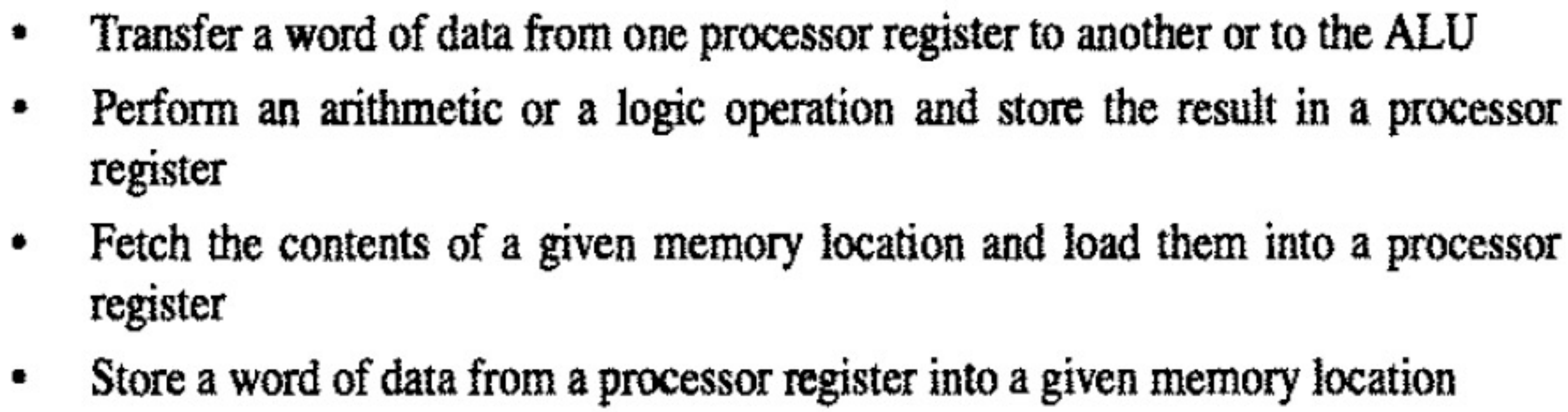
Lecture-15
(UNIT-II)

Typical operations involved during instruction execution



- Transfer a word of data from one processor register to another or to the ALU
- Perform an arithmetic or a logic operation and store the result in a processor register
- Fetch the contents of a given memory location and load them into a processor register
- Store a word of data from a processor register into a given memory location

Register Transfers

Example: Transferring the content of register of R_1 to the content of register of R_4

MOV R1,R4

(Input and output gating signals are used here)

- Enable the output of register R1 by setting $R1_{out}$ to 1. This places the contents of R1 on the processor bus.
- Enable the input of register R4 by setting $R4_{in}$ to 1. This loads data from the processor bus into register R4.

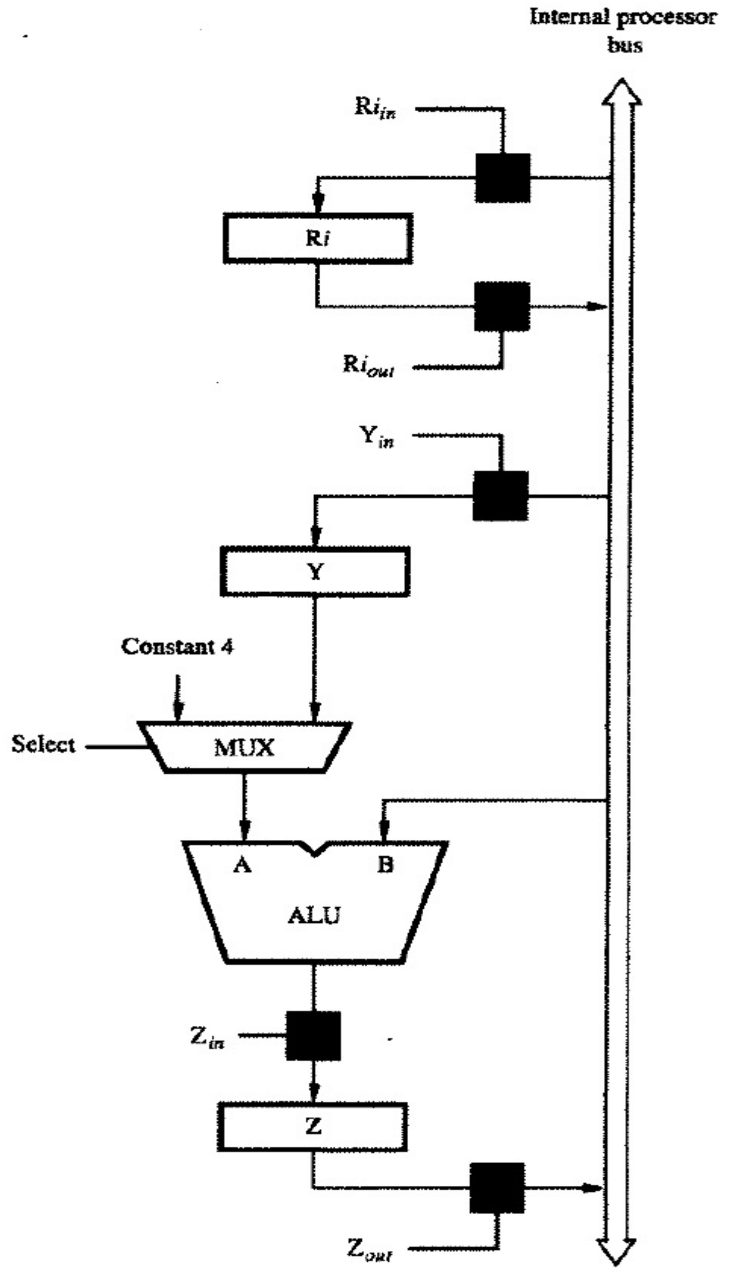
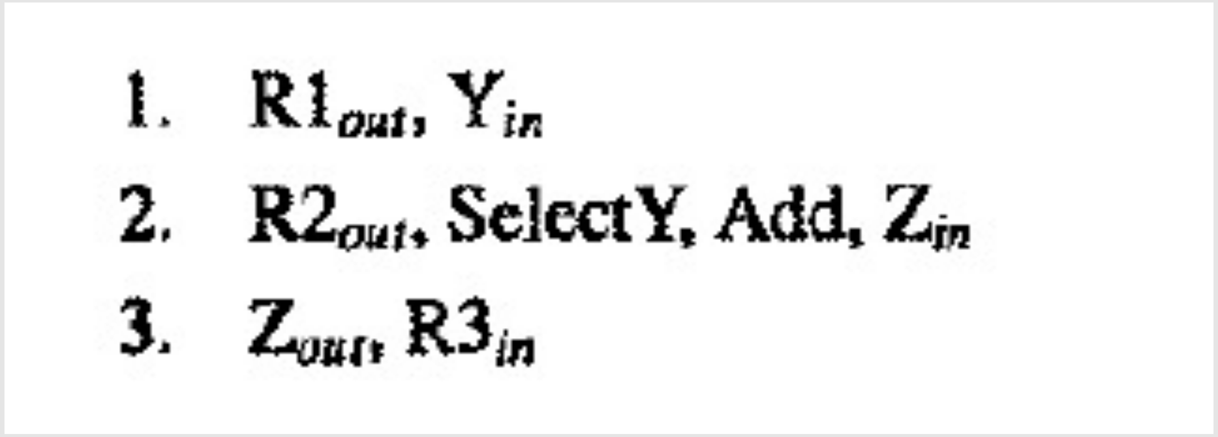


Figure 7.2 Input and output gating for the registers in Figure 7.1.

Performing and Arithmetic or Logic Operation



1. $R1_{out}$, Y_{in}
2. $R2_{out}$, SelectY, Add, Z_{in}
3. Z_{out}, $R3_{in}$

- Example: Add the content of registers R1 to R2 and store the result in register R3

Fetching a word from Memory

Example: Mov (R1), R2

Assumptions:
1. Output of MAR is available at all times
2. Each step can be completed in one clock cycle except Step 3
3. Control Signal Called MFC (Memory Function Complete) is used to take care of speed difference between Processor and Memory

1. MAR ← [R1]
2. Start a Read operation on the memory bus
3. Wait for the MFC response from the memory
4. Load MDR from the memory bus
5. R2 ← [MDR]

1. $R1_{out}$, MAR_{in}, Read
2. MDR_{inE}, WMFC
3. MDR_{out}, $R2_{in}$

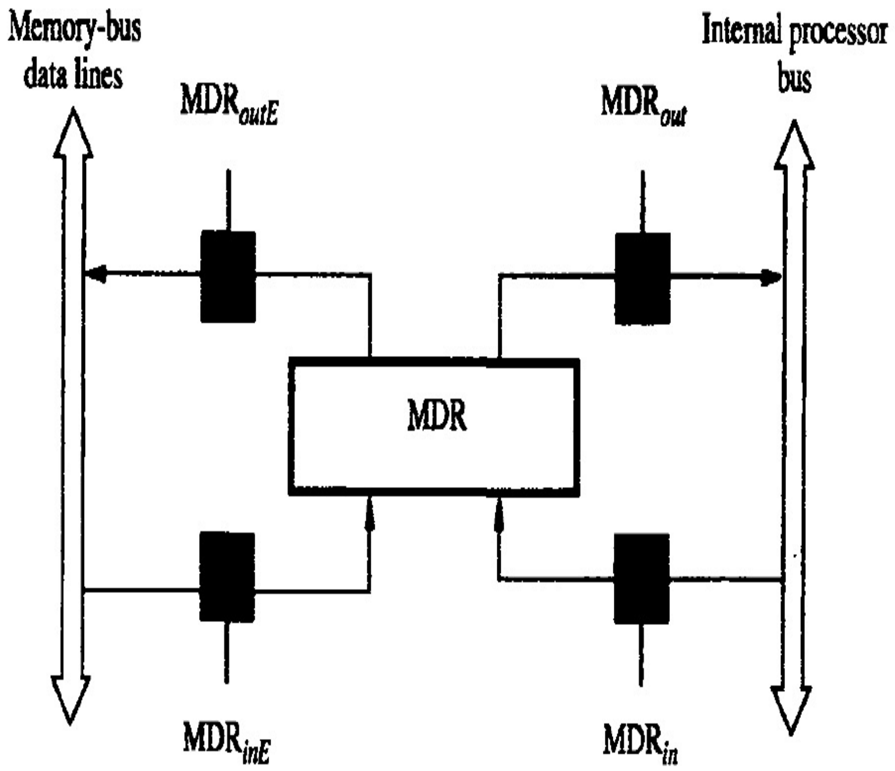


Figure 7.4 Connection and control signals for register MDR.

Timing of Memory Read Operation

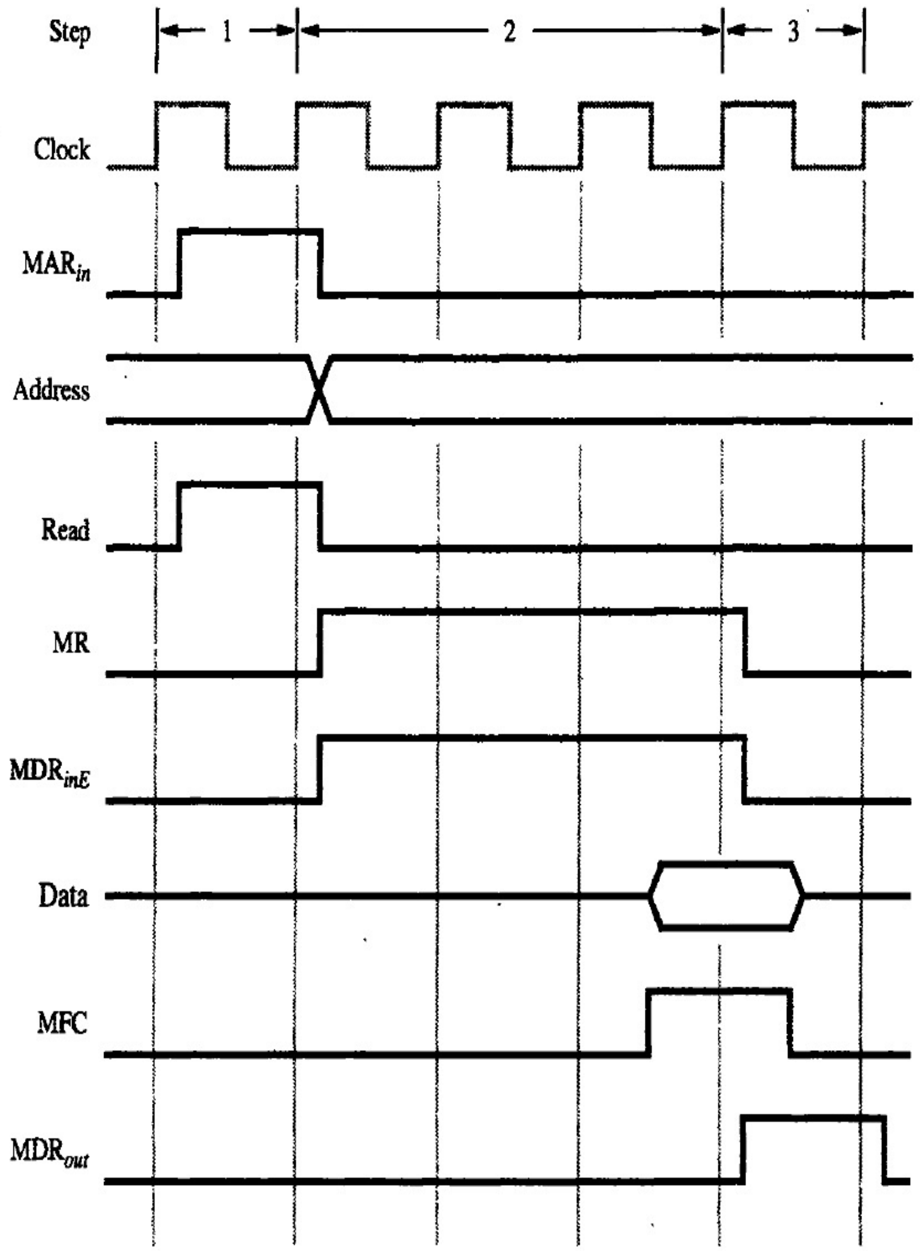
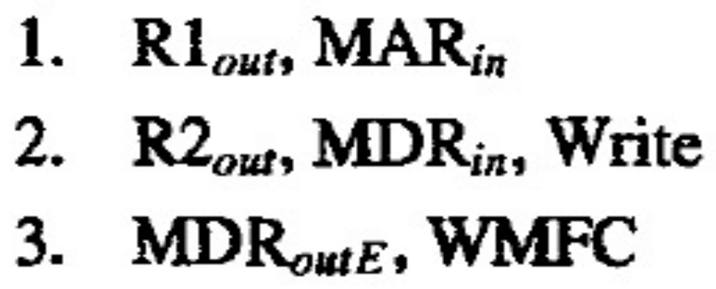


Figure 7.5 Timing of a memory Read operation.

Storing a word in Memory

Example: Move R2,(R1)



1. $R1_{out}$, MAR_{in}
2. $R2_{out}$, MDR_{in}, Write
3. MDR_{outE}, WMFC

Lecture-16
(UNIT-II)

Execution of complete instruction Add (R3), R1

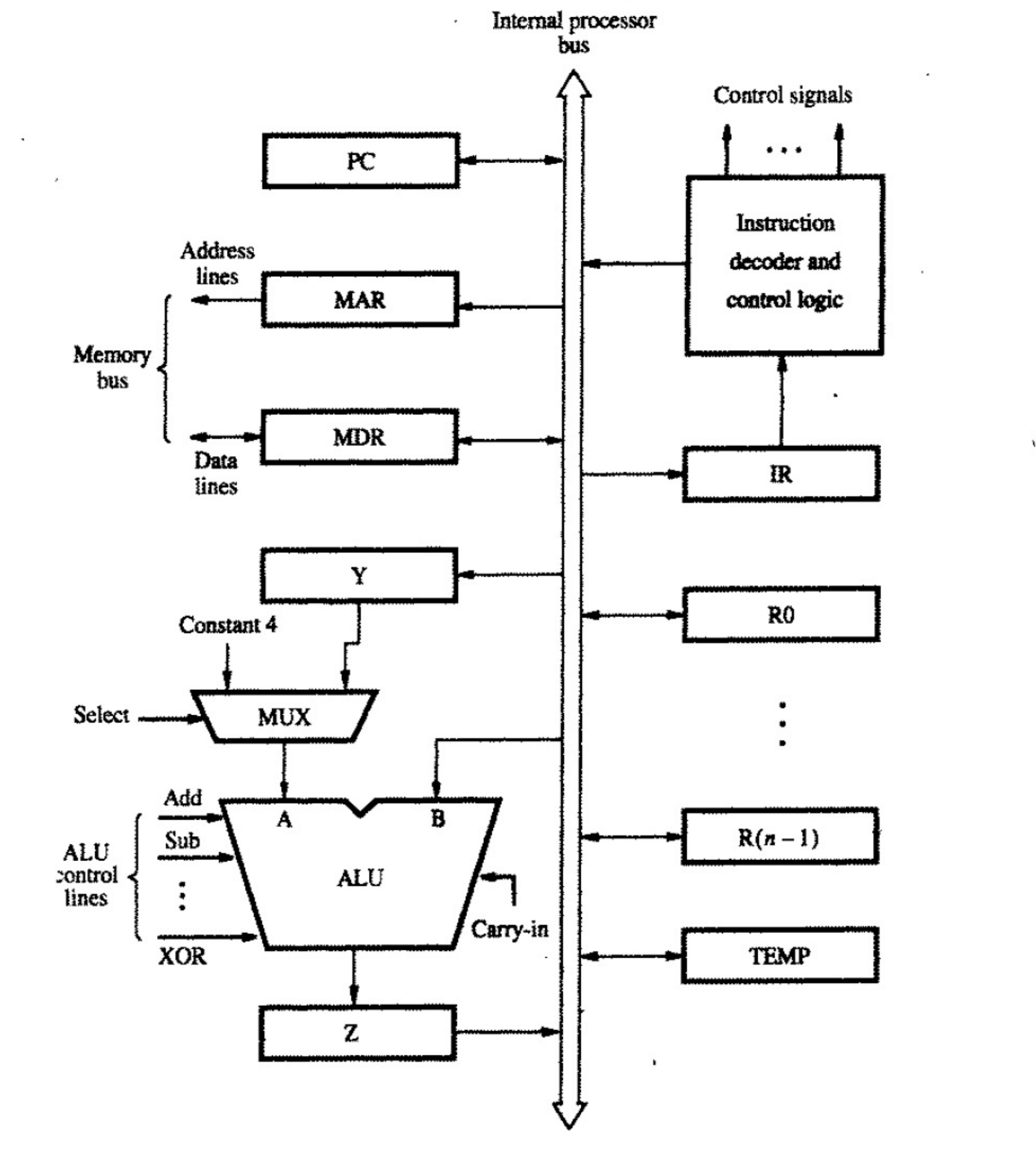


Figure 7.1 Single-bus organization of the datapath inside a processor.

Step	Action
1	PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}
2	Z_{out}, PC_{in}, Y_{in}, WMFC
3	MDR_{out}, IR_{in}
4	$R3_{out}$, MAR_{in}, Read
5	$R1_{out}$, Y_{in}, WMFC
6	MDR_{out}, SelectY, Add, Z_{in}
7	Z_{out}, $R1_{in}$, End

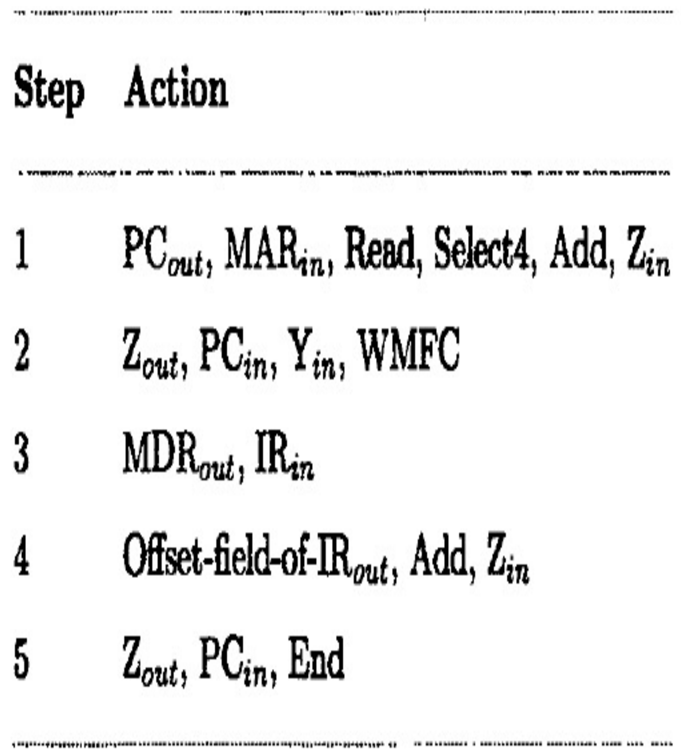
Figure 7.6 Control sequence for execution of the instruction Add (R3),R1.

Instruction Fetch: Steps 1-3
Instruction Execution: Steps 4-7

Executing Branch Instruction

- Branch instruction replaces the address of the PC with the branch address
- This value is obtained by adding offset X to the updated value of PC
- The offset value is the difference between the branch target address and the address of the instruction following the branch instruction
- For *conditional branch* instruction Branch<0, step 4 is replaced by

Offset-field-of-IR_{out}, Add, Z_{in}, If $N = 0$ then End



Step	Action
1	PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}
2	Z_{out}, PC_{in}, Y_{in}, WMFC
3	MDR_{out}, IR_{in}
4	Offset-field-of-IR_{out}, Add, Z_{in}
5	Z_{out}, PC_{in}, End

Figure 7.7 Control sequence for an unconditional Branch instruction.

Multiple Bus Organization

- To *reduce the number of steps* needed multiple internal paths are provided to perform *many steps in parallel*
- *Three ports* for register file, two for output and one for input
- Can be performed in a single step(clock cycle)
- There is no need of registers Y and Z here
- If needed ALU may pass one of its two input operands unmodified to Bus C (*control signals: R=A or R=B*)
- *Incrementor unit* eliminates the need for Adding 4 to PC
- Constant 4 to MUX before ALU can still be useful for incrementing other memory addresses, in instructions such as *LoadMultiple* and *StoreMultiple*

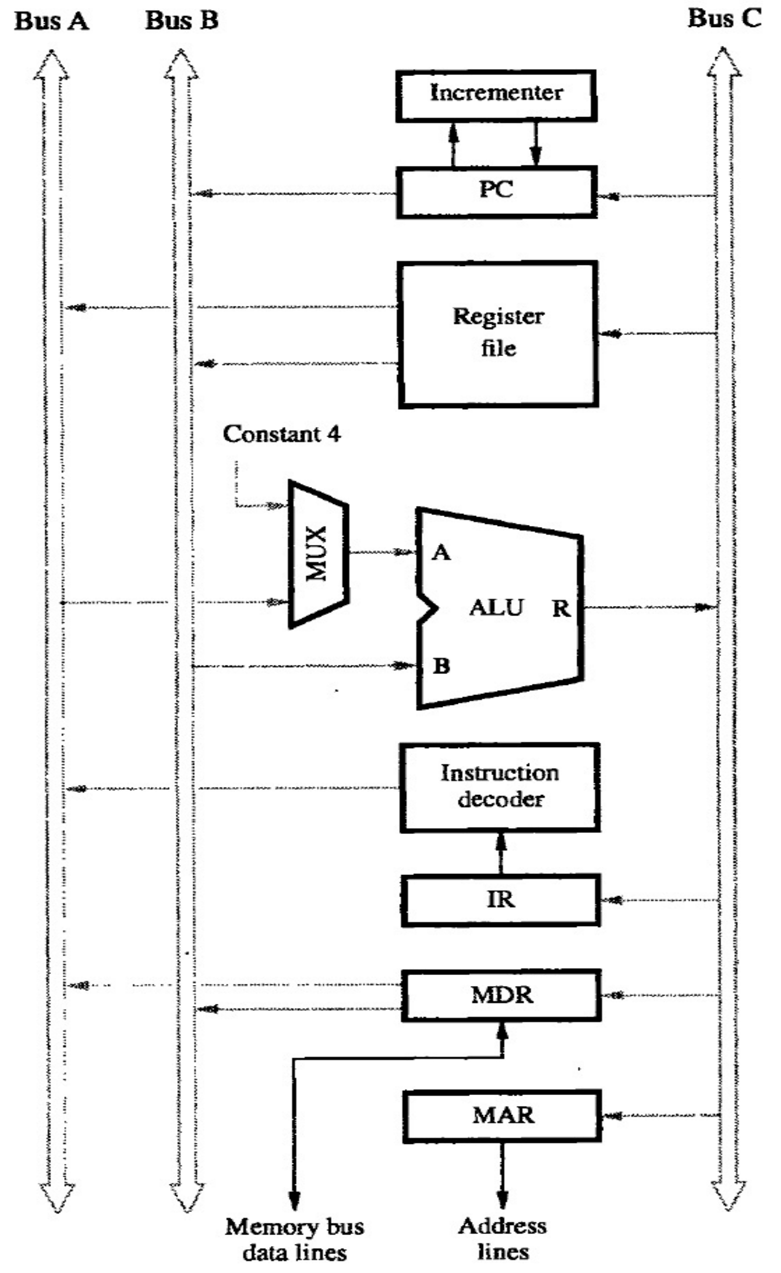
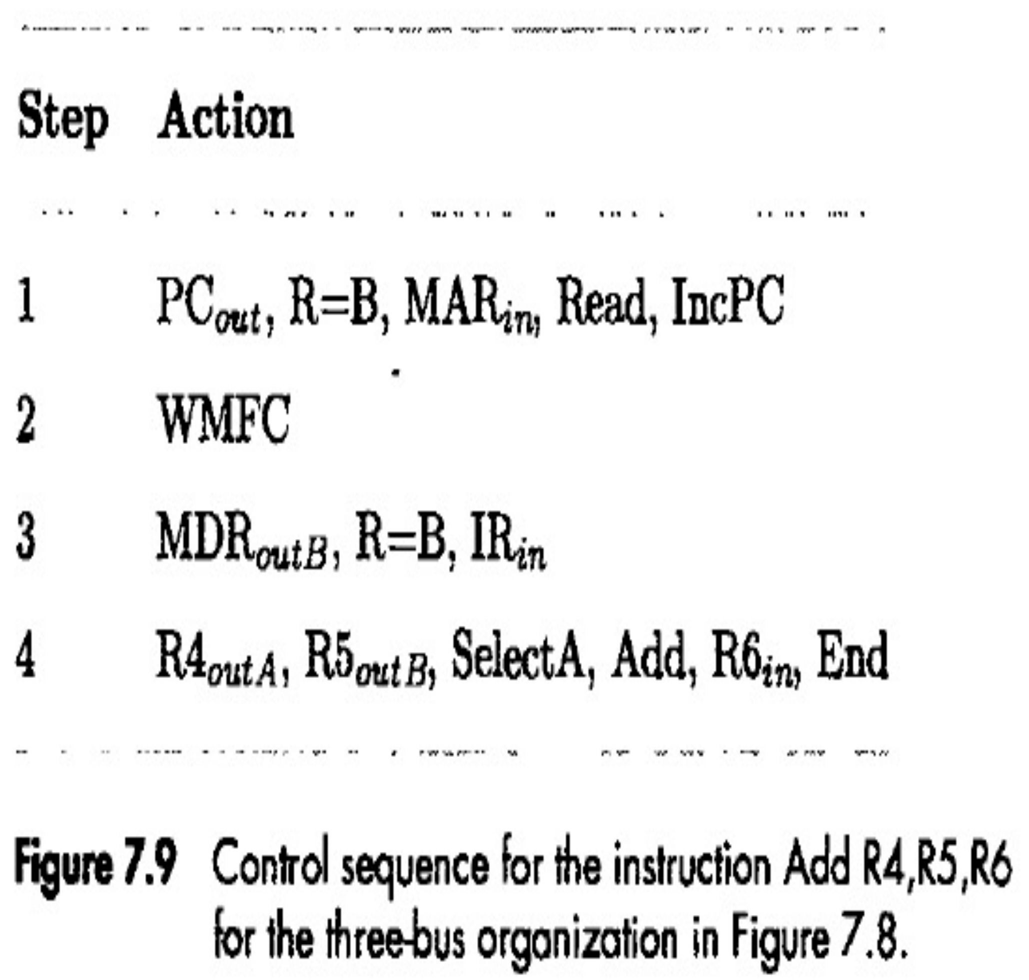


Figure 7.8 Three-bus organization of the datapath.

Ex. Add R4,R5,R6 on Three-Bus Processor

- Step 1-3 Instruction Fetch
- Step 4 Instruction execution
- Advantage: Number of clock cycles needed to perform an instruction are significantly reduced using multiple buses
- **Home Work:**
  - Write the step wise control signals for the same instruction using single-bus processor



Step	Action
1	PC_{out}, R=B, MAR_{in}, Read, IncPC
2	WMFC
3	MDR_{outB}, R=B, IR_{in}
4	$R4_{outA}$, $R5_{outB}$, SelectA, Add, $R6_{in}$, End

Figure 7.9 Control sequence for the instruction Add R4,R5,R6 for the three-bus organization in Figure 7.8.