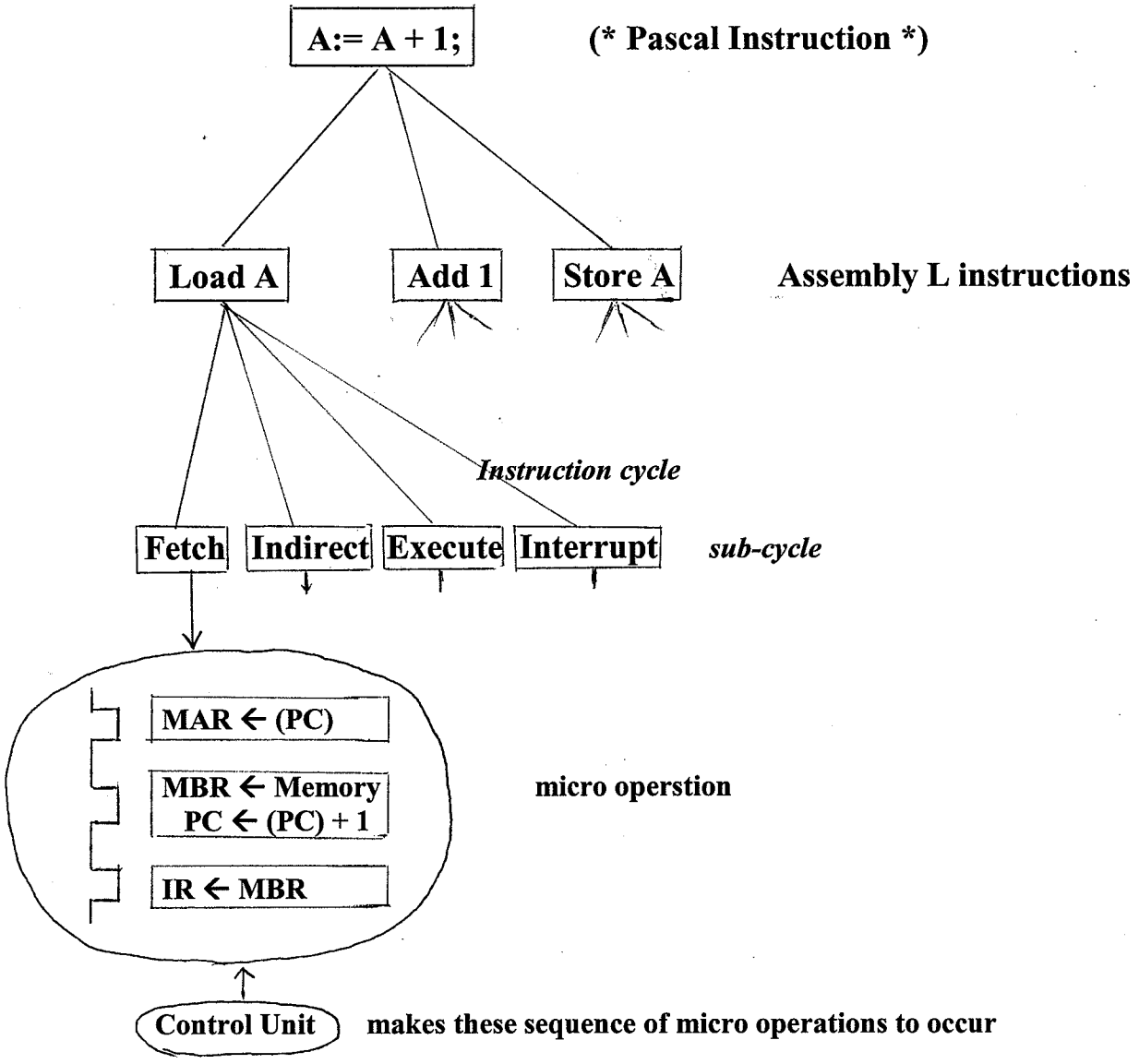
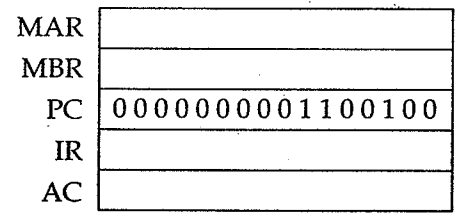
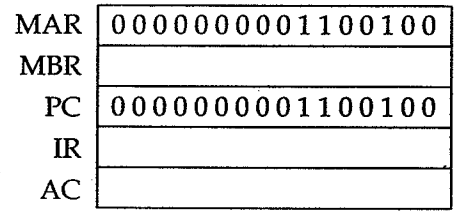
16. Control Unit



Note. CU is a combinational circuit

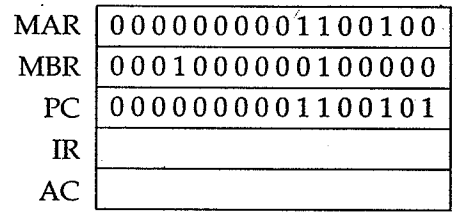


(a) Beginning



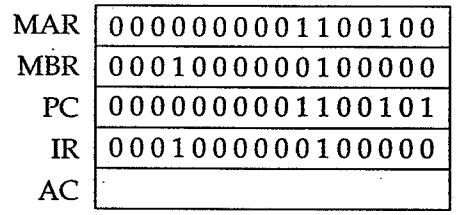
t_1: MAR ← (PC)

(b) First Step



t_2: MBR ← Memory
PC ← (PC) + 1

(c) Second Step



t_3: IR ← (MBR)

(d) Third Step

FIGURE 16.2. Sequence of events, fetch style

The Indirect Cycle

t_1: MAR $\leftarrow$ (IR(Address))
t_2: MBR $\leftarrow$ Memory
t_3: IR(Address) $\leftarrow$ (MBR(Address))

The Execute Cycle

ADD R1, X

t_1: MAR $\leftarrow$ (IR(address))
t_2: MBR $\leftarrow$ Memory
t_3: R1 $\leftarrow$ (R1) + (MBR)

The Interrupt Cycle

t_1: MBR $\leftarrow$ (PC)
t_2: MAR $\leftarrow$ Save-address
PC $\leftarrow$ Routine-address
t_3: Memory $\leftarrow$ (MBR)

The Instruction Cycle

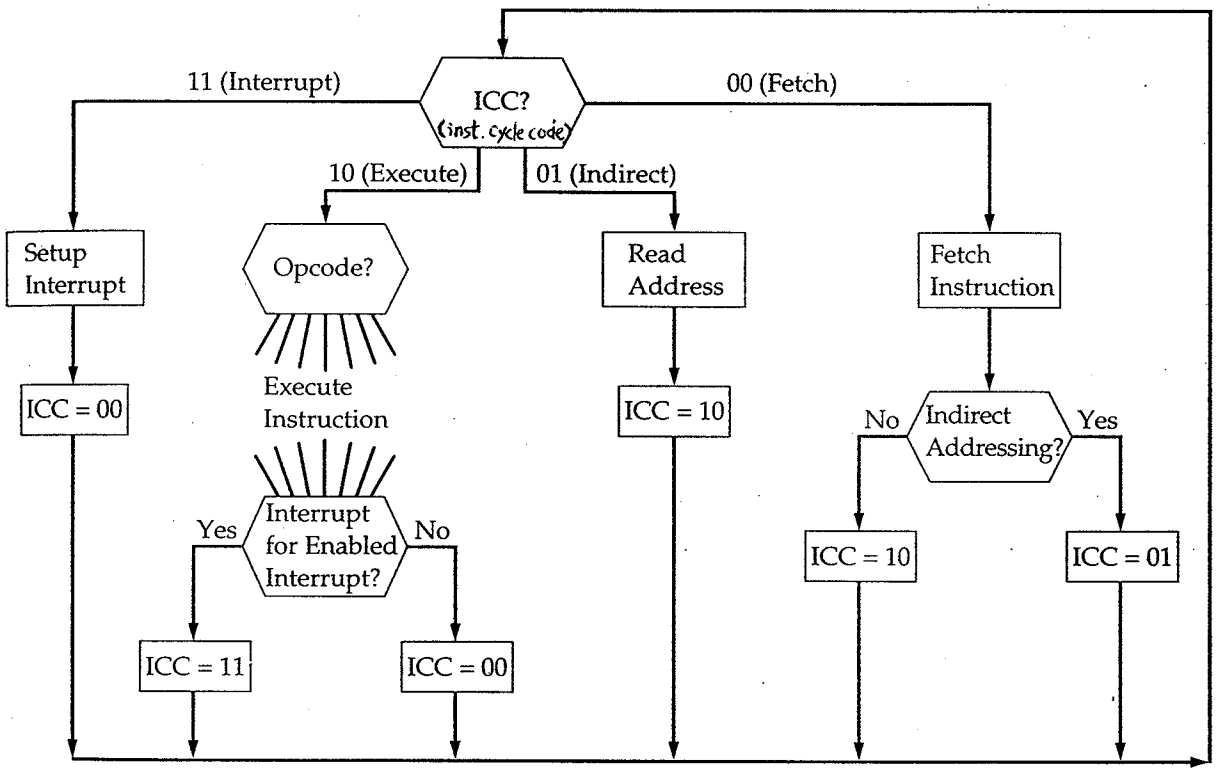


FIGURE 16.3. Flowchart for instruction cycle

16.2

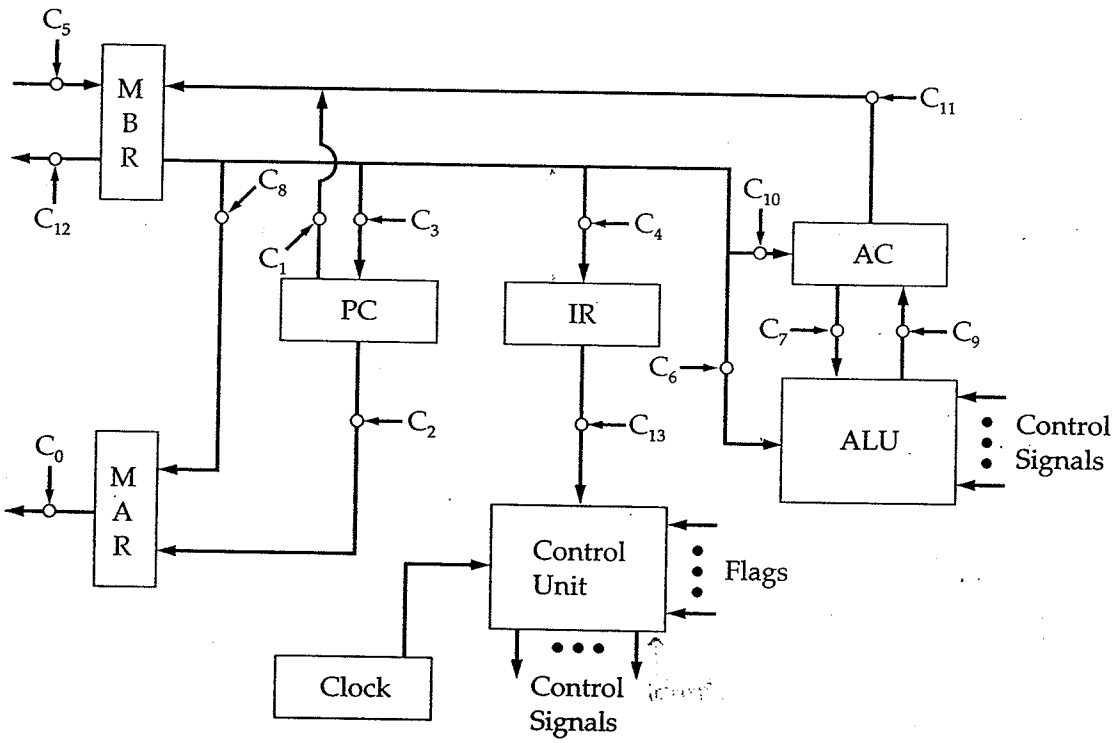


FIGURE 16.5. Data paths and control signals

TABLE 16.1 Micro-operations and Control Signals

Micro-operations		*Active Control Signals*
Fetch:	t_1: MAR ← (PC)	C_2
	t_2: MBR ← Memory	C_5, C_R
	PC ← (PC) + 1	not shown
	t_3: IR ← (MBR)	C_4
Indirect:	t_1: MAR ← (IR(Address))	C_8
	t_2:MBR ← Memory	C_5, C_R
	t_3: IR(Address) ← (MBR(Address))	C_4
Interrupt:	t_1: MBR ← (PC)	C_1
	t_2: MAR ← Save-address	
	PC ← Routine-address	not shown
	t_3: Memory ← (MBR)	C_{12}, C_W

C_R = Read control signal to system bus.
C_W = Write control signal to system bus.

Internal CPU Organization

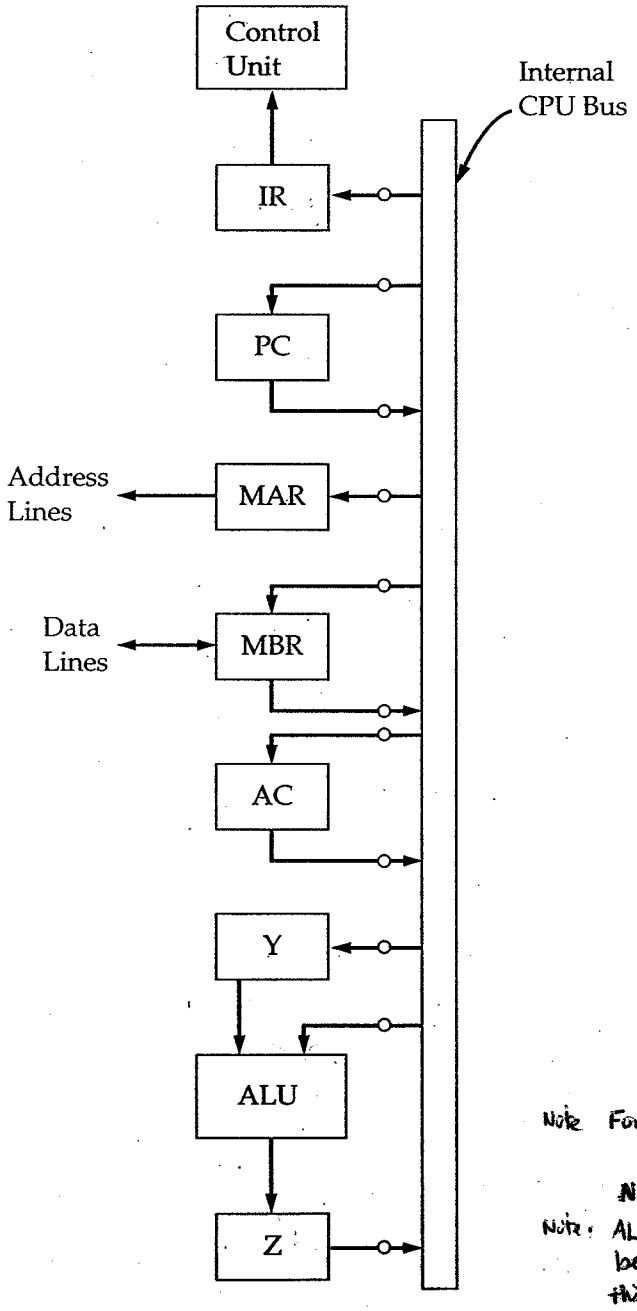


FIGURE 16.6. CPU with internal bus

ADD

t_1 : MAR ← (IR(addy))
t_2 : MBR ← Memory
t_3 : Y ← (MBR)
t_4 : Z ← (AC)+(Y)
t_5 : AC ← Z

Note: For multiple general-purpose register case, add additional registers, Y and Z.
Now, ADD needs 5 micro operations.

Note: ALU is combinational circuits, output cannot be directly connected to the bus, because this output would feed back to the input. So we need Z for temporary storage.

16.3

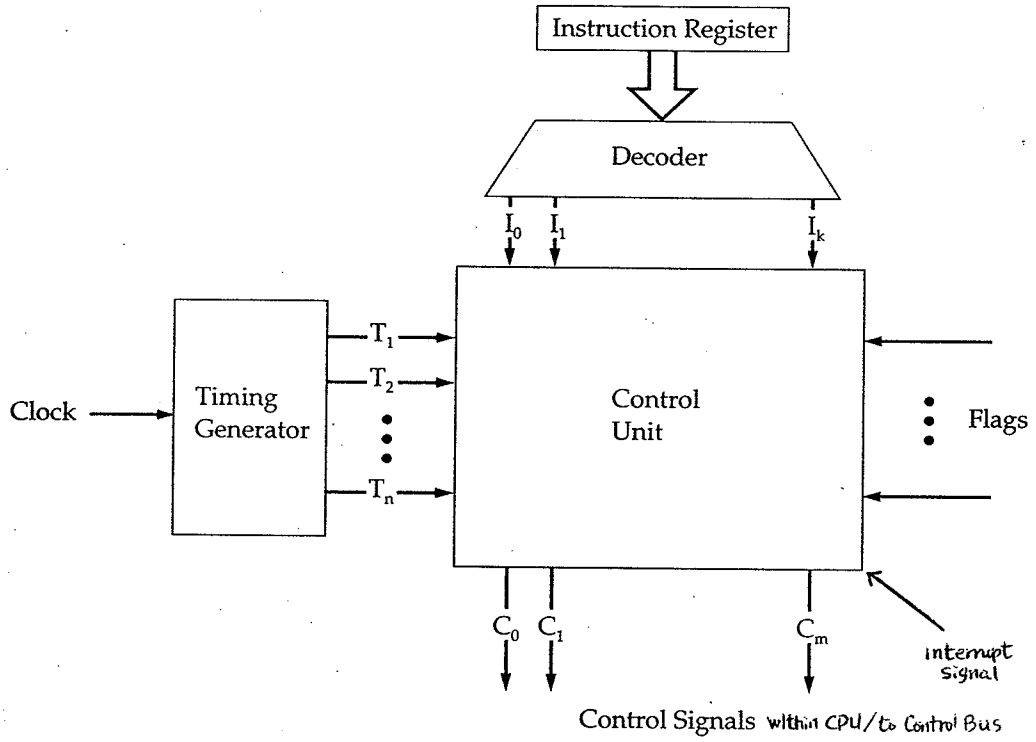


FIGURE 16.10. Control unit with decoded inputs

PQ = 00	Fetch Cycle
PQ = 01	Indirect Cycle
PQ = 10	Execute Cycle
PQ = 11	Interrupt Cycle

$$C_5 = \overline{P} \cdot \overline{Q} \cdot T_2 + \overline{P} \cdot Q \cdot T_2 + P \cdot \overline{Q} \cdot (LDA + ADD + AND) \cdot T_2$$

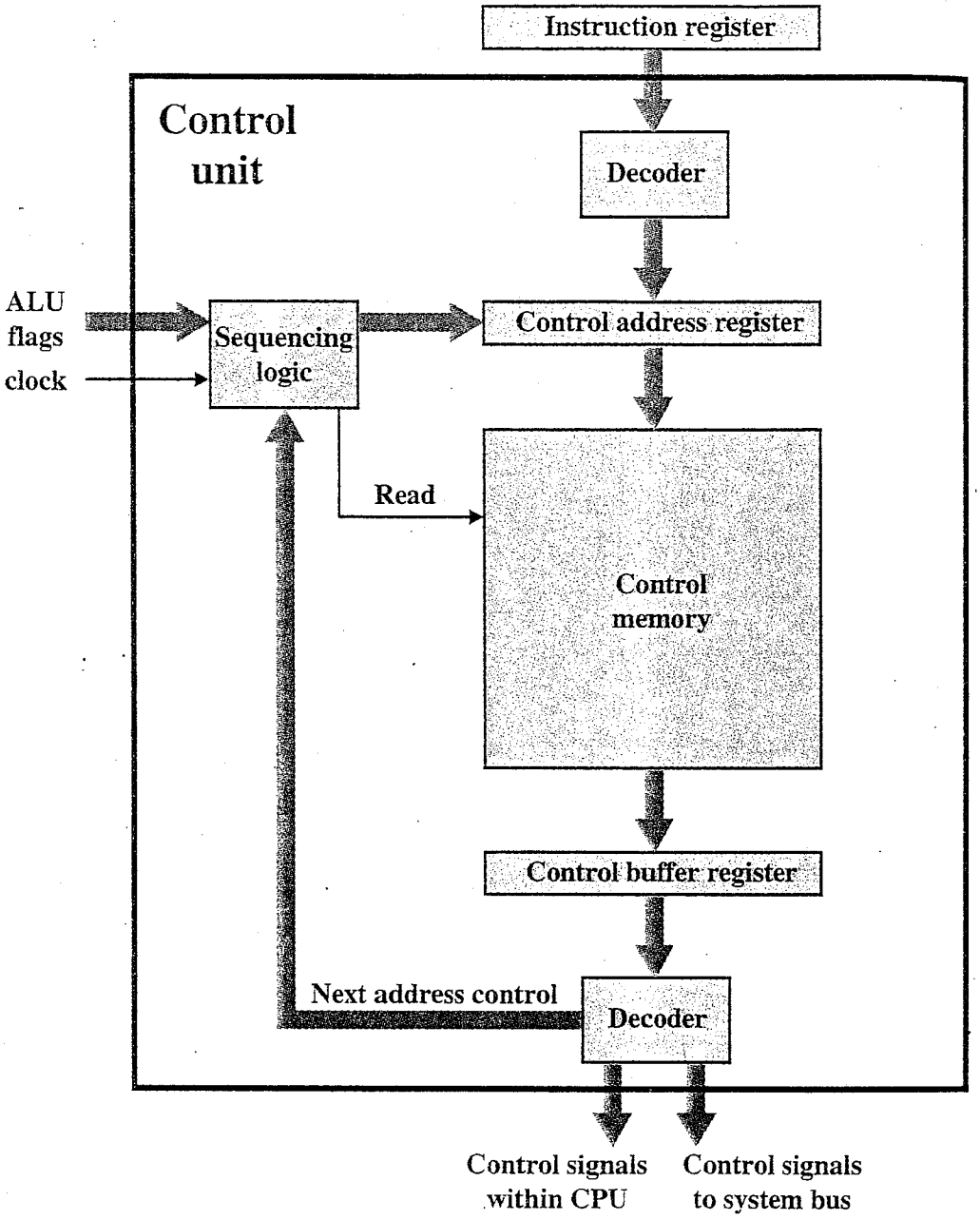
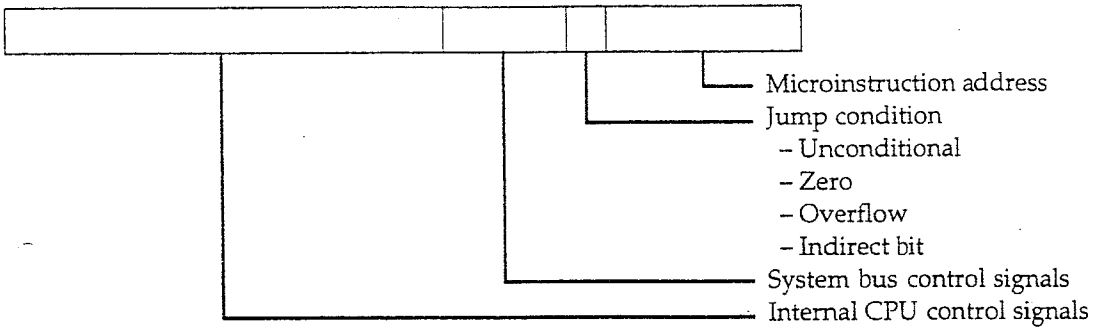


Figure 17.4 Functioning of Microprogrammed Control Unit



(a) Horizontal microinstruction

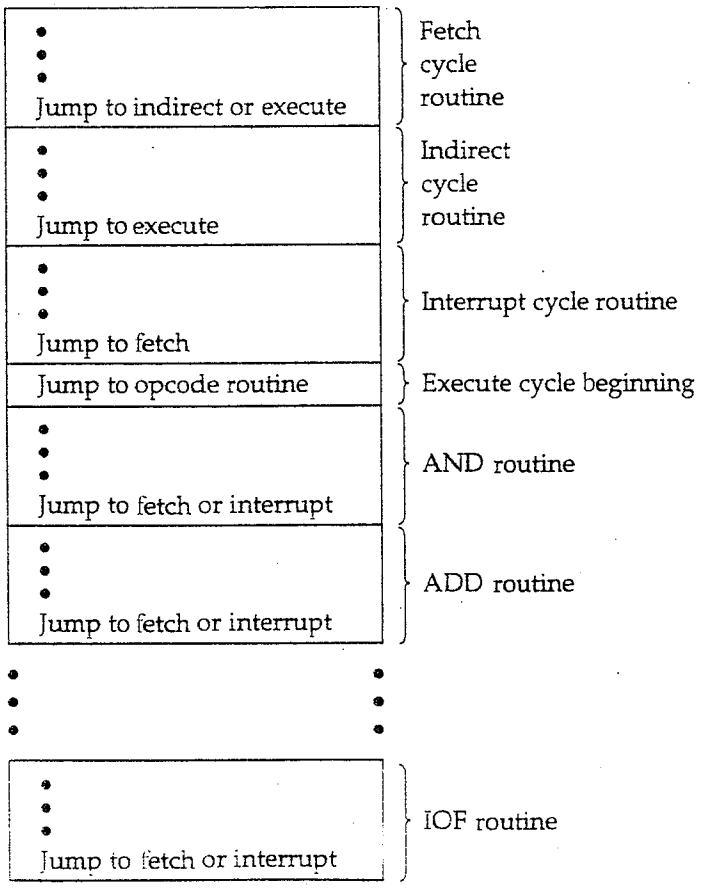


Figure 17.2 Organization of Control Memory

Methods used to achieve high performance

. Bit-parallel memory

. Bit-parallel arithmetic (carry look-ahead adder)

. I/O processor
- **cpu speed vs. I/O speed**
- **interrupt-driven**

. Memory interleaving
- **memory speed vs. cpu speed**

Ex. 4-way interleaving (ATLAS)

Straight storage format

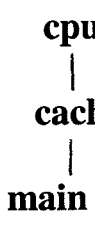
A[0,0] A[0,1] A[0,2] A[0,3]
A[1,0] A[1,1] A[1,2] A[1,3]
A[2,0] A[2,1] A[2,2] A[2,3]
A[3,0] A[3,1] A[3,2] A[3,3]
memory bank

Skewed storage format

A[0,0] A[0,1] A[0,2] A[0,3]
A[1,3] A[1,0] A[1,1] A[1,2]
A[2,2] A[2,3] A[2,0] A[2,1]
A[3,1] A[3,2] A[3,3] A[3,0]

SIMD – using index register

. Associative memory
- **content-addressable memory**
- **cache memory**



hit ration: 95%

principle of locality
temporal locality
spatial locality

. Multiple functional units

Ex. CDC 6600 – 10 functional units

. Pipelined functional units

pipelined adder

. Instruction look-ahead

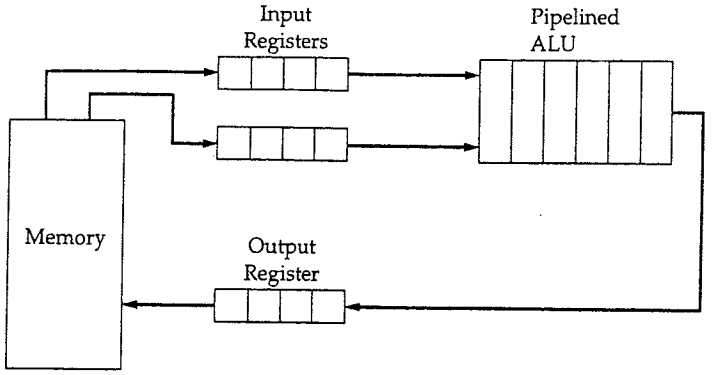
. Instruction pipelining

instruction fetch
decode
operand fetch
execute

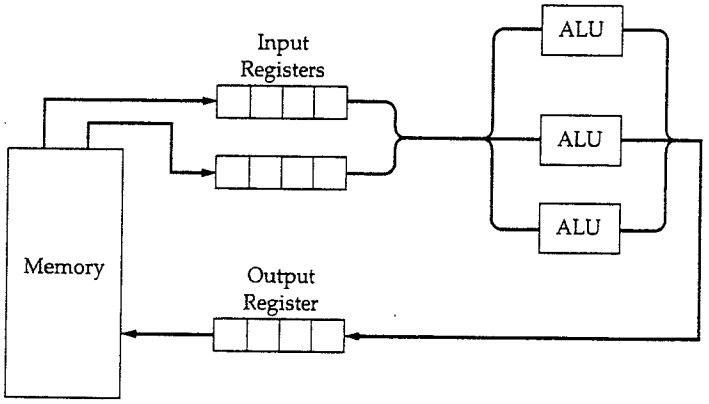
Ex. Amdahl 470 has 12 cycles/instruction

. Data pipelining

- pipelined vector processors

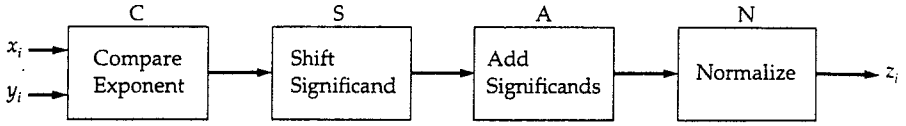


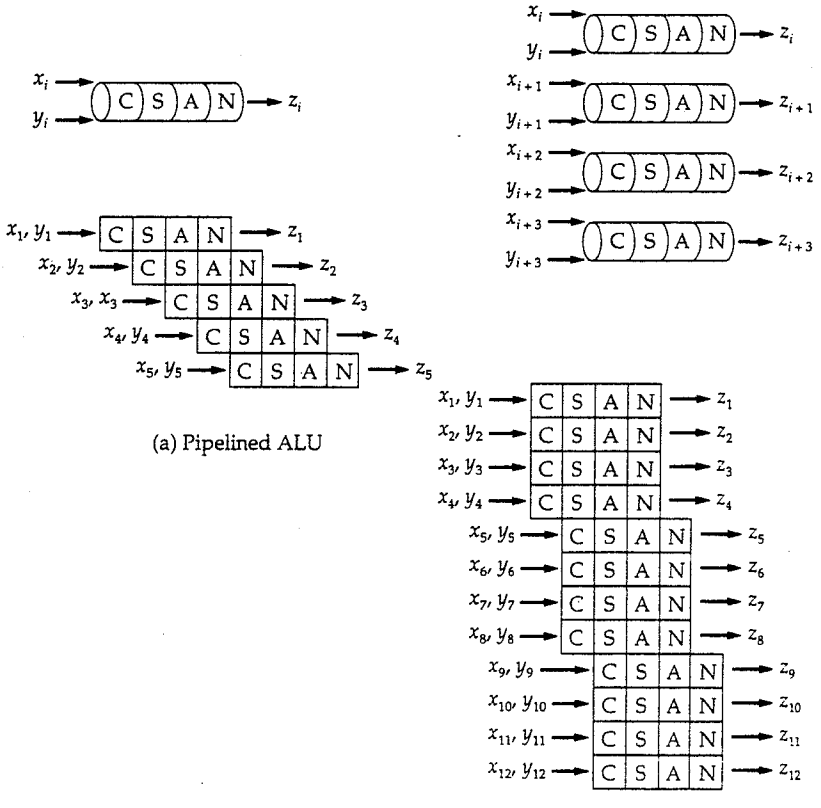
(a) Pipelined ALU



(b) Parallel ALUs

Figure 16.13 Approaches to Vector Computation.





(a) Pipelined ALU

(b) Four Parallel ALUs

Figure 16.14 Pipelined Processing.