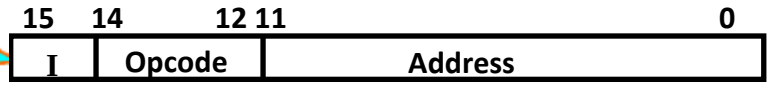
Basic Computer Instructions

Basic Computer Instruction Format

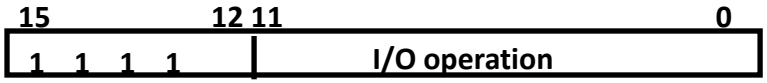
1. Memory-Reference Instructions (OP-code = 000 ~ 110)

I=0 : Direct,
I=1 : Indirect



2. Register-Reference Instructions (OP-code = 111, I = 0)

3. Input-Output Instructions (OP-code =111, I = 1)



Basic Computer Instructions

- **Only 3 bits are used for operation code**
- **It may seem computer is restricted to eight different operations**
- **however register reference and input output instructions use remaining 12 bit as part of operation code**
- **so total number of instruction can exceed 8**
- **Infact total no. of instructions chosen for basic computer is 25**

Basic Computer Instructions

Symbol	Hex Code I = 0	Hex Code I = 1	Description
AND	0xxx	8xxx	AND memory word to AC
ADD	1xxx	9xxx	Add memory word to AC
LDA	2xxx	Axxx	Load AC from memory
STA	3xxx	Bxxx	Store content of AC into memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear AC
CLE	7400		Clear E
CMA	7200		Complement AC
CME	7100		Complement E
CIR	7080		Circulate right AC and E
CIL	7040		Circulate left AC and E
INC	7020		Increment AC
SPA	7010		Skip next instr. if AC is positive
SNA	7008		Skip next instr. if AC is negative
SZA	7004		Skip next instr. if AC is zero
SZE	7002		Skip next instr. if E is zero
HLT	7001		Halt computer
INP	F800		Input character to AC
OUT	F400		Output character from AC
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrupt on
IOF	F040		Interrupt off

- In basic computer instruction format, the opcodes 000 to 110 are reserved for …….

a) Register Reference Instructions
b) Input Output Instructions
c) Memory Reference Instructions
d) None of the above

Instruction Set Completeness

A computer should have a set of instructions so that the user can construct machine language programs to evaluate any function that is known to be computable.

The set of instructions are said to be complete if computer includes a sufficient number of instruction in each of the following categories :

- **Functional Instructions**
  - **Arithmetic, logic, and shift instructions**
  - **ADD, CMA, INC, CIR, CIL, AND, CMA, CLA**
- **Transfer Instructions**
  - **Data transfers between the main memory and the processor registers**
  - **LDA, STA**
- **Control Instructions**
  - **Program sequencing and control**
  - **BUN, BSA, ISZ**
- **Input/output Instructions**
  - **Input and output**
  - **INP, OUT**

Control Unit

- **Control unit (CU) of a processor translates from machine instructions to the control signals for the microoperations that implement them**

- **Control units are implemented in one of two ways**

  Hardwired Control

  CU is made up of sequential and combinational circuits to generate the control signals

  Advantage : optimized to provide fast mode of operations

  Disadvantage : requires changes in wiring if design has been modified

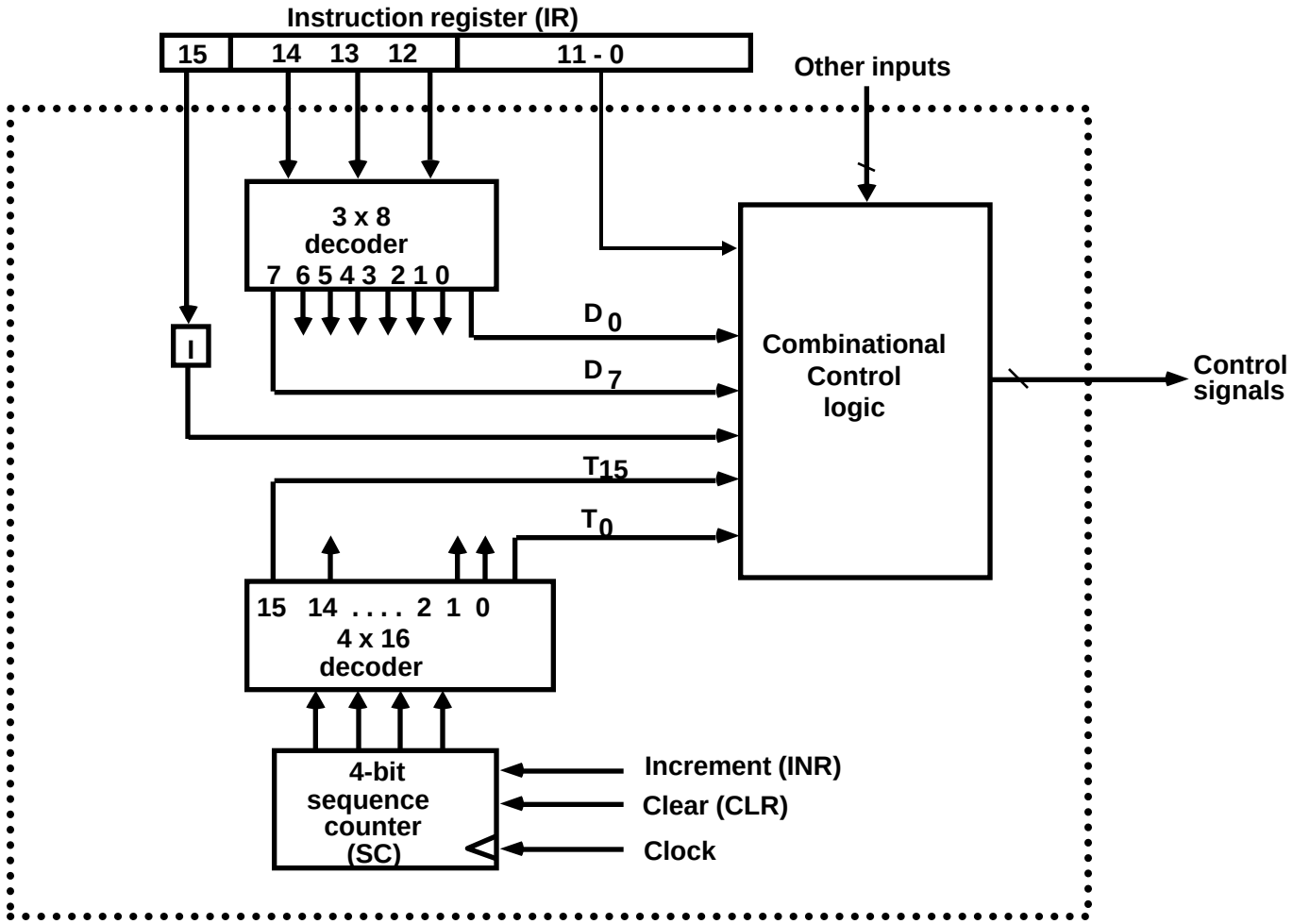
  Microprogrammed Control

  A control memory on the processor contains microprograms that activate the necessary control signals

- **We will consider a hardwired implementation of the control unit for the Basic Computer**

Timing and Control

Control unit of Basic Computer

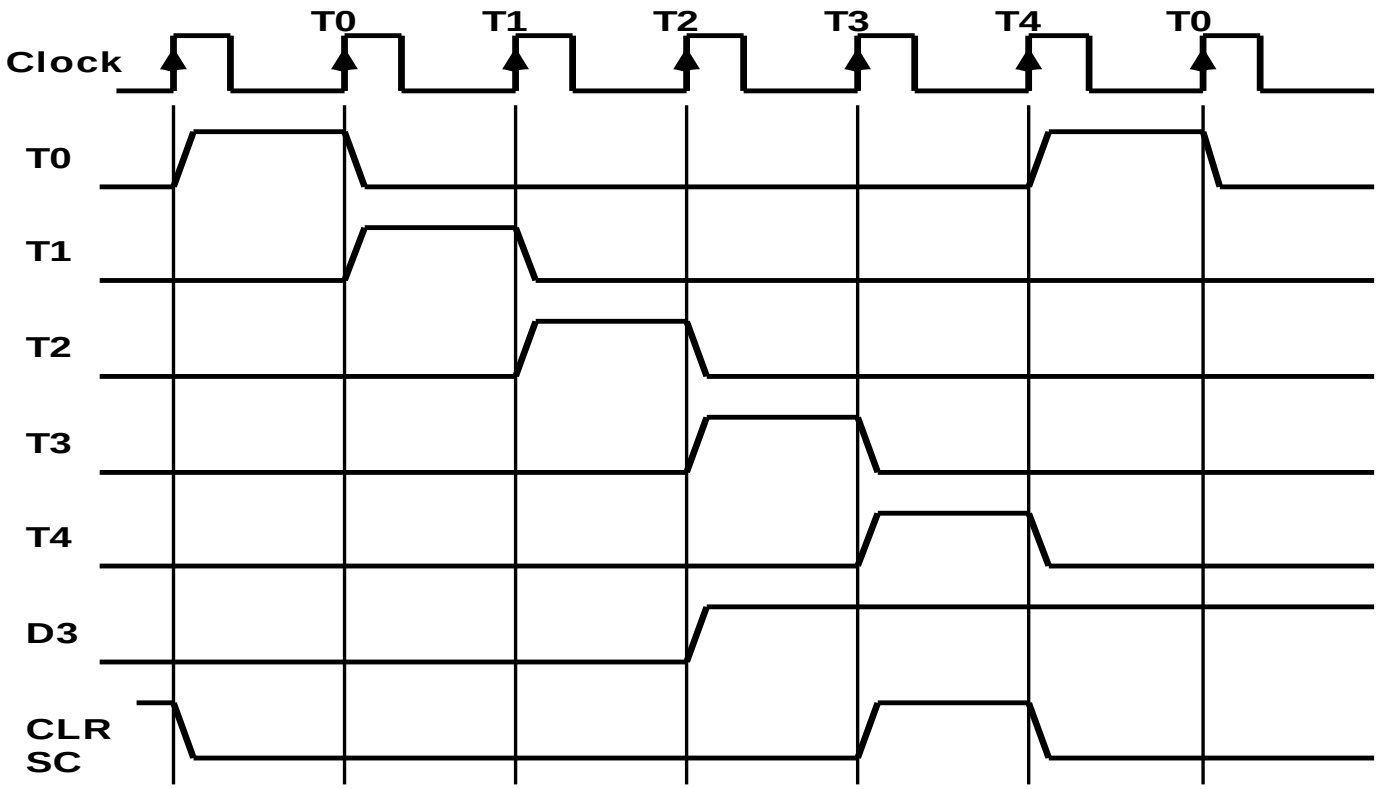


Timing Signals

- Generated by 4-bit sequence counter and 4×16 decoder
- The SC can be incremented or cleared.

- Example: $T_0, T_1, T_2, T_3, T_4, T_0, T_1, \ldots$
  Assume: At time T_4, SC is cleared to 0 if decoder output D3 is active.

D_3T_4: SC ← 0



- Which of the following counters is used in the design of Hardwired control unit?

a) 3-bit Sequence Counter
b) 4-bit Sequence Counter
c) 5-bit Sequence Counter
d) 6-bit Sequence Counter

Instruction Cycle

- **In Basic Computer, a machine instruction is executed in the following cycle:**
  1. **Fetch an instruction from memory**
  2. **Decode the instruction**
  3. **Read the effective address from memory if the instruction has an indirect address**
  4. **Execute the instruction**

- **After an instruction is executed, the cycle starts again at step 1, for the next instruction**

***Note*: Every different processor has its own (different) instruction cycle**

Fetch and Decode

Initially PC loaded with address of first instruction and Sequence counter cleared to 0, giving timing signal T_0

T0: AR ← PC

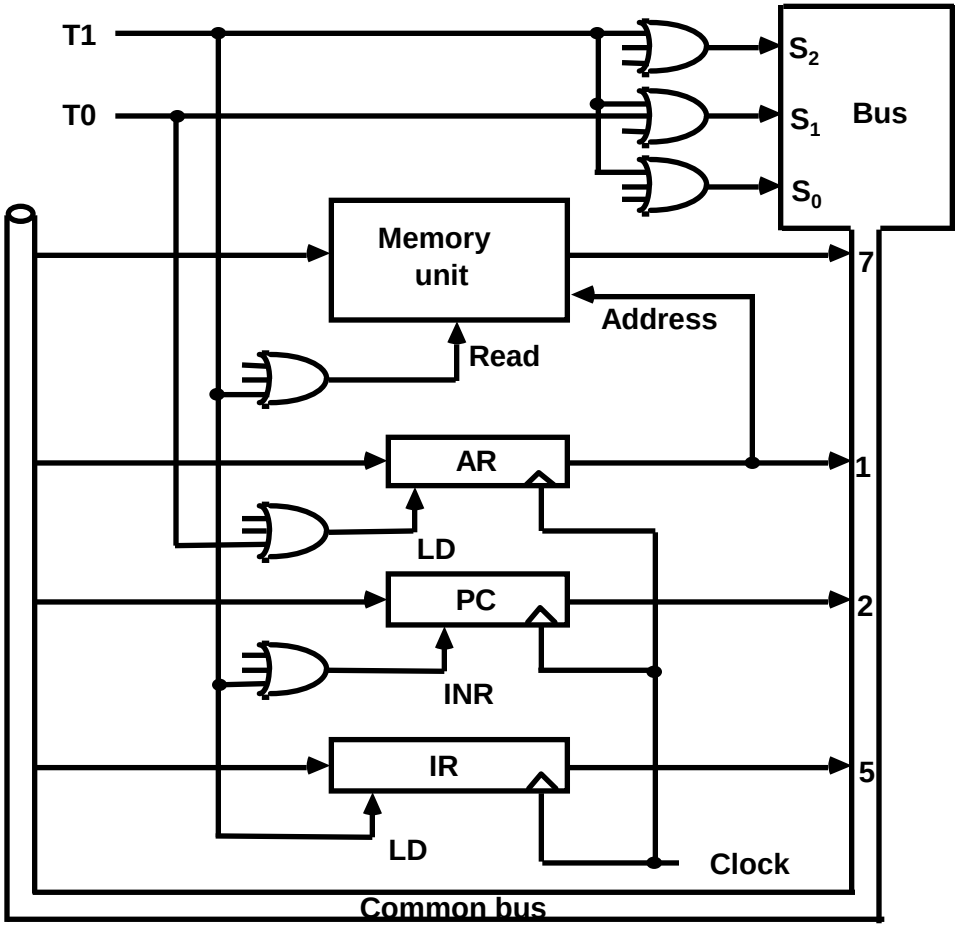
T1: IR ← M [AR], PC ← PC + 1

T2: D0, . . . , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)

Fetch and Decode

Fetch and Decode

T0: AR ← PC ($S_0S_1S_2$=010, T0=1)
T1: IR ← M [AR], PC ← PC + 1 (S0S1S2=111, T1=1)
T2: D0, . . . , D7 ← Decode IR(12-14), AR ← IR(0-11), I ← IR(15)



Fetch and Decode

- Figure shows how first two statements are implemented in bus system
- At T_0 :
  - 1. Place the content of PC into bus by making $S_2S_1S_0$=010
  - Transfer the content of bus to AR by enabling the LD input of AR
- At T_1 :
  - 1. Enable read input of memory
  - 2. Place content of bus by making S2S1S0=111
  - 3. Transfer content of bus to IR by enabling the LD input of IR
  - 4. Increment PC by enabling the INR input of PC

Determine the Type of Instructions

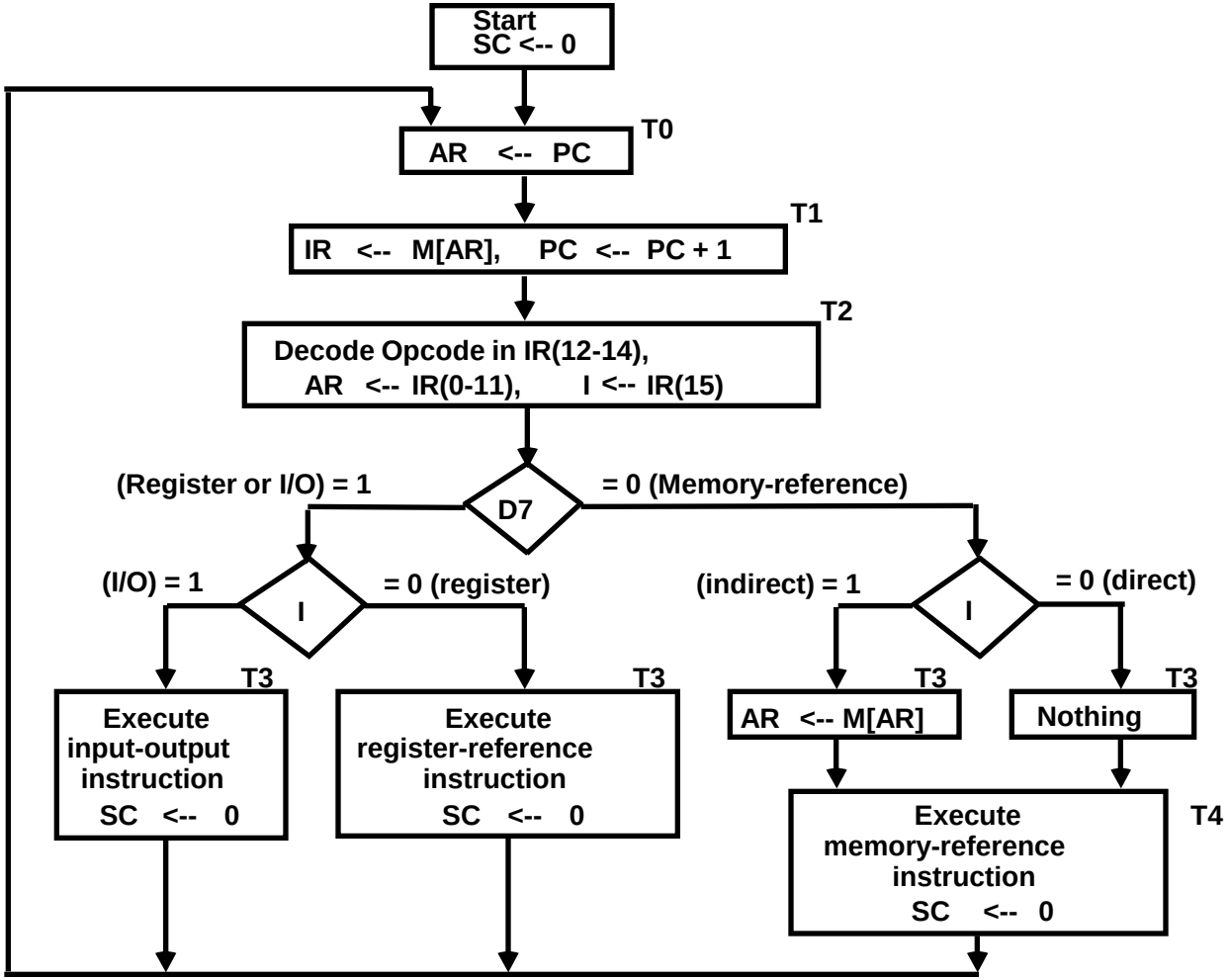


Fig : Flow chart for Instruction Cycle