UNIT 1

Functional Blocks of a Computer: Introduction, Block diagram of digital computer, Instruction codes, Computer Registers, Common bus system, Computer instructions, Instruction cycle and Instruction set, Register Transfer Language.

Data Representation: Fixed and floating point arithmetic- Addition, Subtraction, Multiplication, Division.

Control unit Design: Hardwired control unit, Control memory, Address sequencing, Micro-programmed control unit design, Hardwired Vs Micro-programmed design.

MEMORY REFERENCE INSTRUCTIONS

Symbol	Operation Decoder	Symbolic Description
AND	D_0	$AC \leftarrow AC \wedge M[AR]$
ADD	D_1	$AC \leftarrow AC + M[AR], E \leftarrow C_{out}$
LDA	D_2	$AC \leftarrow M[AR]$
STA	D_3	$M[AR] \leftarrow AC$
BUN	D_4	$PC \leftarrow AR$
BSA	D_5	$M[AR] \leftarrow PC, PC \leftarrow AR + 1$
ISZ	D_6	$M[AR] \leftarrow M[AR] + 1$, if $M[AR] + 1 = 0$ then $PC \leftarrow PC+1$

- The effective address of the instruction is in AR and was placed there during timing signal T_2 when I = 0, or during timing signal T3 when I = 1
- Memory cycle is assumed to be short enough to be completed in a CPU cycle
- The execution of MR Instruction starts with T_4

AND to AC

D_0T_4: $DR \leftarrow M[AR]$ Read operand

D_0T_5: $AC \leftarrow AC \wedge DR, SC \leftarrow 0$ AND with AC

ADD to AC

D_1T_4: $DR \leftarrow M[AR]$ Read operand

D_1T_5: $AC \leftarrow AC + DR, E \leftarrow C_{out}, SC \leftarrow 0$ Add to AC and store carry in E

MEMORY REFERENCE INSTRUCTIONS cont.

LDA: Load to AC

D_2T_4: DR ← M[AR]

D_2T_5: AC ← DR, SC ← 0

STA: Store AC

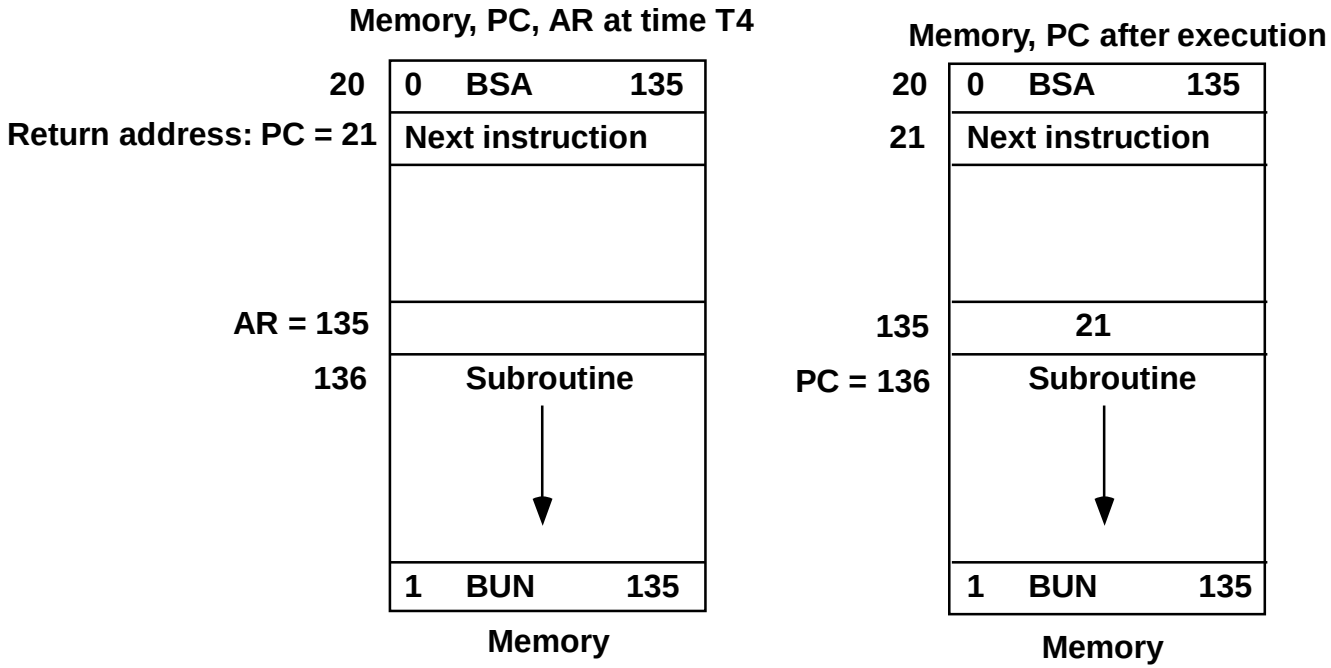
D_3T_4: M[AR] ← AC, SC ← 0

BUN: Branch Unconditionally

D_4T_4: PC ← AR, SC ← 0

BSA: Branch and Save Return Address

M[AR] ← PC, PC ← AR + 1



Memory Reference Instructions$^{cont.}$

BSA: executed in a sequence of two micro-operations:

D_5T_4: M[AR] ← PC, AR ← AR + 1

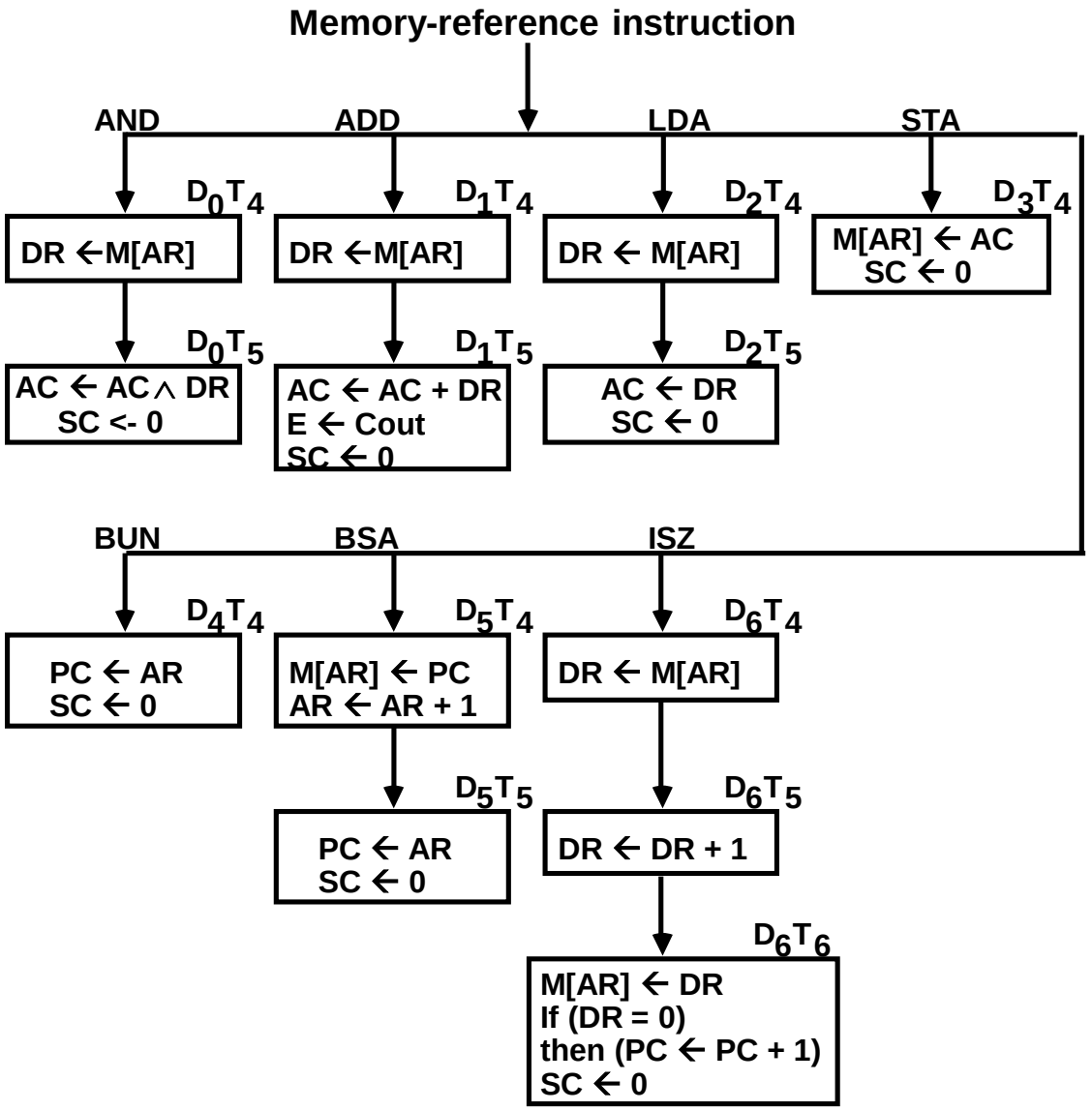
D_5T_5: PC ← AR, SC ← 0

ISZ: Increment and Skip-if-Zero

D_6T_4: DR ← M[AR]

D_6T_5: DR ← DR + 1

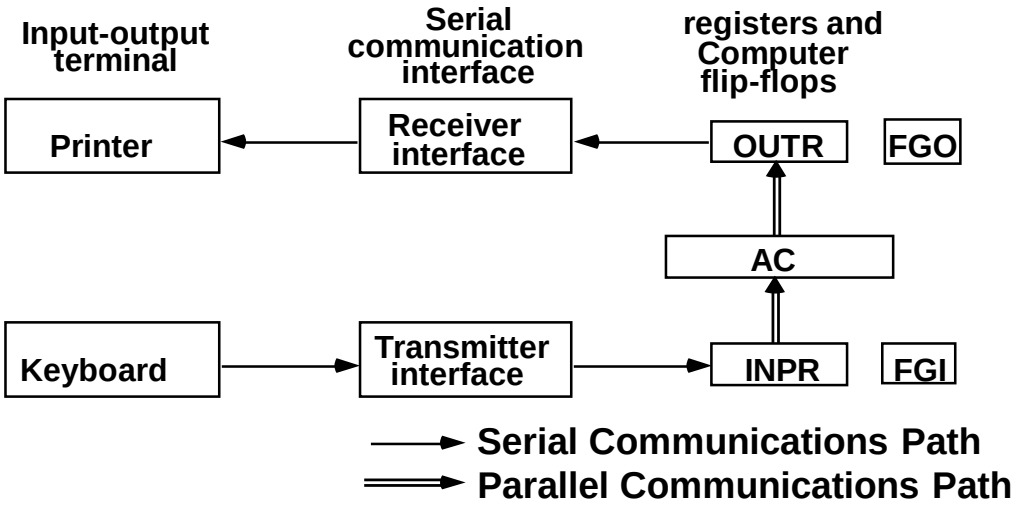
D_6T6: M[AR] ← DR, if (DR = 0) then (PC ← PC + 1), SC ← 0


Memory-reference instruction
AND
ADD
LDA
STA
D_0T_4
D_1T_4
D_2T_4
D_3T_4
DR ← M[AR]
DR ← M[AR]
DR ← M[AR]
M[AR] ← AC
SC ← 0
D_0T_5
D_1T_5
D_2T_5
AC ← AC ∧ DR
SC <- 0
AC ← AC + DR
E ← Cout
SC ← 0
AC ← DR
SC ← 0
BUN
BSA
ISZ
D_4T_4
D_5T_4
D_6T_4
PC ← AR
SC ← 0
M[AR] ← PC
AR ← AR + 1
DR ← M[AR]
D_5T_5
D_6T_5
PC ← AR
SC ← 0
DR ← DR + 1
D_6T_6
M[AR] ← DR
If (DR = 0)
then (PC ← PC + 1)
SC ← 0

Input-Output Instructions

- Instructions and data stored in memory must come from some input device
- Computational results must be transmitted to the user through some output device
- For the system to communicate with an input device, serial information is shifted into the input register INPR
- To output information, it is stored in the output register OUTR

Input-Output Instructions cont.



Input-Output Instructions$^{cont.}$

- INPR and OUTR communicate with a communication interface serially and with the AC in parallel. They hold an 8-bit alphanumeric information
- I/O devices are slower than a computer system → we need to synchronize the timing rate difference between the input/output device and the computer.
- FGI: 1-bit input flag (Flip-Flop) aimed to control the input operation

Input-Output Instructions cont.

- FGI is set to 1 when a new information is available in the input device and is cleared to 0 when the information is accepted by the computer
- FGO: 1-bit output flag used as a control flip-flop to control the output operation
- If FGO is set to 1, then this means that the computer can send out the information from AC. If it is 0, then the output device is busy and the computer has to wait!

Input-Output Instructions cont.

- The process of input information transfer:
  - Initially, FGI is cleared to 0
  - An 8-bit alphanumeric code is shifted into INPR (Keyboard key strike) and the input flag FGI is set to 1
  - As long as the flag is set, the information in INPR cannot be changed by another data entry
  - The computer checks the flag bit; if it is 1, the information from INPR is transferred in parallel into AC and FGI is cleared to 0
  - Once the flag is cleared, new information can be shifted into INPR by the input device (striking another key)

Input-Output Instructions cont.

- The process of outputting information:
  - Initially, the output flag FGO is set to 1
  - The computer checks the flag bit; if it is 1, the information from AC is transferred in parallel to OUTR and FGO is cleared to 0
  - The output accepts the coded information (prints the corresponding character)
  - When the operation is completed, the output device sets FGO back to 1
  - The computer does not load a new data information into OUTR when FGO is 0 because this condition indicates that the output device is busy to receive another information at the moment!!

Input-Output Instructions

- Needed for:
  - Transferring information to and from AC register
  - Checking the flag bits
  - Controlling the interrupt facility
- The control unit recognize it when D_7=1 and I = 1
- The remaining bits of the instruction specify the particular operation
- Executed with the clock transition associated with timing signal T_3
- Input-Output instructions are summarized next

Input-Output Instructions

$D_7IT_3 = p$
$IR(i) = B_i$, i = 6, …, 11

INP	pB_{11}:	AC(0-7) ← INPR, FGI ← 0	Input char. to AC
OUT	pB_{10}:	OUTR ← AC(0-7), FGO ← 0	Output char. from AC
SKI	pB_9:	if(FGI = 1) then (PC ← PC + 1)	Skip on input flag
SKO	pB_8:	if(FGO = 1) then (PC ← PC + 1)	Skip on output flag
ION	pB_7:	IEN ← 1	Interrupt enable on
IOF	pB_6:	IEN ← 0	Interrupt enable off