CAO: Lecture 20
Fetch-Decode-Execute cycle (typically 3 to 5 stage)

Topics Covered

- Program controlled data transfer
- Input-output instructions
- Program-controlled input/output
- Interrupt initiated input/output
- Flowchart for interrupt cycle
- Register transfer operations in interrupt cycle

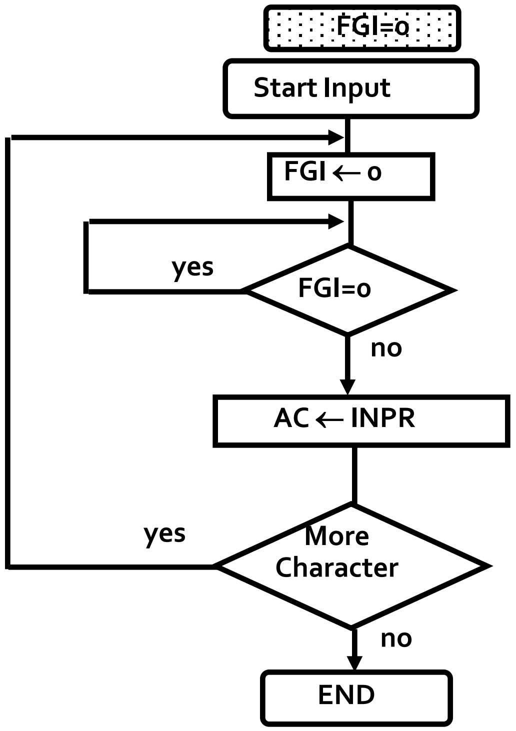
PROGRAM CONTROLLED DATA TRANSFER

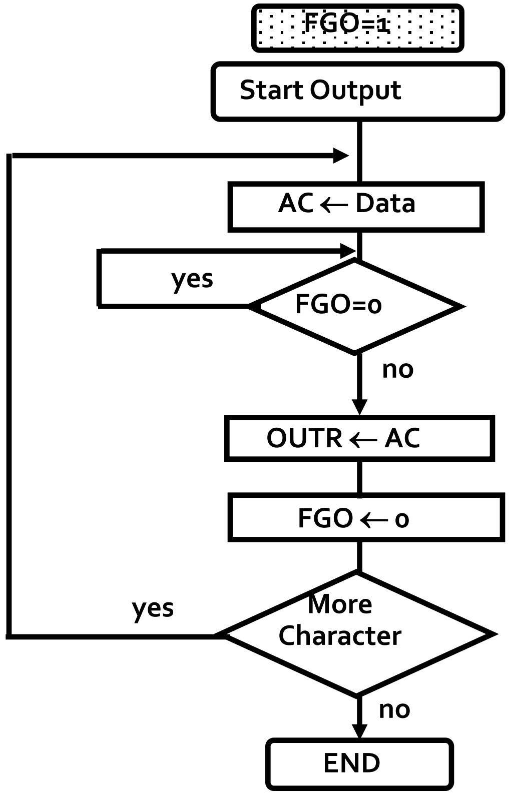
-- CPU --

```
/* Input */        /* Initially FGI = 0 */
  loop:  If FGI = 0 goto loop
         AC ← INPR,  FGI ← 0

/* Output */       /* Initially FGO = 1 */
  loop:  If FGO = 0 goto loop
         OUTR ← AC,  FGO ← 0
```

-- I/O Device --

```
loop: If FGI = 1 goto loop
      INPR ← new data, FGI ← 1

loop: If FGO = 1 goto loop
      consume OUTR, FGO ← 1
```





INPUT-OUTPUT INSTRUCTIONS

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

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

PROGRAM-CONTROLLED INPUT/OUTPUT

- Program-controlled I/O
  - Continuous CPU involvement
    I/O takes valuable CPU time
  - CPU slowed down to I/O speed
  - Simple
  - Least hardware

Input

```
LOOP,    SKI   DEV
         BUN  LOOP
         INP   DEV
```

Output

```
LOOP,    LDA   DATA
LOP,     SKO  DEV
         BUN  LOP
         OUT  DEV
```

INTERRUPT INITIATED INPUT/OUTPUT

-

Open communication only when some data has to be passed --> *interrupt*.

- The I/O interface, instead of the CPU, monitors the I/O device.

- When the interface founds that the I/O device is ready for data transfer, it generates an interrupt request to the CPU

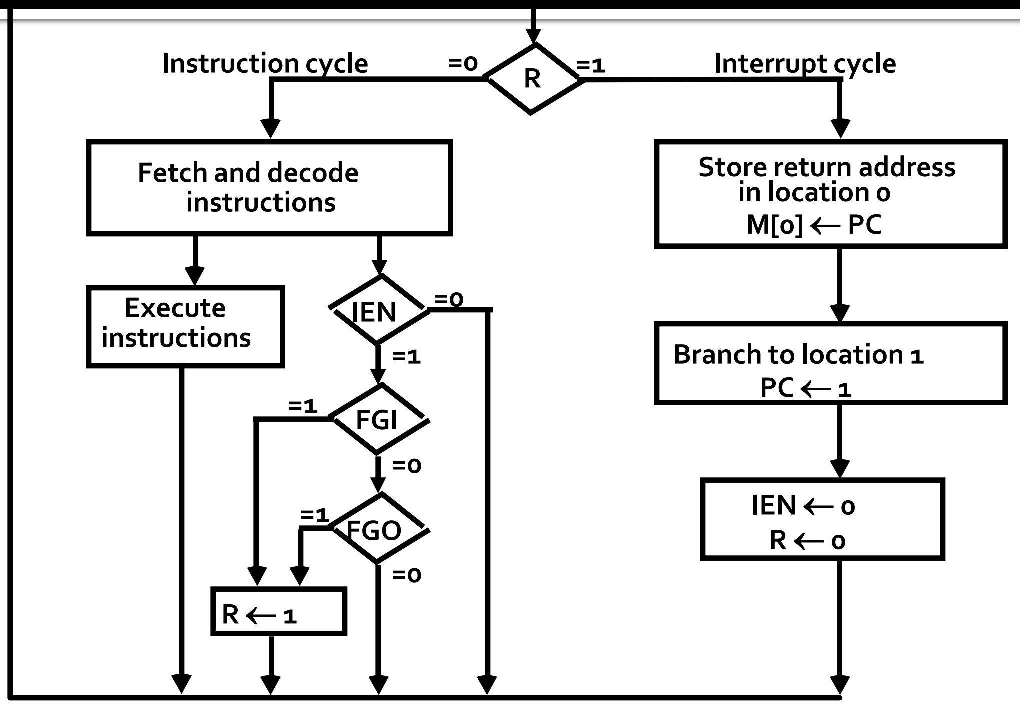
- Upon detecting an interrupt, the CPU stops momentarily the task it is doing, branches to the service routine to process the data transfer, and then returns to the task it was performing.

*

IEN (Interrupt-enable flip-flop)

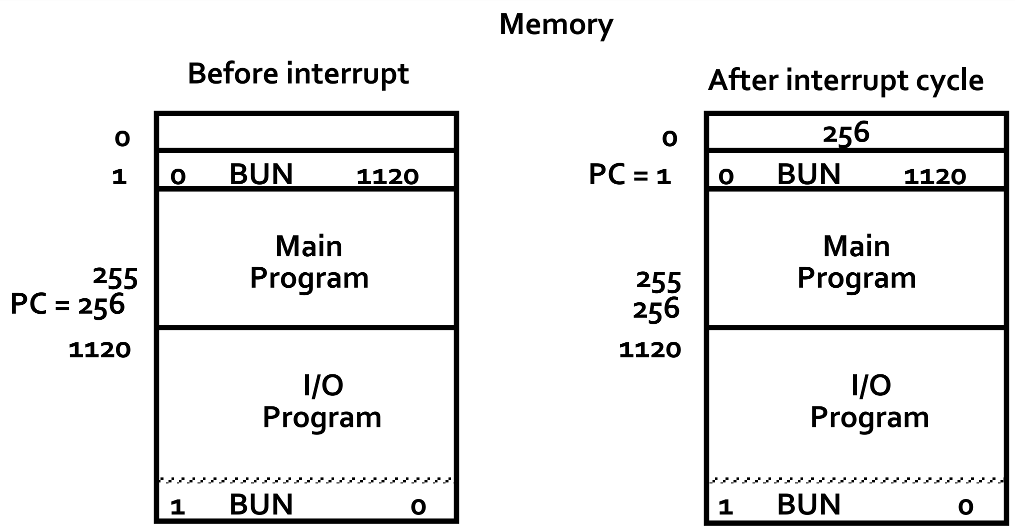
- can be set and cleared by instructions
- when cleared, the computer cannot be interrupted

FLOWCHART FOR INTERRUPT CYCLE



- The interrupt cycle is a HW implementation of a branch and save return address operation.
- At the beginning of the next instruction cycle, the instruction that is read from memory is in address 1.
- At memory address 1, the programmer must store a branch instruction that sends the control to an interrupt service routine
- The instruction that returns the control to the original program is "indirect BUN 0"

REGISTER TRANSFER OPERATIONS IN INTERRUPT CYCLE



Register Transfer Statements for Interrupt Cycle

- R F/F $\leftarrow$ 1 if IEN (FGI + FGO)$T_0'T_1'T_2'$

$\Leftrightarrow T_0'T_1'T_2'$ (IEN)(FGI + FGO): R $\leftarrow$ 1

- The fetch and decode phases of the instruction cycle must be modified ➔Replace T_0, T_1, T_2 with $R'T_0, R'T_1, R'T_2$
- The interrupt cycle :

RT_0: AR $\leftarrow$ 0, TR $\leftarrow$ PC

RT_1: M[AR] $\leftarrow$ TR, PC $\leftarrow$ 0

RT_2: PC $\leftarrow$ PC + 1, IEN $\leftarrow$ 0, R $\leftarrow$ 0, SC $\leftarrow$ 0