Bahan Presentasi Arsitektur dan Organisasi Komputer

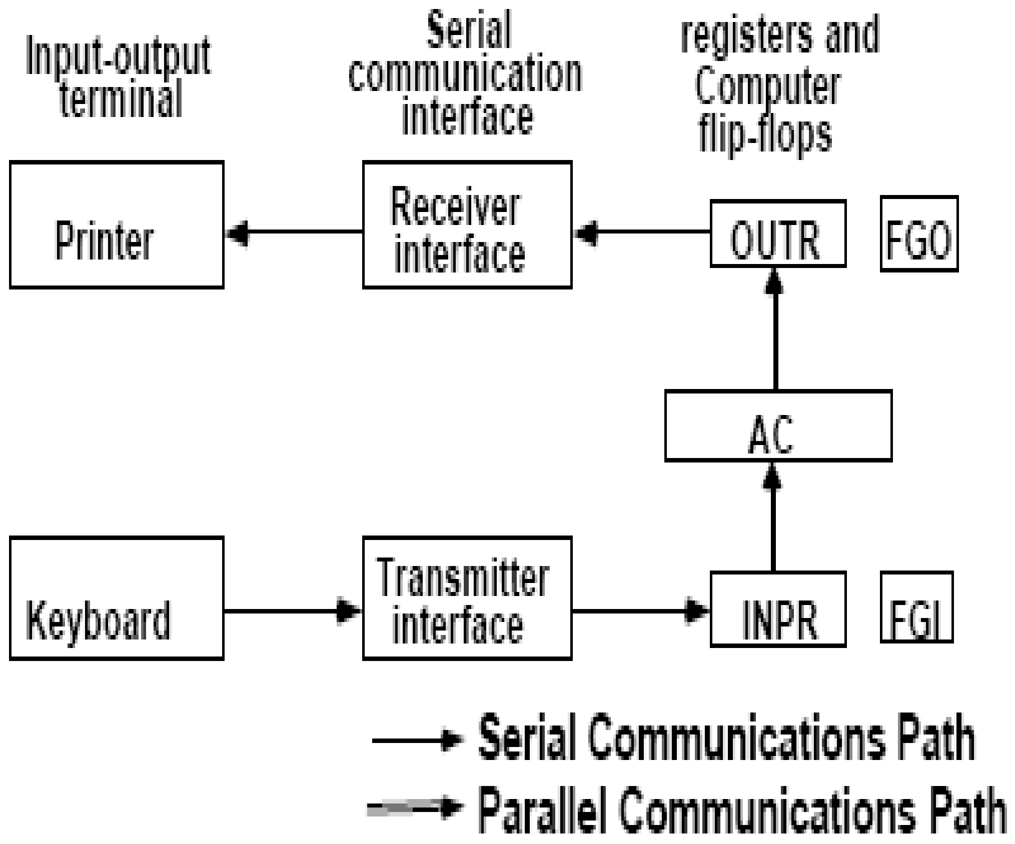
Materi : Interrupt

Disusun oleh :
Handrey Verryano (0904102)
Imam Fachmi N
Fatah Ramadhan
Agustianto Nur Kusuma

INPUT-OUTPUT AND INTERRUPT (1)

- **A Terminal with a keyboard and a Printer**
- **The terminal sends and receives serial information**
- **The serial info. from the keyboard is shifted into INPR**
- **The serial info. for the printer is stored in the OUTR**
- **INPR and OUTR communicate with the terminal serially and with the AC in parallel.**
- **The flags are needed to synchronize the timing difference between I/O device and the computer**

INPUT-OUTPUT AND INTERRUPT (2)


Input-output terminal
Serial communication interface
registers and Computer flip-flops
Printer
Receiver interface
OUTR
FGO
AC
Keyboard
Transmitter interface
INPR
FGI
Serial Communications Path
Parallel Communications Path

INPR	Input register - 8 bits
OUTR	Output register - 8 bits
FGI	Input flag - 1 bit
FGO	Output flag - 1 bit
IEN	Interrupt enable - 1 bit

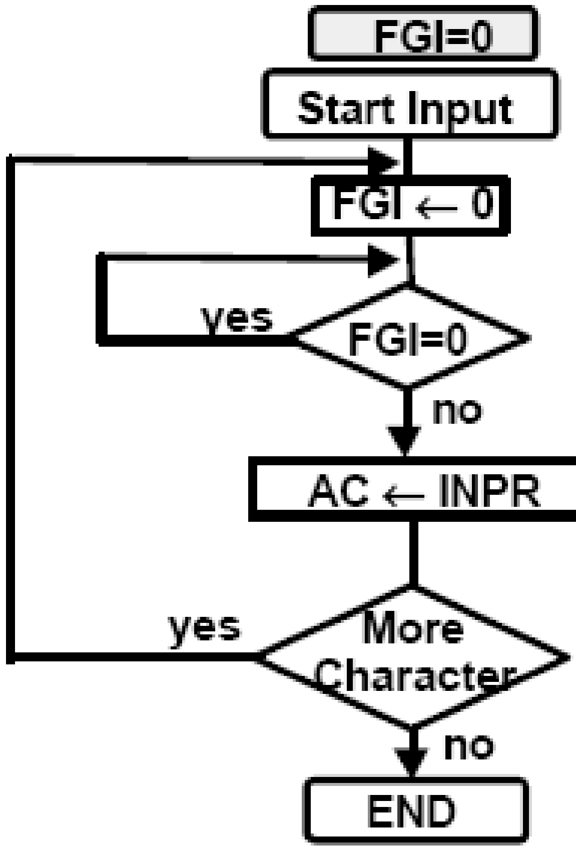
PROGRAM CONTROLLED DATA TRANSFER (1)

-- 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**

PROGRAM CONTROLLED DATA TRANSFER (2)

-- CPU --



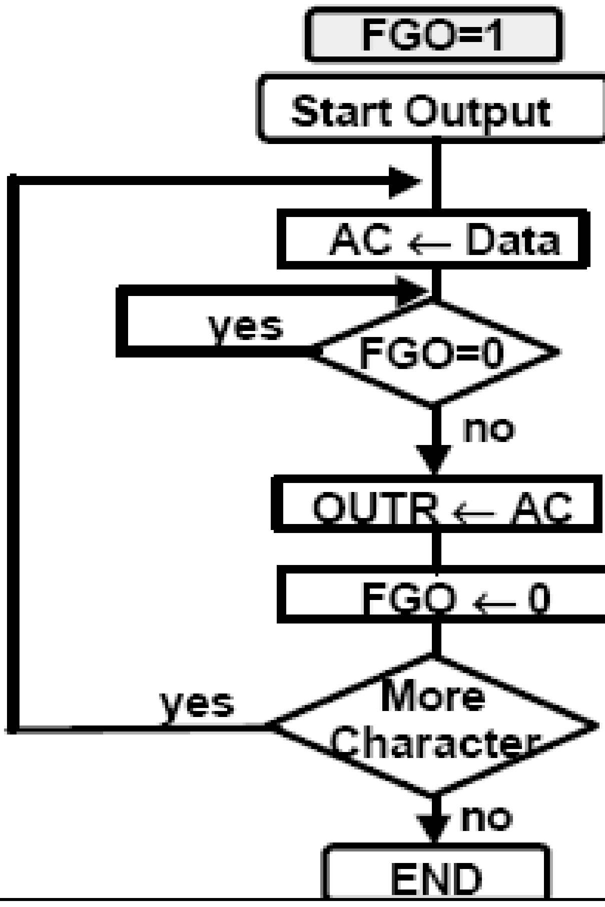
PROGRAM CONTROLLED DATA TRANSFER (3)

-- I/O Device --

- **loop: If FGI = 1 goto loop**
  - **INPR $\leftarrow$ new data, FGI $\leftarrow$ 1**
- **loop: If FGO = 1 goto loop**
  - **consume OUTR, FGO $\leftarrow$ 1**

PROGRAM CONTROLLED DATA TRANSFER (4)

-- I/O Device --



INPUT-OUTPUT INSTRUCTIONS

- **D7IT3 = p**
- **IR(i) = Bi, i = 6, … , 11**

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**
  - ***I/O and Interrupt***

- **Input**

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

- **Output**

```
– LOOP,     LD DATA
– LOP,      SKO DEV
            BUN LOP
            OUT DEV
```

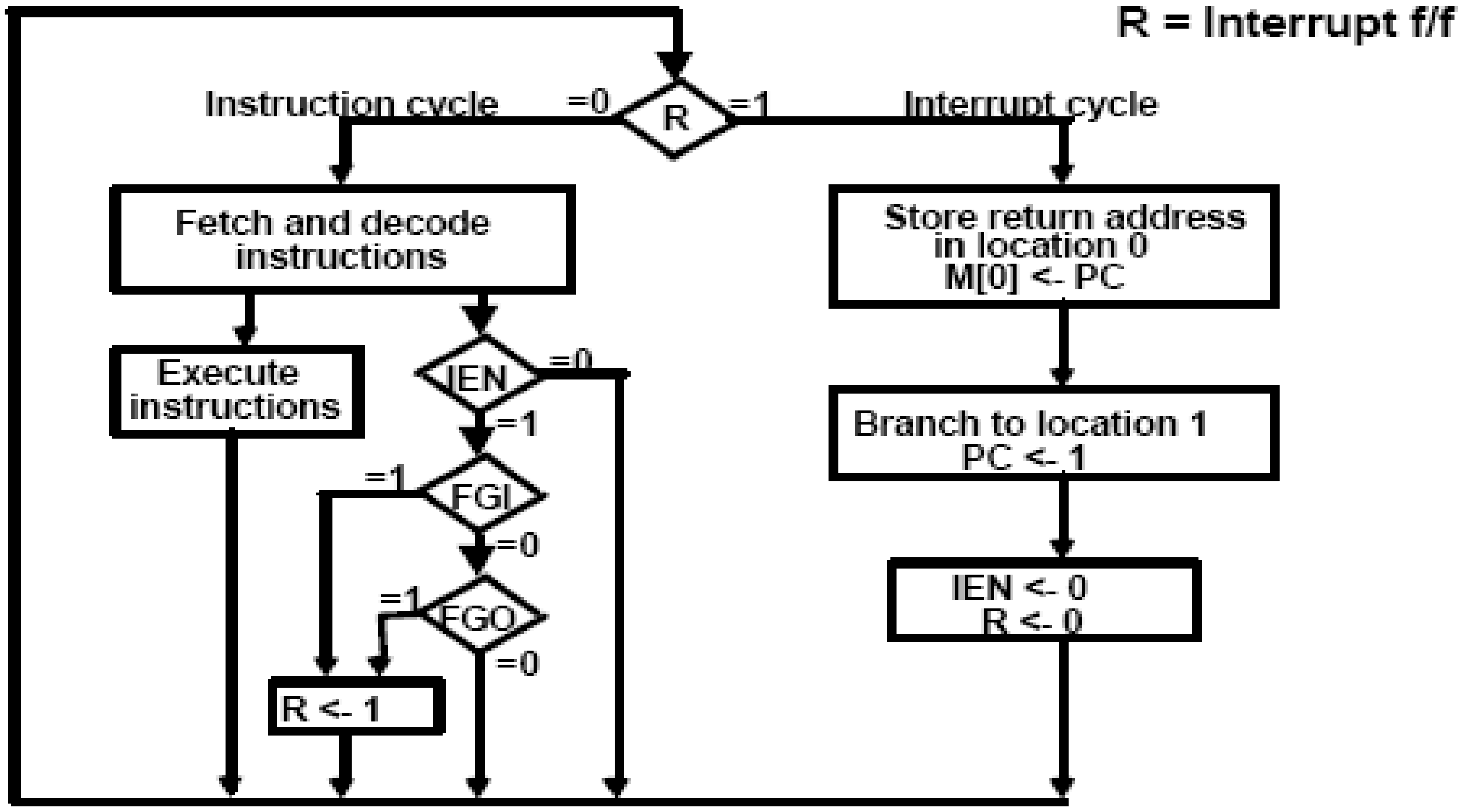
INTERRUPT INITIATED INPUT/OUTPUT

- **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 finds 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 (1)

- **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"**

FLOWCHART FOR INTERRUPT CYCLE (2)

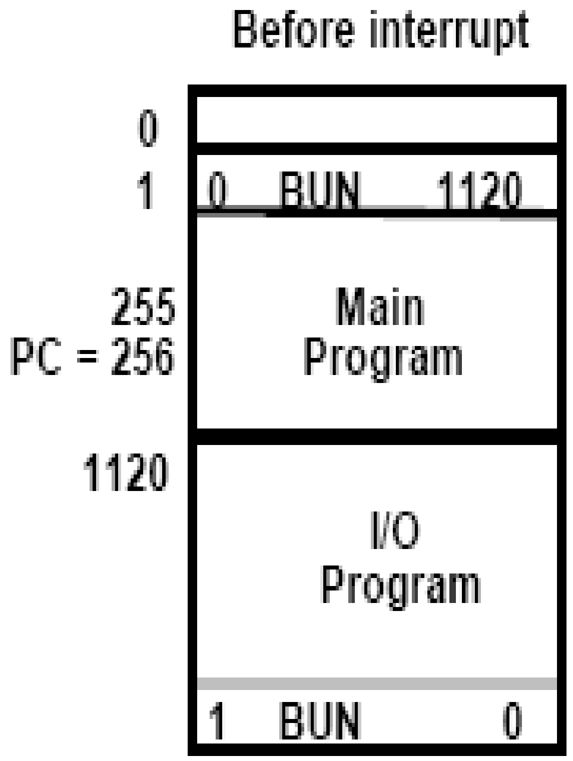


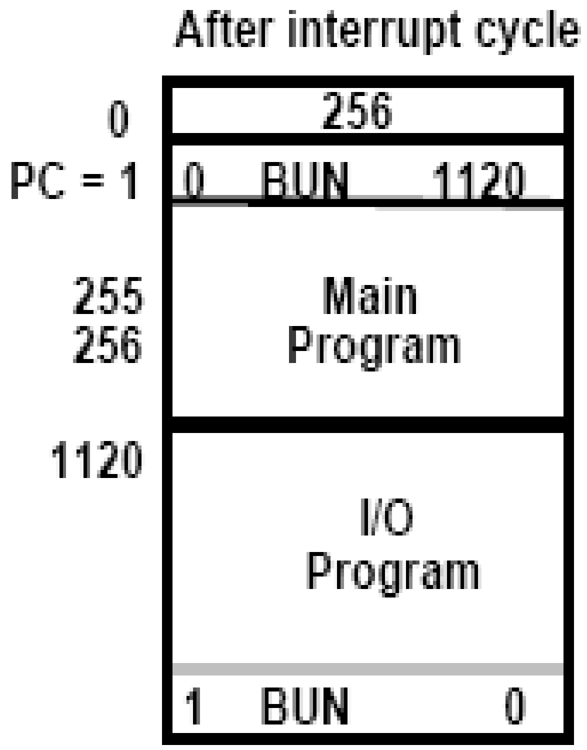
REGISTER TRANSFER OPERATIONS IN INTERRUPT CYCLE (1)

- **R F/F $\leftarrow$ 1**
  if IEN (FGI + FGO)T0’T1’T2’
  Then T0’T1’T2’ (IEN)(FGI + FGO): R $\leftarrow$ 1
- **The fetch and decode phases of the instruction cycle must be modified:**
  - **Replace T0, T1, T2 with R'T0, R'T1, R'T2**
- **- The interrupt cycle :**
  - **RT0: AR $\leftarrow$ 0, TR ¬ PC**
  - **RT1: M[AR] $\leftarrow$ TR, PC $\leftarrow$ 0**
  - **RT2: PC $\leftarrow$ PC + 1, IEN $\leftarrow$ 0, R $\leftarrow$ 0, SC $\leftarrow$ 0**

REGISTER TRANSFER OPERATIONS IN INTERRUPT CYCLE (2)

- **Memory**





Interrupts

- An interruption of the normal sequence of execution
- Improves processing efficiency
- Allows the processor to execute other instructions while an I/O operation is in progress
- A suspension of a process caused by an event external to that process and performed in such a way that the process can be resumed

Classes of Interrupts

- Program
  - arithmetic overflow
  - division by zero
  - execute illegal instruction
  - reference outside user’s memory space
- Timer
- I/O
- Hardware failure

Interrupt Handler

- A program that determines nature of the interrupt and performs whatever actions are needed
- Control is transferred to this program
- Generally part of the operating system

Interrupt Cycle

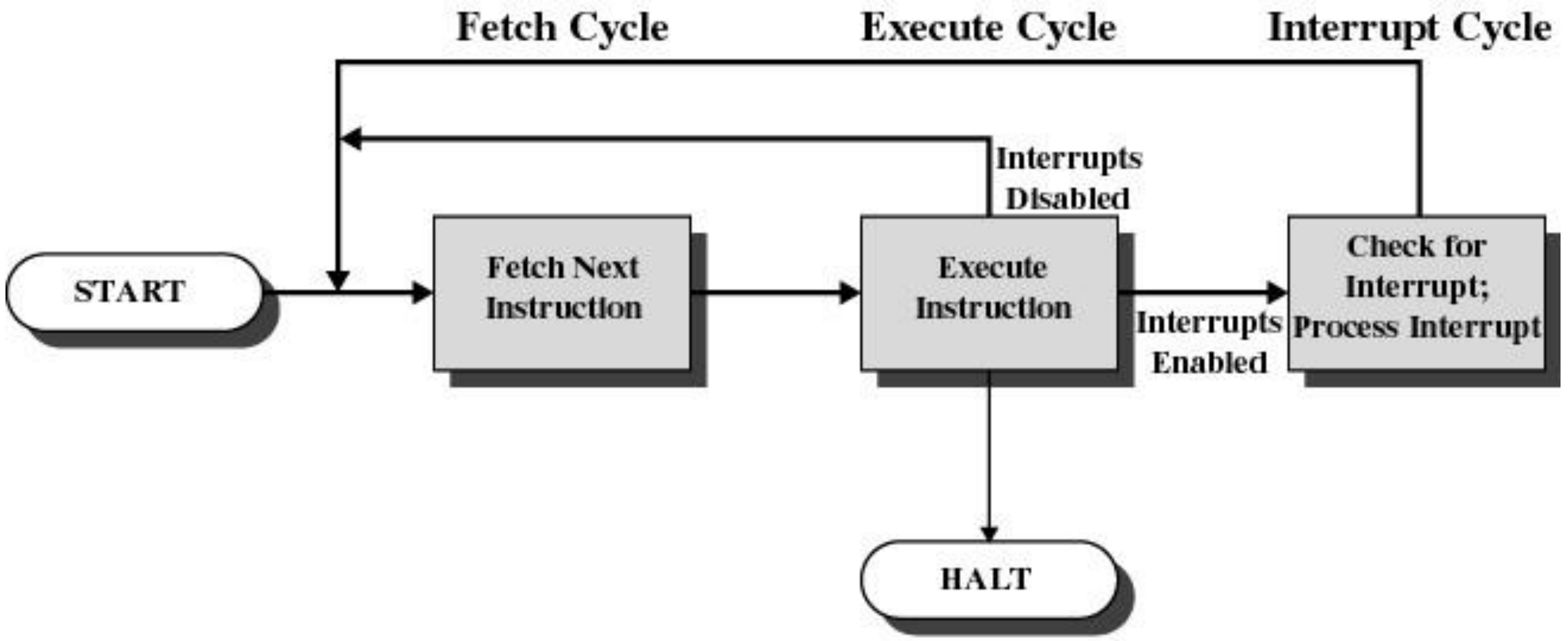


Figure 1.7 Instruction Cycle with Interrupts

Interrupt Cycle

- Processor checks for interrupts
- If no interrupts fetch the next instruction for the current program
- If an interrupt is pending, suspend execution of the current program, and execute the interrupt handler

Computer - I/O Module (1)

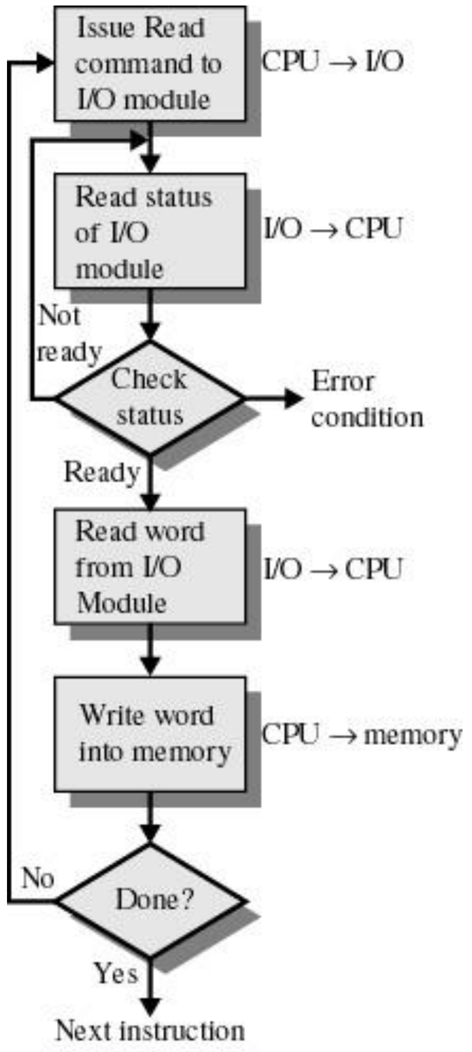
- Classification based on data stream.
  - *Block Oriented Device*
    - Information saved as fixed sized block.
    - Write using Direct access method.
    - Example : disk, optical disk, tape, etc.
  - *Character Stream Oriented Device*
    - *Information saved using character stream.*
    - Example : terminal, printer line, network interface.

Computer - I/O Module (2)

- Three techniques I/O devices connection:
  - *Programmed I/O*
  - *Interrupt Driven I/O*
  - *Direct Memory Access*

Programmed I/O

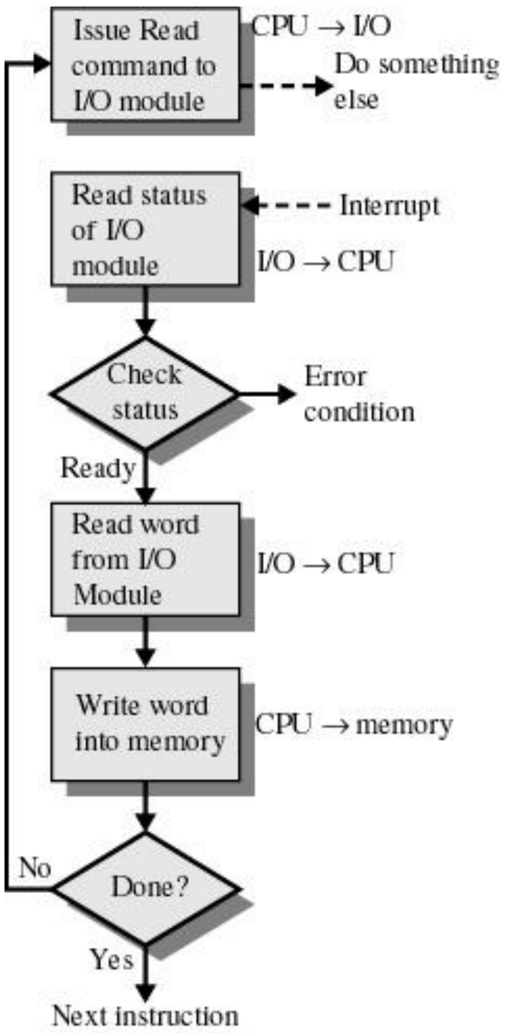
- I/O module performs the action, not the processor
- Sets appropriate bits in the I/O status register
- No interrupts occur
- Processor checks status until operation is complete



(a) Programmed I/O

Interrupt-Driven I/O

- Processor is interrupted when I/O module ready to exchange data
- Processor is free to do other work
- No needless waiting
- Consumes a lot of processor time because every word read or written passes through the processor



(b) Interrupt-driven I/O

Referensi

- Slide Presentasi Organisasi Komputer oleh **Enjang Ali Nurdin, M.Kom**
- Slide Presentasi Sistem Komputer oleh Eddy Prasetyo Nugroho