Thursday, 17 October 2024 14:12

Operating System

← Manages input-output operations and establishes communication services between the user and device

- Exploit the hardware resources of 1 or more processors
- Provides a set of services to system users
- Manages secondary memory n I/O services

BASIC ELEMENTS

① MAIN MEMORY

→ Volatile ← A computer storage that temporarily holds data being used or processed

→ Referred to as real memory or primal memory

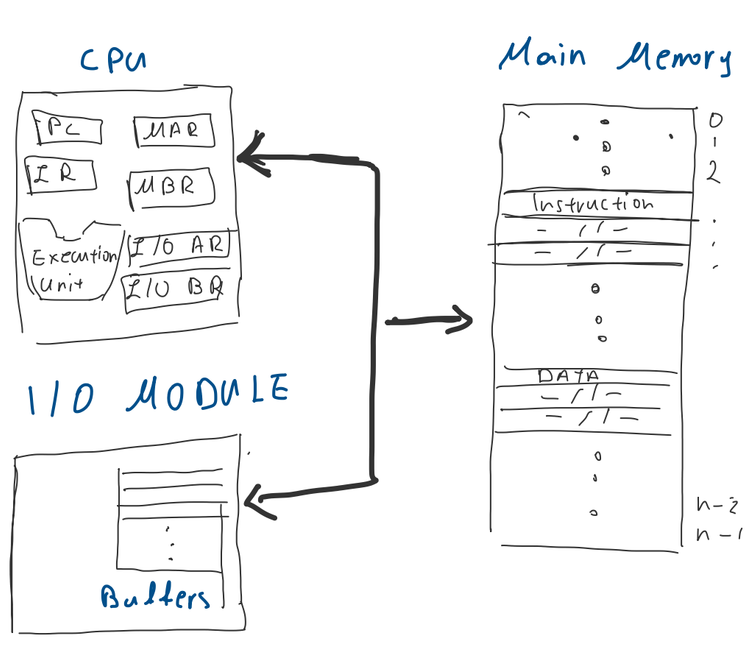
② I/O MODULES

- Secondary Memory Devices
- Communications equipment
- Terminals ← Text based interface to the computer

③ SYSTEM BUS

- Communicatio among processors, main memory and I/O modules

COMPUTER COMPONENTS: TOP-LVL VIEW


CPU
PC
MAR
IR
MBR
Execution Unit
I/O AR
I/O BR
I/O MODULE
Buffers
Main Memory
0
1
2
Instruction
DATA
n-2
n-1

- Memory Address Register (possess memory address of data that needs to be fetched and stored)
- Memory Buffer Register: The register in a computer's CPU that stores the data being transferred to n' from the immediate access storage.
- I/O AR : I/O Address Register is used to specify the address of a particular I/O device.
- I/O Buffer Register

Do we know what a buffer is thoug

Answer: A buffer is a region of memory used to store data temporarily while it is being moved from 1 place to another

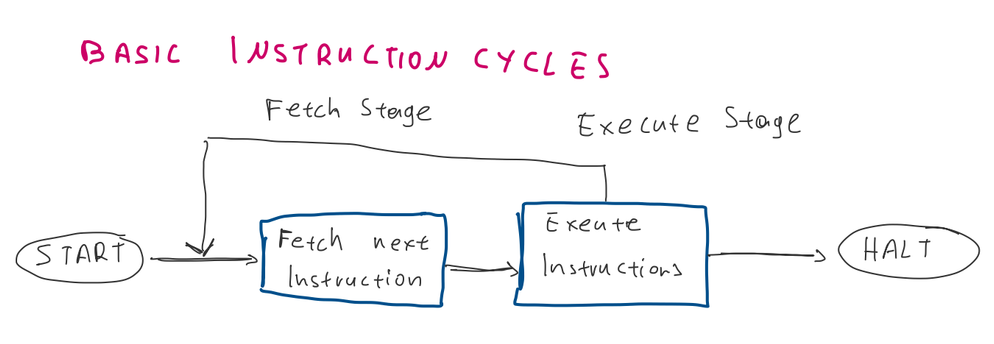
INSTRUCTION EXECUTION

2 Steps

① Processor reads (fetches) instructions from memory

② Processor executes ∀ instruction

BASIC INSTRUCTION CYCLES


Fetch Stage
Execute Stage
START
Fetch next Instruction
Execute Instructions
HALT

INTERRUPTS

- Interrupt the normal sequencing of the processor
- Most I/O devices are slower than the processor
  - Processor must pause to wait for device

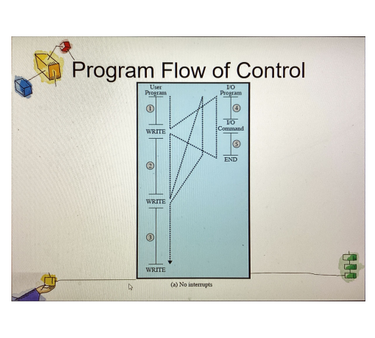
Classes of interrupts

① . **Program** Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by 0, attempt to execute an illegal machine instruction and reference outside a user's allowed memory space.

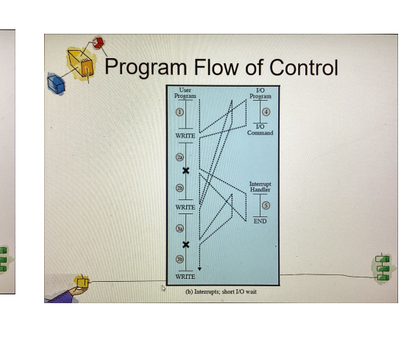
② **Timer** Generated by a timer within a processor. This allows the operating system to perform certain functions on a regular basis

③ **I/O** Generated by an I/O controller, to signal normal completion of a operation or to signal a variety of error conditions

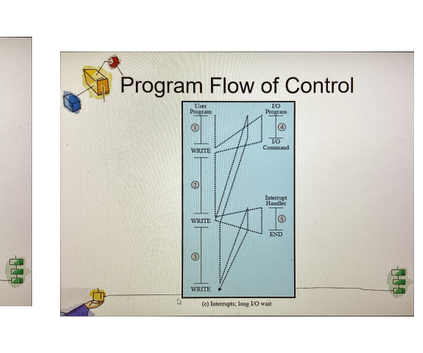
④ **Hardware failure:** Generated by a failure, such as power failure or memory parity error


Program Flow of Control

(a) No interrupts


Program Flow of Control

(b) Interrupts; short I/O wait

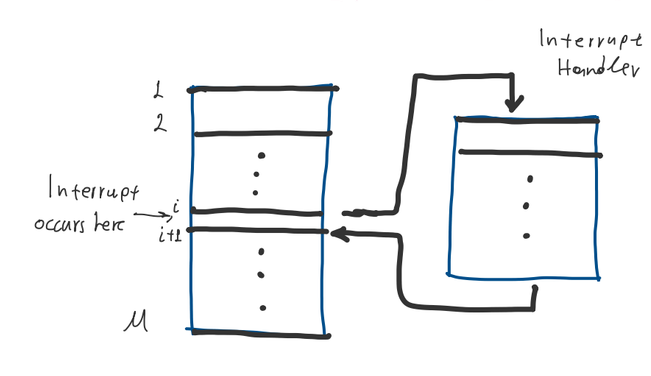

Program Flow of Control

(c) Interrupts; long I/O wait

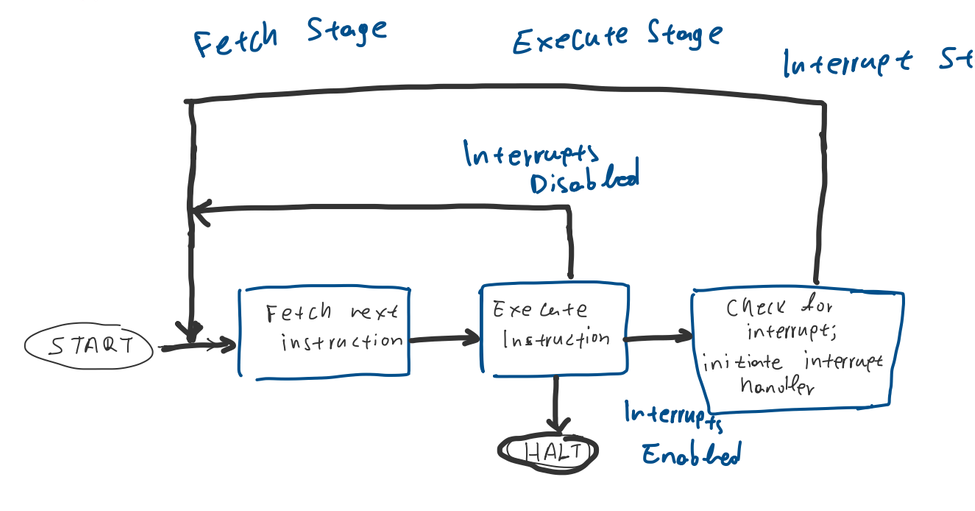
INTERRUPT STAGE

- Processor checks for interrupts
- if interrupt
  - → Suspend execution of program
  - → Execute interrupt-handler routine

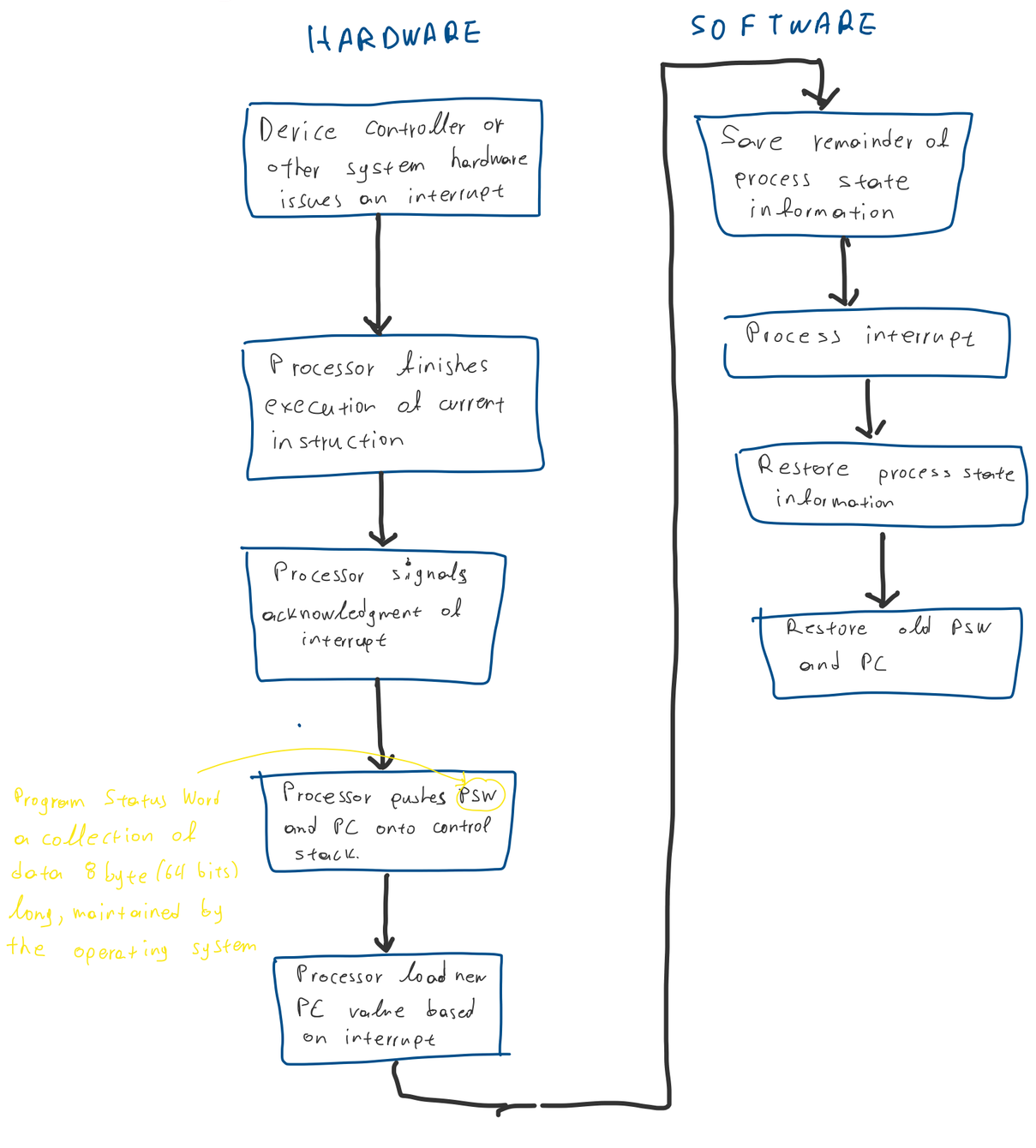
TRANSFER OF CONTROL VIA INTERRUPTS


Interrupt Handler
1
2
Interrupt occurs here
i
i+1
M

INSTRUCTION CYCLE WITH INTERRUPTS

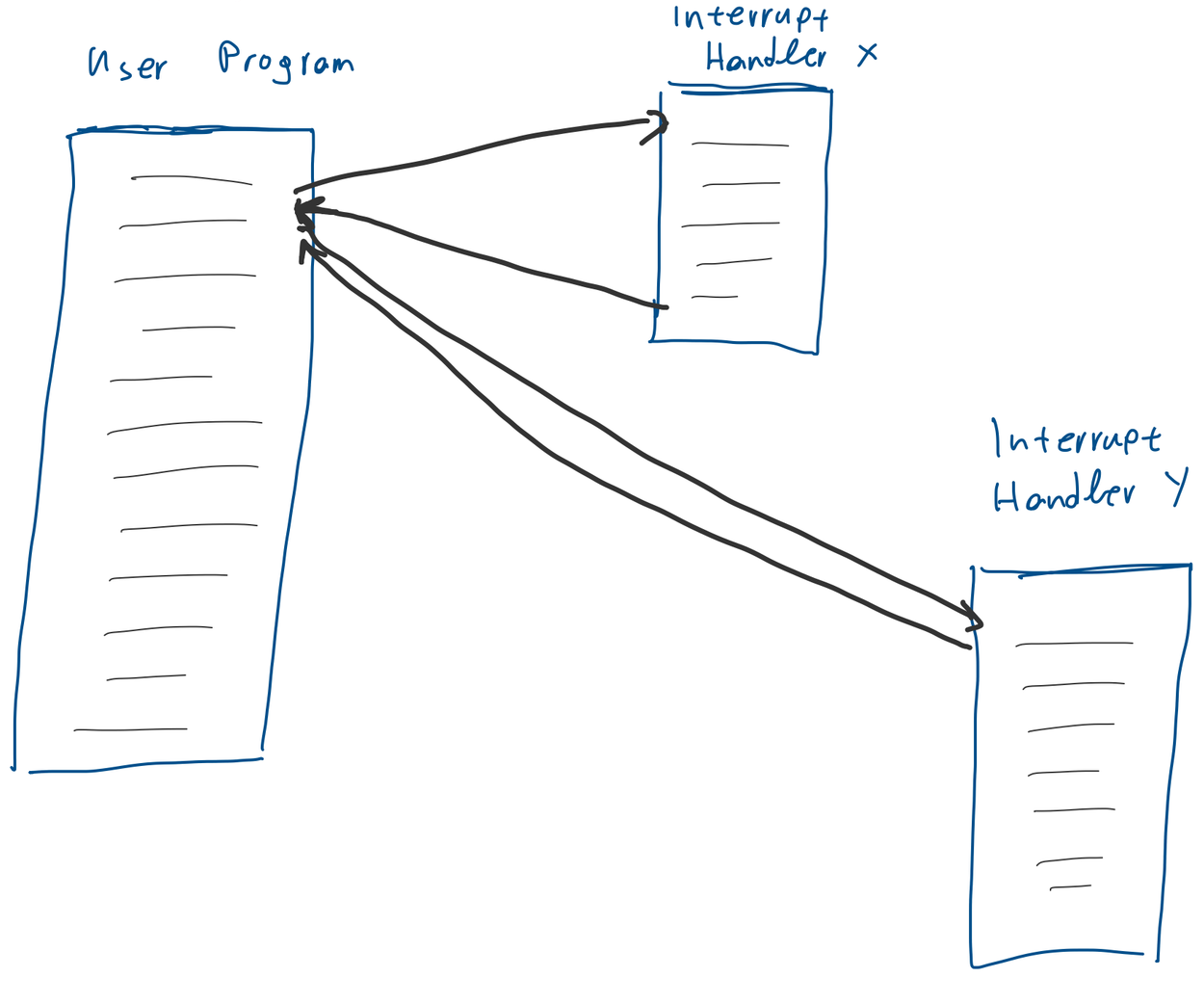

Fetch Stage
Execute Stage
Interrupt Stage
Interrupts Disabled
START
Fetch next instruction
Execute Instruction
Check for interrupts; initiate interrupt handler
HALT
Interrupts Enabled

SIMPLE INTERRUPT PROCESSING


HARDWARE
SOFTWARE
Device Controller or other system hardware issues an interrupt
Processor finishes execution of current instruction
Processor signals acknowledgment of interrupt
Processor pushes PSW and PC onto control stack.
Processor load new PC value based on interrupt
Save remainder of process state information
Process interrupt
Restore process state information
Restore old PSW and PC

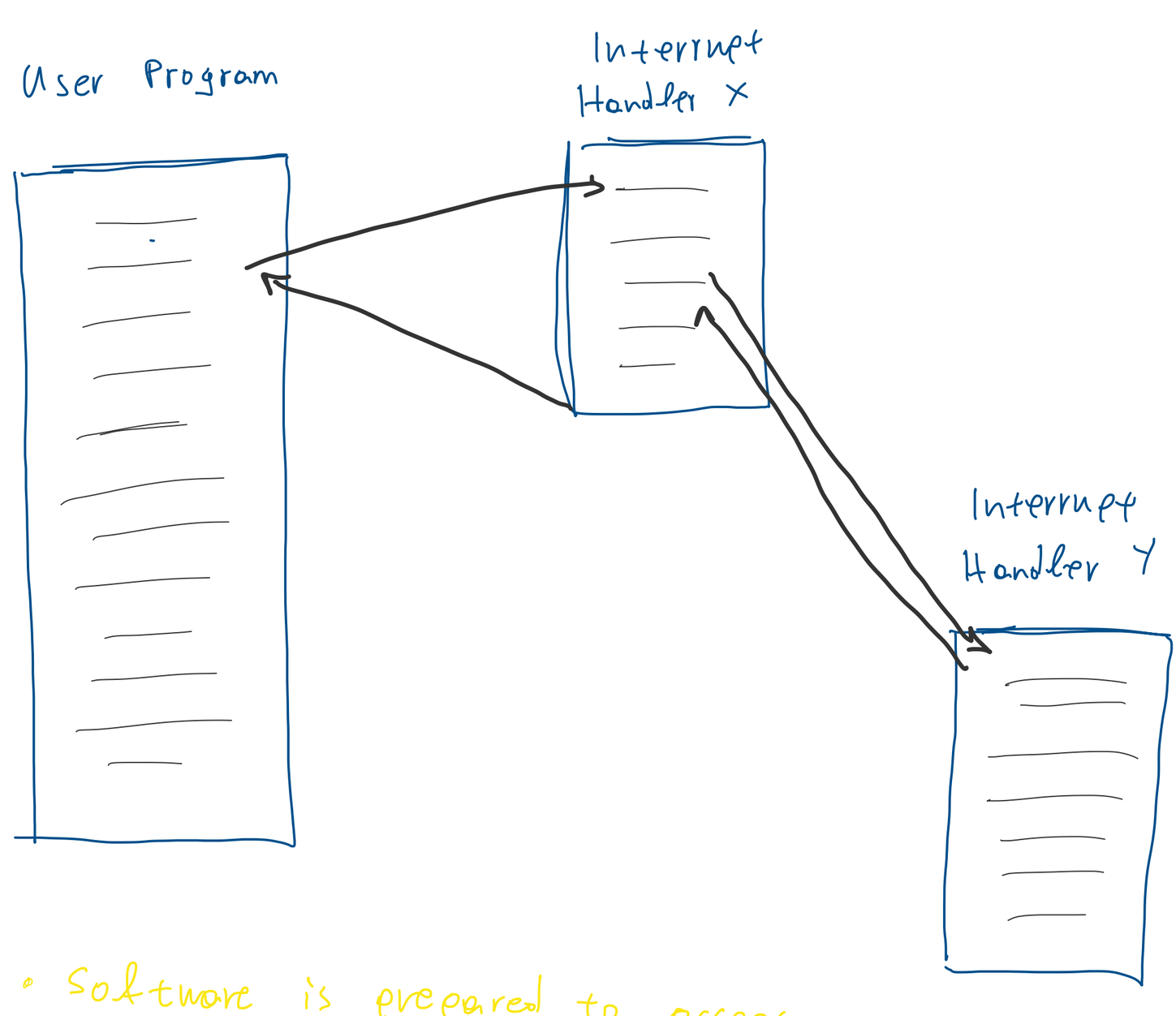
Program Status Word a collection of data 8 byte (64 bits) long, maintained by the operating system

SEQUENTIAL INTERRUPT PROCESSING


User Program
Interrupt Handler X
Interrupt Handler Y

- All interrupts will be processed sequentially

NESTED INTERRUPT PROCESSING


User Program
Interrupt Handler X
Interrupt Handler Y

- Software is prepared to accept another interrupt, even before it finishes handling the current interrupt

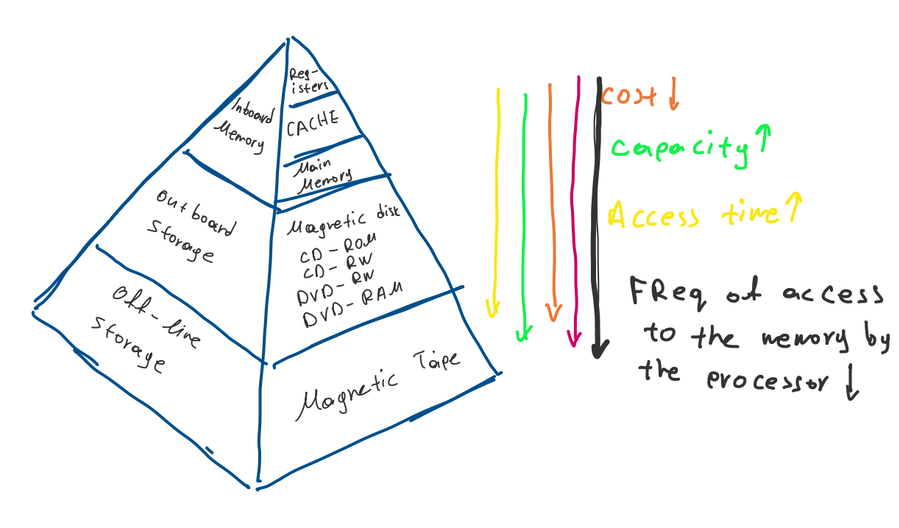
MULTIPROGRAMMING

- Processor has more than 1 program to execute
- The sequence in which programs are executed depend on their relative priority and whether they are waiting for I/O
- After an interrupt handler completes, control may not return to the program that was executing at the time of the interrupt.

MEMORY HIERARCHY

- Faster access time, greater cost per bit
- Greater capacity, smaller cost per bit
- Greater capacity, slower access speed

Memory hierarchy


Registers
CACHE
Main Memory
Inboard Memory
Magnetic disk
CD-ROM
CD-RW
DVD-RW
DVD-RAM
Outboard Storage
Magnetic Tape
Off-line Storage
cost ↓
capacity ↑
Access time ↑
FReq of access to the memory by the processor ↓

GOING DOWN THE HIERARCHY

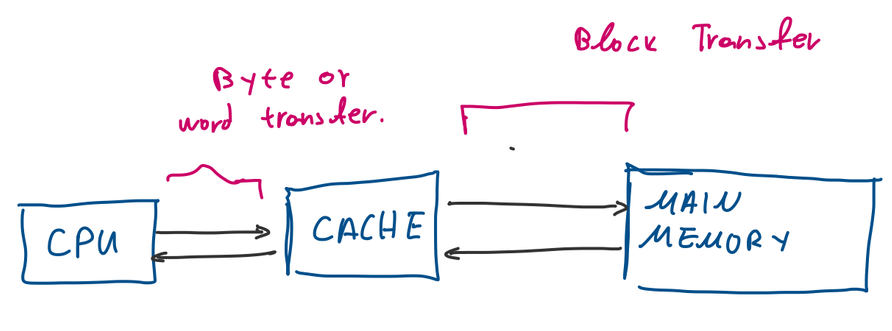
- Decreasing cost per bit
- Increasing capacity
- Increasing access time.
- Decreasing frequency of access to the memory by the processor

SECONDARY MEMORY

- Auxiliary Memory
- External
- Non volatile
- Used to store program and data files

CACHE MEMORY

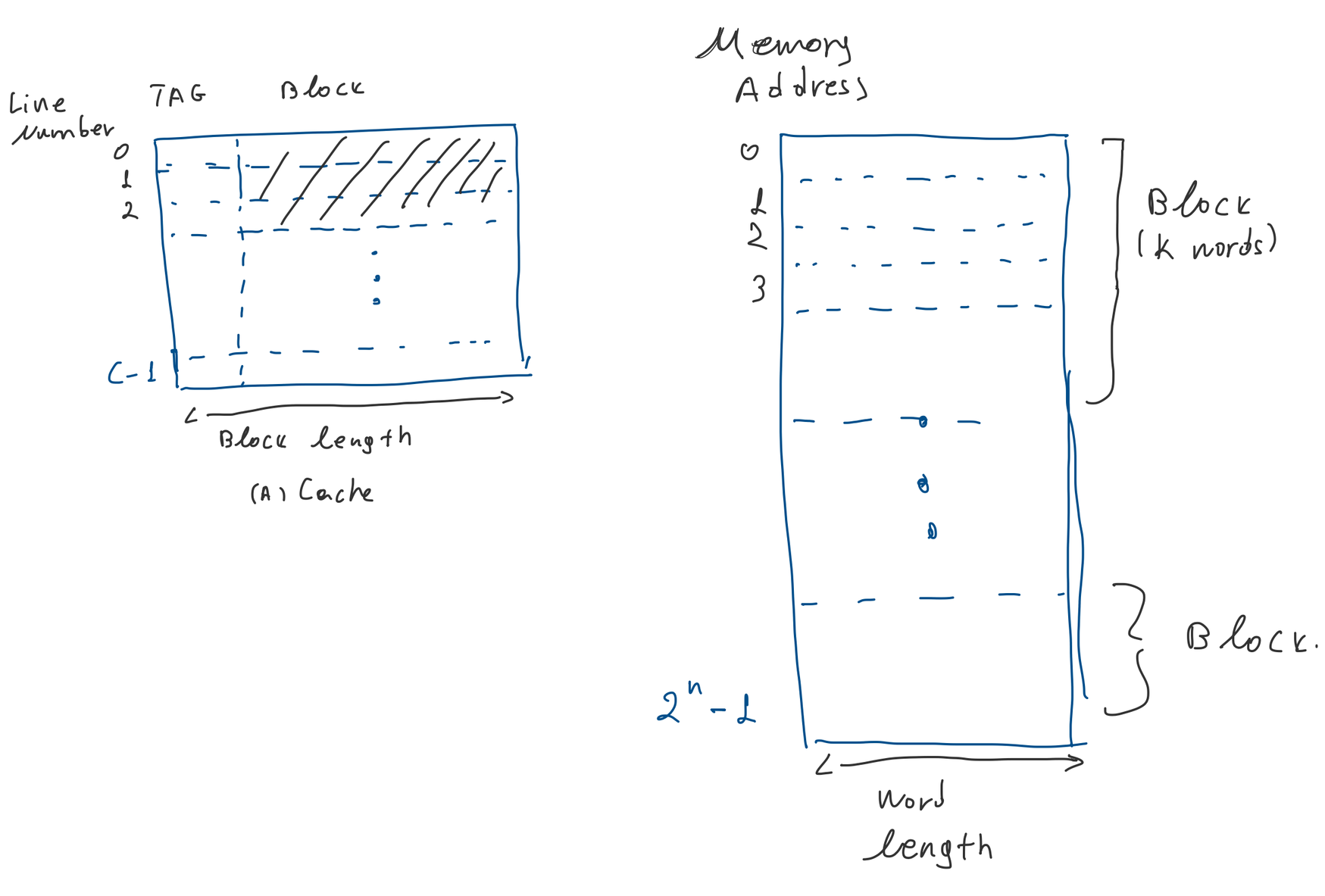
- Processor speed faster than memory access speed
- Exploit the principle of locality with a small fast memory


Byte or word transfer.
Block Transfer
CPU
CACHE
MAIN MEMORY

CACHE PRINCIPLES

- Contains copy of a portion of main memory
- Processor first checks cache
- if desired data item not found, relevant block of memory read into cache.
- Because of locality of reference, it is likely that future memory references are in that block

CACHE/MAIN-MEMORY STRUCTURE


Line Number
TAG
Block
0
1
2
C-1
Block length
Memory Address
0
1
2
3
Block (K words)
Block.
$2^n - 1$
Word length

(A) Cache

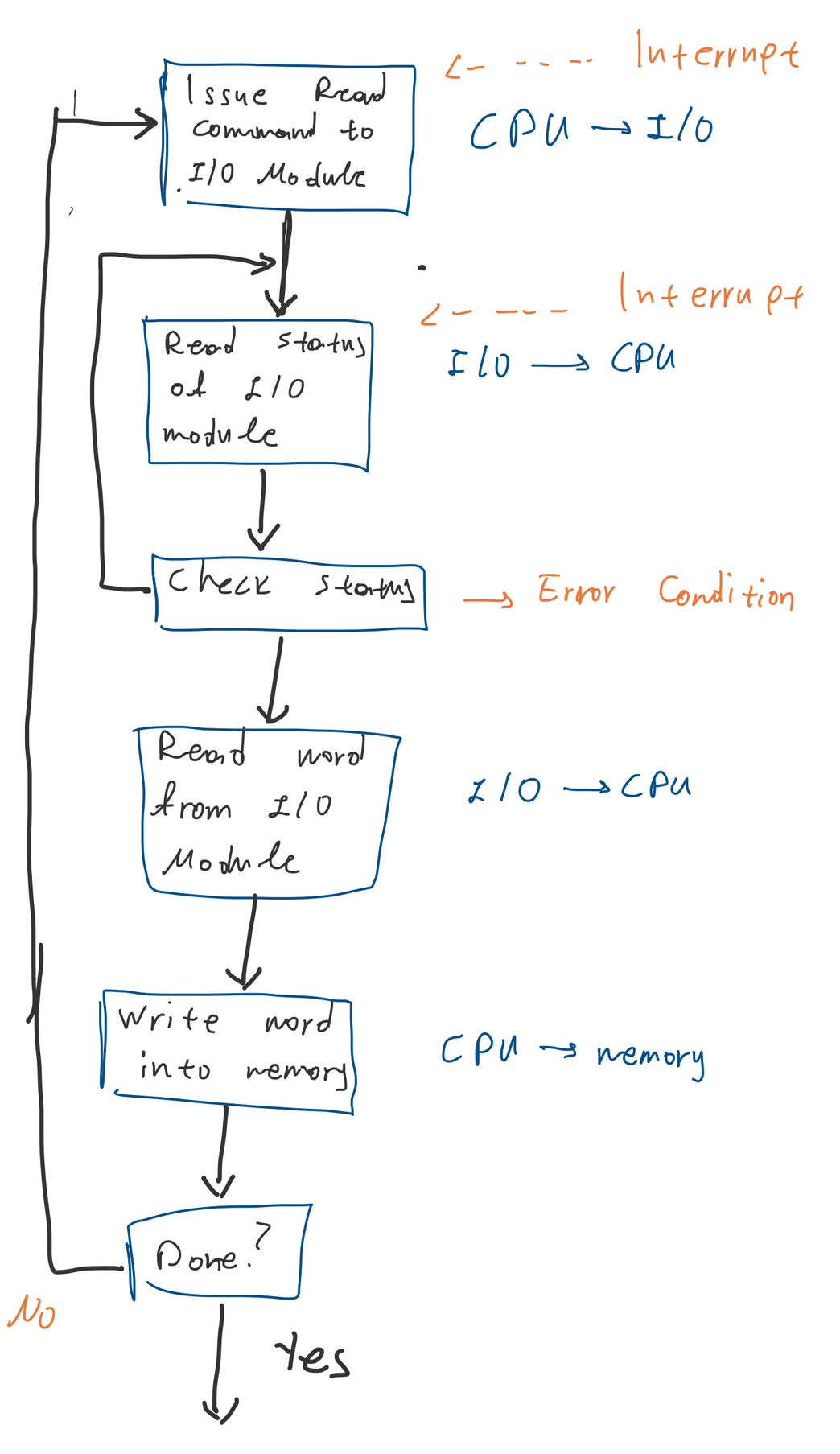
(b) Main Memory

∘ PROGRAMMED I/O

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

Interrupt Driven - I/O

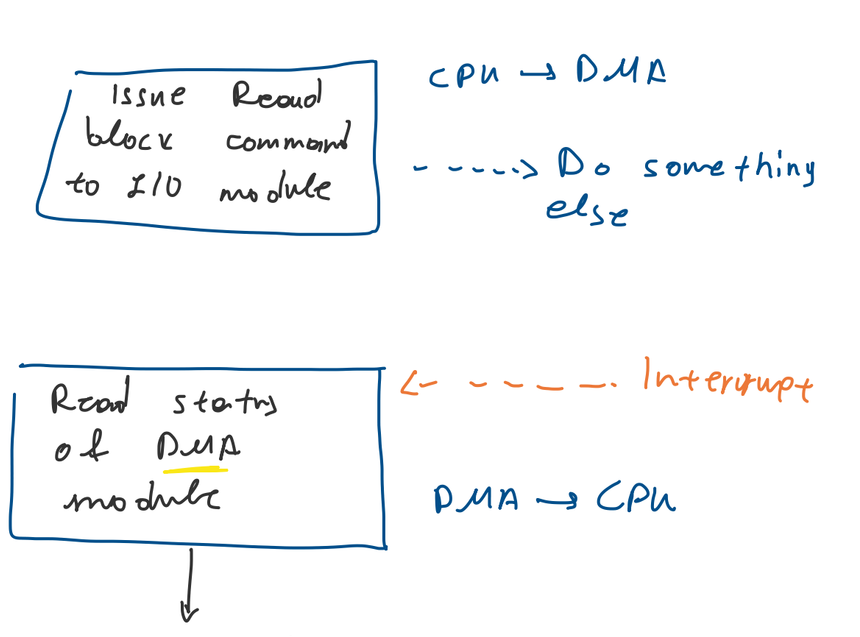
- No needless waiting
- Consumes a lot of processor time because every word read or written passes through the processor


Issue Read command to I/O Module
← - - - Interrupt
CPU → I/O
Read status of I/O module
← - - - Interrupt
I/O → CPU
Check status
→ Error Condition
Read word from I/O Module
I/O → CPU
Write word into memory
CPU → memory
Done?
No
Yes

a Programmed I/O

∘ DIRECT MEMORY ACCESS

- Transfers a block of data directly to or from memory
- An interrupt is sent when the transfer is complete
- More efficient


Issue Read block command to I/O module
CPU → DMA
- - - -> Do something else
Read status of DMA module
← - - - - Interrupt
DMA → CPU
Next instruction

(c) Direct Memory Access