Comp Org Overview

- 4 subsystems
  i) Processor-core
  ii) Signal Chain
  iii) Memory
  iv) Comms and UI

- Signal Chain

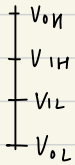
  i) ADC - DAC (+filter)
  - nyquist : sample freq $\geq 2^*$ (signal freq)

  ii) Interfaces
  - serial / parallel bits + single / bi direction
  - input / output / I & O
  - compatible if electrically + communication protocal

$V_{OH} \geq V_{IH}$, $V_{OL} \leq V_{IH}$

o/i : output/input ; H/L : high/low

min max



- differential signals have better noise tolerance and can be clocked at higher frequencies
  → greater margin
  → external interference cancellation
  → difference b/w V+ & V- remains constant

- communications protocal : no of bits , type of synchronization
  types of data + formatting

- Parallel Data transfer
  - → synchronous
  - → preffered when sustained high speed is needed
  - → prone to signal skew + crosstalk ← limit max clock freq
    - ↓ temp, capacitance, printed circuit board dimensions
    - ↘ frequency, number of lines (ground plane helps!)
  - → needs only strobe signal, simpler hw interface

- Serial data transfer
  - → more complex hw interface, slower
  - → prefered for long distances
  - → simplex (1D), half duplex (2-D but 1 at a time), full duplex
  - → sync (common clock signal); asyn (both have clock signals not the same, pre-fixed clock freq is used)
  - → sync serial data transfer
    - · master/slave · master recieves data on posedge, passes on
      - ↘ provides clock | negedge, MSB first;
    - · SPI : serial peripheral interface : synchronous
      - > slave select low ⇒ start transfer
      - > MOSI / MISO used depending on direction of transfer
      - > multiple slaves allowed
  - → async serial data transfer
    - · sync words are used,
      possible skew as 2 separate clocks

- UART : universal asynchronous receiver transmitter

transfer
- > idle : logic 1
- > start : logic 0
- > baud rate : no of signal state changes / second
- > parity bit
- > stop : logic 1
- > LSB first : waveform will have MSB first

receiver
- > internal clock runs at k* baud rate (k=2,4 etc) so that sampling is at midpoint of data bit

- Data transfer Mechanism

i) Polling / Programmed I/O
- CPU uses 100% of resources to poll I/O port continuously once control reg for data transfer have been initialised.
- inefficient use of CPU ; very simple SW implementation

ii) Interrupt
- signal (Interrupt request) $\longrightarrow$ CPU $\xrightarrow{\text{yes}}$ Interrupt Service routine (ISR)
  - interrupt latency delay
- uses a vector table w/ indexes corresponding to IR's and a col w/ ISRs
- complex hw, time efficient, allows prioritization

iii) DMA (Direct Memory Access)
- controls Data transfer, shares same bus as CPU
- mem - mem , I/O peri - I/O peri , mem - I/O peri
- gen address, read write ops
- fetches data to DMAC buffer , then deposits it (src → DMAC ; DMAC → dest)
- issues DMA Interrupt to CPU after data transfer to inform that data is ready to be processed.
- good for specific formats

- Burst mode
  - transfers multiple units, suspends CPU for a while, fast

- Cycle stealing
  - one unit at a time can occur b/w CPU instructions/pipeline stages
  - slow, CPU is freer, preferred in security systems
  - transparent mode
    - transfer only when CPU is not using bus
    - slowest
    - least CPU disruptions
    - complex hw

backup storage : HDD not SSD
cost, ∞ read write

Memory

- CPU : reg, buffer, cache → SRAM
- int system → subset currently executed → DRAM, NOR
- storage : everything → NAND : SSD
- semi-conductor
  - SSD, SRAM, DRAM, Flash
  - Volatile
    - SRAM } XIP
    - DRAM } XIP
      - not exactly but they make it work
  - non volatile
    - EEPROM (larger blk faster cheaper)
    - Flash
      - NAND
      - NOR
- magnetic
  - HDD →

- faster, more expensive
- large : 6 transistors
  - → 2 : data flow
  - → 4 : store data
- low power consumptione
- power supply ⇒ data stored
- can access specific locations

- slower, cheaper
- smaller : 1 transistor : charge flow
  - 1 capacitor : holds charge
- 1 → charged ; 0 → discharge
- every read destroys data : rewrite is needed.
- periodic refrest, large power consumption

- all cells > V_T ⇒ bit line o/p = 0
- no XIP
- memory is accessed at page level
- less wires, used in mass storage
- combined I/O databus

- any cell has $V > V_T$ (logic 0) ⇒ bit line o/p = 0
- expensive as comp to NAND
- supports XIP
- can store program code
- random reading ✓
- erase at block lvl
- separate data, address lines

SSD

- flash
- limited read + write
- expensive
- longevity :
  - i) wear levelling
  - ii) external RAM buffer
  - iii) error correction code
- small & light as comp. to HDD and faster

- logic ≡ orientation of magnetic elements
- platter, track, sector, head = no of surface
- cylinder : track / surface
- $T_A = T_S + T_R$ (request to head = request to track + track to sector)
- T_T : after head in position to transfer data
- $T_R = \frac{0.5}{RPS}$; $T_T = \underbrace{\frac{N}{D_S \cdot D_T}}_{\text{no of track to access}} \times \frac{1}{RPS}$ (D_T : sector / track, D_S : byte / sector)
- disk defragmentation
- zone bit recording format
- LBA : logical block addressing (2^50 bytes of data can be addressed)
- CHS : cylinder, head, sector
- ↑↑ are addressing schemes

performance : storage, high RPM, short seek time

FGT : floating gate transistor

- mosfet
- 2 states : Erased → logic 1 (T : on)
  - Programmed → 0 (T : off)
- to program : large +ve voltage
- check state : inject voltage b/w
  - $V_1 < V < V_T$ ← increased threshold post prog
  - if ON ⇒ erased
  - if OFF ⇒ programmed

Cache

- fast go-b/w between CPU & system

- small parts of memory are transferred to the cache based on:
  i) Principle of locality
     ↳ locality of space (sequential) (local)
     ↳ locality of time (loop) (temporal)

- cache mapping : direct mapped

> main memory address $\xrightarrow{\text{conv}}$ cache add. | tag | blk | Offset |
  ↳ same no of bits

> offset bits = $\log_2$ (size of block) [size of cache block = size of MM block]
> block bits = $\log_2$ (size of MM / (size of cache * blocksize))
> tag bits = whatever is left.

> tag values are compared to check if the memory is correct

- cache memory replacement
  i) LRU (least recently used) (slows down cache)
  ii) FIFO (one that has been there longest) (doesn't take reusability into account)

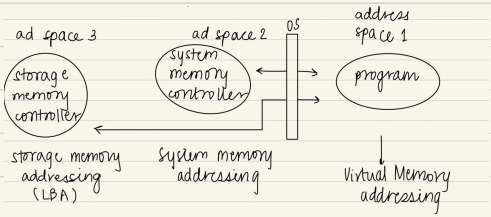
- write policy
  i) write through : CPU writes in cache & main memory (bus crowding) (system)
  ii) write back : memory is only updated when block is selected for replacement
     ↑ cache coherency issue

- write miss : when block that CPU wants to update is not in the cache
  i) write allocate : get block to cache
  ii) write no-allocate : write directly in main memory

- effective access time : $EAT = H * Access_{cache} + (1-H) * (Access_{cache} + Access_{mm})$

Virtual Memory

- address translation is done by the OS



- Virutal address : address used by prog, generated by compiler

- VM can be contiguous

- address mapping : paging

VM size > memory size → the word size is same

VM	P/M/S M	valid
page	frame	1 0

page table
↳ in MM

offset is the same

no of bits det by size

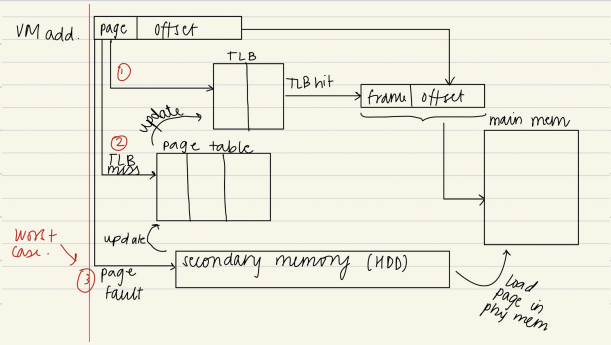
VM page #	off
physical frame #	off
tag	block	off

① VM → MM add
② CPU → cache hit/miss
③ hit → CPU takes info from cache
miss → cache manages transfers info from system/storage

TLB : translation lookaside buffer → in processor
↳ SRAM

↓

subset of page table

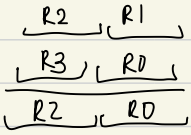


w12

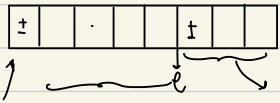
- unsigned number : carry = 1 ⇒ overflow
  overflow ⇒ nothing
- signed number : overflow = 1 ⇒ overflow
  carry ⇒ not indicative of much

} error

- multibit arith : ADD R0, R0, R1
  ADC R2, R2, R3
  ↳ carry bit added



- 2's comp range : $-2^{N-1} \rightarrow 2^{N-1} - 1$
- fixed-point representation : range vs precision
  ↓
  resolution
- floating point :

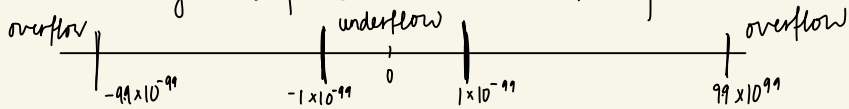


sign — mantissa (base val) — exponent (position of radix)

range : $(-9.99 * 10^{+99} \rightarrow 9.99 * 10^{99})$

size of exp : range
size mantissa & value of exponent : precision

std : 1 digit before radix (binary : 1)



IEEE 754

	S	← 8 →	← 23 →	
32 bits :	I	exp · E	frac F	b = 127 float c

	← 11 →	← 52 →	
64 bits :	S	exp E	frac F	b = 1023 double c

S : 0 → +ve
1 → −ve

Exponent = E − Bias (E − b)

F : 1 · F
↳ only frac starting from $2^{-1} \ldots 2^{-23/52}$

$(1)^S \times (1 + f_1 2^{-1} + f_2 2^{-2} + \ldots) \times 2^{E-b}$

mantissa = 1 · F

E is a positive value, bias allows it to be signed.

8 bits ⇒ 0 → 255
−127 → 128 ?

normalised :	E :	1 → 254	1 · F : anything
denormalised	E :	00000000	0 · F : nonzero
zero	E :	00000000	F : 00000....000
infinity	E :	11111111	F : 0000...000
NAN	E :	11111111	F : non-zero

E = −126 ← not −127

32 bits

IEEE 754 (floating pt) → precision changes

- range : $-2^{*}2^{128} \rightarrow \sim 2^{128}$
- precision : less than 2^{-126}

fixed point → uniform precision

- range $\approx 2^{32}$ unsigned
- max precision $\approx 2^{-32}$

solns.

offset : 2^{10} :. 10 bits

0 11000000000

11

digital = $\frac{V}{V_{ref}} \times 2^8$

1.5

25

00000000

32

$V = \frac{3}{8} = 0.75$

$O/P = I/P \times \frac{2^n}{V_{ref}}$

2.5

10.1

1.01×2^1

$1 = E - 127$

$E = 128$