

Q1.1: What is a Computing system?

A device Capable of Performing mathematical and logical operations in accordance with a predetermined set of instructions, and it consists of 3 main Components:

- (1) Processor : performs The instructions & Calculations
- (2) Memory : storing The programs code & Data
- (3) I/O Peripherals : interacting with The user, enabling The system to Communicate with The outside world

⇒ Embedded system is a Computing system with limited resources (Processor, memory, I/O) used For performing a specific Task:

⇒ General purpose system
(Multi Task)

specific Purpose system
(Specific Task)

- | | |
|---|--|
| 1) designed to be Versatile and
Capable of Performing a wide range
of Tasks | 1) designed for a specific task
or set of tasks. |
| 2) not always required to operate
in real-time | 2) operates in real-time
ورجحه انه يتألم ببعض ادوات بالوقت
"Time-Critical" |

* Real Time: Correct function at The Correct Time

يعني ينجز المهام في الوقت المحدد يعني متلاً لو النظام بيأخذ وقت
مختلط او متساًع على حسبه في حاسع تشغله "load" يقاده دار
"load driver" حسب ال RT وينفذ المهام في وقت مختار

- | | |
|--|--|
| 3) Typically have access to
more resources and aren't as
Constrained by Power or memory
limitations | 3) Operates with limited
resources such as memory,
processing Power and energy |
|--|--|

⇒ Embedded System Challenges:

(-) Cost (-) Size (-) Performance (-) Power Consumption

⇒ Embedded system Implementation Techniques:

System on Board (Sb)

High Cost

large size

High Power Consumption

Moderate Performance

القابل للتعديل (Configurability) =
نظام كامل على لوحة واحدة (System on Board)

System on Chip (SoC)

low Cost

Small Size

Low Power Consumption

Moderate Performance

أقل قابلية للتعديل (Configurability) ←
نظام متكامل في جهاز واحد (SoC)

→ for development and design phase

→ for production Phase

⇒ Micro Controller

a Complete system on chip that
contains processor, memory and I/O

Micro Processor

one element from the needed
elements in order to function

used in ES with limited space
and power

used in larger and more complex
Computer systems

lower cost and easier to use

higher performance but more
complex to design

Robotics, automotive and industrial control
Used in applications such as desktop computers, servers and gaming consoles

8051, PIC, AVR

AMD Ryzen, ARM Cortex-A
and Intel Pentium

⇒ what is The difference between:

1) Processor:

(V.T) Vacuum Tubes كانت بتطلُّق زمان على أي مهالج بستمدة في طريق اد وعليه كانت كبيرة جتنا لدرجة صنَّى كامل بعده زمانه امناً لوقت بستمدة على كل اد Processors.

2) Microprocessor:

لما كان سيليكون - Industry (S.I.) و بعدها ~~الحاسوب~~ اد فعلى سلسلة Proessors الصغيرة الابدية وهي اد MP. بعدها مولنار لوقت كل بعها نفس المجم و مارش في اد آ. د. دول فبيحتش فارقة عادي

3) C.P.U (central processing unit):

ما يبقى من الـ Processor في أكثر من system بسيف في واحد هو الـ Master والمعالج = المذكورة ببساطة هي مثلاً الـ

(-) GPU: Graphic Processing Unit (-) DSP: digital signal processing

⇒ Processor in detail:

A Processor contains ALU, Control Unit (CU) and Register Files، وروك الـ Bus بيكسر اي سلوك او سلوك

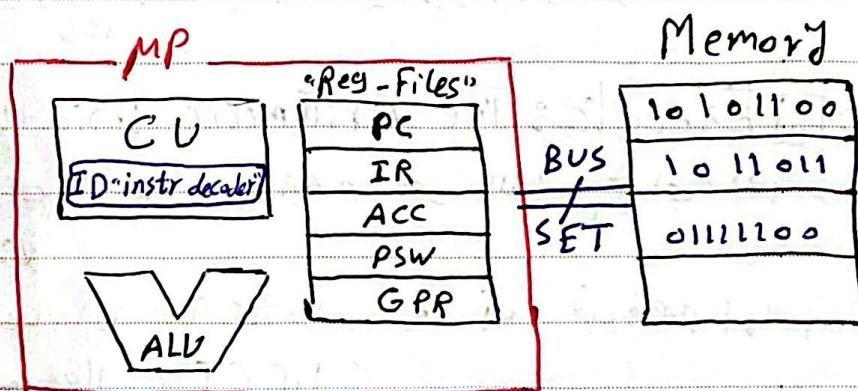
Bus Set:

بيانات حاسوبية

(-) Data Bus

(-) Address Bus

(-) Control Bus



⇒ Processor Operation:

(1) Fetch CU

(2) Decode ID

in instruction "نفس" [instruction] | instruction |

Reg Bank | IR | memory | ad |

عنصر الذي يأوي له

Instruction Register

الخطوة

تَعَالَى تَفْهِمُ الْكَلَامَ دَوْدَهُ وَادِرَهُ وَادِرَهُ

⇒ ① Fetch: The Control Unit (C.U) is responsible for fetching the instruction from memory via bus-set. It also controls the Register-files and Instruction-Reg. We can store the fetched instruction in the Instruction-Reg and then feed it to the ALU.

⇒ ② Decode: The Instruction Decoder (ID) is responsible for decoding the fetched instruction into operation codes (OP-Codes). The ID also controls the C.U to select the appropriate OP-Code based on the fetched instruction.

instruction	OP-Code
ADD	011
SUB	101
AND	110

1) Instruction Set:

The OP-Code is a binary number which defines the operation. It is used by the C.U to control the ALU.

Each instruction has its own unique instruction-set processor. All instructions follow the same basic structure.

2) Instruction Format:

OP-Code	OP1	OP2
3-bits	2 bits	3 bits
Ex: 10111011		
101	11	011

⇒ SUB

The instruction format consists of three fields: OP-Code, OP1, and OP2. The OP-Code is 3 bits, OP1 is 2 bits, and OP2 is 3 bits. The OP-Code is used to identify the instruction, while OP1 and OP2 are used to control the ALU.

⇒ ③ Execute: The Arithmetic Logic Unit (ALU) is responsible for performing the arithmetic and logical operations. It takes two operands and produces a result.

At the same time, the ALU also updates the Accumulator.

ALU is composed of logic-gates. The output of the ALU is stored in the Accumulator.

Question: Is The Compiler "Target specific"?

يعني هل او Compiler يعتمد على Processor ام Compiler مستقل عن Processor؟

الـ Assembly Code يتحول الى C-Code مثلما يتحول الى Assembly instructions ويكون معرفه بـ Processor instruction set فهو "Target Specific".

مثال: يمكن حاجه في الجمع يكون ADP بعدها : ADD
و AD MP الثاني بعدها : add

Question: A code Compiled for processor X, could it be executed on processor Y?

No it Cannot be Executed because The Compiler is "Target specific".

⇒ Instruction Set Architecture (ISA).

1. Reduced Instruction Set Computing (RISC)

2. Complex Instruction Set Computing (CISC)

نماذج المعمارية بحسب كل عام وتصنيفها ناشرها

لوقولنا في بقى هنالك العمليات

$$\text{int } X = 3 * 4$$

⇒ CISC:

تحتاج الكود لـ line واحد اساسي في ALU في logic gates في كل Instruction بحسب (Complex Instructions) بحسب عدد الاعداد التي تدخل في كل Instruction.

صيغة الـ Instructions كثيرة جداً جداً وهي بسبب عدد الاعداد التي تدخل في كل Instruction.

⇒ RISC:

تحتاج الكود لـ 4 بعدين لـ Assembly Instructions كلها هو صيغة هر فـ ALU من هنديه وقت لفهم الامر ده عالشان ده. الـ 4 اوامر التي بيديها صيغة هر فـ كلها موجودة في ID من صيغة هر ده، وبالتالي كلها في 1 clock cycle.

RISC

1) Performance:

وقت اقل في الـ Search | وقت اقل في الـ Search
 ملائمة بسيط او امر كثير جداً فهذا يزيد ، غالباً بمقدار 50% | ملائمة كله على بعضه بسرعه كام امر بسيط
 وقت اكبر في تنفيذ الكود | وقت اكبر في تنفيذ الكود
 ملائمة هو line واحد والعليه كلها متخرجه | ملائمة قسم الغرب لـ 4 lines
 فضاً ضد كل line في clock cycle على طول

CISC

1) Performance:

وقت اكبر في الـ Search | وقت اكبر في الـ Search

ملائمة بسيط او امر كثير جداً فهذا يزيد ، غالباً بمقدار 50% | ملائمة كله على بعضه بسرعه كام امر بسيط
 وقت اقل في تنفيذ الكود | وقت اقل في تنفيذ الكود

ملائمة هو line واحد والعليه كلها متخرجه | ملائمة قسم الغرب لـ 4 lines
 فضاً ضد كل line في clock cycle على طول

~~• Same Performance~~

2) Size : [large ID small ALU] 2) Size : [large ALU small ID]

~~Hard-wired Decoding Methods Memory Mapped~~

ملائمة الـ instrs سوية فهم فار ID | ملائمة الـ instrs كثير جداً فهم فار ID
 حاري فحصمه تكون اكبر | غالباً حاري فار ID جميع صفات

simple instructions بتنفذ ALU ملائمة الـ instrs

ردي الجميع وكمه فحصمه تكون صغير

بس لوكارنتي هي هيست الـ Decoding | ملائمة الـ ALU يصرف في تنفيذ الـ instrs

بس فحصمه حافظه ALU فالاعتبارات الـ Complex



already Hard-wired in The ID ↑

instructions in The ↑

Hardware part

Searches in memory ↓

Power Consumption | ↓

instructions in memory

بس خالدنا في كمها، نسبة الـ

ALU less RISC وار CISC

~~• Same Size~~

3).

P-Consumption | 3):

P-Consumption

ALU↑
ID↑

ALU↑
ID↓

~~• Same P~~

4) Cost : HW↓ SW↑

4) Cost : HW↑ SW↓

قوى د حفظ

قوى د حفظ

ALU↓

Same cost

قوى د حفظ

وهو المي فات نفهم انه ال CISC لا يعتمد على العكس ولكن المعمول يرجع لا بنيات الشركه mentality

ARM: RISC

Intel: RISC

⇒ not all The ISA are only CISC or RISC

في حياتك يومياً لبعض الأسباب تستخدم بعض الأجهزة

• (→) O ISC "One Instruction Set Computing"

(-) N ISC "No" " "

(-) Z ISC "Zero" " " " "

⇒ Register files:

الريجسستر هو نوع من انواع الـ memory وار موجوده في Processor

(-) Program Counter (PC): hold Address of next instruction to be fetched

بطريقه كلما اراد fetch instruction الى الصرف فلسا يكل اليه

(-) Instruction Register (IR): hold Instruction to be decoded and executed

في البداية execute و بعد ما decode instruction

"ARM needs to fetch and execute, decode and

(-) Accumulator (ACC): holds The Result of The ALU

(-) Processor status word (PSW) / Flag Registers: word byte holds flags about The result "status"

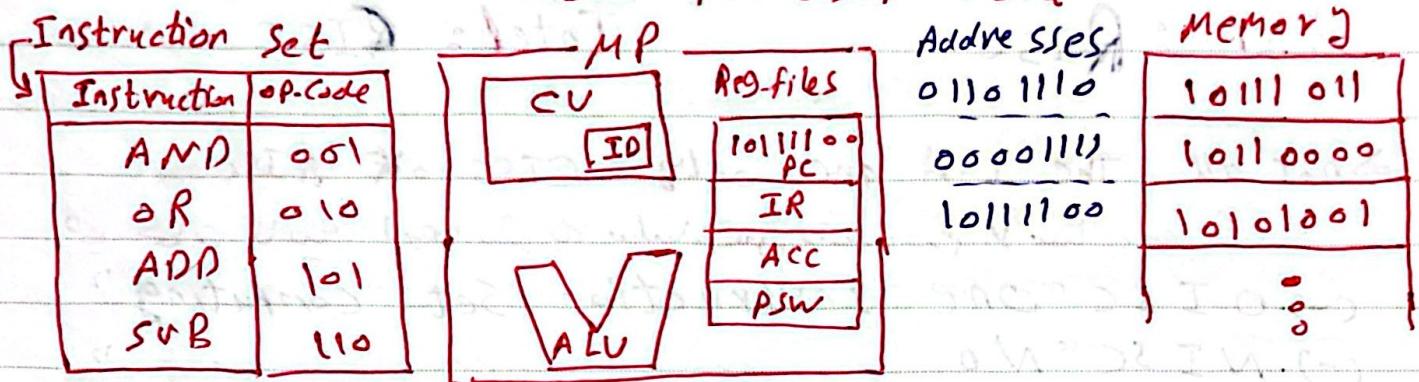
(-) General Purpose Register (GPR):

Registers موجوده في الـ CPU و يستخدمها في اي وقت يحتاج فيه

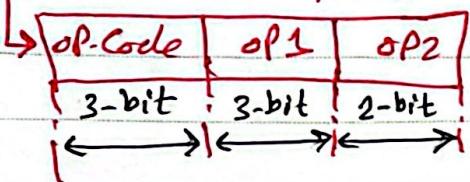
بيانات او دعوه يعني

و يستخدمون لتخزين الـ Pointers او operands

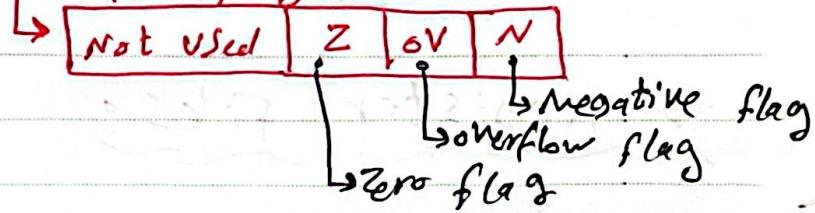
Example Consider The following processor, Perform one Complete processor cycle



Instruction format



PSW/Flag Register



next instruction لـ instruction PC كل دورة ①
 وبذلك أو لا يأخذ من بحاجة ~~CLOCK~~

(IR) Instruction Reg next instruction لـ instruction IR clock ② fetch

لـ instruction decode clock ③ decode

(1) Instruction set (2) Instruction Format



وبالإضافة إلى ذلك في ADD وباقي عمليات الجمع والطرح

ابقاء خلاصت عملية

ACC [0000 0011] ← [0000 0011] + [0000 0011] ④ Execute

وتحطيم إشارات خارجية PSW/flag-Reg. عالمة القيمة لا تغيره صفر و/or overflow قيمة سابقة ولا في

Reg-files

Go final Result :

PC	1011100
IR	1010101
ACC	0000 0011
PSW	0000 0000

#

⇒ Computer Architectures: Mem I/O MP II

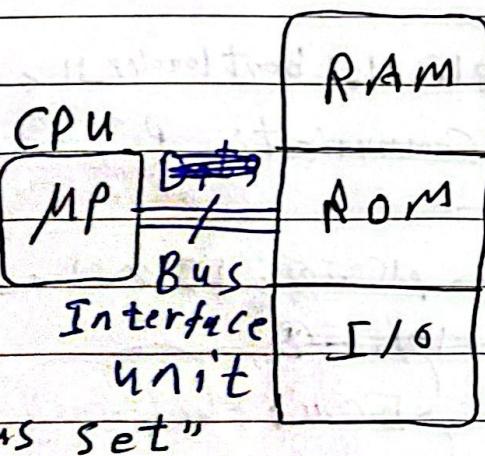
Von-Neumann

هي معمارية Von-Neumann وهي معماري Harvard معاصرة لها نفس المكونات ونفس الوظائف ولكنها تمتلك معماري Harvard كنوع خارجي للذاكرة.

Harvard

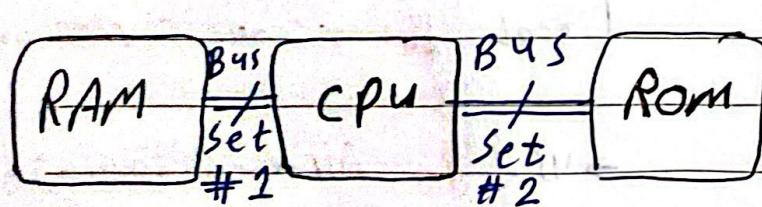
1 Von-Neumann (One-Memory System)

الذاكرة واحدة (Mem) والـ I/O هي مدمجة.



الذاكرة واحدة (Mem) والـ I/O هي مدمجة. RAM و ROM يشاركان في نفس الوقت إلى المدخلات والoutputs. RAM يرسل Data Bytes (Data bus) إلى CPU، و CPU يرسل Address bytes (Address bus) إلى RAM. RAM يرسل Data bytes إلى CPU، و CPU يرسل Address bytes إلى RAM. RAM هو المخزن الأول لـ Assembly language.

2 Harvard



الذاكرة واحدة (Mem) والـ I/O هي مدمجة. RAM و ROM يشاركان في نفس الوقت إلى المدخلات والoutputs. RAM يرسل Data bytes (Data bus) إلى CPU، و CPU يرسل Address bytes (Address bus) إلى RAM. RAM يرسل Data bytes إلى CPU، و CPU يرسل Address bytes إلى RAM. RAM هو المخزن الأول لـ Assembly language.

RAM: Assembly should be transferred to (Load / store).

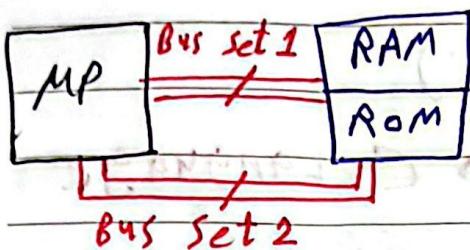
(we will use C-Language as the compiler makes it)

ROM: Assembly should be transferred to (Read / write).

(we must use Assembly to deal with ROM)

3 Modified Harvard Architectures:

(can be used as both Harvard and Von-Neumann)



Bus set 1: RAM + ROM

يُستخدم مع الـ哈佛 معملاً باستخدام الـC-Lang

Bus set 2: only ROM

يُستخدم بالذات أقر، ابتدأه لـROM فقط

on Assembly level

(computers)
Von-Neumann

I/O PrePhy LS

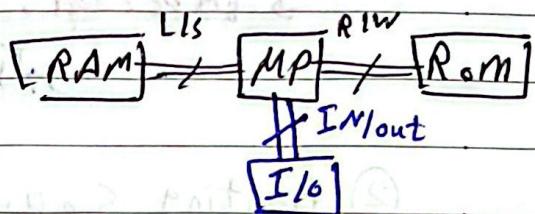
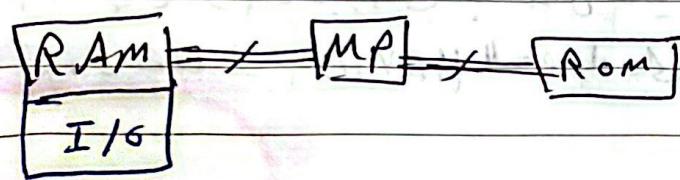
C-Embedded
Harvard

Memory Mapped I/O

Isolated I/O

"Port Mapped I/O"

يُستخدم توصيل I/O بالـRAM أو ROM بـMP، I/O يُعتبر جزءاً من الـRAM أو ROM



(RAM / I/O) → use C-Lang

(load/store) → RAM "C"

(ROM) → use Assembly

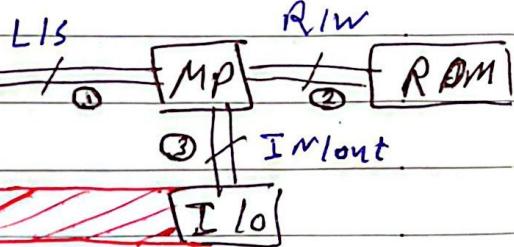
(Read/write) → ROM assembly

(IN/out) → I/O assembly

⇒ what is happening in AVR Architecture Mc?

- بـ AVR I/O ليس متساو

1 Bus set 1: through C-Lang but



You can't talk with both RAM, I/O together

2 Bus set 3: through Assembly

فـ AVR، كل من RAM و I/O له رسائله، i.e.,

RAM I/O

{ دatasheet الاتصالات في AVR }

AVR datasheet → Register summary → Address

\$ 3F (\$ 5F)

Assembly

→ C-Language

العنوان الى الاتصالات بالـ C او بالـ Assembly

In term of why in Embedded systems we use
Question Harvard, and in Computing systems
we use Von-Neumann??

① Signal Length:

64-bit

→ Computing: Von-Neumann structure فومن مون هيكل
wires will be used to connect between Bus Sets بusses
ملايين من الأجهزة التي تستخدم نفس خطوط

→ Embedded: Harvard structure هيكل هاروارد
قادرة على إنشاء بusses أكبر بكثير

② Booting Sequence:

→ Computing: It's needed to all Buses Set will
be connected to the main memory bus واحداً تبعياً

→ Embedded: It's needed to have a long time
for booting because it's in plastic case وارجع الى MP و LCD
فلازم اقعد اقوى من الفلاش وارجع الى MP و LCD
فلازم قوي bus set

⇒ Pipe Line: هو نوع من المعمارية يسمى RISC أو CISC، وهو يختلف عن معماريات معروفة مثل APP64 و CISK، AMD، INTEL، SUN.

☒ Von-Neumann: Does not support pipelining

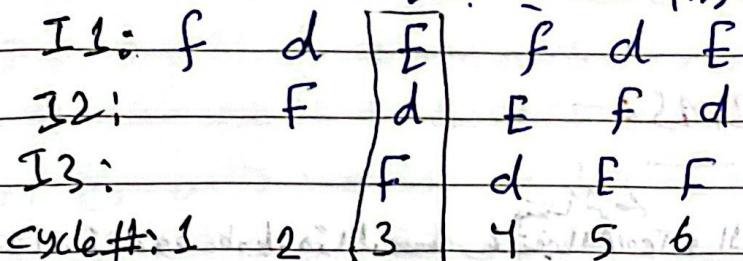
Idle بحث عن Execute وDecode وFetch

$f \rightarrow d \rightarrow E \rightarrow f \rightarrow d \rightarrow E \dots$

☒ Harvard: Does support pipelining

Idle Execute وDecode وFetch

تالي ديناميكي لـ instruction cache بعد كل Instruction



⇒ Pipelining reduces

Execution time as

it performs more cycles

⇒ RISC supports Pipelining: كل تعليمات بسيطة في

تحتاج "Reduced" number of stages

⇒ CISC Does not support Pipelining:

"Complex" 3 cycles في تعليمات معقدة ومتعددة في فراغات كثيرة وواسعة

Harvard (جهاز إلكتروني مدمج) و CISC
and RISC (ISA: Information-set Architecture)

1:23:30

دجع الغلاش

التاريخ:

اتسيل "السرمه"

الموضوع:

Interview what is The Meaning of ATMega32 MC question being 8-bits MC ??

Means That it has 8-bits data bus
يعني ابي اتعاوز تتعاوز ما فيها فالمرة الواحدة

لو الميكروكسيستر MC ١١ 8-bits حابه اكبر منه

كمان ادجام المريجسترز موجود في كونيك

⇒ Clock systems:-

- ① RC Vibration ميتوهه اول
 - اولاً تذبذب، وقت طوله جهاز المترacer ، وتغير الـ Temp والـ EMI
- ② Ceramic Resonator تذبذب متوسط ، رقم متواتر ، متوسط
- ③ Crystal Oscillator تذبذب عالي ، رقم كويسيه ، واستقرار جيد

EMI ١١ اجهزة الراديو و الجارو

Vibration ميتوهه اول

السيراميك والكريستال : مواد بتتذبذب Clock لود لها وبالتالي بلا من يتم استخدامها فالعربى لأنها بتو له اساساً في ترجع ميتوهه

اولاً : يستخدم في حابه في RC او Vibration

⇒ what is memory?

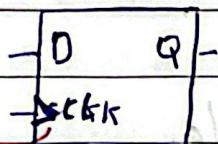
مجموعLocations المخزنة في مجموع instructions هي
MC Architect بحسب Bytes يكونون في Location وكل

"Every bit stored in a flipflop"

⇒ Rising Edge D flipflop

Rising edge موجة ارتفاع

و بعد ارتفاع يدخل المدخل لـ D



⇒ Register: Set of bits "Location" and Can be

Serial or Parallel PIPo → PISO

bits → 4 F.F

الريجستر هو صندوق يبسّم صوره له وظيفه وغرضه هنا هو

Reg + H.W Circuit لارزوم يكتفى حاجته Peripherals لأن كل

special Mem تتحدد الوظيفة

Periph بمتاعته الـ Reg #

H.W Circuit

لكل bit غالريجستر له وظيفه

والوظيفه بتاعتي هي اني افهم المفهوم H.W.Circuit

بناعي و اكتر سو ختوير بروح يكتفى فالريجستر اعمل كذا في الـ H.W.Circuit

حيث ينفذ التغير اليه انا عاوزه في الـ H.W.Circuit

⇒ Memory Aspects:-

- 1] Capacity: يعنی حجم الذاكرة وقدره bits
- 2] speed: Read time والـ R/W time (Access time) اسرع في write لان عملية write اسرع من read
- 3] Organization: لعمليات الـ 16x16 و 4x4 KB من الموجودة في السوق

4x4 KB 16x16

Memory Types

Volatile

غير مستدام

Non-Volatile

غير متغير

Hybrid

مدعوم بالأسنة

①

②

③

⇒ Volatile Memory (RAM):

DRAM
SRAM

يعد الـ RAM من الـ volatile memory، حيث ان البيانات التي فيها تختفي

عند اقراصها او اثنين فيها

الوقت الذي يستغرقه لاسترجاعها اول مكالمة في الـ RAM هو نفس

Random Access Memory

الوقت الذي يستغرقه ابي مكالمة ثانية (ROM)

(ويعني انه الـ RAM اسرع من الـ ROM)

لذلك فهو امثل في الـ working (بياناته تتسارع)

انشاء الـ Runtime

1) Dynamic RAM

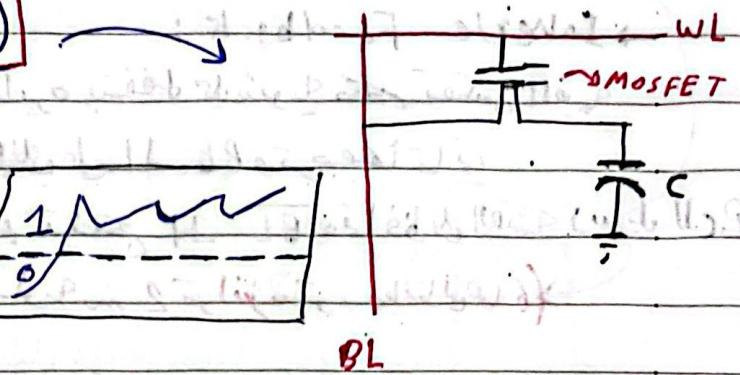
- Based on Capacitors (and Simple MOSFET Type) which discharging in time = T resulting in the need of Refreshment to keep the signal of 1 from being discharged. Solution is with (Refreshment Circuit)

BL: قيمة الـ bit لـ مسحونـ بـ 1 او مفـ غـ بـ 0

WL: المجموعـ بـ اـ لـ الـ bits او الـ word

MOSFET: الـ مـ مـ حـ اـ فـ تـ لـ الـ المـ لـ نـ مـ نـ اـ نـ يـ خـ سـ قـ يـ اـ دـ اـ لـ الـ مـ نـ مـ نـ اـ نـ يـ

Capacitor: الـ مـ بـ يـ شـ حـ دـ يـ فـ رـ خـ



Simple Hardware Design

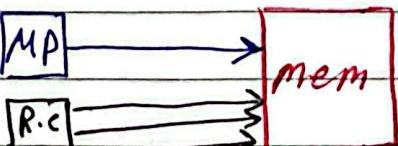
Low Cost per bit

High Density

Low Power Consumption

High Access time

MP is the first priority of Refreshment circuit



Larger Size

2) Static RAM

- ☒ Based on Transistors
- each bit consisted of F.F
- and each F.F at least has 6 Transistors

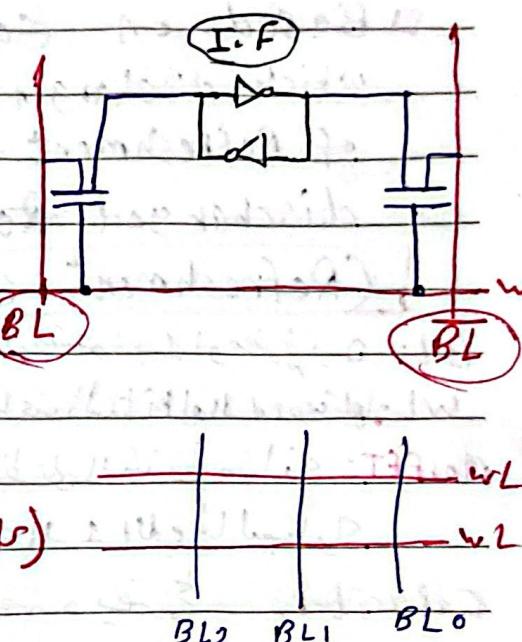
→ Inverse Feedback:

~~شرحها: دائرة بعفدها معاو ينعكس القيمه~~

~~التي تدخل BL وترجعها تاني~~

~~(R.C) علاوه: تنتهي BL وتحافظ على القيمه (يبدل الـ C)~~

~~لأنه تكون في 2 ترانزistor وبالتالي لها 2~~



☒ Complex Hardware design

☒ High Cost Per bit

☒ Low Density

☒ Lower Power Consumption

↳ Refreshment. C

☒ Low Access time

☒ ~~Small~~ Size

2 Non-Volatile Memory (ROM)

Based on floating gate Mosfet (FGMOSFET)

Higher Access time Than any Volatile Memory

Code is Applied to Rom using Burner

also called (Flasher - Flash driver)

to know what Burner do, First you need to Know FG-MOSFET

Floating Gate MOSFET:

SiO_2 علیه سطح

Control Gate (C.G): علیه سطح

drain side الشانة اليمانية ترحب بـ high Voltage

floating Gate الشانة العلوية ترحب بـ floating Gate

drain side high Voltage الشانة اليمانية ترحب بـ (Charge in FG)

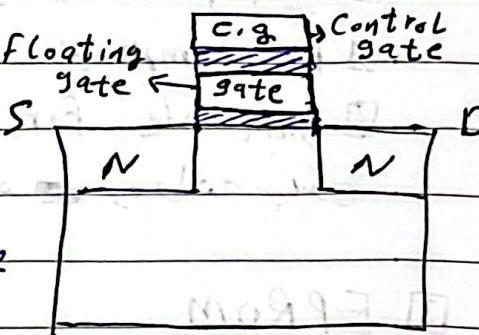
الشانة العلوية ترحب بـ +ve: Active High

floating Gate الشانة العلوية ترحب بـ -ve: Active Low

(+) Active high الشانة العلوية ترحب بـ

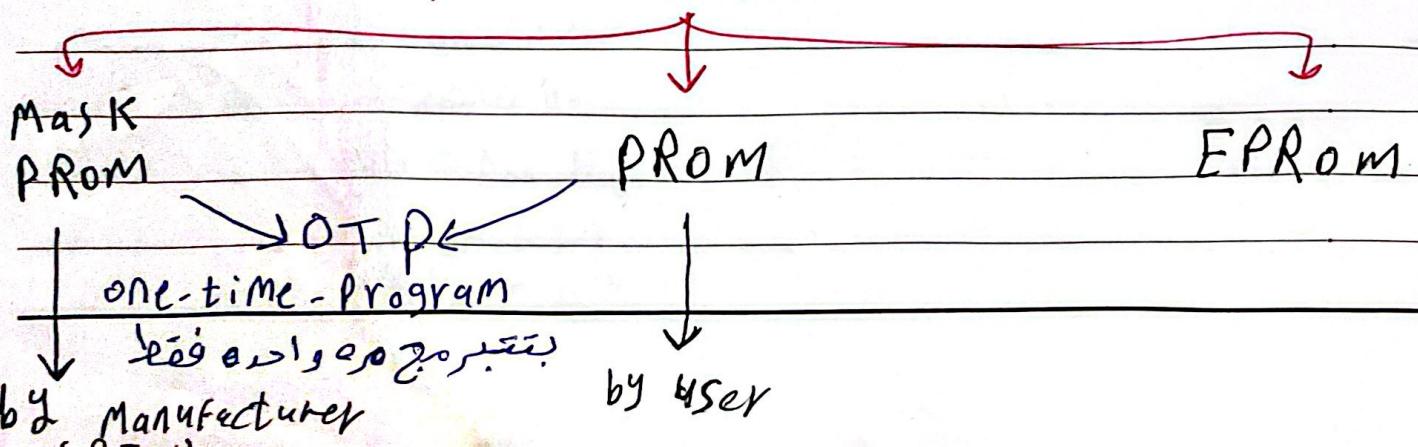
"1: Erasing state"

"0: Programming"



high Voltage الشانة العلوية ترحب بـ Burner الشانة العلوية ترحب بـ وباقياً

ROM TYPES



1 MASK PROM

O.T.P: one-time-program يعنى بستبرمج منه واحدة بس ومانقدر تغير الكود

Programmed by The Manufacturer "BIOS" chip

Suitable for Mass Production مناسب للانتاج الكبير

لأنها رخيصة ولا بتمني منها فور حفظ البروكشن فمفي حاجة لها لعب الأطفال يعني

2 PROM

O.T.P: one-time-program يعنى بستبرمج منه واحدة بس

Programmed by The user يمكن للمبرمج بمحاسنات

Suitable for Production Phase products مناسب لمرحلة الانتاج

بكون جاهز، انتابعقة الكود اللي انت عاوزه منه واحدة بس

3 EEPROM

Erasable by UltraViolet beam يمكن إزالته بالأشعة فوق البنفسجية

Can be Erased and reprogrammed Multiple times يمكن إزالته وإعادة تعيينه

→ (Endurance) لام

Save data to Long time (10 years)

As it based on U.V Erasing, can be erased by Radiation

③ Hybrid

☒ Read/write Memory
(from RAM)

☒ Non-Volatile Mem
(from ROM)

Hybrid Types

EEPROM
Electrically...
→ Byte Access ←

Flash

"BLOCK"

→ Sector Access ←

NVRAM

① EEPROM

- ☒ Electrically Erasable PROM
- ☒ Byte Access
- ☒ High Endurance "to 100,000"
- ☒ High Cost per bit
- ☒ Can be internal (through Register) or external (through Communication protocol) like (I₂C, SPI)

② Flash

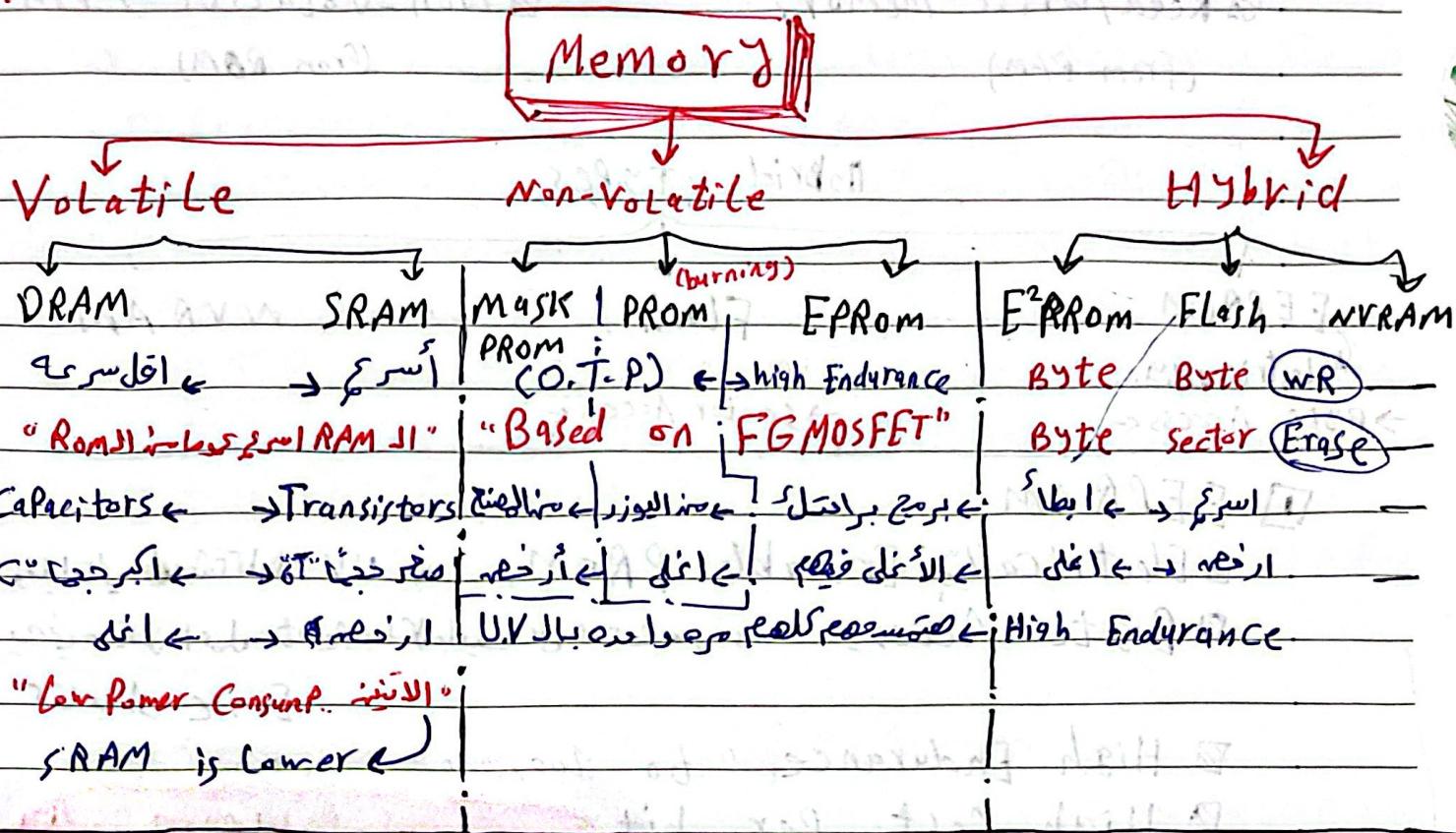
- ☒ Faster Than EEPROM
 - ☒ Sector Access
- 16 KB ~~or 256B~~ او حتى 256B و المساحة

→ Sector Access is an Adv or disadv??

أي ميزة لا تتوفر في سرعه و فلوس و يتم حل
Problema باستبدال Sector لـ byte
F.F.E (Flash E²PRom Emulator)
sector swapping

	Flash	E ² PRom
write	Byte	Byte
read	Byte	Byte
Erase	Sector	Byte

→ Access to instructions :-



لوعند صرورة وهي خواصها اقل حجم و اسرع سرعه من
CPU Reg, Cache memory, Main memory (RAM), Sec memory
(HDD)

1 CPU Registers

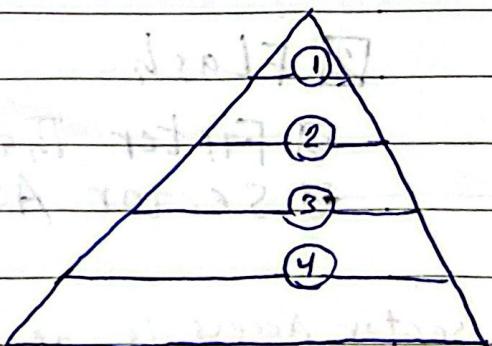
MP بدلها من بحث Bus بالذاكرة فيه جهاز MP

يتكلم الريجستير باسم طول

2 Cache memory

"Per bit" و بالذات اسرع و نوعيه

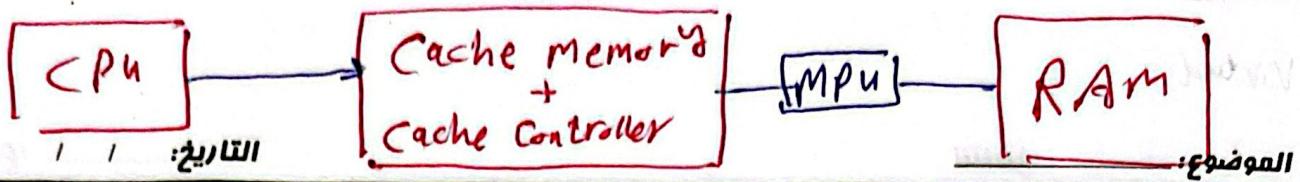
L1, L2 ... MP ي تكون في مساحات على حسب قرابة MP



3 Main Memory

ابطأ منها Cache على حجمها أكبر

4 Sec memory



نحو كـ Cache Memory + Cache Controller يزيد سرعة الوصول إلى RAM ولكن وصلنا لنقطة ثابتة حيث سرعة RAM أقل وبالتالي حصل y_{GP} gap بين MP و RAM.

ولذلك يتبع الـ Cache memory و بالتالي استعملا بالـ MP يعود حاجة بدوره في الأول ولو ماقصرنا RAM لـ Cache Controller \rightarrow Miss و يجب هنا انتظار MP لـ RAM.

Hit \rightarrow لا يسأل ثانية و دوره في ذلك أنه يتحقق فوراً.

Speed

MP
y_{GP}
RAM

t

⇒ FPU - Floating Point Unit : (ARM modes)

الوحدة المسئولة عن إجراء العمليات الحسابية على الأرقام الـ floating point وهي لها طريقة تخزين مختلفة عن الأرقام العادلة اسمها "Floating point Representation".

ARM → ARM Cortex-M → M3 "No FPU"

ARM → ARM Cortex-M → M4 "has FPU but disabled by default"

⇒ MPU - Memory Protection Unit:

Main memory (RAM) || Cache memory || يحظر معاً بينهم
MPU يحظر RAM منRegisters بيكوه لـ Physical RAM || physical.

- 1) Secure: محفوظة أو تكتب فيها
- 2) Not secure: مفتوحة أو تكتب فيها

لو عندك أكثر من MP وعاوز تغير RAM فالـ MP يحظر RAM وينفع أنت بتغيير خواصي يعني لو يكتبون فيها ما فيه

Virtual

التاريخ:

الموضوع:

→ MMU - Memory Management Unit: "Embedded Linux"

تقدر تردد الـ MP اى 8Goles او RAM سعياً لـ 8G با طريقة انتهاك القراءة (Access) من طرف مapping doing (V.A - Virtual - Addresses) (P.A - physical Addresses)

→ (الرام المبتكعي واحد بيجا وعاوز اسفله ابلوكشن متلاع 8 بيجا)

(Linux) يطلع على MMU من خلال MC #