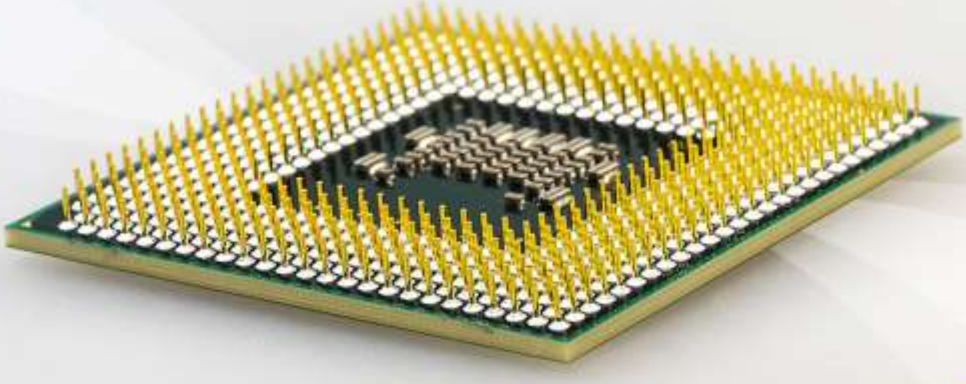
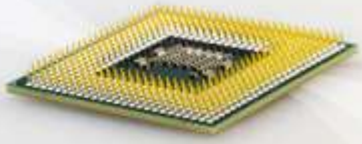


Von Neumann architecture

11.1A Computer systems

Lerning objectives



- describe the interaction of CPU with peripheral devices
- describe the purpose of CPU components, system bus and main memory

ASSESSMENT CRITERIA

- Describe the functions of CU, ALU
- Describe the pupose of system bus
- Describe the process Fetch – execute – decode cycle

Key Words

Control bus	is used to provide control signals to most parts of a computer system.
Address bus	is used to address the main system memory
Data bus	Is used to transfer data
Peripherals	devices that can be easily removed and plugged into a computer system.

Dedicated registers

Memory Address Register (MAR) – The address in main memory that we wish to fetch the instruction from

Memory Buffer Register (MBR) – Holds the instruction that has been fetched from memory

Current instruction register (CIR) – Holds the contents of MBR, which is the in struction to be processed

Program Counter (PC) – This holds the memory address of the location to fetch the next instruction from.

Status Register (SR) – Gives information on such things as overflow/underflow, interrupts, parity. The status register allows an instruction to depend the outcome of the previous instruction.

Accumulator – Holds the results of arithmetic and logic computations

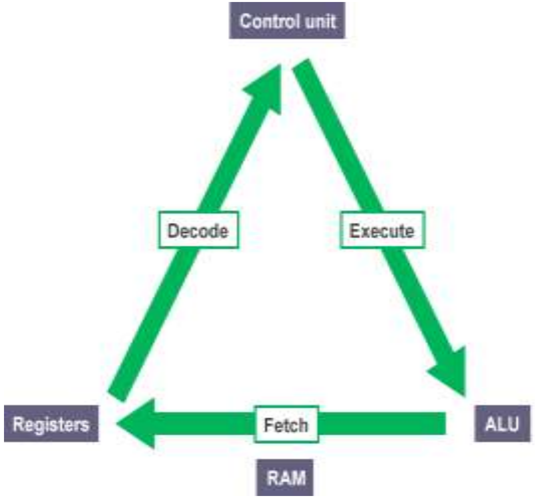
Fetch-decode-execute

https://www.youtube.com/watch?v=5s-faIDZI64&ab_channel=SENIOR.PROFESSOR

Register transfer notation

1. [MAR] ← [PC]
2. PC ← [PC] + 1
3. MDR ← [[MAR]]
4. CIR ← [MDR]
5. DECODE
6. EXECUTE
7. GO TO STEP 1

Fetch-decode-execute cycle



The instructions are loaded into memory

The processor fetches the instruction from the main memory

The instruction is decoded so the CPU knows what to do with the instruction

The processor then decodes and executes the instruction

The result of the instruction can be stored in memory

The next instruction is then fetched from main memory and the cycle repeats itself

Fetch

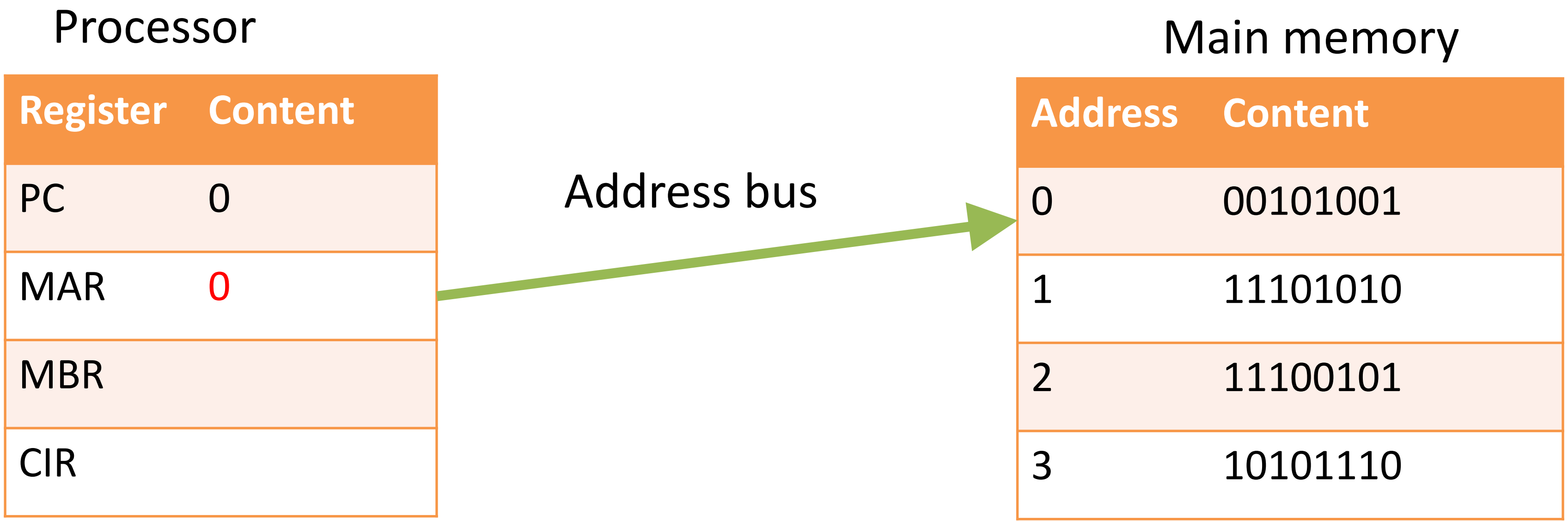
Processor

Register	Content
PC	0
MAR	
MBR	
CIR	

Main memory

Address	Content
0	00101001
1	11101010
2	11100101
3	10101110

Fetch: MAR ← [PC]



Processor

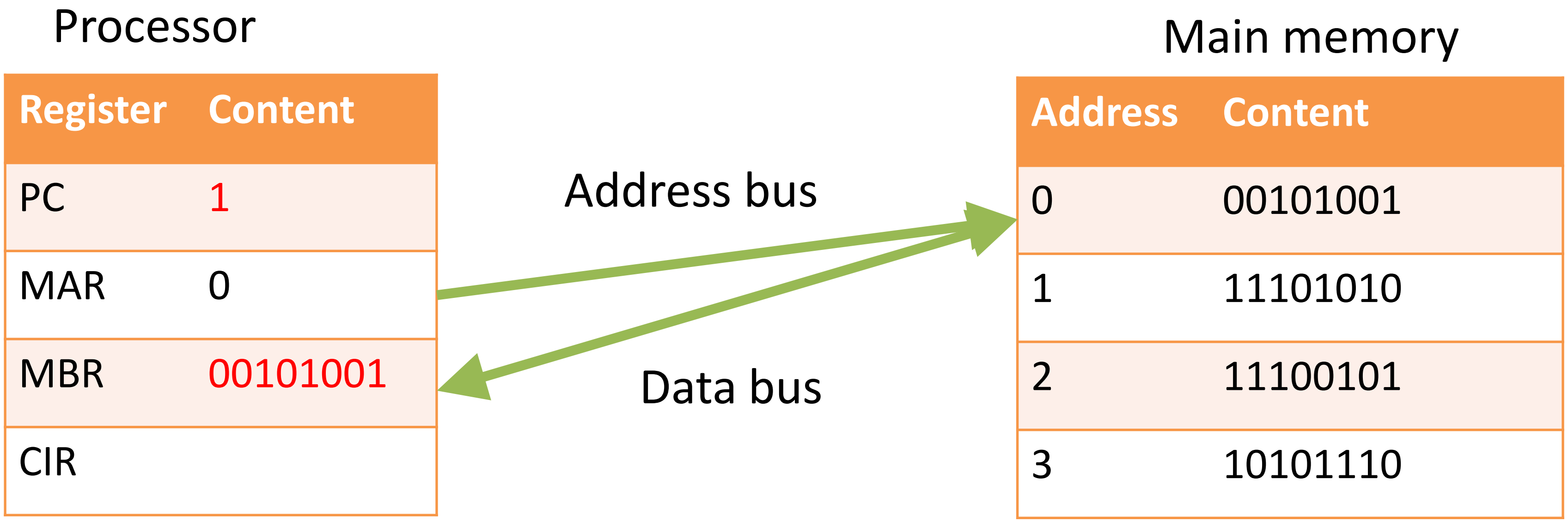
Register	Content
PC	0
MAR	0
MBR	
CIR	

Main memory

Address	Content
0	00101001
1	11101010
2	11100101
3	10101110

- Transfer the contents of the PC to the MAR
- The address in MAR is transferred to main memory along the address bus

Fetch: $\text{MBR} \leftarrow [\text{Memory}]_{\text{MAR\_address}}$; $\text{PC} \leftarrow [\text{PC}] +1$



Processor

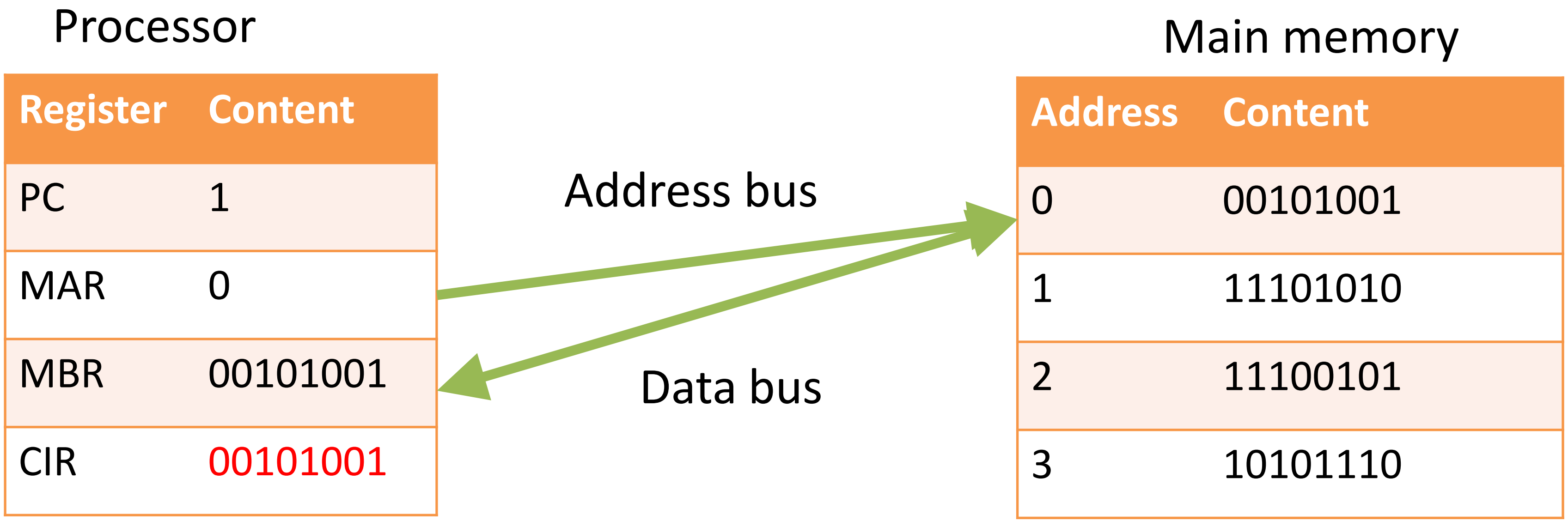
Register	Content
PC	1
MAR	0
MBR	00101001
CIR	

Main memory

Address	Content
0	00101001
1	11101010
2	11100101
3	10101110

- The contents at that address is transferred back to the MBR along the data bus
- At the same time the PC is incremented by one to get ready for the next cycle. This can b e done because the two steps do not rely on one another.

Fetch: CIR ← [MBR]



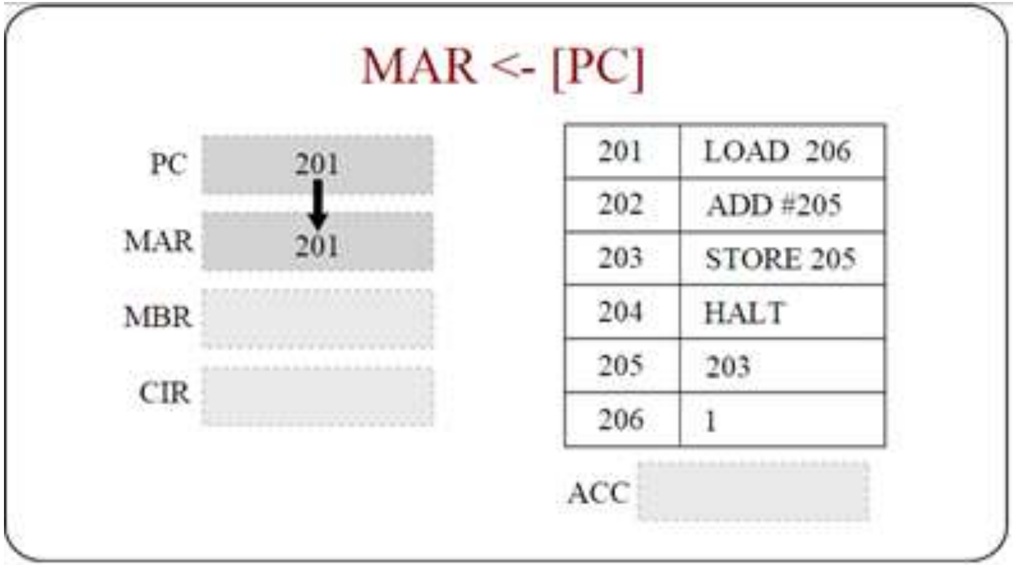
Processor

Register	Content
PC	1
MAR	0
MBR	00101001
CIR	00101001

Main memory

Address	Content
0	00101001
1	11101010
2	11100101
3	10101110

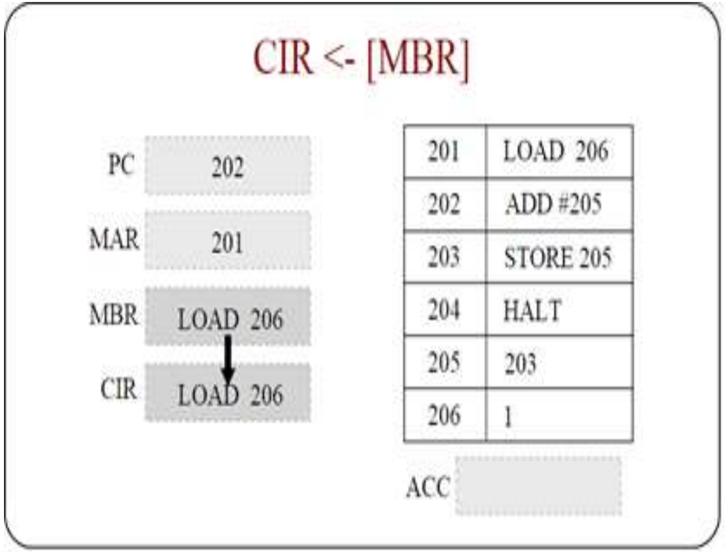
- Transfer the contents of the MAR to the CIR ready to be decoded.
- This allows the MBR to be used to store the next instruction during the decode and execution stages.



201	LOAD 206
202	ADD #205
203	STORE 205
204	HALT
205	203
206	1

Step by step description of fetch-execut-cycle

1. COPY THE ADDRESS THAT IS IN PROGRAM COUNTER(PC)INTO THE MEMORY ADDRESS REGISTER (MAR)



CIR <- [MBR]

PC 202

MAR 201

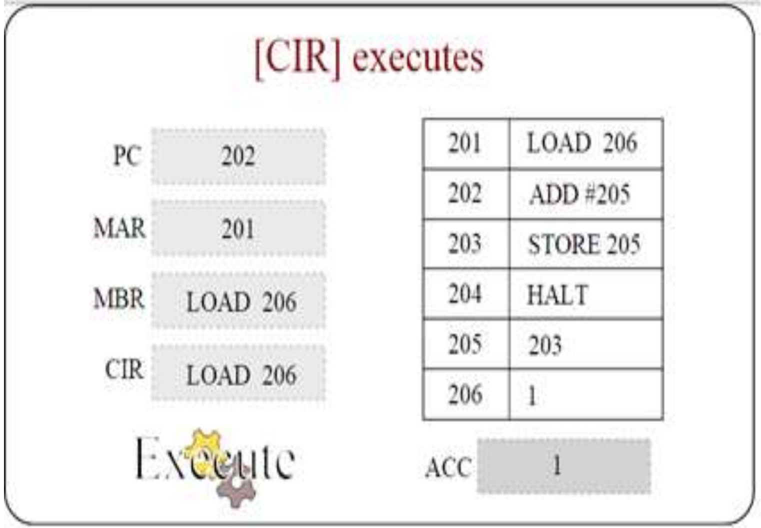
MBR LOAD 206

CIR LOAD 206

201	LOAD 206
202	ADD #205
203	STORE 205
204	HALT
205	203
206	1

ACC

2. INCREMENT THE PC (READY FOR NEXT FETCH)
3. LOAD THE INSTRUCTION THAT IS IN THE MEMORY ADDRESS GIVEN BY THE MAR INTO THE MEMORY DATA REGISTER (MDR)



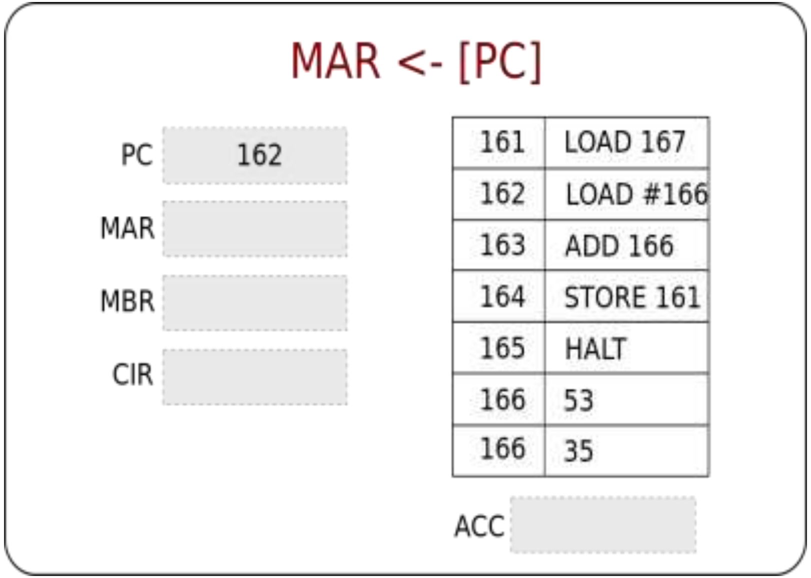
[CIR] executes

PC	202
MAR	201
MBR	LOAD 206
CIR	LOAD 206

201	LOAD 206
202	ADD #205
203	STORE 205
204	HALT
205	203
206	1

ACC	1

4. THE MBR LOADS THE CURRENT INSTRUCTION REGISTER WITH THE INSTRUCTION TO BE EXECUTED.



MAR <- [PC]

PC	162
MAR	
MBR	
CIR	

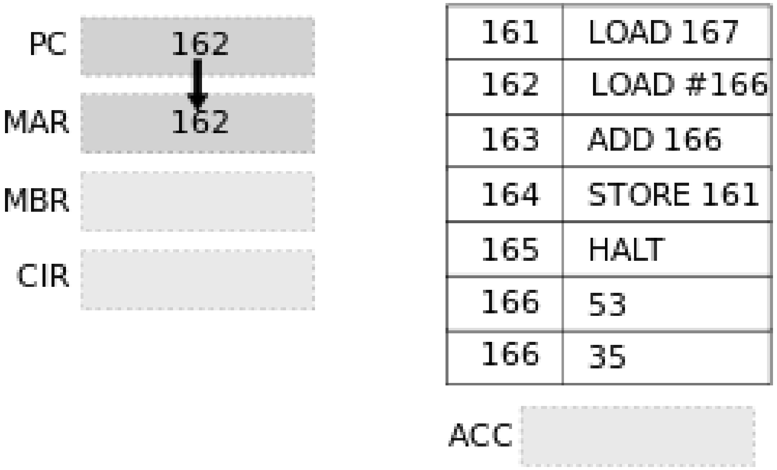
161	LOAD 167
162	LOAD #166
163	ADD 166
164	STORE 161
165	HALT
166	53
166	35

ACC	

Activity

Complete the following diagrams showing each step of the fetch decode execute cycle:

MAR <- [PC]

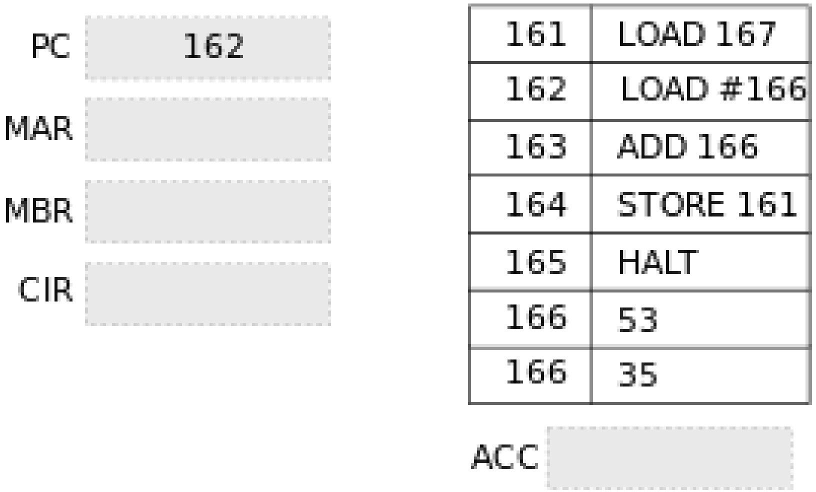


PC	162
MAR	162
MBR	
CIR	

161	LOAD 167
162	LOAD #166
163	ADD 166
164	STORE 161
165	HALT
166	53
166	35

ACC	

MBR <- [Memory]$_{\text{MAR address}}$; PC <- [PC]+1



PC 162

MAR

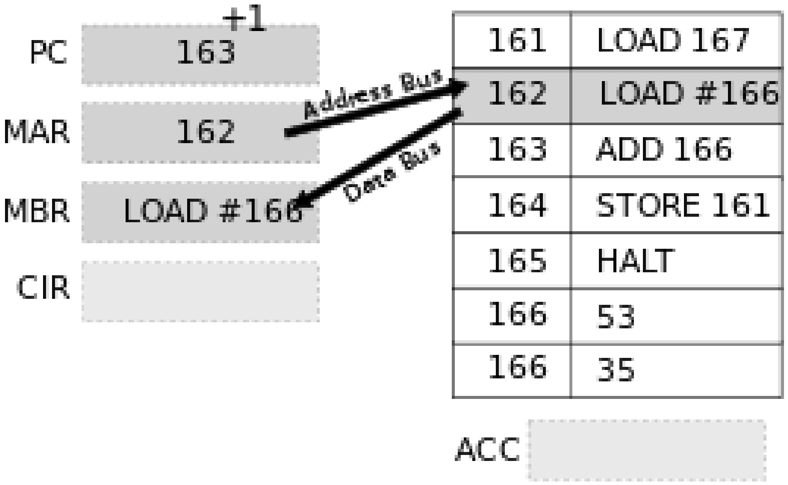
MBR

CIR

161	LOAD 167
162	LOAD #166
163	ADD 166
164	STORE 161
165	HALT
166	53
166	35

ACC

MBR <- [Memory]; PC <- [PC]+1$_{\text{MAR a}}$



PC 163

MAR 162

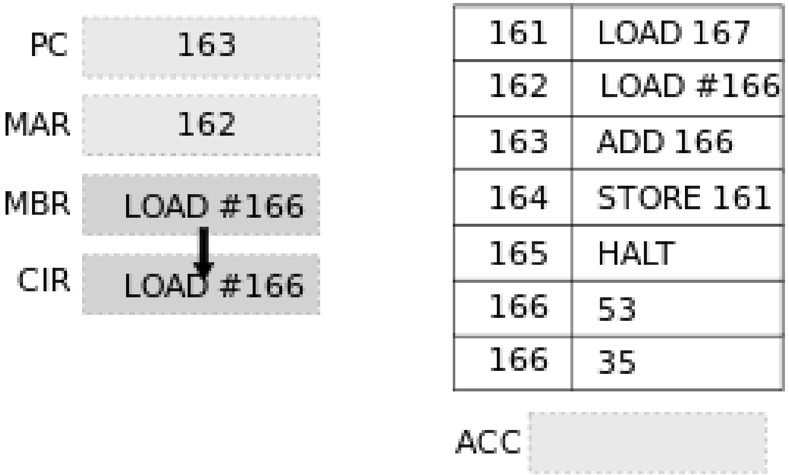
MBR LOAD #166

CIR

161	LOAD 167
162	LOAD #166
163	ADD 166
164	STORE 161
165	HALT
166	53
166	35

ACC

CIR <- [MBR]



PC 163

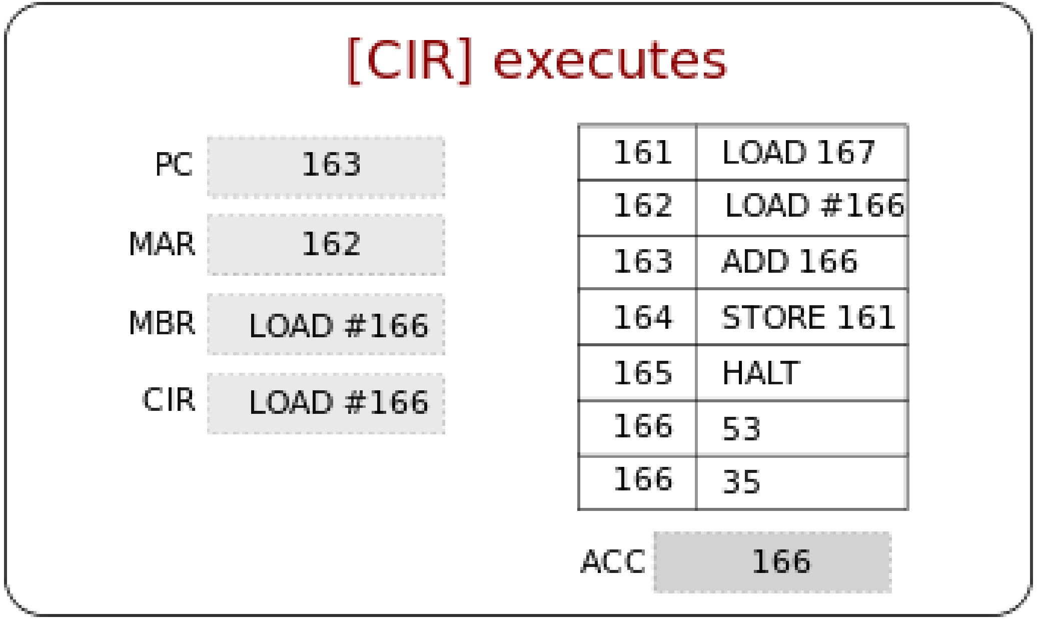
MAR 162

MBR LOAD #166

CIR LOAD #166

161	LOAD 167
162	LOAD #166
163	ADD 166
164	STORE 161
165	HALT
166	53
166	35

ACC



[CIR] executes

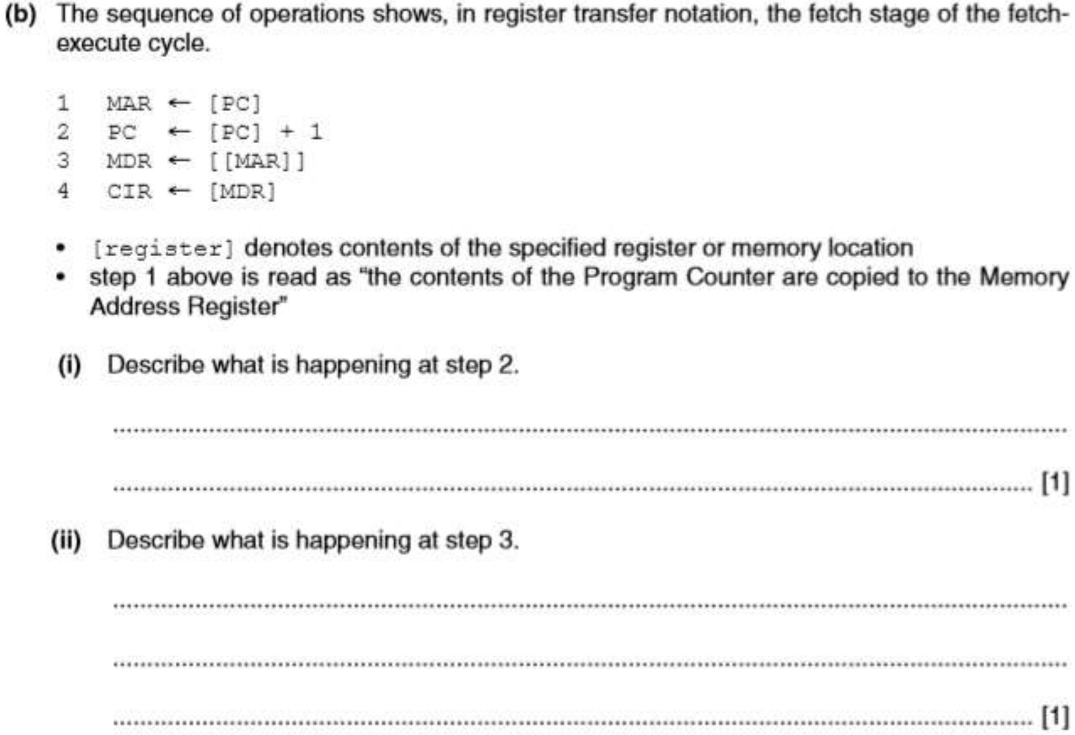
PC	163
MAR	162
MBR	LOAD #166
CIR	LOAD #166

161	LOAD 167
162	LOAD #166
163	ADD 166
164	STORE 161
165	HALT
166	53
166	35

ACC 166

Continue till 165

A Level questions



(b) The sequence of operations shows, in register transfer notation, the fetch stage of the fetch-execute cycle.

```
1   MAR ← [PC]
2   PC  ← [PC] + 1
3   MDR ← [[MAR]]
4   CIR ← [MDR]
```

- `[register]` denotes contents of the specified register or memory location
- step 1 above is read as “the contents of the Program Counter are copied to the Memory Address Register”

(i) Describe what is happening at step 2.

....

.... [1]

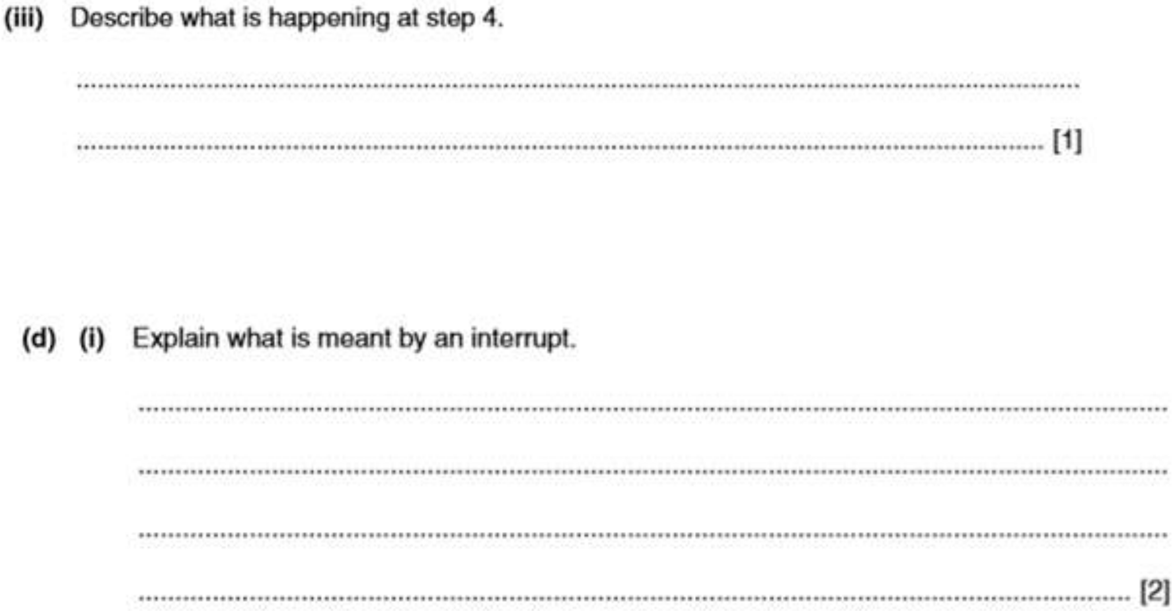
(ii) Describe what is happening at step 3.

....

....

.... [1]

A Level questions



(iii) Describe what is happening at step 4.

....

.... [1]

(d) **(i)** Explain what is meant by an interrupt.

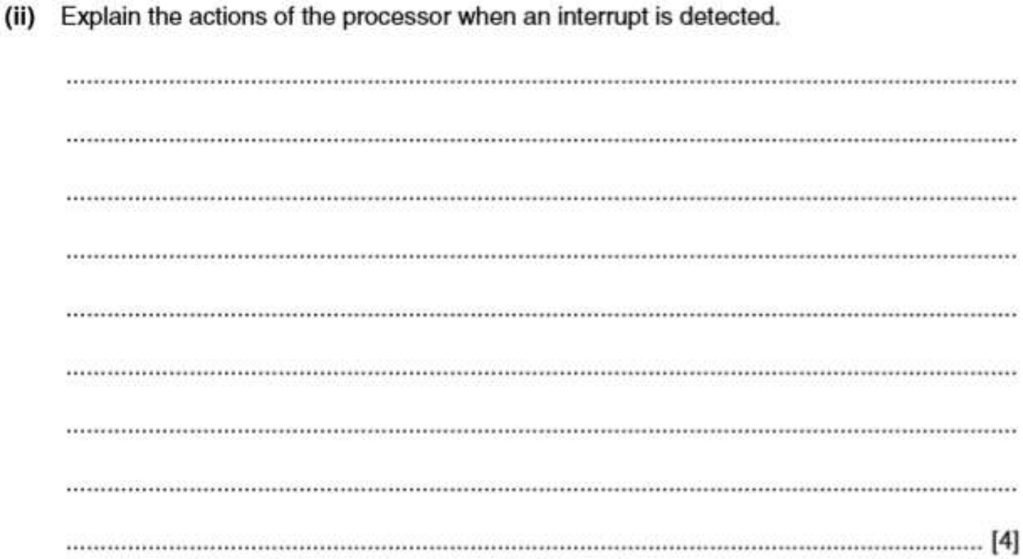
....

....

....

.... [2]

A Level questions



(ii) Explain the actions of the processor when an interrupt is detected.

....

....

....

....

....

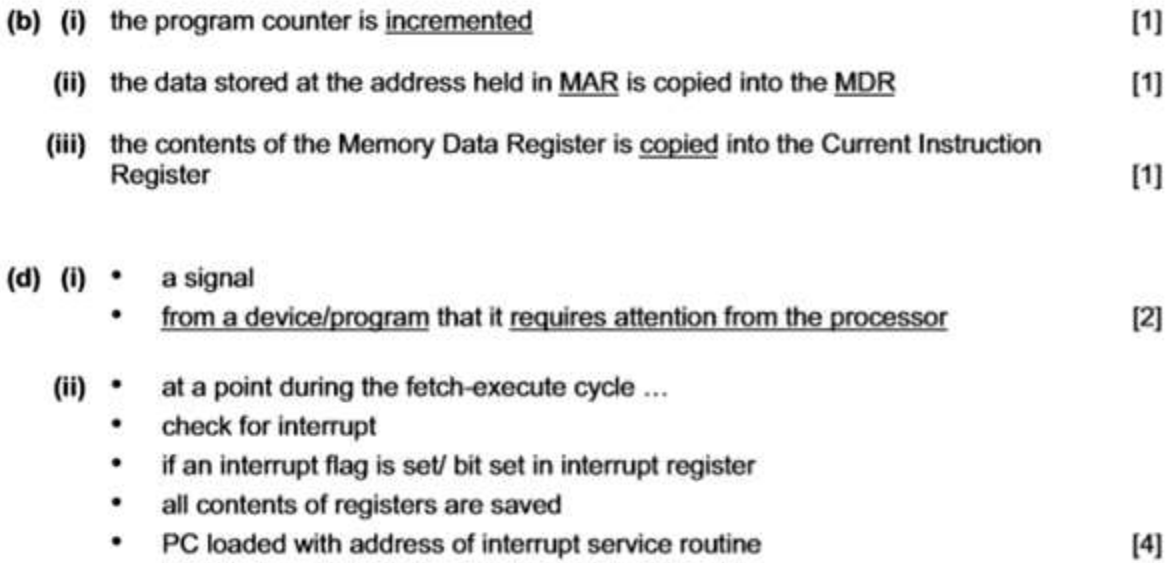
....

....

....

.... [4]

Answers



(b) (i) the program counter is incremented [1]

(ii) the data stored at the address held in MAR is copied into the MDR [1]

(iii) the contents of the Memory Data Register is copied into the Current Instruction Register [1]

(d) (i)

- a signal
- from a device/program that it requires attention from the processor [2]

(ii)

- at a point during the fetch-execute cycle …
- check for interrupt
- if an interrupt flag is set/ bit set in interrupt register
- all contents of registers are saved
- PC loaded with address of interrupt service routine [4]

3 A processor has one general purpose register, the Accumulator (ACC), and several special purpose registers.

(a) Complete the following description of the role of the registers in the fetch-execute cycle by writing the missing registers.

The .... holds the address of the next instruction

to be loaded. This address is sent to the ....

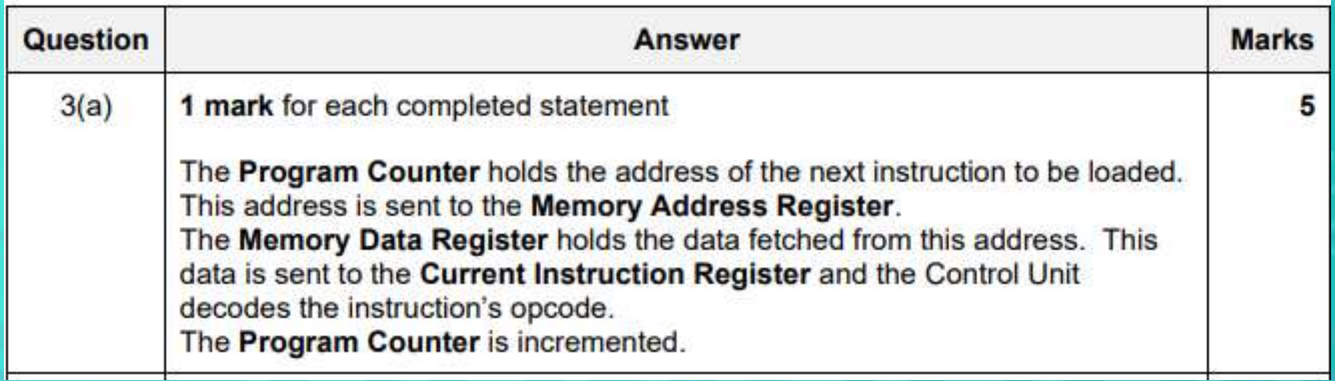
The .... holds the data fetched from this address.

This data is sent to the .... and the Control Unit

decodes the instruction's opcode.

The .... is incremented.

[5]



Question	Answer	Marks
3(a)	**1 mark** for each completed statement The **Program Counter** holds the address of the next instruction to be loaded. This address is sent to the **Memory Address Register**. The **Memory Data Register** holds the data fetched from this address. This data is sent to the **Current Instruction Register** and the Control Unit decodes the instruction's opcode. The **Program Counter** is incremented.	**5**

Work with pastpaper

Lesson Feedback

3

3 things you have learned today

2

2 things were interesting for you

1

1 thing I want to learn more