

# UD02. Functional parts of a computer.

## Part 2

---

Desarrollo de Aplicaciones Web

1er Curso

Curso 2020-2021

Autor: Vicent Bosch

[vicent.bosch@ceedcv.es](mailto:vicent.bosch@ceedcv.es)



**Reconocimiento - NoComercial - CompartirIgual (by-nc-sa):** No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

# Recordatorio

---



**Esta presentación no  
sustituye los apuntes  
disponibles en el aula  
virtual.**



**Las apuntes oficiales  
son los que tenéis en  
el aula virtual**

# Contents

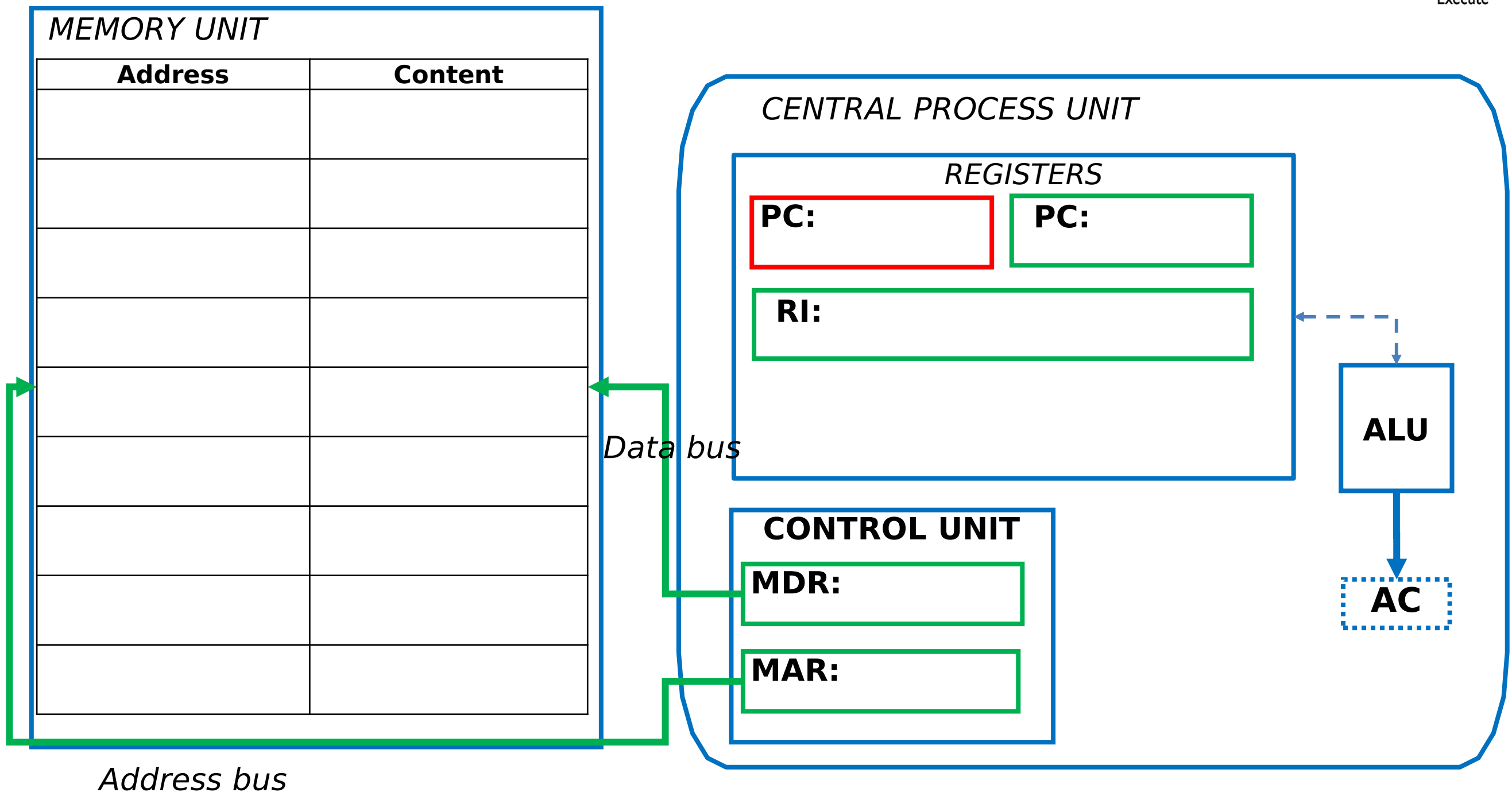
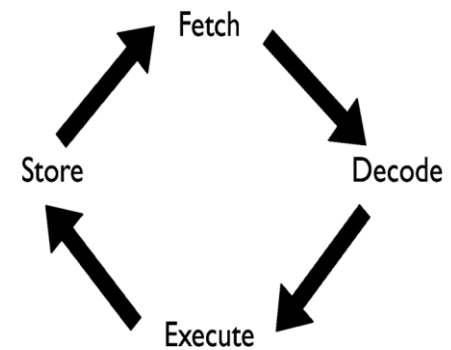
---

1. Instruction cycle
2. Configure Visual Studio Code
3. Introduction to Python
4. Questions

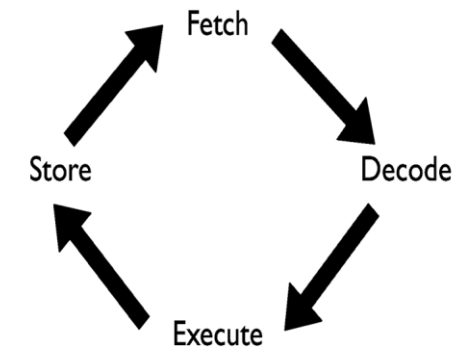
# 1. Instruction Cycle. Exercise 2

Code	Instruction	Description
ENT M(m)	000mmmmm	Read data from keyboard to memory.
SAL M(m)	001mmmmm	Show data on screen from memory.
CAR R0, M(m)	010mmmmm	Store content of a memory address in register R0.
ALM M(m), R0	011mmmmm	Store content of R0 in a memory address.
MOV Rx, Ry	1000xxyy	Copy content of RY to RX ( <u>X, Y are register numbers</u> ).
SUM Rx, Ry	1001xxyy	Add RX+RY and it is stored in RX.
RES Rx, Ry	1010xxyy	Subtract RX-RY and it is stored in RX.
MUL Rx, Ry	1011xxyy	Multiply RX * RY and it is stored in RX.
DIV Rx, Ry	1100xxyy	Divide RX / RY and it is stored in RX.

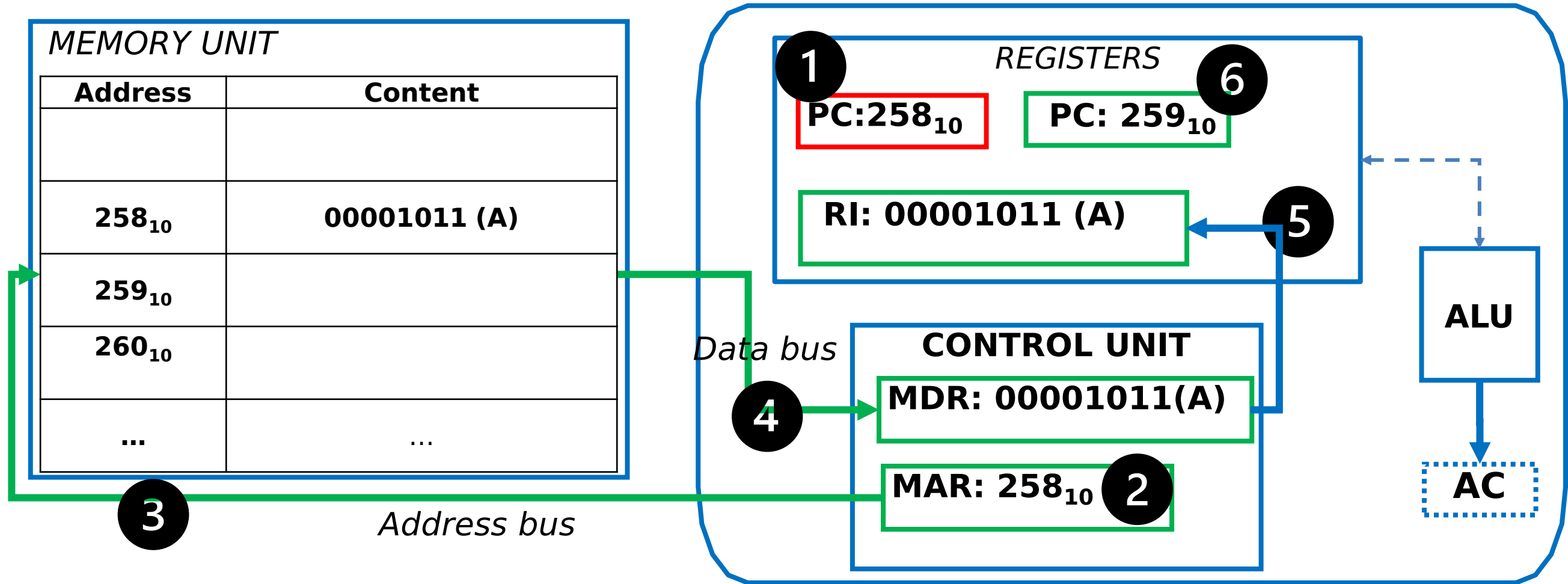
# 1. Instruction 1 →



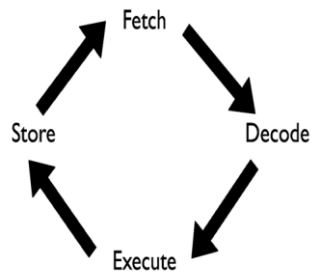
# 1. Common steps for each instruction → FETCH



## CENTRAL PROCESS UNIT



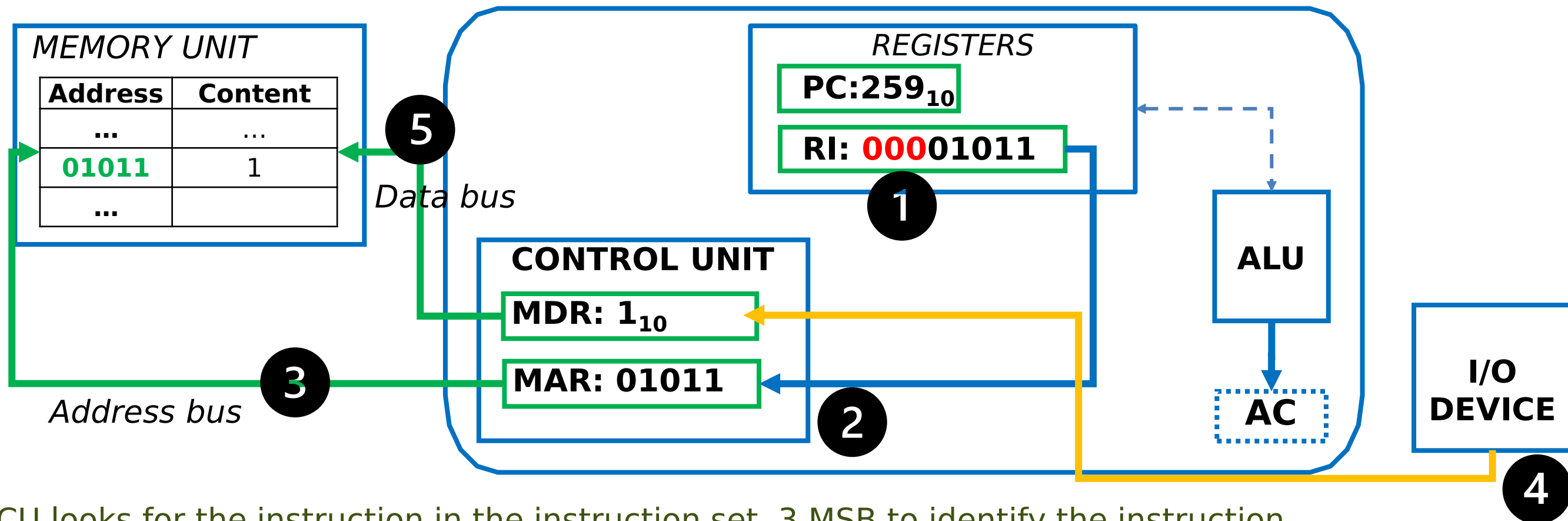
1. CU looks for the instruction in the memory address pointed by the PC
2. CU stores in the MAR the content of PC
3. CU sends signals to the address bus to point to the address stored in the MAR
4. The content stored in memory address accessed is stored in the MDR
5. The content of MDR is sent and stored in the RI
6. The CU updates the PC (increases in 1)
7. The CU can now decode the instruction → Next phase: DECODE



# 1. Instruction 1 → 00001011 (A)

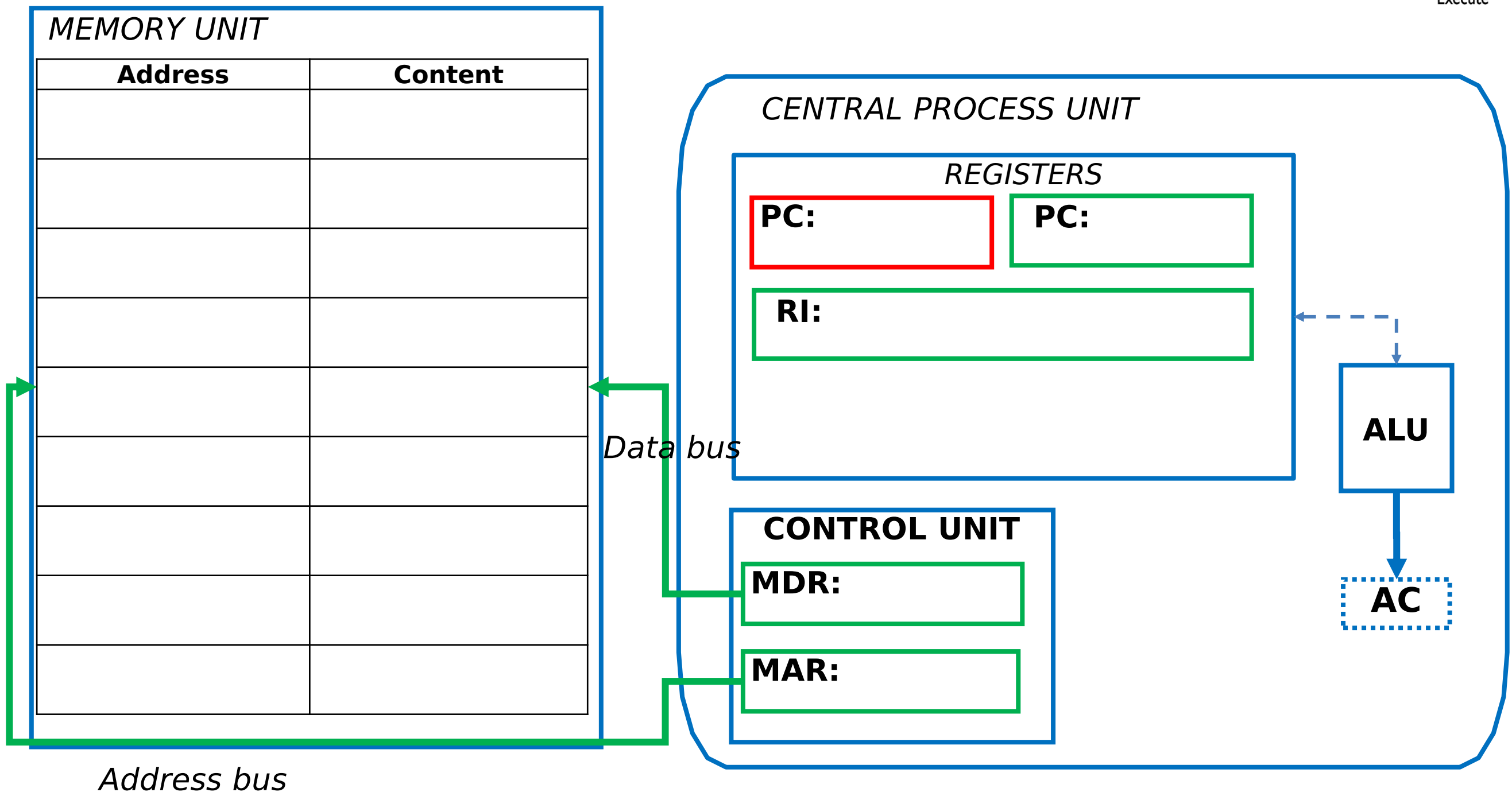
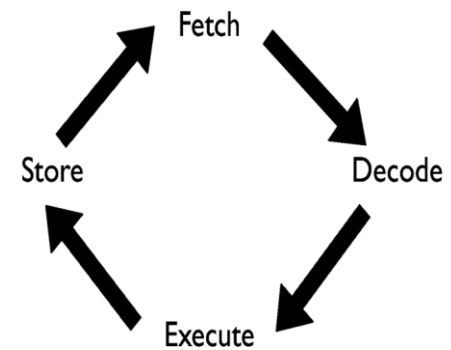
**00001011 (A)**    ENT M(m) 000**01011**    Read data (A=1) **from** keyboard **to memory**.

CENTRAL PROCESS UNIT



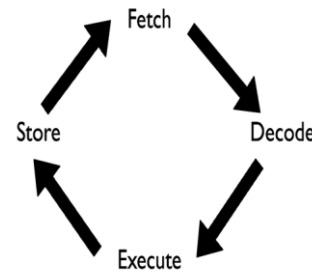
1. CU looks for the instruction in the instruction set. 3 MSB to identify the instruction.
2. The others 5 bits are used to point to a memory address, therefore this address is stored in the MAR
3. CU sends signals to the address bus to point to the address stored in the MAR.
4. The value stored in A (A=1), sent by the keyboard (I/O device) is stored in the MDR.
5. CU sends signals to the data bus to send the data stored in the MDR to the memory address previously accessed through the address bus.

# 1. Instruction 2 →

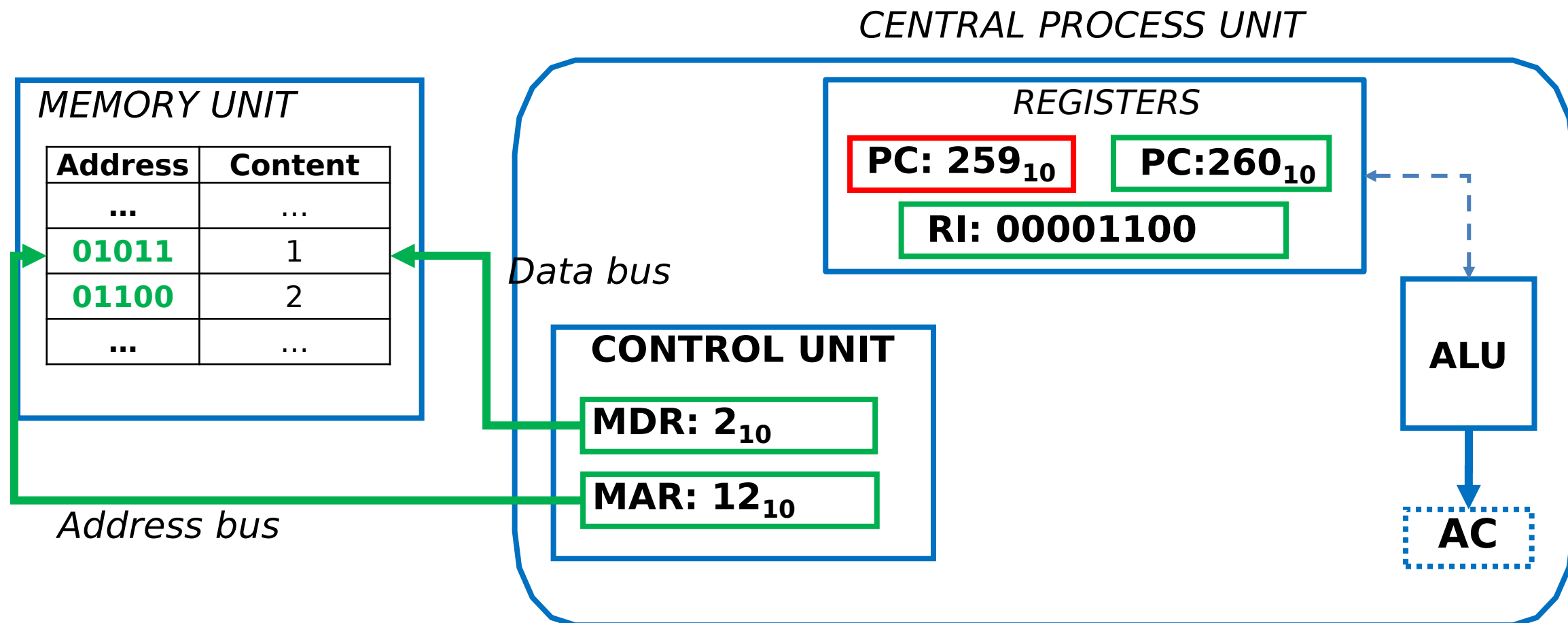




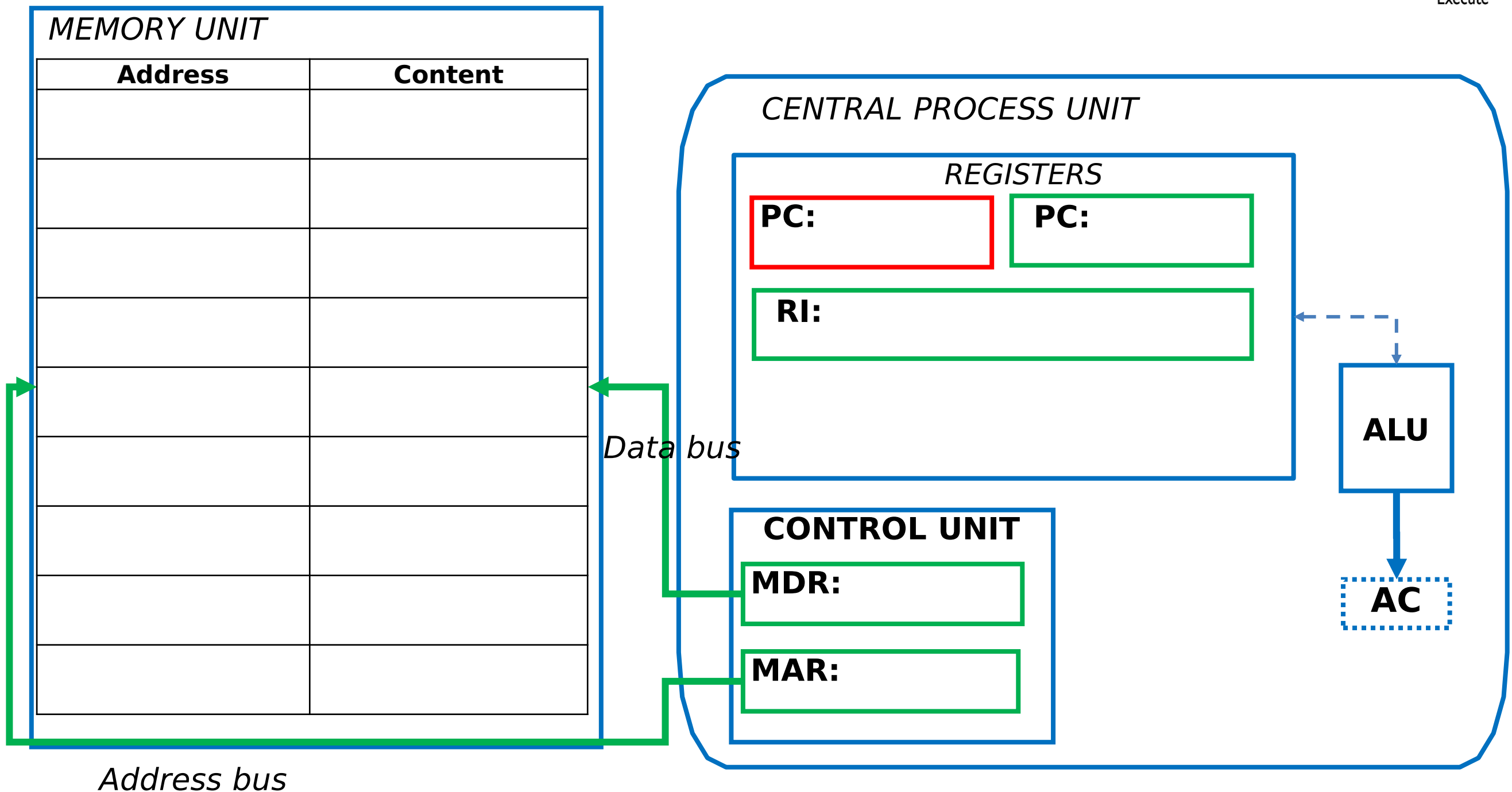
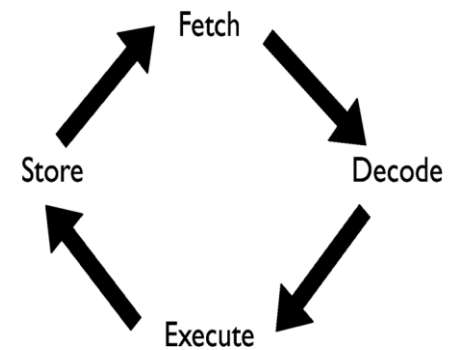
# 1. Instruction 2 → 00001100(B)



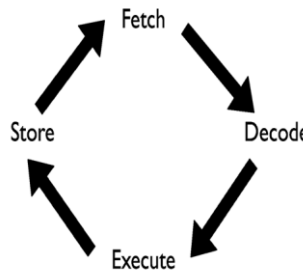
**000**01100 (B)    ENT M(m)    000**01100**    Read data (B=2) **from** keyboard **to memory**.



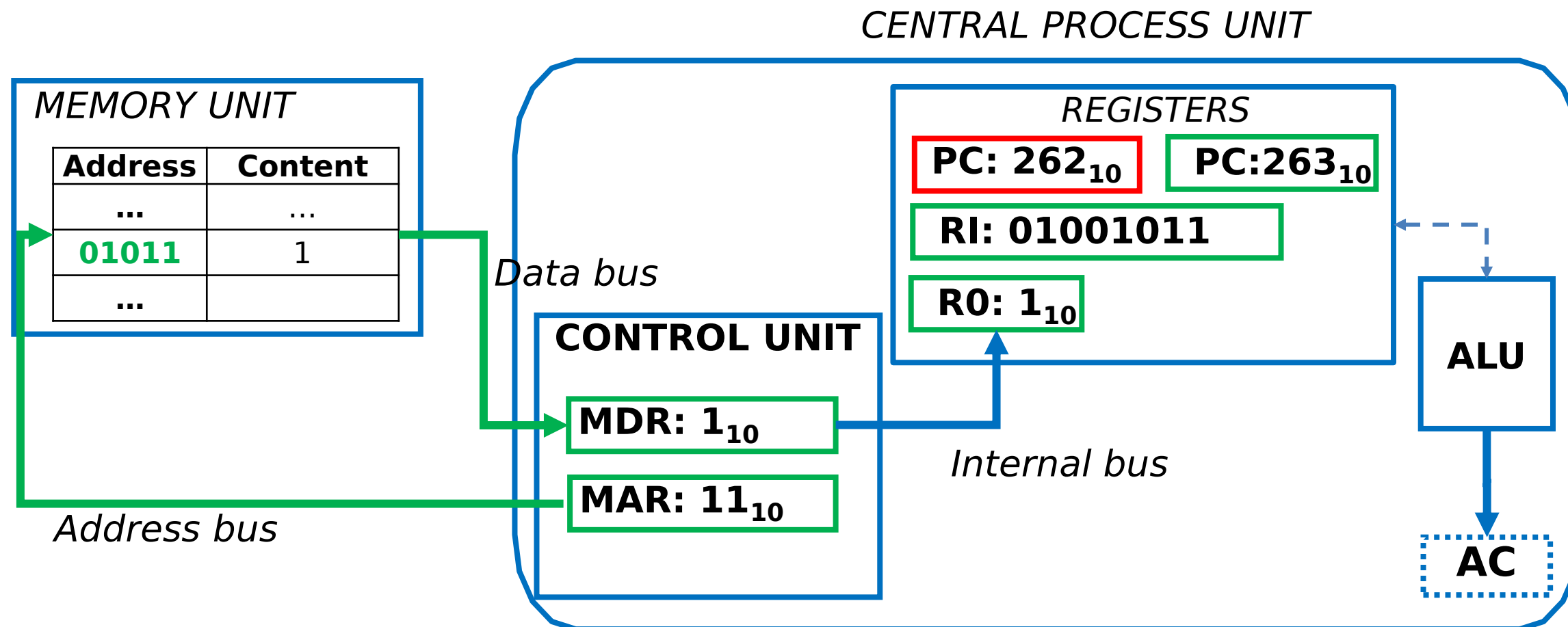
# 1. Instruction 5 →



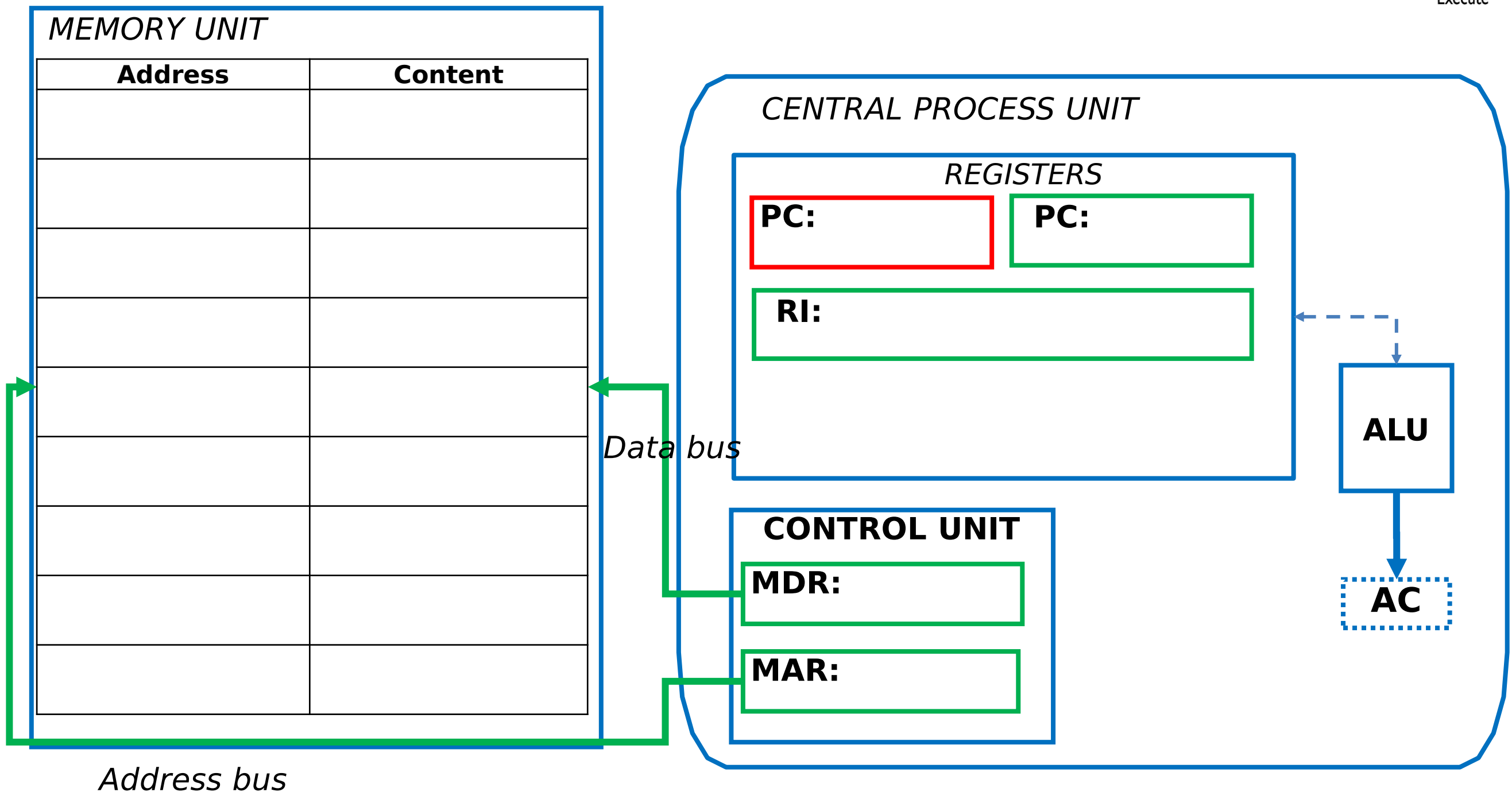
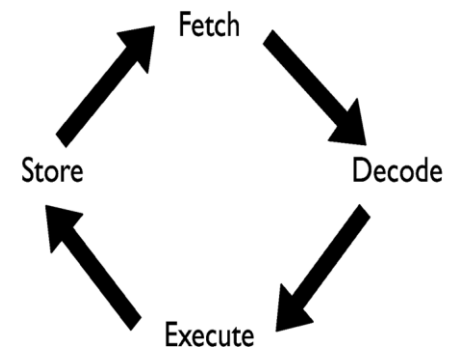
# 1. Instruction 5 → 01001011



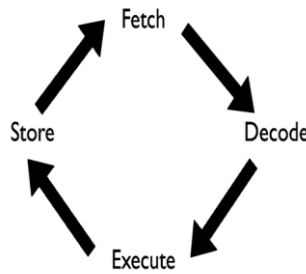
**01001011** CAR R0, M(m) 01001011 Store content of a **memory address 11** in register R0.



# 1. Instruction 6 →



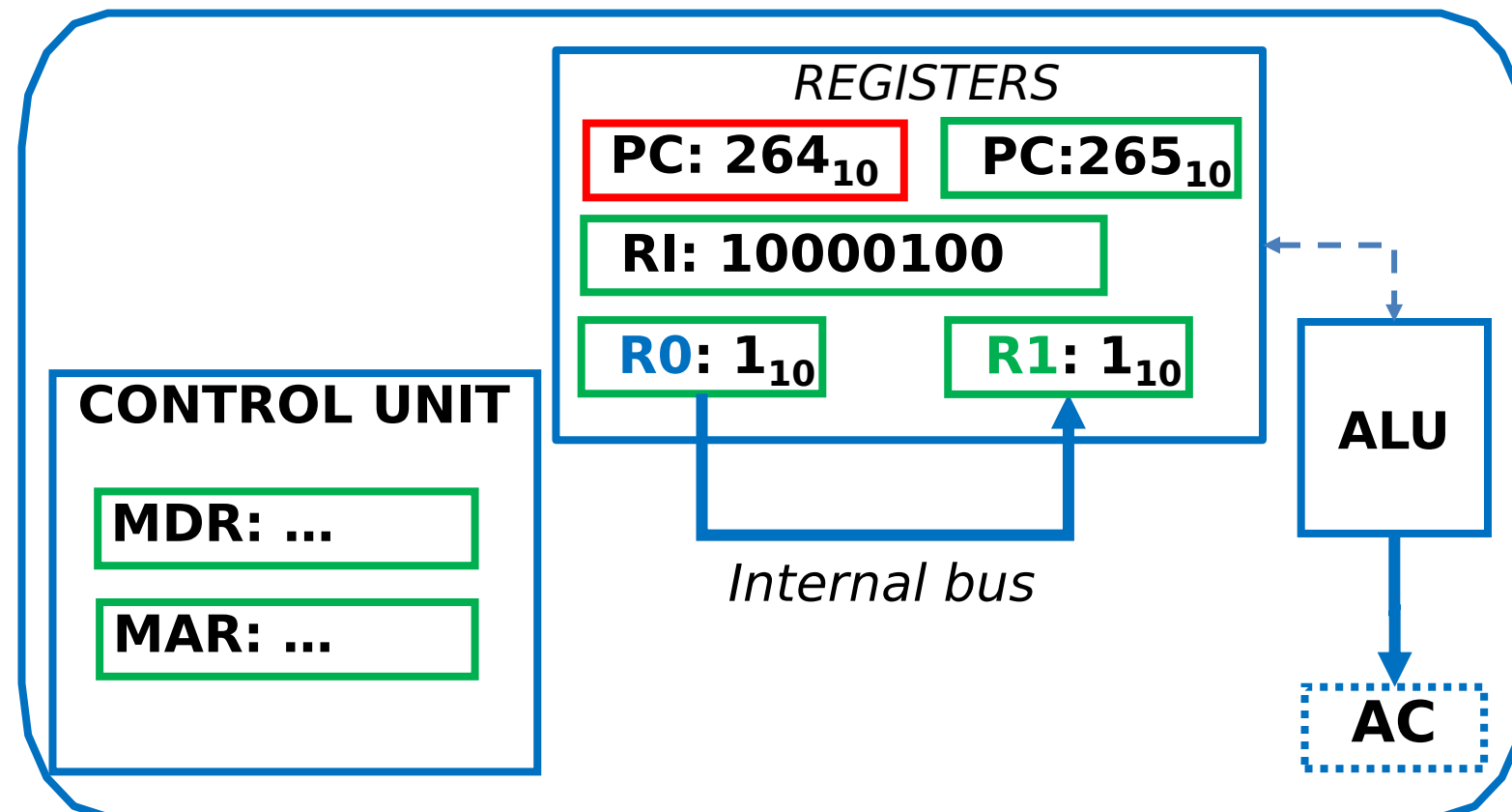
# 1. Instruction 6 → 10000100



**10000100**    MOV Rx, Ry 1000**0100**    Copy content of **R0** to **R1**

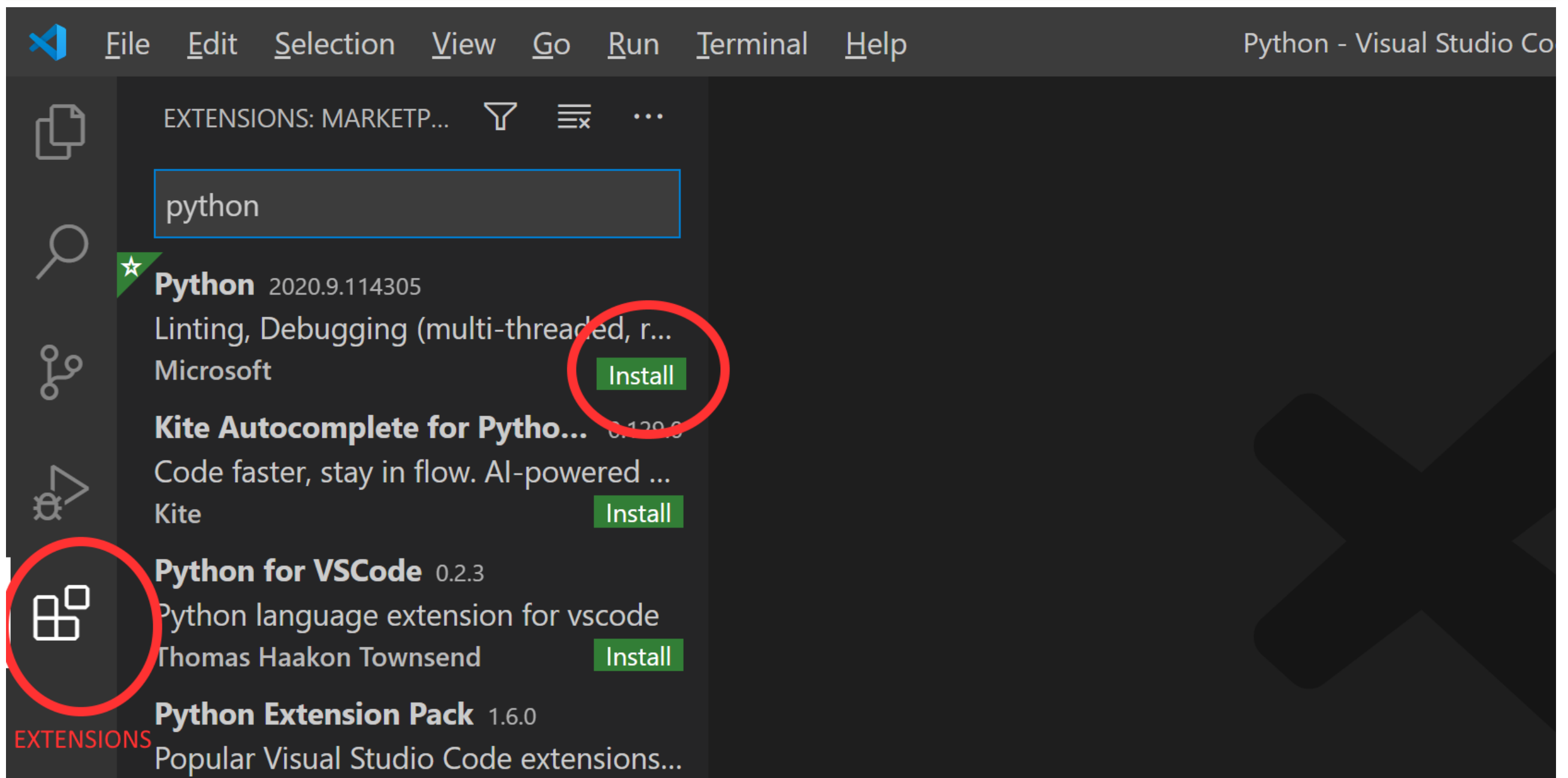
## CENTRAL PROCESS UNIT

MEMORY UNIT	
Address	Content
...	...
...	...
...	...



## 2. Configure VS Code

- First, install [Visual Studio Code](#)
- Second, open VS Code and install the Python Extension. Click on the extension's icon and search for “python”. Then click on “install” and finally restart VS Code.



## 2. Configure VS Code

- Verify Python is working properly.
- Open a Terminal in VS Code, using the top menu: Terminal → New Terminal.
- Write “py -0”

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

2

```
PS C:\> py -0
```

```
Installed Python(s) found by C:\WINDOWS\py.exe Launch
-3.9-64 *
```

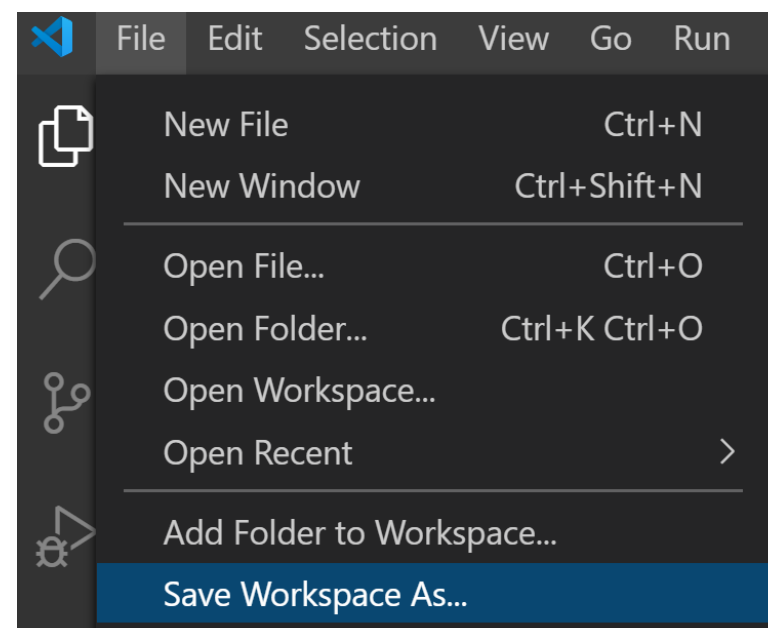
```
PS C:\> █
```

## 2. Configure VS Code

- Create a workspace for your exercises.
- Create a folder.
- Open a terminal, locate the folder created and type “code.”



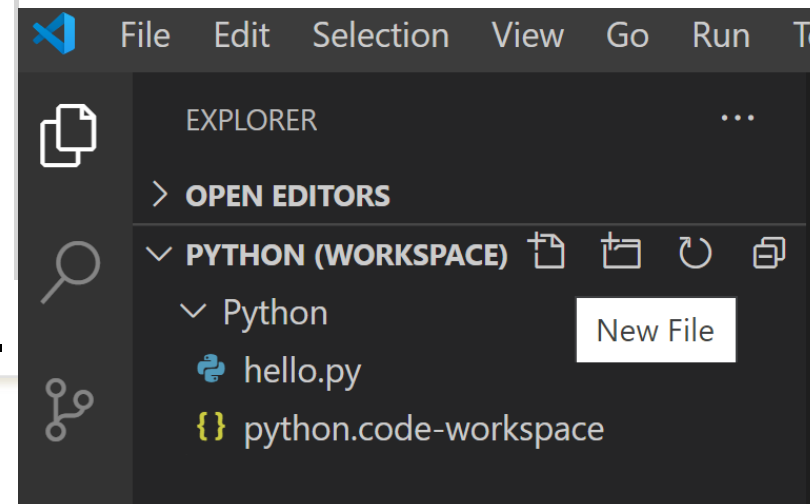
- and save your workspace.





## 3. Introduction to Python

- Open VStudio Code and create the first file in Python.



- Type  
print("hello") and  
RUN!!!!

A screenshot of the Visual Studio Code editor. The 'hello.py' file is open in the editor, showing the code `print("hello bit")`. The terminal at the bottom shows the command `PS C:\Users\V\Dropbox\FP\SI_CEED\Python> & C:/Users/V/AppData/Local/Programs/Python/Python39/python.exe c:/Users/V/Dropbox/FP/SI_CEED/Python/hello.py` and the output `hello bit`. A red circle highlights the 'Run' button (a green play icon) in the top right corner of the editor window.

# Questions? Share in the forum

---

If we have an instruction set with 12 instructions, how many bits at least do we need in the *op\_code* field of the instruction format to decode all the instructions?

Think of an instruction where you will use immediate and direct addressing modes.

In an instruction, do we have to decode all the bits?  
Justify your answer.

# Questions?

---

# Questions?

---

# Next week?

---

- ✓ Contents of Unit 3 will be opened during the afternoon on Friday 30<sup>th</sup>.
- ✓ Please complete the TC survey that it will be published in Unit 3. It helps to plan each TC.