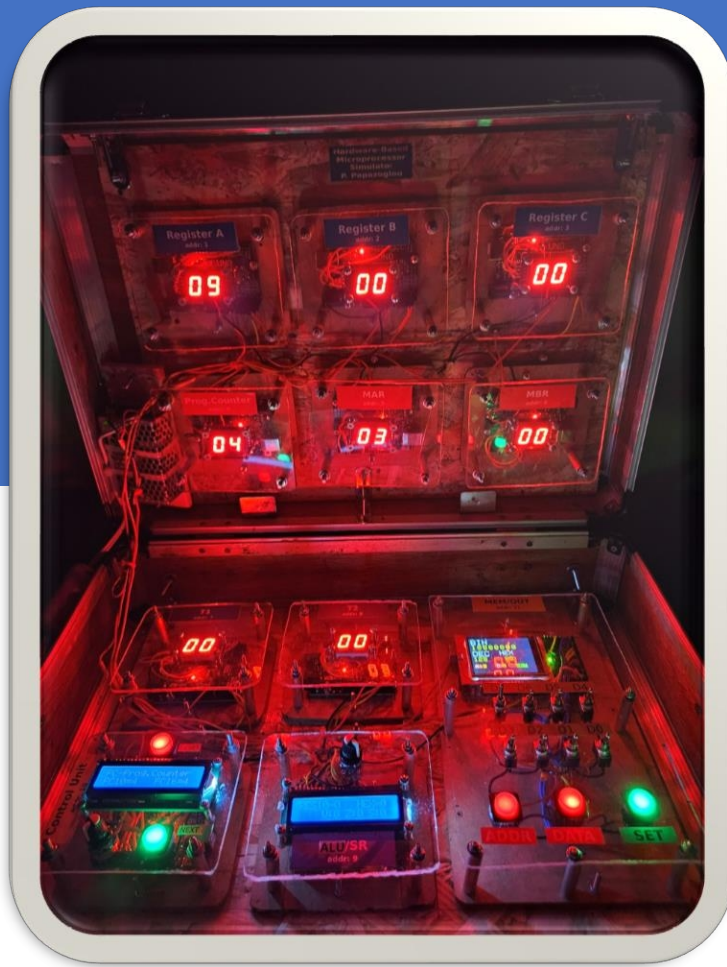


Hardware-Oriented Microprocessor Simulator (HOMS)



Quick User Guide v1.0

Designer-Developer
(C) Panayotis (Panos)
Papazoglou



HOMS Map

Hardware-Based
Microprocessor
Simulator
P. Papazoglou

Register A
addr: 1

Register B
addr: 2

Register C
addr: 3

General and
Special
Purpose
Registers

Prog. Counter
addr: 4

MAR
addr: 5

MBR
addr: 6

Arithmetic
& Logic
Unit (ALU)
and Status
Register

T1
addr: 7

T2
addr: 8

Memory
and Output
system /
Memory
data entry

MEM/OUT
addr: 11

Control
Unit

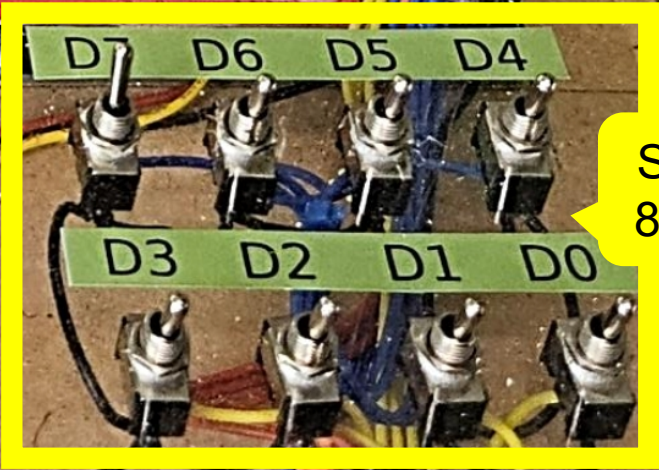
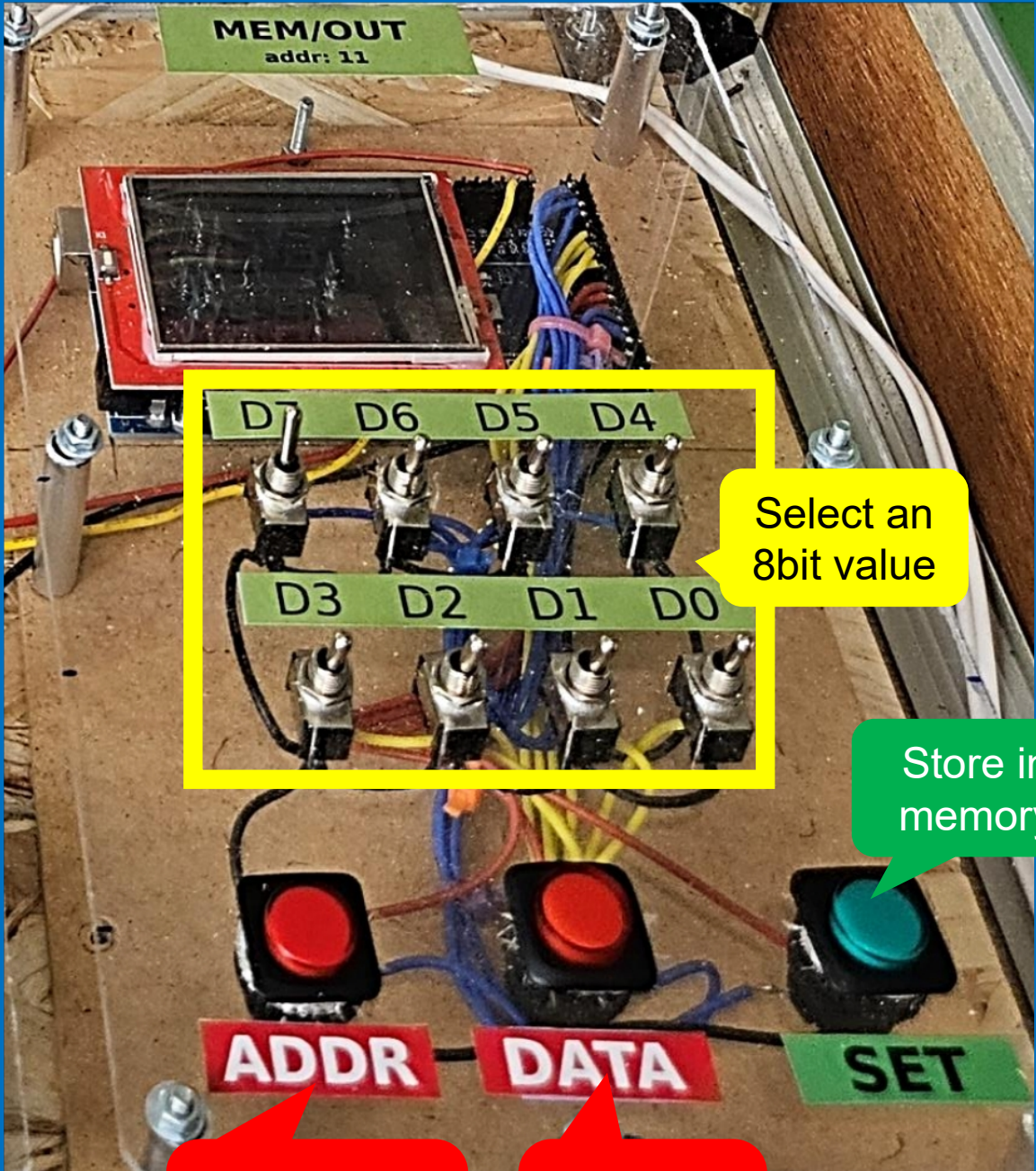
Control Unit
addr: 10

ALU/SR
addr: 9

ADDR DATA SET

D7 D6 D5 D4
D3 D2 D1 D0

MEMORY-OUTPUT UNIT



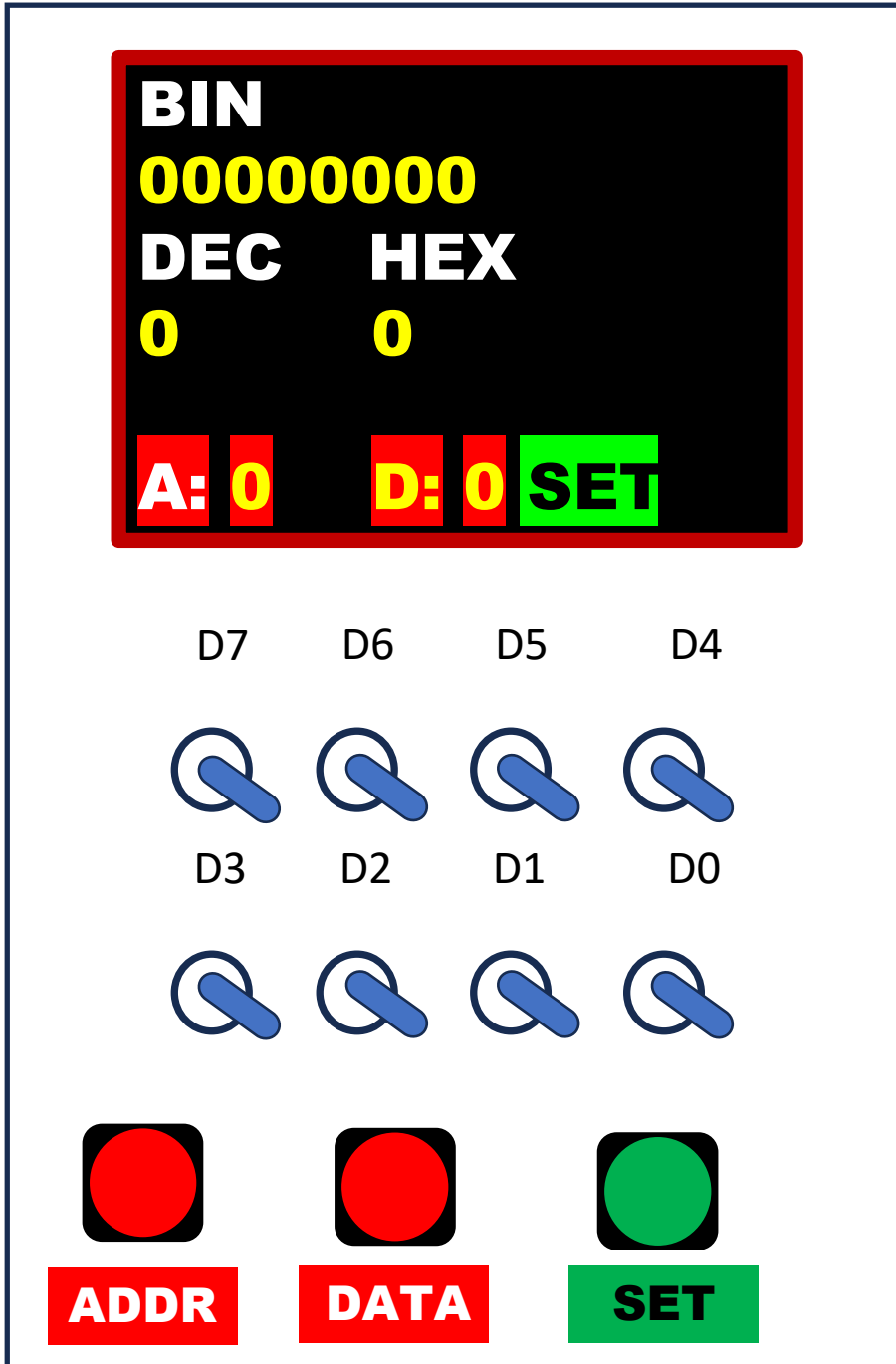
Select an 8bit value

Store in memory

Select 8bit value as ADDRESS

Select 8bit value as DATA

Storing data in memory



MEMORY-OUTPUT UNIT

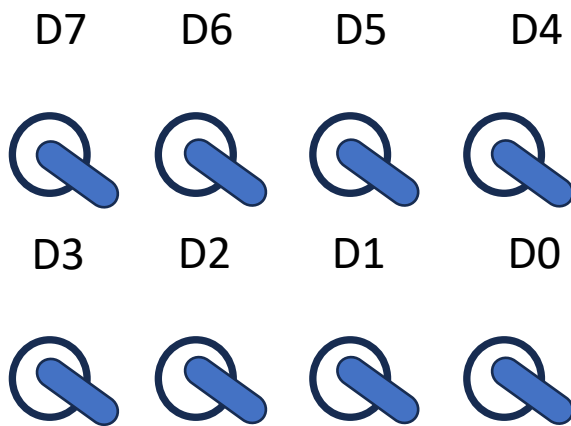
Memory

Example program

ADDR	DATA
0	4
1	8
2	10
3	0
4	17
5	0

Instruction	Byte code (decimal)	Address (content) (in decimal)
LOD A,8	(dec) 04 08, (hex) 04 08	00 (04), 01 (08)
INC A	(dec) 10 00, (hex) 0A 00	02 (10), 03 (00)
STOP	(dec) 17 00, (hex) 11 00	04 (17), 05 (00)

STEP 1 – Set all switches down

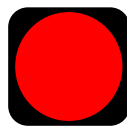


BIN
00000000

DEC **HEX**
0 0

A: 0 **D: 0** **SET**

STEP 2 – Press ADDR button



ADDR

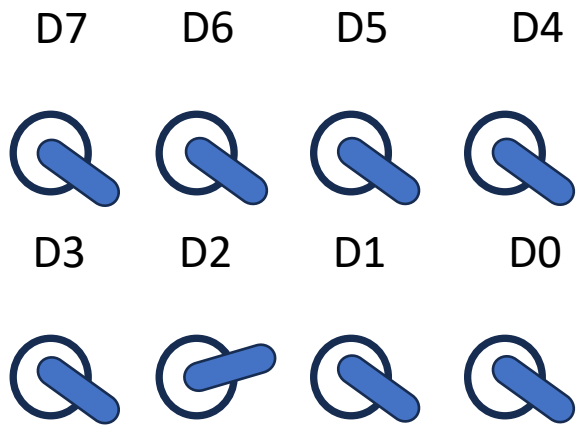
BIN
00000000

DEC **HEX**
0 0

A: 0 **D: 0** **SET**

Set 0 as
address

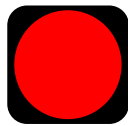
STEP 3 – Set switches for number 4



Number 4 is displayed on screen

BIN
00000000
DEC **HEX**
4 4
A: 0 **D: 0** **SET**

STEP 4 – Press DATA button



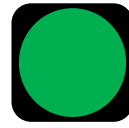
DATA

Set 4 as data

BIN
00000000
DEC **HEX**
4 4
A: 0 **D: 4** **SET**

STEP 5 – Store data in memory

Press →



SET



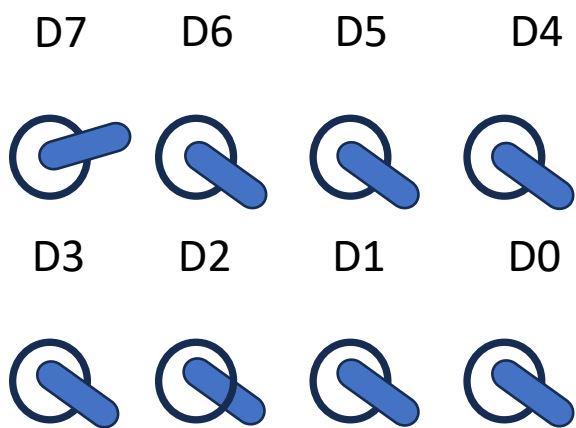
The color of SET indication will be changed to blue momentarily

Procedure to store bytes in memory

1. Set switches in the binary representation of the desired **ADDRESS** (confirm value on screen)
2. Press **ADDR** button to set the above value as **ADDRESS** (field A: on screen)
3. Set switches in the binary representation of the desired **DATA** (confirm value on screen) – byte that will be stored in the above **ADDRESS**
4. Press **DATA** button to set the above value as **DATA** (field D: on screen)
5. Press **SET** button to store DATA in the desired **ADDRESS** – Confirm the momentarily color change of the button SET

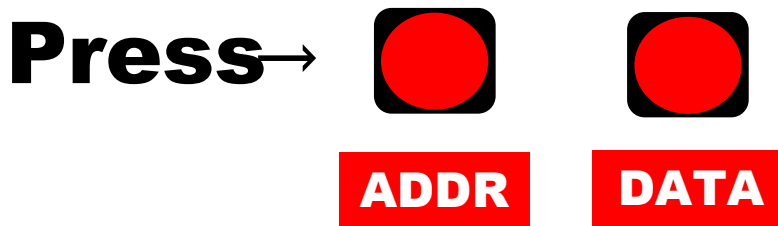
Display memory contents (1)

Change switches position for forming the number 128



BIN	
10000000	
DEC	HEX
128	80
A: 0	D: 0 SET

Press successive the buttons:

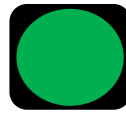


Confirm values on screen

BIN	
10000000	
DEC	HEX
128	80
A: 128	D: 128 SET

Display memory contents (2)

Press →



SET

ADDR	DATA	ADDR	DATA
0	4	11	0
1	8	12	0
2	10	13	0
3	0	14	0
4	17	15	0
5	0	16	0
6	0	17	0
7	0	18	0
8	0	19	0
9	0	20	0
10	0	21	0

EXIT

Press **SET** button to exit

* The above data represent the bytes of the example program

How to avoid manual data entry

- The data entry in memory is painful especially when the program has many bytes
- The program can be stored automatically in memory, avoiding the manual procedure
- For automatic data entry, the needed program bytes have to be inserted directly into the memory/output unit source code
- Insert program bytes by declaring an array of bytes:

```
byte prog[]={4, 8, 10, 0, 6, 0, 14, 0, 17, 0};
```

Demo program		
Instruction	Byte code (decimal)	Address (content) (in decimal)
LOD A,8	(dec) 04 08, (hex) 04 08	00 (04), 01 (08)
INC A	(dec) 10 00, (hex) 0A 00	02 (10), 03 (00)
MOV B,A	(dec) 06 00, (hex) 06 00	04 (06), 05 (00)
ADD A,B	(dec) 14 00, (hex) 0E 00	06 (14), 07 (00)
STOP	(dec) 17 00, (hex) 11 00	08 (17), 09 (00)

Create a new function for storing bytes from the `prog[]` array to memory array (`mem[]`). This function has to be inserted at the beginning of the `setup()` section.

```
void set_prog()  
{  
    for(int i=0;i<10;i++) mem[i]=prog[i];  
}
```