

BÁO CÁO THỰC HÀNH MARIE ASSEMBLY

1. Giới thiệu

Trong bài thực hành này, mục tiêu là làm quen với ngôn ngữ Assembly trên **MARIE Simulator**, một công cụ giả lập đơn giản để nghiên cứu kiến trúc máy tính.

Ba ví dụ được thực hiện gồm:

1. Mô phỏng tính tổng thời gian xử lý của nhiều tiến trình.
2. Tính tổng hai số nhập từ bàn phím.
3. Tính trung bình thời gian xử lý của ba tiến trình.

#Các lệnh cơ bản thường dùng:

Trong MARIE Assembly, có một tập nhỏ từ khóa (lệnh) và ký hiệu. Ý nghĩa ngắn gọn để học code:

Lệnh điều khiển (Instructions)

- ORG n → Đặt địa chỉ bắt đầu của chương trình tại ô nhớ n.
- LOAD X → Nạp giá trị tại biến/ô nhớ X vào thanh ghi tích lũy (AC).
- STORE X → Lưu giá trị trong AC vào biến/ô nhớ X.
- ADD X → Cộng giá trị tại X vào AC.
- SUBT X → Trừ giá trị tại X khỏi AC.
- INPUT → Nhập một giá trị từ bàn phím vào AC.
- OUTPUT → Xuất giá trị trong AC ra màn hình.
- HALT → Dừng chương trình.

Điều khiển luồng (Flow Control)

- JUMP LABEL → Nhảy đến nhãn LABEL (địa chỉ lệnh mới).
- SKIPCOND 000 → Nếu AC < 0 thì bỏ qua lệnh kế tiếp.
- SKIPCOND 400 → Nếu AC = 0 thì bỏ qua lệnh kế tiếp.
- SKIPCOND 800 → Nếu AC > 0 thì bỏ qua lệnh kế tiếp.

(Ba dạng SKIPCOND kiểm tra điều kiện để điều khiển vòng lặp / rẽ nhánh.)

Định nghĩa dữ liệu

- LABEL, DEC value → Khai báo một biến với giá trị số thập phân value.
- LABEL, HEX value → Khai báo một biến với giá trị hexa.
- LABEL, DEC 0 → Khai báo biến khởi tạo = 0 (chỗ chứa dữ liệu).

Ví dụ:

X1, DEC 5 / Khai báo biến X1 = 5

ONE, DEC 1 / Khai báo biến ONE = 1

TEMP, DEC 0 / Biến tạm khởi tạo bằng 0

Nhãn (Labels)

- Ví dụ LOOP1, SUBT ONE → LOOP1 là nhãn, giúp đặt điểm để JUMP quay lại.
- Không phải lệnh, chỉ là tên để đánh dấu dòng code.

👉 Tóm lại:

- LOAD / STORE / ADD / SUBT: xử lý dữ liệu.
- INPUT / OUTPUT: nhập xuất.
- JUMP / SKIPCOND: điều khiển vòng lặp, rẽ nhánh.
- DEC / HEX: khai báo biến.
- ORG / HALT: quản lý chương trình.
- Bảng tóm tắt lệnh MARIE Assembly:

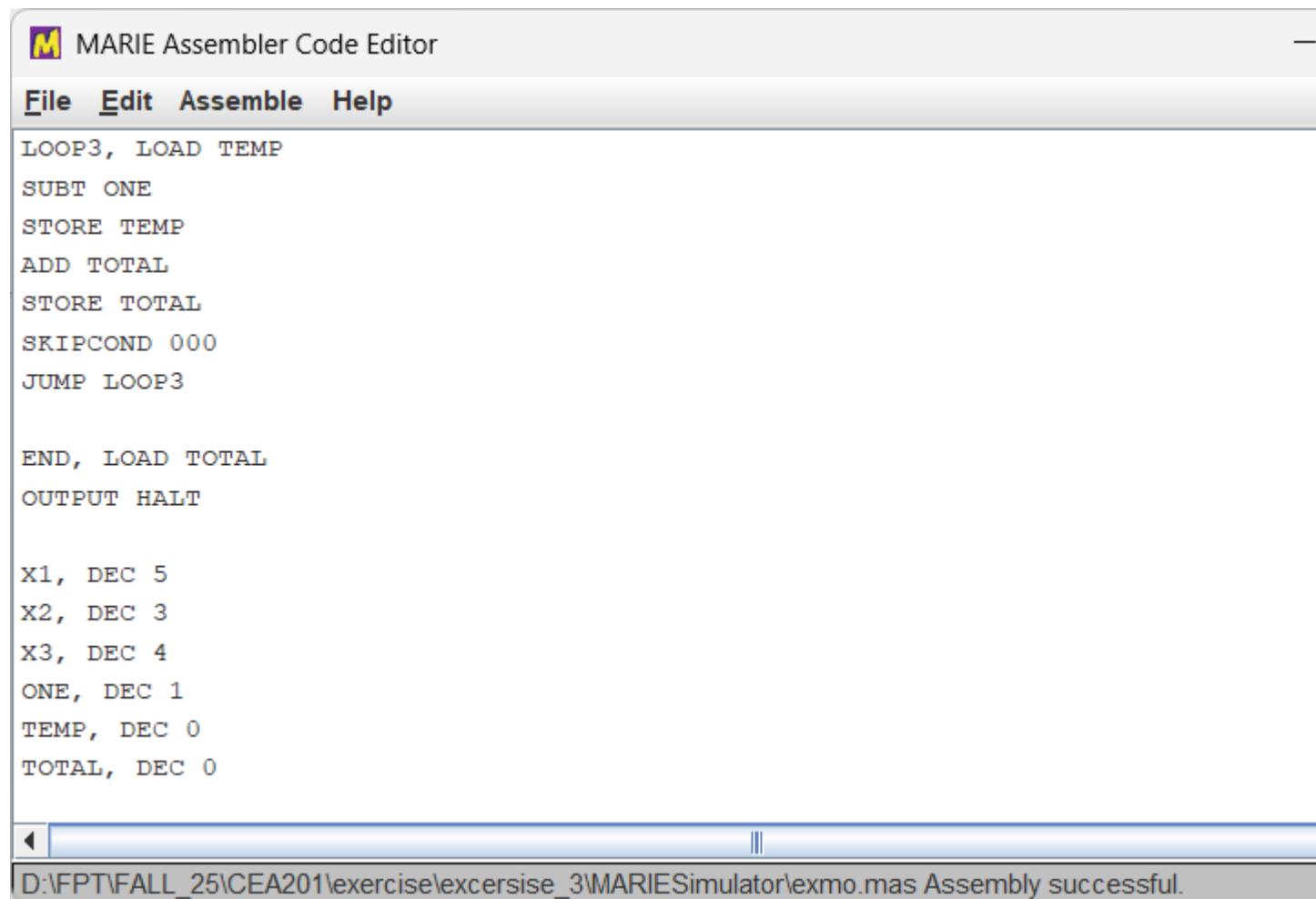
Lệnh / Tùy chọn	Ý nghĩa	Ví dụ
ORG n	Đặt địa chỉ bắt đầu chương trình tại ô nhớ n.	ORG 100
LOAD X	Nạp giá trị tại ô nhớ/biến x vào thanh ghi AC.	LOAD A
STORE X	Lưu giá trị trong AC vào ô nhớ/biến x.	STORE RESULT
ADD X	Cộng giá trị tại x vào AC.	ADD B
SUBT X	Trừ giá trị tại x khỏi AC.	SUBT ONE
INPUT	Nhập giá trị từ bàn phím vào AC.	INPUT
OUTPUT	Xuất giá trị trong AC ra màn hình.	OUTPUT
HALT	Dừng chương trình.	HALT

JUMP LABEL	Nhảy tới nhãn LABEL.	JUMP LOOP
SKIPCOND 000	Nếu AC < 0 thì bỏ qua lệnh kế tiếp.	SKIPCOND 000
SKIPCOND 400	Nếu AC = 0 thì bỏ qua lệnh kế tiếp.	SKIPCOND 400
SKIPCOND 800	Nếu AC > 0 thì bỏ qua lệnh kế tiếp.	SKIPCOND 800
LABEL, DEC value	Khai báo biến với giá trị thập phân value.	X, DEC 5
LABEL, HEX value	Khai báo biến với giá trị hệ 16.	Y, HEX 1F
LABEL, DEC 0	Khai báo biến khởi tạo bằng 0 (biến trống).	TEMP, DEC 0
LABEL,	Nhãn (dùng để đánh dấu, kết hợp với JUMP).	LOOP1, SUBT ONE

2. Ví Du thực hành:

+Ví dụ 1: Tính tổng thời gian xử lý 3 process

Code:



The screenshot shows the MARIE Assembler Code Editor interface. The menu bar includes File, Edit, Assemble, and Help. The assembly code is as follows:

```

MARIE Assembler Code Editor

File Edit Assemble Help

LOOP3, LOAD TEMP
SUBT ONE
STORE TEMP
ADD TOTAL
STORE TOTAL
SKIPCOND 000
JUMP LOOP3

END, LOAD TOTAL
OUTPUT HALT

X1, DEC 5
X2, DEC 3
X3, DEC 4
ONE, DEC 1
TEMP, DEC 0
TOTAL, DEC 0

```

The status bar at the bottom displays the message "D:\FPT\FALL_25\CEA201\exercise\excersise_3\MARIESimulator\exmo.mas Assembly successful."

Giải thích:

- X1, X2, X3 lần lượt là thời gian xử lý của 3 tiến trình.
- Vòng lặp LOOP1, LOOP2, LOOP3: trừ dần từng tiến trình về 0.
- Mỗi lần trừ 1, cộng thêm 1 vào TOTAL.
- Cuối chương trình, biến TOTAL được xuất ra bằng lệnh OUTPUT.

Kết quả chạy:

- Tổng thời gian xử lý = **12** ($5 + 3 + 4$).

Ý nghĩa:

- Mô phỏng cách tính tổng thời gian CPU khi xử lý nhiều tiến trình.

M Assembly Listing for exmo.mas

```

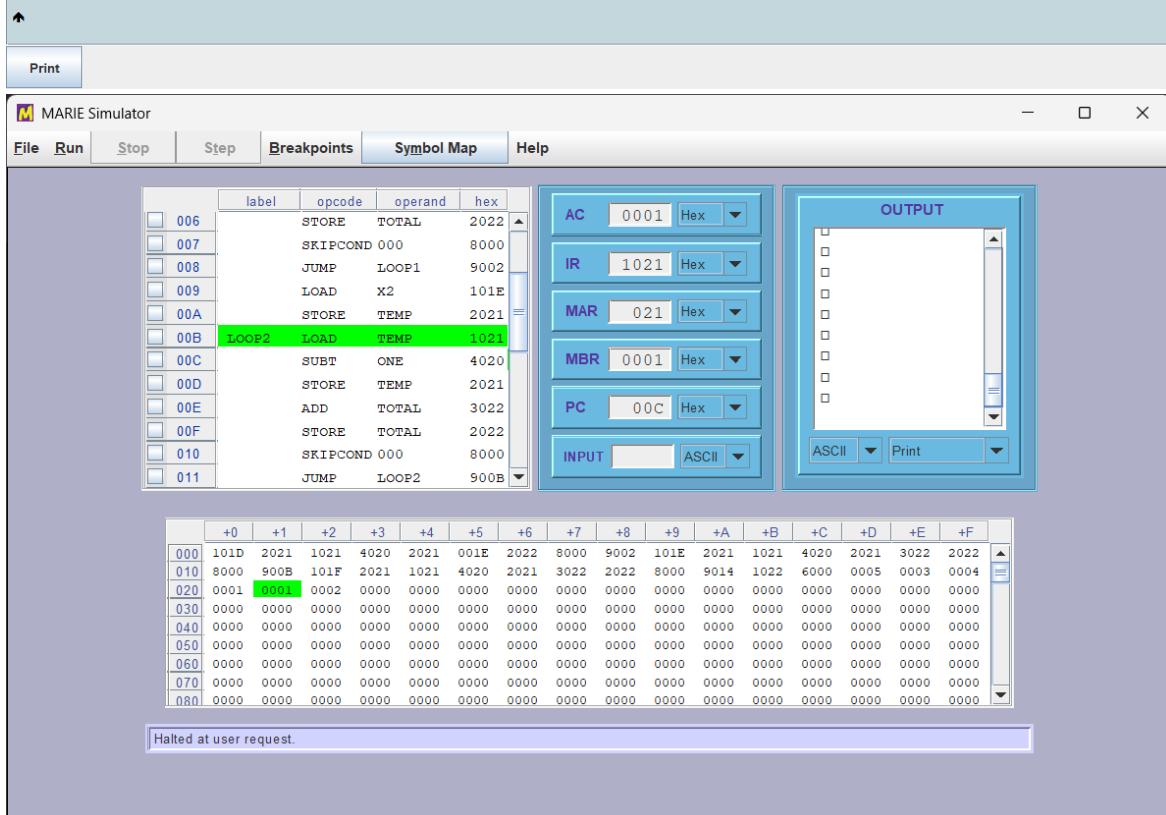
016 2021 |      STORE TEMP
017 3022 |      ADD TOTAL
018 2022 |      STORE TOTAL
019 8000 |      SKIPCOND 000
01A 9014 |      JUMP LOOP3
|
01B 1022 | END   LOAD TOTAL
01C 6000 |      OUTPUT
|
01D 0005 | X1    DEC 5
01E 0003 | X2    DEC 3
01F 0004 | X3    DEC 4
020 0001 | ONE   DEC 1
021 0000 | TEMP   DEC 0
022 0000 | TOTAL  DEC 0

```

Assembly successful.

SYMBOL TABLE

Symbol	Defined	References
END	01B	
LOOP1	002	008
LOOP2	00B	011
LOOP3	014	01A
ONE	020	003, 00C, 015
TEMP	021	001, 002, 004, 00A, 00B, 00D, 013, 014, 016
TOTAL	022	005, 006, 00E, 00F, 017, 018, 01B
X1	01D	000
X2	01E	009
X3	01F	012



3. Ví dụ 2: Tính tổng 2 số nhập từ bàn phím

Code:

The screenshot shows the MARIE Assembler Code Editor window. The menu bar includes File, Edit, Assemble, and Help. The assembly code in the editor is as follows:

```
ORG 100
INPUT      / Nhập số A
STORE A
INPUT      / Nhập số B
STORE B
LOAD A
ADD B
STORE SUM
OUTPUT    HALT
          / Xuất kết quả

A, DEC 0
B, DEC 0
SUM, DEC 0
```

In the status bar at the bottom, it says "D:\FPT\FALL_25\CEA201\exercise\excercise_3\MARIESimulator\exmo.mas Assembly successful."

Giải thích:

- Người dùng nhập 2 số nguyên từ bàn phím.
- Chương trình cộng 2 số này lại.
- Xuất kết quả bằng OUTPUT.

Kết quả chạy:

- Nếu nhập 7 và 9, OUTPUT = **16**.

Ý nghĩa:

- Rèn luyện cách dùng lệnh nhập/xuất (I/O) và phép cộng cơ bản trong MARIE.

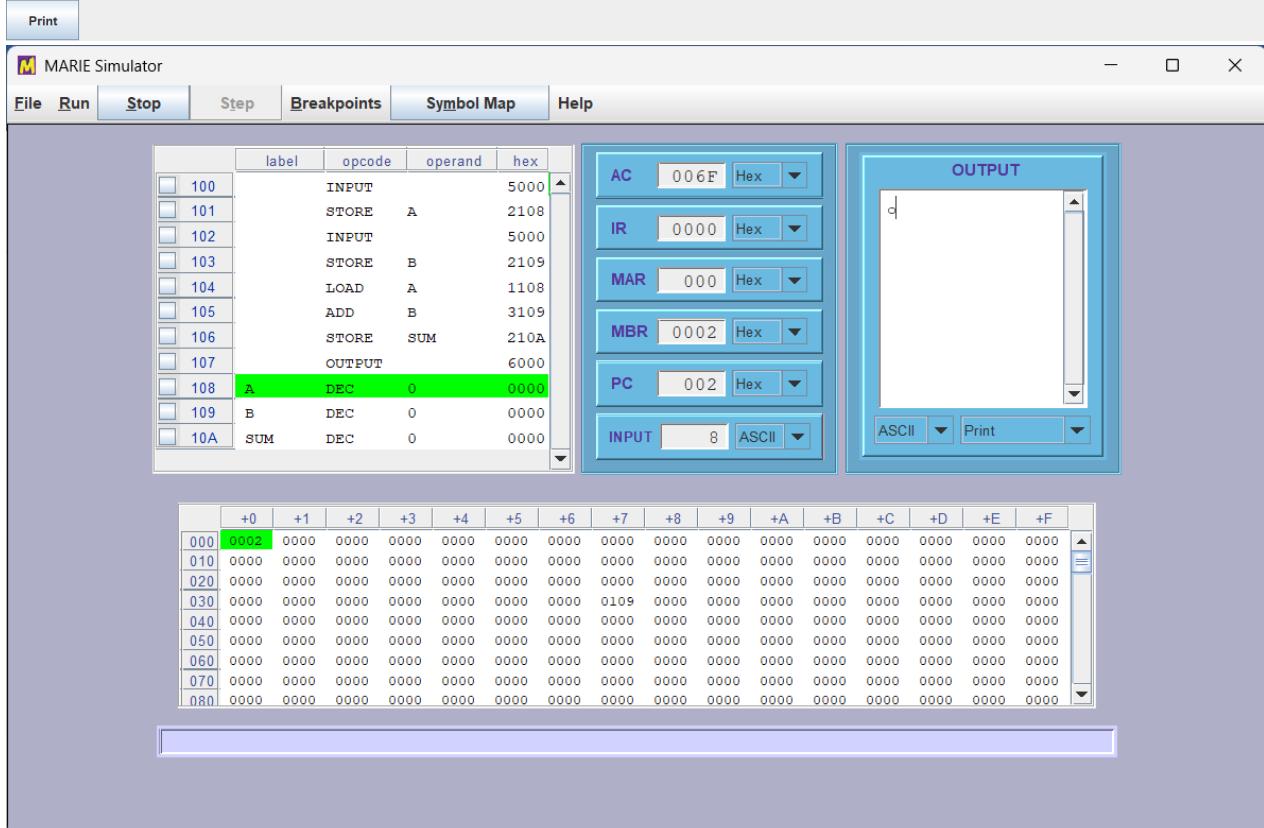
```
M Assembly Listing for exmo.mas
Assembly listing for: exmo.mas
Assembled: Sat Sep 27 13:52:32 ICT 2025

|      ORG 100
100 5000 |      INPUT          / Nh?p s? A
101 2108 |      STORE A
102 5000 |      INPUT          / Nh?p s? B
103 2109 |      STORE B
104 1108 |      LOAD A
105 3109 |      ADD B
106 210A |      STORE SUM
107 6000 |      OUTPUT
|
|      / Xu?t k?t qu?
108 0000 |  A      DEC 0
109 0000 |  B      DEC 0
10A 0000 |  SUM    DEC 0
```

Assembly successful.

SYMBOL TABLE

Symbol	Defined	References
A	108	101, 104
B	109	103, 105
SUM	10A	106



4. Ví dụ 3: Tính trung bình 3 process

Code:

The screenshot shows the MARIE Assembler Code Editor window. The menu bar includes File, Edit, Assemble, and Help. The code area contains the following assembly instructions:

```
ADD X2
ADD X3
STORE SUM
LOAD SUM
SUBT THREE
STORE SUM
LOAD SUM
SUBT THREE
STORE SUM
LOAD SUM
OUTPUT HALT

X1, DEC 5
X2, DEC 3
X3, DEC 4
SUM, DEC 0
THREE, DEC 3
```

The status bar at the bottom displays the message "D:\FPT\FALL_25\CEA201\exercise\excercise_3\MARIESimulator\exami.mas Assembly successful."

M Assembly Listing for exmo.mas

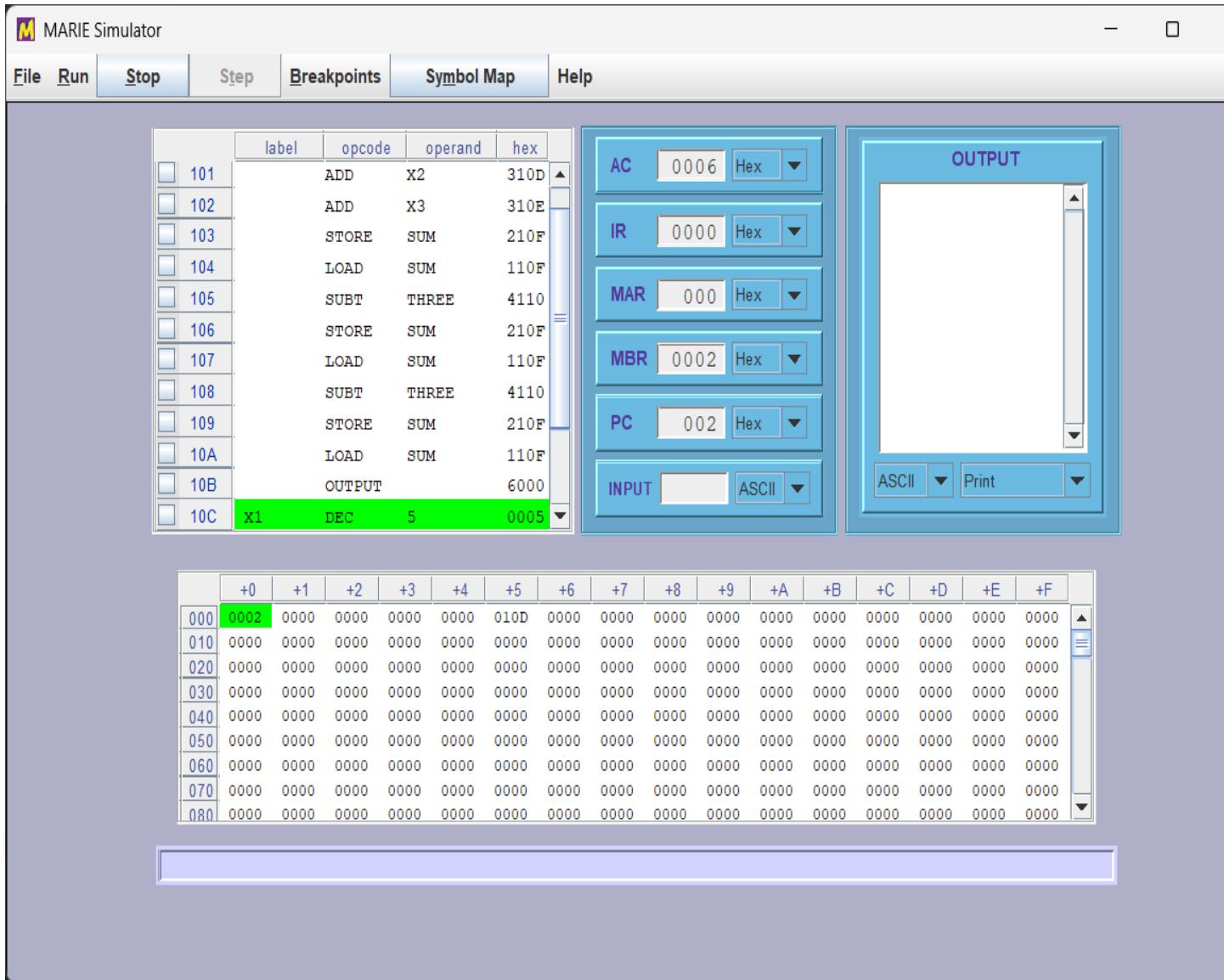
Assembly listing for: exmo.mas
Assembled: Sat Sep 27 13:47:26 ICT 2025

```
          |      ORG 100
100 110C |      LOAD X1
101 310D |      ADD X2
102 310E |      ADD X3
103 210F |      STORE SUM
104 110F |      LOAD SUM
105 4110 |      SUBT THREE
106 210F |      STORE SUM
107 110F |      LOAD SUM
108 4110 |      SUBT THREE
109 210F |      STORE SUM
10A 110F |      LOAD SUM
10B 6000 |      OUTPUT
|
10C 0005 |  X1    DEC 5
10D 0003 |  X2    DEC 3
10E 0004 |  X3    DEC 4
10F 0000 |  SUM   DEC 0
110 0003 |  THREE DEC 3

Assembly successful.

SYMBOL TABLE
-----
Symbol | Defined | References
-----+-----+-----+
SUM   | 10F   | 103, 104, 106, 107, 109, 10A
THREE | 110   | 105, 108
X1    | 10C   | 100
X2    | 10D   | 101
X3    | 10E   | 102
-----+
```

Print **Close**



(Lưu ý: MARIE không có lệnh chia trực tiếp, nên phép chia 3 được mô phỏng bằng cách trừ dần 3 cho đến khi gần bằng 0. Đây là phương án thủ công minh họa.)

Giải thích:

- Cộng X1, X2, X3 vào biến SUM.
- Thực hiện phép chia 3 bằng cách trừ 3 liên tục (ở đây minh họa số nhỏ).
- Xuất kết quả trung bình.

Kết quả chạy:

- Với $X_1=5$, $X_2=3$, $X_3=4 \rightarrow$ Trung bình ≈ 4 .

Ý nghĩa:

- Mô phỏng cách xử lý phép chia và tính trung bình trong môi trường Assembly hạn chế.

5. Kết luận

- Thông qua 3 ví dụ, ta thấy MARIE Assembly có thể được dùng để mô phỏng các phép tính số học và xử lý tiến trình đơn giản.
- Các lệnh cơ bản được sử dụng: **LOAD, STORE, ADD, SUBT, SKIPCOND, JUMP, INPUT, OUTPUT, HALT**.
- Bài thực hành giúp hiểu rõ hơn cách CPU giả lập hoạt động và là nền tảng để tiếp cận các kiến thức nâng cao về kiến trúc máy tính và hệ điều hành.