#### ĐẠI HỌC CÔNG THƯƠNG TP. HỒ CHÍ MINH KHOA CÔNG NGHỆ ĐIỆN – ĐIỆN TỬ BỘ MÔN TỰ ĐỘNG HÓA

-----



#### CUỐI KỲ THIẾT KẾ SỐ

## TÊN ĐỀ TÀI NGHIÊN CỬU RISC 8 BIT MÔ PHỎNG LÊN VIVADO

SVTH: LÊ THẠCH ANH TÀI

MSHV: 2002210206

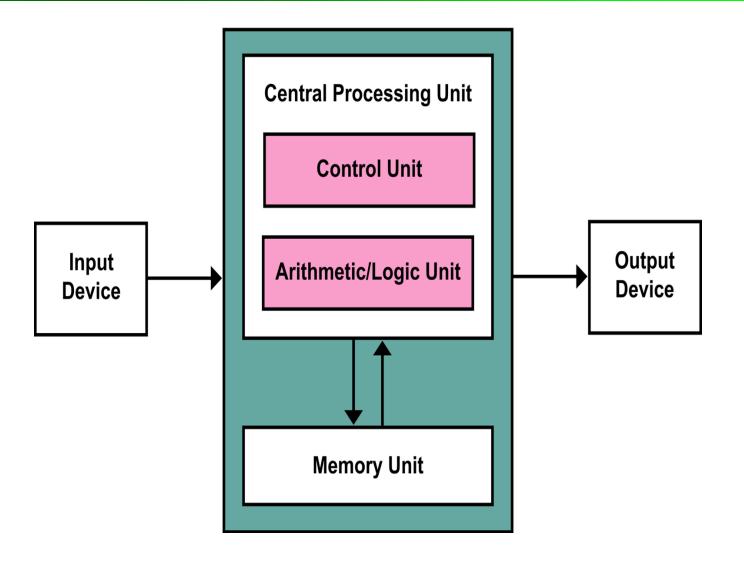
- I. KIÉN TRÚC VONEUMAN
- II. CHIPA12
- III. 3 KHỐI CHỰC NĂNG
- IV. KẾT QUẢ THỰC NGHIỆM



## KIÉN TRÚC VONEUMAN

## Kiến trúc Von Neumann

Hiện nay trên thới giới bất kỳ CPU nào bên trong đều là kiến trúc Von Neumann. Gồm 3 khối Control Unit, Processor, Memory Unit

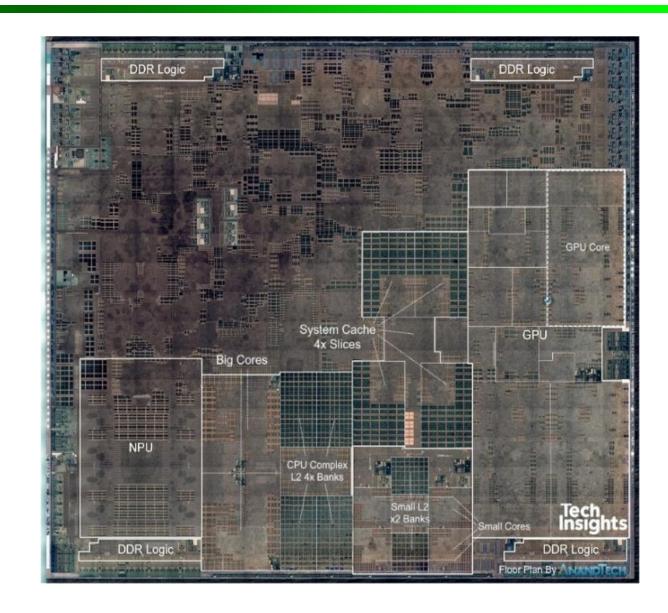


TP. HCM, 15/01/2025



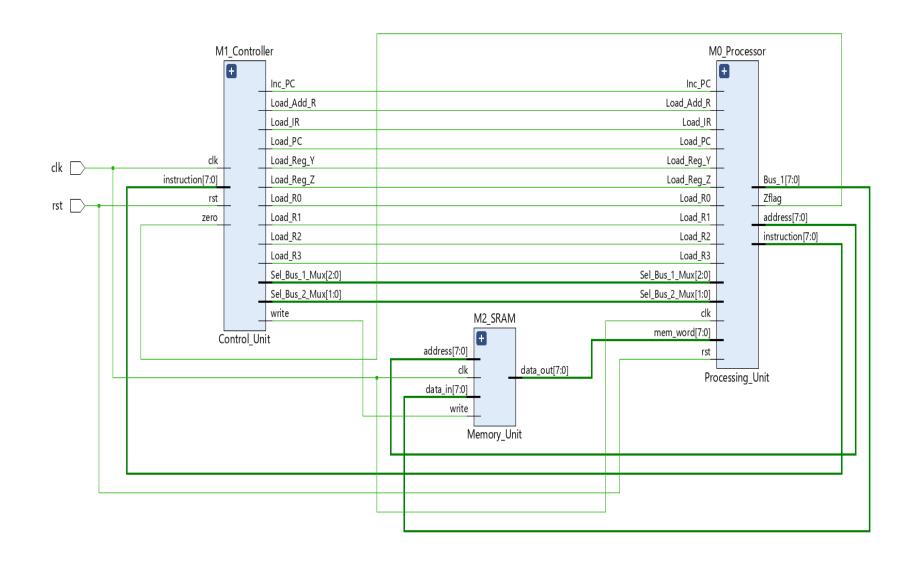
#### II. CHIPA12

- CPU và bộ nhớ chip A12
- Chip A12 trên iphone XS ta thấy có CPU và bộ nhớ Cache.
- CPU chứa 2 khối điều khiển và khối xử lý
- Bộ nhớ Cache cấu tạo từ các SRAM



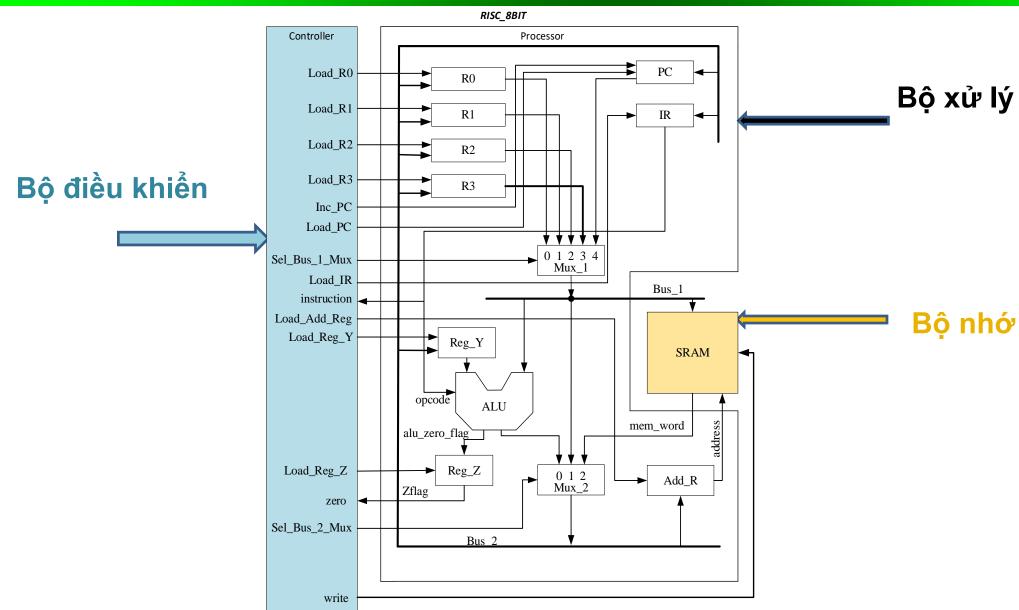


## 3 KHỐI CHỨC NĂNG



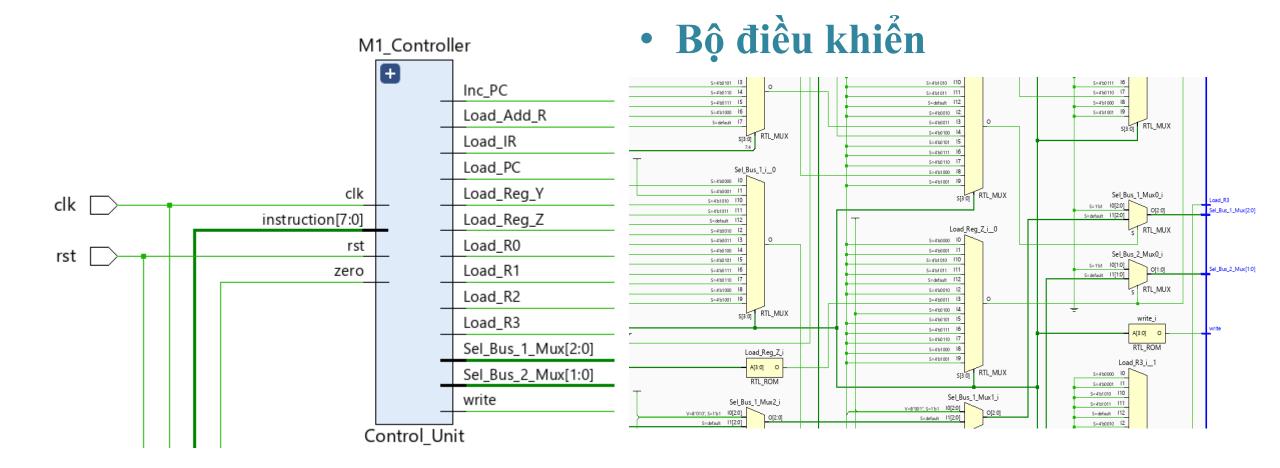


## III. 3 KHỐI CHỨC NĂNG





## BỘ ĐIỀU KHIỂN





## III.

## **BỘ NHỚ TĨNH**

### • Bộ nhớ nhanh

- Bao gồm 255 địa chỉ, mỗi địa chỉ chứa 8bit.

	SRAM				
data_in	ADDRESS	DATA			
	0	00010100			
	1	11111111			
	2	00000000			
	3	01010101			
	4	00000000			
	5	00000000			
	6	00000000			
	7	00000000			
	8	00000000			
address	9	00000000			
write	10	00000000			
data_out					
	255	255			

TP. HCM, 15/01/2025



## III.

# CÂU LỆNH 8 BIT

#### • RISC 8bit

Lệnh ngắn

Opcode				Nguồn		Ðích	
0	0	0	1	0	0	1	0

Lệnh dài

Opcode			Nguồn		Ðích		
0	1	0	1	1	0	ı	1
			Địa	chỉ			
0	0	0	1	1	1	0	1

		Lệnh 8 bit			
Lệnh	opcode	nguồn	đích	Hành động	
NOP	0000			không làm gì	
ADD	0001	nguồn	đích	đích <= nguồn +đích	
SUB	0010	nguồn	đích	đích <= đích − nguồn	
AND	0011	nguồn	đích	đích <= nguồn & đích	
NOT	0100	nguồn	đích	đích <= ~ nguồn	
RD	0101		đích	Lấy giá trị từ ô nhớ trong địa chỉ lưu vào đích	
WR	0110	nguồn		Đọc giá trị từ ô nhớ trong địa chỉ lưu vào nguồn	
BR	0111			Đọc giá trị từ ô nhớ trong địa chỉ lưu vào PC	
BRZ	1000			Đọc giá trị từ ô nhớ trong địa chỉ lưu vào PC	
HALT	111			dừng tất cả lại đến khi reset	

TP. HCM, 15/01/2025

Các giá trị trong ngoặc vuông là các giá trị địa chỉ

 $SRAM[0] = 8'b0101\_00\_10$ 

SRAM [1] = 130;

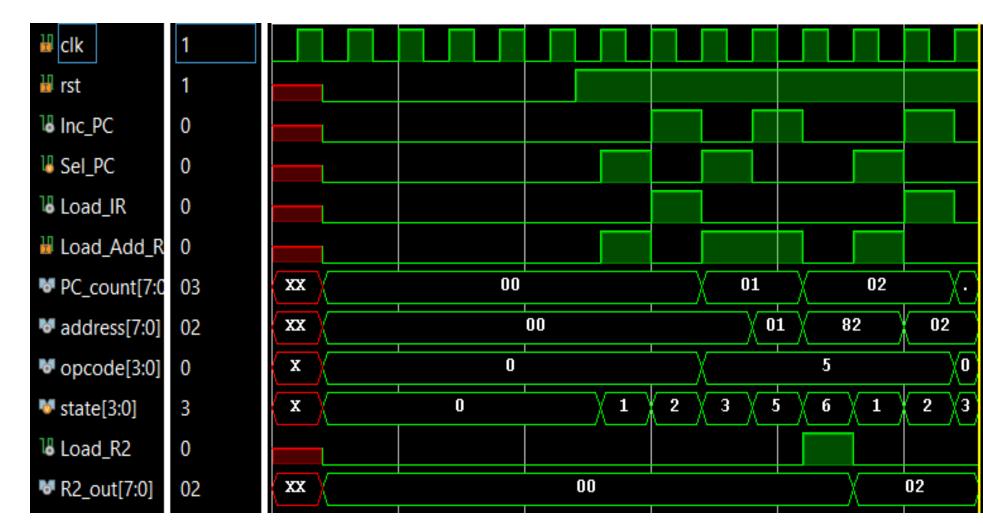
SRAM [130] = 2;

Nạp giá trị 2 ở ô nhớ số 130

vào thanh ghi đa năng R2

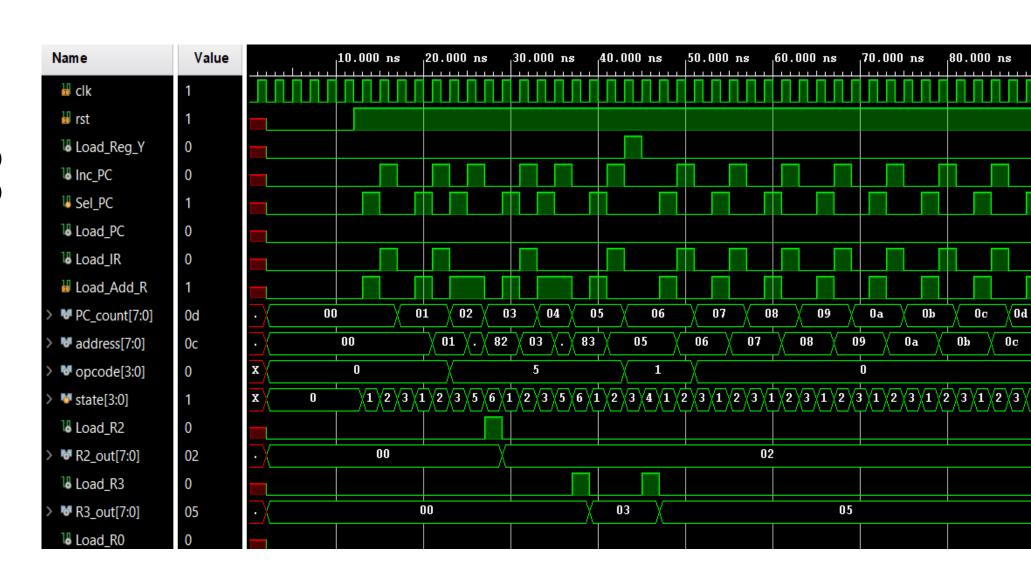
tương ứng với 2 bit cuối 10 ở

địa chỉ 0



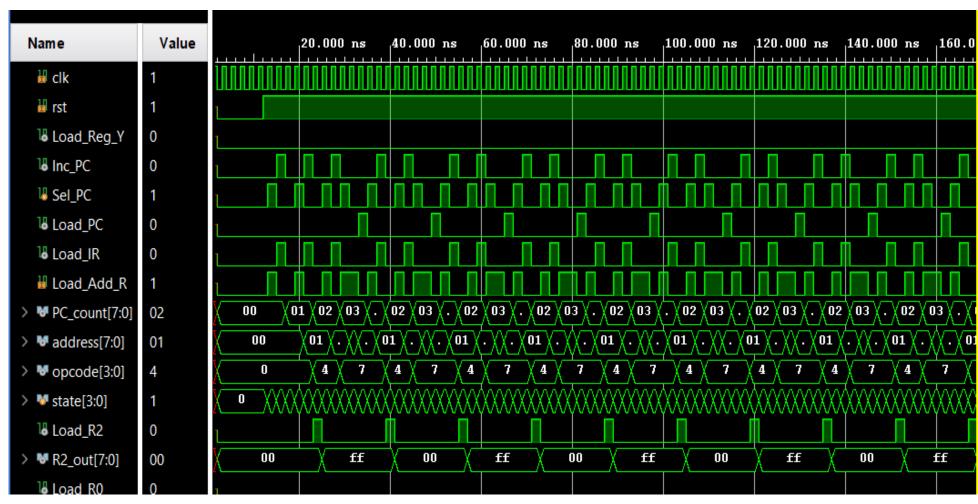
## LÊNH ADD

$SRAM[0] = 0000\_00\_00$
SRAM[1] = 0101_00_10
SRAM[2] = 130
SRAM[130] = 2
SRAM[3] = 0101_00_11
SRAM[4] = 131
SRAM[131] = 3
SRAM[5] = 0001 10 11



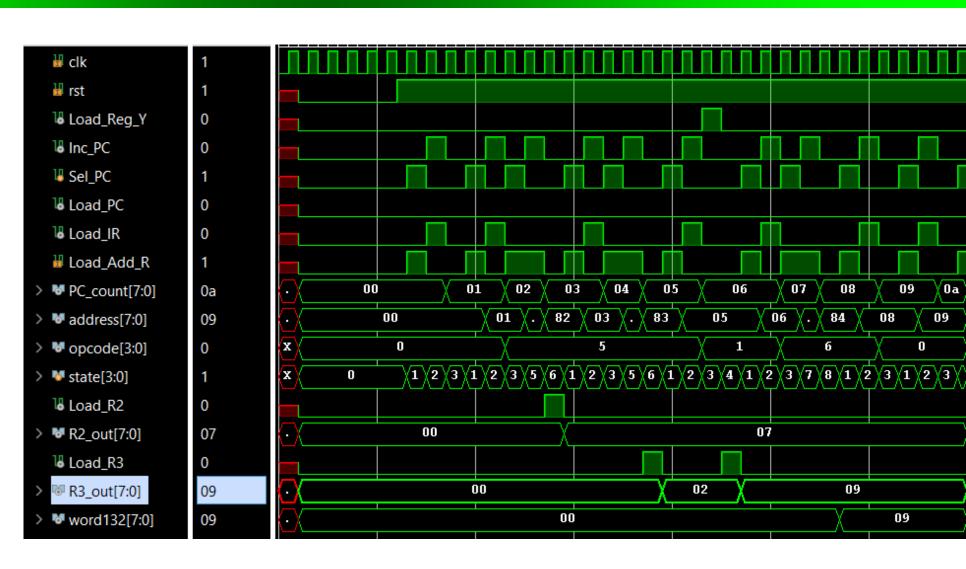
SRAM[0]= 8'b0000\_00\_00 SRAM[1]= 8'b0100\_10\_10 SRAM[3] =8'b0111\_00\_00 SRAM[4] = 140 SRAM[140] = 1

00000000 =00 ~ 11111111 =ff

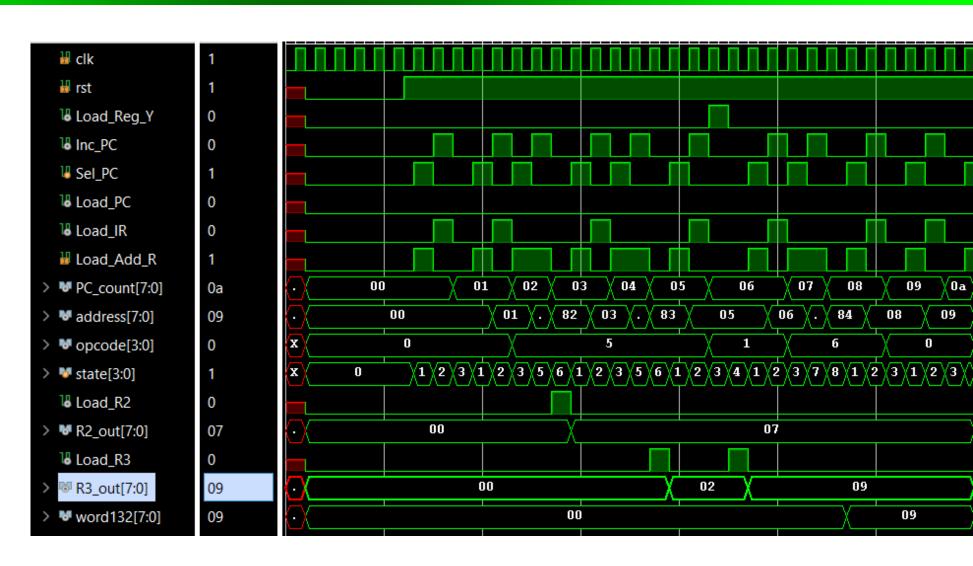


## LÊNH GHI (WR)

```
SRAM[0] = 8'b0000\_00\_00;
SRAM[1] = 8'b0101\_00\_10;
SRAM[2] = 130;
SRAM[3] = 8'b0101\_00\_11;
SRAM[4] = 131;
SRAM[5] = 8'b0001\_10\_11;
SRAM[6] = 8'b0110_11_00;
SRAM[7] = 132;
SRAM[130] = 7;
SRAM[131] = 2;
SRAM[132] = 0;
```



```
SRAM[0] = 8'b0000\_00\_00;
SRAM[1] = 8'b0101\_00\_10;
SRAM[2] = 130;
SRAM[3] = 8'b0101\_00\_11;
SRAM[4] = 131;
SRAM[5] = 8'b0001\_10\_11;
SRAM[6] = 8'b0110_11_00;
SRAM[7] = 132;
SRAM[130] = 7;
SRAM[131] = 2;
SRAM[132] = 0;
```



## KÉT QUẢ GHI

SRAM [0] = 8'b0000\_00\_00; // NOP

SRAM [1] = 8'b0101\_00\_10; // Đọc 130 lưu R2

SRAM [2] = 130;

SRAM [3] = 8'b0101\_00\_11; // Đọc 131 lưu R3

SRAM [4] = 131;

SRAM [5] = 8'b0101\_00\_01; // Đọc 128 lưu R1

SRAM [6] = 128;

SRAM [7] = 8'b0101\_00\_00; // Đọc 129 lưu R0

SRAM [8] = 129; // byte thu hai la lay dia chi

SRAM [9] = 8'b0010\_00\_01; // Trừ R1-R0 lưu R1

SRAM [10] = 8'b1000\_00\_00; // BRZ

SRAM [11] = 134; // Giữ địa chỉ cho BRZ

SRAM [12] = 8'b0001\_10\_11; // Cộng R2+R3 lưu R3

SRAM [13] = 8'b0111\_00\_11; // BR

SRAM [14] = 140;

// Load data

SRAM [128] = 6;

SRAM [129] = 1;

SRAM [130] = 2;

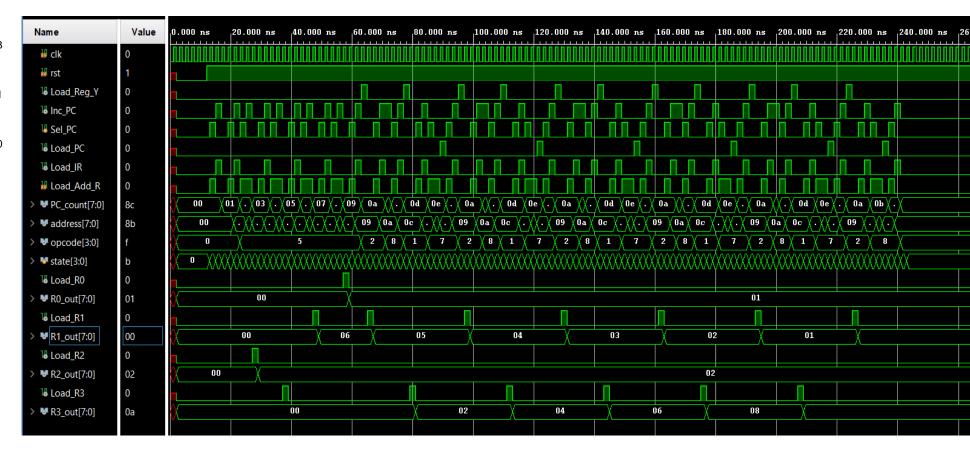
SRAM[131] = 0;

SRAM [134] = 139;

SRAM [139] = 8'b1111\_00\_00; // HALT

SRAM [140] = 9;

// Recycle





# CẨM ƠN CÁC ANH/CHỊ VÀ CÁC BẠN ĐÃ XEM

TP. HCM, 15/01/2025

SVTH: LE THACH ANH TAI