ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

TỔ CHỨC VÀ CẤU TRÚC MÁY TÍNH II

LỚP: IT012.N21.2

BÁO CÁO THỰC HÀNH SỐ 6 (LAB 06)

Giảng viên hướng dẫn: Nguyễn Thành Nhân

Sinh viên: Hồ Trọng Hiển

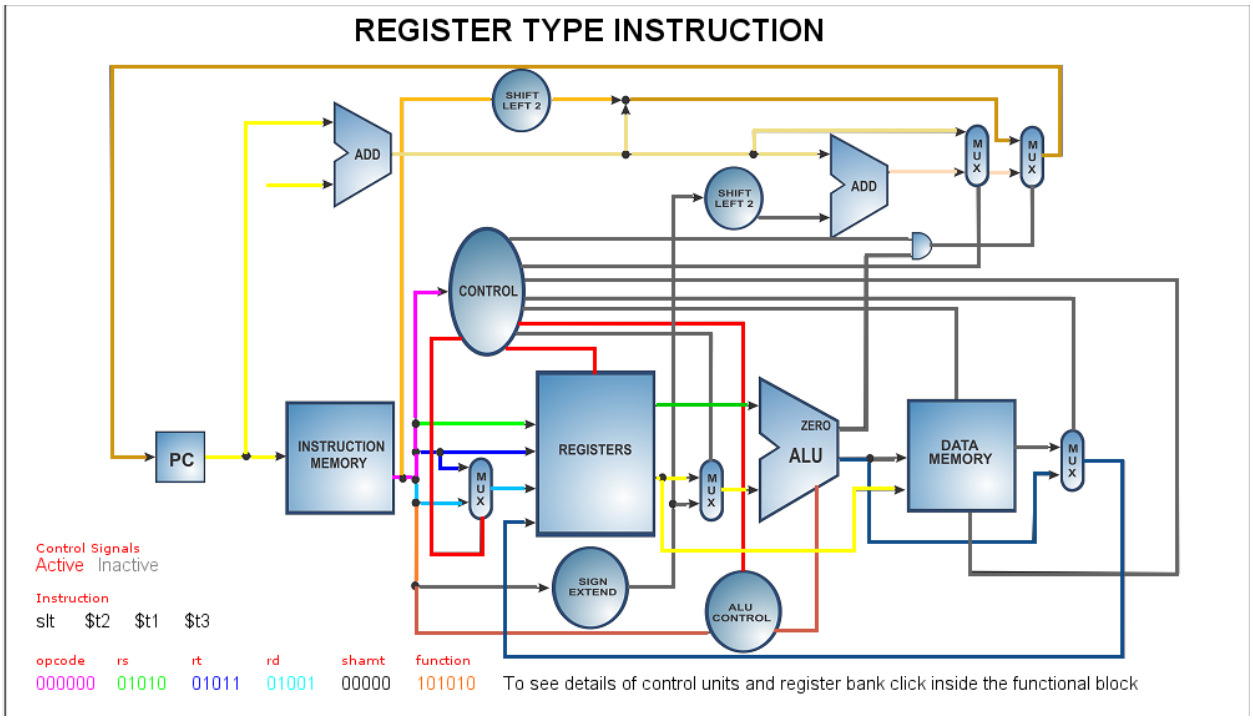
MSSV: 22520414

MỤC LỤC

Phần 3:

- Các lệnh còn lại sẽ được giải thích ở phần 4.

- slt $t2, $t1, $t3



Từ thanh ghi PC qua Instruction Memory, từ đó lệnh sẽ được nạp và truyền địa chỉ của từng thành phần đi. slt là R-type nên nó sẽ phân ra:

1. Đường màu tím chính là opcode (31:26), nó sẽ đi vào một khối Control để nó biết được lệnh đó thuộc loại lệnh I và sau đó nó đi đến các khối Register, ALU control, mux để giúp các khối đó làm việc đúng với định dạng lệnh đó.

2. Đường màu xanh lá (25:21) là địa chỉ của toán hạng nguồn thứ nhất và đi vào Register.

3. Đường màu lam (20:16) là địa chỉ của toán hạng nguồn thứ hai và đi vào Register.

4. Đường màu xanh biển (15-11) là địa chỉ của toán hạng đích rd, vì đây là R-type nên RegDst = 1. Qua đó qua bộ MUX sẽ chọn rd vào Register để sau này tiến hành ghi.

5. Sau khi đọc xong dữ liệu trong toán hạng nguồn 1 và 2 thì toán hạnh nguồn 1 sẽ được đẩy vào ALU còn ALUSrc = 0 để chọn toán hạng nguồn 2 đưa vào Register.

6. Lúc này trong ALU sẽ thực hiện phép toán trừ và kiểm tra hiệu vừa thực hiện, nếu là 0 hay một số dương thì lúc này sẽ trả kết quả 0 ra khỏi ALU, còn nếu hiệu âm thì trả ra 1.

7. MemtoReg = 0 nên sẽ lấy kết quả được trả thẳng từ ALU để quay lại Register vào phần WriteData.

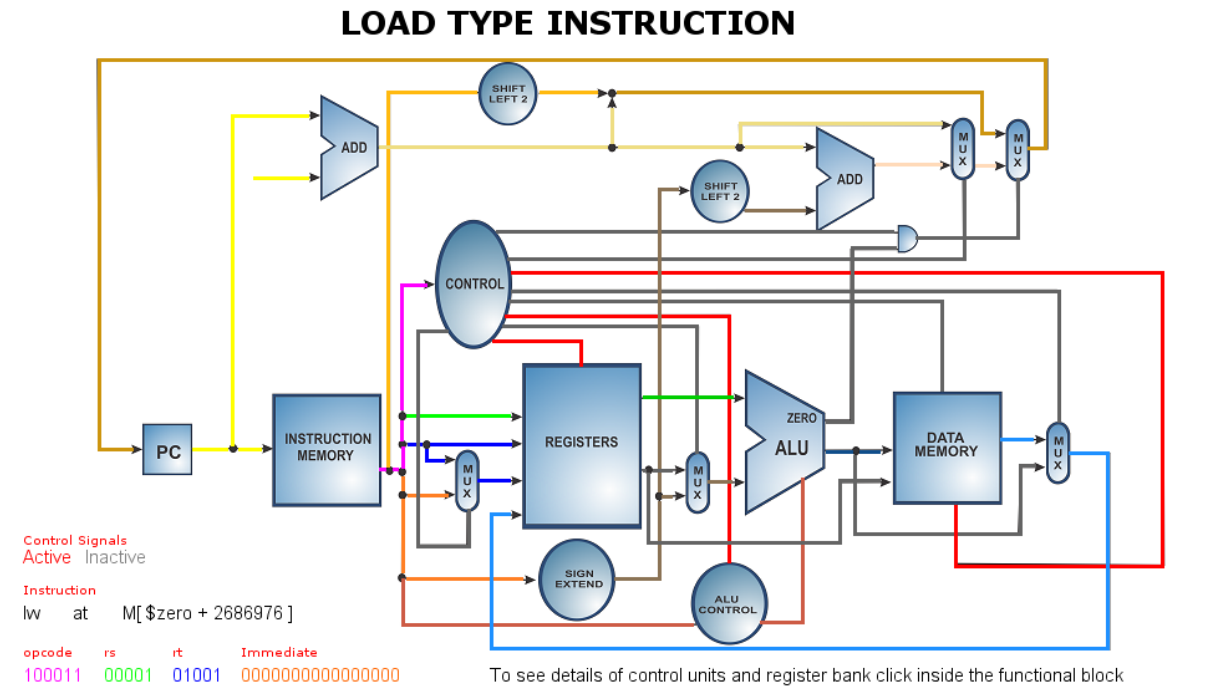
8. RegWrite = 1 nên tiến hành lưu kết quả vừa có vào toán hạng đích.

9. PC lúc này sẽ tăng lên 4 và được gán ngược lại PC, sang lệnh tiếp theo.

Phần 4.1:

```
        .data
a: .space 4
b: .space 4
c: .space 4
d: .space 4
        .text
lw $t1,a        # lenh lw thuoc loai I , dua theo bang MIPS Reference data
addi $t1,$0,6         # lenh addi thuoc loai I , dua theo bang MIPS Reference data
lw $t2,b        # lenh lw thuoc loai I , dua theo bang MIPS Reference data
addi $t2,$0,5         # lenh addi thuoc loai I , dua theo bang MIPS Reference data
lw $t3,c        # lenh lw thuoc loai I , dua theo bang MIPS Reference data
sub $t3,$t1,$t2       # lenh sub thuoc loai R , dua theo bang MIPS Reference data
lw $t4,d        # lenh lw thuoc loai I , dua theo bang MIPS Reference data
add $t4,$t1,$t2       # lenh add thuoc loai R , dua theo bang MIPS Reference data
```

```
lw   $t1 ,  a
```



- Trên đây là datapath của lệnh lw $t1,a và các lệnh lw trện cũng tương tự

- Giải thích:

• Đầu tiên là từ PC chính là địa chỉ của dòng lệnh đó đi tới Instruction memory

• Trong Instruction memory sẽ nạp cái lệnh đó vào và truyền địa chỉ của từng thành phần trong lệnh đó đi. Vì ở đây là lệnh addi thuộc loại lệnh I nên nó sẽ phân ra các đường:

1. Đường màu tím chính là opcode (31:26), nó sẽ đi vào một khối Control để nó biết được lệnh đó thuộc loại lệnh I và sau đó nó đi đến các khối Registers, ALU control, Data memory để giúp các khối đó làm việc đúng với định dạng lệnh đó.

2. Đường màu xanh lá (25:21) là địa chỉ của toán hạng nguồn thứ nhất và đi vào Register.

3. Đường màu lam (20:16) là địa chỉ của toán hạng đích(vì đây là lệnh lw) đi vào Register.

4. Đường màu cam (15:11) cũng được đi vào register.

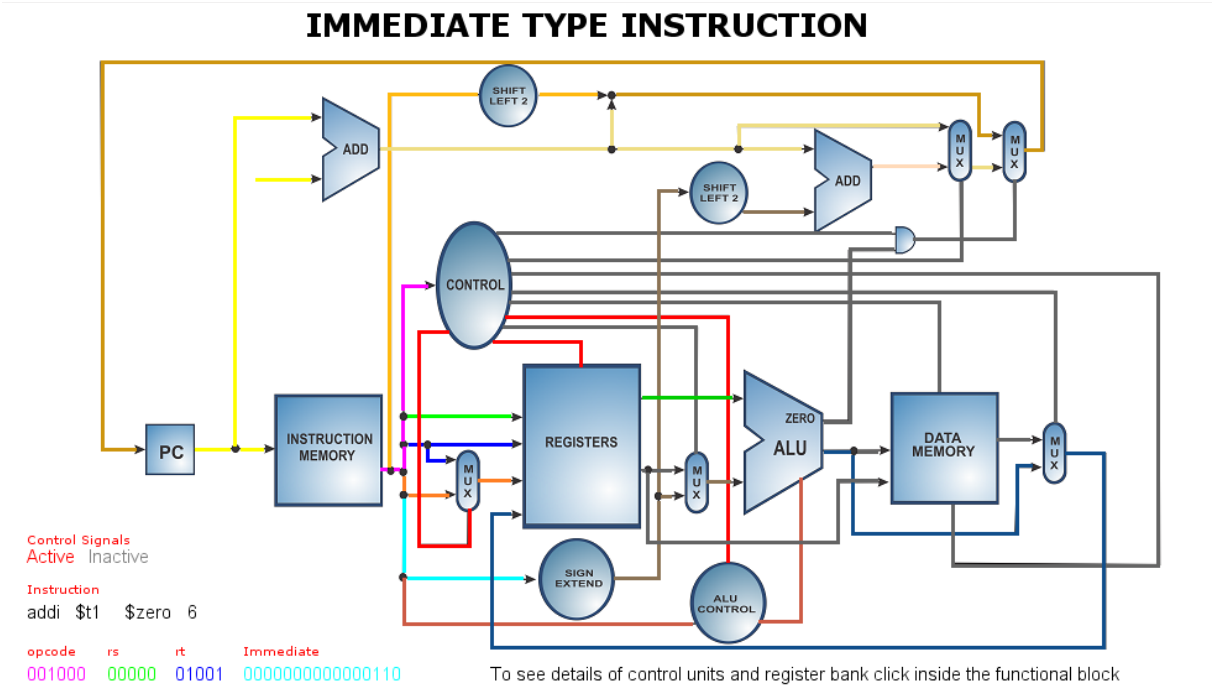
Ta thấy được có 4 đường vào register nhưng do đây là loại I nên chỉ có đường của toán hạng nguồn thứ nhất đi qua và đi tới ALU (đường đi ra đấy là giá trị của thanh ghi).

5. Đường màu cam (15:0) là số immediate đi qua Sign extend để mở rộng từ 16 lên 32 bit và cũng đi tới ALU. Cái đường màu đỏ gạch rẽ ra từng đường này là (5:0) nó đi vào ALU control để điều khiển ALU thực thi chức năng.

- Nhờ các đường dữ liệu trên ta thấy được có 3 đường sẽ đi vào ALU và ở đây ALU sẽ thực thi lệnh tải dữ liệu từ memory từ địa chỉ của a trong memory sau đó kết quả được dưa ra từ data memory sẽ quay ngược lại register và lưu vào toán hạng đích.

- Trong lúc thực hiện tất cả các lệnh trên thì ta thấy PC có rẽ lên 1 nhánh cùng với một nhánh ở Instruction memory sẽ kết hợp lại và dc PC+4 và đưa ngược lại PC.

addi $t1, $0,6 và lệnh addi $t2,$0,5 cũng tương tự :



- Trên đây là datapath của lệnh addi $t1,$0,6

- Giải thích:

- Đầu tiên là từ PC chính là địa chỉ của dòng lệnh đó đi tới Instruction memory

- Trong Instruction memory sẽ nạp cái lệnh đó vào và truyền địa chỉ của từng thành phần trong lệnh đó đi. Vì ở đây là lệnh addi thuộc loại lệnh I nên nó sẽ phân ra các đường:

1. Đường màu tím chính là opcode (31:26), nó sẽ đi vào một khối Control để nó biết được lệnh đó thuộc loại lệnh I và sau đó nó đi đến các khối Register, ALU control, mux để giúp các khối đó làm việc đúng với định dạng lệnh đó.

2. Đường màu xanh lá (25:21) là địa chỉ của toán hạng nguồn thứ nhất và đi vào Register.

3. Đường màu lam (20:16) là địa chỉ của toán hạng đích(vì đây là lệnh addi) đi vào Register.

4. Đường màu cam (15:11) cũng được đi vào register.

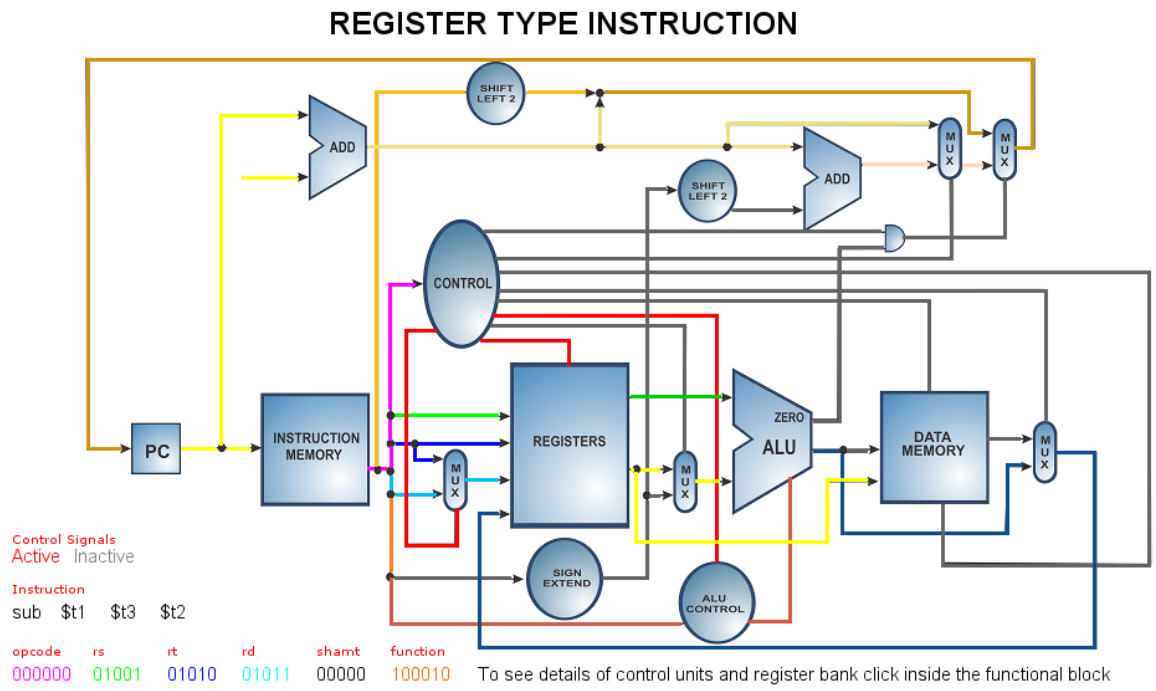
Ta thấy được có 4 đường vào register nhưng do đây là loại I nên chỉ có đường của toán hạng nguồn thứ nhất đi qua và đi tới ALU (đường đi ra đấy là giá trị của thanh ghi).

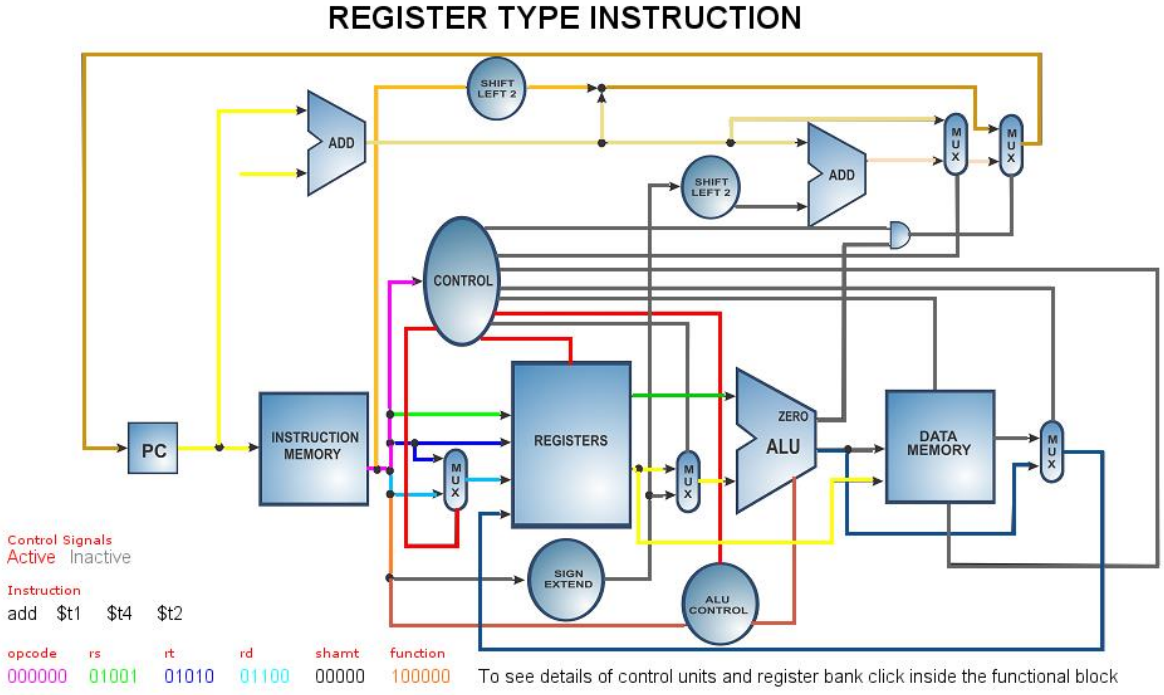
5. Đường màu xanh dương(15:0) là sô tức thời đi qua Sign extend để mở rộng từ 16 lên 32 bit và cũng đi tới ALU. Cái đường màu đỏ gạch rẽ ra từng đường này là (5:0) nó đi vào ALU control để điều khiển ALU thực thi chức năng.

- Nhờ các đường dữ liệu trên ta thấy được có 3 đường sẽ đi vào ALU và ở đây ALU sẽ thực thi lệnh cộng trực tiếp số tức thời với giá trị của thanh ghi của toán hạng nguồn thứ nhất và sau đó kết quả được dưa ra sẽ quay ngược lại register và lưu vào toán hạng đích.

- Trong lúc thực hiện tất cả các lệnh trên thì ta thấy PC có rẽ lên 1 nhánh cùng với một nhánh ở Instruction memory sẽ kết hợp lại và dc PC+4 và đưa ngược lại PC.

sub $t3, $t1, $t2 , add $t4, $t1, $t2 :





- Trên đây là datapath của 2 lệnh sub $t3,$t1,$t2 và add $t4,$t1,$t2

- Giải thích:

• Đầu tiên là từ PC chính là địa chỉ của dòng lệnh đó đi tới Instruction memory

• Trong Instruction memory sẽ nạp cái lệnh đó vào và truyền địa chỉ của từng thành phần trong lệnh đó đi. Vì ở đây là lệnh addi thuộc loại lệnh R nên nó sẽ phân ra các đường:

1. Đường màu tím chính là opcode (31:26), nó sẽ đi vào một khối Control để nó biết được lệnh đó thuộc loại lệnh R và sau đó nó đi đến các khối Register, ALU control, mux để giúp các khối đó làm việc đúng với định dạng lệnh đó.

2. Đường màu xanh lá (25:21) là địa chỉ của toán hạng nguồn thứ nhất và đi vào Register.

3. Đường màu lam (20:16) là địa chỉ của toán hạng nguồn thứ 2 đi vào Register.

4. Đường màu xanh dương (15:11) là toán hạng đích và cũng được đi vào register.

Ta thấy được có 4 đường vào register nhưng do đây là loại R nên chỉ có đường của toán hạng nguồn thứ nhất và thứ hai đi qua và đi tới ALU (đường đi ra đấy là giá trị của thanh ghi).

5. Cái đường màu đỏ gạch rẽ ra từng đường xanh dương là (5:0) nó đi vào ALU control để điều khiển ALU thực thi chức năng.

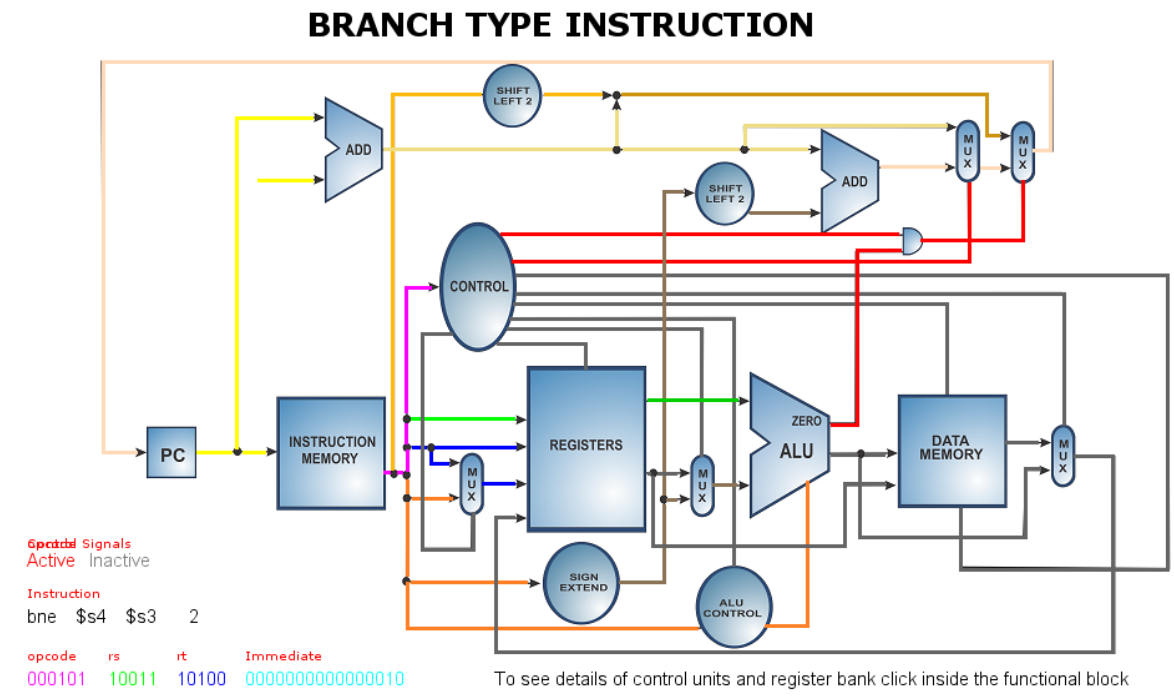
• Nhờ các đường dữ liệu trên ta thấy được có 3 đường sẽ đi vào ALU và ở đây ALU sẽ thực thi lệnh cộng giá trị thanh ghi của toán hạng nguồn thứ nhất và giá trị thanh ghi của toán hạng nguồn thứ hai, sau đó kết quả được dưa ra sẽ quay ngược lại register và lưu vào toán hạng đích.

• Trong lúc thực hiện tất cả các lệnh trên thì ta thấy PC có rẽ lên 1 nhánh cùng với một nhánh ở Instruction memory sẽ kết hợp lại và dc PC+4 và đưa ngược lại PC.

Phần 4.2:

```
.data
# s3: i , s4: j, s0: f, s1: g, s2: h.
.text
        bne $s3,$s4,else
        add $s0,$s1,$s2
        j end
else:
        sub $s0,$s1,$s2
end:
```

bne $s4, $s3, end :



- Trên đây là datapath của bne $s3,$s4,loop

- Giải thích:

• Đầu tiên là từ PC chính là địa chỉ của dòng lệnh đó đi tới Instruction memory

• Trong Instruction memory sẽ nạp cái lệnh đó vào và truyền địa chỉ của từng thành phần trong lệnh đó đi. Vì ở đây là lệnh addi thuộc loại lệnh I nên nó sẽ phân ra các đường:

1. Đường màu tím chính là opcode (31:26), nó sẽ đi vào một khối Control để nó biết được lệnh đó thuộc loại lệnh I và sau đó nó đi đến các khối Register, ALU control, mux để giúp các khối đó làm việc đúng với định dạng lệnh đó.

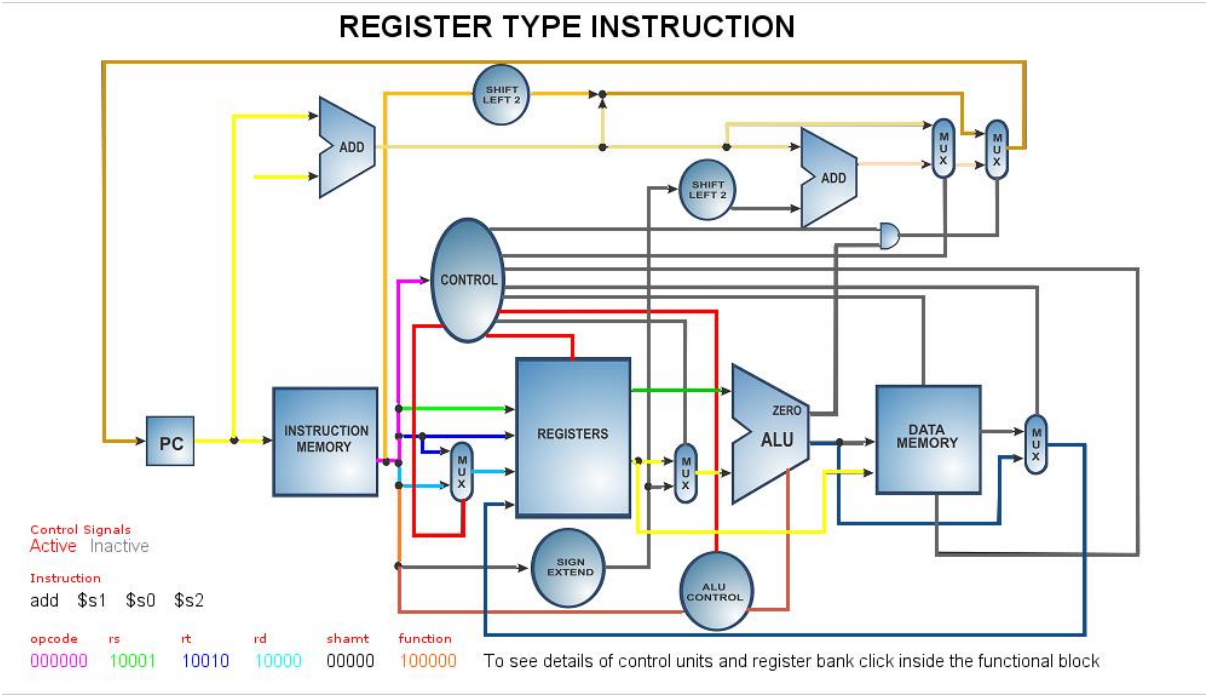
2. Đường màu xanh lá (25:21) là địa chỉ của toán hạng nguồn thứ nhất và đi vào Register.

3. Đường màu lam (20:16) là địa chỉ của toán hạng nguồn thứ 2 đi vào Register.

4. Đường màu cam sẽ đi vào ALU control và đi đến ALU để cho biết ALU phải thực hiện so sánh $s3 và $s4.

• Trong ALU lúc này sẽ nhận được tín hiệu từ ALU control và thực thi so sánh giữa $s3 và $s4 có khác nhau không , nếu bằng nhau thì PC nó vẫn nhảy lên PC+4,nếu khác thì ALU sẽ truyền và tác động lên làm cho PC nhảy tới địa chỉ của label loop và truyền về PC.

add $s1, $s0, $s2 :



- Trên đây là datapath của lệnh add $s0,$s1,$s2

- Đầu tiên là từ PC chính là địa chỉ của dòng lệnh đó đi tới Instruction memory

• Trong Instruction memory sẽ nạp cái lệnh đó vào và truyền địa chỉ của từng thành phần trong lệnh đó đi. Vì ở đây là lệnh addi thuộc loại lệnh R nên nó sẽ phân ra các đường:

1. Đường màu tím chính là opcode (31:26), nó sẽ đi vào một khối Control để nó biết được lệnh đó thuộc loại lệnh R và sau đó nó đi đến các khối Register, ALU control, mux để giúp các khối đó làm việc đúng với định dạng lệnh đó.

2. Đường màu xanh lá (25:21) là địa chỉ của toán hạng nguồn thứ nhất và đi vào Register.

3. Đường màu lam (20:16) là địa chỉ của toán hạng nguồn thứ 2 đi vào Register.

4. Đường màu xanh dương (15:11) là toán hạng đích và cũng được đi vào register.

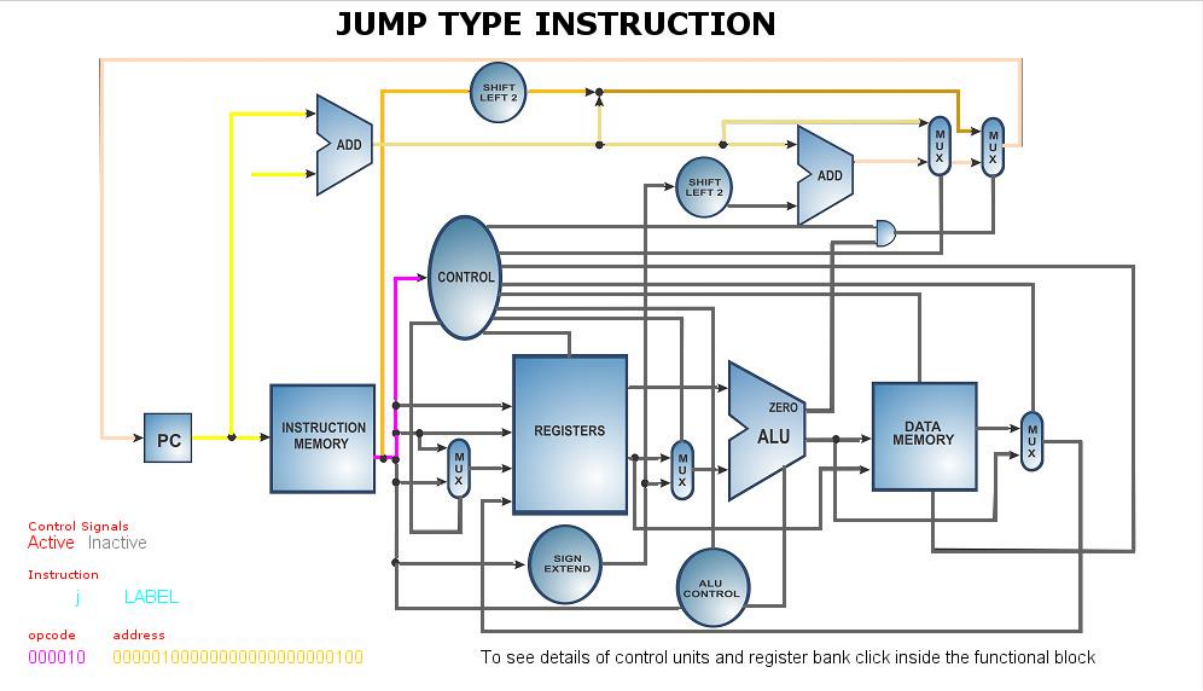
Ta thấy được có 4 đường vào register nhưng do đây là loại R nên chỉ có đường của toán hạng nguồn thứ nhất và thứ hai đi qua và đi tới ALU (đường đi ra đấy là giá trị của thanh ghi).

5. Cái đường màu đỏ gạch rẽ ra từng đường xanh dương là (5:0) nó đi vào ALU control để điều khiển ALU thực thi chức năng.

- Nhờ các đường dữ liệu trên ta thấy được có 3 đường sẽ đi vào ALU và ở đây ALU sẽ thực thi lệnh cộng giá trị thanh ghi của toán hạng nguồn thứ nhất($s1) và giá trị thanh ghi của toán hạng nguồn thứ hai($s2), sau đó kết quả được dưa ra sẽ quay ngược lại register và lưu vào toán hạng đích($s0).

- Trong lúc thực hiện tất cả các lệnh trên thì ta thấy PC có rẽ lên 1 nhánh cùng với một nhánh ở Instruction memory sẽ kết hợp lại và dc PC+4 và đưa ngược lại PC.

j end



- Trên đây là datapath của lệnh j Exit
- Giải thích:

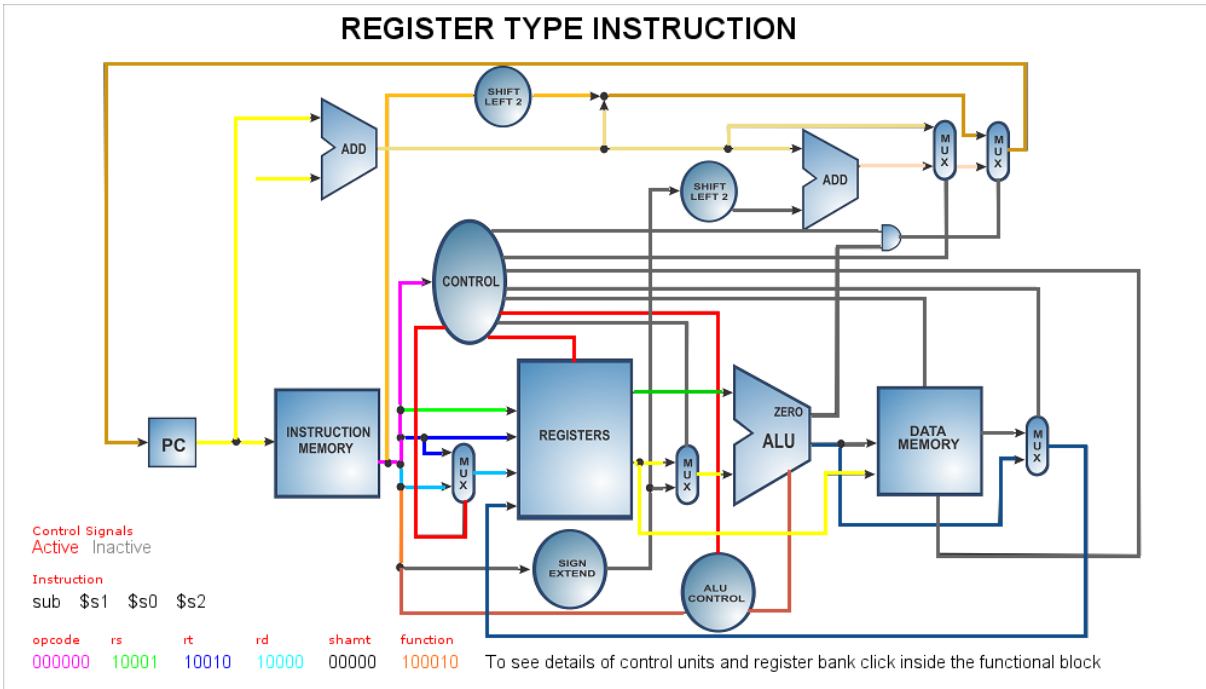
• Đầu tiên là từ PC chính là địa chỉ của dòng lệnh đó đi tới Instruction memory

• Trong Instruction memory sẽ nạp cái lệnh đó vào và truyền địa chỉ của từng thành phần trong lệnh đó đi. Vì ở đây là lệnh addi thuộc loại lệnh J nên nó sẽ phân ra các đường:

1. Đường màu tím opcode (31:26) đi vào khối Control để biết được đây là định dạng loại J và sau đó đưa thông tin tới các bộ điều khiện khác để thực thi đúng chức năng.

2. Màu cam (25:0) là địa chỉ của label Exit nó sẽ đi lên trên xử lý và đưa cái địa chỉ đó về PC để thực hiện lệnh tiếp theo ở label Exit.

sub $s1, $s0, $s2



- Trên đây là datapath của lệnh sub $s0,$s1,$s2

- Đầu tiên là từ PC chính là địa chỉ của dòng lệnh đó đi tới Instruction memory

• Trong Instruction memory sẽ nạp cái lệnh đó vào và truyền địa chỉ của từng thành phần trong lệnh đó đi. Vì ở đây là lệnh addi thuộc loại lệnh R nên nó sẽ phân ra các đường:

6. Đường màu tím chính là opcode(31:26), nó sẽ đi vào một khối Control để nó biết được lệnh đó thuộc loại lệnh R và sau đó nó đi đến các khối Register, ALU control, mux để giúp các khối đó làm việc đúng với định dạng lệnh đó.

7. Đường màu xanh lá (25:21) là địa chỉ của toán hạng nguồn thứ nhất và đi vào Register.

8. Đường màu lam (20:16) là địa chỉ của toán hạng nguồn thứ 2 đi vào Register.

9. Đường màu xanh dương (15:11) là toán hạng đích và cũng được đi vào register.

Ta thấy được có 4 đường vào register nhưng do đây là loại R nên chỉ có đường của toán hạng nguồn thứ nhất và thứ hai đi qua và đi tới ALU (đường đi ra đấy là giá trị của thanh ghi).

10. Cái đường màu đỏ gạch rẽ ra từng đường xanh dương là (5:0) nó đi vào ALU control để điều khiển ALU thực thi chức năng.

- Nhờ các đường dữ liệu trên ta thấy được có 3 đường sẽ đi vào ALU và ở đây ALU sẽ thực thi lệnh trừ giá trị thanh ghi của toán hạng nguồn thứ nhất($s1) cho giá trị thanh ghi của toán hạng nguồn thứ hai($s2), sau đó kết quả được dưa ra sẽ quay ngược lại register và lưu vào toán hạng đích($s0).

- Trong lúc thực hiện tất cả các lệnh trên thì ta thấy PC có rẽ lên 1 nhánh cùng với một nhánh ở Instruction memory sẽ kết hợp lại và dc PC+4 và đưa ngược lại PC.