**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**TỔ CHỨC VÀ CẤU TRÚC MÁY TÍNH II (IT012)**

**A blue logo with a planet and text

Description automatically generated**

**Sinh viên: Trần Nguyễn Thái Bình**

**MSSV: 23520161**

**Giảng viên hướng dẫn: Nguyễn Thành Nhân**

**IT012.O21.1 – Lab 06**

**MỤC LỤC**

[Chức năng các cổng ra của khối Control 3](#_Toc168263994)

[1.1. Giới thiệu về datapath trong MARS 5](#_Toc168263995)

[1.2. Các thành phần của datapath trong MARS: 5](#_Toc168263996)

[2. Phần 2 5](#_Toc168263997)

[3. Phần 3 - Chạy và quan sát quá trình xử lý các lệnh sau thông qua datapath trên MARS 7](#_Toc168263998)

[3.1. add $t1, $t2, $t3 7](#_Toc168263999)

[3.2. addi $t1, $t1, 5 9](#_Toc168264000)

[3.3. sub $t1, $t2, $3 10](#_Toc168264001)

[3.4. lw $t1, 4($t2); # $t2 = 0x10010000 10](#_Toc168264002)

[3.5. sw $t1, 8($t2); # $t2 = 0x10010020 12](#_Toc168264003)

[3.6. J label; Label: exit 13](#_Toc168264004)

[3.7. slt $t1, $t2, $t3 14](#_Toc168264005)

[4. Phần 4 14](#_Toc168264006)

[4.1. Chuyển chương trình sau sang MIPS 14](#_Toc168264007)

[4.2. Chuyển chương trình sau sang MIPS 17](#_Toc168264008)

# **CHỨC NĂNG CÁC CỔNG RA CỦA KHỐI CONTROL**

|  |  |  |
| --- | --- | --- |
| **TÊN** | **CHỨC NĂNG** | **HÌNH ẢNH ĐÍCH ĐẾN** |
| RegDst | Điều khiển nhận **rd** hoặc **rt** đi vào tập thanh ghi **Registers**. |  |
| Branch | Điều chỉnh địa chỉ kế tiếp cho **PC** thay vì chỉ dịch trái 2 bit. |  |
| MemRead | Cho phép đọc từ **Data Memory**. |  |
| MemToReg | Load dữ liệu từ **Data Memory** vào **Registers**. |  |
| ALUOp | Gửi tín hiệu đến **ALU Control** để chọn ra phép tính cho **ALU**. |  |
| MemWrite | Cho phép ghi vào **Data Memory**. |  |
| ALUSrc | Chọn **Read data 2** *(từ* ***Registers****)* hoặc **Sign-extend** đi vào **ALU**. |  |
| RegWrite | Cho phép ghi vào **Registers**. |  |

# **CÁC CÔNG ĐOẠN CỦA 1 LỆNH**

|  |  |  |
| --- | --- | --- |
| **THỨ TỰ** | **CÔNG ĐOẠN** | **HÌNH ẢNH** |
| 1 | Nạp lệnh - Từ **PC** 🡪 **Instruction Memory**. |  |
| 2 | Giải mã - Qua **Registers** dưới sự điều dẫn bởi **Control**. |  |
| 3 | Thực thi - Qua **ALU**. |  |
| 4 | Truy xuất bộ nhớ - Qua **Data Memory**. |  |
| 5 | Lưu kết quả - Trả kết quả về **Registers**. |  |

1. **Hướng dẫn**
   1. **Giới thiệu về datapath trong MARS**

B1: Mở chương trình mô phỏng MARS, sau khi viết code xong chọn tool->MIPS X-Ray để mở cửa sổ mô phỏng datapath.

A computer screen shot of a computer

Description automatically generated

B2: Bấm connect to MIPS => Assemble => chạy từng bước và quan sát quá trình thực thi lệnh trên datapath.

* 1. **Các thành phần của datapath trong MARS:**
* PC: Thanh ghi để lưu địa chỉ của lệnh được thực thi kế tiếp. o ALU: Bộ tính toán số học gồm một số phép toán như: add, or, not…
* ALU Control: Bộ điều khiển các phép toán của ALU o Instruction Memory: Lưu trữ các lệnh sẽ được thực thi.
* Bank of Register: Tập 32 thanh ghi được sử dụng trong kiến trúc MIPS
* Control Unit: Bộ đưa ra các lệnh điều khiển cho ALU, MUX, Register Bank…
* Data Memory: Vùng lưu trữ dữ liệu chương trình ( RAM).
* Sign Extend: Bộ mở rộng bít.
* Shift Left: Bộ dịch trái bít.
* Multiplexers: Bộ chọn.
* Adders: Bộ cộng.

1. **Phần 2**

**Datapath của một số lệnh cơ bản trong MARS:**

A computer screen shot of a computer program

Description automatically generated

*Thực thi lệnh R-type*

A computer screen shot of a diagram

Description automatically generated

*Thục thi lệnh J-type*

A computer screen shot of a computer program

Description automatically generated

*Thực thi lệnh I-type*

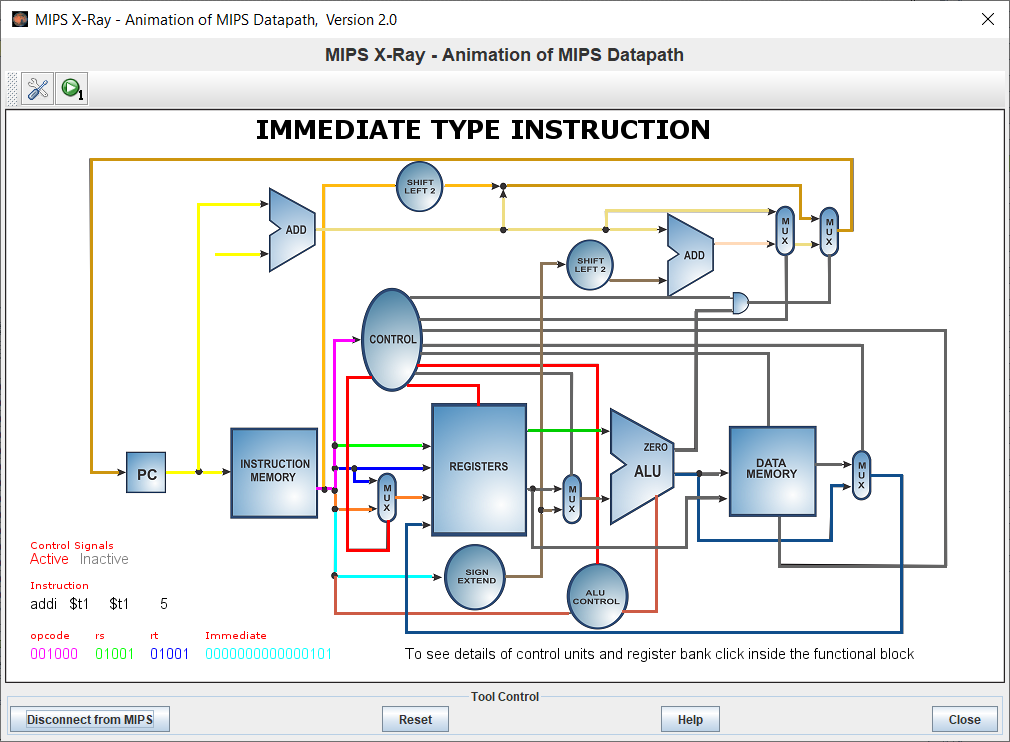
1. **Phần 3 - Chạy và quan sát quá trình xử lý các lệnh sau thông qua datapath trên MARS**
   1. **add $t1, $t2, $t3**

A screenshot of a computer

Description automatically generated

* Giải thích:
  + Thanh ghi **PC** chứa địa chỉ lệnh cần thực hiện và truyền vào:
    - Bộ **Intruction Memory** tách từng bit đi vào:
      * **Control**:
        + Điều khiển **RegDest** nhận **rd**.
        + Điều khiển **ALUSrc** nhận **Read Data 2** từ **Registers** truyền vào **ALU**.
        + Điều khiển **ALU Control**.
        + Kích hoạt **Register Write**.
        + Điều khiển **MemToReg** chọn kết quả từ **ALU** trả về **Registers**.
      * Và các đường dẫn vào khác *(như hình)*.
    - Bộ **add A diagram of a diagram

      Description automatically generated** để dịch bit địa chỉ của PC sang 2 bits *(tức lệnh kế tiếp theo)* và quay trở lại **PC**.
  + **Các công đoạn**:
    - 1 – Mã máy: 0000 0001 0100 1011 0100 1000 0010 0000
    - 2 *–* lấy giá trị từ **$t1**, **$t2**.
    - 3 – giá trị của **$t2** cộng với giá trị của **$t3**.
    - 5 – ghi kết quả vào **$t1**.
  1. **addi $t1, $t1, 5**



* Giải thích:
  + Thanh ghi **PC** chứa địa chỉ lệnh cần thực hiện và truyền vào:
    - Bộ **Intruction Memory** tách từng bit đi vào:
      * **Control**:
        + Điều khiển **RegDest** nhận **rt**.
        + Điều khiển **ALUSrc** nhận **Sign-extend** truyền vào **ALU**.
        + Điều khiển **ALU Control**.
        + Kích hoạt **Register Write**.
        + Điều khiển **MemToReg** chọn kết quả từ **ALU** trả về **Registers**.
      * Và các đường dẫn vào khác *(như hình)*.
    - Bộ **add A diagram of a diagram

      Description automatically generated** để dịch bit địa chỉ của PC sang 2 bits *(tức lệnh kế tiếp theo)* và quay trở lại **PC**.
  + **Các công đoạn**:
    - 1 – Mã máy: 0010 0001 0010 1001 0000 0000 0000 0101
    - 2 – lấy giá trị từ **$t1**.
    - 3 – giá trị của **$t1** cộng với giá trị đầu ra của **Sign-extend**).
    - 5 – ghi kết quả vào **$t1**.
  1. **sub $t1, $t2, $3**

A screenshot of a computer

Description automatically generated

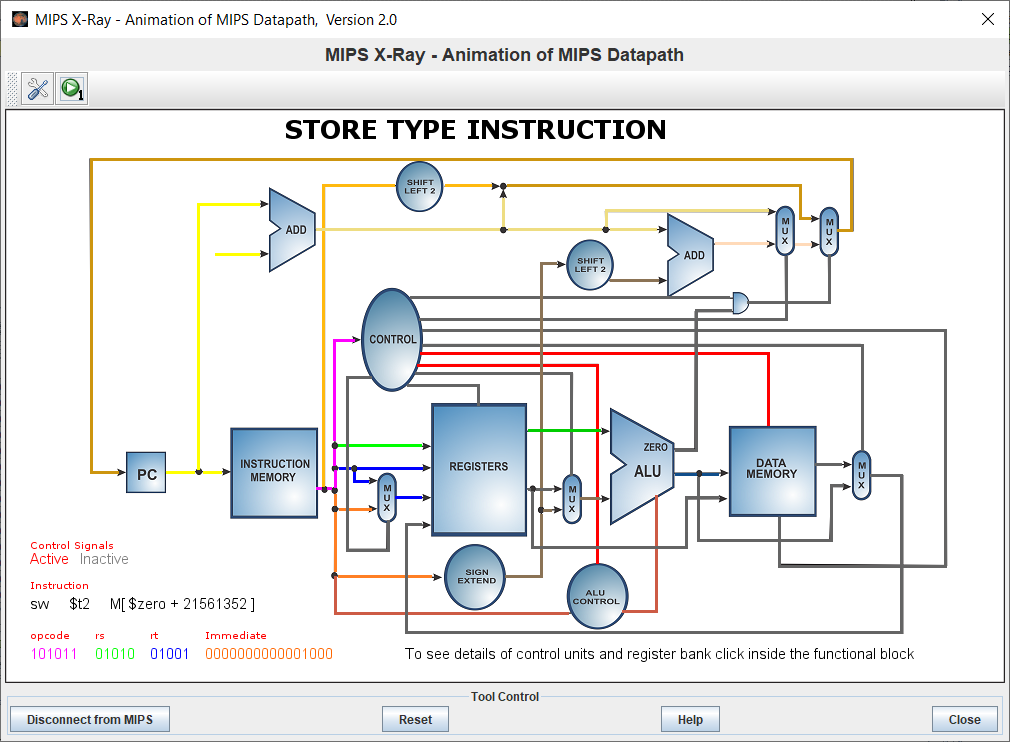
* Mã máy: 0000 0001 0100 0011 0100 1000 0010 0010
* Thanh ghi **$3** là **$v1**.
* Giải thích: tương tự lệnh **add $t1, $t2, $t3** ở mục **3.1.** Khác biệt ở tín hiệu từ **Control** đến **ALU Control** và từ **ALU Control** đến **ALU** *(phép tính trừ)***.**
  1. **lw $t1, 4($t2); # $t2 = 0x10010000**

A screenshot of a computer

Description automatically generated

* Giải thích:
  + Thanh ghi **PC** chứa địa chỉ lệnh cần thực hiện và truyền vào:
    - Bộ **Intruction Memory** tách từng bit đi vào:
      * **Control**:
        + Điều khiển **RegDest** nhận **rt**.
        + Điều khiển **ALUSrc** nhận **Sign-extend** truyền vào **ALU**.
        + Điều khiển **ALU Control**.
        + Kích hoạt **MemRead**.
        + Kích hoạt **Register Write**.
        + Điều khiển **MemToReg** chọn kết quả từ **ALU** trả về **Registers**.
      * Và các đường dẫn vào khác *(như hình)*.
    - Bộ **add A diagram of a diagram

      Description automatically generated** để dịch bit địa chỉ của PC sang 2 bits *(tức lệnh kế tiếp theo)* và quay trở lại **PC**.
  + **Các công đoạn**:
    - 1 – mã máy: 1000 1101 0100 1001 0000 0000 0000 0100
    - 2 – lấy giá trị của **$t2**.
    - 3– giá trị của **$t2** cộng với giá trị đầu ra của **Sign-extend** *(4)*: 0x10010004.
    - 4 – lấy giá trị tại địa chỉ của kết quả công đoạn 3 từ **Data Memory**.
    - 5 *–* ghi kết quảvào **$t1**.
  1. **sw $t1, 8($t2); # $t2 = 0x10010020**



* Giải thích:
  + Thanh ghi **PC** chứa địa chỉ lệnh cần thực hiện và truyền vào:
    - Bộ **Intruction Memory** tách từng bit đi vào:
      * **Control**:
        + Điều khiển **RegDest** nhận **rt**.
        + Điều khiển **ALUSrc** nhận **Sign-extend** truyền vào **ALU**.
        + Điều khiển **ALU Control**.
        + Kích hoạt **MemWrite**.
        + Kích hoạt **Register Write**.
      * Và các đường dẫn vào khác *(như hình)*.
    - Bộ **add A diagram of a diagram

      Description automatically generated** để dịch bit địa chỉ của PC sang 2 bits *(tức lệnh kế tiếp theo)* và quay trở lại **PC**.
  + **Các công đoạn**:
    - 1 – mã máy: 1010 1101 0100 1001 0000 0000 0000 1000
    - 2 – lấy giá trị của **$t1**, **$t2**.
    - 3– giá trị của **$t2** cộng với giá trị đầu ra của **Sign-extend** *(4)*: 0x10010028.
    - 4 – lưu vào **$t1** ở **Data Memory** *(địa chỉ kết quả của công đoạn 3)*.
  1. **J label; Label: exit**

A screenshot of a computer program

Description automatically generated

* Giải thích:
  + Thanh ghi **PC** chứa địa chỉ lệnh cần thực hiện và truyền vào:
    - Bộ **Intruction Memory** tách từng bit đi vào:
      * Bộ add **A diagram of a diagram

        Description automatically generated** để dịch bit địa chỉ của PC sang 2 bits.
      * **Control**:
        + Kích hoạt **Branch**.
  + **Các công đoạn**:
    - 1 – mã máy: 0000 1000 0001 0000 0000 0000 0000 0001
    - Giá trị của **Sign-extend** dùng để tính toán và “and” với kết quả của bước dịch địa chỉ của PC sang 2 bits. Sau đó sẽ trả kết quả về PC.
  1. **slt $t1, $t2, $t3**

A screenshot of a computer

Description automatically generated

* Mã máy: 0000 0001 0100 1011 0100 1000 0010 1010
* Giải thích: tương tự lệnh **add $t1, $t2, $t3** ở mục **3.1.** Khác biệt ở tín hiệu từ **Control** đến **ALU Control** và từ **ALU Control** đến **ALU,** kết quả trả về là **0** hoặc **1.**

1. **Phần 4**
   1. **Chuyển chương trình sau sang MIPS**

1. int a, b, c, d;

2. a = 6;

3. b = 5;

4. c = a - b;

5. d = a + b;

* Các biến được lưu trong memory.
* Xác định các lệnh tương ứng là loại lệnh nào (R-type, I-Type, J-Type)? Giải thích?
* Kết nối chương trình với MIPS X-Ray trong MARS. Chạy từng bước các lệnh và ở mỗi lệnh giải thích quá trình thực thi lệnh đó trên datapath trong MARS.

**Bài làm:**

1. .data

2. a: .word 6

3. b: .word 5

4. c: .word 0

5. d: .word 0

6.

7. .text

8. main:

9. # Load to registers

10. lw $s0, a

11. lw $s1, b

12. lw $s2, c

13. lw $s3, d

14.

15. # Calculate

16. sub $s2, $s0, $s1

17. add $s3, $s0, $s1

18.

19. # Store to memory

20. sw $s2, c

21. sw $s3, d

* Giả thích code

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Loại lệnh** |
| 1 🡪 5 | Gán giá trị cho a, b, c, d |  |
| 10 | Load giá trị a lưu vào register **$s0**. | **I** |
| 11 | Load giá trị b lưu vào register **$s1**. | **I** |
| 12 | Load giá trị c lưu vào register **$s2**. | **I** |
| 13 | Load giá trị d lưu vào register **$s3**. | **I** |
| 16 | Lấy **$s0 - $s1** rồi lưu vào register **$s2**. | **R** |
| 17 | Lấy **$s0 + $s1** rồi lưu vào register **$s3**. | **R** |
| 20 | Lưu giá trị của **$s2** vào c. | **I** |
| 21 | Lưu giá trị của **$s3** vào d. | **I** |

* Kết quả được xem từ register

A screenshot of a number

Description automatically generated

* Kết nối chương trình MIPS

|  |  |  |
| --- | --- | --- |
| **Dòng** | **HÌNH ẢNH** | **DATAPATH** *(đã giải thích ở mục 3)* |
| 10 |  | **3.4** |
| 11 |  | **3.4** |
| 12 |  | **3.4** |
| 13 |  | **3.4** |
| 16 |  | **3.3** |
| 17 |  | **3.1** |
| 20 |  | **3.5** |
| 21 |  | **3.5** |

* 1. **Chuyển chương trình sau sang MIPS**

1. if (i == j)

2. f = g + h;

3. else f = g – h;

* i được lưu trong **$s3**, j trong **$s4**, f trong **$s0**, g trong **$s1**, h trong **$s2**.
* Phải sử dụng lệnh bne và j trong chương trình.
* Kết nối chương trình với MIPS X-Ray trong MARS. Chạy từng bước các lệnh và ở mỗi lệnh giải thích quá trình thực thi lệnh đó trên datapath trong MARS.

**Bài làm:**

1. .data

2. i: .word 2 3

3. j: .word 3 3

4. f: .word 0 0

5. g: .word 5 4

6. h: .word 4 7

7.

8. .text

9. main:

10. # Load to registers

11. lw $s3, i

12. lw $s4, j

13. lw $s0, f

14. lw $s1, g

15. lw $s2, h

16.

17. # Check if i == j

18. bne $s3, $s4, ELSE

19. add $s0, $s1, $s2

20. j END

21.

22. ELSE:

23. sub $s0, $s1, $s2

24.

25. END:

* Giả thích code

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Loại lệnh** |
| 1 🡪 6 | Gán giá trị cho i, j, f, g, h. |  |
| 11 | Load giá trị i lưu vào register **$s3**. | **I** |
| 12 | Load giá trị j lưu vào register **$s4**. | **I** |
| 13 | Load giá trị f lưu vào register **$s5**. | **I** |
| 14 | Load giá trị g lưu vào register **$s6**. | **I** |
| 15 | Load giá trị h lưu vào register **$s7**. | **I** |
| 18 | So sánh **$s3** và **$s4,** nếu bằng thì thực hiện tiếp dòng 19, còn không bằng thì nhảy đến **label** ELSE. | **I** |
| 19 | Lấy **$s1 + $s2** rồi gán vào **$s0.** | **R** |
| 20 | Nhảy đến **label** end. | **J** |
| 23 | Lấy **$s1 - $s2** rồi gán vào **$s0.** | **R** |

* Kết quả được xem từ register

Trái

A screenshot of a computer

Description automatically generated

Phải kev

A screenshot of a computer

Description automatically generated

* Kết nối chương trình MIPS

|  |  |  |
| --- | --- | --- |
| **Dòng** | **HÌNH ẢNH** | **DATAPATH** *(đã giải thích ở mục 3)* |
| 11 | Trái:    Phải: | **3.4** |
| 12 | Trái:  Phải: | **3.4** |
| 13 | Trái:    Phải: | **3.4** |
| 14 | Trái:    Phải: | **3.4** |
| 15 | Trái:    Phải: | **3.4** |
| 18 | Trái:    Phải: | **Giải thích mở rộng** |
| 19 | Trái:  Phải: | **3.1** |
| 20 |  | **3.6** |
| 23 |  | **3.3** |

* Giải thích lệnh bne
  + - Giả sử có lệnh bne **$s3**, **$s4**, **BRANCH**

A computer screen shot of a computer program

Description automatically generated

* Giải thích: có một số điểm giống các lệnh I đã được giải thích. Bên cạnh đó có một số sự khác biệt:
  + **Các công đoạn**:
    - 1 – Mã máy: 0001 0110 0111 0100 0000 0000 0000 0010
    - 2 – lấy giá trị từ **$s3, $s4**.
    - 3 – Bên trong **ALU**, sẽ so sánh giá trị hiệu của **$s3** và **$s4** với 0.
    - Nếu biểu thức ở công đoạn 3 là sai *(không bằng)* thì **Zero** sẽ bằng 1, và ngược lại. Từ đó sẽ được tính toán với bộ add để điều chỉnh địa chỉ nhảy dựa trên đầu ra của **Sign-extend** và trả về PC.