HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

**SCHOOL OF INFORMATION AND COMMUNICATION AND TECHNOLOGY**

**---□**&**□---**



**Final Report**

**COMPUTER ARCHITECTURE LAB**

**Lecturer: Lê Bá Vui**

**Student: Kiều Thái Thịnh – Nguyễn Văn Dương**

**Group: 8**

**Class:143684**

***School year 2023-2024 ……***

Contents

## Group members

|  |  |  |
| --- | --- | --- |
| **Name** | **ID** | **Topic** |
| Kiều Thái Thịnh | 20215246 | 02. Vẽ hình trên màn hình Bitmap |
| Nguyễn Văn Dương | 20215189 | 07. Chương trình kiểm tra cú pháp lệnh MIPS |

## Topic 02: Vẽ hình trên màn hình Bitmap

**Giải thích hàm:**

* Hàm main: khởi tạo vị trí đầu tiên cho tâm của hình tròn và gọi hàm để thực hiện vẽ hình tròn
* Hàm main1: nạp vào địa chỉ của Keybroad and Display MMIO Simulator

Chứa các hàm thực hiện nhập vào trong Keybroad để điều khiển và check xem lệnh điều khiển

* Hàm draw\_circle: Dùng để vẽ hình tròn với việc nhập vào địa chỉ của Bitmap Display vào $s4 dựa trên các thông số của hình tròn lưu lần lượt tại $t8, $t9, $t0 là vị trí trục Ox, vị trí trục Oy, màu hình tròn(Do $t4 chứa thời gian sleep sau mỗi lần di chuyển của hình tròn)
* Hàm speed\_up: Dùng để tăng tốc, truy cập đến sau khi nhập vào ‘z’ trên Keybroad

Tăng tốc bằng cách giảm giá trị $t4 xuống còn 1 nửa

* Hàm speed\_down: Dùng để giảm tốc, truy cập đến sau khi nhập vào ‘x’ trên Keybroad

Tăng tốc bằng cách tăng giá trị $t4 lên gấp đôi

* Hàm turn\_right: dùng để xử lý khi nhập vào để đi sang phải

Tăng giá trị $t8 thêm 2 đơn vị mỗi lần chạy

* Hàm turn\_left: dùng để xử lý khi nhập vào để chạy sang trái

Giảm giá trị $t8 mất 2 đơn vị mỗi lần chạy

* Hàm turn\_up: dùng để xử lý khi nhập vào để chạy lên trên

Giảm giá trị $t9 mất 2 đơn vị mỗi lần chạy

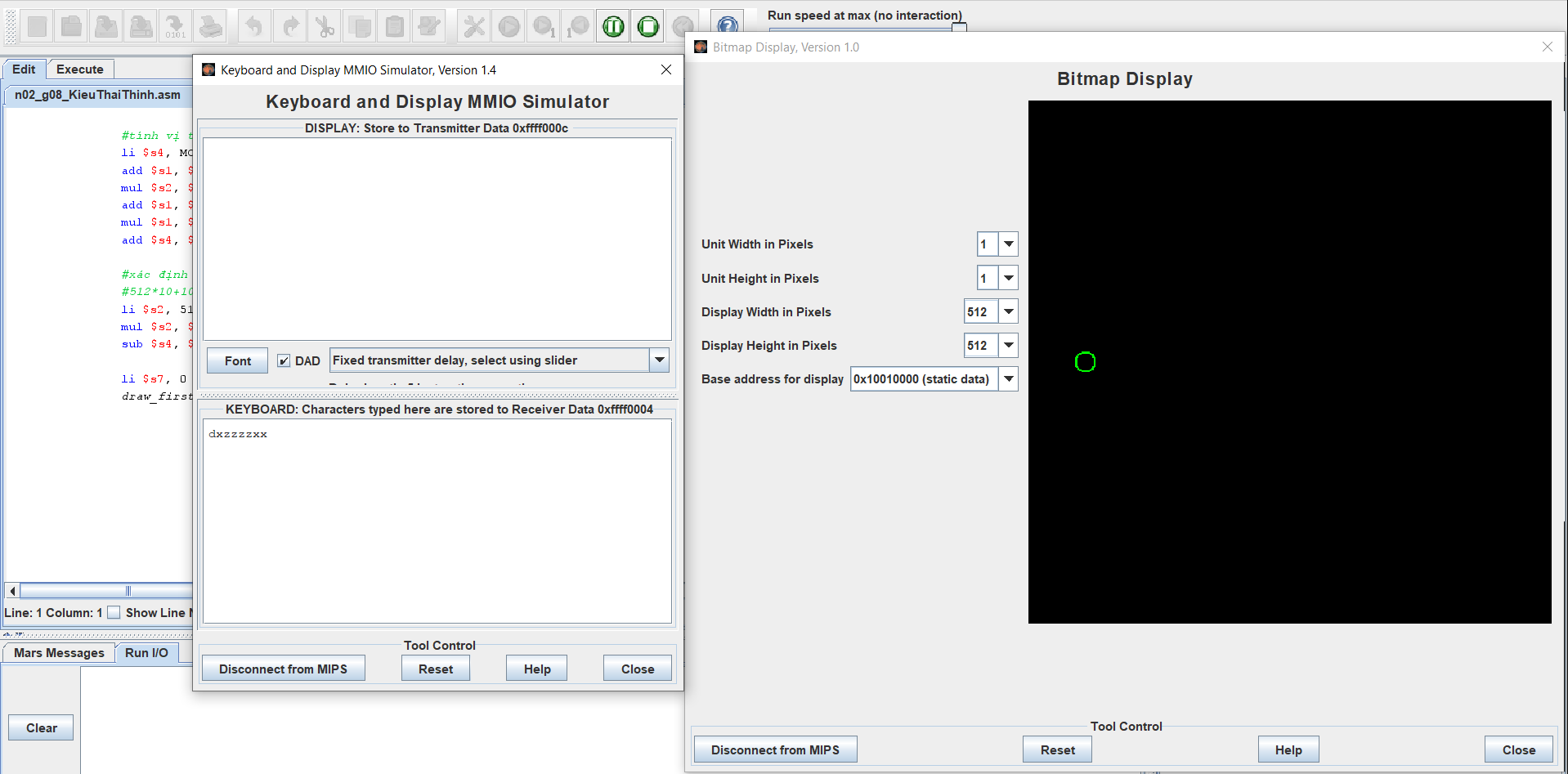
* Hàm turn\_down: dùng để xử lý khi nhập vào để chạy đi xuống

Tăng giá trị $t9 thêm 2 đơn vị mỗi lần chạy

* Hàm speed: Thực hiện ngủ bằng $t4 ms

Xác định được tốc độ chạy

**Results:**

****

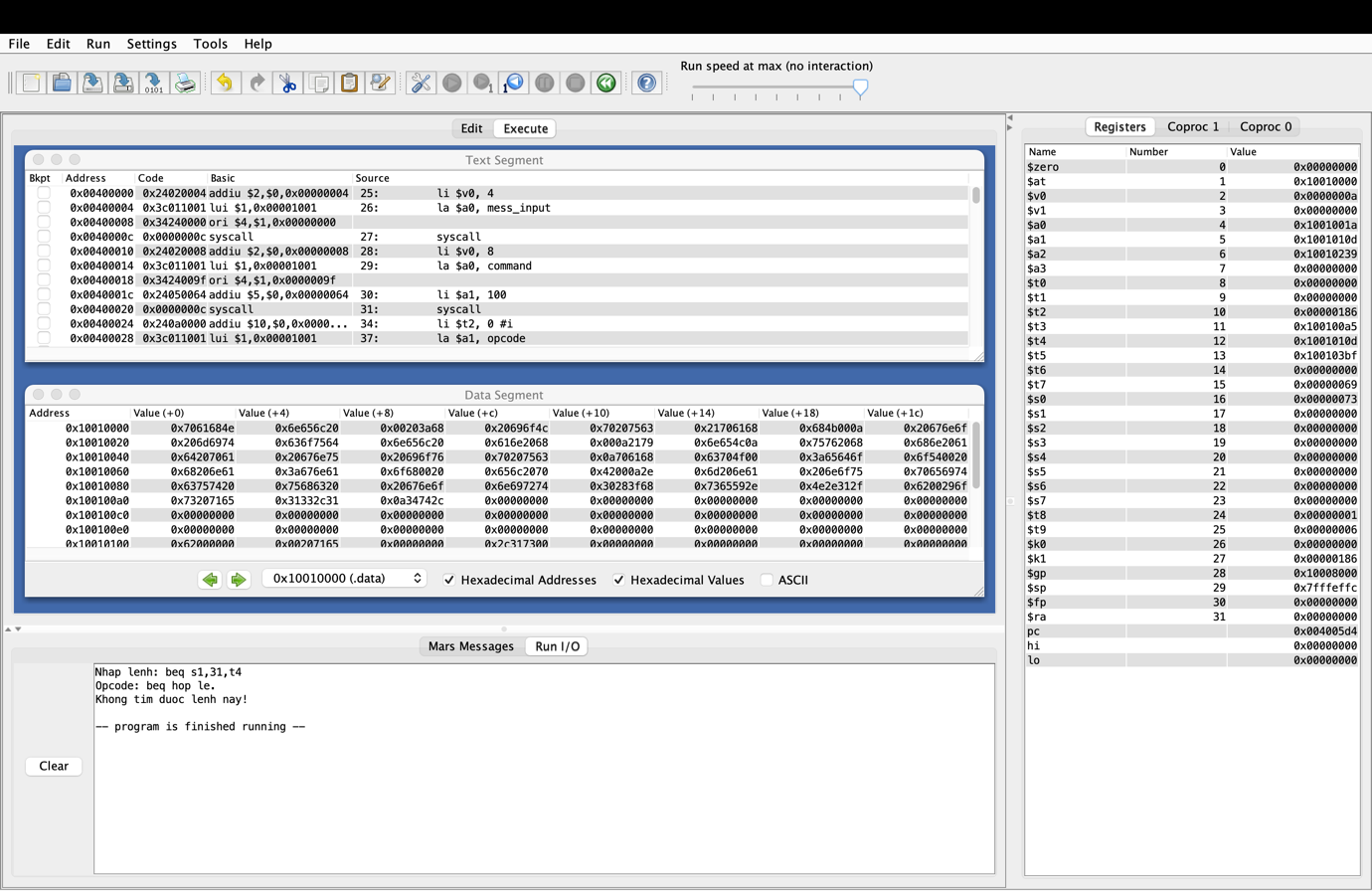
## Topic 07: Chương trình kiểm tra cú pháp lệnh MIPS

**Giải thích hàm:   
 1. enter\_input:**

* + Phần này yêu cầu người dùng nhập vào một lệnh MIPS và lưu vào bộ đệm **command**.

1. **main:**
   * Bắt đầu thực thi chính tại đây. Phần này khởi tạo các bộ đếm và gọi hàm Read\_Opcode để xử lý opcode.
2. **Read\_Opcode:**
   * Đọc phần opcode của lệnh từ bộ đệm **command** và lưu vào bộ đệm **opcode**. Nó tìm kiếm ký tự khoảng trắng để xác định điểm kết thúc của opcode.
3. **Processing\_Opcode:**
   * So sánh opcode đã đọc với các opcode đã được định nghĩa trước trong **CommandData**. Sử dụng vòng lặp để so sánh từng ký tự của opcode để xác nhận.
4. **Read\_operand\_1, Read\_Operand\_2, Read\_Operand\_3:**
   * Các hàm này đọc và xử lý các toán hạng dựa trên loại được xác định trong **CommandData**. Các loại bao gồm thanh ghi, số nguyên, nhãn hoặc toán hạng rỗng.
5. **Check\_Token\_Register:**
   * Xác nhận nếu toán hạng là một thanh ghi bằng cách so sánh với danh sách tên các thanh ghi hợp lệ được lưu trong **TokenData**.
6. **Check\_Integer:**
   * Xác nhận nếu toán hạng là một số nguyên. Nó kiểm tra mỗi ký tự để đảm bảo rằng nó là một chữ số hoặc dấu trừ cho số âm.
7. **Check\_Label:**
   * Xác nhận nhãn toán hạng bằng cách kiểm tra mỗi ký tự có hợp lệ dựa trên **CharData**, bao gồm chữ cái, chữ số và một số ký tự đặc biệt.
8. **Check\_Null\_Token:**
   * Kiểm tra các toán hạng rỗng hoặc không có giá trị.
9. **continue và resetAll:**
   * Điều khiển vòng lặp của chương trình, hỏi người dùng liệu họ muốn tiếp tục hoặc khởi động lại chương trình với dữ liệu nhập mới.
10. **notFound, error, end, TheEnd:**
    * Xử lý các tình huống kết thúc khác nhau như xác nhận hợp lệ, lỗi, hoặc kết thúc chương trình.

**Results:**

****