Hướng dẫn thực hiện

Chia nhóm 2 người chọn 2 Projects, mỗi người thực hiện 1 bài + hiểu bài còn lại.

Cách thức đánh giá:

- Trình bày mã nguồn, chạy mô phỏng MIPSIT, trả lời câu hỏi của giáo viên
- Viết báo cáo in quyển (theo nhóm) trình bày phân tích cách làm, mã nguồn và kết quả chạy mô phỏng. Yêu cầu báo cáo chi tiết, rõ ràng
- Nộp chương trình hợp ngữ, bản mềm. Qui định cách đặt tên file như sau:

nxx_gyy_StudentName.asm

+ xx: số thứ tự của bài. Ví dụ SV làm bài sô 8 thì xx là 08, SV làm bài số 10 thì xx là 10.
+ yy: số thứ tự của nhóm. Ví dụ SV thuộc nhóm 7 thì yy là 07, SV thuộc nhóm 18 thì yy là 18.
+ StudentName: là tên của sinh viên, tiếng Việt, không dấu, không có kí tự space và viết hoa chữ cái đầu. Ví dụ sinh viên Nguyễn Văn A thì StudentName là NguyenVanA, sinh viên Vũ Thanh Thủy thì StudentName là VuThanhThuy.
+ Đuôi file là .asm nếu lập trình bằng MARS, hoặc .s nếu lập trình bằng MIPSIT.
+ Ví dụ, sinh viên tên Lê Mạnh Hùng, thuộc nhóm 21, làm bài 8, lập trình bằng MARS, thì cần gửi cho giáo viên bản mềm với tên file là n08_g21_LeManhHung.asm

Lưu ý khi làm bài:

- Chương trình nên có giao diện menu chọn (nếu cần), có thể lặp lại nhiều lần.
- Các thao tác nhập liệu nên có kiểm tra, xác thực tính hợp lệ của dữ liệu
- Mã nguồn nên tổ chức thành chương trình con (nếu cần), (xem thêm lưu ý trong phần requirements)
- Viết mã nguồn sáng sủa, dễ đọc, dễ hiểu (có căn lề, chú thích)
- Báo cáo nên trình bày về: phân tích, cách làm (thuật toán), mã nguồn, hình ảnh kết quả mô phỏng.

1. Curiosity Marsbot

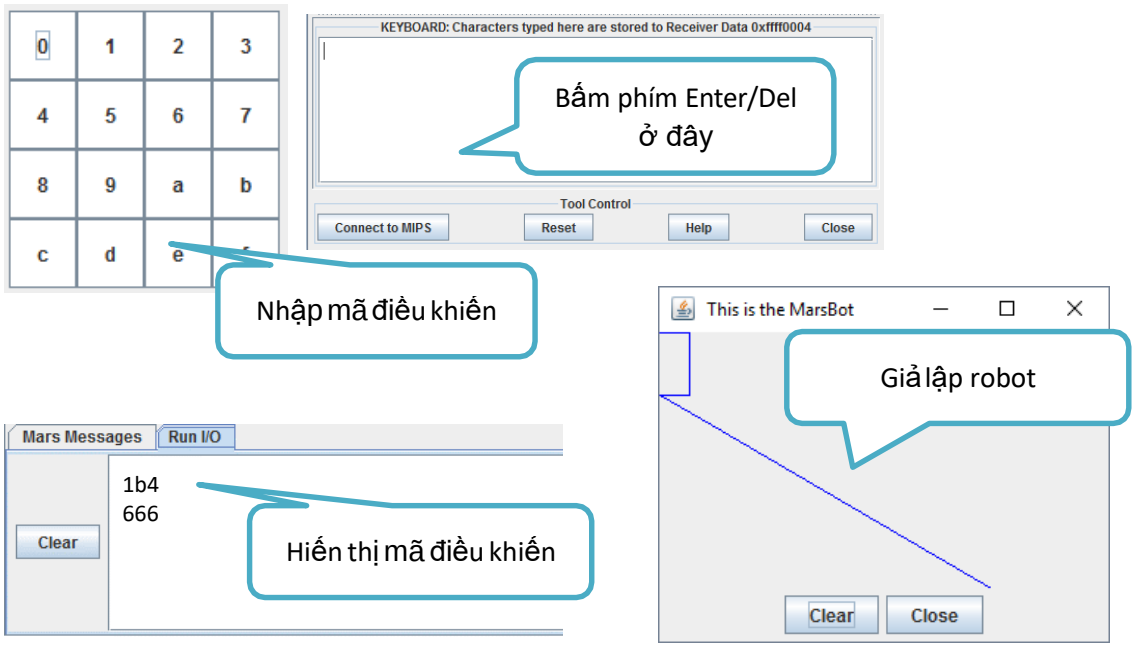
Xe tự hành Curioisity Marsbot chạy trên sao Hỏa, được vận hành từ xa bởi các lập trình viên trên Trái Đất. Bằng cách gửi đi các mã điều khiển từ một bàn phím ma trận, lập trình viên điều khiển quá trình di chuyển của Marbot như sau:

Mã điều khiển	Ý nghĩa
1b4	Marbot bắt đầu chuyển động
c68	Marbot đứng im
444	Rẽ trái 90* so vơi phương chuyển động gần đây và giữ hướng mới
666	Rẽ phải 90* so vơi phương chuyển động gần đây và giữ hướng mới
dad	Bắt đầu để lại vết trên đường
cbc	Chấm dứt để lại vết trên đường
999	Tự động quay trở lại theo lộ trình ngược lại. Không vẽ vết, không nhận mã khác cho tới khi kết thúc lộ trìnhngược. Mô tả: Marsbot được lập trình để nhớ lại toàn bộ lịch sử các mã điều khiển và khoảng thời gian giữa các lần đổi mã. Vì vậy, nó có thể đảo ngược lại lộ trình để quay về điểm xuất phát (dù có thể lệch một chút do hàm syscall sleep không thực sự thời gian thực)

Sau khi nhận mã điều khiển, Curioisity Marsbot sẽ không xử lý ngay, mà phải đợi lệnh kích hoạt mã từ bàn phím Keyboard & Display MMIO Simulator. Có 2 lệnh nhưvậy:

Kích hoạt mã	Ý nghĩa
Phím Enter	Kết thúc nhập mã và yêu cầu Marbot thực thi
Phím Del	Xóa toàn bộ mã điều khiển đang nhập dở dang.

Hãy lập trình để Marsbot có thể hoạt động như đã mô tả.
Đồng thời bổ sung thêm tính năng: mỗi khi gửi một mã điều khiển cho Marsbot, hiển thị mã đó lên màn hình console để người xem có thể giám sát lộ trình của xe.



2. Vẽ hình trên màn hình Bitmap

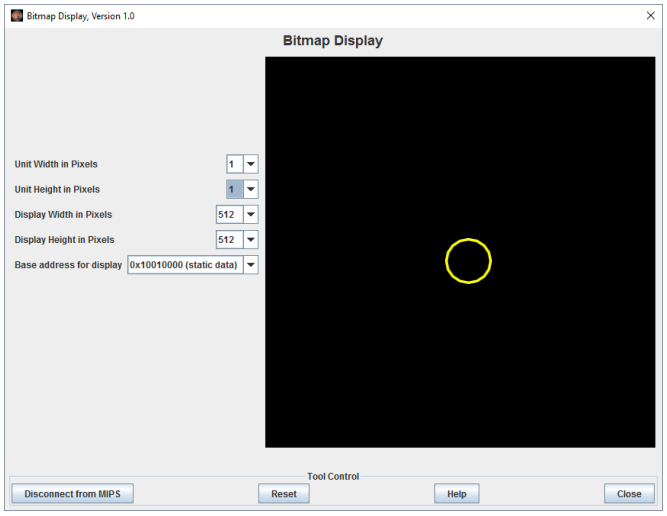
Viết một chương trình sử dụng MIPS để vẽ một quả bóng di chuyển trên màn hình mô phỏng Bitmap của Mars). Nếu đối tượng đập vào cạnh của màn hình thì sẽ di chuyển theo chiều ngược lại.

Yêu cầu

- Thiết lập màn hình ở kích thước 512x512. Kích thước 1 pixel 1x1.
- Quả bóng là một đường tròn

Chiều di chuyển phụ thuộc vào phím người dùng bấm, gồm có (di chuyển lên (W), di chuyển xuống (S), Sang trái (A), Sang phải (D) trong bộ giả lập Keyboard and Display MMIO Simulator). Tốc độ bóng di chuyển là không đổi. Vị trí bóng ban đầu ở giữa màn hình.

Gợi ý: Để làm một đối tượng di chuyển thì chúng ta sẽ xóa đối tượng ở vị trí cũ và vẽ đối tượng ở vị trí mới. Để xóa đối tượng chúng ta chỉ cần vẽ đối tượng đó với màu là màu nền



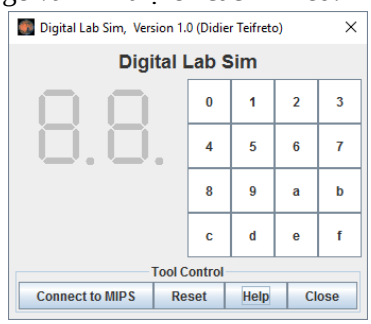
`

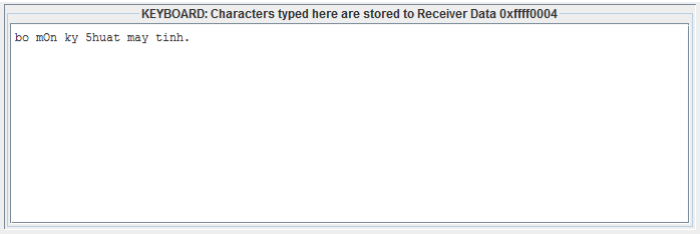
3. Kiểm tra tốc độ và độ chính xác khi gõ văn bản

Chương trình sau sẽ đo tốc độ gõ bàn phím và hiển thị kết quả bằng 2 đèn led 7 đoạn. Nguyên tắc:

- Cho một đoạn văn bản mẫu, cố định sẵn trong mã nguồn. Ví dụ *"bo mon ky thuat may tinh"*
- Sử dụng bộ định thời Timer (trong bộ giả lập Digi Lab Sim) để tạo ra khoảng thời gian để đo. Đây là thời gian giữa 2 lần ngắt, chu kì ngắt.
- Trong thời khoảng đó, người dùng nhập các kí tự từ bàn phím. Ví dụ nhập "*bo m**O**n ky **5**huat may tinh*".

Chương trình cần phải đếm số kí tự đúng (trong ví dụ trên thì người dùng gõ sai chữ **O** và **5**) mà người dùng đã gõ và hiển thị lên các đèn led.





4. Postscript CNC Marsbot

Máy gia công cơ khí chính xác CNC Marsbot được dùng để cắt tấm kim loại theo các đường nét được qui định trước. CNC Marsbot có một lưỡi cắt dịch chuyển trên tấm kim loại, với giả định rằng:

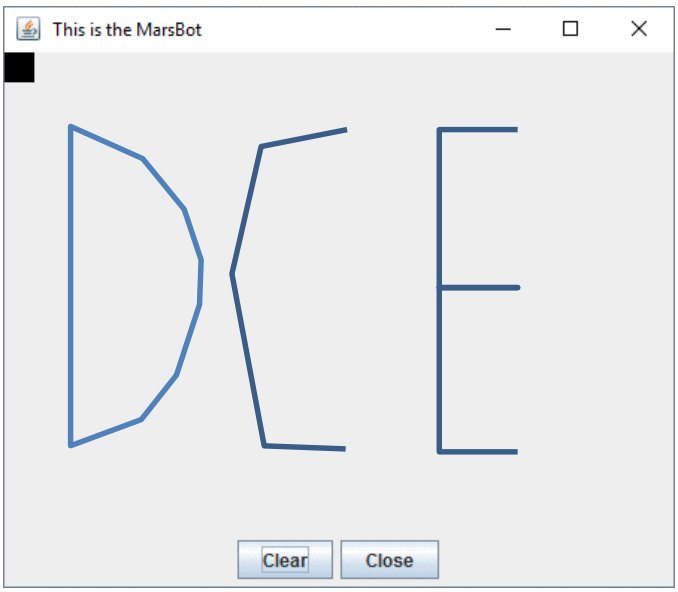
- Nếu lưỡi cắt dịch chuyển nhưng không cắt tấm kim loại, tức là Marsbot di chuyển nhưng không để lại vết (Track)
- Nếu lưỡi cắt dịch chuyển và cắt tấm kim loại, tức là Marsbot di chuyển và có để lại vết.

Để điều khiển Marsbot cắt đúng như hình dạng mong muốn, người ta nạp vào Marsbot một mảng cấu trúc gồm 3 phần tử:

- <Góc chuyển động>, <Thời gian>, <Cắt/Không cắt>
- Trong đó <Góc chuyển động> là góc của hàm HEADING của Marsbot
- <Thời gian> là thời gian duy trì quá trình vận hành hiện tại
- <Cắt/Không cắt> thiết lập lưu vết/không lưu vết

Hãy lập trình để CNC Marsbot có thể:

- Thực hiện cắt kim loại như đã mô tả
- Nội dung postscript được lưu trữ cố định bên trong mã nguồn
- Mã nguồn chứa 3 postscript và người dùng sử dụng 3 phím 0, 4, 8 trên bàn phím Key Matrix để chọn postscript nào sẽ được gia công.
- Một postscript chứa chữ DCE cần gia công. Hai script còn lại sinh viên tự đề xuất (tối thiểu 10 đường cắt)



Postscript

20, 120,1,30,2100,1,90,3400,0….

0	1	2	3
4	5	6	7
8	9	a	b
c	d	e	f

5. Chương trình kiểm tra cú pháp lệnh MIPS

Trình biên dịch của bộ xử lý MIPS sẽ tiến hành kiểm tra cú pháp các lệnh hợp ngữ trong mã nguồn, xem có phù hợp về cú pháp hay không, rồi mới tiến hành dịch các lệnh ra mã máy. Hãy viết một chương trình kiểm tra cú pháp của 1 lệnh hợp ngữ MIPS bất kì (không làm với giả lệnh) như sau:

- Nhập vào từ bàn phím một dòng lệnh hợp ngữ. Ví dụ *beq s1,31,t4*
- *K*iểm tra xem mã opcode có đúng hay không? Trong ví dụ trên, opcode là beq là hợp lệ thì hiện thị thông báo *"opcode: beq, hợp lệ"*
- Kiểm tra xem tên các toán hạng phía sau có hợp lệ hay không? Trong ví dụ trên, *toán hạng s1 là hợp lệ, 31 là không hợp lệ, t4 thì khỏi phải kiểm tra nữa vì toán hạng trước đã bị sai rồi.*
- Cho biết lệnh hợp ngữ đó cần bao nhiêu chu kì thì mới thực hiện xong.

Gợi ý: nên xây dựng một cấu trúc chứa khuôn dạng của từng lệnh với tên lệnh, kiểu của toán hạng 1, toán hạng 2, toán hạng 3, số chu kì thực hiện.

6. Mô phỏng ổ đĩa RAID 5

Hệ thống ổ đĩa RAID5 cần tối thiểu 3 ổ đĩa cứng, trong đó phần dữ liệu parity sẽ được chứa lần lượt lên 3 ổ đĩa như trong hình bên. Hãy viết chương trình mô phỏng hoạt động của RAID 5 với 3 ổ đĩa, với giả định rằng, mỗi block dữ liệu có 4 kí tự. Giao diện như trong minh họa dưới. *Giới hạn chuỗi kí tự nhập vào có độ dài là bội của 8.*

Trong ví dụ sau, chuỗi kí tự nhập vào từ bàn phím *(DCE.****ABCD1234HUSTHUST*) sẽ được chia thành các block 4 byte. Block 4 byte đầu tiên "DCE." sẽ được lưu trên Disk 1, Block 4 byte tiếp theo "****" sẽ lưu trên Disk 2, dữ liệu trên Disk 3 sẽ là 4 byte parity được tính từ 2 block đầu tiên với mã ASCII là
6e='D' xor '*' ; 69='C' xor '*'; 6f='E' xor '*' ; 04='.' xor '*'

```
Nhap chuoi ki tu : DCE.****ABCD1234HUSTHUST
     Disk 1                Disk 2                Disk 3
 ------------------    ------------------    ------------------
|      DCE.      |    |      ****      |    [[ 6e,69,6f,04]]
|      ABCD      |    [[ 70,70,70,70]]    |      1234      |
[[ 00,00,00,00]]    |      HUST      |    |      HUST      |
 ------------------    ------------------    ------------------
```

7. Vẽ hình bằng kí tự ascii

Cho hình ảnh đã được chuyển thành các kí tự ascii như hình vẽ. Đây là hình của chữ DCE có viền * và màu là các con số.

```
                                                 *************
**************                                  *3333333333333*
*222222222222222*                                *33333********
*22222******222222*                              *33333*
*22222*      *22222*                             *33333********
*22222*       *22222*       *************       *3333333333333*
*22222*       *22222*     **11111*****111*      *33333********
*22222*       *22222*   **1111**       **       *33333*
*22222*      *222222*   *1111*                   *33333********
*22222*******222222*   *11111*                  *3333333333333*
*2222222222222222*     *11111*                   *************
***************        *11111*
      ---               *1111**
    / o o \              *1111****    *****
    \   > /               **111111***111*
     -----                  ***********     dce.hust.edu.vn
```

- Hãy hiển thị hình ảnh trên lên giao diện console (hoặc giao diện Display trong công cụ giả lập Keyboard and Display MMIO Simulator)
- Hãy sửa ảnh để các chữ cái DCE chỉ còn lại viền, không còn màu số ở giữa, và hiển thị
- Hãy sửa ảnh để hoán đổi vị trí của các chữ, thành ECD, và hiển thị. Để đơn giản, các hoạt tiết đính kèm cũng được phép di chuyển theo.
- Hãy nhập từ bàn phím kí tự màu cho chữ D, C, E, rồi hiển thị ảnh trên với màu mới.

Chú ý: ngoài vùng nhớ lớn chứa ảnh được chứa sẵn trong code, không được tạo thêm vùng nhớ mới để chứa ảnh hiệu chỉnh.

8. Máy tính bỏ túi

Sử dụng 2 ngoại vi là bàn phím và led 7 thanh để xây dựng một máy tính bỏ túi đơn giản. Hỗ trợ các phép toán +, -, *, /. Do trên bàn phím không có các phím trên nên sẽ dùng các phím

- Bấm phím a để nhập phép tính +
- Bấm phím b để nhập phép tính –
- Bấm phím c để nhập phép tính *
- Bấm phím d để nhập phép tính /
- Bấm phím f để nhập phép =

Yêu cầu cụ thể như sau:

- Khi nhấn các phím số, hiển thị lên LED, do chỉ có 2 LED nên chỉ hiện thị 2 số cuối cùng. Ví dụ khi nhấn phím 1 ➔ hiện thị 01. Khi nhấn thêm phím 2 ➔ hiển thị 12. Khi nhấn thêm phím 3 ➔ hiển thị 23.
- Sau khi nhập số, sẽ nhập phép tính + - * /
- Sau khi nhấn phím f (dấu =) , tính toán và hiển thị kết quả lên LED.

Chú ý: Do bài toán sẽ có rất nhiều trường hợp xảy ra, yêu cầu là thực hiện được phép tính và hiển thị lên LED, các yêu cầu về bắt lỗi, các trường hợp tràn số, ….

9. Trò chơi rắn săn mồi

Sử dụng Bitmap Display và Keyboard and Display MMIO simulator trong mars để mô phỏng trò chơi rắn

săn mồi.
Yêu cầu:

- Chạy code để tạo một con rắn trên Bitmap Display
- Các lệnh điều khiển được đọc từ Keyboard MMIO với: W- rắn đi lên, A-rắn di chuyển sang trái, S-rắn di chuyển xuống dưới, D- rắn di chuyển sang phải. Khi một trong các kí tự trên được bấm (W, A, S, D), trò chơi bắt đầu
- Trò chơi kết thúc khi rắn cắn chính mình hoặc đập vào thành