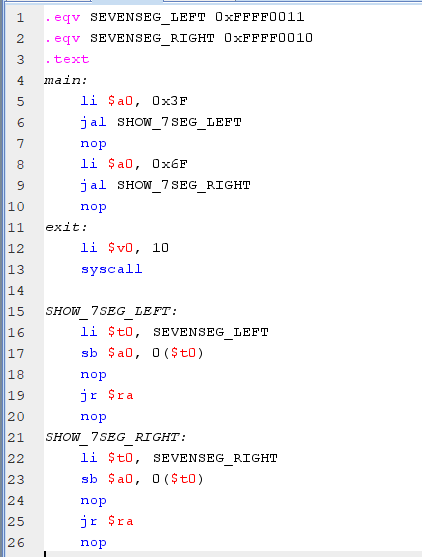
**Bài thực hành số 10**

**Lớp : 139365 – Học phần: Thực hành Kiến trúc máy tính**

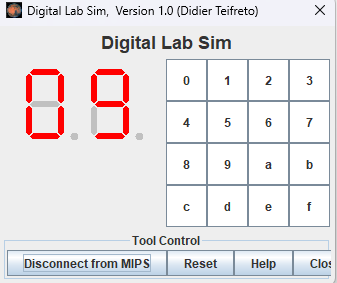
**Họ và tên : Nguyễn Thị Thùy Dung MSSV : 20215009**

**Bài 1:** Yêu cầu hiện ra 2 chữ số cuối mã số sinh viên (09)



*Thực hiện gõ chương trình với công cụ MARS*

* Kết quả:



* Giải thích :

Gán địa chỉ của đèn bên trái là 0XFFFF0011, địa chỉ đèn bên phải là 0X FFFF0010

Để hiện thị số 0 thì thanh g không được sáng

. g f e d c b a

0 0 1 1 1 1 1 1 -> 0x3F

Để hiện thị số 9 thì thanh e không được sáng

. g f e d c b a

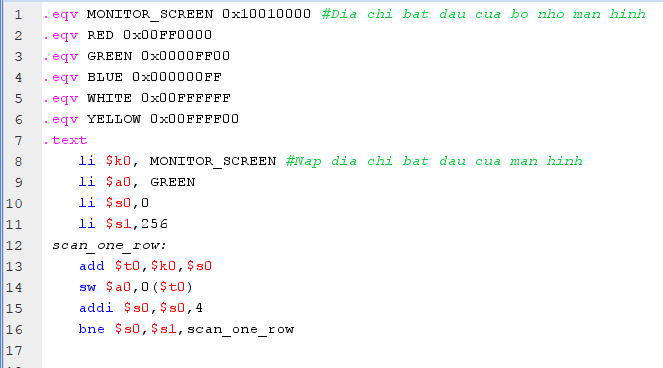
0 1 1 0 1 1 1 1 -> 0x6F

Gán $a0 = 0x3F, sau đó nhảy đến chương trình con SHOW\_7SEG\_LEFT, khởi tạo $t0 có giá trị là địa chỉ của đèn bên trái, dùng sb (store byte) để lưu giá trị $a0 vào nơi có địa chỉ là giá trị của $t0, khi đó đèn bên trái hiện lên số 0 như kết quả

Gán $a0 = 0x6F, sau đó nhảy đến chương trình con SHOW\_7SEG\_RIGHT, khởi tạo $t0 có giá trị là địa chỉ của đèn bên phải, dùng sb (store byte) để lưu giá trị $a0 vào nơi có địa chỉ là giá trị của $t0, khi đó đèn bên phải hiện lên số 9 như kết quả

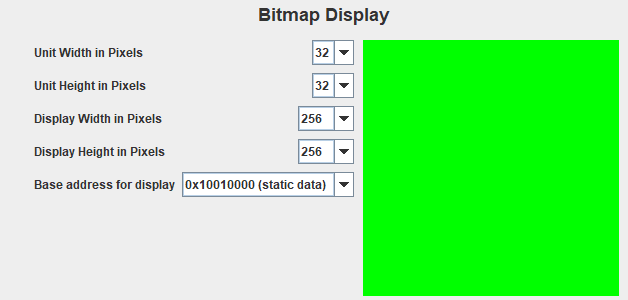
Sau đó quay lại chương trình chính bằng lệnh jr $ra gán $v0 = 10 và gọi hệ thống để thoát chương trình

**Bài 2:**



*Thực hiện gõ chương trình với công cụ MARS*

* Kết quả :



* Giải thích:

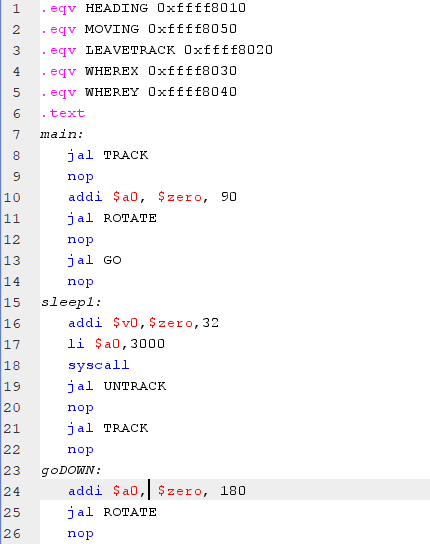
Khởi tạo địa chỉ bộ nhớ mà hình là 0x10010000, khởi tạo các màu cơ bản là : red (0x00FF0000), green(0x0000FF00), blue (0x000000FF),white (0x00FFFFFF), yellow (0x00FFFF00)

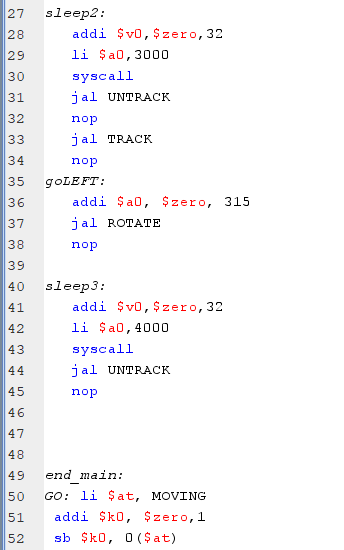
Gán $k0 bằng địa chỉ bắt đầu màn hình (0X10010000), gán $a0 có giá trị màu xanh (0X0000FF00), $s0 = 0, $s1 = 256

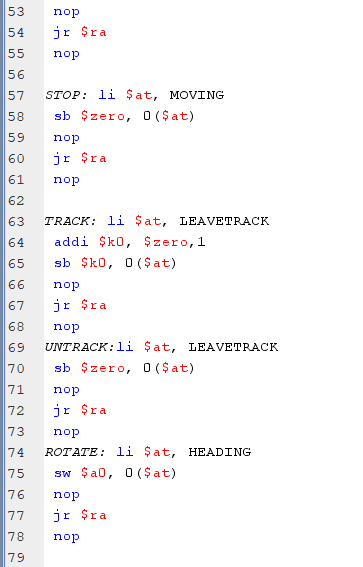
Thực hiện vòng lặp scan\_one\_row: $t0 = $k0 + $s0 (bằng địa chỉ đầu màn hình cộng với biến chạy $s0), sau đó sw (store word) $a0 vào nơi có địa chỉ là giá trị của thanh ghi $t0 lưu giữ thì ô đó sẽ có màu xanh . Cứ thực hiện vòng lặp cho đến khi $s0 = $s1 = 256(\*). Như vậy cả Bitmap sẽ có màu xanh như kết quả trên

(\*): Ở Bitmap Display trên được khởi tạo : Unit Width in Pixel là 32 tức là 1 ô có chiều rộng 32 bit (4 byte), Unit Height in Pixels là 32 tức là 1 ô có chiều cao 32 bit (4 byte), Display Width : 256 tức là chiều rộng là 256 bit suy ra có 8 ô

**Bài 3:**







*Thực hiện gõ chương trình với công cụ MARS*

* Kết quả :



* Giải thích :

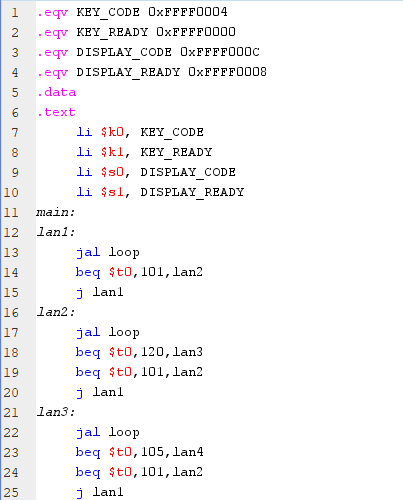
Khởi tạo các địa chỉ của các chỉ lệnh : Heading (0xffff8010), moving (0xffff8050), leavestrack(0xffff8020), wherex(0xffff8030), wherey(0xffff8040)

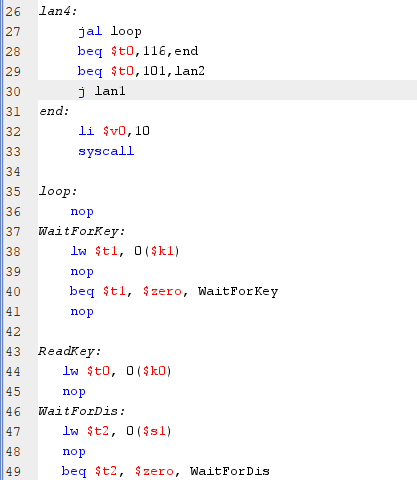
Để vẽ hình tam giác, trước tiên em vẽ đường ngang bằng cách nhảy tới hàm track bắt đầu đường track mới lưu giá trị $k0 = 1 vào ô có địa chỉ leavestrack, sau đó sử dụng hàm rotate gán $a0 = 90 cho ô có địa chỉ heading để nó xoay sang ngang, tiếp đến là hàm Go gán giá trị $k0 =1 cho ô có địa chỉ Moving để nó bắt đầu di chuyển và để nó di chuyển trong 3s với $v0 =32 , $a0 = 3000 . Cuối cùng là hàm untrack gán $zero cho ô có địa chỉ Leavestrack để nó giữ đường track cũ

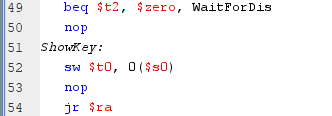
Tiếp tục vẽ đường dọc (go down ) bằng cách tạo track mới bằng hàm track sử dụng hàm rotate gán $a0 = 180 cho ô có địa chỉ heading để nó quay xuống dưới và để nó di chuyển trong 3s với $v0 =32 , $a0 = 3000 . Cuối cùng là hàm untrack gán $zero cho ô có địa chỉ Leavestrack để gữ đường track cũ

Cuối cùng là vẽ đường chéo sang trái (goLEFT) bằng cách sử dụng hàm rotate gán $a0 = 315 để nó quay sang trái để nó di chuyển trong 3s với $v0 =32 , $a0 = 3000 . Cuối cùng là hàm untrack gán $zero cho ô có địa chỉ Leavestrack để giữ đường track

**Bài 4:**

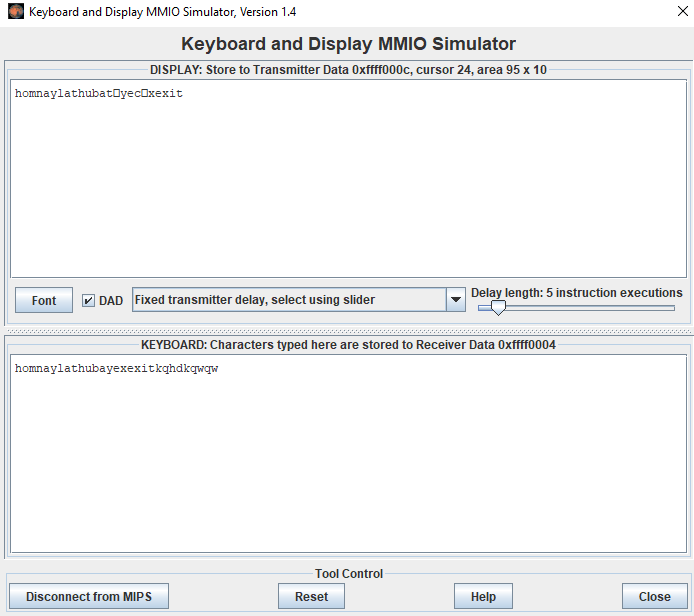
****

****

****

*Thực hành chương trình bằng công cụ MARS*

* Kết quả :

**

* Giải thích:

Ban đầu khởi tạo các địa chỉ key\_code(0xFFFF0004), key\_ready(0xFFFF0000), display\_code(0xFFFF000C), display\_ready(0xFFFF0008) và lần lượt gán các địa chỉ lần lượt cho $k0, $k1, $s0, $s1

Thực hiện vòng lặp : hàm đợi việc nhập WaitForKey lấy giá trị của từ ô có địa chỉ $k1 (key\_ready) lưu vào $t1, nếu $t1 = 0 thì vẫn quay lại waitforkey, nếu $t1 = 1 thì nhảy đến ReadKey, lấy giá trị từ ô có địa chỉ $k0 (Key\_code) lưu vào $t0, rồi nhảy đến WaitForDis lấy giá trị từ ô có địa chỉ $s1 (display\_ready) lưu vào $t2. Nếu $t2 = 0 thì lại chạy WaitForDis. Ngược lại sẽ chạy hàm ShowKey để lưu giá trị $t0 vào $s0 (Display\_code) để in ra

Ở mỗi vòng lặp như vậy kiểm tra ký tự vừa nhập $t0 với 101 (e), nếu được thì kiểm tra ký tự tiếp theo $t0 với 120 (x), nếu đúng tiếp tục kiểm tra $t0 với 105 (i) , cuối cùng kiểm tra $t0 với 116 (t). Nếu đúng tất cả sẽ kết thúc nhảy đến end và kết thúc chương trình với $v0 = 10 và syscall

Nếu không thỏa mãn ở kí tự vào sẽ quay về kiểm tra lại.