**REPORT WEEK 10**

Họ và tên: Nguyễn Văn Hưng

MSSV: 20225634

Lớp: Thực hành kiến trúc máy tính – Mã lớp: 147799

1. **Assignment 1:**
2. Đề bài: Hiển thị 2 chữ số cuối của MSSV
3. Code:

.eqv SEVENSEG\_LEFT 0xFFFF0010 # Dia chi cua den led 7 doan trai.

# Bit 0 = doan a;

# Bit 1 = doan b; ...

# Bit 7 = dau .

.eqv SEVENSEG\_RIGHT 0xFFFF0011 # Dia chi cua den led 7 doan phai

.text

main:

li $a0, 0x7D # set value for segments

jal SHOW\_7SEG\_LEFT # show

li $a0, 0x6D # set value for segments

jal SHOW\_7SEG\_RIGHT # show

exit: li $v0, 10

syscall

endmain:

#---------------------------------------------------------------

# Function SHOW\_7SEG\_LEFT : turn on/off the 7seg

# param[in] $a0 value to shown

# remark $t0 changed

#---------------------------------------------------------------

SHOW\_7SEG\_LEFT: li $t0, SEVENSEG\_LEFT # assign port's address

sb $a0, 0($t0) # assign new value

jr $ra

#---------------------------------------------------------------

# Function SHOW\_7SEG\_RIGHT : turn on/off the 7seg

# param[in] $a0 value to shown

# remark $t0 changed

#---------------------------------------------------------------

SHOW\_7SEG\_RIGHT: li $t0, SEVENSEG\_RIGHT # assign port's address

sb $a0, 0($t0) # assign new value

jr $ra

1. Text segment

A screenshot of a computer

Description automatically generated

1. Giải thích

Đoạn mã này điều khiển hai hiển thị bảy đoạn, một bên trái và một bên phải. Trong hàm **main**, giá trị được gán cho mỗi hiển thị và sau đó hiển thị được gọi để hiển thị chúng. Trong các hàm **SHOW\_7SEG\_LEFT** và **SHOW\_7SEG\_RIGHT**, địa chỉ của hiển thị được tải vào thanh ghi, và sau đó giá trị được lưu vào địa chỉ đó để hiển thị trên hiển thị tương ứng.

1. Kết quả

A screenshot of a computer

Description automatically generated

1. **Assignment 2:**
2. Đề bài: Nhập vào một số nguyên từ bàn phím, hiển thị 2 chữ số cuối của số nguyên đó.
3. Code:

.eqv SEVENSEG\_LEFT 0xFFFF0011 # Dia chi cua den led 7 doan trai.

# Bit 0 = doan a; # Bit 1 = doan b; ...

# Bit 7 = dau .

.eqv SEVENSEG\_RIGHT 0xFFFF0010 # Dia chi cua den led 7 doan phai

.data

message: .asciiz "Nhap vao mot so nguyen: "

.text

main:

li $v0, 4

la $a0, message

syscall

li $v0, 5

syscall

move $s0, $v0

li $t2, 10

div $s0, $t2

mfhi $t1

case0r: bne $t1, 0, case1r

li $a0, 0x3F

jal SHOW\_7SEG\_RIGHT

j defaultr

case1r: bne $t1, 1, case2r

li $a0, 0x6

jal SHOW\_7SEG\_RIGHT

j defaultr

case2r: bne $t1, 2, case3r

li $a0, 0x5B

jal SHOW\_7SEG\_RIGHT

j defaultr

case3r: bne $t1, 3, case4r

li $a0, 0x4F

jal SHOW\_7SEG\_RIGHT

j defaultr

case4r: bne $t1, 4, case5r

li $a0, 0x66

jal SHOW\_7SEG\_RIGHT

j defaultr

case5r: bne $t1, 5, case6r

li $a0, 0x6D

jal SHOW\_7SEG\_RIGHT

j defaultr

case6r: bne $t1, 6, case7r

li $a0, 0x7D

jal SHOW\_7SEG\_RIGHT

j defaultr

case7r: bne $t1, 7, case8r

li $a0, 0x7

jal SHOW\_7SEG\_RIGHT

j defaultr

case8r: bne $t1, 8, case9r

li $a0, 0x7F

jal SHOW\_7SEG\_RIGHT

j defaultr

case9r: bne $t1, 9, defaultr

li $a0, 0x6F

jal SHOW\_7SEG\_RIGHT

j defaultr

defaultr:

sub $s0, $s0, $t1

div $s0, $t2

mflo $t3

div $t3, $t2

mfhi $t1

case0l: bne $t1, 0, case1l

li $a0, 0x3F

jal SHOW\_7SEG\_LEFT

j defaultl

case1l: bne $t1, 1, case2l

li $a0, 0x6

jal SHOW\_7SEG\_LEFT

j defaultl

case2l: bne $t1, 2, case3l

li $a0, 0x5B

jal SHOW\_7SEG\_LEFT

j defaultl

case3l: bne $t1, 3, case4l

li $a0, 0x4F

jal SHOW\_7SEG\_LEFT

j defaultl

case4l: bne $t1, 4, case5l

li $a0, 0x66

jal SHOW\_7SEG\_LEFT

j defaultl

case5l: bne $t1, 5, case6l

li $a0, 0x6D

jal SHOW\_7SEG\_LEFT

j defaultl

case6l: bne $t1, 6, case7l

li $a0, 0x7D

jal SHOW\_7SEG\_LEFT

j defaultl

case7l: bne $t1, 7, case8l

li $a0, 0x7

jal SHOW\_7SEG\_LEFT

j defaultl

case8l: bne $t1, 8, case9l

li $a0, 0x7F

jal SHOW\_7SEG\_LEFT

j defaultl

case9l: bne $t1, 9, defaultl

li $a0, 0x6F

jal SHOW\_7SEG\_LEFT

j defaultl

defaultl:

li $v0, 10

syscall

endmain:

#---------------------------------------------------------------

# Function SHOW\_7SEG\_LEFT : turn on/off the 7seg

# param[in] $a0 value to shown

# remark $t0 changed

#---------------------------------------------------------------

SHOW\_7SEG\_LEFT:

li $t0, SEVENSEG\_LEFT # assign port's address

sb $a0, 0($t0) # assign new value

jr $ra

#---------------------------------------------------------------

# Function SHOW\_7SEG\_RIGHT : turn on/off the 7seg

# param[in] $a0 value to shown

# remark $t0 changed

#---------------------------------------------------------------

SHOW\_7SEG\_RIGHT:

li $t0, SEVENSEG\_RIGHT # assign port's address

sb $a0, 0($t0) # assign new value

jr $ra

1. Giải thích

Đoạn mã này thực hiện hai chức năng chính:

1. **Hiển thị số nguyên từ bàn phím lên hai hiển thị bảy đoạn bên trái và bên phải**:
   * Trước tiên, chuỗi "Nhap vao mot so nguyen: " được in ra màn hình để yêu cầu nhập một số nguyên.
   * Số nguyên được nhập từ bàn phím và lưu vào thanh ghi **$v0**.
   * Số nguyên này được chia lấy phần dư cho 10 để lấy giá trị của hàng đơn vị, và kết quả được lưu vào thanh ghi **$t1**.
   * Các trường hợp từ 0 đến 9 được kiểm tra để xác định giá trị của hàng đơn vị và hiển thị lên hiển thị bảy đoạn bên phải bằng cách gọi hàm **SHOW\_7SEG\_RIGHT**.
   * Sau đó, số nguyên được chia cho 10 để loại bỏ hàng đơn vị và kết quả được lưu vào thanh ghi **$s0**.
   * Quá trình tương tự được lặp lại cho hàng chục, nhưng giá trị hiển thị được gửi tới hiển thị bảy đoạn bên trái thông qua hàm **SHOW\_7SEG\_LEFT**.
2. **Hàm SHOW\_7SEG\_LEFT và SHOW\_7SEG\_RIGHT**:
   * Hai hàm này được sử dụng để hiển thị giá trị được truyền vào lên hiển thị bảy đoạn tương ứng (bên trái hoặc bên phải). Địa chỉ của hiển thị bảy đoạn được tải vào thanh ghi **$t0**, sau đó giá trị được lưu vào địa chỉ đó để hiển thị lên hiển thị bảy đoạn.
3. Kết quả

A screenshot of a computer

Description automatically generated

1. **Assignment 3:**
2. Đề bài: Nhập vào ký tự, hiển thị 2 chữ số cuối của mã ASCII của ký tự đó.
3. Code:

.eqv SEVENSEG\_LEFT 0xFFFF0011 # Dia chi cua den led 7 doan trai.

# Bit 0 = doan a; # Bit 1 = doan b; ...

# Bit 7 = dau .

.eqv SEVENSEG\_RIGHT 0xFFFF0010 # Dia chi cua den led 7 doan phai

.data

message: .asciiz "Nhap vao mot ky tu: "

.text

main:

li $v0, 4

la $a0, message

syscall

li $v0, 12

syscall

move $s0, $v0

li $t1, -1

For: addi $t1, $t1, 1 #Lay gia tri ma ascii cua ky tu roi luu vao thanh $t1

beq $s0, $t1, EndFor

j For

EndFor:

move $s0, $t1

li $t2, 10

div $s0, $t2

mfhi $t1

case0r:

bne $t1, 0, case1r

li $a0, 0x3F

jal SHOW\_7SEG\_RIGHT

j defaultr

case1r:

bne $t1, 1, case2r

li $a0, 0x6

jal SHOW\_7SEG\_RIGHT

j defaultr

case2r:

bne $t1, 2, case3r

li $a0, 0x5B

jal SHOW\_7SEG\_RIGHT

j defaultr

case3r: bne $t1, 3, case4r

li $a0, 0x4F

jal SHOW\_7SEG\_RIGHT

j defaultr

case4r: bne $t1, 4, case5r

li $a0, 0x66

jal SHOW\_7SEG\_RIGHT

j defaultr

case5r: bne $t1, 5, case6r

li $a0, 0x6D

jal SHOW\_7SEG\_RIGHT

j defaultr

case6r: bne $t1, 6, case7r

li $a0, 0x7D

jal SHOW\_7SEG\_RIGHT

j defaultr

case7r: bne $t1, 7, case8r

li $a0, 0x7

jal SHOW\_7SEG\_RIGHT

j defaultr

case8r: bne $t1, 8, case9r

li $a0, 0x7F

jal SHOW\_7SEG\_RIGHT

j defaultr

case9r: bne $t1, 9, defaultr

li $a0, 0x6F

jal SHOW\_7SEG\_RIGHT

j defaultr

defaultr:

sub $s0, $s0, $t1

div $s0, $t2

mflo $t3

div $t3, $t2

mfhi $t1

case0l:

bne $t1, 0, case1l

li $a0, 0x3F

jal SHOW\_7SEG\_LEFT

j defaultl

case1l:

bne $t1, 1, case2l

li $a0, 0x6

jal SHOW\_7SEG\_LEFT

j defaultl

case2l:

bne $t1, 2, case3l

li $a0, 0x5B

jal SHOW\_7SEG\_LEFT

j defaultl

case3l:

bne $t1, 3, case4l

li $a0, 0x4F

jal SHOW\_7SEG\_LEFT

j defaultl

case4l:

bne $t1, 4, case5l

li $a0, 0x66

jal SHOW\_7SEG\_LEFT

j defaultl

case5l:

bne $t1, 5, case6l

li $a0, 0x6D

jal SHOW\_7SEG\_LEFT

j defaultl

case6l:

bne $t1, 6, case7l

li $a0, 0x7D

jal SHOW\_7SEG\_LEFT

j defaultl

case7l:

bne $t1, 7, case8l

li $a0, 0x7

jal SHOW\_7SEG\_LEFT

j defaultl

case8l:

bne $t1, 8, case9l

li $a0, 0x7F

jal SHOW\_7SEG\_LEFT

j defaultl

case9l:

bne $t1, 9, defaultl

li $a0, 0x6F

jal SHOW\_7SEG\_LEFT

j defaultl

defaultl:

li $v0, 10

syscall

endmain:

#---------------------------------------------------------------

# Function SHOW\_7SEG\_LEFT : turn on/off the 7seg

# param[in] $a0 value to shown

# remark $t0 changed

#---------------------------------------------------------------

SHOW\_7SEG\_LEFT:

li $t0, SEVENSEG\_LEFT # assign port's address

sb $a0, 0($t0) # assign new value

jr $ra

#---------------------------------------------------------------

# Function SHOW\_7SEG\_RIGHT : turn on/off the 7seg

# param[in] $a0 value to shown

# remark $t0 changed

#---------------------------------------------------------------

SHOW\_7SEG\_RIGHT:

li $t0, SEVENSEG\_RIGHT # assign port's address

sb $a0, 0($t0) # assign new value

jr $ra

1. Giải thích

Đoạn mã này là một chương trình MIPS assembly đơn giản để hiển thị một ký tự được nhập từ bàn phím lên hai hiển thị bảy đoạn, một bên trái và một bên phải.

1. Đầu tiên, địa chỉ của hai hiển thị bảy đoạn, một bên trái và một bên phải, được định nghĩa bằng các hằng số **.eqv**.
2. Chương trình in ra màn hình chuỗi "Nhap vao mot ky tu: " để yêu cầu người dùng nhập một ký tự.
3. Sau đó, một ký tự được nhập từ bàn phím và lưu vào thanh ghi **$v0**.
4. Ký tự này sau đó được chuyển từ mã ASCII sang giá trị thập phân và lưu vào thanh ghi **$s0**.
5. Giá trị thập phân này được chia lấy phần dư cho 10 để lấy giá trị của hàng đơn vị, sau đó được kiểm tra và hiển thị lên hiển thị bảy đoạn bên phải.
6. Tương tự, giá trị được chia cho 10 để loại bỏ hàng đơn vị, và kết quả được lưu vào thanh ghi **$s0**. Quá trình này lặp lại cho hàng chục, và giá trị được hiển thị lên hiển thị bảy đoạn bên trái.
7. Cuối cùng, chương trình kết thúc và thoát.
8. Kết quả

A screenshot of a computer

Description automatically generated

1. **Assignment 4:**
2. Đề bài: Vẽ ô bàn cờ vua kích thước 8x8, chọn 2 màu bất kỳ (khác màu đen).
3. Code:

.eqv MONITOR\_SCREEN 0x10010000

.eqv RED 0x00FF0000

.eqv GREEN 0x0000FF00

.text

li $k0, MONITOR\_SCREEN

li $s0, 2

li $t0, -1 # Khoi tao j

lap1: addi $t0, $t0, 1

beq $t0, 8, todo

li $t1, -1 # Khoi tao i

lap2: addi $t1, $t1, 1

beq $t1, 8, Endlap2

div $t0, $s0

mfhi $t2

div $t1, $s0

mfhi $t3

bne $t2, 0, nhan1

bne $t3, 0, lap2

j nhan2

nhan1: beq $t3, 0, lap2

nhan2: sll $s1, $t0, 3

add $s1, $s1, $t1

sll $s1, $s1, 2

add $s2, $s1, $k0

li $t4, GREEN

sw $t4, 0($s2)

j lap2

Endlap2:

j lap1

todo:

li $s0, 2

li $t0, -1 # Khoi tao j

For1: addi $t0, $t0, 1

beq $t0, 8, Exit

li $t1, -1 # Khoi tao i

For2: addi $t1, $t1, 1

beq $t1, 8, EndFor2

div $t0, $s0

mfhi $t2

div $t1, $s0

mfhi $t3

beq $t2, 0, loop1

bne $t3, 0, For2

j loop2

loop1: beq $t3, 0, For2

loop2: sll $s1, $t0, 3

add $s1, $s1, $t1

sll $s1, $s1, 2

add $s2, $s1, $k0

li $t4, RED

sw $t4, 0($s2)

j For2

EndFor2:

j For1

Exit: li $v0, 10

syscall

1. Giải thích

1. Định nghĩa hằng số:

- `.eqv MONITOR\_SCREEN 0x10010000`: Địa chỉ của màn hình máy tính.

- `.eqv RED 0x00FF0000`: Mã màu đỏ.

- `.eqv GREEN 0x0000FF00`: Mã màu xanh lá cây.

2. Khởi tạo thanh ghi:

- `li $k0, MONITOR\_SCREEN`: Khởi tạo thanh ghi `$k0` với địa chỉ của màn hình máy tính.

3. Vòng lặp ngoài (`lap1`):

- Lặp từ `0` đến `7` (`beq $t0, 8, todo`).

- Vòng lặp trong (`lap2`):

- Lặp từ `0` đến `7` (`beq $t1, 8, Endlap2`).

- Tính toán và kiểm tra điều kiện để xác định vị trí của pixel trong màn hình.

- Nếu điều kiện được thỏa mãn, một pixel được thiết lập thành màu xanh lá cây.

- Sau khi xử lý mỗi pixel, vòng lặp trong tiếp tục.

4. Đánh dấu vòng lặp tiếp theo (`todo`):

- Đây là một phần lặp tương tự như `lap1` và `lap2`, nhưng lần này sử dụng màu đỏ.

5. Sau khi tạo ra mẫu hình vuông với màu xanh lá cây và màu đỏ, chương trình kết thúc bằng cách gọi lệnh hệ thống để kết thúc chương trình.

Vậy là đoạn mã này tạo ra một mẫu hình vuông trên màn hình máy tính với các pixel được tô màu đỏ và xanh lá cây.

1. Kết quả

A screenshot of a computer

Description automatically generated

1. **Assignment 5:**
2. Đề bài: Nhập vào 2 điểm có tọa độ (x1, y1) và (x2, y2) (x1 khác x2 và y1 khác y2), vẽ và tô màu hình chữ nhật có 2 góc là 2 điểm trên với viền màu đỏ rộng 1 đơn vị và nền màu xanh lá.
3. Code:

.eqv MONITOR\_SCREEN 0x10010000

.eqv RED 0x00FF0000

.eqv GREEN 0x0000FF00

.data

x1: .asciiz "Nhap x1: "

y1: .asciiz "Nhap y1: "

x2: .asciiz "Nhap x2: "

y2: .asciiz "Nhap y2: "

error1: .asciiz "Error: x2 phai khac x1. Moi nhap lai!\n"

error2: .asciiz "Error: y2 phai khac y1. Moi nhap lai!\n"

.text

li $k0, MONITOR\_SCREEN

li $v0, 4

la $a0, x1

syscall

li $v0, 5

syscall

move $s0, $v0

li $v0, 4

la $a0, y1

syscall

li $v0, 5

syscall

move $s1, $v0

NhapX2: li $v0, 4

la $a0, x2

syscall

li $v0, 5

syscall

move $s2, $v0

beq $s2, $s0, Error1

NhapY2: li $v0, 4

la $a0, y2

syscall

li $v0, 5

syscall

move $s3, $v0

beq $s3, $s1, Error2

j loop

Error1: li $v0, 4

la $a0, error1

syscall

j NhapX2

Error2: li $v0, 4

la $a0, error2

syscall

j NhapY2

loop:

slt $t0, $s0, $s2

slt $t1, $s1, $s3

beq $t0, 0, Case3

beq $t1, 0, Case2

Case1: add $v0, $s1, $zero

For1: bgt $v0, $s3, Exit

add $v1, $s0, $zero

For2: bgt $v1, $s2, EndFor2

beq $v0, $s1, In1

beq $v0, $s3, In1

beq $v1, $s0, In1

beq $v1, $s2, In1

sll $t8, $v0, 6

add $t8, $t8, $v1

sll $t8, $t8, 2

li $a1, GREEN

add $a2, $k0, $t8

sw $a1, 0($a2)

add $v1, $v1, 1

j For2

In1: sll $t8, $v0, 6

add $t8, $t8, $v1

sll $t8, $t8, 2

li $a1, RED

add $a2, $k0, $t8

sw $a1, 0($a2)

add $v1, $v1, 1

j For2

EndFor2:

add $v0, $v0, 1

j For1

Case2: add $v0, $s3, $zero

For3: bgt $v0, $s1, Exit

add $v1, $s0, $zero

For4: bgt $v1, $s2, EndFor4

beq $v0, $s1, In2

beq $v0, $s3, In2

beq $v1, $s0, In2

beq $v1, $s2, In2

sll $t8, $v0, 6

add $t8, $t8, $v1

sll $t8, $t8, 2

li $a1, GREEN

add $a2, $k0, $t8

sw $a1, 0($a2)

add $v1, $v1, 1

j For4

In2:sll $t8, $v0, 6

add $t8, $t8, $v1

sll $t8, $t8, 2

li $a1, RED

add $a2, $k0, $t8

sw $a1, 0($a2)

add $v1, $v1, 1

j For4

EndFor4:

add $v0, $v0, 1

j For3

Case3: beq $t1, 0, Case4

add $v0, $s1, $zero

For5: bgt $v0, $s3, Exit

add $v1, $s2, $zero

For6: bgt $v1, $s0, EndFor6

beq $v0, $s1, In3

beq $v0, $s3, In3

beq $v1, $s0, In3

beq $v1, $s2, In3

sll $t8, $v0, 6

add $t8, $t8, $v1

sll $t8, $t8, 2

li $a1, GREEN

add $a2, $k0, $t8

sw $a1, 0($a2)

add $v1, $v1, 1

j For6

In3:sll $t8, $v0, 6

add $t8, $t8, $v1

sll $t8, $t8, 2

li $a1, RED

add $a2, $k0, $t8

sw $a1, 0($a2)

add $v1, $v1, 1

j For6

EndFor6:

add $v0, $v0, 1

j For5

Case4: add $v0, $s3, $zero

For7: bgt $v0, $s1, Exit

add $v1, $s2, $zero

For8: bgt $v1, $s0, EndFor8

beq $v0, $s1, In4

beq $v0, $s3, In4

beq $v1, $s0, In4

beq $v1, $s2, In4

sll $t8, $v0, 6

add $t8, $t8, $v1

sll $t8, $t8, 2

li $a1, GREEN

add $a2, $k0, $t8

sw $a1, 0($a2)

add $v1, $v1, 1

j For8

In4:sll $t8, $v0, 6

add $t8, $t8, $v1

sll $t8, $t8, 2

li $a1, RED

add $a2, $k0, $t8

sw $a1, 0($a2)

add $v1, $v1, 1

j For8

EndFor8:

add $v0, $v0, 1

j For7

Exit: li $v0, 10

syscall

1. Giải thích

1. Đầu tiên, chương trình hiển thị các thông báo yêu cầu người dùng nhập vào các giá trị x1, y1, x2, y2.

2. Sau đó, nó sẽ lưu giữ các giá trị người dùng nhập vào trong các biến $s0, $s1, $s2, $s3.

3. Chương trình kiểm tra xem x2 có khác x1 không. Nếu không, nó hiển thị một thông báo lỗi và yêu cầu người dùng nhập lại x2.

4. Sau khi xác định được x2 khác x1, chương trình tiếp tục kiểm tra y2 có khác y1 không. Nếu không, nó hiển thị một thông báo lỗi và yêu cầu người dùng nhập lại y2.

5. Khi đã có tất cả các giá trị hợp lệ của x và y, chương trình bắt đầu vẽ hình chữ nhật.

6. Nó sử dụng các vòng lặp để duyệt qua các điểm trên màn hình và tô màu cho hình chữ nhật.

7. Sau khi vẽ xong, chương trình kết thúc.

1. Kết quả

A screenshot of a computer

Description automatically generated

A number and numbers on a white background

Description automatically generated