BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH

LAB 10

Họ và tên: Phạm Vân Anh

MSSV: 20214988

Mã lớp: 139365

ASSIGNMENT 1:

1. Code

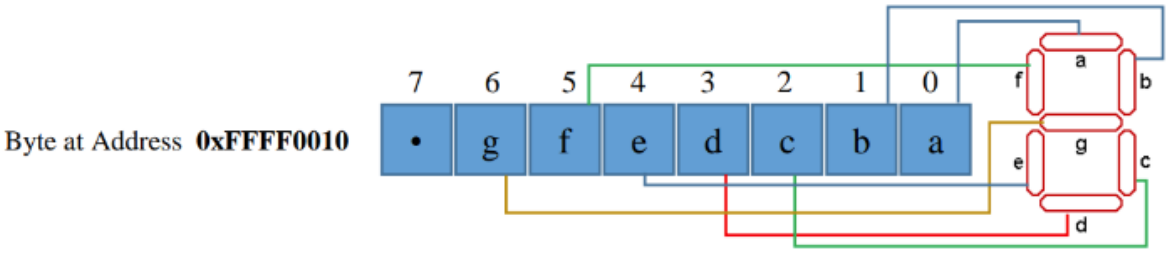
```
#Laboratory Exercise 10 Home Assignment 1
#PhamVanAnh_20214988

.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
.text
main:
      li    $a0, 0x6F           # set value for segments
      jal   SHOW_7SEG_LEFT      # show
      nop
      li    $a0, 0xFF           # set value for segments
      jal   SHOW_7SEG_RIGHT     # show
      nop
exit:
      li    $v0, 10
      syscall
endmain:
```

```
21  #---------------------------------------------------------------
22  # Function SHOW_7SEG_LEFT : turn on/off the 7seg
23  # param[in] $a0 value to shown
24  # remark $t0 changed
25  #---------------------------------------------------------------
26  SHOW_7SEG_LEFT:
27        li    $t0, SEVENSEG_LEFT # assign port's address
28        sb    $a0, 0($t0)        # assign new value
29        nop
30        jr    $ra
31        nop
32  #---------------------------------------------------------------
33  # Function SHOW_7SEG_RIGHT : turn on/off the 7seg
34  # param[in] $a0 value to shown
35  # remark $t0 changed
36  #---------------------------------------------------------------
37  SHOW_7SEG_RIGHT:
38        li    $t0, SEVENSEG_RIGHT # assign port's address
39        sb    $a0, 0($t0)         # assign new value
40        nop
41        jr $ra
42        nop
```

Thực hiện gõ chương trình vào công cụ MARS

2. Giải thích

- SEVENSEG_LEFT và SEVENSEG_RIGHT đều đã có sẵn địa chỉ tùy theo nhà sản xuất



- SHOW_7SEG_LEFT và SHOW_7SEG_RIGHT là các hàm hiển thị số ra theo led 7 thanh.
- $a0 là mã nhị phân của số mình muốn hiển thị, muốn đoạn đèn nào sáng ta sẽ gán bit của thanh đó là 1, còn tắt là 0.
- Như trong bài, muốn hiển thị số
  - 9: Đèn a, b, f, g, c, d sáng và đèn e tắt. Mã nhị phân nhập vào sẽ là 0x11110111 = 0x6F
  - 8: Tất cả các đèn đều sáng. Mã nhị phân nhập vào sẽ là 0x11111111 = 0Xff

3. Kết quả

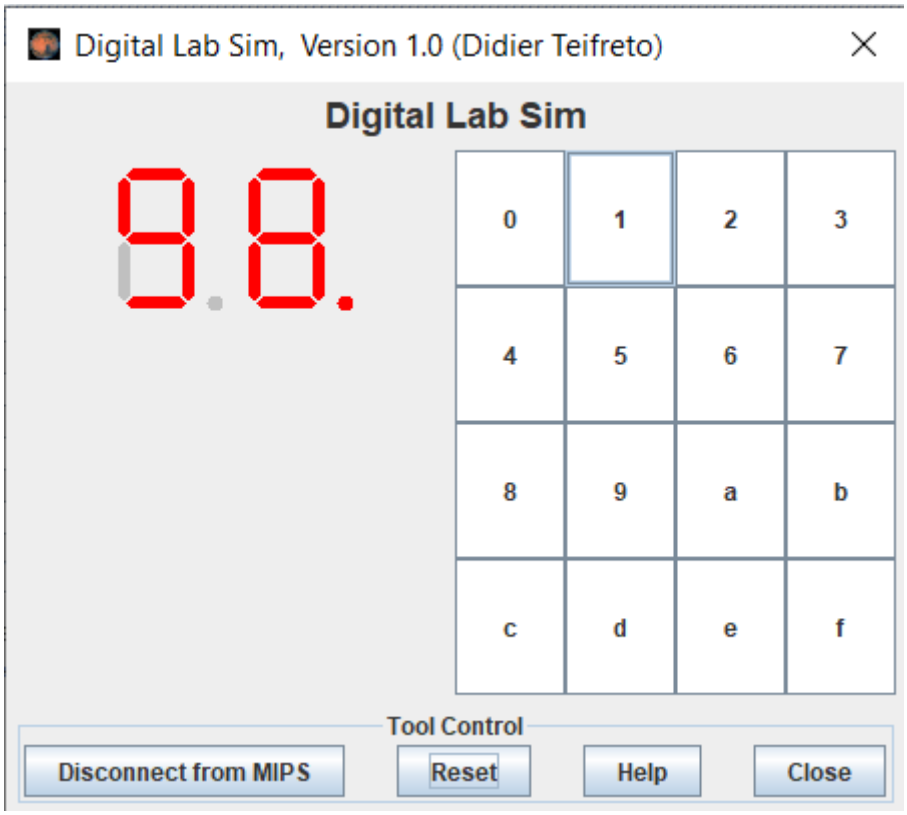
- Địa chỉ $a0 khi hiển thị số 9:

$a0	4	0x0000006f

- Địa chỉ $a0 khi hiển thị số 8:

$a0	4	0x000000ff

- Kết quả:



ASSIGNMENT 2:

1. Code

```
#Laboratory Exercise 10 Home Assignment 2
#PhamVanAnh_20214988

.eqv MONITOR_SCREEN 0x10010000 #Dia chi bat dau cua bo nho man hinh
.eqv RED 0x00FF0000 #Cac gia tri mau thuong su dung
.eqv GREEN 0x0000FF00
.eqv BLUE 0x000000FF
.eqv WHITE 0x00FFFFFF
.eqv YELLOW 0x00FFFF00

.text
      li    $k0, MONITOR_SCREEN #Nap dia chi bat dau cua man hinh
      li    $t0, WHITE
      sw    $t0, 0($k0)
      nop

      li    $t0, WHITE
      sw    $t0, 28($k0)
      nop

      li    $t0, WHITE
      sw    $t0, 36($k0)
      nop

      li    $t0, WHITE
      sw    $t0, 56($k0)
      nop

      li    $t0, WHITE
      sw    $t0, 68($k0)
      nop
```

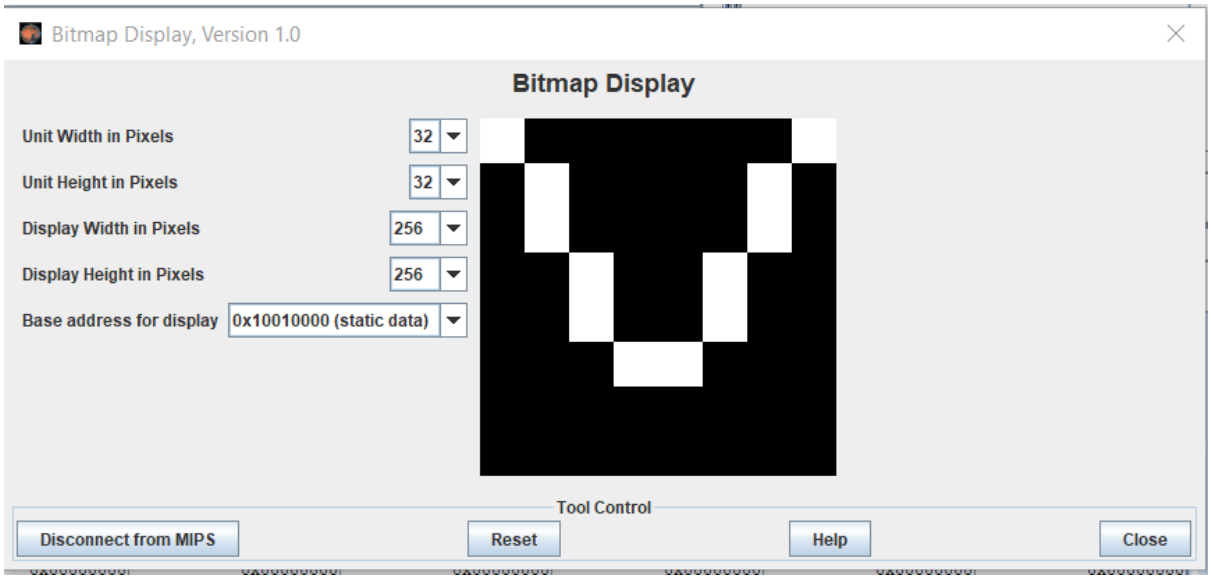
```
32
33          li    $t0, WHITE
34          sw    $t0, 88($k0)
35          nop
36
37          li    $t0, WHITE
38          sw    $t0, 104($k0)
39          nop
40
41          li    $t0, WHITE
42          sw    $t0, 116($k0)
43          nop
44
45          li    $t0, WHITE
46          sw    $t0, 136($k0)
47          nop
48
49          li    $t0, WHITE
50          sw    $t0, 148($k0)
51          nop
52
53          li    $t0, WHITE
54          sw    $t0, 172($k0)
55          nop
56
57          li    $t0, WHITE
58          sw    $t0, 176($k0)
59          nop
```

Thực hiện gõ chương trình vào công cụ MARS

2. Giải thích

- Hiển thị theo pixel trên màn hình 256x256 và mỗi pixel là 32x32 nên ta có 8 hàng và 8 cột.
- Xây dựng 8 mảng, mỗi mảng 8 phần tử tương ứng theo hàng và mỗi phần tử của mảng là địa chỉ của màu để in
- Mỗi pixel có địa chỉ cách nhau 4 byte, nên ta cộng thêm địa chỉ rồi in ra màu

3. Kết quả


Bitmap Display, Version 1.0
Bitmap Display
Unit Width in Pixels
32
Unit Height in Pixels
32
Display Width in Pixels
256
Display Height in Pixels
256
Base address for display
0x10010000 (static data)
Tool Control
Disconnect from MIPS
Reset
Help
Close

ASSIGNMENT 3:

1. Code

```
#Laboratory Exercise 10 Home Assignment 3
#PhamVanAnh_20214988
.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359
                        # 0 : North (up)
                        # 90: East (right)
                        # 180: South (down)
                        # 270: West (left)
.eqv MOVING 0xffff8050  # Boolean: whether or not to move
.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):
                        # whether or not to leave a track
.eqv WHEREX 0xffff8030  # Integer: Current x-location of
#MarsBot
.eqv WHEREY 0xffff8040 # Integer: Current y-location of
#MarsBot
.text
main:
      jal   TRACK             # draw track line
      nop
      addi  $a0, $zero, 90 # Marsbot rotates 90* and start
#running
      jal   ROTATE
      nop
      jal   GO
      nop
sleep1:
      addi  $v0, $zero, 32 # Keep running by sleeping in 1000 ms
      li    $a0, 1000
      syscall

      jal   UNTRACK # keep old track
      nop
      jal   TRACK # and draw new track line
      nop
```

```
34  goDOWN:
35        addi  $a0, $zero, 180 # Marsbot rotates 180*
36        jal   ROTATE
37        nop
38
39  sleep2:
40        addi  $v0,$zero,32 # Keep running by sleeping in 2000 ms
41        li    $a0,2000
42        syscall
43        jal   UNTRACK # keep old track
44        nop
45        jal   TRACK # and draw new track line
46        nop
47  goLEFT:
48        addi  $a0, $zero, 270 # Marsbot rotates 270*
49        jal   ROTATE
50        nop
51
52  sleep3:
53        addi  $v0,$zero,32 # Keep running by sleeping in 1000 ms
54        li    $a0,1000
55        syscall
56        jal   UNTRACK # keep old track
57        nop
58        jal   TRACK # and draw new track line
59        nop
```

```
61  goASKEW:
62        addi  $a0, $zero, 150 # Marsbot rotates 120*
63        jal   ROTATE
64        nop
65
66
67  sleep4:
68        addi  $v0,$zero,32 # Keep running by sleeping in 5000 ms
69        li    $a0,10000
70        syscall
71
72        jal   UNTRACK # keep old track
73        nop
74        jal   TRACK # and draw new track line
75        nop
76
77  goASKEW1:
78        addi  $a0, $zero, 30 # Marsbot rotates 120*
79        jal   ROTATE
80        nop
81  sleep5:
82        addi  $v0,$zero,32 # Keep running by sleeping in 5000 ms
83        li    $a0,5000
84        syscall
85
86        jal   UNTRACK # keep old track
87        nop
88        jal   TRACK # and draw new track line
89        nop
90
91  end_main:
```

```
93  #------------------------------------------------------------
94  # GO procedure, to start running
95  # param[in] none
96  #------------------------------------------------------------
97  GO:
98        li    $at, MOVING # change MOVING port
99        addi  $k0, $zero,1 # to logic 1,
100       sb    $k0, 0($at) # to start running
101       nop
102       jr    $ra
103       nop
104 #------------------------------------------------------------
105 # STOP procedure, to stop running
106 # param[in] none
107 #------------------------------------------------------------
108 STOP:
109       li    $at, MOVING # change MOVING port to 0
110       sb    $zero, 0($at) # to stop
111       nop
112       jr    $ra
113       nop
114 #------------------------------------------------------------
115 # TRACK procedure, to start drawing line
116 # param[in] none
```

```
117 #------------------------------------------------------------
118 TRACK:
119       li    $at, LEAVETRACK # change LEAVETRACK port
120       addi  $k0, $zero,1 # to logic 1,
121       sb    $k0, 0($at) # to start tracking
122       nop
123       jr    $ra
124       nop
125 #------------------------------------------------------------
126 # UNTRACK procedure, to stop drawing line
127 # param[in] none
128 #------------------------------------------------------------
129 UNTRACK:
130       li    $at, LEAVETRACK # change LEAVETRACK port to 0
131       sb    $zero, 0($at) # to stop drawing tail
132       nop
133       jr    $ra
134       nop
135 #------------------------------------------------------------
136 # ROTATE procedure, to rotate the robot
137 # param[in] $a0, An angle between 0 and 359
138 # 0 : North (up)
139 # 90: East (right)
140 # 180: South (down)
141 # 270: West (left)
142 #------------------------------------------------------------
143 ROTATE:
144       li    $at, HEADING # change HEADING port
145       sw    $a0, 0($at) # to rotate robot
146       nop
147       jr    $ra
148       nop
```

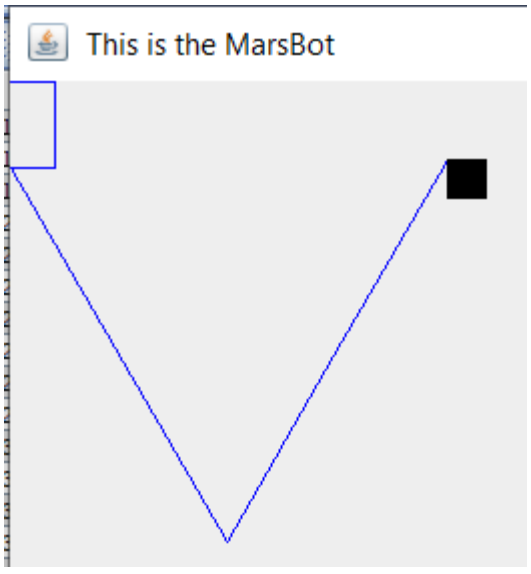
Thực hiện gõ chương trình vào công cụ MARS

2. Giải thích

- Hàm ROTATE: chỉnh góc của con trỏ để đi tiếp

-
- Hàm STOP: set giá trị của MOVING về 0 để dừng trỏ
- Hàm GO: set MOVING về 1 → con trỏ sẽ di chuyển theo góc cho trước qua hàm ROTATE
- Hàm TRACK: set LEAVETRACK về 1 → bắt đầu vẽ
- Hàm UNTRACK: ngược với TRACK, set LEAVETRACK về 0 → dừng vẽ
- Các hàm loop để set thời gian con trỏ di chuyển theo hướng vừa up của hàm ROTATE để vẽ theo ý muốn
- Vẽ chữ V:
  - Quay xuống 1 góc 150 độ, delay 10s
  - Quay lên 1 góc 30 độ

3. Kết quả



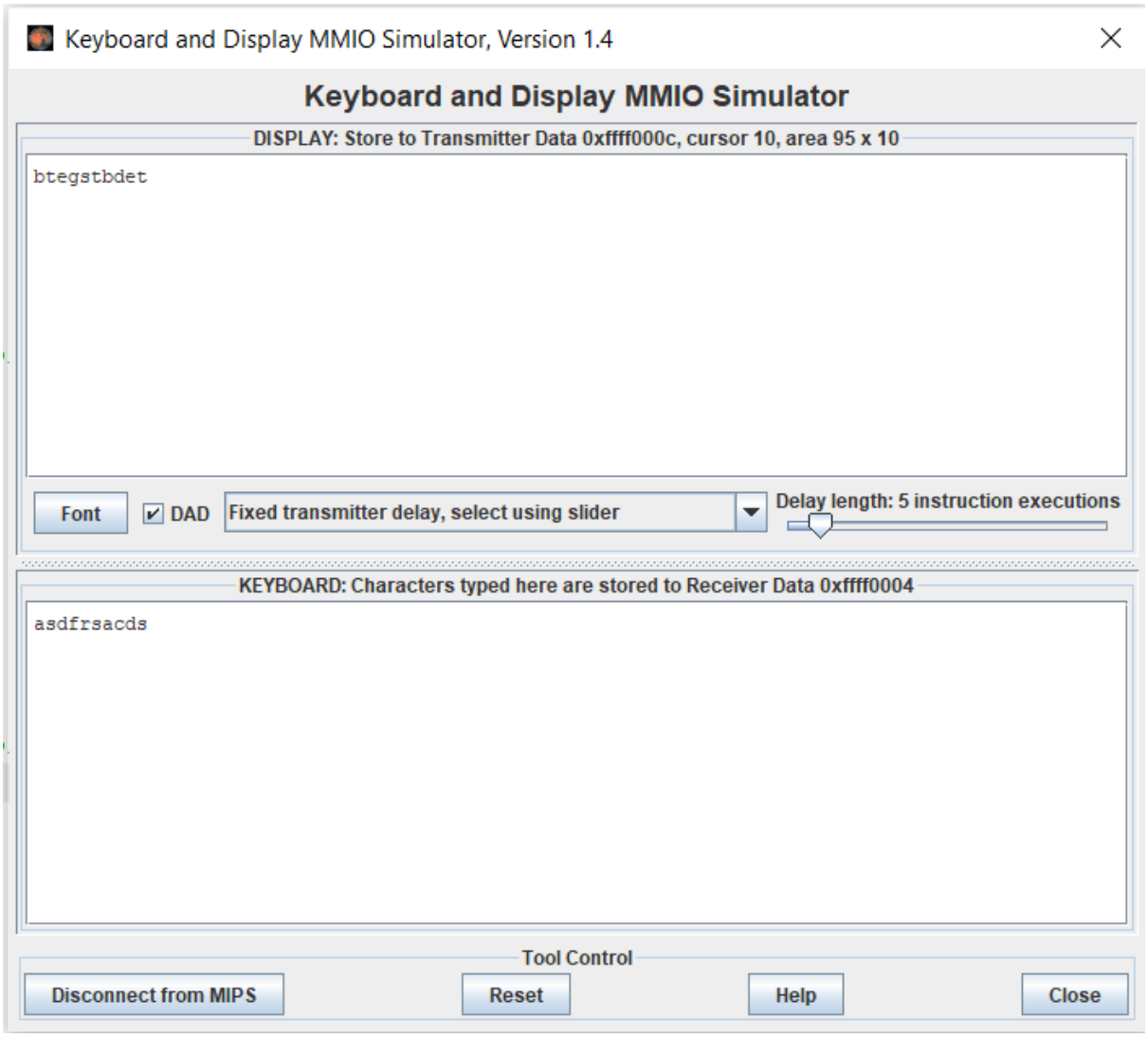
ASSIGNMENT 4:

1. Code

```
#Laboratory Exercise 10 Home Assignment 4
#PhamVanAnh_20214988

.eqv KEY_CODE 0xFFFF0004      # ASCII code from keyboard, 1 byte
.eqv KEY_READY 0xFFFF0000     # =1 if has a new keycode ?
                              # Auto clear after lw
.eqv DISPLAY_CODE 0xFFFF000C  # ASCII code to show, 1 byte
.eqv DISPLAY_READY 0xFFFF0008 # =1 if the display has already to do
                              # Auto clear after sw
.text
      li    $k0, KEY_CODE
      li    $k1, KEY_READY

      li    $s0, DISPLAY_CODE
      li    $s1, DISPLAY_READY
loop: nop

WaitForKey:
      lw    $t1, 0($k1) # $t1 = [$k1] = KEY_READY
      nop
      beq   $t1, $zero, WaitForKey # if $t1 == 0 then Polling
      nop
#---------------------------------------------------
ReadKey:
      lw    $t0, 0($k0) # $t0 = [$k0] = KEY_CODE
      nop
#---------------------------------------------------
WaitForDis:
      lw    $t2, 0($s1) # $t2 = [$s1] = DISPLAY_READY
      nop
      beq   $t2, $zero, WaitForDis # if $t2 == 0 then Polling
      nop
```

```
#---------------------------------------------------
Encrypt:
      addi  $t0, $t0, 1 # change input key
#---------------------------------------------------
ShowKey:
      sw    $t0, 0($s0) # show key
      nop
#---------------------------------------------------
      j loop
      nop
```

2. **Giải thích**
    - KEY_CODE 0xFFFF0004: địa chỉ để đọc mã ASCII từ bàn phím
    - KEY_READY 0Xffff0000: cờ báo lệnh có phím được ấn hay không. Nếu có giá trị bằng 1, nghĩa là có mã ASCII mới sẵn sàng để đọc từ bàn phím. Tự động xóa sau khi được đọc.
    - DISPLAY_CODE 0xFFFF000C: mã ASCII cần hiển thị
    - DISPLA_READY 0xFFFF0008: cờ báo lệnh màn hình có sẵn sàng hiển thị hay không. Nếu có giá trị bằng 1, nghĩa là mã ASCII hiển thị lên bàn phím. Giá trị được xóa sau khi hiển thị.
    - Load giá trị của KEY_CODE và KEY_READY vào $k0 và $k1 tương ứng
    - Load giá trị của DISPLAY_CODE và DISPLAY_READY vào $s0 và $s1 tương ứng
    - Hàm WaitForKey: vòng lặp so sánh
        - Nếu $t1 = KEY_READY = 0 → vòng lặp tiếp tục
        - Nếu $t1 = KEY_READY = 1 → chuyển tới ReadKey
    - Hàm WaitForKey: vòng lặp chờ mã ASCII từ bán phím
        - Nếu KEY_READY = 0 → vòng lặp tiếp tục
        - Nếu KEY_READY = 1 → Readkey
    - Hàm ReadKey: đọc mã ASCII từ bàn phím và lưu vào $t0
    - Hàm WaitForDis: vòng lặp chờ đến khi màn hình sẵn sàng để hiện thị
        - Nếu DISPLAY_READY = 0 → vòng lặp tiếp tục
        - Nếu DISPLAY_READY = 1 → Encrypt
    - Hàm Encrypt: Thực hiện mã hóa đơn giản mã ASCII, +1 giá trị vào mã ASCII rồi mã hóa
    - Hàm ShowKey: ghi mã ASCII được mã hóa thanh ghi DISPLAY_CODE → hiển thị lên màn hình
3. **Kết quả**


Keyboard and Display MMIO Simulator, Version 1.4
Keyboard and Display MMIO Simulator
DISPLAY: Store to Transmitter Data 0xffff000c, cursor 10, area 95 x 10
btegstbdet
Font
DAD
Fixed transmitter delay, select using slider
Delay length: 5 instruction executions
KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004
asdfrsacds
Tool Control
Disconnect from MIPS
Reset
Help
Close