



Instituto Politécnico Nacional



Escuela Superior de Cómputo

POV

TAREA 6

Materia:

Introducción a los microcontroladores

Grupo:

3CM16

Profesor:

Pérez Pérez José Juan

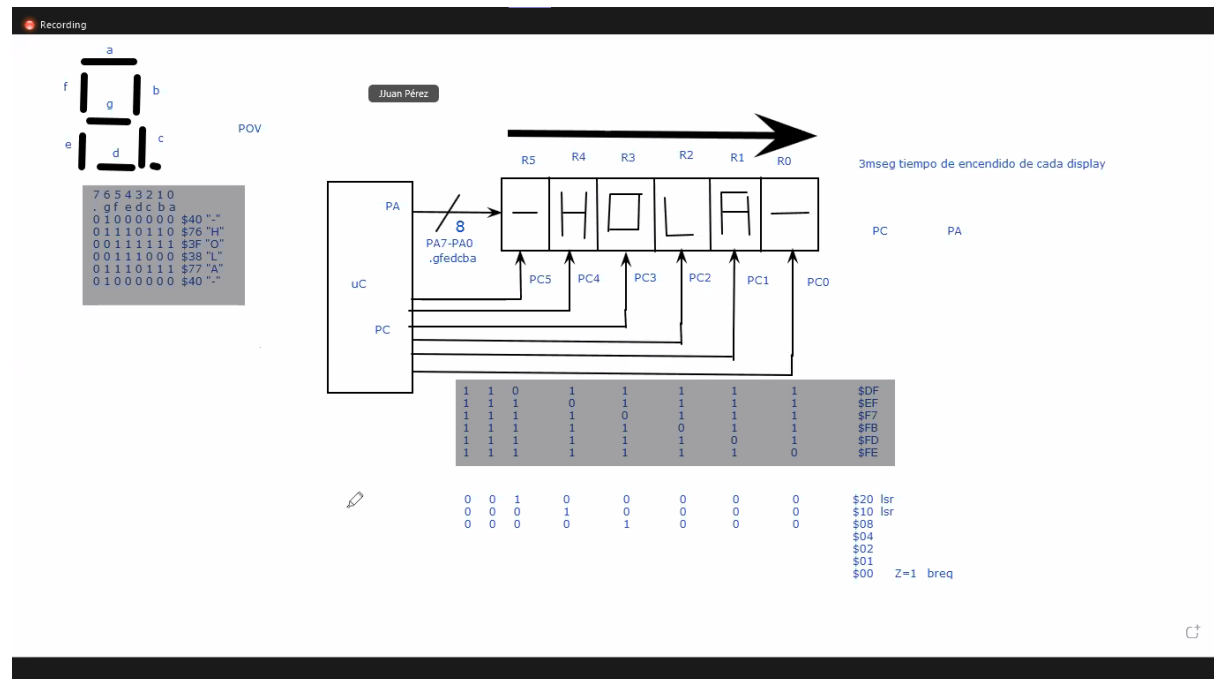
Integrantes:

Castro Cruces Jorge Eduardo

Fecha:

jueves, 21 de octubre de 2021

Realizar un programa en ensamblador para el ATmega 8535, para mostrar un mensaje en un conjunto de displays de 7 segmentos multiplexados.



Código del programa

```
1.      .include "m8535def.inc"
2.      .def aux = r16
3.      .def dat = r17
4.      .def dsp = r18
5. reset:
6.      rjmp main
7.      .org $009
8.      rjmp barre
9. main:
10.     ldi aux, low(RAMEND)
11.     out spl, aux
12.     ldi aux, high(RAMEND)
13.     out sph, aux
14.     rcall config_io
15.     ldi dsp, 1; 0000 0001
16.     ldi ZH, high(datos<<1)
17.     ldi ZL, low(datos<<1)
18. ciclo:
19.     nop
20.     nop
21.     nop
22.     rjmp ciclo
23. datos:
24.     .db $40, $77, $38, $3F, $76, $40
25. barre:
26.     out porta, zh
27.     com dsp
28.     out portc, dsp
29.     com dsp
30.     lpm dat, z+
31.     out porta, dat
32.     lsl dsp
33.     sbrc dsp, 7
34.     rjmp otro
35. sal:
36.     reti
37. otro:
38.     ldi dsp, 1; 0000 0001
39.     ldi ZH, high(datos<<1)
40.     ldi ZL, low(datos<<1)
41.     rjmp sal
42.
43. config_io:
44.     ser aux
45.     out ddra, aux
46.     out ddrc, aux
```

```
47.      ldi aux,2
48.      out tccr0,aux
49.      ldi aux,1
50.      out tmsk,aux
51.      clr zh
52.      sei
53.      ret
```

Simulación en AVR Studio 4

AVR Studio - [C:\Users\georg\Desktop\ESCOM\8vo Semestre\Micros\Tareas\POV_Display\POV_AVR\POV.asm]

File Project Build Edit View Tools Debug Window Help

Trace Disabled

Processor

Name	Value
Program Counter	0x000000
Stack Pointer	0x0000
X pointer	0x0000
Y pointer	0x0000
Z pointer	0x0000
Cycle Counter	0
Frequency	4.0000 MHz
Stop Watch	0.00 us
SREG	00000000
Registers	

```
.include "m8535def.inc"
.def aux = r16
.def dat = r17
.def dsp = r18

reset:
    rjmp main
    .org $009
    rjmp barre

main:
    ldi aux, low(RAMEND)
    out spl, aux
    ldi aux, high(RAMEND)
    out sph, aux
    rcall config_io
    ldi dsp, 1; 0000 0001
    ldi ZH, high(datos<<1)
    ldi ZL, low(datos<<1)

ciclo:
    nop
    nop
    nop
    rjmp ciclo

datos:
    .db $40, $77, $38, $3F, $76, $40

barre:
    out porta, zh
    com dsp
    out portc, dsp
```

I/O View

ANALOG_COMPARATOR

Name	Value
AD_CONVERTER	
ANALOG_COMPARA...	
CPU	
EEPROM	
EXTERNAL_INTERRUPT	
PORTA	
PORTB	
PORTC	
PORTD	
SPI	
TIMER_COUNTER_0	
TIMER_COUNTER_1	
TIMER_COUNTER_2	
TWI	
USART	

Build

ATmega8535 memory use summary [bytes]:

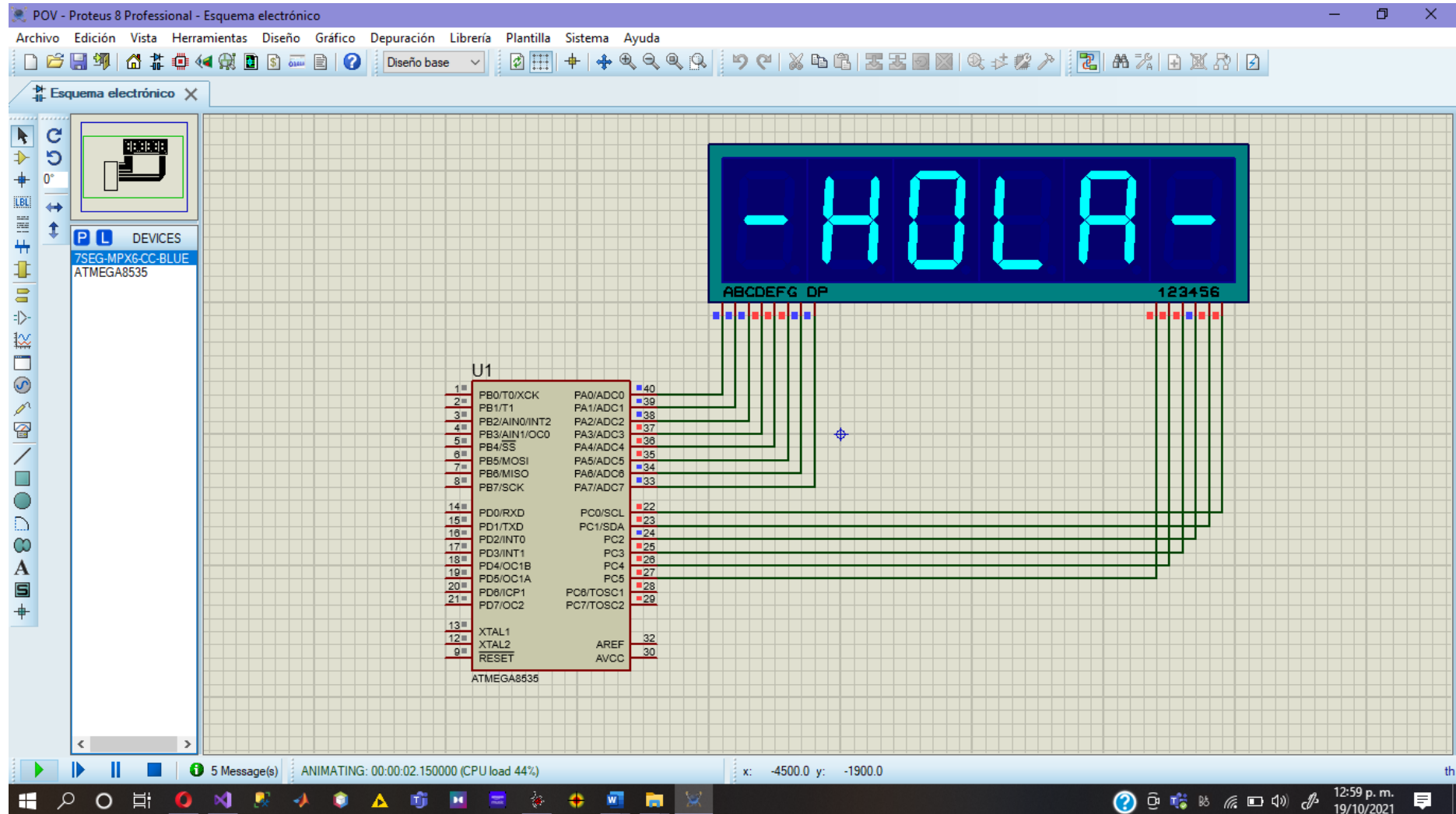
Segment	Begin	End	Code	Data	Used	Size	Use%
[.cseg]	0x000000	0x000062	76	6	82	8192	1.0%
[.dseg]	0x000060	0x000060	0	0	0	512	0.0%
[.eseg]	0x000000	0x000000	0	0	0	512	0.0%

Build Message Find in Files Breakpoints and Tracepoints

ATmega8535 AVR Simulator Auto Stopped Ln 6, Col 1 CAP NUM OVR

12:57 p. m. 19/10/2021

Simulación en Proteus 8 Professional



Conclusiones

- **Castro Cruces Jorge Eduardo**

En esta práctica la parte más complicada no fue la simulación de este, si no, la codificación. Y es que no sabía por donde empezar. Por tanto, tuve que revisar las clases pasadas para poder ayudarme a entender mejor como hacer esta tarea.

La parte de conversión de binario a hexadecimal fue sencilla, y es que tengo cierta experiencia manejando displays de 7 segmentos, ya que el semestre anterior curse la materia de Arquitectura de computadoras con el profesor Gelacio, y eso me ayudó en parte a lograr esta tarea.g