



Instituto Politécnico Nacional



Escuela Superior de Cómputo

Decodificador Hexadecimal

TAREA 3

Materia:

Introducción a los microcontroladores

Grupo:

3CM16

Profesor:

Pérez Pérez José Juan

Integrantes:

Castro Cruces Jorge Eduardo

Cortes Ramírez Roberto Carlos

Domínguez Acosta José Praxedes

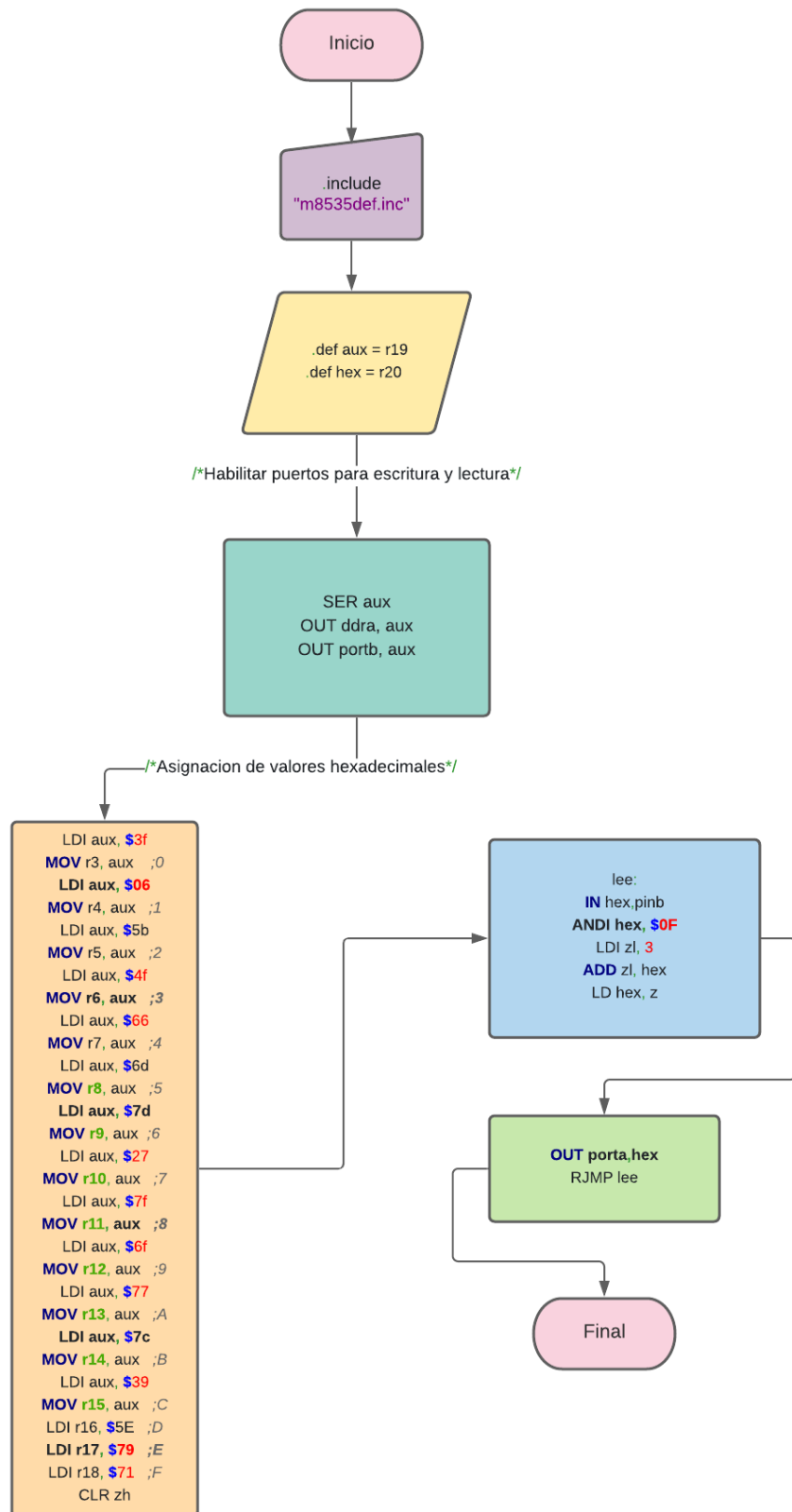
Fecha:

domingo, 10 de octubre de 2021

Descripción del problema

Descripcion: Desarrollar un programa que muestre el valor hexadecimal, en el puerto A, del valor ingresado en el puerto B.

Diagrama de flujo del programa



Código del programa

```
1. .include "m8535def.inc"
2.     .def aux = r19
3.     .def hex = r20
4.     SER aux
5.     OUT ddra, aux
6.     OUT portb, aux
7.
8.     LDI aux, $3f
9.     MOV r3, aux      ;0
10.    LDI aux, $06
11.    MOV r4, aux      ;1
12.    LDI aux, $5b
13.    MOV r5, aux      ;2
14.    LDI aux, $4f
15.    MOV r6, aux      ;3
16.    LDI aux, $66
17.    MOV r7, aux      ;4
18.    LDI aux, $6d
19.    MOV r8, aux      ;5
20.    LDI aux, $7d
21.    MOV r9, aux      ;6
22.    LDI aux, $27
23.    MOV r10, aux     ;7
24.    LDI aux, $7f
25.    MOV r11, aux     ;8
26.    LDI aux, $6f
27.    MOV r12, aux     ;9
28.    LDI aux, $77
29.    MOV r13, aux     ;A
30.    LDI aux, $7c
31.    MOV r14, aux     ;B
32.    LDI aux, $39
33.    MOV r15, aux     ;C
34.    LDI r16, $5E     ;D
35.    LDI r17, $79     ;E
36.    LDI r18, $71     ;F
37.    CLR zh
38.    lee:
39.    IN hex, pinb
40.    ANDI hex, $0F
41.    LDI z1, 3
42.    ADD z1, hex
43.    LD hex, z
44.    OUT porta, hex
45.    RJMP lee
```

Simulación en AVR Studio 4

AVR Studio - [C:\Users\georg\Desktop\ESCOM\8vo Semestre\Micros\Tareas\HEX\HEX.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 = r19
.def hex = r20
SER aux
OUT ddra, aux
OUT portb, aux

LDI aux, $3f
MOV r3, aux ;0
LDI aux, $06
MOV r4, aux ;1
LDI aux, $5b
MOV r5, aux ;2
LDI aux, $4f
MOV r6, aux ;3
LDI aux, $66
MOV r7, aux ;4
LDI aux, $6d
MOV r8, aux ;5
LDI aux, $7d
MOV r9, aux ;6
LDI aux, $27
MOV r10, aux ;7
LDI aux, $7f
MOV r11, aux ;8
LDI aux, $6f
MOV r12, aux ;9
LDI aux, $77
```

I/O View

ANALOG_COMPARATOR

Name	Value
AD_CONVERTER	
ANALOG_COMPARATOR	
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]:

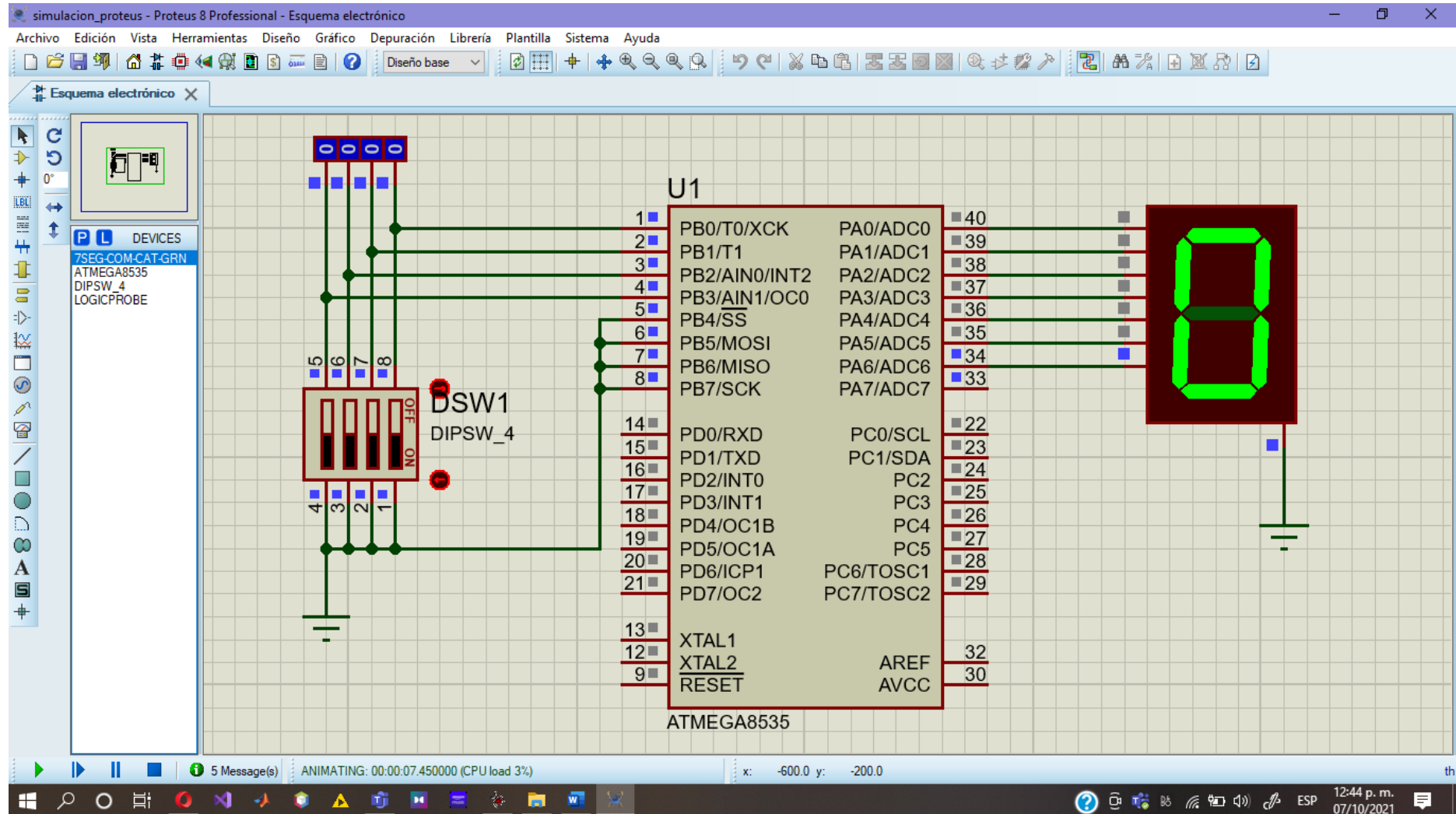
Segment	Begin	End	Code	Data	Used	Size	Use%
[.cseg]	0x000000	0x000050	80	0	80	8192	1.0%
[.dseg]	0x000060	0x000060	0	0	0	512	0.0%
[.eseg]	0x000000	0x000000	0	0	0	512	0.0%

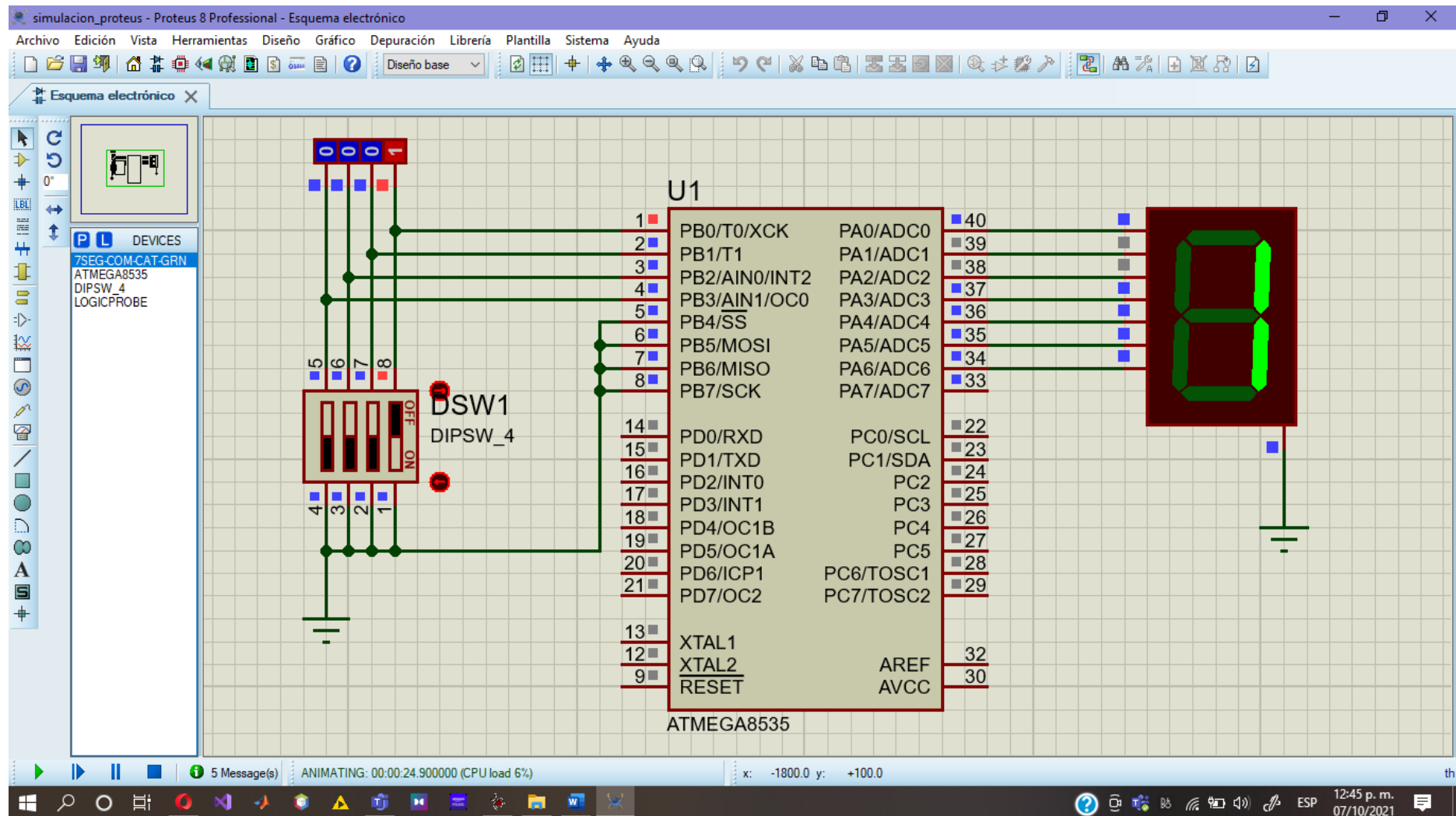
Build Message Find in Files Breakpoints and Tracepoints

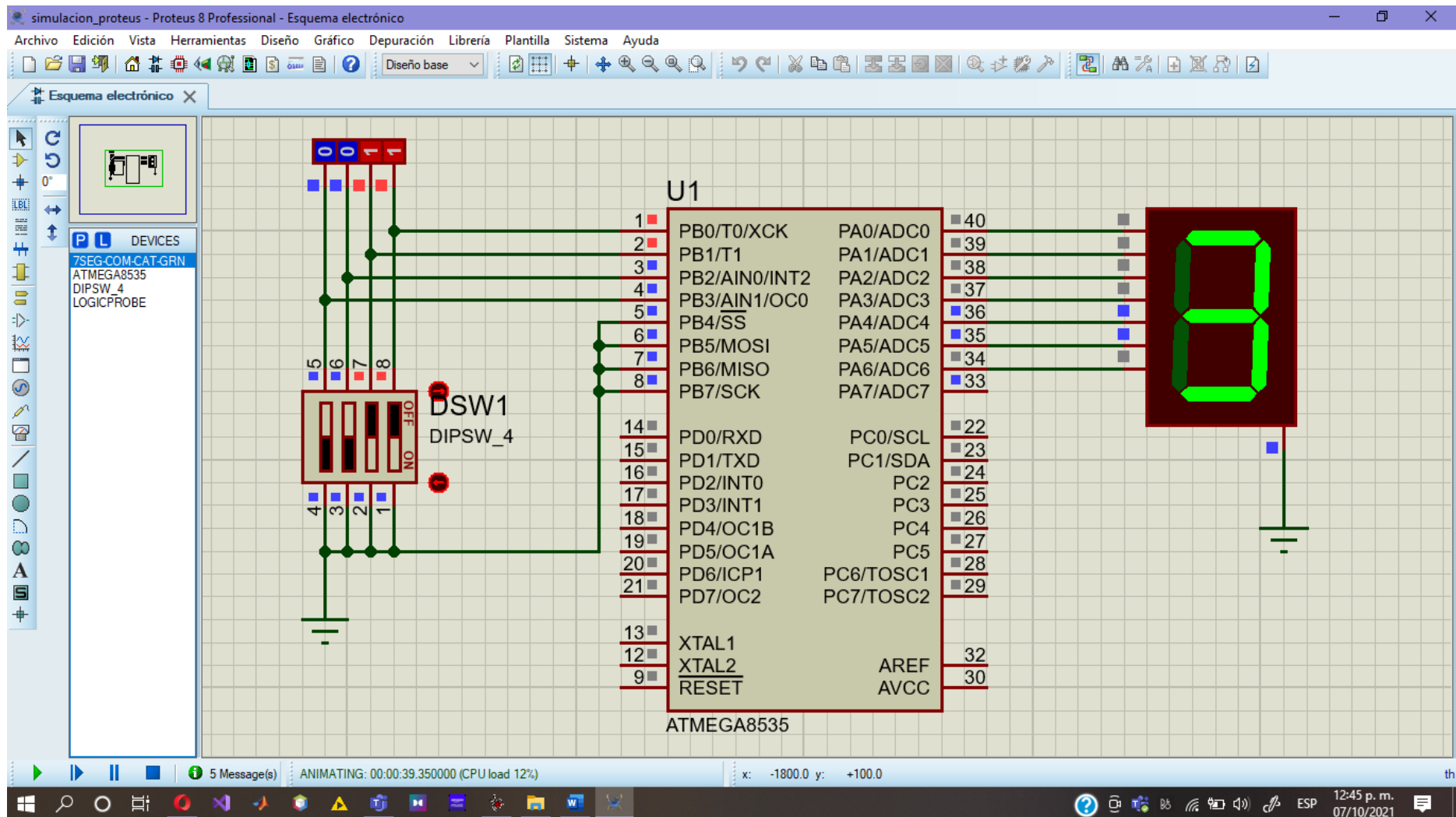
ATmega8535 AVR Simulator Auto Stopped Ln 4, Col 1 CAP NUM OVR

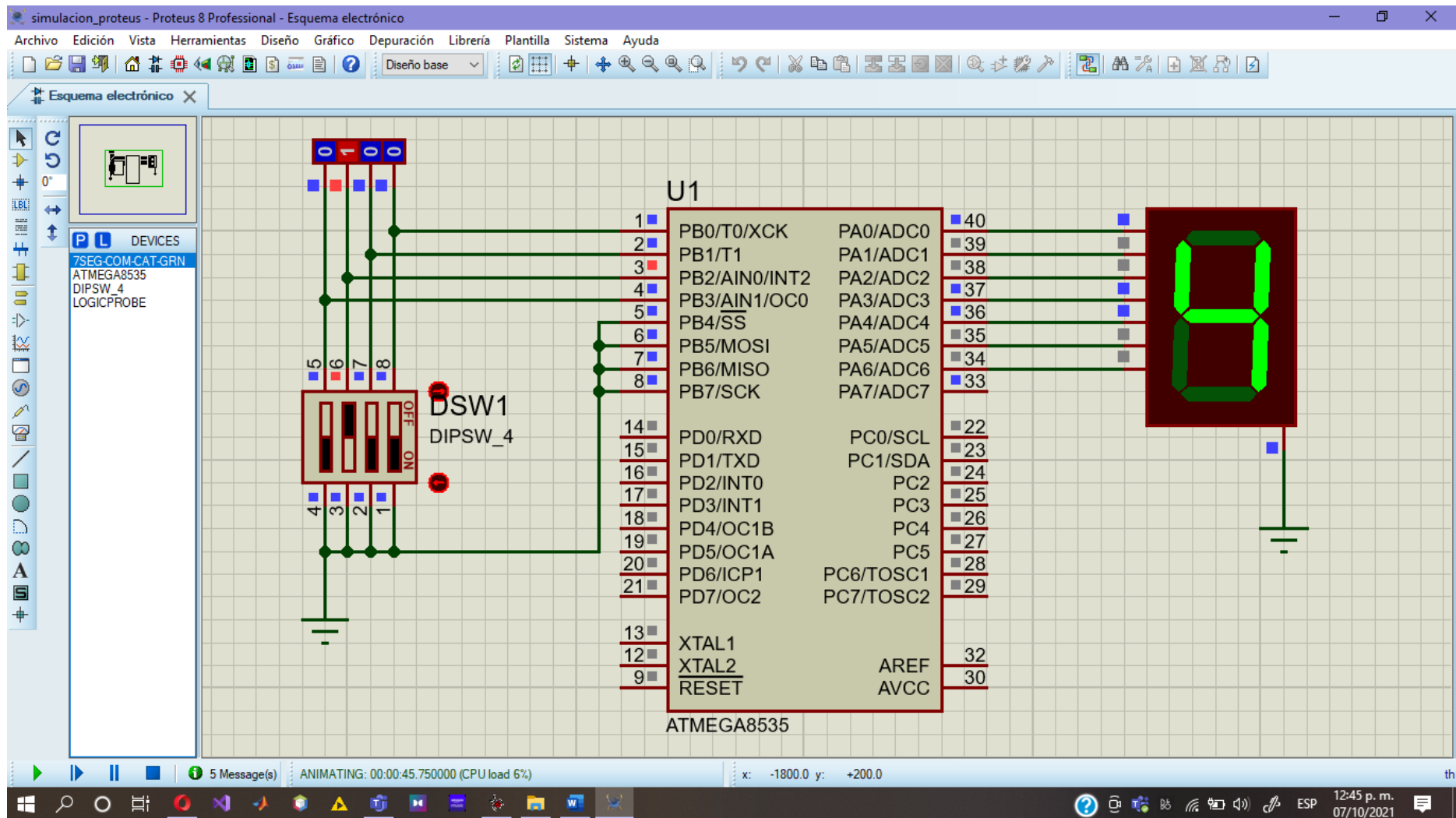
12:41 p. m. 07/10/2021

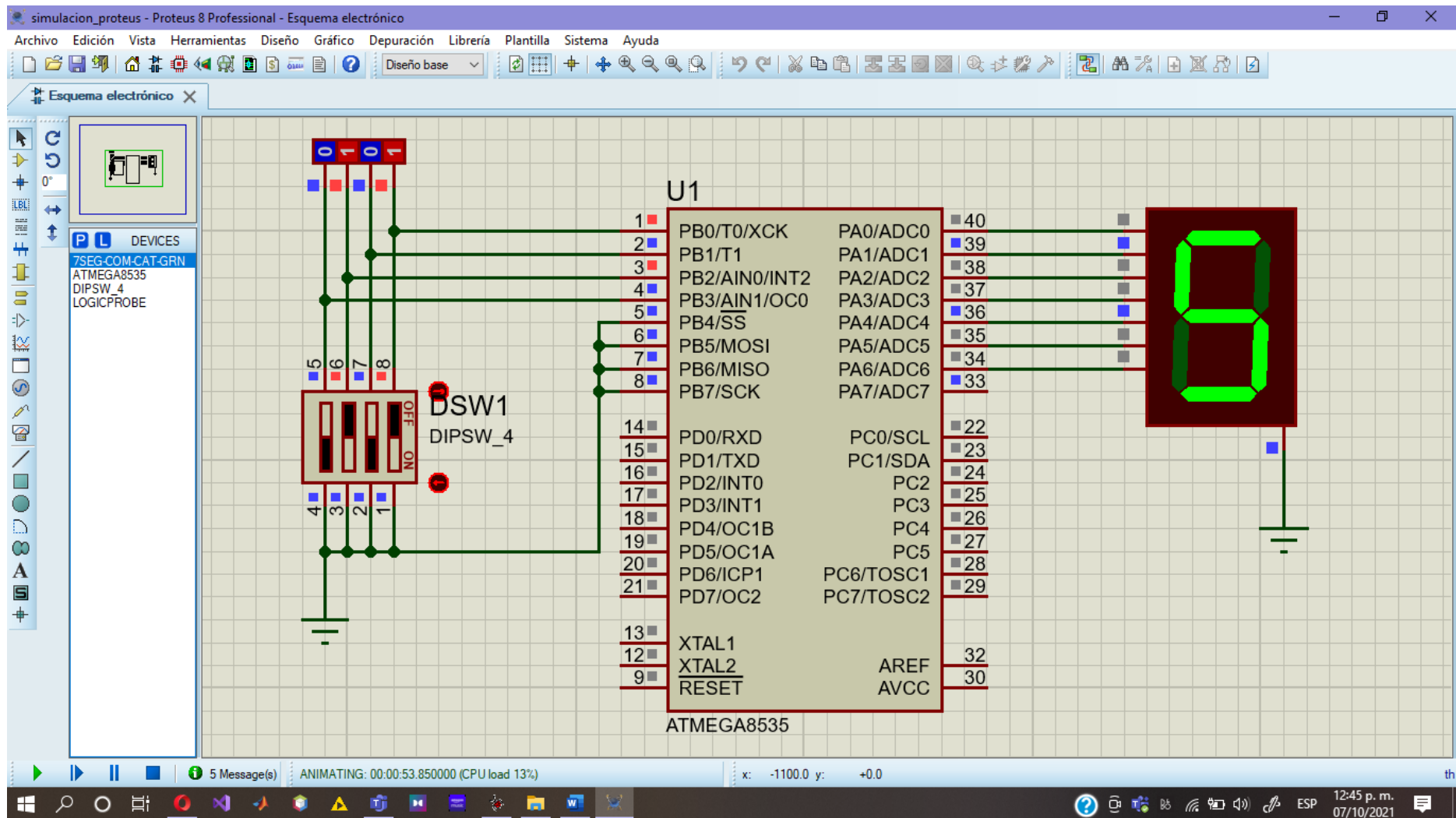
Simulación en Proteus 8 Professional

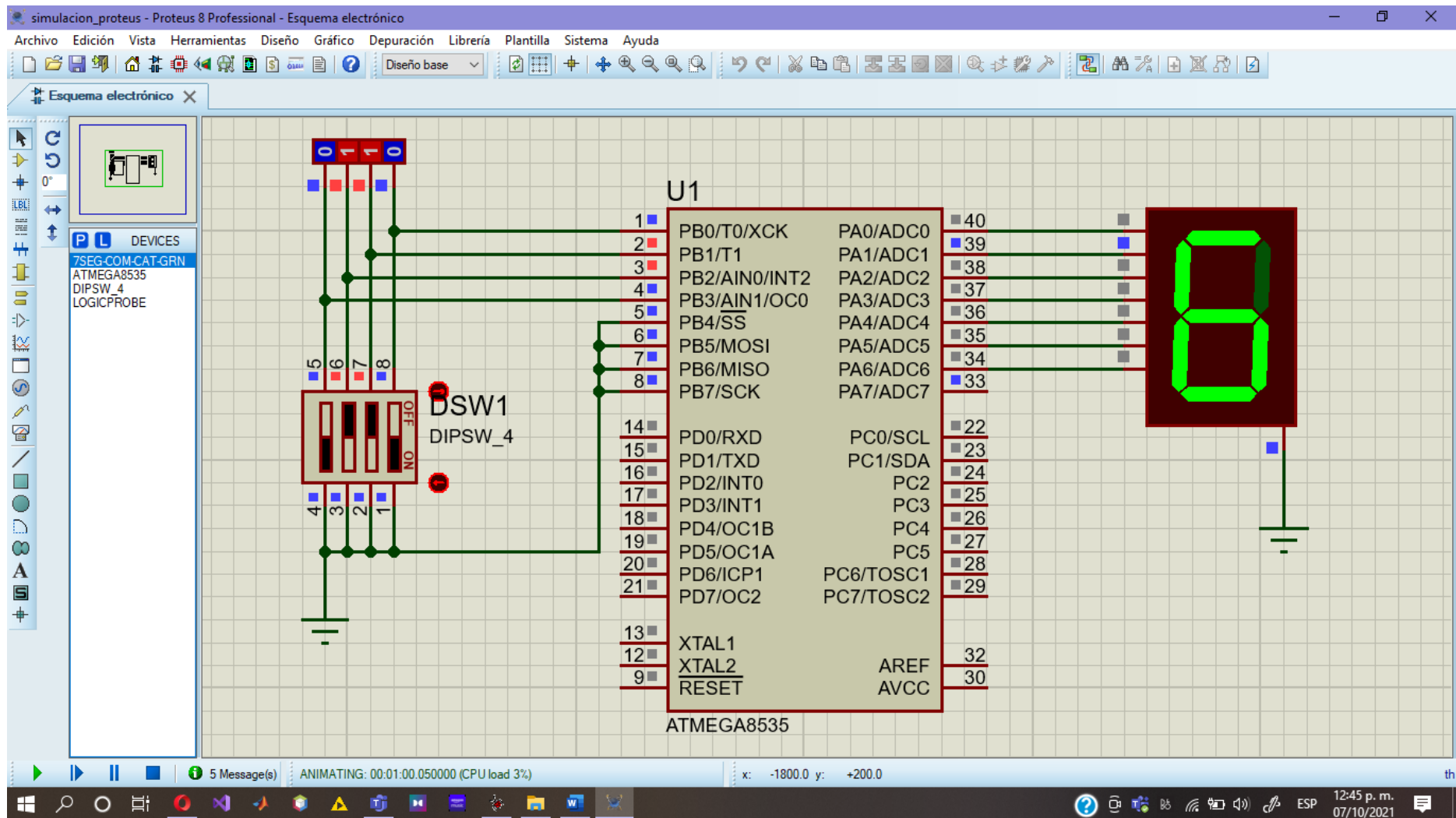


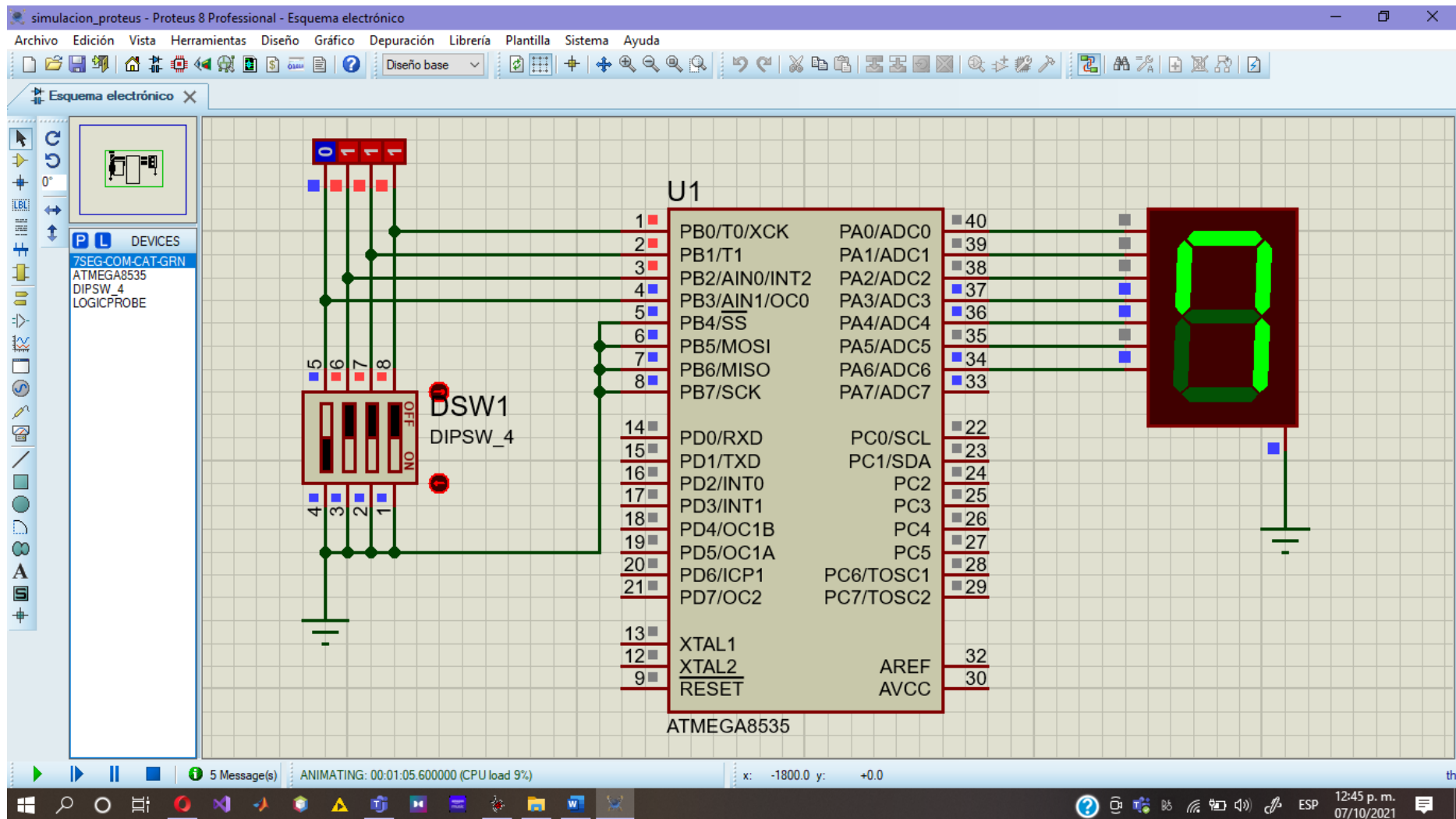


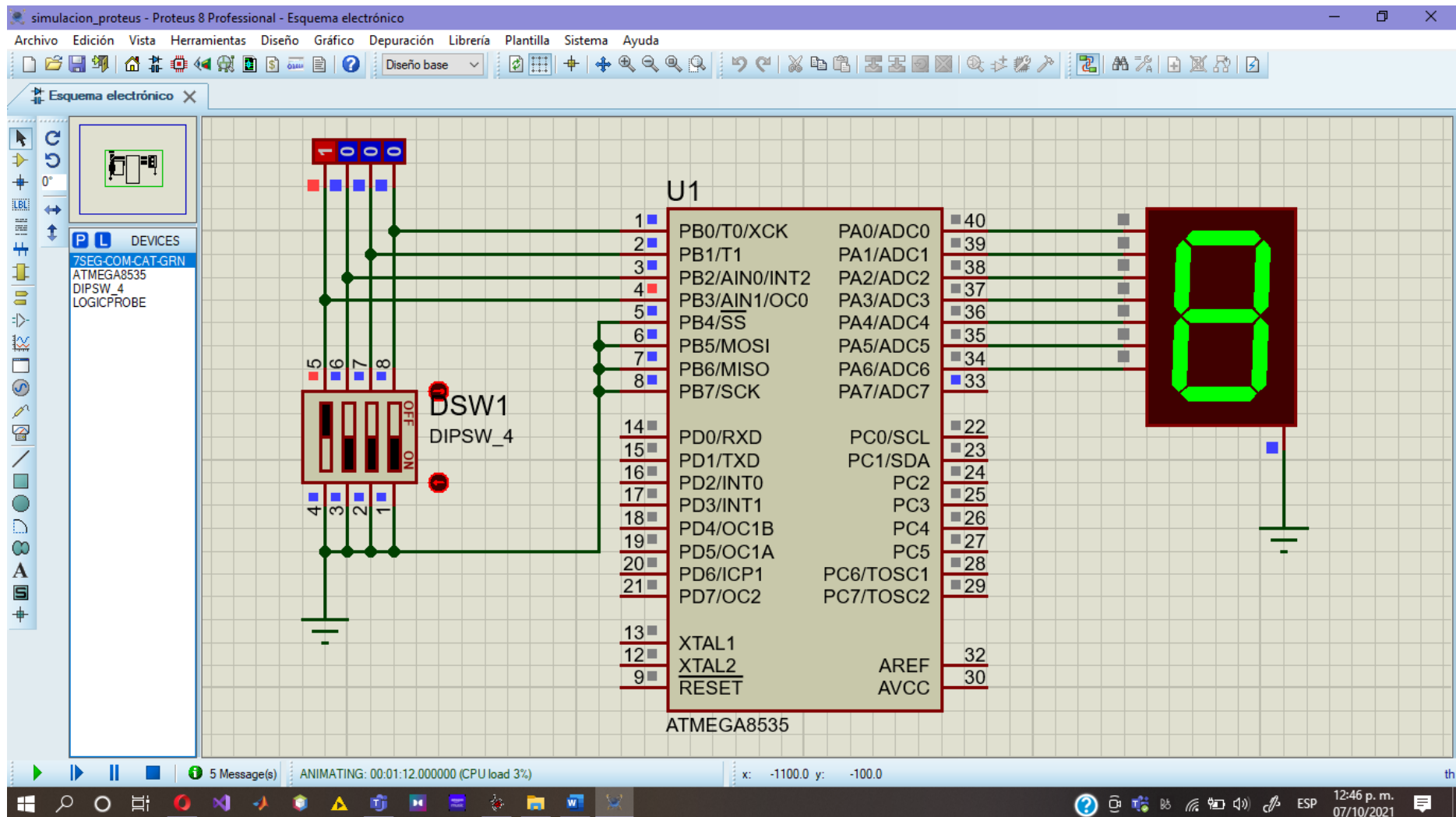


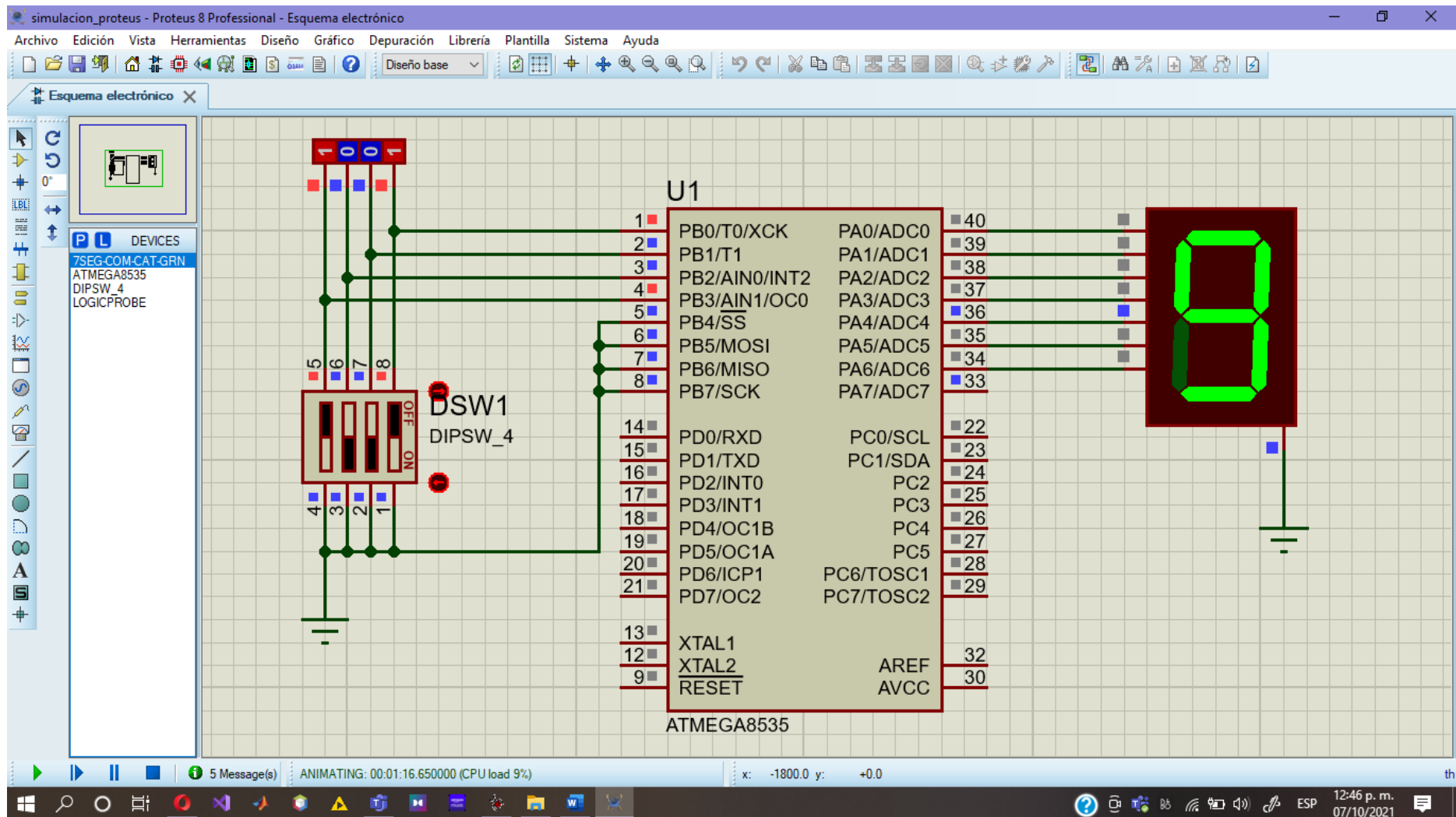


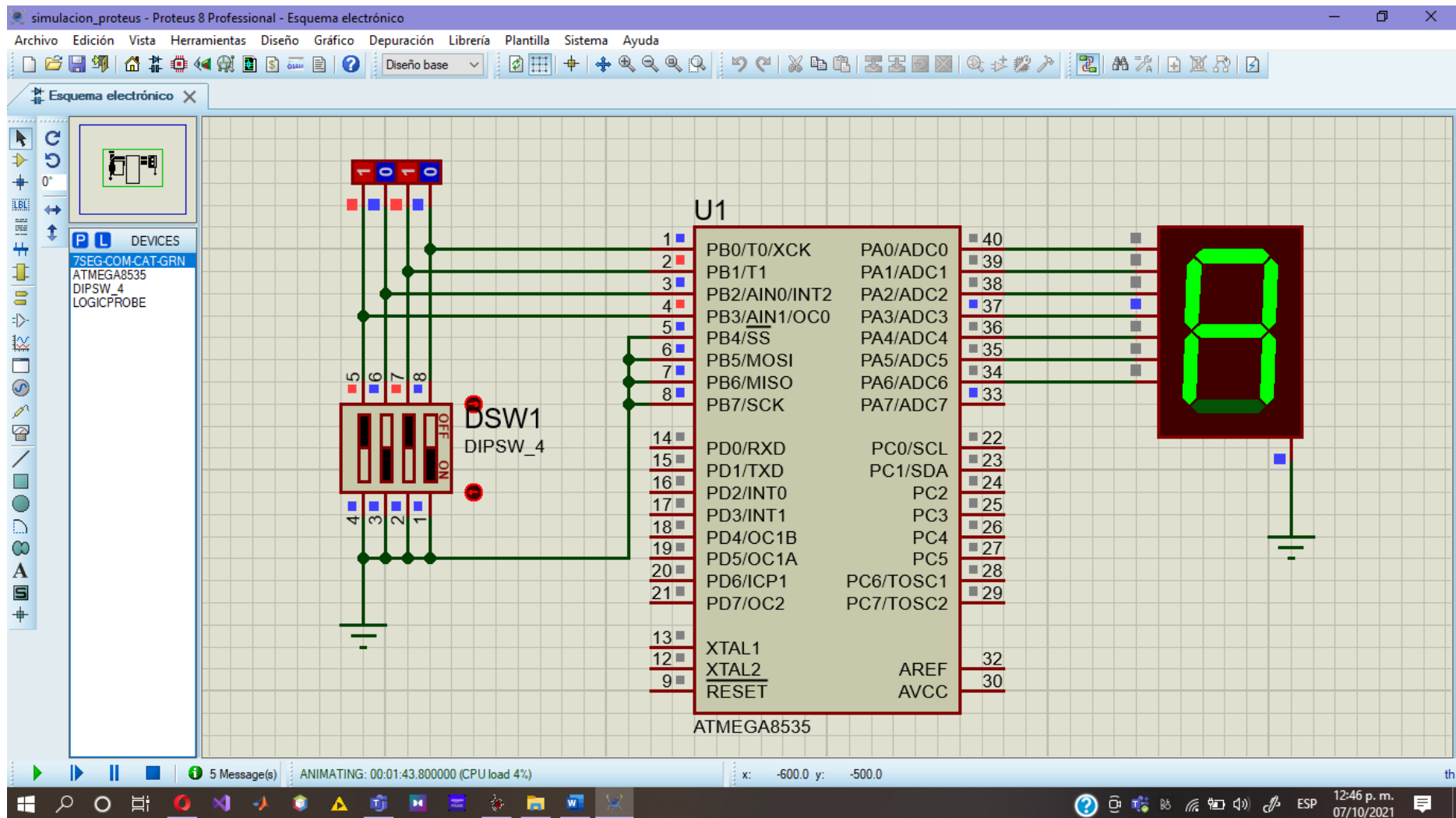


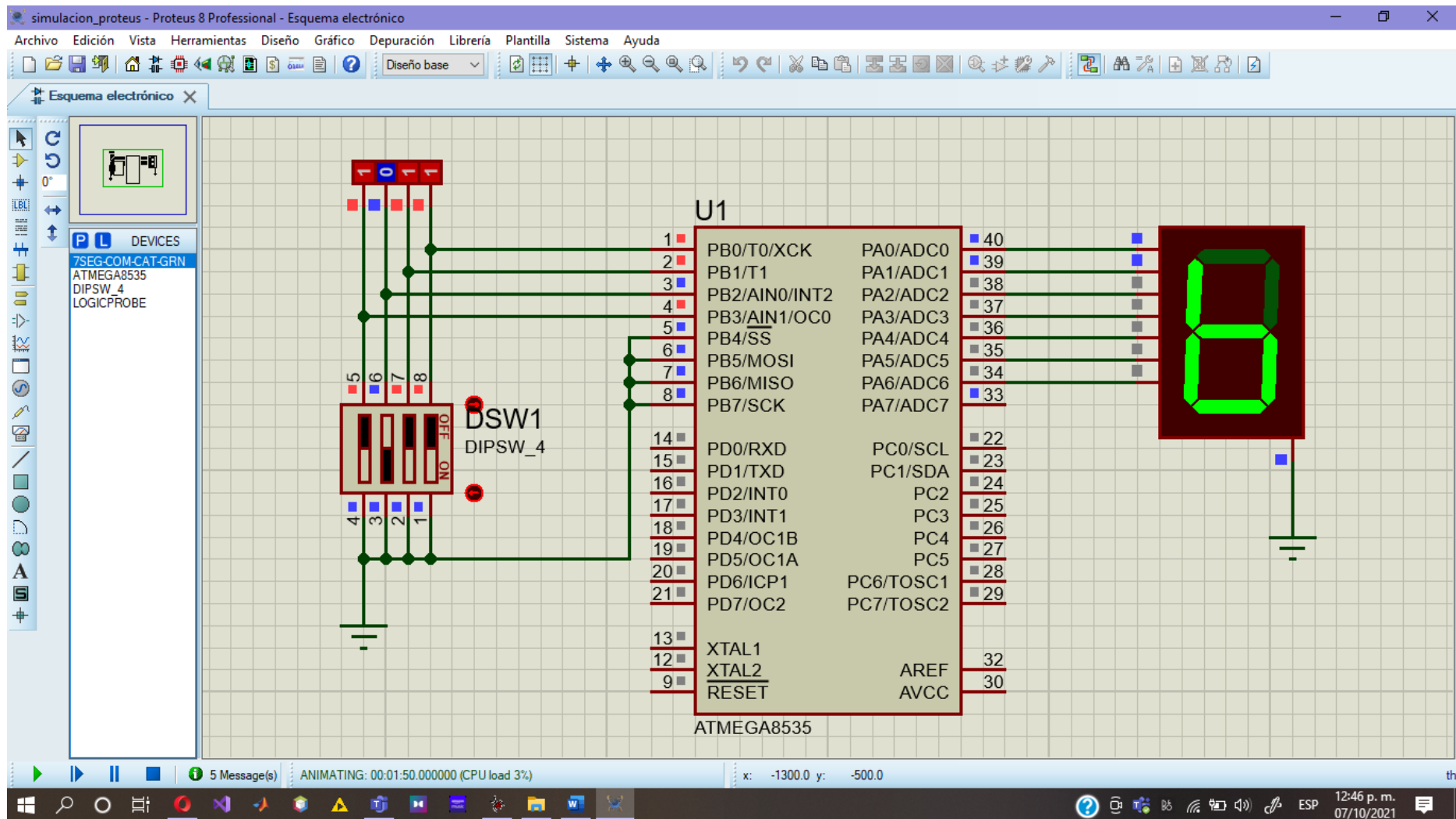


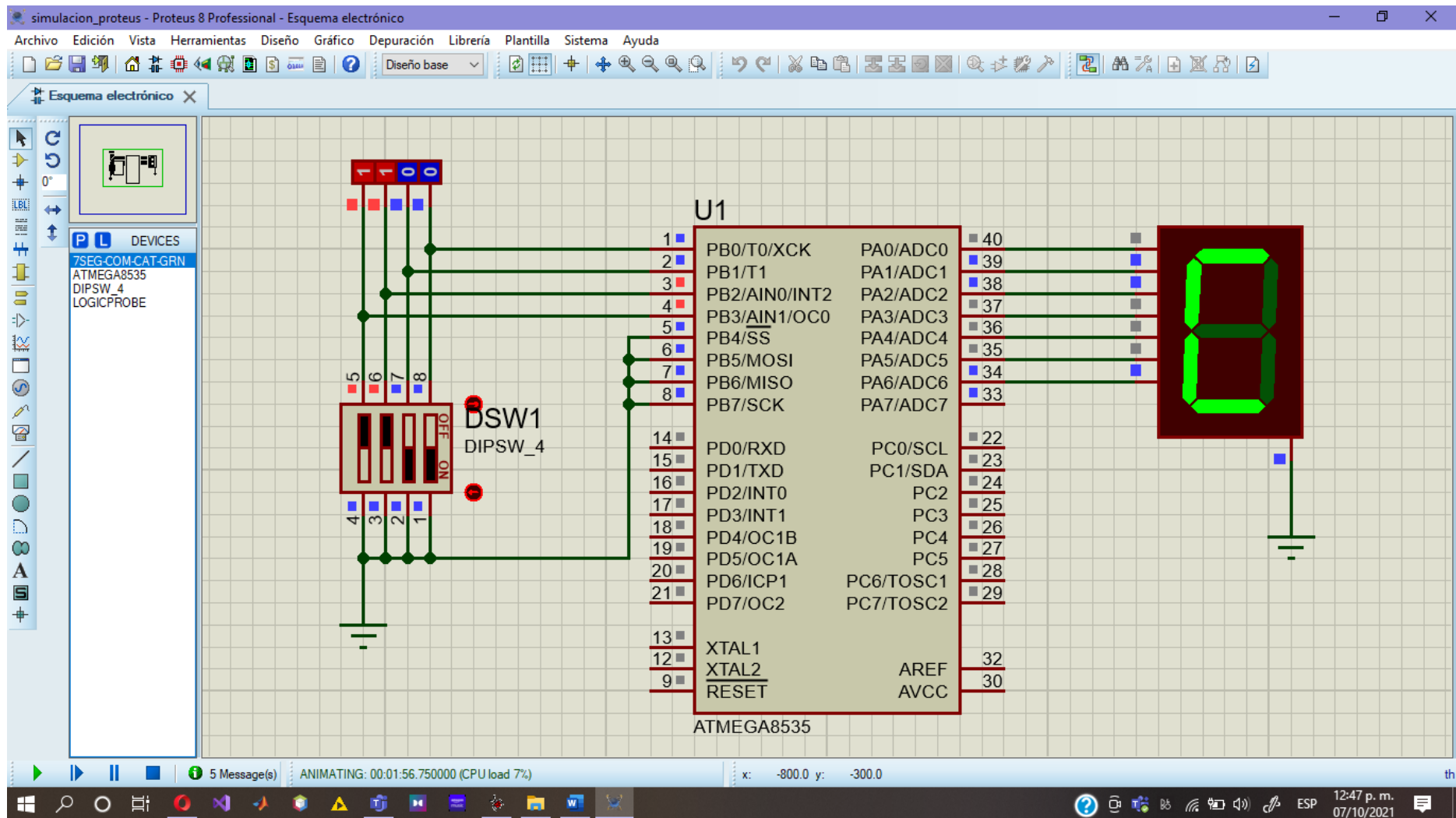


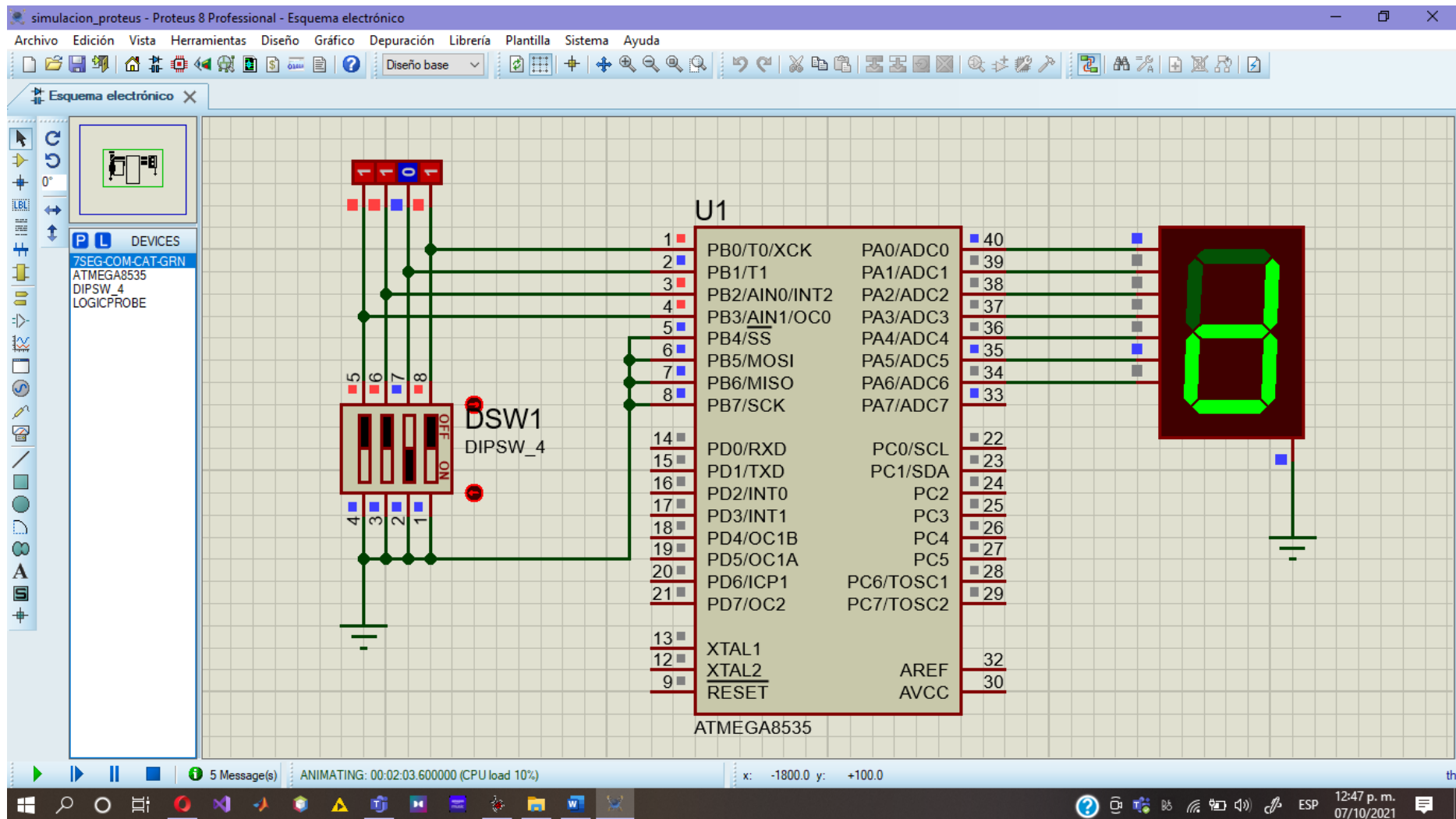


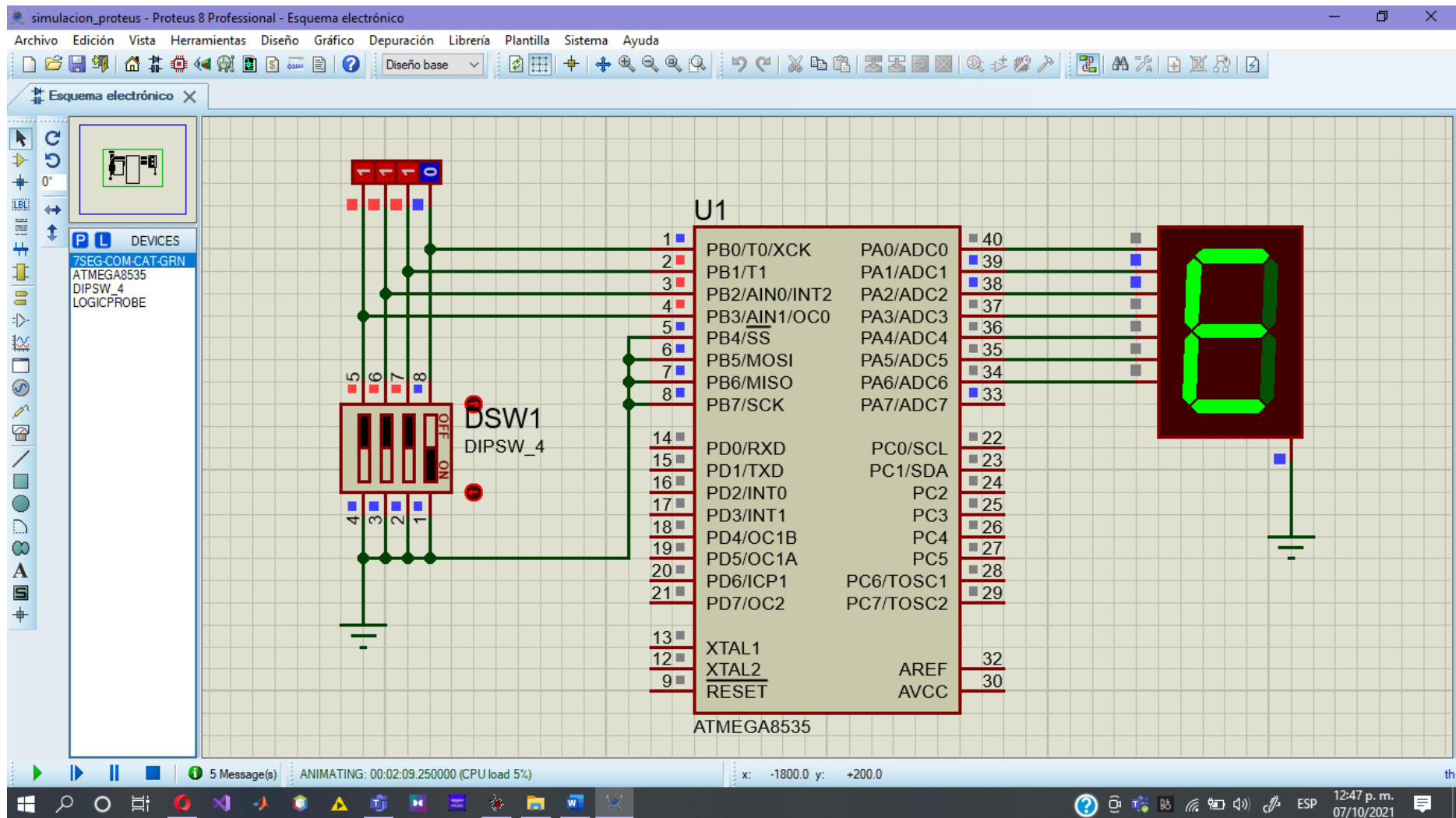


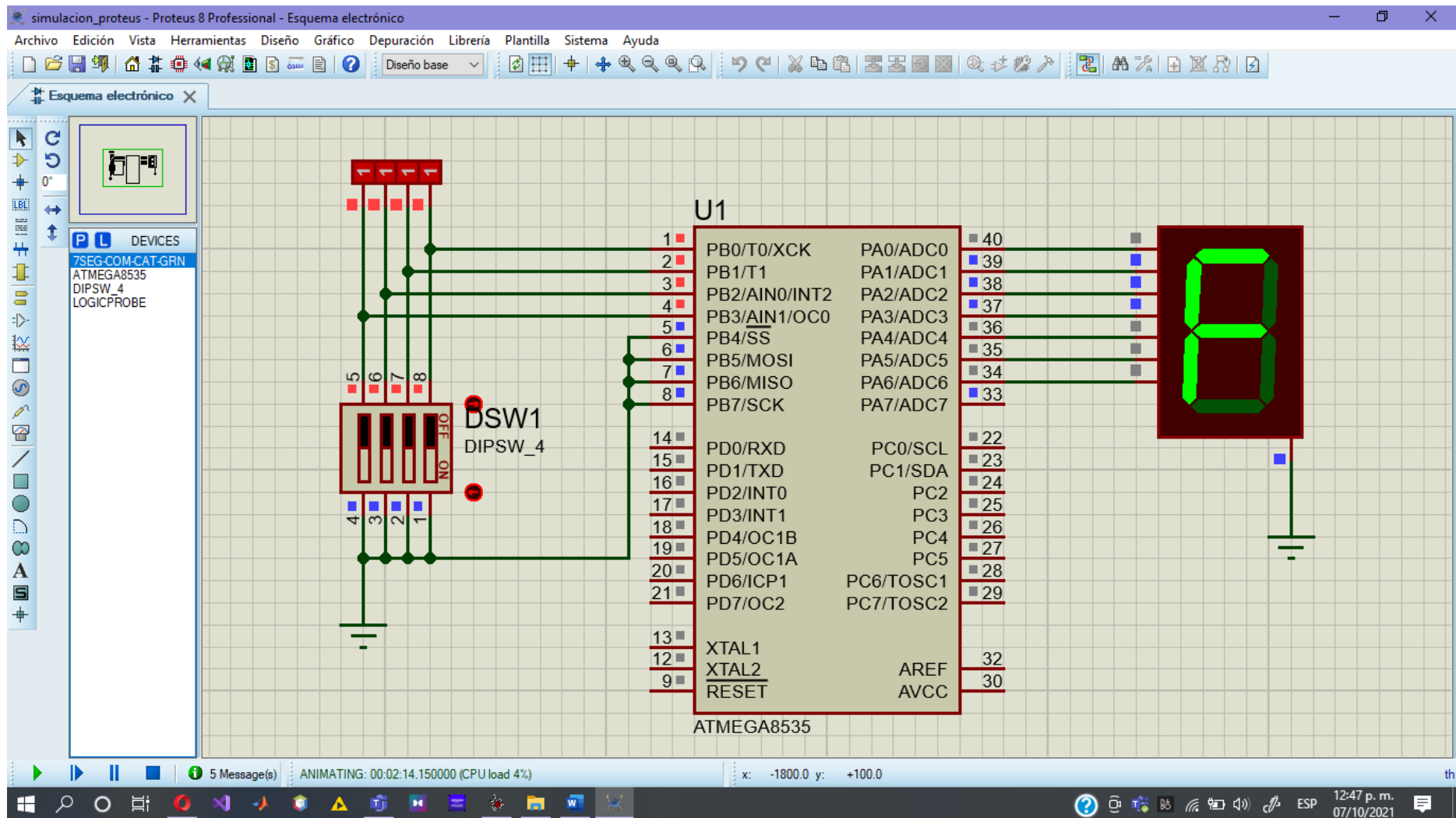












Conclusiones

- **Castro Cruces Jorge Eduardo**

Esta práctica se logró el objetivo principal, que fue desarrollar un programa que muestre a la salida A el valor hexadecimal dependiendo del valor en binario ingresado a la entrada B. La simulación fue relativamente sencilla, ya que en mi materia de Instrumentación, el profesor nos pedía realizar simulaciones en el programa Proteus 8 Professional.

- **Cortes Ramírez Roberto Carlos**

En esta práctica se pudo realizar un programa en ensamblador que consistía en que muestre en la salida A el valor hexadecimal dependiendo del valor en binario ingresado a la entrada B, la cual se logró sin ningún problema pero primero empezamos haciendo el diagrama de flujo, para poder saber cómo iba a funcionar el programa, también después se realizó una simulación con la herramienta Proteus y esto fue sencillo ya que con la experiencia de la carrera hemos logrado manejar mejor la herramienta.

- **Domínguez Acosta José Práxedes**

Desarrollando el código pudimos generar la impresión de los primeros valores hexadecimales (en los puertos de A) obtenidos de la inserción de valores binarios (en los puertos B), demostrando lo anterior en una simulación de Proteus 8 Professional donde se pueden apreciar que las 16 combinaciones de bits en la entrada generan su propio valor en hexadecimal en el display de 7 segmentos.