INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

“Manejo de Servomotores con PWM”

Integrantes del Equipo:

Contreras Cardoso Adolfo

Martínez Alvarado Bryan Alexis

Maya Martínez Alonso Rubén

Pérez Gómez Santiago

Grupo:

3CM15

Profesor:

Ing. José Juan Pérez Pérez

Asignatura:

Introducción a los Microcontroladores

Planteamiento del Problema

Reescriba el programa utilizado para manejar un servomotor por medio de una señal PWM, de manera que ahora se pueda mover hasta en ocho posibles valores distintos.

Desarrollo

Para realizar la implementación se utilizó como base el programa propuesto por el profesor, con el cual se consiguió dividir los 180 grados del servomotor en 8 posibles valores, para esto se utilizó una diferencia de tiempo de 0.1428 milisegundos, en otras palabras, una diferencia de 25.7 grados. Como añadido se simplificó el código modificando la subrutina para llamar 'n' veces al retardo. A continuación se muestra la obtención del retardo con la calculadora en línea, es necesario mencionar que se añadieron 4 ciclos de reloj debido a las instrucciones que se utilizan, de esta manera se consigue un retardo más exacto:

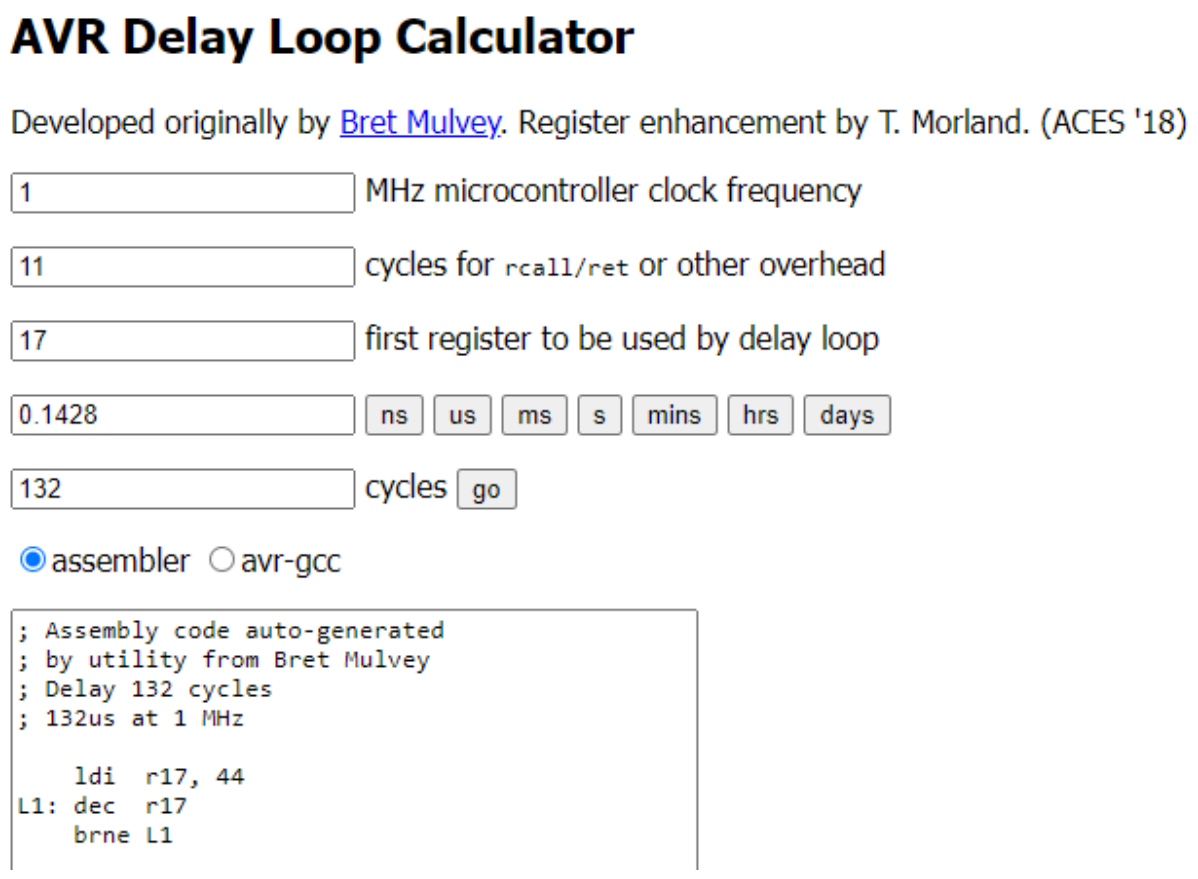

Figura 1. Cálculo del retardo de 0.1428ms

El diagrama de flujo correspondiente al resultado es el siguiente:

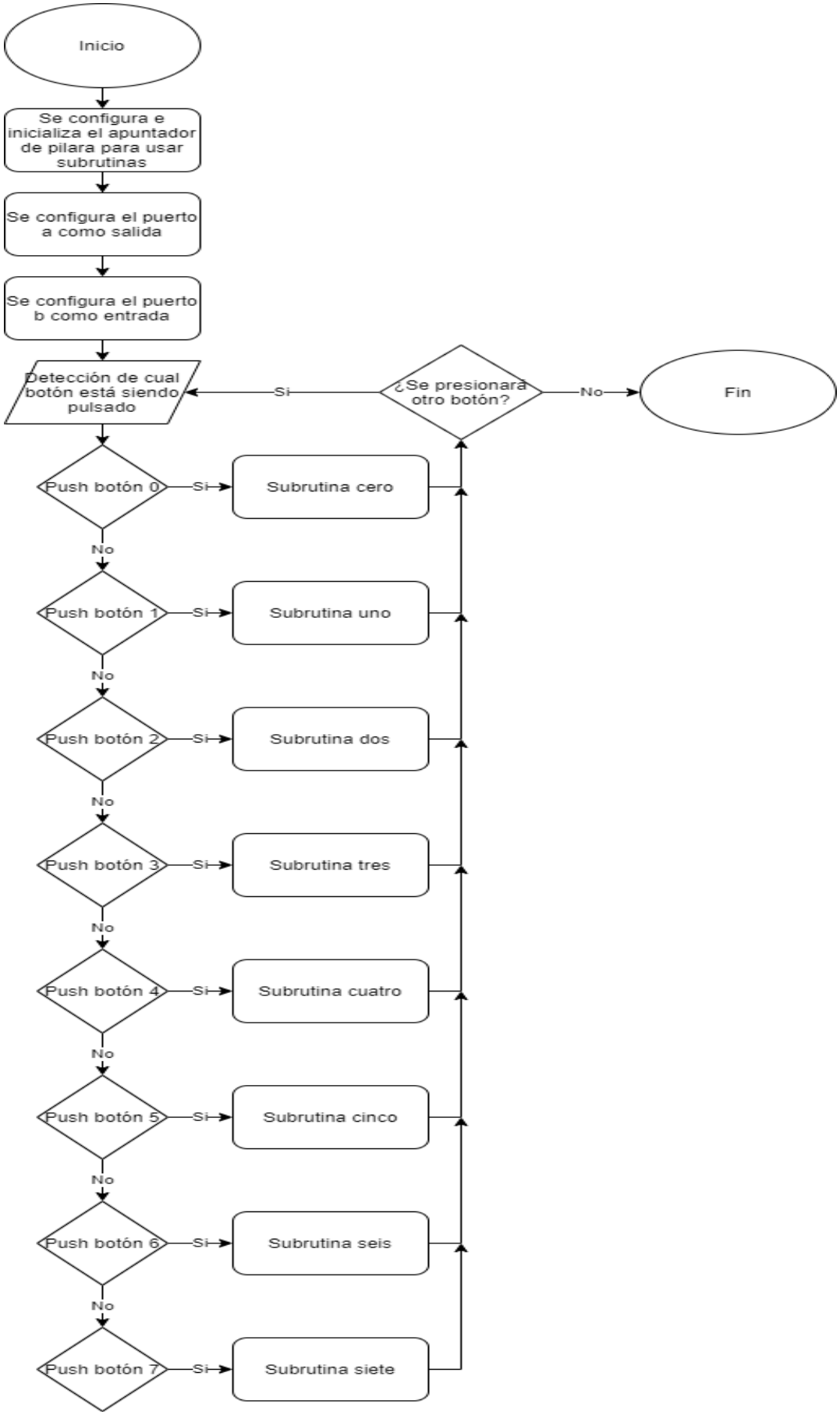

Figura 2. Diagrama de Flujo

El respectivo código en ensamblador es el siguiente:

```
;codigo para manejar un servomotor con
;una señal pwm realizada manualmente
;posee 8 posibles valores distintos

;para calcular el retardo se hace:
;1 / (# de secciones - 1)
;1 / 7 = 0.1428ms

;para calcular el tiempo de estado alto:
; 1 + (retardo * # de seccion) / retardo
;para la posicion cero:
; 1 + (0.1428 * 0) / 0.1428 = 7 veces

;para calcular el tiempo de estado bajo:
; 19 - (retardo * # de seccion) / retardo
;para la posicion cero:
; 19 - (0.1428 * 0) / 0.1428 = 133 veces

.include "m8535def.inc"
.def aux = r16

;configurando apuntador de
;pila para usar subrutinas
ldi aux,low(ramend)
out spl,aux
ldi aux,high(ramend)
```

Figura 3. Primera Sección de Código

```
      out sph,aux

      ;configurando puerto a como
      ;salida y puerto b como entrada
      ser aux
      out ddra,aux
      out portb,aux

  main:
      ;si se presiona el push cero
      ;salta a la subrutina cero
      ;(el push genera valor de cero)
      sbis pinb,0
      rcall cero

      ;si se presiona el push uno
      sbis pinb,1
      rcall uno

      ;si se presiona el push dos
      sbis pinb,2
      rcall dos

      ;si se presiona el push tres
      sbis pinb,3
      rcall tres
```

Figura 4. Segunda Sección de Código

```
    ;si se presiona el push cuatro
    sbis pinb,4
    rcall cuatro

    ;si se presiona el push cinco
    sbis pinb,5
    rcall cinco

    ;si se presiona el push seis
    sbis pinb,6
    rcall seis

    ;si se presiona el push siete salta
    ;a la subrutina ciento_ochenta
    sbis pinb,7
    rcall ciento_ochenta

    ;creando un ciclo infinito
    rjmp main

cero:
    ;colocando un valor de uno por
    ;1ms para la posicion cero grados
    sbi porta,0
    ldi aux,7
    rcall ciclarRetardos
```

Figura 5. Tercera Sección de Código

```
    ;colocando un valor de cero por
    ;19ms para la posicion cero grados
    cbi porta,0
    ldi aux,133
    rcall ciclarRetardos

    ret

uno:
    ;colocando un valor de uno por
    ;1.1428ms para la posicion uno
    sbi porta,0
    ldi aux,8
    rcall ciclarRetardos

    ;colocando un valor de cero por
    ;18.8572ms para la posicion uno
    cbi porta,0
    ldi aux,132
    rcall ciclarRetardos

    ret
```

Figura 6. Cuarta Sección de Código

```
dos:
    ;colocando un valor de uno por
    ;1.2856ms para la posicion uno
    sbi porta,0
    ldi aux,9
    rcall ciclarRetardos

    ;colocando un valor de cero por
    ;18.7144ms para la posicion uno
    cbi porta,0
    ldi aux,131
    rcall ciclarRetardos

    ret

tres:
    ;colocando un valor de uno por
    ;1.4284ms para la posicion uno
    sbi porta,0
    ldi aux,10
    rcall ciclarRetardos

    ;colocando un valor de cero por
    ;18.5716ms para la posicion uno
    cbi porta,0
    ldi aux,130
    rcall ciclarRetardos
```

Figura 7. Quinta Sección de Código

```
    ret

cuatro:
    ;colocando un valor de uno por
    ;1.5712ms para la posicion uno
    sbi porta,0
    ldi aux,11
    rcall ciclarRetardos

    ;colocando un valor de cero por
    ;18.4288ms para la posicion uno
    cbi porta,0
    ldi aux,129
    rcall ciclarRetardos

    ret

cinco:
    ;colocando un valor de uno por
    ;1.714ms para la posicion uno
    sbi porta,0
    ldi aux,12
    rcall ciclarRetardos
```

Figura 8. Sexta Sección de Código

```
    ;colocando un valor de cero por
    ;18.286ms para la posicion uno
    cbi porta,0
    ldi aux,128
    rcall ciclarRetardos

    ret

seis:
    ;colocando un valor de uno por
    ;1.8568ms para la posicion uno
    sbi porta,0
    ldi aux,13
    rcall ciclarRetardos

    ;colocando un valor de cero por
    ;18.1432ms para la posicion uno
    cbi porta,0
    ldi aux,127
    rcall ciclarRetardos

    ret
```

Figura 9. Séptima Sección de Código

```
ciento_ochenta:
    ;colocando un valor de uno por
    ;2ms para la posicion 180 grados
    sbi porta,0
    ldi aux,14
    rcall ciclarRetardos

    ;colocando un valor de cero por
    ;18ms para la posicion 180 grados
    cbi porta,0
    ldi aux,126
    rcall ciclarRetardos

    ret

ciclarRetardos:
    ;este codigo se cicla hasta que
    ;aux llegue al valor de cero
    rcall retardo
    dec aux
    brne ciclarRetardos

    ret
```

Figura 10. Octava Sección de Código

```
retardo:
    ;codigo para el retardo de 0.1428ms

    ; Assembly code auto-generated
    ; by utility from Bret Mulvey
    ; Delay 132 cycles
    ; 132us at 1 MHz

        ldi  r17, 44
    L1: dec  r17
        brne L1

    ret
```

Figura 11. Novena Sección de Código

Para la simulación, se muestran la primera, tercera, quinta y última posición, así como el osciloscopio en la última posición:

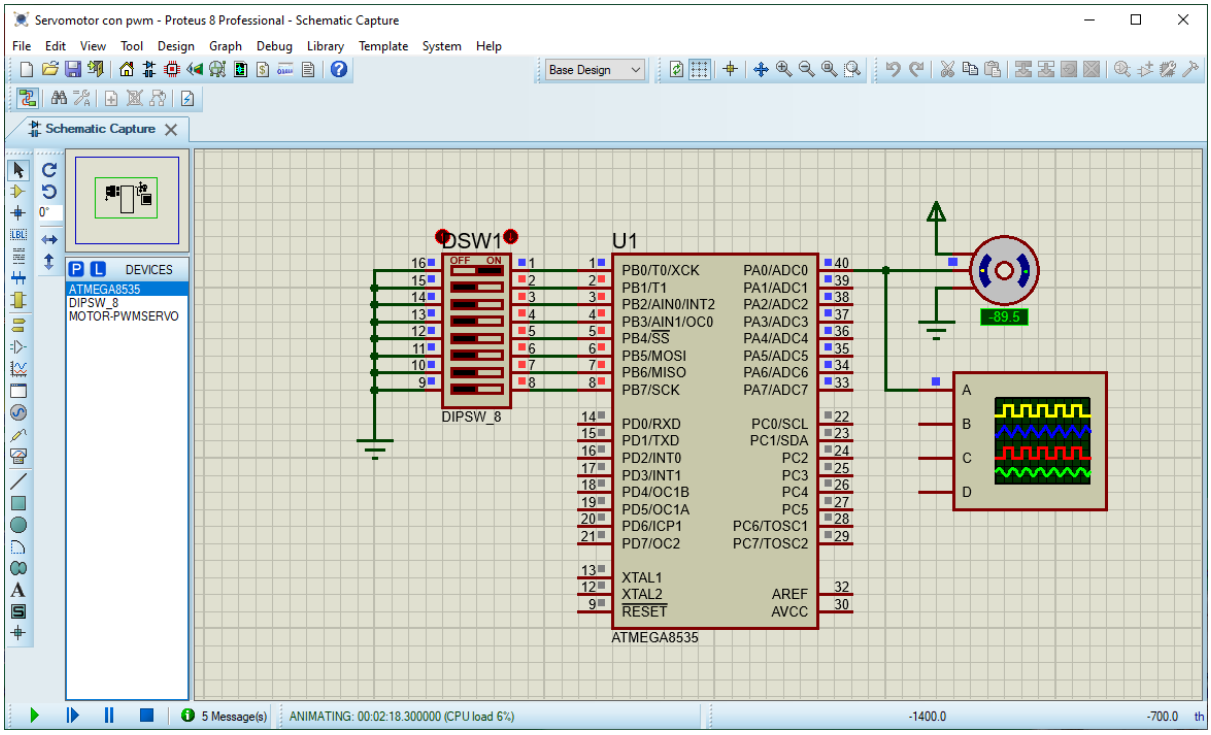

Figura 12. Primera Posición (0 grados)

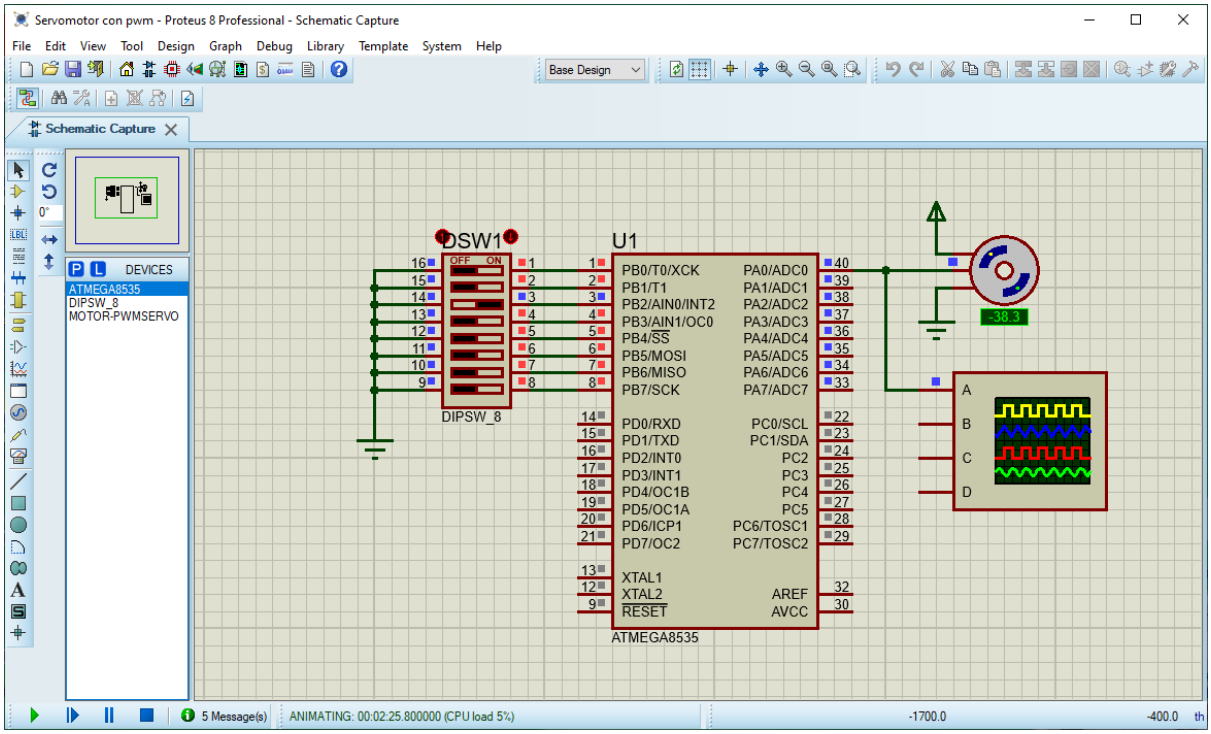

Figura 13. Tercera Posición (77.1 grados)

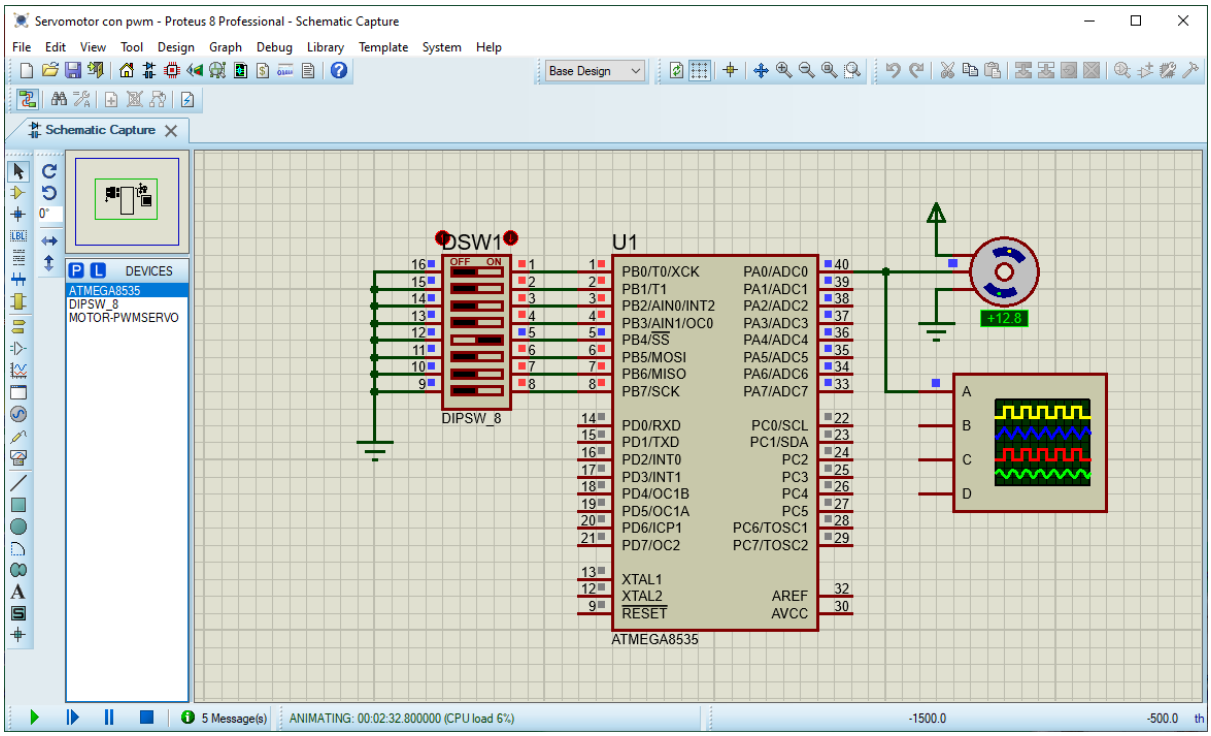

Figura 14. Quinta Posición (128.5 grados)

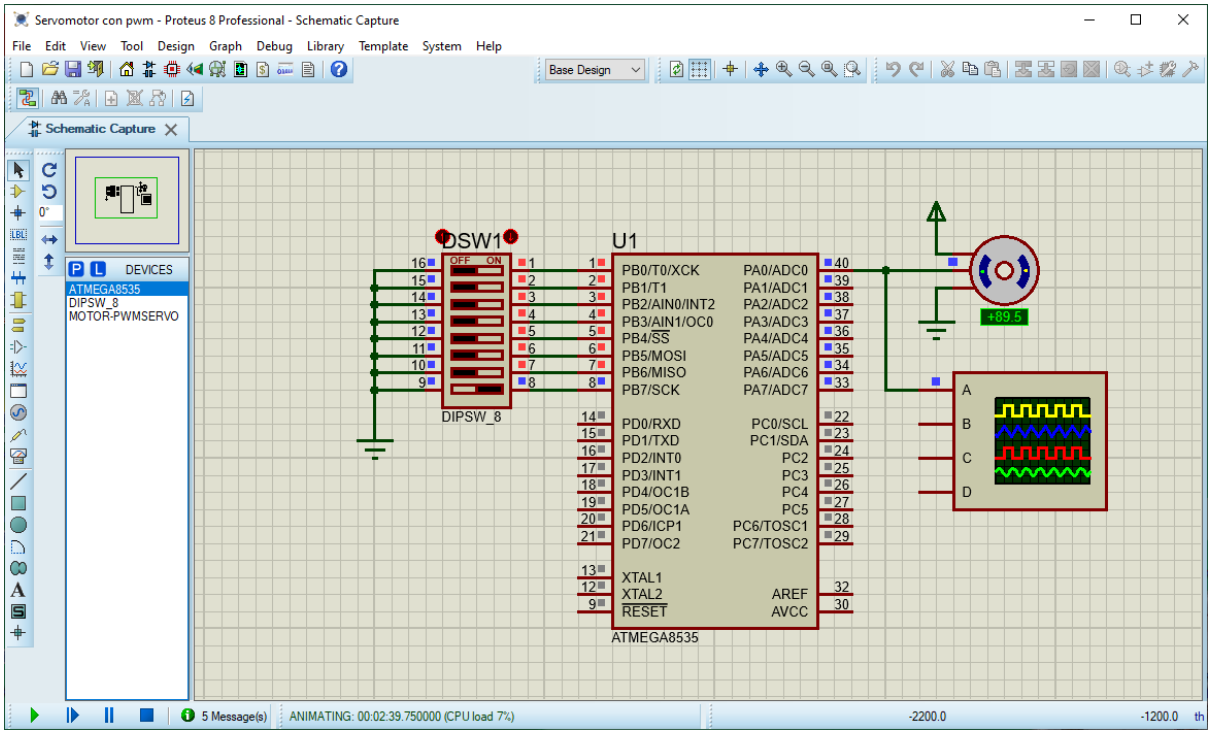

Figura 15. Última Posición (180 grados)

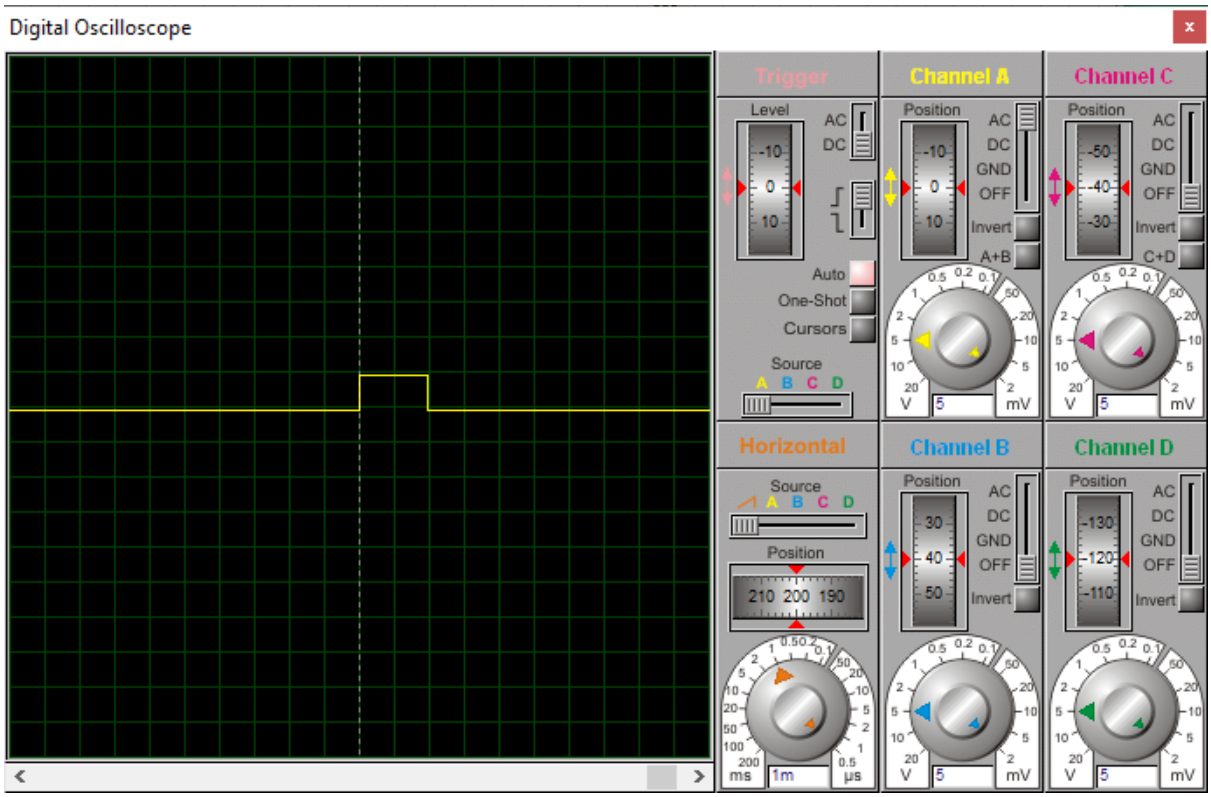

Figura 16. Osciloscopio en la Última Posición (180 grados)

Por último, cabe señalar que las posiciones no son totalmente exactas debido a que las demás instrucciones utilizadas en el programa causan retardos, además de la pérdida de números decimales, por lo que es prácticamente imposible conseguir posiciones más exactas.

Conclusiones Individuales

Contreras Cardoso Adolfo

La elaboración de esta tarea fue en parte sencilla pues solo se tenía que dividir en segmentos para que se pudiera hacer girar el servomotor en algún ángulo y también aprendimos como utilizar y generar la señal pwm, en lo personal esta práctica me gustó ya que no había trabajado antes con un servomotor.

Martínez Alvarado Bryan Alexis

Durante la realización de esta práctica hemos observado cómo a través de un código podemos controlar un servomotor para que adquiera distintos grados de movimiento de acuerdo a los que se requiera, he logrado aplicar los conocimientos adquiridos en la clase, de manera teórica y ver su funcionamiento en el ámbito práctico

Maya Martínes Alonso Rubén

El uso de servomotores nos ayuda a comprender mejor cómo trabajar con los retardos, y el uso del osciloscopio nos ayuda a entender de forma visual cómo funciona nuestro programa con más detalle ya que vemos claramente el cambio de la señal a través del tiempo. En general esta práctica fue bastante complicada de comprender y con mucho código pero a la vez fue bastante didáctica ya que aprendí muchas cosas con esta práctica

Pérez Gómez Santiago

Hace tiempo había trabajado con servomotores, sin embargo, desconocía su funcionamiento, por lo que adquirir estos conocimientos resulta de gran utilidad, todo esto en conjunto con aprender su manipulación y la generación de la señal PWM, pues en algún punto tenemos que trabajar con robots y todo tipo de circuitos electrónicos, es así que se puede finalizar este trabajo de manera exitosa.