INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO


# REPORTES DE PRACTICAS 2DO PARCIAL

## 3CM8

**UNIDAD DE APRENDIZAJE**

INTRODUCCIÓN A LOS MICROCONTROLADORES

**PROFESOR.** JOSÉ JUAN PEREZ PEREZ

**PRESENTA.**

Gálvez Reyes Angel Alexander

Jonathan Hernández Martínez

Núñez García Tania Itzel

Quiros Díaz Verónica Jackeline

Zúñiga Morales Rodrigo

**SEMESTRE**

2019-A

# ÍNDICE

# INSTITUTO POLITÉCNICO NACIONAL

## ESCUELA SUPERIOR DE CÓMPUTO

# REPORTE DE PRACTICA 5. DECODIFICADOR DE NÚMEROS

3CM8

**UNIDAD DE APRENDIZAJE**

INTRODUCCIÓN A LOS MICROCONTROLADORES

**PROFESOR.** JOSÉ JUAN PEREZ PEREZ

**PRESENTA.**

Gálvez Reyes Angel Alexander

Jonathan Hernández Martínez

Núñez García Tania Itzel

Quiros Díaz Verónica Jackeline

Zúñiga Morales Rodrigo

**SEMESTRE**

2019-A

## INTRODUCCIÓN

Esta práctica se encuentra dividida en dos partes: La primera de éstas consiste en elaborar un circuito el cual tenga como entrada un número binario del 0 al 9 decimal. Posteriormente

el microcontrolador busca en sus registros el número correspondiente y despliega el hexadecimal equivalente para que el display de 7 segmentos muestre el número correctamente. La segunda parte de la práctica no es muy distinta a la primera, con la novedad de que se extiende la numeración hasta el número 15 pero en hexadecimal lo que indica que veremos letras a partir del 9. Cabe destacar que no existen tantos registros para guardar todos estos números.

## MATERIAL

- 1 Protoboard
- 1 Microcontrolador ATMEGA8535
- 1 Push button
- 1 Capacitor electrolítico de 1µF
- 1 Resistencia de 2KΩ
- 1 Dip Switch de 8 terminales
- 2 Cables banana-caimán
- 20 Cables jumper
- Display de 7 segmentos cátodo
- 2 Resistencias de 1KΩ

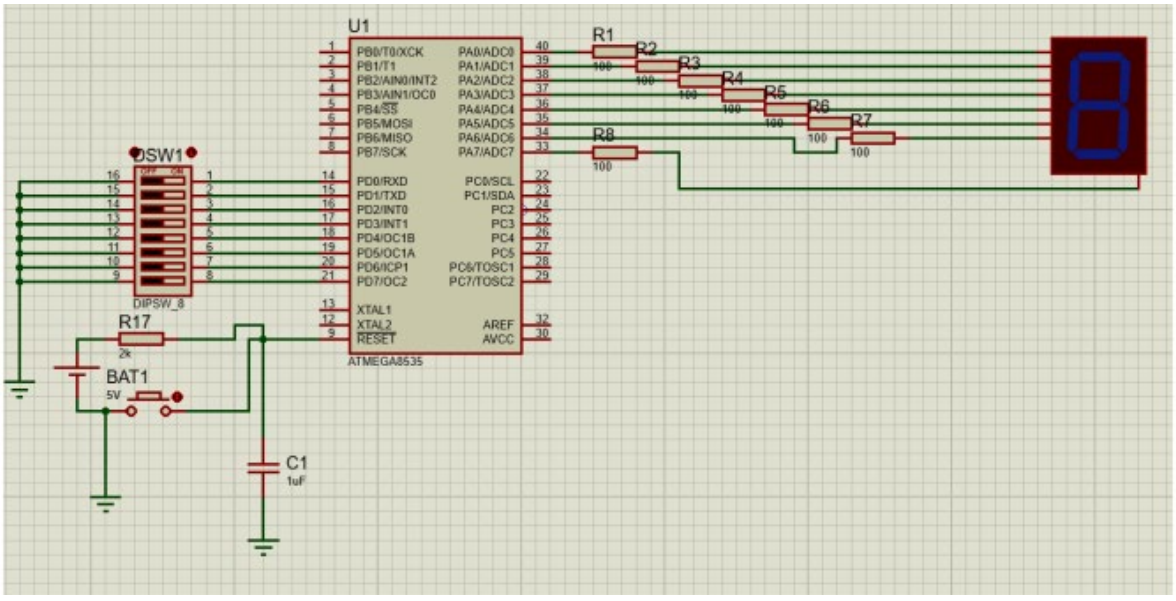
## DESARROLLO

A partir de una combinación binaria en el dipswitch se muestra en el display de 7 segmentos el número en decimal hasta el número 9 para la práctica mientras que en la el número máximo fue el 15 pero mostrado en forma hexadecimal, además de que si se ingresa una opción inválida se implementó mostrar una raya a pesar de que el profesor había indicado mostrar una "e".

Enseguida se muestra el código empleado para dicha práctica, así como el armado del circuito, mediante una simulación en proteus.

```
        .include"m8535def.inc";libreria para incluir los comandos del microcontrolador
        .def dato = R16  ;Definimos a "aux" al reg16 por comodidad
        .def aux = R17 ;Definimos a "aux" al reg17 por comodidad
        ldi r18, $3f
        ldi r19, $06,1;Los primeros numeros (0-9), salidas del display en hexadecimal. //0
        ldi r20, $5b;2
        ldi r21, $4f;3
        ldi r22, $66;4
        ldi r23, $6d;5
        ldi r24, $7d;6
        ldi r25, $27;7
        ldi r26, $7f;8
        ldi r27, $6f;9
        ldi aux,low(RAMEND);Manejo de la pila, parte baja
        out spl,aux; Activamos el funcionamiento del stack pointer
        ldi aux,high(RAMEND);Manejo de la pila, parte alta
        out sph,aux;Activamos el funcionamiento del stack pointer
        ser aux;cargamos con 1's el reg16
        out DDRC,aux;Establecemos los puertos C y A como salidas
        out PORTB,aux;Indicamos que el puerto B es entrada
        out DDRA,aux;Establecemos
uno: ;subrutina de limpieza
        clr dato
dos:;subrutina de
        rcall deco; llamada de subrutina de decodificador
        out portc, dato;cargamos el valor de dato a la salida del puerto C
        rcall delay ;Llamada a subrutina de retardo
        in aux, pinb;se lee el valor del puerto b y se carga a aux
```

```
        andi aux,$80;se realiza un and
        brmi tres;condicion si es 1, realiza la subrutina tres
        inc dato;se incrementa "dato" cuando finalice tres
        cpi dato,10;y en su lugar se copia un 10
        brne dos;si es 0 realiza la subrutina dos
        rjmp dos;finalmente se hace recursividad, brinca a dos , de nuevo
tres:
        dec dato;decrementa el dato
        brmi cuatro
        rjmp dos
cuatro:
        ldi dato,9;se carga un 9 decima y se brinca a subrutina dos
        rjmp dos
deco:
        ldi zl,18;activacion del registro z
        andi dato, $0f;con ayuda de un and
        add zl,dato
        ld aux,z;el dato se carga a aux
        out porta,aux;finalmente se envia el resultado "aux" al puerto a
        ret; regresa a donde se llamo deco
delay:; Subrutina de retardo
      ldi  R28, $BF;se almacena este valor para ocuparlo con registro de prop gral.
WGLOOP0:  ldi  R29, $3A;se almacenan valores para activar los registros y utilizarlos
WGLOOP1:  ldi  R30, $FA
WGLOOP2:  dec  R30;decrementa en un lugar al reg30
      brne WGLOOP2;si resulta un cero, realiza una recursividad
      dec  R29; y decrementa reg
      brne WGLOOP1,si resulta que hay cero, realiza lo que hay en esa subrutina
      dec  R28; decrementa el reg28
      brne WGLOOP0; si continua con cero,realiza esa subrutina
      nop
      ret
```



## CONCLUSIONES

Gálvez Reyes Angel Alexander

Esta práctica no tuvimos tantas complicaciones, solamente tuvimos pequeños problemas para la conexión y el display utilizado. Sin embargo pudimos ver que los decodificadores nos permiten direccionar espacios de memoria, en este caso pudimos convertir bits binarios a hexadecimal, son útiles para programas que hagan manejo de memoria.

Jonathan Hernández Martínez

Para el desarrollo de la práctica tuvimos algunas dificultades para el armado del circuito ya que algunas conexiones estaban mal pero en la simulación los resultados salen de manera correcta por lo que hubo que revisar el circuito, en cuanto a la programación fue complicada pero con lo visto en clase hizo más fácil está tarea.

Núñez García Tania Itzel

Para concluir esta práctica, se puede decir que ésta no contó con un gran nivel de dificultad, puesto que fue la primera como tal en hacer uso de un display, con lo que ya teníamos un poco de experiencia. En lo que sí tuvimos que poner más atención y corregir equivocaciones fue en el armado del circuito, que si bien no era muy grande, si tuvimos algunas complicaciones que nos costó trabajo identificar, pues no funcionó las primeras veces. Sin embargo, se pudo llevar a cabo la práctica.

Quiros Díaz Verónica Jackeline

A pesar de que las dos partes de la práctica fueron relativamente sencillas de armar y correr, se presentaron problemas sencillos a la hora de conectar el display de 7 segmentos, pues si estaba mal conectado, el número que se mostraría, no sería el correcto y por lo tanto la práctica en sí estaría mal, por lo que se procedió a corregir esos pequeños problemas.

Zúñiga Morales Rodrigo

Para esta práctica ya que fue la primera en la que utilizabamos display no se complicó tanto después de que logramos conectarlo ya que debíamos ser cuidadosos en los pines del display para que se mostrara bien el numero ya que el código podía estar bien pero enviará la señal al pin equivocado y mostraría otro resultado.

# INSTITUTO PÓLITECNICO NACIONAL

## ESCUELA SUPERIOR DE CÓMPUTO


REPORTE DE PRACTICA 6. UNIFILA

3CM8

**UNIDAD DE APRENDIZAJE**

INTRODUCCIÓN A LOS MICROCONTROLADORES

**PROFESOR.** JOSÉ JUAN PEREZ PEREZ

**PRESENTA.**

Gálvez Reyes Angel Alexander

Jonathan Hernández Martínez

Núñez García Tania Itzel

Quiros Díaz Verónica Jackeline

Zúñiga Morales Rod

**SEMESTRE**

2019-A

## INTRODUCCIÓN

Esta práctica consta de modelar y simular cinco ventanillas de un banco, en el cual los turnos (0-99), van corriendo y dirigiéndose a cada ventanilla. La ventanilla será asignada por nosotros, mediante un push; el turno será asignado automáticamente, es decir, por cada caja que asignemos el contador del turno, avanzara de uno en uno. Ejemplo: Turno: 03, Caja 5; Turno 04, Caja 3; Turno 05, Caja 1, y así sucesivamente.

## MATERIAL

- 1 Protoboard
- 1 Microcontrolador ATMEGA8535
- 1 Push button
- 1 Capacitor electrolítico de 1µF
- 1 Resistencia de 2KΩ
- 1 Dip Switch de 8 terminales
- 2 Cables banana-caimán
- 20 Cables jumper
- Display de 7 segmentos cátodo
- 2 Resistencias de 1KΩ

## DESARROLLO

Para esta práctica, se hará uso de los conocimientos adquiridos en las prácticas pasadas, las cuales refieren al manejo de un display de 7 segmentos. Los elementos adicionales que serán añadidos a dicha práctica será el uso de subrutinas, así como la implementación de retardos de ¼ de segundo y contadores.

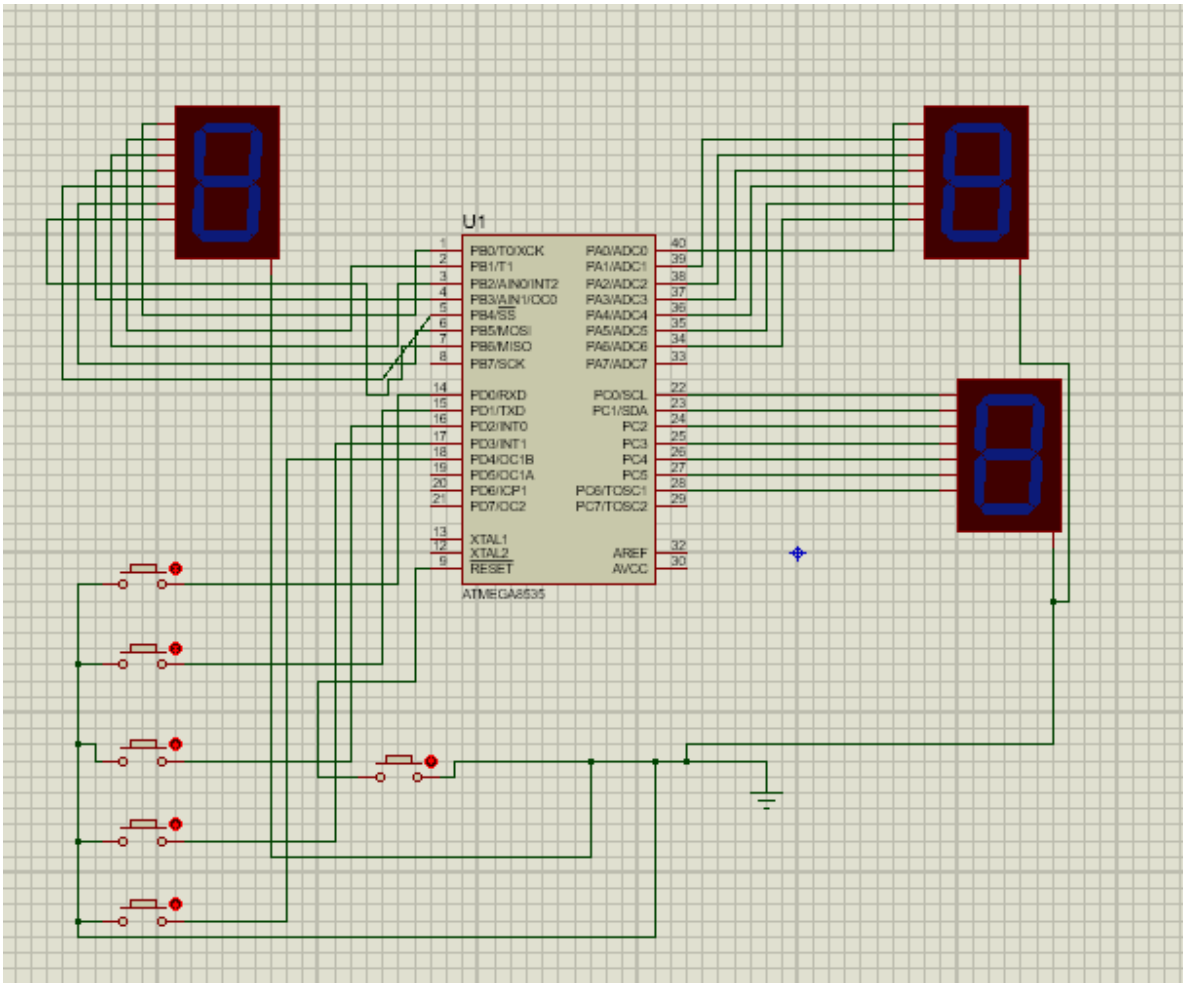
El código implementado en ensamblador, es el siguiente:

```
.include "m8535def.inc";comandos del microcontrolador

.def aux = r16;definiciones de registros para comodidad
.def vent = r17
```

```
        .def turn =r18
;configuracion de la pila
        ldi aux, low(ramend)
        out spl,aux
        ldi aux, high(ramend)
        out sph,aux
      ser aux
;configuracion de puertos
        out ddra, aux
        out ddrb, aux
        out ddrc, aux
        out portd, aux
        clr zh
;Definicion de numeros(0-9) en
hexadecimal
        ldi r20,$3f//0
        ldi r21,6;1
        ldi r22,$5b;2
        ldi r23,$4f;3
        ldi r24,$66;4
        ldi r25,$6d;5
        ldi r26,$7d;6
        ldi r27,$27;7
        ldi r28,$7f;8
        ldi r29,$6f;9
        clr turn;cargamos con ceros
        ldi vent,1;cargamos con 1
sigu:;indica el turno siguiente
        rcall incturn;llama la subrutina
        rcall muestra;llama a  muestra de
display
espera:;subrutina encargada de asignar
la ventanilla
        sbis pind,0;condicion para la caja
1
        rjmp uno
        sbis pind,1;condicion para la
caja2
        rjmp dos
        sbis pind,2;condicion para la
caja3
        rjmp tres
        sbis pind,3;condicion para la
caja4
        rjmp cuatro
        sbis pind,4;condicion para la
caja5
        rjmp cinco
        rjmp espera;recursividad
uno: ;subrutina para caja1
        ldi vent,1
        sbic pind,0
        rjmp sigu
        rjmp uno
dos:;subrutina para caja2
        ldi vent,2
        sbic pind,1
        rjmp sigu
        rjmp dos
tres:;subrutina para caja3
        ldi vent,3
        sbic pind,2
        rjmp sigu
        rjmp tres
cuatro:;subrutina para caja4
        ldi vent,4
        sbic pind,3
        rjmp sigu
        rjmp cuatro
cinco:;subrutina para caja5
        ldi vent,5
        sbic pind,4
        rjmp sigu
        rjmp cinco
muestra:;subrutina encargada de mostrar
las salidas en los displays
        mov aux,vent
        rcall deco
        out portc,aux
        mov aux,turn
        andi aux,$0f
        rcall deco
        out porta,aux
        mov aux,turn
        swap aux
        andi aux,$0f
        rcall deco
        out portb,aux
        ret
deco:;Condiguracion para la
decodificacion de los numeros
        ldi zl,20
        add zl,aux
        ld aux,z
        ret
incturn:;configuracion para la interrupcion
de los turnos
        inc turn
        mov aux,turn
        andi aux,$0f
        cpi aux,$0a
        brne mbcd
```

```
ldi aux,6                    clr turn
add turn,aux          mbcd:
cpi turn,$a0                 ret
brne mbcd
```




CONCLUSIONES

Gálvez Reyes Angel Alexander

La práctica fue algo sencilla, sin embargo se tuvieron algunos problemas de conexión de displays y cables. Este circuito nos sirve para hacer representaciones gráficas de circuitos y poder hacer un contador para los fines que se requieran.

Jonathan Hernández Martínez

En cuanto al contador hecho en ensamblador en mi opinión fue un poco más sencillo que en vhdl por lo que se me hizo muy interesante verlo en otro lenguaje y que fuera un poco más sencillo.

Núñez García Tania Itzel

Si bien, a lo largo de la carrera, ya hemos tenido experiencia con los "contadores", ahora, con el uso de Ensamblador, fue un poco diferente y, en lo personal, un poco más difícil. Sin embargo, la práctica resultó no ser tan complicada y, además, me sirvió para entender un poco más acerca de Ensamblador. En la parte del armado, realmente tampoco hubo complicaciones importantes.

Quiros Díaz Verónica Jackeline

Considero que esta práctica fue relativamente fácil; Y aunque no me he acostumbrado a ensamblador, creo que, al programar y simularla, resulto funcional. Hubo un momento en que tuvimos un inconveniente en el armado del circuito, pero fue solo una mal conexión entre los displays y los puertos del atmega, de ahí en fuera funciono correctamente.

Zúñiga Morales Rodrigo

Aunque el armado del circuito fue lo complicado ya que eran más displays y se debía revisar si estaban bien conectados, al momento de realizar y simular el programa no hubo complicaciones, solo como en la primer practicar, revisar la conexión de los displays.

# INSTITUTO POLITÉCNICO NACIONAL

## ESCUELA SUPERIOR DE CÓMPUTO


# REPORTE DE PRÁCTICA 7. CONTADOR DE 00 HASTA 99

3CM8

**UNIDAD DE APRENDIZAJE**

INTRODUCCIÓN A LOS MICROCONTROLADORES

**PROFESOR.** JOSÉ JUAN PEREZ PEREZ

**PRESENTA.**

Gálvez Reyes Angel Alexander

Jonathan Hernández Martínez

Núñez García Tania Itzel

Quiros Díaz Verónica Jackeline

Zúñiga Morales Rod

**SEMESTRE**

2019-A

## INTRODUCCIÓN

Un contador es un circuito en el que sus salidas siguen una secuencia fija que cuando acaba vuelve a empezar, o circuitos que reciben sus datos en forma serial ordenado en distintos intervalos de tiempo. Los pulsos de entrada pueden ser pulsos de reloj u originarse en una fuente externa y pueden ocurrir a intervalos de tiempo fijo o aleatorio.

## MATERIAL

- 1 Protoboard
- 1 Microcontrolador ATMEGA8535
- 1 Push button
- 1 Capacitor electrolítico de 1µF
- 1 Resistencia de 2KΩ
- 1 Dip Switch de 8 terminales
- 2 Cables banana-caimán
- 20 Cables jumper
- Display de 7 segmentos cátodo
- 2 Resistencias de 1KΩ

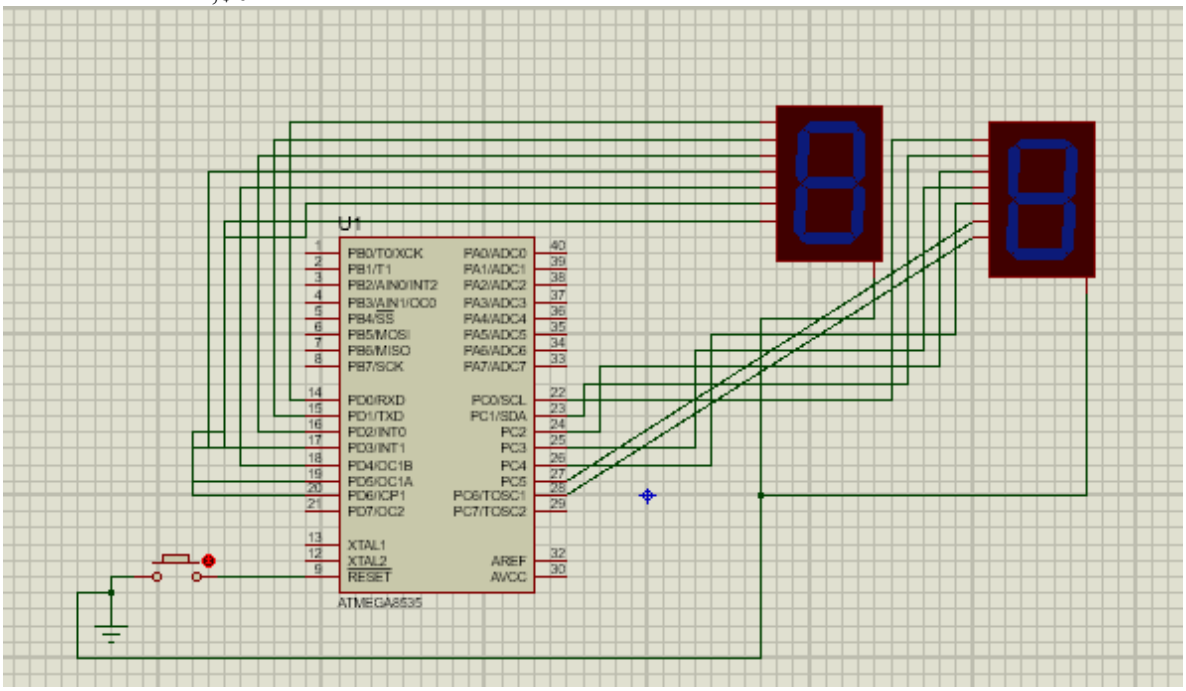
## DESARROLLO

Para la implementación de esta práctica, se retomaron conceptos del decodificador, contador y de retardos, para asegurar que el intervalo de cada numero fuese de 5 segundos. Aquí el objetivo era que mediante dos displays se observa que el contador vaya avanzando del 00 hasta el 99 y en su término, se reinicie y continúe de nuevo con el 00.

Aquí se proporciona el código realizado para dicha práctica:

```
.include"m8535def.inc";librerias para comandos  del microcontrolador
.def aux = r16;definiciones de nombres para cada registro
.def cuenta = r9
rjmp Start;Comienza la subrutina del contrador
.org $008;brinda el tiempo deseado
rjmp incrementa
; Lo que tiene que hacer si se detectan tres flancos del T0
Start:;Configuracion de la pila usada
        ldi aux, low(RAMEND)
        out spl, aux
        ldi aux, high(RAMEND)
        out sph, aux
        ser aux
;Configuracion de puertos
        out ddrd, aux
        out ddrc, aux
        out portb, aux
        out porta, aux
;Valores para el decodificador
        ldi r20,$3f
        ldi r21,6
        ldi r22,$5b
        ldi r23,$4f
        ldi r24,$66
        ldi r25,$6d
        ldi r26,$7d
        ldi r27,$27
        ldi r28,$7f
        ldi r29,$6f
;Configuración de los timer
        ldi r18,$10
        ldi aux, $02
        out mcucr,aux
        ldi aux, $40
        out gicr, aux
;Configurar el timer 1
        ldi aux, 3
        out tccr1b,aux
;Configurar timsk
        ldi r19,$b4 ;para contar 5 segundos
        clr cuenta
        out tcnt1l,r19;activación de las interrupciones
        dec r19
        out tcnt1h,r19
```

```
        inc r19
        sei
        ldi aux,4
        out timsk,aux
        ldi r17,$00
main:;Subrutina de control
        ;out porta,r17
        rcall muestra
        rjmp main
incrementa:
;in r17,pina
;inc r17
        rcall incturn
        reti
muestra:
        mov aux,r17
        andi aux,$0f
        rcall deco
        out portc,aux
        mov aux,r17
        swap aux
        andi aux,$0f
        rcall deco
        out portd,aux
        ret
deco:;Configuración del decodificador
        ldi zl,20
        add zl,aux
        ld aux,z
        ret
incturn:;Configuracion de los turnos
        inc r17
        mov aux,r17
        andi aux,$0f
        cpi aux,$0a
        brne mbcd
        ldi aux,6
        add r17,aux
        cpi r17,$a0
        brne mbcd
        clr r17
mbcd:
        ret
```




## CONCLUSIONES

Gálvez Reyes Angel Alexander

El armado y programado de la práctica fue exitosa, aunque hay detalles con los ciclos que no son tan transparentes a la hora de cambiar el número. Los contadores

son útiles para cualquier cosa, desde un “for” para programación hasta para llevar orden en procesos administrativos de una empresa.

Jonathan Hernández Martínez

Está práctica tuvo su nivel de realización un poco más difícil ya que había ocasiones que el circuito hacia cosas que no debería de hacer o que en el simulador funcionaba de manera correcta pero en el circuito debido al ruido en el push-botton no salía en resultado esperado.

Núñez García Tania Itzel

Lo interesante de esta práctica recayó realmente en analizar los problemas que teníamos con el contador, ya que no en todas las iteraciones o en el paso de los números, el tiempo era el mismo: 5 segundos. Fue además un poco más complicado pues en la simulación se hacía de manera correcta. Pero al final se pudo resolver en el armado y se realizó la práctica con éxito.

Quiros Díaz Verónica Jackeline

La elaboración de esta práctica, no fue muy difícil, solo que para implementar el retardo de 5 segundos, fue necesario realizar unos cuantos trucos, para que tardará el tiempo deseado; Esto a través de una serie de ciclos y lo relevante aquí fue que todas las operaciones se hacían internamente y no se observaba falla alguna en los displays.

Zúñiga Morales Rodrigo

Con esta práctica tuvimos una complicación ya que en un principio parecía realizar bien el conteo cada 5seg pero en unos casos pasaba dos números en vez de uno, pero se logró resolver el problema y acabar la práctica correctamente.

# INSTITUTO PÓLITECNICO NACIONAL

## ESCUELA SUPERIOR DE CÓMPUTO


## REPORTE DE PRÁCTICA 8. HOLA ESTÁTICO

3CM8

**UNIDAD DE APRENDIZAJE**

INTRODUCCIÓN A LOS MICROCONTROLADORES

**PROFESOR.** JOSÉ JUAN PEREZ PEREZ

**PRESENTA.**

Gálvez Reyes Angel Alexander

Jonathan Hernández Martínez

Núñez García Tania Itzel

Quiros Díaz Verónica Jackeline

Zúñiga Morales Rod

SEMESTRE

2019-A

## INTRODUCCIÓN

Los multiplexores son circuitos combinacionales con varias entradas y una salida de datos, y están dotados de entradas de control capaces de seleccionar una, y sólo una, de las entradas de datos para permitir su transmisión desde la entrada seleccionada a la salida que es única.

Podemos decir que la función de un multiplexor consiste en seleccionar una de entre un número de líneas de entrada y transmitir el dato de un canal de información único. Por lo tanto, es equivalente a un conmutador de varias entradas y una salida.

## MATERIAL

- 1 Protoboard
- 1 Microcontrolador ATMEGA8535
- 1 Push button
- 1 Capacitor electrolítico de 1µF
- 1 Resistencia de 2KΩ
- 1 Dip Switch de 8 terminales
- 2 Cables banana-caimán
- 20 Cables jumper
- Display de 7 segmentos cátodo
- 2 Resistencias de 1KΩ

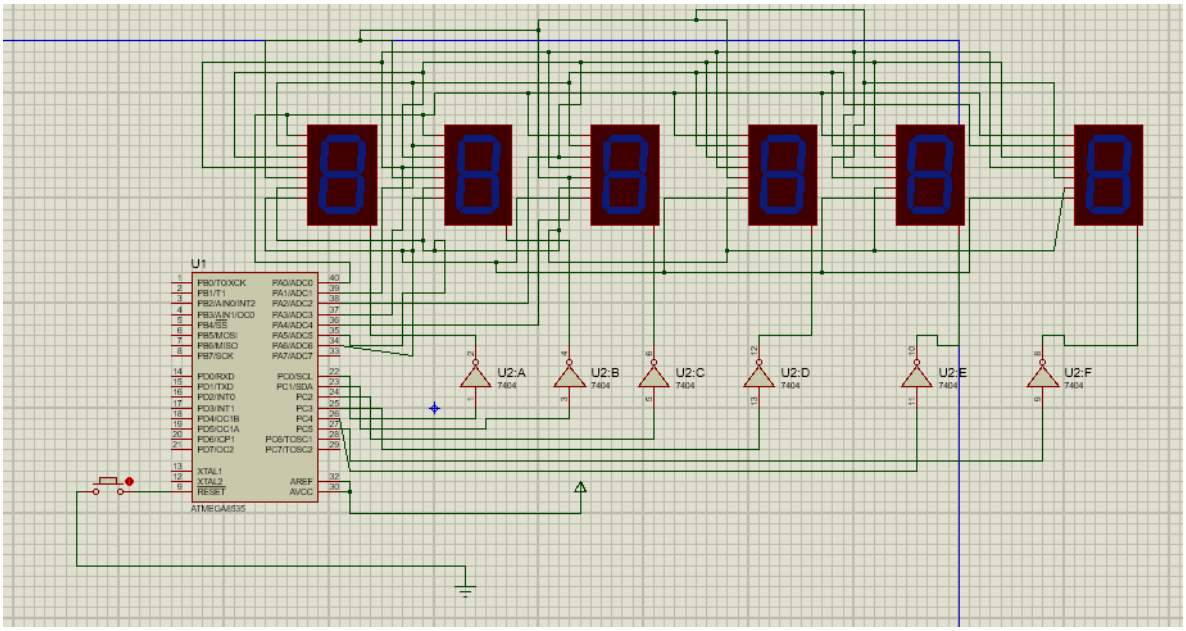
## DESARROLLO

Mediante la multiplexación de 6 displays, se mostrará la palabra “ -HOLA- ”. Para ello, se deberán mostrar las salidas de 0’s y 1’s, en el puerto b y c. El puerto B se encargará de realizar las operaciones para la demostración de las letras en el display, mientras que el puerto C, se encargará de la inversión de las salidas para que ello ocurre en el puerto B.

Enseguida se muestra el código:

```
.include"m8535def.inc";incluye los comandos que se utilizan del microcontrolador
;Definicion de las letras en hexadecimal
        .equ Gi = $40
        .equ H = $76
        .equ O = $3f
        .equ L = $38
        .equ A = $77
        .macro ldb
          ldi r16,@1
          mov @0,r16
        .endm
;Definicion de registros
        .def col =r17
        .def dato =r18
```

```
;Configuración de la pila
        ldi dato,low(RAMEND)
        out spl,dato
        ldi dato,high(RAMEND)
        out sph,dato
        ser dato
;Configuración de los puertos
        out ddrb,dato
        out ddrc,dato
;Distribución de letras.
        ldb r0,Gi
        ldb r1,A
        ldb r2,L
        ldb r3,O
        ldb r4,H
        ldb r5,Gi
        clr zh
dos:
        clr zl
        ldi col,1
uno:;Comportamiento
        out portc,col
        ld dato,z+
        out portb,dato
        rcall delay
        out portb,zh
        lsl col
        cpi col,$40
        brne uno
        rjmp dos
;Retardo
delay: ret
                ldi r19,$1f
WGLOOP0:ldi r20,$2a
WGLOOP1:dec r20
                brne WGLOOP1
                dec r19
                brne WGLOOP0
                nop
        ret
```




## CONCLUSIONES

Gálvez Reyes Angel Alexander

El armado fue bueno, aunque la complejidad del código es mayor debido a los recursos que utiliza en ensamblador. La utilidad de los multiplexores es extensa, desde un serializador hasta realizar funciones lógicas de compuertas. Se pueden

utilizar en las FPGA como las nexys para simular un procesador con Xilinx y realizar operaciones aritméticas y lógicas.

Jonathan Hernández Martínez

En cuanto a la realización de esta práctica se dificulto ya que había temas vistos en clase que no habían quedado claros por lo que hubo que investigarlos antes de implementarlos en cuanto a multiplexado nos mostró que es más eficiente pero a su vez es más complicado de realizar.

Núñez García Tania Itzel

En conclusión, se puede decir que el multiplexado fue de una gran ayuda pues ayudó a optimizar y/o simplificar ciertas cuestiones en cuanto a programación se refiere para operaciones lógicas, etc. Además, considero que el armado del circuito hubiese sido más complicado, aunque claro, en lo personal, fue un poco más difícil que los anteriores.

Quiros Díaz Verónica Jackeline

En primera instancia, creo que el lenguaje ensamblador es muy eficiente para desarrollar programas que, en tamaño, podrían ser extensos en otros lenguajes de alto nivel. Tal es el caso de esta palabra, cada bit hubiera sido un byte en lenguaje y por tanto se hubiera desperdiciado espacio. En segunda instancia, considero que esta práctica fue fácil y entendible a lo que el multiplexado se refiere, que es muy útil.

Zúñiga Morales Rodrigo

Para esta práctica el armado fue mucho más complicado que las anteriores pero ya una vez armado el circuito lo demas seria al momento de realizar el programa aunque el diagrama se ve más complicado al momento de programar no fue gran problema, es de ayuda aplicar el multiplexado.

# INSTITUTO PÓLITECNICO NACIONAL

## ESCUELA SUPERIOR DE CÓMPUTO


## REPORTE DE PRÁCTICA 9. SCROLL

3CM8

**UNIDAD DE APRENDIZAJE**

INTRODUCCIÓN A LOS MICROCONTROLADORES

**PROFESOR.** JOSÉ JUAN PEREZ PEREZ

**PRESENTA.**

Gálvez Reyes Angel Alexander

Jonathan Hernández Martínez

Núñez García Tania Itzel

Quiros Díaz Verónica Jackeline

Zúñiga Morales Rod

**SEMESTRE**

2019-A

INTRODUCCIÓN

Un temporizador contador PIC es un registro que aumenta su valor en una unidad con cada 4 ciclos de reloj al cual se encuentre funcionando el microcontrolador PIC, si por ejemplo la frecuencia del oscilador es de 4MHz, entonces el ciclo de trabajo del microcontrolador PIC será de 1us, por lo que el temporizador contador PIC aumentará su valor de uno en uno en cada microsegundo; por ejemplo, cuando el temporizador aumenta su valor en 10 unidades habrán transcurrido 10us.

El registro en los microcontroladores PIC donde se guardan y realizan los aumentos de uno en uno del temporizador PIC, es llamado registro temporizador contador y es representado por TMRx, donde x es el número de temporizador contador PIC que puede ser 0, 1, 2, dependiendo del número de temporizadores con que cuente el microcontrolador PIC; el temporizador contador PIC puede ser de 8 bits o de 16 bits

MATERIAL

- 1 Protoboard
- 1 Microcontrolador ATMEGA8535
- 1 Push button
- 1 Capacitor electrolítico de 1µF
- 1 Resistencia de 2KΩ
- 1 Dip Switch de 8 terminales
- 2 Cables banana-caimán
- 20 Cables jumper
- Display de 7 segmentos cátodo
- 2 Resistencias de 1KΩ

DESARROLLO

Partiendo de la idea de la práctica anterior, ahora se deben mostrar 3 mensajes, los cuales se escogerán con ayuda de push Bottom. La diferencia recae en que cada uno de ellos deberá realizar un barrido, es decir un movimiento automático.

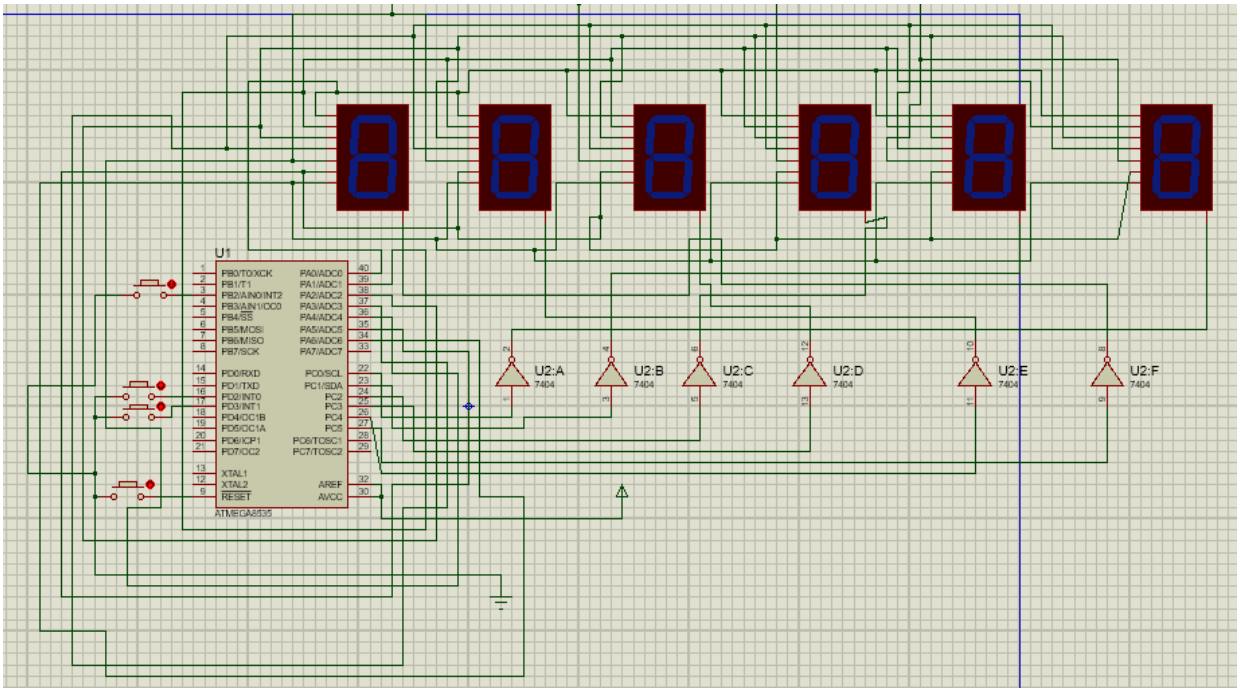
El código empleado en esta práctica, es el siguiente:

```
.include"m8535def.inc";Incluye los comandos para el microcontrolador
;Definicion de letras del abecedario en hexadecimal
.def aux = r16
.def col = r17
.equ A = $77
.equ B = $7c
.equ C = $39
.equ D = $5e
.equ E = $79
.equ F = $71
.equ G = $7d
.equ H = $76
.equ I = $06
.equ J = $1e
```

```
.equ L = $38
.equ N = $37
.equ O = $3f
.equ P = $73
.equ Q = $67
.equ R = $50
.equ S = $6d
.equ T = $78
.equ U = $3e
.equ Ye = $6e
;Macros
.macro ldb
        ldi aux, @1
        mov @0,aux
        .endm
.macro mensaje
        ldb r9, @0
        ldb r8, @1
        ldb r7, @2
        ldb r6, @3
        ldb r5, @4
        ldb r4, @5
        ldb r3, @6
        ldb r2, @7
        ldb r1, @8
        ldb r0, @9
        .endm
;configuración del Reset
reset:
        rjmp main; vector de reset
        rjmp texto1; vector INT 0
        rjmp texto2; vector INT 1
        .org $008
        rjmp corre; vector timer1
        rjmp barre; vector timer0
        .org $012
        rjmp stop; vector INT2
;Subrutina de control
main:
;Configuración de la Pila
        ldi aux, low(ramend)
        out spl, aux
        ldi aux, high(ramend)
        out sph, aux
        rcall config_io
        rcall texto0
        clr zh
        clr zl
        ldi col,1
        out portc,col
        ld aux, z
        out porta,aux
;Subrutina de negacion
uno: nop
        nop
        rjmp uno
;Configuracion de puertos
config_io:
        ser aux
        out ddra,aux
        out portb,aux
        out ddrc, aux
        out portd, aux
        ldi aux,3
;Configuracion de timers e interrupciones
        out tccr0,aux; oreescala ck/64
        ldi aux,2
        out tccr1b,aux; oreescala ck/8
        ldi aux, $01; 0000 0001
        out timsk,aux; toie0
        ldi r18,193; para contar 63 4ms
        ldi aux,$0a; 0000 1010
        out mcucr,aux; registro de control
del censado 0 y 1
        ldi aux, $e0;
        out gicr, aux
        sei
        ret
;Configuración de mensajes
texto0:
        mensaje 0,0,H,U,E,L,U,N,N,0,0
        rcall limpia
        ret
texto1:
        mensaje 0,0,G,L,O,R,I,A,0,0,0
        rcall limpia
        reti
texto2:
        mensaje P,O,L,I,T,E,C,N,I,C
        rcall limpia
        reti
;configuración de limpieza de registros
limpia:
        clr r10
        clr r11
        clr r12
        clr r13
        clr r14
        clr r15
        ret
;Configuracion del movimiento
barre:
```

```
        out tcnt0, r18
        out porta,zh
        inc zl
        lsl col
        brne dos; si z=0
        ldi col,1
        clr zl
dos:
        out portc,col
        ld aux,z
        out porta,aux
        reti
;Configuracion del recorrido
corre:
        mov r15,r14
        mov r14,r13
        mov r13,r12
        mov r12,r11
        mov r11,r10
        mov r10,r9
        mov r9,r8
        mov r8,r7
        mov r7,r6
        mov r6,r5
        mov r5,r4
        mov r4,r3
        mov r3,r2
        mov r2,r1
        mov r1,r0
        mov r0,r15
        reti

stop: ;Configuración del alto total de los mensajes
        push aux
        push col
        in aux, timsk
        ldi col, $04; 0000 0100
        eor aux,col
        out timsk,aux
        pop col
        pop aux
        reti
```




## CONCLUSIONES

Gálvez Reyes Angel Alexander

Esta práctica muestra un nivel de dificultad un poco alto para nuestro equipo debido a que tuvimos que entender la lógica para el temporizador, el limpiado y la forma en cómo se despliega y barre los mensajes, sin embargo fue exitosa su realización. Los temporizadores son ideales para proveer bases de tiempo de alta precisión para la implementación de relojes de tiempo real.

Jonathan Hernández Martínez

Para la realización de esta práctica pudimos notar que cada vez es más complicado implementar las prácticas ya que sí los temas anteriores no están claros hay que retomarlos también podemos ver que hay que dedicarles más tiempo y verificar el funcionamiento del circuito ya que algunos cables dejaron de funcionar.

Núñez García Tania Itzel

Si bien antes ya había tenido alguna experiencia con el "barrido" y "limpieza" de palabras en displays en otra materia, en esta práctica sentí esa complejidad mayor, tal vez por el lenguaje de programación, pero me ayudó a entender mejor cómo funciona. Y claro que la mayor labor fue saber cómo programar estas funciones en los displays, pues no era cosa fácil. Pero al final, pudimos resolver esas dudas y entender la práctica, después de algunos inconvenientes al momento de hacerla.

Quiros Díaz Verónica Jackeline

Para esta práctica, sentí que el grado de dificultad subió a pesar de que no fue difícil diseñarla, fue laboriosa realizarla, ya que nos topamos con algunos inconvenientes a la hora de programar las funciones del barrido, la limpieza, etc. Más que nada, se complicó un poco la lógica para desarrollarla; lo bueno fue que por fin nos quedó y en la simulación fue un éxito.

Zúñiga Morales Rodrigo

En esta práctica fue más complicada al momento de armar y realizar el programa ya que en vez de mostrar un mensaje se mostrarán más, la complicación fue al momento de realizar el barrido y borrado de lo que ya mostraba, a pesar de eso al final la práctica fue realizada correctamente

# INSTITUTO POLITÉCNICO NACIONAL

## ESCUELA SUPERIOR DE CÓMPUTO


REPORTE DE PRÁCTICA 10. PULSACIONES DURANTE 5 SEG.

3CM8

**UNIDAD DE APRENDIZAJE**

INTRODUCCIÓN A LOS MICROCONTROLADORES

**PROFESOR.** JOSÉ JUAN PEREZ PEREZ

**PRESENTA.**

Gálvez Reyes Angel Alexander

Jonathan Hernández Martínez

Núñez García Tania Itzel

Quiros Díaz Verónica Jackeline

Zúñiga Morales Rod

**SEMESTRE**

2019-A

## INTRODUCCIÓN

Un temporizador contador PIC es un registro que aumenta su valor en una unidad con cada 4 ciclos de reloj al cual se encuentre funcionando el microcontrolador PIC, si por ejemplo la frecuencia del oscilador es de 4MHz, entonces el ciclo de trabajo del microcontrolador PIC será de 1us, por lo que el temporizador contador PIC aumentará su valor de uno en uno en cada microsegundo; por ejemplo, cuando el temporizador aumenta su valor en 10 unidades habrán transcurrido 10us.

El registro en los microcontroladores PIC donde se guardan y realizan los aumentos de uno en uno del temporizador PIC, es llamado registro temporizador contador y es representado por TMRx, donde x es el número de temporizador contador PIC que puede ser 0, 1, 2, dependiendo del número de temporizadores con que cuente el microcontrolador PIC; el temporizador contador PIC puede ser de 8 bits o de 16 bits

## MATERIAL

- 1 Protoboard
- 1 Microcontrolador ATMEGA8535
- 1 Push button
- 1 Capacitor electrolítico de 1µF
- 1 Resistencia de 2KΩ
- 1 Dip Switch de 8 terminales
- 2 Cables banana-caimán
- 20 Cables jumper
- Display de 7 segmentos cátodo
- 2 Resistencias de 1KΩ

## DESARROLLO

Para esta práctica, el conteo de pulsaciones, se debía observar en dos display. El problema consta de realizar una configuración, que al cabo de oprimir un push button varias veces en un rango de cinco segundos, se debe mostrar mediante un contador, el número de las pulsaciones realizadas.

El código empleado en esta práctica, es el siguiente:

```
.include"m8535def.inc"
.def aux = r16
.def cuenta = r9
rjmp Start
rjmp pulso ; INT0
.org $008
rjmp temp0 ; Interrupción del timer0
Start:
        ldi aux, low(RAMEND)
        out spl, aux
        ldi aux, high(RAMEND)
        out sph, aux
        ser aux
        out ddra, aux
        out ddrc, aux
        out portb, aux
        out portd, aux
        ;Valores para el decodificador
        ldi r20,$3f
        ldi r21,6
        ldi r22,$5b
        ldi r23,$4f
```

```
        ldi r24,$66
        ldi r25,$6d
        ldi r26,$7d
        ldi r27,$27
        ldi r28,$7f
        ldi r29,$6f
        ;Configuración de los timer
        ldi r18,$10
        ldi aux, $02
        out mcucr,aux
        ldi aux, $40
        out gicr, aux
        ;Configurar el timer 1
        ldi aux, 3
        out tccr1b,aux
        ;Configurar timsk
        ldi r19,$b4 ;para contar 5 segundos
        clr cuenta
        out tcnt1l,r19
        dec r19
        out tcnt1h,r19
        inc r19
        sei
        ldi aux,4
        out timsk,aux
        ldi r17,$00
main:
        nop
        nop
        rjmp main
temp0:
        rcall muestra
        ldi r17,$00
        reti
muestra:
        ;rcall prueba
        mov aux,r17
        andi aux,$0f
        rcall deco
        out portc,aux
        mov aux,r17
        swap aux
        andi aux,$0f
        rcall deco
        out porta,aux
        ret
prueba:
        mov aux,r17
        andi aux,$0f
        cpi aux,$0a
        brne mbcd
        ldi aux,6
        add r17,aux
        cpi r17,$a0
        brne mbcd
        clr r17
mbcd:
        ret
deco:
        ldi zl,20
        add zl,aux
        ld aux,z
```

```
        ret                                    rcall prueba
pulso:                                         reti
    inc r17
```

## CONCLUSIONES

Gálvez Reyes Angel Alexander

Esta práctica muestra un nivel de dificultad un poco alto para nuestro equipo debido a que tuvimos que entender la lógica para el temporizador, el limpiado y la forma en cómo se despliega y barre los mensajes, sin embargo fue exitosa su realización. Los temporizadores son ideales para proveer bases de tiempo de alta precisión para la implementación de relojes de tiempo real.

Jonathan Hernández Martínez

Para la realización de esta práctica pudimos notar que cada vez es más complicado implementar las prácticas ya que sí los temas anteriores no están claros hay que retomarlos también podemos ver que hay que dedicarles más tiempo y verificar el funcionamiento del circuito ya que algunos cables dejaron de funcionar.

Núñez García Tania Itzel

Si bien antes ya había tenido alguna experiencia con el "barrido" y "limpieza" de palabras en displays en otra materia, en esta práctica sentí esa complejidad mayor, tal vez por el lenguaje de programación, pero me ayudó a entender mejor cómo funciona. Y claro que la mayor labor fue saber cómo programar estas funciones en los displays, pues no era cosa fácil. Pero al final, pudimos resolver esas dudas y entender la práctica, después de algunos inconvenientes al momento de hacerla.

Quiros Díaz Verónica Jackeline

Para esta práctica, sentí que el grado de dificultad subió a pesar de que no fue difícil diseñarla, fue laboriosa realizarla, ya que nos topamos con algunos inconvenientes a la hora de programar las funciones del barrido, la limpieza, etc. Más que nada, se complicó un poco la lógica para desarrollarla; lo bueno fue que por fin nos quedó y en la simulación fue un éxito.

Zúñiga Morales Rodrigo

En esta práctica fue más complicada al momento de armar y realizar el programa ya que en vez de mostrar un mensaje se mostrarán más, la complicación fue al

momento de realizar el barrido y borrado de lo que ya mostraba, a pesar de eso al final la práctica fue realizada correctamente

# INSTITUTO POLITÉCNICO NACIONAL

## ESCUELA SUPERIOR DE CÓMPUTO


## REPORTE DE PRÁCTICA EXAMEN 1- 2DO PARCIAL

3CM8

**UNIDAD DE APRENDIZAJE**

INTRODUCCIÓN A LOS MICROCONTROLADORES

**PROFESOR.** JOSÉ JUAN PEREZ PEREZ

**PRESENTA.**

Gálvez Reyes Angel Alexander

Jonathan Hernández Martínez

Núñez García Tania Itzel

Quiros Díaz Verónica Jackeline

Zúñiga Morales Rod

**SEMESTRE**

2019-A

## INTRODUCCIÓN

Un contador es un circuito en el que sus salidas siguen una secuencia fija que cuando acaba vuelve a empezar, o circuitos que reciben sus datos en forma serial ordenado en distintos intervalos de tiempo. Los pulsos de entrada pueden ser pulsos de reloj u originarse en una fuente externa y pueden ocurrir a intervalos de tiempo fijo o aleatorio

## MATERIAL

- 1 Protoboard
- 1 Microcontrolador ATMEGA8535
- 1 Push button
- 1 Capacitor electrolítico de 1µF
- 1 Resistencia de 2KΩ
- 1 Dip Switch de 8 terminales
- 2 Cables banana-caimán
- 20 Cables jumper
- Display de 7 segmentos cátodo
- 2 Resistencias de 1KΩ

## DESARROLLO

Partiendo de la práctica de pulsaciones, la configuración es similar. El problema recae en que mediante dos leds conectados a la salida de algún puerto de A y C, se debe establecer una conexión a través del microcontrolador a dos push button. Cuando un push se oprime se prenderá una led, y en caso de que se oprima de nuevo ese push o el otro, se apagara y encendera el segundo led. Así sucesivamente, deben prenderse y apagarse los leds, mediantes interrupciones realizadas por los push.

El código empleado en esta práctica, es el siguiente:

```
.include "m8535def.inc"
 .def aux = r16
.def col = r17
.def cuenta = r18
.def estado1 = r21
.def estado2 = r22
.def estado3 = r23
reset:
        rjmp main; vector de reset
        rjmp pulsos; vector INT0
        .org $004
        rjmp bar;vector timer2
        .org $008
        rjmp cin; vector timer1 5 SEG
        rjmp start; vector timer0 3 PULSASCIONES
        .org $013
        rjmp start;
main:
        ldi aux, low(ramend)
        out spl, aux
        ldi aux, high(ramend)
        out sph, aux
        rcall config_io
        rcall conf
        clr zh
        clr zl
        clr aux
        out porta, aux
uno:nop
        nop
```

```
        rjmp uno
config_io:
        ser aux
        out ddra, aux
        out portb, aux
        out portd, aux
        ldi aux, 3
        out tccr2, aux; preescala ck/64
        ldi aux, 4
        out tccr1b, aux; preescala ck/256
        ldi aux, $41; 0100 0001
        out timsk, aux; toei2, toei1, TOEI0
        ldi aux,$06;0000 0110
        out tccr0, aux; Habilitar T0
        ldi aux, $02; flanco de bajada INT0
        out mcucr, aux
        ldi aux, $40
        out gicr, aux
        ldi r18, 193; para contar 63 4ms
        ldi r19, $b4; para 5seg
        ldi r20, 253
        out tcnt0, r20
        sei
        ret
conf:
        ldi estado1, $01
        ldi estado2, $02
        ldi estado3, $03
        ret
bar:
        reti
cin:    in aux, pina
        cpse estado3,aux; comparo el estado del puerto
        rjmp sigue2
        rjmp comp2

        sigue2: ldi cuenta, $00
        out porta, cuenta
        ldi aux, $41
        out timsk, aux; Deshabilita t1
        reti

        comp2: out porta, estado2
        ldi aux, $41
        out timsk, aux; Deshabilita t1
        reti
start:
        ldi aux, $41
        out timsk, aux; Deshabilita t1
        in aux, pina

        cpse estado3,aux; comparo el estado del puerto
        rjmp sigue3
        rjmp comp3

        sigue3:
                out porta, estado2
        reti
        comp3: clr aux
        rcall reinicio
```

```
        out porta, aux
        reti

pulsos:
        ldi aux, $45; 0100 0101
        out timsk, aux; toei2, toei1, TOEI0
        out tcnt1l, r19
        dec r19
        out tcnt1h, r19
        inc r19

        in aux, pina
        cpse estado2,aux; comparo el estado del puerto
        rjmp sigue
        rjmp comp1

        sigue: out porta, estado1
        reti
        comp1:
        out tcnt0, r20
        out porta, estado3
        reti
reinicio:
        ldi aux, 4
        out tccr1b, aux; preescala ck/256
        ldi aux, $41; 0100 0001
        out timsk, aux; toei2, toei1, TOEI0
        ldi aux,$06;0000 0110
        out tccr0, aux; Habilitar T0
        ldi aux, $02; flanco de bajada INT0
        out mcucr, aux
        ldi aux, $40
        out gicr, aux
        ldi r18, 193; para contar 63 4ms
        ldi r19, $b4; para 5seg
        ldi r20, 253
        out tcnt0, r20
ret
```