Instituto Politécnico Nacional

Escuela superior de Cómputo

Unidad de aprendizaje: Introducción a los microcontroladores

Prácticas ***1er Parcial***

Profesor: Pérez Pérez José Juan

Grupo: 3CM8

Alumnos:

Gálvez Reyes Ángel Alexander
Jonathan Hernández Martínez
Nuñez García Tania Itzel
Quiros Díaz Verónica Jackeline
Zuniga Morales Rodrigo

Indice

Práctica 1. Suma

Objetivo.

En esta práctica los alumnos capturarán y simularán los siguientes programas con la finalidad de familiarizarse con el entorno de AVR Studio, además de comprender la lógica que emplea el microcontrolador ATMega8535 por medio del lenguaje ensamblador.

Introducción.

AVR Studio es un entorno de desarrollo Integrado (IDE) de software desarrollado por Atmel. Proporcionando una plataforma de desarrollo única para las familias de microcontroladores AVR de 8 bits.

Material y equipo.

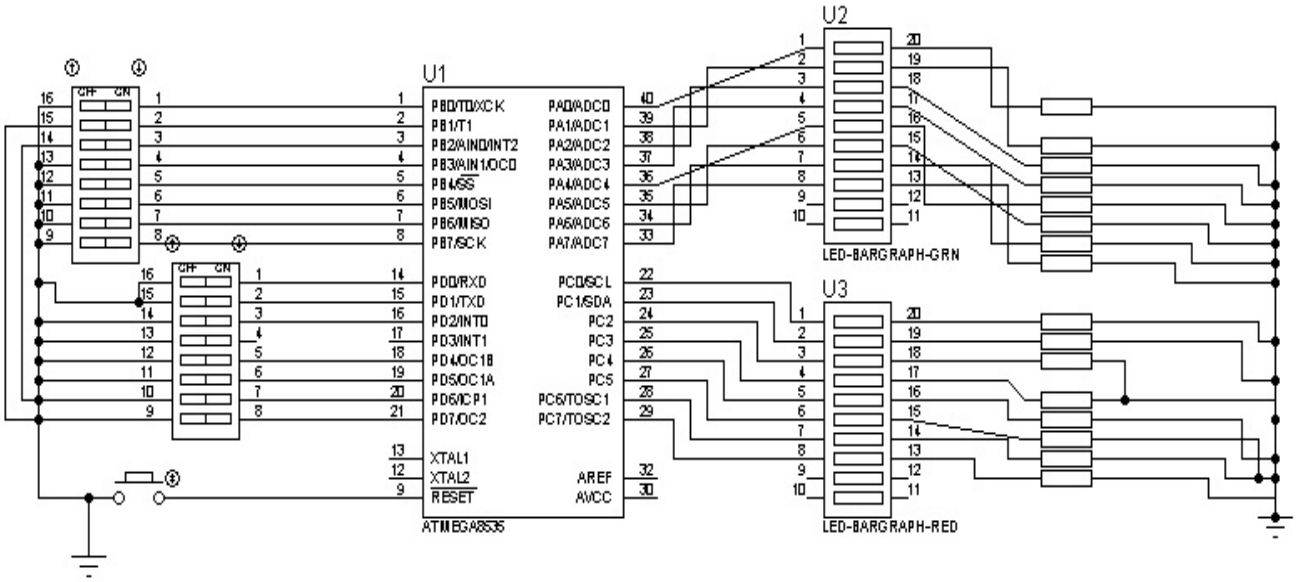
- Equipo de cómputo con AVR Studio.
- Fuente de Voltaje
- Resistencias 100 Ohms
- 2 Protoboards
- 2 Dip switch
- 16 Leds
- Push Button
- ATMega8535

Desarrollo.

El problema es escribir un programa que lea los datos del puerto B, los sume y muestre la suma en el puerto A y el contenido del SREG en el Puerto C.

```
.include "m8535def.inc" ;Se incluye librería del microcontrolador
.def aux= R16 ;se define un registro auxiliar
.def cont = R17 ;se define un contador
ser aux ;cargamos 1s al registro aux
out DDRA,aux ;salida
out DDRC, aux ;salida
out PORTB, aux ;habilitar pull up
aqui:
     in aux,PINB  ;entrada de datos de aux al pin B
     out PORTA, cont  ;salida a puerto A para contador
     out PORTC, aux ;Se despliega la suma
     inc cont ;se incrementa contador
     loop:
          dec aux  ;decrementamos aux
          nop  ;dejamos esperar un tiempo
          brne loop  ;si loop es igual a 0 proceder al salto
          rjmp aqui  ;realiza un salto a aqui
```

Diagrama.



Para el diseño de la práctica,se establecieron, como entrada, los dos dip switch conectados a los puertos B y D del Microcontrolador ATmega8535. Además de conectar el push buttom para el Reset que se encuentra incorporado en el micro. Enseguida para las salidas (Puertos A y C), se colocaron 16 leds, 8 de diferente color y por cada puerto. En la imagen se muestran dos barras de leds, para mejor comodidad del diseño, pero en si, son para el mismo objetivo. Finalmente se colocó una resistencia por cada led, que conectan a tierra. Este diseño, se aplica de igual manera que para la práctica 2,3,4 y el examen.

Conclusiones.

Galvez Reyes Angel Alexander

Después de la ejecución de la práctica adquirimos conocimiento práctico relacionado a la programación en el estudio AVR la cual será de gran ayuda para prácticas posteriores.

Jonathan Hernández Martínez

Tras el correcto desarrollo y análisis de la práctica en cuestión pudimos comprender mejor cómo navegar dentro del estudio AVR para después poder ejecutar los programas que se pidan.

Nuñez García Tania Itzel

Está práctica fue muy didáctica ya que pudimos interactuar con la interfaz que nos ofrece el entorno de AVR studio y practicar algunos comandos sencillos

Quiros Díaz Verónica Jackeline

En un principio, sufrimos ciertos problemas con el armado del circuito, pero una vez realizado este armado, pudimos reafirmar el conocimiento práctico como el ensamble de algunos componentes necesarios para el cableado así como la programación de los mismos.

Zuñiga Morales Rodrigo

Gracias a la exitosa realización de la práctica pudimos reafirmar conocimiento práctico como el ensamble de algunos componentes necesarios para el cableado así como la programación de los mismos.

Práctica 2. Suma de Nibbles

Objetivo.

En esta práctica los alumnos desarrollaran un programa, el cual deberá leer el dato en los puertos y de acuerdo a su signo, este será enviado al puerto A o C, respectivamente. Además de configurar los pull-ups.

Introducción.

Cuando algún pin o los pines de AVR se han configurado como entradas digitales, mediante el registro PORTx se activa o desactiva unas resistencias Pull Up internas al microcontrolador AVR, cuando el bit del registro PORTx correspondiente a algún pin que es utilizado como entrada digital, se pone a 1 se activará la resistencia pull up

correspondiente al pin, cuando se configura como 0 la resistencia pull up de ese pin estará desactivada.

Material y equipo.

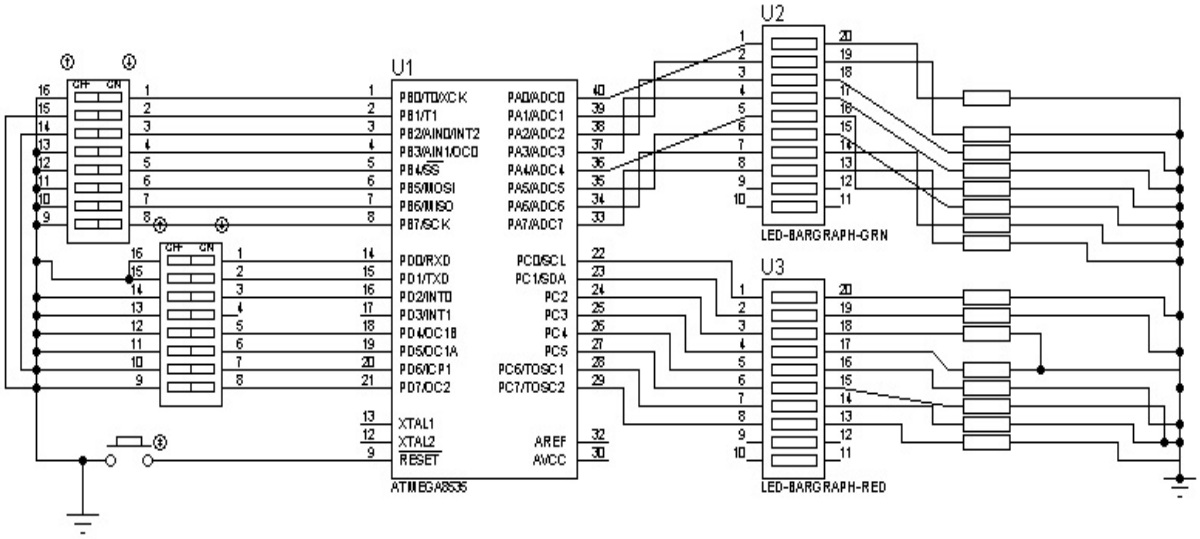
- Equipo de cómputo con AVR Studio.
- Fuente de Voltaje
- Resistencias 100 Ohms
- 2 Protoboards
- 2 Dip switch
- 16 Leds
- Push Button
- ATMega8535

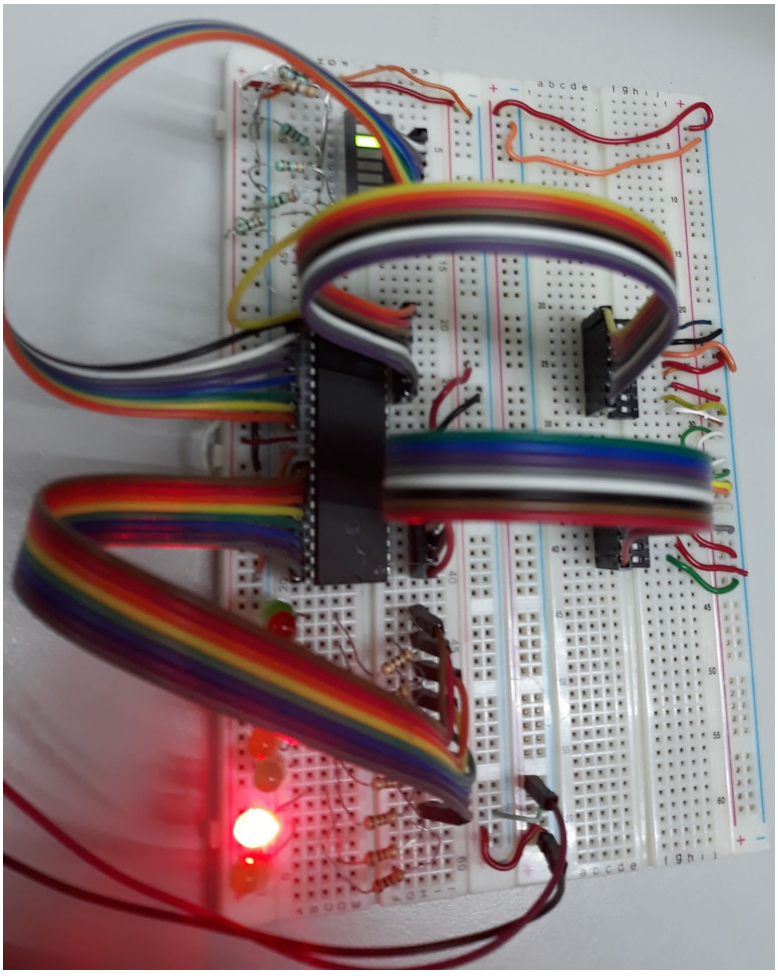
Desarrollo.

Si el bit 7 del puerto B es 0 entonces dividir el dato del puerto C en 2 nibbles y sumarlos, el resultado mostrarlo en puerto A. Si el bit 7 del puerto B es 1 entonces dividir el dato del puerto D en 2 nibbles, sumarlos y mostrarlos en el puerto C.

```
.include"m8535def.inc"
;definimos las variables a usar
.def aux = r16
.def datb = r18
.def a = r20
.def b = r21
ser aux
out ddra,aux ; nuestras salidas
out ddrc,aux
out portb,aux ; nuestras entradas
out portd,aux

loop:
     in datb,pinb ;se agregan los nibbles
     mov a,datb ;aqui ponemos el primer nibble
     andi a,$0F
     mov b,datb ;aqui ponemos el segundo nibble
     swap b
     andi b,$0F
     add a,b ;se suman los nibbles
     out portc, a ; se muestran los nibbles
     rjmp loop
```

Diagrama.





Conclusiones.

Galvez Reyes Angel Alexander

Siempre hay complejidad a la hora de armar los circuitos, fuera de eso, era necesario entender el funcionamiento de los nibbles para que se puedan sumar y mostrarse en pantalla.

Jonathan Hernández Martínez

En la implementación de esta práctica tuvimos problemas en el circuito ya que tuvimos que armar otro circuito porque el primero dejó de funcionar. También hubo que hacer algunas modificaciones en el código para que funcionara de manera adecuada.

Nuñez García Tania Itzel

Considero que esta práctica, a pesar de no parecer muy difícil al principio, se nos complicó de más debido al circuito, que no funcionaba y no sabíamos con exactitud por qué. Pero al armarlo de nuevo, logró funcionar de manera correcta.

Quiros Díaz Verónica Jackeline

Para esta práctica, tuvimos algunos inconvenientes con el funcionamiento del programa pero, conforme se realizaban pruebas, pudimos sacar la práctica y llevarla a funcionar en el circuito.

Zuñiga Morales Rodrigo

Debido a problemas con el circuito, algunos componentes y el buen funcionamiento de algunos micros nos costó mucho esta práctica pero se logro hacer funcionar hasta el tercer circuito y usando otros micros.

Práctica 3. Contador con Delay

Objetivo.

En esta práctica los alumnos desarrollaran un programa, el cual deberá de tener un contador de suma, pero con delays (o retrasos) de tiempo definidos. Para ello emplearemos el ATMega8535 y una computadora con AVR Studio para la programación del microcontrolador y posteriormente hacer el armado del circuito para la práctica.

Introducción.

Para el microcontrolador ATMega8535, delay nos permite crear subrutinas de tiempo por milisegundos. En esta práctica se usará para crear subrutinas de tiempo de varios (250) milisegundos en un contador, además de deco para poder mostrar tal conteo en displays.

Material y equipo.

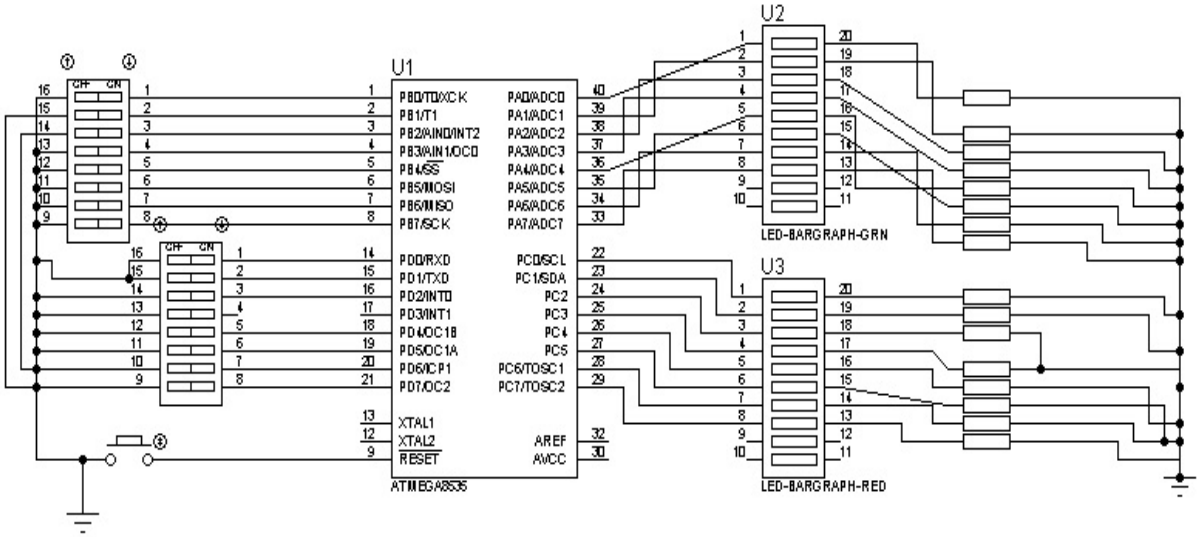
- Equipo de cómputo con AVR Studio.
- Fuente de Voltaje
- Resistencias 100 Ohms
- 2 Protoboards
- 2 Dip switch
- 16 Leds
- Push Button
- ATMega8535

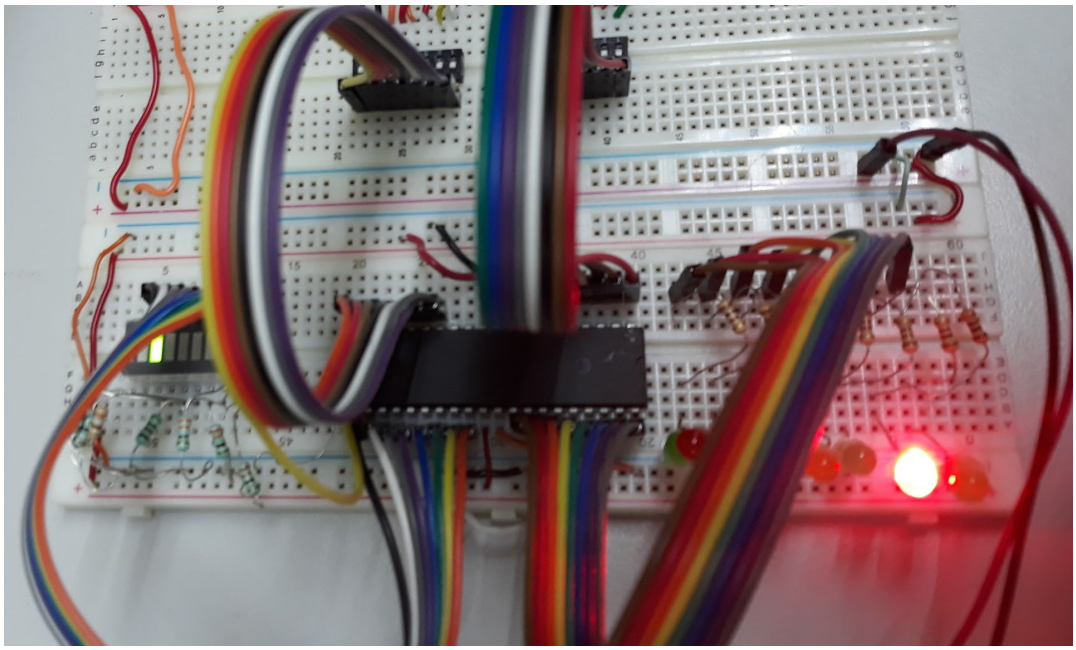
Desarrollo.

Escribir un programa que realice lo siguiente: el puerto A se configura como salida, el puerto B se deja como entrada y se habilitan los pull-up. Al inicio del funcionamiento deberá estar el dato $00 en el puerto B y mostrarse #00 en el puerto A al mover el dato del puerto B a algún dato diferente de $00, este debera sumarse al mostrado en el puerto A y mostrarse en el mismo puerto A una sola vez para sumar otro dato (al nuevo dato mostrado), se deberá regresar a $00 el dato en el puerto B y cambiar al nuevo dato, y así sucesivamente.

```
     .include "m8535def.inc"
     .def cta=r18
     .def nvcs=r19
     .def aux=r16
     ldi aux,low(ramend)
     out spl,aux
     ;se declaran las entradas y salidas con pull ups
     ldi aux,high(ramend)
     out sph,aux
     ser aux
     out ddra,aux
     out portb,aux
     clr cta

nvo:
     in nvcs,pinb ;esta es la entrada del dato en puerto B
     out porta, cta ;esta es la salida en puerto A
nret:
     rcall delay ;es la llamada para poner los datos en 0 y
hacer el retraso
     dec nvcs
     brne nret ;si nret no es 0 se salta
     inc cta ;incrementa el contador de suma
     rjmp nvo
delay:
     ldi r17,0 ;carga 0 a los dos registros que utilizamos
auxiliares
     ldi aux,0
et1:
     dec r17 ;decrementa
     brne et1 ;si et1 es 0 hace el salto para decrementar r17
     dec aux ;decrementa aux
     brne et1
     ret
```

Diagrama.





Conclusiones.

Galvez Reyes Angel Alexander

Esta práctica fue buena para entender cómo puede recrear el funcionamiento de un ralentizador o retrasos para poder optimizar el manejo de recursos y tomar decisiones adecuadas. Al pasarlo al circuito se pudo ver cómo se ralentiza o corre más rápido la cuenta en los leds de forma binaria.

Jonathan Hernández Martínez

La realización de estas prácticas son un poco sencillas pero como somos nuevos en este lenguaje resulta ser un poco más complicado y al momento de pasarlo al circuito funciona

de manera distinta por lo que se dificulta saber si es el circuito o la programación del micro.

Nuñez García Tania Itzel

En lo personal, comprender el funcionamiento de los retrasos no fue muy sencillo al inicio de la práctica, sobre todo para entender su lógica y llevarla a código. Sin embargo, después de algunos intentos en la programación y además en el armado del circuito, se logró terminar la práctica con éxito.

Quiroz Diaz Verónica Jackeline

En un principio resultó un poco difícil desarrollar esta práctica, puesto que no comprendimos en primera instancia la lógica del delay. Después de aminorar este problemita y continuando con detalles del circuito físico, se logró terminar la práctica satisfactoriamente.

Zuñiga Morales Rodrigo

Aunque la práctica un poco confusa al principio, una vez armado el circuito y realizar las pruebas con el código se logró realizar la práctica y comprendí el funcionamiento del delay.

Práctica 4. Contador de Sumas

Objetivo.

Se necesita realizar un contador de sumas y desplegar la suma que se está realizando. Para ello necesitaremos el ATMega8535 y una computadora con AVR Studio para poder programar el microcontrolador y posteriormente pasar al armado del circuito contador de sumas.

Introducción.

Los contadores de sumas son útiles, por ejemplo, para desplegar el conteo en el puerto C, mientras que la suma se realiza en el puerto A y se muestra. Con un dipswitch en el puerto B se inicia desde cero y se pone un número, después se ponen en ceros y luego se vuelve a poner otro número. El contador del puerto C va a esperar un cierto tiempo hasta que se refleje la suma y vuelva a contar nuevamente.

Material y equipo.

- Equipo de cómputo con AVR Studio.
- Fuente de Voltaje
- Resistencias 100 Ohms
- 2 Protoboards
- 1 Dip switch
- 16 Leds
- Push Button
- ATMega8535

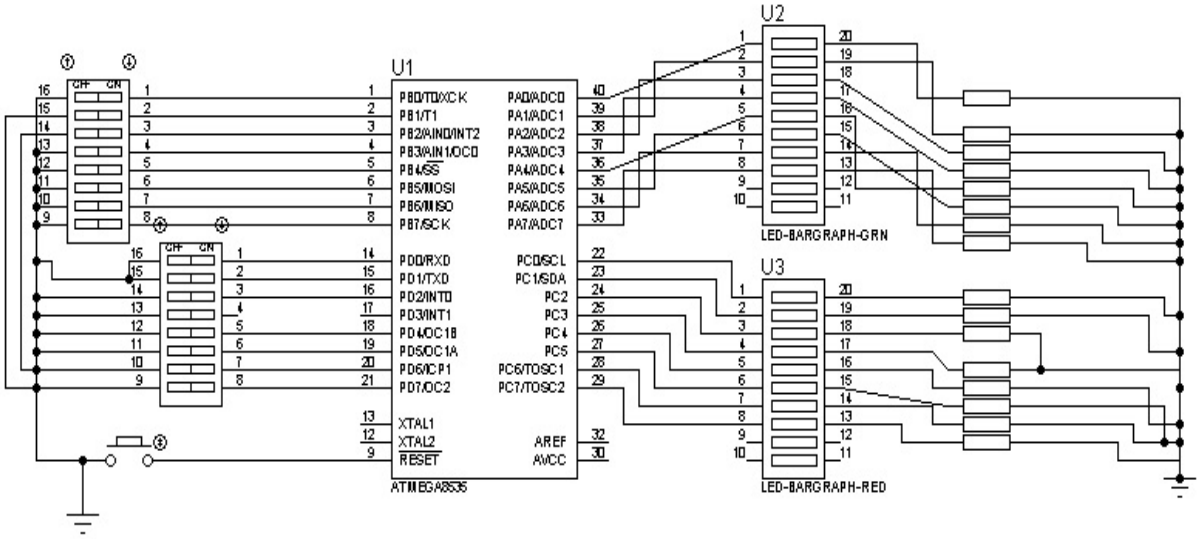
Desarrollo.

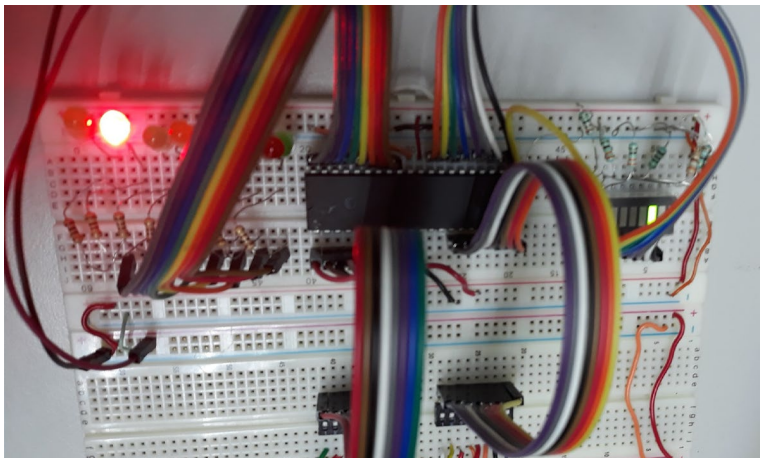
El puerto A y el puerto C se configuran como salida, el puerto B se deja como entrada y se habilitan los pull-up. Al inicio del funcionamiento deberá estar el dato del puerto B mostrado en el puerto A, al mover el dato del puerto B (los datos a sumar son 1,2,4,8,16,32,64 y 128) a algún dato diferente de $00, este deberá sumarse al mostrado en el puerto A, en el puerto C deberá mostrarse el número de sumas que se han realizado, para sumar otro dato (al nuevo dato mostrado), se deberá regresar a $00 el dato en el puerto B y cambiar al nuevo dato y así sucesivamente.

```
.include"m8535def.inc"
.def aux = r16
.def data = r17
.def datb = r18
.def datc = r19

ser aux ;Rellena de ceros el registro aux
out ddra,aux ;Despliega aux en el puerto a
out portb, aux ;Despliega aux en el puerto b
out ddrc, aux ;Despliega aux en el puerto c
clr datc ; limpia el registro c


loop:
     in datb,pinb
     cpi datb,0 ;compara los registros en caso de ser cero
     breq loop ;Compara si es uno para regresar
     in data, pina
     add data, datb; suma los datos del puerto a y b
```

```
      out porta, data ;despliega el resultado en el puerto a
      inc datc ;incrementa en uno el dato de del puerto c
      out portc, datc

aki:
      in datb,pinb
      cpi datb,0 ;compara los registros en caso de ser cero
      brne aki ;Si es cero regresa
      rjmp loop
```

Diagrama.





Conclusiones.

Galvez Reyes Angel Alexander

Esta práctica es integral, junta lo que hemos aprendido en el parcial, fue un tanto laboriosa pero al final se podía hacer sumas y a la vez mostrar cuántas sumas hemos hecho después de cada transición de los números que se sumaban con el dip switch. Esto es bueno para poder saber qué se está sumando y cuantas veces en la lógica de un programa de forma física.

Jonathan Hernández Martínez

Cómo podemos ver esta práctica hace uso de las prácticas anteriores por lo que se supone que debería de ser más sencilla su implementación pero nos toma mucho tiempo hacer las pruebas en el circuito ya que siempre hay algún impedimento para su liberación.

Nuñez García Tania Itzel

Esta práctica en especial, resultó ser más laboriosa debido a lo que se pedía en ella. Nos costó especial trabajo poder conjuntar todo y, en lo personal, el acomodo del circuito. Además, la programación del código y todo en conjunto, como poder lograr que se vieran los números que se estaban sumando en unos displays y, por otro lado, el resultado de esta suma. Al final, pudimos organizar todas esas ideas y realizar la práctica de manera correcta.

Quiroz Diaz Verónica Jackeline

Esta práctica resultó mucho más difícil que las anteriores, puesto que abarcaba, en esencia, todo lo que vimos en el transcurso del parcial. Nos tomó mucho tiempo y paciencia llevarla a cabo, pero finalmente se logró y salió bien.

Zuñiga Morales Rodrigo

La práctica fue complicada ya que se necesitaba conocer de las anteriores prácticas y algunas aun no nos quedaban pero después de varias pruebas se logró realizar .

Práctica Examen. Nibbles (Positivos/Negativos)

Objetivo.

Se necesita realizar la implementación del examen en código ensamblador. Para ello necesitaremos el ATMega8535 y una computadora con AVR Studio para poder programar el microcontrolador y posteriormente pasar al armado del circuito para realizar la cuenta de números positivos y negativos y la suma que se realizó.

Introducción.

Un nibble es una colección de 4 bits. No sería un tipo de dato interesante a excepción de que con un nibble se presenta un número BCD y también que un nibble puede representar un dígito hexadecimal. Su interés se debe a que cada cifra en hexadecimal (0, 1, 2,..., 9, A, B, C, D, E, F) se puede representar con un cuarteto, puesto que 2 elevado a la 4 es 16 (24=16). También el cuarteto es la base del sistema de codificación BCD.

Material y equipo.

- Equipo de cómputo con AVR Studio.
- Fuente de Voltaje
- Resistencias 100 Ohms
- 2 Protoboards
- 2 Dip switch
- 16 Leds
- Push Button
- ATMega8535

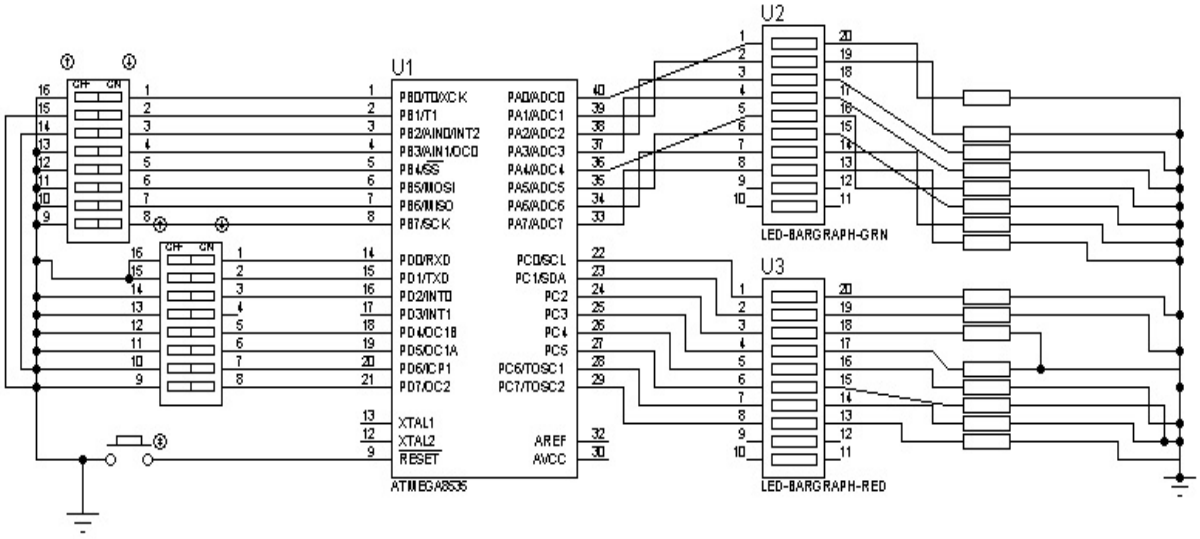
Desarrollo.

Se requiere escribir un programa en ensamblador para el microcontrolador que cuente los números datos negativos y el número de datos no negativos que se han sumado (dato en el puerto B) por cada vez que se pulsa y suelta el botón conectado en el bit 0 del puerto D, la suma se debe mostrar en el puerto A, la cuenta se debe mostrar en dos grupos de 4 bits de forma binaria en el puerto C, nibble alto negativos y nibble bajo no negativos (la cuenta máxima de números sumados será 15 ($0F) para la parte baja y 15 ($F0) para la parte alta.

```
.include"m8535def.inc"
.def data = r17
.def aux = r16
.def datb = r18
.def cont = r19
.def a = r20
.def b = r21
.def op = r22
.def positivos = r23 ; Se asigna un registro para usar #s
positivos
.def negativos = r24 ; Se asigna un registro para usar #s
negativos
.def result = r25


ser aux ; se rellena aux de 1s
clr cont ; se limpia el contador
out ddra,aux ; salida en puerto A
out ddrc,aux ; salida en puerto C
out portb,aux ; entrada en puerto B
out portd,aux ; entrada en puerto D


loop:
     in datb,pinb ; Recibe dato en puerto B
     cpi datb,0 ; compara si el dato en puerto B es cero
     breq loop ; Si es 0 entonces se regresa

     mov aux, datb ; Metemos el dato de B en un auxiliar

     mov a,datb ; Metemos ese mismo dato en A (es 1er nibble)
     andi a,$0F ; Se le aplica un andi con 0F a A
     mov b,datb ; Metemos el segundo nibble a B
     swap b ; Intercambiamos bits de B
     andi b,$0F ; Se le aplica un andi con 0F a B

     in op,pind ; Recibe desde puerto D la operación a realizar
     cpi op,0 ; Compara si ese dato recibido es 0
     breq resta ; Si es 0 entonces va a restar
     add a,b ; Si es 1 entonces va a sumar a con b (los 2 nibbles)
     rjmp signo ; Salta a metodo signo
```

```
resta:
     sub a,b ; Resta a con b

signo:
     brmi negcase ; Verifica si es un número negativo
     inc positivos ; Si no es negativo, incrementa positivos
     rjmp resultados ; Salta a resultados

negcase:
     inc negativos ; Incrementa los negativos

resultados:
     out porta,a ; Manda resultados a puerto A

     mov result, negativos ; Mete negativos al resultado
     andi result,$0F ; Realiza un andi al resultado
     swap result ; Intercambia bits del resultado
     or result, positivos ; Realiza un OR al resultado con los
positivos

     out portc, result ; Muestra el resultado final a puerto C

aki:
     in datb,pinb ; Recibe dato de puerto B
     cpse datb, aux ; Compara aux con dato del puerto B si son
iguales, esto es para detectar cambios en el dato B
     rjmp loop
     rjmp aki

     in datb,pinb ; Recibe dato de B
     cpi datb,0 ; Compara si dato de B es cero
     brne aki ; si es asi, va a comparar el dato de nuevo
     rjmp loop ; Salta a loop
```

Diagrama.



En este caso en todas las prácticas del primer parcial (incluyendo el examen) tuvimos que utilizar el mismo circuito, para el bit 0 utilizamos parte del dip switch para poder simular la interrupción y que cuenten los datos negativos y positivos.

Conclusiones.

Galvez Reyes Angel Alexander

Este examen nos ayuda a entender que un nibble se trata de un conjunto de cuatro bits. Lógicamente un byte puede decirse que se compone de dos nibbles. Resulta interesante que un número hexadecimal como BEEF pueda ser convertido a su equivalente binario tan fácilmente como tomar el primer número “B” y escribirlo directamente en binario cómo “1011”, luego tomar la “E” e igualmente reescribir a “1110”.

Jonathan Hernández Martínez

Desarrollar esta práctica nos permite conocer un poco sobre nibbles y su representación de binario a hexadecimal, así como usar un poco el programa proteus para la simulación del circuito y hacer una simulación previa antes de pasarlo al circuito. Aunque en comparación de los resultados del circuito y de la simulación los resultados fueron muy distintos debido a las conexiones o al ruido en el circuito.

Núñez García Tania Itzel

El desarrollo de esta práctica me pareció sumamente interesante debido a que nos permitió conocer el funcionamiento de las conversiones de números binarios y para poder separarlos en nibbles, esto fue en especial un poco complicado, más para poder reflejar la simulación en el circuito físico. Sin embargo, y a pesar de pequeñas variaciones y complicaciones con los componentes, se llevó a buen término.

Quirós Diaz Verónica Jackeline

Para este examen práctico, en un principio no entendíamos el funcionamiento lógico del problema, pero una vez que comprendimos el concepto de nibble y las conversiones posibles en binario que se pueden hacer, pudimos realizar la operación de asignar los números positivos y negativos con ellos. En simulación, el funcionamiento resultó exitoso y sin ningún inconveniente, pero una vez plasmado en físico, resultaron variaciones que en ocasiones afectan el funcionamiento.

Zuñiga Morales Rodrigo

Esta práctica nos permite ver como un nibble puede representar un dígito hexadecimal, ya que al separar los números en no negativos y negativos, ver cómo se han sumado y mostrarse en grupos de 4 bits donde se verá el nibble alto negativo y nibble bajo no negativo así convirtiéndose en un dígito binario.