



INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

“REPORTES PRÁCTICAS 1-4”



EQUIPO: 1

- Contreras Leal Armando David
- Herrera Monter Daniel Isaías

Unidad de Aprendizaje: Introducción a los Microcontroladores

Profesor: Pérez Pérez José Juan

Grupo: 3CM6

Índice

Práctica 1	3
Objetivo:.....	3
Material.....	3
Introducción Teórica	3
Desarrollo.....	4
Simulaciones	6
Conclusiones	8
Bibliografía	8
Práctica 2	9
Objetivo:.....	9
Material.....	9
Introducción teórica	9
Desarrollo.....	10
Simulaciones	12
Conclusiones	13
Bibliografía:	13
Práctica 3	14
Objetivo:.....	14
Material.....	14
Introducción Teórica	14
Desarrollo.....	15
Simulaciones.	16
Conclusiones	17
Bibliografía	17
Práctica 4	18
Objetivo:.....	18
Material.....	18
Introducción	18
Desarrollo.....	19
Simulaciones	20
Conclusiones	22
Bibliografía	22

Práctica 1

Objetivo: El objetivo de esta práctica es realizar un programa que realice una suma en un puerto determinado, para luego en otro puerto distinto mostrar el resultado, así como los estados de las distintas banderas al momento de realizar la operación.

Material

Para la realización de esta práctica se necesitan lo siguientes materiales:

- 1 microcontrolador ATM8535
- 1 pinzas de corte
- 1 dip switch de 8 interruptores
- 2 metros de cable del No. 14
- 2 barras led de 8 segmentos
- 1 protoboard
- 1 fuente de voltaje que pueda proveer al menos 5V.

Introducción Teórica

Un microcontrolador es un circuito integrado programable, capaz de ejecutar órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

Los microcontroladores están diseñados para reducir el costo económico de un sistema en particular. Por eso el tamaño del CPU, la cantidad de memoria y los periféricos incluidos dependerán de la aplicación.

La unidad central de procesamiento (conocida por las siglas CPU) es el hardware dentro de un ordenador u otros dispositivos programables, que interpreta las instrucciones de un programa informático mediante la realización de operaciones básicas aritméticas, lógicas y de entrada/salida del sistema.

La memoria es el dispositivo que retiene, memoriza o almacena datos informáticos durante algún periodo de tiempo. La memoria proporciona una de las principales funciones de la computación moderna: el almacenamiento de información y conocimiento. Es uno de los componentes fundamentales de la computadora, que interconectada a la CPU y los dispositivos de entrada/salida, implementan lo fundamental de modelos de computadora de la arquitectura Von Neumann.

Un periférico de entrada/salida o E/S es aquel tipo de dispositivo periférico de un computador capaz de interactuar con los elementos externos a ese sistema de forma bidireccional, es decir, que permite tanto que sea ingresada información desde un sistema externo, como emitir información a partir de ese sistema.

La suma o adición binaria es análoga a la de los números decimales. La diferencia radica en que en los números binarios se produce un acarreo (carry) cuando la suma excede de uno, mientras que en decimal se produce un acarreo cuando la suma excede de nueve.

Para realizar la suma y resta binaria se debe tener en cuenta las siguientes consideraciones:

- Los números o sumandos se suman en paralelo o en columnas, colocando un número encima del otro. Todos los números bajo la misma columna tienen el mismo valor posicional.
- El orden de ubicación de los sumandos no importa (propiedad conmutativa).
- La resta de dos números binarios puede obtenerse sumando al minuendo el “complemento a dos” del sustraendo.

Desarrollo

El desarrollo de esta práctica consistió en dos partes, la primera es el armado del circuito y la segunda es la creación del programa para que el microcontrolador trabajara acorde a lo que pretendíamos realizar.

El circuito se encuentra conectado a una fuente de voltaje que provee 5 volts, esto ayuda a que el microcontrolador funcione de forma óptima, sin embargo, para evitar que las barras de led se quemen es necesario utilizar resistencias.

Finalmente, para terminar con la parte correspondiente al armado del circuito debemos mencionar que para el desarrollo de futuras prácticas el circuito que se utilizara puede llegar a ser el mismo (como lo es el caso de la práctica 2), siendo el programa cargado en el microcontrolador lo que haga la diferencia en el funcionamiento de dicho circuito. En la imagen 1 podemos ver el circuito armado, siendo los leds rojos aquellos que muestran las banderas de la operación y los leds verdes el resultado de dicha operación.

El primer paso para realizar el programa fue abrir AVR Studio y crear el documento correspondiente, el nombre que se le dio al programa fue Suma.asm ya que el objetivo de este programa es realizar una suma.

Una vez que creamos el documento empezamos por llamar a la librería que nos ayudaría a utilizar el microcontrolador ATM8535 para hacer esto utilizamos la línea de código `.include "m8535def.inc"`. A continuación, definimos las entradas y salidas del programa, recordemos que este programa no solo debe realizar la suma, sino que además debe de mostrar las banderas que se activan al momento de realizar una operación.

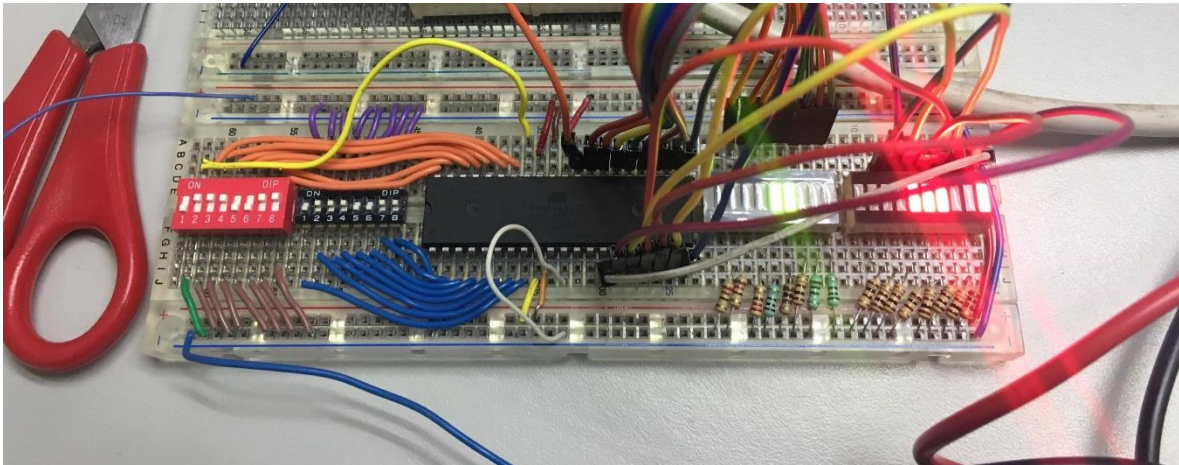


Imagen 1 Circuito armado.

El siguiente paso consiste en almacenar en memoria los valores que introduzca el usuario a través del dip switch, esto se realiza en las dos primeras líneas debajo de la etiqueta `Loop`. Para realizar la suma de los valores que hemos almacenado utilizamos la instrucción `add` seguido de los datos a los que les efectuaremos la operación (en este caso `datb` y `data`) el resultado de la suma se guardara en `datb`.

Finalmente debemos de mostrar el resultado para ello utilizamos el puerto `c`, en cuanto a las banderas que se han activado al momento de realizar la operación utilizamos `SREG` el cual podemos cargar en `valr` y luego mostrar el resultado en el puerto `d`.

El código realizado para esta práctica se muestra en la imagen 2.

```

Suma.asm

.include "m8535def.inc"
.def aux = r16
.def data = r17
.def datb = r18
.def datc = r19
.def valr = r15

ser aux //Salidas
out ddrd,aux
out ddrc,aux
out portb,aux //Entradas
out porta,aux

Loop:
in datb,pinb //Ingresamos a mem entrada en b
in data,pina //Ingresamos a mem entrada en a
add datb,data //Sumamos el resultado
out portc,datb

in valr,SREG //Cargamos lo que hay en sreg
out portd,valr //lo mandamos a la salida en d

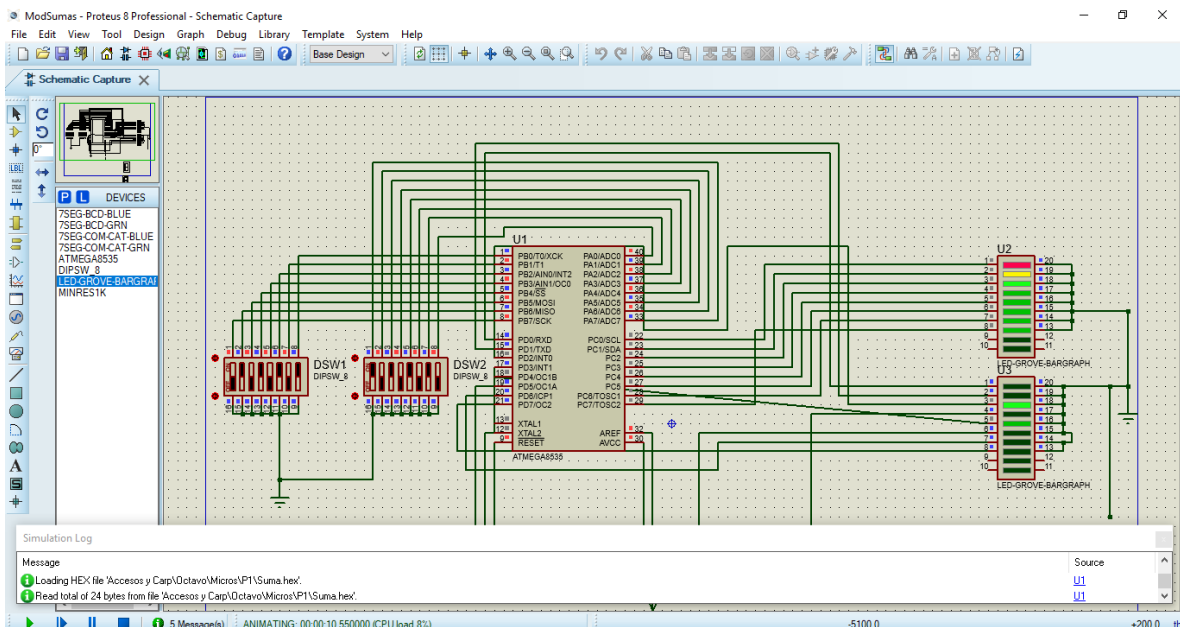
Esp:
rjmp Esp

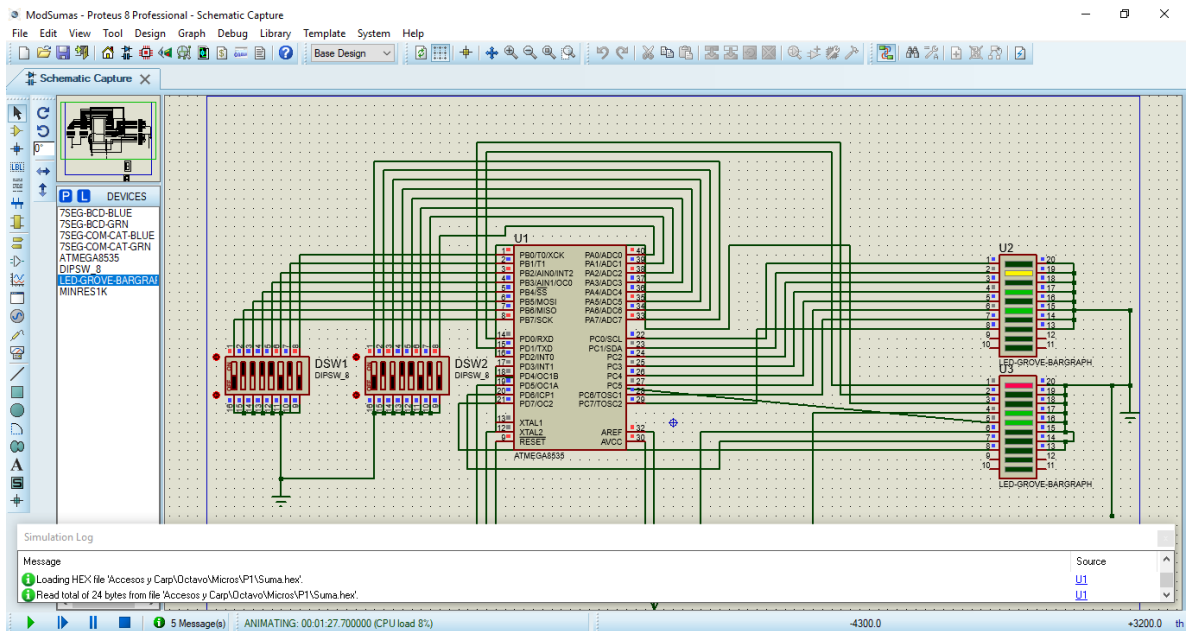
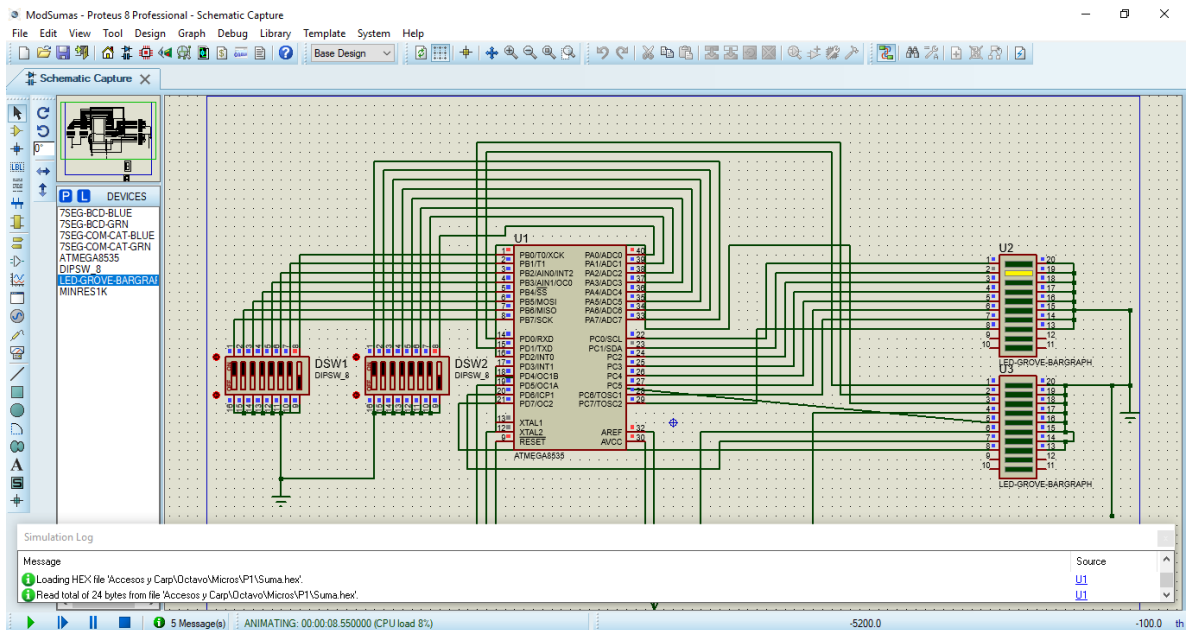
```

Imagen 2 Programa realizado para la práctica 1.

Simulaciones

A continuación, se muestra la simulación utilizada para la realización del programa (imágenes 3,4 y 5 como se puede apreciar los cambios que realiza el circuito son acordes a lo que se introduzca en el dip switch.





Imágenes 3, 4 y 5 Simulación del circuito realizado en la práctica 1, esto con el fin de asegurarse de que funcionara el programa acorde con el objetivo de la práctica.

Conclusiones

Daniel: En esta práctica fuimos capaces de realizar un programa que pudiese realizar una operación aritmética básica (suma) y mostrar dicho resultado haciendo uso del microcontrolador ATM8535.

David: Cuando realizamos una operación binaria internamente el microcontrolador activa o desactiva ciertas banderas que entre otras cosas ayudan a saber si hubo desbordamiento, si el número resultante es positivo o negativo, etc. Es gracias a estas banderas que el microcontrolador puede realizar diversas operaciones a partir de sumas.

Bibliografía

<https://es.wikipedia.org/wiki/Microcontrolador>

https://es.wikipedia.org/wiki/Unidad_central_de_procesamiento

[https://es.wikipedia.org/wiki/Memoria_\(inform%C3%A1tica\)](https://es.wikipedia.org/wiki/Memoria_(inform%C3%A1tica))

<http://www.ladelec.com/teoria/electronica-digital/401-suma-binaria>

Práctica 2

Objetivo: El objetivo de esta práctica es realizar un circuito capaz de llevar a cabo una cuenta cuyo intervalo de tiempo entre cada número sea definido por el usuario, esto se debe hacer utilizando el microcontrolador ATM8535.

Material

Para la realización de esta práctica se necesitan diversos materiales que ayudarán a la construcción del circuito, estos materiales son los siguientes:

- 1 microcontrolador ATM8535
- 1 pinzas de corte
- 1 dip switch de 8 interruptores
- 2 metros de cable del No. 14
- 2 barras led de 8 segmentos
- 1 protoboard
- 1 fuente de voltaje que pueda proveer al menos 5V.

Introducción teórica

El tiempo es una magnitud física con la que medimos la duración o separación de los acontecimientos. El tiempo permite ordenar los sucesos en secuencias, estableciendo un pasado, un futuro y un tercer conjunto de eventos ni pasados ni futuros respecto a otro. En mecánica clásica a esta tercera clase se llama presente y está formada por eventos simultáneos a uno dado.

Las formas e instrumentos para medir el tiempo son de uso muy antiguo, y todas ellas se basan en la medición del movimiento, del cambio de material de un objeto a través del tiempo, que es lo que puede medirse. Todos los relojes modernos desde la invención del reloj mecánico han sido contruidos con el mismo principio. Los microcontroladores utilizan y un reloj interno basado en este mismo principio, algunos funcionan a velocidad de reloj con frecuencias tan bajas como 4kHz, con un consumo de baja potencia. Por lo general, tendrán la capacidad de mantenerse a la espera de un evento como pulsar un botón o de otra interrupción; así el consumo de energía durante el estado de reposo puede ser sólo de nanowatts, lo que hace que muchos de ellos sean adecuados para aplicaciones con batería de larga duración. Otros microcontroladores pueden servir para roles de rendimiento crítico, donde sea necesario actuar más como un procesador digital de señal con velocidades de reloj y consumo de energía más altos.

Un microcontrolador difiere de una unidad central de procesamiento normal, debido a que es más fácil convertirla en una computadora en funcionamiento, con un mínimo de circuitos integrados externos de apoyo. La idea es que el circuito integrado se coloque en el dispositivo, enganchado a la fuente de energía y de información que necesite. Un microprocesador tradicional no le permitirá hacer esto, hay que agregarle los módulos de entrada y salida (puertos) y memoria para almacenamiento de la información. Los microcontroladores están diseñados para reducir el costo económico de un sistema en particular. Por eso el tamaño del CPU, la cantidad de memoria y los periféricos incluidos dependerán de la aplicación.

En esta práctica se construirá un circuito utilizando el microcontrolador ATM8535 que sea capaz de llevar una cuenta y cuya velocidad este controlada por el usuario al momento de mover determinados interruptores que le permitirán interactuar con el circuito, para esto es necesario crear un programa que sea capaz de leer las entradas, además de llevar la cuenta y mostrarla por medio de las barras led.

Desarrollo

Al igual que en la práctica anterior, la realización de esta práctica se dividió en dos segmentos, el primero el circuito físico el cual es prácticamente idéntico al de la práctica pasada. La segunda parte consiste en el programa realizado en AVR Studio el cual cambia totalmente con respecto al de la práctica pasada, siendo este el que hace que el funcionamiento del circuito cambie totalmente.

El programa que creamos para esta práctica se muestra en la imagen 1, empezamos por definir las entradas y salidas que se utilizaran, para esta práctica utilizaremos una variable auxiliar que nos ayudara a indicar nuestra salida.

Una vez definidas las entradas y salidas limpiamos la variable cta, para limpiar esta variable utilizamos la instrucción clr, a continuación, debemos de leer la entrada (no olvidemos que parte de la funcionalidad del circuito consiste en que el tiempo de retraso varíe según lo indicado por el usuario). Ya que terminamos de leer la entrada llamamos a la función delay la cual nos permite consumir el tiempo, si no agregáramos dicha función el cambio sería instantáneo y no tendría en cuenta el valor introducido por el usuario. La llamada a la función delay se hace utilizando la instrucción rcall, la cual nos permite llamar a una etiqueta que se encuentre en el programa.

Finalmente tenemos que hacer un decremento del valor ingresado hasta que sea cero, una vez que se vuelve cero decrementamos cta y saltamos a la etiqueta nvo, para realizar un salto debemos de usar el comando rjmp.

```
P2.asm

.include "m8535def.inc"
.def cta = r18 //Definimos registros
.def nvcs = r19
.def aux = r16
ldi aux, low(ramend)
out spl, aux
ldi aux, high(ramend)
out sph, aux
ser r16 //Usamos auxiliar para
out ddrc, r16 //Indicar nuestra salida
out portb, r16
clr cta //Limpiamos cta

nvo:
in nvcs, pinb //Leemos la entrada
out portc, cta //Sacamos la cuenta

nret:
rcall delay // decrementamos uno al valor ingresado
dec nvcs //hasta que sea 0 , volvemos a nret.
brne nret //Si ya es 0, incrementamos cta y saltamos al nvo.
inc cta
rjmp nvo

delay:
ldi r17, 0
ldi aux, 0
et1: dec r17 //Hacemos operaciones para consumir tiempo
brne et1
dec aux
brne et1
ret
```

Imagen 1 código creado para la realización de esta práctica.

En la imagen 2 podemos ver el circuito armado, en esta se puede apreciar el primer led encendido siendo este el inicio de la cuenta, a medida que la cuenta progresa los diversos leds que se tienen van encendiendo y apagando, en la imagen 3 podemos apreciar cómo han encendido ya varios leds luego de que pasa determinado tiempo.

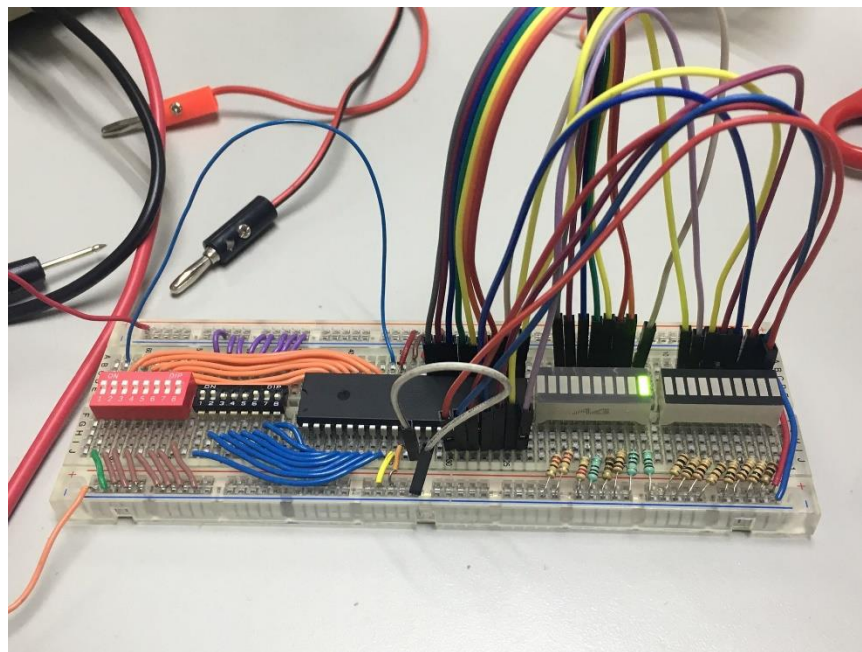


Imagen 2 Circuito armado, el cual empieza a contar.

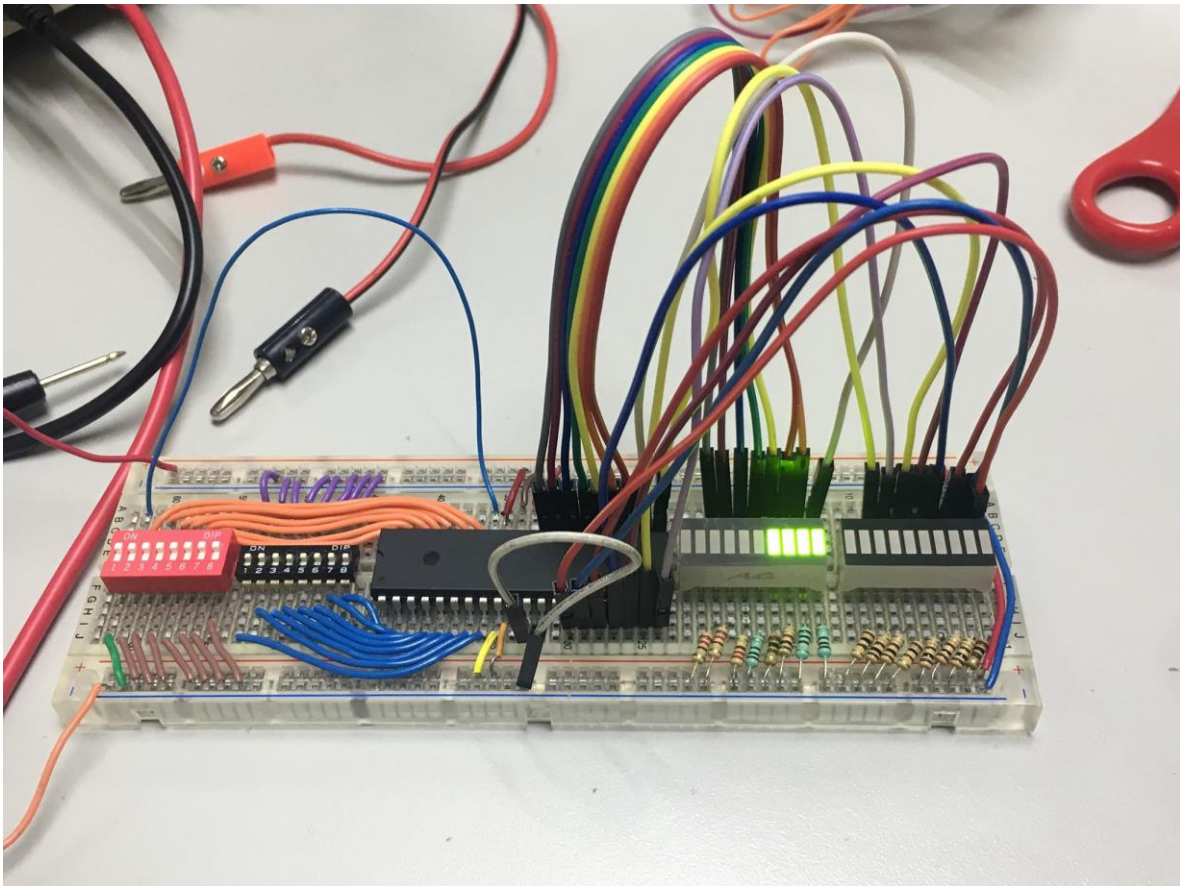
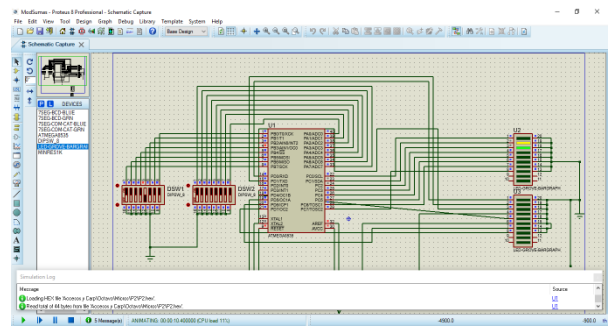
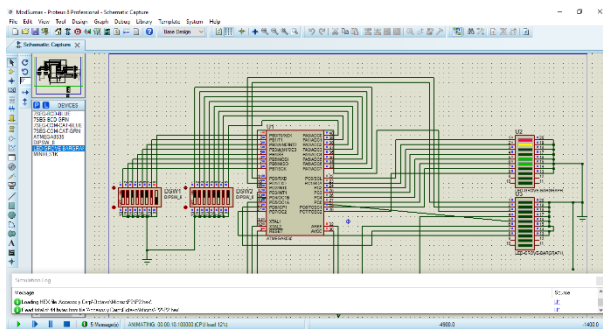


Imagen 3 Una vez que ha transcurrido cierta cantidad de tiempo el circuito enciende leds, acorde a la cuenta realizada.

Simulaciones

A continuación, se muestra la simulación del programa hecha en Proteus, en ambas imágenes (imágenes 4 y 5) se puede ver como los leds (parte derecha de la imagen) van cambiando acorde al tiempo que transcurre, es importante recordar que la velocidad de cambio está dada por las entradas que se introdujeron.



Imágenes 4 y 5 Simulación de la práctica 2

Conclusiones

Daniel: En esta práctica aprendimos a controlar la velocidad de cambio de las luces gracias a que podemos consumir tiempo haciendo que el microcontrolador se encuentre atrapado en un ciclo el cual estará definido por la cantidad de tiempo que se requiere, hay que tener en cuenta que existen herramientas que nos permiten introducir los datos para que de esta forma no tengamos que realizar el calculo.

David: En esta práctica logramos cumplir el objetivo al realizar un circuito el cual tuviese un cambio en la velocidad de las luces, esto se logro gracias a que dentro del programa que realizamos el microprocesador consumió tiempo al momento de entrar un ciclo cuyo número de repeticiones fue dado por los interruptores activados en el dip switch.

Bibliografía:

<https://es.wikipedia.org/wiki/Tiempo>

<https://es.wikipedia.org/wiki/Microcontrolador>

Práctica 3

Objetivo: Realizar un circuito que realice sumas utilizando dos dip switch, una vez que el resultado sea mostrado por medio de una barra de leds, utilizar una segunda barra de leds para mostrar el número total de operaciones que se han hecho.

Material

Para la realización de esta práctica se necesitan lo siguientes materiales:

- 1 microcontrolador ATM8535
- 1 pinzas de corte
- 1 dip switch de 8 interruptores
- 2 metros de cable del No. 14
- 2 barras led de 8 segmentos
- 1 protoboard
- 1 fuente de voltaje que pueda proveer al menos 5V.

Introducción Teórica

Un programa es un conjunto de procesos e instrucciones específicas que utiliza un computador o sistema de computadores pueda efectuar una tarea determinada, los programas son un elemento imprescindible para el normal funcionamiento de una computadora.

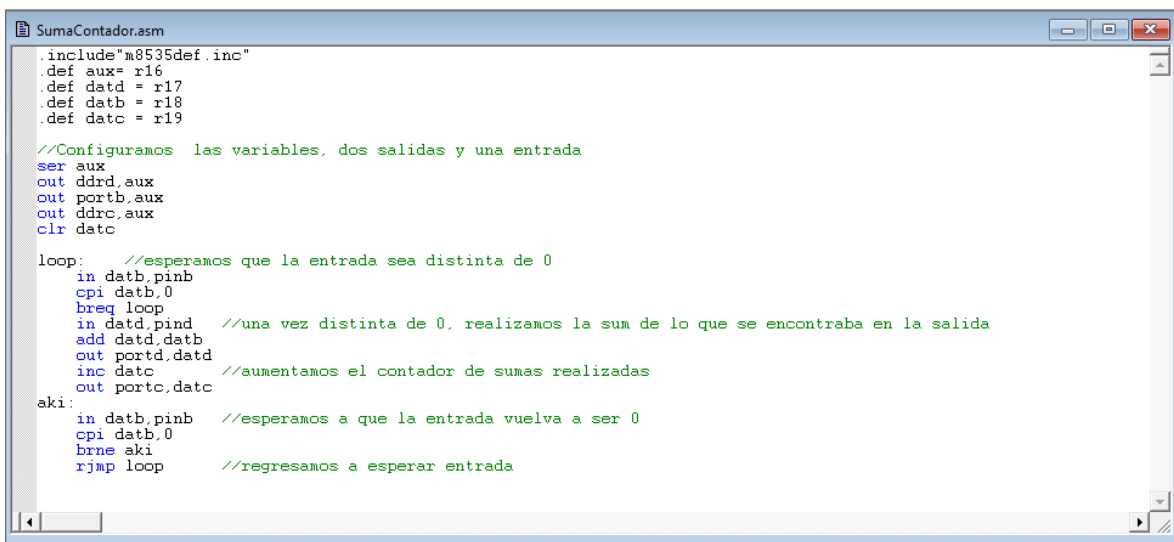
Por lo general, los programas de computador disponen de cierto margen de recursos del sistema informático mientras se ejecutan, muchos programas informáticos son capaces de actualizarse y modificarse a través de la descarga de datos de la internet, manteniéndose al día o redefiniendo sus componentes.

La palabra cálculo proviene del término latino calculus y se refiere a la cuenta, la enumeración o la pesquisa que se lleva a cabo mediante un ejercicio matemático, el cálculo consiste en un algoritmo que permite anticipar el resultado que procederá de ciertos datos que se conocen con anticipación. El conocimiento sobre los cálculos binarios es indispensable para el análisis y diseño de sistemas digitales, las operaciones más simples con las que se suelen trabajar son: la suma, resta multiplicación y división, además, es importante tener en cuenta que al momento de realizar este tipo de operaciones se requiere hacer uso de banderas las cuales nos ayudan por ejemplo a realizar cálculos cuando los operandos difieren de signo, si deseamos realizar una suma tenemos las siguientes banderas:

- La suma de cada bit corresponde al incremento en uno del código binario, cuando en una suma con signo, dos números positivos arrojan un resultado negativo, o dos números negativos arrojan un resultado positivo se dice que hubo desbordamiento u “overflow” este tiende a especificarse en hojas de datos con la letra V.
- Cuando existe acarreo en el último bit de la suma se le conoce como acarreo de salida o “carry out” este se representa como Co.
- Cuando el resultado de una operación es nulo utilizamos la letra Z para representarlo.
- Si el resultado es negativo, el bit de signo vale 1 de lo contrario este posee un valor de 0, cuando queremos representar el bit de signo utilizamos la letra N.

Desarrollo

En esta práctica se tuvo que realizar un código que ayudase al microcontrolador a realizar operaciones a partir de los valores introducidos en un dip switch, hay que tener en cuenta que para realizar la suma el valor de la entrada debe ser distinto de 0, por lo que el cambio no se verá reflejado hasta que el usuario introduzca una entrada. El código utilizado para el desarrollo de esta práctica se muestra en la imagen 1.



```

SumaContador.asm
#include "m8535def.inc"
.def aux = r16
.def datd = r17
.def datb = r18
.def datc = r19

//Configuramos las variables, dos salidas y una entrada
ser aux
out ddrd,aux
out portb,aux
out ddrb,aux
clr datc

loop: //esperamos que la entrada sea distinta de 0
in datb,pinb
cpi datb,0
brne loop
in datd,pind //una vez distinta de 0, realizamos la sum de lo que se encontraba en la salida
add datd,datb
out portd,datd
inc datc //aumentamos el contador de sumas realizadas
out portc,datc

aki:
in datb,pinb //esperamos a que la entrada vuelva a ser 0
cpi datb,0
brne aki
rjmp loop //regresamos a esperar entrada
  
```

Imagen 1 Código realizado para la práctica 3.

El primer paso consistió en declarar a la librería que nos sirve para utilizar el microcontrolador ATM8535, esto se hace con la línea de código `.include "m8535def.inc"`.

El siguiente paso para la realización de esta práctica es la definición y configuración de las variables, los nombres dados a todas estas son: aux, datd, datb y datc teniendo así dos salidas y una entrada. La razón por la que tenemos dos salidas es porque en este programa debemos de mostrar el resultado de la operación realizada, así como un contador que nos muestra cuantas operaciones se han realizado mientras el circuito a esta activo.

El siguiente paso es crear un ciclo que nos ayude a realizar todas las operaciones, primero debemos esperar a que la entrada sea distinta de cero, luego una vez que se a realizado el cambio de valor realizamos la suma de lo que se encontraba en la salida, a continuación, incrementamos en uno nuestro contador de sumas el cual esta representado por la variable datc, finalmente para poder realizar la siguiente operación, debemos esperar a que el valor de la entrada cambie nuevamente a cero, una vez hecho esto regresamos al inicio del ciclo para esperar que nuevamente el valor sea cambiado nuevamente.

Simulaciones.

La simulación realizada se muestra a continuación en las imágenes 2 y 3, en estas se puede ver que previamente se han realizado dos operaciones, esta es la razón por la que en una de las barras de led se puede ver que esta encendido el segundo led, al realizar nuevamente una operación el contador se incrementa en uno dando como resultado que prenda los dos primeros leds, ya que recordemos esta es la forma de representar el número 3 en binario.

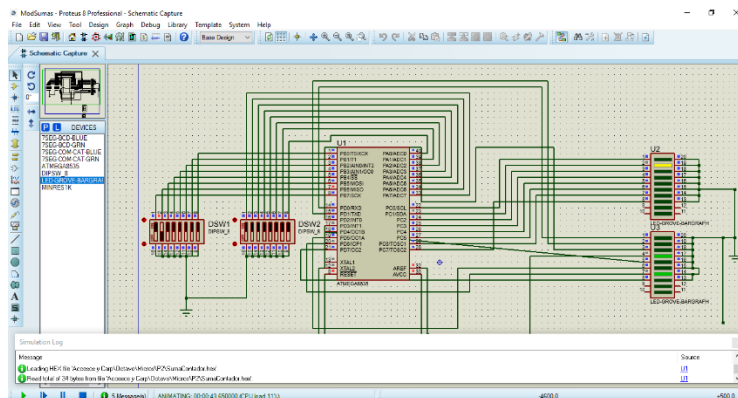


Imagen 2 Se han realizado dos operaciones previamente.

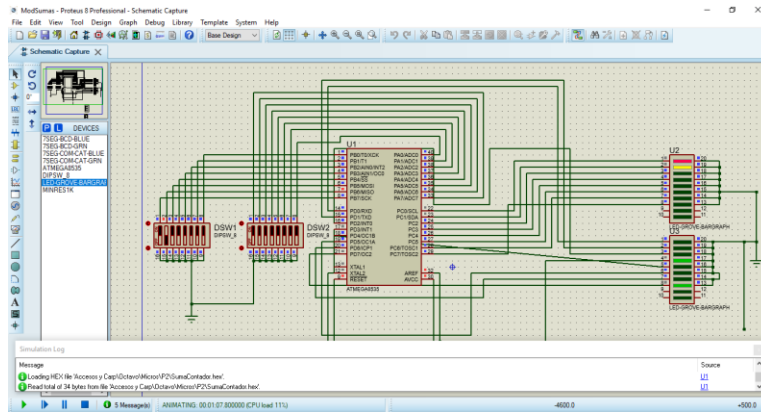


Imagen 3 podemos ver como en la barra de leds se muestra un cambio en el resultado de ambas barras.

Conclusiones

Daniel: Pudimos apreciar en este circuito como al momento de realizar una determinada operación el contador aumento, esto se logro gracias a que el programa que realizamos guardo la cuenta del número total de operaciones realizadas mientras el circuito este energizado.

David: A través del uso de dos salidas y una entrada pudimos realizar un circuito capaz de llevar a cabo una operación dado dos números, además de mostrar la cuenta total de operaciones realizadas, es importante recordar que cuando realizamos una operación se activan diversas banderas que ayudan a saber si el resultado es positivo o negativo.

Bibliografía

<https://concepto.de/programa-informatico/>

<https://www.significados.com/programa/>

<https://definicion.de/programa/>

<http://www.huergo.edu.ar/tcweb/pdf/APCap6.pdf>

<https://definicion.de/calculo/>

Práctica 4

Objetivo: Crear un programa que pueda llevar dos cuentas diferentes utilizando un mismo display, dichas cuentas deberán ser separadas dependiendo de si se suma un valor positivo o uno negativo, además de que en un segundo display deberá mostrarse el resultado obtenido de la operación.

Material

Para la realización de esta práctica se necesitan lo siguientes materiales:

- 1 microcontrolador ATM8535
- 1 pinzas de corte
- 1 dip switch de 8 interruptores
- 2 metros de cable del No. 14
- 2 barras led de 8 segmentos
- 1 protoboard
- 1 fuente de voltaje que pueda proveer al menos 5V.

Introducción

A pesar de que en la actualidad se puede encontrar números negativos representando diferentes situaciones en la vida cotidiana, se conoce que su aparición fue bastante posterior a la de otro tipo de números como por ejemplo los fraccionarios.

Un número negativo es cualquier número cuyo valor es menor que cero y, por tanto, que los demás números positivos, se utilizan para representar pérdidas, deudas, disminuciones o decrecimientos, entre otras cosas. Los números negativos se encuentran dentro de los números racionales, que a su vez forman parte de los números enteros.

Los números negativos son una generalización útil de los números positivos, cuando una magnitud o cantidad puede variar incrementalmente por encima o por debajo de un punto de referencia, usualmente representado por el cero.

Los números negativos con signo pueden sumarse, restarse, multiplicarse y dividirse. También pueden tomarse potencias con números negativos en la base o el exponente.

Para sumar dos números con signo, determinamos el signo y el valor absoluto del resultado del siguiente modo:

- Si ambos sumandos tienen el mismo signo: ese es también el signo del resultado, y su valor absoluto es la suma de los valores absolutos de los sumandos.

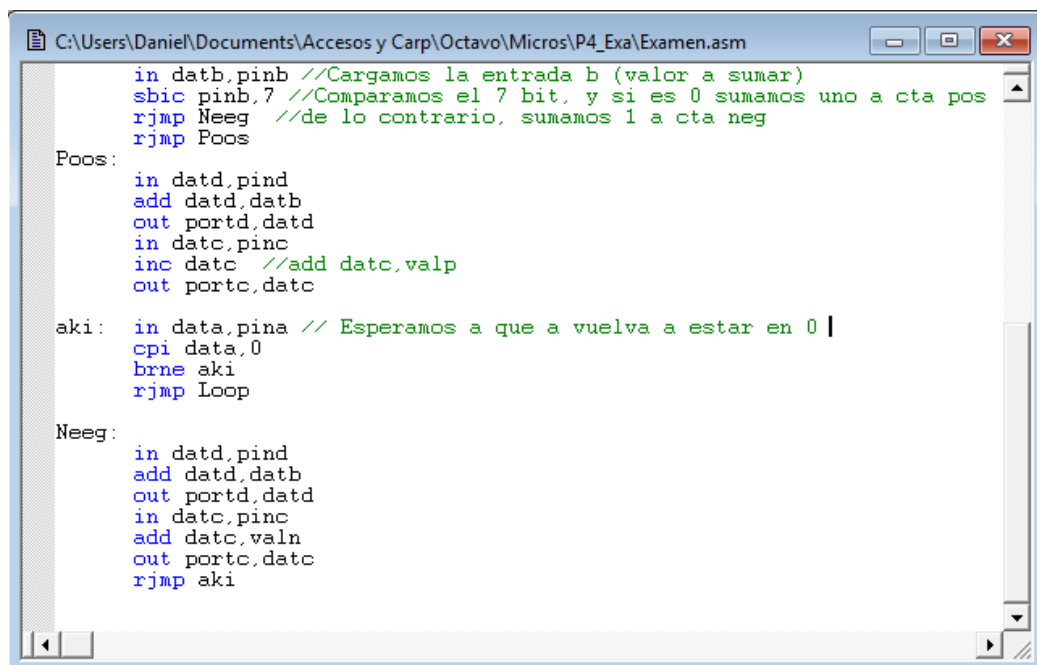
Si ambos sumandos tienen distinto signo:

- El signo del resultado es el signo del sumando con mayor valor absoluto.
- El valor absoluto del resultado es la diferencia entre el mayor valor absoluto y el menor valor absoluto, de entre los dos sumandos.

La suma o adición binaria es análoga a la de los números decimales. La diferencia radica en que en los números binarios se produce un acarreo (carry) cuando la suma excede de uno, mientras que en decimal se produce un acarreo cuando la suma excede de nueve.

Desarrollo

Para realizar esta práctica lo primero que hicimos fue escribir el programa que se muestra a continuación (imagen 1):



```

C:\Users\Daniel\Documents\Accesos y Carp\Octavo\Micros\P4_Exa\Examen.asm

in datb,pinb //Cargamos la entrada b (valor a sumar)
sbic pinb,7 //Comparamos el 7 bit, y si es 0 sumamos uno a cta pos
rjmp Neeg //de lo contrario, sumamos 1 a cta neg
rjmp Poos

Poos:
in datd,pind
add datd,darb
out portd,dard
in datc,pinc
inc datc //add datc, valp
out portc,darc

aki: in data,pina // Esperamos a que a vuelva a estar en 0 |
cpi data,0
brne aki
rjmp Loop

Neeg:
in datd,pind
add datd,darb
out portd,dard
in datc,pinc
add datc, valn
out portc,darc
rjmp aki
  
```

Imagen 1: Código realizado para la elaboración de la práctica 4.

Luego de declarar la librería que nos ayudaría a utilizar el microcontrolador ATM8535, así como las respectivas variables de entrada y salida, empezamos cargando el valor de la entrada que vamos a sumar, a este valor le llamamos b.

A continuación, tuvimos que comparar el valor del bit número 7, recordemos que será este bit el que nos indicará si estamos realizando una suma positiva o negativa, gracias a este bit también podemos saber de que lado del display deberá de aumentar el contador una vez que se a realizado la operación.

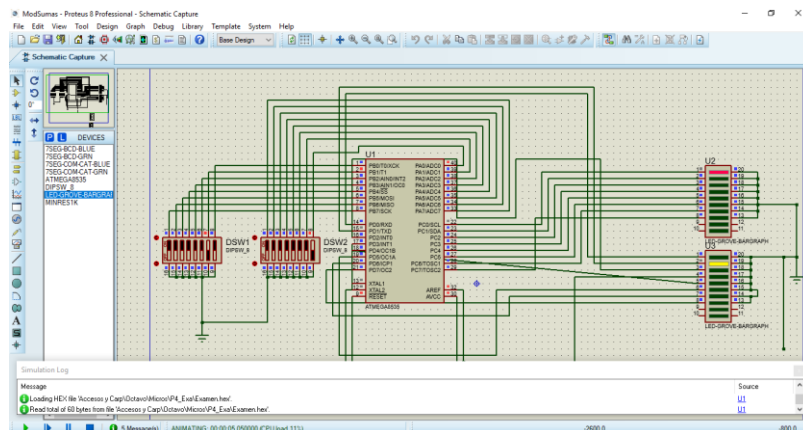
Si el valor del bit número 7 es 0 quiere decir que estamos frente a una operación positiva por lo que deberemos de saltar a la función que nos ayuda a aumentar el contador positivo, esta función en nuestro programa decidimos darle el nombre de Poos. En cambio, si el valor del bit número 7 es 1 quiere decir que la operación es negativa por lo que deberemos saltar a la función que nos permita aumentar el contador negativo, en este programa le hemos dado el nombre de Neeg.

Es importante mencionar que en ambas funciones para poder realizar el salto utilizamos la instrucción rjmp, a la cual le debemos de indicar la etiqueta a la que debe de saltar el programa.

Una vez que hemos realizado la cuenta debemos de volver a esperar a que nuevamente se cambie el valor a cero antes de realizar la próxima operación, es por ello que en el caso de la función Neeg utilizamos nuevamente el comando rjmp para saltar a la etiqueta que esta verificando que el valor sea cambiado nuevamente a 0, pero antes de realizar el salto tanto en la función Poos como Neeg debemos de mostrar el resultado de la operación hecha, y así hemos concluido el programa.

Simulaciones

Las siguientes imágenes muestran las simulaciones hechas para esta práctica, en ellas se puede ver como se van realizando operaciones en donde el bit 7 puede tener el valor de uno o de cero y dependiendo de dicho valor el contador va aumentando su valor ya sea del lado derecho o izquierdo, del mismo modo podemos ver como se muestra en la segunda barra de leds el resultado de las operaciones realizadas.



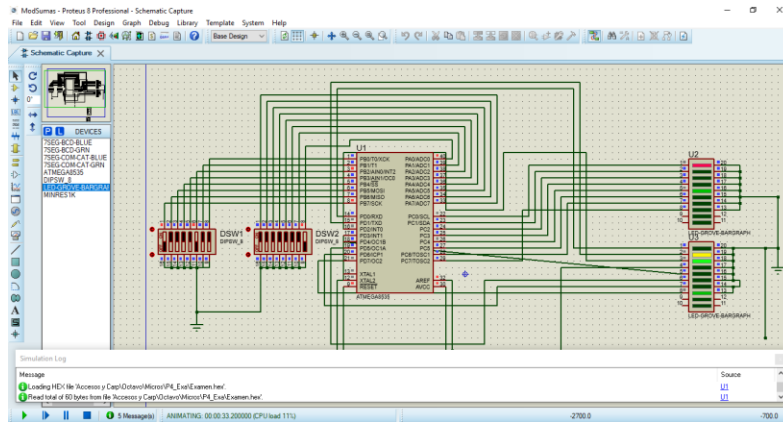


Imagen 1 y 2 se incrementa la cuenta del contador cada que se realiza una nueva operación.

Es importante mencionar que para cualquier operación que hagamos primero debemos de asegurarnos que el valor de entrada se encuentre en cero ya que si no pasa esto entonces no se verán reflejados los cambios en las barras de leds antes cuando hagamos la operación.

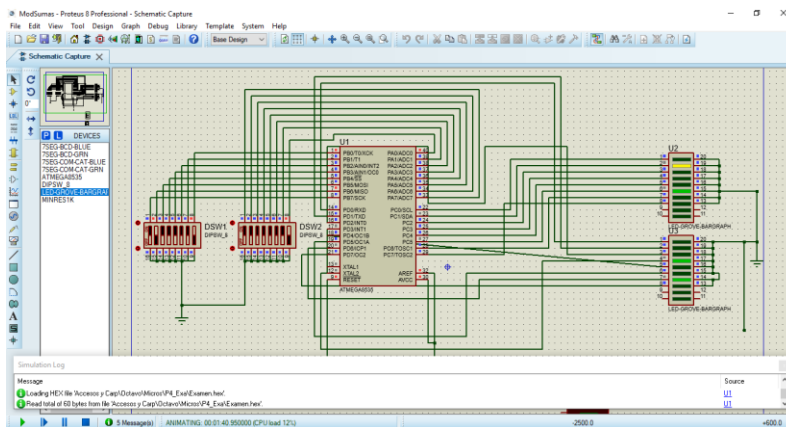
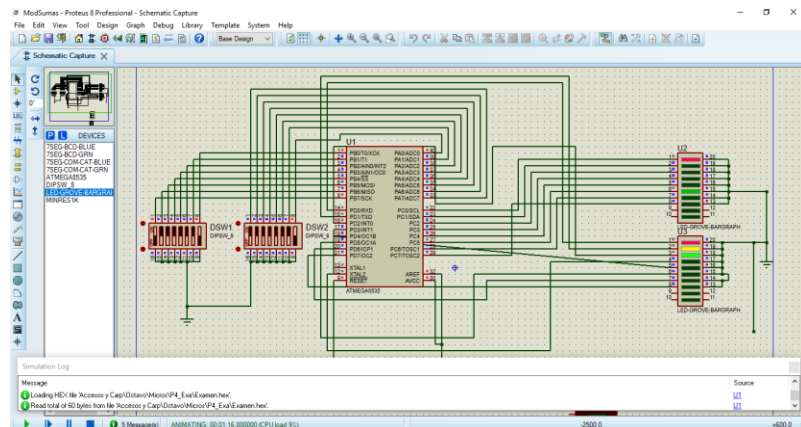


Imagen 3 y 4 de la práctica 4.

Conclusiones

Daniel: En esta práctica aprendimos que las operaciones que podemos realizar cuando utilizamos el sistema decimal también las podemos hacer en binario, sin embargo, al momento de trabajar con un número negativo debemos de usar un bit que nos ayude a saber el signo de ambos operandos, pues si este difiere entonces debemos de realizar un proceso diferente al de la suma ordinaria.

David: Podemos concluir de esta práctica que al momento de realizar operaciones binarias es muy importante tener en cuenta las banderas activadas independientemente de si estamos realizando una suma, resta multiplicación o división estas banderas nos resultaran de gran ayuda puesto a que entre otras cosas nos pueden indicar si el resultado es de valor negativo o positivo, si tuvo desbordamiento o no, entre otras cosas.

Bibliografía

https://es.wikipedia.org/wiki/N%C3%BAmero_negativo

https://www.ecured.cu/N%C3%BAmeros_negativos

<https://es.plusmaths.com/aritmetica/numeros/negativos>