Universidad Tecnológica del Perú

Facultad de Ingeniería Electrónica y Mecatrónica

Curso	:	Microcontroladores
Profesor	:	Domínguez Jesús
Práctica Dirigida Nro	:	1
Tema	:	Programación de contadores en PIC18F4550
Integrantes	:	Espinoza Valera, Jesús Alberto Francisco Alberto Francisco, Jaime Félix Guerrero Isuiza, Mateo
Fecha del Experimento	:	Lunes, 3 de Septiembre del 2018
Hora	:	De 9:30 am a 2:00 pm
Fecha de entrega del informe	:	Martes, 4 de Septiembre del 2018
Hora	:	De 8:00 am a 6:00 pm

2018 - III

UNIVERSIDAD TECNOLÓGICA DEL PERÚ

Contenido

Introducción 3

Capítulo 1: Compendio Teórico 4

1.1. Lenguaje C 4

1.2. MPLAB 4

1.3. PicKit 4

1.4. Copilador XC8 5

1.5. Microcontrolador 5

Capítulo 2: Resultados Obtenidos 7

2.1. Contador Módulo 10 7

2.1.1. Diagrama de flujo ..... **¡Error! Marcador no definido.**

2.1.2. Código 7

2.1.3. Lenguaje y Simulación 7

2.1.4. Circuito Real 8

2.2. Contador Definido por Switchs 9

2.2.1. Diagrama de flujo ..... **¡Error! Marcador no definido.**

2.2.2. Código 9

2.2.3. Lenguaje y simulación 10

2.2.4. Circuito Real 11

2.3. Contador Especial controlado por Switchs 12

2.3.1. Diagrama de flujo ..... **¡Error! Marcador no definido.**

2.3.2. Código 12

2.3.3. Lenguaje y simulación 13

2.3.4. Circuito Real 14

Link del video 16

Observaciones 17

Conclusiones 18

Bibliografía 19

Introducción

El interés del grupo de investigación se basó a partir de la observación de distintos programas que se realizaron en las clases de Microcontroladores. En consecuencia, nos planteamos la interrogante: ¿De qué manera se puede realizar la programación de un Pic de manera física en fin de realizar un proceso en particular? Consideramos que el método de programación mediante el software de PicKit 3 en adición con el MPLAB es el más efectivo para escribir un programa en el PIC 18F4550. De este modo, se ha estructurado el informe en cuatro partes.

En primer lugar, se presenta el marco teórico donde se define el tipo de lenguaje a usar, una breve descripción del circuito usado en los PIC y un concepto básico de los programas usados para lograr este informe. En segundo lugar, se desarrolla la experimentación con distintos programas en el PIC18F4550 .En tercer lugar, se mencionan las recomendaciones ofrecidas por el grupo que desarrolló la investigación, para mejorar el proceso del informe. Finalmente, en la conclusión se afirma que se debe escribir el lenguaje el en programa MPLAB y luego se debe simular en Proteus antes de programar el PIC para evitar que este último se dañe y escribir el programa en el PIC mediante el PickIT 3. Aunque es mediante el método teórico que se obtiene el programa indicado para los determinados procesos del informe.

Capítulo 1: Compendio Teórico

1.1. Lenguaje C

El lenguaje C, es un lenguaje que revoluciono ya que debido a su sencillez, y a su tamaño, pero que sobre todo no está dentro de una aplicación en específica, lo hace más potente. C trabaja con tipos de datos que son directamente tratables por el hardware de la mayoría de computadoras actuales, como son los caracteres, números y direcciones. Estos tipos de datos pueden ser manipulados por las operaciones aritméticas que proporcionan las computadoras.

1.2. MPLAB

Este es un programa que sirve para editar distintos micro controladores que este mismo programa puede soportar, sin embargo una de las cosas importantes es que se puede grabar circuitos integrados. Asimismo cabe recalcar que se debe a empezar a escribir el programa, respetando la directiva y/o parámetros necesarios para compilarlo y grabarlo al chip. Definimos directivas como palabras específicas o palabras que han sido separadas para ordenarle al Mplab que funciones debe configurar a la hora de compilar nuestro programa.

1.3. PicKit

El pickit hace posible la programación de micro controlador utilizando todo lo que incluye en el armado del código en el Mplab, desarrollado por la empresa Microchip. Este pickit es muy fácil de conectar ya que solo se conecta mediante una interfaz USB.

Ahora, una vez definido cada instrumento usado en este proyecto, nosotros hicimos parte de armar todo lo mencionado. Aquí veremos cómo es que mediante el uso del lenguaje C se hizo la programación, y la compilación para generar el código, seria parte del primer proceso para implementar a lo que es

físico, tomando en cuenta que mientras se hacia el primer procedimiento, se tenía que estar implementando el circuito en el protoboard, de manera tal que cumpla con lo que se requiere como salida. Entonces en el proceso de uso de ciertas herramientas usamos el PicKit como un mediador ante lo que ya era simulado y lo que era real para plasmar todo lo que se requería

A continuación se verá cada una de las partes que se ha dado a expresar a manera de introducción, explicando paso a paso según a de desarrollarse.

1.4. Copilador XC8

Un compilador es un programa informático que permite que el lenguaje avanzado usado por el programador (Lenguaje C, BASIC, etc) sea traducido al lenguaje de máquina, el único que puede leer el microcontrolador. El compilador XC8 es uno de los compiladores de la línea de compiladores MPLAB XC de Microchip, el cual es compatible con los microcontroladores PIC de 8 bits, utilizado en este proyecto.

1.5. Microcontrolador

Un microcontrolador es un circuito integrado que en su interior contiene una unidad central de procesamiento (CPU), unidades de memoria (RAM y ROM), puertos de entrada y salida y periféricos. Estas partes están interconectadas dentro del microcontrolador, y en conjunto forman lo que se le conoce como microcomputadora. Se puede decir con toda propiedad que un microcontrolador es una microcomputadora completa encapsulada en un circuito integrado.

Inicialmente todos los microcontroladores adoptaron la arquitectura clásica de Von Neumann, la cual se caracteriza por disponer de una sola memoria principal donde se almacenan datos e instrucciones de manera

indistinta. Después, se empezó a realizar microcontroladores con la nueva arquitectura Harvard, la cual se caracteriza por disponen de dos memorias independientes para datos y para instrucciones, permitiendo accesos simultáneos.

Estos microcontroladores necesitan ser programados para poder realizar las funciones. Debido a la gran dificultad de programar los microcontroladores en el código binario, la programación comúnmente se lleva a cabo a través de un lenguaje de alto nivel, es decir, un lenguaje que utilice frases o palabras semejantes o propias del lenguaje humano. Lenguajes como el C, utilizado en nuestro caso, o BASIC son comúnmente utilizados en la programación de los microcontroladores.

Capítulo 2: Resultados Obtenidos

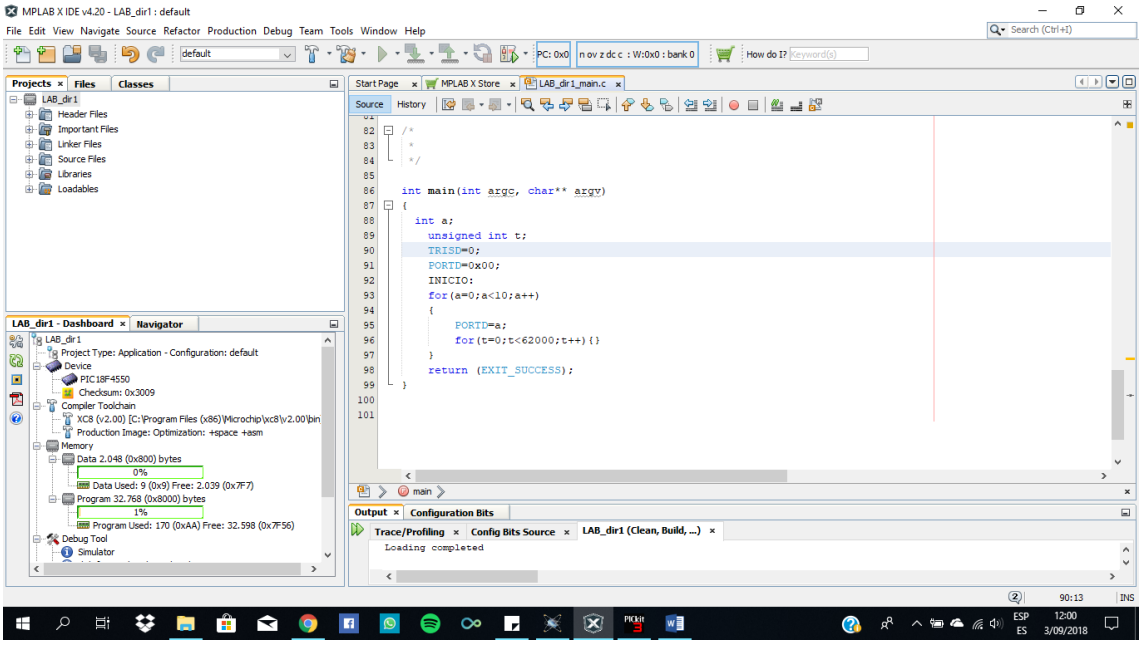
2.1. Contador Módulo 10

2.1.1. Código

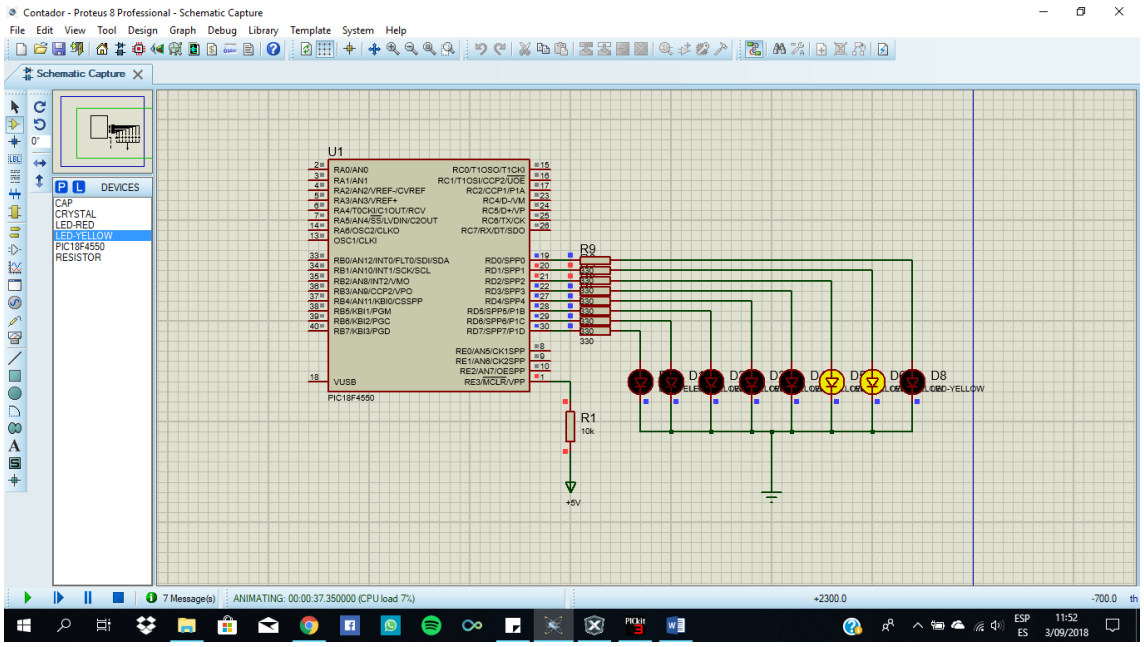
```
int main (int argc, char** argv)
{
    int a;                          // Definir a como entero
        unsigned int t;             // Definir t como entero sin símbolo
        TRISD=0;                    // Definir puerto D como salida
        PORTD=0x00;                 // Definir la salida como 0
        INICIO:                     // Definir etiqueta INICIO
        for (a=0;a<10;a++)          // Establecer  for como contador
        {
                PORTD=a;            // Establecer  el valor de salida como a
                for (t=0;t<62000;t++) { }   // Usar for como retraso en el contador
        }
        return (EXIT_SUCESS);
}
```

2.1.2. Lenguaje y Simulación

A continuación se muestra el código escrito en MPLAB V4.20:

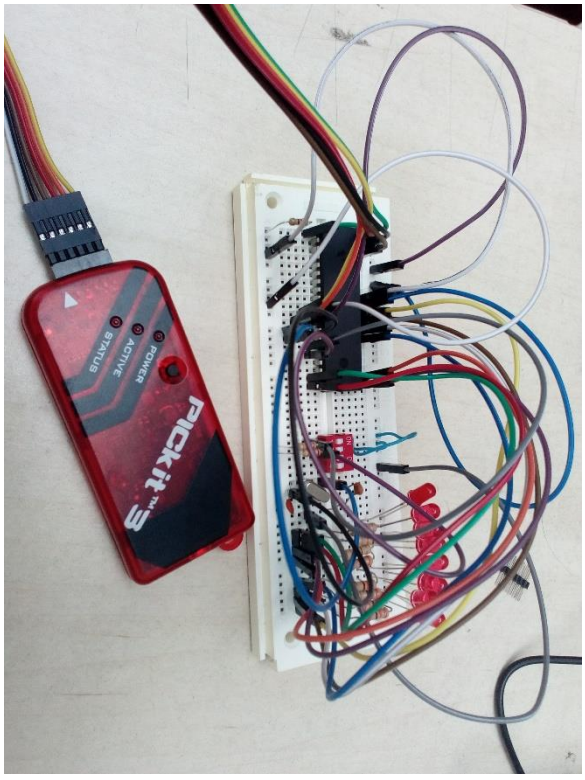


Del mismo modo, a continuación se muestra la simulación del circuito construido en Proteus 8:

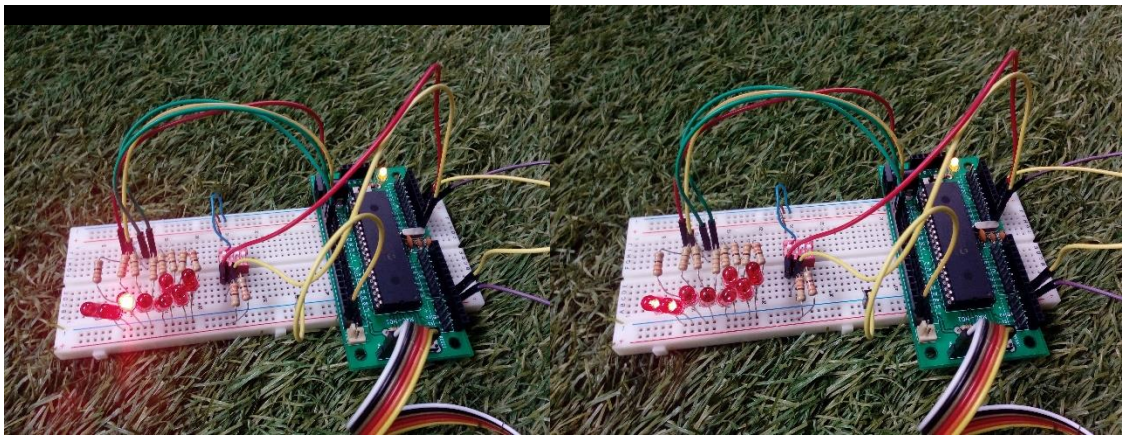


2.1.3. Circuito Real

Para la primera simulación se realizó un circuito en protoboard conectado a base de la simulación del Proteus y el manual del PIC18F4550:



Sin embargo, el cristal que usamos se encontraba defectuoso y se usó una base que contenía ya el circuito ya ensamblado, como se muestra a continuación:



2.2. Contador Definido por Switchs

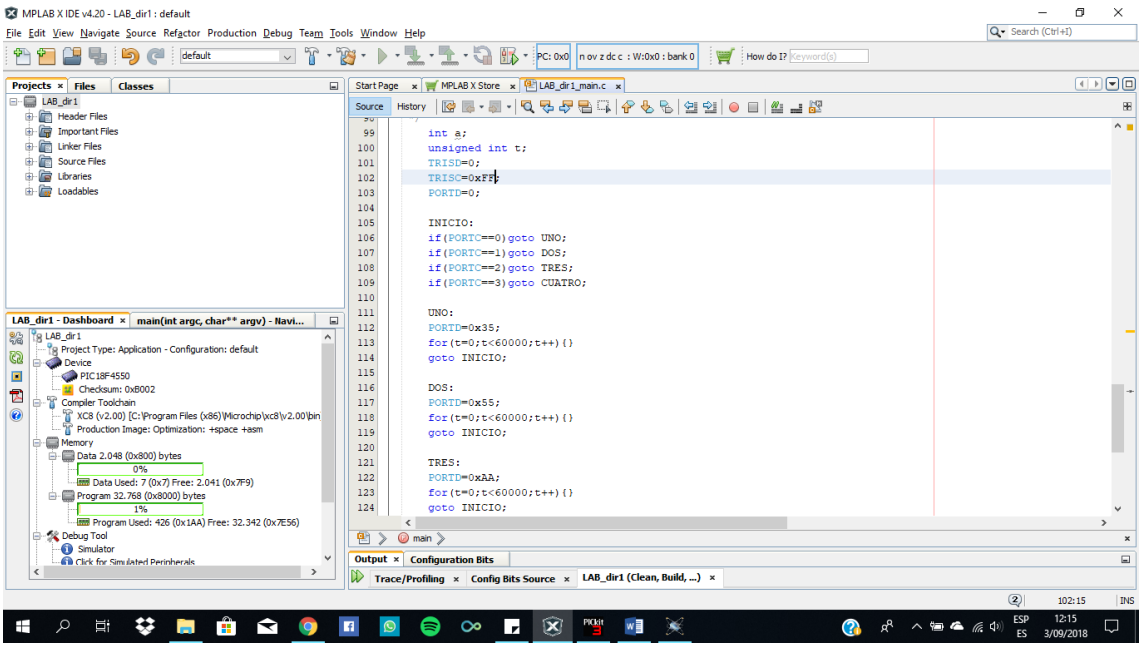
2.2.2. Código

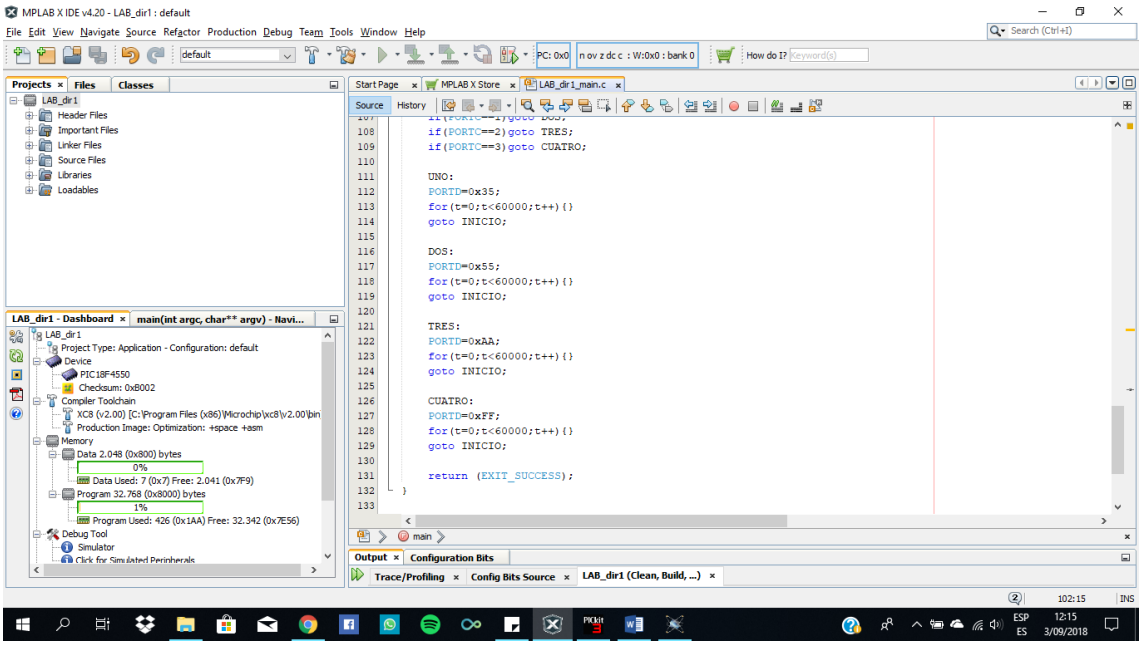
```
int main (int argc, char** argv)
    {
        int a;                          // Definir a como entero
        unsigned int t;                 // Definir t como entero sin símbolo
        TRISD=0;                        // Definir puerto D como salida
        TRISC=0xFF;                     // Definir puerto C como entrada
        PORTD=0x00;                     // Definir la salida como 0

        INICIO:                         // Definir etiqueta INICIO
        if (PORTD==0) goto UNO;         // Establecer primera condicional
        if (PORTD==1) goto DOS;         // Establecer segunda condicional
        if (PORTD==2) goto TRES;        // Establecer tercera condicional
        if (PORTD==3) goto CUATRO;      // Establecer cuarta condicional

        UNO:                            // Definir etiqueta UNO
        PORTD=0x35;                     // Establecer  el valor de salida 35
        for (t=0;t<60000;t++) { }       // Usar for como retraso
        goto INICIO;                    // Volver a la etiqueta INCIO
```

```
DOS:                        // Definir etiqueta DOS
PORTD=0x55;                 // Establecer  el valor de salida 55
for (t=0;t<60000;t++) { }   // Usar for como retraso
goto INICIO;                // Volver a la etiqueta INCIO

TRES:                       // Definir etiqueta TRES
PORTD=0xAA;                 // Establecer  el valor de salida 1010
for (t=0;t<60000;t++) { }   // Usar for como retraso
goto INICIO;                // Volver a la etiqueta INCIO

CUATRO:                     // Definir etiqueta CUATRO
PORTD=0xFF;                 // Establecer  el valor de salida 1515
for (t=0;t<60000;t++) { }   // Usar for como retraso
goto INICIO;                // Volver a la etiqueta INCIO

return (EXIT_SUCESS);
}
```

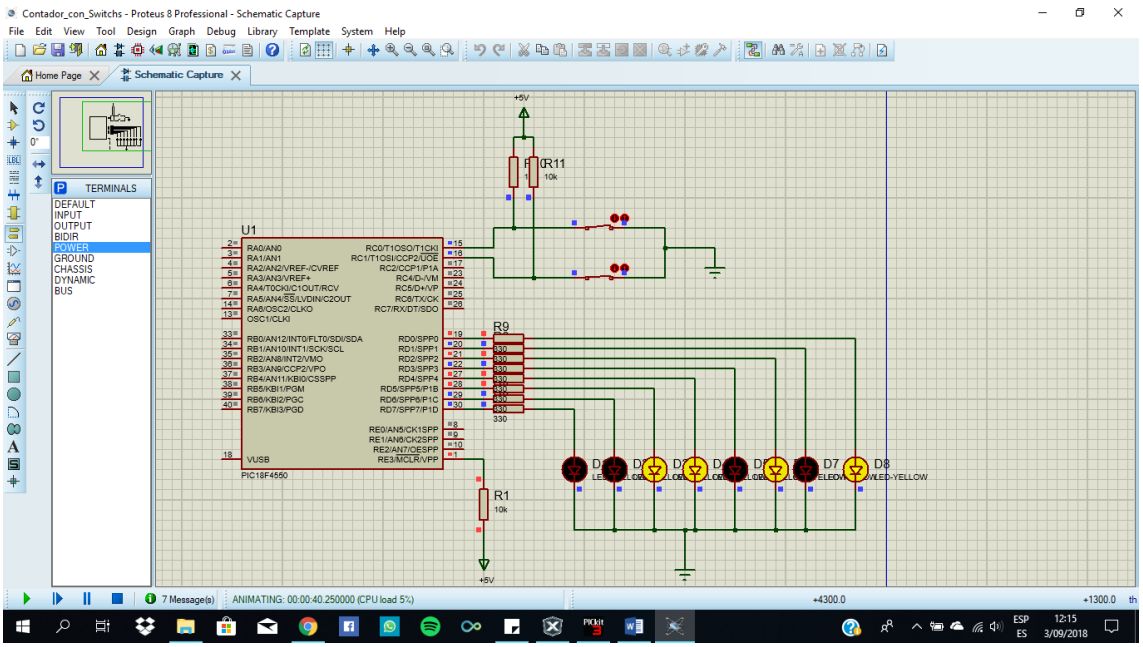
2.2.3. Lenguaje y simulación

A continuación se muestra el código escrito en MPLAB V4.20:



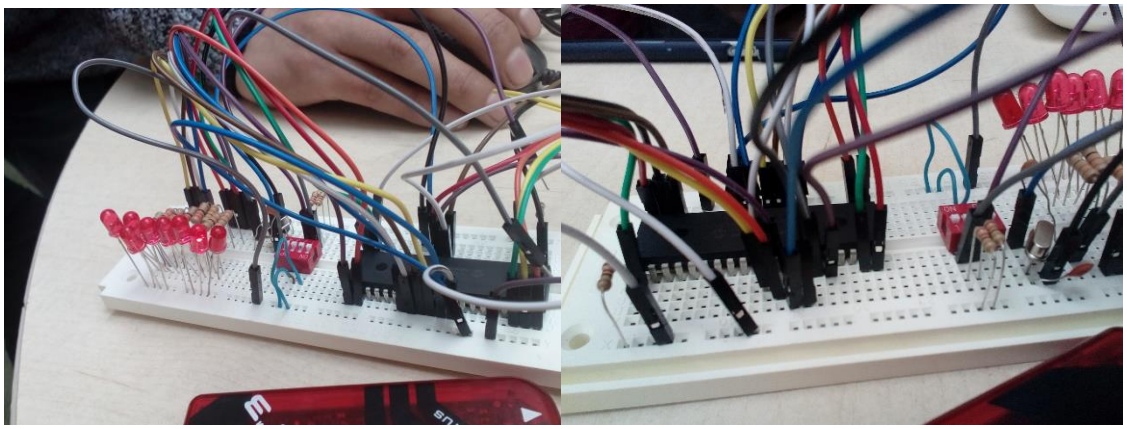


Del mismo modo, a continuación se muestra la simulación del circuito construido en Proteus 8:



2.2.4. Circuito Real

A continuación se muestra el mismo circuito que el que se realizó en la simulación anterior, con la diferencia de que este ya se encontraba reparado y posee un cristal de 4Mhz en total funcionamiento:



2.3. Contador Especial controlado por Switchs

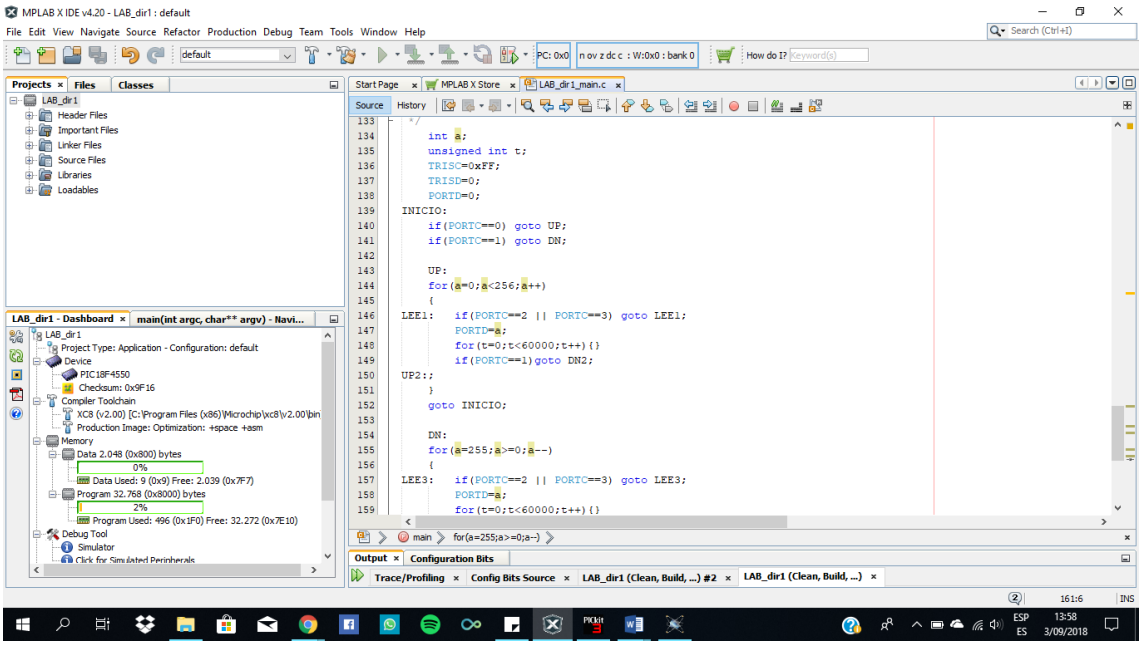
2.3.2. Código

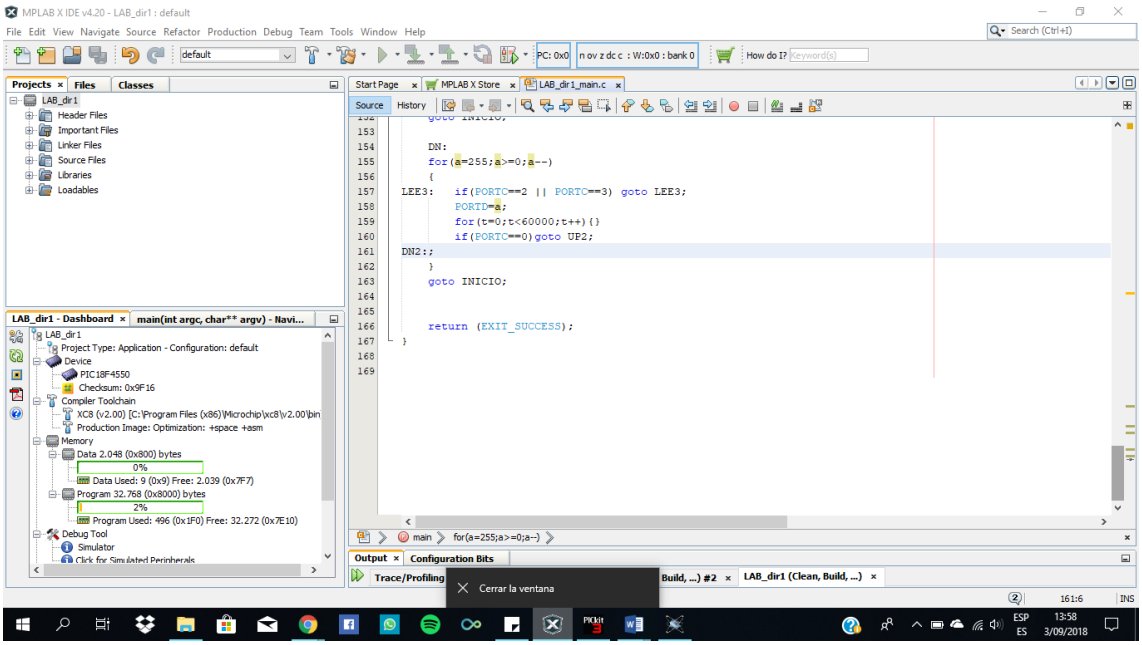
```
int main (int argc, char** argv)
        {
          int a;                        // Definir a como entero
          unsigned int t;               // Definir t como entero sin símbolo
          TRISC=0xFF;                   // Definir puerto D como salida
          TRISD=0;                      // Definir puerto C como entrada
          PORTD=0;                      // Definir la salida como 0
INICIO:                                 // Definir etiqueta INICIO
          if (PORTC==0) goto UP;        // Establecer primera condicional
          if (PORTC==1) goto DN;        // Establecer segunda condicional

          UP:                           // Definir etiqueta UP
          for (a=0;a<256;a++)           // Establecer  contador ascendente
          {
LEE1:                                   // Definir etiqueta LEE1
          if (PORTC==2 || PORTC==3) goto LEE1; // Establecer condicional
          PORTD=a;                      // Establecer  el valor de salida como a
          for (t=0;t<60000;t++) { }     // Usar for como retraso
          if (PORTC==1) goto DN2;       // Establecer segunda condicional
UP2:;                                   // Definir etiqueta UP2
```

```
        }
        goto INICIO;                // Volver a la etiqueta INCIO

        DN:                         // Definir etiqueta DN
        for (a=255;a>=0;a--)        // Establecer  contador descendente
        {
LEE3:                               // Definir etiqueta LEE3
        if (PORTC==2 || PORTC==3) goto LEE3; // Establecer condicional
        PORTD=a;                    // Establecer  el valor de salida como a
        for (t=0;t<60000;t++) { }   // Usar for como retraso
        if (PORTC==0) goto UP2;     // Establecer segunda condicional
DN2:;                               // Definir etiqueta DN2
        }
        goto INICIO;                // Volver a la etiqueta INCIO

  return (EXIT_SUCESS);
}
```

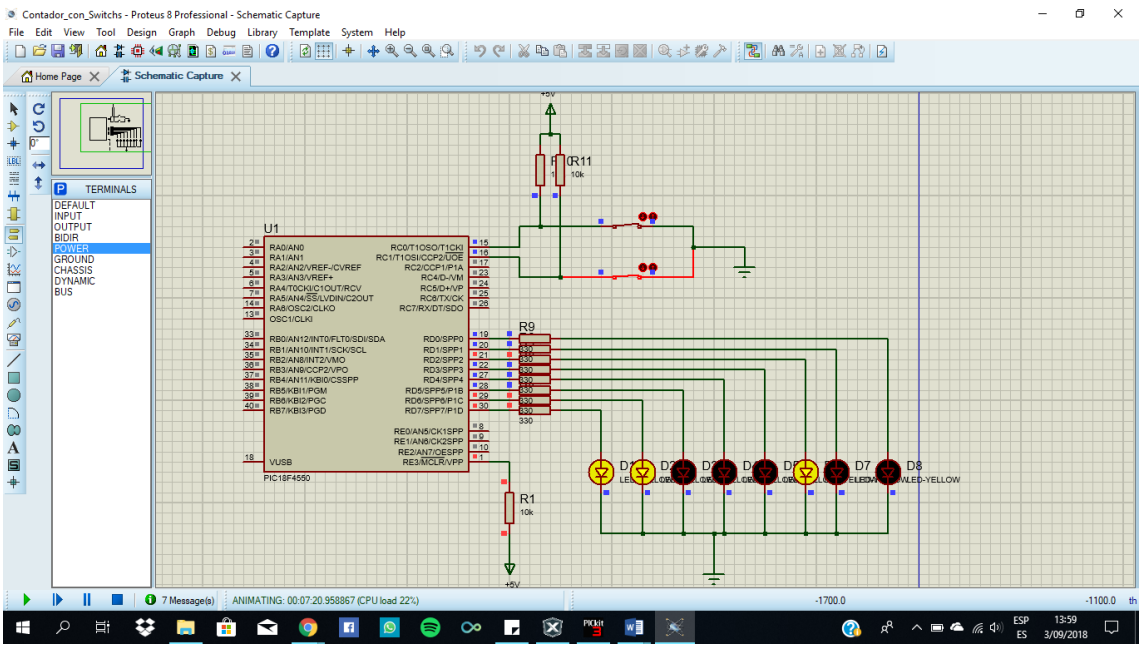
2.3.3. Lenguaje y simulación

A continuación se muestra el código escrito en MPLAB V4.20:





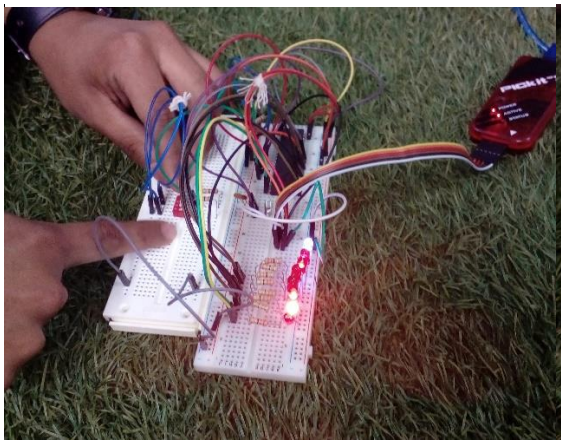
Del mismo modo, a continuación se muestra la simulación del circuito construido en Proteus 8:

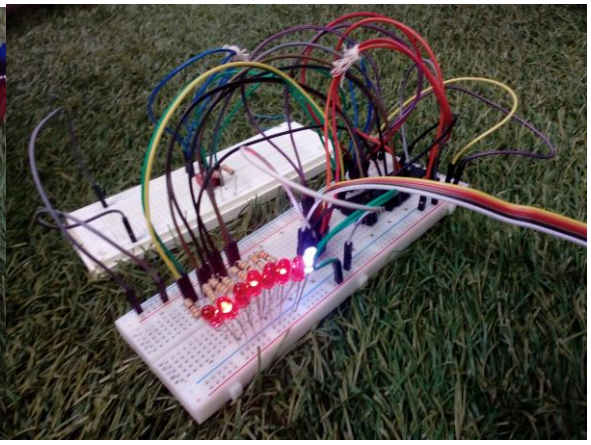


2.3.4. Circuito Real

A continuación se muestra el mismo circuito que el que se realizó en la simulación anterior:

UNIVERSIDAD
TECNOLÓGICA
DEL PERÚ





UNIVERSIDAD TECNOLÓGICA DEL PERÚ

Link del video

https://youtu.be/JQPup2

SGAKE

UNIVERSIDAD TECNOLÓGICA DEL PERÚ

Observaciones

- En nuestro caso, se quemó el cristal resonador durante la prueba del primer ejercicio y tuvimos que comprar otro para la realización de los otros dos.
- El tiempo de prendido de los leds eran muy cortos generando que no se pueda apreciar el buen funcionamiento del PIC, por ello, se cambió la definición de la variable “t” de “int” a “unsigned int” para poder ampliar este retraso.

Conclusiones

- La configuración de los bits del PIC18F4550, en MPLABX, es esencial en la simulación ya que de hacerlo de manera correcta nos la facilitara, pero haciéndolo mal generará un error.
- El cristal resonador es un componente muy frágil que al quemarse genera que no funcione el micro controlador.
- Se logró crear, compilar y simular el proyecto, en lenguaje C, usan MPLABX y el compilador XC8.
- Es posible escribir el programa en el PIC18F4550 mediante el software de PicKit3.
- Es fundamental escribir cada detalle del lenguaje C del programa de manera precisa para evitar errores.
- Es fundamental configurar los bits del programa en fin de que se adapten al circuito simulado y real en el cual se apicara dicho programa.

UNIVERSIDAD TECNOLÓGICA DEL PERÚ

Bibliografía

- MICROCHIP (2018) (https://www.microchip.com/) Sitio web oficial de Microchip; contiene información acerca del compilador XC8 y de MPLABX.
- ELECTRONICA ESTUDIO (2018) (http://www.electronicaestudio.com/index.htm) Sitio web oficial de Electrónica estudio; contiene información acerca de los microcontroladores.
- WIKIPEDIA (2018)(https://es.wikipedia.org/wiki/Wikipedia:Portada) Sitio web oficial de Wikipedia; contiene información de interés.