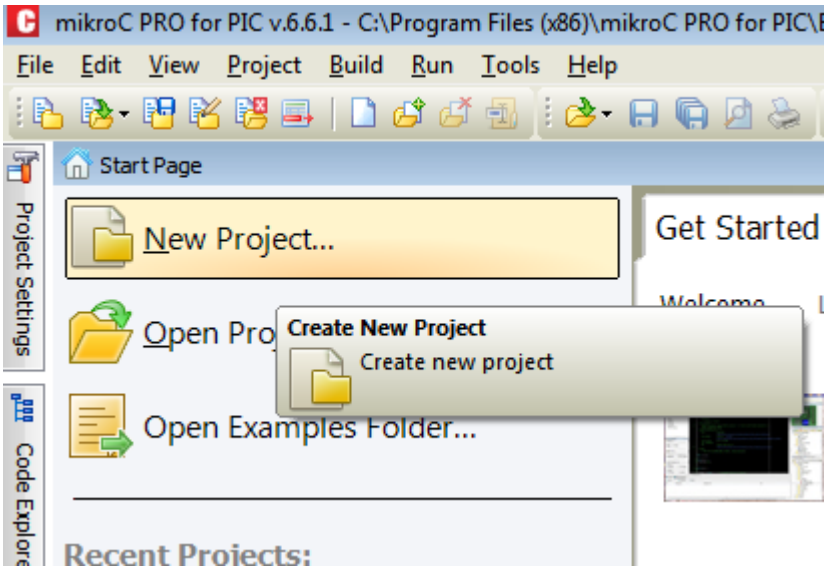
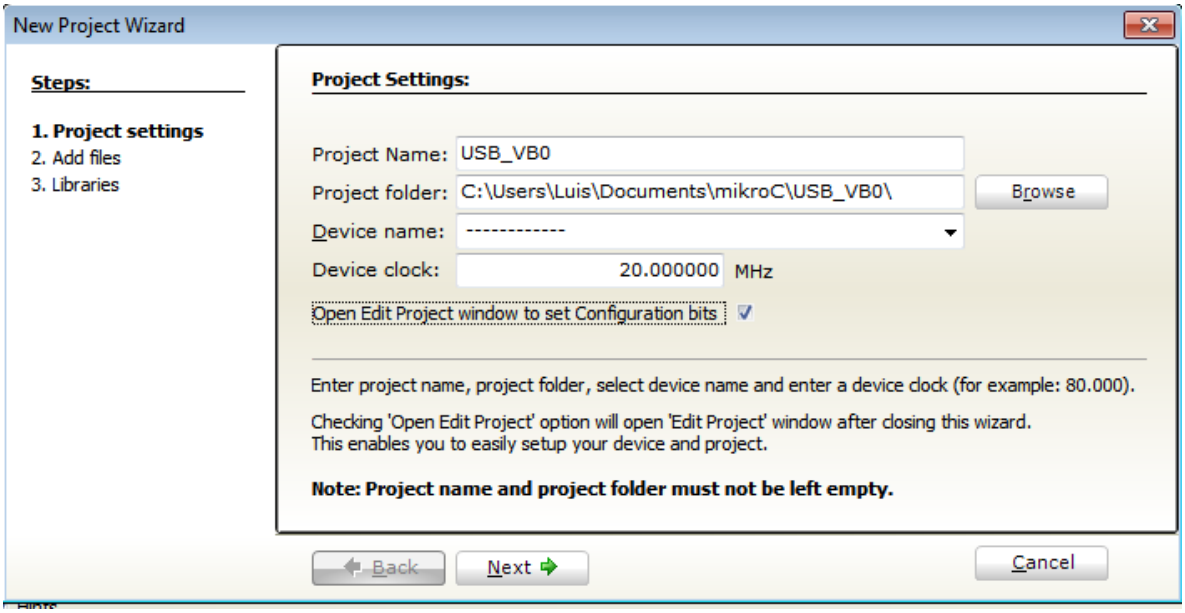
Creación de un nuevo de Proyecto.

Para la creación de un nuevo proyecto y habilitación de la comunicación USB, se comienza por dar click en el botón de New Project al ejecutar mikroC for PIC como se ilustra en la siguiente captura:

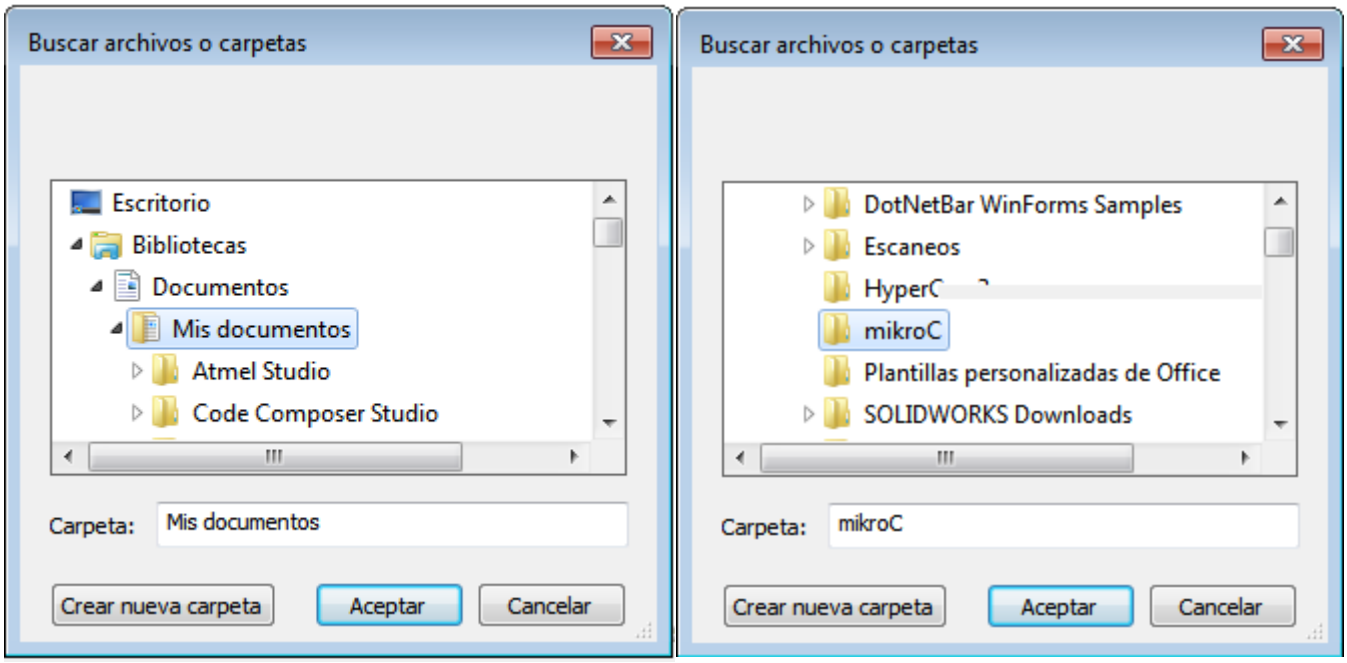


Se despliega la ventana de los ajustes del Proyecto como lo muestra la imagen siguiente:



Se observa que se tiene que asignar un nombre al proyecto y que se tiene que guardar en una carpeta, se sugiere que sea en una creada por el usuario en la sección Mis documentos, puesto que el compilador asigna una carpeta por default en la dirección donde se instaló el programa.

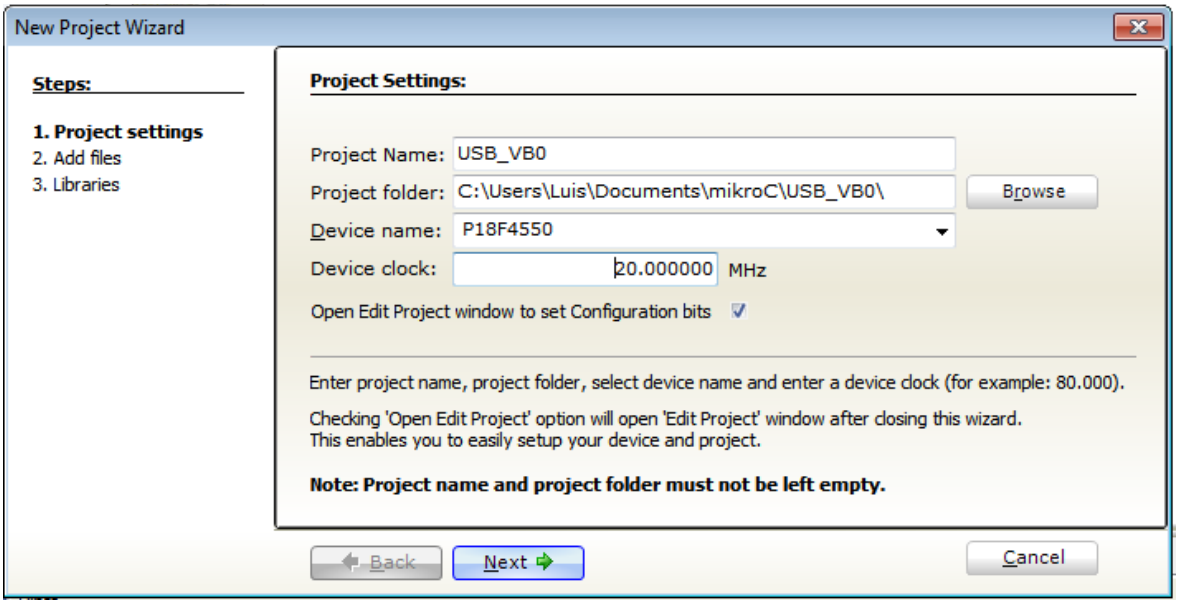
En la imagen subsecuente se observa cómo se puede asignar la localización del proyecto dando click en 'Browse'.Se puede crear nuevas carpetas para la fácil localización del Proyecto.



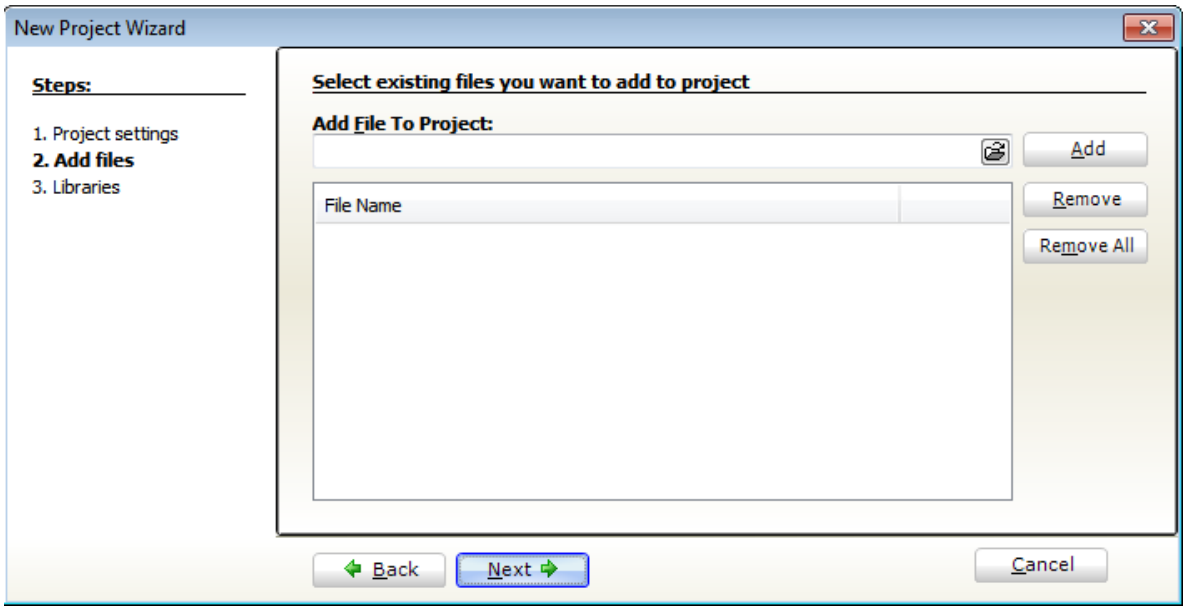
En la opción de Device Name, se necesita buscar el microcontrolador a ocupar, en este caso es el PIC18F4550, abriendo la pestaña se tiene que buscar manualmente como P18F4550.

Se asigna una frecuencia de reloj de 20 MHz y se selecciona la casilla para abrir el editor de Configuración de Bits.

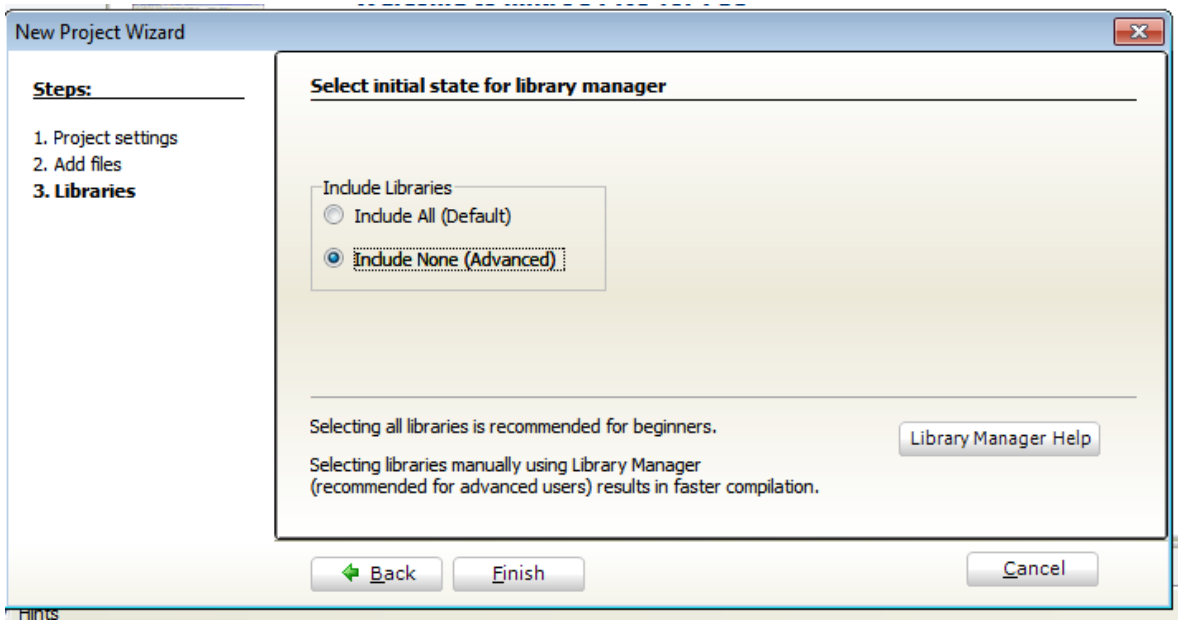
Finalmente se selecciona el botón next. Una ilustración de los pasos anteriores se muestra a continuación.



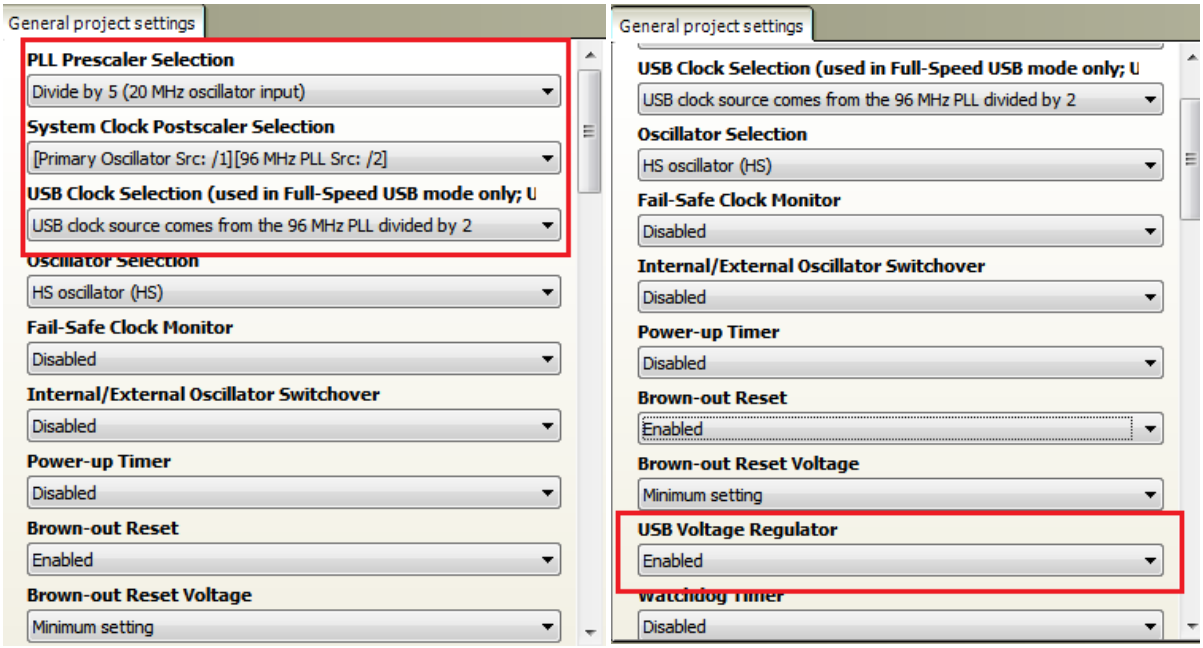
La siguiente ventana nos permite incluir otros archivos al proyecto, en este caso no se incluirá alguno y se da click en el botón Next como se ilustra continuación:



La ultima ventana del asistente, permite incluir o no todas las librerías, se selecciona no incluir ninguna y se oprime en el botón Finish como se ejemplifica en la ilustración subsecuente.

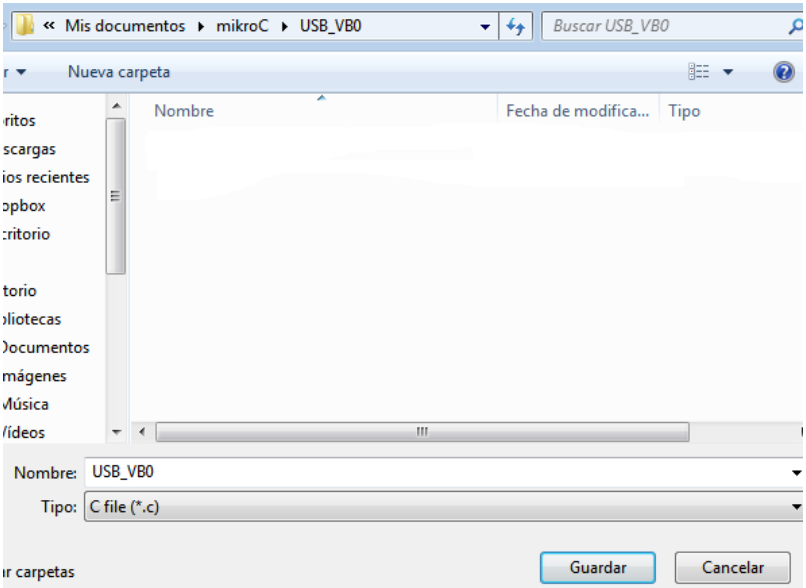


A continuación la ventana del Editor de Configuración de Bits aparecerá, aquí se selecciona los parámetros para obtener una frecuencia de trabajo de 48MHz que, según el fabricante Microchip, necesita trabajar el modulo USB y el microcontrolador. En la siguiente captura se muestran las casillas necesarias para realizar estos ajustes.



Finalmente se da click en el botón OK del editor.

Antes de comenzar con la programación es importante guardar el archivo con extensión .c en la carpeta que recién creamos, se ejemplifica en la siguiente imagen.

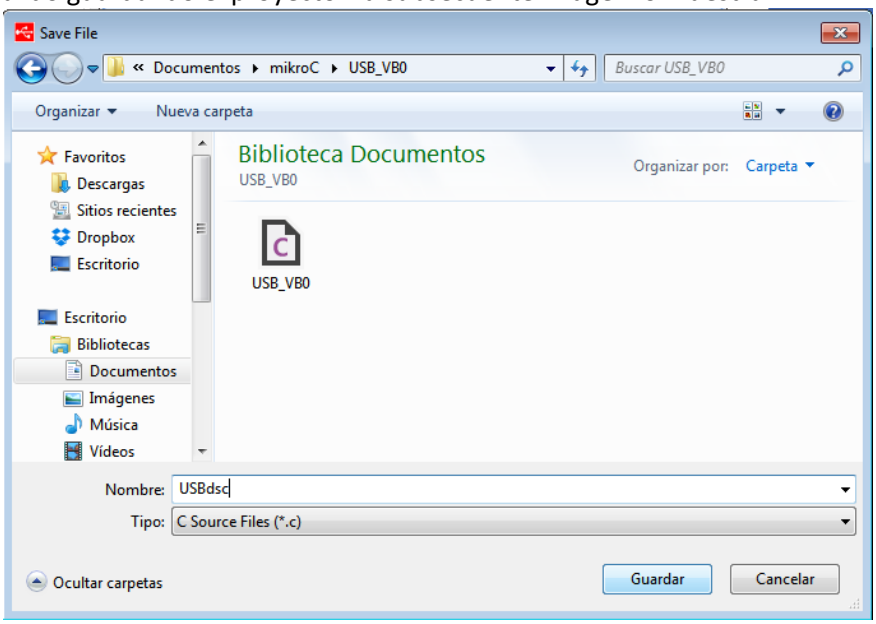


De la pestaña Tools, se elige la opción HID Terminal donde se despliega una ventana como sigue:

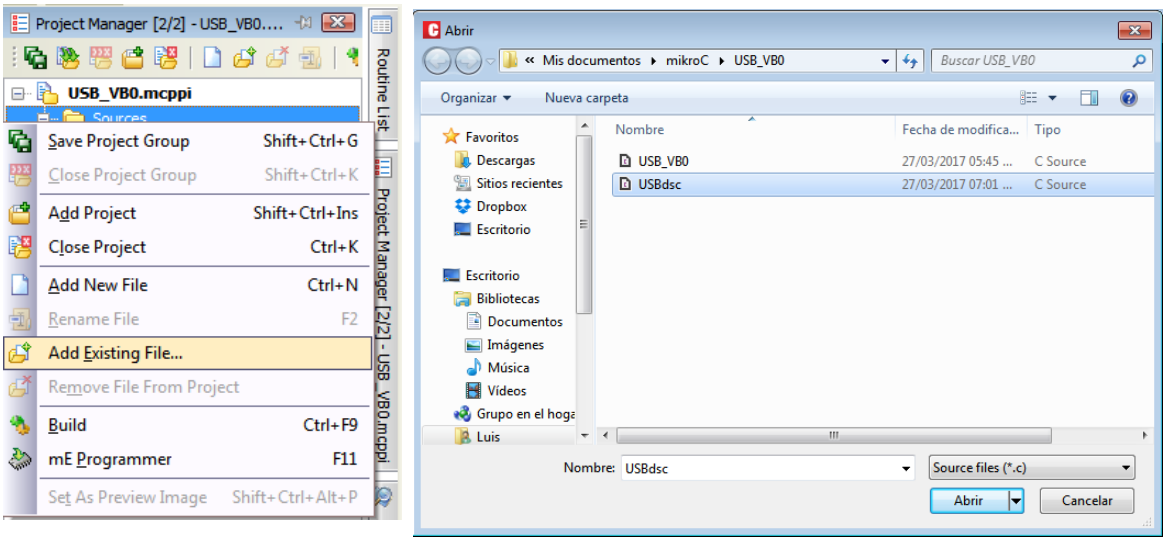
El descriptor define los parámetros que tendrá la interfaz, en la parte superior izquierda, están los números de identificación del Producto y Fabricante, y que, en la plantilla son utilizados para leer el dispositivo.

Las casillas de entrada y salidas de longitud de informe son el tamaño máximo (en bytes) de los buffer, es decir de la memoria de almacenamiento para transferir datos.

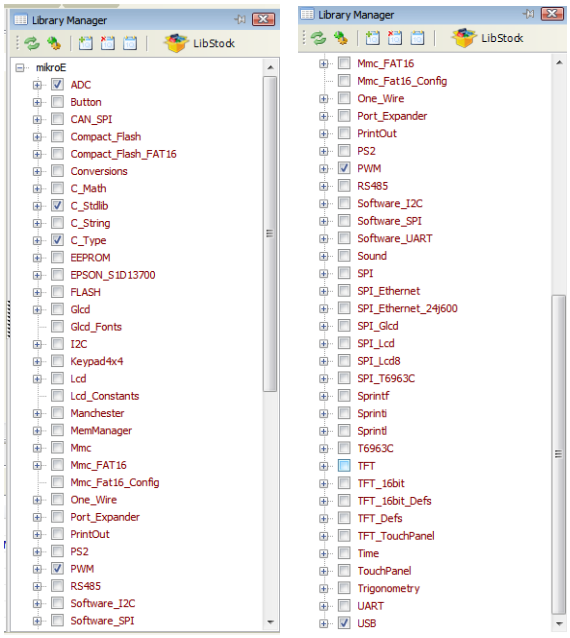
En la parte de abajo se asigna el Nombre del fabricante/desarrollador y el nombre del producto. Se oprime el botón 'Save descriptor' y se abrirá el explorador de Windows, donde se escoge la carpeta donde se ha ido guardando el proyecto. La subsecuente imagen lo muestra.



Se cierra la herramienta USB HID Terminal. Si se pasa el cursor encima de la pestaña Project Manager ubicada en la parte derecha de la pantalla se despliega los archivos ocupados en el proyecto. Es necesario que en la carpeta Source se oprima el botón derecho del mouse y se adhiera un archivo existente, en este caso el descriptor USB recién creado, en el explorador de Windows se selecciona, como la imagen que se ilustra a continuación.



Ahora, si se pasa el cursor por la Pestaña Library Manager se muestran las librerías disponibles propias del compilador mikroC, para no adherir todas por eso se elegirán manualmente, las cuales son las siguientes: ADC,C_Stlib,C_Type,PWM, y USB. En la siguiente captura se ejemplifica.



Implementación de la comunicación USB en el compilador MikroC

El primer paso es asignar el tamaño de los buffers y asignarles una dirección en la memoria RAM del microcontrolador, para esto se toma como referencia la figura 17-5 de la hoja de especificaciones del microcontrolador. Nótese que la dirección fijada para la descripción de los buffers es a partir del número Hexadecimal 400 (a partir de ahora con el prefijo h) el cual es 1024 en decimal y los datos del USB tienen una dirección asignada a partir de 500h (1280 en decimal).

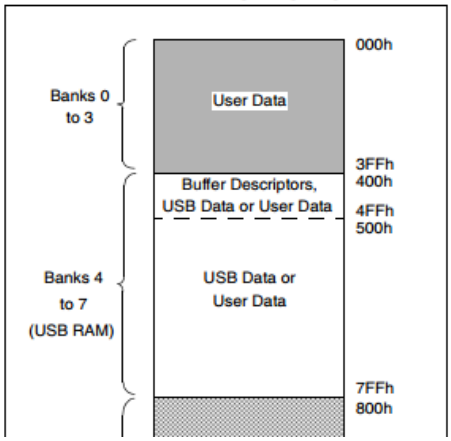
Si la longitud de los informes de entrada y salida son de 64 bytes entonces:

1280+64= 1344 (540h).

Es por tanto que ***los buffers de lectura y escritura del puerto USB se asignaran a las direcciones 500h y 540h.***

Por último y como medida de prevención, se asigna un espacio reservado para los descriptores USB que va del 400h al 500h, la cual es una diferencia de 100h (256 en decimal o 256 bytes).

FIGURE 17-5: IMPLEMENTATION OF USB RAM IN DATA MEMORY SPACE



Subsiguientemente, es importante la revisión de la sección de Ayuda del compilador, en la parte de USB Library se observa cómo se asignan las direcciones propuestas, el ejemplo se basa en el mismo microcontrolador como se ilustra en la captura siguiente:

Library Example

This example establishes connection with the HID terminal that is active on the PC. Upon connection establishment, the HID Device Name will appear in the respective window. After that software will wait for data and it will return received data back. Examples uses `USBdsc.c` descriptor file, which is in the same folder, and can be created by the HID Terminal.

Copy Code To Clipboard

```
unsigned char readbuff[64] absolute 0x500;   // Buffers should be in USB RAM,
unsigned char writebuff[64] absolute 0x540;
```

Con un arreglo de tipo char (que tiene un tamaño de 1 byte) sin signo y la directiva 'absolute' seguido de la dirección asignada.

La directiva 'absolute' es propia del compilador MikroC como indica la siguiente captura.

Directive absolute

Directive `absolute` specifies the starting address in RAM for a variable or a starting address in ROM for a constant. If the variable or constant is multi-byte, higher bytes will be stored at the consecutive locations.

Directive `absolute` is appended to declaration of a variable or constant :

```
// Variable x will occupy 1 byte at address 0x22 :
short x absolute 0x22;

// Variable y will occupy 2 bytes at addresses 0x23 and 0x24 :
int y absolute 0x23;
```

Seguidamente se inicializa la función de interrupción de servicio, es otra función propia del compilador y en la sección de ayuda, marca que se manda a llamar y es inamovible.

Si se sigue revisando el código ejemplo se observa otras funciones que deben ser implementadas como son:

HID_Enable. Recibe las asignaciones de memoria de Lectura y Escritura.

HID_Read. Recibe datos de la computadora y los lee. Si falla la lectura regresa un cero.

HID_Write. Escribe datos al búfer de escritura, es decir, envía datos a la computadora. Si la transmisión falla, regresa un cero.

En la siguiente captura tomada de la sección de ayuda vemos el ejemplo de implementación del código.

```
void main(void){
  ADCON1 |= 0x0F;
  CMCON  |= 7;

  HID_Enable(&readbuff,&writebuff);

  while(1){
    while(!HID_Read())
      ;

    for(cnt=0;cnt<64;cnt++)
      writebuff[cnt]=readbuff[cnt];

    while(!HID_Write(&writebuff,64))
      ;
  }
}
```

Se propone un código similar para la implementación de la comunicación USB como sigue:

```
unsigned char Leer_USB[64]      absolute 0x500;
unsigned char Escribir_USB[64]  absolute 0x540;
unsigned char Reservar_USB[256] absolute 0x400;

void interrupt(){
  USB_Interrupt_Proc();               // USB servicing is done inside the interrupt
}

  char cnt;
  char dato;

void main() {

HID_Enable (&Leer_USB,&Escribir_USB);      // Enable HID communication
while(1){

 dato= Hid_Read();
  if(dato!=0)
  {
   for(cnt=0;cnt<64;cnt++)
     Leer_USB[cnt]=Escribir_USB[cnt];
   }
   while(!HID_Write(&Escribir_USB,64)) ;
 }
}
```

Se propone la declaración de una nueva variable ‘dato’ y que reciba lo que la función Hid_Read lea. De esta forma si Hid_Read está leyendo (ósea que devuelva algo diferente de cero) entrara a realizar alguna acción, por ahora manda y escribe los mismos datos.

Para compilar el programa se oprime el boton build, o se presion Ctrl+F9.Posteriormente se carga el archivo .hex en el microcontrolador.

El microcontrolador cuenta con los pines MCLR,PGD,PGC,VDD y VDD para programar, fisicamente en el circuito impreso se conecta para que no sea necesario sacarlo de su base.

Al cargar el programa y conectar el con un cable USB el circuito con la computadora, lo detectará automaticamente. Para verificarlo se puede ir al Panel de Control de la computadora y en la seccion de Dispositivos e Impresoras se encuentra ‘Interfaz USB’ como se propuso llamarle, la siguiente imagen lo ejemplifica.