



Aguijón

Notas de aplicación

Nota de aplicación 16:

Clock with RTCC

Descripción:

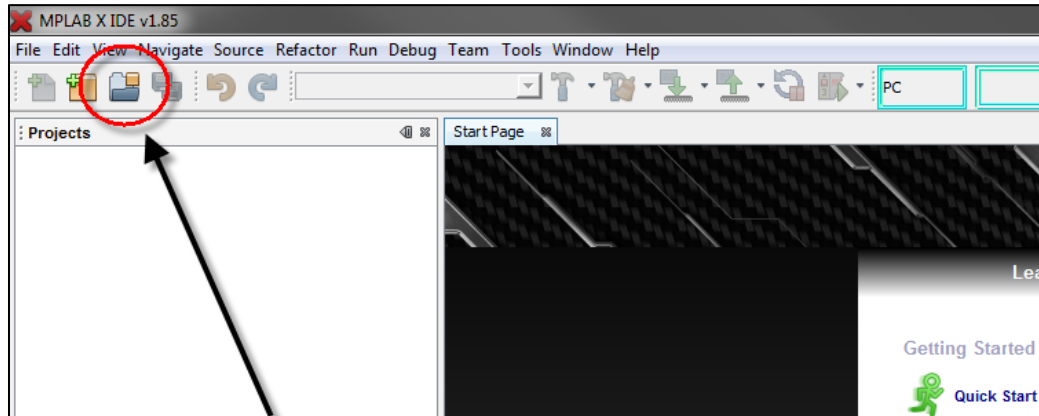
Simular un reloj utilizando el desbordamiento del RTCC.

Herramientas:

1. Aguijón 3.0, Aguijón 4.0 ó Aguijón 4.1
2. MPLAB X®
3. Aguijón HID bootloader
4. Cable USB 'A' to 'A'
5. Librerías HammerHead.

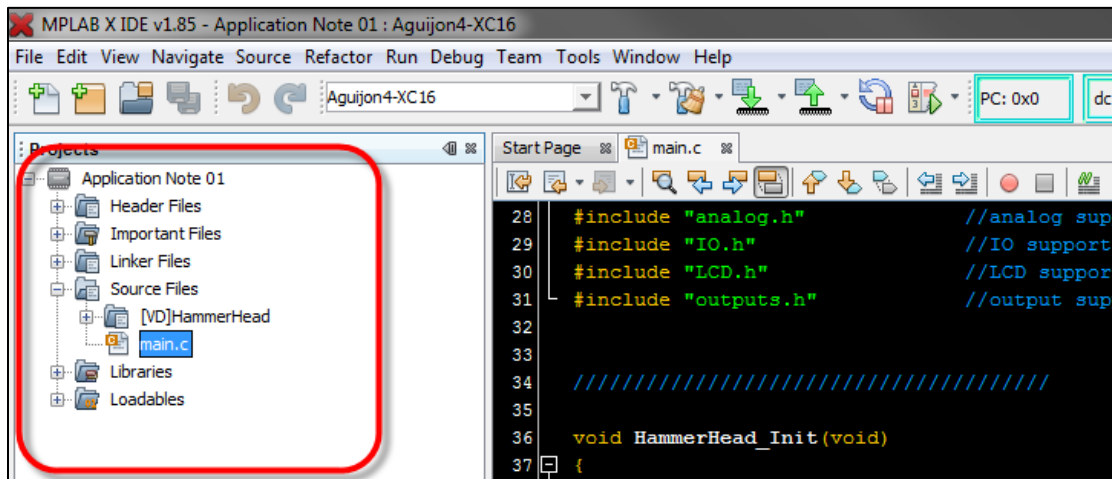
Pasos:

1. Abrir MPLAB X® y cargar el archivo del proyecto: **Application Note 16.X**



Haz 'clic' aquí y
abre el proyecto

1. Abrir el archivo **main.c**



2. Ir a la línea #41.

Utilizaremos la siguiente función:

```
40      /*Timer 1 interrupt service routine*/
41      void __attribute__((interrupt,no_auto_psv)) _RTCCInterrupt(void)
42      {
43          sec++;
44          if (sec>=60){
45              sec=0;
46              min++;
47          }
48          if (min>=60){
49              min=0;
50              hour++;
51          }
52          if (hour>=24){
53              hour=0;
54          }
```

`void __attribute__((interrupt, no_auto_psv)) _RTCCInterrupt(void)`

- Esta función interrumpe el flujo del programa cada vez que el RTCC es desbordado, por lo cual es aquí donde pondremos las instrucciones a realizar cada desbordamiento.

3. Ir a la línea #72

Utilizaremos la siguiente función:

```
65      /*Inputs*/
66      ADC_Init();
67
68      /*LCD*/
69      LCD_Init(LCD_MODE_1);
70
71      /*RTCC*/
72      RTCC_Init();
73
74      #if defined(USE_LCD_EXTRA_FEATURES)
75          LCD_BacklightFadeIn();
76      #else
77          LCD_BacklightSet(BLIGHT_LVL_4);
78      #endif
79
```

RTCC_Init();

- *Función que Inicializa el módulo RTCC y el oscilador.*

4. Ir a la línea #105 y #108

Utilizaremos las siguientes funciones:

```
98     for(;;){
99
100         keyboard=SW_Read();           //Read keyboard
101
102         switch (keyboard)
103         {
104             case 1:
105                 RTCC_Enable();          //Enable RTCC
106                 break;
107             case 2:
108                 RTCC_Disable();         //Disable RTCC
109                 break;
110             case 3:
111                 Timer1_Disable();       //Disable RTCC
112                 sec=0;
```

RTCC_Eneable ();

- Función que habilita el módulo RTCC;

RTCC_Disable ();

- Función que deshabilita el módulo RTCC;

2. Ir a la línea #118

Utilizaremos la siguiente función:

```
111         RTCC_Disable();           //Disable RTCC
112         sec=0;
113         min=0;
114         hour=0;
115         break;
116     case 4:
117         if (mode) {
118             LCD_Init(LCD_MODE_1);   //Lcd double line
119             mode=FALSE;
120         }
121         else {
122             LCD_Init(LCD_MODE_2);   //Lcd double height
123             mode=TRUE;
124         }
125         break;
```

LCD_Init (BOOL mode);

- Esta función inicializa la pantalla LCD en sus dos modos (Doble Altura ó Doble línea);
Dónde:
BOOL mode = Modo que se desea inicializar (LCD_MODE_1 =Doble Línea,
LCD_MODE_2=Doble Altura)

3. Ir a la línea #128

Utilizaremos la siguiente función:

```
121         else {
122             LCD_Init(LCD_MODE_2);           //Lcd double height
123             mode=TRUE;
124         }
125         break;
126     }
127
128     sprintf(lcdMSG,"%%.2u:%%.2u:%%.2u",hour,min,sec);           //Create string
129     LCD_PutStr(1,6,lcdMSG,FALSE);                               //Write lcdMSG
130     LCD_PutStr(2,0,"Start Stop Reset BIG",FALSE);
131
132 }
133 return 0;
134 }
```

sprintf (char *, const char *, ...);

- Función que Genera una cadena de caracteres y lo asigna a una variable; Donde:
Char * = Variable a asignar cadena de caracteres (Variable de tipo char)
const char *, = Cadena de caracteres a asignar a la variable.

4. Ir a la línea #129

Utilizaremos la siguiente función:

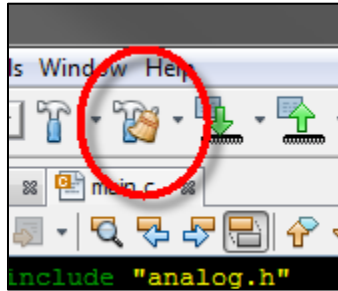
```
122         LCD_Init(LCD_MODE_2);           //Lcd double height
123         mode=TRUE;
124     }
125     break;
126 }
127
128     sprintf(lcdMSG,"%%.2u:%%.2u:%%.2u",hour,min,sec);           //Create string
129     LCD_PutStr(1,6,lcdMSG,FALSE);           //Write lcdMSG
130     LCD_PutStr(2,0,"Start Stop Reset BIG",FALSE);
131
132 }
133 return 0;
134 }
```

LCD_PutStr (int y, int x, char *msg, BOOL clear);

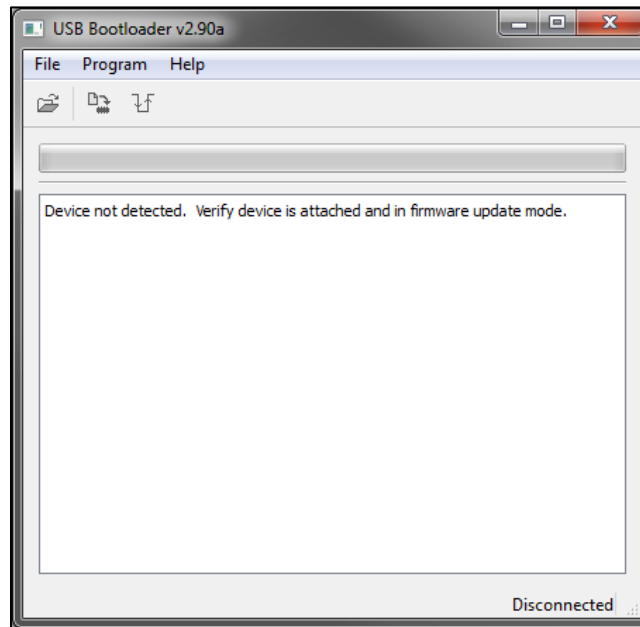
- Función que muestra una cadena de caracteres en la pantalla LCD; donde:
Int y = Coordenada en y (Valor entero del 1 al 2.)
Int X = Coordenada en x(Valor entero del 0 al 20.)
Char *msg = Cadena de caracteres (De 0 a 20 caracteres)
BOOL clear = Determina si se borra la pantalla antes de escribir
(TRUE =Borrar, FALSE =No borrar).

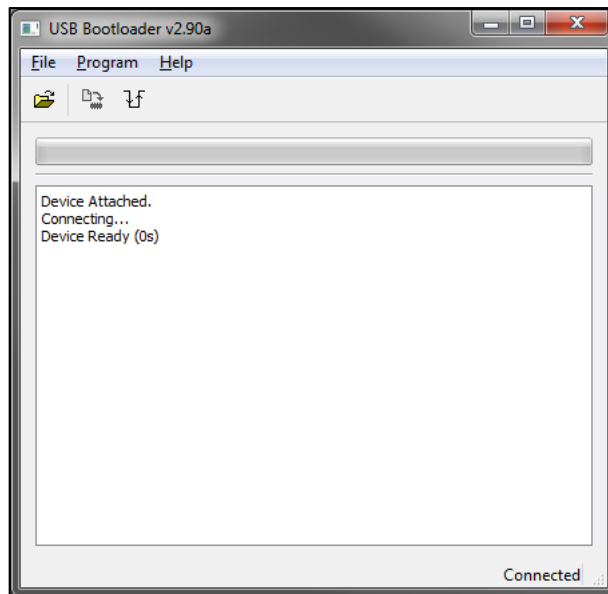
5. Compilar y programar

Al hacer clic en el ícono de compilar, y si no hay errores de compilación, el bootloader será cargado automáticamente.



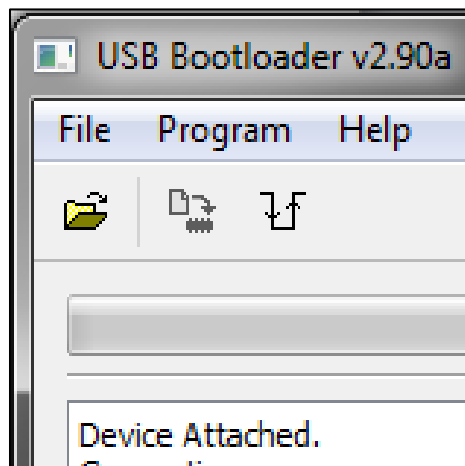
Cuando aparezca la ventana del bootloader, presione el Push-Button número 1 y conecte la fuente de voltaje o encienda el Aguijón y mantenga el PB1 presionado hasta que los LEDs empiecen a parpadear.





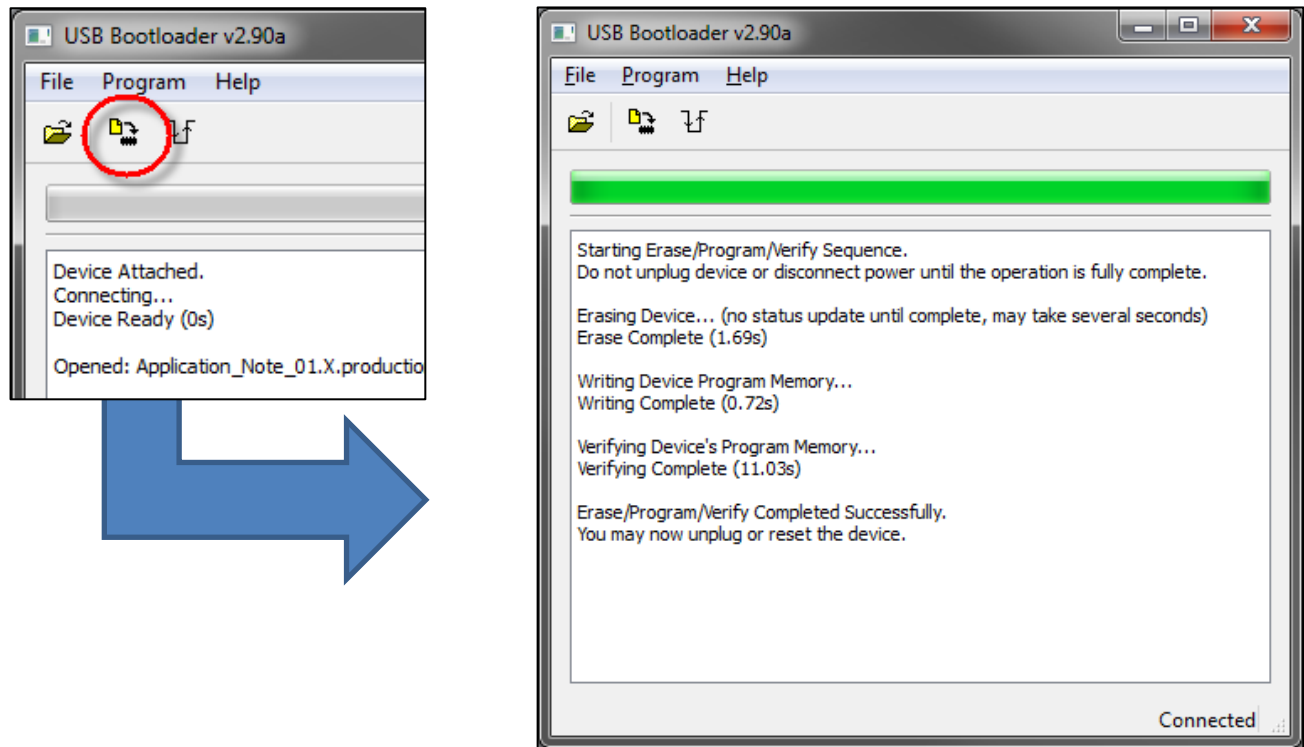
La ventana del Bootloader indicará la conexión establecida con el aguijón:

6. Hacer Clic en Abrir y Cargar el archivo **Application Note 16.X.production.hex**

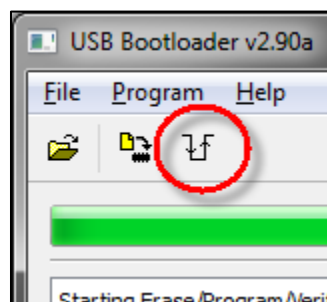


El archivo, depende de la plataforma de hardware.

Una vez cargado el archivo, hacer clic en el ícono de programar.



Hacer clic en el ícono de Reset cuando en la ventana del Bootloader se indique que se terminó de programar con éxito.



Una vez programado podemos verificar el programa corriendo en la tarjeta.

7. Para verificar el funcionamiento del programa verifique el reloj inicie su conteo al presionar el Push-Button 1, se pare al presionar el Push-Button 2, y al presionar el Push-Button 3 se reinicie el conteo, esto en modo doble línea, a menos que se presione el Push-Button 4 y se active el modo doble altura.