



Aguijón

Notas de aplicación

Nota de aplicación 38:

Alarm clock

Descripción:

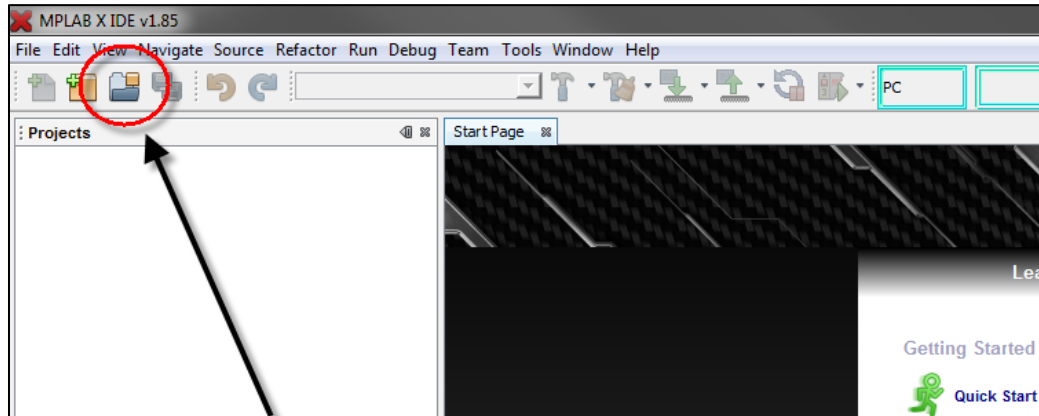
Controlar y atender una serie de alarmas previamente establecidas, basadas en el Reloj de tiempo real incorporado en nuestra tablilla.

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 38.X**



Haz 'clic' aquí y
abre el proyecto

2. Abrir el archivo **main.c**



3. Ir a la línea #112

Utilizaremos la siguiente función:

```
105     for(;;){
106
107         switch (state)
108         {
109             /*Set week day*/
110             case SET_DAY:
111                 //Create a string for the LCD//
112                 LCD_PutStr(1,0,"Today's Date",TRUE);
113
114                 keyboard=SW_Read();    //Read keyboard
115                 if(keyboard==1) {
116                     preday++;
117                     preday=(preday==7) ? 0 : preday;
118                 }
119                 if(keyboard==4)
```

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).

4. Ir a la línea #114.

Vamos a utilizar la siguiente función:

```
107         switch (state)
108         {
109             /*Set week day*/
110             case SET_DAY:
111                 //Create a string for the LCD//
112                 LCD_PutStr(1,0,"Today's Date",TRUE);
113
114                 keyboard=SW_Read(); //Read keyboard
115                 if(keyboard==1) {
116                     preday++;
117                     preday=(preday==7) ? 0 : preday;
118                 }
119                 if(keyboard==4)
120                     state=SET_HOUR; //Go to SET HOUR state
121
```

SW_Read ();

- Función que lee el puerto de PUSH BUTTONS;
Devuelve un valor equivalente al Push-Button presionado (carácter del 1 al 4)

5. Ir a la línea #130.

Vamos a utilizar la siguiente función:

```
123         LCD_PutStr(2,18,"OK",FALSE);
124         break;
125
126         /*Set Hours & Minutes*/
127         case SET_HOUR:
128             //Create a string for the LCD//
129             LCD_PutStr(1,0,"Set Hour",TRUE);
130             Config_hour(); //Read keyboard & adjust Hour
131             if(next) {
132                 next = FALSE;
133                 TIME_Set(SET_WKDYHR,preday+1,prehour);
134                 TIME_Set(SET_MINSEC,premin,00);
135                 state=SET_ALARM; //Go to SET ALARM state
136                 RTCC_Enable();
137             }
```

Config_hour ();

- Función que lee el puerto de PUSH BUTTONS, para así mismo incrementar los valores previos de horas y minutos, mientras despliega en la pantalla LCD el valor de la hora a configurar. ;

6. Ir a la línea #133

Utilizaremos la siguiente función:

```
126      /*Set Hours & Minutes*/
127      case SET_HOUR:
128          //Create a string for the LCD//
129          LCD_PutStr(1,0,"Set Hour",TRUE);
130          Config_hour();      //Read keyboard & adjust Hour
131          if(next) {
132              next = FALSE;
133              TIME_Set(SET_WKDYHR,preday+1,prehour);
134              TIME_Set(SET_MINSEC,premin,00);
135              state=SET_ALARM;    //Go to SET ALARM state
136              RTCC_Enable();
137          }
138          break;
139
140      /*Set Alarm*/
```

TIME_Set (char set , char first , char second);

- Función que ajusta los valores del reloj de tiempo real serial RS232
Esta función debe ser llamada las veces necesarias para ajustar todos valores necesarios;
donde:

Char set	Char first	Char second
SET_YEAR	NULL	YEAR
SET_MTHDY	MONTH	DAY
SET_WKDYHR	WEEK DAY	HOUR
SET_MINSEC	MINUTES	SECONDS

1. Ir a la línea #136

Utilizaremos las siguientes funciones:

```
129         LCD_PutStr(1,0,"Set Hour",TRUE);
130         Config_hour();           //Read keyboard & adjust Hour
131         if(next) {
132             next = FALSE;
133             TIME_Set(SET_WKDYHR,preday+1,prehour);
134             TIME_Set(SET_MINSEC,premin,00);
135             state=SET_ALARM;      //Go to SET ALARM state
136             RTCC_Enable();
137         }
138         break;
139
140         /*Set Alarm*/
141         case SET_ALARM:
142             //Create a string for the LCD//
143             sprintf(lcdMSG,"Set alarm %i",alarm_num+1);
```

RTCC_Eneable ();

- Función que habilita el módulo RTCC;

RTCC_Disable ();

- Función que deshabilita el módulo RTCC;

7. Ir a la línea #143

Utilizaremos la siguiente función:

```
136         RTCC_Enable();
137     }
138     break;
139
140     /*Set Alarm*/
141     case SET_ALARM:
142         //Create a string for the LCD//
143         sprintf(lcdMSG,"Set alarm %i",alarm_num+1);
144         LCD_PutStr(1,0,lcdMSG,TRUE);
145         Config_hour(); //Read keyboard & adjust Hour
146         if(next) {
147             Alarm[alarm_num].Hour = prehour;
148             Alarm[alarm_num].Min = premin;
149             state=SET_ALARMDAY; //Go to SET ALARMDAY state
150             next = FALSE;
```

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.

8. Ir a la línea #166.

Utilizaremos la siguiente función:

```
159         LCD_PutStr(1,0,"Select days with...",TRUE);
160     else
161         LCD_PutStr(1,0,"Dip-sw & press Here",TRUE);
162     i = (i>40) ? 0 : i;
163     LCD_PutStr(2,0,"Su Sa Fr Th We Tu Mo",FALSE);
164     keyboard=SW_Read();
165
166     LEDport Set (DIPSW_Read(),ON);
167     if(keyboard==4) {
168         Alarm[alarm_num].Days = DIPSW_Read();
169         alarm_num++;
170         state=(alarm_num==ALARMS) ? CLOCK : SET_ALARM; //Go to
171     }
172     break;
173
```

DIPSW_Read ();

- Esta función lee el puerto de DIP-SWITCH;
Regresa un valor equivalente al valor del DIP-SWITCH (carácter del 1 al 255)

9. Ir a la línea #172.

Vamos a utilizar la siguiente función:

```
170     state=(alarm_num==ALARMS) ? CLOCK : SET_ALARM; //Go to SET_ALARM or CLOCK state
171     }
172     break;
173
174     /*Clock*/
175     case CLOCK:
176         //Create a string for the LCD//
177         sprintf(lcdMSG,"%02i:%02i:%02i",TIME_Get(GET_HOURS),TIME_Get(GET_MINUTES),TIME_Get(GET_SECONDS));
178         LCD_PutStr(1,5,name_day[TIME_Get(GET_WEEKDAY)-1],TRUE);
179         LCD_PutStr(2,5,lcdMSG,FALSE);
180
181         /*if the current time is equal to the alarm*/
182         for (alarm_num=0 ; alarm_num <ALARMS ; alarm_num++) {
183             if(((Alarm[alarm_num].Days>>(TIME_Get(GET_WEEKDAY)-1)) & 0b1)
184                 && (Alarm[alarm_num].Hour == TIME_Get(GET_HOURS))
```

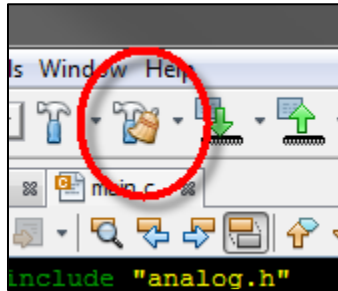
TIME_Get (char get);

- Función que lee el valor del reloj de tiempo real;
Devuelve un valor equivalente al valor actual del parámetro seleccionado (entero del 0 al 99)

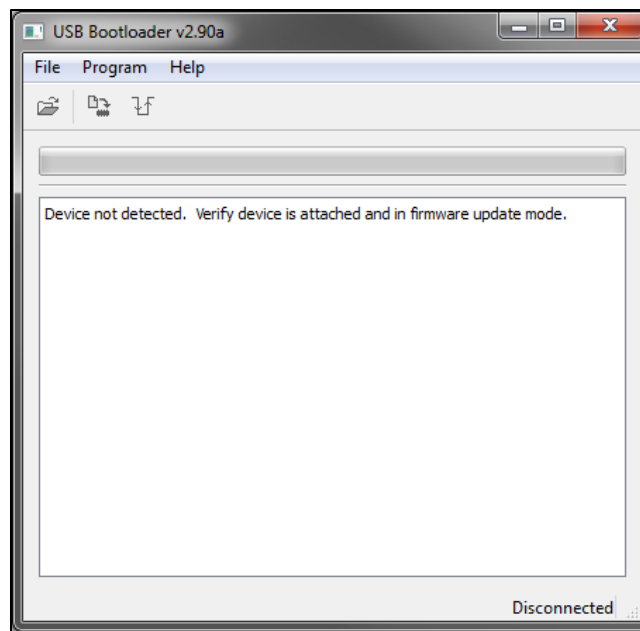
GET_YEAR	0 – 99
GET_MONTH	1 – 12
GET_DAY	1 – 31
GET_WEEKDAY	1 – 7
GET_HOURS	0 – 24
GET_MINUTES	0 – 60
GET_SECONDS	0 – 60

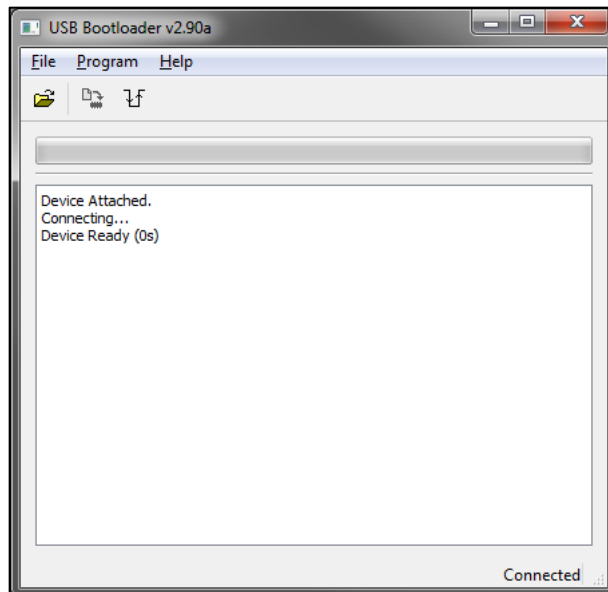
10. Compilar y programar

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



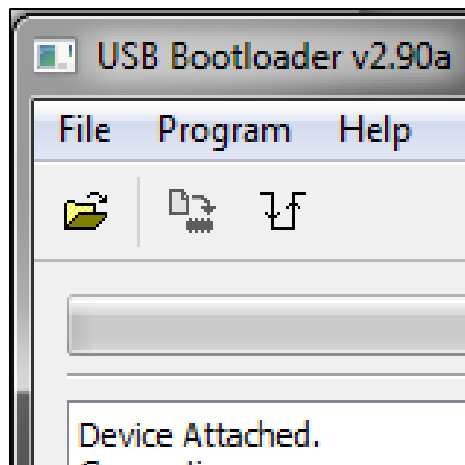
Cuando aparezca la ventana del bootloader, presione el Push-Button número 1 y conecte la fuente de voltaje o encienda el Aguijón y mantenga el PB1 presionado hasta que los LEDs empiecen a parpadear.





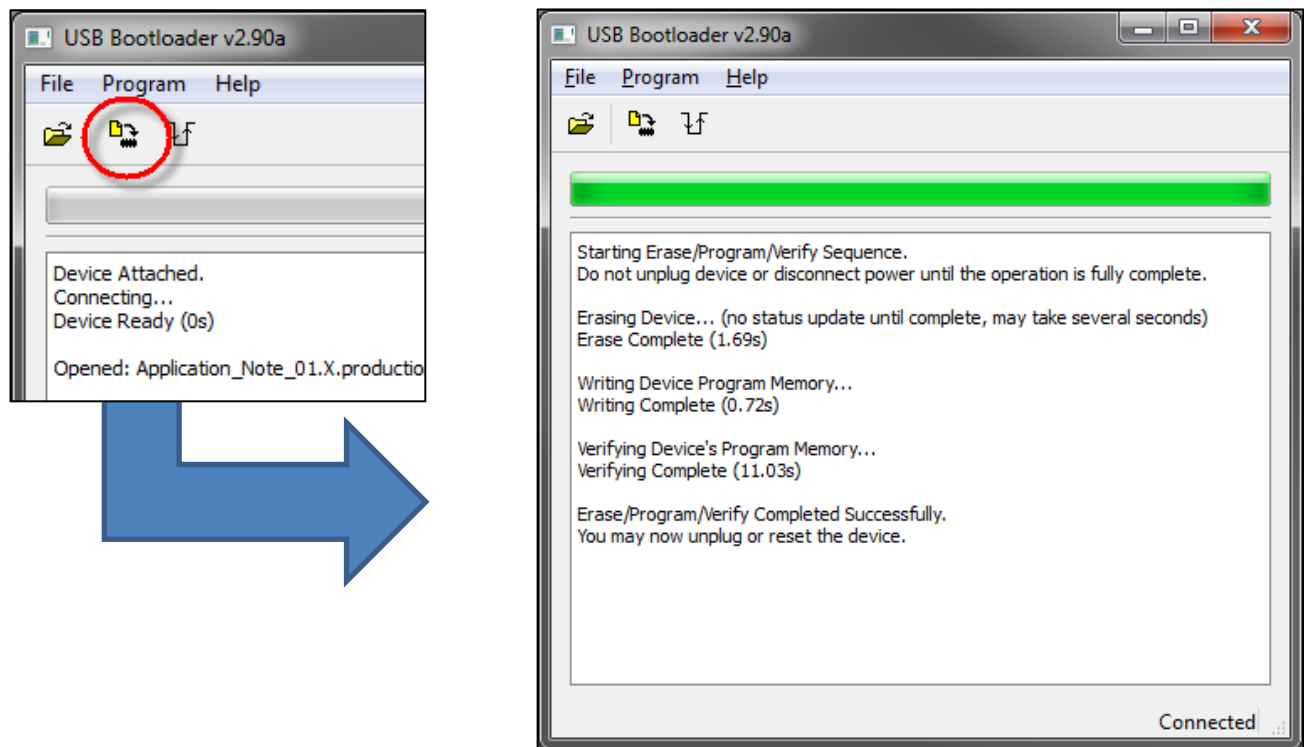
La ventana del Bootloader indicará la conexión establecida con el aguijón:

11. Hacer Clic en Abrir y Cargar el archivo **Application Note 38.X.production.hex**

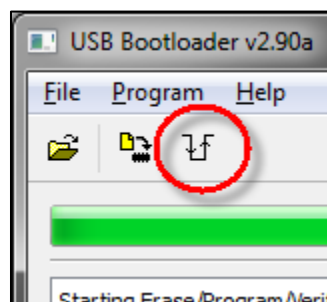


El archivo, depende de la plataforma de hardware.

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.

12. Para verificar el funcionamiento del programa verifique que al iniciar nuestro programa este pida que se ajuste tanto la hora como la cantidad de alarmas indicadas, para luego desplegar continuamente la hora y el día de la semana, cuando alguna de las alarmas se cumpla este deberá de producir un sonido generado desde el Buzzer, así como conmutar el relevador #4.