



*Aguijón*

---

*Notas de aplicación*

## **Nota de aplicación 26:**

### **Make a spreadsheet**

#### **Descripción:**

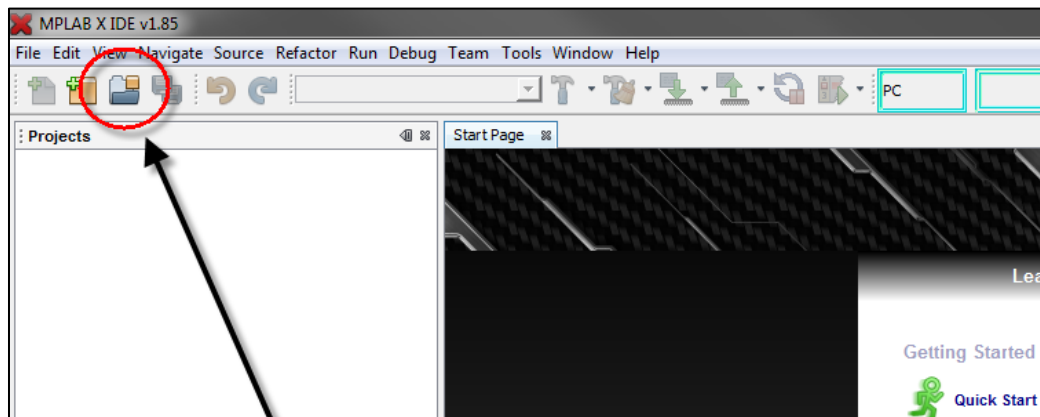
*Crear una hoja de cálculo (.CSV) a partir de una serie de mediciones en el puerto de entrada ADC.*

#### **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.
6. Memoria usb 2.0

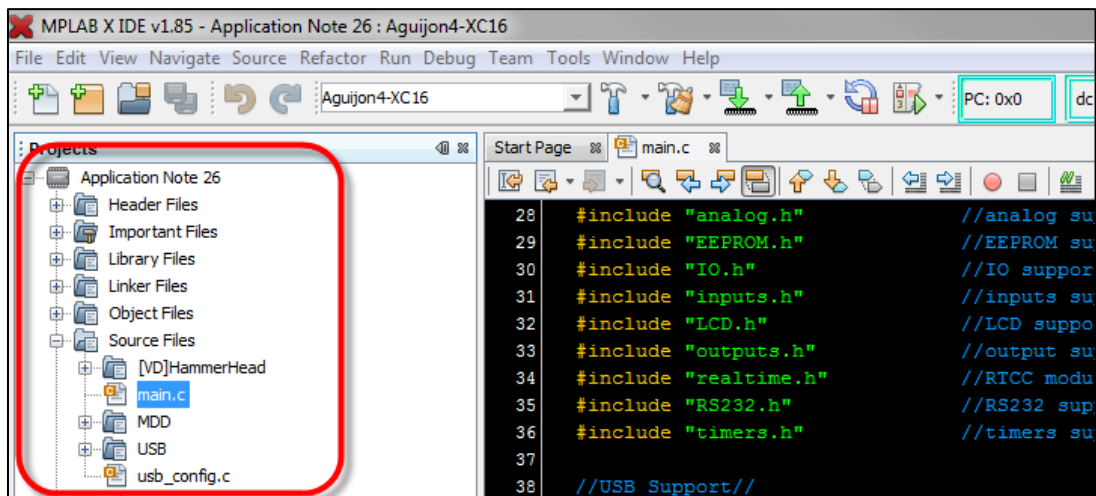
## Pasos:

1. Abrir MPLAB X® y cargar el archivo del proyecto: **Application Note 26.X**



Haz 'clic' aquí y  
abre el proyecto

2. Abrir el archivo *main.c*



3. Ir a la línea #132

Utilizaremos la siguiente función:

```
125 LCD_PutStr(2,0,"Application Note 26",FALSE);
126 delayms(1000);
127 LCD_PutStr(1,0,"Aguijon USB MSD",TRUE);
128 LCD_PutStr(2,0,"Insert flash drive",FALSE);
129
130 for(;;){
131     //USB stack process function
132     USBTasks();
133
134     //if thumbdrive is plugged in
135     if(USBHostMSDSCSIMediaDetect()){
136         InsertChime();
137         //now a device is attached
138
139         //See if the device is attached and in the right format
```

### **USBTasks ();**

- Esta función es la herramienta necesaria para configurar e inicializar nuestro dispositivo USB, esta debe ser llamada antes que cualquier otra función relacionada con el puerto USB

4. Ir a la línea #135

Utilizaremos la siguiente función:

```
128 LCD_PutStr(2,0,"Insert flash drive",FALSE);
129
130 for(;;){
131     //USB stack process function
132     USBTasks();
133
134     //if thumbdrive is plugged in
135     if(USBHostMSDSCSIMediaDetect()){
136         InsertChime();
137         //now a device is attached
138
139         //See if the device is attached and in the right format
140         if(FSInit()){
141
142             LCD_PutStr(1,0,"Start measurements",TRUE);
```

**USBHostMSDSCSIMediaDetect ( );**

- Función que determina si un dispositivo de almacenamiento masivo está conectado y listo para utilizar; donde:  
Devuelve un valor booleano dependiendo del estado del dispositivo  
(TRUE= MSD presente y listo, FALSE = MSD no está presente o no está listo).

5. Ir a la línea #140.

Utilizaremos la siguiente función:

```
133
134 //if thumbdrive is plugged in
135 if(USBHostMSDSCSIMediaDetect()){
136     InsertChime();
137 //now a device is attached
138
139 //See if the device is attached and in the right format
140 if(FSInit()){
141
142     LCD_PutStr(1,0,"Start measurements",TRUE);
143
144     //Opening a file in mode "a"
145     myFile = FSfopen("test.csv","a");
146     delayms(500); //Delay 500 milliseconds
147 }
```

**FSInit ();**

- Esta función inicializa el dispositivo físico que debe ser conectado al microcontrolador ;  
donde:  
Devuelve un valor booleano dependiendo del estado de la inicialización  
(TRUE= Inicialización exitosa, FALSE = Inicialización no exitosa).

6. Ir a la línea #142

Utilizaremos la siguiente función:

```
135         if(USBHostMSDSCSIMediaDetect()){
136             InsertChime();
137             //now a device is attached
138
139             //See if the device is attached and in the right format
140             if(FSInit()){
141
142                 LCD_PutStr(1,0,"Start measurements",TRUE);
143
144                 //Opening a file in mode "a"
145                 myFile = FSfopen("test.csv","a");
146                 delaysms(500); //Delay 500 milliseconds
147
148                 //Write some data to the new file.
149                 FSfwrite("\x0D Measuring number,Value",1,25,myFile);
```

**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).

7. Ir a la línea #145

Utilizaremos la siguiente función:

```
138
139 //See if the device is attached and in the right format
140 if(FSInit()){
141
142     LCD_PutStr(1,0,"Start measurements",TRUE);
143
144     //Opening a file in mode "a"
145     myFile = FSfopen("test.csv","a");
146     delays(500); //Delay 500 milliseconds
147
148     //Write some data to the new file.
149     FSfwrite("\x0D Measuring number,Value",1,25,myFile);
150
151     for(i=1;i<=10;i++)
152     {
```

**FSfopen (char \* fileName, char \*mode);**

- Función que edita un archivo ya sea este nuevo o existente; donde:  
**Char \*FileName** = El nombre del archivo a abrir o crear. (Cadena de caracteres, incluyendo extensión del archivo.)  
**Char \*mode** = Modo del uso que se le desea dar al archivo; donde:
  - W:** Crea un archivo nuevo o reemplaza uno existente
  - R:** Lee datos de un archivo existente
  - A:** Agrega datos a un archivo existente
  - W+:** Crea un archivo nuevo o reemplaza uno existente (También habilita lectura)
  - R+:** Lee datos de un archivo existente (También habilita escritura)
  - A+:** Agrega datos a un archivo existente (También habilita lectura)



8. Ir a la línea #149

Utilizaremos la siguiente función:

```
142         LCD_PutStr(1,0,"Start measurements",TRUE);
143
144         //Opening a file in mode "a"
145         myFile = FSfopen("test.csv","a");
146         delayms(500); //Delay 500 milliseconds
147
148         //Write some data to the new file.
149         FSfwrite("\x0D Measuring number,Value",1,25,myFile);
150
151         for(i=1;i<=10;i++)
152         {
153             ADCvalue=ADC_Range(0,100); //Read ADC port
154
155             //Create a strings for the LCD//
156             sprintf(lcdMSG,"Measurement# %i =%i",i,ADCvalue);
```

**FSfwrite (void \*data\_to\_write, unsigned long size, unsigned long n, FSFILE \*stream);**

- Función que escribe datos en algún archivo; donde:  
**data\_to\_write** = Apuntador a la cadena de caracteres a escribir.  
**Size** = Tamaño de las unidades en bytes  
**n** = Número de unidades a transferir.  
**Stream** = Apuntador a la estructura del archivo

9. Ir a la línea #153.

Utilizaremos la siguiente función:

```
146         delays(500); //Delay 500 milliseconds
147
148         //Write some data to the new file.
149         FSfwrite("\x0D Measuring number,Value",1,25,myFile);
150
151         for(i=1;i<=10;i++)
152         {
153             ADCvalue=ADC_Range(0,100); //Read ADC port
154
155             //Create a strings for the LCD//
156             sprintf(lcdMSG,"Measurement# %i =%i",i,ADCvalue);
157             LCD_PutStr(1,0,lcdMSG,TRUE);
158
159             //Create a strings for the FILE//
160             sprintf(fileMSG,"\x0D Measurement# %i ,%i ",i,ADCvalue);
```

**ADC\_Range (int min\_val, int max\_val);**

- Esta función lee el puerto ADC y devuelve un valor proporcional al Rango establecido;  
Donde:  
**Int min\_val** = valor mínimo del rango (Entero de 0 a 1023)  
**Int max\_val** = valor máximo del rango (Entero de 0 a 1023)  
Devuelve un valor proporcional al rango (Entero de min\_val a max\_val).

10. Ir a la línea #156

Utilizaremos la siguiente función:

```
149         FSfwrite("\x0D Measuring number,Value",1,25,myFile);
150
151         for(i=1;i<=10;i++)
152         {
153             ADCvalue=ADC_Range(0,100);           //Read ADC port
154
155             //Create a strings for the LCD//
156             sprintf(lcdMSG,"Measurement# %i =%i",i,ADCvalue);
157             LCD_PutStr(1,0,lcdMSG,TRUE);
158
159             //Create a strings for the FILE//
160             sprintf(fileMSG,"\x0D Measurement# %i ,%i ",i,ADCvalue);
161             FSfwrite(fileMSG,1,22,myFile);
162             delaysms(500);                       //Delay 500 milliseconds
163         }
```

***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.

11. Ir a la línea #165

Utilizaremos la siguiente función:

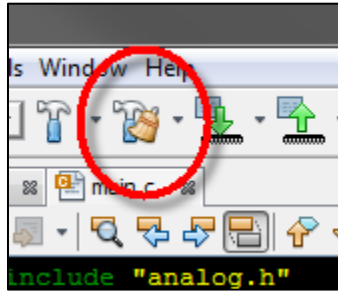
```
158
159         //Create a strings for the FILE//
160         sprintf(fileMSG, "\x0D Measurement# %i ,%i ", i, ADCvalue);
161         FSfwrite(fileMSG, 1, 22, myFile);
162         delayms(500); //Delay 500 milliseconds
163     }
164     //Always make sure to close the file.
165     FSfclose(myFile);
166
167     //Just sit here until the device is removed.
168     LCD_PutStr(1, 0, "DONE!", TRUE);
169     LCD_PutStr(2, 0, "Safe to remove USB", FALSE);
170     while (USBHostMSDSCSIMediaDetect()) { USBTasks(); }
171
172     RemoveChime();
```

**FSfclose (FSFILE \*fo);**

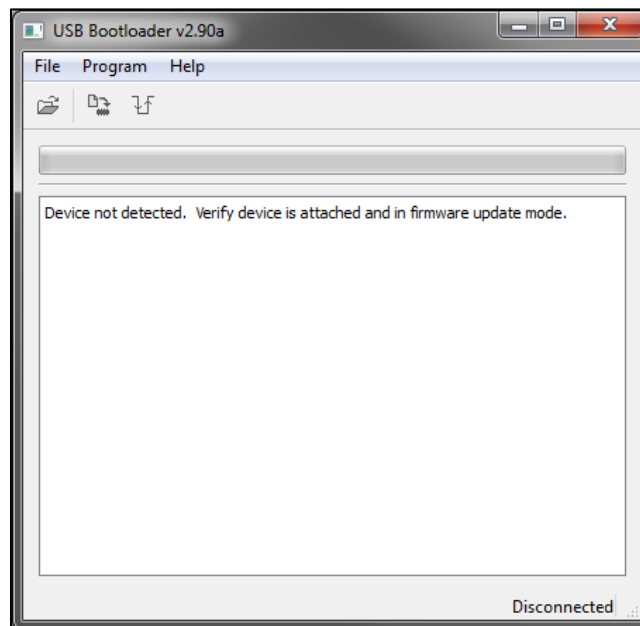
- Función que Actualiza la información y cierra el archivo seleccionado; donde:  
**FSFILE \*fo** = Apuntador a la estructura del archivo a cerrar.

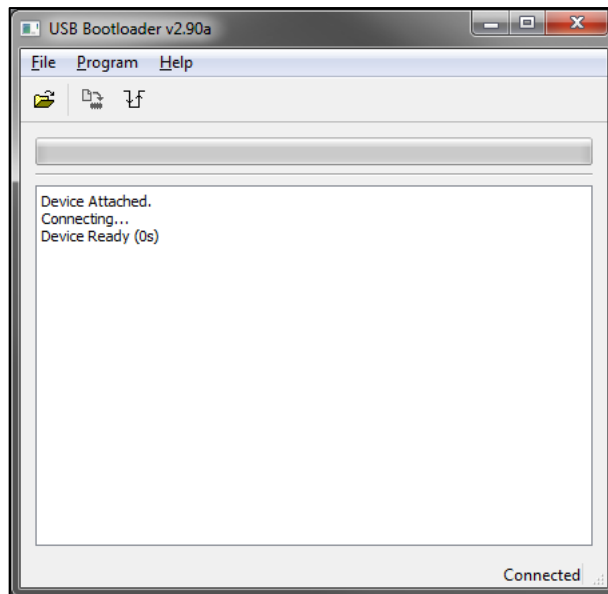
## 12. Compilar y programar

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



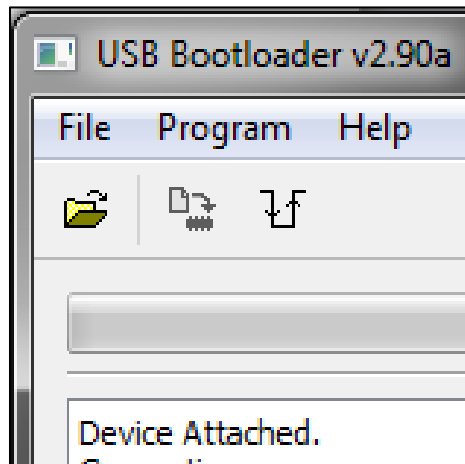
Cuando aparezca la ventana del bootloader, presione el Push-Button número 1 y conecte la fuente de voltaje o encienda el Aguijón y mantenga el PB1 presionado hasta que los LEDs empiecen a parpadear.





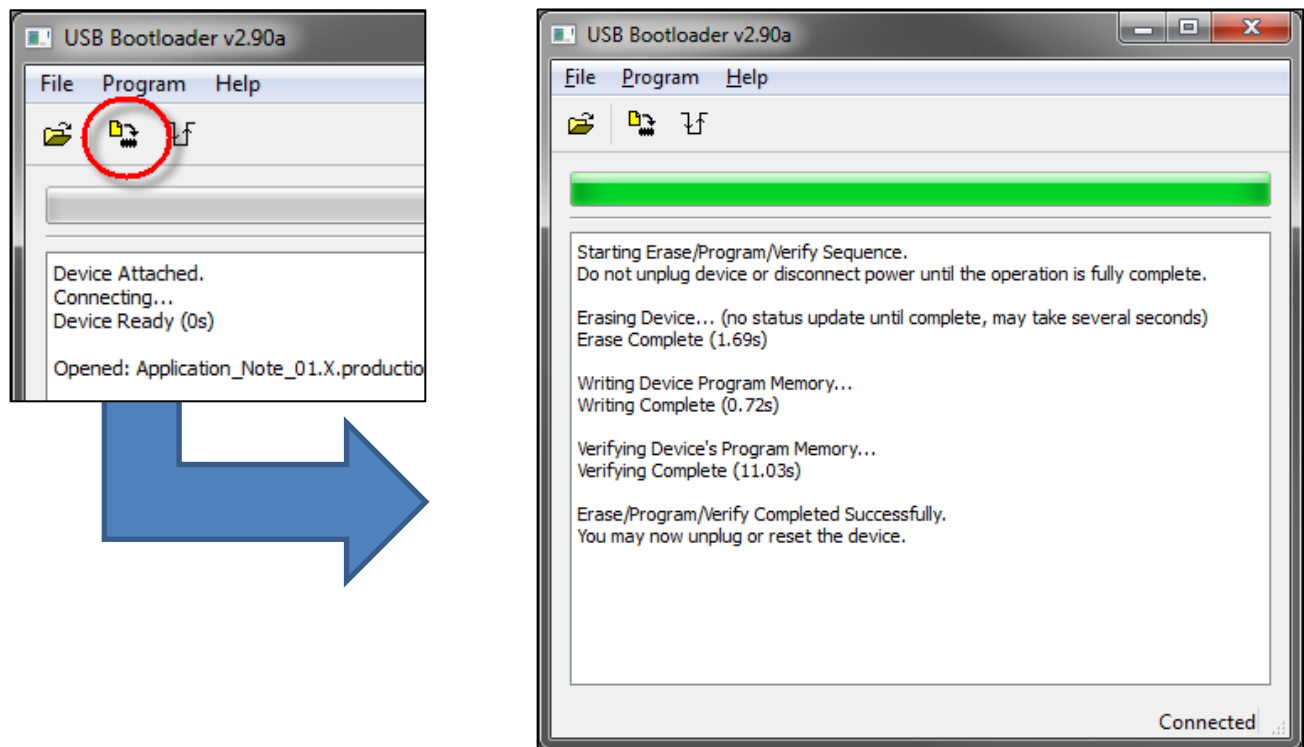
La ventana del Bootloader indicará la conexión establecida con el aguijón:

13. Hacer Clic en Abrir y Cargar el archivo **Application Note 26.X.production.hex**

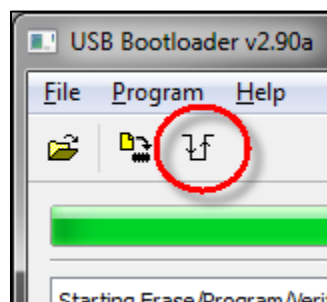


El archivo, depende de la plataforma de hardware.

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



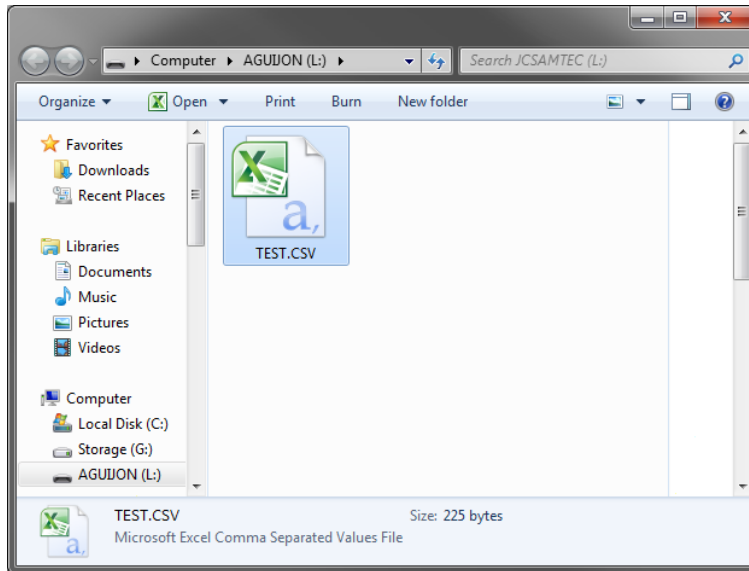
Hacer clic en el ícono de Reset cuando en la ventana del Bootloader se indique que se terminó de programar con éxito.



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

14. Para verificar el funcionamiento del programa verifique que al conectar un dispositivo de almacenamiento USB, se genera un archivo de tipo Hoja de cálculo con el nombre "TEST.CSV", el cual deberá contener una tabla con los valores enumerados de 10 muestreos tomados desde el puerto ADC.

15. Archivo **TEST.csv**



16. *Tabla de valores adquiridos.*

A screenshot of the Microsoft Excel application window titled 'TEST.CSV - Microsoft Excel'. The 'Home' tab is active, showing the ribbon with 'Clipboard', 'Font', 'Alignment', 'Number', 'Styles', 'Cells', and 'Editing' groups. The spreadsheet displays the following data:

	A	B	C	D	E	F
1						
2	Measuring number	Value				
3	Measurement# 1	10				
4	Measurement# 2	10				
5	Measurement# 3	91				
6	Measurement# 4	53				
7	Measurement# 5	28				
8	Measurement# 6	5				
9	Measurement# 7	0				
10	Measurement# 8	0				
11	Measurement# 9	60				
12	Measurement# 10	8				
13						
14						
15						
16						
17						
18						
19						

The status bar at the bottom shows 'Ready', the 'TEST' worksheet tab, and a zoom level of 100%.



17. Estos valores están disponibles para ser manipulados o utilizados, por ejemplo graficarlos.

