



Aguijón

Notas de aplicación

Nota de aplicación 14:

LCD Advanced (B)

Descripción:

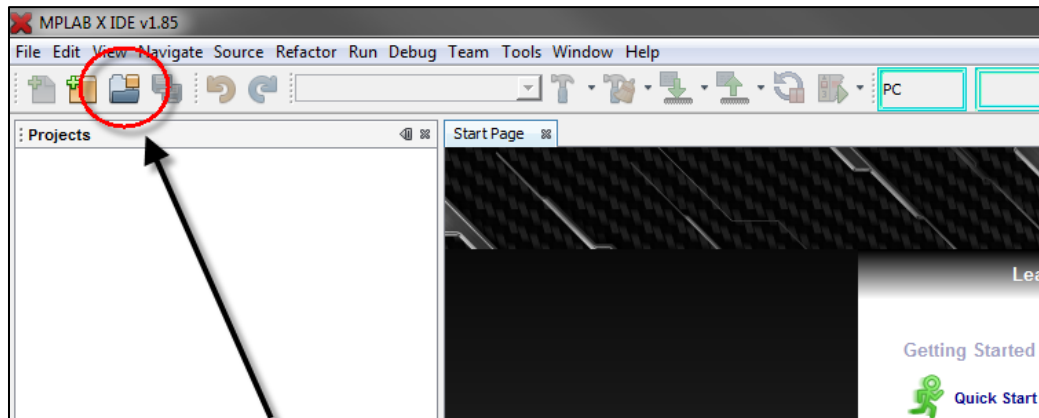
Generar y visualizar caracteres personalizados en la pantalla LCD.

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 14.X**



Haz 'clic' aquí y
abre el proyecto

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



3. Ir al directorio: **Application Notes\Tools\[VD]CCG**

Y abrir el archivo “cgc.htm”

[VD]Custom Character Generator for Aguijón LCD

Click pixels to generate output.

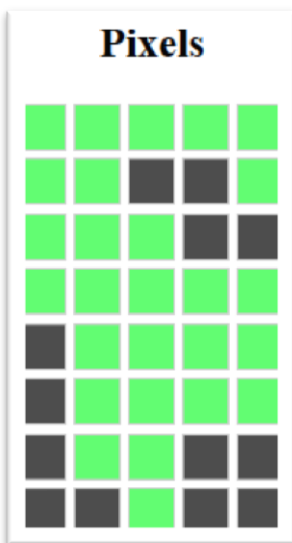
Pixels



Output

Copy the output to your code

```
unsigned char customChar[] = {  
    0b00000,  
    0b00110,  
    0b00011,  
    0b00000,  
    0b10000,  
    0b10000,  
    0b10011,  
    0b11011  
};
```



Los cuadros representan los pixeles de la LCD.

Al hacer clic en cualquier pixel, genera información del carácter nuevo.

El botón ‘clear’ borra todos los pixeles activos.

El botón ‘invert’ invierte los inactivos y los activos.

Output

```
unsigned char customChar[] = {  
    0b00000,  
    0b00110,  
    0b00011,  
    0b00000,  
    0b10000,  
    0b10000,  
    0b10011,  
    0b11011  
};
```

En la sección 'Output' se genera información relacionada con el carácter personalizado en forma de un arreglo en lenguaje C.

Estos arreglos se pueden agregar en el código y están a disposición del LCD

4. Ir a la línea #35.

Utilizaremos la siguiente función:

```
34  /* Custom chars */
35  unsigned char customChar1[] = {           //Create Custom Character 1
36      0b00100,
37      0b00100,
38      0b11111,
39      0b00000,
40      0b01110,
41      0b10101,
42      0b00100,
43      0b00100
44  };
45
46  unsigned char customChar2[] = {           //Create Custom Character 2
47      0X04,
48      0X04,
49      0X1F,
50      0X0E,
51      0X04,
52      0X1F,
53      0X04,
54      0X04
55  };
56
57  unsigned char customChar3[] = {           //Create Custom Character 3
58      0x00,
```

- Mediante el uso de la herramienta [VD] CCG se generaron los arreglos de cada 1 de los 8 caracteres personalizado.

5. Ir a la línea #160

Utilizaremos la siguiente función:

```
155 HammerHead_Init(); //initialize [VD]HammerHead
156 LCD_IntroAnimation();
157 LCD_PutStr(1,0,"Vinagron Digital",TRUE);
158 LCD_PutStr(2,0,"Application Note 14",FALSE);
159
160 LCD_StoreCustomChar(customChar1,0); //Store custom char
161 LCD_StoreCustomChar(customChar2,1); //Store custom char
162 LCD_StoreCustomChar(customChar3,2);
163 LCD_StoreCustomChar(customChar4,3);
164 LCD_StoreCustomChar(customChar5,4);
165 LCD_StoreCustomChar(customChar6,5);
166 LCD_StoreCustomChar(customChar7,6);
167 LCD_StoreCustomChar(customChar8,7);
168
169 for(;;){
```

LCD_StoreCustomChar (char *rows, char cgram_loc);

- Función que guarda los caracteres personalizados en una localidad de la CGRAM; donde:
Char *rows = Arreglo de bits que representa el carácter (arreglo de 8 enteros)
Char cgram_loc = Localidad de la CGRAM a escribir (Carácter del 1 al 8).

6. Ir a la línea #171.

Vamos a utilizar la siguiente función:

```
164         LCD_StoreCustomChar(customChar5,4);
165         LCD_StoreCustomChar(customChar6,5);
166         LCD_StoreCustomChar(customChar7,6);
167         LCD_StoreCustomChar(customChar8,7);
168
169     for(;;) {
170
171         keyboard=SW_Read();
172
173         if(keyboard==1)
174         {
175
176             LCD_PutStr(1,0,"Store 8 custom chars",TRUE);    //Write "Store 8..."
177
178             LCD_GotoYX(2,5);                                //LCD goto Y2, X5
```

SW_Read ();

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

7. Ir a la línea #176

Utilizaremos la siguiente función:

```
169     for (;;) {
170
171         keyboard=SW_Read();
172
173         if (keyboard==1)
174         {
175
176             LCD_PutStr(1,0,"Store 8 custom chars",TRUE); //Write "Store 8..."
177
178             LCD_GotoYX(2,5); //LCD goto Y2, X5
179             LCD_PutCustomChar(0); //Put Custom Char 0
180             LCD_PutCustomChar(1); //Put Custom Char 1
181             LCD_PutCustomChar(2);
182             LCD_PutCustomChar(3);
183             LCD_PutCustomChar(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).

8. Ir a la línea #178

Utilizaremos la siguiente función:

```
171     keyboard=SW_Read();
172
173     if(keyboard==1)
174     {
175
176         LCD_PutStr(1,0,"Store 8 custom chars",TRUE);    //Write "Store 8...
177
178         LCD_GotoYX(2,5);                                //LCD goto Y2, X5
179         LCD_PutCustomChar(0);                            //Put Custom Char 0
180         LCD_PutCustomChar(1);                            //Put Custom Char 1
181         LCD_PutCustomChar(2);
182         LCD_PutCustomChar(3);
183         LCD_PutCustomChar(4);
184         LCD_PutCustomChar(5);
185         LCD_PutCustomChar(6);
```

LCD_GotoYX (int Y, int X);

- Función que envía el cursor del LCD a las coordenadas Y,X; donde:
Int y = Coordenada en y (Valor entero del 1 al 2.)
Int X = Coordenada en x (Valor entero del 0 al 20.)

9. Ir a la línea #179

Utilizaremos la siguiente función:

```
172
173     if (keyboard==1)
174     {
175
176         LCD_PutStr(1,0,"Store 8 custom chars",TRUE);    //Write "Store 8..."
177
178         LCD_GotoYX(2,5);                                //LCD goto Y2, X5
179         LCD_PutCustomChar(0);                            //Put Custom Char 0
180         LCD_PutCustomChar(1);                            //Put Custom Char 1
181         LCD_PutCustomChar(2);
182         LCD_PutCustomChar(3);
183         LCD_PutCustomChar(4);
184         LCD_PutCustomChar(5);
185         LCD_PutCustomChar(6);
186         LCD_PutCustomChar(7);
```

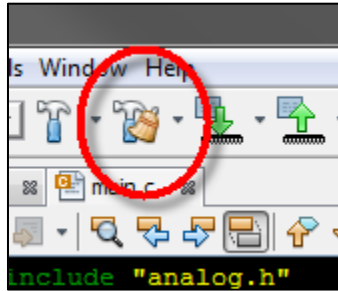
LCD_PutCustomChar (char cgram_log);

- Función que envía un carácter personalizado previamente almacenado en la CGRAM; donde:

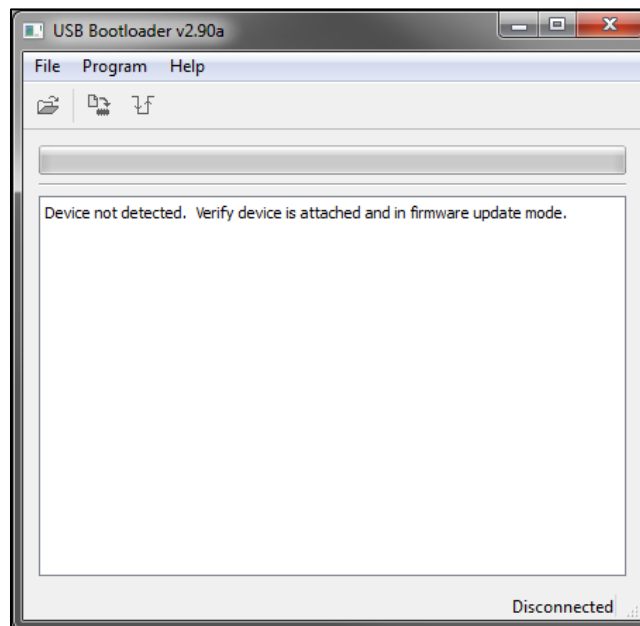
Char cgram_loc = Localidad de la CGRAM donde está almacenado el carácter (Carácter del 1 al 8).

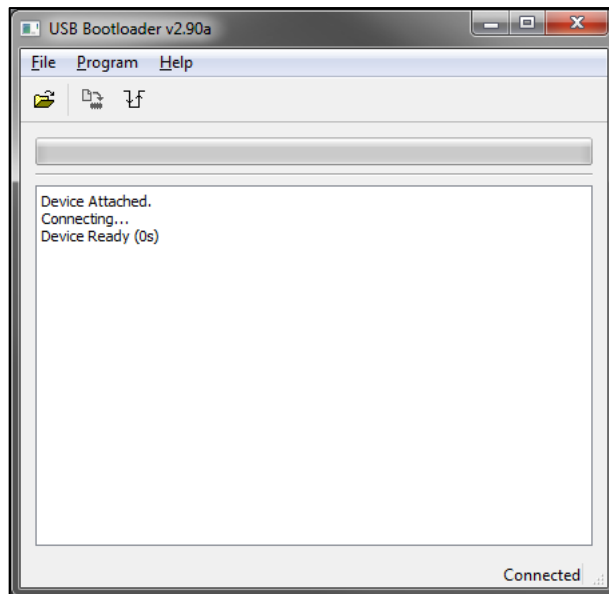
10. Compilar y programar

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



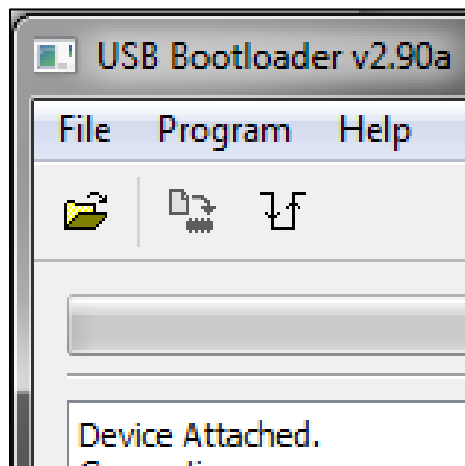
Cuando aparezca la ventana del bootloader, presione el Push-Button número 1 y conecte la fuente de voltaje o encienda el Aguijón y mantenga el PB1 presionado hasta que los LEDs empiecen a parpadear.





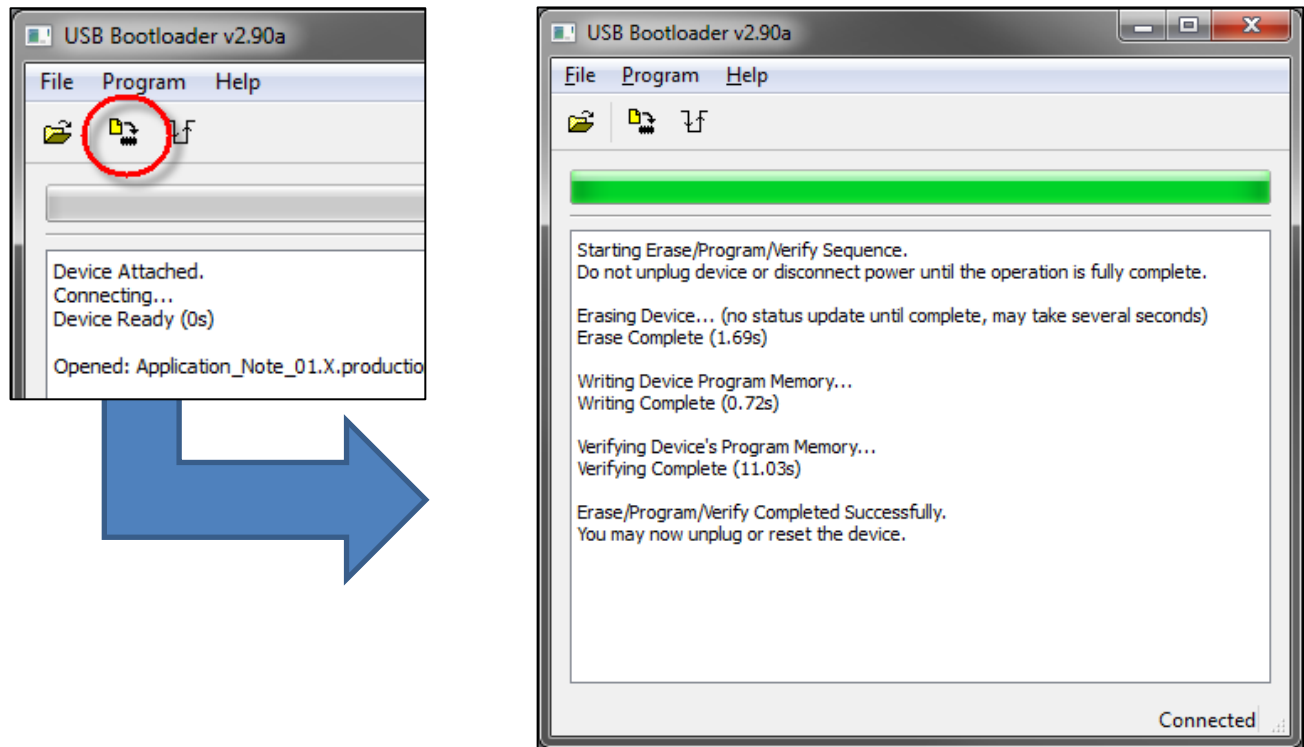
La ventana del Bootloader indicará la conexión establecida con el aguijón:

11. Hacer Clic en Abrir y Cargar el archivo **Application Note 14.X.production.hex**

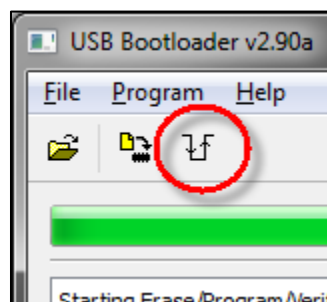


El archivo, depende de la plataforma de hardware.

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 presionar el Push-Button 1 se muestren en pantalla los 8 caracteres personalizados.