



Aguijón

Notas de aplicación

Nota de aplicación 39:

Waveform generator

Descripción:

Mediante el uso de los puertos de expansión como bus SPI controlar el DAC MCP4921, y por medio de este generar diferentes tipos de ondas.

Herramientas:

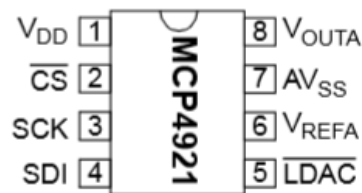
1. Aguijón 3.0, Aguijón 4.0 ó Aguijón 4.1
2. MPLAB X®
3. Aguijón HID bootloader
4. Cable USB 'A' to 'A'
5. Librerías HammerHead.
6. MCP4921 o MCP4922
7. Osciloscopio

Pasos:

1. MCP4921

Convertidor Digital Analógico de 12Bits de resolución, Comunicado a través de un bus SPI.

8-Pin PDIP, SOIC, MSOP

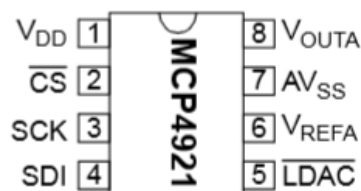


MCP4921 Pin No.	Symbol	Function
1	V _{DD}	Positive Power Supply Input (2.7V to 5.5V)
2	$\overline{\text{CS}}$	Chip Select Input
3	SCK	Serial Clock Input
4	SDI	Serial Data Input
5	LDAC	Synchronization input used to transfer DAC settings from serial latches to the output latches.
6	V _{REFA}	DAC _A Voltage Input (AV _{SS} to V _{DD})
7	AV _{SS}	Analog ground
8	V _{OUTA}	DAC _A Output



2. Diagrama de conexión

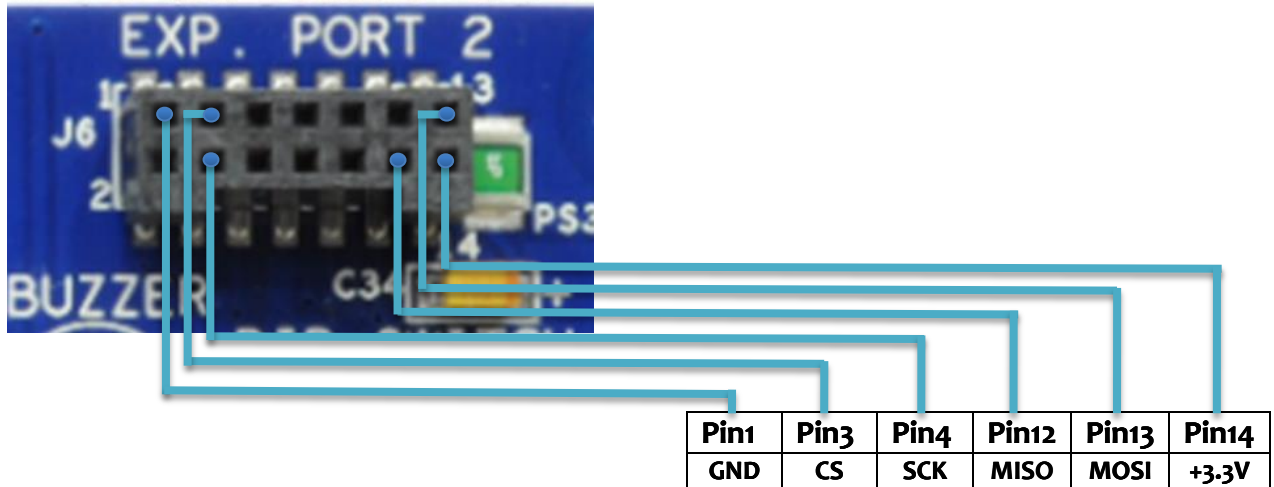
8-Pin PDIP, SOIC, MSOP



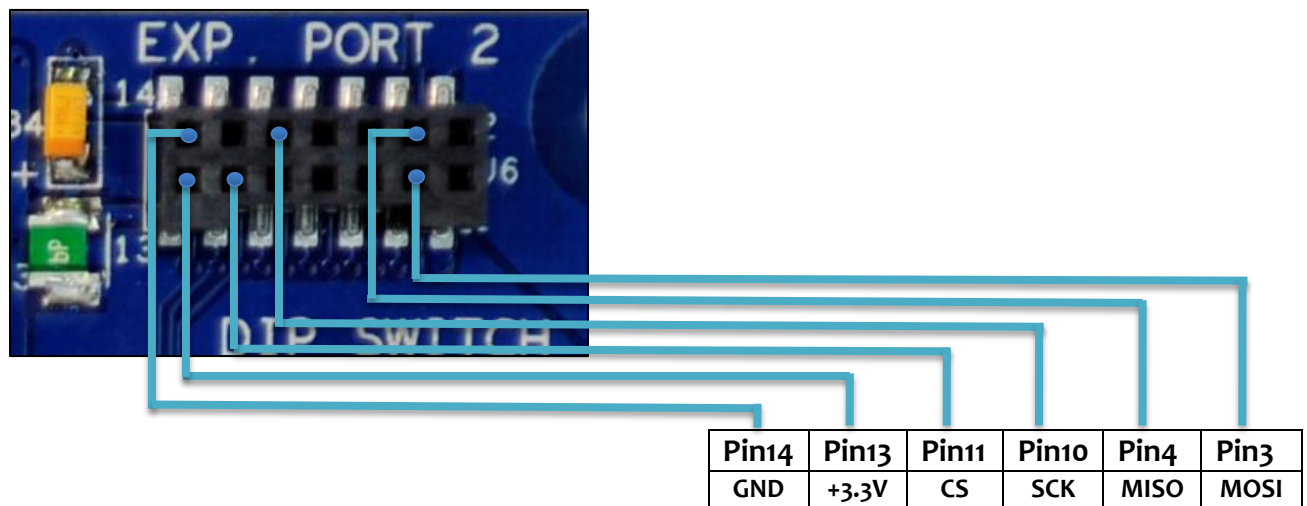
*Los pines **LDAC** y **AVss** deben ser conectados a GND, mientras que los pines **Vdd** y **VREFA** deberá ser conectado a VDD

- **VOUTA** será nuestra salida y es ahí donde se deberá conectar nuestro osciloscopio.

Puerto de expansión 2, Aguijon 4.0

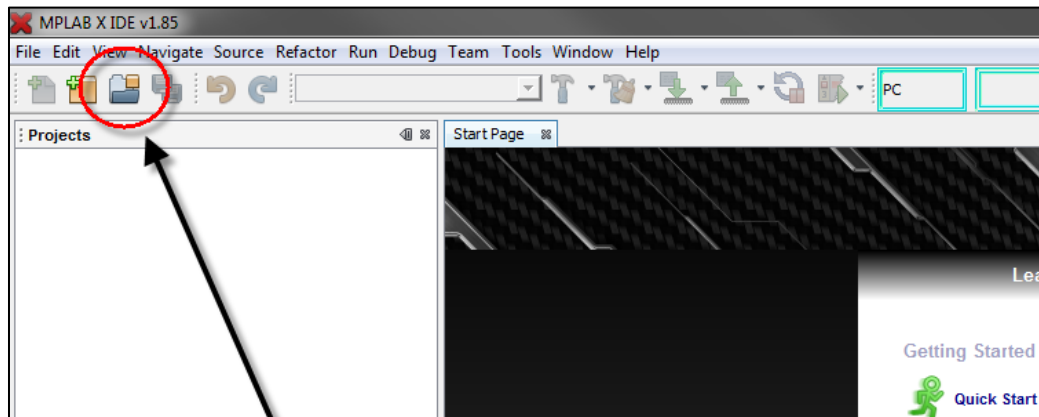


Puerto de expansión 2, Aguijon 4.1



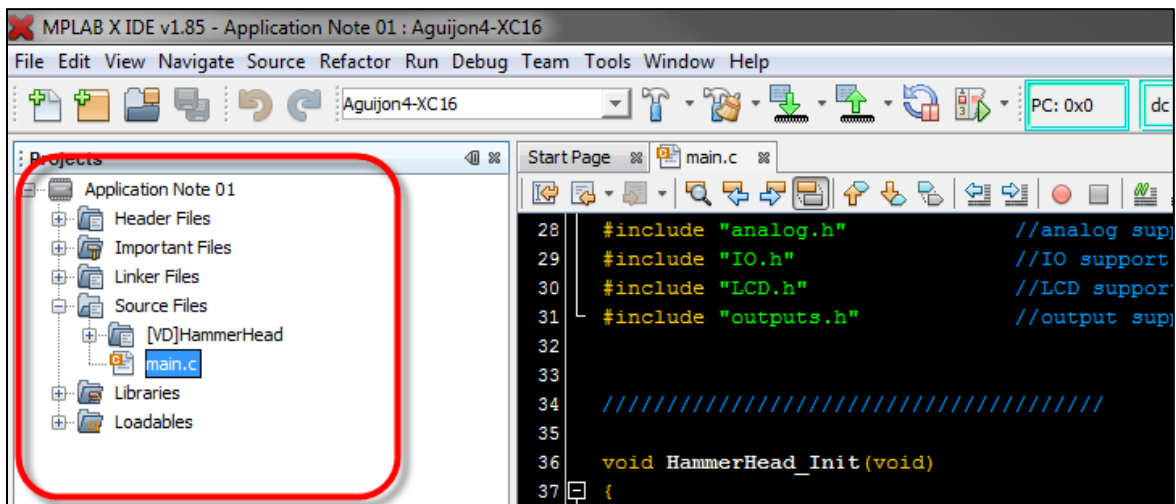
* Ver archivo **Cambios REV4.1**

3. Abrir MPLAB X® y cargar el archivo del proyecto: **Application Note 39.X**



Haz 'clic' aquí y
abre el proyecto

4. Abrir el archivo **main.c**



1. Ir a la línea #52

Utilizaremos la siguiente función:

```
45      /*LCD*/  
46      LCD_Init(LCD_MODE_1);  
47  
48      /*SPI*/  
49      IO_SPIInit();  
50  
51      /*Timer*/  
52      Timer1_Init();  
53      Timer1_Enable();  
54  
55  
56  #if defined(USE_LCD_EXTRA_FEATURES)  
57      LCD_BacklightFadeIn();  
58  #else
```

Timer1_Init ();

- Función que Inicializa los registros de control del Timer 1 y establece un Prescaler de 1:1

2. Ir a la línea #105 y #108

Utilizaremos las siguientes funciones:

```
46 LCD_Init(LCD_MODE_1);
47
48 /*SPI*/
49 IO_SPIInit();
50
51 /*Timer*/
52 Timer1_Init();
53 Timer1_Enable();
54
55
56 #if defined(USE_LCD_EXTRA_FEATURES)
57     LCD_BacklightFadeIn();
58 #else
59     LCD_BacklightSet(BLIGHT_LVL_4);
60 #endif
```

Timer1_Eneable ();

- Función que habilita el módulo Timer 1;

Timer1_Disable ();

- Función que deshabilita el módulo Timer 1;

5. Ir a la línea #83.

Vamos a utilizar la siguiente función:

```
76     LCD_PutStr(1,0,"Vinagron Digital",TRUE);
77     LCD_PutStr(2,0,"Application Note 39",FALSE);
78     delayms(1000);
79     LCD_Clear();
80
81     for(;;){
82
83         keyboard = SW_Read();
84
85         switch (keyboard)
86         {
87             case 1:
88                 WAVE_Type(SINE);           //Set sine wave
89                 LCD_PutStr(1,0,"SINE WAVE",TRUE);
90                 break;
```

SW_Read ();

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

6. Ir a la línea #88

Utilizaremos la siguiente función:

```
81     for(;;){
82
83         keyboard = SW_Read();
84
85         switch (keyboard)
86         {
87             case 1:
88                 WAVE_Type(SINE);           //Set sine wave
89                 LCD_PutStr(1,0,"SINE WAVE",TRUE);
90                 break;
91             case 2:
92                 WAVE_Type(TRIANGLE);       //Set triangle wave
93                 LCD_PutStr(1,0,"TRIANGLE WAVE",TRUE);
94                 break;
95             case 3:
```

WAVE_Type(int type);

- Función que establece el tipo de onda que se desea generar; Donde:
type = Tipo de onda (SINE, TRIANGLE, RAMP, SQUARE)

7. Ir a la línea #89

Utilizaremos la siguiente función:

```
82
83     keyboard = SW_Read();
84
85     switch (keyboard)
86     {
87         case 1:
88             WAVE_Type(SINE);           //Set sine wave
89             LCD_PutStr(1,0,"SINE WAVE",TRUE);
90             break;
91         case 2:
92             WAVE_Type(TRIANGLE);       //Set triangle wave
93             LCD_PutStr(1,0,"TRIANGLE WAVE",TRUE);
94             break;
95         case 3:
96             WAVE_Type(RAMP);           //Set ramp wave
```

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 #106.

Utilizaremos la siguiente función:

```
99         case 4:
100             WAVE_Type(SQUARE);           //Set square wave
101             LCD_PutStr(1,0,"SQUARE WAVE",TRUE);
102             break;
103         }
104
105         /*Set frequency from 0 to 1000 hz*/
106         input_freq = ADC_Range(0,1000);
107         WAVE_Frequency(input_freq);
108
109         /*Create string for the LCD*/
110         sprintf(lcdMSG,"%i HZ",input_freq);
111         LCD_PutStr(2,0,lcdMSG,FALSE);
112
113         delayms(5);
```

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).

9. Ir a la línea #107

Utilizaremos la siguiente función:

```
100         WAVE_Type(SQUARE);        //Set square wave
101         LCD_PutStr(1,0,"SQUARE WAVE",TRUE);
102         break;
103     }
104
105     /*Set frequency from 0 to 1000 hz*/
106     input_freq = ADC_Range(0,1000);
107     WAVE_Frequency(input_freq);
108
109     /*Create string for the LCD*/
110     sprintf(lcdMSG,"%i HZ",input_freq);
111     LCD_PutStr(2,0,lcdMSG,FALSE);
112
113     delayms(5);
114 }
```

WAVE_Frequency (UINT32 frequency);

- Función que ajusta la frecuencia de la onda generada a través del DAC; donde:
frequency= Frecuencia de oscilación en Hertz (valor entero de 1 a 1000)

10. Ir a la línea #110

Utilizaremos la siguiente función:

```
103     }
104
105     /*Set frequency from 0 to 1000 hz*/
106     input_freq = ADC_Range(0,1000);
107     WAVE_Frequency(input_freq);
108
109     /*Create string for the LCD*/
110     sprintf(lcdMSG,"%i HZ",input_freq);
111     LCD_PutStr(2,0,lcdMSG,FALSE);
112
113     delayms(5);
114 }
115 return 0;
116 }
```

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 #113.

Utilizaremos la siguiente función:

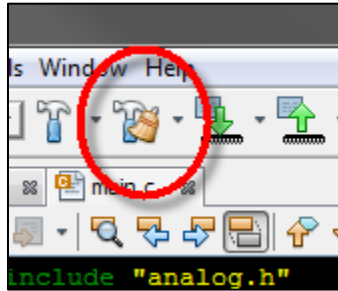
```
102         break;
103     }
104
105     /*Set frequency from 0 to 1000 hz*/
106     input_freq = ADC_Range(0,1000);
107     WAVE_Frequency(input_freq);
108
109     /*Create string for the LCD*/
110     sprintf(lcdMSG,"%i HZ",input_freq);
111     LCD_PutStr(2,0,lcdMSG,FALSE);
112
113     delayms(5);
114 }
115 return 0;
116 }
```

Delayms (ms);

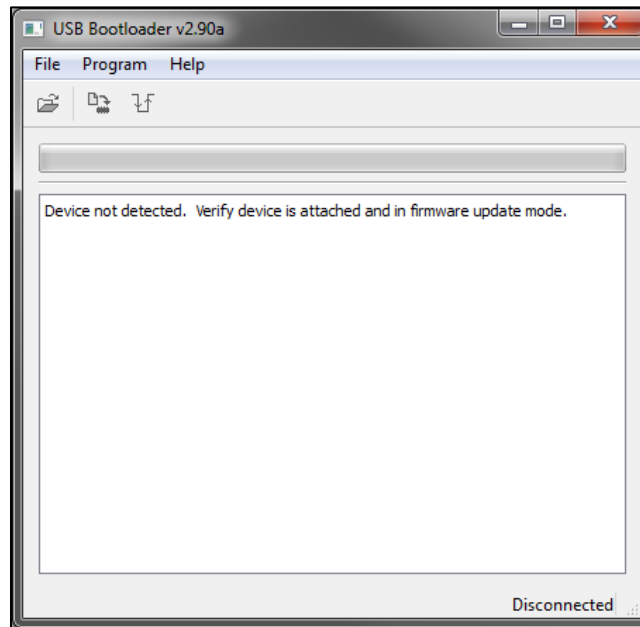
- Función que Pausa el programa por un tiempo determinado (en milisegundos); donde:
ms = el número de milisegundos que se desea pausar el programa.

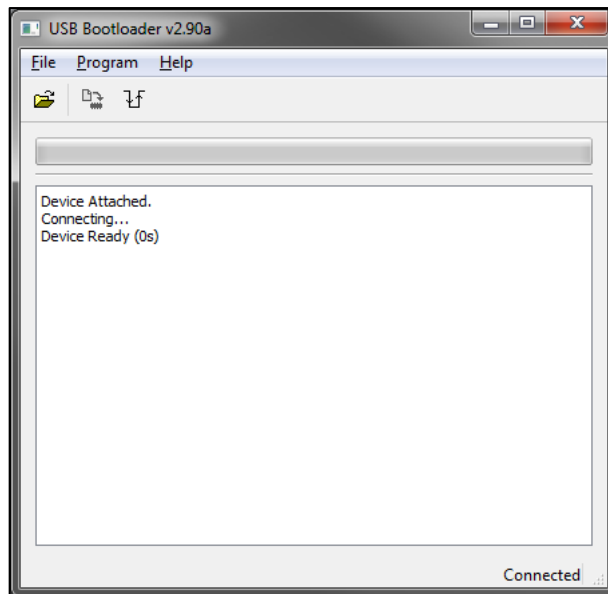
12. Compilar y programar

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



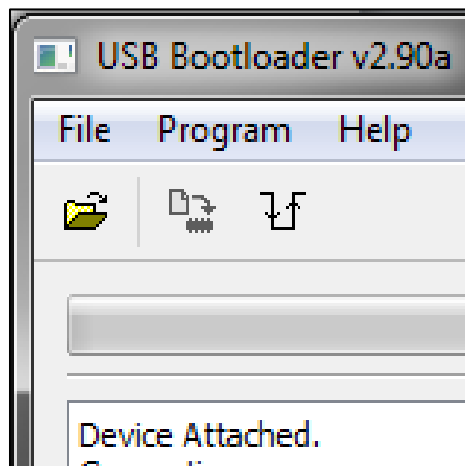
Cuando aparezca la ventana del bootloader, presione el Push-Button número 1 y conecte la fuente de voltaje o encienda el Aguijón y mantenga el PB1 presionado hasta que los LEDs empiecen a parpadear.





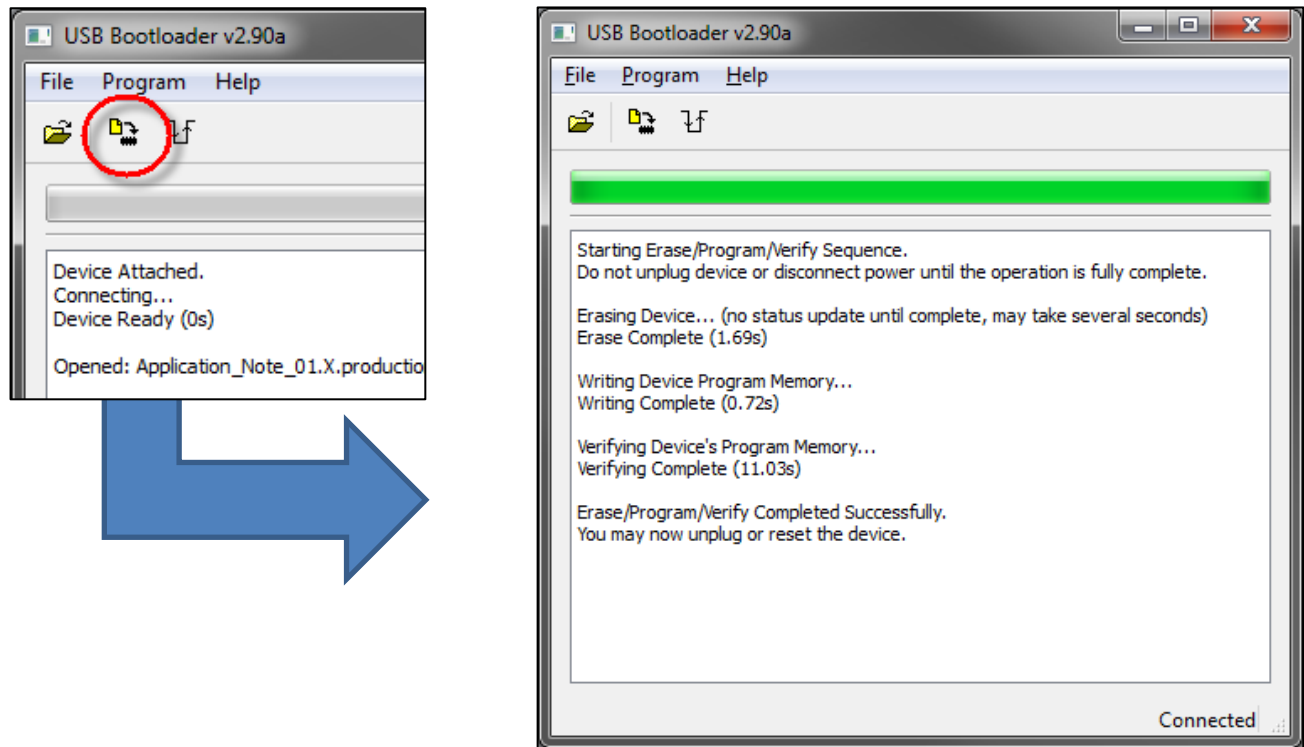
La ventana del Bootloader indicará la conexión establecida con el aguijón:

13. Hacer Clic en Abrir y Cargar el archivo **Application Note 39.X.production.hex**

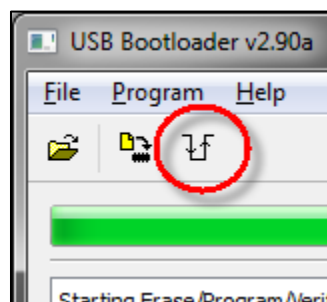


El archivo, depende de la plataforma de hardware.

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 presionar los diferentes Push Buttons se generen diferentes tipos de ondas, a las cuales se les deberá de poder ajustar la frecuencia de oscilación a través del valor introducido en el puerto ADC, dicho valor deberá ser desplegado continuamente en la pantalla LCD al igual que el tipo de onda generada..