



*Aguijón*

---

*Notas de aplicación*

## **Nota de aplicación 32:**

### *Read the frequency of the AC signal*

#### **Descripción:**

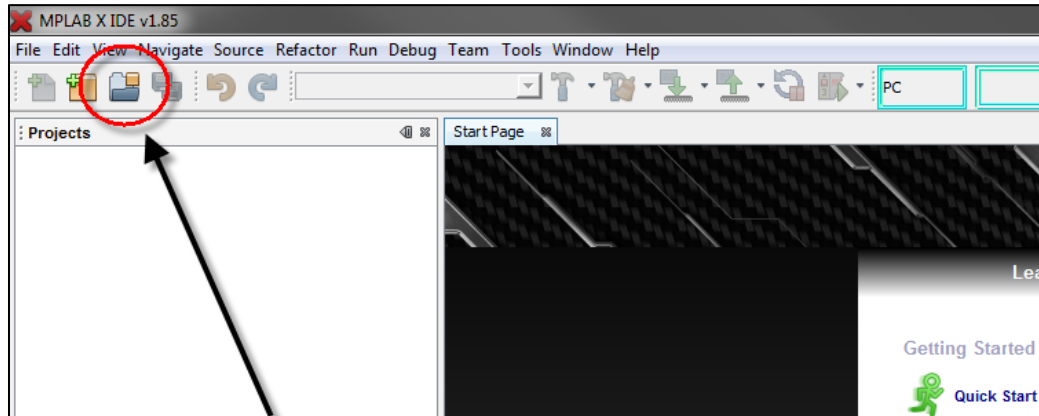
*Medir la frecuencia de oscilación en las líneas de 120v AC*

#### **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. Resistencia de potencia 33kΩ

## Pasos:

1. Abrir MPLAB X® y cargar el archivo del proyecto: **Application Note 32.X**



Haz 'clic' aquí y  
abre el proyecto

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



### 3. Ir a la línea #54

Utilizaremos la siguiente función:

```
47      /*LCD*/  
48      LCD_Init(LCD_MODE_1);  
49  
50      /*PWM*/  
51      OC_PWMChannelConfig(1); //Assigns PWM to open collector output #1  
52  
53      /*Interrupts*/  
54      Interrupts_Init();  
55  
56  
57      #if defined(USE_LCD_EXTRA_FEATURES)  
58          LCD_BacklightFadeIn();  
59      #else  
60          LCD_BacklightSet(BLIGHT_LVL_4);  
61      #endif
```

#### **Interrupts\_Init();**

- Función que configura los registros necesarios para la utilización de interrupciones externas, configura el periférico de entrada Opto-Aislada #3 como la entrada de la interrupción.
- Dicha función deberá ser llamada antes que cualquier otra relacionada con las interrupciones externas.

4. Ir a la línea #81.

Utilizaremos la siguiente función:

```
74 HammerHead_Init(); //initialize [VD]HammerHead
75 LCD_IntroAnimation();
76 LCD_PutStr(1,0,"Vinagron Digital",TRUE);
77 LCD_PutStr(2,0,"Application Note 32",FALSE);
78
79 for(;;){
80
81     frequency=AC_Frequency(); //Read AC Frequency
82
83     /*Create a string for the LCD*/
84     sprintf(lcdMSG1,"AC Frequency= %.2f",frequency);
85     LCD_PutStr(1,0,lcdMSG1,TRUE);
86
87     delayms(150); //Delay 150 milliseconds
88 }
```

#### **AC\_Frequency();**

- Esta función lee la frecuencia introducida en la entrada Opto-Aislada #3; Donde:  
Devuelve un valor de tipo flotante, proporcional a la frecuencia leída.
- Esta función está diseñada para lectura de bajas frecuencias, como la que se obtiene de las tomas de AC, Para mediciones de altas frecuencias vea el ejemplo de **Input Capture** (Application Note 31).

5. Ir a la línea #84

Utilizaremos la siguiente función:

```
77 LCD_PutStr(2,0,"Application Note 32",FALSE);
78
79 for(;;){
80
81     frequency=AC_Frequency();    //Read AC Frequency
82
83     /*Create a string for the LCD*/
84     sprintf(lcdMSG1,"AC Frequency= %.2f",frequency);
85     LCD_PutStr(1,0,lcdMSG1,TRUE);
86
87     delayms(150);    //Delay 150 milliseconds
88
89 }
90 return 0;
91 }
```

**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.

6. Ir a la línea #85

Utilizaremos la siguiente función:

```
78
79     for(;;){
80
81         frequency=AC_Frequency();    //Read AC Frequency
82
83         /*Create a string for the LCD*/
84         sprintf(lcdMSG1,"AC Frequency= %.2f",frequency);
85         LCD_PutStr(1,0,lcdMSG1,TRUE);
86
87         delayms(150);    //Delay 150 milliseconds
88
89     }
90     return 0;
91 }
92
```

**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 #97

Utilizaremos la siguiente función:

```
78
79     for(;;){
80
81         frequency=AC_Frequency();           //Read AC Frequency
82
83         /*Create a string for the LCD*/
84         sprintf(lcdMSG1,"AC Frequency= %.2f",frequency);
85         LCD_PutStr(1,0,lcdMSG1,TRUE);
86
87         delayms(150);    //Delay 150 milliseconds
88
89     }
90     return 0;
91 }
92
```

#### ***Delayms (ms);***

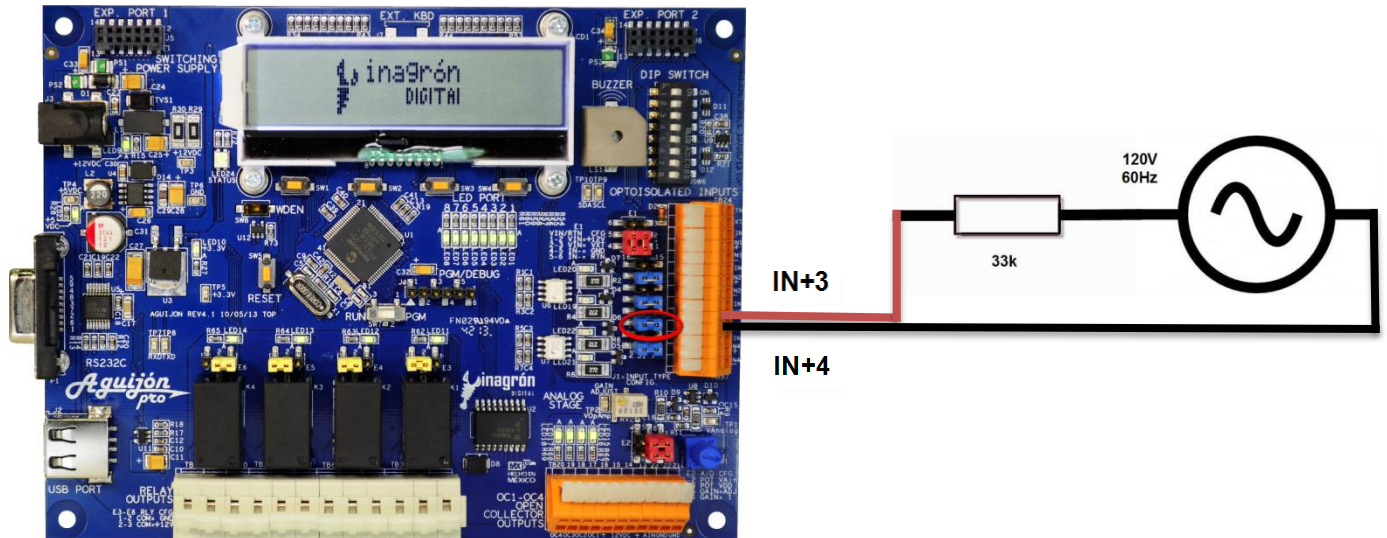
- Función que Pausa el programa por un tiempo determinado (en milisegundos); donde: **ms** = el número de milisegundos que se desea pausar el programa.



## Diagrama de conexión

\* Ver archivo Cambios REV4.1

### Diagrama Aguijon 4.0 y 4.1

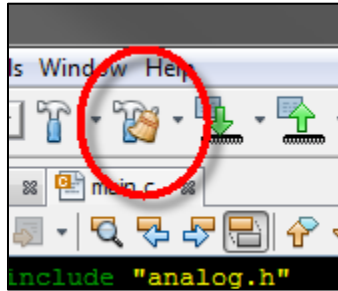


### \*NOTA

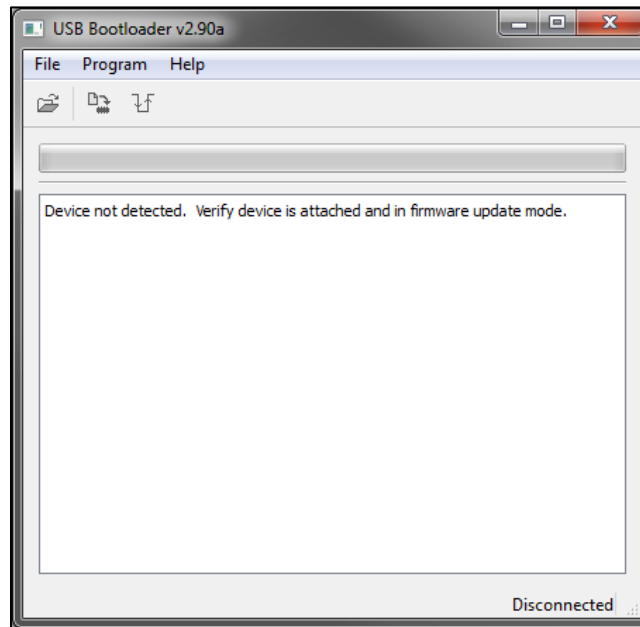
**Es necesario retirar todos los jumpers de la serie de pines 'E' y 'J', a excepción Del jumper J7-8**

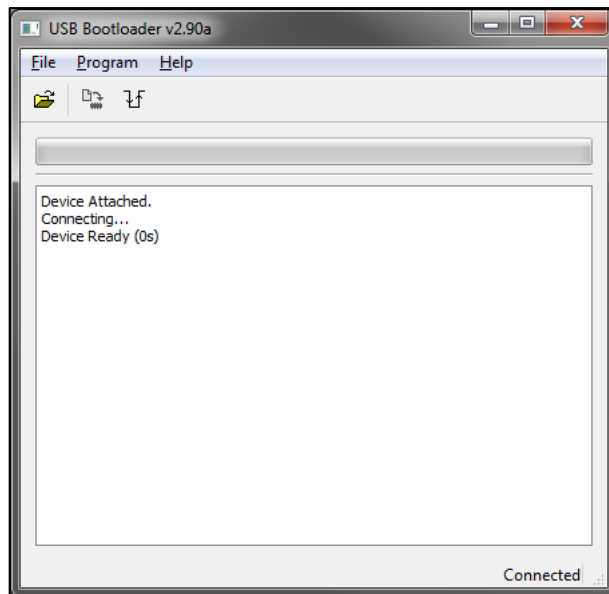
## 8. Compilar y programar

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



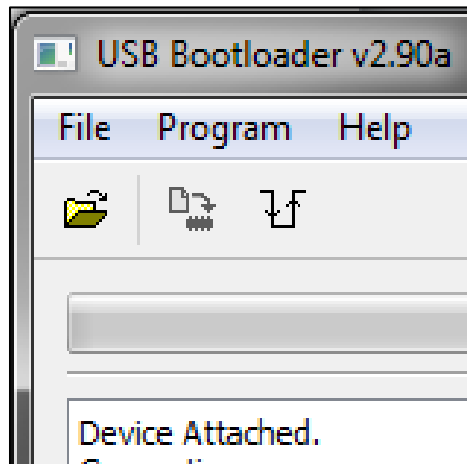
Cuando aparezca la ventana del bootloader, presione el Push-Button número 1 y conecte la fuente de voltaje o encienda el Aguijón y mantenga el PB1 presionado hasta que los LEDs empiecen a parpadear.





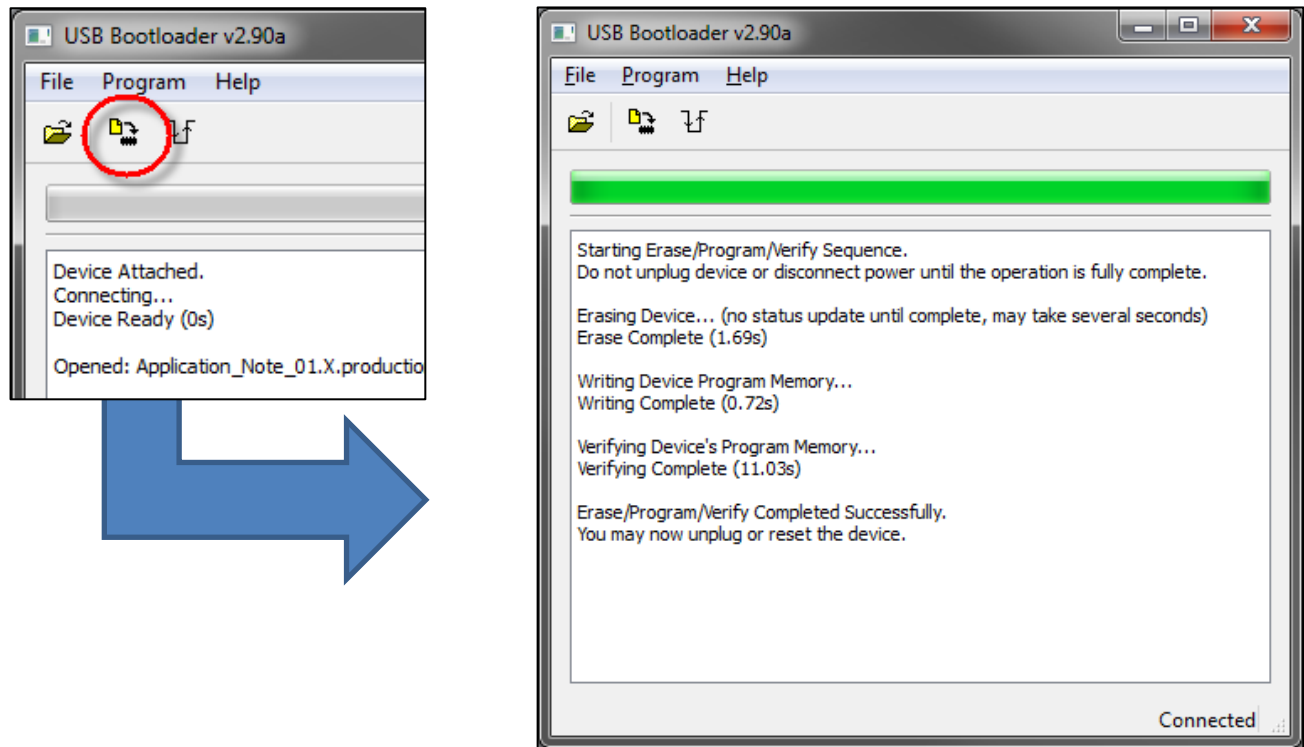
La ventana del Bootloader indicará la conexión establecida con el aguijón:

9. Hacer Clic en Abrir y Cargar el archivo **Application Note 32.X.production.hex**

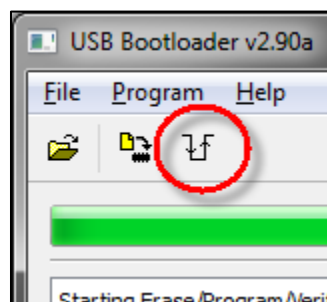


El archivo, depende de la plataforma de hardware.

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.

10. Para verificar el funcionamiento del programa verifique que al introducir a través de la resistencia de potencia la corriente suministrada por las líneas de 110V, se despliegue sobre la pantalla LCD la frecuencia de la línea.