



Aguijón

Notas de aplicación

Nota de aplicación 34:

Servo-Motor

Descripción:

Realizar un control de la posición en grados de hasta 6 servomotores simultáneamente.

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. 1 Uln2003 (o similar)
7. 6 Resistencias 10k Ω
8. 1 Diodo rectificador 1N4001 (o similar)
9. Servomotores 180° 5v

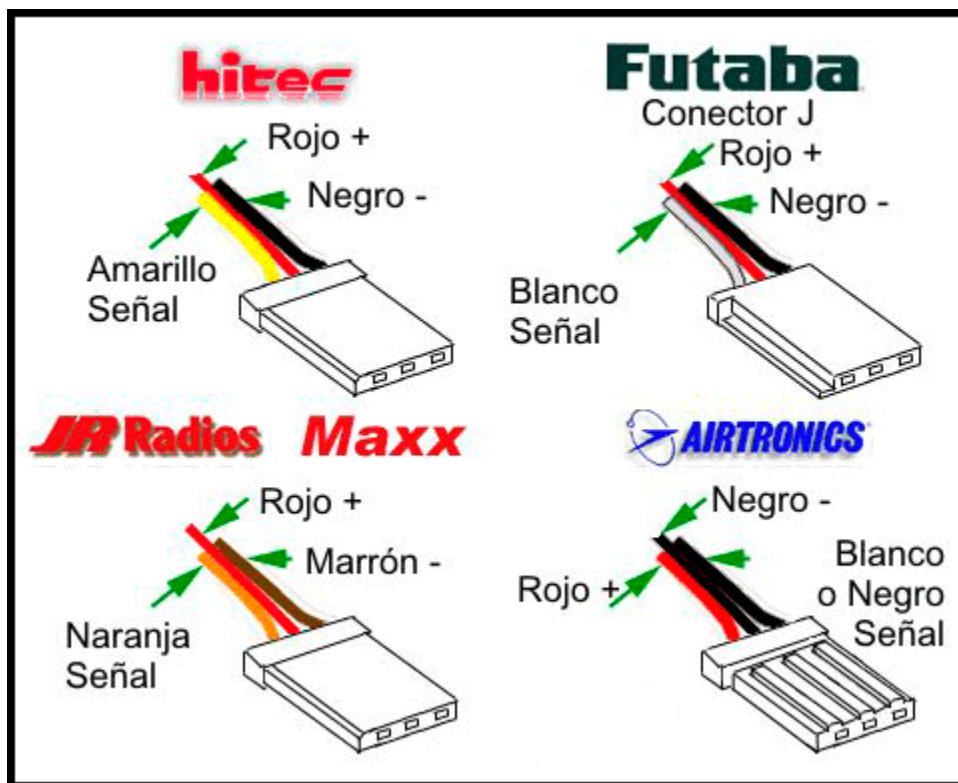
Pasos:

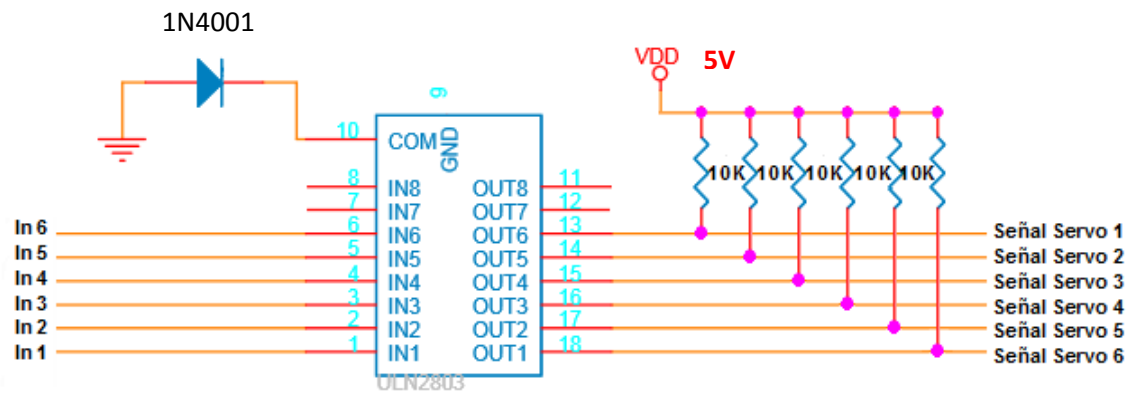
1. SERVOMOTORES

Un servomotor (también llamado servo) es un dispositivo similar a un motor de corriente continua que tiene la capacidad de ubicarse en cualquier posición dentro de su rango de operación, y mantenerse estable en dicha posición.

Los servomotores hacen uso de la modulación por ancho de pulsos (PWM) para controlar la dirección o posición de los motores de corriente continua.

2. Diagrama de conexión

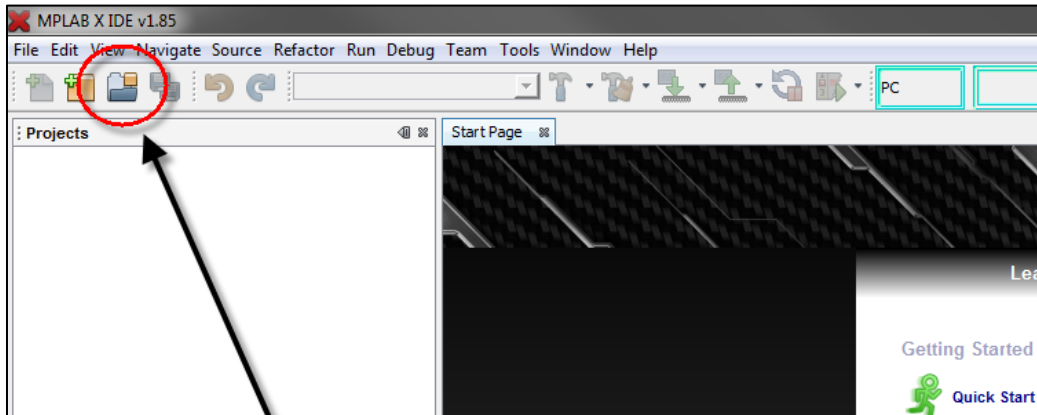




Aguijon 4.0 & 4.1

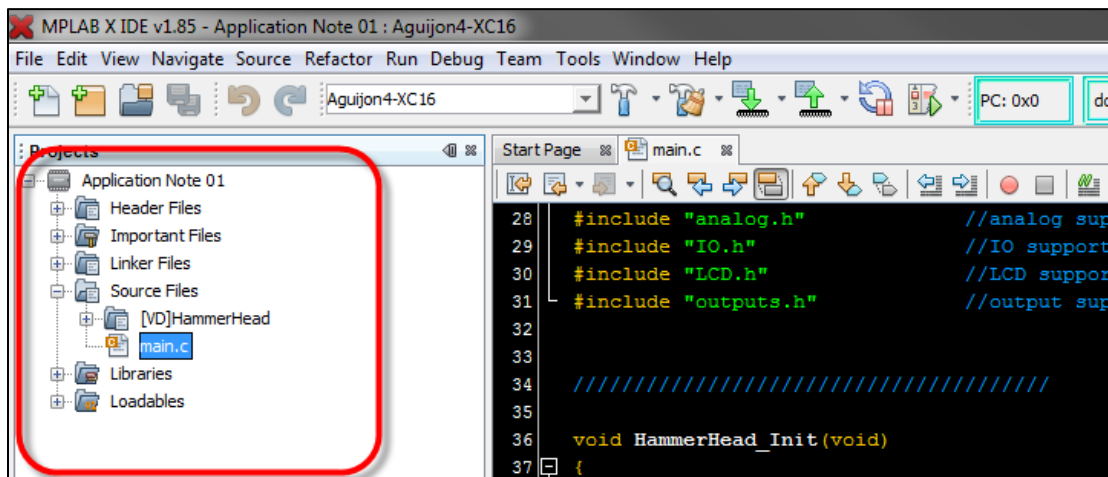
Pin en Aguijon	Pin en ULN2003
Puerto de Expansión 2 PIN3	IN 1
Puerto de Expansión 2 PIN4	IN 2
Puerto de Expansión 2 PIN5	IN 3
Puerto de Expansión 2 PIN6	IN 4
Puerto de Expansión 2 PIN9	IN 5
Puerto de Expansión 2 PIN10	IN 6

3. Abrir MPLAB X® y cargar el archivo del proyecto: **Application Note 34.X**



Haz 'clic' aquí y
abre el proyecto

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



5. Ir a la línea #50.

Utilizaremos la siguiente función:

```
43      /*Inputs*/
44      ADC_Init();
45
46      /*LCD*/
47      LCD_Init(LCD_MODE_1);
48
49      /*Servomotor*/
50      SERVO_Init();
51      SERVO_Config(1);
52      SERVO_Config(2);
53
54      #if defined(USE_LCD_EXTRA_FEATURES)
55          LCD_BacklightFadeIn();
56      #else
57          LCD_BacklightSet(BLIGHT_LVL_4);
```

SERVO_Init ();

- Función que inicializa el timer 2 con un Prescaler de 8, el cual será la base para manejar los PWM que controlaran los servomotores.
Es necesario llamar esta función, antes que cualquier otra relacionada con Servomotores.

6. Ir a la línea #51.

Utilizaremos la siguiente función:

```
44     ADC_Init();
45
46     /*LCD*/
47     LCD_Init(LCD_MODE_1);
48
49     /*Servomotor*/
50     SERVO_Init();
51     SERVO_Config(1);
52     SERVO_Config(2);
53
54     #if defined(USE_LCD_EXTRA_FEATURES)
55         LCD_BacklightFadeIn();
56     #else
57         LCD_BacklightSet(BLIGHT_LVL_4);
58     #endif
```

SERVO_Config (unsigned char num);

- Función que configura el Servomotor seleccionado, asignándole al pin de salida específico, un PWM ; donde:
num = El número de servomotor que se desea configurar (Numero entero del 1 al 6).

7. Ir a la línea #81.

Utilizaremos la siguiente función:

```
74 LCD_PutStr(2,0,"Application Note 34",FALSE);
75
76 for(;;){
77
78     /* 'i' increases from 0 to 180, while 'j' decreases 180 to 0 */
79     for(i=0,j=180 ; i<=180 ; i++,j--)
80     {
81         SERVO_Set(1,i);          //Move servo 1
82         SERVO_Set(2,j);          //Move servo 2
83
84         /*Create a string for the LCD*/
85         sprintf(lcdMSG,"Serv1=%3i Serv2=%3i",i,j);
86         LCD_PutStr(1,0,lcdMSG,TRUE);
87
88         delayms(30);              //Delay 30 Milliseconds
```

SERVO_Set (char num, int degrees);

- Función que mueve el servomotor seleccionado a la posición indicada de 0° a 180° ;
donde:
num = El número de servomotor que se desea mover (Numero entero del 1 al 6).
degrees= Posición en grados a donde se desea mover el servomotor (Numero entero del 0 a 180)

8. Ir a la línea #85

Utilizaremos la siguiente función:

```
78      /* 'i' increases from 0 to 180, while 'j' decreases 180 to 0 */
79      for(i=0,j=180 ; i<=180 ; i++,j--)
80      {
81          SERVO_Set(1,i);          //Move servo 1
82          SERVO_Set(2,j);          //Move servo 2
83
84          /*Create a string for the LCD*/
85          sprintf(lcdMSG,"Serv1=%3i  Serv2=%3i",i,j);
86          LCD_PutStr(1,0,lcdMSG,TRUE);
87
88          delayms(30);              //Delay 30 Milliseconds
89      }
90
91      /* 'i' decreases from 180 to 0, while 'j' increases 0 to 180 */
92      for(i=180,j=0 ; i>=0 ; i--,j++)
```

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.

9. Ir a la línea #86

Utilizaremos la siguiente función:

```
79     for(i=0,j=180 ; i<=180 ; i++,j--)
80     {
81         SERVO_Set(1,i);          //Move servo 1
82         SERVO_Set(2,j);          //Move servo 2
83
84         /*Create a string for the LCD*/
85         sprintf(lcdMSG,"Serv1=%3i  Serv2=%3i",i,j);
86         LCD_PutStr(1,0,lcdMSG,TRUE);
87
88         delayms(30);              //Delay 30 Milliseconds
89     }
90
91     /* 'i' decreases from 180 to 0, while 'j' increases 0 to 180 */
92     for(i=180,j=0 ; i>=0 ; i--,j++)
93     {
```

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).

10. Ir a la línea #88.

Utilizaremos la siguiente función:

```
81     SERVO_Set(1,i);           //Move servo 1
82     SERVO_Set(2,j);           //Move servo 2
83
84     /*Create a string for the LCD*/
85     sprintf(lcdMSG,"Serv1=%3i  Serv2=%3i",i,j);
86     LCD_PutStr(1,0,lcdMSG,TRUE);
87
88     delayms(30);               //Delay 30 Milliseconds
89 }
90
91     /* 'i' decreases from 180 to 0, while 'j' increases 0 to 180 */
92     for(i=180,j=0 ; i>=0 ; i--,j++)
93     {
94         SERVO_Set(1,i);           //Move servo 1
95         SERVO_Set(2,j);           //Move servo 2
```

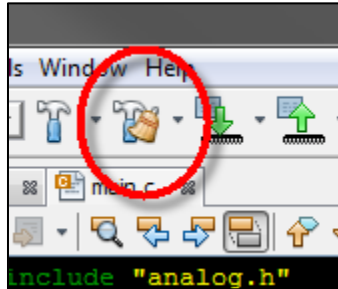
Delayms (ms);

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

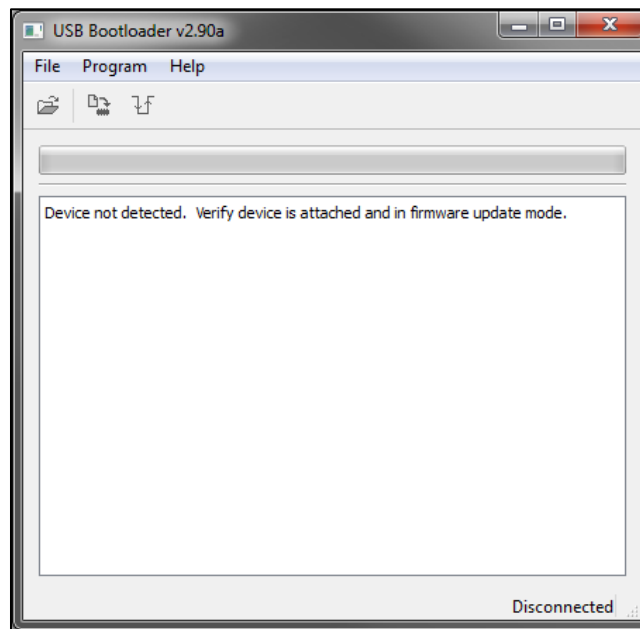
11. Compilar y programar

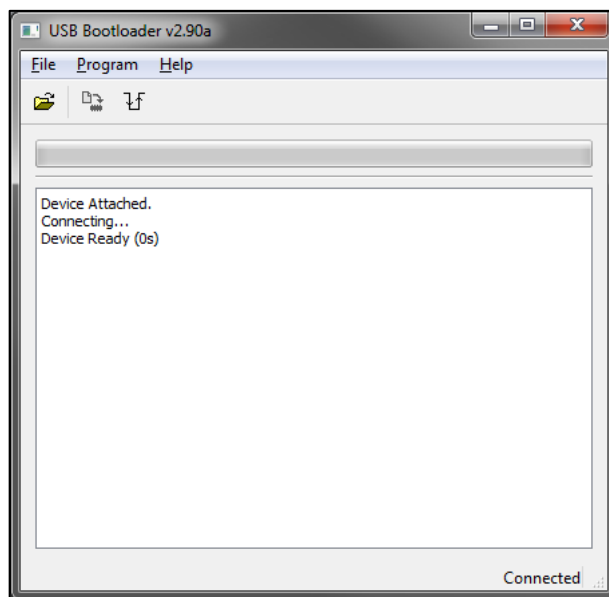
Asegúrese de tener el Aguijón apagado.

Hacer clic en el icono de compilar



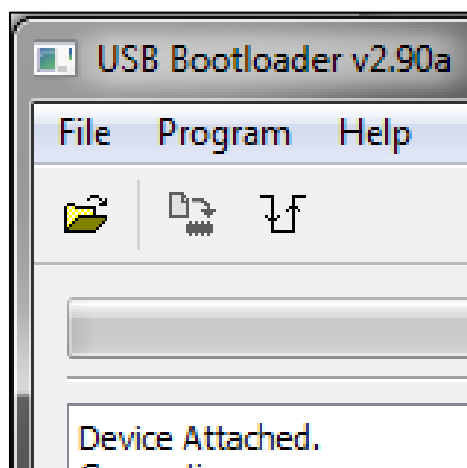
Cuando aparezca la ventana del bootloader, presione el Push-Button número 1 y conecte la fuente de voltaje o encienda el Aguijón y mantenga presionado hasta que los LEDs empiecen a parpadear.



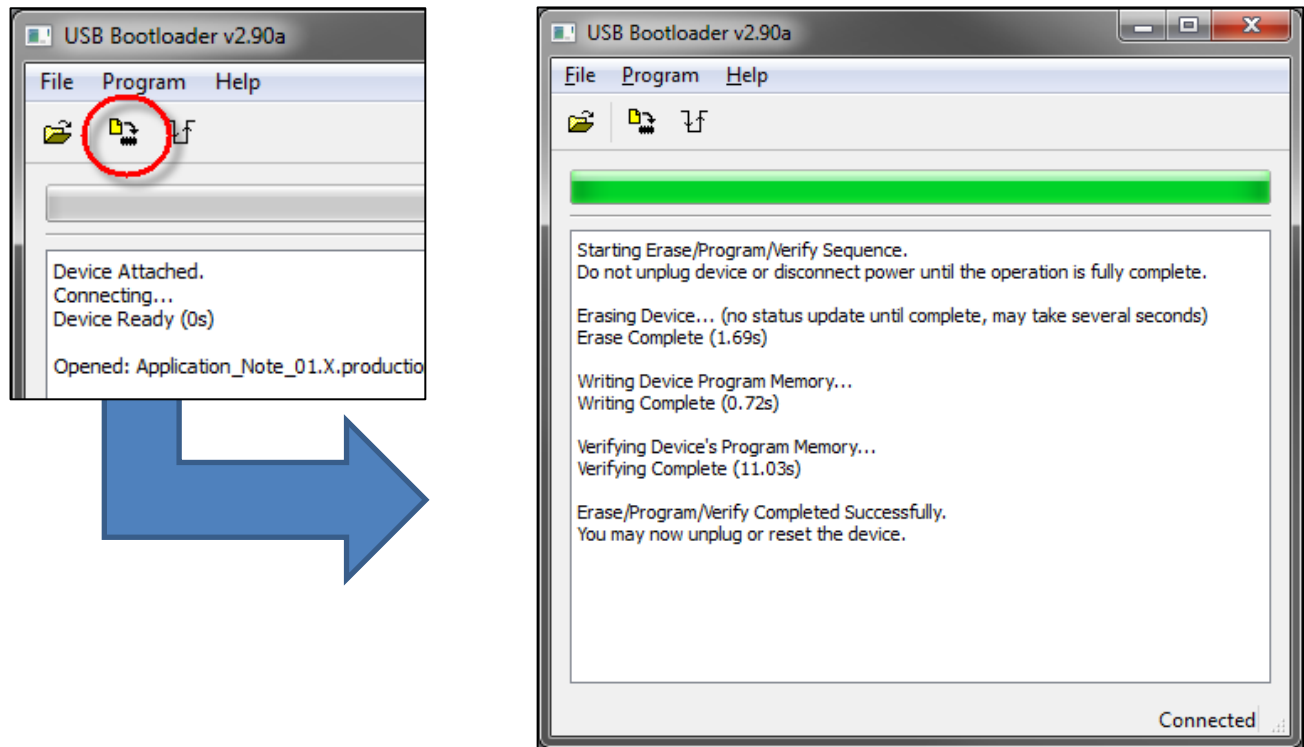


La ventana del Bootloader indicara la conexión establecida con el Aguijón:

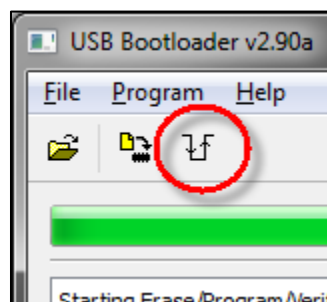
12. Hacer clic en Abrir y cargar el archivo **Application Note 34.X.production.hex**



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



Hacer clic en el icono 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 con la tarjeta.

13. Para verificar el funcionamiento del programa verificar que al realizar la conexión correcta, el servomotor numero 1 gire de 0° a 180° y viceversa, mientras que el servomotor numero 2 gire de 180° a 0° y viceversa.