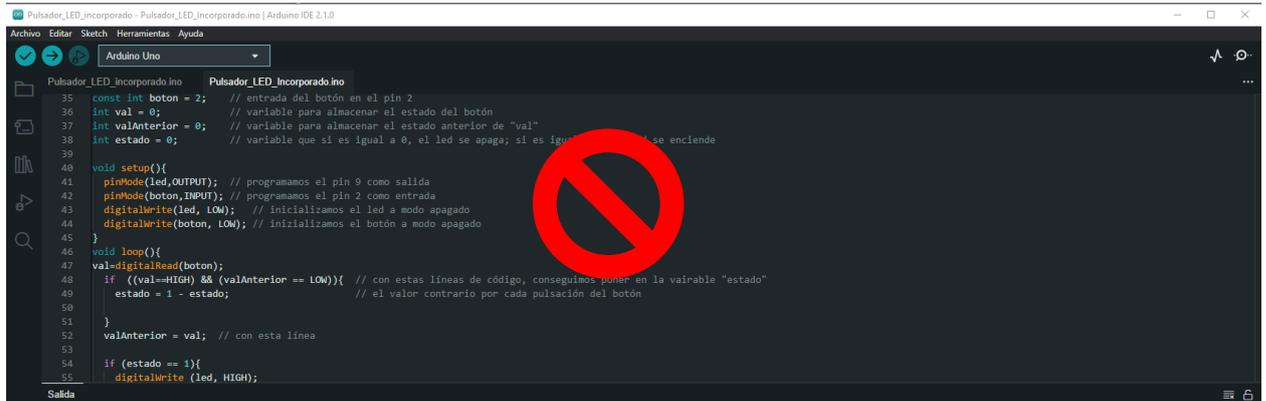


Tutorial MMJoy2 – Configuración y uso.

Una de las grandes ventajas del uso del software MMJoy2 para la construcción de cockpits (o cualquier dispositivo de juego tipo HID) es la ausencia de programación con código Arduino.



El sistema MMJoy2 permite convertir las placas basadas en los chips de ATMEL que más adelante se relacionan, en un joystick para juegos, además de poder configurar y calibrar todos aquellos componentes que queramos usar en su construcción.

Hay que entender que MMJoy2 tiene varias funciones que, en su conjunto, nos facilitan la creación de dispositivos de juego a través de la instalación de su firmware, con el procedimiento de bootloader, en las placas.

Si bien Windows 10 ya reconoce estas como dispositivos "plug & play", no será capaz de saber qué componentes tiene instalados y como queramos utilizarlos en la composición de nuestro cockpit. Para ello es necesario configurarlo todo a través de la interface de MMJoy2.

En resumen, y para un mejor entendimiento de lo que es y para que se utiliza el software MMJoy2 sería:

- a) Para instalar en las placas compatibles el firmware de MMJoy2.
- b) Para configurar, gestionar y calibrar todos los componentes que estén conectados a esas placas.

Las placas que son compatibles con el sistema aquí descrito son aquellas que llevan los chips, fabricados por ATMEL, siguientes:

at90usb646 | at90usb1286 | tmega32u4

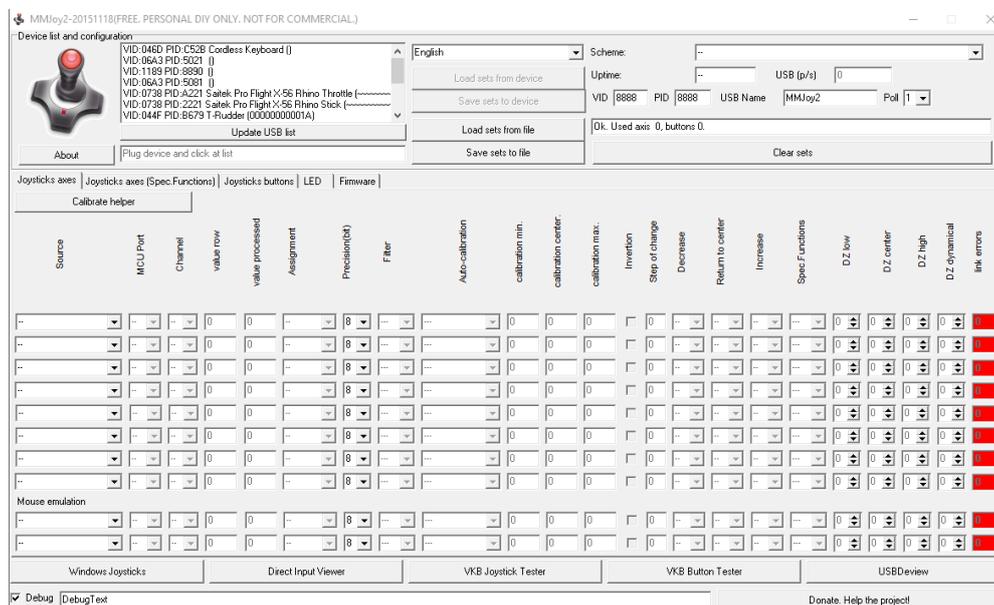
La fuente de alimentación es de 5 voltios y la frecuencia de reloj de 16 MHz. Estos chips tienen soporte USB (Windows los va a reconocer inmediatamente como dispositivos HID).

Existen estos chips sueltos para poder trabajar con ellos directamente, pero lo más recomendable y sencillo es adquirir aquellas placas que ya los llevan integrados, como las siguientes:

- Sparkfun Pro-Micro.
- Arduino Leonardo.
- Arduino Micro.
- PJR Teensy 2.0
- PJR Teensy ++ 2.0.

En el primer anexo de este tutorial, están las imágenes de las placas del listado anterior con las equivalencias que el programa MMJoy2 le asigna a cada entrada de las mismas, lo cual es necesario conocer para configurar más adelante nuestra interface.

A continuación, intentaré explicar los pasos para llevar a cabo todo esto y aclarar aquellos puntos donde yo me he encontrado con algún escollo o “problema”.



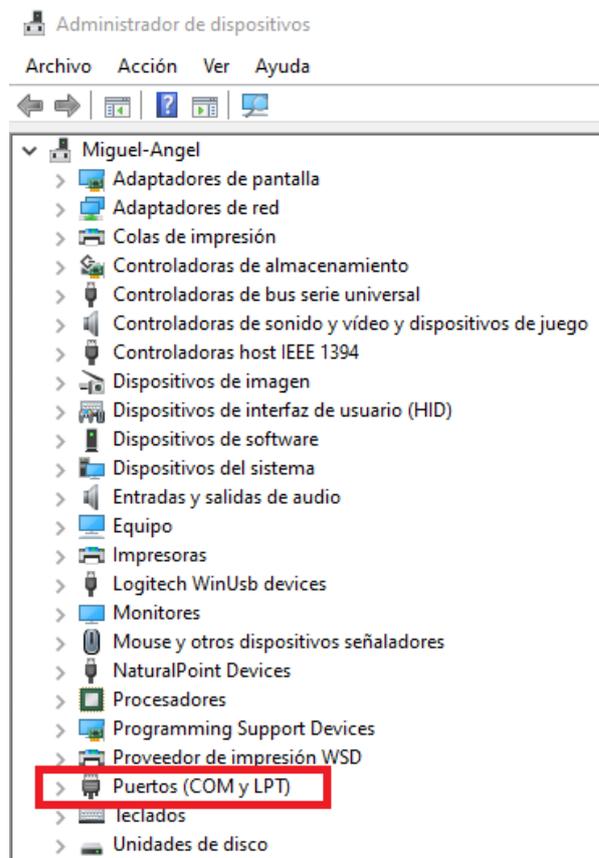
Bootloader a la placa Arduino.

Hay que cambiar el firmware que la placa lleva instalado dentro, de esta forma, la comunicación entre el programa MMJoy2 y la placa será el adecuado.

El primer paso es conectar la placa a un puerto USB.

Para conocer el número correcto del puerto al que se conecta, podemos utilizar la herramienta de Windows "Administrador de dispositivos", la cual deberemos tener abierta y a la vista en todo este proceso.

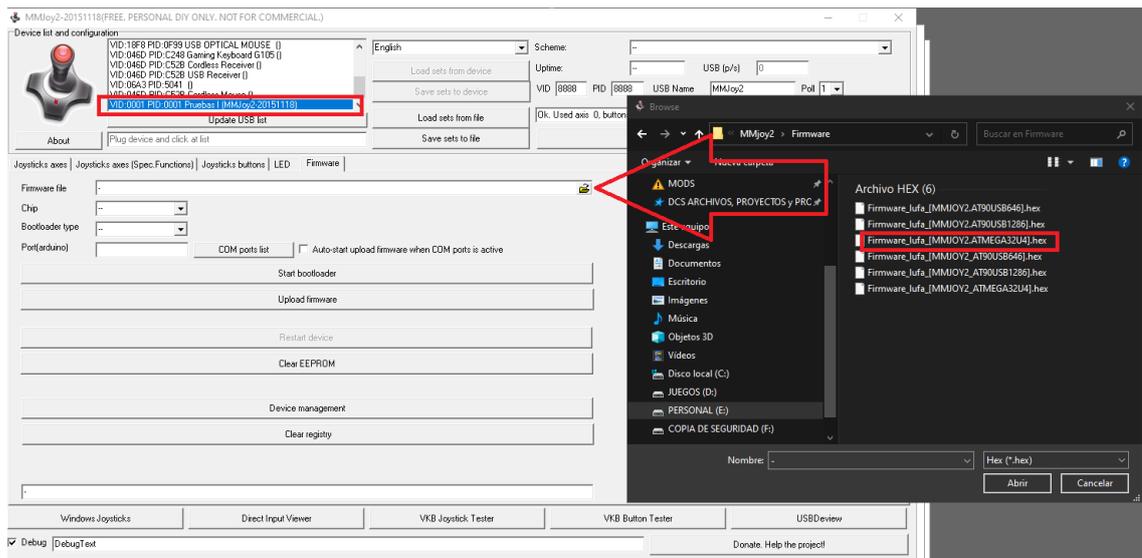
Si bajamos por el listado de la izquierda hasta el apartado donde pone "Puertos (COM y LPT)" y lo desplegamos, veremos cómo está conectada nuestra placa Leonardo en un puerto COM determinado (en mi caso el COM5).



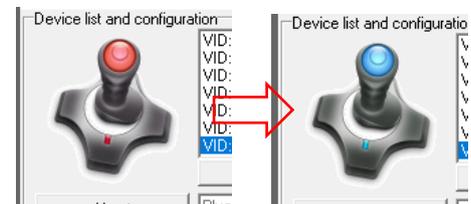
Con el administrador de dispositivos a la vista y el programa MMJoy2 abierto, vamos a proceder a cargar en la placa Arduino el firmware que permitirá la correcta comunicación y configuración a través del MMJoy2.

En la parte superior izquierda de la interface de MMJoy2 hay un cuadro donde se muestra un listado de todos aquellos dispositivos USB conectados, vamos a buscar y seleccionar uno que vendrá nombrado con algo parecido a esto:

"VID:8888 PID 8888 (MMJoy2-20151118)"



Cuando lo tengamos seleccionado, el icono de MMJoy2 de la izquierda cambiará de rojo a azul, eso quiere decir que el programa está conectado a ese dispositivo.



Luego configuraremos una serie de casillas en la sección "Firmware" de la forma siguiente:

- **Firmware file.**

Desplegaremos la carpeta y buscaremos en la localización donde hayamos guardado el MMJoy2 una carpeta denominada "Firmware", dentro de ella encontraremos varios perfiles en formato .hex, a nosotros nos interesa el "Firmware_lufa_[MMJOY2.ATMEGA32U4]" que coincide con el tipo de chip que tiene nuestro Arduino.

- **Chip.**

Aquí elegiremos el "ATmega32U4".

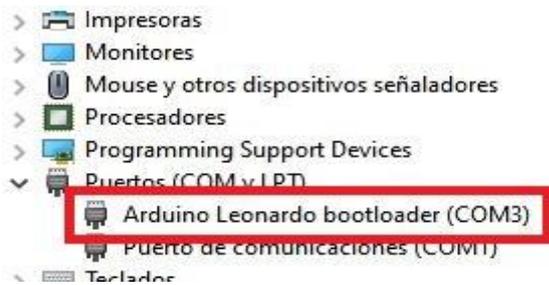
- **Bootloader type.**

Seleccionaremos el nombre de la marca de nuestra placa "Arduino".

- **Port(Arduino).**

En este apartado es importante tener a mano la aplicación del "Administrador de dispositivos", pues en ella podremos ver el puerto COM exacto en el que está conectada la placa.

Aquí hay que tener una precaución (por lo menos así me sucedió a mi) y es que, el puerto COM al que se conecta la placa Arduino

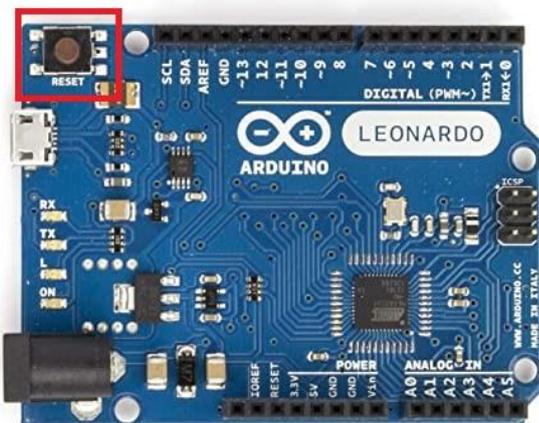


en un principio según la conectamos (por ejemplo, COM5), cambia al entrar en modo bootloader (por ejemplo, a COM6).

Así que, si nos limitamos a seleccionar de forma automática el puerto COM5 en esta casilla, nos dará un error al no poder cargar el firmware del MMJoy2 en la placa.

Lo que yo hice fue observar en el "Administrador de dispositivos" el puerto COM al que se quedaba la placa conectada cuando ésta entraba en el modo bootloader (en mi caso concreto era el COM6), y lo escribía a mano en la casilla que nos ocupa, de ese modo, cuando resetee la placa, automáticamente se conectó al puerto correcto y me permitió volcar el firmware en ella.

Una vez que hayamos hecho todos estos pasos, vamos a meter a la placa en el modo bootloader, es un proceso ligeramente diferente si se trata de la Leonardo R3 o de la Pro Micro.

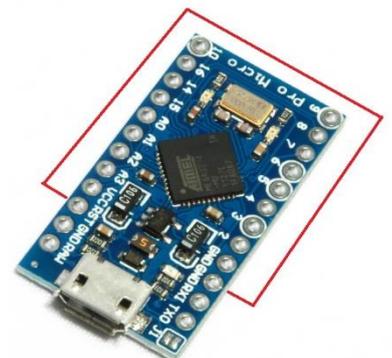


En la Leonardo es tan sencillo como activar el botón de RESET que la misma tiene, al hacerlo con el "Administrador de dispositivos" a la vista, veremos como desaparece unos segundos para entrar a continuación en el modo bootloader.

En ese momento y durante unos segundos, se podrá instalar en la placa el firmware de MMJoy2 pulsando el botón

"Upload firmware", se abrirá una ventana del sistema donde se podrá ver el proceso de carga.

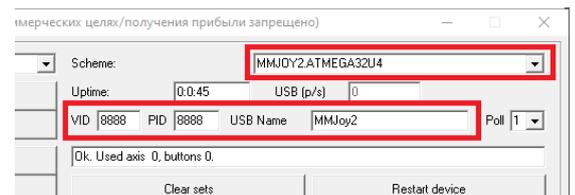
Para llevar a cabo el reset de la placa Pro Micro que no dispone de ese botón, es necesario puentear los pines GND y RST de la placa, recordar tener a la vista el "Administrador de dispositivos".



El dispositivo desaparecerá unos segundos del listado de USB de la interface de MMJoy2 para aparecer luego, volveremos a seleccionarlo para conectarla otra vez.

En ese momento, en nuestra placa Arduino se habrá instalado el firmware apropiado para poder configurar todos los componentes que queramos colocar en la placa, así como para cambiar esta de nombre, cosa que es sumamente importante para que no entre en conflicto con otras que queramos utilizar.

El cambio de nombre se hace en las casillas de la parte superior derecha de la siguiente forma:

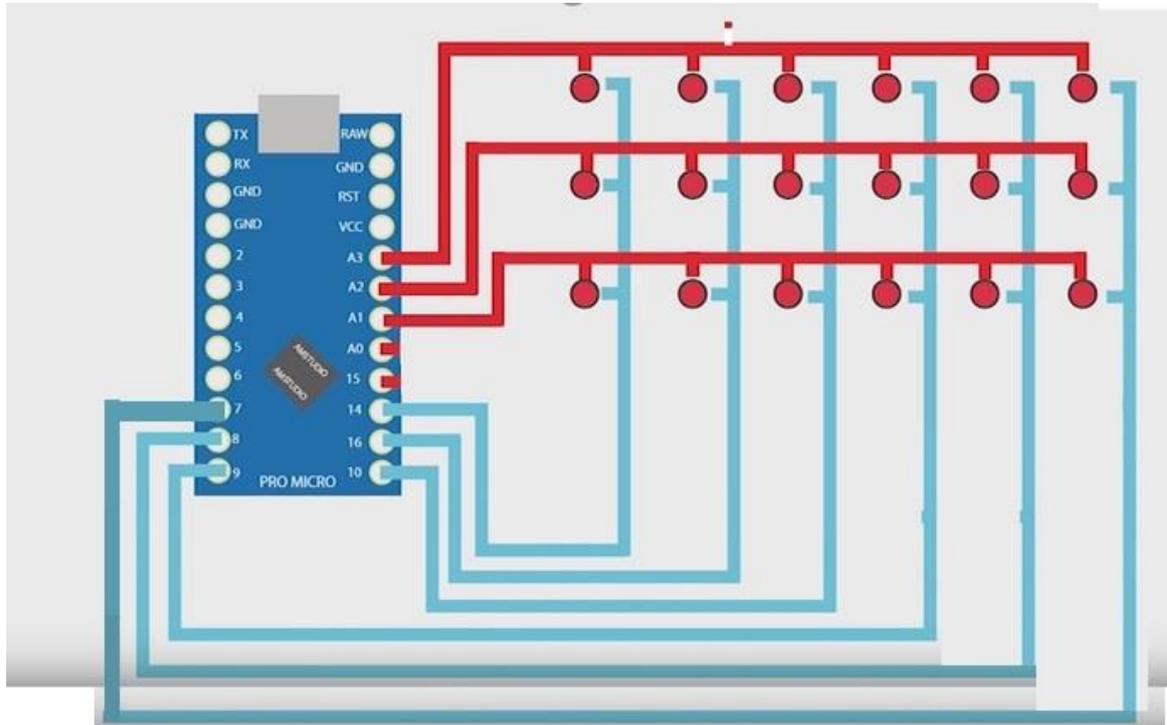


- **Scheme.**
Seleccionamos MMJOY2.ATMEGA32U4
- **VID.**
Por defecto vendrá "8888", debemos cambiarlo, en mi caso a "0001"
- **PID.**
Por defecto vendrá "8888", también hay que modificarlo, en mi caso "0001"
- **USB Name.**
Aquí pondremos un nombre intuitivo pues será éste por el que lo reconozcamos en Windows 10, en mi caso "Prueba"

Estos son los pasos básicos para instalar el firmware de MMJoy2 en nuestra placa para poder configurarla a nuestro gusto con todos aquellos dispositivos que necesitemos, hay que tener cuidado cuando hagamos modificaciones en algún componente para no cambiar el nombre ni los números VID y PID de la placa (si esto pasara, es tan sencillo como repetir los pasos descritos anteriormente, pero nos puede llevar a confusiones a la hora de programar varias placas diferentes).



Cualquier perfil que hagamos se puede guardar a través del botón "Save sets to file", como también recuperar uno ya creado con "Load sets from file".

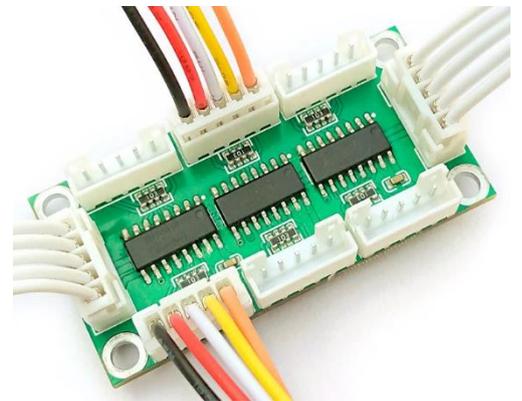


pinos del Arduino posibles, esto se lleva a cabo en la sección "Joysticks buttons", en el recuadro marcado como "Button matrix", si bien es una opción que yo no he considerado pues es más trabajosa que la que a continuación paso a comentar.

MMJoy2 está preparado para poder reconocer dos tipos de tarjetas de expansión (shift registers) que son la 74HC165 y la 4021, en mi caso he utilizado la primera de ellas.

Las tarjetas de expansión son una forma sencilla y eficaz de poder aumentar, de forma considerable, la cantidad de entradas que queramos para nuestros proyectos.

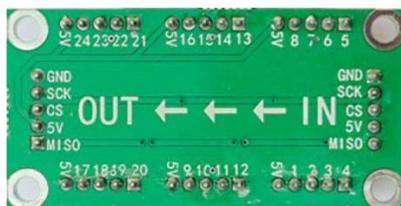
Estas tarjetas suelen venir ensambladas en PCBs las cuales traen tres chips 74HC165 cada una, lo que nos otorga la posibilidad de conectar 24 entradas más por cada tarjeta de expansión que coloquemos.



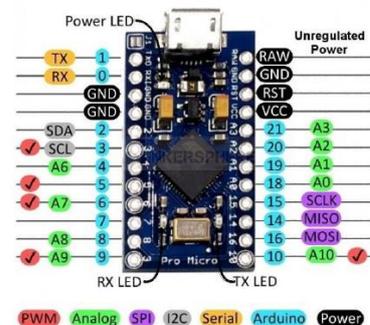
Otra enorme virtud es que se pueden poner en serie unas detrás de otras (la propia placa trae unos conectores de entrada "IN" y de salida "OUT") con lo que poner varias seguidas es muy sencillo.

La tarjeta de expansión dispone de unos pines determinados, las equivalencias entre la tarjeta de expansión, la placa Arduino Pro-Micro y el programa MMJoy2 son las siguientes:

El esquema de conexiones es el siguiente:



| 74HC165 | ARDUINO | MMJoy2 |
|---------|---------|-----------|
| GND | GND | B1 |
| SCK | 15 | B2 - MOSI |
| CS | 16 | B3 |
| 5 V | VCC | |
| MISO | 14 | |



Una vez dicho todo esto, paso a comentar la forma y los parámetros que hay que colocar para que el programa MMJoy2 pueda cargar en la

placa la configuración adecuada para saber lo que tiene que leer y como ha de hacerlo.



En la sección denominada "Shift register" pondremos los siguientes valores:

- En el primer desplegable elegiremos **"74HC165"**.
- En el segundo desplegable elegiremos la cantidad de chips que va a tener conectada la placa, recordar que cada tarjeta de expansión lleva 3 chips, así que, si vamos a colocar un shift register en nuestra placa, el valor que tenemos que colocar en este apartado es **"3"**.

Si quisiéramos colocar más tarjetas de expansión, deberíamos decirle al programa el número total de chips que tiene que reconocer, es decir, pPara dos shift registers poner "6", para tres shift registers poner "9", y así sucesivamente.

- En el desplegable nombrado como **"CS"** tendremos que colocar el valor equivalente que le da MMJoy2 a la entrada (observar equivalencia de conexiones), en mi caso es **"B2"**.
- En el desplegable nombrado como **"MISO"** seleccionaremos el valor que MMJoy2 le da a la entra equivalente, como en el caso anterior, para mí el **"B3"**.

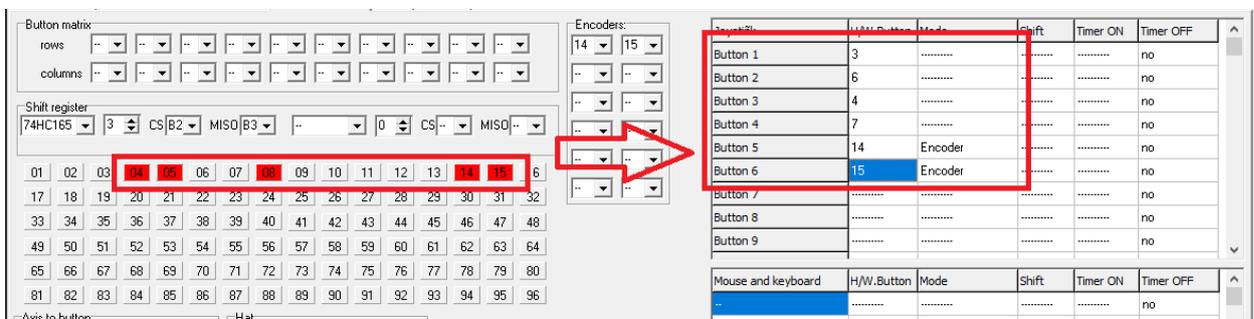
Una vez tengamos configurado el software de esta forma, cargaremos la configuración a nuestra tarjeta Pro Micro con el botón "Save sets to device", esperaremos a que se cargue, luego se desconectará la placa por unos segundos y, al cabo de otros pocos más, volverá a aparecer en el listado.

Para poder ver si reconoce los dispositivos que tenemos conectados, tendremos que volver a seleccionarla en ese listado (si el logo del MMJoy2 aparece en rojo, es que no hay ningún dispositivo seleccionado, si aparece en azul, si lo hay).

Si todo ha ido bien y se ha hecho correctamente, en la matriz de casillas numeradas que hay debajo, se iluminara en rojo aquella entrada que activemos, comprobando que la Pro Micro reconoce y lee el dispositivo de la tarjeta de expansión.

Aclarar que la matriz de 96 cuadros que se muestra en la interface sirve para la comprobación visual de la conexión física de cualquier componente a nuestra tarjeta Arduino a través de los shift registers, es un modo de saber que se activa una señal digital y donde se encuentra.

Estas señales no se trasladan a nuestro hardware de juego, para eso es necesario asignar el número de señal que aparece en la matriz cuadriculada a un botón físico.



Así pues, si bien MMJoy2 lee las entradas, tenemos que programarle el tipo y número de botones en la sección de la derecha donde vienen listados 64 botones con sus características.

Si no hacemos esto, Windows 10 reconocerá la placa como un dispositivo de juego, pero sin botones.

En la parte superior derecha de la ventana del software MMJoy2 hay una casilla que pone, de forma predeterminada "Ok, Used axis 0, buttons 0", esta nos va a servir para que, de un solo vistazo, podamos comprobar cuantas entradas tiene reconocidas la placa en ese momento.

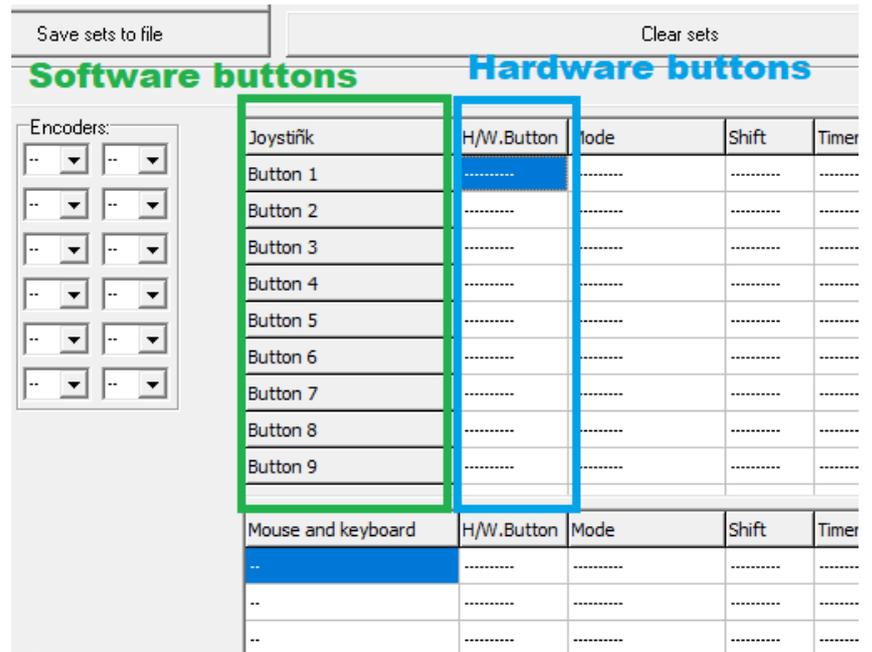
Ok. Used axis 0, buttons 6.

Pongamos como ejemplo que tengo conectado un botón en una de las entradas de nuestro shift register (cableado de forma adecuada) y que, al pulsarlo, en la matriz de casillas numeradas, se ilumina el número "03", esto quiere decir que se detecta una entrada en el pin 3 de la shift register,

pero no tiene por qué coincidir con el botón número 3 de nuestro dispositivo de juego, me explico.

En la parte derecha de la ventana de MMJoy hay un listado de botones numerados del "Button 1" al "Button 64", cada cual tiene cinco características concretas (en la versión de MMJoy2 20161101 cambia un poco y parece que se simplifica), dependiendo de lo que queramos que se simule en el momento de la pulsación del botón físico.

Para que Windows 10 reconozca que nuestro dispositivo de juego personalizado tiene un botón, hay que poner en la casilla correspondiente al "Button 1", en la característica "H/W.Button" el número 3, si os dais cuenta, el desplegable que se abre ahí tiene 96 valores, tantos como salillas numeradas en la matriz de la izquierda.



Lo que estamos haciendo aquí es asignarle la señal eléctrica número 3 a un botón, el botón físico que se va a reconocer como el número 1, y en la siguiente columna se le va a decir al programa MMJoy2 cómo queremos que trate esa señal.

Características de los botones.

A cada asignación de botones virtuales, se le puede asignar un modo de comportamiento que viene determinado por la opción que seleccionemos del desplegable.

En la versión de MMJoy2 2015118, existen cuatro columnas que determinan la forma de comportarse el botón que seleccionemos, se va a simular de una forma concreta.

La primera de esa columna se denomina “**Mode**”, aquí se abre un abanico de posibilidades bastante grande pues, a través de software, vamos a poder simular la señal de un botón físico de formas diferentes.

A saber:

- **Button(norm).** [Sólo versión 20161101]
Aquí podemos simular la pulsación de un botón corriente, se activa cuando pulsamos y se desactiva cuando soltamos.
- **Button(invert).** [Sólo versión 20161101]
Aquí simulamos a través de software la acción contraria a la anterior, cuando pulsamos se desactiva la señal y cuando dejamos de pulsar se activa
- **Switch.**
Aquí podemos simular la acción de un interruptor de palanca, es decir, si tenemos un botón normal conectado podremos sacar una señal cuando lo pulsemos y, a su vez, otra cuando lo soltemos.
Sería como tener conectado un switch de dos posiciones tipo on-on.
Con esta opción, el programa va a activar el botón en cada cambio de estado de la entrada asociada, pero no tiene en cuenta la posición en la que se encuentra nuestro botón.
Es interesante, si seleccionamos esta opción, programar en la columna denominada “Timer ON” uno de los tres delays que nos ofrece, por ejemplo “Timer 1”, más adelante explicaré donde y cómo se configuran estos delays.
- **Switch ON.**
Con esta opción, tan solo se va a detectar el momento en que activemos la entrada, es decir, se va a simular una pulsación cuando entre una señal eléctrica.
Se va a detectar la posición “ON”.
Se puede programar un delay, de los presets existentes, en la opción “Timer ON”.
Se puede asignar la misma entrada a dos botones diferentes siempre que a uno de ellos le pongamos esta opción y al otro la siguiente, por ejemplo:

Button 1 = 5 [Switch ON] y Button 2 = 5 [Switch OFF]).

| Joystiňk | H/W.Button | Mode | Shift | Timer ON | Timer OFF |
|----------|------------|------------|-------|----------|-----------|
| Button 1 | 5 | Switch ON | ----- | Timer 1 | no |
| Button 2 | 5 | Switch OFF | ----- | Timer 1 | no |

- **Switch OFF.**

Esta opción es la contraria a la anterior, se va a poder programar la activación de una señal cuando se deje de activar la entrada, es decir, se va a simular una pulsación en el momento en que dejemos de pulsar el botón.

Se va a detectar la posición "OFF".

Se puede programar un delay, de los presets existentes, en la opción "Timer ON".

Se puede asignar la misma entrada a dos botones diferentes siempre que a uno de ellos le pongamos esta opción y al otro la anterior, por ejemplo:

Button 1 = 5 [Switch ON] y Button 2 = 5 [Switch OFF]).

- **Soft Switch.**

Con la presente opción, se puede simular la acción de un interruptor de palanca para asignárselo a un botón corriente.

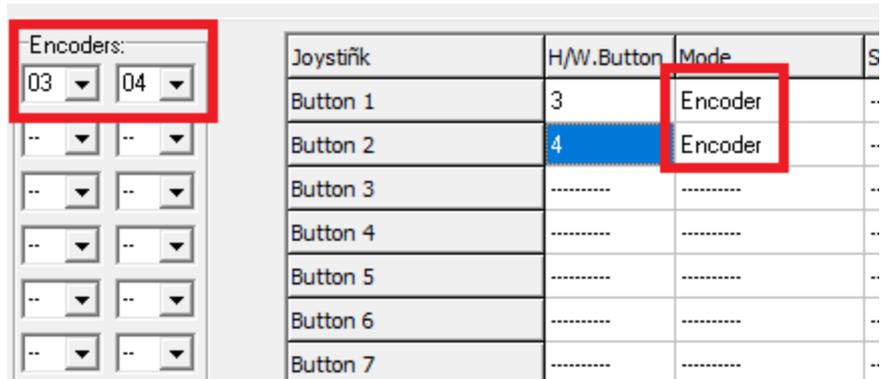
En un botón sin enclavamiento, cuando pulsamos se activa la señal e, inmediatamente al soltar se desactiva.

Se puede asignar esta opción para poder hacer que un botón de esta clase se comporte como uno de palanca de modo que, una vez se pulse el botón, el switch simulado cambiará de posición, y no volverá a cambiar hasta volver a pulsarlo por segunda vez.

- **Encoder.**

Esta opción se activa automáticamente cuando asignamos un número de entrada física en la columna llamada "H/W.Button" si previamente se ha configurado en la sección habilitada para los Encoders.

Más adelante se va a explicar como conectar este tipo de componente.



Dependiendo de la versión de MMJoy2 que se utilice, se supone que, a partir de la casilla "Button 33", no se va a poder seleccionar ninguna de las opciones de las que venimos hablando, es como si a partir de ese botón virtual, no se pudiera simular un comportamiento diferente al de un botón normal, sin embargo, la versión que yo utilizo sí lo permite, desconozco el motivo.

La siguiente es la llamada "**Shift**" y tampoco sé para qué y cómo se utiliza, pero para el correcto funcionamiento de nuestro botón, dejarla en blanco es perfecto.

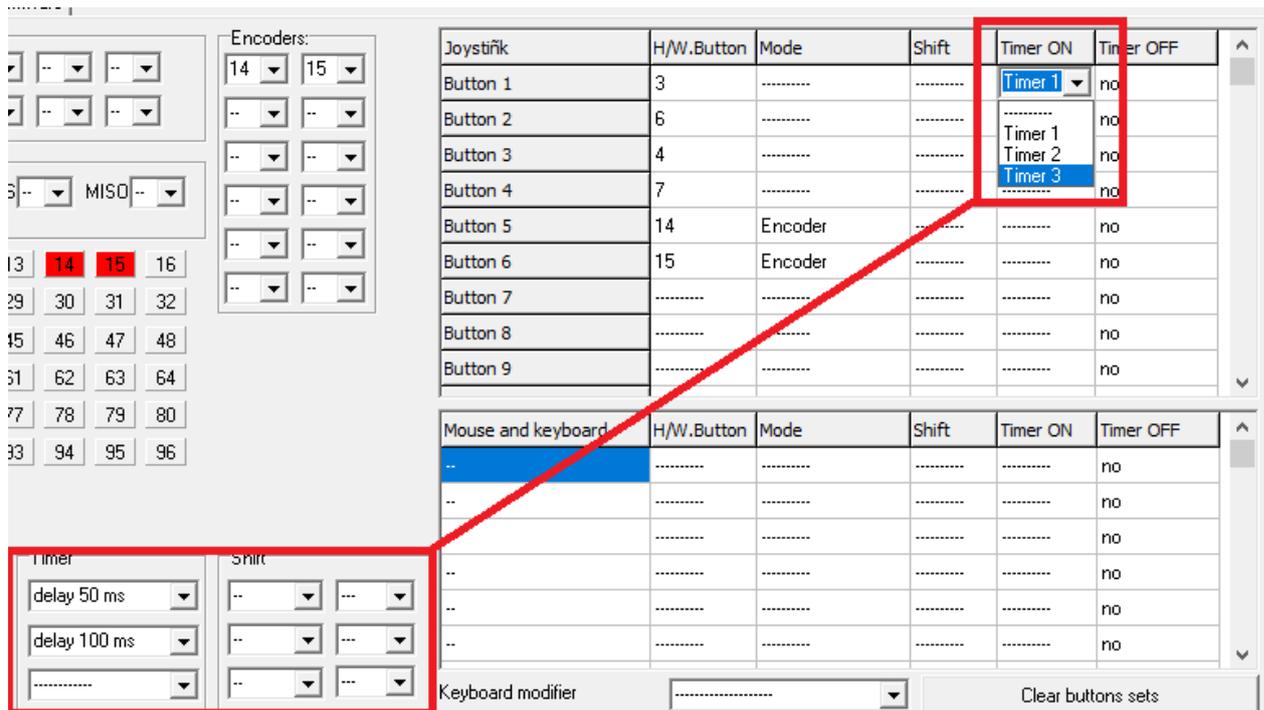
La denominada "**Timer ON**" (en la versión más moderna viene nombrada como "Timer") consigue un retardo a la hora de activar la entrada con respecto a la pulsación, hay tres opciones que son:

- "**Timer 1**", "**Timer 2**" y "**Timer 3**".

En la parte central inferior de la interface de MMJoy2 existe una sección donde se pueden configurar esas tres opciones de "Timer".

Esas opciones van desde los 50 ms hasta los 500 ms, aparte de poder dejarla en blanco, por supuesto.

Una vez programado el retardo en milisegundos de la opción "Timer", podremos escogerla en la característica de la que estamos hablando, tan solo se pueden configurar y escoger tres.



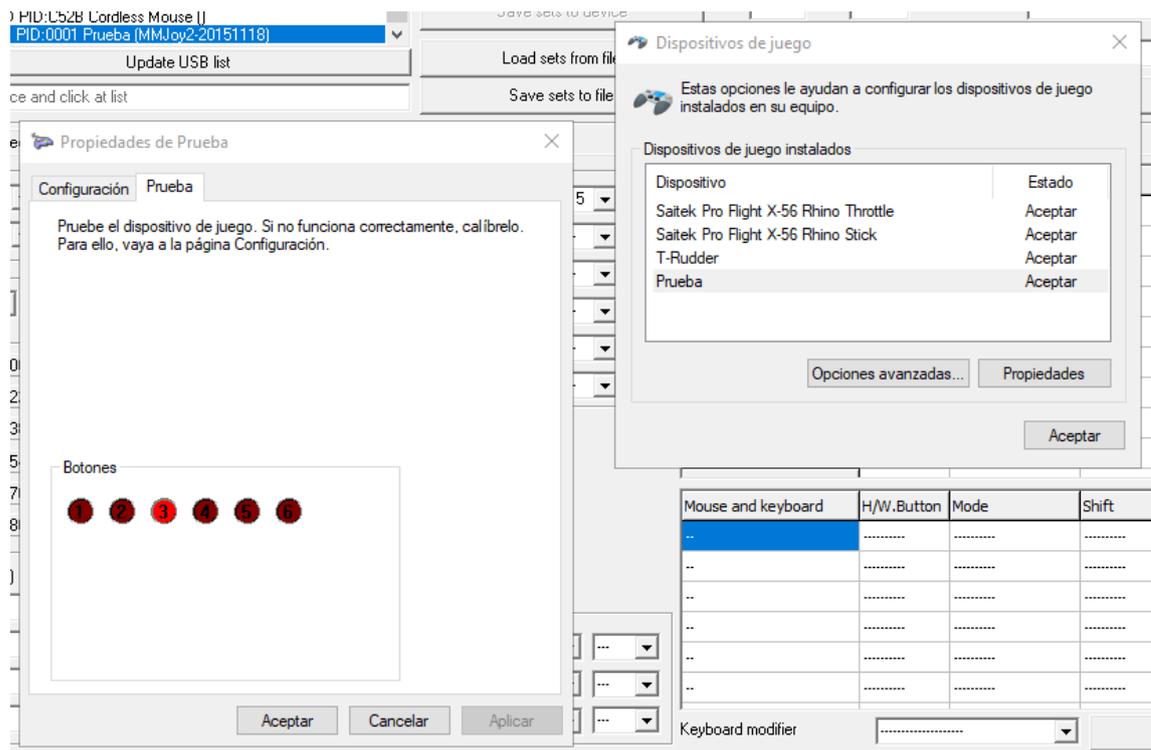
La última de nombre "**Timer OFF**" no tengo muy claro que hace todavía, pero no es necesaria para lo que estamos haciendo, además de que en la versión 20161101 del software MMJoy2 ya no aparece.

Notar que no es necesario que, en la configuración de la lista de 64 botones, tenga por qué coincidir su número con el de la casilla de la matriz numerada, en ésta se muestran las entradas de la tarjeta de expansión mientras que en el listado de "Buttons" se configura que entrada es la que activa qué botón y lo que hace en concreto.

Una vez hechas estas configuraciones, habrá que cargarla en nuestra placa, picaremos en "Save sets to device", se desconectará la placa una vez que se haya cargado el firmware y desaparecerá del listado de dispositivos USB.

Pasados unos segundos volverá a aparecer, en ese instante, si abrimos la aplicación de Windows denominada "Dispositivos de juego" picando en la opción "Windows Joysticks" en la parte inferior izquierda de la pantalla del MMJoy2, se abrirá la misma.

En la aplicación de Windows vamos a poder ver aquellos dispositivos de juegos que tenemos conectados, entre ellos nuestra placa llamada "Prueba".



Si lo seleccionamos y hacemos doble clic sobre el mismo, se abrirá otra ventana denominada “Propiedades de Prueba” en la que van a aparecer los botones (inputs) que le hayamos configurado en la sección “Buttons”.

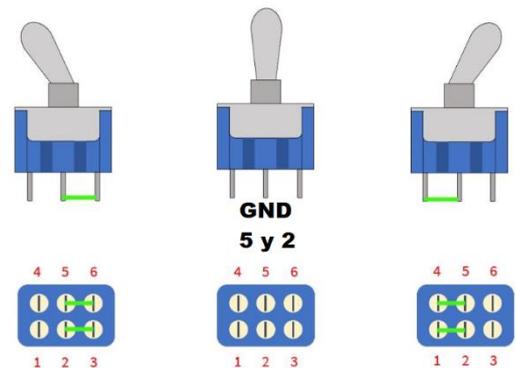
En el ejemplo que nos ocupa son:

- Button 1 = 3
- Button 2 = 6
- Button 3 = 4
- Button 4 = 7
- Button 5 = 14 Encoder
- Button 6 = 15 Encoder

Los switches o botones funcionan básicamente de la misma manera, aunque son dispositivos diferentes, los dos dan como resultado una señal de salida.

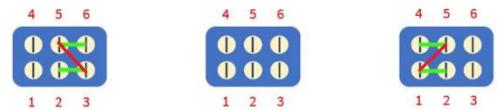
Uno de los componentes que más vamos a utilizar en la creación de un dispositivo de juego es el switch en cualquiera de sus variantes (on-on, on,off, on-on-on, on-off-on, mon-on-mon, etc...), y hay uno en concreto con el que hay que tener una precaución, se trata del switch de tres posiciones y dos polos.

El switch dispone de dos filas de tres pines cada fila, entre estas dos filas no hay conexión, el pin del medio de las dos filas corresponde al GND.



En la posición intermedia, donde el mástil del switch está recto, se activan las casillas centrales 5 y 2, si puenteamos estas dos entre sí, lo que vamos a conseguir es interconectar pines de filas diferentes, por ejemplo:

- Izquierda = 5-2-6-3
- Derecha = 5-2-4-1
- Central = 5-2



Si queremos utilizar los tres contactos del switch de tres posiciones sin que se conecten entre sí, es necesario cablear las dos filas de que dispone ese switch a dos conectores diferentes, así se podrán utilizar las tres posiciones del mismo de forma independiente y correcta.

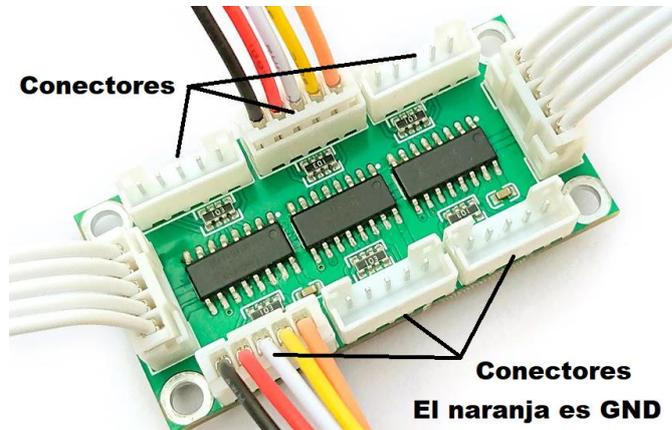
Se supone que queremos usar las tres posiciones del switch, para ello es imprescindible cablear los seis pines de este componente, con la peculiaridad de que, el pin del centro nos va a dar dos señales.

He probado con diferentes conexiones y tan solo funciona correctamente si tenemos cableados todos los pines del switch de tres posiciones, en las imágenes siguientes se puede ver la forma en la que Windows reconoce esas pulsaciones.



Para poder asignar en el simulador las tres entradas y que funcionen como queremos, eso es utilizando la de los dos extremos más la central, es imprescindible colocar las dos señales que salen de los pines centrales del switch (la 2 y la 3) a la misma casilla de nuestro simulador.

Si no lo hacemos de esa forma, cuando cambiemos de posición la palanca del interruptor, habrá una ocasión en la que el switch no vuelva al centro, supongo que es debido a la forma en la que está construido este componente.



Se puede cablear una fila a uno de los conectores y la otra a otro diferente, sin olvidarnos nunca de conectar correctamente el GND.

- Potenciómetros.

Una vez hecho el bootloader en nuestra placa de forma correcta, hay que proceder a configurar el software MMJoy2 para que sepa lo que tiene que leer y como ha de hacerlo, para que pueda reconocer todos aquellos componentes que tenemos cableados a la placa.

Para calibrar potenciómetros hay que acudir a la opción "Joysticks axes" y configurarlo de la siguiente forma:

- **Source.**

Aquí pondremos la opción "IntSensor", esto hace referencia a que se va a tratar la señal desde el sensor interno de la placa Arduino.

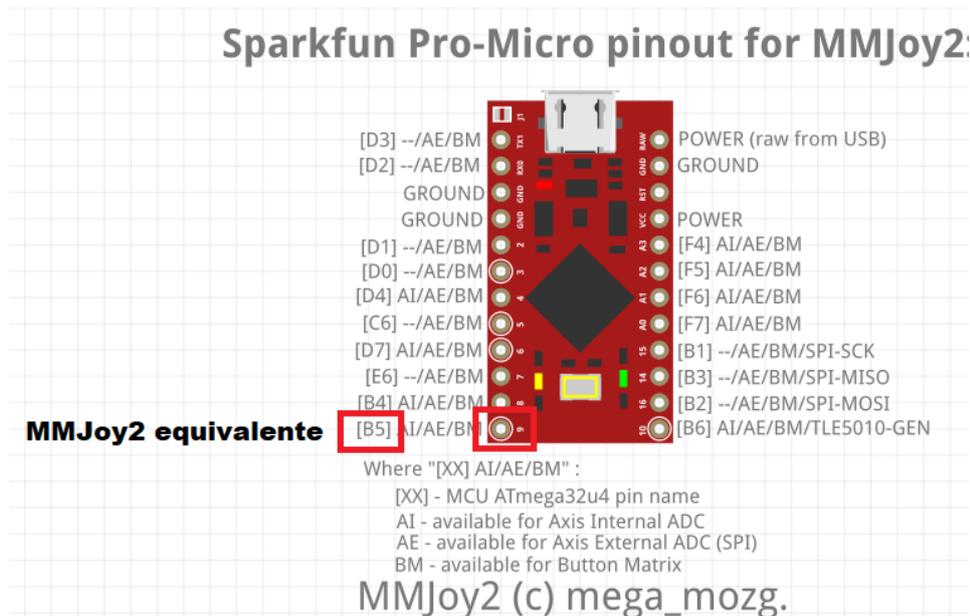
En el desplegable se da la posibilidad de seleccionar diferentes opciones que hacen referencia a otros componentes que se pueden utilizar para tratar este tipo de señales.

- **MCU Port.**

El número del pin donde está conectado el de datos del potenciómetro que corresponde al pin central del mismo.

Hay que tener en cuenta la equivalencia que hace MMJoy2 respecto a la placa Pro Micro, si se conecta el cable del potenciómetro del pin central a la entrada analógica número 9

de la placa Pro-Micro, deberemos poner en esta casilla el valor "B5".



- **value row.**
Aquí es muy importante girar el potenciómetro hasta que alcance su valor máximo (1023), más adelante explicare el motivo.
- **value processed.**
Aquí aparecerá el mismo valor que en el apartado anterior
- **Asignment.**
Colocaremos la opción "Slider".
En caso de que queramos que nuestro potenciómetro represente a un eje "X", "Y" o "Z", así lo tendremos que seleccionar.
- **Precision(bit).**
Se puede dejar por defecto en "8", yo he colocado el valor "10".
La precisión está relacionada con la sensibilidad a la hora de leer el potenciómetro.
- **Auto-calibration.**
Este es un punto importante pues, dependiendo de la opción que pongamos aquí, Windows va a reconocer los valores y posición del potenciómetro de formas diferentes.
A continuación, se explican como altera el comportamiento del potenciómetro según pongamos una u otra:
a) auto.w center.

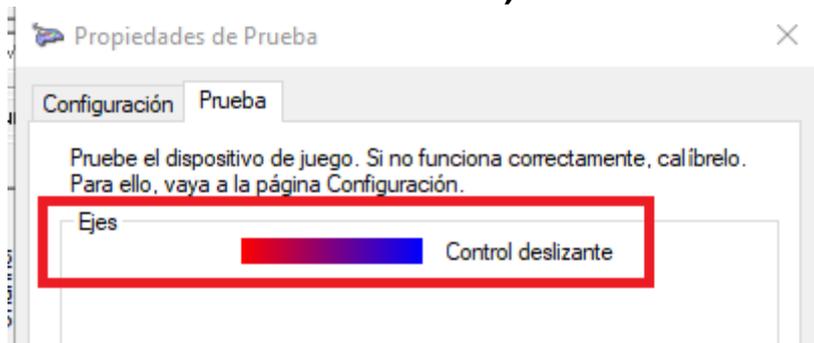
Si ponemos esta opción, en el momento en que conectemos nuestro dispositivo de juego, el valor inicial del potenciómetro va a comenzar justo en el centro, el valor de la izquierda será -50%, el del centro será 0% y el de la derecha +50%.

Al girarlo va a ir desde el centro hasta la derecha para aparecer por la izquierda hasta volver al centro.

Este comportamiento nos puede interesar si queremos asignarlo al stick o joy donde la posición de reposo es la central, otro ejemplo sería asignarlo al pitch, aw o roll donde el centro es el valor inicial.

Imprescindible calibrar el valor máximo y mínimo.

b) auto.w/o center.



En esta ocasión, el valor inicial del potenciómetro nada más conectarlo será igual a "0", si está en su extremo izquierdo. En caso de que al conectarlo se encuentre en un valor intermedio, al moverlo, se sincronizará con el que le

corresponda, la cuestión es que va a transitar entre el 0% como valor mínimo y el 100% como valor máximo.

Optimo por ejemplo para simular el throttle o para controlar la intensidad de la luz de un panel.

Imprescindible calibrar el valor máximo y mínimo.

c) saved.w center.

Similar al "auto.w center" pero sin necesidad de asignarle valor mínimo ni máximo.

d) saved.w/o center.

Similar al "auto.w/o center" pero sin necesidad de asignarle valr mínimo ni máximo.

El resto de casillas no es necesario asignarlas.

En la versión que yo utilizo de MMJoy2 (2015118) en la ventana relacionada con la configuración de los potenciómetros, hay un botón con

nombre “Calibrate helper” con el que podemos decirle al software cuales son los valores máximos y mínimos del potenciómetro.

| axis name | current | minimum | center | maximum |
|-----------|---------|---------|--------|---------|
| Slider | 1023 | 1023 | - | 1023 |
| -- | 0 | - | - | - |
| -- | 0 | - | - | - |
| -- | 0 | - | - | - |
| -- | 0 | - | - | - |
| -- | 0 | - | - | - |
| -- | 0 | - | - | - |
| -- | 0 | - | - | - |
| -- | 0 | - | - | - |
| -- | 0 | - | - | - |

Clear data Set center

Save calibration

En un primer momento el “Slider” tomará los valores iniciales de 1023, si procedemos a girar el mismo de un lado al otro, veremos cómo los valores asociados a “current” y “minimum” cambian tomando, al final del proceso, los valores adecuados.

Estos valores corresponden con:

- A la izquierda totalmente = 0.
- A la derecha totalmente = 1023.

Para que todos aquellos cambios que hagamos en la placa Pro Micro, a través del software MMJoy2, surtan efecto y funcionen correctamente como interface de juego, es necesario comprobar y calibrar el dispositivo con la aplicación nativa de Windows 10.

Para tal efecto, buscaremos "Dispositivos de juego", en la ventana que se abrirá aparecerán todos los dispositivos de juego instalados en el sistema, entre ellos el que estamos programando nosotros con MMJoy2

Para que no entren en conflicto unos dispositivos con otros, después de haber realizado el bootloader a la placa, hay que darle un nombre diferente y unos números VID y PID distintos, como se explicó al comienzo de este manual.

Algo importante a la hora de conectar potenciómetros a la placa Arduino es que tenemos que hacerlo directamente sobre la misma, no podemos conectar los pines del potenciómetro a través de las tarjetas de expansión.

A tener en cuenta estos dispositivos, tenemos que tener la precaución de usar las conexiones analógicas de la placa que estemos utilizando en cada caso, por ejemplo, en la placa de la derecha se utilizarán los pines marcados con una "A" (Analog inputs).

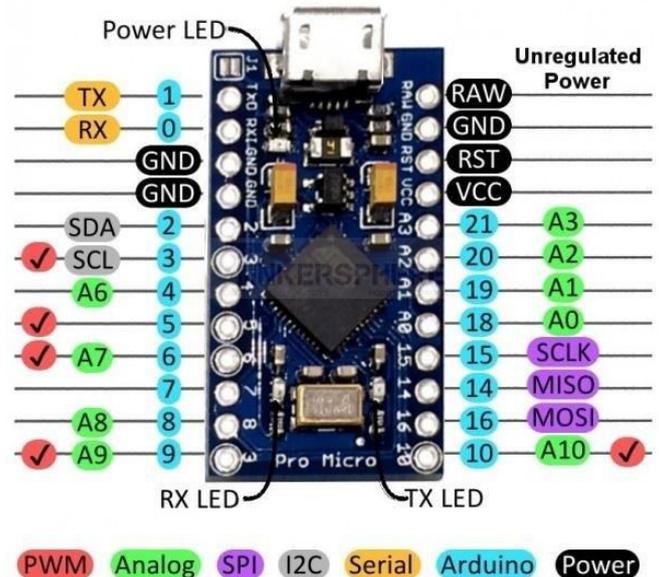
Una vez hecho esto y haber configurado todas las entradas que necesitemos, ya sean ejes, encoders, botones, etc.... tenemos que cargar la configuración en la placa, esto se hace pulsando el botón "Save sets to device".

También se puede guardar una copia de la programación que hemos llevado a cabo a través del botón "Save sets to file", de esa forma podremos volcarla en otra placa con "Load setc from file".

Si queremos modificar alguna programación de una de nuestras placas por el motivo que sea, la podemos conectar a MMJoy2 y volcar el setup de la placa al software con "Load sets from device", así aparecerán todos los parámetros y podremos modificar lo que nos convenga.

Luego solo resta volver a volcar el perfil modificado de nuevo a la tarjeta del modo que ya conocemos.

Es importante tener presente que el límite de ejes que va a soportar el setup que estemos creando con MMJoy2, será de 8, con lo que no podremos conectar más de 8 potenciómetros por placa.

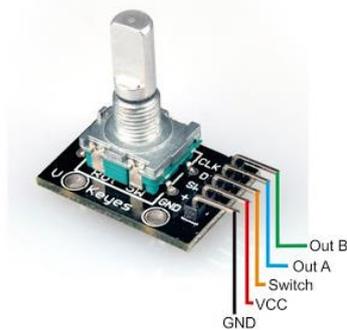


- Encoders.

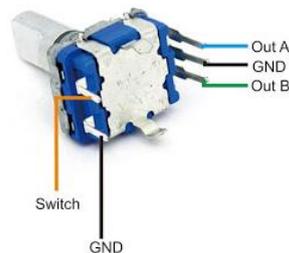
Los Encoders son dispositivos que diferencian el sentido de giro de su eje, no tienen un valor mínimo ni máximo como los potenciómetros y se pueden girar a un lado u otro de forma indefinida.

También son capaces de medir la velocidad angular a la que gira pues genera pulsos a intervalos regulares, a cada paso se genera una señal que puede ser utilizada electrónicamente.

With Breakout Board



Without Breakout



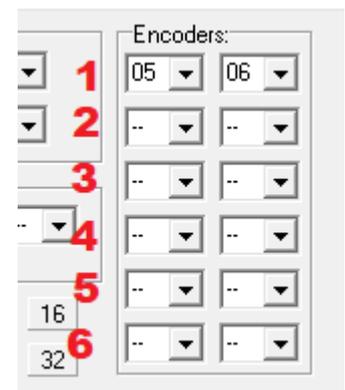
Se pueden encontrar soldados sobre una PCB o bien sueltos, la forma de conectarlos es la de la imagen de la izquierda.

La señal procedente de este tipo de dispositivo la vamos a tratar como si de un botón

normal se tratase, es una señal eléctrica que manda cada vez que se acciona, si bien distingue los cambios de estado y la dirección en la que gira.

El MMJoy2 dispone de una parte destinada a los encoders, es posible conectar hasta 6 de ellos a la misma placa,

Los dispositivos encoder se pueden cablear a través de las shift register, teniendo en cuenta sus pines (GND, VCC, DataA, DataB, Switch), los hay que disponen de un botón pulsador integrado también.



Una vez conectado el mismo a la placa y esta al programa MMJoy2 a través del puerto USB, procederemos seleccionando la placa como ya sabemos, en el recuadro superior izquierdo donde está el listado de unidades USB.

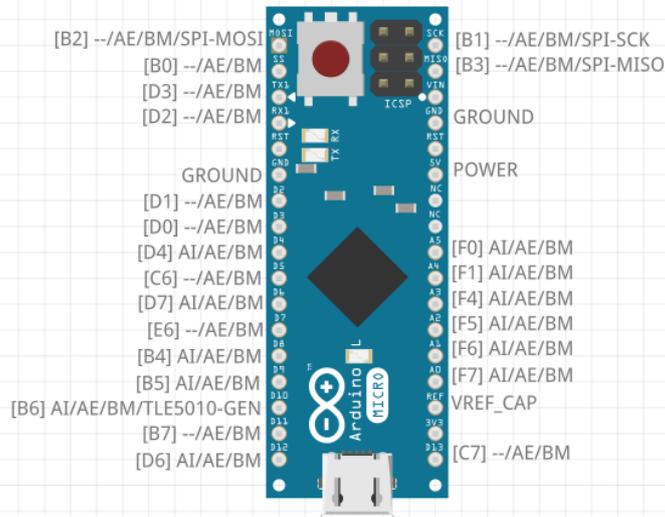
La seleccionamos y damos doble clic para conectarla.

El sentido de giro nos da igual pues, a la hora de aplicar la salida de los botones "Button 7" y "Button 8" a nuestro simulador, vamos a seleccionar el que nos venga bien.

Los botones 7 y 8 de nuestro dispositivo de juegos, serán reconocidos por Windows como botones corrientes, lo mismo que en perfil del juego, la peculiaridad de utilizar estos dispositivos se basa en su capacidad de generar una señal en cada cambio de estado y sentido, con la sensibilidad determinada en el perfil de MMJoy2.

ANEXO I – Placas compatibles.

Arduino Micro pinout for MMJoy2:



Where "[XX] AI/AE/BM" :

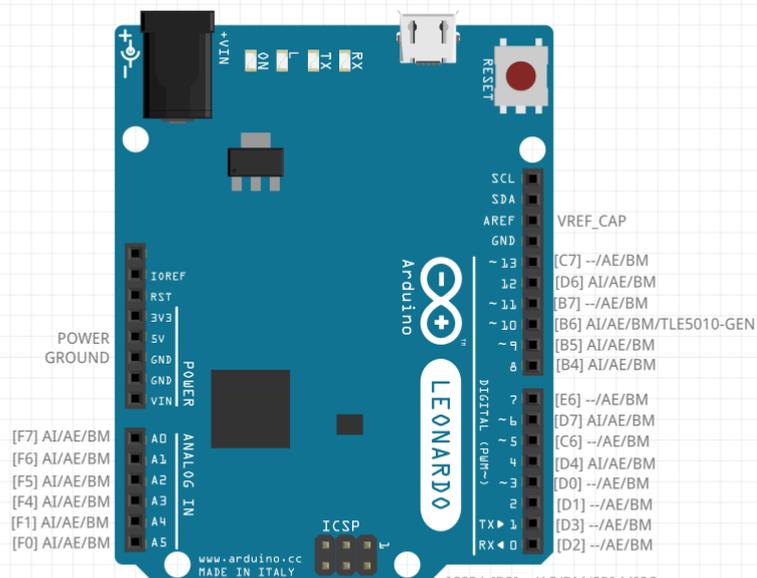
- [XX] - MCU ATmega32u4 pin name
- AI - available for Axis Internal ADC
- AE - available for Axis External ADC (SPI)
- BM - available for Button Matrix

MMJoy2 (c) mega_mozg.

FREE. PERSONAL DIY ONLY. NOT FOR COMMERCIAL.

fritzing

Arduino Leonardo pinout for MMJoy2:



Where "[XX] AI/AE/BM" :

- [XX] - MCU ATmega32u4 pin name
- AI - available for Axis Internal ADC
- AE - available for Axis External ADC (SPI)
- BM - available for Button Matrix

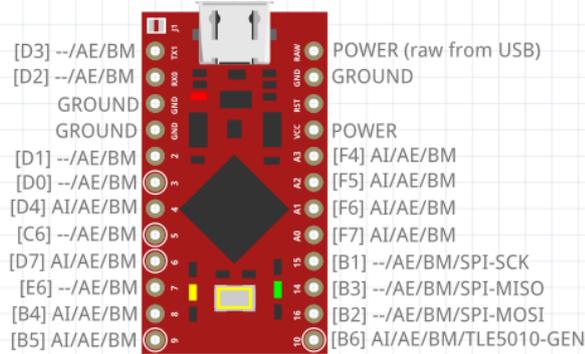
- ICSP1 [B3] --/AE/BM/SPI-MISO
- ICSP4 [B2] --/AE/BM/SPI-MOSI
- ICSP3 [B1] --/AE/BM/SPI-SCK

MMJoy2 (c) mega_mozg.

FREE. PERSONAL DIY ONLY. NOT FOR COMMERCIAL.

fritzing

Sparkfun Pro-Micro pinout for MMJoy2:



Where "[XX] AI/AE/BM" :

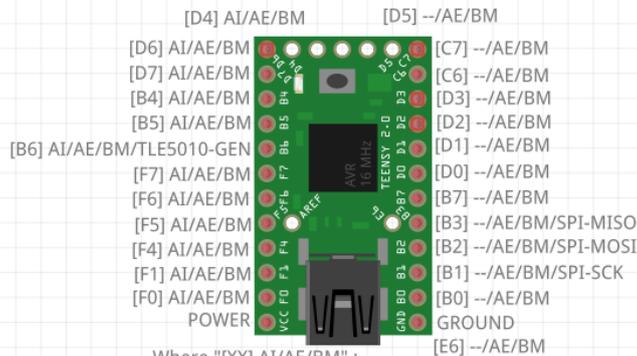
- [XX] - MCU ATmega32u4 pin name
- AI - available for Axis Internal ADC
- AE - available for Axis External ADC (SPI)
- BM - available for Button Matrix

MMJoy2 (c) mega_mozg.

FREE. PERSONAL DIY ONLY. NOT FOR COMMERCIAL.

fritzing

PJRC Teensy 2.0 pinout for MMJoy2:



Where "[XX] AI/AE/BM" :

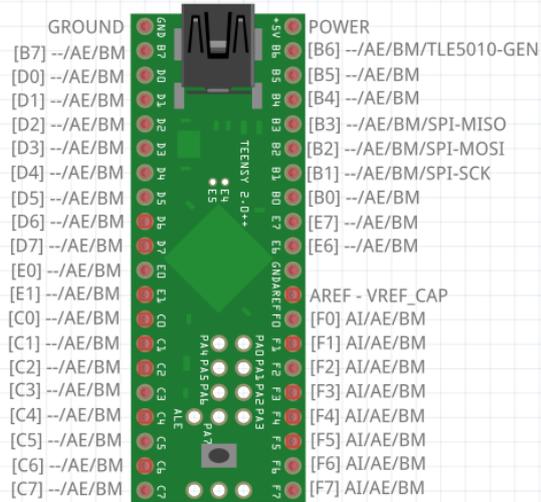
- [XX] - MCU ATmega32u4 pin name
- AI - available for Axis Internal ADC
- AE - available for Axis External ADC (SPI)
- BM - available for Button Matrix

MMJoy2 (c) mega_mozg.

FREE. PERSONAL DIY ONLY. NOT FOR COMMERCIAL.

fritzing

PJRC Teensy ++ 2.0 pinout for MMJoy2:



Where "[XX] AI/AE/BM" :

[XX] - MCU AT90USB1286 pin name

AI - available for Axis Internal ADC

AE - available for Axis External ADC (SPI)

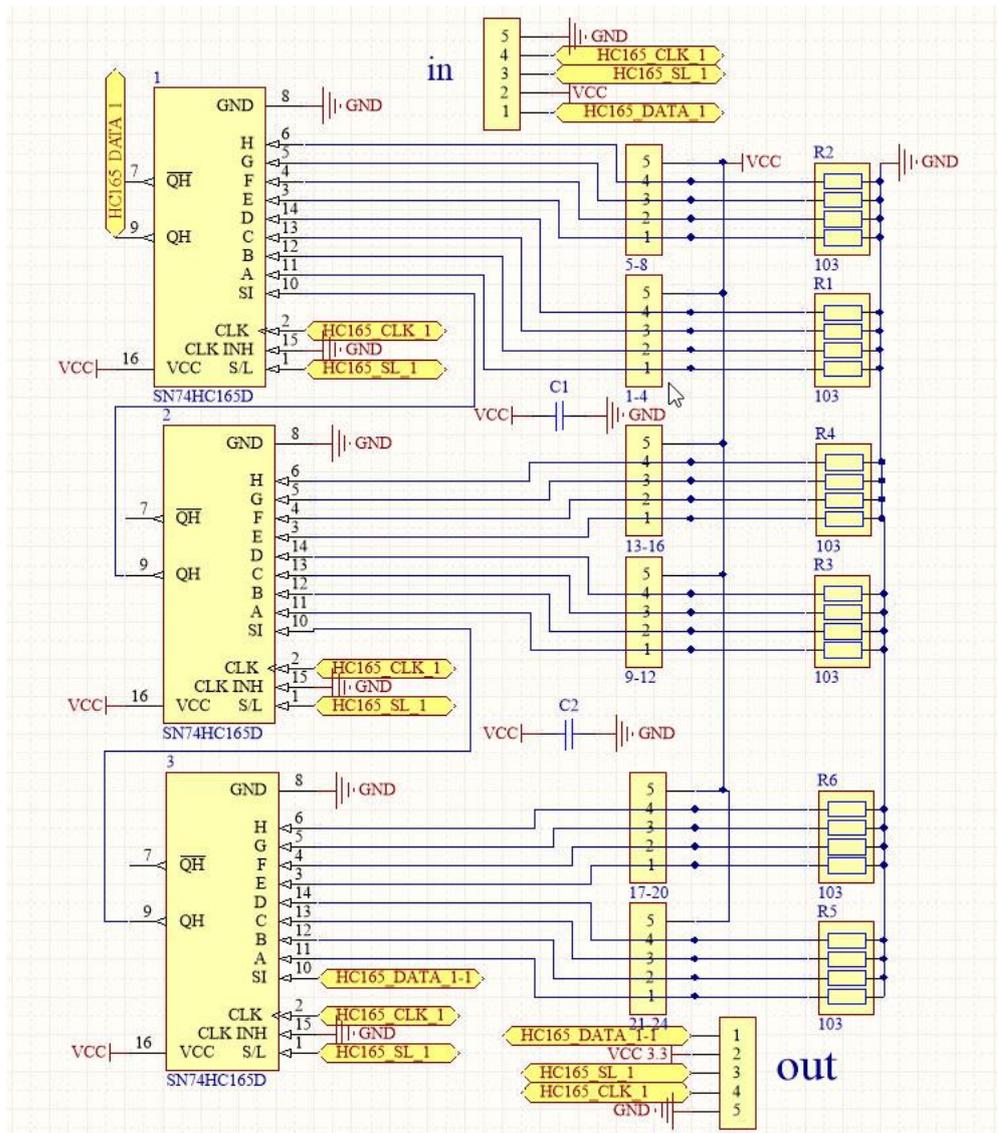
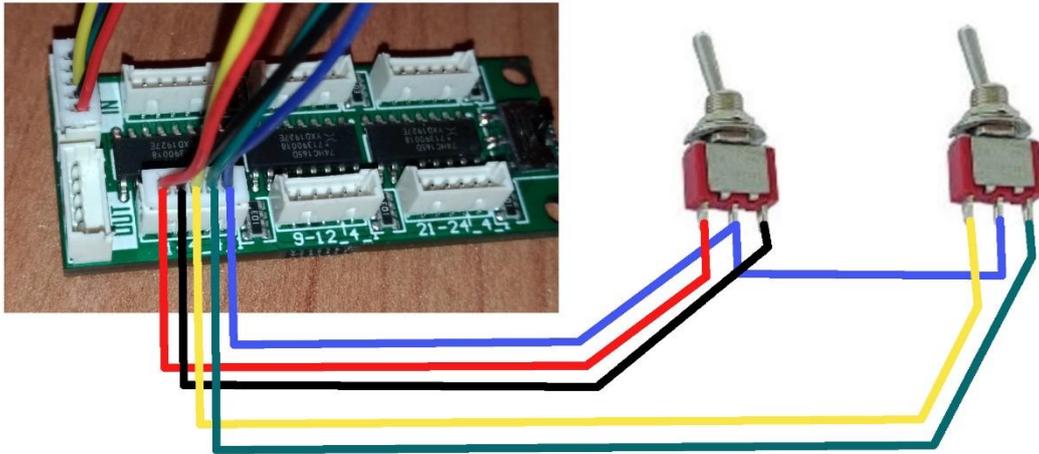
BM - available for Button Matrix

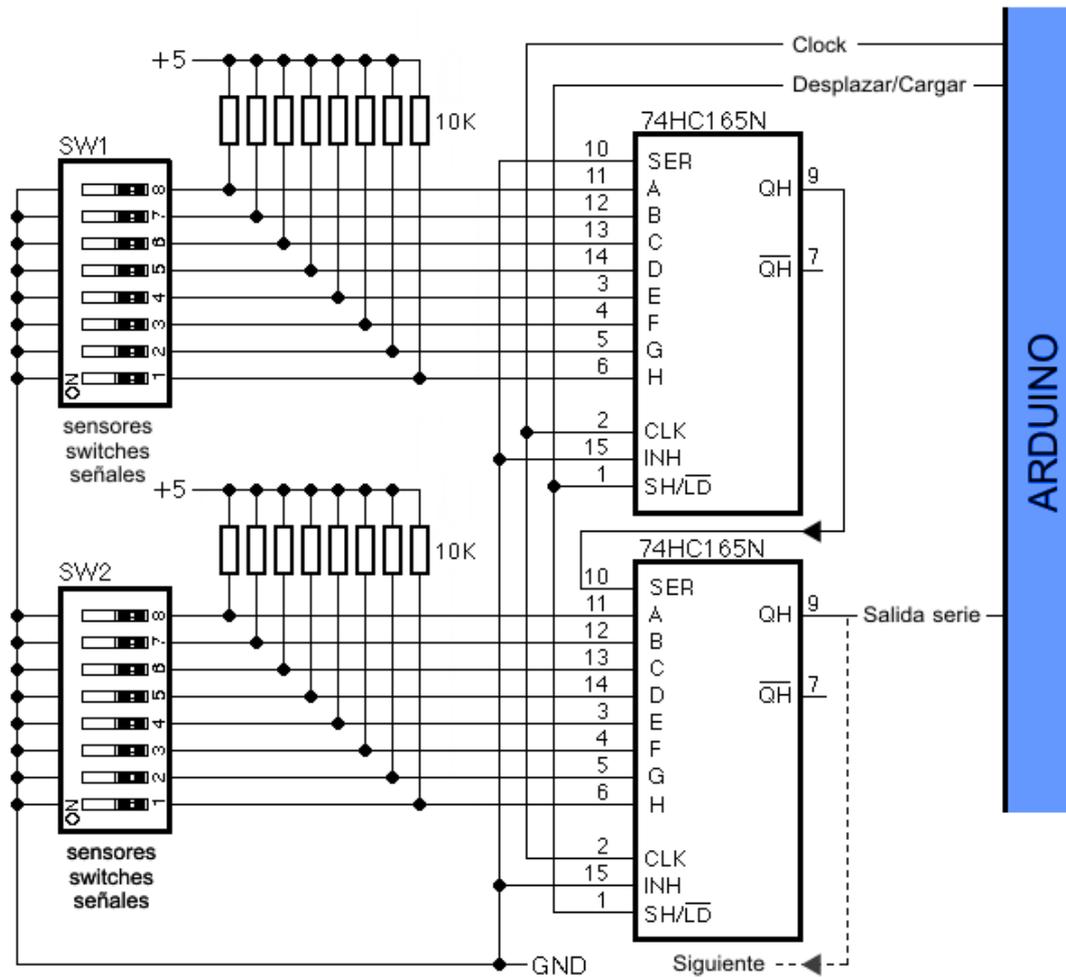
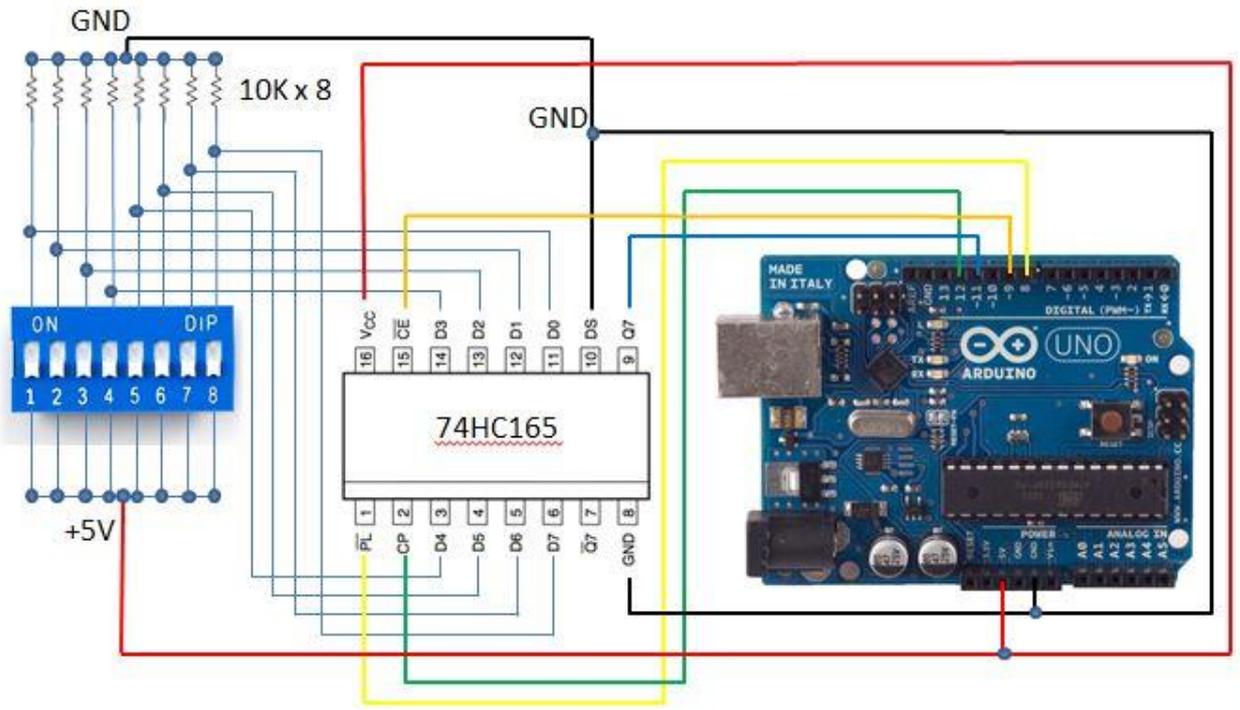
MMJoy2 (c) mega_mozg.

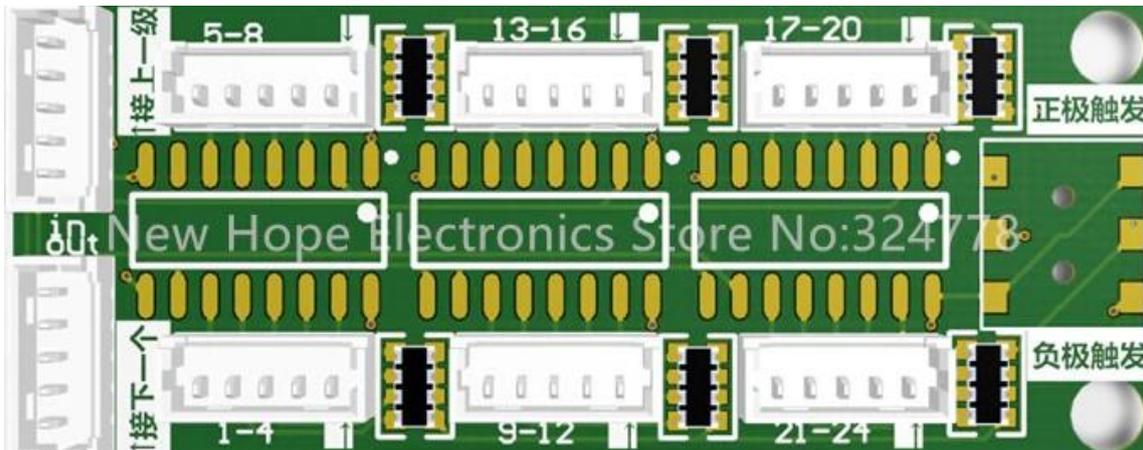
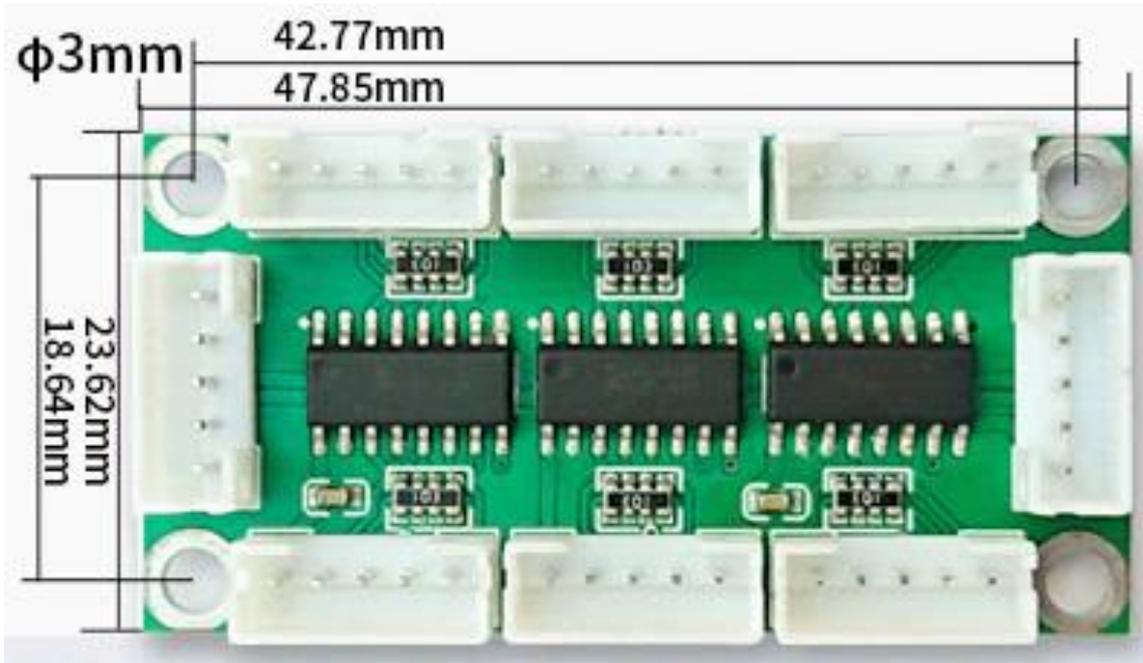
FREE. PERSONAL DIY ONLY. NOT FOR COMMERCIAL.

fritzing

ANEXO II – Conexiones y esquemas.





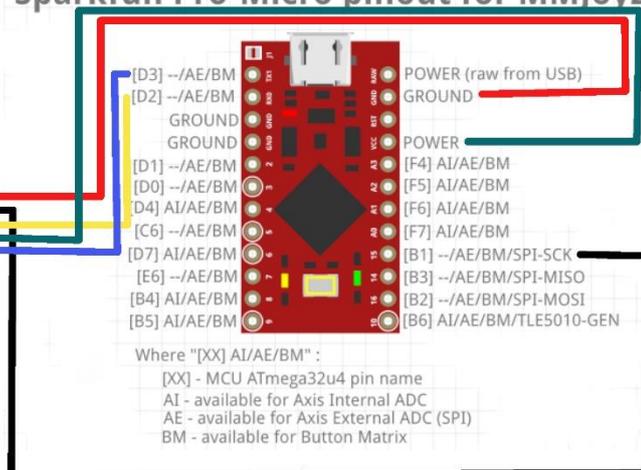


74HC165D Expansion Board Sparkfun Pro-Micro pinout for MMjoy2:



Conexiones:

- GND --- GROUND
- SCK --- B1
- CS --- (seleccionable) D2
- 5v --- POWER
- MISO--- (seleccionable) D3



Configuracion en mmjoy2 (version 20161101)

En pestaña "shift register"

Chip --> 74HC165 (el de esta placa otras podrian tener CD4021)
 "3" ---> numero de chips en la placa
 SR-CS--> Pin elegido para CS (D2 en este caso)
 SR-DATA--> Pin elegido para MISO (D3 en este caso)

