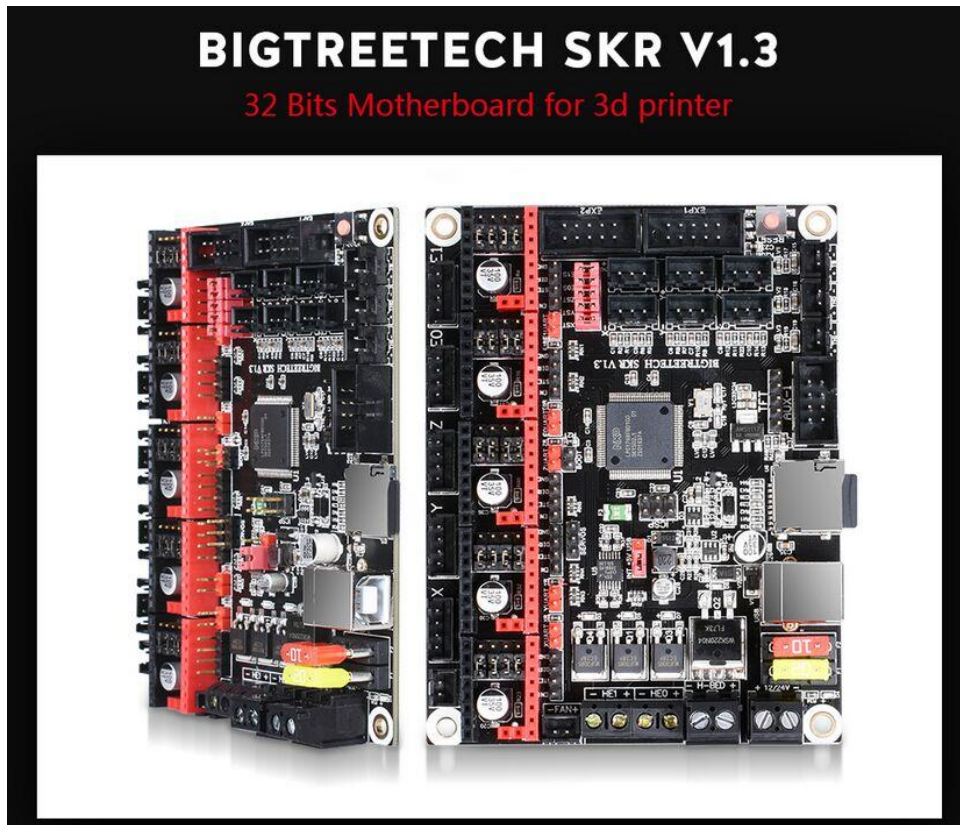


# Guía BIGTREE TECH/BIQU SKR V1.3

Canal telegram [https://t.me/SKR\\_board\\_32bits](https://t.me/SKR_board_32bits) (Versión 06-09-2019. @LoKuS77)



Me ha llegado una bonita placa negra SKR V1.3 de aliexpress ¿Y ahora qué? ☺

Toda la info oficial como esquemas, cableados y hardware sobre esta placa de 32 bits se encuentra en el

**Sitio oficial GITHUB de BIGTREE TECH.** <https://github.com/bigtreetech/BIGTREE TECH-SKR-V1.3>

**Esta guía muestra como instalar el firmware Marlin 2.0 firmware en la placa SKR V1.3 en general.  
No tiene en cuenta la máquina u otros ajustes específicos de tu hardware!  
Este documento se base en S.O. Windows.**

Puedes usar VSCode o Atom.IO + Platform IO para compilar Marlin2.0 en la BIGTREE TECH SKR V1.3

Las instrucciones dan por hecho que vas a usar una pantalla LCD de tipo 128x64 estándar REPRAP\_DISCOUNT\_FULL\_GRAPHIC\_SMART\_CONTROLLER con dos cables cinta de 10 pins. →



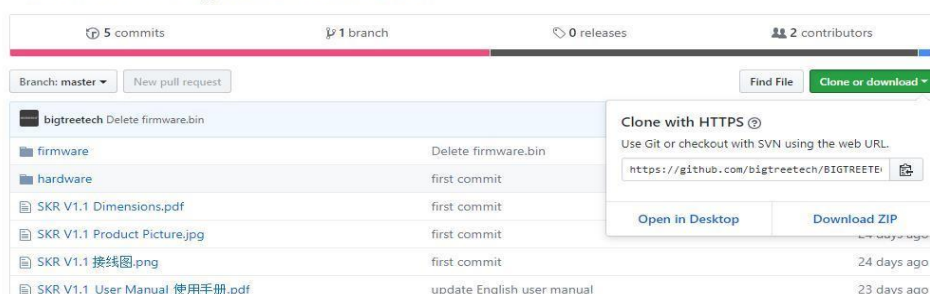
Descarga el firmware Marlin2.0 de su página github oficial: [Github de firmware Marlin 2.0](https://github.com/bigtreetech/BIGTREE TECH-SKR-V1.3) y haz click en

Clone or download

Click en "Download ZIP"

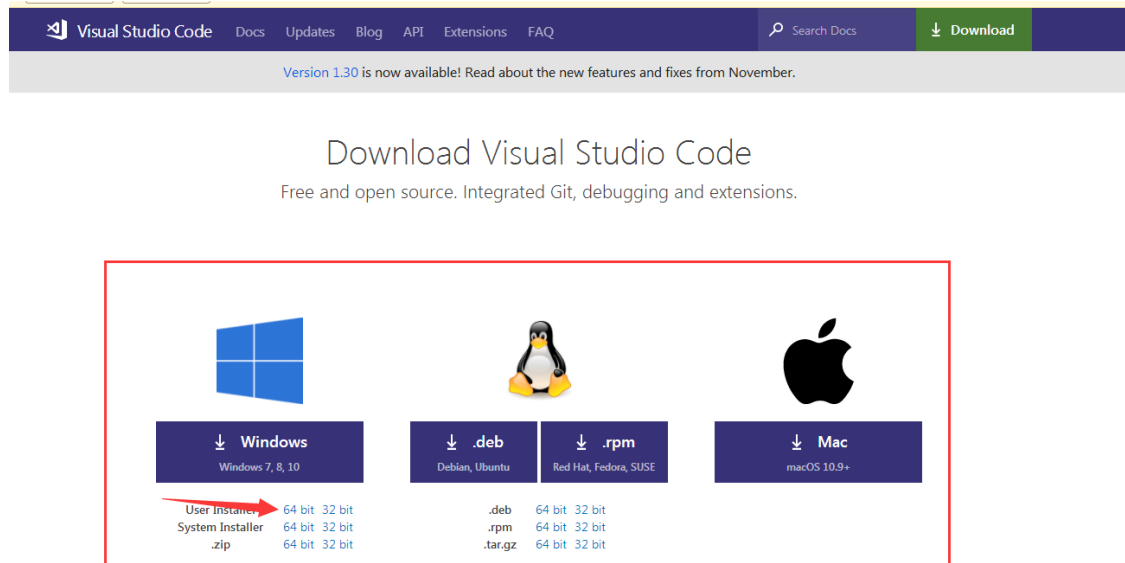
En general la copia del firmware Marlin 2.0 que BIGTREE TECH tiene en su Github suele estar obsoleta, así que como norma general NO LA USES. Utiliza [Marlin 2.0 oficial](https://github.com/MarlinFirmware), y modifícala según tus necesidades.

32bit board with LPC1768, support marlin2.0 and smoothieware

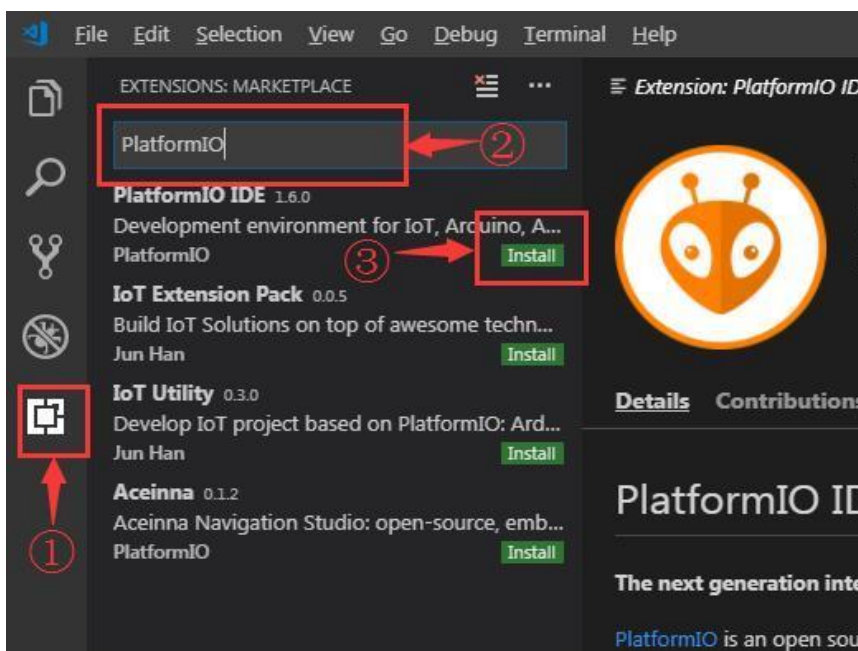


Cuando la descarga esté completa, descomprímela en una ruta conocida. Aconsejable una **ruta corta**, tipo c:\mi marlin\  
Muchas veces Marlin da errores de compilación en por utilizar **rutas en disco muy largas**, tipo al escritorio  
(que realmente puede ser “c:\users\manolito perez gomez\desktop\cosasquemebajo\marlin\el de ayer\marlin2.0-bugfix\” )


En este ejemplo, usamos *Visual Studio Code*. Descarga VScode de <https://code.visualstudio.com/Download>  
Elige la versión que coincida con tu sistema operativo.



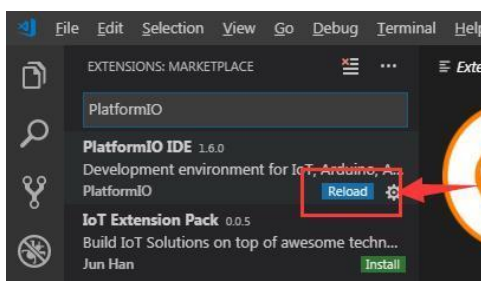
Después de terminar la descarga, haz doble click para realizar la instalación. Una vez instalado, abre VScode.



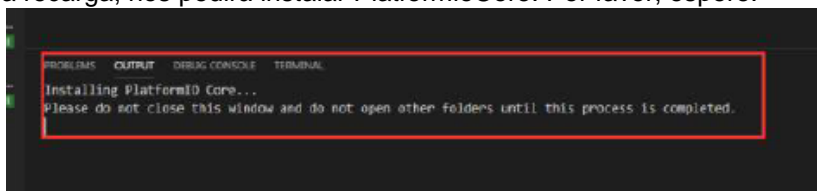
Dentro de VScode, necesitaremos también instalar el plugin **PlatformIO**, siguiendo los pasos:

- Click en la figura  del paso 1
- Escribimos **PlatformIO** en el recuadro del paso 2.
- Hacemos click en Install en PlatformIO IDE 1.xx , del paso 3.

Después de completar la descarga, necesitaremos hacer una recarga (Reload)

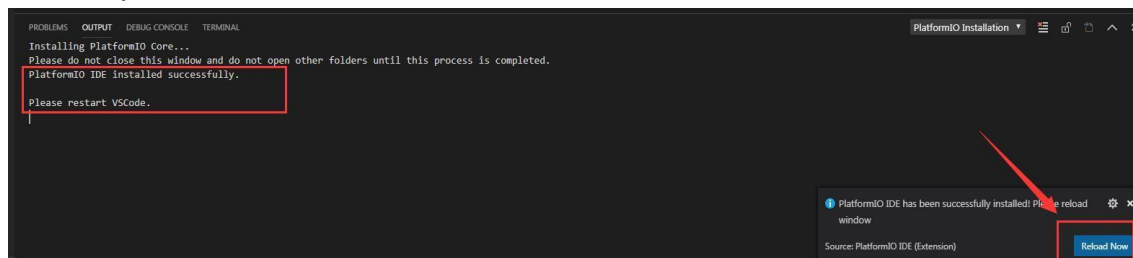


Tras la recarga, nos pedirá instalar PlatformIOCore. Por favor, espere.



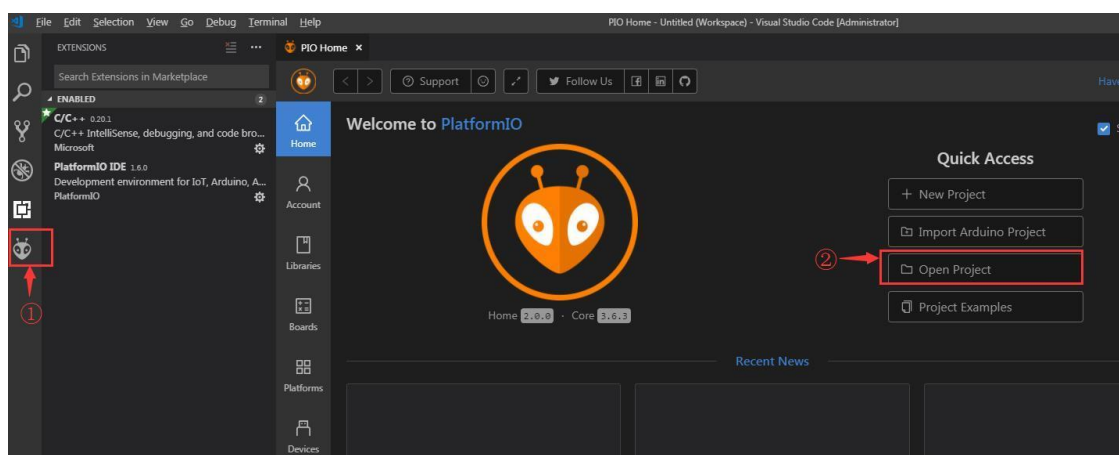
Cuando la instalación se complete, nos hará falta hacer de nuevo un reload.

Tras ello, ya tendremos PlatformIO instalado.

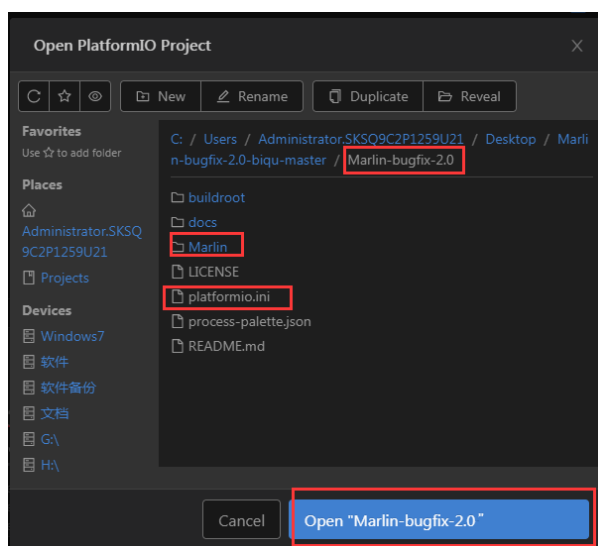


En la esquina inferior izquierda de VSCode, veremos un icono hormiguita que es el del plugin PlatformIO.

Haremos click en él (1), y después en Open Project (2), para abrir el proyecto.



Busca en tu disco duro la carpeta donde habías descomprimido marlin2.0 descargado previamente, eligiendo la carpeta que contenga el fichero “platformio.ini” (habitualmente “Marlin-bugfix-2.0”), y haz click en OPEN

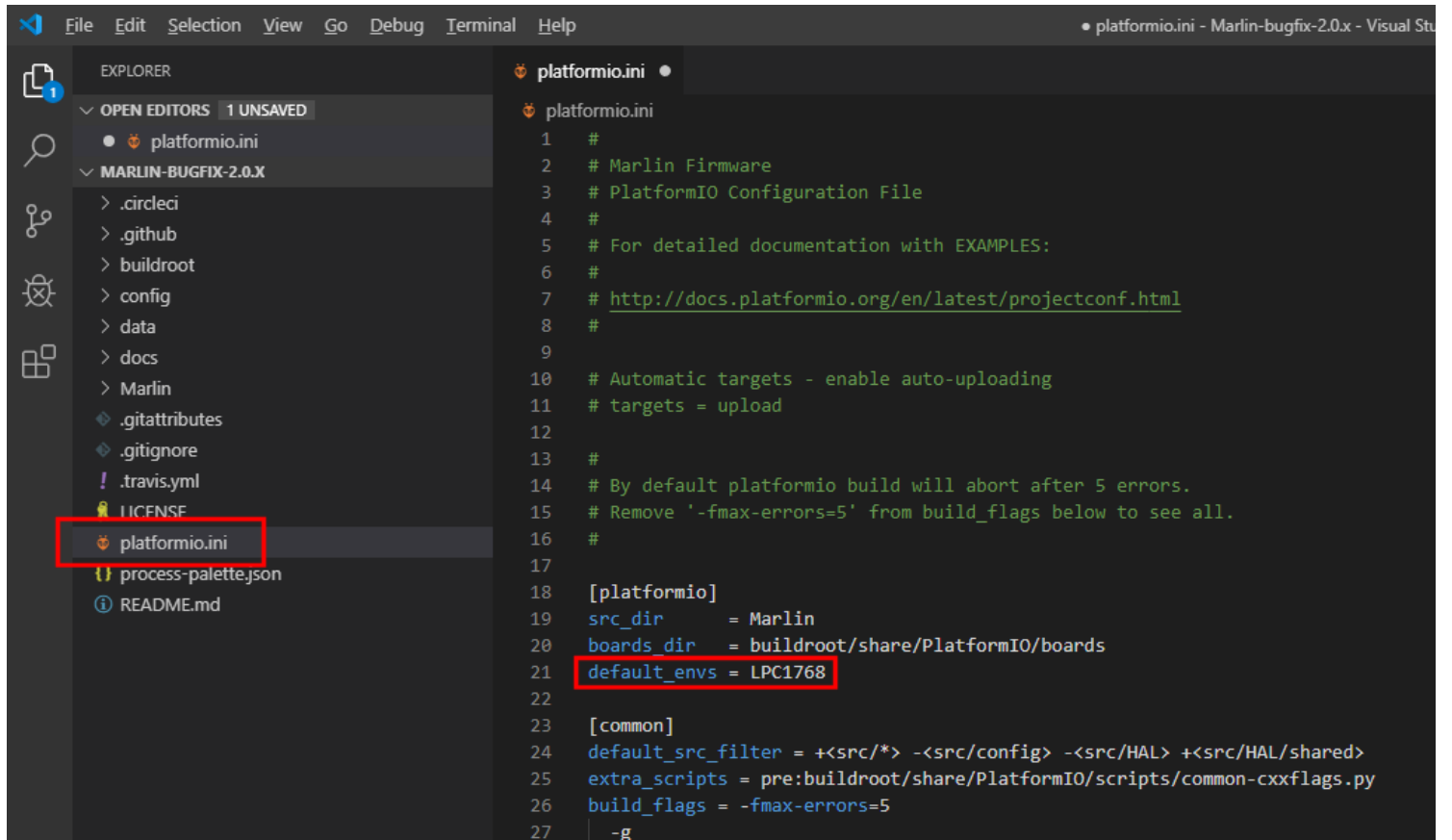


Después de abrir el proyecto, iremos al fichero **platformio.ini** y file y cambiaremos el entorno por defecto (default environment ) de megaatmega2560 a LPC1768, editando la línea

default\_envs = **LPC1768**.

NOTA 1: En versiones anteriores de marlín 2.0, hasta Julio2019, esta variable se llamaba **env\_default** en lugar de **default\_envs**

NOTA 2: Si tienes una SKR Mini / SKR Mini E3 / SKR Mini E3 DIP / SKR PRO, el valor no es **LPC1768**, míralo en el [Apéndice N°1](#)

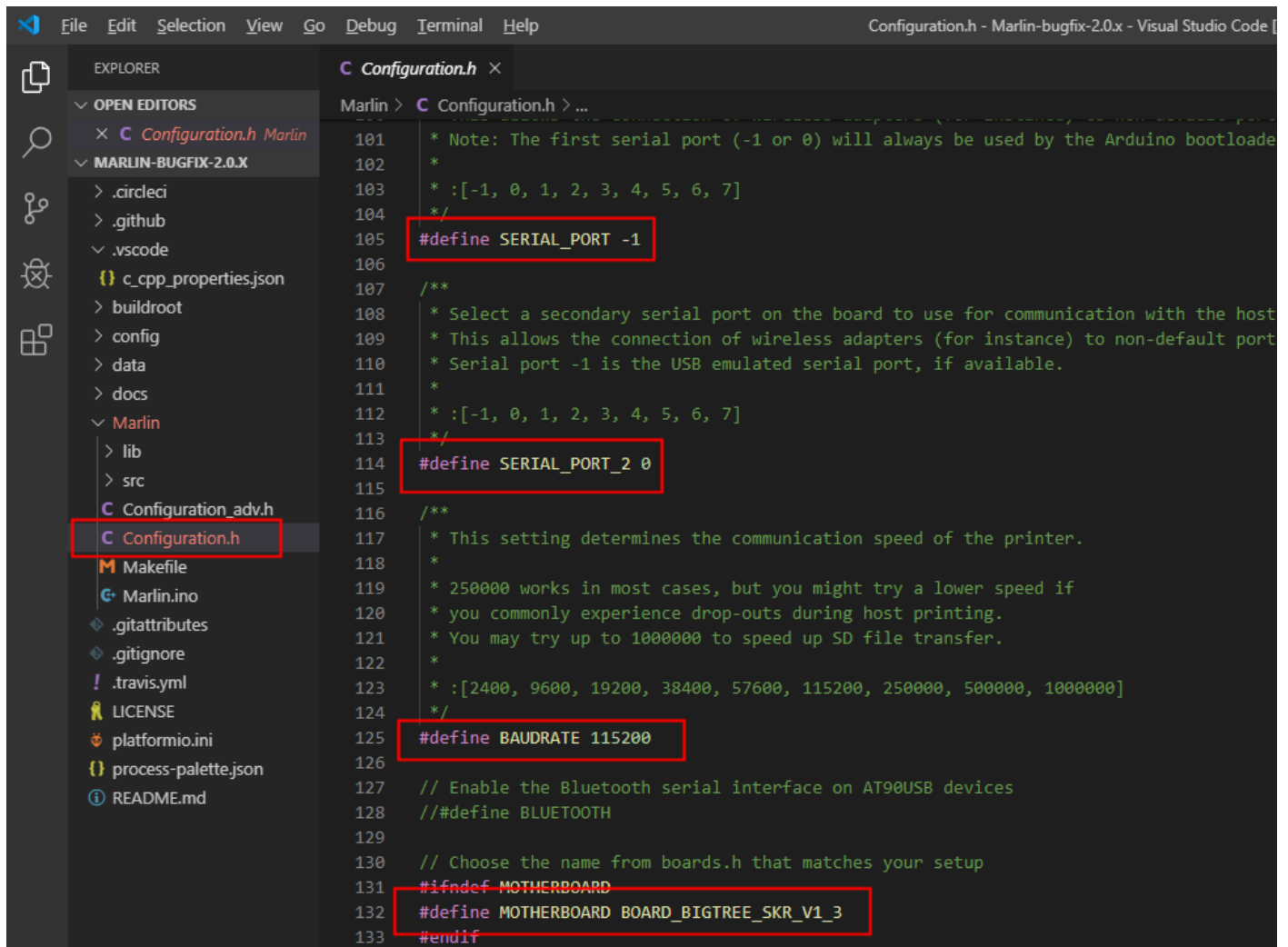


```
platformio.ini
1 #
2 # Marlin Firmware
3 # PlatformIO Configuration File
4 #
5 # For detailed documentation with EXAMPLES:
6 #
7 # http://docs.platformio.org/en/latest/projectconf.html
8 #
9
10 # Automatic targets - enable auto-uploading
11 # targets = upload
12 #
13 #
14 # By default platformio build will abort after 5 errors.
15 # Remove '-fmax-errors=5' from build_flags below to see all.
16 #
17
18 [platformio]
19 src_dir = Marlin
20 boards_dir = buildroot/share/PlatformIO/boards
21 default_envs = LPC1768
22
23 [common]
24 default_src_filter = +<src/*> -<src/config> -<src/HAL> +<src/HAL/shared>
25 extra_scripts = pre:buildroot/share/PlatformIO/scripts/common-cxxflags.py
26 build_flags = -fmax-errors=5
27 -g
```

Guardaremos cambios en el fichero platformio.ini, y después iremos al fichero **configuration.h**

Si no están ya hechos los cambios, pondremos:

```
#define SERIAL_PORT -1
#define SERIAL_PORT_2 0
#define BAUDRATE 115200
#define MOTHERBOARD BOARD_BIGTREE_SKR_V1_3
```



```
Configuration.h
101 * Note: The first serial port (-1 or 0) will always be used by the Arduino bootload
102 *
103 * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]
104 */
105 #define SERIAL_PORT -1
106
107 /**
108 * Select a secondary serial port on the board to use for communication with the host
109 * This allows the connection of wireless adapters (for instance) to non-default port
110 * Serial port -1 is the USB emulated serial port, if available.
111 *
112 * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]
113 */
114 #define SERIAL_PORT_2 0
115
116 /**
117 * This setting determines the communication speed of the printer.
118 *
119 * 250000 works in most cases, but you might try a lower speed if
120 * you commonly experience drop-outs during host printing.
121 * You may try up to 1000000 to speed up SD file transfer.
122 *
123 * :[2400, 9600, 19200, 38400, 57600, 115200, 250000, 500000, 1000000]
124 */
125 #define BAUDRATE 115200
126
127 // Enable the Bluetooth serial interface on AT90USB devices
128 // #define BLUETOOTH
129
130 // Choose the name from boards.h that matches your setup
131 #ifndef MOTHERBOARD
132 #define MOTHERBOARD BOARD_BIGTREE_SKR_V1_3
133 #endif
```



Si usamos pantalla de tipo **LCD12864**:

En el fichero configuration.h, descomentaremos

```
#define REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER
```

(Si utilizamos otro tipo de pantalla, descomentaremos la que proceda).

Si el LCD no da imagen, revisa el apartado [Problemas LCD](#)



## Config de ventiladores.

**-Venti Capa:** Si quieres usar un ventilador de capa, conéctalo al conector CNC FAN. Marlin (y tu laminador al generar Gcode) regulan la potencia del ventilador por software. Más info: [Gcode M106](#)

**-Venti Hotend:** Si quieres usar el ventilador del hotend (o el de la electrónica) controlado de forma automática **ON/OFF al llegar a 50°C**, conéctalo a la salida HE1 (E1 heater) y modifica el fichero configuration\_adv.h file:

```
//#define E0_AUTO_FAN_PIN -1 cambiarlo a  
#define E0_AUTO_FAN_PIN FAN1_PIN
```

(Sería lo mismo que ponerlo como E0\_AUTO\_FAN\_PIN P2\_04, porque en el fichero de pins de SKR se asigna "#define FAN1\_PIN → P2\_04" si tenemos un solo extrusor)

**-Venti Electrónica/drivers:** Si quieres un ventilador para la placa o drivers, que se active cuando los drivers están activos iremos al fichero configuration\_adv.h y cambiaremos:

```
#define USE_CONTROLLER_FAN
```

```
#if ENABLED (USE_CONTROLLER_FAN)
```

```
#define CONTROLLER_FAN_PIN P1_26
```

// define un pin personalizado para el venti de placa

```
#define CONTROLLERFAN_SECS 60
```

// duración en segundos de auto apagado

```
#define CONTROLLERFAN_SPEED 255
```

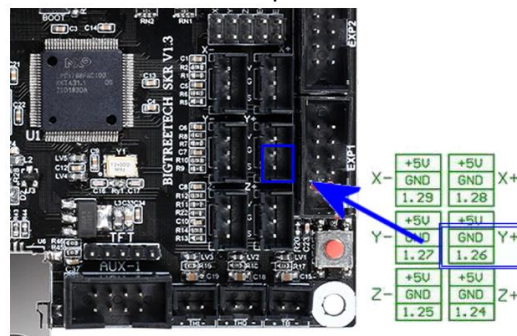
// 255 == full speed

```
#endif
```

En la salida del endstop **Y-max** conectaremos un pequeño interruptor mosfet con un conector JST XH2.54 en el en los pins **P1-26 y GND**. No es algo habitual, y yo personalmente no lo he probado.

Foto de ejemplo y [ENLACE \(1€\)](#)

¿más info?--> [Youtube](#)



## Sensor de Filamento (opcional)

Si vamos a utilizar sensor de filamento, la configuración por defecto utiliza el pin de endstop Xmax (1.28). Deberemos editar el fichero **configuration.h**:

-Descomentar **#define FILAMENT\_RUNOUT\_SENSOR**

-Cambiar de False a True: **#define FIL\_RUNOUT\_INVERTING true**

Fichero **configuration\_adv.h**:

-Descomentar **#define ADVANCED\_PAUSE\_FEATURE**

y también **#define NOZZLE\_PARK\_FEATURE**

Si quisiéramos utilizar el sensor de filamento en otro lugar distinto de XMAX, deberemos modificar el fichero de pins SKR V1.3 (Marlin-bugfix-2.0.x\Marlin\src\pins\lpc1768\pins\_BIGTREE\_SKR\_V1.3.h), ya que por defecto es:

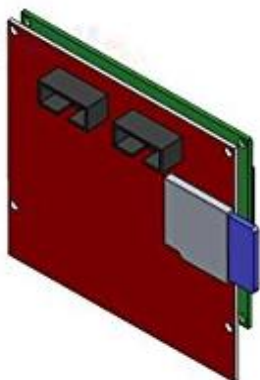
```
#define FIL_RUNOUT_PIN P1_28 (conector Endstop Xmax)
```

X-	+5V	+5V	X+
	GND	GND	
	1.29	1.28	
Y-	+5V	+5V	Y+
	GND	GND	
	1.27	1.26	
Z-	+5V	+5V	Z+
	GND	GND	
	1.25	1.24	

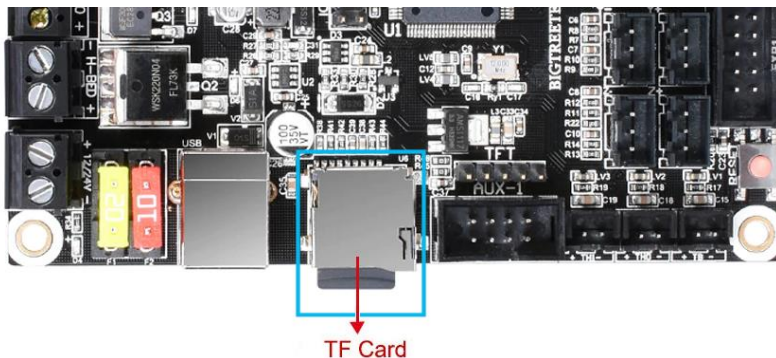
## Selección de tarjeta SD

Si vamos a utilizar tarjeta SD ( habremos descomentado en configuration.h el valor **#define SDSUPPORT** ), deberemos elegir si vamos a utilizar la tarjeta SD integrada con nuestra pantalla/TFT, o si vamos a utilizar la propia tarjeta microSD incorporada en la placa Skr V1.3

(Existe también una tercera opción, menos habitual, que es utilizar una tarjeta externa, en el pin que definamos nosotros)



VS



(o bien)



**SD externas**

En versiones anteriores de marlin2, había que toquitar el fichero de pins pins\_BIGTREE\_SKR\_V1.3.h, pero esto ya no es necesario:

Para elegir cuál de los 3 modos queremos usar, deberemos editar en el fichero **Configuration\_Adv.h** la línea

```
#if HAS_SDCARD_CONNECTION
  // #define SDCARD_CONNECTION ONBOARD
descomentándola y cambiándola a
```

```
#define SDCARD_CONNECTION ONBOARD
#define SDCARD_CONNECTION LCD
#define SDCARD_CONNECTION CUSTOM_CABLE
```

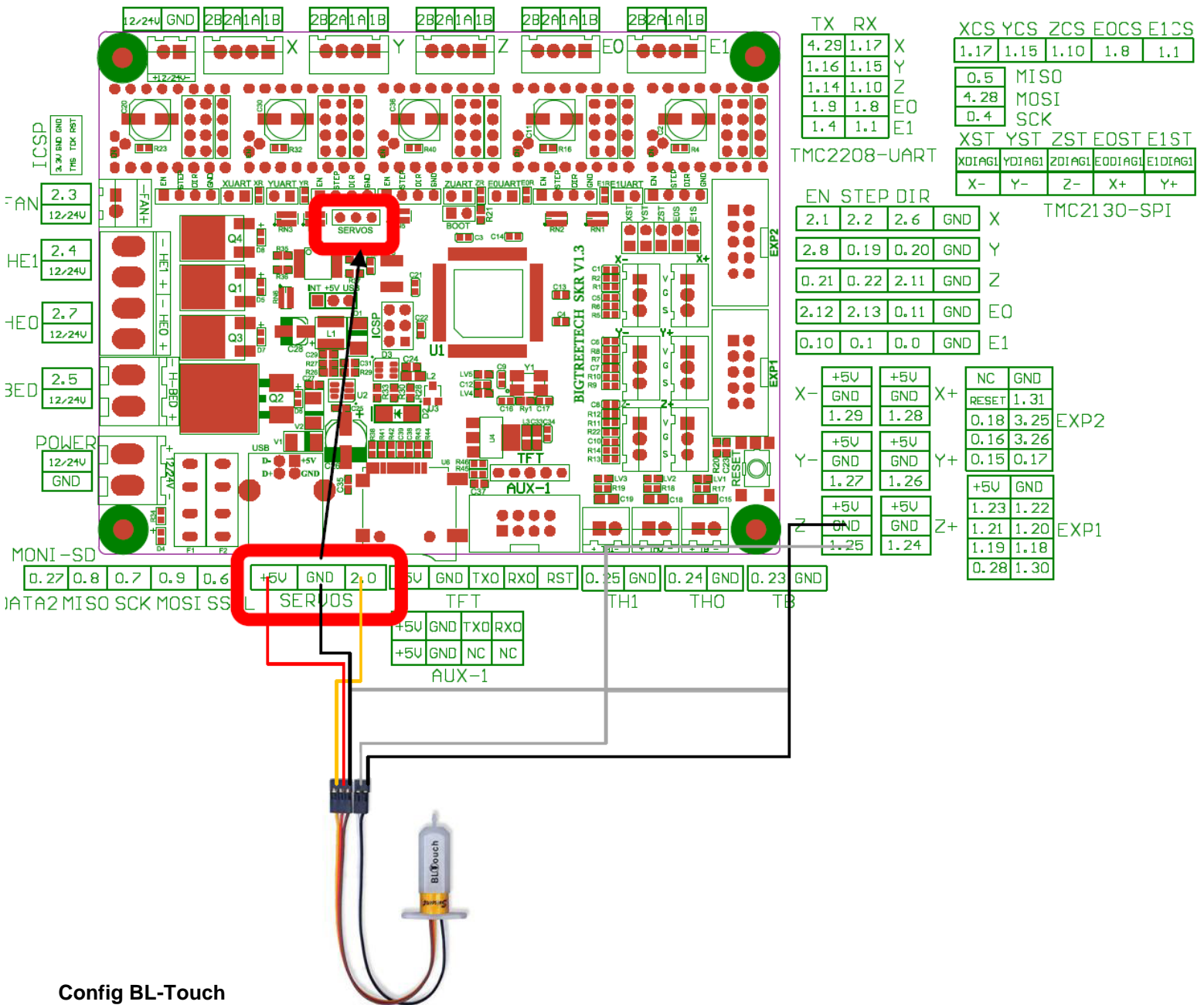
```
//Para utilizar la tarjeta interna
//Para utilizar la tarjeta de nuestra pantalla lcd/tft
//Para tarjetas externas definidas en el fichero de pins.
```

**Briconsejo SD:** Conviene además descomentar la línea del configuración.h **#define SD\_CHECK\_AND\_RETRY** así en caso de hacer una lectura incorrecta de Gcode en la SD, marlin **vuelve a intentarlo en lugar de dar error.**

No es habitual que haya fallos de lectura en SD interna de la placa, pero sí son habituales en la SD del lcd, sobre todo si utilizamos un **cable largo** y no está apantallado con papel de aluminio o Ferrita:



## Diagrama de pins Bigtreetech SKR V1.3 y Bltouch:



### Config BL-Touch

La placa SKR V1.3 tiene un Puerto dedicado para servos. **Se debe revisar el cableado antes de usarlo!**

Para conectar un ANTCLABS BL-touch se utilizan 3 cables al Puerto servo.

En la placa SKR v1.3 el puerto servo tiene = (+)(-)(pulso). En cambio el BL-touch es = (-)(+)(puls).

El pin servo es "2.0" ( #define SERVO0\_PIN P2\_00)

Se deben intercambiar el positivo (+) y negativo (-) en el conector Bltouch.

(o bien usar un cable intermedio adicional que realice el cruce).

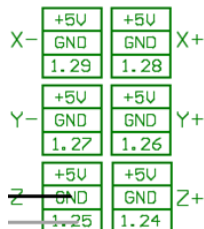
Si necesitas reordenar los cables, es muy sencillo cambiarlos, con un cutter o pinza, levantando la pestañita negra y sacando el conector metálico:



En el BLTouch original (puede cambiar en los clones, o 3Dtouch chinos) los cables son:

- ROJO +5V (+)
- MARRÓN GND, masa. (-)
- NARANJA señal,(pulso)

El conector de 2 cables se conecta al puerto ENDSTOP de Zmin (endstop Z-, entre pin 1\_25 y masa) →



Para activar en Marlin BL-Touch, deberemos realizar los siguientes cambios en el fichero **Configuration.h**

Cambiaremos las siguientes líneas:

```
##define BLTOUCH cambiar a  
#define BLTOUCH
```

```
##define AUTO_BED_LEVELING_BILINEAR cambiar a  
#define AUTO_BED_LEVELING_BILINEAR
```

```
// #define Z_SAFE_HOMING cambiar a  
#define Z_SAFE_HOMING
```

Problemas conocidos Bltouch: Si tu Bltouch original, TL-Touch, o clon baja el pincho durante la impresión, vete al fichero Marlin/src/feature/**bltouch.h** y cambia el siguiente código:

```
#define BLTOUCH_STOW 100 // por defecto es 90  
#define BLTOUCH_SELFTEST 130 // por defecto es 120
```

## BabyStepping

Si quieres activar el [Babystepping](#) iremos al fichero **Configuration\_adv.h** y descomentaremos:

```
#define BABYSTEPPING  
#define DOUBLECLICK_FOR_Z_BABYSTEPPING  
#define BABYSTEP_ALWAYS_AVAILABLE
```

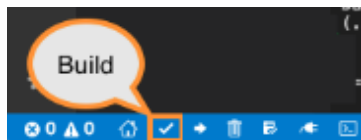
(Revisar seguramente **sobra**)

```
#define BABYSTEP_ZPROBE_OFFSET  
#define BABYSTEP_ZPROBE_GFX_OVERLAY  
#define BABYSTEP_MULTIPLICATOR 20 // (por defecto es 1)
```

## Compilación

Una vez terminadas las modificaciones en la config de Marlin (ante cualquier duda consultad la [biblia de marlin](#) de [StaticBoards](#)), compilaremos.

Para comenzar a compilar pulsaremos en VScode las teclas **Ctrl+Mayúsculas+B**, o bien click en el icono BUILD.



PlatformIO comenzará automáticamente a descargar las librerías que necesite, y compilar los componentes.

Puede costarle hasta 5 minutos, tiempo más que de sobra para ir a mear ☺... o mirar el correo, o bien

saludar a la gente del canal **SKR V1.3 32bits** de Telegram ( [https://t.me/SKR\\_board\\_32bits](https://t.me/SKR_board_32bits) )



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Compiling .pioenvs\LPC1768\FrameworkArduino\wInterrupts.cpp.o
Compiling .pioenvs\LPC1768\FrameworkArduino\wire.cpp.o
Compiling .pioenvs\LPC1768\FrameworkArduino\arduino.cpp.o
Compiling .pioenvs\LPC1768\FrameworkArduino\main.cpp.o
Compiling .pioenvs\LPC1768\FrameworkArduino\pwm.cpp.o
Archiving .pioenvs\LPC1768\libFrameworkArduino.a
Linking .pioenvs\LPC1768\firmware.elf
Checking size .pioenvs\LPC1768\firmware.elf
Building .pioenvs\LPC1768\firmware.bin
Memory Usage -> http://bit.ly/pio-memory-usage
DATA: [== ] 23.2% (used 7564 bytes from 32568 bytes)
PROGRAM: [== ] 20.5% (used 97368 bytes from 475136 bytes)
===== [SUCCESS] Took 179.05 seconds =====

===== [SUMMARY] =====
Environment megaatmega2560 [SKIP]
Environment megaatmega1280 [SKIP]
Environment at90usb1286_cdc [SKIP]
Environment at90usb1286_dfu [SKIP]
Environment DUE [SKIP]
Environment DUE_USB [SKIP]
Environment DUE_debug [SKIP]
Environment LPC1768 [SUCCESS]
Environment LPC1769 [SKIP]
Environment melzi [SKIP]
Environment melzi_optiboot [SKIP]
Environment rambo [SKIP]
Environment sanguino_atmega644p [SKIP]
Environment sanguino_atmega1284p [SKIP]
Environment STM32F1 [SKIP]
Environment STM32F4 [SKIP]
Environment ARMED [SKIP]
Environment teensy35 [SKIP]
Environment malyanm200 [SKIP]
Environment esp32 [SKIP]
Environment fysetc_f6_13 [SKIP]
===== [SUCCESS] Took 179.08 seconds =====

Terminal will be reused by tasks, press any key to close it.
```

## SUCCESS!! :

Tras finalizar la compilación con éxito (**SUCCESS**), se habrá generado un fichero “**firmware.bin**” dentro de la carpeta \pioenvs\LPC1768\

( ...Ruta de marl ñ en tu PC\ Marlin-bugfix-2.0.x\pio\build\LPC1768\firmware.bin )

Copia ese fichero en la tarjeta microSD, ponla en la placa, y resetea o reiníciala.

Durante el inicio, la placa grabará ese firmware en la memoria interna de la placa SKR. El fichero una vez subido a la placa se renombra automáticamente en la SD a “firmware.cur” (de Current, actual...)

NO ES NECESARIO que esté la SD insertada, el firmware está ya almacenado en la propia memoria interna de la placa. Es aconsejable renombrar y guardar como backup tu actual firmware.cur, si tu actual marlin 2.0 compilado estaba funcionando OK. Este podrá volver a flashearse, si lo dejamos de nuevo renombrado como firmware.bin

Esa misma tarjeta puede utilizarse para almacenar ficheros Gcode e imprimir desde ellos (si la has configurado como ONBOARD en la [sección SD](#))

**NO SUCCESS!!** 🙄 Si la compilación hubiera dado errores. Revisa en la misma ventana negra de la consola más arriba, en busca del **primer error (en rojo)**, y búscalo en san google, sigue buscándolo, busca otra vez más, y ya por fin.. busca otra vez... y después pregunta en telegram a alguien en busca de ayuda.

Nota: Durante la compilación aparecen **alertas (en amarillo)**, pero estas pueden ignorarse en el 99% de los casos. Son los **errores (en rojo)** los que harán que la compilación no termine con éxito.

## LCD. Problemas

Si tu LCD 128x64 pixeles ( REPRAP\_DISCOUNT\_FULL\_GRAPHIC\_SMART\_CONTROLLER) no está funcionando, revisa lo siguiente:

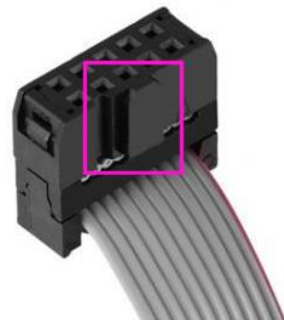
-Si la retroiluminación (luz de fondo) está encendida, pero no se muestran caracteres, has elegido un tipo de pantalla incorrecto en la configuración del firmware en el fichero configuration.h. Cámbialo.

-Si la pantalla permanece apagada, quizás el cable EXP1 esté puesto en el conector EXP2. Conecte el cable EXP1 en el conector EXP1 de la placa.

-Si la pantalla permanece apagada estando EXP1 bien conectado en su lugar (exp1), deberemos quitar la pestaña de bloqueo del conector plástico, para poder conectarlo girado 180 grados (es decir, del revés). Esta pestaña puede cortarse con un cutter, o alicates, de forma sencilla.

Una vez cortado en ambos EXP1 y EXP2, los conectaremos girados.

NOTA: Esta operación puede realizarse sin miedo, **NO es posible romper el LCD o la placa** por conectarlos del revés.



**-Problemas conocidos:** Si una vez instalada la pantalla, la ruedecita se comporta raro, podremos cambiar en el fichero Configuration.h lo siguiente:

Descomentar

```
#define ENCODER_PULSES_PER_STEP 4
```

y


```
#define REVERSE_ENCODER_DIRECTION
```

## RASPBERRY PI

Conectando la SKR V1.3 Con Raspberry Pi3B+ a través de conexión serie sin cable USB.

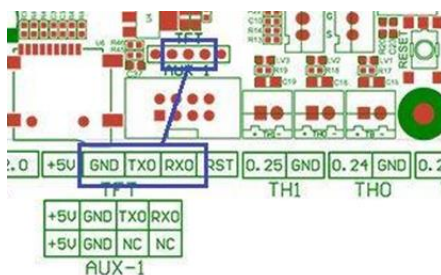
Conecte TXD0 a RXD0 y RX a TX (conexión cruzada) y masa (GND)

PI 3B+ @ GPIO connector TXD0 - RXD0 – GND (8-0-10)



		Pin no.		
DC Power	3.3V	1	2	5V
SDA1, I²C	GPIO 2	3	4	5V
SCL1, I²C	GPIO 3	5	6	GND
GPIO_GCLK	GPIO 4	7	8	GPIO 14
	GND	9	10	GPIO 15
GPIO_GEN0	GPIO 17	11	12	GPIO 18
GPIO_GEN2	GPIO 27	13	14	GND
GPIO_GEN3	GPIO 22	15	16	GPIO 23
DC Power	3.3V	17	18	GPIO 24
SPI_MOSI	GPIO 10	19	20	GND
SPI_MISO	GPIO 9	21	22	GPIO 25
SPI_CLK	GPIO 11	23	24	GPIO 8
	GND	25	26	GPIO 7
I²C ID EEPROM	DNC	27	28	DNC
	GPIO 5	29	30	GND
	GPIO 6	31	32	GPIO 12
	GPIO 13	33	34	GND
	GPIO 19	35	36	GPIO 16
	GPIO 26	37	38	GPIO 20
	GND	39	40	GPIO 21

SKR V1.3 @ TFT connector TX – Rx - GND

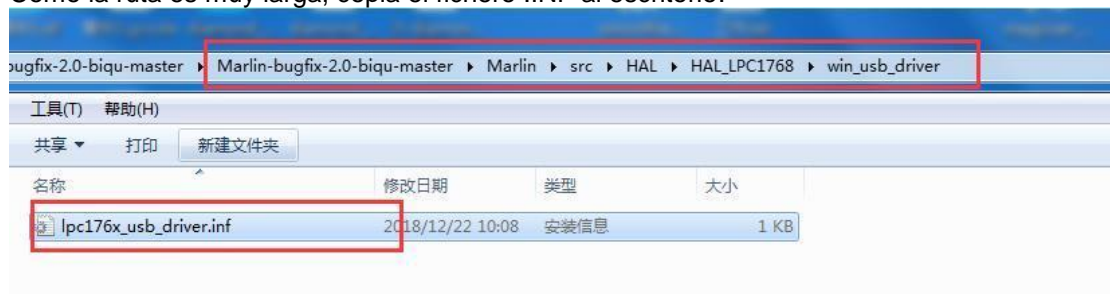


## Driver sistema operativo

Debido al Plug & Play, en Windows 10 se instalará de forma automática un driver llamado MARLIN al conectar por USB la placa. Aparecerá un puerto serie MARLIN. No instales ningún otro driver, ni los drivers de Smoothieware si usas Windows 10!

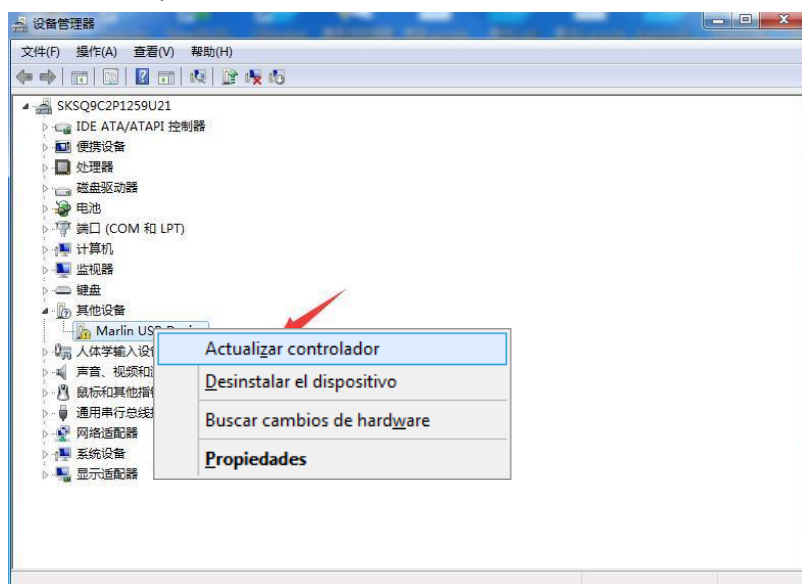
Para el resto de Sistemas operativos Windows, se necesitará instalar manualmente un driver de puerto serie USB , que está disponible en la carpeta debajo indicada.

Como la ruta es muy larga, copia el fichero .INF al escritorio.



Abriremos el administrador de dispositivos (Tecla Windows+Pausa y click admin dispositivos), o bien ejecutar *compmgmt.msc*. y buscaremos el dispositivo con símbolo de error/advertencia.

Botón derecho en el dispositivo, click en Actualizar controlador



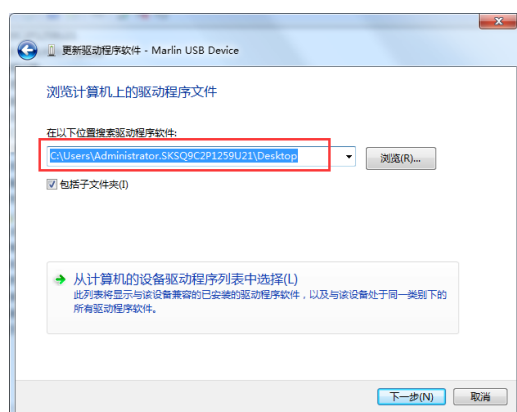
Buscar de forma manual.

→ [Buscar software de controlador en el equipo](#)  
[Buscar e instalar el software de controlador de forma manual.](#)

Elegiremos la ruta de la carpeta donde tenemos el fichero **lpc176x\_usb\_driver.inf**.

Si la habíamos copiado en el escritorio estará en C:\Users\tu\_usuario\Desktop\

, y hacemos click en siguiente.



Si aparecen alertas del cortafuegos, haremos click en Instalar Siempre.



Si la instalación del driver finaliza con éxito, el driver ya estará instalado. **Recuerda el número de Puerto.**

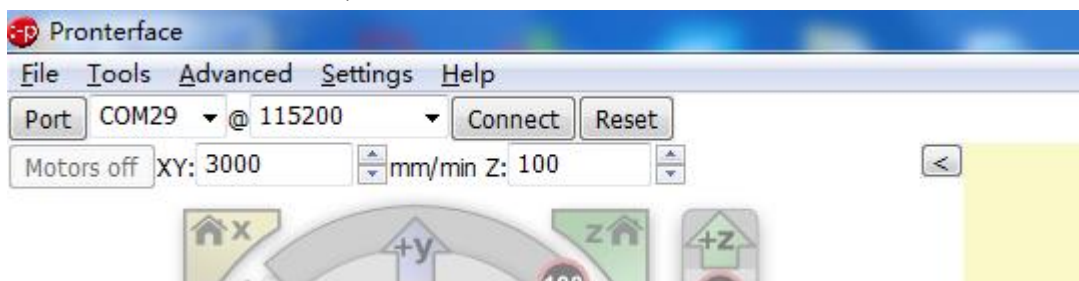


## Pruebas / Conexión USB / Status

Abre la consola de [Printrun/Pronterface](#) o bien la de [Repetier Host](#) (por ejemplo), elige tu número de puerto serie y velocidad en baudios (ej 115200), y haz click en Connect.

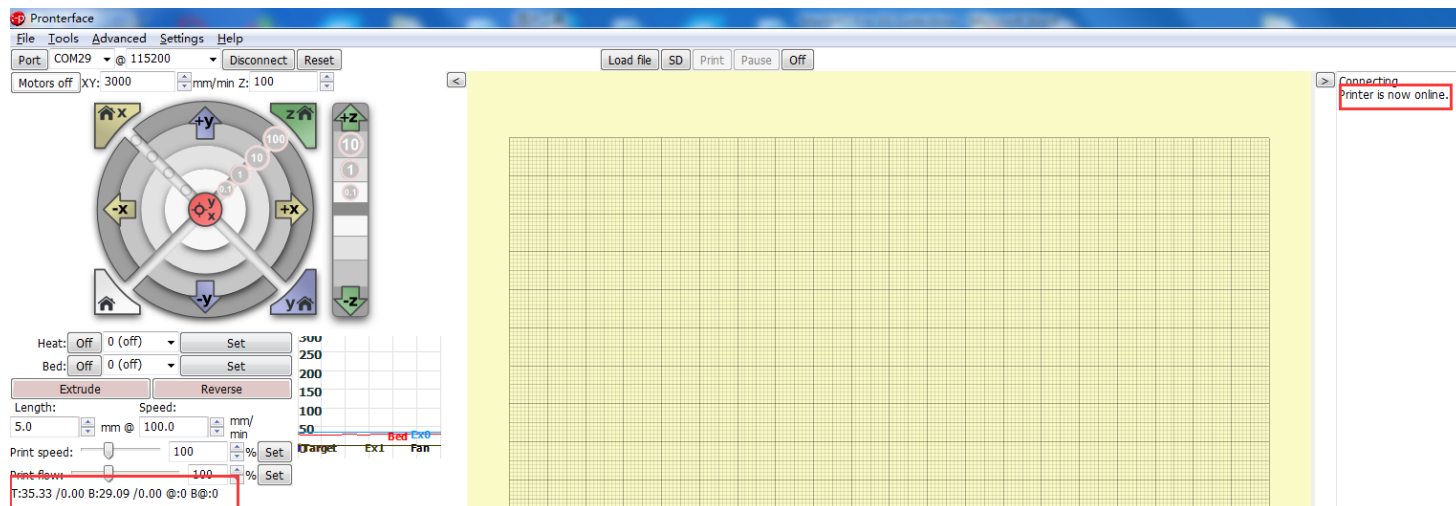
Simplify también tiene una consola manual parecida, pero es bastante floja.

(El valor de la velocidad en baudios será 115200, o la velocidad que hayas puesto en tu configuration.h , en la variable `#define BAUDRATE 115200` )



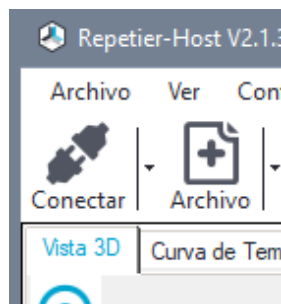
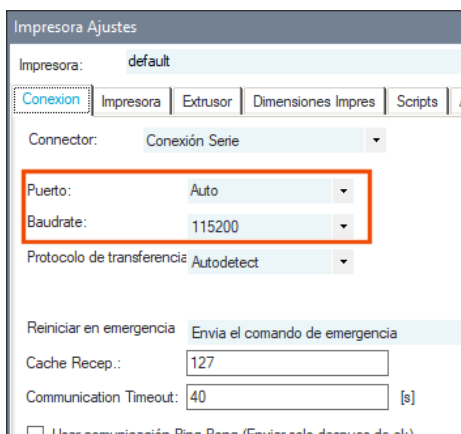
Tu número de puerto posiblemente sea distinto. Una vez le damos a connect, en la ventana amarilla a la derecha, podremos ver si ha conectado OK, mostrando que la impresora está Online.

Esto indicará que ha conectado bien. Ahora puedes utilizar tu PC para controlar la máquina, hacer endstop de cada eje, mover los ejes, o lanzar comandos Gcode.

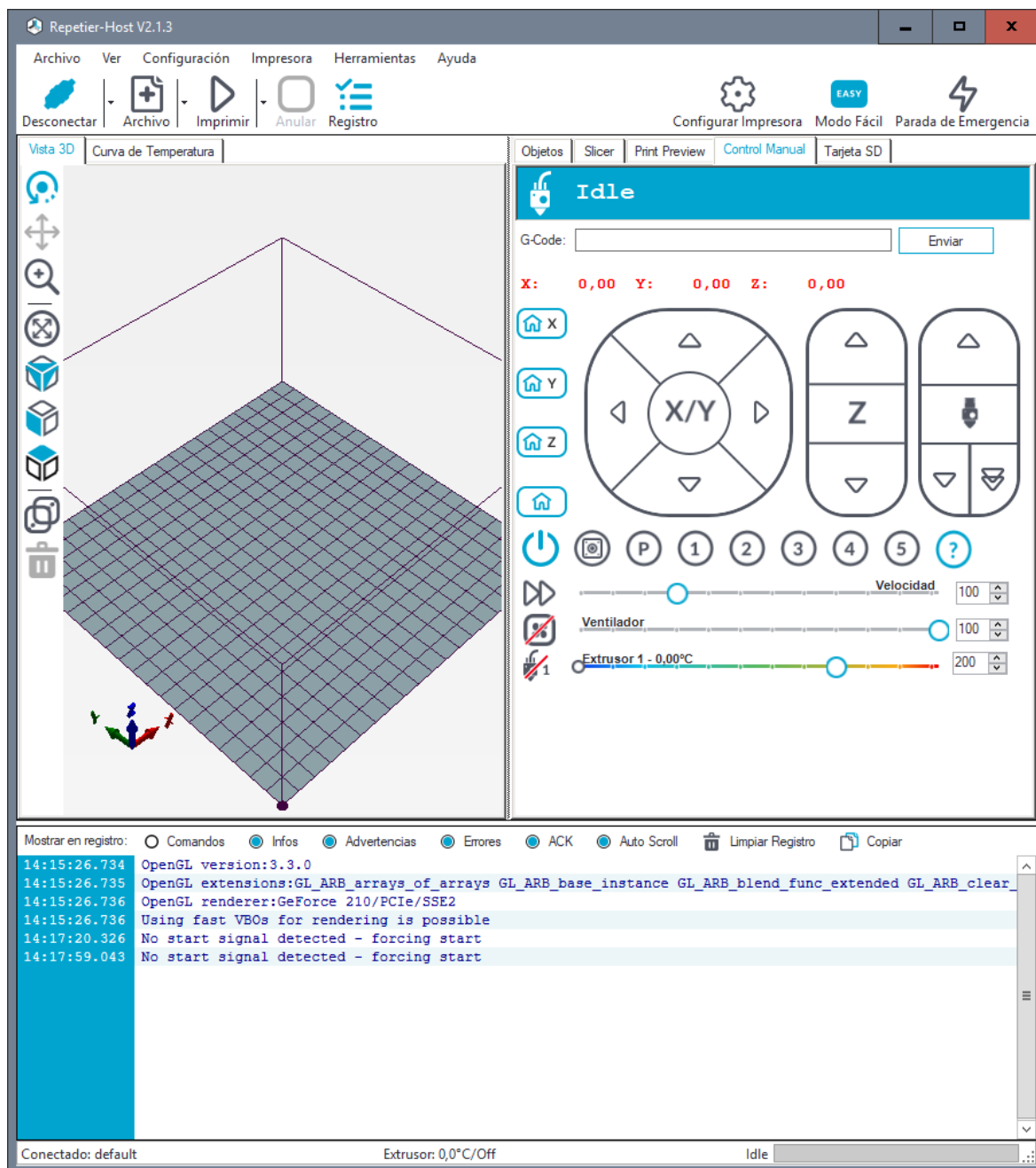




Si utilizas **Repetier Host**, en configuración/ configurar impresora, pon el puerto en AUTO y elige velocidad en baudios. y tras aceptar, pulsamos en Conectar.



Una vez conectado, podremos utilizar/probar manualmente la impresora desde el control manual, enviar Gcodes sueltos, y ver su respuesta en la parte inferior, como por ejemplo la configuración actual de la máquina tras enviar un Gcode **M503** (mostrar datos eeprom) o **M122** (mostrar estado drivers Trinamics TMC). Si no apareciera la ventana inferior con el log de texto, pulsaremos sobre el botón Registro.



## DRIVERS: JUMPERS STEP/DIR

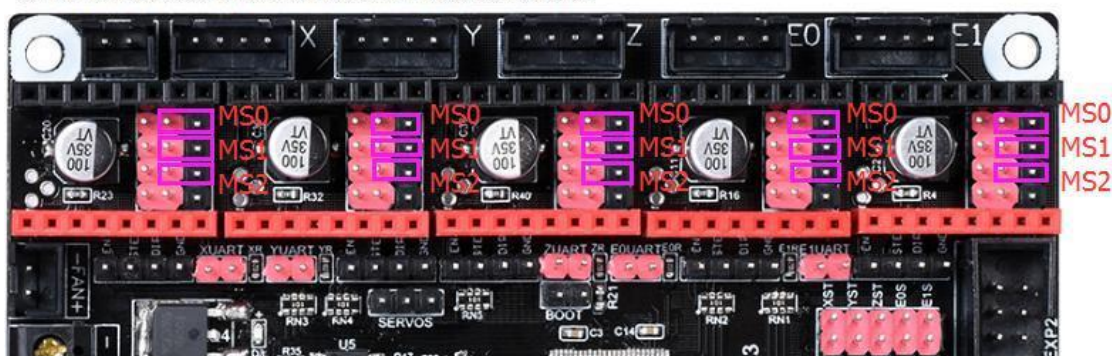
Ajuste de jumpers de driver para uso en modo **standalone** (ni UART ni SPI)

El modo standalone, también es conocido como el modo modo clásico **STEP/DIR**.

Dado que nuestra placa soporta los modos Spi o Uart para controlar los drivers por software con jumpers, sin utilizar un solo cablecillo externo, el modo STEP/DIR no es el más recomendado. Es menos versátil, aunque más sencillo de configurar para algunos usuarios, o marcas-modelo de drivers. Se configuran los jumpers, se pincha el driver, y se le ajusta la tensión Vref, desde el potenciómetro. Es el modo de toda la vida (placas Ramps, etc).

### STEP/DIR Mode

Contrast with various stepper motor drive subdivision selection tables, connect the purple frame in the figure below with short-circuit cap. MS0, MS1, MS2 can also be expressed as MS1, MS2, MS3. Different driver numbers are different but use method is the same.



En el modo SPI/DIR, los drivers se configuran con los zócalos para jumpers MS0, MS1 y MS2, siguiendo las tablas del fabricante de cada driver. En este modo sólo se utilizan (poniendo o quitando jumpers) las zonas indicadas en los 3 recuadros ROSA. El resto no se usa.

### TMC2100

Los drivers de tipo TMC2100 se controlan mediante 3 posibles estados.

0 = pin conectado a GND/MASA mediante cablecillo.

1 = pin conectado a +V con jumper entre rosa y negro.

OPEN = pins al aire SIN jumper

TMC2100	steps	MS0	MS1	MS2	Interpolado	modo
	Full	0	0	Open	No	Spreadcycle
	1/2	1	0	Open	No	Spreadcycle
	1/4	Open	1	Open	SI 256	Spreadcycle
	1/16	0	1	Open	No	Spreadcycle
	1/4	1	1	Open	No	Spreadcycle
	1/4	Open	1	Open	SI 256	Spreadcycle
	1/16	0	1	Open	SI 256	Spreadcycle
	1/4	1	Open	Open	SI 256	Stealthchop1
	1/16	Open	Open	Open	SI 256	Stealthchop1

### TMC2208

Los drivers de tipo TMC2208 se controlan mediante 2 posibles estados:

0 = SIN jumper.

1 = Pin conectado a +V con jumper entre rosa y negro

TMC2208	steps	MS0	MS1	MS2	Interpolation	Mode
	1/2	0	0	0	1/256	Stealthchop2
	1/4	0	1	0	1/256	Stealthchop2
	1/8	0	0	0	1/256	Stealthchop2
	1/16	1	1	0	1/256	Stealthchop2

### TMC2209

Los drivers de tipo TMC2209 se controlan mediante 2 posibles estados, más un pin aparte SPREAD:

0 = SIN jumper.

1 = Pin conectado a +V con jumper entre rosa y negro

steps	MS0	MS1	Interpolado	
8	0	0	SI 256	
32	0	1	SI 256	Distinto a TMC2208!
64	1	0	SI 256	Distinto a TMC2208!
16	1	1	SI 256	

### PIN SPREAD:

GND O ABIERTO	→ STEALTHCHOP
VCC_IO	→ SPREADCYCLE

### TMC2130

Los driver TMC2130 drivers tienen 3 posibles estados

Los drivers de tipo TMC2130 se controlan mediante 3 posibles estados:

0 = pin conectado a GND/MASA mediante cablecillo.

1 = pin conectado a +V con jumper entre rosa y negro.

OPEN = pins al aire SIN jumper

TMC2130	steps	MS0	MS1	MS2	Interpolation	Mode
	Full	0	0	Open	No	Spreadcycle
	1/2	1	0	Open	No	Spreadcycle
	1/2	Open	1	Open	1/256	Spreadcycle
	1/4	0	1	Open	No	Spreadcycle
	1/16	1	1	Open	No	Spreadcycle
	1/4	Open	1	Open	1/256	Spreadcycle
	1/16	0	Open	Open	1/256	Spreadcycle
	1/4	1	Open	Open	1/256	Stealthchop1
	1/16	Open	Open	Open	1/256	Stealthchop1

### A4988, DRV8825, LV8729

Los siguientes drivers tienen 2 posibles estados.

0 = SIN jumper entre rosa y negro

1 = Pin conectado a V+ (CON jumper entre rosa y negro)

A4988	steps	MS0	MS1	MS2	Interpolation	Mode
	Full	0	0	0	No	None
	1/2	1	0	0	No	None
	1/4	0	1	0	No	None
	1/8	0	1	0	No	None
	1/16	1	1	1	No	None

DRV8825	steps	MS0	MS1	MS2	Interpolation	Mode
	Full	0	0	0	No	None
	1/2	1	0	0	No	None
	1/4	0	1	0	No	None
	1/8	1	1	0	No	None
	1/16	0	0	1	No	None
	1/32	1	1	1	No	None

LV8729	Steps	MS0	MS1	MS2	Interpolation	Mode
	Full	0	0	0	No	None
	1/2	1	0	0	No	None
	1/4	0	1	0	No	None
	1/8	1	1	0	No	None
	1/16	0	0	1	No	None
	1/32	1	0	1	No	None
	1/64	0	1	1	No	None
	1/128	1	1	1	No	None

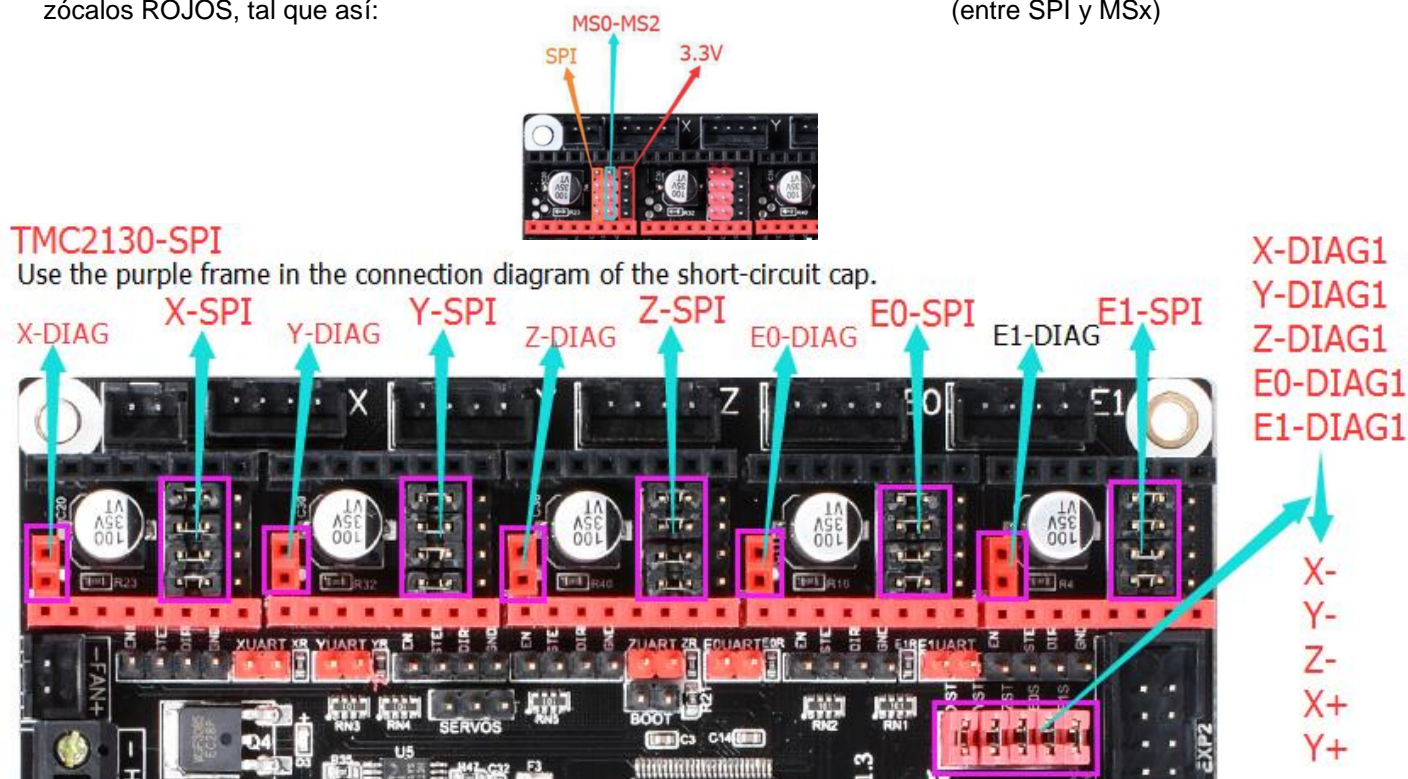


# DRIVERS: JUMPERS SPI/UART

## Ajuste de jumpers de driver para uso en modo **SPI / UART**

En el modo spi/uart la electrónica se comunica con el driver utilizando un puerto serie. Este puerto serie puede ser de tipo UART (en el caso de los TMC2208, TMC2209) o de tipo SPI (en el caso de los TMC2130, TMC5160, TMC5161, etc).

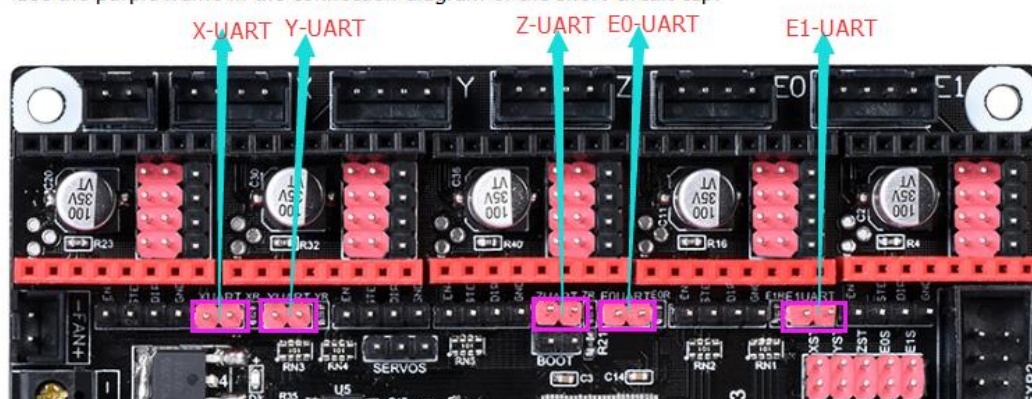
**JUMPERS SPI:** Siguiendo las instrucciones de biqu, para poner drivers en modo SPI, deberemos utilizar jumpers en los 4 zócalos ROJOS, tal que así:



**JUMPERS SPI:** Siguiendo las indicaciones del fabricante, para poner drivers en UART, deberemos quitar los 4 jumpers bajo el driver, y poner el jumper Uart de cada uno, según la imagen:

### TMC2208-UART-Mode

Use the purple frame in the connection diagram of the short-circuit cap.



NOTA: Esto **NO es válido** para SKR MINI – SKR MINI E3 / SKR MINI E3 DIP / SKR PRO

Si además queremos usar modo sin Endstops (tanto para SPI como para UART), deberemos utilizar jumpers en X-Diag1, Y-Diag1, etc.

Si nuestro driver no está preparado para SPI o UART, **deberemos SOLDARLO.** ([ver guía soldar drivers aquí](#)) Hay fabricantes (aliexpress) que venden el mismo driver preparado ya para spi/uart, o sin preparar.



Under construction



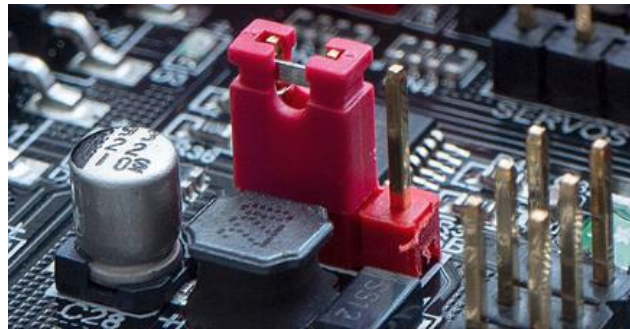
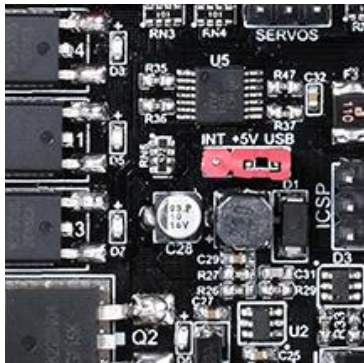
Under construction

Estamos de obras por aquí...

Si el Jumper está configurado para usarse como USB, 5v (hacia la derecha), la placa se alimenta de la tensión que recibe del USB, y los drivers TMC dirán CONNECTION ERROR.

Si estamos compilando-probando Marlin sólo con corriente del USB, los TMC también darán “tmc connection error”

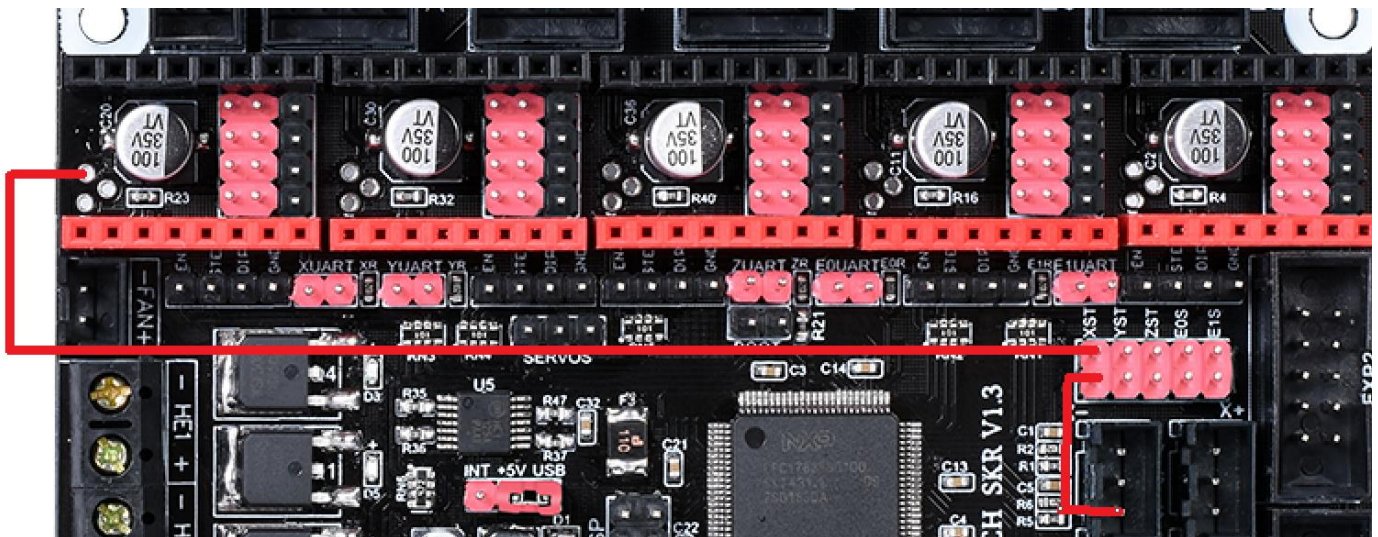
Se requiere alimentar la placa a 12 ó 24v para que los TMC funcionen bien.



## SENSORLESS\_HOMING (STALLGUARD)

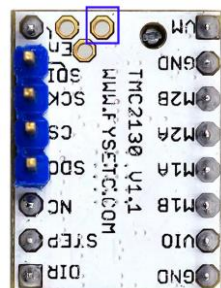
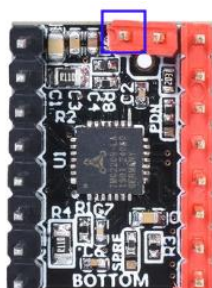
El modo sin Endstops de los TMC2130/TMC2209 requiere poner jumpers en X-diag1, Y-diag1, etc.

Esto, internamente, está conectado así:



En el modo sin endstops, el driver Trinamics TMC detecta (utilizando Stallguard) que algo está frenando al motor (al haber llegado este al final de carrera). En ese caso, el driver Tmc envía por el pin Diag0 una señal. Esta señal, llega al pin de ENDSTOP.

Requiere utilizar drivers TMC que tengan la función [STALLGUARD](#). Es decir, drivers TMC2130 o bien TMC2209. (y TMC516x). Estos driver además deberán tener soldado el pin DIAG0 (azul), para que conecte con el de la placa SKR.



El modo sin endstops, se habilita en Marlin descomentando en configuration\_adv.h

### **#define SENSORLESS\_HOMING**

Tras habilitarlo, deberemos configurar la sensibilidad de cada eje

En Marlin por defecto (justo debajo, mismo fichero)

```
#define X_STALL_SENSITIVITY 8
```

y

```
#define Y_STALL_SENSITIVITY 8
```

Podemos ajustar la sensibilidad en caliente con el comando [Gcode M914](#)

Los TMC2130 van de lo más duro +63 a lo más sensible -64

Los TMC2239 van de lo más duro 0 a lo más sensible 255

Si lo configuramos muy sensible, el motor ni llegará a moverse, falso positivo, endstop siempre pulsado.

Si lo configuramos muy duro, el motor puede que nunca detecte endstop o al hacerlo pegue golpes fuertes al tope físico.

No hay medidas mágicas, depende de cada motor. Algo intermedio suele estar bien (entre 60 y 150 para un TMC2209 y entre -20 y 20 para un TMC2130).

Si queremos poner, por ejemplo, para un TMC2209 un valor de X de 80, y un valor de Y de 105, usando el Gcode M914 sería:

```
M914 X80
```

```
M914 Y105
```

```
M500 (guardar cambios eeprom)
```

No es aconsejable utilizar Stallguard en el eje Z. Al ser el husillo ratio 1:1, sobre el propio eje del motor, para cuando el driver detectase que “algo está frenando el motor”, es muy posible que tengas el nozzle incrustado en tu cama doblada.

Nunca-unca (never ever☺) haré la prueba.

Si algún valiente se anima, hay que descomentar **#define SENSORLESS\_PROBING**

Mas info: [http://marlinfw.org/docs/hardware/tmc\\_drivers.html](http://marlinfw.org/docs/hardware/tmc_drivers.html)

En breve más sobre TMC...

# WINDOWS 10: SÓLO DRIVERS FIRMADOS WINDOWS

## Windows 10: Deshabilitar el uso obligatorio de controladores firmados

### ¿Cómo puedo instalar controladores que no están firmados digitalmente?

Windows 10 predeterminadamente fuerza el uso obligatorio de controladores firmados. Esto puede desactivarse para instalar controladores que no están firmados digitalmente. Siga los pasos a continuación para deshabilitar el uso obligatorio de controladores firmados.

**Forma sencilla:** Abrimos un CMD como administrador, y ejecutamos el comando:



```
bcdedit.exe /set nointegritychecks on
```

Tras ello, reinicia el ordenador. Este cambio es permanente.

(Si alguna vez quisieras volver a habilitar el uso de sólo drivers firmados → `bcdedit.exe /set nointegritychecks off` )

**Forma "Microsoft"** (80 pasos, más incómodo):

Siga los pasos a continuación para deshabilitar el uso obligatorio de controladores firmados:

1. Click en botón de inicio , y configuración 
2. Click **Actualización y seguridad**.
3. Click en **Recuperación**. 
4. Click **Reiniciar ahora** en el apartado **Inicio avanzado**.
5. Click **Solucionar problemas**.
6. Click **opciones avanzadas**.
7. Click **Configuración de inicio**.
8. Click on **Reiniciar**.
9. En la pantalla de Configuración de inicio pulse 7 o F7 para deshabilitar el uso obligatorio de controladores firmados.

El equipo se reiniciará y podrá instalar controladores que no tengan firma digital. Si has usado el método Microsoft, al reiniciar el equipo nuevamente se habilitará de nuevo el uso obligatorio de controladores firmados.

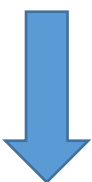
---

Este documento no ha sido creado por BIGTREETECH!

Documento original creado por 2019 [Jupa Creations](#). Netherlands.

Traducción inicial y múltiples añadidos/correcciones posteriores por @lokus77 telegram

Sigue



## Apéndices:

### Apéndice N°1.

**Fichero platformio.ini** : Valores de default\_envs para otras placas distintas a “skr 1.3”

Hasta hace poco (Julio 2019) esta variable se llamaba **env\_default**, pero ahora es **default\_envs** )

La placa SKR V1.3 utiliza 'default\_envs = LPC1768' dado que su CPU es una LPC1768.

Para otras placas SKR, el 'default\_envs' es distinto:

- Arduinomega, mks gen, genL, etc etc, el clásico (y por defecto)

**default\_envs = megaatmega2560**

- SKR PRO

**default\_envs = BIGTREE\_SKR\_PRO**

fichero configuration.h: #define MOTHERBOARD BOARD\_BIGTREE\_SKR\_PRO\_V1\_1

Marlin2 editado por biqu para esta placa: <https://github.com/bigtreetech/BIGTREETECH-SKR-PRO-V1.1/tree/master/firmware/Marlin-SKR-Pro>

- SKR MINI E3

**default\_envs = STM32F103R\_bigtreetech**

(hasta hace poco era **BIGTREE\_SKR\_MINI** pero ha cambiado a primeros Agosto 2019)

fichero configuration.h: #define MOTHERBOARD BOARD\_BIGTREE\_SKR\_MINI\_E3

Marlin2 editado por biqu para esta placa: <https://github.com/bigtreetech/BIGTREETECH-SKR-mini-E3/tree/master/firmware>

- SKR MINI E3 DIP

**default\_envs = STM32F103R\_bigtreetech**

(hasta hace poco era **BIGTREE\_SKR\_MINI** pero ha cambiado a primeros Agosto 2019)

fichero configuration.h: #define MOTHERBOARD BOARD\_BIGTREE\_SKR\_E3\_DIP

Marlin2 editado por biqu para esta placa: <https://github.com/bigtreetech/BIGTREETECH-SKR-E3-DIP-V1.0/tree/master/Firmware/Marlin-2.0.x-SKR-E3-DIP>

- SKR MINI v1.1

**default\_envs = STM32F103R\_bigtreetech**

(hasta hace poco era **BIGTREE\_SKR\_MINI** pero ha cambiado a primeros Agosto 2019)

fichero configuration.h: #define MOTHERBOARD BOARD\_BIGTREE\_SKR\_MINI\_V1\_1

Marlin2 editado por biqu para esta placa: <https://github.com/bigtreetech/BIGTREETECH-SKR-MINI-V1.1/tree/master/firmware>



- Otras NO SKR:

Valid names are :

ARMED	melzi
at90usb1286_cdc	melzi_optiboot
at90usb1286_dfu	mks_robin
BIGTREE_BTT002	mks_robin_lite
BIGTREE_SKR_PRO	mks_robin_mini
DUE	mks_robin_nano
DUE_debug	rambo
DUE_USB	SAMD51_grandcentral_m4
esp32	sanguino_atmega1284p
fysetc_f6_13	sanguino_atmega644p
include_tree	STM32F103R
jgaurora_a5s_a1	STM32F103R_bigtree
linux_native	STM32F103R_fysetc
LPC1768	STM32F103V_longer
LPC1769	STM32F4
malayanm200	STM32F407VE_black
megaatmega1280	STM32F7
megaatmega2560	teensy31
	teensy35