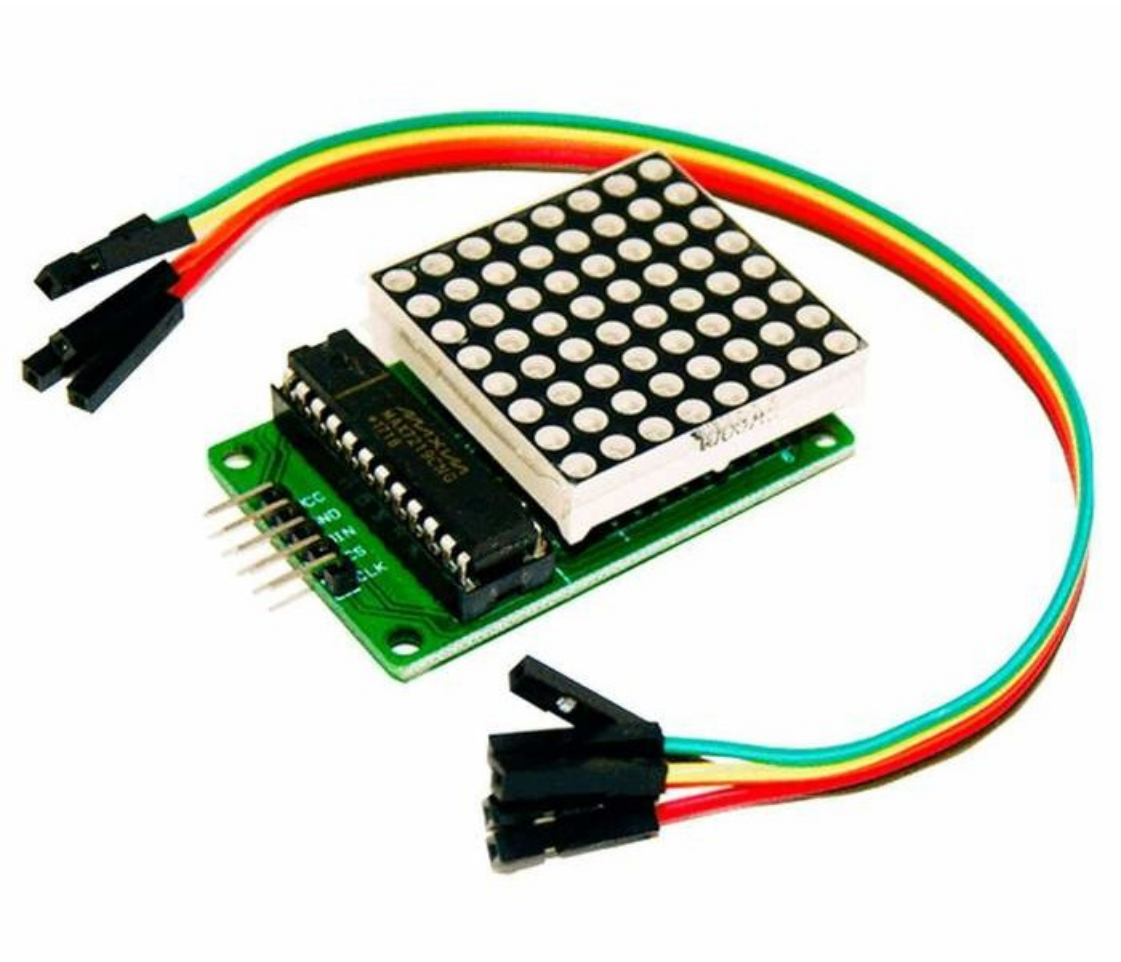
Bienvenido a nuestro sitio web

Gracias por elegir nuestro *módulo de pantalla matriz de 64 LED* de AZ-Delivery. En las siguientes páginas le explicaremos cómo configurar y utilizar el dispositivo.

¡Que te diviertas!



Areas de aplicación
Educación y enseñanza: uso en escuelas, universidades e instituciones de formación para enseñar los conceptos básicos de electrónica, programación y sistemas integrados. Investigación y desarrollo: Uso en proyectos de investigación y desarrollo para crear prototipos y experimentos en los campos de la electrónica y la informática. Desarrollo de prototipos: Uso en el desarrollo y prueba de nuevos circuitos y dispositivos electrónicos. Proyectos Hobby and Maker: utilizado por entusiastas y aficionados a la electrónica para desarrollar e implementar proyectos de bricolaje.

Conocimientos y habilidades requeridos.
Conocimientos básicos de electrónica e ingeniería eléctrica. Conocimientos de programación, especialmente en el lenguaje de programación C/C++. Capacidad para leer esquemas y diseñar circuitos simples. Experiencia trabajando con componentes electrónicos y soldadura.

Condiciones de operación
El producto sólo puede funcionar con los voltajes especificados en la hoja de datos para evitar daños. Se requiere una fuente de alimentación CC estabilizada para su funcionamiento. Al realizar la conexión a otros componentes y circuitos electrónicos, se deben observar los límites máximos de corriente y voltaje para evitar sobrecargas y daños.

Condiciones ambientales
El producto debe utilizarse en un ambiente limpio y seco para evitar daños causados por la humedad o el polvo. Proteja el producto de la luz solar directa (UV), ya que esto puede afectar negativamente la vida útil de la pantalla.

Uso previsto
El producto está diseñado para su uso en entornos educativos, de investigación y desarrollo. Se utiliza para desarrollar, programar y crear prototipos de proyectos y aplicaciones electrónicos. El producto no pretende ser un producto de consumo terminado, sino más bien una herramienta para usuarios con conocimientos técnicos, incluidos ingenieros, desarrolladores, investigadores y estudiantes.

Uso inadecuado previsible
El producto no es adecuado para uso industrial o aplicaciones relevantes para la seguridad. No se permite el uso del producto en dispositivos médicos o para fines de aviación y viajes espaciales.

desecho
¡No lo deseche con la basura doméstica! Su producto es acorde al europeo. Directiva sobre residuos de aparatos eléctricos y electrónicos que deben eliminarse de forma respetuosa con el medio ambiente. Las valiosas materias primas contenidas en ellos se pueden reciclar. convertirse en. La aplicación de esta directiva contribuye a la protección del medio ambiente y la salud. Utilice el punto de recogida habilitado por su municipio para devolver y Reciclaje de aparatos eléctricos y electrónicos viejos. N.º registro RAEE: DE 62624346

descarga electrostática
La pantalla es sensible a las descargas electrostáticas (ESD), que pueden dañar o destruir los componentes electrónicos. Tenga en cuenta las siguientes instrucciones de seguridad para evitar riesgos de ESD: Atención: Las cargas electrostáticas en su cuerpo pueden dañar la pantalla. Nota: Conéctese a tierra usando una muñequera antiestática conectada a una superficie con conexión a tierra o tocando una superficie metálica con conexión a tierra antes de manipular la pantalla. Atención: Utilice bolsas y tapetes antiestáticos para proteger la pantalla. Nota: Coloque la pantalla sobre una alfombra de trabajo antiestática y guárdela en bolsas antiestáticas cuando no esté en uso. Nota: Un lugar de trabajo limpio y conectado a tierra minimiza el riesgo de ESD. Acción: Mantenga su lugar de trabajo limpio y libre de materiales que puedan generar cargas electrostáticas. Asegúrese de que todas las superficies utilizadas estén conectadas a tierra. Atención: Un lugar de trabajo limpio y conectado a tierra minimiza el riesgo de ESD. Nota: Mantenga su lugar de trabajo limpio y libre de materiales que puedan generar cargas electrostáticas. Asegúrese de que todas las superficies utilizadas estén conectadas a tierra.

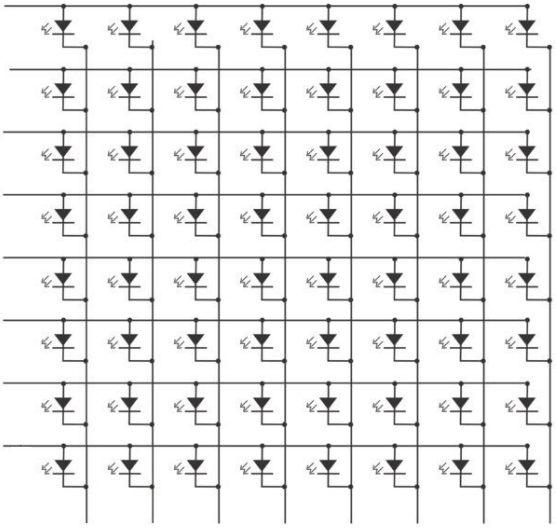
instrucciones de seguridad
Aunque la pantalla cumple con los requisitos de la Directiva RoHS (2011/65/UE) y no contiene sustancias peligrosas en cantidades superiores a los límites permitidos, aún pueden existir riesgos químicos residuales. Tenga en cuenta las siguientes instrucciones de seguridad: Atención: La parte posterior de la pantalla y la placa de circuito pueden liberar residuos químicos de fabricación o durante el funcionamiento. Nota: Use guantes protectores cuando manipule o instale la pantalla durante

mucho tiempo para evitar irritación de la piel. Precaución: Los componentes electrónicos pueden emitir pequeñas cantidades de compuestos orgánicos volátiles (COV), especialmente si la pantalla es nueva. Nota: asegúrese de trabajar en un área bien ventilada para minimizar la concentración de vapores en el aire. Precaución: No utilice productos químicos ni disolventes agresivos para limpiar la pantalla, ya que pueden dañar la capa protectora o los componentes electrónicos. Nota: Utilice un paño de limpieza antiestático o un limpiador especial para componentes electrónicos para limpiar cuidadosamente la pantalla. Aunque la pantalla cumple con los requisitos de la Directiva RoHS (2011/65/UE) y no contiene sustancias peligrosas en cantidades superiores a los límites permitidos, aún pueden existir riesgos químicos residuales. Tenga en cuenta las siguientes instrucciones de seguridad: Atención: La parte posterior de la pantalla y la placa de circuito pueden liberar residuos químicos de fabricación o durante el funcionamiento. Nota: Use guantes protectores cuando manipule o instale la pantalla durante mucho tiempo para evitar irritación de la piel. Precaución: Los componentes electrónicos pueden emitir pequeñas cantidades de compuestos orgánicos volátiles (COV), especialmente si la pantalla es nueva. Nota: asegúrese de trabajar en un área bien ventilada para minimizar la concentración de vapores en el aire. Precaución: No utilice productos químicos ni disolventes agresivos para limpiar la pantalla, ya que pueden dañar la capa protectora o los componentes electrónicos. Nota: Utilice un paño de limpieza antiestático o un limpiador especial para componentes electrónicos para limpiar cuidadosamente la pantalla. La pantalla contiene componentes electrónicos sensibles y una capa superior. Un manejo inadecuado o una presión excesiva pueden causar daños a la pantalla o lesiones. Observe las siguientes instrucciones de seguridad para evitar riesgos mecánicos: Atención: La tapa del display es frágil y puede romperse si se manipula incorrectamente. Nota: Evite aplicar una presión fuerte o doblar la pantalla. Manipule la pantalla con cuidado y sólo por la placa de circuito para evitar roturas. Precaución: Las caídas o los impactos pueden agrietar la superficie de la pantalla y dañar los componentes electrónicos de la parte posterior. Nota: Evite dejar caer la pantalla y protéjala de impactos. Utilice una superficie suave cuando trabaje para evitar rayones. Atención: Si la pantalla se rompe, los trozos de vidrio afilados pueden provocar lesiones. Nota: Si la pantalla se rompe, manipule los fragmentos con cuidado y use guantes protectores para evitar cortes. Deseche las piezas de vidrio de forma segura. Nota: Una fijación incorrecta puede provocar tensión mecánica y rotura de la pantalla. Acción: Coloque la pantalla de forma segura y sin presión excesiva. Utilice soportes o carcasas adecuados para montar la pantalla de forma estable. Precaución: Los métodos de limpieza inadecuados pueden rayar o dañar la superficie. Nota: Utilice únicamente paños suaves y antiestáticos para limpiar la pantalla. Evite agentes de limpieza agresivos y fricciones fuertes. La pantalla funciona con voltajes y corrientes eléctricas que, si se usan incorrectamente, pueden provocar descargas eléctricas, cortocircuitos o incendios. Tenga en cuenta las siguientes instrucciones de seguridad: Atención: Utilice el producto únicamente con los voltajes especificados. Nota: Los límites de rendimiento del producto se pueden encontrar en la hoja de datos asociada. Nota: Las fuentes de voltaje inadecuadas pueden dañar la pantalla o provocar situaciones peligrosas. Acción: Utilice únicamente fuentes de alimentación o baterías probadas y adecuadas para alimentar sus circuitos. Asegúrese de que la fuente de voltaje cumpla con los requisitos de la pantalla. Precaución: Evite cortocircuitos entre los conectores y componentes del producto. Nota: Asegúrese de que ningún objeto conductor toque o puentee la placa de circuito. Utilice herramientas aisladas y preste atención a la disposición de las conexiones. Precaución: No realice ningún trabajo en el producto cuando esté conectado a una fuente de alimentación. Nota: Desconecte el producto de la alimentación antes de realizar cambios en el circuito o conectar o quitar componentes. Nota: busque señales de daños eléctricos, como humo, olores inusuales o decoloración. Acción: Si ocurren tales signos, apague la alimentación inmediatamente e inspeccione minuciosamente el circuito para detectar errores. La pantalla puede generar calor durante el funcionamiento, lo que podría provocar sobrecalentamiento, quemaduras o incendios si se manipula incorrectamente. Tenga en cuenta las siguientes instrucciones de seguridad: Atención: Algunos componentes de la pantalla pueden calentarse durante el funcionamiento o en caso de error. Medida: Después de apagarla, deje que la pantalla se enfríe lo suficiente antes de tocar directamente los componentes individuales de la parte posterior. Evite el contacto directo con componentes calientes. Precaución: La sobrecarga puede provocar un calentamiento excesivo de los componentes electrónicos. Nota: Asegúrese de que la fuente de alimentación y voltaje cumpla con las especificaciones de la pantalla y no cause sobrecarga.

Índice

Introducción

La pantalla matricial de 64 LED es una pantalla con matrices de LED apiladas en una matriz de 8x8 LED. Hay 8 filas de 8 LED, creando una matriz de 64 LED (*64 píxeles*). Cada LED necesita cables de alimentación y tierra para funcionar. Para evitar cablear los 64 LED individualmente, los LED se conectan para formar una matriz de LED como se muestra en la siguiente imagen:



Al disponer los LED en una matriz, el número de pines de salida se reduce a 16, que siguen siendo demasiados, por lo que en este módulo se utiliza un chip controlador. La pantalla matricial de 64 LEDs utiliza el chip controlador llamado MAX7219. El chip controlador se puede conectar en serie a otros chips controladores en la cadena tipo margarita, como en la *pantalla matriz de 4x64 LED* de AZ-Delivery. Es posible apilar más, pero tendrá que utilizar una fuente de alimentación externa para pantallas adicionales.

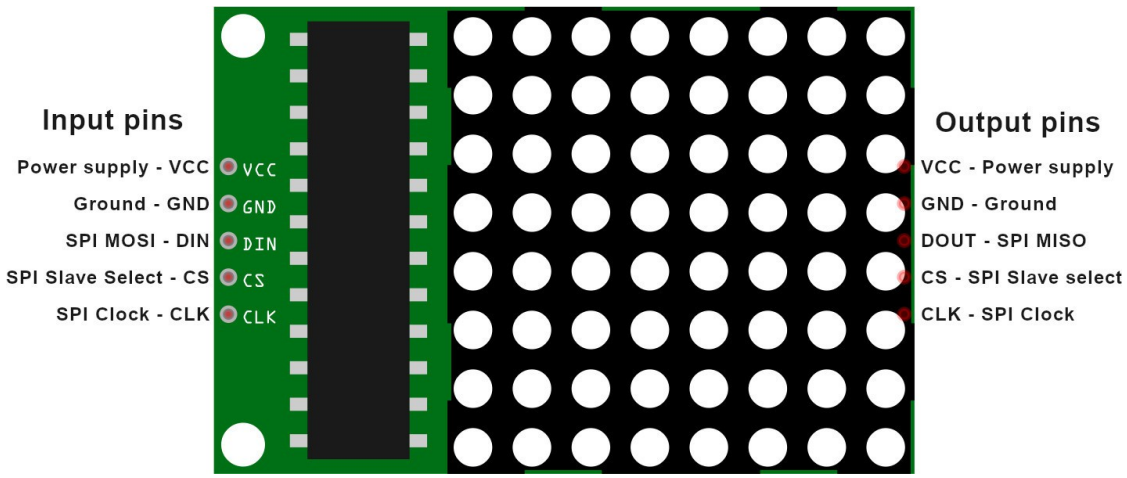
El chip MAX7219 utiliza la interfaz SPI para comunicarse con los microcontroladores. "SPI" significa interfaz de comunicación periférica serie síncrona y se utiliza para la comunicación a corta distancia, principalmente en sistemas embebidos. Sincrónica significa que los datos se envían y reciben dentro de un ciclo de reloj determinado, y serie significa que los datos se envían en serie, bit a bit.

Especificaciones

"	Rango de la tensión de funcionamiento:	+3,3 V a +5 V CC
"	Color de los LED: Control de luminosidad:	Rojo digital y analógico
"	Dimensiones:	32 x 50 x 15 mm [1,3 x 2 x 0,6 pulg].
"	Interfaz SPI de 10 MHz	
"	Pantalla oculta al encender	
"	Control de pantallas LED con cátodo común	

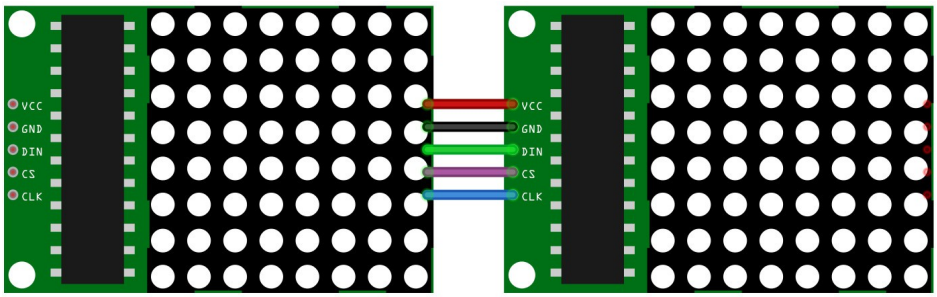
Asignación de clavijas

La pantalla matriz de 64 LEDs tiene cinco pines de entrada y cinco pines de salida. El diagrama de asignación de pines se muestra en la siguiente figura:



El chip controlador utiliza una interfaz SPI, donde: DIN > SPI MOSI - Salida maestro Entrada esclavo CS > SPI SS - Selección de esclavo CLK > Reloj SPI - Señal de reloj

Las pantallas pueden conectarse en serie (apiladas), con el pin de salida de la primera pantalla conectado a los pines de entrada de la segunda, y así sucesivamente. Esto puede verse en la siguiente imagen:



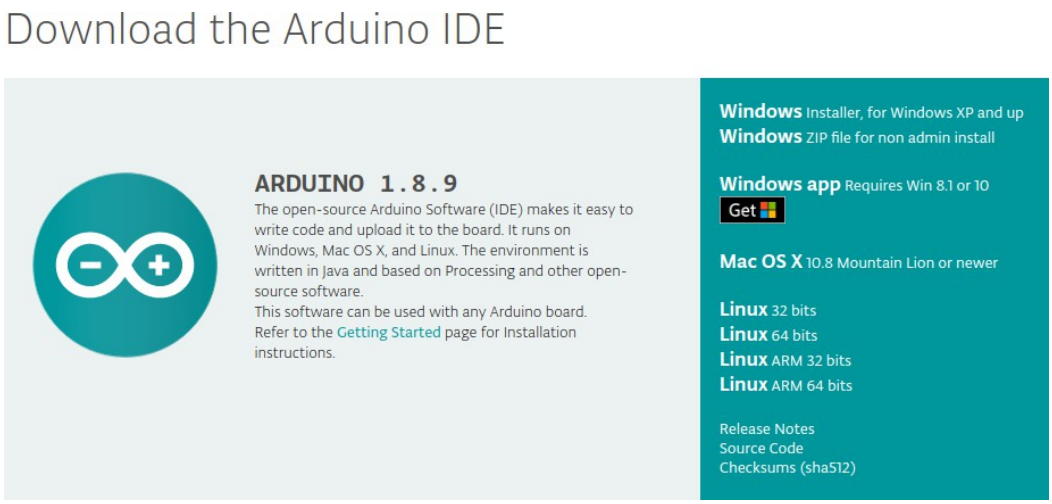
Primera pantalla > Segunda pantalla

VCC	>	VCC
GND	>	GND
DOUT	>	DIN
CS	>	CS
CLK	>	CLK

Después de conectar varias pantallas, el número de dispositivos conectados en el ejemplo de boceto (o ejemplo de script Python) debe actualizarse en consecuencia.

Cómo configurar el IDE de Arduino

Si el IDE de Arduino no está instalado, sigue el *enlace* y descarga el archivo de instalación para el sistema operativo de tu elección.



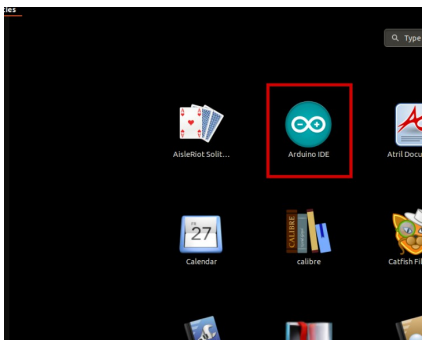
Para usuarios `de Windows`: Haga doble clic en el archivo `.exel` descargado y siga las instrucciones de la ventana de instalación.

Para los usuarios *de Linux*, descarga un archivo con la extensión *.tar.xz* que hay que extraer. Una vez extraído, vaya al directorio extraído y abra el terminal en ese directorio. Dos
.sh, el primero llamado *arduino-linux- setup.sh* y el segundo llamado *install.sh.*

Para ejecutar el primer script en el terminal, abra el terminal en la carpeta extraída y ejecute el siguiente comando:
sh arduino-linux-setup.sh nombre_usuario
nombre_usuario - es el nombre de un superusuario en el sistema operativo Linux. Debe introducirse una contraseña para el superusuario al iniciar el comando. Espere unos minutos a que se complete el script.

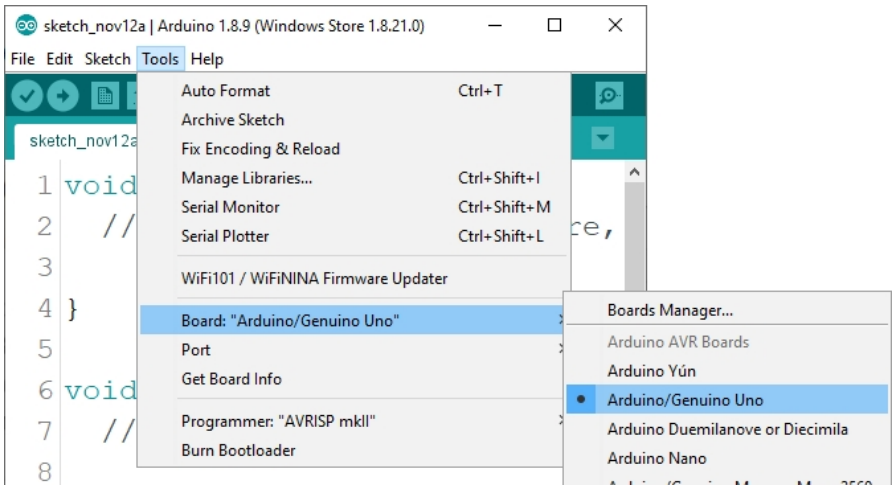
El segundo script con el nombre *install.sh* ***script*** debe ser utilizado después de la instalación del primer script. Ejecute el siguiente comando en el terminal (directorio extraído): **sh install.sh**

Después de instalar estas secuencias de comandos, vaya a *Todas las aplicaciones*, donde se encuentra el icono
Arduino IDE está instalado.



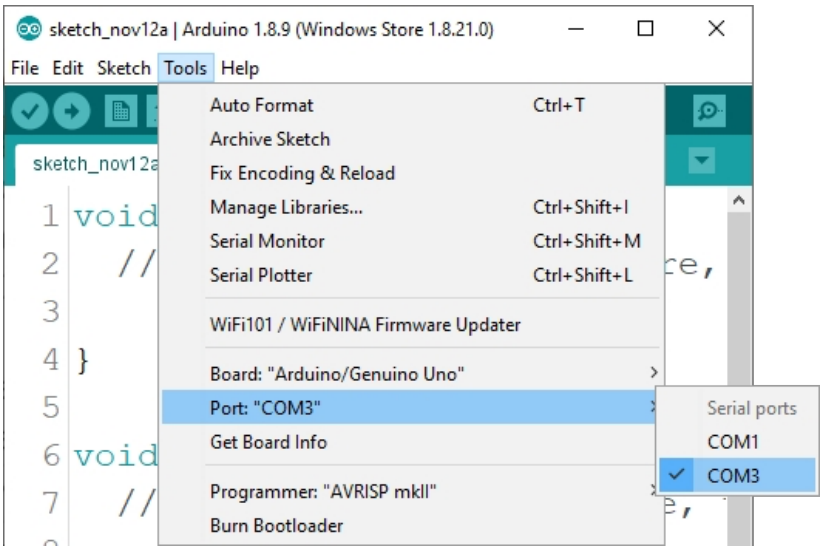
Casi todos los sistemas operativos vienen con un editor de texto preinstalado (por ejemplo, *Windows* con *Notepad*, *Linux* Ubuntu con *Gedit*, *Linux Raspbian* con *Leafpad*, etc.). Todos estos editores de texto son perfectamente adecuados para el propósito del eBook.

Primero comprueba si tu PC puede reconocer una placa microcontroladora. Abra el IDE Arduino recién instalado y vaya a : *Herramientas > Tablero > {nombre de su tablero aquí} {el nombre de tu placa aquí}* debería ser el *Arduino/Genuino Uno,* como se puede ver en la siguiente imagen:



Hay que seleccionar el puerto al que está conectada la placa microcontroladora. Vaya a: *Herramientas > Puerto > {el nombre del puerto va aquí}* y si la placa microcontroladora está conectada al puerto USB, el nombre del puerto se puede ver en el menú desplegable de la imagen anterior.

Si se utiliza el IDE Arduino en Windows, los nombres de los puertos son los siguientes:



Para los usuarios *de Linux,* por ejemplo, el nombre del puerto es */dev/ttyUSBx*, donde *x* es un número entero entre *0* y *9.*

Cómo configurar la Raspberry Pi y Python

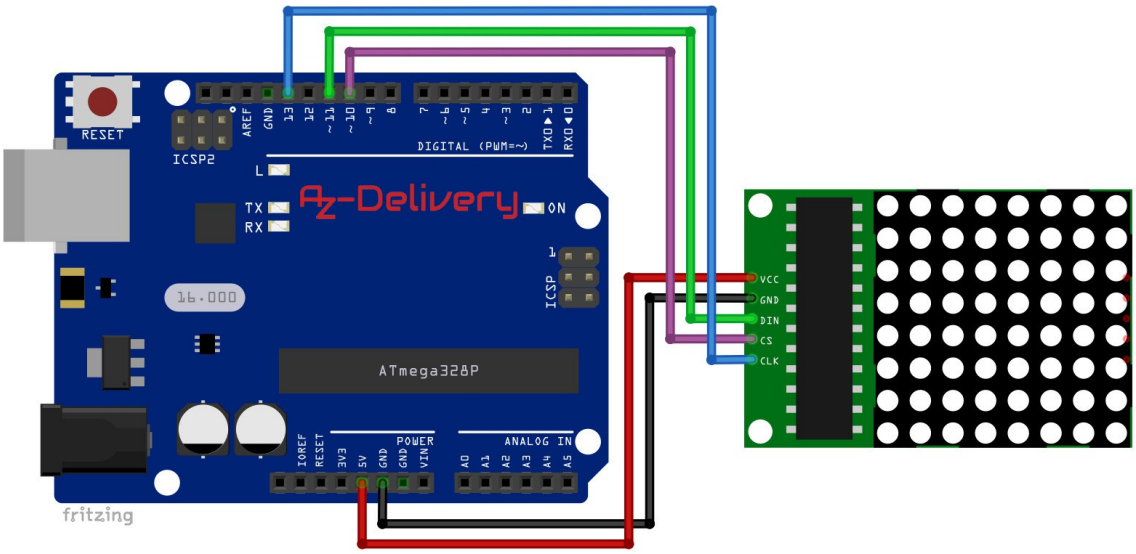
Primero hay que instalar el sistema operativo para la Raspberry Pi y, a continuación, configurar todo para que pueda utilizarse en modo headless. El modo Headless permite una conexión remota a la Raspberry Pi sin necesidad de una pantalla de PC, ratón o teclado. Las únicas cosas que se utilizan en este modo son la propia Raspberry Pi, la fuente de alimentación y la conexión a Internet. Todo esto se explica en detalle en el eBook gratuito:

Guía de inicio rápido de Raspberry Pi

El sistema operativo Raspbian se suministra con *Python* preinstalado.

Conexión de la pantalla a la placa Atmega328P

Conecta la pantalla a la placa del microcontrolador como se muestra a continuación:



Clavija del módulo			**>Board Pin**	
VCC		>	5V	**Cable rojo**
GND		>	GND	**Cable negro**
DIN	(MOSI)	>	D11 Pin	**Cable verde**
CS	(SS)		>D10 Pin	**Cable morado**
CLK	(SCK)		>D13 Pin	**Cable azul**

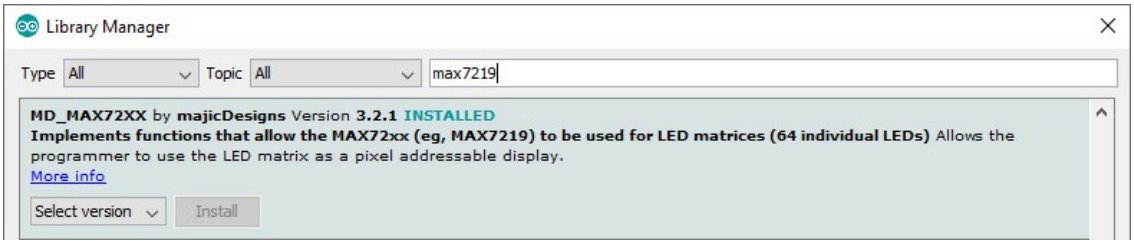
NOTA: Dado que existen diferentes versiones del módulo, asegúrese de leer las designaciones de las patillas antes de conectar el módulo. Las designaciones alternativas de las patillas se indican entre paréntesis.

Biblioteca para el IDE Arduino

Para utilizar la pantalla con la placa Atmega328P, se recomienda descargar la librería externa para la misma. La librería utilizada en este eBook se llama *MD_MAX72XX*. Para instalar la biblioteca, abra el IDE de Arduino y vaya a:

Herramientas > Gestionar bibliotecas

y cuando aparezca una nueva ventana, introduce *MAX72XX en* el campo de búsqueda. Instala la librería *MD_MAX72XX* de *majicDesigns como se muestra en la* siguiente imagen:



Existen muchas versiones de estas pantallas. Para determinar cuál se está utilizando, debe ejecutarse el HardwareMapper Sketch suministrado con la librería. Para abrir el sketch, vaya a :

Archivo > Ejemplos > MD_MAX72XX > MD_MAX72xx_HW_Mapper

y cambie la siguiente línea de código:

```
#define VELOCIDAD_SERIAL 57600
```

en: `#define SERIAL_SPEED 9600`

Carga el sketch en el Atmega 328P y abre el Serial Monitor *(Herramientas > Serial Monitor)*. A continuación, sigue las instrucciones de tres pasos que la placa microcontroladora envía al Serial Monitor, como se muestra a continuación:

```
[MD_MAX72xx Hardware mapping utility]

INTRODUCTION
------------
How the LED matrix is wired is important for the MD_MAX72xx library as different
LED modules are wired differently. The library can accommodate these, but it
needs to know what transformations need to be carried out to map your board to the
standard coordinate system. This utility shows you how the matrix is wired so that
you can set the correct *_HW module type for your application.

The standard functions in the library expect that:
o COLUMNS are addressed through the SEGMENT selection lines, and
o ROWS are addressed through the DIGIT selection lines.

The DISPLAY always has its origin in the top right corner of a display:
o LED matrix module numbers increase from right to left,
o Column numbers (ie, the hardware segment numbers) increase from right to left (0..7), and
o Row numbers (ie, the hardware digit numbers) increase down (0..7).

There are three hardware setting that describe your hardware configuration:
- HW_DIG_ROWS - HardWare DIGits are ROWS. This will be 1 if the digits map to the rows
                of the matrix, 0 otherwise
- HW_REV_COLS - HardWare REVerse COLumnS. The normal column coordinates orientation
                is col 0 on the right side of the display. This will be 1 if reversed.
                (ie, hardware 0 is on the left).
- HW_REV_ROWS - HardWare REVerse ROWS. The normal row coordinates orientation is row
                0 at top of the display. This will be 1 if reversed (ie, row 0
                is at the bottom).

These individual setting then determine the model type of the hardware you are using.

INSTRUCTIONS
------------
1. Wire up one matrix only, or cover up the other modules, to avoid confusion.
2. Enter the answers to the question in the edit field at the top of Serial Monitor.

=====================================================

STEP 1 - DIGITS MAPPING (rows)
------------------------------
In this step you will see a line moving across the LED matrix.
You need to observe whether the bar is scanning ROWS or COLUMNS,
and the direction it is moving.
>> Enter Y when you are ready to start: Y
Dig-0-1-2-3-4-5-6-7
>> Enter Y if you saw ROWS animated, N if you saw COLUMNS animated: Y
>> Enter Y if you saw the line moving BOTTOM to TOP, or enter N otherwise: N

STEP 2 - SEGMENT MAPPING (columns)
----------------------------------
In this step you will see a dot moving along one edge of the LED matrix.
You need to observe the direction it is moving.
>> Enter Y when you are ready to start: Y
Seg-G-F-E-D-C-B-A-DP
>> Enter Y if you saw the LED moving LEFT to RIGHT, or enter N otherwise: N

STEP 3 - RESULTS
----------------
Your responses produce these hardware parameters

HW_DIG_ROWS     1
HW_REV_COLS     0
HW_REV_ROWS     0

Your hardware matches the setting for FC-16 modules. Please set FC16_HW.
```

Para obtener el resultado correcto, alinee la pantalla como se indica en el esquema de conexiones. El lado izquierdo de la pantalla es el lado en el que se encuentran los pines de entrada de la pantalla.

La asignación de hardware resultante es *FC16_HW. Por* lo tanto, si utiliza el módulo de pantalla matriz AZ- Delivery 64 LED en bocetos de la biblioteca, establezca la macro de *HARDWARE_TYPE* en este valor de resultado.

Ejemplo de croquis

```
#include <MD_MAX72xx.h >
#define HARDWARE_TYPE MD_MAX72XX::FC16_HW
#define MAX_DEVICES 1
#define CLK_PIN13     // o SCK
#define DATA_PIN11    // o MOSI
#define CS_PIN10      // o SS
#define               DELAYTIME100 // en milisegundos
MD_MAX72XX mx = MD_MAX72XX(HARDWARE_TYPE, CS_PIN, MAX_DEVICES);
void scrollText(char *p) {
  uint8_t charWidth;
  uint8_t cBuf[8];
  mx.clear();
  while (*p != '\0') {
    charWidth = mx.getChar(*p++, sizeof(cBuf) / sizeof(cBuf[0]), cBuf);
    for (uint8_t i=0; i<=charWidth; i++) {
      mx.transform(MD_MAX72XX::TSL);
      if (i < charWidth) {
        mx.setColumn(0, cBuf[i]);
      }
      delay(DELAYTIME);
    }
  }
}
void filas() {
  mx.clear();
  for (uint8_t row=0; row<ROW_SIZE; row++)
    { mx.setRow(row, 0xff);
    delay(2*DELAYTIME);
    mx.setRow(fila, 0x00);
  }
}
```

```
void columnas() {
  mx.clear();
  for (uint8_t col=0; col<mx.getColumnCount(); col++) {
    mx.setColumn(col, 0xff);
    delay(DELAYTIME/MAX_DEVICES);
    mx.setColumn(col, 0x00);
  }
}
void intensity() {
  uint8_t row;
  mx.clear();
  for (int8_t i=0; i<=MAX_INTENSITY; i++)
    { mx.control(MD_MAX72XX::INTENSITY,
    i); mx.setRow(0, 0xff);
    delay(DELAYTIME*10);
  }
  mx.control(MD_MAX72XX::INTENSIDAD, 8);
}
```

```
void transformación() {
  uint8_t flecha[COL_SIZE] =
  {
    0b00001000,
    0b00011100,
    0b00111110,
    0b01111111,
    0b00011100,
    0b00011100,
    0b00111110,
    0b00000000 };
  MD_MAX72XX::transformType_t t[] = {
    MD_MAX72XX::TSL, MD_MAX72XX::TSL, MD_MAX72XX::TSL, MD_MAX72XX::TSL,
    MD_MAX72XX::TSL, MD_MAX72XX::TSL, MD_MAX72XX::TSL, MD_MAX72XX::TSL,
    MD_MAX72XX::TSL, MD_MAX72XX::TSL, MD_MAX72XX::TSL, MD_MAX72XX::TSL,
    MD_MAX72XX::TSL, MD_MAX72XX::TSL, MD_MAX72XX::TSL, MD_MAX72XX::TSL,
    MD_MAX72XX::TFLR,
    MD_MAX72XX::TSR, MD_MAX72XX::TSR, MD_MAX72XX::TSR, MD_MAX72XX::TSR,
    MD_MAX72XX::TSR, MD_MAX72XX::TSR, MD_MAX72XX::TSR, MD_MAX72XX::TSR,
    MD_MAX72XX::TSR, MD_MAX72XX::TSR, MD_MAX72XX::TSR, MD_MAX72XX::TSR,
    MD_MAX72XX::TSR, MD_MAX72XX::TSR, MD_MAX72XX::TSR, MD_MAX72XX::TSR,
    MD_MAX72XX::TRC,
    MD_MAX72XX::TSD, MD_MAX72XX::TSD, MD_MAX72XX::TSD, MD_MAX72XX::TSD,
    MD_MAX72XX::TSD, MD_MAX72XX::TSD, MD_MAX72XX::TSD, MD_MAX72XX::TSD,
    MD_MAX72XX::TFUD,
    MD_MAX72XX::TSU, MD_MAX72XX::TSU, MD_MAX72XX::TSU, MD_MAX72XX::TSU,
    MD_MAX72XX::TSU, MD_MAX72XX::TSU, MD_MAX72XX::TSU, MD_MAX72XX::TSU,
    MD_MAX72XX::TINV,
    MD_MAX72XX::TRC, MD_MAX72XX::TRC, MD_MAX72XX::TRC, MD_MAX72XX::TRC,
    MD_MAX72XX::TINV };
  mx.clear();
  mx.control(MD_MAX72XX::UPDATE, MD_MAX72XX::OFF);
  for (uint8_t j=0; j<mx.getDeviceCount(); j++) {
    mx.setBuffer(((j+1)*COL_SIZE)-1, COL_SIZE, arrow);
  }
```

```
  // una pestaña
  mx.control(MD_MAX72XX::UPDATE, MD_MAX72XX::ON);
  delay(DELAYTIME);
  mx.control(MD_MAX72XX::WRAPAROUND, MD_MAX72XX::ON);

  for (uint8_t i=0; i<(sizeof(t)/sizeof(t[0])); i++) {
    mx.transform(t[i]);
    delay(DELAYTIME*4);
  }
  mx.control(MD_MAX72XX::WRAPAROUND, MD_MAX72XX::OFF);
}

void showCharset() {
  mx.clear();
  mx.update(MD_MAX72XX::OFF);
  for (uint16_t i=0; i<256; i++) {
    mx.clear(0);
    mx.setChar(COL_SIZE-1, i);
    if (MAX_DEVICES >= 3) {
      char hex[3];
      sprintf(hex, "%02X", i);
      mx.clear(1);
      mx.setChar((2*COL_SIZE)-1, hex[1]);
      mx.clear(2);
      mx.setChar((3*COL_SIZE)-1, hex[0]);
    }
    mx.update();
    delay(DELAYTIME*2);
  }
  mx.update(MD_MAX72XX::ON);
}
```

```
void setup() {
  mx.begin();
}

void loop() {
  scrollText("Gráficos");
  rows();
  columnas();
  intensidad();
  transformación();
  showCharset();
  delay(3000);
}
```

El sketch comienza con la inclusión de la librería *MD_MAX72XX* y continúa con varias definiciones de macros.

A continuación, especifique qué pantalla se utiliza estableciendo la macro `HARDWARE_TYPE` en el valor `FC16_HW` (el valor resultante del `sketch HardwareMapper`).

La macro `MAX_DEVICES` define entonces cuántas pantallas están conectadas en la cadena. En este sketch, se establece en el valor "*1"* (porque sólo se utiliza una pantalla). Si se conectan varias pantallas de matriz de LED en una cadena, cambie este valor en consecuencia.

Hay tres macros más que definen los pines de la interfaz SPI.

A continuación se crea el *objeto mx*, que representa el objeto pantalla que se utiliza en todo el sketch.

A continuación, se crean varias funciones.

La primera función se llama `scrollingText()`, tiene un argumento y no devuelve ningún valor. La función se utiliza para desplazar texto por la pantalla. El valor del argumento es el texto de desplazamiento, un valor de cadena.

La función *scrollingText()* utiliza la función de la biblioteca MD_MAX72XX con el nombre *setColumn()*. La función set *Column()* enciende y apaga una columna de LEDs en la pantalla. El resto de la función scrollingText() es el algoritmo para el efecto de desplazamiento de texto. La función acepta la cadena p y la convierte en una matriz de enteros sin signo que representan los caracteres de la cadena de texto. A continuación, esta matriz se muestra en la pantalla. Al final de la función, se establece el tiempo de retardo, que corresponde a la velocidad del texto que se desplaza.

La segunda función se llama *rows()*, que no tiene argumentos y no devuelve ningún valor. La función utiliza una función *setRow()* de la biblioteca *MD_MAX72XX*. La función setRow() tiene dos argumentos. El primer argumento representa el número de fila, donde *0* significa la primera fila empezando por arriba. El segundo argumento es un entero en forma hexadecimal, que representa un *valor de 8 bits* para una matriz de píxeles (*8* píxeles en una fila). En este caso, el valor *0xFF* (*0b1111 1111*) se utiliza para *encender* todos los píxeles de una línea en particular. Por ejemplo, para encender sólo los dos primeros LEDs de la fila, utilice el valor *0x03.*

La tercera función se llama *columns()*, que no acepta argumentos y no devuelve ningún valor. Es casi igual que la función *rows()*, pero en este caso la matriz de columnas de píxeles se pone en *ON*. La función utiliza una función *setColumn()* de la biblioteca *MD_MAX72XX*. La función setColumn()

es igual que la función `setRow()`, la diferencia es que la función `setColumn()` controla las columnas.

La cuarta función se llama `intensity()`, que no tiene argumentos y no devuelve ningún valor. Se utiliza para *encender* sólo la primera fila de LEDs en la pantalla. La función utiliza la función control() de la biblioteca *MD_MAX72XX*. La función control() tiene dos argumentos, el primero de los cuales es una propiedad del objeto mx. En este caso, se controla la propiedad con el nombre `MD_MAX72XX::INTENSITY`, es decir, el nombre de la función de `intensidad`. El segundo argumento es un número entero que representa el grado de intensidad y cuyo valor oscila entre *0* y *15*. El valor por defecto es *8*.

La quinta función se llama `transformation()`, que no tiene argumentos y no devuelve ningún valor. Se utiliza para animar imágenes de mapa de bits en la pantalla. Los datos de la imagen bitmap se almacenan en una matriz de enteros sin signo. Esta matriz contiene *8* elementos, y cada elemento contiene un número binario de 8 dígitos. Estos números binarios representan las líneas de la pantalla, donde el valor binario *0 representa* un LED apagado y el valor *1* representa un LED *encendido.*

A continuación, se crea otro array con el nombre t, que contiene las enumeraciones utilizadas para las animaciones. El formato de la enumeración es *MD_MAX72XX::{enumeración}*, donde la enumeración tiene uno de los siguientes valores:

TSL - Desplazar un píxel a la izquierda, *TSR*
- Desplazar un píxel a la derecha, *TSU*
- Subir un píxel, *TSD* - Bajar un píxel,

TFLR - Voltear de izquierda a derecha, *TFUD*
- voltear de arriba abajo,

TRC - Gire 90 grados en el sentido de las agujas del reloj,

TINV - Invierte los píxeles (los píxeles en *estado* ON pasan a estado *OFF* y viceversa).

A continuación se utiliza la función *setBuffer()*, que tiene tres argumentos y no devuelve ningún valor. Se utiliza para establecer el búfer de datos del chip controlador MAX7219. El primer y segundo argumento son la posición *X* e *Y* de la imagen. El *valor X comienza en la esquina superior izquierda de la pantalla* y termina en la esquina superior derecha de la pantalla; el *valor Y* comienza en la esquina superior izquierda de la pantalla y termina en la esquina inferior izquierda de la pantalla. El tercer argumento representa los datos de la imagen de mapa de bits.

Para mostrar los datos, primero hay que desactivar el refresco de pantalla, luego establecer el almacén de datos y después activar el refresco de pantalla. Esto se puede ver en las siguientes líneas de código:

```
mx.control(MD_MAX72XX::UPDATE, MD_MAX72XX::OFF);
mx.control(MD_MAX72XX::WRAPAROUND, MD_MAX72XX::ON);
for (uint8_t i=0; i<(sizeof(t)/sizeof(t[0])); i++) {
  mx.transform(t[i]); delay(DELAYTIME*4);
}
mx.control(MD_MAX72XX::UPDATE, MD_MAX72XX::ON);
```

A continuación, se animan los datos visualizados. Para ello se utiliza la función `control()`, la propiedad *WRAPAROUND* del *objeto* mx y la función `transform()`. El bucle for se utiliza para recorrer todas las enumeraciones de la matriz t.

La última función se llama `showCharsset()`, que no tiene argumento y no devuelve ningún valor. Se utiliza para mostrar todos los caracteres UNICODE que se pueden utilizar con la pantalla de matriz LED. Aquí se `utiliza` la función `update()` de la biblioteca `MD_MAX72XX`, que se comporta como la función control() con la propiedad `UPDATE`. La función update() tiene un argumento cuyo valor puede ser uno de los dos valores:

`MD_MAX72XX::OFF` y

`MD_MAX72XX::ON`

En primer lugar, el valor del argumento de la función *update() se establece* en *MD_MAX72XX::OFF*, lo que desactiva las actualizaciones de la memoria de datos. A continuación, se realizan y guardan los cambios en la memoria de datos; el carácter visualizado se modifica realmente. Y entonces el valor del argumento de la función *update() se establece en* el valor *MD_MAX72XX::ON*. El bucle for se utiliza para recorrer todos los caracteres UNICODE (*256* caracteres).

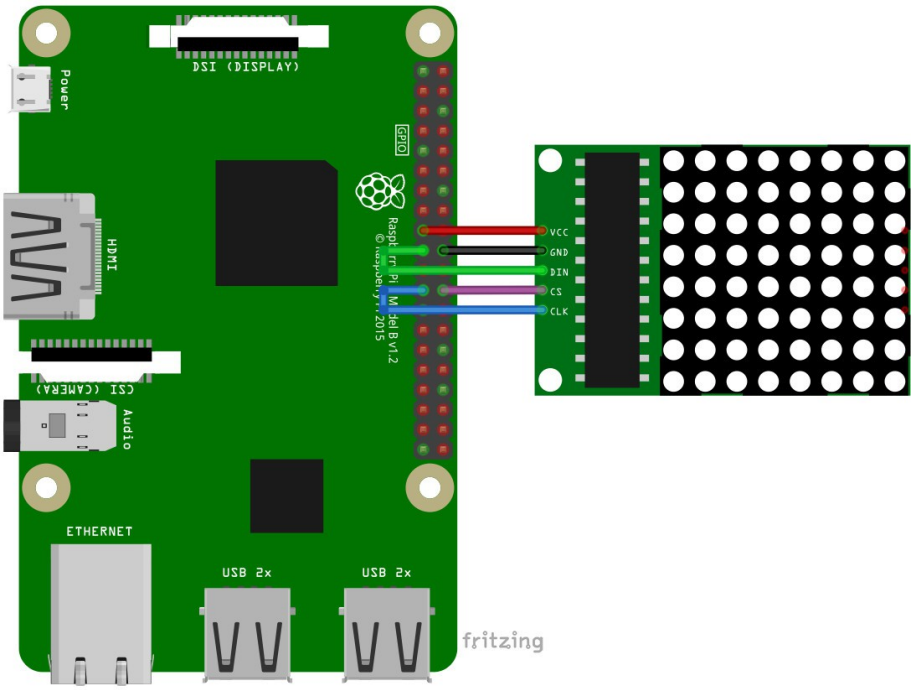
En la función *setup(),* el objeto pantalla se inicializa con la siguiente línea de código: `mx.begin()`

En la función *loop(),* todas las funciones generadas se ejecutan una tras otra.

Para obtener más información sobre estas funciones de la biblioteca *MD_MAX72XX, visite* la documentación oficial del *sitio de* la biblioteca.

Conexión de la pantalla a la Raspberry Pi

Conecta la pantalla a la Raspberry Pi como se muestra a continuación:



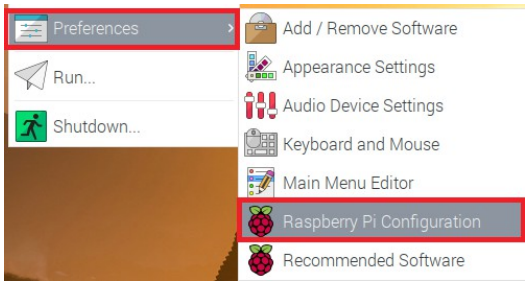
Clavija del módulo		>Pin de	Frambuesa Pi		
VCC		>	3V		**Cable rojo**
GND		>	GND		**Cable negro**
DIN	(MOSI)	>	GPIO10	[pin 19]	**Cable verde**
CS	(SS)	>	GPIO8	[pin 24]	**Cable morado**
CLK	(SCK)	>	GPIO11	[pin 23]	**Cable azul**

Activación de la interfaz SPI

Para utilizar inicialmente la pantalla con una Raspberry Pi, la interfaz SPI debe estar activada en Raspbian. Para ello, vaya a :

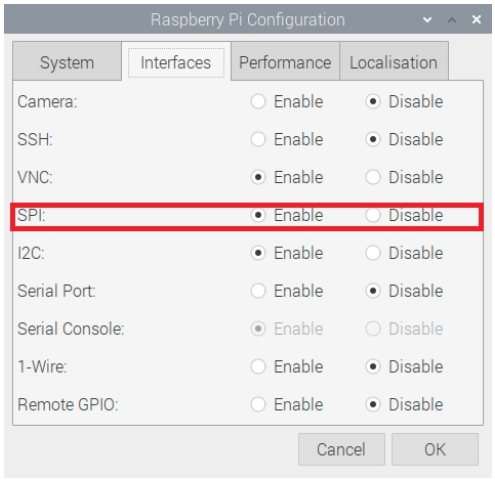
Menú Aplicaciones > Preferencias > Configuración de Raspberry Pi

como se muestra en la siguiente imagen:



A continuación, abra la pestaña *Interfaces*, configure el *SPI*

y actívelo como se muestra en la siguiente imagen:



Bibliotecas y herramientas para Python

Para utilizar el módulo con la Raspberry Pi, se recomienda descargar una biblioteca externa para ello. La biblioteca utilizada en este eBook se llama *luma.led_matrix*. Para utilizar la biblioteca, es necesario instalar las dependencias. Para ello, abra el terminal y ejecute el siguiente comando:

```
sudo apt install build-essential python-dev python-pip libfreetype6-dev libjpeg-dev libopenjp2-7 libtiff5
```

A continuación, instale la última versión de la *luma.*led_matrix-Library ejecutando el siguiente comando:

```
sudo -H pip3 install --upgrade luma.led_matrix
```

Script Python S

```
tiempo de importación
from luma.led_matrix.device import max7219
from luma.core.interface.serial import spi, noop
from luma.core.render import canvas
from luma.core.virtual import viewport
from luma.core.legacy import text, show_message
de      luma.core.legacy.font    importar  proporcional,    CP437_FONT,
TINY_FONT, SINCLAIR_FONT, LCD_FONT

def demo(n, block_orientation, rotate, inreverse):
    serial = spi(port=0, device=0, gpio=noop())
    device = max7219(serial, cascaded=n or 1,
        block_orientation=orientación_del_bloque, rotate=rotar o 0,
__    bloques_ordenados_en_orden_inverso=inverso)

    print('Dispositivo creado')

    msg = 'MAX7219 LED Matrix Demo'
    print(msg)
    show_message(device, msg, fill='white',
__    font=proportional(CP437_FONT), scroll_delay=0.05)
    time.sleep(2)
```

```
msg = 'Encender toda la
pantalla' print(msg)
show_message(device, msg, fill='white',
    font=proportional(CP437_FONT), scroll_delay=0.05)
con canvas(device) como draw:
    para i en rango(anchura.dispositivo):
        for j in range(dispositivo.altura):
            draw.point((i, j), fill='blanco')

time.sleep(1)

msg = 'Brillo'
print(msg)
show_message(device, msg, fill='white')
time.sleep(1)
para _ en rango(5):
    para intensidad en rango(16):
        con canvas(device) como draw:
            para i en rango(anchura.dispositivo):
                for j in range(dispositivo.altura):
                    draw.point((i, j), fill='blanco')
        device.contrast(intensidad * 16)
        time.sleep(0.2)

device.contrast(0x80)
time.sleep(1)
```

```
msg = 'Fuente alternativa'
print(msg)
show_message(device, msg, fill='white', font=SINCLAIR_FONT,
  scroll_delay=0.05)
time.sleep(1)

msg = 'Fuente proporcional - ¡los caracteres están
apretados!' print(msg)
show_message(device, msg, fill='white',
    font=proportional(SINCLAIR_FONT), scroll_delay=0.05)
time.sleep(1)

msg = 'Tiny font - la fuente más pequeña para pantalla de matriz
LED' print(msg)
show_message(device, msg, fill='white',
  font=proportional(TINY_FONT), scroll_delay=0.05)
time.sleep(1)

msg = 'Caracteres CP437'
print(msg)
show_message(device, msg)
time.sleep(1)
para x en rango(256):
    con canvas(device) como draw:
        text(draw, (0, 0), chr(x), fill='white')
        time.sleep(0.5)
```

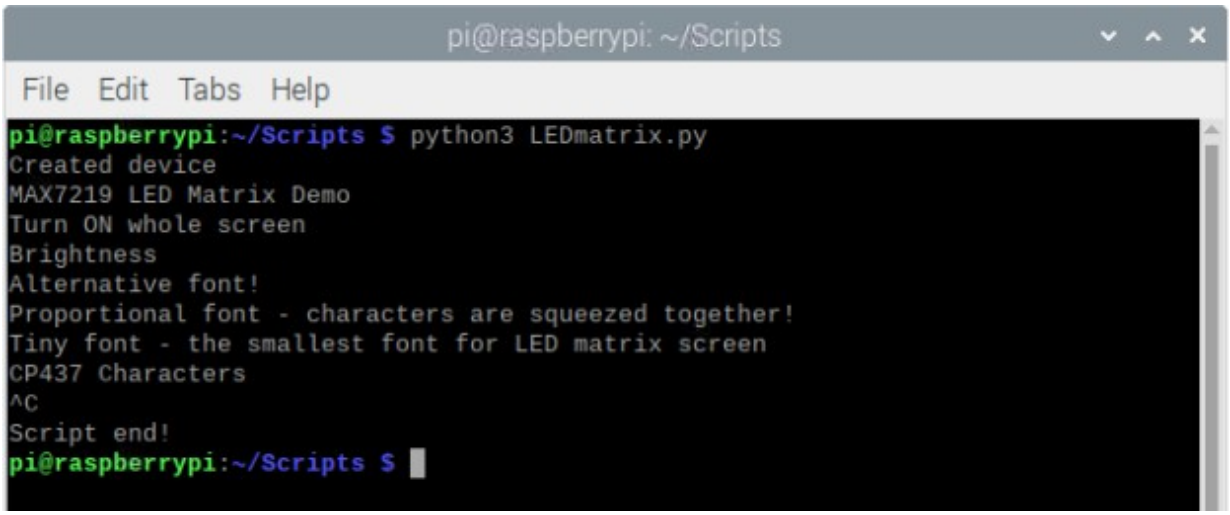
```
print('[¡Pulsa CTRL + C para finalizar el
script!]') try:
    mientras sea verdad:
        demo(1, 0, 3, Falso)
        # el primer argumento: número de pantallas en cascada
        # el segundo argumento: orientación del bloque -> 0,
        90, -90 # el tercer argumento: rotar -> 0=0, 1=90,
        2=180, 3=270 # el cuarto argumento: orden inverso ->
        Verdadero/Falso time.sleep(1)

except KeyboardInterrupt:
    print('\n¡Fin del
    script!')
```

El código se basa en el código del siguiente *enlace*.

Guarde el script con el nombre *LEDmatrix.py*. Para ejecutar el script, abra el terminal en el directorio en el que está guardado el script y ejecute el siguiente comando:

python3 ledMatrix.py

El resultado debería parecerse al de la imagen siguiente:



Para finalizar la exploración, pulse CTRL + C en el teclado.

El script comienza con la importación de las librerías necesarias.

A continuación, se crea la función *demo()*, que tiene cinco argumentos y no devuelve ningún valor. Se utiliza para ejecutar todo el código de ejemplo que contiene funciones de la *librería luma.led_matrix.* Los argumentos son:

"n - representa el número de pantallas matriciales LED en cascada

" *block_orientation* - se utiliza para corregir la orientación de una sola pantalla, los valores son: *0, 90* y *-90*; el valor por defecto es *0*

" *rotar* - que rota el contenido en la pantalla; sus valores son

0, 1, 2 y 3, que representan los ángulos *0°, 90°, 180° y 270°* respectivamente; el ajuste por defecto es *1*

reverse" - es un valor booleano, y si se *establece en True*, el orden de las pantallas se invierte; el valor por defecto es *False*

El objeto de comunicación SPI llamado *serial* y el objeto dispositivo llamado *device* se crean dentro de la función *demo().* El *objeto serial* se utiliza para crear el objeto *dispositivo.*

Para mostrar un mensaje desplazable (horizontalmente) en la pantalla, se utiliza la función *show_message()* de la *biblioteca luma.*led_matrix. La función tiene cinco argumentos y no devuelve ningún valor. Los tres últimos argumentos son opcionales. El primer argumento es un objeto *dispositivo*; el segundo es el mensaje; el tercero es el argumento *relleno*; el cuarto es el argumento *fuente*; y el quinto es el *argumento retardo_desplazamiento*.

El valor del argumento *fill* es el nombre del color, por ejemplo colores como: *blanco, azul, rojo*, etc. Si se utiliza un color distinto del negro, el mensaje se muestra en rojo, ya que el color rojo es el color de los LED de la pantalla. Si se almacena el valor *negro* en el argumento *Relleno, los LED de la pantalla se* apagan.

El argumento *fuente se* utiliza para establecer la fuente específica del texto. El argumento *scroll_delay se* utiliza para cambiar la velocidad de desplazamiento del texto.

La función *contrast()* se utiliza para cambiar el nivel de brillo, como en la siguiente línea de código: `device.contrast(number)`
donde el número es el nivel de brillo. Hay *16* valores diferentes para el nivel de brillo, pero este valor debe ser un número que, dividido por *16*, dé como resultado un número entero, ya que los registros internos del chip controlador MAX7219 se configuran de esta manera.

Hay dos formas de utilizar fuentes. La primera es almacenar el nombre de la fuente en el argumento fuente de la función *show_message(). La segunda es* utilizar la función proportional() para establecer el valor del argumento font. La función *proportional()* tiene un argumento cuyo valor es el nombre de la fuente. La diferencia entre estas dos formas de utilizar las fuentes es que la función *proportional()* muestra las fuentes proporcionalmente al tamaño de la pantalla.

Para encender o apagar ciertos LEDs de la pantalla, utilice la función *point().* La función debe utilizarse con el objeto de dibujo de la *biblioteca luma.core.render.canvas.* La función point() tiene dos argumentos y no devuelve ningún valor. El primer argumento es una *tupla de* dos elementos, donde los elementos son la *posición X* e *Y* del LED específico de la pantalla. El segundo argumento es el argumento de *relleno*.

Para encender todos los LEDs de la pantalla, utilice las siguientes líneas de código:

```
con canvas(device) como draw:
    para i en rango(anchura.dispositivo):
        for j in range(dispositivo.altura):
            draw.point((i, j), fill='blanco')
```

Para mostrar un carácter específico en la pantalla, utilice la función *text()*. La función tiene cuatro argumentos y no devuelve ningún valor. El primer argumento es el objeto carácter de la *librería luma.core.render.canvas*. El segundo argumento es la *tupla de* dos elementos, donde los elementos son la *posición X* e *Y* de la esquina superior izquierda del carácter. El tercer argumento es el carácter que se mostrará. El cuarto argumento es el *argumento de relleno*.

Después de la función *demo(), se* crea el bloque de código *try-except*. Para pausar el script: *CTRL + C* en el teclado. Esto se denomina pausa de teclado. Cuando se produce la pausa de teclado, se ejecuta el bloque de código "try-except" y se muestra en el terminal el mensaje *¡Fin de script!*

El bucle indeterminado *(while True:) se* crea en el bloque de código *try*. Dentro del bucle indeterminado se ejecuta la función *demo()* con la siguiente línea de código: `demo(1, 0, 3, False)`

Los argumentos utilizados son:
" *1* es el número de imágenes en cascada, en este caso sólo una imagen,
"0 es la orientación del bloque,
"3 es la rotación del contenido en la pantalla, donde *3 = 270°*, y
" La secuencia normal de pantallas en cascada es *incorrecta*.

Lo ha conseguido. Ya puede utilizar nuestro módulo para sus proyectos.

¡Ahora es tu turno! Desarrolla tus propios proyectos e instalaciones domésticas inteligentes. Te mostraremos cómo hacerlo de forma sencilla y comprensible en nuestro blog. Allí te ofrecemos scripts de ejemplo y tutoriales con pequeños proyectos interesantes para iniciarte rápidamente en el mundo de la microelectrónica. Internet también te ofrece innumerables oportunidades para aprender más sobre microelectrónica.

Si busca otros productos microelectrónicos y accesorios de alta calidad, AZ-Delivery Vertriebs GmbH es el lugar adecuado para usted. Le ofrecemos numerosos ejemplos de aplicación, instrucciones de instalación detalladas, libros electrónicos, bibliotecas y, por supuesto, el apoyo de nuestros expertos técnicos.

https://az-delivery.de

¡Diviértete!

Pie de imprenta

https://az-delivery.de/pages/about-us