

ABX00047

PLACA PORTADORA ARDUINO MKR IOT



Descripción:

Placa portadora MKR iot Arduino con sensores y pantalla integrados.

Nota: Esta placa es pasiva, necesita una placa MKR para funcionar.

Detalles del Producto:

Ranuras de conector

- Interfaz sencilla con una amplia gama de módulos y sensores de ranura
- 2x sensores analógicos de salida
- 1x interfaz I2C

ST HTS221 Sensor de humedad

- Detección capacitiva
- 0-100% rango de detección de humedad
- Precisión de humedad: $\pm 3.5\%$ rH, 20 a +80 rH
- Precisión de temperatura: ± 0.5 °C, 15 a +40 °C
- Interfaz I2C para placa Arduino MKR
- Bajo consumo de energía (2 μ A a 1 Hz tasa de muestreo)

Sensor IMU de 6 ejes ST LSM6DSOX

- Acelerómetro de 3 ejes
- 2/ ± 4 / ± 8 / ± 16 g escala completa
- Giroscopio de 3 ejes
- 125/ ± 250 / ± 500 / ± 1000 / ± 2000 dps(grados por segundo)
- Interfaz I2C para placa Arduino MKR
- Bajo consumo de energía (0.55mA máx.)

Broadcom APDS-9660 RGB y sensor de gestos

- Luz ambiental y color RGB
- Detección de proximidad
- Detección de gestos
- Filtro de bloqueo UV / IR
- Interfaz I2C para placa Arduino MKR
- 5x botones capacitivos qTouch

Pantalla TFT redondeada de 1,3 "

- Resolución 240 x 240
- 36 x 40 mm

ST LPS22HBTR sensor de presión

- Rango de presión absoluta de 260 a 1260 hPa (0,25 a 1,24 atm)
- Interfaz I2C para placa Arduino MKR

Memoria

- Ranura para tarjetas SD

Energía

- Soporte de batería LI-ION 18650 3.7 v
- Carga de batería USB a través de la placa MKR (Funciona hasta 48 h con un 3.7v 2500 mAh)

Relays

- 2x relays KEMET EE2-5NU-L
- Voltaje de bobina de 5 V
- Corriente 2A
- 220 V CC, 250 V CA
- Sin enclavamiento
- Común, normalmente abierto y contactos normalmente cerrados
- Indicador de estado LED

I/O

- Todos los sensores cuentan con función de despertador
- 2x conectores analógicos Grove
- 1x conector I2C Grove
- 2x conector de relay

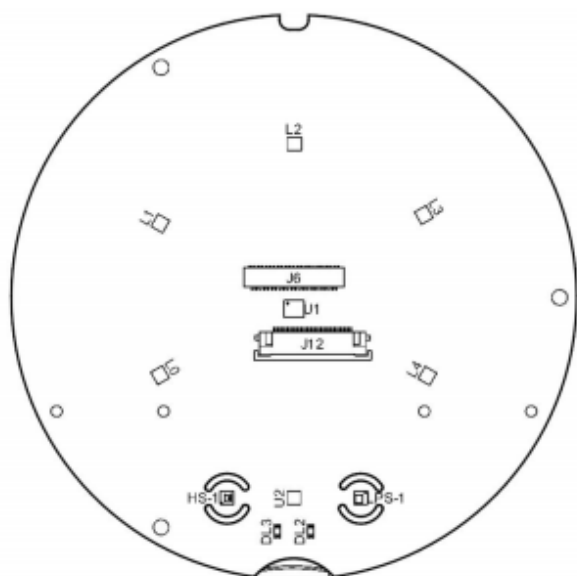
Periféricos

- 5x botones capacitivos
- Vibrador
- 5x LED RGB digitales
- Pantalla OLED redondeada de 1,3"

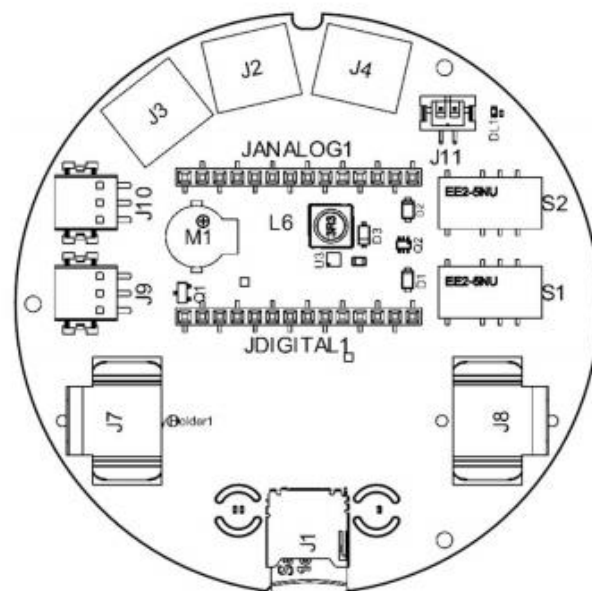
Información de seguridad

- Clase A

Front view

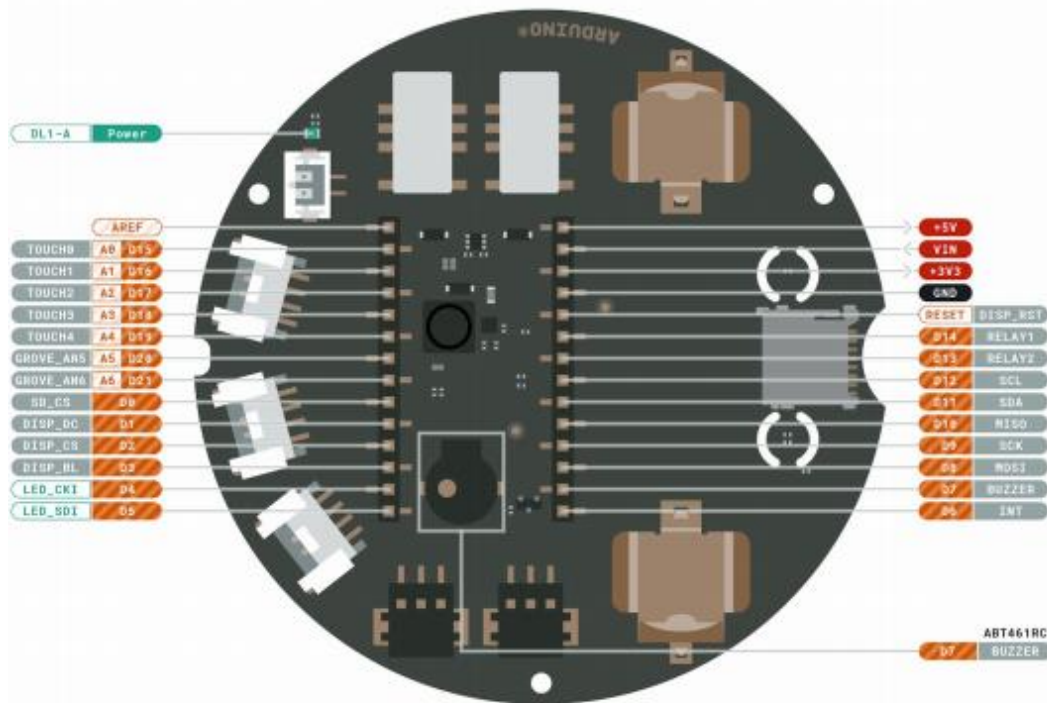


Back view



Ref.	Descripción	Ref.	Descripción
U1	IC IMU de 6 ejes LSM6DSOXTR	U3	Controlador de impulso LED LV52204MTTBG IC
U2	APDS-9660 RGB Sensor de gestos IC	HS-1	IC del sensor de humedad HTS221
LPS-1	IC del sensor de presión LPS22HBTR	L0-L4	APA1022020-2018 RGB LED IC
J6	Conector de pantalla FH26W-45S-0.3SHW (60)	J12	Conector de pantalla SFV24R-1STBE1HLF
J2,J3	Conectores de ranura analógica	J4	Conector de ranura I2C
J11	Conector de energía B2B-PH-SM4-TB (LF) (SN)	J9, J10	1771033 conector de relay de 3 clavijas
JDIGITAL1	MKR Power y conector de pin digital	JANALG1	MKR AREF y conector de pin analógico
S1,S2	EE2-5NU-L 2A 250V Mecánico Relay	M1	Vibrador
J1	Módulo micro SD	J7, J8	Soporte Li-Ion 18650

5. Connector Pinouts



■ Ground
 ■ Digital Pin
 ■ Analog Pin
 ■ Other Pin



This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

Puertos de energía

Pin	Función	Tipo	Descripción
1	GND	Energía	Tierra
2	VBAT	Energía	Terminal positiva de batería 18650

A5

Pin	Función	Tipo	Descripción
1	AN5	Analógico	Entrada analógica 5
2	NC	NC	No conectada
3	+5V	Energía	Riel de alimentación de + 5,0 V
4	GND	Energía	Tierra

A6

Pin	Función	Tipo	Descripción
1	AN6	Analógico	Entrada analógica 6
2	NC	NC	No conectada
3	+5V	Energía	Riel de alimentación de + 5,0 V
4	GND	Energía	Tierra

I2C

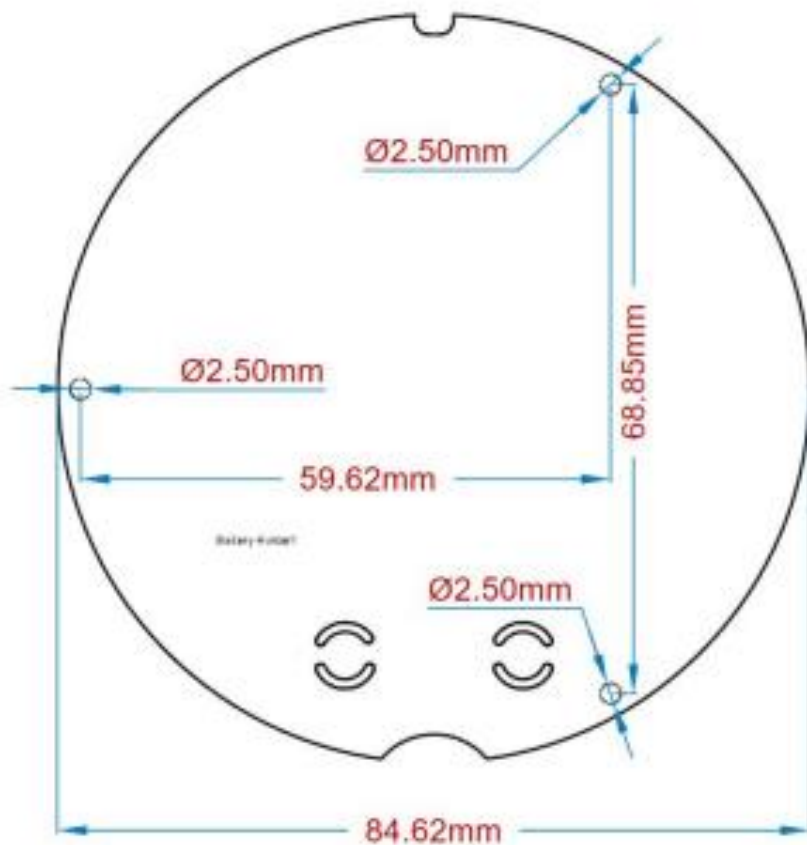
Pin	Función	Tipo	Descripción
1	SLC	Digital	Señal de reloj I2C
2	SDA	Digital	Señal de datos I2C
3	+5V	Energía	Riel de alimentación de + 5,0 V
4	GND	Energía	Tierra

Relay 1/Relay 2

Pin	Función	Tipo	Descripción
1	NC	Switch	Normalmente cerrado
2	COM	Switch	Común
3	NO	Switch	Normalmente abierto

Índices de máximos absolutos

Símbolo	Descripción	Min	Typ	Max	Unidades
Tmax	Límite térmico máximo	-30	20	85	°C
VBATTmax	Voltaje de entrada máximo de la entrada de la batería	3.2	3.7	4.3	V
ARELAYmax	Corriente máxima a través del interruptor de relay	-	-	2	A
Pmax	Consumo máximo de energía	-	-	5000	mW



En el siguiente enlace puede validar las funciones que se pueden realizar con un módulo MKR:
<https://www.youtube.com/watch?v=HqhgG9x6OE>

NOTA: SE RECOMIENDA UTILIZAR MODÚLOS MKR, LOS MODÚLOS SE VENDEN POR SEPARADO

MARCA

 **Distribuidor autorizado**



Como hacer un Datalogger midiendo la temperatura y la humedad con Arduino.

Introducción

En este artículo mostraremos como realizar un datalogger que almacenará las lecturas de temperatura y humedad dentro de una microSD para posteriormente visualizarse en el software de Excel.

Los elementos que se utilizarán son los siguientes:

- Arduino MKR Zero - [ABX00012](#)
- Carrier Arduino MKR IoT - [ABX00047](#)
- MicroSD de 16GB - [MICROSD-16GB-U3-C10](#)
- Fuente de alimentación 5V con conector microUSB - [SAW15-050-3000UD](#)

Conexión de hardware

Para iniciar con la conexión correcta del hardware siga los siguientes pasos:

1.- Coloque el Arduino MKR Zero encima del Carrier Arduino MKR IoT como se muestra en la Figura1.

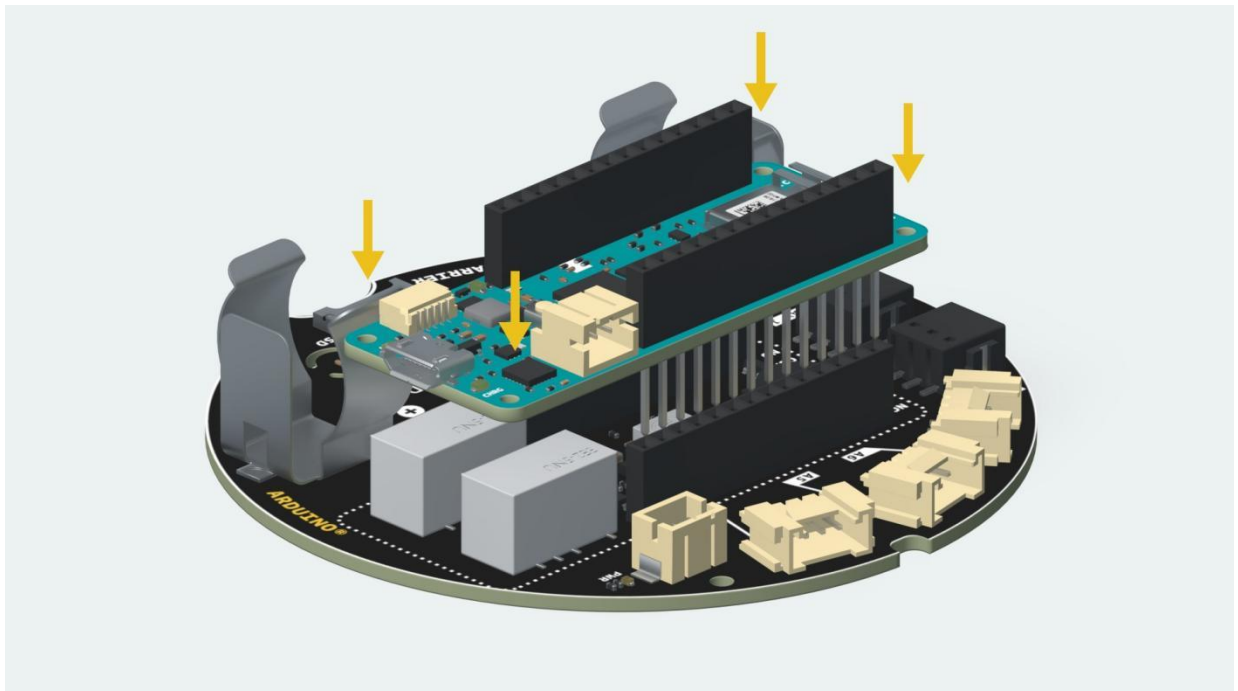


Figura1.

2.- Introduzca la microSD en el Arduino MKR Zero como se muestra en la Figura2.

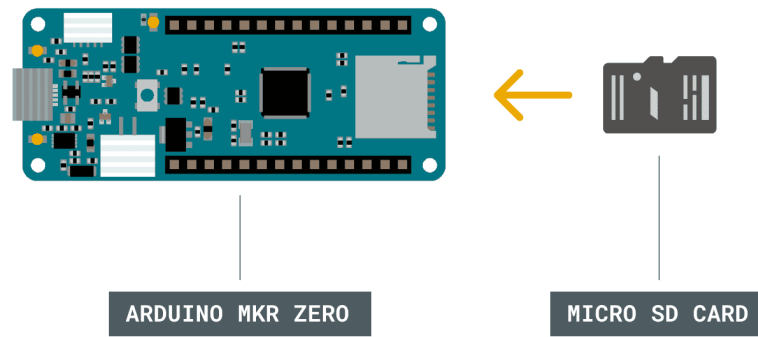


Figura2.

3. - Conecte la tarjeta Arduino MKR Zero previamente montada sobre Carrier Arduino MKR IoT a su computadora como se muestra en la Figura3.



Figura3.

Nota: Asegúrate de conectar la tarjeta de la misma forma que en la Figura1. Una buena manera de hacerlo es mirar los números de pin de la placa y de la Carrier y asegurarse de que coinciden.

Programación

A continuación veremos paso a paso la estructura del código:

I. Declaración de librerías y variables.

En este apartado declaramos las librerías y variables necesarias para el programa.

```
#include <Arduino_MKRIoTCarrier.h>
MKRIoTCarrier carrier;
const int chipSelect = 0;
float temp1=0.0;
float humidity = 0.0;
String dataString="";
```

II. Función Setup()

En la función Setup() establecemos las condiciones iniciales del programa, tales como velocidad de transmisión / recepción del puerto serial, inicialización correcta de la microSD y la inicialización correcta del Carrier Arduino MKR IoT.

```
void setup() { Serial.begin(9600);
  SD.begin(chipSelect);
  delay(5000);
  if (!carrier.begin()) {
    Serial.println("Error al inicializar los sensores");
  }
  else{
    Serial.println("Sensor iniciado correctamente!");
  }
  while (!SD.begin(chipSelect)) {
    Serial.println("Error al leer microsd");
  }
  Serial.print("Leyendo SD.....");
  Serial.println("SD preparada");
  //CARRIER_CASE = false;
}
```

III. Función void loop()

Dentro de la función void loop están todas las funciones que se ejecutaran en el programa ciclicamente.

IV. Función tempSensor()

La función voidSensor() se encargará de leer la temperatura proveniente del sensor integrado en el Carrier Arduino MKR IoT, almacenará la variable e imprimirá por serial su valor en grados Celsius.

```
void tempSensor(){  
    temp1 = carrier.Env.readTemperature(CELSIUS);  
    Serial.print("Temperature = ");  
    Serial.print(temp1);  
    Serial.println(" °C");  
}
```

V. Función humiditySensor

La función humiditySensor() se encargará de leer la humedad proveniente del sensor integrado en el Carrier Arduino MKR IoT, almacenará la variable e imprimirá por serial su valor en porcentaje.

```
void humiditySensor(){  
    humidity = carrier.Env.readHumidity();  
    Serial.print("Humidity = ");  
    Serial.print(humidity);  
    Serial.println(" %");  
    Serial.println();  
}
```

VI. Función `SensorstorageSD()`

Esta función se encarga de almacenar en la microSD las variables que contienen las lecturas de temperatura y humedad en un formato .csv, formato que permite ser exportado fácilmente a Excel.

```
void storageSD(){
    dataString += "Temperatura," + String(temp1) + ",C," + "Humedad," + String(humidity) + ",%";
    File dataFile = SD.open("datalog.csv", FILE_WRITE);
    if (dataFile) {
        dataFile.println(dataString);
        dataFile.close();
    }
    else {
        Serial.println("error opening datalog.txt");
    }
}
```

Código completo

```
#include <Arduino_MKRloTCarrier.h>
MKRloTCarrier carrier;
const int chipSelect = 0;
float temp1=0.0;
float humidity = 0.0;
String dataString="";

void setup() {
  Serial.begin(9600);
  SD.begin(chipSelect);
  delay(5000);

  if (!carrier.begin()) {
    Serial.println("Error al inicializar los sensores");
  }
  else{
    Serial.println("Sensor iniciado correctamente!");
  }

  while (!SD.begin(chipSelect)) {
    Serial.println("Error al leer microsd");
  }

  Serial.print("Leyendo SD.....");
  Serial.println("SD preparada");
  //CARRIER_CASE = false;
}

void tempSensor(){
  temp1 = carrier.Env.readTemperature(CELSIUS);
  Serial.print("Temperature = ");
  Serial.print(temp1);
  Serial.println(" °C");
}

void humiditySensor(){
  humidity = carrier.Env.readHumidity();
  Serial.print("Humidity = ");
  Serial.print(humidity);
  Serial.println(" %");
  Serial.println();
}

void storageSD(){
  dataString += "Temperatura," + String(temp1) + ",C," + "Humedad," + String(humidity) + ",%";
  File dataFile = SD.open("datalog.csv", FILE_WRITE);
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
  }
  else {
    Serial.println("error opening datalog.txt");
  }
}

void loop() {
  dataString = "";
  tempSensor();
  humiditySensor();
  storageSD();
  delay(1000);
}
```