Universidad Tecnológica de Chihuahua

TECNOLOGÍAS DE LA INFORMACIÓN: DESARROLLO DE SOFTWARE MULTIPLATAFORMA

PRINCIPIOS DE IOT

Reporte de investigación sobre Hardware Abierto

Docente: Evelyn Hinojos

Oscar Ariel Quintana Merino
TID41M

Hardware Abierto

Hardware abierto o “Open Source Hardware” se refiere al diseño de las especificaciones de un objeto la cual están licenciadas de tal forma en la que dicho objeto puede ser estudiado, modificado, creado y distribuido por cualquiera. Este conjunto de principios de diseño y prácticas legales no se limita a un tipo específico de objeto, pudiendo aplicarse a automóviles, sillas, computadoras, robots o incluso casas.

La principal diferencia entre el hardware abierto y otros tipos de hardware radica en su accesibilidad y la posibilidad de modificación. Todo hardware abierto debe ir acompañado de documentación, incluyendo archivos de diseño y código fuente. Si incorpora software, este debe ser de código abierto o tener interfaces documentadas para facilitar la escritura de software de código abierto.

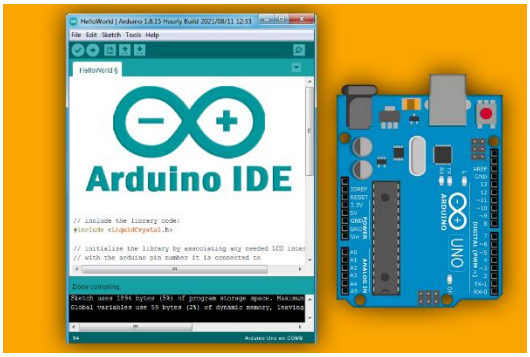
A diferencia del software de código abierto, los proyectos de hardware abierto involucran inversión monetaria en materiales físicos. Sin embargo, iniciativas como el Open Source Hardware Central Bank buscan asegurar financiamiento sostenible para proyectos de hardware abierto.

Entorno de programación de Hardware abierto

El entorno de programación del hardware abierto es un conjunto de herramientas y software utilizado para diseñar, programar y desarrollar proyectos basados en hardware abierto. Este entorno se centra en proporcionar a los usuarios las herramientas necesarias para crear y modificar el diseño de dispositivos físicos, incentivando la transparencia, la colaboración y la accesibilidad en el proceso de desarrollo del Hardware Abierto.

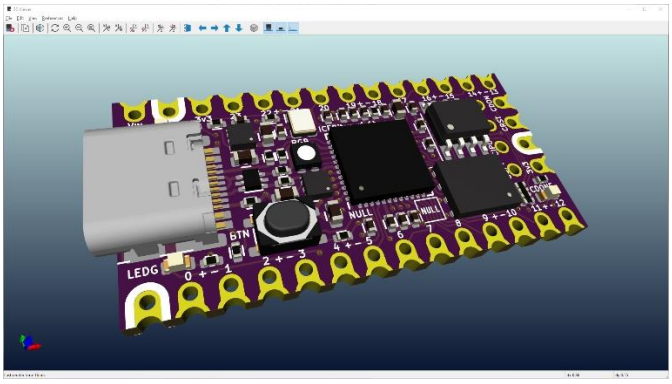
El entorno de programación para hardware abierto puede variar según el tipo de hardware y las preferencias del desarrollador, pero algunos entornos comunes incluyen:

- **Arduino IDE:**
  - Es uno de los entornos más populares para hardware abierto. Utiliza un lenguaje de programación propio basado en C/C++. Arduino es ampliamente utilizado en proyectos de electrónica y robótica.

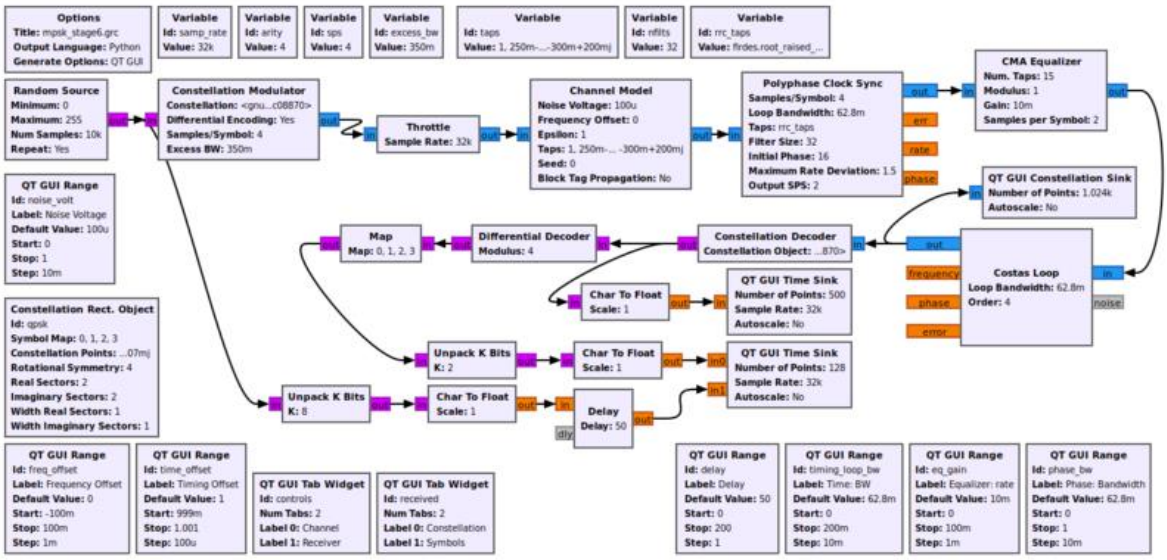


- **PlatformIO:**
  - Una plataforma que soporta múltiples frameworks y placas de desarrollo, incluyendo Arduino, ESP-IDF, STM32, entre otros. Proporciona un entorno unificado para el desarrollo de hardware abierto.

- **KiCad:**
  - Aunque KiCad es principalmente un software de diseño de circuitos, también proporciona herramientas para la generación de archivos de fabricación y ensamblaje. Es utilizado para proyectos de diseño de hardware.



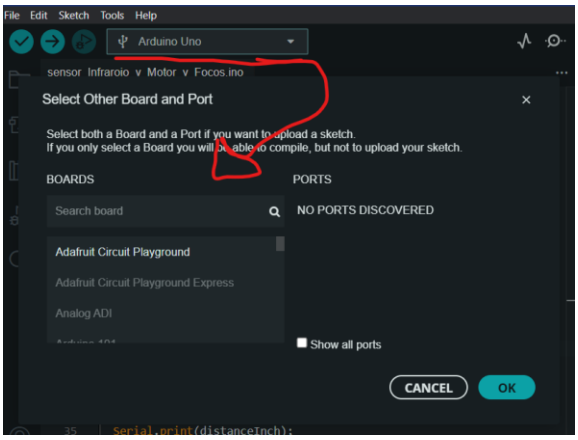
- **GNU Radio:**
  - Para proyectos de comunicación por radio definida por software (SDR), GNU Radio es una herramienta esencial. Permite la implementación de procesamiento de señales en tiempo real en hardware abierto.



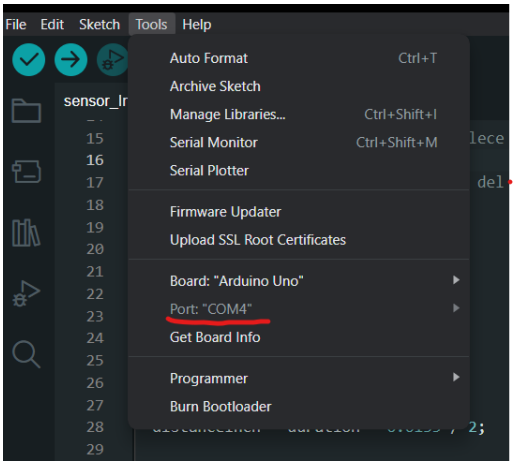
Ejemplo de configuración del hardware abierto (Arduino IDE)

Un ejemplo de configuración básica para el hardware abierto Arduino implica la creación de un simple programa "Hola Mundo" que hace parpadear un LED. A continuación, se presenta un ejemplo utilizando el entorno de desarrollo integrado (IDE) de Arduino:

1. **Configuración de Hardware:**
    - Conecta un LED a uno de los pines digitales de la placa Arduino. Por ejemplo, puedes conectar el ánodo del LED al pin digital 13 y el cátodo a tierra (GND).
2. **Abrir el Entorno de Desarrollo Arduino:**
    - Descarga e instala el entorno de desarrollo Arduino desde el sitio web oficial. Abre el IDE de Arduino.
3. **Seleccionar el Tipo de Placa:**
    - Ve a "Herramientas" -> "Placa" y selecciona "Arduino Uno" o el modelo de placa que estés utilizando.



4. **Seleccionar el Puerto:**
    - Ve a "Herramientas" -> "Puerto" y selecciona el puerto al que está conectada tu placa Arduino.



5 **Escribir el Código:**

- Utiliza el siguiente código como ejemplo en el IDE de Arduino:

```
// Definir el pin al que está conectado el LED
const int ledPin = 13;

// Configuración del programa
void setup() {
  // Inicializar el pin del LED como salida
  pinMode(ledPin, OUTPUT);
}

// Bucle principal del programa
void loop() {
  // Encender el LED
  digitalWrite(ledPin, HIGH);

  // Esperar 1 segundo
  delay(1000);

  // Apagar el LED
  digitalWrite(ledPin, LOW);

  // Esperar 1 segundo
  delay(1000);
}
```

6. **Cargar el Programa en la Placa:**

- Conecta la placa Arduino a tu computadora y presiona el botón de "Subir" (flecha hacia la derecha) en el IDE de Arduino para cargar el programa en la placa.

7. **Observar el LED:**

- Observa el LED conectado al pin 13 parpadeará cada segundo.

Sintaxis Principal

A continuación, se presentará la sintaxis principal de uno de los lenguajes de programación principales, se utilizará el lenguaje de programación utilizado en Arduino IDE para modificar un Arduino:

Estructura del Programa:

- Un programa de Arduino generalmente tiene dos funciones principales: **setup()** y **loop()**. La función **setup()** se ejecuta una vez al inicio del programa, y la función **loop()** se ejecuta repetidamente.

```
void setup() {
  // Código de inicialización
}

void loop() {
  // Bucle principal
}

```

Declaración de Variables:

- Se pueden declarar variables usando tipos de datos como int, float, char, etc. También se pueden utilizar tipos de datos específicos de Arduino como byte.
-
```
int ledPin = 13;
```

Entrada/Salida (E/S):

- Para configurar pines como entradas o salidas, se utiliza pinMode().

```
pinMode(ledPin, OUTPUT);
```

Control de Flujo:

- Se utilizan estructuras de control de flujo estándar como if, else, while, for, etc.

```
if (condition) {
  // Código si la condición es verdadera
} else {
  // Código si la condición es falsa
}
```

Funciones de E/S Digital y Analógica:

- Para leer o escribir valores en pines digitales se utilizan **digitalRead()** y **digitalWrite()**. Para pines analógicos, se usan **analogRead()** y **analogWrite()**.

```
digitalWrite(ledPin, HIGH);
int value = analogRead(sensorPin);
```

Funciones de Retorno:

- Se pueden usar funciones con valores de retorno y parámetros.

```
int sum(int a, int b) {
  return a + b;
}
```

Delay:

- Para agregar pausas en el programa, se utiliza la función **delay()**.

```
delay(1000); // Pausa de 1 segundo
```

Comentarios:

- Los comentarios se pueden agregar usando // para comentarios de una línea o /* */ para comentarios de varias líneas.

```
// Esto es un comentario de una línea

/*
  Esto es un comentario
  de varias líneas
*/
```

Fuentes de Consulta:

- Opensource.com. (2023). What is open hardware? Recuperado de https://opensource.com/resources/what-open-hardware

- SG Noticias. (2020). Open hardware. Recuperado de https://sg.com.mx/revista/open-hardware

- Arduino. (2023). Arduino Software. Recuperado de https://www.arduino.cc/en/software