PUERTOS GPIO Y PROGRAMACIÓN ORIENTADA A OBJETOS EN LA RESOLUCIÓN DE PROBLEMAS

K. Peña, E. Simbaña, MJ. Vizuete

Resumen
El uso de las PCB fue un paso gigante en la evolución de los dispositivos electrónicos, ya que proporcionó un método innovador para la conexión de elementos sin necesidad de utilizar cables eléctricos(Monk,S.2016).A su vez la Poo ha revolucionado el mundo de la programación ya que su principal importancia radica en que, favorece la creación de programas de calidad, fuerza en mantenimiento, en extensión y reutilización de programas. Está basada en el modo de pensar del hombre y en el modo de trabajar de la máquina, el elemento básico de esta programación no es solo la función sino un ente denominado objeto.

I. INTRODUCCIÓN

El objeto de investigación de este artículo es comprender el modo en que se desarrolla el diseño de un servidor web Rest y un cliente rest en el cual está basado en un sistema basado en cliente-servidor, mismo en el que clientes y los servidores tienen partes distintas que desempeñar y todo esto se logrará través de la herramienta Node-Red con nodos http.Esto fin de que el usuario forme parte activa de esto.

II. MARCO TEÓRICO

A)Pines GPIO (General Purpose Input Output):

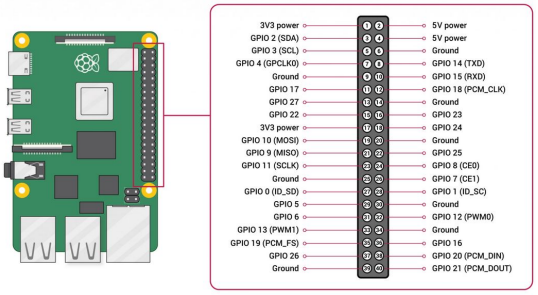
General Purpose Input Output (GPIO) se trata de un sistema de entrada y salida de propósito general, este consta de una serie de pines o conexiones que se pueden usar como entradas o salidas. Estos pines son digitales, lo que significa que sólo pueden tener dos estados. Tienen una dirección para recibir o enviar corriente (entrada, salida respectivamente) y todo esto es controlable por lenguajes de programación como Python, JavaScript, node-RED, entre otros. Los pines trabajan con una tensión de 3,3 V y un consumo máximo de corriente de 16 mA.[1]

1)Control de GPIO con Python

Para esto, lo primero es importar la librería que trabaja con la raspberry pi, la cual es RPi.GPIO.
Existen dos maneras de identificar la nomenclatura de los pines:

- Board: Este es el método más simple, y se refiere a la ubicación física de los pines. Empezando por la parte superior izquierda del GPIO, se encuentra el pin físico 1 que proporciona alimentación 3v3. A la derecha de ese pin se encuentra el 2 que proporciona 5v de potencia. El número de pines seguirá creciendo a medida que se desciende por las columnas. A la izquierda se encontrarán todos los pines con números impares y a la derecha los pares.
- Broadcom: La numeración de pines de Broadcom (BCM), se refiere a los pines que están conectados directamente al SoC del Raspberry Pi. Se puede decir que sus conexiones son directas al cerebro de la raspberry.[2]

2) Tipos de pines de la Raspberry



- **Pines de alimentación:** Se puede apreciar pines de 5v, 3v3 (limitados a 50 mA) y tierra (GND o Ground), que aportan alimentación a estos voltajes para los circuitos.
- **DNC (Do Not Connect):** son pines que por el momento no tienen función, pero en futuras implementaciones si son utilizados. Se encuentran en modelos más primitivos de la Raspberry Pi. En las actuales placas han sido marcados como GND.

- **GPIO normales:** son conexiones configurables que se pueden programar para los distintos proyectos.
- **GPIO especiales:** dentro de éstos se encuentran algunos pines destinados a una interfaz UART, con conexiones TXD y RXD que sirven para comunicaciones en serie. También podemos ver otros como SDA, SCL, MOSI, MISO, SCLK, CE0, CE1, etc…, entre ellos destacan:
    - PWM con el que se puede regular el ancho del pulso.
    - SPI es una interfaz de comunicación.
    - I2C está compuesto por la señal de datos (GPIO2) y el reloj (GPIO3). Además de EEPROM Data (GPIO0) y EEPROM Clock (GPIO1).
    - Serial, otra comunicación muy práctica con pines TX (GPIO14) y RX (GPIO 15) como los que puedes encontrar en la placa Arduino UNO.[3]

B) Programación orientada a objetos (POO) en Python

Se define como un paradigma de la programación, donde se organiza el código en unidades denominadas clases, de las cuales se crean objetos que se relacionan entre sí para conseguir los objetivos de las aplicaciones (López, L. 2006).

Es una forma especial de programar, más cercana a cómo expresamos las cosas en la vida real que otros tipos de programación.

1) Ventajas POO

- Agiliza el desarrollo de software
- Facilita el mantenimiento del software.
- Permite crear sistemas más complejos.
- Fomenta la reutilización y extensión del código.
- Facilita la creación de programas visuales.
- Relacionar el sistema al mundo real.
- Construcción de prototipos

2) Elementos de POO

Clases

Las clases son los modelos o representaciones donde posteriormente se van a crear objetos similares, esta es una plantilla que describe los detalles de un objeto individual.

Se compone de tres cosas: un nombre, atributos y operaciones. [4]

consta de métodos y de datos que resumen las características comunes de dicho conjunto.

Las clases son similares a los tipos abstractos de datos y equivalen a modelos que describen cómo se construyen ciertos tipos de objetos. Cada vez que se construye un objeto a partir de una clase estamos creando lo que se llama una instancia de esa clase.

Objeto

Se trata de un ente abstracto usado en programación que permite separar los diferentes componentes de un programa, simplificando su elaboración, depuración y posteriores mejoras.

Los objetos se integran, a diferencia de los métodos procedurales, tanto los procedimientos como las variables y datos referentes al objeto.(Ramırez, A. O. 2010)

A los objetos se les otorga ciertas características en la vida real. Los objetos se componen de 3 partes fundamentales: métodos, eventos y atributos.

Métodos de un objeto

Se trata de funciones, pero en POO se les llama métodos y representan acciones que puede realizar el objeto, a más de los servicios que realizará el objeto dentro del programa.

Cuando el objeto es creado podemos acceder a sus propiedades y métodos de una forma muy simple: nombre de objeto seguido de un punto y el método o propiedad a la que se quiere acceder.

Siempre el primer parámetro de todos los métodos que se definen en una clase es self, este hace referencia al objeto o instancia que invoca al método.

El método __init__ es un método especial de una clase en Python. El objetivo fundamental de este método es inicializar los atributos del objeto que creamos.

Otras características del método __init__ son:

- Se ejecuta inmediatamente luego de crear un objeto.
- No puede retornar datos.
- Puede recibir parámetros que se utilizan normalmente para inicializar atributos.
- Es un método opcional, de todos modos es muy común declararlo.

sintaxis:
def __init__([parámetros]):
    [algoritmo]

Atributos

Características que aplican al objeto solo en el caso en que el sea visible en pantalla por el usuario; entonces sus atributos son el aspecto que refleja, tanto en color, tamaño, posición, si está o no habilitado.

- Las variables de instancia también denominados miembros datos, son declaradas en la clase pero sus valores son fijados y cambiados en el objeto. Además de las variables de instancia hay variables de clase, las cuales se aplican a la clase y a todas sus instancias.

C)Lógica digital

La lógica digital es una ciencia de razonamiento aplicada a circuitos eléctricos que realizan decisiones de tipo (SI) y (NO), donde una serie de circunstancias particulares ocurre, una acción resultara y siempre es el mismo para una serie dada de circunstancias.(Floyd, T,2017)

La posibilidad de predecir el resultado final permite el diseño de sistemas digitales a partir de circuitos básicos llamados compuertas, además de la ayuda de la matemática booleana permite la creación de sistemas electrónicos digitales para casi cualquier evento que necesitemos realizar.[6]

III. EJERCICIOS PROPUESTOS

D)Explicación del código fuente

Como primer punto para la correcta ejecución de nuestro código, llamamos a las librería para el control de los pines, y la librería asociada al tiempo:

```
import RPi.GPIO as GPIO
import time
```

Posterior a esto, asignamos el número de pin que queremos que actúe como led en el programa, a más de elegir el modo en el que se llamarán los pines, y configurarlos como salida.

```
pin1=3
pin2=8
GPIO.setmode (GPIO.BOARD)
GPIO.setup (pin1,GPIO.OUT)
GPIO.setup (pin2,GPIO.OUT)
```

Denominamos la clase del programa.

Seguido de esto nombraremos el primer método menú con su parámetro self; este nos mostrará en pantalla las distintas opciones a ejecutar, las cuales serán ingresadas por medio de la selección de pines.

```
class Alarma:

    def menu(self):
      opc= input('''
      ######### ALARMA #########

      pin11: si se detecta GAS
      pin13: si se detecta HUMO.
      pin15: si la temperatura es superior a 45°C
      pin16: si la temperatura es superior a 60°C

      ''')
      return opc
```

```
Señales de ingreso
 ######### BOMBA DE RIEGO #########

 pin19: si la tierra está seca
 pin21: si hay restricciones en el riego (es verano)
 pin23: si es de día
 pin24: si el depósito de agua está vacío
```

El siguiente método nombrado “llama_pin”, tiene como parámetros self y el pin asignado anteriormente. Aquí es donde se activará el led, inicializando en alto, con un intervalo de 5 segundos de tiempo y diez repeticiones. Posterior a esto se limpiará el GPIO con GPIO.cleanup().

```
def llama_pin(self,pin1):
  print("Alarma activada")
  for i in range(0,10):
    GPIO.setup (3,GPIO.OUT)
    GPIO.output(3,GPIO.HIGH)
    time.sleep(0.5)
    GPIO.output(3,GPIO.LOW)
    time.sleep(0.5)
    GPIO.cleanup()
```

Como tercer método usamos “opción” que tiene como parámetros self y opc que retorna en el primer método. Con esto haremos que que se encienda la alarma o se active la bomba dependiendo de los pines que hayamos seleccionado.

Para ingresar por teclado los pines se usa GPIO.input(pin) seguido de la salida en la que se encontrará el pin, la cual puede ser GPIO.HIGH o GPIO.LOW.

IV. METODOLOGÍA

En el desarrollo de este estudio se empleó principalmente el método de revisión bibliográfica, ya que ayudándonos de diversas fuentes bibliográficas obtuvimos datos e información del tema respectivo los cuales fueron analizados , esto nos permitió afianzar los conceptos necesarios para el desarrollo y resolución de los problemas planteados. La investigación se ejecutó por medio de herramientas como así como la herramienta Github y del mismo modo la información recolectada permitió reconocer los principales métodos a utilizar en una arquitectura web rest de la misma forma se encontró la herramienta

XAMPP para la creación de un servidor local en el cual se pueda trabajar

V. RESULTADOS

A partir del análisis y revisión de información se logró desarrollar y ejecutar a través de la plataforma create.withcode.uk dos programas que dan solución a los problemas propuestos de lógica digital en el que mediante los puertos GPIO de la Raspberry Pi se establecen las condiciones necesarias para que tanto una alarma de incendios y una bomba de agua se activen o permanezcan apagadas.Además esto se logró gracias a la programación orientada a objetos que nos permite definir clases y objetos que optimizan nuestro código.

VI. CONCLUSIONES

-El paradigma de programación orientado a objetos facilita el desarrollo de un código ya que nos permite declarar clases y objetos , lo cual nos aportará con diferentes beneficios entre estos:

- Reusabilidad. Cuando hemos diseñado adecuadamente las clases, se pueden usar en distintas partes del programa y en numerosos proyectos.

- Mantenibilidad. Debido a las sencillez para abstraer el problema, los programas orientados a objetos son más sencillos de leer y comprender, pues nos permiten ocultar detalles de implementación dejando visibles sólo aquellos detalles más relevantes.

- Modificabilidad. La facilidad de añadir, suprimir o modificar nuevos objetos nos permite hacer modificaciones de una forma muy sencilla.

- Fiabilidad. Al dividir el problema en partes más pequeñas podemos probarlas de manera independiente y aislar mucho más fácilmente los posibles errores que puedan surgir.

-Los pines GPIO de la Raspberry son de gran utilidad ya que pueden ser usados tanto como entradas como salidas,es así que en este caso una serie de pines mismos que son seleccionados por el usuario nos permitieron definir las condiciones para que la alarma y la bomba se activen o permanezcan inactivas.sus respectivos atributos.

VII. RECOMENDACIONES

Para un buen manejo del código y una excelente programación se recomienda al usuario tener pleno conocimiento de la librería RPi.GPIO y sus diferentes sentencias reservadas pues con este conocimiento la programación en una Raspberry será mucho más amena y mejor elaborada.

Mediante la plataforma create.withcode los programas presentan cierto error al trabajar con pines se recomienda al usuario utilizar la función GPIO.cleanup() al final de cada ejecución de los pines pues esta limpiara nuestra Raspberry y no permitirá que entre en un bucle interrumpiendo nuestra codificación.

VIII. BIBLIOGRAFÍA

[1]Raihan, M. K. J., Rahaman, M. S., Sarkar, M. K., & Mahfuz, S. (2013). Raspberry Pi image processing based economical automated toll system. Global Journal of Research In Engineering.

[2] Rodríguez, H. G., Guzmán, U. D., Atilano, R. M., Rangel, N. B., & Rodríguez, M. J. (2018). Computadoras de placa reducida Raspberry Pi 3 y Asus Tinker Board/Reduced plate computers Raspberry Pi 3 and Asus Tinker Board. *RECI Revista Iberoamericana de las Ciencias Computacionales e Informática*, *7*(14), 46-56.

[3]Pi, R. (2015). Raspberry pi 3 model b. online].(https://www. raspberrypi. org.

[4]Challenger-Pérez, I., Díaz-Ricardo, Y., & Becerra-García, R. A. (2014). El lenguaje de programación Python. Ciencias Holguín, 20(2), 1-13.

[5] Ramırez, A. O. (2010). Python como primer lenguaje de programación. Publicación interna del Tecnológico de Monterrey, Campus Estado de México.

[6] Mano, M. M. (1982). Logica Digital y Diseño. Pearson Educación.