



FIME

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica

Cómputo Integrado

Práctica #4: Sistema de Alarma

Catedrático: Carlos Adrian Perez Cortez

Equipo #12

Nombre	Matricula	Carrera
Jorge Alberto Morales Reyes	1895340	ITS
Kevin Isaias Martinez Saucedo	1886121	ITS
Brandon Aguilar García	2082351	ITS
Nelson Andres Mendez Martinez	1930671	ITS

Grupo: 003

Hora: N4

Ciudad Universitaria, San Nicolás de los Garza, Nuevo León

Fecha: 10/9/2024

Objetivo

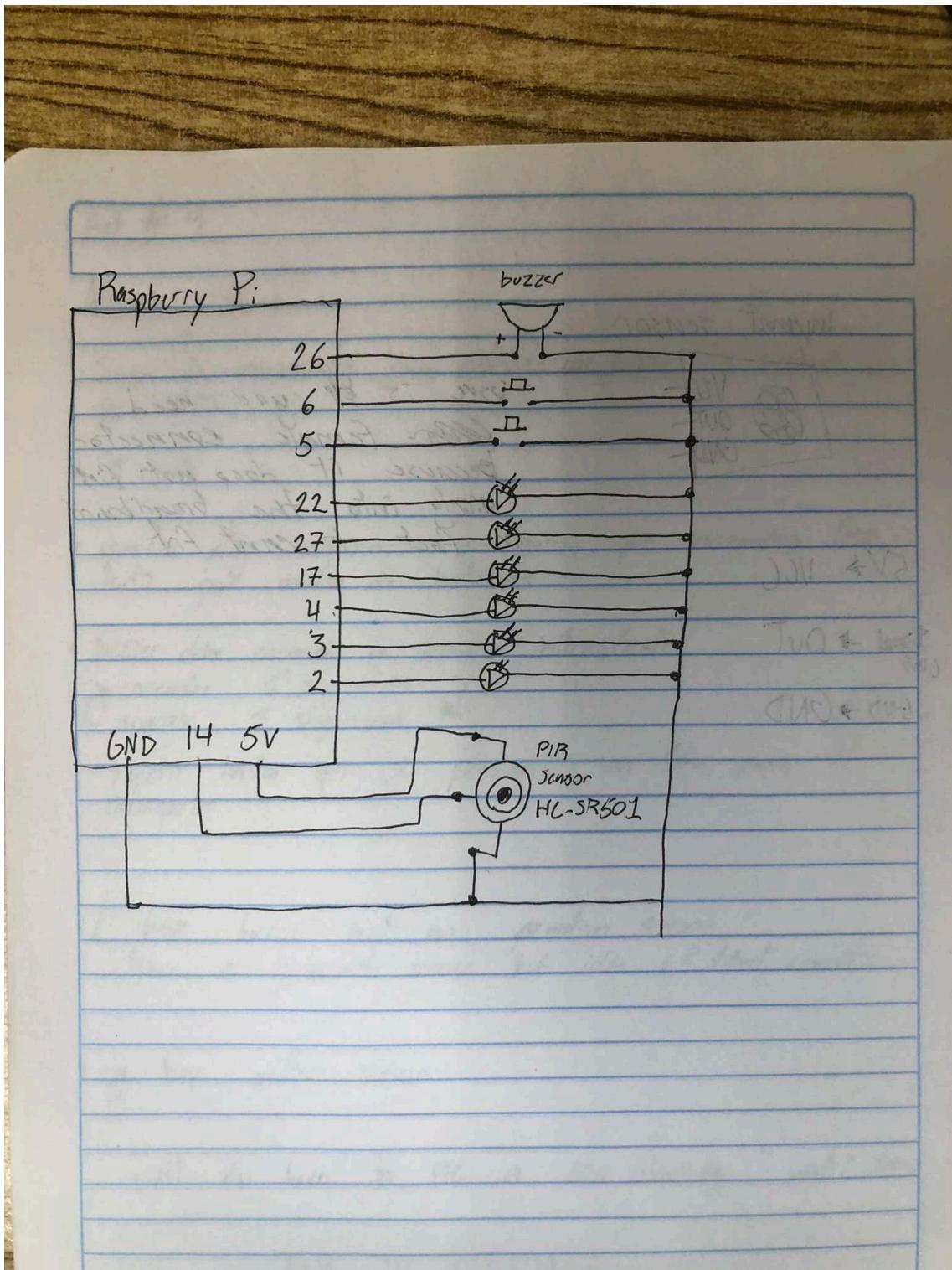
El objetivo de esta práctica es desarrollar un sistema de alarma utilizando una Raspberry Pi que detecte movimiento a través de un sensor PIR. Al detectar movimiento, el sistema activará un buzzer y encenderá una serie de LEDs como señales de alerta.

Hallazgos y Dificultades

Durante el desarrollo del sistema de alarma, el ensamblaje del circuito no presenta dificultades significativas. La conexión del sensor de movimiento, los LEDs y el buzzer fue sencilla, siguiendo las especificaciones y diagramas adecuados. Sin embargo, el mayor desafío surgió al escribir el código, específicamente en la implementación de los tiempos de espera de 5 o 2 segundos sin bloquear la ejecución del programa. Utilizar una función que espera 5 o 2 segundos sería fácil, pero estas funciones bloquean el resto del código que corren. Esto significa que si tienes código que espera si se presiona un botón para apagar la alarma, no funcionará hasta que la función deja de bloquear, esto no es conveniente y esto molestaría a cualquier usuario de este sistema de alarma. Utilizamos la función `time.time()` de manera similar a `millis()` en Arduino para gestionar el tiempo sin detener el flujo del código, lo que resultó complicado al principio.

Aunque no se batallo al armar el circuito porque es simple, lo que si presento dificultades es, *¿Que sensor usar?* Nuestro equipo decidió que tenía que ser un sensor digital porque el Raspberry Pi no tiene puertos analógicos. Investigamos varias opciones de sensores para detectar movimiento. La elección final recayó en el sensor PIR HC-SR501 debido a su fiabilidad y facilidad de uso en aplicaciones de detección de movimiento.

Código/Sketch



Código:

```
from gpiozero import MotionSensor, Buzzer, Button, LED
import time

time.sleep(1)

def mode1_1(leds):
    for i in range(3):
        leds[i].on()
    for i in range(3,6):
        leds[i].off()

def mode1_2(leds):
    for i in range(3):
        leds[i].off()
    for i in range(3,6):
        leds[i].on()

def mode2_1(leds):
    apagar(leds)
    leds[1].on()
    leds[3].on()
    leds[5].on()

def mode2_2(leds):
    apagar(leds)
    leds[0].on()
    leds[2].on()
    leds[4].on()

def apagar(leds):
    for led in leds:
        led.off()

buzzer = Buzzer(26)
sensor = MotionSensor(14)
btn1 = Button(6)
btn2 = Button(5)

led1 = LED(22)
```

```

led2 = LED(27)
led3 = LED(17)
led4 = LED(4)
led5 = LED(3)
led6 = LED(2)
leds = [led1, led2, led3, led4, led5, led6]

alarma = False
buzzer_prendido = False
apagar(leds)
mode = 2

lastT = time.time()

while True:
    curT = time.time()

    if btn1.is_pressed:
        print("cambiar de mode")
        if mode == 1:
            mode = 2
            time.sleep(0.1)
        else:
            mode = 1
            time.sleep(0.1)
    if btn2.is_pressed:
        print("apagar alarma")
        alarma = False
        buzzer.off()
        apagar(leds)

    if alarma:
        if buzzer_prendido:
            if (curT - lastT) >= 1:
                buzzer.off()
                buzzer_prendido = False
            lastT = curT
            if mode == 1:
                mode1_1(leds)
            else:
                mode2_1(leds)

```

```

else:
    if (curT - lastT) >= 0.5:
        buzzer.on()
        buzzer_prendido = True
        lastT = curT
        if mode == 1:
            mode1_2(leds)
        else:
            mode2_2(leds)

else:
    sensor.wait_for_motion()
    alarma = True
    buzzer_prendido = True
    buzzer.on()

```

Conclusión General

Para concluir en esta práctica se desarrolló un sistema de alarma con Raspberry Pi fue una experiencia enriquecedora tanto en el ámbito del hardware como en el software. Aunque el ensamblaje del circuito no representó mayores complicaciones, el verdadero reto radicó en la programación. Particularmente en el manejo del tiempo sin bloquear la ejecución del código, para lo cual se utilizó la función `time.time()` de manera similar a `millis()` en Arduino. La implementación del sensor PIR HC-SR501 demostró ser una elección adecuada, proporcionando una detección confiable de movimiento. A través de este proyecto, se lograron adquirir valiosas habilidades en la integración de componentes electrónicos y en la escritura de código eficiente, lo que refuerza la importancia de planificar tanto el hardware como el software de manera conjunta.

Conclusiones Individuales

Jorge Alberto Morales Reyes:

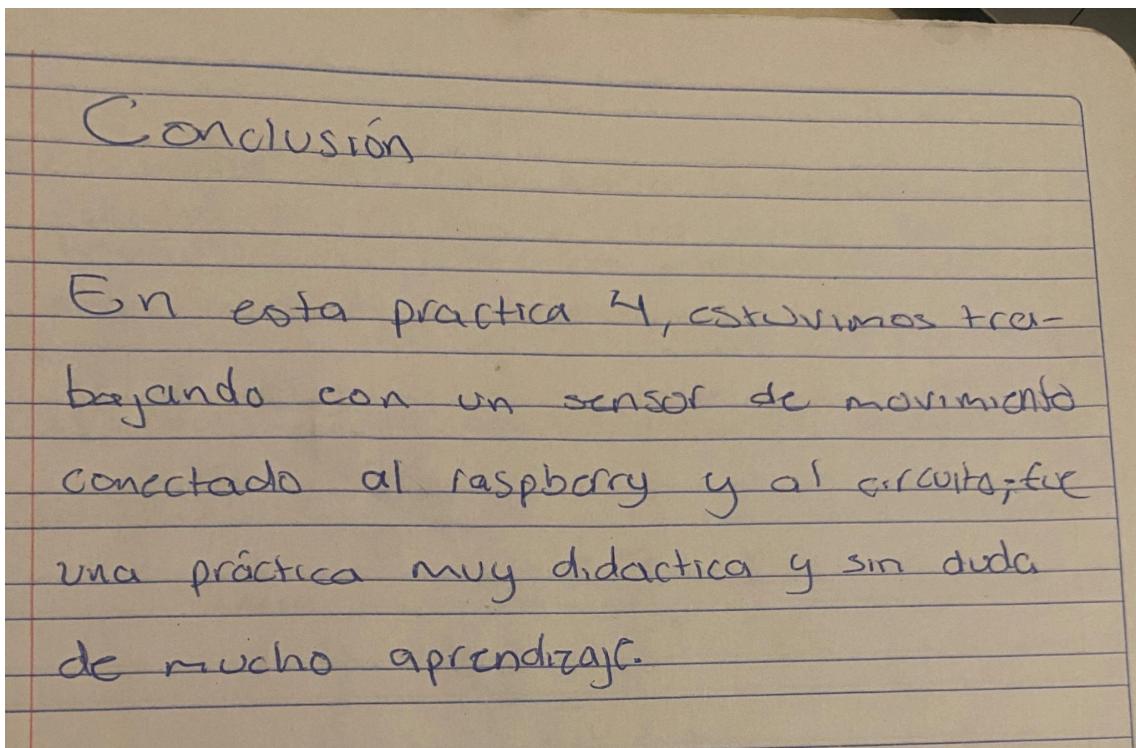
Esta practica fue interesante porque fue el primera en que se ha usado un sensor. Personalmente este fue la primera vez que utilicé un sensor para un circuito y estoy interesado en que otros sensores podrían utilizar, especialmente para mi proyecto final.

Kevin Isaias Martinez Saucedo:

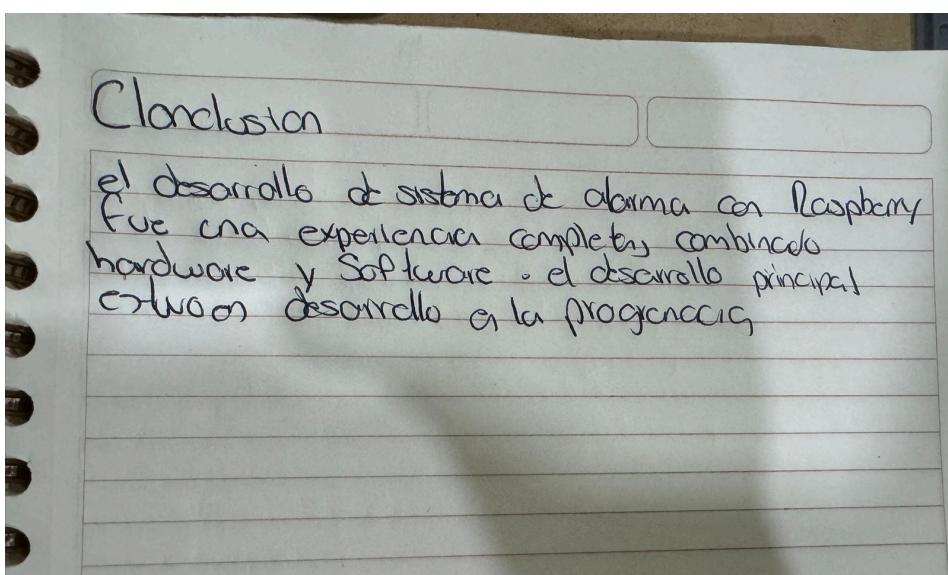
Conclusión

Para esta práctica fue necesario utilizar un sensor de proximidad en el cual al detectar el movimiento se reproduce una alarma al igual que nos avisa, simulando así un sistema de seguridad.

Brandon Aguilar García:



Nelson Andres Mendez Martinez:



Demostración

https://drive.google.com/file/d/19s5y6Z5oUyhPymwOTFxwyDdhRU1yXXT0/view?usp=drive_link