Proyecto Final
Curso de Sistemas Operativos y Laboratorio

Detector de movimiento usando Raspberry PI

John Haider Giraldo
Wilmer Santiago Soto Vidal
Alejandro Cifuentes

Resumen

Este proyecto utiliza un zumbador (buzzer) y un sensor infrarrojo pasivo (PIR) para crear un sistema de detección de movimientos. Cuando el sensor PIR detecta movimiento, genera una señal que activa el zumbador, emitiendo un sonido de alerta. La implementación se realiza utilizando una Raspberry Pi como controlador principal, donde un programa en Python maneja la interacción entre el sensor y el zumbador a través de los pines GPIO. El sistema operativo basado en Linux en la Raspberry Pi gestiona los recursos de hardware y asegura la ejecución del código, permitiendo que el sistema funcione de manera automatizada.

Introducción

¿Cuál es la necesidad y/o problema que aborda el desafío seleccionado?
El proyecto de detector de movimiento utilizando una Raspberry Pi busca mejorar la seguridad y protección de un lugar mediante el uso de un sensor infrarrojo pasivo (PIR) y un zumbador (buzzer). Al detectar movimiento, el sistema genera una alerta en tiempo real a través de un sonido. Este tipo de proyecto tiene aplicaciones prácticas en sistemas de seguridad, donde es fundamental la correcta gestión del hardware por parte del sistema operativo para garantizar una respuesta eficiente y oportuna.

El sistema operativo de la Raspberry Pi, basado en Linux, gestiona los recursos de hardware, como los pines GPIO, que permiten al software controlar el sensor y el zumbador. De esta manera, el proyecto depende del sistema operativo para la correcta interacción entre el software y los componentes físicos, asegurando que las señales de entrada y salida se procesen de manera efectiva.

¿Por qué es importante el desarrollo de este desafío en el contexto tecnológico actual?
En un contexto tecnológico donde la Internet de las Cosas (IoT) y la seguridad son cada vez más relevantes, es esencial que los sistemas operativos gestionen de manera eficiente los recursos de hardware. Estos dispositivos deben ser capaces de proporcionar alertas en tiempo real, lo que implica una óptima gestión de procesos, memoria y entrada/salida por parte del sistema operativo. La capacidad de manejar múltiples procesos en paralelo y administrar las interrupciones de hardware es clave para el éxito de estos sistemas de seguridad.

Este proyecto también facilita la exploración de cómo los sistemas operativos embebidos, como los utilizados en la Raspberry Pi, pueden contribuir a desarrollar tecnologías más accesibles y asequibles para la seguridad en el hogar o en entornos industriales. La interacción entre el hardware y el software mediante el sistema operativo asegura una solución completa y robusta.

Antecedentes o marco teórico

¿Cuáles son los principales aspectos teóricos necesarios para comprender el desafío y llevarlo a cabo?
Para desarrollar este proyecto de detector de movimiento utilizando una Raspberry Pi, es fundamental entender los siguientes aspectos teóricos:

- **Programación en Raspberry Pi:** Es necesario conocer el uso de lenguajes como Python para interactuar con los pines GPIO y controlar dispositivos externos, como el sensor PIR y el zumbador.

- **Integración con sensores externos:** Comprender cómo los sensores, como el sensor infrarrojo pasivo (PIR), detectan cambios en el entorno físico y comunican esos datos a través de señales digitales que pueden ser procesadas por la Raspberry Pi.

- **Conceptos básicos de electrónica:** Incluir conocimientos sobre cómo conectar y alimentar dispositivos electrónicos externos (como sensores y zumbadores) a través de la Raspberry Pi.

¿Qué relación tiene esta teoría con los temas del curso de Sistemas Operativos (en la parte teórica y para el laboratorio)?

El proyecto está profundamente relacionado con los sistemas operativos en varios aspectos clave, que también son parte fundamental del curso:

- **Interrupciones de hardware:** Aunque el proyecto actual no utiliza interrupciones de hardware explícitamente, en sistemas más avanzados, las interrupciones son fundamentales para notificar al sistema operativo cuando un dispositivo de hardware, como el sensor PIR, detecta un evento (por ejemplo, movimiento). Esto permite una respuesta rápida y eficiente del sistema operativo para manejar la señal.

- **Controladores de dispositivos:** El sistema operativo de la Raspberry Pi, basado en Linux, utiliza controladores de dispositivos (drivers) para gestionar la comunicación entre el hardware y el software. En este proyecto, la interacción con los pines GPIO, que permite leer los datos del sensor y activar el zumbador, está mediada por estos controladores, lo que ilustra claramente la función de los controladores en el manejo de hardware periférico.

- **Gestión de recursos de hardware:** El sistema operativo también es responsable de gestionar los recursos de hardware, como los pines GPIO, asegurando que los procesos en ejecución puedan acceder a ellos de manera coordinada y sin conflictos.

- **Manejo de I/O (Entrada/Salida):** En el contexto del proyecto, el sistema operativo gestiona las operaciones de entrada/salida, permitiendo que el software lea los datos del sensor PIR (entrada) y controle el zumbador (salida). Esto está directamente relacionado con el manejo de I/O, que es un tema fundamental en el curso de sistemas operativos.

Objetivos (principal y específicos)

Principal
Desarrollar un sistema de detección de movimiento utilizando una Raspberry Pi, un sensor de movimiento PIR y un buzzer, con el fin de crear una solución accesible para la seguridad y monitoreo de espacios, gestionada por el sistema operativo para una interacción fluida entre el hardware y el software.

Específicos
- **Integrar y programar el sensor PIR con la Raspberry Pi:** Utilizar los pines GPIO controlados por el sistema operativo para recibir señales de entrada del sensor PIR.
- **Configurar el buzzer para emitir un sonido de alerta:** Utilizar el sistema operativo para gestionar la salida hacia el buzzer y asegurar que emita una señal sonora en respuesta al movimiento detectado.
- **Diseñar e implementar el sistema de alerta en tiempo rea**l: Garantizar que el sistema operativo maneje eficientemente las operaciones de entrada/salida (I/O) y los procesos, permitiendo una respuesta rápida y en tiempo real.

- **Documentar el desarrollo y los resultados del proyecto:** Registrar el proceso de desarrollo, con especial enfoque en cómo el sistema operativo gestiona los recursos de hardware (pines GPIO) y los procesos de software.

Metodología

¿Cuáles son las principales herramientas que podrían utilizarse para implementar la solución?

Hardware

- **Raspberry Pi**: La plataforma que ejecuta el sistema operativo y gestiona la interacción entre hardware y software.
- **Sensor PIR:** Detecta el movimiento y envía señales al sistema operativo a través de los pines GPIO.
- **Buzzer:** Genera una alerta sonora cuando se activa mediante los pines GPIO gestionados por el sistema operativo.
- **Protoboard y cables de conexión:** Facilitan las conexiones entre los dispositivos.

Software

- **Python:** Lenguaje de programación utilizado para desarrollar el programa que interactúa con el sistema operativo y controla los pines GPIO de la Raspberry Pi.
- **RPi.GPIO:** Librería que facilita la interacción entre el software (Python) y el hardware (pines GPIO), gestionada por el sistema operativo.

¿Qué actividades son necesarias para cumplir los objetivos?

1. I**nvestigación y planificación del proyecto:** Definir cómo el sistema operativo gestionará los recursos de hardware y el software.
2. **Preparación y adquisición de materiales a usar:** Conseguir los componentes necesarios y preparar el entorno de trabajo, incluyendo el sistema operativo en la Raspberry Pi.
3. **Montaje del hardware:** Conectar el sensor PIR y el buzzer a la Raspberry Pi mediante los pines GPIO.
4. **Desarrollo del software:** Programar en Python el código que interactúa con el sistema operativo para controlar los sensores y el buzzer.
5. **Pruebas y ajustes**: Validar que el sistema operativo maneje correctamente las señales de entrada del sensor PIR y la salida hacia el buzzer.
6. **Documentación del proyecto:** Detallar cómo el sistema operativo gestionó el hardware y los procesos involucrados en la detección de movimiento.

Diseño de Experimentos

¿Cómo espera validar la funcionalidad y el rendimiento de la solución desarrollada en este desafío?

Título de experimento	Validación de la precisión del sensor PIR y respuesta del buzzer
Descripción	Se desea evaluar la capacidad del sistema para poder detectar movimiento y verificar que posteriormente se emita la señal sonora mediante el buzzer.
Razonamiento	La correcta detección de movimiento y posterior activación del buzzer son fundamentales para el correcto funcionamiento del sistema.
Información importante	Es importante recordar que para el funcionamiento efectivo del sistema es necesario que el sensor PIR sea bastante preciso y no tenga una tasa de falsos positivos alta. También que el buzzer emita el sonido en el momento que se detecte el movimiento.
Hipótesis	Asumiendo que el sensor PIR esté calibrado de forma correcta este debería tener una tasa de detección positiva verdadera del 95% al menos.
Predicciones	1. El sensor PIR detecta la mayoría de movimientos frente a él de forma correcta. 2. El buzzer emite el sonido de forma correcta todas las veces o la mayoría.
Métodos	1. Se prepara el entorno de pruebas y se verifica que todo el sistema esté configurado correctamente 2. Se simulan diferentes pruebas en distintas situaciones. Se prueban el ángulo de detección y la distancia, luego se anota el comportamiento del buzzer. 3. Se analizan los datos que se recolectaron y se determina la tasa de detecciones correctas y falsos positivos.
Métricas	Al analizar los datos se tendrán en cuenta los falsos positivos y la tasa de detecciones correctas para así determinar la precisión del sensor PIR
Normalización de datos	Los datos obtenidos se convertirán en porcentajes para así poder analizar los resultados obtenidos.

Réplicas y estadísticas	Se repetirá cada escenario de prueba 5 veces para tener múltiples datos para tener una idea general de la precisión.
Controles	Se asegurará que en el entorno de pruebas no exista alguna otra interferencia de movimiento.
Conclusiones	Las conclusiones se sacan de la comparación entre la tasa de detección esperada (95%) y la tasa de detección correcta que arrojó el sensor PIR. Un buen o mal desempeño entonces nos dice que tan confiable es el sistema.

Cronograma

- **Desarrolle un diagrama de Gantt que refleje el desarrollo de las actividades.**

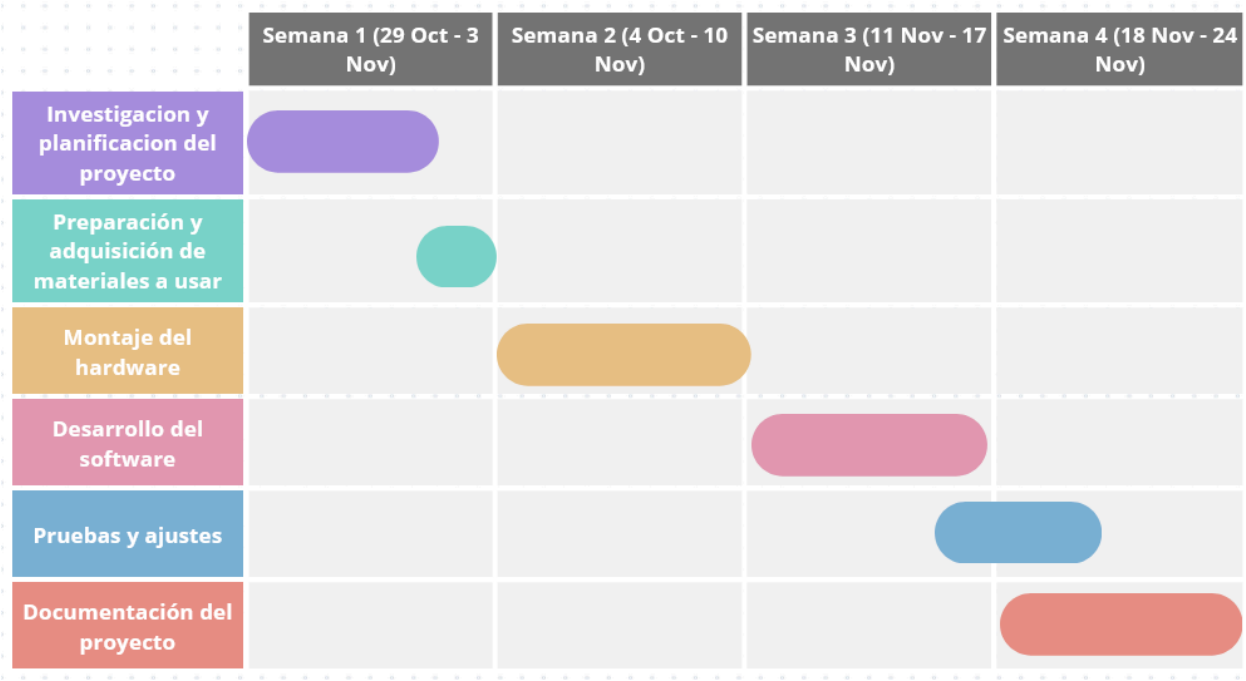

Imagen 1. *Cronograma de actividades.*

Referencias

[Raspberry Pi detector de movimiento con sensor infrarrojo pasivo PIR y buzzer, código en Python.]