Universidad Autónoma de Baja California

Facultad de Ciencias Químicas e Ingeniería

MICROCONTROLADORES

Practica No. 3

Manejo de la sección de E/S del microcontrolador ESP32

Docente: Garcia Lopez Jesus Adan

Alumno: Gómez Cárdenas Emmanuel Alberto

Matricula: 01261509

Indice

Objetivo:

El alumno se familiarizará con la configuración y uso de puertos mediante el uso del ESP-IDF SDK.

Material:

- Computadora Personal y tarjeta de desarrollo del ESP32

Teoría:

Técnicas anti-rebote de botones: La tecnica anti rebote es utilizada para prevenir a un botón de ser presionado varias veces debido a el rebote mecanico. Para realizar esto se cuenta con diferentes métodos**.**

- **Por Software:** Este método consta de realizar una función o rutina que se encargue de ignorar los pulsos de un boton por un periodo de tiempo de ser presionado, de esta forma, registrando solamente el primer pulso.
- **Por Hardware:** Esta técnica involucra el uso de componentes externos como capacitores, resistencias, y disparadores Schmitt, para eliminar el rebote. Con los componentes se crea un filtro low-pass que elimina cualquier ruido de alta frecuencia.
- **Combinación:** En esta técnica se combinan los 2 métodos mejorando la confiabilidad.
- **Basada en interrupciones:** El ESP32 tienen interrupciones de hardware integradas que se pueden utilizar para activar una rutina de servicio de interrupción (ISR) cuando se presiona un botón.

Manejo de Puertos de Entrada y Salida: El microcontrolador ESP32 cuenta con pines de entrada y salida de propósito general (GPIO) que pueden ser configurados y controlar otros componentes electricos. Para configurarse un GPIO se deben seguir los siguientes pasos:

1. **Configurar pin:** Antes de utilizar cualqueir GPIO debe ser configurado como entrada o salida, para ello puede ser utilizado la siguiente instrucción Gpio_reset_pin(PIN_GPIO); y gpio_set_direction(PIN_GPIO, GPIO_MODE_(output|input|input_output);
2. **Escribir al pin (output):** Cuando el pin es configurado como salida se le puede escribir datos usando la funcion gpio_set_level o gpio_set_direction.
3. **Leer el pin (input):** Para leer data del pin, puede usarse la funcón gpio_get_level.

Gómez Cárdenas Emmanuel Alberto
INGENIERIA EN COMPUTACION

Desarrollo

- Revisar documentación del SDK sobre la sección de entrada y salida ESP-IDF GPIO.
- Revisar la configuracion del SDK del parámetro CONFIG_FREERTOS_HZ, para esta práctica se necesitará que esté configurado a 1000. Eso logrará que nuestros retardos puedan tener resolución de 1 mili-segundo.
- Implementar el juego del Rebote (algo similar a “Pong” de 1-dimensión) en base al esquemático de la Fig. 1 y el código proporcionado en el repositorio.

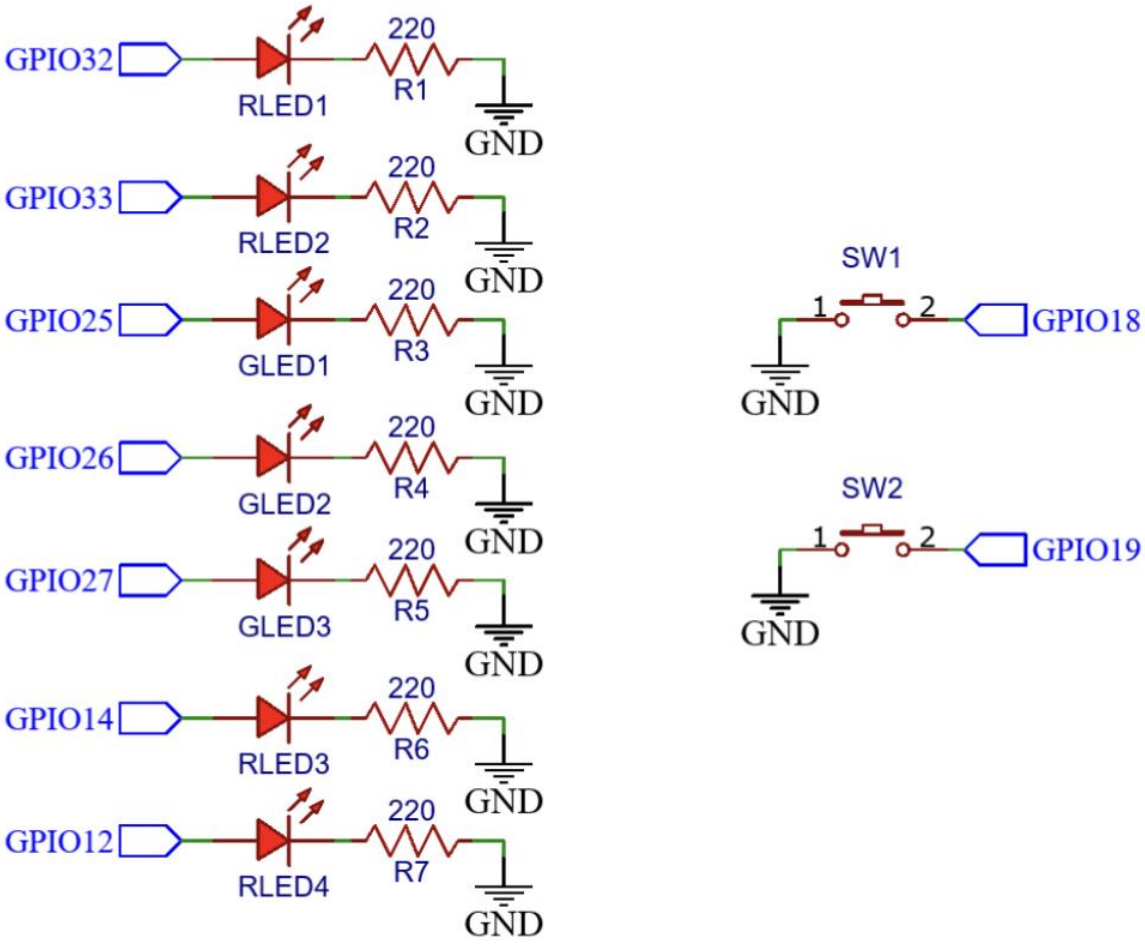

Fig. 1. Esquemático

Funciones a implementar:

Void initIO(void)

Inicialización requerida de los puertos utilizados en esta práctica, según la Fig 1.

eButtonState_t checkButtons(eButtonId_t *buttonNumber)

Retorna el ID del botón detectado y su posible estado eBtnUndefined, eBtnShortPressed o eBtnLongPressed. Donde el umbral para una larga duración es cualquiera que sea mayor a 1 segundo. Es importante ignorar el rebote mecánico, un ejemplo de este rebote se puede apreciar en la Fig 2.

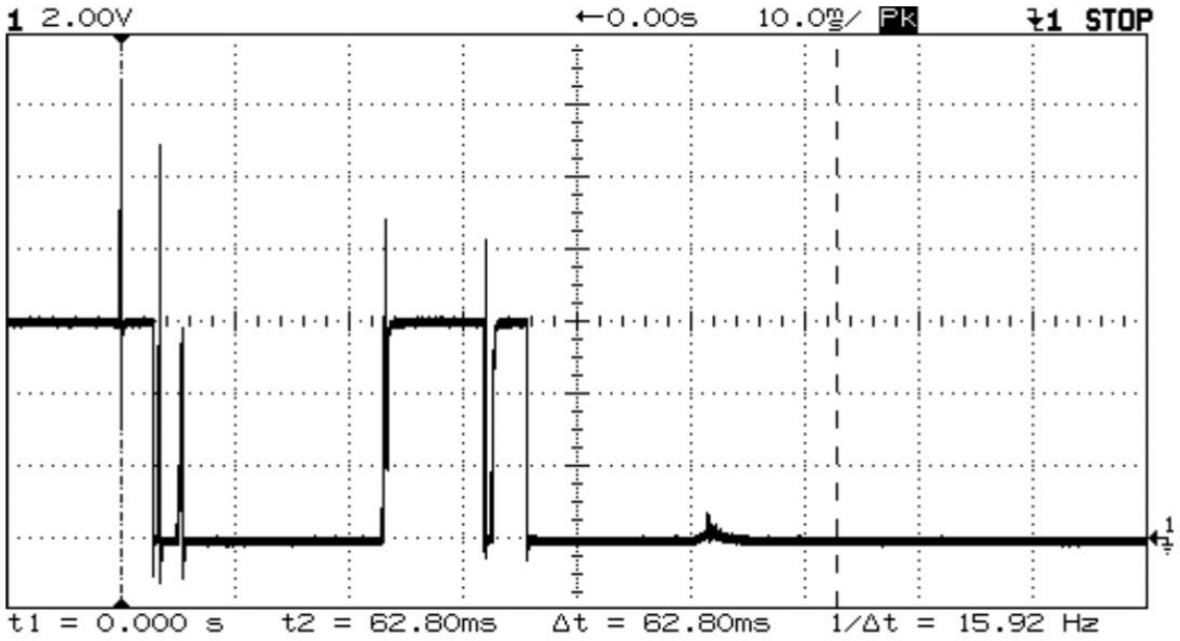

Fig. 2. Ejemplo del rebote mecánico de un botón.

Bool playSequence(eGameState_t gameState)

Muestra el patrón actual que refleja el estado del juego. Estos estados son los siguientes:

- **eWaitForStart:** Secuencia de walking-zero del LED mas a la izquierda al LED de mas a la derecha, actualizandose cada 500ms
- **eOngoingGame:** Secuencia de walking-one del MSB al LSB con rebote, actualizandose cada 300ms con decremento de 20ms en cada rebote.
- **eEnd:** Muestra el puntaje obtenido por el jugador (con un factor de escala de 0.5) sobre un solo LED, donde el LED de mas a la izquierda representa un puntaje de 2, y el de mas a la derecha de 14. Este LED deberá estar parpadeando con un patrón de 300ms de encendido y 50ms de apagado.
- **ePlayerInputState_t checkPlayerInput(eButtonState_t buttonState, eButtonId_t, buttonID):** Revisa el estado actual de los botones y valida que sean presionados en una ventana de tiempo correrecta. Esta ventana está dada de la siguiente manera, cuando la “pelota” vaya en sentido de los LEDs a la izquierda, el usuario debe presionar el botón de la izquierda cuando alguno de los dos LEDs del extremo esten encendidos, y misma racionalizacion en el otro sentido.

Nota: Es importante tener en cuenta que **ninguna de las dos funciones anteriores debe bloquear la tarea,** ya que ambas deben aparentar que están corriendo al mismo tiempo. Para lograr esto, hacer uso de la variable global de mili-segundos.

Conclusiones y comentarios

En conclusion, en este reporte aprendimos como funcionan las tecnicas de anti-rebote, así como a manejar de los GPIOs, uilizando lo aprendido para desarrollar un juego similar a “Pong” de 1-dimensión, inicializando los GPIOs. El desarrollo de este proyecto nos demuestrá como se debe manejar los GPIOs de un ESP32, lo cuál nos ayudará con las practicas siguientes.