Práctica 3

Manejo de la sección de E/S del microcontrolador ESP32

Objetivo: Mediante esta práctica el alumno analizará la implementación de retardos por software, así como también se familiarizará con la configuración y uso de puertos.

Equipo: - Computadora Personal y tarjeta de desarrollo del ESP32.
Teoría: - Técnicas de anti-rebote de botones
- Charlieplexing (aplicado para control de LEDs)

Desarrollo:

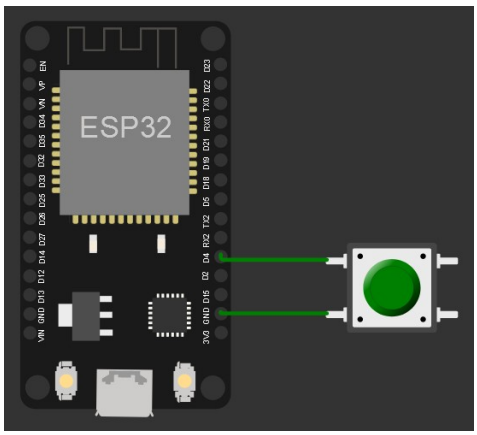
Leer el la guiá de inicio del ESP-IDF:
https://docs.espressif.com/projects/esp-idf/en/stable/esp32/get-started/index.html
Sabiendo de antemano que es necesario realizar la configuración de las herramientas de desarrollo, puesto que PlatformIO ya lo encapsula.

Familiarizarse con la tarjeta de desarrollo que han adquirido:
https://docs.espressif.com/projects/esp-idf/en/stable/esp32/hw-reference/esp32/get-started-devkitc.html
(*Buscar el esquemático correcto si es que compraron uno diferente*)

Leer sección de los periféricos de entrada y salida del SDK:
https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/gpio.html

Una vez realizada la lectura continuamos con otro ejemplo introductorio, el cual se encuentra dentro del folder /*print example*/. Dentro de este ejemplo podrán ver como se podria usar printf() al incluir <stdio.h> (y el SDK se encarga de conectar la tubería). Para poder ver las impresiones, es necesario el uso de una terminal (que ya fue cubierto en la primer lectura).

Así mismo, en este ejemplo también se muestra la configuración de un puerto de entrada al cual podemos conectar un botón y el LED de la tarjeta de desarrollo se encenderá mientras este presionado el botón. Ejemplo de la conexión:



Verificar la información del microcontrolador en la impresión del programa de ejemplo. Deberían de ver algo como:

```
This is esp32 chip with 2 CPU core(s), WiFi/BT/BLE, silicon
revision 0, 4MB external flash
```

Modificaciones a realizar:

Ahora pasamos al folder de */game/*, dentro de este ya encontraran la lógica de un pequeño juego de reacción de tiempo.

Antes de que empezar a modificar el código, van a ocupar hacer un cambio dentro de los archivos generados por PlatformIO. Abrir el archivo *.pio\build\esp-32s\config\ sdkconfig.h*, cambiar la linea:

```
#define CONFIG_FREERTOS_HZ 100
```

por:

```
#define CONFIG_FREERTOS_HZ 1000
```

Eso lograra que nuestros retardo puedan tener resolución de 1 mili-segundo.

Funciones a implementar:

1. `void initIO(void);`
   Inicialización requerida de los puertos utilizados en esta práctica, según la Fig. 2.

2. `uint8_t checkBtn(void);`
   Retorna el estado del botón, detectando entre *eBtnUndefined*, *eBtnShortPressed* y *eBtnLongPressed*. Donde el umbral para una larga duración es cualquiera que sea mayor a 1 segundo. Es importante ignorar el rebote mecánico que se puede apreciar en la Fig 1.

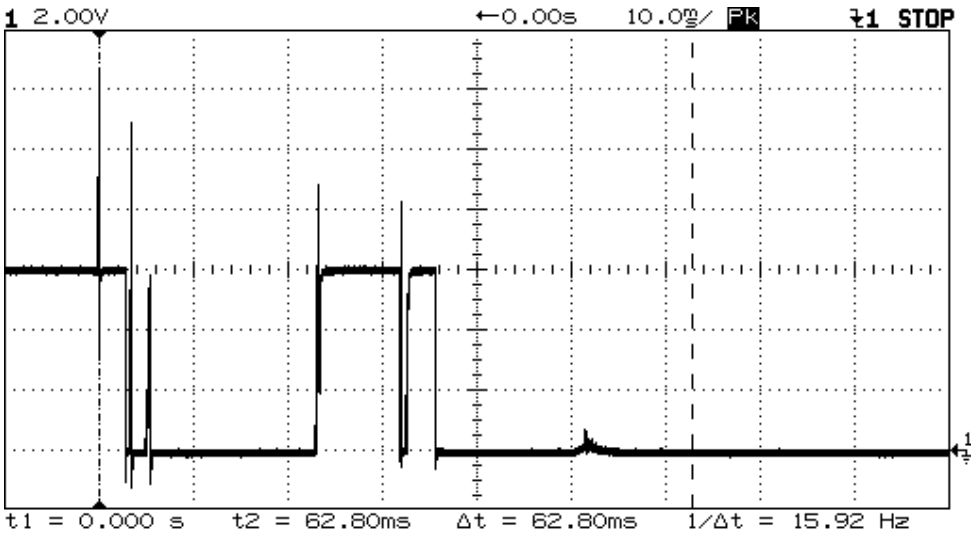


Fig. 1. Ejemplo del rebote mecánico de un botón.

3. `void updateLeds(uint8_t gameState)`
   Muestra el patrón actual que refleja el estado del juego. Estos estados son los siguientes:
   - ***eWaitForStart***:
     Sequencia de *walking-cero* del MSB al LSB, actualizándose cada 100 ms (s_0=0b11111111, s_1=0b01111111, …, s_8=0b11111110, y repetir)
   - ***eStartCount:***
     Secuencia aleatoria que se actualiza cada 100 ms.
   - ***eEndCount:***
     Todos los LEDs apagados.
   - ***eYouLoose:***
     Secuencia donde medio *nibble* esta apagado y el otro prendido, alternando cada 500 ms.
   - ***eYouWin:***
     Secuencia que alterna todos los LEDs se encienden y apagan simultáneamente alternando cada 250 ms.

   Realizar los ajustes necesarios de tal forma que **no** se perciba que parpadean.

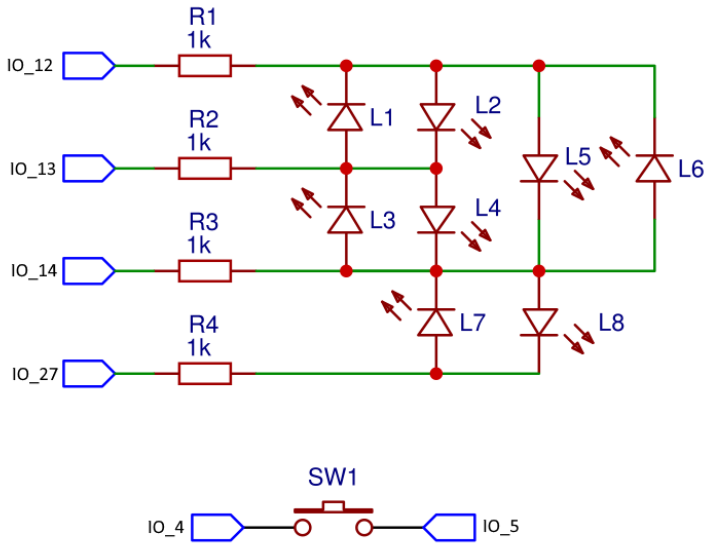


Fig. 2. Esquemático

Nota: Es importante tener en cuenta que ninguna de las dos funciones anteriores debe bloquear el proceso, ya que que ambas deben aparentar que están corriendo al mismo tiempo. Para lograr esto, hacer uso de la variable global de mili-segundos. No debería ser necesario tocar alguna otra parte del código, mas que solo esas dos funciones.

Comentarios y Conclusiones.

Bibliografía y Referencias.