UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

ELO 312
Laboratorio de Estructura de Computadores
Segundo semestre 2019

DEPARTAMENTO DE ELECTRONICA

0. Parte previa (*aún incompleta…*)

En la experiencia anterior se explicó que el **manejo de los pines de entrada y salida** de los puertos de propósito general (**GPIO**) es fundamental en el trabajo con micro-controladores, ya que nos permiten interactuar con el medio. Podemos medir o capturar eventos externos con pines de entrada y actuar sobre el medio mediante pines de salida. El manejo más básico que se puede hacer es tratar esos pines como entradas y/o salidas digitales (niveles lógicos "0" y "1").

Para realizar esta experiencia, **tenga siempre presente el "pinout"** diagrama de su tarjeta. Cree un nuevo proyecto para ***Atollic trueSTUDIO***, inicializando/configurando primero la tarjeta a través de **CubeMX**, a fin de trabajar **con los puertos de I/O de la tarjeta (GPIO)**.
…

*Para utilizar los **leds**, los **botones** y/o los **interruptores** de la **tarjeta I/O** que se les entregó junto con el resto de las cosas del KIT, asegúrese primero de conectar correctamente la placa y de alimentar correctamente la tarjeta I/O con **3.3V** y **GND** en los pines donde esto se indica.*

IMPORTANTE: En la subcarpeta "\Drivers\STM32L4xx_HAL_Driver\Src" que encontrará entre los archivos de su proyecto, generado por CubeMX, busque el archivo "stm32l4xx_hal_gpio.c". **NO LO MODIFIQUE**. Deje una copia a otra parte y abra el archivo con un editor de texto. Úselo como un *manual* para el manejo de los puertos de entrada y salida. En la sección "IO operation functions" encontrará la documentación de las funciones que le serán de mucha utilidad en esta experiencia:

- HAL_GPIO_ReadPin(·)
- HAL_GPIO_WritePin(·)
- HAL_GPIO_TogglePin(·)
- HAL_GPIO_EXTI_Callback(·)

1. En el Laboratorio

1.1. Trabajar con GPIO e interrupciones de los puertos

1.1.1. Configure un pin de un puerto de su elección como entrada y otro pin del mismo puerto como salida.

1.1.2. Conecte el pin de salida a uno de los leds de la tarjeta I/O y el pin de entrada a un switch de la tarjeta I/O. Escriba un programa que, dependiendo del estado lógico leído en el pin de entrada, haga que el ciclo de trabajo del led cambie, por ejemplo, de 50% a 25%.

1.1.3. Muestre la señal generada en la salida en el osciloscopio o analizador lógico *SALEAE*, que solicitará a Pañol para utilizarlo durante el laboratorio.

1.1.3.1. Aproveche de medir la frecuencia en la señal de salida.

1.1.4. Siguiendo las indicaciones de su profesor, en CubeMX define el mismo pin que antes utilizó como entrada como "EXT_INTX", habilite las interrupciones de este pin *en el canto de bajada y de subida* de la señal, de modo que se pueda generar una interrupción cada vez que cambien el estado del switch. Además, define ahora 8 pines del mismo puerto como salida y conéctelos a los leds de la tarjeta I/O.

1.1.4.1. Escriba una función `void f1_port (void)` que haga "algo interesante" (póngase creativo) con los pines del puerto que seleccionó como salida, y otra función `void f2_port (void)` que haga algo "también interesante" (pero diferente de la otra función) con los pines de salida del puerto.

1.1.4.2. IMPORTANTE: Siguiendo las indicaciones de su profesor, en otro archivo llamado "my_it_callbacks.c" que agregará al proyecto, define la función `HAL_GPIO_EXTI_Callback()`, que es la función que se ejecutará cada vez que se genera una interrupción por el pin de entrada del puerto. Lo que debe hacer esta función es cambiar el puntero a función `void (*function_pointer) (void)` de modo de definir cuál de las dos funciones antes mencionadas se va a ejecutar en el while loop de `main(·)`.

1.1.5. Muestre el funcionamiento correcto de su programa en los leds de la tarjeta I/O.

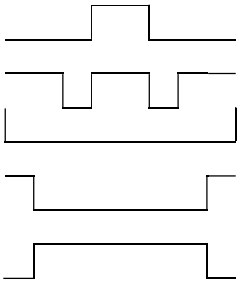
1.2.Trabajar con GPIO e interrupciones de los termporizadores (TIMERS)

1.2.1. Siguiendo las indicaciones de su profesor, configure el temporizador 2 (TIMER 2) del microcontrolador para operar con el reloj interno (Clock source → internal clock) y contar en modo UP, y habilite las interrupciones.

1.2.1.1. IMPORTANTE: Siguiendo las indicaciones de su profesor, en el mismo archivo "my_it_callbacks.c" que agregó antes al proyecto, define también la función `HAL_TIM_PeriodElapsedCallback()`, que es la función que se ejecutará cada vez que se genera una interrupción por el timer. Dicha función ahora ejecutará lo que en el punto anterior se ejecutaba en el while loop de `main(·),` el cual en este punto tiene que estar vacío. Idealmente, elimine este while loop de `main(·)` y en su lugar escriba un código que deje la CPU del microcontrolador en modo de bajo consumo.

1.2.2. Muestre el funcionamiento correcto de su programa en los leds de la tarjeta I/O.

1.2.3. Escriba un código en la función `HAL_TIM_PeriodElapsedCallback()` que, con cierta periodicidad, actualice cada el estado de los 8 pines de salida de modo que, en función del tiempo se pueda reconocer "algún patrón interesante" (póngase creativo), como el que se ve en la siguiente figura (que no tiene nada interesante). Aquí se muestran como ejemplo diferentes conjuntos de valores que se aplican a 5 pines de salida



1.2.4. Muestre la señal generada en las 8 salidas en el osciloscopio o analizador lógico *SALEAE*, que solicitará a Pañol para utilizarlo durante el laboratorio.