**PRÁCTICA 2: Traffic Light + Moore FSM**

**FINALIDAD**

El objetivo de la práctica es diseñar un circuito referente a un semáforo, tanto de vehículos como de peatones, así como el funcionamiento del semáforo mediante una máquina estados finita (FSM) de Moore, usando para ello el lenguaje C, el software STM32CubeMX junto al IDE de desarrollo VSCode + Platform IO.

**OBJETIVO**

El funcionamiento que se busca simular es la siguiente secuencia:

El paso de vehículos estará en verde y únicamente se pondrá en rojo cuando se pulse el botón. El flujo de eventos es el siguiente:

• Pulsar botón del semáforo

• Esperar 3 segundos

• Pasar el control de vehículos del verde al amarillo en la calle principal

• Esperar 3 segundos

• Poner el semáforo de vehículos en rojo y el de peatones en verde.

• Espera 15 segundos

• Poner semáforo de peatones en verde intermitente

• Esperar 3 segundos

• Poner semáforo de peatones en rojo y el principal en verde

**Esquema STM32 CubeMX**

Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

**Esquema del circuito (en aplicación Fritzing)**

Imagen que contiene Esquemático

Descripción generada automáticamente

**Componentes**

- Microcontrolador SMT32F-F411RE

- Placa de expansión Base Shield Seeed Groove V2

- 1 x LED amarillo

- 2 x LED rojo

- 2 x LED verde

- Pulsador con conector de 4 pines

- Distintos cables macho-macho

- Conector con 4 pines (para botón)

- Resistencias (2x 75Ω, 2x 220Ω, 1 x 330Ω)

**Máquina de estados finita de Moore**

Diagrama

Descripción generada automáticamente

Dónde las variables que inician por C se asocian a las propias del semáforo de los vehículos mientras que el que inicia por P es el referente al de los peatones.

**Solución propuesta**

El código desarrollado se implementa en el main, por lo que es ejecutado por la CPU. En esta ocasión, no es necesario introducir interrupciones (práctica 3), por lo que el código es bastante sencillo y legible.

Hemos continuado con los accesos a registros tal y como hicimos en la práctica 1.

Texto

Descripción generada automáticamenteLa lógica del código es sencilla: si el botón se presiona comenzará la secuencia propia del comportamiento de dos semáforos, de vehículos y peatones.

En primer lugar, se parte del estado en el que el LED rojo de peatones (D2) y el verde de vehículos (D13) están activados. Tras 3 segundos, se apagará el LED verde (D13) y se encenderá el amarillo (D12). Al pasar nuevamente 3 segundos, activaremos el paso de los peatones encendiendo el LED verde de los peatones (D4) y el LED rojo de los vehículos (D11). Al pasar 5 segundos, pasamos a la función “task\_parpadeo()” que exponemos a continuación.

En el else, se reestablece el estado “normal” del programa antes de que el botón sea pulsado.

MÉTODO PARPADEO

Este método es el utilizado en el último estado, dónde el LED de los vehículos está en rojo y el de los peatones está en verde pero parpadeante. Para ello, hemos utilizado un bucle for (para realizar dos repeticiones del kernel del bucle y evitar que sea demasiado extenso) que repetirá la acción de encendido y apagado del pin D4, propio del LED verde de los peatones.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamentePINES Y PUERTOS GENERADOS POR STM32 CUBEX