Laboratorio 2:
GPIOs, Timers y FSM

Jafet Soto Arrieta, B77543
Universidad de Costa Rica, Escuela de Ingeniería Eléctrica
IE-0623 Laboratorio de Microcontroladores
11 de abril del 2024
jafet.soto@ucr.ac.cr

***Resumen*—**

Esta practica se centra en desarrollar un cruce de semáforos simplificado utilizando GPIOs, temporizadores y máquinas de estado finito (FSM) para el microcontrolador ATtiny4313.

I. Introducción

Para elaborar el simulador de semáforos primero debemos tomar en cuenta el sistema que detecta la condición de inicio; se diseñará un sistema de 2 botones con un circuito de debouncing similar al que se aplicó en el laboratorio pasado; además, el inicio de la ejecución debe ser activado por el sistema de atención de interrupción del microprocesador.

El sistema de salida estará compuesto por 7 LEDs:

- 3 LEDs para representar el semáforo vehicular.
- 4 LEDs para los 2 semáforos peatonales, 2 LEDs por semaforo.

Figura 1: Diagrama del diseño del sistema se simulación de semáforos.

Se debe desarrollar el firmware que permita generar las señales requeridas para la practica. Utilizando los timers del sistema, y desarrollando la rutina de atención de interrupciones adecuada.

II. Nota teórica

II-A. Características generales del ATtiny4313

El ATtiny4313 es un microcontrolador AVR de 8 bits con alto rendimiento y bajo consumo de energía. Utiliza una arquitectura RISC avanzada que permite ejecutar la mayoría de las instrucciones en un solo ciclo de reloj, alcanzando hasta 20 MIPS a 20 MHz. Posee memorias de datos y programa no volátiles, periféricos como temporizadores, PWM, comparador analógico y UART, además de características especiales como depuración en chip y modos de bajo consumo. Está disponible en diversos encapsulados con 18 líneas de E/S programables y opera en un amplio rango de voltajes y temperaturas industriales. Su consumo de energía es notablemente bajo en diferentes modos de operación.

II-B. Diagrama de bloques

El microcontrolador combina un conjunto de instrucciones eficiente con 32 registros de trabajo de propósito general, ofreciendo un rendimiento hasta diez veces más rápido que los microcontroladores convencionales. Con características como memoria Flash y EEPROM programables en el sistema, temporizadores flexibles, interfaz serial y modos de ahorro de energía seleccionables. Además, es compatible con una amplia gama de herramientas de desarrollo que facilitan la programación y la depuración del sistema.

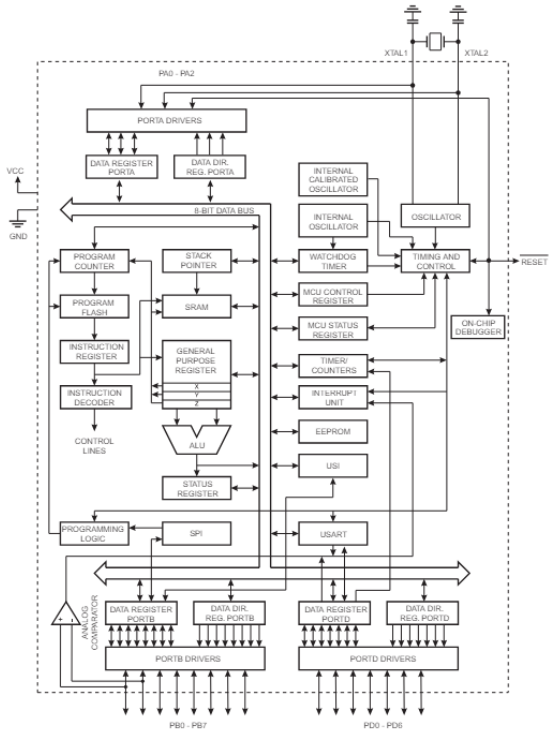

Figura 2: Diagrama de bloques del microcontrolador ATtiny4313. [1]

II-C. Diagrama de pines

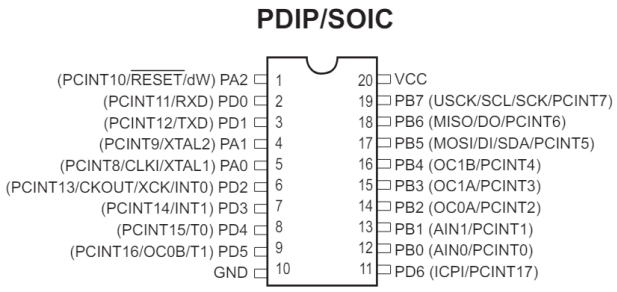

Figura 3: Diagrama de pines del microcontrolador ATtiny4313. [1]

- **VCC**: Voltaje de alimentación digital.
- **GND**: Tierra.
- **Port A (PA2-PA0)**: Puerto A de E/S bidireccional de 3 bits con resistencias de pull-up internas. PA2 también funciona como pin de RESET.
- **Port B (PB7 -PB0)**: Puerto B de E/S bidireccional de 8 bits con resistencias de pull-up internas.
- **Port D (PD6-PD0)**: Puerto D de E/S bidireccional de 7 bits con resistencias de pull-up internas.
- **RESET**: Entrada de reset. Un nivel bajo durante más tiempo que el pulso mínimo generará un reset.
- **XTAL1**: Entrada al amplificador oscilador inversor y al circuito de operación de reloj interno.
- **XTAL2**: Salida del amplificador oscilador inversor.

En la notate´orica debe incluir la informaci´on del microcontrolador(caracter´ısticas generales, diagrama de bloques, diagrama de pines y caracter´ısticas el´ectricas), perif´ericos utilizados (esto incluye descripci´on de registros e instrucciones seg´un aplique), componentes electr´onicos complementarios; as´ı como tambi´en el dise˜no del circuito justificando los valores o funci´on de los componentes electr´onicos/digitales utilizados (debe incluir una lista de la cantidad de componentes y sus precios) e informaci´on de los conceptos fundamentales adicionales que se ven en clase.

III. Desarrollo y análisis.

El sistema de detección de la condición de inicio se implementa mediante el puerto D2, el cual activa el sistema de interrupciones para el registro de atención de interrupciones externas INT0. Durante esta etapa inicial, se ha diseñado un circuito simplificado de debouncing para garantizar una detección precisa de la señal de inicio, ignorando cualquier posible ruido que pueda afectar el sistema.

Al presionar ambos botones, se establece una conexión que lleva el pin D2 a un nivel de voltaje de +5V. Es relevante mencionar que estos pines tienen una capacidad máxima de corriente de 40 mA, aunque se ha decidido limitarla a 30 mA para asegurar un funcionamiento estable y seguro.

Para determinar el valor de las resistencias que acompañan a los LEDs, se ha utilizado la Ley de Kirchhoff, considerando que la tensión de los LEDs es de 2.0 V. Por lo tanto, para un solo LED, la resistencia se calcula como R = (VDD - VLED) / IMax, donde VDD es la tensión de alimentación (5V) y IMax es la corriente máxima (25 mA). Esto nos lleva a un valor de resistencia de 120Ω.

Finalmente el circuito diseñado es el siguiente:

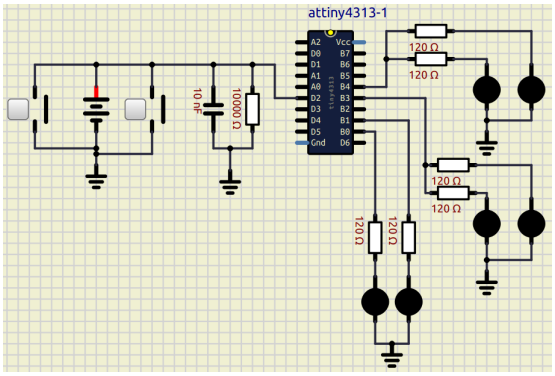

Figura 4: Circuito diseñado para el laboratorio 2.

Se plantea manipular los LEDs de los semáforos por cuatro pines diferentes, el puerto PB1 representa la señal Roja del semáforo de carros, el puerto PB2 controla el estado de la luz verde del semáforo de carros, lo semáforos peatonales se representan de forma paralela, manipulando las luces rojas, de ambos, por el puerto PB3 y las luces peatonales verdes por PB4.

El código desarrollado implementa un simulador de semáforos utilizando el microcontrolador AVR ATtiny4313. Comenzamos por incluir las librerías necesarias, como avr/io.h y avr/interrupt.h, para el manejo de los registros del microcontrolador y las interrupciones, respectivamente.

Luego, se definen los estados de la Máquina de Estado Finito (FSM) en la enumeración Signal_State. Estos estados representan las diferentes etapas del funcionamiento del semáforo, como el estado de espera (IDL), configuración (SETUP), detención de vehículos (STOP_V), activación del sistema (SYSTEM_ON), desactivación (DEACTIVATE) y apagado del sistema (SYSTEM_OFF).

El programa utiliza dos variables importantes: Time para contar el tiempo transcurrido en cada estado y TIMER_COUNTER para contar el tiempo transcurrido desde la última interrupción del temporizador.

La función configureInterrupts() se encarga de configurar los pines de entrada/salida, las interrupciones externas (INT0) para detectar el botón de inicio, y el temporizador (TIMER0) para generar interrupciones periódicas.

En la función FSM(), se implementa la lógica de la Máquina de Estado Finito. Dependiendo del estado actual, se realizan diferentes acciones, como cambiar los estados de los LED

para simular el funcionamiento del semáforo y reiniciar los contadores de tiempo cuando sea necesario.

La función BLINK_LEDs() se encarga de realizar el parpadeo de los LED según ciertas condiciones, como el estado actual de la FSM y el tiempo transcurrido desde la última interrupción del temporizador.

Sin embargo, a pesar de los esfuerzos realizados en el diseño y desarrollo del firmware, se encontraron dificultades en la implementación práctica. Al ejecutar el firmware en SimulIDE, se encontró un mensaje de error İnterrupts::retI Error: No active Interrupt", seguido de la repetición continua de este mensaje en la terminal. Además, al presionar el botón de inicio, el programa colapsaba y se mostraba el mensaje "Segmentation fault (core dumped).[en] la terminal.

Estos problemas impidieron la correcta ejecución del laboratorio y la validación del simulador de semáforos diseñado. Se requiere una revisión adicional del firmware y una depuración más exhaustiva para identificar y corregir las causas de estos errores.

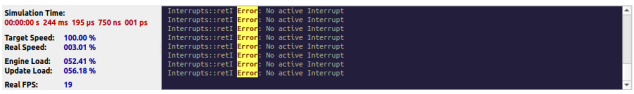

Figura 5: Mensaje de de error en la terminal del simulador.

IV. Conclusiones y recomendaciones

En este laboratorio, se realizó un exhaustivo trabajo para implementar un simulador de semáforos utilizando el microcontrolador ATtiny4313. Se diseñaron y configuraron los circuitos necesarios, se desarrolló el firmware correspondiente y se aplicaron técnicas de debouncing para garantizar un funcionamiento preciso y confiable del sistema.

Sin embargo, a pesar de los esfuerzos dedicados, lamentablemente no fue posible completar la implementación debido a problemas de ejecución encontrados durante las pruebas en el simulador. Se identificaron errores que requieren una depuración más detallada del código para resolverlos y lograr el funcionamiento esperado.

A pesar de este contratiempo, es importante destacar que el trabajo realizado hasta el momento abarca de manera óptima todos los aspectos necesarios para la implementación del simulador de semáforos. Se han diseñado circuitos adecuados, se ha desarrollado el firmware necesario y se han aplicado técnicas de debouncing para garantizar la fiabilidad del sistema.

Como recomendación para trabajos futuros, se sugiere dedicar más tiempo a la depuración del código y a la identificación precisa de posibles errores. Además, es importante llevar a cabo pruebas exhaustivas en diferentes entornos y condiciones para asegurar un funcionamiento robusto del sistema. Con un enfoque riguroso en la depuración y pruebas adicionales, se espera poder superar los obstáculos encontrados y lograr la implementación exitosa del simulador de semáforos en el microcontrolador ATtiny4313.

Referencias

[1] *ATtiny2313A / ATtiny4313 Datasheet. (s.f.). Microchip Technology Inc. Recuperado de https://www.microchip.com*