Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE-0624 Laboratorio de Microcontroladores
II ciclo 2022

Laboratorio 2

GPIOs, Timers y FSM

Juan Ignacio Montealegre Salazar B95001
Luis Javier Herrera B93840

Profesor: Marco Villalta Fallas

Martes 20 de septiembre del 2022

Índice

Índice de figuras

Índice de tablas

1. Resumen

En esta práctica de laboratorio se desarrolla un cruce de semáforos simplificado utilizando LEDs, botones y el microcontrolador ATtiny4313, de tal forma que se representará una vía vehicular y un pasa peatonal. El semáforo vehicular posee un total de tres leds de colores verde, amarillo y rojo para regular el flujo de vehículos. Por otro lado, el paso peatonal posee dos semáforos, uno en cada lado, donde cada semáforo posee un botón para solicitar cruzar, así como dos luces de color verde y rojo para permitir o no el paso de peatones. Para ello, se utilizan interrupciones y temporizadores internos del microcontrolador. Además, se trabaja planteando el sistema como una máquina de estados. Con todo lo anterior, se diseña el semáforo en SimulIDE, donde también se simula y se comprueba su comportamiento.

Palabras clave: *temporizador, interrupciones, microcontrolador, semáforo, SimulIDE*

2. Nota teórica

En esta sección se describe el microcontrolador utilizado en esta ocasión, así como cada periférico utilizado (registros e instrucciones/funciones), componente electrónico complementario y el diseño del circuito final. El proceso de pruebas realizadas para llegar al circuito final se abarca en la sección "Desarrollo y Análisis de Resultados".

2.1. Microcontrolador ATtiny4313

Para la práctica de laboratorio 2 se trabajó con el microcontrolador ATtiny4313 del fabricante Atmel. Este es un microcontrolador de 8 bits que cuenta con alto rendimiento y bajo consumo de potencia. Posee con un total de 120 instrucciones donde la mayoría se ejecutan en solamente un ciclo de reloj y está basado en la arquitectura RISC (Reduced Instruction Set Computer). Además, este tiene una memoria de tipo flash de 2/4 Kbytes, una memoria EEPROM programable de 128/256 Bytes y una memoria SRAM interna de 128/256 Bytes. Un aspecto distintivo de este microcontrolador es que posee dos contadores de 8 y 16 bits con 'prescaler' separado y modos de comparación. También, tiene la capacidad del manejo de interrupciones ya sea por medio de los contadores y por medios externos o internos [1]. En la Figura 1 se puede ver la distribución de pines del microcontrolador.

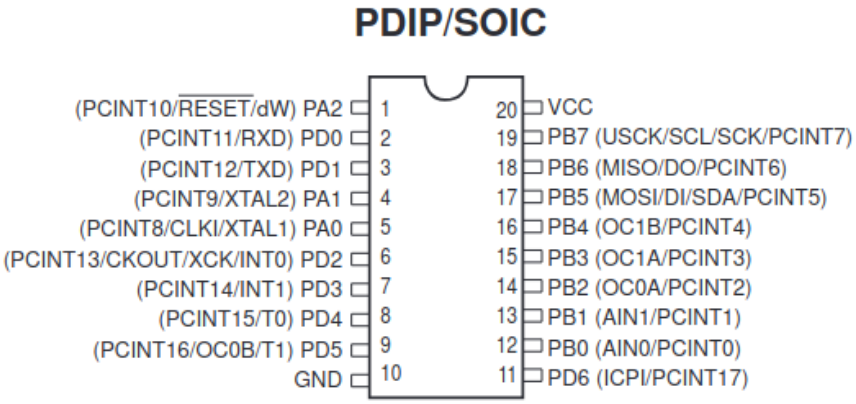


Figura 1: Distribución de pines del ATtiny4313 [1]

Como se aprecia en la Figura 1, el ATtiny 4313 cuenta con 18 pines programables de tipo GPIO y cuenta con un voltaje de operación que varía entre los 1.8 y 5.5 V. Los pines que llevan el prefijo 'INT' en sus nombres pueden ser usados para el manejo de interrupciones. A nivel

de software, las interrupciones consisten en pequeñas subrutinas a realizar donde el microcontrolador detiene todo lo que estaba haciendo para ejecutar la interrupción. Estas pueden ser activadas por cambios de estado en alguno de los pines disponibles en el chip. Además, este tiene la capacidad de operar a frecuencias de 4 MHz, 10 MHz y 20 MHz donde dependiendo de la frecuencia, su límite inferior de voltaje de operación puede llegar a variar con esta. En la Figura 2 se puede apreciar el diagrama de bloques interno del microcontrolador estudiado en el laboratorio. Se destacan las partes más importantes que lo componen como la memoria FLASH, la ALU, el oscilador, registros y temporizadores.

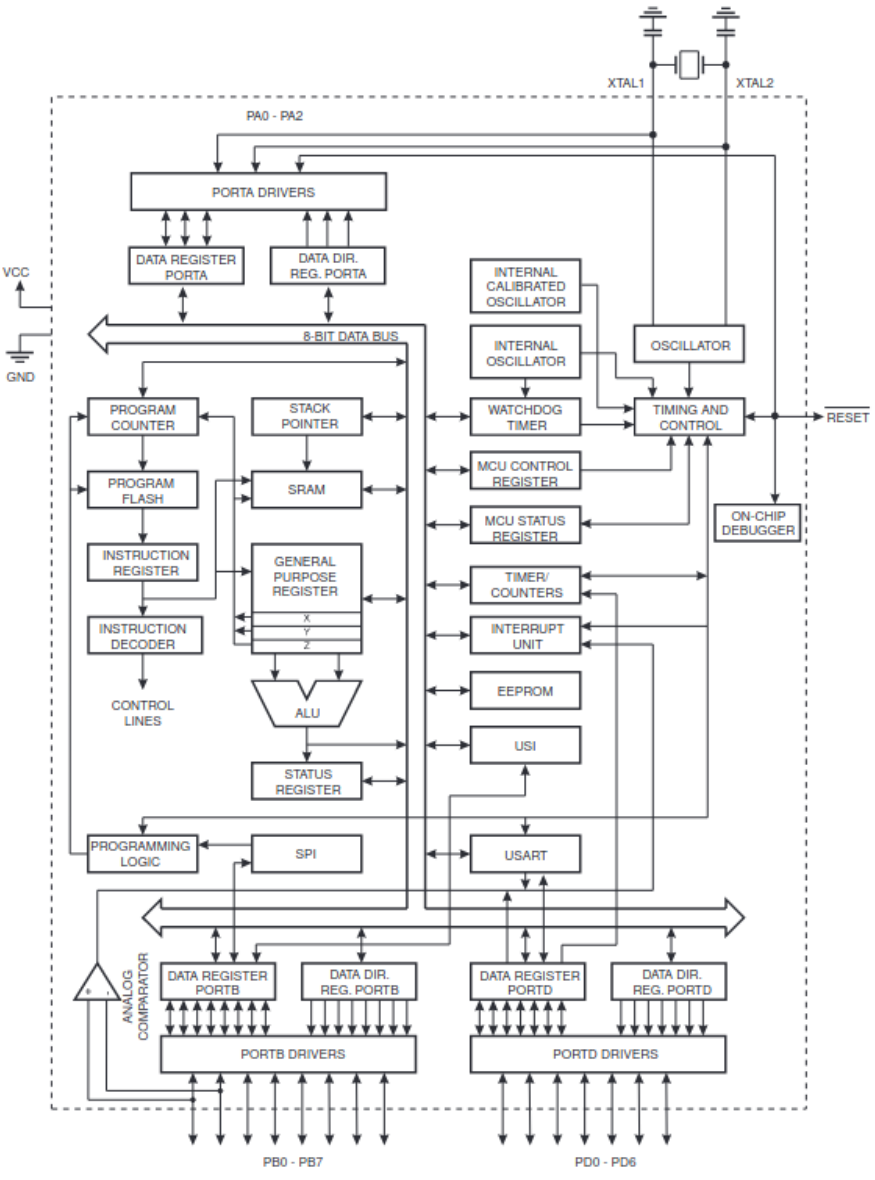


Figura 2: Diagrama de bloques del microcontrolador ATtiny4313 [1].

En la Figura 3 se observa el banco de registros de uso general del CPU del microcontrolador. Prácticamente todas las instrucciones tiene acceso completo a estos registros y todas se ejecutan en un solo ciclo de reloj [1]. Son un total de 32 registros de uso general donde a cada uno de estos se les asigna una dirección de memoria con el fin de tenerlos mapeados.

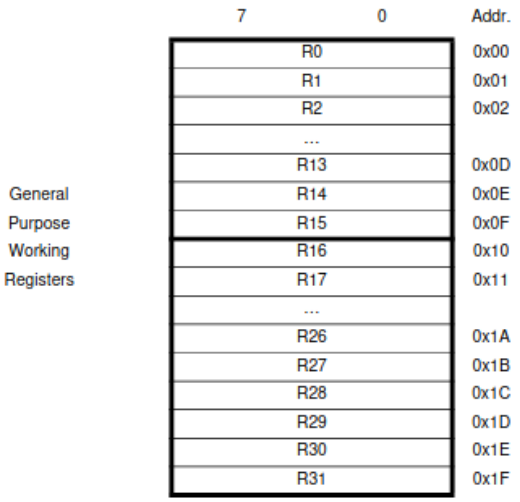


Figura 3: Banco de registros generales del microcontrolador ATtiny4313 [1].

2.2. Componentes electrónicos adicionales

2.2.1. LEDs de colores

Para este laboratorio se utilizaron diodos de tipo LED de colores rojo, amarillo y verde para simular los mismos colores que tiene un semáforo. El funcionamiento de estos es bastante simple, se cuenta con dos terminales: el cátodo y el ánodo. Si se crea una diferencia de voltaje en el cual el cátodo se encuentra a un mayor voltaje que el ánodo, el diodo va a emitir luz. Caso contrario se va a encontrar apagado. Dependiendo del color que se quiera emitir estos van a tener diferentes rangos de operación en el voltaje.

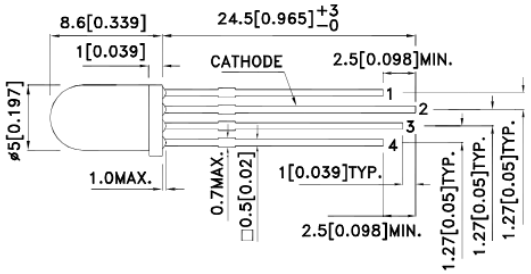


Figura 4: Diagrama de pines de LED básico [2]

2.3. Capacitores

Los capacitores son dispositivos electrónicos bastante comunes, estos se utilizan para almacenar energía dentro de un campo eléctrico interno. Es un componente de tipo pasivo y su estructura se compone de dos placas conductores separadas por un material dieléctrico. Para el caso del presente laboratorio estos se utilizan para controlar el efecto de 'rebote' que se genera a la hora de presionar el botón.

2.4. Diseño de Circuito

Luego de realizar diversas pruebas y cambios en el diseño, se llega al diseño del circuito que se muestra en la siguiente figura.

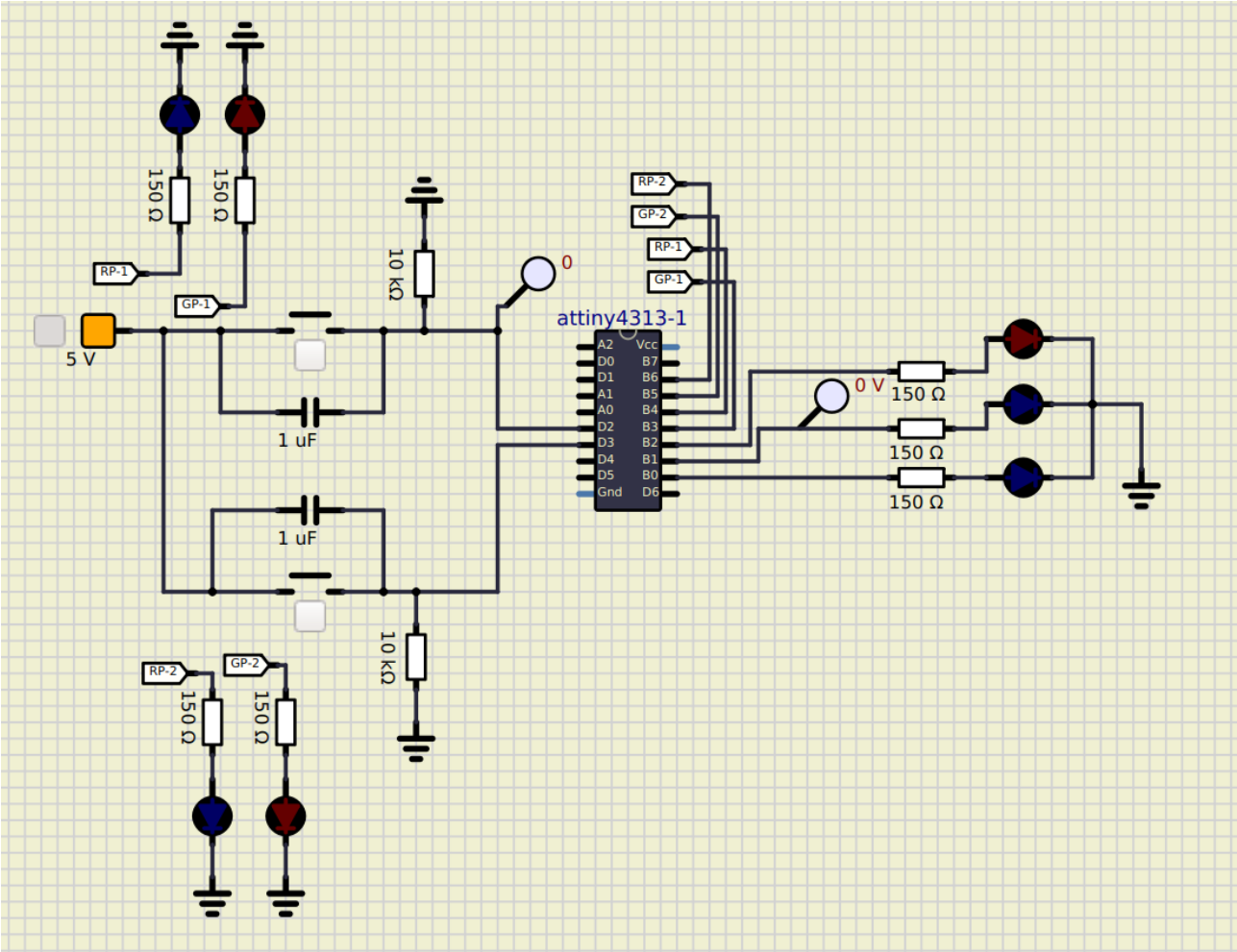


Figura 5: Diseño de circuito final en SimulIDE

Primeramente se tuvo que diseñar el valor de resistencia para el resistor de pull-down. Este se utiliza como modo de complemento al botón pulsador para asegurar que cuando el botón se pulse, se propaguen los 5 V al pin de entrada del microcontrolador. Este resistor se va a encargar de mantener el pin de entrada bajo hasta el momento en que se active el respectivo botón y se desee poner la entrada en alto. Se elige entonces un valor grande para este resistor de $R_p = 10$ kΩ con el fin de asegurarse por completo de que este valor va a ser mucho mayor que el valor de resistencia interna del botón R_{int}. Con esto presente, al presionar el botón:

$$V_{Rp} = \frac{5V \cdot 20k\Omega}{(20k + R_{\text{int}})} \longrightarrow R_p \gg R_{\text{int}} \tag{1}$$

Pero R_{int} se toma como despreciable, entonces:

$$V_{Rp} \approx \frac{5V \cdot 10k\Omega}{10k\Omega} = 5V = V_{pin\_entrada} \tag{2}$$

Con respecto a los valores de resistencia para los resistores que se conectan en cada LED de colores se tomó en cuenta que el valor típico de corriente máximo que soportan los LEDs es de alrededor de 150 mA con un voltaje de alimentación máximo de 5 V según las hojas de

fabricante. Como se puede puede evidenciar en la Figura 6, con un resistor de 150 Ω para cada uno de los LEDs se tiene una corriente de 13.64 mA con un voltaje de 4.45 V. Estos valores indican entonces que los LEDs se encuentran bastante seguros en caso de que se pueda superar sus máximas capacidades y posiblemente quemarse.

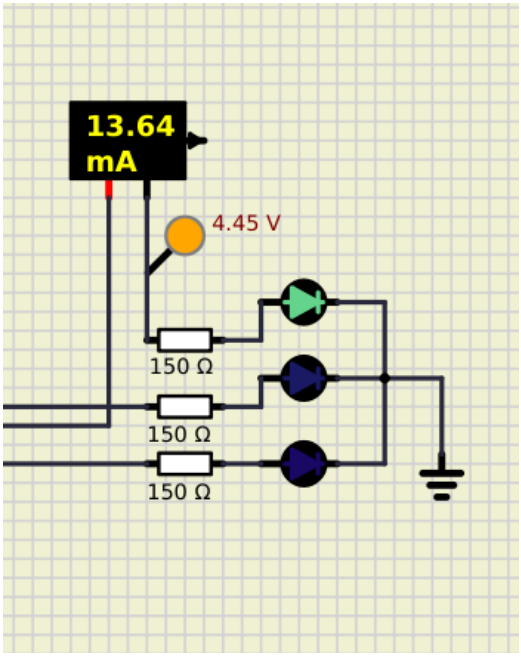


Figura 6: Mediciones de corriente y voltaje para LEDs.

2.4.1. Lista de componentes

La lista de componentes utilizados en el experimento se muestra en la Tabla 1.

Componente	Tipo	Precio	Cantidad
Resitor (10K Ω)	-	$ 0.02	2
Resistor (150 Ω)	-	$ 0.07	7
Microcontrolador	ATtiny4313	$ 1.1850	1
Led	Amarillo	$ 0.015	1
Led	Rojo	$ 0.03	3
Led	Verde	$ 0.03	3
Botón	-	$ 0.5	2
Capacitores (1μF)	-	$ 5.12	2
Total		$ 6.97	21

Tabla 1: Lista de componente utilizados en el laboratorio.

3. Desarrollo y Análisis de Resultados

En esta sección se comenta de forma detallada el desarrollo del proyecto. Primero se explica el desarrollo del circuito como tal, y luego el diseño del programa a partir del programa deseado.

3.1. Desarrollo del circuito

El circuito propuesto para resolver el problema del laboratorio se conforma principalmente de dos secciones. La primera es la parte de acción donde el usuario debe accionar el botón para lograr que sucedan los cambios de luces indicados. Esta se muestra en la Figura 7.

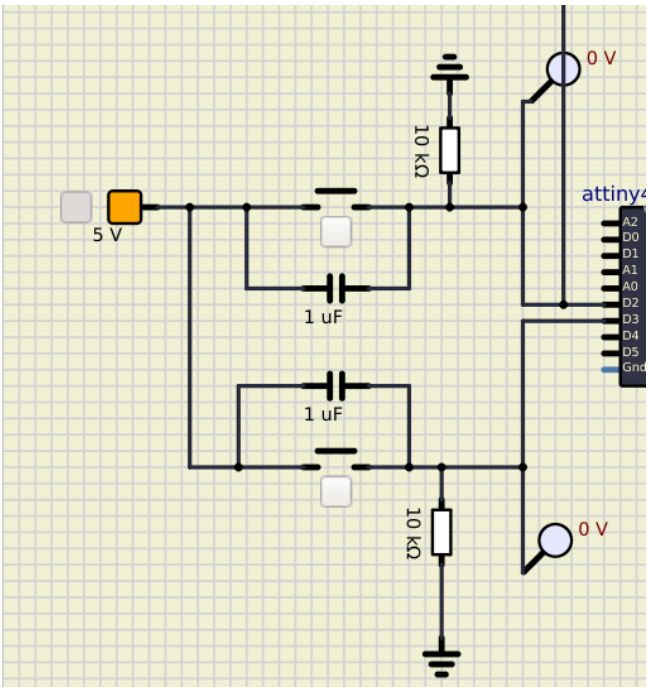


Figura 7: Sección del circuito donde se encuentran los botones para accionar el cambio de colores en los LEDs.

Cada uno de los botones cuenta con su resistor en configuración de pull-down para lograr pasar buenos ceros lógicos, así como estar en paralelo con un capacitor que funciona como filtro para evitar el efecto de rebote, y además se utilizaron las terminales D3 y D2 del microcontrolador como entradas para leer el voltaje cuando se quiera pasar al estado de paso peatonal. Esto se realizó a conveniencia ya que justo estas terminales cuentan con la posibilidad de accionar una interrupción al leer un cambio de estado ya sea por flanco creciente o decreciente. La otra parte del circuito se caracterizaba más por sus características visuales donde dependiendo del color de los LEDs encendidos se estaba en los estados de paso vehicular, transición de estados o paso peatonal. Para estos de utilizaron las terminales desde la B0 hasta la B6 del microcontrolador en configuración como salidas. Esta sección se puede apreciar en la Figura 8.

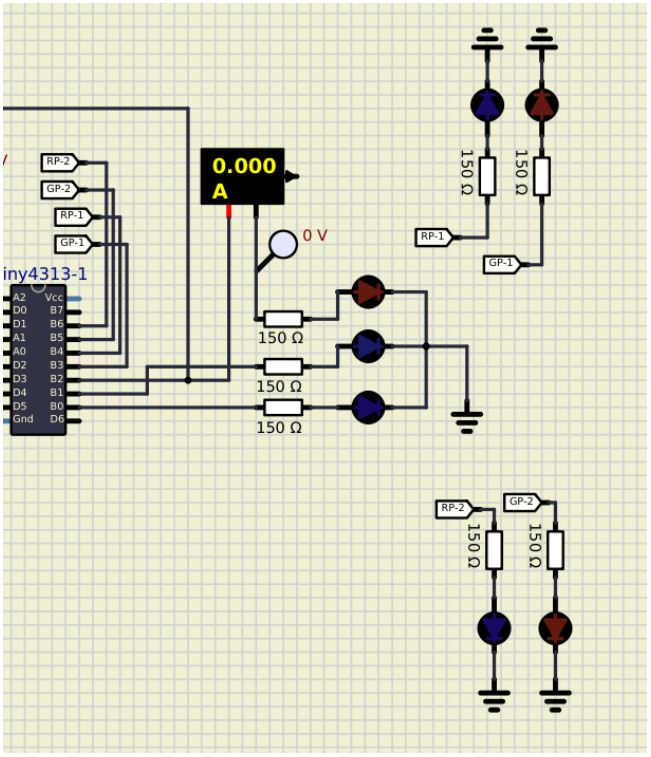


Figura 8: Sección del circuito para visualizar el color de los LEDs y saber en qué estado se encuentra.

Como forma de demostración de la funcionalidad del circuito se colocó osciloscopio para medir los voltajes en el botón justo en el momento cuando se presiona el mismo y en LED color verde que indica el paso vehicular. En la Figura 9 se observan estas mediciones. La señal amarilla es el voltaje medido en el botón a lo largo del tiempo y la señal blanca es la salida de voltaje que le llega al LED verde de paso vehicular.

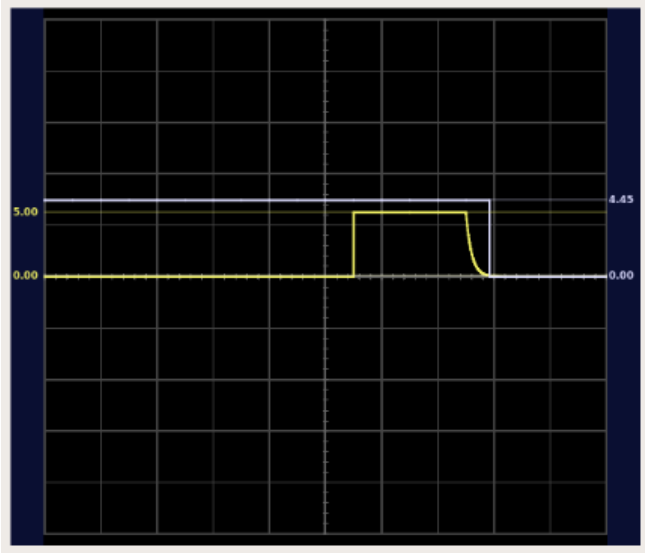


Figura 9: Funcionamiento del circuito en el osciloscopio.

Se aprecia en la Figura anterior justo en el momento cuando se presiona el botón la luz verde empieza la secuencia de parpadeo para indicar que se va a pasar al estado de paso peatonal. Luego, es esperara el debido tiempo utilizando los contadores internos del microcontrolador para volver de nuevo al estado vehicular hasta que alguien presiones el botón.

3.2. Desarrollo del programa

Antes de comenzar con el desarrollo del programa, el cual se elaborará bajo el lenguaje de programación C, es importante realizar un diagrama de flujo para tener presente los casos y estados que implican el proceso que se quiere implementar.

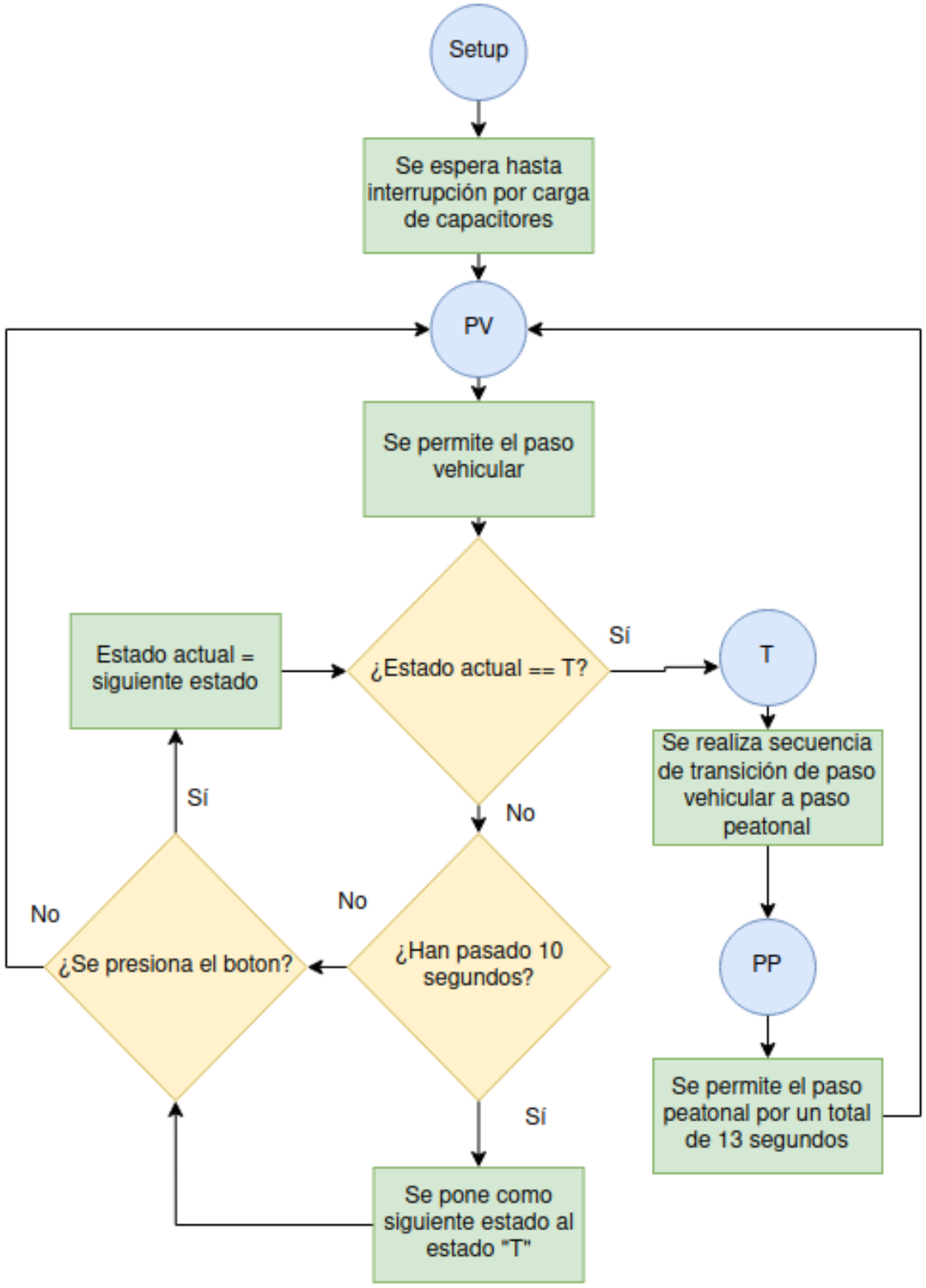


Figura 10: Diagrama de flujo del proyecto

Como se puede apreciar en el diagrama de flujo, el programa se trabajará planteándolo como una máquina de estados con un total de 4 estados, donde estos son:

- Setup: estado de inicialización. Todas las salidas están en bajo
- PV (Paso Vehicular): estado donde se permite el paso vehicular
- T (Transición): estado donde se realiza una secuencia de transición de PV a PP.
- PP (Paso Peatonal): estado donde se permite el paso de peatones

El programa inicia por defecto en el estado de Setup, donde todas las entradas se mantienen en bajo. Este estado se crea ya que, debido a que los botones están en configuración de pull-up y en paralelo con capacitores, estos mientras se cargan conducen corriente y por lo tanto ponen en alto las entradas, lo cual se toma como una interrupción. Este estado es necesario para que la interrupción inicial no altere el funcionamiento del estado PV de paso vehicular. Es importante mencionar que las interrupciones asociadas a los botones del circuito se encargan de cambiar el estado actual de la máquina de estados.

Una vez que sucede esta interrupción, se cambia incondicionalmente al estado PV, donde se pone en alto la salida del GPIO asociado a la LED verde del semáforo vehicular, así como las LEDs rojas de ambos semáforos peatonales. Además, se agrega como variable de próximo estado al estado T. Este comportamiento se mantiene hasta detectar una interrupción en los pines GPIO asociados a los botones, es decir hasta que uno de estos de presione. Sin embargo, bien es sabido que antes de permitir el paso de peatones el paso vehicular debe haber estado activo cierta cantidad de tiempo. Por esta razón, se programa primero una función de retraso que utiliza los temporizadores internos del microcontrolador, de tal forma que se puede esperar una cantidad de segundos indicada, o bien esperar medio segundo. De esta forma, se llama esta función para que el estado PV esté activo con normalidad por 10 segundos, para luego habilitar una variable que indica que ya es permitido el cambio del estado. Cuando pasan los 10 segundos, si algún botón se presionó antes, se cambiará inmediatamente al estado T. Caso contrario será necesario esperar a que algún botón se presiones para cambiar de estado.

En el estado de transición T, primero se llama una función propio que se encarga de alterar entre dos salidas por medio segundo por un total de 3 veces (el proceso dura 3 segundos) para de esa forma lograr que las luces parpadeen. De esta forma, primero se hace que la luz verde del semáforo vehicular parpadee por 3 segundos, para luego apagarse y encender la luz amarilla del paso vehicular. Esta se mantiene encendida por 2 segundos, para luego proceder a apagarse y encender la luz roja del paso vehicular. Durante todo este proceso las luces rojas de los pasos peatonales se encuentran encendidas, y las luces verdes están apagadas. Antes de cambiar al siguiente estado, el circuito mantiene este estado durante 1 segundo más.

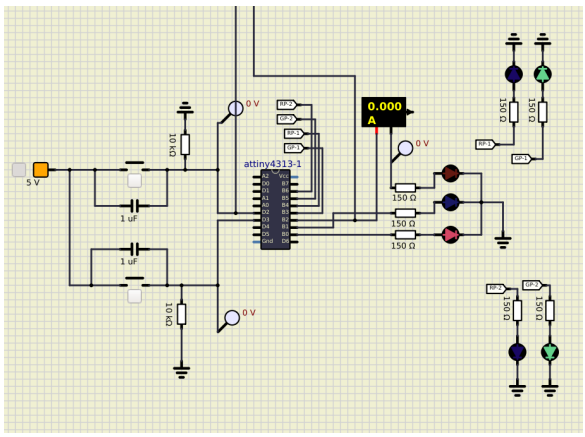
Finalmente, en el estado de paso peatonal PV se encienden las luces verdes de los pasos peatonales y se apagan sus respectivas luces rojas. Luego, el programa espera durante 10 segundos, para luego iniciar la secuencia de parpadeo con las luces verdes de los semáforos peatonales por 3 segundos y proceder a apagarse, para luego encender las luces rojas de los semáforos peatonales. Se espera durante un segundo, y luego se pasa incondicionalmente al estado PV, para repetir todo el comportamiento mencionado. Todo esto se desarrollo bajo el lenguaje de programación C y utilizando los registros del microcontrolador ATtiny4313.

3.3. Repositorio de Git

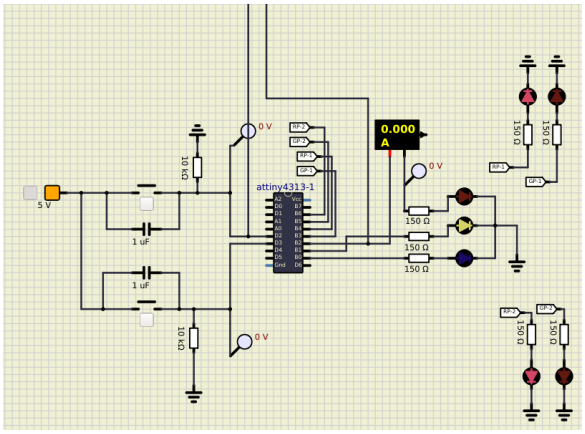
Este proyecto se trabajó en un repositorio de Git para poder tener un control de las diversas versiones de todos los archivos que componen el proyecto. Se utilizó la plataforma de GitHub y el repositorio se encuentran en este link. Este laboratorio se encuentra en el directorio "Laboratorio 2".

3.4. Funcionalidad del circuito y del programa

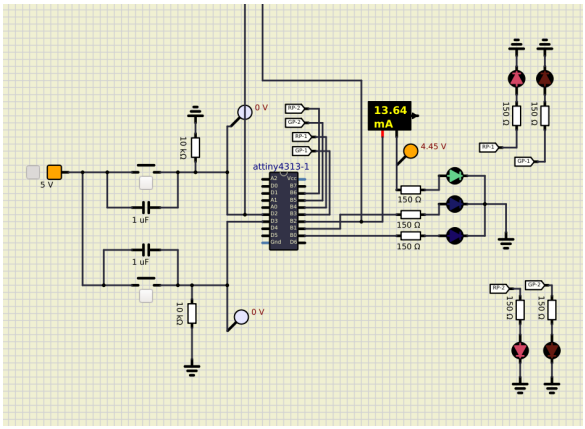
A manera de comprobación del funcionamiento del diseño creado se adjuntaron las capturas que se muestran en la Figura 11. Primeramente, se tiene el estado de paso vehicular que se muestra en la Figura 11c donde se destaca que ambos LEDs de paso peatonal se encuentran en rojo indicando que los peatones aún no pueden pasar y el LED encendido en el semáforo vehicular es el de color verde. Seguidamente, se presiona el botón y se inicia la secuencia de tiempos descrita en el enunciado del laboratorio. Para este momento, el software en el microcontrolador se va a encontrar en un estado de transición donde los peatones aun no van poder pasar y en el semáforo vehicular se enciende el LED de color amarillo. Esto se ilustra en la Figura 11b. Por último, después de toda la secuencia descrita anteriormente, se pasa al estado de paso peatonal mostrado en la Figura 11a. Para este instante los LEDs verdes que indican el paso peatonal se van a encender y el LED de color rojo del semáforo vehicular también se va a encontrar activo. Cabe resaltar que, después de terminar esta secuencia, se va a volver al estado 'default' de paso vehicular hasta que se vuelva a presionar el botón de paso peatonal.



(a) Estado de paso peatonal.



(b) Estado de transición.



(c) Estado de paso vehicular.

Figura 11: Descripción de funcionamiento del circuito

4. Conclusiones y Recomendaciones

Durante esta práctica de laboratorio fue posible explorar el funcionamiento de la lectura de pines utilizando interrupciones, lo cual puede ser de gran utilidad para no estar leyendo constantemente el estado de un pin. Además, se observó que diversos microcontroladores poseen temporizadores internos para poderlos utilizar como contadores, y de esa forma poder contabilizar el tiempo de los procesos sin requerir necesariamente de recursos externos. Fue posible implementar estos dos conceptos para implementar una máquina de estados que representa el funcionamiento de un semáforo vehicular de una vía y dos semáforos peatonales. Nuevamente se ponen en práctica las habilidades de diseño para proponer un circuito robusto que funcione de forma estable, así como la capacidad de poder implementar un microcontrolador no conocido previamente, donde en este caso correspondió al ATtiny4313.

Como recomendación nuevamente se tiene revisar de forma detallada la documentación relacionada al microcontrolador, así como buscar si es posible ejemplos de las modificaciones de sus registros, ya que estos datos no se mencionan en la hoja del fabricante y no siempre es sencillo encontrar la dirección de la librería asociada al microcontrolador. Además, a pesar de que el resultado obtenido es muy satisfactorio, se presenta en algunos casos una imprecisión en el tiempo de décimas de segundo, por lo que sería bueno intentar, siempre utilizando los temporizadores del microcontrolador, tiempos más precisos. También es importante siempre poder identificar los estados de cada proceso, ya que esto facilitará su representación y, de esa forma, su simulación. En conclusión, fue posible aplicar los conceptos de interrupciones y temporizadores para poder desarrollar un semáforo que se representa por medio de una máquina de estados.

5. Referencias

[1] Atmel. 8-bit microcontroller with 2/4k bytes in-system programmable flash. Disponible en: https://pdf1.alldatasheet.com/datasheet-pdf/view/392236/ATMEL/ATTINY4313.html, 2011.

[2] Kingbright. T-1 3/4 (5mm) full color led lamp. Disponible en: https://www.arduino.cc/documents/datasheets/LEDRGB-L-154A4SURK.pdf.

6. Anexos

Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 120 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
- Data and Non-volatile Program and Data Memories
  - 2/4K Bytes of In-System Self Programmable Flash
    - Endurance 10,000 Write/Erase Cycles
  - 128/256 Bytes In-System Programmable EEPROM
    - Endurance: 100,000 Write/Erase Cycles
  - 128/256 Bytes Internal SRAM
  - Programming Lock for Flash Program and EEPROM Data Security
- Peripheral Features
  - One 8-bit Timer/Counter with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare and Capture Modes
  - Four PWM Channels
  - On-chip Analog Comparator
  - Programmable Watchdog Timer with On-chip Oscillator
  - USI – Universal Serial Interface
  - Full Duplex USART
- Special Microcontroller Features
  - debugWIRE On-chip Debugging
  - In-System Programmable via SPI Port
  - External and Internal Interrupt Sources
  - Low-power Idle, Power-down, and Standby Modes
  - Enhanced Power-on Reset Circuit
  - Programmable Brown-out Detection Circuit
  - Internal Calibrated Oscillator
- I/O and Packages
  - 18 Programmable I/O Lines
  - 20-pin PDIP, 20-pin SOIC, 20-pad MLF/VQFN
- Operating Voltage
  - 1.8 – 5.5V
- Speed Grades
  - 0 – 4 MHz @ 1.8 – 5.5V
  - 0 – 10 MHz @ 2.7 – 5.5V
  - 0 – 20 MHz @ 4.5 – 5.5V
- Industrial Temperature Range: -40°C to +85°C
- Low Power Consumption
  - Active Mode
    - 190 µA at 1.8V and 1MHz
  - Idle Mode
    - 24 µA at 1.8V and 1MHz
  - Power-down Mode
    - 0.1 µA at 1.8V and +25°C

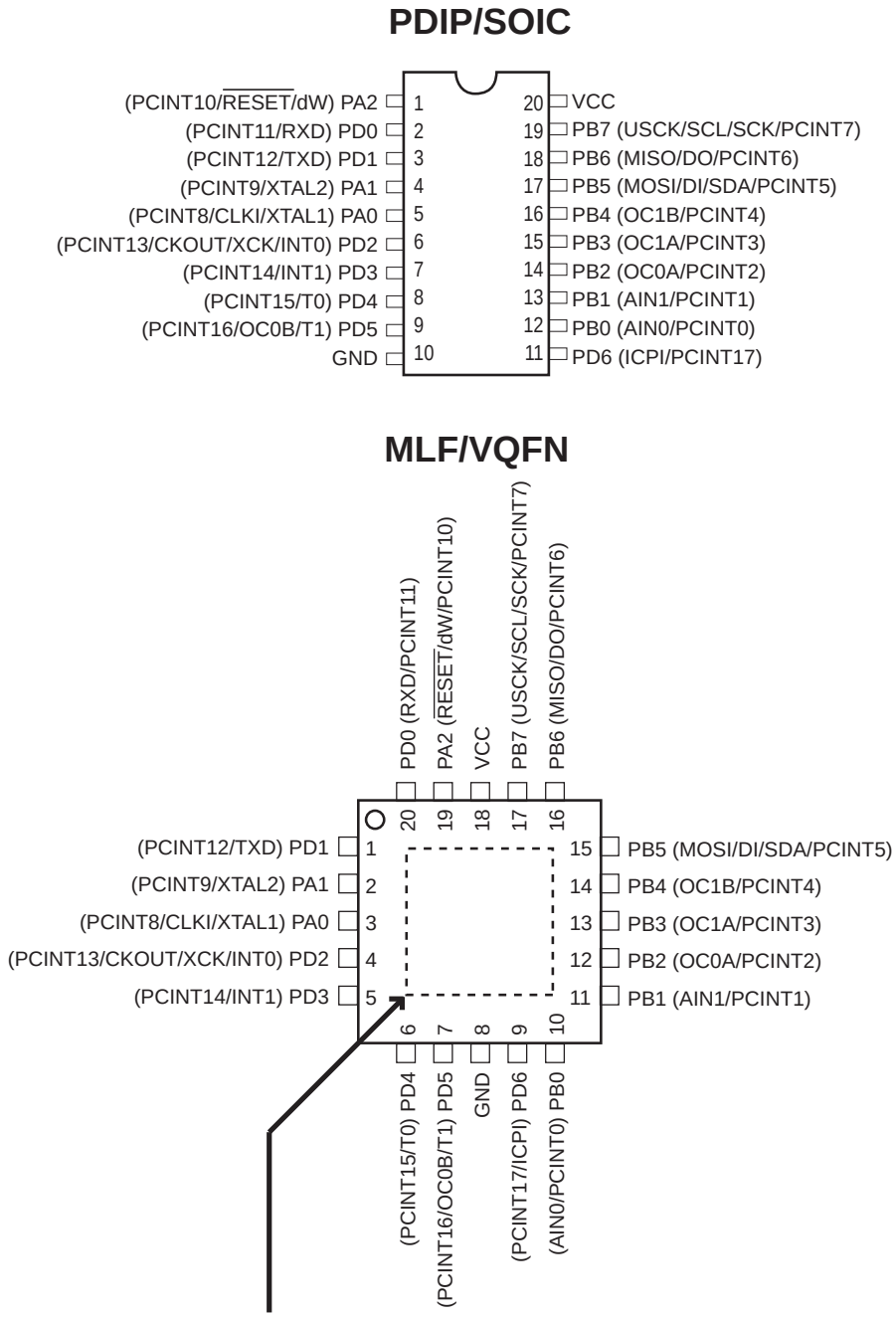
8-bit AVR® Microcontroller with 2/4K Bytes In-System Programmable Flash

ATtiny2313A
ATtiny4313

Rev. 8246B–AVR–09/11

1. Pin Configurations

Figure 1-1. Pinout ATtiny2313A/4313



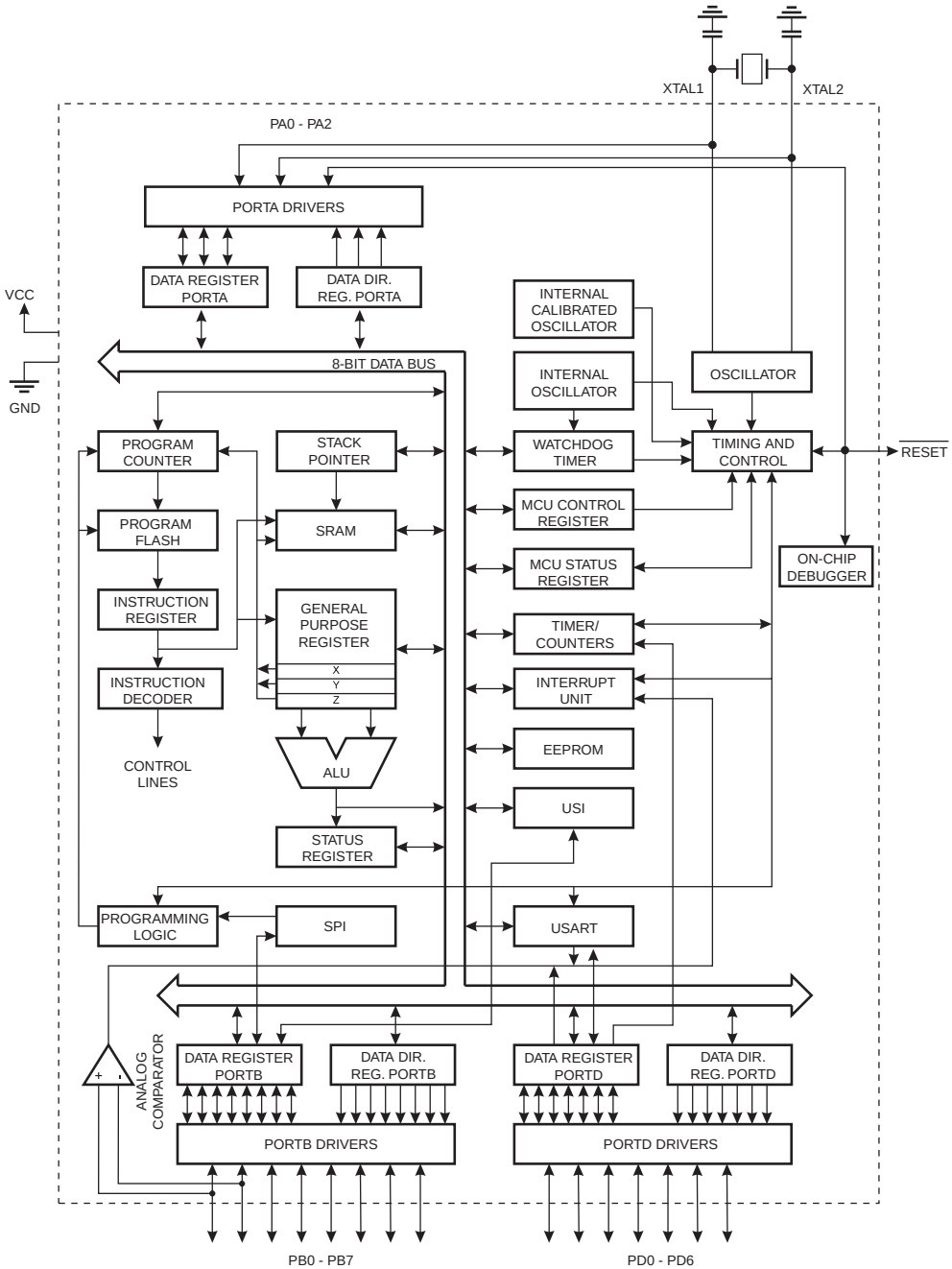
NOTE: Bottom pad should be soldered to ground.

2. Overview

The ATtiny2313A/4313 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATtiny2313A/4313 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



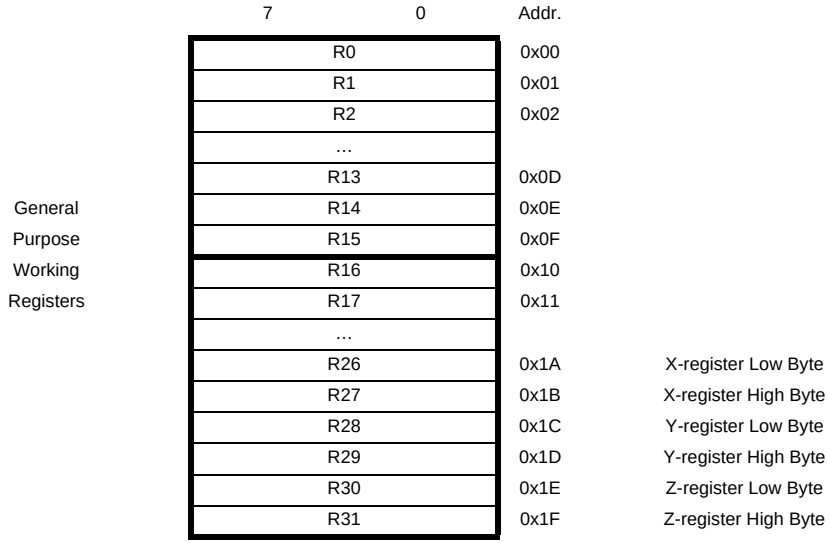
4.4 General Purpose Register File

The Register File is optimized for the AVR Enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the Register File:

- One 8-bit output operand and one 8-bit result input
- Two 8-bit output operands and one 8-bit result input
- Two 8-bit output operands and one 16-bit result input
- One 16-bit output operand and one 16-bit result input

Figure 4-2 shows the structure of the 32 general purpose working registers in the CPU.

Figure 4-2. AVR CPU General Purpose Working Registers



Most of the instructions operating on the Register File have direct access to all registers, and most of them are single cycle instructions.

As shown in Figure 4-2, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y- and Z-pointer registers can be set to index any register in the file.

4.4.1 The X-register, Y-register, and Z-register

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the data space. The three indirect address registers X, Y, and Z are defined as described in Figure 4-3.

Kingbright

T-1 3/4 (5mm) FULL COLOR LED LAMP

Part Number: L-154A4SURKQBDZGC

Hyper Red
Blue
Green

ATTENTION
OBSERVE PRECAUTIONS
FOR HANDLING
ELECTROSTATIC
DISCHARGE
SENSITIVE
DEVICES

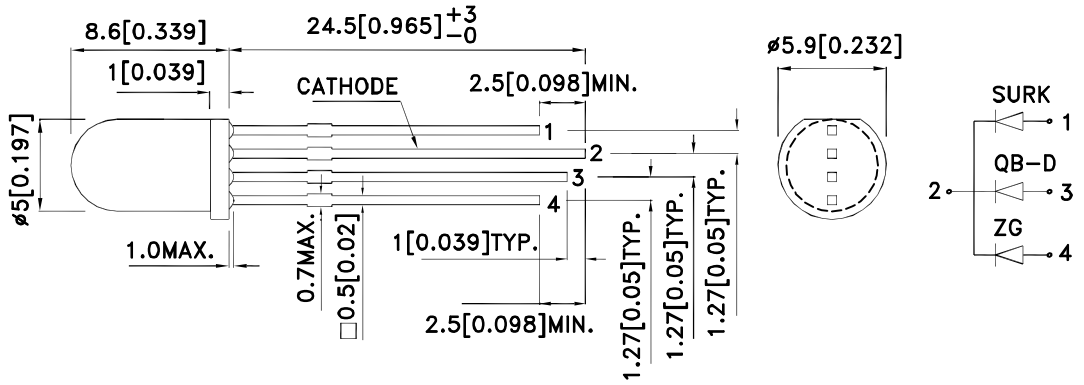
Features

- Uniform light output.
- Low power consumption.
- Long life-solid state reliability.
- RoHS compliant.

Description

The Hyper Red source color devices are made with Al-GaInP on GaAs substrate Light Emitting Diode.
The Blue source color devices are made with InGaN Light Emitting Diode.
The Green source color devices are made with InGaN on Sapphire Light Emitting Diode.
Static electricity and surge damage the LEDS.
It is recommended to use a wrist band or anti-electrostatic glove when handling the LEDs.
All devices, equipment and machinery must be electrically grounded.

Package Dimensions



Notes:
1. All dimensions are in millimeters (inches).
2. Tolerance is ±0.25(0.01") unless otherwise noted.
3. Lead spacing is measured where the leads emerge from the package.
4. The specifications, characteristics and technical data described in the datasheet are subject to change without prior notice.