UNIVERSIDAD DE COSTA RICA

Escuela de Ingeniería Eléctrica

IE-0624: Laboratorio de Microcontroladores

Laboratorio 2
GPIOs, Timers y FSM

Prof. Marco Villalta

Estudiantes:
Sofía Fonseca Muñoz, B42634

25 de enero de 2023

Índice

1. Introducción/Resumen

Este laboratorio consiste en el uso del microcontrolador ATtiny4313 y componentes que complementen su uso para la creación del juego Simón dice. En el juego hay 4 leds y cuatro botones de colores correspondientes entre rojo, azul, verde y amarillo. Para ganar el juego, se debe adivinar la secuencia aleatoria correctamente. Existen 14 secuencias y el tiempo en que se muestra cada una de ellas disminuye en cada nivel.

El juego se programó utilizando interrupciones tanto para el tiempo como para la activación de los botones.

Como parte del algoritmo del juego se hizo una máquina de estados de 6 estados que consisten en Tiempo de Espera, Empieza el juego, se muestra secuencia, se introduce secuencia, se pasa de nivel y se pierde el juego. Cada uno de ellos está explicado a detalle en este informe.

El repositorio correspondiente a este laboratorio se puede encontrar en el enlace https://github.com/sofifon/IE0624-2023/tree/main/Lab2_Simon.

2. Nota teórica

2.1. Microcontrolador

El microcontrolador asignado a este proyecto corresponde al ATtiny4313. Este es un microcontrolador de alto rendimiento basado en la arquitectura RISC. Cuenta con 4 KB de memoria Flash, 128B de EEPROM y 128B de SRAM. Además cuenta con 18 pines de propósito general de entrada y salida, 32 registros de propósito general y periféricos como 2 contadores internos, interrupciones internas y externas, el protocolo USART, un Watchdog timer con oscilador interno y modos de ahorro de energía que se seleccionan por medio de software.[1]

Una de las principales características de este microcontrolador, es su capacidad para funcionar a bajo poder, ya que puede operar entre 1.8 y 5.5 V.

A continuación se muestra un diagrama de pines del microprocesador donde se muestran el nombre que identifica a cada pin y entre paréntesis los registros asociados a cada pin. Hay 3 registros relacionados a cada grupo de pines, uno de ellos es el DDRx, otro el PORTx y el PINx, donde x puede ser A, B o D. Con el registro DDR se inicializan los pines y se establecen como entradas o salidas, con el registro PORT se lee el valor del pin y con el registro PIN se escribe en el pin, independientemente si se ha programado como entrada o salida. Además si se quieren utilizar por interrupciones se debe configurar el pin PCMSKx, junto con otros registros que se explicarán más adelante.

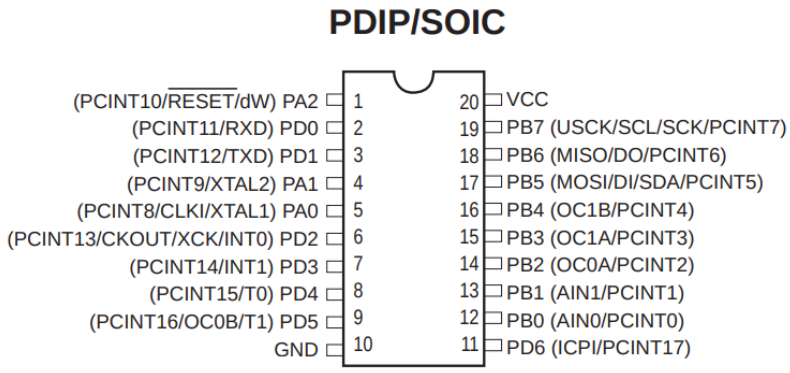


Figura 1: Diagrama de pines del procesador ATtiny4313. Tomada de https://ww1.microchip.com/downloads/en/DeviceDoc/doc8246.pdf

En el diagrama de bloques de este microprocesador, se puede ver como los 32 registros de propósito general se conectan a la ALU, permitiendo que dos registros pasen información a la vez y lo hacen muy rápido. Además, se pueden observar otros bloques que se observaron anteriormente como es el Watchdog timer, las comunicaciones seriales, la RAM, el EEPROM y los puertos IO.

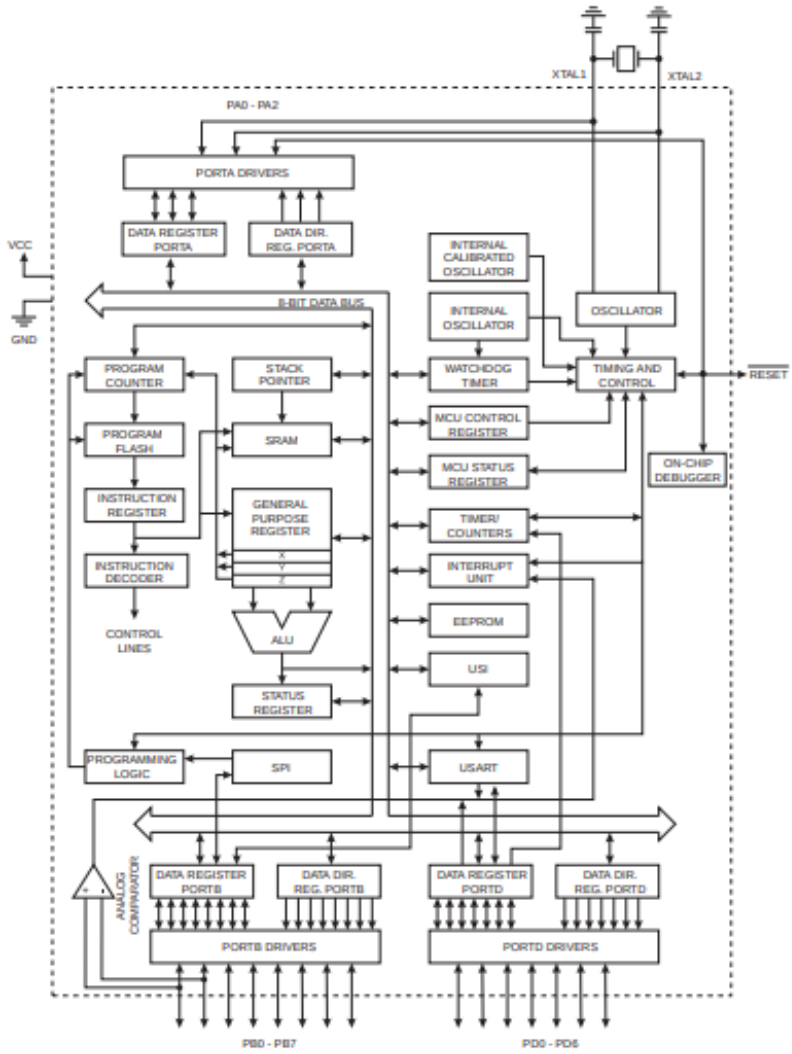
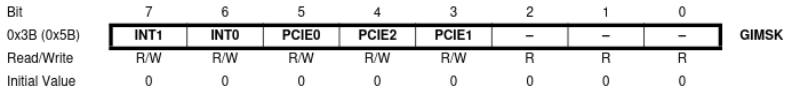


Figura 2: Diagrama de bloques del procesador ATtiny4313. Tomada de https://ww1.microchip.com/downloads/en/DeviceDoc/doc8246.pdf

2.1.1. Registros utilizados

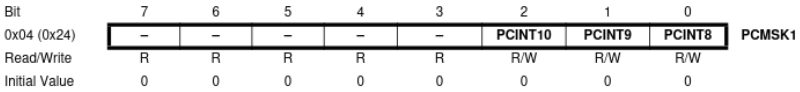
Existen diferentes tipos de rutinas que se pueden ejecutar por interrupción que tienen niveles de prioridad, en el nivel más alto de prioridad se encuentra el reset, luego siguen otras como por temporizador y por último las interrupciones por pines. GIMSK, PCMSK y MCUCR son los registros que se deben modificar para activar las interrupciones. El registro GIMSK habilita las interrupciones por pines de entrada. El registro MCUCR establece en qué flanco la interrupción debe de ser habilitada. Los registros PCMSK1 y PCMSK2, adem´as de los registros INT0 y INT1 establece cuáles registros específicos tienen permisos para activar las interrupciones.[2]

Como se necesitaban 4 botones en este proyecto que se activaran por interrupciones diferentes, se utilizaron INT0, INT1, PCMSK1 y PCMSK2 para cada uno de los botones.



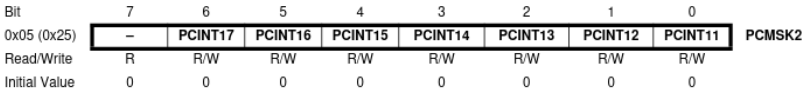
Bit	7	6	5	4	3	2	1	0	
0x3B (0x5B)	INT1	INT0	PCIE0	PCIE2	PCIE1	–	–	–	GIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

Figura 3: Configuración general del registro GIMSK. [2]



Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	–	–	–	–	–	PCINT10	PCINT9	PCINT8	PCMSK1
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Figura 4: Configuración general del registro PCMSK1. [2]



Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	–	PCINT17	PCINT16	PCINT15	PCINT14	PCINT13	PCINT12	PCINT11	PCMSK2
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Figura 5: Configuración general del registro PCMSK2. [2]

Luego existen otros registros como el TIMSK que habilita las interrupciones por el temporizador interno. Junto con el prescaler, se puede generar un contador por interrupciones que cuente segundos. Para cambiar el prescaler se deben modificar otros registros. Con el registro TCCR0B dividir el temporizador hasta 1024 bits o escoger un temporizador externo. El registro TCNT0 inicializa el temporizador. [2]

2.1.2. Características eléctricas

En la siguiente tabla se pueden encontrar las características físicas de este microcontrolador, las cuales se deben tomar en cuenta para la implemetación en la que se desea utilizar el microcontrolador.[2]

Especificaciones eléctricas	
Temperatura de operación	-55°C to +125°C
Temperatura de almacenamiento	-65°C to +150°C
Voltaje entre RESET y Vss	-0.5V to +13.0V
Voltaje entre los otros pines y Vss	-0.5V to Vcc +0.5V
Máximo voltaje de operación	6.0V
Máxima corriente por cada pin	40.0 mA
Máxima corriente total	200.0 mA

2.2. Bibliotecas

A nivel de software, se utilizó programaci on en C, donde se incluyeron las siguientes bibliotecas:

- **avr/interrupt.h**: Activa las interupciones en el microcontrolador.
- **avr/io.h**: Mapea adecuadamente cada uno de los puertos IO del microcontrolador, y define sus nombres adecuadamente a la documentación.

2.3. Diseño del circuito

El diseño completo del circuito eléctrico que se utilizó se puede observar en la siguiente imagen:

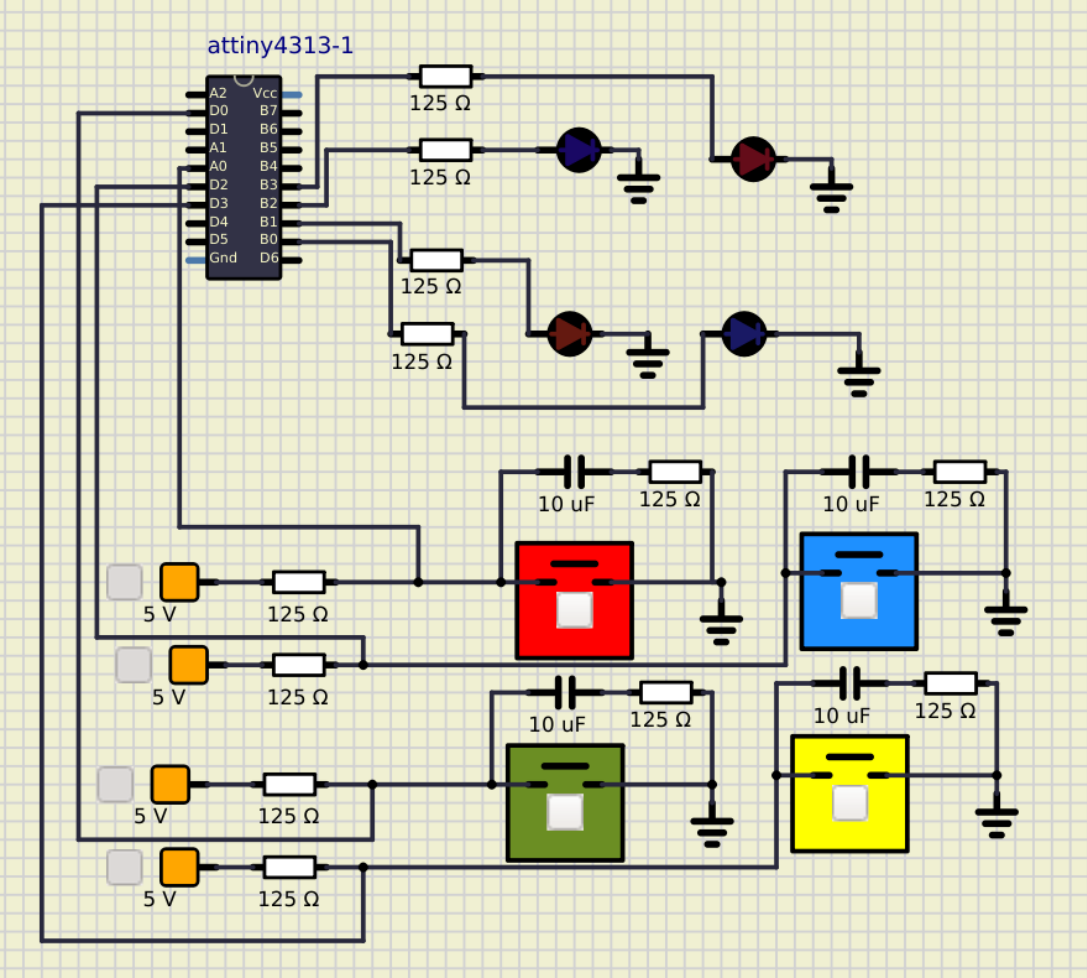


Figura 6: Diseño completo del circuito del semáforo. Imagen propia.

En el diseño se escogió 4 pines de entrada correspondiente a los botones y 4 pines de salida, ycorrespondientes a los 4 leds, ya que cada uno cuenta con funcionamientos distintos. Cada uno de las conexiones fueron realizadas como se muestra en la siguiente tabla:

Pin	**Configuracion**	**Elemento**	**Funcion**
PD3	Entrada	Bot\'on amarillo	Activar el juego o activar luz amarilla.
PA0	Entrada	Bot\'on rojo	Activar el juego o activar luz roja.
PD2	Entrada	Bot\'on azul	Activar el juego o activar luz azul.
PD0	Entrada	Bot\'on verde	Activar el juego o activar luz verde.
PB3	Salida	LED azul	Mostrar el color azul de la secuencia.
PB2	Salida	LED rojo	Mostrar el color rojo de la secuencia.
PB1	Salida	LED verde	Mostrar el color verde de la secuencia.
PB0	Salida	LED amarillo	Mostrar el color amarillo de la secuencia.

Se escogieron los pines B0 a B3 para los LEDs para mantener la consistencia en la programación.

En el caso de los botones, se escogieron pines que estratégicamente fueran a funcionar para tener 4 tipos de interrupciones diferentes que correspondieran a los registrozs y que dos botones no activen la misma interrupción.

Por el lado eléctrico, en el caso del botón se utiliza una fuente de 5V, por lo que se debe encontrar un valor de resistencia para que la corriente no sobrepase los 40mA que permiten los pines individualmente. Utilizando la ecuación acontinuación se puede encontrar que se necesita una resistencia de 125 Ω. El capacitor que se agrega para que la corriente inicial no sobrepase los límites y pueda causar problemas con el microprocesador.

$$V = IR \Rightarrow R = \frac{V}{I} = \frac{5V}{40mA} = 125\ \Omega \tag{1}$$

Por lo que la conexión del botón queda así:

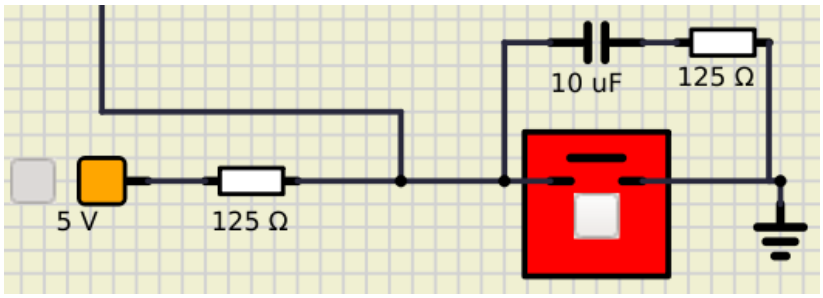


Figura 7: Conexión del botón. Imagen propia.

En el caso de los LEDs, estas también no soportan una corriente mayor a los 40 mA, por lo que utilizando de nuevo la ecuación 1, se puede determinar que se necesitan resistencias de 125 Ω. Se puede ver en la siguiente imagen:

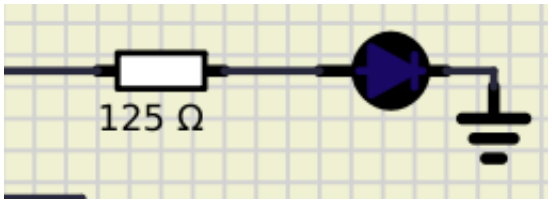


Figura 8: Conexión de los LEDs del juego. Imagen propia.

2.4. Componentes electrónicos

Para la creación del juego Simón dice, se utilizó la lista de componentes que se encuentra en la siguiente tabla. Los precios de cada componente fueron tomados de https://www.crcibernetica.com/ tomando en cuenta un tipo de cambio de 595 colones equivalente a 1 USD. Además los precios de las resistencias y los LEDs se calcularon a partir de paquetes grandes dividiendo su valor entre la cantidad, ya que por su valor no se consiguen individualmente.

Componente	**Precio (colones)**
ATtiny4313	1255
4 x Mini Push Button Switch	845
12 x Resistor 125 ohm	360
4 x LEDs	75
10 x Capacitores 10uF	595
Fuente 5 V	4518
Precio total	**7650**

El precio total de todos los componentes necesarios para construir el juego sería 7650 colones.

3. Desarrollo/Análisis de resultados

El desarrollo del software para este circuito consistió de dos partes elementales: el manejo de las interrupciones y la creación de la máquina de estados. Para la máquina de estados se escogieron 6 estados, correspondientes a las seis tipos de actividad distinta que existen en el juego. En la siguiente imagen se encuentra una representación:

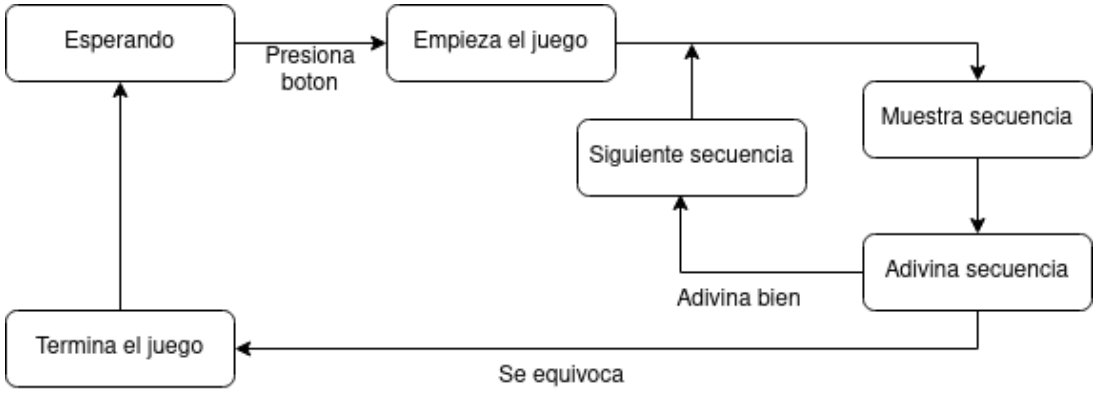


Figura 9: Estados escogidos según las instrucciones dadas.

Cada estado se definió como se muestra a continuación:

- **Esperando:** Es el estado base, es decir, en el que empieza el juego. Espera a que el jugador esté listo para empezar. Todos las funcionalidades se encuentran pausadas esperando que alguno de los botones sea presionado por el . Cuando esto sucede, se pasa al siguiente estado.
- **Empieza el juego:** En este estado se inicializan las variables del juego, incluyendo escoger una secuencia, reiniciar los contadores del nivel de secuencia, entre otros. Además, se encienden todos los LEDs dos veces indicando al jugador que el juego va a iniciar.
- **Muestra secuencia:** En este estado se lee el arreglo que contiene la secuencia de luces que se quiere que el jugador adivine. Se toma en cuenta el largo de la secuencia basado en el nivel en el que el jugador se encuentra. Cuando termina, se apagan todas las luces y se pasa al siguiente estado.
- **Adivina secuencia:** En este estado se espera hasta que el jugador presione alguno de los botones para compararlo con la secuencia a adivinar. Cuando se presiona el botón, se enciende la luz correspondiente por una fracción de segundo. Si el botón presionado es correcto y la secuencia no ha terminado, espera por la siguiente interacción. Si la sencuencia termina, eso significa que la persona adevinó y se pasa al estado para establecer la siguiente secuencia. Si el botón presionado es incorrecto, pasa al estado de terminar el juego.
- **Siguiente secuencia:** En este estado se establece las variables para la siguiente secuencia, incluyendo aumentar el largo de la secuencia y disminuir el tiempo de exposición de cada luz en la secuencia. Cuando termina, vuelve al estado de mostrar la secuencia.
- **Termina el juego:** En este estado se encienden todos los LEDs 3 veces para indicar al jugador que ha perdido. Cuando termina regresa al estado de espera.

Por el lado de las interrupciones, fue necesario crear dos tipos de funciones, una que es activada por los botones y otra por el temporizador. Para el primer tipo de función se utilizó ina variable auxiliar llamada color, la cual contiene un valor entre 0 y 3 o 5. Entre 0 y 3 corresponden a los colores que se presentan en los botones y el 5 es para indicar que ningún color ha sido escogido. Cada una de las funciones asigna el valor entre 0 y 3 según corresponde con su botón.

La interrupción por temporizador posee un contador que aumenta con cada ciclo de reloj. Debido a que se pudo disminuir la frecuencia del reloj interno a 60 Hz, cada vez que el contador llega a 60 se completa un segundo. A parte de ese contador, existen otros dos que se encargan de saber cuando ha pasado 200 ms de segundo para disminuir el tiempo en cada ciclo y un segundo para el resto de las señales. Cuando el contador interno llega a 60, el contador de los segundos aumenta, así es como la máquina de estados sabe cuánto tiempo ha pasado.

A continuación se va a hacer una pequeña demostración del funcionamiento, sin embargo para mejores resultados en el enlace https://youtu.be/RdaxniYiVv4 se puede ver una demostración de la simulación completa en SIMULIDE del juego Simón dice. Todas las imágenes son parte de la misma demostración por lo que en la esquina superior izquierda de cada imagen se puede ver un temporizador de la simulación que demuestra el tiempo que ha pasado.

La primera imagen se toma después de que el botón ha sido presionado:

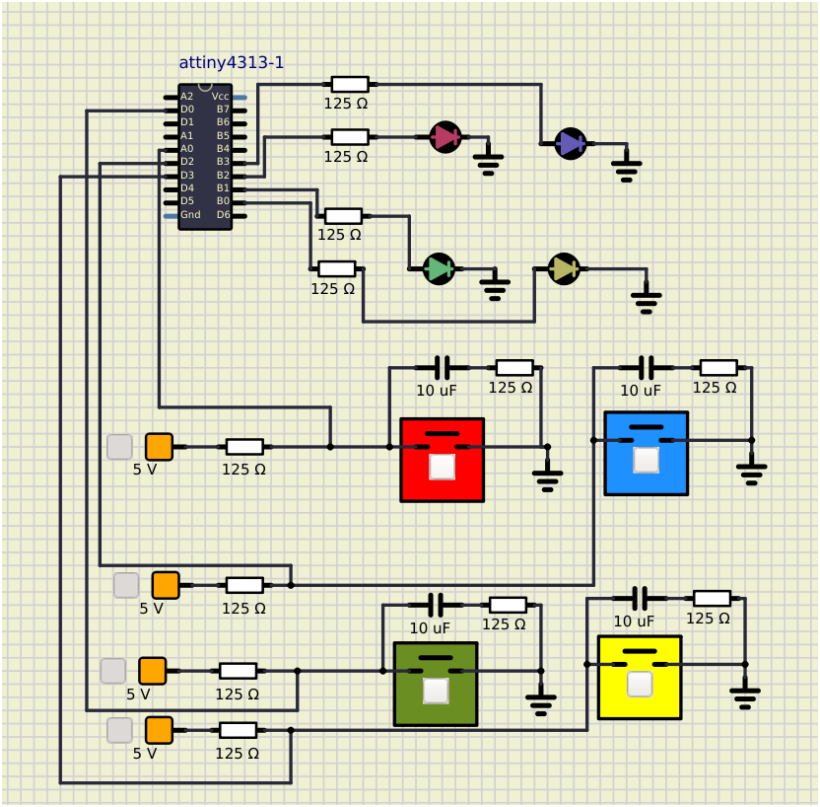


Figura 10: Demo del juego empezando.

Una vez que parpadea, inicia la secuencia:

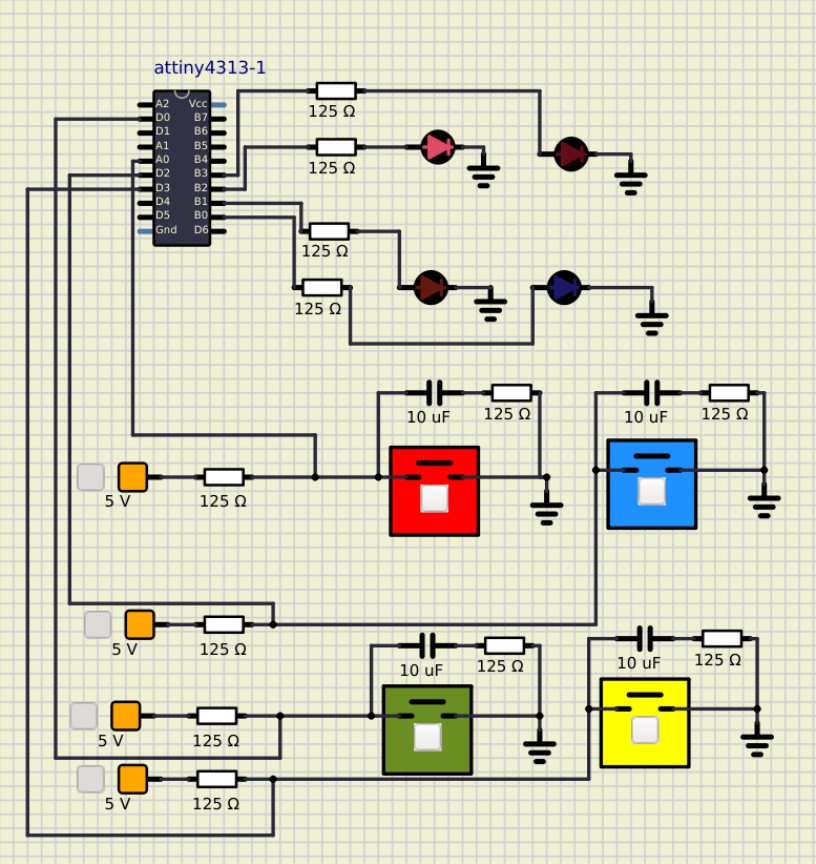


Figura 11: Demo del juego enseñando la secuencia inicial.

Cuando se adivina la secuencia correctamente como se ve en esta imagen:

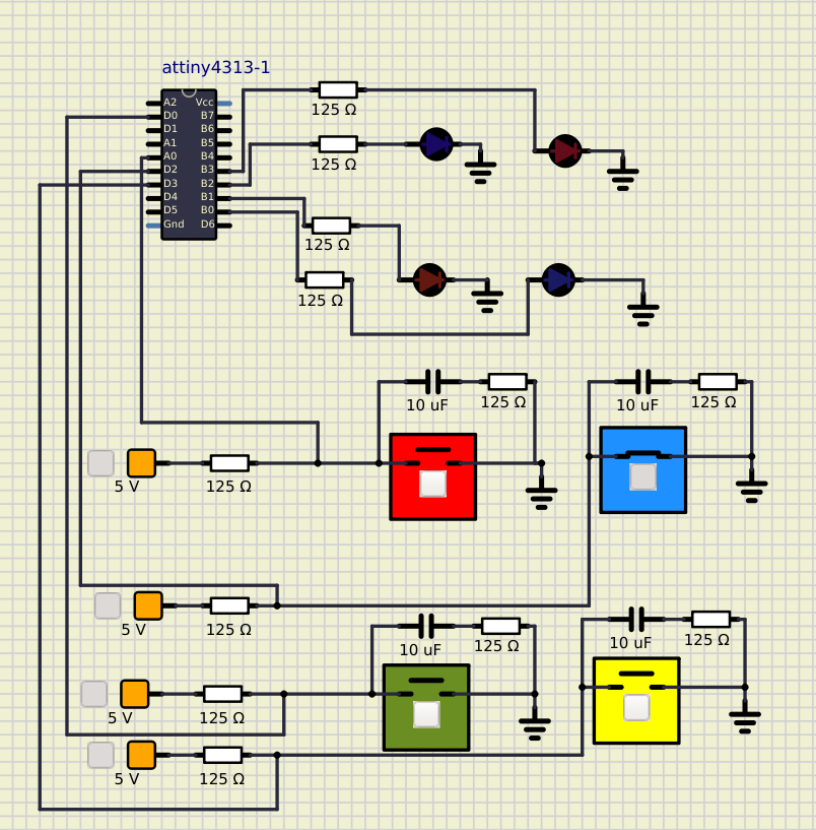


Figura 12: Demo del jugador presionando el botón correcto.

Es cuando inicia la siguiente secuencia:

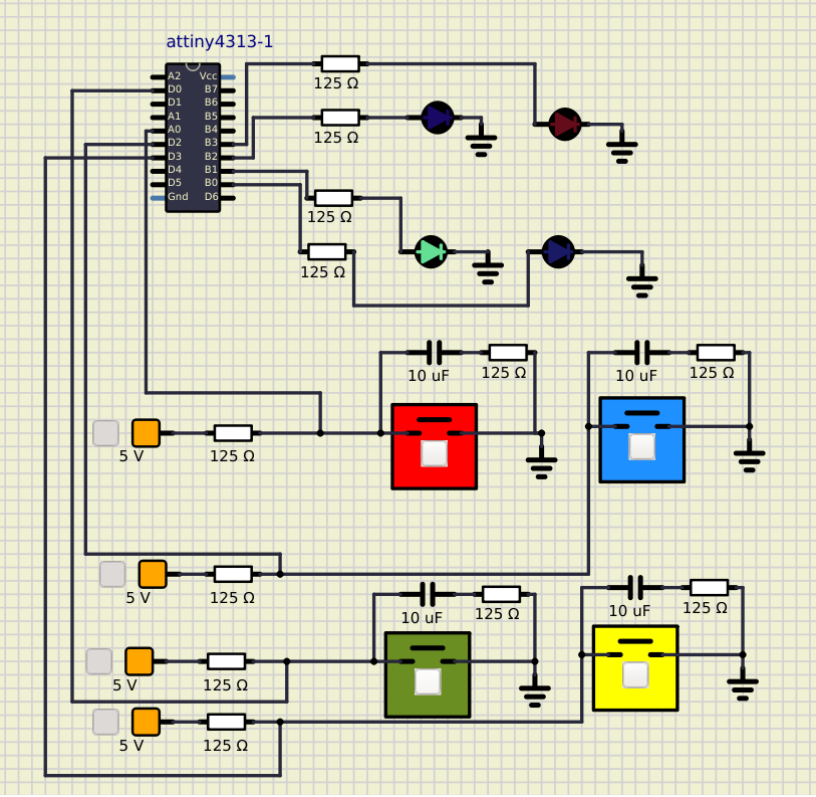


Figura 13: Demo del inicio de la siguiente secuencia.

4. Conclusiones y recomendaciones

- Considerar el sistema como una máquina de estados facilita su programación para un caso como este que se deben programar las salidas en diferentes momentos de tiempo.
- El agregar un prescaler ayuda a disminuir la frecuencia a la que corre el microcontrolador para lograr los tiempos que se desean en el programa.
- Las programaciones por interrupción no deberían tener una duración muy alta porque impiden que el microcontrolador haga sus funciones principales.

Referencias

1. Microchip (2022) ATtiny4313.Recuperado de: https://www.microchip.com/en-us/product/ATtiny4313

2. Atmel. (2011) ATtiny2313A. ATtiny4313. Recuperado de: https://ww1.microchip.com/downloads/en/DeviceDoc/doc8246.pdf

5. Apéndice

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

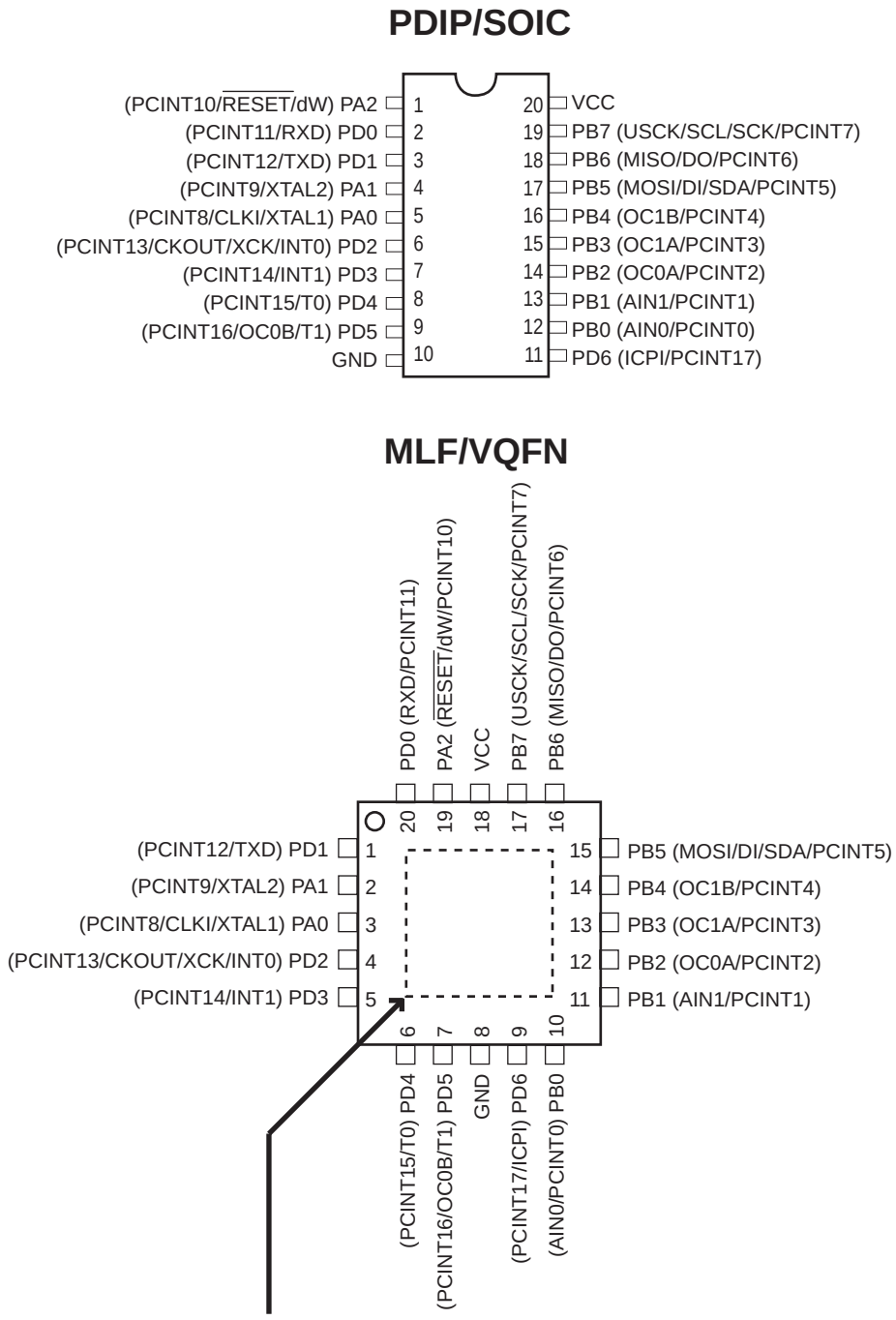
8-bit AVR® Microcontroller with 2/4K Bytes In-System Programmable Flash

ATtiny2313A
ATtiny4313

Rev. 8246B–AVR–09/11

1. Pin Configurations

Figure 1-1. Pinout ATtiny2313A/4313



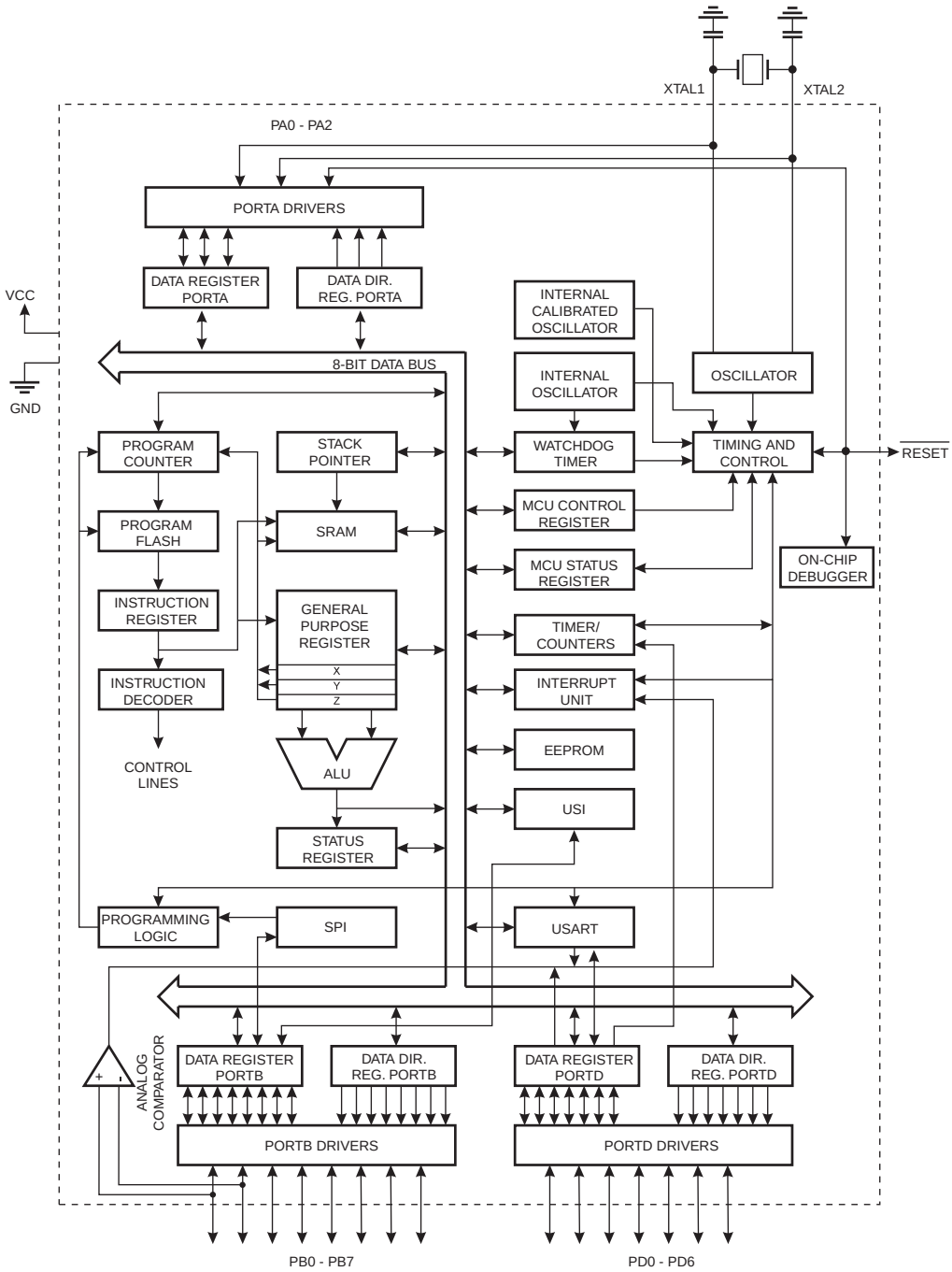
NOTE: Bottom pad should be soldered to ground.

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



C Code Example

```
unsigned char EEPROM_read(unsigned int ucAddress)
{
  /* Wait for completion of previous write */
  while(EECR & (1<<EEPE));
  /* Set up address register */
  EEAR = ucAddress;
  /* Start eeprom read by writing EERE */
  EECR |= (1<<EERE);
  /* Return data from data register */
  return EEDR;
}
```

Note: See “Code Examples” on page 7.

5.4 Register Description

5.4.1 EEAR – EEPROM Address Register

Bit	7	6	5	4	3	2	1	0	
0x1E (0x3E)	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEAR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	X	X	X	X	X	X	X	X	

- **Bit 7 – EEAR7: EEPROM Address**

This is the most significant EEPROM address bit of ATtiny4313. In devices with less EEPROM, i.e. ATtiny2313A, this bit is reserved and will always read zero. The initial value of the EEPROM Address Register (EEAR) is undefined and a proper value must therefore be written before the EEPROM is accessed.

- **Bits 6..0 – EEAR6..0: EEPROM Address**

These are the (low) bits of the EEPROM Address Register. The EEPROM data bytes are addressed linearly in the range 0...(128-1). The initial value of EEAR is undefined and a proper value must be therefore be written before the EEPROM may be accessed.

5.4.2 EEDR – EEPROM Data Register

Bit	7	6	5	4	3	2	1	0	
0x1D (0x3D)	EEDR7	EEDR6	EEDR5	EEDR4	EEDR3	EEDR2	EEDR1	EEDR0	EEDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..0 – EEDR7..0: EEPROM Data**

For the EEPROM write operation the EEDR Register contains the data to be written to the EEPROM in the address given by the EEAR Register. For the EEPROM read operation, the EEDR contains the data read out from the EEPROM at the address given by EEAR.

9.3 Register Description

9.3.1 MCUCR – MCU Control Register

The External Interrupt Control Register contains control bits for interrupt sense control.

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	PUD	SM1	SE	SM0	ISC11	ISC10	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 3, 2 – ISC11, ISC10: Interrupt Sense Control 1 Bit 1 and Bit 0**

The External Interrupt 1 is activated by the external pin INT1 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT1 pin that activate the interrupt are defined in Table 9-2. The value on the INT1 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt

Table 9-2. Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request.
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

- **Bits 1, 0 – ISC01, ISC00: Interrupt Sense Control 0 Bit 1 and Bit 0**

The External Interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT0 pin that activate the interrupt are defined in Table 9-3. The value on the INT0 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 9-3. Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

9.3.2 GIMSK – General Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
0x3B (0x5B)	INT1	INT0	PCIE0	PCIE2	PCIE1	–	–	–	GIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 2..0 – Res: Reserved Bits**

These bits are reserved and will always read as zero.

- **Bit 7 – INT1: External Interrupt Request 1 Enable**

When the INT1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control bits (ISC11 and ISC10) in the External Interrupt Control Register A (EICRA) define whether the external interrupt is activated on rising and/or falling edge of the INT1 pin or level sensed. Activity on the pin will cause an interrupt request even if INT1 is configured as an output. The corresponding interrupt of External Interrupt Request 1 is executed from the INT1 Interrupt Vector.

- **Bit 6 – INT0: External Interrupt Request 0 Enable**

When the INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control bits (ISC01 and ISC00) in the External Interrupt Control Register A (EICRA) define whether the external interrupt is activated on rising and/or falling edge of the INT0 pin or level sensed. Activity on the pin will cause an interrupt request even if INT0 is configured as an output. The corresponding interrupt of External Interrupt Request 0 is executed from the INT0 Interrupt Vector.

- **Bit 5 – PCIE0: Pin Change Interrupt Enable 0**

When the PCIE0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), pin change interrupt 0 is enabled. Any change on any enabled PCINT7..0 pin will cause an interrupt. The corresponding interrupt of Pin Change Interrupt Request is executed from the PCI0 Interrupt Vector. PCINT7..0 pins are enabled individually by the PCMSK0 Register.

- **Bit 4 – PCIE2: Pin Change Interrupt Enable 2**

When the PCIE2 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), pin change interrupt 2 is enabled. Any change on any enabled PCINT17..11 pin will cause an interrupt. The corresponding interrupt of Pin Change Interrupt Request is executed from the PCI2 Interrupt Vector. PCINT17..11 pins are enabled individually by the PCMSK2 Register.

- **Bit 3 – PCIE1: Pin Change Interrupt Enable 1**

When the PCIE1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), pin change interrupt 1 is enabled. Any change on any enabled PCINT10..8 pin will cause an interrupt. The corresponding interrupt of Pin Change Interrupt Request is executed from the PCI1 Interrupt Vector. PCINT10..8 pins are enabled individually by the PCMSK1 Register.

Table 10-4 relates the alternate functions of Port A to the overriding signals shown in Figure 10-5 on page 60.

Table 10-4. Overriding Signals for Alternate Functions in PA2..PA0

Signal Name	PA2/RESET/dW/PCINT10	PA1/XTAL2/PCINT9	PA0/XTAL1/PCINT8
PUOE	$\overline{\text{RSTDISBL}}$[1] + DEBUGWIRE_ENABLE[2]	EXT_OSC[3]	EXT_CLOCK[4] + EXT_OSC[3]
PUOV	1	0	0
DDOE	$\overline{\text{RSTDISBL}}$[1] + DEBUGWIRE_ENABLE[2]	EXT_OSC[3]	EXT_CLOCK[4] + EXT_OSC[3]
DDOV	DEBUGWIRE_ENABLE[2] • $\overline{\text{debugWire Transmit}}$	0	0
PVOE	$\overline{\text{RSTDISBL}}$[1] + DEBUGWIRE_ENABLE[2]	EXT_OSC[3]	EXT_CLOCK[4] + EXT_OSC[3]
PVOV	0	0	0
PTOE	0	0	0
DIEOE	$\overline{\text{RSTDISBL}}$[1] + DEBUGWIRE_ENABLE[2] + PCINT10 • PCIE1	EXT_OSC[3] + PCINT9 • PCIE1	EXT_CLOCK[4] + EXT_OSC[3] + (PCINT8 • PCIE1)
DIEOV	DEBUGWIRE_ENABLE[2] + ($\overline{\text{RSTDISBL}}$[1] • PCINT10 • PCIE1)	$\overline{\text{EXT\_OSC}}$[3] + PCINT9 • PCIE1	(EXT_CLOCK[4] • $\overline{\text{PWR\_DOWN}}$) + ($\overline{\text{EXT\_CLOCK}}$[4] • $\overline{\text{EXT\_OSC}}$[3] • PCINT8 • PCIE1)
DI	dW/PCINT10 Input	PCINT9 Input	CLKI/PCINT8 Input
AIO		XTAL2	XTAL1

Notes:
1. RSTDISBL is 1 when the fuse is “0” (Programmed).
2. DebugWIRE is enabled when DWEN Fuse is programmed and Lock bits are unprogrammed.
3. EXT_OSC = crystal oscillator or low frequency crystal oscillator is selected as system clock.
4. EXT_CLOCK = external closk is selected as system clock.

10.2.2 Alternate Functions of Port B

The Port B pins with alternate function are shown in Table 10-5.

Table 10-5. Port B Pins Alternate Functions

Port Pin	Alternate Function
PB0	AIN0: Analog Comparator, Positive Input PCINT0:Pin Change Interrupt 0, Source 0
PB1	AIN1: Analog Comparator, Negative Input PCINT1: Pin Change Interrupt 0, Source 1
PB2	OC0A:: Timer/Counter0 Compare Match AOutput PCINT2: Pin Change Interrupt 0, Source 2
PB3	OC1A: Timer/Counter1 Compare Match A Output PCINT3: Pin Change Interrupt 0, Source 3

Table 10-5. Port B Pins Alternate Functions

Port Pin	Alternate Function
PB4	OC1B: Timer/Counter1 Compare Match B Output PCINT4: Pin Change Interrupt 0, Source 4
PB5	DI: USI Data Input (Three Wire Mode) SDA: USI Data Input (Two Wire Mode) PCINT5: Pin Change Interrupt 0, Source 5
PB6	DO: USI Data Output (Three Wire Mode) PCINT6: Pin Change Interrupt 0, Source 6
PB7	USCK: USI Clock (Three Wire Mode) SCL : USI Clock (Two Wire Mode) PCINT7: Pin Change Interrupt 0, Source 7

- **Port B, Bit 0 – AIN0/PCINT0**
  - AIN0: Analog Comparator Positive input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the Analog Comparator.
  - PCINT0: Pin Change Interrupt Source 0. The PB0 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port B, Bit 1 – AIN1/PCINT1**
  - AIN1: Analog Comparator Negative input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the analog comparator.
  - PCINT1: Pin Change Interrupt Source 1. The PB1 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port B, Bit 2 – OC0A/PCINT2**
  - OC0A: Output Compare Match A output. The PB2 pin can serve as an external output for the Timer/Counter0 Output Compare A. The pin has to be configured as an output (DDB2 set (one)) to serve this function. The OC0A pin is also the output pin for the PWM mode timer function.
  - PCINT2: Pin Change Interrupt Source 2. The PB2 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port B, Bit 3 – OC1A/PCINT3**
  - OC1A: Output Compare Match A output: The PB3 pin can serve as an external output for the Timer/Counter1 Output Compare A. The pin has to be configured as an output (DDB3 set (one)) to serve this function. The OC1A pin is also the output pin for the PWM mode timer function.
  - PCINT3: Pin Change Interrupt Source 3: The PB3 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port B, Bit 4 – OC1B/PCINT4**
  - OC1B: Output Compare Match B output: The PB4 pin can serve as an external output for the Timer/Counter1 Output Compare B. The pin has to be configured as an output (DDB4 set

(one)) to serve this function. The OC1B pin is also the output pin for the PWM mode timer function.

- PCINT4: Pin Change Interrupt Source 4. The PB4 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port B, Bit 5 – DI/SDA/PCINT5**
  - DI: Three-wire mode Universal Serial Interface Data input. Three-wire mode does not override normal port functions, so pin must be configured as an input. SDA: Two-wire mode Serial Interface Data.
  - PCINT5: Pin Change Interrupt Source 5. The PB5 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port B, Bit 6 – DO/PCINT6**
  - DO: Three-wire mode Universal Serial Interface Data output. Three-wire mode Data output overrides PORTB6 value and it is driven to the port when data direction bit DDB6 is set (one). However the PORTB6 bit still controls the pull-up enabling pull-up, if direction is input and PORTB6 is set (one).
  - PCINT6: Pin Change Interrupt Source 6. The PB6 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port B, Bit 7 – USCK/SCL/PCINT7**
  - USCK: Three-wire mode Universal Serial Interface Clock.
  - SCL: Two-wire mode Serial Clock for USI Two-wire mode.
  - PCINT7: Pin Change Interrupt source 7. The PB7 pin can serve as an external interrupt source for pin change interrupt 0.

Table 10-10. Overriding Signals for Alternate Functions in PD3..PD0

Signal Name	PD3/INT1/ PCINT14	PD2/INT0/XCK/CKOUT/ PCINT13	PD1/TXD/ PCINT12	PD0/RXD/PCINT11
PUOE	0	0	TXD_OE	RXD_OE
PUOV	0	0	0	PORTD0 • $\overline{\text{PUD}}$
DDOE	0	0	TXD_OE	RXD_EN
DDOV	0	0	1	0
PVOE	0	XCKO_PVOE	TXD_OE	0
PVOV	0	XCKO_PVOV	TXD_PVOV	0
PTOE	0	0	0	0
DIEOE	INT1 Enable + PCINT14	INT0 Enable/ XCK Input Enable/PCINT13	PCINT12	PCINT11
DIEOV	PCINT14	PCINT13	PCINT12	PCINT11
DI	INT1 Input/ PCINT14	INT0 Input/XCK Input/ PCINT13	PCINT12	RXD Input/PCINT11
AIO	–	–	–	–

10.3 Register Description

10.3.1 MCUCR – MCU Control Register

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	**PUD**	**SM1**	**SE**	**SM0**	**ISC11**	**ISC10**	**ISC01**	**ISC00**	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – PUD: Pull-up Disable**

When this bit is written to one, the pull-ups in the I/O ports are disabled even if the DDxn and PORTxn Registers are configured to enable the pull-ups ({DDxn, PORTxn} = 0b01). See “Configuring the Pin” on page 56 for more details about this feature.

10.3.2 PORTA – Port A Data Register

Bit	7	6	5	4	3	2	1	0	
0x1B (0x3B)	–	–	–	–	–	**PORTA2**	**PORTA1**	**PORTA0**	**PORTA**
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

10.3.3 DDRA – Port A Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x1A (0x3A)	–	–	–	–	–	**DDA2**	**DDA1**	**DDA0**	**DDRA**
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

10.3.4 PINA – Port A Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x19 (0x39)	–	–	–	–	–	PINA2	PINA1	PINA0	PINA
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

10.3.5 PORTB – Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x18 (0x38)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

10.3.6 DDRB – Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x17 (0x37)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

10.3.7 PINB – Port B Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x16 (0x36)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

10.3.8 PORTD – Port D Data Register

Bit	7	6	5	4	3	2	1	0	
0x12 (0x32)	–	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

10.3.9 DDRD – Port D Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x11 (0x31)	–	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

10.3.10 PIND – Port D Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x10 (0x30)	–	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

11.9.2 TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
0x33 (0x53)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – FOC0A: Force Output Compare A**

The FOC0A bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0A bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0A output is changed according to its COM0A1:0 bits setting. Note that the FOC0A bit is implemented as a strobe. Therefore it is the value present in the COM0A1:0 bits that determines the effect of the forced compare.

A FOC0A strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0A as TOP.

The FOC0A bit is always read as zero.

- **Bit 6 – FOC0B: Force Output Compare B**

The FOC0B bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0B bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0B output is changed according to its COM0B1:0 bits setting. Note that the FOC0B bit is implemented as a strobe. Therefore it is the value present in the COM0B1:0 bits that determines the effect of the forced compare.

A FOC0B strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0B as TOP.

The FOC0B bit is always read as zero.

- **Bits 5:4 – Res: Reserved Bits**

These bits are reserved bits in the ATtiny2313A/4313 and will always read as zero.

- **Bit 3 – WGM02: Waveform Generation Mode**

See the description in the “TCCR0A – Timer/Counter Control Register A” on page 82.

- **Bits 2:0 – CS02:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter. See Table 11-9 on page 86.

Table 11-9. Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter0, transitions on the T0 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

11.9.3 TCNT0 – Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
0x32 (0x52)	TCNT0[7:0]								TCNT0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Timer/Counter Register gives direct access, both for read and write operations, to the Timer/Counter unit 8-bit counter. Writing to the TCNT0 Register blocks (removes) the Compare Match on the following timer clock. Modifying the counter (TCNT0) while the counter is running, introduces a risk of missing a Compare Match between TCNT0 and the OCR0x Registers.

11.9.4 OCR0A – Output Compare Register A

Bit	7	6	5	4	3	2	1	0	
0x36 (0x56)	OCR0A[7:0]								OCR0A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register A contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0A pin.

11.9.5 OCR0B – Output Compare Register B

Bit	7	6	5	4	3	2	1	0	
0x3C (0x5C)	OCR0B[7:0]								OCR0B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register B contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0B pin.

22. Electrical Characteristics

22.1 Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0 mA
DC Current V_{CC} and GND Pins	200.0 mA

*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

22.2 DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
V_{IL}	Input Low Voltage except XTAL1 and $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5		$0.2V_{CC}$ $0.3V_{CC}$	V
V_{IH}	Input High-voltage except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$ (1) $0.6V_{CC}$ (1)		V_{CC} +0.5 (2)	V
V_{IL1}	Input Low Voltage XTAL1 pin	V_{CC} = 1.8V - 5.5V	-0.5		$0.1V_{CC}$	V
V_{IH1}	Input High-voltage XTAL1 pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.8V_{CC}$ (1) $0.7V_{CC}$ (1)		V_{CC} +0.5 (2)	V
V_{IL2}	Input Low Voltage $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 5.5V	-0.5		$0.2V_{CC}$	V
V_{IH2}	Input High-voltage $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 5.5V	$0.9V_{CC}$ (1)		V_{CC} +0.5 (2)	V
V_{IL3}	Input Low Voltage $\overline{\text{RESET}}$ pin as I/O	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5		$0.2V_{CC}$ $0.3V_{CC}$	V
V_{IH3}	Input High-voltage $\overline{\text{RESET}}$ pin as I/O	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$ (1) $0.6V_{CC}$ (1)		V_{CC} +0.5 (2)	V
V_{OL}	Output Low Voltage(3) (Except Reset Pin)(5)	I_{OL} = 20 mA, V_{CC} = 5V I_{OL} = 10mA, V_{CC} = 3V			0.8 0.6	V V
V_{OH}	Output High-voltage(4) (Except Reset Pin)(5)	I_{OH} = -20 mA, V_{CC} = 5V I_{OH} = -10 mA, V_{CC} = 3V	4.2 2.4			V V
I_{IL}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin low (absolute value)			1(6)	µA
I_{IH}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin high (absolute value)			1(6)	µA
R_{RST}	Reset Pull-up Resistor		30		60	kΩ
R_{pu}	I/O Pin Pull-up Resistor		20		50	kΩ