Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE0624 – Laboratorio de Microcontroladores
II ciclo 2022

Laboratorio 1

Introducción a microcontroladores y manejo de GPIOS

Kenny Wu Wen C08592
Grupo 01

Profesor: MSc. Marco Villalta Fallas

2 de septiembre de 2023

Índice

Índice de figuras

1. Resumen

En este laboratorio se va a tener una introducción del microcontrolador PIC, en especifico el PIC12F683. Se va a utilizar los pines del PIC como GPIOs (*General Purpose Input Outputs*) con el fin de armar un simulador de dados con leds.

El simulador de dados va a tener seis leds que se van a encender de manera pseudo-aleatoria entre el número 1 y 6, cuando se pulse un botón, los leds encendidos se van a mantener así por un periodo de tiempo.

Para lograr el simulador primeramente se va a tener que revisar la hoja del microcontrolador y como funciona cada pin de este; para luego escribir el código en C, el firmware, para el microcontrolador.

Para finalizar, se va a armar el circuito con los componentes necesarios para el simulador y para la protección de los componentes y microcontrolador para hacerle pruebas.

El repositorio para el laboratorio es el siguiente: GIT

2. Nota teórica

2.1. PIC12F683

Las siglas en ingles PIC significan Controlador de Interfaz Programable, este es un circuito electrónico programable a muchas tareas. Pueden funcionar como temporizadores o controladores de producción y mucho más. Se utiliza mucho en sistemas de alarma, sistemas de control de computación, telefonos y en general en cualquier dispositivo electrónico [1].

Existen muchos tipos de PIC, para este laboratorio se trabajara con el PIC12F683 y algunas de sus características son:

- Capacidad de manejo de interrupciones
- Temporizador Watchdog
- Modo de ahorro de energía
- Convertidor analogico digital
- Seis pines I/O
- PWM
- ICSP (Programación In-Circuit Serial)

Tenemos la tabla resumen del funcionamiento de cada pin:

2.2. Entradas/Salidas de Propósito General (GPIO)

Los GPIO son pines de señales digitales que se encuentran en un circuito electrico/integrado para usarse como entrada o salida o ambos por medio de software/firmware, sin embargo, no tienen una función especifica [2].

En el PIC12F683, tenemos seis pines que se pueden utilizar como GPIOs como lo indica la hoja de fabricante, pero se debe tomar en cuenta que el GPIO3 solo funciona como entrada y no salida, al ser un registro en el cual solo se puede leer su valor y no modificarlo por medio del software.

Para configurar los pines como salidas tenemos primero que configurar el registro **TRIOSTATE** en modo salida exceptuando el bit 3 que solo es entrada como se observa en la figura 3.

8-Pin Diagram (PDIP, SOIC)

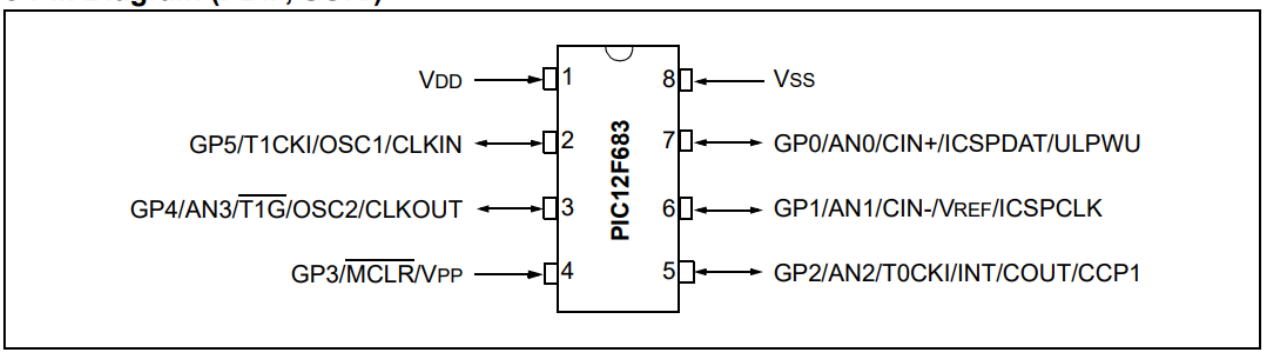

Figura 1: Diagrama del PIC12F683 y sus ocho pines

TABLE 1: 8-PIN SUMMARY

I/O	Pin	Analog	Comparators	Timer	CCP	Interrupts	Pull-ups	Basic
GP0	7	AN0	CIN+	—	—	IOC	Y	ICSPDAT/ULPWU
GP1	6	AN1/VREF	CIN-	—	—	IOC	Y	ICSPCLK
GP2	5	AN2	COUT	T0CKI	CCP1	INT/IOC	Y	—
GP3(1)	4	—	—	—	—	IOC	Y(2)	MCLR/VPP
GP4	3	AN3	—	T1G	—	IOC	Y	OSC2/CLKOUT
GP5	2	—	—	T1CKI	—	IOC	Y	OSC1/CLKIN
—	1	—	—	—	—	—	—	VDD
—	8	—	—	—	—	—	—	VSS

Note 1: Input only.
2: Only when pin is configured for external MCLR.

Figura 2: Tabla resumida del funcionamiento de los pines

REGISTER 4-2: TRISIO GPIO TRI-STATE REGISTER

U-0	U-0	R/W-1	R/W-1	R-1	R/W-1	R/W-1	R/W-1
—	—	TRISIO5(2,3)	TRISIO4(2)	TRISIO3(1)	TRISIO2	TRISIO1	TRISIO0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented**: Read as '0'

bit 5:4 **TRISIO<5:4>:** GPIO Tri-State Control bit
1 = GPIO pin configured as an input (tri-stated)
0 = GPIO pin configured as an output

bit 3 **TRISIO<3>:** GPIO Tri-State Control bit
Input only

bit 2:0 **TRISIO<2:0>:** GPIO Tri-State Control bit
1 = GPIO pin configured as an input (tri-stated)
0 = GPIO pin configured as an output

Note 1: TRISIO<3> always reads '1'.
2: TRISIO<5:4> always reads '1' in XT, HS and LP OSC modes.
3: TRISIO<5> always reads '1' in RC and RCIO and EC modes.

Figura 3: Registro TRI-STATE

Luego, ya se va a poder controlar/modificar los valores del registro **GPIO** al ser salidas, teniendo en cuenta siempre que el bit 3 es de lectura como se muestra en la figura 4.

REGISTER 4-1: GPIO: GENERAL PURPOSE I/O REGISTER

U-0	U-0	R/W-x	R/W-0	R-x	R/W-0	R/W-0	R/W-0
—	—	GP5	GP4	GP3	GP2	GP1	GP0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented**: Read as '0'

bit 5-0 **GP<5:0>**: GPIO I/O Pin bit
1 = Port pin is > VIH
0 = Port pin is < VIL

Figura 4: Registro GPIO

2.3. LEDs

Los LEDs (*light-emitting diode*) son diodos emisores de luz. Contienen dentro un semiconductor el cual al recibir una diferencia de potencial continua, produce luz, fenómeno conocido como electro luminiscencia [3].

Cuando se conecta un diodo LED, la tensión que aguanta es normalmente de 2 V, en caso de ser mayor se debe colocar una resistencia en serie para no quemar el LED.

En este laboratorio se van a usar LEDs amarillos de 3mm con un costo de 100 colones por unidad. Funcionan a 2V y 20mA [4].

Como los pines del PIC entregan una tensión de 5 V aproximadamente, por lo tanto la resistencia de protección va ser:

$$R_{led} = \frac{5-2}{20 \times 10^{-3}}$$
$$\boxed{R_{led} = 150\Omega} \tag{1}$$

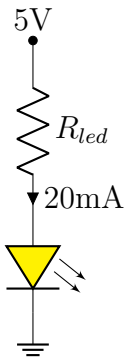

Figura 5: LED conectado a un pin del PIC

Para dos LEDs en serie con los mismas parámetros:

$$R_{led} = \frac{5 - 2 \cdot 2}{20 \times 10^{-3}}$$
$$\boxed{R_{led} = 50\Omega} \tag{2}$$

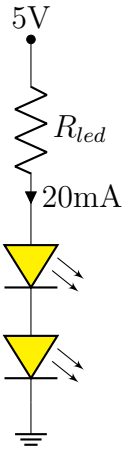

Figura 6: LED conectado a un pin del PIC

2.4. Botón

Para el botón conectado al pin de GPIO3 del PIC, se va a utilizar configuración pull-up, osea, estará lógica 1 cuando no se pulse el botón y en lógica 0 cuando se pulse. Tambíen se le conoce como lógica negativa.

Como se observa en la figura 7, la señal que va a recibir el GPIO3 va ser alta hasta que se pulse el botón. Utilizar esta configuración nos permite proteger el microcontrolador para que la corriente que le llegue sea cercana a cero. Para eso, vamos a utilizar una resistencia de:

$$R_{boton} = 1k\Omega \tag{3}$$

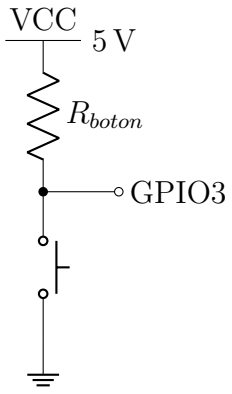

Figura 7: Configuración pull-up

Tambien hay que considerar el rebote al presionar el botón, ya que no es ideal el botón, por ende existen varios métodos para eliminar este efecto, tanto por software como hardware. Para este laboratorio se utilizara un método por hardware que es incluir un capacitor para que

existe un tiempo de subida y bajada de tensión de salida al presionar el boton para así eliminar el efecto de rebote [5].

Para calcular el valor del capacitor en paralelo al botón, hay que establecer el valor de T, en nuestro caso si queremos sea de $T = 100\mu$s, implica que $C = \frac{R}{T} = \frac{100 \times 1 - ^{-6}}{1 \times 10^3} = 100$nF.

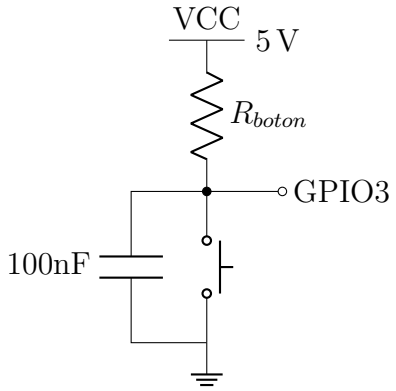

Figura 8: Configuración pull-up

2.5. SimulIDE

SimulIDE es un programa pensado para la simulación de circuitos generales y microcontroladores como PIC, AVR y Arduinos. Este simulador es muy completo y bastante potencial. En nuestro caso, vamos a utilizar el microcontrolador PIC para el laboratorio y para eso se va a utilizar el módulo gpsim que se encargara de los requisitos necesarios para tener disponible el PIC [6].

2.6. Compilador C de dispositivo pequeños (SDCC)

Como dice sus siglas, es un compilador de software libre para C enfocado en microcontroladores de 8 bits (PIC es uno de ellos). En 2007, se volvió el único compilador de lenguaje C de código abierto para microcontroladores Intel 8051 y compatibles. Es uno de los compiladores más descargados y usados para su enfoque. El compilador funciona para Linux, MacOS y Windows, todos 32 y 64 bits [7].

2.7. Generación de números aleatorios

Existen muchas formas de generar números aleatorios tanto con software como con hardware. Para este laboratorio se va a generar números aleatorios con software con contadores.

Para eso se va a utilizar una variable que va a aumentando conforme aumenta el tiempo y cuando se pida el valor del número aleatorio este va a retornar el valor que llego el contador y se va a reiniciar como se observa en la figura 9.

2.8. Algoritmo de simulador de dados

El algoritmo va a iniciar con la etapa de inicialización de los registros TRISIO y GPIO. Para TRISIO poner todos los pines como salidas y GPIO poner todo los pines en bajo. Luego, sigue la etapa de inicialización de variables en este caso ocuparemos tres: time, random y value. Después, vamos a tener el código principal que tiene tres fases, la primera es reiniciar las variables a su valor inicial, la segunda es que ocurre cuando se presiona el botón y que ocurre cuando no se presiona.

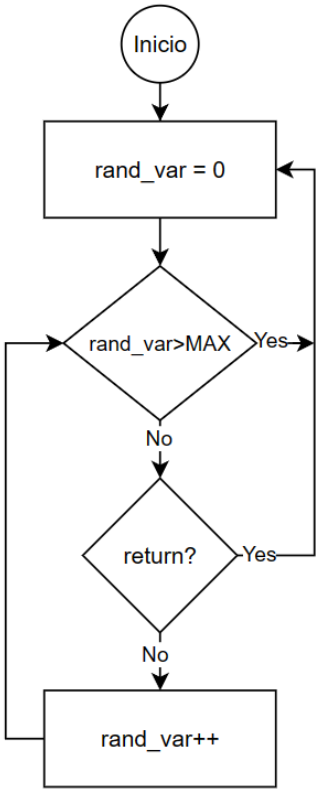

Figura 9: Diagrama de flujo del algoritmo simple de numero aleatorio basado en un contador

1. Cuando se presiona el botón, se va a escribir en el registro GPIO uno de los valores de value y luego se vuelve a poner en cero después de un delay.

2. Si no se presiono el botón, el valor de random seguirá aumentando y se reiniciara si llega a ser mayor a 6.

La lógica del algoritmo viene de que solo ocupa conectar 4 pines del PIC, ya que existe un patrón en los dados, por eso la variable value contiene seis patrones/valores guardados que son las seis caras de los dados.

El patrón del dado viene dado de la siguiente:

1. Caso 1: Solo se prende el LED central, osea 1 pin

2. Caso 2: Se prende dos LEDs superior derecha e inferior izquierda, y solo se necesita 1 pin para ambos LEDs

3. Caso 3: Se prende tres LEDs, osea los pines de Caso 1 y Caso 2.

4. Caso 4: Se prende cuatro LEDs, los de Caso 2 y los LEDs superior izquierda e inferior derecha con un pin.

5. Caso 5: Se prende el Caso 4 junto al Caso 1.

6. Caso 6: Se prende el Caso 4 junto a dos LEDs laterales centrales con un pin.

En la figura 11, se puede observar lo mencionado anteriormente, en donde cada par de LEDs están conectados en serie, exceptuando el central (rojo)

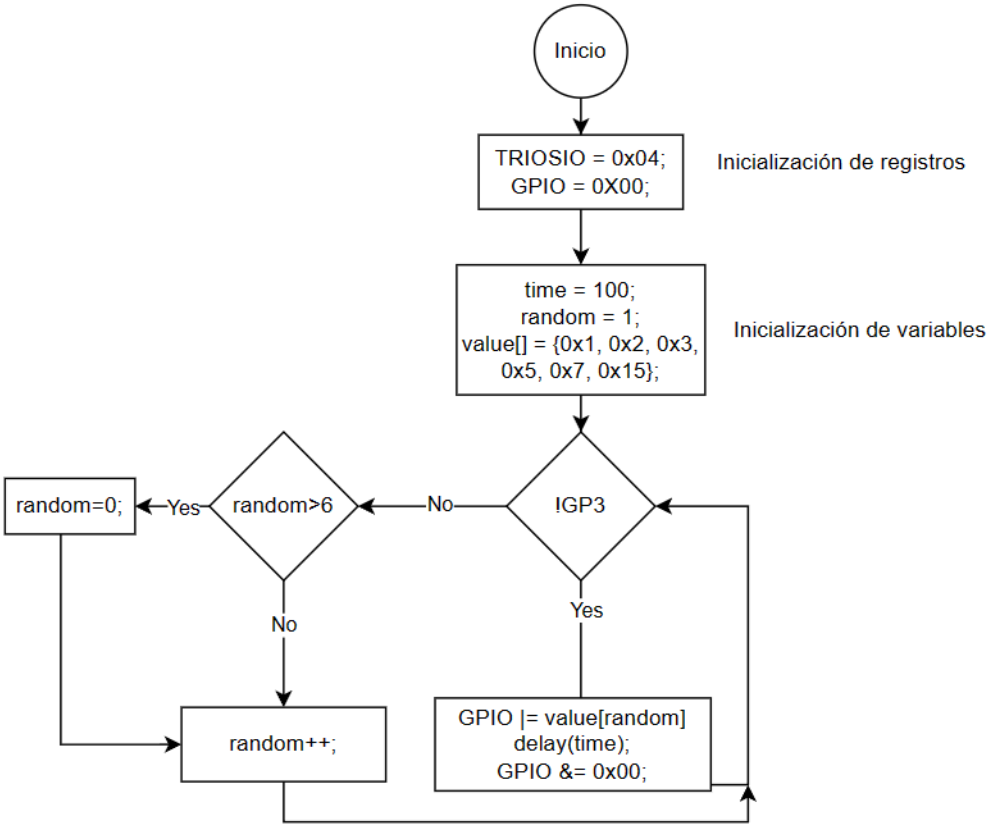

Figura 10: Algoritmo de simulador de dados

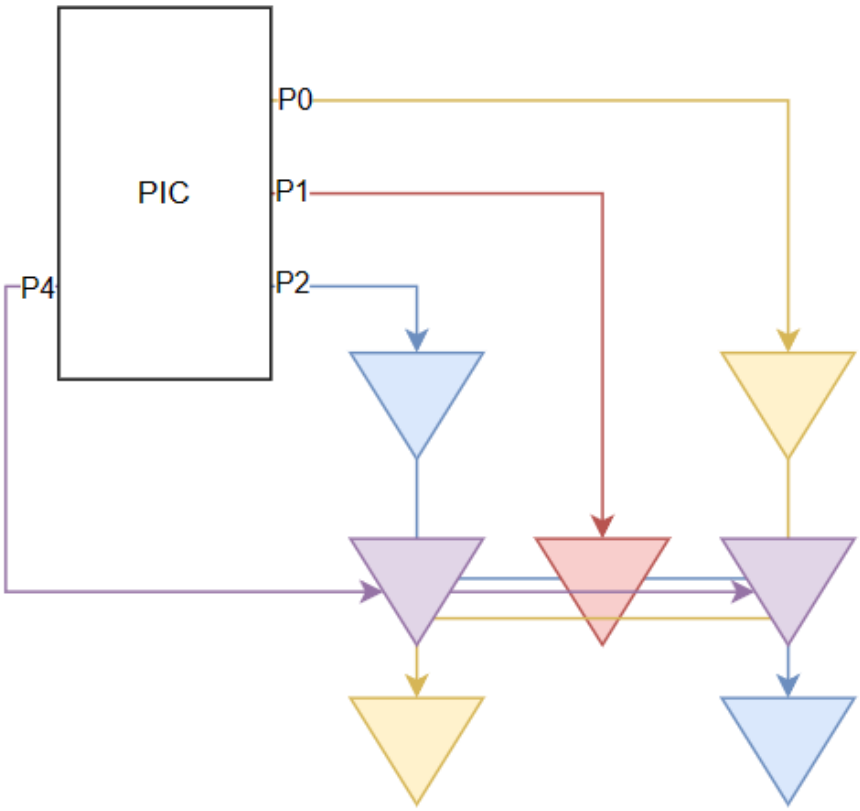

Figura 11: Dado patrón

3. Desarrollo/Análisis de resultados

3.1. Análisis de funcionalidad electrónica

Cuando el circuito está sin presionarse el boton, como se observa en la figura 12, la corriente por el PIC del ping P3 es casi nula pero con tensión en alta como se esperaba. También se puede apreciar una respuesta transitoria al prender el simulador, ya que el capacitor se empieza a cargar.

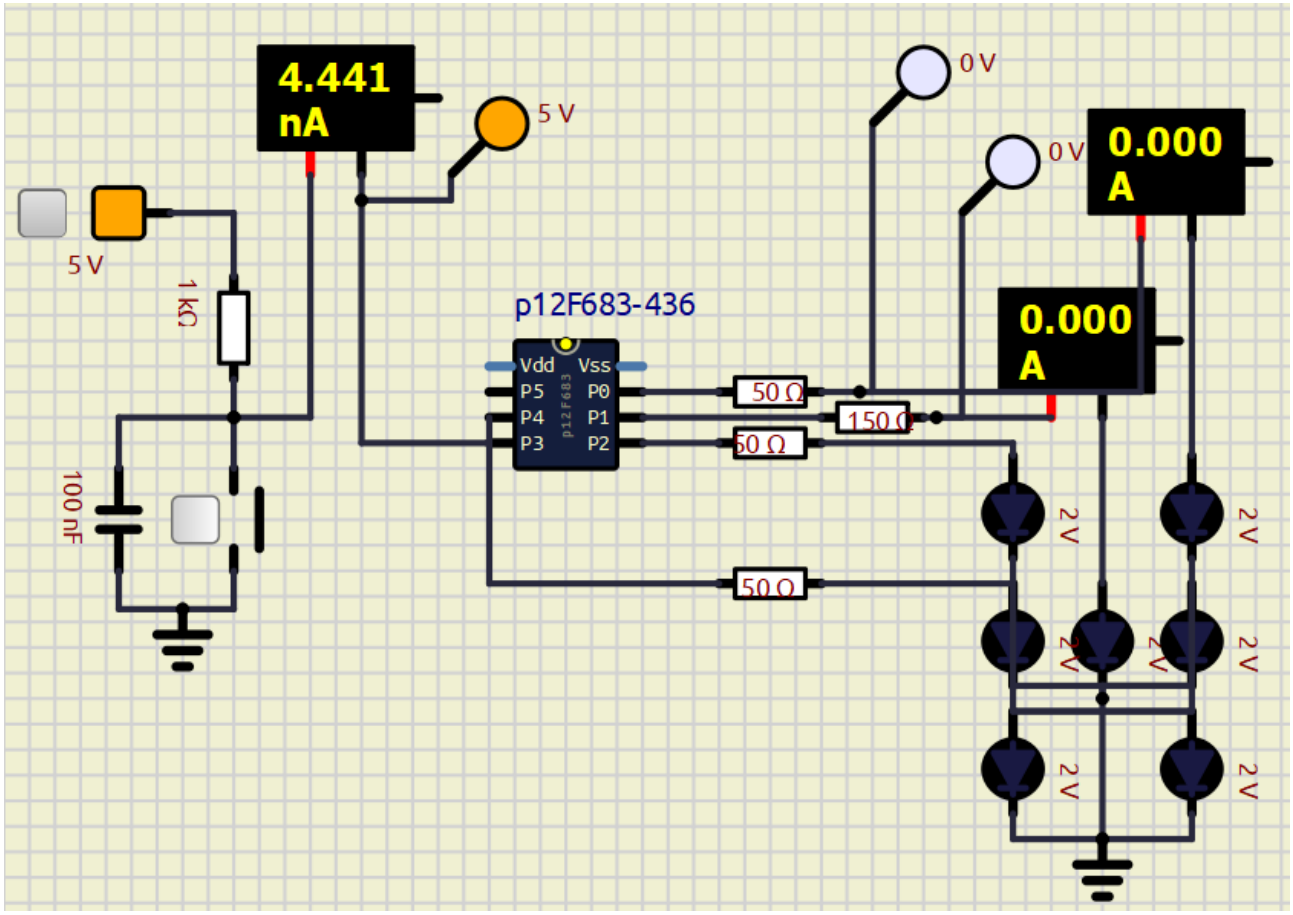

Figura 12: Circuito simulador de dados con botón sin presionar

Cuando se presiona el botón por un instante, se observa en la figura 14 que se muestra en los LEDs el patrón de cara de 3 del dado y los valores de corriente no sobre pasan al máximo valor de los LEDs de 20mA y la tensión es 4.01 V para dos LEDs y 2.01V para un LED, que son valores esperados y cercanos a los teóricos, la diferencia puede deberse a resistencias internas de los LEDs.

También se puede observar en la figura 13 que la señal de subida del pin GPIO3, al presionar el botón llega a su valor máximo en 500μs, el cuál tiene sentido ya que tenemos un constante de tiempo de 100μs.

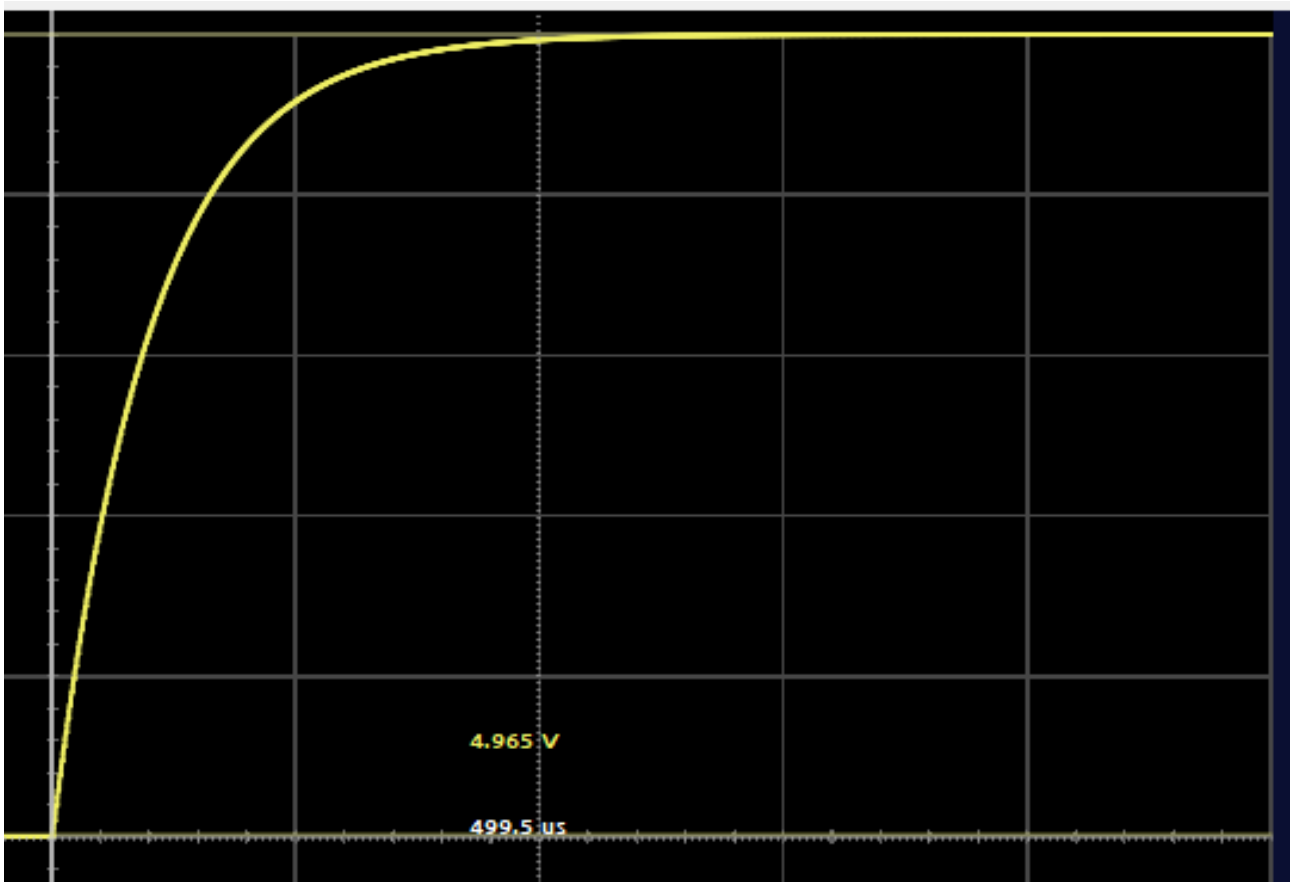

Figura 13: Curva de tensión del botón

3.2. Análisis de la funcionalidad del programa

Como se observa en la figura 14, al presionarlo se activaron cinco LEDs de forma aleatoria. También, se debe tomar en cuenta que un LED siempre va a estar prendido ya que el dado va de 1 a 6.

Cuando se presiona el botón por un momento, se muestra en los 5 LEDs conectados al PIC, un valor aleatorio ed 1 a 6 por un momento y luego vuelve a apagarse los LEDs conectados al microcontrolador.

Si se mantiene presionado el botón, se observa que los LEDs van a una secuencia de 1 a 6, perdiendo su aleatoriedad, por ende, la aleatoriedad viene de darle al botón un momento aleatorio o dependiendo del tiempo.

En sección 5.2, se podrá observar todas las caras del dado obtenidos con el simulador.

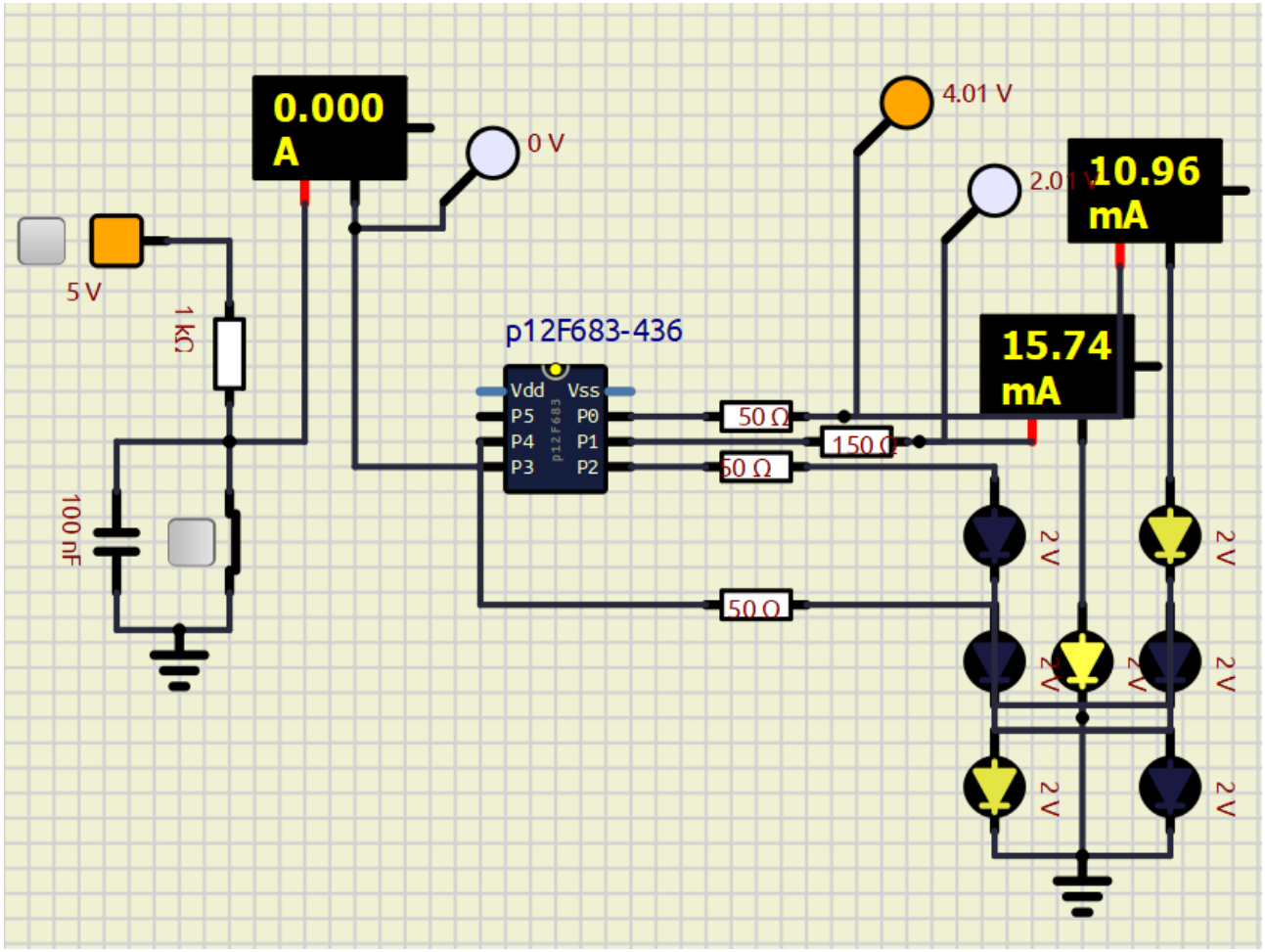

Figura 14: Circuito simulador de dados con botón presionado

4. Conclusiones y recomendaciones

- Siempre calcular la resistencia de protección para los LEDs.
- Tomar en cuenta la corriente o posibles cortos al microcontrolador, con el fin de cuidarlo.
- Tener un orden en los cables y un esquemático de como serán las conexiones antes de hacerlas.
- A la hora de escribir el firmware, tener orden en el código para entender bien que hace cada sección del código.
- Se logro crear un algoritmo pseudo-aleatorio basado en el momento que se presiona el botón en un instante y no seguido.
- En el SimulIDE, se pueden observar los valores de los registros del PIC en tiempo real, para facilitar el debugeo.

Referencias

[1] "What is a pic microcontroller? what can it do?" TechnologyStudent.com, 2023. [Online]. Available: https://www.technologystudent.com/pics/picgen1.html

[2] "Gpio (general purpose input/output) definition," TechTerms.com, 2023. [Online]. Available: https://techterms.com/definition/gpio

[3] "¿qué es un led? tipos de led, aplicaciones y usos." Visual Led, 2023. [Online]. Available: https://visualled.com/glosario/que-es-un-led/

[4] "Led 3mm amarillo," DBU Electronics, 2023. [Online]. Available: https://www.dbuelectronics.cr/leds/624-led-3mm-amarillo.html

[5] C. M. Maxfield", "Cómo implementar el rebote de hardware para interruptores y relés," https://www.digikey.com/es/articles/how-to-implement-hardware-debounce-for-switches-and-relays, 2021, [Consultado el 30 de agosto de 2023].

[6] "Simulide: Simulador de diseños de pic y arduino," Neoteo, 2023. [Online]. Available: https://www.neoteo.com/simulide-simulador-de-disenos-de-pic-y-arduino/

[7] "Compilador sdcc," Wikiwand, 2023. [Online]. Available: https://www.wikiwand.com/es/Compilador_SDCC

5. Apéndice

5.1. Precios de componentes

Cuadro 1: Especificaciones de cada interfaz físico

Componente	Precio (Colones)	Cantidad
PIC	2000	1
1K ohm	200	1
50 ohm	300	3
150 ohm	300	1
Botón	200-500	1
Cerámico 0.1uF	80	2

5.2. Salidas del simulador

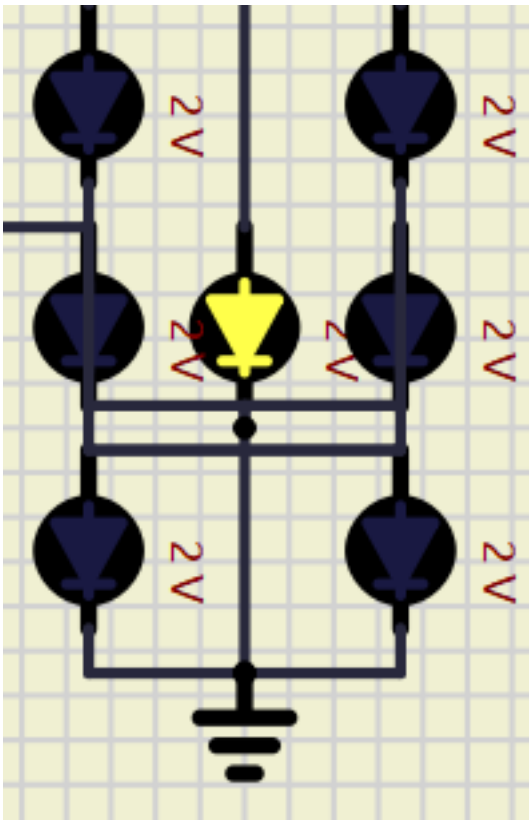

Figura 15: Cara uno del dado

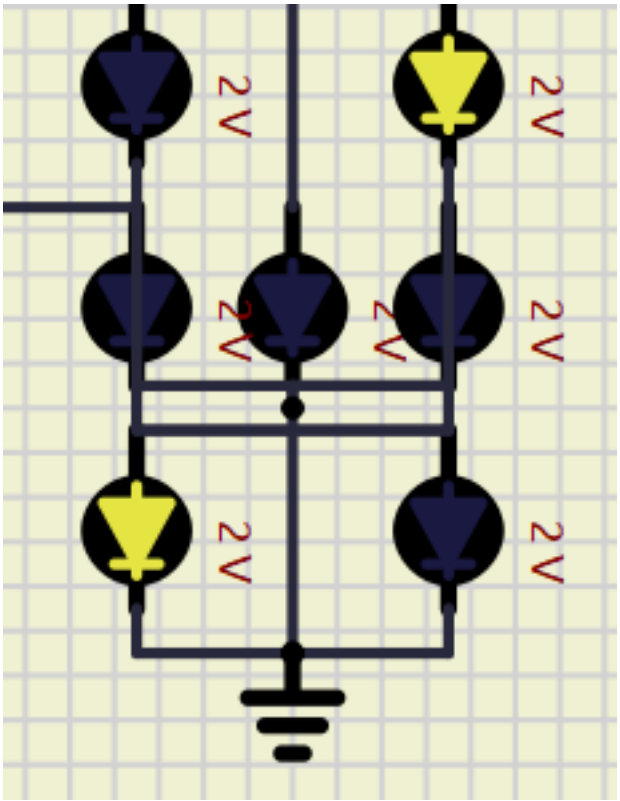

Figura 16: Cara dos del dado

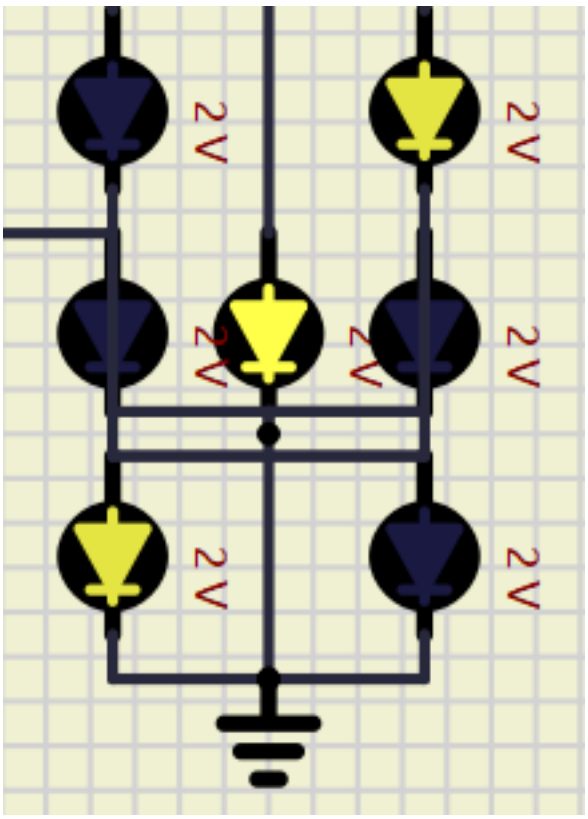

Figura 17: Cara tres del dado

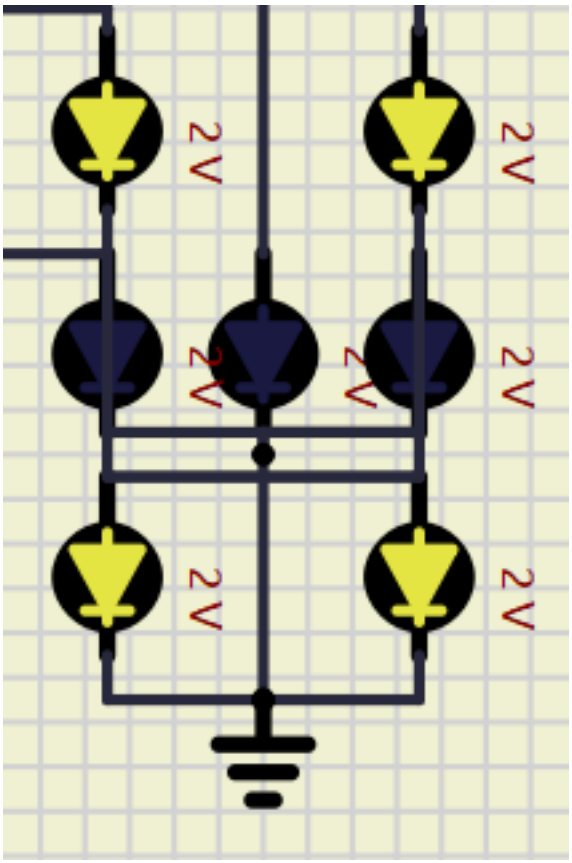

Figura 18: Cara cuatro del dado

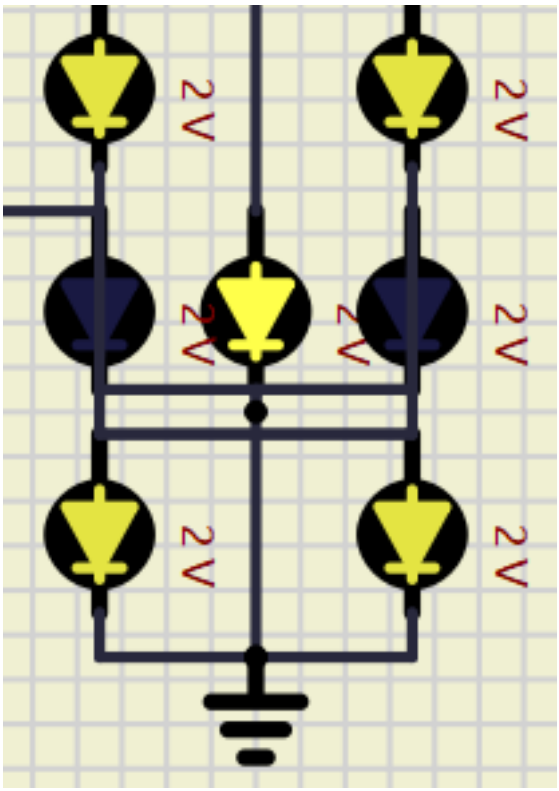

Figura 19: Cara cinco del dado

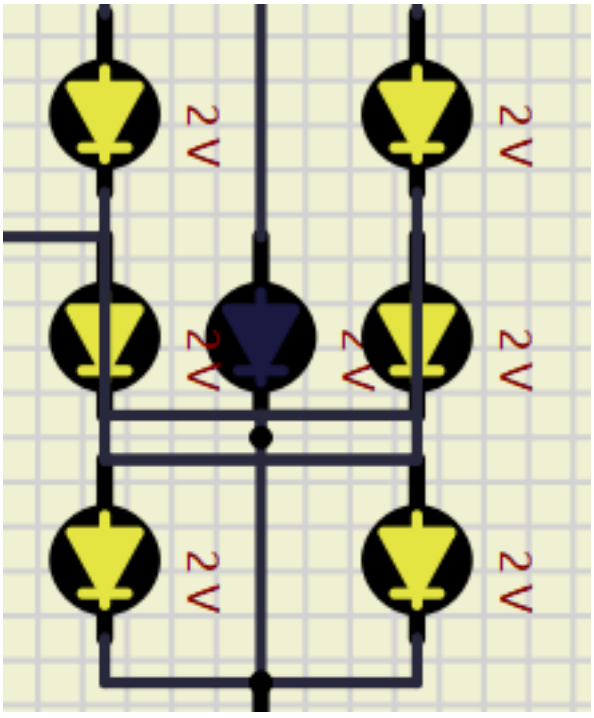

Figura 20: Cara seis del dado