Texto

El contenido generado por IA puede ser incorrecto.

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computadores

Curso: Fundamentos de Sistemas Computacionales

Primer Proyecto “Light Hunt”

1 semestre 2025

Prof. Milton Villegas Lemus

Asistente: Asly Barahona

Estudiantes: María José Solano Cordero,2024314761

Emmanuel García Quiros,2025065962

Fecha de Entrega:26 de mayo del 2025

**Introducción**

En la siguiente documentación presentamos nuestro proyecto “Light Hunt” inspirado en la propuesta de recrear “Cat and Mise” adaptado al aprendizaje de sistemas empotrados y con algunas diferencias explicadas más adelante en esta documentación. Para este proyecto creamos un prototipo de juguete que permite reaccionar ante patrones de luz mediante la presión de una botones.

Objetivo del Proyecto:

Implementar un sistema empotrado interactivo que utilice botones, luces y sonidos para crear un juego basado en reacciones a patrones de luces.

Importancia del Proyecto:

Este proyecto nos permite un desarrollo de habilidades en electrónica mediante los componentes utilizados y sus respectivas soldaduras, programación de una Raspberry Pi Pico/W que nos permitió controlar el juego y diseño de sistemas, aplicando conocimientos teóricos en un proyecto práctico.

Componentes y sus distintos usos en el proyecto:

Para los patrones de luces utilizamos diodos leds que nos permiten visualizar las secuencias correctamente y para corresponder a este patrón utilizamos botones.

Para la reproducción de sonidos característicos utilizamos el buzzer que nos permite tener un juego más entretenido y para el cambio de niveles utilizamos un dip switch.

Nivel de completitud del proyecto  
El prototipo implementado cumple con las siguientes funcionalidades:

* Generación de patrones aleatorios en la matriz de LEDs.
* Detección de pulsaciones de botones con retroalimentación visual (LEDs) y sonora (buzzer).
* Lógica de progresión entre niveles (sets) al completar games (5 aciertos por game).
* Reinicio del sistema mediante un interruptor externo.

Así como no se logró cumplir con:

* Selección de modos de juego (Principiante, Avanzado, Leyenda) mediante interruptores DIP.
* Visualización del puntaje en el display de 7 segmentos, manejado por el registro de corrimiento.

**Conclusiones**

1. La soldadura debe ser resistente, pero a su vez tratarla con cuidado.
2. El estaño viejo pierde capacidad para soldar correctamente.
3. El calor del cautín puede pasar atreves de todo el componente a soldar.
4. Un cautín sucio o con residuos adheridos dificulta la unión de componentes.
5. Comprar componentes diferentes a los solicitados puede afectar de mayor o menor medida la realización del proyecto.
6. La elección de componentes accesibles (como la Raspberry Pi Pico W) facilitó la replicabilidad del proyecto, aunque algunos detalles de conexión requirieron ajustes manuales.
7. El proyecto integra hardware y software de manera efectiva para crear un juego interactivo.
8. La retroalimentación visual (LEDs) y auditiva (Buzzer) mejora la experiencia del usuario.
9. El display 7 segmentos muestra el puntaje en hexadecimal de manera efectiva.
10. Cada componente debe ser alimentado con su debida corriente para evitar quemarlos.
11. Los leds más nuevos tendrán un mayor brillo mejorando la experiencia, mientras que los viejos habrán perdido la intensidad del brillo.
12. La base por utilizar para el hardware es duradera y fácil de manejar para el cableado.
13. El Hardware es agradable a la vista para el usuario.
14. El control de dos acciones para poder recibir el botón y ver los leds sin congelar no podemos utilizar solo “time”.
15. El uso de MicroPython permite una programación eficiente en la Raspberry Pi Pico W.
16. El registro de corrimiento simplifica la conexión con el display de 7 segmentos.
17. La aleatoriedad en los patrones de LEDs hace el juego impredecible.
18. Crear animaciones más elaboradas para celebrar logros.
19. El proyecto cumple con los objetivos de aprendizaje de sistemas embebidos.
20. La Raspberry Pi Pico W es adecuada para proyectos de esta envergadura.
21. El manejo de interrupciones podría mejorar la eficiencia del código.
22. El proyecto fomenta el trabajo en equipo y la resolución de problemas.
23. La documentación es esencial para el mantenimiento del proyecto.
24. El uso de PWM en el buzzer permite variedad de sonidos.
25. El prototipo demuestra la aplicación práctica de conceptos teóricos.
26. El proyecto es escalable y puede extenderse con nuevas funcionalidades.
27. La cooperación de todos los integrantes del grupo es esencial para la elaboración de proyectos.
28. Identificar y recordar la funcionalidad de los cables facilita el trabajo de armar el circuito.
29. El correcto entendimiento de los circuitos permite la correcta elaboración del trabajo.
30. Cada prueba del sistema eléctrico debe hacerse con cuidado para no quemar los componentes.
31. El display necesita resistencias de bajo valor para proyectar bien la luz de sus leds y visualizar correctamente los datos que se programan.
32. Si la batería está gastada puede proporcionar menos luz en los led.
33. Cables demasiados largos, delgados o con mala conexión pueden causar caídas de voltaje adicionales.
34. En entornos con ruido eléctrico la señal puede afectar el rendimiento.
35. Los botones de encendido están soldados al revés por tanto están intercambiadas sus funciones.

**Recomendaciones**

1. Soldar acompañado o con ayuda para mayor facilidad.
2. Utilizar estaño comprado recientemente para mejor soldadura y unión de piezas.
3. Usar pinzas para sostener los componentes durante la soldadura.
4. Esterilizar el cautín antes de cualquier intento de soldar.
5. Verificar el código de los componentes antes de comprar para evitar compras erróneas.
6. Documentar esquemáticos detallados del circuito (con herramientas como Fritzing o KiCad) para futuras revisiones o colaboraciones, asegurando consistencia en las conexiones.
7. Revisar hoja de datos y códigos de los componentes para su correcto uso.
8. Documentar claramente las conexiones para facilitar futuras modificaciones.
9. Incluir más sonidos personalizados para cada acción.
10. Implementar un sistema de guardado de puntaciones altas.
11. Conseguir leds nuevos con una iluminación llamativa.
12. Usar cajas que no se utilicen, para crear el hardware.
13. Decorar el hardware para hacerlo agradable y llamativo.
14. Para recibir los botones y visualizar los leds debemos utilizar utime para que no se congele.
15. Explorar más bibliotecas de MicroPython para optimizar el código.
16. Documentar el protocolo de comunicación con el registro.
17. Asegurar que la generación de patrones sea equilibrada.
18. La animación de LEDs y sonidos refuerzan la retroalimentación.
19. El proyecto cumple con los objetivos de aprendizaje de sistemas embebidos.
20. Incluir más ejercicios prácticos en futuras iteraciones.
21. Explorar otras placas para comparar rendimiento.
22. Implementar interrupciones para los botones.
23. Realizar sesiones de retroalimentación entre equipos.
24. Usar herramientas como Sphinx para generar documentación automática.
25. Experimentar con más frecuencias y duraciones.
26. Incluir un manual de usuario detallado.
27. Plantear futuras mejoras como pantallas LCD o conectividad Wifi.
28. Ponerse de acuerdo para distribuir el trabajo pendiente equitativamente.
29. Rotular los cables con un mismo objetivo o identificarlos por colores.
30. Aclarar dudas del proyecto con el profesor o asistente.
31. Poner resistencias de 330ohm es mejor que poner de 1kohm para que la corriente que pase sea la correcta y proyecte bien la luz del led.
32. Comprar baterías nuevas y cargadas para mejor rendimiento del proyecto.
33. Usar cables cortos y de calibre adecuado, y verifica las soldaduras o conexiones.
34. Usa filtros (condensadores) cerca de la alimentación del display.
35. Verificar el datasheet o el diagrama del botón para identificar los pines COM, NO y NC.

**Análisis de Resultados**

El resultado final del proyecto es bastante aferrado al funcionamiento final planteado en las instrucciones, sin embargo, hay fallas o características que no se lograron conseguir adecuadamente:

* El juego se enciende e inicia su funcionamiento al momento de conectar corriente proveniente de las baterías, activando el sistema, debido a falla en soldadura, el “O” alimenta el circuito y “I” apaga el circuito (funcionamiento inverso).
* Los leds se encendiendo de manera aleatoria y mantienen su luz durante el tiempo establecido dependiendo del nivel de dificultad, esperando respuesta de los botones.
* Al momento de tocar los botones según el led encendido se suma la puntuación correspondiente.
* Si el botón no se presiona durante el tiempo establecido o se presiona el incorrecto se cuenta como error.
* Si el jugador comete un error aun es capaz de seguir jugando, en lugar de acabar y reiniciar el juego.
* Los números mostrados en el 7 segmento no se observan con claridad o la línea correspondiente no se enciende, sin embargo, la lógica en la programación si se logró completar.
* Este fallo en el display se puede deber a que la batería usada ya está gastada mala conexión de cables o resistencias
* El sonido correspondiente a la acción de acierto, error o cambio de nivel se reproduce correctamente.
* Dependiendo si se cometió un acierto, error o cambio de nivel se reproduce una animación en los leds.
* Para reiniciar el juego es mediante el botón de balancín que alimenta el circuito, por tanto, es necesario cambiar de estado en el botón.
* La dificultad aumenta según el nivel en el que se encuentre el jugador.
* El hardware es agradable a la vista, sin embargo, tiene un agujero en un costado donde se puede observar las conexiones eléctricas.
* El cableado se encuentra ubicado correctamente y conduce la corriente sin a ver contacto entre cables.
* El hardware es agradable a la vista y resistente, pero al momento de presionar los botones se hunde levente la cubierta debido a que este está hueco por dentro.
* Se utilizan dos switches de encendido para enviar corriente al sistema, uno con 5 voltios y otro con 3.3 voltios. El switch para selección de niveles tuvo problemas a la hora de la soldadura e implementación física y no se logró incorporar, sin embargo, se avanzó la lógica de este en la programación.

**Bibliotecas utilizadas**

En este proyecto fueron utilizadas tres Bibliotecas que son fundamentales para el desarrollo correcto, vamos a conocer un poco de ellas:

* Machine

El propósito de esta Biblioteca es proporcionar acceso directo al hardware, incluyendo GPIO, PWM, ADC, UART, 12C, SPI y otras interfaces. Acá la utilizamos para acceder a los pines de la Raspberry Pi Pico/W y poder realizar el funcionamiento correcto.

**GPIO:** Configurar pines como entrada/salida.

**PWM:** Generar señales moduladas por ancho de pulso.

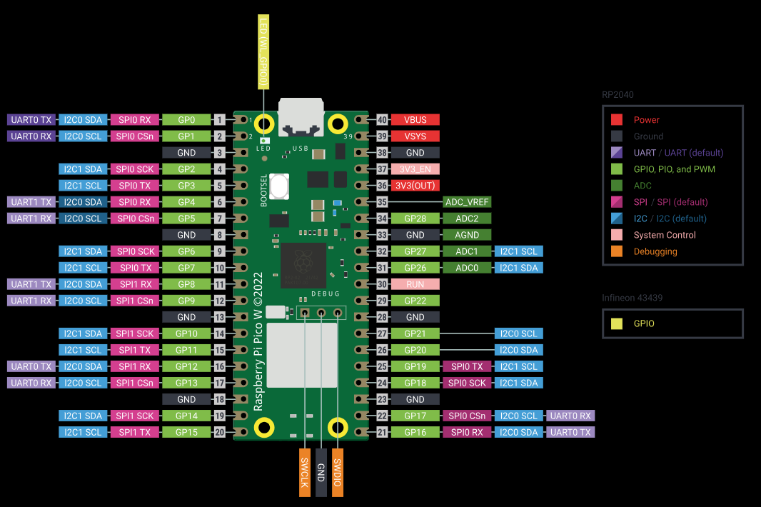


Figura 1. Pines Raspberry Pi Pico/W

* Utime

Para solucionar el problema del congelamiento que provocaba utilizar Time, se decidió utilizar utime la cual tiene como propósito manejar retardos, mediciones de tiempo y conversiones de formato de tiempo.

**sleep(seconds):** Pausa la ejecución durante seconds segundos.

**sleep\_ms(ms)** y **sleep\_us(us):** Retardos en milisegundos y microsegundos.

**ticks\_ms():** Obtiene el tiempo actual en milisegundos (útil para mediciones).

* Random

Esta Biblioteca tiene como propósito generar números aleatorios para patrones o decisiones no deterministas, para este proyecto fue utilizada para la realización de las secuencias de leds.

**randint(a, b):** Entero aleatorio entre a y b (incluidos).

**random():** Float aleatorio en el rango [0.0, 1.0).

**choice(seq):** Selecciona un elemento aleatorio de una secuencia.

Texto

El contenido generado por IA puede ser incorrecto.

Figura 2. Ejemplo de uso de Random

**Diagramas**

* Diagrama de Arquitectura

Este diagrama representa la arquitectura general del prototipo del juguete usando una Raspberry Pi Pico W. En el centro del esquema se encuentra la Raspberry, que actúa como núcleo de control del sistema, controlando tanto las entradas como las salidas.

Entradas al sistema:

Interruptor ON/OFF: Permite energizar o apagar completamente el sistema.

Botones (2×2): Usada por el jugador para responder a los patrones de luces encendidas.

DIP switches: Determinan el nivel de dificultad (Principiante, Avanzado o Leyenda).

Salidas del sistema:

LEDs 2×2: Presenta los patrones visuales que el jugador debe replicar.

Registro de corrimiento 74LS164: Componente intermedio que permite enviar datos desde la Raspberry al display

Display de 7 segmentos: Muestra el progreso del jugador en formato hexadecimal

Buzzer: Emite sonidos de retroalimentación según las acciones del jugador (acierto, error o cambio de nivel).

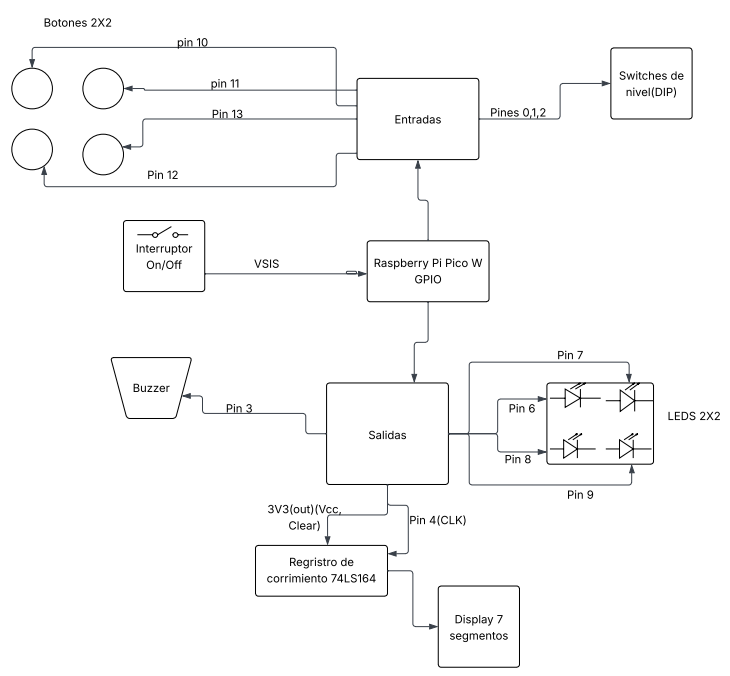


Figura 3. Diagrama de arquitectura del proyecto

* **Diagrama de Clases**

Diagrama

El contenido generado por IA puede ser incorrecto.Este esquema de clases del proyecto ilustra la estructura fundamental del sistema dividida en dos elementos clave: DisplayController y Main Game. DisplayController se ocupa de administrar el display de 7 segmentos, con un atributo para guardar el valor presente y un procedimiento (update\_display ()) para su actualización. Main Game incluye los componentes fundamentales del juego, tales como LEDs, botones, interruptores y buzzer, además de variables de estado (nivel y aciertos). Adicionalmente, incluye métodos fundamentales como leer\_nivel (), generar\_patron (), mostrar\_patron (), y funciones para sonidos y animaciones. La conexión entre ambas clases es de un solo sentido: El juego principal emplea DisplayController para mostrar las calificaciones, evidenciando modularidad del diseño.

Figura 4. Diagrama de Clases del proyecto.

**Referencias**

*Active Passive Buzzer*. (s. f.). Components101. <https://components101.com/misc/buzzer-pinout-working-datasheet>

alldatasheet.es. (n.d.). *74LS164 descarga*. <https://www.alldatasheet.es/datasheet-pdf/download/12631/ONSEMI/74LS164.html>

alldatasheet.es. (s. f.). *BIR-BM13E4G-2 Datasheet(PDF)*. BRIGHT LED ELECTRONICS CORP. <https://www.alldatasheet.es/datasheet-pdf/pdf/219258/BRIGHT/BIR-BM13E4G-2.html>

alldatasheet.es. (s. f.-b). *DIP-3 Datasheet(PDF)*. AG Electronica. <https://www.alldatasheet.es/datasheet-pdf/pdf/2015587/AGELECTRONICA/DIP-3.html>

alldatasheet.es. (s. f.-c). *LM7805 Datasheet(PDF)*. Fairchild Semiconductor. <https://www.alldatasheet.es/datasheet-pdf/pdf/82833/FAIRCHILD/LM7805.html>

alldatasheet.es. (s. f.-b). *TOS5161 Datasheet(PDF)*. List Of Unclassifed Manufacturers. <https://www.alldatasheet.es/datasheet-pdf/pdf/96548/ETC/TOS5161.html>

Alzate, O. M. F. (n.d.). *Hoja de datos raspberry pi pico*. CodigoElectronica. <http://codigoelectronica.com/blog/hoja-de-datos-raspberry-pi-pico>

*Cómo conectar un Buzzer a Arduino (zumbador) – Arduino Spain*. (s. f.). <https://sp.arduino-france.site/zumbador-arduino/>

De todo un poco. (2022, 17 febrero). *No quemes tu regulador de voltaje 7805 - Realiza los cálculos antes de conectar.* [Vídeo]. YouTube. <https://www.youtube.com/watch?v=wnfI1qtuKHk>

https://www.linkedin.com/in/jainish31. (2023, June 14). *How To Use Python For Random String Generation | LambdaTest*. LambdaTest. <https://www.lambdatest.com/blog/python-random-string/>

*LucidChart | Diagramming powered by Intelligence*. (s. f.). Lucidchart. <https://www.lucidchart.com/>

*machine — functions related to the hardware — MicroPython latest documentation*. (s. f.). <https://docs.micropython.org/en/latest/library/machine.html>

Ohms, & Ohms. (2024, 30 julio). *Cómo conectar un Push Button a Arduino - 330ohms*. 330ohms. <https://www.330ohms.com/blogs/blog/como-conectar-un-push-button-a-arduino>

Para Arduino. (2024, 18 marzo). *Cómo conectar un DIP switch deslizante con Arduino - Para Arduino*. <https://paraarduino.com/componentes/interruptores/dip-switch/como-conectar-un-dip-switch-deslizante-con-arduino/>

*Push button switch*. (s. f.). Components101. <https://components101.com/switches/push-button>

Raspberry Pi. (2024). *Pico-series microcontrollers - Raspberry Pi documentation*. Raspberrypi.com. <https://www.raspberrypi.com/documentation/microcontrollers/pico-series.html>

*Tinkercad*. (s. f.). Tinkercad. <https://www.tinkercad.com/dashboard>

*time – time related functions — MicroPython latest documentation*. (s. f.). <https://docs.micropython.org/en/latest/library/time.html>

**Diagrama, Esquemático

El contenido generado por IA puede ser incorrecto.Anexo**

Figura 5. Diagrama realizado para conexiones de circuito.