SISTEMAS EMBEBIDOS

Integrantes:

Daniela Basilio

Melissa Castro

Shirley Esquibel

Docente:

Ing. Ronald Solís

Proyecto:

"Whac-A-Mole" con Niveles

PAO II 2025 - 2026

Contenido

Objetivo general

Desarrollar un juego interactivo tipo "Whac-A-Mole" utilizando dos microcontroladores (PIC16F887 y ATmega328P), con niveles progresivos de dificultad, en el que el jugador debe golpear topos que aparecen aleatoriamente en un tiempo limitado, mientras escucha sonidos de retroalimentación y observa animaciones visuales.

Objetivos específicos

- Diseñar la lógica del juego para que, conforme el jugador acumule aciertos, el tiempo para golpear los topos se reduzca progresivamente, aumentando la velocidad de aparición de los topos con cada nuevo nivel alcanzado.
- Establecer una comunicación eficaz entre el PIC16F887 (controlador de la lógica del juego) y el ATmega328P (controlador de la pantalla y animaciones), permitiendo que el PIC envíe las instrucciones de qué topo debe aparecer y el ATmega muestre los resultados y animaciones correspondientes.
- Implementar efectos sonoros y animaciones en la pantalla para brindar retroalimentación al jugador en función de sus aciertos o fallos, incluyendo sonidos de victoria, derrota y animaciones de nivel, para mantener la dinámica y el interés en el juego.

Descripción técnica del juego

El juego Whac-A-Mole se basa en la interacción de un jugador con un conjunto de botones para golpear "topos" (representados como indicadores luminosos) que aparecen aleatoriamente en la pantalla. El objetivo es golpear tantos topos como sea posible antes de que se acabe el tiempo asignado para cada ronda.

El juego está estructurado en niveles, donde cada nivel tiene un tiempo límite en el que el jugador debe golpear los topos antes de que se acabe el tiempo. A medida que el jugador avanza en los niveles, la dificultad aumenta:

- **Niveles:** Los niveles se basan en la velocidad de aparición de los topos. En los niveles más altos, los topos se generan más rápido, reduciendo el tiempo disponible para golpear. El tiempo para golpear un topo comienza en 2 segundos. Si el jugador acierta un topo, el tiempo para la siguiente ronda se reduce en 30ms. Con cada 10 puntos (aciertos) obtenidos, el jugador sube de nivel. Cada subida de nivel reduce aún más el tiempo de aparición de los topos. El mínimo tiempo posible para golpear un topo es de 400ms. Al alcanzar el umbral de 10 puntos, el nivel se incrementa, y se realizan animaciones de nivel superior en la pantalla (toda encendida y un apagado diagonal de los LEDs).
- **Puntuación:** Cada vez que el jugador acierta un topo, se incrementa su puntuación. Si el jugador acumula una cantidad de aciertos determinada, sube de nivel.

- **Sonidos y Animaciones:** El juego utiliza una serie de sonidos y animaciones visuales para proporcionar retroalimentación al jugador sobre el éxito o el fallo, como el sonido de acierto, derrota y el "level up" con efectos visuales. Además, cada topo cuenta con un sonido especifico.

Capturas de la simulación en Proteus

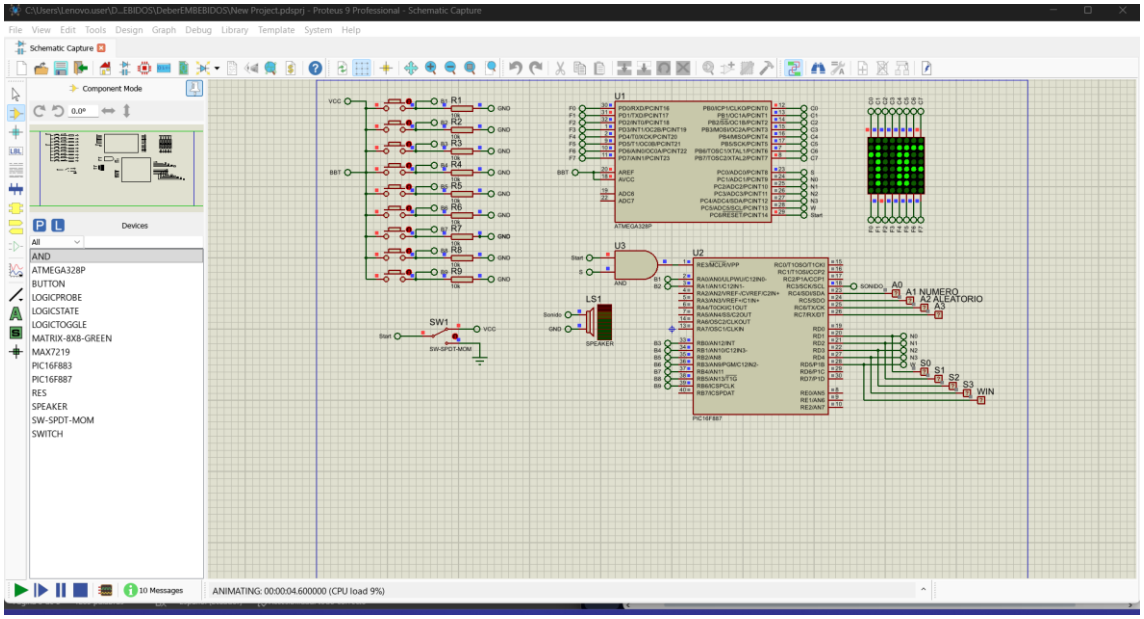

Ilustración 1: Inicio del juego (Presenta el mensaje "Empezamos").

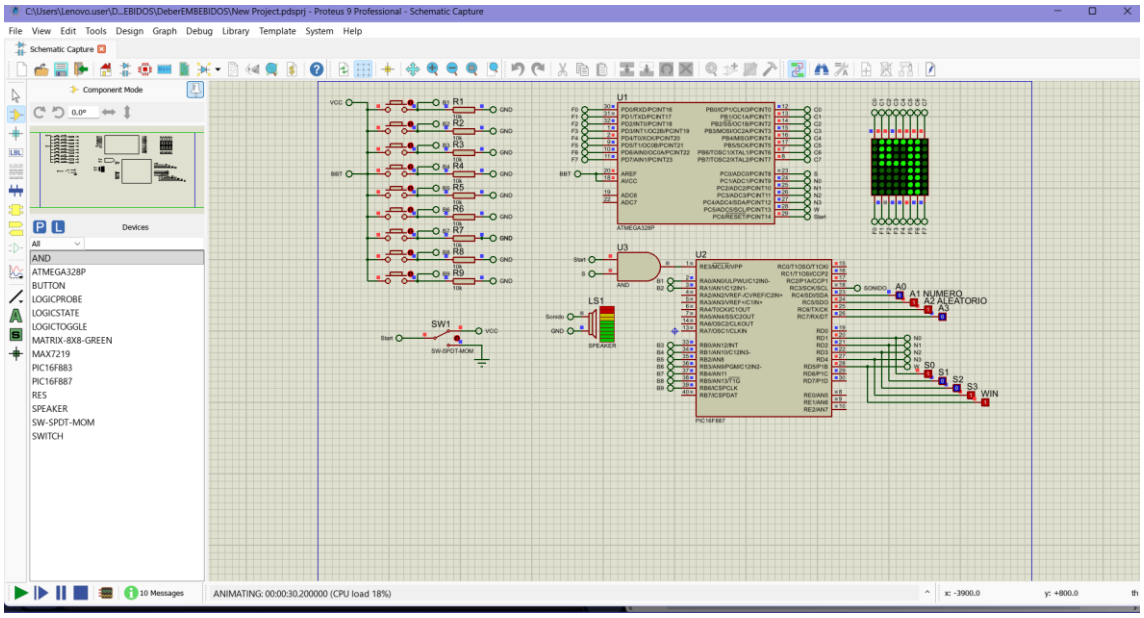

Ilustración 2: Muestra los topos como números.

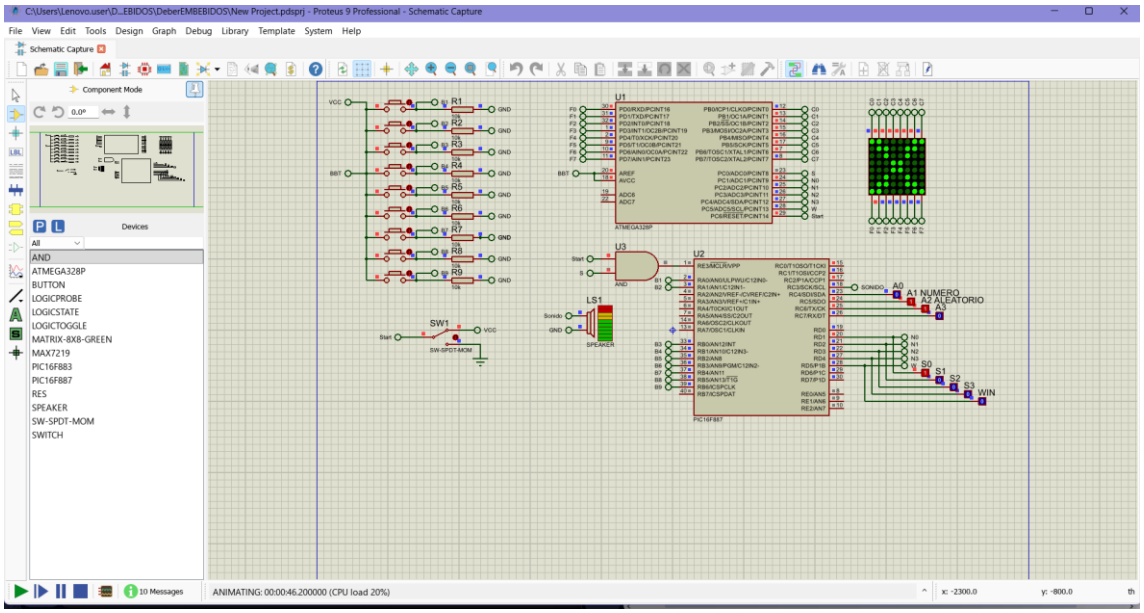

Ilustración 3: Muestra X cuando el jugador no presiona a tiempo el botón con el número indicado.

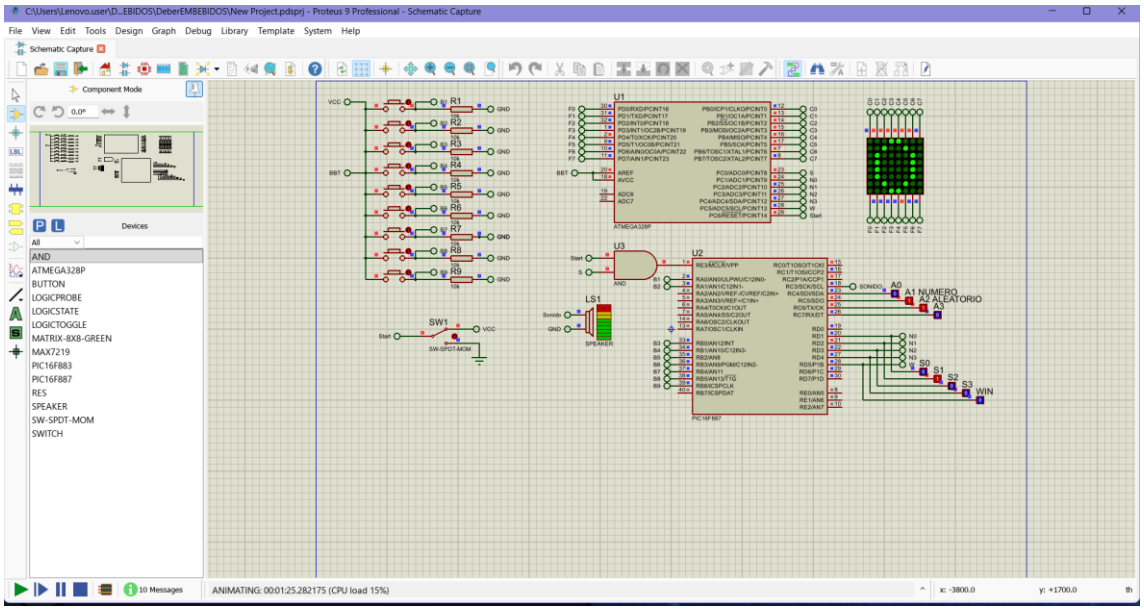

Ilustración 4: Muestra un círculo cuando el jugador acierta.

Explicación del código

- **ATmega328P**

Es el microcontrolador encargado de gestionar la pantalla y mostrar los resultados, animaciones y mensajes que el PIC envía. Este microcontrolador recibe datos desde el PIC a través de la comunicación de pines, en particular el pin RD5 (indicado como PC5 en el código) y el pin de inicio START_PIC (PC0).

Funciones Importantes:

- **Mostrar Imágenes:** El ATmega328P utiliza la función mostrarMatriz() para mostrar imágenes predefinidas (como 'X', 'O', '?') en la pantalla. Estas imágenes se configuran como matrices de 8 bits.
- **Animaciones:** La función animacionLevelUp() permite mostrar una animación de subida de nivel, apagando progresivamente los LEDs en forma diagonal.

Comunicación con el PIC16F887:

- El ATmega328P se comunica con el PIC16F887 para recibir las instrucciones del juego (como el número de topo y los comandos para la animación de nivel).
- La comunicación entre los microcontroladores se realiza a través de pines digitales. El PIC16F887 controla las filas de la pantalla y el ATmega328P controla las columnas, lo que permite la visualización de los números y las animaciones.

Esquema de Comunicación entre Microcontroladores (PIC16F887 y ATmega328P):

1. **Conexión de pines:** El PIC16F887 utiliza el puerto PORTD para controlar las filas de la pantalla (para mostrar los topos y los resultados) y el ATmega328P utiliza PORTB y PORTD para controlar las columnas y las animaciones.
2. **Protocolo de Comunicación:** El protocolo entre ambos microcontroladores es simple y basado en el estado de los pines:
    - El PIC16F887 envía información sobre el topo (número del topo o imágenes) y la instrucción para mostrar la animación o el resultado al ATmega328P.
    - El ATmega328P recibe esta información y la utiliza para actualizar la pantalla y las animaciones.
3. **Inicio del Juego:** El ATmega328P empieza con la animación "EMPECEMOS" y luego espera la señal del PIC16F887 para proceder con el juego (cuando el botón es presionado).
4. **Sincronización del Juego:** El PIC envía las actualizaciones de puntuación, tiempo y estado del juego (como acierto, derrota y nivel) al ATmega328P, que las refleja visualmente.

- ## PIC16F887

Es el microcontrolador principal que gestiona la lógica del juego, genera los topos, y recibe las interacciones del usuario mediante los botones.

Variables Principales:

- mole: Representa el número del topo que debe aparecer (de 1 a 9).

- buttonPressed: Almacena el valor del botón presionado por el jugador.
- timer: Lleva el control del tiempo para cada ronda.
- score: Lleva el puntaje total del jugador.
- level: Mantiene el nivel actual del jugador.
- base_timeout: Tiempo base para la generación de un topo en nivel 1.
- current_timeout: Tiempo para la ronda actual, que se ajusta conforme avanza el juego.
- timeout_reduction: Reducción en el tiempo de aparición del topo con cada acierto.

Funciones Importantes:

- **Funciones de Sonido:** El PIC16F887 utiliza la función Sound_Play para generar tonos musicales en función de los aciertos, derrotas, y eventos como la subida de nivel.
- **Funciones de Animación:** Utilizan la variable PORTD para controlar qué LEDs están encendidos, representando las imágenes de los topos (por ejemplo, CIRCULO (acierto), PERDER, etc.) y la animación de subida de nivel.
- **Botones:** La función check_buttons() con rebote asegura que el jugador pueda presionar un botón correctamente y realiza un seguimiento de cuál se ha presionado.

Secuencia del Juego:

1. **Generación de un Topo:** Un número aleatorio (entre 1 y 9) es asignado a la variable mole, representando qué topo debe ser mostrado.
2. **Tiempo de Espera:** El PIC espera que el jugador presione el botón correspondiente al topo dentro de un tiempo determinado (current_timeout).
3. **Comprobación de Acierto:** Si el jugador presiona el botón correspondiente antes de que se acabe el tiempo, se aumenta la puntuación y se reduce el tiempo para la siguiente ronda.
4. **Nivel Up:** Al alcanzar 10 puntos, el nivel sube, y el tiempo para aparecer los topos se reduce, aumentando la dificultad.
5. **Fin de la Ronda:** Si el jugador no acierta el topo dentro del tiempo o presiona un botón incorrecto, se reproduce una melodía de derrota y se reinicia el juego.

Descripción del esquema de comunicación entre microcontroladores

1. **Conexión de pines**: El PIC16F887 utiliza el puerto PORTD para controlar las filas de la pantalla (para mostrar los topos y los resultados) y el ATmega328P utiliza PORTB y PORTD para controlar las columnas y las animaciones.
2. **Protocolo de Comunicación:** El protocolo entre ambos microcontroladores es simple y basado en el estado de los pines:
    a. El PIC16F887 envía información sobre el topo (número del topo o imágenes) y la instrucción para mostrar la animación o el resultado al ATmega328P.
    b. El ATmega328P recibe esta información y la utiliza para actualizar la pantalla y las animaciones.
3. **Inicio del Juego:** El ATmega328P empieza con la animación "EMPECEMOS" y luego espera la señal del PIC16F887 para proceder con el juego (cuando el botón es presionado).
4. **Sincronización del Juego:** El PIC envía las actualizaciones de puntuación, tiempo y estado del juego (como acierto, derrota y nivel) al ATmega328P, que las refleja visualmente.

Enlace a Github

https://github.com/danibasilio887/Tarea2-Grupo5.git