Universidad de San Carlos de Guatemala
Primer semestre
Facultad de ingeniería
Escuela de Ciencias y Sistemas
Laboratorio Arquitectura de Computadores y Ensambladores 1

MANUAL TECNICO

Roberto Carlos Gómez Donis 202000544

Edgardo Andrés Nil Guzmán 20180119

César André Ramírez Dávila 202010816

José Manuel Ibarra Pirir 202001800

Angel Francisco Sique Santos 202012039

pintarAvion(): Esta función se encarga de dibujar el avión en el buffer de la matriz LED. Dependiendo del valor de la variable CAMBIAR_DIRECCION, se establecen los píxeles correspondientes para representar el avión en una orientación específica.

borrarAvion(): Esta función se utiliza para borrar el avión del buffer de la matriz LED. Al igual que en la función anterior, la orientación del avión se determina mediante la variable CAMBIAR_DIRECCION.

generarObjetivos(): Esta función se encarga de generar objetivos de manera aleatoria en el juego. El número de objetivos generados depende del nivel del jugador. Los objetivos se colocan en el buffer de la matriz LED según su posición, altura y desplazamiento aleatorio.

respaldoTorres(): Esta función realiza una copia de las torres presentes en el buffer principal (buffer) al buffer de respaldo (copiatorres). Esto se hace para guardar una copia de las torres antes de realizar ciertas operaciones en el juego.

vaciarRespaldo(): Esta función vacía el buffer de respaldo (copiatorres), estableciendo todos sus valores en cero. Se utiliza para limpiar el buffer de respaldo antes de realizar nuevas operaciones.

ocultarNivel(): Esta función oculta todos los píxeles en el buffer principal (buffer), estableciendo todos sus valores en cero. Se utiliza para borrar cualquier representación visual en la matriz LED antes de iniciar un nuevo nivel.

pintarLED(int x, int y): Esta función se utiliza para controlar los LEDs en la matriz LED directamente sin usar el controlador LED. Toma las coordenadas x e y como parámetros y enciende o apaga el LED correspondiente en la matriz LED.

mostrarMatriz(): Esta función muestra el contenido del buffer principal (buffer) en la matriz LED. Utiliza el controlador LED (matriz) para establecer los valores de los LEDs en función de los valores en el buffer.

reiniciarJuego(): Esta función se utiliza para reiniciar el juego. Establece todas las variables relacionadas con el juego en sus valores iniciales, reinicia las puntuaciones y los contadores, y limpia los buffers.

sinVidas(): Esta función se llama cuando el jugador se queda sin vidas en el juego. Almacena el puntaje actual en la matriz de datos y reinicia el juego.

impactoAvionTorre(): Esta función verifica si hay colisión entre el avión y las torres en el juego. Si se detecta una colisión, desplaza el avión hacia arriba y disminuye la cantidad de vidas del jugador. Si el jugador se queda sin vidas, llama a la función sinVidas().

verificarNivel(): Esta función verifica si se debe avanzar al siguiente nivel en el juego. Comprueba si no quedan torres en el buffer principal o si la altura máxima de las torres supera un cierto valor. Si se cumple alguna de estas condiciones, incrementa el nivel y realiza las acciones correspondientes.

verificarNivel(): Es responsable de verificar si se debe avanzar al siguiente nivel en el juego.

mostrarVidasRestantes(): Es una función que se encarga de mostrar la cantidad de vidas restantes en una matriz de LEDs de tamaño 8x16 cuando el juego se encuentra en pausa.

movimiento_0(): Esta función se encarga de cambiar la dirección del movimiento del texto en un bucle. Cuando se detecta una pulsación en el botón "BTN_D" y el estado anterior del botón era "HIGH", se cambia el valor de la variable "controller_init_matrix" entre verdadero y falso. Además, se imprime un mensaje por el puerto serial indicando el cambio de dirección del movimiento.

pintarBarra(int porcentaje, int posiX): Esta función se utiliza para dibujar una barra de progreso en la matriz de LEDs de tamaño 8x16. Recibe como parámetros el porcentaje de llenado de la barra y la posición inicial en el eje X donde se dibujará la barra. Dependiendo del valor del porcentaje, se establecen los elementos de la matriz de LEDs correspondientes para representar la barra de manera gráfica.

mostrarEstadisticas(): Esta función se encarga de mostrar estadísticas en forma de barras de progreso en la matriz de LEDs de tamaño 8x16. Calcula la suma total de los datos almacenados en un arreglo y luego calcula el porcentaje de cada valor en relación con la suma total. A partir de estos porcentajes, invoca la función pintarBarra() para dibujar las barras de progreso correspondientes en diferentes posiciones de la matriz de LEDs.

impactoAvionTorre(): Esta función se encarga de verificar si hay una colisión entre el avión y las torres en el juego. Si se detecta una colisión, desplaza el avión hacia arriba y disminuye la cantidad de vidas del jugador. Si el jugador se queda sin vidas, llama a la función sinVidas().

verificarNivel(): Esta función se encarga de verificar si se debe avanzar al siguiente nivel en el juego. Comprueba si no quedan torres en el buffer principal o si la altura máxima de las torres supera un cierto valor. Si se cumple alguna de estas condiciones, incrementa el nivel y realiza las acciones correspondientes.

mostrarVidasRestantes(): Esta función se encarga de mostrar la cantidad de vidas restantes en una matriz de LEDs de tamaño 8x16 cuando el juego se encuentra en pausa.

pintarMenuPrincipal(): Esta función se utiliza para dibujar el menú principal en la matriz de LEDs de tamaño 8x16. Se establecen los elementos de la matriz de LEDs correspondientes para representar el avión, las estadísticas y la configuración del juego. Además, se verifica la pulsación de los botones para cambiar al modo de juego, al modo de estadísticas o al modo de configuración.

menu_mode(): Esta función se encarga de manejar el modo de menú. Si la partida no ha iniciado, se llama a la función pintarMenuPrincipal() y mostrarMatriz() para mostrar el menú principal en la matriz de LEDs.

change_init_0(): Esta función verifica si el botón "BTN_K" ha sido presionado durante un cierto tiempo y cambia el estado de la variable "controller_init_matrix" a falso. Si se detecta el cambio, se cambia al modo de juego

initial_mode(): Esta función maneja la lógica de los botones en el modo inicial. Llama a la función change_init_0() y movimiento_0() para verificar y actualizar el estado de los botones.

move_left_1(): Esta función se encarga del movimiento hacia la izquierda en el modo de juego.

move_right_1(): Esta función se encarga del movimiento hacia la derecha en el modo de juego.

game_mode(): Esta función maneja el modo de juego. Si la partida no ha iniciado, se llama a la función pintarMenuPrincipal() y mostrarMatriz() para mostrar el menú principal en la matriz de LEDs. Si la partida ha iniciado, se realiza la lógica del juego, incluyendo el movimiento del avión, la generación de objetivos, la detección de colisiones, el manejo de los botones y la pausa del juego.

estadisticas_mode(): Esta función maneja el modo de estadísticas. Actualmente no se encuentra implementada.

configuration_mode(): Esta función maneja el modo de configuración. Muestra barras de progreso para la velocidad y las vidas iniciales del juego en la matriz de LEDs. Lee los valores de los potenciómetros para ajustar la longitud de las barras y actualiza el estado de las variables correspondientes. Permite regresar al modo inicial al presionar el botón "BTN_K".

initiate_buttons(): Esta función se encarga de configurar los pines de los botones como entradas con resistencia de pull-up.