Titulo:	"Yast Denz"

Ciclo Lectivo **2021**	Curso N°	R2004	Grupo N°	1

Integrantes

Apellido Nombres	Legajo	Calificación individual	Fecha
Albero, Federico	1756989		
Catá, Juan Sebastián	1756977		
Lugano, Damián Gabriel	1756990		

Calificación grupal:		Fecha:

Profesor:	Mariana Prieto Canalejo
Auxiliar/es Docente:	Jorge Escolá

Observaciones primera entrega	
Observaciones segunda entrega	

Índice

1 Desarrollo de la idea fuerza

El proyecto “Yast Denz” intenta replicar a los juegos de Play Station “Just Dance” o “Dance Dance Revolution”, en los cuales el usuario deberá intentar replicar una secuencia de pasos de baile sobre el dance pad (alfombra de baile). La implementación del mismo consiste en un sistema inalámbrico de dos estaciones, en donde por un lado se tendrá una placa que interconecta al LPC845 junto al ESP 8266 y a los demás periféricos. La segunda estación consta de una computadora en donde se ejecutará la interfaz gráfica del programa y se conectará con la estación anterior mediante Wi-Fi.

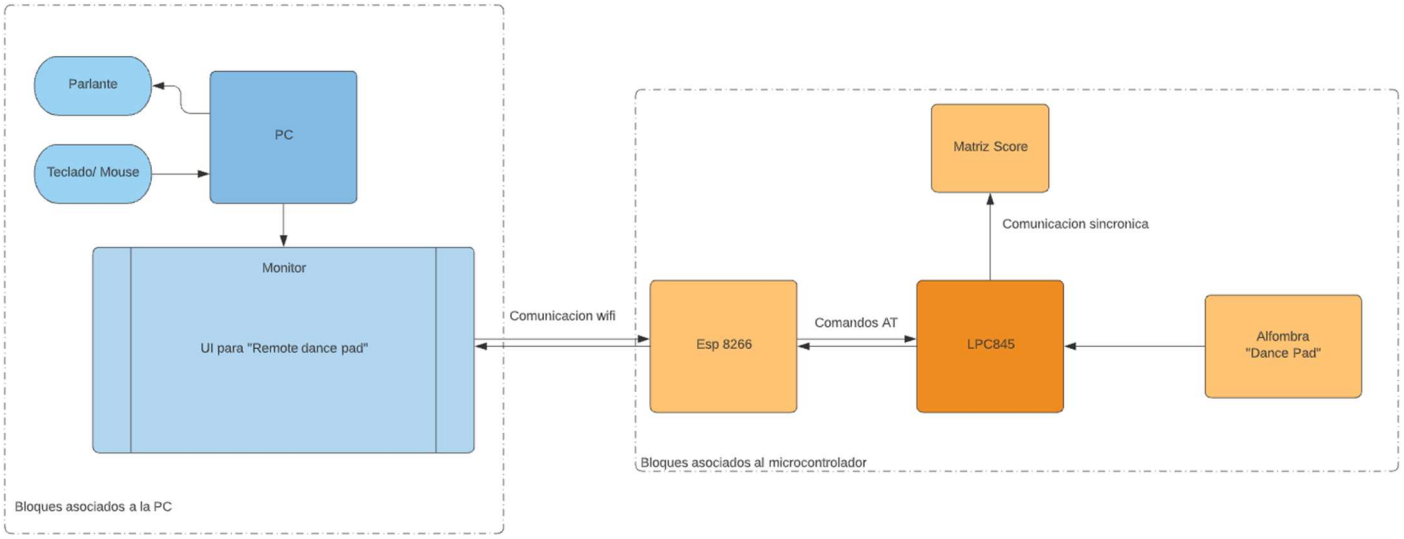
2 Introducción

2.1 Objetivos

- Poder aplicar los conocimientos aprendidos a lo largo de la cursada de la materia en las distintas etapas del proyecto: el uso de clases en C++ para la interfaz gráfica, la bifurcación de procesos mediante threads, la programación del microcontrolador a través de su IDE implementando una máquina de estados, entre otros.
- Utilizar técnicas de software aplicables a elementos de hardware para optimizar su funcionamiento, como el antirrebote para un sistema de múltiples teclas.
- Desarrollar drivers y primitivas para hardware desconocido mediante el uso de sus hojas de datos, como en el caso de la matriz de puntos led.

2.2 Diagrama en bloques

Ilustración 1, diagrama en bloques



3 Descripción detallada

3.1 Bloques

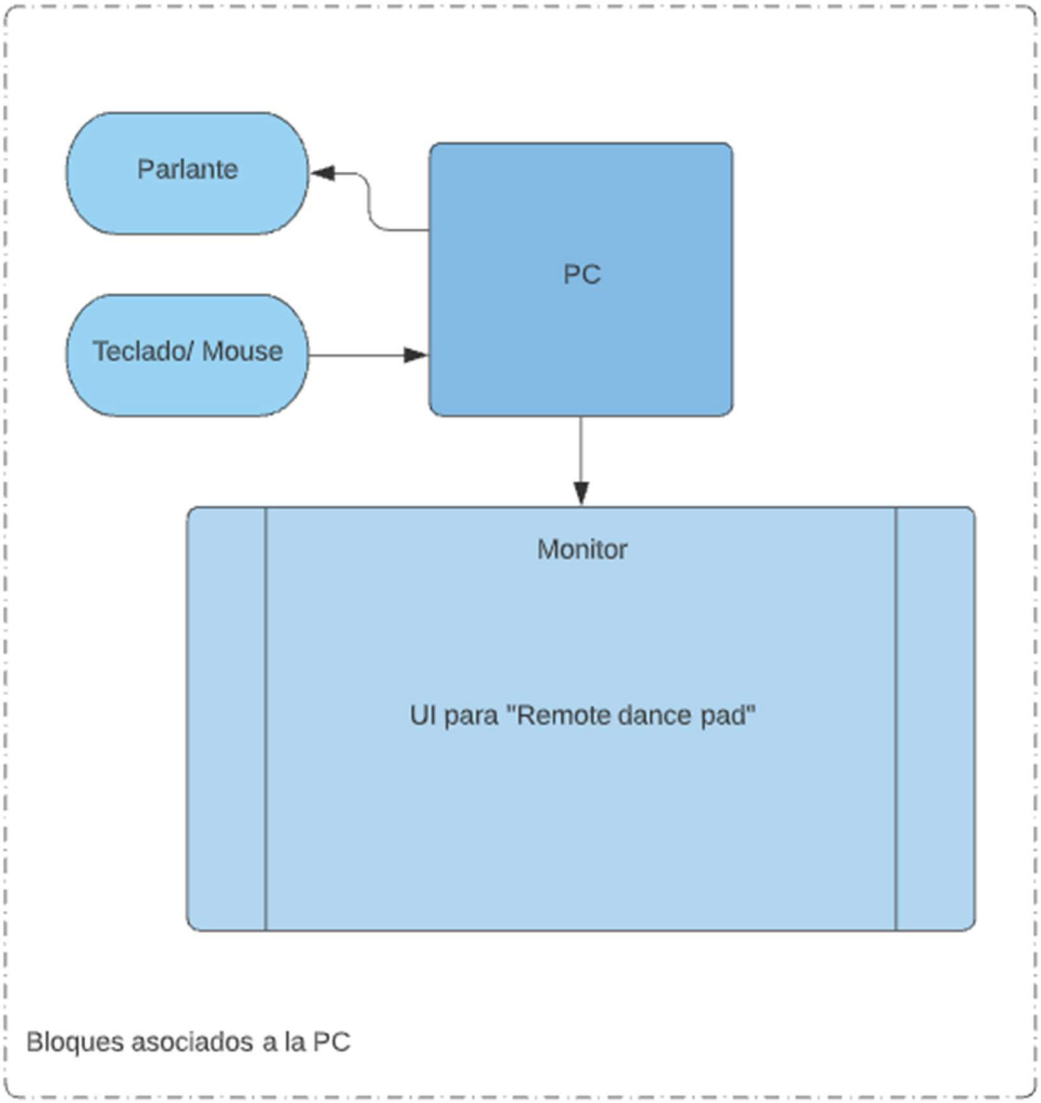


Ilustración 2, bloques asociados a la PC

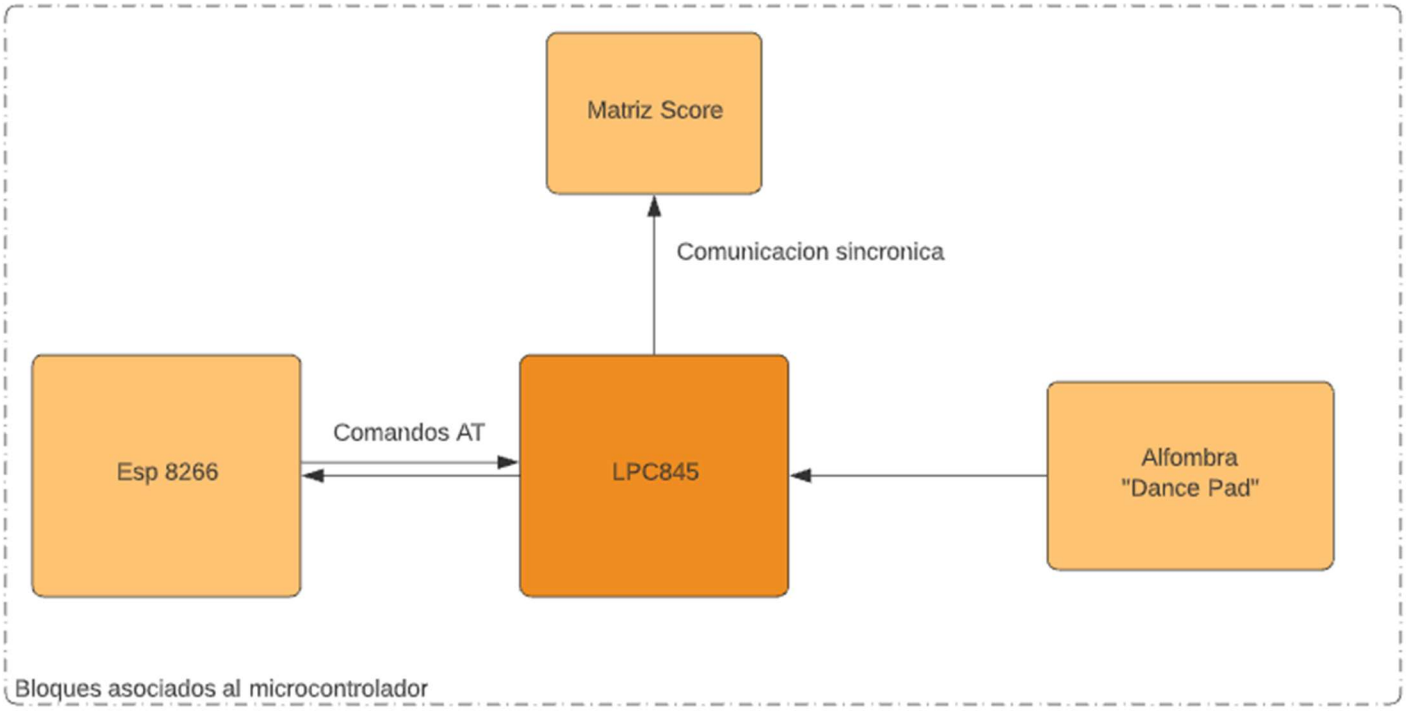


Ilustración 3, bloques asociados al microcontrolador

3.2 Especificaciones

Bloques asociados a la PC:
En la PC se encontrará la interfaz gráfica con el menú de opciones para elegir la dificultad entre otras opciones. Una vez se empieza el juego desde la computadora, la idea es que esta se encargue también de mostrar la secuencia de botones a presionar y de reproducir la canción por los parlantes. La interfaz se va a poder controlar con las teclas de la alfombra, con el teclado de la PC o con una serie de switches que se encuentran en el PCB

Bloques asociados al microcontrolador:
El LPC845 va a ser el encargado de interpretar la comunicación de la alfombra y de comprobar que la tecla presionada se corresponda con la secuencia pautada. Además, se mostrarán los puntos que acumule el usuario durante el transcurso del juego en una matriz de LED que se irá actualizando constantemente.

La alfombra dance pad será la principal plataforma de interacción con el usuario, en la cual se basa el juego. En ella se podrán presionar distintas teclas con los pies, con la intención de justamente generar el baile. Cada tecla que se ingresa se envía directamente al micro, donde el dato es procesado.

Por último, el ESP8266 va a ser el encargado de comunicar la PC con el microcontrolador vía wifi. En específico, se va a encargar de enviar tramas con la información de que teclas se presionaron y en qué momento.

4 Descripción del hardware utilizado

4.1 Descripción de los elementos de hardware en conjunto a su desarrollo de software para su funcionamiento

- Módulo de 4 matrices de puntos de led 8x8:

  Se describirá el funcionamiento de este periférico a partir de la siguiente imagen extraída desde la datasheet del mismo *(figura n°1)*. Este hardware cuenta con 2 pines de alimentación (Vcc y Gnd) y otros 3 (Din, Load, Clk), para los cuales se desarrollaron drivers para poder settear el estado (alto o bajo) de sus salidas de manera más conveniente.

  El CI que utiliza este hardware es el Max7219, el cual cuenta con un registro de desplazamiento de 16 bits y una memoria SRAM de 8x8 bits. La información entonces se empaqueta en tramas de 16 bits, en donde los primeros 8 corresponden a la dirección del registro al cual se quiere acceder dentro de la memoria del integrado y los 8 restantes a la data. Existen entonces 8 registros los cuales se encargan de controlar cada uno una fila de la matriz. El estado de estos registros se refleja directamente en los estados de los leds de cada fila de la matriz.

  Como se puede ver en el diagrama temporal, la secuencia de carga de un dato es la siguiente:

  Para comenzar la carga de datos, primero hay que colocar en estado bajo el pin LOAD. Seguido se comienza a enviar bit por bit desde el D15 (MSB) hasta el D0 (LSB). Cada bit se envía desde el pin Din cuando hay un flanco ascendente en Clk. Para esto, se hace uso de las funciones **sendAddress()** y **sendData(),** las cuales se encargan de hacer la secuencia de envío de sus 8 bits correspondientes. Una vez enviada la trama de 16 bits se coloca en estado alto el pin Load, cargando así en memoria la información enviada.

  Como se tienen 4 displays en cascada, hay que tener en cuenta que lo ingresado por el pin Din de la matriz saldrá por el pin Dout 16 pulsos de Clk después, por lo que se aprovecha esta estrategia para poder comunicarse con la fila n-esima de cada uno.

  Para esto se utilizará un vector global de 32 posiciones, haciendo referencia a las 32 filas que hay entre los 4 displays. En cada posición del vector se carga un valor en hexadecimal correspondiente a los leds que se tienen que encender de tal manera de formar un dígito visible entre las 8 filas de un mismo display. De esto último se encargan las funciones primitivas **separarCifras()** y **print().** Existe otra función, **show()**, la cual se ejecuta en cada interrupción del systick y se encarga de refrescar de a una fila de las 4 matrices, por lo que al cabo de 32 interrupciones (equivalente a 32 milisegundos aproximadamente) se actualiza el valor que imprimen las 4 matrices sin visualizar ningún tipo de efecto fantasma.

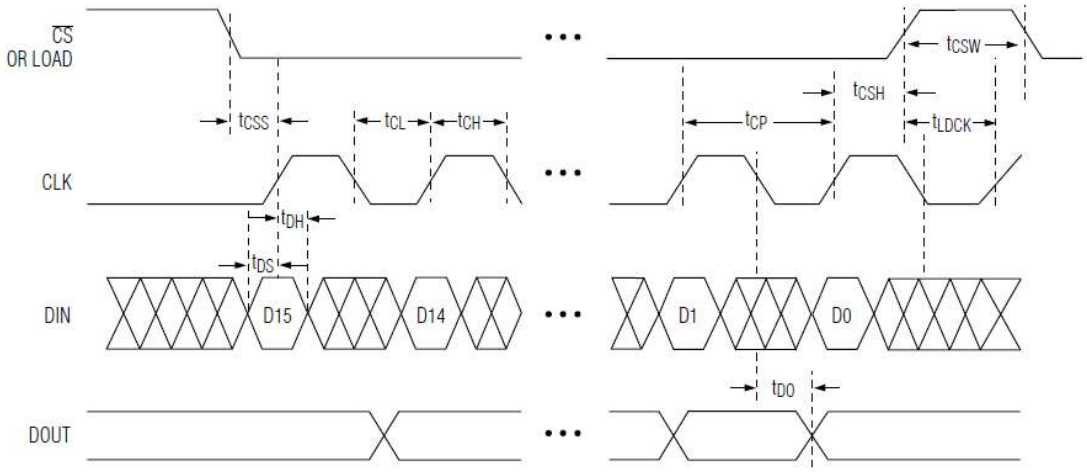


Ilustración 4, diagrama temporal de la secuencia de carga de los 16 bits que conforman la trama.

- Displays de 7 segmentos cátodo común:

  Este hardware ya fue utilizado durante la cursada, por lo que su funcionamiento ya es familiar.

  Lo principal a destacar es que se utilizó una estrategia de software para poder controlar los segmentos de los displays directamente con los pines de GPIO del microcontrolador, sin la necesidad de incorporar circuitería adicional al circuito, a excepción de unos resistores de 100 Ω para limitar su corriente.

  La conexión del circuito se puede visualizar en la siguiente sección, ilustración n°6, en donde todos los segmentos de los displays están conectados en paralelo a las salidas gpio (a través de los resistores de 100Ω) del LPC845 y los comunes de cada display están conectados por separado, permitiendo así poder activar cada uno de manera individual. La idea es ir encendiendo los distintos dígitos de a uno con su valor correspondiente a alta frecuencia de tal manera de que se genere un efecto óptico en nuestros ojos y parezca que estos están todos encendidos al mismo tiempo.

  Para ello, se desarrollarán los siguientes drivers y primitivas: La primitiva **Display()** se encarga se recibir el valor que se desea imprimir en los displays y separar sus cifras, almacenándolas en un vector global. El driver **BarridoDisplay()** se ejecuta de manera sincrónica en la interrupción del systick y se encarga de encender los segmentos correspondientes al dígito que se quiere imprimir en el display mediante una tabla de conversión. Luego, se enciende solamente el dígito al que corresponde dicho valor, habiendo tomado previamente la precaución de apagar todos los dígitos al inicio de la función para evitar el “efecto fantasma”. Al tratarse de displays de cátodo común, estos se activan colocando un 0 lógico en el pin que controla su común, provocando que circule la corriente a través de los diodos internos y se iluminen los segmentos. Caso contrario, si se coloca un 1 en el común queda desactivado dicho display.

- Alfombra de baile:

Para implementar este hardware se adquirió un dance pad como los usados para jugar en PS2 *(ilustración n°5)* y se lo ha modificado. Dentro del mismo se encontraba un PCB que, en su funcionamiento normal, interpretaba los pulsos de los botones mecánicos y se comunicaba con la consola de videojuegos mediante el protocolo de comunicación SPI.

Características y funcionamiento

Los bornes de los botones en el PCB (*ilustración n°6)* tienen el siguiente layout:

Pin	Botón	Pin	Botón
1	Select	7	↑
2	Cruz	8	Cuadrado
3	←	9	→
4	Triangulo	10	Circulo
5	↓	11	Start
6	Común(GND)	-	-

Ilustración 5, dance pad / alfombra de baile

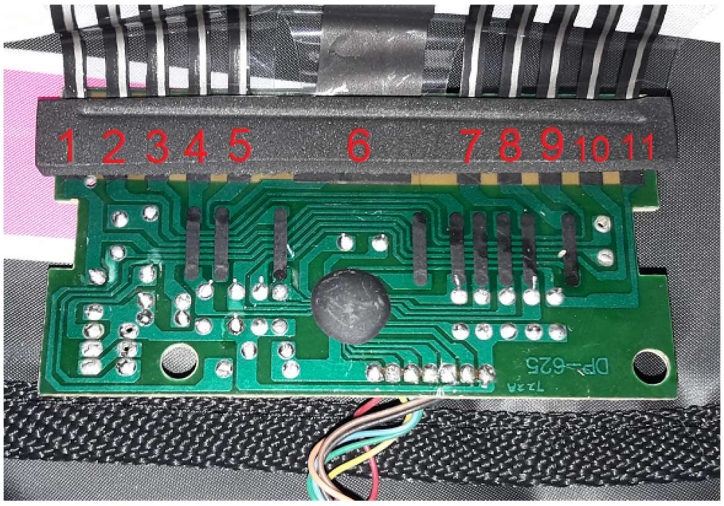


Ilustración 6, layout de la placa interna de la alfombra.

El dance pad consta de una alfombra con varias capaz de diferentes materiales:

- La primera y la quinta capa es de un material similar a la espuma.
- La segunda capa es el común, en este caso GND. Consta de una lámina conductiva que une uno de los bornes de todos los botones.
- La tercera capa es también de un material similar a la espuma, pero con pequeños orificios que permite la conexión entre bornes de la capa 2 y 4 solo cuando se hace presión.
- En la cuarta capa están todos los bornes de los botones por separado.

Cuando se pisa uno de los botones, la capa 2 y 4 hacen contacto, permitiendo la circulación de corriente eléctrica.

Implementación

Debido a que este circuito interconectaba varios botones entre sí, lo que se hizo fue inhabilitar el resto de la placa y solo dejar disponible los bornes de los botones. Posteriormente se les soldaron cables para usarlos como si fueran switches normales *(ilustración n°7).*

En el LPC845 se configuraron 6 entradas con pull up interno. Por defecto las entradas van a estar en 1 y cuando la entrada se pone en 0, quiere decir que uno de los botones fue presionado.

El driver **DriverTecladoSW()** que se ejecuta en la interrupción del systick, se encarga de chequear el estado de los botones cada 1ms teniendo en cuenta los posibles rebotes mecánicos de los switches de la alfombra. Finalmente, desde la aplicación se usa la primitiva **getKey()** que chequea, por nivel, que botón fue presionado.

Ilustración 7, placa interna de la alfombra con las extensiones de cables soldadas y anulando su funcionamiento original.

- ESP8266 Nodemcu:

El Nodemcu es un sistema embebido Wi-Fi generalmente utilizado para IOT. En este caso se utiliza para comunicar bidireccionalmente la interfaz gráfica con el LPC845. Para esto se le cargó el firmware que permite manejarlo con comandos AT mediante su puerto serie.

Algunos comandos AT utilizados son los siguientes:

Comando	Descripción
AT+CWMODE=mode	Establece el módulo en el modo dado 1 = Modo estación (cliente) 2 = Modo AP (huésped) 3 = Modo AP + Estación (modo dual)
AT+CWJAP=ssid,pwd	El módulo se conecta a la red con el nombre ssid indicado y la contraseña pwd suministrada
AT+CIPMUX=mode	mode: 0 = Conexión unica 1 = Múltiples conexiones, hasta 4
AT+CIPSTART=id,type, adr, port	Empieza una conexión como cliente en IP y puerto específicos
AT+CIPSEND=id,length	Establece la longitud de datos a enviarse en la conexión número 1

Una vez que el ESP8266 está conectado a la misma red Wi-fi que la PC de la interfaz gráfica, el LPC845 es el encargado de enviarle los comandos AT para configurarlo como un cliente TCP y conectarlo a un socket de la PC donde en simultáneo se está corriendo la interfaz gráfica. Para esto, se implementó una máquina de estados llamada **ESP8266_conf()** que envía los siguientes comandos de configuración por puerto serie:

- AT+CIPMUX=1
- AT+CIPSTART=1,"TCP","192.168.0.72",4000
- AT+CIPSEND=1,5
- READY

Estos comandos son enviados cada 1 segundo para su correcto procesamiento. Una vez que se enviaron todos estos comandos, queda establecida una conexión Wi-Fi entre el LPC845 y la PC como un canal para comunicarse.

4.2 Circuitos

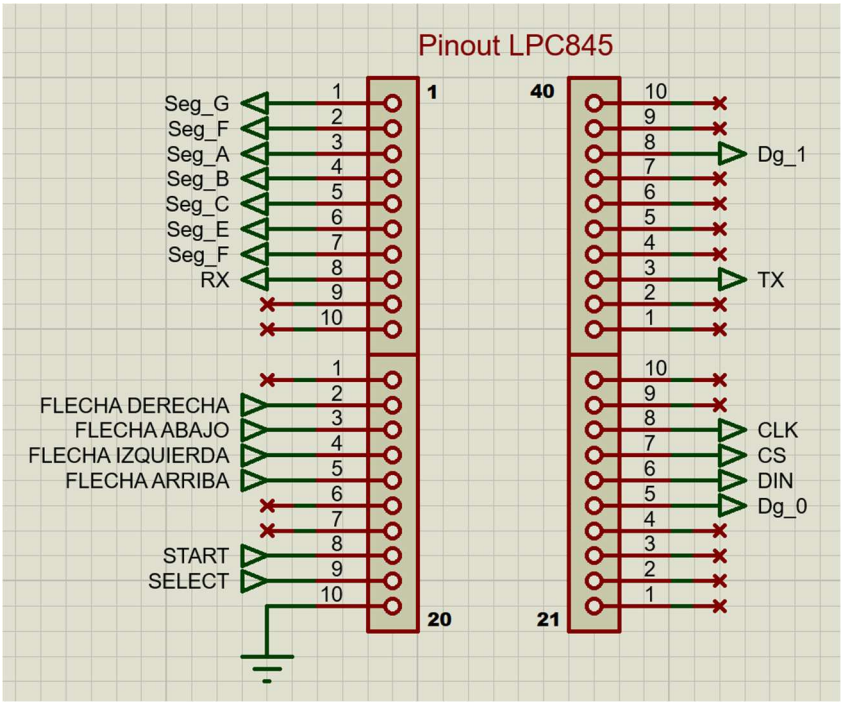


Ilustración 8, diagrama esquemático con las conexiones del LPC845. Se indica la distribución de los pines con los números en negrita

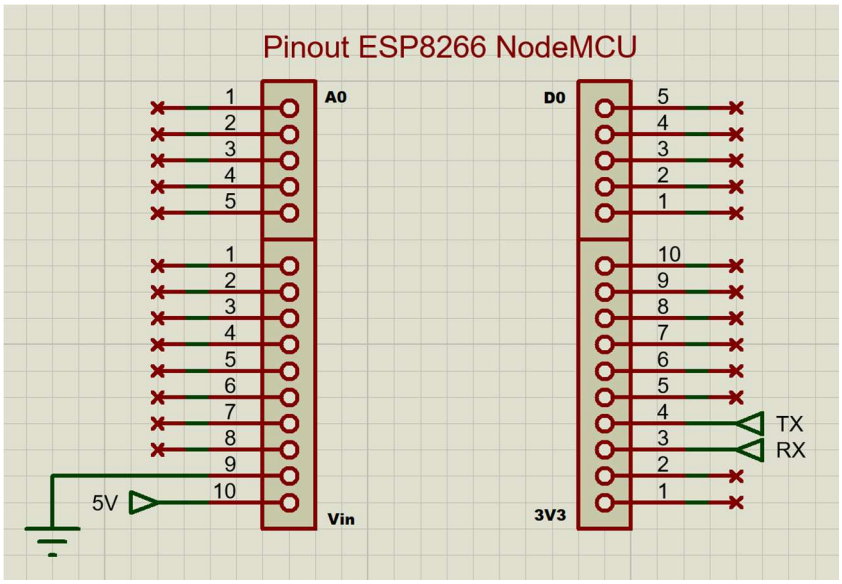


Ilustración 9, diagrama esquemático con las conexiones del ESP8266 NodeMCU. Se indica la distribución de los pines con las referencias marcadas en negrita

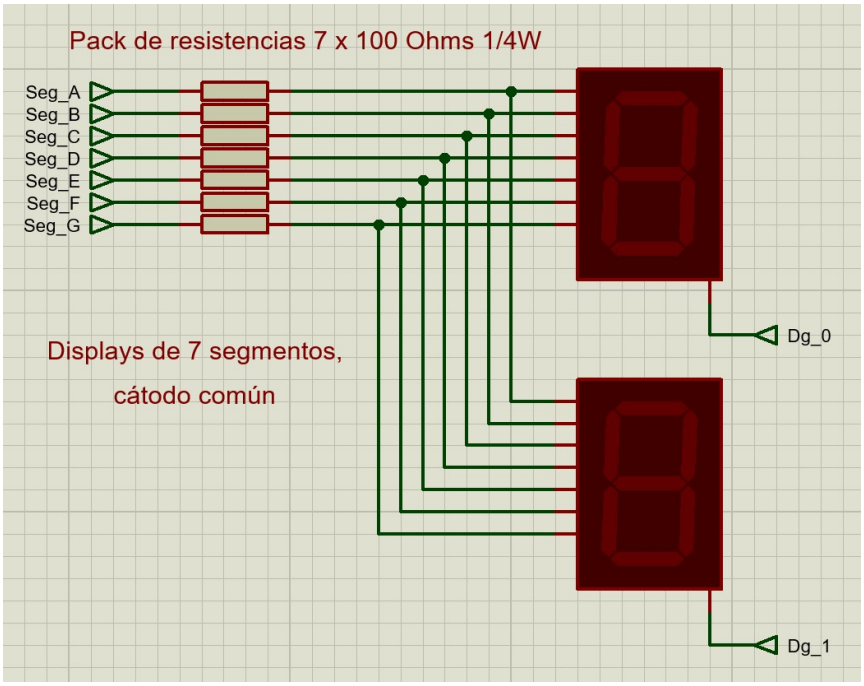


Ilustración 10, diagrama esquemático de conexionado de los displays de 7 segmentos junto a las resistencias limitadoras de corriente.

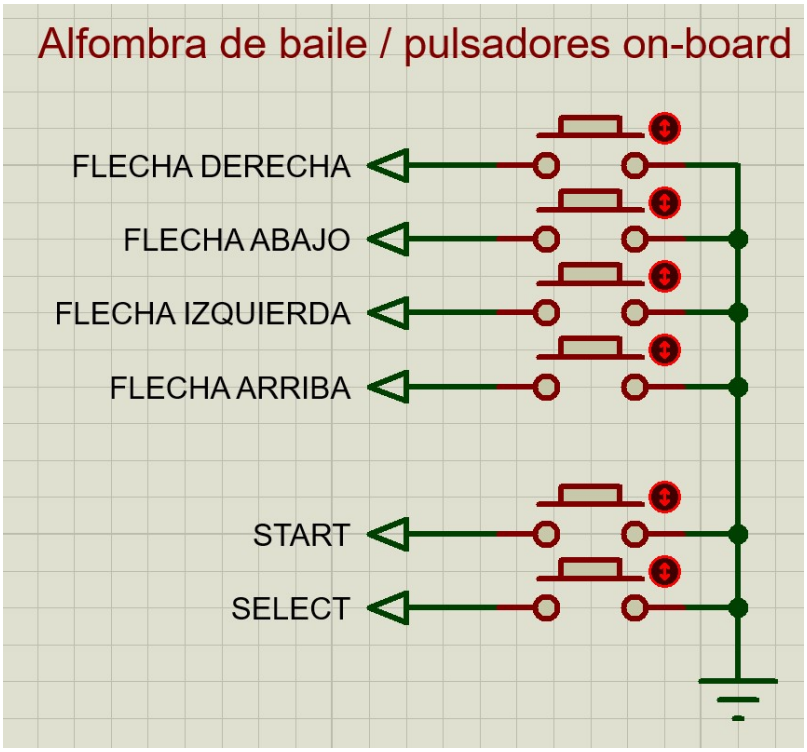


Ilustración 11, diagrama esquemático de conexionado de los botones individuales que componen la alfombra de baile. Solamente se utilizan las 4 flechas principales y los botones Start y Select, no los que se encuentran en las diagonales.

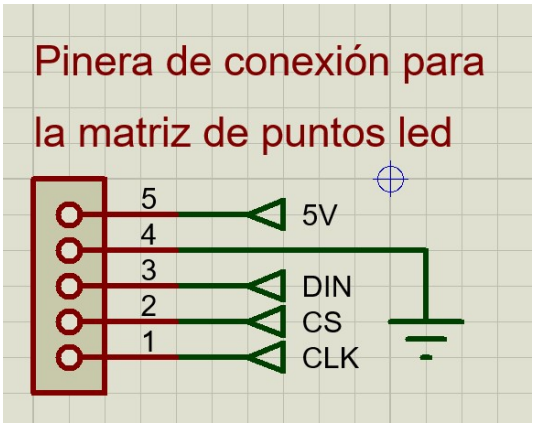


Ilustración 12, pinera de salida para la conexión de la matriz de puntos de led

Ilustración 13, referencias utilizadas en los circuitos esquemáticos

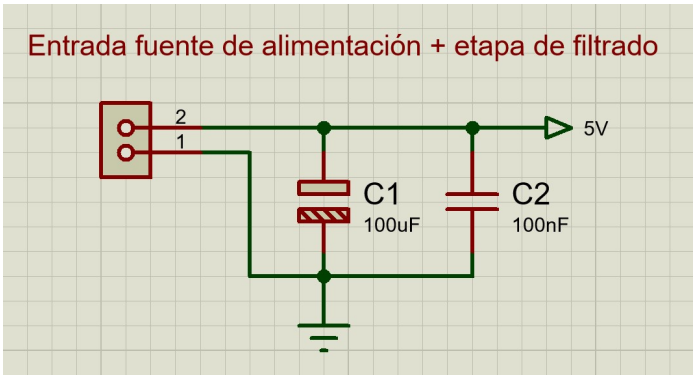


Ilustración 14, bornera de entrada de alimentación para el circuito. Se agregan dos capacitores en paralelo a la alimentación a modo de filtrado. Se unen todas las referencias a masa (Gnd) en el circuito.

4.3 Resumen mappeo pines LPC845

Puerto - Pin	Funcionalidad	Puerto - Pin	Funcionalidad
P0.0	-	P0.18	Segmento A
P0.1	Digito 1	P0.19	Segmento B
P0.2	-	P0.20	Segmento C
P0.3	-	P0.21	Segmento D
P0.4	-	P0.22	Segmento E
P0.5	-	P0.23	Rx
P0.6	Tx	P0.24	-
P0.7	-	P0.25	-
P0.8	Start	P0.26	Flecha Derecha
P0.9	Select	P0.27	Flecha abajo
P0.10	-	P0.28	Flecha Izquierda
P0.1	-	P0.29	Flecha arriba
P0.12	Digito 0	P0.30	-
P0.13	Din	P0.31	-
P0.14	Cs/Load	P1.0	Led verde
P0.15	Clk	P1.1	Led azul
P0.16	Segmento G	P1.2	Led rojo
P0.17	Segmento F	-	-

4.4 Links a hojas de datos

- Max7219 (CI controlador de cada matriz de puntos led de 8x8):

https://datasheets.maximintegrated.com/en/ds/MAX7219-MAX7221.pdf

- Esp8266 NodeMcu:

https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

https://www.espressif.com/sites/default/files/4a-esp8266_at_instruction_set_en_v1.5.4_0.pdf

- LPC845, manual de usuario y documentación del microcontrolador:

https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/general-purpose-mcus/lpc800-cortex-m0-plus-/lpc845-breakout-board-for-lpc84x-family-mcus:LPC845-BRK

5 Realización de una placa – PCB del circuito

Para poder realizar todas las conexiones que demanda el circuito de una forma segura y estable, minimizando el riesgo de un "falso contacto", se decidió realizar un PCB en el cual montar ambos microcontroladores y los displays de 7 segmentos junto con sus resistencias. Además, se decidió agregar unos pulsadores on–board en paralelo a las teclas de la alfombra, de tal manera que se pueda debuggear el proyecto de manera más sencilla sin necesidad de conectar la alfombra todo el tiempo.

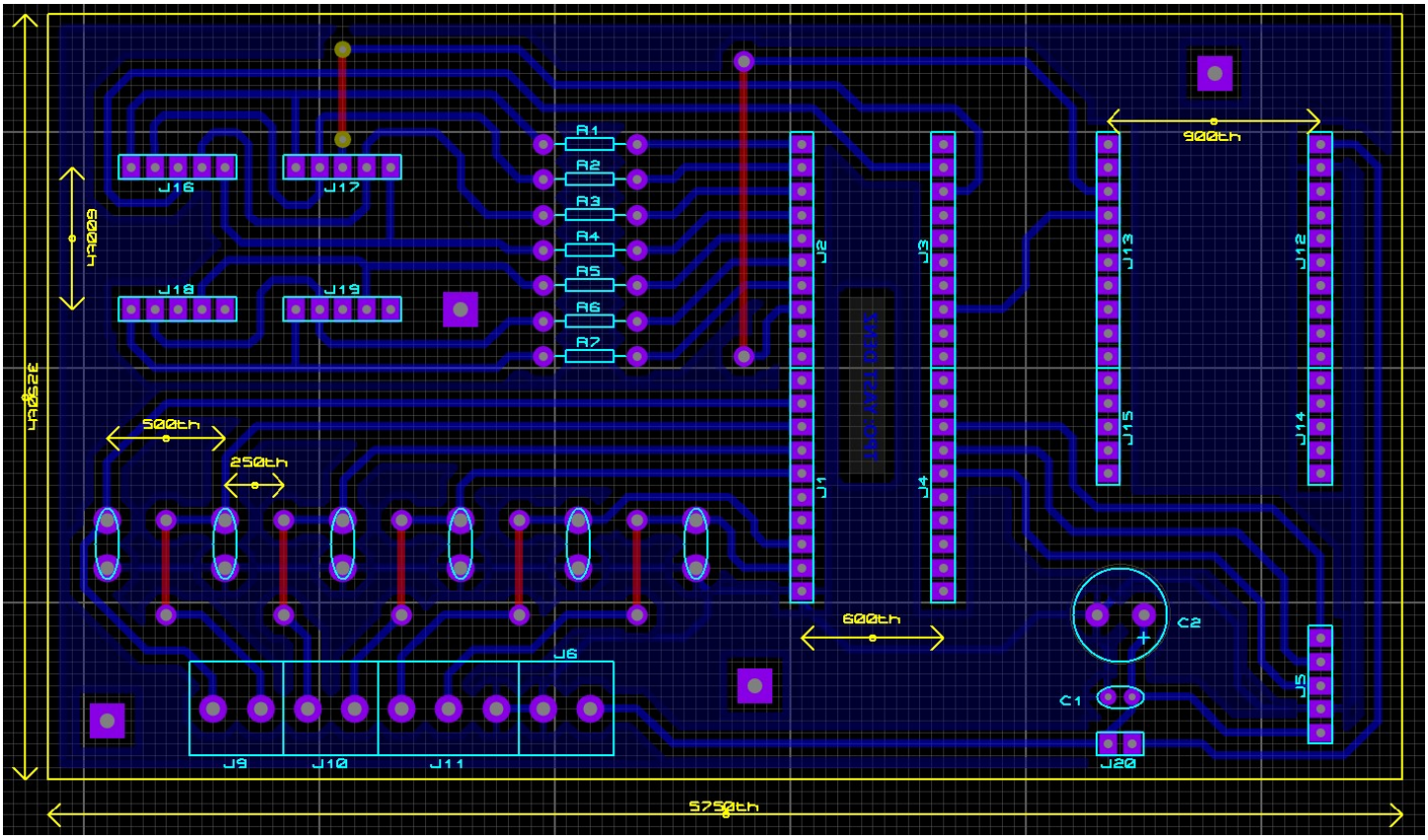


Ilustración 15, diseño del PCB. Se utilizó el programa "Proteus" para ello.

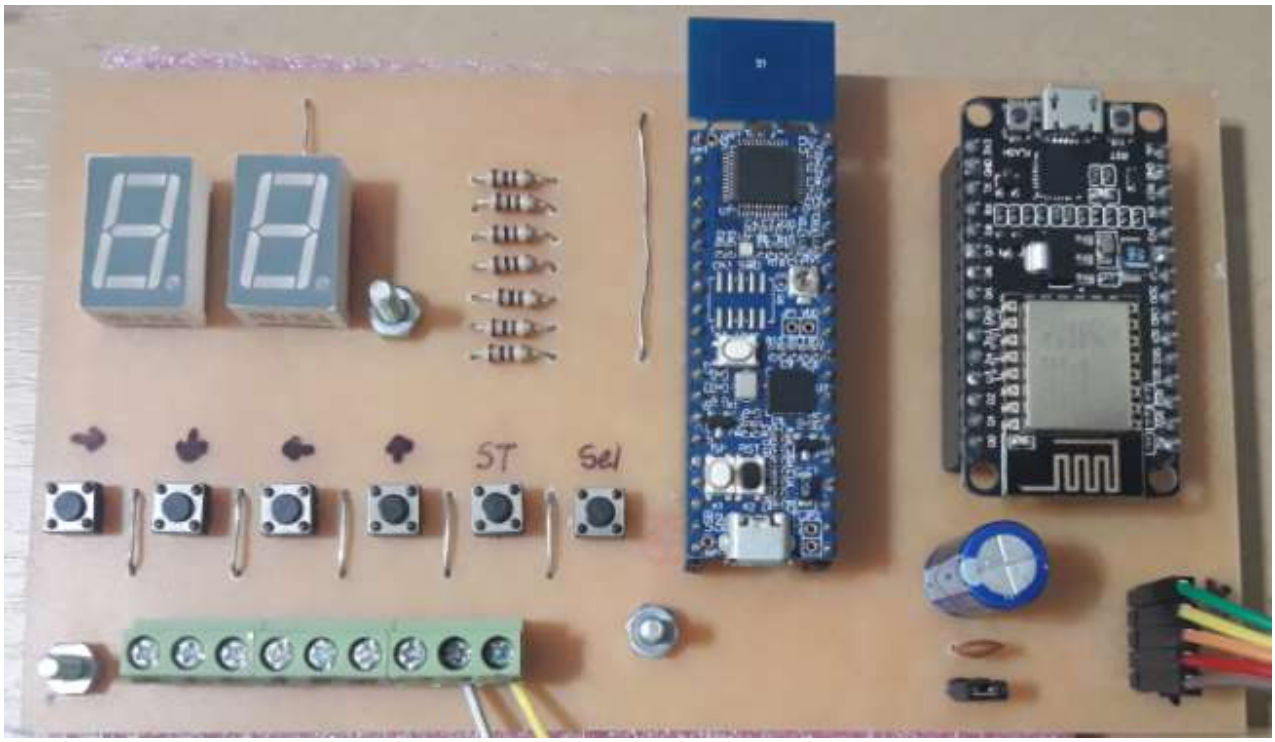


Ilustración 16, PCB terminado y con los microcontroladores incrustados correctamente.

6 Interfaz gráfica

La interfaz de usuario está compuesta por dos instancias: la primera consta de un menú, donde se establecen las configuraciones pertinentes a la siguiente etapa. Esta última, corresponde a la pantalla del juego, donde se muestran las flechas que el jugador debe presionar. Estas se mueven hacia un destino, que se corresponde, apenas toque la flecha con el mismo, al momento exacto en el que se tiene que presionar la tecla. Si el usuario deja pasar la flecha, la misma se mantiene en la posición de destino unos instantes hasta desaparecer. En el caso contrario, cuando el usuario oprime la tecla correspondiente, la flecha se desvanece inmediatamente. Una vez terminado el juego, en la misma pantalla se le comunica al usuario si ganó o perdió, y tiene la opción de jugar de vuelta, o salir del juego al menú principal.

El menú principal es un QT Widgets Applicattion, que consta de un MainWindow con los diversos widgets para configurar el programa:

- Un carrusel para seleccionar la dificultad, hecho con dos QToolButton.
- Un QFileDialog para seleccionar el nivel a jugar.
- Una QComboBox para la resolución y otra para los fotogramas por segundo.
- Un QLineEdit para el puerto a usar en la conexión entre el microcontrolador y la PC.
- Un QPushButton para comenzar el juego.

La pantalla de juego está programada en C++ con la ayuda de la librería Allegro. Elegimos esta librería porque no era posible desarrollar este código con QT, y porque ya sabíamos manejarla de un principio. Para unificarlas, hicimos que el menú haga un llamado al sistema operativo (system()) cuando se presiona el botón para empezar a jugar.

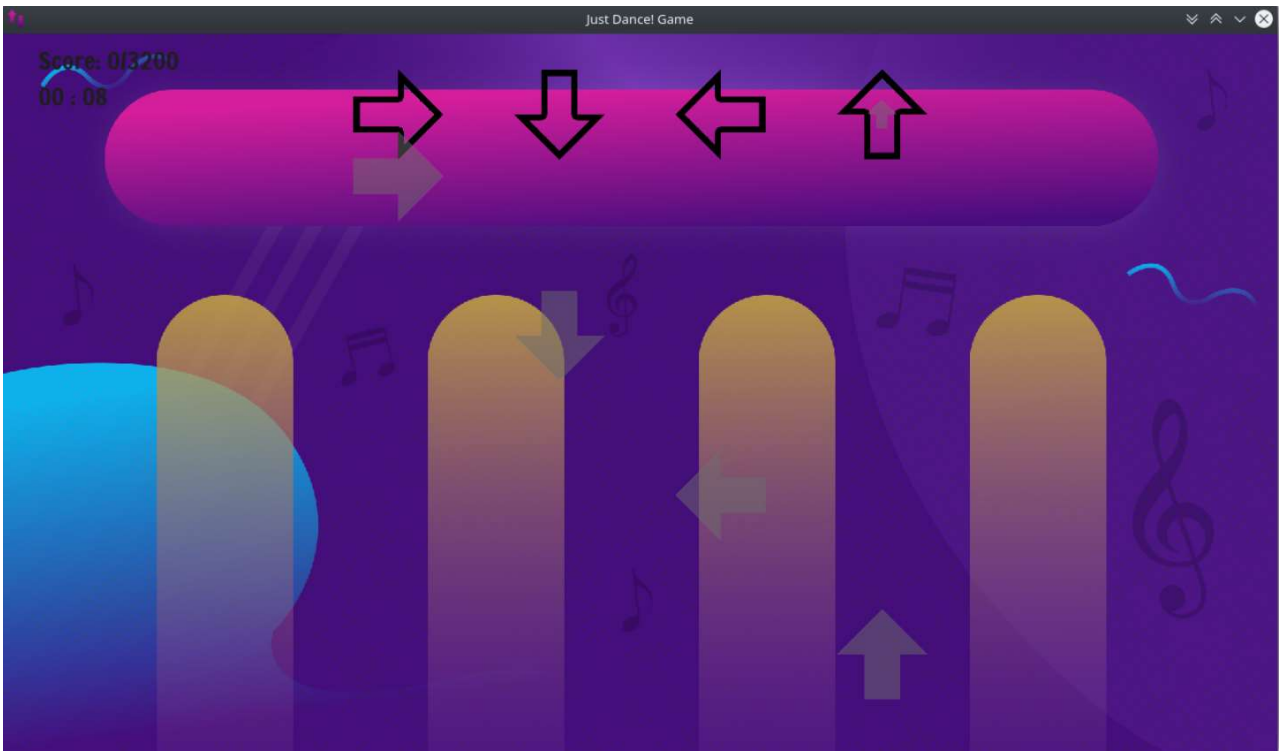


Ilustración 18, pantalla del juego

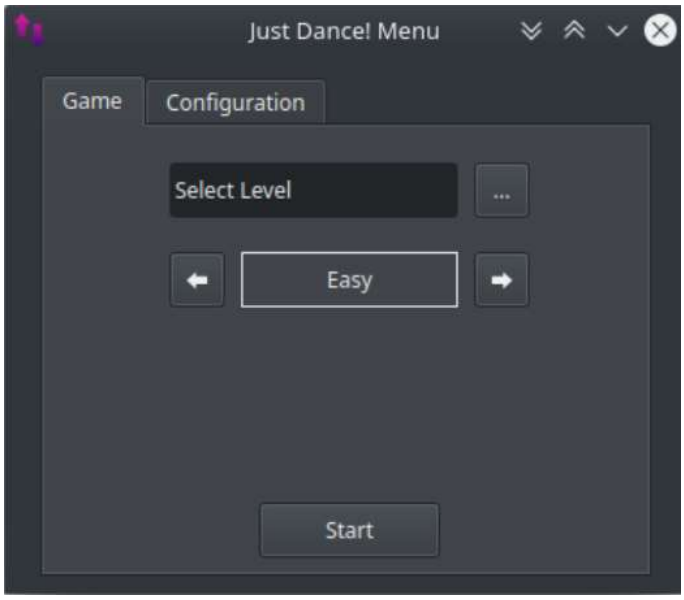


Ilustración 17, menú del juego

7 Máquina de estados de la aplicación

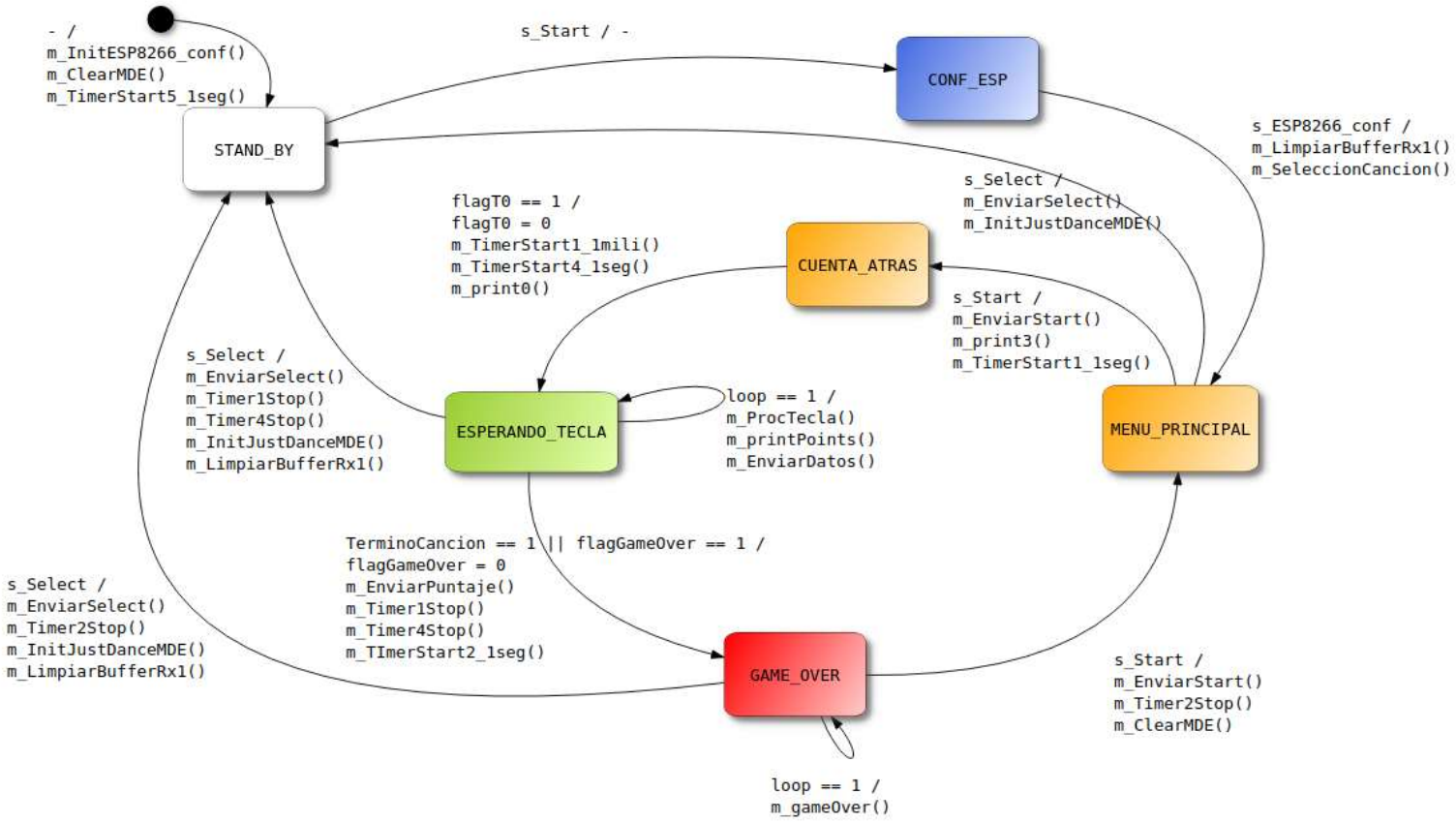


Ilustración 19, máquina de estados principal del programa

8 Problemas encontrados a lo largo del desarrollo del TPO

8.1 Puesta en marcha

- Uno de los primeros problemas con el que nos encontramos ocurrió cuando quisimos juntar los desarrollos en drivers y primitivas de la lectura de la alfombra en forma de teclado y la secuencia de impresión de la matriz de puntos led. Ambos elementos poseen funciones que se invocan de manera periódica dentro del **SysTick_Handler()** y aparentemente la función **show()** demoraba demasiado tiempo dentro de la interrupción, lo que provocaba que no funcionase de manera correcta la detección de las teclas de la alfombra. Dicho problema se solucionó recortando la cantidad de filas de la matriz que se encendían dentro de cada interrupción, por lo que se necesitaron más llamados a dicha función para poder refrescar la matriz completa pero sin llegar al punto de visualizar ningún efecto fantasma o titileo en esta.

- También se presentó un inconveniente en la implementación del contador realizado con displays de 7 segmentos (cátodo común), el cual lleva la cuenta de los segundos transcurridos (de 0 hasta 99) desde que se empieza a reproducir la canción del juego. Lo que sucedía era que, por algún motivo, el pin P0.0 que utilizamos para que controle el común de uno de los dígitos de los displays permanecía todo el tiempo en 0 (estado bajo), por lo que al ser displays de cátodo común esto provocaba que esté encendido constantemente, produciendo un "efecto fantasma" en la impresión de los números. Cabe resaltar que este mismo hardware lo probamos de manera aislada en otro LPC845 con el mismo mappeo de los pines gpio y dicho pin (P0.0) funcionaba correctamente, pero al querer utilizar este hardware junto con el resto de los periféricos que componen el tpo, este pin no funcionaba de la manera esperada y decidimos reemplazarlo por el P0.12. A partir de dicha modificación, se solucionó el inconveniente manifestado anteriormente.

- Uno de los aspectos más desafiantes del proyecto fue hacer funcionar la comunicación serie entre el LPC845 y el ESP8266. El problema principal es que el ESP8266 está constantemente enviando datos por el puerto serie, y muchas veces estos datos no eran útiles por lo que solo llenaban el buffer de recepción y entorpecían la búsqueda del mensaje que realmente se necesitaba recibir. Además, otro problema importante es que no siempre llega la totalidad de la trama al LPC845, a veces llega completa, pero a veces se pierden datos en el medio entre el ESP y el LPC por lo esto afectó bastante al desarrollo y funcionamiento del programa.

- En la interfaz gráfica, inicialmente pensábamos implementar la librería de Allegro en conjunto con QT, pero esto nos fue imposible. QT utiliza funciones bloqueantes para su interfaz gráfica, mientras que la parte de Allegro se debía refrescar cada vez que pasábamos por el código: eran completamente incompatibles. A partir de esto surge la idea del llamado al sistema operativo.

8.2 Falta de información

Hubo dificultades al momento de cargar el firmware que permite controlar al ESP8266 mediante comandos AT ya que anteriormente había sido usado y no tenía el programa cargado de fábrica. El problema no fue que no se encontró información en internet, sino que, al parecer, la mayoría no funcionaban correctamente. Luego de varios intentos, se pudo encontrar el firmware original para poder cargarlo al ESP8266.

9 Beneficios encontrados a lo largo del desarrollo del TPO

9.1 Valoración del TPO expresado por los integrantes del grupo

Damián: Aprendí a controlar bastante bien el puerto serie del microcontrolador y a intentar implementar varias soluciones al mismo problema cuando algo no funciona. Además, me sirvió aprender a usar el ESP8266 ya que es muy

útil para otras aplicaciones. Lo que más valoro es el esfuerzo de mi grupo y de la ayuda de los docentes para lograr un proyecto que fue muy desafiante.

Federico: Personalmente rescato la experiencia de aprender a programar los drivers y primitivas de la matriz de puntos de led a partir de la lectura de su respectiva hoja de datos y de algunos consejos recibidos por parte de los docentes de la materia. Esto último fue fundamental ya que en internet no encontré mucha información útil acerca del tema porque casi todos los desarrollos estaban hechos en la plataforma de Arduino y ya empaquetados en librerías.

Juan Sebastián: En lo personal, el trabajo me ayudó mucho a fortalecer mis conocimientos de C/C++, y además me dió las pautas para desarrollar en conjunto con microcontrolador y PC, no sólo desde el lado de la programación sino también en el trabajo en equipo. Muchos algoritmos pensados para la PC no se pueden aplicar en el microcontrolador, entonces es necesario cambiar su implementación.

9.2 Desde su lugar de alumno que elementos agregaría o quitaría para el desarrollo del TPO

Damián: Estaría bueno poder definir la idea del TPO mas adentrado al tema de programación del LPC ya que es difícil elegir un proyecto sin saber con qué conocimientos vamos a contar. En mi caso, no hubiera elegido hacer este proyecto sabiendo que conocimientos iba a tener en el momento de hacer el TPO, hubiera elegido un proyecto con periféricos como teclado matricial y displays y con menos relevancia de la parte grafica.

Federico: Si bien hay que considerar que el 2021 fue un año particular debido a las restricciones impuestas por la pandemia, creo que hubiera sido de gran utilidad haber tenido algunas clases más presenciales en los laboratorios de electrónica de la facultad. Sobre todo en la recta final del armado del TPO, en donde empiezan a aparecer varios inconvenientes con el armado del mismo y algunos se hacen difícil de solucionar de forma virtual.

Juan Sebastián: Desde mi punto de vista, no se nos debería dar la opción de conectar la PC al resto del proyecto por WiFi, trabajando en conjunto el ESP8266 con el LPC845. Esto lo digo porque, si bien la programación de sockets en la PC y de comandos AT en el microcontrolador es digerible, la comunicación entre el ESP y el LPC es lo que más problemas nos dió. Recibir datos en el LPC desde la computadora fue imposible, y el envío de datos era en ocaciones no fidedigno.

10 Conclusiones

Teniendo en cuenta la vasta cantidad de inconvenientes que fueron surgiendo a lo largo del desarrollo del TPO, se pudo lograr una versión del proyecto tal que cumple con nuestras expectativas y los objetivos planteados en el inicio de este trabajo. Cabe destacar que las herramientas y experiencias adquiridas durante el transcurso de este trabajo serán útiles para futuros proyectos o en un ámbito laboral.

11 Fuentes de información utilizadas

https://liballeg.org/a5docs/trunk/
https://naylampmechatronics.com/blog/21_tutorial-esp8266-parte-i.html