12-5-2021

TOY2JOY

ELCO 2021

Noelia Blázquez García, María Gómez Heredero, Carlos José Llorente Cortijo, Rodrigo Moratilla Alarcón, Imanol Torres Inchaurza

Índice

Tabla de Ilustraciones:

Introducción

Motivación

Cuando dio comienzo la asignatura de ELCO, se plantearon varias posibilidades sobre la temática del proyecto que debía ser desarrollado en el transcurso del semestre. De entre todas, finalmente se escogió un proyecto con una utilidad clara y de gran valor: la educación.

Hoy en día, muchos niños empiezan en los colegios con toda su ilusión y motivación, la cual va decayendo con el paso de los años. Esto se debe a razones de carácter médico (déficit de atención), problemas en el hogar, acoso escolar, problemas de autoestima o problemas de bajo rendimiento. Así, las consecuencias son el aburrimiento en entornos escolares, la desmotivación o la falta de interés de forma sistemática [1].

En este proyecto se pretende generar un producto que sea capaz de educar a los niños de forma que el aprendizaje sea divertido, dinámico y fácilmente adaptable a todos los niveles de rendimiento. De esta manera se aspira acabar con muchos de los problemas anteriormente mencionados. Asimismo, se pretende alcanzar un bien mayor al brindar la capacidad de educar a los más jóvenes en las habilidades más básicas, que van a ser necesarias durante el resto de su vida. En definitiva, el objetivo final es hacer posible una educación en la que los más pequeños no pierdan aquello que les caracteriza: la ilusión.

Por otro lado, está comprobado que la niñez es la etapa en la cual se retienen con mayor eficacia los conocimientos, siendo ésta aún mayor cuando se ve estimulada por los sentidos de la vista y el oído [2]. Con el proyecto debe incorporar varios idiomas de manera que los niños se puedan ir familiarizando con ellos.

Objetivos

El proyecto tiene un claro enfoque hacia los niños de edades hasta unos cinco años. Este producto no está únicamente orientado a su uso en escuelas o colegios. De hecho, se pretende conseguir que el proyecto se extienda tanto en hogares como en guarderías. De esta manera se transformará la perspectiva de un niño hacia el colegio, pasando de ser una simple obligación a un lugar donde divertirte y hacer actividades distintas.

La idea de la apariencia del producto final consiste en una manta interactiva, flexible, resistente e impermeable.

Por otro lado, se pretenden incorporar muchos idiomas para introducir las lenguas extranjeras y fomentar la capacidad de aprendizaje típica del público objetivo. Se partirá en un principio del español, francés e inglés.

En cuanto a la funcionalidad del producto, en primer lugar, debe elegirse el idioma. Después, se podrá elegir el juego deseado, que estará vinculado a ciertas capacidades como pueden ser realizar multiplicaciones e interpretar el resultado correcto, ser consciente de cómo suenan los números o letras y saber asociarlos a su forma de representarse, o algo tan básico como saber asociar los colores con su nombre. De esta manera, será necesario brindar al producto de la capacidad de generar señales acústicas.

Metodología

Para el desarrollo del producto se han seguido varios pasos.

- Análisis de necesidades y planteamiento de objetivos: se pensó en las necesidades básicas que se querían cubrir. Con ello, se decidió que sería necesario añadir al producto final la posibilidad de manejar varios idiomas y diferentes juegos interactivos.
- Análisis y diseño de HW: se analizaron diferentes componentes existentes en el mercado que debían satisfacer los requerimientos del producto. Así, se escogieron los más aptos para los requisitos establecidos.
- Diseño e implementación del SW: se diseñó el SW con el objetivo de cumplir con la funcionalidad y requisitos planteados anteriormente en el documento. Posteriormente, se implementó este diseño haciendo uso del HW previamente escogido.
- Creación del prototipo final: se agregaron las diferentes líneas del proyecto en un producto inicial. Para ello, se soldaron todos los componentes HW, se realizó el montaje diseñado, se cosieron los elementos correspondientes y se cargó el SW entre otras tareas para formar el prototipo.

Diseño e Implementación

Durante la fase de decisión de la temática del proyecto, se expuso la idea de una manta o alfombra flexible que fuese interactiva con el usuario y que por lo tanto se emplease con fines educativos. Posteriormente, en la fase de diseño, se concretó que el producto se centraría en apoyar el aprendizaje de símbolos alfanuméricos en diferentes idiomas y estaría enfocado a un público infantil de entre 3 y 5 años. Además, se segregó la implementación en los siguientes ámbitos:

Hardware

La elección de los componentes se realizó teniendo en cuenta en todo momento la flexibilidad del conjunto, así como la interactividad del producto. Por consiguiente, se optó por tiras flexibles de LED WS2812B[1] organizadas en forma de "S" para conseguir una figura cuadrada, sustituyendo así a otros formatos de display flexibles que las superaban en precio. Por simplicidad, a partir de este momento se denominará matriz a la tira de LED, puesto que a todos los efectos son equivalentes excepto a lo relativo a la iluminación a través del software, que se explicará más adelante. Las dimensiones son de 8 cm x 8 cm en área y 5 mm de altura, y consta con una densidad de píxeles de 1 pixel/cm^2 (64 píxeles en total).

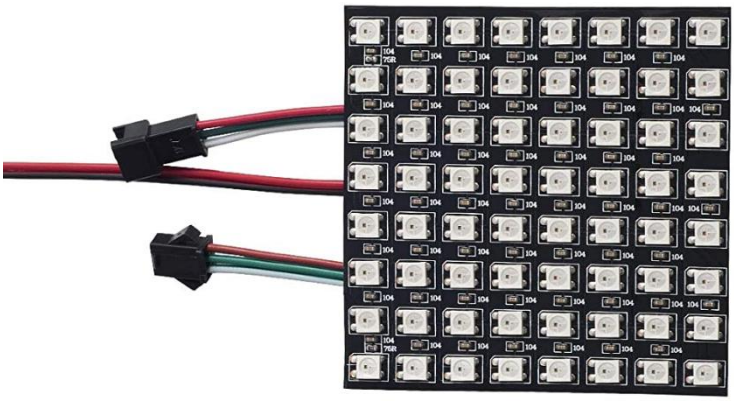


Ilustración 1: Matriz de LED WS2812B

A continuación, se pasó a la elección de los elementos que generasen las señales de entrada del sistema. Tras valorar varias opciones, se optó por seguir el ejemplo expuesto en Intructables [3] en el que se empleaban sensores capacitivos[2] debido a su reducido tamaño y a su capacidad de activarse con el contacto sin necesidad de pulsación. Para otorgar resiliencia y durabilidad al diseño, se rodearon los chips de cartón pluma para generar una estructura sólida que posteriormente se envolvió en tela y cosió para conformar el botón. Cabe destacar que en el producto final esta estructura sólida se sustituiría por algo más agradable al tacto y acolchado, como espuma compresible. Por último, en la región libre de cartón pluma, encima de los sensores, se realizó una costura de hilo conductor (nylon) para que cerrase el circuito cuando se presionase la tela y ésta se acercase al sensor.

[1] Obtenidas a través de Amazon y AliExpress.
[2] Obtenidos a través de Amazon.

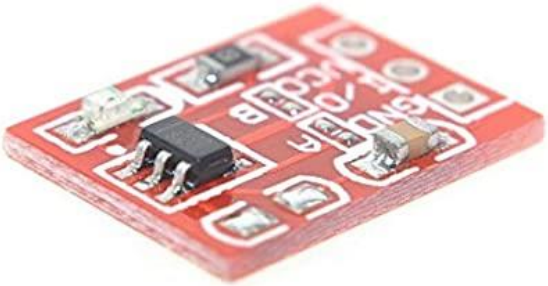


Ilustración 2: Sensor capacitivo TTP223.

Ilustración 3: Microchip ESP32.

Una vez elegidos los elementos imprescindibles del circuito, se procedió a seleccionar el microcontrolador. Inicialmente se propuso emplear un Arduino Uno por su simplicidad y accesibilidad, aunque se desestimó esta idea al comprobar que no tenía un manejo de las interrupciones ni un número de puertos acorde a las necesidades del diseño. De esta manera, se concluyó que un modelo ESP32[3] sería óptimo para el proyecto. El único problema era que la alimentación de algunos componentes debía ser de 5V y este microchip solo ofrecía 3.3V, pero después de unas modificaciones del diseño y las comprobaciones correspondientes se solventó este inconveniente.

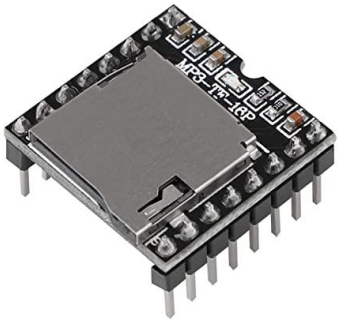


Ilustración 4: DF-Player

En relación con el sonido del dispositivo, se valoró almacenar las diferentes pistas de audio en el microcontrolador, pero esto exigía emplear uno que contase con la memoria suficiente. Como alternativa surgió la posibilidad de incorporar un dispositivo adicional que leyese de una memoria externa los diferentes sonidos. Este dispositivo llamado DF-Player[4] emplea una tarjeta SD como fuente y transmite los datos al altavoz[5] del producto.

Por último, la alimentación del sistema se realizó con una agrupación de cuatro pilas AA colocadas en serie que ofreciesen un total de 6 V.

Software

En esta sección se explicarán las diferentes directrices de diseño respectivas a la parte SW del proyecto. Además, se comentarán los aspectos relativos a la implementación que se consideren más importantes.

En primer lugar, se tuvo que tomar una decisión en cuanto al entorno de desarrollo sobre el que se llevaría a cabo la implementación. Se buscó un entorno gratuito, fácil y con un nivel de

[3] Obtenido a través de Amazon.
[4] Obtenido a través de Amazon.
[5] Obtenido en la tienda Telkron.

profesionalidad aceptable. De esta manera, se trabajó en *Visual Studio Code* mediante la extensión *PlatformIO*[6].

Por otro lado, en cuanto al *framework* utilizado, el ESP32 permite trabajar tanto con Arduino como con ESP32-IDF. Se escogió el primero por considerarse más adecuado para el objetivo del proyecto al tener gran accesibilidad a las librerías específicas para cada dispositivo HW y al ejercer una abstracción lo suficientemente precisa para el uso requerido.

Adicionalmente, se creó un repositorio en *GitHub* para poder realizar un control de versiones, soportar el trabajo en equipo y crear ramas paralelas a la principal en las que poder desarrollar sin temor a equivocarse.

Con este escenario planteado, se comenzó a diseñar la máquina de estados o FSM que gobernaría el comportamiento del sistema. Se optó por una máquina de tipo Mealy para tener constancia de cuando se producía cada acción (al producirse la transición). Además, esta ventaja propició la activación de una función de ahorro de consumo energético mediante un *light sleep*[7] cuando no se realizase ninguna acción. De esta manera la máquina de estados se muestra a continuación:

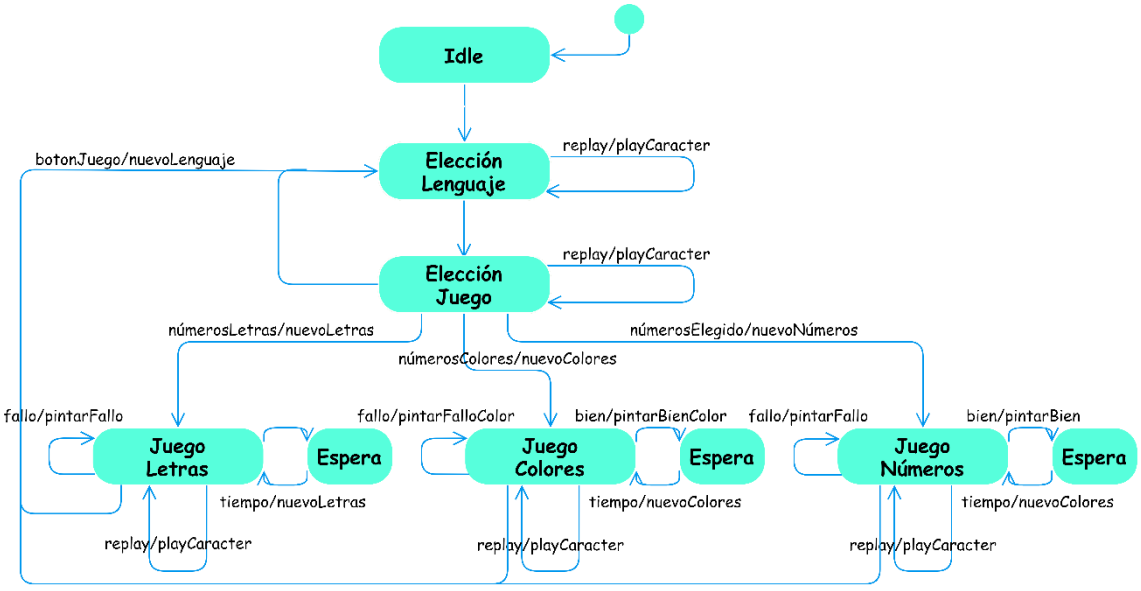


Ilustración 5: FSM implementada en el proyecto.

El comportamiento descrito por la FSM es el siguiente. En primer lugar, se debe elegir el lenguaje pulsando el botón correspondiente. Entonces, se escogerá el juego deseado pulsando alguno de los botones asociados. Posteriormente, se iniciará el juego eligiendo un carácter correcto y tres que no lo sean para mostrarlos en las matrices. Si se falla, se procederá a marcar como errónea la matriz correspondiente. Si se acierta, se marcará como correcta. Tras esto, se esperará un tiempo determinado y se volverá a jugar. Si se desea cambiar de juego, se debe pulsar el botón "nuevoJuego". Si se desea repetir la última pista interpretada, se debe pulsar el botón "repetirCaracter".

[6] PlatformIO:
[7] SleepModes ESP32

En cuanto a la implementación de la máquina de estados, se construyó haciendo uso de los archivos *fsm.cpp* y *fsm.h*[8] usados con anterioridad en varias asignaturas de la carrera. Así, solo se debían implementar las funciones de guarda y las correspondientes a cada transición.

Es importante mencionar que las transiciones llevan asociadas un cambio de los gráficos mostrados en las matrices y, en la mayoría de las veces, una reproducción de una pista de audio. Por ello, para la implementación de estas funciones es necesario hacer uso de las librerías que permiten manejar el DF-Player y las matrices.

La librería del DF-Player permite instanciar un objeto de la clase *DFRobotDFPlayerMini* con métodos que hacen posible el comienzo de la comunicación vía UART mediante dos pines configurados por el usuario, la elección del volumen deseado y la reproducción de la pista de audio que se desee. Es importante mencionar, que la librería admite navegar por el sistema de archivos de la tarjeta miniSD. Este hecho fue aprovechado para crear carpetas correspondientes a cada idioma. Dentro de estas carpetas la nomenclatura dada a los ficheros de audio era común, facilitando la reproducción de las pistas.

```
myDFPlayer.playFolder(language, num);
```

Ilustración 6:Instrucción con la que se reproduce el archivo num.mp3 en la carpeta language

En cuanto a la librería de las matrices, ésta permite crear objetos de la clase *Adafruit_NeoPixel* que han sido usados para representar cada una de las matrices. Con los métodos de la clase, se puede fijar los valores RGB y el valor de brillo para cada píxel de la matriz. Como ya se ha mencionado, el término matriz es en realidad erróneo, puesto que se trata de una tira de 64 píxeles de forma cuadrada. Por consiguiente, para el encendido de cada led se tuvo que hallar la correspondencia entre el píxel en cuestión y su equivalencia matricial.

El procedimiento de trabajo de las matrices es el siguiente: las funciones de transición de la FSM cambiaban el carácter que debía ser representado en cada matriz (valores RGB y de brillo) mientras que otra función se encargaba de refrescar lo mostrado en las matrices de forma recursiva. Este comportamiento se implementó mediante una aproximación por objetos.

En la siguiente figura se muestra el diagrama de clases simplificado en el que se basa el proyecto. *Fsm_t* es la clase que representa la máquina de estados donde *fsm_fire* es el método utilizado para accionarla. *Fsm_data_t* es la clase que contiene los datos y funciones necesarias para el funcionamiento de la FSM. Sus métodos son usados para cambiar la representación de las matrices (uso del método de la clase *Adafruit_NeoPixel* con la información del atributo *caracterActual*) y para cambiar el carácter que debe ser mostrado por éstas (cambio de los valores de *caracterActual*). *MatrizLED_t* es la clase que representa cada una de las matrices (la clase *Adafruit_NeoPixel* no está incluida debido a que se ha generado como variable estática). Por último, *caracter_t* representa cada uno de los caracteres.

[8] Apéndice A

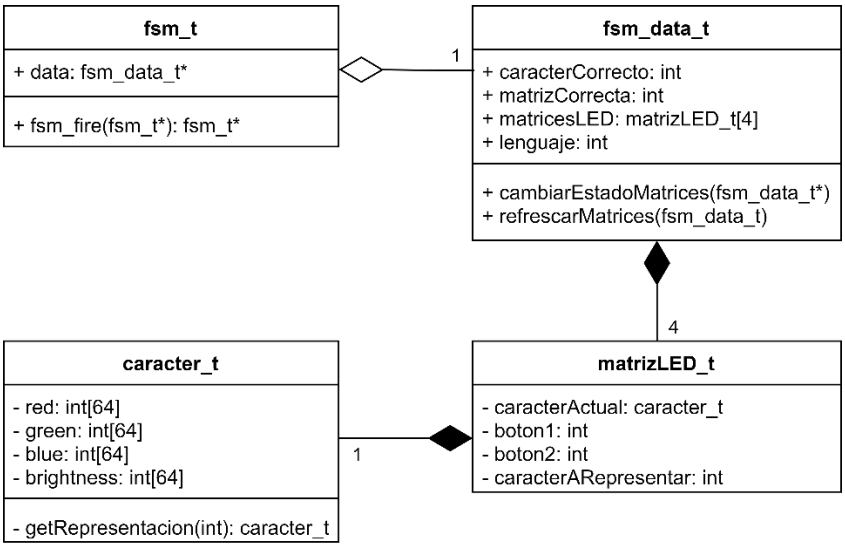


Ilustración 7: Diagrama UML de las clases relacionadas con las matrices

Finalmente, en cuanto a los botones, estos son asociados a cada matriz mediante el puerto al que están conectados (atributos *boton1* y *boton2* de *matrizLED_t*). Su actuación era captada por interrupciones vinculadas a cada puerto GPIO usado. De esta manera, se almacenaba el botón que había causado la última interrupción y, así, se conseguía realizar las transiciones correctamente.

Prototipo Final

Una vez adquirido el Hardware y programado el Software, se unió todo con el fin de conseguir el prototipo final. Este no cubre todos los objetivos fijados de la idea inicial, pero para la fabricación del producto final se harían modificaciones que cumplieran con todos ellos, haciendo del producto uno totalmente jugable, flexible y a prueba de niños.

Para la creación del prototipo, se siguió un proceso el cual se describirá a continuación. Para su fabricación, lo primero que se hizo fue la soldadura de todos los cables a las matrices y botones, después se cosieron los botones con la tela conductora y, por último, se implementó todo a la espuma, caja y ESP32.

En primer lugar, la soldadura se realizó en un laboratorio de la universidad con el esquema siguiente y el resultado obtenido:

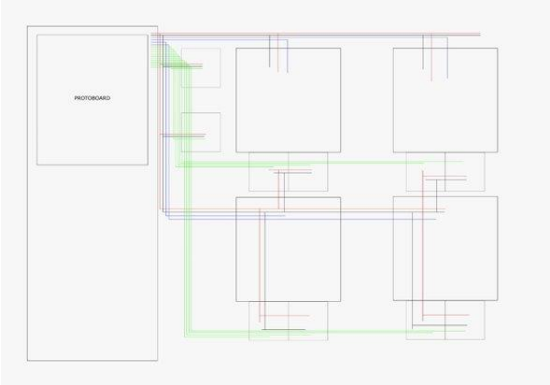


Ilustración 9: Esquema de referencia.

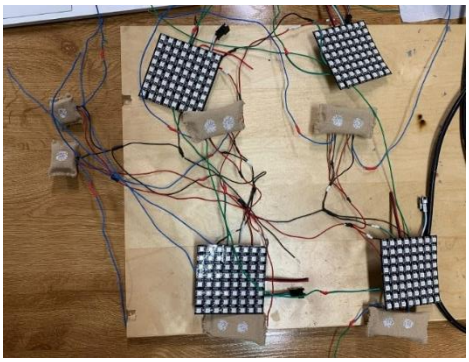


Ilustración 8: Resultado obtenido.

Como se observa, los cables salientes de la caja son los siguientes:

- 4 cables de datos de salida, uno para cada matriz.
- 1 cable a tierra que se distribuye a cada matriz y botón.
- 1 cable de Vcc que también se distribuye por todas las matrices y botones.
- 10 cables de datos de entrada, uno para cada sensor capacitivo.

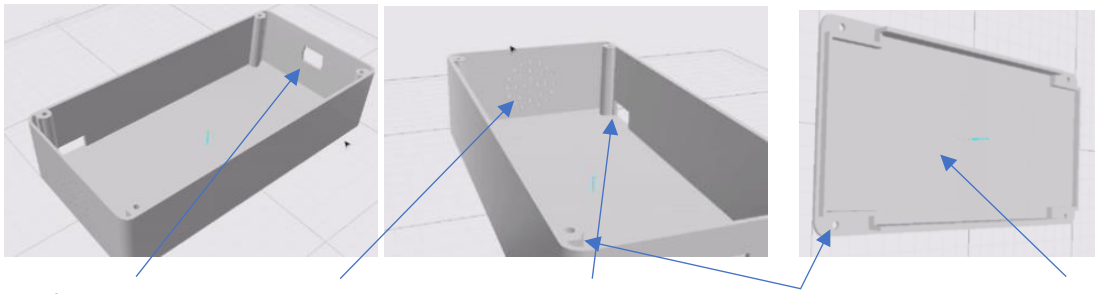
Por otro lado, los botones que se encuentran a la izquierda sirven, el superior para volver al estado inicial y jugar a otro juego (botón “nuevoJuego”) y el inferior para repetir la pregunta en caso de no haberla oído (botón “repetirCaracter”).

Para la conexión de las matrices con el ESP32, la idea inicial consistía en poner todas las matrices en línea para soldar solo un cable a la protoboard, pero al realizarse las pruebas, las matrices colapsaban y se decidió separar los cables a uno por matriz.

Por último, en el lado izquierdo del esquema se dejaría para para la protoboard, el micro, el DF-Player y el altavoz.

En cuanto al diseño de la caja, con la página vectary.com, se diseñó este componente con las siguientes características: un agujero en una esquina para sacar los cables, otra ristra de agujeros pequeños en un lateral para la salida del sonido, donde se pegará el altavoz y otro agujero en el lado contrario, para el botón basculante, que su única función es encender el micro. La caja se cierra con una tapa atornillada y en ella se ha puesto el logo de nuestro producto, “Toy2Joy”. La caja es rígida, ya que se tenía que proteger la protoboard de la manipulación de niños y para ello

se necesitaba salir un poco de la idea de flexibilidad para esta parte del proyecto. Sin embargo, para el producto final se diseñará una PCB flexible que permita cumplir la especificación inicial de flexibilidad.



Por último, para el diseño del prototipo final en el que se unió todos los componentes, se siguió el siguiente proceso:

- Se pegaron 6 capas de goma eva como base para la protección de los cables situados entre la goma y la espuma.
- Se cortó la espuma al tamaño preciso, haciendo un agujero correspondiente a cada matriz y cada botón para la posterior introducción de los mismos.
- Se rellenaron estos agujeros con un poco de espuma sobrante para insertar la matriz y los botones y que estos se quedaran a nivel.
- Se metieron los cables por el orificio de la caja para su conexión con el ESP32.
- Se recubrió con tela para adornar y cubrir los elementos del sistema.

El prototipo final queda reflejado en las siguientes imágenes:

Ilustración 10: Montaje del sistema.

Ilustración 11: Prototipo final.

Comercialización

El desarrollo de este prototipo llevó a pensar qué modificaciones se llevarían a cabo cuando este juguete se introdujera al mercado. Como se mencionó antes, se tenía una visión del producto como una manta interactiva, flexible, resistente. Sin embargo, durante el desarrollo del producto, se afrontaron dificultades técnicas que reflejaron que las características técnicas planteadas inicialmente eran demasiado ambiciosas para un primer prototipo.

Consecuentemente, se plantean los siguientes cambios de cara a la comercialización:

- Cambiar toda la circuitería interna del prototipo por una PCB flexible de forma que se simplifique y sistematice todo el cableado y la colocación de los componentes.
- Cambiar los materiales de fabricación por otros más adecuados, como por ejemplo otro tipo de telas que sean impermeables como el nylon o el neopreno, aportando más resistencia que una tela convencional.
- Optimizar el diseño en cuanto a estética, para que quede más pulido y dar un toque de profesionalidad, así como intentar reducir el peso al mínimo posible para favorecer la manejabilidad de los niños.

El coste de producción total de este proyecto ha sido de 56€ Este coste se podría reducir a más de la mitad debido a que los cortos plazos de entrega disponibles encarecían el coste. Del total, casi la mitad del precio se ha debido a las matrices de LEDs, ya que es el factor más importante en nuestro proyecto. El elemento más costoso después de las matrices ha sido el altavoz. El resto se distribuye en materiales, componentes, cables, etc.

PROTOTIPO	
COMPONENTES	PRECIO (€)
Sensores capacitivos (8)	8,46
Hilo conductor	1,50
Espuma	1,20
Cables	3,60
Botón	2,20
Altavoz	5,00
Goma eva	4,50
ESP32	3,09
Matrices LEDS (4)	23,20
DF Player	3,25
TOTAL 56 €	

Haciendo un estudio de los precios a largo plazo, sin fechas de entrega inmediatas, pidiendo al por mayor y con mayor flexibilidad temporal, se abaratarían los costes. Se ha estimado que el coste total del producto oscilaría en torno a los 24 €. Aplicando un porcentaje de 100% para obtener ganancias, finalmente, el precio de cara al mercado podría ser de 47,50 €. Añadiendo los gastos de logística, el precio final propuesto es de 49.99 €.

PRODUCTO FINAL	
COMPONENTES	PRECIO (€)
Sensores capacitivos (8)	6,08
Hilo conductor	1,20
Espuma	0,80
Cables	1,50
Botón	0,38
Altavoz	0,16
Goma eva	0,48
ESP32	1,10
Matrices LEDS (4)	11,20
DF Player	0,85
TOTAL 23,75€	

Conclusión y Mejoras

En cuanto al futuro relacionado con este proyecto, se pretenden realizar varias mejoras y cambios aparte de los necesarios para posibilitar la comercialización.

Las posibles mejoras que se proponen son las siguientes:

- Mejorar la implementación de manera que cada juego no se vea representado de forma individual en la máquina de estados si no que sea tratado como una clase. De esta manera, el sistema será más escalable y será más fácil crear nuevos juegos[9].
- Permitir una conexión externa por medio de un dispositivo (móvil, tablet, ordenador) a Través del cual se puedan descargar de la página oficial. De esta forma los padres/profesores podrán cambiar los juegos del producto con mucha facilidad. También se creará un apartado para que la gente pueda diseñar sus propios juegos y, tras una verificación por parte del equipo, validarlos y subirlos a la web para que la comunidad pueda disfrutar de ellos.
- Llevar el producto a distintos tamaños para hacerlo compatible con nuevos juegos y escalar los ya existentes a las nuevas dimensiones. Adicionalmente, se podría no sólo aumentar el número de matrices sino también aumentar la densidad de píxeles de las mismas, para ofrecer una mayor resolución y ser capaces de representar imágenes más complejas y elaboradas al usuario, mejorando así, la calidad del producto.

[9] Apéndice B

Bibliografía

[1] C. Requero, «La desmotivación de los niños en el colegio.,» hacerfamilia.com, 24 01 2018. [En línea]. Available: https://www.hacerfamilia.com/educacion/desmotivacion-colegio-20180123143947.html.

[2] M. C. G. CASTELLÓN, «UNIR - La Universidad en Internet,» 26 07 2018. [En línea]. Available: https://www.unir.net/educacion/revista/no-solo-los-ojos-tambien-los-oidos-son-clave-para-el-aprendizaje-escolar/. [Último acceso: 05 2020].

[3] Plusea, «Intrctables Circuits,» 2008 08 23. [En línea]. Available: https://www.instructables.com/Three-Fabric-Buttons/. [Último acceso: 2021 03 15].

[4] C. Requero y R. Regidor, «El desarrollo de la memoria de los niños,» hacerfamilia.com, 08 06 2020. [En línea]. Available: https://www.hacerfamilia.com/psicologia/noticia-desarrollo-memoria-ninos-20150414100822.html.

Apéndice A

El código que conforma el archivo *fsm.cpp* es el siguiente.

```
#include <stdlib.h>
#include "fsm.h"

fsm_t*
fsm_new (int state, fsm_trans_t* tt, void* user_data)
{
  fsm_t* fsm = (fsm_t*) malloc (sizeof (fsm_t));
  fsm_init (fsm, state, tt, user_data);
  return fsm;
}

void
fsm_init (fsm_t* fsm, int state, fsm_trans_t* tt, void* user_data)
{
  fsm->current_state = state;
  fsm->tt = tt;
  fsm->user_data = user_data;
}

void
fsm_destroy (fsm_t* fsm)
{
  free(fsm);
}

void
fsm_fire (fsm_t* fsm)
{
  fsm_trans_t* t;
  for (t = fsm->tt; t->orig_state >= 0; ++t) {
    if ((fsm->current_state == t->orig_state) && t->in(fsm)) {
      fsm->current_state = t->dest_state;
      if (t->out)
        t->out(fsm);
      break;
    }
  }
}
```

El código que forma parte de *fsm.h* es el siguiente:

```
#ifndef FSM_H_
#define FSM_H_

typedef struct fsm_t fsm_t;

typedef int (*fsm_input_func_t) (fsm_t*);
typedef void (*fsm_output_func_t) (fsm_t*);

typedef struct fsm_trans_t {
  int orig_state;
  fsm_input_func_t in;
  int dest_state;
  fsm_output_func_t out;
} fsm_trans_t;

struct fsm_t {
  int current_state;
```

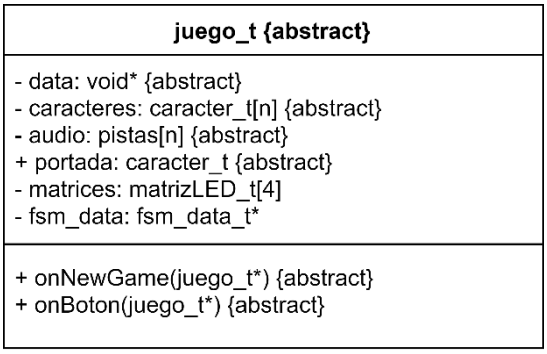
```
  fsm_trans_t* tt;
  void* user_data;
};

fsm_t* fsm_new (int state, fsm_trans_t* tt, void* user_data);
void fsm_init (fsm_t* fsm, int state, fsm_trans_t* tt, void* user_data);
void fsm_fire (fsm_t* fsm);
void fsm_destroy (fsm_t* fsm);

#endif /* FSM_H_ */
```

Apéndice B

Para mejorar la escalabilidad del proyecto se ha pensado aplicar una aproximación por objetos. A continuación, se muestra un posible diagrama de clases a partir del cual poder comenzar a desarrollar la idea.



Se basará en la definición de una clase abstracta que debe ser implementada por el usuario. Se deberá implementar métodos que llamar al comenzar un juego nuevo y al pulsar uno de los botones. Tendrá como atributos a implementar los caracteres que se utilizarán en ese juego, el audio utilice y el carácter que debe ser mostrado en la pantalla de selección de juego.

Además, hará uso de la información de las matrices (para indicar el carácter a representar, que botones hay asociados a que matrices).

De esta manera, la clase *fsm_data_t* deberá albergar los diferentes juegos y asociarlos a una matriz de la pantalla de inicio (que a su vez está asociada a unos botones determinados).