Simon Says

Leticia Badillo Prieto
Grado de Tecnologías de Telecomunicación
Sistemas encastados

Jordi Bécares Ferrés
Pere Tuset Peiró

Junio 2018

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Simon Says*
Nombre del autor:	*Leticia Badillo Prieto*
Nombre del consultor/a:	*Jordi Bécares Ferrés*
Nombre del PRA:	*Pere Tuset Peiró*
Fecha de entrega (mm/aaaa):	MM/AAAA
Titulación:	*Grado de Tecnologías de Telecomunicación*
Área del Trabajo Final:	*Sistemas encastados*
Idioma del trabajo:	*Español*
Palabras clave	*Juego, empotrado, Simon*
Resumen del Trabajo:	

El proyecto está basado en el entorno de los sistemas embebidos. Los sistemas empotrados nos permiten realizar tareas de manera eficiente y optimizando los recursos utilizados.

En la actualidad existen multitud de sistemas embebidos en nuestra vida diaria.

El sistema empotrado que se va a construir consiste en la creación de un juego de los ochenta , famoso por su aparatoso sistema electrónico de diseño circular, secuencia de colores y sonidos. Este juego ayuda a fortalecer las habilidades de memoria visual, hablamos del juego Simon.

Para llevar a cabo el desarrollo del juego, se tiene que elegir el hardware y el software que más se adecue a nuestras necesidades.

Tal y como se ha planteado la implementación del juego, el hardware nos deberá proporcionar todo lo necesario para que el usuario pueda interactuar con el juego.

El software a implementar dotará al hardware de toda la funcionalidad que requiere. El software se debe pensar para el hardware seleccionado, así se obtendrá un software optimizado y se podrá sacar el máximo partido posible al hardware seleccionado.

Abstract (in English, 250 words or less):

The project is based on the scope of embedded systems. Embedded systems allow us to perform tasks efficiently and optimize the resources used.

In today's world, there are a lot of embedded systems in our life.

The system that has been built, is a game of the 80s. It's a electronic game with round shape. It has sounds and colours. This game is called Simon.

To develop the game, it's necessary to choose the correct hardware and software.

The used hardware must have an interface where the user can play with the game.

The choosen software gives the hardware everything to need. The software is specific to the hardware, so It's a optimized software.

Índice

Lista de ilustraciones

Lista de tablas

1. Introducción

1.1 Contexto y justificación del trabajo

En el proyecto se desarrolla el juego de Simon Says. Este juego fue creado en 1978 por Ralph Baer y Howard J. Morrison siendo uno de los precursores en la industria de los videojuegos que por aquella época empezaba a nacer. Se trataba de una consola circular de juego que encendía luces y emitía sonidos que el jugador debía repetir en el mismo orden de la secuencia mostrada, pulsando las teclas de colores.

No fue hasta mediados de los noventas cuando este juego se desarrolló en la versión arcade. Fue desarrollado para Little Caesars Pizza y lo llamaron Brain Teaser.

A lo largo de los años la empresa que creó el juego fue lanzando diferentes modelos. Uno de los más populares fue Simon Pocket. Otras compañías también lanzaron juegos similares, siendo éstos un calco a Simon. Por ejemplo, la compañía Atari lanzó Touch Me que era una maquinita que disponía de una pantalla led donde se veía la puntuación y una botonera de colores para que el jugador reprodujera la secuencia. La finalidad del juego era el mismo que el de Simon.

Hoy día este juego lo comercializa la empresa Hasbro, ha lanzado al mercado nuevas variantes pero siempre conservando el estilo original de Simon.

Actualmente todos los juegos que pueden existir para máquinas arcades o como el caso de Simon, una especie de máquina arcade, pueden ser desarrollados bajo sistemas embebidos que permiten reducir el tamaño del dispositivo. También gracias a los sistemas embebidos se puede disponer de varios juegos en un mismo dispositivo. Estos juegos comparados con los de las máquinas arcades ofrecen mejor usabilidad, conectividad y gráficos.

En el siguiente proyecto, se va a desarrollar el juego clásico como se verá de forma más detallada en los siguientes apartados. Estará formado por un Launchpad de Texas Instruments, un kit que contiene la pantalla, joystick y botones para jugar y un módulo WIFI que lo dota de conectividad.

1.2 Descripción del trabajo

El proyecto consiste en una vez realizado un estudio de los elementos hardware y software existentes ser capaz de seleccionar la placa y periféricos de desarrollo, así como conocer la mecánica del juego Simon para adaptarlo al sistema embebido que se ha seleccionado.

Para conseguir esto será necesario buscar los elementos adecuados, seleccionar el entorno de desarrollo que más nos convenga. También es necesario saber las librerías existentes para el control de los elementos que interactúan con el usuario, así como crear las que necesitemos. Con toda la información software disponible se tiene que ser capaz de desarrollar el juego en el entorno que se ha elegido, adaptándolo a la pantalla, movimiento de joystick, etc.

La mecánica del Simon consiste en la repetición por parte del jugador de una secuencia mostrada.

El juego ejecuta una secuencia de cuatro colores asociado a cuatro sonidos que el usuario deberá memorizar para posteriormente reproducirla.

La forma de reproducir la secuencia es accionando el pulsador del color correspondiente, en nuestro caso, será moviendo el joystick en la dirección correcta.

La secuencia se debe reproducir tal cual el juego la ha mostrado, si no se reproduce exactamente igual, el jugador habrá perdido.

El juego dispone de varios niveles, en cada nivel la secuencia a reproducir será mayor. El primer nivel la secuencia dispone de cuatro valores. La dificultad irá aumentando y la secuencia pasará a tener una longitud mayor, incrementándose en cada nivel en dos.

Existe también la variante multijugador, se conectarán varios jugadores en línea y se reproducirá la misma secuencia para todos. Ganará el usuario que mayor puntuación obtenga.

El juego termina cuando el usuario falla una de las secuencias reproducidas.

1.3 Objetivos del TFG

El objetivo principal del proyecto es el desarrollo y correcto funcionamiento del juego Simon corriendo sobre la placa de desarrollo. Para obtener este objetivo, se han creado objetivos parciales divididos en principales y secundarios. Los objetivos principales serán aquellos que doten de la funcionalidad principal al juego y los objetivos secundarios serán los que añaden valor a la aplicación.

1. Objetivos principales
   a. Generación de secuencias
   b. Incluir distintos niveles de dificultad en el juego
   c. Guardado de las partidas
   d. Dotar al sistema de wifi para partidas multijugador
   e. Generación de secuencias multijugador
   f. Pausar juego
   G. Mostrar puntuación

2. Objetivos secundarios
   a. Login usuarios
   b. Partidas simultáneas entre jugadores
   c. Jugar entre usuarios que se conectan desde fuera de la red donde está ejecutandose el servidor
   d. Incluir sonido
   e. Mostrar menú

1.4. Enfoque y método seguido

Existen varias estrategias para elaborar un proyecto:

1. Desarrollar un producto nuevo. Consiste en desarrollar algo desde cero, sin nada que nos sirva de guía. Se tendrán que definir los objetivos acorde con los requisitos que se quieran.
2. Adaptar un producto existente. Esta estrategia intenta aprovechar un producto que ya existe en el mercado y a partir de ahí adaptarlo a nuestras necesidades.
3. Desarrollar un producto nuevo a partir de uno existente. Se buscan elementos que ya existen que nos sirvan de base para nuestro desarrollo. A partir de ellos se modifican y se adaptan para obtener el objetivo final.

Para la realización del proyecto la estrategia que más nos conviene es la de "Desarrollar un producto nuevo a partir de uno existente".
Nos basta con encontrar los elementos hardware que cumplan con nuestros objetivos y adaptarlo a nuestras necesidades. A nivel funcional también se partirá del juego existente y se adaptará al sistema embebido elegido.

A continuación, se definen las etapas fundamentales en el desarrollo del proyecto:

1. Documentación y análisis

Esta primera fase es fundamental para la implementación del juego. Es una fase de investigación donde hay que ver que opciones hardware (placa de desarrollo, periféricos, etc) y software (sistema RTOS a utilizar) existen y cuales se adaptan a nuestros requisitos. Es la etapa más laboriosa ya que hay que estudiar de manera minuciosa las diferentes opciones, ya que una mala opción podrá repercutir a futuro en el desarrollo de la aplicación.

Una vez que se ha estudiado las opciones de hardware y software hay que estudiar también el funcionamiento del juego, sus reglas, el desarrollo del mismo y ver cuales son viables para implementar y cuales no.

2. Desarrollo y pruebas

Una vez seleccionados los elementos hardware, el desarrollo de la aplicación se ejecuta bajo el IDE elegido. A la vez que se va elaborando el código que compondrá la aplicación, se realizan pruebas sobre ella para detectar posibles errores.

A partir de los requisitos de la aplicación, se llevan a cabo unas series de pruebas para cumplir con los casos de usos de la aplicación. Si se detectara algún error, en cada iteración de las pruebas se solucionarán esos errores para así tener la aplicación limpia de errores.

Sobre cada versión mejorada se vuelve a probar y a detectar que los fallos se han solucionado y no se han introducido más que perjudiquen a los objetivos tanto principales como secundarios que se marcaron en el comienzo del proyecto.

3. Memoria

Debe existir un documento donde quede reflejado todo el trabajo realizado a lo largo del proyecto. Este documento es la memoria que se desarrollará en paralelo con la ejecución del proyecto. La memoria contendrá todas las fases de las que se compone la realización del trabajo, así como un manual de usuario, posibles errores y posibles mejoras que ha futuro se pudieran implementar.

1.5. Planificación del trabajo

Como se ha comentado en el punto anterior, el proyecto se ha dividido en tres partes.

1. Documentación y análisis. En esta fase se incluyen las tareas:
    a. Investigación y documentación sobre las herramientas de desarrollo que se van a utilizar.
    b. Documentación sobre las librerías existentes para comprobar que se adecuan a nuestro proyecto y si no es así la creación de nuevas librerías tanto para el manejo del joystick como del LCD.
    c. Documentación e investigación sobre el montaje de servidor para las partidas online.
    d. Documentación e investigación sobre el montaje de BBDD para almacenamiento de los datos de las partidas online.
    e. Pruebas y estudio de los ejemplos proporcionados por Energia para entender el funcionamiento de los diferentes elementos del Educational BoosterPack, así como del MSP432 y el módulo wifi.
2. Desarrollo y pruebas. En esta parte del trabajo se encuentra todo lo relacionado con la codificación y pruebas de la aplicación. Se divide en diferentes tareas a fin de cumplir con los objetivos marcados. Estos objetivos son:
    1. Objetivos principales
        a. Generación de secuencias
        b. Incluir distintos niveles de dificultad en el juego
        c. Guardado de las partidas
        d. Dotar al sistema de wifi para partidas multijugador
        e. Generación de secuencias multijugador
        f. Pausar juego
        g. Mostrar puntuación

2. Objetivos secundarios
    a. Login usuarios
    b. Partidas simultáneas entre jugadores
    c. Jugar entre usuarios que se conectan desde fuera de la red donde está ejecutandose el servidor
    d. Incluir sonido
    e. Mostrar menú
3. Memoria. Aquí es donde se van a desarrollar todas las tareas que tienen que ver con este documento. También incluye la parte de la creación de la presentación.

A continuación se muestra la planificación inicial y la planificación final ya que las tareas para las diferentes fases del proyecto se han tenido que ir reestimando y reordenando para cumplir los objetivos.

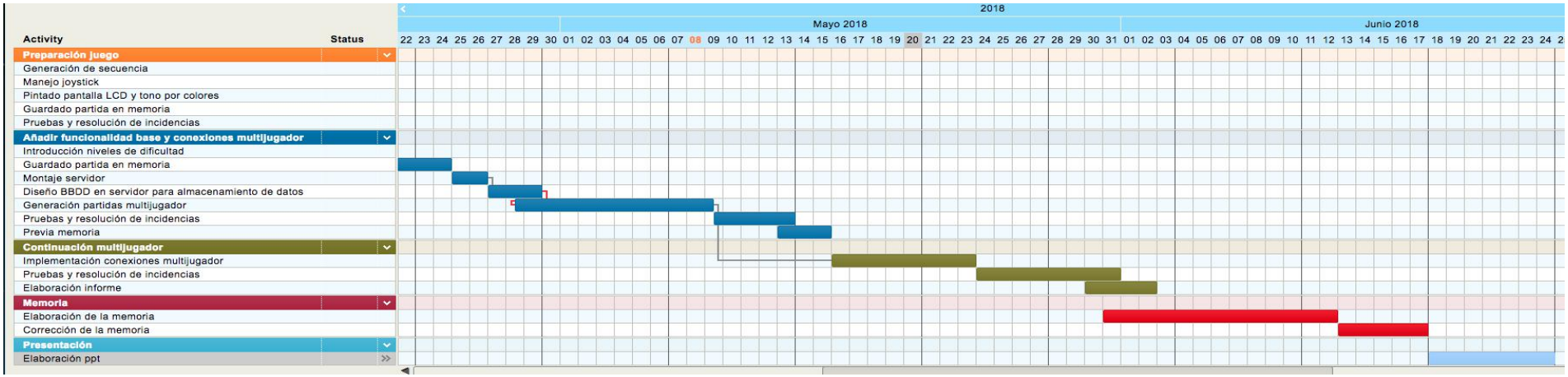
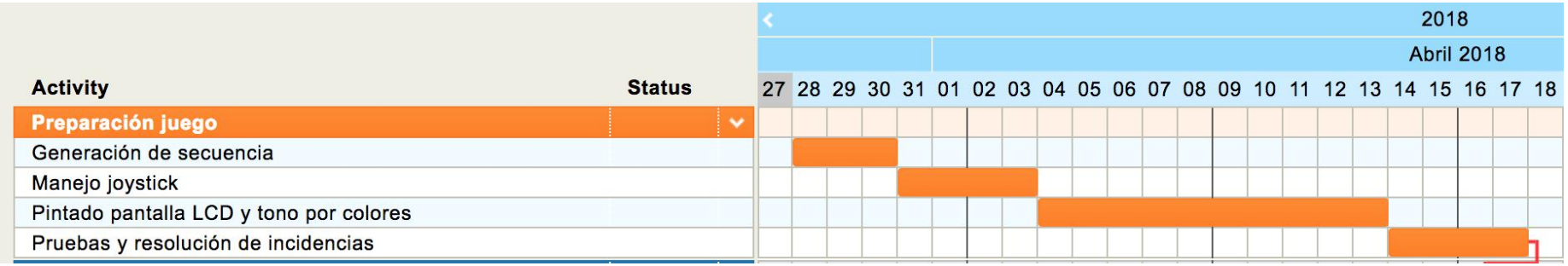
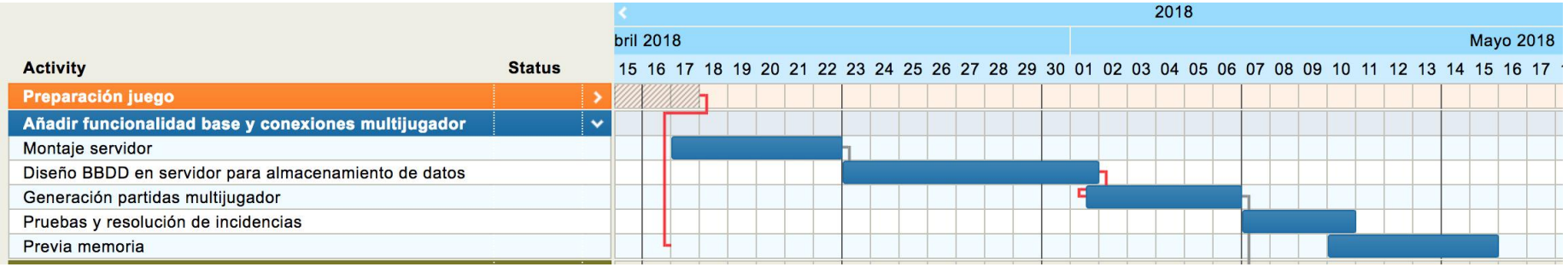


Ilustración 1-Planificación inicial


2018
Abril 2018
Activity
Status
27 28 29 30 31 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18
Preparación juego
Generación de secuencia
Manejo joystick
Pintado pantalla LCD y tono por colores
Pruebas y resolución de incidencias


2018
bril 2018
Mayo 2018
Activity
Status
15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17
Preparación juego
Añadir funcionalidad base y conexiones multijugador
Montaje servidor
Diseño BBDD en servidor para almacenamiento de datos
Generación partidas multijugador
Pruebas y resolución de incidencias
Previa memoria

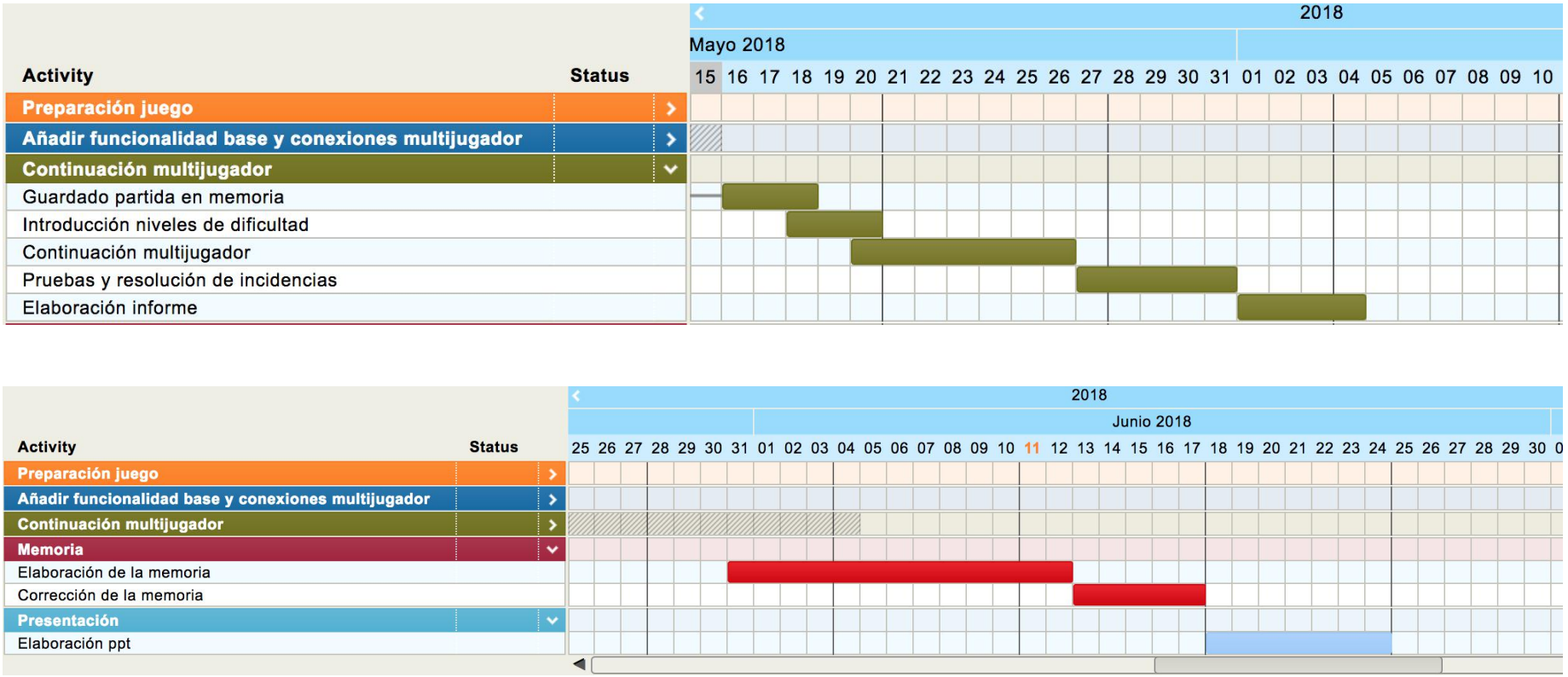


Ilustración 2 - Planificación final

Para poder llevar a cabo el seguimiento de las tareas, desde el comienzo se elaboró un diagrama de Gantt que ha ido evolucionando a medida que se ha desarrollado el proyecto.
Los diagramas se pueden ver en la ilustración 1 e ilustración 2.

Como se puede observar, al igual que ocurre en los proyectos reales, suele haber diferencias entre la planificación inicial y la planificación final.
A grandes rasgos las diferencias más significativas han sido que las tareas se han tenido que cambiar de fase, se han priorizado algunas como por ejemplo la generación de partidas multijugador y otras como el guardado de memoria o la pausa del juego han pasado a un segundo plano, desarrollándose después cuando el grueso de la aplicación estaba concluida.

1.6. Recursos empleados

Para el desarrollo de este proyecto se han utilizado tanto elementos hardware como software.

Estos elementos son:

1. Kit de sistemas embebidos de la UOC:

    - Launchpad MSP432P401R
    - BOOSTXL-EDUMKII - Educational BoosterPack MKII
    - CC3100 BoosterPack
    - PC Portátil :MacBook Pro (2,7 GHz Intel Core i5,8GB RAM y 125GB de disco duro)
    - PC Portátil: Intel Core i5 (2.4GHZ ,16GB RAM y 500GB de disco duro)

2. Elementos software:

    - Window 7 64 bits
    - MacOs Sierra
    - Pages
    - Code Composer Studio
    - Eclipse
    - BBDD oracle

3. Documentación:

- http://energia.nu/guide/
- http://energia.nu/reference/
- https://www.hackster.io/
- https://www.freertos.org/
- http://www.43oh.com/
- Sistemas empotrados en tiempo real. José Daniel Muñoz Frías.

1.7. Producto obtenido

El resultado de juntar los elementos hardware y software han dado la implementación del juego Simon. Esto se ha obtenido gracias a que con el KIT proporcionado se es capaz de ofrecer al usuario de una interfaz compuesta por el joystick, la pantalla, el buzzer y los botones que permiten jugar. Así como también el módulo WIFI dota a la aplicación de conectividad para las partidas en linea.

1.8. Breve descripción de los otros capítulos de la memoria

A continuación se muestra una descripción de los capítulos que va a contener la memoria.

Capítulo 2 : Antecedentes

En este capítulo, vamos a ver el estado del arte de los sistemas embebidos, donde están presentes y los principales sistemas operativos embebidos que hay, así como las razones que han llevado a elegir uno frente al otro. Se verá también las distintas alternativas del dispositivo seleccionado.

Capitulo 3: Descripción funcional

En este capítulo se verá todo lo relacionado con la funcionalidad del dispositivo, las características hardware y software empleadas, así como interactúan entre ellas.
Se explicará también en este capítulo los diferentes módulos, librerías y driver que componen el sistema.

Capítulo 4: Descripción detallada

Este capítulo tratará de una forma más detallada el hardware empleado, que son el Launchpad y el BoosterPack.

Capítulo 5; Viabilidad técnica

Se tratará de analizar si el proyecto es posible técnicamente con los recursos seleccionados y si con ellos se cumplen los objetivos marcados.
Capítulo 6 : Valoración económica
En este capítulo se estudiará el aspecto económico del proyecto, el coste necesario para llevarlo acabo. Estos costes incluirán tanto los recursos software, hardware y de personal de desarrollo.

Capítulo 7: Conclusiones

Este capítulo servirá para plasmar las conclusiones finales del proyecto, si se han cumplido los objetivos marcados. También se incluirán objetivos que han quedado por desarrollar y líneas futuras de trabajo.

Capítulo 8: Glosario

Contiene el glosario con los principales términos que se han utilizado a lo largo del proyecto.

Capítulo 9: Bibliografía

En este capítulo, veremos la bibliografía empleada para el desarrollo del proyecto y la memoria.

Capitulo 10; Anexos

En los anexos, se incluirán un manual de instalación, los esquemas técnicos del Launchpad y una guía de uso.

2. Antecedentes

2.1. Estado del arte

En este apartado se estudiará el estado del arte de los elementos que son necesarios para el desarrollo del proyecto.Estos son: sistemas embebidos y sistemas operativos embebidos.

2.2.1. Sistemas embebidos

El objetivo de un sistema embebido es llevar a cabo una serie de tareas de manera eficiente y optimizando los recursos disponibles.
Tienen que dar respuesta a la mayor variedad posible de escenarios. Esto se consigue gracias a los sistemas operativos.

Estos sistemas pueden controlar por ejemplo la temperatura de un lugar o como en nuestro caso el desarrollo de un juego. Los sistemas embebidos son dispositivos compuestos generalmente por un microprocesador y un software que se ejecute sobre este. Normalmente estos sistemas poseen una interfaz para interactuar con ellas. [1]

Los sistemas empotrados cuentan con una memoria donde se almacena el programa que se va a ejecutar.[2]

2.2.2. Sistemas operativos embebidos

Los sistemas embebidos se pueden programar en lenguaje ensamblador propio o utilizando compiladores específicos que utilizan C o C++. También se pueden utilizar lenguajes de alto nivel como Java, cuando no se requiere un tiempo de respuesta crítico. Por eso se opta por una programación ad hoc.

Para la realización del proyecto, ya que tiene que ser un sistema en tiempo real, se ha optado por TI-RTOS que es propio del fabricante del kit de desarrollo. Este sistema operativo ofrece todo lo necesario para el proyecto, tiene un gran número de driver, así como una gran ayuda en internet.Para facilitar el desarrollo del software, TI-RTOS también incluye controladores de consumo de energía que funcionan con el Kernel de TI-RTOS. Aquí se puede ver una lista de los controladores compatibles con TI-RTOS. TI-RTOS proporciona la estructura subyacente de Energia MT.

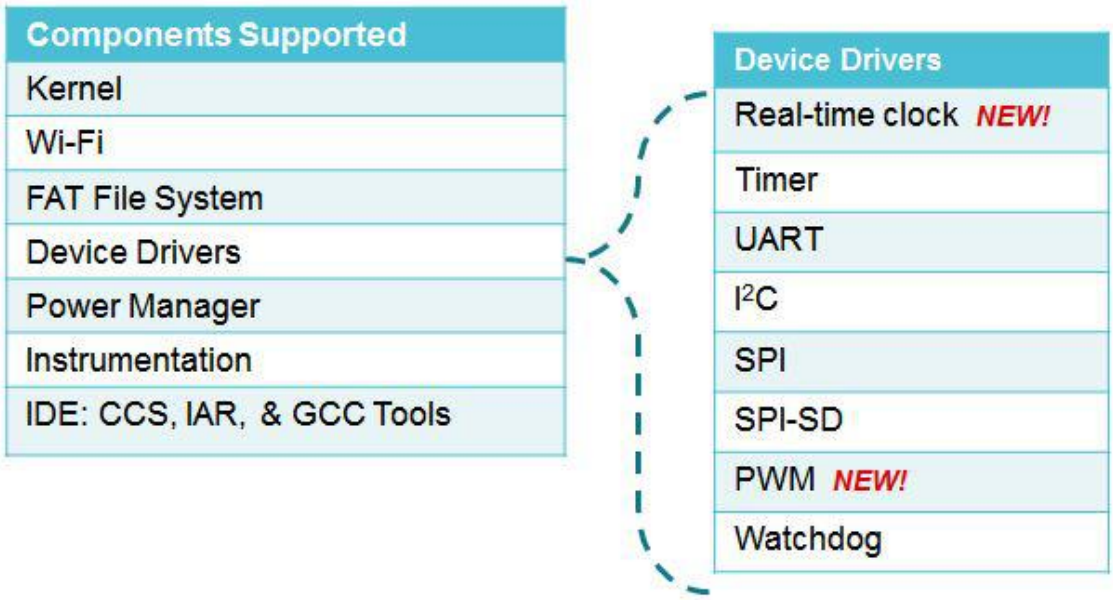


Ilustración 3 - Drivers soportados por TI-RTOS

Otra opción para realizar el proyecto hubiera sido utilizar FreeRTOS. Éste es un sistema operativo en tiempo real. Tiene licencia GPL pero con una excepción que permite "código propietario a los usuarios siguen siendo de código cerrado, manteniendo el núcleo como código abierto" . El núcleo de este sistema operativo está escrito en C para hacerlo más legible aunque tiene algunas rutinas en ensamblador. Con FreeRTOS hay una tarea ejecutándose a la vez,es el planificador de tareas el que decide cual tiene que ejecutarse. Las tareas tienen un orden de prioridad, así se ejecutarán de forma correcta.

Estudiadas las dos opciones se optar por utilizar Energia ya que la forma de programar la aplicación es más familiar para mi que si utilizara FreeRTOS. También se ha visto que TI-RTOS tiene todo lo necesario para nuestra aplicación.

Para desarrollar el proyecto existen dos plataformas, Code Composer Studio y IDE Energia. El interfaz de Energia es más simple pero cuenta con todo lo necesario para programar. En un principio se empezó a trabajar con el de Energia pero luego viendo el potencial que tenia CCS se optó por continuar con el desarrollo del proyecto sobre esta plataforma, ya que tiene un apartado donde se pueden incorporar los proyecto elaborados con TI-RTOS bajo Energia.

2.2. Estudio de mercado

En la actualidad existen otras placas similares a la que se utiliza para el desarrollo de la aplicación. Una de ella es la placa Arduino que se utiliza tanto en el ámbito académico como en el ámbito industrial. Modelos Arduino existen varios, en la siguiente ilustración se muestra un ejemplo de uno de los modelos.

Ilustración 4 - Placa Arduino

Lo que quizás no exista, es algo similar al BoosterPack donde está todo lo necesario que necesita el usuario para interactuar con el juego. También existen en el mercado otro dispositivos WIFI similares.

Por otro lado, actualmente se pretende integrar todo en un único dispositivo, por lo que las consolas portátiles o juegos como el Simon se han exportado a dispositivos móviles o tablets.
El juego de Simon se ha desarrollado para diferentes plataformas, como pueden ser web, Android o IOS.

Los primeros juegos Simon eran máquinas simples, con pulsadores y luces, como se puede ver en la siguiente ilustración. Recientemente este juego ha evolucionado en formato físico, han creado un dispositivo táctil, no hace faltar pulsar ningún botón.

Ilustración 5 - Distintos dispositivos Simon

3. Descripción funcional

Para obtener el resultado adecuado, antes de empezar con el desarrollo es conveniente analizar con detenimiento la solución software que vamos a implementar y todas las posibles opciones con las que contamos.

3.1. Sistema Simon Says

En este apartado, vamos a ver el esquema del sistema desarrollado, así como los diferentes elementos interaccionan entre si.

3.1.1. Componentes físicos y diagramas

Los componentes físicos y sus características son:

- Launchpad MSP432P401R: Esta placa permite desarrollar aplicaciones de bajo consumo. Es una placa que se utiliza sobretodo en el ámbito de la educación. A esta placa para el desarrollo del proyecto se le ha añadido un módulo WIFI y un BoosterPack. Está placa tiene todos los elementos necesarios para nuestra aplicación, por lo que no es necesario soldar o añadir ningún elemento más. A continuación vemos en la ilustración las partes de la placa utilizadas.

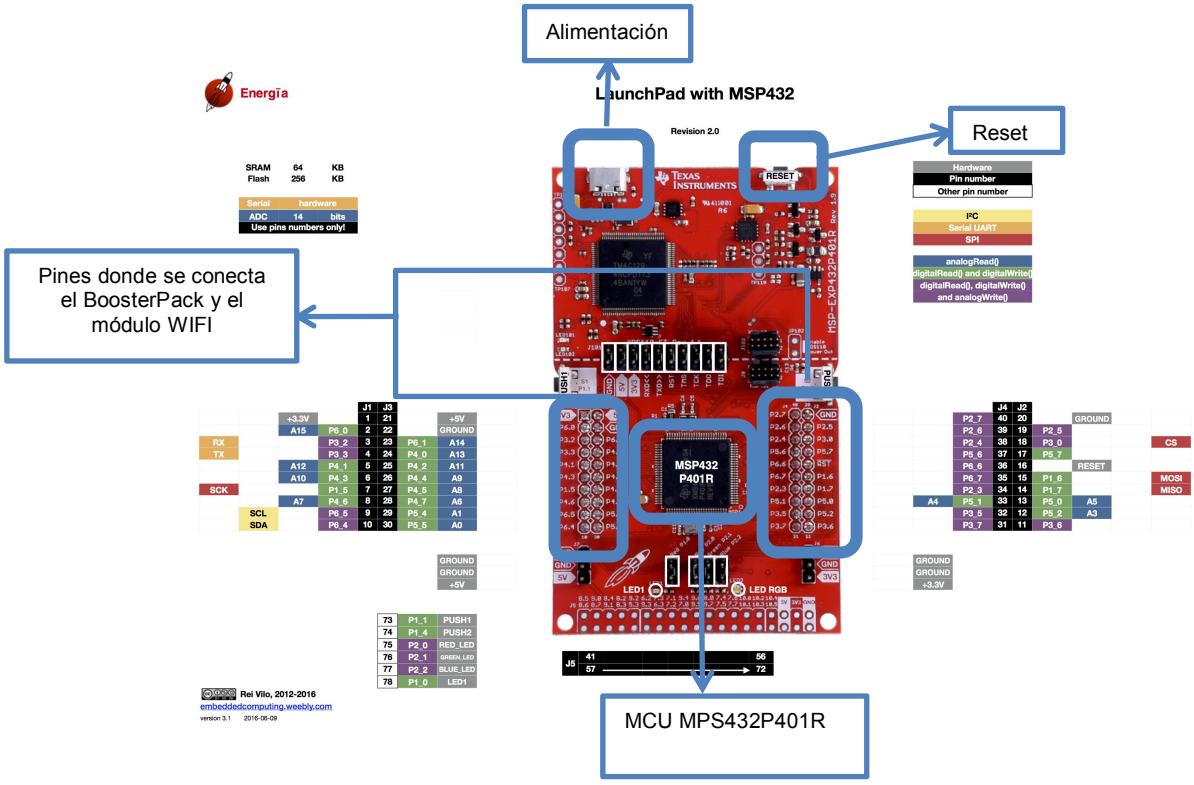


Ilustración 6 - LaunchPad

- Educational BoosterPack MKII: Este elemento es el que permite la conexión del usuario con el micro. Es un elemento que incluye dispositivos de entrada y salida, asi como un LCD, joystick,botones, LED. En la ilustración 7 se puede ver este elemento.

Ilustración 7 - BoosterPack

Los elementos del BoosterPack empleados son:

1. Joystick. El BoosterPack envía la información que obtiene de la interacción del usuario con el joystick. Es utilizado para reproducir la secuencia y para pausar el juego.

2. Pantalla LCD. El LaunchPad envía la secuencia de colores, textos y cuenta atrás, para que el usuario pueda seguir la dinámica del juego.

3. Buzzer. Se utiliza para reproducir un tono cada vez que aparece un color en la secuencia. Cuando el usuario ha perdido también reproduce una melodía al igual que cuando tiene que empezar a reproducir la secuencia.

4. Botones. Se utilizan para indicar si se desea jugar en modo multijugador.

- CC3100 BoosterPack

Es el módulo que permite la conexión WIFI. En la aplicación se utiliza para las partidas multijugador. Es el encargado de conectar de conectarse a la red y poder enviar peticiones al servidor. El módulo es el mostrado en la siguiente ilustración:

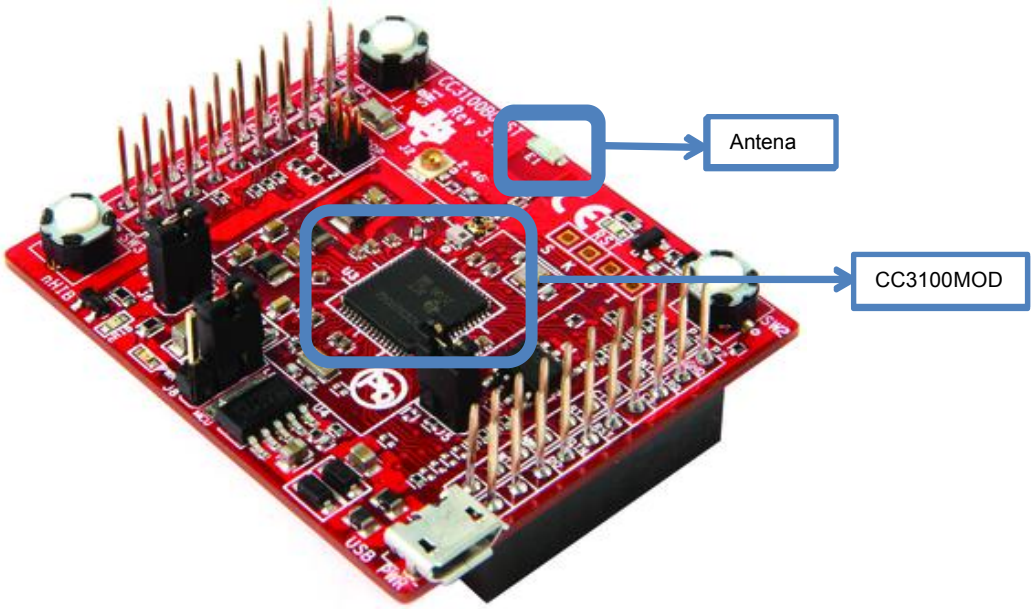


Ilustración 8 - CC3100 BoosterPack

- Servidor corriendo en un PC

  Es el encargado de recibir las peticiones http para el modo multijugador. También se encarga de guardar en BBDD un histórico de partidas entre jugadores.

Para poder comunicarse unos componentes con otros es necesario un software. Existen dos tipos de software:

1. Software de sistema, es el encargado de dar soporte a las aplicaciones. Está formado por los drivers, el sistema operativo y el middleware.

2. Software de aplicación, es el software que define para qué va a servir el sistema embebido. Es el punto de interacción con el usuario.

La arquitectura de nuestra aplicación está compuesta por tres capas.

Ilustración 9 - Capas del sistema

De manera más específica, la aplicación queda como vemos en la siguiente ilustración:

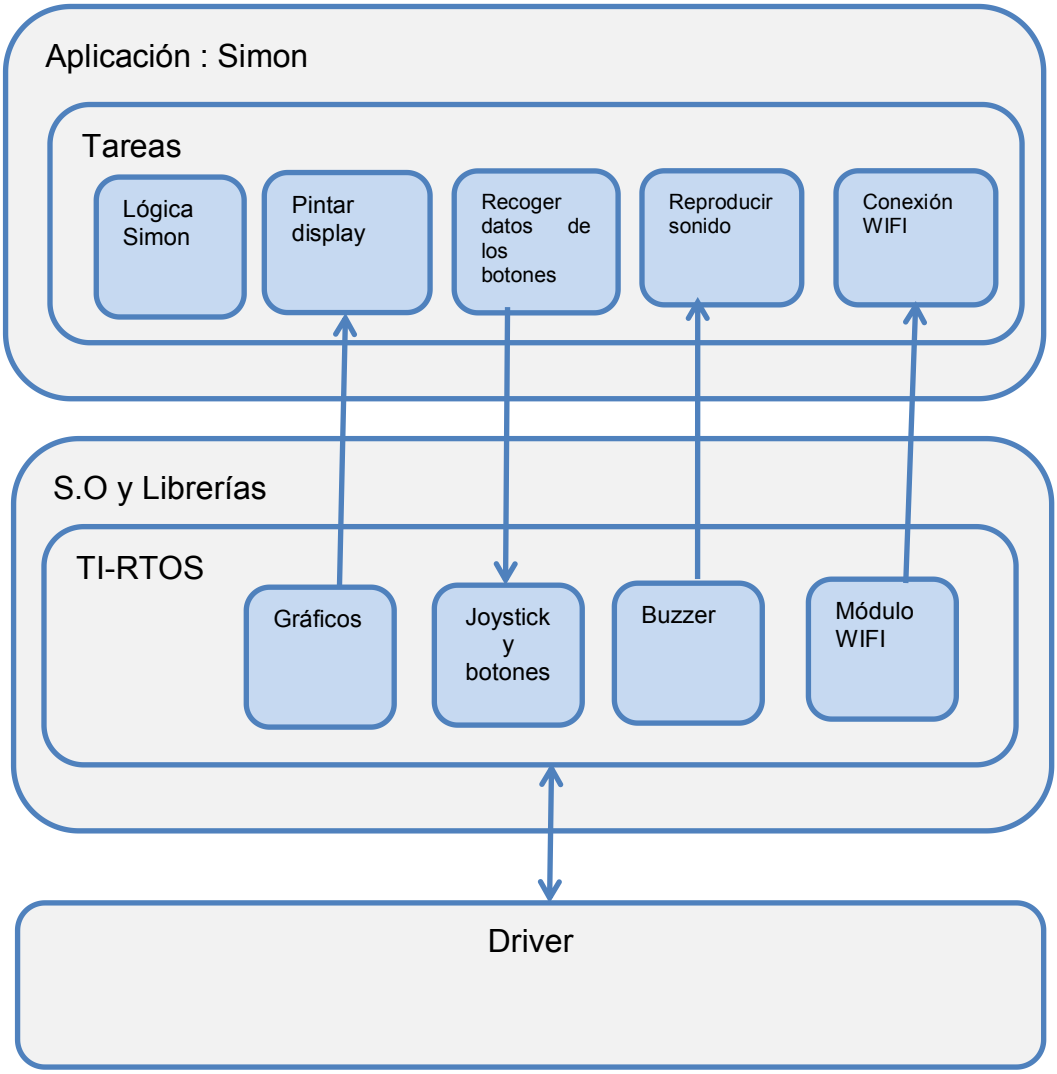


Ilustración 10 - Arquitectura Aplicación

a. Drivers. Son los encargados de gestionar el hardware del sistema embebido. Nos permite acceder al hardware ya sea para enviar o recibir datos. Este sistema embebido cuenta con un módulo de Entrada y Salida de Propósito General. Este módulo va a permitir controlar todos los elementos de la aplicación BoosterPack y módulo Wifi.

b. SO y librerías. En nuestro caso vamos a diferenciar dos tipos de SO, el que se va a utilizar para desarrollar la aplicación del juego y el que se va a emplear para el desarrollo del servidor

para la parte multijugador. Ambos sistemas operativos tienen el mismo propósito, es abstraer el software del hardware de manera que sean independiente uno de otro y se puedan realizar de forma mas simple el desarrollo de aplicaciones. Es también tarea del sistema operativo gestionar la comunicación entre hardware y software para que se opere de manera eficiente.

Las principales librerías empleadas son:

1. Librería encargada de los gráficos. Nos permite acceder al display LCD, así se consigue plasmar toda la la interfaz del juego.

2. Librería para Joystick y botones. Es la que nos permite recoger todas las acciones que se realizan con estos elementos.

3. Librería Buzzer. Es la encargada de que se pueda acceder al mismo y que se reproduzcan los tonos que se necesitan.

4. Librería WIFI. Nos permite conectar la aplicación vía WIFI para la partida multijugador.

Por otro lado tenemos las librerías empleadas para el servidor.

1.Librería para peticiones HTTP. Nos permite recibir, procesar y dar respuesta a las peticiones que envían los jugadores cuando están conectados.

2.Librería BBDD. Se encarga de conectar el servidor a una base de datos y poder así almacenar las puntuaciones y partidas en modo multijugador.

c. Aplicación. Es la última capa del sistema, la que interactúa con el usuario. Ésta proporciona toda la lógica que queremos desarrollar para acceder al hardware del dispositivo.

Las tareas de la aplicación son:

- Lógica Simon Says

- Pintar display

- Recoger datos de botones y joystick

- Almacenar puntuación si procedo

- Realización de peticiones http al servidor para juego online.

- Reproducir sonido.

La ejecución de todo juego se realiza de manera cíclica:

- Input: Es donde se recoge la operativa del usuario, en nuestro caso sería la dirección que indique el usuario con el joystick y la pulsación de botones.

-Update: A partir de los datos que se han recogido de la interacción con el jugador, el juego actualiza su estado, continúa la lógica del juego, por ejemplo si el usuario ha pulsado pausa, se pausa el juego, si ha introducido la secuencia correcta, se ejecuta la siguiente secuencia o si se ha pulsado que se quiere conectar vía online, se intenta la conexión al servidor.

-Render: Se genera la respuesta a la acción del usuario en pantalla, por ejemplo si está moviendo el joystick, aparece en pantalla el color de la posición que ha movido o si ha pulsado pausa se le indica que la aplicación está pausada.

Al ser una ejecución cíclica, una vez que el render termina se vuelve al input.

3.1.2. Requisitos funcionales

Los requisitos funcionales son de los que constara el proyecto.A partir de estos requisitos se elaboraran los casos de uso que contendrá todo lo necesario para alcanzar el objetivo del desarrollo del juego.

Requisito funcional 1	**Introducción al juego**
Descripción	El sistema cuenta con una vista inicial para que el actor pueda saber cómo jugar.

Tabla 1 - Requisito funcional 1

Requisito funcional 2	Vista del juego
Descripción	Esta constará de la secuencia que el usuario tiene que reproducir y distintos mensajes que le indican cuando es el turno y el tiempo que le queda para completar la secuencia. Si se escoge el modo jugada online, también indicará al usuario todo el proceso de conexión online, se mostrarán mensajes cuando se está conectando al servidor que generará la pantalla y cuando se conecte contra quien va a jugar.

Tabla 2 - Requisito funcional 2

Requisito funcional 3	Puntuación
Descripción	Muestra y en caso que proceda se almacena la puntuación obtenida por el jugador.

Tabla 3 - Requisito funcional 3

Requisito funcional 4	Pausa
Descripción	El sistema proporciona la opción de pausar el juego una vez que la secuencia se ha reproducido y antes de comenzar a imitar la secuencia reproducida.

Tabla 4 - Requisito funcional 4

Requisito funcional 5	Jugada multijugador
Descripción	El sistema proporciona la opción de conexión WIFI para poder jugar contra otro usuario conectado. Una vez que el sistema se conecta y existen dos usuarios conectados, se generan las partidas.

Tabla 5 - Requisito funcional 5

3.1.3. Requisitos no funcionales

Los requisitos no funcionales son los que no imponen restricciones en el diseño o la implementación del proyecto.

Estos requisitos no tienen que ver directamente con el comportamiento funcional del sistema. Son requisitos como la eficiencia, precio, interfaz de usuario.

Requisito no funcional 1	Interfaz amigable
Descripción	La interfaz tiene que ser simple e intuitiva para que sea atractiva para el usuario.

Tabla 6 - Requisito no funcional 1

Requisito no funcional 2	Sistema embebido de bajo coste
Descripción	Para el proyecto hay que buscar el equilibrio entre el coste del sistema embebido y las posibilidades que nos ofrece.

Tabla 7 - Requisito no funcional 2

Requisito no funcional 3	Libre de licencias de pago
Descripción	Al ser un proyecto educativo, hay que desarrollar el sistema bajo licencias libres.

Tabla 8 - Requisito no funcional 3

3.1.4. Desarrollo Software

Una vez que hemos visto los diferentes requisitos funcionales y no funcionales, vamos a ver que actores existen y cómo interactúan con el sistema.

3.1.4.1. Identificación de actores

En este apartado veremos la función del actor en la aplicación. Depende de la opción que se elija existirá un solo actor o varios. Si se opta por la opción online existirán dos actores/jugadores y si es la opción simple un solo jugador/actor. Estos actores van a ser los únicos en interactuar con el sistema. Siempre al comienzo del juego se encontrará con un menú explicativo de la dinámica del mismo. Una vez pasado el menú empezará el juego en si.

3.1.4.2. Modelo de casos de usos

En el siguiente apartado, vamos a ver los principales casos de uso del sistema y una breve descripción de los mismos.

Los casos de usos son los siguiente:

1. Introducción al juego

En la primera pantalla, el jugador tendrá sólo la opción de leer cómo funciona el juego, ver la máxima puntuación acumulada hasta el momento y decidir en que modo quiere jugar. Este caso se uso se verá reflejado en la ilustración 11.

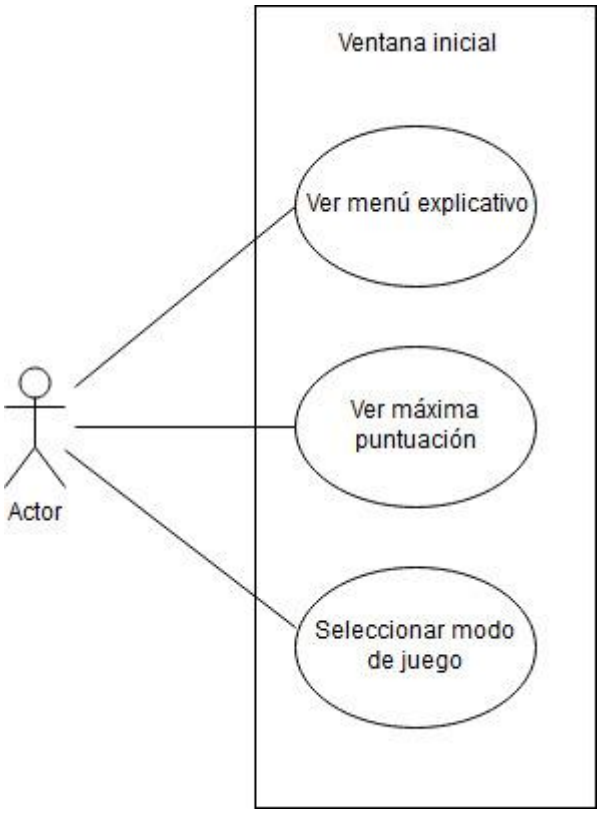


Ilustración 11 - Modelo de caso de uso,Ventana Inicial

2. Juego

Este es el principal caso de uso. En este caso de uso el jugador podrá:

- Ver la secuencia del juego que se reproduce para posteriormente imitarla él.

- Ver el tiempo que le queda para reproducir la secuencia

- Cada vez que mueva el joystick, se podrá ver en pantalla el color que corresponde a ese movimiento.

- Ver la puntuación obtenida después de reproducir correctamente la secuencia.

- Ver el nivel cuando corresponda. Cada cuatro reproducciones la secuencia aumenta en dos colores más. Empieza en una secuencia de cuatro colores.

- Si se conecta modo online, ver todo el proceso de conexión, obtención de secuencia y guardado de partida.

- Ver cuando el juego está pausado. Cuando se pulsa el centro del joystick, aparece en pantalla el texto de que la aplicación está pausada.

- Ver cuando ha perdido. Si se ha reproducido de manera incorrecta la secuencia, se muestra por pantalla el texto “GAME OVER” y se reproduce un sonido de error.

En la siguiente ilustración se puede ver este caso de uso.

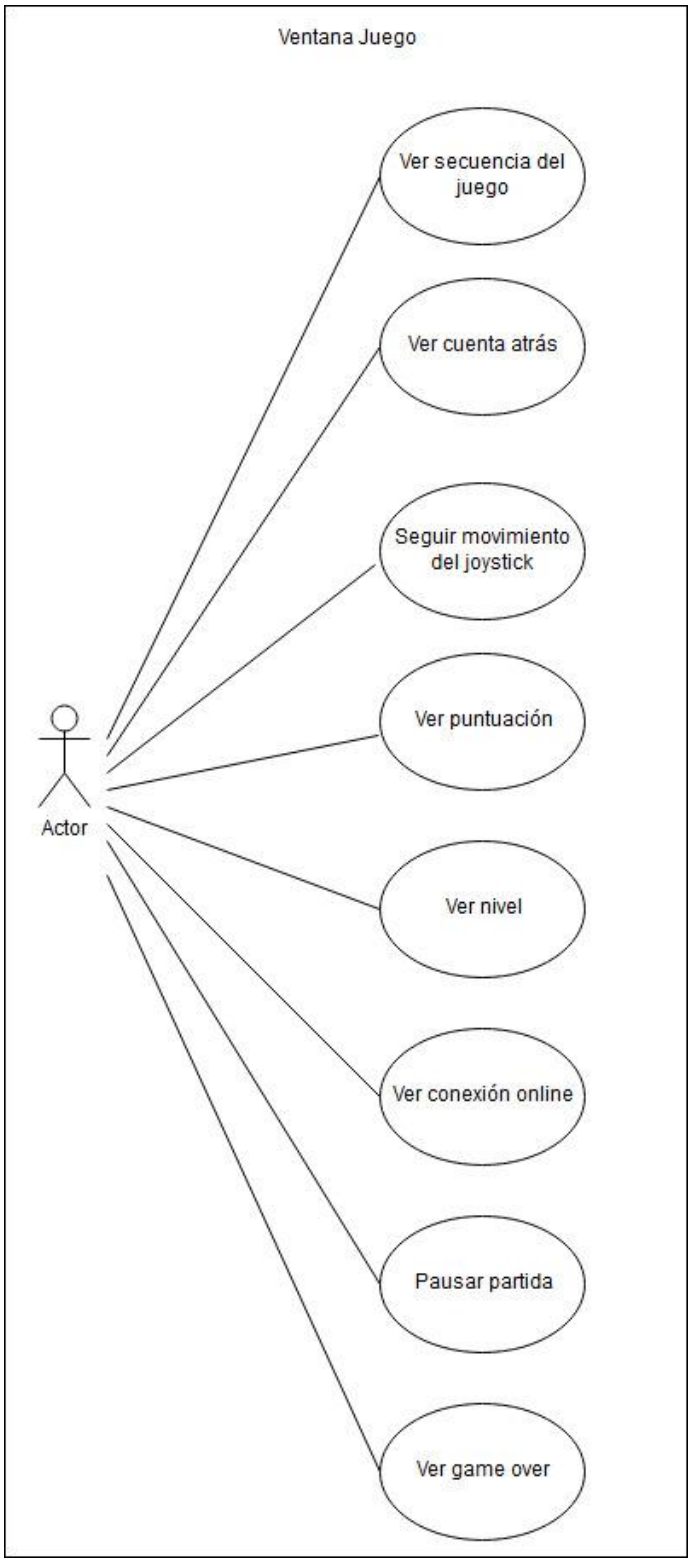


Ilustración 12- Modelo de caso de uso,Ventana juego

3. Modo online

Este caso de uso es una variante del anterior, se produce cuando el usuario ha pulsado la opción de jugar online contra otro jugador. Este caso de uso incluye las siguientes acciones:

- Ver la conexión WIFI, se indica por pantalla en todo momento el estado de la conexión WIFI
- Ver conexión servidor, se puede ver por pantalla cuando está buscando oponente y conectándose al servidor para obtener la secuencia.
- Al igual que el modo de un solo jugador se puede ver la generación de secuencia. Este modo también incluye el seguimiento del juego.
- Seguimiento guardado de puntuación. En el servidor se sigue un histórico de partidas, se puede ver por la pantalla que está guardando la puntuación.
- Ver si el oponente se ha desconectado o no hay ninguno conectado. Muestra por pantalla pasado un tiempo si no se ha encontrado oponente. También se muestra por pantalla si el oponente se ha desconectado del juego.
- Fallos de conexión. Si no se puede conectar al WIFI o al servidor se muestra por pantalla.
- Ver puntuación máxima. Se recoge del histórico del servidor la puntuación máxima obtenida por el jugador que se ha conectado.

Este caso de uso queda reflejado en la siguiente ilustración:

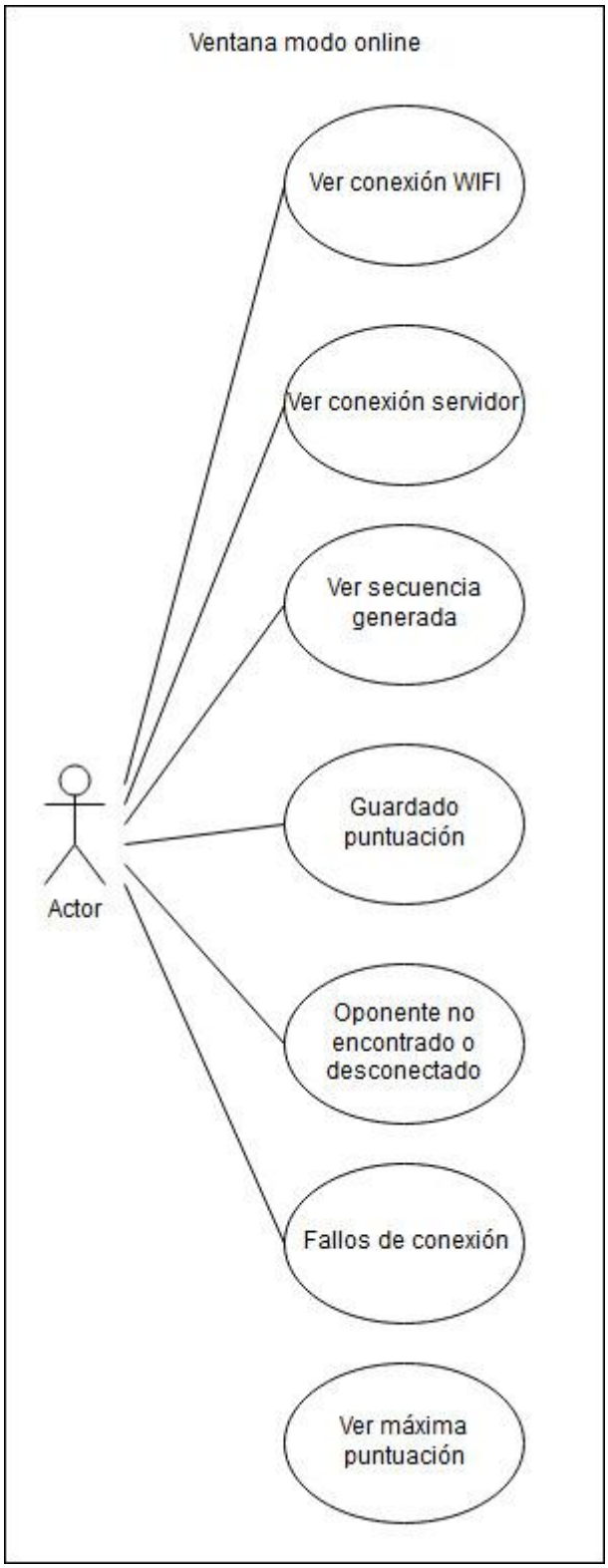


Ilustración 13- Modelo caso de uso, ventana modo online

3.1.4.3. Diagrama de actividad

Trabajar con Energia y TI-RTOS, hace que el desarrollo sea simple, se crea una estructura que siguen todos los sistemas embebidos. En la primera parte se declarar librerías, variables y métodos y una segunda parte donde existen dos funciones principales que son setup() y loop(), como se puede ver en la ilustración 14.

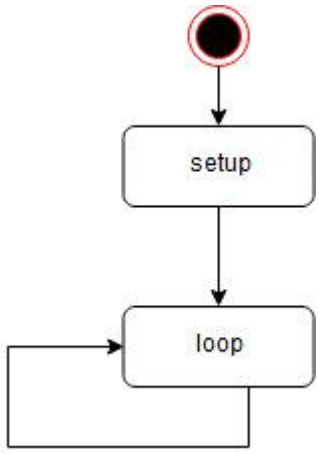


Ilustración 14 - Diagrama secuencia Energia

El diagrama de secuencia de la función setup() es el mostrado a continuación. En él se pueden ver las funciones que se ejecutan en esta parte:

- Se muestra el menú explicativo de cómo jugar.
- Se le da la opción al usuario de esperar 15 segundos para jugar en modo de un solo jugador o si pulsa el botón B se conectará online para modo multijugador.

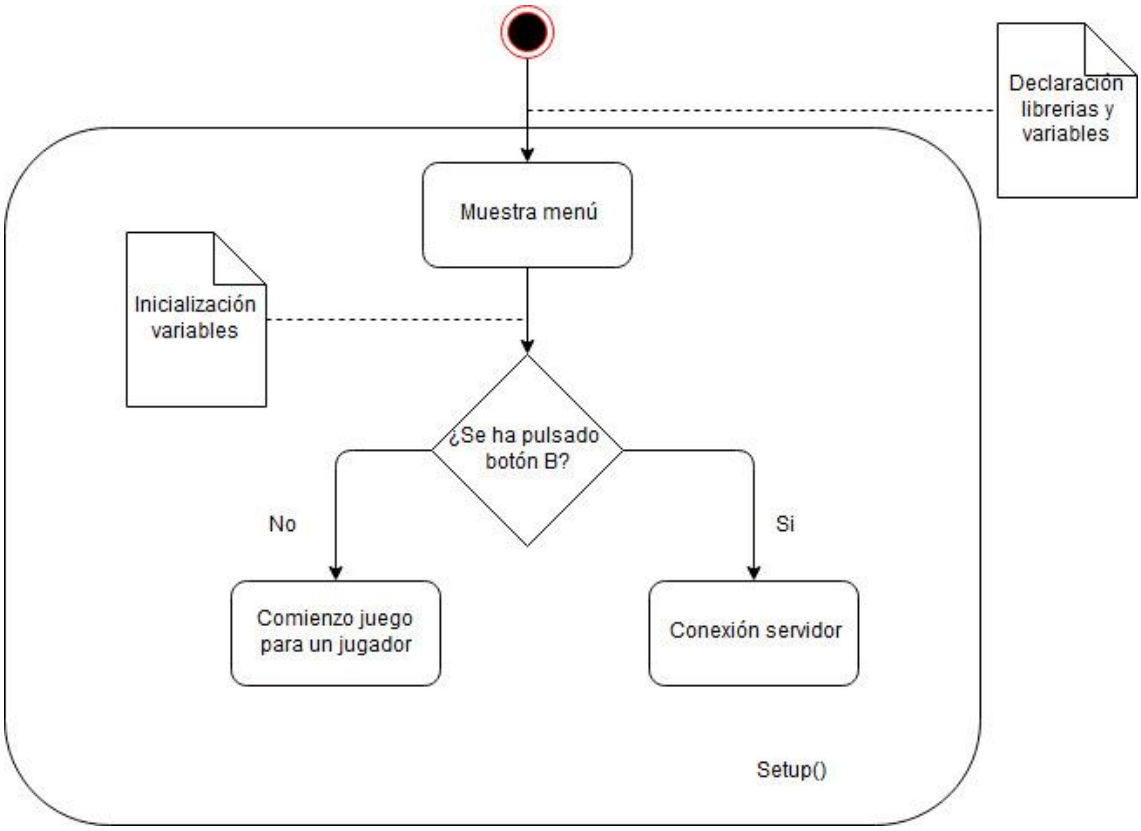


Ilustración 15 - Diagrama secuencia setup()

La secuencia del juego comienza comprobando si el jugador ha decidido jugar online o en local. Si ha decido jugar online se conecta al servidor que será el encargado de generar la secuencia. Si ha decido jugar en local se reproduce la secuencia.

Si se juega online, hay que esperar que otro usuario se conecte. Si pasado dos minutos no se ha conectado ningún usuario, se le mostrará al usuario actual la pantalla de inicio para que juegue de manera local.

Una vez que se ha obtenido la secuencia para jugar, sea de un modo o de otro. El usuario contará con un minuto para reproducir correctamente la secuencia. Si la ejecución es correcta, se guarda la puntuación y se genera una nueva secuencia. Cuando se han reproducido de manera correcta cuatro secuencias el nivel de dificultad aumenta. Las secuencias tendrán más de cuatro combinaciones una vez que el nivel aumente. Las secuencias a reproducir las realizará el usuario a través del joystick. En caso de que falle en la reproducción de la secuencia, aparecerá de nuevo la pantalla de inicio y el usuario comenzará de nuevo.

Durante la ejecución del juego si el usuario ha pulsado la pausa,se corresponde con pulsar en el centro el joystick, se comprobará que se ha producido esa interrupción. Si se ha pulsado antes de que el usuario comenzara a reproducir la secuencia, la pausa se realizará, sino

advertirá al usuario que ya ha comenzado a jugar y que no es posible la pausa hasta que reproduzca la secuencia actual.

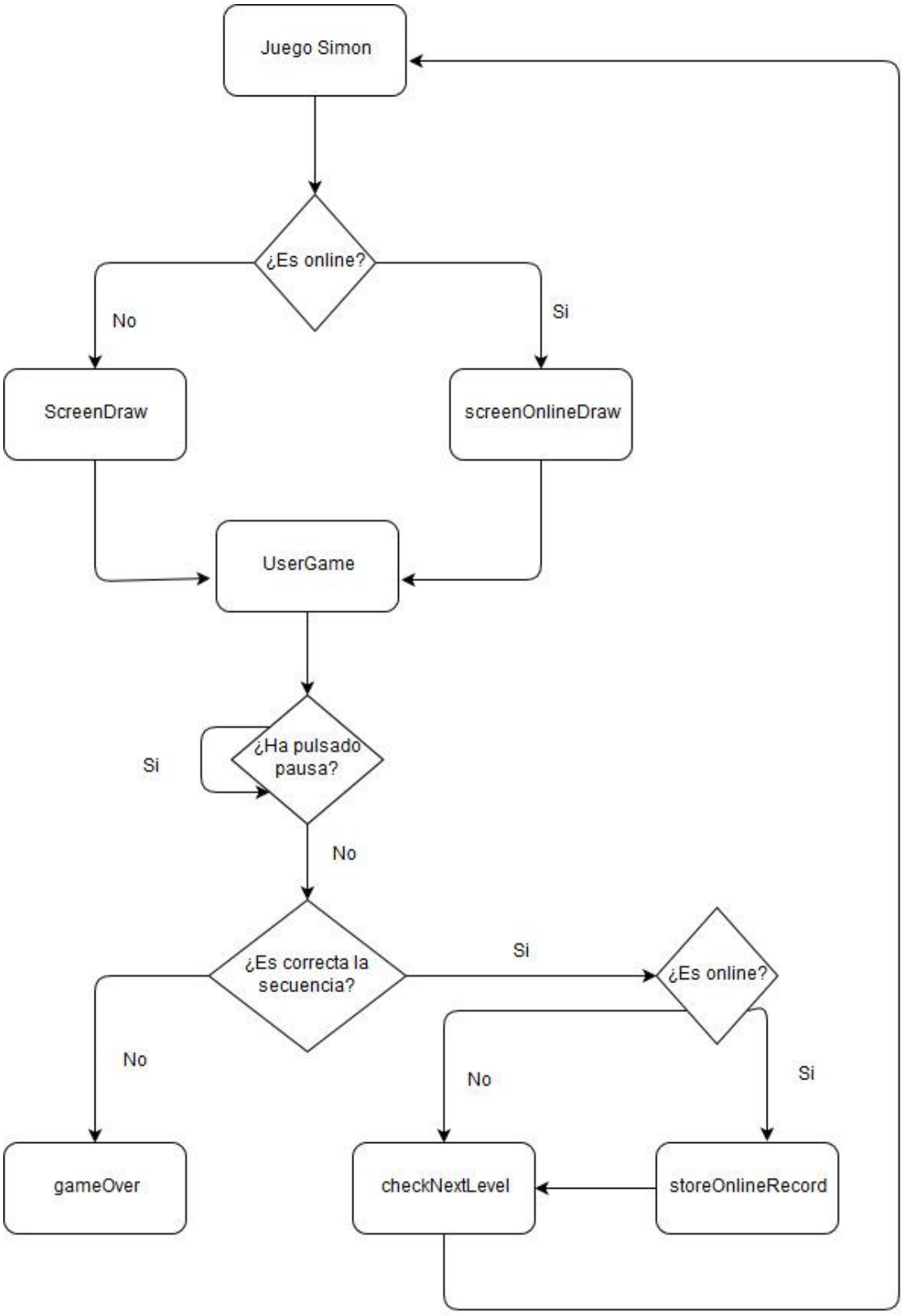
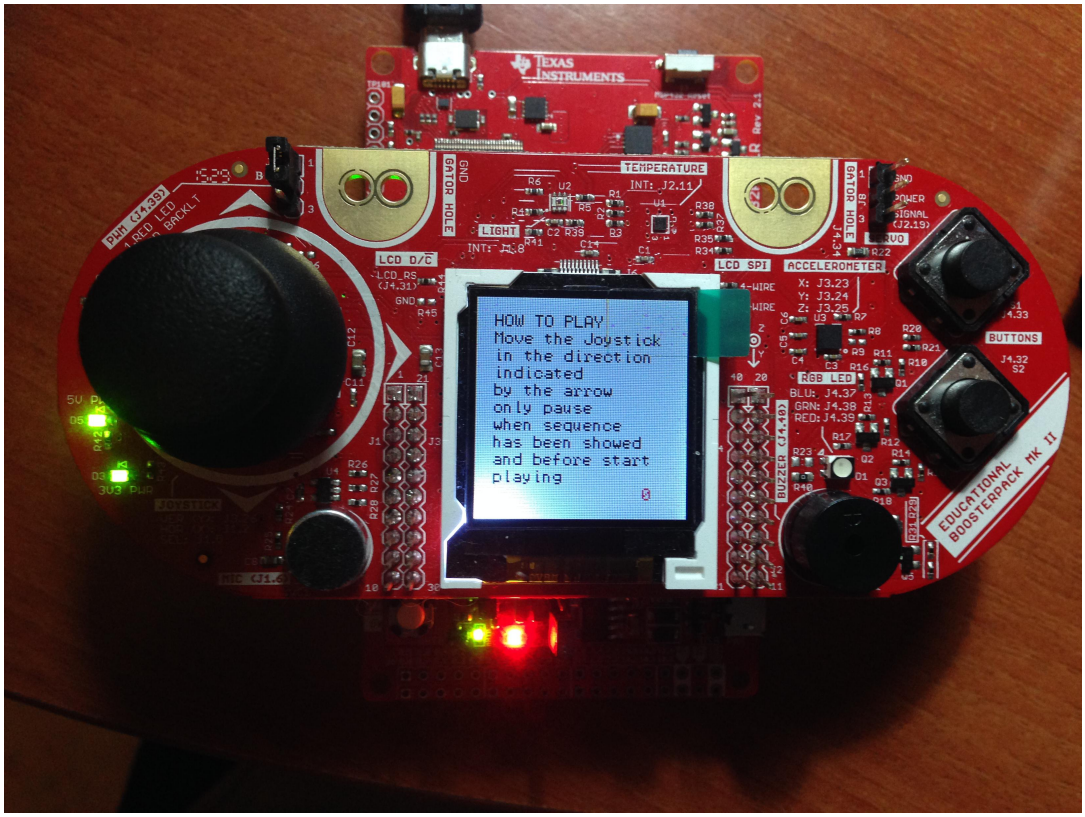


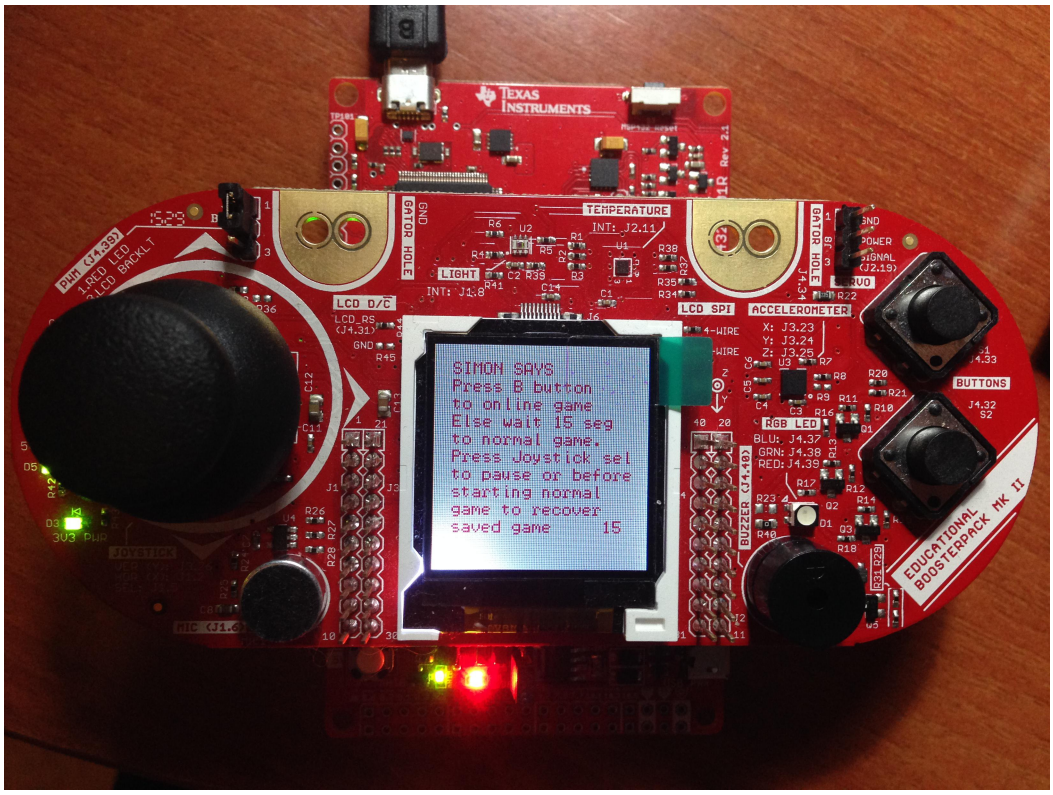
Ilustración 16 - Diagrama secuencia loop()

3.1.4.4 Interfaz de usuario

En este proyecto la interfaz de usuario es importante ya que se trata de un juego.Ésta es la mostrada a continuación.

La primera pantalla que ve el jugador es donde explica cómo jugar y la máxima puntuación obtenida.





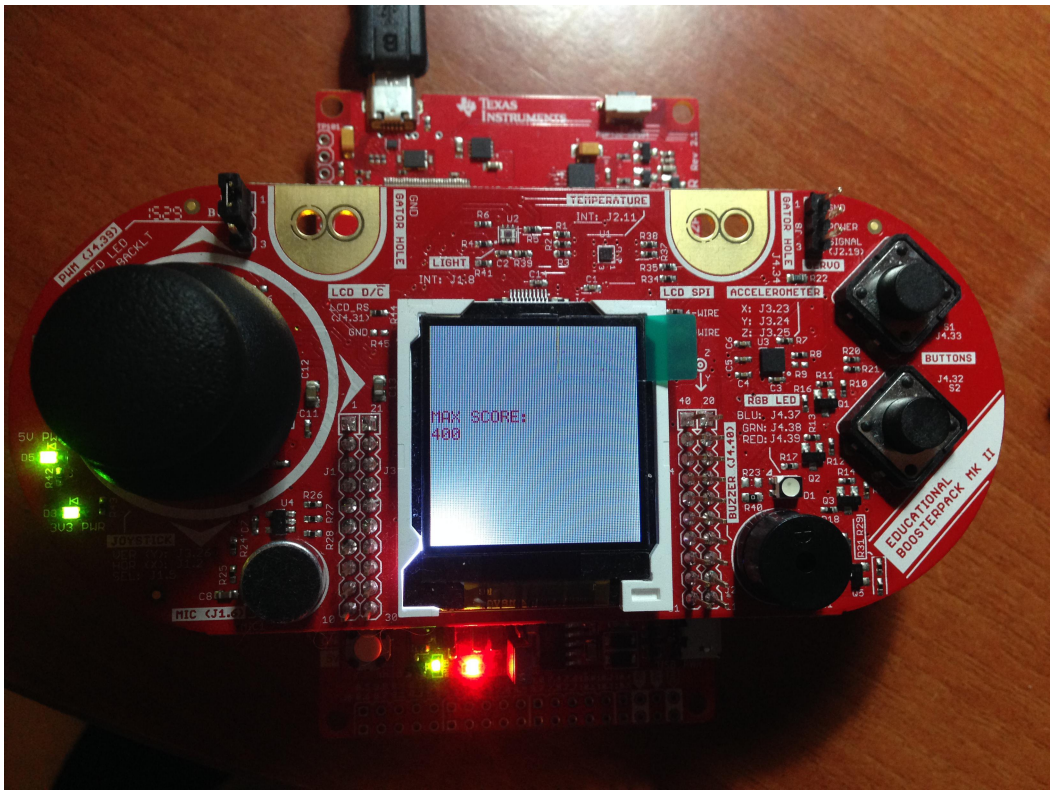
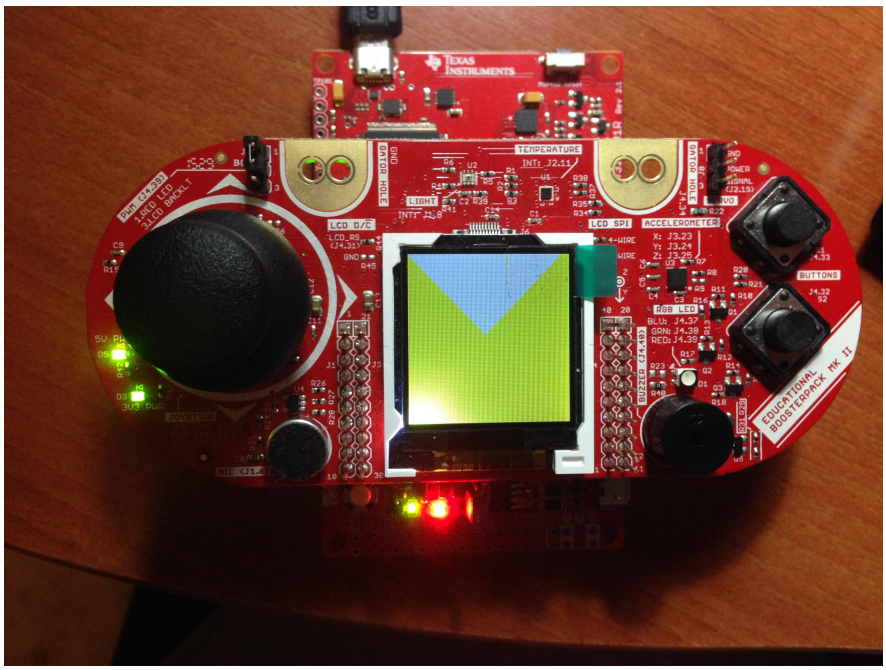
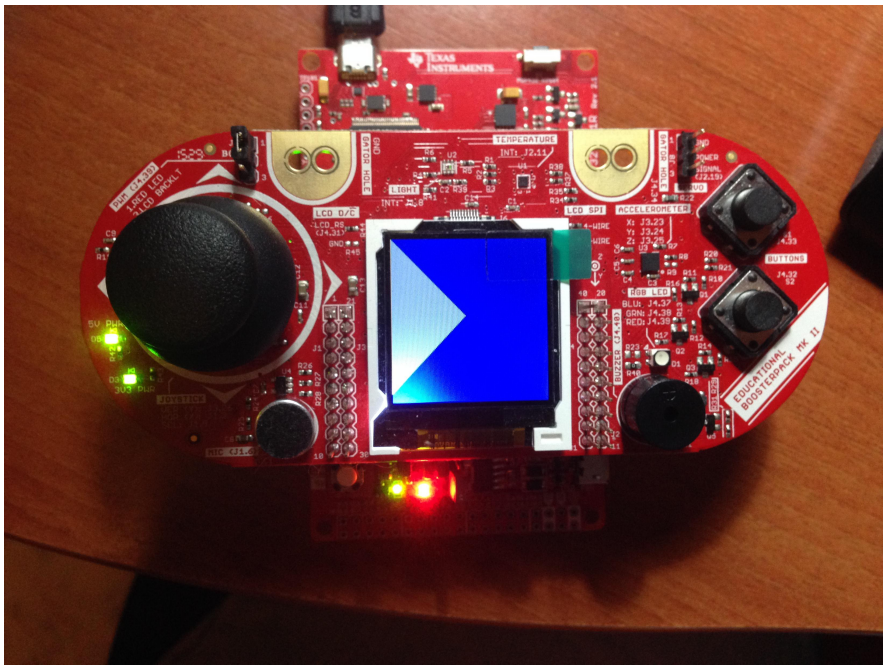


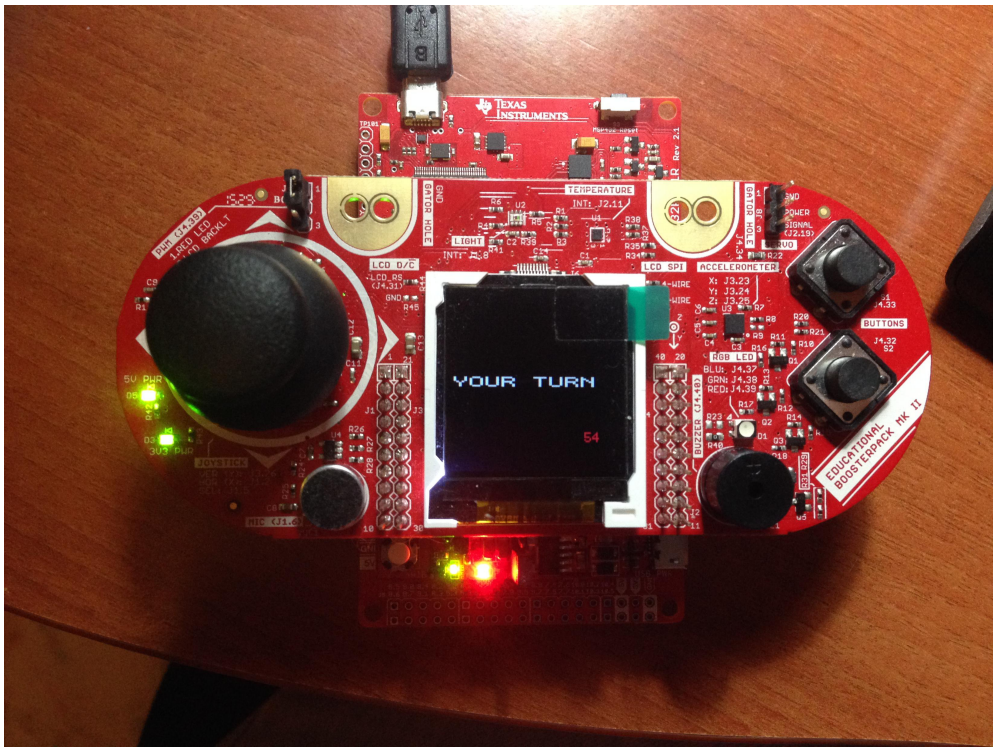
Ilustración 17- Pantallas iniciales

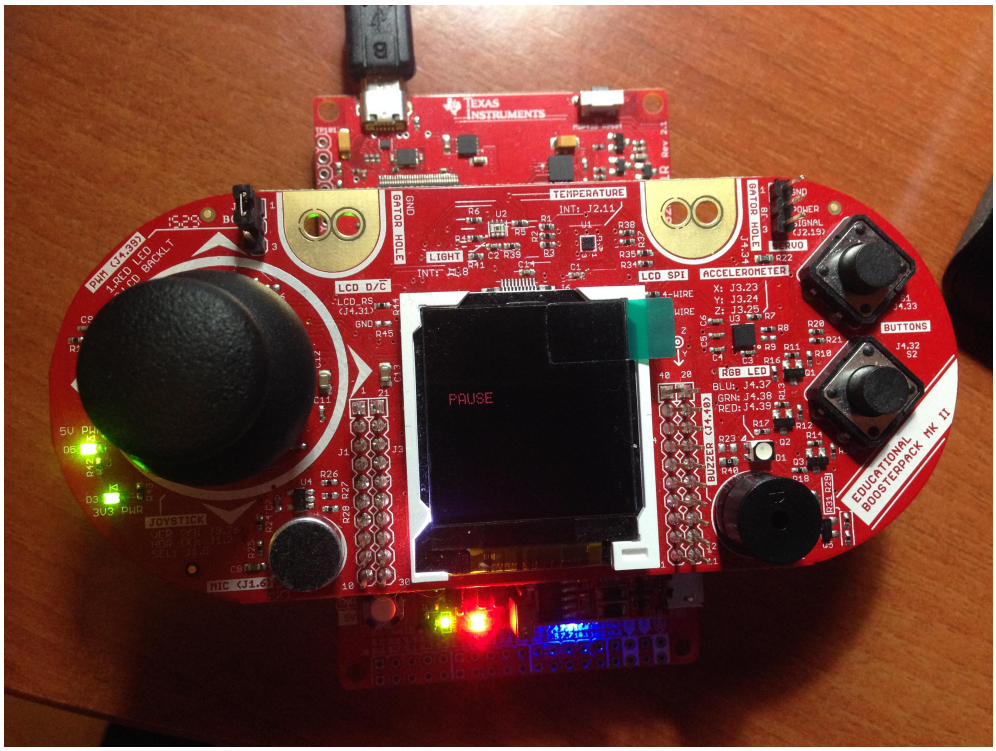
Una ver que se ha esperado los 15 segundos para jugar normal, la pantalla presenta la secuencia a reproducir. Cuando termina le indica al usuario que es su turno y una cuenta atrás.

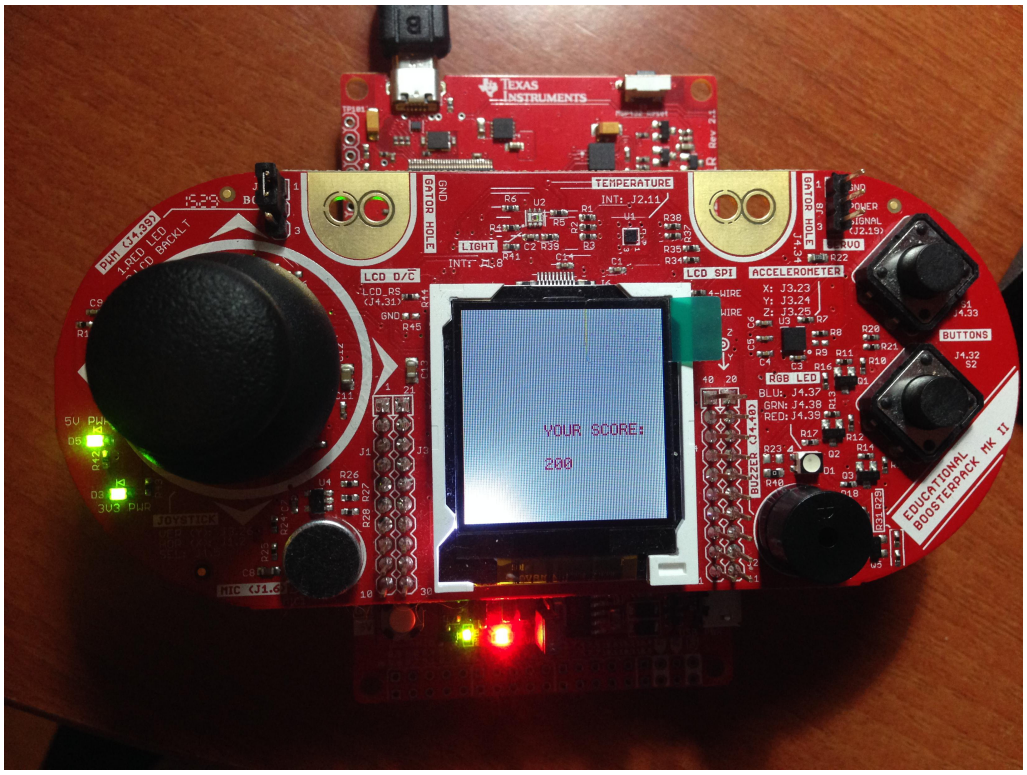
Si el usuario reproduce la secuencia de manera correcta, se presenta por pantalla la puntuación, si no es así se indica que ha fallado y se vuelve al menú inicial. Si se pulsa pausar también se indica en pantalla.






YOUR TURN
54


PAUSE



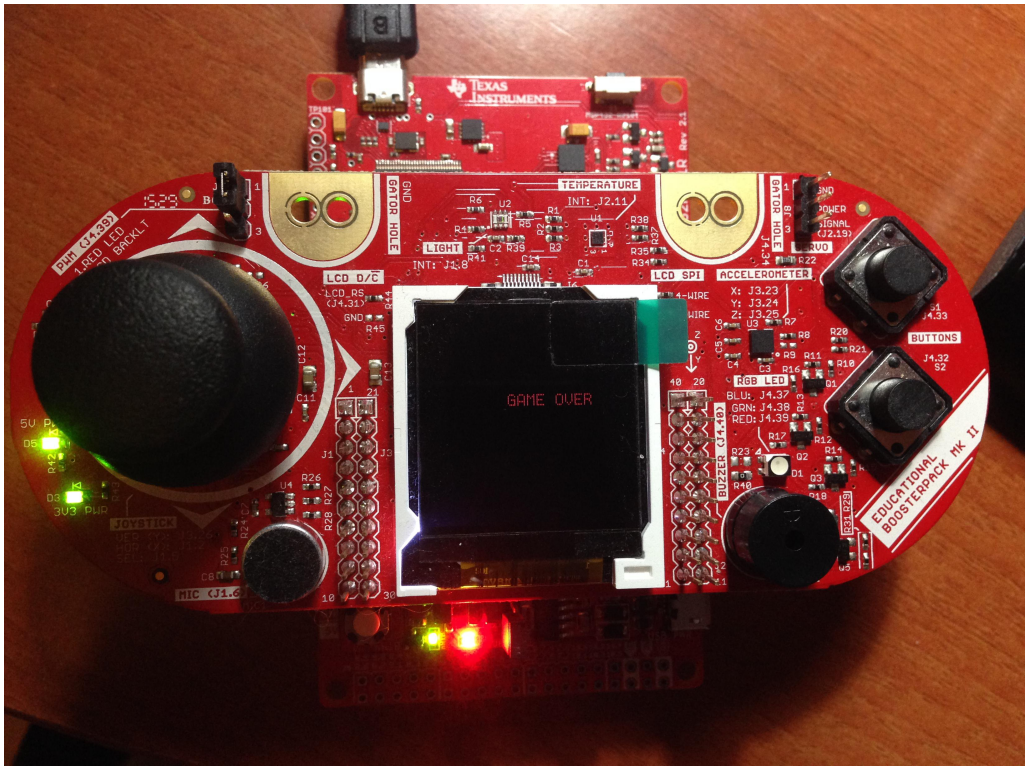
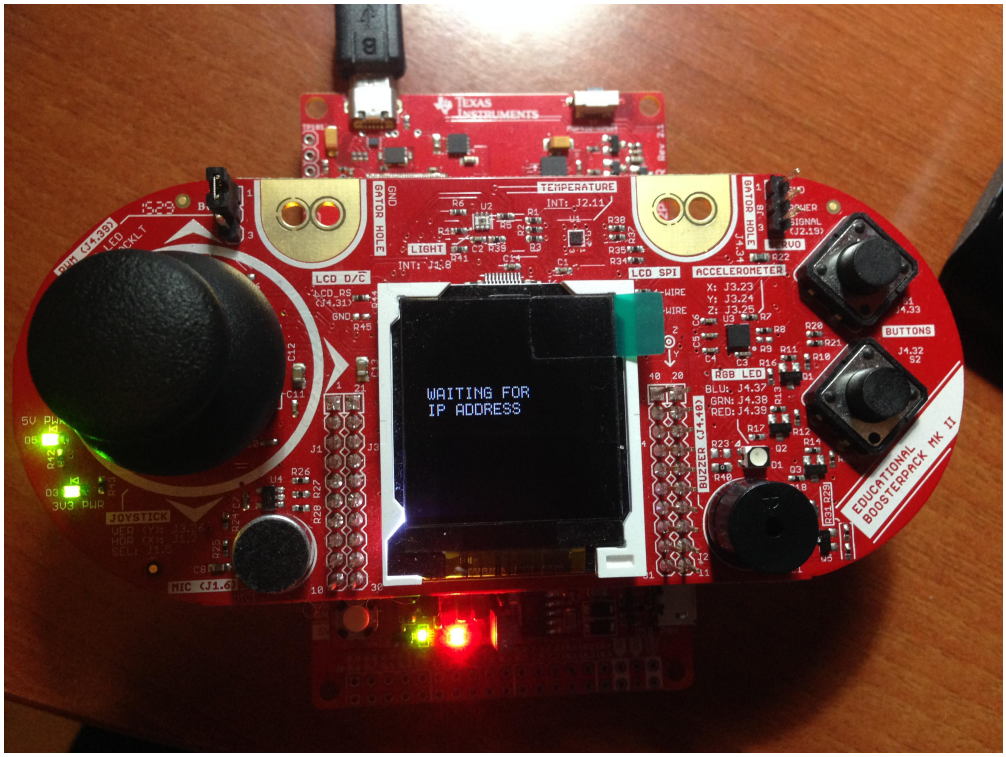


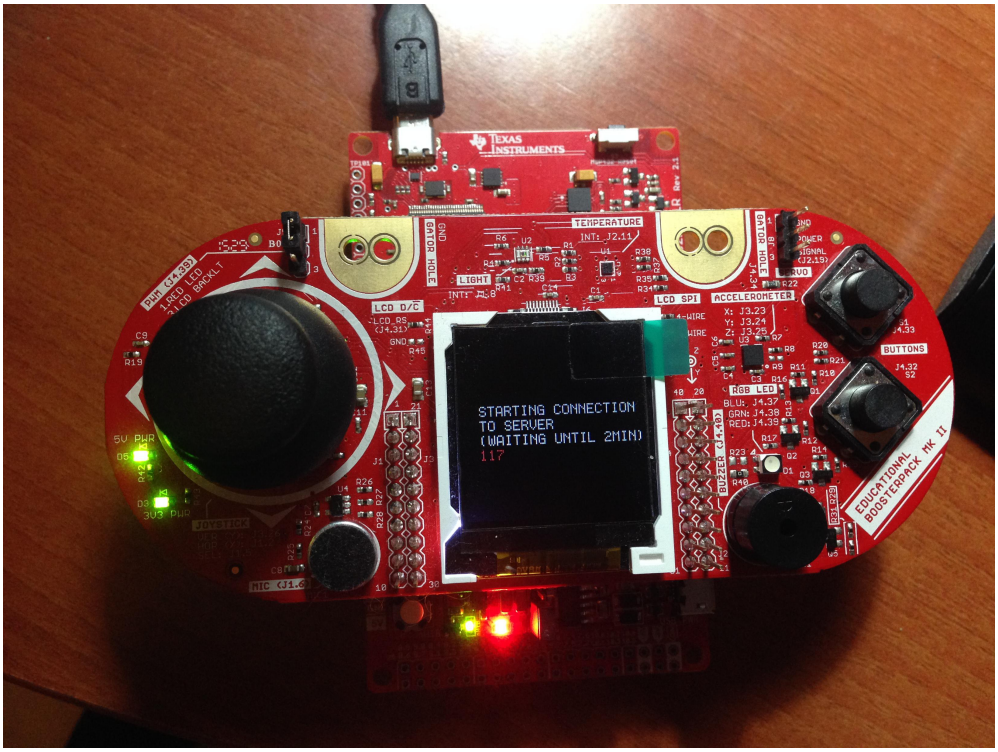
Ilustración 18- Desarrollo del juego

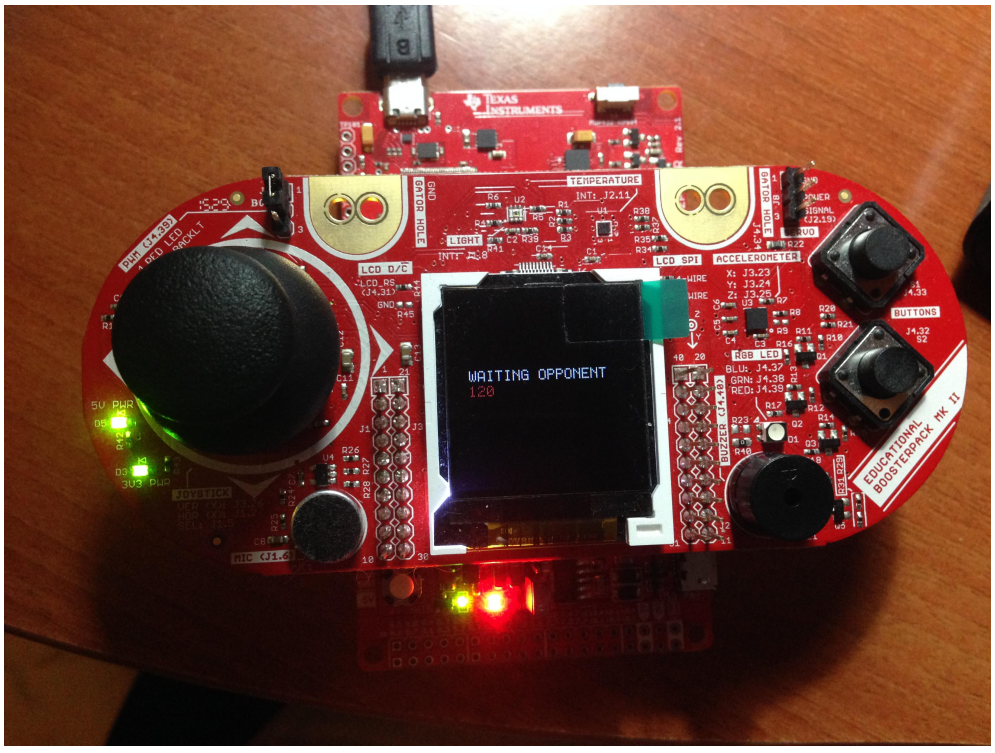
El otro modo de jugar es online, durante el proceso de conexión a WIFI y de conexión a servidor en la pantalla se va mostrando la evolución. Se puede ver en las siguientes ilustraciones.

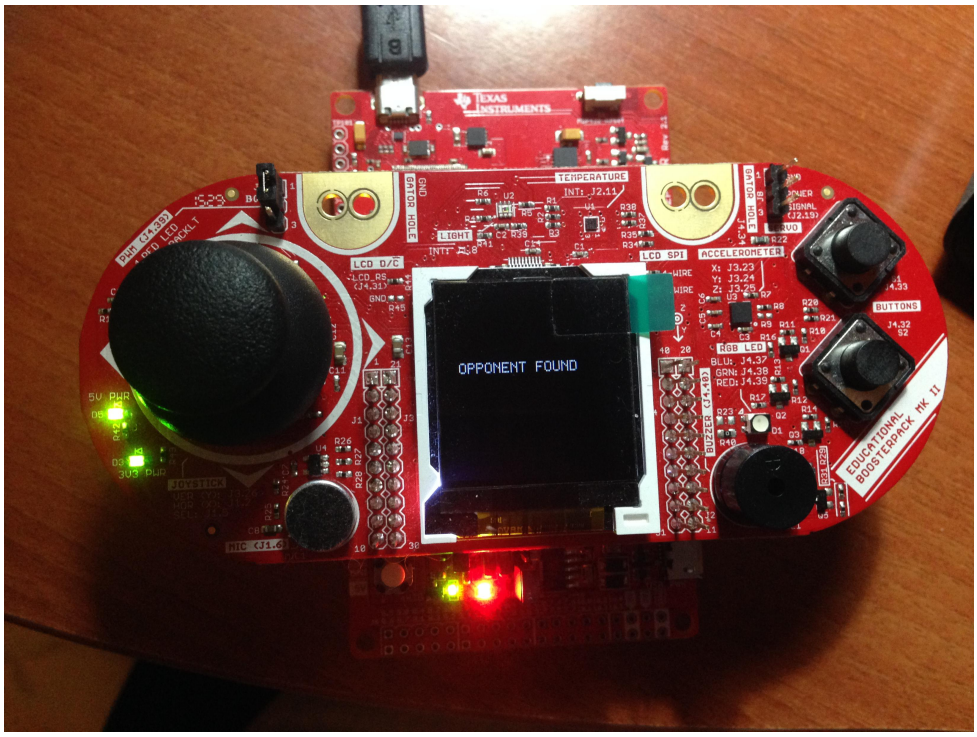
Una vez que el usuario ha reproducido la secuencia correctamente se almacena en servidor el histórico de la partida. Tanto si se almacena correctamente la puntuación como si no,se puede ver en pantalla el resultado de la operación de guardado.

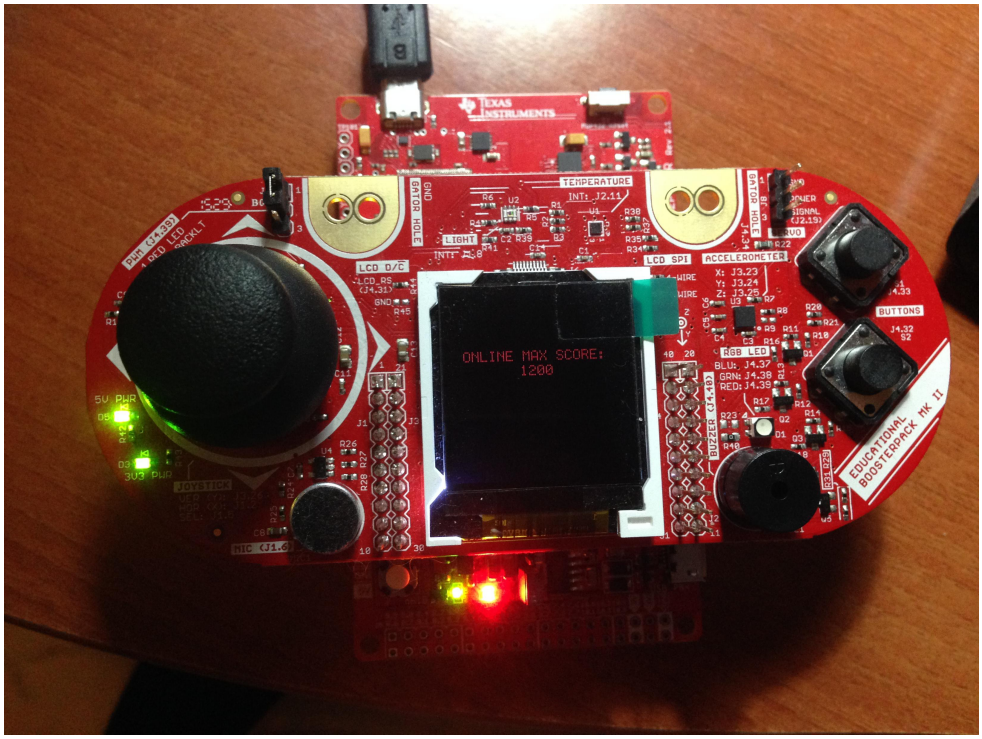
Cuando en modo multijugador se conecta al servidor y no encuentra oponente, también se informa por pantalla que no se ha encontrado con quien jugar y se vuelve al menú inicial.




STARTING CONNECTION
TO SERVER
(WAITING UNTIL 2MIN)
117


WAITING OPPONENT
120


OPPONENT FOUND


ONLINE MAX SCORE:
1200

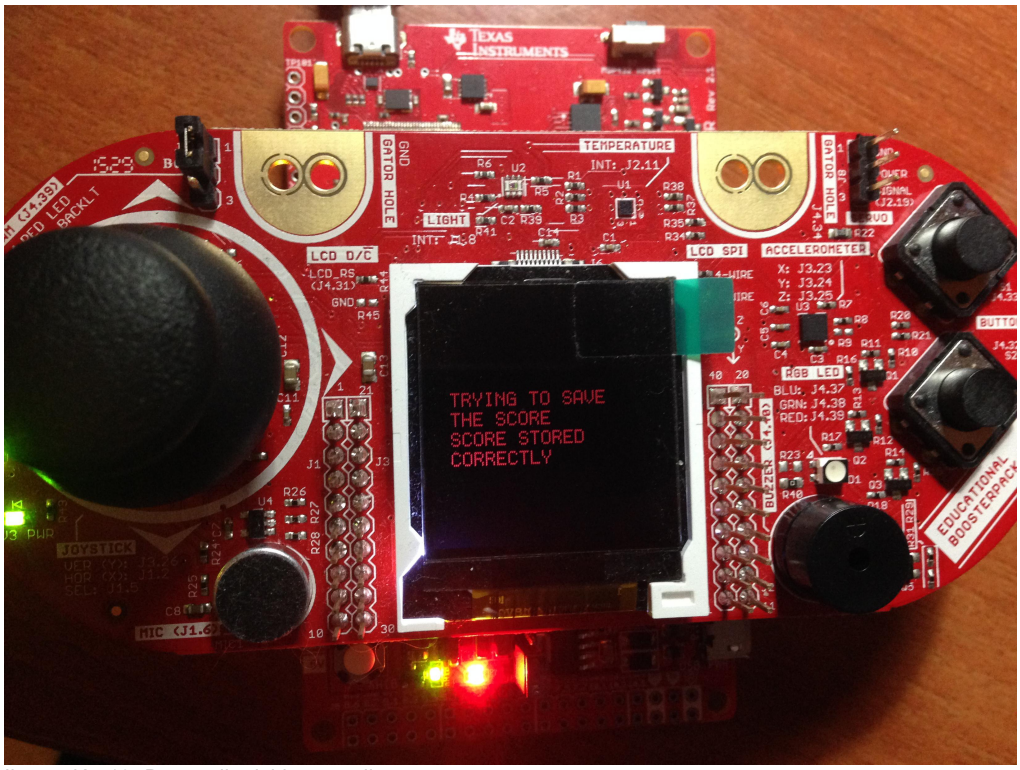


Ilustración 19- Desarrollo del juego online

4. Descripción detallada

Este apartado incluye las especificaciones técnicas del hardware empleado.

1. Launchpad MSP432P401R :

o MCU MSP432P401R de bajo consumo y alto rendimiento, que se compone por:

▪ Procesador a 48MHz 32-bit ARM Cortex M4F con unidad de punto flotante y aceleración DSP.

▪ Consumo: 80uA/MHz active and 660nA RTC standby operation

▪ Digital: Advanced Encryption Standard (AES256) Accelerator, CRC, DMA, HW MPY32

▪ Memoria: 256KB Flash, 64KB RAM

▪ Timers: 4 x16-bit, and 2 x 32-bit

▪ Communication: Up to 4 I2C, 8 SPI, 4 UART

o 40 pin BoosterPack Connector, and support for 20 pin BoosterPacks

o Onboard XDS-110ET emulator featuring EnergyTrace+ Technology

o 2 buttons and 2 LEDs for User Interaction

o Back-channel UART via USB to PC

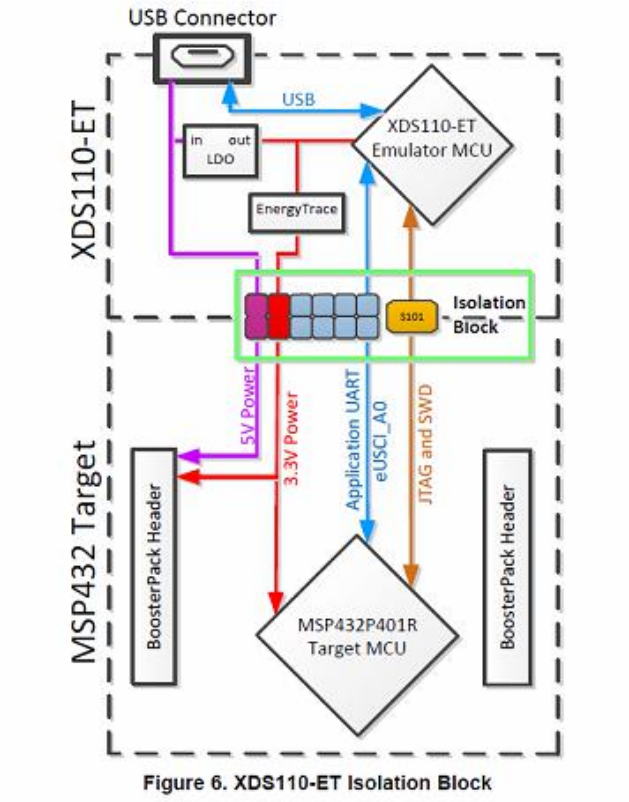


Figure 6. XDS110-ET Isolation Block

Ilustración 20 - Diagrama LaunchPad

2. Las características técnicas del Educational BoosterPack MKII:

- TI OPT3001 Light Sensor
- TI TMP006 Temperature Sensor
- Servo Motor Connector
- 3-Axis Accelerometer
- User Push Buttons
- RGB Multi-color LED
- Buzzer
- 40-pin Stackable BoosterPack Connector
- Color TFT LCD Display
- Microphone
- 2-Axis Joystick with Pushbutton

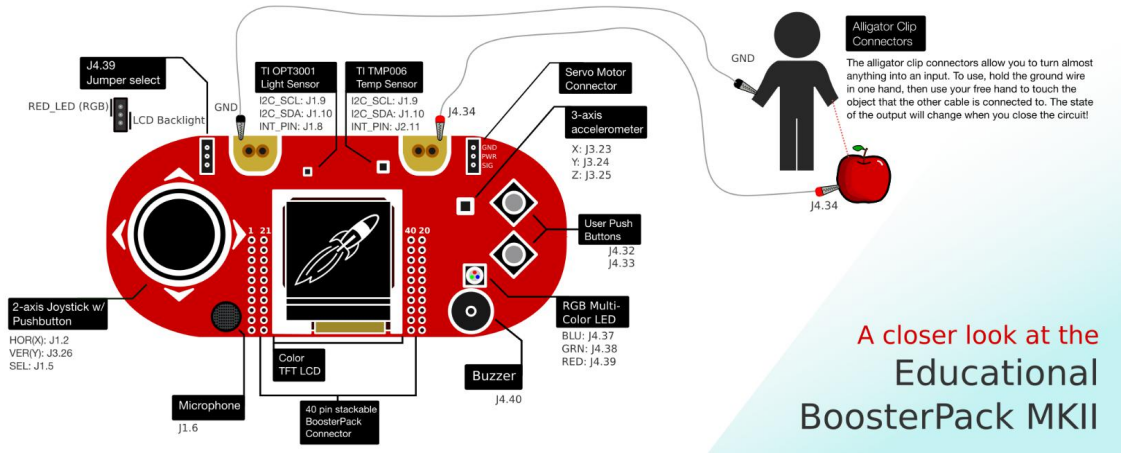


Ilustración 21 - Diagrama BoosterPack

3. Las características técnicas de CC3100 BoosterPack son:

• Wi-Fi CERTIFIED™ Chip – Application Throughput

• Wi-Fi Network Processor Subsystem

Host Interface

• Power-Management Subsystem

• Clock Source

• Package and Operating Temperature

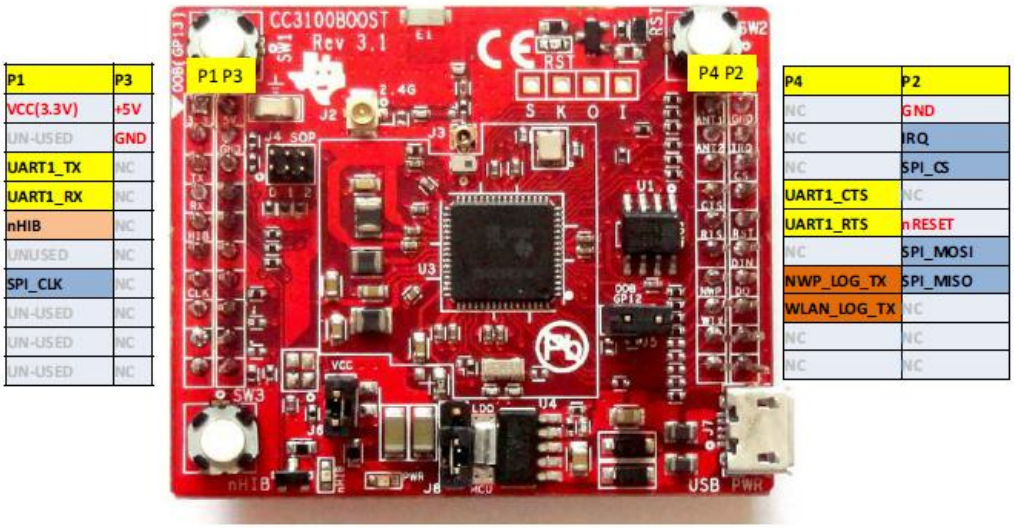


Figure 3. Signal Assignments

Table 4. Outer Row Connectors

Pin No	Signal Name	Direction	Pin No	Signal Name	Direction
P1.1	VCC (3.3 V)	IN	P2.1	GND	IN
P1.2	UNUSED	NA	P2.2	IRQ	OUT
P1.3	UART1_TX	OUT	P2.3	SPI_CS	IN
P1.4	UART1_RX	IN	P2.4	UNUSED	NA
P1.5	nHIB	IN	P2.5	nRESET	IN
P1.6	UNUSED	NA	P2.6	SPI_MOSI	IN
P1.7	SPI_CLK	IN	P2.7	SPI_MISO	OUT
P1.8	UNUSED	NA	P2.8	UNUSED	NA
P1.9	UNUSED	NA	P2.9	UNUSED	NA
P1.10	UNUSED	NA	P2.10	UNUSED	NA

Ilustración 22 - Diagrama CC3100

4. Las características técnicas para alojar el servidor para las partidas online son:

- PC Portátil: Intel Core i5 (2.4GHZ ,16GB RAM y 500GB de disco duro)

5. Viabilidad técnica

Antes de comenzar el proyecto hay que realizar un análisis técnico para saber si se pueden lograr los objetivos marcados con los medios y el tiempo que se dispone.

Para el juego de Simon los requisitos que hay que cumplir son los siguientes:

1. El sistema embebido tiene que ser capaz de procesar toda la lógica del juego.
2. Debe de existir una interfaz de usuario para que pueda interactuar con el juego. Esta interfaz gráfica sería la pantalla LCD.
3. Tiene que existir una serie de elementos con los que el usuario pueda comunicarse con el juego. Estos elementos son pulsadores, botones y joystick.
4. Debe existir conexión a internet para las partidas multijugador.
5. El sistema tiene que estar dotado de un servidor que acepte peticiones de los usuarios.
6. El sistema tiene que tener conexión y una estructura de BBDD donde almacenar la información que necesita.
7. El sistema debe contar con un sistema sonoro.

Adicionalmente es deseable que el sistema cuente con un servidor al cual se pueda acceder fuera de la red local, para poder jugar con usuarios conectados fuera de la red y una estructura de login en la que poder registrase para jugar online.

Se analizan estos requisitos para ver si con el kit de LaunchPad, BoosterPack, CC3100 BoosterPack y PC disponible se cumplen las necesidades.

- El Launchpad MSP432P401R, es un dispositivo de bajo consumo y coste, está pensado para pequeños proyectos. Ya que el juego no requiere de gran cantidad de memoria ni de operaciones muy pesadas, en principio este LaunchPad nos serviría para ejecutar el juego.
- El Educational BoosterPack MKII, contiene todo lo necesario para que el usuario pueda jugar, esto es pantalla LCD, joystick,botones, buzzer. Estos elementos son más que suficiente para el desarrollo del juego.
- CC3100 BoosterPack, es un módulo de Texas Instruments, que dota al sistema de conexión wifi. Tras el análisis este módulo es más que suficiente para que el jugador pueda jugar en modo online.

- PC. Para el montaje del servidor y la BBDD es necesario un PC. Ya que es un servidor sencillo, que lo único que hace es recibir peticiones http y dar una respuesta, el PC utilizado es más que suficiente para esta necesidad. La BBDD a emplear también es sencilla, almacena datos básicos de los jugadores, por lo que el PC anteriormente comentado es también válido.

Este análisis técnico nos hace pensar que la creación del juego es viable. Hay que analizar también si con el software existente se cubren las necesidades.
Se necesita:

- Entorno de programación eficiente y que disponga de todo lo necesario para desarrollar el juego.
- Entorno de programación compatible con el Launchpad.
- Documentación sobre el IDE y ejemplo que se ejecuten sobre el Launchpad para entender el funcionamiento.

Tras analizar también la parte software, se llega a la conclusión que se puede desarrollar el juego bajo Energia, ya que cuenta con todo lo necesario, tiene gran cantidad de ejemplos y una comunidad online que sirve de gran ayuda.

En resumen, se puede afirmar que el proyecto es técnicamente viable, que se pueden cumplir los objetivos principales y secundarios.

6. Valoración económica

En este apartado se desglosará el coste que llevaría el desarrollo del proyecto, tanto en materiales como en recursos humanos.

6.1. Hardware

En la siguiente tabla, podemos ver los elementos hardware utilizados para el desarrollo del proyecto.

Descripción	Unidades	Precio unitario	Importe
LaunchPad MSP432P401R	1	10.87€ (13$)	10.87€ (13$)
Educational BoosterPack MKII	1	25€(30$)	25€(30$)
CC3100 BoosterPack	1	16,72€ (20$)	16,72€ (20$)
PC Server	1	600 €	600 €
Total			652,59 €

Tabla 9 Valoración económica. Materiales

6.2. Software

El coste de software es el mostrado en la siguiente tabla:

Descripción	Unidades	Precio unitario	Importe
Window 7 64 bits	1	45,47 €	45,47 €
Pages	1	0 €	0 €
CCS	1	0 €	0 €
Eclipse	1	0 €	0 €
Total			45,57 €

Tabla 10 Valoración económica. Software

6.3. Desarrollo

Para calcular el coste del desarrollo, se han tenido en cuenta las diferentes fases del proyecto:

1. Toma de requisitos y documentación: Esta es la fase más importante del proyecto ya que se van a marcar cuales son los objetivos, así como el estudio para familiarizarse con el entorno y el desarrollo del proyecto. Esta primera fase es una fase de investigación.
2. Desarrollo software: En esta etapa se elaboran los requisitos obtenidos en la primera etapa.
3. Pruebas e informe final: Antes de terminar el proyecto, se ejecutan una fase de prueba para ver si se cumplen con los objetivos y detectar problemas. Si la fase de prueba es satisfactoria se da por concluido el proyecto, si no es así hay que revisar, corregir y volver a pasar las pruebas. En cada fase de prueba se realiza un informe donde quedan plasmadas las pruebas realizas, los fallos encontrados y las correcciones que se deberían de realizar.

Al ser un proyecto de poca complejidad, una única persona podría realizar el desarrollo y otra persona la fase de prueba. El coste de estas personas queda reflejado en la siguiente tabla. Se considera que cada persona tiene un coste de unos 20€/hora. Las horas de desarrollar el informe tendrán un coste de 10€.

Descripción	Unidades	Precio unitario	Importe
Toma de requisitos y documentación	86	20 €	1720 €
Desarrollo Software	172	20 €	3440 €
Prueba	172	20 €	3440 €
Informe	40	10€	400€
Total			9000 €

Tabla 11 Valoración económica. Desarrollo 1

Se ha tenido en cuenta que una persona trabaja a la semana 43 horas. Para la toma de requisitos, se ha empleado una semana y dos personas.Para el desarrollo del producto se han empleado 4 semanas y sólo 1 persona. Para las pruebas y las correcciones se han empleado 4 semanas.

En total el coste sería:

Descripción	Unidades	Precio unitario	Importe
Hardware	1	652,59 €	652,59 €
Software	1	45,47 €	45,47 €
Desarrollo	1	12040 €	9000 €
Total			9698,06 €

Tabla 12 Valoración económica. Costes totales

6.4. Costes de industrialización

Si este proyecto se llegara a comercializar, el coste del proyecto sería el siguiente teniendo en cuenta que para la toma de requisitos la persona encargada será un analista, cuya tarifa por hora será de 30€ y por un analista programador cuya tarifa seria 25€ , para la parte del desarrollo se utilizará un programador cuya tarifa horaria será de 20€, por ultimo para la fase de pruebas se empleará a un tester cuyo salario será de 15€ la hora. Por tanto:

Descripción	Unidades	Precio unitario	Importe
Toma de requisitos y documentación	86 (43 analista/43 AP)	30-25 €	2365 €
Desarrollo Software	172	20 €	3440 €
Prueba e informe	172 (86 programador/86 tester)	15-20 €	3010 €
Total			8815 €

Tabla 13 Valoración económica. Costes de industrialización,desarrollo

También se incluye el coste de la fabricación del hardware. Hay que incluir a parte del Launchpad ,el BoosterPack, CC3100 una carcasa y una batería. Para el precio de la carcasa se ha buscado una carcasa para una placa arduino ya que para este Launchpad no hay.

Descripción	Unidades	Precio unitario	Importe
Launchpad MSP432P401R	1	11.02€	11.02€
Educational BoosterPack MKII	1	25.44€	25.44€
CC3100 SimpleLink	1	16.97€	16.97€
Batería	1	1€	1€
Carcasa	1	5.47€	5.47€
Total			59.9€

Tabla 14 Valoración económica. Coste de industrialización,fabricación

El precio unitario para industrializarlo sería en torno a los 60€. Si se produjeran en cantidades más grandes, el precio podría verse reducido un poco. Aún así es un precio bastante elevado para un objeto como este. Hoy día hay en el mercado por menos de 30€ hay máquinas arcade con juegos [1]. Habría que buscar una placa más barata de bajo consumo y los elementos que permiten al usuario interactuar con la aplicación también tendrían que ser más baratos, el joystick, botones y pantalla por separado, no en el Boosterpack. El módulo WIFI también se podría buscar una alternativa de precio más reducido o eliminar esa opción en el producto a comercializar.

7. Conclusiones

Al comienzo del proyecto se marcaron unos objetivos principales y secundarios que se deberían alcanzar si se quería dar el proyecto como finalizado. A continuación se analizarán estos objetivos con el fin de sacar conclusiones sobre la elaboración del proyecto.

7.1. Objetivos

Los objetivos que se han cumplido son:

a. Realización de la secuencia.

b. Interfaz gráfica donde el usuario puede observar la secuencia a implementar.

c. Manejo de los diferentes botones,joystick etc con los que el usuario interactúa con el juego.

d. Incluir distintos niveles de dificultad en el juego.

e. Guardado de las partidas.

f. Dotar al sistema de wifi para partidas multijugador. Este objetivo era primordial para que el jugador pudiera conectarse con otro jugador para partidas online. Gracias al módulo CC3100 BoosterPack se ha dotado al juego de conexión WIFI para las partidas multijugador.

g. Generación de partidas multijugador. Se ha implementado un servidor que recibe peticiones http cuando el usuario se conectar. Una vez que el usuario está conectado, espera un tiempo establecido a que otro usuario se conecte. Cuando los dos usuarios se han conectado comienza la partida y se generan las secuencias que tienen que reproducir ambos jugadores. Cuando algún jugador pierde se almacena la puntuación que llevaba hasta el momento en BBDD.

h. Pausar juego.

i. Incluir tono sonoro con cada color de la secuencia, cuando empieza el turno del usuario y cuando pierde.

Objetivos no conseguidos:

a. Login usuarios. Este objetivo añadiría valor al proyecto, pero no entra dentro del alcance, ya que no tiene que ver con el sistema embebido en si.

b. Jugar entre usuarios que se conectan desde fuera de la red donde está ejecutándose el servidor.

7.2. Conclusiones

Como se puede ver en el apartado anterior hay objetivos que se han cumplido y objetivos que no. Los objetivos principales se han cumplido por lo que hacen al juego funcional. Las posibles mejoras no se han realizado por lo que quedaría como línea futura.
Se ha sido capaz de desarrollar un juego que corra sobre el Launchpad MSP432R401, el Educational BoosterPack MKII y el CC3100 BoosterPack. Así como desarrollar un servidor para la otra parte de la aplicación que es el juego entre varios usuarios vía wifi.

7.3. Autoevaluación

Este proyecto ha supuesto un reto personal y una gran satisfacción el haberlo conseguido. El principio fue duro ya que era algo totalmente desconocido para mi. Tuve que documentarme bastante sobre todo lo relacionado con los sistemas embebidos y pedir consejo sobre como abordar el tema de programar el sistema embebido.

Una vez que esta primera fase se ha superado, el hecho de programar el sistema ha resultado un poco complicado ya que tuve que cambiar la forma de pensar para programar algo que está constantemente corriendo en un bucle. Gracias al consejo del consultor de implementarlo con Energia pude empezar a ver y modificar distintos ejemplos y entender cómo se programa de esa forma. La única experiencia que tengo programando, a lo que actualmente me dedico, no es la programación de sistemas embebidos, sino de una programación orientada a objeto tanto para aplicaciones web como de escritorio.

Cuando ya se conoce el funcionamiento de Energia,no es complicado de programar el dispositivo, el lenguaje es muy parecido a Java que es el lenguaje que domino aunque le veo algunas deficiencias,Energía no es tan potente en mi opinión. Eso si TI-RTOS cumple totalmente su función para la aplicación en tiempo real. También es un sistema poco pesado, ocupa poca memoria y es rápido en la respuesta en tiempo real.

Desarrollando este proyecto me he dado cuenta, que conocido un lenguaje de programación es fácil adaptar tus conocimientos a cualquier otro y que es más interesante el desarrollo para sistemas embebidos que la realización de una aplicación web.

Debido a la dedicación parcial al proyecto, la línea temporal de realización se ha tenido que ir modificando para poder cumplir los objetivos en los plazos establecidos.

Después de todo el desarrollo, quitando los momentos de estrés ha sido una buena experiencia, que me da la oportunidad de seguir investigando en esta línea, quizás en un futuro adquiriendo los conocimientos necesarios pase de desarrollar aplicaciones web a este tipo de trabajo.

Por la labor de documentación que realicé al principio, me gustaría también seguir en esta línea con el lenguaje Arduino y quizás “cacharrear” un poco con la famosa RaspberryPi.

7.4. Líneas de trabajo futuro

Las líneas futuras que se podría incluir en este proyecto serían:

- Mejorar las partidas online. Crear un módulo de login, para que el usuario se pueda dar de alta con una contraseña y usuario y poder jugar.
- Incluir la posibilidad de conectarse un usuario fuera de la red.
- Si el producto se quisiera comercializar también como línea futura podría ser la investigación de alguna placa de precio menor en donde pueda funcionar el juego para que sea rentable.
- Generación por parte del usuario de sus propias secuencia para jugar con ellas posteriormente, no como hasta ahora que es el sistema el que genera estas secuencias.

8. Glosario

ARM: arquitectura de procesador y microcontroladores.

BBDD: Base de datos

Cortex: Arquitectura de procesadores de ARM

CCS: Code Composer Studio.

Energia: herramienta de desarrollo.

Eclipse: herramienta de desarrollo.

Joystick: dispositivo de control de dos ejes.

LED: Diodo emisor de luz.

LCD: representación visual por cristal liquido.

RTOS: Sistema Operativo de Tiempo Real

9. Bibliografía

La bibliografía utiliza en este proyecto es la citada a continuación:

- http://energia.nu/guide/
- http://energia.nu/reference/
- https://www.hackster.io/
- https://www.freertos.org/
- http://www.43oh.com/
- https://aprendiendoarduino.wordpress.com
- http://www.ti.com/tool/MSP-EXP432P401R
- https://stackoverflow.com
- Sistemas empotrados en tiempo real. José Daniel Muñoz Frías.

10.Anexos

10.1. Ejecución programa

En los anexos, se incluirá una guía de instalación, compilación y ejecución del programa con CCS. También se incluirá cómo ejecutar el servidor para el modo multijugador.

Para el desarrollo del juego se ha utilizado el programa Code Composer Studio y Eclipse para la parte de servidor.

Una vez instalado ambos programas, se va a explicar cómo crear el proyecto y código de la aplicación.

1. Aplicación Simon (Code Compose Studio)

En la ventana principal de CCS accedemos a File-->New--> Other--> Energia Sketck. Aparece la siguiente pantalla:

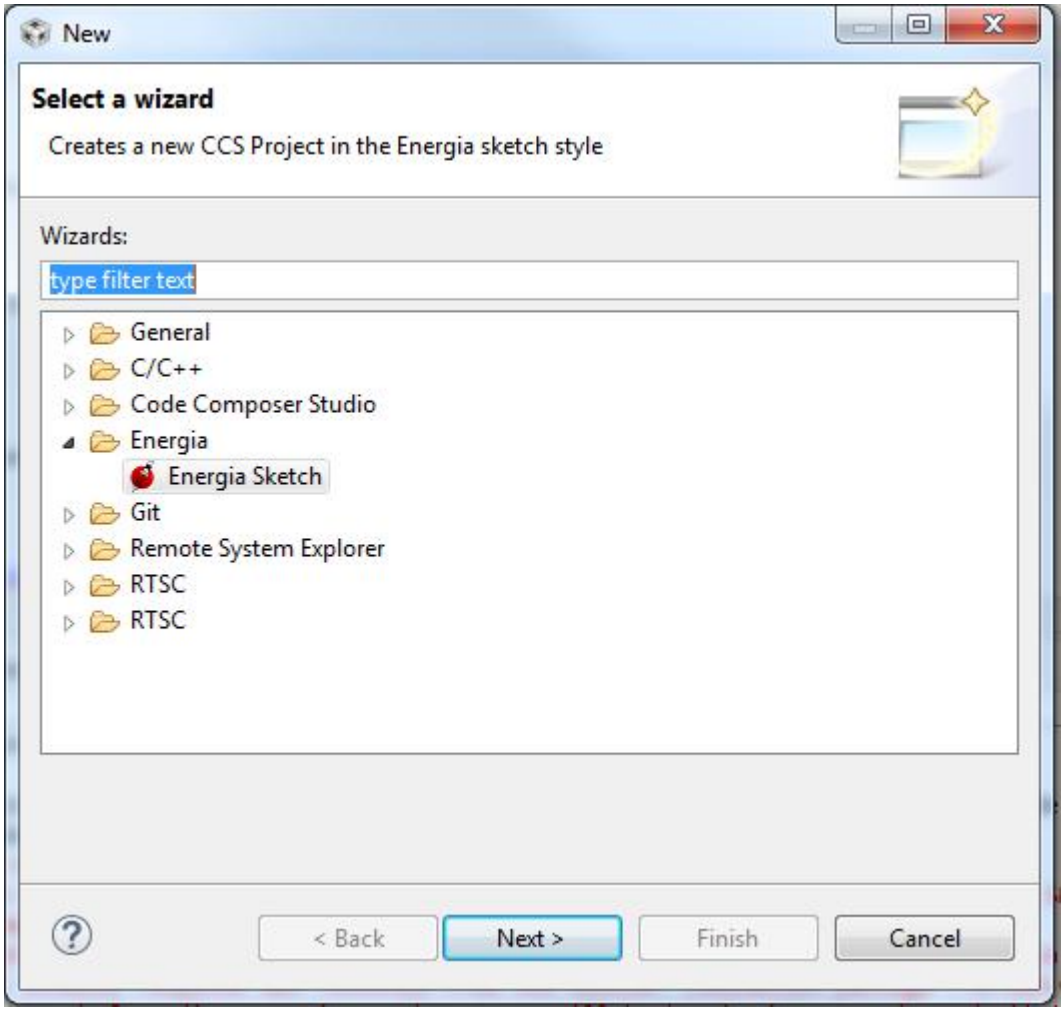


Ilustración 23 - Creación proyecto Energia

Le damos a siguiente y nos aparecerá una pantalla para poder darle nombre al proyecto:

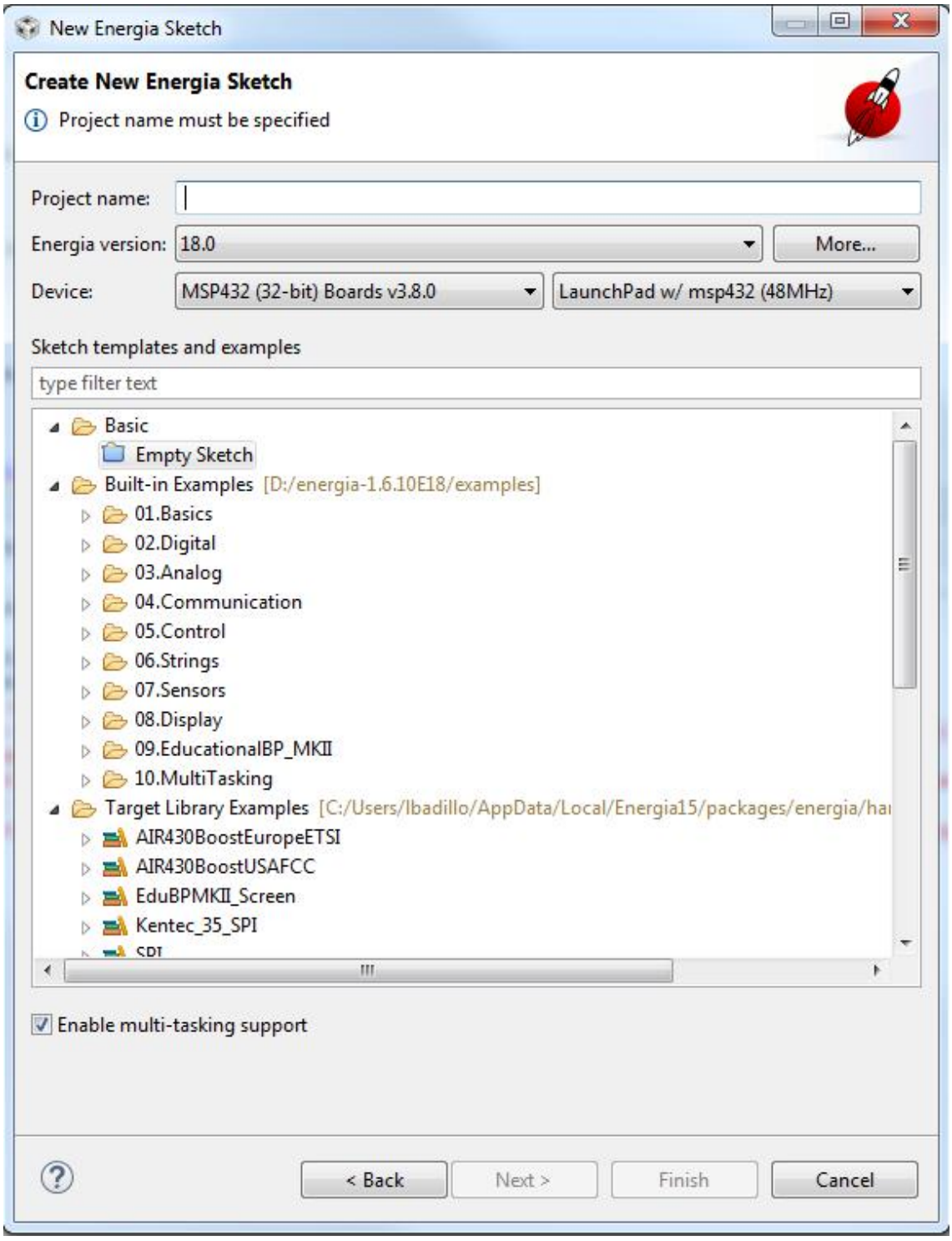


Ilustración 23a - Creación proyecto Energia

Una vez creado el proyecto para ejecutarlo, basta con seleccionar el botón de debug:

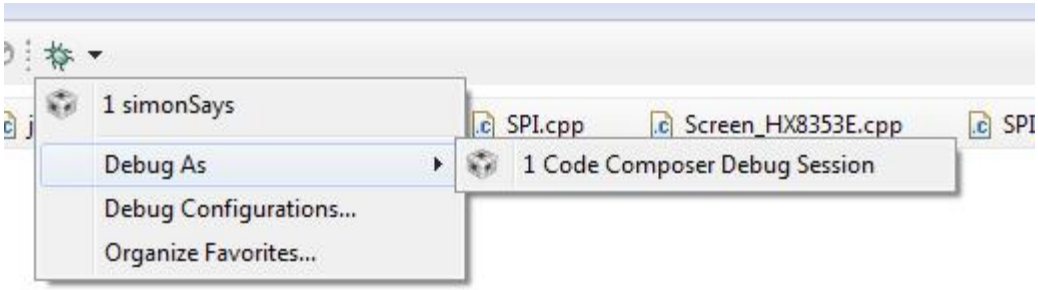


Ilustración 24 - Arranque aplicación

El programa arrancará, cuando arranca se para en setup, basta con pulsar F8 para que continúe con la ejecución. Una vez que se ha ejecutado se podrá ver en la pantalla del BoosterPack el menú de inicio de la aplicación. Para seguir la ejecución del programa nos podemos apoyar en el terminal, donde se verán las trazas puestas en el código.

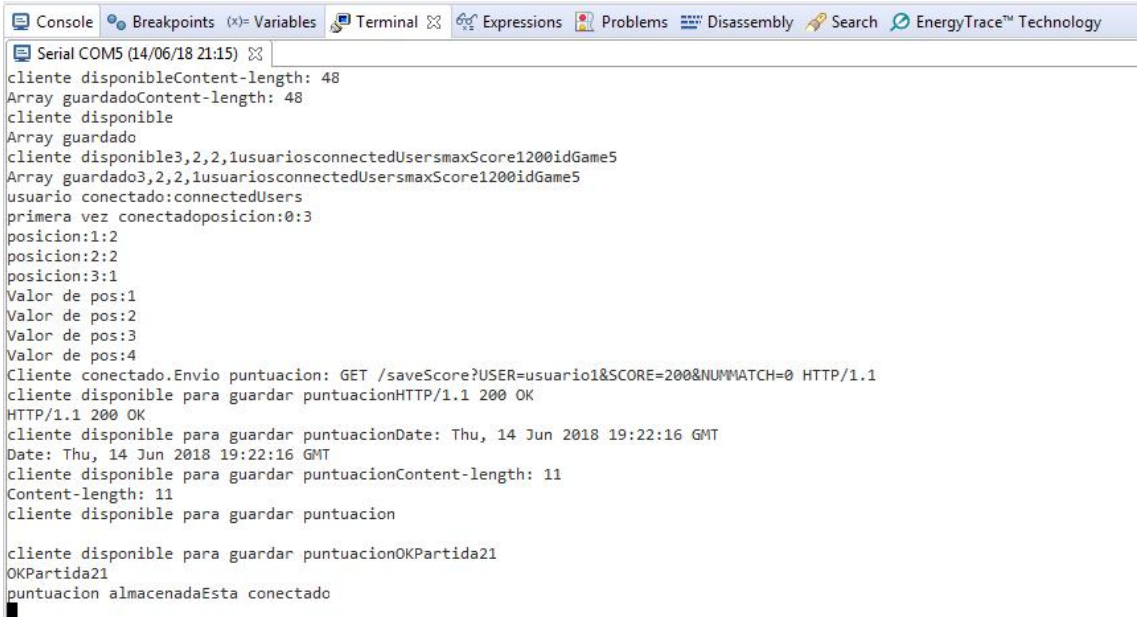


Ilustración 25 - Trazas terminal

2. Aplicación Servidor (Eclipse)

Una vez abierto la aplicación Eclipse, se crea un proyecto del tipo Dynamic Web Proyect. También hay que instalarse un servidor, en nuestro caso Tomcat V7.

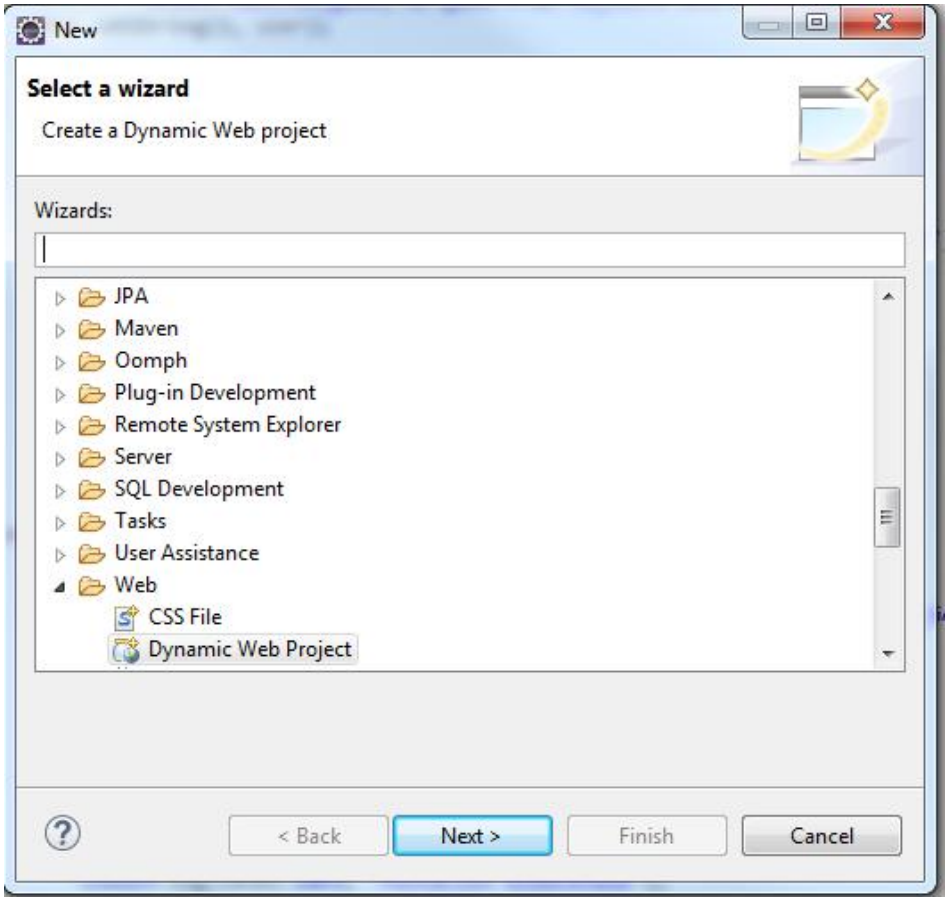


Ilustración 26 - Creación proyecto servidor

La estructura del proyecto servidor quedaría de la siguiente forma:

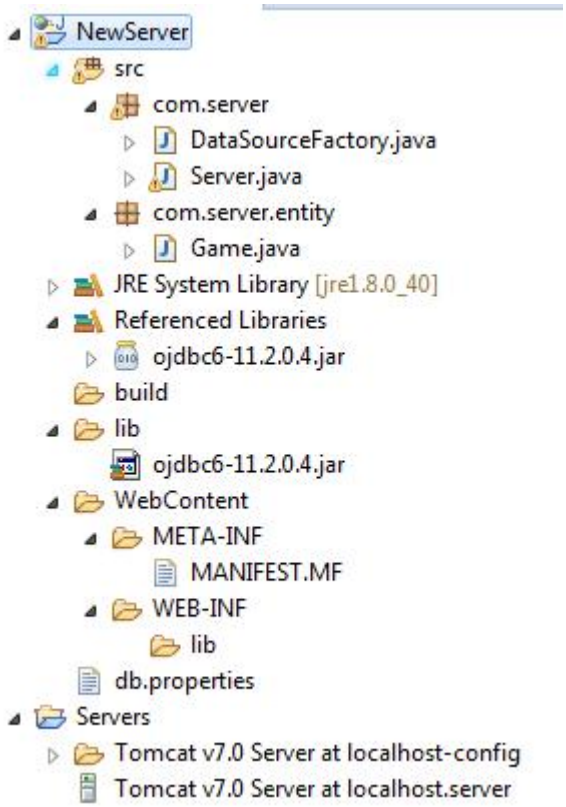


Ilustración 27 - Estructura proyecto servidor

Para poder almacenar el histórico de partidas online también se procede a crear una BBDD con los datos proporcionados en db.properties. La tabla que hay que crear es la proporcionada en el fichero export.

Para arrancar el servidor basta con pulsar run-->server

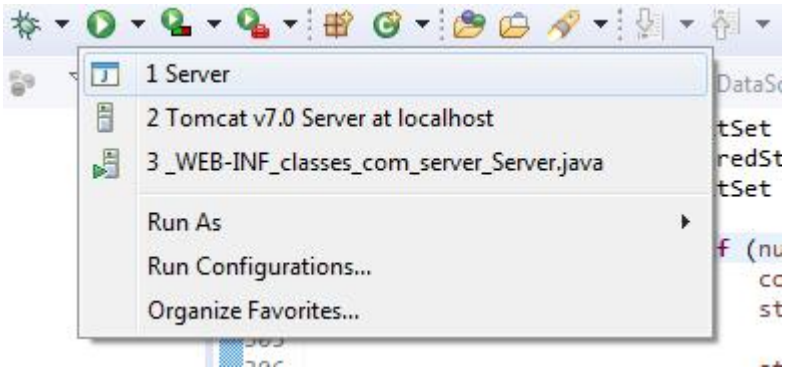


Ilustración 28 - Arranque proyecto servidor

También nos podemos apoyar en la consola de eclipse para ver la evolución del servidor, las tareas que realiza.

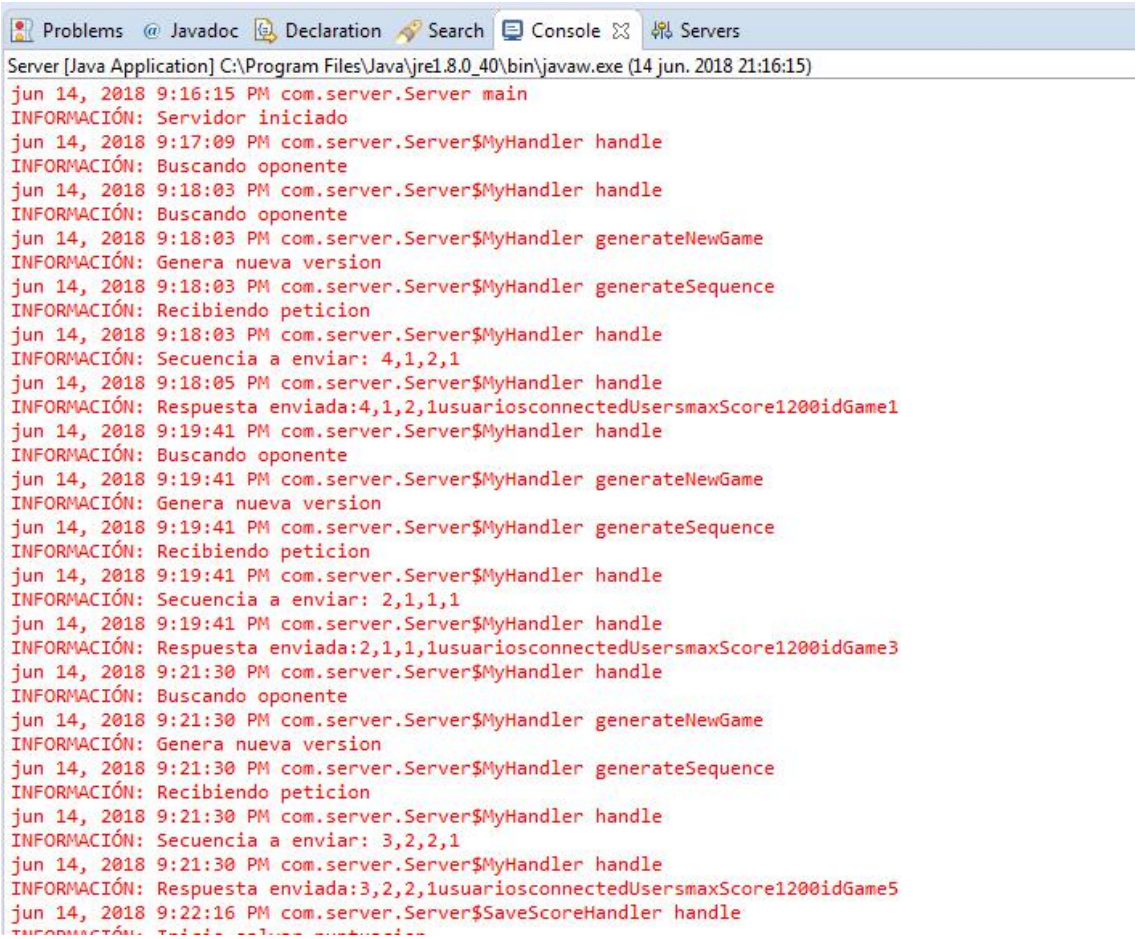


Ilustración 29 - Trazas proyecto servidor

10.2. Código empleado

Para la parte del proyecto de sonido de game over y la conexión a WIFI, se ha empleado código utilizado en otros proyectos de ejemplos.