Universidad Nacional de La Matanza

Departamento de Ingeniería e Investigación Tecnológicas

Sistemas Operativos Avanzados

Internet of Things

Sistemas Embebidos y Android

Molestador

Segundo cuatrimestre – Año 2018

–

Dia de cursada: martes Turno: Noche Aula: 266

Docentes:

- Lic. Graciela de Luca
- Ing. Waldo Valiente
- Ing. Esteban Carnuccio
- Ing. Mariano Volker
- Ing. Sebastian Barillaro
- Ing. Gerardo Garcia

Integrantes:

- Luna, Damián DNI: 38.468.480
- Massaccisi, Daniel Alberto DNI: 38.457.964
- Morales Peña, Pablo Gabriel DNI: 36.159.514
- Romero, Federico Andrés DNI: 38.709.857

Objetivo del Trabajo Práctico

Implementar los conocimientos obtenidos durante la cursada para desarrollar un dispositivo IoT que consiste en un despertador el cual propone cumplir una serie de desafíos para desactivar la alarma. Para este proyecto se utilizó un sistema embebido y una aplicación Android.

Problema y estrategia

Es muy normal para aquellas personas que necesitan levantarse temprano para cumplir con un horario la utilización de un reloj despertador para evitar quedarse dormido e incumplir con sus obligaciones. Pero también es cotidiano que en el momento en que suena la alarma, y afectados por la somnolencia, muchos opten por posponerla para poder seguir descansando unos minutos más. Este acto se lo conoce como *snooze*.

El problema que trae aparejada esta actividad es que muchas personas terminan quedándose dormidas y despertando mucho más tarde de lo que habían planeado. Para mitigar este deseo de postergar el despertar, hemos planteado una solución que consiste en dificultar el apagado de la alarma mediante el cumplimiento de ciertos pasos o desafíos, que obligan al usuario a levantarse y aplicar las facultades cognitivas a disposición al momento de saltar de la cama. Una vez cumplidos los desafíos y apagada la alarma el sujeto dispondrá de mayor actividad cerebral para poder arrancar con el cumplimiento de sus tareas y responsabilidades.

Descripción detallada del funcionamiento

Al encender el Molestador se le debe indicar la hora actual mediante una opción en la aplicación Android. Después de esto, el dispositivo quedara a la espera de una alarma, lo cual tambien se configura con la aplicación. Cuando la hora de la alarma se cumple, comenzará a sonar el buzzer en el Molestador, se prenderá el led y se recibirá una notificación en el celular. Para desactivarla son necesario el cumplimento de cuatro desafíos, dos realizados en el embebido y los otros dos en el smartphone.

Los pasos en el embebido son los siguientes:

1. Presionar los botones en una secuencia aleatoria
2. Utilizando el Joystick y visualizando la pantalla, colocar tres veces un pixel en un rectángulo.

Los pasos en el celular son:

1. Realizar una determinada cantidad de pasadas con la mano sobre el sensor de proximidad hasta completar un círculo de progreso en la pantalla.

2. Inclinar alternadamente de izquierda a derecha el celular una determinada cantidad de veces hasta completar también otro círculo de progreso visualizado en la pantalla.

Una vez terminados estos desafíos se apagará la alarma. Cabe destacar que tanto el dispositivo Android y el embebido hacen uso de sus respectivos sensores de iluminación que son los que verificarán que se ha encendido la luz de la habitación. Si no se dispone de suficiente luz ambiente no se podrá avanzar en los desafíos, por lo que el usuario está obligado a encenderla.

Problemas durante el desarrollo

I. Problema con la pantalla

La misma no funcionaba debido a que las salidas del Arduino Esplora son todas de 5v y la misma trabaja en 3.3v. La solución fue utilizar divisores de tensión en cada una de sus conexiones.

II. Problema con el Bluetooth

El Bluetooth comprado originalmente, cuyo modelo era un clon del HC-10 (MLT-BT05), no enviaba datos, pero si recibía. Sospechamos que la falla era de fábrica, por lo que decidimos comprar otro Bluetooth, pero de un modelo diferente (HC-05).

Diagrama de componentes

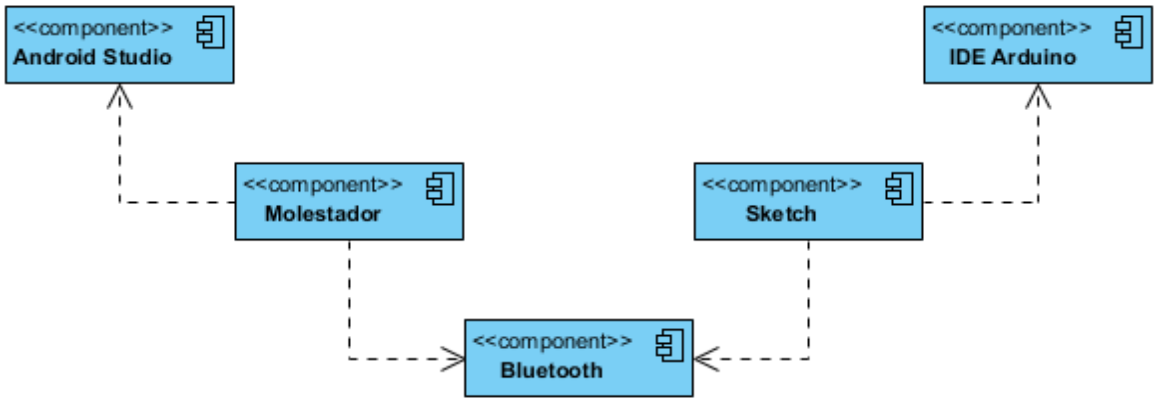


Diagrama de conexiones

Diagrama de conexión con pantalla LFT

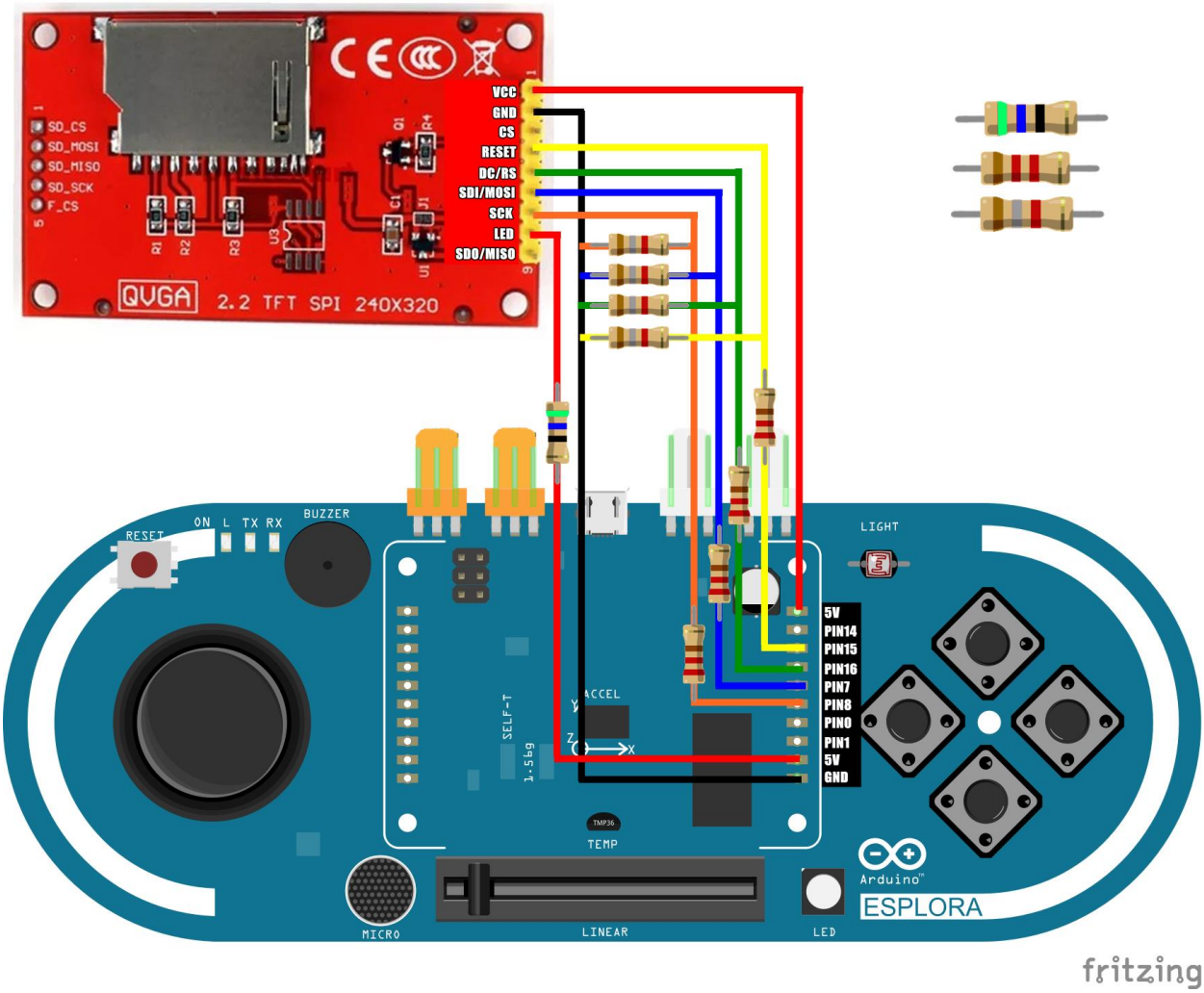
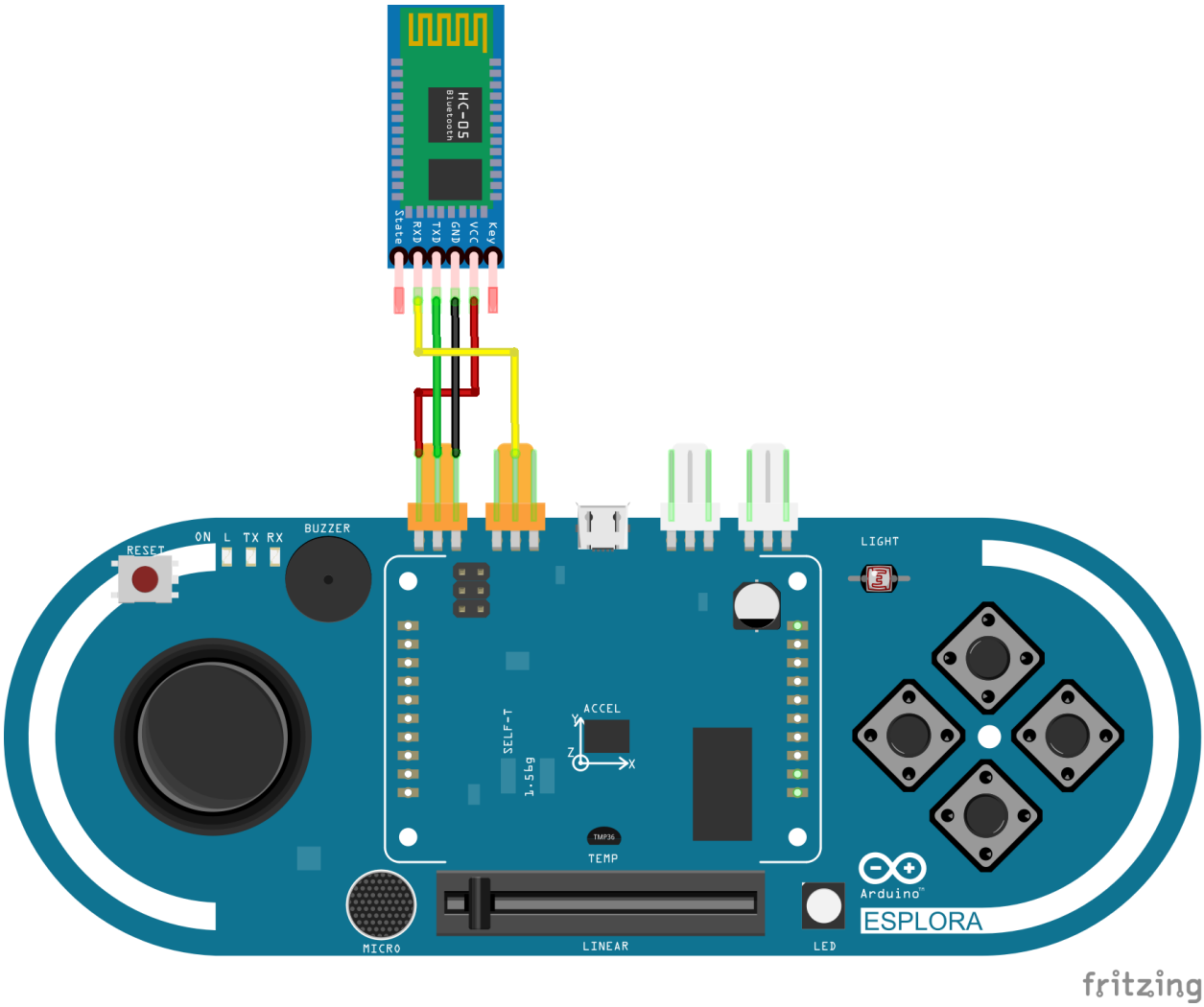


Diagrama de conexión con dispositivo Bluetooth



Conexión

Los actuadores y sensores utilizados en el proyecto se encuentran ya integrados a la placa del Arduino Esplora por lo que no realizamos ningún trabajo de conexión sobre los mismos. Donde sí se realizó la conexión fue en el caso del dispositivo bluetooth y de la pantalla LCD.

Con el Bluetooth se utilizaron cables Arduino para conectarlo a los pines Tinkerkit, ubicados en la parte superior izquierda de la placa.

Para la pantalla, se utilizó una placa QVGA 2.2 TFT SPI, conectada a través de una placa perforada para adaptar los niveles de tensión y amperaje de salida del Arduino a la entrada de la pantalla, se omitió la conexión opcional para almacenamiento SD de la pantalla.

Componentes Hardware

I. Arduino Esplora

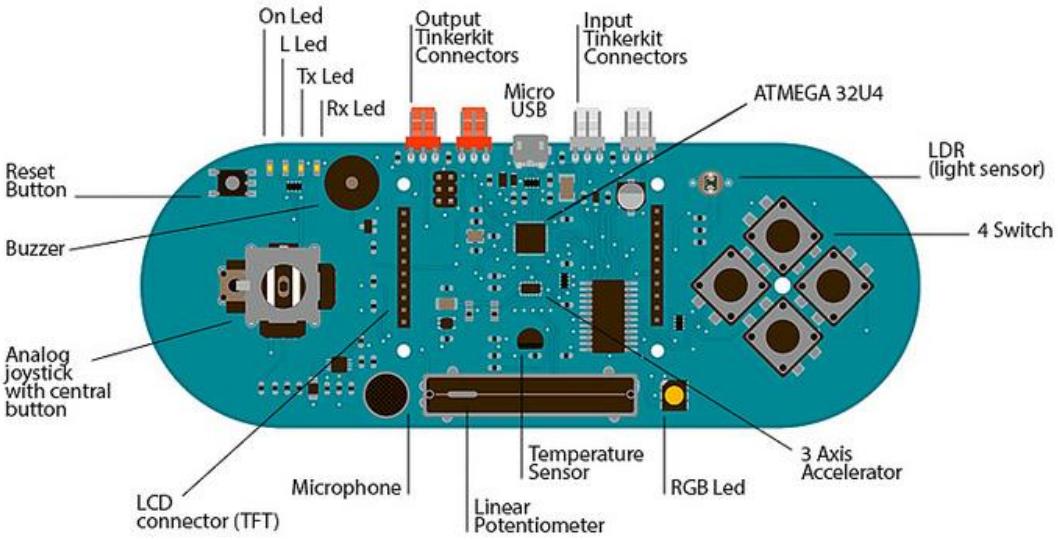
- Microcontrolador: ATMEGA32U4
- 5V Tensión de funcionamiento
- Memoria Flash 32 KB de los cuales 4 KB utilizado por gestor de arranque
- SRAM 2.5 KB
- EEPROM 1 KB
- Velocidad de reloj 16 MHz

Sensores incorporados:

- Sensor de luz (LDR)
- Sensor de temperatura
- 4 pulsadores
- Joystick (pad analógico) con botón central

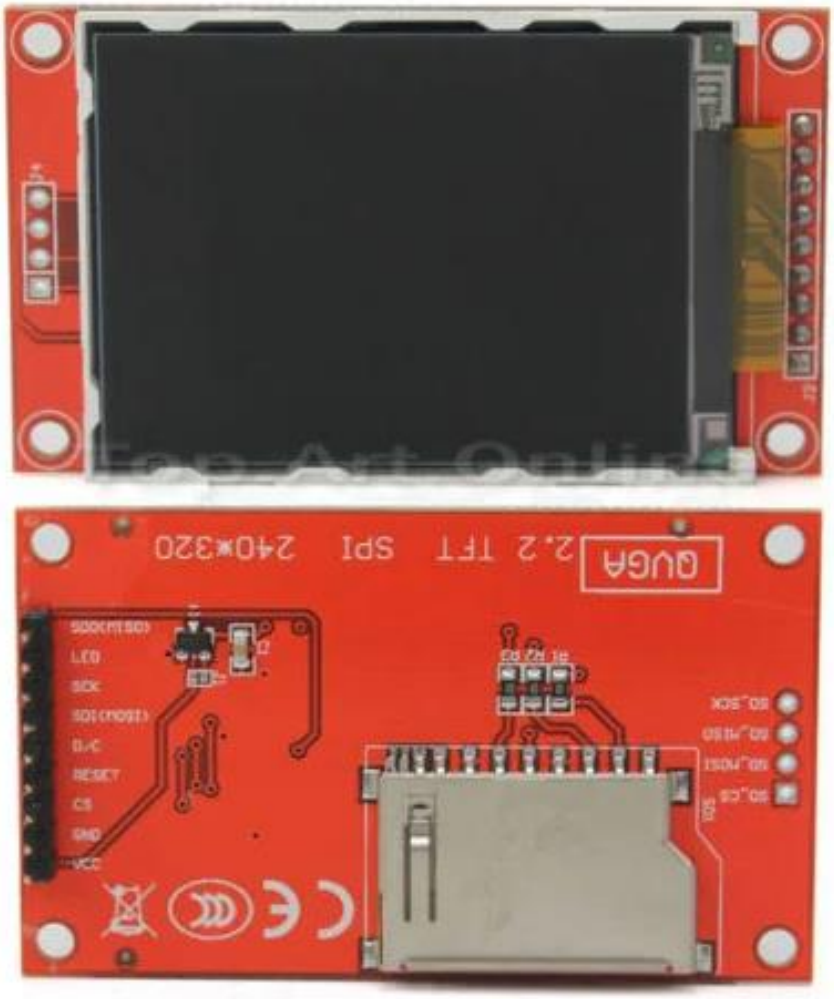
Actuadores:

- Buzzer
- LED RGB



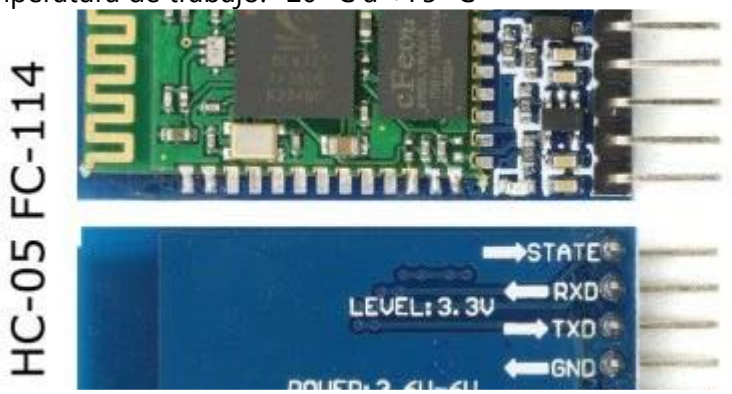
II. Pantalla QVGA 2.2 TFT SPI

- Modelo: ILI9341
- Tamaño: 2.2 pulgadas
- Dimensiones: 67 mm de longitud, 40 mm de ancho y 4mm de grosor
- Resolución: 320x240
- Alimentación: 5v.
- Lógica: 3.3v
- Almacenamiento: MicroSD
- Colores: 16 millones
- Driver IC ILI9341



III. Bluetooth HC-05

- Voltaje de Operación: +3.6VDC - 6VDC
- Consumo Corriente: 50mA
- Interface: Serial TTL
- Protocolo Bluetooth: Bluetooth Specification v2.0+EDR
- Frecuencia: Banda ISM 2.4GHz
- Modulación: GFSK (Gaussian Frequency Shift Keying)
- Potencia de transmisión: =4dBm, Class 2
- Sensibilidad: = -84dBm a 0.1% BER
- Velocidad: 1Mbps
- Seguridad: Autenticación y encriptación
- Perfil: Bluetooth serial port
- Temperatura de trabajo: -20 °C a +75 °C



Componentes Software

I. Adafruit_GTX

Biblioteca con las primitivas para dibujar en la pantalla.

II. Adafruit_ILI9341

Extensión de la biblioteca anterior para utilizar esta pantalla particularmente.

III. TimeLib

Biblioteca para manejar el tiempo, permite llevar fácilmente cuenta del tiempo.

IV. TimeAlarms

Extensión de la biblioteca anterior, agrega la funcionalidad de programar alarmas.

V. SoftwareSerial

Biblioteca para comunicaciones seriales, la usamos para definir pines digitales específicos para el bluetooth.

VI. Android Studio

Entorno de desarrollo para la aplicación de Android utilizando el lenguaje Java

Comunicaciones con los dispositivos

Para realizar la comunicación con el sistema Arduino, se implementó una conexión mediante un canal Bluetooth. Cada vez que se quiera utilizar el Molestador se debe actualizar la hora de este y programar la alarma. Esta información es proporcionada por la aplicación y para llevar a cabo dicha comunicación se implementaron clases Thread, BluetoothSocket, BluetoothDevice, BluetoothAdapter y Handler. Una vez que se activa la alarma entre el Molestador y la aplicación se realiza un pasaje de información con respeto a los pasos a completar para la desactivarla.

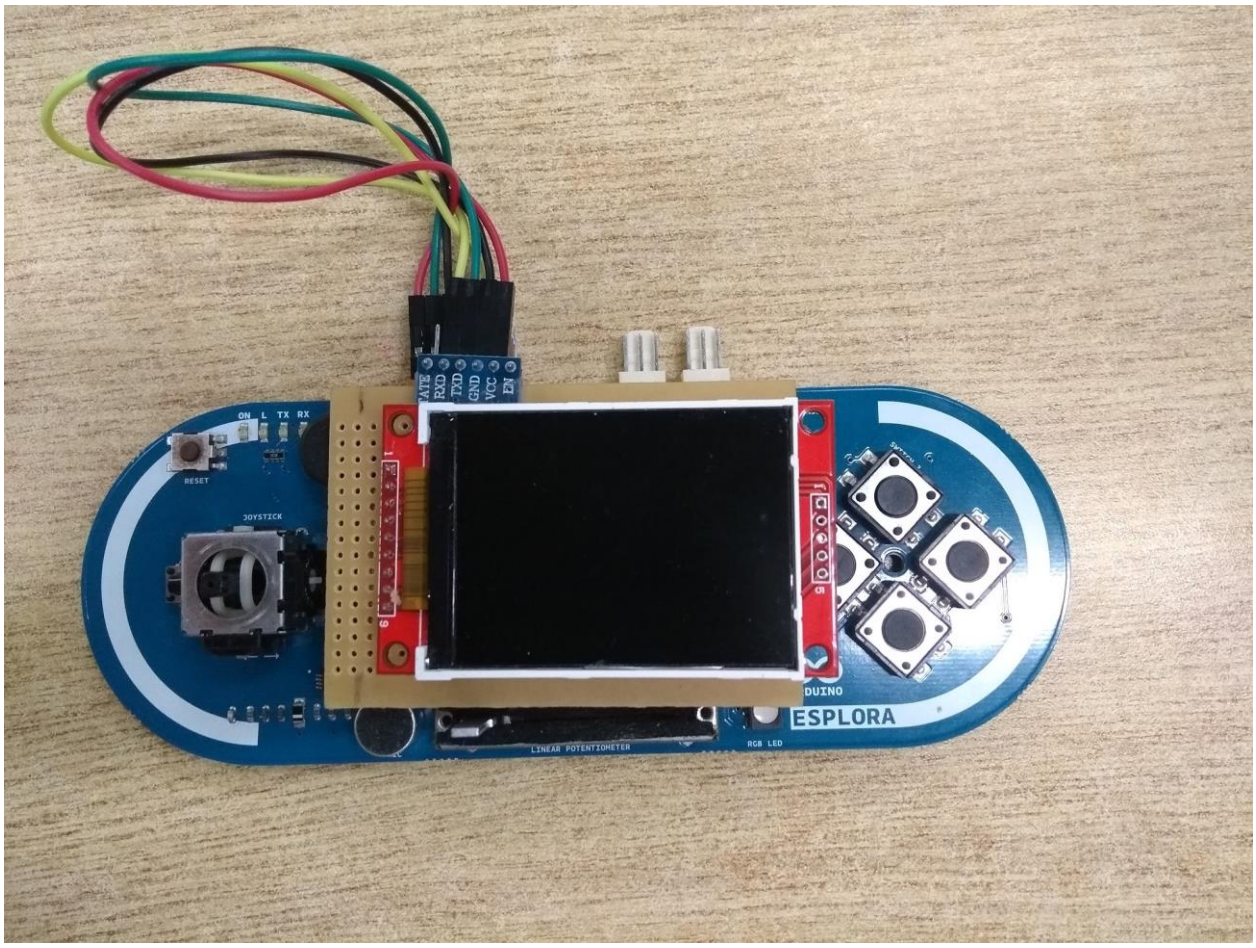
Lenguajes y entorno de desarrollo

Para el desarrollo de la aplicación Android se trabajó en Android Studio, un entorno de desarrollo integrado basado en el software IntelliJ IDEA de JetBrains, donde se utilizó lenguaje Java para la implementación de las distintas clases, y lenguaje XML para la declaración de los componentes de la aplicación.

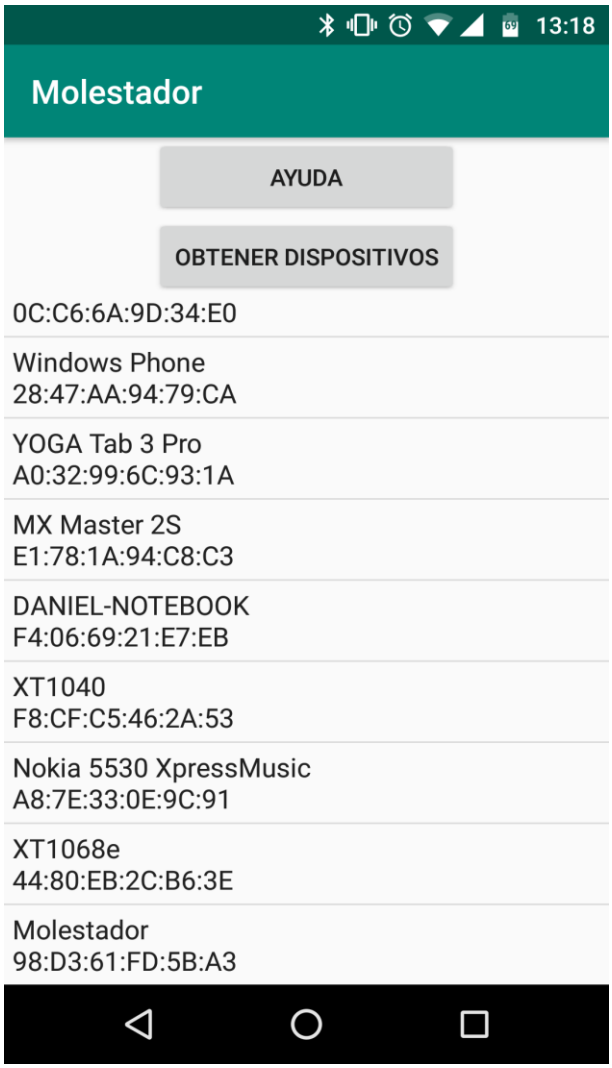
Para el desarrollo del sistema embebido se trabajó en Arduino IDE versión 1.8.6 y se programó en el lenguaje Arduino que está basado en C y soporta todas las funciones del estándar C y algunas de C++.

Fotografías y capturas

Sistema embebido



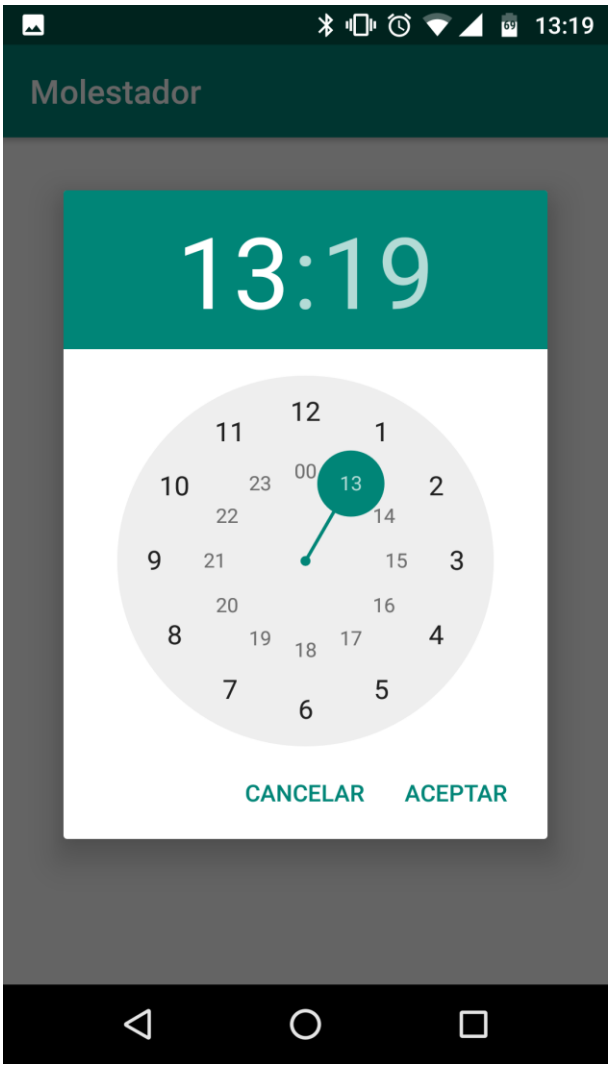
Sistema Android



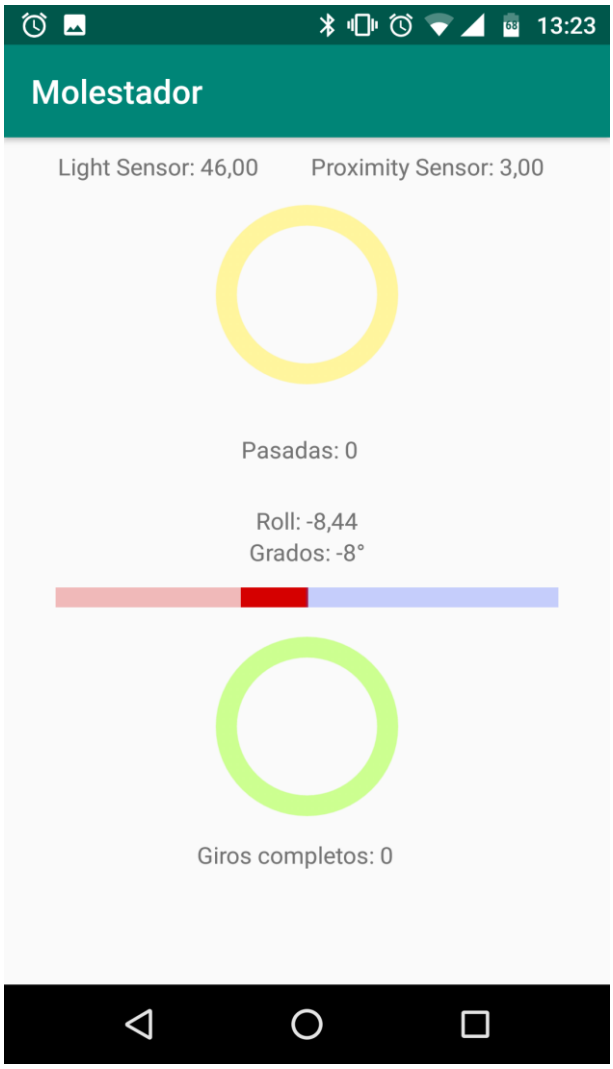
Pantalla principal de la app

Ayuda de la app

Pantalla de configuración de hora y alarma



Selector de hora para la alarma



Pasos para apagar la alarma

Fuentes consultadas

Conexión bluetooth:

http://www.martyncurrey.com/bluetooth-modules/

Conexión pantalla:

https://www.instructables.com/id/Arduino-TFT-display-and-font-library/