UNIVERSITAT JAUME I

ESCOLA SUPERIOR DE TECNOLOGIA I CIÈNCIES EXPERIMENTALS

MÀSTER UNIVERSITARI EN ENGINYERIA INDUSTRIAL

DISEÑO E IMPLEMENTACIÓN DE LOS SISTEMAS AUTOMÁTICOS DE LAS SALAS DE ESCAPE DE LASER GAME EVOLUTION CASTELLÓN

TRABAJO FIN DE MÁSTER

AUTOR

Francisco Sayas Doménech

DIRECTOR

Julio Ariel Romero Pérez

Castellón, noviembre de 2020

"Si sales vivo de este agujero arrasarás en el mundo entero.
¡Hazte invencible! ¡Hazte ingeniero! ¡Hazte industrial!"

Alumnos de la Escuela Técnica Superior de Ingenieros
Industriales de la Universidad Politécnica de Madrid, 2009

Agradecimientos

- A Julio Ariel Romero, por su mentorización y apoyo en la realización de este proyecto. A Javier Ares y a Raquel Castelló, por su confianza en mi persona para diseñar una pequeña parte de su gran aventura empresarial.
- A Diego Centelles, a Alberto Cancho, a José Javier Fernández, a Raúl Marín y al resto de compañeros del Interactive and Robotic Systems Lab. A Guillem Monrós, a Salvador Francisco Torro, a Sergio Chiva y al conjunto del Grupo de Fluidos Multifásicos. Por error, no expresé por escrito mi agradecimiento hacia ellos en mi Trabajo Final de Grado y es de justicia que al menos lo haga aquí, por toda su ayuda en las experiencias vividas durante años en ambos grupos de investigación.
- A los compañeros y a los miembros del personal de la Universitat Jaume I que me han acompañado en esta etapa, tanto en mis actividades académicas como en las extracurriculares. Al Parque Científico, Tecnológico y Empresarial de la Universitat Jaume I, por tantos años soportando muchas de mis iniciativas y por su apoyo para el prototipado y la fabricación de trabajos de estudiantes.
- A las asociaciones Hackerspace Castellón y engiOn, entidades que han marcado mi vida universitaria y en las que he podido crecer personal y profesionalmente, disfrutando de la ciencia y la tecnología. A mis compañeros de UJI Electric Racing Team, por contar conmigo para formar parte de un sueño que hicimos realidad.
- A mi familia, bajo riesgo de ser desheredado si se me olvida mencionarla, y a mis amigos.
- Y, con un recuerdo especial, a José Enrique Juliá, quien me brindó la oportunidad de iniciarme en el mundo de la investigación. A Quique.

Gracias

Francisco Sayas
Noviembre de 2020

Resumen

Antecedentes: Laser Game Evolution Castellón es un centro de ocio en el que se requiere diseñar e implementar múltiples sistemas automáticos para tres salas de escape o escape rooms. Estas habitaciones tematizadas son parte de un juego en el que se encierra a un grupo de participantes en su interior, con el objetivo de que los jugadores resuelvan distintos desafíos para escapar antes de que finalice un tiempo determinado. La mayoría de las salas de escape existentes contienen retos que consisten en abrir diferentes tipos de candados mecánicos. Para sus salas, la empresa quiere hacer uso de sistemas electrónicos y nuevas tecnologías con el fin de crear experiencias de juego innovadoras que sorprendan a los participantes.

Justificación: La empresa desea implementar diversos sistemas automáticos en las salas de escape, tales como la apertura de puertas y cajones, el encendido de luces y la detección de objetos del juego. Por las características de los diferentes retos, los automatismos deben ser diseñados, programados e integrados en las salas de forma personalizada, ya que cada juego es único y no existen soluciones comerciales.

Objetivos: Diseñar e implementar los siguientes sistemas automáticos: Modificación de un armario eléctrico para apertura mediante combinación de pulsadores. Modificación de una centrifugadora de laboratorio para apertura mediante código numérico. Parpadeo de una bombilla según un código. Apertura de un cajón mediante una combinación de golpes. Apertura de una puerta mediante control de acceso. Apertura de un compartimento secreto tras situar un objeto del juego en un lugar determinado. Simulación de medida en indicador analógico a partir del accionamiento de válvulas hidráulicas. Detección del traslado de un objeto entre dos localizaciones en un tiempo determinado.

Metodología: Estudio de las necesidades del cliente para cada mecanismo. Diseño de diferentes alternativas de implementación. Desarrollo de prototipos basados en la plataforma Arduino para validar las soluciones elegidas. Documentación de la implementación final y de las tareas de instalación y mantenimiento.

Palabras clave: Sala de escape, escapismo, Arduino, sistemas automáticos, domótica, electrónica.

Información del Proyecto

- Título: Diseño e implementación de los sistemas automáticos de las salas de escape de Laser Game Evolution Castellón
- Destinatarios:
  - Universitat Jaume I
  - Laser Game Evolution Castellón
- Promotor: Javier Ares Armero (Laser Game Evolution Castellón Sociedad Limitada)
  - CIF: B12990420
  - Dirección: Calle Peri 15 Núm. 1, 14, (CP 12006) Castellón
  - Teléfono: 964747030
  - Correo electrónico: castellon@lasergame-evolution.com
- Proyectista: Francisco Sayas Doménech
  - Titulación:
    - Graduado en Ingeniería en Tecnologías Industriales
    - Estudiante de Máster Universitario en Ingeniería Industrial
  - Entidad: Universitat Jaume I

Parte I

Índice

Índice

Índice de figuras

Parte II

Memoria

Capítulo 1

Objeto

El presente Proyecto describe el diseño y la implementación de una serie de sistemas automáticos destinados a formar parte de varios de los juegos de escapismo del centro de ocio Laser Game Evolution Castellón.

Las salas de escape o escape rooms son parte de un juego en el que se encierra a un grupo de participantes en su interior, con el objetivo de que los jugadores resuelvan distintos desafíos para escapar antes de que finalice un tiempo determinado. La mayoría de las salas de escape existentes basan sus retos en abrir diferentes tipos de candados mecánicos. Por ello, la empresa quiere hacer uso de sistemas electrónicos y nuevas tecnologías para crear experiencias de juego innovadoras que sorprendan a los participantes.

En concreto, se desarrollan los siguientes automatismos:

- Modificación de un armario eléctrico para apertura mediante combinación de pulsadores.
- Modificación de una centrifugadora de laboratorio para apertura mediante código numérico.
- Parpadeo de una bombilla según un código.
- Apertura de un cajón mediante una combinación de golpes.
- Apertura de una puerta mediante control de acceso.
- Apertura de un compartimento secreto tras situar un objeto del juego en un lugar determinado.
- Simulación de medida en indicador analógico a partir del accionamiento de válvulas hidráulicas.
- Detección del traslado de un objeto entre dos localizaciones en un tiempo determinado.

Debido a las características especiales que presentan los diferentes retos de las salas de escape, los automatismos deben ser diseñados, programados y preparados para la integración en las salas de forma personalizada, ya que cada juego de escapismo es único y no existen soluciones comerciales adaptadas.

Capítulo 2

Alcance

El ámbito de aplicación de este Proyecto es el desarrollo y la implementación de sistemas automáticos exclusivamente para las salas de escape de Laser Game Evolution Castellón. No obstante, el mismo puede servir como referencia para proyectar otras instalaciones similares.

El alcance del Proyecto se limita a ofrecer una solución de carácter electrónico, seleccionando los componentes necesarios, describiendo el conexionado entre ellos y su programación. Los aspectos mecánicos necesarios para implementar cada automatismo son responsabilidad del Promotor. Si bien el Proyectista diseña algunos elementos estructurales que sirven para realizar los diferentes prototipos, el desarrollo mecánico de los automatismos no forma parte del alcance de este Proyecto.

Sin ánimo de que en un montaje en particular no existan otras consideraciones indicadas en el Proyecto, en general los siguientes preceptos relacionados con el alcance y acordados entre el Promotor y el Proyectista son respetados:

- La definición de la temática que envuelve a cada automatismo o de la función que desempeña cada montaje en el juego de escape es responsabilidad del Promotor, y es de carácter confidencial. Esta información no es requerida ni detallada en el Proyecto, aunque en algún caso es compartida con el Proyectista con el objetivo de que este entienda mejor el propósito del sitema automático y sea de ayuda en la definición de los requisitos de diseño.
- La elección de contraseñas, códigos, secuencias... propias de cada juego es responsabilidad del Promotor. El mismo debe comunicarlas al Contratista para su implementación durante la ejecución del Proyecto en el código fuente de la programación de los automatismos. En ningún caso los secretos definitivos son expuestos en el Proyecto. Los códigos existentes en el Proyecto son ejemplos utilizados para verificar el funcionamiento de los prototipos.
- La selección y el montaje del componente de cada sistema automático que interactúa con el mecanismo o con el medio es responsabilidad del Promotor. Las características de estos componentes necesarias para el proceso de diseño de los automatismos deben ser notificadas al Proyectista.

- La instalación de la red eléctrica de baja tensión de las salas de escape es responsabilidad del Promotor, que debe proveer una toma de corriente en las proximidades de las localizaciones de la sala en las que se requiera un automatismo. La selección de la fuente de alimentación de los sistemas automáticos es responsabilidad del Proyectista.

Capítulo 3

Antecedentes

Laser Game Evolution Castellón Sociedad Limitada es la empresa responsable de un centro de ocio ubicado en la calle Peri 15 número 1, 14, 12006 Castellón de la Plana. El centro debe su nombre a su principal instalación, una serie de laberintos en los que los clientes pueden realizar partidas de Laser Game, también llamado Laser Tag o Laser Combat. Este es un juego deportivo que simula un combate entre varios participantes.

Como actividad alternativa, la empresa desea habilitar tres salas de escape. Dos de ellas se encuentran en la primera planta del centro. Estas dos salas son referenciadas por el nombre de los juegos que se desarrollarán en su interior: Contagio y Secuestro. La tercera sala está ubicada en la planta baja y actualmente se encuentra en construcción, sin tener nombre definitivo. En términos de apariencia, las salas se asemejan a las habitaciones normales de una vivienda, con la salvedad de que están decoradas para cada juego.

Figura 3.1: Anuncio de Contagio

> *"¿Conseguirás escapar o serás la próxima víctima? Tras una serie de desapariciones en Castellón decidís acudir al despacho de un detective en busca de respuestas. Pero despertáis desorientados y encerrados en un extraño e inquietante lugar, no recordáis nada... vuestros peores temores se confirman, ¡habéis sido secuestrados!" Descripción comercial de Contagio*

Figura 3.2: Anuncio de Secuestro

> *"¿Serás capaz de salvar la Humanidad en 60 minutos? Un letal virus amenaza con acabar con la raza humana, vosotros sois los encargados de ir en búsqueda del valioso antídoto. Para ello, deberéis adentraros en el laboratorio del Dr Hawkins que lleva años clausurado en cuarentena por un accidente vírico, ¡tened cuidado! una vez dentro no hay marcha atrás...estaréis contagiados!" Descripción comercial de Secuestro*

El Promotor no desea que el Proyectista tenga acceso a más datos de los estrictamente necesarios para realizar su cometido, por lo que solicita que los diseños contemplen su instalación de forma flexible. Además se requiere que, en la medida de lo posible, los sistemas puedan ser modificados en el futuro en caso de ser necesario para añadir nuevas características.

Para realizar la instalación física los sistemas automáticos, el Promotor cuenta con cajas eléctricas de conexión, tubos y canales protectoras de diferentes tamaños.

La centrifugadora de laboratorio, máquina a modificar, se cede al Proyectista para el desarrollo de la solución.

Capítulo 4

Normas y referencias

4.1 Disposiciones legales y normas aplicadas

4.1.1 Marcado CE

Las Directivas de Marcado CE que serían de aplicación en caso de que el resultado del Proyecto quisiera ser comercializado son las siguientes:

- Compatibilidad Electromagnética (EMC).
- Restricción uso de sustancias peligrosas en aparatos eléctricos y electrodomésticos (ROHS).

4.1.2 Comisión Electrotécnica Internacional (CEI)

- SC 17C Assemblies.
- TC 21 Secondary cells and batteries.
- SC 21A Secondary cells and batteries containing alkaline or other non-acid electrolytes.
- TC 32 Fuses.
- TC 44 Safety of machinery - Electrotechnical aspects.
- SC 47F Micro-electromechanical systems.
- SC 48D Mechanical structures for electrical and electronic equipment.
- TC 94 All-or-nothing electrical relays.

4.1.3 Comité Europeo de Normalización Electrónica (CENELEC)

- CLC/SR 23 Electrical accessories.
- CLC/SR 23B Plugs, socket-outlets and switches.
- CLC/SR 25 Quantities and units.
- CLC/SR 47F Micro-electromechanical systems.
- CLC/SR 48D Mechanical structures for electronic equipment.
- CLC/TC 94 Relays.
- CLC/SR 94 All-or-nothing electrical relays.
- CLC/SR 103 Transmitting equipment for radiocommunication.
- CLC/BTTF 60-1 Assembly of electronic equipment.

4.1.4 IEC

- IEC 62620:2014 Secondary cells and batteries containing alkaline or other non-acid electrolytes - Secondary lithium cells and batteries for use in industrial applications.

4.1.5 ISO

- ISO 17894:2005 Ships and marine technology – Computer applications – General principles for the development and use of programmable electronic systems in marine applications.

4.1.6 Asociación Española de Normalización (AENOR)

AENOR ratifica mediante normas UNE algunas de las normas expuestas en este capítulo.

4.2 Programas de cálculo

En el presente Proyecto no se utiliza ningún programa de ingeniería específico para realizar los cálculos. Sin embargo, es de interés resaltar que se utiliza el siguiente software para la compilación del código fuente de los programas de los automatismos.

- Arduino IDE 1.8.13
    - Paquete de soporte para tarjetas SparkFun AVR Boards 1.1.13
    - Librería Keypad 3.1.1
    - Librería MFRC522 1.5.1
- El software de la plataforma Arduino, Arduino IDE, puede ser descargado en su sitio oficial, https://www.arduino.cc/en/software. El software es escrito, desarrollado y soportado por Arduino.cc y la comunidad mundial de Arduino. Se distribuye bajo la Licencia Pública General de GNU, versión 2.

- El paquete de soporte para tarjetas SparkFun AVR Boards 1.1.13 puede ser descargado e instalado del gestor de tarjetas de Arduino IDE. Sparkfun distribuye su código, firmware y software bajo la licencia MIT.
- La librería Keypad puede ser descargada e instalada a través del gestor de librerías de Arduino IDE. La librería fue creada por Mark Stanley, Alexander Brevig, es mantenida por la comunidad en https://github.com/Chris–A/Keypad y se distribuye bajo la Licencia Pública General de GNU, versión 3.
- La librería MFRC522 puede ser descargada e instalada a través del gestor de librerías de Arduino IDE. La librería fue creada por Miguel Balboa, y es actualizada y mantenida por la comunidad de GitHub en https://github.com/makerspaceleiden/rfid. Se distribuye bajo la licencia Unlicense, licencia equivalente de dominio público.

4.3 Plan de gestión de la calidad aplicado durante la redacción del Proyecto

En ausencia de un plan de gestión dedicado, las siguientes acciones han sido desarrolladas para intentar garantizar la calidad del Proyecto.

- Revisión de los documentos del curso de refuerzo para la preparación del trabajo fin de grado dirigido a alumnos matriculados en la asignatura de TFG de los grados en ingeniería del ámbito industrial (2018), organizado por el Departamento de Ingeniería Mecánica y Construcción de la Escuela Superior de Tecnología y Ciencias Experimentales de la Universitat Jaume I.
- Comunicación continua con el Promotor.
- Estudio previo de normativa y reglamentación aplicable.
- Realización de prototipos electrónicos de las diferentes alternativas.
- Redacción del Proyecto según norma UNE 157001:2014: «Criterios generales para la elaboración formal de los documentos que constituyen un proyecto técnico».
- Establecimiento de un protocolo de revisión y corrección de errores.
- Revisión por parte del tutor académico del Proyecto.
- Revisión por parte de un Ingeniero Eléctrico de las alterntivas desarrolladas.
- Revisión por parte de un Ingeniero Industrial de la coherencia y cohesión de los desarrollos expuestos en el Proyecto.
- Revisión por parte de una Química de errores de tipografía y digitación.
- Evaluación por parte del Promotor de la funcionalidad de la solución proyectada.

4.4 Bibliografía

Analog Devices (s.f.). *ADXL345 3-Axis, ±2 g/±4 g/±8 g/±16 g Digital Accelerometer*. Ver. E.

Atmel Corporation (abr. de 2016). *ATmega16U4/32U4 DATASHEET*. Ver. 7766J.

Honeywell (feb. de 2015). *SS39ET/SS49E/SS59ET Series Linear Hall-effect Sensor ICs*. Ver. 4.

International Rectifier (nov. de 2003). *IRF520NPbF*.

Mean Well (17 de abr. de 2020). *25W Single Output Switching Power Supply RS-25 series*.

Microchip Technology (mayo de 2018). *MIC5219 500 mA Peak Output LDO Regulator*. Ver. A.

— (jun. de 2020). *ATmega4808/4809 Data Sheet*. Ver. B.

Monolithic Power Systems (1 de jul. de 2015). *MPM3610 – SYNCHRONOUS STEP-DOWN MODULE WITH INTEGRATED INDUCTOR*. Ver. 1.01.

Ningbo Songle Relay (s.f.). *SRD Series SUBMINATURE HIGH POWER RELAY 7A/10A/15A*.

NXP Semiconductors (27 de abr. de 2016). *MFRC522, Standard performance MIFARE and NTAG frontend*. Ver. 3.9.

OMRON (jun. de 2009). *Solid State Relay G3MB*.

— (2018). *G2R PCB Power Relay*.

MY-Semi (jul. de 2013). *MY9221 12-Channel LED Driver With Grayscale Adaptive Pulse Density Modulation Control*. Ver. 3.0.

TE Connectivity (13 de nov. de 2020). *Alcoswitch Gemini A, Toggle Switches, Double Pole - Double Throw Configuration (Pole-Throw), 1/4-40, Sealed Toggle Switch, Copper, Silver*.

Texas Instruments (feb. de 2020). *LM2596 SIMPLE SWITCHER Power Converter 150-kHz 3-A Step-Down Voltage Regulator*. Ver. E.

4.5 Otras referencias

Para la realización del Proyecto es de gran utilidad la documentación encontrada en las páginas electrónicas de los sistemas con los que se trabaja, así como la disponible en foros, blogs y otros canales de discusión oficiales de cada producto, siendo de mención especial:

- Arduino. https://www.arduino.cc/
- SparkFun Electronics. https://www.sparkfun.com/

- Shenzhen Keyes DIY Robot. https://www.keyestudio.com/
- Seeed Technology. https://www.seeedstudio.com/
- Mouser Electronics. https://www.mouser.es/
- E-Pulse Servicios de Internet. https://tienda.bricogeek.com/
- Solectro Shop. https://solectroshop.com/

Las imágenes de los dispositivos utilizados han sido obtenidas del sitio oficial de los respectivos fabricantes. En ningún caso la propiedad de las mismas pertenece al autor del Proyecto. Del mismo modo, los esquemas técnicos como los que describen las dimensiones o el conexionado de un producto han sido extraídos de las hojas de datos correspondientes.

Por otra parte, el Proyectista cuenta con la experiencia de un trabajo académico anterior, titulado «Proyecto de adaptación de un vehículo operado a distancia para experimentación subacuática multipropósito». La estructura general del Proyecto y algunas consideraciones en su desarrollo son similares a las del anterior.

Capítulo 5

Definiciones y abreviaturas

5.1 Definiciones

Armario: Sistema automático consistente en un gran cuadro eléctrico industrial que se abre mediante la introducción de una combinación en los pulsadores habilitados a tal efecto.

Centrifugadora: Sistema automático consistente en una vieja máquina de laboratorio que se abre mediante la introducción de una combinación en su teclado digital.

Bombilla: Sistema automático consistente en una fuente de luz que parpadea según una determinado patrón al presionar un pulsador de la sala.

Cajón: Sistema automático consistente en un compartimento que se abre golpeando su tapa según una combinación adecuada.

Puerta: Sistema automático consistente en un acceso entre dos partes de una estancia que se abre mediante un control de acceso con tarjeta sin contacto.

Compartimento: Sistema automático consistente en una compuerta oculta que se abre al introducir un tapón en el desagüe del lavabo de la sala.

Tuberías: Sistema automático consistente en un conjunto de válvulas situadas en tuberías hidráulicas interconectadas y en las que se simula la variación de un indicador analógico al actuar sobre las mismas.

Batería: Sistema automático consistente en un objeto que debe ser transportado de un lugar a otro para permitir la apertura de una compuerta.

Capítulo 6

Requisitos de diseño

6.1 Condicionantes de diseño

Aunque cada sistema automático presenta unas condiciones particulares para su diseño, en líneas generales las siguientes características son comunes a todos ellos:

- **Funcionalidad**: los automatismos deben comportarse según las necesidades establecidas por el Promotor, considerando los factores que puedan derivar en una condición no prevista en la programación que provoque un funcionamiento anómalo.
- **Fiabilidad**: los sistemas deben funcionar correctamente durante la hora de duración de los distintos juegos, de forma repetida varias veces al día. Un fallo en alguno de ellos podría suponer arruinar la experiencia de juego de los participantes, con la consiguiente pérdida económica y de reputación de la empresa.
- **Operabilidad**: las soluciones adoptadas deben poder ser puestas en marcha para un nuevo juego fácilmente, por un operador sin conocimientos técnicos específicos.
- **Integrabilidad**: en general, los dispositivos tienen que ser integrados en pequeños espacios. Además, el diseño de las soluciones tiene que prever su instalación en unas salas de las que no se facilitan planos físicos o eléctricos. Los sistemas deben ejecutar sus acciones mediante los actuadores que posee el cliente.
- **Resistencia**: el funcionamiento eléctrico o electrónico de los sistemas debe evitar ser alterado por golpes que puedan recibir por parte de los jugadores durante el transcurso del juego.
- **Modularidad**: los sistemas automáticos deben poder ser alterados, reprogramados o ampliados según sus necesidades, de un modo rápido y sin necesidad de rediseñarlos e implementarlos por completo.
- **Viabilidad económica**: los distintos automatismos deben ser relativamente asequibles.
- **Seguridad**: los dispositivos no deben poner en peligro la integridad física de las personas.

6.2 Sistemas requeridos

6.2.1 Apertura de un armario mediante una combinación en pulsadores

El Armario es un viejo cuadro eléctrico industrial, con tres grandes conmutadores y un panel con pilotos de colores. El Promotor desea modificar el armario para que el mismo, inicialmente bloqueado, pueda ser abierto tras haber activado los tres conmutadores, encendiendo las distintas luces durante la maniobra.

El Armario cuenta con una toma de corriente en su parte posterior.

6.2.2 Apertura de una centrifugadora mediante un código numérico

La Centrifugadora es una máquina de laboratorio retirada, que se quiere modificar para que su tapa superior se abra únicamente al introducir una combinación numérica de cuatro dígitos mediante un teclado externo.

La centrifugadora debe abrirse inmediatamente si la combinación tecleada es correcta. En el caso de que pasen tres segundos sin haber realizado ninguna pulsación, la máquina debe considerar que el usuario ha desistido en su intento de introducir el código. Si el código es erróneo, no debe ocurrir nada.

6.2.3 Parpadeo de una bombilla según una secuencia

La Bombilla consiste en una lámpara led instalada en un rincón de la estancia y apagar según una secuencia mediante un pulsador ubicado en otra parte de la sala. En el estado inicial, la bombilla debe encontrarse apagada, y volver a ese estado al finalizar la secuencia. El pulsador incorpora un led que debe permanecer encendido mientras la bombilla se encuentre reproduciendo la secuencia. Durante la secuencia, las actuaciones sobre el pulsador que puedan producirse no deben tener ningún efecto sobre el automatismo. Al finalizar, puede producirse de nuevo la activación de la bombilla.

En la instalación existente, la bombilla es operada con un conmutador normal.

6.2.4 Apertura de un cajón mediante una combinación de golpes

El Cajón es uno de los compartimentos de un mueble. El Promotor desea que el mismo permanezca cerrado durante el juego y únicamente se abra al actuar sobre el mueble con una determinada combinación de golpes. Una vez abierto, el cajón no debe volver a cerrarse hasta que el operario del juego no reinicie el automatismo.

Se dispone de una toma de corriente en el lugar donde se encuentra el mueble.

6.2.5 Control de acceso en una puerta

La Puerta es un paso entre dos partes de la sala que debe abrirse al acercar un objeto del juego a un lector inalámbrico instalado en la pared.

La puerta no debe volver a cerrarse hasta que así lo desee el operador del juego.

En las proximidades de la puerta se encuentra una caja de conexiones de la instalación eléctrica.

6.2.6 Apertura de una compuerta mediante la detección de un objeto

La Compuerta consiste en una trampilla ubicada en la pared que debe abrirse al introducir un tapón en el desagüe de un lavabo presente en la sala.

Del mismo modo que con otros mecanismos, se debe impedir el cierre de la Compuerta una vez abierta por primera vez.

6.2.7 Simulación hidráulica en tuberías

Las Tuberías son un conjunto de conductos con una serie de válvulas en instaladas en ellos. Los tubos están conectados entre sí, simulando un circuito que tiene una única salida. En ella hay instalado un indicador analógico de voltaje.

Se requiere que la aguja del voltímetro varíe la medida en función del estado de las válvulas de las tuberías. La detección de cada válvula debe ser digital, simulando que la misma se encuentra cerrada o abierta a partir de una posición media. La medida a indicar debe partir de un modelo matemático que tenga cierta coherencia con alguna característica de los fluidos.

6.2.8 Detección del traslado de un objeto entre dos localizaciones

Este sistema automático consiste en un objeto que se necesita trasladar de un lugar a otro de la sala sin que sufra movimientos bruscos. Para ello, el Promotor desea que sea instalado un indicador en el elemento en el se pueda ver una simulación de la energía que contiene el objeto. Este indicador debe decrementar su valor si el objeto es transportado de una forma poco cuidadosa.

El objeto debe poder activar un actuador al ser posicionado en la ubicación de recepción únicamente si mantiene un cierto nivel de energía. En caso de que esta se agote, el elemento tiene que poder reponerla al colocarlo en otra localización específica de la sala.

Por su relativo parecido con este componente, se bautiza a este automatismo como Batería.

6.3 Actuadores

El Promotor posee o instalará principalmente dos tipos de actuadores:

- Un electroimán genérico de 12 V, que a dicha tensión consume aproximadamente 200 mA.
- Una cerradura electrónica invertida de 12 V, que a dicha tensión consume aproximadamente 150 mA.

Como ya se ha mencionado, es un requisito que las soluciones proyectadas sean compatibles con estos dispositivos.

Capítulo 7

Análisis de soluciones

7.1 Consideraciones generales

7.1.1 Metodología y plataforma de desarrollo

Para realizar el desarrollo de los automatismos, el procedimiento a seguir es el prototipado de los mismos mediante la construcción de prototipos y pruebas experimentales. El Proyectista dispone de una serie de componentes de bajo coste que puede utiliza para elaborar y verificar los diseños, evaluando distintas alternativas y analizando las ventajas e inconvenientes de cada una de ellas. Con los prototipos funcionando, se proyectan las soluciones finales a ejecutar.

En la mayoría de los sistemas automáticos es necesario utilizar un microcontrolador. Existen muchas plataformas de desarrollo que incluyen microcontroladores de empresas como Microchip Technology, STMicroelectronics, Renesas Electronics, Infineon Technologies, Texas Instruments... Entre la infinidad de opciones disponibles, se determina emplear productos del proyecto Arduino. Esta es prácticamente la plataforma de referencia en la actualidad de prototipado rápido de dispositivos electrónicos. Arduino es un proyecto de código abierto tanto en el software como en el hardware, sus dispositivos son relativamente económicos, es multiplataforma y existe una gran comunidad de usuarios que contribuyen a su desarrollo y difusión compartiendo sus prototipos, ideas o conocimiento de forma libre. Si bien Arduino puede ser un proyecto sin un uso industrial extendido y considerado más en los sectores educativos o artísticos, las necesidades del Cliente hacen que esta sea una plataforma idónea. Los sistemas a implementar no son especialmente críticos y Arduino permite abstraer gran parte de los detalles de programación de un microcontrolador. Con esta plataforma se puede realizar fácilmente un diseño e implementación rápidos y una solución modular y escalable, de acuerdo con los requisitos del Proyecto.

Los desarrolladores de Arduino publican abiertamente los esquemas de diseño de sus placas. Sin embargo, Arduino es una marca registrada y solo los productos autorizados pueden distribuirse bajo este nombre. Massimo Banzi, uno de los fundadores de Arduino, clasifica en un artículo del blog oficial de Arduino las placas entre clónicas, derivadas, compatibles y falsificaciones.

- Oficiales: tienen soporte oficial en el entorno de desarrollo integrado de Arduino (Arduino IDE), siguen una diseño estandarizado, están documentadas en el sitio web, licencidadas para utilizar el nombre y el logotipo de Arduino y son construidas por fabricantes autorizados.
- Clónicas: reproducen los diseños de Arduino, normalmente sin introducir grandes cambios.
- Derivadas: son productos que derivan de los diseños de Arduino, pero aportan una innovación sustancial.
- Falsificaciones: intentan imitar el diseño de una placa oficial, infringiendo el uso de la marca Arduino.

El Proyectista dispone de varios modelos de placas de desarrollo oficiales de Arduino y placas compatibles con Arduino. Entre ellas, se decide realizar los prototipos principalmente con la placa compatible Pro Micro de SparkFun Electronics. Esta es una alternativa derivada de la placa oficial Arduino Pro Mini. Respecto a esta, su diseño sustituye el microcontrolador ATmega328P por un ATmega32U4, añadiendo conectividad USB y eliminando así la necesidad de disponer de un conversor UART a USB externo como los FT232R de FTDI Chip. Además, su tamaño de 33 mm x 18 mm la convierte en una de las placas más pequeñas compatibles con Arduino, sin renunciar a un gran surtido de patillas de entrada y salida.

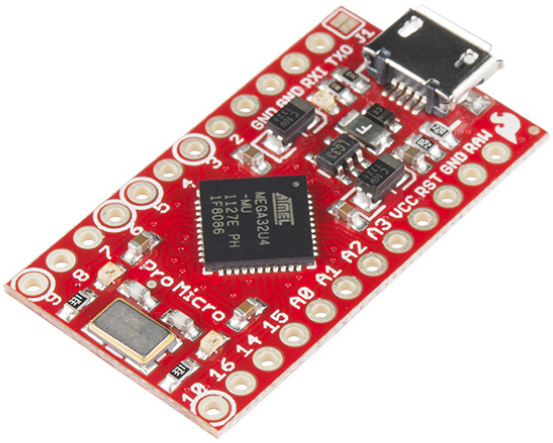


Figura 7.1: Placa de desarrollo Pro Micro de SparkFun

Por otra parte, los prototipos se realizarán a partir de una serie de módulos populares de bajo coste, ampliamente utilizados y cuyo funcionamiento ha sido documentado en múltiples montajes realizados con Arduino. Muchos de ellos son distribuidos sin tener un origen, fabricante o referencia concreto. El Proyectista tiene una extensa colección de módulos de este tipo y se basa en ellos para testear rápidamente las diferentes alternativas. Sin embargo, la selección de componentes utilizados en la solución proyectada se realiza garantizando la calidad y la correcta distribución de los mismos.

7.1.2 Fuente de alimentación

La mayoría de los sistemas automáticos a desarrollar requieren una fuente de alimentación conectada a la red eléctrica. Debido a que los actuadores de los que dispone el Promotor funcionan a 12 V, esta es la tensión de alimentación de los montajes. Teniendo en cuenta que el regulador del voltaje de la placa Pro Micro entrega 500 mA a 5 V (2,5 W) como máximo, que un actuador consume aproximadamente 200 mA (2,4 W) y que el resto de componentes no demandan una intensidad destacable, se preselecciona una fuente de alimentación conmutada de baja potencia (25 W), sobradamente capacitada para ejecutar todas las pruebas. No obstante, el consumo total de cada mecanismo se calcula teóricamente y se comprueba que no supere el límite que la fuente puede aportar. En los automatismos que por sus condiciones particulares requieran de otro tipo de alimentación, este es indicado.

Figura 7.2: Fuente de alimentación Mean Well RS-25-12

7.1.3 Módulo de control

De forma similar a la alimentación, la mayoría de mecanismos requieren realizar una acción mediante un único actuador que no puede ser controlado directamente mediante el microcontrolador. Por ello, es necesario utilizar un relé. Se utiliza para ello un módulo de control cuya presencia es recurrente en varios de los diseños.

Este módulo incorpora un selector de nivel de disparo, que permite configurar mediante un saltador la lógica de funcionamiento del relé. Además, incorpora un optoacoplador que aísla el control de la bobina del relé, elemento que puede producir picos de tensión, de la salida del microcontrolador. El relé incorporado es el modelo SRD-12VDC-SL-C de Ningbo Songle Relay. Este es un relé de un polo y doble vía que en corriente continua a 30 V soporta 10 A, por lo que realizará su función en los prototipos sin problemas. El consumo de esta versión, con una resistencia en la bobina de 400 Ohms, es de 30 mA a 12 V (0,36 W). Se puede esperar que el módulo completo presente un consumo similar.

Cualquier otro tipo de elemento de control del actuador que requiera un automatismo en particular es indicado en la descripción del mismo.

Como norma general, el control de los actuadores de los sistemas automáticos se intenta implementar con lógica inversa. Es decir, en el estado normal de cada automatismo el microcontrolador

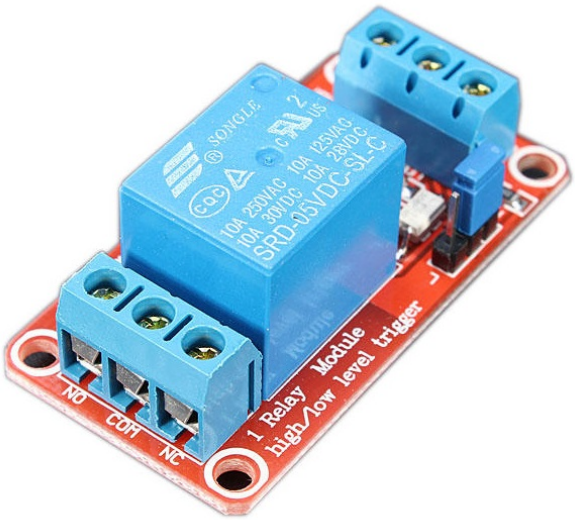


Figura 7.3: Módulo de control con relé

mantiene la señal de control activa. Cuando el jugador interactúa sobre el sistema y el actuador debe entrar en funcionamiento, se desactiva la señal en el microcontrolador. Este modo de funcionamiento permite que, si falla la alimentación de la placa de desarrollo o hay algun problema en la comunicación con el módulo de control, el mecanismo del juego quede abierto o se active.

Del mismo modo se planifica con atención, para todos los automatismos en los que es posible, la conexión del actuador final con el relé. Los modelos de cerraduras de los que dispone el Promotor son de funcionamiento invertido: se mantienen cerrados si son alimentados, se abren en caso contrario. Los electroimanes funcionan de una forma análoga, ya que al estar alimentados generan el campo magnético que mantiene el mecanismo cerrado.

El criterio de conexionado del relé es utilizar las vías como entrada y los polos como salida. Es decir, en este caso, la alimentación se administra a través del contacto normalmente abierto, y el actuador se conecta entre el contacto común y la masa. La masa puede ser conectada al contacto normalmente cerrado, aunque no se considera necesario y por simplificar el cableado esta acción no se realiza. Este modo de proceder está fundamentado en que, en diseños de electrónica digital más delicados, el Proyectista intenta tener conectado siempre el contacto común, ya sea a la alimentación o a la masa, evitando dejarlo en un estado indefinido (salvo durante el tiempo de conmutación, y en los diseños en los que no se hace uso de resistencias pull-up/pull-down). En conclusión, a pesar de la distinta configuración que pueda tener el relé, el uso que se le suele dar en los automatismos es el de un polo normalmente abierto (SPNO).

De esta forma, para que el mecanismo esté bloqueado o inactivo, la señal de control del microcontrolador debe estar conectada, la bobina entonces se encontrará excitada y el relé permitirá el paso de la alimentación al actuador.

Resumiendo:

- Si el cable de alimentación se corta, el mecanismo se abre.

- Si el cable de señal de control al relé se corta, el mecanismo se abre.
- Si la bobina del relé se quema y pierde su continuidad, el mecanismo se abre.

El inconveniente de este montaje es que la bobina del relé queda activa durante largos períodos de tiempo. No obstante, su consumo es despreciable frente a los que presentan los actuadores.

Por otra parte, la lógica de control del módulo es indiferente. Si el módulo activa el relé con una señal alta, si el cable se corta el relé quedará desactivado. Y del mismo modo, en un módulo con disparo a nivel bajo el relé también se desactiva, ya que si se corta el cable el módulo pierde la conexión a la masa del microcontrolador y la entrada se queda en nivel alto.

Este tipo de diseño es arriesgado desde el punto de vista de la funcionalidad de los mecanismos, ya que cualquier fallo provoca la activación del automatismo correspondiente (la desactivación de los actuadores que mantienen cerrados los mecanismos) y altera la experiencia de juego de los participantes. Sin embargo, es preferible poder detectar así cualquier tipo de anomalía para que pueda ser corregida y, sobre todo, garantizar la seguridad de las personas en el caso de los mecanismos pueden bloquear el paso.

7.1.4 Pulsadores de parada de emergencia

Siguiendo con el pensamiento en la seguridad, se proyecta la instalación de pulsadores de parada de emergencia en todos los automatismos que pueden franquear el paso de las personas. A pesar de que en principio no está prevista la actuación de más de un actuador en este tipo de montajes, se selecciona un pulsador DPST-NC en previsión de posibles ampliaciones futuras.

Figura 7.4: Pulsador de parada de emergencia MCA20L-V4E02Q6R de Multicomp Pro

Este pulsador es de 40 mm de diámetro y tiene una bombilla incandescente incorporada de 12 V, aunque no se considera necesario su uso.

Este tipo de elementos deben ser colocados de forma que puedan cortar la línea del actuador de cada montaje, deben ser accesibles y deben estar correctamente señalizados.

7.1.5 Cableado

El cableado utilizado en la alimentación de la placa de desarrollo y en la conexión de los actuadores de los prototipos es un producto dirigido principalmente hacia aplicaciones de audio, aunque cumple esta función sin problemas. Este cable tiene la ventaja de utilizar los colores rojo y negro, que son asociados (solo para este tipo de cableado) a tensión de 12 V y masa.

Figura 7.5: Cable Lexman 2x1,5 mm2

Los conectores que unen los distintos módulos son cables de puente, de cobre, de 2,54 mm, con las terminales cuadradas y de sección 18 AWG (0,82 mm2). Se utilizan cables de tipo macho-macho, macho-hembra o hembra-hembra, según sean requeridos.

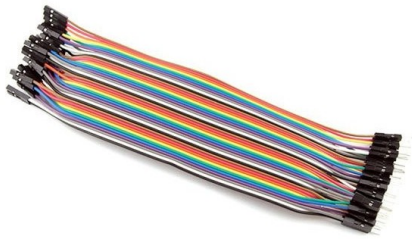


Figura 7.6: Cable puente

Además, para la elaboración de los prototipos se realizan diferentes soldaduras con estaño. Este es un aspecto que se requiere evitar al máximo en el diseño final, ya que resta modularidad e incrementa los tiempos de implementación de las soluciones.

7.2 Armario

El cuadro eléctrico cuenta con tres grandes interruptores rotativos de doble polo y una vía (DPST) alineados horizontalmente en la puerta derecha. Sobre cada interruptor se sitúa un piloto de señalización amarillo. Entre la fila de pilotos y la de interruptores existe un área despejada en la que el Promotor desea instalar ocho interruptores destinados a la introducción de la contraseña. En la puerta izquierda, existen dos filas de seis pilotos, rojos y verdes. Sin posibilidad de estudiar físicamente el armario, se dispone de un diagrama básico que muestra cómo están dispuestos aproximadamente estos elementos.

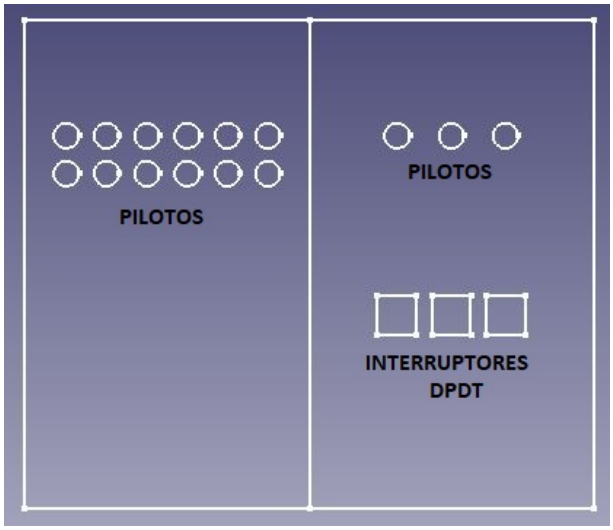


Figura 7.7: Módulo de control

Para permitir la apertura y el cierre de la puerta, el Promotor ha instalado una cerradura eléctrica de las características descritas anteriormente. Se comprueba experimentalmente el consumo de los pilotos a 12 V, determinando que este no es superior a los 20 mA. Como se desea encender el grupo de indicadores de la puerta izquierda simultáneamente, se dispone una conexión en paralelo de todos ellos. Este tipo de montaje requiere que la línea que los alimente soporte 240 mA.

El armario, como se ha indicado, es de baja tensión (230 V), por lo que se presupone que los interruptores presentes, a pesar de no conocer su modelo, pueden funcionar en este montaje a una intensidad muy inferior a su intensidad de diseño.

Con estas premisas y sabiendo que la cerradura consume 150 mA, la intensidad total del montaje, incluyendo los pilotos de los interruptores, es de 390 mA. Esta intensidad no es muy elevada, por lo que se decide prescindir de instalar un microcontrolador y controlar el encendido de los pilotos y la apertura del armario directamente con los los interruptores en los que se introduce la secuencia.

Al ser los tres interruptores del armario de tipo DPST se puede separar la lógica del circuito en dos líneas, y hacer que una se encargue de encender los indicadores individuales y otra de la actuación final. Es posible incluso controlar cada piloto asociado con su interruptor correspondiente, pero el Promotor no desea este funcionamiento. De la forma especificada, el segundo piloto solo se enciende si el primero lo ha hecho, y del mismo modo, el resto de pilotos únicamente se encienden si los dos primeros han sido activados. Considerando este funcionamiento y a pesar de no ser necesario,

se unen las líneas de salida de los tres interruptores del armario. De este modo se reparte entre ambos polos la poca carga que deben soportar y además se constituye un sistema redundante, en el que si una de las líneas falla, el circuito puede seguir funcionando con normalidad por la otra.

Esta solución es relativamente sencilla y no se considera la realización de un prototipo. Se seleccionan directamente el tipo de interruptores en los que se debe introducir la secuencia secreta del juego. Existe una amplia variedad de de interruptores con diferentes configuraciones que se le presenta al Promotor, optando este por interruptores de palanca redondeada y corta, de tres posiciones. Para el cometido se seleccionan los interruptores A211SYCQ04 de TE Connectivity, de un polo y tres vías (SP3T). A pesar de que se encuentran disponibles otras alternativas SP3T en las que la conexión se realiza mediante conectores rápidos FASTON, estas son notablemente más costosas, por lo que se requerirá soldar los interruptores seleccionados al circuito.

Figura 7.8: Interruptor de palanca A211SYCQ04 de TE Connectivity

Con estos interruptores, que soportan hasta 2 A de corriente continua, se podría controlar perfectamente la activación de la cerradura de forma directa. No obstante, con la motivación de seguir unas buenas prácticas y debido a que no supone un sobrecoste excesivo, se decide separar el circuito de control del circuito de actuación mediante un relé. En este caso, al no contar con un microcontrolador en el montaje, se prescinde del uso de un módulo y se selecciona un relé para su conexión directa. Por el mismo motivo se descarta la instalación de un diodo de retorno.

El relé seleccionado para tal cometido es el G2R-1-T DC12 de Omron, una alternativa al popular G2R-2 en su versión de un único polo normalmente abierto (SPDT - 1c) con terminales FASTON de conexión rápida. Se recuerda que las cerraduras empleadas por el Promotor permanecen cerradas cuando se alimentan, abriéndose al cesar el suministro de energía. El relé no queda excitado hasta que todos los interruptores están en la posición correcta y se debe abrir la puerta, por lo que el contacto normalmente cerrado debe ser el que transmita la alimentación a la cerradura.

La conexión de los pilotos se realiza tras los interruptores correspondientes responsables de su encendido.

Figura 7.9: Relé G2R-1-T DC12 de Omron

7.3 Centrifugadora

A diferencia de lo que ocurre con el resto de montajes, a la centrifugadora de laboratorio se tiene acceso, ya que así puede ser estudiada y modificada convenientemente. La máquina presenta en su panel de control un temporizador mecánico para ajustar el tiempo de rotación, un selector de velocidad, un indicador de revoluciones por minuto, un botón de apertura de la tapa y tres indicadores del estado de la tapa, alimentación de la máquina y movimiento del rotor. El Promotor no desea utilizar ninguno de ellos, simplemente requiere abrir la máquina mediante un código numérico.

Figura 7.10: Centrifugadora de laboratorio

La centrifugadora presenta en su interior tres placas de circuito impreso: una alberga cuatro indicadores de ocho segmentos que componen la pantalla de las revoluciones, otra contiene la circuitería de alimentación de la máquina y control del motor y la última es la placa principal de control. En la parte de alimentación se puede distinguir el circuito para adaptar el voltaje de alimentación de red al nivel de la placa y la parte dedicada al control, con un controlador en un zócalo DIP20 y las conexiones de los indicadores y controles del panel. Se aprecia que el actuador que abre y cierra la tapa es un solenoide y de forma experimental se determina que se debe aplicar un pulso de de 5 V y 50 ms de duración para conseguir efectuar esta operación.

Para introducir la combinación numérica se decide emplear un teclado matricial de membrana popular y económico.

Figura 7.11: Teclado matricial de membrana

El teclado presenta cuatro líneas eléctricas horizontales y cuatro líneas verticales. Las teclas se ubican en cada encrucijada de las líneas. Al pulsar una tecla, se conectan las líneas horizontal y vertical situadas tras la misma, permitiendo que un microcontrolador detecte la pulsación. Proyectos como la librería Keypad, desarrollada por Mark Stanley y Alexander Brevig, permiten abstraer el funcionamiento del hardware y obtener directamente los carácteres de las teclas pulsadas.

Con esta información, se plantean tres posibles modificaciones:

- Sustituir toda la electrónica por una solución basada en Arduino.
- Aprovechar la electrónica existente.
- Sustituir el microcontrolador.

La solución más elegante, a criterio del Proyectista, es sustituir el microcontrolador presente en la placa de circuito impreso por uno del mismo modelo o compatible, y conectar el teclado a las patillas no utilizadas mediante algún tipo de controlador externo que permita una conexión SPI o I2C. Esta solución, no obstante, requiere un notable trabajo adicional para programar en otra plataforma, por lo que se descarta.

Sin embargo, sí que es posible aprovechar parte de la circuitería existente. Se retira el microcontrolador y se realiza el prototipo de la modificación con una placa clónica de Arduino UNO R3 junto con un relé que actúe sobre la patilla responsable de controlar el solenoide, aprovechando la alimentación de la máquina.

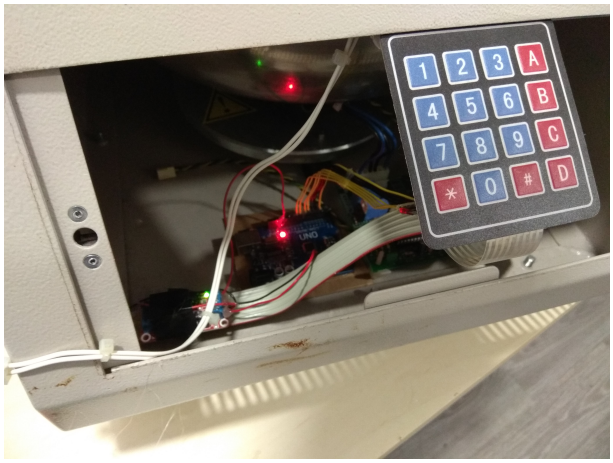


Figura 7.12: Prototipo de la modificación de la centrifugadora

7.4 Bombilla

La Bombilla que precisa un control se encuentra integrada en la instalación eléctrica, y puede encenderse y apagarse mediante su correspondiente interruptor. Para intentar modificar al mínimo posible la instalación eléctrica, el control se realiza mediante un relé de estado sólido, un componente apto para controlar cargas de corriente alterna de 230 V y que puede ser instalado en sustitución del interruptor. Esta opción es preferida frente a un relé mecánico principalmente por dos motivos: el prematuro desgaste que sufriría este en un automatismo que requiera abrirlo y cerrarlo repetidamente y el sonido producido, que desvelaría el código del juego oculto sin que los jugadores tengan que fijar su atención en la bombilla.

El relé de estado sólido controlará una carga de 230 V, por lo que sería recomendable aislar galvánicamente la red de dicha carga de la circuitería de control. Se selecciona con este fin un módulo que integra el relé y un optoacoplador.

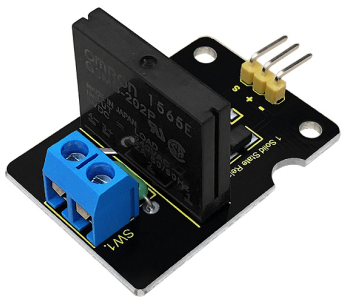


Figura 7.13: Módulo KS0263 de Keyestudio

El pulsador es un mecanismo que activa un final de carrera. Este cuenta con un contacto común, uno normalmente abierto y otro normalmente cerrado.

Figura 7.14: Pulsador gigante

El led del pulsador se controla con una señal de 12 V, por lo que se utiliza un módulo con un MOSFET. El pulsador se conecta a una entrada digital. Es importante completar este montaje activando la resistencia interna de pull-up del Arduino en la patilla correspondiente, para evitar tener falsas interpretaciones de pulsaciones.

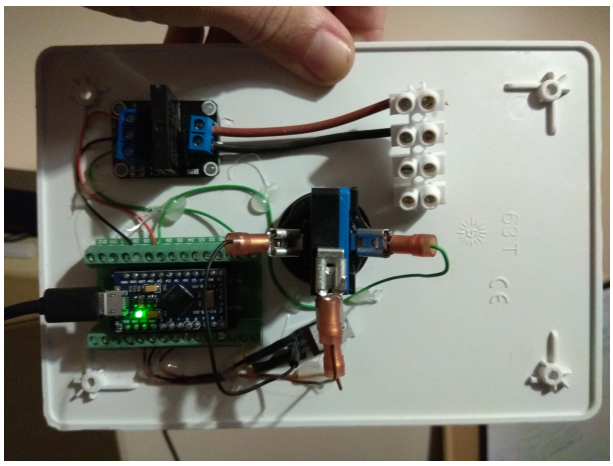


Figura 7.15: Prototipo del automatismo de la bombilla

7.5 Cajón

La correcta detección de los golpes, en un primer momento, no parece un objetivo fácil. Inicialmente, se plantean hasta cinco opciones que podrían realizar esta tarea:

- Un sensor de vibración digital
- Un sensor piezoeléctrico
- Un acelerómetro
- Un micrófono
- Una combinación de los sensores anteriores

Las alternativas se presentan en el orden lógico que, a criterio del Proyectista, podrían tener un funcionamiento más eficaz.

El sensor piezoeléctrico consta de un transductor que se fija sobre una superficie y que genera un voltaje al recibir vibraciones. Se cree que puede ser la solución ideal, ya que quedaría fijado perfectamente al baúl y se podría realizar una calibración en el microcontrolador al ser una señal analógica.

El sensor de vibración digital también podría funcionar adecuadamente. En este caso, el transductor es una pequeña esfera metálica situada en el interior de un cilindro con dos contactos que, al vibrar el conjunto, la esfera cortocircuita. Podría ser una buena solución, aunque al ser de naturaleza digital, la señal obtenida para cada golpe sería un conjunto de pulsos indeterminado.

El acelerómetro y el micrófono son otras alternativas para detectar los golpes. Sin embargo, el Proyectista duda del buen desempeño de las mismas. Las deformaciones al golpear una superficie no son relativamente grandes y un acelerómetro que detecte con buena precisión esos movimientos no suele ser económico. El micrófono supone medir las ondas mecánicas producidas por los golpes en otro medio, el aire, de un modo más indirecto, por lo que quizá no sea tan fiable.

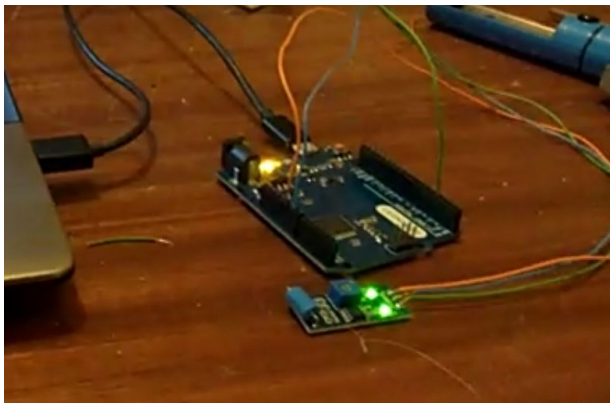


Figura 7.16: Prueba del sensor de vibración digital

Con estas consideraciones, la intención inicial del Proyectista es probar varias de las opciones. El primer montaje, en el que se prueba el sensor de vibración digital, funciona de forma aceptable. Sin embargo la segunda prueba, con el transductor piezoeléctrico, presenta mejores resultados: detecta cualquier pequeño golpe que se produce en una superficie y produce una señal relativamente limpia. Por ello, se decide realizar el montaje con este transductor.

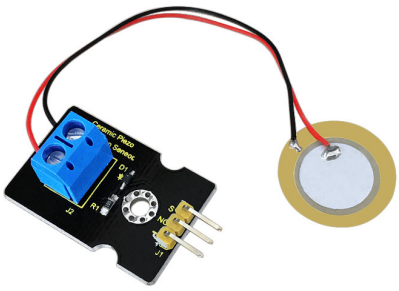


Figura 7.17: Módulo de sensor piezoeléctrico KS0272 de Keyestudio

7.6 Puerta

En la Puerta se debe implementar un control de acceso. La puerta dispone de una cerradura electrónica. Se requiere que esta pueda abrirse al acercar a un lector un objeto de la sala determinado.

Existen múltiples sistemas de control de acceso en el mercado. No obstante, atendiendo a la posible futura integración de todos los mecanismos en una misma plataforma de supervisión y control, se decide implementar el mismo también mediante Arduino. Con ello, si se desea en el futuro se podrá ampliar la funcionalidad para que el dispositivo pueda comunicarse del modo que se estime conveniente.

Por sus características, el control de acceso claramente apunta hacia una solución de tipo RFID / NFC. Se selecciona uno de los módulos de este tipo ampliamente utilizados con Arduino, que contiene el lector y grabador sin contacto MFRC522 de NXP Semiconductors.

Este automatismo presenta una particularidad: el módulo funciona a 3,3 V. Se plantea emplear un conversor de nivel bidireccional para conectar las señales de comunicación entre el módulo y la placa de desarrollo, y así evitar utilizar otro modelo diferente. Sin embargo, instalar dispositivos adicionales en líneas de comunicación no suele ser una gran idea, y tras experimentar diversos problemas con un montaje de pruebas, se decide emplear una placa Pro Micro de 3,3 V.

Esta elección conlleva dos inconvenientes adicionales: los módulos de relé utilizados en el resto de mecanismos no funcionan correctamente con una señal de 3,3 V, y la diferencia de tensión entre la que suministra la fuente de alimentación (12 V) y la de funcionamiento del Pro Micro (3,3 V) es considerablemente elevada, lo que hace que el regulador integrado en la placa se caliente en exceso. Si bien es posible que pueda funcionar dentro de sus límites, se decide decrementar la tensión a un nivel intermedio antes de alimentar a la placa, mediante un segundo regulador. En cuanto al relé, se modifica una de las resistencias del módulo para que el fototransistor que

Figura 7.18: Módulo lector/grabador de tarjetas sin contacto

controla la bobina permita el paso de la misma intensidad que cuando el módulo es controlado a 5 V.

El regulador utilizado para realizar el prototipo es un LM7805, con una tensión de salida de 5 V. No obstante, para la solución definitiva se selecciona un módulo convertidor reductor basado en el LM2596, del que se espera que sea más eficiente debido a su funcionamiento por conmutación.

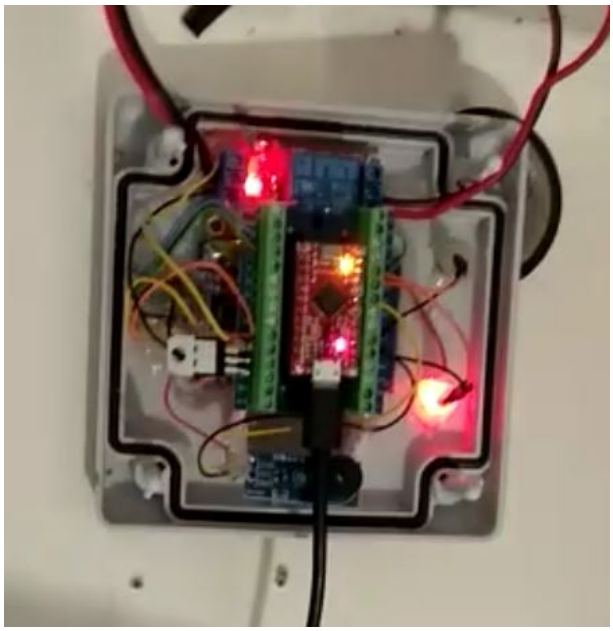


Figura 7.19: Prototipo del control de acceso de la puerta

Además, se proponen dos características adicionales que aportan valor y realismo a la solución: la instalación de un par de ledes que indiquen si la puerta está abierta o cerrada, y la conexión

de un zumbador para que emita un sonido al detectar el objeto. Esta sugerencia es de agrado del Promotor.

7.7 Compuerta

Según indica el Promotor, la distancia entre el sanitario y la compuerta será de aproximadamente un metro, y habrá una toma de corriente y espacio suficiente para ubicar los componentes del automatismo. Para realizar el montaje mecánico el Promotor ha instalado un electroimán y una cerradura. Esto es debido a que la compuerta se abre impulsada por un resorte: la cerradura no funciona si el bulón soporta una fuerza lateral, por lo que necesita que el electroimán esté actuando, y el electroimán por sí mismo no es capaz de mantener la compuerta cerrada si un jugador la fuerza.

Para poder abrir la compuerta, este montaje requiere que el bulón de la cerradura se retraiga antes de que se desactive el electroimán. Por ello se utilizarán dos relés. Mediante pruebas experimentales se determina que es necesario un tiempo de aproximadamente 200 ms para que la cerradura se abra completamente antes de que el electroimán pueda dejar de ser alimentado.

En cuanto al sensor a utilizar en este automatismo, se plantean varias posibilidades:

- Un lector RFID/NFC
- Un sensor de luz (fotorresistencia, fototransistor, fotodiodo...)
- Un sensor de campo magnético

La similitud con el automatismo anterior (acercar un objeto a una posición) sugiere el empleo de nuevo de un RFID. Sin embargo, en este caso el espacio es muy limitado: el mecanismo debe quedar oculto en el desagüe del lavabo y este tipo de módulos suelen necesitar un área plana considerable. Por ello, se descarta esta opción.

También es posible realizar el automatismo mediante un sensor de luz instalado en el desagüe. Al colocar el tapón el cambio de luz puede ser detectado. No obstante, esta opción queda rápidamente descartada, ya que cualquier objeto, o simplemente la mano de un jugador, podría bloquear la luz y abrir así la compuerta.

Queda la tercera opción, que parece la más adecuada: se instala en el tapón un pequeño imán y se coloca en el desagüe un sensor de efecto Hall. Así al colocar el tapón, este puede ser detectado. El transductor es diminuto, por lo que puede ser escondido adecuadamente en el desagüe.

Se escoge el modelo de imán de neodimio más pequeño del mercado para instalarlo en el tapón, con el fin de evitar que el mismo pueda interferir o dañar dispositivos que puedan llevar los jugadores, como relojes electrónicos o marcapasos.

En relación al sensor, se selecciona un popular módulo que incluye un sensor SS49E de Honeywell. Sin embargo, para instalarlo adecuadamente se requerirá una pequeña modificación: se debe desoldar el transductor del módulo y volverlo a soldar mediante tres cables. Esto posibilita que se pueda colocar el sensor en la cabeza del desagüe, y que el resto del módulo pueda ser escondido en el sifón. Los cables deben ser preferentemente negros y estar colocados estratégicamente para evitar que puedan ser vistos por los jugadores.

Es importante destacar que la distancia de los cables que unen el sensor y el módulo debe ser la mínima, ya que este módulo incluye el amplificador, y un cable de señal largo con una señal débil es prácticamente una antena de interferencias. Es por ello que se debe realizar el montaje del modo descrito.

Aunque se considera implementar algún tipo de elemento que permita la comunicación a larga distancia, ya que el lavabo está separado aproximadamente un metro del lugar en el que se encontrará la placa de desarrollo, se realizan pruebas experimentales en las que no se puede apreciar ningún tipo de problema leyendo el sensor, por lo que se descarta complicar el montaje.

7.8 Tuberías

El sistema automático de las tuberías debe detectar la apertura y cierre de cinco válvulas, y según una función especificada por el Promotor, mostrar un resultado en un indicador de medida analógica. La detección debe ser binaria. En el mercado existen válvulas que informan de su estado. Sin embargo, estas soluciones no son económicas. Por ello, el Promotor utilizará válvulas y tuberías de PVC.

Para detectar el la apertura y el cierre de las válvulas, se propone utilizar una fuente de luz y un transductor fotoeléctrico, situando cada elemento en un lado de la válvula a modo de optoacoplador casero. Así, al abrir la válvula se permite el paso de la luz, y al cerrarla, la misma queda bloqueada. Este montaje también permite capturar una señal analógica, y así poder tener cierta flexibilidad para establecer umbrales de detección.

Se consideran tres tipos de fuente de luz posibles:

- Una bombilla
- Un led
- Un láser

Una bombilla incandescente de bajo voltaje sería una fuente de luz abundante. Sin embargo, el consumo es elevado respecto al resto de opciones, desprende calor y su vida útil es menor.

Un led estándar no emitiría tanta luz como una bombilla. Sin embargo, el consumo sería menor, pudiendo ser directamente controlado por la placa Arduino.

El láser produciría una fuente de luz potente y de bajo consumo. Su inconveniente es su direccionalidad, por lo que se debería diseñar un buen soporte tanto para el láser como para el transductor del otro lado de la válvula, alinearlos bien y evitar que se desajusten con el uso del mecanismo.

Tras evaluar las distintas alternativas, se decide realizar el montaje con el led. A pesar de su baja emisión de luz respecto al resto de opciones, se confía en que sea suficiente y que un transductor con el rango adecuado no tenga problemas en apreciar esta intensidad. Se podría utilizar un led RGB para poder experimentar si existe alguna variación en el sensor frente a distintos longitudes de onda, pero finalmente se selecciona un módulo con led blanco.

Para seleccionar el sensor se consideran las siguientes opciones:

- Una fotorresistencia

Figura 7.20: Módulo de led KS0016 de Keyestudio

- Un fototransistor
- Un fotodiodo
- Una célula fotoeléctrica

La fotocélula queda descartada, debido a que el montaje debe quedar oculto dentro de la tubería. Un fotodiodo tiene una elevada velocidad de respuesta, pero no puede realizar una lectura analógica del nivel de luz detectado.

7.9 Batería

Este automatismo difiere notablemente del resto y supone todo un reto de diseño. Existe un elemento móvil, que debe indicar su estado. Este aspecto involucra el uso de un acumulador, que deberá ser protegido adecuadamente para evitar su sobredescarga. Y, por otra parte, hay dos zonas especiales con las que el objeto debe interactuar. Si fuera necesario instalar un microcontrolador en cada una de ellas, prácticamente triplicaría el precio de cualquier otro mecanismo. Además, en la zona de destino no solo se requiere realizar la detección de posición, sino también comunicar su estado de carga para que el automatismo actúe o no.

Se considera la posibilidad de instalar un lector RFID en el objeto móvil, y disponer dos etiquetas en la zona. Sin embargo, y teniendo en cuenta que las baterías suelen tener dos polos, se plantea realizar algún tipo de montaje que incluya el uso de dos contactos.

Se decide utilizar una placa de desarrollo Arduino MKR WiFi 1010, ya que tiene un circuito integrado gestor de energía que permite conectar una batería y realizar un uso adecuado de ella. Con este circuito la batería puede ser cargada y descargada con seguridad, evitando sobretensiones y sobredescargas.

Figura 7.21: Placa de desarrollo Arduino MKR WiFi 1010

Como este sistema automático está destinado a ser móvil, se debe prestar especial atención a la construcción del mismo. Por ello se decide emplear el escudo Arduino MKR Connector Carrier y módulos de la familia Grove de SeeedStudio. El módulo de barra de ledes de esta colección es muy adecuado para realizar el indicador de carga, y se dispone también de un amplio rango de módulos con acelerómetros de distintas sensibilidades, entre los que se elige el ADXL345 por su popularidad y precio. Con todo ello se monta un prototipo que detecta los cambios de aceleración y simula en el indicador de barra de ledes una pérdida de energía.

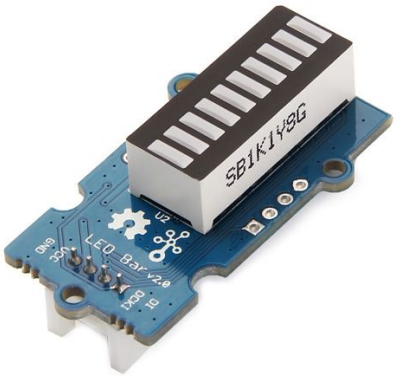


Figura 7.22: Módulo de barra de ledes de SeeedStudio

Para interactuar con las zonas de carga y descarga, se decide emplear únicamente dos contactos del la placa Arduino con el fin de hacer la simulación de la batería más realista. Uno de ellos necesariamente tiene que ser una masa común entre los circuitos, por lo que solo queda una patilla para detectar la zona de carga y actuar sobre la de descarga. Se utiliza un módulo de relé de disparo bajo, por lo que la entrada del módulo se utilizar para leer una señal de 5 V y certificar que la batería está en la zona de descarga. Para la zona de carga, simplemente se cortocircuitan los dos terminales. Así, la estrategia consiste en leer una entrada analógica constantemente y averiguar si está en la zona de carga (0 V), la zona de descarga (5 V) o en el aire (ruido próximo a 0 V).

Figura 7.23: Módulo acelerómetro ADXL345 de SeeedStudio

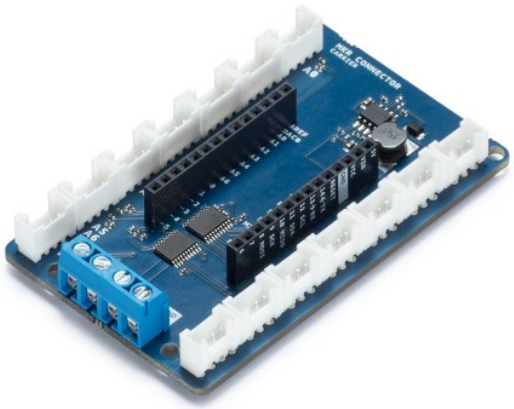


Figura 7.24: Escudo de conectores del sistema Grove

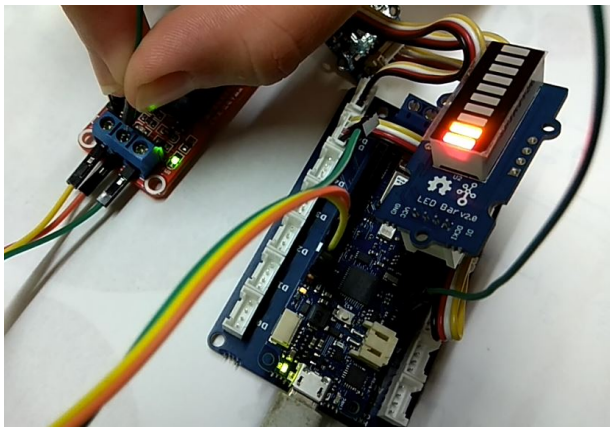


Figura 7.25: Prueba de la Batería en zona de carga

Capítulo 8

Resultados finales

8.1 Plataforma de implementación

A pesar de que en los prototipos se han empleado varios tipos de placas de desarrollo, la solución se proyecta con placas oficiales de Arduino.

El equipo de Arduino ha desarrollado una amplia gama de placas de desarrollo en las que hay versiones de reducido tamaño. Los productos más básicos disponibles son las placas Arduino Nano, Arduino Micro y Arduino Nano Every. En el segmento profesional, existen la familia Portenta (orientada a un uso industrial de alto rendimiento), la familia MKR (orientada al internet de las cosas con diversas tecnologías de comunicación) y la familia Nano (que incorpora tecnología Bluetooth y diversos sensores integrados en sus placas).

En general, el tipo de automatismos a implementar no requiere características avanzadas ni una potencia extraordinaria. Por ello se selecciona la placa Arduino Nano Every, moderna y económica, consistente en una versión actualizada de la placa Arduino Nano clásica.

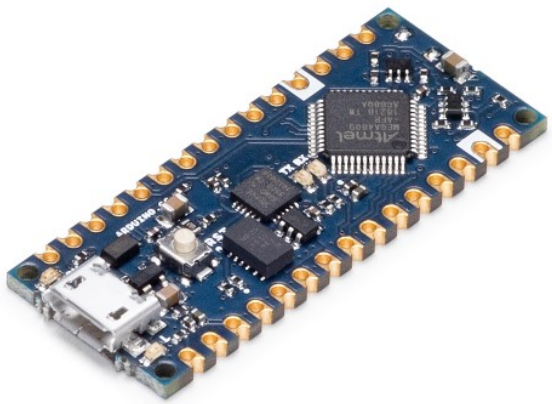


Figura 8.1: Placa de desarrollo Arduino Nano Every

8.2 Armario

En el Plano 1 se representa la solución desarrollada para el Armario.

La combinación exacta con la que hay que conetcar los interruptores de palanca debe ser comunicada durante la instalación por el Promotor.

8.3 Centrifugadora

En el Plano 2 se representa la solución desarrollada para la Centrifugadora. Utiliza como placa controladora un Arduino Nano Every.

El código exacto que hay que introducir en el teclado para poder abrir la tapa debe ser comunicado durante el montaje por el Promotor.

El código se debe introducir en la siguiente línea, respetando el formato:

Configuración de Centrifugadora

```
1   char code[] = {'0', '1', '2', '3', 'A'};
```

El código completo se encuentra en el Anexo «Programas de Arduino».

8.4 Bombilla

En el Plano 3 se representa la solución desarrollada para la Bombilla. Utiliza como placa controladora un Arduino Nano Every.

La secuencia exacta que seguirá la bombilla al parpadear debe ser comunicada durante el montaje por el Promotor.

La secuencia se debe introducir en la siguiente línea, respetando el formato:

Configuración secuencia Bombilla

```
1   unsigned int code[] = {5, 2, 4, 1, 3};
```

El tiempo que permanece la bombilla encendida y apagada y el tiempo mínimo de espera entre dos secuencias también puede ser configurado:

Configuración Bombilla

```
1   #define TIME_HIGH 400
2   #define TIME_LOW 400
3   #define TIME_WAIT 2000
```

El código completo se encuentra en el Anexo «Programas de Arduino».

8.5 Cajón

En el Plano 4 se representa la solución desarrollada para el Cajón. Utiliza como placa controladora un Arduino Nano Every.

La combinación de golpes que hay que darle a la tapa del cajón para que este sea abierto será comunicada durante el montaje por el Promotor. Esta es una combinación basada en los tiempos de espera entre golpes.

La secuencia se debe introducir en la siguiente línea, respetando el formato:

Configuración de Cajón

```
1     unsigned long secret[] = {1000, 1500, 2500, 3500, 4000};
2
```

Existen otros parámetros que pueden ser configurados como el nivel de intensidad mínimo para que un golpe sea considerado como tal, el tiempo durante el que se ignora posibles rebotes de la señal, la tolerancia máxima para aceptar un golpe como válido o el tiempo en el que se considera que un jugador ha desistido de su intento de introducir el código:

Configuración de Cajón

```
1     const int THRESHOLD_PIEZO = 300;
2     const int DELAY_FILTER = 50; // ms
3     const int TOLERANCE = 200; // ms
4     const int TIMEOUT = 3000; // ms
5
```

El código completo se encuentra en el Anexo «Programas de Arduino».

8.6 Puerta

En el Plano 5 se representa la solución desarrollada para la Puerta. Utiliza como placa controladora un SparkFun Pro Micro.

Durante la instalación se debe configurar una tarjeta maestra, una tarjeta de reseteo y una etiqueta RFID esclava para los jugadores. La tarjeta maestra permite agregar tags de jugadores, la tarjeta de reseteo permite abrir y cerrar la puerta y la etiqueta únicamente permite abrirla.

La tarjeta maestra se introduce en el sistema al borrar por completo todo con el botón previsto para ello. Este botón sólo es utilizado durante el montaje y su instalación es opcional. Las etiquetas esclavas quedan registradas en el sistema al pasar antes que ellas la tarjeta maestra.

El identificador de la tarjeta de reseteo se debe introducir en las siguientes líneas:

Configuración de Puerta

```
resetCard[0] = 39;
resetCard[1] = 1861;
resetCard[2] = 1042;
resetCard[3] = 4;
```

El código completo se encuentra en el Anexo «Programas de Arduino».

8.7 Compuerta

En el Plano 6 se representa la solución desarrollada para la Puerta. Utiliza como placa controladora un Arduino Nano Every.

En este automatismo hay que realizar una buena calibración, ya que por seguridad los imanes que han sido seleccionados no son demasiado potentes y el rango de valores que detecta el sensor es relativamente pequeño. El retraso con el que se abre la puerta y el número de muestras de filtrado de la señal analógica también pueden ser configurados:

Configuración de Compuerta

```
#define LOW_TRIGGER_VALUE 335
#define HIGH_TRIGGER_VALUE 570
#define DOOR_DELAY 1000
#define FILTER_SAMPLES 50
```

El código completo se encuentra en el Anexo «Programas de Arduino».

8.8 Tuberías

En el Plano 7 se representa la solución desarrollada para las Tuberías. Utiliza como placa controladora un Arduino Nano Every.

El peso que tiene cada válvula sobre la medida final puede ser configurado en las siguientes líneas:

Configuración de Tuberías

```
const int COEF_VALVE_1 = 5;
const int COEF_VALVE_2 = 2;
const int COEF_VALVE_3 = 3;
const int COEF_VALVE_4 = 1;
const int COEF_VALVE_5 = 4;
```

Los valores máximos y mínimos que captan los sensores de cada válvula deben ser calibrados para un correcto funcionamiento:

Calibración Tuberías

```
1   const int MAX_VALVE_1 = 800;
2   const int MAX_VALVE_2 = 800;
3   const int MAX_VALVE_3 = 800;
4   const int MAX_VALVE_4 = 800;
5   const int MAX_VALVE_5 = 800;
6
7   const int MIN_VALVE_1 = 200;
8   const int MIN_VALVE_2 = 200;
9   const int MIN_VALVE_3 = 200;
10  const int MIN_VALVE_4 = 200;
11  const int MIN_VALVE_5 = 200;
```

El código completo se encuentra en el Anexo «Programas de Arduino».

8.9 Batería

En el Plano 8 se representa la solución desarrollada para la Batería. Utiliza como placa controladora un Arduino MKR WiFi 1010.

La sensibilidad de la batería, los ratios de carga y descarga y el nivel mínimo para que pueda activar el relé en la zona de descarga son opciones que pueden ser configuradas para ajustar la dificultad del juego:

Configuración de Batería

```
1     const float chargeRate = 0.3;
2     const float dischargeConstant = 1.0;
3     const int chargeThreshold = 3;
4     const int loadThreshold = 1000;
5     const float activeThreshold = 10.0;
6     const float delayRelay = 300;
7
```

El código completo se encuentra en el Anexo «Programas de Arduino».

Capítulo 9

Planificación

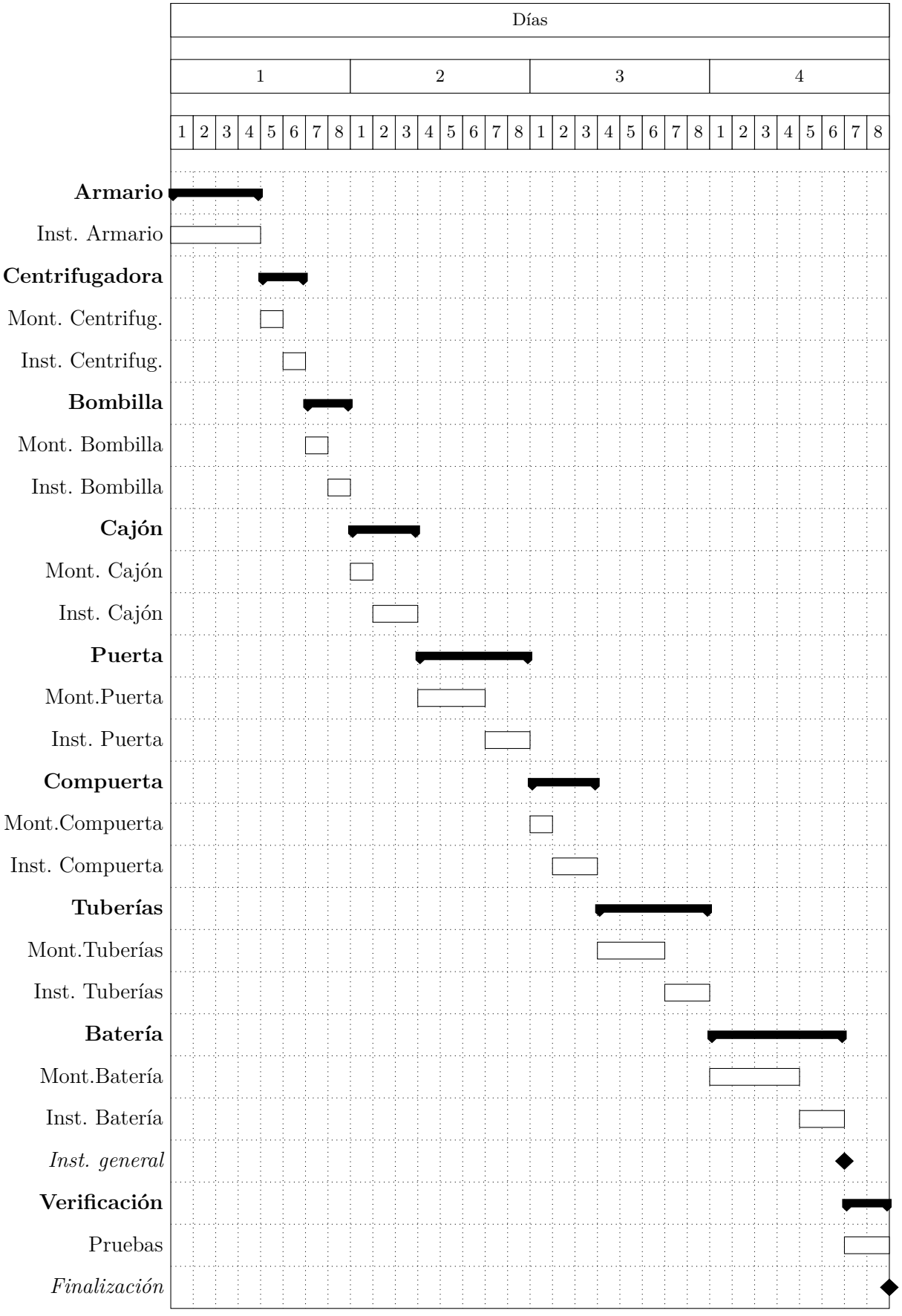
Se consideran tres tipos de tarea a realizar sobre los diversos automatismos:

Montaje: Preparación de los componentes electrónicos necesarios, realización de conexionado entre los mismos, programación del dispositivo y prueba de funcionamiento.

Instalación: Integración del sistema en su ubicación en la sala de juego, calibración y verificación de funcionamiento.

Verificación: Prueba general exhaustiva de todos los dispositivos en presencia del Promotor.

La ejecución del Proyecto la realiza un único instalador.


Días
1
2
3
4
1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
Armario
Inst. Armario
Centrifugadora
Mont. Centrifug.
Inst. Centrifug.
Bombilla
Mont. Bombilla
Inst. Bombilla
Cajón
Mont. Cajón
Inst. Cajón
Puerta
Mont.Puerta
Inst. Puerta
Compuerta
Mont.Compuerta
Inst. Compuerta
Tuberías
Mont.Tuberías
Inst. Tuberías
Batería
Mont.Batería
Inst. Batería
Inst. general
Verificación
Pruebas
Finalización

Capítulo 10

Orden de prioridad entre los documentos

1. Planos
2. Presupuesto
3. Memoria

Parte III

Anexos

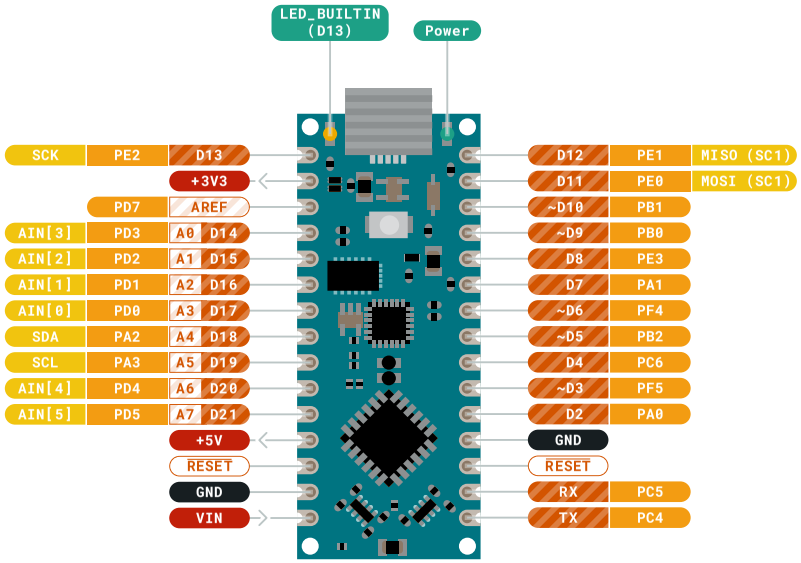
Capítulo 1

Disposiciones de patillas

ARDUINO

ARDUINO NANO EVERY

STORE.ARDUINO.CC/NANO-EVERY



- Ground
- Power
- LED
- Internal Pin
- SWD Pin
- Digital Pin
- Analog Pin
- Other Pin
- Microcontroller's Port
- Default

- **MAXIMUM** current per pin 40mA, 20mA recommended
- **MAXIMUM** current 200mA for the entire package
- The total current of each port power group **should not exceed** 100mA

VIN 7-21 V input to the board.

ARDUINO.CC

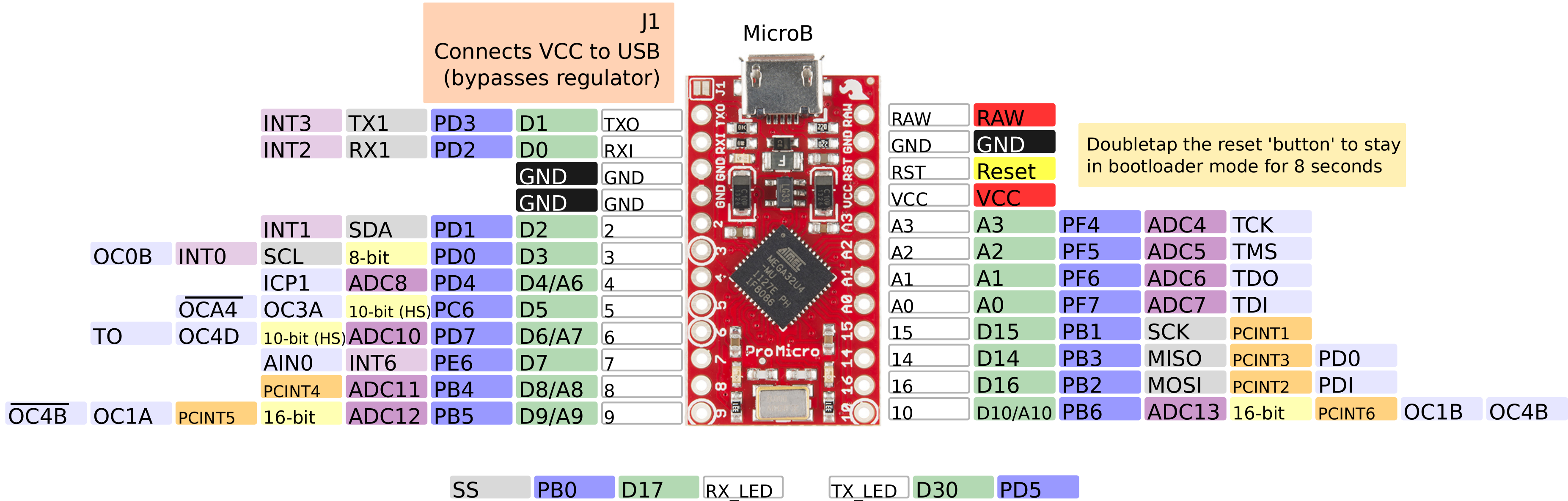
Last update: 30/06/2020

Pro Micro (Dev-12587)

8MHz/3.3V

Name | ADC
Power | PWM
GND | Serial
Control | Ext Interrupt
Arduino | PC Interrupt
Port | Misc

The Arduino IDE renders all PWM pins as 8-bit



Power

RAW: 4V-16V

VCC: 3.3V at 500mA

USB

HID enabled

VID: 0x1B4F

PID: 0x9203 (bootloader); 0x9204 (sketch)

ATmega32U4
- Built in USB 2.0
- Absolute maxiumum VCC: 6V
- Maximum current for chip: 200mA
- Maximum current per pin: 40mA
- Recommended current per pin:20mA
- 8-bit Atmel AVR
- Flash Program Memory: 32kB
- EEPROM: 1kB
- Internal SRAM 2.5kB
- ADC:10-bit
- PWM:8bit
- High Speed PWM with programmable resolution from 2-11 bits

LEDs

Power: Red

RX: Yellow

TX: Green

Serial

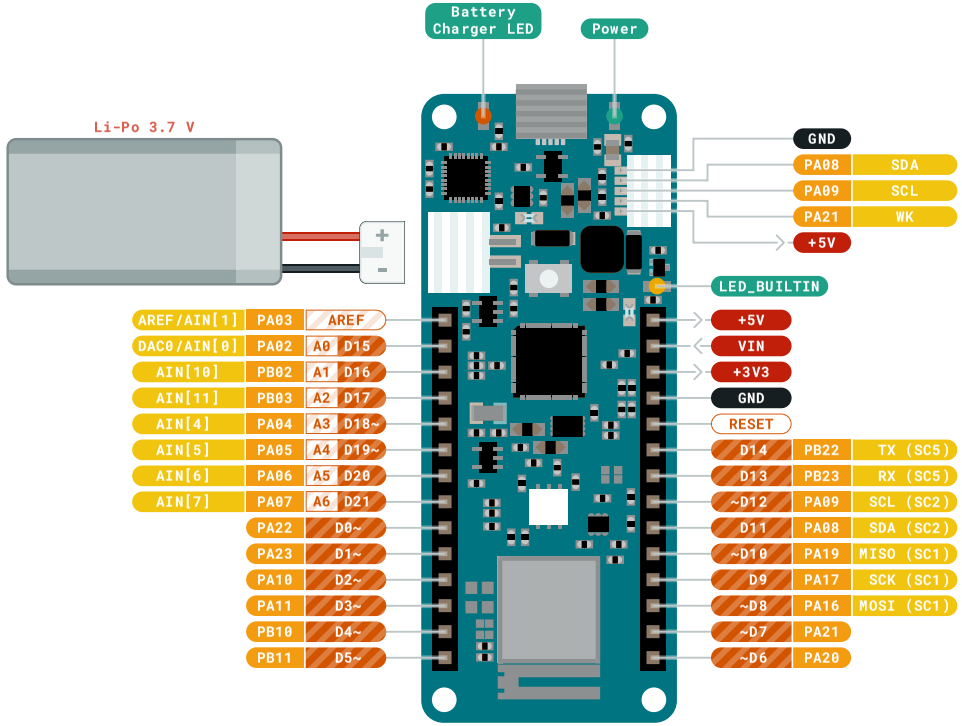
Use Serial for the USB connection

Use Serial1 for the hardware serial connection

ARDUINO

ARDUINO
MKR WIFI 1010

STORE.ARDUINO.CC/MKR-WIFI-1010



- Ground
- Power
- LED
- Internal Pin
- SWD Pin
- Digital Pin
- Analog Pin
- Other Pin
- Microcontroller's Port
- Default

MAXIMUM current per pin is 7mA

MAXIMUM source current is 46mA

MAXIMUM sink current is 65mA per pin group

VIN Input voltage to the board.

ARDUINO.CC

Last update: 7/08/2020

Capítulo 2

Programas de Arduino

2.1 Licencia

Creative Commons Legal Code

CC0 1.0 Universal

CREATIVE COMMONS CORPORATION IS NOT A LAW FIRM AND DOES NOT PROVIDE LEGAL SERVICES. DISTRIBUTION OF THIS DOCUMENT DOES NOT CREATE AN ATTORNEY-CLIENT RELATIONSHIP. CREATIVE COMMONS PROVIDES THIS INFORMATION ON AN .ªS-IS"BASIS. CREATIVE COMMONS MAKES NO WARRANTIES REGARDING THE USE OF THIS DOCUMENT OR THE INFORMATION OR WORKS PROVIDED HEREUNDER, AND DISCLAIMS LIABILITY FOR DAMAGES RESULTING FROM THE USE OF THIS DOCUMENT OR THE INFORMATION OR WORKS PROVIDED HEREUNDER.

Statement of Purpose

The laws of most jurisdictions throughout the world automatically confer exclusive Copyright and Related Rights (defined below) upon the creator and subsequent owner(s) (each and all, an .ºwner") of an original work of authorship and/or a database (each, a "Work").

Certain owners wish to permanently relinquish those rights to a Work for the purpose of contributing to a commons of creative, cultural and scientific works (Çommons") that the public can reliably and without fear of later claims of infringement build upon, modify, incorporate in other works, reuse and redistribute as freely as possible in any form whatsoever and for any purposes, including without limitation commercial purposes. These owners may contribute to the Commons to promote the ideal of a free culture and the further production of creative, cultural and scientific works, or to gain reputation or greater distribution for their Work in part through the use and efforts of others.

For these and/or other purposes and motivations, and without any expectation of additional consideration or compensation, the person associating CC0 with a Work (the .Affirmer"), to the extent that he or she is an owner of Copyright and Related Rights in the Work, voluntarily elects to apply CC0 to the Work and publicly distribute the Work under its terms, with knowledge of his or her Copyright and Related Rights in the Work and the meaning and intended legal effect of CC0 on those rights.

1. Copyright and Related Rights. A Work made available under CC0 may be protected by copyright and related or neighboring rights (Çopyright and Related Rights"). Copyright and Related Rights include, but are not limited to, the following:

i. the right to reproduce, adapt, distribute, perform, display, communicate, and translate a Work; ii. moral rights retained by the original author(s) and/or performer(s); iii. publicity and privacy rights pertaining to a person's image or likeness depicted in a Work; iv. rights protecting against unfair competition in regards to a Work, subject to the limitations in paragraph 4(a), below; v. rights protecting the extraction, dissemination, use and reuse of data in a Work; vi. database rights (such as those arising under Directive 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the legal protection of databases, and under any national implementation thereof, including any amended or successor version of such directive); and vii. other similar, equivalent or corresponding rights throughout the world based on applicable law or treaty, and any national implementations thereof.

2. Waiver. To the greatest extent permitted by, but not in contravention of, applicable law, Affirmer hereby overtly, fully, permanently, irrevocably and unconditionally waives, abandons, and surrenders all of Affirmer's Copyright and Related Rights and associated claims and causes of action, whether now known or unknown (including existing as well as future claims and causes of action), in the Work (i) in all territories worldwide, (ii) for the maximum duration provided by applicable law or treaty (including future time extensions), (iii) in any current or future medium and for any number of copies, and (iv) for any purpose whatsoever, including without limitation commercial, advertising or promotional purposes (the "Waiver"). Affirmer makes the Waiver for the benefit of each member of the public at large and to the detriment of Affirmer's heirs and successors, fully intending that such Waiver shall not be subject to revocation, rescission, cancellation, termination, or any other legal or equitable action to disrupt the quiet enjoyment of the Work by the public as contemplated by Affirmer's express Statement of Purpose.

3. Public License Fallback. Should any part of the Waiver for any reason be judged legally invalid or ineffective under applicable law, then the Waiver shall be preserved to the maximum extent permitted taking into account Affirmer's express Statement of Purpose. In addition, to the extent the Waiver is so judged Affirmer hereby grants to each affected person a royalty-free, non transferable, non sublicensable, non exclusive, irrevocable and unconditional license to exercise Affirmer's Copyright and Related Rights in the Work (i) in all territories worldwide, (ii) for the maximum duration provided by applicable law or treaty (including future time extensions), (iii) in any current or future medium and for any number of copies, and (iv) for any purpose whatsoever, including without limitation commercial, advertising or promotional purposes (the "License"). The License shall be deemed effective as of the date CC0 was applied by Affirmer to the Work. Should any part of the License for any reason be judged legally invalid or ineffective under applicable law, such partial invalidity or ineffectiveness shall not invalidate the remainder of the License, and in

such case Affirmer hereby affirms that he or she will not (i) exercise any of his or her remaining Copyright and Related Rights in the Work or (ii) assert any associated claims and causes of action with respect to the Work, in either case contrary to Affirmer's express Statement of Purpose.

4. Limitations and Disclaimers.

a. No trademark or patent rights held by Affirmer are waived, abandoned, surrendered, licensed or otherwise affected by this document. b. Affirmer offers the Work as-is and makes no representations or warranties of any kind concerning the Work, express, implied, statutory or otherwise, including without limitation warranties of title, merchantability, fitness for a particular purpose, non infringement, or the absence of latent or other defects, accuracy, or the present or absence of errors, whether or not discoverable, all to the greatest extent permissible under applicable law. c. Affirmer disclaims responsibility for clearing rights of other persons that may apply to the Work or any use thereof, including without limitation any person's Copyright and Related Rights in the Work. Further, Affirmer disclaims responsibility for obtaining any necessary consents, permissions or other rights required for any use of the Work. d. Affirmer understands and acknowledges that Creative Commons is not a party to this document and has no duty or obligation with respect to this CC0 or use of the Work.

2.2 Centrifugadora

```
/* LGE Centrifuge
 *
 * Written in 2020 by Francisco Sayas
 * Some lines are from HelloKeypad example of Keypad library, written by
 * Mark Stanley and Alexander Brevig
 *
 * To the extent possible under law, the author(s) have dedicated all copyright
 * and related and neighboring rights to this software to the public domain
 * worldwide. This software is distributed without any warranty.
 *
 * You should have received a copy of the CC0 Public Domain Dedication along
 * with this software. If not, see
 * <http://creativecommons.org/publicdomain/zero/1.0/>.
 *
 */

#include<Keypad.h>

#define PIN_RELAY 13
#define CODE_TIMEOUT 3000

char code[] = {'0', '1', '2', '3', 'A'};
size_t codeLen;
char* inputCode;
int pos = 0;
unsigned long last_key_time;

const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
```

```
};
byte rowPins[ROWS] = {9, 8, 7, 6};
byte colPins[COLS] = {5, 4, 3, 2};
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {

  codeLen = sizeof(code)
  inputCode = malloc(sizeof(char) * codeLen);
  inputCode[codeLen] = '\0';

  pinMode(PIN_RELAY, OUTPUT);

  digitalWrite(PIN_RELAY, LOW);
  last_key_time = millis();

}

void loop() {

  char key = keypad.getKey();

  if(key) {
    inputCode[pos] = key;
    pos++;
    last_key_time = millis();
  }

  if(pos = codeLen){

    if(inputCode == code) {
      digitalWrite(PIN_RELAY, HIGH);
      delay(50);
      digitalWrite(PIN_RELAY, LOW);
    }

    pos = 0;

  }

  if(millis() > last_key_time + CODE_TIMEOUT & pos != 0){
    pos = 0;
  }

}
```

Código 2.1: Código fuente del programa de Centrifugadora

2.3 Bombilla

```
/* LGE Lamp
 *
 * Written in 2020 by Francisco Sayas
 *
 * To the extent possible under law, the author(s) have dedicated all copyright
 * and related and neighboring rights to this software to the public domain
 * worldwide. This software is distributed without any warranty.
 *
 * You should have received a copy of the CC0 Public Domain Dedication along
```

```
 * with this software. If not, see
 * <http://creativecommons.org/publicdomain/zero/1.0/>.
 *
 */

#define TIME_HIGH 400
#define TIME_LOW 400
#define TIME_WAIT 2000

#define PIN_SWITCH 3
#define PIN_LAMP 10
#define PIN_LED 9

unsigned int code[] = {5, 2, 4, 1, 3};
size_t codeLen;

void setup() {

  codeLen = sizeof(code);

  pinMode(PIN_SWITCH, INPUT_PULLUP);
  pinMode(PIN_LAMP, OUTPUT);
  pinMode(PIN_LED, OUTPUT);

  digitalWrite(PIN_LAMP, HIGH);
  digitalWrite(PIN_LED, LOW);

}

void loop() {


  if(digitalRead(PIN_SWITCH)){

    digitalWrite(PIN_LED, HIGH);

    for(int i = 0; i < codeLen; i++){
      for(int j = 0; j < code[i]; j++){
        digitalWrite(PIN_LAMP, LOW);
        delay(TIME_HIGH);
        digitalWrite(PIN_LAMP, HIGH);
        delay(TIME_LOW);
      }
      delay(TIME_WAIT);
    }

    digitalWrite(PIN_LED, LOW);

  }

}
```

Código 2.2: Código fuente del programa de Bombilla

2.4 Cajón

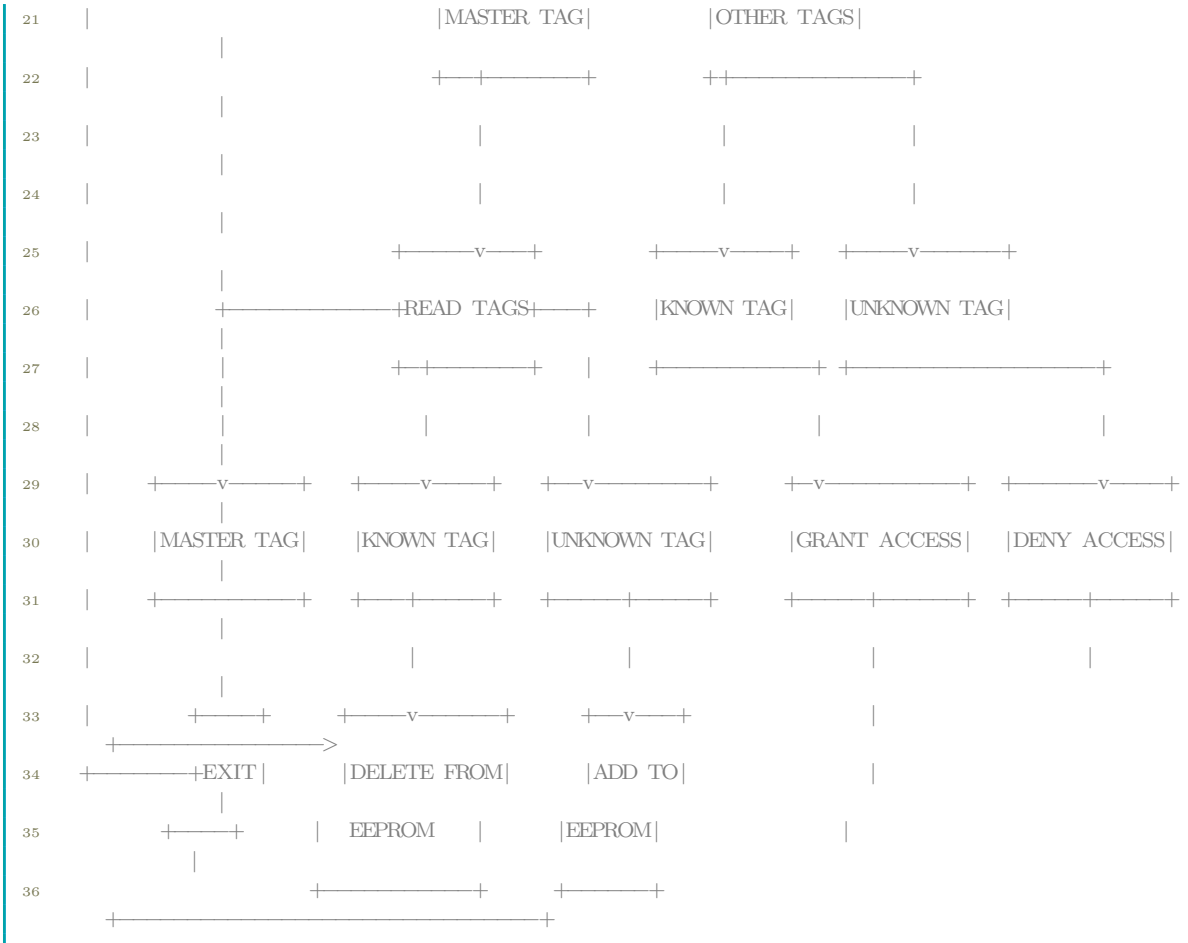
```
/* LGE Drawer
 *
 * Written in 2020 by Francisco Sayas
 *
 * To the extent possible under law, the author(s) have dedicated all copyright
 * and related and neighboring rights to this software to the public domain
 * worldwide. This software is distributed without any warranty.
 *
 * You should have received a copy of the CC0 Public Domain Dedication along
 * with this software. If not, see
 * <http://creativecommons.org/publicdomain/zero/1.0/>.
 *
 */

#define PIN_PIEZO A0
#define PIN_RELAY 9

unsigned long secret[] = {1000, 1500, 2500, 3500, 4000};
size_t lenSecret = sizeof(secret);

const int THRESHOLD_PIEZO = 300;
const int DELAY_FILTER = 50; // ms
const int TOLERANCE = 200; // ms
const int TIMEOUT = 3000; // ms

int vibration = 0;
int knockIdx = 0;
unsigned long knockTime;
unsigned long currentKnock = 0;
unsigned long lastKnock = 0;

void setup() {

  pinMode(PIN_PIEZO, INPUT);
  pinMode(PIN_RELAY, OUTPUT);

}

void loop() {

  vibration = analogRead(PIN_PIEZO);

  if (vibration > THRESHOLD_PIEZO){ // Knock!

    if (knockIdx > 0){

      lastKnock = currentKnock;
      currentKnock = millis();
      knockTime = currentKnock - lastKnock;

      if (knockTime - TOLERANCE < secret[knockIdx] < knockTime + TOLERANCE){

        // Correct knock!
        if (knockIdx == lenSecret - 1){
          // Correct code!
          digitalWrite(PIN_RELAY, LOW);
          delay(500);
          digitalWrite(PIN_RELAY, HIGH);
          knockIdx = 0;
```

```
60          }
61
62        } else {
63          // Fail!
64          knockIdx = 0;
65        }
66
67      } else {
68        currentKnock = millis();
69      }
70
71      knockIdx++;
72      delay(DELAY_FILTER);
73
74    }
75
76    // Timeout
77    if (millis() > currentKnock + TIMEOUT){
78      knockIdx = 0;
79    }
80
81  }
```

Código 2.3: Código fuente del programa de Cajón

2.5 Puerta

```
1  /*
2
       ----------------------------------------------------------------------------
3      Example sketch/program showing An Arduino Door Access Control featuring RFID,
       EEPROM, Relay
4
       ----------------------------------------------------------------------------
5      This is a MFRC522 library example; for further details and other examples see:
       https://github.com/miguelbalboa/rfid
6
7      Modified by Francisco Sayas to suit a particular case of an escape room game:
8      - Now, the door remains open after been unlocked by a know tag.
9      - Added "reset card" that can close the door (with UID hardcoded in setup())
10     - Added buzzer
11
12     This example showing a complete Door Access Control System
13
14   Simple Work Flow (not limited to) :
15                                              +-------------+
16   +--------------------------------------->READ TAGS
       +^-----------------------------------------------+
17     |                                  +--------------------+
                |
18     |                                  |                    |
                |
19     |                                  |                    |
                |
20     |                          +-------v-----+        +-------v-----+
                |
```




```
Use a Master Card which is act as Programmer then you can able to choose card
 holders who will granted access or not

* **Easy User Interface**

Just one RFID tag needed whether Delete or Add Tags. You can choose to use Leds
 for output or Serial LCD module to inform users.

* **Stores Information on EEPROM**

Information stored on non volatile Arduino's EEPROM memory to preserve Users' tag
  and Master Card. No Information lost
if power lost. EEPROM has unlimited Read cycle but roughly 100,000 limited Write
 cycle.

* **Security**
To keep it simple we are going to use Tag's Unique IDs. It's simple and not
 hacker proof.

@license Released into the public domain.

Typical pin layout used:

-----------------------------------------------------------------------------

             MFRC522      Arduino       Arduino    Arduino      Arduino
 Arduino
```

```
58                Reader/PCD   Uno/101        Mega      Nano v3    Leonardo/Micro   Pro
      Micro
59    Signal      Pin          Pin            Pin       Pin        Pin              Pin
60
      ---------------------------------------------------------------------------------------

61    RST/Reset   RST          9              5         D9         RESET/ICSP-5     RST
62    SPI SS      SDA(SS)      10             53        D10        10               10
63    SPI MOSI    MOSI         11 / ICSP-4    51        D11        ICSP-4           16
64    SPI MISO    MISO         12 / ICSP-1    50        D12        ICSP-1           14
65    SPI SCK     SCK          13 / ICSP-3    52        D13        ICSP-3           15
66 */
67
68 #include <EEPROM.h>      // We are going to read and write PICC's UIDs from/to EEPROM
69 #include <SPI.h>         // RC522 Module uses SPI protocol
70 #include <MFRC522.h>  // Library for Mifare RC522 Devices
71
72 /*
73   Instead of a Relay you may want to use a servo. Servos can lock and unlock door
       locks too
74   Relay will be used by default
75 */
76
77 // #include <Servo.h>
78
79 /*
80   For visualizing whats going on hardware we need some leds and to control door lock
       a relay and a wipe button
81   (or some other hardware) Used common anode led,digitalWriting HIGH turns OFF led
       Mind that if you are going
82   to use common cathode led or just seperate leds, simply comment out #define
       COMMON_ANODE,
83 */
84
85 //#define COMMON_ANODE
86
87 #ifdef COMMON_ANODE
88 #define LED_ON LOW
89 #define LED_OFF HIGH
90 #else
91 #define LED_ON HIGH
92 #define LED_OFF LOW
93 #endif
94
95 #define redLed 5    // Set Led Pins
96 #define greenLed 6
97 #define blueLed 17
98
99 #define buzzer 4
100 #define relay 9     // Set Relay Pin
101 #define wipeB 3     // Button pin for WipeMode
102
103 boolean match = false;          // initialize card match to false
104 boolean programMode = false;  // initialize programming mode to false
105 boolean replaceMaster = false;
106
107 boolean doorOpen = false;
108
109 uint8_t successRead;    // Variable integer to keep if we have Successful Read from
       Reader
110
```

```
111 byte storedCard[4];   // Stores an ID read from EEPROM
112 byte readCard[4];   // Stores scanned ID read from RFID Module
113 byte masterCard[4];   // Stores master card's ID read from EEPROM
114 byte resetCard[4];   // Stores reset card ID
115
116 // Create MFRC522 instance.
117 #define SS_PIN 10
118 #define RST_PIN 8
119 MFRC522 mfrc522(SS_PIN, RST_PIN);
120
121 ///////////////////////////////////////// Setup /////////////////////////////////////
122 void setup() {
123
124   resetCard[0] = 39;
125   resetCard[1] = 186;
126   resetCard[2] = 104;
127   resetCard[3] = 4;
128
129   //Arduino Pin Configuration
130   pinMode(buzzer, OUTPUT);
131   pinMode(redLed, OUTPUT);
132   pinMode(greenLed, OUTPUT);
133   pinMode(blueLed, OUTPUT);
134   pinMode(wipeB, INPUT_PULLUP);   // Enable pin's pull up resistor
135   pinMode(relay, OUTPUT);
136   //Be careful how relay circuit behave on while resetting or power-cycling your
        Arduino
137   digitalWrite(relay, HIGH);    // Make sure door is locked
138   TXLED0;
139   digitalWrite(redLed, LED_ON);  // Make sure led is off
140   digitalWrite(greenLed, LED_OFF);  // Make sure led is off
141   digitalWrite(blueLed, LED_OFF); // Make sure led is off
142
143   //Protocol Configuration
144   Serial.begin(9600);  // Initialize serial communications with PC
145   SPI.begin();           // MFRC522 Hardware uses SPI protocol
146   mfrc522.PCD_Init();    // Initialize MFRC522 Hardware
147
148   //If you set Antenna Gain to Max it will increase reading distance
149   //mfrc522.PCD_SetAntennaGain(mfrc522.RxGain_max);
150
151   Serial.println(F("Access Control Example v0.1"));   // For debugging purposes
152   ShowReaderDetails();  // Show details of PCD - MFRC522 Card Reader details
153
154   //Wipe Code - If the Button (wipeB) Pressed while setup run (powered on) it wipes
        EEPROM
155   if (digitalRead(wipeB) == LOW) {  // when button pressed pin should get low,
        button connected to ground
156     digitalWrite(redLed, LED_ON); // Red Led stays on to inform user we are going to
         wipe
157     Serial.println(F("Wipe Button Pressed"));
158     Serial.println(F("You have 10 seconds to Cancel"));
159     Serial.println(F("This will be remove all records and cannot be undone"));
160     bool buttonState = monitorWipeButton(10000); // Give user enough time to cancel
        operation
161     if (buttonState == true && digitalRead(wipeB) == LOW) {    // If button still be
         pressed, wipe EEPROM
162       Serial.println(F("Starting Wiping EEPROM"));
163       for (uint16_t x = 0; x < EEPROM.length(); x = x + 1) {    //Loop end of EEPROM
        address
164         if (EEPROM.read(x) == 0) {              //If EEPROM address 0
```

```
165           // do nothing, already clear, go to the next address in order to save time
      and reduce writes to EEPROM
166         }
167         else {
168           EEPROM.write(x, 0);       // if not write 0 to clear, it takes 3.3mS
169         }
170       }
171       Serial.println(F("EEPROM Successfully Wiped"));
172       digitalWrite(redLed, LED_OFF);  // visualize a successful wipe
173       delay(200);
174       digitalWrite(redLed, LED_ON);
175       delay(200);
176       digitalWrite(redLed, LED_OFF);
177       delay(200);
178       digitalWrite(redLed, LED_ON);
179       delay(200);
180       digitalWrite(redLed, LED_OFF);
181     }
182     else {
183       Serial.println(F("Wiping Cancelled")); // Show some feedback that the wipe
      button did not pressed for 15 seconds
184       digitalWrite(redLed, LED_OFF);
185     }
186   }
187   // Check if master card defined, if not let user choose a master card
188   // This also useful to just redefine the Master Card
189   // You can keep other EEPROM records just write other than 143 to EEPROM address 1
190   // EEPROM address 1 should hold magical number which is '143'
191   if (EEPROM.read(1) != 143) {
192     Serial.println(F("No Master Card Defined"));
193     Serial.println(F("Scan A PICC to Define as Master Card"));
194     do {
195       successRead = getID();            // sets successRead to 1 when we get read
      from reader otherwise 0
196       digitalWrite(blueLed, LED_ON);    // Visualize Master Card need to be defined
197       delay(200);
198       digitalWrite(blueLed, LED_OFF);
199       delay(200);
200     }
201     while (!successRead);                  // Program will not go further while you
      not get a successful read
202     for ( uint8_t j = 0; j < 4; j++ ) {        // Loop 4 times
203       EEPROM.write( 2 + j, readCard[j] );  // Write scanned PICC's UID to EEPROM,
      start from address 3
204     }
205     EEPROM.write(1, 143);                  // Write to EEPROM we defined Master Card
      .
206     Serial.println(F("Master Card Defined"));
207   }
208   Serial.println(F("-------------------"));
209   Serial.println(F("Master Card's UID"));
210   for ( uint8_t i = 0; i < 4; i++ ) {          // Read Master Card's UID from EEPROM
211     masterCard[i] = EEPROM.read(2 + i);    // Write it to masterCard
212     Serial.print(masterCard[i], HEX);
213   }
214   Serial.println("");
215   Serial.println(F("-------------------"));
216   Serial.println(F("Everything is ready"));
217   Serial.println(F("Waiting PICCs to be scanned"));
218   cycleLeds();    // Everything ready lets give user some feedback by cycling leds
219 }
```

```
220
221
222 ////////////////////////////////////// Main Loop
    /////////////////////////////////////
223 void loop () {
224   do {
225     successRead = getID();  // sets successRead to 1 when we get read from reader
    otherwise 0
226     // When device is in use if wipe button pressed for 10 seconds initialize Master
     Card wiping
227     if (digitalRead(wipeB) == LOW) { // Check if button is pressed
228       // Visualize normal operation is iterrupted by pressing wipe button Red is
    like more Warning to user
229       digitalWrite(redLed, LED_ON);  // Make sure led is off
230       digitalWrite(greenLed, LED_OFF);  // Make sure led is off
231       digitalWrite(blueLed, LED_OFF); // Make sure led is off
232       // Give some feedback
233       Serial.println(F("Wipe Button Pressed"));
234       Serial.println(F("Master Card will be Erased! in 10 seconds"));
235       bool buttonState = monitorWipeButton(10000); // Give user enough time to
    cancel operation
236       if (buttonState == true && digitalRead(wipeB) == LOW) {    // If button still
    be pressed, wipe EEPROM
237         EEPROM.write(1, 0);                  // Reset Magic Number.
238         Serial.println(F("Master Card Erased from device"));
239         Serial.println(F("Please reset to re-program Master Card"));
240         while (1);
241       }
242       Serial.println(F("Master Card Erase Cancelled"));
243     }
244     if (programMode) {
245       cycleLeds();              // Program Mode cycles through Red Green Blue
    waiting to read a new card
246     }
247     else {
248       //normalModeOn();     // Normal mode, blue Power LED is on, all others are off
249       delay(1);
250     }
251   }
252   while (!successRead);   //the program will not go further while you are not
      getting a successful read
253   if (programMode) {
254     if ( isMaster(readCard) ) { //When in program mode check First If master card
    scanned again to exit program mode
255       Serial.println(F("Master Card Scanned"));
256       Serial.println(F("Exiting Program Mode"));
257       Serial.println(F("-----------------------------"));
258       programMode = false;
259       return;
260     }
261     else {
262       if ( findID(readCard) ) { // If scanned card is known delete it
263         Serial.println(F("I know this PICC, removing..."));
264         deleteID(readCard);
265         Serial.println("-----------------------------");
266         Serial.println(F("Scan a PICC to ADD or REMOVE to EEPROM"));
267       }
268       else {                    // If scanned card is not known add it
269         Serial.println(F("I do not know this PICC, adding..."));
270         writeID(readCard);
271         Serial.println(F("-----------------------------"));
```

```
272         Serial.println(F("Scan a PICC to ADD or REMOVE to EEPROM"));
273       }
274     }
275   }
276   else {
277     if ( isMaster(readCard)) {    // If scanned card's ID matches Master Card's ID -
       enter program mode
278       programMode = true;
279       Serial.println(F("Hello Master - Entered Program Mode"));
280       uint8_t count = EEPROM.read(0);    // Read the first Byte of EEPROM that
281       Serial.print(F("I have "));     // stores the number of ID's in EEPROM
282       Serial.print(count);
283       Serial.print(F(" record(s) on EEPROM"));
284       Serial.println("");
285       Serial.println(F("Scan a PICC to ADD or REMOVE to EEPROM"));
286       Serial.println(F("Scan Master Card again to Exit Program Mode"));
287       Serial.println(F("-----------------------------------"));
288     } else if ( isReset(readCard) ) {
289       Serial.println(F("Reset Card Scanned"));
290       if (!doorOpen){
291         granted(5000);
292       } else {
293         closeDoor(5000);
294       }
295     }
296     else {
297       if ( findID(readCard) ) { // If not, see if the card is in the EEPROM
298         Serial.println(F("Welcome, You shall pass"));
299         granted(5000);          // Open the door lock for 300 ms
300       }
301       else {      // If not, show that the ID was not valid
302         Serial.println(F("You shall not pass"));
303         denied();
304       }
305     }
306   }
307 }
308
309 /////////////////////////////////////////  Access Granted
       ///////////////////////////////////
310 void granted ( uint16_t setDelay) {
311   tone(buzzer, 440, 150);
312   delay(200);
313   tone(buzzer, 440, 150);
314   //delay(500);
315   digitalWrite(blueLed, LED_OFF);   // Turn off blue LED
316   digitalWrite(redLed, LED_OFF);  // Turn off red LED
317   digitalWrite(greenLed, LED_ON);   // Turn on green LED
318   digitalWrite(relay, LOW);      // Unlock door!
319   TXLED1;
320   doorOpen = true;
321   //delay(setDelay);           // Hold door lock open for given seconds
322   //digitalWrite(relay, HIGH);    // Relock door
323   //TXLED0;
324   //delay(1000);             // Hold green LED on for a second
325 }
326
327 ////////////////////////////////////////// Access Denied
       ///////////////////////////////////
328 void denied() {
329   digitalWrite(greenLed, LED_OFF);  // Make sure green LED is off
```

```
330   digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
331   digitalWrite(redLed, LED_ON);   // Turn on red LED
332   delay(1000);
333 }
334
335 ////////////////////////////////////////  Close
      ////////////////////////////////////
336 void closeDoor ( uint16_t setDelay) {
337   tone(buzzer, 440, 450);
338   delay(500);
339   digitalWrite(blueLed, LED_OFF);   // Turn off blue LED
340   digitalWrite(redLed, LED_ON);  // Turn on red LED
341   digitalWrite(greenLed, LED_OFF);   // Turn off green LED
342   digitalWrite(relay, HIGH);      // Relock door!
343   TXLED0;
344   doorOpen = false;
345   //delay(setDelay);           // Hold door lock open for given seconds
346   //digitalWrite(relay, HIGH);     // Relock door
347   //TXLED0;
348   //delay(1000);             // Hold green LED on for a second
349 }
350
351
352 ////////////////////////////////////// Get PICC's UID
      ////////////////////////////////////
353 uint8_t getID() {
354   // Getting ready for Reading PICCs
355   if ( ! mfrc522.PICC_IsNewCardPresent()) { //If a new PICC placed to RFID reader
      continue
356     return 0;
357   }
358   if ( ! mfrc522.PICC_ReadCardSerial()) {   //Since a PICC placed get Serial and
      continue
359     return 0;
360   }
361   // There are Mifare PICCs which have 4 byte or 7 byte UID care if you use 7 byte
      PICC
362   // I think we should assume every PICC as they have 4 byte UID
363   // Until we support 7 byte PICCs
364   Serial.println(F("Scanned PICC's UID:"));
365   for ( uint8_t i = 0; i < 4; i++) {  //
366     readCard[i] = mfrc522.uid.uidByte[i];
367     Serial.print(readCard[i], HEX);
368   }
369   Serial.println("");
370   mfrc522.PICC_HaltA(); // Stop reading
371   return 1;
372 }
373
374 void ShowReaderDetails() {
375   // Get the MFRC522 software version
376   byte v = mfrc522.PCD_ReadRegister(mfrc522.VersionReg);
377   Serial.print(F("MFRC522 Software Version: 0x"));
378   Serial.print(v, HEX);
379   if (v == 0x91)
380     Serial.print(F(" = v1.0"));
381   else if (v == 0x92)
382     Serial.print(F(" = v2.0"));
383   else
384     Serial.print(F(" (unknown),probably a chinese clone?"));
385   Serial.println("");
```

```
386   // When 0x00 or 0xFF is returned, communication probably failed
387   if ((v == 0x00) || (v == 0xFF)) {
388     Serial.println(F("WARNING: Communication failure, is the MFRC522 properly
      connected?"));
389     Serial.println(F("SYSTEM HALTED: Check connections."));
390     // Visualize system is halted
391     digitalWrite(greenLed, LED_OFF);  // Make sure green LED is off
392     digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
393     digitalWrite(redLed, LED_ON);   // Turn on red LED
394     while (true); // do not go further
395   }
396 }
397
398 ////////////////////////////////////// Cycle Leds (Program Mode)
      ///////////////////////////////////
399 void cycleLeds() {
400   digitalWrite(redLed, LED_OFF);  // Make sure red LED is off
401   digitalWrite(greenLed, LED_ON);   // Make sure green LED is on
402   digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
403   delay(200);
404   digitalWrite(redLed, LED_OFF);  // Make sure red LED is off
405   digitalWrite(greenLed, LED_OFF);  // Make sure green LED is off
406   digitalWrite(blueLed, LED_ON);  // Make sure blue LED is on
407   delay(200);
408   digitalWrite(redLed, LED_ON);   // Make sure red LED is on
409   digitalWrite(greenLed, LED_OFF);  // Make sure green LED is off
410   digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
411   delay(200);
412 }
413
414 ////////////////////////////////////// Normal Mode Led
      ///////////////////////////////////
415 void normalModeOn () {
416   digitalWrite(blueLed, LED_ON);  // Blue LED ON and ready to read card
417   digitalWrite(redLed, LED_ON);  // Make sure Red LED is off
418   digitalWrite(greenLed, LED_OFF);  // Make sure Green LED is off
419   digitalWrite(relay, HIGH);    // Make sure Door is Locked
420   TXLED0;
421 }
422
423 ////////////////////////////////////// Read an ID from EEPROM
      ///////////////////////////////
424 void readID( uint8_t number ) {
425   uint8_t start = (number * 4 ) + 2;    // Figure out starting position
426   for ( uint8_t i = 0; i < 4; i++ ) {     // Loop 4 times to get the 4 Bytes
427     storedCard[i] = EEPROM.read(start + i);   // Assign values read from EEPROM to
      array
428   }
429 }
430
431 ////////////////////////////////////// Add ID to EEPROM
      ///////////////////////////////////
432 void writeID( byte a[] ) {
433   if ( !findID( a ) ) {     // Before we write to the EEPROM, check to see if we
      have seen this card before!
434     uint8_t num = EEPROM.read(0);     // Get the numer of used spaces, position 0
      stores the number of ID cards
435     uint8_t start = ( num * 4 ) + 6;  // Figure out where the next slot starts
436     num++;                // Increment the counter by one
437     EEPROM.write( 0, num );     // Write the new count to the counter
438     for ( uint8_t j = 0; j < 4; j++ ) {   // Loop 4 times
```

```
439       EEPROM.write( start + j, a[j] );  // Write the array values to EEPROM in the
      right position
440     }
441     successWrite();
442     Serial.println(F("Succesfully added ID record to EEPROM"));
443   }
444   else {
445     failedWrite();
446     Serial.println(F("Failed! There is something wrong with ID or bad EEPROM"));
447   }
448 }
449
450 ////////////////////////////////////////// Remove ID from EEPROM
      ////////////////////////////////////
451 void deleteID( byte a[] ) {
452   if ( !findID( a ) ) {     // Before we delete from the EEPROM, check to see if we
      have this card!
453     failedWrite();      // If not
454     Serial.println(F("Failed! There is something wrong with ID or bad EEPROM"));
455   }
456   else {
457     uint8_t num = EEPROM.read(0);   // Get the numer of used spaces, position 0
      stores the number of ID cards
458     uint8_t slot;       // Figure out the slot number of the card
459     uint8_t start;      // = ( num * 4 ) + 6; // Figure out where the next slot
      starts
460     uint8_t looping;    // The number of times the loop repeats
461     uint8_t j;
462     uint8_t count = EEPROM.read(0); // Read the first Byte of EEPROM that stores
      number of cards
463     slot = findIDSLOT( a );   // Figure out the slot number of the card to delete
464     start = (slot * 4) + 2;
465     looping = ((num - slot) * 4);
466     num--;      // Decrement the counter by one
467     EEPROM.write( 0, num );   // Write the new count to the counter
468     for ( j = 0; j < looping; j++ ) {         // Loop the card shift times
469       EEPROM.write( start + j, EEPROM.read(start + 4 + j));   // Shift the array
      values to 4 places earlier in the EEPROM
470     }
471     for ( uint8_t k = 0; k < 4; k++ ) {         // Shifting loop
472       EEPROM.write( start + j + k, 0);
473     }
474     successDelete();
475     Serial.println(F("Succesfully removed ID record from EEPROM"));
476   }
477 }
478
479 ////////////////////////////////////////// Check Bytes
      ////////////////////////////////////
480 boolean checkTwo ( byte a[], byte b[] ) {
481   if ( a[0] != 0 )      // Make sure there is something in the array first
482     match = true;       // Assume they match at first
483   for ( uint8_t k = 0; k < 4; k++ ) {   // Loop 4 times
484     Serial.println(a[k]);
485     Serial.println(b[k]);
486     if ( a[k] != b[k] )     // IF a != b then set match = false, one fails, all fail
487       match = false;
488   }
489   if ( match ) {       // Check to see if if match is still true
490     return true;       // Return true
491   }
```

```
492   else  {
493     return false;       // Return false
494   }
495 }
496
497 ////////////////////////////////////////// Find Slot
      ////////////////////////////////////
498 uint8_t findIDSLOT( byte find[] ) {
499   uint8_t count = EEPROM.read(0);       // Read the first Byte of EEPROM that
500   for ( uint8_t i = 1; i <= count; i++ ) {    // Loop once for each EEPROM entry
501     readID(i);                // Read an ID from EEPROM, it is stored in storedCard
      [4]
502     if ( checkTwo( find, storedCard ) ) {   // Check to see if the storedCard read
      from EEPROM
503       // is the same as the find[] ID card passed
504       return i;         // The slot number of the card
505       break;          // Stop looking we found it
506     }
507   }
508 }
509
510 ////////////////////////////////////////// Find ID From EEPROM
      ////////////////////////////////////
511 boolean findID( byte find[] ) {
512   uint8_t count = EEPROM.read(0);     // Read the first Byte of EEPROM that
513   for ( uint8_t i = 1; i <= count; i++ ) {    // Loop once for each EEPROM entry
514     readID(i);          // Read an ID from EEPROM, it is stored in storedCard[4]
515     if ( checkTwo( find, storedCard ) ) {   // Check to see if the storedCard read
      from EEPROM
516       return true;
517       break;  // Stop looking we found it
518     }
519     else {    // If not, return false
520     }
521   }
522   return false;
523 }
524
525 ////////////////////////////////////////// Write Success to EEPROM
      ////////////////////////////////////
526 // Flashes the green LED 3 times to indicate a successful write to EEPROM
527 void successWrite() {
528   digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
529   digitalWrite(redLed, LED_OFF);  // Make sure red LED is off
530   digitalWrite(greenLed, LED_OFF);  // Make sure green LED is on
531   delay(200);
532   digitalWrite(greenLed, LED_ON);   // Make sure green LED is on
533   delay(200);
534   digitalWrite(greenLed, LED_OFF);  // Make sure green LED is off
535   delay(200);
536   digitalWrite(greenLed, LED_ON);   // Make sure green LED is on
537   delay(200);
538   digitalWrite(greenLed, LED_OFF);  // Make sure green LED is off
539   delay(200);
540   digitalWrite(greenLed, LED_ON);   // Make sure green LED is on
541   delay(200);
542 }
543
544 ////////////////////////////////////////// Write Failed to EEPROM
      ////////////////////////////////////
545 // Flashes the red LED 3 times to indicate a failed write to EEPROM
```

```
546 void failedWrite() {
547   digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
548   digitalWrite(redLed, LED_OFF);  // Make sure red LED is off
549   digitalWrite(greenLed, LED_OFF);  // Make sure green LED is off
550   delay(200);
551   digitalWrite(redLed, LED_ON);   // Make sure red LED is on
552   delay(200);
553   digitalWrite(redLed, LED_OFF);  // Make sure red LED is off
554   delay(200);
555   digitalWrite(redLed, LED_ON);   // Make sure red LED is on
556   delay(200);
557   digitalWrite(redLed, LED_OFF);  // Make sure red LED is off
558   delay(200);
559   digitalWrite(redLed, LED_ON);   // Make sure red LED is on
560   delay(200);
561 }
562
563 ////////////////////////////////////////// Success Remove UID From EEPROM
      ///////////////////////////////////
564 // Flashes the blue LED 3 times to indicate a success delete to EEPROM
565 void successDelete() {
566   digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
567   digitalWrite(redLed, LED_OFF);  // Make sure red LED is off
568   digitalWrite(greenLed, LED_OFF);  // Make sure green LED is off
569   delay(200);
570   digitalWrite(blueLed, LED_ON);  // Make sure blue LED is on
571   delay(200);
572   digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
573   delay(200);
574   digitalWrite(blueLed, LED_ON);  // Make sure blue LED is on
575   delay(200);
576   digitalWrite(blueLed, LED_OFF);   // Make sure blue LED is off
577   delay(200);
578   digitalWrite(blueLed, LED_ON);  // Make sure blue LED is on
579   delay(200);
580 }
581
582 ////////////////////// Check readCard IF is masterCard
      ///////////////////////////////////
583 // Check to see if the ID passed is the master programing card
584 boolean isMaster( byte test[] ) {
585   if ( checkTwo( test, masterCard ) )
586     return true;
587   else
588     return false;
589 }
590
591 bool monitorWipeButton(uint32_t interval) {
592   uint32_t now = (uint32_t)millis();
593   while ((uint32_t)millis() - now < interval)  {
594     // check on every half a second
595     if (((uint32_t)millis() %500) == 0) {
596       if (digitalRead(wipeB) != LOW)
597         return false;
598     }
599   }
600   return true;
601 }
602
603 ////////////////////// Check readCard IF is resetCard
      ///////////////////////////////////
```

```
604 // Check to see if the ID passed is the reset card
605 boolean isReset( byte test[] ) {
606   if ( checkTwo( test, resetCard ) )
607     return true;
608   else
609     return false;
610 }
```

Código 2.4: Código fuente del programa de Puerta

2.6 Compuerta

```
1  /* LGE Gate
2   *
3   * Written in 2020 by Francisco Sayas
4   *
5   * To the extent possible under law, the author(s) have dedicated all copyright
6   * and related and neighboring rights to this software to the public domain
7   * worldwide. This software is distributed without any warranty.
8   *
9   * You should have received a copy of the CC0 Public Domain Dedication along
10  * with this software. If not, see
11  * <http://creativecommons.org/publicdomain/zero/1.0/>.
12  *
13  */
14
15 #define LOW_TRIGGER_VALUE 335
16 #define HIGH_TRIGGER_VALUE 571
17 #define DOOR_DELAY 1000
18 #define FILTER_SAMPLES 50
19
20 #define PIN_SENSOR_ANALOG A0
21 #define PIN_LOCK 10
22 #define PIN_MAGNET 9
23
24
25 int analog_value;
26 int analog_values[FILTER_SAMPLES];
27 int sum_values;
28 bool digital_value;
29 bool door_state;
30
31 void setup() {
32   //Serial.begin(9600);
33   pinMode(PIN_SENSOR_ANALOG, INPUT);
34   pinMode(PIN_LOCK, OUTPUT);
35   pinMode(PIN_MAGNET, OUTPUT);
36   digitalWrite(PIN_LOCK, HIGH);
37   digitalWrite(PIN_MAGNET, HIGH);
38   TXLED0;
39
40   for (int i = 0; i < FILTER_SAMPLES - 1; i++){
41     analog_values[i] = 0;
42   }
43
44   door_state = false; // true: open, false: close
45 }
46
47
```

```
48 void loop() {
49
50   sum_values = 0;
51   for (int i = 0; i < FILTER_SAMPLES - 1; i++){
52     analog_values[i] = analog_values[i+1];
53     sum_values += analog_values[i];
54   }
55   analog_values[FILTER_SAMPLES - 1] = analogRead(PIN_SENSOR_ANALOG);
56   sum_values += analog_values[FILTER_SAMPLES - 1];
57
58   //Serial.println(sum_values / FILTER_SAMPLES);
59
60   if (door_state == false){
61    if (sum_values / FILTER_SAMPLES > HIGH_TRIGGER_VALUE){
62       digitalWrite(PIN_LOCK, LOW);
63       delay(200);
64       digitalWrite(PIN_MAGNET, LOW);
65      TXLED1;
66       door_state = true;
67       delay(DOOR_DELAY);
68     }
69   } else if (sum_values / FILTER_SAMPLES <= LOW_TRIGGER_VALUE){
70     digitalWrite(PIN_LOCK, HIGH);
71     digitalWrite(PIN_MAGNET, HIGH);
72     TXLED0;
73     door_state = false;
74     delay(DOOR_DELAY);
75     delay(1000);
76   }
77
78 }
```

Código 2.5: Código fuente del programa de Compuerta

2.7 Tuberías

```
1  /* LGE Pipes
2   *
3   * Written in 2020 by Francisco Sayas
4   * Maybe it is better to do the job with "for" loops
5   *
6   * To the extent possible under law, the author(s) have dedicated all copyright
7   * and related and neighboring rights to this software to the public domain
8   * worldwide. This software is distributed without any warranty.
9   *
10  * You should have received a copy of the CC0 Public Domain Dedication along
11  * with this software. If not, see
12  * <http://creativecommons.org/publicdomain/zero/1.0/>.
13  *
14  */
15
16 #define PIN_GAUGE 9
17 #define PIN_VALVE_1 A0
18 #define PIN_VALVE_2 A1
19 #define PIN_VALVE_3 A2
20 #define PIN_VALVE_4 A3
21 #define PIN_VALVE_5 A4
22
23 const int COEF_VALVE_1 = 5;
```

```
24 const int COEF_VALVE_2 = 2;
25 const int COEF_VALVE_3 = 3;
26 const int COEF_VALVE_4 = 1;
27 const int COEF_VALVE_5 = 4;
28
29 const int MAX_VALVE_1 = 800;
30 const int MAX_VALVE_2 = 800;
31 const int MAX_VALVE_3 = 800;
32 const int MAX_VALVE_4 = 800;
33 const int MAX_VALVE_5 = 800;
34
35 const int MIN_VALVE_1 = 200;
36 const int MIN_VALVE_2 = 200;
37 const int MIN_VALVE_3 = 200;
38 const int MIN_VALVE_4 = 200;
39 const int MIN_VALVE_5 = 200;
40
41 const int MAX_PWM_GAUGE = 212; // 10V
42
43 int THRESHOLD_VALVE_1;
44 int THRESHOLD_VALVE_2;
45 int THRESHOLD_VALVE_3;
46 int THRESHOLD_VALVE_4;
47 int THRESHOLD_VALVE_5;
48
49 int MAX_VALUE;
50
51 int valve1;
52 int valve2;
53 int valve3;
54 int valve4;
55 int valve5;
56
57 boolean statValve1;
58 boolean statValve2;
59 boolean statValve3;
60 boolean statValve4;
61 boolean statValve5;
62
63 void setup() {
64
65   THRESHOLD_VALVE_1 = (MAX_VALVE_1 + MIN_VALVE_1) / 2;
66   THRESHOLD_VALVE_2 = (MAX_VALVE_2 + MIN_VALVE_2) / 2;
67   THRESHOLD_VALVE_3 = (MAX_VALVE_3 + MIN_VALVE_3) / 2;
68   THRESHOLD_VALVE_4 = (MAX_VALVE_4 + MIN_VALVE_4) / 2;
69   THRESHOLD_VALVE_5 = (MAX_VALVE_5 + MIN_VALVE_5) / 2;
70
71   MAX_VALUE = COEF_VALVE_1 + COEF_VALVE_2 + COEF_VALVE_3 + COEF_VALVE_4 +
       COEF_VALVE_5;
72
73   pinMode(PIN_VALVE_1, INPUT);
74   pinMode(PIN_VALVE_2, INPUT);
75   pinMode(PIN_VALVE_3, INPUT);
76   pinMode(PIN_VALVE_4, INPUT);
77   pinMode(PIN_VALVE_5, INPUT);
78 }
79
80 void loop() {
81
82   valve1 = analogRead(PIN_VALVE_1);
83   valve2 = analogRead(PIN_VALVE_2);
```

```
84    valve3 = analogRead(PIN_VALVE_3);
85    valve4 = analogRead(PIN_VALVE_4);
86    valve5 = analogRead(PIN_VALVE_5);
87
88    if (valve1 > THRESHOLD_VALVE_1){
89      statValve1 = 1;
90    } else {
91      statValve1 = 0;
92    }
93    if (valve2 > THRESHOLD_VALVE_2){
94      statValve2 = 1;
95    } else {
96      statValve2 = 0;
97    }
98    if (valve3 > THRESHOLD_VALVE_3){
99      statValve3 = 1;
100   } else {
101     statValve3 = 0;
102   }
103   if (valve4 > THRESHOLD_VALVE_4){
104     statValve4 = 1;
105   } else {
106     statValve4 = 0;
107   }
108   if (valve5 > THRESHOLD_VALVE_5){
109     statValve5 = 1;
110   } else {
111     statValve5 = 0;
112   }
113
114   setGauge(
115     statValve1,
116     statValve2,
117     statValve3,
118     statValve4,
119     statValve5);
120
121 }
122
123
124 void setGauge(int s1, int s2, int s3, int s4, int s5){
125
126   int value1 = COEF_VALVE_1 * s1;
127   int value2 = COEF_VALVE_2 * s2;
128   int value3 = COEF_VALVE_3 * s3;
129   int value4 = COEF_VALVE_4 * s4;
130   int value5 = COEF_VALVE_5 * s5;
131
132   int value = value1 + value2 + value3 + value4 + value5;
133   int gaugeValue = map(value, 0, MAX_VALUE, 0, 212);
134   analogWrite(PIN_GAUGE, gaugeValue);
135
136 }
```

Código 2.6: Código fuente del programa de Tuberías

2.8 Batería

```
/* LGE Battery
 *
 * Written in 2020 by Francisco Sayas
 * Some lines are from HelloKeypad example of Keypad library, written by
 * Mark Stanley and Alexander Brevig
 *
 * To the extent possible under law, the author(s) have dedicated all copyright
 * and related and neighboring rights to this software to the public domain
 * worldwide. This software is distributed without any warranty.
 *
 * You should have received a copy of the CC0 Public Domain Dedication along
 * with this software. If not, see
 * <http://creativecommons.org/publicdomain/zero/1.0/>.
 *
 */

#include <Wire.h>
#include <ADXL345.h>
#include <Grove_LED_Bar.h>

const float chargeRate = 0.3;
const float dischargeConstant = 1.0;
const int chargeThreshold = 3;
const int loadThreshold = 1000;
const float activeThreshold = 10.0;
const float delayRelay = 300;

ADXL345 adxl;
Grove_LED_Bar bar(6, 5, 0);

float percentLevel = 100.0;
int barLevel;

int location;
double xyz[3];
double ax, ay, az;
double lastax, lastay, lastaz;
double incax, incay, incaz;

void setup()
{
  Serial.begin(9600);

  pinMode(A0, INPUT);

  adxl.powerOn();
  adxl.getAcceleration(xyz);
  lastax = xyz[0];
  lastay = xyz[1];
  lastaz = xyz[2];

  bar.begin();
}

void loop()
{
  location = analogRead(A0);
  if (location < chargeThreshold) { // Charge zone

```

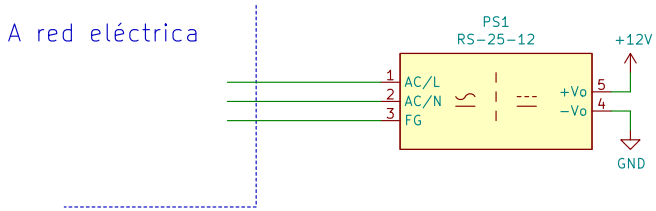
```
60      //Serial.println("CHARGE ZONE!");
61      percentLevel = percentLevel + chargeRate;
62
63    } else if ((location > loadThreshold) & (percentLevel > activeThreshold)) { //
        Load zone
64
65      //Serial.println("LOAD ZONE!");
66      pinMode(A0, OUTPUT);
67      digitalWrite(A0, LOW);
68      percentLevel = 0.0;
69      delay(delayRelay);
70      pinMode(A0, INPUT);
71
72
73    } else { // Between two zones
74
75      adxl.getAcceleration(xyz);
76      ax = xyz[0];
77      ay = xyz[1];
78      az = xyz[2];
79
80      incax = ax - lastax;
81      incay = ay - lastay;
82      incaz = az - lastaz;
83
84      lastax = ax;
85      lastay = ay;
86      lastaz = az;
87
88      percentLevel =  percentLevel - dischargeConstant * (abs(incax) + abs(incay) +
        abs(incaz));
89      if (percentLevel <= 0.0) {
90        percentLevel = 0.0;
91      }
92      //Serial.println(percentLevel);
93      //Serial.println(location);
94
95    }
96
97    barLevel = int(round(mapFloat(percentLevel, 0, 100, 1, 10)));
98    bar.setLevel(barLevel);
99
100 }
101
102 float mapFloat(float x, float in_min, float in_max, float out_min, float out_max) {
103   return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
104 }
```

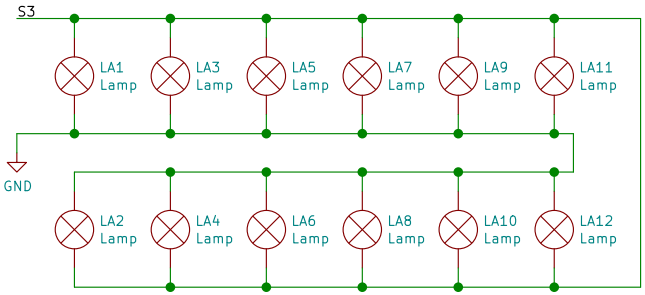
Código 2.7: Código fuente del programa de Batería

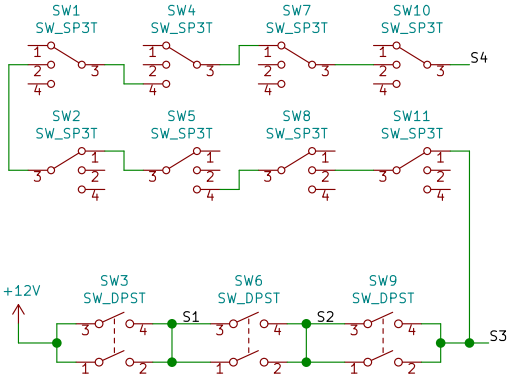
Parte IV

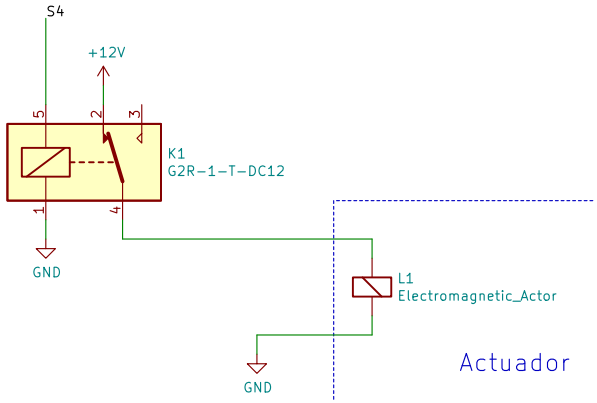
Planos

Índice







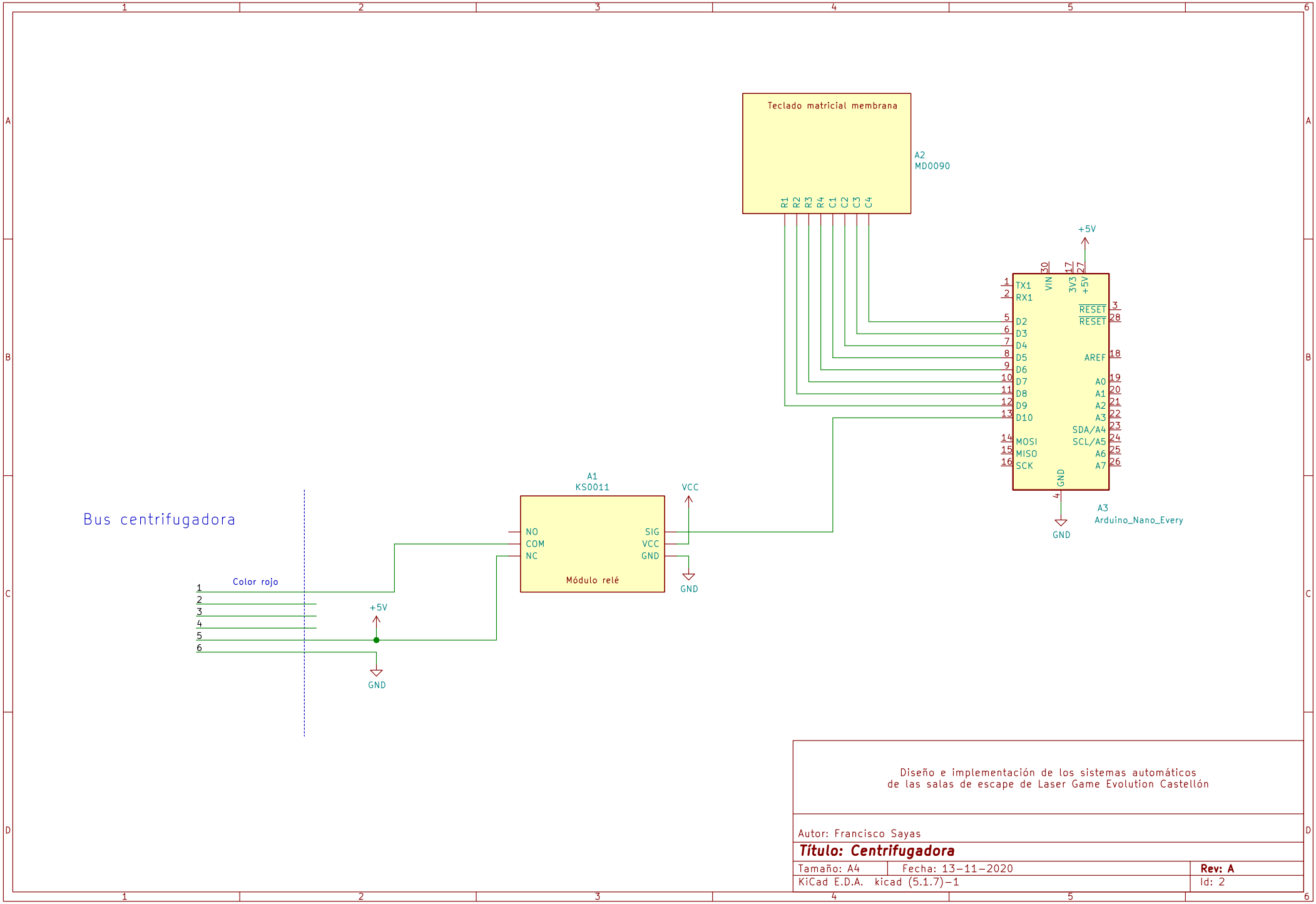


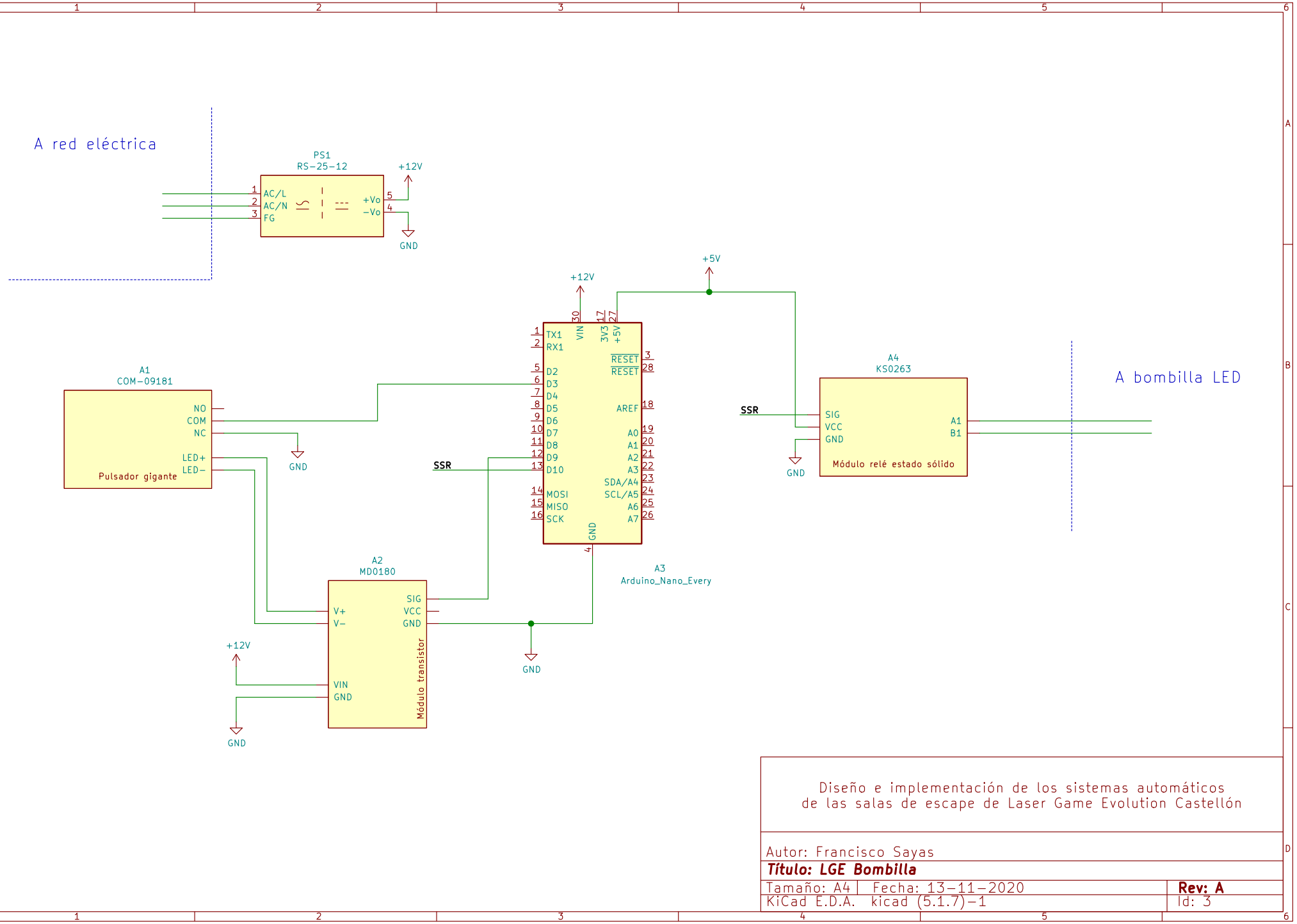
Diseño e implementación de los sistemas automáticos
de las salas de escape de Laser Game Evolution Castellón

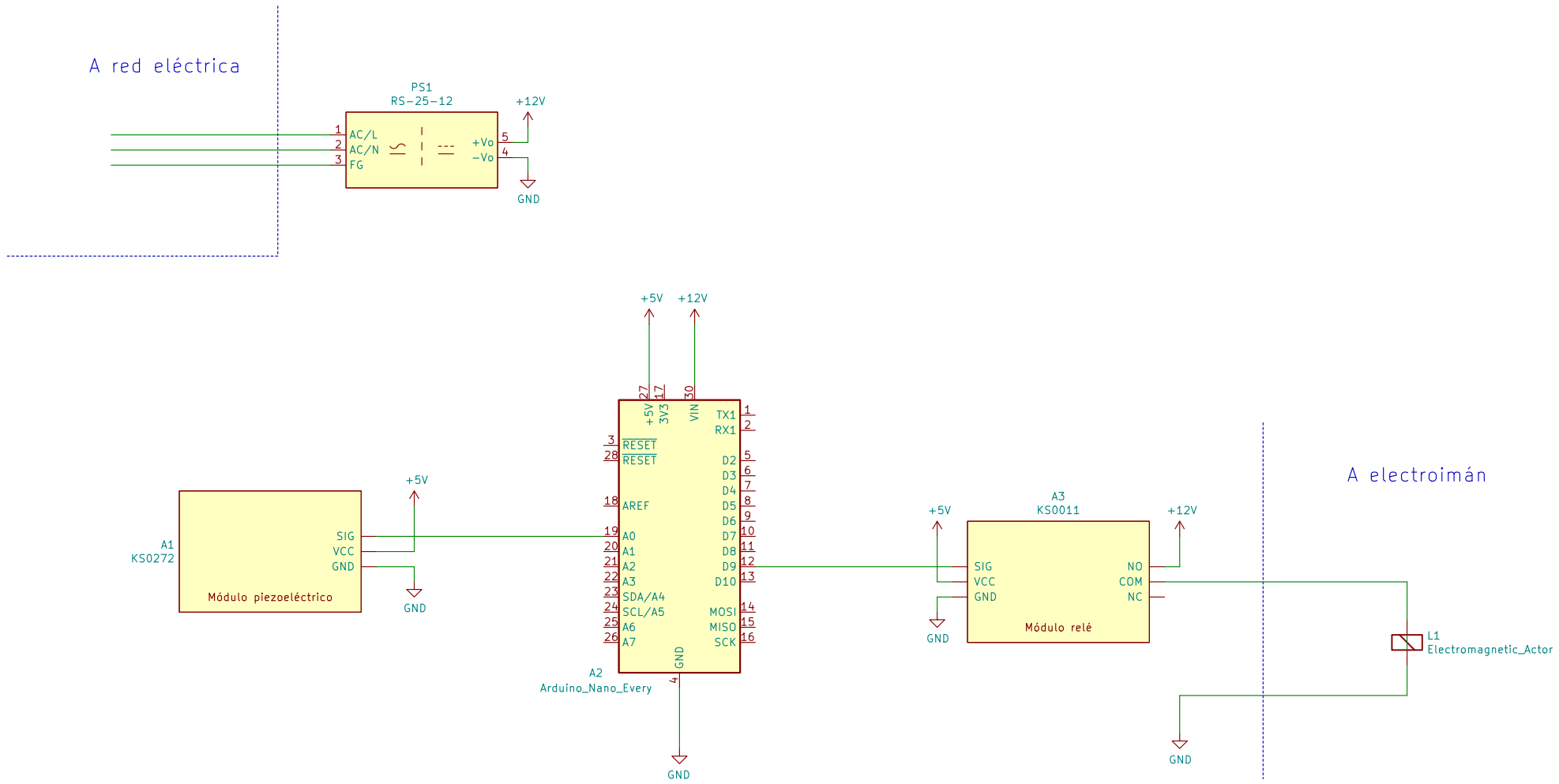
Autor: Francisco Sayas

Título: Armario

Tamaño: A4	Fecha: 13-11-2020	**Rev: A**
KiCad E.D.A.	kicad (5.1.7)-1	Id: 1


Teclado matricial membrana
A2
MD0090
R1
R2
R3
R4
C1
C2
C3
C4
+5V
TX1
RX1
D2
D3
D4
D5
D6
D7
D8
D9
D10
MOSI
MISO
SCK
VIN
3V3
+5V
RESET
RESET
AREF
A0
A1
A2
A3
SDA/A4
SCL/A5
A6
A7
GND
A3
Arduino_Nano_Every
A1
KS0011
NO
COM
NC
SIG
VCC
GND
Módulo relé
VCC
GND
Bus centrifugadora
Color rojo
+5V
GND
Diseño e implementación de los sistemas automáticos
de las salas de escape de Laser Game Evolution Castellón
Autor: Francisco Sayas
Título: Centrifugadora
Tamaño: A4
Fecha: 13-11-2020
Rev: A
KiCad E.D.A. kicad (5.1.7)-1
Id: 2


A red eléctrica
PS1
RS-25-12
AC/L
AC/N
FG
+Vo
-Vo
+12V
GND
A1
COM-09181
NO
COM
NC
LED+
LED-
Pulsador gigante
GND
A2
MD0180
V+
V-
SIG
VCC
GND
VIN
GND
Módulo transistor
+12V
GND
SSR
A3
Arduino_Nano_Every
TX1
RX1
D2
D3
D4
D5
D6
D7
D8
D9
D10
MOSI
MISO
SCK
VIN
3V3
+5V
RESET
RESET
AREF
A0
A1
A2
A3
SDA/A4
SCL/A5
A6
A7
GND
+12V
+5V
GND
SSR
A4
KS0263
SIG
VCC
GND
A1
B1
Módulo relé estado sólido
GND
A bombilla LED
Diseño e implementación de los sistemas automáticos
de las salas de escape de Laser Game Evolution Castellón
Autor: Francisco Sayas
Título: LGE Bombilla
Tamaño: A4
Fecha: 13-11-2020
KiCad E.D.A. kicad (5.1.7)-1
Rev: A
Id: 3

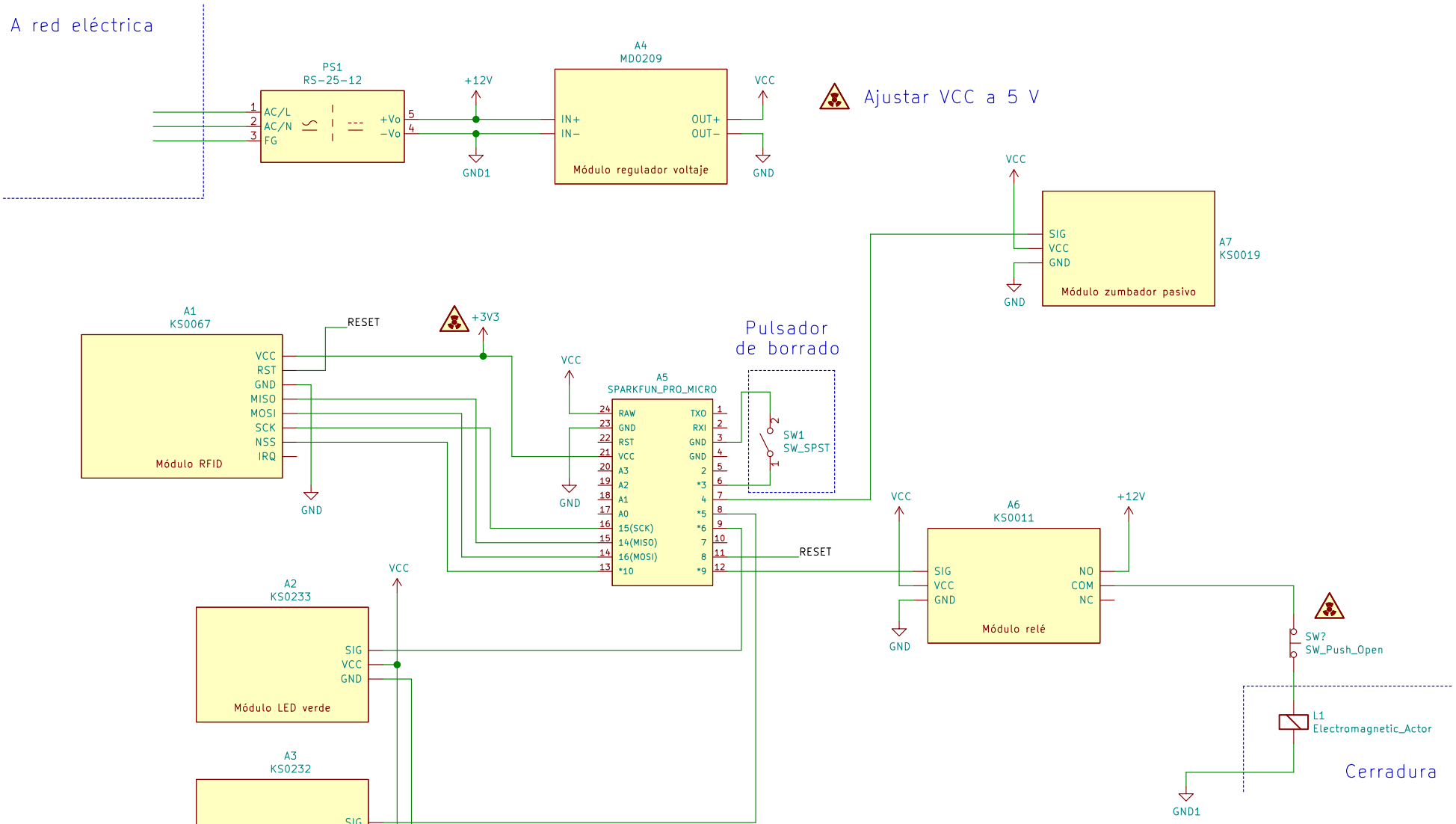


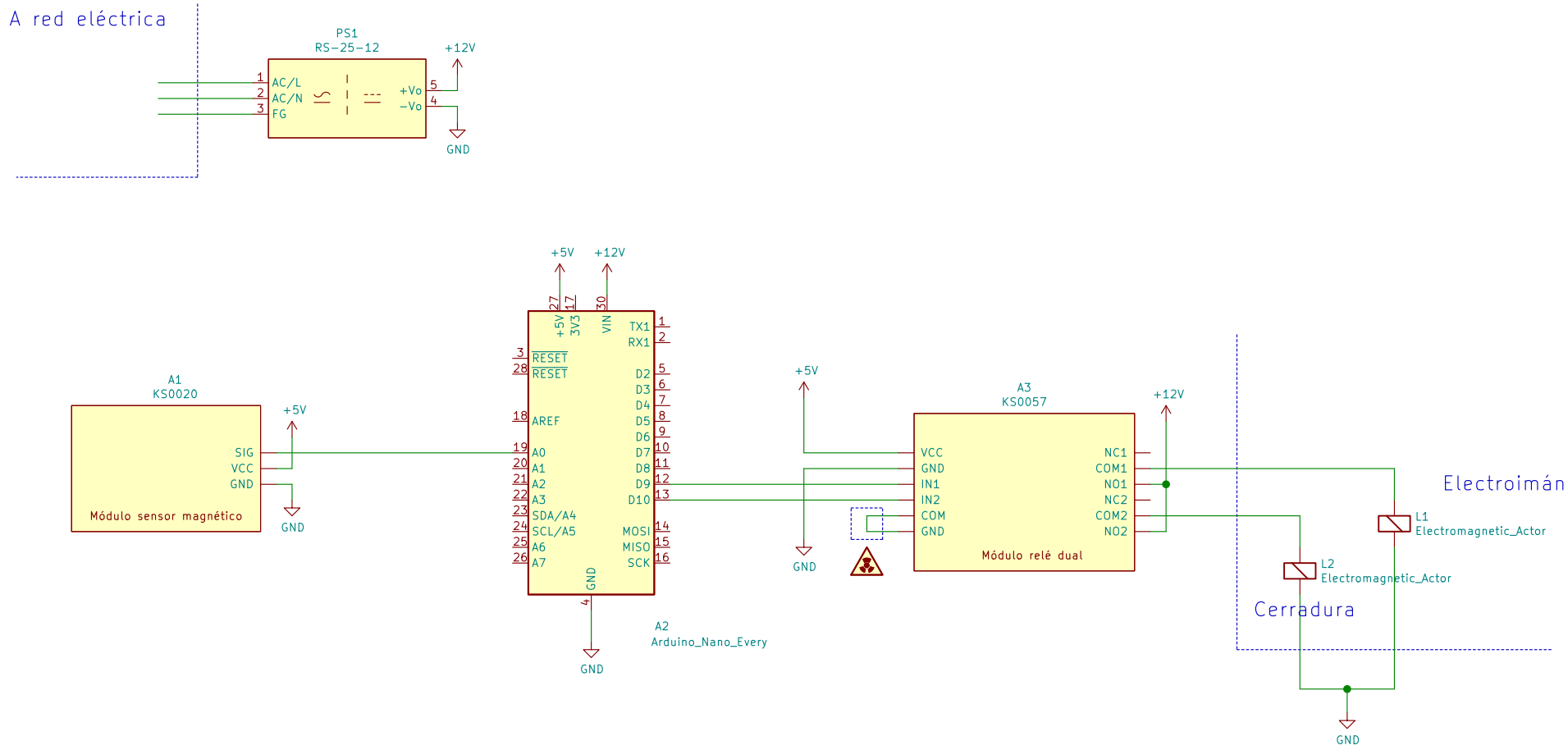
Diseño e implementación de los sistemas automáticos
de las salas de escape de Laser Game Evolution Castellón

Autor: Francisco Sayas

Título: LGE Cajón

Tamaño: A4	Fecha: 13-11-2020	**Rev: A**
KiCad E.D.A. kicad (5.1.7)-1		Id: 4


A red eléctrica
PS1
RS-25-12
AC/L
AC/N
FG
+Vo
-Vo
+12V
GND1
A4
MD0209
IN+
IN-
OUT+
OUT-
Módulo regulador voltaje
VCC
GND
Ajustar VCC a 5 V
A7
KS0019
SIG
VCC
GND
Módulo zumbador pasivo
A1
KS0067
RST
MISO
MOSI
SCK
NSS
IRQ
Módulo RFID
RESET
+3V3
A5
SPARKFUN_PRO_MICRO
RAW
GND
RST
VCC
A3
A2
A1
A0
15(SCK)
14(MISO)
16(MOSI)
*10
TXO
RXI
GND
GND
2
*3
4
*5
*6
7
8
*9
Pulsador
de borrado
SW1
SW_SPST
A6
KS0011
NO
COM
NC
Módulo relé
+12V
SW?
SW_Push_Open
L1
Electromagnetic_Actor
Cerradura
A2
KS0233
Módulo LED verde
A3
KS0232
Módulo LED rojo
Diseño e implementación de los sistemas automáticos
de las salas de escape de Laser Game Evolution Castellón
Autor: Francisco Sayas
Título: LGE Puerta
Tamaño: A4
Fecha: 13-11-2020
KiCad E.D.A. kicad (5.1.7)-1
Rev: A
Id: 5

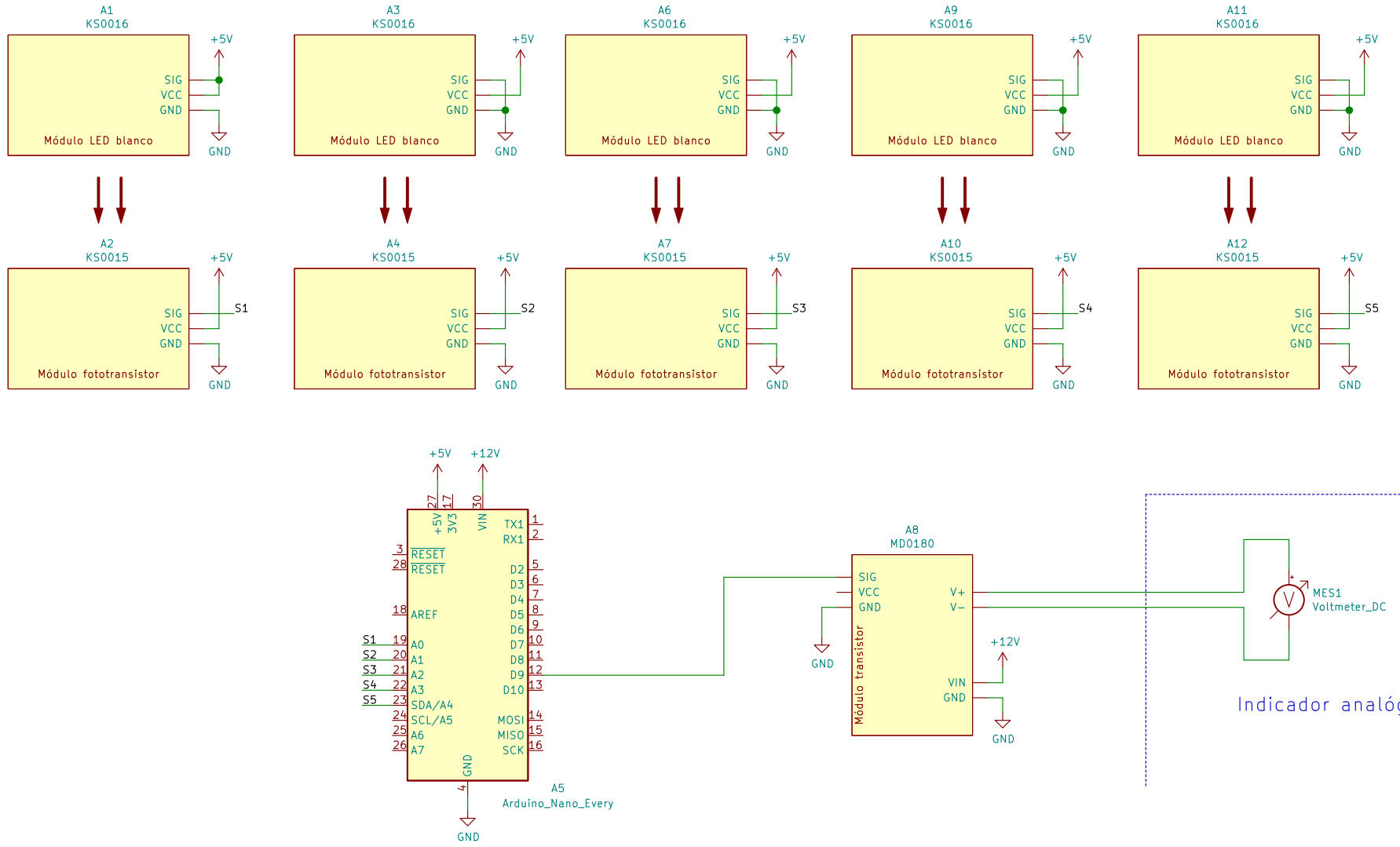


Diseño e implementación de los sistemas automáticos
de las salas de escape de Laser Game Evolution Castellón

Autor: Francisco Sayas

Título: LGE Compuerta

Tamaño: A4	Fecha: 13-11-2020	**Rev: A**
KiCad E.D.A.	kicad (5.1.7)-1	Id: 6

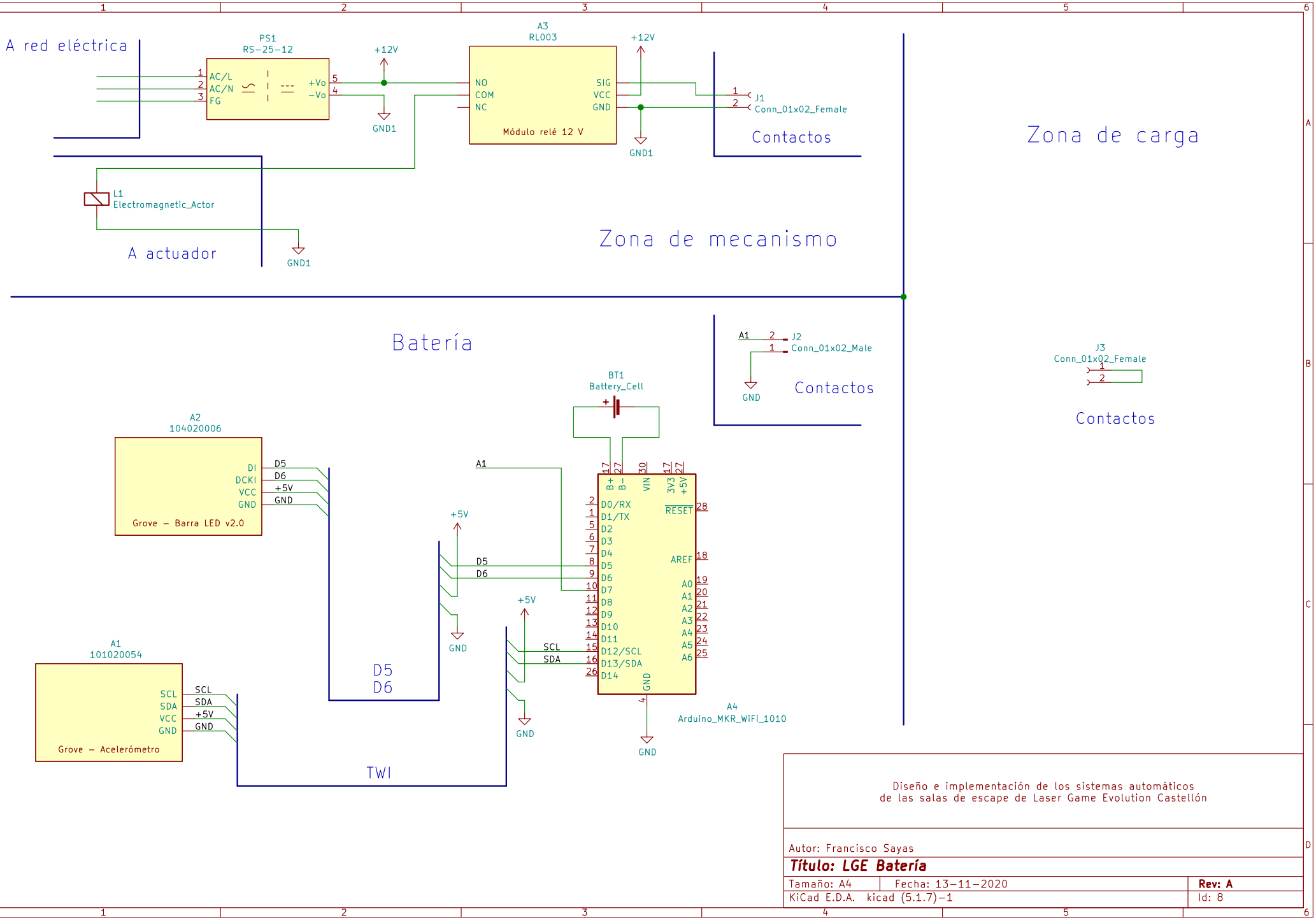


Diseño e implementación de los sistemas automáticos
de las salas de escape de Laser Game Evolution Castellón

Autor: Francisco Sayas

Título: LGE Tuberías

Tamaño: A4	Fecha: 13−11−2020	**Rev: A**
KiCad E.D.A. kicad (5.1.7)−1		Id: 7


A red eléctrica
PS1
RS-25-12
AC/L
AC/N
FG
+Vo
-Vo
+12V
GND1
A3
RL003
NO
COM
NC
SIG
VCC
GND
Módulo relé 12 V
+12V
GND1
J1
Conn_01x02_Female
Contactos
Zona de carga
L1
Electromagnetic_Actor
A actuador
GND1
Zona de mecanismo
Batería
A1
2
J2
Conn_01x02_Male
GND
Contactos
J3
Conn_01x02_Female
Contactos
BT1
Battery_Cell
A2
104020006
DI
DCKI
VCC
GND
Grove – Barra LED v2.0
D5
D6
+5V
GND
A1
B+
B-
VIN
3V3
+5V
D0/RX
D1/TX
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12/SCL
D13/SDA
D14
RESET
AREF
A0
A1
A2
A3
A4
A5
A6
GND
A4
Arduino_MKR_WiFi_1010
A1
101020054
SCL
SDA
VCC
GND
Grove – Acelerómetro
D5
D6
TWI
Diseño e implementación de los sistemas automáticos
de las salas de escape de Laser Game Evolution Castellón
Autor: Francisco Sayas
Título: LGE Batería
Tamaño: A4
Fecha: 13-11-2020
Rev: A
KiCad E.D.A. kicad (5.1.7)-1
Id: 8

Parte V

Mediciones

Capítulo 1

Mediciones

1.1 Lista de materiales

Partida / Concepto	Distribuidor	Ref. distribuidor	Unidades
Material fungible			
Cable de alimentación	Leroy Merlin	17915002	50
Kit de cables de desarrollo	Solectroshop	K0114	4
Armario			
Interruptor de palanca	RS	782-7234	8
Relé	Mouser	653-G2R-1-T-DC12	
Fuente de alimentación	Mouser	709-RS25-12	
Centrifugadora			
Placa de desarrollo	Mouser	782-ABX00033	
Teclado matricial de membrana	Solectroshop	M0025	
Módulo de relé	Solectroshop	RL001	
Bombilla			
Placa de desarrollo	Mouser	782-ABX00033	
Pulsador gigante	Mouser	474-COM-09181	
Módulo de relé de estado sólido	Keyestudio	KS0263	
Módulo de transistor	Solectroshop	A0022	
Fuente de alimentación	Mouser	709-RS25-12	

Cajón			
Placa de desarrollo	Mouser	782-ABX00033	
Módulo de sensor piezoeléctrico	Keyestudio	KS0272	
Módulo de relé	Solectroshop	RL001	
Fuente de alimentación	Mouser	709-RS25-12	
Puerta			
Placa de desarrollo	Mouser	474-DEV-12587	
Kit de desarrollo de tarjetas sin contacto	Keyestudio	KS0067	
Módulo de relé	Solectroshop	RL001	
Módulo regulador de voltaje	Solectroshop	A0004	
Módulo LED rojo	Keyestudio	KS0232	
Módulo LED verde	Keyestudio	KS0233	
Módulo zumbador pasivo	Keyestudio	KS0019	
Fuente de alimentación	Mouser	709-RS25-12	
Compuerta			
Placa de desarrollo	Mouser	782-ABX00033	
Kit de imanes de neodimio	Asterkat	223006606051	
Módulo de sensor magnético	Keyestudio	KS0020	
Módulo de relé de doble canal	Keyestudio	KS0057	
Fuente de alimentación	Mouser	709-RS25-12	
Tuberías			
Placa de desarrollo	Mouser	782-ABX00033	
Módulo LED blanco	Keyestudio	KS0016	5
Módulo de fototransistor	Keyestudio	KS0015	5
Fuente de alimentación	Mouser	709-RS25-12	
Batería			
Placa de desarrollo	Mouser	782-ABX00023	
Escudo de conexiones	Mouser	782-ASX00007	
Kit de cables de desarrollo	Mouser	713-110990028	
Módulo de acelerómetro	Mouser	713-101020054	
Módulo de barra LED	Mouser	713-104020006	
Módulo de relé	Solectroshop	RL001	
Batería de polímero de litio	Bricogeek	BAT-0012	
Fuente de alimentación	Mouser	709-RS25-12	

1.2 Mano de obra

Partida / Concepto		Horas
Ingeniería	**Ingeniero industrial**	
Diseño Armario		4
Diseño Centrifugadora		8
Diseño Bombilla		4
Diseño Cajón		4
Diseño Puerta		4
Diseño Compuerta		4
Diseño Tuberías		4
Diseño Batería		8
Instalación	**Instalador autorizado**	
Montaje e instalación Armario		4
Montaje e instalación Centrifugadora		2
Montaje e instalación Bombilla		2
Montaje e instalación Cajón		3
Montaje e instalación Puerta		5
Montaje e instalación Compuerta		3
Montaje e instalación Tuberías		5
Montaje e instalación Batería		6
Verificaciones		2

Parte VI

Presupuesto

Capítulo 1

Presupuesto

1.1 Consideraciones

- Las tareas de diseño, realizadas por un ingeniero, se presupuestan a $50\,€\,h^{-1}$. Las tareas de montaje, realizadas por un técnico, se presupuestan a $30\,€\,h^{-1}$. Esos precios incluyen implícitos el beneficio industrial y los gastos generales del Proyecto.
- El presupuesto no incluye seguros, costes de certificación y visado, permisos y licencias o cualquier otro concepto similar que no esté especificado en el mismo.

1.2 Presupuesto

Partida / Concepto	Cantidad (uds.)	Precio (€/ud.)	Precio (€)	
Consumibles				49,48
Cable de alimentación	50	0,66	33	
Kit de cables de desarrollo	4	4,12	16,48	
Armario				57,53
Interruptor de palanca	8	5,45	43,6	
Relé		5,29	5,29	
Fuente de alimentación		8,64	8,64	
Centrifugadora				14,2
Placa de desarrollo		11,79	11,79	
Teclado matricial de membrana		1,07	1,07	
Módulo de relé		1,34	1,34	
Bombilla				37,06
Placa de desarrollo		11,79	11,79	
Pulsador gigante		10,12	10,12	
Módulo de relé de estado sólido		4,73	4,73	
Módulo de transistor		1,78	1,78	
Fuente de alimentación		8,64	8,64	
Cajón				25,77
Placa de desarrollo		11,79	11,79	
Módulo de sensor piezoeléctrico		4	4	
Módulo de relé		1,34	1,34	
Fuente de alimentación		8,64	8,64	
Puerta				41
Placa de desarrollo		16,9	16,9	
Kit de desarrollo de tarjetas sin contacto		5,6	5,6	
Módulo de relé		1,34	1,34	
Módulo regulador de voltaje		1,29	1,29	
Módulo LED rojo		2,41	2,41	
Módulo LED verde		2,01	2,01	
Módulo zumbador pasivo		2,81	2,81	
Fuente de alimentación		8,64	8,64	
Compuerta				28,53
Placa de desarrollo		11,79	11,79	
Kit de imanes de neodimio		0,88	0,88	
Módulo de sensor magnético		2,81	2,81	
Módulo de relé de doble canal		4,41	4,41	
Fuente de alimentación		8,64	8,64	

Partida / Concepto	Cantidad (uds.)	Precio (€/ud.)	Precio (€)	
Tuberías				48,53
Placa de desarrollo		11,79	11,79	
Módulo LED blanco	5	2,81	14,05	
Módulo de fototransistor	5	2,81	14,05	
Fuente de alimentación		8,64	8,64	
Batería				89,58
Placa de desarrollo		28,71	28,71	
Escudo de conexiones		19,4	19,4	
Kit de cables de desarrollo		3,3	3,3	
Módulo de acelerómetro		8,39	8,39	
Módulo de barra LED		3,3	3,3	
Módulo de relé		1,34	1,34	
Batería de polímero de litio		16,5	16,5	
Fuente de alimentación		8,64	8,64	
Ingeniería				2000
Diseño Armario	4	50	200	
Diseño Centrifugadora	8	50	400	
Diseño Bombilla	4	50	200	
Diseño Cajón	4	50	200	
Diseño Puerta	4	50	200	
Diseño Compuerta	4	50	200	
Diseño Tuberías	4	50	200	
Diseño Batería	8	50	400	
Instalación				960
Montaje e instalación Armario	4	30	120	
Montaje e instalación Centrifugadora	2	30	60	
Montaje e instalación Bombilla	2	30	60	
Montaje e instalación Cajón	3	30	90	
Montaje e instalación Puerta	5	30	150	
Montaje e instalación Compuerta	3	30	90	
Montaje e instalación Tuberías	5	30	150	
Montaje e instalación Batería	6	30	180	
Verificaciones	2	30	60	

TOTAL SIN IVA	3351,68 €
IVA (21 %)	703,85 €
TOTAL PRESUPUESTO	4055,53 €

CUATRO MIL CINCUENTA Y CINCO EUROS CON CINCUENTA Y TRES CÉNTIMOS