Universidad Carlos III de Madrid
www.uc3m.es

UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Departamento de Ingeniería de Sistemas y Automática

INTEGRACIÓN DE UNA TARJETA RFID DE BAJO COSTE EN EL FRAMEWORK ROBOT OPERATING SYSTEM

Trabajo Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

Autor: Gustavo González de la Cruz
Tutor: Álvaro Castro González

Leganés, Septiembre 2014

AGRADECIMIENTOS

En primer lugar, me gustaría dar mi más sincero agradecimiento al tutor de este Trabajo Fin de Grado, Álvaro Castro González, no sólo por darme la posibilidad de poder realizarlo, sino también por ofrecerme y darme su ayuda siempre que la he requerido.

A los compañeros de laboratorio, en especial a David García Godoy, Alberto Ramón Jañez y a Irene Pérez Encinar por estar siempre disponibles para ayudarme a resolver cualquier problema que me ha surgido durante la realización de este trabajo.

A mis amigos de Aranjuez, en especial a Jorgis, Miguelito, Peña, Lallo, Goncho, Grana, Varo, Javi...y a muchos otros a los que no nombro porque me tiraría líneas y líneas escribiendo.

Tampoco me quiero olvidar de mis compañeros de equipo, ya que sin alguno de ellos, seguramente, no hubiese sido capaz de llegar hasta aquí.

A mis padres y a mi hermana por todo el apoyo que me han dado durante tantos años.

Y cómo no, gracias a Alex, por todos estos meses, por aguantarme en los peores momentos y por dejarme disfrutar junto a ella en los mejores.

A todos ellos, gracias.

ÍNDICE.

ÍNDICE DE FIGURAS.

ÍNDICE DE TABLAS.

1 INTRODUCCIÓN.

1.1 Motivación.

Cada vez son más numerosas las aplicaciones que utilizan las tecnologías de identificación automática. En los últimos años, estas aplicaciones están teniendo una importancia muy grande. Un ejemplo de ello es el caso de los sistemas de identificación por radiofrecuencia RFID, ya que estos sistemas están creciendo en el campo de la robótica, en donde se necesitan nuevas formas de interactuar con el robot.

Los sistemas RFID pueden ser integrados en robots personales para llevar a cabo tareas relacionadas con la identificación de objetos o como soporte a otros sistemas sensoriales.

En este caso, se trabajará con un sistema RFID en la frecuencia de 13.56 MHz, el cual le permite interaccionar con él. Este sistema presenta un buen funcionamiento para aplicaciones de reconocimiento de objetos a corta distancia.

Este proyecto nace de la idea de poder identificar tarjetas RFID que contengan cierta información y esta pueda ser utilizada para diversas actividades.

El Trabajo Fin de Grado se ha llevado a cabo en el Departamento de Ingeniería de Sistemas y Automática. Para la realización del proyecto son necesarios conocimientos de C++, de Arduino y de programación de hardware a bajo nivel. Estos conocimientos son adquiridos en diversas asignaturas cursadas en el Grado en Ingeniería Electrónica Industrial y Automática, como son Programación, Informática Industrial I, Electrónica Digital y Fundamentos de Ingeniería Electrónica.

1.2 Objetivo.

El objetivo de este trabajo es desarrollar el control de una tarjeta RFID. Para lograrlo, es necesario desarrollar las funcionalidades con las que conseguir utilizar dicha tarjeta, tanto para su escritura como para su lectura. Además, la tarjeta debe integrarse con el software de control de robot ROS (Robot Operating System) para que permita distribuir los datos de tarjetas RFID detectadas al resto de la arquitectura de control del robot.

Para conseguir este objetivo, hay que alcanzar una serie de sub-objetivos:

- Búsqueda y análisis de los dispositivos RFID disponibles en el mercado que cumplen con los requerimientos software y hardware del sistema.
- Conexionado del hardware de los distintos dispositivos utilizados.
- Programación de la placa Arduino Mega 2560.
- Comunicar el dispositivo RFID con el resto de la arquitectura.

1.3 Herramientas empleadas.

Para el desarrollo del Trabajo Fin de Grado se ha hecho uso de las siguientes herramientas:

- Software de Arduino.
- Software ROS.
- Sistema Operativo Ubuntu.
- Lenguaje de programación C++.

1.4 Estructura del documento.

Esta memoria describe el proceso y la metodología para alcanzar el objetivo del Trabajo Fin de Grado.

- En el capítulo 2, se explica en qué consiste la tecnología RFID utilizada en este trabajo, los elementos que intervienen y su funcionamiento. También se describen las características de cada banda de frecuencia en la que pueden operar los sistemas RFID y su estandarización. Además, al final del capítulo se introducen brevemente las aplicaciones RFID en el campo de la robótica.

- En el capítulo 3, se describe el proceso realizado para conseguir este proyecto, diferenciando dos partes: la relacionada con el diseño del hardware y la que comprende la programación de Arduino y ROS.

- Los resultados obtenidos de las pruebas realizadas con este sistema RFID se muestran en el capítulo 4.

- Las conclusiones y las líneas futuras de trabajo que surgen con este trabajo se recogen en el capítulo 5.

- En el capítulo 6, se estima el presupuesto de este proyecto.

- El capítulo 7 comprende las referencias bibliográficas del trabajo.

- Los anexos se encuentran en el capítulo 8.

2 TECNOLOGÍAS UTILIZADAS.

2.1 Tecnología RFID.

2.1.1 Introducción.

RFID (Radio Frequency IDentification) es un sistema de almacenamiento y recuperación de datos remoto que usa dispositivos denominados tags RFID (también llamados tarjetas, transpondedores o etiquetas). El objetivo fundamental que busca la tecnología RFID es transmitir la identidad de un objeto (similar a un número de serie único) mediante ondas de radio. Las tecnologías RFID se agrupan dentro de las denominadas Auto ID (automatic identification) [1].

En la Figura 1 se muestra una perspectiva de las tecnologías de identificación automática más relevantes.

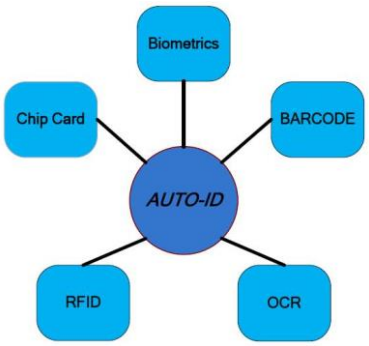


Figura 1 – Tecnologías de identificación automáticas

El principal motivo por el que la tecnología RFID no entrara antes en el mercado fue su coste, relativamente más alto que el del código de barras. Hace años no era demasiado práctico etiquetar objetos en donde el método para identificarlos fuese más caro que el propio objeto.

Por el contrario, actualmente el uso de la tecnología RFID se está extendiendo gracias a los avances en la electrónica y a la reducción de los costes, que hace que nuevas aplicaciones RFID sean viables económicamente. Como se puede ver en la Tabla 1, la tecnología RFID ofrece una serie de ventajas sobre la utilización del código de barras.

Característica	RFID	Código de Barras
Capacidad	Superior cantidad de información.	Limitación de información.
Identificación	Depende del producto.	Estandarizada.
Actualización	Lectura y escritura.	Sólo lectura.
Flexibilidad	No requiere línea de visión para lectura.	Requiere línea de visión para lectura.
Lectura	Múltiple lectura.	Una lectura por vez.
Tipo de lectura	Lee a través de diversos materiales y superficies.	Lee sólo en superficie.
Precisión	No requiere intervención humana. 100% automático.	Sí requiere intervención humana.
Durabilidad	Soporta ambientes agresivos (intemperie, químicos, humedad, temperatura).	Puede dañarse fácilmente.

Tabla 1 – Cuadro comparativo de las tecnologías RFID y código de barras

Las etiquetas RFID son unos dispositivos de pequeño tamaño, similares a una pegatina, que normalmente se incorporan o adhieren a objetos, animales o, en algunos casos, a personas. Disponen de antenas que les hacen recibir y responder a peticiones por radiofrecuencia desde un emisor-receptor RFID [1].

En la Figura 2 se muestra el esquema general de funcionamiento de un sistema RFID.

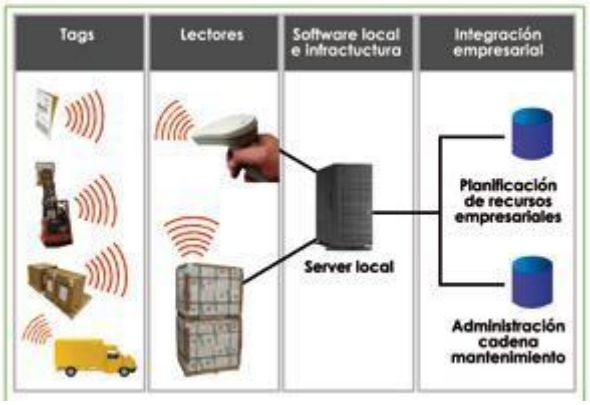


Figura 2 – Esquema general de funcionamiento de un sistema RFID

2.1.2 Arquitectura.

El modo de funcionamiento de un sistema RFID es el siguiente: la etiqueta, que posee los datos de identificación del producto, animal o persona al que se ha incorporado, genera una señal de radiofrecuencia con estos datos. El lector RFID es capaz de leer esa señal, encargarse de procesar la información y traducirla a formato digital a la aplicación específica que utiliza RFID.

Un sistema RFID consta de los siguientes tres componentes:

- ***Etiqueta RFID:*** se compone de una antena, un transductor radio y un chip. El objetivo de la antena es permitir que el chip, que contiene la información, pueda transmitirla. Este chip dispone de una memoria interna que tiene una capacidad dependiente del modelo (puede variar entre decenas y millares de bytes). Existen diferentes tipos de memoria:
    - Sólo lectura: el código de identificación que contiene es único y es personalizado durante la fabricación de la etiqueta, no se puede modificar.
    - De lectura y escritura: la información de identificación puede ser modificada por el lector.
    - Anticolisión. Se trata de etiquetas especiales que permiten que un lector identifique varias de ellas al mismo tiempo.
- ***Lector de RFID:*** se compone de una antena, un transceptor y un decodificador. El lector se encarga de enviar señales de forma periódica para localizar etiquetas dentro de su rango. Al captar la señal de una etiqueta, extrae la información de su identificación y la envía al subsistema de procesamiento de datos.

- ***Subsistema de procesamiento de datos o Middleware RFID:*** proporciona los medios de proceso y almacenamiento de datos [1].

La Figura 3 representa los elementos de los que consta un sistema RFID.

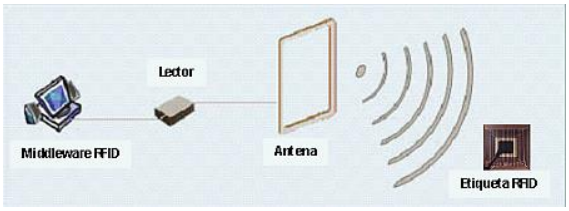


Figura 3 – Elementos de un sistema RFID

2.1.3 Clasificación de los sistemas RFID.

Los sistemas RFID se pueden clasificar en función del rango de frecuencias que utilizan. Por lo tanto, existen cuatro tipos de sistemas (como se muestra en la Figura 4):

- ***LF:*** De baja frecuencia (entre 125 y 134,2 KHz).
- ***HF:*** De alta frecuencia (13,56 MHz).
- ***UHF:*** Frecuencia ultra elevada (entre 868 y 956 MHz).
- ***Microondas:*** (2,45 GHz) [1].

Banda de Frecuencias	Características del Sistema	Ejemplos de Aplicaciones
LF (de 100 a 500 kHz). Típico 125 a 134 kHz. Internacional	Corto alcance. Poca velocidad de transmisión Relativamente económico Gran penetración en los materiales (líquidos) Trabaja bien junto a metales	Control de acceso Identificación de animales Control de inventario EAS (Antirrobo) Llaves de automóvil
HF. Típico 13,56 MHz. Internacional	Corto/medio alcance Velocidad de transmisión media Puede leer a través de líquidos y en entornos húmedos Problemático junto a metales Moderadamento caro Posibilidad de actuar como lector o etiqueta en función del escenario de utilización (NFC)	Control de acceso Tarjetas inteligentes EAS (Antirrobo) Inventario en bibliotecas Gestión de almacén Control de equipajes Gestión de lavandería Identificación de pacientes Pago con el móvil y captura de datos con sólo acercar el móvil (NFC)
UHF (de 400 a 1.000 MHz) Típico 850 - 950 MHz	Largo alcance Alta velocidad de transmisión Mecanismos de anticolisión Problemático con líquidos y metales Problemático en entornos húmedos En metal genera interferencias Moderadamente caro	Gestión de artículos Gestión de la cadena de suministro Gestión de almacén Gestión de expediciones Trazabilidad
Microondas (de 2,4 a 6 GHz)	Medio alcance Características similares a los tags UHF pero con mayor velocidad de transmisión Mayor precio	Control ferroviario Peajes de autopista Localización

Figura 4 – Clasificación de los sistemas RFID

2.1.4 Tarjetas RFID.

Las etiquetas RFID pueden ser de tres tipos: activas, pasivas o semipasivas. Las etiquetas pasivas no necesitan ninguna fuente de alimentación interna, ya que toman esa energía requerida del lector. Sin embargo, las etiquetas activas y las semipasivas necesitan alimentación, que normalmente será suministrada por una pequeña pila.

Las etiquetas RFID más abundantes son las pasivas. Esto se debe a que su fabricación es mucho menos costosa que la de las activas y semipasivas. Además, estas no necesitan batería o fuente de alimentación.

Pese a estas ventajas, existen otros motivos por los que es muy recomendable el uso de las tarjetas activas, ya sea la exactitud, el funcionamiento en ciertos ambientes (cerca del agua, cerca de un metal, etc.) o confiabilidad. Estas características hacen que la utilización de tarjetas activas sea más común cada día [1].

2.1.5 Etiquetas pasivas.

Como se ha comentado anteriormente, las etiquetas pasivas no poseen alimentación eléctrica. Los lectores reciben una señal, la cual induce una pequeña corriente eléctrica que es suficiente para operar el circuito integrado CMOS de la etiqueta, de forma que ésta puede generar y transmitir una respuesta.

La respuesta de una etiqueta pasiva, normalmente, es corta, debido a las preocupaciones por la energía y el coste. El dispositivo suele ser muy pequeño, ya que la ausencia de fuente de alimentación favorece la reducción del tamaño.

En la práctica, las etiquetas pasivas tienen distancias de lectura que varían entre los 10 milímetros hasta cerca de los 6 metros, dependiendo del tamaño de la antena de la etiqueta y de la potencia y frecuencia en la que opera el lector [1].

En la Figura 5 se muestra un ejemplo de etiqueta pasiva.

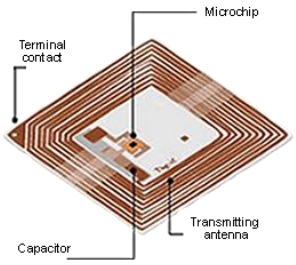


Figura 5 – Etiqueta pasiva

2.1.6 Etiquetas activas.

A diferencia de las etiquetas pasivas, las activas disponen de una fuente de alimentación propia, la cual proporciona corriente a sus circuitos integrados, lo que hace posible propagar la señal al lector. De esta manera, las señales que transmiten las etiquetas activas son más potentes que las pasivas, por lo que, en entornos de mayor dificultad para la radiofrecuencia, serán mucho más eficientes las primeras. Los ejemplos más claros de entornos dificultosos para la transmisión son el agua y el metal.

Suele ser habitual que las etiquetas activas envíen el nivel de batería al lector, ya que, de esta forma, es posible conocer con antelación su estado y se pueden sustituir aquellas que están a punto de agotarse.

El tiempo de vida de las baterías depende de cada modelo de etiqueta y también de la actividad de este, aunque, por lo general, es del orden de años.

La gran mayoría de las etiquetas activas poseen un rango efectivo de unos 100 metros y, en algunos casos, integran sensores de registro de temperatura u otras variables, con lo que es posible monitorizar entornos de alimentación o productos farmacéuticos.

Otros sensores asociados con RFID incluyen humedad, vibración, luz, radiación, temperatura y componentes atmosféricos como el etileno. Además de mucho más rango (500 m), tienen capacidades de almacenamiento mayores (memoria generalmente entre 4 y 32 kB) y la habilidad de guardar información adicional enviada por el transceptor.

En la actualidad, se ha conseguido reducir el tamaño de las etiquetas activas hasta tal punto que se pueden encontrar tarjetas del tamaño de una moneda [1].

A continuación, en la Figura 6 y en la Figura 7 se pueden observar los distintos tamaños y formas que pueden adoptar las etiquetas.

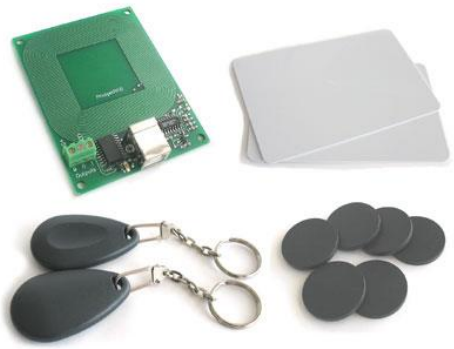


Figura 6 – Distintos tipos de tag

Ventajas e inconvenientes.

La ventaja principal de las etiquetas activas respecto a las pasivas es el mayor rango de lectura de las primeras, ya que el rango de las primeras pueden llegar hasta los 100 metros, mientras que el de las pasivas es de unos pocos metros.

Además, estas tarjetas son mucho más fiables (tienen menos errores) que las pasivas debido a su capacidad de establecer sesiones con el lector.

Mientras que la capacidad de almacenamiento de las tarjetas pasivas se encuentra entre 1 y 4 kB, las activas tienen un máximo de memoria de 32 kB, por lo que estas últimas son recomendables si se va a requerir una gran capacidad de memoria.

Por el contrario, cabe resaltar el precio de las etiquetas activas (entre 30 y 90 €), que es muy superior al de las pasivas (unos pocos céntimos de €), así como la dependencia de alimentación por baterías y su vida útil, que en general es mucho más corta [1].

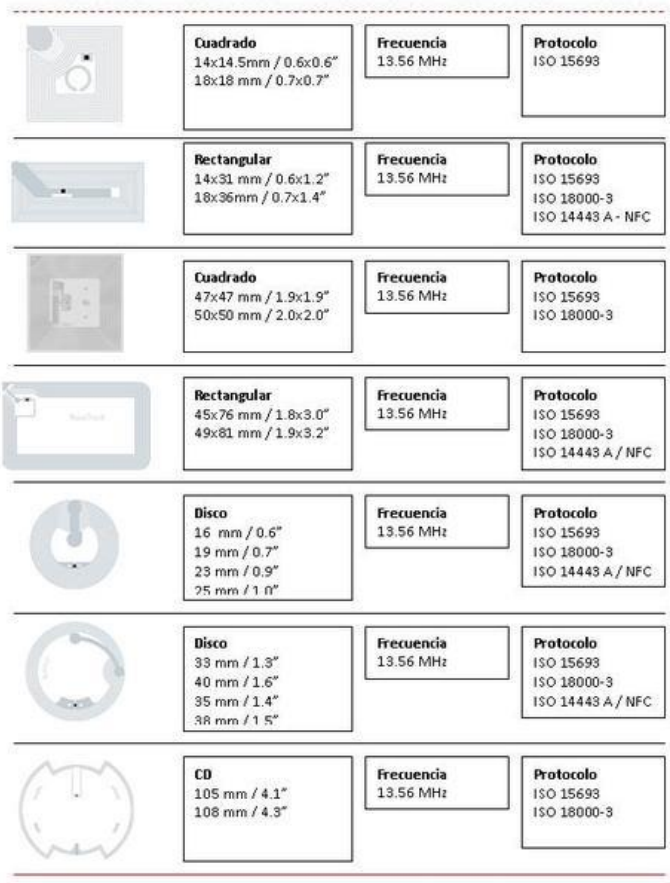


Figura 7 – Formatos de los diferentes tipos de etiquetas

2.1.7 Etiquetas semipasivas.

Las etiquetas semipasivas y las activas tienen en común que ambas poseen una fuente de alimentación propia. Sin embargo, las semipasivas utilizan la energía de esta fuente para alimentar el chip y no para transmitir la señal generada. Para ello, utilizan la energía del lector. Este diseño mejora el tiempo de respuesta de la etiqueta y aumenta el rango de lectura.

Debido a su fuente de energía, los tags semipasivos poseen mayor capacidad de memoria. Utilizando la energía de baterías pequeñas, estas etiquetas se pueden utilizar para controlar las entradas de sensores.

La batería puede permitir al circuito integrado de la etiqueta estar constantemente alimentado y eliminar la necesidad de diseñar una antena para recoger potencia de una señal entrante.

Este tipo de etiqueta tiene una fiabilidad comparable a la de las activas, a la vez que pueden mantener el rango operativo de una pasiva. También suelen durar más tiempo que las activas [1].

2.1.8 Posicionamiento de las etiquetas.

Por lo general, no es estrictamente necesario que la recepción de la energía del lector sea óptima para poder actuar sobre las etiquetas pasivas. Aun así, existen casos en los que la distancia entre ambos elementos y la potencia efectiva emitida es fijada. En situaciones como esta, se necesita conocer en qué estados se puede trabajar de manera óptima con ellos [1].

Se definen los siguientes puntos para especificar la localización de las etiquetas en un objeto marcado:

- ***R (resonance spot):*** A esta distancia, una etiqueta pasiva puede operar de manera óptima dentro del determinado nivel de potencia efectiva radiada (ERP).

- ***L (live spot):*** A esta distancia, una etiqueta pasiva aún puede obtener la energía de un lector para operar dentro del nivel ERP especificado.

- ***D (dead spot):*** A esta distancia, una etiqueta pasiva todavía puede obtener la energía de un lector, pero la potencia obtenida es insuficiente para operar dentro de las etiquetas de nivel ERP especificado [2].

2.1.9 Asociación de etiquetas.

Existen tres tipos de tarjetas RFID en función de la unión que tengan con el objeto identificado:

- ***Etiquetas acoplables:*** Son muy populares debido a que las aplicaciones RFID son amplias y diversas, por lo que requieren tipos flexibles y diferentes de archivos adjuntos. Más del 85% de las etiquetas RFID disponibles en el mercado son de este tipo. Las etiquetas acoplables están diseñadas para ser adheridas en la superficie de los objetos identificados con medios de fijación semipermanentes o temporales permanentes. La forma más popular de estas etiquetas son las sensibles a la presión, comúnmente conocidas como etiquetas inteligentes.

- ***Etiquetas implantables o etiquetas embebidas:*** Estas etiquetas suelen estar diseñadas para su implantación permanente o a largo plazo. Uno de los usos más destacable es el de la rastreabilidad de los animales y de la lectura mecánica de los documentos de viaje (MRTD). El beneficio fundamental de las etiquetas implantables es la unidad entre las etiquetas y sus objetos identificados. Dejando de lado de algunas cuestiones éticas sobre el etiquetado RFID con humanos, los problemas de uso de implantes en humanos o animales pueden ser la irritación de los tejidos y los problemas de migración tisular.

- ***Etiquetas de inserción:*** Las etiquetas de inserción están diseñadas para tener poco o ningún contacto con los objetos identificados. El objetivo principal de las etiquetas es permitir la identificación de ciertos objetos sin proceso de unión y sin alterar los objetos identificados, preservando así el estado original de los objetos. La mayoría de las etiquetas de inserción son etiquetas no adhesivas que se pueden insertar fácilmente en materiales impresos, en sus cubiertas o en otros paquetes de productos [3].

2.1.10 Uso actual.

El uso actual de la tecnología RFID depende en gran parte de las frecuencias que se empleen en los sistemas, del coste del mismo o de su alcance.

La ventaja de los sistemas que utilizan bajas frecuencias es que son más baratos, aunque tienen la desventaja de que su rango de funcionamiento es menor.

Los sistemas que emplean frecuencias más altas proporcionan distancias mayores de lectura y velocidades de lectura más rápidas.

Por lo tanto, los sistemas de baja frecuencia se utilizan en aplicaciones como la identificación de animales (Figura 8), el seguimiento de barricas de cerveza o como llave de automóviles con sistema antirrobo. En ocasiones se insertan en pequeños chips en mascotas, para que puedan ser devueltas a su dueño en caso de pérdida.

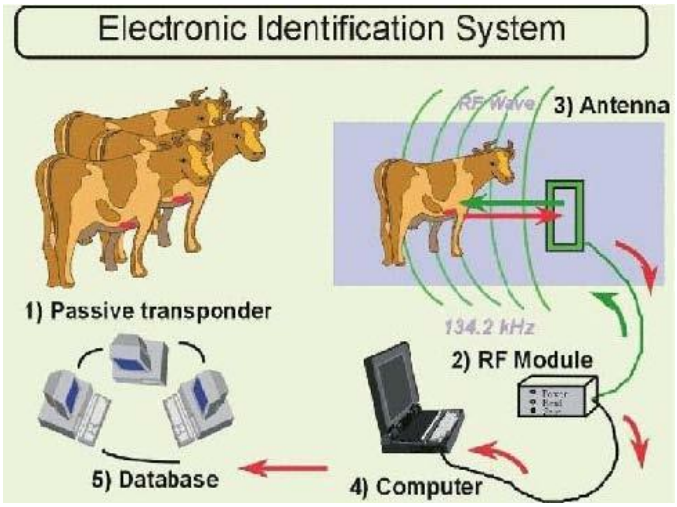


Figura 8 – Ejemplo de tecnología RFID en animales

Las etiquetas RFID de alta frecuencia se utilizan en bibliotecas y seguimiento de libros (Figura 9), de palés, de artículos de ropa, de equipaje en aerolíneas, control de acceso en edificios y, últimamente, en pacientes de centros hospitalarios para hacer un seguimiento de su historia clínica. Un uso extendido de las etiquetas de alta frecuencia es el de identificación de acreditaciones, sustituyendo a las anteriores tarjetas de banda magnética. Sólo es necesario acercar estas tarjetas a un lector para autentificar al portador.

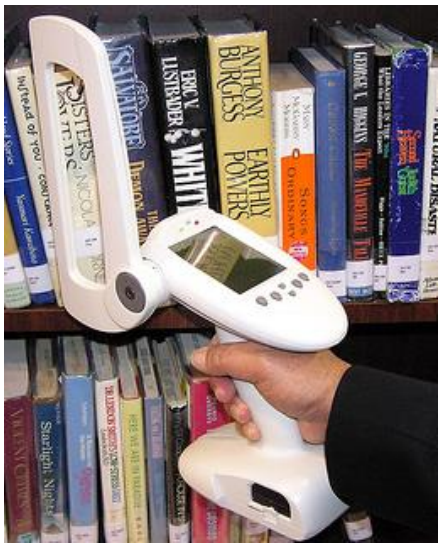


Figura 9 – Ejemplo de tecnología RFID en bibliotecas

Es muy común la utilización de etiquetas RFID de UHF en el entorno comercial, en especial en el seguimiento de palés y envases, en el de camiones y remolques, en sistemas de distribución de uniformidad en hospitales o en ropa.

Las etiquetas RFID de microondas se utilizan en el control de acceso en vehículos de gama alta.

Actualmente, la aplicación más importante de RFID es la logística, como vemos en la Figura 10. El uso de esta tecnología permite tener localizado cualquier producto dentro de la cadena de suministro. En cuanto a la trazabilidad, las etiquetas RFID son de gran utilidad, ya que pueden ser grabadas y poder conocer el tiempo que el producto estuvo almacenado, en qué sitios, etc. De esta manera, se pueden lograr importantes optimizaciones en el manejo de los productos en las cadenas de abastecimiento, teniendo como base el mismo producto, e independizándose prácticamente del sistema de información [1].

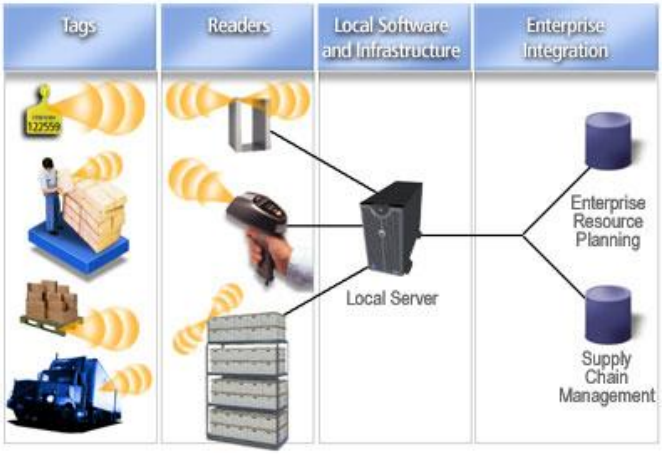


Figura 10 – Ejemplo de tecnología RFID en logística

2.1.11 Aplicaciones potenciales.

Las etiquetas RFID son la alternativa más clara para reemplazar a los códigos de barra, tecnología un tanto arcaica y que cada vez parece más obsoleta. Pese a que el número de ventajas de los sistemas RFID sobre los códigos de barras es grande, probablemente no lleguen a sustituir del todo a estos últimos. El principal motivo para que este reemplazo no se lleve a cabo de forma total es por su coste, puesto que es relativamente alto. Para algunos artículos con un coste más bajo la capacidad de cada etiqueta de ser única se puede considerar exagerado, aunque tendría algunas ventajas tales como una mayor facilidad para llevar a cabo inventarios.

Los códigos RFID son tan largos que cada etiqueta RFID puede tener un código único, mientras que los códigos de barras actuales se limitan a un único código para todos los casos de un producto particular.

Que los códigos RFID sean únicos hace posible que un producto pueda ser seguido de forma individual a la vez que se transporta o se mueve, hasta que llega al consumidor. De esta forma, las compañías pueden combatir el hurto y o que el producto sea extraviado.

Otra propuesta es la de utilizar la tecnología RFID con el fin de comprobar los almacenes. Con esta medida, se podría sustituir al encargado de la caja por un sistema automático que no necesite la captación de códigos de barras. Sin embargo, es bastante improbable que esto se realice si no hay una reducción significativa en el coste de las etiquetas actuales.

Actualmente, se está llevando a cabo una investigación sobre la tinta que se puede utilizar como etiqueta RFID, que reduciría costes de forma significativa. Aun así, faltan todavía algunos años para que esto dé sus frutos [1].

2.2 Arduino.

Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. Se creó para diseñadores, aficionados y cualquier interesado en crear entornos u objetos interactivos.

Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede afectar a aquello que le rodea controlando leds, motores y otros actuadores. El microcontrolador en la placa Arduino se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing) [4].

Sin embargo, es posible utilizar otros lenguajes de programación y aplicaciones populares en Arduino, debido a que usa la transmisión serial de datos soportada por la mayoría de los lenguajes mencionados. Para los que no soportan el formato serie de forma nativa, es posible utilizar un software intermediario que traduzca los mensajes enviados por ambas partes para permitir una comunicación fluida. Algunos ejemplos son:

- 3DVIA Virtools.
- Adobe Director.
- BlitzMax.
- C.
- C++.
- C#.

- Cocoa/Objective-C.
- Flash.
- Gambas.
- Isadora.
- Instant Reality.
- Java.
- Liberlab.
- Mathematica.
- Matlab.
- MaxMSP.
- Minibloq.
- Perl.
- Php.
- Physical Etoys.
- Processing.
- Pure Data.
- Python.
- Ruby.
- Scratch for Arduino (S4A).
- Squeak.
- SuperCollider.
- VBScript.
- Visual Basic .NET.
- VVVV.

Arduino está basado en C y soporta todas las funciones del estándar C y algunas de C++. La sintaxis del lenguaje de programación Arduino es una versión simplificada de C/C++ y tiene algunas diferencias respecto a Processing. Debido a que Arduino está basado en C/C++ mientras que Processing se basa en Java, existen varias diferencias en cuanto a la sintaxis de ambos lenguajes y el modo en que se programa.

Las bibliotecas estándar que ofrece Arduino son las siguientes:

- ***Serial:*** Lectura y escritura por el puerto serie.
- ***EEPROM:*** Lectura y escritura en el almacenamiento permanente.
- ***Ethernet:*** Conexión a Internet mediante "Arduino Ethernet Shield". Puede funcionar como servidor que acepta peticiones remotas o como cliente. Se permiten hasta cuatro conexiones simultáneas.
- ***Firmata:*** Es una biblioteca de comunicación con aplicaciones informáticas utilizando el protocolo estándar del puerto serie.
- ***LiquidCrystal:*** Control de LCDs con chipset Hitachi HD44780 o compatibles. La biblioteca soporta los modos de 4 y 8 bits.
- ***Servo:*** Biblioteca para el control de servo motores. A partir de la versión 0017 de Arduino la biblioteca soporta hasta 12 motores en la mayoría de las placas Arduino y 48 en la Arduino Mega.
- ***SoftwareSerial:*** Comunicación serie en contactos digitales. Por defecto Arduino incluye comunicación sólo en los pines 0 y 1 pero gracias a esta biblioteca puede realizarse esta comunicación con los restantes.
- ***Stepper:*** Control de motores paso a paso unipolares o bipolares.
- ***Wire:*** Envío y recepción de datos sobre una red de dispositivos o sensores mediante Two Wire Interface (TWI/I2C). Las bibliotecas Matrix y Sprite de Wiring son totalmente compatibles con Arduino y sirven para manejo de matrices de diodos LED. También se ofrece información sobre distintas bibliotecas desarrolladas por diversos colaboradores que permiten realizar muchas tareas.

Los usuarios de Arduino tienen la posibilidad de escribir sus propias bibliotecas. Esto permite disponer de código que puede reutilizarse en otros proyectos, mantener el código fuente principal separado de las bibliotecas y la organización de los programas construidos es más clara [5].

Las placas pueden ser hechas a mano o compradas/montadas de fábrica; el software puede ser descargado de forma gratuita. Los ficheros de diseño de referencia (CAD) están disponibles bajo una licencia abierta, así pues el usuario es de adaptarlos a sus necesidades [4].

2.3 ROS.

ROS (siglas de Robot Operating System) es un marco de trabajo para el desarrollo de software para robots.

ROS provee los servicios estándar de un sistema operativo. Algunos de ellos son la abstracción del hardware, el control de dispositivos de bajo nivel, la implementación de funcionalidad de uso común, el paso de mensajes entre procesos y el mantenimiento de paquetes. Este software está basado en una arquitectura de grafos donde el procesamiento toma lugar en los nodos que pueden recibir, mandar y multiplexar mensajes de sensores, control, estados, planificaciones y actuadores, entre otros.

La librería está orientada para un entorno Linux (Ubuntu) aunque también se está adaptando a otros sistemas operativos, considerados como "experimentales".

ROS tiene dos partes básicas: la parte del sistema operativo, ros, como se ha descrito anteriormente, y ros-pkg, una suite de paquetes aportados por la contribución de usuarios que implementan funcionalidades tales como la localización y mapeo simultáneo, planificación, percepción, simulación, etc.

ROS es un software libre bajo términos de licencia BSD. Esta licencia permite libertad para uso comercial e investigador. Las contribuciones de los paquetes en ros-pkg están bajo una gran variedad de licencias diferentes.

Las áreas que incluye ROS son:

- Un nodo principal de coordinación.
- Publicación o subscripción de flujos de datos: imágenes, estéreo, láser, control, actuador, contacto, etc.
- Multiplexación de la información.
- Creación y destrucción de nodos.
- Los nodos están perfectamente distribuidos, permitiendo procesamiento distribuido en múltiples núcleos, multiprocesamiento, GPUs y clústeres.
- Login.

- Parámetros de servidor.
- Testeo de sistemas.

Las áreas que incluirán las aplicaciones de los paquetes de ROS son:

- Percepción.
- Identificación de Objetos.
- Segmentación y reconocimiento.
- Reconocimiento facial.
- Reconocimiento de gestos.
- Seguimiento de objetos.
- Egomoción.
- Comprensión de movimiento.
- Estructura de movimientos (SFM).
- Visión estéreo: percepción de profundidad mediante el uso de dos cámaras.
- Movimientos.
- Robots móviles.
- Control.
- Planificación.
- Agarre de objetos. [6]

Estos son los conceptos básicos que hay que conocer de ROS:

- ***Nodos***: Los nodos son unos archivos ejecutables dentro de un paquete de ROS que utilizan para comunicarse con otros nodos.
- ***Mensajes:*** Son unos tipos de dato de ROS que se utilizan durante la suscripción y publicación de un topic.
- ***Topics:*** Los nodos pueden publicar mensajes a través de un topic o suscribirse a ellos para recibir mensajes.

- ***roscore:*** Primer comando que se ha de ejecutar cuando se utiliza ROS.
- ***rosrun:*** Hace posible usar el nombre del paquete para ejecutar directamente un nodo.

3 DISEÑO DE LA SOLUCIÓN.

En este capítulo se va a explicar el desarrollo que se ha llevado a cabo para poder realizar este proyecto. Por un lado, se describirán los elementos hardware del sistema, sus características, funcionalidades y conexionado. Por el otro, se detallará la programación utilizada en el software de Arduino y los comandos que se han de ejecutar en ROS para poder conectar el sistema.

3.1 Hardware.

Para la realización de este trabajo van a ser necesarios tres tipos de componentes: una antena que sea capaz de leer las tarjetas, un microcontrolador que descodifique la información y otro microcontrolador que procese el flujo de información y ejecute las acciones que hagan funcionar correctamente el sistema. Para ello, se han buscado los dispositivos que mejor se adecúen a este proyecto, en relación a su funcionalidad, características y coste.

3.1.1 Módulo RFID SM130.

Se necesita un microcontrolador que descodifique la información que le llega al sistema a través de la antena. Para ello se utilizará el módulo RFID SM130.

El SM130 (Figura 11) es un módulo de 28 pines que incluye todos los componentes necesarios para una Mifare RFID 13.56 MHz, además de una antena PCB.

Figura 11 – Módulo RFID SM130

El módulo se comunica a través UART o I2C con protocolos simples. También cuenta con dos entradas de uso general y dos salidas de propósito general para los interruptores, relés, etc. [7].

El módulo Mifare SM130 13.56 MHz RFID es la primera solución de bajo coste de la industria para aplicaciones Mifare. Este módulo soporta tarjetas Mifare Classic de 1Kbyte, Mifare Classic de 4Kbyte y Mifare Ultralight (512 bits).

Las tarjetas Mifare Classic (a menudo llamadas tarjetas inteligentes sin contacto) son las más utilizadas y se consideran las tarjetas inteligentes sin contacto de mayor éxito. La familia Mifare Classic es utilizada en aplicaciones RFID donde se requiere una recogida rápida de datos y de muy alta seguridad. Esta familia de etiquetas tiene una rápida velocidad de comunicación (106 Kbit/s) y utiliza técnicas de encriptación muy fuertes. No es posible copiar o modificar el contenido de las etiquetas Mifare Classic sin la clave correcta, una vez que ésta se protege.

Por tanto, Mifare es ideal para aplicaciones de dinero electrónico, acceso seguro, almacenamiento de datos y sistemas de recolección de datos rápida. Además, la tecnología de antena impresa hace que sea posible encontrar etiquetas Mifare muy finas y de bajo coste. Mifare es la solución perfecta para muchas aplicaciones, como por ejemplo, su utilización en tarjetas para el transporte público, en tarjetas para el control de acceso, en juegos, banca, salud, teléfono público, gobierno, internet, contadores inteligentes, etc.

El propio módulo realiza todas las operaciones de demodulación, decodificación, y encriptación. Los usuarios no necesitan conocer el concepto de RFID, el único conocimiento necesario es saber cómo controlar el módulo de control externo sobre UART o bus I2C con protocolos simples, que se explica en hojas de datos de dispositivos [8].

En la Figura 12 se muestra el PinOut del módulo y en la Tabla 2 la información del propio.

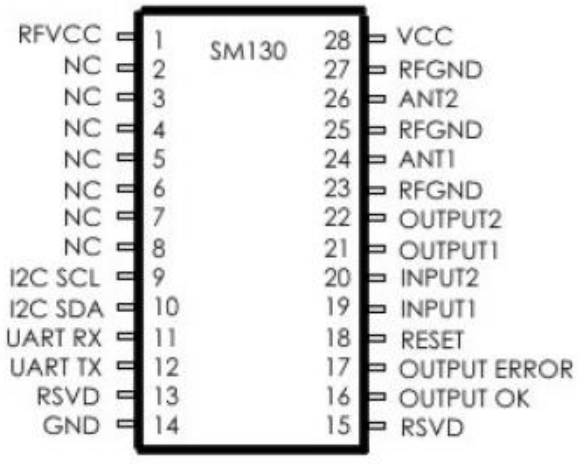


Figura 12 – PinOut del módulo RFID SM130

Pin #	Pin Name	Description
1	RFVCC	5V Supply Voltage – should be connected VCC pin externally
2	NC	No Connection
3	NC	No Connection
4	NC	No Connection
5	NC	No Connection
6	NC	No Connection
7	NC	No Connection
8	NC	No Connection
9	I2C SCL	I2C Clock
10	I2C SDA	I2C Data
11	UART RX	UART RX pin of SM130 (0 – 5V CMOS)
12	UART TX	UART TX pin of SM130 (0 – 5V CMOS)
13	RSVD	Reserved. Do not connect this pin
14	GND	Ground
15	RSVD	Reserved. Do not connect this pin
16	OUTPUT OK	Status Pin .If High communication established with Mifare® tag successfully
17	OUTPUT ERROR	Status Pin .If High communication could not be established with Mifare® tag successfully
18	RESET	Hardware Reset pin. If this pin held high SM130 will be in reset mode. A short logic high (pulse) will reset SM130
19	INPUT1	TTL Input pin. Its status can be access ed via UART or I2C bus. Not mandatory for Mifare® applications just there to expand inputs of a complete system
20	INPUT2	TTL Input pin. Its status can be access ed via UART or I2C bus. Not mandatory for Mifare® applications just there to expand inputs of a complete system
21	OUTPUT1	TTL Output pin. It can be written logic 0 or logic 1 via UART or I2C bus. Not mandatory for Mifare® applications just there to expand outputs of a complete system
22	OUTPUT2	TTL Output pin. It can be written logic 0 or logic 1 via UART or I2C bus. Not mandatory for Mifare® applications just there to expand outputs of a complete system
23	RFGND	Ground - Should be connected to ground externally and for better performance can be connected to antenna ground plane
24	ANT1	Antenna Pin to drive and demodulate. Should be connected to one of the symmetrical antenna/inductor end
25	RFGND	Ground - Should be connected to ground externally and for better performance can be connected to antenna ground plane
26	ANT2	Antenna Pin to drive and demodulate. Should be connected to one of the symmetrical antenna/inductor end
27	RFGND	Ground - Should be connected to ground externally and for better performance can be connected to antenna ground plane
28	VCC	5V Supply Voltage – should be connected VCC pin externally

Tabla 2 – Información del PinOut del módulo RFID SM130

3.1.2 Plataforma de Evaluación para el Módulo RFID SM130.

La información que decodifica el SM130 tiene que ser recogida de alguna manera. Para recoger esos datos, se requiere una plataforma de evaluación para el módulo. Además, esta placa permite conectar también el microcontrolador de Arduino con el módulo SM130.

SparkFun RFID Evaluation Shield – 13.56MHz (Figura 13) es una plataforma de evaluación para el módulo RFID SM130. Esta plataforma incluye una cabecera Xbee (en este proyecto no se va a utilizar) y una antena PCB. También dispone de un layout para ser utilizado por un microcontrolador de Arduino. Al tener una antena integrada, esta placa también puede ser utilizada como una antena para el módulo RFID SM130.

La plataforma de evaluación viene implementada con interruptores, resistencias y LEDs, pero no incluye ni el módulo RFID ni la placa Arduino, que deben ser conectadas más tarde mediante soldaduras [9].

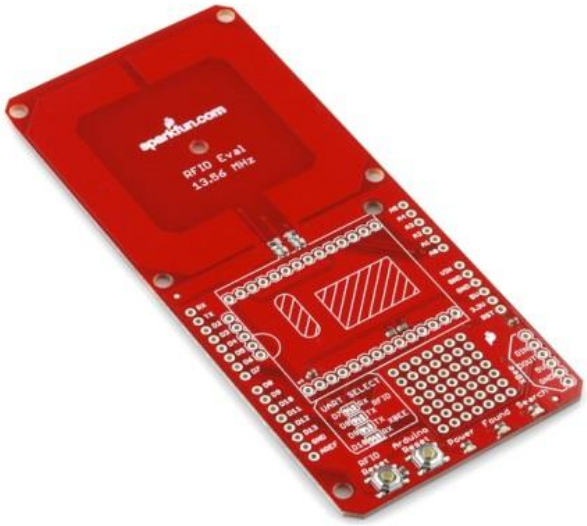


Figura 13 - Plataforma de Evaluación para el Módulo RFID SM130

3.1.3 Arduino Mega 2560.

Arduino Mega 2560 es una actualización de Arduino Mega, al cual remplaza; es una placa electrónica basada en el Atmega2560. Cuenta con 54 pines digitales de entrada / salida (de los cuales 15 se pueden utilizar como salidas PWM), 16 entradas analógicas, 4 puertos serie, un oscilador de 16MHz, una conexión USB, un conector de alimentación, un conector para la programación ICSP y un botón de reset. Esta placa contiene todo lo necesario para dar apoyo al microcontrolador; basta con conectarlo a un ordenador con un cable para comenzar a trabajar con él. La placa Mega es compatible con la mayoría de las plataformas diseñadas para el Arduino Duemilanove o Diecimila.

Microcontrolador	ATmega2560
Tensión de funcionamiento	5V
Tensión de entrada (recomendada)	7-12V
Tensión de entrada (límites)	6-20V
Pines digitales E/S	54 (de los que 15 proporcionan salida PWM)
Pines de entrada analógicos	16
Corriente DC por Pin E/S	40 mA
Corriente DC por Pin de 3.3V	50 mA
Memoria Flash	256 KB (de los que 8 KB son utilizados por el gestor de arranque)
SRAM	8 KB
EEPROM	4 KB
Velocidad del reloj	16 MHz

Tabla 3 – Tabla resumen de las características de la placa Arduino Mega 2560

Arduino Mega puede ser alimentado a través de la conexión USB o con una fuente de alimentación externa. La fuente de alimentación se selecciona automáticamente.

La alimentación externa (no USB) puede venir con un adaptador de CA a CC o la batería. El adaptador se puede conectar a través del conector de alimentación de la placa mediante un enchufe de 2.1mm. Los cables desde una batería se pueden insertar en los cabezales de pin GND y Vin del conector de alimentación.

La placa puede funcionar con un suministro externo de 6 a 20 voltios. Si se suministra con menos de 7V, sin embargo, el pin de 5V puede suministrar menos de cinco voltios y la junta puede ser inestable. Si se utiliza más de 12 V, el regulador de voltaje se puede sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios.

Los pines de alimentación son los siguientes:

- ***VIN:*** Tensión de entrada para la placa Arduino cuando se utiliza una fuente de alimentación externa se puede suministrar tensión a través de este pin.

- ***5V:*** Pin que proporciona una tensión regulada de 5V. La placa puede ser alimentada desde la toma de alimentación de CC (7 - 12 V), desde el conector USB (5V), o a través del pin VIN (7-12V). El suministro de tensión a través de los pines de 5V o 3.3V no pasa por el regulador, y puede dañar la placa. No se aconseja esto último.

- ***3V3:*** Pin que suministra una salida de 3.3 V. La corriente máxima que puede circular es de 50 mA.

- ***GND:*** Pines de tierra.

- ***IOREF:*** Este pin de la placa Arduino proporciona la tensión de referencia a la que opera el microcontrolador.

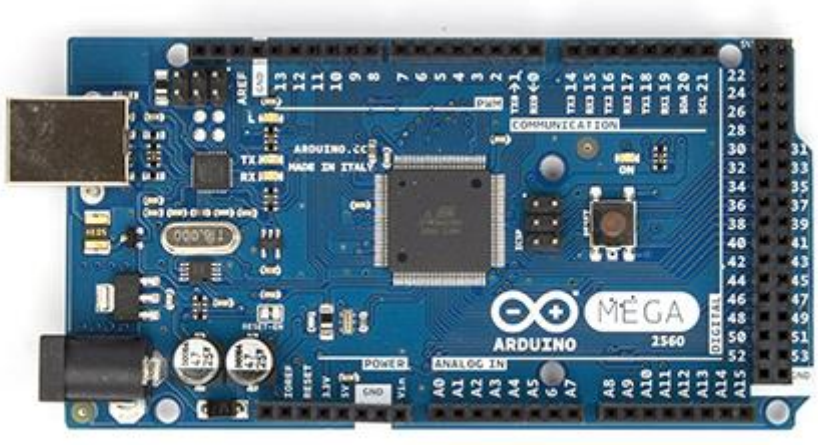


Figura 14 – Arduino Mega 2560

La placa Arduino Mega 2560 (Figura 14) dispone de una memoria flash de 256 KB para el almacenamiento de código (8 KB se usan para el gestor de arranque), 8 KB de SRAM y 4 KB de EEPROM.

Cada uno de los 54 pines digitales que se encuentran en la placa pueden ser utilizados como entrada o salida, utilizando las funciones pinMode(), digitalWrite(), y digitalRead().

Estos pines funcionan a 5V. Cada pin puede proporcionar o recibir un máximo de 40 mA y tiene una resistencia de pull-up (desconectado por defecto) de 20 a 50 kOhm.

Además, algunos pines tienen funciones especiales:

- ***Serial: Pin 0(RX), Pin 1(TX):*** Se utiliza para recibir (RX) y transmitir (TX) datos en serie TTL.
- ***Serial 1: 19 (RX) y 18 (TX):*** Se utiliza para recibir (RX) y transmitir (TX) datos en serie TTL.
- ***Serial 2: 17 (RX) y 16 (TX):*** Se utiliza para recibir (RX) y transmitir (TX) datos en serie TTL.
- ***Serial 3: 15 (RX) y 14 (TX):*** Se utiliza para recibir (RX) y transmitir (TX) datos en serie TTL.
- ***Interrupciones externas:*** Los pines 2 (interruptor 0), 3 (interruptor 1), 18 (interruptor 5), 19 (interruptor 4), 20 (interruptor 3) y 21 (interruptor 2) pueden configurarse para activar una interrupción, un flanco ascendente o descendente, o un cambio en el valor.
- ***PWM:*** Los pines que se encuentran entre el pin 2 y el 13 (ambos inclusive) y los pines 44, 45 y 46 pueden proporcionar una salida PWM de 8 bits con la función analogWrite().
- ***SPI:*** El pin 50 (MISO), el 51 (MOSI), el 52 (SCK) y el 53 (SS) soportan la comunicación SPI utilizando la librería SPI.
- ***LED:*** Hay un LED incorporado conectado al pin digital 13. Cuando el pin es activado, el LED se enciende, cuando es desactivado, se apaga.
- ***TWI:*** Los pines 20 (SDA) y 21 (SCL) soportan la Comunicación TWI utilizando la librería Wire.

Existen un par de pines que también tienen funcionalidades especiales:

- ***AREF***: Tensión de referencia para las entradas analógicas. Se utiliza con la función analogReference().

- ***Reset:*** Resetea el microcontrolador. Normalmente se utiliza para añadir un botón de reinicio.

Arduino Mega2560 tiene una serie de pines para comunicarse con otros microcontroladores, con un ordenador, o con otro Arduino. Esta placa ofrece cuatro puertos serie. Además, proporciona un puerto com virtual para software. El software de Arduino incluye un monitor del puerto serie que permite enviar datos desde y hacia la placa. Los LEDs RX y TX parpadean cuando se están transmitiendo datos a través de la placa.

La biblioteca SoftwareSerial permite la comunicación en serie en cualquiera de los pines digitales de la placa.

Otras comunicaciones que soporta Arduino Mega2560 son las comunicaciones TWI y SPI. El software de Arduino incluye una librería Wire para simplificar el uso del bus TWI.

Arduino Mega se puede programar con el software propio de Arduino. Este modelo viene precargado con un gestor de arranque que le permite cargar nuevo código sin el uso de un programador de hardware externo. Se comunica mediante el protocolo original STK500 (referencia, archivos de cabecera C) [10].

3.1.4 Conexionado.

Después de elegir los dispositivos que mejor se adecúan al sistema, se va a indicar las conexiones a realizar entre ellos.

En primer lugar, se conecta el módulo RFID SM130 a su plataforma de evaluación. Para ello, se deben soldar todos los pines del módulo al lugar que tiene reservado en la plataforma de evaluación, a excepción del pin 15, que no ha de conectarse.

En segundo y último lugar, se procede a conectar la plataforma de evaluación a la placa Arduino Mega 2560.

A continuación, se muestra una tabla que resume las conexiones entre estos dos elementos:

Plataforma de evaluación	**Arduino**
RX	RX (Pin 0)
TX	TX0 (Pin 1)
D8	TX3 (Pin 14)
D7	RX3 (Pin 13)
GND	GND
5V	5V
RST	Reset

Tabla 4 – Conexiones entre la placa Arduino y la plataforma de evaluación

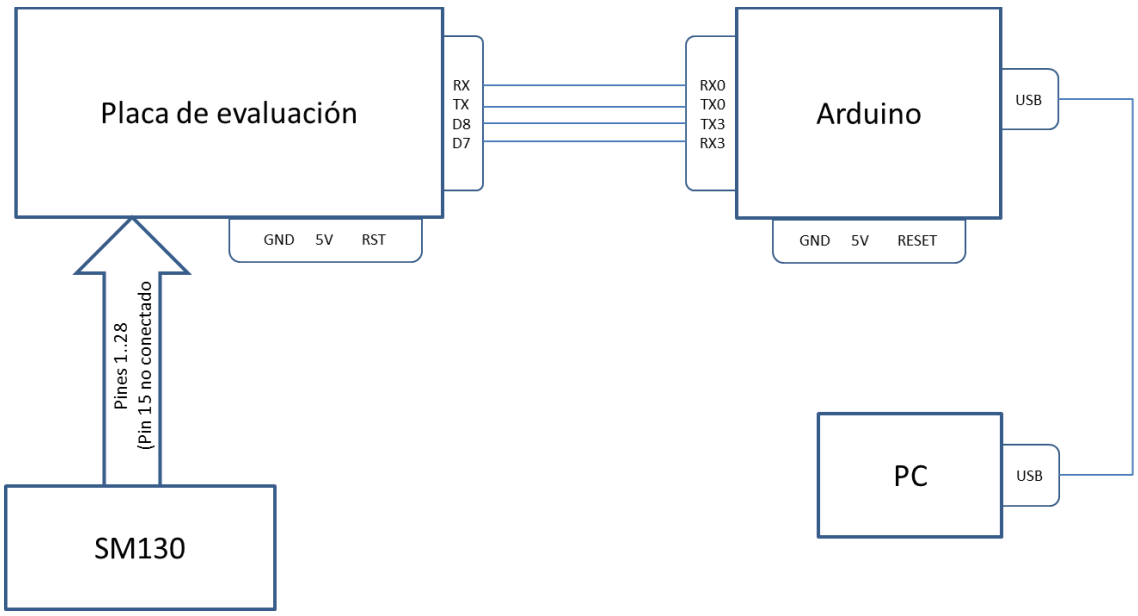


Figura 15 – Diagrama de conexiones

3.2 Software.

Una vez detallados los elementos que van a componer el proyecto, se va a describir el software utilizado en él. Este va a constar de dos aplicaciones: el software de Arduino y ROS.

3.2.1 Arduino.

El programa que se debe implementar en la tarjeta Arduino debe seguir el siguiente flujo:

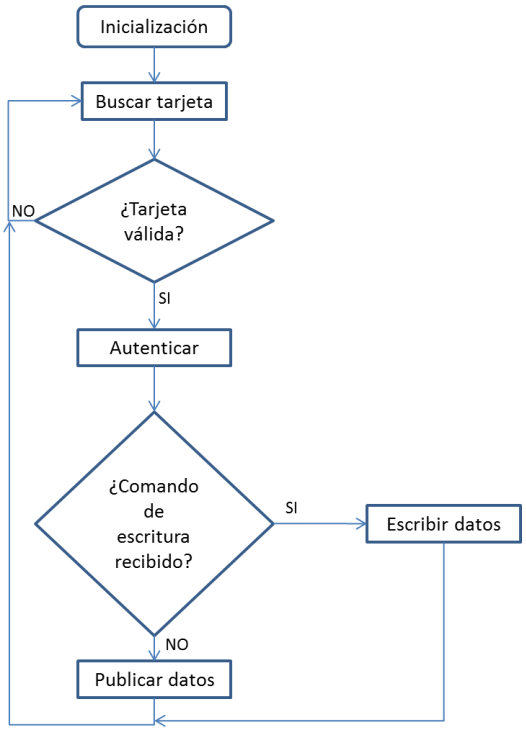


Figura 16 – Diagrama de flujo del sistema

Para ello, la programación que va a requerir la tarjeta Arduino es la siguiente:

En primer lugar, se ha de crear una función que inicialice el programa. Esta será la función setup. Dentro de ella, se encontrará la función set_baud_rate , que selecciona el ratio de baudios a los que se va a trabajar. Además, en setup, se van a inicializar dos puertos serie: uno de ellos va a servir para la transmisión y recepción de datos, el otro, para depurar. También se va a inicializar el nodo de ROS y se va a suscribir a un topic. Por último, se va a llamar a la función halt, que detiene la búsqueda de tags.

A continuación, se crea el bucle principal, que recibe el nombre de loop. Dentro de esta función, se llama a la función seek, la cual envía un comando para que el lector comience a buscar tarjetas en su campo. Posteriormente, se llama a las funciones parse (se encarga de leer el puerto serie y, si se ha detectado una tarjeta, guardar su ID) y detect_tag (analiza si se detecta una tarjeta válida en el campo del lector y el tipo de etiqueta que es).

Una vez que encuentra una tarjeta, el programa va a verificar que el tipo de etiqueta detectada es correcta (Mifare de 1K, de 4K o Ultralight). Si es así, el programa pasa a autenticar la tarjeta, lee los datos que contiene y los publica. Cuando ha terminado de publicarlos, el lector volverá a buscar en su rango de funcionamiento tarjetas válidas.

Por lo tanto, si una etiqueta es detectada, se analiza de qué tipo es y se llama a su función read_serial correspondiente. La función read_serial_1K es la encargada de leer los datos de los bloques de una tarjeta Mifare Classic de 1 Kb de memoria, autenticarlos y mostrarlos. Las funciones read_serial_4K y read_serial_Ultra son análogas a la función read_serial_1K, con la salvedad de que la primera se llama para tarjetas Mifare Classic de 4 Kb de memoria y la segunda para Mifare Ultralight. La función read_data va a ser la que envíe el comando de lectura al microcontrolador para que se lean los datos de la tarjeta. Los bloques de la etiqueta se autenticarán con la función authenticate. Por último, la función publish_data será la encargada de publicar los datos de la tarjeta y mostrarlos.

También existe la posibilidad de escribir datos en las etiquetas, por lo que, tras la inicialización, se puede mandar un comando con los datos que se quieran escribir y, una vez que el lector detecta una etiqueta, los nuevos datos se escribirán en ella, se autenticarán y se publicarán. Esta acción se lleva a cabo con la función write_msg, que va a ser la encargada de escribir nuevos datos en la tarjeta, siempre y cuando se envíe el comando de nueva escritura.

Todas estas funciones pueden verse con mayor detalle en el capítulo 8 (Anexos).

3.2.2 ROS.

Tras cargar el programa en la placa Arduino Mega 2560, se va a proceder a utilizar ROS para que toda la arquitectura quede conectada.

El primer paso es la creación de un paquete y su construcción. Dentro del paquete, se ha definido un mensaje con dos campos, el ID y los datos de la tarjeta. Ambos campos se definen como uint8 (unsigned 8-bit int), pero el del ID tendrá siempre el mismo tamaño (8 bytes), ya que el ID de una tarjeta Mifare no varía. Por su parte, el campo del vector de datos será de la longitud que se desee.

Una vez creado, se debe utilizar el comando “roscore” para arrancar el rosmaster y así poder utilizar los canales de comunicación que proporciona ROS.

De esta forma, ya se está en disposición de lanzar el programa en ROS y poder utilizar el sistema RFID.

4 RESULTADOS OBTENIDOS.

4.1 Introducción.

Una vez implementado el sistema, en este capítulo se va a comentar las pruebas realizadas. Su objetivo es el de verificar que los dispositivos adquiridos cumplen con los requisitos de reconocimiento de etiquetas y con la capacidad para leer y escribir datos en ellas correctamente.

4.2 Detección de etiquetas.

El objetivo es comprobar que el lector detecta e identifica las etiquetas que se encuentran dentro de su zona de funcionamiento. También se van a analizar los distintos resultados obtenidos en función de la posición que adopte la etiqueta al entrar en el rango de lectura.

Se ha probado a colocar una tarjeta cerca del lector y este ha sido capaz de detectarla e identificarla. Acto seguido, se ha ejecutado la misma acción con otra tarjeta distinta, y el resultado es el mismo.

Además, se ha intentado situar dos etiquetas a la vez en el lector RFID, pero sólo ha sido capaz de leer una de ellas.

En cuanto al posicionamiento de las etiquetas al entrar en el rango de cobertura, no existe diferencia entre que la tarjeta se sitúe paralela al lector o lo haga en otra posición distinta, ya que la tarjeta es detectada e identificada igualmente, salvo en el caso de que entre de manera perpendicular en el campo, puesto que si tiene un ángulo cercano a los 90º, la etiqueta no es detectada.

4.3 Rango de lectura.

En este apartado, el objetivo es el de conocer el rango de cobertura del lector RFID, es decir, la distancia máxima en la que los datos de las etiquetas se leen de manera óptima.

Se ha colocado una tarjeta dentro del rango de lectura y, poco a poco, se ha ido aumentando la distancia entre el tag y el lector hasta que este último no era capaz de detectar dicha etiqueta o se producía un error en la lectura. La distancia máxima obtenida ha sido 5,5 cm.

4.4 Tiempo de lectura.

En esta ocasión, se va a tratar de hallar el tiempo que tarda el sistema en leer la información contenida en una tarjeta RFID.

Se han realizado varias lecturas de la misma tarjeta, cada vez a una distancia diferente, y se han tomado los tiempos de lectura. Aproximadamente, el tiempo que necesita el sistema para leer los datos del tag (47 bloques de memoria) es de un segundo. La distancia a la que se sitúa la etiqueta no influye.

4.5 Tiempo de escritura.

En este último apartado, el objetivo es el de averiguar el tiempo que se tarda en escribir los datos en las etiquetas.

El proceso a realizar es muy similar al del apartado anterior, pero esta vez no se realiza una lectura, esta vez se escribirán datos en la tarjeta. El tiempo de escritura es prácticamente el mismo que en el caso anterior, en torno a un segundo. De nuevo, la distancia a la que se coloca el tag no modifica el tiempo de escritura.

5 CONCLUSIONES Y LÍNEAS FUTURAS.

5.1 Conclusiones.

En este trabajo se ha realizado el estudio y la implementación del control de una tarjeta en un sistema RFID. Este sistema trabajará en la frecuencia de 13.56 MHz.

El proyecto nace de la idea de poder detectar e identificar etiquetas RFID que posean datos e información, la cual pueda ser utilizada para distintas actividades.

Como ya se ha comentado en el primer capítulo, las aplicaciones que utilizan las tecnologías de identificación automática cobran cada vez más importancia y están más presentes cada día. Esto se debe, en gran parte, a las numerosas ventajas que ofrece respecto a otras tecnologías de identificación. Un ejemplo de ello son los sistemas de identificación por radiofrecuencia RFID.

Un sector en el que está creciendo mucho la aplicación de sistemas RFID es en el de la robótica, ya que se necesitan nuevas formas de interactuar con los robots.

El primer paso para llevar a cabo este proyecto es el de conocer a fondo qué es y cómo funciona la tecnología RFID, cuáles son sus limitaciones, sus ventajas y los usos que se le puede dar.

Seguidamente, se ha realizado una búsqueda de los dispositivos que mejor se adaptasen a nuestros requerimientos, intentando siempre que estos elementos cumpliesen las funcionalidades necesarias, que sus características se adecuasen al sistema y que su coste fuese lo menor posible sin que esto influyese negativamente en su funcionamiento. Estos dispositivos son los mencionados en el Capítulo 3, un elemento que disponga de una antena, la cual sea capaz de leer etiquetas (placa de evaluación del módulo SM130), un microcontrolador que descodifique la información (módulo RFID SM130) y otro microcontrolador que procese el flujo de información y ejecute las acciones que hagan funcionar correctamente el sistema (Arduino Mega 2560).

A continuación, se ha llevado a cabo el conexionado de los elementos hardware del sistema. Esto no habría sido posible sin la ayuda de los datasheet de cada uno de los dispositivos.

Una vez elegido y conectado todo el hardware del sistema, se procede a programar la placa Arduino Mega 2560 para que el sistema sea capaz de detectar una tarjeta que entra en su rango de cobertura, lea la información que contiene y, si se desea, se pueda escribir nuevos datos en dicha etiqueta.

Por último, se comunica el dispositivo RFID con el resto de la arquitectura mediante el software de ROS, por lo que, de esta forma, ya se tiene el sistema implementado y listo para utilizarse.

En cuanto a los objetivos, se ha logrado que el sistema sea capaz de identificar tarjetas RFID, de leer su información, mostrarla y también de escribir nuevos datos en las mismas. Además, se ha conseguido que toda la arquitectura opere correctamente y pueda ser utilizada por cualquier usuario, por lo que se han cumplido los objetivos propuestos.

5.2 Líneas futuras.

El principal inconveniente de este sistema RFID es que, a la hora de publicar los datos en el software de ROS, existen algunos problemas por los que no se muestran más de un número de datos (unos 80 bytes). Esto se debe a que el módulo rosserial limita el programa, aunque se puede solucionar modificando su configuración. Por tanto, esta podría ser una mejora a realizar en el fututo.

Es de esperar que el coste de las etiquetas cada vez sea menor, por lo que, en un futuro, se podrían obtener tarjetas con una capacidad de memoria mucho mayor a un precio más asequible.

Las aplicaciones que se pueden llevar a cabo utilizando el sistema implementado son muy diversas, puesto que permite identificar etiquetas RFID y obtener su información de manera automática. Además, el diseño de funcionamiento está pensado para poder rápidamente adaptarlo a nuevas tarjetas con otros formatos.

Una de estas aplicaciones puede ser la integración de este sistema en un robot social. De esta forma, el robot podría identificar una tarjeta que posea cierta información, con lo que una persona estaría interactuando con el robot. Un ejemplo de aplicación con esta integración sería la del desarrollo de un juego para niños o ancianos; éste podría consistir en que el robot lanzase una pregunta y la persona utilizase una de las tarjetas de las que dispone (en las que previamente se habría guardado la información correspondiente) para contestar a esa pregunta. Estos juegos podrían tratar desde la identificación de objetos (el robot pregunta por un objeto y la respuesta sería la etiqueta que contuviese la información de ese objeto) hasta la práctica de un idioma extranjero (el robot dice una palabra en español y la persona contesta acercando la tarjeta que contiene la información de esa palabra en inglés).

6 PRESUPUESTO.

En este capítulo se va a estimar los costes asociados al proyecto.

En el cálculo del presupuesto de este trabajo se han incluido, tan sólo, los costes asociados al material.

En los precios que figuran en cada elemento de la tabla 5 no se ha añadido el IVA, por lo cual, se añade posteriormente para calcular el coste final.

Hardware	**Cantidad**	**Precio unitario**	**Total**
Arduino Mega 2560 (con cable USB)	1	34.89 €	34.89 €
Módulo RFID SM130	1	26.43 €	26.43 €
Placa de evaluación del módulo SM130	1	17.60 €	17.60 €
Tarjetas Mifare Classic 1K	4	3.49 €	13.96 €
Coste total sin IVA			92.88 €
IVA (16%)			14.86 €
Coste total con IVA			**107.74 €**

Tabla 5 – Costes del material

Así pues, el coste total del material asciende a 107.74 euros

7 REFERENCIAS BIBLIOGRÁFICAS.

[1] RFID – Wikipedia. Disponible en http://es.wikipedia.org/wiki/RFID. [Consulta: 18 de Febrero de 2015].

[2] The Art and Science of RFID Tagging - RFID Asia. Disponible en http://www.rfid-asia.info/2006/12/art-and-science-of-rfid-tagging.htm. [Consulta: 18 de Febrero de 2015].

[3] RFID Tag Attachments - RFID Asia. Disponible en http://www.rfid-asia.info/2006/12/rfid-tag-attachments.htm. [Consulta: 18 de Febrero de 2015].

[4] Home page – Arduino. Disponible en http://www.arduino.cc/es/pmwiki.php?n=. [Consulta: 18 de Febrero de 2015].

[5] Arduino – Wikipedia. Disponible en https://es.wikipedia.org/wiki/Arduino. [Consulta: 18 de Febrero de 2015].

[6] Sistema Operativo Robótico – Wikipedia. Disponible en http://es.wikipedia.org/wiki/Sistema_Operativo_Robótico. [Consulta: 18 de Febrero de 2015].

[7] RFID Module-SM130 Mifare (13.56 MHz) – Sparkfun. Disponible en https://www.sparkfun.com/products/10126. [Consulta: 18 de Febrero de 2015].

[8] RFID Module-SM130 Mifare (13.56 MHz) – Sparkfun. Disponible en https://www.sparkfun.com/datasheets/Sensors/ID/SM130.pdf. [Consulta: 18 de Febrero de 2015].

[9] RFID Module-SM130 Mifare (13.56 MHz) – Sparkfun. Disponible en https://www.sparkfun.com/products/10406. [Consulta: 18 de Febrero de 2015].

[10] Arduino Mega 2560 – Arduino. Disponible en http://arduino.cc/en/Main/ArduinoBoardMega2560. [Consulta: 18 de Febrero de 2015].

8 ANEXOS.

Programación de la placa Arduino Mega 2560.

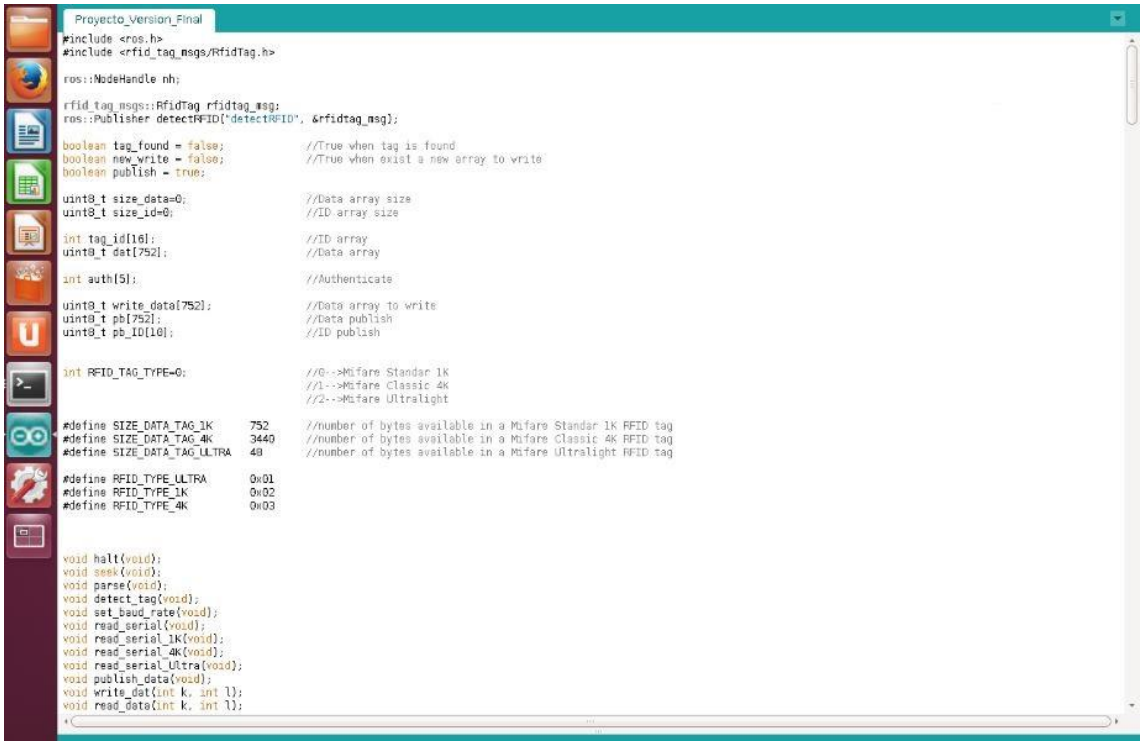


Figura 17 – Librerías, variables y declaración de las funciones de la programación de Arduino

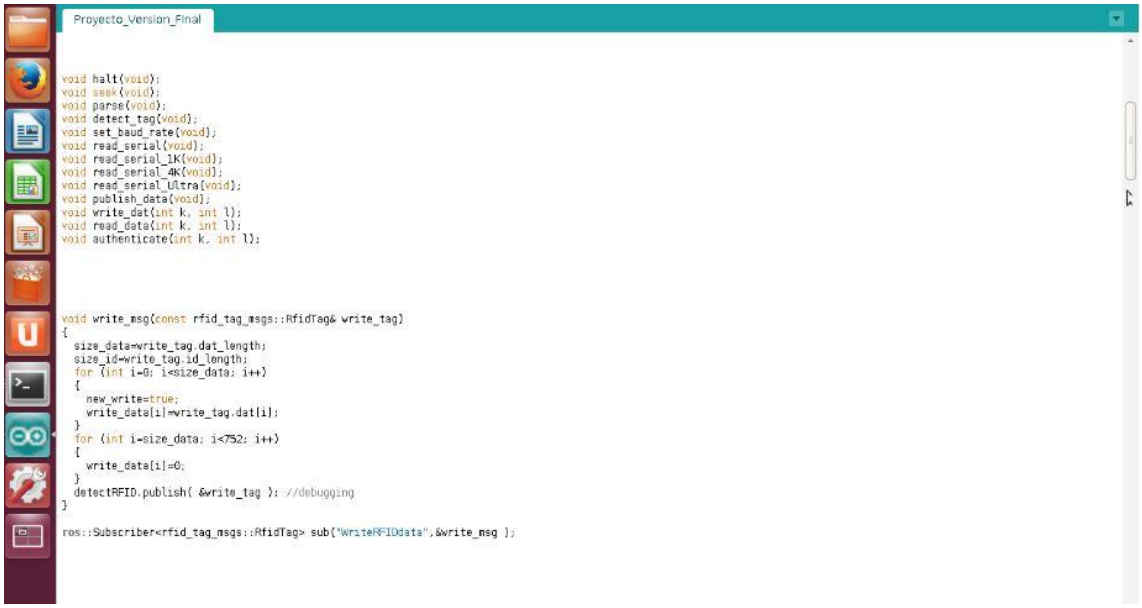


Figura 18 – Declaración de las funciones de la programación de Arduino y función write_msg

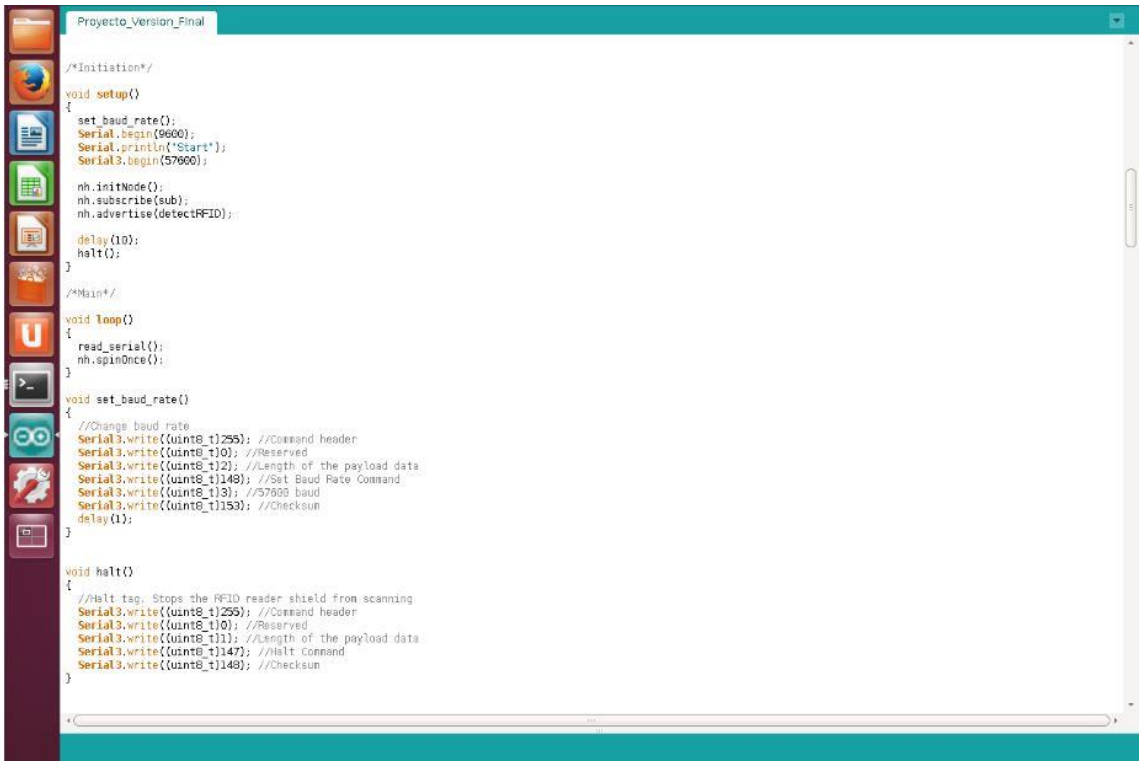


Figura 19 – Funciones setup, loop, set_baud_rate y halt

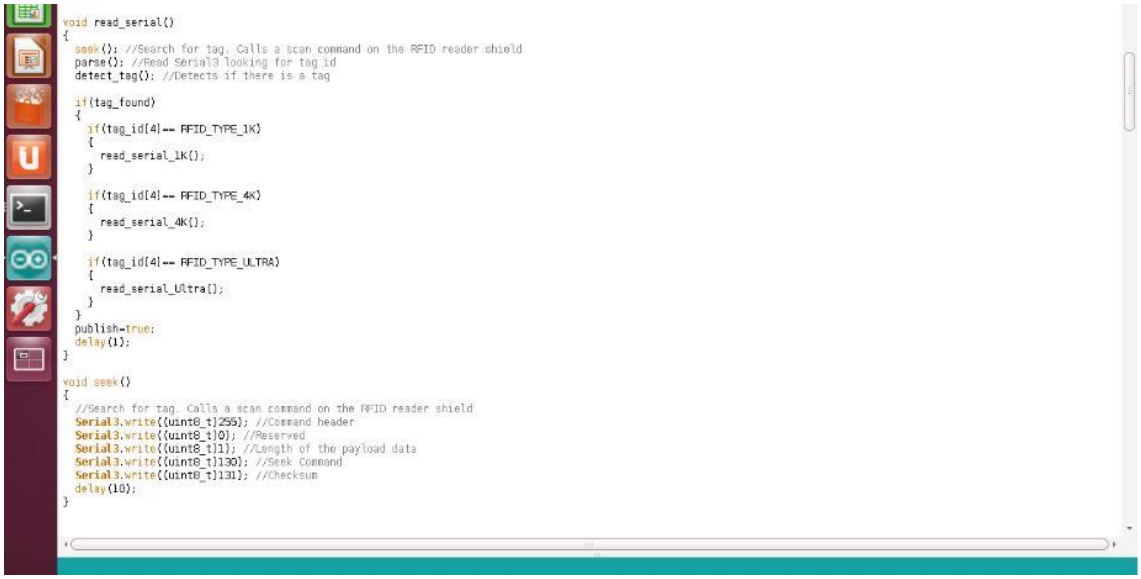


Figura 20 – Funciones read_serial y seek

```
void parse()
{
  //Indicates the RFID tag type
  while(Serial3.available())
  {
    if(Serial3.read() == 255)
    {
      for(int i=1;i<11;i++)
      {
        tag_id[i]= Serial3.read();
        pb_ID[8-i]=tag_id[i];
      }
    }
  }

  if(tag_id[4]== RFID_TYPE_ULTRA)
  {
    RFID_TAG_TYPE = 2;
    size_data = SIZE_DATA_TAG_ULTRA;
    size_id = 4;
  }

  if(tag_id[4]== RFID_TYPE_1K)
  {
    RFID_TAG_TYPE = 0;
    size_data = SIZE_DATA_TAG_1K;
    size_id = 4;
    }

  if(tag_id[4]== RFID_TYPE_4K)
  {
    RFID_TAG_TYPE = 1;
    size_data = SIZE_DATA_TAG_4K;
    size_id = 4;
  }
}
```

Figura 21 – Función parse

```
void detect_tag()
{
  if(tag_id[2] == 0x06)
  {
    tag_found = true;
  }
  else if(tag_id[2] == 0x09)
  {
    tag_found = true;
  }
  else
  {
    tag_found = false;
  }
}

//Read Serial for 1K tag
void read_serial_1K()
{
  int l=0;
  int i=1;
  for(int k=1; k<63; k++)
  {
    if(k!=(4*i)-1)
    {
      authenticate(k,l);
      l=l+16;
    }
    else
    {
      i++;
    }
  }
  new_write=false;
  publish_data();
}
```

Figura 22 – Funciones detect_tag y read_serial_1K

```
//Read Serial for 4K tag
void read_serial_4K()
{
  int l=0;
  int i=1;
  int m=1;
  for(int k=1; k<127; k++)
  {
    if(k!=(4*i)-1)
    {
      authenticate(k,l);
      l=l+16;
    }
    else
    {
      i++;
    }
  }
  for(int k=128; k<255; k++)
  {
    if(k!=(16*m)-1)
    {
      authenticate(k,l);
      l=l+16;
    }
    else
    {
      m++;
    }
  }
  new_write=false;
  publish_data();
}

//Read Serial for Ultra tag
void read_serial_Ultra()
{
  int l=0;
  for(int k=4; k<16 ;k++)
  {
    authenticate(k,l);
    l=l+16;
  }
  new_write=false;
  publish_data();
}
```

Figura 23 – Funciones read_serial_4K y read_serial_Ultra

```
void write_dat(int k, int l)
{
  int chksum=0;
  int cont=0;
  Serial3.write((uint8_t)255); //Command Header
  Serial3.write((uint8_t)0); //Reserved
  Serial3.write((uint8_t)10); //Length of the payload data
  Serial3.write((uint8_t)137); //Write Value Block Command
  Serial3.write((uint8_t)k); //Number of Block to write

  for(int i=l; i<16+l; i++)
  {
    Serial3.write((uint8_t)write_data[i]); //Send 16 bytes from earlier
    cont = cont +write_data[i]; //Calculate checksum
  }

  chksum=10+137+k+cont;
  Serial3.write((uint8_t)chksum); //Write checksum
  delay(10);
  cont=0;
  chksum=0;
}
```

Figura 24 – Función write_dat

```
void read_data(int k, int l)
{
  int chksum=0;
  Serial3.write((uint8_t)255); //Command header
  Serial3.write((uint8_t)0); //Reserved
  Serial3.write((uint8_t)2); //Length of the payload data
  Serial3.write((uint8_t)134); //Read Block Command
  Serial3.write((uint8_t)k); //Number of Block to read
  chksum = 2+134+k; //Calculate checksum
  Serial3.write((uint8_t)chksum); //Checksun

  //Get response
  delay(10);
  while(Serial3.available())
  {
    if(Serial3.read()==255)
    {
      for(int j=l; j<21+l; j++)
      {
        dat[j+l]= Serial3.read();
      }
    }
  }
  for(int j=l; j<16+l; j++)
  {
    pb[j]=dat[j+l+4];
  }
  chksum=0;
}
```

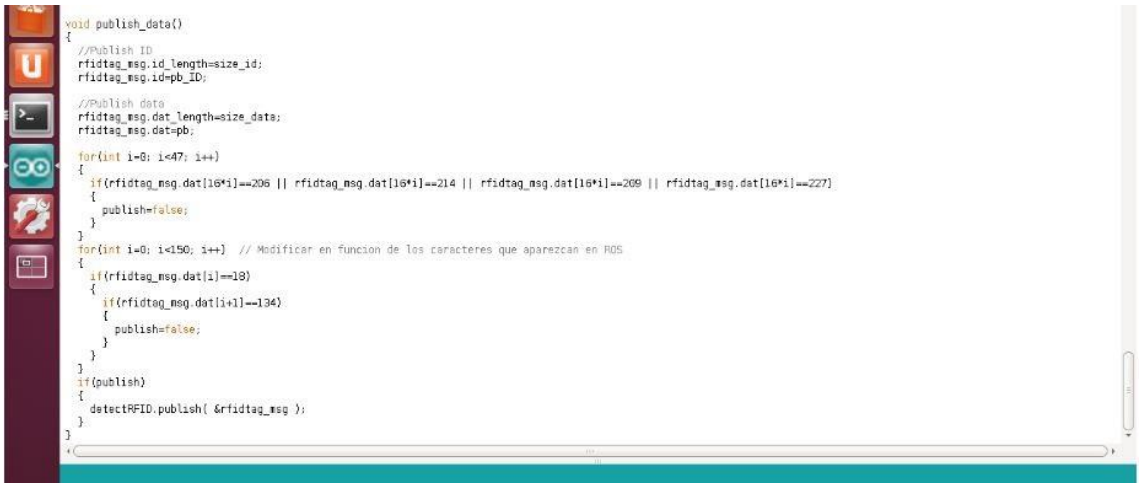
Figura 25 – Función read_data

```
void authenticate(int k, int l)
{
  int chksum=0;
  Serial3.write((uint8_t)255); //Command header
  Serial3.write((uint8_t)0); //Reserved
  Serial3.write((uint8_t)3); //Length of the payload data
  Serial3.write((uint8_t)133); //Authenticate Command
  Serial3.write((uint8_t)k); //Number of Block to authenticate
  Serial3.write((uint8_t)255); //Key Type
  chksum = 3+133+k-1; //Calculate checksun
  Serial3.write((uint8_t)chksum); //Checksun

  //Get Response
  delay(10);
  while(Serial3.available())
  {
    if(Serial3.read() == 255)
    {
      for(int i=1; i<5; i++)
      {
        auth[i]= Serial3.read();
      }
    }
  }

  if(auth[4]==76)
  {
    if(new_write)
    {
      write_dat(k,l);
    }
    read_data(k,l);
  }
  else
  {
  }
  chksum=0;
}
```

Figura 26 – Función authenticate



```
void publish_data()
{
  //Publish ID
  rfidtag_msg.id_length=size_id;
  rfidtag_msg.id=pb_ID;

  //Publish data
  rfidtag_msg.dat_length=size_data;
  rfidtag_msg.dat=pb;

  for(int i=0; i<47; i++)
  {
    if(rfidtag_msg.dat[16*i]==206 || rfidtag_msg.dat[16*i]==214 || rfidtag_msg.dat[16*i]==209 || rfidtag_msg.dat[16*i]==227)
    {
      publish=false;
    }
  }
  for(int i=0; i<150; i++)  // Modificar en funcion de los caracteres que aparezcan en ROS
  {
    if(rfidtag_msg.dat[i]==18)
    {
      if(rfidtag_msg.dat[i+1]==134)
      {
        publish=false;
      }
    }
  }
  if(publish)
  {
    detectRFID.publish( &rfidtag_msg );
  }
}
```

Figura 27 – Función publish_data