Diseño de un dispensador automático de alimento concentrado para cerdos

Dania Carolina González González

Universidad Nacional Abierta y a Distancia - UNAD

Escuela de Ciencias Básicas Tecnología e Ingeniería - ECBTI

Ingeniería Electrónica

Noviembre 2021

Diseño de un dispensador automático de alimento concentrado para cerdos

Dania Carolina González González

Asesor: Mg Pedro Torres Silva

Universidad Nacional Abierta y a Distancia - UNAD

Escuela de Ciencias Básicas Tecnología e Ingeniería - ECBTI

Ingeniería Electrónica

Noviembre 2021

Dedicatoria

A mis padres por motivarme y apoyarme cada día, pero

sobre todo, por siempre creer en mí.

Agradecimientos

Al Ingeniero Pedro Torres por sus enseñanzas y orientaciones

para realizar este proyecto aplicado.

Resumen

Aunque durante los últimos años el sector Porcícola colombiano ha venido creciendo y muestra gran dinamismo, factores como el aumento de las importaciones de carne de cerdo a consecuencia del Tratado de Libre Comercio y los elevados costos de producción particularmente en la alimentación, generan incertidumbre y preocupación, especialmente para los pequeños y medianos porcicultores.

Actualmente en el mercado es posible encontrar una amplia gama de dispensadores automáticos para la alimentación de especies animales, tanto domesticas como productivas, como es el caso de la ganadería, la avicultura, la piscicultura o la porcicultura a gran escala. Sin embargo, son pocos los pequeños y medianos productores que tienen acceso a este tipo de tecnología, limitando su competitividad en el mercado.

Mediante el presente proyecto aplicado se pretende diseñar e implementar un dispensador de alimento concentrado para cerdos, empleando materiales asequibles y económicos, con el fin de automatizar el proceso de administración de concentrado, mejorando su aprovechamiento al proveer raciones alimenticias proporcionales a los requerimientos de la tabla nutricional en horarios exactos. Como resultado, esperamos contribuir en la superación de las dificultades de este sector productivo, brindando herramientas de tecnificación que aporten significativamente para hacer de la porcicultura colombiana una actividad más productiva y competitiva, especialmente para los porcicultores de nuestra región.

Palabras clave: Dispensador, porcicultura, automatización, competitividad

Abstract

Although during the last years the Colombian pork sector has been growing and shows great dynamism, factors such as the increase of pork imports as a consequence of the Free Trade Agreement and the high production costs, particularly in feed, generate uncertainty and concern, especially for small and medium pig farmers.

Currently in the market it is possible to find a wide range of automatic dispensers for feeding animal species, both domestic and productive, as is the case of livestock, poultry, fish or pig farming on a large scale. However, few small and medium producers have access to this type of technology, limiting their competitiveness in the market.

Through this applied project, we intend to design and implement a concentrated feed dispenser for pigs, using affordable and economical materials, in order to automate the process of concentrate administration, improving its utilization by providing feed rations proportional to the requirements of the nutritional table at exact times. As a result, we hope to contribute in overcoming the difficulties of this productive sector, providing technification tools that contribute significantly to make Colombian swine farming a more productive and competitive activity, especially for the pig farmers of our region.

Key words: Dispenser, pig farming, automation, competitiveness.

Introducción

Durante los últimos años, el sector porcícola colombiano ha mostrado un gran dinamismo, aumentando su producción y manteniendo su crecimiento. Esta industria se muestra cada vez más sólida teniendo una participación muy importante en el PIB agropecuario, como lo muestran las cifras reportadas por el DANE de 2016.

Pese a lo anteriormente mencionado, el reto es continuar fortaleciendo la porcicultura, aumentando la producción y comercialización de carne de cerdo, mediante el empleo de herramientas de tecnificación, investigación y transferencia de tecnología, bienestar animal y bioseguridad. En el sector Porcícola tenemos un gran potencial exportador, por eso uno de los objetivos del gobierno nacional es lograr la apertura de nuevos mercados internacionales. Sin embargo, gran parte de la producción nacional porcícola continúa siendo generada mediante sistemas tradicionales. De ahí la importancia de buscar alternativas que contribuyan en el logro de los objetivos y el fortalecimiento y consolidación de este importante sector productivo.

El presente proyecto denominado, "diseño de un dispensador automático de alimento concentrado para cerdos" permitirá la dosificación exacta del alimento concentrado, evitando el desperdicio y aumentando la rentabilidad de las instalaciones. Dentro de sus ventajas podemos mencionar, que está fabricado con materiales económicos, pero de óptima calidad, para que sea económicamente asequible especialmente para los pequeños y medianos productores. De igual manera éste aparato podrá ser adaptado para la alimentación de otras especies animales tanto domesticas como productivas, siempre que se trate de alimentos concentrados, cuya aprobación se rige por la norma ISO 14001 (Organización Internacional de Estandarización), referente al Sistemas de Gestión Ambiental (SGA) la cual busca que las empresas demuestren que son responsables y comprometidas con la protección del medio ambiente. Así mismo, este

dispensador se constituye en una herramienta de trabajo y de tecnificación, disminuyendo la brecha de los productores de pequeña y gran escala y posibilitando su competencia en el mercado.

Con el diseño y ejecución del presente proyecto es posible mostrar la importancia de la aplicación de la electrónica tanto para facilitar los procesos mediante herramientas de tecnificación como para superar dificultades en el sector productivo, en este caso específicamente en el sector de la porcicultura, generando un impacto positivo en los productores de nuestra región.

Índice

Lista de tablas

Lista de figuras

Lista de anexos

Definición del problema

Antecedentes del problema

Los aparatos dispensadores de alimento para animales han despertado el interés de varios investigadores durante los últimos años.

En el año 2007, Luis Ernesto cadena Flórez, estudiante del programa de ingeniería industrial de la Universidad Autónoma de Occidente, desarrolló como trabajo de grado un dosificador automático para alimentar cerdos. Este sistema automatizado seria programado desde una unidad que controla los dosificadores, el ganadero programaría cuántas veces y en qué cantidad alimentar los animales. Como metodología para desarrollar el dosificador se utilizó el Despliegue de la Función de Calidad (QFD, Quality Function Deployment).

Se concluyó que empleando el dosificador se logra reducir el tiempo invertido hasta en un 90% y se reducen costos de operación.

En la Universidad Surcolombiana, Erika Álvarez desarrolló un prototipo como proyecto de grado, el cual consistía en automatizar un dosificador de alimento con su respectivo método de dispersión, además del monitoreo en tiempo real de la temperatura del estanque.

Para el monitoreo de la temperatura presente en el agua, se realizó la adquisición de los datos por medio de un sensor de temperatura, seguidamente la digitalización de esta información se hizo mediante un microcontrolador, para poder realizar la respectiva visualización a través de una pantalla LCD.

Para el 2007, Morales y Villalba, de la universidad San Buenaventura de Bogotá, para el desarrollo de su proyecto de grado, desarrollan un dispensador de alimento para perros, con el fin de hacer más fácil y eficaz el proceso de suministro alimenticio a determinadas horas y en horarios

específicos dependiendo del peso y tamaño del animal. La capacidad máxima del sistema es de 5 kg y los perros no pueden superar los 30 kg, y los intervalos de dispensado son de 3, 6 o 9 horas. También se pudo observar la implementación de los sistemas acoplados, “electrónico-mecánico”.

En el año 2013, Zapata y Gil, desarrollaron un prototipo de dispensador automático de comida para mascotas, donde el principal componente es una tarjeta Raspberry Pi, y a través de una plataforma Android establecer los horarios de descargue del alimento.

El servidor web instalado y configurado en la Raspberry Pi, se diseña para atender y responder a la petición del navegador que se soliciten usando el protocolo HTTP. Este servidor es uno de los elementos centrales de la red de trabajo, el cual permite la conexión de equipos remotos tales como el smartphone del usuario permitiendo el control, monitoreo y conexión con la aplicación móvil. El servidor web se enlaza directamente con la aplicación móvil, la cual dará las pautas de interacción del usuario con el comedero. La orden de ejecución para la activación del GPIO es dada desde el servidor web al código elaborado en PHP, quien a su vez hace un llamado al código PYTHON el cual coloca en HIGH el pin correspondiente. Basado en Raspberry pi controlado mediante una aplicación móvil, cumpliendo así los objetivos del proyecto, automatizando una tarea diaria como lo es alimentar nuestras mascotas.

Sergio Andrés Peñuela Argüello y Anderson Alvear Betancourt, estudiantes de la Universidad Surcolombiana realizaron el proyecto “Automatización de un sistema dosificador de alimento para alevinos teniendo en cuenta el porcentaje de oxígeno y la temperatura del agua en un ambiente controlado” en el cual llevaron a cabo el desarrollo de un sistema integral para la optimización de un dosificador de alimentos para alevinos donde realizaban la medición remota de variables físicas como el porcentaje de oxígeno disuelto y temperatura del agua; la

instrumentación electrónica para el monitoreo de dichas variables y la automatización, para el control del dosificador de alimento

Formulación del problema

El sector Porcícola colombiano está conformado principalmente por productores tradicionales de pequeña escala y con difícil acceso al crédito y la tecnología. Acontecimientos como la aprobación del Libre Comercio (TLC) de Colombia con otros países, han generado el aumento progresivo de importaciones de carne de cerdo mientras los elevados costos de producción en mano de obra e insumos, generan un panorama desalentador, fundamentalmente para los pequeños y medianos porcicultores, quienes difícilmente podrán competir con los grandes productores nacionales y mucho menos con productores de otros países.

La alimentación de los animales es quizás una de las actividades más importantes en este tipo de sistemas productivos, es por ello que se debe disponer del tiempo necesario para suplir sus necesidades nutricionales de manera apropiada, evitando la sobrealimentación y el desperdicio de alimento.

Entonces, ¿Cómo podríamos contribuir en el mejoramiento de la productividad y competitividad de los pequeños y medianos productores porcícolas de nuestra región?

Descripción del problema

A nivel mundial la porcicultura ha crecido considerablemente. Sin embargo, en los países en desarrollo, la mitad de la producción porcina actual sigue manteniéndose bajo sistemas tradicionales de producción a pequeña escala, fundamentalmente de subsistencia (Departamento de Agricultura y Protección del Consumidor, 2014).

En América del Sur la porcicultura se ha visto afectada por problemas inherentes a las economías de los países en desarrollo, como lo son el acceso a crédito, los altos impuestos, la inestabilidad económica y las altas tasas de interés entre otros.

En Colombia el consumo de carne de cerdo ha aumentado considerablemente durante los últimos años, según El Fondo Nacional de la Porcicultura (Porkcolombia), en 2018 se registró un consumo de 10,3 kilogramos por habitante y para poder responder a esta demanda se han tenido que realizar crecientes importaciones.

Una de las mayores problemáticas del sector porcícola está relacionado con el aumento de las importaciones de carne, debido al Tratado de Libre Comercio (TLC), firmado por Colombia con otros países, permitiendo que a nuestro país ingresen importaciones de manera progresiva, como se puede analizar al comparar el volumen de importaciones de productos y subproductos que se presentó de enero a agosto de 2019, el cual fue de 77.102 toneladas con respecto al año 2018, cuando este volumen alcanzó las 66.242 toneladas, provenientes principalmente de Estados Unidos, Chile y Canadá y que representa un aumento del 16.4% (Sistema de información de Gestión y Desempeño de Organizaciones de Cadenas, 2019). De igual manera los elevados costos de producción especialmente de insumos de alimentación, generan un panorama desalentador, fundamentalmente para los pequeños y medianos porcicultores. Esto sumado a la falta de políticas proteccionistas, la falta de redes colaborativas de producción, infraestructura, capacitación y transferencia tecnológica generan una brecha entre productores y dificultan su competencia en el mercado Nacional, así como con otros países.

En la mayoría de los sistemas productivos, el costo de alimentación puede variar entre 65% -75% del costo total para producir un cerdo de crecimiento-acabado a peso de mercado. Las estimaciones sugieren que el 2% al 20% de la alimentación se desperdicia. Los ensayos muestran

que un rango de 2% a 5,8% de la alimentación se desperdicia en el comedero (NRC, 1994; Mateos et al, 1995; Mateos et al, 1996).

De otro lado, a nivel regional el año 2017 el Departamento del Huila representaba el 2.75% de la producción porcina a nivel nacional (Huila, 2017). Una cifra baja si se considera que existe una gran demanda del producto y potencial para el desarrollo de proyectos porcícolas en los diferentes Municipios, incluido el Municipio de Palermo.

Por todo lo anterior se considera de fundamental importancia el desarrollo de proyectos que permitan la transferencia de tecnología, en nuestro caso el diseño de un dispensador automático de alimento concentrado para cerdos, con el cual se espera contribuir significativamente en la superación de la problemática expuesta.

Justificación

El rápido crecimiento de la población mundial genera un aumento en la demanda de alimentos, dentro de los cuales la carne de cerdo es la de mayor consumo a nivel mundial. Junto con el de las aves de corral, el porcino es el subsector pecuario de mayor crecimiento. La producción porcina está distribuida por todo el mundo, con excepción de algunas regiones con condiciones culturales y religiosas especiales. Además de contribuir a la seguridad alimentaria, el cerdo también puede representar una red de seguridad financiera, desempeñar una función en las tradiciones culturales o generar ingresos adicionales.

Según el Fondo Nacional de la Porcicultura, el sector Porcícola colombiano ha mostrado un crecimiento anual promedio superior a 7.2% durante los últimos diez años, al punto que a octubre de 2019 la producción de carne de cerdo alcanzó las 367.408 toneladas, con una participación del 4.8% del PIB pecuario en el año 2018 (Sistema de información de Gestión y Desempeño de Organizaciones de Cadenas, 2019). Reportes del Instituto Colombiano Agropecuario (ICA), indican que, en 2018, se contabilizaron 239.199 granjas porcícolas en todo el país con una población porcina de más de 5,5 millones de animales. Se destacan Antioquia (45%), Valle del Cauca (15%), Eje Cafetero (8,6%), Meta (5,6%) y Atlántico (2,9%), como las regiones con mayor producción de cerdos (González, 2019).

Lo anteriormente mencionado indica que es necesario generar estrategias que permitan fortalecer la porcicultura colombina y transformarla en una actividad prolífica, rentable y competitiva. Para ello es de fundamental importancia desarrollar proyectos que implementen la investigación y la transferencia de tecnología.

Con la presente propuesta de trabajo de grado, se propone el diseño de un dispensador automático de alimento concentrado para cerdos, que sin lugar a dudas se constituye en una

alternativa en la disminución de costos de producción tanto en mano de obra como de alimentación. Esto debido a que después de programado, será un dispositivo electrónico el encargado de proporcionar el concentrado alimenticio a los animales en el horario indicado, de manera exacta, supliendo sus requerimientos nutricionales, mejorado su aprovechamiento y evitando el desperdicio. De esta manera se pretende lograr una producción más eficiente con capacidad de competir en el mercado y por ende con mejores ingresos en especial para nuestros pequeños y medianos productores.

Objetivos

Objetivo general:

Diseñar e implementar un dispensador automático de alimento concentrado para cerdos, en una unidad productiva del Municipio de Palermo-Huila.

Objetivos específicos:

Indagación sobre equipos similares en el mundo.

Apropiación del proceso de alimentación, identificación de variables, construcción del modelo.

Diseño del prototipo

Prueba del prototipo y ajustes finales

Marco referencial

Marco teórico

Porcicultura

Generalidades: El sistema de producción de cerdos se considera muy eficiente, entre otras cosas debido a que estos animales son muy precoces y prolíficos, además su ciclo reproductivo es corto y transforma fácilmente los nutrientes. Así mismo la carne de cerdo es valorada por su alto contenido de nutrientes para la alimentación humana. Se estima que una porción de cien (100) gramos de carne magra de cerdo, proporciona a un adulto el 52% de las proteínas, el 35% de hierro, el 28% de fósforo, el 26% de zinc, el 74% de la tiamina, el 40% de la vitamina B12, el 25% del niacina, el 22% de la vitamina B6, el 19% de la riboflavina y el 9% de las calorías requeridas diariamente. Los beneficios mencionados han contribuido a estimular la producción y el consumo de carne de cerdo tanto a nivel mundial como nacional, regional y local.

Suministro de alimento en sistemas porcinos. La alimentación de los animales es uno de los factores más importantes debido a que representa aproximadamente el 80% de los costos de producción, además el tiempo de duración para alcanzar el peso ideal para la venta depende de la precisión en la cantidad y la calidad del alimento aportado a estos en cada ración diaria. Generalmente los pequeños y medianos porcicultores almacenan el alimento concentrado por bultos y alimentan a los animales de forma manual. Los bultos son transportados hasta la instalación de los animales y el alimento se mide con algún recipiente para finalmente ser suministrado. Existen alimentos específicos y balanceados para cada etapa de crecimiento, bien

sea iniciación, levante, ceba, gestación o lactancia, los cuales serán suministrados en las cantidades recomendadas e incrementarse gradualmente con el paso del tiempo.

Tabla 1

Etapas de desarrollo de un cerdo.

ETAPA	TIEMPO	PESO
Lactancia	2 meses	20Kg
Crecimiento	2 meses	60Kg
Engorda	2 meses	90-100Kg

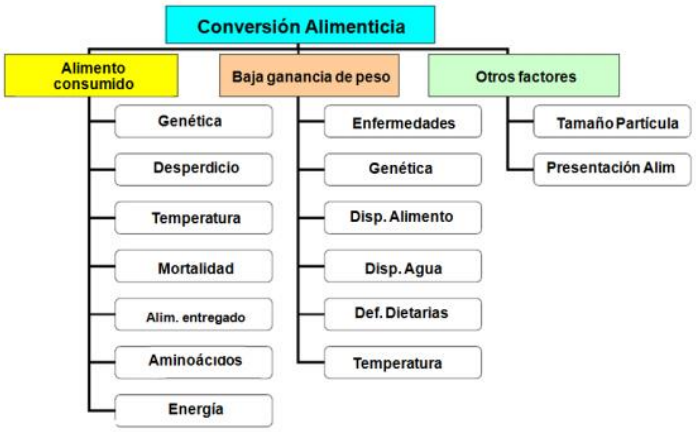
Tabla 2

Alimentación por etapas de desarrollo.

Peso cerdo (Kg)	**20**	**25**	**30**	**35**	**40**	**50**	**55**	**60**	**65**	**70**	**75**	**80**	**85**	**90**
Ración diaria (Kg)	**1**	**1,2**	**1,4**	**1,6**	**1,8**	**2,1**	**2,2**	**2,3**	**2,4**	**2,5**	**2,6**	**2,7**	**2,8**	**2,9**
Ración por comida (g)	**500**	**600**	**700**	**800**	**900**	**1050**	**1100**	**1150**	**1200**	**1250**	**1300**	**1350**	**1400**	**1450**
Total, alimento semana (Kg)	**7**	**8.4**	**9.8**	**11.2**	**12.6**	**14.7**	**15.4**	**16.1**	**16.8**	**17.5**	**18.2**	**18.9**	**19.6**	**20.3**

Figura 1

Principales factores que afectan la conversión alimenticia.



Nota. (Factores que afectan la conversión alimenticia en cerdos Fernando J. Bártoli)

Automatización en la alimentación porcina. Frecuentemente los programas nutricionales están centrados en la formulación y en la elaboración de la dieta, sin embargo, la forma en que se proporciona el alimento (tipos de comedero) a los cerdos puede influir notablemente en mejorar la eficiencia alimenticia y en el coste de la alimentación (Goodband et al, 2009).

Un sistema adecuado de alimentación para cerdos implica tanto el tipo de alimentación en función de la forma física (solida, liquida, mixta, húmeda), así como la forma o sistema de suministro (automatizada, semiautomatizada, manual) (Moreno et al., 1996; DeRouchey y Richert, 2010).

La mecanización de la alimentación, ha permitido que el tiempo de suministro de alimento se reduzca, y en la mayoría de los casos la cantidad de alimento es homogénea. Sin embargo, el sistema tiene un coste mayor, debido a la instalación de tornillos sin fin, tolvas, y tubos de caída de alimento en su caso. Para la utilización de este sistema es necesario considerar

la forma física del alimento. Además, depende de factores como la etapa productiva y fisiológica de los animales, tipo de suministro (restringido o ad libitum) (Moreno et al., 1996; DeRouchey y Richert, 2010).

El sistema de suministro de alimento automatizado presenta algunas ventajas frente a los métodos de distribución manual (Moreno et al., 1996):

• Reducción del tiempo de dedicación laboral a las tareas de la granja

• Disminución del estrés tanto de los animales, como de los trabajadores, al reducir el tiempo de estos en la nave.

• Reducción de fallo reproductivo en las hembras por estrés

• Mejora el control y ajuste de las raciones, beneficiando el índice de conversión

• Facilita el empleo de subproductos sustitutivos de cereales.

• Menor incidencia de lesiones físicas en los trabajadores.

Sin embargo, también presenta inconvenientes:

• Aumento de la inversión y coste de mantenimiento de los equipos e instalaciones

• Previsión de cortes de flujo, mediante sistemas de accionamiento manual

. • Problemas para adaptar los equipos (sistemas automáticos) en las construcciones existentes.

• Dificultan o imposibilita la distribución de alimento (diferente tipo) en animales de etapas distintas ubicados en la misma nave.

• Adaptación de los animales a algunos tipos de sistemas automáticos

• Posible producción excesiva de polvo en sistemas que vierten el alimento al suelo, además de que hay menor control en el desperdicio de alimento.

• Excesiva confianza de los trabajadores en cuanto a la eficiencia del sistema, lo que induce a un bajo nivel de supervisión del estado físico de los animales debido a la cantidad de alimento

suministrado. El alimento se reparte típicamente a través del conducto de alimentación en forma de tornillo sin fin (en espiral) a corrales individuales o a compartimientos de almacenamiento (DeRouchey y Richert, 2010).

Comederos. Existen diferentes diseños de comederos, los más comunes son para sistemas de alimentación tradicional o alimentación seca de flujo continuo por gravedad, comederos para alimentación combinada húmeda y seca; y comederos de tolva con medición de suministro. Los comederos deben ofrecer el espacio suficiente para que los animales consuman con facilidad y comodidad el alimento suministrado y este no sea indisponible o desperdiciado (DeRouchey y Richert, 2010). Los comederos para cerdos hasta 25 Kg de peso vivo, deben contar con espacios para el alimento de 17.8-20.3cm de ancho, la bandeja de alimentación de 12.7-15.24 cm de profundidad con un borde frontal de 7.62-10.1cm. Para cerdos en etapa de crecimiento y finalización, el espacio del comedero debe ser de 30.48-33.02 cm de ancho, con una profundidad de 25.4-30.48cm y un borde de 10.16-15.24 cm (cerdos hasta 120 Kg) (DeRouchey y Richert, 2010).

Desperdicio de alimento. Una alimentación oportuna puede reducir los costos de alimentación, pues este se constituye en el mayor gasto de una operación de crecimiento-finalización en sistema productivo de cerdos. El costo del alimento puede variar entre 65% -75% del costo total para producir un cerdo de crecimiento-acabado a peso de mercado. Las estimaciones sugieren que el 2% al 20% de la alimentación en estos sistemas se desperdicia.

Tabla 3

Pérdidas por desperdicio de alimento

Porcentaje de perdida/ Desperdicio de alimento	Alimento para ganancia hasta 93 Kg	Costo / Cerdo	Costo / Cerdo	Conversión Alimenticia
		Alimento consumido	Perdidas / Desperdicio	
0%	260	$82.34	$0.00	2.80
2%	266	$83.99	$1.65	2.86
4%	271	$85.63	$3.29	2.91
6%	276	$87.28	$4.94	2.97
8%	281	$88.93	$6.59	3.02
10%	286	$90.57	$8.23	3.02
15%	299	$94.69	$12.35	3.22
20%	312	$98.81	$16.47	3.36

Nota. *Gestión del alimento en el comedero de cerdos en crecimiento – finalización*. (12 de 10 de 2021) de razasporcinas.com: https://razasporcinas.com/gestion-del-alimento-en-el-comedero-de-cerdos-en-crecimiento-finalizacion/

Dispensadores automáticos de alimento

En la actualidad es posible encontrara en el mercado una amplia gama de aparatos que agilizan el proceso de alimentación de los animales especialmente para el sector productivo, ofreciendo ventajas especialmente en relación a la exactitud en la racionalización del alimento, control de problemas de salud en los animales generados durante la manipulación del alimento, reducción de mano de obra entre otros.

Historia de las máquinas dispensadoras

El rápido crecimiento de la población trae consigo grandes necesidades que pueden ser satisfechas mediante el uso de la tecnología y la innovación, logrando que las tareas sean

realizadas de manera ágil, eficiente y oportuna. Sin duda alguna la creación de los aparatos dispensadores ha revolucionado la industria al punto que en la actualidad encontramos dispensadores para múltiples productos. Estos han sido adaptados para cumplir un propósito en particular de acuerdo a la necesidad que se tenga y también al sector económico a implementar.

Se creé que la primera máquina expendedora fue diseñada por Herón de Alejandría En Egipto, se empleaba para dispensar agua bendita en los templos. Era un mecanismo sencillo que funcionaba con una moneda de dos dracmas. Sin embargo, con la Revolución Industrial en Inglaterra al principio de los años ochenta aparecieron las primeras máquinas para la venta de tarjetas postales. En 1888, la compañía Thomas Adams Gum de Estados Unidos, instalo máquinas dispensadoras de chicle en el metro de Nueva York. Años más tarde un restaurante de Filadelfia empleo máquinas expendedoras para su funcionamiento y surgieron máquinas que ofrecían variedad de artículos.

Fue hasta el año 1920 donde aparecieron las primeras máquinas automáticas para vender bebidas gaseosas servidas en vasos desechables, para el año 1926, se creó la primera máquina de venta de cigarrillos, mientras 1946, surgieron las máquinas dispensadoras de café caliente y se extienden por todo el mundo (Blog Tecnowebstudio, 2016).

Actualmente uno de los tipos de dispensadores más empleados a nivel mundial es el dispensador de alimento para animales, especialmente para mascotas como perros y gatos. Sin embargo, también han sido desarrollados prototipos para alimentación de animales destinados a la producción como es el caso de las aves, peces, ganado vacuno o porcino. No obstante, el acceso a este tipo de tecnología es limitado para pequeños y medianos productores.

Marco conceptual

Conceptos básicos

Dispensador: La definición de dispensador se refiere a el que concede, confiere, adjudica, da, depara, otorga o cede. Etimológicamente proviene del verbo activo transitivo "dispensar" y del sufijo "dor" que indica el que suele realizar la acción, también viene del latín "dispensātor" (Definiciona, 2016).

Automático: es aquello perteneciente o relativo al autómata. Este término proviene del griego automatos que significa "con movimiento propio" o "espontáneo"**.** Por lo tanto, la noción de automático puede hacer referencia a distintas cuestiones. Un mecanismo automático funciona por sí solo, ya sea en su totalidad o en parte (Definiciona, 2009).

Dispensador automático de alimento

Los dispensadores de alimento seco, son comederos tradicionales, a los que se les ha unido un depósito de comida que funciona de manera automática según sea programado (Mascotas, 2017).

Concentrado

Alimento combinado con otro para mejorar el balance nutritivo del producto y que será posteriormente diluido y mezclado para producir un suplemento o un alimento completo (AAFCO, 2000).

Transferencia de tecnología

Según la RAE, se entiende por transferir el pasar o llevar algo de un lugar a otro, así como ceder a otro los derechos, atribuciones o dominio que se tiene sobre un bien u objeto. Por su parte, la transferencia de tecnología corresponde a un proceso que ayuda a que los conocimientos requeridos para el desarrollo y producción de un producto, la aplicación de un proceso o la

prestación de un servicio, sean transmitidos de un transmisor a un receptor, que requiere tal conocimiento para incorporarlo en el desarrollo de sus productos/servicios, o a la instrumentación de sus procesos. La transferencia implica un flujo de conocimiento del transmisor al receptor, a cambio de un flujo monetario que es pagado por el receptor como contraprestación por la entrega de dicho conocimiento. Por este motivo, la transferencia debe incluir en sus etapas iniciales un proceso de negociación y acuerdos en cuanto al valor del conocimiento, forma de pago, garantías de protección a la propiedad intelectual y confidencialidad, entre otras (rutanmedellin, 2013).

Mercado

Según el Diccionario de la Real Academia Española, el mercado es el "conjunto de consumidores capaces de comprar un producto o servicio". Para Patricio Bonta y Mario Farber, autores del libro "199 Preguntas Sobre Marketing y Publicidad", el mercado es "donde confluyen la oferta y la demanda. En un sentido menos amplio, el mercado es el conjunto de todos los compradores reales y potenciales de un producto. Allan L. Reid, autor del libro "Las Técnicas Modernas de Venta y sus Aplicaciones", define el mercado como "un grupo de gente que puede comprar un producto o servicio si lo desea. Para Philip Kotler, Gary Armstrong, Dionisio Cámara e Ignacio Cruz, autores del libro "Marketing", un mercado es el "conjunto de compradores reales y potenciales de un producto. Desde la perspectiva del economista Gregory Mankiw, autor del libro "Principios de Economía", un mercado es "un grupo de compradores y vendedores de un determinado bien o servicio (Thompson, 2005).

Internet de las cosas (IoT)

Tomando la definición dada por el Grupo de Soluciones Empresariales basadas en Internet (IBSG, Internet Business Solutions Group) de Cisco, IoT (Internet of things), Internet de las cosas es sencillamente el punto en el tiempo en el que se conectaron a internet más “cosas u

objetos" que personas. Hasta el 2003, había 0,08 dispositivos por persona conectados a internet, pero a partir del 2008 se notó un aumento exponencial, lo cual ha hecho que Cisco IBSG, con miras al futuro, se pronunciará respecto al tema, destacando que para el 2020 habrá aproximadamente 50 mil millones de dispositivos conectados a internet.

El ser humano ha logrado avanzar, gracias a su facilidad de comunicación, a tal punto que desarrolla métodos para facilitar aún más esa comunicación, lo cual optimiza la adquisición de sabiduría. Con la invención del internet, se ha podido sintetizar la obtención de información; es aquí donde juega un papel muy importante los avances en el Internet de las cosas, ya que gracias a él podemos tener pleno conocimiento y control sobre lo que sucede a nuestro alrededor, tanto así que es posible hacer uso de sensores de pequeños tamaños en plantas, animales y fenómenos geológicos, para poder adquirir datos y predecir catástrofes ambientales, enfermedades, controlar procesos o simplemente determinar el estado meteorológico de lugares específicos.

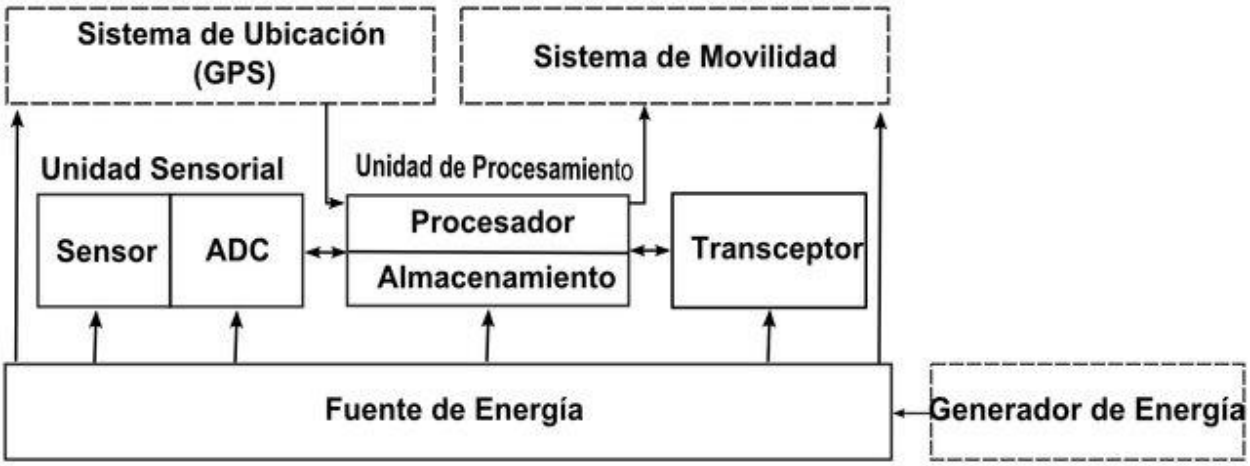
En palabras más técnicas, IoT consiste en la integración de sensores y dispositivos en objetos cotidianos, para así adquirir información que pueda ser enviada a internet, a través de redes fijas e inalámbricas. El IoT implica que todo objeto puede ser fuente de datos, ya sea que estén integrados en hogares, entornos de trabajo o lugares públicos.

Nodo sensor

Un nodo sensor es un dispositivo que es capaz de realizar algún procesamiento, recopilar información sensorial y comunicarse con otros nodos conectados a la red. Los componentes principales de un nodo sensor son un microcontrolador, transceptor, memoria externa, fuente de alimentación y uno o más sensores.

Figura 2

Componentes de un sensor.



Nota: Carletti, E. J. (12 de 10 de 2021). *Sensores - Conceptos generales*. Obtenido de http://robots-argentina.com.ar/: http://robots-argentina.com.ar/Sensores_general.htm

Microcontrolador

Es un circuito integrado que es el componente principal de una aplicación embebida. Es como una pequeña computadora que incluye sistemas para controlar elementos de entrada/salida. También incluye a un procesador y por supuesto memoria que puede guardar el programa y sus variables (flash y RAM). Funciona como una mini PC. Su función es la de automatizar procesos y procesar información.

Elementos de un microcontrolador:

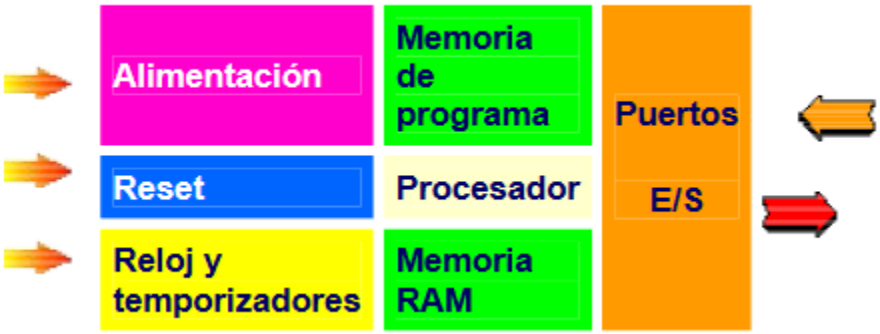
-Microprocesador.

-Periféricos (unidades de entrada/salida).

-Memoria.

Figura 3.

Estructura de un microcontrolador.



Nota. *Microcontroladores*. (14 de 09 de 2021). Obtenido de bibing.us.es: http://bibing.us.es/proyectos/abreproy/11141/fichero/PFC%252F3+Microcontroladores.pdf

Diseño metodológico

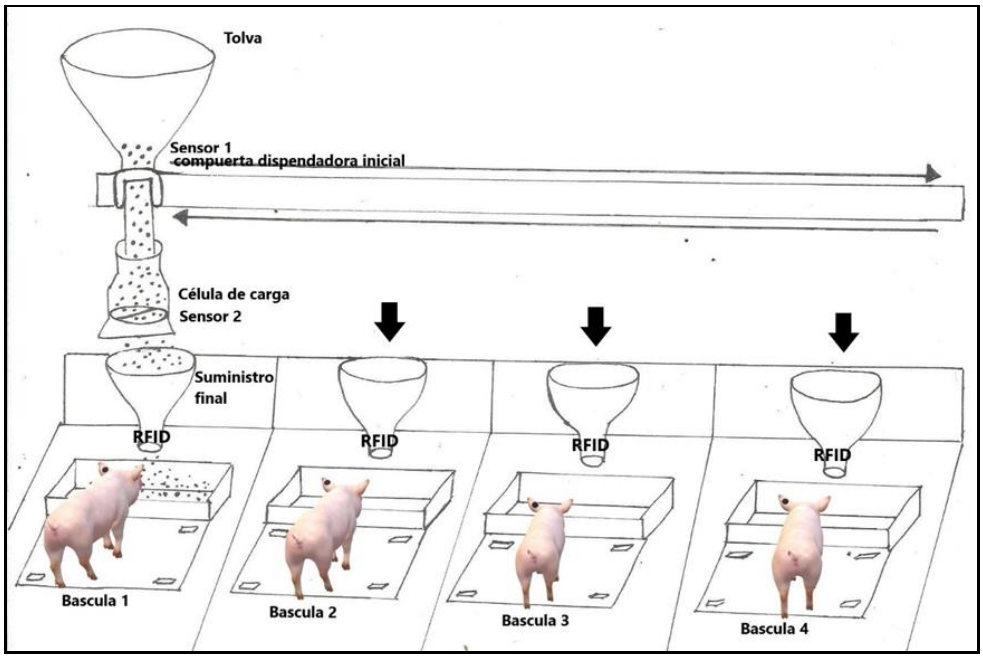
Fase 1. Revisión bibliográfica

Una vez identificada y definida la problemática, principalmente en el ámbito nacional, regional y local, se procedió a realizar una revisión bibliográfica que permitió la referenciación de antecedentes y estudios de caso de gran utilidad para el análisis de las diferentes alternativas propuestas para el desarrollo del presente proyecto aplicado.

Diseño y especificaciones del dispensador de alimento

Figura 4

Prototipo del dispensador



Componentes del sistema. El dispensador automático de alimento para cerdos consta de los siguientes elementos:

- Tanque almacenamiento

- Tubería en PVC
- Servo motor
- Microcontrolador
- Bascula
- Sistema de identificación - RFID

Los materiales empleados son económicos, pero de buena calidad y de fácil adquisición en el mercado.

Fase 2. Reconocimiento del proceso de alimentación, definición de otras variables y diseño del prototipo: Se realizó un reconocimiento del proceso de alimentación convencional de los animales. Se encontró que generalmente los pequeños y medianos porcicultores almacenan el alimento de los animales en sacos de 40kg que son transportados hasta las instalaciones en donde se miden las raciones a suministrar en los horarios establecidos, empleando algún recipiente. Este proceso demanda de cierto tiempo dependiendo del tamaño del sistema productivo y el número de animales.

Posteriormente se establecieron los criterios iniciales de diseño del dispensador automático para alimento, las especificaciones, los requerimientos y las características del aparato a diseñar, considerando que aporte significativamente en la solución de la problemática expuesta.

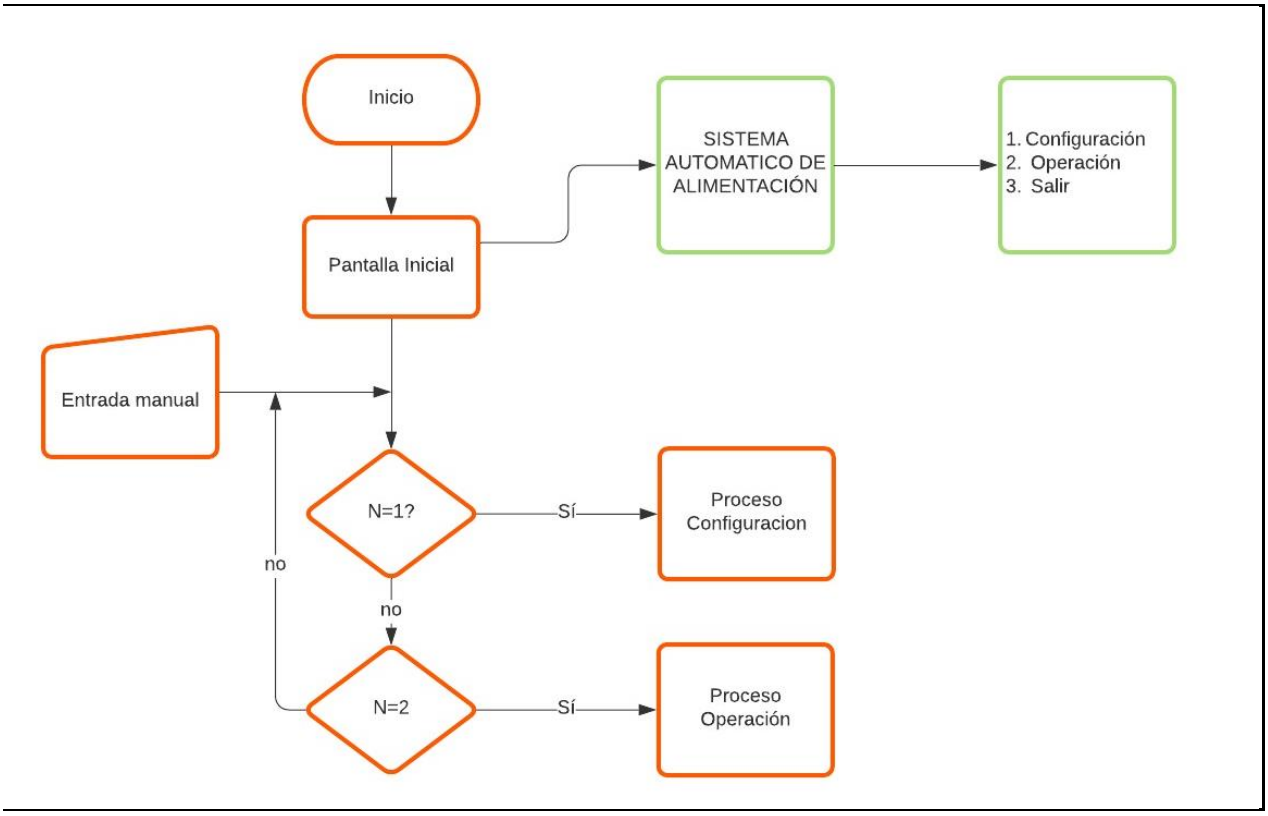
Se consideraron varias alternativas. Pero se opta por la que de diseña en este trabajo.

Diagrama de flujo prototipo

Programa principal

Figura 5

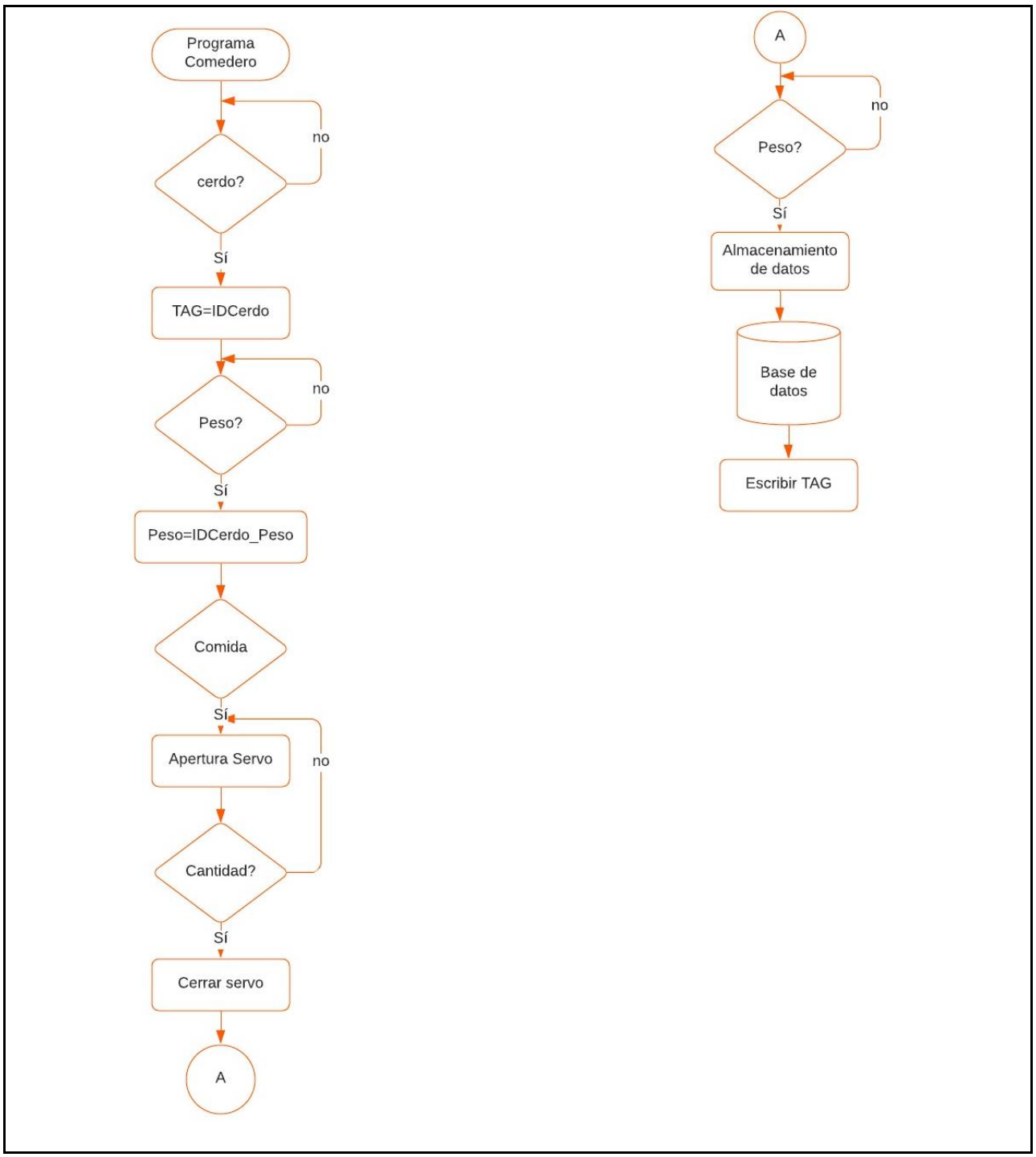
Diagrama de flujo programa principal.



Automáta

Figura 6

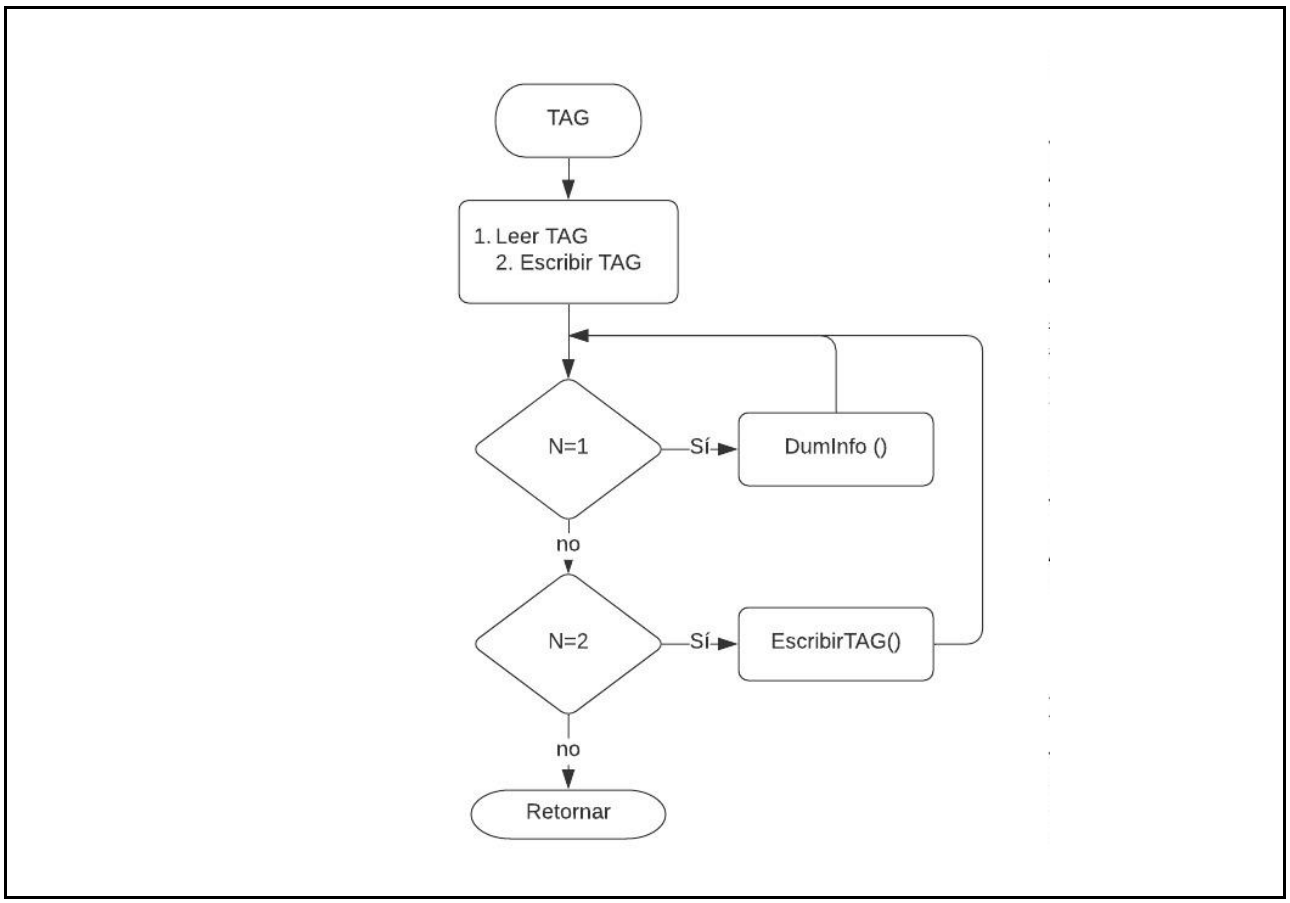
Diagrama de flujo autómata.



TAG

Figura 7

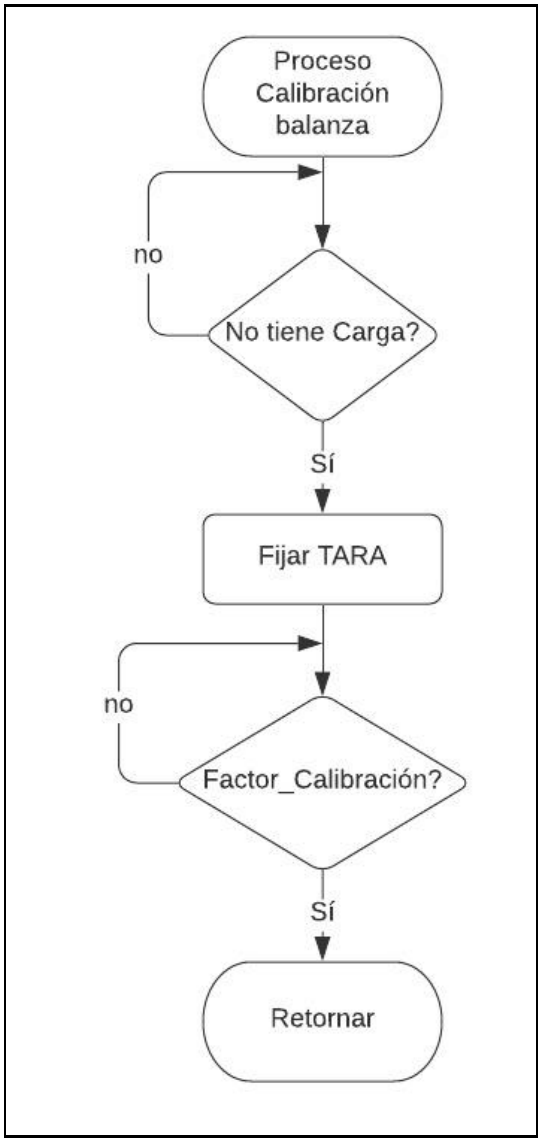
Diagrama de flujo TAG.



Balanza

Figura 8

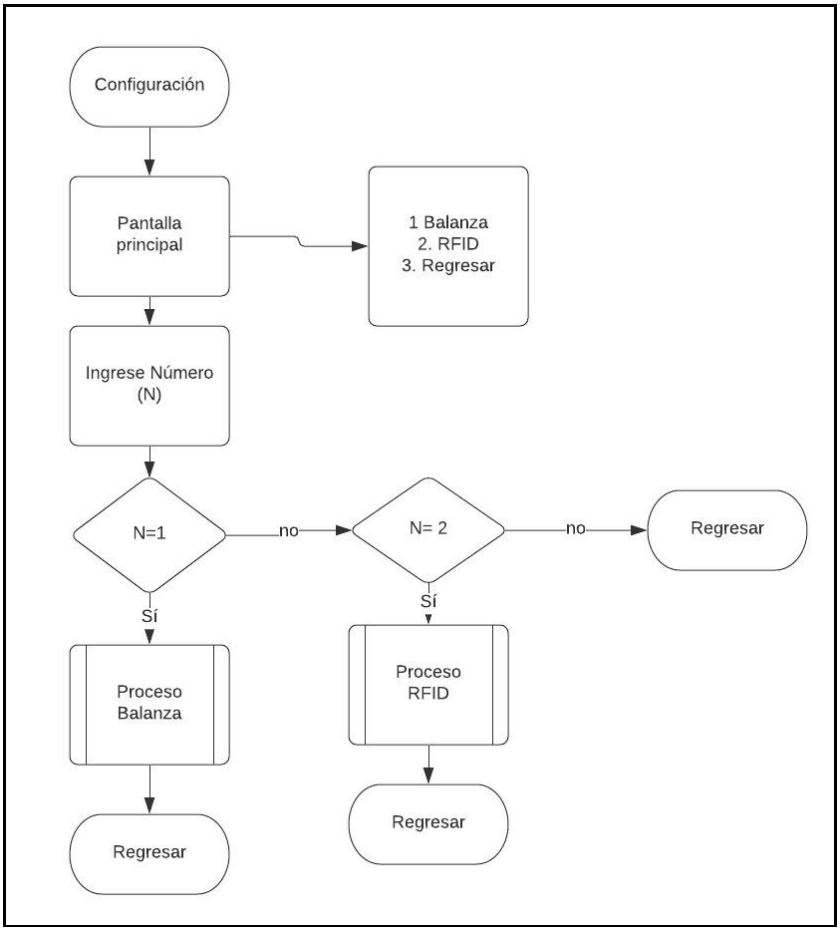
Diagrama de flujo balanza.



Proceso configuración

Figura 9

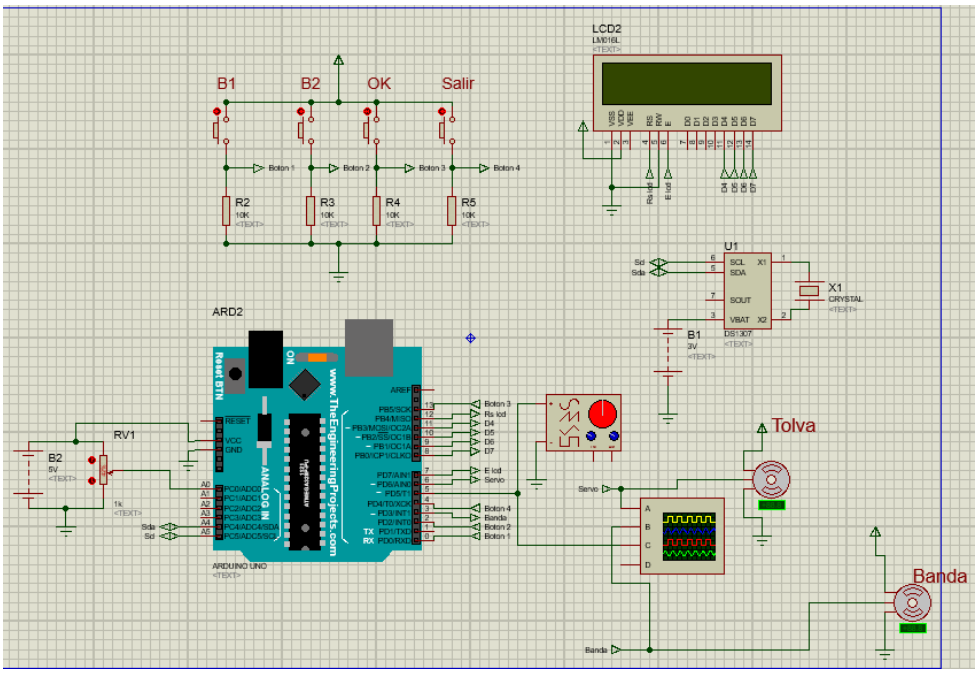
Diagrama de flujo configuración.



Fase 3. Construcción del prototipo:

Figura 10

Esquemático 1



Luego de realizado el respectivo diseño, se procede a la construcción del prototipo del dispensador de alimento para cerdos incluyendo la parte electrónica/programable, se verifica su funcionalidad mediante la realización de pruebas. Finalmente, se determina si el modelo físico diseñado cumple con las especificaciones iniciales.

Consiste en una tolva donde se depositará una gran cantidad de alimento concentrado, contará con un sensor Venturi, que dejará pasar el alimento en pequeñas cantidades para que posteriormente sea censada en un recipiente que en su interior llevará una célula de carga, la cual cumplirá la función de verificar que la cantidad de alimento dispensado sea el solicitado. Para determinar la cantidad de alimento que debe consumir cada cerdo, se instalará en una de sus orejas un dispositivo RFID, (Identificación por radiofrecuencia), y una báscula junto a la zona destinada para el consumo del concentrado, ya que cuando el animal tenga hambre se desplazará

a la zona habitual de alimentación, y será reconocido por el sistema, el cual almacenará datos como su peso, y cantidad de comida ingerida, para posteriormente enviarlos a la web y tener un monitoreo en tiempo real de la nutrición y estado del cerdo.

El suministro de alimento se realizará en base a la tabla nutricional, relacionando el peso y edad de cada animal, puesto que es posible que cada cerdo se encuentre en una etapa de desarrollo distinta; es decir, que cada uno tendrá alimentación personalizada, pata evitar la sobre alimentación y desperdicio del concentrado. Los horarios se realizarán en intervalos de 9 horas, consumiendo un total de dos raciones diarias.

También se podrá realizar una aplicación móvil, que permitirá conocer el estado actual de cada cerdo, y manipular el prototipo de manera manual si así lo requiere el usuario.

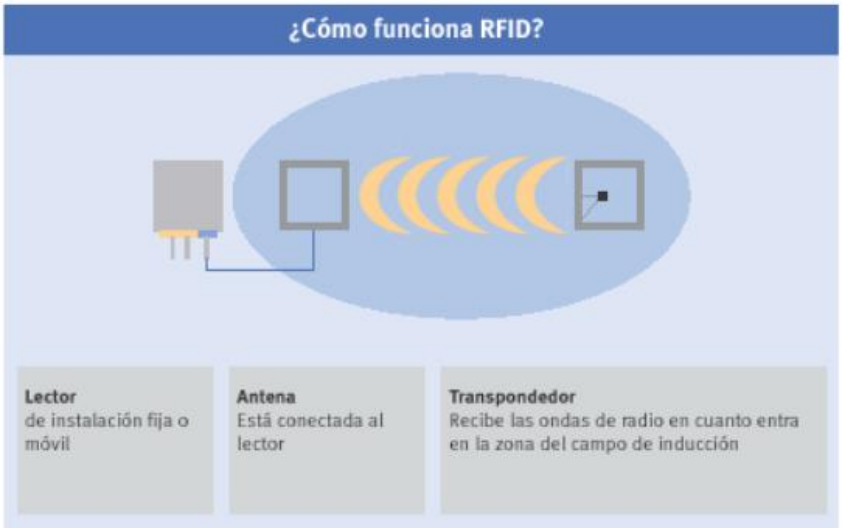
Definición de los sensores para el desarrollo del prototipo.

En el desarrollo del prototipo, se seleccionaron los siguientes sensores:

- Celda de carga. Para el prototipo se hace uso de una sola celda de 20Kgs. Pero para la aplicación final, es un arreglo de celdas de carga de 200 Kgs. Las cuales se conectan sobre una misma tarjeta acondicionadora (HX711).
- Sistema de RFID. En el prototipo se hace uso de una de las varias gamas de frecuencia, trabaja a distancias menores de 2 m.

Figura 11

Funcionamiento de RFID



Nota. *Lectura de tarjetas RFID con arduino y lector mifare RC522*. (16 de 10 de 2021). Obtenido de www.luisllamas.es: https://www.luisllamas.es/arduino-rfid-mifare-rc522/

Rango de trabajo

Baja Frecuencia (125KHz) -Su principal ventaja es su aceptación en todo el mundo, funcionan cerca de los metales y su uso está ampliamente difundido. La distancia de lectura es inferior a 1,5 metros, por lo que las aplicaciones más habituales son la identificación de animales, barriles de cerveza, auto key and lock o bibliotecas, etc.

Alta Frecuencia (13.56 MHz) -Fueron desarrolladas como un sustituto barato y de perfil pequeño. Se pueden adaptar a etiquetas de papel populares en Bibliotecas, identificación de pacientes, movimientos de equipajes de avión o acceso a edificios como también en activos fijos, son sensibles a la presencia de metal y requieren ciertas especificaciones de montaje. Normalmente

se utilizan en aplicaciones tales como la trazabilidad de los productos, siendo su alcance mayor en relación con las frecuencias más bajas.

Ultra Alta Frecuencia (860-960 MHz) -pueden ser leídas a mayores distancias y pueden detectar una mayor cantidad de tags al mismo tiempo. Se usan en el rastreo de líneas de abastecimiento. Tienen grandes ventajas en la recolección automática de datos ya que se elimina el personal. Se usan en Transporte, Sector Salud, Aeronáutica.

Microndas (2.45 GHzo más) -Por lo general son usadas en sistemas RFID activos ofreciendo largas distancias y altas velocidades de transmisión. Tienen un costo más elevado y se usan para seguir vagones de ferrocarril o el pago de casetas de peaje, control de containeres

Figura 12

Ejemplo de RFID para lecturas a 13.56 MHZ

Nota. *Lectura de tarjetas RFID con arduino y lector mifare RC522*. (16 de 10 de 2021).

Obtenido de www.luisllamas.es: https://www.luisllamas.es/arduino-rfid-mifare-rc522/

Código Arduino

Se inicia ingresando las librerías necesarias para el funcionamiento del dispositivo

```
// include the library code:
```

```
#include <SoftwareSerial.h
#include <LiquidCrystal.h>
#include <DS1307.h>
#include <Wire.h>
#include <Servo.h>
```

Luego se crean dos objetos, uno tipo servo llamado “servo” que controlará la tolva y la apertura de esta; el segundo objeto tipo servo está llamado como banda, que será la encargada de indicar hasta donde se desplazará la tolva, y cuando llegue a ese lugar abrir la tolva y que la banda recorra los lugares destinados a cada cerdo.

```
Servo servo;
Servo banda;
```

Luego se crean unas variables tipo String, y es un array que contiene los ID de los cerdos, desde el 1 hasta el 4 que es la cantidad de cerdos establecida en el proyecto, se escogieron números en decenas entre 100 y 1000 y se pone el valor y el equivalente en hexadecimal

```
String Hex="";
String Id[4]={"30012c","400190","5001f4","7002bc"};
String ContFrecu()
```

En práctica esto se realizará empleando en el Arduino un módulo receptor y también emisor, que envía una señal de frecuencia donde al otro extremo se encontrará una tarjeta con un microchip, usualmente en forma de llavero y estos reciben la señal que llega del módulo del Arduino, se alimentan con esa señal y devuelven una señal atenuada ya que un poco de esa energía se consumió alimentándose para funcionar y la envían modulada en frecuencia y dicha modulación en frecuencia al llegar nuevamente al receptor que envió la primera señal la demodula en números de entre 8, 16 ó 32 números que da como resultado el código único de esa tarjeta, que también pueda estar compuesto de números y letras (Hexadecimal), y dicha ID, va dentro de una señal que se envía al código del Arduino y va dentro de la frecuencia, y para suplir

esto y no crear un número tan extenso, por ejemplo se crearía 300 Hertz y su equivalente en hexadecimal y así sucesivamente:

```
String Id[4]={"30012c","400190","5001f4","7002bc"};
```

En el mismo string se tendrá que leer un pin y ver la frecuencia de la señal que entra por dicho pin, ya que será nuestro identificador; para este caso se seleccionó el pin 5, debido a que posee pwm, al que se le enviarán pulsos y deberá contar sólo los flancos ascendentes durante medio segundo, y cuando termine de contarlos, los multiplicará por 2, y nos dirá que, en el periodo de 1 segundo, hay 300 pulsos por ejemplo, o 300 flancos de subida y por lo tanto será una señal de 300 Hertz.

Despues se declararán dos variables, d y dA, que van a estar en bajo, (low), lo que quiere decir que se iniciará en 0,0, es decir sin flancos de subida.

```
 bool d = LOW;
  bool dA = LOW;
```

Esta función quiere decir que al tiempo actual se le suman 500 milisegundos, y se compara con el tiempo medido, si supera, los 500 milisegundos, ya no se cuenta más, pero durante medio segundo lo que se hace es verificar la entrada que hay en el pin número 5, si hay un alto o un bajo, es decir si se tienen 5V o 0V.

```
unsigned long T1=millis()+500UL;
```

Por ejemplo: Si en el pin 5 hay un alto (5V), d=(5), pero como se declara booleana se identificará como un HIGH, entonces d=HIGH, y dA=LOW, lo que quiere decir que hubo un flanco de subida y lo cuenta, por lo tanto sólo se estarían contando los flancos de subida y multiplicando por 2.

```
d = digitalRead(5);
```

```
    if (d == HIGH && dA == LOW)
    {
```

Entonces para la variable que simula la báscula, lo que se hizo es leer un pin analógico que estará conectado a un potenciómetro y lo que hará es variar. Si el potenciómetro está en su parte más baja equivaldrá a 0, pero si se encuentra en su máximo su equivalente será 1023, ya que los pines analógicos en Arduino entregan valores de 0 a 1023.

```
float Peso()
{
  lectura = analogRead(A0)*(70.0/1023.0)+20.0;
  delay(100);
  return lectura;
}
```

Entonces, si llega un valor de 0 sencillamente la multiplicación del código será igual 0, y la lectura del peso equivaldría a 20 Kg ya que en la tabla de pesos por etapa de desarrollo el peso mínimo es de 20kg, y cuando el valor sea 1023 dará 90 kg que es el peso máximo ingresado en la tabla.

Seguido de esto estará la función llamada ración, que primero irá a la variable peso, para después retornar a la variable lectura, la cual tendrá el peso en kilogramos.

```
float Racion()
{
  float P=Peso();
  float Rac=2*pow(10,-5)*pow(P,4)-2.5*pow(10,-3)*pow(P,3)+25.5*pow(10,-
3)*pow(P,2)+16.77*P+0.1637;
  return Rac;
```

Después se tiene una de las funciones primordiales para el funcionamiento del dispositivo, que es el identificador de cerdos, que tomará la frecuencia a la que se estaba sumando el hexadecimal y se preguntará si está dentro de los valores permitidos.

```
}
int IdCer()
{
```

Cuando se entre al while infinito se mostrará en pantalla “esperando”, lo que significa que está a la espera de la llegada de un cerdo.

```
lcd.setCursor(0, 0);
   lcd.print("Esperando....");
   for(int j=0;j<4;j=j+1)
   {
```

Despues de esto lo que se hará es un “for”, desde 0 hasta 4, ya que sólo serán 4 cerdos, y nos enviará a “ContFrecuency” para verificar si es igual a uno de los identificadores establecidos anteriormente.

```
if (ContFrecu()==Id[j])
```

Seguido de esto se creó una función que habilita la tolva y el motor en cuanto se identifique la llegada de uno de los cerdos, dicha habilitación se realizará con un IdCont, que es un array de 4, pero no contiene el identificador del cerdo, si no que contendrá una variable booleana, que cuando equivale a 0, significa que no ha comido, y cuando cambia a 1, significa que ya obtuvo la ración en el horario establecido anteriormente.

```
int aux=0;
 int Idcont[4]={0,0,0,0};
 while(aux<4)
```

Entonces, se solicitará la identificación del cerdo y se verificará que no haya tomado su ración de comida y se iniciará con el proceso de dispensado. Pero llegar a esto aun se debe tener en cuenta el proceso que requiere la banda transportadora y cuando la tolva esté ubicada en la posición solicitada se abrirá y descargará el concentrado.

Por ejemplo, si la ración es de 300 gr, el ciclo “for” permanecerá abierto por 500 milisegundos, y sí es una ración de 1000 o 1200 gr, se mantendrá abierto durante 1,2 Segundos al 100%, y después de medio segundo lo que se hará es cerrar la tolva, seguido de eso se devolvería

el motor dado del caso, (en posición del cerdo 1, no retrocede), y se apagan los motores. Y cuando las alarmas se activen nuevamente los valores se pondrán en 0, para que se puedan alimentar nuevamente en los horarios que establezca el usuario.

```
 lcd.setCursor(0, 0);
      lcd.print("Cerdo#: ");
      lcd.print(id+1);
      lcd.setCursor(0, 1);
      lcd.print("Rac: ");
      lcd.print(Racion());
      lcd.print("grs");
      for (int i = 0; i <=60*id; i += 1){
      banda.attach(3) ;
      banda.write(i);
      delay(15);
      }
      for (int i =0; i <=180; i += 1){
      servo.attach(6) ;
      servo.write(i);
      delay(15);
      }
      delay(Racion());
      for (int i = 180; i >=0; i -= 1){
      servo.attach(6) ;
      servo.write(i);
      delay(15);
      }

      delay(15);
      for (int i =60*id; i >=0; i -= 1){
      banda.attach(3) ;
```

Posteriormente se configura el void setup, que será la primera función en ejecutarse dentro del programa, aquí es donde establecemos algunos criterios que requieren una ejecución única. Inicialmente se establece la configuración para ca uno de los botones o pulsadores y la tarea que ejecutarán.

```
void setup() {
  //Declaracion de botones como entrada
  pinMode(boton1, INPUT); // Arriba
  pinMode(boton2, INPUT); // Abajo
  pinMode(boton3, INPUT);
  pinMode(boton4, INPUT); //ok
  pinMode(5,INPUT); //Identificador
```

La siguiente parte consta de un menú principal, donde se muestra en pantalla cada una de las opciones asignadas a cada pulsador para proceder con la ejecución del programa.

```
void Menu_principal (){
   do{
       lcd.clear();
     do{
   lcd.setCursor(0, 0);
   lcd.print("1- Reloj");
   lcd.setCursor(0, 1);
   lcd.print("2- Alarmas");

   if (digitalRead (boton1)==HIGH){
   delay(T);
   Reloj ();

   }
   if (digitalRead (boton2)==HIGH){
   delay(T);
   Alarmas ();
   }
   } while(digitalRead(boton4) != HIGH);
   delay(T);
   } while(digitalRead(boton4) != HIGH);
   delay(T);
   esperar_activacion();
```

Y de esta manera se hace la configuración para cada una de las variables relacionadas al reloj, cómo: mostrar fecha y hora, ajustar fecha y hora, configurar horas y minutos.

```
void Reloj (){
do{
   lcd.setCursor(0, 0);
   lcd.print("1- Mostrar Reloj");
   lcd.setCursor(0, 1);
   lcd.print("2- Ajustar Reloj");

   if (digitalRead (boton1)==HIGH){
   delay(T);
   lcd.clear ();
   Mostrar_Fecha_Hora ();
   }
   if (digitalRead (boton2)==HIGH){
   lcd.clear ();
   delay(T);
   sprintf(horaActual, "%02d:%02d:%02d", Choras,Cminutos, Csegundos);
   lcd.setCursor(4, 1); // abajo a la izquierda
   lcd.print(horaActual);
   Ajustar_Fecha_Hora ();
   }
   } while(digitalRead(boton4) != HIGH);
```

```
    Menu_principal();
    }
```

Luego se ajustan las alarmas, que pueden ser hasta 20 en los horarios que el usuario establezca.

```
void Alarmas (){
do{
    lcd.setCursor(0, 0);
    lcd.print("1- Mostrar Alarmas");
    lcd.setCursor(0, 1);
    lcd.print("2- Ajustar Alarma");

    if (digitalRead (boton1)==HIGH){
    delay(T);
    lcd.clear ();
    Mostrar_Alarmas ();
    }
    if (digitalRead (boton2)==HIGH){
    lcd.clear ();
    delay(T);
    sprintf(horaActual, "%02d:%02d:%02d", Choras,Cminutos, Csegundos);
    lcd.setCursor(4, 1); // abajo a la izquierda
    lcd.print(horaActual);
    Ajustar_Alarmas ();
    }
    } while(digitalRead(boton4) != HIGH);

    Menu_principal();
    }
```

Finalmente, después de configuradas las alarmas, aparecerá en pantalla “esperarar reacctivación” y así sucesivamete para cada una de las alarmas hasta que el programa retorne.

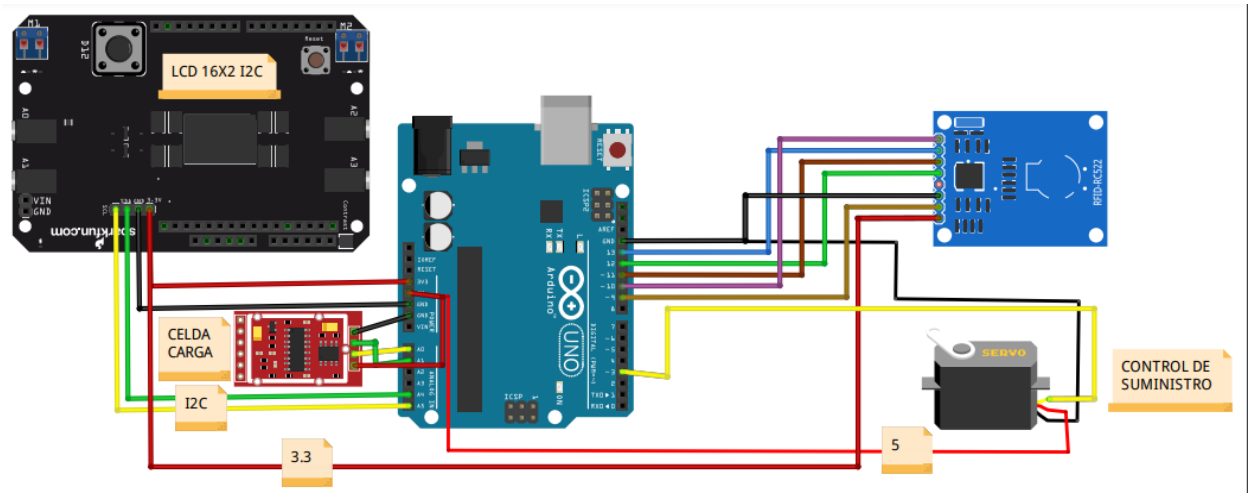
```
void esperar_activacion(){
    lcd.clear ();
 do{ lcd.setCursor(0, 0);
    lcd.print("Esperando alarma");

    for (contador = 0; contador <=20; contador++){
    lcd.setCursor(0, 1);

    RTC.get(rtc,true);
    resulhora=rtc[2]-Ahora[contador];
    resulmin=rtc[1]-Amin[contador];
    lcd.setCursor(4, 1);
    sprintf(horaActual, "%02d:%02d:%02d",rtc[2],rtc[1],rtc[0]);
    lcd.print(horaActual);
```

```
    if (resulhora==0 & resulmin==0){
    lcd.clear ();
    int a = habcom();
    esperar ();
    }
    }

    } while(digitalRead(boton4) != HIGH);
    delay(100);
    Menu_principal();
}


void esperar (){

  do{
    lcd.setCursor(0, 0);
    lcd.print("Reactivacion en :");
    lcd.setCursor(2, 1);
    RTC.get(rtc,true);
    resulhora=rtc[2]-Ahora[contador];
    resulmin=rtc[1]-Amin[contador];
    conteo=60-rtc[0] ;
    lcd.print(conteo);
    lcd.print(" Segundos");
} while(resulhora==0 & resulmin==0);

  lcd.clear();
  esperar_activacion();
}
```

Esquemático general del prototipo:

Figura 13

Esquemático 2.



Empleando la librería <**LiquidCrystal_I2C**.h>, nos permite crear un objeto, que se usará para gestionar el display LCD. Como argumentos recibe una serie de números que se refieren a pines concretos de la placa Arduino, conectados a diferentes pines del display. Los dos primeros números (el 12 y el 11) se refieren a los pines conectado a los puntos RS y E del display. Los cuatro últimos números se refieren a los pines de D4 a D7 del bus de datos del display. En general, esta configuración es la más simple.

Para establecer la comunicación del RFID, con el Arduino se hacen las conexiones de la siguiente manera:

SDA (SS) -> D10

SCK -> D13

MOSI -> D11

MISO -> D12

IRQ -> NO CONECTADO

GND -> GND

RST -> D9

3.3v -> 3.3v

Para conectar esta pantalla a nuestro Arduino tendremos que conectar 4 pines de la siguiente forma:

GND -> GND

VCC -> 5V

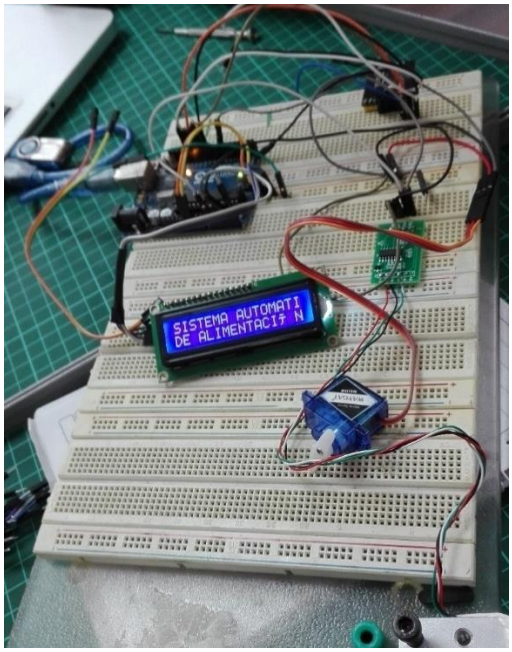
SDA -> A4

SCL -> A5

```
void setup() {
// Inicializar el LCD
  Serial.begin(9600);

// Iniciar sensor - Celda de carga con amplificador HX711
    bascula.begin(pinData, pinClk);

// Iniciar LCD
    lcd.init();
    Serial.println("DHTxx test!");

//Encender la luz de fondo.
    lcd.backlight();

    lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd.
Primera posición(columna:0) de la primera línea(fila:0)
    lcd.print("SISTEMA AUTOMATICO");//Se Muestra texto en la lcd
    lcd.setCursor(0,1); // Se define posiciones donde imprimir en la lcd
    lcd.print("ALIMENTACIÓN CERDO");//Se Muestra texto en la lcd
```

Figura 14.

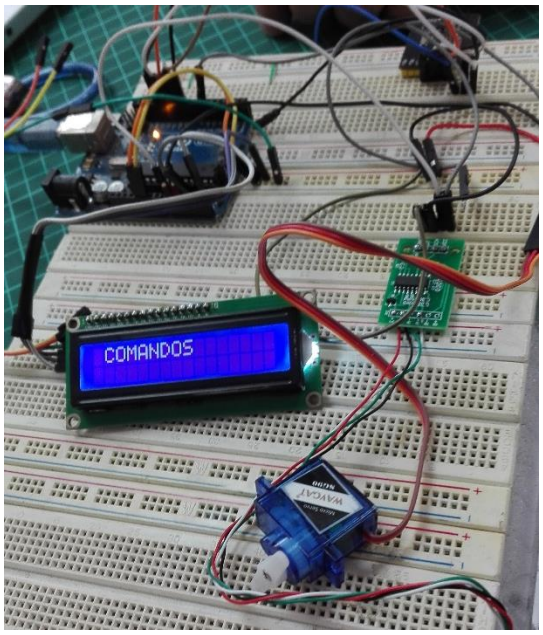
Prueba impresión de texto en pantalla




```
    delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

    lcd.clear(); // Se limpia LCD
    lcd.setCursor(2,1); //Se define posiciones donde imprimir en la lcd
    lcd.print(" COMANDOS");//Se Muestra texto en la lcd
    delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd
```

Figura 15

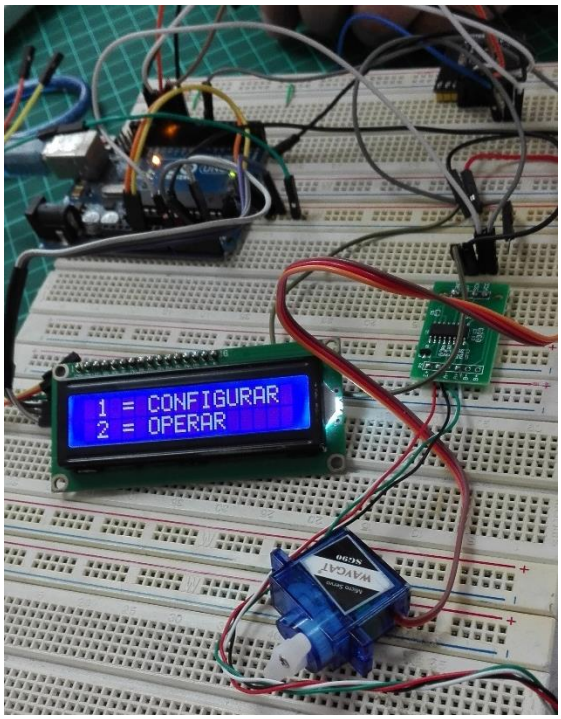
Prueba impresión de texto en pantalla, comandos




```
lcd.clear(); // Se limpia LCD
lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
lcd.print(" 1 = CONFIGURAR ");//Se Muestra texto en la lcd
lcd.setCursor(0,1); //Se define posiciones donde imprimir en la lcd
lcd.print(" 2 = OPERAR ");//Se Muestra texto en la lcd
```

Figura 16

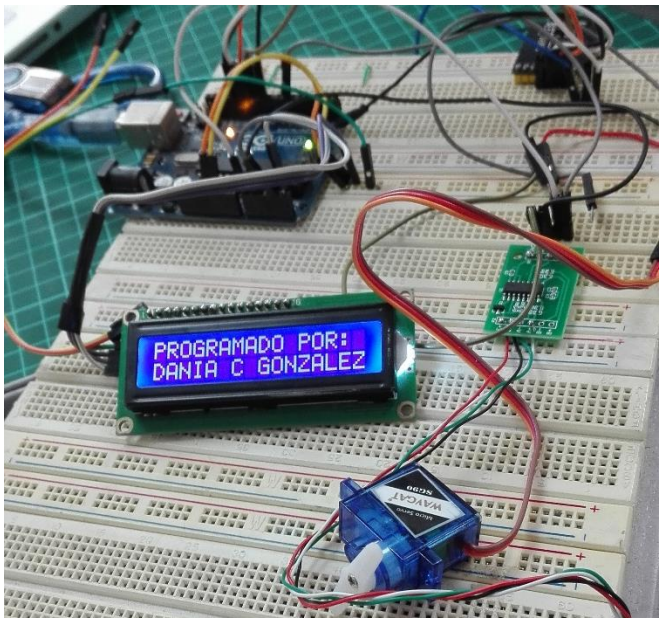
Prueba impresión de texto en pantalla, configuraciones.




```
delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

 lcd.clear(); // Se limpia LCD
 lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
 lcd.print(" 3 = SALIR  ");//Se Muestra texto en la lcd

 delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

 lcd.clear(); // Se limpia LCD
 lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
 lcd.print("PROGRAMADO POR:");//Se Muestra texto en la lcd
 lcd.setCursor(0,1);//Se define posiciones donde imprimir en la lcd
 lcd.print("DANIA GONZALEZ");//Se Muestra texto en la lcd
 delay(1000); // Retardo de 1 segundo mostrando el texto en la lcd
```

Figura 17.

Prueba impresión de texto en pantalla



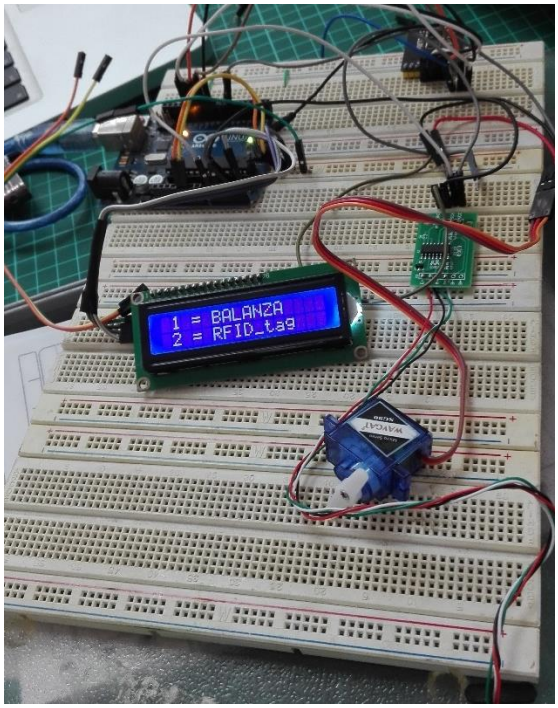

```
    while (!Serial);    // No hacer nada si no se abre ningún puerto serie
(añadido para Arduinos basados en ATMEGA32U4)
    SPI.begin();       // Inicia bus SPI
    mfrc522.PCD_Init();   // Inicia MFRC522

    delay(4);        // Retraso opcional. Algunos tarjetas necesitan más
tiempo después del init para estar listo.

    mfrc522.PCD_DumpVersionToSerial();  // Mostrar detalles de PCD - MFRC522.
Detalles del lector de tarjetas
    Serial.println(F("Escanear PICC para ver UID, SAK, tipo y bloques de
datos..."));
}
}

void loop() {

   if(Serial.available() > 0){
             Estado = Serial.read();
        }

       if (Estado == '1') { // Comparamos la variable Estado, si Estado == a
"1", llama la funcion Avanzar

          Configurar();
           Estado = 0;
       }
```

```
       if (Estado == '2')// Comparamos la variable Estado, si Estado == a
"2", llama la funcion Retroceder
       {
          Operacion();
          Estado = 0;
       }
        if (Estado == '3')// Comparamos la variable Estado, si Estado == a
"3", llama la funcion Detener
       {
          Detener();
           Estado = 0;
       }

}


void Configurar() {
    lcd.clear(); // Se limpia LCD
    lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
    lcd.print(" 1 = BALANZA ");//Se Muestra texto en la lcd
    lcd.setCursor(0,1); //Se define posiciones donde imprimir en la lcd
    lcd.print(" 2 = RFID_tag ");//Se Muestra texto en la lcd

}

void loop() {

   if(Serial.available() > 0){
             Estado = Serial.read();
        }

       if (Estado == '1') { // Comparamos la variable Estado, si Estado == a
"1", llama la funcion Avanzar

          Configurar();
           Estado = 0;
       }

       if (Estado == '2')// Comparamos la variable Estado, si Estado == a
"2", llama la funcion Retroceder
       {
          Operacion();
          Estado = 0;
       }
        if (Estado == '3')// Comparamos la variable Estado, si Estado == a
"3", llama la funcion Detener
       {
          Detener();
           Estado = 0;
       }

}


void Configurar() {
    lcd.clear(); // Se limpia LCD
```

```
    lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
    lcd.print(" 1 = BALANZA ");//Se Muestra texto en la lcd
    lcd.setCursor(0,1); //Se define posiciones donde imprimir en la lcd
    lcd.print(" 2 = RFID_tag ");//Se Muestra texto en la lcd
```

Figura 18

Prueba impresión de texto en pantalla, balanza.

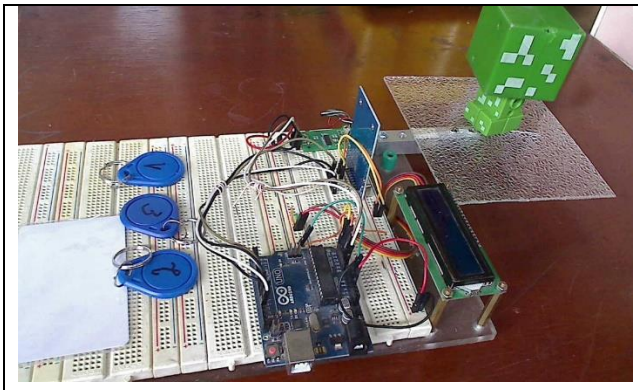



```
    delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

    lcd.clear(); // Se limpia LCD
    lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
    lcd.print(" 3 = REGRESAR  ");//Se Muestra texto en la lcd

    if (Estado == '1') { // Comparamos la variable Estado, si Estado == a
"1", llama la funcion Avanzar

          balanza_tara();
           Estado = 0;
       }

       if (Estado == '2')// Comparamos la variable Estado, si Estado == a
"2", llama la funcion Retroceder
       {
          rfid_tag();
          Estado = 0;
       }
```

```
        if (Estado == '3')// Comparamos la variable Estado, si Estado == a
"3", llama la funcion Detener
       {
          return;

       }

    delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

    lcd.clear(); // Se limpia LCD
    lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
    lcd.print(" 3 = REGRESAR  ");//Se Muestra texto en la lcd

    if (Estado == '1') { // Comparamos la variable Estado, si Estado == a
"1", llama la funcion Avanzar

          balanza_tara();
           Estado = 0;
       }

       if (Estado == '2')// Comparamos la variable Estado, si Estado == a
"2", llama la funcion Retroceder
       {
          rfid_tag();
          Estado = 0;
       }
        if (Estado == '3')// Comparamos la variable Estado, si Estado == a
"3", llama la funcion Detener
       {
          return;

       }
```

Figura 19

Prototipo armado



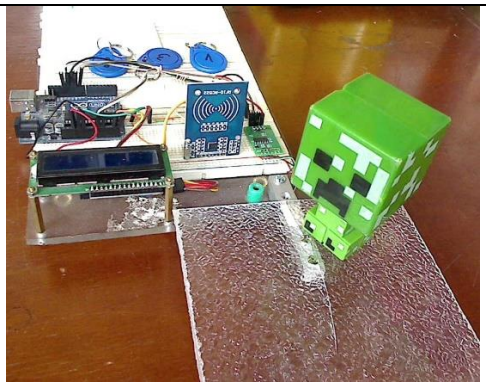


Tabla 4

Presupuesto estimado para prototipos futuros.

RECURSO	DESCRIPCIÓN	CANTIDAD	VALOR UNIDAD	
Equipo Humano	Mano de obra diseño e instalación del dispensador de alimento.	-	$200.000	$200.000
Equipos y Software	Necesarios para la realización del sistema.	-	$150.000	$150.000
Microcontrolador	Arduino UNO o similares, se encargará de la parte lógica	1	$50.000	$50.000
Sensores tipo compuerta	Venturi, encargado de soltar gradualmente el alimento	2	$30.000	$60.000
RFID y módulo RC522	Necesario para el reconocimiento de cada cerdo.	Aprox. 4	$10.000	$40.000
Célula de carga Y módulo HX711	Se emplearán en la parte del censo de la comida y para las básculas.	Aprox. 10	$15.000	$150.000
Motor	Necesario para el funcionamiento eléctrico básico.	1		$300.000
Tolva y elementos de bricolaje.	Elementos principalmente elaborados en material reciclable.	-	$50.000	$50.000
. Materiales y suministros	Esenciales para la construcción de la instalación.	-	$200.000	$200.000
	TOTAL $ 1’200.000			

Resultados y discusión

Trabajo futuro y posibles mejoras:

Mediante ciertos ajustes es posible adecuar el proyecto para cubrir las necesidades alimenticias en caballerizas, graneros, establos y diversos criaderos donde el control de alimento resulta crucial para el desempeño del animal y así mismo la calidad.

Pero es necesario incorporar más dispositivos para lograr tener el sistema en funcionamiento. Se puede pensar en una solución con un sistema Single Board Computer (SBC), como por ejemplo el Raspberry Pi 3B+ o el 4. Allí se podría construir bases de datos, los reportes a la nube (IoT) o las publicaciones similares a MQTT.

Conclusiones

De acuerdo al diseño se concluye que es apto para ser empleado en pequeña y mediana producción porcícola para optimizar la calidad de la carne, la salud de los animales y reducir los costos por perdida de alimento.

Es posible dispensar el concentrado en los horarios establecidos por el usuario en base a la etapa de desarrollo, de cada porcino, y mediante el uso de la tarjeta RFID se podrán almacenar/crear una curva de crecimiento por cerdo.

Mediante el uso del Arduino uno, el dispositivo es de fácil acceso al campesino, debido a que se ensambla con componentes económicos y se puede emplear en diversos animales de consumo.

Bibliografía

Alvarez, E. (2005). *Sistema electrónico dosificador de alimento para la producción de alevinos aplicado a la piscicultura.* Neiva: Universidad Surcolombiana. Facultad de Ingeniería.

Andersson Alvear Betancourt,Sergio Andrés Peñuela Argüello. (2019). *Automatización de un sistema dosificador de alimento para alevinos teniendo en cuenta el porcentaje de oxígeno y la temperatura del agua en un ambiente controlado.* Neiva, Huila: Universidad Surcolombiana.

Blog Tecnowebstudio. (28 de Enero de 2016). *Historia de las primeras máquinas expendedoras*. tecnowebstudio: https://tecnowebstudio.com/historia-de-las-primeras-maquinas-expendedoras/

Claudia Ximena Villalba Linares, Nestor Javier Morales Galarza. (2007). *Dispensador de comida para canes de uso domestico.* Bogotá, D.C: Universidad de San Buenaventura .

Definiciona. (3 de Marzo de 2009). *Automático* https://definicion.de/automatico/

Definiciona. (22 de Enero de 2016). *Dispensador*. https://definiciona.com/dispensador/

Departamento de Agricultura y Protección del Consumidor. (28 de Noviembre de 2014). *Cerdos y la producción animal*. http://www.fao.org/ag/againfo/themes/es/pigs/production.html

Flórez, L. E. (2007). *Desarrollo de un dosificador automático para alimentar cerdos*. Universidad Autónoma De Occidente Facultad de ingeniería: https://red.uao.edu.co/bitstream/10614/6338/1/T04347.pdf

González, X. (27 de Febrero de 2019). *El Sector Porcícola Colombiano Mueve Al Año $2,6 Billones En Términos De Producción*. agronegocios: https://www.agronegocios.co/ganaderia/el-sector-porcicola-colombiano-mueve-al-ano-26-billones-en-terminos-de-produccion-2832964

Huila, R. D. (2 de Agosto de 2017). *Porcicultores en la ruta de la formalidad*. Diario Del Huila: https://www.diariodelhuila.com/porcicultores-en-la-ruta-de-la-formalidad

Jorge Ivan Zapata Valencia, Daniel Alejandro Gil Agudelo. (2017). *Diseño e implementación de un prototipo de dispensador automático de comida para animales basado en raspberry pi controlado mediante una aplicación móvil.* Pereira: Universidad Tecnológica de Pereira, Facultad de ingeniería.

Mascotas, B. (7 de Septiembre de 2017). *¿Qué utilidades tiene los dispensadores de comida para perros?* bigmascotas: https://bigmascotas.com/blog/utilidades-los-dispensadores-comida-perros/

rutanmedellin. (5 de Enero de 2013). *TRANSFERENCIA DE TECNOLOGÍA*. rutanmedellin: https://www.rutanmedellin.org/es/recursos/abc-de-la-innovacion/item/transferencia-de-tecnologia

Sistema de información de Gestión y Desempeño de Organizaciones de Cadenas. (Diciembre de 2019). *Cadena Carnica Porcina*. Minagricultura: https://sioc.minagricultura.gov.co/Porcina/Documentos/2019-12-30%20Cifras%20sectoriales.pdf

Thompson, I. (Diciembre de 2005). *Definición de Mercado*.promonegocios: https://www.promonegocios.net/mercadotecnia/mercado-definicion-concepto.html

Resumen analítico educativo - RAE

Título del texto	Diseño de un dispensador automático de alimento concentrado para cerdos
Nombres y Apellidos del Autor	Dania Carolina González González
Año de la publicación	2021
Resumen del texto: Mediante el presente proyecto aplicado se pretende diseñar e implementar un dispensador de alimento concentrado para cerdos, empleando materiales asequibles y económicos, con el fin de automatizar el proceso de administración de concentrado, mejorando su aprovechamiento al proveer raciones alimenticias proporcionales a los requerimientos de la tabla nutricional en horarios exactos. Como resultado, esperamos contribuir en la superación de las dificultades de este sector productivo, brindando herramientas de tecnificación que aporten significativamente para hacer de la porcicultura colombiana una actividad más productiva y competitiva, especialmente para los porcicultores de nuestra región.	
Palabras Claves	Dispensador, porcicultura, automatización, competitividad, optimización
Problema que aborda el texto: Diseño de un prototipo de dispensador automático para el beneficio de los productores de cerdo, reduciendo costos y minimizando el desperdicio de alimento.	
Objetivos del texto: Diseño e implementación de un dispensador automático de alimento concentrado para cerdos, en una unidad productiva del Municipio de Palermo-Huila. Indagación sobre equipos similares en el mundo. Apropiación del proceso de alimentación, identificación de variables, construcción del modelo.	
Hipótesis planteada por el autor: Los dosificadores son dispositivos utilizados para regular el despacho de un producto en las diferentes etapas de un proceso. Están compuestos por servomotores, motores eléctricos, electroimanes, cilindros neumáticos y/o reguladores electrónicos. Existen diversas clases de dosificadores que se clasifican de acuerdo al modo de servicio y a la naturaleza de la sustancia a manipular. Y dentro de la categoría de dosificadores volumétricos de sólidos secos, existen al menos tres tipos de mecanismo de dosificación, los de tornillo, (Sin fin), compuerta rotativa, y banda rodante. Se identificó el sector porcícola como un área ideal para automatizar, principalmente para los pequeños y medianos productores. Mediante el prototipo se reducirán costos en alimentación y se obtendrá un producto de mayor calidad en un tiempo estimado de seis meses, pero para ello primero se tuvo que indagar sobre la porcicultura en Colombia, y conocer cifras, siendo alarmante el porcentaje de sobre alimentación y así mismo el desperdicio de concentrado por cerdo. Debido a esto, y como proyecto de opción de grado del programa de Ingeniería Electrónica, se realizaron varios prototipos, y se plantearon varios componentes, como el Arduino Leonardo, rasberry, entre otros, pero se optó por el Arduino uno, debido a su fácil adquisición en el mercado y reducción de costos, ya que este prototipo se podrá elaborar con materiales de reciclaje, o como lo prefiera el usuario.	

Finalmente se elaboró el dispositivo con, una tarjeta RFID y su módulo RC522, llaveros de acceso, celdas de carga con módulo HX711, servomotor, Y una pantalla LED 16x2; y mediante el uso de la tarjeta RFID se podrán almacenar/crear una curva de crecimiento por cerdo. A través de ciertos ajustes es posible adecuar el proyecto para cubrir las necesidades alimenticias en caballerizas, graneros, establos y diversos criaderos donde el control de alimento resulta crucial para el desempeño del animal y así mismo la calidad.

Tesis principal del autor:
Los dispensadores automáticos, son una gran herramienta de apoyo para el sector agropecuario, optimizando las tareas diarias del campo, es por ello que el prototipo "Dispensador automático de alimento concentrado para cerdos" es un dispositivo al alcance de pequeños y medianos productores de cerdo por su bajo costo y fácil instalación.

Argumentos expuestos por el autor:
Inicialmente se consultan cifras respecto a la producción de cerdo, cantidad de concentrado diario, número de raciones y pesos ideales según etapa de desarrollo, y tabla de pérdidas por suministro inadecuado de las raciones; en la justificación y en base a las cifras anteriores se propone el desarrollo de un dispensador automático para cerdos el cual daría solución a la problemática anteriormente mencionada, optimizando las tareas diarias del porcicultor, posteriormente se realizó una investigación y se elaboró el marco referencial y teórico, conociendo los inicios de los dispensadores y proyectos relacionados, proporcionando una idea más amplia respecto a la automatización. En el capítulo I, se encuentra la estructura básica del dispensador, y sus respectivos diagramas de flujo por proceso, para e capitulo II, se encuentra el desarrollo del código. Finalmente se agrega la bibliografía y los anexos.

Conclusiones del texto:
De acuerdo al diseño se concluye que es apto para ser empleado en pequeña y mediana producción porcícola para optimizar la calidad de la carne, la salud de los animales y reducir los costos por perdida de alimento.
Es posible dispensar el concentrado en los horarios establecidos por el usuario en base a la etapa de desarrollo, de cada porcino, y mediante el uso de la tarjeta RFID se podrán almacenar/crear una curva de crecimiento por cerdo.

Bibliografía citada por el autor:

Bibliografía

Alvarez, E. (2005). *Sistema electrónico dosificador de alimento para la producción de alevinos aplicado a la piscicultura.* Neiva: Universidad Surcolombiana. Facultad de Ingeniería.

Andersson Alvear Betancourt,Sergio Andrés Peñuela Argüello. (2019). *Automatización de un sistema dosificador de alimento para alevinos teniendo en cuenta el porcentaje*

de oxígeno y la temperatura del agua en un ambiente controlado. Neiva, Huila: Universidad Surcolombiana.

Blog Tecnowebstudio. (28 de Enero de 2016). *Historia de las primeras máquinas expendedoras*. tecnowebstudio: https://tecnowebstudio.com/historia-de-las-primeras-maquinas-expendedoras/

Claudia Ximena Villalba Linares, Nestor Javier Morales GALARZA. (2007). *Dispensador de comida para canes de uso domestico.* Bogotá, D.C: Universidad de San Buenaventura .

Definiciona. (3 de Marzo de 2009). *Automático* https://definicion.de/automatico/

Definiciona. (22 de Enero de 2016). *Dispensador*. https://definiciona.com/dispensador/

Departamento de Agricultura y Protección del Consumidor. (28 de Noviembre de 2014). *Cerdos y la producción animal*. http://www.fao.org/ag/againfo/themes/es/pigs/production.html

FLÓREZ, L. E. (2007). *Desarrollo de un dosificador automático para alimentar cerdos*. Universidad Autónoma De Occidente Facultad de ingeniería: https://red.uao.edu.co/bitstream/10614/6338/1/T04347.pdf

González, X. (27 de Febrero de 2019). *El Sector Porcícola Colombiano Mueve Al Año $2,6 Billones En Términos De Producción*. agronegocios: https://www.agronegocios.co/ganaderia/el-sector-porcicola-colombiano-mueve-al-ano-26-billones-en-terminos-de-produccion-2832964

Huila, R. D. (2 de Agosto de 2017). *Porcicultores en la ruta de la formalidad*. Diario Del Huila: https://www.diariodelhuila.com/porcicultores-en-la-ruta-de-la-formalidad

Jorge Ivan Zapata Valencia, Daniel Alejandro Gil Agudelo. (2017). *Diseño e implementación de un prototipo de dispensador automático de comida para animales basado en raspberry pi controlado mediante una aplicación móvil.* Pereira: Universidad Tecnológica de Pereira, Facultad de ingeniería.

Mascotas, B. (7 de Septiembre de 2017). *¿Qué utilidades tiene los dispensadores de comida para perros?* bigmascotas: https://bigmascotas.com/blog/utilidades-los-dispensadores-comida-perros/

rutanmedellin. (5 de Enero de 2013). *TRANSFERENCIA DE TECNOLOGÍA*. rutanmedellin: https://www.rutanmedellin.org/es/recursos/abc-de-la-innovacion/item/transferencia-de-tecnologia

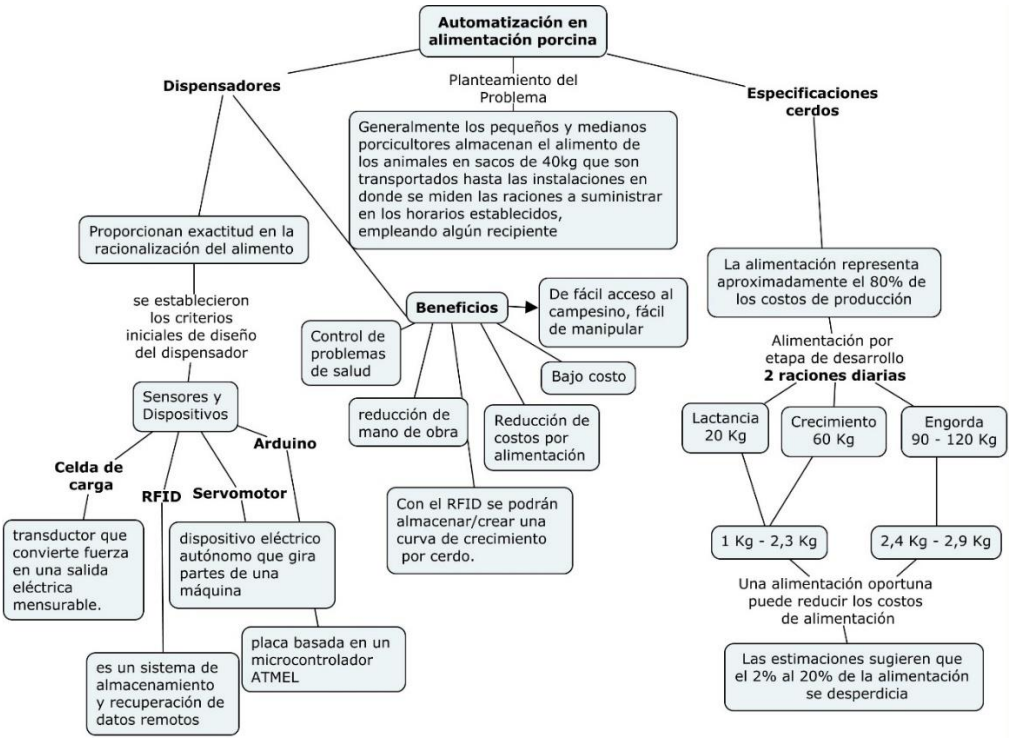
Sistema de información de Gestión y Desempeño de Organizaciones de Cadenas. (Diciembre de 2019). *Cadena Carnica Porcina*. Minagricultura:

https://sioc.minagricultura.gov.co/Porcina/Documentos/2019-12-30%20Cifras%20sectoriales.pdf

Thompson, I. (Diciembre de 2005). *Definición de Mercado*.promonegocios: https://www.promonegocios.net/mercadotecnia/mercado-definicion-concepto.html

Nombre y apellidos de quien elaboró este RAE	Dania Carolina González González
Fecha en que se elaboró este RAE	13 de octubre 2021

Imagen (mapa conceptual) que resume e interconecta los principales conceptos encontrados en el texto:



Comentarios finales

Mediante el uso del Arduino uno, el dispositivo es de fácil acceso al campesino, debido a que se ensambla con componentes económicos y se puede emplear en diversos animales de consumo.

Anexo 1 - Sensores

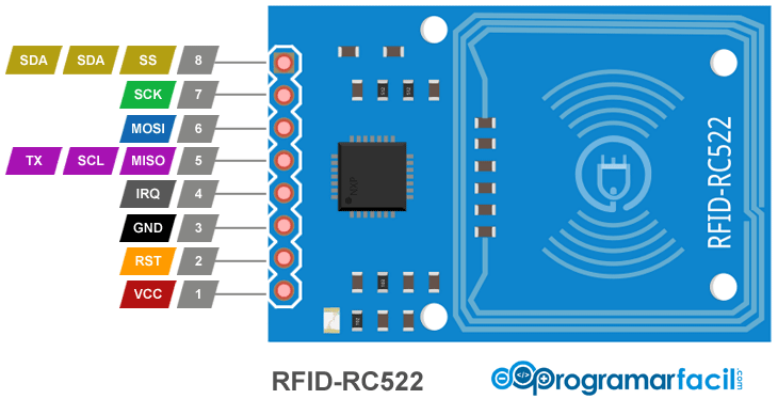
Sistema RFID:

Datasheet RC522

- ***VCC***: pin de alimentación del lector RFID RC522. Admite un voltaje de alimentación entre 2,5V y 3,3V.
- ***RST***: es un pin para encender y apagar el módulo. Mientras el pin esté en estado LOW se mantendrá apagado sin apenas consumir. Cuando el estado cambia a HIGH el RC522 se reinicia.
- ***GND***: pin de tierra o GND.
- ***IRQ***: pin de interrupción que alerta al microcontrolador cuando una etiqueta RFID se acerca al lector RFID RC522.
- ***MISO/SCL/TX***: este pin tiene tres funciones. Cuando la interfaz SPI está habilitada funciona como salida de esclavo y entrada de máster. Cuando está activada la interfaz I2C funciona como señal de reloj y como salida serie cuando la interfaz UART está habilitada.
- ***MOSI***: entrada en la interfaz SPI.
- ***SCK***: señal de reloj de la interfaz SPI.
- ***SS/SDA/RX***: el pin actúa como entrada de señal cuando la interfaz SPI está habilitada. Si la interfaz I2C está activa actúa como entrada de datos y como entrada de datos serie cuando la interfaz UART está habilitada.

Figura 17

RC522



Nota. *Lectura de tarjetas RFID con arduino y lector mifare RC522*. (16 de 10 de 2021). www.luisllamas.es: https://www.luisllamas.es/arduino-rfid-mifare-rc522/

Conexión del entre el módulo RFID y Arduino

Su principio de funcionamiento consiste en pasar un TAG, cerca de un lector RFID, el TAG tiene la capacidad de enviar información al lector. Dicha información puede ser desde un simple código o todo un paquete de información guardo en la memoria del Tag.

Tabla 5

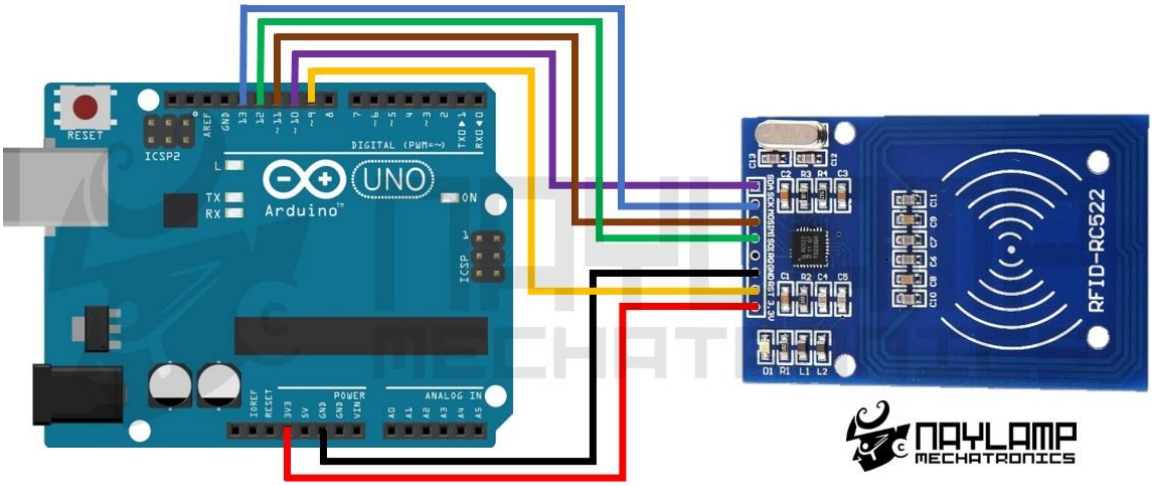
Conexión entre el Arduino y el RFID

Módulo RC522	Arduino Uno, Nano
SDA (SS)	10
SCK	13
MOSI	11
MISO	12

Módulo RC522	Arduino Uno, Nano
IRQ	No conectado
GND	GND
RST	9
3.3V	3.3V

Figura 20

Conexión Arduino uno – RC522



Nota. *Lectura de tarjetas RFID con arduino y lector mifare RC522*. (16 de 10 de 2021). www.luisllamas.es: https://www.luisllamas.es/arduino-rfid-mifare-rc522/

Celda de carga

Comunicación celda de carga – Arduino uno (HX711)

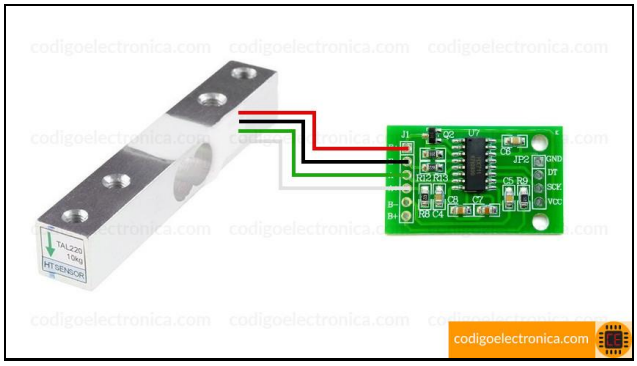
El HX711 es un convertidor analógico a digital de precisión de 24 bits (ADC) diseñado para básculas de pesaje y aplicaciones de control industrial para interactuar directamente con un sensor de puente.

El chip HX711 posee internamente la electrónica para la lectura del puente de Wheatstone formado por la celda de carga y también un conversor ADC de 24 bits. Se comunica con el microcontrolador por medio de un protocolo de tipo serial mediante 2 pines (Clock y Data).

Conexión modulo hx711 con celda de carga:

Figura 21

Celda de carga y módulo HX711

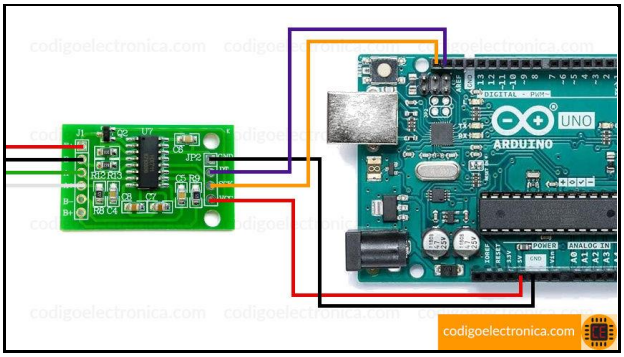


Nota. *Arduino celda de carga con hx711*. (17 de 09 de 2021). Obtenido de codigoelectronica.com: http://codigoelectronica.com/blog/arduino-celda-de-carga-con-hx711

Conexión hx711 con Arduino:

Figura 22

comunicación Arduino uno, y módulo HX711



Nota. *Arduino celda de carga con hx711*. (17 de 09 de 2021). Obtenido de codigoelectronica.com: http://codigoelectronica.com/blog/arduino-celda-de-carga-con-hx711

Anexo 2 - Código Arduino

Código completo:

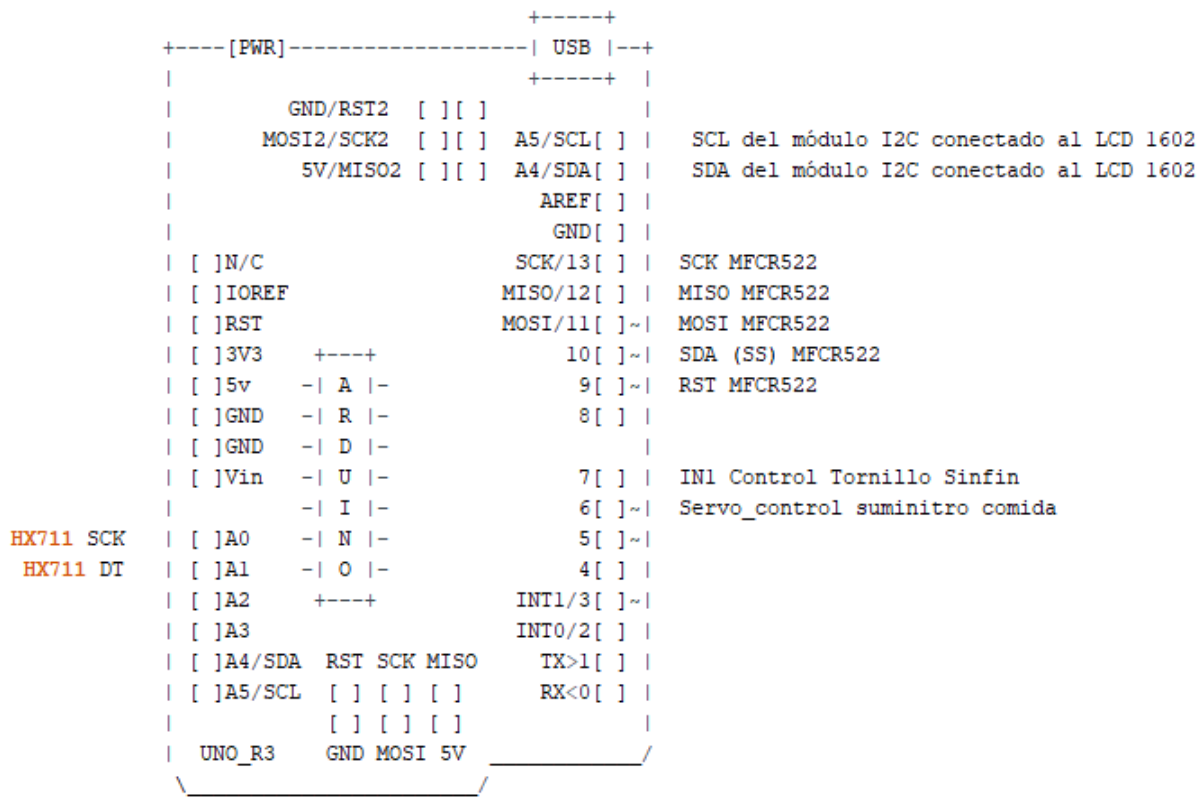
/*

*Proyecto de grado

* Dania Carolina Gonzalez Gonzalez

* Sistema automatico de alimentación de cerdos

* ESQUEMA DE CONEXION



NOTAS:

- La alimentación y la masa del módulo LCM 1602 I2C V1 van directamente conectadas a VCC (+5V) y GND
  respectivamente.
- Conexiones del transmisor de celda de carga HX711:
  - ALIMENTACIÓN:
    - Pin VCC del HX711 --> +5V de Arduino.
    - Pin GND del HX711 --> GND de Arduino.
  - ENTRADAS:
    - Pin E+ del HX711 --> Cable Rojo de la celda de carga.
    - Pin E- del HX711 --> Cable Negro de la celda de carga.

```
      - Pin A- del HX711 --> Cable Blanco de la celda de carga.
      - Pin A+ del HX711 --> Cable Verde de la celda de carga.
    - SALIDAS:
      - Pin SCK del HX711 --> Pin analógico A0 de Arduino.
      - Pin DT del HX711 --> Pin Analógico A1 de Arduino.
* UNAD - 2021
*/

// Librerias necesarias
#include <Wire.h>                    //Conexiòn al protocolo I2C
#include <LiquidCrystal_I2C.h>          //Libreria para LCD_I2C
#include <Servo.h>                   //Libreria para Servo_dosificador de alimento

#include <SPI.h>                    //comunicacion con tarjeta RFID - Serial Peripheral Interface
#include <MFRC522.h>                 //libreria MFRC522 - RFID

#include <HX711.h>                   //Libreria para el amplificador de la celda de carga - báscula

#define RST_PIN 9                    // Pin de reset del RC522,
#define SS_PIN  10                   // Pin de  SS para el RC522,

byte pinData = A1;                  // Pin analógico A1 para el pin DT del transmisor de celda de
carga HX711
byte pinClk = A2;                   // Pin analógico A0 para el pin SCK del transmisor de celda de
carga HX711

const int IN1= 7;                    // Coneciòn de motor tornillo sinfin llenador de alimentos

HX711 bascula;                       // Creación del objeto para el transmisor de celda de carga
HX711

Servo servoMotor;                    // Declaramos la variable para controlar el servo

float factor_calibracion = 20780.0;      //Parámetro para calibrar el peso y el sensor. Valor inicial
float Estado = 0;
int LecturaUID[16];

/*Típica distribución de pines utilizada para tarjeta MFRC522:
 * ------------------------------------
 *             MFRC522      Arduino
 *             Reader/PCD   Uno/101
 * Signal      Pin          Pin
 * ------------------------------------
 * RST/Reset   RST          9
 * SPI SS      SDA(SS)      10
 * SPI MOSI    MOSI         11 / ICSP-4
```

```
 * SPI MISO    MISO        12 / ICSP-1
 * SPI SCK     SCK         13 / ICSP-3
 */


LiquidCrystal_I2C lcd(0x27,16,2);        //Crear el objeto lcd  dirección  0x27 y 16 columnas x 2
filas
MFRC522 mfrc522(SS_PIN, RST_PIN);        //Crear instancia para el MFRC522
//MFRC522::MIFARE_Key key;

// FUNCIONES:
void Configurar(void);
void Operacion(void);
void Calibrar_Balanza(void);
void Configurar_rfid_tag(void);
void rfid_Leer_tag();
void rfid_Escribir_tag();
void rfid_tag();
void rfid_Lectura_Escritura(void);
void Pesar(void);
void dump_byte_array(void);

void setup() {

   Serial.begin(9600);
   servoMotor.attach(6);


   bascula.begin(pinData, pinClk);       // Iniciar sensor - Celda de carga con amplificador HX711
   bascula.set_scale();                  // Aplicar la calibración
   bascula.tare();                      // Iniciar la tara. No tiene que haber nada sobre el peso
   long zero_factor = bascula.read_average(); // Obtener una lectura de referencia
   Serial.print("Zero factor: ");          // Mostrar la primera desviación
   Serial.println(zero_factor);

   lcd.init();                         // Iniciar LCD
   lcd.backlight();                     //Encender la luz de fondo.

   lcd.setCursor(0,0);                  //Se define posiciones donde imprimir en la lcd. Primera
posición(columna:0) de la primera línea(fila:0)
   lcd.print("SISTEMA AUTOMATICO");      //Se Muestra texto en la lcd
   lcd.setCursor(0,1);                  // Se define posiciones donde imprimir en la lcd
   lcd.print("DE ALIMENTACION ");        //Se Muestra texto en la lcd

   delay(2000);                         // Retardo de 1 segundo mostrando el texto en la lcd
```

```
   lcd.clear();                    // Se limpia LCD
   lcd.setCursor(0,0);             //Se define posiciones donde imprimir en la lcd
   lcd.print("PROGRAMADO POR:");    //Se Muestra texto en la lcd
   lcd.setCursor(0,1);             //Se define posiciones donde imprimir en la lcd
   lcd.print("DANIA C GONZALEZ");   //Se Muestra texto en la lcd
   delay(1000);                    // Retardo de 1 segundo mostrando el texto en la lcd

   while (!Serial);                // No hacer nada si no se abre ningún puerto serie (añadido para
Arduinos basados en ATMEGA32U4)
   SPI.begin();                    // Inicia bus SPI
   mfrc522.PCD_Init();             // Función  que inicializa RFID-MFRC522.PCD= Proximity
Coupling Device (Unidad lectora)

   delay(4);                       // Retraso opcional. Algunos tarjetas necesitan más tiempo después
del init para estar listo.

}


void loop() {

   lcd.clear(); // Se limpia LCD
   lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
   lcd.print(" COMANDOS");//Se Muestra texto en la lcd
   delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

   lcd.clear(); // Se limpia LCD
   lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
   lcd.print(" 1 = CONFIGURAR ");//Se Muestra texto en la lcd
   lcd.setCursor(0,1); //Se define posiciones donde imprimir en la lcd
   lcd.print(" 2 = OPERAR ");//Se Muestra texto en la lcd

   delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

   if(Serial.available() > 0){
         Estado = Serial.read();
       }

      if (Estado == '1') { // Comparamos la variable Estado, si Estado == a "1", llama la funcion
Configurar

         Configurar();
          Estado = 0;
      }
```

```
      if (Estado == '2')// Comparamos la variable Estado, si Estado == a "2", llama la funcion
Operacion
      {
        Operacion();
        Estado = 0;
      }
}

void Configurar() {
   lcd.clear(); // Se limpia LCD
   lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
   lcd.print(" 1 = BALANZA ");//Se Muestra texto en la lcd
   lcd.setCursor(0,1); //Se define posiciones donde imprimir en la lcd
   lcd.print(" 2 = RFID_tag ");//Se Muestra texto en la lcd
   lcd.clear(); // Se limpia LCD
   lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
   lcd.print(" 3 = REGRESAR  ");//Se Muestra texto en la lcd

   delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

   if (Estado == '1') {                // Comparamos la variable Estado, si Estado == a "1", llama la
funcion balanza_tara

        Calibrar_Balanza();
         Estado = 0;
      }

      if (Estado == '2')              // Comparamos la variable Estado, si Estado == a "2", llama la
funcion rfid_tag
      {
        Configurar_rfid_tag();
        Estado = 0;
      }
       if (Estado == '3')             // Comparamos la variable Estado, si Estado == a "3", llama la
funcion retornar a void loop
      {
        return;

      }

}

void Operacion(){

   // Proceso 1: detectar presencia del cerdo
       rfid_tag();
```

```
   // proceso 2: pesar y tomar decisión de la cantidad de alimento
      Pesar();
   // grabar el peso



    return;                    // Regresar a void loop
}
void Calibrar_Balanza() {
/*-------------------------------------------------------------------------------------------------------------
--
 *  Este programa permite calibrar la báscula. Se utiliza para determinar el factor de calibración.
 * También puede servir para generar el factor cero útil en básculas que tienen una masa
permanente.
 * -------------------------------------------------------------------------------------------------------------
--
 * El programa base: https://github.com/sparkfun/HX711-Load-Cell-
Amplifier/blob/master/firmware/SparkFun_HX711_Calibration/SparkFun_HX711_Calibration.i
no
 * Está sujeto a la licencia Beerware lo que viene a decir que el código es de dominio público.
 *
 * Para empezar la configuración de la báscula tienes que iniciar este programa sin tener ningún
peso encima de la báscula y conectar con PC.
 * Una vez empiecen a aparecer medidas de peso en el monitor serie, coloca algún objeto con un
peso conocido sobre la báscula.
 * Luego, en el monitor serie pon + o - (en símbolo) para ajustar el factor de calibración
(factor_calibracion) hasta que la salida conincida
 * con el peso conocido del objeto.
 *
 * Por ejemplo, puedes poner un kilo de arroz o un kilo de naranjas. Mi recomendación es que si
tienes otro peso en casa primero lo peses en ese peso y
 * así sabrás con exactitud cuanto pesa.
 *
 * Una vez el peso coincida apunta el valor del factor de calibración (factor_calibracion).
 *
*/
    Serial.println("HX711 programa de calibracion");
    Serial.println("Quita cualquier peso de la bascula");
    Serial.println("Una vez empiece a mostrar informacion de medidas, coloca un peso conocido
encima de la bascula");
    Serial.println("Presiona + para incrementar el factor de calibracion");
    Serial.println("Presiona - para disminuir el factor de calibracion");


    bascula.set_scale();               // Aplicar la calibración. La escala por defecto es 1
```

```
    bascula.tare(20);                   // Iniciar la tara. El peso actual es considerado Tara. O no tiene que haber nada sobre el peso. Promedio de 20 lecturas

    long zero_factor = bascula.read_average();  // Obtener una lectura de referencia

    Serial.print("Zero factor: ");        // Mostrar la primera desviación
    Serial.println(zero_factor);

    Serial.print("Leyendo: ");              // Mostrar la información para ajustar el factor de calibración
    Serial.print(bascula.get_units(), 1);   // Se obtiene el valor necesario para calcular la ESCALA
    Serial.print(" kgs");
    Serial.print(" factor_calibracion: ");
    Serial.print(factor_calibracion);
    Serial.println();

  if (Serial.available())                   // Obtener información desde el monitor serie y calibrar
  {
    char temp = Serial.read();
    if (temp == '+')
      factor_calibracion += 10;
    else if (temp == '-')
      factor_calibracion -= 10;
  }
    lcd.clear(); // Se limpia LCD
    lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
    lcd.print(" Factor Calibración:  ");
    lcd.setCursor(0,1); //Se define posiciones donde imprimir en la lcd
    lcd.print(factor_calibracion);

    return factor_calibracion;

}

void Configurar_rfid_tag() {
/*-----------------------------------------------------------------------------------------------------------
--
 * leer datos de un PICC (es decir, una etiqueta o tarjeta RFID) utilizando un lector RFID basado en MFRC522.
 * Lector en la interfaz SPI de Arduino.
 * -----------------------------------------------------------------------------------------------------------
--
 */

    lcd.clear(); // Se limpia LCD
```

```
   lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
   lcd.print(" 1 = LEER TARJETA ");//Se Muestra texto en la lcd
   lcd.setCursor(0,1); //Se define posiciones donde imprimir en la lcd
   lcd.print(" 2 = ESCRIBIR _DATO");//Se Muestra texto en la lcd

   delay(2000); // Retardo de 1 segundo mostrando el texto en la lcd

   lcd.clear(); // Se limpia LCD
   lcd.setCursor(0,0); //Se define posiciones donde imprimir en la lcd
   lcd.print(" 3 = REGRESAR  ");//Se Muestra texto en la lcd

   if (Estado == '1') {                // Comparamos la variable Estado, si Estado == a "1", llama la
funcion balanza_tara

         rfid_Leer_tag();
          Estado = 0;
      }

      if (Estado == '2')              // Comparamos la variable Estado, si Estado == a "2", llama la
funcion rfid_tag
      {
        rfid_Escribir_tag();
        Estado = 0;
      }
       if (Estado == '3')             // Comparamos la variable Estado, si Estado == a "3", llama la
funcion retornar a void loop
      {
        return;

      }

}

void rfid_Leer_tag()
{
   lcd.setCursor(0, 0);

   MFRC522::MIFARE_Key key;
   for (byte i = 0; i < 6; i++) key.keyByte[i] = 0xFF;

   byte block;        //algunas variables que necesitamos
   byte len;
   MFRC522::StatusCode status;

   if ( mfrc522.PICC_IsNewCardPresent())     // Revisamos si hay nuevas tarjetas  presentes
      {
```

```
        if ( mfrc522.PICC_ReadCardSerial())    //Seleccionamos una tarjeta
      {
          // Enviamos serialemente su UID
          lcd.clear();
          lcd.print("Card UID");
          Serial.print("Card UID:");
          lcd.setCursor(0, 1);
          for (byte i = 0; i < mfrc522.uid.size; i++) {
                Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
                lcd.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
                Serial.print(mfrc522.uid.uidByte[i], HEX);
                lcd.print(mfrc522.uid.uidByte[i], HEX);
          }
          Serial.println();
          // Terminamos la lectura de la tarjeta  actual
          mfrc522.PICC_HaltA();
      }
      delay(2000);
      lcd.clear();
      lcd.setCursor (0, 0);
      lcd.print("Lectura RFID");
      lcd.setCursor (0, 1);
      lcd.print("Pasar Tarjeta");

  }
}




void rfid_Escribir_tag() {

 /*
 * ---------------------------------------------------------------------------
 * Función de muestra escribir bloques de datos en una MIFARE Classic PICC
 * (= tarjeta/etiqueta).
 * ---------------------------------------------------------------------------
 */

    // Preparar la clave - todas las teclas están ajustadas a FFFFFFFFFFh en la entrega del chip
desde la fábrica.
    MFRC522::MIFARE_Key key;
    for (byte i = 0; i < 6; i++) key.keyByte[i] = 0xFF;
    while ( ! mfrc522.PICC_IsNewCardPresent()) {          // Busque nuevas tarjetas
      //return;
    }
```

```
    // Seleccione una de las tarjetas
    while ( ! mfrc522.PICC_ReadCardSerial()) {
      //return;

    byte buffer[34];
    byte block;
    MFRC522::StatusCode status;
    byte len;

    Serial.setTimeout(20000L) ;     // esperar 20 segundos para la entrada de la serie

    lcd.clear();
    lcd.setCursor (0, 0);
    lcd.print(F("Codigo fin #"));       // Pregunte los datos personales: Número del cerdo
    len = Serial.readBytesUntil('#', (char *) buffer, 30) ;   // leer el nombre en la tarjeta, finaliza
con #
    for (byte i = len; i < 30; i++) buffer[i] = ' ';

    block = 1;
    //Serial.println(F("Autenticación mediante clave A..."));
    status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block,
&key, &(mfrc522.uid));
    if (status != MFRC522::STATUS_OK) {
      lcd.setCursor (0, 0);
      lcd.print(F("PCD_no valida:"));
      Serial.println(mfrc522.GetStatusCodeName(status));
      return;
    }
    else lcd.print(F("PCD valida: "));                //Imprimir desde la memoria Flash

    // Bloque de escritura
    status = mfrc522.MIFARE_Write(block, buffer, 16);
    if (status != MFRC522::STATUS_OK) {
      lcd.setCursor (0, 0);
      lcd.print(F("MIFARE_Write() falló: "));        //Imprimir desde la memoria Flash
      Serial.println(mfrc522.GetStatusCodeName(status));
      return;
    }
    else Serial.println(F("MIFARE_Write() éxito: "));
    Serial.println(" ");
    mfrc522.PICC_HaltA(); // Halt PICC
    mfrc522.PCD_StopCrypto1();  // Stop encryption on PCD
    }
}
```

```
void Pesar() {

   bascula.set_scale(factor_calibracion); // Establecemos la ESCALA calculada anteriormente
   bascula.tare(20); // El peso actual es considerado Tara.

   lcd.setCursor(0,0);
   lcd.print("BALANZA DIGITAL");
   lcd.setCursor(0,1);
   lcd.print("Listo para pesar");
   delay(5000);  // Esperamos 5 segundos para comenzar a pesar

   lcd.setCursor(0,0);
   lcd.print("BALANZA DIGITAL");
   lcd.setCursor(0,1);
   lcd.print("Peso: ");
   lcd.print(bascula.get_units(20),3); // Se obtiene el valor real del peso en Kg del elemento
   lcd.print(" kg  ");
}

void rfid_tag(){
/* Operación sobre la tag.
 *
 *
 */
}
```

Anexo 3 - Lecturas/Escrituras RFID

VOLCADO DE LA MEMORIA DE CADA LLAVERO Y CADA MEMORIA

16:48:11.994 -> Firmware Version: 0x92 = v2.0
16:48:11.994 -> Scan PICC to see UID, SAK, type, and data blocks...
16:48:15.492 -> Card UID: 47 B4 4B 49
16:48:15.492 -> Card SAK: 08
16:48:15.492 -> PICC type: MIFARE 1KB
16:48:20.768 ->

Tarjeta 1
16:48:29.074 -> Card UID: 39 A9 EE B2
16:48:29.074 -> Card SAK: 08
16:48:29.074 -> PICC type: MIFARE 1KB
16:48:29.074 -> Sector Block 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 AccessBits
16:48:29.165 -> 15 63 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:29.256 -> 62 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:29.348 -> 61 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:29.439 -> 60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:29.531 -> 14 59 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:29.578 -> 58 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:29.671 -> 57 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:29.761 -> 56 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:29.807 -> 13 55 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:29.900 -> 54 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:29.993 -> 53 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.085 -> 52 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.132 -> 12 51 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:30.270 -> 50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.317 -> 49 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.363 -> 48 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.499 -> 11 47 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:30.545 -> 46 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.637 -> 45 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.729 -> 44 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.823 -> 10 43 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:30.869 -> 42 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:30.962 -> 41 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.053 -> 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.100 -> 9 39 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:31.193 -> 38 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.285 -> 37 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.379 -> 36 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.425 -> 8 35 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:31.562 -> 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.608 -> 33 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.700 -> 32 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.792 -> 7 31 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:31.838 -> 30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:31.930 -> 29 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:32.022 -> 28 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:32.113 -> 6 27 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:32.161 -> 26 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]

```
16:48:32.252 ->       25   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:32.343 ->       24   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:32.388 ->   5   23   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:32.527 ->       22   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:32.572 ->       21   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:32.664 ->       20   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:32.754 ->   4   19   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:32.846 ->       18   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:32.892 ->       17   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:32.983 ->       16   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.074 ->   3   15   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:33.166 ->       14   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.257 ->       13   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.303 ->       12   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.395 ->   2   11   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:33.486 ->       10   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.534 ->        9   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.627 ->        8   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.719 ->   1    7   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:33.811 ->        6   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.858 ->        5   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:33.997 ->        4   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:34.043 ->   0    3   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:34.135 ->        2   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:34.229 ->        1   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:34.274 ->        0   39 A9 EE B2  CC 08 04 00  62 63 64 65  66 67 68 69  [ 0 0 0 ]

Llavero 1
16:48:15.492 -> Sector Block   0  1  2  3   4  5  6  7   8  9 10 11  12 13 14 15  AccessBits
16:48:15.628 ->  15    63   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:15.674 ->        62   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:15.767 ->        61   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:15.859 ->        60   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:15.953 ->  14    59   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:15.998 ->        58   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.091 ->        57   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.183 ->        56   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.229 ->  13    55   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:16.368 ->        54   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.413 ->        53   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.505 ->        52   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.597 ->  12    51   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:16.691 ->        50   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.738 ->        49   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.830 ->        48   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:16.922 ->  11    47   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:16.967 ->        46   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:17.060 ->        45   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:17.151 ->        44   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:17.241 ->  10    43   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:17.332 ->        42   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:17.378 ->        41   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:17.468 ->        40   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:17.557 ->   9    39   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:17.647 ->        38   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
```

```
16:48:17.692 ->       37   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:17.827 ->       36   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:17.873 ->   8   35   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:17.965 ->       34   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.056 ->       33   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.102 ->       32   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.196 ->   7   31   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:18.289 ->       30   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.382 ->       29   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.428 ->       28   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.521 ->   6   27   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:18.615 ->       26   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.661 ->       25   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.754 ->       24   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.846 ->   5   23   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:18.939 ->       22   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:18.985 ->       21   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.122 ->       20   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.168 ->   4   19   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:19.259 ->       18   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.350 ->       17   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.395 ->       16   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.485 ->   3   15   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:19.576 ->       14   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.666 ->       13   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.758 ->       12   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.849 ->   2   11   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:19.894 ->       10   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:19.987 ->        9   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:20.077 ->        8   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:20.122 ->   1    7   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:20.260 ->        6   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:20.306 ->        5   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:20.400 ->        4   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:20.493 ->   0    3   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:20.585 ->        2   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:20.630 ->        1   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:20.723 ->        0   47 B4 4B 49  F1 08 04 00  62 63 64 65  66 67 68 69  [ 0 0 0 ]

Llavero 2
16:48:41.359 -> Card UID: A5 F9 FE 7B
16:48:41.359 -> Card SAK: 08
16:48:41.359 -> PICC type: MIFARE 1KB
16:48:41.405 -> Sector Block   0  1  2  3   4  5  6  7   8  9 10 11  12 13 14 15  AccessBits
16:48:41.451 ->   15    63   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:41.587 ->       62   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:41.633 ->       61   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:41.724 ->       60   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:41.815 ->   14    59   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:41.861 ->       58   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:41.953 ->       57   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:42.047 ->       56   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:42.137 ->   13    55   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:42.182 ->       54   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:42.317 ->       53   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
```

```
16:48:42.362 ->       52   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:42.453 ->  12   51   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:42.544 ->       50   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:42.589 ->       49   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:42.681 ->       48   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:42.773 ->  11   47   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:42.866 ->       46   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:42.913 ->       45   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.005 ->       44   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.096 ->  10   43   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:43.187 ->       42   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.277 ->       41   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.323 ->       40   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.415 ->  9    39   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:43.509 ->       38   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.599 ->       37   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.647 ->       36   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.741 ->  8    35   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:43.833 ->       34   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:43.879 ->       33   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.016 ->       32   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.062 ->  7    31   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:44.155 ->       30   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.247 ->       29   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.292 ->       28   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.383 ->  6    27   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:44.474 ->       26   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.567 ->       25   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.613 ->       24   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.753 ->  5    23   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:44.798 ->       22   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.892 ->       21   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:44.984 ->       20   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.030 ->  4    19   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:45.122 ->       18   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.214 ->       17   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.306 ->       16   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.353 ->  3    15   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:45.445 ->       14   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.535 ->       13   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.582 ->       12   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.674 ->  2    11   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:45.765 ->       10   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.857 ->        9   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:45.949 ->        8   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:46.041 ->  1     7   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:46.086 ->        6   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:46.175 ->        5   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:46.265 ->        4   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:46.311 ->  0     3   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:46.448 ->        2   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:46.493 ->        1   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:46.585 ->        0   A5 F9 FE 7B  D9 08 04 00  62 63 64 65  66 67 68 69  [ 0 0 0 ]
```

Llavero 3

```
16:48:51.100 -> Card UID: 5A D8 EE 80
16:48:51.100 -> Card SAK: 08
16:48:51.145 -> PICC type: MIFARE 1KB
16:48:51.145 -> Sector Block   0  1  2  3   4  5  6  7   8  9 10 11  12 13 14 15  AccessBits
16:48:51.238 ->   15    63   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:51.331 ->         62   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:51.422 ->         61   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:51.469 ->         60   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:51.561 ->   14    59   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:51.653 ->         58   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:51.699 ->         57   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:51.791 ->         56   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:51.881 ->   13    55   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:51.974 ->         54   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.021 ->         53   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.159 ->         52   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.204 ->   12    51   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:52.295 ->         50   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.388 ->         49   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.434 ->         48   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.525 ->   11    47   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:52.617 ->         46   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.709 ->         45   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.754 ->         44   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:52.891 ->   10    43   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:52.936 ->         42   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.028 ->         41   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.120 ->         40   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.166 ->   9     39   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:53.257 ->         38   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.347 ->         37   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.440 ->         36   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.485 ->   8     35   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:53.623 ->         34   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.669 ->         33   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.760 ->         32   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.854 ->   7     31   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:53.900 ->         30   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:53.991 ->         29   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.085 ->         28   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.176 ->   6     27   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:54.222 ->         26   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.317 ->         25   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.409 ->         24   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.456 ->   5     23   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:54.548 ->         22   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.642 ->         21   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.736 ->         20   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.781 ->   4     19   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:54.920 ->         18   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:54.966 ->         17   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:55.055 ->         16   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:55.145 ->   3     15   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
16:48:55.190 ->         14   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
16:48:55.281 ->         13   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
```

16:48:55.373 -> 12 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:55.463 -> 2 11 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:55.557 -> 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:55.603 -> 9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:55.696 -> 8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:55.788 -> 1 7 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:55.879 -> 6 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:55.925 -> 5 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:56.017 -> 4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:56.108 -> 0 3 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [0 0 1]
16:48:56.199 -> 2 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:56.245 -> 1 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [0 0 0]
16:48:56.337 -> 0 5A D8 EE 80 EC 08 04 00 62 63 64 65 66 67 68 69 [0 0 0]

DATOS DE ESCRITURA Y LECTURA DEL SISTEMA RFID

LECTURAS TARJETAS ORIGINALES
16:52:33.899 -> 1. Escribir
16:52:33.899 -> 2. Leer
16:52:41.183 -> 2
16:52:41.965 -> **Card Detected:**
16:52:41.965 -> Card UID: 39 A9 EE B2
16:52:42.012 -> Card SAK: 08
16:52:42.012 -> PICC type: MIFARE 1KB
16:52:42.058 ->
16:52:43.067 -> 1. Escribir
16:52:43.067 -> 2. Leer
16:52:43.067 ->
16:52:43.067 ->
16:52:43.067 -> 1. Escribir
16:52:43.067 -> 2. Leer
16:52:52.262 -> 2
16:52:53.140 -> **Card Detected:**
16:52:53.140 -> Card UID: 47 B4 4B 49
16:52:53.140 -> Card SAK: 08
16:52:53.140 -> PICC type: MIFARE 1KB
16:52:53.187 ->
16:52:53.232 -> **End Reading**
16:52:53.232 ->
16:52:54.194 -> 1. Escribir
16:52:54.194 -> 2. Leer
16:52:54.194 ->
16:52:54.194 ->
16:52:54.194 -> 1. Escribir
16:52:54.239 -> 2. Leer
16:53:03.565 -> 2
16:53:04.434 -> **Card Detected:**
16:53:04.434 -> Card UID: A5 F9 FE 7B
16:53:04.479 -> Card SAK: 08
16:53:04.479 -> PICC type: MIFARE 1KB
16:53:04.525 ->
16:53:04.525 -> **End Reading**
16:53:04.525 ->
16:53:05.530 -> 1. Escribir
16:53:05.530 -> 2. Leer

16:53:05.530 ->
16:53:05.530 ->
16:53:05.530 -> 1. Escribir
16:53:05.530 -> 2. Leer
16:53:13.118 -> 2
16:53:15.543 -> **Card Detected:**
16:53:15.543 -> Card UID: 5A D8 EE 80
16:53:15.589 -> Card SAK: 08
16:53:15.589 -> PICC type: MIFARE 1KB
16:53:15.636 ->
16:53:15.636 -> **End Reading**

ESCRIBIR LLAVERO 1
16:53:16.644 -> 1. Escribir
16:53:16.644 -> 2. Leer
16:53:23.243 -> 1
16:53:43.771 -> Card UID: 47 B4 4B 49 PICC type: MIFARE 1KB
16:53:43.817 -> Entra codigo terminando con #
16:53:43.864 -> PCD_Authenticate() success:
16:53:43.864 -> MIFARE_Write() success:

LECTURA LLAVERO 1
16:53:43.911 -> 1. Escribir
16:53:43.911 -> 2. Leer
16:53:49.838 -> 2
16:53:50.802 -> **Card Detected:**
16:53:50.802 -> Card UID: 47 B4 4B 49
16:53:50.848 -> Card SAK: 08
16:53:50.848 -> PICC type: MIFARE 1KB
16:53:50.848 ->
16:53:50.848 -> CERDO 1
16:53:50.894 -> **End Reading**

ESCRIBIR LLAVERO 2
16:53:51.859 -> 1. Escribir
16:53:51.859 -> 2. Leer
16:54:05.878 -> 1
16:54:21.915 -> Card UID: A5 F9 FE 7B PICC type: MIFARE 1KB
16:54:21.961 -> Entra codigo terminando con #
16:54:22.007 -> PCD_Authenticate() success:
16:54:22.007 -> MIFARE_Write() success:

LECTURA LLAVERO 2
16:54:22.054 -> 1. Escribir
16:54:22.054 -> 2. Leer
16:54:25.095 -> 2
16:54:26.246 -> **Card Detected:**
16:54:26.246 -> Card UID: A5 F9 FE 7B
16:54:26.246 -> Card SAK: 08
16:54:26.246 -> PICC type: MIFARE 1KB
16:54:26.291 ->
16:54:26.291 -> CERDO 2
16:54:26.338 -> **End Reading**

ESCRIBIR LLAVERO 3

16:54:27.304 -> 1. Escribir
16:54:27.304 -> 2. Leer
16:54:27.304 ->
16:54:27.304 ->
16:54:27.304 -> 1. Escribir
16:54:27.349 -> 2. Leer
16:54:36.764 -> 1
16:54:47.803 -> Card UID: 5A D8 EE 80 PICC type: MIFARE 1KB
16:54:47.803 -> Entra codigo terminando con #
16:54:47.848 -> PCD_Authenticate() success:
16:54:47.895 -> MIFARE_Write() success:

LECTURA LLAVERO 3
16:54:47.940 -> 1. Escribir
16:54:47.940 -> 2. Leer
16:54:51.061 -> 2
16:54:52.032 -> **Card Detected:**
16:54:52.032 -> Card UID: 5A D8 EE 80
16:54:52.078 -> Card SAK: 08
16:54:52.078 -> PICC type: MIFARE 1KB
16:54:52.124 ->
16:54:52.124 -> CERDO 3
16:54:52.124 -> **End Reading**

ESCRIBIR TARJETA ADMINISTRADOR
16:54:53.135 -> 1. Escribir
16:54:53.135 -> 2. Leer
16:55:00.385 -> 1
16:55:18.655 -> Card UID: 39 A9 EE B2 PICC type: MIFARE 1KB
16:55:18.700 -> Entra codigo terminando con #
16:55:18.746 -> PCD_Authenticate() success:
16:55:18.746 -> MIFARE_Write() success:

LECTURA TARJETA ADMINISTRADOR
16:55:18.839 -> 1. Escribir
16:55:18.839 -> 2. Leer
16:55:23.530 -> 2
16:55:24.630 -> **Card Detected:**
16:55:24.630 -> Card UID: 39 A9 EE B2
16:55:24.675 -> Card SAK: 08
16:55:24.675 -> PICC type: MIFARE 1KB
16:55:24.722 ->
16:55:24.722 -> DANIA GONZALEZ
16:55:24.722 -> **End Reading**