PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

MEJORAMIENTO TECNOLÓGICO DE UNA BALANZA DINÁMICA BASADA EN UN SISTEMA EMBEBIDO

Tesis para optar el Título de Ingeniero Electrónico**,** que presenta el bachiller:

Luis Enrique Fernández Bardales

ASESOR: Ing. Gonzalo Cucho Padín

Lima, noviembre del 2013

RESUMEN

Actualmente, a nivel mundial el uso de sistemas embebidos tiene cada día mayor demanda. Claro ejemplo de ello se puede apreciar en los diferentes lugares en que se presenta: los móviles, los reproductores MP3, consolas de videojuegos, impresoras e incluso electrodomésticos como hornos microondas o lavadoras que contienen sistemas embebidos que facilitan su operación.

Un campo de aplicación de los sistemas embebidos, es el control y automatización de máquinas industriales. Éste es el caso de las balanzas dinámicas que se abordará en el desarrollo de la presente tesis. El objetivo del sistema embebido en una balanza dinámica es adquirir los datos, analizarlos y de acuerdo a ello realizar una acción. Además de estas características, el sistema embebido brinda la posibilidad de almacenar los datos mediante un gestor de base de datos y la modificación del diseño de la interfaz. Características que se desean implementar a una máquina que carece de ellos a manera de mejoramiento tecnológico.

Tomando en cuenta estas necesidades, el objetivo general de este trabajo de tesis es presentar las características importantes y fundamentales que se requiere para la selección del sistema embebido para una máquina industrial, en este caso una balanza dinámica. También dar una introducción del manejo de una herramienta de objetos gráficos para el diseño de una interfaz gráfica que es presentada a través de una pantalla táctil.

Luego de la etapa de implementación del sistema se obtuvieron dos resultados importantes. Se ha logrado el control de los principales componentes de una balanza dinámica mediante una interfaz gráfica, diseñado con una herramienta de objetos gráficos llamada QT. Finalmente, se consiguió el almacenamiento de los valores de peso a través de un gestor de base de datos llamado SQLite.

FACULTAD DE
CIENCIAS E
INGENIERÍA

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

TEMA DE TESIS PARA OPTAR EL TÍTULO DE INGENIERO ELECTRÓNICO

Título	: Mejoramiento tecnológico de una balanza dinámica basada en un sistema embebido.
Área	: Electrónica #1137
Asesor	: Gonzalo Cucho
Alumno	: Luis Fernández Bardales
Código	: 20067224
Fecha	: 5 de mayo del 2013

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
Facultad de Ciencias e Ingeniería
0 7 JUL. 2013
DANIEL TORREALVA DAVILA
DECANO

Descripción y Objetivos

Una balanza industrial es una máquina automática que monitorea los valores de los pesos de diversos productos que son trasportados a través de una línea de producción. Frecuentemente se ubica al final de esta linea y clasifica los productos de acuerdo a sus respectivos pesos.

Le empresa SISCODE S.A. cuenta con una balanza dinámica, la cual se encuentra en buen estado y operativa. Esta balanza esta compuesta por una celda de carga para medir los pesos y envía esta data mediante el protocolo RS232 a un controlador digital, y un variador de frecuencia para controlar la velocidad del motor de la faja. Sin embargo al usar la máquina se presentan ciertos inconvenientes que se desea mejorar. (1) el modo de acceso a los datos almacenados durante el proceso de pesado, ya que la única manera de adquirir estos valores es imprimiéndolos y (2) la manera en que estos son presentados al operador. Por ello el desarrollo de un sistema de control y automatización basado en un sistema embebido se presenta como una alternativa eficiente y de bajo costo para solucionar estos inconvenientes

De este modo los objetivos de la presente tesis son: realizar la selección del sistema embebido y su sistema operativo apartir de análisis de los detalles técnicos de estos, analizar las señales de lectura de sensor de peso y la de control del variador para adaptarlos al sistema embebido, investigar y utilizar una base de datos para sistemas embebidos y realizar el diseño e implementación de una interfaz gráfica de usuario

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
SECCIÓN ELECTRICIDAD Y ELECTRÓNICA
Dr. Ing. BENJAMÍN CASTAÑEDA APHAN
Coordinador de la Especialidad de Ingeniería Electrónica

MÁXIMO 50 PÁGINAS

FACULTAD DE
CIENCIAS E
INGENIERÍA

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

TEMA DE TESIS PARA OPTAR EL TÍTULO DE INGENIERO ELECTRÓNICO

Título : Mejoramiento tecnológico de una balanza dinámica basada en un sistema embebido.

Índice

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
SECCIÓN ELECTRICIDAD Y ELECTRÓNICA

Dr. Ing. BENJAMÍN CASTAÑEDA APHAN
Coordinador de la Especialidad de Ingeniería Electrónica

MÁXIMO 50 PÁGINAS

ÍNDICE

INTRODUCCIÓN

El presente trabajo de tesis tiene como objetivo mejorar ciertas características de una balanza dinámica industrial mediante el uso de un sistema embebido. En un principio se describe la problemática de la cual surge la necesidad de seleccionar un sistema embebido para lograr los objetivos y diseñar la interfaz gráfica para el control.

El segundo capítulo presenta conceptos generales sobre los diferentes tipos de sistemas embebidos existentes y los parámetros importantes para su elección. Además se presentan definiciones generales del sistema tales como el sistema operativo, la celda de carga, variadores de frecuencia, el entorno gráfico y el principio de funcionamiento de las pantallas táctiles.

En el tercer capítulo se realiza la selección mediante el análisis de las principales características de los elementos empleados. Del mismo modo, se describe el programa de la interfaz gráfica, sus librerías, sistema operativo empleado y los módulos utilizados para su diseño. El capítulo concluye con la descripción de los algoritmos utilizados para el programa, presentación de diagramas de flujo e imágenes utilizadas para el diseño de la interfaz.

Finalmente, en el cuarto capítulo se desarrollan las pruebas en base al sistema embebido seleccionado dentro de la balanza dinámica. Para ello se presentan imágenes de la balanza dinámica con las modificaciones realizadas y la terminal de acceso al sistema embebido que muestra la información interna para probar el funcionamiento de la base de datos y el programa en general.

CAPÍTULO 1

PROBLEMÁTICA ACTUAL: BALANZA DINÁMICA DE LA EMPRESA SISCODE S.A.

1.1. Balanza dinámica de la empresa SISCODE S.A.

1.1.1. Empresa SISCODE S.A.

La empresa SISCODE S.A. fue fundada en el año 1995. Su visión es de brindar soluciones integrales para procesos industriales de fin de línea. Se inició como representante de ventas de codificadoras industriales en el Perú, para luego convertirse en una empresa especializada, además de las codificadoras, en empaque, envasado, transporte, control de calidad y cerrado de bolsas.
El progresivo crecimiento de la empresa ha dado como resultado la creación de nuevas áreas, como es el caso del área de proyectos. Esta área es la encargada de todo lo referente a electrónica y mecánica. Unos de sus proyectos es el origen de la problemática del presente documento y, está relacionada con una balanza dinámica que se menciona con mayor profundidad en los siguientes capítulos.

1.1.2. Balanza dinámica

La Empresa SISCODE S.A. cuenta con una balanza dinámica, una maquina automática que monitorea los valores de pesos de diversos productos que son transportados a través de una línea o faja. El modelo de la balanza es CWC-230MS de la marca Rehoo. La principal razón de la compra de la balanza fue para su venta. La balanza cuenta con las siguientes características [1]:

- Posee una estructura de acero inoxidable.
- Tiene una celda de carga.
- Tiene predefinidos 100 productos en su sistema. Estos son características que permiten obtener con mayor precisión los valores de los pesos dependiendo el tipo del producto.
- Posee una interfaz gráfica de la cual se puede acceder por una pantalla táctil de 7 pulgadas.
- Es posible modificar la velocidad del motor de la faja mediante la interfaz que posee.

Figura 1.1: Balanza dinámica modelo CWC-230MS

1.1.3. Declaración de la problemática

La balanza dinámica, es un dispositivo diseñado para medir y registrar pesos por eje y pesos por línea completa. A diferencia de las balanzas estáticas los sistemas con balanzas dinámicas son capaces de medir sin necesidad de interrumpir el flujo de tránsito y no requieren que la faja se detenga por completo. Las balanzas dinámicas son utilizadas para detectar violaciones a los tamaños y pesos de la línea permitidas por las normas, y eventualmente la aplicación de multas.

La balanza dinámica modelo CWC-230MS, se encuentra en buen estado y operable. Sin embargo al usar la máquina se presentan ciertos inconvenientes con respecto a la manipulación de los valores de peso que se monitorean. Principalmente en dos aspectos: (1) el modo de acceso a los datos que monitorea la balanza y (2) la interfaz gráfica. Tales inconvenientes pretenden ser solucionados de tal manera que el costo de la balanza con la mejora no sea elevada. Por ello el desarrollo de un sistema de control de automatización con un sistema embebido se presenta como una alternativa eficiente y barata para solucionar estos inconvenientes.

El control de pesaje de los productos es un proceso esencial en la industria, por lo que un ineficiente control de esta significará problemas económicos para la empresa. La mayoría de empresas en el Perú, que poseen alto índice de productividad, no poseen un sistema adecuado de pesaje de sus productos, por lo que se generan incremento de los costos.

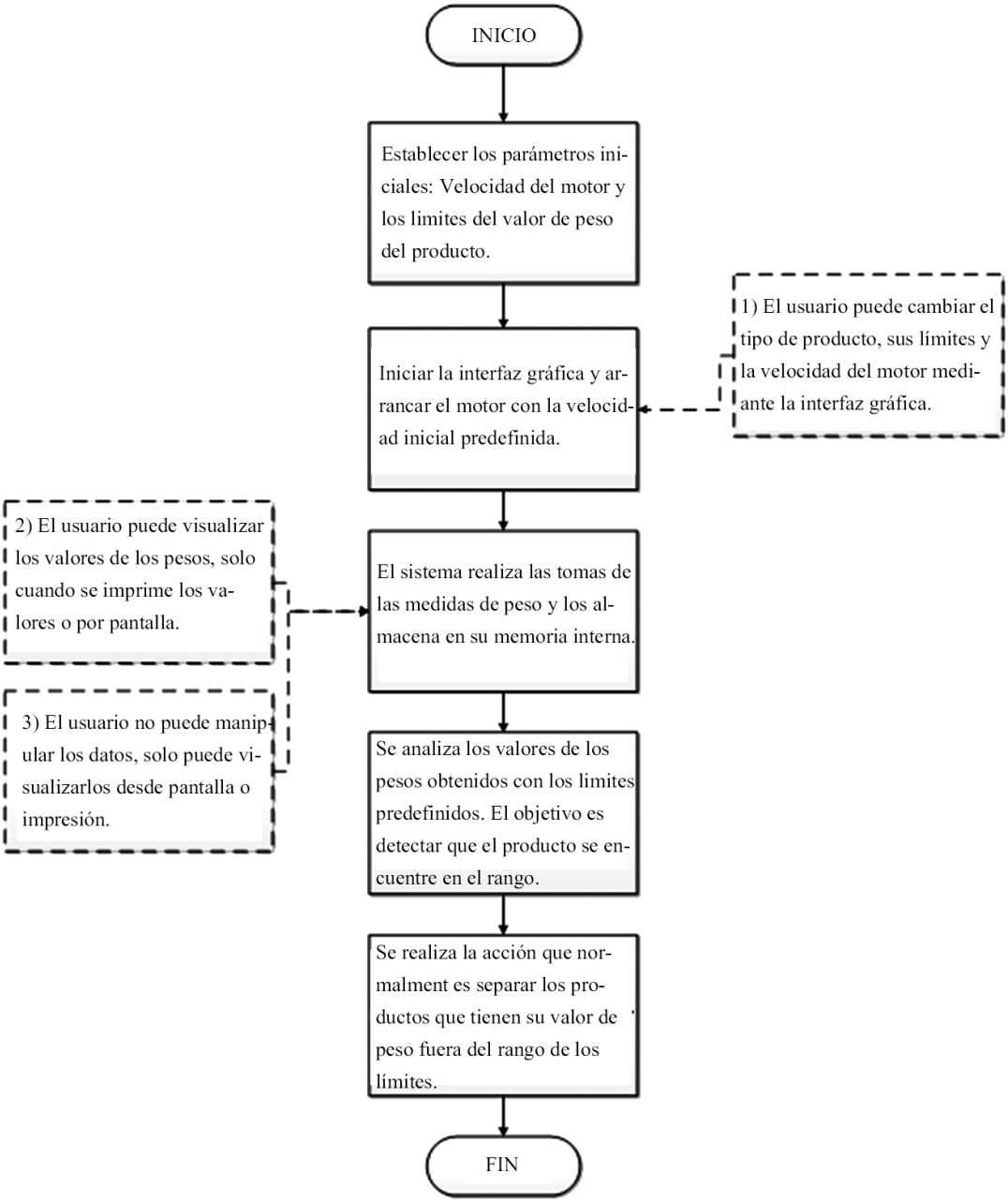

Figura 1.2: Diagrama de proceso de la balanza dinámica

En la figura 1.2 se muestra el diagrama del proceso de la balanza dinámica, en el cual se indica la manera en que las variables son manipuladas y la manera en que el usuario accede a los datos. Siendo los puntos 1, 2 y 3 los puntos que se desean mejorar.

El primer punto, es la posibilidad de poder cambiar el tipo de producto, sus límites y la velocidad del motor que controla la faja. La balanza CWC-230MS posee un sistema con esas opciones y se deseó que el nuevo sistema mantenga estas. El tipo de producto se refiere a las características de un tipo en particular de producto que se desea monitorear, harina por ejemplo. Dentro de cada tipo de producto se guardan los límites de peso, límites en que se debería encontrar el peso del tipo de producto. En caso de no encontrarse dentro del rango el sistema envía una señal de alerta para que pueda ser separada de forma mecánica.

El control de velocidad del motor también es un factor importante. La idea es utilizar el variador de frecuencia para modificar la velocidad del motor que controla el movimiento de la faja. Con el uso de la interfaz gráfica de la balanza dinámica es posible modificar la frecuencia a gusto del usuario.

El segundo punto es la posibilidad de poder visualizar los valores de peso a tiempo real. La balanza CWC-230MS no posee esta opción, solo se puede visualizar los valores de pesos cuando se han acumulado cierta cantidad de valores o al momento de imprimirlos.

El tercer punto es la posibilidad de poder manipular los datos almacenados en la memoria. El objetivo es tener la opción de guardar la data en una memoria externa. De esta manera, estos datos son enviados a un sistema principal donde son procesados o analizados.

Finalmente, existen varias formas de medir el peso. Las formas manuales que requieren instrumentos manipulados por personal adicional toman mucho más tiempo en realizarse, y a su vez no garantizan una medición precisa, retrasando así un proceso que debería ser rápido y eficiente. Asimismo, las actividades del proceso se tornan laboriosas, lo que origina fatiga y estrés en los trabajadores y descenso en su productividad.

1.1.4. Síntesis sobre el asunto de estudio

En la actualidad, es común la preocupación de mejorar el nivel de control y monitoreo de los productos en las industrias, para la reducción en la utilización de materiales, y de los costos. Asimismo, la actividad de medición y cálculo manual del peso de objetos y posteriormente a la introducción a una base de datos abarca mucho tiempo, es repetitiva, tediosa y, además, contiene errores. Ante estas necesidades se han creado dispositivos de medición, cuyo desarrollo ha llevado al uso de nuevas tecnologías, entre las cuales tenemos las balanzas dinámicas. De esta manera, la automatización del proceso de medición permite obtener los pesos del objeto de manera más eficiente y confiable.

Entre las Variables de identificación y clasificación en el medio general están la tendencia tecnológica, demanda y tendencia mundial debido a que el uso de sistemas de basados en sistemas embebidos es común alrededor del mundo y es muy solicitado.
Entre las variables de identificación y clasificación en el medio específico están los usuarios que normalmente son empresas industriales que requieren el control de sus productos en la última línea y los proveedores.
Y por último entre las variables de identificación y clasificación en el medio organizacional está el modo en que se ejecuta el sistema, los diferentes algoritmos y sus componentes que la conforman.

CAPÍTULO 2

CONCEPTUALIZACIONES GENERALES

2.1. Estado de Arte

2.1.1. Presentación del asunto de estudio.

El objetivo de este capítulo es comprender el funcionamiento de la balanza dinámica. Para ello, se requiere profundizar los conceptos de los componentes que la conforman, para luego proceder a las modificaciones. Primero se describirá a la balanza dinámica en general, luego se hará lo propio para los sistemas embebidos disponibles en el mercado y finalmente se abordaran los conceptos relacionados al sensor de peso y el variador de frecuencia.

2.1.1.1. Balanza Dinámica

Una balanza dinámica es una máquina industrial automática que monitorea los valores de los pesos de diversos productos que son trasportados a través de una línea de producción. Frecuentemente se ubica al final de esta línea y clasifica los productos de acuerdo a sus respectivos pesos.

Son diversos los usos que pueden ser asignados a una balanza dinámica. Por ejemplo: el monitoreo de los valores de peso muy bajos y muy altos de los productos, la verificación de los valores de pesos de los productos estén de acuerdo a alguna norma establecida o un rango de algún estándar establecido, la detección de algún componente extraviado dentro de una caja de productos, la clasificación de los productos de acuerdo a su peso, entre otros.

La posición y dimensiones del objeto o producto medidos por la balanza dinámica son características importantes que determinan la eficiencia de ella. El producto ideal es aquel que tiene sus dimensiones uniformes, que no vibra o cambia de posición durante su medición. Cuanto más inestable es la ubicación del producto que pasa sobre el sensor de peso de la balanza, mayor tiempo se requerirá para realizar la medida.

2.1.1.2 Dispositivo Embebido

Existen diferentes definiciones de un dispositivo embebido o sistema embebido. El autor Daniel W. Lewis define a un sistema embebido en su libro [2] como un dispositivo electrónico que tiene incorporado un microprocesador. El propósito del microprocesador es de simplificar el diseño de la aplicación en el sistema y de esta manera brindar flexibilidad; es decir si se desea eliminar algún error, agregar alguna característica o realizar una modificación solo sería por medio del software del dispositivo.

Según el autor Christopher Hallinan [3], un sistema embebido está relacionado con las siguientes características:

- Contiene un motor de procesamiento, tal como un microprocesador.
- Está diseñada para una aplicación específica o un propósito particular.
- Incluye una interfaz de usuario para el control.
- Normalmente sus recursos son limitados. Por ejemplo, el dispositivo puede no tener disco duro o puede requerir ciertas condiciones para operar con su batería.

Con estas dos definiciones se podría resaltar que la definición de un sistema embebido marca distancia con la definición de una computadora de escritorio. Un sistema embebido no es una computadora de escritorio en pequeñas dimensiones, esta clasificación ya existe y se les denomina dispositivos SFF (Small Form Factor) y tienen otro propósito en el mercado. Los dispositivos SFF tienen el mismo rendimiento que sus equivalentes en computadoras de escritorio pero en menor espacio. Las tarjetas embebidas microITX, picoITX y miniATX [22] son algunos ejemplos de placa principales para computadoras SFF. Existen sistemas embebidos especiales como es el PC/104 consortium. El PC/104 es un sistema embebido que contiene un software y hardware compatible con un bus especifico que puede ser apilado por otros dispositivos PC/104 como una torre de bloques como se puede apreciar en la figura 2.1.

Figura 2.1: Dispositivos PC/104 empilados como una torre de bloques.

La estructura de un sistema embebido usualmente está conformada por los siguientes componentes: el microprocesador, la memoria Flash para el programa y almacenamiento de datos, la memoria RAM dinámica síncrona (SDRAM), reloj de tiempo real, comunicación RS232, comunicación vía USB y Ethernet [3]. Componentes que se pueden apreciar en la figura 2.2.

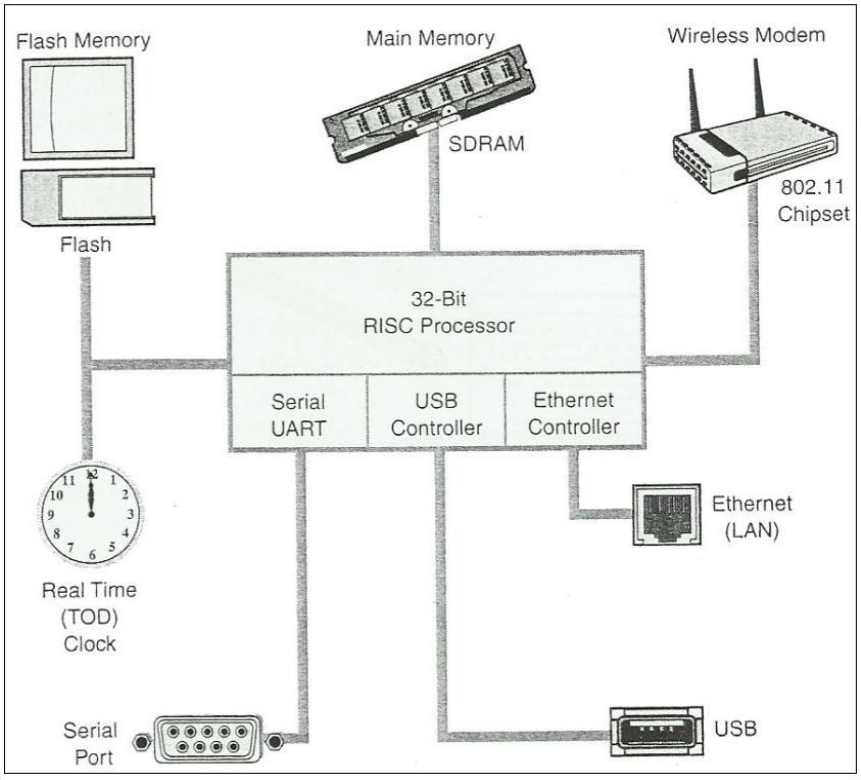

Figura 2.2: Componentes que conforman un sistema embebido. [3]

2.1.2. Estado de investigación.

2.1.2.1. Balanzas Dinámicas

En la tabla 2.1 se muestran algunos equipos comerciales y sus especificaciones.

Tabla 2.1: Balanzas dinámicas comerciales.

Equipo	Fabricante	Especificaciones	Imagen
CWC230	Rehoo	- Pantalla táctil con control de velocidad - Celda de carga HBM (Marca reconocida de venta de sensores) [24]	 **Figura 2.3:** Balanza dinámica CWC230 [1]
CK 2500	Lock Inspection Systems	- Memoria para hasta 100 productos - Control de velocidad - Celda de carga dinámica. - Puerto serial y de red.	 **Figura 2.4:** Balanza dinámica CK 2500. [4]
DACS-G-S015	Ishida	- Pantalla no táctil con botones de control. - Capacidad hasta 1.5 kilos. - Tiene puerto USB para acceder la data.	 **Figura 2.5:** Balanza DACS. [5]

2.1.2.2. Sistemas Embebidos

Para tener una noción clara de las especificaciones de los sistemas embebidos, definiremos los siguientes parámetros técnicos:

a) Frecuencia de reloj: Se refiere a la frecuencia en la cual los procesos del CPU se ejecutan. Su unidad es en Hertz. Es un parámetro muy importante para los sistemas embebidos ya que nos da una idea de cuánto tardaría en ejecutarse una instrucción del programa en ella.

b) Memoria RAM: Para los dispositivos embebidos a igual que las computadoras de escritorio, la memoria RAM es la memoria volátil. .Es un parámetro muy importante ya que nos brinda una idea de cuanta información puede soportar el dispositivo embebido mientras se ejecuta el programa, especialmente para la elaboración de la interfaz gráfica ya que la emisión de imágenes, animaciones y colores son los procesos que consumen más memoria. La memoria RAM dinámica síncrona (SDRAM) es el tipo de memoria que normalmente los sistemas embebidos utilizan.

c) Memoria de almacenamiento: La memoria Flash es el tipo de memoria no volátil que se utiliza en los dispositivos embebidos como medio de almacenamiento. Son dos tipos de memorias Flash que predominan, estas son la memoria Flash tipo NOR y la memoria Flash tipo NAND. La memoria tipo NAND es la más rápida en cuestión de lectura y escritura y también la de menor costo. La memoria tipo NOR tiene una gran ventaja frente a la de tipo NAND por su acceso aleatorio (Random Access) [6]. Resultando de este modo la memoria NOR adecuado para el almacenamiento de software y la memoria NAND para almacenamiento masivo de datos [7].

Tabla 2.2: Comparación entre memoria NOR y NAND. [25]

Propiedad	**NOR**	**NAND**
Precio	4x/MB	x/MB
Random Access	Si	No
Tiempo de escritura	10us/palabra	200us/página
Tiempo de lectura	100ns/palabra	50us/página

d) Entrada y salidas de propósito general (GPIO): Es usual que los sistemas embebidos tengan un grupo de pines de salida y entrada. La configuración de los pines se realiza mediante el programa que se utiliza y con ayuda de un módulo el control de GPIO. La cantidad de pines depende de cada dispositivo embebido.

En la siguiente tabla se muestran algunos sistemas embebidos comerciales.

Tabla 2.3: Sistemas embebidos comerciales.

Dispositivo	Procesador	Especificaciones	Imagen
Arduino Due	AT91SAM3X8E	- CPU: 85 MHz - RAM: 2KB - Máxima memoria: 512kb - GPIO:54	 **Figura 2.6:** Arduino. [8]
Mini2440	ARM9	- CPU: 533 MHz - RAM: 64MB - Máxima memoria: 32 GB - GPIO:34 -Soporta: Windows CE y Linux.	 **Figura 2.7:** Mini2440. [9]
Raspberry Pi	ARM11	- CPU: 700 MHz - RAM: 512MB - Máxima memoria: 32 GB - GPIO: 26 -Soporta: Linux	 **Figura 2.8:** Raspberry. [10]

PCduino	ARM Cortex-A8	-CPU: 1Ghz -RAM: 1GB -Máxima memoria 32GB. -GPIO: 9 -Soporta: Linux	 **Figura 2.9:** PCduino [26]
Cubieboard	ARM-Cortex-A8	- CPU: 1 GHz - RAM: 1 GB - Máxima memoria: 32 GB - GPIO: 96 -Soporta: Linux	 **Figura2.10:**Cubieboard[11]

2.2. Patente del funcionamiento de balanzas dinámica con celda de carga.

Para la investigación y desarrollo de la tesis se ha buscado diversos patentes con la finalidad de comprender el funcionamiento de una balanza dinámica. El estado de arte comprende libros, elementos comerciales y patentes. Siendo estos dos últimos los que proveen de información no tan técnica sino más bien los que nos brinda una idea de cómo los elementos electrónicos son agrupados para generar un nuevo producto. Información general para comprender el sistema.

2.2.1. Método de pesaje dinámico para la determinación de valor de carga y la resolución del mismo.

(Titulo original: DYNAMIC WEIGHING METHOD OF DETERMINING A LOAD MEASUREMENT VALUE AND THE RESOLUTION THEREOF): Método desarrollado por Lauri Holm, el cual consiste la integración de la señal generada por la carga que pasa sobre la plataforma de una balanza dinámica con ayuda de una celda de carga [12]. Según el trabajo de Lauri la señal que resulta

cuando la carga pasa sobre la celda de carga es una señal de peso que empieza desde un nivel básico y llega a un nivel pico para caer posteriormente al nivel básico como se puede observar en la figura 2.8. El valor del peso se calcula mediante el promedio de la sumatoria de los valores que se obtienen en el intervalo cercano al pico, es decir la integral de este intervalo.

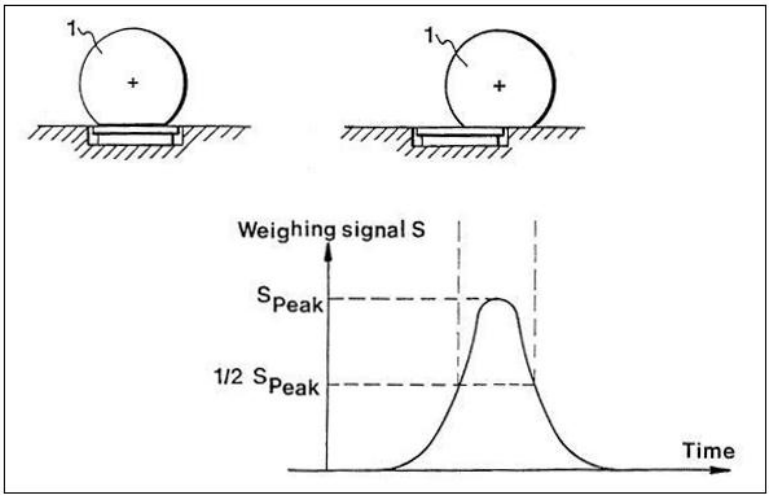

Figura 2.11: Imagen de la señal de peso vs tiempo. [12]

2.3. Conceptualizaciones generales.

A continuación, procedemos a definir ciertos conceptos que estarán implicados en el desarrollo de la presente investigación.

2.3.1. Arquitectura del procesador

El componente principal del sistema embebido es el microprocesador. Vale recalcar que se trata de un microprocesador y no de un Microcontrolador en la definición de un sistema embebido. Un microprocesador se basa solo en un CPU muy diferente es el caso de un Microcontrolador que, adicionalmente al CPU, contiene otros periféricos, RAM, flash etc. en su estructura interna. El Arduino es un claro ejemplo de un dispositivo que utiliza un Microcontrolador. Por ello, a pesar de que cumple con las demás características de un sistema embebido, algunos autores no lo consideran así.

Se puede clasificar a los procesadores en dos categorías, los procesadores independientes y los procesadores integrados. Los procesadores independientes

son los que se dedican exclusivamente al procesamiento. Los procesadores independientes tienden a ser los que tienen mayor rendimiento [13]. Estos procesadores requieren de circuitos adicionales para el soporte de su básico funcionamiento. Por ejemplo el procesador independiente IBM 970FX requiere de un chipset llamado 855GM para el control memoria y gráficos. La terminología usada para referir a estos circuitos adicionales es *northbridge*. Otros procesadores independientes son el Intel Pentium M, Intel Atom y el Freescale MPC7448. El Intel Atom tiene como característica su bajo consumo de potencia y compatibilidad con antiguos procesadores de 32 bits. EL procesador Freescale MPC7448 pertenece a la cuarta generación de arquitectura PowerPC o G4, este procesador es ampliamente usado en procesamiento de señales y redes.

Existen varias arquitecturas de procesadores, y cada una de estas tiene su ejemplar de procesadores integrados o también llamado SOC (system on chip) [23]. La arquitectura PowerPC está liderando en el mercado de los sistemas embebidos relacionados a telecomunicaciones. La arquitectura MIPS en los equipos de bajo consumo. Y la arquitectura ARM es muy usada en móviles.

2.3.1.1. Arquitectura ARM

ARM (Advanced RISC Machine) es una familia de procesadores mantenidas y promovidas por la compañía Británica ARM Holdings. A diferencia de las otras compañías productoras de procesadores, la compañía ARM Holdings no fabrica sus propios procesadores. ARM Holdings diseña los núcleos de sus procesadores y permite a sus clientes fabricar los chips con los procesadores diseñados como ellos deseen.

ARM ha tenido una gran acogida en el mercado de dispositivos electrónicos embebidos, claro ejemplo en la diversidad de dispositivos que lo usan tales como el PSP de Sony PlayStation Portable, Apple, Iphone, BlackBerry Storm, Tom Tom Go 300 GPS los mobiles de Motorola Droid.

2.3.2. Celda de Carga

El elemento fundamental para cualquier sistema de pesaje es el sensor de peso, denominada celda de carga. La celda de carga es un transductor que es utilizado para transformar una fuerza a una señal eléctrica. Es un dispositivo electromecánico, basado en la deformación mecánica. Este presenta diferentes configuraciones geométricas y pueden detectar rangos desde gramos hasta toneladas. Normalmente son de acero o aluminio.

Las celdas de cargas se pueden clasificar de acuerdo al tipo de señal que generan o al método que se utiliza para detectar el peso. Las más conocidas son las hidráulicas, las neumáticas y las basadas en galgas extensiométricas.

2.3.2.1. Celda de carga tipo hidráulico.

Las celdas de carga tipo hidráulico usan un fluido para transmitir la fuerza aplicada a la balanza y transformarla a una señal eléctrica. A medida que la fuerza crece, la presión del fluido hidráulico crece. Su precisión puede estar dentro del 0.25%, lo cual es aceptable para la mayoría de las aplicaciones con balanza dinámicas. Debido a que el sensor no tiene componentes electrónicos es ideal para zonas o regiones abiertas y expuestas a elementos que puedan afectar su precisión; es decir son inmunes a problemas eléctricos como rayos, soldaduras o humedad.

2.3.2.2. Celda de carga tipo Neumática

Las celdas de carga Neumáticas utilizan varias cámaras de amortiguación para proporcionar mayor precisión que una celda de carga hidráulica. Se utilizan normalmente para medir pesos relativamente pequeños en industrias donde la prioridad es la limpieza y seguridad. Son inmunes al cambio de temperatura y no contienen líquidos que pudiese contaminar el proceso si se rompiese el diafragma. La desventaja de estos tipos de celdas son su baja velocidad de respuesta y la necesidad de aire o nitrógeno limpio y seco.

2.3.2.3. Celda de carga basadas en galgas extensiométricas

Las celdas de carga basadas en galgas extensiométricas convierten la fuerza actuante sobre ella en señales eléctricas medibles mediante las denominadas galgas extensiométricas. Una galga extensiométrica es un sensor que sirve para medir la deformación. La galga está unida a una viga o miembro estructural que se deforma cuando el peso se aplica. El esfuerzo que deforma a la celda produce una variación en su resistencia eléctrica y es lo que se mide.

Dentro de las celdas de carga basadas en las galgas extensiométricas también existe una variedad de tipos diferenciándose en tamaño, fuerza, modo que se realiza la señal y otros. Algunos tipos de estos se puede apreciar en la tabla 2.4.

Tabla 2.4: Características de diferentes tipos de celda de carga.

Tipo	Rango de peso	Precisión	Ventajas	Desventajas
celdas de cargas mecánicas				
Celdas de carga hidráulicas	Hasta 5000 toneladas	0,25%	Soporta altos impactos, no insensibles a la temperatura	Costosas, complejas
Celdas de carga neumáticas	Amplia	Alta	Intrínsecamente seguras, no contienen líquidos	Respuesta lenta, requiere aire limpio y seco
Celda de carga extensiométricas				
Celdas de carga de flexión de viga	5 a 2500 Kg	0,03%	Bajo costo, construcción sencilla	Extensómetros están expuestos, requieren una protección
Celdas de carga de cizallamiento	5 a 2500 Kg	0,03%	Rechazo de altas cargas, mejor sellado y protección	
Celdas de carga de depósitos	hasta 250 toneladas	0,05%	Sirven para movimientos de carga	Ninguna protección de carga horizontal
Celdas de carga de botón y arandela	0 a 25 toneladas / 0 a 100 Kg típico	1%	Pequeñas, económicas	Cargas deben centrarse, ningún movimiento de carga permitido
Celdas de carga Monoplato	4 a 1000 Kg		Pueden aceptar el peso fuera del centro	

De la lista de tipos de celdas de cargas mostradas en la tabla 2.4, el más utilizado es el del tipo Monoplato. Este tipo de celda puede ser utilizada en diversas aplicaciones de pesaje, su característica principal es la posibilidad de aceptar cargas fuera del centro. Son muy utilizadas en las balanzas y pequeñas plataformas con dimensiones no mayores a 800x800mm, su capacidad oscila entre los 4kg y 1000kg.

2.3.3. Variador de frecuencia

Un variador de frecuencia es un equipo destinado a modificar la frecuencia y, por tanto, la velocidad, de un motor de inducción asíncrono. Este genera una corriente alterna con la frecuencia y la tensión necesaria para accionar dicho motor de corriente alterna.

El principio del funcionamiento se basa en tres etapas. La primera etapa es la rectificación de la entrada alterna usando un puente rectificador. La segunda etapa consiste en filtrar o suavizar la señal mediante un banco de condensadores. La última etapa es el inversor, en esta etapa se toma la señal rectificada y filtrada y se le convierte a una señal alterna con tensión y frecuencia variable. Las etapas se pueden ver en la figura 2.9:

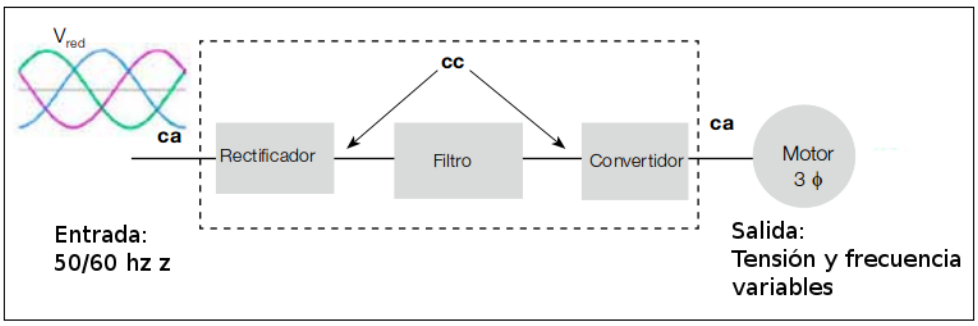

Figura 2.12: Diagrama de bloques del variador de frecuencia. [14]

2.3.4. Pantalla táctil

La balanza dinámica requiere de un mecanismo de interacción para que el usuario pueda comunicarse con el dispositivo. Esta característica clasifica al sistema como un sistema gestual [15]. Cada sistema de este tipo está conformado por tres partes: el sensor, el comparador y el actuador. Los tres pueden conformar un solo componente o varios. El comparador normalmente es el Microcontrolador o procesador del sistema, el actuador puede ser un motor u otro dispositivo. El sensor es el componente electrónico que se encarga de detectar los cambios del entorno. Estos cambios pueden ser de varios tipos, presión, sonido o iluminación etc. Es con el sensor que se define el tipo de un sistema gestual que se está trabajando.

Debido al gran uso de Smartphone, móviles y Tablet, las pantallas táctiles se han convertido en un medio de comunicación común de la persona con los

dispositivos. Pero no solo en dispositivos móviles, sino también en dispositivos industriales. Las pantallas táctiles eliminan la necesidad de periféricos adicionales o integrados, como el teclado o ratón. Una pantalla táctil es la combinación de sensor y comparador según la clasificación de un sistema gestual. Existen varias tecnologías de pantallas táctiles, siendo las más usadas la resistiva y la capacitiva.

2.3.4.1. Pantalla táctil resistiva.

La pantalla táctil resistiva está conformada por dos o más capas. Unas de las capas son de vidrio o acrílico que están envueltas con un conductor eléctrico y la otra capa, la opuesta, es de un material resistivo. El principio de funcionamiento de las pantallas táctiles de este tipo se basa que cuando el usuario o un objeto ejercen presión a una capa externa, las capas hacen contacto, provocando de esta manera un cambio de corriente. El programa recibe la posición de este punto y logra detectar la pulsación.

Las pantallas táctiles resistivas son más baratas, durables y soportan el polvo y el agua. Además de ser las más precisas puede usarse con cualquier material para la pulsación ya que su principio se basa en la presión que ejercen las capas. Entre sus inconvenientes se encuentra el uso de materiales filudos en ella provoca daños fatales, la imposibilidad de detectar varias pulsaciones en diferentes puntos al mismo tiempo y la reducción de brillo debido a la cantidad de capas que tiene. [16]

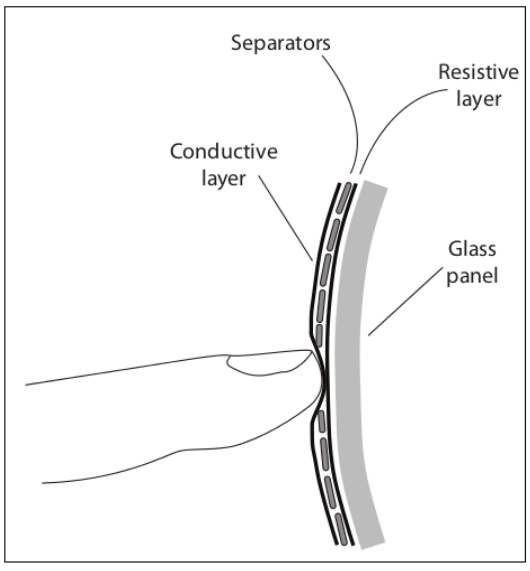

Figura 2.13: Funcionamiento de pantalla resistiva. [17]

2.3.4.2. Pantalla táctil capacitiva

Una pantalla táctil capacitiva utiliza un panel de vidrio que tiene una de sus caras recubierta por una capa de material conductor. Los dedos son parcialmente conductores, por ello cuando se hace contacto con la pantalla se induce una acumulación de carga en el área tocada. Esta carga acumulada es medida por circuitos en los bordes de la pantalla y luego esta información es enviada para su siguiente procedimiento.

La ventaja del funcionamiento de este tipo de pantalla táctil es que no altera la resolución de la imagen y tampoco limita el brillo. La desventaja es que, por su tecnología, las pantallas táctil tipo capacitivas dependen del contacto físico de la piel o materiales especiales para su funcionamiento. [17]

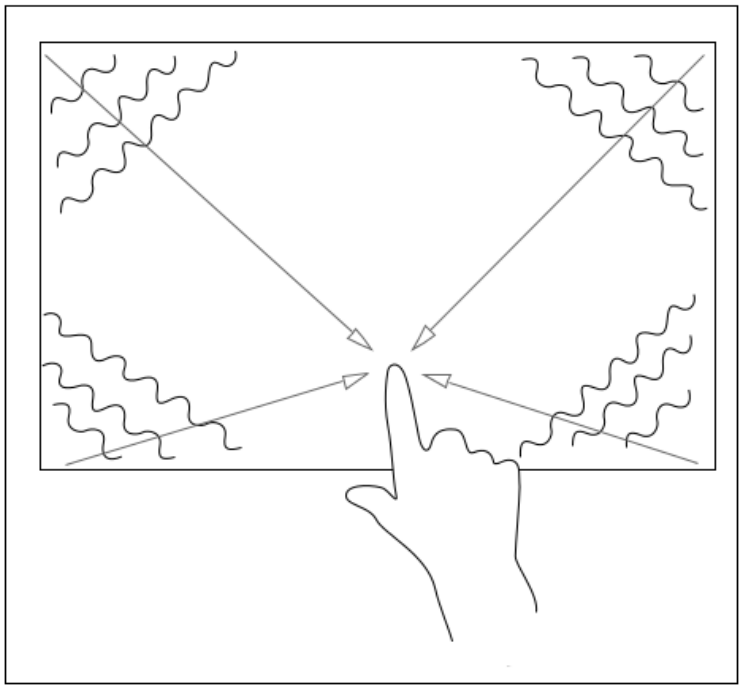

Figura 2.14: Funcionamiento de pantalla capacitiva. [17]

2.3.5. Sistema operativo embebido.

2.3.5.1. Windows embebido

Se le denomina Windows CE al sistema operativo de Microsoft enfocado para sistemas embebidos. Este sistema operativo soporta las cuatro principales arquitecturas de procesadores: Intel x86, ARM, MIPS y SuperH. El Windows CE es un sistema operativo de 32 bits y de tiempo real.

Para la construcción de la plataforma en donde se va ejecutar la aplicación, haciendo uso del sistema operativo Windows CE, se requiere de características como la arquitectura, el tipo de procesador y otros del sistema embebido que se va utilizar. Para esta construcción y para la creación de la aplicación se utiliza la herramienta Visual Studio.

La programación con Visual Studio se realiza mayormente en C++ con ayuda de las librerias CRT (C-Run Time), las planillas ATL (Active Template Library) y las clases MFC (Microsoft Fundation Class). La depuración puede realizarse con el depurador que viene por defecto en el Visual Studio y también se puede simular en este medio. Para lograr con éxito la instalación de la aplicación y el sistema operativo al sistema embebido se requiere del paquete BSP (Board Support Package) [18]. El BSP es prácticamente la información de las características del sistema embebido para que el programa Visual Studio pueda identificarlos, tales como el tipo de procesador, los drives y otros.

2.3.5.2. Linux embebido

A diferencia de Windows CE, se entiende como Linux embebido a la colección de diferentes distribuciones de Linux para sistemas embebido que existen tales como Montavista, WindRIver, Timesys, Denx y otros. Estas distribuciones de Linux están enfocadas para el uso del kernel en sistemas embebidos, existen distribuciones que no son exclusivamente para estos pero que tienen sus versiones que pueden ser soportadas. Un ejemplo de esto es la distribución Debian, esta distribución tiene una versión denominada Raspbian que está orientada directamente para su uso con el dispositivo embebido Raspeberry.

El sistema operativo Linux no es un sistema operativo de tiempo real, se requiere instalar software adicional para que soporte aplicaciones de tiempo real como por ejemplo RTLinux. Se puede usar diferentes lenguajes para la creación de

aplicaciones tales como C++, Java, Perl, Python, etc. Y soporta la mayoría de arquitecturas. Existe una gran comunidad que desarrollan de forma continua diferentes drivers y otros componentes para los proyectos en sistema embebidos con Linux [18].

La construcción del sistema operativo requiere de buen conocimiento del sistema embebido que se va utilizar. Normalmente su construcción es mediante la terminal y para la detección de errores se utiliza la herramienta GDB debugger.

2.3.6. Herramienta de Objetos Gráficos

Para la creación de entornos gráficos se utilizan las denominadas Herramientas de Objetos Gráficos (Widget toolkit). Las herramientas de Objetos Gráficos son librerías que contienen todos los elementos visuales para la construcción de la interfaz. Se le llama Objeto Gráfico al elemento o componente básico de la interfaz gráfica de usuario. Este componente muestra la información disponible de la aplicación al usuario, se puede considerar a estos como simples bloques visuales. Estos bloques contienen la información de la aplicación e interactúan con ella. Ejemplos de Objetos Gráficos son los menús, botones, barras, campos para entrada de texto, etc.

En la actualidad existen varias Herramientas de Objetos Gráficos tales como el GTK+, el Qt y el wxWidgets. Las dos últimas utilizan el lenguaje de programación C++ y el primero lenguaje de programación C. Los tres mencionados son los más populares y son libres, pueden utilizarse en sistemas operativos Linux, Mac OS, Windows.

2.3.6.1. GTK+

Es una herramienta de objetos gráficos. Su nombre proviene de las iniciales de la frase en inglés GIMP Toolkit. Donde GIMP es el nombre del programa que sirve para editar y manipular imágenes en GNU, el equivalente del Photoshop en Windows. GTK se originó como librería soporte para la creación de GIMP, luego de continuas mejoras hace que se independice bajo el nombre de GTK. Es unas de las más populares herramientas en sistemas Linux/Unix en conjunto con el QT.

GTK está desarrollado en C, pero aun así GTK sigue el modelo de programación orientado a objetos. El modelo se basa en asociar determinadas acciones, es decir los eventos, a determinadas operaciones o funciones. Los eventos definen un cambio de estado a determinado objeto, de manera que la aplicación puede ser informada del cambio.

Para que el modelo funcione correctamente, se usa lo que se llama el bucle de eventos o bucle principal. El bucle de eventos es un bucle interno del GTK+, en donde se comprueba los estados de cada uno de los elementos de la aplicación e informa tales cambios a los elementos que requieren de dicha información.

2.3.6.2. Plataforma QT (Cute)

Es otra herramienta de objetos gráficos. Está escrito en C++, sin embargo es posible utilizar Qt con otros lenguajes a través de conectores. El desarrollo de aplicaciones en Qt es muy apropiado para proyectos de gran escala, como por ejemplo el entorno de escritorio KDE o el reproductor multimedia VLC que fueron diseñados con esta herramienta.

Qt toma todas las características del lenguaje C++, todos los mecanismos del C++ han sido adoptados por esta biblioteca e incluso mejorados. De aquí radica la ventaja de Qt con otras herramientas, ya que al estar implementada directamente en C++, no requiere de un conector o recubrimiento C++ adicional. Por ello no se tiene alguna dificultad en adaptar los Widgets, como resultado la calidad en tiempo de ejecución es más alta, pues no se requiere usar referencias innecesarias y además, no depende de ninguna otra biblioteca, aparte de sí misma.

2.3.7. Base de datos

Una base de datos es un conjunto de información que por preferencia, están relacionadas entre sí y ordenadas. Está conformada por dos tipos de data: los datos y los metadatos. Los datos son los nombres, direcciones, puestos de trabajo de una empresa, etc. Los metadatos son la información de estos datos, es decir los campos de cada tabla, por los que se encuentran agrupados los datos. En el caso del ejemplo anterior los metadatos serian: nombre, dirección y puesto de trabajo.

La manera en que se organizan estos datos representa lo que se denomina el modelo de base de datos. Dos modelos de base de datos son: el modelo jerárquico y relacional. El modelo jerárquico consiste en una estructura tipo árbol invertido, cada tabla tiene una tabla padre. La desventaja del modelo jerárquico ocurre en el caso cuando se tiene una base de datos con gran cantidad de información, si se desea acceder a una tabla a partir de otra tabla con mismo nivel jerárquico se tiene que pasar por varias tablas padres para llegar a ella. El modelo relacional corrige esta desventaja, sin abandonar por completo la jerarquía de los datos. Esto hace que el modelo relacional pueda relacionar tablas sin importar el nivel jerárquico en que se encuentren.[Begining database disign by Gavin Powel]

Los programas que utilizan bases de datos con el modelo relacional se le denominan RDBMS (Relational Database Managment System). Siendo MySQL y SQLite unos de los más usados programas RDBMS del medio.

2.3.7.1. MySQL

MySQL es el RDBMS de código libre más utilizado en el mundo. MySQL utiliza el lenguaje de propósito especial diseñado para el manejo de información de una base de datos denominado SQL. Es muy utilizado en los servidores para páginas webs. Dichas páginas se les denominan páginas dinámicas y normalmente MySQL es usado en conjunto con el lenguaje PHP para el diseño de estas. MySQL puede almacenar varios tipos de datos desde un dato simple como un carácter hasta un dato extenso como una imagen y la cantidad de datos depende del Hardware.

MySQL está escrito en C y C++. Es multiplataforma, pues puede funcionar en Windows, Linux o Mac. Muchos de los lenguajes de programación existentes incluyen librerías para acceder a mysql de forma directa. Estos incluyen el "MySQL connector /NET" de visual studio de Microsoft, el "JDBC" de java, y "QDatabase" de Qt.

2.3.7.2. SQLite

SQLite es otra base de datos con modelo relacional muy utilizado en los sistemas embebidos. Del mismo modo que MySQL, SQLite utiliza el lenguaje SQL para la manipulación y creación de información de la base de datos. Pero a diferencia de MySQL, el lenguaje SQL de SQLite es una versión antigua, versión que carece funciones especiales.

No es muy robusto, su librería ocupa un espacio aproximado de 230 KB. Está escrita en lenguaje C. Es multiplataforma y es soportado por varios lenguajes tales como java, C++, php, python etc.

2.4. Propuesta.

El presente trabajo, propone un sistema embebido con arquitectura ARM como solución a la problemática planteada. El nuevo sistema embebido propuesto pretende remplazar al que se tenía incorporado en la balanza dinámica y se propone usar el sistema operativo Linux embebido por motivos de costo de licencia. Para poder manipular los valores de pesos almacenados se le va instalar al sistema un gestor de base de datos, con ella se pretende crear tablas y una base de datos para el control de los valores de peso adquiridos por la celda de carga. Se propone seleccionar una herramienta de objetos gráficos para la creación de la interfaz y el uso de un lenguaje de programación de objetos como C++ para construirla. Se propone un nuevo sensor de peso, es decir una nueva celda de carga, y utilizar el mismo variador de frecuencia que venía con la balanza dinámica para evitar mayor costo.

2.5. Objetivos

El objetivo general es:

- Realizar la selección del sistema embebido, sistema operativo y una herramienta de Objetos gráficos a partir de análisis de los detalles técnicos de estos

Los objetivos específicos son:

- Analizar las señales de lectura de sensor de peso y de control del variador para adaptarlos al sistema embebido
- Diseñar e implementar una interfaz gráfica de usuario.

CAPÍTULO 3
DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE CONTROL BASADO EN UN SISTEMA EMBEBIDO

En el segundo capítulo se explicó los conceptos fundamentales y se presentó los elementos más importantes que conforman una balanza dinámica. En esta nueva sección se presentan los componentes que se eligieron y los criterios tomados en cuenta para su elección. También se mostrara el diseño del programa y se hablara sobre los detalles más resaltantes de ella. El programa es una interfaz que controla la celda de carga y el variador de frecuencia.

3.1 Descripción del sistema

El diagrama de bloques del sistema de estabilización a desarrollar se muestra en la figura 3.1. Las partes fundamentales son el sistema embebido (unidad de control), la celda de carga (sensor), el variador de frecuencia (actuador) y la pantalla táctil (interfaz de entrada). Para iniciar el proceso de detección de pesos, el operador al encender la balanza, selecciona el producto que se desea monitorear por medio de la interfaz que se presenta en la pantalla táctil. Dentro del sistema embebido se encuentra almacenada una lista de productos, lista que puede ser accedida desde la interfaz. Es posible también crear un nuevo producto con la interfaz.

Una vez seleccionado el producto, se procede a elegir la velocidad del motor, para ello dentro de la interfaz se puede seleccionar la frecuencia de la velocidad. Para poder visualizar los valores de los pesos tomados también se hace uso una de las aplicaciones de la interfaz. Una vez ingresada a la aplicación se comienza a pedir valores de peso que detecta la celda de carga y lo muestra en la pantalla.

3.2 Hardware empleado

El variador de frecuencia es el único componente que se tomó de la Balanza Dinámica CWC-230MS. La celda de carga, el sistema embebido y la pantalla táctil son los componentes que se remplazaron sin embargo solo los dos últimos se seleccionaron debido a que la empresa ya había comprado una celda de carga

En esta sección primero se va a describir el variador de frecuencia para luego presentar la celda de carga, sistema embebido y la pantalla táctil que se usaron.

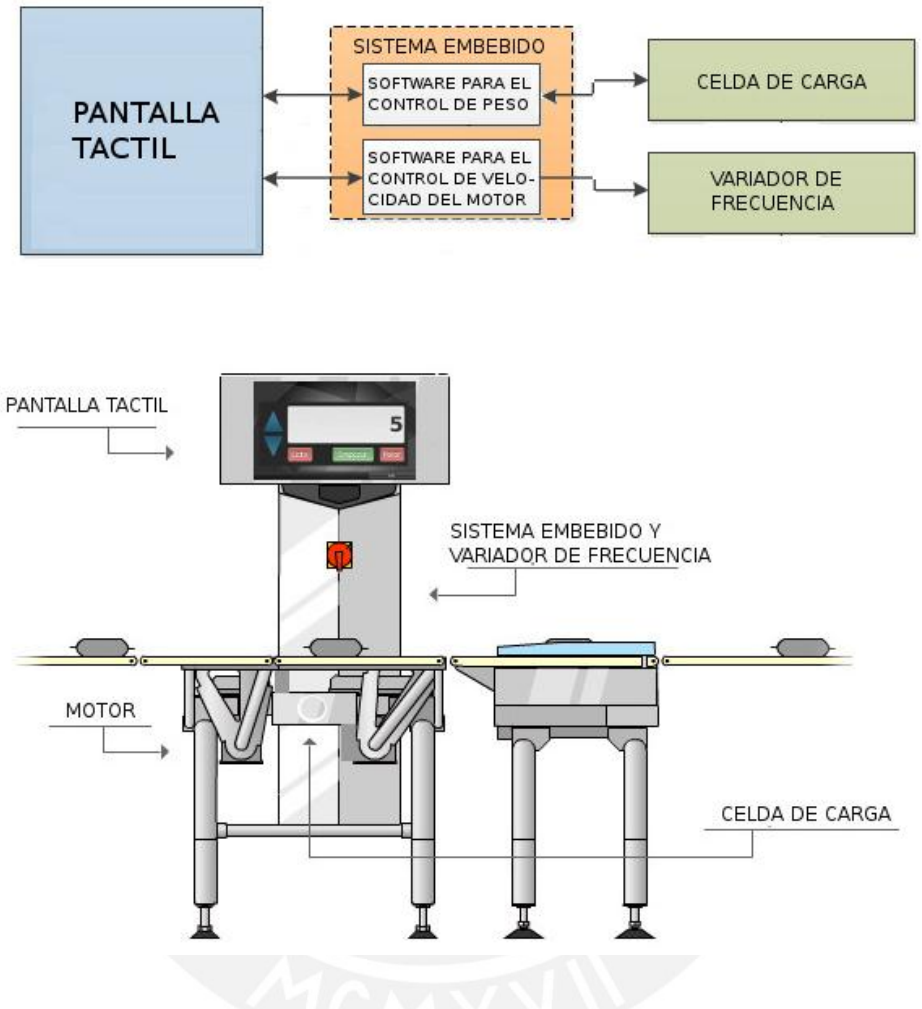

Figura 3.1: Diagrama de bloques del sistema y ubicación de los componentes.

3.2.1 Variador de frecuencia

La balanza CWC-230MS tiene un variador de frecuencia, este es el que se va utilizar en este trabajo para que el precio de la mejora que se realiza en la balanza no aumente. El modelo del variador es BVF00072GK de Panasonic. Este variador de 0.75 Kw que contiene su propio panel de control como se muestra en la figura 3.2 también permite modificar la velocidad del motor mediante una señal PWM de frecuencia 1 Hz. El ciclo de trabajo de la señal PWM es la que determina la frecuencia brindada al motor.

3.2.2 Celda de carga

El requerimiento fundamental para la celda de carga es que el grado de protección eléctrica y de impacto sea alto y que tenga un componente digital para facilitar el acceso a los datos mediante protocolos predefinidos.

Para este trabajo, la celda de carga que se uso es la celda modelo PW15AH, es una celda de carga tipo Monoplato que tiene el tipo de protección IP69K que significa que la celda de carga tiene máxima protección contra infiltración de polvo y agua. Es una celda Digital, debido a esto contiene diversos filtros y protocolos para acceder a los datos en ella. El tipo de comunicación de esta celda es RS485 con 1200, 2400, 4800, 19200, 38400, 57600 o 115200 baudios

3.2.3 Arquitectura del sistema embebido

En la actualidad, con relación a los sistemas embebidos, existen dos arquitecturas líderes en el mercado: la arquitectura ARM y el Intel x86. El precio, el consumo de energía, el soporte de los periféricos requeridos son las características que definen al tipo de procesador que se va utilizar. La gran diferencia entre los procesadores ARM y x86 es el diseño de sus arquitecturas, la arquitectura ARM utiliza instrucciones reducidas (RISC) por el otro lado el x86 utiliza instrucciones complejas (CISC).

La arquitectura CISC tiene largas instrucciones que permiten ejecutar ciertas instrucciones a solo un paso en comparación con la arquitectura RISC. Sin embargo las arquitecturas RISC tienen mayor control de sus instrucciones, esto llevo como resultado que Intel incorpore decodificadores a sus procesadores para convertir las instrucciones CISC a su equivalente RISC. Esta conversión requiere consumo de potencia, haciendo muy atractivos a los procesadores ARM por su bajo consumo de potencia en comparación con x86. Esta es una de las razones por lo que los procesadores ARM dominan el mercado de dispositivos electrónicos pequeños como los Smartphones o Tables. [29]

Características como rendimiento y fácil integración de periféricos, Intel es la mejor opción. Esta es la razón principal por lo que los procesadores de Intel dominan el mercado de computadoras de escritorio. Supóngase que se requiere analizar gran cantidad de datos y se requiere mayor velocidad de procesador se

puede remplazar el procesador Intel con otro procesador con mayor velocidad sin realizar otra acción adicional en el software. Esto no es posible con la arquitectura ARM. Cada procesador ARM tiene características propias, lo que significa que el software requiere ser adaptado.

Con respecto a costos la arquitectura ARM es la que tiene menor valor, la razón de esto está relacionada de forma directa con las dimensiones y características que contiene la arquitectura ARM, está diseñada en conjunto con otros fabricantes de chips, esto adiciona diferentes características por una determinada aplicación; es decir que se puede encontrar la tarjeta con las características que se requieren en un proyecto de sistema embebido sin tener que realizar ajustes al procesador.

Para la aplicación de la presente tesis el costo es el principal factor para la elección de arquitectura, se pretende que el costo de mejoramiento de la balanza sea lo mínimo posible. El gran tiempo que se va utilizar el dispositivo hace que el consumo de potencia también sea un factor importante a considerar. Por todos estos motivos se ha elegido la arquitectura ARM como la arquitectura ideal para el sistema embebido que se va a utilizar.

3.2.4 Selección del Sistema embebido

Como requerimiento principal, el sistema embebido debe tener un puerto de comunicación para acceder a los datos de la celda de carga, tipo serial. También debe tener al menos 3 pines de salida para controlar el variador de frecuencia: uno de estos pines para la señal PWM que se requiere para el control de la velocidad y los otros dos pines para arrancar y detener el variador. Con estas características mencionadas, es posible utilizar el Arduino como sistema embebido, pero cabe recordar que se requiere de una interfaz gráfica. Con el Arduino es posible crear una interfaz básica con su módulo comercial que tiene una pantalla resistiva 3.2’ pulgadas y una resolución de 320x240.

Tabla 3.1: Comparación entre el Arduino, Mini2440 y el Raspberry PI.

SISTEMA EMBEBIDO	ARDUINO	MINI2440	RASPEBRRY PI
Integrado Principal	ATmega328	ARM9	ARM11
GPIO	14	34	26
Frecuencia de Reloj	16 MHz	400-533 MHz	700 MHz
RAM	2 KB	64 MB	512 MB
Puerto Ethernet	No	Si	si
Puertos USB	Un puerto.	Un puerto.	Dos puertos.
Capacidad máxima de almacenamiento.	32KB-512KB	32GB mediante memoria SD.	32GB mediante memoria SD.
Sistema Operativo	No tiene	Linux Embebido	Linux Embebido
Precio	19 $ [32]	77 $ [33]	44 $ [34]

En la tabla 3.1 se puede apreciar las características de tres sistemas embebidos, los tres tienen suficientes salidas GPIO para controlar el Variador de frecuencia por lo que en este aspecto los tres cumplen con lo requerido.

Se observa que el Arduino, en comparación con los otros sistemas, tiene dos limitaciones. La primera es que no cuenta con un sistema operativo; esto es una desventaja con respecto a la base de datos, ya que se requiere que se guarde considerable cantidad de data en ella. La segunda limitación es el máximo tamaño de pantalla que puede brindar el arduino que, como se mencionó previamente, es mediante una pantalla táctil de 2.3 pulgadas. Sin embargo el tamaño de la pantalla de la Balanza antes del mejoramiento era de 7 pulgadas, por lo que la pantalla del sistema embebido debe tener como mínimo esta dimensión. Por estas dos características el Arduino no sería ideal para la implementación.

Con respecto a los otros sistemas embebidos que son el Mini2440 y el Raspberry Pi, estos cumplen con todos los requerimientos. Pese a que el Raspberry Pi ofrece mejores características con respecto a rendimiento, se optó por el Mini2440 debido a que esta ofrecía una pantalla táctil ya preinstalada.

3.2.5 Selección de la pantalla táctil

El principal requerimiento para la selección de la pantalla táctil es que pueda resistir las diversas condiciones que se presentan en una área industrial. En el capítulo anterior se presentaron dos tipos de pantallas táctiles, la capacitiva y la resistiva. La ventaja de la pantalla capacitiva, en comparación con la pantalla resistiva, es que no pierden su brillo y soportan multi-touch pero entre sus desventajas se encuentran su bajo soporte al polvo, agua y cambios de temperatura. Por el otro lado las pantallas resistivas soportan todas estas condiciones y son más económicas. Es por estas características que las pantallas resistivas son el tipo de pantalla táctil ideal para este trabajo.

Con las características sobre sistema embebido y pantallas táctiles mencionadas en la sección 3.2.4 y 3.2.5, se seleccionó el TQ2440 como el sistema ideal para el trabajo que se va realizar en esta tesis. El TQ2440 es un paquete que contiene un sistema embebido mini2440 y una pantalla táctil resistiva de 7 pulgadas que se mencionara con mayor detalle en la sección 3.2.5.1.

3.2.5.1 Sistema embebido TQ2440

El TQ2440 es una tarjeta de desarrollo que posee un procesador con arquitectura ARM y diversos puertos de salida para el control de diferentes aplicaciones. Posee las siguientes características:

- El dispositivo soporta sistemas operativos Linux 2.6.29 (Soporta distribución Qtopia-2.2.0) o Windows CE.
- El procesador es el modelo SAMSUNG S3C2440A de 400-533MHz, es un microprocesador de 16/32 bits RISC con núcleo de arquitectura ARM920T.
- Memoria RAM dinámica y síncrona de 64MB.
- Memoria flash 256Mb tipo NAND y 2MB tipo NOR. En la memoria tipo NOR se encuentra guardada el software de arranque del sistema embebido y en la memoria tipo NAND se encuentra el resto de software como por ejemplo el Linux Kernel instalado.

Figura 3.2: Sistema embebido TQ2440. [19]

3.3 Software empleado

3.3.1 Selección del Sistema operativo

Para el desarrollo del software se decidió usar el sistema operativo Linux embebido por motivo económico. El costo de licencia del producto Windows CE es de 3 dólares y el costo de la herramienta para la creación de plataformas es de 995 dólares. [20] Adicionalmente a esto, también se realizó una comparación de sus otras características como se puede apreciar en la tabla 3.2.:

Tabla 3.2: Comparación entre Linux embebido y Windows CE.

	Linux embebido	Windows CE
Herramientas	A pesar de las variantes realizadas han mejorado considerablemente en la facilidad de uso de las herramientas, Windows sigue siendo más sencillo.	Microsoft ha realizado varios avances y cambios en su sistema operativo compacto que ha posibilitado la evolución de sus programas con un entorno más amigable.
Costo del Software	Muchos de los programas disponibles en Linux son gratuitos o libres.	La mayoría de los programas disponibles no son gratuitos y el costo depende de que tan sofisticado y conocida sea el programa.

Rendimiento	Linux no es tiempo real. Por lo que no es ideal para proceso de automatización. Aunque existen extensiones que posibilitan esta característica. [30]	Windows CE es un sistema operativo a tiempo real, es decir interactúa con el exterior, emite respuestas correctas y cumple con las restricciones temporales .[31]

3.3.2 Selección de base de datos

Para la selección de base de datos se identificó dos posibles candidatos, MySQL y el SQLite. Ambos son dos gestores de base de datos libres que brindan las facilidades requeridas para el control de datos. También es posible la conversión de las tablas a un archivo .csv., formato que puede abrir cualquier software de hojas de cálculo.

MySQL, comparado con SQLite, brinda más posibilidades como el manejo de privilegios de usuario y enlace entre cliente y servidor. La desventaja de Mysql para la aplicación de la tesis es que es un sofisticado y robusto gestor de datos, es decir requiere mayor configuraciones para su instalación en un sistema embebido.

SQLite es mucho más compacto en comparación con MySQL, brinda determinadas posibilidades suficientes para el manejo de datos. No es muy robusto y existe una versión para sistemas embebidos. No posee control de privilegios de usuario ni un enlace cliente con servidor pero permite todo los requerimientos para la aplicación de la tesis. Por ello se eligió este gestor de base de datos para el desarrollo de la aplicación.

Ambos gestores de base de datos pueden ser manejados mediante librerías especiales de las herramientas de objetos gráficos. Por ejemplo el Qt tiene por defecto instalado la librería para controlar el SQLite pero no para el MySQL, si se desease utilizar este último tiene que instalarlo manualmente.

3.3.3 Selección de la herramienta de objetos gráficos

En el capítulo 2 se presentaron dos herramientas de objetos muy utilizadas en los sistemas operativos Linux, el QT y el GTK+. Para el diseño de la interfaz en esta tesis se utilizó el QT por las siguientes ventajas:

- Existe una extensiva documentación para el manejo del Qt, en donde se muestran diferentes ejemplos, librerías y una gran comunidad.
- Aunque GTK+ y QT utilizan similar cantidad de memoria al ejecutarse sus aplicaciones. GTK+ no tiene un equivalente al módulo QtQuick, que supera lo términos de rendimiento de animaciones y transacciones al GTK+.
- Para el uso de las librerías de Qt no se utiliza las características más avanzadas del C++, por lo que es más sencillo la programación con respecto al GTK+.

3.3.4 Diseño de la interfaz gráfica.

A continuación se detallara la estructura del programa de la interfaz que se desarrolló con la herramienta de objetos gráficos Qt. Todo el código desarrollado se encuentra en anexos, por lo que se solo comentara de los puntos más importantes del programa.

El programa está dividido en dos partes, una parte que está escrita en C++ y otra se encuentra escrito en QML. El QML es un lenguaje declarativo del Qt que fue especialmente creado para el diseño de interfaces. El QML permite añadir animaciones e imágenes al entorno gráfico y también se puede crear funciones con ayuda del lenguaje JavaScript.

La lógica del programa fue escrito en lenguaje C++, esta parte del programa es la encargada de controlar los contadores, almacenamiento de datos, y la comunicación serial. Por el otro lado la parte gráfica, las animaciones y los diseños visuales fueron escritos en QML. A continuación se procederá a hablar de estas dos secciones del programa, primero el escrito en C++ y luego la parte escrita en QML.

3.3.4.1 Código en C++.

Utilizando el lenguaje C++ se han creado varias clases para facilitar y ordenar el manejo del programa, cada una de ellas tiene el propósito de crear objeto con las funciones y señales que requiere el programa. Dichas clases se pueden apreciar en la figura 3.5. La clase principal se le ha denominado “MainClass” es la clase que permite encapsular todo el programa en un objeto para luego mostrarla.

La clase llamada “Sqlite” es la clase diseñada para el manejo de base de datos, cualquier dato que se desee guardar a la base de datos se envía a esta clase, del mismo modo si se desease mostrar un dato se envía una función de requerimiento del dato a esta clase.

Las clases “Icon1”, “Icon4”, “Icon5” son las clases que agrupan el programa de acuerdo a los iconos que se van a utilizar. Se van a utilizar 6 iconos, pero solo 3 de ellas requieren de una lógica escrita en C++, el resto de iconos son botones y animaciones escritas mayormente en lenguaje QML. El icono 1 es el que presenta los pesos adquiridos por la balanza a tiempo real y verifica que estos pesos se encuentren en el rango definido. Por ello el objeto de la clase “Icon1” tiene funciones de entrada para que adquiera los parámetros de peso, producto, límites y contadores. A partir de estos datos el objeto crea la animación en QML, una aguja que cambia su ángulo dependiendo el valor del peso. El icono 4 es el que muestra los datos almacenados en la base de datos y posibilita el acceso a ella para modificarla. La clase “Icon4” posibilita la creación del objeto que brinda todas las funciones necesarias para lograr el propósito del icono 4. Dentro del icono 4 se selecciona el producto que se desea usar para que se muestre en el icono 1 por ello existe una función que envía los parámetros seleccionados desde el icono 4 a icono 1. El icono 5 es el que permite guardar los datos contenidos en el sistema embebido a una memoria SD externa, por ello la clase “Icon5” tiene una función como entrada para obtener los valores de los pesos.

Las clases “TimerWeigth”, “TimerSpeed” y “TimerClock” utilizan en su estructura una o más clases denominadas “QTimer”, que es un temporizador. El temporizador se utiliza para controlar el flujo de datos que se requiere. Por ejemplo el objeto proveniente de la clase “TimerWeigth”, tiene un temporizador que a cada segundo envía una petición de valor de peso a la celda y luego lo envía al objeto de la clase “Icon5” para que sea guardado en la base de datos.

La clase “TimerSpeed” utiliza dos temporizadores para crear una señal PWM que será enviada al variador.

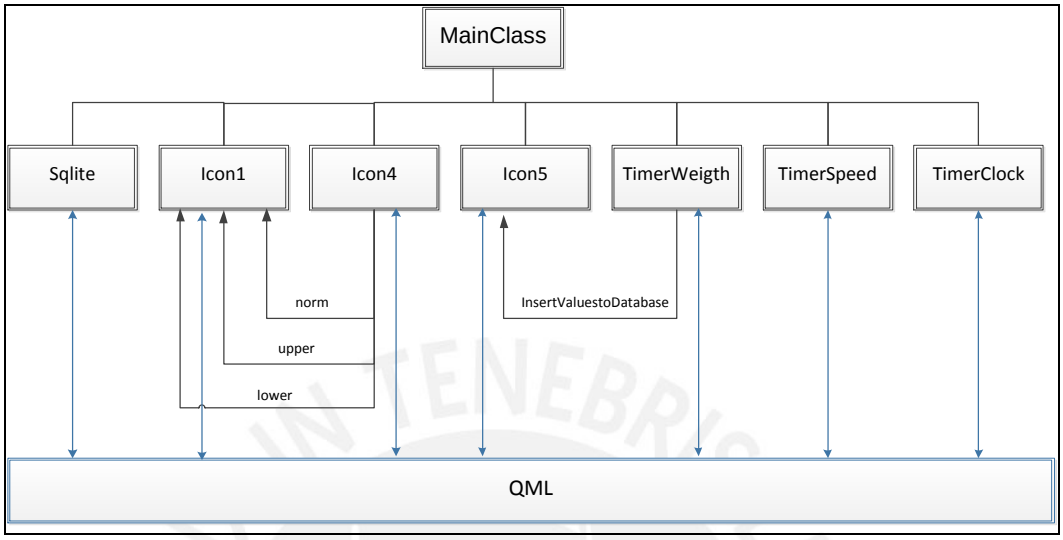

Figura 3.3: Jerarquía de clases creadas para el código en C++.

3.3.4.1.1 Programa para la comunicación con la celda de carga.

La celda de carga, como se mencionó, tiene sus propios protocolos y un tipo de comunicación serial. Por lo que se requería una forma de acceder a ella, debido a esto se utilizó un módulo adicional a los que posee las librerías del Qt, un módulo de comunicación serial.

El módulo que se utilizó para la comunicación del puerto RS232 es el módulo llamado QextSerialPort versión 1.2 Beta. El QextSerialPort es una extensión de las librerías del Qt para el control de los parámetros y manipulación de información que brinda el puerto que se desea utilizar. La instalación depende del modo que se desee utilizar el módulo, los pasos se detallan en su página web [21].

Las características del puerto que se utilizaron son:

Tabla 3.3: Características de comunicación serial.

Característica	Valor
Velocidad	9600 baudios
Paridad	Par
Numero de bits	8
Bits de paridad	1

Con estas características, el código en C++ para configurar el puerto utilizando el módulo QextSerialPort es:

```
port = new QextSerialPort("/dev/ttyUSB0");
port ->setBaudRate(BAUD9600);
port ->setParity(PAR_EVEN);
port ->setDataBits(DATA_8);
port ->setStopBits(STOP_1);
```

Una vez que el puerto está configurado, se procede a realizar la petición del valor del peso a la celda de carga. Para lograr obtener el valor del peso obtenida por la celda de carga, se hace uso de los protocolos que tiene la celda. Dichos protocolos se encuentran en el documento PW20i que está en anexos. Mediante esta gran cantidad de protocolos se puede construir diferentes comandos de petición a la celda para que realice el requerimiento deseado. Existen varias maneras de pedir el valor del peso mediante los protocolos, una de estas maneras es haciendo uso del protocolo "MAV?". El protocola "MAV?" envía el peso si solo si el peso a variado, caso contrario envía un valor constante.

El protocolo "S31" es el comando que inicia la trama de petición, de este modo el comando de petición de valor de peso cuando cambia resulta ser "S31;MAV?;". Se puede ver en la figura 3.6 el procedimiento de la comunicación entre el sistema embebido TQ2440 y la celda de carga.

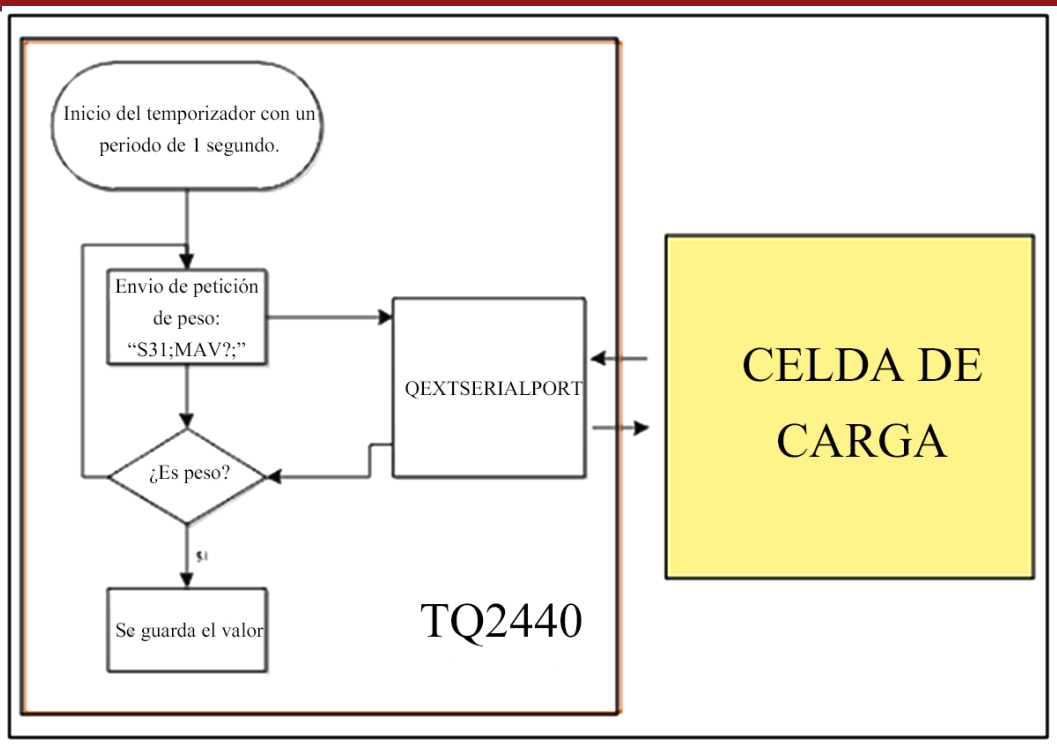

Figura 3.4: Diagrama de Flujo del programa para la comunicación serial.

3.3.4.1.2 Programa para el control del variador de frecuencia.

Para el control del variador de frecuencia se hace uso de tres pines del sistema embebido TQ2440. Un pin se utilizó para detener y arrancar el variador de frecuencia, y otro pin para conectar las tierras del variador y el embebido. El último pin es la señal PWM de frecuencia 1Hz cuyo ciclo de trabajo define la frecuencia del motor a controlar.

Para crear la seña PWM lo que se hace es usar dos temporizadores, un temporizador siempre en alta y la otra en baja. Los dos temporizadores son llamados "timer1" y "timer2". Estos dos temporizadores son usados en forma continua, es decir primero se inicia un temporizador luego al terminar empieza el otro y después cuando este termina, el primero vuelve a iniciarse y así sucesivamente creándose de este modo la señal PWM. Para que la frecuencia de la PWM no varíe, el programa asegura que los periodos de las ondas sumen siempre 1 segundo (1Hz). El programa realiza la modificación del periodo de los temporizadores y de este modo la variación de la frecuencia del variador.

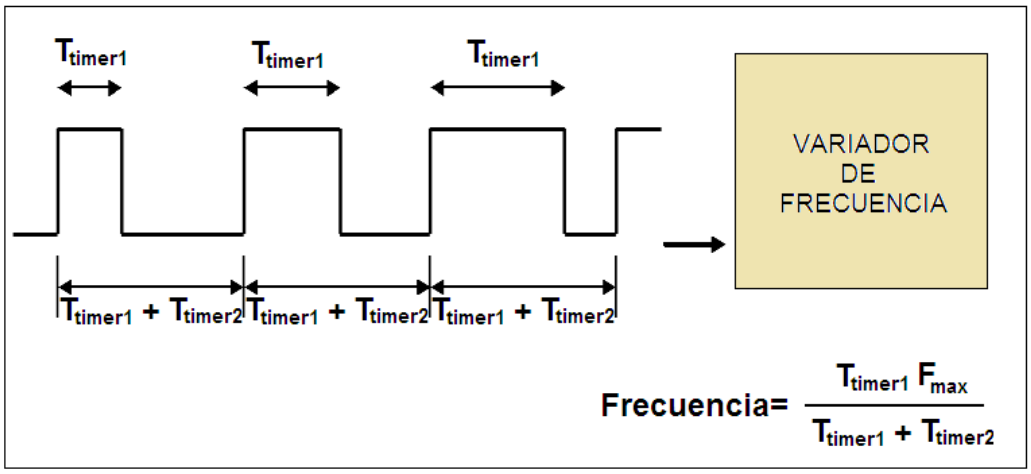

Figura 3.5: Señal PWM creada por el programa. Las clases “timer1” y “timer2” son los temporizadores que ayudan para la creación

3.3.4.2 Código en QML.

El lenguaje QML brinda una forma declarativa de crear las interfaces. Este tipo de programación facilita la creación de interfaces detallados por lo mismo que ha sido diseñado con ese objetivo.

Para el diseño de la parte visual con QML, se estudió los diferentes modos para manipular y visualizar los datos de la aplicación. Cada diseño depende del tipo de aplicación, cuanta información se desea controlar y de la imaginación del diseñador. La complejidad del diseño puede ser desde un simple botón hasta una simulación de botones y pantallas debido a ello se realizó incluso un bosquejo antes su diseño.

Figura 3.6: Ventana de iconos de la interfaz gráfica.

Al iniciarse la interfaz gráfica se muestra es la una primera ventana de iconos (ver figura 3.8). El primer icono, el de esquina superior izquierda es el icono que nos permite acceder a la aplicación que muestra el valor del peso en tiempo real. La aplicación cuenta con una animación que simula una aguja y muestra una tabla. La tabla contiene dos contadores: un contador que es la cantidad de productos buenos, estos que tienen el valor del peso en el rango definido en la base de datos, y otro contador que muestran la cantidad de productos fuera del rango. En esta misma tabla también se muestra el peso ideal, el máximo peso y su mínimo peso.

El diseño que se realizó se basó en una balanza analógica. Esta tiene dos animaciones, la primera es la animación de la aguja que muestra que tan lejos está el valor del peso con el valor ideal de peso. La segunda animación es el cambio de imagen del fondo; si el peso está fuera del rango el fondo de la imagen cambia a un color rojo, caso contrario si el peso está dentro del rango el fondo es verde. El archivo fundamental para el funcionamiento de la animación es el “dial.qml” dado en las siguientes líneas:

```
Image {
        id: needle
        x: imagedial.width/2;
        y: 302
        smooth: true
        source: "images/pointer2.png"
        transform: Rotation {
            id: needleRotation
            origin.x: needle.width/2; origin.y: 15
            angle: (root.value+90);

            Behavior on angle {
                SpringAnimation {
                    spring: 1.4
                    damping: .15
                }
            }
          }
          }
```

Este segmento de código del archivo “dial.qml” es la parte esencial para el funcionamiento de la animación de la aguja. Se usa la función “Rotation” cuya primera acción es indicar el eje de las coordenadas mediante “origin”. Luego para manipular el ángulo se brinda la información del valor del ángulo en la línea después de “angle:”. Las otras variables como “Behavior” y “smooth” son

parámetros para adornar la animación. También se implementó, como se mencionó, un indicador para demostrar si el peso se encuentra en el rango, para ello se cambios de estado con las siguientes líneas:

```
states: [
          State {
             name: "Icon1_state"; when: (dial.value>90+angledown)&&
(dial.value<angleup+90)
          PropertyChanges { target: imagedial; source:"images/Fond-dial-meter-
verde.png" } }          ]
```

Lo que realiza estos comandos es modificar el fondo de la imagen cada vez que el ángulo se encuentre dentro del rango. Son dos fondos, uno rojo y uno verde. De esta manera se crea una animación de cambio de color.

El archivo “dial2complete.qml” utiliza al archivo “dial.qml” como un componente mediante las siguientes líneas:

```
    Dial2 {
        id: dial

        value: valuedial
        normdial: normdisplay
        upperdial: uppervalue(normdisplay,upperdisplay)
        lowerdial: lowervalue(normdisplay,lowerdisplay)
        angleup: angleupper(lowerdisplay,upperdisplay)
        angledown: anglelower(lowerdisplay,upperdisplay)
    }
```

Como se puede ver es posible llamar a un archivo QML desde otro como si fuera algún otro componente. Se declara con el nombre que se creó el archivo en este caso con el nombre “Dial” {} y se procede a definir los parámetros requeridos. El valor del ángulo es obtenido mediante el valor de peso que se obtiene del enlace entre QML y C++, el valor del peso está definido mediante la variable “normdisplay”. Es posible verificar esto al leerlo en el código completo. Por ejemplo en el archivo “icon1.cpp” las líneas:

```
void SetNormString(const QString norm)
  {   QDeclarativeContext *context = m_view->rootContext();
      context->setContextProperty("normdisplay",norm);
  }
```

En esta línea lo que se hace es modificar la variable “normdisplay” con el valor 23 y esta puede ser accedida en cualquier archivo qml. En la figura 3.9 se muestra el resultado final.

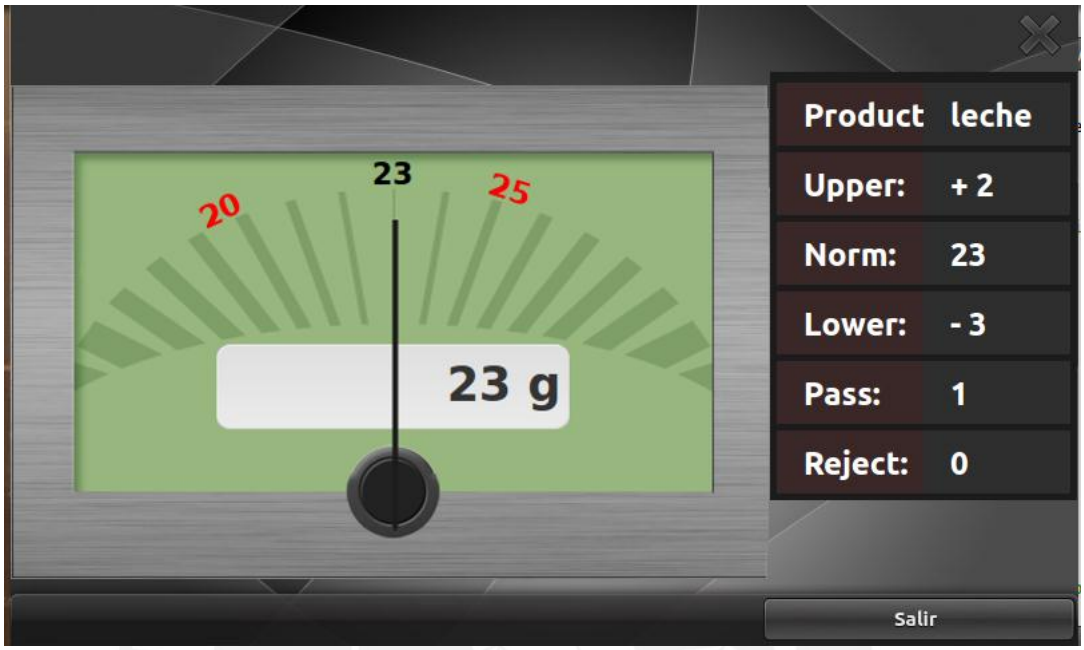

Figura 3.7: Aplicación de la interfaz gráfica para la muestra de valores de peso en tiempo real.

El tercer icono de la ventana de iconos, el que se encuentra en la parte superior derecha, permite acceder a la aplicación para el control del variador. Dentro de esta aplicación se tienen cinco botones, dos botones para disminuir y aumentar la frecuencia. El cambio de valor de la frecuencia no se realiza de inmediato, se tiene otro botón llamado “listo”, al presionar este botón se procede a cambiar el valor de la frecuencia. Los dos últimos botones son para arrancar y detener el variador y también se cuenta con una pantalla que muestra el valor de la frecuencia.

Como se mencionó para la creación de la interfaz en esta aplicación se crearon 3 botones y una pantalla para mostrar la frecuencia que gira el motor que controla la faja. El código completo está en el archivo “icon3.qml”. Dentro de este se puede resaltar las siguientes líneas:

```
Grid{
 id:gridButtons
 anchors.verticalCenter: parent.verticalCenter
 columns: 1
 spacing: 20
 width:buttonup.width
```

```
x:20
    Tools.TriangleButton
    {
      id: buttonup; width: 100; height: 100; opacity: 0.9;
      onClicked: TimerSpeedBinding.setF(TimerSpeedBinding.F+1)}

    Tools.TriangleButton
    {
       transform: Rotation{ origin.x:buttondown.width/2; origin.y:buttondown.height/2;
angle: 180}
       id: buttondown; width: 100; height: 100; opacity: 0.9
       onClicked: TimerSpeedBinding.setF(TimerSpeedBinding.F-1)}
   }
```

Donde “TriangleButton” es un archivo QML que define el tipo de botón triangular. Se puede observar que se puede llamar más de una vez el mismo tipo de componente. El segundo botón usa la función “Rotation” para girar el botón en el sentido opuesto, de esta manera se tiene un botón triangular que apunta al sentido opuesto del otro botón. En cada botón se define el parámetro “onClicked” que convierte al componente en un elemento que interactúa al hacer Click dentro de su área. La figura 3.10 muestra el resultado final de la interfaz.

Figura 3.8: Aplicación de la interfaz gráfica para el control del variador de frecuencia.

El cuarto icono, el que se encuentra en la parte inferior izquierda, permite acceder a la aplicación de control de base de datos. El diseño se encuentra en la figura 3.11. El diseño cuenta con una tabla en donde se muestran los valores de los productos con sus respectivos pesos ideales y los limites. En esta misma

aplicación se puede modificar los datos y almacenar nuevos mediante los botones y los espacios en blancos que se encuentran en el diseño.

El quinto icono, ubicado en el centro de la parte inferior, es el que permite acceder a la aplicación para grabar los datos almacenados en el sistema embebido a una memora SD.

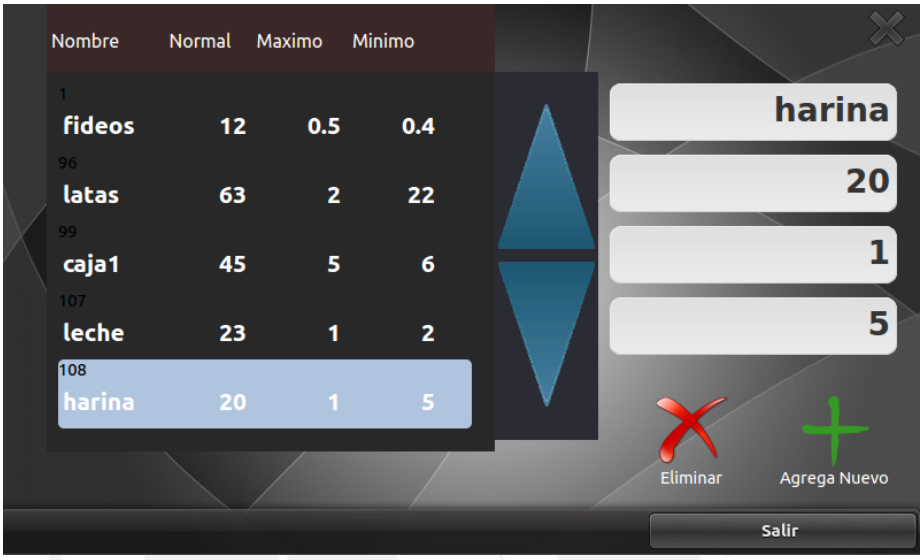

Figura 3.9: Aplicación para el acceso de base de datos.

Figura 3.10: Diseño del teclado para el ingreso de texto a la base de datos.

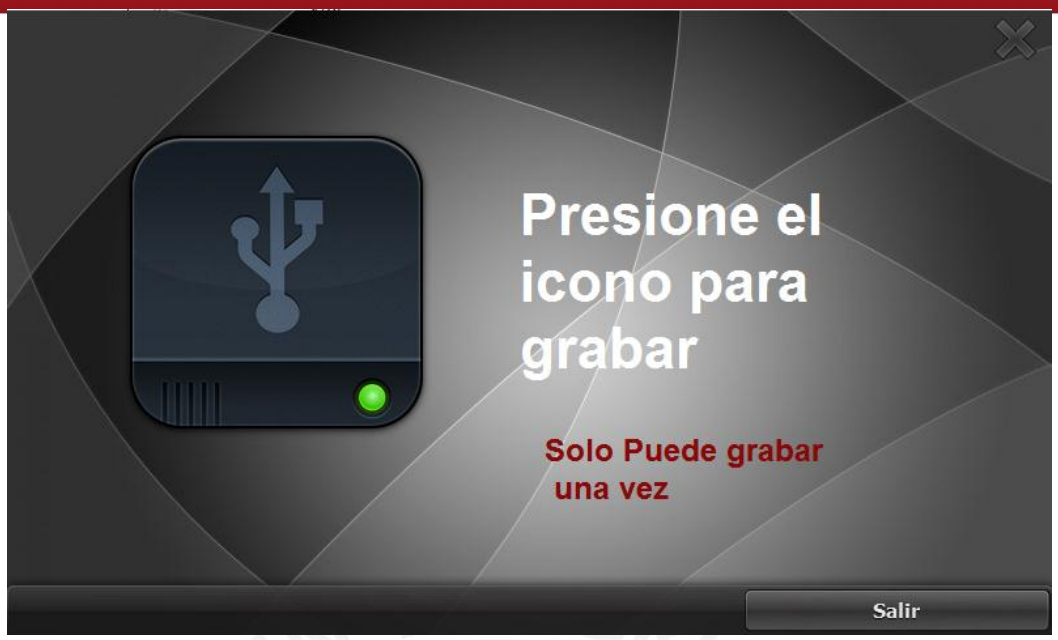

Figura 3.11: Aplicación de la interfaz gráfica para el guardado de datos en una memoria externa, en este caso una memoria SD.

CAPÍTULO 4

PRUEBAS Y RESULTADOS

4.1 Balanza Dinámica.

El Sistema control de la balanza dinámica, como se mencionó en la sección anterior, está conformada por tres componentes principales: la celda de carga modelo PW15 HI, el variador de frecuencia y el sistema embebido. Para poder comunicar el sistema embebido con la celda de carga y el variador de frecuencia se utiliza el puerto serial RS232 y los puertos GPIO respectivamente.
En la figura 4.1 se puede apreciar el resultado final de la balanza dinámica.

Figura 4.1: Resultado final de la balanza dinámica en la sala de exposición de la empresa SISCODE.

Figura 4.2: Variador de frecuencia dentro de la Balanza dinámica.

Figura 4.3: Celda de carga debajo de la Banda Transportadora.

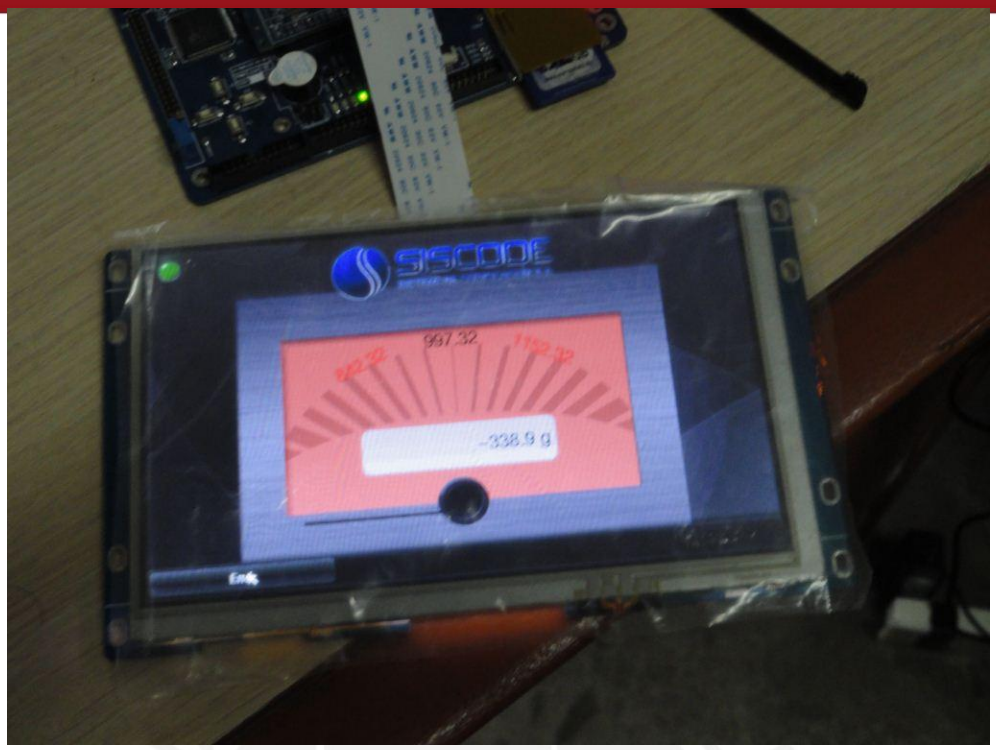

Figura 4.4: Sistema embebido funcionando con el programa mostrando un valor fuera del rango

Figura 4.5: Sistema embebido funcionando con el programa mostrando un valor dentro del rango

4.2 Resultados del programa.

Al ejecutarse el programa en el sistema embebido lo que se ejecuta primero de forma automática es, si no existe, la creación de la base de datos general y dos tablas dentro de ella para el control la data. La base de datos se le ha llamado “test3.db” y a las tablas “t1” y “TableWeigth”. Siendo “t1” la tabla en el cual se almacenan los productos y sus características como peso normal y límites. Y la tabla “TableWeigth” la tabla en donde se almacenan los valores de los pesos que se requieren para ser guardados en una memoria SD.

Esto se puede comprobar al acceder al dispositivo embebido mediante la terminal como se muestra en la figura 4.4. Se puede observar que se ha creado la base de datos “test3.db” como se deseaba y dentro de ella se han creado las dos tablas cuyas características se detallan después de realizar el comando “.schema”. También se puede ver que se han creado 6 productos dentro de la tabla t1: fideos, latas, caja1, leche, harina y azúcar, también se puede visualizar sus pesos normales que son 12, 63, 45, 23, 20 y 45 gramos respectivamente.

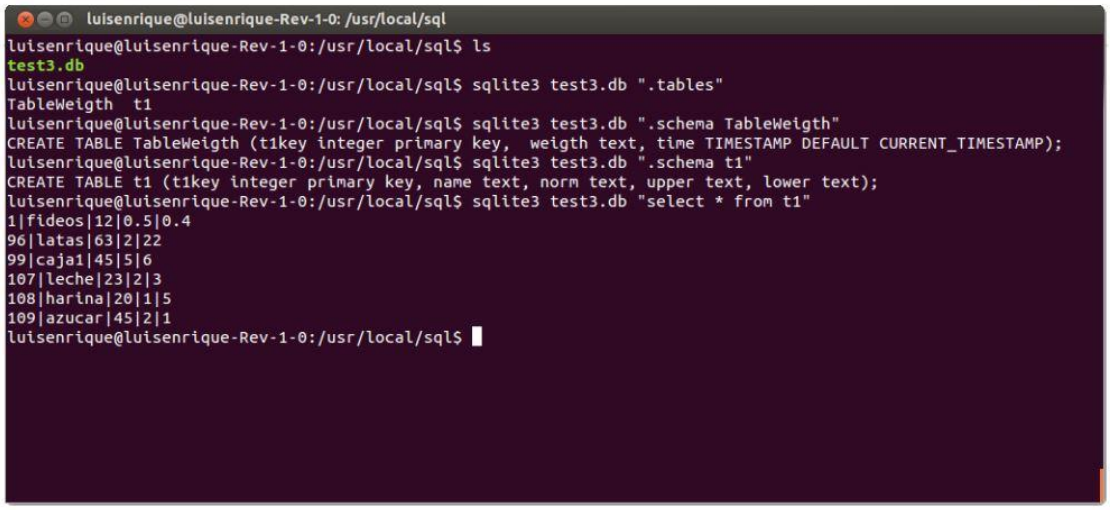

Figura 4.4: Terminal que muestra el acceso al sistema embebido para la comprobación de la existencia de base de datos y tablas.

La tabla “TablaWeigth” es la tabla que se usa para guardar los valores de peso que brinda la celda de carga. Como se puede observar en la figura 4.4, dentro de la tabla adicionalmente al parámetro “text”, que es requerido para el valor del peso, se tiene el parámetro “timestamp default current_timestamp” que guarda el valor de los instantes en que se guarda el peso. En la tabla 4.1 se muestran los valores de peso obtenidos con una pesa de 1 kilo.

Tabla 4.1 Valores de peso obtenidos al usar la pesa de un kilo sobre la plataforma de la balanza dinámica.

954,811	9:30:50
954,700	9:31:23
954,614	9:31:33
954,723	9:31:57
954,812	9:32:04
954,736	9:32:20
954,651	9:32:48
954,714	9:33:50
954,816	9:34:12

4.3 Costos.

El costo del sistema embebido TQ2440 y la pantalla táctil de 7 pulgadas usados en este proyecto es de 170 dólares hasta la fecha [35]. El costo de la celda de carga es de 630 dólares [36], resultando de este modo un total de 800 dólares el costo total del diseño e implementación de la mejora.

CONCLUSIONES

De acuerdo a los resultados obtenidos se logró que el sistema embebido controle los diversos componentes por medio de sus salidas: puerto serial y GPIO. A su vez la selección de la arquitectura y procesador de este influyó en el buen rendimiento para el análisis de la data obtenida por medio de la celda de carga.

La comunicación entre la celda de carga y el sistema embebido fue un éxito, la cual permitió al sistema en general obtener el valor de peso a tiempo real. Las Celdas de Carga Digital tienen en la actualidad un amplio uso en el campo industrial. Estos equipos son capaces de comunicarse con otros equipos digitales por medio de comunicación serial.

Con lo referente al programa implementado con QT en el sistema embebido, el uso del QML para el diseño de interfaz permitió y facilitó la creación de diversos diseños elegantes e innovadores para la aplicación. Aunque el QML es reciente, contiene todo lo que se requiere para una interfaz moderna y atractiva, justamente lo que se deseaba mejorar de la balanza dinámica que se compró. También podemos concluir que la separación del código del programa en dos partes: la lógica escrita en C++ y el diseño del aspecto de la interfaz escrita en QML, no solo ordeno el código sino que también brindo un mayor rendimiento al ejecutarse el programa. Esto es debido a que se está utilizado cada lenguaje, C++ y QML, con el propósito que se le designo.

El uso del módulo de QT llamado QextserialPort para el intercambio de información de la celda de carga y el sistema embebido en forma serial dio resultados favorables. Además se diseñó e implementó una terminal virtual, que incluye un teclado y una pequeña pantalla para comprobar la comunicación entre el sistema embebido y la celda de carga. Lo cual ha facilitado el manejo y la comunicación del puerto serial RS232.

Finalmente, podemos concluir que el algoritmo implementado en el programa del sistema embebido para la interfaz gráfica, la determinación del valor del peso que brinda la celda de carga y la modificación de la frecuencia del variador dio resultados favorables para las condiciones definidas.

RECOMENDACIONES

1) Con respecto a la parte mecánica, para obtener del valor de peso con mayor precisión es necesario verificar la adecuada posición de la celda de carga y la banda transportadora que está encima de ella. Es recomendable que estas se encuentren lo más fija posible para que la celda de carga se relacione directamente con el movimiento de presión que se ejerce al colocarse un objeto encima de la banda transportadora y no a través de elementos intermedios.

2) Para mejorar los resultados con respecto a la interfaz gráfica y control de base de datos. Se recomienda usar un sistema embebido con mayor memoria RAM si se desease utilizar animaciones y figuras más sofisticadas. También es recomendable utilizar distribuciones de Linux embebidos conocidas como el Debian que soportan gestores de base de datos con acceso a red como por ejemplo el MySql.

3) Para la comunicación serial con el QT es recomendable usar un módulo como el QextserialPort para facilitar el propósito. Es posible comunicarse sin este módulo haciendo uso de la clase QIODevice pero requiere el uso de más funciones y eventos para el control de los pines que tiene el puerto serial RS232.

4) Para la creación del ejecutable adecuado mediante las librerías que brinda Qt, se debe tener en cuenta no solo el sistema operativo sino la versión y características del kernel y el sistema embebido que se pretende usar como interfaz. Por ejemplo al utilizar la tarjeta TQ2440 se tenía un límite de tamaño de la fuente de texto. Y no soportaba algunos formatos de imágenes.

5) Para mayor rendimiento del programa de la interfaz, al implementar animaciones o figuras interactivas en QML es recomendable aumentar la interacción entre C++ y QML, y utilizar el menor número de funciones JavaScript dentro de los archivos QML y remplazarlos con funciones en código C++. Es incluso una recomendación que se nos plantea en la página web de Nokia [27]

BIBLIOGRAFIA

[1] Página oficial de la marca industrial Rehoo. REHOO INDUSTRIAL COMPANY
< http://www.rehoo.com.hk/products/type2/CWC-230NS/index.html>
Consulta 25 de junio.

[2] Daniel W.Lewis. 2012. Fundamentals of Embedded software with the ARM Cortex-M3

[3] Christopher Hallinan, 2006 .Embedded Linux Primer.

[5] Balanza DACS-G de ISHIDA. Consulta 25 de junio.
<http://www.ishida.com/products/dacs_g/dacs_g_specifications.html>
Consulta: Junio 2013

[6] NAND flash Memory. Consulta 25 de junio.
<https://www.youtube.com/watch?v=LIX69Mpmqko>
Consulta: Junio 2013

[7] Understanding NAND and NOR flash memories. Consulta 25 de junio.
<http://info.quadros.com/blog/bid/101731/Understanding-NAND-and-NOR-Flash-Devices>
Consulta: Junio 2013

[8] Características de arduino due.
<http://arduino.cc/en/Main/ArduinoBoardDue>
Consulta: Julio 2013

[9] Características del sistema embebido 2440.
<http://www.friendlyarm.net/products/mini2440>
Consulta: Junio 2013

[10] Características del sistema embebido Raspberry Pi.
<http://en.wikipedia.org/wiki/File:RaspberryPi.jpg>
Consulta: Junio 2013

[11] Características del sistema embebido Cubieboard.
<http://www.allwinnertech.com/en/devkit/CubieBoard.html>
Consulta: Junio 2013

[12] Patente desarrollada por Lauri Holm
"DYNAMIC WEIGHING METHOD OF DETERMINING A LOAD MEASUREMENT VALUE AND THE RESOLUTION THEREOF"
Consulta: Junio 2013

[14] Manuel Álvarez Pullido ,2000. Convertidores de Frecuencia, Controladores de Motores y SSR.

[15] Dan Saffer, 2008. Designing Gestural Interfaces.

[16] Using resistive touch screens for human/machine interface
< http://www.ti.com/lit/an/slyt209a/slyt209a.pdf >
Consulta: Junio 2013

[17] Fia Stenmark , 2008. Design of touch screen interface for a mobile position aware instant messaging client.

[18] Windows Embedded 7
<http://www.microsoft.com/windowsembedded/en-us/evaluate/windows-embedded-compact-7.aspx>
Consulta: Junio 2013

[19] Imagen del TQ2440
<http://www.cutedigi.com/popup_image.php?pID=4328&image=0>
Consulta: Junio 2013

[21] Getting Started with QextSerialPort. Consulta el 15 de junio.
< http://code.google.com/p/qextserialport/wiki/QextSerialPort_1_2_RC>
Consulta: Junio 2013

[22] Duane Wessels y Matthew J. Weaver 2006. Make Projects: Small Form Factor PCs.

[23] Jon Masters, Gilard Ben-Yossef y Philippe Gerum, 2008. Building Embedded Linux Systems.

[24] Página oficial de HBM.
<www.hbm.com >
Consulta: Junio 2013

[25] Two Flash Technologies Compared: NOR vs NAND
https://focus.ti.com/pdfs/omap/diskonchipvsnor.pdf
Consulta: Junio 2013

[26] Caracteristicas del sistmea embebido PCduino
<https://www.sparkfun.com/products/11712>
Consulta: Junio 2013

[27] QML performance tips and tricks. Consulta el 24 de junio
http://harmattan-dev.nokia.com/docs/library/html/guide/html/Developer_Library_Best_practices_for_application_development_QML_performance_tips_and_tricks.html/>
Consulta: Junio 2013

[28] Imagen de celda de carga. Consulta 25 de junio.
<http://www.hbm.com/en/menu/products/weighing-components/single-point-load-cells/pw15ah/>
Consulta: Junio 2013

[29] New Analysis Casts Doubt On Intel's Smartphone Performance vs. ARM
<http://hothardware.com/News/New-Analysis-Casts-Doubt-On-Intels-Smartphone-Performance-vs-ARM-/>
Consulta: Junio 2013

[30] Intro to Real-Time Linux for Embedded Developers
<http://www.linux.com/news/featured-blogs/200-libby-clark/710319-intro-to-real-time-linux-for-embedded-developers>
Consulta: Junio 2013

[31] Stanislav Pavlov, Pavel Belevsky, 2008. Windows Embedded CE 6.0 Fundamentals

[32] Atmega2560-16au Atmega16u2 Board + Gratis Cable Para 2012 Arduino Mega 2560 R3
<http://www.ebay.com/itm/ATmega2560-16AU-ATMEGA16U2-Board-Free-Cable-For-2012-ARDUINOs-MEGA-2560-R3-/171037664571?pt=LH_DefaultDomain_0&hash=item27d2a3a53b>
Consulta: agosto 2013

[33] 256m Mini2440 s3c2440 Arm9 Board Soporte Windows Ce 5 y 6 / Linux 2.6 / Android
<http://www.ebay.com/itm/256M-Mini2440-S3C2440-ARM9-Board-support-Windows-CE-5-and-6-Linux-2-6-Android-/271025359857?pt=LH_DefaultDomain_0&hash=item3f1a5ecbf1>
Consulta: agosto 2013

[34] Raspberry Pi 2.0 Model B 512MB RASPBRRY
<http://www.ebay.com/itm/NEW-IN-BOX-Raspberry-Pi-2-0-Model-B-512MB-Element-14-Linux-System-Board-/221202690655?pt=LH_DefaultDomain_0&hash=item3380b5325f>
Consulta: agosto 2013

[35] Sistema embebido TQ2440 S3C2440
<http://www.cutedigi.com/development-tools/arm9-development-board/tq2440-s3c2440-arm920t-development-board-7-touchscreen-lcd.html>
Consulta: febrero 2015

[36] RICE LAKE WEIGHING SYSTEMS
<http://www.ricelake.com/products/load-cells/single-points/hbm-single-points/pw15ah-single-point-stainless-steel/view/parts>
Consulta: febrero 2015