Universidad Tecnológica Nacional

Proyecto Final

Diseño e implementación de un sistema embebido de impresión inteligente

---

Autor:

- Javier Agustín Serrano Vázquez

Director:

Proyecto final presentado para cumplimentar los requisitos académicos para acceder al título de Ingeniero Electrónico

en la

Facultad Regional Paraná

Octubre de 2019

Declaración de autoría:

Yo declaro que el Proyecto Final “Diseño e implementación de un sistema embebido de impresión inteligente” y el trabajo realizado es propio.
Declaro:

- Este trabajo fue realizado en su totalidad, o principalmente, para acceder al título de grado de Ingeniero electrónico, en la Universidad Tecnológica Nacional, Regional Paraná.
- Se establece claramente que el desarrollo realizado y el informe que lo acompaña no han sido previamente utilizados para acceder a otro título de grado o pre-grado.
- Siempre que se ha utilizado trabajo de otros autores, el mismo ha sido correctamente citado. El resto del trabajo es de autoría propia.
- Se ha indicado y agradecido correctamente a todos aquellos que han colaborado con el presente trabajo.
- Cuando el trabajo forma parte de un trabajo de mayores dimensiones donde han participado otras personas, se ha indicado claramente el alcance del trabajo realizado.

Firmas:

Fecha:

Agradecimientos:

Al inmensurable esfuerzo de mis padres por permitirme estudiar, a su innegable apoyo incondicional y sus sabias palabras de guía.

A mis hermanos, con su apoyo y ejemplo me dan fuerzas para seguir en todo momento.

A mi novia y compañera de vida, que me ha acompañado desde el día uno en todos los obstáculos que se me han ido presentando.

A mis abuelos, tíos, primos y amigos por ser una fuente de energía positiva absoluta.

A todo el equipo de IPVISION por darme la posibilidad de trabajar de forma remota y por tiempo parcial mientras terminaba mis estudios.

A la universidad tecnológica nacional y la educación pública argentina.

Javier Agustín Serrano Vázquez

Universidad Tecnológica Nacional

Abstract

Facultad Regional Paraná

Ingeniero en Electrónica

Diseño e implementación de un sistema embebido de impresión inteligente

Javier Agustín Serrano Vázquez

Abstract:

In order to provide an automated solution to a problem at the time you want to print in a specialized center, the feasibility of building a product based on a raspberry pi that can integrate all the necessary modules to become independent of human intermediation is raised.

It was done by integrating different modules, such as a CH926 coin selector, a raspberry pi 3B +. a touch screen, an A4 printer.

At the same time, software was developed so that these elements can interact with each other, such as a pulse counter, an algorithm to select the files entered through a USB storage device and a code to generate a graphical interface easily interpretable by the user.

An affordable product was obtained that provides the expected functionality of being a completely independent of intermediation between the user and the final product, in this case an A4 print, but that can simply be adapted for any type of printer (3D, plans, photographs)

Keywords:

Smart printing, raspberry pi, coin selector, automated, vending machine.

Resumen:

Con el objetivo de proporcionar una solución automatizada a una problemática al momento que se desea imprimir en un centro especializado, se plantea la factibilidad de construir un producto en base a una raspberry pi que pueda integrar todos los módulos necesarios para independizarse de la intermediación humana.
El mismo fue realizado mediante la integración de distintos módulos, como son un selector de monedas CH926, una raspberry pi 3B+. una pantalla táctil, una impresora A4. Al mismo tiempo se desarrolló software para que estos elementos puedan interactuar entre si, como un contador de pulsos, un algoritmo para seleccionar los archivos ingresados mediante un dispositivo de almacenamiento USB y un código para generar una interfaz gráfica fácilmente interpretable por el usuario.
Se obtuvo un producto de un costo accesible que brinda la funcionalidad esperada de ser un equipo totalmente independiente de intermediación entre el usuario y el producto final, en este caso una impresión A4, pero que simplemente puede ser adaptado para cualquier tipo de impresora (3D, planos, fotografías)
Palabras Clave:

Impresión inteligente, raspberry pi, sensor de monedas, automatización, máquina expendedora.

Reconocimientos:

.

A todos los docentes y compañeros que me han acompañado durante toda la carrera.

Índice:

Lista de Figuras:

Lista de Tablas

Lista de Abreviaciones

- RPI: Raspberry Pi
- HDMI: High Definition Media Interface
- V: Volts
- DC: Corriente Continua
- A: Amper
- CUPS: Common Unix Printing System
- E / S: Entrada / Salida
- Ctvs.: Centavos
- GPIO: Puertos de entrada salida de uso general
- KHz: Kilohertz
- C: Lenguaje de programación C
- BCM: Broadcom Chip Model

Lista de Símbolos

Dedicado a:

Mis padres Juan Carlos y Lourdes, mis hermanos Juan Andrés y Martín, y a mi novia Karen, las personas más importantes en mi vida.

Capítulo 1: Introducción

La tecnología avanza cada vez a pasos cada vez más difíciles de seguir. La automatización de procesos mediante sistemas informáticos es una realidad difícil de detener.

Uno se puede llegar a preguntar ¿Cuáles son las ventajas de automatizar y simplificar procesos? Pues la respuesta suele ser en algunos casos sencilla, "Para ahorrar tiempo".

Claro, como dice el famoso dicho, el tiempo es oro, sin embargo, existe una inmensidad de factores a evaluar al momento de tomar la decisión de invertir en esfuerzos y desarrollos para reemplazar una tarea, actualmente manual, por un sistema automatizado.

Tales consideraciones para evaluar son:
¿El sistema me ahorrará tiempo?
¿Se ganará rapidez, momento y velocidad?
¿El proceso a automatizar es algo que se debe realizar con una determinada frecuencia?
¿Se reducirá esfuerzo y horas de trabajo?
y lo más importante de todo para saber si la automatización del sistema vale la pena:
¿tiempo desarrollando sistema vs tiempo ganado?

Con esta breve introducción acerca de la automatización de procesos, puedo presentar la idea de este proyecto, el cual se basa en resolver una problemática planteada al momento de realizar impresiones (del tipo oficio, A4, A3, planos, fotografías, modelos 3D y demás)

Normalmente cuando una persona desea imprimir algo, el procedimiento es el siguiente:
Encontrar un lugar de impresiones, acudir con un dispositivo de almacenamiento a, interactuar con un empleado del local de impresiones, confirmar el archivo o imprimir y pagar el valor requerido.

Esto se convierte en un problema cuando, necesitamos imprimir un documento y necesitamos acudir a un centro de impresiones, la cantidad de gente en el local, el tiempo de espera, la interacción con el empleado del local y, por si fuera poco, acomodarnos a los horarios de apertura de ciertos locales.
Cuando necesitamos una impresión a la madrugada o por un apuro, tenemos que esperar hasta que un centro de impresión este abierto.

Este proyecto consiste en poder crear un sistema inteligente de impresión y cobro el cual sea independiente de algún tipo de control humano, el cual pueda funcionar 24h y pueda ser adaptado a cualquier sistema de impresión como documentos, planos (A3, A2, A1, A0), fotografías, modelos 3d, etc.
El enfoque es diseñar un prototipo adaptador y demostrar su funcionamiento y escalabilidad hacia cualquier sistema de impresión para dar comodidad y facilidad al usuario y también ampliar las opciones brindadas en un centro de impresión, fotografía, impresión 3D al ofrecerle la alternativa de tener su servicio disponible las 24 horas.

Mediante el uso de una interfaz táctil, y dispositivos de cobro como detectores de monedas o billetes o sistema de lectura RFID.

Actualmente podemos encontrar soluciones integradoras que su precio sobrepasa dos mil dólares americanos.

Estudio de Mercado:

Target

- Centros de fotocopiado e impresión que deseen ampliar su horario de funcionamiento.
- Funcionabilidad de manera independiente como un stand en un centro comercial o en una facultad o juzgado (lugares donde se realizan muchas impresiones) o un lugar aislado teniendo como target a todo usuario que desee imprimir en cualquier horario del día.
  - Pruebas de concepto. ¿Es un producto útil?
- Este producto nace como solución a un problema, que son los estudiantes que muchas veces imprimen en un lugar lento, caro, y poco cómodo debido a la cantidad que debe imprimir y los horarios en los que necesita usar este servicio.
  - Pruebas de producto. Competencia. ¿Qué hay en el mercado? ¿En qué se diferencia de lo que voy a hacer?
- Existen productos que en precio están alrededor de 2000 dólares, y que solo permiten un tipo de impresión.
- Este sistema se enfoca a que puede ser usado en cualquier sistema de impresión.
- La idea es proporcionar una solución que abarque distintos tipos de impresión y sobre todo accesible de precio.

Objetivos del proyecto

-Realizar encuestas para verificar la problemática actual al momento de realizar impresiones.

-Desarrollar una solución utilizando elementos que se consigan en el mercado local.

-Implementar una solución de precio accesible.

Elementos necesarios

-Raspberry pi 3

-Pantalla táctil compatible con raspberry pi

-Selector de monedas

-Impresora

-Algoritmos para interactuar con el usuario

-Interfaz gráfica

Capítulo 2: Desarrollo

Para poder entender el enfoque y todas las partes estudiadas y analizadas en este proyecto, presento a continuación un diagrama en bloques del sistema en general.

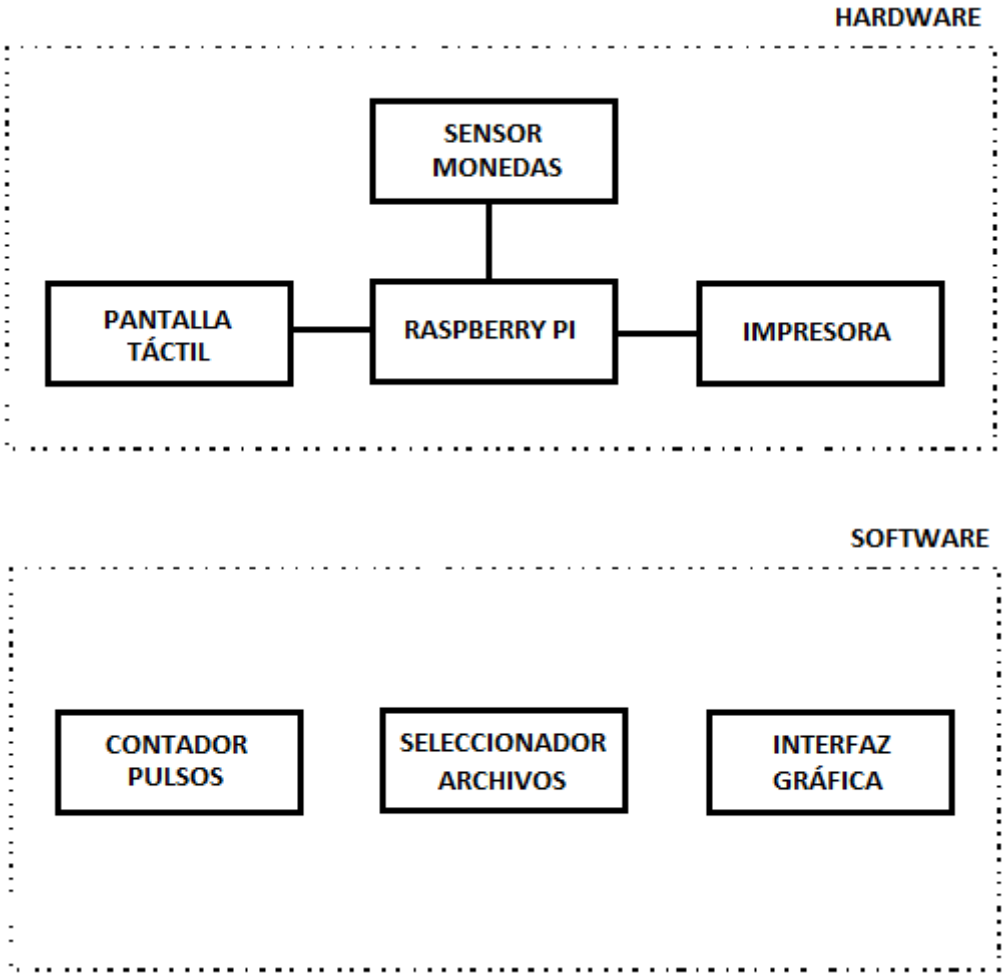


Ilustración 1 Diagrama General

Para poder realizar este sistema, se utilizará una computadora de tamaño de una tarjeta de crédito (raspberry pi) y una pantalla táctil para hacer la interfaz hombre-maquina.
Se colocará un sensor de monedas para realizar el cobro de la impresión.
Al mismo tiempo se lo conectará a una impresora A4.
Después de todas las configuraciones necesarias, es necesario contar con algoritmos que puedan interpretar la salida del sensor de monedas y también analizar la cantidad de hojas de un archivo para calcular el precio de la impresión.
Una interfaz grafica que le permita al usuario realizar la transacción independientemente.

Se detalla a continuación cada uno de los bloques expresados en el diagrama general.

Raspberry pi

Una raspberry pi es una computadora que dispone de las siguientes características técnicas:

-Broadcom BCM2837B0, Cortex-A53 (ARMv8) SoC de 64 bits a 1.4GHz
-SDRAM LPDDR2 de 1 GB
-LAN inalámbrica IEEE 802.11.b / g / n / ac de 2.4GHz y 5GHz, Bluetooth 4.2, BLE
-Gigabit Ethernet sobre USB 2.0 (rendimiento máximo 300 Mbps)
-Cabecera GPIO extendida de 40 pines
-HDMI de tamaño completo
-4 puertos USB 2.0
-Puerto de cámara CSI para conectar una cámara Raspberry Pi
-Puerto de pantalla DSI para conectar una pantalla táctil Raspberry Pi
-Salida estéreo de 4 polos y puerto de video compuesto
-Puerto Micro SD para cargar su sistema operativo y almacenar datos
-Entrada de alimentación de 5V / 2.5A DC
-Compatibilidad con alimentación por Ethernet (PoE) (requiere un PoE HAT separado)

El objetivo de usar una computadora (en este caso una raspberry pi) es simplemente por la factibilidad de disponer de un sistema operativo (Linux), que nos facilita los drivers para comunicación por USB o conectividad inalámbrica.

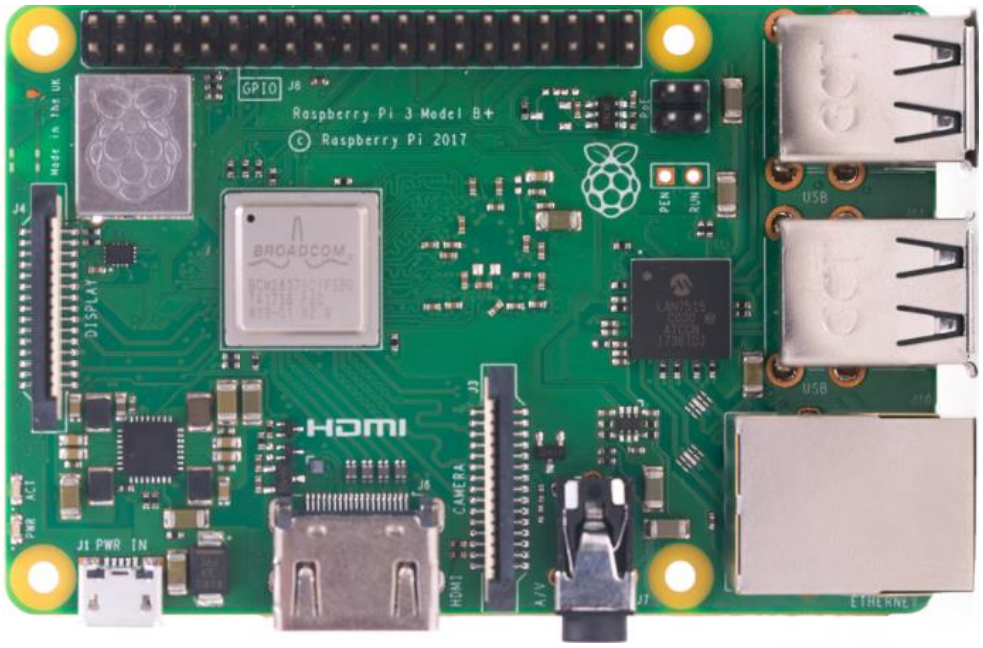


Ilustración 2 Raspberry Pi

Banana Pi M3

-Allwinner A83T ARM Cortex-A7 Octa-Core 1.8 GHz
- PowerVR SGX544MP1
- 2 GB LPDDR3
- 8G eMMC flash integrado, ranura MicroSD, puerto SATA 2.0 (puente de USB a SATA
- 10/100/1000 Mbit/s Ethernet
- Wi-Fi 802.11 b/g/n
- Bluetooth 4.0
- Puerto HDMI y salida de audio multicanal / MIPI DSI para LCD
- 3.5mm jack and HDMI
- 2 x puertos USB 2.0
- GPIO (x28)

Ilustración 3 Banana Pi M3

Los computadores del tamaño de una tarjeta de crédito presentadas son alternativas que se han estudiado para implementar esta solución.
Si bien ambas poseen características similares, se ha optado por usar una Raspberry Pi, debido a que es un elemento que se encuentra fácil en el mercado en caso de necesitar un reemplazo.

Se ha analizado la posibilidad de armar una placa de esquema ordenador desde cero, sin embargo, el bajo coste y la alta funcionalidad de estas tarjetas (raspberry pi, banana pi) desestima totalmente la inversión de tiempo y dinero que implicaría hacer la placa desde cero.

Una vez que se ha conseguido una raspberry pi, es necesario la instalación de un sistema operativo que soporte la arquitectura ARM linux, las opciones son las siguientes:

-Raspbian
-Ubuntu MATE
-Pidora
-Arch Linux ARM

La opción es utilizar el sistema operativo Raspbian, perteneciente a la distribución Debian por ser el sistema operativo más conocido.

La raspberry pi es el eje de este proyecto, debido a que es la interfaz física a donde todos los elementos irán conectados, y al mismo tiempo, es donde se encuentran los algoritmos que hemos mencionado anteriormente.

La experiencia con la configuración y puesta en marcha de una raspberry pi ha sido un camino de aprendizaje sobre todo acerca del sistema operativo linux, manejo de comandos de consola y sobre todo aprendizaje sobre el lenguaje mayormente utilizado para automatizar procesos en este sistema operativo: bash.

Pantalla Táctil

Existen distintas alternativas de pantallas para la raspberry pi, no obstante en esta aplicación necesitamos de una pantalla táctil, en la cual podemos se encuentra:

Ilustración 4 Pantalla táctil de 7 pulgadas

Ilustración 5 Pantalla Táctil

Este LCD de pantalla táctil Raspberry Pi de 7" brinda la capacidad de crear un dispositivo independiente que puede utilizarse como una tableta personalizada o una interfaz interactiva todo en uno para un proyecto con la Raspberry Pi 3. LCD de Pantalla táctil capacitiva de 800 x 480 que se conecta a la Raspberry Pi a través de una placa adaptadora incluida que maneja todas sus necesidades de conversión de energía y señal.

Lo que hace que esta pantalla LCD sea excelente es el hecho de que solo requiere que se conecten dos conexiones a la Raspberry Pi; alimentación desde el puerto GPIO del Pi y un cable plano que se conecta al puerto DSI presente en todos los Raspberry Pi.

Ilustración 6 Controladora pantalla táctil

Ilustración 7 Conexión pantalla táctil

La conexión y puesta en marcha es bastante simple, sin embargo se encontró un problema al momento de encenderlo, puesto que se ha comprado una carcasa para proteger la pantalla, esta carcasa tiene una orientación específica para poder apoyarla.
Cuando se encendió la pantalla, la imagen estaba al revés, para poder corregir esto fue necesario seguir los siguientes pasos:

A través de la consola de linux ejecutar el comando

sudo nano /boot/config.txt

colocar la línea

lcd_rotate=2

De la siguiente manera:

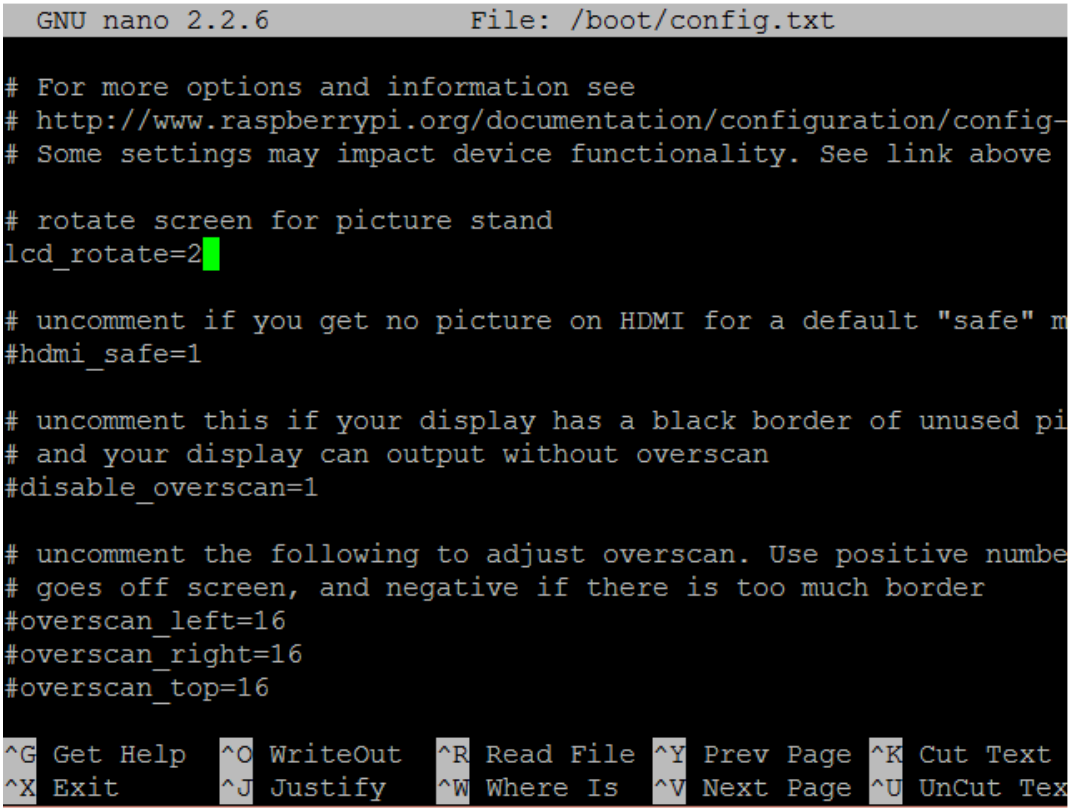


Ilustración 8 Configuración rotación pantalla

La pantalla nos permite tener una interfaz de interacción con la raspberry pi.
Sin embargo, la pantalla táctil esta pensada para la interacción con el usuario, para programar y poder utilizar la raspberry pi para realizar el desarrollo se utiliza un monitor convencional con su conexión HDMI.

Sensor De Monedas

Como se ha mencionado anteriormente, para que este pueda ser un sistema totalmente independiente de cualquier intermediario humano, se busca un dispositivo que pueda detectar el ingreso de monedas y pueda enviar una señal que pueda ser interpretada por la rpi.

Se utiliza el sensor CH926 que tiene las siguientes características:

- 12 V DC
- 15mm a 29 mm diámetro de monedas permitidas.
- 1.8mm a 3.0 mm espesor de monedas permitidas.
- 10 C a 60 C temperatura de trabajo.

Ilustración 9 Sensor de monedas

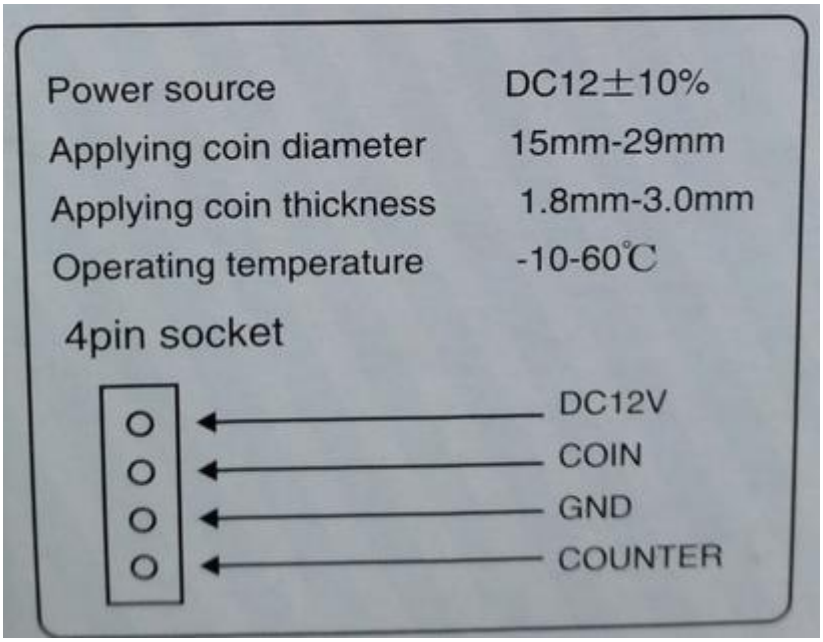


Ilustración 10 Especificaciones sensor de monedas

El sensor CH926 es una familia de dispositivos destinados a reconocer monedas, para usarse en máquinas expendedoras o máquinas recreativas. Las variantes difieren en el número de tipos de

monedas que pueden reconocer (3, 5, 6 u 8 tipos para CH923 / JY923, CH925 / JY925, CH926 / JY926 o CH928 / JY928, respectivamente). Tienen diseños y embalajes variables.

La documentación de tal sensor para poder configurarlo correctamente es:

Manual of CH–926

CH–926 is a multi coin selector , can accept up to 6 kinds of different coins at the same time. This type of coin selector is widely used in Vending machine , Arcade Game , Message chair , and other self – management system .CH–926 is mainly based on material , weight and size to identify coins. We use the most up to date algorithm to design software ,There fore CH–926 is very stable and accurate even when environment changes . such as temperature , humidity etc··· In order to increase the accuracy , we suggest different version of coins use different channel to set up .

Specifications

Coin diameter ： 15mm~32mm	Atmospheric pressure ： 86Kpa—106Kpa
Coin thickness ： 1.2mm~3.8mm	Working humidity ： ≤95%
Working voltage ： DC +12V ± 10%	Speed ： ≤0.6秒
Working current ： 65mA ± 5%	Accuracy rate of identification ： 99.5%
Signal output ： pulse	

Features

a. Capable of accepting all worldwide Coins and Tokens.
b. Intelligent CPU software control, and high accuracy .
c. Self–programming without PC.
d. Accept 1~6 different kind of coins at the same time.
e. Free to set up pulses' output.
f. Prevent not only electric shock but also electromagnetic interference.
g.Automatic self–test for problems.

The Process of Setup for Parameters

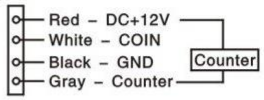
1. Press the "Add" and "minus" buttons at the same time for about three seconds, then the letter "A" will appear from the LED display.
2. Press the "setup" button once, and the letter "E" will appear. Next, use the buttons to choose how many kinds of coins you would like to use; then press the "setup" button again to finish.
3. The letter "H" will appear after pressing the button. Use the "Add" and "minus" buttons to choose how many samples you would like to insert later. Next press the "setup" button again to finish.
4. The letter "P" will appear after pressing the button. Use the "Add" and "minus" buttons to choose the amount of output's signals/pulses you want. The quantity limited is 50 times. Next, press the "setup" button to finish.
5. The letter "F" will appear after pressing the button. Use the "Add" and "minus" buttons to choose accuracy. The value is from 1~30, and 1 is the most accurate. Normally, 5~10 will be fine.Next, press the "setup" button to finish.
6. So far, you have successfully set up the first coin. please repeat all above procedures until you have set up all the coins. The letter "A" will appear again after all above procedures are finished.
7. Press the "setup" button, and the letter "E" will appear. Finally, turn off and turn on the power. The setup will be stored.

You can start sampling after the setup is finished. Please choose at least 20 coins. The sampling process will affect the accuracy of coin selector.

Sampling

1. Press the "setup" button, then letter "A" will appear from the LED display.
2. Press the "setup" button again, then letter "A1" will appear. Next, start to insert sample coins. The LED display will show how many coins you insert. The letter "A1" will appear again after finished.
3. Press the "setup" button again, then the letter "A2" will appear. Next, Start to sample 2nd coin, and repeat No. 1 and No. 2 until all the coins are set up .
4. After finished the sampling, press the "setup" button. The letter "A"will appear, then turn off and turn on the power. Now you can start to use it.

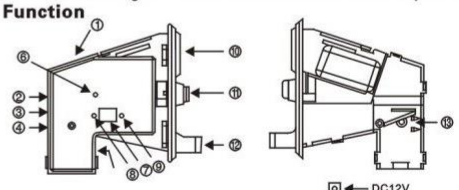
Connections' manual

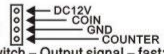


FAQ:

A. Coins can't pass through
   1. Check if the connection is correct or poor connection.
   2. Check if sampling is appropriate.
   3. Check if there is something stuck inside.
B. Can't identify outputs
   1. Check if 2 way switches –NO. NC.– are correct.
   2. Check if Singal wire is connected correctly .
   3. Check if there is pull–up resistor with signal wire(not allowed).
   4. Check if there is something stuck in the channel where the coins pass out.
C. Coins can not pass smoothly
   1. Check if the parameter is set up appropriately.
   2. Check the accuracy switch
   3. Check if there is something stuck in the channel where the coins pass out.

Name and Function





① LED light – The light of indicidation.
② 4pin Socket
③ Two way switch□signal output NO. or NC.
④ Three way switch – Output signal – fast: 20ms , medium: 50ms , slow: 100ms
⑤ 2pin Socket – Electromagnetic valve DC 12V
⑥ "Set up" button
⑦ LED display
⑧ "Add" button – Plus "+" value
⑨ "Minus "button – Minus "–" value
⑩ Coin slot
⑪ Press–button for removal of blocked coin
⑫ Return slot of Coin
⑬ Position of electromagnetic strobe

Ilustración 11 Documentación sensor de monedas

Este dispositivo tiene un controlador incorporado, el cual debe ser configurado para detectar el tipo de moneda que se desea.

Este sensor ha sido configurado con monedas de dólar estadounidense.

La configuración se ejecuta de la siguiente manera:

- Se configura el número de tipos de monedas para reconocer y, para cada una de ellas, cuántas muestras se utilizarán para aprender ese tipo de monedas, el número de pulsos que se generarán cuando se detecte ese tipo de monedas y la precisión del filtro para eso tipo de moneda (1 = restrictiva, 30 = suelta).

- Alimentar al sensor de monedas con tantas muestras como haya definido anteriormente para cada tipo de moneda. Se recomienda utilizar un mínimo de 15 monedas por cada tipo para cubrir todas las variaciones. El dispositivo acepta hasta 30 muestras por tipo.

Aunque la documentación anterior dice que la corriente de trabajo es de 65 mA, el pico al detectar una moneda que pasa es muy superior a 100 mA, así la fuente de alimentación debe ser capaz de proveer hasta 1A.

Operación

Una vez configurado, para cada moneda reconocida, el dispositivo produce un tren de pulsos <n> en la línea “COIN”, <n> que depende del tipo de moneda configurado en el paso 1 anterior. El cuarto cable (gris) está etiquetado como "CONTADOR" en el dispositivo, sin embargo esto no representa una funcionalidad para este desarrollo.

Ahora se presenta una eventualidad, la salida de la línea “COIN” tiene una salida de 5V, esto no es aceptado en la rpi ya que podría quemar el dispositivo, por lo que se optó por una solución del tipo adaptador de tensión.

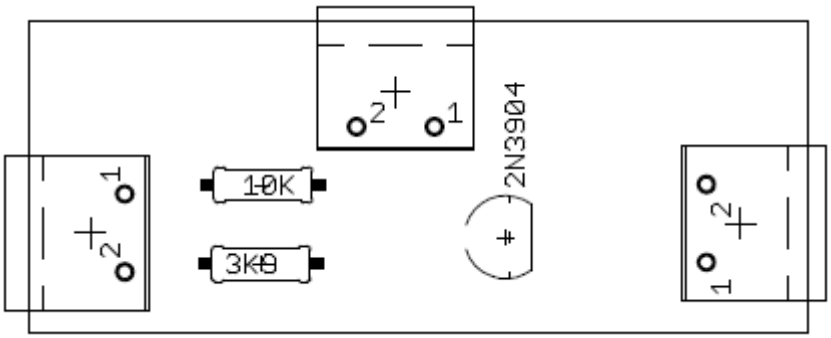


Ilustración 12 Adaptador de tensión.

El adaptador de tensión consta de un transistor de uso general 2N3904 conectado en modo corte/saturación.

Ilustración 13 Sensor

Usando un interruptor en la parte posterior, se puede elegir el estado "en reposo" de la línea COIN para que esté flotando (normalmente abierto, NO) o configurado en GND (normalmente cerrado, NC).

Cada pulso COIN puede tener una longitud diferente de acuerdo con el otro interruptor en la parte posterior. Las duraciones observadas son las siguientes:

Rápido 20ms / pulso

Medio 50ms / pulso

Lento 70ms / pulso

Cada pulso es seguido por una pausa de 100 ms en estado inactivo (sin importar la configuración de velocidad seleccionada). Aquí hay algunos ejemplos para ilustración:

Ilustración 14 Moneda configurada para 4 pulsos, rápido.

Ilustración 15 Moneda configurada para 1 pulso, medio.

Ilustración 16 Moneda configurada para 1 pulso, lento.

Se encuentra un problema al insertar dos monedas inmediatamente una después de la otra.

Ilustración 17 Problema de inserción rápida.

Como se observa, no hay pausa entre las monedas, por lo que no hay forma de dividir el tren de pulsos con seguridad.

Lo que significa es que, si configuramos valores arbitrarios a las monedas, como, por ejemplo, 1,2,3,4,5,6, entonces 2 monedas de valor 3 insertadas rápidamente no se pueden distinguir de 1 moneda de valor 6.

La única forma de resolver el problema de la "inserción rápida" es olvidarse de determinar las "monedas" y centrarse en el "valor", lo que significa elegir un número de pulsos que sea proporcional al valor nominal de la moneda. Pero si se elige una unidad demasiado pequeña (por ejemplo, 1 pulso / centavo), una moneda de $ 2 activará 200 pulsos, lo que no es práctico (el tren durará al menos 24 segundos) Pero también imposible ya que el dispositivo está limitado a 50 pulsos por tipo.

En consecuencia, con este dispositivo, es imposible cubrir de manera confiable un rango de monedas donde el valor más grande es> = 100 veces el más pequeño. Por ejemplo, uno nunca cubrirá de manera confiable el rango completo de USD de 0.01 a 1 USD.

En la documentación también se habla de un "modo AP" que parece actuar como un divisor y solo emite un pulso después de que se haya alcanzado un umbral de <n> "pulsos internos" (como se configuró con el ajuste P). Eso reduce la cantidad y la longitud de los trenes de pulso que tiene que manejar, pero, por supuesto, no cambia fundamentalmente la limitación de alcance.

La solución sensible es cubrir un rango reducido y dar a un pulso el valor del divisor común más grande de todas las monedas admitidas. Por ejemplo, en este proyecto se cubre 0.05 a 1 USD y asigna 1 pulso a 5 centavos.

El inconveniente es que los valores grandes tardan una cantidad considerable de tiempo en decodificarse sin embargo es algo menor con respecto al anterior problema que no se podía identificar correctamente las monedas.

El CH926 es un dispositivo muy confiable, pero requiere un diseño de software cuidadoso para evitar sus dificultades, será detallado más adelante en la sección de “contador de pulsos”.

Impresora

Para poder imprimir desde un sistema operativo linux tenemos que utilizar CUPS.

CUPS (acrónimo de Common UNIX Printing System) es un sistema de impresión modular para sistemas operativos de computadora tipo Unix que permite que una computadora actúe como un servidor de impresión. Una computadora que ejecuta CUPS es un host que puede aceptar trabajos de impresión desde computadoras cliente, procesarlos y enviarlos a la impresora adecuada.

CUPS consta de una cola de impresión y un planificador, un sistema de filtro que convierte los datos de impresión a un formato que la impresora comprenderá y un sistema de fondo que envía estos datos al dispositivo de impresión. CUPS utiliza el Protocolo de impresión de Internet (IPP) como base para administrar colas y trabajos de impresión. También proporciona las interfaces de línea de comando tradicionales para los sistemas de impresión System V y Berkeley, y brinda soporte para el protocolo Line Printer Daemon del sistema de impresión Berkeley y soporte limitado para el protocolo de bloque de mensajes del servidor (SMB). Los administradores del sistema pueden configurar los controladores de dispositivo que suministra CUPS editando archivos de texto en formato de descripción de impresora PostScript (PPD) de Adobe. Hay varias interfaces de usuario para diferentes plataformas que pueden configurar CUPS, y tiene una interfaz basada en web incorporada. CUPS es un software gratuito, provisto bajo la Licencia Apache.

CUPS proporciona un mecanismo que permite enviar trabajos de impresión a impresoras de manera estándar. Los datos de impresión van a un planificador que envía trabajos a un sistema de filtro que convierte el trabajo de impresión en un formato que la impresora comprenderá. El sistema de filtro luego pasa los datos a un back-end, un filtro especial que envía datos de impresión a un dispositivo o conexión de red. [9] El sistema hace un uso extensivo de PostScript y rasterización de datos para convertir los datos a un formato adecuado para la impresora de destino.

CUPS ofrece un sistema de impresión estándar y modular que puede procesar numerosos formatos de datos en el servidor de impresión. Antes de CUPS, era difícil encontrar un sistema estándar de administración de impresoras que pudiera acomodar a la gran variedad de impresoras en el mercado utilizando sus propios lenguajes y formatos de impresora. Por ejemplo, los sistemas de impresión System V y Berkeley eran en gran medida incompatibles entre sí, y requerían scripts complicados y soluciones alternativas para convertir el formato de datos del programa a un formato imprimible. A menudo no podían detectar el formato de archivo que se estaba enviando a la impresora y, por lo tanto, no podían convertir el flujo de datos de forma automática y correcta. Además, la conversión de datos se realizó en estaciones de trabajo individuales en lugar de en un servidor central.

CUPS permite a los fabricantes de impresoras y desarrolladores de controladores de impresoras crear más fácilmente controladores que funcionan de forma nativa en el servidor de impresión. El procesamiento se realiza en el servidor, lo que permite una impresión basada en red más fácil que con otros sistemas de impresión Unix.

El planificador CUPS implementa el Protocolo de impresión de Internet (IPP) sobre HTTP. Una aplicación auxiliar (cups-lpd) convierte las solicitudes del protocolo Line Printer Daemon (LPD) a IPP. El programador también proporciona una interfaz basada en web para administrar trabajos de impresión, la configuración del servidor y la documentación sobre CUPS en sí.

Un módulo de autorización controla qué mensajes IPP y HTTP pueden pasar a través del sistema. Una vez que los paquetes IPP / HTTP están autorizados, se envían al módulo del cliente, que escucha y procesa las conexiones entrantes. El módulo del cliente también es responsable de ejecutar programas CGI externos según sea necesario para admitir impresoras, clases y monitoreo y administración del estado del trabajo basados en la web. Una vez que este módulo ha procesado sus solicitudes, las envía al módulo IPP que realiza la validación del Identificador Uniforme de Recursos (URI) para evitar que un cliente eluda cualquier control de acceso o autenticación en el servidor HTTP. El URI es una cadena de texto que indica un nombre o dirección que puede usarse para referirse a un recurso abstracto o físico en una red.

El planificador permite clases de impresoras. Las aplicaciones pueden enviar solicitudes a grupos de impresoras en una clase, lo que permite al programador dirigir el trabajo a la primera impresora disponible en esa clase. Un módulo de trabajos gestiona trabajos de impresión, enviándolos a los procesos de filtro y backend para la conversión final y la impresión, y supervisando los mensajes de estado de esos procesos.

El planificador CUPS utiliza un módulo de configuración, que analiza los archivos de configuración, inicializa las estructuras de datos CUPS e inicia y detiene el programa CUPS. El módulo de configuración detendrá los servicios CUPS durante el procesamiento del archivo de configuración y luego reiniciará el servicio cuando se complete el procesamiento.

Un módulo de registro maneja el registro de eventos del planificador para acceso, error y archivos de registro de página. El módulo principal maneja los tiempos de espera y el envío de solicitudes de

E / S para las conexiones de los clientes, vigila las señales, maneja los errores y las salidas del proceso secundario y recarga los archivos de configuración del servidor según sea necesario.

Es de esta forma que los drivers de la impresora que es elegida se maneja a través de CUPS.

La configuración se realiza a través del explorador de la siguiente forma:

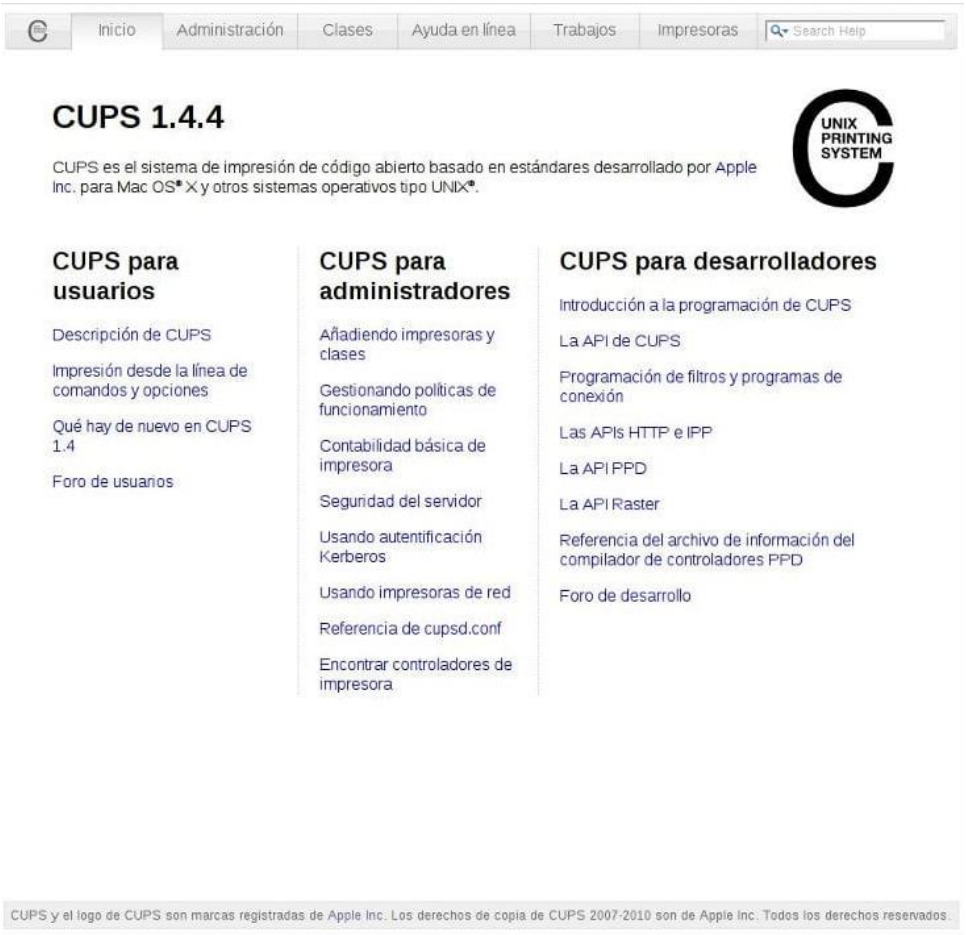


Ilustración 18 Configuración CUPS

Ilustración 19 Añadir Impresora

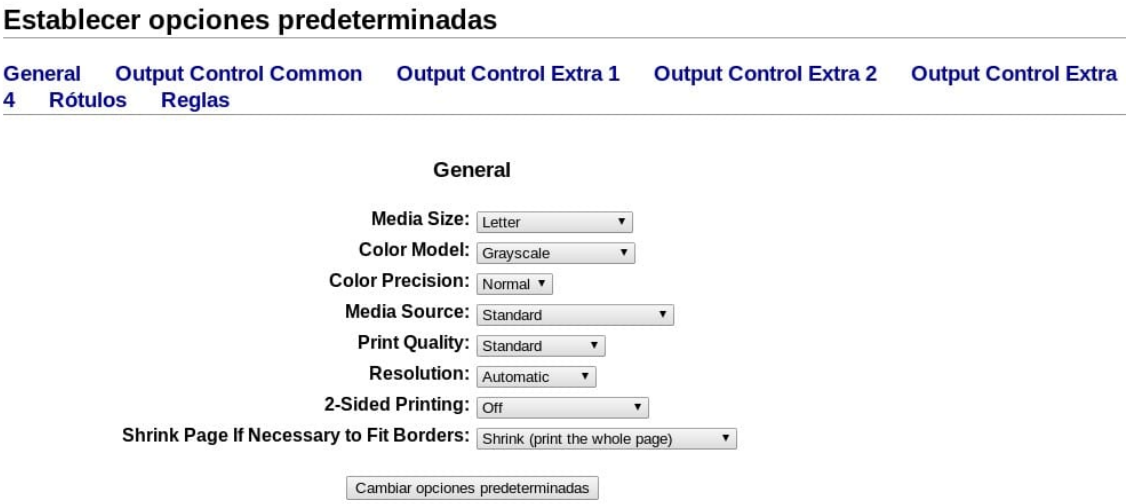


Ilustración 20 Configuración Impresora

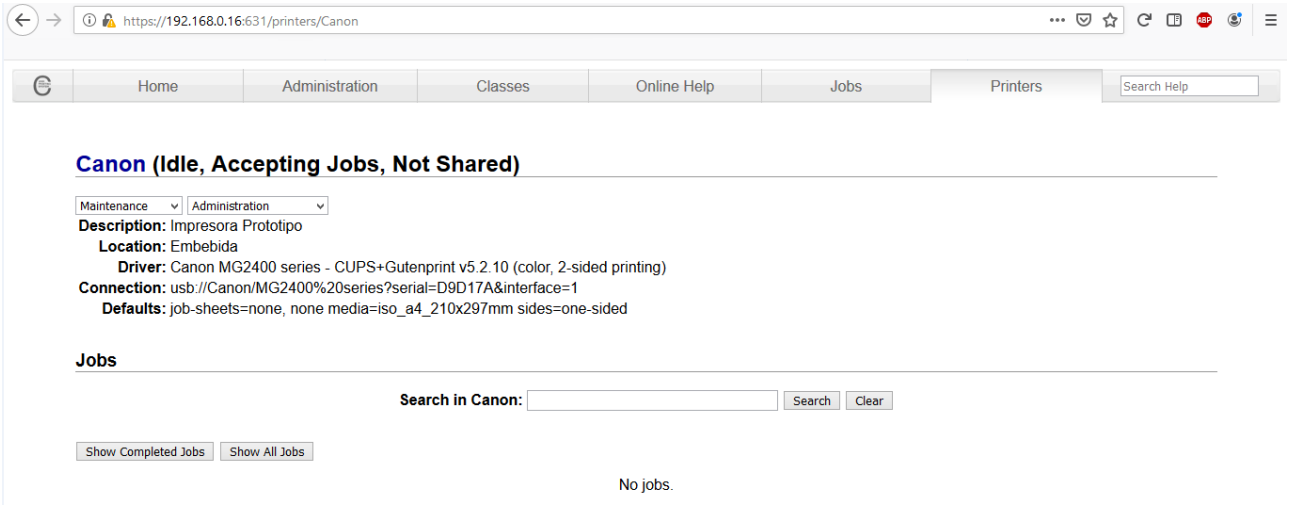


Ilustración 21 Impresora Configurada

Una vez que tenemos la impresora configurada podemos mandar a imprimir desde cualquier editor. Sin embargo, el objetivo de este desarrollo es enviar a imprimir archivos una vez que se ha cobrado el valor total, por lo que es necesario que podamos enviar a imprimir con algún comando de consola.

Los comandos utilizados se detallan a continuación como parte de los algoritmos implementados.

Contador De Pulsos

El sensor de monedas va a generar <n> pulsos con cada inserción de monedas.
Se ha configurado el sensor de tal forma que genere 1 pulso por cada 5 centavos de dólar.
De esta forma se tiene:

5ctvs 1 pulso

10ctvs 2 pulsos

25ctvs 5 pulsos

Se ha mencionado que el sensor de monedas ira conectado a la rpi a través de un adaptador de tensión, debido a que la salida del sensor son 5 V y la entrada de los GPIO de la rpi es 3.3 V.

La problemática ahora es encontrar un método efectivo para poder leer los puertos GPIO de la rpi. Según la ley de Nyquist, para poder muestrear una señal, se debe poseer una frecuencia de muestreo según la siguiente formula:

$$fm=2*fs$$

Donde:

fm = frecuencia de muestreo.

fs = frecuencia de la señal.

Y considerando que el sensor de monedas va a estar emitiendo señales de un período: 10 ms.

Por lo tanto, la frecuencia de la señal sería 1 Khz.

En consecuencia, por el teorema de Nyquist la frecuencia de muestreo debería ser de al menos 2Khz.

En la práctica, se entiende que este valor es el mínimo valor para poder representar una señal. Sin embargo, se recomienda, para una mejor interpretación de la señal, que la frecuencia de muestreo sea 10, 20 o incluso 100 veces mayor que la frecuencia de la señal.

Se puede leer los puertos GPIO de la raspberry pi con cualquier lenguaje de programación.

Sin embargo, ante la necesidad especifica de disponer al menos 100Khz de frecuencia de muestreo, se hace una comparación entre los distintos lenguajes, las librerías utilizadas y la frecuencia de muestreo en cada uno de los casos.

Language	Library	Tested / version	Square wave
Shell	/proc/mem access	2015-02-14	2.8 kHz
Shell / gpio utility	WiringPi gpio utility	2015-02-15 / 2.25	40 Hz
Python	RPi.GPIO	2015-02-15 / 0.5.10	70 kHz
Python	wiringpi2 bindings	2015-02-15 / latest github	28 kHz
Ruby	wiringpi bindings	2015-02-15 / latest gem (1.1.0)	21 kHz
C	Native library	2015-02-15 / latest RaspPi wiki code	22 MHz
C	BCM 2835	2015-02-15 / 1.38	5.4 MHz
C	wiringPi	2015-02-15 / 2.25	4.1 – 4.6 MHz
Perl	BCM 2835	2015-02-15 / 1.9	48 kHz

Ilustración 22 Velocidad lectura puertos según lenguaje

Dada esta perspectiva, podríamos programar el algoritmo de contador de pulsos con Python o con el lenguaje C.

Dentro de las posibilidades de programación en lenguaje C, se puede utilizar una librería nativa de C, programar directamente los registros del microprocesador BCM 2835 o utilizar la librería WiringPi.

WiringPi es una librería de acceso GPIO basada en PIN escrita en C para los dispositivos BCM2835, BCM2836 y BCM2837 utilizados en todas las Raspberry Pi. Se lanzó bajo la licencia GNU LGPLv3 y se puede usar desde C, C ++ y RTB (BASIC), así como en muchos otros idiomas con envoltorios adecuados. Está diseñado para que las personas que hayan usado el sistema de "cableado" Arduino lo conozcan y son familiares. para uso de programadores experimentados de C / C ++.

Entonces debido a la experiencia previa con el lenguaje C y la similitud con una librería ampliamente utilizada con la placa Arduino, se ha tomado la decisión de implementar el algoritmo con el lenguaje C y con el uso de la librería WiringPi.

Se ha desarrollado el siguiente algoritmo:

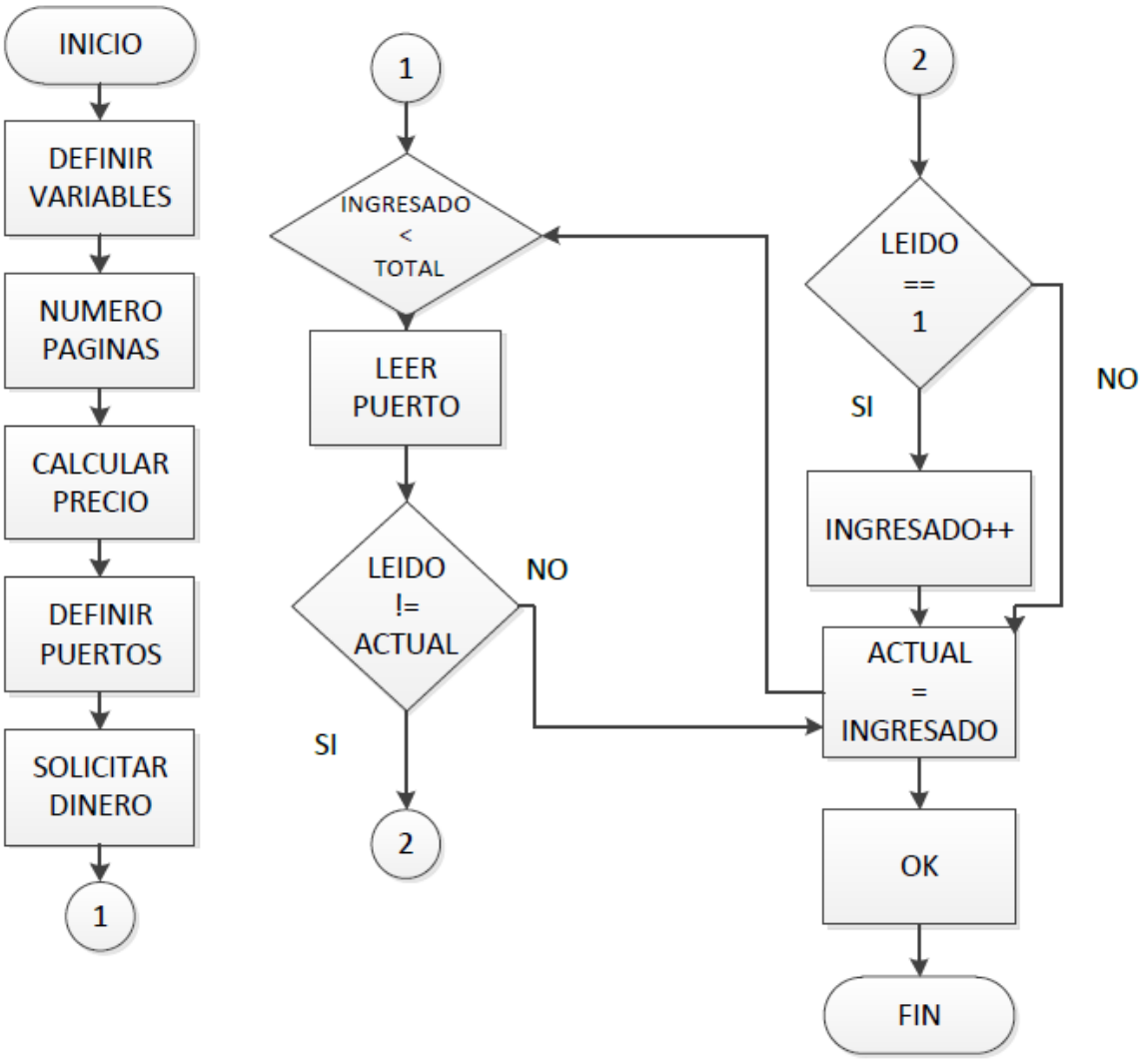


Ilustración 23 Diagrama Flujo Contador de Pulsos

```
#include <stdio.h>
#include <stdlib.h>
#include <wiringPi.h>
#define COIN 27
#define INDICADOR  17
#define ENCENDIDO 04

int main(){
char name[6];
int pag,costo;
FILE *ifp;
ifp=fopen("/home/pi/Documents/vending/beta/datos.txt","r");
fscanf(ifp,"%s %d",name,&pag);
fclose(ifp);
costo=1*pag;//Aca se cambia el valor de la impresion por pagina.
float dinero= costo*0.05;
system("clear");
printf("Coin Acceptor\n");
printf("Por favor espere un momento\n");
wiringPiSetupGpio ();
pinMode(COIN,INPUT);
pinMode(INDICADOR,OUTPUT);
pinMode(ENCENDIDO,OUTPUT);
digitalWrite(ENCENDIDO,1);
delay(2000);
digitalWrite(INDICADOR,1);
int b=0,x=0,p=0;
int moneda=0;
system("clear");
printf("Ud va a imprimir %d paginas\n ",pag);
printf("Ingrese $%.2f centavos\n",dinero);

while(p<costo)
{
delay(1);
x=digitalRead(COIN);
if (x!=b)
        {

                if(x==1)

                {

                p=p+1;

                system("clear");

printf("****************************************2019*************************\n")
;

                printf("Ud ha insertado %d ctvs en total\n",moneda);
```

```
                }
        }
b=x;
}

digitalWrite(INDICADOR,0);
digitalWrite(ENCENDIDO,0);
printf("Se ha ingresado el monto correcto \n");
return 0;}
```

Explicación del algoritmo desarrollado:

En primera instancia se definen las librerías que se utilizan:

```
#include <stdio.h>
#include <stdlib.h>
#include <wiringPi.h>
```

La distribución de los GPIO en la rpi es la siguiente:

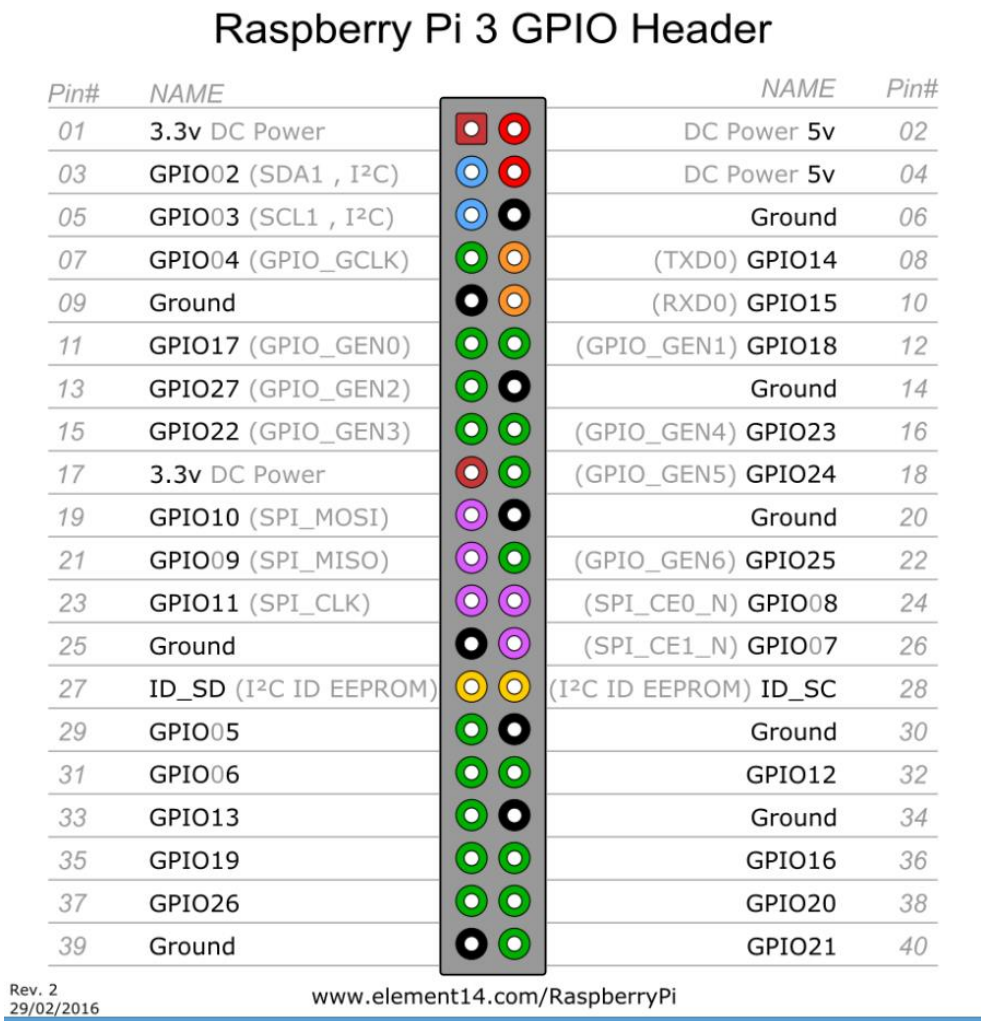


Ilustración 24 Distribución puertos raspberry pi

Entonces procedemos a describir los puertos que se van a utilizar:

```
#define COIN 27
#define INDICADOR  17
#define ENCENDIDO 04
```

El puerto 27 como lector de monedas, el puerto 17 como un indicador LED de ingreso de monedas y el puerto 04 como indicador de encendido.
Es una buena práctica asignar a los puertos un valor único con el parámetro DEFINE para escalabilidad del producto.

Para poder calcular cuanto es el valor que debe ingresar el usuario, se hace uso de un archivo de texto llamado datos.txt el cual será explicado en el siguiente algoritmo “seleccionador de archivos”

En esta porción del código se obtiene la cantidad de paginas a imprimir para poder calcular el costo de la impresión.

```
ifp=fopen("/home/pi/Documents/vending/beta/datos.txt","r");
fscanf(ifp,"%s %d",name,&pag);
fclose(ifp);
costo=1*pag;//Aca se cambia el valor de la impresión por página.
float dinero= costo*0.05;
system("clear");
```

A continuación, se definen los puertos como entrada o salida y se le alerta al usuario cual es la cantidad que debe ingresar:

```
printf("Coin Acceptor\n");
printf("Por favor espere un momento\n");
wiringPiSetupGpio ();
pinMode(COIN,INPUT);
pinMode(INDICADOR,OUTPUT);
pinMode(ENCENDIDO,OUTPUT);
digitalWrite(ENCENDIDO,1);
delay(2000);
digitalWrite(INDICADOR,1);
int b=0,x=0,p=0;
int moneda=0;
system("clear");
printf("Ud va a imprimir %d paginas\n ",pag);
printf("Ingrese $%.2f centavos\n",dinero);
```

Entonces es el momento de contar los pulsos, para eso se ingresa en una estructura WHILE en la cual la condición de salida es cuando el usuario a ingresado la totalidad del valor monetario de la impresión.

```
while(p<costo)
{
delay(1);
x=digitalRead(COIN);
if (x!=b)
        {

                if(x==1)
```

```
                {
                p=p+1;
                system("clear");
printf("****************************************2019*************************\n")
;
                printf("Ud ha insertado %d ctvs en total\n",moneda);
                }
        }
b=x;
}
```

Lo que hace esta parte del código es verificar el estado del puerto por cada ciclo de while, esta armado de tal forma que detecte cuando se generen cambios en el estado(alto, bajo) del pulso generado por el sensor de monedas.

Por ultimo se le notifica al usuario que el monto ingresado ha sido el correcto y se puede imprimir el archivo (también se explica en el siguiente algoritmo seleccionador de archivos).

Un punto interesante es la siguiente línea:

```
delay(1);
```

Un retraso de 1 milisegundo dentro de la estructura WHILE, debido a la rapidez de lectura que estamos utilizando.
Antes de que el algoritmo este terminado con esta línea, se experimentaron varios errores y fue motivo de varias modificaciones en la lógica.
La explicación es la velocidad de lectura utilizada.

Con esta sección se ha explorado como interpretar las señales proporcionadas por el sensor de monedas.
Una vez que la información ha pasado de la parte de hardware a un software, hemos dado un paso importante en este desarrollo pues a partir de este momento, lo que falta es una interacción a través de software con la impresora y se puede lograr el objetivo.

Seleccionador de Archivos

Con este modulo se busca crear una interfaz que permita seleccionar un archivo que ha sido ingresado mediante un dispositivo de almacenamiento USB, y brindar la información correspondiente al código de contador de pulsos para que pueda realizar el calculo del precio de la impresión en función de la cantidad de hojas a imprimir.

Para este desarrollo utilizamos un lenguaje de scripting BASH debido a su facilidad para interactuar con el sistema operativo linux.

```
#!/bin/bash
#validar si el script no esta ya en ejecución
```

```
if [ -e ~/Documents/vending/beta/pdi/txt ]
then
exit

Fi
echo hola > ~/Documents/vending/beta/pdi.txt
let f=0
let b=0
while [ $f -eq 0 ]
do
archivo=$(zenity --file-selection --title="Seleccione un archivo a imprimir" --filename=/media/pi/
2> /dev/null )
if [ ! -e "$archivo" ]
then
exit
fi
validar=$(head -c 4 "$archivo")
if [ $validar = "%PDF" ]
then
f=1
else
zenity --info --title="ADVERTENCIA" --text="Por favor seleccione un archivo PDF" 2> /dev/null
fi


if [ $f -eq 1  ]
then
zenity --question --text " Desea imprimir $archivo " 2> /dev/null
        if [  $? -eq 1 ]
        then
        f=0
        fi

fi

done
pdfinfo "$archivo" | grep Pages > ~/Documents/vending/beta/datos.txt

~/Documents/vending/beta/ejecutable

rm ~/Documents/vending/beta/datos.txt

rm ~/Documents/vending/beta/pid.txt

lp "$archivo"
```

En la primera parte de este código se define el lenguaje SHELL a utilizar y también se realiza una validación que no exista otra instancia de este programa en ejecución ya que podría alterar los valores de cantidad de paginas y el costo de la impresión.

```
#!/bin/bash
#validar si el script no esta ya en ejecución
if [ -e ~/Documents/vending/beta/pdi/txt ]
then
exit
```

A continuación se define una estructura WHILE en donde haciendo uso de la herramienta ZENITY podemos generar una interfaz grafica en donde el usuario puede explorar por el sistema de archivos y elegir un documento a imprimir, en este caso, ingresado por un dispositivo de almacenamiento USB.

```
while [ $f -eq 0 ]
do
archivo=$(zenity --file-selection --title="Seleccione un archivo a imprimir" --filename=/media/pi/
2> /dev/null )
if [ ! -e "$archivo" ]
then
exit
fi
validar=$(head -c 4 "$archivo")
if [ $validar = "%PDF" ]
then
f=1
else
zenity --info --title="ADVERTENCIA" --text="Por favor seleccione un archivo PDF" 2> /dev/null
fi


if [ $f -eq 1  ]
then
zenity --question --text " Desea imprimir $archivo " 2> /dev/null
        if [  $? -eq 1 ]
        then
        f=0
        fi

fi

done
```

Para este Desarrollo particular se ha limitado únicamente a archivos PDF, es por eso que existe una validación en el algoritmo del tipo de archivo.

Se hace uso de la herramienta ZENITY que permite crear cuadros de dialogo, ideal para la interacción con el usuario.

Por ultimo se hace uso del comando PDFINFO para obtener la cantidad de paginas totales del documento y poder realizar el calculo del precio de la impresión.

pdfinfo "$archivo" | grep Pages > ~/Documents/vending/beta/datos.txt

Se ejecuta el programa de contador de pulsos, ya compilado previamente.

~/Documents/vending/beta/ejecutable

Se borran los archivos temporales creados para este programa únicamente.

rm ~/Documents/vending/beta/datos.txt

rm ~/Documents/vending/beta/pid.txt

Con este comando imprimimos, después de la configuración del servidor CUPS como se detallo previamente.

lp "$archivo"

Diseño Completo

Ilustración 25 Producto Final

De esta manera se han evaluado cada uno de los módulos necesarios para poder obtener el producto final.

Un sistema embebido de impresión, totalmente independiente de un intermediario humano, que puede efectuar cobros y que puede funcionar las 24 horas del Dia.

Se evalúa entonces el comportamiento entre todos los módulos juntos.

La raspberry pi es la encargada de la conexión de los sensores, del procesamiento de datos y el manejo de las interfaces de comunicación.

La pantalla táctil nos permite interactuar con el producto.

El sensor de monedas nos permite independizarnos de un intermediario humano para el momento del cobro de la impresión.

El desarrollo ha implicado desarrollo de software en lenguaje C, BASH, un manejo del sistema operativo linux avanzado para poder entender las instrucciones de consola que se usan para cada caso particular, investigación acerca del servidor CUPS, realización de una placa PCB de adaptador de tensión y en general manejar circuitería y cableado, y mucha investigación para poder ensamblar piezas que se encuentran en el mercado, no obstante no existe un desarrollo similar utilizando estos componentes sencillos de conseguir.

Capítulo 3: Resultados

En primer lugar, para demostrar la aceptación de este producto, se presentan los resultados de las encuestas en donde se evalúa cual es la problemática al momento de imprimir.

Las encuestas fueron realizadas a 64 estudiantes de arquitectura.

En primer lugar, se desea estudiar la respuesta del publico en general para verificar que realmente se trata o no de una problemática existente.
El objetivo de un emprendimiento es proporcionar una solución a un problema existente.

Verificamos que el 77% de encuestados presentan problemas al momento de imprimir.

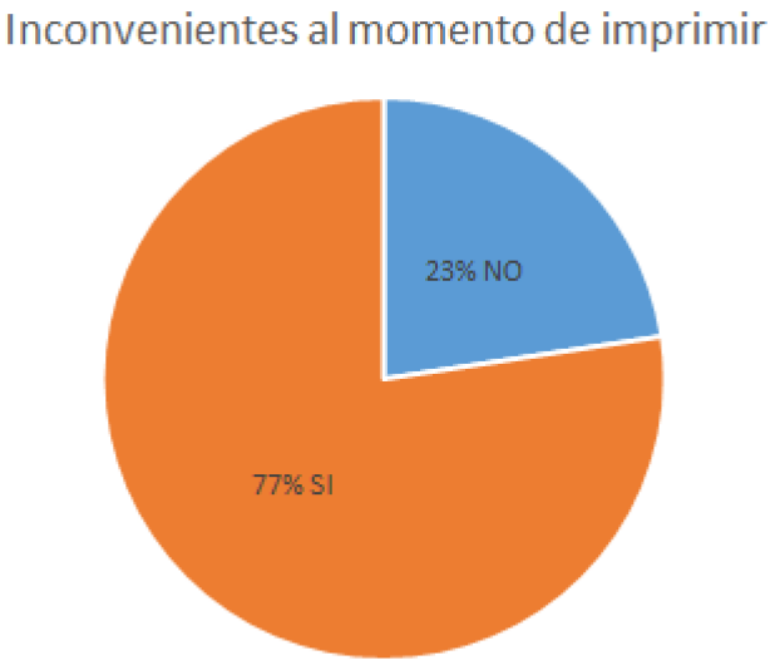


Ilustración 26 Resultado Encuesta 1

Se busca saber cuál es la problemática al momento de imprimir, se realiza una pregunta abierta para entender todas las posibilidades.

Problemas	
Calidad	54%
Precio	46%
Incomodidad	35%
Tiempo	25%
Formato	12%
Calidad & Precio	31%
Calidad y Tiempo	19%
Precio y Tiempo	12%
Calidad Precio Tiempo	2%

Ilustración 27 Resultado Encuesta 2

Se observa que los encuestados responden que la calidad de sus impresiones no siempre es la mejor, y que el precio no es el correcto.

Se consulta además la cantidad de impresiones por semana por alumno, para entender cuál sería el mercado y que tan fácil seria generar ganancias con el nuevo producto desarrollado.

Ilustración 28 Resultado Encuesta 3

Por ultimo se ha preguntado el porque elige imprimir en el lugar donde imprime, y la respuesta demuestra que a pesar de los problemas que puedan llegar a tener, en la mayor parte del tiempo van a imprimir al lugar que tienen más cerca.

Ilustración 29 Resultado Encuesta 4

Los resultados de la encuesta nos sirven como fundamento para el desarrollo del prototipo.

Ilustración 30 Resultado final

Se obtiene finalmente un producto que puede estar al servicio de los usuarios las 24 horas del día. Cada uno de los módulos fue probado y los resultados fueron satisfactorios.

Las pruebas de impresión han resultado satisfactorias con un tiempo de respuesta entre que se finaliza el proceso de cobro y se imprime de 20 segundos.

Capítulo 4: Análisis de Costos

Los costos de los materiales utilizados en el proyecto se presentan en la siguiente tabla

Raspberry Pi	3600
Pantalla Tactil	4200
Sensor Monedas	4200
Desarrollo placa potencia	2000
total	14000

Tabla 1 Costo Materiales

El costo de mano de obra se calcula según el tiempo invertido en este desarrollo que fue de 300 horas, calculando a $400/hora, se obtiene un total de: $120000 pesos argentinos.

Por lo que el costo total del prototipo es: $134000 pesos argentinos.

Una producción en serie se podría realizar en 8 horas, por lo cual el costo por unidad sería: $17200 pesos argentinos.

Si se vendiese en $30000 pesos argentinos, la amortización del costo del diseño se lograría al venderse 10 unidades.

Capítulo 5: Discusión y Conclusión.

En este proyecto hemos vistos los aspectos necesarios para disponer de un prototipo, que puede ser analizado para ser producido en serie.
Si bien en el enfoque de este desarrollo fue únicamente para archivos pdf en una impresora A4, la escalabilidad y futuras mejoras a este producto son amplias.

El algoritmo de contador de pulsos del selector de monedas junto con la interfaz grafica se puede reutilizar para cualquier tipo de impresión, con unas modificaciones al código para comunicarse con impresoras 3D, impresoras de planos o impresoras de fotografías.

En el mercado actualmente se encuentran soluciones integradas y no personalizables por un precio superior a los $70000 pesos argentinos.

Por lo tanto, el resultado es satisfactorio tanto en desarrollo a un precio accesible y además porque responde a una problemática real con la que se encuentran estudiantes y usuarios en general al momento de acudir a un centro de impresión.

Este producto, puede estar empotrado a la pared de los negocios de impresión, el cual estaría disponible las 24 horas del día y así ampliaría el margen de ganancia del negocio y a la gente le brinda un servicio de alta disponibilidad.

El estudio de este prototipo abre las puertas a futuras mejores como por ejemplo, una comunicación inalámbrica entre usuario y producto, nuevos dispositivos de cobro como

sensores de billetes, lectores rfid, lectores de huellas digitales, agregar una cámara para monitoreo y conectar el producto a la red para fácil mantenimiento.

Capítulo 6: Literatura Citada.

- Documentación Raspberry Pi
  https://www.raspberrypi.org
- Documentación Banana Pi
  http://www.banana-pi.org
- Sweet, Michael (June 9, 1999). "A Bright New Future for Printing on Linux". Linux Today. Archived from the original on October 5, 2007.
- Lavry, Dan (2 de enero de 2012). «Sampling Theory For Digital Audio» (en inglés). Lavry Engineering, Inc. Consultado el 6 de abril de 2017.
- "GNU Bash". Softpedia. SoftNews. Retrieved April 9, 2016.
- Documentación WiringPi
  http://www.wiringpi.com/