Universidad de la República
Facultad de Ingeniería

Proyecto Lapix

Mauro Di Leonardi
Pablo Iguini
Diego Viganó

Tutor: Juan Pechiar

20 de octubre de 2010

Agradecimientos

A nuestras familias y amigos, Juan Pechiar, José Luis Vila, Tabaré Aviega, Santiago Reyes, Fiorella Haim, Isabel Amigo, Ismael Schinca, Miguel Barreto, Mariano Cebey, Fernando Silveira, Leonardo Etcheverry, Cecilia San Román, Matias Poloni, Gerardo Robert, los docentes del curso Taller Encararé donde surge el proyecto y al Instituto de Ingeniería Eléctrica.

Resumen

En este documento se describen los elementos que constituyen el Proyecto Lapix. Se relata el origen del proyecto, así como los objetivos buscados y su alcance. Se explica la tecnología utilizada y se le da un panorama general al lector acerca de qué se trata el proyecto. Luego se explica en detalle todo lo referente al hardware desarrollado así como el software generado, en donde se incluyen técnicas de filtrado de ruido. También se explicitan las características obtenidas, incluyendo el consumo de los dispositivos. Se cuentan varios de los problemas enfrentados a lo largo del proyecto, se fijan pautas de como seguir una vez terminado el proyecto y se realiza un análisis general del resultado obtenido.

En este proyecto se construye un prototipo funcional de un lápiz electrónico inalámbrico que permite escribir y dibujar sobre las pantallas de las XO (computadoras del plan Ceibal), o cualquier superficie lisa, y que además funciona como un sustituto del mouse. Puede utilizarse en cualquier PC con sistema operativo Linux y es portable a Windows.

Cuenta con un lápiz electrónico inalámbrico, un dispositivo de recepción, un driver y un acrílico protector. El lápiz electrónico emite señales desde la punta. Dispone de 2 botones, uno primario, otro secundario y tiene un interruptor de encendido. El receptor se encarga de detectar las señales emitidas por el lápiz, procesarlas y comunicarse con la PC anfitriona a través del puerto USB. El Driver es el software instalado en la PC que se encarga de recibir la información enviada por el receptor, procesarla y realizar el despliegue correspondiente en pantalla.

Contenido

Capítulo 1

Introducción

1.1. Origen del Proyecto

La idea de este proyecto de fin de carrera surge en un curso del segundo semestre del 2008, llamado Taller Encararé. El mismo es dictado por el Instituto de Ingeniería Eléctrica de la Facultad de Ingeniería.

La dinámica del curso consiste en asignarle un área de trabajo a un grupo de estudiantes. Estos deberán detectar problemáticas de la vida real asociadas con su área asignada y encontrarles una solución ingenieril viable. Con la solución encontrada se desarrolla un Plan de Proyecto y un Plan de Negocios para llevar la misma al ámbito productivo.

El área asignada en dicha ocasión fue el Plan Ceibal, que se encontraba en pleno auge en nuestro país. Uno de los problemas detectados, fue un mal funcionamiento del touchpad de las computadoras XO, entorpeciendo su uso y dificultando un ágil aprendizaje.

A partir de esto, surgió la idea de elaborar una herramienta que cuente con las funcionalidades de un mouse y a su vez, transforme la XO en un cuaderno digital en el que se puede escribir y dibujar directamente de forma electrónica.

1.2. Objetivos y Alcance

En el proyecto Lapix se busca realizar un prototipo funcional de un lápiz electrónico inalámbrico para utilizar, en principio, con las computadoras del

Plan Ceibal pero que pueda ser portable a otras máquinas, generando para ello el software adecuado.

La idea es que el dispositivo pueda utilizarse para escribir y dibujar en la superficie de la pantalla de las computadoras como si fuera papel, reconociéndose los trazos realizados y desplegándolos instantáneamente de forma de generar la impresión de que realmente se está dibujando sobre ésta, pero que también sirva como mouse inalámbrico y que sea posible utilizarlo sobre otras superficies lisas.

El objetivo del proyecto será realizar un prototipo funcional que cumpla con las especificaciones de diseño y que permita observar y probar la correcta funcionalidad del mismo.

Lapix consta de un lápiz que emite señales que son recibidas y procesadas por un elemento receptor conectado a la computadora por medio del puerto USB, éste será el encargado de suministrar la información necesaria al driver que corre en la XO para que despliegue en pantalla el trazado y el pulsado de botones. Al conectar el receptor a la XO se despliega un software de calibración para coordinar la posición de la punta física con lo visto en pantalla.

Para esto, el receptor debe ser capaz de sujetarse a la pantalla y en caso de utilizarse sobre otras superficies lisas, debe poder ser fácilmente ubicado. El lápiz consta de un botón en la punta, el cual auspicia de botón primario del mouse, otro ubicado en un costado con la función de botón secundario y un interruptor con la función de encendido y apagado.

1.2.1. Especificaciones de diseño

Al inicio del proyecto, se establecieron las siguiente prioridades:

1. Que reproduzca el trazo hecho lo suficientemente bien y simultáneamente, como para que un niño pueda usarlo sin que afecte su caligrafía.
2. Minimizar el uso de recursos de las computadoras del Plan Ceibal.
3. Que sea adaptable a otros sistemas.
4. Que sea lo suficientemente simple para que un niño pueda utilizarlo sin problemas.

A su vez, se aspiraban las siguientes especificaciones:

- *Área de cobertura:* hasta el 90 % de una pantalla 15"(230 x 305mm)
- *Resolución:* 200 DPI (dots per inch = ppp puntos por pulgada)
- *Muestreo:* 75 muestras / segundo
- *Exactitud:* 0.5mm
- *Retraso:* El retraso se espera sea tal que genere la sensación de simultaneidad necesaria para que un niño usuario de la XO se sienta cómodo utilizando el lápiz sobre la pantalla.
- *Interferencia entre dispositivos:* Se ignorarán señales que se encuentren por fuera de un radio de 50cm del receptor.

Además, se buscaban las siguientes características adicionales deseables:

- Lograr implementarlo por medio de 1 o 2 puertos USB como máximo
- Lograr ejecutar el programa de calibración automáticamente al conectarlo a la PC.
- Poder realizar trazos simples, sobre todo en diagonales, es decir, que no se vean trazos escalonados.
- Lograr que la punta del lápiz no raye la pantalla con el uso normal de éste.
- Lograr que el lápiz tenga una autonomía suficiente para su uso diario.

1.3. Método y Tecnología Utilizada

El método que se utiliza para la detección de la punta, es el llamado "Método del Rayo Trueno" [1] [2] [3]. El mismo consiste en la emisión simultánea de dos señales: una señal "rápida" y una señal "lenta". La idea es que a la llegada de la señal "rápida" (rayo) se comienza a tomar el tiempo hasta el arribo de la señal "lenta" (trueno). Luego, sabiendo la velocidad de propagación de la señal lenta se puede calcular, despreciando el retardo de la señal rápida, la distancia desde la que fueron emitidas.

El mismo principio se aplica a Lapix. En nuestro caso consiste en emitir simultáneamente una señal de infrarrojo y otra de ultrasonido de forma omnidireccional desde la punta del lápiz. Las mismas son recibidas por el receptor que cuenta con un sensor receptor de infrarrojo y dos sensores receptores de ultrasonido separados una cierta distancia fija. Dado que la velocidad de propagación del sonido puede considerarse constante, y que la de infrarrojo es mucho mayor que ésta, se puede obtener la distancia a la que se encuentra el punto de emisión de las señales con respecto a cada uno de los sensores receptores de ultrasonido, contando desde la llegada de la señal de infrarrojo hasta el arribo de las de ultrasonido. Con estas dos distancias se puede obtener el punto de origen de las señales emitidas al interceptar las dos circunferencias cuyos radios son las distancias halladas y por ende, la posición de la punta del lápiz en el plano.

Realizando el proceso anteriormente descripto repetidamente se logra muestrear la posición de la punta, logrando obtener la representación del trazo realizado. Más adelante se deduce la frecuencia de muestreo a utilizar.

El análisis realizado para la detección de la punta se limita al plano dado que nuestra prioridad es que funcione correctamente sobre superficies. Al levantar el lápiz respecto de la superficie donde se encuentra el receptor se sigue detectando la posición de la punta pero sobre el plano imaginario que forman la punta y los dos receptores de ultrasonido, por lo que si nuestra superficie es la pantalla y la punta se levanta en forma perpendicular, en lugar de que el puntero permanezca fijo en pantalla, se ve como se aleja un poco ya que no se distingue si el emisor se encuentra en la superficie o en un plano más elevado. Otro factor importante para limitar el análisis es que los sensores receptores tienen directividad horizontal y a más de 2cm de elevación no se detecta correctamente en algunas zonas.

Para comunicar el estado de los botones, la información del pulsado de los mismos es transmitida por medio de una codificación en la señal de infrarrojo.

En el receptor, luego de los sensores, las señales obtenidas son convertidas a formato digital de forma de transmitir la información del pulsado de los botones y de las distancias obtenidas hacia la XO por el puerto USB, en donde el driver se encarga de filtrar y ubicar la posición de la punta del lápiz en pantalla. Para esto previamente se requiere de la ejecución de un software de calibración de forma de coordinar la posición física de la punta con la desplegada en pantalla.

Cabe destacar que, luego de una ardua investigación, se opta por este método en particular dado que, además de que la idea de fondo es bastante sencilla, tiene una implementación considerablemente económica, con el potencial de brindar una alta resolución y precisión en la detección de las señales y es el usado en los productos comerciales más parecidos a nuestro proyecto. Además se disponía de las hojas de datos de la punta emisora [4] y de los receptores de ultrasonido [5], las cuales incorporaban circuitos ejemplos los cuales podíamos tomar como punto de partida.

1.4. Productos Similares

De la investigación en patentes y productos similares, surge que existe una vasta cantidad de dispositivos orientados a la digitalización de la escritura. Estos dispositivos pueden clasificarse básicamente en dos tipos [6] : los que disponen de una tabla digitalizadora, y los que no.

Nuestro dispositivo cae en la segunda categoría y esta acompañado de una gran cantidad de otros similares. Sin embargo, existen diferencias sustanciales que permiten que se distinga del resto, constituyendo un producto nuevo e independiente.

A pesar de la alta similitud de Lapix con el Lápiz Duo (ver anexo F), nuestro proyecto ha sido diseñado para aprovechar las características de las computadoras del Plan Ceibal, principalmente en cuanto a que utilizan linux y sus pantallas pueden plegarse dandoles la forma y tamaño de un e-reader, permitiendo transformar a estas en un cuaderno electrónico con todas las ventajas que la tecnología ofrece de forma económica y simple.

Para ver una lista de proyectos, patentes y productos similares, ir al anexo “Patentes y Productos Similares” F

1.5. Geometría

En esta sección se expondrán los fundamentos de geometría para entender el funcionamiento de Lapix.

Dado que funciona con el método del rayo trueno, lo que se mide es el tiempo transcurrido desde que llega al receptor la señal de infrarrojo hasta que llega la de ultrasonido. La velocidad del sonido en el aire a una temperatura de 20ºC es de 343,42m/s y la velocidad del infrarrojo es la velocidad de la luz (3x10^{8}m/s). Dado que hay seis órdenes de diferencia entre una y otra velocidad, para los cálculos se considera que la señal de infrarrojo llega instantáneamente al receptor. De esta forma midiendo el tiempo transcurrido desde que llega la señal de infrarrojo hasta que llega la de ultrasonido, se obtiene un valor proporcional a la distancia desde la punta emisora a cada uno de los receptores de ultrasonido.

Vale aclarar que la velocidad del sonido varía con la temperatura, la presión y el tipo de material. En el caso del Lapix el material será siempre el aire. La variación de la velocidad del sonido con la presión es muy chica, pero es considerable la variación con la temperatura. En la ecuación 1.1 se muestra la relación entre la velocidad del sonido y la temperatura en el aire a presión constante.

$$Vs = Vo + BT \tag{1.1}$$

Donde:
Vo = 331,3m/s
B = 0,606m/sºC
T es la temperatura en ºC [7]

Para los cálculos se tomará como velocidad del sonido 343,42m/s (el valor de ésta a 20ºC)[1].
Llamando Vs a la velocidad del sonido, t al tiempo transcurrido entre la llegada de las dos señales y d a la distancia desde el emisor hasta el receptor

[1]Notar que con una variación de 20ºC en la temperatura ambiente se tiene un 3 % de variación en la velocidad de propagación. Esto se refleja en un error de hasta 1cm en la detección de la posición a una distancia de 30cm y de hasta 2mm a 5cm de distancia, por lo que se aconseja volver a calibrar si se presentan variaciones en la temperatura ambiente de más de 10ºC principalmente en caso de utilizarse sobre la pantalla.

de ultrasonido, la relación entre éstos está dada por la ecuación 1.2.

$$Vs = \frac{d}{t} \tag{1.2}$$

El tiempo t es medido mediante un contador con un clock de 80MHz. Llamando N a la cantidad de períodos de 80MHz, la relación ente la distancia d y N es:

$$Vs = \frac{d}{t} = \frac{d}{\frac{N}{80\times10^6 Hz}} \Rightarrow d = Vs \times \frac{N}{80 \times 10^6 Hz} \tag{1.3}$$

Se desprende entonces que la cantidad de períodos de reloj es proporcional a la distancia.

En lugar de trabajar con la distancia desde la punta emisora a los receptores de ultrasonido, se emplea un sistema de coordenadas rectangulares (x, y) que utiliza como unidad un ciclo de reloj de 80MHz, que facilita el posterior pasaje a pixels. El sistema de coordenadas rectangulares se eligió de forma que el punto $(0, 0)$ se encuentra en el punto medio de los receptores de ultrasonido, ambos receptores se encuentran sobre el eje x y el sentido de los ejes se definió como se muestra en la figura 1.1.

Dado que el Lapix funciona con dos receptores de ultrasonido, y que el receptor de ultrasonido mide el tiempo desde que llega la señal de infrarrojo, hasta que llega cada una de las señales de ultrasonido, para ubicar la punta del lápiz debemos hallar la intersección entre dos circunferencias, una con centro en el receptor de ultrasonido 1 y con radio la distancia entre él y la punta, y la otra circunferencia con centro en el receptor de ultrasonido 2 y radio la distancia entre él y la punta. Como se vio anteriormente, la relación entre la distancia, el tiempo y los períodos de reloj son proporcionales, por lo que da igual usar cualquiera de ellos como unidad. Se decidió utilizar los períodos de reloj de 80MHz.

La distancia entre los receptores de ultrasonido y el punto (0,0) del sistema de coordenadas elegido es de 4cm, o 9318 períodos de un reloj de 80MHz, a este valor se le llama N. Se llama $N1$ y $N2$ a las distancias desde la punta del emisor hasta los receptores de ultrasonido 1 y 2 respectivamente, medido

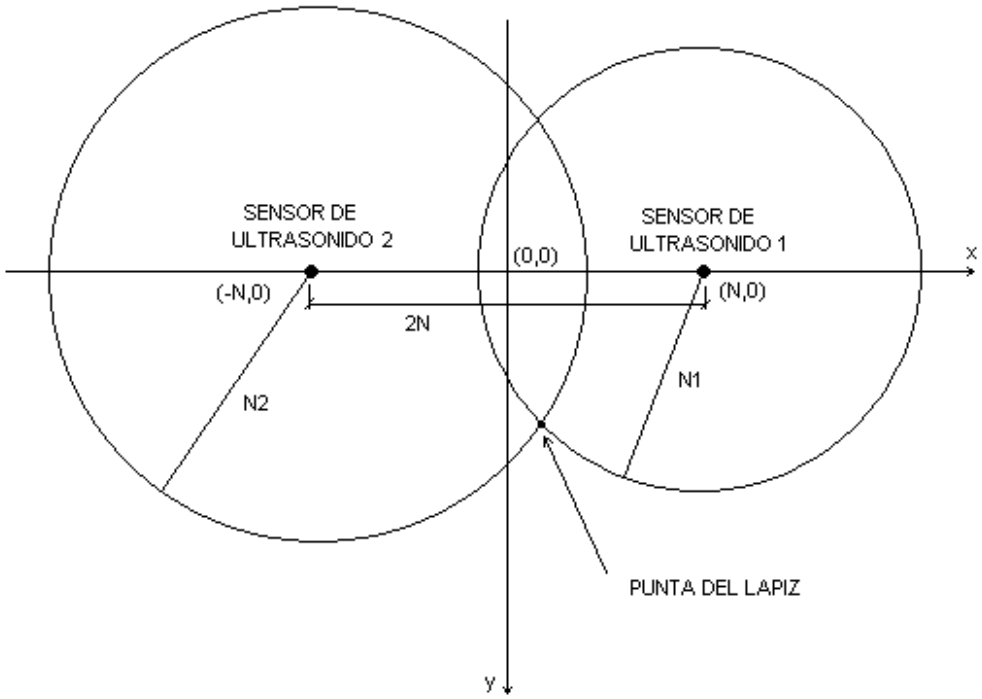

Figura 1.1: Pasaje a coordenadas rectangulares

también en períodos de reloj de 80MHz. Considerando que la punta emisora y los receptores se encuentran en un mismo plano paralelo a la pantalla del PC, en la intersección de las 2 circunferencias antes descritas hallamos la posición de la punta del lápiz. Como se muestra en la figura 1.1 se toma como punto de intersección el que tenga coordenada y positiva.

Para la representación del movimiento del puntero en la pantalla se debe pasar a un sistema de coordenadas rectangulares que utiliza como unidad el pixel. Para el pasaje del sistema de coordenadas que tiene como unidad la cantidad de períodos de reloj de 80MHz al que tiene como unidad el pixel se utilizó una transformación lineal. En la sección 3.4.4 se detalla como se implementó el cambio entre estos sistemas de coordenadas.

1.6. Diagrama de bloques

En esta sección se presenta una representación básica de la relación entre los distintos bloques que forman el proyecto y una breve explicación de la función de cada uno.

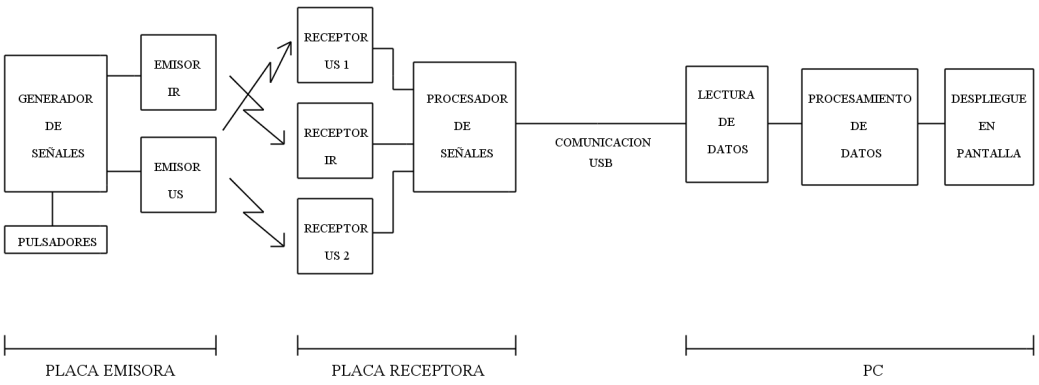

Figura 1.2: Diagrama de bloques

Generador de señales: es el bloque encargado de generar las señales de entrada a los circuitos emisores de infrarrojo y ultrasonido. Su elemento principal a nivel de hardware es un PIC12. Lee el estado de los pulsadores y lo codifica en la señal de infrarrojo.

Pulsadores: está formado por 2 pulsadores que implementan el click derecho e izquierdo, el bloque generador de señales se encarga de leer el estado de éstos.

Modulo emisor de infrarrojo: es el circuito encargado de generar la señal de infrarrojo, los emisores son 2 fotodiodos ubicados en la punta del lápiz. La señal de entrada al circuito es recibida del módulo generador de señales.

Modulo emisor de ultrasonido: es el circuito encargado de generar la señal de ultrasonido a emitir, la señal de entrada al circuito es recibida del módulo generador de señales.

Módulo receptor de infrarrojo: se encarga de generar a partir de la señal de infrarrojo recibida una señal eléctrica digital que vale 1 cuando

se recibe infrarrojo. Está formado por el sensor receptor de infrarrojo y su circuito de filtrado.

Módulos receptores de ultrasonido: los módulos receptores de ultrasonido son 2, idénticos entre sí. Cada uno de ellos está formado por el sensor receptor de ultrasonido, su circuito de filtrado correspondiente y el circuito para digitalización de la señal. A la salida de cada uno de los módulos receptores de ultrasonido se tiene una señal digital que vale 0 cuando se recibe ultrasonido y 1 el resto del tiempo.

Procesador de señales: se encarga de medir el tiempo desde que llega la señal de infrarrojo hasta que llega cada una de las señales de ultrasonido. Además se encarga de decodificar a partir de la señal de infrarrojo qué pulsador está presionado. También implementa la interfaz USB para enviar los tiempos medidos y estado de los pulsadores hacia el PC. La inteligencia de este bloque radica sobre un PIC32.

Comunicación USB: es el nexo entre la placa receptora y el PC. Es la vía de comunicación por la cual el procesador de señales envía al PC los tiempos y estado de los pulsadores mencionados en el párrafo anterior. Físicamente está formado por el cable USB.

Lectura de datos: es el encargado de gestionar la comunicación USB y la lectura de datos enviados por la placa receptora a través de esta interfaz. Implementado completamente a nivel de software.

Procesamiento de datos: este bloque es el encargado de llevar los datos con las medidas de tiempo recibidas de la placa receptora en datos que el PC pueda entender para poder desplegar en pantalla los movimientos de la punta del lápiz, el click derecho y el click izquierdo. En este bloque se incluye el software de calibración encargado de coordinar la posición de la punta física del lápiz con lo visto en pantalla.

Despliegue en pantalla: este bloque se encarga de desplegar todos los eventos de mouse en pantalla. Los eventos son movimiento del cursor, click derecho presionado y click izquierdo presionado. Estos eventos se actualizan cada vez que se recibe información desde la placa receptora.

Capítulo 2

Hardware

2.1. Emisor

La placa emisora no es otra cosa más que el lápiz y por este motivo es que se trató de hacer lo más sencilla posible de forma de disminuir la cantidad de componentes y obtener un tamaño adecuado. De este modo y como nuestro objetivo es poder detectar electrónicamente la posición real de la punta del lápiz sobre la pantalla de una PC o sobre cualquier otra superficie lisa, es que el lápiz se limita a emitir señales para detectar su posición y enviar la información correspondiente al pulsado de los botones que le dan la funcionalidad de mouse.

Para la emisión de las señales de infrarrojo y ultrasonido, utilizadas para la detección de su posicionamiento, se utiliza la punta emisora PT80KHZ-01 [4] que posee dos LEDs infrarrojos y un dispositivo piezoeléctrico ubicados de tal forma que le permite emitir ambos tipos de señales de manera omnidireccional.

Esta punta fue elegida principalmente porque fue la única encontrada comercialmente en la búsqueda inicial y se lograron adquirir varias copias de ésta como muestras gratis, cortesía de Measurement Specialties [8]. Además, como puede verse en la imagen 2.1, la misma está diseñada para lápices electrónicos y hace pareja con los receptores de ultrasonido SR80KHZ-01 [5] que también fueron adquiridos en dicha instancia de forma gratuita.

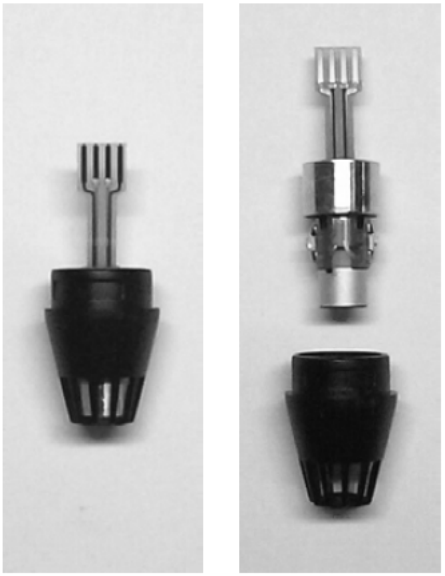

Figura 2.1: Punta emisora utilizada. Fuente hoja de datos de PT80KHZ-01 [4].

2.1.1. Circuito Diseñado

El esquemático del circuito emisor puede verse en la figura 2.2.
Los archivos Eagle de este esquemático pueden encontrarse dentro del CD en CD/circuitos/esquematicos y layouts/.

La idea de este circuito es que sea de bajo consumo, pueda ser alimentado con cualquier fuente de continua entre 5V y 10V, y cuente con la menor cantidad de componentes posibles para minimizar el espacio ya que va a ir dentro de un lápiz.

Su función es la de generar y emitir las señales periódicas de ubicación así como las correspondientes al pulsado de los botones.

La punta emisora de ultrasonido e infrarrojo PT80KHZ-01 se conecta a este circuito a través de un flex (ver figura 2.1). Para ello, se seleccionó el conector HLW4S-2C7LF [9]. Dentro del mercado local no pudo encontrarse uno que se ajustara a las dimensiones del flex, por lo que se tuvo que importar y éste conector fue el único que se encontró que parecía tener las dimensiones adecuadas. Sin embargo, a pesar de que funciona, para tener un buen contacto fue necesario agregarle un pequeño trozo de papel ya que el

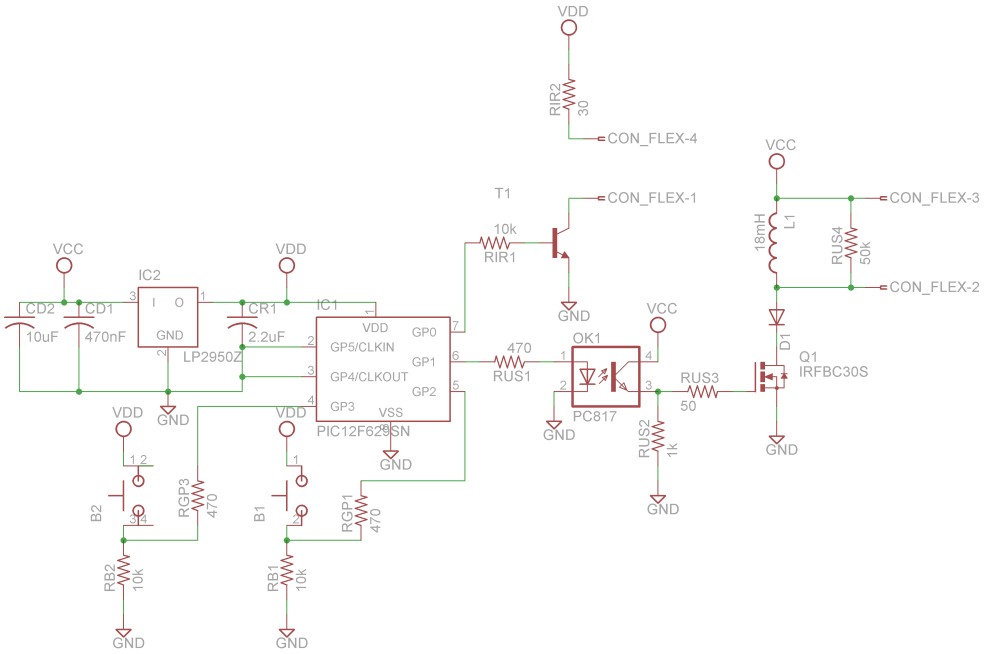

Figura 2.2: Esquemático del circuito emisor

grosor del flex es menor que el del conector.
Por lo que un trabajo a futuro debería ser seleccionar otro que se ajuste mejor.

El circuito emisor se divide en un bloque de emisión de infrarrojo, uno de emisión de ultrasonido y en otro correspondiente a la generación de las señales y la lógica del circuito del cual se encarga un microcontrolador PIC12.

2.1.2. Detalles del Circuito

Emisión de Infrarrojo

Al comienzo se partió del circuito ejemplo de la hoja de datos de la punta emisora pero fue descartado porque no era compatible con nuestras necesidades. De esta forma se optó por la utilización de un circuito más sencillo con un transistor NPN como se muestra en la figura 2.3.
Por más referencias ir a la sección 7.1.1.

El circuito se diseñó de manera de que por los LEDs pasara una corriente de 20mA con una caída de voltaje de 1.2 V en cada uno. Dichos valores

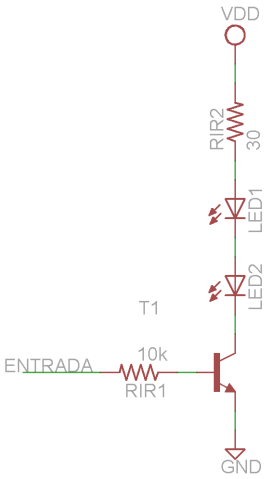

Figura 2.3: Bloque emisor de infrarrojo

fueron extraídos de la hoja de datos de la punta emisora y corresponden a un funcionamiento típico. Por la sencillez y pocos requerimientos de este circuito con un transistor NPN común como el BC547 [10] o el MMBT3904 [11] en el caso de montage superficial, es más que suficiente ya que ambos presentan una ganancia grande en corriente de aproximadamente $\beta = 200A/A$ para una corriente por el colector de 20mA como es nuestro caso. Además, son fáciles de conseguir y de bajo costo.

Hallando la corriente por los LEDs asumiendo un funcionamiento continuo en saturación, con una alimentación del circuito de $V_{DD} = 3{,}3\text{V}$, se llega a que:

$$I_D = \frac{(V_{DD} - V_{CESat} - 2V_{DON})}{R_{IR2}}$$

donde V_{CESat} es aproximadamente 0.3V y V_{DON} es la caída de 1.2V en cada LED. Observar que I_D es sensible a pequeñas variaciones en V_{DON} y V_{CESat}.

Despejando se obtiene que:

$$R_{IR2} = 30\Omega$$

Para trabajar en saturación, se debe cumplir que $I_C << \beta I_B$, de esto se obtiene que $I_B > 10I_C/\beta = 1\text{mA}$.

Entonces, con un voltaje de entrada de 3.3V impuesto por el PIC12 y dado que estos transistores poseen un $V_{BE} \cong 0{,}7\text{V}$ se llega a que:

$$\frac{(3{,}3V - V_{BE})}{R_{IR1}} > 1\text{mA}$$

De donde se despeja que:

$$R_{IR1} < 2{,}6\text{k}\Omega$$

Sin embargo, para reducir el consumo del circuito, se decidió aumentar el valor de R_{IR1} a 10kΩ, de forma que el transistor se mantenga en zona activa y la corriente tomada del PIC12 sea menor. Con esto, y dado que el PIC12 suministra 3.3V en la entrada del circuito, la corriente de base al emitir señal se reduce aproximadamente a:

$$I_B = \frac{(3{,}3V - 0{,}7V)}{R_{IR1}} = 260\mu\text{A}$$

Experimentalmente se vio que esos valores de resistencias son los óptimos.

Es de interés destacar que se requiere una alimentación de al menos 3.0V para que el circuito funcione y para valores mayores, basta con modificar el valor de la resistencia R_{IR2} de forma que la corriente siga fija en 20mA.

Emisión de Ultrasonido

En este caso, dado que se constató un buen funcionamiento del circuito sugerido en la hoja de datos de la punta emisora (ver figura 2.4), se optó por utilizarlo.

Según la hoja de datos, el transductor ultrasónico requiere una señal excitadora de alto voltaje de pico a aproximadamente 85kHz para lograr los umbrales de recepción requeridos, con una forma particular como la que se muestra en la figura 2.5.

Ahora se analizará el circuito para corroborar y entender cómo es que genera esta señal.

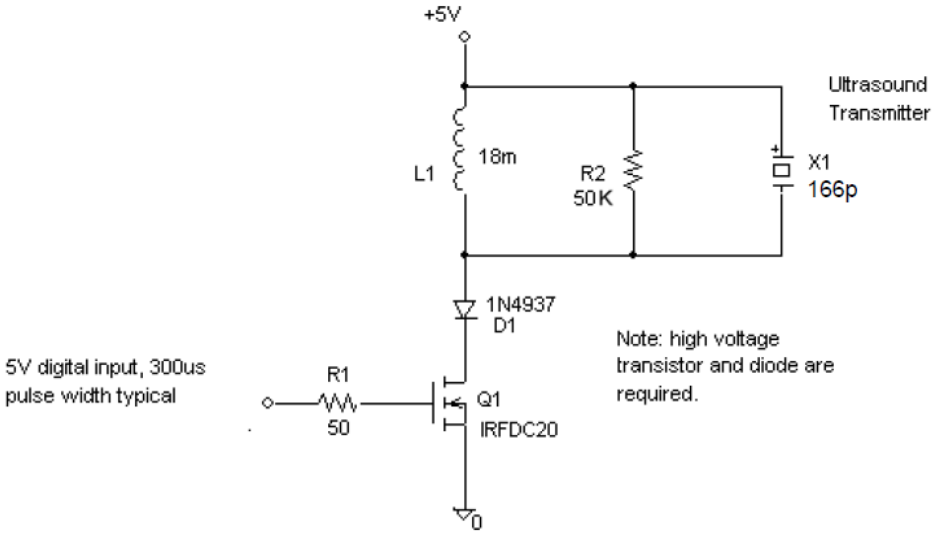

Figura 2.4: Circuito ejemplo de hoja de datos del PT80KHZ-01 [4]

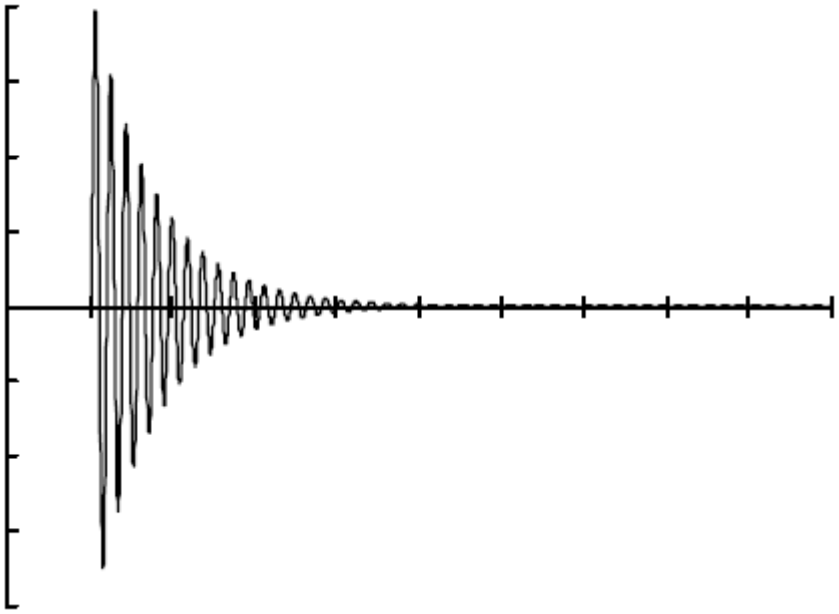

Figura 2.5: Señal excitadora - Eje x 50μs/div - Eje y 100V/div - Fuente: hoja de datos del PT80KHZ-01 [4]

Como se indica en la figura 2.4, el circuito se alimenta a 5V y es excitado con un pulso de 5V de amplitud y 300μs de duración. Simulando, se halla que

mientras éste se encuentra en alto, el transistor funciona en zona lineal, por lo que la corriente por el drain (y el diodo) crece linealmente hasta unos 68mA que es cuando baja a cero y el transistor se corta, de modo que esa corriente comienza a circular por el RLC. En el condensador pasan unos 45mA de pico, de esta forma podemos hallar el voltaje de pico para una frecuencia de 85kHz, correspondiente a la frecuencia media de operación del transductor:

$$V_P = \frac{I_P}{2\pi f C} = 580V$$

Esto explica el alto voltaje que se genera y se extingue rápidamente por la resistencia. Para disminuir el voltaje de pico máximo, alcanza entonces con acortar el pulso a menos de 300μs.

Dado que se tiene un circuito RLC, la frecuencia de oscilación está dada por la ecuación:

$$f_{OS} = \frac{1}{2\pi\sqrt{LC}}$$

que se obtiene de igualar la corriente que pasa por el inductor con la que pasa por el capacitor:

$$L\omega = \frac{1}{C\omega}$$

Según los datos del circuito y despreciando las capacidades parásitas, se obtiene:

$$f_{OS} = 92\text{kHz}$$

En la simulación se observa que da un valor de 87.7kHz. Sin embargo experimentalmente se pudo comprobar que ronda los 85kHz como se desea.

La resistencia R_1 no afecta en gran medida las características del circuito y se utiliza con el objetivo de limitar la corriente por el gate del transistor. Valores muy altos de ésta (5kΩ) provocan una disminución en el voltaje máximo alcanzado a unos 250V.

Cambios en R_2 afectan la exponencial que atenúa la sinusoide de resonancia. Valores chicos (con $5k\Omega$ se obtiene sólo un pico) disipan rápidamente la corriente que resuena en el RLC, y valores grandes ($500k\Omega$) generan una atenuación prolongada con decenas de picos notables, ya que menos corriente

es disipada por la resistencia mientras el transistor se mantiene cortado.

La función del diodo 1N4937 [12] es la de evitar que se tome corriente a través del transistor durante las oscilaciones. Esto es, que la corriente generada al momento del corte del transistor no pueda ir en el sentido drain-source. De lo contrario sólo se vería un solo pulso de voltaje elevado. Por este motivo, cualquier diodo capaz de soportar unos 600V puede utilizarse para dicha función, dado que los reverse-recovery-time (trr) de estos diodos suelen ser bajos, del orden de los cientos de ns. De esta forma, en el prototipo SMD se utilizó el diodo S2J-E3/52T [13] en lugar del sugerido.

El inductor utilizado es el 09P-183J-50 [14]. Fue seleccionado por cumplir con los requerimientos y tener un bajo costo. Su gran tamaño es una de las principales limitantes para minimizar la placa emisora.

El transistor utilizado fue cambiado por el IRFBC30 [15] dado que el IRFDC20 [16] sugerido no llegó en la primera importación, no se encontraba en el mercado local y este otro tiene características básicas similares.
Ambos soportan un $V_{DS} = 600\text{V}$, el sugerido tiene una $R_{DS(ON)} = 4{,}4\Omega$ y un $I_{DMAX} = 2{,}2\text{A}$, en tanto que el adquirido las mejora, presentando una $R_{DS(ON)} = 2{,}2\Omega$ y un $I_{DMAX} = 3{,}6\text{A}$, valores más que suficientes para nuestros requerimientos.

Sin embargo, el sugerido presenta un encapsulado de menores dimensiones al adquirido (el sugerido es HEXDIP y el adquirido TO-220), lo cual es de gran utilidad en la construcción de la placa en SMD, dado que este componente es el de mayor tamaño y uno de los grandes limitantes en el diseño. Cabe destacar que tanto el transistor IRFDC20 como el IRFBC30 utilizado, requieren un $V_{GS} > 4\text{V}$ para funcionar lo que impone una cota mínima para el valor de la señal de entrada.

A modo de verificación del funcionamiento del circuito, se muestra en la figura 2.6 la señal excitadora simulada con el programa Multisim en la terminal negativa del transductor de ultrasonido (la terminal positiva es constante igual a 5V).

Para implementar el emisor de ultrasonido utilizando un PIC12 como generador del pulso excitador, se considera necesario incorporar un optoacoplador de forma de separar el pin de salida del pic de este circuito. Esto se debe a los altos voltajes de pico que se manejan en el transistor los cuales pueden llegar a provocar picos de corriente elevados por el gate. Para res-

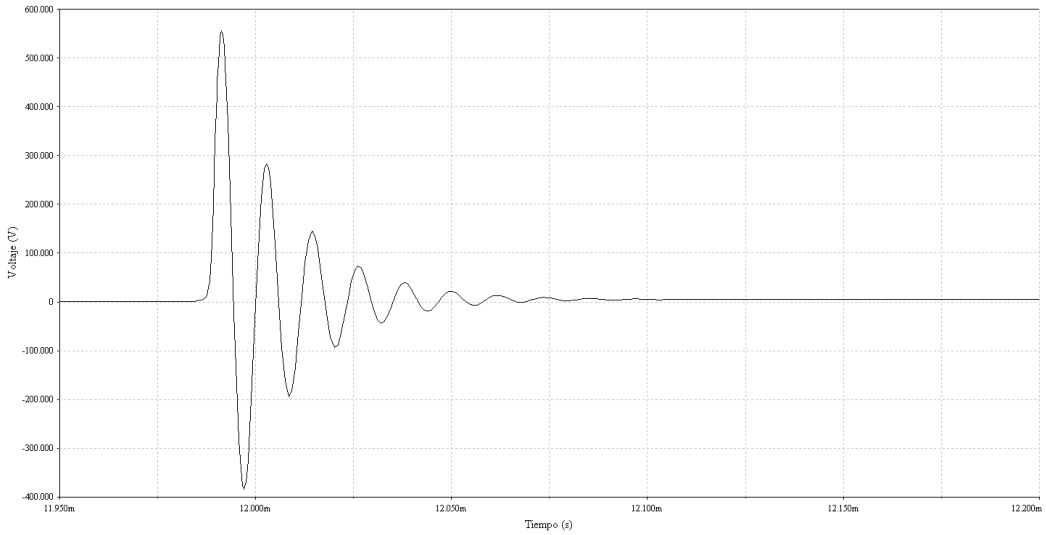

Figura 2.6: Simulación en Multisim de la señal excitadora del transductor

paldar esta suposición se simuló obteniéndose que el transistor toma picos de hasta 100mA por el gate y según la hoja de datos del PIC12 [17], éste es capaz de entregar hasta 25mA. Cabe resaltar que durante la realización de pruebas previas a la incorporación del optoacoplador, un PIC12 se quemó y se atribuyó a la falta de éste. También se estudió usar un transistor en lugar del optoacoplador pero no se obtuvieron resultados satisfactorios.

De esta forma, el bloque emisor de ultrasonido queda implementado como se muestra en la figura 2.7.

El optoacoplador utilizado fue el PC817 [18] o de forma equivalente puede usarse el KB817 [19]. Estos fueron elegidos por ser comunes y de bajo costo.

Es necesario poner la resistencia RUS1 de forma que el voltaje que genera el PIC12 pueda generar una corriente capaz de excitar al optoacoplador. Por medio de simulaciones se determinó que, alimentando al PIC12 con 3.3V, valores mayores a 680Ω generan corrientes menores a la requerida y valores más bajos incrementan el consumo de corriente. Si se alimenta el PIC12 con 5V, este valor es adecuado, pero puede utilizarse una resistencia de hasta 1kΩ si se quiere bajar el consumo. Sin embargo, experimentalmente se comprobó que para el caso de alimentar a 3.3V, el valor de 680Ω no es suficiente por lo que se utilizan 470Ω. Dado que se contaba con poco tiempo y probar con otros valores de resistencia no provocaría cambios notorios se optó por

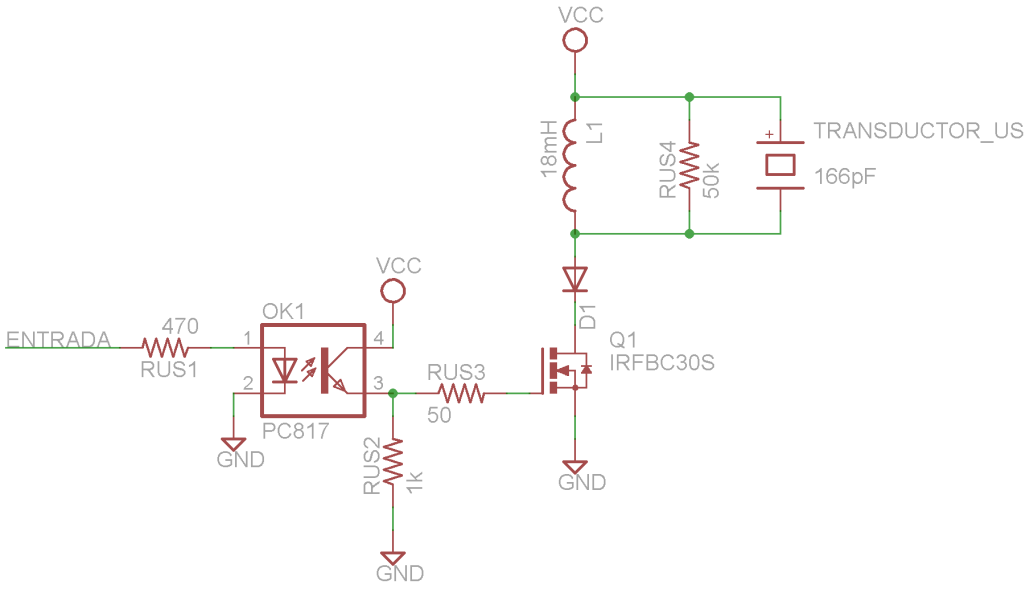

Figura 2.7: Bloque emisor de ultrasonido completo

dejar la resisencia de 470Ω. En el caso de alimentar con 5V, el valor de 1kΩ para esta resistencia es suficiente y se ubica entre el PIC12 y el optoacoplador en lugar de poner a tierra para que a su vez proteja el PIC12.

RUS2 se utiliza para generar el voltaje necesario para prender el transistor. Valores menores a 500Ω no son suficientes para prenderlo y mayores a 2kΩ comienzan a limitar la corriente que puede tomar el transistor por el gate en el momento en que se corta el optoacoplador, lo que se traduce en un menor voltaje de pico generado en el transductor de ultrasonido. Por lo que se toma RUS2 de 1kΩ.

Debido a que el transistor requiere un $V_{GS} > 4$V para funcionar, el voltaje de alimentación del optoacoplador debe ser mayor a 4.7V lo que impone que se utilice al menos 5V como alimentación de éste para asegurarse un buen funcionamiento.

A pesar de que el resto del circuito de ultrasonido puede alimentarse a menores voltajes y aún así funcionar correctamente, siempre y cuando se modifique el largo del pulso excitador de forma adecuada, cuanto menor voltaje se utilice, mayor será el consumo de corriente ya que el largo del pulso debe crecer mucho, lo que provoca un mayor consumo de corriente media en un período de tiempo. Por ende, es conveniente alimentar todo el circuito con la

misma fuente y cuanto mayor sea el valor de voltaje utilizado, menor será el consumo medio de corriente (el incremento en consumo en el optoacoplador es menor que la reducción en el consumo de la emisión ultrasónica). Para obtener mayor información al respecto, acudir a la sección 5.1.1.

Para concluir esta sección, el circuito implementado genera una señal excitadora adecuada de $400Vp$ máximo, en la terminal negativa del transductor de ultrasonido, como la que se muestra en las imágenes 2.8.

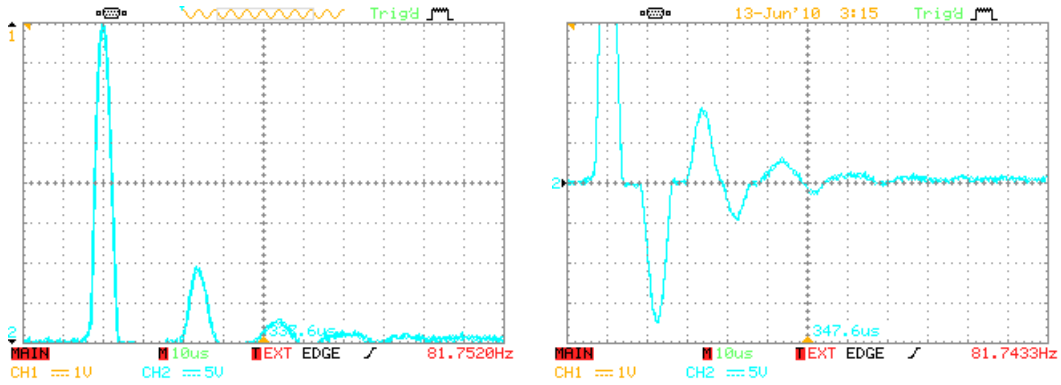

Figura 2.8: Señal excitadora del transductor de ultrasonido. Capturas tomadas con osciloscopio digital Gwinsted GDS-2062 usando punta divisora x10.

Generación de Señales y Lógica del Circuito

El esquemático del bloque puede verse en la figura 2.9.

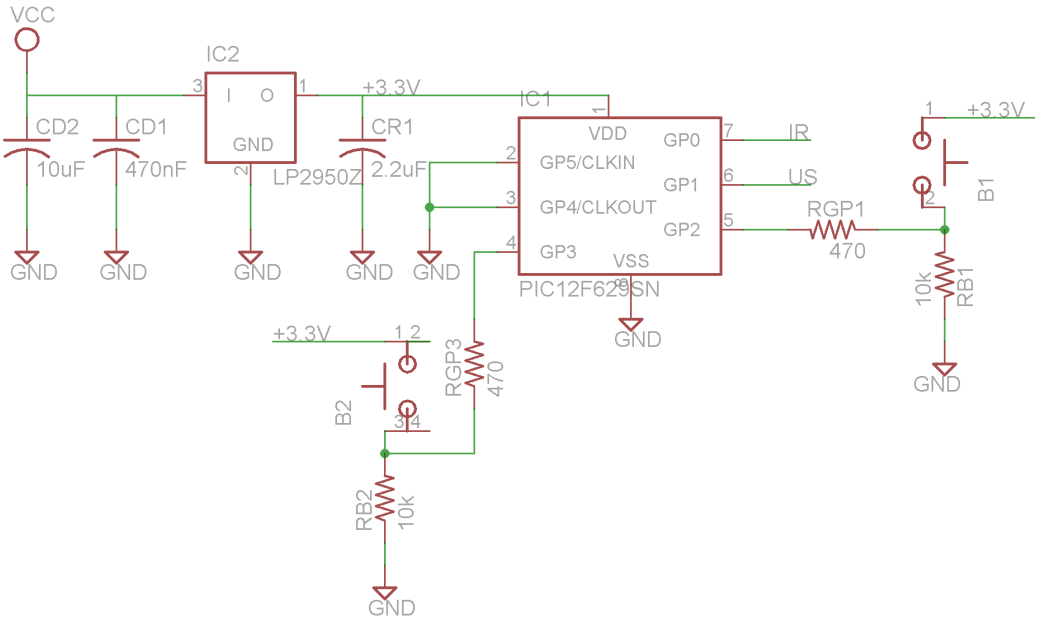

Figura 2.9: Bloque generador de señales

Como se mencionó anteriormente, el microcontrolador elegido para usar en el emisor es el PIC12F629 [17]. Él es el encargado de generar las señales que controlan los emisores de ultrasonido e infrarrojo, así como de detectar el pulsado de los botones y enviar dicha información codificada a través del infrarrojo hacia el receptor.

Para esta tarea, cualquier microcontrolador con al menos 2 pines de entrada, 2 de salida, una frecuencia de funcionamiento de al menos 500kHz y con 1 contador que permita contar hasta un tiempo de 1/75s, sería suficiente. Se eligió este integrado porque además de ajustarse a los requerimientos, es de bajo costo, bajo consumo, de fácil manejo y esta ampliamente difundido por lo que se cuenta con mucha información disponible del mismo.

Se tuvo especial cuidado en las corrientes que es capaz de entregar, razón por la cual se utiliza el optoacoplador, y también se llevan a tierra los pines que no se utilizan de modo de evitar posibles entradas de interferencia elec-

tromagnética.

Para la conexión de los botones, se respetaron las especificaciones del PIC12 que sugieren colocar resistencias en lugar de llevarlos a tierra directamente. A modo de reducir el consumo se utilizan resistencias de 10kΩ en la puesta a tierra.

Dado que el objetivo del proyecto es un prototipo funcional, para tener versatilidad en cuanto a la alimentación del emisor, a la vez de reducir el consumo y estabilizar el voltaje de alimentación, fue que se decidió incorporar un regulador de tensión lineal de 3.3V LP2950CZ3.3 [20]. Este fue seleccionado por ser fácilmente adquirible en el mercado local, por su bajo costo y porque sólo requiere de un capacitor adicional en su salida de 2.2μF.

Para darle robustez al emisor y que sea capaz de alimentarse con cualquier tipo de fuente sin recibir interferencias, fue que se le coloca CD1 y CD2 de 470ηF y 10μF respectivamente. CD1 es un capacitor cerámico con el fin de filtrar las componentes de ruido de alta frecuencia y CD2 es uno electrolítico con el fin de filtrar las componentes principales de ruido a bajas frecuencias.

2.1.3. Alimentación

Como se vio anteriormente, el circuito emisor requiere ser alimentado con una fuente continua de al menos 5V. Para dicho propósito, una solución sencilla puede ser a través de un cable conectado directamente al puerto USB o conectado al receptor, dado que como se verá más adelante, éste también se alimenta con los 5V que brinda dicho puerto y es capaz de suministrar la corriente necesaria para que ambos funcionen simultáneamente.

Sin embargo, se considera interesante poder brindar la posibilidad de que el lápiz sea inalámbrico, por lo que más adelante se expone un breve estudio sobre los distintos tipos de baterías que pueden utilizarse con tal propósito.

De las posibilidades que ofrece el mercado en cuanto a baterías, se optó por una costosa y complicada de implementar, aunque si imaginamos este proyecto como un producto a comercializar, también es la de menor costo de mantenimiento por parte del usuario, la que se presume genera menor contaminación ambiental, académicamente es interesante de desarrollar y sirve como forma de constatar que es posible utilizarla.

Esta opción se trata de las baterías recargables de alta difusión en productos como celulares y notebooks, las basadas en Litio-Ión o Litio-Polimero que tienen alta durabilidad, alta capacidad de carga, no tienen memoria de carga y poseen tamaños compactos. Sin embargo, también requieren cuidados especiales y su recarga no es sencilla.

Baterías

Para poder seleccionar los distintos tipos de baterías que pueden llegar a utilizarse se definen ciertas características mínimas que deben cumplir:

1. Deben tener una capacidad típica tal que dado el consumo del emisor de 5,6mA en el peor caso puedan brindar una autonomía de al menos 4 horas por carga si son recargables y de al menos 100 horas si no lo son.

2. Como se utilizan dentro de un lápiz, sus dimensiones deben ser tales que puedan ser colocadas dentro de una cavidad de como máximo 2 cm de diámetro y 6 cm de largo.

Además, se debe tener presente que se debe poder suministrar 5V, lo que implica utilizar más de una en serie si son de menor voltaje o la incorporación de un circuito DC/DC step-up los cuales suelen tener un consumo de corriente considerable. Otro factor importante es que su costo debe ser razonable.

Indagando en páginas web de empresas especializadas en baterías tales como PowerStream [21], All-Battery [22], BatterySpace [23] y teniendo en cuenta las consideraciones anteriores, se llega a que las baterías de Li-Ion (o Li-Polimero) de tamaño AAA son las que mejor se adaptan.

Algunas de las opciones consideradas dentro de este tipo son:

1. PST10440-con with MOLEX 5264 2-pin 3.7V, 300mAh [24]
2. UltraFire LC rechargeable 10440 [1] 500mAh 3.6V [25]
3. TrustFire 10440 600mAh 3.7V [26]

Luego se vio que las opciones 2 y 3 no se recomiendan ya que son baterías poco confiables de origen chino.

También se consideraron opciones no recargables:

4. Ultralife BA-5372-U [27]

Con las baterías de Litio se debe presentar especial cuidado tanto en su recarga como en su protección. Se vieron las especificaciones [28] de la recarga en las cuales se establecen corrientes de recarga que deben mantenerse constantes y luego reducirse hasta completar la carga a cierto voltaje. Se observa que la construcción de un cargador para este tipo de baterías requiere ciertos cuidados especiales [29][30][31].

Por otro lado, dado que el voltaje de las baterías encontradas ronda los 3.7V, se debe incluir en el circuito un DC/DC step-up converter de modo de alimentar la parte que requiere más de 5V. Esto se debe a la imposibilidad de cargar baterías de Litio en serie, por lo que la opción de utilizar dos baterías para alcanzar el voltaje requerido no es viable.

Por el consumo de los circuitos y la capacidad de estas baterías, se obtiene una independencia considerable que ronda las 40 horas de uso por carga,

[1] 10440 representa 10 mm x 44 mm y equivale al tamaño AAA

con una vida útil del orden de los 500 ciclos y una demora en la recarga de unas 3 o 4 horas.

Pilas no recargables incrementarían el costo para el usuario, tienen una vida útil de a lo sumo 200 horas en el mejor de los casos y contaminan aún más el medio ambiente. Presentan la ventaja de que pueden facilitar el diseño si es una de 6V como la Ultralife BA-5372-U [27].

Es recomendable como tarea a futuro realizar un estudio más refinado en cuanto a costos y consumo para determinar una configuración óptima, la cual tal vez requiera reguladores de voltaje lineales o conmutados como por ejemplo el LP2950cz3.3 utilizado.

Por lo pronto, para un diseño refinado del Lapix, se considera que lo ideal es alimentarlo con una batería Li-Ion 10440 recargable de 3.7V, como la que se muestra en la imagen 2.10, incorporarle un circuito de protección y utilizar un DC/DC step-up de salida 5V.[2]

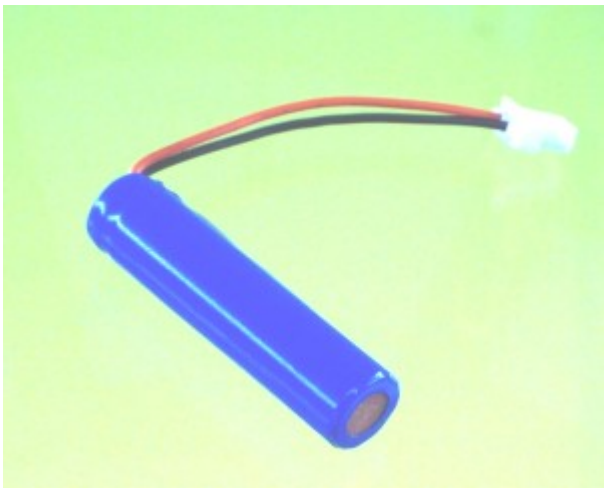

Figura 2.10: Batería Utilizada

Protecciones

Las baterías a base de Li-Ion requieren cuidados especiales en cuanto a su uso y carga dado que sin un circuito que las proteja pueden llegar a

[2]Notar que estas baterías ya incorporan cables y en caso contrario, no debe de soldarse nada en sus terminales dado que las baterías no soportan altas temperaturas.

inflarse, incendiarse e incluso explotar.

Algunas de las características que deben cumplir estos circuitos son [32] :

- Protección contra polaridad invertida
- Las baterías no deben cargarse cuando la temperatura se encuentra por debajo de 0° C o por encima de 45° C
- La corriente de carga no debe ser muy elevada, típicamente por debajo del 70 % del valor de corriente indicado en la capacidad de la batería C dada en mAh.
- Protección contra cortocircuitos
- Un fusible permanente debe abrirse si se aplica mucho voltaje en las terminales de la batería.
- Protección contra sobrecarga, la corriente debe detenerse si el voltaje de una celda aumenta por encima de 4.30 V.
- Protección contra sobre descarga, evita la descarga cuando el voltaje de la batería es menor a 2.3 Volts por celda (varía con el fabricante).
- Un fusible se abre si la batería es expuesta a temperaturas mayores a 100° C.

Para cumplir estos requisitos, existen integrados que engloban dichas funciones y hacen manejables de forma segura a estas baterías[33].

Varias empresas comercializan circuitos impresos a base de estos integrados, los fabrican en tamaños reducidos, con varias formas de acuerdo al formato de la batería, así como para paquetes de varias unidades y con distintos tipos de prestaciones. En [34] pueden verse ejemplos de estos impresos.

Dado que se contaba con poco tiempo y este trabajo no era central para el proyecto se opta por adquirir el circuito impreso (figura 2.11) "PCB for 3.7V of Li-Ion Battery (2.0A limit)"[35] en lugar de construir uno propio.

Este PCB fue seleccionado por presentar dimensiones adecuadas para nuestra batería y por tener un bajo costo. Su circuito se basa en el integrado DW01 y dentro de su hoja de datos [37], se encuentra una versión más sencilla

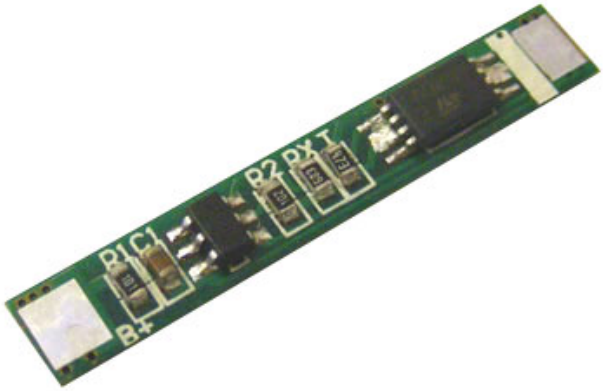

Figura 2.11: PCB adquirido, dimensiones 29mm x 4.5mm. Por más especificaciones consultar [36]

del esquemático del impreso adquirido como se muestra en la figura 2.12 dado que el impreso adquirido además cuenta con un termistor y un ID resistor.

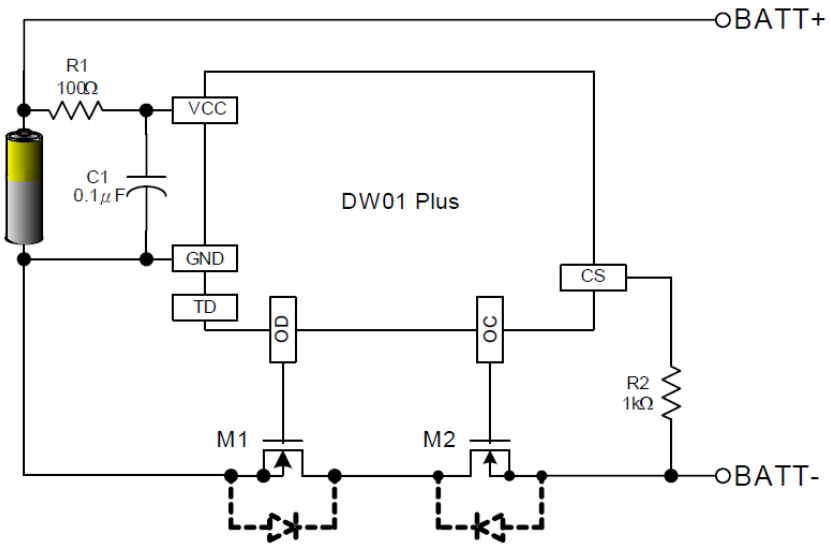

Figura 2.12: Esquemático de protección de la batería

Los transistores MOSFET son implementados con el integrado 5N20V [38] y se utilizan para evitar voltajes superiores a los 4.30V. R1 y C1 se utilizan para desacoplar la batería y R2 está para sensar la corriente y detec-

tar la presencia de un cargador. Además, la placa cuenta con un termistor ($T = 47k\Omega$) que actúa como fusible contra altas temperaturas y un ID resistor ($Rx = 68k\Omega$) para identificar la batería.

DC/DC Step-up

Para obtener los 5V necesarios de forma de alimentar el circuito emisor, se utiliza un circuito DC/DC step-up que se basa en el integrado NCP1400 (NCP1400ASN50T1 para 5V) [39].

Este integrado fue elegido porque además de ajustarse perfectamente a nuestros requerimientos, que son: voltaje de entrada entre 1V y 4V, voltaje de salida 5V estable y corriente de salida hasta 100mA que es mayor a nuestro consumo máximo; presenta un encapsulado pequeño, es de bajo costo y también forma parte de un circuito impreso comercializado por la empresa Sparkfun [40], la cual nos proporciona todos los datos necesarios para reproducir el impreso y es el recomendado en la hoja de datos del integrado salvo por un pequeño cambio en el valor del condensador de desacople de la salida.

A diferencia de la protección de la batería, en este caso se prefirió construir el impreso en lugar de comprarlo con el objetivo de modificar levemente el diseño a modo de acoplarlo más fácilmente dentro del emisor sin que incremente su diámetro. Además, de esta forma se abaratan costos.

El esquemático del circuito utilizado es el mismo al proporcionado por la empresa y se muestra en la figura 2.13.

A partir de este esquemático se genera el layout mostrado en la imagen 2.14.

Los archivos Eagle del esquemático y el Layout pueden encontrarse en CD/circuitos/equematicos y layouts/.

La placa finalmente construida se muestra en las figuras 2.15.

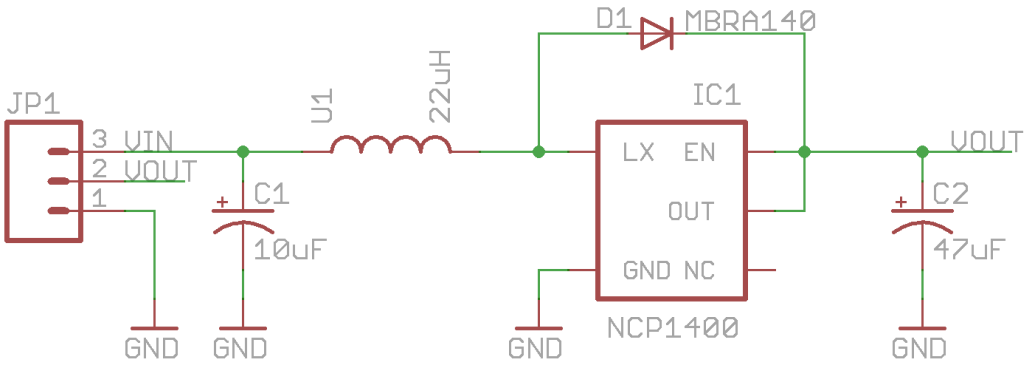

Figura 2.13: Esquemático del DC/DC Step-up

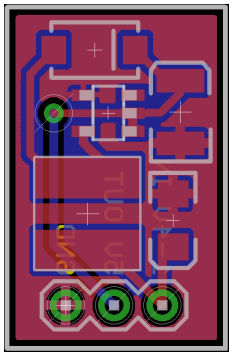

Figura 2.14: Layout del DC/DC Step-up

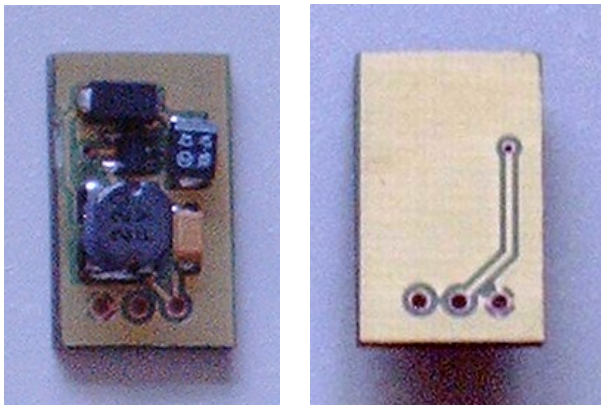

Figura 2.15: DC/DC Step-up construido visto de arriba y abajo

Cargador

La construcción de un cargador adecuado para baterías del tipo Li-Ión puede realizarse de diversas formas, por ejemplo con un circuito basado en

un regulador de tensión [41].

Pero al igual que sucede con el DC/DC step-up, en este caso también existen integrados que facilitan la delicada tarea de carga de estas baterías y reducen el tamaño del circuito al necesitar menos componentes que lo acompañen. Uno de ellos es el MAX1555 [42] y es el elegido para realizar esta tarea ya que, además de ser específico para esto, también forma parte de una circuito impreso comercializado por la empresa Sparkfun [43], la que nos vuelve a proporcionar todos los datos necesarios para reproducir el impreso.

El MAX1555 es capaz de cargar una celda única de Li-Ión tanto por medio del puerto USB como por un cargador externo. Al estar conectado al USB la corriente de carga es limitada internamente a 100mA que corresponde con la cantidad de corriente que es capaz de entregar el puerto en su configuración por defecto (otras configuraciones permiten que entregue hasta 500mA) y al utilizar un cargador externo esta corriente se incrementa a 280mA lo que acelera el proceso de cargado. Cabe destacar que el integrado es capaz de funcionar correctamente al tener ambas fuentes conectadas simultáneamente. El cargador externo puede ser cualquier fuente de continua entre 3.7V y 7V, pero se recomienda utilizar una de 5V. También presenta un pin indicador de carga que se activa por negado durante el proceso de cargado y se apaga yendo a un estado de alta impedancia cuando la corriente de carga se reduce a menos de 50mA, lo que indica una carga completa de la batería.

De forma de facilitar su uso y como se cuenta con el espacio requerido, el cargador forma parte del circuito del receptor, por lo que se adapta el diseño proporcionado por Sparkfun de modo que el lápiz se coloque sobre el receptor y a través de un par de conectores se cargue la batería.
Para más información ir a la sección 2.2.3.

Los conectores utilizados son los que se muestran en las imágenes 2.16, fueron adquiridos en el mercado local pero pueden sustituirse por cualquier otro par que posea dos terminales, con dimensiones razonables y de preferencia que sean cilíndricos como lo suelen ser los de alimentación.

El esquemático resultante es el correspondiente a la figura 2.17 y es similar al que aparece en la hoja de datos del MAX1555.

La señal de carga es implementada por medio de un LED. En el diseño aparecen dos en paralelo con el objetivo de poder elegir uno de ellos para colocar en el impreso dado que uno correspondiente al encapsulado SMD y

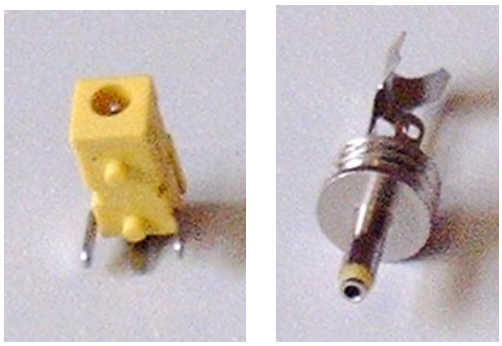

Figura 2.16: Conector hembra usado en el lápiz y conector macho usado en el receptor.

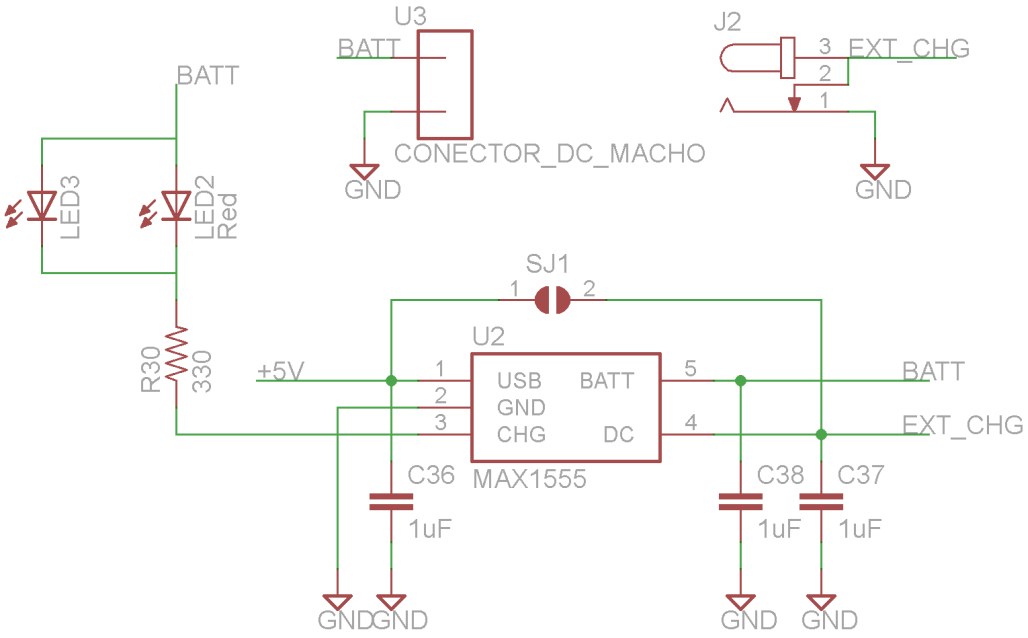

Figura 2.17: Esquemático del cargador

otro al DIP. Para acelerar el proceso de carga sin forzar el puerto USB, se da la posibilidad de colocar un conector de continua hembra de 2mm (DC power jack). Se utiliza un solder-pad de forma de tener la posibilidad de cortocircuitar la señal de entrada de USB con la del cargador externo, con esto se puede tomar hasta 280mA por el puerto USB y se acelera la carga. Sin embargo, en caso de cortocircuitar estas señales con el solder-pad, no debe utilizarse bajo ningún concepto el conector externo de continua simultáneamente con el conector USB.

Dado que el cargador forma parte del receptor, el diseño del impreso así como la placa construida pueden encontrarse en la sección 2.2.3.

2.1.4. Placas Construidas

Durante el transcurso del proyecto se construyeron diversas placas para realizar pruebas de los distintos bloques que componen los circuitos. En esta sección se expondrán solamente las placas que componen el primer prototipo funcional y el prototipo final en formato SMD. El primer prototipo fue realizado con componentes "through-hole" y en un PCB de Pertinax de una capa, lo que derivó en un gran tamaño tanto para el emisor como el receptor, dificultando su uso pero que reúne todas las funcionalidades requeridas del proyecto. En tanto que el prototipo funcional final en formato SMD fue construido en un PCB de fibra de vidrio de doble capa, lo que permitió reducir drásticamente su tamaño y fue posible incorporarle características adicionales además de brindarle ergonomía al lápiz.

Criterios de minimización de ruido electromagnético

A la hora de desarrollar la placa emisora así como también la receptora fue de vital importancia tener las mayores precauciones posibles para lograr disminuir el ruido electromagnético. Se tuvieron mayores cuidados aún al desarrollar la placa receptora ya que se pudo ver experimentalmente que los bloques correspondientes a la recepción del ultrasonido se ven fuertemente afectados por ruido electromagnético producido por la placa emisora en el RLC generador de la señal excitadora del ultrasonido. Las consideraciones que se tuvieron en cuenta fueron las siguientes:

- Diseñar pistas de alimentación suficientemente gruesas, planos de tierra importantes con mayor capacidad de transportar corriente y blindar las componentes de ruido generadas por otras componentes. El plano de tierra debe ser sumamente importante, y en general es conveniente que llene todos los espacios no utilizados en la placa, cubriéndola en forma de rejilla o completamente. En caso de que el circuito tenga dos tierras, una analógica y otra digital, permitiendo un buen funcionamiento en forma independiente, las tierras deben desarrollarse separadamente y tocarse sólo en un punto.

- Colocar pequeños capacitores entre alimentación y tierra en los casos en que haya chips que requieran mayores corrientes de alimentación a altas frecuencias, ubicándolos lo más cerca posible de estos componentes.

- Al momento de hacer las pistas, no hacer ángulos rectos ya que a altas frecuencias éstos hacen las veces de antenas y son altamente suscepti-

bles a actuar como líneas de transmisión debido al cambio puntual de impedancias.

- No hacer pistas largas ya que estas se asemejan a una antena que capta ruidos externos. Además, pistas más finas son menos susceptibles al ruido.
- Un componente que se comunique con otro que esté afuera de la placa con un conector debe ubicarse lo más cerca posible a ese conector.

Primer prototipo

Para este primer prototipo se optó por un diseño simple faz con el objetivo de ser fabricado rápidamente con el método de la plancha y pese a que se trató de minimizar su tamaño para ponerlo dentro de un encapsulado similar a un lápiz, no se pudo reducir a menos de 4.5cm x 16cm (figura 2.18). Los archivos correspondientes del PCB Wizard pueden encontrarse en CD/circuitos/esquematicos y layouts.

El principal causante del ruido electromagnético es el circuito RLC formado por RUS4, L1 y la capacitancia equivalente del piezoeléctrico emisor de ultrasonido. Por esto es que se colocaron lo más cerca posible uno del otro y del conector del flex de la punta emisora además de reducirse el grosor de las pistas relevantes en esa parte. Los otros componentes que manejan altos voltajes son el diodo y el MOS que se decidieron colocar detrás del pulsador primario para que no oficiaran de obstáculos, ya que éste se planeaba colocar en forma vertical de modo que se pudiera tocar con algo cilíndrico y alargado que pasara a través de la punta emisora, la cual es hueca. Finalmente como se optó a un diseño en SMD, el pulsador primario se soldó acostado de forma que la parte mecánica no complicara. Se dejaron amplios márgenes de tierra que rodean la placa con el objetivo de poder blindarla por completo en caso de ser necesario o sólo blindar hasta el botón secundario. Debajo del conector de la punta, se dejó una zona de tierra con dos orificios, con el objetivo de poder adosar la punta y luego cortar la placa dándole forma de lápiz.

En el extremo superior se encuentran los pines de tierra y Vcc a los cuales se les conecta cocodrilos que van hacia una batería de 9V o cualquier otra fuente de continua entre 5V y 10V. Hay que tener en cuenta que al variar el voltaje de alimentación es necesario variar acordemente el largo del pulso excitador de ultrasonido en el código del PIC12 de forma de mantener una excitación de igual magnitud. La resistencia RIR2 que gradúa la potencia

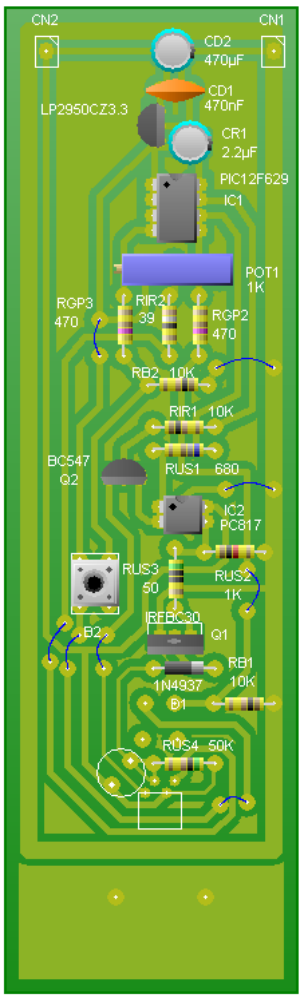

Figura 2.18: Diseño en PCB Wizard de primer prototipo de la placa emisora, dimensiones 4.9cm x 16.5cm.

de emisión del infrarrojo fue sustituida por la serie de una resistencia fija de 39Ω y un potenciómetro de 1kΩ con el fin de poder ajustar este parámetro sin correr riesgos de generar corrientes excesivamente altas. Este emisor del primer prototipo es capaz de funcionar sin blindaje, pero si se aproxima a menos de 2 cm de los receptores de ultrasonido, a menos que se blinde, no se detecta señal.

La placa construida se muestra en las imágenes 2.19.

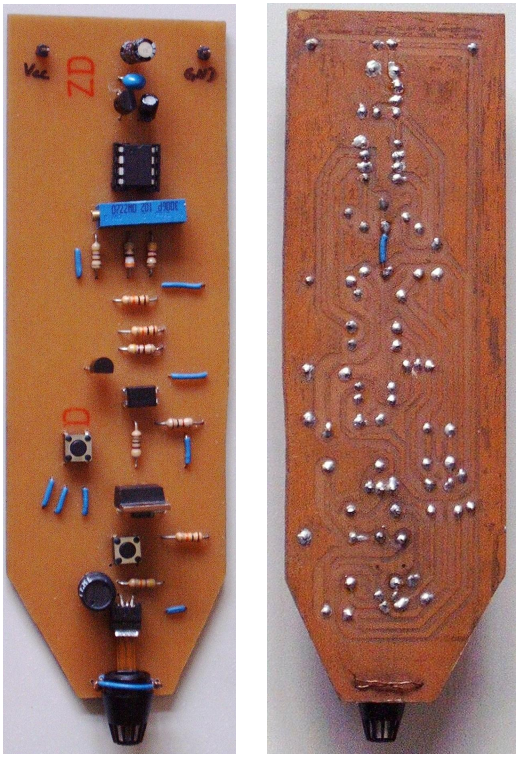

Figura 2.19: Primer prototipo visto de arriba y de abajo.

Prototipo en SMD

Buscando un nuevo diseño capaz de encapsularse de modo de brindar la ergonomía necesaria para utilizarse como un lápiz, es que se procede a construir una nueva versión del prototipo en formato SMD doble capa. El esquemático de esta versión es el mismo al del primer prototipo, con la diferencia de que este nuevo diseño puede ocupar tan sólo 16mm x 60mm y lo que principalmente limita que este tamaño se reduzca es el inductor y el transistor utilizados en la emisión del ultrasonido.

Para su diseño se mantienen las mismas consideraciones en cuanto al ruido electromagnético, por eso se utilizan pistas más finas en el RLC y se ponen los componentes resonantes lo más juntos posible entre sí. Por lo mismo, se deja un borde exterior de tierra que rodea el impreso para poder blindarlo.

El pulsador primario correspondiente al botón izquierdo en un mouse, se acopla al circuito a través de cables, ya que es ubicado de forma que se

presione al tocar la punta del lápiz contra una superficie [3]. Para este pulsador se utilizó uno que se extrajo de una disquetera vieja y pueden adquirirse en el mercado local. La ventaja de estos, es que no requieren presionarlos con fuerza, por lo que con un leve contacto de la punta contra la pantalla o contra cualquier superficie basta para ser detectado. Para el botón secundario correspondiente al botón derecho en un mouse, se utiliza el mismo tipo de pulsador que los usados con el primer prototipo y se encuentra ubicado en la capa del fondo, opuesto al inductor el cuál va acostado en la capa del frente. En la parte inferior se dejó una capa de tierra con el objetivo de ser cortada en la medida que fuera necesario al momento de encapsularlo y sujetar la punta emisora.

Con la idea de poder dar la opción de alimentar directamente desde el puerto USB, se dejan 2 pads correspondientes a D+ y D- con el simple propósito de darle mayor agarre al cable. El PIC12 en el primer prototipo tenia un formato DIP y estaba acoplado a la placa a través de un zócalo, lo que permitía retirarlo fácilmente para su programación[4]. Sin embargo, en este caso por tener un encapsulado SMD y para no incrementar el ancho del impreso (lo que incrementa el diámetro del lápiz), se opta por soldar cuatro cables en las terminales correspondientes al PIC para dicho propósito[5].

El layout del prototipo del emisor en SMD puede verse en la figura 2.20.

Los archivos Eagle de este Layout pueden encontrarse en CD/circuitos/-esquematicos y layouts.

La placa construida se muestra en las imágenes 2.21.

[3]Por más información ver la sección 2.1.5

[4]Otra buena solución para esa placa hubiese sido realizar una programación in-circuit

[5]Por información sobre como programar el PIC12 ver CD/referencias/41191d-PIC12F629-675-PIC16F630-676 Memory Programming.pdf y sobre como construir un programador se puede consultar el enlace: `http://www.jdm.homepage.dk/newpic.htm`

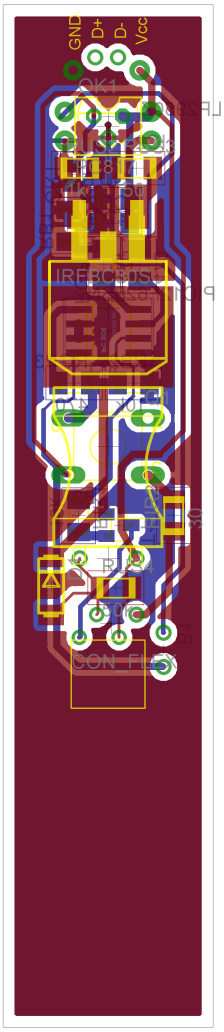

Figura 2.20: Layout del emisor en SMD, dimensiones 18.5mm x 90.0mm.

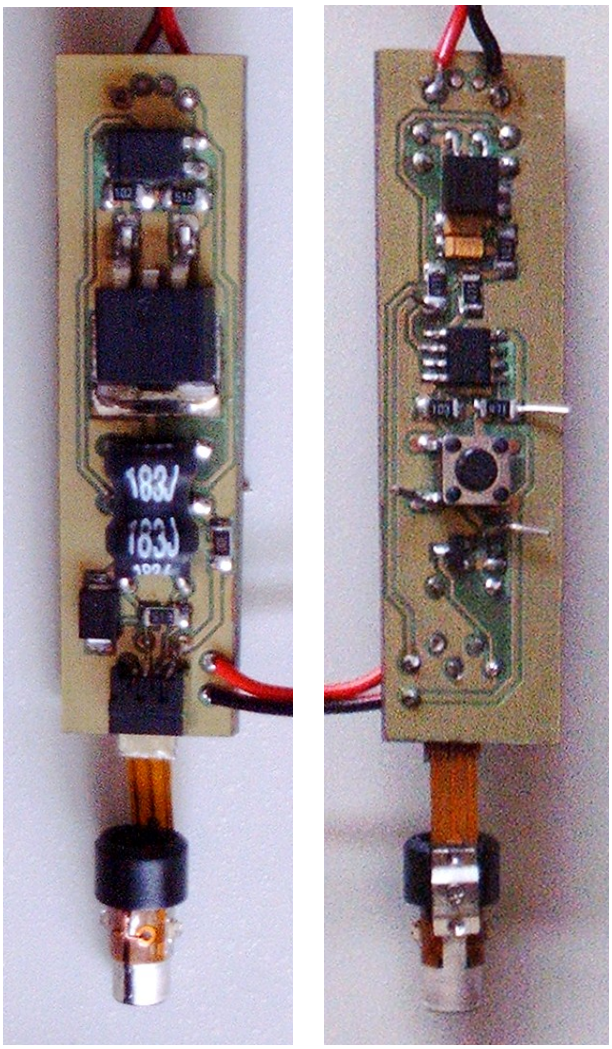

Figura 2.21: Prototipo SMD visto de arriba y de abajo.

2.1.5. Encapsulado

Para que el lápiz tenga la ergonomía necesaria, consiga robustez, se facilite el uso y pueda incorporarse la batería con su protección, el DC/DC step-up, un interruptor de encendido y apagado y el conector para cargar la batería con la placa receptora, es que se construye un encapsulado como el que se muestra en la figura 2.22.

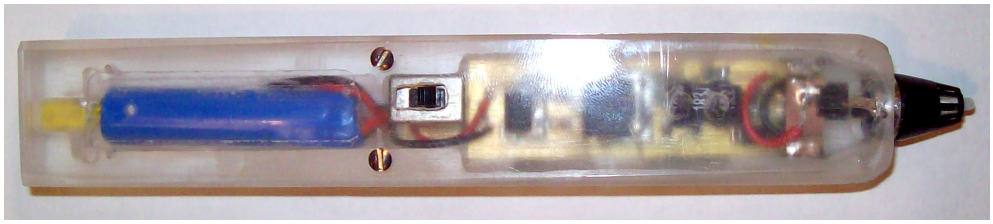

Figura 2.22: Emisor con encapsulado, dimensiones 2.0cm x 2.5cm x 16.5cm.

La punta del lápiz está conectada mecánicamente al botón primario de forma de que cuando se presiona se genera un "click" correspondiente con el botón izquierdo de un mouse. El pulsador utilizado fue extraído de una disquetera vieja pero puede conseguirse en el mercado local. Fue seleccionado debido a que se requiere de muy poca fuerza para presionarlo. Aunque tiene un amplio recorrido, mecánicamente se logró limitarlo y se obtuvo como resultado que con apenas 1mm de recorrido y muy poca fuerza se detecte el pulsado sin que se hunda la punta, dando la sensación de que con apenas tocar la pantalla se esté escribiendo.

En un costado cercano a la punta cuenta con el botón secundario correspondiente con el "click" derecho del mouse. Esto puede observarse en la figura 2.23.

En la figura 2.24 puede verse el emisor desarmado. Aquí se aprecia que el encapsulado fue construido artesanalmente con acrílico transparente y cuenta con una tapa removible con tornillos. En el extremo opuesto a la punta se encuentra el conector de la batería de forma de que al apoyar el lápiz sobre el conector macho del receptor se inicie el proceso de carga. Entre la protección de la batería y la alimentación positiva del DC/DC step-up se encuentra el interruptor de encendido y apagado.

Con esto se obtiene la forma final del emisor similar a un lápiz, con un tamaño razonable y un peso liviano de 50gr.

Figura 2.23: Botones del emisor.

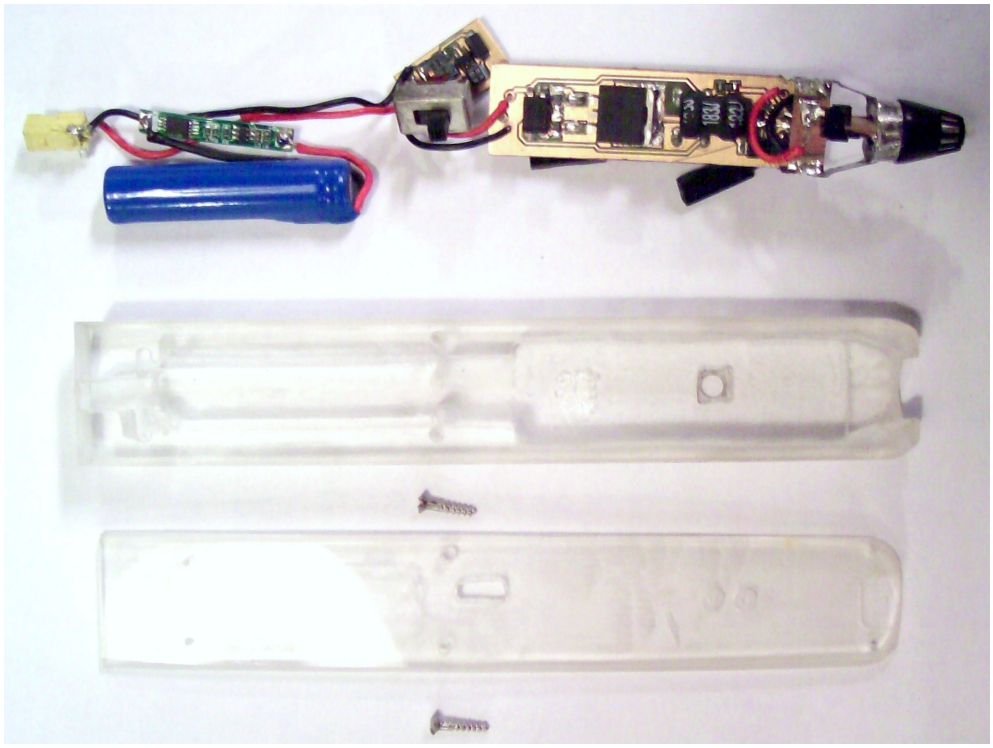

Figura 2.24: Emisor desarmado.

2.1.6. Estudio de Seguridad Humana

Dado que el circuito de emisión de ultrasonido maneja voltajes altos para excitar los emisores, se realiza un análisis para comprobar que el mismo no es perjudicial para el ser humano.

La resistencia del cuerpo humano depende de varios factores, entre ellos: masa corporal, estado de humedad de la piel, frecuencia, etc.

El cuerpo se modela como (figura 2.25) [44] donde se puede observar que la resistencia total depende del valor de la frecuencia.

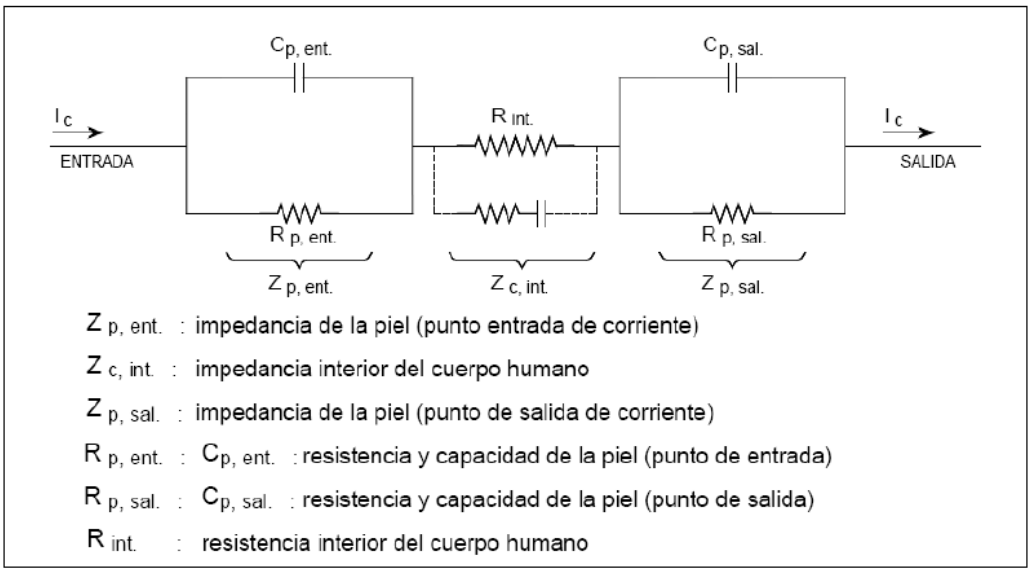

Figura 2.25: Modelo del cuerpo Humano

Para los valores más comunes de frecuencia existen gráficas experimentales de los efectos producidos (figura 2.26) [44]

La figura 2.26 es una gráfica para corrientes entre 50 y 100Hz donde se muestran los límites para las zonas con distintos efectos:

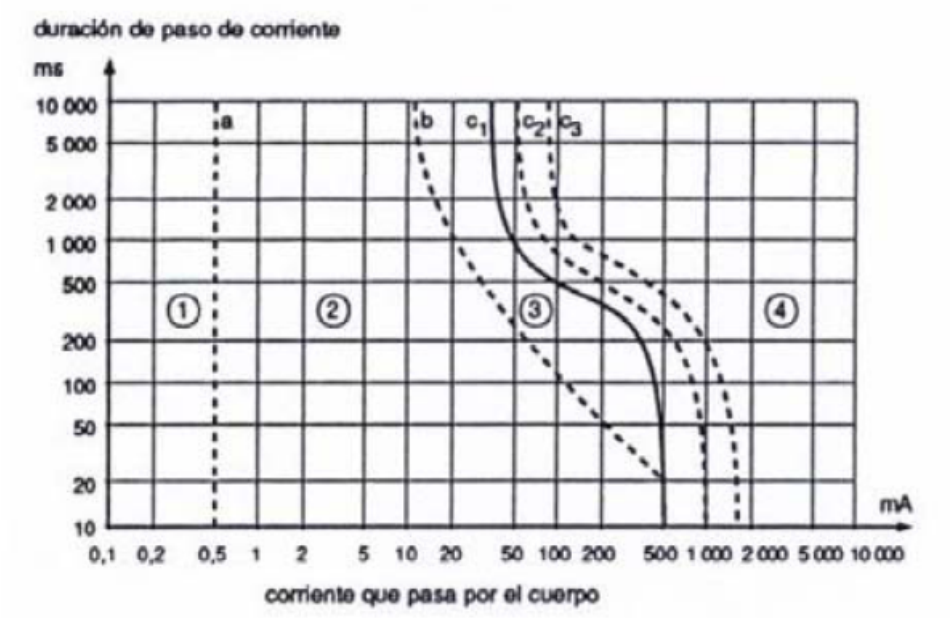

Figura 2.26: Efectos según frecuencia de la corriente sobre el cuerpo humano

- Curva a: umbral de percepción
- Curva b: umbral de no soltar
- Curva c_1: umbral fibrilación ventricular
- Curva c_2: probabilidad fibrilación ventricular 5 %
- Curva c_3: probabilidad fibrilación ventricular 50 %

Para valores más altos de frecuencia se dificulta la cantidad de información disponible.

Sin embargo tenemos la siguiente tabla [45]:

BODILY EFFECT	DIRECT CURRENT (DC)	60 Hz AC	10 kHz AC
Slight sensation felt at hand(s)	Men = 1.0 mA Women = 0.6 mA	0.4 mA 0.3 mA	7 mA 5 mA
Threshold of perception	Men = 5.2 mA Women = 3.5 mA	1.1 mA 0.7 mA	12 mA 8 mA

Painful, but voluntary muscle control maintained	Men = 62 mA	9 mA	55 mA
	Women = 41 mA	6 mA	37 mA
Painful, unable to let go of wires	Men = 76 mA	16 mA	75 mA
	Women = 51 mA	10.5 mA	50 mA
Severe pain, difficulty breathing	Men = 90 mA	23 mA	94 mA
	Women = 60 mA	15 mA	63 mA
Possible heart fibrillation after 3 seconds	Men = 500 mA	100 mA	
	Women = 500 mA	100 mA	

Donde podemos observar que con el aumento de la frecuencia, se hace necesaria más corriente para generar efectos adversos (respecto de corrientes AC).

Del documento [46] se obtiene que para frecuencias comprendidas entre 10kHz y 100kHz, el umbral de percepción se encuentra entre 10mA y 100mA (rms).

En general podemos suponer que la resistencia de la piel es 1000-2000Ω y que la resistencia interna es de 500Ω. Sabiendo que nuestro circuito genera por aproximadamente $200\mu s$ una señal que tiene al inicio un pico máximo de 400Vp y asumiendo el peor caso de resistencia del cuerpo (se desprecia la resistencia de la piel), tenemos una corriente de 800mA que duraría menos de $6\mu s$. El valor de corriente podría considerarse potencialmente peligroso, sin embargo es un valor instantáneo y no un RMS por lo que no se tiene como evaluar.

Se realizan algunas simulaciones, primero se conecta una resistencia de 500Ω a uno de los bornes del capacitor del circuito de emisión y otro a tierra. Lo que se puede observar es que por la resistencia circulan 10mA de continua, los mismos (según la tabla anterior) entrarían en el umbral de percepción pero no presentarían riesgo para la salud. Recordar además que se está en el peor caso de resistencia del cuerpo.

Una segunda simulación consiste en conectar una resistencia de 500Ω

en paralelo con el capacitor, en este caso se observa una corriente RMS de 5.74mA que para el caso de 10kHz está fuera del umbral de percepción por lo que también lo estará para el caso de 85kHz, dado que como se observa en la tabla, a medida que aumenta la frecuencia el umbral de percepción también aumenta.

Conclusión

Se evalúa considerando como peor caso la resistencia del cuerpo en 500Ω, se ve que el único caso donde se está en el nivel de percepción pero se está fuera del nivel del peligro es para el caso de contacto circuito-tierra. Para dicho tipo de contacto, se observa que la resistencia minima de manera de no entrar en el area de peligro es de 100Ω caso para el cual se tiene 50mA(dc).

Dado lo anterior se puede decir que el lápiz no representa peligro alguno para la salud del usuario siempre que su resistencia corporal este por encima de 100Ω e incluso en ese caso el daño que se puede ocasionar dependerá del tiempo de exposición que, dado que la duración de los pulsos es casi instantánea, no es demasiado prolongada. De todas formas en caso de comercialización y dado que en su gran mayoría se espera que los usuarios sean niños, se recomienda el aislamiento del circuito emisor de US.

2.2. Receptor

El receptor es el responsable de detectar las señales de infrarrojo y ultrasonido que envía el lápiz, procesarlas, contar la diferencia de tiempos entre ellas y mandarle esa información, además de la del pulsado de los botones, al driver alojado en la PC a través del puerto USB.

Como se pretende poder escribir sobre la pantalla, el receptor tiene que ir adosado a ésta y por eso es que cuenta con un sistema de enganche que le permite usarse sobre la XO, sobre un monitor LCD o sobre cualquier superficie lisa horizontal.

Adicionalmente, también cuenta con la capacidad de recargar la batería alojada en el emisor.

2.2.1. Circuito Diseñado

El circuito receptor está compuesto de varios bloques los cuales se van a exponer de a uno para dar una mejor claridad de lectura. En caso de querer ver el esquemático completo, se puede consultar la figura G.3 que se encuentra en el anexo G.

Este circuito se alimenta directamente con los 5V que brinda el puerto USB y como está pensado para usarse con la XO, un sistema portátil que utiliza batería, es de importancia que este no le consuma demasiada corriente ni recursos, como lo son capacidad de procesamiento y memoria disponible, a su anfitrión.

Se divide en dos bloques receptores de ultrasonido idénticos que comparten un sub-bloque generador de un umbral de comparación fijo, un bloque receptor de infrarrojo, un pequeño bloque para la carga de la batería del emisor y uno correspondiente al microcontrolador del receptor basado en un PIC32 y encargado de la lógica, el conteo y comunicación USB de las señales con la PC.

A continuación se explicará cada bloque con excepción del bloque responsable de la carga de la batería el cuál ya fue tratado en la sección 2.1.3.

2.2.2. Detalles del Circuito

Recepción de Ultrasonido

Para detectar las señales de ultrasonido se utilizan los sensores SR80KHZ-01 [5] que hace juego con la punta emisora PT80KHZ-01 [4] utilizada en el lápiz, los cuales fueron recibidos como muestras gratis cortesía de Measurement Specialties [8]. Estos filmes piezoeléctricos transductores de ultrasonido presentados en la figura 2.27, tienen una directividad horizontal superior a los 180 grados y una vertical de ±25 grados lo que los hacen adecuados para detectar las señales emitidas dentro del área de cobertura establecida.

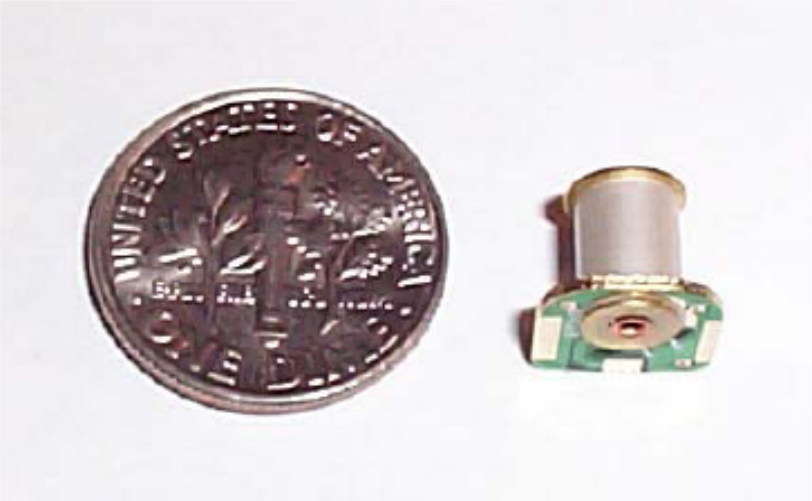

Figura 2.27: Sensor receptor de ultrasonido. Fuente hoja de datos del SR80KHZ-01 [5].

La señal típica generada tras la excitación ultrasónica provocada por la punta emisora es del tipo de la que se muestra en la figura 2.28, la que en este caso corresponde a una respuesta en el tiempo obtenida tras la excitación con una sinusoide decadente de máximo 800 V p-p de la punta emisora, a una distancia de 30cm del receptor y con una ganancia del receptor de +26dB.

Dado que las señales generadas por el transductor, tras la recepción de la señal de ultrasonido, son del orden de algunos mV, es necesario realizar una amplificación antes de transformar la señal al formato digital. Para esto, al igual que sucede con la emisión, la hoja de datos de los transductores

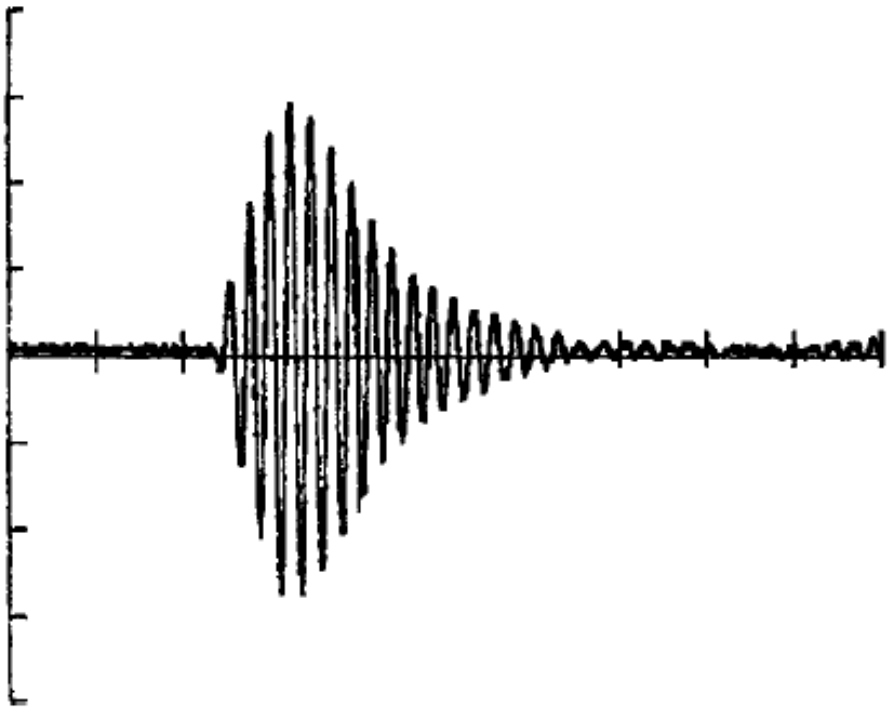

Figura 2.28: Eje x tenemos 50μs/div. Eje y 5mV/div. Fuente hoja de datos del SR80KHZ-01 [5].

de recepción nos brinda un circuito ejemplo como se muestra en la figura 2.29.

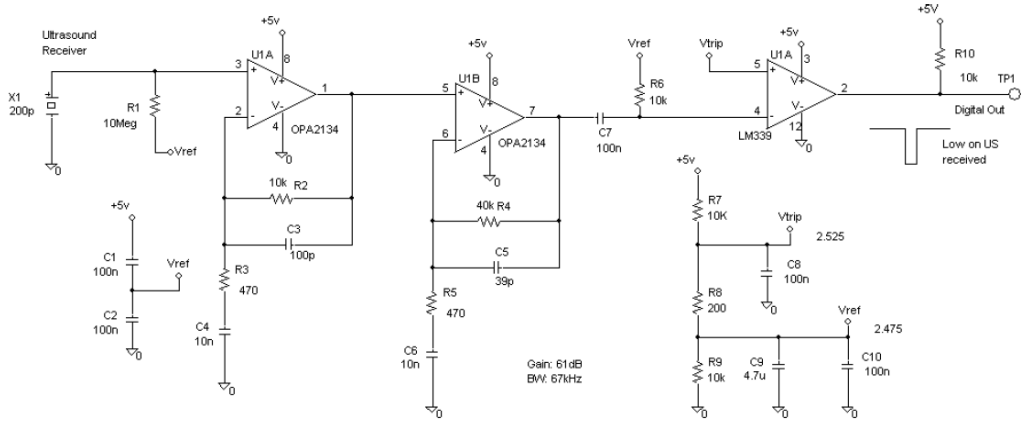

Figura 2.29: Circuito ejemplo para la recepción. Fuente hoja de datos del SR80KHZ-01 [5].

El circuito realiza una amplificación en dos etapas selectivas en frecuencia, con una ganancia total de 61dB y un ancho de banda de 67kHz centrado aproximadamente en el rango situado entre los 80kHz y los 90kHz en el cual se encuentra nuestra señal a recibir. Finalmente, hace una conversión digital por medio de un comparador en base a un umbral fijo.

Para hacer la amplificación de dos etapas utiliza dos amplificadores operacionales de audio OPA2134 [47] de alto rendimiento y de muy bajo ruido. Ambos amplificadores vienen en un mismo integrado. En la hoja de datos del piezoeléctrico se sugiere blindar esta etapa de preamplificación y es justamente la parte del receptor que mayores problemas presenta frente a interferencia electromagnética generada en el RLC del emisor al emitirse la señal de ultrasonido.

La etapa de comparación es hecha con un comparador LM339 [48] de bajo consumo y bajo offset, que en un mismo integrado presenta cuatro comparadores. Como en todo el receptor sólo necesitamos dos de ellos, este integrado está sobredimensionado y un buen trabajo a futuro para reducir costos puede ser encontrar otro integrado que se ajuste mejor a nuestras necesidades y que sólo posea 2 comparadores.

Un detalle no menor a resaltar del circuito es que para conseguir un correcto funcionamiento, la alimentación negativa de los amplificadores operacionales OPA2134 [47] en lugar de ir a tierra como se indica en el circuito sugerido, es necesario darle una referencia negativa (-5V por ejemplo) respecto a la tierra de entrada para que el amplificador no sature en la salida.

Dado que la fuente de voltaje disponible se trata del puerto USB que brinda 5V y que los amplificadores operacionales OPA2134 pueden funcionar desde ±2,5V, es que el circuito se modifica como se muestra en la figura 2.30. Aprovechando que la señal Vref es de aproximadamente 2.5V, es bastante estable e inmune a interferencias dado que posee varios capacitores de desacople, es que se utiliza como referencia de tierra para el piezoeléctrico receptor así como para el amplificador[6]. Además, se modifica el voltaje de la salida digital de 5V a 3.3V de forma de disminuir un poco el consumo de corriente y aprovechando que se cuenta con una fuente estable de este valor en el receptor ya que el microcontrolador así lo requiere.

[6]Los condensadores C4 y C6 del amplificador también pueden ir conectados a tierra con lo que se mejora un poco el diseño del impreso construido

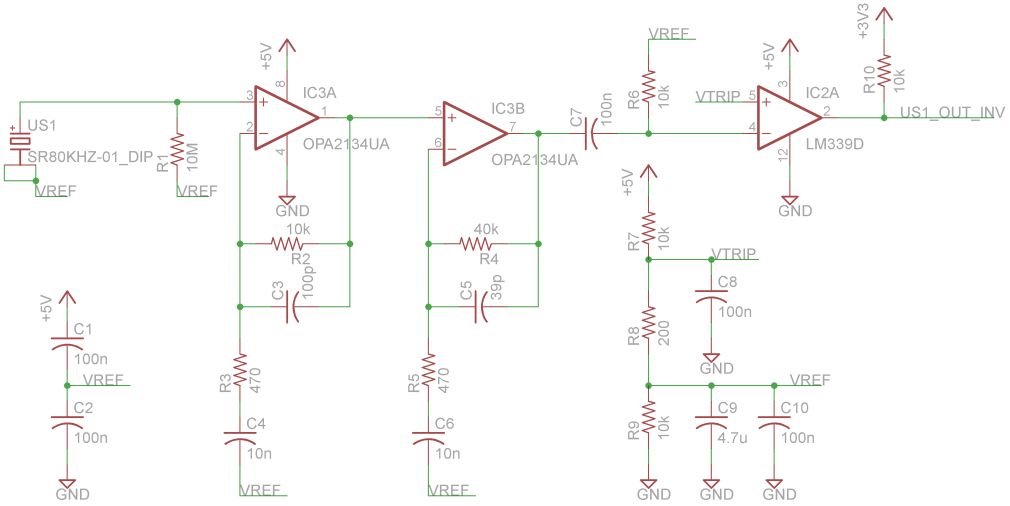

Figura 2.30: Esquemático del receptor de ultrasonido utilizado

Seguimiento de la señal del transductor

Al recibir la señal transmitida de ultrasonido el transductor genera una señal del estilo a la de la figura 2.28 centrada en Vref. Esta señal es amplificada en dos etapas por el circuito formado con el OPA2134. La primera etapa tiene una transferencia $H_1(s) = v1/vin$ de la forma:

$$H_1(s) = \frac{s^2 + s(\frac{C_4R_2+C_3R_2+C_4R_3}{R_2R_3C_4C_3}) + \frac{1}{R_2R_3C_4C_3}}{s^2 + s(\frac{C_3R_2+C_4R_3}{R_2R_3C_4C_3}) + \frac{1}{R_2R_3C_4C_3}}$$

donde $v1$ es el voltaje en el pin 1 del operacional y vin es el voltaje generado por el transductor. Su gráfica de ganancia se muestra en la figura 2.31.

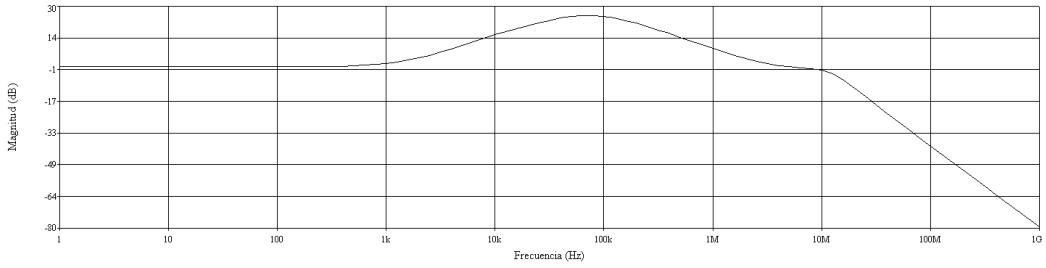

Figura 2.31: Transferencia de la primer etapa

Para la segunda etapa se tiene una transferencia $H_2(s) = v7/v1$ de la forma:

$$H_2(s) = \frac{s^2 + s(\frac{C_6R_4+C_5R_4+C_6R_5}{R_4R_5C_6C_5}) + \frac{1}{R_4R_5C_6C_5}}{s^2 + s(\frac{C_5R_4+C_6R_5}{R_4R_5C_6C_5}) + \frac{1}{R_4R_5C_6C_5}}$$

donde $v7$ y $v1$ corresponden a los voltajes en los pines 7 y 1 respectivamente. La gráfica correspondiente de ganancia se muestra en la figura 2.32.

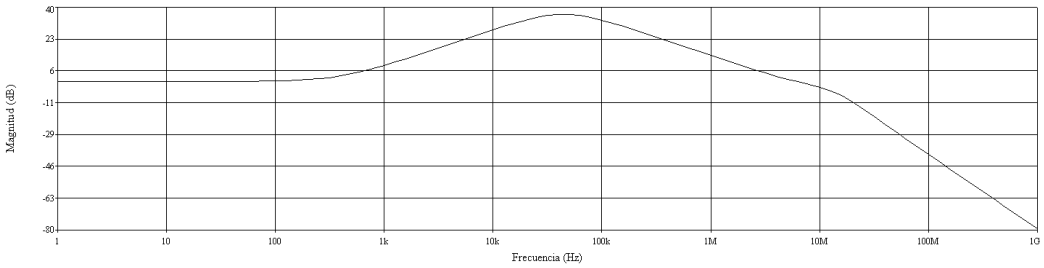

Figura 2.32: Transferencia de la segunda etapa

De esta forma, realizando el producto de ambas transferencias, a la salida de la segunda etapa en el pin 7 del operacional, tenemos una transferencia total correspondiente con la gráfica que aparece en la figura 2.33, donde se ve que la ganancia total máxima es de 61dB con un ancho de banda de aproximadamente 70kHz, centrado cerca de los 60kHz, por lo que para nuestro rango de funcionamiento entre 80kHz y 90kHz, se consigue una amplificación bastante buena de la señal.

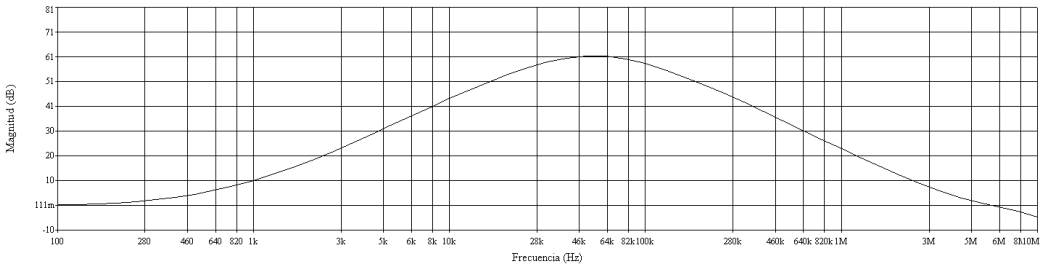

Figura 2.33: Transferencia total del amplificador de ultrasonido

En este punto, tenemos la señal generada por el transductor amplificada cerca de $10^{6,1}$ veces y centrada en Vref en la entrada negativa del comparador LM339. Esto es dado que en continua, el condensador C7 se puede ver como un circuito abierto y como el comparador tiene una resistencia de entrada de gran valor, podemos suponer que por R6 no hay corriente y por lo tanto no hay caída de voltaje, fijando la entrada correspondiente al pin 4 del LM339

en Vref. A su vez, en alterna, el condensador C7 actúa como un cortocircuito y R6 está a tierra, por lo que se copia la señal de la salida de la segunda etapa en la entrada negativa del comparador y como la otra entrada positiva (pin 5), está fija en Vtrip (voltaje del umbral), cualquier señal de amplitud mayor a la diferencia entre Vtrip y Vref, es emitida a la salida como un pulso digital activo por nivel bajo.

C1 y C2 se encuentran para estabilizar la señal Vref y eliminar ruidos, ya que cualquier señal generada en alterna es enviada a tierra y a Vcc.

Esta configuración explicada es idéntica para ambos receptores de ultrasonido y comparten el sub-bloque responsable de generar Vref y Vtrip. R8 establece la diferencia entre los valores de estos voltajes, por lo que es la responsable de fijar el umbral. Experimentalmente se pudo comprobar que el valor sugerido es el adecuado para dicho propósito, ya que filtra la interferencia presente cuando no se recibe señal y deja pasar el primer pulso recibido (ver figura 2.28) para emisiones ubicadas en cualquier lugar dentro del área de cobertura buscado. Los condensadores cumplen la función de desacoplar ruidos y estabilizar los voltajes.

Recepción de Infrarrojo

Para la recepción de infrarrojo, la hoja de datos de la punta emisora PT80KHZ-01 [4] recomienda que el detector posea características similares a las del emisor: i.e. ángulo de visión amplio, alta sensibilidad y una curva de respuesta adecuada para usar a una longitud de onda de 940nm. Además, sugiere utilizar un fotodiodo BPW83 [49] de Vishay Semiconductors que además de ajustarse a lo anterior, trae incorporado un filtro de luz solar.

Sin embargo, se optó por utilizar un módulo receptor de infrarrojo TSOP-4836 [50] como el que se muestra en la figura 2.34 dado que estos pequeños integrados también poseen características similares a las del BPW83, adicionalmente incorporan internamente el circuito necesario para filtrar y demodular la señal de infrarrojo recibida y dan directamente una salida digital ahorrando la necesidad de implementar estos circuitos.

Estos módulos son usados comúnmente en televisores y sistemas que utilicen controles remotos, se alimentan con 5V, tienen alta inmunidad al ruido y distorsiones y emiten un pulso digital activo en bajo al recibir una ráfaga de entre 10 y 70 pulsos de infrarrojo centrados en una frecuencia de

Figura 2.34: Módulo receptor de infrarrojo

36kHz. Pueden recibir ráfagas mayores a 70 pulsos pero luego exigen períodos de inactividad de al menos 4 veces la duración de la ráfaga.

Internamente implementan el diagrama de bloques que se muestra en la figura 2.35 donde se ve como un AGC (Automatic Gain Control o control automático de ganancia) fija la ganancia de un filtro pasabanda centrado a la frecuencia de la portadora de 36kHz, para luego demodular la señal recibida que lleva esta portadora y generar el pulso digital de salida correspondiente. Desde que la señal de infrarrojo transmitida alcanza el módulo hasta que se genera el pulso digital, transcurre un tiempo fijo de aproximadamente 218μs.

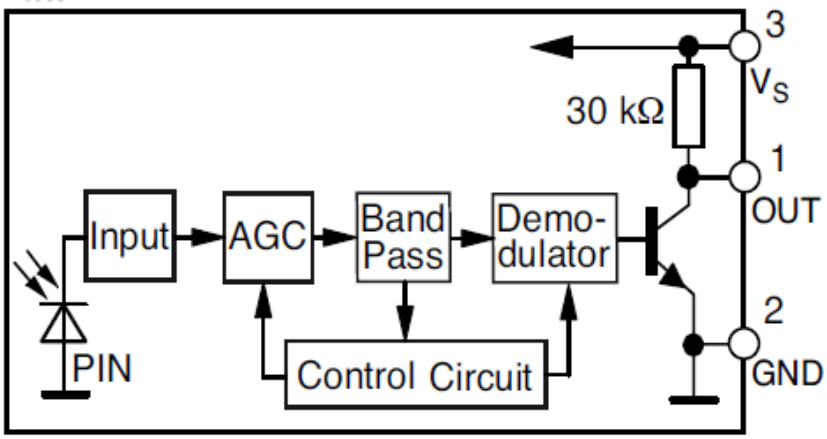

Figura 2.35: Diagrama de bloques interno del TSOP4836. Fuente hoja de datos del TSOP4836 [50].

Para incorporar el módulo al circuito receptor se utiliza el circuito indicado en la figura 2.36.

En la hoja de datos del módulo se recomienda utilizar una resistencia

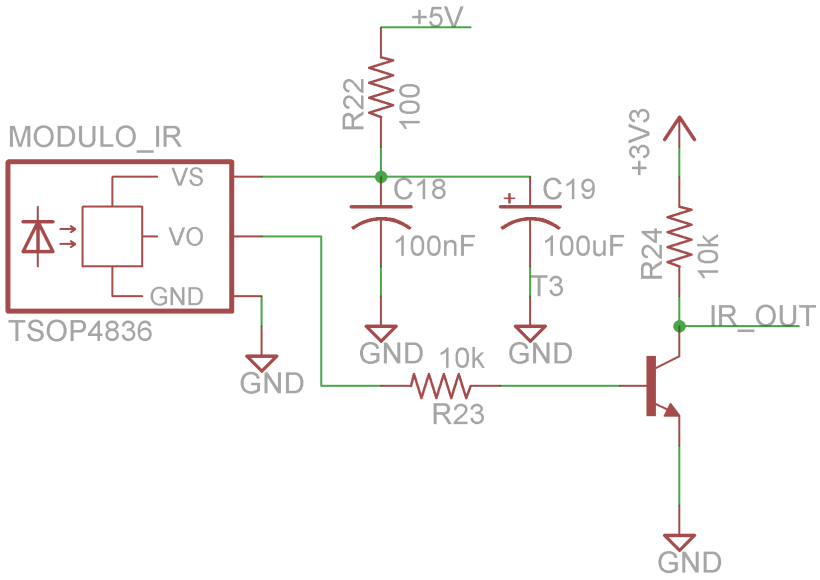

Figura 2.36: Esquemático del bloque receptor de infrarrojo

de 100Ω junto con un capacitor de 4,7μF para filtrar perturbaciones en la alimentación. Pero experimentalmente se observó que al utilizar el puerto USB de ciertas PCs como fuente, el valor del condensador sugerido no era suficiente para que el ruido no se reflejara en la señal de salida como un incremento de jitter, por lo que fue sustituido por el paralelo de uno de 100μF con uno de 100nF de forma de suprimir las componentes principales de ruido y las de alta frecuencia respectivamente.

La salida digital es invertida a través de un BC547 [10] o con un MMBT-3904 [11] en el caso de montage superficial, de forma de adecuar la señal para ser usada con el microcontrolador.

Lógica y comunicación USB

El microcontrolador del receptor es el encargado de implementar la lógica correspondiente a la obtención de las diferencias de tiempo entre la llegada del pulso periódico de posicionamiento de infrarrojo y la llegada del pulso de ultrasonido a cada uno de los dos transductores correspondientes. Para esto utiliza dos timers que cuentan dichos intervalos y luego estos datos son enviados hacia el driver que se encuentra en la PC anfitriona a través del puerto USB. Además, se encarga de decodificar la señal de infrarrojo para

determinar el pulsado de los botones del lápiz y enviar esa información junto con la de los contadores.

El integrado utilizado para dicho propósito es el PIC32MX460F512L [51]. Es capaz de contar ambos tiempos de forma simultánea e independiente a una velocidad de 80MHz por medio de sus dos timers de 32 bits. Tiene implementada una interfaz USB que permite el uso del integrado como *USB device* y puede funcionar en modalidad bus-powered. Además, cuenta con muchas características adicionales dentro de las que se destaca una capacidad de memoria de 512KB y una RAM de 32KB.

Este integrado fue elegido principalmente porque cumple todos los requerimientos necesarios inclusive con cierta holgura, es relativamente de bajo costo, se cuenta con mucha información disponible del mismo, cuenta con varias ventajas respecto a su programación y el kit de desarrollo utilizado es relativamente económico y amigable.

Por más detalles respecto al proceso de elección, referirse al documento "Elección del chip para el receptor.pdf" que se encuentra adjunto en el CD [52].

Dimensionado del Microprocesador

Para cumplir con las especificaciones buscadas, es necesario que el microprocesador cuente con ciertas características mínimas las cuales se detallan en el documento "Elección del chip para el receptor.pdf" que se encuentra adjunto en el CD [52].

A continuación se explica cómo se determinó la frecuencia de los contadores y el tamaño en bits de estos. Adicionalmente se establece un criterio teórico para la separación que deben tener los transductores de ultrasonido.

De las especificaciones vistas en la sección 1.2.1 tenemos que la resolución deseada es de 200 DPI, lo que implica que se desea poder detectar puntos contiguos separados 0.127mm entre sí. De esto sale que la mínima distancia que queremos detectar equivale a la diferencia en el peor caso entre un punto y un receptor de ultrasonido y el punto contiguo y el mismo receptor de ultrasonido. Considerando nuestro receptor y su área de cobertura máxima, se puede demostrar que el peor caso se da cuando el punto origen se encuentra

en la mediatriz de los sensores de US, ubicado lo más separado posible de los sensores y éste se mueve perpendicularmente al eje mencionado. El sentido de movimiento que tome es indistinto por la simetría de la configuración. Esto se puede ver mejor en la figura 2.37.

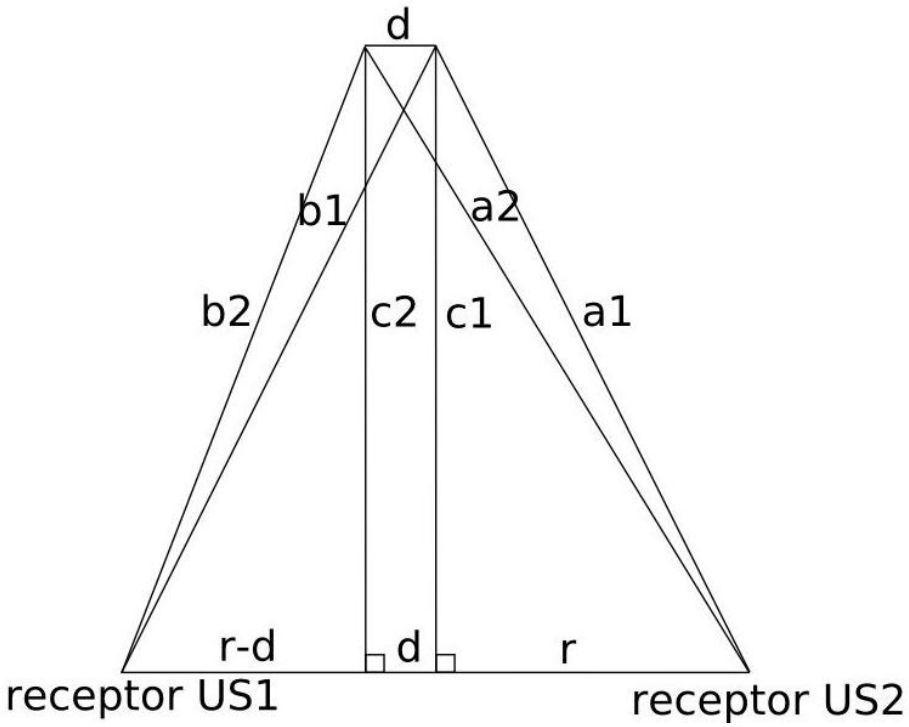

Figura 2.37: Mínima distancia deseada de detección

Una forma de ver que este es efectivamente el peor caso sin la necesidad hacer cuentas, es dibujar muchas circunferencias con radios que difieren en una distancia fija y concéntricas con ambos sensores y observando que en el punto mencionado es donde se genera la mayor fuente de incertidumbre si quisiéramos detectar ese punto sólo conociendo los radios de las circunferencias concéntricas a los sensores y el error en la medida de este.

Volviendo a la figura 2.37, nuestros puntos contiguos se encuentran separados d = 0.127 mm, y están a c1 = c2 = 30.5 cm separados con respecto a la recta que une US1 con US2. De acá se obtiene que la mínima distancia que tenemos que detectar es b1 - b2 (a2 - a1 es un poco mayor, esto se puede ver sustituyendo por valores).
Entonces:

$$b1 - b2 = \sqrt{c1^2 + r^2} - \sqrt{c1^2 + (r - d)^2}$$

Se ve claramente que la mínima distancia a medir crece a medida que separamos los receptores entre si, por lo que tomamos r = 5 cm de forma de que sea lo más grande posible y que el receptor no sea algo incomodo de manipular (recordar que va sobre una XO).

De esta forma:

$$b1 - b2 = 20\mu\text{m}$$

Teniendo en cuenta la tabla 2.1 y tomando un peor caso de 40ºC de temperatura ambiental, tenemos que la mínima diferencia de tiempo que debemos medir es 57.8ns, lo que equivale a una frecuencia de 17.3MHz y para poder muestrearlo minimizando los errores, tomamos un reloj 3 veces mayor con lo que resulta en un muestreo de 51.9MHz.

Temperatura (ºC)	Velocidad en el aire (m/s)
0	332
10	338
20	344
30	349
40	355

Cuadro 2.1: Relación entre temperatura y velocidad del ultrasonido en el aire [53] [54]

De acá sale que precisamos un muestreo de 50MHz por parte del microprocesador.

Si eligiéramos r = 3 cm, siguiendo el mismo razonamiento llegaríamos a que precisamos un muestreo de 85.8MHz.

Para ver el tamaño de los contadores en bits, se toma la mayor distancia a la que puede estar el punto a medir dentro del área de cobertura en relación a uno de los sensores de ultrasonido. Considerando el receptor centrado en el borde de 23 cm y tomando el vértice opuesto al sensor a considerar, se ve que la máxima distancia cumple:

$$l_{max} = \sqrt{(0{,}115 + 0{,}05)^2 + 0{,}305^2} = 0{,}3468m$$

Entonces, el tiempo máximo en el peor caso sería: $t_{max} = 0{,}001044578s$ Un contador de 16 bits nos permitiría contar $2^{16} = 65536$ Entonces, para una frecuencia de 50MHz, tengo un período de 20ns por lo que podría contar hasta $N_{max} = 52229$, por lo que 16 bits alcanzarían pero no 15, ya que $2^{15} = 32768$.
Ahora, con 16 bits, y con ese tiempo máximo, puedo alcanzar una frecuencia de hasta 62,7MHz, por lo que para frecuencias mayores precisamos contadores de mayor tamaño.

Asumiendo que el ultrasonido y el infrarrojo funcionaran correctamente en superficies mayores se podría aplicar el mismo razonamiento, lamentablemente no es el caso.

El integrado finalmente utilizado cuenta con contadores de 32 bits y una frecuencia de muestreo de 80MHz por lo que cumple los requerimientos mínimos.

Adecuación de Señales

Para realizar el conteo de los tiempos, los timers de 32 bits del PIC32MX-460F512L [51] sólo comienzan a contar cuando detectan un flanco de subida en el pin correspondiente al timer y permanecen contando, incrementando su valor en cada flanco del reloj de 80MHz, hasta que detectan un flanco de bajada en dicho pin. Para esto, fue necesario incorporar una compuerta lógica AND HCF4081B [55] con la señal de salida del infrarrojo y cada una de las señales de salida del ultrasonido de manera que los timers contaran el tiempo entre la llegada del infrarrojo y cada señal de ultrasonido. La salida del AND va a los timers 2 y 4 por los pines T2CK y T4CK del PIC32. De esta forma comienzan a contar con el flanco de subida de la llegada de infrarrojo, hasta el flanco de bajada correspondiente a la llegada de ultrasonido. Recordar que el circuito de infrarrojo pone un nivel alto cuando recibe señal y que el de ultrasonido pone un nivel bajo. El timer es reseteado luego por software.

Se utiliza la compuerta HCF4081B dado que es la que mejor se ajustaba a las necesidades dentro del mercado local. El integrado trae 4 compuertas, por lo que para optimizar, como trabajo a futuro se debería determinar una compuerta más adecuada.

Las dos salidas de los circuitos de ultrasonido se invierten para llevarlas

a los pines int1 e int4, los cuales escriben la bandera correspondiente. Los inversores son iguales al del circuito de recepción de IR.

Dado que el PIC32MX460F512L requiere una alimentación externa de 3.3V, se agrega un regulador de voltaje lineal LP2950CZ3.3 [20] de igual forma que en el emisor. Dicho voltaje se utiliza para alimentar los inversores y el AND de forma de que el PIC32 vea voltajes de entrada de 3.3V y a modo de reducir consumo.

Conexionado

Para la incorporación del PIC32 a la placa receptora, según su hoja de datos [51], deben alimentarse todos sus pines V_{DD} en 3.3V y colocarse un capacitor de 100nF entre V_{DD} y tierra. También deben conectarse a tierra todos sus pines V_{SS}. Debe además conectarse un oscilador de cristal de 8MHz en los pines del oscilador principal en paralelo con una resistencia de 1MΩ y capacitores de desacople de 22pF. Este oscilador se utiliza para generar mediante PLL los 80MHz que es la frecuencia de operación del PIC32 y además se generan los 48MHz necesarios para el funcionamiento del módulo USB interno del PIC32.

Se incorpora además a la placa un pulsador que se conecta al pin MCLR para la implementación del reset. A la salida del pin 17 del PIC32 se conecta un LED que indica el correcto funcionamiento del circuito. Además se incorpora un conector hembra USB para el funcionamiento de la interfaz USB. Del pin VBUS de este conector se obtienen 5V que luego del conversor LP2950 se convierten a 3.3V para la alimentación del PIC32 y el resto de la placa receptora.

La señal de infrarrojo se conecta al pin RA7 (pin 92). La señal de salida del receptor de ultrasonido 1 se conecta al pin INT1 (pin18) y la del receptor de ultrasonido 2 al pin INT4 (pin 67). La salida del AND lógico entre la señal de infrarrojo y la señal de ultrasonido 1 se conecta al pin T2CK (pin 6) y la salida del AND lógico entre la señal de infrarrojo y la señal de ultrasonido 2 se conecta al pin T4CK (pin 8).

El núcleo y lógica digital del integrado requieren 1.8V para funcionar los cuales pueden obtenerse externamente o utilizando un regulador interno con tal propósito. En este caso se opta por usar el regulador interno y para ha-

bilitarlo se conecta el pin VREG (pin 86) a V_{DD} y se agrega un condensador de 10uF en paralelo con uno de 100nF en el pin VDDCORE (pin 85) para mantener la estabilidad del regulador.

Para poder realizar una programación "in-circuit" del integrado se agrega un conector RJ12. El conexionado de este junto con el resto del bloque se observa en la figura 2.38.

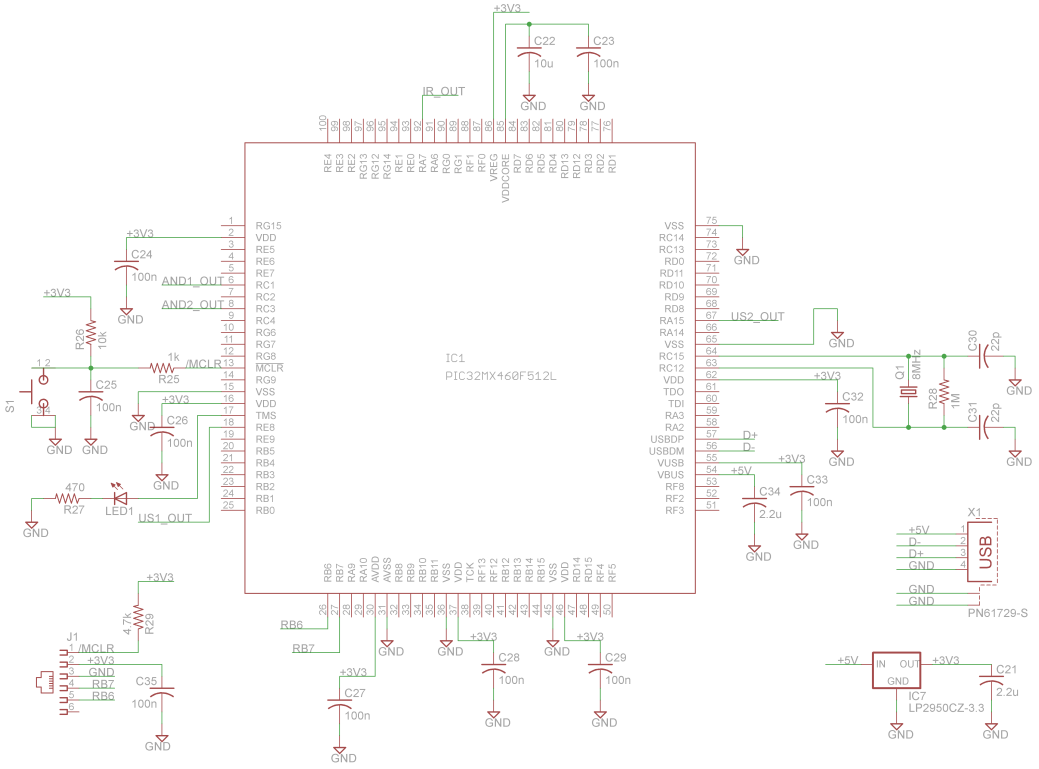

Figura 2.38: Conexionado del Microcontrolador

Programador

El programador que se utilizó fue el MPLAB ICD 3, el cual es capaz de depurar una aplicación si se utiliza con el MLAB IDE que fue el entorno de desarrollo escogido para la elaboración del código.

El MPLAB ICD 3 se conecta al PC a través de una interfaz USB y con el microcontrolador a programar a través de un conector RJ12 de 6 pines,

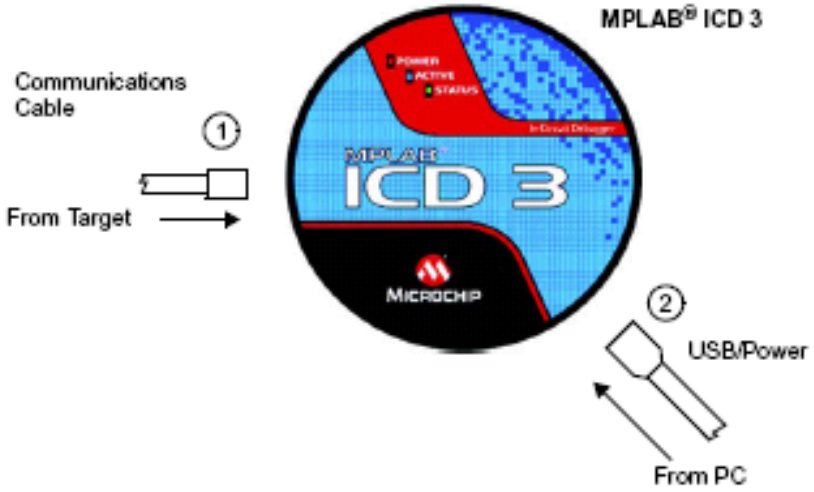

Figura 2.39: MPLAB ICD 3. Figura tomada de MPLAB ICD 3 In-Circuit Debugger User's Guide [56]

aunque sólo utiliza 5 de éstos [57].

El pin 1 del conector RJ12 se conecta al pin MCLR del PIC32 con una resistencia de 4.7kΩ entre éste y VDD. El pin 2 se conecta a VDD con un capacitor de 100nF entre éste y tierra. El pin 3 se conecta a tierra. El pin 4 se conecta al pin RB7 del PIC32, el pin 5 al RB6 del PIC32, y el pin 6 no es utilizado. Este conexionado se puede ver en la figura 2.38.

Este programador fue escogido por ser una fuerte herramienta para depurar, soportando multiples breakpoints y permitir el acceso a los registros al depurador.

Para más información acerca del MPLAB ICD 3 visitar la página de Microchip [58].

2.2.3. Placas Construidas

Al igual que con el emisor, en esta sección se expondrán las placas que componen el primer prototipo funcional y el prototipo final en formato SMD. Dado que las placas correspondientes del emisor y el receptor fueron construidas en paralelo, comparten las mismas características, por lo que el primer prototipo también fue realizado con componentes "through-hole" y en un PCB de Pertinax de una capa, lo que derivó en un gran tamaño, dificultando su uso pero que reúne todas las funcionalidades requeridas del proyecto. En tanto que el prototipo funcional final en formato SMD fue construido en un PCB de fibra de vidrio de doble capa, lo que permitió reducir drásticamente su tamaño y fue posible incorporarle características adicionales.

Los criterios de minimización de interferencias utilizados para la construcción de estas placas son los mismos a los expuestos en la sección 2.1.4.

Primer prototipo

Nuevamente para este primer prototipo se adoptó un diseño simple faz con el objetivo de ser fabricado rápidamente con el método de la plancha y pese a que se trato de minimizar su tamaño para poder fabricarle un sistema de forma de adosarlo a la XO o cualquier pantalla, no se pudo reducir a más que unos 17cm x 14cm (figura 2.40). Los archivos correspondientes del PCB Wizard pueden encontrarse en CD/circuitos/esquematicos y layouts/.

La incorporación del microcontrolador a la placa se hizo mediante una PIM (Plug in module) del PIC32 que permite interconectar el integrado a través de zócalos, por lo que la placa contaba con las 4 tiras de 25 pines necesarias. Para programar el integrado de la PIM se utiliza el kit de desarrollo.

El conexionado de la PIM es igual que el explicado para el integrado salvo que los 4 pines correspondientes al puerto USB están cambiados. Esto fue detectado luego de construida la placa por lo que para que este diseño funcione es necesario sustituir la conexión de los pines 54, 55, 56 y 57 por los pines 1, 2 o 16, 89 y 90 respectivamente.
Además, a este diseño le falta las conexiones del encendido del regulador interno explicado anteriormente.

Además se dejó previsto el espacio y las conexiones necesarias para conectar un oscilador secundario de 32kHz que no es utilizado por el código desarrollado, pero que se dejó como previsión.

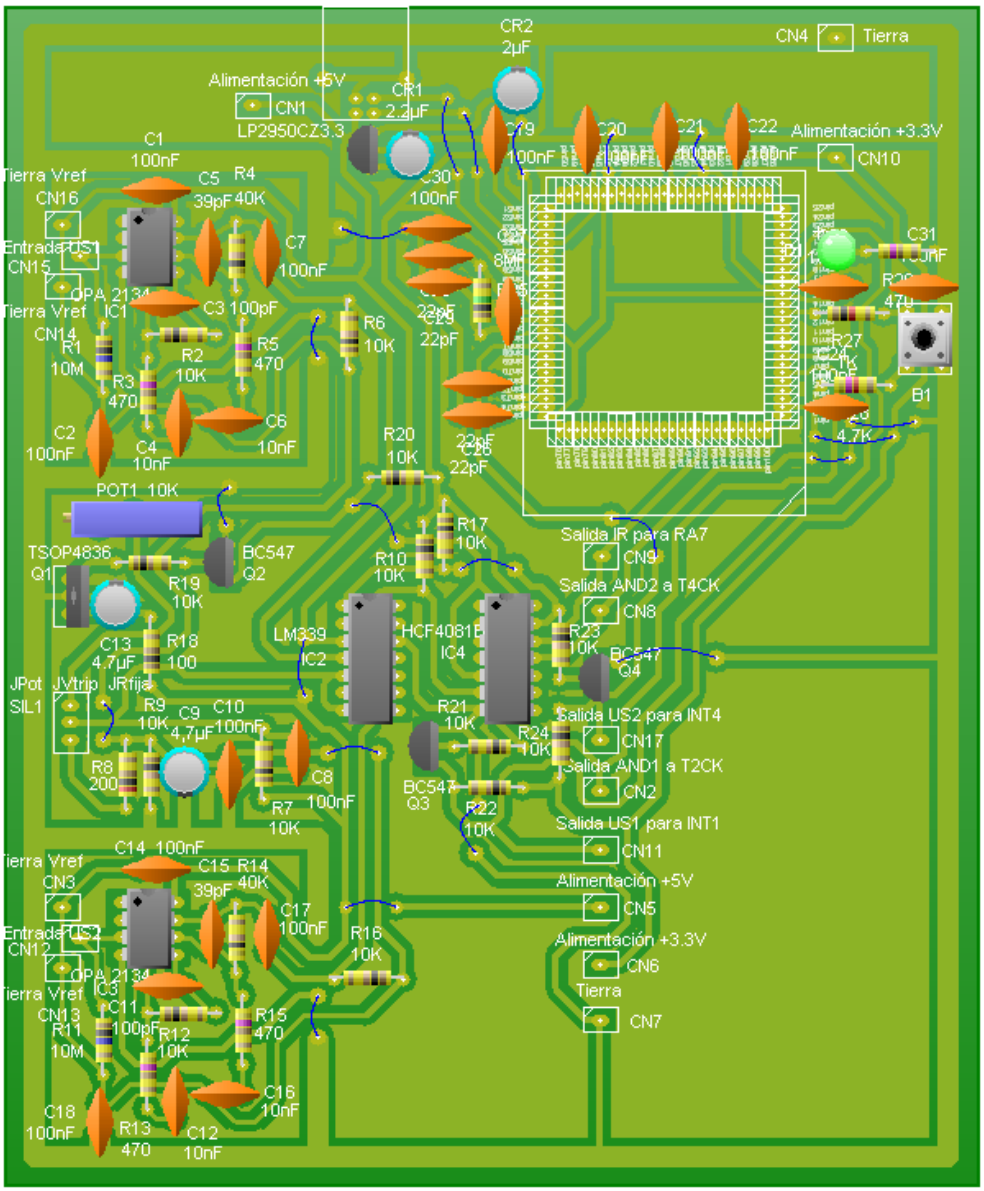

Figura 2.40: Diseño en PCB Wizard de primer prototipo de la placa receptora, dimensiones 17.3cm x 14.3cm.

Para la incorporación del resto de los integrados se utilizaron zócalos de forma de facilitar su extracción en caso de así requerirse,

Como con el emisor, se tomó la precaución de dejar amplios bordes co-

rrespondientes a tierra para poder blindar fácilmente toda la placa en caso de ser necesario. También se dejaron márgenes gruesos de tierra rodeando a los circuitos preamplificadores de forma de poder blindarlos. Se dejaron pines correspondientes a las salidas que van al PIC32 así como de alimentación y tierra, de forma de poder utilizar la placa Explorer16 perteneciente al kit de desarrollo en caso de no utilizar la PIM adosada en la placa. Cuenta con un botón de reset y un LED indicador del encendido del PIC32.

La placa se conecta directamente al puerto USB sin necesidad de ningún otro hardware. El circuito preamplificador se diseño de forma de minimizar las pistas y se colocaron los componentes lo más cerca posible el uno del otro, así como que las pistas relevantes en esta parte son más finas que las otras para minimizar ruidos.

Los sensores receptores de ultrasonido se colocaron a 10cm de distancia uno del otro y en el medio se puso el módulo receptor de infrarrojo. La separación de los sensores receptores se estableció en 10cm con el criterio de maximizar la exactitud en las medidas y siendo esta una distancia tolerable en cuestión de tamaño. Partiendo de esto, fue que se diseñó de forma de minimizar el tamaño del resto de la placa.

El circuito generador del Vtrip y Vref se colocó en el medio y se puso un jumper de forma de seleccionar la resistencia fija R8 de 200Ω o un potenciómetro de $10k\Omega$ de forma de poder variar el umbral.

Dado que esta placa fue utilizada exclusivamente para realizar pruebas se utilizó un blindaje temporal de papel aluminio que cubría toda la capa de cobre y parte de la cara de arriba. Blindando sólo el bloque de amplificación de ultrasonido no se consiguieron buenos resultados.

La placa construida se muestra en las figuras 2.41 y 2.42.

Prototipo en SMD

Junto con la construcción del emisor en SMD se realiza también la versión correspondiente del receptor de forma de encapsularlo y construirle un sistema de enganche que permita adosarlo a la XO, a un monitor LCD o usarlo sobre cualquier superficie lisa horizontal.

La principal limitante para reducir el tamaño del receptor una vez que

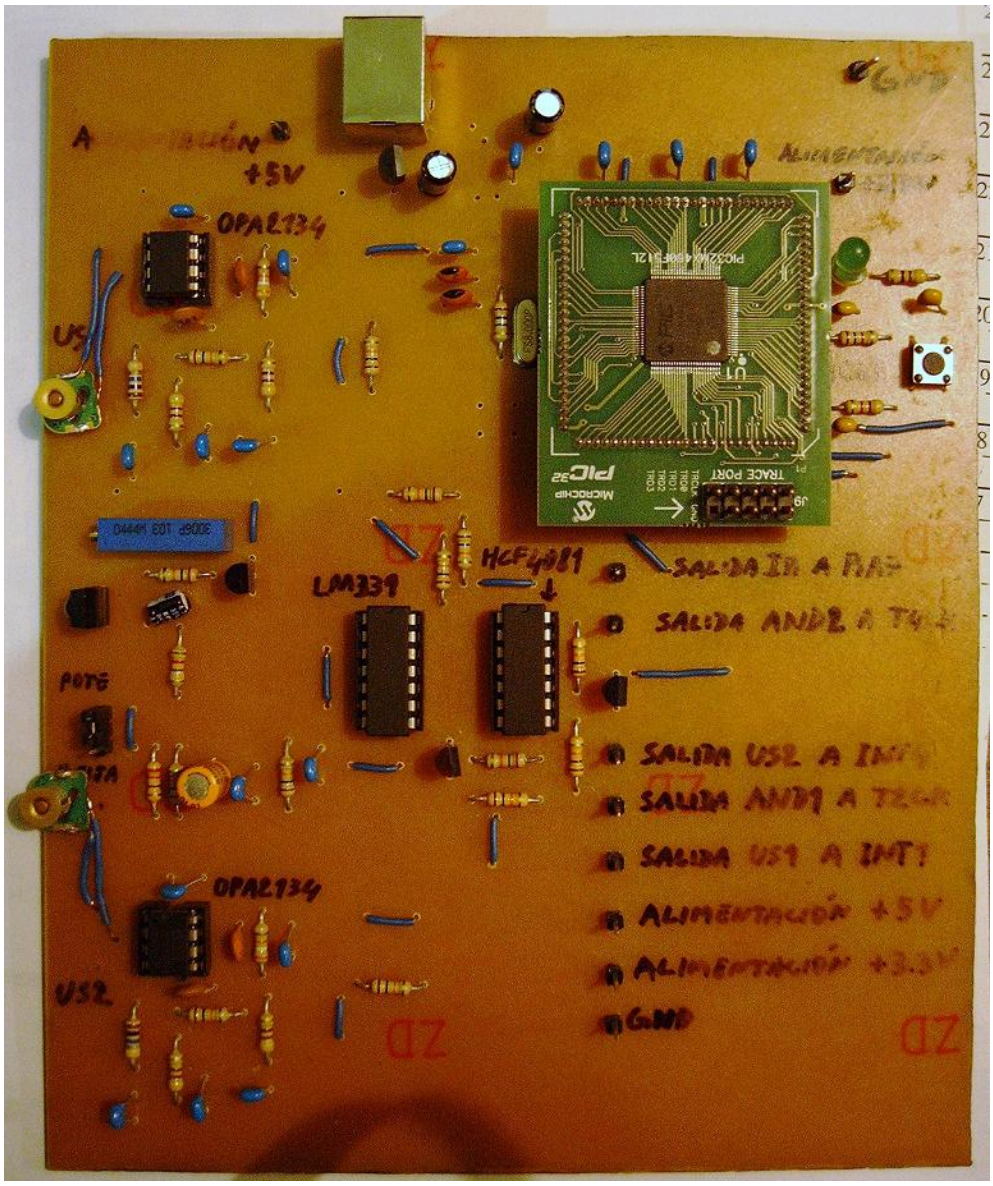

Figura 2.41: Receptor primer prototipo visto de arriba

se considera un diseño doble capa con componentes en SMD es la separación relativa de los transductores piezoeléctricos de ultrasonido. Otros dispositivos similares, como la Pegasus NoteTaker que incorporaba la Classmate, poseen una separación relativa menor a los 6cm, por lo que se consideró que los 10cm seleccionados podían llegar a reducirse, pero hasta ese entonces no se había probado el funcionamiento del prototipo con otras distancias.

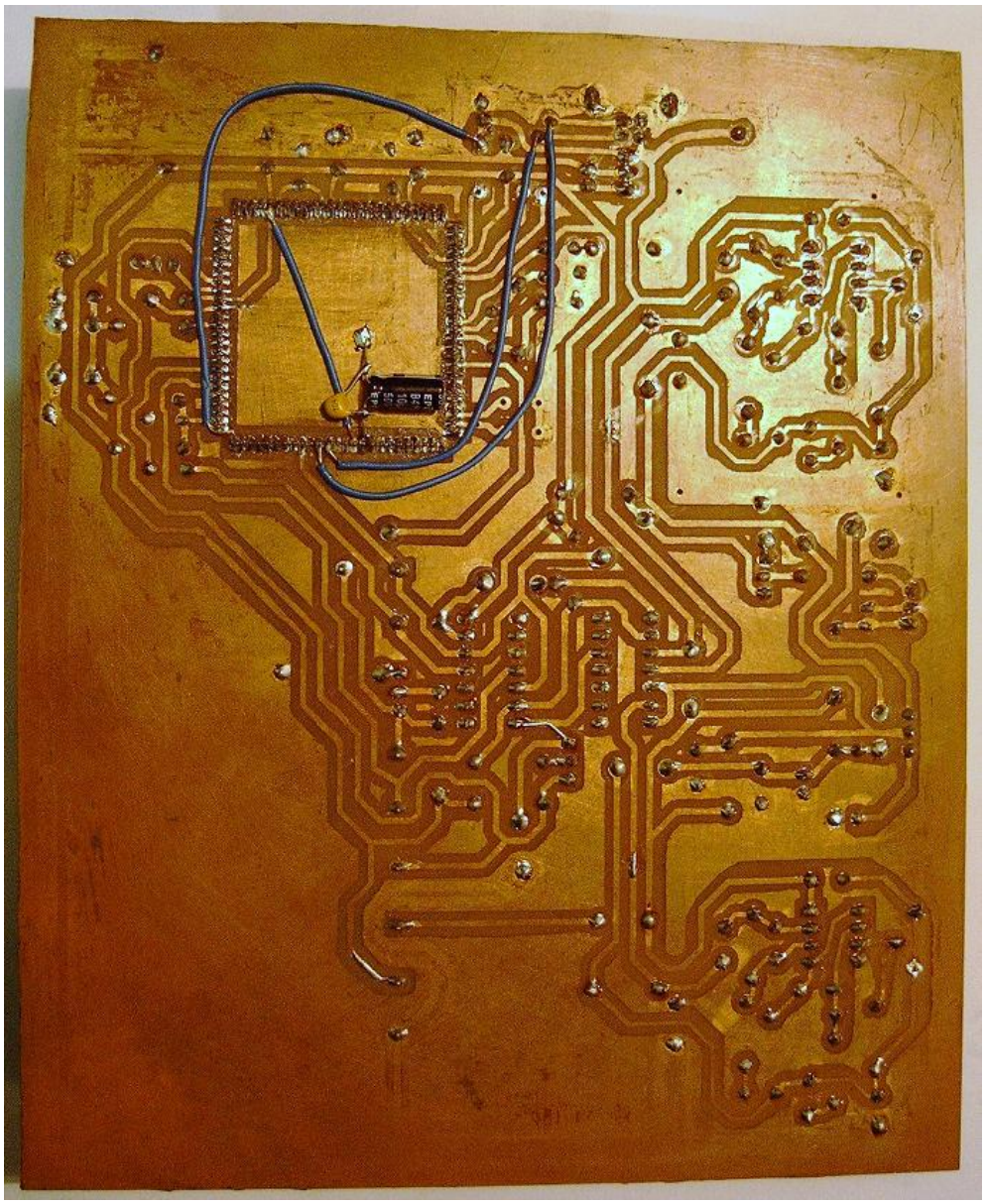

Figura 2.42: Receptor primer prototipo visto de abajo

Al momento de diseñar el prototipo en SMD la placa receptora del primer prototipo se encontraba en uso, en pruebas para tratar de reducir el jitter presente en la recepción de infrarrojo, por lo que no era posible ni conveniente realizar pruebas con distintas separaciones para ver si la diferencia de distancias generaba un incremento en el error de detección que fuera relevante. Apoyándose en que otros dispositivos comerciales funcionan con

menos de 6cm de distancia relativa, se optó por realizar un primer diseño en base a una separación de 6cm de los sensores y antes de construir la placa verificar el buen funcionamiento.

Lamentablemente, al realizar las pruebas pertinentes se observó que para distancias de 8cm o 10cm no presentaban diferencias pero para 6cm hay un incremento del error notorio y como ya se estaba por debajo del umbral de tolerancia de error, se elabora el segundo diseño a partir del anterior el que finalmente se construye.

Primer diseño

Para generar el diseño se parte de una separación de los transductores de ultrasonido de 6cm y se ubica el módulo receptor de infrarrojo en el medio de estos. Luego se sigue el diseño tratando de minimizar su tamaño y al mismo tiempo inmunizarlo al ruido electromagnético.

Con este criterio se deja la mayor cantidad de superficie de tierra posible de forma reducir al máximo las interferencias. A su vez, esta placa pude blindarse completamente quedando expuestas sólo unas pocas pistas cercanas al módulo receptor de infrarrojo, a las cuales la interferencia electromagnética no les afecta.

Para realizar esto, como sucede con el resto de las placas expuestas, se deja un borde externo de tierra lo suficientemente grueso como para poder blindar sin problemas toda la capa inferior de la placa, la que contiene la mayor cantidad de componentes e incluye los circuitos más sensibles al ruido.

Con la capa superior representada en rojo en la figura 2.43, se deja la zona de adelante donde se encuentran los sensores, los LEDs y el botón, cubierta por una capa de tierra. A partir del trazo gris que la atraviesa de izquierda a derecha, el que representa una chapa conductora, se puede blindar rodeando el resto de los componentes y pistas expuestos de la placa. La circunferencia de color gris ubicada al fondo a la derecha, representa una chapa conductora cilíndrica que deja un orificio en el blindaje y el encapsulado de forma de poder introducir el lápiz para cargar la batería con el conector macho del centro.

El diseño completo puede verse en la figura 2.43 donde aparece una zona de color grisáceo que rodea la placa para representar la ubicación del blindaje

y del encapsulado.

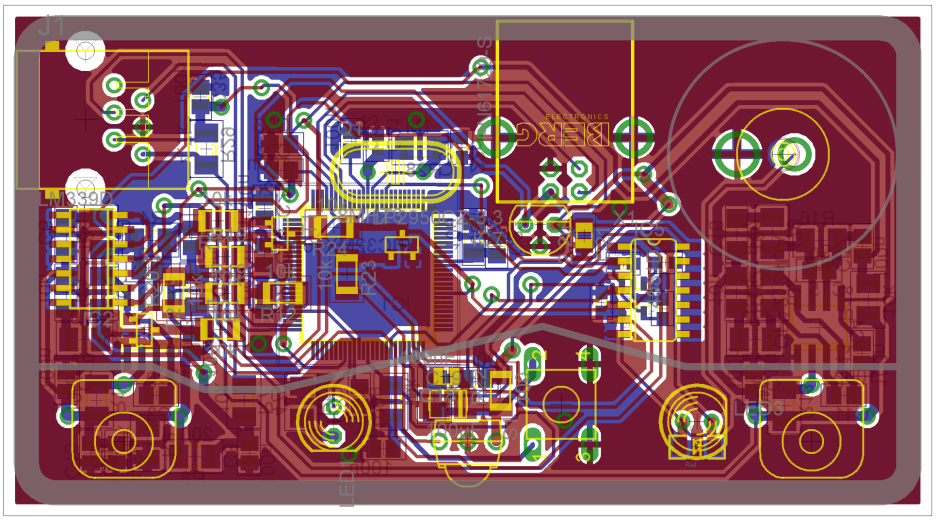

Figura 2.43: Layout del primer diseño del receptor en SMD, dimensiones 81.0mm x 44.5mm.

Los archivos Eagle de este Layout pueden encontrarse en CD/circuitos/-esquematicos y layouts/.

Los bloques del circuito fueron distribuidos de forma de minimizar la cantidad de orificios metalizados (puentes entre capas) y al mismo tiempo, se colocaron de forma que cada señal de interés tenga uno de manera de poder utilizarlo como puntos de prueba para verificar el correcto funcionamiento del circuito. Los orificios metalizados sobrantes son principalmente conexiones de tierra lo que le da mayor robustez.

Los receptores de ultrasonido a pesar de que son SMD, se colocan soldándoles alambres de forma de atravesar la placa. En la capa inferior se ubican los circuitos de amplificación correspondientes a cada uno de ellos, lo más cercano posible a estos. Esto se hace para mejorar el blindaje de estos bloques que son los más sensibles al ruido y disminuir la distancia entre el receptor y el amplificador.

Separación de receptores de ultrasonido

Dado que se requería minimizar al máximo las proporciones del receptor, se analizó qué tanto se podían acercar los receptores de US, de forma de

reducir el tamaño pero sin perder precisión.

Cuando se dimensionó el microcontrolador del receptor (sección 2.2.2), se vió que en un peor caso, con un criterío en que se toma un reloj con período 3 veces menor que el mínimo tiempo a medir, se necesita una frecuencia de 85.8MHz para una separación de receptores de 6 cm. Dado que los contadores utilizados son de 80MHz, en condiciones normales y utilizando un creterio de elección del reloj no tan conservador, se puede considerar que el receptor puede funcionar perfectamente si la distancia de los sensores es 6cm. Sin embargo, dada la existencia del jitter mencionado en la sección 6.1 se hace necesario relevar empíricamente el desempeño del prototipo si se reduce la distancia. En esencia se comparará el desempeño del prototipo a diferentes distancias y se tomará la mínima que funcione mejor.

Se tomaron como distancias posibles entre sensores 2 opciones: 8cm y 6cm. El primer prototipo fue construido a 10 cm por lo que las medidas que se tomen se compararán con éste al momento de evaluar el desempeño.

Como forma de evaluar el desempeño a las distancias dadas, se tomaron muestras con el prototipo. El tomado de muestras consiste en dejar la punta estática en posiciones estratégicamente elegidas y luego hallar la varianza de las muestras. La distancia que presente menor varianza general será la elegida para la construcción.

Cabe destacar que las pruebas de separación de los transductores de ultrasonido se realizan soldándole alambres largos a los sensores del primer prototipo de forma de poder variar su distancia relativa. Estos alambres captan demasiado ruido electromagnético, por lo que se vuelve dificultoso obtener señales limpias. Es por lo tanto necesario "limpiar" las señales antes de hallarles la varianza, de lo contrario se obtendría un valor que no es representativo del sistema.. El limpiado consiste simplemente en graficar las muestra y descartar las que quedan evidentemente fuera del promedio.

Muestreado y análisis

Se toman las muestras en 16 puntos. Básicamente, estos puntos comprenden todas las esquinas, el contorno y el interior del área de trabajo, es decir un tamaño equivalente a una hoja A4.

Para tomar las muestras se utiliza una versión especial del driver (versión 1.51 para muestras). Esta no requiere calibración, ya que se define un área de trabajo particular (análoga al caso de trabajar fuera de pantalla) y además arroja todos los valores de posición en pantalla. Esta versión puede encontrarse dentro del CD en CD/software/Driver/Driver_tot_v1.51_para_muestras/. Luego al ejecutar el driver se hace que todas las salidas vayan a un archivo logrando así guardar todas las muestras tomadas. El muestreado se realiza por varios segundos de manera de asegurarse de tener una cantidad representativa de valores

Luego de tomar las muestras en los puntos elegidos, se procede al análisis de las mismas para lo cual se utiliza Matlab. Como primer paso se procede al "limpiado"de las muestras tal como se mencionaba anteriormente.

Cabe destacar que las muestras fueron tomadas en unidades de pantalla, es decir en pixeles. Para realizar el análisis de la varianza se convierten las mismas a mm realizando la transformada inversa de la implementada en el driver. Los programas utilizados para el análisis de las muestras pueden encontrarse en el CD.

Finalmente, una vez que se prepararon las muestras, se halla la varianza de las mismas y se construye una tabla comparativa. Se halla la varianza en el eje x e y independientemente.

A continuación se muestran algunos puntos evaluados y sus respectivas varianzas:

	varX(mm^2)	varX(mm^2)	varY(mm^2)	varY(mm^2)
PUNTO	$D = 8cm$	$D = 6cm$	$D = 8cm$	$D = 6cm$
punto 6	0,0097	0,04	0,0039	0,0073
punto 7	0,0043	0,0197	0,0037	0,0031
punto 8	0,0124	0,0462	0,0039	0,0162
punto 9	0,1695	0,2223	0,0227	0,0548
punto 10	0,1351	0,1091	0,0076	0,0121
punto 11	0,197	0,2692	0,0244	0,0205
punto 14	0,405	0,1651	0,0095	0,0166
punto 15	0,05050	78,18280	0,00550	38,57600
punto 16	0,03760	104,30930	0,00710	93,98700

Los puntos 15 y 16 son descartados ya que evidentemente el alto valor de varianza que presenta para $D = 6cm$ es debido a algún inconveniente al

momento de tomar las muestras. También se observa que en los puntos 10 y 14 se da una extraña condición en la que con $D = 6cm$ da mejor valor. Se atribuye lo anterior a algún error al momento de tomar la medición.

Recordemos que la idea es evaluar qué tanta calidad se pierde si el receptor se realiza con $D = 6cm$ en lugar de 8. Como puede apreciarse en la tabla, en la mayoría de los puntos con $D = 8cm$ da mejor que con $D = 6cm$ o cual es de esperarse.

Sin embargo, al hallar la varianza promedio se tienen los siguientes valores:

	ejeX	ejeX	ejeY	ejeY
	$D = 8cm$	$D = 6cm$	$D = 8cm$	$D = 6cm$
Varianza (mm)	0,0088	0,0353	0,00383	0,00887
Desviación estándar (mm)	0,09381	0,18788	0,06192	0,09416

De la tabla anterior, se puede observar que si se utilizase como distancia entre sensores 6cm el desempeño se vería afectado dado que en el ejeX la desviación estándar supera el valor de especificación requerido.

Conclusión

Dado el análisis anterior, se elige como distancia para la construcción del segundo prototipo como $D = 8cm$.

Además de basarnos en los valores anteriores, también se observaron gráficas donde se aprecia claramente una mejora al trabajar con $D = 8cm$, algunas de las mismas pueden verse en el anexo E.

Cabe destacar que de no estar presente el jitter mencionado, no deberían haber inconvenientes para realizar el prototipo con $D = 6cm$.

Diseño construido

Finalmente se modifica el primer diseño de forma de incrementar a 8cm la distancia relativa de los sensores de ultrasonido y se construye un diseño que presenta zonas libres del lado derecho y zonas de pistas y componentes amontonados a la izquierda.

Como se vio anteriormente, esta placa cuenta con un LED que indica el buen funcionamiento del microcontrolador y un conector RJ12 para programarlo, un botón de reset, un conector USB del tipo B hembra para conectarlo a la PC y además cuenta con el sistema de carga de baterías explicado en la sección 2.1.3.

Este segundo diseño que posee más espacio se le incorpora la posibilidad de poner un conector DC para utilizar una alimentación externa de 5V para cargar la batería del lápiz más rápidamente. Pero finalmente se optó por no usarlo y simplemente se utiliza el solder-pad dejado con el propósito de poder tomar mayor corriente del puerto USB. Esto fue dado que no se disponía de una fuente adecuada para utilizar con el conector y se consideró mejor usar un cargador USB de los que sí se dispone, el que al conectarlo a un toma AC de 220V genera los 5V del USB.

El diseño final con el cual se construye el prototipo en SMD puede observarse en la figura 2.44.

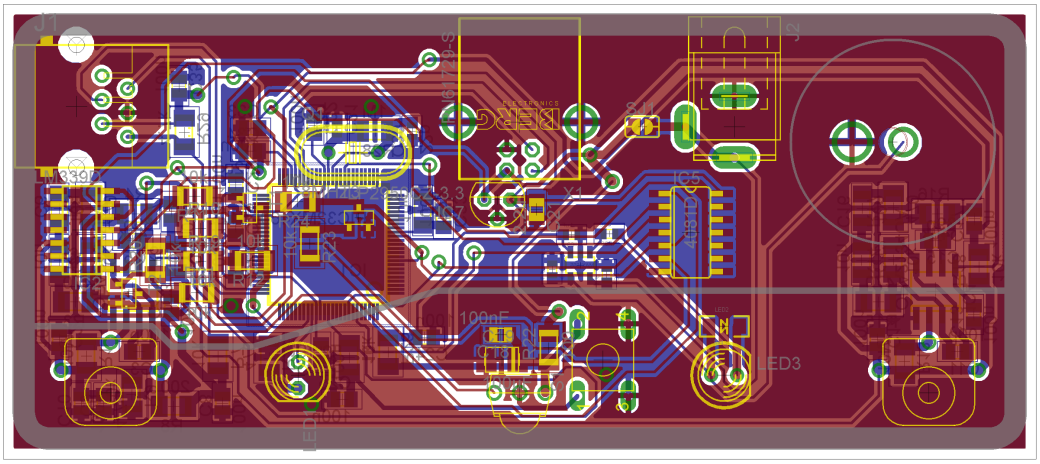

Figura 2.44: Layout del diseño construido del receptor en SMD, dimensiones 101.0mm x 44.5mm.

Los archivos Eagle de este Layout pueden encontrarse en CD/circuitos/-esquematicos y layouts/.

La placa construida se muestra en las figuras 2.45 y 2.46.

Luego de construida la placa se blindó la capa de abajo; la capa de arriba

no fue necesario blindarla para obtener un correcto funcionamiento.

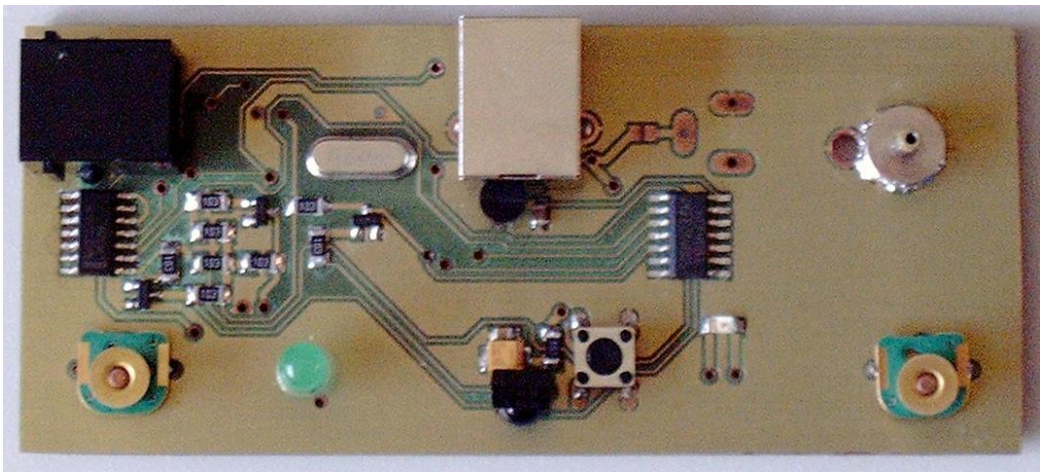

Figura 2.45: Receptor SMD visto de arriba.

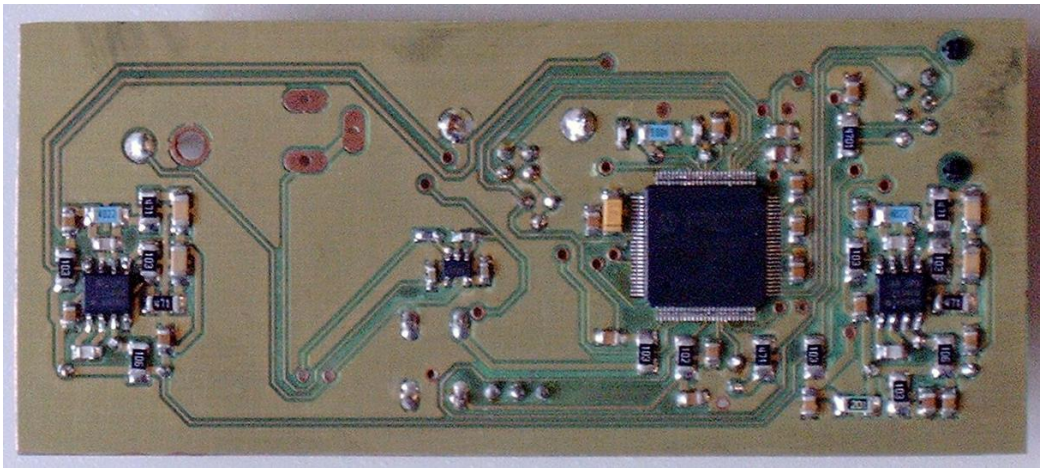

Figura 2.46: Receptor SMD visto de abajo.

2.2.4. Encapsulado

Del mismo modo que sucede con el emisor, se construye un encapsulado para el receptor de forma de protegerlo e incorporarle un sistema de agarre para poder engancharlo a la XO, a un monitor de pantalla plana o para usarlo sobre una superficie lisa.

En la figura 2.47 se ve la última versión alcanzada del encapsulado del receptor a la que lamentablemente le faltó incorporarle el sistema de agarre por falta de tiempo.

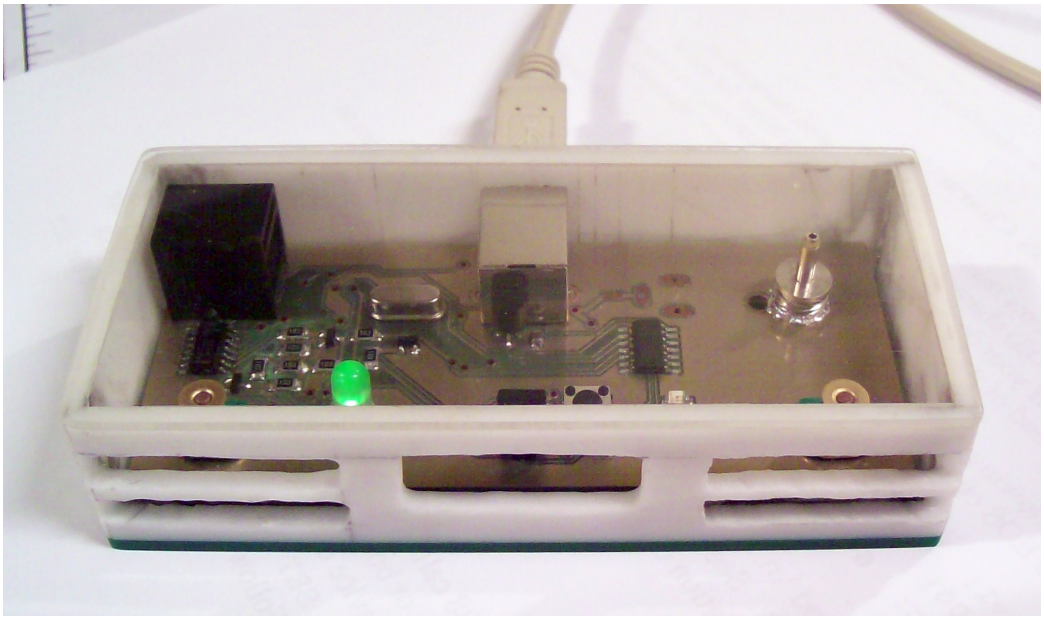

Figura 2.47: Encapsulado del receptor, dimensiones 5cm x 10.9cm x 2.6cm.

Este encapsulado fue construido con acrílico blanco en las paredes, verde en el piso y transparente en el techo removible que auspicia de tapa. Al igual que el emisor, el receptor encapsulado pesa 50gr.

Además del sistema de agarre, faltó generarle una abertura en la tapa para poder insertar el emisor y cargar la batería. Junto con la abertura se debería incorporar topes de forma que el emisor quede firmemente sujeto al receptor.

La pared delantera tampoco tiene la forma deseada, ya que se esperaba generar rejillas horizontales con varias aberturas juntas y barras finas que

permitieran una buena recepción de señal ultrasónica desde toda el área de interés. En la disposición actual se disminuyó levemente el área de cobertura en el ultrasonido por interferencia del propio encapsulado ya que el ultrasonido requiere no tener obstáculos en la línea de vista. Tampoco se le agregó un acrílico transparente (o rojo) protector al receptor de infrarrojo y se dejó al descubierto. Originalmente se esperaba generar una pared curva (o rectangular) para ampliar el rango de visión de los sensores pero no se contó con el tiempo suficiente para construirla.

2.3. Acondicionamiento de Señales

En esta sección se explica cuáles son las señales generadas por el microcontrolador del emisor para excitar los circuitos de infrarrojo y de ultrasonido y a qué se debe que tengan esa forma.

Lo que se busca es poder medir el tiempo que demora la propagación del ultrasonido desde el emisor hasta el receptor y dado que su velocidad de propagación en el aire a una cierta temperatura puede considerarse constante, con eso obtener la distancia a la que se encuentra el emisor respecto del receptor. Además, se desea comunicar al receptor el pulsado de los botones del lápiz.

Para poder medir el tiempo de propagación, la idea básica es emitir simultáneamente la señal de ultrasonido junto con la de infrarrojo y aprovechando que el tiempo de propagación del infrarrojo es despreciable respecto al del ultrasonido, contar desde la llegada de infrarrojo hasta la llegada del ultrasonido. Sin embargo, existen factores que hacen que esto se modifique.

En primer lugar, el modulo de recepción de infrarrojo requiere ráfagas de pulsos a 36kHz para generar un 1 lógico a la salida del módulo receptor. Los trenes de pulsos deben tener entre 10 y 70 pulsos. Si tiene menos de 10 pulsos no es suficiente para excitar el receptor, si se utilizan más de 70 se exigen períodos de inactividad de al menos 4 veces la duración de la ráfaga. Para funcionar a 36kHz deberían generarse trenes de pulsos de 13,88μs en alto y el mismo tiempo en bajo. Como el microcontrolador sólo permite generar trenes de pulsos de largo múltiplo de 1μs se optó por utilizar pulsos de 14μs de largo, lo que corresponde con una frecuencia de 35.71kHz. Dado que la frecuencia utilizada no se aleja demasiado de los 36kHz se puede utilizar de esta forma con iguales resultados.

Otro aspecto a tener en cuenta sobre el funcionamiento del infrarrojo es que hay un retardo en el modulo receptor de 220μs que varía de forma despreciable con el emisor ubicado dentro del área de cobertura. En la figura 2.48 se pueden ver los efectos antes descritos.

Los timers del microcontrolador funcionan en el modo *gated* por el cual se activan cuando en su pin gate hay un 1 lógico y se desactivan cuando este baja a 0. Los timers deben contar el tiempo desde que llega la señal de infrarrojo hasta que llega la señal de ultrasonido. Para poder hacer esto se conecta el gate del timer a la salida de un AND lógico entre la señal de

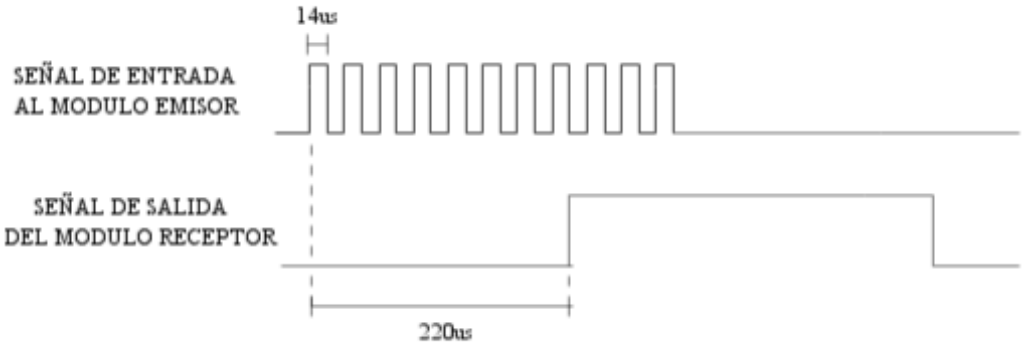

Figura 2.48: Señal de infrarrojo emitida y recibida

infrarrojo y la de ultrasonido, manteniendo la de infrarrojo en alto hasta que llegue la de ultrasonido como se ve en la figura 2.49.

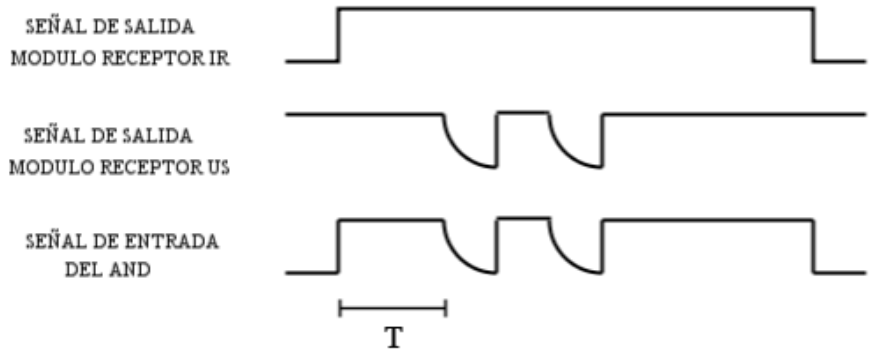

Figura 2.49: Señal de entrada al gate del timer

La mayor distancia entre emisor y receptor que interesa medir es de $34,2cm$, que son 996μs considerando la velocidad del sonido como $343,42m/s$. Es por esto que no tiene sentido enviar una señal de infrarrojo que tenga un largo superior a éste. Para tener un poco de reserva se eligió tomar una señal de 1078μs de largo, formada por un tren de pulsos de 39 pulsos de 14μs cada uno.

Un efecto negativo que se ve en el funcionamiento del ultrasonido es que logra filtrarse en el receptor interferencia electromagnética. Esta interferencia genera perturbaciones en la señal, provocando que el tiempo medido por el microcontrolador no sea el correcto, como se ve en la figura 2.50. Vale aclarar para entender la figura que la señal de ultrasonido se genera en el flanco de bajada del pulso de entrada al circuito emisor de ultrasonido.

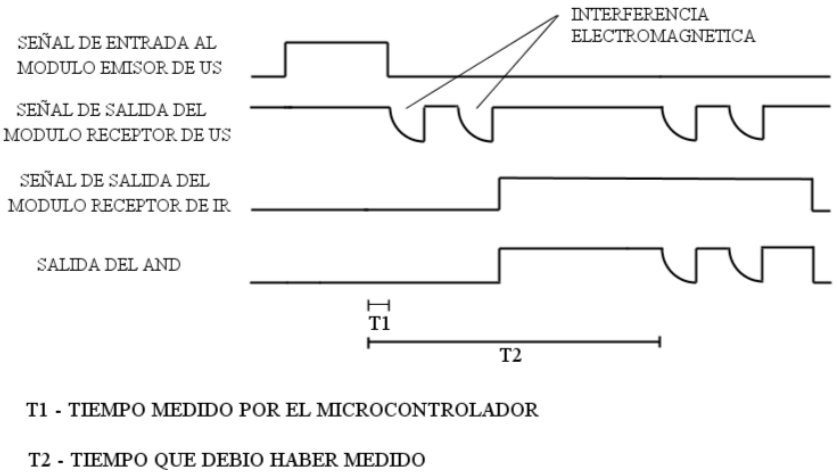

Figura 2.50: Efecto de la interferencia electromagnética

Para evitar ese efecto se retrasa la emisión de la señal de infrarrojo 18μs con respecto a la de ultrasonido y luego este tiempo debe ser agregado al tiempo medido por el microcontrolador. En la figura 2.51 se ve como se logra eliminar el efecto de la interferencia electromagnética.

Visto esto las señales a emitir son las que se muestran en la figura 2.52.

El último efecto a considerar para definir las señales a generar es el jitter en la señal de salida del circuito receptor de infrarrojo. El jitter es provocado porque el AGC del módulo receptor de infrarrojo requiere una referencia de luz para estabilizarse. Cuando se hicieron las primeras pruebas con el receptor de infrarrojo se notó que si se emitía utilizando como señal de excitación para el emisor trenes de pulsos de 22 pulsos de largo y un tiempo muerto de igual largo se lograba obtener valores de jitter muy pequeños, por debajo de lo necesario para cumplir con las especificaciones. Pero si se comienzan a utilizar trenes más largos, de 30 pulsos o más, el jitter se agranda notoriamente. Dado esto se efectuaron multiples pruebas para lograr encontrar una señal que minimizara el jitter.

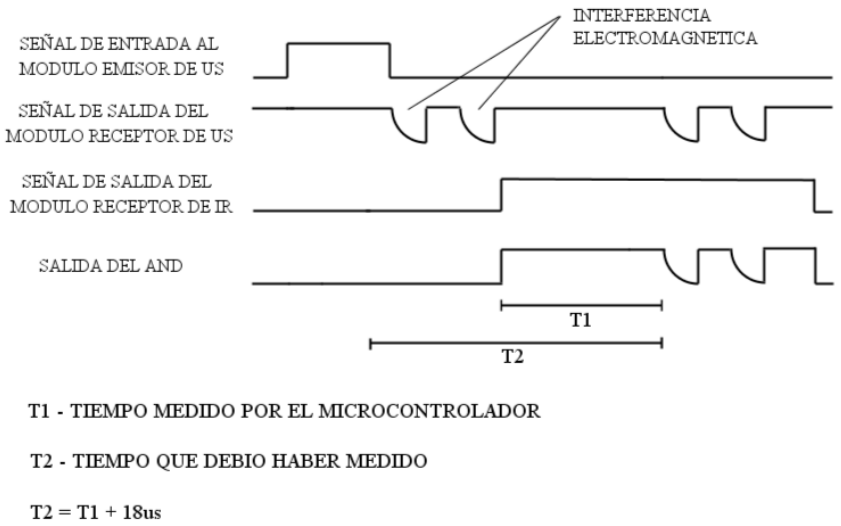

Figura 2.51: Retraso de la señal de infrarrojo para eliminar efecto de la interferencia electromagnética

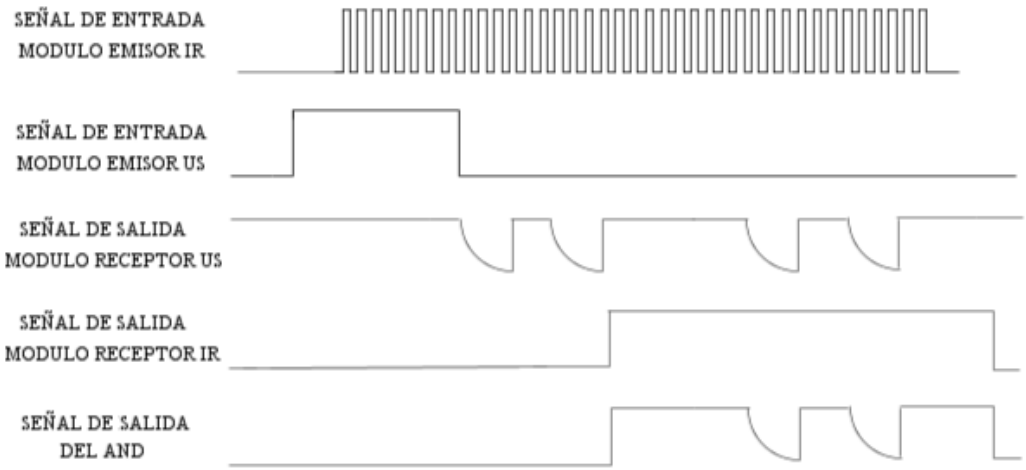

Figura 2.52: Señales de infrarrojo y ultrasonido en emisión y recepción

En las pruebas se intentó enviar periódicamente un pulso para mantener cierta referencia en el receptor pero no se notaban mejoras. Además se probó emitir trenes de pulsos de pocos pulsos entre muestra y muestra, o enviar algunos trenes de pulsos antes del tren de pulsos principal. La señal que logró minimizar el jitter consistía en emitir antes del tren de pulsos prin-

cipal 2 trenes de pulsos previos de 30 pulsos de 14μs cada uno con un tiempo muerto entre ellos de 825μs. Esta señal se muestra en la figura 2.53 junto con la señal de entrada al circuito emisor de ultrasonido, estas son las señales definitivas escogidas para la emisión.

Figura 2.53: Señales de entrada al módulo emisor de infrarrojo y ultrasonido escogidas

Capítulo 3

Software

3.1. Comunicación USB

En esta sección se brinda una breve descripción sobre el funcionamiento de la interfaz USB, necesario para comprender los conceptos que se manejarán más adelante.

3.1.1. Generalidades

EL USB (Universal Serial Bus) es una arquitectura que le brinda a un PC la posibilidad de conectar una variedad de dispositivos. El USB está pensado para intercambio de datos entre un dispositivo que se comporte como maestro (*host*) y dispositivos conectados a él (*esclavos*). Los sistemas USB tienen un solo host, el cual puede estar conectado a un amplio rango de dispositivos simultáneamente, los que deben compartir el ancho de banda. Los sistemas USB están formados por tres partes: USB host, USB device e interconexión USB.

3.1.2. Topología

El direccionamiento de dispositivos esclavos se hace mediante un identificador de 7 bits, por lo que se puede direccionar hasta 127 dispositivos por hub. Dado que el ruteo de datos se hace por broadcast, todos los esclavos dentro de una jerarquía reciben los datos, pero sólo los procesa el esclavo al que va dirigido dado que es el único con ese identificador.

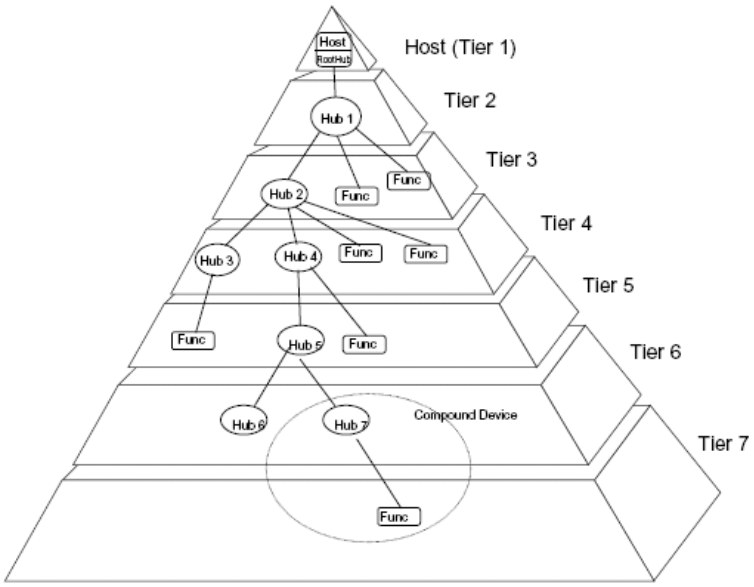

Figura 3.1: Topología USB. Fuente: Universal Serial Bus Specification [59]

3.1.3. Interfaz física

La señal de transferencia USB y alimentación son transportados por cuatro cables trenzados. La señal de transferencia es transportada en dos de éstos. En los otros dos cables se transportan las señales de VBUS y GRND para brindar alimentación al dispositivo conectado, brindando 5V entre estos dos cables y un máximo de 500mA. Los dispositivos que utilizan la alimentación que les brinda el host USB son llamados *bus-powered*, en cambio los que obtienen su alimentación de una fuente externa son llamados *self-powered*.

Hay tres velocidades de transmisión:

- High-speed: a una tasa de 480Mb/s
- Full-speed: a una tasa de 12Mb/s
- Low-speed: a una tasa de 1.5Mbp/s

3.1.4. Protocolo del bus

Cuando el host debe transmitir algo a un dispositivo, comienza enviando un *token packet* describiendo el tipo de transacción, dirección de la transacción, dirección del dispositivo y número de *endpoint* al que va dirigido. El

esclavo que tenga esa dirección es el único que va a procesar ese mensaje. Luego de esto la fuente (ya sea el host o device) enviará los paquetes de datos o un mensaje que indique que ya no tiene más datos para transmitir. El dispositivo responderá con un mensaje indicando que la transacción tuvo éxito.

Cuando se da la conexión, el host se encarga de habilitar el puerto y asignarle una dirección única al dispositivo. La acción de identificar y asignar una única dirección al dispositivo es llamada enumeración. Cuando el dispositivo es desconectado el hub deshabilita el puerto y manda una indicación al host de que el dispositivo fue removido.

3.1.5. Flujo de datos

El USB soporta intercambio de datos y de información de control en comunicaciones uni y bi-direccionales. Las transacciones toman lugar entre el software del host y un endpoint particular de un dispositivo. El endpoint es una porción del device que es el terminal de la comunicación entre el host y el device. Al canal lógico entre el software del host y el endpoint del device se lo denomina pipe. En la figura 3.2 se puede ver la lógica descrita.

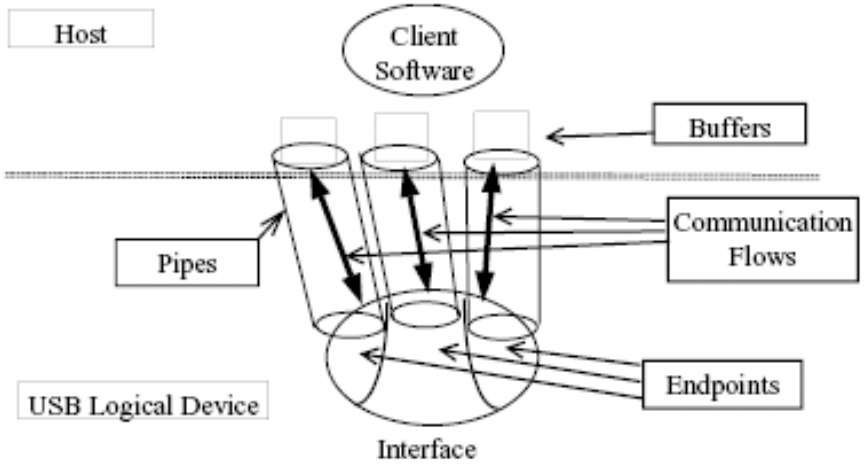

Figura 3.2: Flujo de datos USB. Fuente: Universal Serial Bus Specification [59]

Los endpoint pueden ser de control o de datos. El endpoint de control

es el endpoint 0 y es utilizado para transmitir información de configuración y estado del dispositivo, éstos endpoints son bi-direccionales y debe haber obligatoriamente al menos uno por dispositivo. Los endpoints de datos son uni-direccionales, y se clasifican según la dirección en la que transfieren los datos. Si los datos se transmiten en el sentido host-device se dice que es un endpoint de tipo OUT, y si se transmite en el sentido device-host son endpoints del tipo IN. Varios endpoints se agrupan para formar un dispositivo lógico denominado interfaz. En todo dispositivo físico habrá por lo menos una interfaz y cada interfaz tendrá por lo menos un endpoint de control. Un caso en el que hay dos dispositivos lógicos en un dispositivo físico sería por ejemplo un teclado con mouse incorporado.

Los *pipe* se clasifican en *stream* y *message*. Se dice que son *stream pipes* cuando los datos que circulan por él no tienen una estructura USB definida. Se denominan *message pipes* cuando tienen una estructura USB definida.

La arquitectura USB comprende cuatro tipo de transferencias básicas:

- Control transfers: son las utilizadas para configurar el dispositivo en el momento en el que este es conectado y para control de los otros canales lógicos que implemente el dispositivo.
- Bulk data transfers: son utilizados para transmitir cantidades relativamente grandes de datos en un solo envío. Se utilizan para acceder a un archivo en un disco o enviar un archivo a una impresora.
- Interrupt data transfers: son utilizados cuando hay envíos de información periódicos de baja tasa de transferencia, en dispositivos que respondan a la perceptibilidad humana, por ejemplo teclado o mouse.
- Isochronous data transfers: se utilizan cuando es necesario garantizar el tiempo de envío, como contrapartida no cuentan con control de errores. Son utilizados cuando hay transferencia de información multimedia en tiempo real, por ejemplo, una cámara web.

3.1.6. Estados de un device USB

Un dispositivo USB tiene una cantidad finita de estados posibles, a continuación se describen los mismos.

Attached: es el primer estado en el que esta luego de conectarse.

Powered: se encuentra en este estado cuando ya está alimentado, ya sea desde VBUS o una fuente externa.

Default: se encuentra en este estado luego de recibir un reset USB.

Addressed: luego de que el host USB le ha asignado una dirección.

Configured: estará en este estado luego de que haya respondido a los request de configuración del host.

Suspended: es un estado utilizado para ahorro de energía y debe salir de éste cuando se detecta actividad en el bus USB.

3.1.7. Descriptores

Todos los dispositivos USB tienen tablas de descriptores asociadas a él. Los descriptores son tablas que contienen información sobra las características del dispositivo. Los descriptores son jerárquicos, y los de mayor nivel informan al host sobre la existencia de los de menor nivel. Los descriptores no pueden ser modificados por el host, y el device se los transmite al host por el endpoint de control a través de control transfers.

Los descriptores comunes a todos los tipos de dispositivos son los siguientes:

- Device descriptor: describe información general sobre el dispositivo, tal como identificador de vendedor, identificador de producto, y las distintas configuraciones del dispositivo.
- Configuration descriptor: describe información sobre una configuración particular de un dispositivo, por lo que puede haber más de un configuration descriptor por dispositivo, tantos como configuraciones. Estos descriptores contienen información sobre la cantidad de interfaces de esa configuración, si el dispositivo funcionará en la modalidad self-powered o bus-powered y define el máximo valor de corriente que exigirá del puerto USB en caso de trabajar en ésta última modalidad.
- Interface descriptor: describe una interfaz específica de un dispositivo. La información que contiene define la cantidad de endpoints que tendrá esa interfaz (sin considerar el endpoint de control), a qué clase, subclase y protocolo pertenece esa interfaz. Las clases, subclases y

protocolos clasifican las distintas interfaces. Hay algunas clases, subclases y protocolos que ya están estandarizados, pero también pueden ser específicas del vendedor.

- Endpoint descriptor: contiene información sobre la dirección del endpoint, el tipo de trasferencia que implementa, el tamaño máximo de los paquetes.

Hay además descriptores opcionales, estos son los string descriptors que contienen información para que el host muestre al usuario.

3.1.8. Modelo de capas USB

La arquitectura USB implementa un modelo de tres capas, las dos superiores a nivel de software y la de más bajo nivel en hardware. La capa más baja es denominada bus interface layer. La capa de nivel medio es llamada device layer, los terminales son por el lado del dispositivo los endpoint y por el lado del host el client software, la información entre uno y otro circula a través de transacciones. La capa de más alto nivel es llamada function layer y está constituida por las interfaces. En la figura 3.3 se muestra la interacción entre las distintas capas.

3.1.9. Características a cumplir por el microcontrolador

Para implementar el USB device se decidió utilizar un microcontrolador que tuviera un módulo USB incorporado, y que además tuviera las características necesarias para poder implementar el procesamiento de señales necesario. Las características necesarias fueron discutidas anteriormente y se listan a continuación:

- Tener dos contadores que puedan contar a una velocidad de por lo menos 50MHz simultánea e independientemente.
- En caso de trabajar a menos de 65MHz (y más que 50MHz) deben ser de al menos 16 bits y en caso de trabajar a mayor velocidad, deben ser más grandes.
- Estos deben poder iniciar el conteo simultáneamente al arribo de una señal externa (Ir) y registrar el valor del contador separadamente al arribo de otras (ultrasonido).

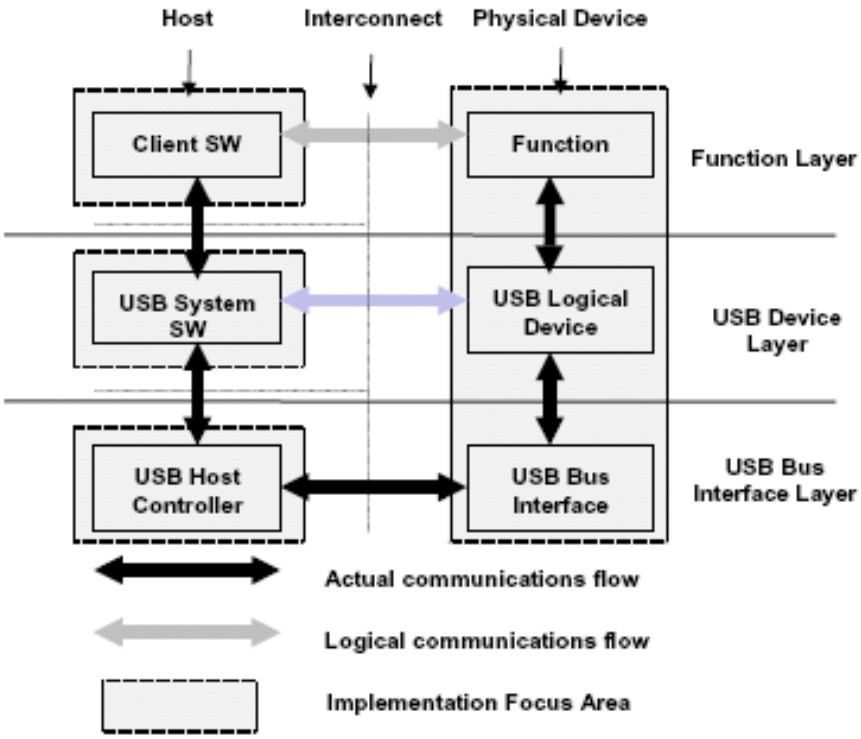

Figura 3.3: Modelo de capas USB. Fuente: Universal Serial Bus Specification [59]

A estas características se le debe sumar que debe tener implementada una interfaz USB que permita el uso del chip como device, y que esta interfaz USB debe poder funcionar en la modalidad bus-powered.

3.1.10. Prestaciones del microcontrolador elegido

El microcontrolador elegido fue el PIC32MX512L. Éste tiene un procesador que es capaz de funcionar a 80MHz, 2 timers de 32 bits capaces de funcionar a esta misma frecuencia y un módulo USB que da facilidades para la implementación del client USB, permitiendo el funcionamiento del dispositivo USB en la modalidad bus-powered. Las prestaciones que brinda el módulo USB son aún mayores, permitiendo el uso del dispositivo como un full-speed o low-speed host, full-speed device u On-The-Go, que le da la posibilidad de funcionar como host y device simultáneamente. Se utilizará sólo la funcionalidad como full-speed device. Permite la implementación de varias

configuraciones e interfaces y un máximo de 32 endpoints, además del endpoint de control.

3.2. Software PIC12

El PIC12 es el microcontrolador encargado de generar las señales de entrada a los circuitos de emisión de infrarrojo y ultrasonido, controlando simultáneamente el estado de los pulsadores con los que se implementaron el click izquierdo y derecho. Este microcontrolador tiene un reloj interno de 4MHz, y un tiempo de ciclo de 1μs, suficiente para generar las señales requeridas. El software fue elaborado usando el entorno de desarrollo MPLAB IDE, que es el sugerido por Microchip Inc. y de su propiedad. El código se escribió en assembler, lo que permite un buen control de las señales a generar.

3.2.1. Descripción de señales a generar

Como se vio anteriormente, las señales a generar por el PIC12 son las que se muestran en la figura 3.4, estas señales deben repetirse cada 1/75s como se ve en la figura 3.5.

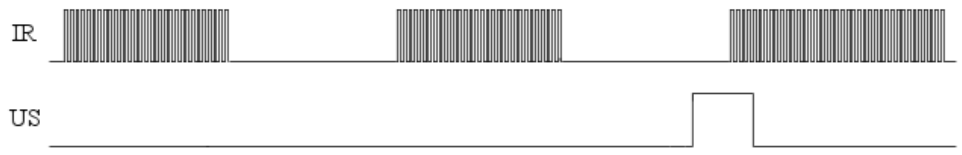

Figura 3.4: Señal de infrarrojo y ultrasonido generadas por el PIC12

Figura 3.5: Muestras consecutivas de las señal de infrarrojo y ultrasonido generadas por el PIC12

La señal de entrada al circuito emisor de infrarrojo está formada por trenes de pulsos que se repiten cada 1/75s, dado que el Lapix trabaja a 75 muestras por segundo. En todos los casos los pulsos tienen 14μs y el tiempo muerto que les sigue tiene el mismo largo. Los primeros dos trenes de pulsos son idénticos, y están formados por 30 pulsos. Estos 2 trenes de pulsos se utilizan para que el AGC del receptor de infrarrojo quede siempre en el mismo valor de comparación al pasar las muestras, lo que ayuda a disminuir el jitter en infrarrojo. Estos 2 trenes de pulsos los llamamos pulsos previos. El tiempo entre el primer tren de pulsos y el segundo es de 825μs, al igual que el tiempo entre el segundo y el tercer tren de pulsos. El largo del tercer tren de pulsos depende del estado de los pulsadores. Si ninguno de los clicks está presionado el tercer tren de pulsos tendrá 39 pulsos, si el click izquierdo está presionado se envían 53 pulsos, si es el click derecho es el que está presionado se envían 68 pulsos, y en caso de estar ambos clicks presionados se envían 82 pulsos. En la imagen 3.6 se muestra el largo del tercer tren de pulsos en cada uno de los casos.

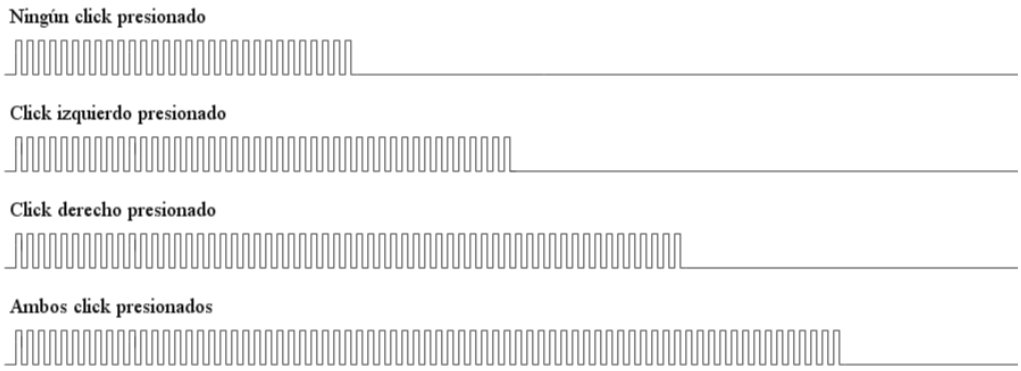

Figura 3.6: Largo del tercer tren de pulsos

La señal de entrada al circuito de ultrasonido que debe generar el PIC12 es más sencilla que la de infrarrojo. Se trata sólo de un pulso de 308μs de largo, que se repite cada 1/75s al igual que la señal de entrada al circuito de emisión de infrarrojo. Lo que debe cumplir la señal de entrada al circuito de emisión de ultrasonido es que debe tener el flanco de subida 106μs antes que el tercer tren de pulsos de infrarrojo, como se muestra en la figura 3.4.

3.2.2. Implementación

Para el sensado del estado de los pulsadores se conectó el pulsador correspondiente al click izquierdo al pin GP2 del PIC12 y el correspondiente al click derecho al pin GP3. Estos dos son pines de propósito general, los que se fijaron como entradas. Se considera que el click está presionado cuando hay VDD en estos pines. Los pines GP0 y GP1 del PIC12, también son de propósito general y se fijaron como salidas. El pin GP0 se utiliza para generar la señal de entrada al circuito de infrarrojo. El pin GP1 se utiliza para generar la señal de entrada al circuito de ultrasonido.

El software está dividido en cinco bloques. En el primer bloque se declaran las constantes y variables. Además se fijan los registros para el timer a utilizar y para determinar qué pines se utilizarán como entradas digitales y cuáles como salidas digitales.

El segundo bloque se encarga de generar el primer tren de pulsos y el tiempo muerto que le sigue. Para generar el tren de pulsos se entra en un loop que fija o borra cada 14μs (que son 14 ciclos de reloj del PIC12) el registro que maneja el pin GP0. Además se dispara un contador que aumenta en 1 cada vez que se genera un pulso y cuando éste llega a 30 se sale del loop. Luego se entra a otro bucle para generar el tiempo muerto, cada vuelta a este loop es de 14μs y en este caso se sale del loop cuando el contador llega a 57.

El tercer bloque es similar al anterior. El tren de pulsos se genera de igual forma, pero hay un cambio en la implementación del tiempo muerto, dado que se debe comenzar a generar el pulso de ultrasonido. El control del largo del tiempo muerto se implementa de igual forma que en el tiempo muerto anterior, mediante un loop y un contador, saliendo del loop cuando el contador llega a 57 al igual que en el bloque anterior. Para controlar cuando se produce el flanco de subida en la señal de ultrasonido se utiliza el mismo contador, pero cuando este llega a 51 se fija el registro que maneja el pin GP1, generando así el flanco de subida en la señal de ultrasonido, la que se mantendrá hasta que se borre el registro que maneja el pin GP1. De esta forma se logra que el pulso de ultrasonido comience 106μs antes que el tercer tren de pulsos de la señal de infrarrojo.

El cuarto bloque es el encargado de generar el tercer tren de pulsos. Se implementa de igual forma que los otros dos trenes de pulsos, pero varía la cantidad de pulsos. Además debe generar el flanco de bajada en la señal de ultrasonido lo que se implementó borrando el bit 1 del registro GPIO. De esta

forma el largo total del pulso de ultrasonido queda en 308μs. Para lograr que el largo del tercer pulso cambie según qué click está presionado, se creó una variable llamada CANTIDAD_PULSOS que se inicializa en 39 cada 1/75s. Si el click izquierdo está presionado (lo que se sabe sensando el bit 2 del registro GPIO) se suma 14 a la variable CANTIDAD_PULSOS, y si está el click derecho presionado (lo que se sabe sensando el bit 3 del registro GPIO) se suma 29 a la variable CANTIDAD_PULSOS. Se sale del loop que genera el tren de pulsos cuando el contador llega al valor de CANTIDAD_PULSOS.

El quinto bloque, solo se utiliza para esperar que pasen los 1/75s. Se implementa mediante un loop en el cual se sensa si el valor del registro TMR0_OPT vale 51, momento en el que habrán pasado 1/75s desde que se comenzó a ejecutar el código. Cuando esta condición se cumpla se vuelve al principio del programa.

3.3. Software PIC32

El software para el PIC32 se desarrolló en lenguaje de programación C, utilizando el entorno de desarrollo MPLAB IDE. Para entender de mejor manera la implementación del software se lo divide en 2 módulos, el de procesamiento de las señales recibidas y el client USB, estos dos módulos funcionan en simultáneo. El primer módulo se encarga de medir el tiempo transcurrido desde que llega la señal de infrarrojo hasta que llega la de ultrasonido, decodificar qué pulsador está presionado y cargar esta información en un registro. El client USB se encarga de manejar la interfaz USB y enviar por esta interfaz el mensaje generado por el módulo de procesamiento de señales. A continuación se describirán con mayor detalle.

3.3.1. Cliente USB

Para la comunicación se implementa un device USB con una sola configuración, una interfaz y con 2 endpoints. El endpoint 0 es utilizado como endpoint de control y es el utilizado en el proceso de enumeración. El otro, endpoint 1, es del tipo IN, y permite el envío de información desde el dispositivo hacia el PC, por lo que periódicamente se enviará la distancia desde la punta del lápiz hasta los receptores de ultrasonido. Dado que el envío de información es periódico y la tasa de transferencia es baja se utilizaron transferencias del tipo interrupt data. Éstas son las utilizadas por dispositivos como el mouse, de características similares al Lapix en cuanto a la comunicación USB.

El stack del firmware para el PIC32 se puede dividir en cuatro capas, como se muestra en la figura 3.7.

La capa de aplicación consiste en el firmware de alto nivel. Se puede comunicar tanto con el USB firmware stack como con otro software en el sistema.

Function driver provee control de las funciones específicas a la aplicación. Pueden existir varias en un dispositivo, pero el Lapix implementará solo una.

Device layer se encarga principalmente de manejar los protocolos USB especificados en Chapter 9 de Universal Serial Bus Specification [59]. También provee de acceso al USB a cualquiera de las funciones que puedan estar implementadas sobre él.

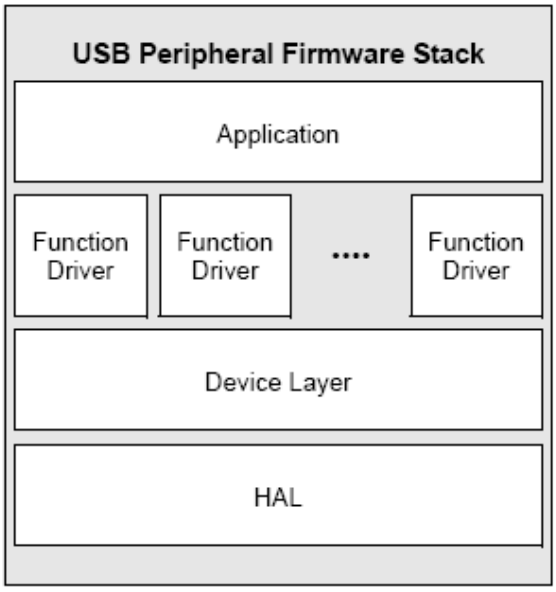

Figura 3.7: Stack USB para el PIC32. Fuente: USB Device Stack for PIC32 Programmer's Guide [60]

La *Hardware Abstraction Layer* HAL se encarga del control del hardware.

El módulo USB se comporta de forma tal que el software debe ser desarrollado para las 3 capas de mayor nivel, y para la capa de menor nivel solo hay que configurar los registros de acuerdo a lo necesitado. Esto facilita considerablemente la implementación de un dispositivo USB.

Para el desarrollo del USB client se partió de un código ejemplo creado por Microchip Inc., disponible en su sitio web para descargar [61]. Este ejemplo implementa un Mouse USB, cuya configuración es similar al Lapix en cuanto a la comunicación USB. El código del que se partió está protegido bajo las leyes de los derechos de autor de Microchip. Microchip permite el uso de material bajo sus derechos de autor sin necesidad de pedirle permiso siempre y cuando sea con fines educacionales, ya sea reportes académicos, tesis, libros de texto, presentaciones o artículos que sean educacionales por naturaleza. En caso de que se desee utilizarlo con fines comerciales se debe pedir permiso a Microchip por escrito. Para más información sobre el uso de material bajo los derechos de autor de Microchip visitar su sitio web al respecto [62].

Para que el código sea fácilmente entendible se divide en varios archivos, los que implementan distintas funciones y contienen información sobre distintos descriptores. A continuación se da un repaso sobre las funciones y declaraciones implementadas en cada archivo.

main.c

Este archivo es el que contiene el loop del procesamiento de señales, que se verá en detalle más adelante. En cuanto a la comunicación USB sólo se encarga de llamar a la función USBInitialize() que inicializa el módulo USB y luego que entra en el loop se llama periódicamente a la función USBTask() la que se encarga del manejo del estado del bus USB. Si esta función no es llamada regularmente el funcionamiento del USB deja de ser correcto porque no se actualiza el estado del bus.

usb_config.c

Archivo utilizado para la configuración del stack del firmware. En el se definen la cantidad de endpoints a utilizar, el tamaño de paquete máximo del endpoint de control. En el se definen además algunos descriptores USB generales, estos son device descriptor, configuration descriptor, interface descriptor, endpoint descriptor y string descriptors.

usb_device.c

En este archivo se define la interfaz para la capa device. Se hace el manejo de los pedidos de descriptores por parte del host, el manejo de los pedidos de configuración del host, de los pedidos de interfaz del host y de los pedidos de información de estado. Además prepara los periféricos del dispositivo luego de que éste se ha conectado al host, se encarga del manejo de las transferencias por el endpoint de control, se encarga del manejo de un reset de la interfaz USB y del manejo de los eventos de bus.

usb_hal.c

En este archivo se define la interfaz para la HAL. Las funciones que se implementan se encargan de setear los descriptores para los buffers, proveer información del número de endpoint, dirección y otra información para identificar paquetes de información a enviar. También se encarga de setear los tamaños de los buffers en los que se implementarán los endpoints, se encarga

además de dar servicios a un endpoint cuando éste lo necesite, implementa funciones para notificar que hay una interrupción por un reset, suspend o resume USB. También implementa las funciones para setear la dirección del sistema en el USB y preparar el sistema para una transferencia.

class.c

En este archivo se definen las estructuras y valores del descriptor de clase. Además se definen las funciones encargadas de inicializar el valor del descriptor de clase, la encargada de manejar los eventos de clase, la función encargada de realizar las transferencias de datos, la encargada de manejar los pedidos de descriptores de clase por parte del host, la función encargada de recibir los reportes de clase del host; además de funciones auxiliares utilizadas por las funciones antes mencionadas.

usb.h

Es el encabezado con las funciones para inicializar el stack USB y la función USBTask que ejecuta las tareas para mantener en funcionamiento el bus.

usb_common.h

Es el encabezado que contiene definiciones para el stack USB tales como el valor para el campo endpoint_number de las transferencias, valor en campo de las transferencias que corresponde a los tipos de eventos de bus, valor en los campos de transferencia de los errores de transmisión, entre otros.

usb_config.h

En este encabezado se establece el valor del vendorID (para el lápiz se eligió 0xAAAA y el valor del productID se escogió 0xBBBB, no se pudo verificar si estos valores no están siendo usados por otro dispositivo). Además se define que el modulo va a ser utilizado como dispositivo, se define la función que se encargará del manejo de los eventos de mouse y la cantidad de endpoints a utilizar, cantidad de pipes, el tamaño máximo de paquetes en el endpoint de control. También se define la función que se encarga de entregar los descriptores a la capa device y brindar la configuración del endpoint. Por último se define que el dispositivo se utilizará en el modo self powered y tendrá soporte para remote wake-up.

usb_ch9.h

En este archivo se definen las estructuras de los descriptores USB. Se definen además el valor de distintos atributos, como el que se utiliza para indicar la dirección del endpoint, que tipo de transferencia va a manejar el endpoint, y el máximo tamaño de paquete del endpoint.

usb_device.h

Es este archivo se define la estructura para configuración de endpoints, se define además la función que identifica que descriptor esta siendo solicitado por el host para poder dar respuesta a su pedido. Por último se definen las funciones que inicializan el function driver, la función que manda una señal al host de resume, la que indica el último error ocurrido, y la función que se encarga del envío de datos hacia el host.

usb_device_local.h

En el se define un identificador para saber en qué estado está el endpoint de datos, la función que indica el estado en que se encuentran las transferencias, y se definen las banderas para indicar si el dispositivo funcionará self o bus-powered, si está en estado suspendido o conectado.

usb_hal.h

En este archivo se definen las funciones encargadas de enviar una señal de resume al host, la que fija el valor de la dirección USB del dispositivo, la que se encarga de habilitar o deshabilitar el pull=up o pull=down de las resistencias internas del modulo, la función que determina si hay en ese momento una sesión USB válida o no, la función que brinda un mapa de bits de los errores USB más recientes producidos, la función que verifica si hubo eventos de bus, las funciones que paran un endpoint o pipe, la función que prepara el módulo para enviar datos por él, la que configura un endpoint, la función para inicializar o reinicializar el HAL y la funciona para deshabilitar el módulo USB.

usb_hal_local.h

En este encabezado se definen constantes, máscaras y estructuras que se utilizan a la hora de configurar un endpoint, pipe, buffer o configurar el funcionamiento de la HAL. Además se definen las funciones para obtener la configuración de la HAL, borrar su estado, y obtener o borrar errores produ-

cidos en la HAL.

Para el funcionamiento del módulo USB lo que se hace es llamar al principio del programa a la función USBInitialize() que se encarga de inicializar el stack de software USB, incluido el HAL. Luego se debe llamar regularmente a la función USBTask() que se encarga del manejo de los eventos de bus que se puedan producir. Esta función también es llamada por interrupciones para el manejo de los eventos de bus y para responder a los pedidos de descriptores del host. Para la transferencia de datos al host se utiliza la función SendReport().

Las transferencias de datos que se implementaron envían 7 bytes. Tres de estos bytes se utilizan para enviar la diferencia temporal entre la señal de infrarrojo y la señal de ultrasonido del receptor de ultrasonido 1. Otros tres bytes se utilizan para enviar la diferencia temporal entre la señal de infrarrojo y la señal de ultrasonido del receptor de ultrasonido 2. El último byte se utiliza para enviar el estado del click izquierdo y derecho. En la figura 3.8 se muestra la estructura de la transferencia. Las diferencias temporales entre la señal de infrarrojo y las señales de ultrasonido son medidas en periodos de reloj de 80MHz, que es la máxima frecuencia a la que pueden funcionar los timers internos del PIC32. Se llama $N1$ y $N2$ a la diferencia temporal medida en períodos de reloj de 80MHz entre que llega la señal de infrarrojo y llegan las señales de ultrasonido respectivamente.

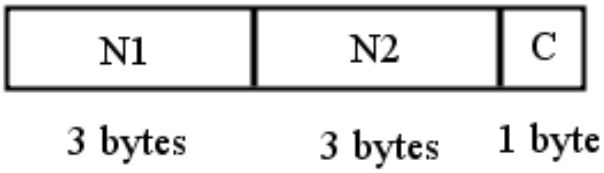

Figura 3.8: Estructura de la transferencia a enviar

3.3.2. Procesamiento de señales recibidas

Recordemos que el módulo receptor de infrarrojo genera un pulso cuando se está recibiendo infrarrojo. La señal de salida de este módulo se conectó al pin RA7 del PIC32, se hace para conocer por software si se está recibiendo la señal de infrarrojo y medir su largo para identificar si hay algún click presionado. Los módulos receptores de ultrasonido 1 y 2 generan un tren de pulsos cada uno cuando están recibiendo la señal de ultrasonido. Las señales

de salida de los módulos de ultrasonido 1 y 2 se conectaron a los pines INT1 e INT4 del PIC32 y se utilizan para conocer por software si se han recibido ambas señales de ultrasonido mientras se recibe la señal de infrarrojo.

Se utilizan dos timers de 32 bits para medir el tiempo transcurrido desde que llega la señal de infrarrojo hasta que llegan las dos señales de ultrasonido. Los timers de 32 bits se setearon para trabajar en el modo *gated*. En esta modalidad los timers comienzan a contar cuando llega un flanco de subida al pin T2CK (en el caso del timer23) y al pin T4CK (en el caso del timer 45), y se detienen cuando llega un flanco de bajada a dichos pines. A uno de esos pines se conectó la salida de un AND lógico entre la señal de salida del receptor de infrarrojo y la señal de salida del receptor de ultrasonido 1. Al otro pin se conectó la salida de un AND lógico entre la señal de salida del receptor de infrarrojo y la señal de salida del receptor de ultrasonido 2. De esta forma se consigue que los timers cuenten exactamente el tiempo transcurrido desde que llega la señal de infrarrojo hasta que llega cada una de las señales de ultrasonido, dado que a la salida de las compuertas AND se genera una señal que se pone en 1 cuando llega la señal de infrarrojo y vuelve a 0 cuando llega la señal de ultrasonido.

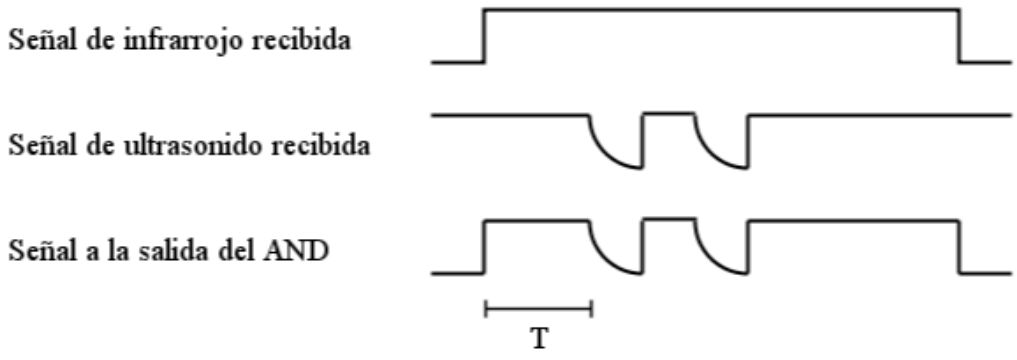

Figura 3.9: Señales recibidas por el PIC32

Para medir el largo de la señal de infrarrojo recibida se utiliza un timer de 16 bits. Esto sirve para conocer si se está recibiendo la señal que indica que no hay ningún click presionado, o que está sólo el click izquierdo presionado,

o el click derecho, o ambos clicks.

Al finalizar el procesamiento de la señal recibida se enviará a la XO un reporte de 7 bytes por la interfaz USB. Los primeros 3 bytes contienen el valor del contador del timer23, lo que se puede traducir como el tiempo transcurrido desde que llega la señal de infrarrojo hasta que llega la señal de ultrasonido al receptor de ultrasonido 1. En los siguientes 3 bytes se envía el valor del contador del timer45, lo que se puede traducir como el tiempo transcurrido desde que llega la señal de infrarrojo hasta que llega la señal de ultrasonido al receptor de ultrasonido 2. En el último byte se envía el estado de los clicks. Si no hay ningún click presionado se envía un 0 en éste byte, de estar presionado el click izquierdo se envía 1, si es el click derecho el que está presionado se envía 2 y de estar presionados ambos clicks se envía 3.

El algoritmo comienza inicializando los timers y luego queda en espera de la señal de infrarrojo.

Una vez que llega la señal de infrarrojo, se ejecuta la función hallar_largo_IR() que devuelve un entero que indica cuál de los clicks está presionado o devuelve 5 en caso de que la señal recibida sea considerada una interferencia. Se considera como interferencia cualquier señal de infrarrojo que no tenga un largo mayor a 950μs. Vale recordar que la señal de infrarrojo tiene distinto largo dependiendo del estado de los clicks. La función hallar_largo_IR() devuelve 0 si el largo de la señal de infrarrojo corresponde con no hay ningún click presionado, devuelve 1 en caso de que el largo de la señal corresponda a está el click izquierdo presionado, devuelve 2 en caso que la señal se corresponda con la codificación de click derecho presionado y devuelve 3 en caso de que el largo de la señal indique que están ambos clicks presionados.

El entero que devuelve la función hallar_largo_IR() se carga en una variable llamada largo_IR. Si largo_IR vale 5 quiere decir que la señal recibida era solo una interferencia y se vuelve al principio del programa, donde se vuelven a inicializar los timers. Si largo_IR era distinto de 5 se continúa con el procesamiento de las señales.

El algoritmo continúa verificando que ésta no sea la primera vez que se recibe la señal de infrarrojo, de ser así, se vuelve al principio del código, ya que por ser la primera vez que se realiza el procesamiento de la señal, los timers podrían haber comenzado a contar cuando la señal de infrarrojo ya se estaba recibiendo.

Luego de esto se pasa a verificar que las señales de ultrasonido hayan llegado. Para esto es necesario testear si las banderas que indican la llegada de las señales de ultrasonido valen 1, de valer 0 quiere decir que alguna o ambas señales de ultrasonido no llegaron.
Si alguna o ambas señales de ultrasonido no llegaron se vuelve al principio del programa. Si ambas señales de ultrasonido llegaron entonces se carga el valor del contador del timer23 en los 3 primeros bytes del reporte, en los siguientes 3 bytes se carga el valor del contador del timer45 y en el último byte se carga el valor de la variable largo_IR (que sirve para identificar que click está presionado). Por último se envía el reporte hacia la XO mediante la comunicación USB y se vuelve al principio del código (donde se inicializaban los timers). En la figura 3.10 se muestra el diagrama de flujo del algoritmo antes descrito. Vale aclarar que en todos los casos para saber si han llegado las señales de infrarrojo y ultrasonido se hace polling.

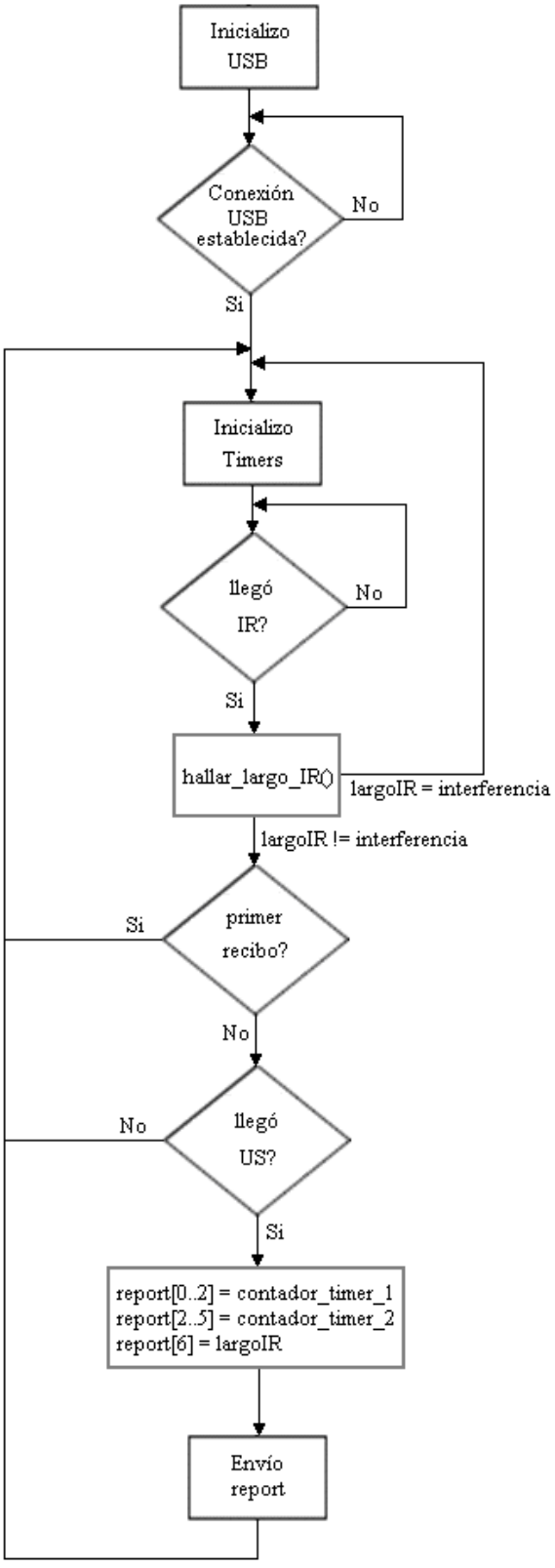

Figura 3.10: Diagrama de flujo del algoritmo

3.4. Driver

Introducción

En este capítulo se describe cómo se realizó la comunicación entre el hardware desarrollado y la PC poniendo énfasis en las librerías y funciones utilizadas. Se describe también la realización del software de calibración y de filtrado.

3.4.1. USB Host

Para lograr la correcta interacción entre el hardware desarrollado y la XO, o cualquier otra máquina, se debía desarrollar una aplicación que lograra comunicar las dos partes de manera satisfactoria. Como estaba definido en las especificaciones, esta comunicación sería via USB, por lo que la aplicación que corriera en el host (computadora con la cual se este usando Lapix) debería ser capaz de tomar información del puerto USB, interpretarla y desplegarla en pantalla.

El primer paso para lograr esto fue desarrollar el USB Host, es decir una aplicación capaz de reconocer un dispositivo que se conectara al USB y que lograra tomar la información que éste enviara. En primera instancia se evaluó y estudio durante mucho tiempo realizar un driver en Kernel de Linux. Sin embargo, se encontró una herramienta más práctica para realizar dicha tarea por lo que se desestimó esta opción.

Finalmente se optó por utilizar la libUSB para desarrollar la comunicación con el dispositivo.

La librería USB

La librería USB o libUSB[63], es una librería creada para realizar funciones básicas con el puerto USB. Oculta al usuario todos los procedimientos necesarios para lograr la comunicación con el USB, y tiene una vasta cantidad de funciones que dan la flexibilidad necesaria para realizar un driver para hardware USB. Conllevaba una gran complejidad intentar implementar todo esto a nivel de Kernel. La libUSB permite la programación directamente en C, haciendo las veces de nexo entre el Kernel y el usuario.

Funciones de la libUSB

En la página oficial de la libUSB[63] se puede tener acceso a la API de la misma. Basándonos en esta documentación y en ejemplos de implementación encontrados se escribe una aplicación que utiliza las siguientes funciones (se enumeran las más representativas):

- usb_init
- usb_find_busses
- usb_find_devices
- usb_claim_interface
- recv_usb

Las funciones anteriores, son básicas para lograr establecer una correcta comunicación, más adelante se explica el funcionamiento de cada una de ellas.

3.4.2. Lectura de datos

Utilizando las funciones anteriormente mencionadas, se realiza una aplicación en C que se encarga de reconocer el dispositivo y es capaz de recibir los datos que éste envía.

En la figura 3.11 se muestra el diagrama de funcionamiento del driver. En la misma se pueden apreciar las etapas necesarias para lograr establecer la comunicación y recibir datos. Primero se buscan los buses usb que haya en la máquina. Cada vez que se encuentra uno, se fija qué hay en el mismo. Para ello se consulta la información básica de identificación de usb, en este caso nos interesa el "Product_Id" y el "Vendor_Id" que son los parámetros que utilizaremos para identificar nuestro producto.

Los descriptores de cada bus se comparan con los de Lapix que son:

- Product_Id = 0xBBBB
- Vendor_Id = 0xAAAA

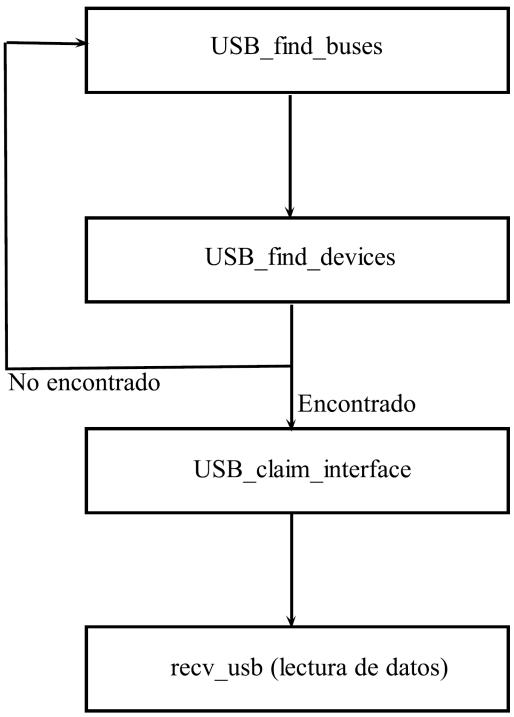

Figura 3.11: Esquema de funcionamiento del driver en la etapa inicial

Estos son fijados al momento de programar el PIC32 que se encuentra en el receptor. Si el descriptor de alguno de los dispositivos coincide con el de Lapix se puede proceder con el driver, pudiendo leer datos desde el mismo. De lo contrario, el proceso se mantiene en espera.

Los datos se leen respetando la estructura mediante la cual se envían. Se leen 7 bytes provenientes del usb, los primeros 3 son el valor del primer contador, los siguientes 3 el valor del segundo contador, el séptimo es el estado del click. Para ver más detalle sobre como se envían referirse a la sección 3.3.

3.4.3. Cambio de coordenadas

Como se mencionó anteriormente el PIC32 envía periódicamente un mensaje con el tiempo transcurrido desde que llega la señal de infrarrojo hasta que llega la señal de ultrasonido a cada uno de los receptores de ultrasonido. El cambio de coordenadas consiste en pasar estas dos distancias a un sistema de coordenadas rectangulares, logrando de esta forma ubicar la punta del lápiz en este sistema. Llamaremos $N1$ a la distancia temporal al receptor de

ultrasonido 1 y $N2$ a la distancia al receptor de ultrasonido 2.

El punto $(0, 0)$ del sistema de coordenadas rectangulares creado se considera en el punto medio entre ambos sensores de ultrasonido, además ambos sensores están ubicados sobre el eje x. La distancia entre los sensores de ultrasonido es de 8cm, considerando la velocidad del sonido como 343,42m/s, da un tiempo de 232.95μs, que son 18636 períodos de reloj de 80MHz. Se llamó N a la distancia entre los sensores de ultrasonido y el punto $(0, 0)$, este valor es la mitad del número de períodos mencionado anteriormente, por lo que N vale 9318. Teniendo esto en cuenta y considerando los receptores de ultrasonido como puntos, el sensor de ultrasonido 1 se encuentra en el punto $(N, 0)$ del sistema de coordenadas y el sensor de ultrasonido 2 se encuentra en el punto $(-N, 0)$.

Como $N1$ y $N2$ representan las distancias desde la punta a los sensores, lo que hay que hacer para pasarlo a coordenadas polares es la intersección entre dos circunferencias, una con centro en el sensor de ultrasonido 1 y radio $N1$ y la otra con centro en el sensor de ultrasonido 2 y radio $N2$. Vale destacar que al valor de $N1$ y $N2$ que envía el PIC32 se le debe sumar 1440, que se corresponde con un adelanto de 18μs que tiene la señal de ultrasonido. Si las circunferencias que se muestran en la figura 3.12 se cortan en dos puntos se toma el que tiene coordenada y positiva, si se cortan en un solo punto se toma ese punto, y si no se cortan en ningún punto se descarta esa muestra y se queda en espera de la siguiente. Se considera que la intersección entre ambas circunferencias no será infinitos puntos dado que para que esto ocurra ambos sensores deben estar ubicados en el mismo punto.

El pasaje de las distancias recibidas a coordenadas rectangulares se implementó dentro de la función leer_usb(). Esta función se encarga de leer los reportes que llegan vía USB por parte del PIC32 y pasarlos a un sistema de coordenadas rectangulares. La función retorna una variable llamada Punta, que contiene el valor x e y de la posición de la punta del lápiz. En el caso en que no haya puntos de intersección entre las circunferencias se carga el valor -1 en la coordenada y de la variable Punta, lo que se definió como valor de error.

3.4.4. Software de calibración

Luego del cambio del sistema de coordenadas, se obtiene la posición de la punta del emisor con respecto a los receptores de ultrasonido, utilizando

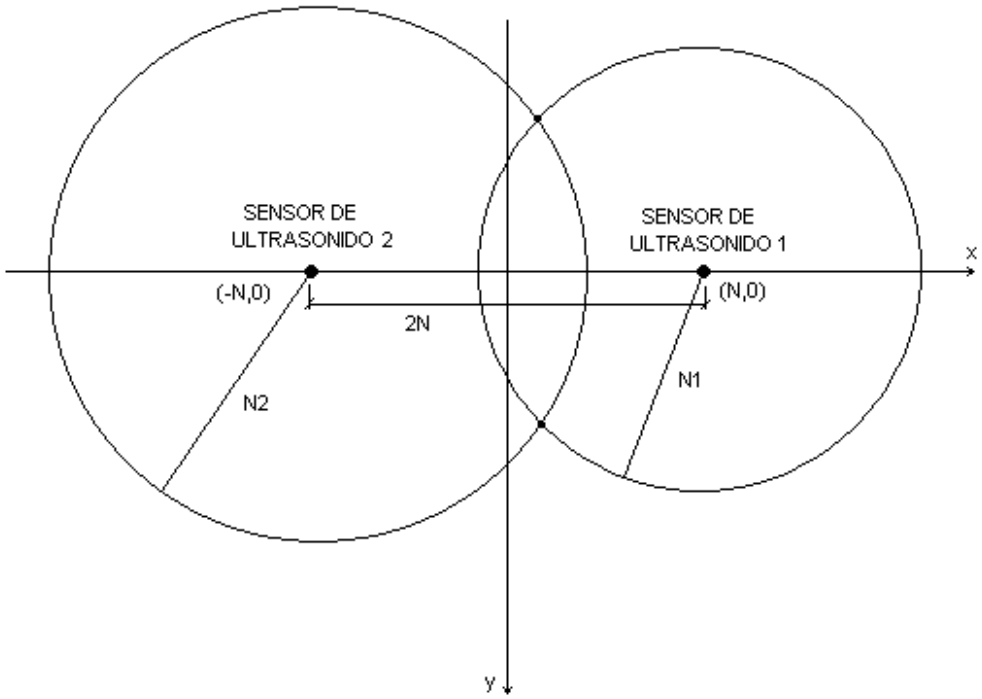

Figura 3.12: Cambio de sistema de coordenadas

como unidad un período de reloj de 80MHz, a éste se le llama sistema de coordenadas A. La pantalla del PC trabaja con un sistema de coordenadas rectangular donde la unidad es el pixel, al que se le llama sistema de coordenadas B. Para poder indicar la posición de la punta del lápiz mediante la posición del cursor en la pantalla es necesario hacer una correspondencia del sistema de coordenadas A en el sistema de coordenadas B. Esta correspondencia será la encargada de absorber el problema de que el eje que une a los receptores de ultrasonido quede girado y trasladado con respecto a la pantalla, además de hacer el cambio de unidades.

Para la correspondencia de un sistema de coordenadas en otro se optó por una transformación lineal.

Como ambos sistemas de coordenadas son rectangulares y en dos dimensiones, bastaría con conseguir una transformación que lleve 2 puntos de un sistema de coordenadas en 2 puntos del otro. Los puntos del sistema de coordenadas B son elegidos arbitrariamente, y los puntos del sistema de coordenadas A deben ser obtenidos de la calibración, proceso por el cual se consigue establecer qué punto del sistema A se corresponde en que punto del sistema de coordenadas B. Los puntos que se obtienen del sistema A tienen cierta incertidumbre, dado que son los medidos por el receptor, por lo cual se decidió trabajar con 4 puntos en lugar de 2. Esto permite que la correspondencia sea más ajustada. Con los 4 puntos de cada sistema de coordenadas se utiliza mínimos cuadrados para ver cuál es la transformación que mejor se ajusta.

Implementación

Antes de llamar al software que realizaba la calibración, se definen los puntos de la pantalla, o sea del sistema de coordenadas B. Para esto se llama a la función XGetWindowAttributes() y se obtienen el largo y ancho de la pantalla en pixeles. Los puntos del sistema de coordenadas B se definen de forma tal que estén alejados de cada borde de la pantalla 20 pixeles en el eje x y 20 pixeles en el eje y. La elección se hizo tratando de que estén lo más alejados posible uno del otro, pero que se los pueda encontrar en la pantalla rápidamente. A estos puntos se los denomina $XO1$, $XO2$, $XO3$ y $XO4$.

Luego de que estos puntos se han definido, se llama a la función software_de_calibracion(), la cual se encarga de desplegar uno por uno los 4 puntos en pantalla en sentido horario. Al final de esta función se obtienen los 4 puntos del sistema de coordenadas A. A estos 4 puntos se los denomina $L1$, $L2$, $L3$ y $L4$. El procedimiento para cada punto es el siguiente: se muestra el punto en pantalla, luego se espera a que el usuario presione con la punta del lápiz sobre el punto en la pantalla momento en el que se tomará cuál es la posición de la punta del lápiz. Este es el punto en el sistema de coordenadas B con el que se debe hacer corresponder el punto desplegado en pantalla. Luego de repetir este procedimiento para los cuatro puntos se termina la calibración. Se procede entonces a hallar la transformación M que lleve los puntos del sistema A en los puntos del sistema B como se muestra en la figura 3.14.

Antes de buscar la transformación que lleva un sistema de coordenadas en el otro, se realizan 2 traslaciones, una en cada uno de los sistemas de coordenadas llamadas $T1$ para la traslación en el sistema de coordenadas

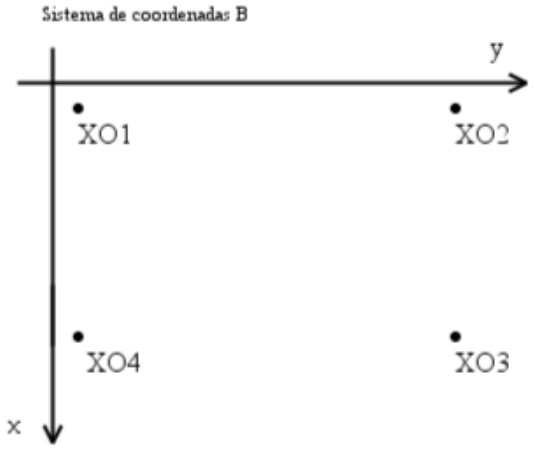

Figura 3.13: Puntos del sistema de coordenadas B utilizados para la calibración

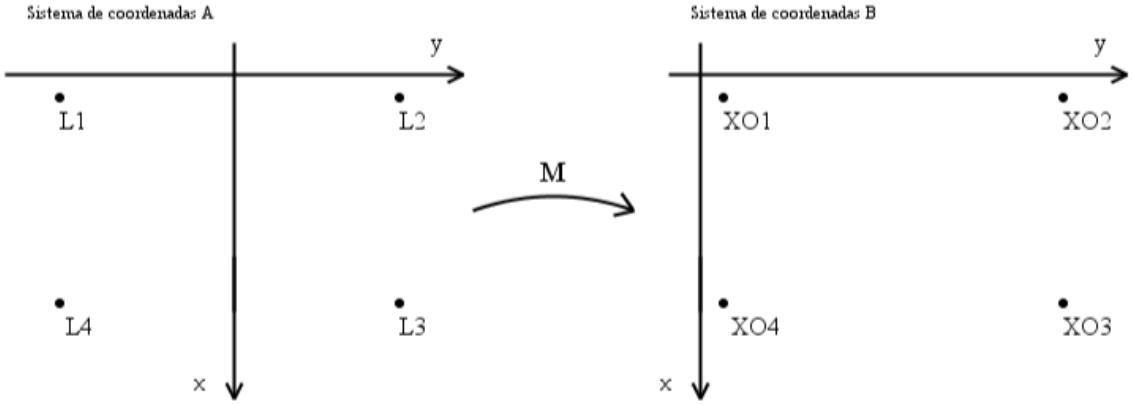

Figura 3.14: Transformación M

A y $T2$ para el sistema de coordenadas B. La traslaciones se realizan de forma tal que el punto $(0, 0)$ de los nuevos sistemas de coordenadas A' y B' queden en el punto medio de los cuatro puntos anteriormente hallados. Las traslaciones se realizan para lograr mayor resolución en el centro de la pantalla. Los puntos en estos nuevos sistemas de coordenadas pasan a llamarse $XO1'$, $XO2'$, $XO3'$, $XO4'$, $L1'$, $L2'$, $L3'$ y $L4'$, correspondientes a $XO1$,

$XO2$, $XO3$, $XO4$, $L1$, $L2$, $L3$ y $L4$ luego de las traslaciones correspondientes.

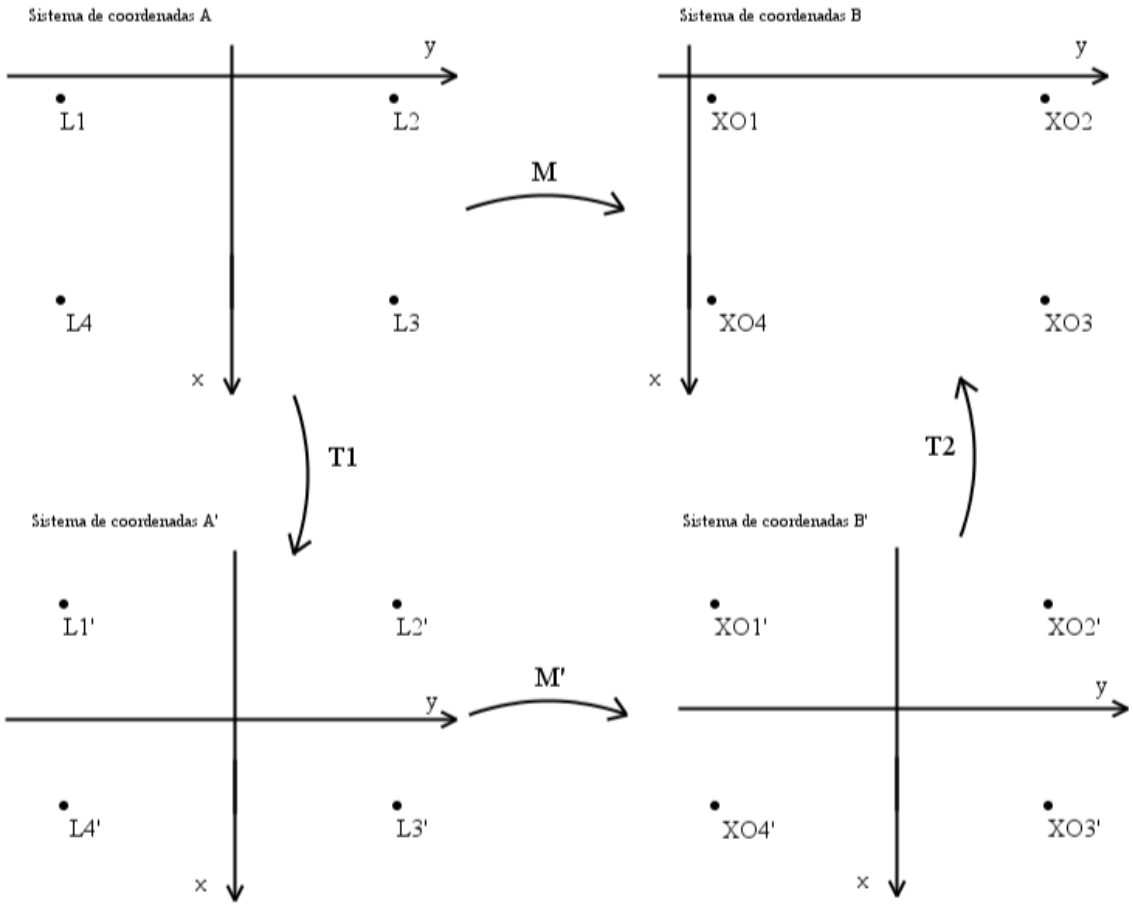

Figura 3.15: Descomposición de la transformación M

Luego de las traslaciones se halla la transformación lineal M' que hace corresponder los cuatro puntos del sistema de coordenadas A' en los cuatro puntos del sistema de coordenadas B'. Para esto se busca una matriz de 2x2, utilizando mínimos cuadrados para hallar la que mejor ajuste. En la figura 3.15 se muestra un esquema de las traslaciones y transformaciones utilizadas.

Resulta de esto que para realizar la transformación M, se debe realizar la traslación $T1$, la transformación M' y la traslación $T2$, en ese orden.

Hallada esta matriz, para cada punto que llegue se le aplicarán las traslaciones y transformación para lograr el cambio de coordenadas.

El hecho de utilizar 4 puntos ayuda a absorber pequeñas desviaciones en el proceso de calibración, logrando así mejor tolerancia a errores en este proceso. Pero esta tolerancia es limitada y corrimientos grandes de los puntos correctos en el proceso de calibración generan grandes desviaciones entre la punta del lápiz y el cursor en la pantalla.

3.4.5. El filtro de Kalman

Motivación

Dada la naturaleza funcional de Lapix, este se encuentra en un entorno ruidoso. Debido a la aplicación que se espera del mismo, se deben minimizar todo tipo de errores a la hora de representar los trazos sobre la pantalla.

En condiciones normales, y sin ninguna prevención, cualquier alteración o pequeño ruido generará un error en la representación de los trazos. La magnitud del mismo dependerá de la magnitud de la perturbación. Se debe entonces encontrar la o las maneras de minimizar al máximo todas las posibles fuentes de ruido.

Este concepto se tuvo en cuenta desde que se comenzó a concebir el diseño de Lapix. Se calculó y estudió la frecuencia a la que debía realizarse el muestreo de manera de lograr reducir a un nivel aceptable el error producido por la cuantización (2.2.2). También se estudió el impacto del error de redondeo, el cual se determinó era despreciable. En el ámbito hardware, las placas fueron diseñadas de manera de minimizar las posibles interferencias electromagnéticas externas. Incluso en este ámbito se blindó totalmente el receptor para atenuar estos efectos.

Si bien todas estas medidas adoptadas son efectivas en cada uno de sus niveles y juntas contribuyen a minimizar el error, dado que Lapix funcionará en un ordenador existe otra posibilidad a implementar para contribuir con la minimización del error. Esta posibilidad es implementar un filtrado mediante software el cual combata el natural ruido ambiente y además, sea efectivo contra posibles perturbaciones que generen errores mayores y permintan seguir el trazo en pantalla correctamente.

A la hora de implementar un filtro de software, abundan las posibilidades. Sin embargo, pare este caso particular, donde el filtrado debe realizarse en tiempo real, no se dispone a priori de todas las muestras a filtrar y que además el espacio de memoria para guardar datos para utilizar en el filtrado es limitado, el espacio de filtros aplicables se reduce considerablemente. Uno de los integrantes realizó el curso brindado por el IIE “Tratamiento Estadístico de Señales” [64] (en el cual como monografía final se desarrolla un filtro aplicable al proyecto Lapix). El filtro seleccionado en dicho trabajo es el filtro de Kalman.

En este capítulo se aborda entonces el desarrollo del filtro de Kalman para Lapix, donde se muestra además un beneficio adicional a la reducción de ruido y es la estilización de los trazos dibujados en pantalla.

Introducción al filtro de Kalman

El filtro de Kalman, es un filtro recursivo. Un rasgo distintivo del mismo es que se basa en describir al sistema en variables de estado, por lo tanto, el filtro requiere un modelo del sistema en cuestión. Otra particularidad interesante, es que la solución es calculada recursivamente, en particular, cada actualización de la estimación de un estado se calcula en base a la estimación del estado anterior y a las nuevas mediciones obtenidas. Es por esto que no requiere almacenar estados anteriores para realizar la estimación, mejor aún, el filtro de Kalman es más eficiente que realizar el filtrado en cada paso habiendo guardado todos los estados anteriores. En sí, el filtro de Kalman es ideal para su implementación en ordenadores y ha sido aplicado exitosamente en varios campos como ser el aeroespacial y el aeronáutico.[65] [66]

Ecuaciones del filtro de Kalman

Como se decía anteriormente, el filtro de Kalman parte de una descripción en variables de estado del sistema. Para ello se define el siguiente sistema:

$$\begin{cases} \underline{x}(k+1) = \underline{\underline{\Phi}}(k)\underline{x}(k) + \underline{w}(k) \\ \underline{y}(k) = \underline{\underline{C}}(k)\underline{x}(k) + \underline{r}(k) \end{cases} \tag{3.1}$$

Donde:

$\underline{x}(k)$: *estados del proceso* $(n\mathtt{x}1)$
$\underline{\underline{\Phi}}(k)$: *dinámica del proceso* $(n\mathtt{x}n)$
$\underline{w}(k)$: *ruido blanco de media nula* $(n\mathtt{x}1)$
$\underline{y}(k)$: *mediciones* $(m\mathtt{x}1)$
$\underline{\underline{C}}(k)$: *matriz de observaciones* $(m\mathtt{x}n)$
$\underline{r}(k)$: *ruido blanco de media nula* $(m\mathtt{x}1)$

El sistema descripto se corresponde con el modelado en variables de estado del sistema.

Como se mencionó anteriormente, el filtro de Kalman se basa en un algoritmo recursivo. Mediante el mismo se realiza una predicción del siguiente paso o estado del sistema, en base al estado anterior, el error cometido en la predicción anterior y otros parámetros. Para realizar estos pasos, además del sistema anterior, se definen algunos parámetros mediante los cuales se llega a la solución del filtro y se realizan los pasos de las iteraciones. Los mismos se presentan a continuación y luego un esquema de funcionamiento del filtro.

Matrices de Autocorrelación

$\underline{\underline{Q}}(k)$: *matriz de autocorrelación de* $\underline{w}(k)$. Representa el ruido del modelo.

$$Q_k = \begin{bmatrix} \sigma_w^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_w^2 \end{bmatrix}$$

$\underline{\underline{R}}(k)$: *matriz de autocorrelación de* $\underline{r}(k)$. Representa el ruido.

$$R_k = \begin{bmatrix} \sigma_r^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_r^2 \end{bmatrix}$$

Covarianza del Error (definición)

$$\underline{\underline{P}}_k = E\{\underline{e}_k \underline{e}_k^*\}$$

$$\underline{e}_k = \underline{x}_k - \underline{\widehat{x}}_k$$

Ganancia de Kalman

$$\underline{\underline{K}}_k = \underline{\underline{P}}_k^- \underline{\underline{C}}_k^t [\underline{\underline{C}}_k \underline{\underline{P}}_k^- \underline{\underline{C}}_k^t + \underline{\underline{R}}_k]^{-1}$$

Actualización del Estimador

$$\underline{\widehat{x}}_k = \underline{\widehat{x}}_k^- + \underline{\underline{K}}_k [\underline{y}_k - \underline{\underline{C}} \underline{\widehat{x}}_k^-]$$

Covarianza del error (iteración)

$$\underline{\underline{P}}_k = (\underline{\underline{I}} - \underline{\underline{K}}_k \underline{\underline{C}}_k) \underline{\underline{P}}_k^-$$

Proyección para el siguiente estado (k = k + 1)

$$\underline{\widehat{x}}_{k+1}^{-} = \underline{\underline{\Phi}}_k \underline{\widehat{x}}_x$$

$$\underline{\underline{P}}_{k+1}^{-} = \underline{\underline{\Phi}}_k \underline{\underline{P}}_k \underline{\underline{\Phi}}_k^t + \underline{\underline{Q}}_k$$

Diagrama de funcionamiento del filtro

El funcionamiento del filtro se da según se muestra en la figura 3.16. Luego de la inicialización del filtro (paso k=0) se realiza la primera proyección (paso k=1). En base a la misma se calcula la ganancia de Kalman y cuando de obtiene la medición y se actualiza la estimación realizada previamente. Posteriormente se calcula la matriz P y se realiza la nueva proyección.

En la sección 3.4.5 se listan las expresiones mediante las cuales se hallan las proyecciones, la ganancia y se actualiza el estimador en el filtro de Kalman.

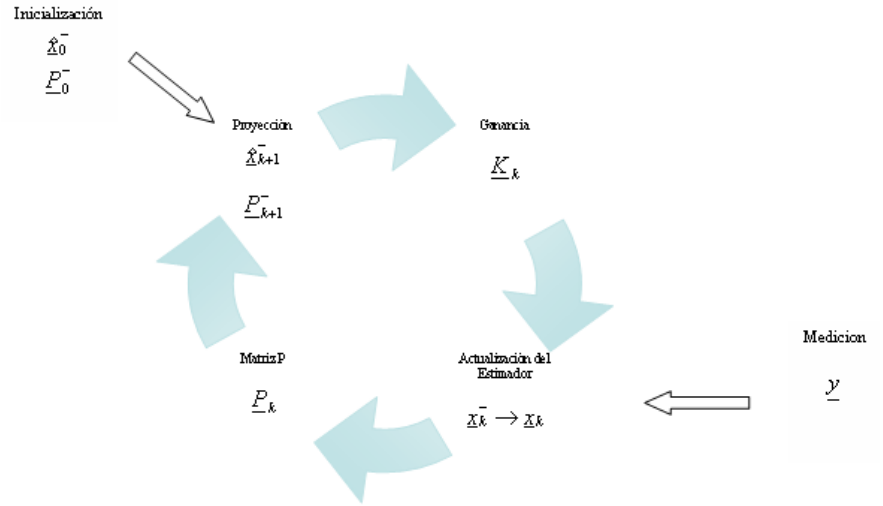

Figura 3.16: Diagrama de funcionamiento del filtro de Kalman

Observaciones sobre la utilización del filtro para el caso de Lapix

Cabe destacar que según el esquema de la figura 3.16, y es como en general se utiliza, el filtro se utiliza para realizar una predicción del siguiente

paso o estado del proceso. Es decir, en general lo que se busca como salida, es la proyección del siguiente estado:

$$\underline{\widehat{x}}_{k+1}^{-}$$

En nuestro caso en particular lo que nos interesa es realizar una disminución del ruido y lograr suavizar el trazo que se aprecia en pantalla, por lo que no es adecuado, y no tiene sentido, tomar la estimación de la posición siguiente.

En su lugar, lo que se tomará como salida del filtro será la actualización del estimador, es decir $\underline{\widehat{x}}_k$. Lo que se logra con esto, es que en base al modelo y a las mediciones anteriores se realiza una estimación de la posición, luego, cuando llega la medición, la cual está contaminada con ruido, se actualiza el estimador y de este modo se contrasta el modelo con la medición ruidosa obteniendo una serie de posiciones correlacionadas entre sí donde se disminuye el efecto del ruido y se suaviza el trazo.

El diagrama de funcionamiento a utilizar se muestra en la figura 3.17.

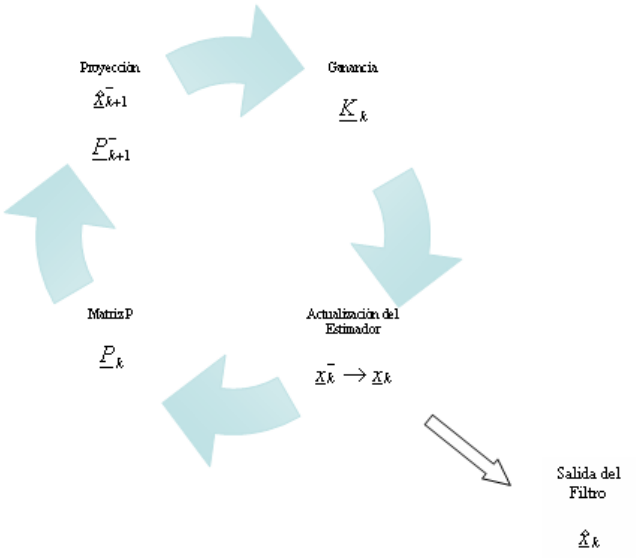

Figura 3.17: Diagrama de funcionamiento a utilizar del filtro de Kalman

Modelo para el Filtro del Kalman

Como se expresó en instancias anteriores, el filtro requiere expresar el sistema a tratar en variables de estado. Esto nos implica tener un modelo del sistema a analizar. En nuestro caso, dado que el filtro será utilizado con un lápiz digitalizador, se hace necesaria la introducción de un modelo adecuado para representar los movimientos de una punta de un lápiz al escribir o realizar distintos tipos de trazos.

Luego de una ardua búsqueda y comparación de modelos [67], el modelo encontrado, sale del trabajo para el doctorado del Sr. Mario Enrique Munich [68].

El mismo trata de una aplicación digitalizadora donde, mediante una cámara, utilizan la diferencia cuadro a cuadro mientras se escribe, para ver dónde se ha generado trazo. En el mismo utilizan un filtro de Kalman para predecir la posición de la punta de modo de no tener que realizar la diferencia de todo el cuadro (implicando esto más procesamiento), pudiendo analizar sólo el sector donde se predice que estará la punta.

El Modelo

La consigna es que el filtro prediga la posición más probable de la punta basándose en la posición, velocidad y aceleración anteriores. De la referencia mencionada anteriormente, se tiene que la mejor forma para cumplir con este objetivo es asumir un modelo de Random-Walk para la aceleración de la punta del lápiz.

El RW en la aceleración

El denominado "Random-Walk" (o RW) [69] es una formalización matemática para representar una trayectoria que consiste en realizar pasos aleatorios sucesivos. Debe su nombre del ejemplo de una persona que toma pasos de largo fijo en direcciones arbitrarias al desplazarse. Tiene aplicaciones en variados campos, por ejemplo se utiliza para modelar la trayectoria de una molécula al viajar en un líquido o en un gas, el camino realizado por un animal salvaje, etc.

Desde un punto de vista más formal, el proceso de Random-Walk resulta cuando señales no correlacionadas son integradas.

La ecuación en variables de estado que representa el random walk es la siguiente:

$$\dot{x} = w \tag{3.2}$$

Donde se tiene que:

- $E[w(t)w(\tau)] = q(t)\delta(t-\tau)$
- w representa un ruido, que en nuestro caso será blanco aditivo y Gaussiano.
- x es el vector de estados elegidos en la representación.

El proceso se representa en la figura 3.18.

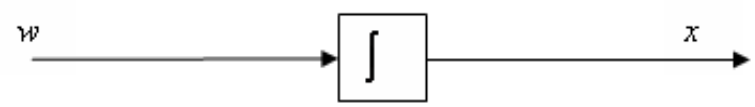

Figura 3.18: Random Walk

Para representar el random-walk en forma discreta, se tiene:

$$x_{k+1} = x_k + w_k \tag{3.3}$$

Trasladando esto a la aceleración de la punta del lápiz (recordar que el modelo adoptado asume un comportamiento de RW para la aceleración de la punta del lápiz), se tiene:

$$\begin{aligned} \dot{a} &= w \\ a_{k+1} &= a_k + w_k \end{aligned} \tag{3.4}$$

Integración del RW al modelo

Visto entonces la forma de expresar la suposición de RW para la aceleración, como factor extra sólo basta considerar las ecuaciones de cinemática para la velocidad y la posición.

Agrupando todo esto, se tiene:

$$\begin{cases} \dot{x}(t) = v(y) \\ \dot{v}(t) = a(t) \\ \dot{a}(t) = n_a(t) \\ y(t) = x(t) + n_y(t) \end{cases} \tag{3.5}$$

Donde:

- $x(t)$, $a(t)$ y $v(t)$ son las componentes bidimensionales de la posición, velocidad y aceleración de la punta respectivamente.
- $n_a(t)$ y $n_y(t)$ son ruido blanco, aditivo, Gaussiano y de media nula.
- $y(t)$, la salida del modelo, es la posición de la punta contaminada con ruido blanco.

Para poder utilizar el modelo presentado, se debe obtener la forma discreta del mismo dado que se trabajará con un sistema discreto por naturaleza. En la siguiente sección se presenta la discretización del modelo junto con la elección de los parámetros del mismo.

Discretización del modelo y parámetros

El modelo presentado en la ecuación 3.5, representa un modelo continuo. Sin embargo el caso de aplicación, dado que se trabaja con muestras de la posición de la punta, es discreto. Se debe entonces llegar a la forma discreta del mismo para que sea aplicable en el caso de Lapix.

A continuación se presenta el modelo discretizado. Los pasos para llegar al mismo pueden se consultados en las referencias citadas. [68] [70] [71] [72]

$$\begin{cases} x(k+1) = x(k) + v(k)\triangle t + \frac{1}{2}a(k)\triangle t^2 \\ v(k+1) = v(k) + a(k)\triangle t \\ a(k+1) = a(k) + n_a(k)\triangle t \\ y(k) = x(k) + n_y(k) \end{cases} \tag{3.6}$$

Donde $\triangle t = 1$

Si llamamos ahora A a la matriz de transición y C a la matriz de observación, el modelo expresado en notación matricial queda de la siguiente forma:

$$\begin{cases} X(k+1) = AX(k) + n_X(k) \\ y(k) = CX(k) + n_y(k) \end{cases} \tag{3.7}$$

Donde:

$$X(k) = \begin{bmatrix} x(k) \\ v(k) \\ a(k) \end{bmatrix} \quad n_X(k) = \begin{bmatrix} 0 \\ 0 \\ n_a(k) \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 & 0{,}5 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0{,}5 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Donde en la matriz $R(k)$ se esta asumiendo que el ruido afecta a las dos componentes de igual manera, y en la matriz $Q(k)$ se hace una diferencia en las varianzas del Random-Walk según sea la componente x o y. Esto nos da 2 posibles parámetros a variar de modo de optimizar el funcionamiento del filtro. Los valores de la potencia del ruido serán determinados en base a las especificaciones del sistema.
Luego se escriben las ecuaciones del filtro de Kalman en base a las expresadas en la sección 3.4.5

Como observación del modelo, notar que se asumen las matrices Q y R invariantes en el tiempo y diagonales, esto implica que las medidas así como las componentes de estado no están correlacionadas. Las únicas componentes con incertidumbre en el modelo son las aceleraciones. Asumir que éstas no están correlacionadas no es muy real dado que al escribir los músculos que controlan el movimiento de la punta no mueven a esta en la dirección x e y independientemente. Sin embargo de esta forma se logra una primera aproximación al problema que permite evaluar la solución implementada.

Determinación de los parámetros

Inicialización del filtro

Para realizar la inicialización del filtro se toma como primer elemento el vector nulo. La matriz P en general se elige de forma tal que asuma que el error inicial es grande, es decir que no se confía en la estimación inicial. Para determinar la misma, se le asigna un valor grande al principio y solo basta realizar algunas simulaciones para ajustar dichos valores y verificar que no se esté confiando en la estimación inicial. En general su inicialización no influye demasiado.

$$X_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad P_0 = \begin{pmatrix} 10e^{50} & 0 & 0 & 0 & 0 & 0 \\ 0 & 10e^{50} & 0 & 0 & 0 & 0 \\ 0 & 0 & 10e^{-1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 10e^{-1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 10e^{-5} & 0 \\ 0 & 0 & 0 & 0 & 0 & 10e^{-5} \end{pmatrix}$$

Varianza del Ruido

Este parámetro representa el ruido presente en todo el sistema. El hardware del proyecto Lapix se realizó intentando minimizar todas las posibles fuentes de ruido e interferencia, sin embargo debido a los problemas mencionados en la sección 2.3 y 6.1 hay una componente de ruido importante que afecta al sistema. Este ruido, proveniente del jitter del receptor de IR, en adición con el resto del ruido, se asumirá que es blanco (de modo de que pueda utilizarse con el modelo de filtro a utilizar) de media nula y Gaussiano.

Asumir estas características es a los efectos de simplificar el problema a un nivel que permita un implementación práctica, evidentemente dado que el ruido probablemente no tenga estas características el modelo se ve afectado, pero sin embargo se logra implementar una solución que se aproxima bastante.

Dadas las características asumidas para el ruido contaminante, para determinar completamente el mismo se necesita determinar la varianza. Para ello una vez que se disponía del hardware completo, y el driver funcionando, se procedió a tomar muestras.

Las muestras consistieron en posicionar al lápiz en ciertos puntos del área de escritura y mantenerlo estático durante cierta cantidad de tiempo suficiente como para generar del orden de mil muestras.

Las muestras son tomadas con la version 6.11 del driver, donde se toma L.x y L.y después del filtrado, es decir, los puntos de corte de las circunferencias expresados en períodos de reloj de 80MHz. Un vez hecho esto, las muestras son procesadas en Matlab, donde son convertidas a cm. Luego se halla la varianza de las muestras, es decir la varianza en cada punto tomado. Finalmente se promedian estos valores obteniéndose un único valor de la varianza del ruido.

Se construyó una grilla en una hoja tamaño A4 sobre la cual se tomaron alrededor de 70 puntos. A continuación se muestran algunas de las muestras tomadas pasadas a mm^2 3.4.5:

	$VarX(mm^2)$	$VarY(mm^2)$
punto1	0,0003964	0,0002
punto2	0,0006394	0,0003
punto3	0,0013244	0,0005
punto4	0,0012322	0,0004
punto5	0.0022472	0,0006

Cuadro 3.1: Algunas varianzas obtenidas de las muestras tomadas

Como se puede apreciar en la tabla anterior, la hipótesis de que ambas coordenadas se contaminan por igual no es aplicable. Es por ello que en el filtro no se asumirá esta hipótesis y se cargarán los valores de varianza correspondientes a cada coordenada. La dependencia de la varianza con al posición se asumió constante y se tomó el peor caso.

Hallando los promedios de varianza se tiene:

Promedio de varianza ejeX (cm^2)	5.84×10^{-004}
Promedio de varianza ejeY (cm^2)	3.65×10^{-004}

Cuadro 3.2: Varianzas obtenidas de las muestras tomadas

En resumen, las varianzas a poner en el driver medidas en períodos de reloj de 80MHz son:

Eje X = 56.36Tck
Eje Y = 5.456Tck

Para el caso de las simulaciones que se detallan a continuación y para la búsqueda de valores óptimos, se utilizará un valor de ruido que implique el peor caso, es decir que lleve al límite las especificaciones requeridas.

Potencia del Random-Walk

Como se mencionaba en las secciones anteriores, el sistema posee dos grados de libertad, estos son las varianzas n_{ax} *y* n_{ay}. El valor de las mismas afecta rotundamente el desempeño del filtro y son determinantes en la efectividad del mismo.

Empíricamente, se puede observar que cuanto más grande sean dichas varianzas, más fielmente se sigue a la señal, sin embargo esto tiene como contraparte que también se sigue al ruido. Si por otro lado son muy pequeñas, se observa una salida del sistema libre de todo ruido y con trazos de curvatura elegante, pero que no siguen fielmente el camino del trazo original.

Como ejemplo de lo mencionado, en la figura 3.19 se muestra un trazo con forma de sinusoide al que se le adiciona un ruido blanco. En este caso el filtro se implemento con valores altos de potencia de Random - Walk. Como resultado de ello se tiene que se sigue exactamente al trazo original, pero también al ruido contaminante, por lo que la señal filtrada se ve muy afectada por el ruido. Para este caso $n_{ax} = 0{,}04$cm y $n_{ay} = 0{,}01$cm. Cabe destacar que las figuras que se muestran a continuación son el resultado de simulaciones realizadas en Matlab, donde se genera el trazo mediante un mouse y luego se se adiciona ruido blanco Gaussiano de media nula.

En cambio, en la figura 3.20 se corre el filtro con valores pequeños de RW. Se puede ver claramente que el trazo queda con curvaturas excelentes (a), sin embargo los cambios en la trayectoria se siguen pero con algo de "retraso" (b).

En sí se tiene un compromiso entre qué tanto se puede disminuir el ruido y qué tanto se puede seguir fielmente a la señal.

Determinación de los valores óptimos del Random - Walk

Como se dejó en claro en las secciones anteriores, el grado de libertad del modelo radica en la potencia asignada al Random - Walk. Se vio que variando este parámetro cambia radicalmente el comportamiento y efectividad de la solución a implementar. El problema radica entonces en determinar de

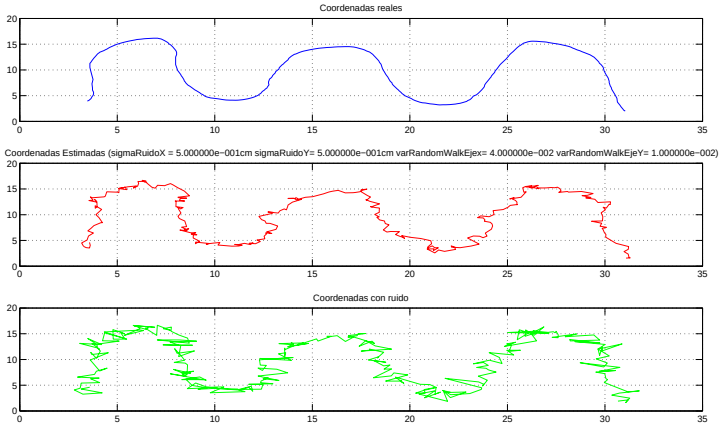

Figura 3.19: Ejemplo con alta potencia de Random - Walk

algún modo la potencia de RW a asignar.

Para tales efectos, se realiza un algoritmo el cual ensaya posibles combinaciones de valores, compara el resultado y devuelve la que haya tenido mejor desempeño.
Para tales efectos se hace necesaria una forma objetiva de medir la efectividad de una combinación de valores de n_{ax} y n_{ay} dada.

El candidato más intuitivo, y el que se utiliza para dichos efectos, es la distancia media cuadrática punto a punto. Se tomará el total de las distancias punto a punto al cuadrado y se divide entre la cantidad de muestras y se le aplica la raíz.

Esto es:

$$Dist = \sqrt{\frac{\sum_i ((x_{iR} - x_{iE})^2 + (y_{iR} - y_{iE})^2)}{N}} \tag{3.8}$$

Donde el subíndice R denota "real" y el E "estimada". N es la cantidad de muestras.

Para correr el algoritmo se utiliza un trazo generado mediante Matlab, con la funcionalidad de obtener la posición del puntero.

Posteriormente, a dicho trazo se le adiciona ruido blanco, Gaussiano y de media nula. La potencia de este ruido se determinará considerando el peor

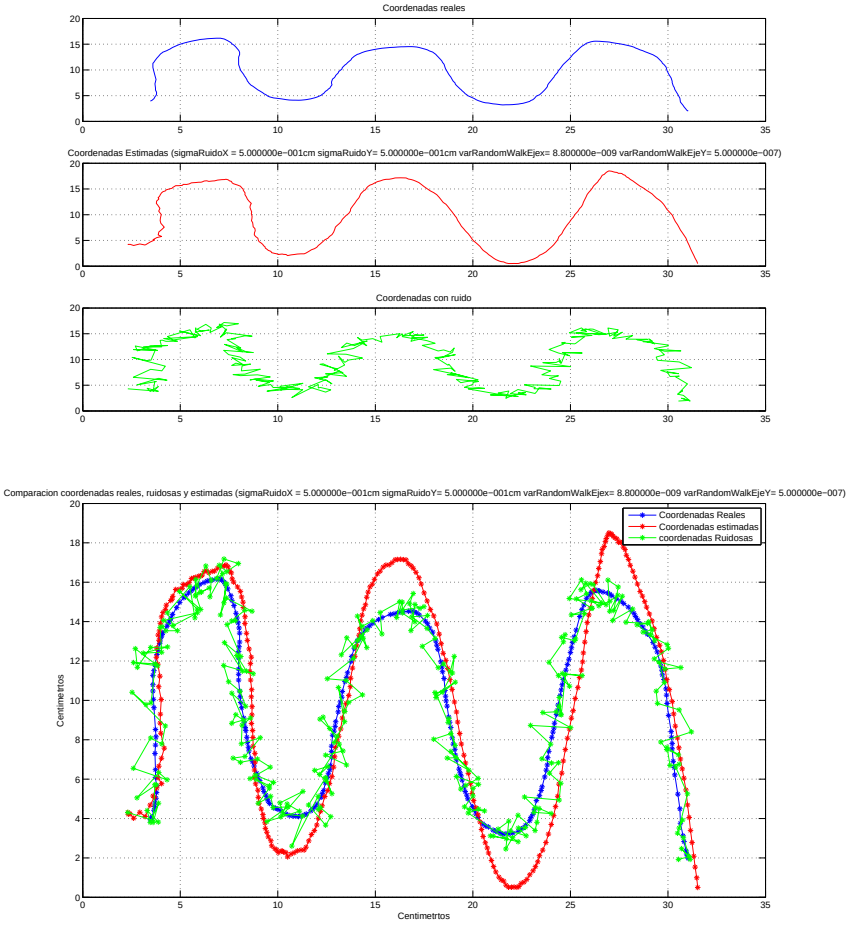

Figura 3.20: Ejemplo con baja potencia de Random - Walk

caso para las especificaciones.

Luego de obtener la señal contaminada, se crea una matriz con todas las combinaciones posibles y razonables, de n_{ax} y n_{ay} y se filtra la misma tantas veces como combinaciones se hayan formado.

El hecho de probar un gran número de combinaciones con un ruido en particular puede provocar que el resultado obtenido no sea verdaderamente el óptimo sino que sea un caso particular para los valores de ruido adicionados en ese caso.

Es por esto que el procedimiento de búsqueda será como se explica a conti-

nuación

- Se realizan todas las combinaciones posibles, y se ensaya cada una varias veces con varios ruidos distintos.
- Para cada una se halla la distancia a la señal real. Finalmente se promedian todas las distancias para una combinación
- Luego se elige entre todas las combinaciones la que haya tenido la menor distancia promedio.

El desempeño se mide según la ecuación 3.8.

Dado que existe una infinidad de combinaciones posibles para los valores de n_{ax} y n_{ay}, se realiza en primera instancia búsquedas con valores muy genéricos de manera de determinar el orden de los valores buscados.

Se evalúa entonces el siguiente conjunto de valores:

$$n_{ax} = [5 \times 10^{1}\ 5 \times 10^{0}\ 5 \times 10^{-1}\ 5 \times 10^{-2}\ 5 \times 10^{-3}\ 5 \times 10^{-4}]$$
$$n_{ay} = [5 \times 10^{1}\ 5 \times 10^{0}\ 5 \times 10^{-1}\ 5 \times 10^{-2}\ 5 \times 10^{-3}\ 5 \times 10^{-4}]$$

que arroja como resultado:

$$n_{ax} = 5{,}0 \times 10^{-4}$$
$$n_{ay} = 5{,}0 \times 10^{-3}$$

Sabiendo entonces aproximadamente el orden de los valores buscados, se realiza una búsqueda más específica en un entorno de los órdenes obtenidos. Se evalúa el siguiente conjunto de valores:

$$n_{ax} = [[5 : 0{,}1 : 9{,}9] \times 1 \times 10^{-3}\ [1 : 0{,}1 : 9{,}9] \times 1 \times 10^{-4}\ [1 : 0{,}1 : 5] \times 1 \times 10^{-5}]$$
$$n_{ay} = [[5 : 0{,}1 : 9{,}9] \times 1 \times 10^{-2}\ [1 : 0{,}1 : 9{,}9] \times 1 \times 10^{-3}\ [1 : 0{,}1 : 5] \times 1 \times 10^{-4}]$$

que arroja como resultado:

$$n_{ax} = 8{,}8 \times 10^{-4}\text{Tck}$$
$$n_{ay} = 5{,}0 \times 10^{-4}\text{Tck}$$

Para evaluar el desempeño de los valores hallados se realizan simulaciones en Matlab. A continuación se presentan dichas simulaciones y además se muestran resultados experimentales.

Simulaciones y Resultados Experimentales

Luego de haber determinado los valores óptimos para la potencia del Random - Walk en las direcciones x e y se realizan algunas simulaciones para comprobar el desempeño del filtro.

Se obtuvieron dos tipos de datos para realizar las simulaciones, datos generados mediante Matlab y datos reales obtenidos directamente de hardware con el cual se utilizará el filtro.

Simulaciones en Matlab

Para evaluar el comportamiento del filtro previo a su utilización en Lapix se realizan simulaciones utilizando un trazo generado artificialmente.

Se realizan varias simulaciones utilizando un mismo trazo y variando la componente de ruido. En primera instancia se utiliza la misma potencia de ruido utilizada para la búsqueda de la potencia óptima de RW.

En la figura 3.21 puede observarse dicha simulación. Cabe destacar, que el criterio que se utiliza para hallar el óptimo busca que la señal estimada esté lo más pegada posible a la señal real. Esto es importante pero sin embargo deja un poco de lado otra evaluación, que las curvas de los trazados parezcan curvas y que no queden alteradas en presencia de ruido. En la figura anterior si se contrasta a simple vista el trazo real con el estimado (figura 3.22) se aprecia cierta diferencia la cual puede no ser aceptable para una aplicación de lápiz digitalizador. Sin embargo, si se incluye la componente de ruido y se comparan las 3 señales (figura 3.23), se puede apreciar claramente que el efecto del filtro es muy beneficioso. Es decir, si bien no se ha representado exactamente el trazo, en una aplicación real (con estas condiciones de ruido) es preferible tener las pequeñas desviaciones que se aprecian a tener un trazo totalmente distorsionado por la componente de ruido.

Sin embargo, como se decía, podemos notar que la estilización del trazo no es tenida tan en cuenta. Dado que sabemos cómo se relaciona la potencia del RW con la forma en que el trazo estimado sigue a la señal, a partir de los óptimos hallados podemos "jugar" con los valores para lograr el grado de curvatura deseada en la señal. Es decir, sabemos que al disminuir los valores de potencia del RW, logramos que la señal estimada siga más lentamente a la real, logrando así los efectos de curvatura deseada. Lograr implementar una condición matemática que tenga en cuenta esta condición requeriría tra-

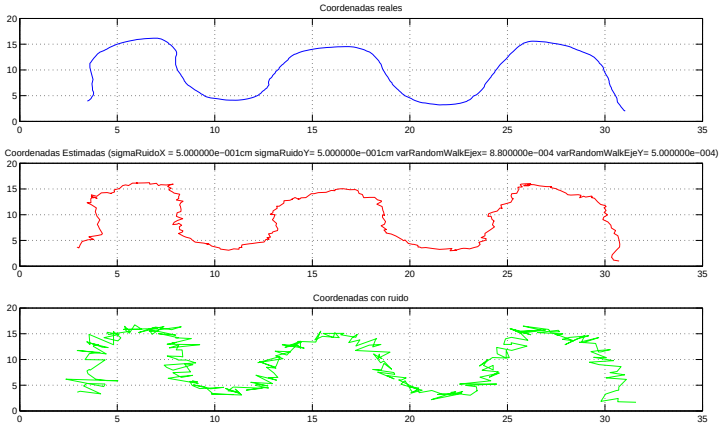

Figura 3.21:

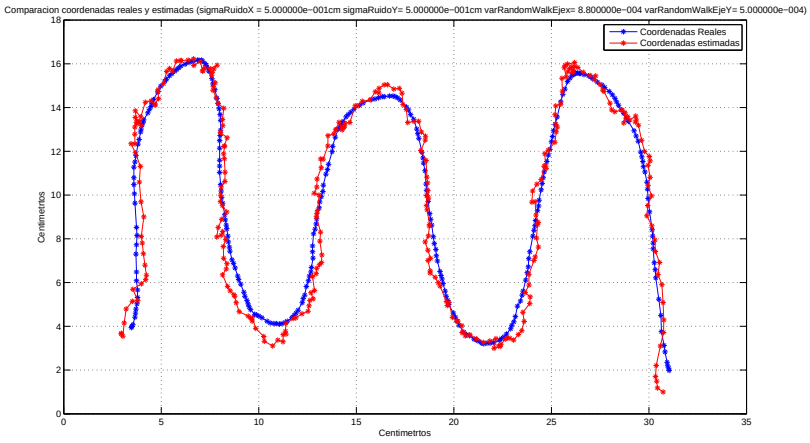

Figura 3.22:

bajar con las derivadas de las señales y el compromiso entre el resultado y el esfuerzo aplicado podría no valer la pena dado que es una condición que puede ser evaluada con total objetividad a simple vista.

Por ejemplo, utilizando los siguientes valores de RW:

$n_{ax} = 8{,}8 \times 10^{-7}\text{Tck}$
$n_{ay} = 5{,}0 \times 10^{-5}\text{Tck}$

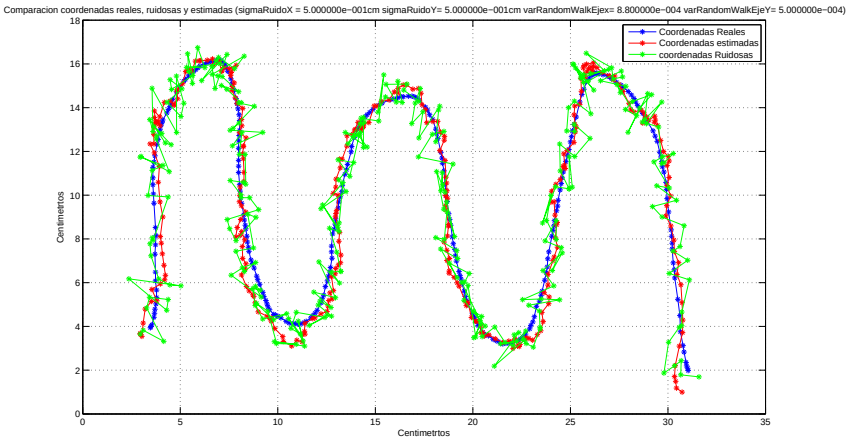

Figura 3.23:

y utilizando la misma señal que en las simulaciones anteriores se logra el resultado que se muestra en la figura 3.24, donde se puede apreciar que mejora la "curvatura" del trazo. El precio que se debe pagar por esto es que, en algunas partes, la señal estimada se aleja un poco de la real. Sin embargo, tengamos en cuenta que el ruido que se le esta sumando al trazo real es extremadamente grande, y nuevamente son preferibles esas pequeñas desviaciones.

Otra simulación realizada consistió en tomar un punto cualquiera de los muestreados para la determinación de la varianza y aplicarle el filtro. El resultado puede verse en la figura 3.25. En la misma puede apreciarse claramente como el filtrado logra reducir la varianza.

A los efectos de lograr evaluar la robustez del filtro se generan condiciones de ruido realmente adversas, como ser $sigma_{ruido} = 2cm$. Los resultados pueden verse en la figura 3.26. A pesar de que la señal ruidosa es prácticamente indistinguible, el filtro logra extraer a grandes rasgos la forma de la señal real. Esta condición es altamente deseable para casos de ruido extremo.

Luego de las anteriores simulaciones, se puede comprobar que los valores óptimos hallados se comportan de buena manera. Por lo tanto se programa el filtro de Kalman en C y se agrega al driver.

A continuación se describen las pruebas realizadas con el propio hardware del Proyecto Lapix.

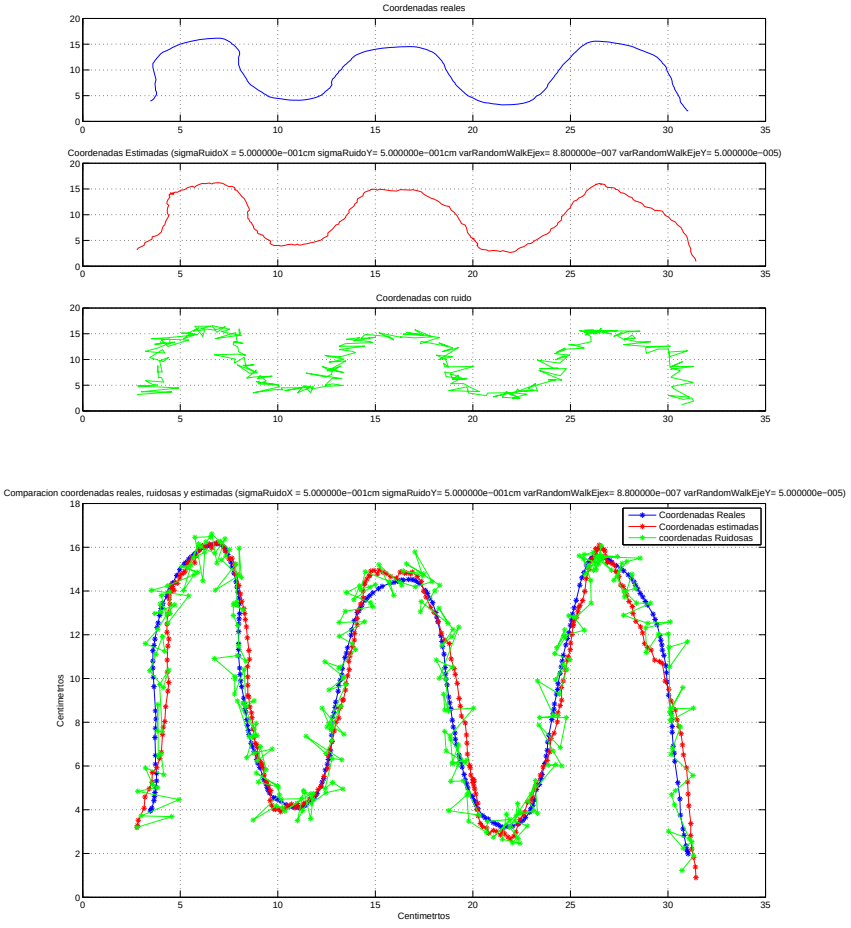

Figura 3.24: Ejemplo ajustando manualmente la potencia de Random - Walk

Comprobación con el hardware

Luego de las simulaciones descriptas anteriormente, como comprobación final de funcionamiento lo que resta es evaluar el desempeño en la realidad. Para ello se realiza un código equivalente al programado en Matlab pero en lenguaje C, el cual es utilizado para el Driver de Lapix.

Una interrogante planteada en el momento de la implementación es en qué etapa del driver es mejor colocar el filtrado. El driver funciona según el siguiente esquema:

1. Se obtienen los valores de los timers

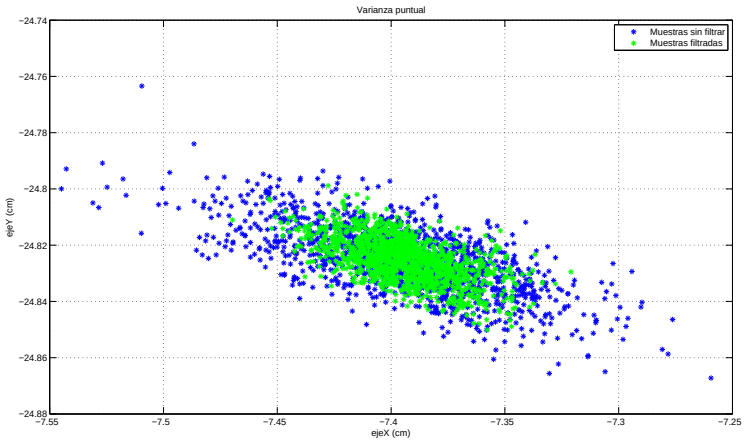

Figura 3.25: Simulación para un punto estático

2. Se hallan los puntos de corte

3. Se convierten a un sistema cartesiano

4. Mediante transformadas se lleva al sistema de la pantalla teniendo en cuenta la calibración

5. Se despliega en pantalla

Colocarlo justo luego del punto 1 no es conveniente dado que en ocasiones, al hallar los puntos de corte se descartan muestras si no existe punto se corte, por lo que se habrían utilizado recursos en vano.

De las otras opciones se elige realizar el filtrado justo después del punto 2, es decir luego de que se obtienen los puntos de corte medidos en unidades de períodos de reloj de 80MHz. La razón de esta elección es que colocarlo luego de la transformación que se realiza en el driver implicaría que se deberían transformar con mucho cuidado los parámetros de ajuste del filtro. Además la rotohomotecia puede no garantizar que se cumpla la relación de distancias en toda el área. Entonces filtrando directamente en el plano de los sensores, se evitan todos estos problemas, además, filtrar luego del punto 4 no agregaría mejoras. Cabe destacar que en este punto es también bastante sencillo el muestreo para obtener por ejemplo los datos de varianza.

En el filtro se cargan los valores de varianza y Random - Walk determinados anteriormente. De la figura 3.27 a la figura 3.32 se pueden ver algunos

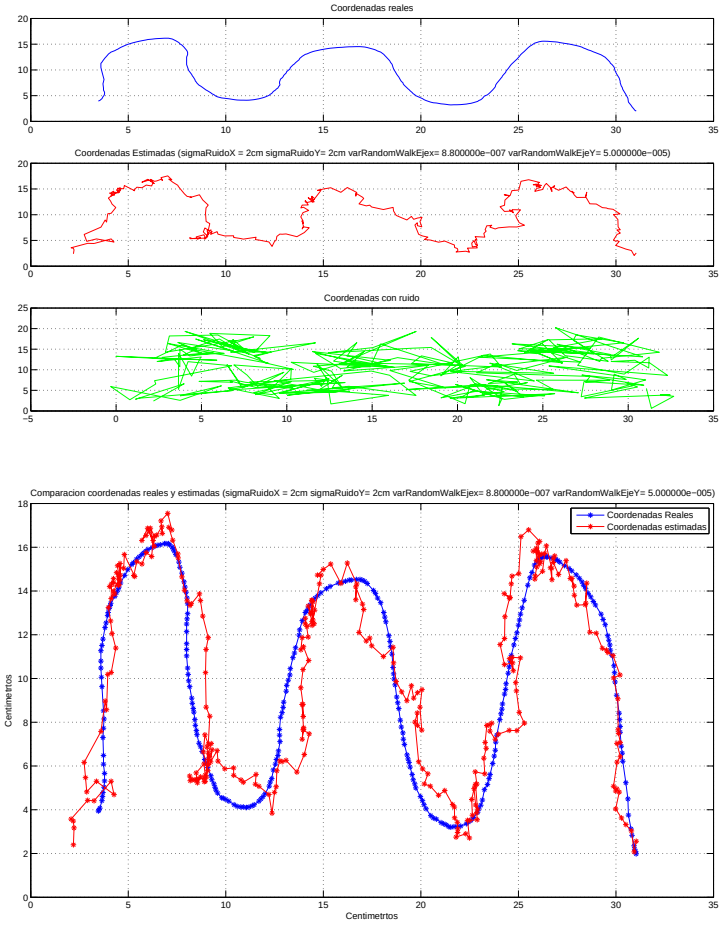

Figura 3.26: Simulación con alta potencia de ruido

ejemplos muestreados directamente con el hardware y software a utilizar.

En las figuras Figura 3.33 a la Figura 3.37 se pueden ver ejemplos de escritura hechos con kolourPaint (aplicación tipo “paint” para Ubuntu).

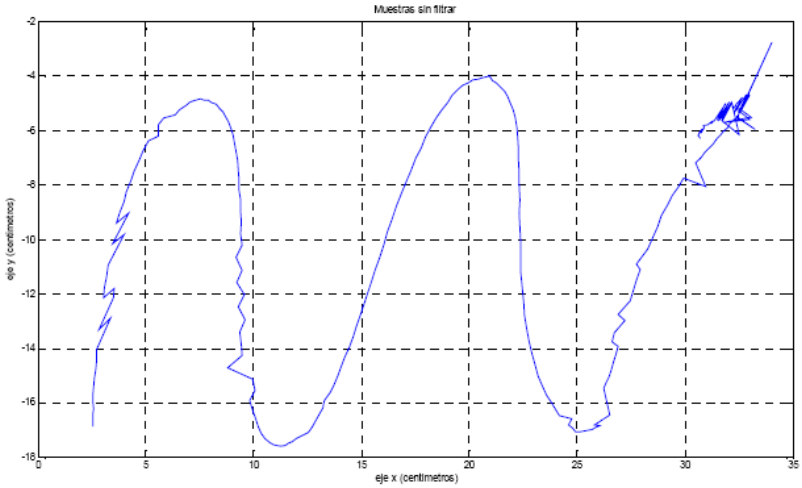

Figura 3.27:

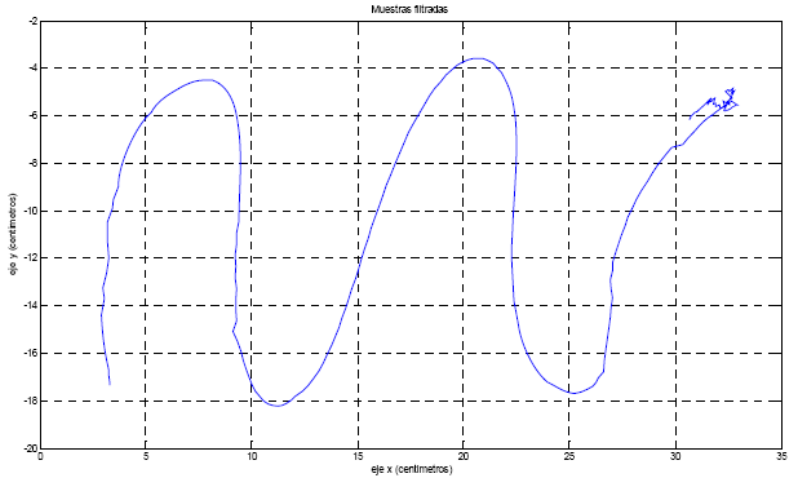

Figura 3.28:

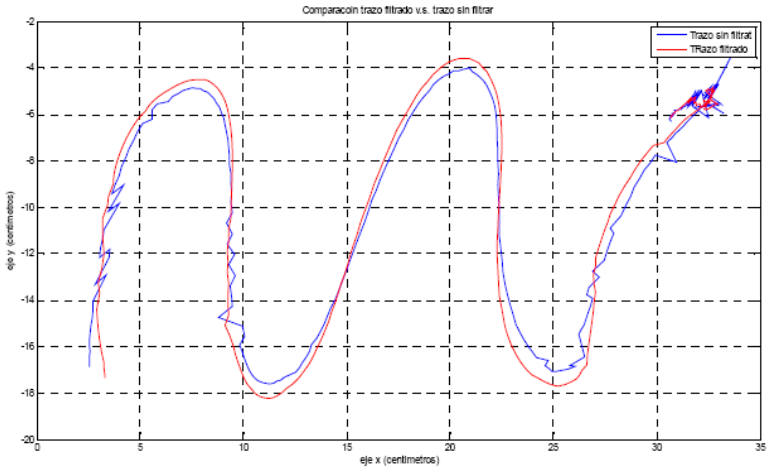

Figura 3.29:

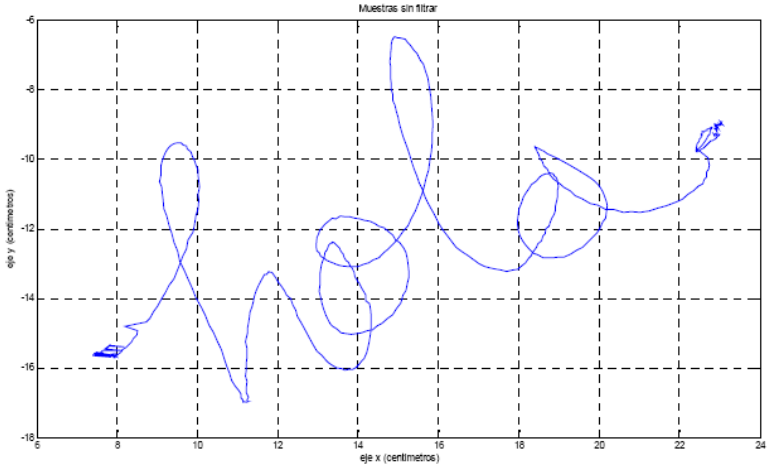

Figura 3.30:

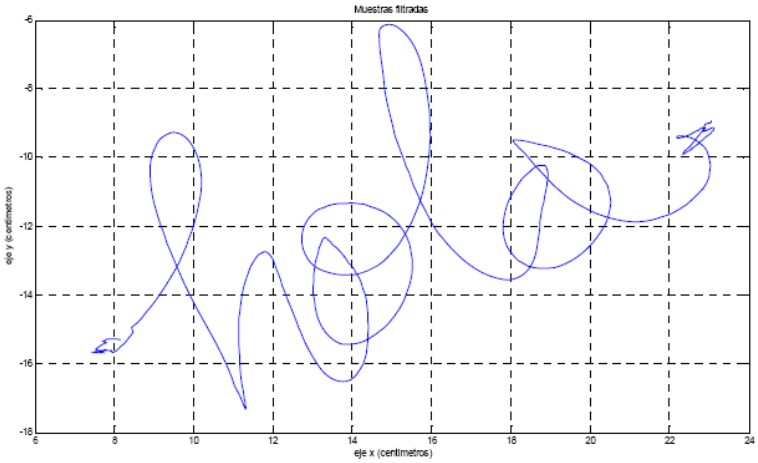

Figura 3.31:

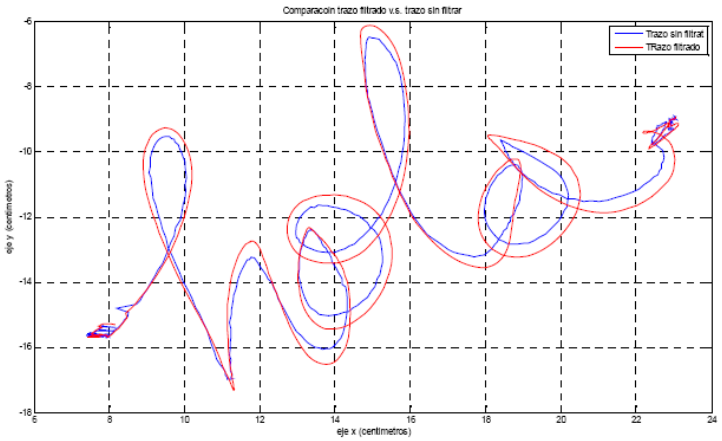

Figura 3.32:

Figura 3.33:

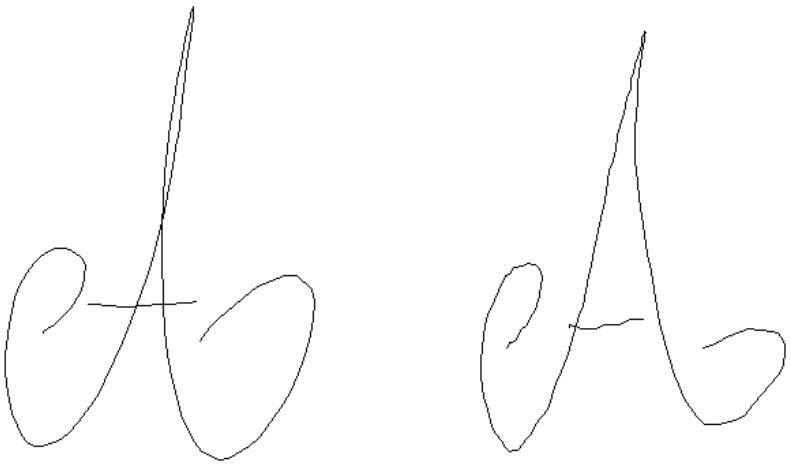

Figura 3.34:

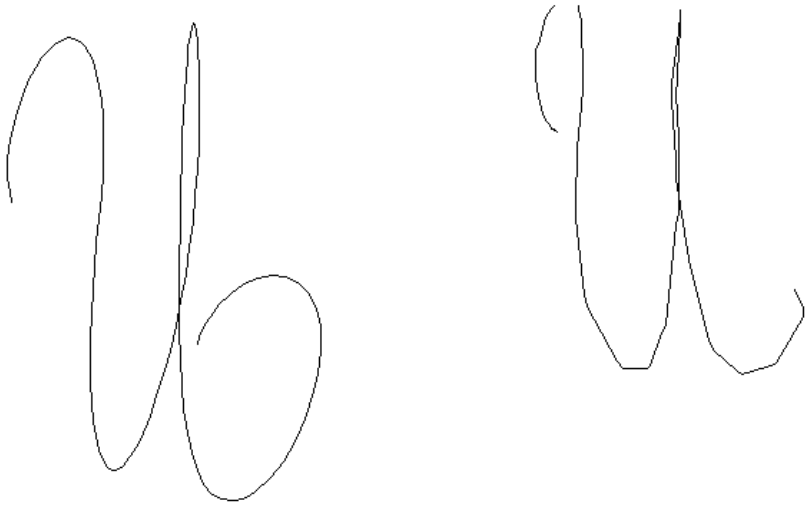

Figura 3.35:

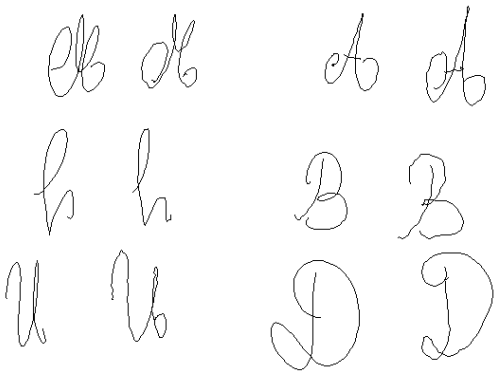

Figura 3.36:

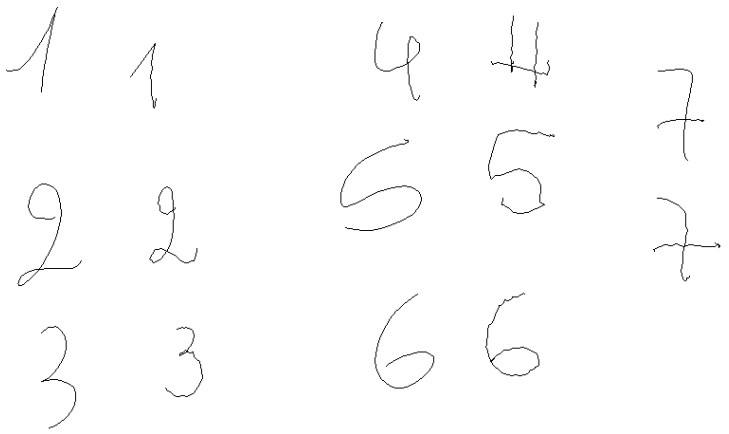

Figura 3.37:

3.4.6. Movimiento del Cursor y clicks

Una vez que se establece correctamente la comunicación con el hardware de Lapix, que se logra recibir datos correctamente, que se realizan todas las transformaciones necesarias para adecuar los datos de posición a un sistema de referencia acorde a la pantalla, y luego de realizado el filtrado correspondiente, se llega a la etapa final donde se debe presentar la información, es decir los movimientos y acciones del lápiz en la pantalla.

Los eventos a mostrar son análogos a los que realiza cualquier mouse común, es decir, desplazamiento del cursor, clicks, etc. Una opción posible para generar dichos eventos, es transmitirle a la máquina la información codificada como protocolo de mouse, y utilizar un driver de mouse adaptado directamente. Esta opción presenta ciertas dificultades ya que dado que la información no se genera teniendo en cuenta este tipo de transmisión habría que "traducir" los movimientos realizados a este protocolo elegido.

Otra opción posible, y la que finalmente fue adoptada, surge de la idea de poder generar mediante software los eventos comunes que realiza cualquier mouse. Es decir, se investigó si existía alguna librería o funciones, compatibles con C, que pudieran manejar el cursor y realizar los eventos requeridos.

Luego de una investigación, se encontró que existía una aplicación que permitía realizar las acciones buscadas y que tenía librerías para utilizar en C. Dicha aplicación es el XServer.

El XServer, servidor que corre en una máquina y puede desplegar eventos en pantalla, permite generar una interfaz gráfica. El sistema de ventanas X distribuye el procesamiento de aplicaciones especificando enlaces cliente-servidor. El servidor provee servicios para acceder a la pantalla, teclado y mouse, mientras que los clientes son las aplicaciones que utilizan estos recursos para interacción con el usuario.

Los movimientos que pueden generarse pueden ser tanto relativos como absolutos, generándose una ventana para estos últimos, lo que lo hace adecuado para las funciones que debe cumplir Lapix, es decir, funcionar como mouse (movimientos relativos) y escribir directamente en pantalla (coordenadas absolutas). Sin embargo finalmente se utilizan sólo movimientos absolutos y se realiza una adaptación para el caso de fuera de pantalla

Para la implementación en C, se recurrió a la librería llamada xlib [73] parte del XServer (sitio oficial [74]) . En la misma se tienen las funciones necesarias para realizar los eventos requeridos. De la misma se utiliza la extensión xtst que es la que permite generar fácilmente "clicks".

Funciones de la xlib y xtst

En la referencia [73] se puede encontrar la lista de funciones disponibles de la librería xlib. A continuación se listan las funciones utilizadas por nuestro driver:

- XWarpPointer()
- XTestFakeButtonEvent()
- XFlush()

XWarpPointer

Esta función es la encargada de generar el movimiento del cursor. Como parámetros se le pasa la coordenada a la cual nos queremos desplazar, referida al origen de la ventana creada, y el display en el cual queremos que se produzca el movimiento.

XTestFakeButtonEvent

Función que simula los clicks. Como parámetro se se la pasa la ventana de referencia, el click que queremos simular y el estado que le queremos asignar, es decir, presionado o liberado.

XFlush

Esta función debe ejecutarse luego de cualquier comando para hacer el mismo efectivo en pantalla.

Las funciones anteriores son las básicas utilizadas para generar los movimientos. Sin embargo, antes de poder generar cualquier actividad en la pantalla se debe generar un entorno adecuado para el XServer.

Conceptos básicos del XServer

El XServer maneja 3 conceptos básicos para organizar su funcionamiento. Se basa en la creación de displays, ventanas (windows), y contextos gráficos (graphic contexts). El manejo de estos conceptos es suficiente para poder realizar las acciones básicas de simulación de eventos de mouse.

Display

Se denomina display al conjunto formado por un usuario con un teclado y/o mouse y una o varias pantallas. Cada display puede manejar varias pantallas, entendiéndose como pantalla el medio físico donde se despliega la información.

Para generar una conexión para un display se utiliza la función:

XOpenDisplay

Esta función abre una conexión al XServer, la cual se utiliza para manejar el display. Se le puede pasar NULL como parámetro donde tomara el valor de display de la variable de entorno.

Window

La ventana o Window es el entorno donde, dentro del display, se realizan las acciones de los programas. Es el concepto que todos conocemos de "ventana". A una ventana se le puede asociar un fondo que sea un pixmap.

Es importante destacar, y en esto se basan las acciones de mover el cursor, que cada ventana tiene su propio sistema de coordenada. Éste tiene su referencia [0,0] en la esquina superior izquierda de la ventana. El eje de las X corre horizontalmente y el eje de las Y verticalmente. Los valores de los ejes se corresponden con el centro de cada pixel de resolución de la pantalla.

Para generar una nueva ventana, una vez que se tiene creado un display, se utiliza la función:

CreateSimpleWindow

Como parámetros se le pasa el display creado, la ventana "padre" que en este caso será la root window, es decir la del sistema, el largo y ancho en pixels y el estilo y color de fondo.

GContext

Genera un contexto gráfico referido a una ventana lo que nos permite, por ejemplo para el caso del software de calibración, desplegar imágenes o dibujos en pantalla.

Para definir un GC se utiliza la funcion:

XCreateGC

Entonces manejando los conceptos mencionados anteriormente y las funciones nombradas se realizan las acciones de simulación de eventos de mouse.

Instalación de las librerías en la XO

Dado que por defecto las librerías utilizadas no están incluidas en la XO, se realizaron una serie de instalaciones para poder compilar y correr los programas en la misma.

Comandos ejecutados en la XO (desde consola)

- yum install make *(para ejecutar archivos makefile)*
- yum install gcc *(para poder compilar)*
- yum install libusb-dev *(necesario para el funcionamiento de la libusb)*
- yum install libX11-devel *(necesario para el funcionamiento de la xlib)*

Luego de instalar las mencionadas librerías se supone que debía funcionar todo a la perfección. Sin embargo al querer correr el programa de prueba 'movermiceabs.c' aparecía un error: "Xlib: extensión "XTEST" missing on display: 0.0". La xtst es la parte que se encarga de hacer los clicks, y posiblemente (aún en desarrollo) se encargará de los movimientos del mouse.

Se realiza una larga búsqueda en internet sobre las posibles razones de dicho problema. Las causas probables iban desde que podían faltar los headers o problemas de configuración, o simplemente problemas de compatibilidad con Sugar.

Finalmente se obtuvo[75] que el problema era que la extensión XTEST viene deshabilitada por defecto en la XO (se podía ver que la extensión no estaba ejecutando el comando 'xdpyinfo' en consola).

Entonces se procede a editar el archivo de configuración 'xorg.conf' (/etc/X11/xorg.conf)

Edición realizada:

Se sustituye la linea:

```
Option "XTEST" "Disable" # Mostly a debugging tool
```

Por:

```
Option "XTEST" "Enable" # Mostly a debugging tool
```

Esto tal como se indica en la página de referencia[75]. Luego de la edición anterior y, previo reinicio de la máquina, queda funcionando correctamente la librería xlib con las extensiones que necesitamos.

Script de ejecución

Dado que el driver debe ser ejecutado desde consola, se generan conflictos con el XServer el cual require de un display para funcionar. Para que funcione correctamente, se debe "exportar" las variables de display. Para ello se genera un script el cual será utilizado cada vez que se desee ejecutar el driver.

Escritura del Script

1. Utilizando un editor de texto generamos un archivo con el siguiente contenido:

```
#!/bin/bash
echo Ejecutando export display....
export DISPLAY=:0.0
echo inicia LAPIX
./lapix
```

   Guardo el archivo con el nombre: "lapix.sh". Es importante poner la extensión ".sh"

2. Copio el script creado a la carpeta /usr/bin. Esto me permite poder ejecutarlo sin importar el directorio donde me encuentre.

3. Le doy permisos de ejecución:

```
chmod +x lapix.sh
```

Luego, para ejecutar el Driver se utilizará este script. [76]

3.4.7. Plataformas soportadas por el Driver

Desde el inicio del proyecto se tuvo como consigna que el dispositivo funcionase no sólo en la XO sino que pudiese ser portable a otras plataformas. Es por ello que a la hora de seleccionar las librerías y métodos a utilizar se consideró este requerimiento como factor de suma importancia.

En primer lugar, se utiliza C como lenguaje de programación, dado que el mismo está ampliamente difundido y es soportado por una gran variedad de sistemas. Para el desarrollo del driver fue también necesario la utilización de 2 librerías fundamentales, una para el manejo de las comunicaciones USB y otra para lograr desplegar los eventos en pantalla. Estas librerías son la libusb y la xlib respectivamente. (Para más información de las mismas refererirse a las secciones correspondientes 3.4.1 y 3.4.6).

Antes de utilizar dichas librerías se buscaron indicios de que las mismas fuesen portables a otras plataformas, dado que su sistema nativo es linux.

Para el caso de la Xlib, parte del XServer, en la página oficial del proyecto [74] se puede acceder a "proyectos relacionados", dentro de los mismo

está el proyecto Cygwin/X [77].

Cygwin/X es el puerto de salida de "X Window System" o XServer a los sistemas que operan con Microsoft Windows. Proporciona un entorno y las librerías necesarias para correr aplicaciones del XServer en Windows. Cabe destacar que también existen otros proyectos similares como Xming o WeirdX [78]

En la página oficial de CygwinX ([77]) se encuentran las instrucciones para su instalación. Al momento de instalarse, debe tenerse en cuenta que se debe seleccionar la opción adecuada para que descargue también el código fuente, el cual es necesario ya que se brindan los archivos necesarios para la compilación del código en Microsoft Windows.

Para el caso de la libusb ([63]), se utiliza el proyecto libusb-win32. Libusb-win32 es el puerto de salida de la libsub a los sistemas que operan con Microsoft Windows. De la página oficial de la libusb ([63]), encontramos ejemplos de proyectos realizados con libusb-win32 ([79]).

También se encuentra la página del proyecto ([80]) donde se pueden ver las instrucciones para su descarga. De igual manera que para el caso de Cygwin, deben descargarse los fuentes para poder realizar la compilación del código en Windows.

Compilación en Windows

Para poder recompilar los códigos realizados para linux en Windows se debió utilizar un compilador C. El seleccionado fue el Dev-C++.

Para lograr una correcta compilación, luego de haber descargado los fuentes de Cygwin y libusb-win32, se le debe indicar a Dev-C++ en qué carpetas debe buscar los fuentes de las funciones que se utilizan. De otro modo no se logrará compilar el código dado que no encontrará las definiciones de las funciones y estructuras utilizadas, básicamente los archivos ".h".

Estado Actual

Debido básicamente a la escasez de tiempo no se logró tener funcionando completamente el driver en Windows. Se logró compilar exitosamente el

código y se logró desplegar la ventana principal del driver. Lo que resta que funcione es la parte correspondiente a la libusb la cual aparentemente no está leyendo los datos.

En principio se cree que este problema se resuelve creando un archivo .info (para Windows) que es el encargado de asociar a nuestro dispositivo el driver realizado.

Capítulo 4

Características Obtenidas

4.1. Introducción

En esta sección se exponen las características que presenta el prototipo en cuanto a resolución, exactitud, retraso, área de cobertura e interferencia entre emisores, a través de datos relevados y pruebas de funcionalidad.

En cuanto al muestreo buscado, dado que el microcontrolador funciona a 80MHz, frecuencia mayor que la necesaria, y como se transmite a una frecuencia impuesta, mayor a los 75Hz buscados, se está en condiciones de dar por válida esta característica.

4.2. Resolución

Para poder evaluar la resolución de Lapix, se realizó una prueba en la cual se tomaron muestras en 88 puntos en una cuadrícula. Luego se analizaron los mismo evaluando la dispersion de las muestras.

4.2.1. Descripción de la prueba

Se tomó una hoja de tamaño A4 (21cm x 29.7cm) y se dividió la misma en líneas horizontales y verticales de manera de lograr 88 puntos con 8 columnas y 11 filas. Las 8 columnas se separaron a 3cm y las 11 filas se separaron a aproximadamente 2.97cm (error de 0.3mm).

Se fija la hoja con cinta adhesiva a la mesa, separada 5mm del receptor el cual está también fijado a la mesa del mismo modo. Se coloca un osciloscopio

para observar salidas representativas en el receptor de manera de comprobar que todo funcione correctamente durante la prueba.

Luego, mediante un dispositivo sujetador (de manera de garantizar que el lápiz esté estático durante el muestreo), se sostiene al lápiz sobre cada punto de la cuadrícula. La punta se encuentra a 5mm de altura desde la hoja.

Finalmente utilizado la versión "6.11_para_muestras" del driver se realiza el muestreo, tomando entre 18 y 20 segundos por muestra lo que da un total de aproximadamente 1300 muestras por punto. Los datos que se almacenan son L.x y L.y antes y después del filtrado, es decir los puntos de corte de las circunferencias medidos en períodos de reloj de 80MHz.

Las muestras de cada punto se guardan con el nombre "puntoij" donde i es la fila y j la columna que corresponde a la grilla.

En las imágenes 4.1 y 4.2 se puede apreciar la configuración para la prueba.

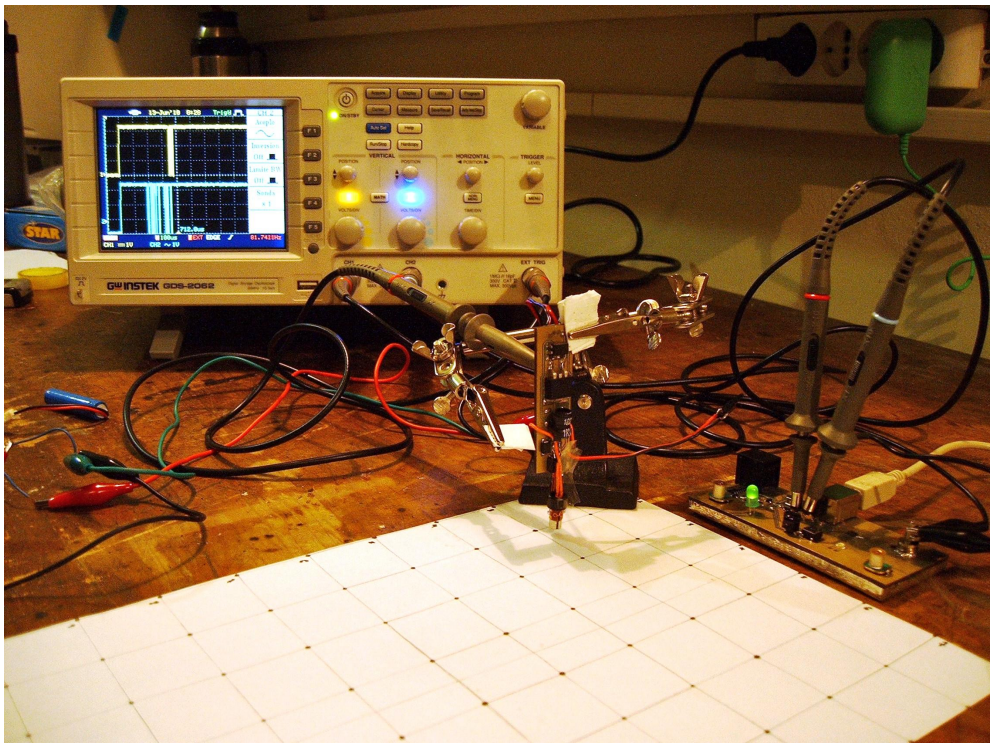

Figura 4.1: Descripción de la prueba

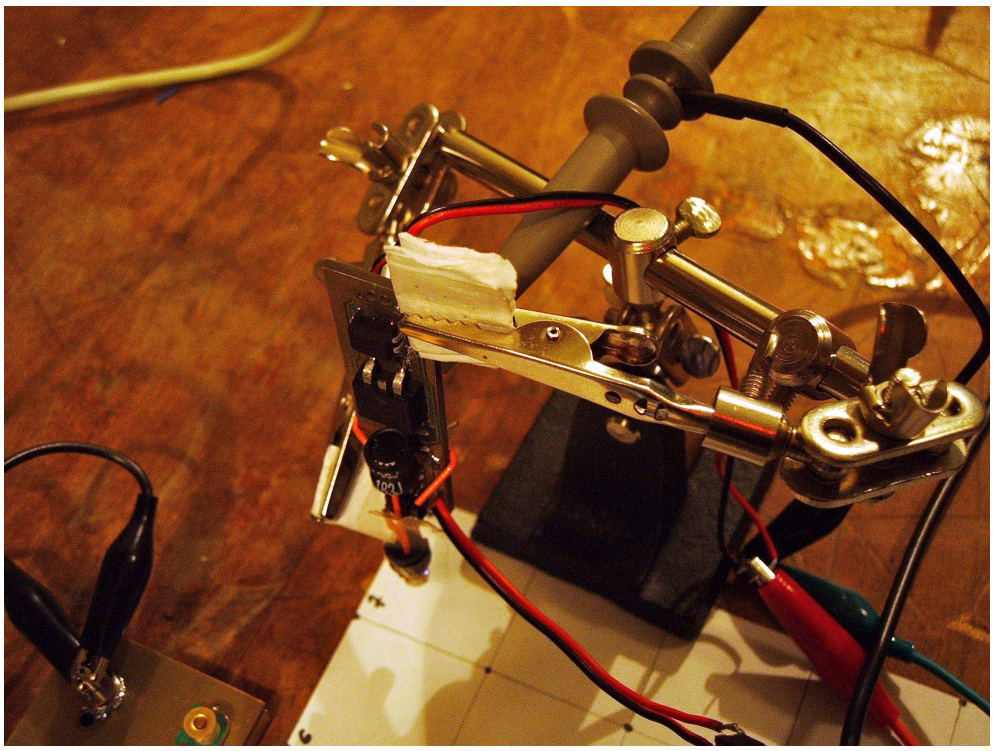

Figura 4.2: Descripción de la prueba - Dispositivo de fijación

4.2.2. Análisis y Gráficas de los datos obtenidos

Para el análisis de los datos se realiza un programa en Matlab el cual se encarga de graficar la grilla muestreada, y de calcular la varianza media por fila y la media total.

Al momento de graficar la grilla notamos que existía un ruido no deseado en las muestras de la primera fila y en la muestra "punto118" correspondiente al último punto muestreado.

El ruido en la primera fila se debe a que en el momento del muestreo el plano de escritura no se colocó a la misma altura que el plano de los sensores, por lo que al estar tan cerca del receptor, la misma placa actuó de interferente.

Para el cálculo de la varianza promedio no se tendrán en cuenta estos puntos por entenderse que si son incluidos alterarían el resultado arrojando un valor que no es representativo del conjunto. Sin embargo si se mostrará la varianza correspondente a toda la grilla.

Grilla y varianza por Filas

En la imagen 4.3 se puede apreciar la grilla obtenida. El receptor está ubicado sobre la linea 0, es decir en la parte superior de la figura. Para que las gráficas queden con la orientación correcta se grafican los opuestos en el eje Y.

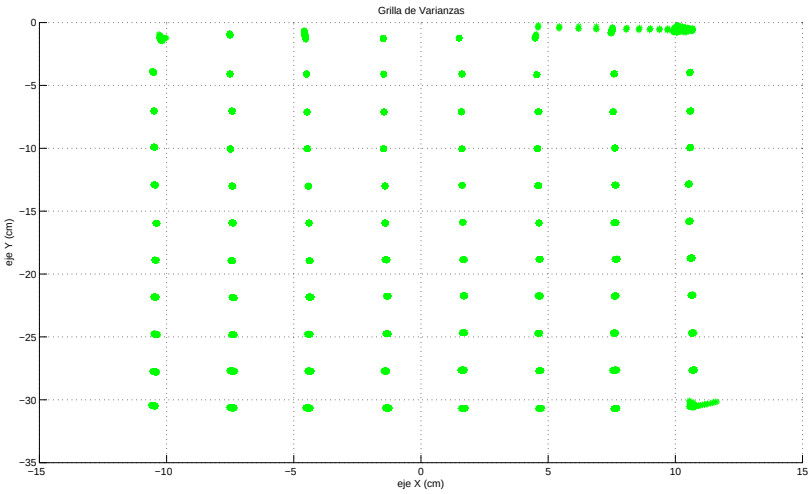

Figura 4.3: Grilla obtenida

Como se puede apreciar (y como es de esperarse), los puntos se van "engrosando" a medida que nos alejamos del receptor, es decir, va aumentando la varianza.

En la figura 4.4 se aprecia la grilla sacando la primera y última fila.

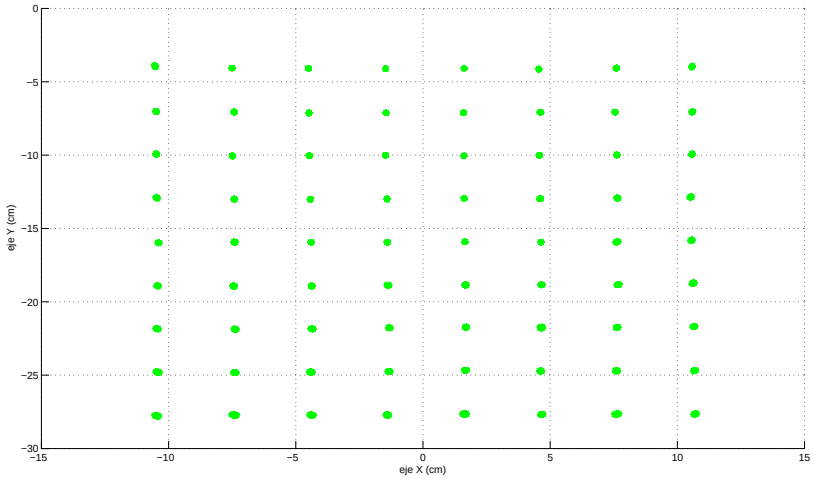

Figura 4.4: Grilla obtenida

A continuación, en la tabla 4.2.2 se muestran las varianzas por filas, donde el sufijo **sf** significa sin filtrado y **cf** con filtrado.

Como se comentaba anteriormente, la varianza en los puntos de la primera y última fila se ve afectada por ruido, por lo que no serán tenidos en cuenta para la determinación de la resolución.

Varianza media

Se halla la media de las varianzas en los ejes X e Y de lo que se obtiene la tabla 4.2.2.

Dado lo anterior, y tomando como varianza las medias filtradas, podemos decir que el dispositivo cuenta con una resolución de:

$$\boxed{Res = 0{,}1868mm}$$

Esta resolución corresponde con aproximadamente 136dpi. Cabe destacar que este valor se halla en un 90 % del área de la A4.

Si se incluyen las filas no tomadas en cuenta, la resolución daría 0.7623mm que serían aproximadamente 33dpi. Sin embargo, quitando sólo la primer fila, se tiene una resolución de 0.2347mm que serían 108 dpi.

	varXsf(cm^2)	varYsf(cm^2)	varXcf(cm^2)	varYcf(cm^2)
fila 1	$4{,}20 \times 10^{-4}$	$1{,}55 \times 10^{-2}$	$5{,}24 \times 10^{-2}$	$6{,}00 \times 10^{-3}$
fila 2	$8{,}96 \times 10^{-5}$	$1{,}82 \times 10^{-4}$	$5{,}65 \times 10^{-5}$	$1{,}20 \times 10^{-4}$
fila 3	$1{,}37 \times 10^{-4}$	$1{,}05 \times 10^{-4}$	$9{,}80 \times 10^{-5}$	$6{,}97 \times 10^{-5}$
fila 4	$1{,}30 \times 10^{-4}$	$8{,}27 \times 10^{-5}$	$7{,}14 \times 10^{-5}$	$4{,}98 \times 10^{-5}$
fila 5	$2{,}21 \times 10^{-4}$	$1{,}01 \times 10^{-4}$	$1{,}26 \times 10^{-4}$	$6{,}31 \times 10^{-5}$
fila 6	$3{,}20 \times 10^{-4}$	$1{,}00 \times 10^{-4}$	$1{,}68 \times 10^{-4}$	$5{,}93 \times 10^{-5}$
fila 7	$5{,}23 \times 10^{-4}$	$1{,}15 \times 10^{-4}$	$2{,}60 \times 10^{-4}$	$7{,}02 \times 10^{-5}$
fila 8	$8{,}10 \times 10^{-4}$	$1{,}60 \times 10^{-4}$	$3{,}30 \times 10^{-4}$	$1{,}00 \times 10^{-4}$
fila 9	$1{,}28 \times 10^{-3}$	$1{,}50 \times 10^{-4}$	$4{,}67 \times 10^{-4}$	$7{,}38 \times 10^{-5}$
fila 10	$2{,}33 \times 10^{-3}$	$2{,}30 \times 10^{-4}$	$8{,}32 \times 10^{-4}$	$1{,}25 \times 10^{-4}$
fila 11	$7{,}06 \times 10^{-3}$	$1{,}24 \times 10^{-3}$	$1{,}86 \times 10^{-3}$	$5{,}09 \times 10^{-4}$

Cuadro 4.1: Varianzas obtenidas de las muestras tomadas

Varianza media ejeX sf(cm^2)	6.49×10^{-004}
Varianza media ejeX cf(cm^2)	2.67×10^{-004}
Varianza media ejeY sf(cm^2)	1.36×10^{-004}
Varianza media ejeY cf(cm^2)	8.13×10^{-005}

Cuadro 4.2: Varianzas obtenidas de las muestras tomadas

Estos valores de resolución tienen en cuenta toda el área, en la parte central de la hoja se tienen mejores resoluciones que llegan a valores de 0.0339mm, o sea, 750dpi. En tanto que los peores puntos relevados llegan a tan solo 5.96mm, o sea, 4dpi.

Es de interés mencionar que si bien la primera fila muestra una muy baja resolución, a pesar de que en la prueba no se refleje por la amplia separación en distancia entre las muestras, esta fila corresponde con una franja muy delgada, menor a 5mm y que principalmente empeora su resolución sobre las esquinas de la hoja.

4.3. Interferencia entre emisores

Claramente en caso de utilizar dos emisores con un mismo receptor dentro del área de cobertura, la señal recibida se vuelve un caos debido a la interferencia mutua que se generan y logrando en el mejor de los casos que el puntero en patalla oscile entre la posición de cada uno de los emisores.

Si consideramos ahora una situación en la que un emisor se encuentra dentro del área de cobertura de un receptor, y otro emisor al que consideramos como interferente se encuentra fuera de ésta, dependiendo de la potencia de transmisión que tenga el emisor interferente será la distancia mínima a la que puede encontrarse del receptor para que no genere ningún tipo de ruido.

Por lo pronto, se realiza una prueba que consiste en alejar un emisor desde el área de cobertura, hasta que no pueda detectarse ni infrarrojo ni ultrasonido barriendo varios ángulos y midiendo la distancia máxima a la que todavía se detecta señal. Cabe destacar que la señal de ultrasonido luego de los 50cm de distancia es escasamente detectada dependiendo fuertemente de los rebotes y obstaculos presentes, mientras que la señal de infrarrojo presenta distancias mucho mayores y se aprecia un efecto en el que una vez que se detecta señal, es posible alejarse bastante, pero una vez que se pierde la señal, no se vuelve a recibir hasta que se acerque una distancia considerable.

Este efecto puede explicarse debido a la sincronización del receptor con la frecuencia de la portadora, la cual requiere un cierto umbral de potencia de recepción para iniciarse pero puede sostenerse a pesar de que la señal se atenúe. Con este efecto lo que se consigue es que se requiere grandes distancias para dejar de recibir, pero distancias cercanas a los 50cm para que se comience a detectar señal. A modo de asegurarse de no tener ningún tipo de interferencia es que se toma el peor caso en que el emisor se aleja para realizar la prueba.

A continuación se muestran las distancias alcanzadas antes de extinguirse la señal para varios ángulos respecto al eje perpendicular a la línea que une los sensores del receptor:

ángulo 0º	3.80m
ángulo 35º	2.00m
ángulo 50º	1.55m
ángulo 90º	0.90m

Cuadro 4.3: Distancias alcanzadas con interferencia en función de los ángulos.

Una vez que el receptor se encuentra encapsulado, al superar los 90º no es posible detectar señal. Como el receptor es simétrico, lo mismo ocurre para ángulos negativos.

De esto se concluye que el área de interferencia es mucho mayor que la

buscada de 50cm de radio respecto al receptor. Sin embargo, se está considerando un peor caso en un espacio libre. Por lo que en un uso normal, dentro de un ambiente con obstaculos, la señal es difícilmente detectada en estas circunstancias y pueden llegar a ser usados hasta cerca de unos 70cm de distancia sin mayores problemas. Principalmente, teniendo en cuenta la directividad que presenta el patrón de recepción y que en donde se recibe con mayor potencia es donde se encuentra el usuario que oficia de obstáculo.

4.4. Exactitud y Retraso

Para poder determinar el retardo que hay entre que se realiza un trazado hasta que es desplegado en pantalla se realiza una prueba que consiste en filmar el trazado de varios tipos de curvas y rectas hechas sobre la pantalla de la XO a distintas velocidades.

Luego, estos videos fueron pasados cuadro a cuadro en el programa Windows Movie Maker y pudimos apreciar que el retardo máximo presente es menor a 80ms, correspondiente con el pasaje entre cada cuadro, tiempo más que aceptable a nuestro entender. Estos videos se encuentran adjuntos en el CD del proyecto en CD/pruebas/prueba_de_retardo/*.

De todos modos, el objetivo de esta característica es poder de alguna forma cuantificar la sensación de simultaneidad que se experimenta al escribir sobre la pantalla. Debido a esto, esta medida presenta gran subjetividad, por lo que se recomienda realizar una prueba con varios indivuos los cuales califiquen, qué tan conformes se sienten en cuanto a la fluidez desplegada. A nuestro parecer, se cuenta con una muy buena fluidez.

Con respecto a la exactitud, ésta depende fuertemente de la calidad de la calibración que se realice y de qué tan firme se encuentre el receptor sobre la XO de forma de que los trazos realizados con el lápiz se sigan fielmente en el despliegue en pantalla. Por esto es que se considera que la exactitud en la detección no es una medida cuantificable.

Capítulo 5

Estudio de Consumo

5.1. Consumo teórico

5.1.1. Emisor

A continuación se describe el consumo teórico instantáneo máximo, en forma aproximada, de la placa emisora. Con este valor podemos saber cuánta corriente máxima de pico tiene que ser capaz de entregar la fuente de alimentación del lápiz. Para que este máximo ocurra se debe dar que tanto el botón primario como el secundario estén presionados.

Corrientes tomadas de pico máximo en peor caso:

- Corriente por resistencia de pulsadores: $3{,}3V/10k\Omega = 330\mu A$. Por ser dos pulsadores la corriente es $660\mu A$.
- Corriente consumida por el regulador de voltaje según su hoja de datos [20]: $150\mu A$.
- Corriente por colector del optoacoplador:
  Asumiendo que el transistor MOS no toma corriente tenemos[1]:

$$\frac{5V - V_{CE(SAT)}}{1k\Omega} = 4{,}8mA$$

  con $V_{CE(SAT)}$ máximo de $0{,}2V$ según hoja de datos [19] del optoacoplador.

[1] En la sección 2.1.2 se indica que de simulaciones se obtienen picos de 100mA. Sin embargo, al incorporar el optoacoplador la corriente que toma para tener el correcto funcionamiento se reduce a menos de 4mA, con una duración despreciable y se producen antes y después del pico generado en el diodo.

- Circuito emisor de ultrasonido según simulación en Multisim: $67{,}6mA$.
- Circuito emisor infrarrojo:

$$\frac{3{,}3V - 2 \times 1{,}2V - 0{,}3V}{30\Omega} = 20{,}0mA$$

- PIC12:
  - corriente de operación según hoja de datos [17]: $760\mu A$.
  - corriente salida infrarrojo:

$$\frac{3{,}3V - V_{BE}}{10k\Omega} = 260\mu A.$$

  - corriente salida de ultrasonido:

$$\frac{3{,}3V - V_F}{470\Omega} = 4{,}5mA$$

    con V_F típico de $1{,}2V$ según hoja de datos [19] del optoacoplador.

De lo anterior se desprende que la corriente instantánea consumida máxima en el peor caso es $\mathbf{98{,}73}\boldsymbol{mA}$.

Ahora se presenta el consumo promedio en un período $T = 1/75s$ (exactamente $13073\mu s$). Al igual que antes, se toma el peor caso en el que ambos botones están presionados. Para el circuito emisor de ultrasonido, la salida del PIC12 de ultrasonido y el optoacoplador, se halla la corriente media integrándola en un período y dividiendo entre el largo de éste (el pulso de ultrasonido emitido tiene una duración de $300\mu s$). Para el circuito emisor de infrarrojo y la salida del PIC12 correspondiente, primero se halla la corriente media en el subperíodo ($T' = 1/36kHz = 28\mu s$) correspondiente a la frecuencia de $36kHz$ a la que opera. Luego, tomando el peor caso en que se manda el tren de pulsos mayor correspondiente a los dos botones presionados (de largo $3976\mu s$, que sale de la suma de los 3 trenes de pulsos $(30+30+82) \times 28\mu s$), se halla la corriente media del período de operación usando la corriente media del subperíodo.

- Corriente por resistencia de pulsadores: $3{,}3V/10k\Omega = 330\mu A$. Por ser dos pulsadores la corriente es $660\mu A$.
- Corriente consumida por el regulador de voltaje según su hoja de datos [20]: $150\mu A$.
- Corriente por colector del optoacoplador: $4{,}8mA \times \frac{300\mu s}{13073\mu s} = 110\mu A$.

- Circuito emisor de ultrasonido:

$$\frac{67{,}6mA \times 300\mu s}{2 \times 13073\mu s} = 776\mu A$$

- Circuito emisor infrarrojo:
  - en T' : $20mA/2 = 10mA$.
  - en T : $10mA \times \frac{3976\mu s}{13073\mu s} = 3{,}0mA$.
- PIC12:
  - corriente de operación según hoja de datos [17]: $760\mu A$.
  - corriente salida infrarrojo:
    - en T' : $260\mu A/2 = 130\mu A$.
    - en T : $130\mu A \times \frac{3976\mu s}{13073\mu s} = 39{,}5\mu A$.
  - corriente salida de ultrasonido: $4{,}5mA \times \frac{300\mu s}{13073\mu s} = 103\mu A$.

Entonces, la corriente promedio consumida en el peor caso es de $\mathbf{5598\mu A}$.

En un uso normal se puede llegar a considerar que el tiempo en que los botones se mantienen presionados es despreciable.

Si no se presiona ninguno de los botones, el largo del tren de pulsos de infrarrojo es: $(30 + 30 + 39) \times 28\mu s = 2772\mu s$.

Entonces, el circuito emisor de infrarrojo en T consume: $10mA \times \frac{2772\mu s}{13073\mu s} = 2{,}1mA$.
Y la salida del PIC12 correspondiente al infrarrojo en T queda: $130\mu A \times \frac{2772\mu s}{13073\mu s} = 27{,}6\mu A$.

De esta forma, la corriente promedio consumida normalmente por el emisor sin usar los botones es de $\mathbf{4026\mu A}$.

Según la hoja de datos [39] del DC/DC step-up, para estos valores de corriente de salida tiene una eficiencia mayor al 80 %. Dado que la eficiencia se define como el cociente entre la potencia de salida y la potencia de entrada, el voltaje de salida es de 5.18V y el voltaje de entrada depende del estado de la batería pero se puede aproximar a un promedio de 3.7V, tenemos que para el caso en que se presionan ambos botones constantemente y considerando

una eficiencia del 80 %, el consumo de la batería utilizada es de $\mathbf{9796\mu A}$.

Análogamente para el caso de uso normal, el consumo de la batería es de $\mathbf{7045\mu A}$.

Como la batería utilizada tiene una capacidad de 300mAh, la duración teórica de esta batería es de aproximadamente **30 horas y media** en el peor caso.

Finalmente, para un uso normal se obtiene que la batería tiene una duración teórica de aproximadamente **42 horas y media**.

5.1.2. Receptor

Para obtener el consumo teórico del receptor, primero obtenemos el consumo de los amplificadores operacionales OPA2134 mediante simulaciones, dado que en su hoja de datos [47] sólo se dan valores de consumo para voltajes de alimentación mayores a $\pm 15V$.

Independientemente de la señal de entrada, el consumo de uno de estos integrados que implementan las dos etapas amplificadoras de uno de los circuitos receptores de ultrasonido es de $1{,}65mA$. Entonces el consumo de estos dos integrados es de $3{,}3mA$.

El bloque generador del umbral de comparación, tiene un consumo de:

$$\frac{5V}{10k\Omega + 10k\Omega + 200\Omega} = 248\mu A$$

En tanto que el consumo del LM339 según su hoja de datos [48], es de $600\mu A$ para todo el integrado a 25^oC.

El consumo de los condensadores de desacople es despreciable y el de los "pull-up" (resistencias de salida) a la salida de los comparadores también se puede considerar despreciar ya que consumen sólo cuando se recibe ultrasonido y es un tiempo despreciable en el período.

La corriente consumida por cada circuito inversor, cuando la salida está en cero son $330\mu A$ $(3{,}3V/10k\Omega)$. Por lo que el total consumido por los inversores es de $990\mu A$ en el peor caso en que todas las salidas estén en cero.

En cuanto al receptor de infrarrojo, de la hoja de datos [50] del TSOP4836 se extrae que la corriente máxima que consume son $1{,}5mA$ y típicamente $1{,}2mA$.

El consumo de la compuerta AND es del orden de los nA, por lo que lo consideramos despreciable en comparación con el consumo de los elementos antes mencionados.

En resumen, tenemos un consumo considerando el peor caso de la placa receptora, sin tomar en cuenta el bloque del microcontrolador, de $\mathbf{6{,}64mA}$.

En tanto que para el bloque del microcontrolador, de la hoja de datos [51] del PIC32, tenemos que la corriente máxima de operación a la frecuencia de trabajo de $80MHz$ es de $75mA$ ($55mA$ típica a $3{,}3V$ y $25°C$), y su corriente máxima de reposo es de $32mA$ ($24mA$ típica a $3{,}3V$ y $25°C$). El consumo del pulsador de reset puede despreciarse y el del LED es de $frac3{,}3V - V_D470\Omega = 3{,}19mA$, donde V_D es la caída de voltaje en el diodo y tiene un valor de $1{,}8V$.

Como no tiene que entregar corriente a ningún dispositivo, ya que sólo tiene entradas que no lo cargan, el consumo máximo del bloque es de $\mathbf{78{,}19mA}$.

De esta forma, la corriente que tiene que entregar el regulador de voltaje es de $79{,}18mA$ (la suma del bloque del microcontrolador con los inversores), menor a los $100mA$ que es capaz de entregar. Según su hoja de datos [20], para este valor de corriente de carga, tiene un consumo de $5mA$.

Sólo nos resta ver el cargador de la batería, el cual según su hoja de datos [42] tiene un consumo máximo de $3mA$ ($1{,}65mA$ de consumo típico alimentado por USB) y el LED se encuentra apagado ya que suponemos no estamos cargando la batería.

Finalmente, llegamos a que el consumo total máximo de la placa receptora, mientras no se está cargando la batería, es de $\mathbf{92{,}83mA}$.

El puerto USB brinda una corriente máxima de $100mA$ cuando un dispositivo es conectado, luego del proceso de enumeración en el cual el host recibe los descriptores del dispositivo, éste puede pedir aumentar ésta corriente hasta un límite máximo de $500mA$. Concluimos entonces que la corriente que nos brinda un solo puerto USB es suficiente para alimentar la placa receptora mientras no se este cargando la batería.

Si no se utiliza el solder-pad del cargador, la corriente de carga máxima es limitada a $100mA$, por lo que es necesario pedirle al puerto USB que brinde al menos $200mA$.

Si se utiliza el solder-pad del cargador, esto significa que el límite de carga máximo de la corriente se incrementa a $340mA$. Por lo que la placa podría llegar a requerir $440mA$ y es necesario modificar un parámetro del código del PIC32 para pedirle al puerto USB que brinde su capacidad máxima de $500mA$.

5.2. Medidas de consumo

5.2.1. Emisor

Experimentalmente se realizaron medidas[2] del consumo de corriente continua de la placa emisora mientras no se pulsaban los botones. La misma se alimenta utilizando el DC/DC step-up conectado a la batería seleccionada a través de su protección.

Consumo del emisor en la entrada del DC/DC step-up: **7,75mA**.
Consumo del emisor en la salida del DC/DC step-up: **4,6mA**.

Tomando las mismas consideraciones que al hallar el consumo teórico, se obtiene que la eficiencia del DC/DC step-up es del 83 %, del orden de la expuesta en su hoja de datos y representa un consumo por parte del mismo del 68 % respecto al del circuito emisor (3,15mA).

Dado que la batería seleccionada tiene una capacidad de 300mAh, la duración de la batería por carga es de casi **39 horas**.

Teóricamente habíamos calculado un valor de 4,0mA de consumo para la placa emisora lo que representa un error del 12 % respecto al medido.

Esta diferencia puede explicarse debido a que el valor tomado para el $V_{CE(SAT)}$ del transistor del circuito emisor de infrarrojo es demasiado conservador. Por estar en zona activa sabemos que V_{CE} es levemente mayor que $V_{CE(SAT)}$ y observando con mayor detenimiento su hoja de datos, obtenemos una gráfica en donde se ve que para el orden de corriente utilizado en el colector ($\approx 20mA$), $V_{CE(SAT)}$ es menor a 0,1V. De esta forma, recalculando la corriente instantánea obtenemos que es de unos 26,7mA. Con este nuevo valor, el consumo total del circuito aumenta a 4753μA. Valor incluso mayor al medido, por lo que una pequeña variación en el valor de la caída de voltaje en este transistor afecta significativamente el consumo del circuito. El resto de los factores que influyen en dicha diferencia, tales como el error en la medida, las dispersiones de las resistencias, las diferencias de calidades de los componentes utilizados respecto a los expuestos en sus hojas de datos, entre otros, tienen un peso menor que el del factor indicado.

[2]Todas las medidas realizadas de voltaje y corriente fueron hechas con un multímetro digital Microtek DT9208L [81].

Si medimos el consumo de la parte del circuito del emisor que es alimentada por el regulador tenemos que:
Consumo en la entrada del regulador: 3,8mA.
Consumo en la salida del regulador: 3,45mA.

De esto obtenemos que el consumo aproximado del regulador de voltaje es de 350μA. Más allá de la incertidumbre del instrumento de medida utilizado, la diferencia con el consumo teórico muestra que la calidad del integrado que adquirimos no es la misma a la que refiere su hoja de datos [20].
Lo anterior representa un 10 % del consumo de esta parte del circuito, un 7,6 % del consumo de la placa y apenas un 4,5 % del consumo total de la batería.

Ahora si vemos el consumo del resto del circuito, o sea, la parte del emisor que trabaja en 5V (la parte de 3,3V se alimenta directamente desde otra batería).
Consumo del emisor en la entrada del DC/DC step-up: 1,85mA.
Consumo del emisor en la salida del DC/DC step-up: 0,8mA.

Esto representa un consumo por parte del DC/DC step-up del 131 % respecto al de la parte del circuito que requiere 5V (1,05mA). Sin embargo, representa sólo un 23 % del consumo total de la placa, por lo que al alimentar con el DC/DC step-up el regulador de voltaje, se desperdicia un 45 % del consumo de la placa (2,1mA). Con cargas tan pequeñas su eficiencia decrece al 60 %, tal como indica su hoja de datos.

De esta forma, si ahora utilizamos el DC/DC step-up sólo para alimentar la parte del circuito que requiere 5V y al regulador lo alimentamos directamente con la batería, obtenemos un consumo de 5,7mA. Sin embargo, el valor de voltaje que da la batería se reduce con el uso, por lo que llegaría un punto en que el regulador no sería capaz de generar los 3,3V, desperdiciando la capacidad de la batería y reduciendo la cantidad de horas posibles de uso.

Si se optara por no usar el regulador de voltaje y alimentar directamente con la batería, el incremento de consumo del circuito al ser alimentado a mayor voltaje superaría el poco consumo del regulador lineal. Sólo el incremento en consumo del PIC12 equivale al consumo del regulador. Además, la potencia de emisión de infrarrojo, tal como esta diseñada, pasaría a depender de la carga de la batería.

Como trabajo a futuro se podría realizar un estudio más completo que

incluyera el uso de un regulador conmutado, los que suelen tener un consumo menor que el lineal utilizado. Además, ver la posibilidad de utilizar más de uno de forma que se ajusten al voltaje mínimo requerido por cada parte del circuito y evaluar otras posibles soluciones.

5.2.2. Receptor

El consumo de corriente medido[3] de toda la placa receptora en un funcionamiento normal, mientras no se carga la batería del emisor, es de **83,7mA**.

Si el receptor se conecta con el emisor apagado, el LED del microcontrolador no se enciende, por lo que el consumo decrece a 80,5mA.

Teóricamente se había obtenido un consumo máximo total del receptor de 92,83mA, lo que representa un 11 % más de consumo respecto al medido. Ésta es una cifra razonable dado que se trata de una cota superior para el consumo respecto a un uso normal.
Veamos el consumo de algunas de las partes del receptor para comprender mejor a que se debe esta diferencia.

Consumo en la entrada del regulador: 72,5mA.
Consumo en la salida del regulador: 69,5mA.

Entonces, el consumo del regulador es de 3mA. Según su hoja de datos [20], para 70mA tiene un consumo de 4mA. Esto muestra que los valores de la hoja de datos se ajustan mejor a la calidad de nuestro componente para consumos altos que para bajos, inclusive mejora su desempeño respecto del indicado.

En el análisis teórico se llega a que la corriente de salida del regulador es de 79,18mA, pero ese valor fue calculado considerando el consumo máximo del PIC32 de 75mA. Si tomamos el consumo típico de 55mA, llegamos a que su consumo es de 59,18mA, por lo que como el resto de los componentes tienen un peso mucho menor en el consumo, su influencia puede despreciarse y se concluye que el microcontrolador opera por encima de su consumo típico pero no alcanza el máximo.

Consumo del cargador de batería mientras permanece inactivo: 1,63mA.

Concuerda con los 1.65mA típicos que aparecen en su hoja de datos [42].

Restándole al consumo total del circuito, la corriente a la entrada del regulador junto con el consumo del cargador, llegamos a que los bloques de recepción de ultrasonido junto con el bloque de recepción de infrarrojo tienen

[3]Todas las medidas realizadas de voltaje y corriente fueron hechas con un multimetro digital Microtek DT9208L DT9208L [81].

un consumo de 9,57mA. Teóricamente teníamos un consumo de 5,65mA para esta parte del circuito que representa un 59 % del medido. Este error puede atribuirse principalmente al consumo de los amplificadores operacionales el cual fue hallado por medio de una simulación de la que se desconoce su precisión, dado que la hoja de datos no brinda datos para nuestro valor de alimentación utilizado. Se recomienda como trabajo a futuro, tomar medidas directas del consumo de estos integrados.

5.2.3. Duración de baterías

Para determinar el impacto del receptor en la duración de la batería de la XO, se mide el tiempo que demora en apagarse a partir de una carga máxima, con y sin el receptor conectado.

La prueba se inicia desconectando el cable de alimentación de la XO mientras se está utilizando el receptor con el puntero ubicado en el Hogar.

La duración de la batería con el receptor conectado es de **2 horas y 44 minutos**.

Luego, se repite el mismo procedimiento pero con el receptor desconectado de la XO y se ubica el puntero en el Hogar. Además cada cierto tiempo periódico se toca levemente el Touchpad de forma que no entre en modo de ahorro de energía.

La duración de la batería sin el receptor conectado es de **3 horas y 7 minutos**.

Se concluye que el receptor reduce la duración de la batería en **23 minutos**.

No se recomienda la carga de la batería del emisor utilizando la batería de la XO. Como trabajo a futuro se podría medir la duración de la batería de la XO mientras se carga el emisor limitando la corriente a $100mA$ y a $340mA$.

En cuanto a la duración de la batería del emisor, se realiza una prueba que consiste en encender el emisor luego de tener la batería cargada, con un voltaje medido en la salida de la protección de $4{,}14V$ ($4{,}18V$ es el máximo que puede llegar y supera la indicación de carga completa del receptor que corresponde con $4{,}12V$) y medir el tiempo hasta que la protección corta la salida de corriente al llegar a $2{,}50V$ y se deja de emitir señales. Cabe aclarar que la batería no es nueva y ya contaba con varios ciclos de uso antes de realizar la prueba.

La duración de la batería del receptor es de **42 horas**.

De esto se deduce que la capacidad de la batería utilizada para la prueba es de 326mAh.

En la figura 5.1 se observa una gráfica generada con valores de voltaje adquiridos cada una hora y en algunos casos menos, en las terminales de salida de la protección de la batería durante la prueba anterior. En ella se aprecia la evolución del voltaje que impone a lo largo del tiempo.

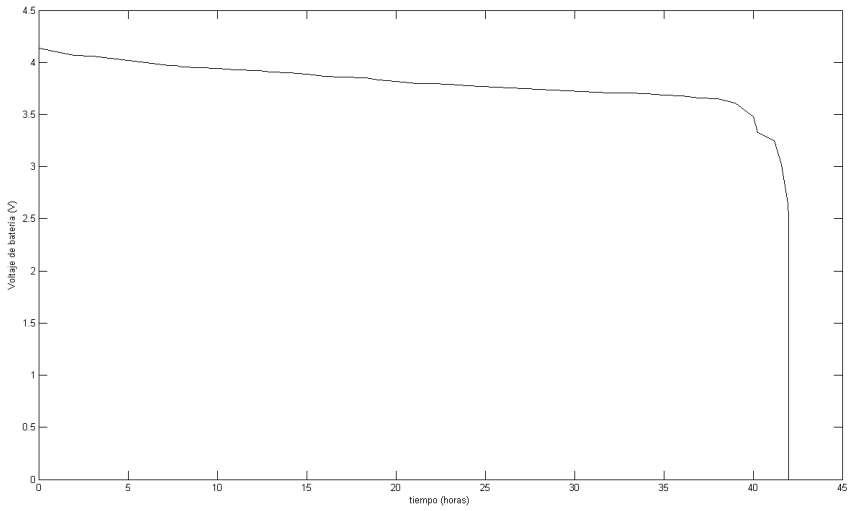

Figura 5.1: Evolución en el tiempo del voltaje impuesto por la batería del emisor.

En cuanto a la duración de la carga de la batería, se realiza una prueba utilizando el cargador del receptor sin utilizar el "solder-pad", por lo que la corriente es limitada a 100mA y el tiempo de carga, correspondiente al momento en que se conecta la batería al receptor hasta que se apaga la luz indicadora de carga completa que corresponde con una corriente menor a 50mA, es de **3 horas y 54 minutos**.

Capítulo 6

Problemas enfrentados

6.1. Infrarrojo

El principal problema que se tuvo en el proyecto y causante de que no se tenga la resolución buscada fue con la detección de infrarrojo. El módulo TSOP4836 [50] presenta un jitter en el pulso de salida digital. Esto quiere decir que el retraso en la generación del pulso de salida, por el cual se comienza a contar hasta la llegada del ultrasonido, es variable en el tiempo.

Estas variaciones son provocadas por el AGC que controla la ganancia del filtro pasabanda. Para estabilizarlo es necesario antes fijar una referencia de luz de forma que no inicie en un estado aleatorio a la llegada del pulso de detección. Para eso es necesario "encenderlo" con un pulso previo.

Experimentalmente, el jitter más pequeño que pudo obtenerse utilizando el lápiz fue de unos $500ns$ promedio, con algunas variaciones esporádicas que llegan hasta $750ns$. Para esto, se emiten 2 ráfagas de 30 pulsos en alto y 30 pulsos de espacio (o tiempo muerto), antes de enviar la ráfaga para iniciar el conteo de la señal.

Si se emite un tren de ráfagas de 22 pulsos en alto y 22 pulsos de espacio, o en otras palabras, si se emite una ráfaga de pulsos durante $600\mu s$ y se esperan otros $600\mu s$ antes de repetir la siguiente ráfaga de $600\mu s$ y así sucesivamente, el jitter se reduce a decenas de ns. Sin embargo, cualquier variación en el largo de los pulsos o los espacios lo incrementan. Esta emisión corresponde con una señal de prueba óptica que se muestra en la hoja de datos [50] del módulo receptor.

Otro factor que influye en el jitter es la estabilidad de la fuente de alimentación. Utilizando la configuración recomendada en la hoja de datos del módulo receptor, al alimentar con el puerto USB de algunas PCs, se puede apreciar un incremento significativo del jitter a unos $3\mu s$ e incluso más. Para reducir este efecto es que se utiliza un condensador de alto valor en la entrada de alimentación junto con otro cerámico para que filtre altas frecuencias.

Además del jitter, el módulo receptor presenta otro tipo de comportamiento no deseable. A partir de cierta distancia para cierta intensidad de recepción de señal, se generan saltos en la duración del retraso de la señal de salida de varios microsegundos.

La potencia con que se emite la señal afecta considerablemente a este efecto, y se requieren más de $20mA$ de corriente por los LEDs de la punta emisora [4] (valor típico de operación) para que los saltos aparezcan por fuera del área de cobertura. Cambios en el largo de las ráfagas y los espacios también afectan la distancia a la que puede estar el emisor sin que se vea este efecto. Para el código de prueba que da la hoja de datos, con una corriente por los LEDs emisores de $20mA$ se logran distancias menores a $20cm$.

Lamentablemente para evitar que los saltos aparezcan dentro del área de cobertura, con esta potencia de transmisión se alcanzan distancias grandes en la que se sigue detectando señal y es por esto que la interferencia entre lápices se da a distancias mayores a 1m respecto del receptor.

El retraso en los flancos de bajada de la señal de salida también presenta jitter y saltos en la duración, por lo que se requieren ráfagas superiores a los 10 pulsos, como indica su hoja de datos, para la codificación de la información del pulsado de los botones.

6.2. Ultrasonido

El principal problema enfrentado con las señales de ultrasonido es la emisión electromagnética generada en el RLC del emisor. Esta emisión se infiltra en el amplificador de recepción, lo que provoca desde un par de pulsos pequeños a la salida del amplificador al tener el emisor a menos de 1cm del receptor, como sucede con el prototipo en SMD, hasta que tape las señales completamente en caso de tener un diseño que no tome en cuenta esta interferencia y no esté blindado.

En el caso del primer prototipo, a pesar de estar blindado, se detectan hasta 5 pulsos de interferencia electromagnética en la salida del amplificador que superan el umbral de recepción, al acercar el emisor al receptor. Para solucionar esto, se retrasa la emisión del infrarrojo respecto a la de ultrasonido, de forma que el pulso de detección se genere luego del último pulso de interferencia. O de forma equivalente, adelantar la señal de ultrasonido.

Esta solución tiene la desventaja que cuánto más se retrase el pulso, mayor será la distancia mínima del emisor respecto del receptor, a la que se detecta señal. Otro paliativo es blindar el emisor.

Para el caso del receptor en SMD, blindando sólo la capa de abajo, apenas llegan a detectarse dos pequeños pulsos que aparecen a distancias menores a 1 cm, del emisor respecto al receptor, y con un pequeño retraso de unos $20\mu s$ es suficiente.

Otro problema es que para lograr valores de amplitud del primer pulso positivo detectado en recepción que se encuentren por encima del valor máximo de ruido blanco detectado, con el emisor dentro del área de cobertura, es necesario utilizar potencias de trasmisión tales que la distancia máxima a la que se llega a detectar señal supera los 50cm puestos como límite para no tener interferencia entre emisores.

Además, existen zonas dentro del área de cobertura en las que si se levanta el emisor a más de $2cm$, el primer pulso de señal recibido decrece tanto su amplitud que no llega a superar el umbral y se generan saltos en la distancia detectada, ya que requiere la recepción de varios pulsos hasta que uno supera el umbral y se detecta. Incluso puede llegar a darse que no se detecte señal. Si se baja el valor del umbral para solucionar este problema, aparecen interferencias que lo superan generando ruido en detección.

Para intentar solucionar estos efectos, se construyó un rectificador con seguidor de envolvente que se colocó entre la salida del amplificador y la entrada del comparador, de forma que se comparara la envolvente de la señal detectada en lugar de sólo el primer pulso.

El circuito es un rectificador de onda completa formado con dos amplificadores operaciones, para los que se usa un integrado OPA2134 y el seguidor de envolvente se logra con un condensador ubicado en la salida del rectificador. Valores pequeños del condensador introducen retardos pequeños en

la señal pero no se cambia mucho respecto a la señal rectificada y valores grandes forman una envolvente más marcada, pero que también introduce un retardo mayor. Valores adecuados para este condensador se encuentran entre $1nF$ y $10nF$. Para estos valores del condensador, la ganancia de la rectificación debe ser mayor a $2V/V$ para compensar la atenuación introducida por el condensador. También es recomendable ajustar el valor del umbral cambiando el valor de la resistencia R8 del circuito de recepción de ultrasonido.

Sin embargo, los resultados no fueron buenos dado que se introduce un jitter en la detección debido a las variaciones de las amplitudes de los pulsos recibidos. Además, la mejora en la altitud a la que se detecta la señal correctamente no mejora notoriamente y en caso de existir interferencia electromagnética se reduce la distancia mínima detectada del emisor dado que los pulsos de interferencia demoran más en extinguirse.

6.3. Microcontrolador del receptor

El principal contratiempo una vez escrito y programado el código, fue la aparición de un problema en la detección del PIC32 al conectarlo al puerto USB. Lo extraño de este problema era que por momentos la conexión se establecía correctamente y por momentos no. Resultó ser un problema en el código, ya que para que la comunicación USB funcione correctamente es necesario llamar periódicamente a la función USBTask(), la que se encarga de actualizar el estado del bus USB, y cuando el microcontrolador quedaba en espera de la llegada de la señal de infrarrojo pasaba un tiempo largo sin llamar a esa función.

Otro contratiempo importante se sufrió por un error en la hoja de datos del PIC32. En ésta se decía que los timers dejaban de contar una vez que la señal conectada al pin gated del timer recibía un flanco de bajada y que no seguía contando hasta que no se reseteara el timer por software. El hecho de que no siguiera contando era importante ya que la señal que se recibe de ultrasonido (que es la encargada de detener el timer) está formada por varios pulsos y se necesitaba que el timer se detenga cuando llegara el primero y que al llegar los demás pulsos no se volviera a activar. Esto no fue así, y el timer seguía contando activado por los restantes pulsos, por lo que fue necesario crear una rutina de atención a la interrupción que se ejecutara cuando llegara el primer pulso de ultrasonido y detenga el timer por software, no

permitiendo que se reactive cuando llegan los restantes pulsos.

Otro problema referente al primer prototipo, fue que la PIM (Plug In Module) del PIC32 no cuenta con hoja de datos y tiene todos los pines iguales a los del PIC con excepción de los 4 correspondientes al puerto USB que se encuentran cambiados. Esto provocó que cuando se creó el primer prototipo no funcionara la interfaz USB (dado que los pines estaban cambiados) y para su corrección se testearon uno por uno cada uno de los pines de la PIC para encontrar su correspondiente pin en la PIM. Cuando se descubrió que alguno de los pines de la PIC no se correspondían con los de la PIM se hicieron las correcciones en el primer prototipo y su funcionamiento fue correcto.

6.4. Driver

En una primera instancia se estudió la posibilidad de desarrollar el driver a nivel de kernel, implementando un módulo que corriera a este nivel. Para esto se estudió el libro Device Driver in Linux. El estudio de este libro fue una de las tareas que más tiempo llevó y de la que no se pudieron obtener frutos dado que se terminó implementando el driver a nivel de usuario utilizando la librería USB. Es por esto que el desarrollo del driver produjo un gran retraso.

6.5. Programación PIC12

Durante reiteradas ocasiones se tuvieron varios problemas al momento de programar los PIC12 principalmente debido a que el programador utilizado usa el puerto serie y no el paralelo como luego vimos era recomendado en varios foros de la web. Un problema bastante importante lo tuvimos al querer programar el PIC12 en formato SMD en la placa a la que se le construyó el encapsulado. Con esa tanda adquirida de PIC12, salvo por el primero con que probamos, los otros tuvieron problemas al momento de establecer la frecuencia del reloj interno lo que derivó en corrimientos mayores al 10 % en la frecuencia y por ende, en un mal funcionamiento del Lapix.

Para solucionar esto se generó un nuevo código en el que se fijó en aproximadamente 14μs los pulsos de infrarrojo en función de la nueva frecuencia del reloj interno. Este código puede encontrarse en el CD en CD/software/PIC12/-PIC12_v7.0/*.

Capítulo 7

Trabajo a futuro

7.1. Modificaciones a los circuitos

7.1.1. Circuitos de infrarrojo

La tarea a futuro más importante de todas es la de determinar otra forma de recibir, y tal vez emitir el infrarrojo, dado que es el causante de que no se consiga la resolución buscada y es la principal fuente de interferencia si se quiere utilizar más de dos lápices emisores simultáneamente a un distancia cercana.

Emitir con circuito ejemplo

Para deshacerse del jitter y las variaciones de retardo que introduce el módulo receptor, se recomienda probar utilizar el circuito emisor sugerido en la hoja de datos de la punta emisora [4]. Este circuito no fue utilizado debido a que no es compatible con el módulo receptor y en un principio, este módulo presentaba ventajas tentadoras.

La primera es que el integrado es capaz de generar la señal digital ahorrando la necesidad de implementar el circuito correspondiente, que a priori, para conseguir la tolerancia al ruido e interferencias necesaria, sería equivalente al que trae. Reduce el tamaño que ocuparía dicho circuito dado que viene dentro de un integrado. Como la señal emitida se modula a $36kHz$, es posible utilizar codificaciones standard para la comunicación de los botones y tal vez para identificar emisores para reducir interferencias. Estos módulos son usados comúnmente en televisores y sistemas que utilicen controles remotos, por lo que tienen amplia difusión y respaldo. Finalmente es de bajo costo.

La opción de emitir con el circuito ejemplo tampoco fue tomada en el transcurso del proyecto, debido a que el mal funcionamiento del módulo no se detectó hasta que ya no era viable implementar otra forma de recepción por falta de tiempo. En forma oportuna, se probó el correcto funcionamiento del módulo y con la resolución de los instrumentos utilizados, y dado que se utilizó la señal de prueba óptica que da la hoja de datos, no pudo verse el efecto del jitter.

El circuito ejemplo sugerido puede verse en la figura 7.1.

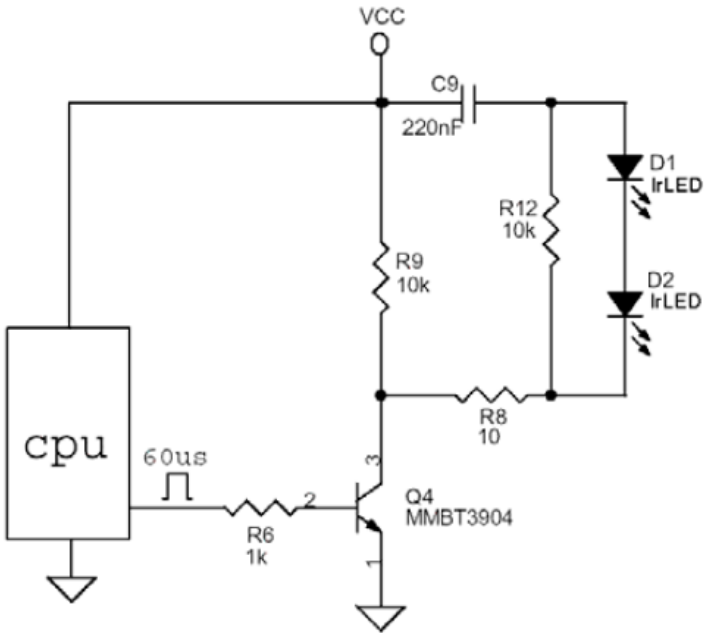

Figura 7.1: Circuito ejemplo sugerido en hoja de datos de punta emisora PT80KHZ-01 [4].

La idea de este circuito es generar un pulso de alto valor de corriente por los LEDs de forma que la intensidad de luz generada supere los márgenes normales y pueda ser detectada fácilmente con el fotodiodo BPW83 [49] aconsejado, a través de un filtro que ignore las señales provenientes del ambiente y deje pasar este pulso.

Los LEDs de la punta soportan $50mA$ de corriente continua, pero toleran picos de corriente de hasta $1{,}2A$ por $10\mu s$. La corriente generada con este circuito alcanza los $200mA$ y se extingue en unos $20\mu s$. Requiere un cierto tiempo antes de poder generar otro pulso.

Para implementar el circuito receptor tal vez alcance con un transistor común BC547, al cual se le adecua un valor de resistencia de base de forma que se active con la llegada del pulso y genere un pulso digital en su salida. Dado que este pulso digital es de corta duración, para poder utilizar el mismo microcontrolador es necesario utilizar un flip-flop que cambie de estado con cada llegada de estos pulsos. La comunicación del pulsado de los botones se puede realizar de forma similar a la que se utiliza actualmente.

Una implementación como ésta, además de ser sencilla y económica, probablemente no genere jitter. En cuanto a la interferencia entre emisores cercanos, habría que realizar pruebas para determinar si con un ajuste en la potencia de transmisión es suficiente para ser detectadas dentro del área de cobertura deseado y que no se detecten a más de $50cm$. De otro modo requeriría otro tipo de paliativo más elaborado y complejo a determinar.

Ajustar el AGC del receptor

Otra forma de intentar solucionar el problema con el jitter es construir el mismo circuito que trae internamente el módulo receptor de forma de poder establecer manualmente el valor del AGC. Si se establece este valor externamente, existen altas probabilidades de que no se perciba jitter a la salida.

Independencia con batería

En caso de mantener el mismo circuito emisor, se puede estudiar la posibilidad de modificarlo de forma que la corriente por los LEDs se independice del valor de la fuente de alimentación del circuito.

Para esto, basta con incorporar dos diodos idénticos en serie apuntando a tierra en la base del transistor y colocar la resistencia del colector en el emisor. De esta forma, cuando llega el pulso excitador se fija el voltaje en la base del transistor en $2V_D$, donde V_D es la caída de voltaje en cada diodo y como el V_{BE} del transistor vale aproximadamente lo mismo que V_D, el voltaje en la resistencia del emisor queda fijo en V_D. Así, con el valor de esta resistencia

fijamos el valor de la corriente que pasa por los LEDs independientemente del voltaje de alimentación.

7.1.2. Inversores del receptor

En la salida de cada circuito receptor de ultrasonido incorporamos un inversor formado con un transistor npn y un par de resistencias para que la señal sea activa por positivo y se conecte a los pines int1 e int4 del PIC32. El propósito de esto es generar una interrupción y fijar una bandera que indique que se recibió ultrasonido. Sin embargo, en la configuración del PIC32, puede establecerse que la bandera se active por negado, así que el uso de estos dos transistores es innecesario, un nuevo diseño debería obviarlos y cambiar el bit 1 y 4 del registro INTCON en el código del PIC32.

7.1.3. Circuito emisor de ultrasonido

El circuito emisor de ultrasonido a pesar de funcionar correctamente, sería deseable modificarlo de forma de reducir el consumo, mejorar la potencia de transmisión y reducir el tamaño físico mínimo que ocupa. Además, este circuito es el causante de necesitar al menos $5V$ para la emisión, lo que implica el uso del DC/DC step-up.

Su tamaño mínimo está fuertemente limitado por el inductor y el transistor utilizados. Para un valor de inductor de $18mH$, no es posible reducir su tamaño debido a que depende de las dimensiones del inductor. Sólo es posible adaptar estas dimensiones al diseño.

En tanto que para el transistor, es posible reducir su tamaño utilizando el transistor sugerido que posee un package HEXDIP que es más pequeño que el utilizado (TO-220).

El uso de otro transistor que pueda funcionar en zona activa con menor voltaje sería crucial para no utilizar el DC/DC step-up.

7.2. Cambio de componentes

Una tarea para optimizar costos y reducir tamaños sería buscar y seleccionar componentes equivalentes a los utilizados que sean más económicos y

posean encapsulados más pequeños.

El conector para el flex de la punta emisora no ajusta correctamente, por lo que debería seleccionarse otro más adecuado.

7.3. Modo de ahorro de energía

Algo sencillo e interesante de implementar para ahorrar consumo en la batería del emisor consiste en utilizar la funcionalidad del modo sleep del PIC12. Esta modalidad consiste en llevar el integrado a un estado en el que se utilizan la mínima cantidad de funciones de forma de ahorrar la energía consumida y reactivarse con la llegada de una señal externa o con una interrupción de un contador.

Durante este modo de operación el emisor no emitiría ningún tipo de señal. Una forma sencilla de implementarlo es por medio de software, utilizando un contador de forma de medir el tiempo desde el último pulsado de algún botón y entrar en este modo luego de 5 o 10 minutos. Para reactivarlo, bastaría con establecer las señales de los pulsadores para tal propósito y presionar alguno de ellos.

El inconveniente de esta solución es que puede llegar a ser molesto por parte del usuario y se debe esperar demasiado tiempo antes de que este entre en modo de ahorro para no entorpecer su uso.

Otra forma de implementar lo mismo y que lo haría transparente para el usuario, sería utilizar un sensor de tilt que detecta la inclinación del lápiz, de forma de mandarlo a dormir luego de permanecer, por ejemplo, 1 minuto en posición horizontal y despertarlo cuando ésta cambie.

7.4. Optimización de consumo

Como se vio en la sección de medidas de consumo del emisor 5.2.1, se podría realizar un estudio más completo que incluyera el uso de un regulador conmutado, los que suelen tener un consumo menor que el lineal utilizado. Además, ver la posibilidad de utilizar más de uno de forma que se ajusten al voltaje mínimo requerido por cada parte del circuito y evaluar otras posibles

soluciones.

El circuito emisor de infrarrojo, tal como está requiere al menos 3,0V para funcionar, el PIC12 según su hoja de datos [17] requiere 2,0V y el de ultrasonido requiere al menos 5,0V. Teniendo en cuenta estos requerimientos, los costos de los integrados, los componentes extra que requieren junto con sus costos y los consumos de cada uno, se debería evaluar cual es la configuración óptima.

Para el caso del receptor, debería estudiarse alguna forma de reducir su consumo, principalmente mejorar el código del PIC32 dado que este integrado representa la mayor parte del consumo de la placa. Otro tipo de mejoras serían, por ejemplo, cambiar el regulador lineal por uno conmutado.

7.5. Mejorar interferencia entre emisores

Una forma de reducir a un tercio la probabilidad de interferencia entre emisores que se encuentren cercanos a un mismo receptor, es hacer que el microcontrolador del receptor se sincronice con el primer emisor del que detecte señal correctamente y luego ignore las señales que se encuentre dentro de los dos tercios del período de emisión en los que no se envían señales.

También sería aconsejable evaluar otro tipos de paliativos contra este tipo de interferencia. Por ejemplo, sincronismo de señales o identificación de emisores.

7.6. Mejora de encapsulados

Los encapsulados construidos tanto para el emisor como para el receptor, fueron realizados artesanalmente y no se contó con el tiempo suficiente para terminarlos.

Al emisor sólo le faltaron detalles mínimos como afinarlo para hacerlo más liviano y redondearlo para darle mayor ergonomía.

En tanto que al encapsulado del receptor le quedó pendiente incorporarle un sistema de enganche para poder sujetarlo a la XO y agregarle una aber-

tura y topes en la tapa de forma de poder insertar el emisor para cargar la batería. Además, para disminuir interferencias del encapsulado en las señales y proteger mejor a los sensores, habría que reconstruir la pared delantera de forma de que cada sensor receptor de ultrasonido tenga línea de vista en toda el área de cobertura.

Es claro que también lo ideal sería rediseñarlos de forma de generar encapsulados capaces de ser producidos industrialmente en grandes cantidades.

7.7. Optimización del software

7.7.1. Calibración

La versión actual del Driver, implementa una interfaz para la calibración básica, que se ejecuta de forma manual, con fondo negro y texto plano. Una mejora, sería implementarla de forma más atractiva y llamativa para los ojos de un niño dado que está pensada para ser usada con las XO.

Esto consistiría en utilizar un fondo más colorido y en lo posible que se adecue a los que tiene la XO, por ejemplo blanco y verde para la actual. Que al enchufar el receptor al puerto USB se ejecute automáticamente y salte en pantalla una ventana indicando que se toque la pantalla con el lápiz si se desea utilizar sobre esta o que se presione el botón secundario de otro modo.

Luego, al iniciar la calibración, que los cuadritos actuales se sustituyan por cruces coloridas y tal vez animadas, o cualquier cosa que llame la atención y determine un punto central para que se presione. Además, podría estar acompañado de una flecha parpadeante con un texto que indique que se presione el centro con la punta del lápiz. Incluso tal vez podrían incluirse sonidos al tocar los puntos de calibración.

7.7.2. Actividad cuaderno

Para poder transformar la XO en un cuaderno con todas sus características más las ventajas que incluye el mundo digital, sería recomendado crear una aplicación específica con este cometido.

La aplicación consistiría en poder desplegar hojas rayadas o lisas en pantalla, en las que se puede escribir al presionar el botón primario, o sea, tocar la pantalla. Con opciones para cambiar el grosor de los trazos, el color de los trazos, con un ícono que al presionarlo transforme el puntero en una goma que se activa con el botón primario y todas las funcionalidades que acompañan cualquier editor de imágenes convencional, como guardar, cargar, salir, etc..

7.7.3. Compatibilidad con Windows y arranque automático

En la sección 3.4.7, se mencionan los avances obtenidos respecto a la compatibilidad con Windows.

Básicamente dado la falta de tiempo no se logró hacer que a libusb funcionara en Windows. El problema es que se debe realizar un archivo .info quien le dice a Windows que debe utilizar nuestro driver.

Con respecto al arranque automático, se realizó un script que se encuentran en el CD en CD/software/Driver/scripts/Inicar_LAPIX. Se copió el ejecutable en la carpeta del runlevel correspondiente y se le dieron permisos de ejecución, sin embargo no se logró que se autoejecutara al conectar al Lapix. Debe determinarse la razón de por qué no se ejecuta o lograr un modo alternativo para el arranque automático.

Capítulo 8

Conclusiones

8.1. Conclusiones generales

El objetivo del proyecto era diseñar y desarrollar un prototipo funcional de un lápiz electrónico inalámbrico capaz de escribir y dibujar en la pantalla de las computadoras del Plan Ceibal logrando un claro reconocimiento de los trazos y desplegándolo en pantalla de forma de generar la sensación de simultaneidad. Se puede afirmar que al término del proyecto este objetivo ha sido cumplido, logrando el diseño, ejecución de hardware y desarrollo de software necesario para el funcionamiento de todas las partes. Se logró que el prototipo desarrollado tuviera un desempeño satisfactorio que se puede constatar al usarlo.

Se logró transformar la computadora del Plan Ceibal en un cuaderno digital con la capacidad de escribir y dibujar cuando se utiliza con este dispositivo. Pero se fue más allá logrando que el prototipo desarrollado pueda funcionar en otras plataformas que utilicen Linux.

Se logró además que el Lapix pueda ser utilizado como un mouse, dado que se implementó el click izquierdo y derecho, permitiendo además el uso tanto dentro como fuera de la pantalla.

Otro objetivo logrado fue poder encontrar una batería que permita el uso diario sin problemas y que dé un largo tiempo de autonomía. Como trabajo extra se logró además utilizar una batería recargable, desarrollar el cargador correspondiente de forma simple e incorporarlo en las placas.

Vale también dejar en claro que las especificaciones buscadas no fueron

obtenidas en su totalidad, pero se cumplieron con las que se habían establecido como prioridad. Las especificaciones habían sido puestas sin un claro conocimiento de la tecnología a utilizar ni sus limitaciones. Se intentó superar características de productos comerciales existentes y dado esto, se impusieron metas que luego se supieron eran muy altas. Más allá de esto, se estudió por qué no se llegó a las especificaciones, teniendo identificado el causante, y se cuenta con distintas ideas concretas para lograr alcanzar lo establecido al principio del proyecto.

8.2. Conclusiones sobre el hardware

El hardware desarrollado cumplió por demás con lo proyectado al comienzo del proyecto, ya que se logró desarrollar ambas placas en diseño SMD, logrando además un comportamiento muy regular del mismo.

Se logró que las placas desarrolladas sean lo suficientemente robustas para ser utilizadas por niños. Las placas no cuentan con componentes delicados, a no ser por los receptores de ultrasonido a los que sería recomendable darles protección ante golpes.

Los microcontroladores de las placas emisora y receptora fueron correctamente escogidos. El utilizado en la placa emisora cumple con todo lo necesario y es eficiente en sus prestaciones. El de la placa receptora tiene capacidad por demás para las funciones a cumplir, por lo que se concluye que fueron adecuadamente seleccionados.

El ultrasonido funciona correctamente tanto en emisión como en recepción, y el problema que se le encontró, referente a la interferencia electromagnética que se filtra, fue solucionado mediante blindaje y software.

La tecnología utilizada para el infrarrojo no permite llegar a las especificaciones establecidas, en particular no permite llegar a la resolución esperada, pero tampoco está muy lejos de hacerlo. Para cumplir este objetivo es necesario cambiar la forma en que se implementa la emisión y recepción, y queda como un trabajo a futuro que se ha comenzado a investigar.

El consumo tanto de la placa emisora como de la receptora es razonable. La placa emisora logra tener una larga autonomía, sumado además al hecho de que se logró implementar una batería recargable. También se logró que la

placa receptora se alimentara sólo de un puerto USB, además de que los recursos que le son consumidos al PC son tales que no afectan en gran medida su funcionamiento normal.

8.3. Conclusiones sobre el software

Los objetivos que se habían establecido en cuanto al software fueron logrados, quedando como único tema pendiente que éste se ejecute automáticamente al conectar el Lapix al PC.

El software del microcontrolador de la placa emisora y receptora se adecuaron exactamente a lo que se esperaba al principio del proyecto.

En cuanto al desarrollo del software para el PC, se logró mejorar lo establecido. Se había planteado como objetivo que el software funcionara en la computadora del Plan Ceibal y fuera fácilmente portable a otras plataformas. Se logró más que esto, dado que el Lapix funciona en otras plataformas Linux y se comenzó a investigar para poder correr el código en Windows.

Siguiendo con el software del PC, se logró que la transformación que lleva la posición de la punta del lápiz a un pixel de la pantalla tuviera buena exactitud. Se logró además que el filtro desarrollado dé un buen acondicionamiento de la señal. También se logró que el software dé siempre la sensación de simultaneidad.

También se debe destacar el hecho de que el software le consume muy pocos recursos al PC, logrando que no se perciban diferencias de rendimiento con el Lapix conectado o sin él, y pudiendo correr el software con otras aplicaciones de la XO sin notar diferencias con respecto a su rendimiento normal.

Bibliografía

[1] Fast and Robust Trilateration for Multi-Robots Tasks - Maxelbot Trilateration Project. http://www.cs.uwyo.edu/~wspears/maxelbot/.

[2] Thomas Kunkel. Hardware architecture of a swarm of robots. Master's thesis, Department of Electrical and Computer Engineeringand The Graduate School of The University of Wyoming, 2004. http://www.cs.uwyo.edu/~wspears/maxelbot/Tom_Final_Writeup.pdf.

[3] Rodney Heil. A trilaterative localization system for small mobile robots in swarms. Master's thesis, Department of Electrical and Computer Engineeringand The Graduate School of The University of Wyoming, 2004. http://www.cs.uwyo.edu/~wspears/maxelbot/rod.thesis.pdf.

[4] Hoja de datos de punta emisora PT80KHZ-01. CD/hojas de datos/Emisor/PT80KHZ-01.pdf.

[5] Hoja de datos de sensores receptores de ultrasonido SR80KHZ-01. CD/hojas de datos/Receptor/SR80KHZ-01.pdf.

[6] Method and System for Digitizing Handwriting. Patent No. 5,977,958.

[7] Wikipedia. http://es.wikipedia.org/wiki/Sonido.

[8] Measurement Specialties, Empresa especializada en sensores. http://www.meas-spec.com/, última visita 14/06/2010.

[9] Hoja de datos del conector de la punta emisora HLW4S-2C7LF. CD/hojas de datos/Emisor/HLW4S-2C7LF.pdf.

[10] Hoja de datos del transistor npn BC547. CD/hojas de datos/Emisor/BC547.pdf.

[11] Hoja de datos del transistor npn MMBT3904. CD/hojas de datos/Emisor/MMBT3904.pdf.

[12] Hoja de datos del diodo de alto voltaje 1N4937. CD/hojas de datos/Emisor/1N4937.pdf.

[13] Hoja de datos del diodo de alto voltaje S2J-E3/52T. CD/hojas de datos/Emisor/S2J-E3-52T.pdf.

[14] Hoja de datos del inductor de 18mH 09P-183J-50. CD/hojas de datos/Emisor/09P-183J-50.pdf.

[15] Hoja de datos del transistor de alto voltaje IRFBC30. CD/hojas de datos/Emisor/IRFBC30.pdf.

[16] Hoja de datos del transistor de alto voltaje IRFDC20. CD/hojas de datos/Emisor/IRFDC20.pdf.

[17] Hoja de datos del microcontrolador PIC12F629. CD/hojas de datos/Emisor/PIC12F629 675.pdf.

[18] Hoja de datos del optoacoplador PC817. CD/hojas de datos/Emisor/PC817.pdf.

[19] Hoja de datos del optoacoplador KB817. CD/hojas de datos/Emisor/KB817-847.pdf.

[20] Hoja de datos del regulador lineal de voltaje LP2950CZ3.3. CD/hojas de datos/Emisor/LP2950CZ3-3.pdf.

[21] PowerStream Technology. http://www.powerstream.com, última visita 07/06/2010.

[22] All-Battery.com. http://www.all-battery.com/, última visita 07/06/2010.

[23] BatterySpace.com. http://www.batteryspace.com/, última visita 07/06/2010.

[24] Rechargeable Cylindrical Lithium Ion Cells from PowerStream. http://www.powerstream.com/li-cylindrical.htm, última visita 07/06/2010.

[25] UltraFire 14500 3.6V Lithium Rechargeable Battery. http://www.tradevv.com/chinasuppliers/brinyte_p_b6c23/china-UltraFire-14500-3-6V-Lithium-Rechargeable-Battery.html, última visita 07/06/2010.

[26] TrustFire 10440 600mAh 3.7V AAA Li-Ion PROTECTED Battery. http://www.batteryjunction.com/trustfire-aaa-600mah-tr10440.html, última visita 07/06/2010.

[27] UB1733 Ultralife Electronic Battery. http://www.mouser.com/ProductDetail/Ultralife/UB1733/?qs=sGAEpiMZZMuXcNZ31nzYhYzVBmcrvUxY5mmb18Z2H4E%3d, última visita 07/06/2010.

[28] Especificaciones de baterias de Litio. http://www.powerstream.com/li.htm, última visita 03/06/2010.

[29] Circuito de Protección para baterias de Li-Ion. http://www.unicrom.com/art_BateriasLitioIon1.asp.

[30] Carga en Baterías de Li-Ion. http://www.buchmann.ca/article18-page3-spanish.asp.

[31] Cargador de baterías de Li-Ion. http://www.pablin.com.ar/electron/circuito/instlab/chgliion/index.htm.

[32] Lithium-ion Battery Charging Basics. http://www.powerstream.com/li.htm.

[33] Lithium-ion (Li-Ion) Battery Protector. http://www.batteryfacts.co.uk/BatteryTypes/LiIonProtectionCiruit.html.

[34] Li-Ion Protection Modules. http://www.aplusproducts.com/gallery/products/batteries/protection_modules.pdf.

[35] PCB for 3.7V of Li-Ion Battery (2.0A limit). http://www.batteryspace.com/PCB-for-3.7V-of-Li-Ion-Battery-2.0A-limit.aspx.

[36] Hoja de datos de impreso protector de baterías Li-Ion 'PCB for 3.7V of Li-Ion Battery (2.0A limit)'. http://www.batteryspace.com/prod-specs/2721.pdf o CD/hojas de datos/Alimentacion/PCB protection for 3,7V of Li-Ion Battery.pdf.

[37] Hoja de datos de protección de baterías de Li-Ion DW01. CD/hojas de datos/Alimentacion/DW01.pdf.

[38] Hoja de datos de MOSFET de protección de baterías de Li-Ion 5N20V. CD/hojas de datos/Alimentacion/5N20V.pdf.

[39] Hoja de datos del DC/DC Step-up NCP1400A). http://www.sparkfun.com/datasheets/Prototyping/Batteries/Voltage_Switcher-NCP1400A-D.pdf o CD/hojas de datos/Alimentacion/NCP1400A-D.pdf.

[40] NCP1400-5V Step-Up Sparkfun Product Info. http://www.sparkfun.com/commerce/product_info.php?products_id=8999.

[41] Cargador de baterías de Li-Ion. http://www.pablin.com.ar/electron/circuito/instlab/chgliion/index.htm.

[42] Hoja de datos del cargador de baterías Li-Ion MAX1555. http://pdfserv.maxim-ic.com/en/ds/MAX1551-MAX1555.pdf o CD/Alimentacion/MAX1551-MAX1555.pdf.

[43] LiPoly Charger - Single Cell 3.7-7V Input based on MAX1551 - Sparkfun Product Info. http://www.sparkfun.com/commerce/product_info.php?products_id=726.

[44] Curso Instalaciones Eléctricas - Facultad de Ingeniería - Universidad de la República. http://iie.fing.edu.uy/cursos/course/view.php?id=68.

[45] All About Circuits. http://www.allaboutcircuits.com/vol_1/chpt_3/4.html, última visita 15/06/2010.

[46] Ministerio de Trabajo y Asuntos Sociales de España. Aspectos particulares de los efectos de la corriente eléctrica. http://www.insht.es/InshtWeb/Contenidos/Documentacion/FichasTecnicas/NTP/Ficheros/401a500/ntp_437.pdf, última visita 15/06/2010.

[47] Hoja de datos del amplificador operacional OPA2134. CD/hojas de datos/Receptor/OPA2134.pdf.

[48] Hoja de datos del comparador LM339. CD/hojas de datos/Receptor/LM339.pdf.

[49] Hoja de datos de fotodiodo receptor de infrarrojo BPW83. CD/hojas de datos/Receptor/BPW83.pdf.

[50] Hoja de datos del módulo receptor de infrarrojo TSOP4836. CD/hojas de datos/Receptor/TSOP4836.pdf.

[51] Hoja de datos del microcontrolador PIC32MX460F512L. CD/hojas de datos/Receptor/PIC32MX3XX-4XX.pdf.

[52] Documento adjunto sobre la elección del integrado del receptor. CD/documentos/Elección del chip para el receptor.pdf.

[53] Velocidad del sonido. http://html.rincondelvago.com/velocidad-del-sonido.html.

[54] Wikipedia, the free encyclopedia: Speed of sound. http://en.wikipedia.org/wiki/Speed_of_sound.

[55] Hoja de datos de la compuerta lógica AND HCF4081B. CD/hojas de datos/Receptor/HCF4081B.pdf.

[56] MPLAB ICD 3 In-Circuit Debugger User's Guide. http://ww1.microchip.com/downloads/en/DeviceDoc/MPLAB_ICD3_UG_51766a.pdf.

[57] Using MPLAB ICD 3. http://ww1.microchip.com/downloads/en/DeviceDoc/DS-51765C.pdf.

[58] MPLAB ICD 3 In-Circuit Debugger. http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en537580&redirects=icd3.

[59] Universal serial bus specification revision 2.0, 2000. http://www.poweredusb.org/pdf/usb20.pdf.

[60] Usb device stack for pic32 programmer's guide. http://ww1.microchip.com/downloads/en/AppNotes/01176A.pdf.

[61] Usb device and embedded host stack for pic32. http://ww1.microchip.com/downloads/en/DeviceDoc/pic32mx_usb_v1.04-LEGACY.zip.

[62] Use of microchip copyrighted material. http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=487&param=en023282.

[63] Sitio oficial de la libUSB. www.libusb.org/.

[64] Curso Tratamiento Estadístico de Señales - Facultad de Ingeniería. http://iie.fing.edu.uy/cursos/course/view.php?id=81.

[65] Simon Haykin. *Adaptative Filter Theory*. Prentice Hall.

[66] Monson H. Hayes. *Statistical Digital Signal Processing and Modeling*. John Wiley & Sons, Inc., 1996.

[67] Acceleration Models. http://de.scientificcommons.org/43286567.

[68] Mario Enrique Munich. *Visual Input for Pen Based Computers*. PhD thesis, California Institute of Technology, Pasadena, California, 2000. In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy.

[69] Random walk model. http://web.duke.edu/~rnau/411rand.htm.

[70] Arthur Gelb, Joseph F. Kasper. *Applied Optimal Estimation*. The M.I.T. Press Massachusetts Institute of Technology, Cambridge Massachusetts and London England, 1974.

[71] Brian D. O. Anderson y John B. Moore. Thomas Kailath, Editor. *Optimal Filtering*. Prentice Hall, 1979.

[72] Oppenheim A.V., Schafer R.W, 2nd Edition. *Discrete-Time Signal Processing*. Prentice Hall, 1998.

[73] The Xlib Manual. http://tronche.com/gui/x/xlib/.

[74] X Org Fundation. http://wiki.laptop.org/go/Remote_display.

[75] Wiki Laptop. http://wiki.laptop.org/go/Remote_display.

[76] Cómo crear Scripts. http://www.ubuntu-es.org/node/14511, última visita 17/06/2010.

[77] Cygwin/X. http://x.cygwin.com/.

[78] X Window System. http://en.wikipedia.org/wiki/X_Window_System.

[79] libusb-win32 Proyects. http://sourceforge.net/apps/trac/libusb-win32/wiki/libusbwin32_examples.

[80] libusb-win32 Page. http://sourceforge.net/apps/trac/libusb-win32/wiki/libusbwin32_examples.

[81] Especificaciones técnicas del multimetro digital Microtek DT9208L. CD/referencias/Multimetro Microtek dt9.pdf.

[82] Especificaciones técnicas de la XO. http://wiki.laptop.org/go/Especificacion_de_hardware.

[83] Especificaciones técnicas del lápiz digital Pegasus. CD/referencias/Dossier PNF DUO.pdf.

[84] Especificaciones técnicas del lápiz digital Pegasus. CD/referencias/Pegasus P220 Specifications.pdf.

[85] Blog Jarfil: Los Mpx del ojo humano. http://blog.jarfil.net/los_mpx_del_ojo_humano.

ANEXOS

Anexo A

Manual de Uso e Instalación

A continuación se describen los pasos necesarios para la instalación y puesta en funcionamiento del driver de Lapix.

A.1. Instalación

A.1.1. Librerías necesarias

Para poder compilar el código del driver en la XO, y en cualquier máquina, es necesaria la instalación de algunas librerías.

Comandos a ejecutar en la XO (desde consola[1]):

- yum install make *(para ejecutar archivos makefile)*
- yum install gcc *(para poder compilar)*
- yum install libusb-dev *(necesario para el funcionamiento de la libusb)*
- yum install libX11-devel *(necesario para el funcionamiento de la xlib)*

A.1.2. Edición del archivo "xorg.conf"

Dado que por defecto en la XO la extensión del XServer XTST esta deshabilitada, se debe editar el archivo de configuración "xorg.conf" (/etc/X11/xorg.conf), tal como se menciona en la sección 3.4.6.

[1]Para acceder a consola se debe presionar las teclas ctrl + alt + Vecindario, luego se debe ingresar como usuario root.

Edición a Realizar

Se sustituye la linea:

```
Option "XTEST" "Disable" # Mostly a debugging tool
```

Por:

```
Option "XTEST" "Enable" # Mostly a debugging tool
```

A.1.3. Script de ejecución

Dado que el driver debe ser ejecutado desde consola, se generan conflictos con el XServer el cual require de un display para funcionar. Para que funcione correctamente, se debe "exportar" las variables de display. Para ello se genera un script el cual será utilizado cada vez que se desee ejecutar el driver (ver sección 3.4.6).

Escritura del Script

1. Utilizando un editor de texto generamos un archivo con el siguiente contenido:

```
#!/bin/bash
echo Ejecutando export display....
export DISPLAY=:0.0
echo inicia LAPIX
./lapix
```

   Guardo el archivo con el nombre: "lapix.sh". Es importante poner la extensión ".sh"

2. Copio el script creado a la carpeta /usr/bin. Esto me permite poder ejecutarlo sin importar el directorio donde me encuentre.

3. Le doy permisos de ejecución:

```
chmod +x lapix.sh
```

Compilación

Una vez copiado los archivos .c y .h del driver se procede a compilar. Se debe haber copiado también el archivo makefile.

Todos los archivos deben encontrarse en una misma carpeta. Luego que los tenemos todos en la misma ubicación debemos realizar la compilación. Para ello, desde consola nos situamos en la carpeta donde se guardaron los archivos del driver y ejecutamos el comando make:

```
make
```

Esto nos generará un ejecutable de nombre Lapix. Para correr el driver, simplemente ejecutamos desde linea de comando "lapix.sh"

A.2. Ejecución

Una vez que se realizaron todos los pasos mencionados en A.1, se pude proceder a la ejecución.

Para ello, parados en la carpeta donde realizamos la compilación, ejecutamos:

```
lapix.sh
```

Luego de ésto, si estamos en consola debemos cambiarnos a modo gráfico[2].

Una vez en modo gráfico, el driver nos pregunta si queremos trabajar sobre la pantalla o fuera de ella.

Trabajar fuera de la pantalla

Para trabajar fuera de pantalla, presionamos click derecho[3] y podemos comenzar a utilizar el lápiz normalmente.

[2]Para ir al modo gráfico se debe presionar ctrl + alt + Inicio.
[3]Botón ubicado en el costado del emisor.

Trabajar sobre de la pantalla

Para seleccionar este modo de trabajo, luego de ejecutar el driver, se debe presionar click izquierdo[4], lo que dará paso a la calibración.

Calibración

Para dar comienzo a la misma se presiona nuevamente click izquierdo.

1. Aparecerá un cuadrado blanco sobre la esquina superior izquierda de la pantalla, se debe colocar la punta del lápiz sobre dicho cuadrado y presionar click izquierdo.
2. Ahora aparecerá el mismo cuadrado sobre la esquina superior derecha, nuevamente se posiciona la punta sobre dicho cuadrado y se presiona click izquierdo.
3. Se realiza el mismo procedimiento para las dos esquinas restantes.

Una vez finalizada la calibración se puede utilizar el lápiz normalmente.

[4]Botón ubicado en la punta del emisor. Presionarlo equivale a tocar una superficie con la punta del emisor.

Anexo B

Lista de componentes y Costos

La lista de componentes se encuentra adjunta dentro del CD del proyecto en CD/documentos/Lista de componentes.pdf.

A partir de esta lista se realiza una tabla con los precios por unidad de cada uno de ellos y otra tabla con los precios al por mayor, las cuales pueden encontrarse adjuntas en CD/documentos/Lista de componentes y precios.xls.

Respecto a esta tabla cabe aclarar ciertos detalles.

La tabla no incluye impuestos, ni costos de envío y no se toma en cuenta el costo que implica la construcción de las placas, ni sus encapsulados. Además, el precio del pulsador primario, obtenido de una disquetera vieja, es sustituido por el costo de los otros pulsadores utilizados por considerarse similar y relativamente despreciable.

Como resultado, se obtiene que el costo de los componentes necesarios para construir un prototipo funcional es de **USD 72.065**.

Esta cifra sirve a nivel de referencia, se considera que realizando un estudio adecuado de mayor profundidad puede llegar a reducirse considerablemente. Lo mismo sucede con el costo al por mayor, el cual es de **USD 53.164**.

El costo del encapsulado fue nulo para el grupo pero el de los impresos fue elevado ya que se mandaron a hacer dentro del mercado local para optimizar tiempos. Su costo fue de USD 66.855 por lo que el prototipo tuvo un costo total de **USD 138.92**[1].

[1]El valor de los impresos se analiza por separado dado que en caso de contar con el equipo adecuado para poder realizarlos su costo puede llegar a tornarse casi despreciable

Para el precio de la punta emisora y los sensores receptores de ultrasonido sólo se dispone del precio por unidad que es de USD 10 cada uno. Por lo que el precio total de estos 3 componentes asciende a USD 30 lo que representa un 41,6 % del costo total de un prototipo y un 56,4 % del costo de un prototipo al por mayor.

De esta forma, una reducción en el costo del prototipo está limitada por la punta emisora y los receptores, por lo que se recomienda consultar a la empresa Measurement Specialties [8] en cuanto el precio al por mayor y en caso de que la reducción en el precio no fuese significativa, se recomienda buscar un sustituto de estos componentes.

El otro componente que le sigue de mayor valor es el PIC32 con un precio de USD 9.1 por unidad y USD 6.3 de a 100. Para optimizar, puede ser sustituido por el PIC32MX440F128L que posee menos memoria pero su valor se reduce en poco más de 1 dólar.

La tabla fue elaborada principalmente en base a la empresa Mouser, por lo que también se recomienda consultar con otros distribuidores para disminuir el costo.

El uso de la batería implica un costo de materiales de USD 10.705 en la construcción de un prototipo y de USD 6.5217 en una construcción al por mayor. Por lo que de utilizar el emisor cableado, los componentes de un prototipo costarían USD 61,36 y USD 46.642 al por mayor.

Anexo C

Comparación planificación vs. hechos

C.1. Fechas planificadas vs. Fechas reales

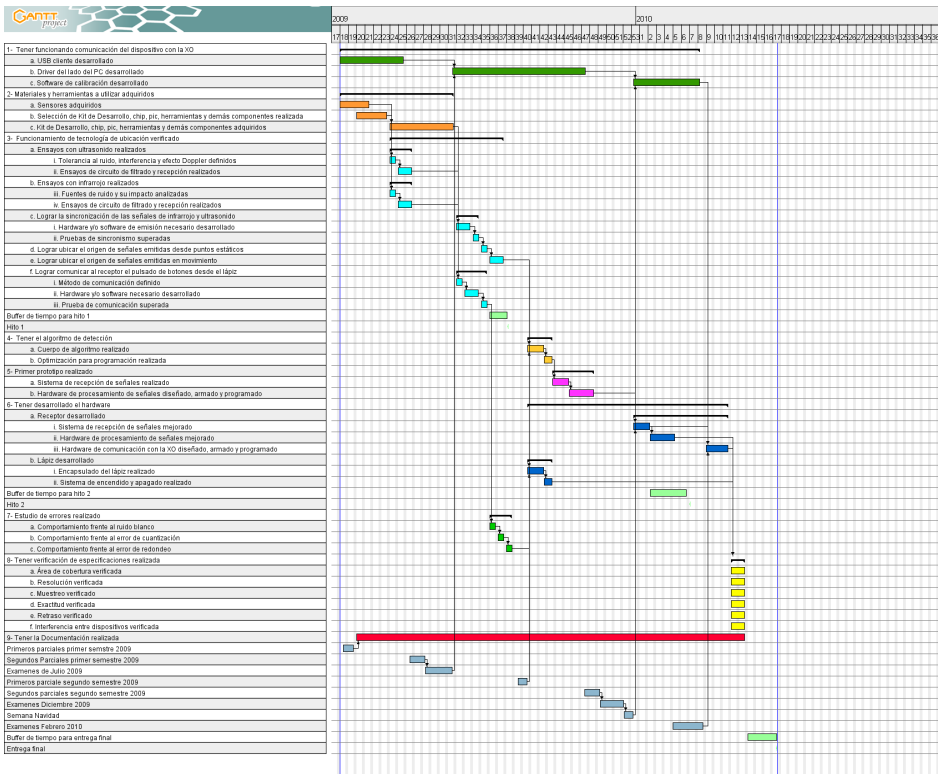

Figura C.1: Gantt Planeado

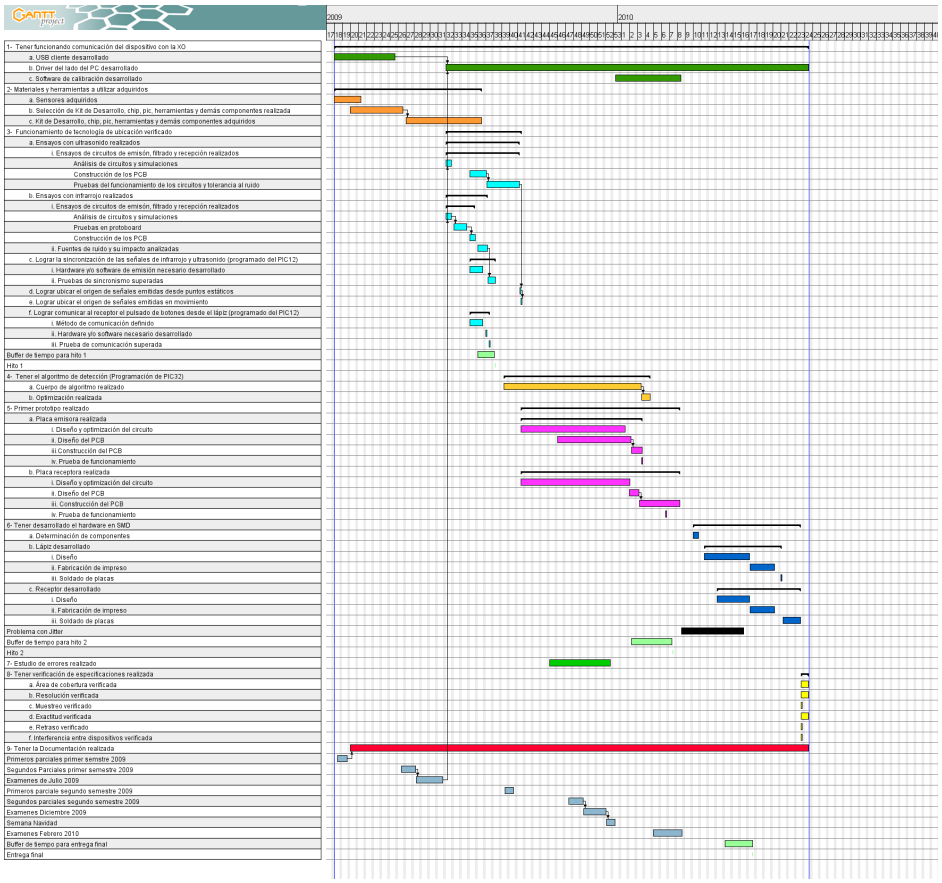

Figura C.2: Gantt Real

C.2. Comentarios sobre los desvíos y sus causas

En la figura C.1 se muestra el gantt propuesto para el plan de proyecto. En la figura C.2 podemos apreciar el gantt real que resultó durante el transcurso del Proyecto.

Se observan algunas diferencias importantes. Entre ellas la realización del driver y el primer prototipo. La razón por la que se extendió la tarea del driver fue que se invirtió mucho tiempo en intentar implementar el mismo a nivel de kernel. Se pasó tiempo leyendo bibliografía para tales efectos pero sin rendir frutos. Finalmente el mismo se implementó a nivel de usuario utilizando librerías específicas para la lectura de datos desde USB y para desplegar movimientos en pantalla.

La otra tarea que llevó considerablemente más tiempo del que se le asignó fue la realización del primer prototipo. Esto básicamente se debió a que la programación del PIC32 presentó grandes obstáculos, lo que derivó en que la programacion del mismo se demorara provocando el atraso en la construcción del prototipo.

Otra razón no menor que influyó en el tiempo de realización del primer prototipo fue el problema del jitter en la recepción de infrarrojo. No se logró resolver adecuadamente (o reducirlo lo mós posible) hasta fines de abril.

Cabe destacar que también se tuvieron demoras en la primera importación realizada lo que provocó que recién se pudiesen comenzar a hacer pruebas de hardware (básicamente recepción de US) dos meses después de lo planeado.

En sí existieron varias causas las cuales llevaron a que los tiempos se dilataran. Pero el principal causante de las diferencias de lo planeado respecto a lo que se llevó a cabo, fue el hecho de que originalmente las tareas se pensaron para ser repartidas entre 4 personas y lamentablemente sólo 3 integrantes las ejecutaron. Esto fue debido a que uno de los integrantes no realizó ningún aporte al proyecto lo que desencadenó en la desvinculación de éste respecto del grupo y finalmente en que perdiera la asignatura.

Todo lo anterior llevó a un pedido de prórroga de 5 semanas, con una extensión de 2 semanas para terminar la documentación.

Anexo D

Obtención de especificaciones

Al inicio del proyecto para obtener las especificaciones aspiradas se tomaron en cuenta diversos factores y se realizaron algunas pruebas. Lo primero que se consideró fueron las especificaciones de la XO de forma de determinar que es lo que es capaz de brindarnos el anfitrión objetivo.

También para tener un marco de referencia se consideraron dispositivos comerciales similares a nuestro proyecto, se tomaron en cuenta las características de los mouse y por último se tuvo la oportunidad de realizar algunas pruebas de escritura gracias a un préstamo del LATU con un Pegasus, lápiz digital comercial que incluía la netbook Classmate de Intel, que fue competencia de la XO en las primeras licitaciones.

En todos los casos, se intenta determinar las características mínimas que se requieren para obtener los resultados esperados, teniendo en cuenta que se pretende disminuir costos, simplificar lo que después se tiene que desarrollar y además, a priori no se sabía con certeza si se iban a poder conseguir.

D.1. Área de cobertura

Para determinar un área de cobertura adecuada se tomó como cota mínima elegible a la pantalla de la XO que es de 7.5" (152,4mm x 114,3mm) [82]. Luego se observó que el lápiz digital de la compañía Pen And Free, el lápiz DUO, que fue el producto comercial encontrado con mayores similitudes con nuestro proyecto, posee un área de cobertura de 15.4" [83], por lo que se considera ésta como la cota máxima elegible por ser un producto comercial ya establecido que no se pretende superar con un prototipo funcional en un

proyecto de fin de carrera.

Por otro lado el Pegasus, que es de una gama bastante económica y de pocas prestaciones dentro de los lápices digitales comerciales, tiene un área de cobertura del tamaño A4 (210mm x 297mm) que equivale a 14.32" [84].

Teniendo en cuenta lo anterior, considerando interesante la posibilidad de utilizarlo sobre una hoja A4 dado que es una medida estándar y razonable, y teniendo en mente que interesa tener portabilidad del lápiz dado que no se sabe de que tamaño puedan ser las pantallas en próximas licitaciones, es que se decide tomar como área de cobertura hasta el 90 % de una pantalla de 15" (230mm x 305mm) que equivale a una A4.

D.2. Resolución

Para obtener una resolución buscada, primero nos planteamos una cota superior y buscamos que tanta resolución puede apreciar una persona. Encontramos que el ojo humano ve unos 2Mpx por cada 15º de diámetro de campo visual. Traducido a una foto, el límite superior estará en los 2Mpx en un área de 8cm x 7cm a 30 cm de distancia, o lo que es lo mismo [85]:

- ≈ 6Mpx en una foto de 15x10cm a 30cm de distancia (≈ 500DPI)
- ≈ 5Mpx en una TV de 17" a 2m de distancia
- ≈ 4Mpx en un cartel de 2m a 10m de distancia (≈ 40DPI)

De acá podemos considerar que unos 500DPI podrían ser una buena cota superior.

Tomando en cuenta que el objetivo principal es usarlo para escribir sobre la pantalla de la XO nos preguntamos que tanta resolución es capaz de dar y encontramos que alcanza hasta 200DPI [82].

Sin embargo, también es de interés que funcione como un mouse y estos suelen tener resoluciones de 400, 800 y 1600 DPI o más. Por lo que sería deseable alcanzar los 400 DPI.

Si observamos los productos comerciales existentes, tenemos que el lápiz DUO tiene una resolución de 600DPI [83] que le permite funcionar como

mouse, en tanto que el Pegasus que es sólo lápiz digitalizador y no funciona como mouse tiene 100DPI de resolución [84].

Dado que nuestra prioridad es poder escribir sobre la pantalla de la XO capaz de dar 200DPI, que la funcionalidad de mouse a pesar de que es deseable no es primordial que sea de alta calidad, tomando en cuenta el mismo que criterio que para el área de cobertura de no intentar superar un producto comercial y apoyados en pruebas de funcionamiento del Pegasus, es que se decide fijar la resolución esperada en 200DPI de forma que el trazo sea reproducido con alta fidelidad y que la funcionalidad de mouse tenga un desempeño aceptable.

D.3. Muestreo

Para determinar una frecuencia de muestreo adecuada primero nos fijamos cuál es la que utilizan los mouse conectados al puerto USB y encontramos que en el caso de utilizarse con Windows XP, se puede ajustar el muestreo en pasos de 80 reportes por segundo (Hz) hasta los 200 reportes por segundo (Hz).

Acudiendo nuevamente a los productos comerciales, tenemos que el Pegasus funciona a 58 muestras / segundo [84] y el lápiz DUO utiliza una frecuencia de 80 muestras / segundo [83].

Con esto acotamos las posibilidades entre 58 y 80 muestras por segundo. Para determinar algo más apropiado se realizaron diversas pruebas. Se pudo comprobar que la frecuencia utilizada por el Pegasus da resultados aceptables pero no muy buenos.

Además, se realizaron filmaciones de escritura las que se pasaron cuadro por cuadro y se consideró que para poder reproducir cualquier tipo de trazo en un uso normal se requieren al menos unas 6 muestras cada 80ms, o sea, 75 muestras / segundo. Los videos y la descripción de algunas pruebas realizadas pueden encontrarse en el CD en CD/pruebas/obtencion_de_especificaciones/*.

D.4. Exactitud

Para determinar un valor de exactitud sólo se encontró como referencia que el Pegasus posee 0.2mm [84], la que nos pareció muy elevada ya que al menos al usarlo sobre la pantalla, la calidad de la calibración va a jugar un papel mucho más importante.

El grosor de un trazo común de lapicera o lápiz mecánico es de aproximadamente 0.5mm, por lo que nos pareció que un error de esta magnitud podía ser tolerable y se eligió esa como exactitud.

D.5. Retraso

Para el retraso no encontramos ninguna referencia en los productos comerciales similares. De esta forma, para intentar determinar un valor razonable observamos que sucede con los mouse. En Windows por defecto se tiene el puerto USB trabajando a 125Hz de los 1000Hz que es capaz de dar, obteniendo un tiempo de respuesta de 8ms. Se puede cambiar esta frecuencia a 250Hz (4ms), 500Hz (2ms) y 1000Hz (1ms) para mejor estos tiempos.

Luego realizamos filmaciones de escritura con el Pegasus para ver sus tiempos de respuesta las cuales pueden encontrarse en el CD en CD/pruebas/-obtencion_de_especificaciones/filmaciones_de_escritura/*.

Notamos que un retraso de 80ms no afecta la percepción visual de continuidad. Sin embargo, el retraso depende de varios factores, principalmente de la cantidad de memoria RAM disponible en el equipo anfitrión y del uso de su procesador, por lo que intentar dar una medida sin saber que tan pesados serán los códigos o que tantos recursos pueden llegar a tomarse del anfitrión, nos pareció muy arriesgado. De esta forma, se decidió definir el retraso como uno tal que genere la sensación de simultaneidad necesaria para que un niño usuario de la XO se sienta cómodo utilizando el lápiz sobre la pantalla.

D.6. Interferencia entre dispositivos

Por último, para determinar una distancia de tolerancia ante interferencias entre Lapixes razonable, se consideró el caso en que se esté utilizando en

una clase y que cada niño cuente con uno propio, por lo que se asume que cada estudiante se encontraría separado el uno del otro a por lo menos 50cm de distancia. De esta forma, se decide definir esta especificación como que se ignorarán señales que se encuentren por fuera de un radio de 50cm del receptor.

Anexo E

Separación de Receptores

A continuación se muestran algunas imágenes donde se graficó la varianza de los puntos cuando la separación entre los receptores era de 8cm y 6cm. En las mismas se aprecia claramente que el ruido es demasiado si se construía a 6cm.

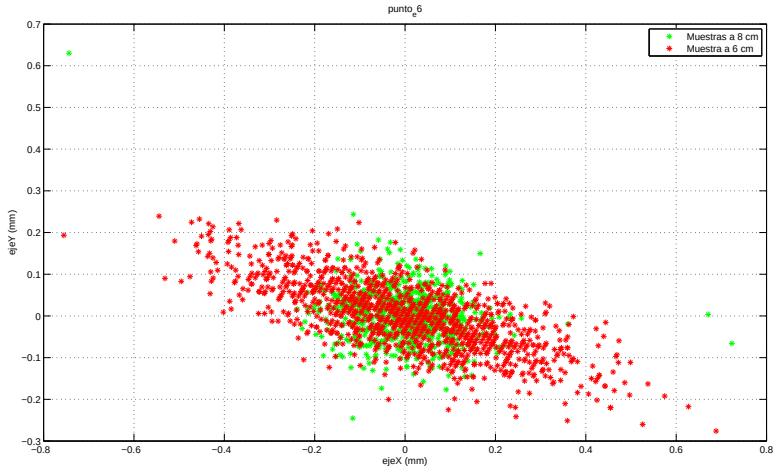

Figura E.1: Varianza puntual con distancia de sensores a 8 y 6 cm

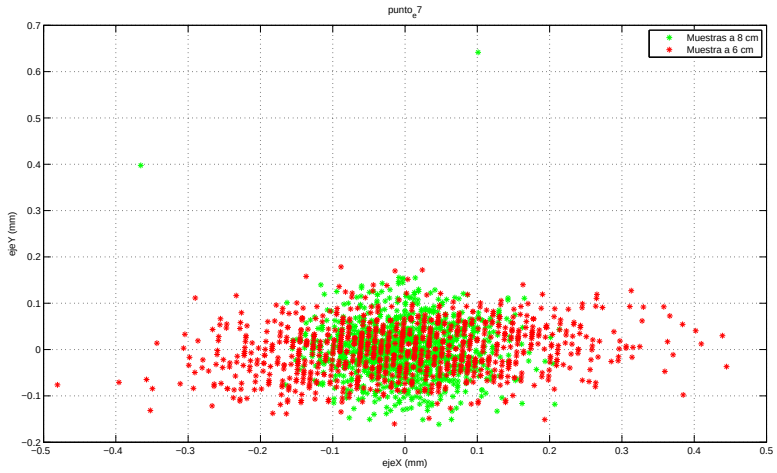

Figura E.2: Varianza puntual con distancia de sensores a 8 y 6 cm

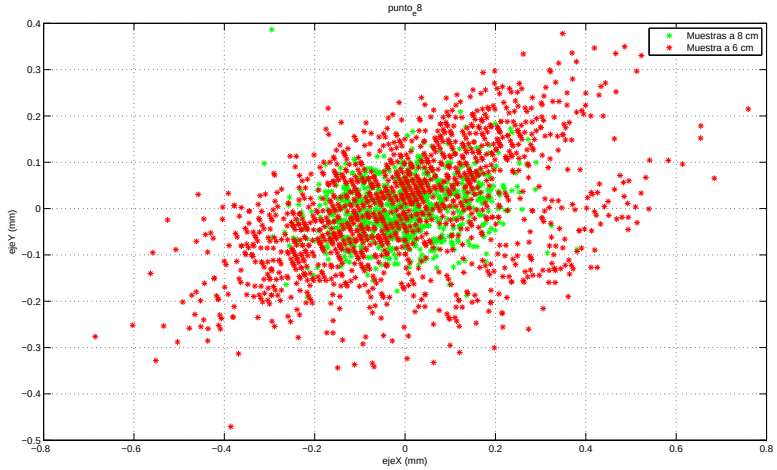

Figura E.3: Varianza puntual con distancia de sensores a 8 y 6 cm

Anexo F

Patentes y Productos Similares

Proyectos similares

Baliza de adquisición tridimensional (BAT)

Realizado por: Andrés Colensky, Alvaro Pérez, Diego Rother
Tutor: Gregory Randall
Año: 2001

Este es un dispositivo capaz de ubicar las coordenadas espaciales de un puntero dentro de una habitación, además de realizar mediante una aplicación una representación gráfica que muestra mediante un dibujo tridimensional el recorrido del puntero.

El sistema se basa en el envío de secuencias pseudo aleatorias a través de cuatro emisores de ultrasonido colocados en posiciones fijas y el sensado de las señales por un transductor ubicado en el puntero que se desea saber su posición.

Resolución: 0.3mm

Limitantes del proyecto BAT:

- la direccionalidad de los transductores dificulta estar sincronizado con tres de ellos al mismo tiempo para poder medir correctamente la posición, por lo que se ve limitado el movimiento del puntero, ya que siempre debe estar mirando a los transductores.
- dado que el ultrasonido es incapaz de atravesar objetos, cuando esto ocurra se darán errores en el posicionamiento.

- introducción de errores dada la reflexión del ultrasonido en paredes, piso, techo, etc.
- introducción de errores por efecto Doppler debido al movimiento relativo entre los emisores y el receptor.

Productos comerciales similares

Macbook Tablet Touch, Tablet power book, Wacom Cintiq

Estos tres son pantallas touchscreen en las cuales se puede escribir.

Wacom, Bamboo, Genius Tablet

Los anteriores dependen de una tableta especial y se orientan al diseño gráfico profesional.

Livescribe Smartpen

Tiene un montón de utilidades pero no es utilizado para digitalización en tiempo real de escritura ni sirve para escribir sobre una pantalla.

Pegasus Digital Pen

Versión comercial del lápiz que trae la Classmate. No tiene utilidades de Mouse ni permite utilizarse sobre la pantalla, sólo sobre papel. Funciona con infrarrojo y ultrasonido.

Algunas de sus especificaciones son:

Coverage area: up to A4 (210 x 297mm)

Resolution: 100 DPI (dots per inch = ppp puntos por pulgada)

Sampling rate: 58 samples/second

Accuracy: 0.2mm

Communication: USB 1.1 Low Speed

Pen Battery: 2 x SR41 batteries (1.55V)

Pen Battery Life Time: 80 hours of continues

Platform support: Windows XP and Vista

Pen And Free Lápiz Duo

El producto existente más parecido a Lapix, es el Lápiz Duo de la empresa coreana Pen And Free, el cual, fue diseñado para poder tomar notas sobre la pantalla de monitores o Laptops con la opción de escribir sobre papel. Pero éste no fue pensado como sustituto de un cuaderno, sino como para tomar algunas notas en pantalla (ya que ésta se encuentra vertical) y adquirir lo escrito en un papel. También cuenta con la funcionalidad de Mouse. Sin embargo, su precio, sin considerar los impuestos de importación, ronda los 50 dólares y no se encuentra disponible en el mercado local. También funciona con infrarrojo y ultrasonido.

Algunas de sus especificaciones son:

Resolución: 600DPI (dots per inch = ppp puntos por pulgada)

Muestreo: 80 muestras / segundo

Comunicación: USB 1.1 o versiones superiores

Batería del lápiz Li-ion o LR41 o Celdas AAA según el modelo

Sistema Operativo: Windows XP o superior

Área de cobertura: 15.4" en el modelo de portátiles y 22"para monitores

CPU: Pentium 500MHz o superior

Memoria: 512 MB o superior.

Fuente de alimentación: DC 5V (USB)

Voltaje del lápiz: 3.7V o 4.5V o 1.2V según el modelo

Tiempo de uso continuo: de 16 a 20 horas según el modelo

Patentes Relacionadas

Se listan a continuación:

"Digital Pen" Pat No.: US 2007/0126716 A1

"Method and system for digitizing handwriting" Pat. No. 5,977,958

"Radiolocation system having writing pen application" Pub. No.: US 2003/0071754 A1

"Writing device for storing handwriting" Pat. No.: US 6,348,914 B1

"Determining the location of the tip of an electronic stylus" Pub. No.: US 2004/0160429 A1

"Electronic light pen system" Pat. No.: US 6,377,249 B1

"Electronic pen device" Pat. No.: US 6,188,392 B1

"Electronic pen input device and coordinate detecting method therefore" Pat. No.: US 6,897,854 B2

"Electronic pen" Pat. No.: US D501,881 S

"Electronic portable apparatus and method" Pat. No.: US 6,577,299 B1

"Electronic stylus with writing feel" Pat. No.: 5,627,348

"Pen positioning system" Pat. No.: US 6,184,873 B1

"Systems and methods for providing a combined pen and mouse input" Pub. No.: US 2006/0250380 A1

"Electronic pen holding" Pub. No.: US 2006/0176288 A1

"Ultrasonic coordinate input apparatus" Pub. No.: US 5,637,839

Anexo G

Esquemáticos y Layouts

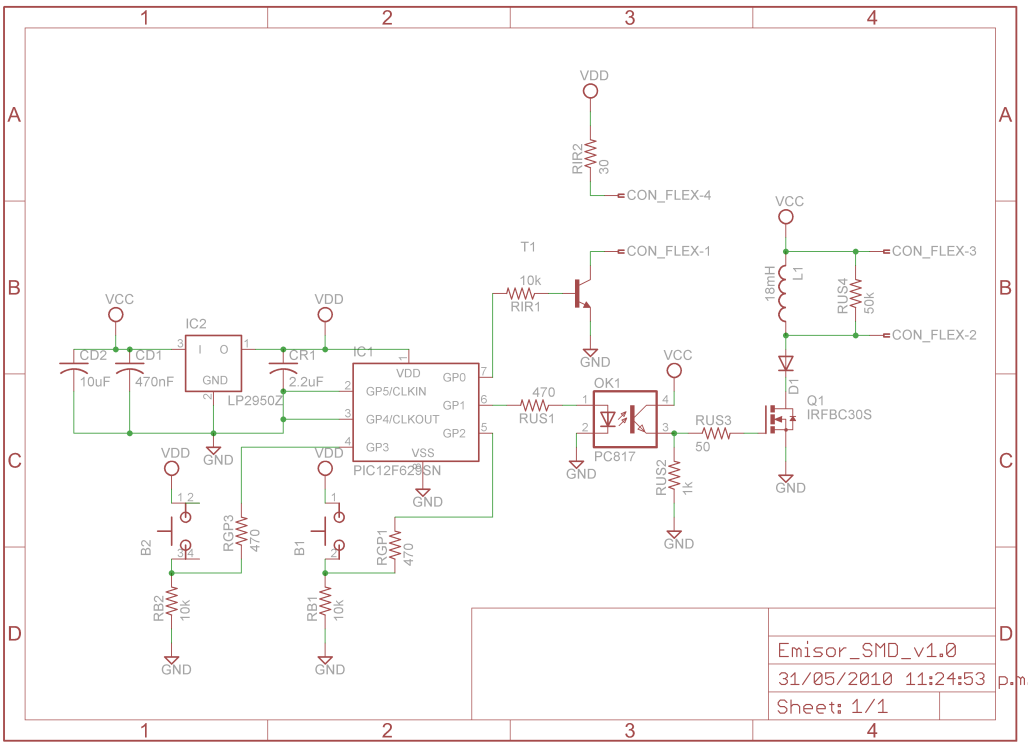

Figura G.1: Esquemático de Emisor en SMD

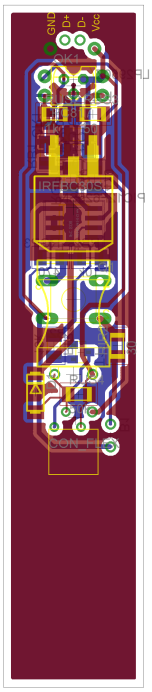

Figura G.2: Layout de Emisor en SMD

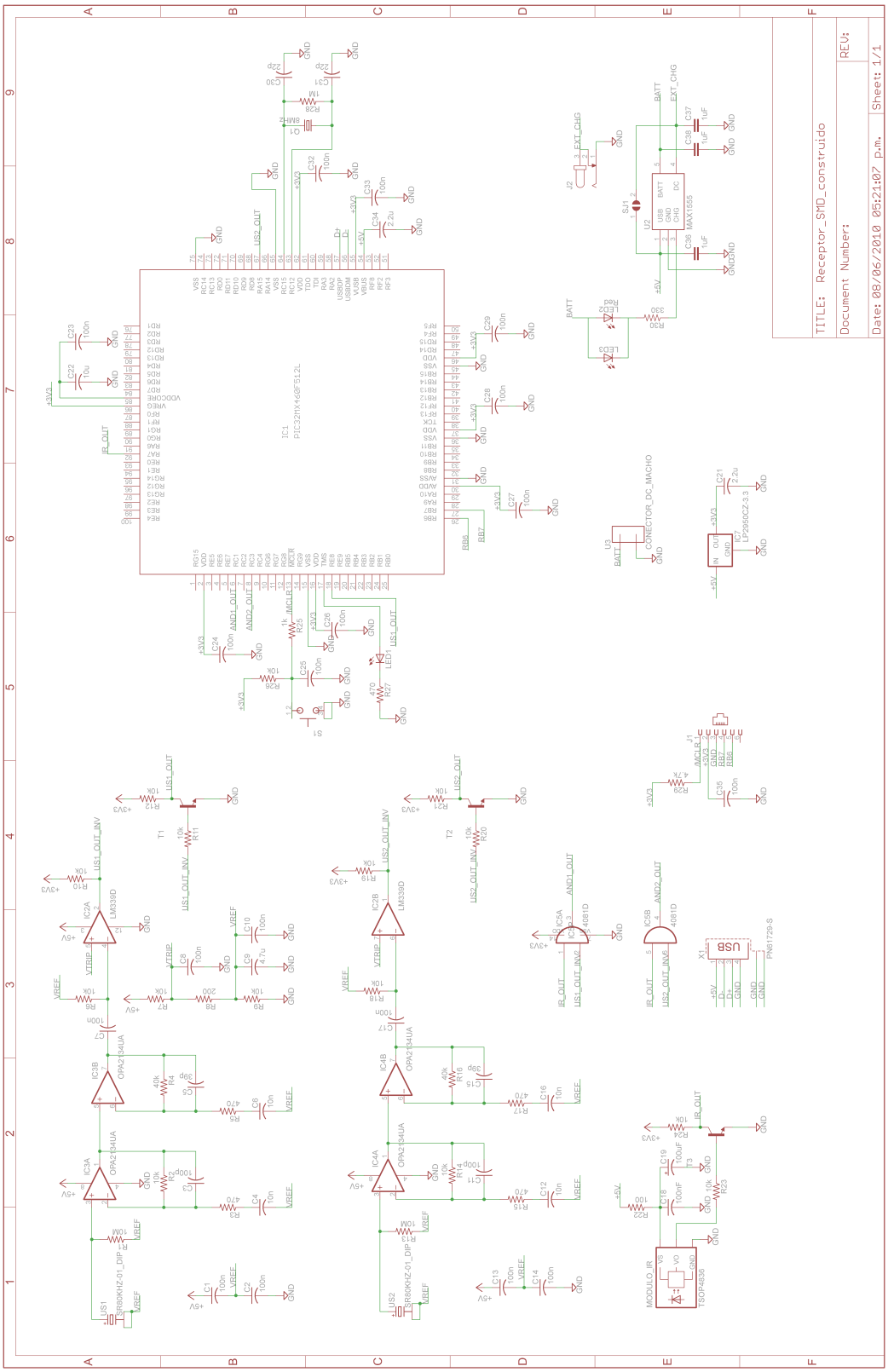

Figura G.3: Esquemático de Receptor en SMD construido

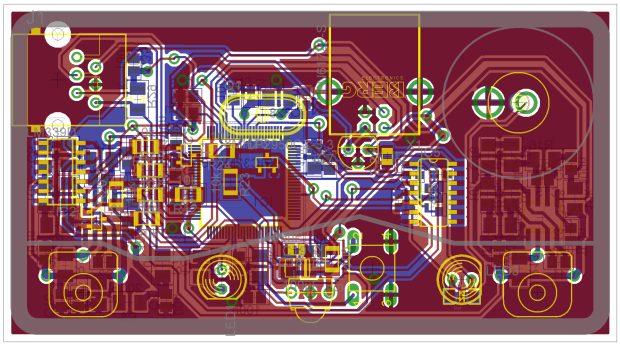

Figura G.4: Layout de primer versión de Receptor en SMD

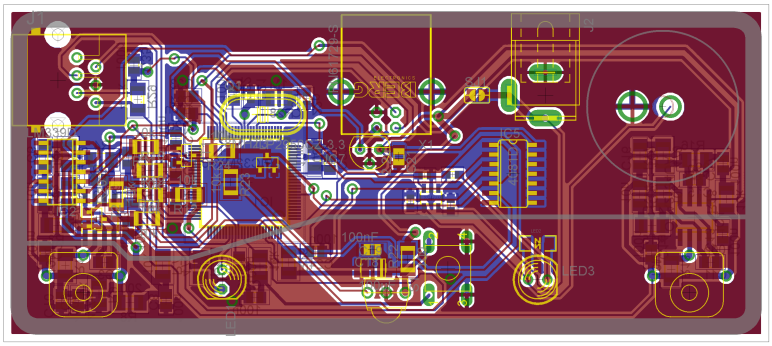

Figura G.5: Layout de Receptor en SMD construido

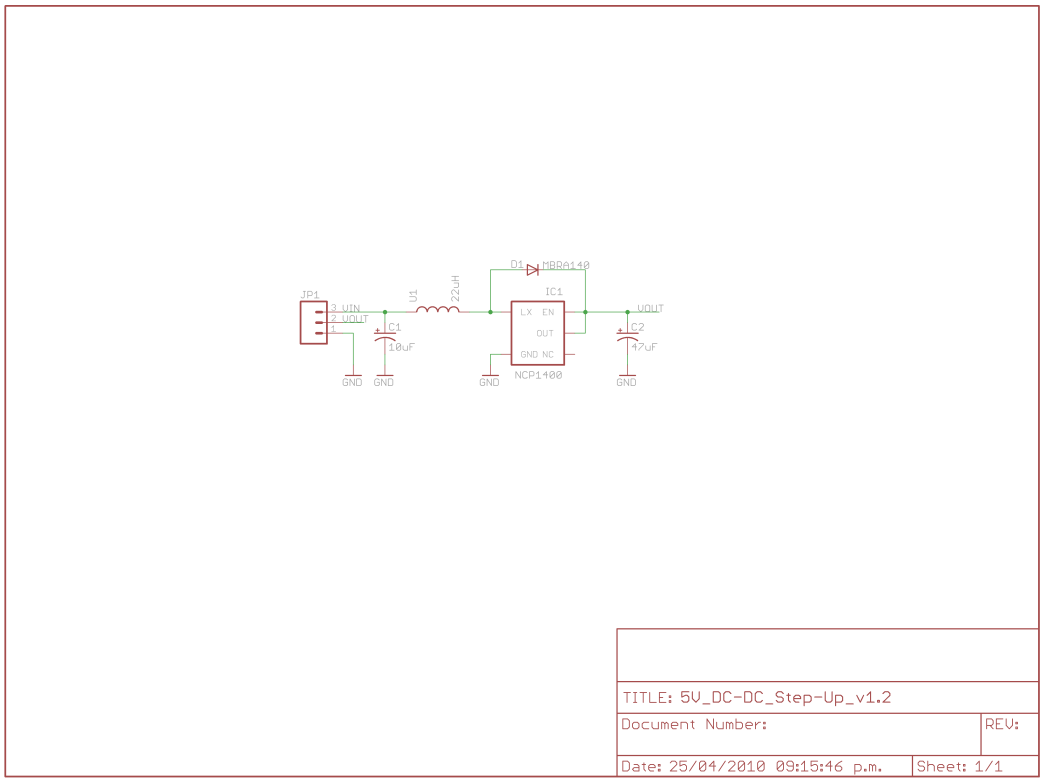

Figura G.6: Esquemático de DC/DC Step-up de 5V

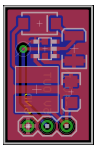

Figura G.7: Layout de DC/DC Step-up de 5V

Anexo H

Contenido del CD

Requerimientos del sistema:

- lectora de cd-rom de 8x o superior
- Microsoft Windows 2000 o superior
- pantalla de 16-bit de color con resolución de 800 x 600
- audio de 16-bit

Contenido del CD:

CD/

Articulo_Lapix.pdf
LEEME
Poster_Lapix.cpt
Poster_Lapix.jpg
Proyecto_Lapix.pdf

CD/archivos_latex/

CD/archivos_latex/articulo/*

CD/archivos_latex/documentacion/*

CD/circuitos/

CD/circuitos/esquematicos_y_layouts/

5V_DC-DC_Step-Up_v1.2.brd
5V_DC-DC_Step-Up_v1.2.sch
Emisor_primer_prototipo_v1.3.lvw
Emisor_primer_prototipo_v2.1.pcb
Emisor_SMD_v1.0.brd
Emisor_SMD_v1.0.sch
LEEME
Receptor_primer_prototipo_v1.6.lvw
Receptor_primer_prototipo_v3.1.pcb
Receptor_SMD_construido.brd
Receptor_SMD_construido.sch
Receptor_SMD_primer_version.brd
Receptor_SMD_primer_version.sch

CD/circuitos/fotos/

5V_DC-DC_Step-Up_abajo.jpg
5V_DC-DC_Step-Up_arriba.jpg
5V_DC-DC_Step-Up_costado.jpg
Conector_hembra_emisor1.jpg
Conector_hembra_emisor2.jpg
Conector_hembra_emisor3.jpg
Conector_hembra_emisor4.PNG
Conector_macho_receptor.PNG
Emisor_SMD_abajo2.jpg
Emisor_SMD_abajo3.jpg
Emisor_SMD_abajo.jpg
Emisor_SMD_arriba2.jpg
Emisor_SMD_arriba.PNG
Receptor_SMD_abajo.JPG
Receptor_SMD_arriba.jpg
Receptor_SMD_costado.jpg
Receptor_SMD_frente.jpg

CD/circuitos/imagenes/

5V_DC-DC_Step-Up_v1.2_esquematico.png
5V_DC-DC_Step-Up_v1.2_layout.png
Emisor_SMD_v1.0_esquematico.png

Emisor_SMD_v1.0_layout.png
Receptor_SMD_construido_esquematico.png
Receptor_SMD_construido_layout.png
Receptor_SMD_primer_version_esquematico.png
Receptor_SMD_primer_version_layout.png

CD/documentos/

Documentacion_del_primer_Hito.pdf
Documentacion_del_segundo_Hito.pdf
Eleccion_del_chip_para_el_receptor.pdf
Lista_de_componentes.pdf
Lista_de_componentes_necesarios.pdf
Lista_de_componentes_y_precios.xls
Plan_Del_Proyecto.pdf

CD/fotos_del_prototipo/

Carga_bateria1.jpg
Carga_bateria2.jpg
Carga_bateria3.jpg
Carga_bateria4.jpg
Carga_bateria5.jpg
Carga_bateria6.jpg
Carga_bateria7.jpg
Carga_bateria8.jpg
Emisor_desarmado1.jpg
Emisor_desarmado2.jpg
Emisor_desarmado3.jpg
Emisor_desarmado4.jpg
Encapsulado_Emisor1.jpg
Encapsulado_Emisor2.jpg
Encapsulado_Emisor3.jpg
Encapsulado_Emisor4.jpg
Encapsulado_Emisor5.jpg
Receptor_encapsulado1.jpg
Receptor_encapsulado2.jpg
Receptor_encapsulado3.jpg
Receptor_encapsulado4.jpg
Uso_fuera_de_pantalla.jpg
Uso_sobre_pantalla1.jpg

Uso_sobre_pantalla2.jpg
Uso_sobre_pantalla3.jpg
Uso_sobre_pantalla4.jpg
Uso_sobre_pantalla5.jpg

CD/hojas_de_datos/

CD/hojas_de_datos/Alimentacion/

5N20V.pdf
DW01.pdf
MAX1551-MAX1555.pdf
NCP1400_5V_Step-Up_v11.pdf
NCP1400A-D.pdf
PCB_protection_for_3,7V_of_Li-Ion_Battery2.pdf
PCB_protection_for_3,7V_of_Li-Ion_Battery.pdf

CD/hojas_de_datos/Emisor/

1N4937.pdf
09P-183J-50.pdf
BC547.pdf
HLW4S-2C7LF.pdf
IRFBC30.pdf
IRFDC20.pdf
KB817-847.pdf
LP2950CZ3-3.pdf
MMBT3904.pdf
PC817.pdf
PIC12F629_675.pdf
PT80KHZ-01.pdf
S2J-E3-52T.pdf

CD/hojas_de_datos/Receptor/

BPW83.pdf
HCF4081B.pdf
LM339.pdf
OPA2134.pdf
PIC32MX3XX-4XX.pdf
SR80KHZ-01.pdf

TSOP4836.pdf

CD/pruebas/

CD/pruebas/obtencion_de_especificaciones/

pruebas_hechas_a_pegasus.txt

CD/pruebas/obtencion_de_especificaciones/Capturas_con_pegasus/

prueba01.JPG
prueba02.JPG
prueba03.JPG
prueba04.jpg
prueba05(filmación2009-03-10_15.20).jpg
prueba05(filmación2009-03-10_15.25).jpg
prueba06(delay).jpg

CD/pruebas/obtencion_de_especificaciones/filmaciones_de_escritura/

2009-03-03_15.11.avi
2009-03-03_15.15.AVI
2009-03-03_15.50.AVI
2009-03-10_15.20.avi
2009-03-10_15.21.avi
2009-03-10_18.00.AVI

CD/pruebas/Pruebas_de_funcionamiento/

25092010009.AVI
25092010010.mp4
25092010011.AVI
25092010012.mp4

CD/pruebas/prueba_de_retardo/

25092010013.AVI
25092010014.AVI
25092010015.AVI

CD/referencias/

41191d-PIC12F629-675-PIC16F630-676_Memory_Programming.pdf
Dossier_PNF_DUO.pdf
Explorer_16_Development_Board_User's_Guide.pdf
Multimetro_Microtek_dt92.pdf
Pegasus_P220_Specifications.pdf
protection_modules.pdf

CD/software/

CD/software/Driver

LEEME

CD/software/Driver/Driver_tot_v1.51_para_muestras/

lapix_usb.c
lapix_usb.h
main.c
Makefile

CD/software/Driver/Driver_tot_v6.11_para_muestras/

lapix_usb.c
lapix_usb.h
main.c
Makefile

CD/software/Driver/Driver_v7.0/

lapix_usb.c
lapix_usb.h
main.c
Makefile

CD/software/Driver/scripts/

Iniciar_LAPIX
lapix.sh

CD/software/PIC12

LEEME

CD/software/PIC12/PIC12_v6.4/*

CD/software/PIC12/PIC12_v7.0/*

CD/software/PIC32

LEEME

CD/software/PIC32/PIC32_V5.0/*