****

**Vázquez González Erick**

**ESIME Zacatenco - 7CV2**

**Prof. Pérez macías cesar israel**

**México D.F. a 17 de noviembre de 2014**

**Transmisión y Recepción Digital de voz**

**10 Páginas**

**Ingeniería en Comunicaciones y Electrónica**

**Academia de Comunicaciones**

**PROYECTO**

**TRANSMISIÓN Y RECEPCIÓN DE VOZ DIGITAL**

**OBJETIVO**

Captar, Digitalizar, procesar, transmitir, decodificar, recibir y reproducir voz de manera inalámbrica

**MARCO TEORICO**

El rango de frecuencias en los que la voz se escucha se muestra en la siguiente imagen.

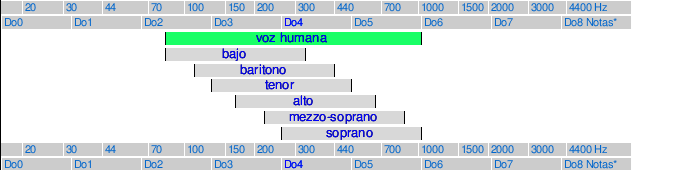
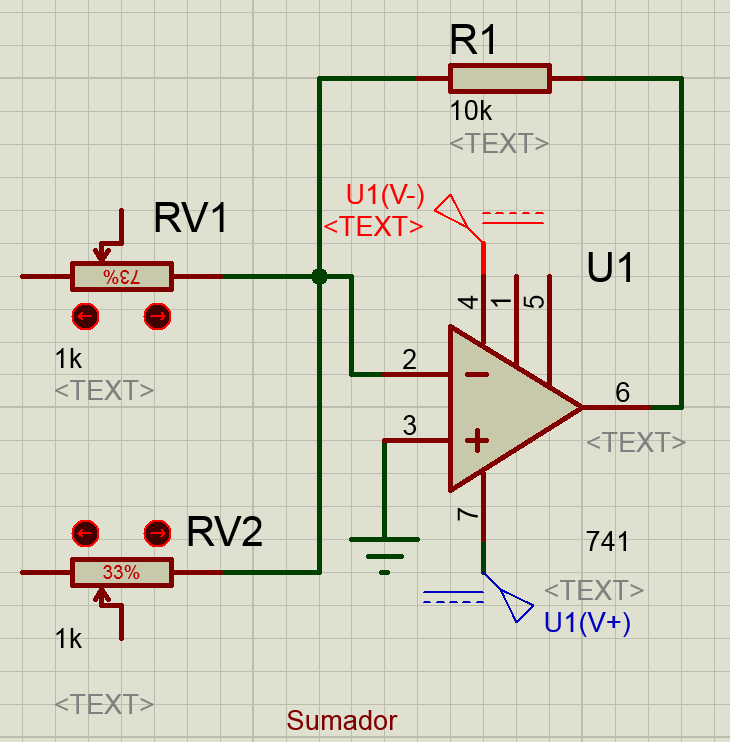


Figura 1: Rangos de frecuencia de la voz humana.

Para tomar en cuenta la tabla anterior, debe saberse que la población de hombres y mujeres en México tienen tonos medios es decir; la tesitura de mujeres: mezzo-soprano y de hombres: tenor.

Ahora bien, el audio es una onda senoidal, por lo que si tenemos en cuenta que existen valores negativos y positivos, entonces hay que hacer una modificación a la señal para que esta sólo adquiera valores positivos que entren al microcontrolador. Esto se logra con un offset.

El offset se genera por la configuración de sumador de un opamp.

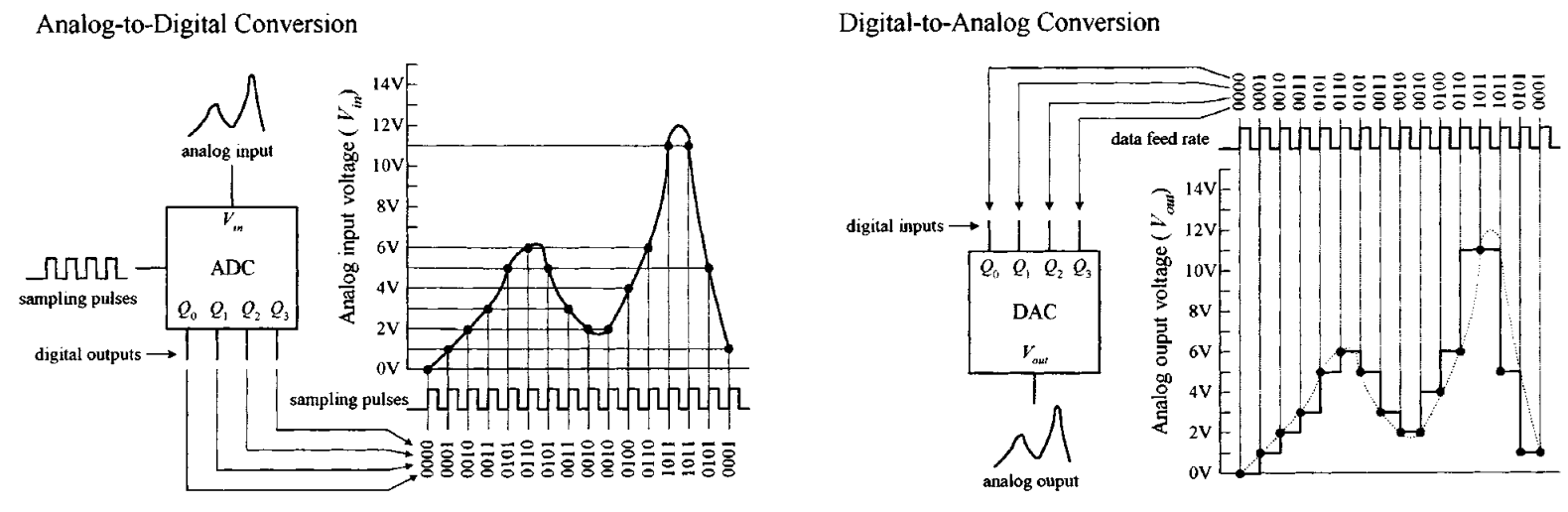
La señal de salida del opamp puede ser la entrada de un microcontrolador para ser convertida en digital, a partir del ADC interno.

Para tomar en cuenta la comunicación inalámbrica, hay que recordar la comunicación serial; sin embargo para que la comunicación inalámbrica sea eficiente pueden compararse los módulos de Xbee y de Bluetooth; como se muestra en la siguiente tabla.



Con referencia en la tabla anterior podemos concluir que en consumo y costo los módulos de bluetooth son más económicos.

La teoría del muestreo demuestra que la reconstrucción exacta de una señal periódica continua en banda base a partir de sus muestras, es matemáticamente posible si la señal está limitada en banda y la tasa de muestreo es superior al doble de su ancho de banda



Existen muchas maneras de muestrear una señal, la más común es el muestreo periódico o uniforme. Este proceso se describe mediante la relación

𝑥(𝑛)= 𝑥𝑎(𝑛𝑇)

Donde x(n) es la señal en tiempo discreto obtenida tomando muestras de la señal analógica 𝑥(𝑛𝑇) cada T segundos.

El muestreo periódico establece una relación entre las variables *t* de tiempo continuo y *n* de tiempo discreto. De hecho, estas variables se relacionan linealmente a través del periodo de muestreo T o equivalentemente, a través de la velocidad de muestreo como

𝑡=𝑛𝑇=n𝑇

Como consecuencia de esta expresión, existe una relación entre la variable frecuencia F de las señales analógicas y la variables frecuencia f de las en tiempo discreto. Para establecer dicha relación si se considera una señal analógica de la forma

𝑥(𝑡)=𝐴𝑐𝑜𝑠(2𝜋𝐹+𝛳)

que, cuando se muestrea periódicamente a una velocidad de 𝐹𝑠=1𝑇 muestras por segundo, da lugar a

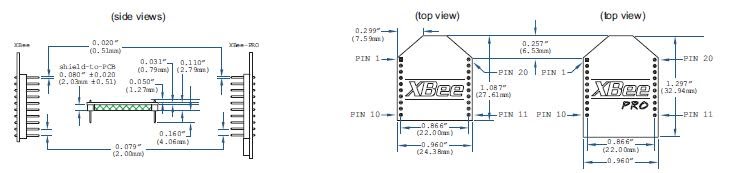
𝑥(𝑡)=𝐴𝑐𝑜𝑠(2𝜋𝐹+𝛳) 𝑥(𝑛)=𝐴cos(2𝜋𝐹𝐹𝑆+𝛳)

Si una señal en tiempo discreto es expresada como (𝑡)=𝐴𝑐𝑜𝑠(2𝜋𝑛𝑓+𝛳)

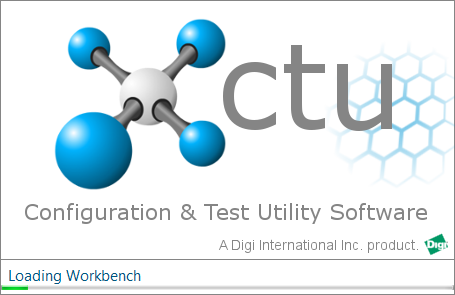
entonces, al comparar la relación de la primera y segunda ecuación obtenemos las variables de frecuencia *F* y *f* las cuales están linealmente relacionadas como 𝑓=𝐹/𝐹𝑠

**DESARROLLO**

El proyecto se realizara con ambos, módulo xbee atravez de otro controlador xbee y modulo bluetooth controlado atravez de una aplicación de Android



Para poder configurar los módulos xbee es requerido un cable USB- FTDI , o en su lugar un arduino uno, en el cual se cortocircuitan los pines RESET con el pin GND, de esta manera el arduino se salta al micro ATMEGA328p permitiéndonos utilizar la tarjeta PCB para configurar el Xbee atravez de USB



Descargaremos el software desarrollado para configurar el Xbee de la siguiente página

<http://www.digi.com/support/productdetail?pid=3352&type=utilities>

Este software nos permitirá configurar nuestros módulos Xbee

Una vez con la conexión serial establecida

Se configuraran los Xbee 1 como coordinador en modo AT y otro como Router en modo AT ambos en modo transparente

**Modo Transparente**

En este modo todo lo que ingresa por el pin 3 (Data in), es guardado en el buffer de entrada y luego transmitido y todo lo que ingresa como paquete RF, es guardado en el buffer de salida y luego enviado por el pin 2 (Data out). El modo Transparente viene por defecto en los módulos Xbee.

Este modo está destinado principalmente a la comunicación punto a punto, donde no es necesario ningún tipo de control. También se usa para reemplazar alguna conexión serial por cable, ya que es la configuración más sencilla posible y no requiere una mayor configuración

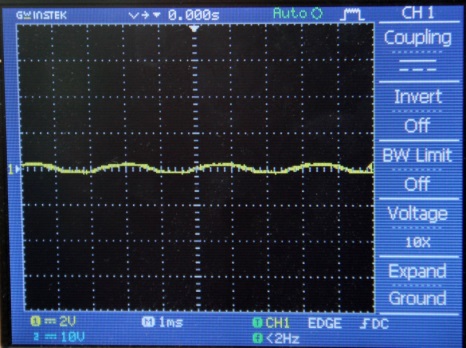
Los pines del Xbee son demasiado pequeños para utilizarse en una protoboard, por lo cual se requirió conseguir adaptadores para protoboard en electronicastudio.com con un local ubicado en el centro de la ciudad de México

El arduino se configurara en modo reset para poder “saltarnos” el microcontrolador del mismo y poder utilizar el dispositivo para configurar nuestros módulos Xbee

En los parámetros más relevantes están

* PANID, Personal área Network, debe ser el mismo valor para ambos **3332**
* Configurar Xbee1 como Coordinador
* Configurar Xbee2 como EndPoint

**Captura de audio**

La amplitud de una señal de audio puede ser de 2V, sin embargo alcanza su máximo voltaje en 2V y su mínimo en -2v esto es un problema ya que los microcontroladores solo pueden medir voltajes entre 0 y 5v, si intentáramos medir los voltajes negativos el microcontrolador solo leerá 0v y terminaríamos cortando la parte negativa de la señal , sin embargo, es posible amplificar y aplicar un offset a señales de audio para que estén colocadas entre este rango de 0 a 5v, idealmente la colocaremos en 2.5v para que su voltaje mínimo sea 0v y su máximo 5v

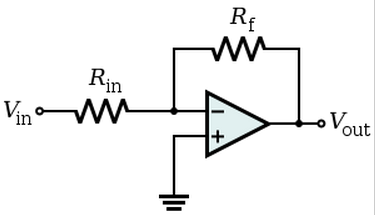


La siguiente imagen muestra la salida de audio de un tono, la señal es relativamente débil solo 200mV esta señal debe ser amplificada al voltaje que queremos (2.5v) Amplificación significa incrementar la amplitud (La distancia máxima entre el 0 y su máximo y mínimo de la señal )

La siguiente señal esta amplificada sin embargo aún se encuentra acoplada en 0 oscilando entre -2.5v y 2.5v por lo que para corregir esto tendremos que aplicar un Offset de CD, que significa cambiar el voltaje promedio de la onda

La siguiente imagen muestra la señal con su voltaje central en 2.5v en lugar de 0v, es decir la señal nunca lleva a 0v , esta señal está preparada para ser recibida por el puerto analógico de nuestro microcontrolador

**Amplificador no inversor**

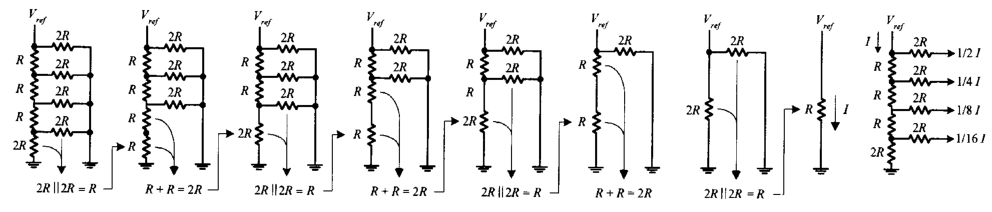


El amplificador es la primera parte del circuito, incrementara la amplitud de la señal de 200mV a 2.5v (idealmente), la otra función del amplificador es proteger la fuente del audio, (lo que sea que genere el audio, teléfono, pc, radio, etc…) del resto del circuito, La señal amplificada requerirá toda la corriente del amplificador , así que cualquier carga puesta después en el cto no será “sentida” por la fuente, utilizaremos nuestros OP AMPS en configuración no inversora

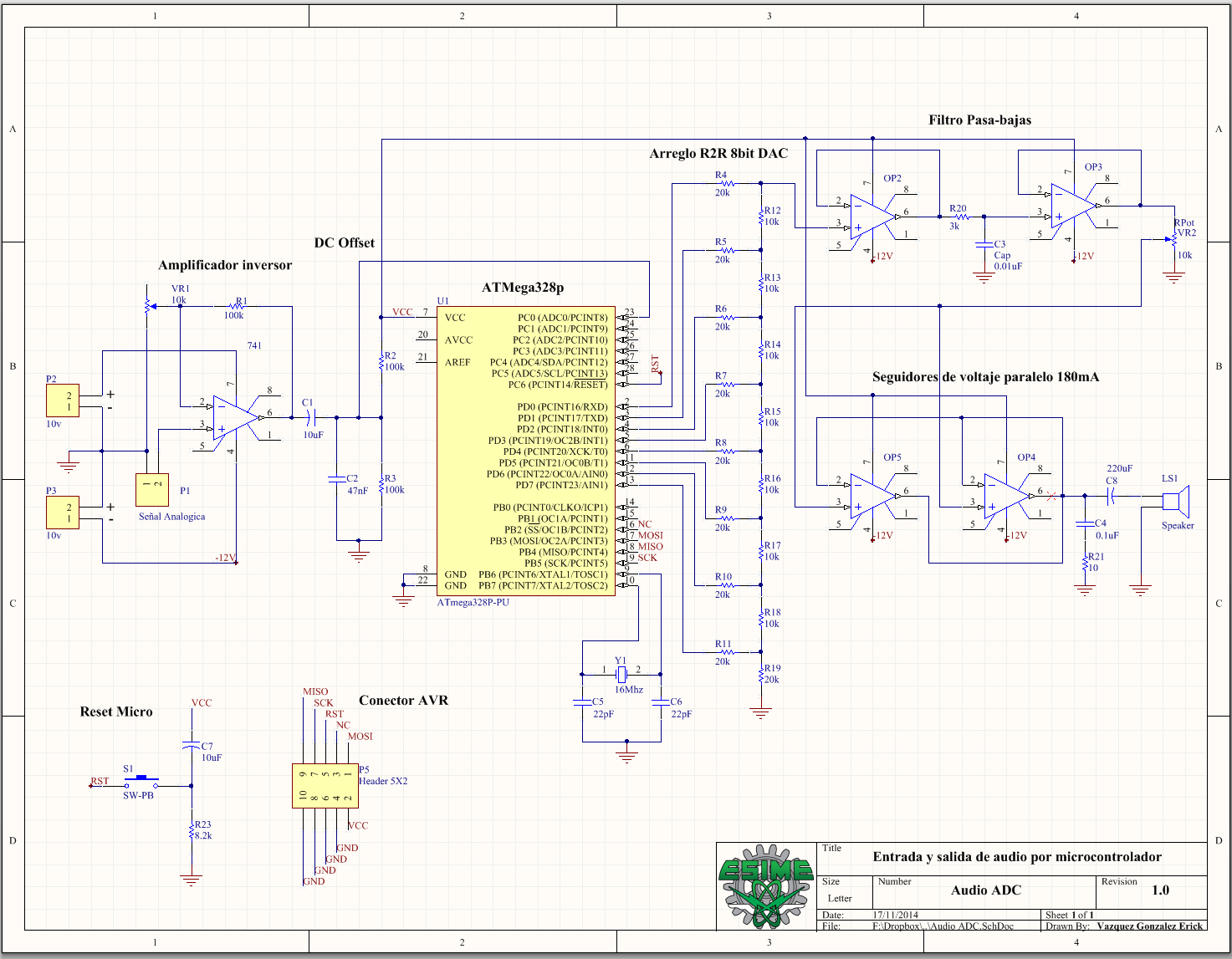
Alimentaremos nuestros OP AMPS con +-10v debido a que la señal no será amplificada nunca más allá de 2.5v utilizaremos un divisor de voltaje para poder obtener 5v para alimentar nuestro microprocesador, así como un capacitor bypass

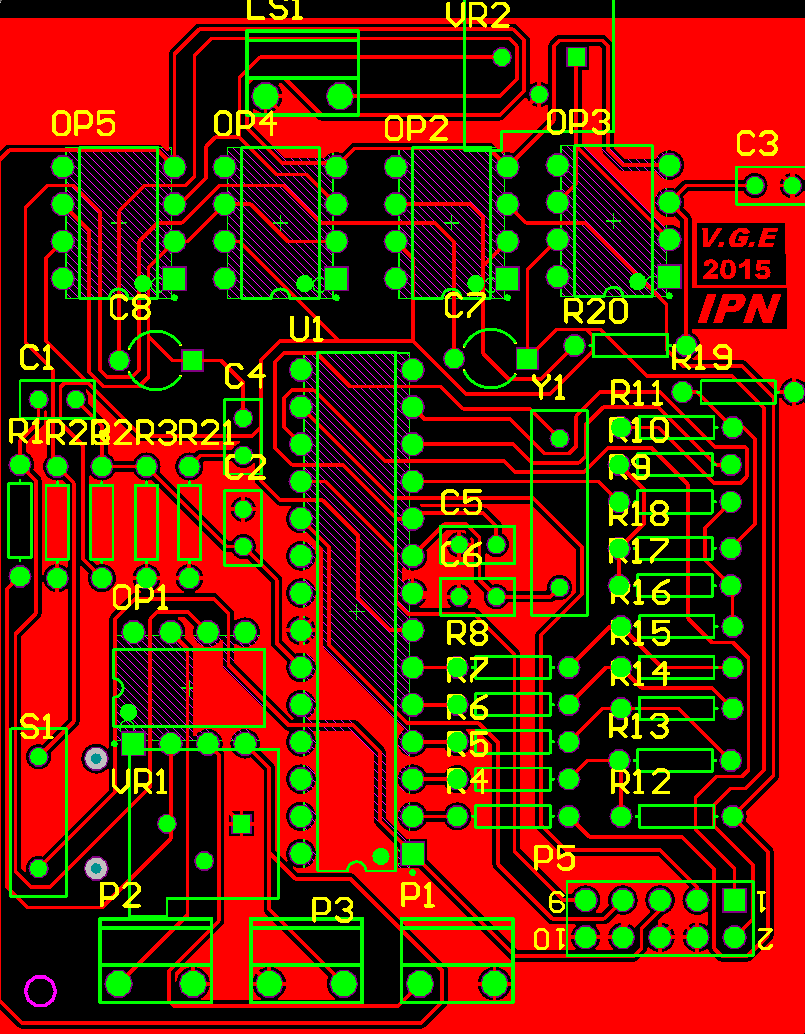
**Arreglo R/2R Escalera ADC**

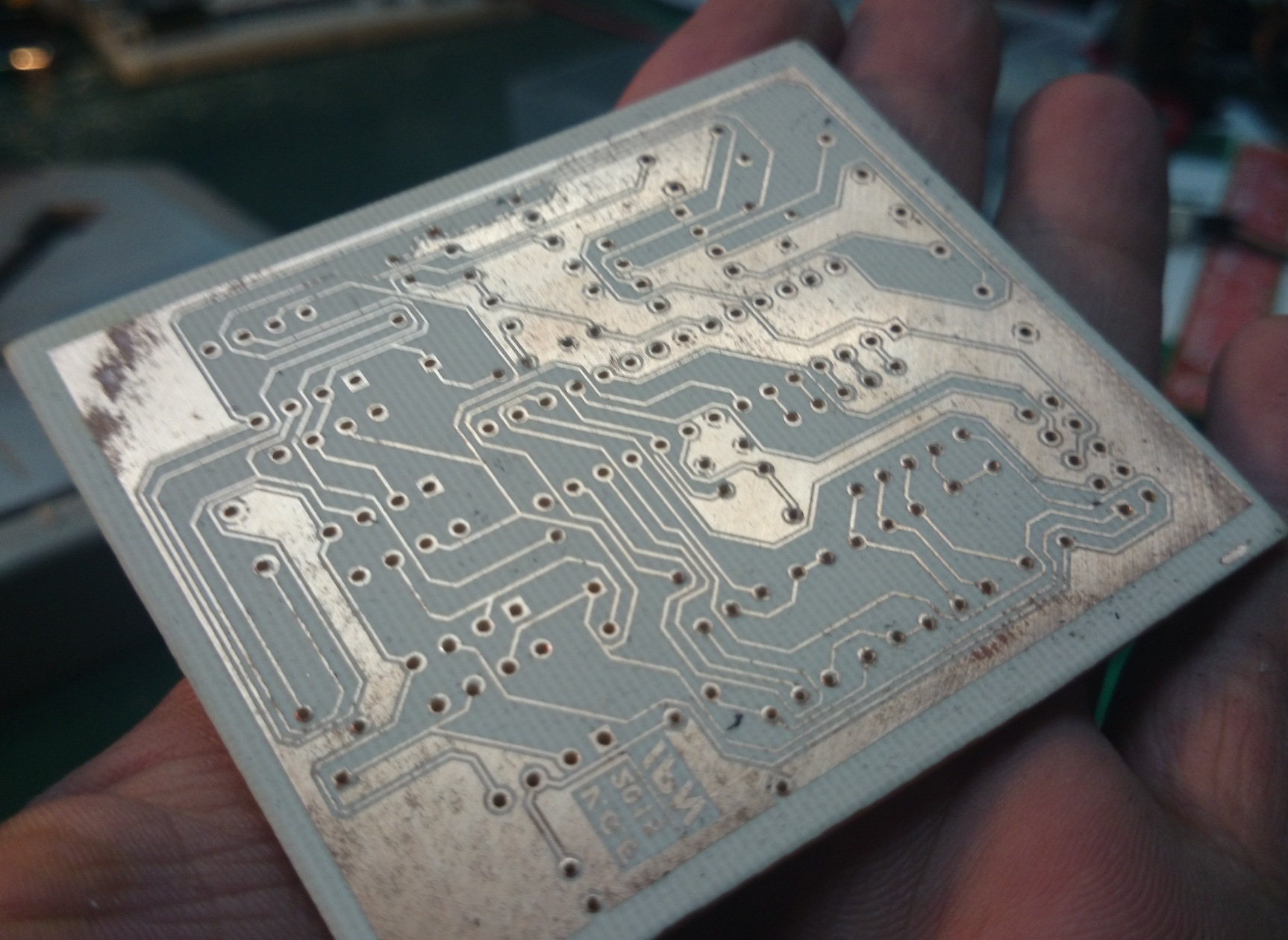
Un arreglo de este tipo utiliza una escalera de resistencias utilizando solo 2 valores R y 2R la figura muestra un arreglo simple de 4 bits , por ahora asumiremos que switches lógicos controlaran las entradas, la corriente siempre es la misma no importa si se activa un switch o varios , la corriente fluye a tierra si el switch se apaga, , sin embargo si este prende fluirá hacia una tierra virtual de la terminal no inversora de un OPAMP el OPAMP convertirá la señal no inversora en 0v vía retroalimentación negativa



**DISEÑO**







**LISTA DE MATERIALES (BOM)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| En 17 de Noviembre del 2014 | | |  |  |  |
|  |  |  |  |  |  |
| **Etiqueta** | **Paquete** | **Cant.** | **Componentes** | **Descripción** | **Costo (USD)** |
|  |  |  |  |  |  |
| 10 | AXIAL-0.3 | 1 | R21 | Resistencia | $0.11 |
| 100k | AXIAL-0.3 | 3 | R1, R2, R3 | Resistencia | $0.33 |
| 10k | AXIAL-0.3 | 7 | R12, R13, R14, R15, R16, R17, R18 | Resistencia | $0.88 |
| 12v | Conector | 2 | P2, P3 | Header, 2-Pin | $2.00 |
| 16Mhz | RAD-0.4 | 1 | Y1 | Cristal Oscilador | $1.20 |
| 20k | AXIAL-0.3 | 9 | R4, R5, R6, R7, R8, R9, R10, R11, R19 | Resistencia | $0.99 |
| 3k | AXIAL-0.3 | 1 | R20 | Resistencia | $0.11 |
| 741 | DIP-8 | 1 | OP1 | FET OPAMP | $1.00 |
| 8.2k | AXIAL-0.3 | 1 | R23 | Resistencia | $0.11 |
| ATmega328P | DIP28 | 1 | U1 | 8-bit AVR uC | $2.82 |
| Cap | RAD-0.1 | 5 | C2, C3, C4, C5, C6 | Capacitor | $1.52 |
| Cap2 | CAPR5-4X5 | 2 | C7, C8 | Capacitor | $0.30 |
| Cap2 | RAD-0.1 | 1 | C1 | Capacitor | $0.20 |
| Header 5X2 | HDR2X5 | 1 | P5 | Header, 5-Pin, Doble Fila | $1.50 |
| Op Amp | DIP-8 | 4 | OP2, OP3, OP4, OP5 | FET OPAMP | $4.00 |
| RPot | VR4 | 2 | VR1, VR2 | Potenciometro | $1.50 |
| Señal Analogica | Conector | 1 | P1 | Header, 2-Pin | $2.00 |
| Speaker | Conector | 1 | LS1 | Loudspeaker | $3.00 |
| SW-PB | RAD-0.3 | 1 | S1 | Switch | $0.50 |

**CODIGO**

int incomingAudio;

void setup(){

for (byte i=0;i<8;i++){

pinMode(i,OUTPUT); //Pins 0-7 como salidas(DAC)

}

}

void loop(){

incomingAudio = analogRead(A0); //Lee la tension en A0

incomingAudio = (incomingAudio+1)/4 - 1; //Escala de 10 bit (0-1023) a 8 bit (0-255)

if (incomingAudio<0){ //Se encarga de valores negativos

incomingAudio = 0;

}

PORTD = incomingAudio;

}

**BIBLIOGRAFIA**

Scherz, Paul. *Practical Electronics for Inventors*. Vol. 1. New York: McGraw-Hill, 2000. Print.