**Simulación de modulación por amplitud de onda (oscilograma y espectro de frecuencia)**

**Objetivo**

Comprender los conceptos relacionados con modulación por amplitud y mostrar cómo es la representación en tiempo y frecuencia de la señal modulada.

**Introducción**

Las señales de transmisión se transportan entre un transmisor y un receptor a través de alguna forma de medio de transmisión. Sin embargo, casi nunca tienen las señales de información una forma adecuada para su transmisión. En consecuencia, se deben transformar a una forma más adecuada. El proceso de imprimir señales de información de baja frecuencia en una señal portadora de alta frecuencia se llama modulación.

La modulación de amplitud (AM, por amplitude modulation; en español se usa “amplitud modulada”) es el proceso de cambiar la amplitud de una señal portadora de frecuencia relativamente alta, en proporción con el valor instantáneo de la señal modulante o moduladora (información). La modulación de amplitud es una forma de modulación relativamente poco costosa y de baja calidad, que se usa para emisiones comerciales de señales de audio y de video. También se usa para radiocomunicaciones móviles en dos sentidos, como por ejemplo los radios de banda civil (CB). Los moduladores de AM son dispositivos no lineales, con dos entradas y una salida. Una entrada es una sola señal portadora de alta frecuencia y amplitud constante, y la segunda está formada por señales de información, de frecuencia relativamente baja, que puede tener una sola frecuencia, o ser una forma compleja de onda, formada a su vez por muchas frecuencias. Las frecuencias que son lo suficientemente altas como para irradiarse en forma eficiente de una antena, y propagarse por el espacio libre se suelen llamar radiofrecuencias, o simplemente RF. En el modulador, la información actúa sobre, o modula, la portadora de RF y produce una forma modulada de onda.

*Arduino*

Arduino es una plataforma de prototipos electrónica de código abierto (open-source) basada en hardware y software flexibles y fáciles de usar. Está pensado para artistas, diseñadores, como hobby y para cualquiera interesado en crear objetos o entornos interactivos.

Arduino puede sentir el entorno mediante la recepción de entradas desde una variedad de sensores y puede afectar a su alrededor mediante el control de luces, motores y otros artefactos. El microcontrolador de la placa se programa usando el Arduino Programming Language (basado en Wiring) y el Arduino Development Environment (basado en Processing). Los proyectos de Arduino pueden ser autónomos o se pueden comunicar con software en ejecución en un ordenador (por ejemplo, con Flash, Processing, MaxMSP, etc.).

**Explicación**

Basados en la practica 3 “Modulación en amplitud y frecuencia”, en concreto a su trabajo previo, se nos ocurrió un simulador de modulación en amplitud.

El principio es simple, ajustar una señal con información a una señal de alta frecuencia para que así la información pueda ser transmitida a largas distancias.

La modulación, en simples palabras, alterar un parámetro de una señal para que pueda ser transportada. En el caso de la modulación de amplitud es, perdone la redundancia, la amplitud. Con este principio simple se determina la siguiente fórmula para obtener la señal modulada:

Donde:



Aquí gráficamente se puede observar que ya en la señal modulada la amplitud de la señal portadora se ve afectada por el comportamiento de la amplitud de la moduladora a través del tiempo.

Es obvio ver que, si la amplitud de la señal portadora es menor a la de la moduladora, también habrá cosas raras; en este caso, se llama sobremodulación. Para evitar esto se utiliza un índice de modulación dado por:

Una modulación ideal es con un índice de modulación de 80%.

En cuestión de frecuencia, el espectro de una señal modulada está conformado por el espectro de la portadora como fp en su posición natural acompañada de dos frecuencias, una posicionada en fp – fm y otra en fp + fm.

Así que en el espectro una señal modulada siempre se espera una frecuencia central y dos señales idénticas, una del lado izquierdo de la central y otra a su lado derecho, a la misma distancia de separación.

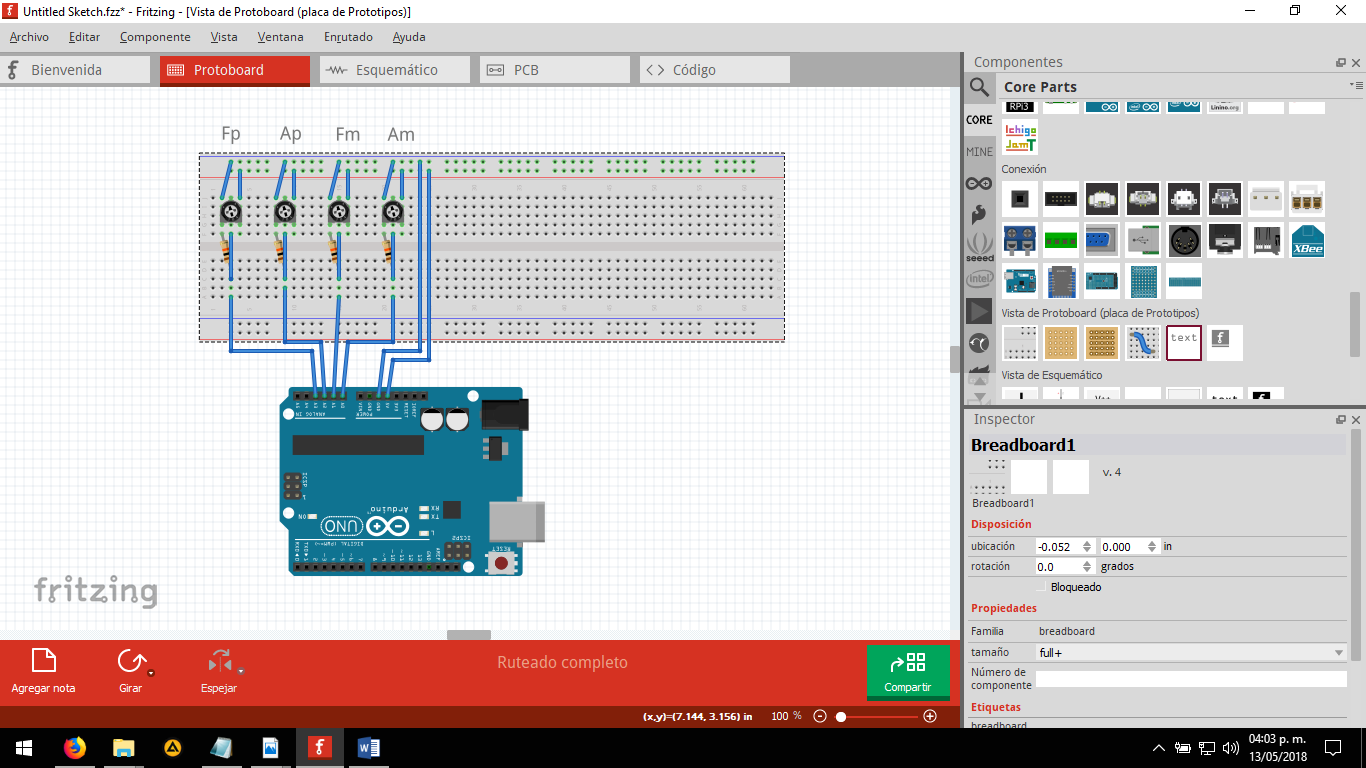


El ancho de banda de esta señal se puede determinar:

Que va de

Desarrollo del proyecto

Se realizo un programa desarrollado en java con conexión a Arduino para su control, que recibe los para metros de amplitud y frecuencia de una señal moduladora y de una señal portadora. La introducción de estos parámetros es a través de perillas (potenciómetros) que indican un valor.

Para mejor comprensión les presentamos el diagrama del circuito. 

Donde Am es la amplitud de la señal moduladora, Fm es la frecuencia de la señal moduladora, Ap es la amplitud de la señal portadora y Fp es la frecuencia de la señal portadora.

El programa (ProyectoSC1) que controla al Arduino hecho en la plataforma es el siguiente:

void setup() {

Serial.begin(9600);

}

void loop() {

int ampm =(int) map(analogRead(0),0,1023,0,128);

double fm = map(analogRead(1),0,1023,0,5);

int ampp = (int) map(analogRead(2),0,1023,0,128);

double fp = map(analogRead(3),0,1023,0,50);

String tor = "/";

String p = ampm +tor+ fm +tor + ampp+tor + fp;

Serial.println(p);

delay(100);

}

En la sección setup invocamos un puerto serie que enviará los parámetros obtenidos por las perillas hacía al programa java.

En la sección loop, que se repetirá hasta que desconectemos el circuito Arduino, reconocerá los valores de las entradas analógicas 0, 1, 2 y 3 como amplitud y frecuencia de la moduladora y amplitud y frecuencia de la portadora, respectivamente. En una sección se concatenan para ser enviados a través del puerto serie para su manipulación posterior.

Que quede claro, el programa se puede manipular para la conveniencia del que lo utilice.

Para el programa java, el cuál nombramos ProyectoSC porque no estábamos seguros de que fuera a funcionar, implementa librerías externas:

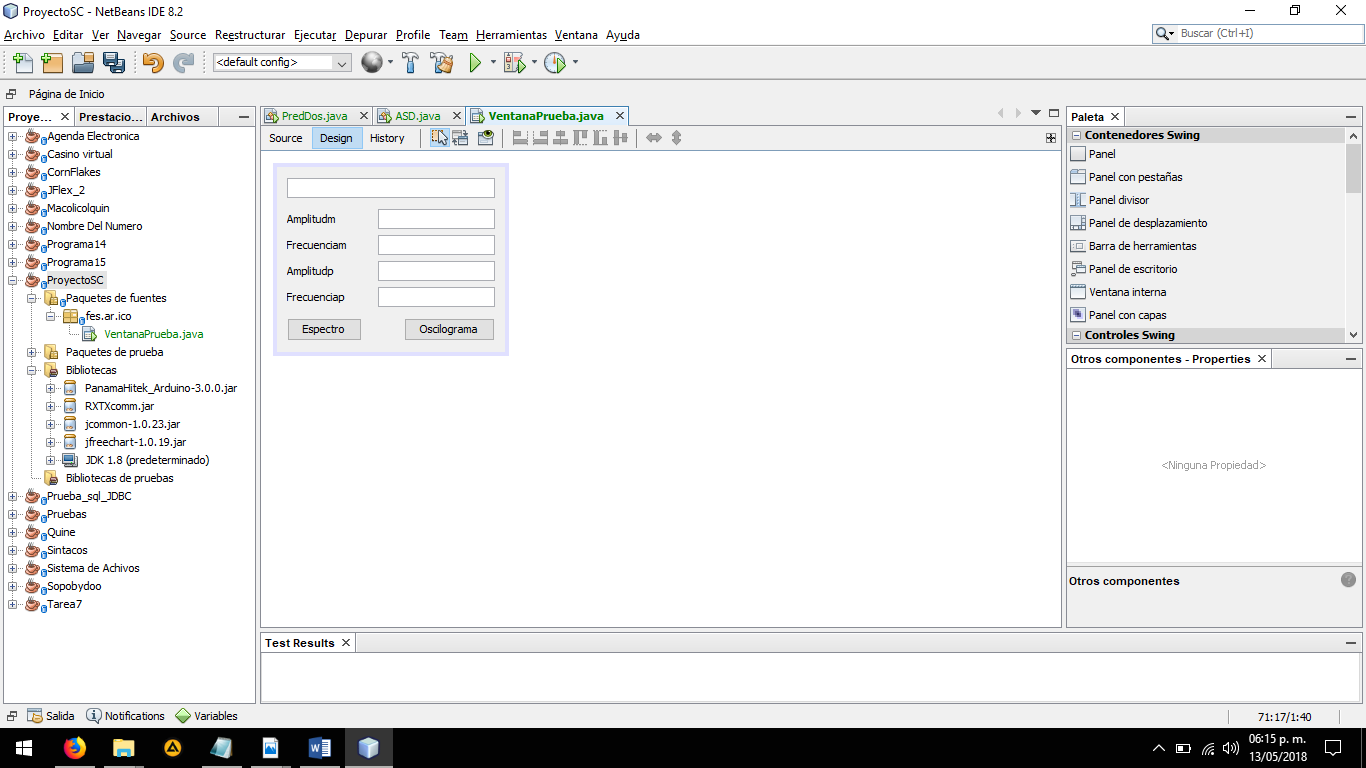
PanamaHitek\_Arduino-3.0.jar para hacer la comunicación en el puerto serie.

RXTXcomm.jar complemento para la librería anterior.

jcommon-1.0.23.jar para hacer series de coordenadas (x,y)..

jfreechart-1.0.19.jar para crear graficas.

Una vez hecho la inclusión de las librerías generamos una ventana con un formulario JFrame llamado VentanaPrueba y colocamos los siguientes atributos:



//componentes que interactúan con el usuario

private javax.swing.JTextField am;

private javax.swing.JButton analizador;

private javax.swing.JTextField ap;

private javax.swing.JTextField fm;

private javax.swing.JTextField fp;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JButton osciloscopio;

private javax.swing.JTextField texto;

//Serie para el osciloscopio

final XYSeries serie = new XYSeries("oscilograna");

//Serie para el analizador de espectros

final XYSeries amp = new XYSeries("espectro de frecuencia");

//colección para el osciloscopio

final XYSeriesCollection coleccion = new XYSeriesCollection();

//colección para el analizador de espectros

final XYSeriesCollection ampcol = new XYSeriesCollection();

//grafica para el oscilograma

JFreeChart grafica;

//grafica para el analizador de espectros

JFreeChart ampgra;

//ventana para el osciloscopio

JFrame ampven = new JFrame("Osciloscopio");

//ventana para el analizador de espectros

JFrame ventana = new JFrame("Analizador de espectros");

//clase Arduino para realizar la conexión al puerto serie

PanamaHitek\_Arduino arduino = new PanamaHitek\_Arduino();

//cadena zapturadora de datos

String[] tomo = new String[4];

//variables de control

int i = 0;

//variables para manejar parámetros Amplitud y frecuencia

int iam = 0;

int ifm = 0;

int iap = 0;

int ifp = 0;

//acción que se realiza cada vez que se recibe un dato del puerto serie

SerialPortEventListener evento = new SerialPortEventListener() {

@Override

public void serialEvent(SerialPortEvent spe) {

try {

//si existe un mensaje guarda los parámetros en las variables de control

if (arduino.isMessageAvailable() == true) {

tomo = arduino.printMessage().split("/");

texto.setText(Arrays.toString(tomo));

iam = Integer.parseInt(tomo[0]);

am.setText(iam + "");

ifm = (int) Double.parseDouble(tomo[1]);

fm.setText(ifm + "");

iap = Integer.parseInt(tomo[2]);

ap.setText(iap + "");

ifp = (int) Double.parseDouble(tomo[3]);

fp.setText(ifp + "");

}

} catch (SerialPortException | ArduinoException ex) {

Logger.getLogger(VentanaPrueba.class.getName()).log(Level.SEVERE, null, ex);

}

}

};

Dentro del constructor se realiza la conexión con el Arduino con la siguiente instrucción

arduino.arduinoRX("COM3", 9600, evento);

Este Arduino ya lo habíamos instanciado en los atributos y le decimos que del COM3, donde se conecta el Arduino a PC, y del puerto serie 9600 realizará por cada mensaje recibido la acción de evento.

Hasta ahora no hace lo que deseamos, que nos muestre una señal modulada. Ahora va la magia:

Para el analizador usamos la acción de click con su botón designado:

private void analizadorMouseClicked(java.awt.event.MouseEvent evt) {

i = 0;

ventana.revalidate();

serie.clear();

serie.add(0, 0);

coleccion.removeAllSeries();

coleccion.addSeries(serie);

grafica = ChartFactory.createXYLineChart("", "Frecuencia", "Amplitud", coleccion, PlotOrientation.VERTICAL, false, true, true);

ChartPanel panel = new ChartPanel(grafica);

ventana.getContentPane().add(panel);

ventana.pack();

ventana.setVisible(true);

while (i < (ifp \* 2)) {

int fpfm1 = ifp - ifm;

int fpfm2 = ifp + ifm;

if (i == fpfm1) {

serie.add(fpfm1, iam);

} else {

serie.add(i, 0);

}

if (i == ifp) {

serie.add(ifp, iap);

} else {

serie.add(i, 0);

}

if (i == fpfm2) {

serie.add(fpfm2, iam);

} else {

serie.add(i, 0);

}

i++;

}

}

Y para el oscilograma usamos igual un click:

private void osciloscopioMouseClicked(java.awt.event.MouseEvent evt) {

i = 0;

ampven.revalidate();

amp.clear();

amp.add(0, 0);

ampcol.removeAllSeries();

ampcol.addSeries(amp);

ampgra = ChartFactory.createXYLineChart("", "Tiempo", "Amplitud", ampcol, PlotOrientation.VERTICAL, false, true, true);

ChartPanel panel1 = new ChartPanel(ampgra);

ampven.getContentPane().add(panel1);

ampven.pack();

ampven.setVisible(true);

while (i < 10000) {

double moduladora = iam \* (Math.sin(6.28 \* i \* ifm));

double modulada = (iap + moduladora) \* (Math.sin(6.28 \* i \* ifp));

amp.add(i, modulada);

i++;

}

}

En total nuestro programa lo que hace es imprimir el espectro y el oscilograma de una señal modulada y, como tiene los aspectos de JFreeChart podemos importar fácilmente las impresiones.

**Conclusiones**

En cuanto a los conceptos de la modulación sabemos que nos ha quedado un poco más claro.

Aunque es un proyecto muy simple, se espera que pueda servir de utilidad para nuestros compañeros que realicen la misma practica en un futuro.

**Bibliografia**

http://arduino.cl/que-es-arduino/

https://www.youtube.com/watch?v=VALfzVn-GSo&list=WL&t=0s&index=611

https://www.youtube.com/watch?v=vKiEB21hkbk&t=920s

https://www.youtube.com/watch?v=NIpjixNP1Qc&t=87s

http://panamahitek.com/arduinojava-graficadora-de-datos-de-sensores/

Couch, W. Leon, II, “Sistema de comunicación digitales y analógicos” 7° edición. Prentice Hall, México, 2008.

Tomasi, Wayne “Sistemas de comunicación digital” 4° edición, Prentice Hall, México, 2003.