|  |  |  |
| --- | --- | --- |
| http://portaltransparencia.gob.mx/pot/imagenServlet?archivo=11171 | **3CM1** | http://www.escom.ipn.mx/Conocenos/PublishingImages/fotoEscudoESCOM.jpg |

****

# **PROYECTO FINAL**

# **“UPDATE MY SENSOR”**

Materia: Instrumentación

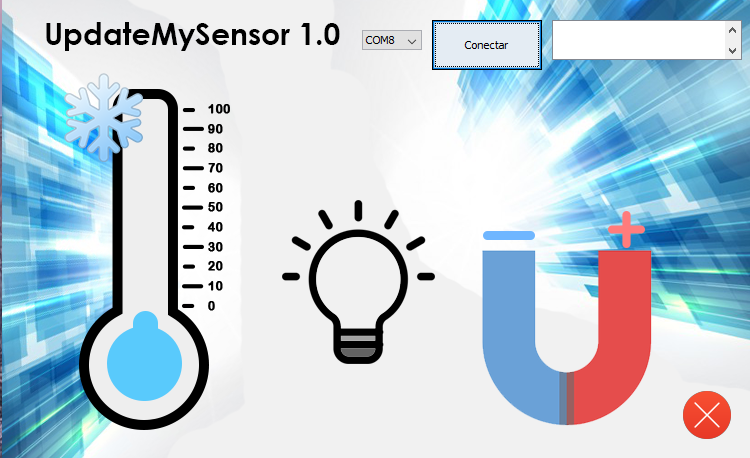
Profesor: Martínez Díaz Juan Carlos

### **Integrantes**

Guerra Vargas Irving Cristóbal

Jiménez Muñoz Arvid

Grupo: 3CM1



### **indice**

[Proyecto Final “Update My Sensor” 1](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677077)

[**Integrantes** 1](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Objetivos 3](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677079)

[Material y equipo 3](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677082)

[Introduccion 4](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677083)

[Descripcion de los componentes 5](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677083)

[Termistor NTCLE100E3 5-6](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[AD620 7-8](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Puente de Wheatstone 9](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[LDR 10-11](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[NE555 12](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[LM331 13-14](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[UGN3503 15-16](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Planteamiento del problema 17](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

[Diagrama a bloques completo 17](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

[Bloques del Sensor de temperatura 18](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

[Bloque Sensor 18-22](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloque CAS 22-23](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Mediciones 24](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloques del Detector de Luz 25](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

[Bloque Fotoresistencia 26-27](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloque Convertidor Frecuencia Voltaje 28-29](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloque CAS 30-31](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Propuesta de Diseño y Calculos 33-34](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Mediciones 35-36](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloques del Sensor Magnetico 37](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

[Bloque Sensor 37-38](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloque CAS 38-39](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Propuesta de Diseño y Calculos 40](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Mediciones 41](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloques Generales del Sistema 42](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

[Bloque Launchpad 42](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloque Interfaz Grafica 43](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloque Base de Datos 44](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Bloque Extra – Pagina WEB 45](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677078)

[Conclusiones 46](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

[Evidencias y Firmas 47-48](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

[Anexos 49-56](file:///C:\Users\Sir-M\Downloads\Prac%20Instru\Practica%205.docx#_Toc527677085)

### **Objetivos**

* Realizar un circuito que permita dar la medición de temperatura en un rango de medición de 0°C a 50°C. Utilizando de sensor una NTC de 10kΩ, linealizarla con un puente de Wheatstone, así como el acondicionamiento de la señal linealizada.
* Realizar un circuito detector de luz, implementando una foto resistencia y un conversor de Frecuencia a Voltaje, así como su etapa de acondicionamiento, para poder detectar diferentes intensidades de luz en Watts.
* Realizar un circuito que permita medir los Gauss emitidos por un imán, siendo capaz de detectar su polo positivo y su polo negativo, en un rango de -1900 Gauss a 400 Gauss, que posteriormente se acondicionara dicha señal.
* Implementar la tarjeta de desarrollo MSP430 Launchpad para poder leer los tres sensores diseñados, y poder enviarlos al puerto serie de una computadora.
* Realizar una interfaz que permita la lectura del puerto de Comunicación Serial de una computadora.
* Diseñar la interfaz para poder mostrar gráficamente los datos recibidos desde el puerto COM de la PC y animar dichos sensores para su mejor visualización gráfica, con la sincronización a una base de datos donde los sensores se estarán actualizando cada cierto periodo de tiempo.
* Implementación de una Pagina WEB que permita la lectura de la ultima actualización de los sensores a la base de datos.

### **Material y Equipo**

Sensor de Temperatura:

* 3 Trimpot de 100KΩ
* 2 resistencias de 10KΩ
* 1 amplificador AD620

Detector de Luz:

* 1 NE555
* 1 LM331
* 1 LM741
* 1 Trimpot 500KΩ
* 1 fotoresistencia de 10kΩ
* 1 capacitor 680pF
* 2 capacitores 10nF
* 1 capacitor 1uF
* 1 resistencia 100kΩ
* 2 resistencias 6.7kΩ
* 2 resistencias 10kΩ

Sensor de Magnetismo

* 1 UGN3503
* 1 resistencia 10kΩ
* 2 resistencias 1kΩ

Equipo General

* 2 Fuentes de Voltaje 15V
* 1 Fuente de Voltaje 5V
* 2 Multímetros.
* Tarjeta de desarrollo MSP430 Launchpad
* 1 computadora
* Conexión a internet estable.

### **Introducción**

En el presente reporte se mostrarán todas las etapas necesarias para que los objetivos de este proyecto se pudieran cumplir. Se especificará el desarrollo de la medición de 2 variables, calor en grados centígrados y el campo magnético en Gauss. Así como la detección de una variable luminosa en Watts.

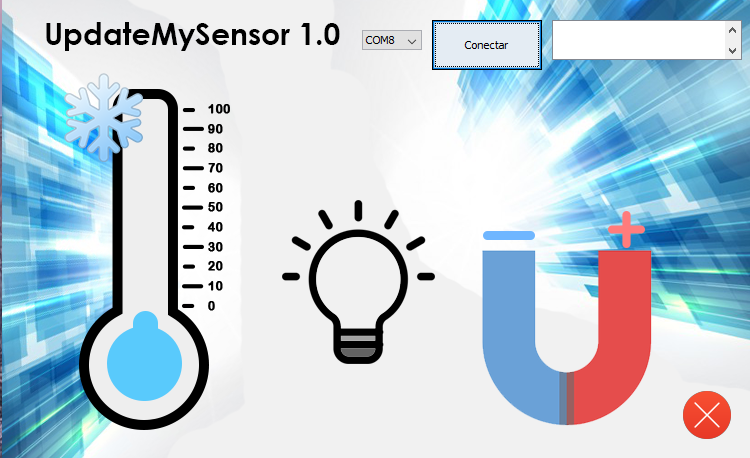
La detección de una variable no es lo mismo que la medición. Un sensor, ya sea de temperatura o de campo magnético, te especifica una relación, entre la variable física y la variable eléctrica que entregara. En cambio, un detector de luz es diferente, porque el diseñador del proyecto define las diferentes intensidades de luz que su detector medirá.

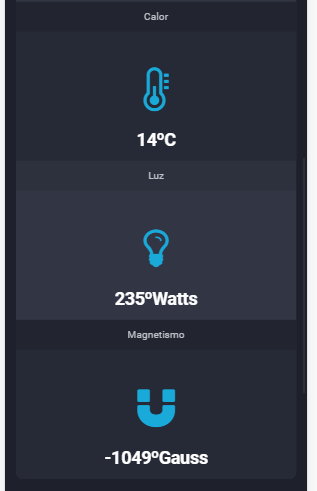
Este proyecto se desarrollará en 5 fases:

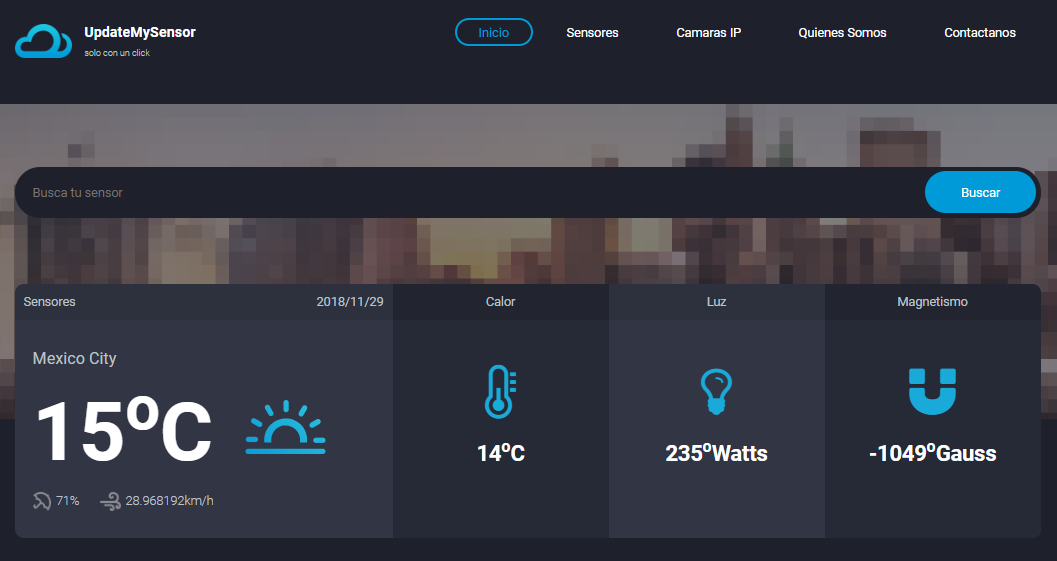
1. Lectura de Variables Físicas
2. Acondicionamiento de Variables
3. Transferencia de señales a través de una tarjeta de desarrollo
4. Lectura de las señales en una interfaz grafica
5. Conexión y actualización a internet para visualización en una página WEB

Este ultimo punto se decidió añadir porque para que un sistema de lectura de sensores tenga una funcionalidad más precisa o bien tenga mas alcance, el internet es el arma principal para poder lograr esto. Ya que con el simple echo de tener el sistema funcionando, encendido y con conexión a internet, en cualquier parte del mundo se tendrá la monitorización de estas lecturas de los sensores.

Interfaz Gráfica:



Interfaces WEB. Escritorio/Móvil



### **DESCRIPCION DE LOS COMPONENTES**

### **TERMISTOR 10KΩ (NTCLE100E3)**

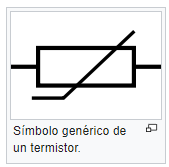
Un termistor es un sensor de temperatura por resistencia. Su funcionamiento se basa en la variación de la [resistividad](https://es.wikipedia.org/wiki/Resistividad) que presenta un semiconductor con la temperatura. El término termistor proviene de **Therm**ally Sensitive Res**istor.**

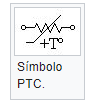
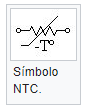
Existen dos tipos fundamentales de termistores:

* [NTC](https://es.wikipedia.org/w/index.php?title=Negative_Temperature_Coefficient&action=edit&redlink=1) (Negative Temperature Coefficient), donde la resistencia disminuye a medida que aumenta la temperatura.
* [PTC](https://es.wikipedia.org/w/index.php?title=Positive_Temperature_Coefficient&action=edit&redlink=1) (Positive Temperature Coefficient), los cuales incrementan su resistencia a medida que aumenta la temperatura

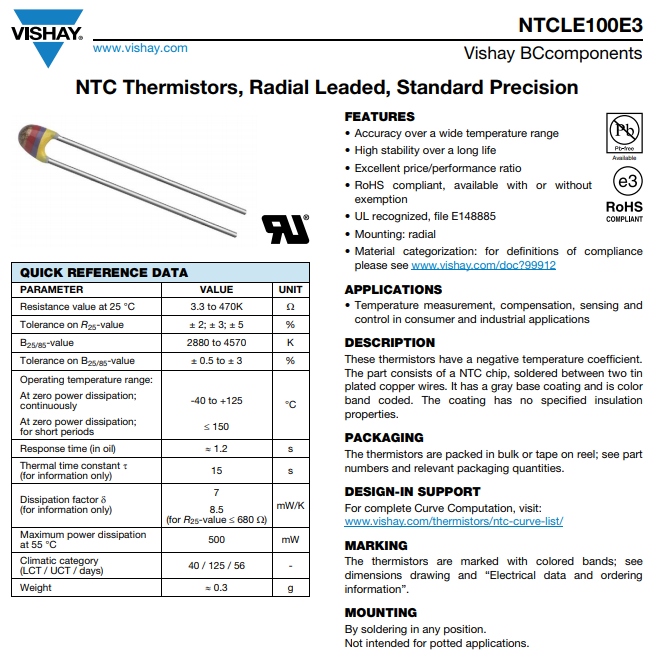
Los termistores de coeficiente de temperatura negativo (NTC) se usan comúnmente como sensores de temperatura o como limitadores de corriente de arranque.

Los termistores de coeficiente de temperatura positivo (PTC) o **posistore** se instalan comúnmente para proteger contra condiciones de sobre corriente, por ejemplo: como fusibles reajustables





### **DATASHEET NTCLE100E3**



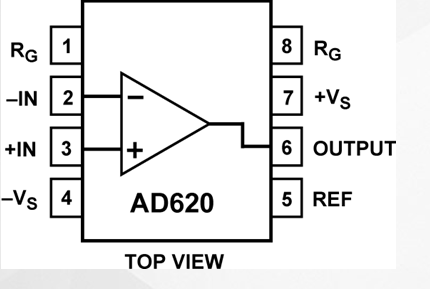
### **AMPLIFICADOR OPERACIONAL DE INSTRUMENTACION AD620**

## El amplificador de instrumentación de Analog Devices es de baja deriva, energía baja, alta exactitud y es ideal para aplicaciones de alta precisión.

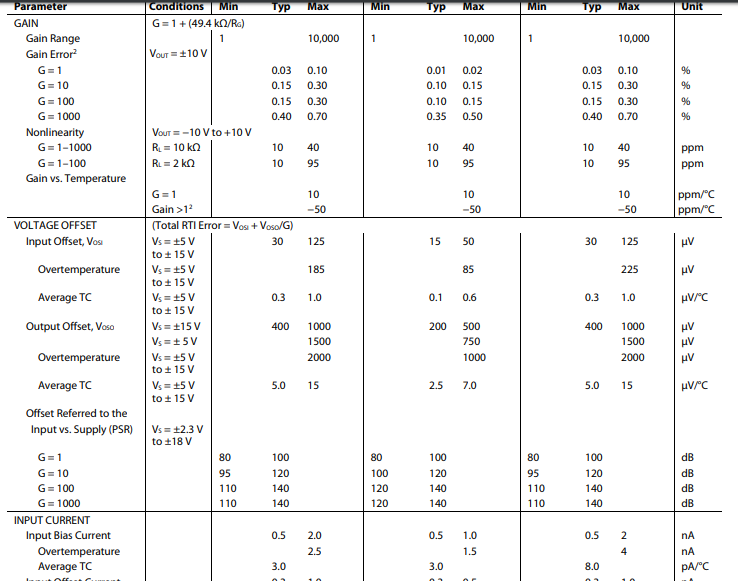
## El AD620, con su alta precisión de no linealidad de 40 ppm (máximo), baja tensión de deriva de 50 μV (máximo) y compensación de deriva de 0,6 μV / ° C (máxima), es ideal para uso en sistemas de adquisición de datos precisos tales como balanzas de peso e interfaces de transductor. Además, el bajo ruido, baja corriente de polarización de entrada y baja potencia de entrada del AD620lo hacen apto para aplicaciones médicas tales como ECG y monitores de presión sanguínea no invasivos.

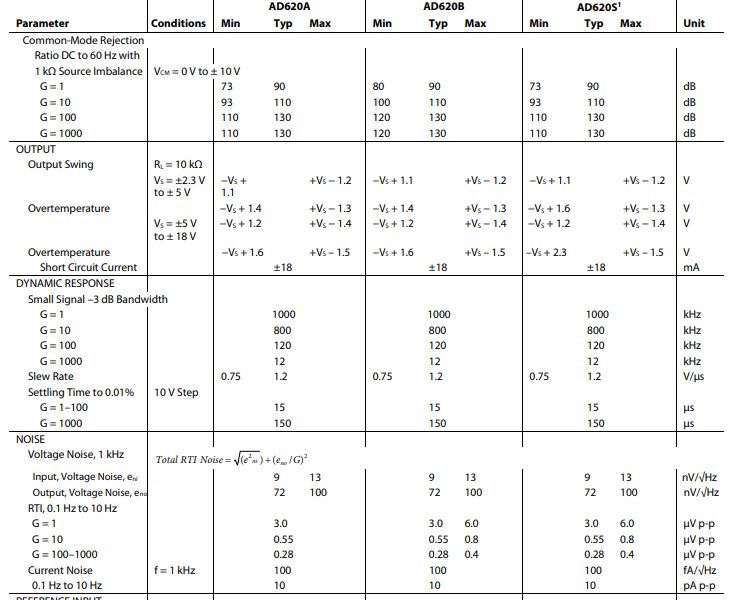
La baja de entrada corriente de polarización de 1,0 nA (máximo) es posible con el uso de proceso Superϐeta en la etapa de entrada. El AD620 trabaja bien como un preamplificador debido a su bajo ruido de tensión de entrada de 9 nV/√Hz a 1 kHz, 0.28 μV p-p en la banda 0.1 Hz a 10 Hz, y 0.1 pA/√Hz de ruido de corriente de entrada. Además, el AD620 es ideal para aplicaciones de multiplexado con su tiempo de adaptación de 15 μs a 0.01%, y su costo es tan bajo que permite con una amp-in por canal.

|  |  |  |
| --- | --- | --- |
| **Características:** | | |
| * Rango de ganancia de 1 a 10.000 con una resistencia externa * Amplio rango de potencia (±2.3 V a ±18 V) * Rendimiento más alto que tres diseños IA de op amp * Baja alimentación, 1.3 mA (máx.) de corriente de alimentación * Voltaje de polarización de entrada 50 μV (máx.) * Deriva de polarización de entrada de 0.6 μV/°C (máx.) |  | * Corriente de polarización de entrada 1,0 nA (máx.), * Relación de rechazo de modo común de 100 dB (mínimo) (G = 10) * Bajo nivel de ruido * Ruido de voltaje de entrada de 9 nV/√Hz a 1 kHz * 0.28 μV p-p ruido (0.1 Hz a 10 Hz) * Ancho de banda de 120 kHz (G = 100) * Tiempo de ajuste de 15 μs a 0.01% |



### **DATASHEET AD620**





### **PUENTE DE WHEATSTONE**

El puente de Wheatstone es un circuito inicialmente descrito en 1833 por Samuel Hunter Christie (1784-1865), Pero fue el Sr. Charles Wheatstone quien le dio muchos usos cuando lo descubrió en 1843. Como resultado este circuito lleva su nombre.

Daremos un ejemplo para ver el funcionamiento teórico del puente de Wheatstone.

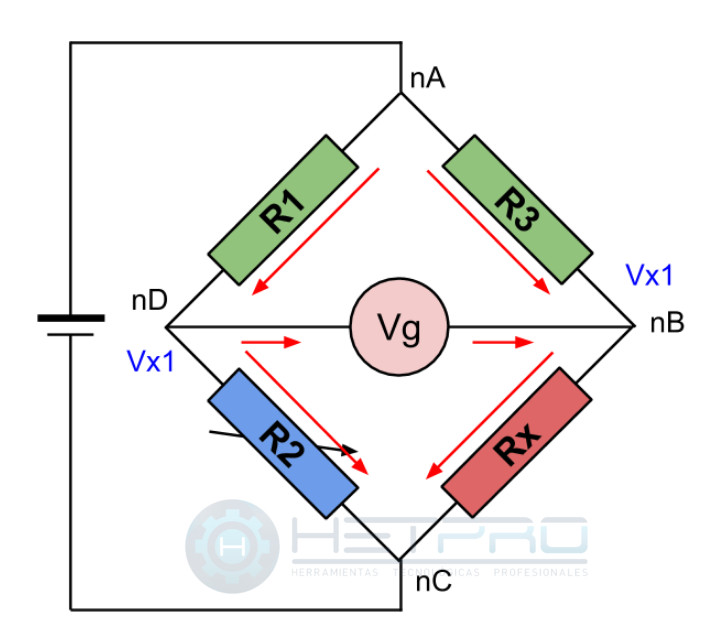
Sistema en igualdad de resistencias, cuando el sistema está en equilibrio.

\[{\frac{R_2}{R_1}}={\frac{R_x}{R_3}}  \]

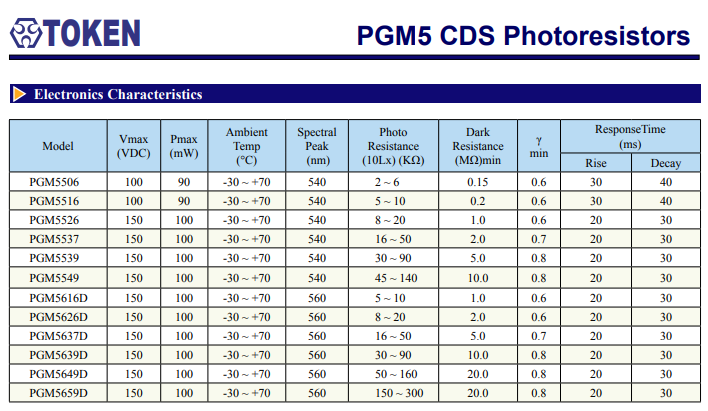
En este caso, para determinar un valor de salida correspondiente a la variación de Rx, consideramos el análisis del circuito. Considerando que el instrumento de medición tiene una resistencia suficientemente alta, la determinación del voltaje de la diferencia de potencial entre Vx1 y Vx2 es igual a:

\[V_G =  {\left(}{\frac{R_2}{R_1 + R_2}}-{\frac{R_X}{R_X + R_3}} {\right)}V_S   \]

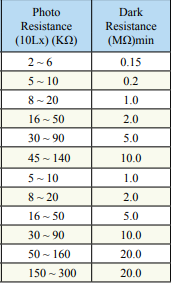
El análisis anterior esta basado en la siguiente figura:



### **fotoresistencia lDR**



En donde tenemos que resaltar 2 columnas esenciales que determinaran las siguientes etapas de la practica:

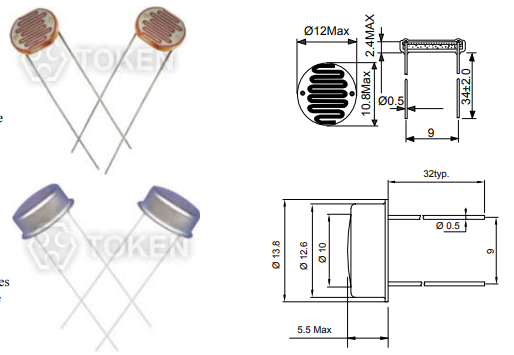


En la primera columna se establece el valor de resistencia con luz y a diferencia de la segunda columna que indica el valor de resistencia en obscuridad total.

Si bien sabemos que las tablas que manejan valores ideales son diferentes a los valores prácticos.

En la realización de esta práctica se utilizó un PGM5659D, que idealmente proporciona a obscuridad total 20MΩ, pero a nosotros, medido dentro de la caja negra nos arrojaba una resistencia de 25MΩ

### **Datasheet del PGM5659D**



Características:

***Resistencia a la luz:***

Medido a 10 lux con luz estándar A (temperatura de color 2854K) y 2hr. Preiluminación a 400-600 lux previa prueba.

2. Resistencia oscura:

Medido 10 segundos después de cerrado 10 lux.

3. Característica gamma:

Entre 10 lux y 100 lux y dado por.

= log (R10 / R100) / log (100/10) = log (R10 / R100)

R10, R10: Resistencia celular a 10 lux y 100 lux. La tolerancia de es 0.1.

4. Pmax:

Max. Disipación de potencia a temperatura ambiente de 25ºC.

5. Vmax:

Max. Voltaje en la oscuridad que puede aplicarse a la celda de forma continua.

### **NE555**

El dispositivo NE555 es un circuito integrado muy estable cuya función primordial es la de producir pulsos de temporización con una gran precisión y que, además, puede funcionar como oscilador.

Sus características más destacables son:

* Temporización desde microsegundos hasta horas.
* Modos de funcionamiento:
  + Monoestable.
  + Astable.
* Aplicaciones:
  + Temporizador.
  + Oscilador.
  + Divisor de frecuencia.
  + Modulador de frecuencia.
  + Generador de señales triangulares.

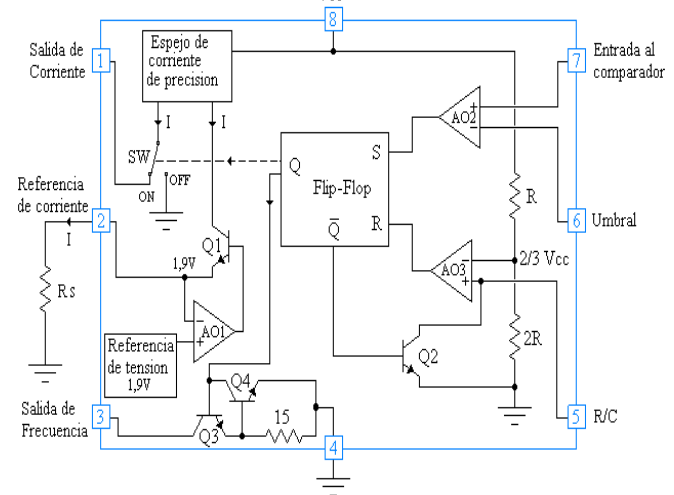
Pasemos ahora a mostrar las especificaciones generales del 555 (Vc = disparo):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Especificaciones generales del 555** | | | | |
| **Vcc** | **5-Voltios** | **10-Voltios** | **15-Voltios** | **Notas** |
| **Frecuencia máxima (Astable)** | 500-kHz a 2-MHz | | | Varia con el Mfg y el diseño |
| **Nivel de tensión Vc (medio)** | 3.3-V | 6.6-V | 10.0-V | Nominal |
| **Error de frecuencia (Astable)** | ~ 5% | ~ 5% | ~ 5% | Temperatura 25° C |
| **Error de temporización (Monoestable)** | ~ 1% | ~ 1% | ~ 1% | Temperatura 25° C |
| **Máximo valor de Ra + Rb** | 3.4-Meg | 6.2-Meg | 10-Meg |  |
| **Valor mínimo de Ra** | 5-K | 5-K | 5-K |  |
| **Valor mínimo de Rb** | 3-K | 3-K | 3-K |  |
| **Reset VH/VL (pin-4)** | 0.4/<0.3 | 0.4/<0.3 | 0.4/<0.3 |  |
| **Corriente de salida (pin-3)** | ~200ma | ~200ma | ~200ma |  |

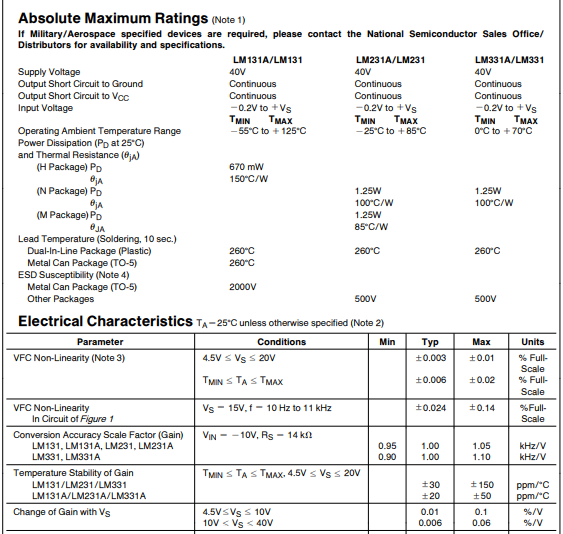
### **LM331**

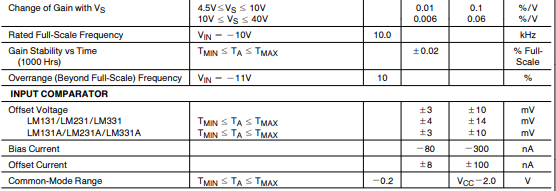
El conversor LM331 es un integrado de gran versatilidad que puede operar con fuente simple y con errores aceptables en el rango de 1 Hz a 10 KHz. Está pensado para realizar tanto la conversión tensión – frecuencia, como para la conversión frecuencia–tensión.

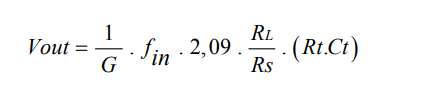
Diagrama interno:



### **DATASHEET LM331**







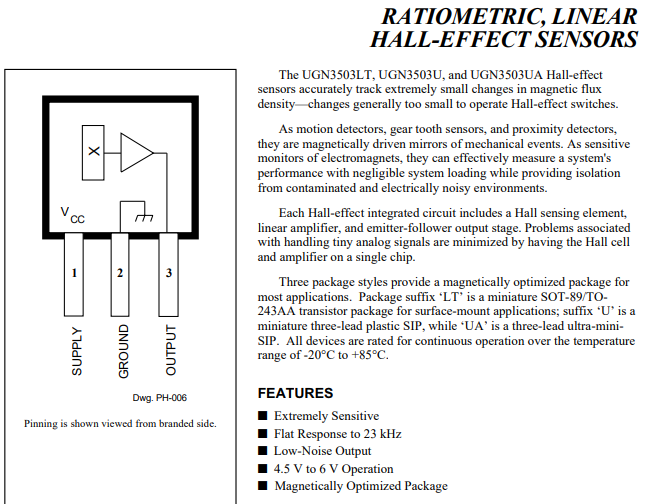
### **UGN 3503**

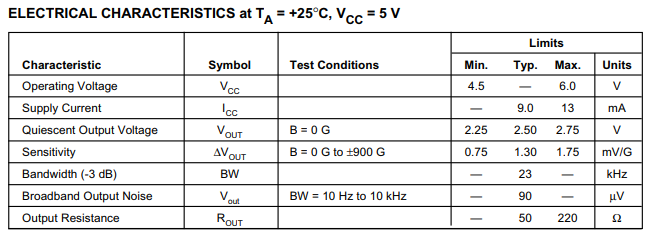
Salida lineal proporcional a cambios en el flujo magnético. Empleado en la detección de pequeños cambios en la densidad del flujo magnético, censado de corriente, movimiento, posición, proximidad, posición de piñones y muescas en ruedas o elementos giratorios, etc.

**Características:**

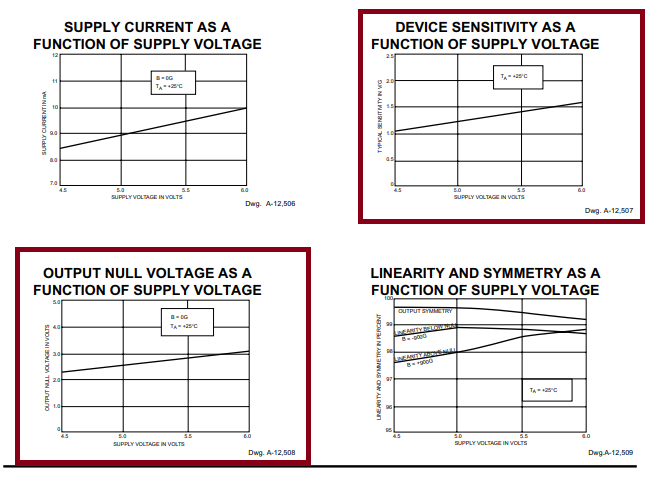
* Extremadamente sensible
* Respuesta plana hasta 23 kHz
* Salida con bajo ruido
* Salida radiométrica
* Sensibilidad: 1.3 mV/G típicamente
* Voltaje de operación: 4.5 V a 6 V
* Encapsulado magnéticamente optimizado
* Encapsulado: SIP de 3 pines

### **DATASHEET UGN 3503**





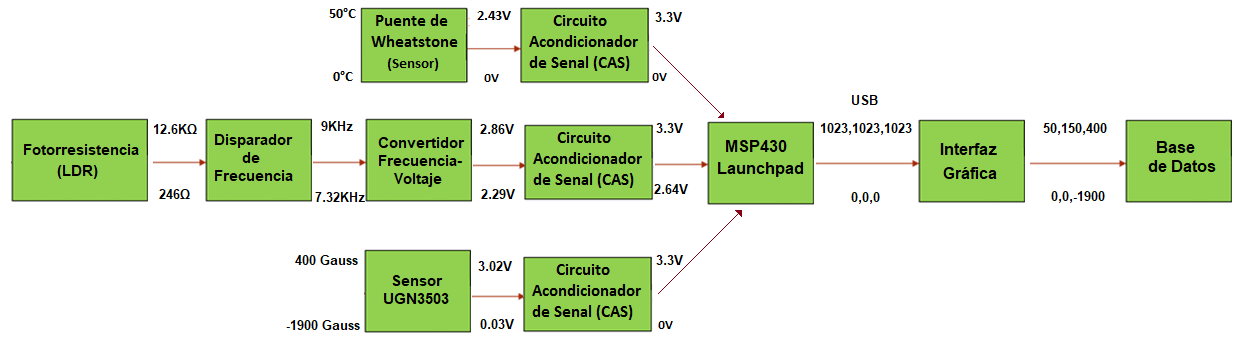
Se pondrá en un recuadro rojo las graficas que se necesitaran para esta práctica (ya que su salida es Radiométrica)



### **Planteamiento del problema**

Implementar los diferentes circuitos de los objetivos, el sensor de calor, de magnetismo y el detector de luz, así como el diseño gráfico de una aplicación de escritorio y una pagina web para la consulta de los datos almacenados.

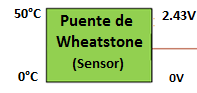
### **Diagrama a bloques COMPLETO**



A continuación, se explicarán detalladamente cada bloque anterior para una mejor comprensión.

### **BLOQUES DEL SENSOR DE TEMPERATURA**

### **bloque del puente de wheatstone (sensor)**



Comenzaremos describiendo el comportamiento y los cálculos para el NTC ya que primero debemos saber bien los datos del NTC para posteriormente ingresarlos a el puente Wheatstone.

Siguiendo el modelo explicado en el apartado del NTC, y basándonos en la teoría proporcionada en clase más los datos proporcionados por el fabricante en la hoja de datos.

Primero definiremos un RANGO de medidas para poder hacer nuestros cálculos. Los cuáles serán de 0°C a 50°C, o bien de 273 K a 323 K

Consultaremos el datasheet, y obtendremos el valor de B



Con esto podemos obtener los alfas a esas diferentes temperaturas:

α (273)=

α (323) = -0.038119794

ahora calculamos las RT’s

Sabemos que

Para linealizar el puente de Wheatstone

Sabemos que Rt = 10000Ω que este dado por el fabricante

Recalcularemos esto de la siguiente forma:

Calculamos a sensibilidad de 3 temperaturas:

Stc = a la sensibilidad del punto central, el cual dado que nuestro rango es de 0° a 50°C nuestro punto central será 25°C en el punto mas lineal de la gráfica.

El cual se calcula de la siguiente manera:

Stc= -80.9117

ST(273)= -96.41

ST(323)=-68.87

Ahora podemos calcular el índice único de A

Donde se expresa de la siguiente manera:

Despejando A nos queda de la siguiente manera:

Sabemos que la relación del puente para obtener 0V a 25°C

la expresión para el voltaje de salida es de la siguiente forma:

Por lo tanto:

R2= 2946.026 lo cual aproximaremos a 3kΩ

Con todo esto, propondremos un voltaje para energizar el circuito, en este caso se eligieron 5V

Por lo tanto:

Donde Rt323 = a la resistencia Rt a 323 °K donde son = 50°c

Ahora que tenemos la intensidad (I) podemos calcular:

Donde Ỽ= es proporcionado por el fabricante = 8.5



Entonces queremos calcular ∆T si despejamos queda

=

Ahora calcularemos el error por autocalentamiento.

Lo calculamos de la siguiente manera

Sabemos que la sensibilidad (S) equivale a que significa a la derivada del Voltaje de salida

=

Donde

B = 3977

La derivación equivale a:

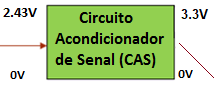
Para finalizar calcularemos los valores finales para las otras dos temperaturas.

Así finalizamos esta etapa del puente de Wheatstone.

Ahora determinaremos algunos valores, cambiando el valor de RX, que es la resistencia de la NTC

|  |  |  |  |
| --- | --- | --- | --- |
| Temperatura °C | Temperatura °K | NTC (Ω) | V0 |
| 0 | **273** |  | **0** |
| 25 | **298** | 10,000 | **1.62** |
| 50 | **323** |  | **2.43** |

### **bloque del circuito acondicionador de señal**



Para esta etapa es necesario usar el ampliador de instrumentación AD620 el cual dadas las características que nos proporciona el fabricante, es lo suficientemente adecuado y útil para realizar el circuito acondicionador, y es sumamente preciso si lo comparamos con los anteriores amplificadores que se han usado.

Como veremos en la etapa anterior se nos arrojaba una señal de voltaje en la salida del puente, bien, ahora en esta etapa corresponde que el voltaje que nos entrega el puente de Wheatstone sea amplificado por el circuito acondicionador.

Como se ha manejado en prácticas anteriores y siguiendo la misma línea, respetaremos el parámetro de salida del circuito acondicionador, el cual está limitado en un rango de 0V a 3.3V.

Por lo tanto, comenzaremos analizando:

Dado que queremos que la salida al entrar 0V equivalga a 0V y a 2.43 equivalga a 3.3

Tenemos que

Ahora

Resaltado la formula de ganancia del amplificador operacional de instrumentación del AD620



Si despejamos la resistencia R que determinara la ganancia del operacional nos queda de la siguiente forma:

:

Teniendo esto en cuenta podemos graficar ahora con la amplificación necesaria.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Temperatura °C | Temperatura °K | NTC (Ω) | V0 | V0(Amplificada) |
| 0 | **273** | 33,944 | **0** | 0 |
| 25 | **298** | 10,000 | **1.62** | 2.2 |
| 50 | **323** | 3,559.57 | **2.43** | 3.3 |

### **Mediciones**

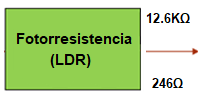
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Mediciones de Sensor NTC** | | | | | | | | |
|  |  | **MEDICIONES REALES** | | | **MEDICIONES IDEALES** | | |  |
| MeD | Temp °c | NTCΩ | V0 | V0(Amplificado) | NTC Ω | V0(Amp) | V0(Amplificado) | ERROR |
| 1 | 18 | 13500 | 0.8 | 1.10 | 13785.45672 | 0.85212486 | 1.157185567 | 1.09252132 |
| 2 | 19 | 13000 | 0.9 | 1.2222 | 13155.11324 | 0.90935027 | 1.234897665 | 1.028236194 |
| 3 | 20 | 12400 | 1.01 | 1.37158 | 12557.60363 | 0.9665475 | 1.31257151 | -4.49564000 |
| 4 | 23 | 10500 | 1.2 | 1.6296 | 10943.63841 | 1.13735978 | 1.544534583 | -5.50751133 |
| 5 | 25 | 10000 | 1.62 | 2.19996 | 10000 | 1.25 | 1.6975 | -29.6 |
| 6 | 27 | 9000 | 1.34 | 1.81972 | 9148.721802 | 1.36114034 | 1.848428585 | 1.55313464 |
| 7 | 29 | 8300 | 1.45 | 1.9691 | 8379.780169 | 1.47038074 | 1.996777047 | 1.38608597 |
| 8 | 30 | 8000 | 1.5 | 2.037 | 8023.382173 | 1.5241741 | 2.069828426 | 1.58604577 |
| 9 | 40 | 5000 | 2 | 2.716 | 5275.206348 | 2.0232782 | 2.747611801 | 1.15051919 |
| 10 | 50 | 3559.57 | 2.43 | 3.29994 | 3559.574473 | 2.43743135 | 3.310031773 | 0 |

### **graficando mediciones ideales**

### **graficando mediciones REALES**

### **BLOQUES DEL detector de luz**

### **bloque fotoresistencia**



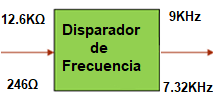
Este bloque es sumamente sencillo, dado que la foto resistencia dependiendo de la luz que se le fue recibida en la caja negra, la cual fue totalmente aislada de luz, en donde se encendía los diferentes focos en la cual obtuvimos la siguiente relación:

|  |  |
| --- | --- |
| Watts | Resistencia (Ω) |
| Obscuridad total | 25 MΩ |
| 25W | 12.6 KΩ |
| 40W | 1.63 KΩ |
| 60W | 920 Ω |
| 75W | 620 Ω |
| 100W | 560 Ω |
| Luz total (los 5 focos encendidos) | 240 Ω |

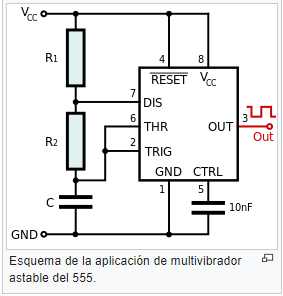
Graficando:

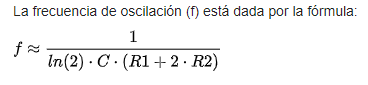
En la gráfica se decidió omitir el valor sin oscuridad, ya que tiende a un valor muy grande. Y analizando, desde aquí podemos darnos cuenta de que los valores de la foto resistencia NO son lineales.

### **Bloque disparador de frecuencia**



Para este bloque, se establece que la entrada de circuito integrado NE555 se determina de la siguiente manera:





En este punto, definimos un valor para el Capacitor muy pequeño, y el valor elegido fue de 680pF. Entre mas pequeño sea este valor, mejores resultados entregara el NE555.

También, teniendo en cuenta que a 12.6KΩ entregara la máxima frecuencia, se decidió que dicha frecuencia fuese de 9 KHz. Con esto, Podemos despejar a R1 de la formula.

Nos da como resultado una resistencia R1 de 210534 Ω.

Entonces, una vez teniendo esto, el N2555 entregara los siguientes valores de frecuencia:

|  |  |  |
| --- | --- | --- |
| Watts | Resistencia (Ω) | Frecuencia (Hz) |
| Obscuridad total | 25 MΩ | 120 Hz |
| 25W | 12.6 KΩ | 9.07 KHz |
| 40W | 1.63 KΩ | 7.8 KHz |
| 60W | 920 Ω | 7.7 KHz |
| 75W | 620 Ω | 7.6 KHz |
| 100W | 560 Ω | 7.4 KHz |
| Luz total (los 5 focos encendidos) | 240 Ω | 7.3 KHz |

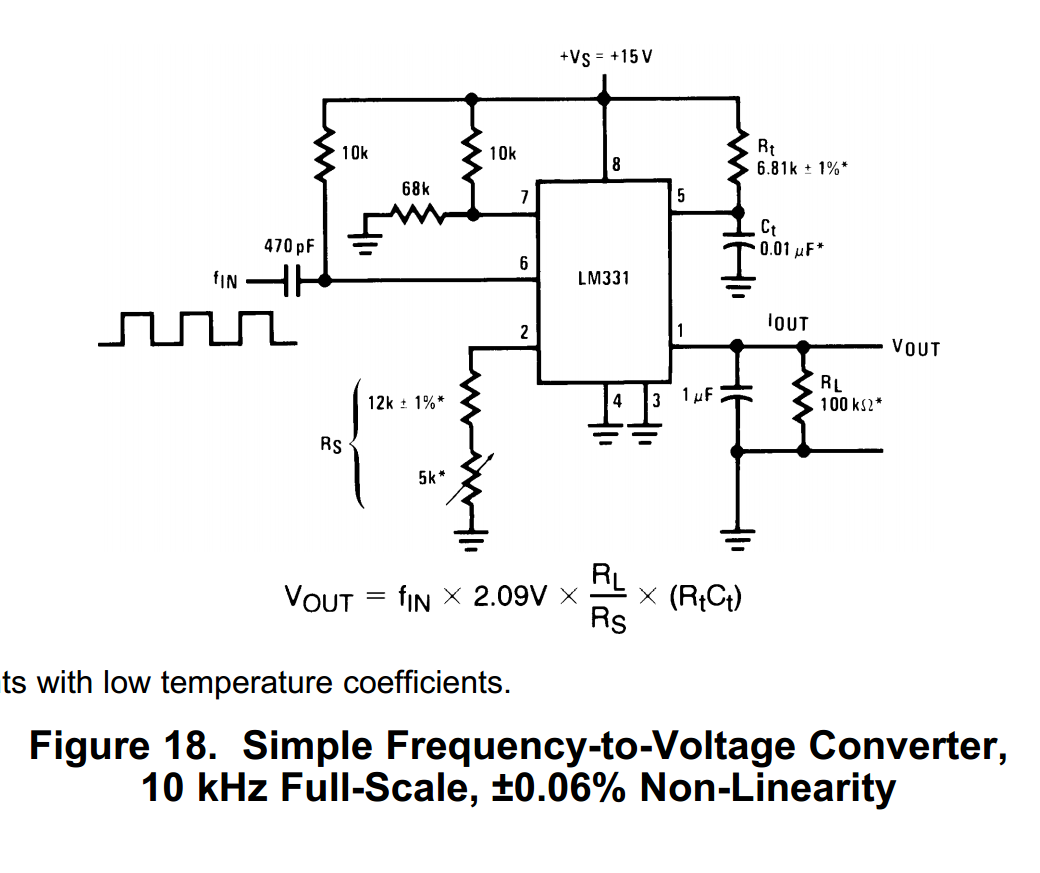
Graficando la frecuencia

**NOTA IMPORTANTE:** Igualmente se omitió el valor de 25MΩ, ya que, siendo tan grande, causa una irregularidad, la cual es que el NE555 toma ese valor como circuito abierto, y en ese valor tiene una incongruencia, ya que marca algo aproximado a 0Hz, y debería de marcar un valor muy superior.

### **BLOQUE CONVERTIDOR frecuencia-voltaje**



Este es un bloque también, muy sencillo, ya que se usa un diagrama que el fabricante del 331 nos brinda, el cual es:



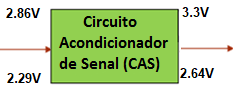
El mismo fabricante nos brinda el Valor de Salida en voltaje que nos va a entregar el LM331, entonces, aplicando dicha formula, tenemos la siguiente tabla:

***15***

|  |  |  |  |
| --- | --- | --- | --- |
| Watts | Resistencia (Ω) | Frecuencia (Hz) | Voltaje (331) |
| Obscuridad total | 25 MΩ | 120 Hz | 0 V |
| 25W | 12.6 KΩ | 9.07 KHz | 2.86 V |
| 40W | 1.63 KΩ | 7.8 KHz | 2.49 V |
| 60W | 920 Ω | 7.7 KHz | 2.46 V |
| 75W | 620 Ω | 7.6 KHz | 2.39 V |
| 100W | 560 Ω | 7.4 KHz | 2.36 V |
| Luz total (los 5 focos encendidos) | 240 Ω | 7.3 KHz | 2.29 V |

**NOTA IMPORTANTE:** Igualmente se omitió el valor de 25MΩ, ya que, siendo tan grande, causa una irregularidad, la cual es que el NE555 toma ese valor como circuito abierto, y en ese valor tiene una incongruencia, ya que marca algo aproximado a 0V, y debería de marcar un valor muy superior.

### **bloque del circuito acondicionador de señal**



En este bloque únicamente tenemos que amplificar 2.86V a 3.3V, para ello, lo más sencillo, rápido, barato y practico, es implementar un LM741, configurado como No inversor.

Primero necesitamos obtener la pendiente de la ecuación, con lo cual usaremos la siguiente ecuación:

Ahora, tomando en cuenta los valores de la gráfica anterior, nos queda que:

Por lo tanto, la ecuación del CAS queda de la siguiente manera:

Por lo tanto, la ecuación nos queda de la siguiente manera:

En este caso tenemos a b = 0, entonces queda la ecuación:

***17***

Tomando los valores principales para probar esta ecuación:

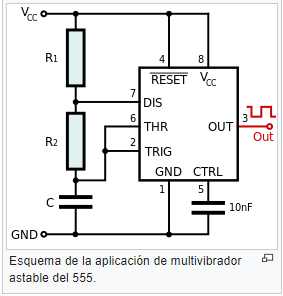
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Watts | Resistencia (Ω) | Frecuencia (Hz) | Voltaje (331) | Voltaje (CAS) |
| Obscuridad total | 25 MΩ | 120 Hz | 0 V | 0 V |
| 25W | 12.6 KΩ | 9.07 KHz | 2.86 V | 3.3 V |
| 40W | 1.63 KΩ | 7.8 KHz | 2.49 V | 2.87 V |
| 60W | 920 Ω | 7.7 KHz | 2.46 V | 2.83 V |
| 75W | 620 Ω | 7.6 KHz | 2.39 V | 2.75 V |
| 100W | 560 Ω | 7.4 KHz | 2.36 V | 2.72 V |
| Luz total (los 5 focos encendidos) | 240 Ω | 7.3 KHz | 2.29 V | 2.64 V |

**NOTA IMPORTANTE:** Igualmente se omitió el valor de 25MΩ, ya que, siendo tan grande, causa una irregularidad, la cual es que el NE555 toma ese valor como circuito abierto, y en ese valor tiene una incongruencia, ya que marca algo aproximado a 0V, y debería de marcar un valor muy superior.

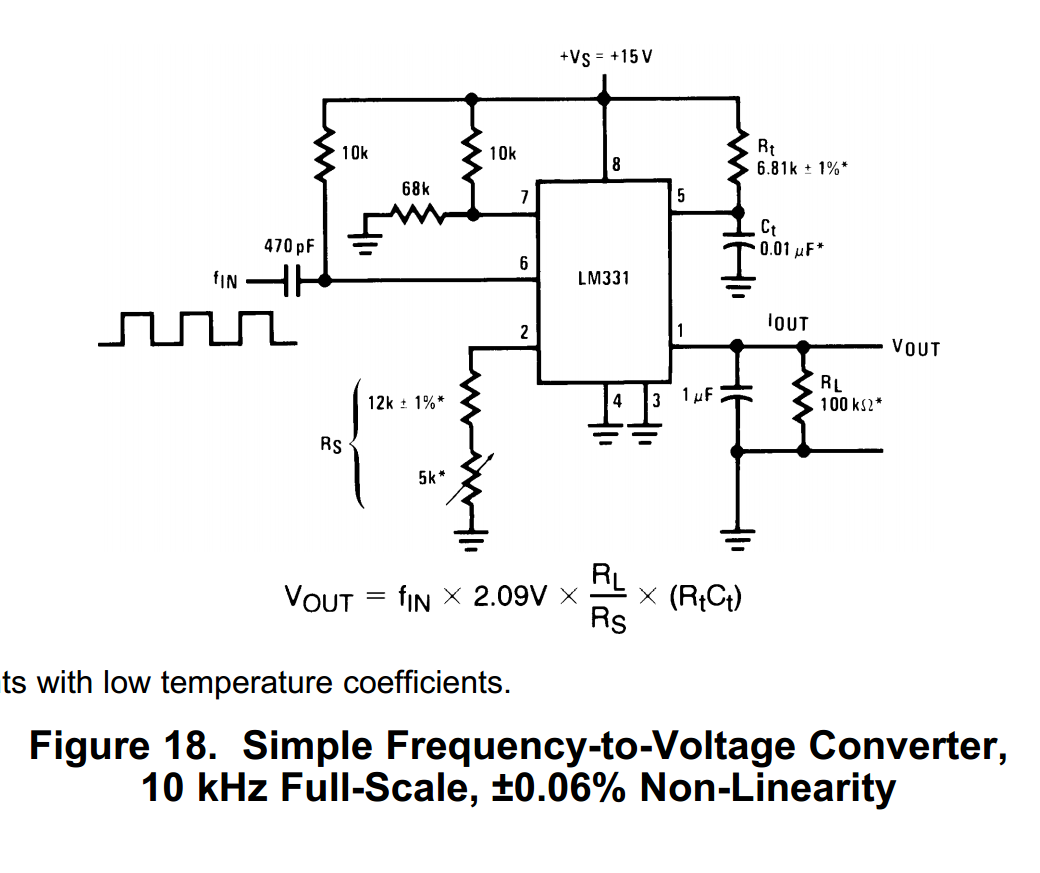
### **propuesta de diseño y calculos**

En esta ocasión, como se han explicado en los bloques anteriores, se estarán tomando los diseños que los fabricantes nos proveen.

Por ejemplo, para la etapa de la Fotorresistencia, se tomará:



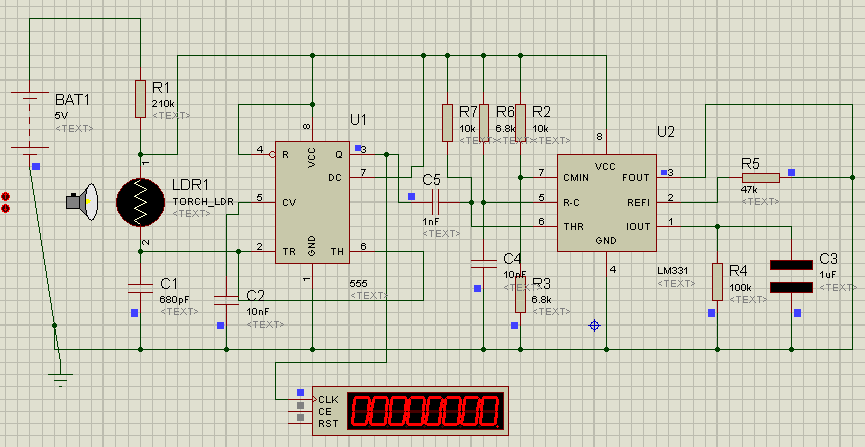
También, para la etapa del LM331. Tomaremos el circuito que nos brinda el fabricante:



Únicamente, para la etapa del CAS, donde encontramos la ecuación:

Si se analiza la ecuación, lo pasaremos por un amplificador configurado como NO inversor. Para ocupar solamente un integrado LM741 con una ganancia de 1.74.

### **Simulación del circuito final**

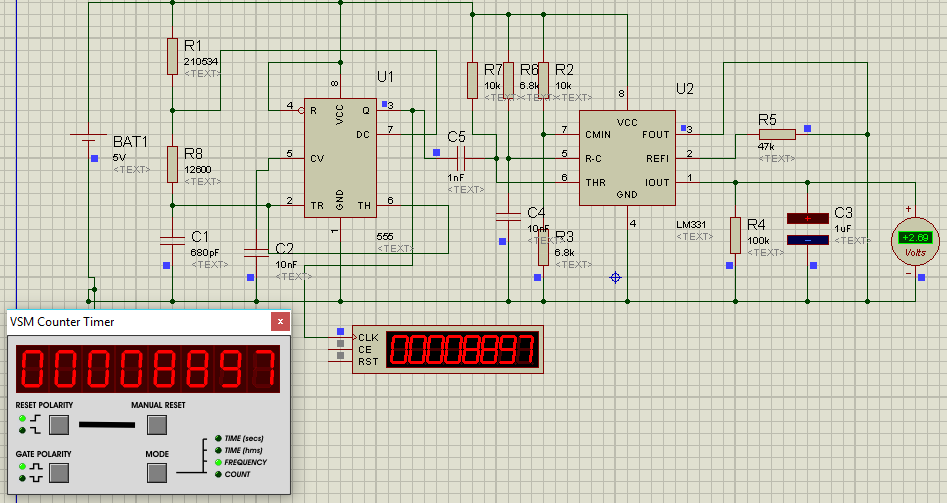


Al momento de querer simularse se intento hacer con una LDR que brinca Proteus, sin embargo, no marcaba frecuencia, y no solo eso, tampoco puedes definir los Watts.

Enconches para una simulación nos basaremos en esta tabla:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Watts | Resistencia (Ω) | Frecuencia (Hz) | Voltaje (331) | Voltaje (CAS) |
| Obscuridad total | 25 MΩ | 120 Hz | 0 V | 0 V |
| 25W | 12.6 KΩ | 9.07 KHz | 2.86 V | 3.3 V |
| 40W | 1.63 KΩ | 7.8 KHz | 2.49 V | 2.87 V |
| 60W | 920 Ω | 7.7 KHz | 2.46 V | 2.83 V |
| 75W | 620 Ω | 7.6 KHz | 2.39 V | 2.75 V |
| 100W | 560 Ω | 7.4 KHz | 2.36 V | 2.72 V |
| Luz total (los 5 focos encendidos) | 240 Ω | 7.3 KHz | 2.29 V | 2.64 V |

Y fijaremos en vez de una LDR una resistencia con el valor que debería de tener a 25W, que es de 12600 Ω y verificaremos que nos dé el resultado.



Como podemos observar, los resultados son muy aproximados a los calculados.

### **Mediciones**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Mediciones detector de Luz** | | | | | | | |  |
|  |  | **MEDICIONES REALES** | | | | | |  |
| Mediciones: | Foco (Watts) | Ω | Hz (555) | V (331) | V (741) | Valor Decimal | Valor Binario |  |
| 1 | Sin Luz | 25000000 | 120 | 0 | 0 | 0 | 0 |  |
| 2 | 25 | 12600 | 9070 | 2,86 | 4,9764 | 254 | 11111110 |  |
| 3 | 40 | 1630 | 7800 | 2,49 | 4,3326 | 218 | 11011010 |  |
| 4 | 60 | 980 | 7700 | 2,46 | 4,2804 | 217 | 11011001 |  |
| 5 | 75 | 620 | 7300 | 2,3 | 4,002 | 204 | 11001100 |  |
| 6 | 100 | 563 | 7400 | 2,36 | 4,1064 | 209 | 11010001 |  |
| 7 | Todos los Focos | 246 | 7320 | 2,29 | 3,9846 | 204 | 11001100 |  |
|  |  |  | **MEDICIONES IDEALES** | | | |  |  |
| Mediciones: | Foco (Watts) | Ω | Hz (555) | V (331) | V (741) | Valor Decimal | Valor Binario | ERROR |
| 1 | Sin Luz | 25000000 | 120 | 0,036285957 | 0,06615215 | 3,373759647 | 11 | 100 |
| 2 | 25 | 12600 | 9070 | 2,742613617 | 5 | 255 | 11111111 | -4,280091889 |
| 3 | 40 | 1630 | 7800 | 2,358587234 | 4,299889746 | 219,2943771 | 11011011 | -5,571672909 |
| 4 | 60 | 980 | 7700 | 2,328348936 | 4,244762955 | 216,4829107 | 11011000 | -5,654266926 |
| 5 | 75 | 620 | 7300 | 2,207395745 | 4,024255788 | 205,2370452 | 11001101 | -4,195181383 |
| 6 | 100 | 563 | 7400 | 2,237634043 | 4,07938258 | 208,0485116 | 11010000 | -5,468542001 |
| 7 | Todos los Focos | 246 | 7320 | 2,213443404 | 4,035281147 | 205,7993385 | 11001101 | -3,458710333 |

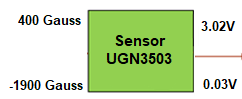
### **graficando mediciones ideales**

### **graficando mediciones reales**

**NOTA IMPORTANTE:** Sabemos que el valor de 25MΩ, siendo tan grande, causa una irregularidad, la cual es que el NE555 toma ese valor como circuito abierto, y en ese valor tiene una incongruencia, ya que marca algo aproximado a 0V, y debería de marcar un valor muy superior. Sin embargo, se ha incluido en estas gráficas, para que se note cual es la irregularidad.

### **BLOQUES DEL SENSOR DE MAGNETISMO**

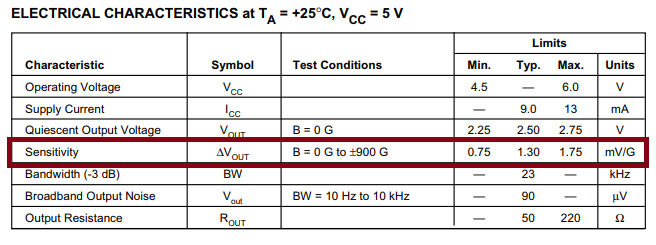
### **bloque SENSOR UGN3503**



En esta etapa del circuito se obtiene los valores de salida del sensor alimentándose con 5V en donde dada la sensibilidad del sensor (1.30mV/Gauss) que nos proporciona el fabricante, por lo cual se puede realizar las medidas de salida del sensor.

Se debe de tener en cuenta que la salida de este sensor es Radiométrica, lo que quiere decir que su voltaje de salida también depende de otra variable (no solo la física), en este caso depende del voltaje de polarización, ya que dependiendo con que voltaje se energice, su sensibilidad cambiara.

Eso se puede observar en el Datasheet:

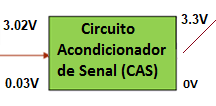


Por lo tanto, obtendremos esta grafica, aplicando los valores en Gauss, mínimo, intermedio y máximo.

|  |  |
| --- | --- |
| Gauss | V0 |
| -1900 | 0.03 |
| 0 | 2.5V |
| 400 | 3.02 |

Graficando…

### **bloque del circuito acondicionador de señal**



Entendiendo el bloque del UGN3503 podemos pasar al siguiente bloque. En el cual con los datos que recibimos del sensor, tenemos que amplificarlos para que en el siguiente bloque pueda recibir valores de 0 a 3.3 Volts.

Por lo que se realizaron los cálculos de la siguiente manera:

V(Tc) = Voltaje Generado por la salida del sensor (10mV/°C)

Tc = Temperatura en grados centígrados

Vo = Voltaje de Salida

Primero necesitamos obtener la pendiente de la ecuación, con lo cual usaremos la siguiente ecuación:

Ahora, tomando en cuenta los valores de la gráfica anterior, nos queda que:

Por lo tanto, la ecuación del CAS queda de la siguiente manera:

Por lo tanto, la ecuación nos queda de la siguiente manera:

En este caso tenemos a b = 0, entonces queda la ecuación:

Teniendo todo esto en cuenta, podemos tomar valores para comprobar que la ecuación es correcta:

Considerando la ecuación:

Tomaremos los tres principales voltajes:

|  |  |  |
| --- | --- | --- |
| Gauss | V0 | V0(Amplificado) |
| -1900 | 0.03 | 0 V |
| 0 | 2.5V | 2.73 V |
| 400 | 3.02 | 3.3 V |

### **PROPUESTA DE DISEÑO Y CALCULOS**

Considerando que del Bloque del sensor UGN3503, tenemos que diseñar un CAS que nos entregue 3.3V cuando el sensor este midiendo 400 Gauss, o bien 3.02 mV, y que entregue 0, cuando mida 0 -1900 Gauss o bien, 0V.

Entonces, teniendo en cuenta que la ecuación final fue:

La misma ecuación, nos está indicando que tenemos que pasarlo por un amplificador no inversor con ganancia de 1.. Por este motivo se eligió esta configuración del amplificador.

La fórmula del Amplificador No Inversor es:

Tenemos entendido que:

Con esto, podemos decir que:

Ahora, si proponemos un Rf de 1KΩ, la ecuación pasa a ser:

Despejando R1, tenemos que:

En este punto, ya hemos encontrado todos los valores de las resistencias, para que el amplificador tenga una ganancia de 1., probaremos la ecuación:

Tomaremos los tres principales voltajes:

|  |  |  |
| --- | --- | --- |
| Gauss | V0 | V0(Amplificado) |
| -1900 | 0.03 | 0 V |
| 0 | 2.5V | 2.73 V |
| 400 | 3.02 | 3.3 V |

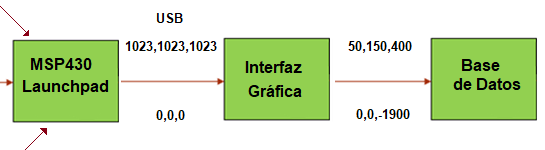
### **Mediciones**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Mediciones de Sensor UGN3503** | | | | | | |
|  |  | **MEDICIONES REALES** | | **MEDICIONES IDEALES** | |  |
| Mediciones: | Valor Gauss | V0 | V0(Amp) | V0 | V0(Amp) | ERROR |
| 1 | -1900 | 0 | 0 | 0 | 0 | 0 |
| 2 | -1200 |  | 1 | 0.94 | 1.027152318 | 2.64345583 |
| 3 | -500 |  | 1.9 | 1.85 | 2.021523179 | 6.01146601 |
| 4 | -200 |  | 2.2 | 2.24 | 2.447682119 | 10.1190476 |
| 5 | 0 | 2.5 | 2.73 | 2.5 | 2.731788079 | 0.06545455 |
| 6 | 50 |  | 2.85 | 2.565 | 2.80281457 | -1.68350168 |
| 7 | 100 |  | 2.9 | 2.63 | 2.87384106 | -0.91024312 |
| 8 | 200 |  | 3 | 2.76 | 3.01589404 | 0.52700922 |
| 9 | 300 |  | 3.14 | 2.89 | 3.15794702 | 0.56831289 |
| 10 | 400 | 3.02 | 3.3 | 3.02 | 3.3 | 0 |

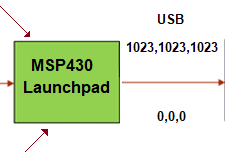
### **graficando mediciones ideales**

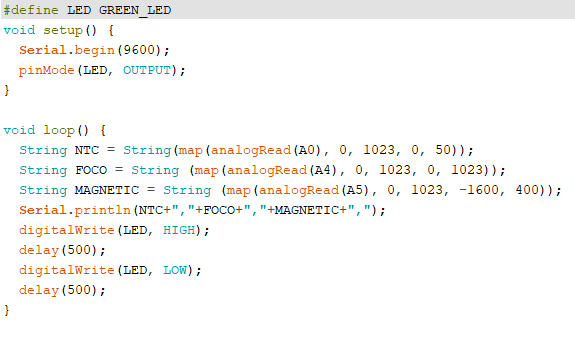
### **graficando mediciones REALES**

### **BLOQUES GENERALES DEL SISTEMA**



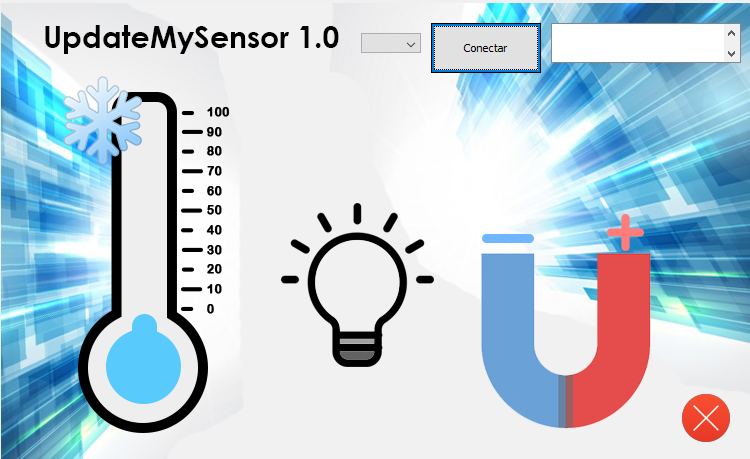
### **BLOQUE MSP430 LAUNCHPAD**





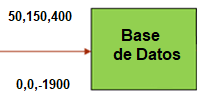
### **BLOQUE Interfaz gráfica**

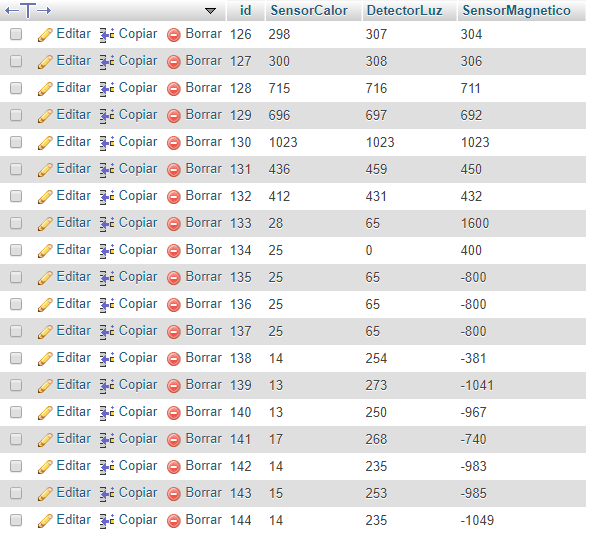




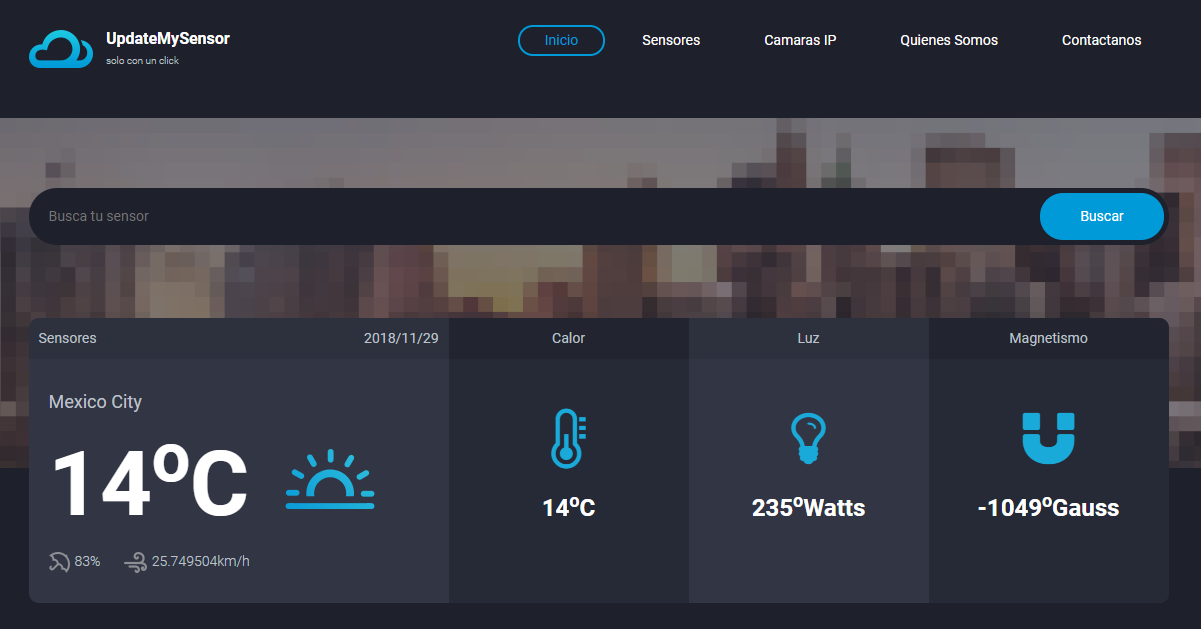
NOTA: Para ver los Códigos, consultar el apartado de ANEXOS

### **bloque base de datos**





### **BLOQUE EXTRA – PAGINA WEB**





### **Conclusiones**

# Jimenez muñoz Arvid

La realización de este proyecto fue interesante ya que fue una recopilación de las practicas hechas anteriores, a pesar de que ya contábamos con las practicas resulto complicado hacer que las 3 trabajaran en conjunto y al mismo tiempo, sin embargo, con poco de tiempo se pudo lograr.

Al momento de realizar el proyecto se nos presentaron algunos retos particulares que al inicio nos causaba confusión, pero retomando la teoría que se nos proporcionó en clase fue posible.

Particularmente sentí que lo más complicado fue el detector de luz, porque nos daba errores que no comprendí, pero basto con preguntarle al profesor y nos resolvió la duda comprendiendo la mejor solución para responder a dicho error.

Por otro lado este proyecto fue de mi agrado porque puse en práctica lo aprendido y como estudiante de ingeniería, determine que es posible desarrollar varios sistemas con diferentes funcionamiento pero, que son realmente útiles en la actualidad debido a que las necesidades de los humanos, nos compromete a innovar sistemas cada vez más eficientes, y la mejor forma de hacerlo empieza por comprender como funciona y como desarrollar estos sistemas, para que algún día lo más seguro será necesario su implementación.

Así dado por terminado el proyecto y teniendo un buen sabor de boca por el éxito de un sistema útil que dará el siguiente paso a nuevos proyectos.

# GUERRA VARGAS IRVING CRISTOBAL

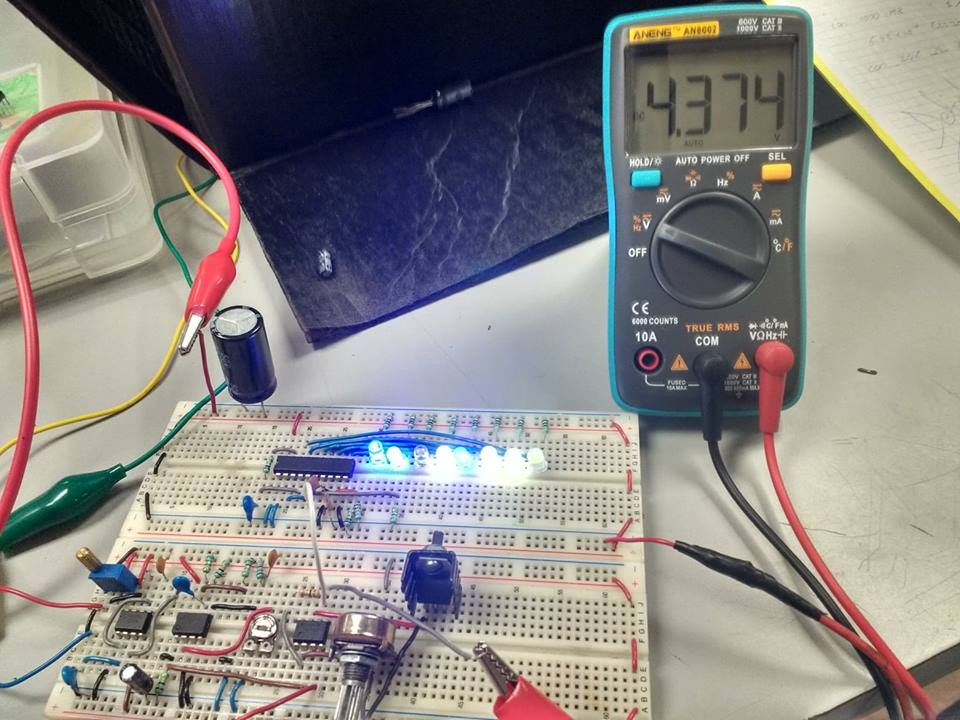
Con este proyecto tan completo, se juntaron practicas anteriores que ya se habían analizado para su acondicionamiento de señal, sin embargo lo interesante en esta práctica, fue proponer un sensor parecido a los cuales ya nos habían enseñado, claro, se selecciono un sensor lineal para que no se complicara demasiado la parte del acondicionamiento. Al elegir un sensor lineal y conocer su sensibilidad las cosas se simplifican mucho, sin embargo, algo particular de los sensores, es si tienen o no salidas radiométricas. Que son salidas que dependen de otra variable además de la física, en este caso, la fuente de alimentación.

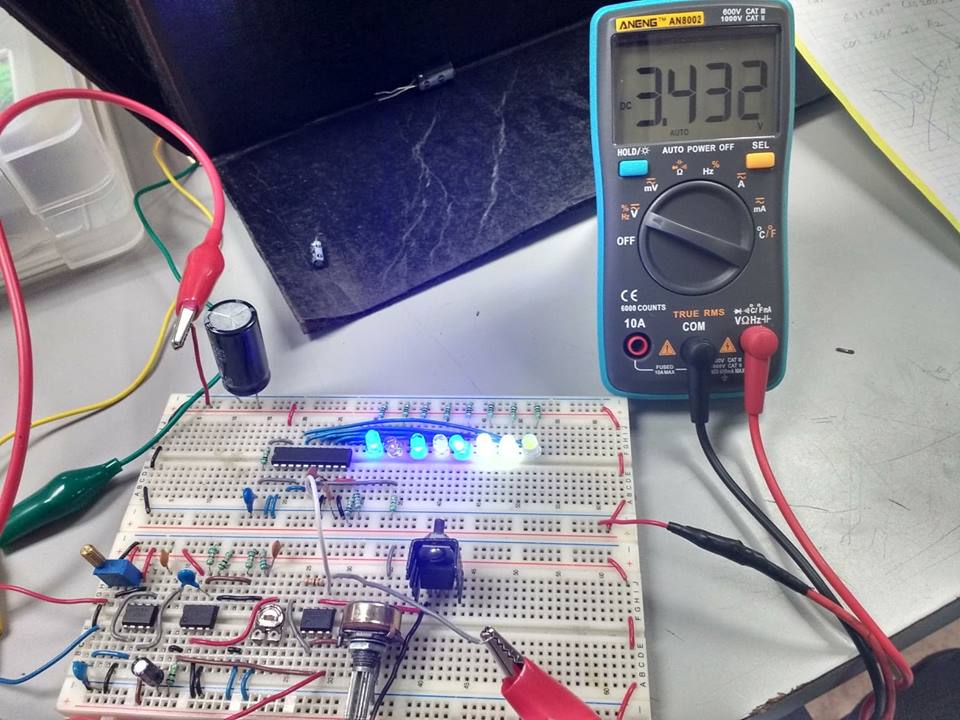
También aprendimos a usar la tarjeta de desarrollo Launchpad, que en lo personal es una tarjeta bastante cara para su aplicación. Es una tarjeta de desarrollo basada en Arduino, modificado por Texas Instruments. En lo personal hubiese elegido un Arduino de menor costo y mejor compatibilidad, sin embargo, ambas tarjetas de desarrollo funcionan a la perfección.

La interfaz gráfica fue también una parte sencilla, ya que fue en java, sin embargo siempre tuve la idea de hacerlo en PHP para si visualización en línea. Pero en PHP es imposible hacer una aplicación a tiempo real ya que su entorno es a servidores y estático. Por lo tanto, la animación se hizo en Java y en PHP consultas por internet a tiempo real.

Fue un proyecto que me gusto realizar ya que tiene una infinidad de aplicaciones reales y muy prácticas, puede ser un sistema de monitoreo de alguna casa o algún lugar.

### **evidencias y firmas**





### **anexos**

Principal.java

1. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("Conectado a el puerto:"+ " " + port);
2. **boolean** success = **false**;
3. [Icon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+icon) iconoCalor;
4. //Aqui configuramos los parámtros del puerto serial
5. SerialPort serialPort = **new** SerialPort(port);
6. **try** {
7. serialPort.openPort();
8. serialPort.setParams(
9. SerialPort.BAUDRATE\_9600,
10. SerialPort.DATABITS\_8,
11. SerialPort.STOPBITS\_1,
12. SerialPort.PARITY\_NONE);
13. serialPort.setEventsMask(MASK\_RXCHAR);
14. serialPort.addEventListener((SerialPortEvent serialPortEvent) -> {
15. **if**(serialPortEvent.isRXCHAR()){
16. **try** {
17. //Aqui realizamos la lectura del puerto COM
18. //Reibiremos Aproximadamente 10 datos para estabilizar las velocidades
19. **if**(firsTIME < 10 ){
20. primerString = serialPort.readString();
21. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string)[] Sensores = primerString.split(",");
22. [Thread](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+thread).sleep(1000);
23. firsTIME++;
24. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("[ TimeOff | OK | "+firsTIME+" ] Estabilizando Velocidades");
25. lblInfo.setText("[ TimeOff | OK | "+firsTIME+" ] Estabilizando Velocidades");
26. lblInfo.setHorizontalAlignment([SwingConstants](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+swingconstants).CENTER);
27. lblInfo.setVerticalAlignment([SwingConstants](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+swingconstants).CENTER);
29. [ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon) imagen = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(getClass().getResource("/Termometro/0.png"));
30. [Icon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+icon) icono = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(imagen.getImage().getScaledInstance(lblCalor.getWidth(),lblCalor.getHeight(),[Image](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+image).SCALE\_DEFAULT));
31. lblCalor.setIcon(icono);
33. imagen = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(getClass().getResource("/Luz/0.png"));
34. icono = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(imagen.getImage().getScaledInstance(lblLuz.getWidth(),lblLuz.getHeight(),[Image](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+image).SCALE\_DEFAULT));
35. lblLuz.setIcon(icono);
37. imagen = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(getClass().getResource("/Magnetismo/0.png"));
38. icono = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(imagen.getImage().getScaledInstance(lblIman.getWidth(),lblIman.getHeight(),[Image](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+image).SCALE\_DEFAULT));
39. lblIman.setIcon(icono);
40. }
41. **if**(firsTIME >= 10){
42. segundoString = serialPort.readString();
43. **if**(primerString.equals(segundoString)){
44. //No sera necesario imprimir y evitamos lagg
45. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("[ TimeOff | OK | "+firsTIME+" ] Mismo dato");
46. lblInfo.setText("[ TimeOff | OK | "+firsTIME+" ] Mismo dato");
47. [Thread](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+thread).sleep(1000);
48. firsTIME++;
49. }**else**{
50. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.print("[ TimeOff | OK | "+firsTIME+" ] Sensor Actualizado -> ");
51. lblInfo.setText("[ TimeOff | OK | "+firsTIME+" ] Sensor Actualizado -> ");
52. primerString = segundoString;
53. segundoString="";
54. rx.setText(primerString);
55. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string)[] parts = primerString.split(",");
56. Calor = parts[0];
57. Luz = parts[1];
58. Magnetico = parts[2];
60. //En esta parte ira la decodificacion a imagenes.
62. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) imgCalor = "";
63. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) imgWatts = "";
64. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) imgIman = "";
66. **int** leerCalor = [Integer](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+integer).parseInt(Calor);
68. **if**(leerCalor >= 0 && leerCalor < 5){
69. imgCalor = "/Termometro/0.png";
70. }**else** **if**(leerCalor >= 5 && leerCalor < 10){
71. imgCalor = "/Termometro/5.png";
72. }**else** **if**(leerCalor >= 10 && leerCalor < 15){
73. imgCalor = "/Termometro/10.png";
74. }**else** **if**(leerCalor >= 15 && leerCalor < 20){
75. imgCalor = "/Termometro/15.png";
76. }**else** **if**(leerCalor >= 20 && leerCalor < 25){
77. imgCalor = "/Termometro/20.png";
78. }**else** **if**(leerCalor >= 25 && leerCalor < 30){
79. imgCalor = "/Termometro/25.png";
80. }**else** **if**(leerCalor >= 30 && leerCalor < 35){
81. imgCalor = "/Termometro/30.png";
82. }**else** **if**(leerCalor >= 35 && leerCalor < 40){
83. imgCalor = "/Termometro/35.png";
84. }**else** **if**(leerCalor >= 40 && leerCalor < 45){
85. imgCalor = "/Termometro/40.png";
86. }**else** **if**(leerCalor >= 45 && leerCalor < 50){
87. imgCalor = "/Termometro/45.png";
88. }**else** **if**(leerCalor >= 50 && leerCalor < 55){
89. imgCalor = "/Termometro/50.png";
90. }**else** **if**(leerCalor >= 55 && leerCalor < 60){
91. imgCalor = "/Termometro/55.png";
92. }**else** **if**(leerCalor >= 60 && leerCalor < 65){
93. imgCalor = "/Termometro/60.png";
94. }**else** **if**(leerCalor >= 65 && leerCalor < 70){
95. imgCalor = "/Termometro/65.png";
96. }**else** **if**(leerCalor >= 70 && leerCalor < 75){
97. imgCalor = "/Termometro/70.png";
98. }**else** **if**(leerCalor >= 75 && leerCalor < 80){
99. imgCalor = "/Termometro/75.png";
100. }**else** **if**(leerCalor >= 80 && leerCalor < 85){
101. imgCalor = "/Termometro/80.png";
102. }**else** **if**(leerCalor >= 85 && leerCalor < 90){
103. imgCalor = "/Termometro/85.png";
104. }**else** **if**(leerCalor >= 90 && leerCalor < 95){
105. imgCalor = "/Termometro/90.png";
106. }**else** **if**(leerCalor >= 95 && leerCalor < 100){
107. imgCalor = "/Termometro/95.png";
108. }**else**{
109. imgCalor = "/Termometro/100.png";
110. }
112. **int** leerWatts = [Integer](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+integer).parseInt(Luz);
113. //El sensor de Luz cambia un poco su logica, ya que entrega:
114. // OBSCURIDAD | 25W | 40W | 60W | 75W | 100W | Total
115. ///   0V      |2.86 | 2.49| 2.46| 2.39| 2.36 | 2.29
116. //Amplificando GANANCIA = 1.1538461538461538
117. //    0V      | 3.3 | 2.87| 2.83| 2.75| 2.72 | 2.64
118. //BITS 3.3 - > 1023
119. // 0 | 1023 | 889.7 | 877.3 | 852.5 | 843.2 | 818.4
121. **if**(leerWatts>=0 && leerWatts<25){ //Oscuridad
122. imgWatts = "/Luz/0.png";
123. }**else** **if**(leerWatts>=25 && leerWatts<40){ //25W
124. imgWatts = "/Luz/1.png";
125. }**else** **if**(leerWatts>=40 && leerWatts<60){ //40W
126. imgWatts = "/Luz/2.png";
127. }**else** **if**(leerWatts>=60 && leerWatts<75){ //60W
128. imgWatts = "/Luz/3.png";
129. }**else** **if**(leerWatts>=75 && leerWatts<100){ //75W
130. imgWatts = "/Luz/4.png";
131. }**else** **if**(leerWatts>=100 && leerWatts<150){ //100W
132. imgWatts = "/Luz/5.png";
133. }**else** **if**(leerWatts>=150 && leerWatts<=500){ //LUZ TOTAL
134. imgWatts = "/Luz/6.png";
135. }

138. **int** leerGauss = [Integer](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+integer).parseInt(Magnetico);
140. **if**(leerGauss >= -1700 && leerGauss < -1500){
141. imgIman = "/Magnetismo/-7.png";
142. }**else** **if**(leerGauss >= -1500 && leerGauss < -1250){
143. imgIman = "/Magnetismo/-6.png";
144. }**else** **if**(leerGauss >= -1250 && leerGauss < -1000){
145. imgIman = "/Magnetismo/-5.png";
146. }**else** **if**(leerGauss >= -1000 && leerGauss < -750){
147. imgIman = "/Magnetismo/-4.png";
148. }**else** **if**(leerGauss >= -750 && leerGauss < -500){
149. imgIman = "/Magnetismo/-3.png";
150. }**else** **if**(leerGauss >= -500 && leerGauss < -250){
151. imgIman = "/Magnetismo/-2.png";
152. }**else** **if**(leerGauss >= -250 && leerGauss < -5){
153. imgIman = "/Magnetismo/-1.png";
154. }**else** **if**(leerGauss >= -5 && leerGauss < 5){
155. imgIman = "/Magnetismo/0.png";
156. }**else** **if**(leerGauss >= 5 && leerGauss < 70){
157. imgIman = "/Magnetismo/1.png";
158. }**else** **if**(leerGauss >= 65 && leerGauss < 130){
159. imgIman = "/Magnetismo/2.png";
160. }**else** **if**(leerGauss >= 130 && leerGauss < 195){
161. imgIman = "/Magnetismo/3.png";
162. }**else** **if**(leerGauss >= 195 && leerGauss < 260){
163. imgIman = "/Magnetismo/4.png";
164. }**else** **if**(leerGauss >= 260 && leerGauss < 325){
165. imgIman = "/Magnetismo/5.png";
166. }**else** **if**(leerGauss >= 325 && leerGauss < 390){
167. imgIman = "/Magnetismo/6.png";
168. }**else** **if**(leerGauss >= 390 && leerGauss < 1700){
169. imgIman = "/Magnetismo/7.png";
170. }

173. [ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon) imagen = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(getClass().getResource(imgCalor));
174. [Icon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+icon) icono = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(imagen.getImage().getScaledInstance(lblCalor.getWidth(),lblCalor.getHeight(),[Image](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+image).SCALE\_DEFAULT));
175. lblCalor.setIcon(icono);
177. imagen = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(getClass().getResource(imgWatts));
178. icono = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(imagen.getImage().getScaledInstance(lblLuz.getWidth(),lblLuz.getHeight(),[Image](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+image).SCALE\_DEFAULT));
179. lblLuz.setIcon(icono);
181. imagen = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(getClass().getResource(imgIman));
182. icono = **new** javax.swing.[ImageIcon](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+imageicon)(imagen.getImage().getScaledInstance(lblIman.getWidth(),lblIman.getHeight(),[Image](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+image).SCALE\_DEFAULT));
183. lblIman.setIcon(icono);
184. ////////////////////////////////////////////////
186. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("Calor:"+Calor+" | Luz:"+Luz+" | Magnetico:"+ Magnetico);
187. lblInfo.setText("Calor:"+Calor+" | Luz:"+Luz+" | Magnetico:"+ Magnetico);
188. [Thread](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+thread).sleep(1000);
189. firsTIME++;
190. }
191. }
192. **if**(firsTIME % 15 == 0){ //Cada minuto
193. conexion.conectar();
194. **int** ok = conexion.saveSensors(primerString);
195. **if**(ok==1){
196. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("[ SERVER | OK ] Datos guardados en BD\n");
197. lblInfo.setText("[ SERVER | OK ] Datos guardados en BD\n");
198. }**else**{
199. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("[ SERVER | FAIL ] Datos no se guardaron en BD\n");
200. lblInfo.setText("[ SERVER | FAIL ] Datos no se guardaron en BD\n");
201. }
202. conexion.cerrar();
203. }
204. } **catch** (SerialPortException ex) {
205. Logger.getLogger([Principal](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+principal).**class**.getName())
206. .log(Level.SEVERE, **null**, ex);
207. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("No se pudo recibir el dato");
208. } **catch** ([InterruptedException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+interruptedexception) ex) {
209. Logger.getLogger([Principal](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+principal).**class**.getName()).log(Level.SEVERE, **null**, ex);
210. }
211. }
212. });
214. sp = serialPort;
215. success = **true**;
216. } **catch** (SerialPortException ex) {
217. Logger.getLogger([Principal](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+principal).**class**.getName())
218. .log(Level.SEVERE, **null**, ex);
219. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println("Error al tratar de conectarse al puerto"+ port);
220. [JOptionPane](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+joptionpane).showMessageDialog(**null**,"No se pudo realizar la conexión, por favor seleccione otro puerto:"+ " " + port );
221. }
222. **return** success;

Conexión.java (Conecta a la base de datos)

1. **public** **class** Conexion {
2. **private** [Connection](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+connection) con = **null**;
3. **private** [Statement](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+statement) st; //Sentencia
4. **private** [ResultSet](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+resultset) rs; //Resultado
5. **void** conectar() {
6. //Se entabla Conexion SQL
7. **try** {
8. **Class**.forName("com.mysql.jdbc.Driver");
9. **try** {
10. con = [DriverManager](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+drivermanager).getConnection("jdbc:mysql://idev4u.com:3306/idevucom\_instrumentacion", "idevucom","\*\*");
11. st = con.createStatement();
12. } **catch** ([SQLException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+sqlexception) e) {
13. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println(e.getMessage());
14. }
15. } **catch** ([ClassNotFoundException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+classnotfoundexception) ex) {
16. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println(ex.getMessage());
17. }
18. }
20. **void** cerrar() {
21. **try** {
22. con.close();
23. } **catch** ([SQLException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+sqlexception) e) {
24. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println(e.getMessage());
25. }
26. }

29. **int** saveSensors([String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) primerString) {
30. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string)[] parts = primerString.split(",");
31. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) Calor = parts[0];
32. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) Luz = parts[1];
33. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) Magnetico = parts[2];
34. [String](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+string) sentenciaSQL = "INSERT INTO lecturas(`SensorCalor`,`DetectorLuz`,`SensorMagnetico`) VALUES ('"+Calor+"','"+Luz+"','"+Magnetico+"')";
35. **try** {
36. st.executeUpdate(sentenciaSQL);
37. **return** 1;
38. } **catch** ([SQLException](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+sqlexception) ex) {
39. [System](http://www.google.com/search?hl=en&q=allinurl%3Adocs.oracle.com+javase+docs+api+system).out.println(ex.getMessage());
40. }
41. **return** 0;
42. }
44. }