El sistema de comunicación por luz visible propuesto consta de dos partes, el transmisor (Tx) y el receptor (Rx).

Por un lado, el transmisor se compone principalmente de un circuito de modulación, el cual tomará la señal del sensor y la convertirá en pulsos modulados para control del emisor de luz. Por otra parte, el receptor tiene como componente principal al circuito de demodulación, el cual se ocupa de convertir la señal obtenida en el sensor de luz visible en las mediciones originales, y posteriormente mostrar dichas mediciones al usuario a través de su interfaz correspondiente.

La salida del transmisor será luz codificada en el espectro visible que tendrá la función de generar pulsos de encendido y apagado pudiendo traducirse este a código digital binario aplicando el cstándar Manchester Diferencial aplicado en ingeniería de redes podremos generar una función de codificación y decodificación digital del código capaz de detectar errores para la prevención de datos erróneos.

El mecanismo general de transformar información en algo que la represente y que sea apto para su transmisión por un medio cualquiera se denomina codificación, y a esos que representan la información se les conoce con el nombre de señales, en este caso la luz que será la señal por la cual se enviará información codificada y se recibirá la misma en un receptor sensible a la luz, para su posterior decodificación e interpretación.

Se escoge la codificación digital para este debido a que los sistemas digitales en cuanto al tratamiento de señales nos otorgan posibilidades de detección de errores, inmudidad al ruido en este caso también el ruido de la señal podrá provenir de fuente de luz en un ambiente con múltiples fuente de luz como es nuestro caso una modulación por componentes analógicos sería imposible.

La única desventaja de este sistema se resume en que se requiere línea vista entre los dispositivos de transmisión y recepción de luz, por tanto no deben existir obstáculos en medio.

Diagrama general del sistema.

### Transmisor

Para el diseño del transmisor, en primer lugar se debe escoger el circuito de modulación. Como se puede observar en la figura, se propone la utilización del raspberry pi zero como codificador de la señal, ya que cuenta con un tamaño compacto, posee entradas analógicas para los sensores y salidas digitales para el emisor de luz, capaces de conmutar a altas frecuencias. Posteriormente se debe seleccionar los sensores de temperatura y de humedad, los cuales deben poseer salidas analógicas. Finalmente, se debe seleccionar un LED con su respectiva protección para el emisor de luz visible.

Diagrama general del transmisor Tx.

**Emisor de luz**

Para el emisor de luz, como se puede observar en la figura, se utilizará un diodo emisor de luz LED de color blanco de 10mm de alta luminosidad. Adicionalmente, es necesaria la utilización de una resistencia de protección, para lo cual se debe tomar en cuenta las siguientes características del LED:

* Caída de voltaje – 2.5V.
* Corriente – 20mA.

Por lo tanto la resistencia de protección será:

Es así que se escoge una resistencia superior a 125Ω, en nuestro caso 220Ω



LED de luz blanca

### Receptor

Para el diseño del receptor, en primer lugar se debe escoger el circuito de demodulación. Como se puede observar en la figura, el propósito es utilizar el Raspberry Pi como circuito demodulador de la señal, ya que posee puertos de propósito general (GPIO) capaces de recibir la señal del sensor de luz y las características computacionales para operar como computadora central en una red de domótica.

Posteriormente se debe seleccionar el sensor de luz visible, el cual debe poseer una salida en voltaje y una elevada velocidad de conmutación. Así mismo, como se observa en la figura, nuestro diseño del sensor cuenta con un panel solar como transductor, un comparador de voltaje para la detección y conversión de los pulsos y un circuito de protección para evitar picos de tensión en el Raspberry Pi. Finalmente, como interfaz de usuario se realizará un software para mostrar las medidas del sensor en el Raspberry, es por ello que son requeridos periféricos de entrada como mouse para la navegación del usuario y un display para poder observar los resultados.

Diagrama general del receptor Rx. (Elaboración propia)

**Sensor de luz**

Como se mencionó anteriormente, el sensor de luz propuesto consta de un panel solar, un comparador de voltaje y un circuito de protección. El primero de ellos, el panel solar, el cual puede ser observado en la figura, cuenta con las siguientes características:

* Voltaje de salida – 0-5.5V.
* Corriente – 100mA.
* Potencia – 0.55W
* Dimensiones – 80mm x 55mm



Panel solar de 5V.

Por otro lado, en la tabla, podemos observar el comportamiento del panel solar midiendo diferentes frecuencias de conmutación del emisor. Cabe destacar que dichos datos fueron obtenidos de manera experimental posterior a la implementación del transmisor. Los factores a tomar en cuenta fueron la percepción visual, es decir si la conmutación del LED es notoria, y la detección, es decir si el panel solar es capaz de devolver un voltaje diferenciado entre los estados de encendido y apagado del LED. Es así que, para el protocolo utilizado se optó por una frecuencia de conmutación de 400 Hz.

|  |  |  |  |
| --- | --- | --- | --- |
| **Periodo (ms)** | **Frecuencia (Hz)** | **Percepción** | **Detección** |
| 160 | 6.25 | Alto | Alta |
| 80 | 12.5 | Alto | Alta |
| 40 | 25 | Alto | Alta |
| 20 | 50 | Alto | Alta |
| 10 | 100 | Medio | Alta |
| 5 | 200 | Leve | Alta |
| 2.5 | 400 | Leve | Media |
| 1.25 | 800 | Ninguna | Mala |
| 0.625 | 1600 | Ninguna | Mala |
| 0.3125 | 3200 | Ninguna | Mala |

Percepción y detección del sensor a diferentes frecuencias (Elaboración propia)

Ahora bien, la señal devuelta por el panel solar está dentro del 0-5.5V, sin embargo los cambios de voltaje entre los estados de encendido y apagado del LED no necesariamente abarcan la totalidad de dicho rango. Es por ello, que es necesaria la implementación de un circuito comparador para devolver una salida digital de 0 a 5V. Como se puede observar en la figura 18, el circuito integrado seleccionado para esta tarea es el LM358P, un amplificador operacional el cual se utilizará en modo comparador.



Amplificador operacional LM358P.

Finalmente, es requerida la utilización de un circuito de protección en la entrada del Raspberry Pi, debido a que este es sensible a los picos de voltaje y sus entradas digitales operan hasta los 3.3V. Es por ello, como se observa en la figura 19, que se opta por un diodo zener de 3.3V

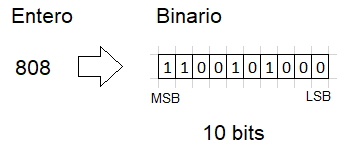


Diodo Zener de 3.3V.

## Diseño de la comunicación

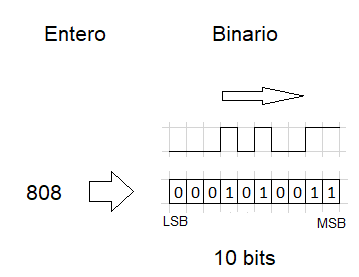
### Lectura del sensor

El primer paso en el diseño del protocolo de comunicación a utilizar, es determinar el número de bits con los que se trabajará inicialmente. Como se utilizará un sensor, es necesario tomar en cuenta las características de las entradas del raspberry pi cero. Es así, como se puede observar en la figura, se realiza la lectura dentro de una palabra de 10 bits, que se pueden expresar en números enteros o binarios.

****

Formato de datos del sensor en entrada analógica.

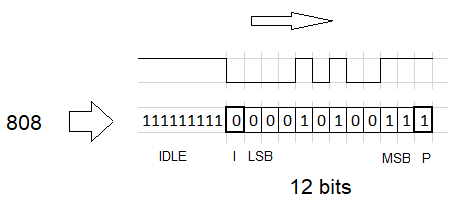
Cabe destacar, que es importante tomar en cuenta la posición de los bits más significativos (MSB) y los menos significativos (LSB). Al momento de expresar el valor como número binario, el MSB se encuentra al lado izquierdo y el LSB al lado derecho. Sin embargo en nuestro protocolo, se enviará la cadena de datos empezando por el LSB y terminando por el MSB, es por ello que, como se muestra en la figura, el flujo de datos se expresa con el LSB al lado izquierdo y el MSB al lado derecho. Así mismo también se puede observar una representación gráfica de la señal.

****

Flujo de datos transmitidos.

### Bits de inicio y parada

Como es en el caso de muchos protocolos asíncronos, en el diseño propuesto es necesario incluir un bit de inicio y un bit de parada, para que el receptor pueda identificar que bloques de datos debe tomar en cuenta para actualizar la lectura del sensor. Es así, como se puede observar en la figura, el diseño cuenta con un bit de inicio con valor de 0, el cual romperá con la cadena de valores 1 en el estado IDLE o de espera. Por otro lado, el bit de parada contará con valor de 1, indicando el fin del bloque de datos. En consecuencia, nuestro bloque de datos ahora cuenta con 12 bits.

****

Bits de inicio y parada.

### Codificación

La codificación de los datos resulta de gran utilidad, ya que nos permite tener niveles de redundancia que permiten verificar si el valor obtenido es correcto o presenta alguna interferencia, en cuyo caso se descarta. En la figura, podemos observar el algoritmo utilizado para realizar la codificación por el método de Manchester Diferencial.

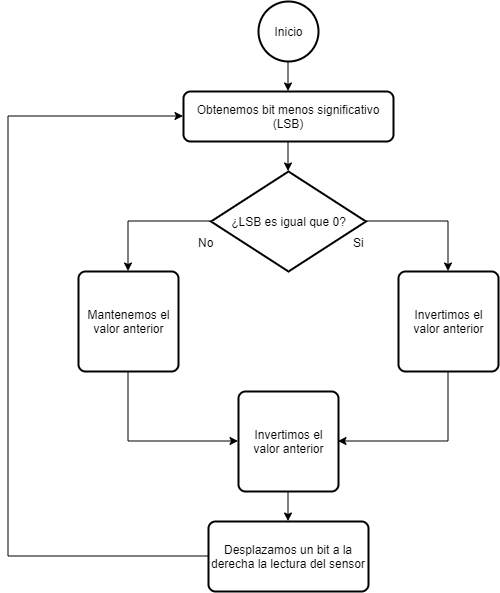
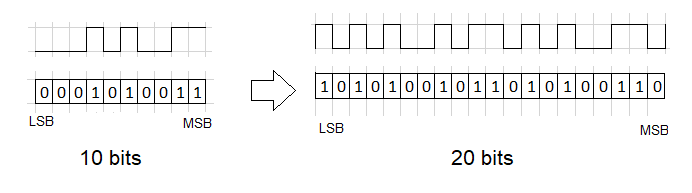
****

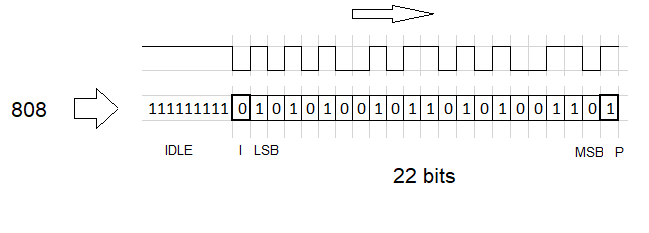
Diagrama de flujo de la codificación por Manchester Diferencial.

Como se mencionó anteriormente, este método consiste en codificar 1 bit como 2 bits, tomando en cuenta los valores anteriores. De ese modo, nuestro algoritmo inicia por leer el bit menos significativo de la lectura del sensor. En caso de ser un valor de 0, este es codificado por medio de dos transiciones, tomando en cuenta el valor anterior. Por otro lado, en el caso de recibir un 1, este es codificado por medio de la copia del nivel anterior y una codificación. Debido a que el bit de inicio es 0, en caso de que el LSB sea 0, este será codificado con 10 y en el caso de que sea un 1, este será codificado con un 01. Finalmente se realiza un desplazamiento hacia la derecha de la palabra leída por el sensor, cambiando así al siguiente bit.

****

Ejemplo de codificación por Manchester Diferencial.

Como se puede observar en la figura, lo que obtenemos efectivamente es una conversión de una palabra de 10 bits a una de 20 bits. Por lo tanto, como se puede observar en la figura, el bloque de datos final a ser transmitido cuenta con 22 bits, incluyendo los de inicio y parada que mantienen su lugar al principio y al final del bloque, sin ser codificados.

****

Flujo completo de datos a transmitir.

### Decodificación

Por otro lado, en lo que se refiere al receptor, el circuito de demodulación espera a recibir la señal del bit de inicio para poder identificar el bloque correcto. Una vez obtenido el bloque, procede a la decodificación de los 20 bits en el formato de Manchester diferencial. Es así que, como se puede observar en el algoritmo expuesto en la figura, iniciamos con un contador i, el cual representa el índice de la palabra codificada, iniciada en 0. Posteriormente comparamos el valor de la palabra codificada en el índice i con el siguiente valor (i + 1). Si los valores son iguales, el valor decodificado equivale a 1, caso contrario equivale a 0. Finalmente se aumenta el índice en 2 pasos y se recorre toda la cadena codificada. Cabe destacar que la primera comparación se realiza entre el bit de inicio y el bit LSB.

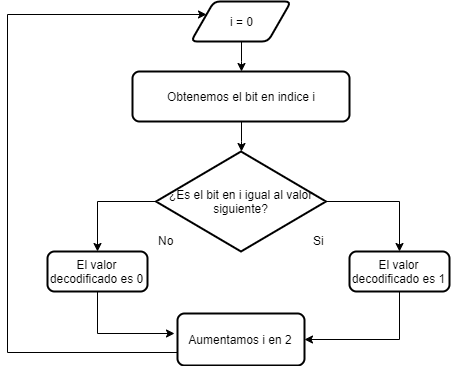
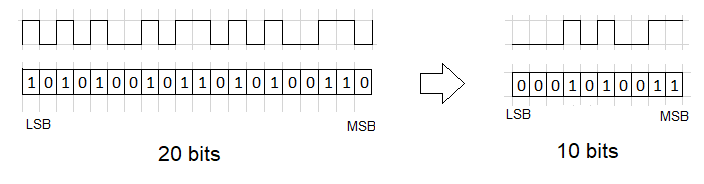
****

Diagrama de flujo de la decodificación por Manchester Diferencial.

Por lo tanto, como lo indica la figura, el proceso nos devuelve el valor original de 10 bits a partir de los 20 bits codificados, tomando en cuenta el valor del bit de inicio. Por último, el demodulador debe representar el valor obtenido en un número que el usuario pueda interpretar, es por ello que realiza una conversión a un número entero, tomando en cuenta la posición del LSB y MSB y posteriormente realiza la normalización de los datos a valores de temperatura y humedad en grados centígrados y porcentaje respectivamente.

****

Ejemplo de decodificación por Manchester Diferencial.

## Diseño del transmisor

### Circuito electrónico

Como se puede observar en la figura, el diseño eléctrico del transmisor no requiere de muchas conexiones. En primer lugar, se conecta la alimentación de 5V del Raspberry pi cero al cable rojo del sensor. En segundo lugar se conecta el GND del Raspberry pi cero a la resistencia de protección del LED y el cable negro del sensor. En tercer lugar se conecta el cable amarillo de humedad a la entrada. Finalmente se conecta el ánodo del LED a la salida digital D10 del Raspberry pi cero.

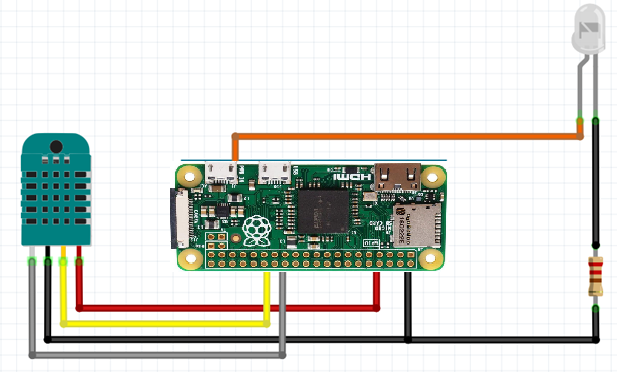
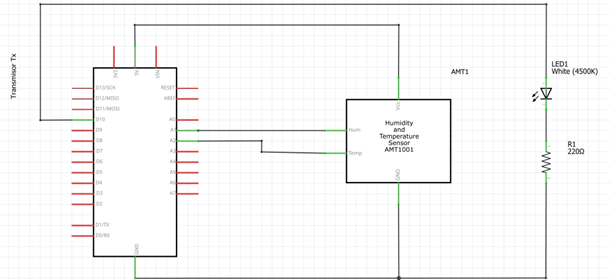
****

Diagrama del circuito del trasmisor.

Por otro lado, en la figura, podemos observar una representación en esquemático del circuito transmisor expuesto.

****

Esquemático del transmisor.

### Programación

El programa ejecutado dentro del Raspberry pi se muestra a continuación:

int analogValue;

int bitValue;

float periodo = 2.5; //400Hz

byte datos[22];

while True:

{

pinMode(13, OUTPUT);

Serial.begin(9600);

//Send Idle

digitalWrite(13,HIGH);

delay(100);

//Read Sensor

analogValue = analogRead(A0);

Serial.println(analogValue);

//Start bit

datos[0] = 0;

//Datos (Manchester Diferencial)

for(int i=2; i<21; i+=2)

{

bitValue = analogValue & 1;

if(bitValue == 0)

datos[i-1] = !datos[i-2];

else

datos[i-1] = datos[i-2];

datos[i] = !datos[i-1];

analogValue = analogValue >> 1;

}

//Stop bit

datos[21] = 1;

for(int i=0; i<22; i++)

{

digitalWrite(13,datos[i]);

Serial.print(datos[i]);

delay(periodo/2);

}

Serial.println("");

}

## Diseño del receptor

### Circuito electrónico

Para el diseño del receptor, el primer factor a tomar en cuenta es el rango de voltajes que devuelve el panel solar en un determinado ambiente. Una vez conocido dicho rango, el positivo del panel solar es conectado a la entrada no inversora del LM358P, para su comparación. Posteriormente, se conecta un potenciómetro entre 5V y GND del Raspberry Pi para realizar un divisor de tensión y, tomando en cuenta el rango anterior, se regula el potenciómetro al valor adecuado y se conecta su señal a la entrada inversora del comparador para actuar como referencia.

Una vez realizada la comparación el LM358P, nos devuelve una tensión de 5V cuando el panel detecta al LED encendido y 0V cuando lo detecta apagado. Sin embargo, Para la entrada del Raspberry Pi, es necesario regular dicha salida a un valor de 3.3V. Es así que, por medio de la implementación de un diodo zener conectado entre la entrada GPIO del Raspberry Pi y tierra para evitar picos de voltaje y una resistencia de 330Ω para regular la corriente, podemos regular dicha tensión de entrada a valores seguros para el Raspberry.

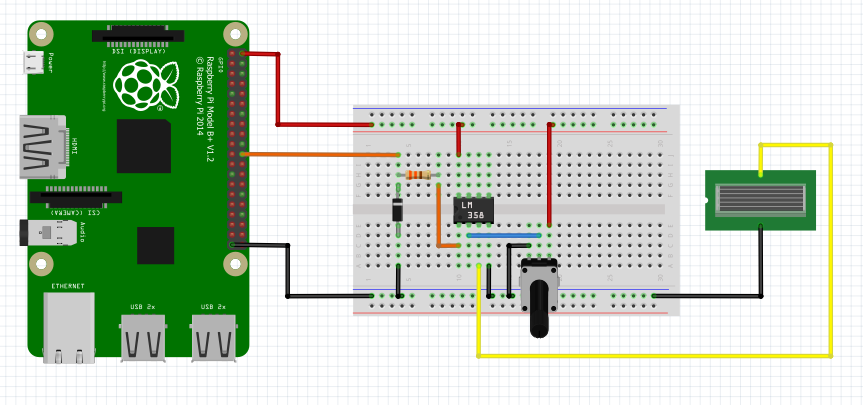
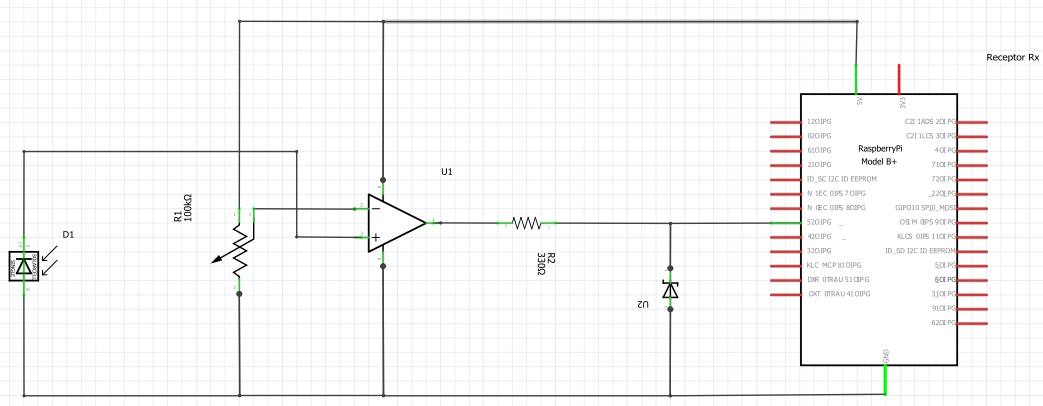


Diagrama del circuito del receptor.

Por otro lado, en la figura podemos observar el diagrama de conexión del circuito receptor. Finalmente en la figura, podemos observar el esquemático de dicho circuito.

****

Esquemático del transmisor.

### Programación

El programa ejecutado dentro del Raspberry Pi se muestra a continuación:

int x

float periodo = 2.5 #400Hz

byte datos[22]

byte resultBits[10]

int resultValue = 0

int bitValue

GPIO.setmode(GPIO.BCM)

GPIO.setup(25,GPIO.IN)

while True:

x = GPIO.input(25);

if(x == 0):

Serial.println("");

datos[0] = x;

Serial.print(datos[0]);

time.sleep(period\*1000/2);

for(int i = 1; i < 22; i++):

x = digitalRead(25);

digitalWrite(13,x);

datos[i] = x;

Serial.print(datos[i]);

time.sleet(period\*1000/2);

Serial.println("")

# Decode

for(int i = 0; i < 20; i+=2):

if(datos[i] != datos[i+1]):

bitValue = 0;

else:

bitValue = 1

Serial.print(bitValue);

resultBits[i/2] = bitValue

Serial.println("");

for(int i = 9; i >= 0; i--):

bitValue = resultBits[i];

resultValue = resultValue | bitValue;

if(i != 0):

resultValue = resultValue << 1;

Serial.println(resultValue);

resultValue = 0;

else:

Serial.print(x);

time.sleep(period\*1000/2);

GPIO.cleanup()

## Integración del sistema

Una vez implementado el transmisor y el receptor, estos deben ser posicionados de manera que el LED emisor y el Panel solar estén en la misma línea de vista. Es así que, en la figura 34, podemos observar un diagrama representativo del sistema completo.

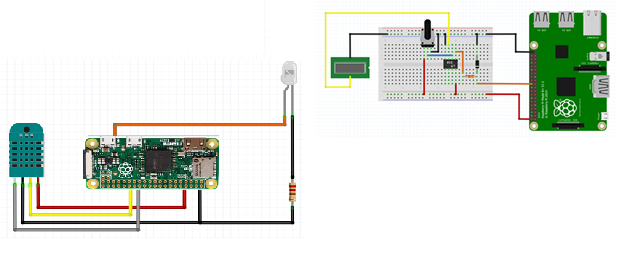
****

Diagrama del sistema integrado.