UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS

(*Universidad del Perú*, Decana de América)

FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

Curso: Internet de las Cosas

Tema: Aplicación de semáforos inteligentes haciendo uso de simulación en CISCO packet tracer 7.3.1

Docente: **Herrera Quispe, Jose Alfredo**

Grupo: hardworKINGS

Integrantes:

- **Palomino Loa, Junior Jose** **18200172**
- **Sánchez Saldaña, Néstor Joaquín** **18200191**
- **Sánchez Muñoz, José Anderson** **18200236**

LIMA – PERÚ

2021

Índice

Homework 1

Diseño (Idea previa) 2

Materiales usados 3

Cambios realizados 4

Capturas del programa 5

Análisis del código 7

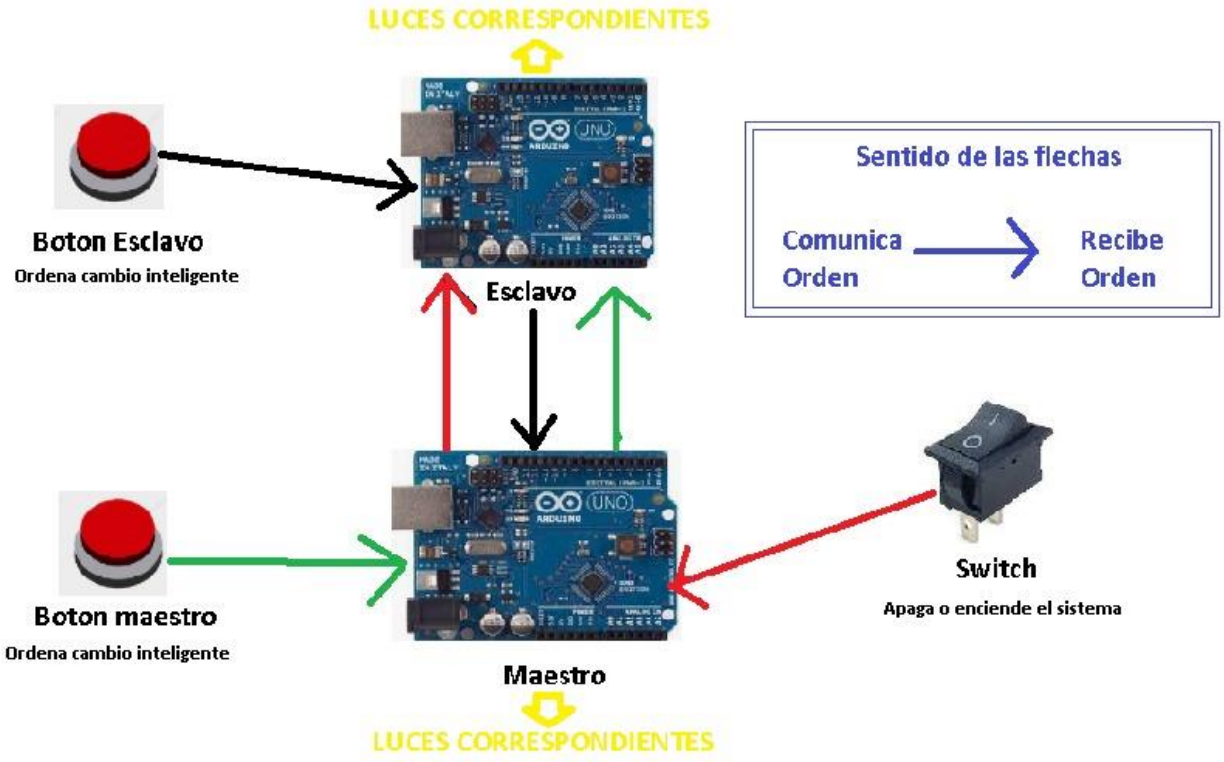
Bibliografía 13

Homework

Smart traffic Light

- Program an Smart traffic light using Arduino over thinkercad or other
  - The traffic light usually is green for 4000 ms, yellow for 500 ms, and red for 4000 ms.
  - Green or red light starts to flash fast 500ms before changing
  - If it is green and a switch is pressed after 1000ms seconds it will automatically change to red with the flashing procedure.Others that you consider intelligent!

Diseño (Idea previa)



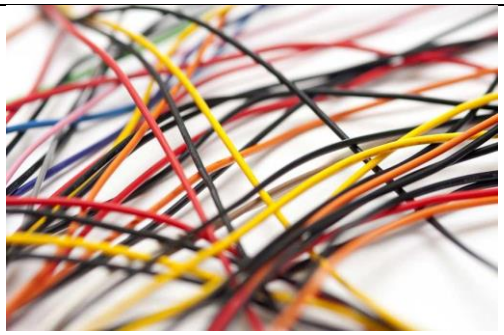
1- El switch envía una señal al **ARDUINO MAESTRO** (si se enciende o se apaga) y esta será repetida al **ARDUINO ESCLAVO**, ejecutando la misma orden.

2- El **botón maestro** envía una señal al **ARDUINO MAESTRO** para cambiar de luz (roja a verde), permitiendo así el paso del peatón. Esta señal se replica al **ARDUINO ESCLAVO**, haciendo que este cambie también (verde a roja), evitando así que se pueda producir un accidente.

3- El **botón esclavo** envía una señal al **ARDUINO ESCLAVO** para cambiar de luz (roja a verde), permitiendo así el paso del peatón. Esta señal se replica al **ARDUINO MAESTRO**, haciendo que este cambie también (verde a roja), evitando así que se pueda producir un accidente.

4- Tanto el **ARDUINO MAESTRO** como el **ARDUINO ESCLAVO** dirigen el comportamiento de un **semáforo peatona**l y uno **automovilístico**, permitiendo que estos enciendan sus luces correspondientes en el tiempo indicado.

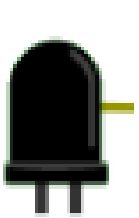
Materiales usados

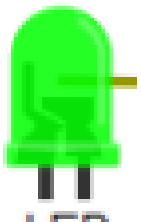
 **Cables (x16)**	 **Botones (x2)**
 **Switch (x1)**	 **Arduino (x2)**

Cambios realizados

LED'S

Por defecto los LED'S en en packet tracer tienen la forma de uno convencional, como se conseguiría en las tiendas.

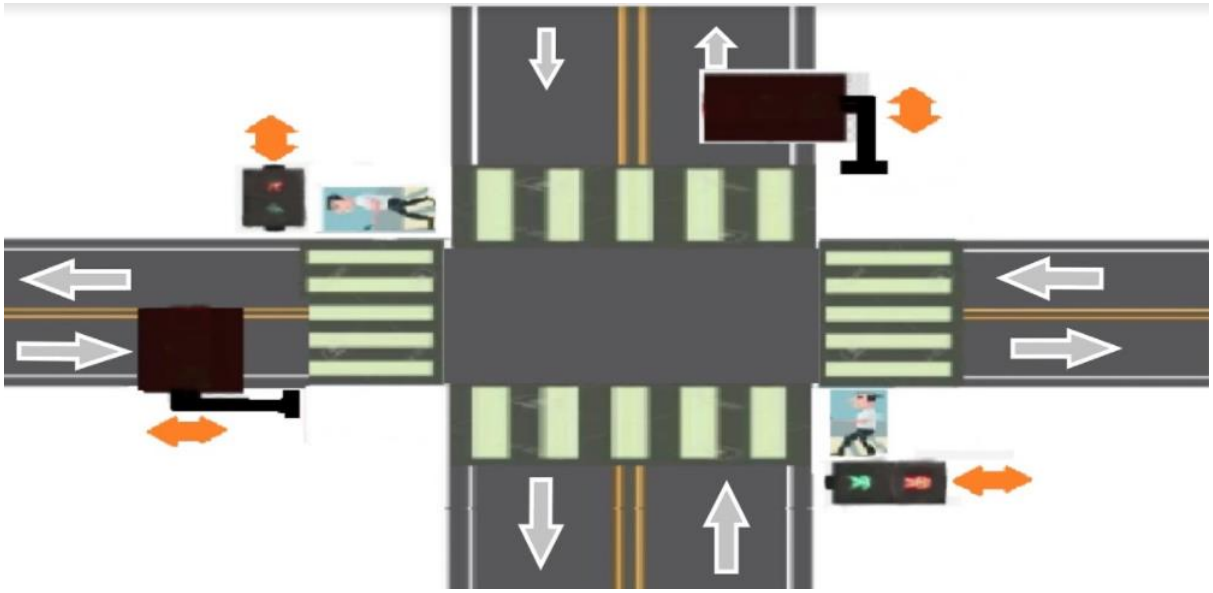




Sin embargo, packet tracer nos ofrece la posibilidad de cambiar el formato de este, por lo que lo aprovecharemos para darles una forma más parecida a lo que deseamos (luces de semáforo), para ello usaremos plantillas (estas se encuentran en el drive).

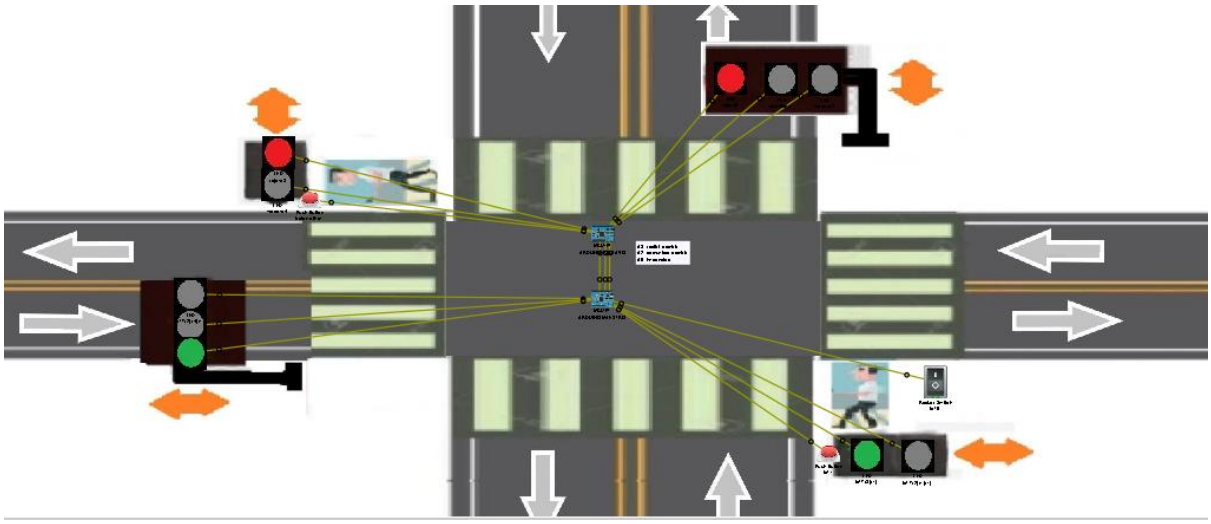
Fondo

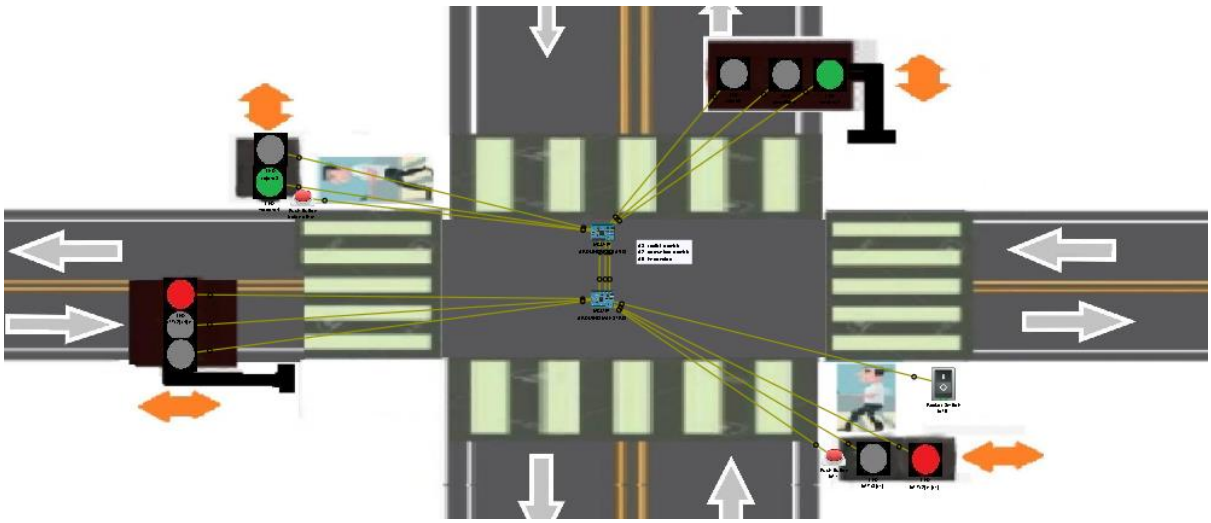
Packet tracer nos permite agregar un fondo que nos ayudará a una simulación más apreciable (este se encontrará en el drive).



Capturas del programa

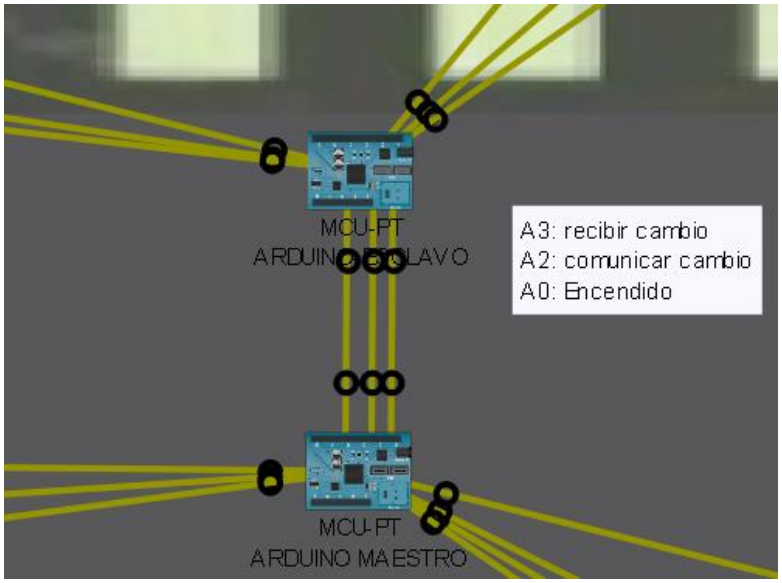
Vista del proyecto en packet tracer 7.3.1



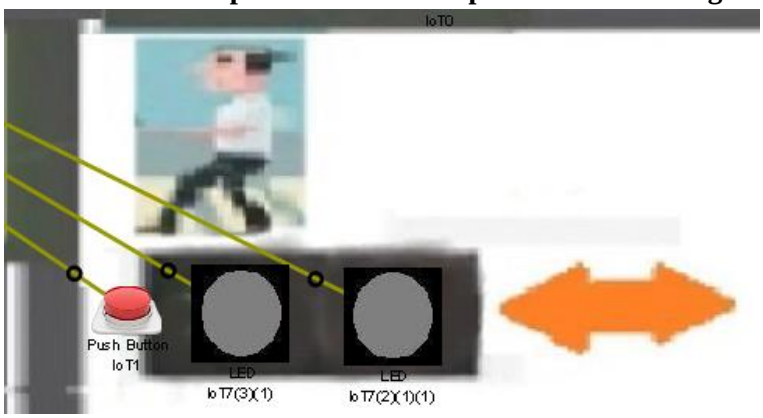


(*) La simulación se encuentra en el video “Demo - Semaforos inteligentes (hardworKINGS).mp4” (se encontrará en el drive)

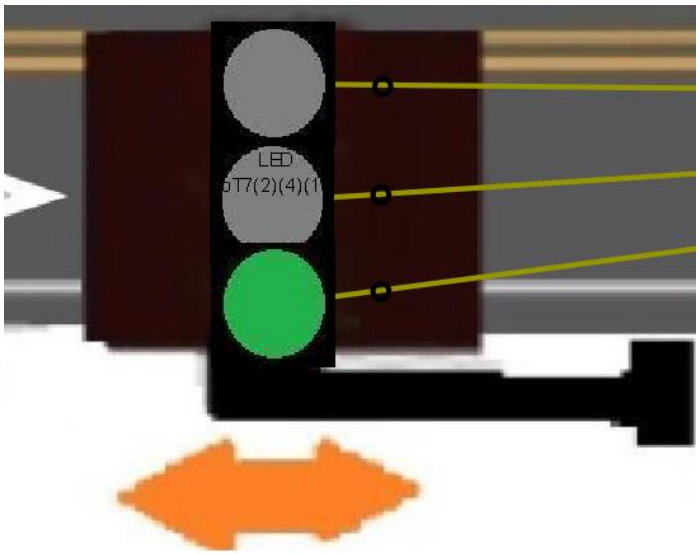
Arduinos conectados



Vista del semáforo peatonal con su respectivo botón inteligente



Vista del semáforo automovilístico



Análisis del código

Nota: Se tomará como referencia el código del ARDUINO MAESTRO para la explicación.

1- Declaramos los identificadores que correspondan a los pines para su fácil identificación.

```
//Pines pertenecientes a los LEDS de los semaforos automovilisticos
    var RA = 0; //LED ROJO AUTO
    var AA = 1; //LED AMBAR AUTO
    var VA = 2; //LED VERDE AUTO

//Pines pertenecientes a los LEDS de los semaforos peatonales
    var RP = 3; //LED ROJO PERSONA
    var VP = 4; //LED VERDE PERSONA

//Pin perteneciente al switch que enciende o apaga el sistema
    var switchEncendido = 5; //Switch que enciende o apaga el sistema
//Pin comunicador sobre el encendido del sistema
    var comunicarEncendido = A0;
//Boton de cambio
    var botonCambio = A1;
//Comunicación de los cambios en los semaforos inteligentes
    var comunicarCambio = A2;
    var recibirCambio = A3;
```

2- Declaramos los identificadores que correspondan a los tiempos de los que hará uso cada luz sea el caso.

```
//Tiempos entre los cambios de las luces
    var tRojo = 4000;
    var tAmbar = 500;
    var tVerde = 4000;
    var tCambio = 1000; //Cambio especial del semaforo inteligente

//Tiempo de parpadeo en el semaforo peatonal antes del cambio
    var tpRojo = 500;
    var tpVerde = 500;
```

3- Declaramos los identificadores que correspondan a los estados de los semáforos (luces ejecutándose)

```
/*Estados de los semaforos (tomando de referencia el automovilistico)
  1. LUZ VERDE
  2. LUZ AMBAR (En el caso del s. peatonal iniciará el parpadeo)
  3. LUZ ROJA*/
    var estadoInicio = 1; //1. LUZ VERDE 2.LUZ AMBAR 3. LUZ ROJA
    var estadoSiguiente;
    var estadoActual;
```

(*) Nota: En este caso el ARDUINO MAESTRO inicia en 1 (luz verde) mientras que el ARDUINO ESCLAVO iniciará en 3 (luz roja).

4- Declaramos la función correspondiente a la acción de parpadeo en las luces en ejecución.

Parámetros:

- luzParpadeante: luz que va a realizar el parpadeo
- tparpadeo: es el tiempo en milisegundos que durará el parpadeo

```
//Parpadeo de la luz antes del cambio
function parpadeo(luzParpadeante,tparpadeo){
    var intervalo = 5; //Intervalo entre parpadeos
    var txparpadeo = tparpadeo/intervalo; //Tiempo entre parpadeos
    digitalWrite(luzParpadeante, LOW);
    delay(txparpadeo);
    digitalWrite(luzParpadeante, HIGH);
    delay(txparpadeo);
    digitalWrite(luzParpadeante, LOW);
    delay(txparpadeo);
    digitalWrite(luzParpadeante, HIGH);
    delay(txparpadeo);
    digitalWrite(luzParpadeante, LOW);
    delay(txparpadeo);
}
```

5-Declaramos la función correspondiente a la acción de encender y apagar las luces.

Parámetros:

- luzA: luz correspondiente al semáforo automovilístico
- luzP: luz correspondiente al semáforo peatonal
- tLuz: tiempo en que la luz estará encendida
- tparpadeo: tiempo en que la luz estará parpadeando
- estadoActual: estado que informa que luces se están ejecutando

```
//Encendido y apagado de una luz VERDE o ROJA
function luzVR(luzA, luzP, tLuz,tparpadeo,estadoActual){
    var t; //Controla cuanto tiempo ha pasado en el semaforo rojo
    var bc=LOW; //Detecta la señal entrante al presionar el boton de cambio
    var rc=LOW; //Recepciona la señal entrante en caso el otro arduino emita señal de cambio
    //Encendemos las luces de ambos semaforos
    digitalWrite(luzA, HIGH);
    digitalWrite(luzP, HIGH);
    //Si la luz es roja y puede darse un cambio inteligente a verde
    if(estadoActual==3){
        t=0;
        /*Si la luz encendida es roja se debe retrasar esta lo que demoraria la luz ambar
        en los otros semaforos (para una mejor sincronización)*/
        while(t<(tLuz-tparpadeo+tAmbar)){
            bc=digitalRead(botonCambio);
            if(bc==HIGH){
                digitalWrite(comunicarCambio,HIGH);
                return 4; //Cambio rojo a verde (modo inteligente)
            }
            t=t+10;
            delay(10);
        }
```

```
    } else if(estadoActual==1) { //Si la luz es verde y puede darse un cambio inteligente a roja
        t=0;
        while(t<(tLuz-tparpadeo)){
            rc=digitalRead(recibirCambio);
            if(rc==HIGH){
                return 5; //Cambio de verde a rojo (modo inteligente)
            }
            t=t+10;
            delay(10);
        }
    }
    parpadeo(luzP,tparpadeo);
    //Apagamos las luces de ambos semaforos
    digitalWrite(luzA, LOW);
    digitalWrite(luzP, LOW);
    //Actualizamos los estados
    if(estadoActual==1){
        return 2; //Verde pasará a ambar
    }else if(estadoActual==3){
        return 1; //Rojo pasará a verde
    }
}
```

(*) Nota: El curso normal de esta función cambia en caso se detecte que se presionó el botón que activa el modo inteligente (sea que ocurriera en el ARDUINO MAESTRO o en el ESCLAVO)

6- Esta función permite el prendido y el apagado de la luz ámbar de los semáforos automovilísticos, a su vez de proceder con la función de “parpadeo” de los semáforos peatonales. Toma como parámetros de referencia:

- luzA: luz correspondiente al semáforo automovilístico
- luzP: luz correspondiente al semáforo peatonal
- tLuz: tiempo en que la luz estará encendida
- tparpadeo: tiempo en que la luz estará parpadeando
- estadoActual: estado que informa que luces se están ejecutando

```
//Encendido y apagado de una luz ambar
function luzAmbar(luzA, luzP, tLuz,tparpadeo,estadoActual){
    //Encendemos las luces de ambos semaforos
    digitalWrite(luzA, HIGH);
    delay(tLuz-tparpadeo); //Le reducimos el tiempo en el que entrara en parpadeo

    /*Si la luz es ambar entonces la luz verde peatonal parpadeará simulando el paso por ambar antes del
    //cambio posterior a rojo*/
    parpadeo(luzP,tparpadeo);

    //Apagamos la
    digitalWrite(luzA, LOW);

    //Actualizamos los estados
    return 3; //Ambar pasa a rojo
}
```

7- Esta función ejecuta el cambio de luz cuando se presiona los botones situados en los semáforos peatonales. El único caso donde se presenta este es cuando el semáforo peatonal se encuentra en rojo y se busca el cambio de este a verde para poder cruzar. Consecuentemente los semáforos automovilisticos se adaptan al cambio, pasando de su **estado de verde a rojo**, permitiendo un paso seguro para el transeúnte. Toma como parámetros de referencia:

- luzA: luz correspondiente al semáforo automovilístico
- luzP: luz correspondiente al semáforo peatonal
- tLuz: tiempo en que la luz estará encendida
- tparpadeo: tiempo en que la luz estará parpadeando
- estadoActual: estado que informa que luces se están ejecutando

```
//Cambio inteligente, rojo a verde (si emite) o verde a rojo (si recibe)
function cambioInteligente(luzA, luzP, tLuz,tparpadeo,estadoActual){
    //Encendemos las luces de ambos semaforos
    digitalWrite(luzA, HIGH);
    digitalWrite(luzP, HIGH);

    delay(tLuz-tparpadeo); //Le reducimos el tiempo en el que entrara en parpadeo

    /*Si la luz encendida es roja se debe retrasar esta lo que demoraria la luz ambar
    en los otros semaforos (para una mejor sincronización)*/
    if(estadoActual==3)delay(tAmbar);

    parpadeo(luzP,tparpadeo);

    //Apagamos las luces de ambos semaforos
    digitalWrite(luzA, LOW);
    digitalWrite(luzP, LOW);

    //Actualizamos los estados
    if(estadoActual==4){
        return 1; //Si es rojo pasará a verde
    }else {
        return 2; //si es verde pasa a rojo. Si es rojo pasa a verde (Independientemente si es el cambio inteligente o no)
    }
}
```

(*) Nota: tomar en cuenta que el tiempo que tarda en cambiar un semáforo automovilístico que se encuentra en rojo a verde debe de considerar el tiempo de la luz ámbar para evitar una desincronización futura con los semáforos peatonales. Además, se debe reflejar el parpadeo en las luces de estos.

8- Se define en el setup los modos de uso de cada pin, sean estos del tipo INPUT u OUTPUT.

```
function setup() {
    pinMode(RA, OUTPUT); //ROJO AUTO
    pinMode(AA, OUTPUT); //AMBAR AUTO
    pinMode(VA, OUTPUT); //VERDE AUTO

    pinMode(RP, OUTPUT); //ROJO PERSONA
    pinMode(VP, OUTPUT); //VERDE PERSONA

    pinMode(switchEncendido, INPUT); //SWITCH DE ENCENDIDO
    pinMode(comunicarEncendido, OUTPUT); //EMISOR DE ENCENDIDO

    pinMode(botonCambio, INPUT); //BOTON DE CAMBIO INTELIGENTE
    pinMode(comunicarCambio, OUTPUT); //EMISOR DE CAMBIO INTELIGENTE
    pinMode(recibirCambio, INPUT); //RECEPTOR DE CAMBIO INTELIGENTE

}
```

9- En esta función se va a tener en cuenta el código que se va a ejecutar continuamente. Para el caso del arduino Maestro el if-else va a depender del estado de switchEncendido ya que dependiendo de esto, el sistema se enciende o se apaga.Y en el caso del arduino Esclavo el if-else va a depender de la variable recibir Encendido y no tendrá la opción de comunicar al arduino Maestro que es lo que debe hacer.

Dentro del if , hay un switch que dependiendo del estadoActual representado principalmente por los casos 1 , 2 y 3 que son los colores de las luces.
Hay que tener en cuenta que los casos de luz verde y roja hay una función que les permite ingresar los parámetros del tipo de luz del semáforo de autos y personas, el tiempo de cada uno de estos semáforos y el estado actual para que calculen el estado siguiente al que serían asignados.

Los casos 4 y 5 son casos especiales ya que dependen de los botones ubicados en los semáforos de peatones.
El caso 4 es el caso del botón del arduino Maestro, cuando el semáforo del peatón esté en rojo, es necesario realizar un cambio inteligente, para que el semáforo pase a verde.
El caso 5 es el caso del botón del arduino Esclavo, cuando el semáforo del otro extremo está en rojo, es necesario realizar un cambio inteligente, para que el semáforo pase a verde.

Dentro del else, significa que el switch estará apagado por lo que se apagan los semáforos de ambos tipos, y el estadoActual toma el valor inicial de estadoInicio.

```
function loop() {
    //Determinamos si el circuito enciende o se queda apagado a traves del switch principal
    var encender = digitalRead(switchEncendido);
    //Encender el sistema
    if (encender == HIGH){
        digitalWrite(comunicarEncendido,HIGH); //Comunicamos al otro Arduino que se encienda
        switch (estadoActual){
        //Luz verde
        case 1: estadoSiguiente = luzVR(VA,VP,tVerde,tpVerde,estadoActual);
                break;
        //Luz ambar
        case 2: estadoSiguiente = luzAmbar(AA,VP,tAmbar,tpVerde,estadoActual);
                break;
        //Luz roja
        case 3: estadoSiguiente = luzVR(RA,RP,tRojo,tpRojo,estadoActual);
                break;
        //Cambio inteligente de rojo a verde
        case 4: digitalWrite(comunicarCambio,LOW);
                estadoSiguiente = cambioInteligente(RA,RP,tCambio,tpRojo,estadoActual);
                break;
        //Cambio inteligente de verde a rojo
        case 5: digitalWrite(recibirCambio,LOW);
                estadoSiguiente = cambioInteligente(VA,VP,tCambio,tpVerde,estadoActual);
                break;
        }
        //Actualizamos el estado
        estadoActual = estadoSiguiente;
    }else{
        digitalWrite(comunicarEncendido,LOW); //Comunicamos al otro Arduino que se apague
        //Apagar sistema
        digitalWrite(RA, LOW);
        digitalWrite(RP, LOW);
        digitalWrite(VA, LOW);
        digitalWrite(VP, LOW);
        digitalWrite(AA, LOW);
        //Volver al estado de inicio
        estadoActual = estadoInicio;
    }
}
```

(*) Nota: Las líneas de código relacionadas a la variable **“comunicarEncendido”** sólo están presentes en el ARDUINO MAESTRO, siendo que el ARDUINO ESCLAVO tendrá el **“recibirEncendido”** que le comunicará cuando este se encienda o apague para replicar su comportamiento.

Bibliografía

- Universidad Católica de Colombia. (2019). Arquitectura IoT para la Prestación del Servicio de Semaforización Inteligente en Bogotá. https://repository.ucatolica.edu.co/bitstream/10983/23709/1/Semaforizacion%20Inteligente%20_625387.pdf

Repositorios

- Drive: https://drive.google.com/drive/folders/1KTNkEpWKQi7qAKWYnGSYfuuV3DGTtvUx?usp=sharing

- GitHub: https://github.com/juniorPalomino25/IOT