Automatización de controles remotos

Descripción:

Este proyecto se enfoca en la replicación y automatización de señales de control remoto infrarrojo (IR) para dispositivos domésticos sin acceso a internet, como aires acondicionados o televisores. La idea es aprender sobre comunicación, IR, principios de electrónica básica y programación en Python, con la finalidad de diseñar un dispositivo usando la placa Micro:bit de forma tal que pueda “aprender” comandos IR específicos y ejecutarlos automáticamente en horarios predefinidos, por ejemplo, para encender o apagar un equipo.

Sitio web:

En el siguiente enlace encontrarán todo lo necesario para el seguimiento del proyecto, así como también sus avances y desafíos en tiempo real. La idea es que el sitio web sea una bitácora en la cual se van viendo los pasos seguidos a lo largo del proyecto.

https://steam.wago.uy/

Materiales utilizados:

- Placa Arduino.
- Modulo receptor IR
- LED IR (emisor)
- Computadora
- Control remoto (de TV, Aire Acondicionado, etc.)
- Resistencias de 150 ohms
- Cables dupont macho-hembra
- Botón pulsador
- Opcional:
  - Protoboard (no se usó)
  - Transistor NPN (BC547 y resistencia) (no se usó)
  - Reloj RTC (DS3231 con módulo IC2) (no se usó)

Detalle de avances:

31/agosto/2025

- Configuración de servidor web para servir sitio web del proyecto.
- Creación del boceto del sitio web
    - Planificación del mismo
    - Realización del boceto en código HTML
- Subida del boceto al servidor para pruebas
- Creación de documentación para entregar como primera entrega (este documento)
- Investigación de viabilidad del proyecto
    - Investigación de herramientas alternativas
    - Viabilidad de realizar el proyecto con microbit

06/setiembre/2025

- Creación del documento de avances final (este documento).

10/ setiembre /2025

- Primeros pasos en la implementación del circuito. Para ello se requirió de lo siguiente:
    - 1 x Placa Arduino Uno o 1 x Placa Micro:bit.
    - 1 x Sensor receptor de infrarrojos (IR), por ejemplo un TSOP1738. Este componente es el que capturará la señal de tu control remoto.
    - 1 x Diodo emisor de infrarrojos (IR LED). Este componente se usa para reproducir la señal grabada.
    - 1 x Resistencia de 220 Ω para limitar la corriente del IR LED.
    - 1 x Pulsador. Lo usarás para alternar entre los modos de "grabación" y "reproducción".
    - 1 x Resistencia de 10 kΩ para el pull-down del pulsador.
    - Cables de conexión (jumpers).

- **Receptor IR (TSOP1738):**
    - Pin OUT al pin digital del Arduino (por ejemplo, pin 11) o Micro:bit.
    - Pin GND al pin GND de la placa.
    - Pin VCC al pin de 5V del Arduino.
- **IR LED:**
    - Pata más larga (Ánodo) a la resistencia de 220 Ω.
    - El otro extremo de la resistencia a un pin digital de la placa (por ejemplo, pin 3).
    - Pata más corta (Cátodo) al pin GND de la placa.
- **Pulsador:**
    - Un terminal al pin digital de la placa (por ejemplo, pin 2).
    - El otro terminal a una resistencia de 10 kΩ.
    - El otro extremo de la resistencia al pin GND de la placa.
    - El terminal que va al pin digital del Arduino también se conecta al pin 5V.

20/setiembre/2025

- En esta instancia se prosiguió con la implementación del proyecto. Dado que ya teníamos el circuito armado, nos enfocamos más en la creación del código.
- Nuevamente nos apoyamos en IA para obtener un primer pantallazo del código.

27/setiembre/2025

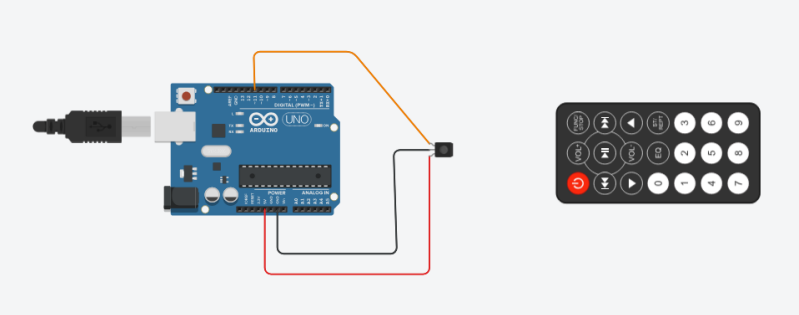
- Nuevamente retomamos el circuito y el código anterior con la finalidad de rever cualquier inconveniente que pudiera tener, tanto en el armado del circuito, como en el código visto anteriormente.
- Dado que el protobard me confunde más de lo que me ayuda, decidí quitarla del medio y conectar directamente.

04/octubre/2025

- Debido a que lo anterior no se pudo hacer funcionar, he decidido investigar si esto es realmente posible de hacer.
- Para ello se buscó en YouTube con el fin de ver si alguien ya ha intentado hacer esto antes. Para mi sorpresa encontré estos dos videos que muestran como lo han hecho otras personas.

04/octubre/2025

- Se da con alguien en YouTube que intenta hacer lo mismo utilizando una placa Arduino.
- Se emula lo visto resultando en lo siguiente:



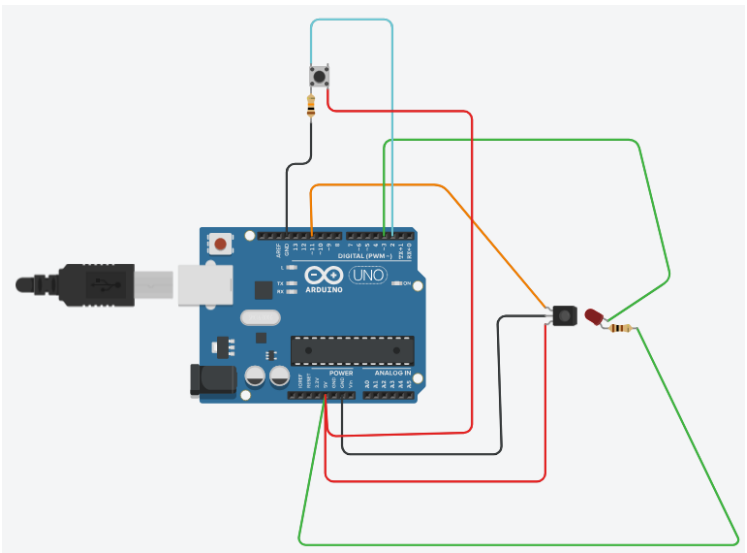
- Codigo utilizado:

```
#include <IRremote>

int RECV_PIN = 11;

// No necesitas crear objetos "irrecv" o "results" aquí
arriba.
// La nueva librería maneja esto de forma interna.

void setup()
{
  Serial.begin(9600);
  Serial.println("Habilitando receptor IR...");
```

```
  // Inicia el receptor en el pin 11.
  // Reemplaza a irrecv.enableIRIn()
  IrReceiver.begin(RECV_PIN, ENABLE_LED_FEEDBACK); //
ENABLE_LED_FEEDBACK es opcional, puedes quitarlo.

 Serial.println("Receptor Habilitado.");
}

void loop()
{
  // Comprueba si se recibió una señal
  // Reemplaza a if (irrecv.decode(&results))
  if (IrReceiver.decode()) {

   // Imprime el valor recibido en HEX
   // Reemplaza a Serial.println(results.value, HEX);
   Serial.println(IrReceiver.decodedIRData.decodedRawData,
HEX);

   // Prepara el receptor para la siguiente señal
   // Reemplaza a irrecv.resume();
   IrReceiver.resume();
  }
  delay(100);
}
```

07/noviembre/2025

- Mejora del circuito para alcanzar objetivo:



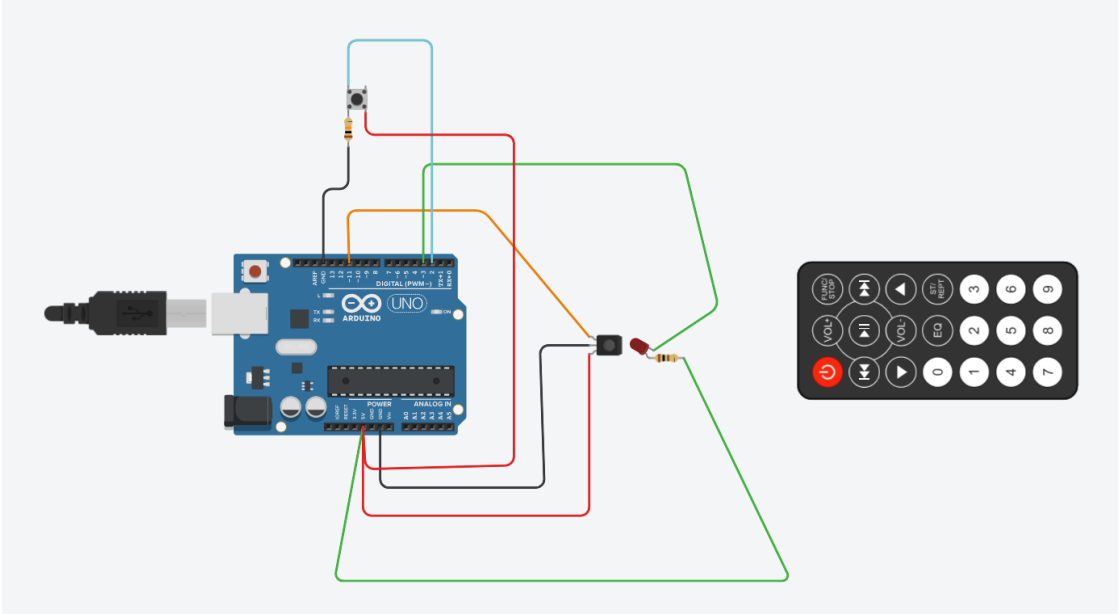
- Codigo utilizado:

```
#include

// Pines
int RECV_PIN = 11;  // Pin del Receptor IR
int SEND_PIN = 3;   // Pin del Emisor IR (LED IR)
int BUTTON_PIN = 2; // Pin de un botón para enviar la señal
guardada

// Variables para guardar el código recibido
unsigned long lastReceivedCode = 0;
// NOTA: Para simplificar y evitar errores de librería en
Tinkercad,
// hardcodearemos el protocolo a NEC (32 bits) si la
decodificación es exitosa.
// Si tu control remoto no es NEC, debes cambiar este
protocolo (ej. SONY, RC5).
const int PROTOCOL_BITS = 32; // Común para NEC

bool buttonState = false;

void setup() {
  Serial.begin(9600);
  pinMode(BUTTON_PIN, INPUT_PULLUP);

  // 1. Inicializar Receptor IR: Pasar el pin de recepción
y la bandera opcional.
  IrReceiver.begin(RECV_PIN, ENABLE_LED_FEEDBACK);

  // 2. Inicializar Transmisor IR: Pasar solo el pin de
envío.
  IrSender.begin(SEND_PIN);

  Serial.println("--- Sistema IR Habilitado ---");
  Serial.println("Presione un boton en el control remoto
para CAPTURAR el codigo.");
  Serial.println("Presione el boton en el pin 2 para
REPRODUCIR el codigo capturado.");
}

void loop() {
  // --- 1. RECEPCIÓN Y ALMACENAMIENTO ---
  if (IrReceiver.decode()) {

    // Solo guardar si la decodificación fue exitosa y no
es código repetido
```

```
    if (IrReceiver.decodedIRData.protocol != UNKNOWN &&
IrReceiver.decodedIRData.decodedRawData != 0) {
        lastReceivedCode =
IrReceiver.decodedIRData.decodedRawData;

        Serial.print("✅ Codigo CAPTURADO: ");
        Serial.print(lastReceivedCode, HEX);
        Serial.print(" (Protocolo: ");
        // Usamos IrReceiver.decodedIRData.protocol para
saber qué protocolo es
        Serial.print(IrReceiver.decodedIRData.protocol);
        Serial.println(")");
    }

    IrReceiver.resume(); // Reanudar la recepción
  }

  // --- 2. REPRODUCCIÓN (al presionar un botón) ---
  if (digitalRead(BUTTON_PIN) == LOW && !buttonState) {
    buttonState = true; // Botón presionado

    if (lastReceivedCode != 0) {
      Serial.print("📡 ENVIANDO Codigo: ");
      Serial.println(lastReceivedCode, HEX);

      // Función de envío NEC (necesita el código y el
número de bits)
      // Si tu control remoto no usa 32 bits, ajusta la
constante PROTOCOL_BITS
      IrSender.sendNEC(lastReceivedCode, PROTOCOL_BITS);

      Serial.println("Señal Enviada.");
    } else {
      Serial.println("⚠️ No se ha capturado ningun codigo
para enviar.");
    }
  }

  // Resetear el estado del botón cuando se suelta
  if (digitalRead(BUTTON_PIN) == HIGH && buttonState) {
    buttonState = false;
  }

  delay(100);
}
```

10/noviembre/2025

- Actualización del circuito para alcanzar objetivo:



- Codigo utilizado:

#include

```
// Pines
int RECV_PIN = 11;  // Pin del Receptor IR
int SEND_PIN = 3;   // Pin del Emisor IR (LED IR)
int BUTTON_PIN = 2; // Pin de un botón para enviar la señal
guardada

// Variables para guardar el código recibido
unsigned long lastReceivedCode = 0;
// *** CORRECCIÓN ***: Renombramos la constante para evitar
conflicto con la librería.
// Protocolo RC-5 (Protocolo 8) usa típicamente 12 bits.
const int RC5_NUM_BITS = 12;

bool buttonState = false; // Estado del botón

void setup() {
  Serial.begin(9600);
  // Usamos INPUT_PULLUP, el botón se conecta a GND
  pinMode(BUTTON_PIN, INPUT_PULLUP);

  // 1. Inicializar Receptor IR
  IrReceiver.begin(RECV_PIN, ENABLE_LED_FEEDBACK);

  // 2. Inicializar Transmisor IR
  IrSender.begin(SEND_PIN);

  Serial.println("--- Sistema IR Habilitado ---");
  Serial.println("Presione un boton en el control remoto
para CAPTURAR el codigo.");
  Serial.println("Presione el boton en el pin 2 para
REPRODUCIR el codigo capturado.");
}

void loop() {
  // --- 1. RECEPCIÓN Y ALMACENAMIENTO ---
  if (IrReceiver.decode()) {

    // Solo guardar si la decodificación fue exitosa y no
es código repetido
    if (IrReceiver.decodedIRData.protocol != UNKNOWN &&
IrReceiver.decodedIRData.decodedRawData != 0) {
        lastReceivedCode =
IrReceiver.decodedIRData.decodedRawData;
```

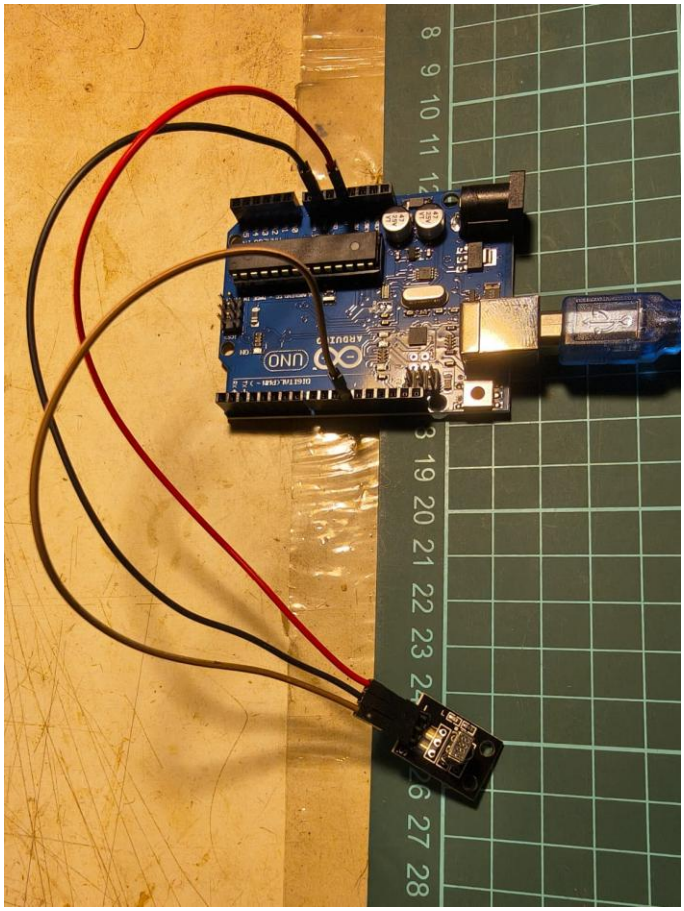
```
        Serial.print("✅ Codigo CAPTURADO: ");
        Serial.print(lastReceivedCode, HEX);
        Serial.print(" (Protocolo: ");
        Serial.print(IrReceiver.decodedIRData.protocol);
        Serial.println(" - ¡RC-5!)");
        Serial.println(">>> Captura exitosa. Presione el
botón en Pin 2 para REPRODUCIRlo. <<<");
    }

    IrReceiver.resume(); // Reanudar la recepción
  }

  // --- 2. REPRODUCCIÓN (al presionar un botón) ---
  if (digitalRead(BUTTON_PIN) == LOW && !buttonState) {
    buttonState = true; // Botón presionado

    if (lastReceivedCode != 0) {
      Serial.print("📡 ENVIANDO Codigo: ");
      Serial.println(lastReceivedCode, HEX);

      // Usamos el nuevo nombre de la constante
      IrSender.sendRC5(lastReceivedCode, RC5_NUM_BITS);

      Serial.println("Señal Enviada.");
    } else {
      Serial.println("⚠️ No hay codigo capturado valido
para enviar.");
    }
  }

  // Resetear el estado del botón cuando se suelta
  if (digitalRead(BUTTON_PIN) == HIGH && buttonState) {
    buttonState = false;
  }

  delay(100);
}#include

// Pines
int RECV_PIN = 11;  // Pin del Receptor IR
int SEND_PIN = 3;   // Pin del Emisor IR (LED IR)
int BUTTON_PIN = 2; // Pin de un botón para enviar la señal
guardada

// Variables para guardar el código recibido
unsigned long lastReceivedCode = 0;
```

```
// *** CORRECCIÓN ***: Renombramos la constante para evitar
conflicto con la librería.
// Protocolo RC-5 (Protocolo 8) usa típicamente 12 bits.
const int RC5_NUM_BITS = 12;

bool buttonState = false; // Estado del botón

void setup() {
  Serial.begin(9600);
  // Usamos INPUT_PULLUP, el botón se conecta a GND
  pinMode(BUTTON_PIN, INPUT_PULLUP);

  // 1. Inicializar Receptor IR
  IrReceiver.begin(RECV_PIN, ENABLE_LED_FEEDBACK);

  // 2. Inicializar Transmisor IR
  IrSender.begin(SEND_PIN);

  Serial.println("--- Sistema IR Habilitado ---");
  Serial.println("Presione un boton en el control remoto
para CAPTURAR el codigo.");
  Serial.println("Presione el boton en el pin 2 para
REPRODUCIR el codigo capturado.");
}

void loop() {
  // --- 1. RECEPCIÓN Y ALMACENAMIENTO ---
  if (IrReceiver.decode()) {

    // Solo guardar si la decodificación fue exitosa y no
es código repetido
    if (IrReceiver.decodedIRData.protocol != UNKNOWN &&
IrReceiver.decodedIRData.decodedRawData != 0) {
        lastReceivedCode =
IrReceiver.decodedIRData.decodedRawData;

        Serial.print("✅ Codigo CAPTURADO: ");
        Serial.print(lastReceivedCode, HEX);
        Serial.print(" (Protocolo: ");
        Serial.print(IrReceiver.decodedIRData.protocol);
        Serial.println(" - ¡RC-5!)");
        Serial.println(">>> Captura exitosa. Presione el
botón en Pin 2 para REPRODUCIRlo. <<<");
    }

    IrReceiver.resume(); // Reanudar la recepción
  }
```

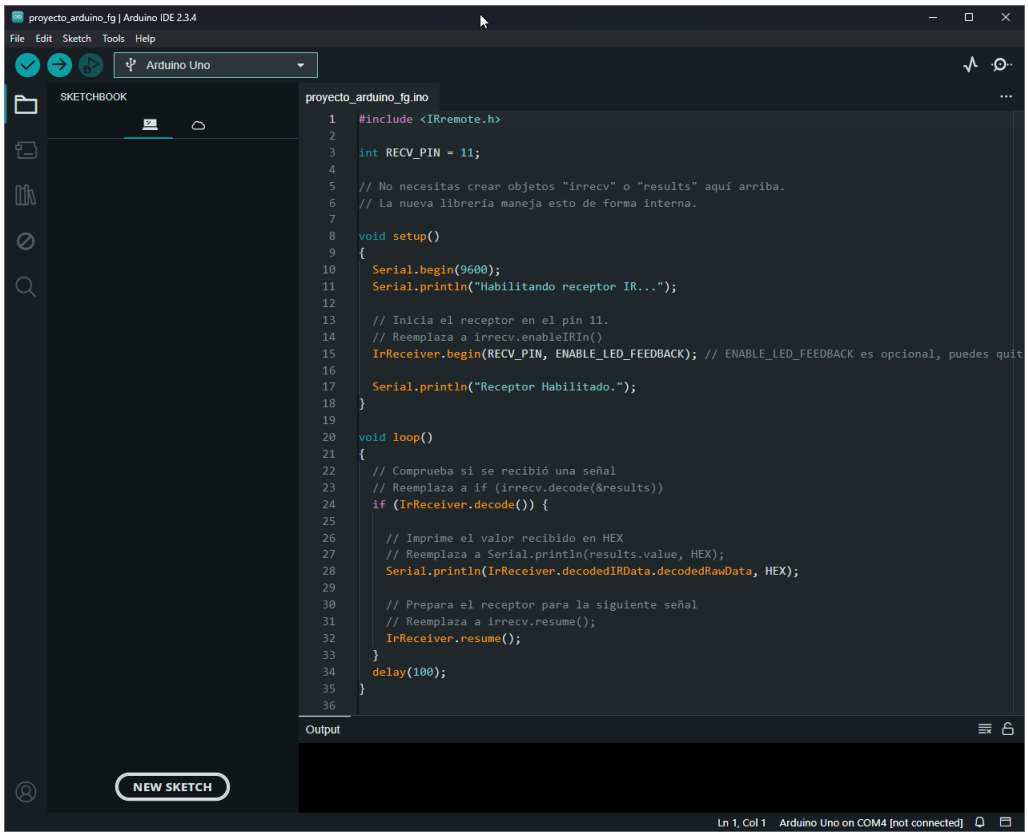
```
  // --- 2. REPRODUCCIÓN (al presionar un botón) ---
  if (digitalRead(BUTTON_PIN) == LOW && !buttonState) {
    buttonState = true; // Botón presionado

    if (lastReceivedCode != 0) {
      Serial.print("📡 ENVIANDO Codigo: ");
      Serial.println(lastReceivedCode, HEX);

      // Usamos el nuevo nombre de la constante
      IrSender.sendRC5(lastReceivedCode, RC5_NUM_BITS);

      Serial.println("Señal Enviada.");
    } else {
      Serial.println("⚠️ No hay codigo capturado valido
para enviar.");
    }
  }

  // Resetear el estado del botón cuando se suelta
  if (digitalRead(BUTTON_PIN) == HIGH && buttonState) {
    buttonState = false;
  }

  delay(100);
}
```

11/noviembre/2025

- Implementación física del circuito.



- Upload de código funcional al Arduino físico mediante Arduino IDE.



Resumen:

El código obtenido el día 8 esta probado y funcionando correctamente (tanto en el ambiente virtual como físico). El sensor recibe la información enviada por el emisor. Dado que esta mas que probado es el que voy a tomar como final. A su vez tengo todos los implementos para demostrar su funcionamiento de forma física.

De todas formas, entiendo que el código obtenido el día 9 esta correcto y debería ser funcional. Lamentablemente no tengo forma de probarlo físicamente (dado que no tengo los implementos necesarios (botón pulsador y resistencias)) y tampoco es posible probarlo virtualmente (en Tinkercad) dado que no se cuenta con un emisor IR (solo existe el LED, el cual solo emite luz y no envía la información guardada del control remoto).

Conclusión:

Se pudo recibir y decodificar las señas de un control remoto en Arduino. Esto se pudo probar tanto virtualmente (en Tinkercad) como físicamente.

Posterior mente se pudo guardar esa información y emitirla a través del emisor IR de Arduino.
Si bien es cierto que esto no pudo verificarse ni virtualmente (por limitantes de Tinkercad) ni físicamente (por no contar con los componentes), tanto el código como la implementación del circuito está avalado por varias IA entre las que destacamos Gemini y ChatGPT.

Lo que no se pudo realizar es el cometido de la tarea inicial, que era guardar los distintos comandos de un control remoto (como pueden ser encender, apagar, frio, calor, subir, bajar, etc.), para luego, programar las instrucciones IR para que se ejecuten a algún momento especifico.

De todas formas, me quedo conforme con lo realizado hasta el momento y lo aprendido en el transcurso del proyecto.