**UNIVERSIDAD DE CONCEPCIÓN**

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA



**549305-1 – Taller de aplicación TIC ll**

Profesor Vincenzo Caro Fuentes

**Mini proyecto 2**

Bastián Baeza Retamal

Pablo Olguin Molina

**Resumen**

En esta actividad, se emplearon dos Arduinos para establecer comunicación mediante señales infrarrojas (IR) utilizando el código Morse. Un Arduino actuó como transmisor, codificando y enviando mensajes, mientras que el otro funcionó como receptor, decodificando y mostrando el mensaje original.

El código Morse es un sistema que representa letras y números a través de señales intermitentes de puntos y rayas. Los puntos son señales cortas de una unidad de tiempo y las rayas son señales largas de tres unidades de tiempo. Los espacios entre elementos de una letra, entre letras, y entre palabras se utilizan para separar los componentes del mensaje, con duraciones específicas para asegurar la correcta interpretación.

**Tabla de Contenidos**

[Descripción en de los implementos utilizados 1](#_Toc167968674)

[1.1 Sensores y dispositivos implementados 1](#_Toc167968675)

[Actividad n°1. Morse IR 1](#_Toc167968676)

[Ítem 1.1 1](#_Toc167968677)

[1.1.a) Enunciado 1](#_Toc167968678)

[1.1 Esquema de circuitos 3](#_Toc167968679)

[1.2 Resultados y comentarios 4](#_Toc167968680)

[1.3 Código de la actividad 4](#_Toc167968681)

## Descripción en de los implementos utilizados

### Sensores y dispositivos implementados

* 1 LED de transmisión IR (KY-005)
* 1 LED de recepción IR (KY-022)
* 2 LED RGB o DUAL (KY-009, KY-011, KY-016 o KY-029)
* 2 buzzer pasivo (KY-006) o activo (KY-012)

# Morse IR

## 

### Enunciado

Para implementar su Morse IR, consideren los siguientes módulos de su kit de 45 Sensores:

* 1 LED de transmisión infrarroja (KY-005).
* 1 LED de recepción infrarroja (KY-022).
* 2 LED RGB o DUAL (KY-009, KY-011, KY-016 o KY-029).
* 2 buzzer pasivo (KY-006) o activo (KY-012).

Con los elementos mencionados, preparen dos circuitos de tal manera que el LED IR de transmisión conectado con el primer Arduino apunte directamente al LED IR de recepción conectado con el segundo Arduino, sin obstáculos presentes entre ambos. Cada LED IR debe ir acompañado de un LED de color que indique cuándo se está transmitiendo (o no) un mensaje con un color diferente para cada caso, evitando que la luz apunte directamente a los LED IR para evitar posibles interferencias. Además, el LED IR de transmisión debe contar con un buzzer que simule el sonido característico de los telégrafos utilizados para enviar códigos Morse, respetando la duración respectiva de los puntos y rayas.

Respecto del programa que debe acompañar a cada Arduino, consideren los siguientes

aspectos prácticos:

Implementación del Transmisor:

* Su código debe contener el alfabeto completo del código Morse (números y letras),

registrando para cada caso la distribución de puntos y rayas mostradas en el Anexo.

* El mensaje a enviarse por código Morse deberá ser ingresado desde el monitor serial encada iteración del loop(), contando con al menos 3 palabras distintas o más, y permitiendo un reenvío de hasta 3 oportunidades antes de solicitar uno nuevo.
* Para el envío de su mensaje, preparen dos funciones que representen los puntos y rayas

del código Morse. Cada función deberá activar el emisor IR y el buzzer escogido durante un tiempo específico, considerando 250 ms como la unidad de tiempo base para el punto y 750 ms para una raya.

* Adicionalmente, implementen las pausas entre cada elemento de una letra (250 ms), entre cada letra de una palabra (750 ms) y entre cada palabra de su mensaje (1750 ms) para asegurar que el receptor pueda interpretar correctamente la secuencia de puntos y rayas.
* Para facilitar la identificación de todos los componentes de su mensaje por parte del receptor (inicio, puntos, rayas, pausas y término), utilicen un comando distinto del protocolo NEC de transmisión IR para cada caso. Por ejemplo:

- Inicio: 0x01

- Puntos: 0x02

- Rayas: 0x03

- Pausa entre elementos: 0x04

- Pausa entre letras: 0x05

- Pausa entre palabras: 0x06

- Término: 0x07

Implementación del Receptor:

* ✓ Su código debe contener el alfabeto completo del código Morse (números y letras),

registrando para cada caso la distribución de puntos y rayas mostradas en el Anexo.

* ✓ Al recibir una señal desde el transmisor, su código deberá identificar los diferentes

comandos asignados para cada parte del mensaje, apoyándose del tiempo que transcurre

entre las señales sucesivas para determinar si representan un inicio, punto, raya, pausas o

un término del mensaje.

* ✓ El mensaje recibido debe traducirse (letra por letra) mediante un proceso de comparación con el diccionario de código Morse previamente definido, el cual deberá mostrarse en la consola serial de Arduino.
* En caso de recibirse la palabra “SOS”, un buzzer deberá emitir un sonido de alerta una

vez se haya decodificado todo el mensaje.

* En caso de recibirse la palabra “TIC”, un buzzer deberá emitir un sonido personalizado una vez se haya decodificado todo el mensaje.

Para conceptualizar la transmisión de un mensaje sencillo con sus pausas respectivas, en el Anexo se presenta un ejemplo con la palabra “SOS”.

Si bien existen múltiples formas de decodificar el código Morse por parte del receptor, el mensaje transmitido debe cumplir con todos los tiempos y elementos solicitados.

### Esquema de circuitos

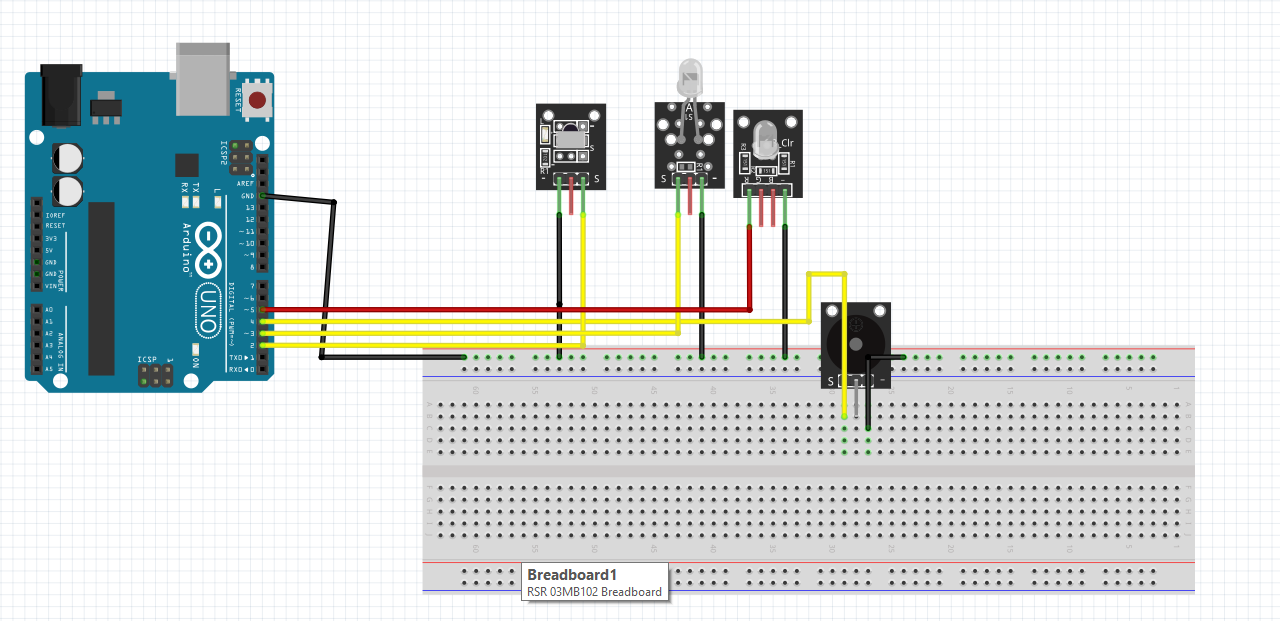


Fig 1.1 Esquema en fritzing

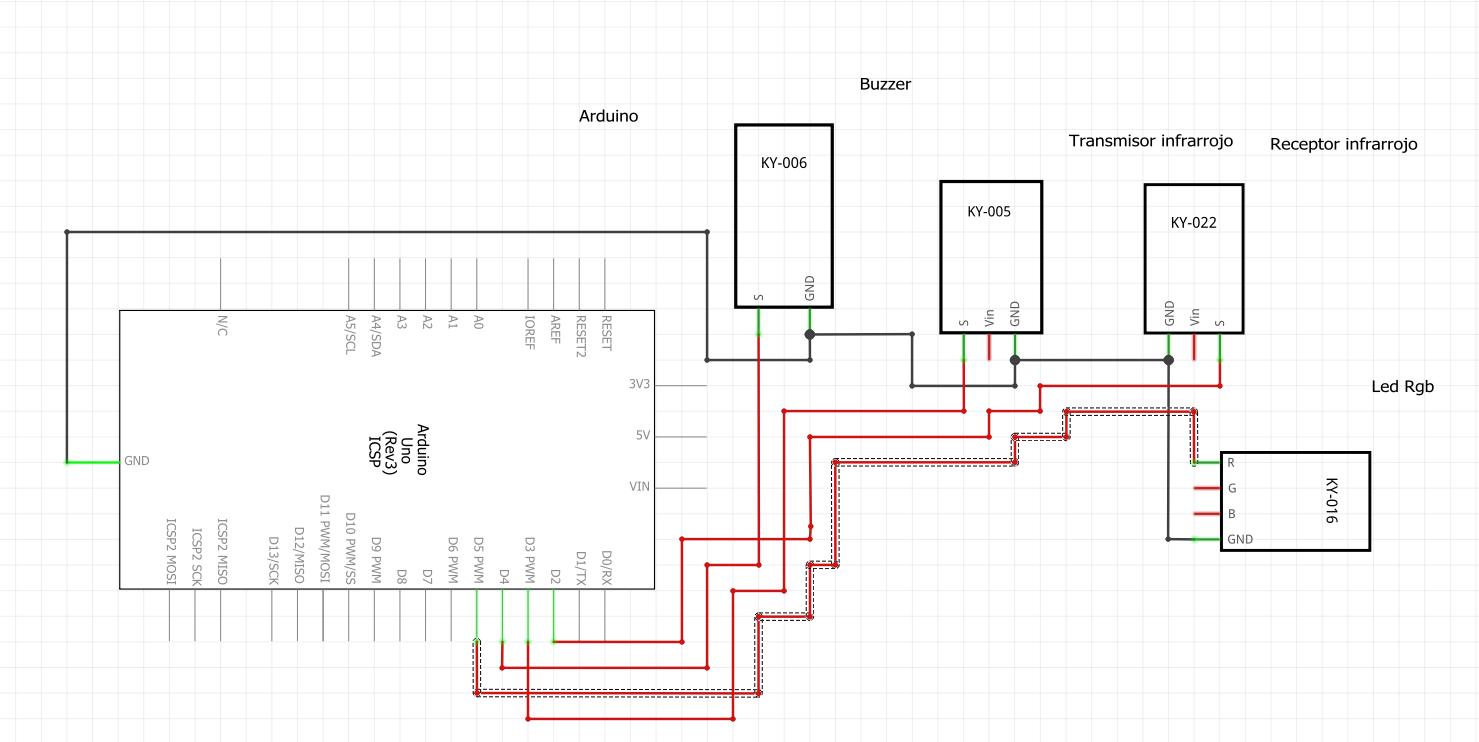


Fig 1.2 Esquema de circuito en fritzing

### Resultados y comentarios

Al implementar nuestro código y hacer las conexiones pertinentes, hemos logrado hacer que el emisor envié nuestro mensaje codificado en morse, luego el receptor infrarrojo decodifica nuestro mensaje en morse y nos lo imprime en el puerto serial del Arduino mostrándonos los puntos y rayas que fueron recibidos y la traducidos al letras del abecedario, a este proceso lo acompaña un led el cual se ilumina según sea punto o raya también el buzzer se enciende cuando se detecta la palabra “SOS” o “TIC” reproduciendo el sonido distintivo para ambas palabras.

### 1.3 Código de la actividad

En este apartado se mostrará el código de ambas arduinos

#### Arduino receptor

#include <IRremote.h>

int pin\_receptor = 2;

int pin\_buzzer = 4;

unsigned long lastButtonPress = 0;

const unsigned long debounceDelay = 50; // 50 milisegundos de retraso

const int tiempo = 150; // Tiempo base en milisegundos para un punto

String MENSAJE\_\_\_1 = "";

void setup() {

    Serial.begin(9600);

    pinMode(pin\_buzzer, OUTPUT);

    IrReceiver.begin(pin\_receptor, DISABLE\_LED\_FEEDBACK); // Iniciar receptor IR

    Serial.println("Receptor IR listo para recibir señales...");

}

void loop() {

    if (IrReceiver.decode()) {

        if (millis() - lastButtonPress > debounceDelay) {

            handleReceivedSignal(IrReceiver.decodedIRData.command);

            lastButtonPress = millis();

            delay(150);

        }

        IrReceiver.resume(); // Prepararse para la siguiente señal

    }

}

void handleReceivedSignal(uint8\_t command) {

    int r = 0;

    if (command != 0x01 && command != 0x05) {

        Serial.print(command, HEX);

        Serial.print(" ");

    }

    if (command == 0x02) {

        MENSAJE\_\_\_1 += '.';

    }

    else if (command == 0x03) {

        MENSAJE\_\_\_1 += '-';

    }

    else if (command == 0x06) {

        for (r = 0; r < 5; r++) {

            MENSAJE\_\_\_1 += '!';

        }

    }

    else if (command == 0x04) {

        MENSAJE\_\_\_1 += '+';

    }

    else if (command == 0x07) {

        decodeMorsemensaje();

    }

}

// ... (el resto de las funciones: decodeMorsemensaje, traducirMorse) ...

void decodeMorsemensaje() {

  String mensajeDecodificado = "";

  String letraMorse = "";

  Serial.println("\nFinal Morse:");

  for (int i = 0; i < MENSAJE\_\_\_1.length(); i++) {

    Serial.print(MENSAJE\_\_\_1.charAt(i));

  }

  Serial.println();

  for (int i = 0; i < MENSAJE\_\_\_1.length(); i++) {

      if ((i% 5) != 0 && i != 0) {

        letraMorse += MENSAJE\_\_\_1.charAt(i); // Agregar punto o guion a la letra Morse actual

      }

      else {

        mensajeDecodificado += traducirMorse(letraMorse); // Traducir letra Morse a carácter

        letraMorse = ""; // Resetear la letra Morse para la siguiente

        letraMorse += MENSAJE\_\_\_1.charAt(i);

      }

  }

  // Traduce la última letra Morse si no hay un espacio al final

  if (letraMorse != "") {

    mensajeDecodificado += traducirMorse(letraMorse);

  }

  mensajeDecodificado = mensajeDecodificado.substring(1);

  if (mensajeDecodificado == "SOS") {

    for (int i = 0; i < 7; i++) {

      tone(pin\_buzzer, 1000, 200); // Tono agudo y corto

      delay(250);

    }

  }

  else if (mensajeDecodificado == "TIC") {

    // Sonido de victoria (ejemplo)

    tone(pin\_buzzer, 440, 500); // Tono La (440 Hz) durante 500 ms

    delay(500);

    tone(pin\_buzzer, 880, 500); // Tono La una octava más alta

    delay(500);

  }

  Serial.println("mensaje recibido: " + mensajeDecodificado);

  MENSAJE\_\_\_1 = ""; // Limpiar el mensaje recibido

  IrReceiver.resume();

}

char traducirMorse(String letraMorse) {

  // Letras

    if (letraMorse == ".-+++") return 'A';

    if (letraMorse == "-...+") return 'B';

    if (letraMorse == "-.-.+") return 'C';

    if (letraMorse == "-..++") return 'D';

    if (letraMorse == ".++++") return 'E';

    if (letraMorse == "..-.+") return 'F';  // Corregido (era "..-." antes)

    if (letraMorse == "--.++") return 'G';

    if (letraMorse == "....+") return 'H';

    if (letraMorse == "..+++") return 'I';  // Corregido (era "..+++" antes)

    if (letraMorse == ".---+") return 'J';

    if (letraMorse == "-.-++") return 'K';

    if (letraMorse == ".-..+") return 'L';

    if (letraMorse == "--+++") return 'M';

    if (letraMorse == "-.+++") return 'N';

    if (letraMorse == "---++") return 'O';

    if (letraMorse == ".--.+") return 'P';

    if (letraMorse == "--.-+") return 'Q';

    if (letraMorse == ".-.++") return 'R'; // Sin cambios

    if (letraMorse == "...++") return 'S';

    if (letraMorse == "-++++") return 'T';

    if (letraMorse == "..-++") return 'U';

    if (letraMorse == "...-+") return 'V';

    if (letraMorse == ".--++") return 'W';

    if (letraMorse == "-..-+") return 'X';

    if (letraMorse == "-.--+") return 'Y';

    if (letraMorse == "--..+") return 'Z';

    if (letraMorse == "!!!!!") return ' ';

  // Números

  if (letraMorse == ".----") return '1';

  if (letraMorse == "..---") return '2';

  if (letraMorse == "...--") return '3';

  if (letraMorse == "....-") return '4';

  if (letraMorse == ".....") return '5';

  if (letraMorse == "-....") return '6';

  if (letraMorse == "--...") return '7';

  if (letraMorse == "---..") return '8';

  if (letraMorse == "----.") return '9';

  if (letraMorse == "-----") return '0';

  ; // Devuelve un carácter de interrogación si no se encuentra la secuencia Morse

}

#### Arduino emisor

#include <IRremote.h>

int pin\_transmisor = 3;

int pin\_receptor = 2;

int pin\_buzzer = 4;

int LED\_PIN\_TRANSMIT = 5;  // Ejemplo para un LED de color

const int tiempo = 150; // Tiempo base en milisegundos para un punto

String MENSAJE\_\_\_1 = "";

void setup() {

  Serial.begin(9600);

  pinMode(pin\_transmisor, OUTPUT);

  pinMode(pin\_buzzer, OUTPUT);

  pinMode(LED\_PIN\_TRANSMIT, OUTPUT);

  IrReceiver.begin(pin\_receptor, DISABLE\_LED\_FEEDBACK);

  IrSender.begin(pin\_transmisor, DISABLE\_LED\_FEEDBACK, 0);

  int numRepeticiones = 1;

}

void loop() {

  if (Serial.available() > 0) {

    Serial.println("Por favor, ingrese el mensaje!!!");

    String mensaje = Serial.readString(); // mensaje = mensaje que ingresamos en el monitor serial

    Serial.println("¿Cuántas veces quieres enviar el mensaje?");

      while (Serial.available() == 0) {

            // Esperar a que el usuario ingrese un número

      }

      int numRepeticiones = Serial.parseInt();

    for (int i = 0; i < mensaje.length(); i++) {

      char c = mensaje.charAt(i);

      Serial.print(c);

    }

    Serial.println();

    Serial.println("Init Morse:");

    F\_enviarmorse(mensaje, numRepeticiones);

  }

}

void F\_enviarmorse(String mensaje, int numRepeticiones)  {

  digitalWrite(LED\_PIN\_TRANSMIT, HIGH); // led q se prende cuando se envia mensaje

  IrSender.sendNEC(0x1234, 0x01, 1);

  delay(tiempo);

  for (int j = 0; j < numRepeticiones ; j++) {

    for (int i = 0; i < mensaje.length(); i++) {

      char c = mensaje.charAt(i);

      if (c == ' ') {

        IrSender.sendNEC(0x1234, 0x06, 1);

        delay(tiempo \* 7); // Pausa palabras

      } else {

        F\_morsificador(c);

        IrSender.sendNEC(0x1234, 0x05, 1);

        delay(tiempo \* 3); // Pausa letra

      }

    }

  }

  IrSender.sendNEC(0x1234, 0x07, 1);

  delay(tiempo);

  tone(pin\_buzzer, 500, tiempo \* 2);

  digitalWrite(LED\_PIN\_TRANSMIT, LOW); // Fin de la transmisión

}

void F\_morsificador(char c) {

  String codigo\_morse = getcodigo\_morse(c);

    Serial.println();

  for (int i = 0; i < codigo\_morse.length(); i++) {

    if (codigo\_morse.charAt(i) == '.') {

      sendDot();

    }

    else if (codigo\_morse.charAt(i) == '-') {

      sendDash();

    }

    else if (codigo\_morse.charAt(i) == '+') {

      elemento\_fantasma();

    }

  }

    for (int i = 0; i < codigo\_morse.length(); i++) {

      char c = codigo\_morse.charAt(i);

      Serial.print(c);

    }

}

void sendDot() {

  IrSender.sendNEC(0x1234, 0x02, 1);

  delay(50);

    for (int i = 0; i < tiempo; i++) {

        analogWrite(pin\_buzzer, 128);  // 50% duty cycle (volumen medio)

        delayMicroseconds(500);       // Aproximadamente 1 kHz

        analogWrite(pin\_buzzer, 0);

        delayMicroseconds(500);

    }

  delay(tiempo);

}

void sendDash() {

  IrSender.sendNEC(0x1234, 0x03, 1);

  delay(50);

    for (int i = 0; i < tiempo; i++) {

        analogWrite(pin\_buzzer, 128);  // 50% duty cycle (volumen medio)

        delayMicroseconds(250);       // Aproximadamente 1 kHz

        analogWrite(pin\_buzzer, 0);

        delayMicroseconds(250);

    }

  delay(tiempo \* 3);

}

void elemento\_fantasma() {

  IrSender.sendNEC(0x1234, 0x04, 1);

  delay(tiempo);

}

String getcodigo\_morse(char c) {

  // Letras

  if (c == 'A' || c == 'a') return ".-+++";

  if (c == 'B' || c == 'b') return "-...+";

  if (c == 'C' || c == 'c') return "-.-.+";

  if (c == 'D' || c == 'd') return "-..++";

  if (c == 'E' || c == 'e') return ".++++";

  if (c == 'F' || c == 'f') return "..-.+";

  if (c == 'G' || c == 'g') return "--.++";

  if (c == 'H' || c == 'h') return "....+";

  if (c == 'I' || c == 'i') return "..+++";

  if (c == 'J' || c == 'j') return ".---+";

  if (c == 'K' || c == 'k') return "-.-++";

  if (c == 'L' || c == 'l') return ".-..+";

  if (c == 'M' || c == 'm') return "--+++";

  if (c == 'N' || c == 'n') return "-.+++";

  if (c == 'O' || c == 'o') return "---++";

  if (c == 'P' || c == 'p') return ".--.+";

  if (c == 'Q' || c == 'q') return "--.-+";

  if (c == 'R' || c == 'r') return ".-.++";

  if (c == 'S' || c == 's') return "...++";

  if (c == 'T' || c == 't') return "-++++";

  if (c == 'U' || c == 'u') return "..-++";

  if (c == 'V' || c == 'v') return "...-+";

  if (c == 'W' || c == 'w') return ".--++";

  if (c == 'X' || c == 'x') return "-..-+";

  if (c == 'Y' || c == 'y') return "-.--+";

  if (c == 'Z' || c == 'z') return "--..+";

  // Números

  if (c == '1') return ".----";

  if (c == '2') return "..---";

  if (c == '3') return "...--";

  if (c == '4') return "....-";

  if (c == '5') return ".....";

  if (c == '6') return "-....";

  if (c == '7') return "--...";

  if (c == '8') return "---..";

  if (c == '9') return "----.";

  if (c == '0') return "-----";

  return ""; // Devuelve una cadena vacía si no se encuentra el carácter

}