Examen práctico
Unidad académica Multidisciplinaria Mante

Asignatura: Programación De Microcontroladores

Nombre del trabajo: Examen Parcial 3

Nombre del docente: Ing. Daniel López Piña

Nombre del alumnos:
Pablo Iván Amaya Montalvo
José Ricardo Vargas Guillén
Medina Rodríguez Oscar
Adrián Carbajal Mejía
José Aldo Orta Solís
Vanessa Rodríguez García
Rolando Martínez Delgado

Especialidad: Ing. Sistemas Computacionales

Grado y Grupo: 8 “EJ”

Fecha: 13 de mayo del 2025

Actividadad a realizar:

Utilizando el salto indexado construir un seguidor de línea que complete la pista

Selección de componentes

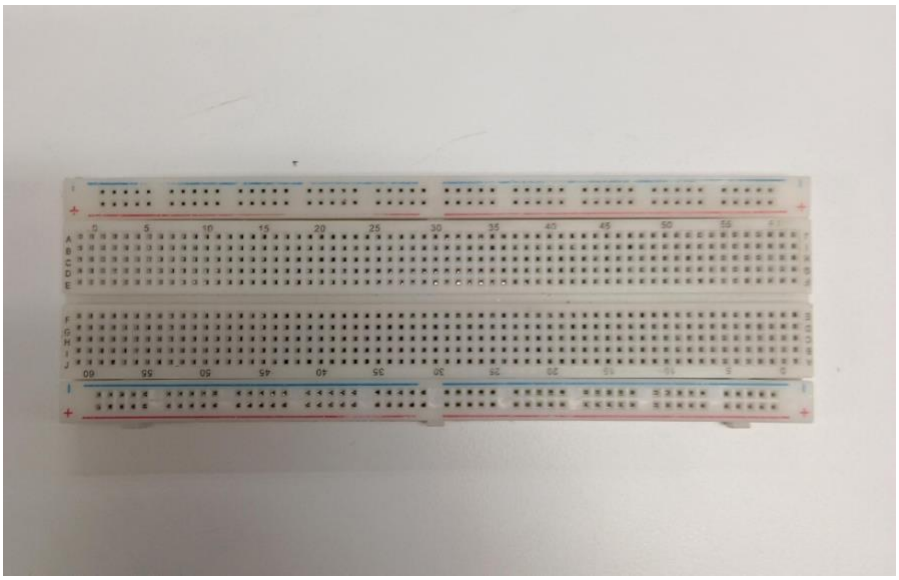
Como primer paso para la construcción del seguidor de línea, realizamos la búsqueda y recolección de los componentes necesarios. Se eligieron elementos básicos de electrónica y robótica, considerando su disponibilidad, compatibilidad y facilidad de integración con el microcontrolador PIC16F84A.

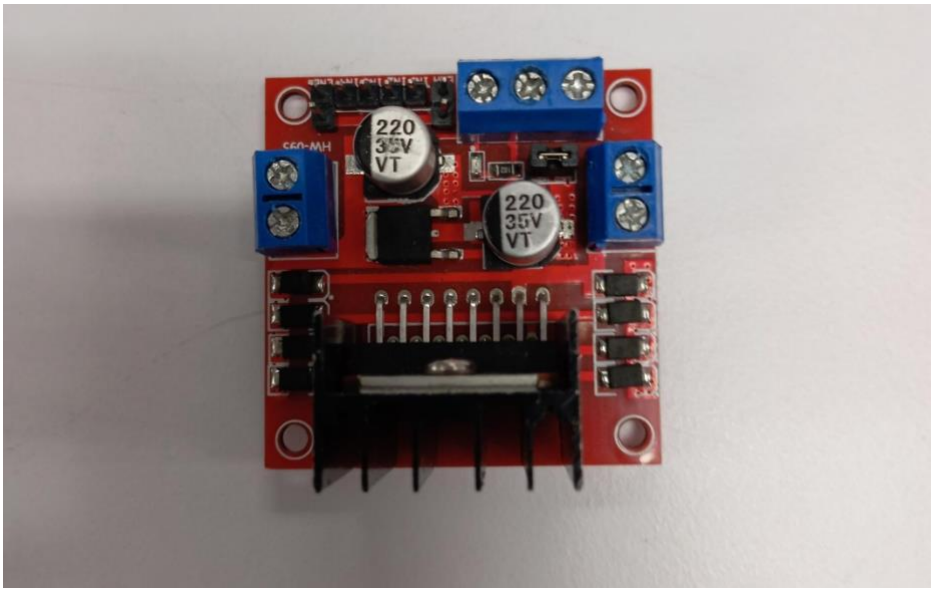
El sistema fue diseñado para detectar la línea mediante sensores infrarrojos y actuar en consecuencia controlando dos motores a través de un puente H. Para facilitar el armado del circuito y las pruebas, se utilizó una protoboard, además de elementos auxiliares como jumpers, un oscilador externo, condensadores, regulador de voltaje y una fuente de alimentación adecuada. Todo el montaje se integró en un chasis con ruedas, portapilas y el conjunto necesario para permitir el desplazamiento del robot en la pista.

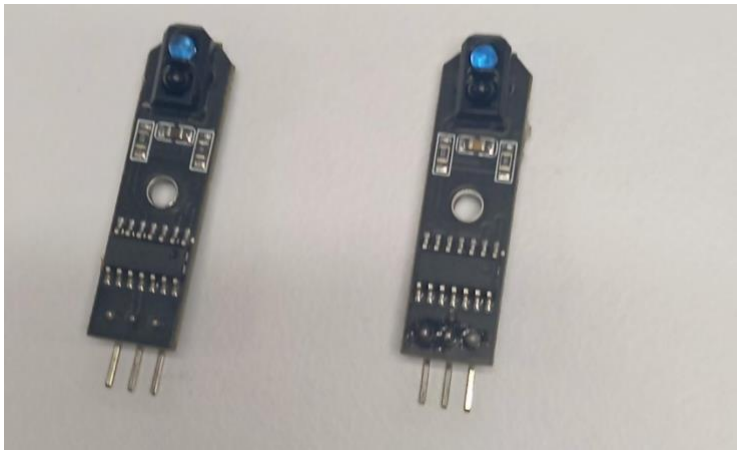
Material necesarios:

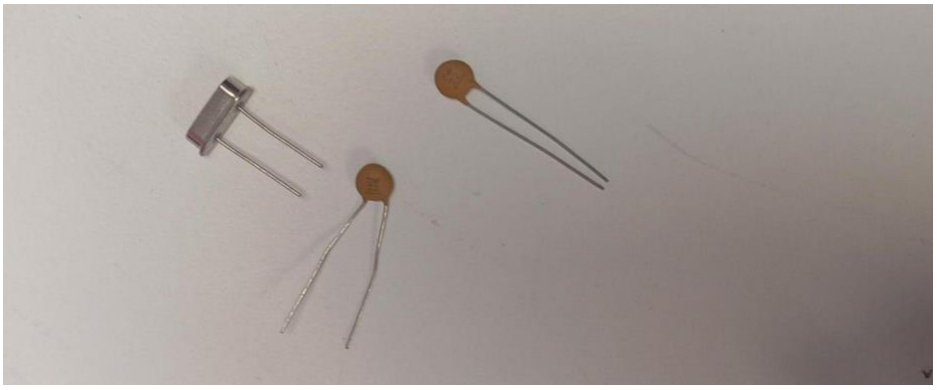
- Puente H
- Protoboard
- Sensores infrarrojos
- Jumpers
- Oscilador 4000
- Condensador cerámico 22mF
- Fuente de alimentación
- 2 Motores
- PIC16F84A
- Chasis
- Pilas
- Ruedas
- Portapilas y pilas
- Regulador

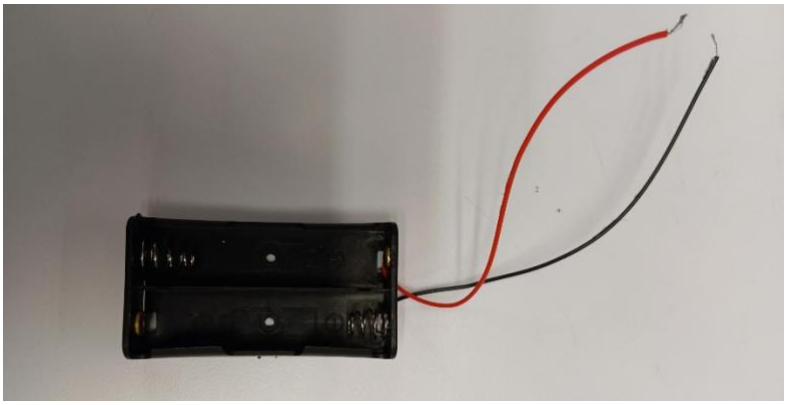
Evidencia:

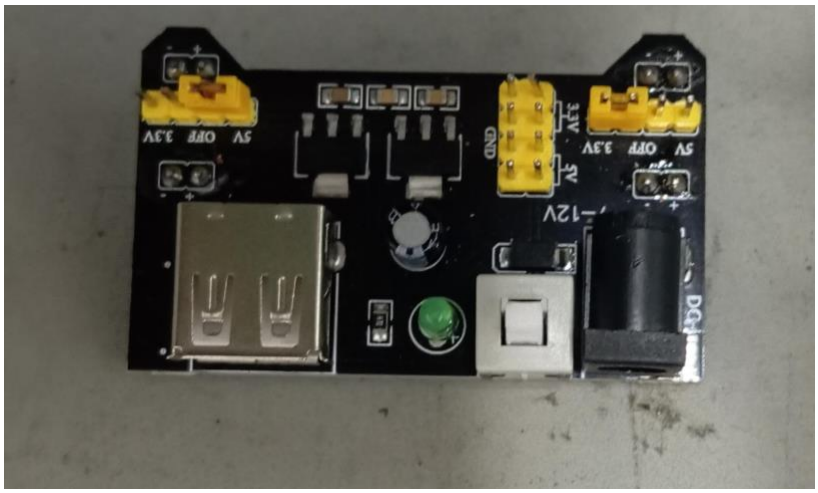


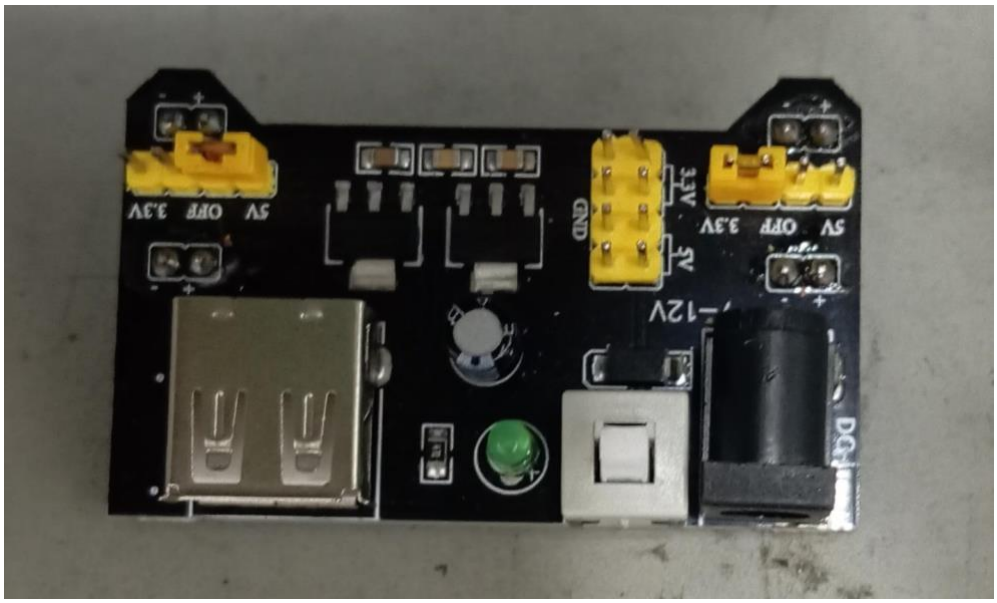

220
35V
VT
220
35V
VT

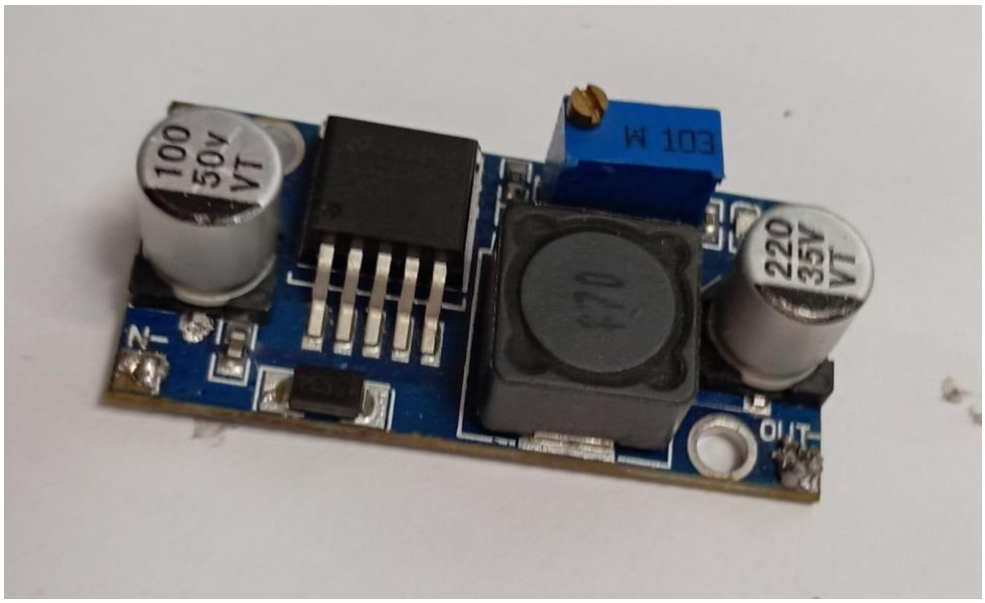


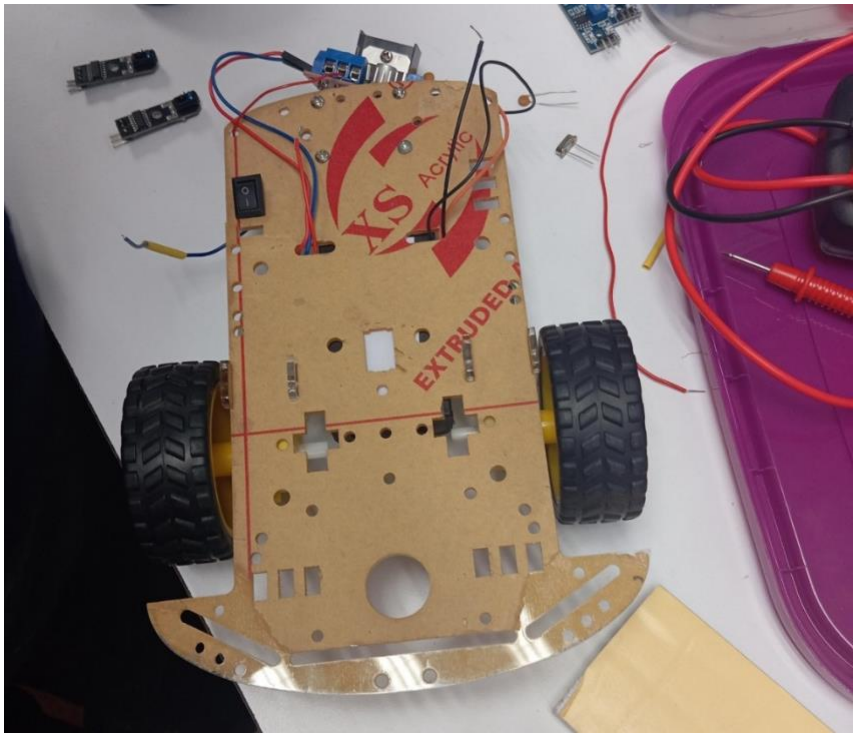





5V OFF 3.3V
GND
3.3V
5V
5V OFF 3.3V
-12V
DC-IN


5V OFF 3.3V
GND
3.3V
5V
5V OFF 3.3V
DC


100
50v
VT
W 103
220
35V
VT
IN-
OUT-


XS
Acrylic
EXTRUDED

Conexiones:

Pines del PIC16F84A
Entradas (Sensores IR):

Sensor	Pin PIC16F84A	Nombre
Sensor Izquierdo	Pin 17	RA0
Sensor Derecho	Pin 18	RA1

Salidas (Motores vía L298N):

Función (L298N)	Pin PIC16F84A	Nombre
IN1 (Motor izquierdo)	Pin 6	RB0
IN2 (Motor izquierdo)	Pin 7	RB1
IN3 (Motor derecho)	Pin 8	RB2
IN4 (Motor derecho)	Pin 9	RB3

Pines del L298N

Función	Conectar a	Nota
IN1	RB0 (pin 6 PIC)	Motor izquierdo
IN2	RB1 (pin 7 PIC)	Motor izquierdo
IN3	RB2 (pin 8 PIC)	Motor derecho
IN4	RB3 (pin 9 PIC)	Motor derecho
ENA (Enable A)	+5V	O usar otro pin si quieres controlarlo
ENB (Enable B)	+5V	
VSS (lógica)	+5V	Alimentación de control
VS (motores)	6–12 V	Alimentación de los motores
GND	GND común	Compartido con PIC y sensores

Sensores IR

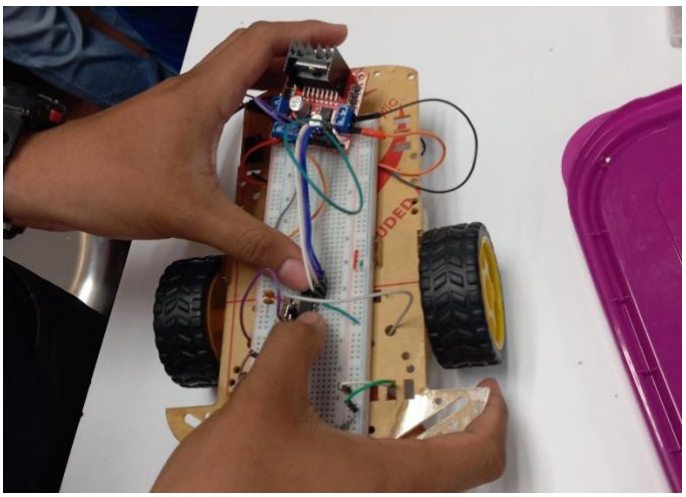
Sensor	Pin del sensor	Conectar a
Sensor Izquierdo	OUT	RA0 (pin 17 del PIC)
Sensor Derecho	OUT	RA1 (pin 18 del PIC)

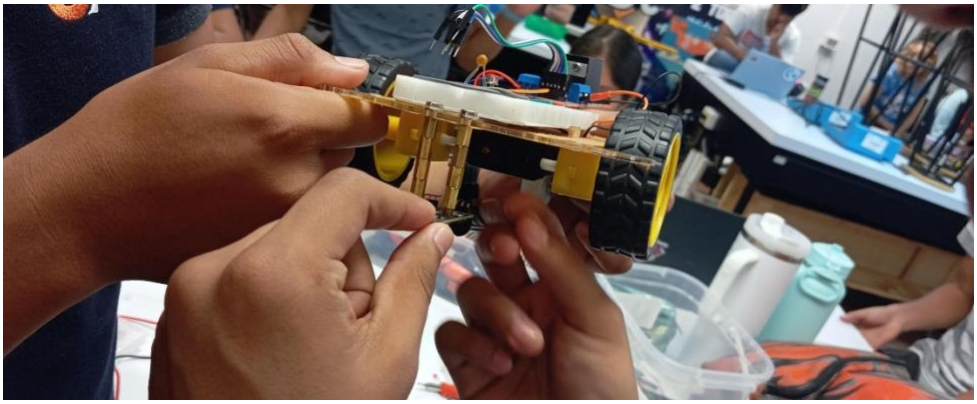
VCC (ambos)	+5 V
GND (ambos)	GND común

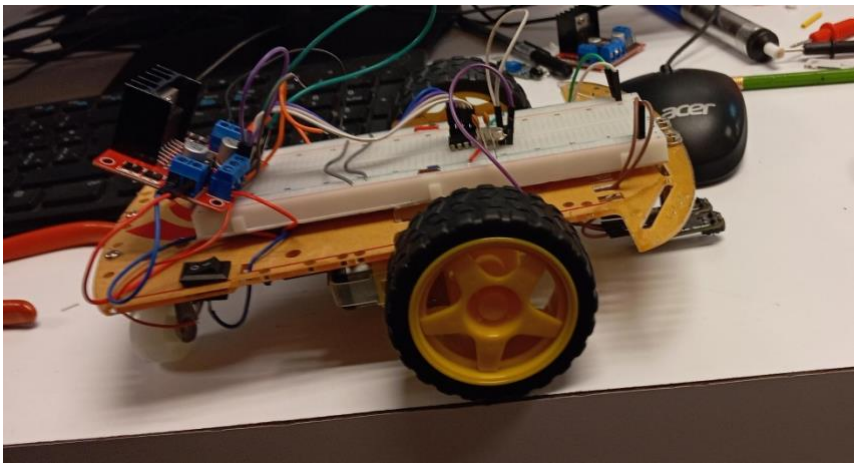
Otros pines del PIC16F84A

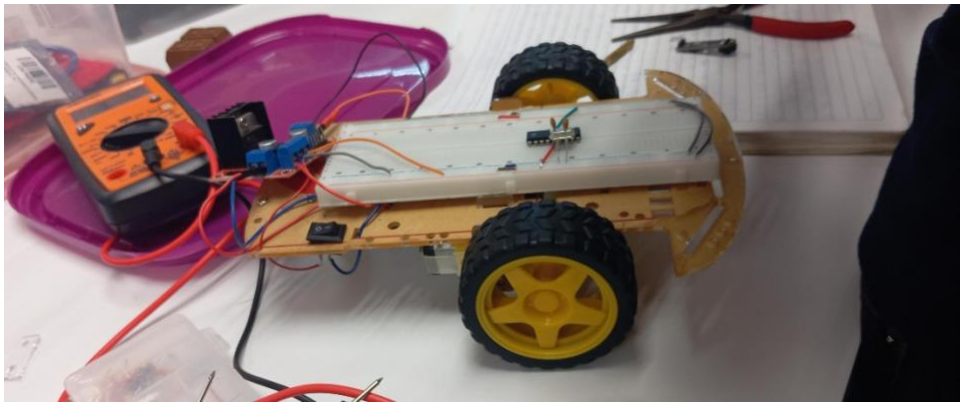
Nombre	Pin	Función
VDD	14	+5 V
VSS	5	GND
OSC1	15	Cristal externo (ej. 4 MHz)
OSC2	16	Cristal externo (ej. 4 MHz)
MCLR	4	Conectar a +5 V con resistencia (10k)

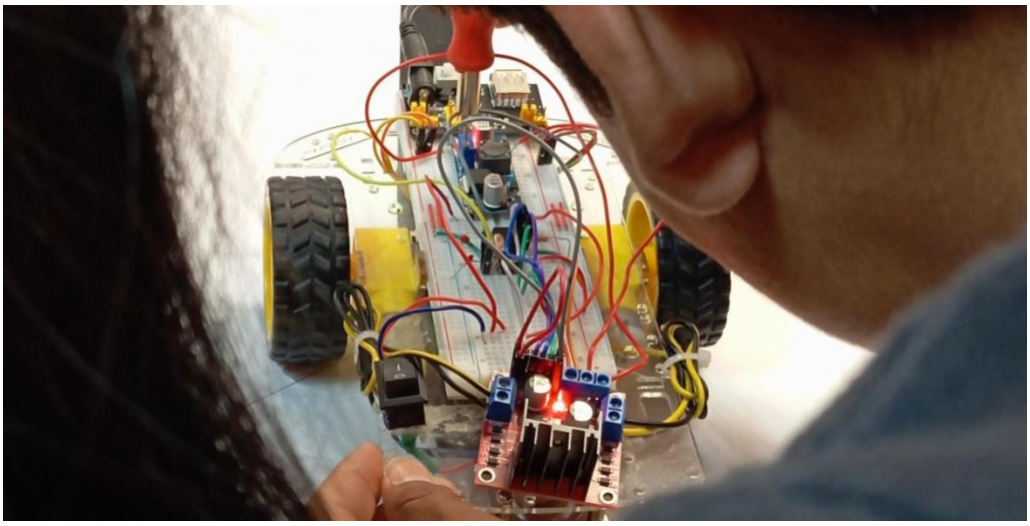
Evidencias :











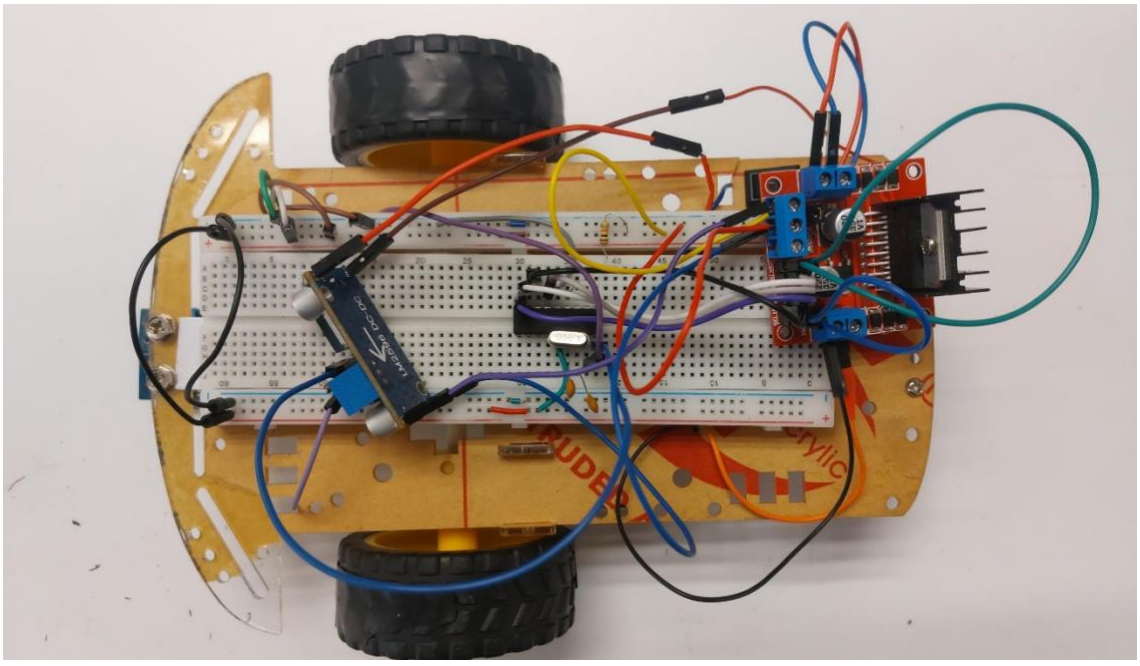
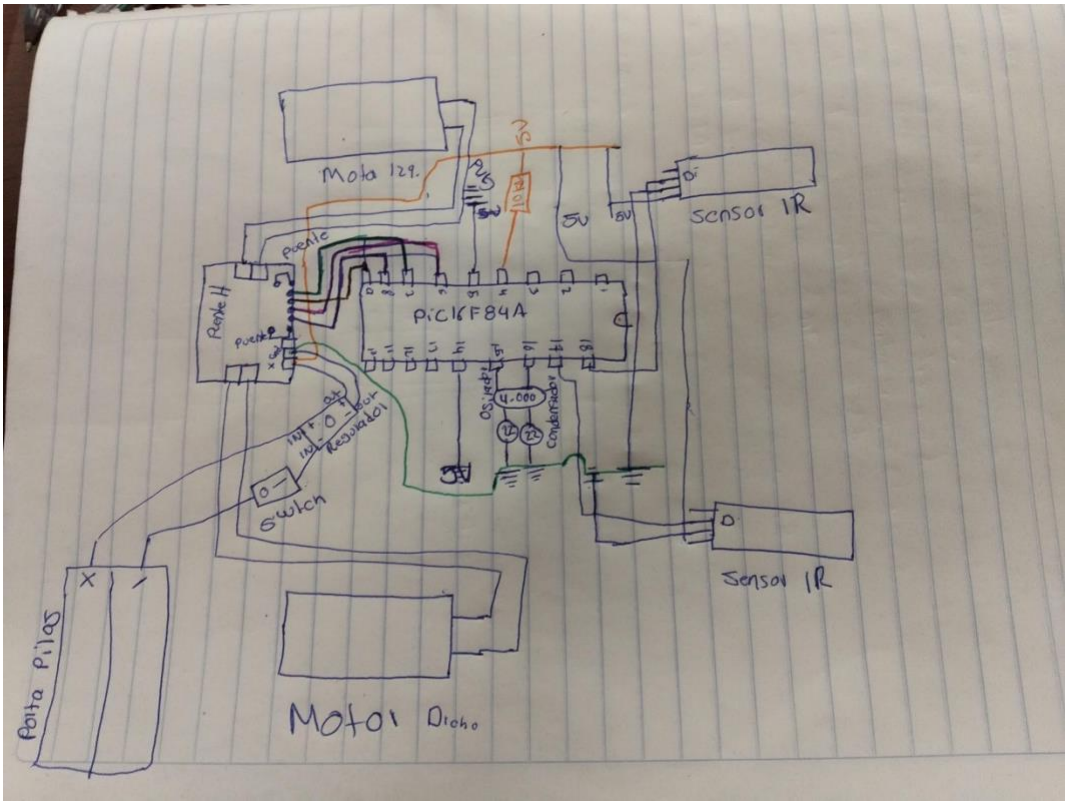


Diagrama:

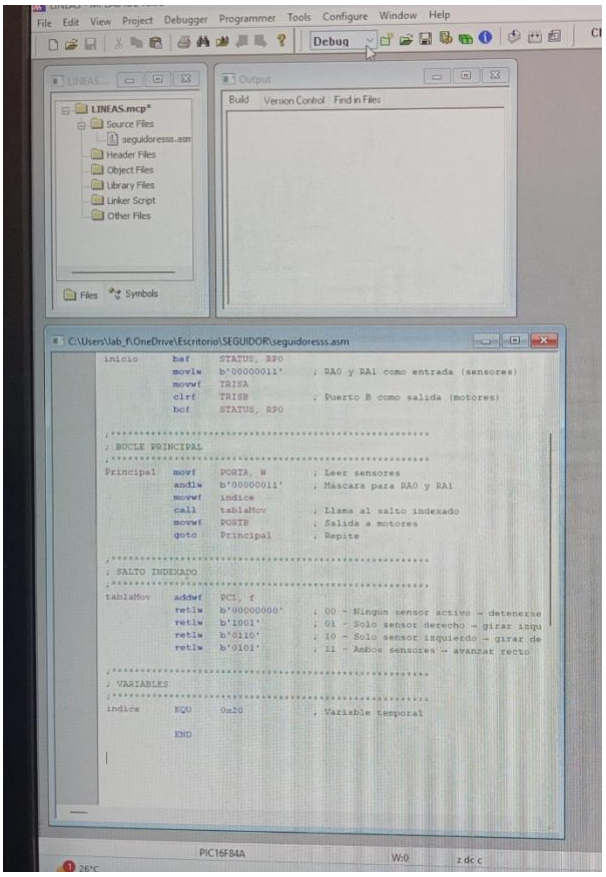

Mota 129.
5V
Sensor IR
Puente
Puente H
PIC16F84A
4.000
Condensador
Switch
Sensor IR
Motor Dicho
Porta Pilas

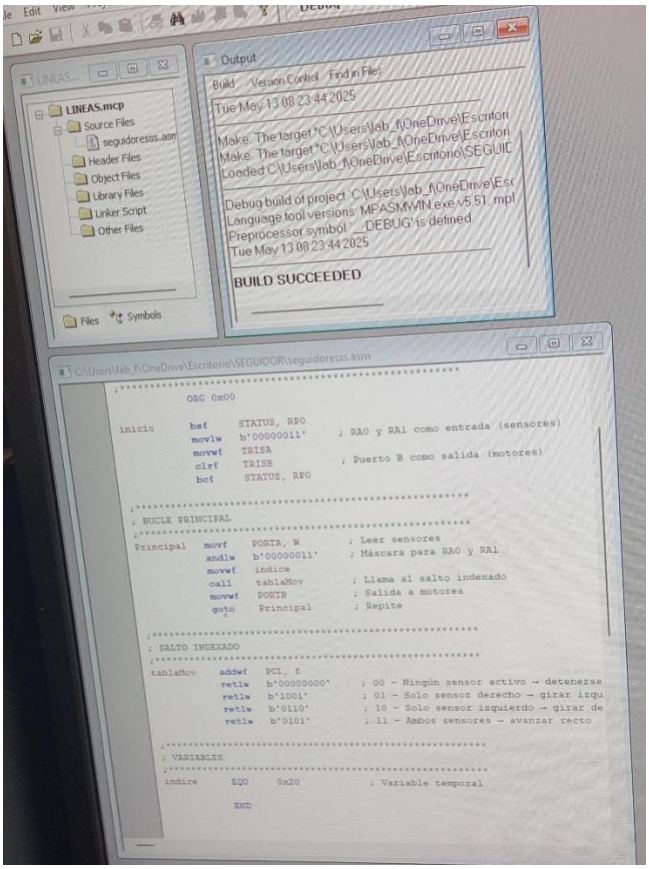
Código:

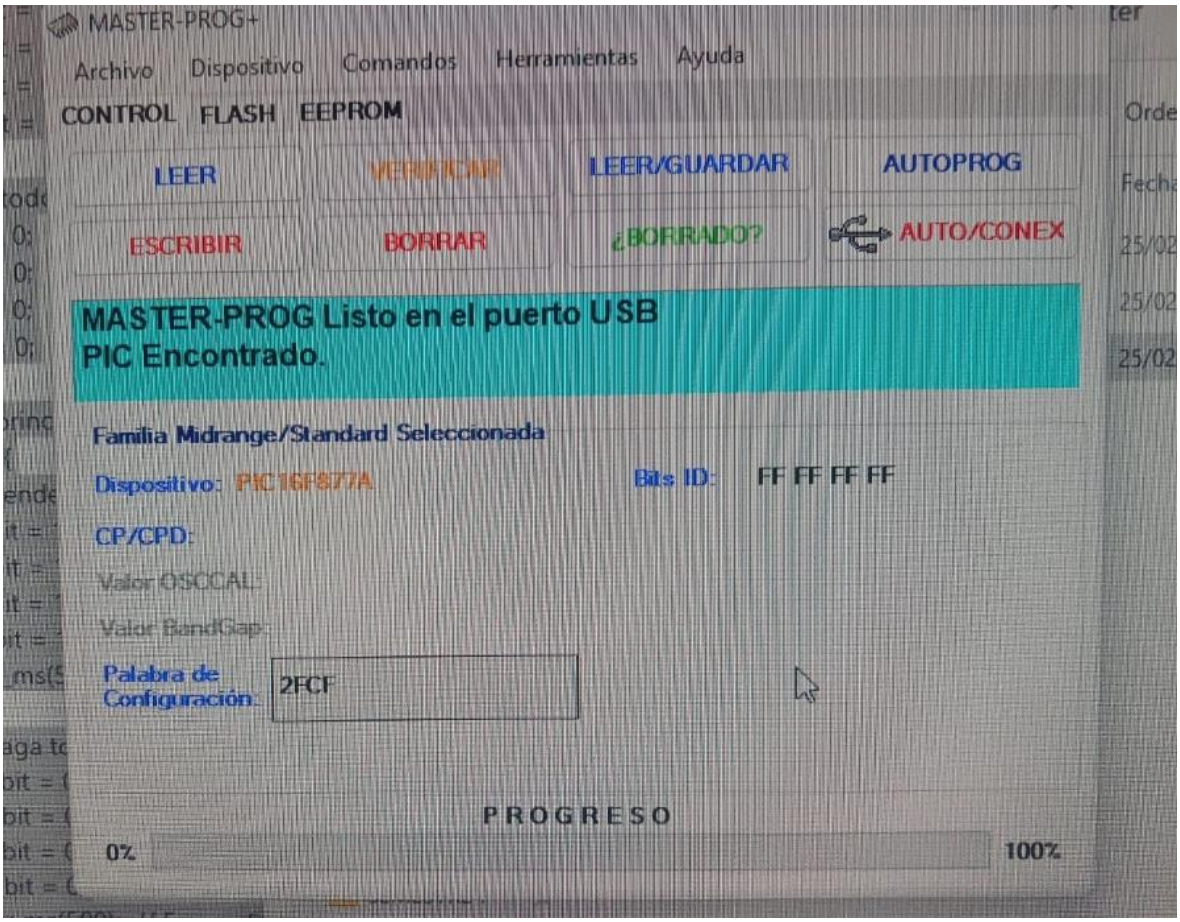
```
; CONFIGURACION ******************************************************
      LIST P=16F84A
      INCLUDE <P16F84A.INC>
      __CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC

; INICIO DEL PROGRAMA ************************************************
      ORG 0

Inicio
      bsf STATUS, RP0      ;
      clrf TRISB           ;
      movlw b'00000011'    ;
      movwf TRISA          ;
      bcf STATUS, RP0      ;

Principal
      movf PORTA, W        ;
      andlw b'00000011'    ;
      addwf PCL, F         ;

; TABLA DE SALTOS ***************************************************

      goto Config0         ;
      goto Config1         ;
      goto Config2         ;
      goto Config3         ;

; CONFIGURACIONES DE MOTORES ****************************************

Config0                    ;
```

```
        movlw b'00000001'    ;

Config1                 ;
        movlw b'00000001'    ;

Config2                 ;
        movlw b'00000010'    ;

Config3                 ;
        movlw b'00000011'    ;

Ejecutar
        movwf PORTB         ;
        goto Principal      ;

        END
```

Evidencia:


File Edit View Project Debugger Programmer Tools Configure Window Help
Debug
LINEAS.mcp*
Source Files
seguidoresss.asm
Header Files
Object Files
Library Files
Linker Script
Other Files
Output
Build Version Control Find in Files
C:\Users\lab_f\OneDrive\Escritorio\SEGUIDOR\seguidoresss.asm
inicio bsf STATUS, RP0
movlw b'00000011' ; RA0 y RA1 como entrada (sensores)
movwf TRISA
clrf TRISB ; Puerto B como salida (motores)
bcf STATUS, RP0
; BUCLE PRINCIPAL
Principal movf PORTA, W ; Leer sensores
andlw b'00000011' ; Máscara para RA0 y RA1
movwf indice
call tablaMov ; Llama al salto indexado
movwf PORTB ; Salida a motores
goto Principal ; Repite
; SALTO INDEXADO
tablaMov addwf PCL, f
retlw b'00000000' ; 00 - Ningun sensor activo → detenerse
retlw b'1001' ; 01 - Solo sensor derecho → girar izqu
retlw b'0110' ; 10 - Solo sensor izquierdo → girar de
retlw b'0101' ; 11 - Ambos sensores → avanzar recto
; VARIABLES
indice EQU 0x20 ; Variable temporal
END
PIC16F84A
W:0


Output
Build Version Control Find in Files
Tue May 13 08:23:44 2025
Make: The target "C:\Users\lab_f\OneDrive\Escritori
Make: The target "C:\Users\lab_f\OneDrive\Escritori
Loaded C:\Users\lab_f\OneDrive\Escritorio\SEGUIC
Debug build of project `C:\Users\lab_f\OneDrive\Esc
Language tool versions: MPASMWIN.exe v5.51, mpl
Preprocessor symbol `__DEBUG' is defined.
Tue May 13 08:23:44 2025
BUILD SUCCEEDED
LINEAS.mcp
Source Files
seguidoresss.asm
Header Files
Object Files
Library Files
Linker Script
Other Files
C:\Users\lab_f\OneDrive\Escritorio\SEGUIDOR\seguidoresss.asm
ORG 0x00
inicio bsf STATUS, RP0
movlw b'00000011' ; RA0 y RA1 como entrada (sensores)
movwf TRISA
clrf TRISB ; Puerto B como salida (motores)
bcf STATUS, RP0
; BUCLE PRINCIPAL
Principal movf PORTA, W ; Leer sensores
andlw b'00000011' ; Máscara para RA0 y RA1
movwf indice
call tablaMov ; Llama al salto indexado
movwf PORTB ; Salida a motores
goto Principal ; Repite
; SALTO INDEXADO
tablaMov addwf PCL, f
retlw b'00000000' ; 00 - Ningún sensor activo → detenerse
retlw b'1001' ; 01 - Solo sensor derecho → girar izqu
retlw b'0110' ; 10 - Solo sensor izquierdo → girar de
retlw b'0101' ; 11 - Ambos sensores → avanzar recto
; VARIABLES
indice EQU 0x20 ; Variable temporal
END


MASTER-PROG+
Archivo
Dispositivo
Comandos
Herramientas
Ayuda
CONTROL FLASH EEPROM
LEER
VERIFICAR
LEER/GUARDAR
AUTOPROG
ESCRIBIR
BORRAR
¿BORRADO?
AUTO/CONEX
MASTER-PROG Listo en el puerto USB
PIC Encontrado.
Familia Midrange/Standard Seleccionada
Dispositivo:
Bits ID:
FF FF FF FF
CP/CPD:
Valor OSCCAL:
Valor BandGap:
Palabra de Configuración:
2FCF
PROGRESO
0%
100%

Pasos realizados:

Desarrollamos este programa con el propósito de controlar el movimiento de un robot mediante dos sensores conectados a los pines RA0 y RA1 del microcontrolador PIC16F84A. Estos sensores seguimiento de línea, permiten al sistema detectar distintos escenarios y responder en consecuencia mediante los motores conectados al puerto B.

Al comenzar, configuramos el sistema para que el puerto B funcione como salida, ya que ahí se conectan los motores, mientras que el puerto A se ajusta para recibir señales de entrada desde RA0 y RA1. Esto permite al microcontrolador interpretar el entorno a través de los sensores.

En el ciclo principal del programa, el microcontrolador lee constantemente el estado de RA0 y RA1. A través de una operación lógica, se aísla la información de esos dos pines y se utiliza su valor combinado para hacer un salto indexado, una técnica eficiente que nos permite dirigir la ejecución a distintas secciones del código, dependiendo del caso. Si ambos sensores están activos, el robot avanza recto. Si solo uno detecta, el robot corrige su dirección encendiendo el motor opuesto. Y si ninguno detecta, realiza un giro para buscar la línea u obstáculo.

Este enfoque evita el uso de estructuras condicionales tradicionales como IF, y en su lugar implementamos una lógica más directa y optimizada para el hardware limitado del PIC. Esta estrategia permite al robot reaccionar en tiempo real a los estímulos, con un comportamiento autónomo sencillo pero funcional.

Como equipo, consideramos que este proyecto nos ayudó a reforzar nuestros conocimientos sobre programación en ensamblador, lógica digital y control de sistemas embebidos, además de fomentar la colaboración y la distribución de tareas técnicas de forma efectiva.

Conclusión:

Al terminar este proyecto pudimos entender mejor cómo funciona un microcontrolador y cómo se puede usar para controlar un robot seguidor de línea. Aunque al principio parecía complicado, poco a poco fuimos entendiendo cómo leer los sensores y cómo hacer que el robot tomara decisiones usando el salto indexado.