UNIVERSIDAD DE EXTREMADURA

ESCUELA DE INGENIERÍAS INDUSTRIALES

Grado en Electrónica Industrial y Automática

INFORMATICA INDUSTRIAL 2023/2024

INFORME DEL TRABAJO PRÁCTICO

GRUPO: T2

ALUMNOS:

AUNIÓN SÁNCHEZ, JAVIER

GUERRERO BECERRA, DAVID

MEDINA PARRA, MANUEL

TERRAZAS LOBATO, ADRIÁN

ZORITA NUÑEZ, JUAN JOSÉ

ÍNDICE

1 OBJETIVOS DEL TRABAJO

El objetivo del trabajo consiste en el diseño, montaje y programación de un robot que pueda circular de forma autónoma cambiando de dirección y sentido en caso de existir obstáculo en su trayecto, añadiendo además señalización de marcha atrás y alumbrado delantero en caso de existir poca luz

El diseño del robot se ha realizado mediante un software de diseño profesional llamado “SolidWorks”, que posteriormente cada pieza creada se ha impreso con impresoras 3D

En la programación, se ha utilizado un microcontrolador PIC16F15376 con una tensión de alimentación de 4,5 V conseguido a través de un porta pilas de tres de 1,5 V cada una conectadas en serie. El microcontrolador ha sido programado mediante MPLAB X IDE en lenguaje ensamblador donde se han utilizado los siguientes recursos de este mismo:

- Configuración de la MCU para que el oscilador interno este configurado con una frecuencia de 32 MHz y también se deshabilite el perro guardián y el oscilador externo.
- Se han utilizado los puertos A y E del microcontrolador donde cada pin corresponde con la siguiente variable:
  - RA0 → Final de carrera derecha → SENSOR1
  - RA1 → Final de carrera izquierda → SENSOR2
  - RA2 → LDR → LDR
  - RA3 → Marcha atrás motor (Puente en H) → INPUT2
  - RA4 → Marcha adelante motor (Puente en H) → INPUT1
  - RA5 → Enable (Puente en H) → ENABLE
  - RA6 → Servomotor control dirección → SERVO
  - RA7 → LED rojo de marcha atrás → LED_ATRAS
  - RE0 → LED´s iluminación frontal → FAROS
- Se han utilizado los temporizadores TMR0 y TMR1 de la siguiente forma:
  - TMRO es utilizado para dar la señal al servomotor de la dirección precisa, a la misma vez que comprueba si ha saltado la interrupción de los finales de carrera (Bigotes) y la lectura del valor de luminosidad de la LDR.
  - TMR1 es un contador el cual es usado para comparar su valor con el del módulo CCP1.
- Implementación del módulo CCP1 de captura y comparación con el Timer1 para la posición del servomotor que controla la dirección del robot.
- Implementación del módulo ADC para realizar una conversión analógico digital del LDR conectado en el pin RA2.

- Se utilizarán dos interrupciones diferentes:
    - Interrupción del desborde del TMR0.
    - Interrupción para la comparación del TMR1 con los valores de CCP1.
    - Interrupción de los periféricos asociado a los finales de carrera (Bigotes).

2 DESCRIPCION Y EXPLICACION PARTE MECANICA

2.1 MONTAJE DE LA ESTRUCTURA

Para el montaje de la mecánica, se ha diseñado nuestro robot a través del programa informático de diseño “SolidWorks”.

La mayor parte del robot será fabricada por nosotros mismo desde el diseño ya mencionado anteriormente con la impresión en plástico (PLA), mediante impresoras 3D (Ender 3 y Creality CR10 max).

2.1.1 Chasis principal

El chasis principal es un rectángulo de 200x150x3 mm. Además, este chasis está adaptado para la cogida de las ruedas.

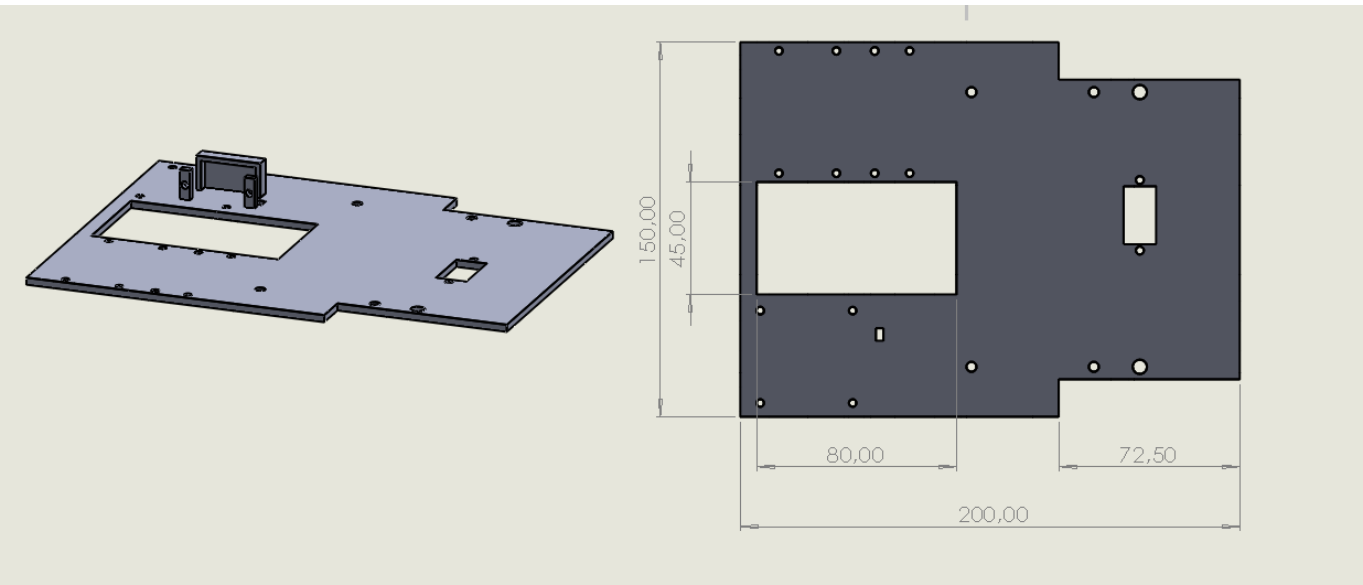

Figura 2.1 Vista en planta y tridimensional del chasis con cotas.

Además, en este chasis se han posicionado diferentes diámetros de taladros para las sujeciones de las diferentes placas electrónicas.

2.1.2 Ruedas delanteras

Las ruedas delanteras son las que permiten el giro del robot, por lo tanto, para que estas no giren libremente, van a estar controladas mediante un servomotor.

Dicho servomotor va a ir acoplado a una leva que realizara el giro mediante la transmisión de Ackerman como se muestra en la figura 2.2.

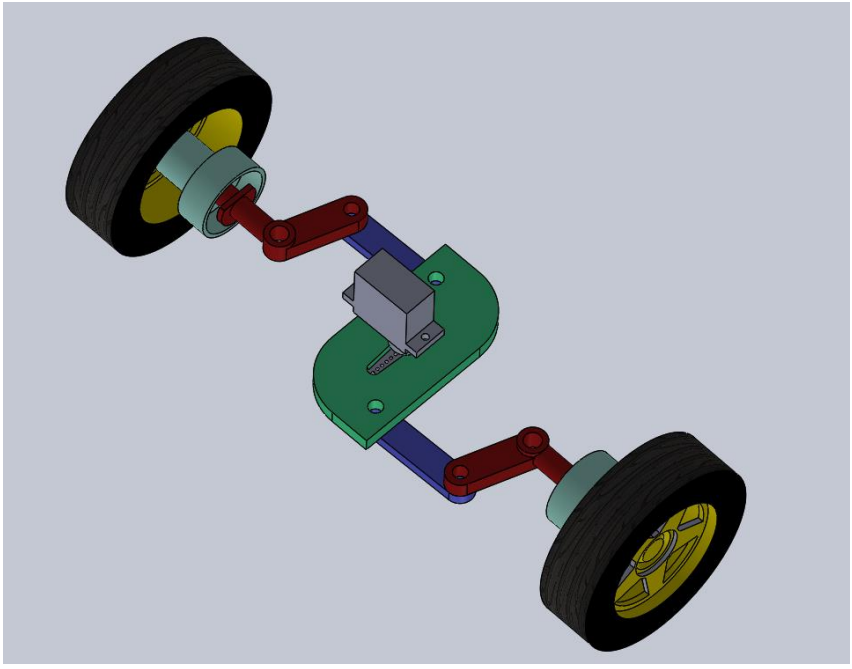

Figura 2.2 Diseño transmisión de Ackerman.

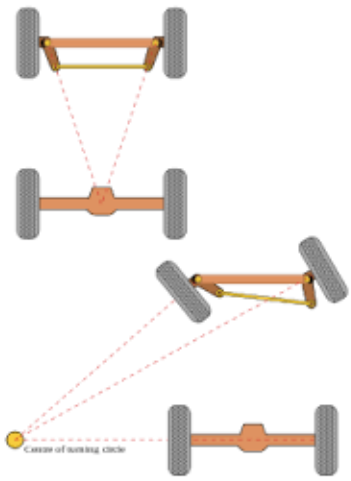

Figura 2.3 Principio de la transmisión de Ackerman

El acoplamiento de las ruedas delanteras se ha hecho mediante el diseño de la pieza de la figura 2.4, donde por un lado encaja la rueda y por el otro un rodamiento de diámetro exterior 24 mm e interior 8 mm.

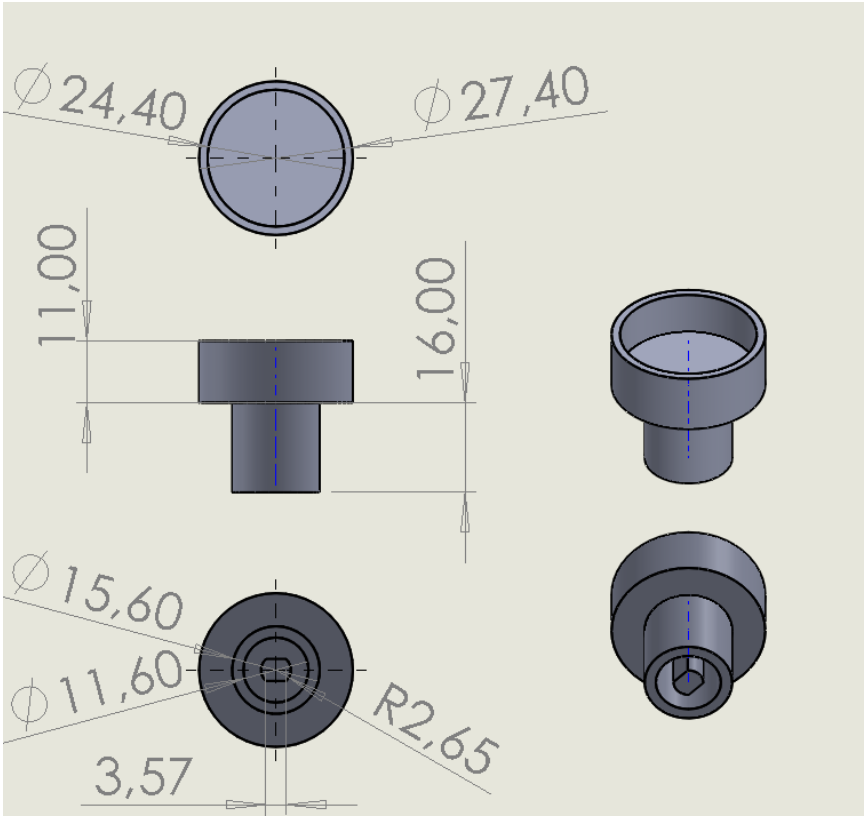

Figura 2.4 Acoplamiento ruedas delanteras.

Posteriormente se han diseñado unos brazos juntos con unas uniones que se atornillan a la leva de dirección con las siguientes características mostradas en las dos siguientes figuras.

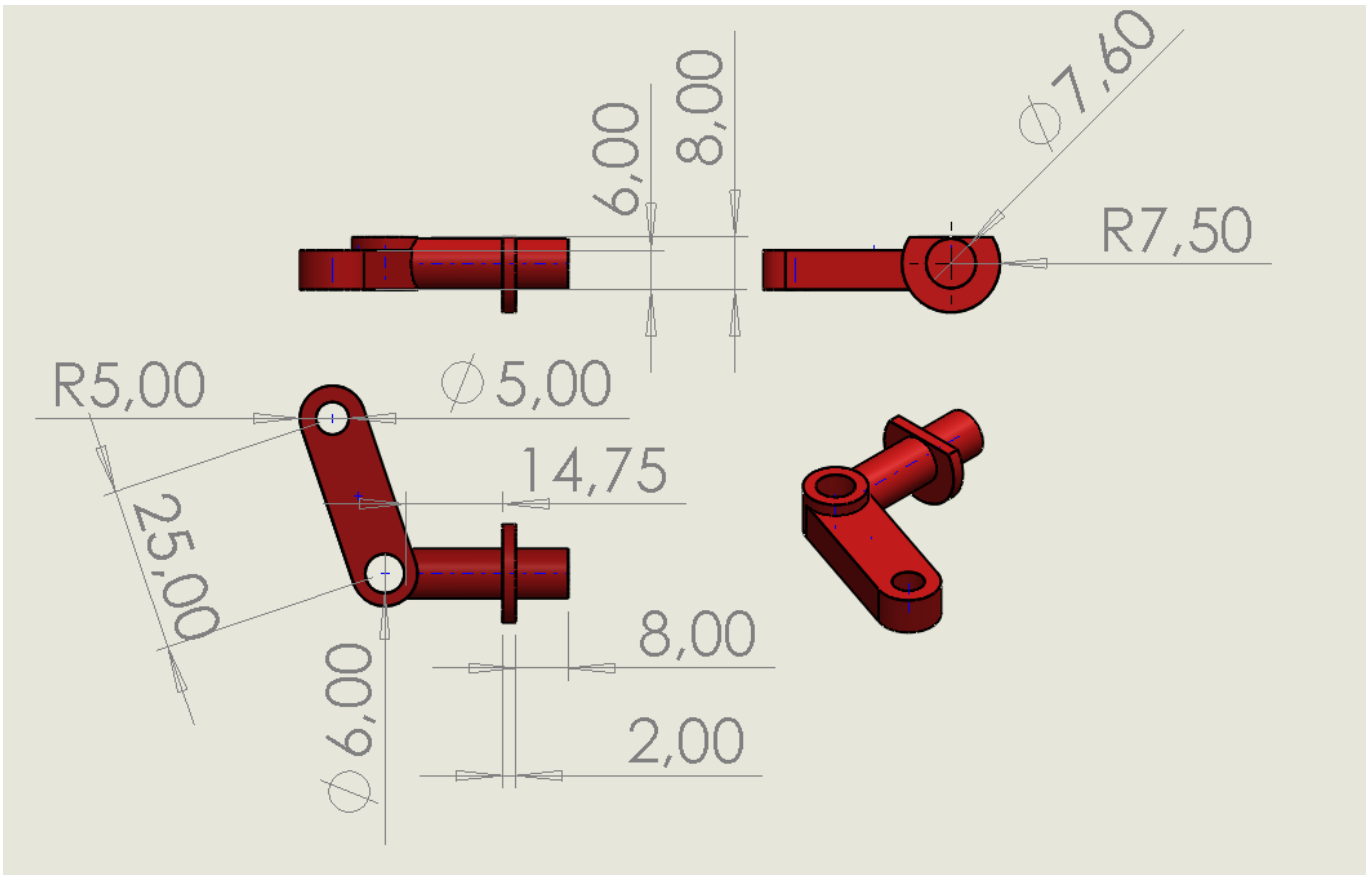

Figura 2.5 Brazo de transmisión de giro acoplado al rodamiento.

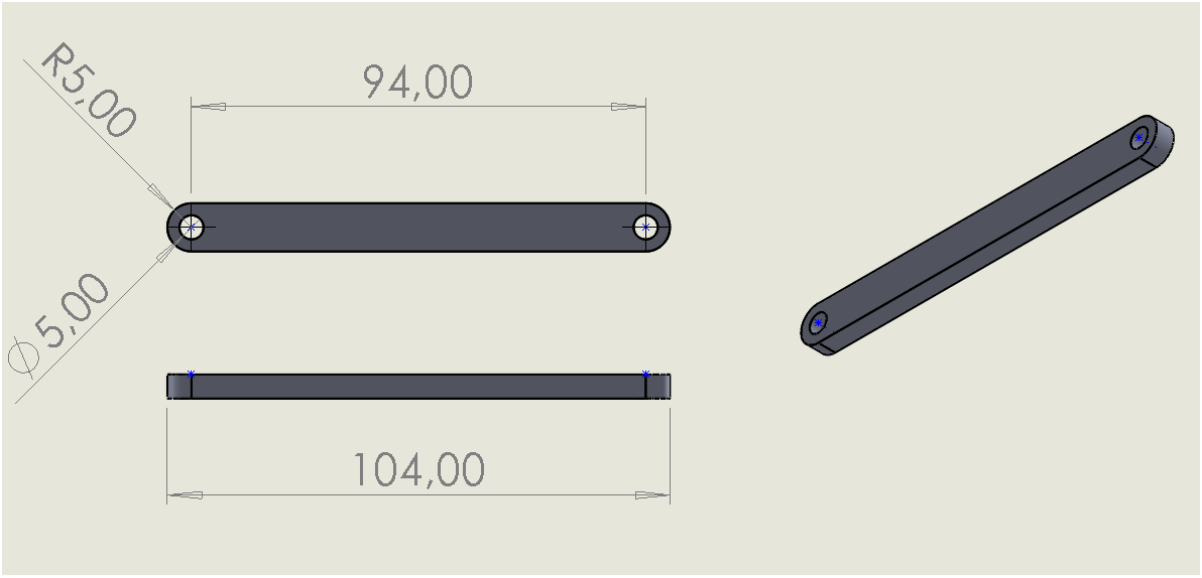

Figura 2.6 Uniones del brazo a la leva.

A continuación, se muestra el diseño de la leva, para ver el acoplamiento que se ha realizado en el servomotor, junto con las dimensiones de la aleta del servomotor.

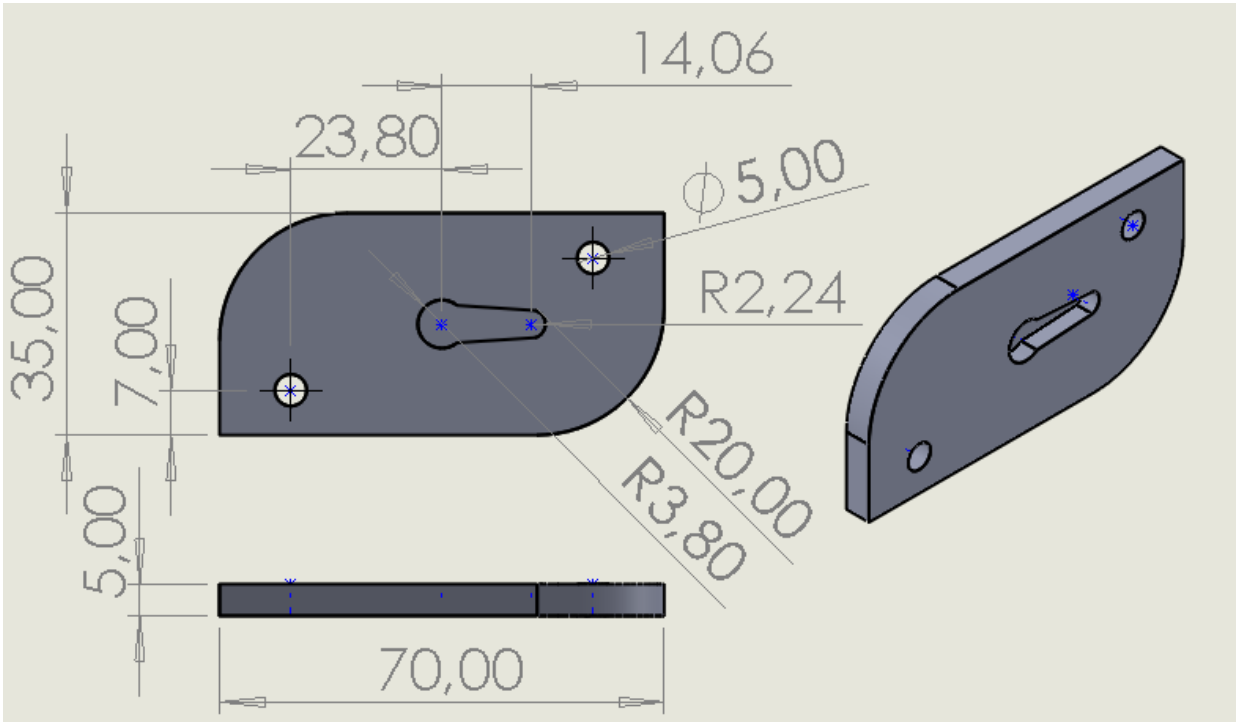

Figura 2.7 Diseño de la leva.

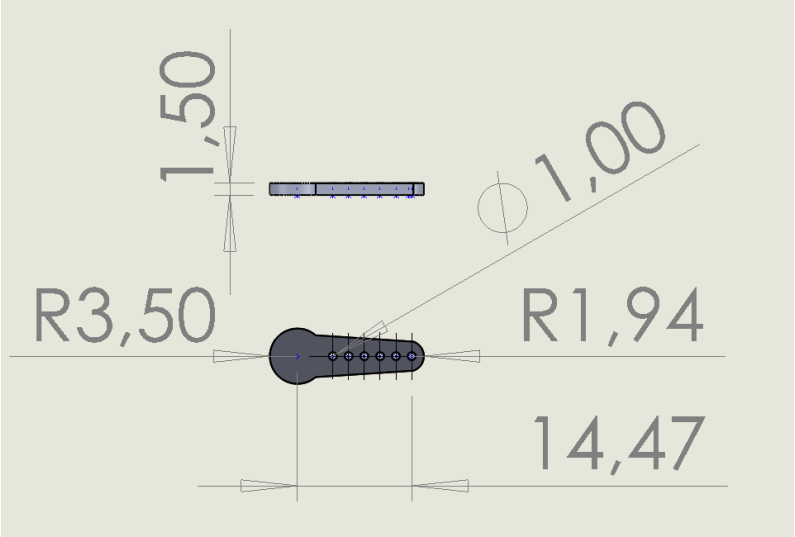

Figura 2.8 Dimensiones aleta simple conectada a la leva.

Las dimensiones del servomotor vienen especificadas en la figura 2.12.

Las uniones de estas piezas se realizan mediante tornillos pasantes de diámetros especificados según los taladros.

2.1.3 Ruedas traseras

En la parte trasera se ha posicionado una única rueda ya que al contar con un solo motor para la transmisión se debería de insertar un diferencial de potencia. Entonces para ahorrarnos ese paso de la mecánica hemos colocado una única rueda centrada acoplada a un motor como se muestra en la figura 2.10.

Tanto las ruedas delanteras como la trasera son del mismo diámetro de 68 mm.

Figura 2.9 Dimensiones ruedas.

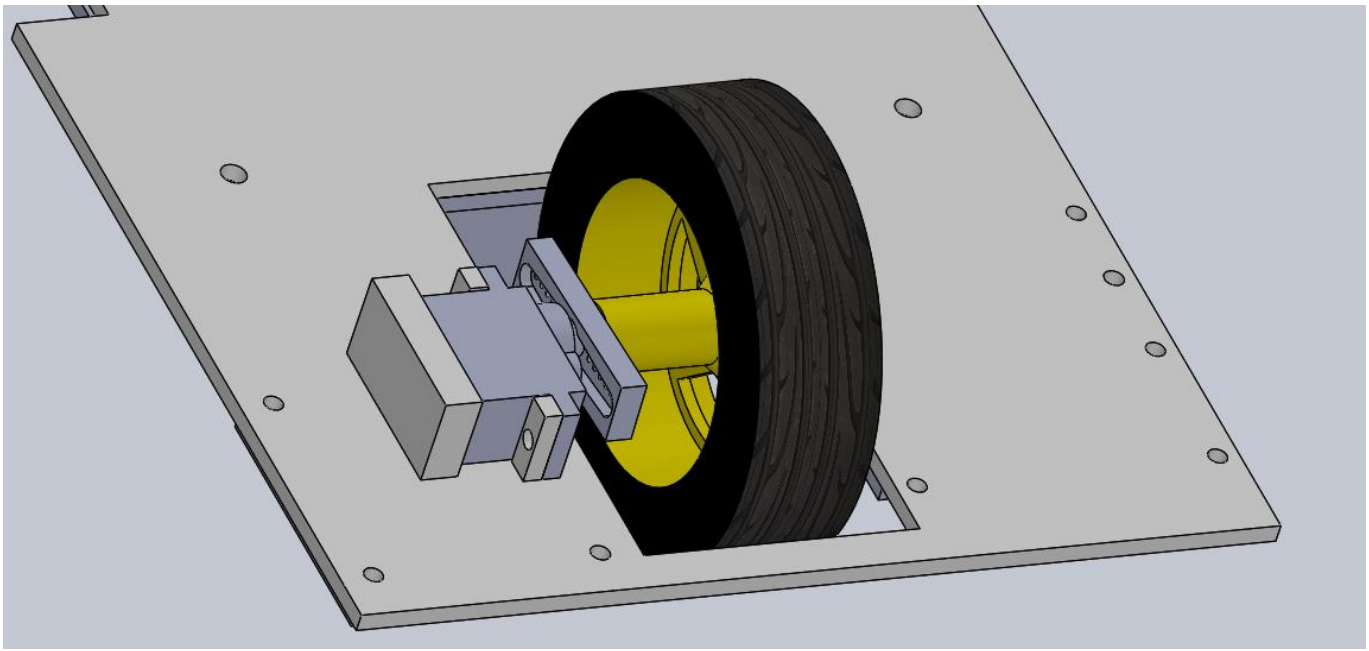

Figura 2.10 Diseño de acoplamiento del motor a la rueda trasera.

Para el acoplamiento del motor a la rueda se ha diseñado una pieza como se muestra en la figura 2.11.

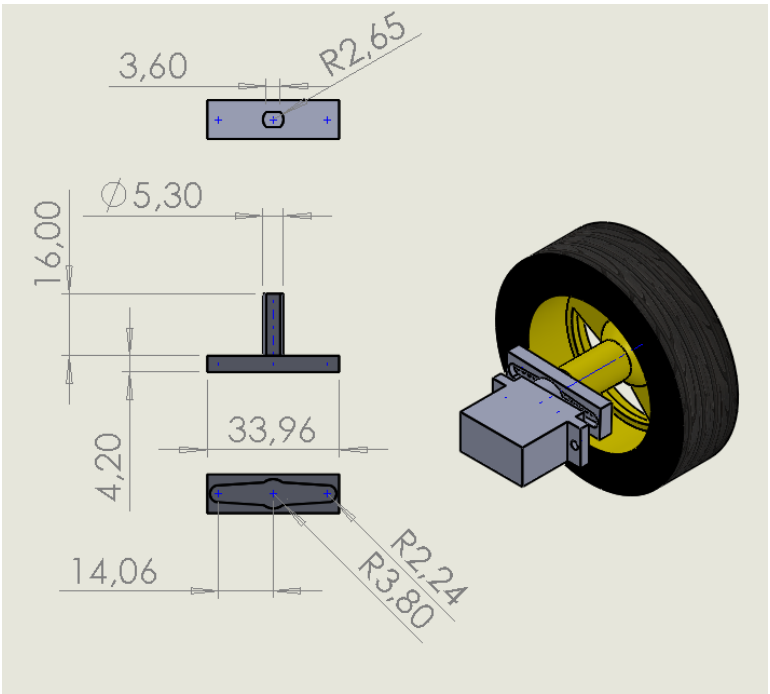

Figura 2.11 Dimensiones y diseño de la pieza que transmite el giro del motor a la rueda.

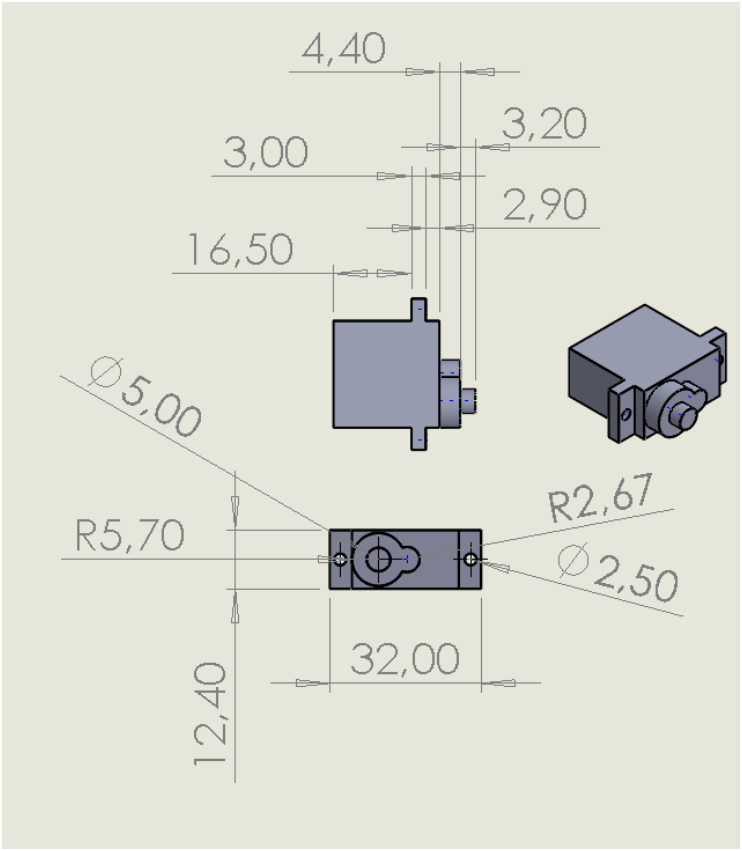

Figura 2.12 Dimensiones del motor de tracción (Las mismas que las del servomotor).

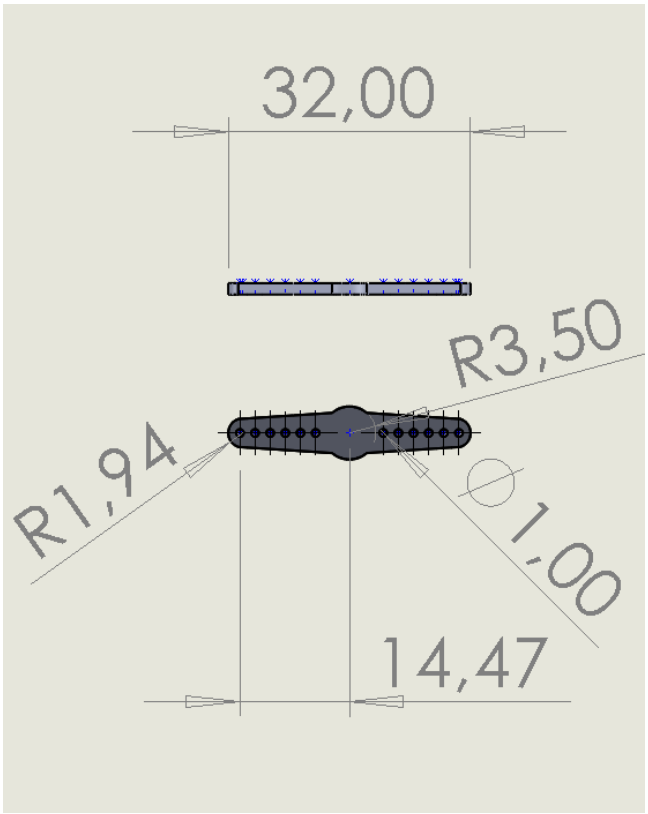

Figura 2.13 Dimensiones aleta doble que se conecta a la parte de giro del motor.

Como parte final de la mecánica se añade adjunto una foto del ensamblaje completo del robot, mostrado en la figura 2.14.

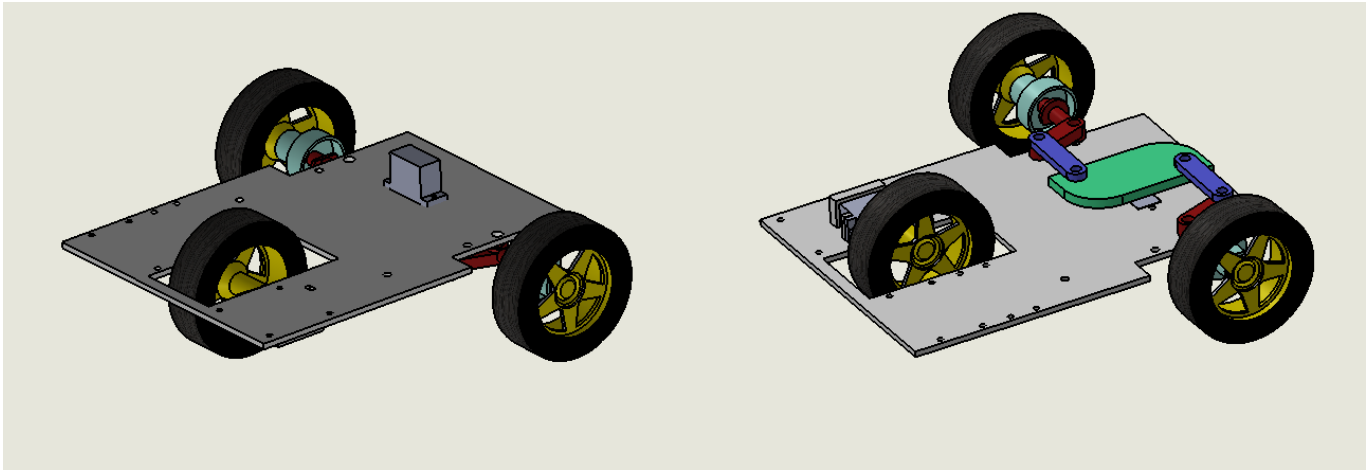

Figura 2.14 Robot.

2.2 UBICACIÓN Y SELECCIÓN COMPONENTES ELECTRÓNICOS

2.2.1 Selección componentes electrónicos

2.2.1.1 Micro servomotor mg90s (tracción).

Utilización de un servomotor para la transmisión de la rueda trasera, previamente lo trucamos siguiendo el pdf que se encuentra en el campus virtual “Trucaje de Servos de RC”.

Características:

- Peso: 13,4 g.
- Dimensiones: 22,5 x 12 x 35,5 mm aprox.
- Velocidad: 0,1s/60º a 4,8V.
- Torque: 1,8 kgf *cm a 4,8V.
- Voltaje de funcionamiento: 4,8V – 6,0V
- Corriente máxima: 400 mA
- Ángulo de rotación: 180º
- Rojo = VCC (+), Marrón = Ground (-), Naranja = Señal PWM.

Figura 2.15 Servomotor

2.2.1.2 Micro servomotor mg90s (dirección).

Utilización de un servomotor para regular la dirección de las ruedas delanteras del camión, mediante la transmisión de Ackermann.

Características:

- Peso: 13,4 g.
- Dimensiones: 22,5 x 12 x 35,5 mm aprox.
- Velocidad: 0,1s/60º a 4,8V.
- Torque: 1,8 kgf *cm a 4,8V.
- Voltaje de funcionamiento: 4,8V – 6,0V
- Corriente máxima: 400 mA
- Ángulo de rotación: 180º
- Rojo = VCC (+), Marrón = Ground (-), Naranja = Señal PWM.

Figura 2.16 Servomotor.

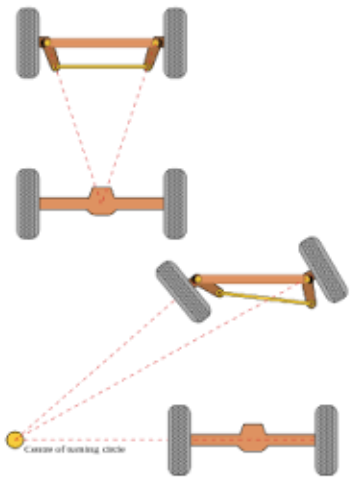

Figura 2.17 Geometría de Ackerman

2.2.1.3 Controlador l298n, doble puente en h.

Controlador L298N, con doble puente H, para control de motores en DC. Se utiliza este controlador para la inversión de giro del motor que transmite el movimiento a la rueda trasera.

Características:

- Voltaje lógico: 5 V.

- Voltaje de operación: 5-24 V.
- Corriente máxima: 2A.

Figura 2.18 Doble puente H.

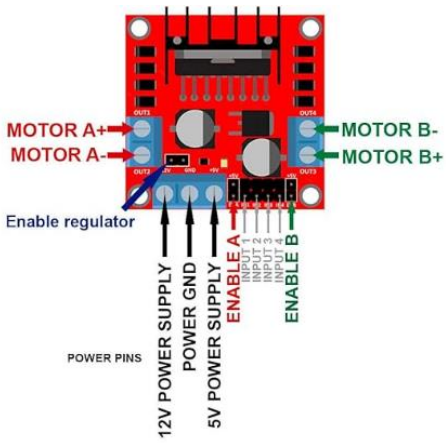

Figura 2.19 Esquema doble puente H.

2.2.1.4 LED marcha atrás.

Diodo LED de color rojo que indica cuando nuestro robot va hacia atrás. Posee las siguientes características:

- Caída directa de 1.8-2.2 VDC.
- Corriente máxima: 20 mA.
- Intensidad luminosa: 120 mcd.

Figura 2.20 Diodo LED rojo..

2.2.1.5 LDR.

Resistencia dependiente de la luz que se encargará de captar la luz del exterior.

Características:

- Voltaje máximo de 100 V.
- “Light resistance” de 20 a 100 kΩ.
- “Dark resistance” mínima de 20 MΩ.

Figura 2.21 Fototransistor.

2.2.1.6 Faros.

Dos diodos LED de color azul que se encenderán cuando la LDR no capten suficiente luz del exterior. Características:

- Caída directa de 3.5-3.8 VDC.
- Corriente máxima: 20 mA.
- Intensidad luminosa: 270 mcd.

Figura 2.22 Diodo LED azul.

2.2.1.7 Pic16f15376 Curiosity nano.

Características:

- Microcontrolador PIC16F15376-I/MV.
- Un LED de usuario amarillo.
- Un interruptor de usuario mecánico.
- Huella para cristal de 32,768 kHz.
- Depurador integrado:
  – Identificación de placa en Microchip MPLAB® X
  – Un LED verde de alimentación y estado
  – Programación y depuración
  – Puerto COM virtual (CDC)
  – Un canal analizador lógico (DGI GPIO)
- Alimentado por el pin 5V.
- Voltaje objetivo ajustable:
  – Regulador LDO MIC5353 controlado por el depurador integrado.
  – Voltaje de salida de 2,3-5,1 V (limitado por el voltaje de entrada USB).
  – Corriente de salida máxima de 500 mA (limitada por la temperatura).

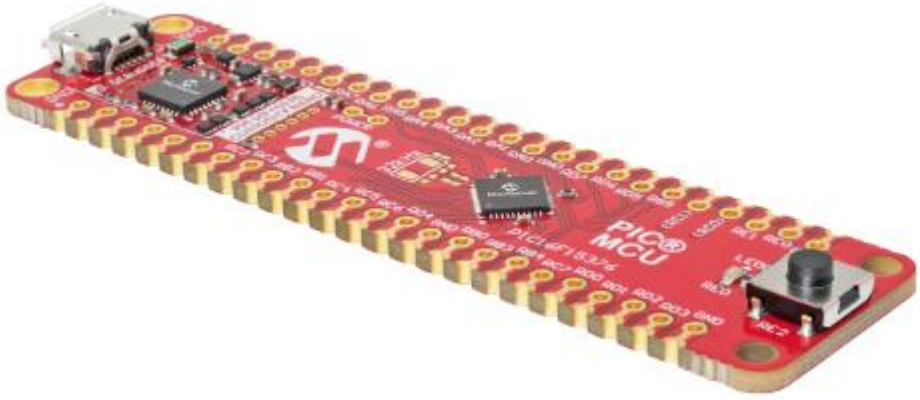

Figura 2.23 PIC16F15376

2.2.1.8 Final de carrera

Utilización de 2 finales de carrera, dispuestos en los laterales del frontal del camión, para la reorientación del mismo en caso de colisión.

El final de carrera es un micro interruptor con acción resorte de palanca larga, con un contacto de cada tipo NO y NC, con voltaje de 5 VDC.

Figura 2.24 Microinteruptor.

2.2.2 Ubicación placas electrónicas

En la parte frontal del robot ira ubicada la placa electrónica de los faros y los bigotes, ya que es la encargada de alumbrar el camino si no detecta luz y los bigotes de corregir la dirección en caso de colisión.

Además, se han diseñado dos tubos para ubicar la LDR y el LED de marcha atrás.

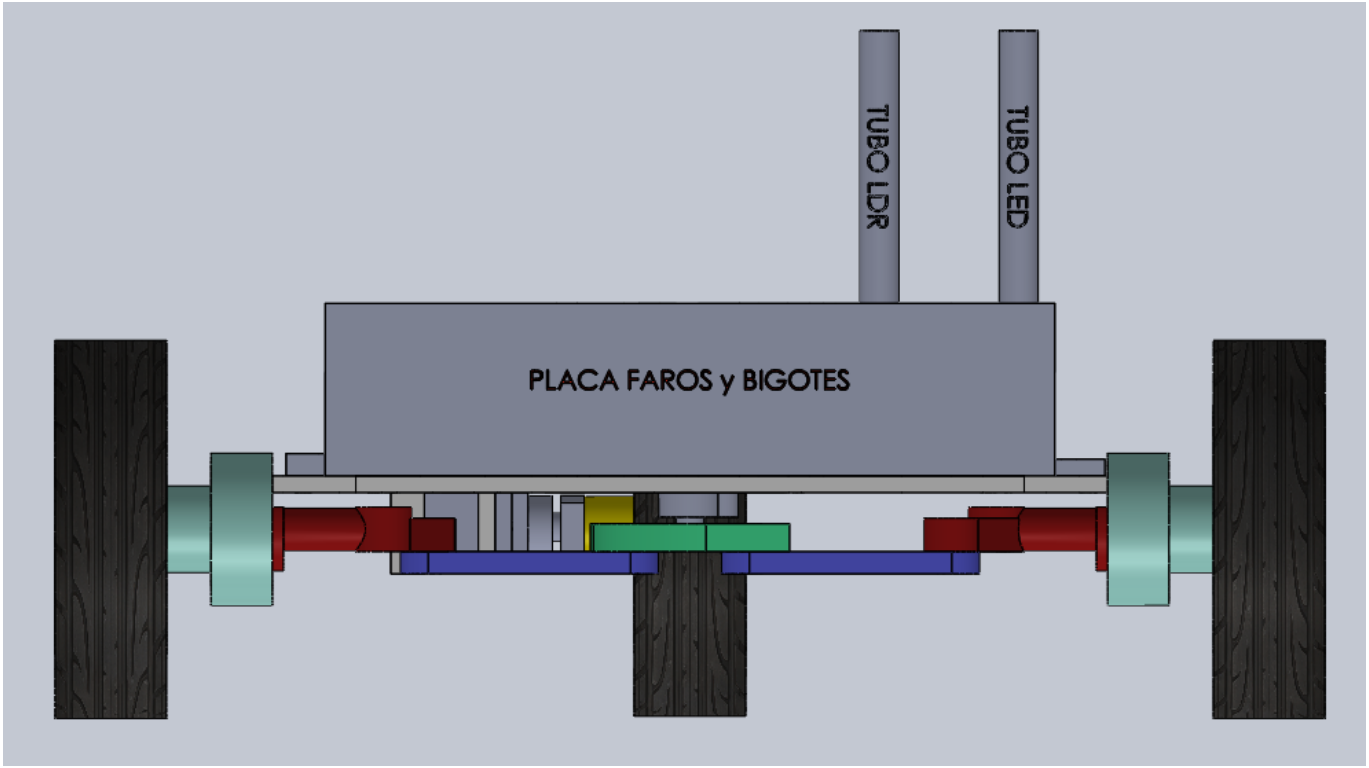

Figura 2.25 Placa de los faros y bigotes.

Las demas placas las situaremos en la base del chasis como se indica en la siguiente figura 2.26.

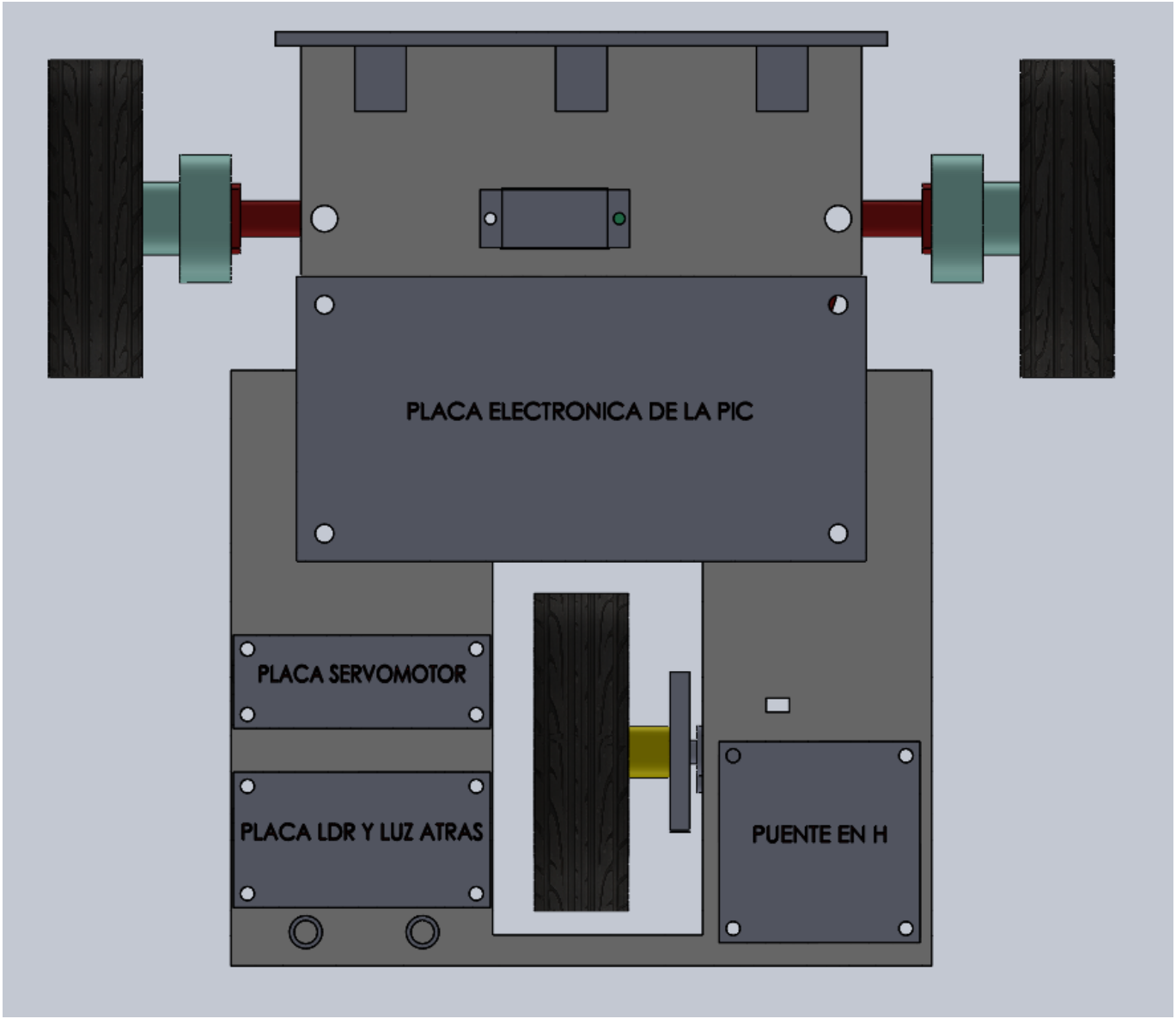

Figura 2.26 Ubicación de las placas electrónicas.

A continuación, se mostrarán imágenes de las placas electrónicas para ubicar los componentes electrónicos que forman parte de cada una de ellas.

En la placa electrónica de figura 2.27 se ha ubicado la LDR que nos informa de la cantidad de luz en el entorno y el LED rojo que se enciende cuando el robot va marcha atrás. Estas dos entradas estan ubicadas en los tubos de la figura 2.25.

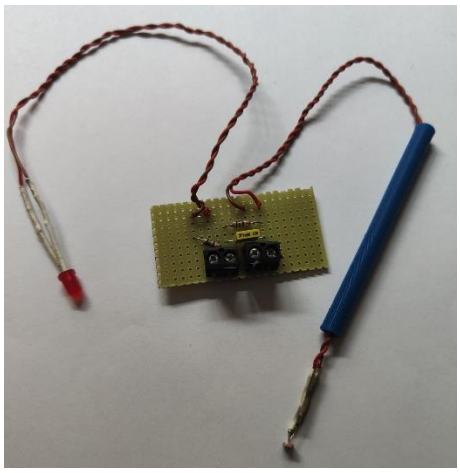

Figura 2.27 Placa electrónica de la LDR y el LED de marcha atrás.

Placa electrónica que controla la posición del servomotor encargado de hacer girar la transmisión de Ackermann ya mencionada anteriormente.

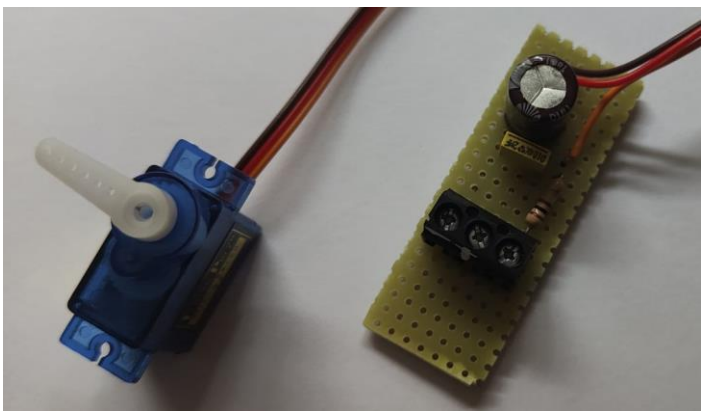

Figura 2.28 Placa electrónica del servomotor.

Esta placa electrónica es la que está situada en el frontal del robot, ya que estan los LED que hemos denominado como faros, que alumbran el camino a seguir y los finales de carrera denominados bigotes, a los que se les ha acoplado unos alambres para tener una distancia de seguridad entre el obstáculo y el robot. Cuando los bigotes sean accionados, el robot ejecuta la acción de corregir el sentido y dirección de la marcha.

Figura 2.29 Placa electrónica de los faros y los bigotes.

La siguiente figura 2.30 corresponde con el puente en H que es el encargado de polarizar el motor de tal forma que gire a izquierdas o derechas.

El motor dispuesto es un servomotor trucado, que es el que manda el par necesario a la rueda trasera para que pueda moverse el robot.

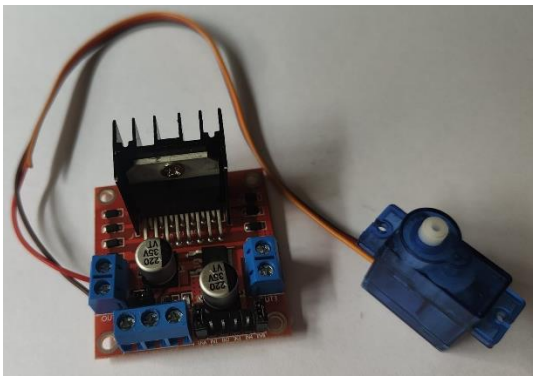

Figura 2.30 Puente en H y motor tracción.

La placa electrónica de la PIC, consta tan solo de la PIC y de todos los terminales donde poder conectar entradas, salidas y alimentación de los circuitos. La alimentación de esta misma, viene a estar conectada en serie junto con un interruptor para poder apagar o encender cuando sea necesario.

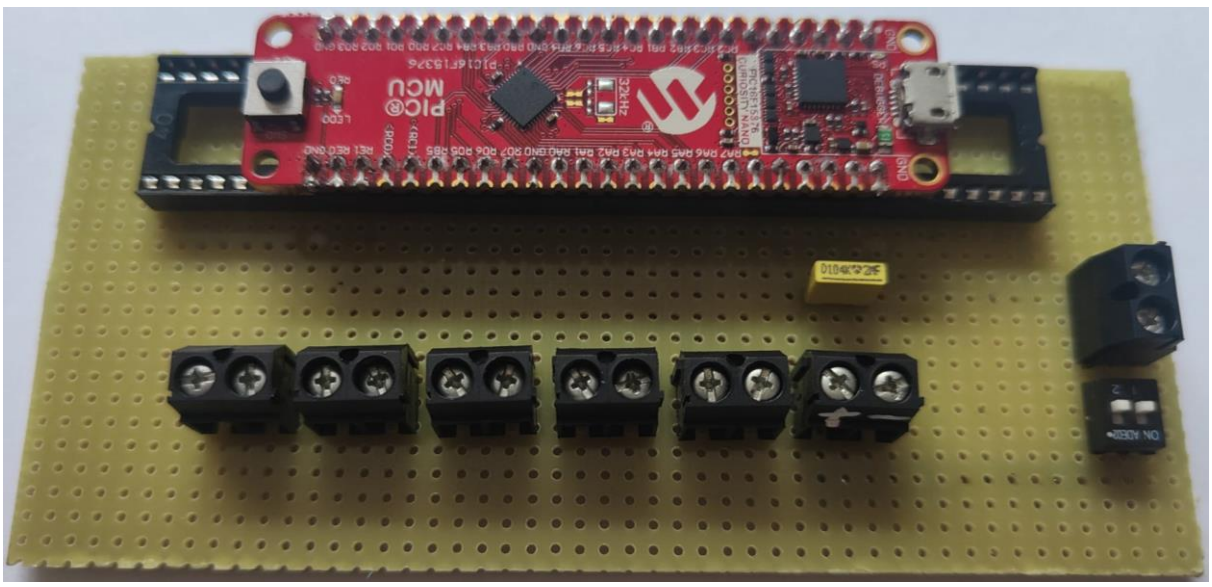

Figura 2.31 Placa electrónica de la PIC.

La alimentación del circuito se realiza mediante un porta pilas, con tres pilas de 1,5 V cada una conectadas en serie, de tal forma que la alimentación es 4,5 V.

Figura 2.32 Alimentación mediante porta pilas.

3 DESCRIPCION Y EXPLICACION PARTE ELECTRONICA

3.1 ESTABLECIMIENTO DE LOS REQUERIMIENTO DE LA FUENTE DE ALIMENTACIÓN.

Para poder elegir la alimentación del circuito necesitamos saber los consumos máximos de cada componente y el voltaje que trabaja cada uno de los componentes que vayamos a utilizar. A continuación, observamos los requerimientos:

- LED de color rojo para indicar cuando va hacia atrás, con un voltaje de 1.8 a 2.2 V y corriente de 20 mA.
- Servomotor con tensión de alimentación desde 4.5-6 V (en nuestro caso alimentaremos con 4.5 V) y un consumo aproximadamente de 420 mA.
- Puente en H: Voltaje de funcionamiento desde 4.5V hasta 35V, corriente máxima de 2x2 A.
- PIC16F15376: Tensión alimentación de 4.5V y consumo máximo de 500mA.
- LED de color azul con un voltaje de 3.5 a 3.8 V y corriente de 20 mA.
- Final de carrera: 3 pines (5A 125-200V).
- Caja de pilas de 4.5 V que cuenta con 3 pilas de 1,5V AA.
- LDR con una “light resistance” de 20 a 100 kΩ y una “dark resistance” mínima de 20 MΩ.

3.2 ESTABLECIMIENTO DEL DIAGRAMA DE BLOQUES DEL SISTEMA.

Diagrama de bloque del robot donde mediante la figura 4.1 se puede visualizar. Como alimentación tendremos la caja de pilas que se conecta con el convertidor de tensión para alimentar nuestra Curiosity nano a 4.5V.

Posteriormente contaremos con:

- Entrada analógica de la LDR.
- Entrada digital de los finales de carrera, utilizados como sensores de proximidad.
- Salida digital del led rojo, utilizado para marcha atrás.
- 3 salidas digitales para controlar el puente en H.
- Salida digital del led rojo, utilizado para marcha atrás.
- Salida digital de dos LEDs azul.
- Salida digital del servomotor encargado de la dirección de nuestro robot.

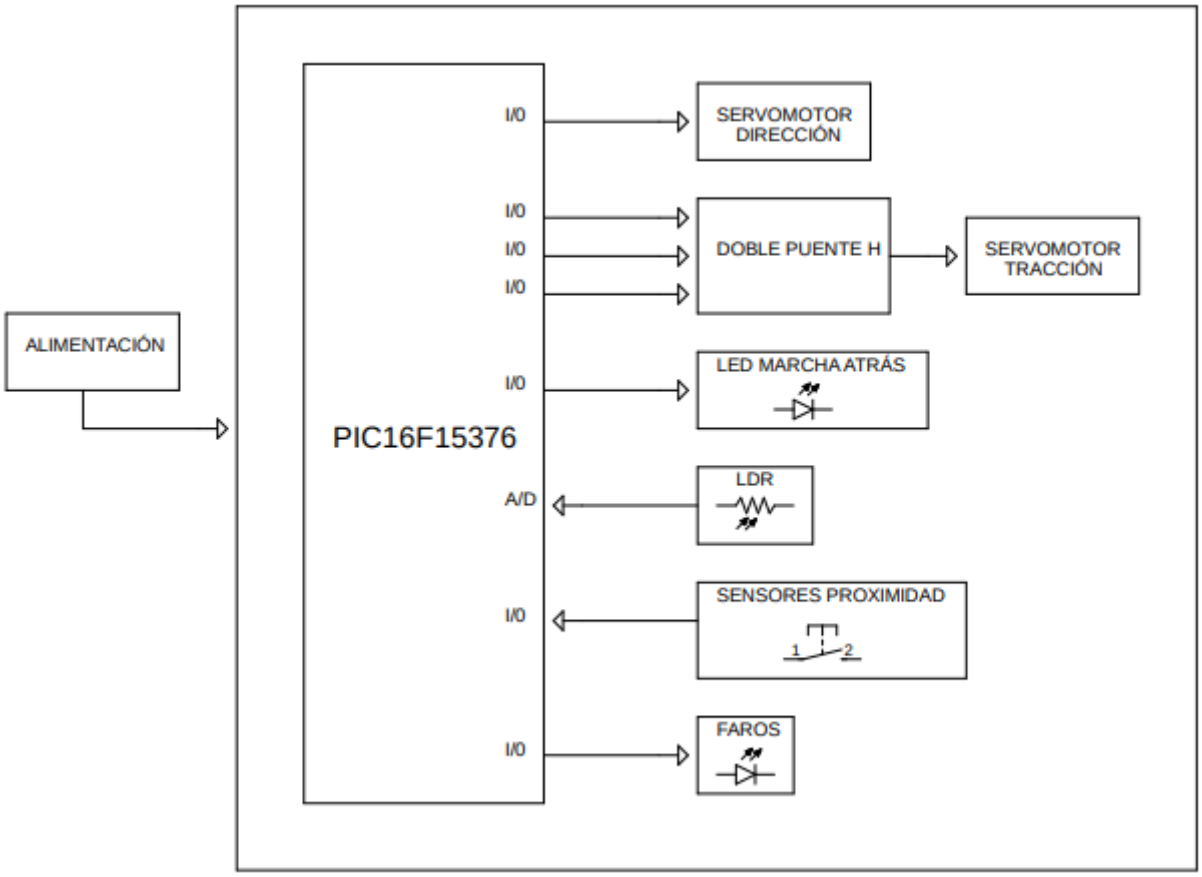

Figura 3.1 Diagrama bloque robot.

3.3 DISEÑO DE ESQUEMAS ELECTRÓNICOS.

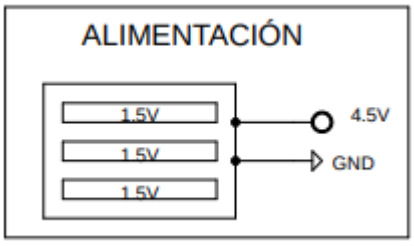

Figura 3.2 Alimentación.

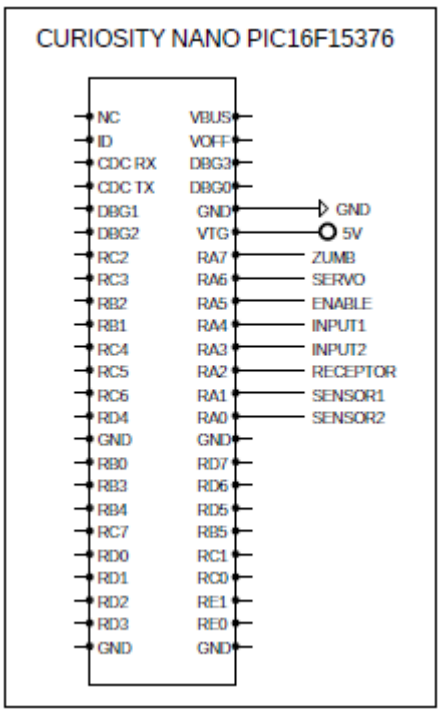

Figura 3.3 Curiosity nano PIC16F15376.

Para los LED a forma de los faros del robot, hemos hecho los cálculos para las resistencias, usando una corriente máxima de 20mA, una V_{fuente} de 4,5V y como son azules, una V_{LED} de 3,5V a 3,8V.

$$R = \frac{V_{fuente} - V_{LED}}{I_{máx}}$$

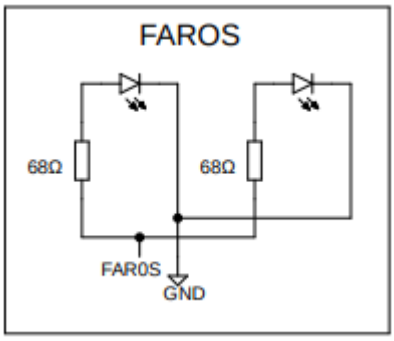

Figura 3.4 Faros.

Nuestra LDR, tiene una “light resistance” (cuando recibe luminosidad) de 20Ω a 100kΩ, y una “dark resistance” (a oscuras) mínima de 20MΩ.

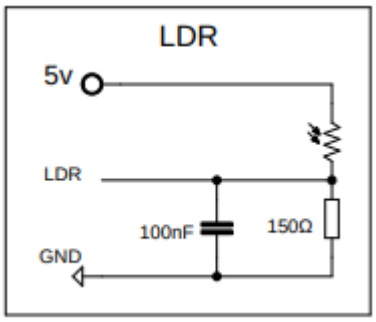

Figura 3.5 LDR.

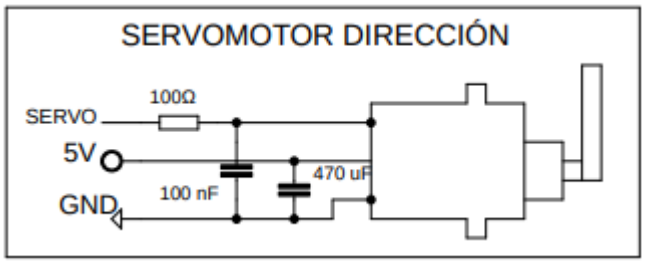

Figura 3.6 Servomotor dirección.

El servomotor de tracción será el trucado, para que funcione a forma de un motor DC, de tal forma que irá hacia delante cuando activemos “Input 1”, y marcha atrás cuando activemos “Input 2”.

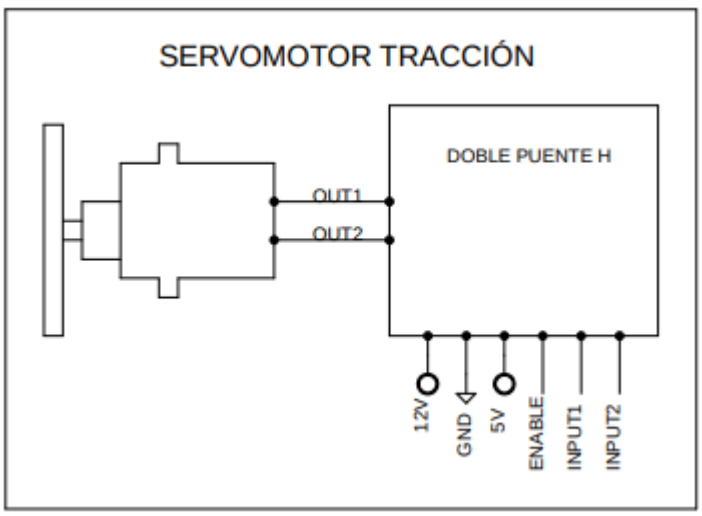

Figura 3.7 Servomotor tracción.

Para el LED de la marcha atrás, hemos hecho un cálculo para elegir la resistencia que tenemos que poner. Este LED es rojo, por lo que tendremos una V_{LED} de 1,8V a 2,2V. En éste tenemos una corriente $I_{máx}$ de 20mA también.

$$R = \frac{V_{LED} - 0}{I_{máx}}$$

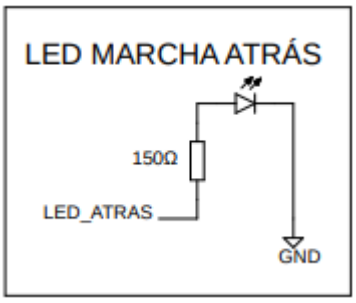

Figura 3.8 Led marcha atrás.

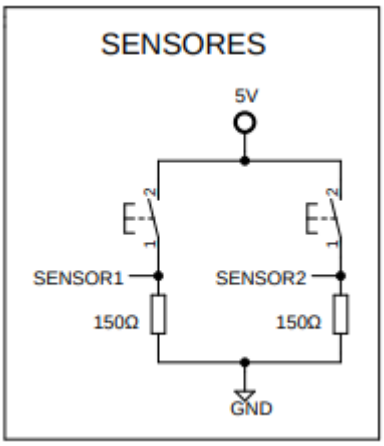

Figura 3.9 Sensores.

Y a continuación mostramos el esquema electrónico completo del proyecto:

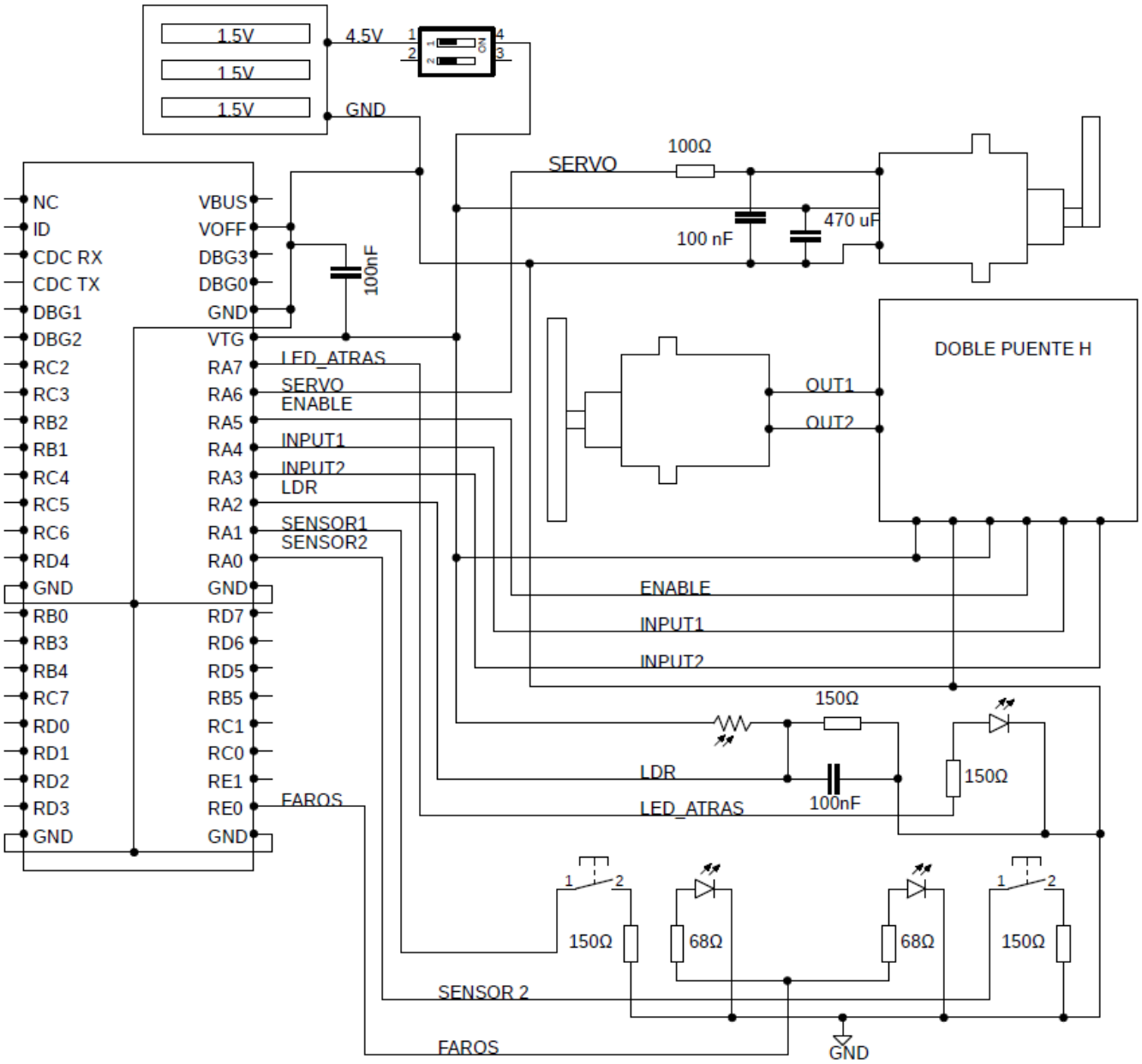

Figura 3.10 Esquema electrónico completo

En el esquema se puede observar que es necesario puentear el pin VOFF con todas las GND, asi como poner un capacitor cerámico de 100nF entre los 4.5v y GND a modo de filtro, para el funcionamiento autónomo del microcontrolador. Se ha añadido un interruptor de encendido, cuyo objetivo es controlar tanto la alimentación de la PIC como la descarga de la fuente de alimentación (3 pilas de 1,5v).

- SENSOR 1 Y 2: Están conectados en un extremo con GND y en otro a los pines RA0 y RA1 respectivamente, en serie con una resistencia de 150Ω.

- FAROS: Conectados por un terminal a GND y por otro con el pin RE0 en serie con una resistencia de protección de 68Ω.
- LDR: Para que nos arroje un valor analógico estable, conectamos una resistencia de pull-up en paralelo con un condensador cerámico de 100nF, donde el pin RA2 se conecta al divisor de tensión. Esta configuración nos arroja un valor máximo cuando capta luz y un valor mínimo cuando no la detecta.
- SERVO: Este dispositivo consta de tres terminales:
    - VCC → conectado a 4.5v
    - GND
    - SEÑAL

  Como se puede ver, conectamos el pin RA6 con una resistencia de protección de 100Ω en serie con un condensador cerámico de filtro de 100nF,el cual sus dos terminales se conectan a señal y a GND. En el terminal de VCC ponemos un condensador electrolítico de 470uF a modo de estabilizar el voltaje de alimentación.

4 DESCRIPCION PARTE SOFTWARE

4.1 CONFIGURACION INICIAL MICROCONTROLADOR

4.1.1 Bits de configuración.

Inicialmente en el módulo config MCU, deshabilitaremos el perro guardián (WDTE = OFF) y el oscilar externo (FEXTOSC = OFF) y configuraremos el oscilador interno a una frecuencia de 32 MHz (RSTOSC = HFINTPLL).

4.1.2 Configuración del puerto A

Configuración de los pines RA0-1 y RA3-7 como digitales, y RA2 como analógico (ANSELA = 00000100B). Al mismo tiempo se configura párrafos abajo, RA0-2 como entradas y RA3-7 como salidas (TRISA = 00000111B).

Activación de la resistencia de pull-up en los pines RA0-1 (WPUA = 00000011B).

Se puede observar en la tabla 4.1 la configuración de cada pin.

4.1.3 Configuración del puerto E

Configuración del pin RE0 como salida digital (ANSELE,0) (TRISE,0) con el comando “bcf” que lo pone a 0.

Tabla 4.1 Configuración de pines.

PUERTO	PINES	A/D	E/S	VARIABLE
PUERTO A	RA0	Digital	Entrada	Final carrera derecha
	RA1	Digital	Entrada	Final carrera izquierdo
	RA2	Analógico	Entrada	LDR
	RA3	Digital	Salida	Marcha atrás motor (Puente en H)
	RA4	Digital	Salida	Marcha adelante motor (Puente en H)
	RA5	Digital	Salida	Enable (Puente en H)
	RA6	Digital	Salida	Servomotor control dirección
	RA7	Digital	Salida	LED rojo de marcha atrás
PUERTO E	RE0	Digital	Salida	LED´s iluminación total

4.1.4 Configuración del módulo convertidor A/D

Este módulo ha sido configurado para la conversión de analógico a digital de la LDR, de tal forma que mirando el datasheet hay que configurar los siguientes registros.

- Registro “ADCON1”:
  - Bits 1-0: Voltaje positivo de referencia (VDD = 00b).
  - Bits 6-4: Selección de la fuente de reloj Fcos/32.
  - Bit 7: Justificación a la izquierda resultado conversión. Descartando los dos bits menos significativos de la conversión.

REGISTER 20-2: ADCON1: ADC CONTROL REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
ADFM	ADCS<2:0>			—	—	ADPREF<1:0>	
bit 7							bit 0

bit 7 **ADFM:** ADC Result Format Select bit
1 = Right justified. Six Most Significant bits of ADRESH are set to ‘0’ when the conversion result is loaded.
0 = Left justified. Six Least Significant bits of ADRESL are set to ‘0’ when the conversion result is loaded.

bit 6-4 **ADCS<2:0>:** ADC Conversion Clock Select bits
111 = FRC (dedicated RC oscillator)
110 = Fosc/64
101 = Fosc/16
100 = Fosc/4
011 = FRC (dedicated RC oscillator)
010 = Fosc/32
001 = Fosc/8
000 = Fosc/2

bit 3-2 **Unimplemented:** Read as ‘0’

bit 1-0 **ADPREF<1:0>:** ADC Positive Voltage Reference Configuration bits
11 = VREF+ is connected to internal Fixed Voltage Reference (FVR) module(1)
10 = VREF+ is connected to external VREF+ pin(1)
01 = Reserved
00 = VREF+ is connected to VDD

Figura 4.1 Registro ADCON1.

- Registro “ADCON0”:
  - Bit 0: habilitación ADC (1 = habilitado).
  - Bit 1: bit de estado de conversión.
  - Bits 7-2: selección del canal analógico.

20.4 Register Definitions: ADC Control

REGISTER 20-1: ADCON0: ADC CONTROL REGISTER 0

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
CHS<5:0>						GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

bit 7-2 **CHS<5:0>: Analog Channel Select bits**

000010 = RA2

bit 1 **GO/$\overline{\text{DONE}}$: ADC Conversion Status bit**

1 = ADC conversion cycle in progress. Setting this bit starts an ADC conversion cycle.
This bit is automatically cleared by hardware when the ADC conversion has completed.

0 = ADC conversion completed/not in progress

bit 0 **ADON: ADC Enable bit**

1 = ADC is enabled

0 = ADC is disabled and consumes no operating current

Figura 4.2 Registro ADCON0.

4.1.5 Configuración del TMR0

El TMR0 lo hemos utilizado para las interrupciones que tiene que atender cuando los Bigotes se chocan indicando que tiene que esquivar un obstáculo. También lo utilizamos para la comprobación con la LDR, de forma que cuando no hay luz ésta actúa como un circuito abierto y se encienden dos LED a forma de faros.

Para que tenga el tiempo de ciclo deseado, de 20ms, hemos utilizado una tabla Excel que nos proporcionó el profesor:

Reloj					T_out máximo					T_post maximo					
Fosc	Divisor	T_pre	Prescaler	T_in	8 bits (256 cuentas)		16 bits (65536 cuentas)		Postscaler	8 bits		16 bits		(deseado)	Nº cuentas
[MHz]	D	[us]	1:M_pre	[us]	[us]	[ms]	[ms]	[s]	1:X	[us]	[ms]	[ms]	[s]	T_post [ms]	N_counts
32	4	0,125	128	16	4096	4,10	1048,58	1,05	1	4096	4,10	1048,58	1,05	20,00	**1250,00**

↑ valores modificables

Valor de recarga (8 bits)	-994,00
Valor de recarga (16 bits)	64286,00
Valor a escribir en TMR0H	251 en base 10
Valor a escribir en TMR0L	30 en base 10
comprobación del total	64286

Figura 4.3 Cálculo de valores del TMR0.

De esta tabla tendremos que tener en cuenta los valores TMR0H (251) y TMR0L (30), que los pondremos en el apartado de definiciones de nuestro proyecto.

Por último, para configurarlo hemos mirado en el datasheet los registros que tenemos que tener en cuenta:

- Registro “T0CON1”:
  - Bit 3-0: bits de selección prescaler (1:128).

- Bit 4: bit de habilitación de asincronización de entrada
- Bits 7-5: bits de selección de fuente de reloj.

REGISTER 25-2: T0CON1: TIMER0 CONTROL REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
T0CS<2:0>			T0ASYNC	T0CKPS<3:0>			
bit 7							bit 0

bit 7-5 **T0CS<2:0>:** Timer0 Clock Source select bits
111 = LC1_out
110 = SOSC
101 = MFINTOSC (500 kHz)
100 = LFINTOSC
011 = HFINTOSC
010 = Fosc/4
001 = T0CKIPPS (Inverted)
000 = T0CKIPPS (True)

bit 4 **T0ASYNC:** TMR0 Input Asynchronization Enable bit
1 = The input to the TMR0 counter is not synchronized to system clocks
0 = The input to the TMR0 counter is synchronized to Fosc/4

bit 3-0 **T0CKPS<3:0>:** Prescaler Rate Select bit
1111 = 1:32768
1110 = 1:16384
1101 = 1:8192
1100 = 1:4096
1011 = 1:2048
1010 = 1:1024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

Figura 4.4 Registro T0CON1.

- Registro “T0CON0”:
  - Bit 3-0: bits de selección postcaler (1:1).
  - Bit 4: Timer0 opera a 16 bits.
  - Bits 7: modulo operativo.

REGISTER 25-1: T0CON0: TIMER0 CONTROL REGISTER 0

R/W-0/0	U-0	R-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
T0EN	—	T0OUT	T016BIT	T0OUTPS<3:0>			
bit 7							bit 0

bit 7 **T0EN:** Timer0 Enable bit
1 = The module is enabled and operating
0 = The module is disabled and in the lowest Power mode

bit 6 **Unimplemented:** Read as ‘0’

bit 5 **T0OUT:** Timer0 Output bit (read-only)
Timer0 output bit

bit 4 **T016BIT:** Timer0 Operating as 16-bit Timer Select bit
1 = Timer0 is a 16-bit timer
0 = Timer0 is an 8-bit timer

bit 3-0 **T0OUTPS<3:0>:** Timer0 output postscaler (divider) select bits
1111 = 1:16 Postscaler
1110 = 1:15 Postscaler
1101 = 1:14 Postscaler
1100 = 1:13 Postscaler
1011 = 1:12 Postscaler
1010 = 1:11 Postscaler
1001 = 1:10 Postscaler
1000 = 1:9 Postscaler
0111 = 1:8 Postscaler
0110 = 1:7 Postscaler
0101 = 1:6 Postscaler
0100 = 1:5 Postscaler
0011 = 1:4 Postscaler
0010 = 1:3 Postscaler
0001 = 1:2 Postscaler
0000 = 1:1 Postscaler

Figura 4.5 Registro T0CON0.

4.1.6 Configuración TIMER1:

El TMR1 lo hemos utilizado en conjunto con el CCP1 a forma de contador ascendente para controlar el giro del servomotor encargado de la dirección.

Para configurar el TMR1 hemos utilizado los siguientes registros que se encuentran en el datasheet:

- Registro “T1CON”:
  - Bits 5-4: Prescaler input (10b = 1:4)
  - Bit 2: Habilitación de la sincronización de la entrada (1 = no sincronizada, 0 = sincronizada).

- Bit 1: Modo de lectura/escritura del timer1(1 = 16 bits, 0 = 8 bits)
- Bit 0: Bit habilitación timer1

REGISTER 26-1: T1CON: TIMER1 CONTROL REGISTER

U-0	U-0	R/W-0/u	R/W-0/u	U-0	R/W-0/u	R/W-0/u	R/W-0/u
—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
- 11 =1:8 Prescale value
- 10 =1:4 Prescale value
- 01 =1:2 Prescale value
- 00 =1:1 Prescale value

bit 3 **Unimplemented:** Read as '0'

bit 2 **$\overline{\text{SYNC}}$:** Timer1 Synchronization Control bit

When TMR1CLK = Fosc or Fosc/4

This bit is ignored. The timer uses the internal clock and no additional synchronization is performed.

ELSE
- 0 = Synchronize external clock input with system clock
- 1 = Do not synchronize external clock input

bit 1 **RD16:** 16-bit Read/Write Mode Enable bit
- 0 = Enables register read/write of Timer1 in two 8-bit operation
- 1 = Enables register read/write of Timer1 in one 16-bit operation

bit 0 **ON:** Timer1 On bit
- 1 = Enables Timer1
- 0 = Stops Timer1 and clears Timer1 gate flip-flop

Figura 4.6 Registro T1CON.

- Registro “T1CLK”:
  - Bits 3-0: Selección de la fuente de reloj (0001b = Fosc/4)

REGISTER 26-3: T1CLK TIMER1 CLOCK SELECT REGISTER

U-0	U-0	U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	—	CS<3:0>			
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	HC = Bit is cleared by hardware

bit 7-4 **Unimplemented:** Read as ‘0’

bit 3-0 **CS<3:0>:** Timer1 Clock Select bits

1111 = Reserved
1110 = Reserved
1101 = LC4_out
1100 = LC3_out
1011 = LC2_out
1010 = LC1_out
1001 = Timer0 overflow output
1000 = CLKR output
0111 = SOSC
0110 = MFINTOSC (32 kHz)
0101 = MFINTOSC (500 kHz)
0100 = LFINTOSC
0011 = HFINTOSC
0010 = Fosc
0001 = Fosc/4
0000 = T1CKIPPS

Figura 4.7 Registro T1CLK.

4.1.7 Configuración modulo CCP1.

El Módulo CCP1 lo usamos para controlar los grados de giro del servomotor.

En este módulo solo usamos el modo comparación asociado al TMR1, para poder dar la señal correspondiente a la dirección de giro deseada.

Para mover el servomotor es necesario dar un pulso en alto de 0,5ms a 2,5ms, y según el ancho de pulso que le demos tendrá una posición concreta, de 0º (0,5ms) a 180º (2,5ms). Al iniciarlo, le daremos unos valores iniciales para que el servo empiece en una posición de 90º. Para configurarlo, tenemos que dar un valor al registro CCP1CON:

- Bit 7: enable del CCP1 (1 = habilitado)
- Bit 3-0: seleccionamos el modo de funcionamiento del CCP1 (1001 = modo comparación poniendo a 0 la salida)

REGISTER 28-1: CCPxCON: CCPx CONTROL REGISTER

R/W-0/0	U-0	R-x	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN	—	OUT	FMT	MODE<3:0>			
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Reset
'1' = Bit is set	'0' = Bit is cleared	

bit 7 **EN:** CCPx Module Enable bit
1 = CCPx is enabled
0 = CCPx is disabled

bit 6 **Unimplemented:** Read as '0'

bit 5 **OUT:** CCPx Output Data bit (read-only)

bit 4 **FMT:** CCPW (Pulse Width) Alignment bit
MODE = Capture mode
Unused
MODE = Compare mode
Unused
MODE = PWM mode
1 = Left-aligned format
0 = Right-aligned format

bit 3-0 **MODE<3:0>:** CCPx Mode Select bits(1)
1111 - 1100 = PWM mode (Timer2 as the timer source)
1110 = Reserved
1101 =Reserved
1100 = Reserved

1011 =Compare mode: output will pulse 0-1-0; Clears TMR1
1010 =Compare mode: output will pulse 0-1-0
1001 =Compare mode: clear output on compare match
1000 =Compare mode: set output on compare match

0111 =Capture mode: every 16th rising edge of CCPx input
0110 =Capture mode: every 4th rising edge of CCPx input
0101 =Capture mode: every rising edge of CCPx input
0100 =Capture mode: every falling edge of CCPx input

0011 =Capture mode: every edge of CCPx input
0010 =Compare mode: toggle output on match
0001 =Compare mode: toggle output on match; clear TMR1
0000 =Capture/Compare/PWM off (resets CCPx module)

Figura 4.8 Registro CCPxCON.

4.1.8 Configuración de las interrupciones

4.1.8.1 Configuración Interrupción del TMRO

Para configurar las interrupciones del TMR0 tenemos que seguir el siguiente orden de acciones:

- Accedemos al banco 14
- Borramos la bandera (PIR0,5)
- Habilitamos la máscara (PIE0,5)
- Accedemos al banco 11
- Habilitar la cuenta en timer0 (T0CON0,7)
- Volvemos al banco 0

- Volvemos al programa principal (return)

4.1.8.2 Configuración Interrupción del CCP1

Para configurar las interrupciones del CCP1 tenemos que seguir el siguiente orden de acciones:

- Accedemos al banco 14
- Borramos la bandera de interrupción (CCP1IF)
- Habilitamos la máscara (CCP1IE)
- Volvemos al banco 0
- Volvemos al programa principal (return)

4.1.8.3 Configuración Interrupción del Periféricos

Para configurar las interrupciones de los periféricos tenemos que indicarle que pines queremos habilitar para la interrupción. En nuestro caso usaremos los pines RA0 y RA1.
En el registro “IOCAP” usando el comando “iorwf”, le metemos un byte en el que las posiciones del mismo, en nuestro caso la posición 0 y 1, estén a 1 (00000011B). Despues de haber configurado que pines queremos habilitar, seguimos este orden de acciones:

- Accedemos al banco 14
- Habilitamos las interrupciones de los pines (IOCIE)
- Accedemos al banco 0
- Habilitamos las interrupciones de periféricos (PEIE)

Despues de haber configurado todo lo anterior, tenemos que habilitar las interrupciones globales activando “GIE”.

4.2 SUBRUTINAS

4.2.1 Bucle principal

- Inicialmente en el bucle principal comprobamos el sentido en el que tiene que ir el robot, controlado por el bit “DIRECCION”, de tal forma que:
  - Valor igual a 0, se activa el pin “ENABLE” y el “INPUT1” del puente en H, haciendo que el servomotor trucado inicie la marcha hacia delante.
  - Valor igual a 1 se procede a comprobar el estado del bit “BIGOTES”, que representa si ha existido una colisión.
- Se procede a comprobar el estado de “BIGOTES:

- Si el valor es igual a 0, no ha existido ninguna colisión y procede a comprobar el estado del bit “LUZ”.
- Si el valor es igual a 1, paramos la marcha del robot (INPUT1 = 0) y posteriormente accedemos a “rutina_esquivar”.

- En la “rutina_esquivar” se hace una doble comprobación de seguridad para verificar el ángulo de las ruedas, posteriormente se enciende el LED trasero (LED_ATRAS = 1) y la marcha atrás (INPUT2 = 1). Que despues de un tiempo establecido sigue el flujo del bucle yendo a testear el bit “LUZ”
- Por último, comprobamos el bit “LUZ” que si es igual a retorno al bucle y si es igual a 1 comprobamos el valor de la LDR:
  - Si el valor de “ZERO “es igual a 0, se apaga.
  - Si el valor de “ZERO” es igual a 1 se enciende.

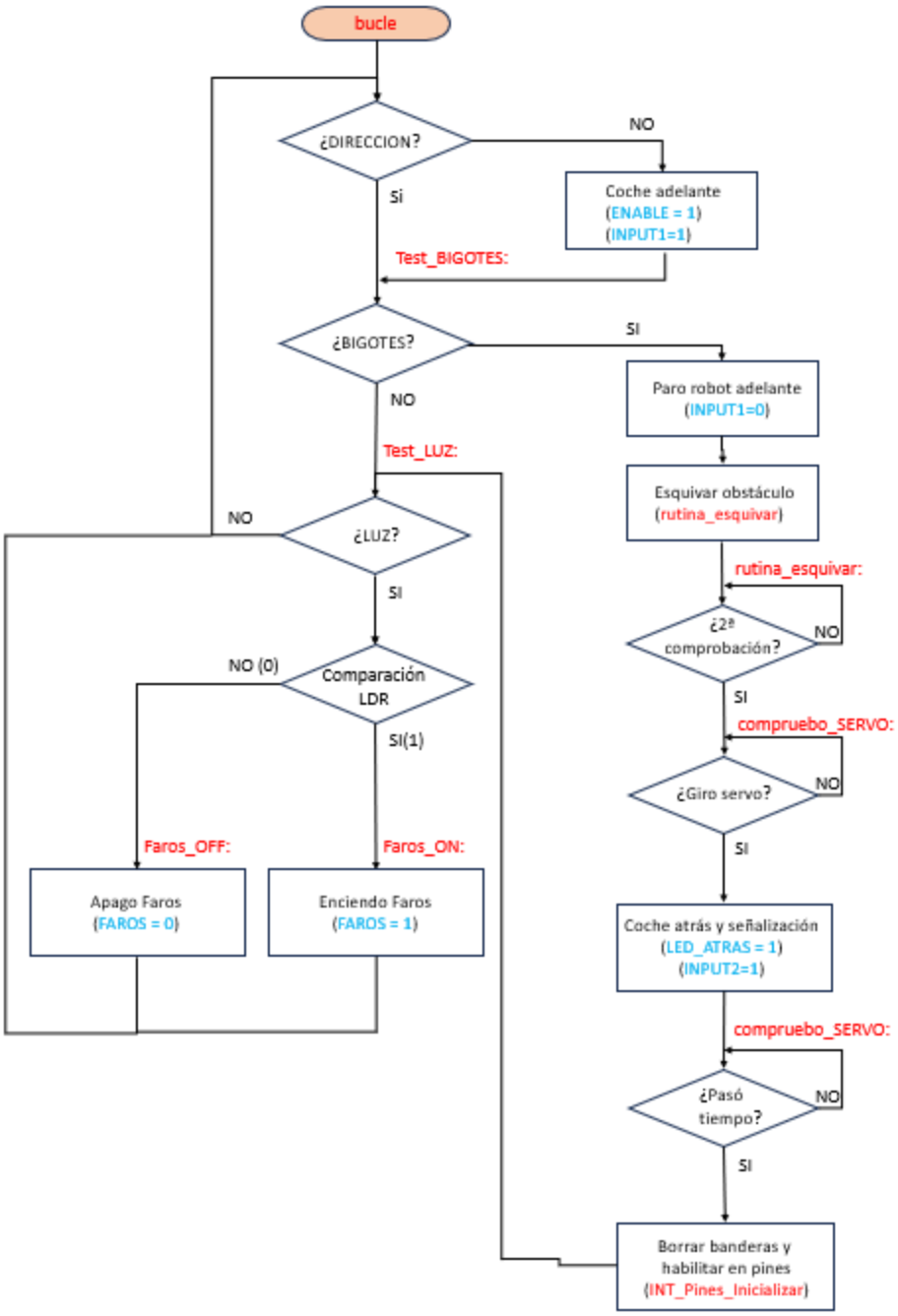

Figura 4.9 Diagrama de bloque del bucle principal.

4.2.2 Rutina servicio interrupciones globales

Interrupciones globales:

- Comprueba la bandera del TMR0, si está en 1 se activa la rutina de servicio de interrupción del TMR0 y si está en 0 pasa a la siguiente instrucción.
- Comprueba la bandera del CCP1 (modulo captura comparación PWM), si está en 1 se activa la rutina de servicio de interrupción del CCP1 y si está en 0 pasa a la siguiente instrucción.
- Comprueba la bandera del IOC (interrupción de periféricos), si está en 1 se activa la detección de la dirección de choque y vuelve a la primera instrucción y si está en 0 vuelve a la primera instrucción.

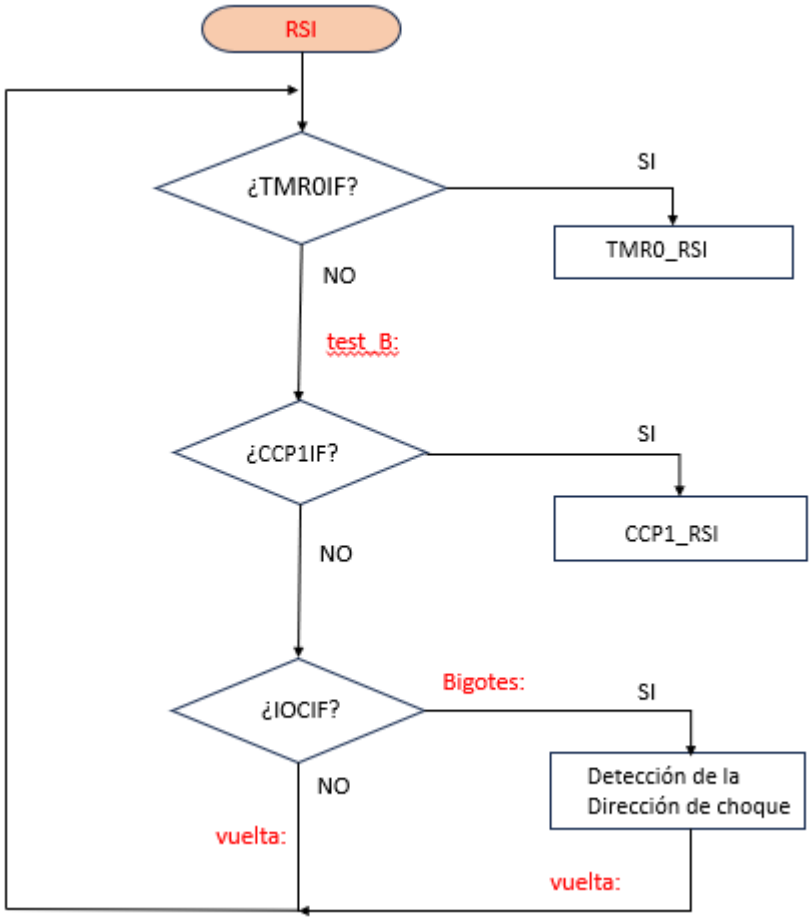

Figura 4.10 Diagrama de bloque de la rutina de servicio de interrupciones globales.

4.2.2.1 Rutina interrupción modulo TMR0

Rutina de servicio de interrupción del TMR0:

- Recarga el TMR0 y pasa a la siguiente instrucción.
- Activa el Gestor de tareas del TMR0 y pasa a la siguiente instrucción.
- Comprueba dirección:
    - Si dirección está a 1: Comprueba bigote izquierdo.
        - Si bigote izquierdo está a 1: Incrementa unidades y comprueba Status, que es la resta del valor 18 y unidades.
            - Si Status está a 1: Inicializa TMR1 y carga en el módulo CCP1 (IZQH→DC_PWM1H; IZQL→DC_PWM1L)
            - Si Statu está a 0: Si está a 0: Pone a 0 unidades y activa LED_ATRÁS_CONMUTAR. Inicializa TMR1 y carga en el módulo CCP1 (IZQH→DC_PWM1H; IZQL→DC_PWM1L).

        - Si bigote izquierdo está a 0: Incrementa unidades y comprueba Status, que es la resta del valor 18 y unidades.
            - Si Status está a 1: Inicializa TMR1 y carga en el módulo CCP1 (DERH→DC_PWM1H; DERL→DC_PWM1L)
            - Si Statu está a 0: Si está a 0: Pone a 0 unidades y activa LED_ATRÁS_CONMUTAR. Inicializa TMR1 y carga en el módulo CCP1 (DERH→DC_PWM1H; DERL→DC_PWM1L).
    - Si dirección está a 0: Inicializa TMR1 y carga en el módulo CCP1 (RECTOH→DC_PWM1H; RECTOL→DC_PWM1L).

- Después de todos los casos explicados en el punto 3, realiza las siguientes acciones:

    - Pone la variable SERVO en 1
    - Activa CCP1_Escritura
    - Pone la variable contador_Veces en 1
    - Pone la variable LUZ en 1
    - Activa Lectura_LDR

    Y vuelve al bucle principal

- Rutina de servicio de interrupción del CCP1: Pone la bandera del CCP1 y el servo a 0 y vuelve al bucle principal

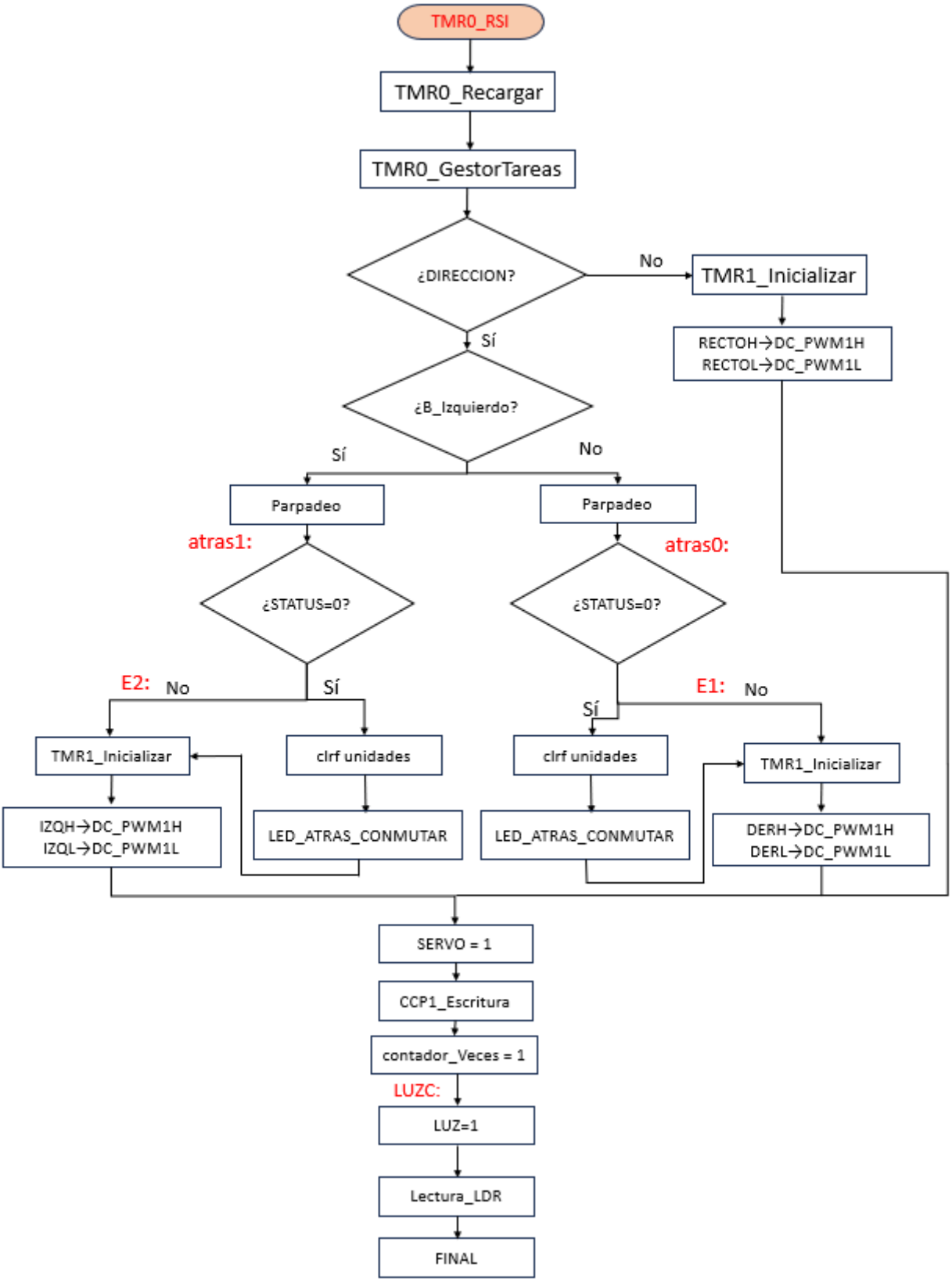

Figura 4.11 Diagrama de bloque de la rutina de interrupción del módulo TMR0.

4.2.2.2 Rutina interrupción modulo CCP1

Rutina de servicio de interrupción del CCP1 (modulo captura comparación PWM): Pone la bandera del CCP1 y el servo a 0 y vuelve al bucle principal.

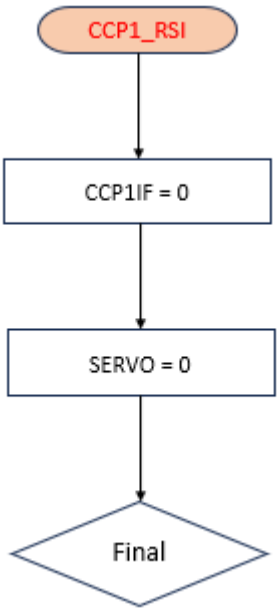

Figura 4.12 Diagrama de bloque de la rutina de interrupción del módulo CCP1.

4.2.2.3 Rutina interrupción de periféricos

IOC (interrupción de periféricos)

- Realiza las siguientes acciones:
  - Pone a 0 las interrupciones de los periféricos
  - Pone a 0 la bandera
  - Pasa la información del byte donde arroja que periférico se ha activado a w.
  - Pone a 1 la variable BIGOTES
  - Pasa la información de w al byte B_Dirección.
- Comprueba el bit 0 de B_Dirección:
  - Si está a 1: pone la variable DIRECCION a 1 y vuelve al bucle principal.
  - Si está a 0: Comprueba el bit 1 B_Direccion:
    - Si está a 1: DIRECCION=1 y vuelve al bucle principal.
    - Si está a 0: Vuelve al bucle principal.

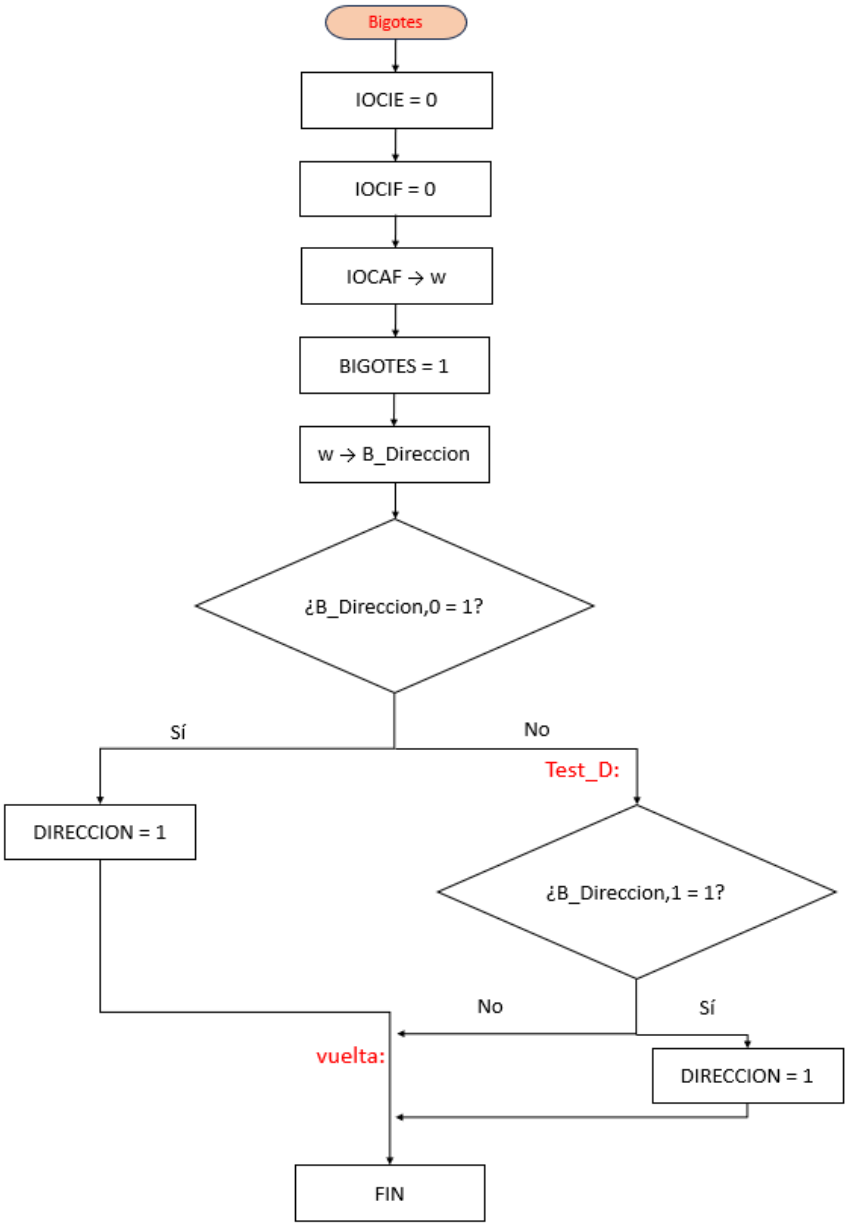

Figura 4.13 Diagrama de bloque de la rutina de interrupción de periféricos.

4.3 MACROS

Se han usado cuatro macros en el proyecto final:

- Para el cambio de banco necesario en las distintas instrucciones (movlb).
- Para cargar valores enteros directamente en registros (movlw y movwf).
- Para conmutar el led indicador de marcha atrás, usando el comando xorwf e indicando el pin.
- Para controlar el tiempo en que se debe conmutar el led indicador de marcha atrás.

5 PRESUPUESTO

Finalmente se ha creado una tabla con todos los materiales utilizados, tanto los que nos ha facilitado la universidad como los que poseíamos nosotros mismos y los que hemos tenido que comprar.

Tabla 5.1 Hoja de presupuestos.

Material	Proveedor	Cantidad	Precio unitario	Precio total
PIC16F15376 CURIOSITY NANO	Farnell Components S.L.	1	22,91 €	22,91 €
MICROSERVOMOTOR MG90S	TECNITRON	2	3,21 €	6,42 €
CONTROLADOR L28N, DOBLE PUENTE EN H	AMAZON	1	3,99 €	3,99 €
RUEDAS	*1	3	0,00 €	0,00 €
FINALES DE CARRERA	Farnell Components S.L.	2	0,48 €	0,96 €
ZÓCALO DE 40 VIAS	Farnell Components S.L.	1	0,34 €	0,34 €
PORTAPILAS DE 3 PILAS	TECNITRON	1	3,35 €	3,35 €
BOBINA PLA, IMPRESIONES 3D	AMAZON	1	16,00 €	16,00 €
PILAS 1.5V	TECNITRON	3	0,70 €	2,10 €
LED ROJO	*1	1	0,00 €	0,00 €
LED AZUL	*1	2	0,00 €	0,00 €
LDR	*2	1	0,00 €	0,00 €
RESISTENCIAS	*1	7	0,00 €	0,00 €
CONDENSADORES	*1	4	0,00 €	0,00 €
BORNEROS	*1	13	0,00 €	0,00 €
INTERRUPTOR	*2	1	0,00 €	0,00 €
PEGAMENTO	EL CORTE CHINO	1	2,50 €	2,50 €
CABLE DE DATOS	EL CORTE CHINO	1	3,50 €	3,50 €
TORNILLOS Y TUERCAS	FERRETERIA VALDEPA	1	0,40 €	0,40 €
ALAMBRE DE N.º 4	FERRETERIA VALDEPA	1	0,70 €	0,70 €
RODAMIENTOS	*2	2	0,00 €	0,00 €
CABLES DE CONEXION Y ESTAÑO	*1	1	0,00 €	0,00 €
			TOTAL	**63,16 €**