“DISEÑO DE UN SISTEMA AUTOMÁTICO DE RIEGO”.

Pedro Quintana Benítez

Trabajo Fin de Grado
Ingeniería Radioelectrónica
Tutor: Prof. Dr. Carlos Corrales Alba

Escuela de Ingenierías Marina, Náutica y Radioelectrónica

Septiembre 2018

ÍNDICE DE CONTENIDOS

1. RESUMEN

“Diseño de un vehículo autómata para el regadío”.

Resumen

A través de este proyecto se pretende diseñar un vehículo autómata capaz de facilitar las tareas de riego en cultivos de grandes dimensiones, de tal manera que la presencia de personal sea meramente optativa, utilizando para ello única y exclusivamente material y conocimientos adquiridos durante la duración de este Grado.

En el documento que se desarrolla en las siguientes páginas, se explica detalladamente el proceso seguido para la elaboración de este prototipo, tanto la programación paso a paso, como los elementos utilizados y el porqué de todo ello, así como los problemas surgidos durante el proyecto y las soluciones que se han llevado a cabo.

“Design of an automatic vehicle for irrigation”.

Abstract

The aim of this project is to design an automatic capable of facilitating irrigation task in large-scale crops, in such a way that the presence of workers is merely optional, using only material and knowledge acquired during the duration of this degree.

In the document that develops in the following pages, the process followed for the elaboration of this prototype is explained in detal, as much the programming step by step, as the elements used and the reason of all this, as well as the problems arisen during the project and the solutions that have been carried out.

2. INTRODUCCIÓN A LOS SISTEMAS DE RIEGO MÓVILES

Desde finales del siglo XIX este tipo de vehículos han sido construidos y mejorados en función de las nuevas tecnologías que se han ido desarrollando. En su primera versión, el ingeniero francés Jhon Winebrenner patentó lo que sería la versión sobre la que se desarrollarían todos los vehículos de estas características, un sistema de traslado de tuberías a caballo.

A medida que se vio su utilidad en el ámbito agrario, diferentes personas y entidades se interesaron por su desarrollo, comprando y vendiendo entre unos y otros la patente del sistema, hasta que acabó en manos de Robert Daugherty, president de Valley Manufacturing Company, una pequeña fábrica de maquinaria que comenzó a comercializarlos. A mediados del siglo XX su popularidad se disparó, coincidiendo con la caducidad de la patente del vehículo, dando lugar a la entrada al negocio de otros fabricantes.

Dado al gran interés que levantó en todo el mundo este vehículo (ya que proporciona grandes ahorros económicos, mayor rendimiento de cultivos, mayor eficiencia en el uso del agua y ahorro en energía, así como la capacidad de regar grandes extensiones de terreno sin dificultad) ha dado un gran salto en estos últimos años en cuanto a la tecnología que utiliza.

Actualmente, estos sistemas tienen los equipos más avanzados del sector (GPS, control por smartphone, monitorización remota a través de distintos software, sensores de obstáculos, capacidad de programación intuitiva en el mismo vehículo...) para ofrecer a sus clientes la máxima eficiencia en sus particulares requisitos, pudiéndose adaptar a cualquier tipo de terreno y cultivo.

Así mismo, se han desarrollado diferentes estructuras para este tipo de vehículos, para cada tipo de servicio que se quiera realizar o para adecuarse correctamente al terreno sobre el que se va a trabajar. Cada vehículo lleva instalado en su estructura los diferentes elementos que necesita para poder llevar a cabo su tarea correctamente (mangueras, aspersores de diferentes tipos, contactores para evitar el desvío del vehículo, etc.). Esto hace que cada clase de estructura realice su tarea con un desplazamiento terminado, una cantidad de flujo de agua o fertilizante determinado (en función de lo programado por el usuario y por elementos de riego que utilice), un sistema para evitar que se desvíe del camino programado o si se desplaza trasversalmente a lo largo de todo el terreno o de manera circular girando en torno a un pívot central, que es el que proporciona tanto energía eléctrica como agua.

3. OBJETIVOS

El objetivo de este proyecto no consiste en mejorar las características ya existentes de estos vehículos, sino que se pretende desarrollar desde cero un equipo semejante con los conocimientos adquiridos durante este grado, utilizando los recursos básicos de los que se dispone en los laboratorios y adquiriendo por cuenta propia otros cuantos.

Por lo que se va a construir un vehículo (estación móvil) y una torre de control (estación fija) que van a simular el funcionamiento de estos vehículos. Se le añadirán una serie de elementos de control a la parte fija para poder actuar sobre la parte móvil a distancia.

El vehículo tendrá la capacidad de moverse de manera autónoma y de manera manual actuando sobre la torre de control, desde donde se controlarán todos los parámetros (velocidad, dirección, riego…).

4. METODOLOGÍA

En primer lugar, como en todo proyecto, se dejaron claros los objetivos a los que se quería llegar y lo que se necesitaría para ello.

La primera función que se debe conseguir para el vehículo a construir es la del movimiento más básico, hacia delante y hacia atrás, construyendo para ello un prototipo de vehículo antes del que va a simular los vehículos sobre los que se ha hablado, así como un sistema que controle la velocidad a la que se mueve el vehículo, ya que es muy importante controlar la velocidad adecuadamente en este tipo de servicios.

Una vez comprobado que el prototipo realiza el desplazamiento correctamente (manual), el siguiente paso es conseguir que se desplace transmitiéndole esa orden a través de la torre de control.

En el momento que se consiga la buena comunicación entre las dos estaciones, se procederá a conseguir que el vehículo sea capaz de realizar movimientos omnidireccionales para conseguir una mayor movilidad sobre el terreno, controlado todo ya a través de la torre de control.

En el momento que se consiga tener prácticamente acabado el movimiento manual del vehículo, se procederá a programar el movimiento autónomo de este, para que en el momento que se logre y se confirme que funciona todo correctamente, se proceda a su montaje en un vehículo que imita la forma y tamaño a escala de esta clase de vehículos, para proceder a ajustar y arreglar los posibles fallos/errores que se den una vez finalizado el montaje en dicha estructura.

Por supuesto, esto es la planificación inicial del proyecto, a medida que se vaya realizando surgirán imprevistos que tendremos que solucionar sobre la marcha, complicaciones con distintos elementos que utilicemos o simplemente fallos en el diseño inicial del vehículo. Todo ello se explicará a medida que se avance en la lectura de este documento.

4.1 HARDWARE

Antes de comenzar a desarrollar la evolución del proyecto, se va a explicar en detalle cada dispositivo y elemento de control elegido para la fabricación y diseño de este prototipo de vehículo, así como sus características técnicas y funcionamiento, para que, en el momento de la explicación técnica del desarrollo del proyecto, se entienda en que consiste el dispositivo o elemento sobre el que se habla.

Como bien indica el título, comenzaremos por el Hardware:

4.1.1 MICROCONTROLADOR

A lo largo del proyecto, se ha ido variando el tipo de microcontrolador que se ha utilizado, comenzando por el 16F886 y variando al 18F6620 según se han visto las necesidades requeridas.

Antes de empezar a hablar de cada microcontrolador utilizado y sus características técnicas, sería conveniente conocer a que se le llama ``microcontrolador´´.

Un microcontrolador es un circuito integrado programable, capaz de ejecutar ordenes grabadas en su memoria, que, a diferencia de un microprocesador, tiene integrado todos los componentes necesarios para la comunicación con los dispositivos periféricos que se le conectan.

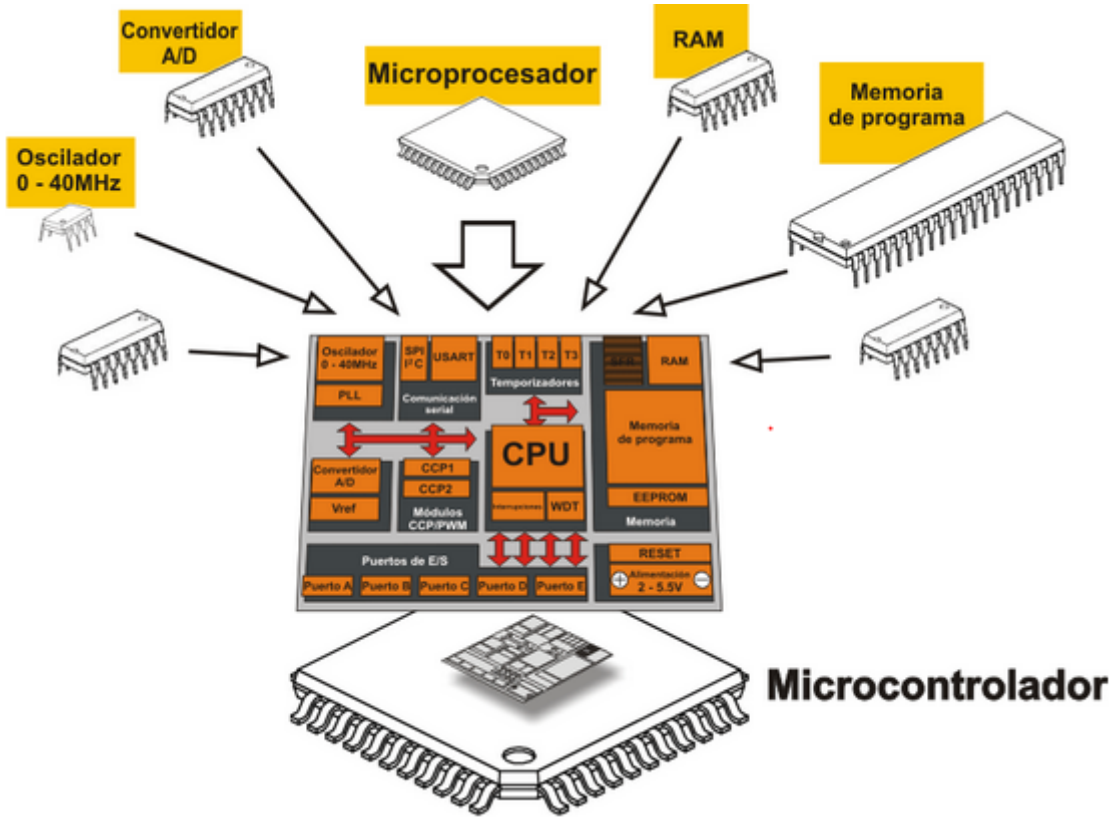


Fig 1. Estructura de un Microcontrolador

En este proyecto, se han utilizado dos tipos diferentes de microcontroladores:

- ***16F886:*** Este PIC, fabricado por *Microchip*, fue la primera elección a la hora de desarrollar este proyecto, pero a la hora de comenzar con la programación del mismo y al ver que la memoria de este se iba a quedar muy corta, se decidió cambiar el PIC a uno con más memoria (18F2620).

  Este PIC consta de 24 I/O de las cuales 11 se pueden utilizar como entradas A/D. En cuanto a lo que memoria de programa se refiere, tiene una capacidad de 8192 pabras de memoria Flash, una memoria RAM de 368 bytes y una E^2PROM de 256 bytes. También consta de 3 temporizadores internos y tiene capacidad de conexión USART.

  Es un PIC muy decente para programaciones básicas, pero cuando el programa tine un tamaño considerable, es necesario cambiar a otra familia de PIC's, que en este caso en particular, ha sido el PIC 18F2620.

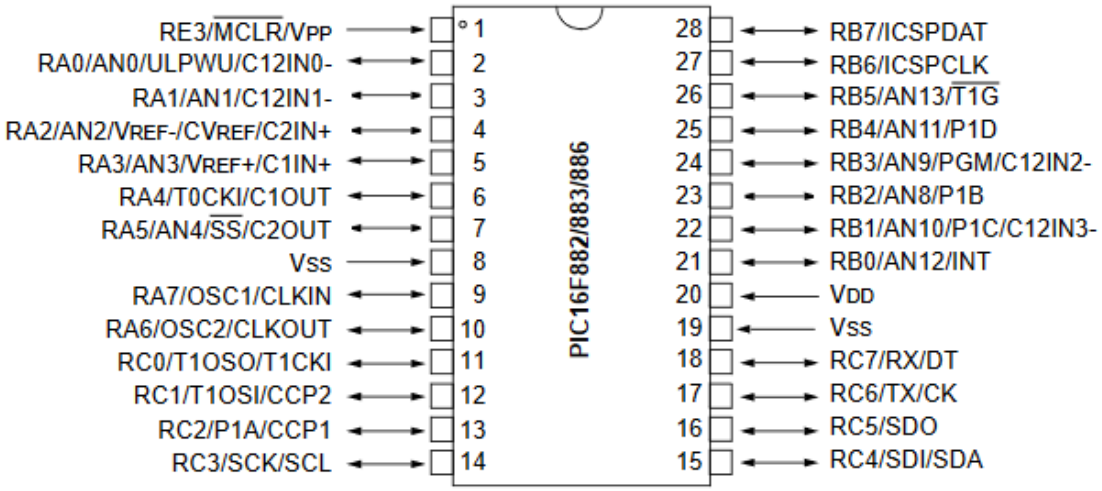


Fig 2. Pines de conexión de un PIC 16F886

- ***18F4620:*** Este PIC, fabricado también por *Microchip,* se eligió por la enorme memória que poseía con respecto al 16F886 y su mayor número de I/O.

  En este caso, el PIC poseé 40 Pines, de los cuales 36 son I/O y de ellas 13 tienen la capacidad de funcionar como entrada A/D. Respecto a la memoria, tiene una memoria Flash de 32768 palabras, una memoria RAM de 4KB y una E^2PROM de 1KB. Consta de 4 temporizadores internos y capacidad de conexión USART.

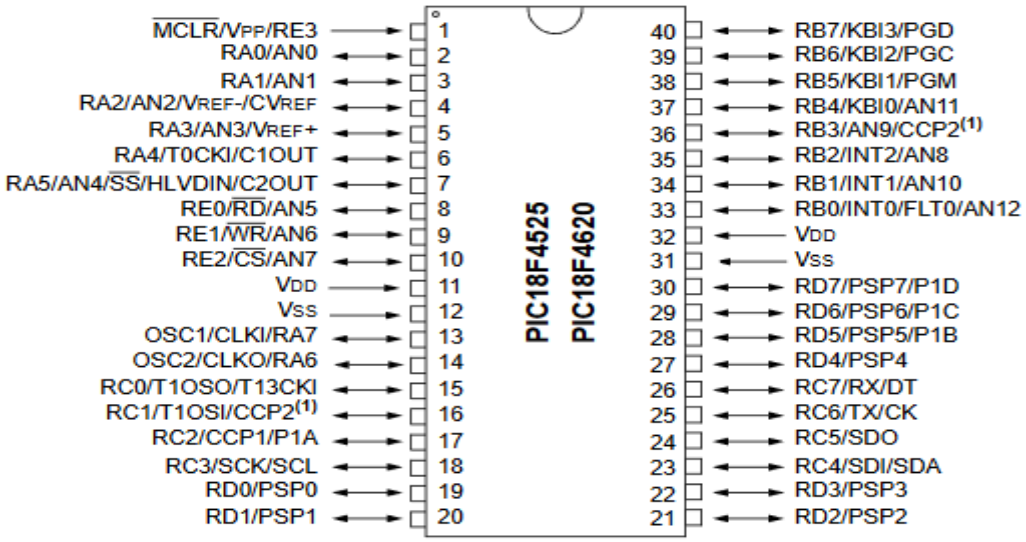


Fig 3. Pines de conexión de un PIC 18F4620

4.1.2 USB PIC'SCHOOL

El USB PIC'School, es una herramienta para el aprendizaje y diseño de aplicaciones basadas en microcontroladores PIC de Arizona Microchip.

Permite trabajar con los dispositivos PIC más relevantes de las familias 12F, 16F y 18F. Con este equipo puedes desarrollar proyectos complejos tanto a nivel de software como de hardware. Dispone de un buen número de los periféricos más utilizados en las aplicaciones reales, y una protoboard para el montaje sin soldadura del hardware necesario en un determinado proyecto. La conexión de este equipo al PC se realiza a través de un puerto USB estándar.

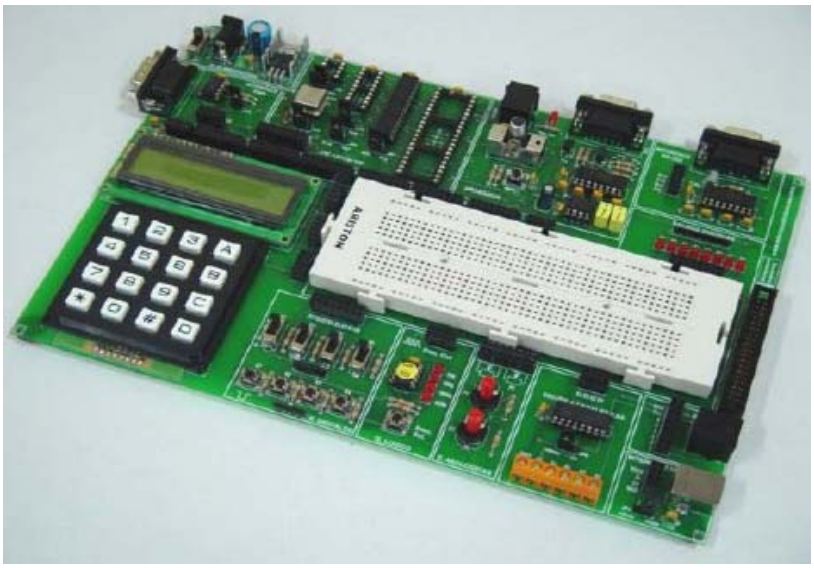


Fig 4. Entrenador PIC'SCHOOL

4.1.3 TRANSCEPTOR NRF24L01

El NRF24L01 es un dispositivo de radiofrecuencia, fabricado por *Nordic Semiconductor* que permite tanto enviar como recibir información de manera simultánea (transceptor).

Este dispositivo trabaja en la banda de frencuancias comprendida entre 2.4GHz y 2.5GHz, la cual es para uso gratuito. La velocidad de transmisión es configurable entre 250Kbps, 1Mbps, y 2Mbps y permite la conexión simultánea con hasta 6 dispositivos.

La banda de frecuencia es de 2400 a 2525 MHz, pudiendo elegir entre 125 canales espaciados a razón de 1MHz. Para evitar problemas de interferencias con las redes Wifi es recomendable utilizar las frecuencias comprendidas entre 2501 a 2525 MHz.

La tensión de alimentación del NRF24L01 es de 1.9 a 3.6V, aunque en este proyecto se ha alimentado con una tensión de 5V, ya que en las datasheet indica que se puede conectar con tal tensión.

Existen dos versiones de módulos NRF24L01, uno con antena integrada en forma de zig-zag y un alcance máximo de 20-30 metros, y la versión de alta potencia que incorpora amplificador y antena externa, que es la que se ha elegido para este proyecto, con un alcance máximo de 700-1000 metros.

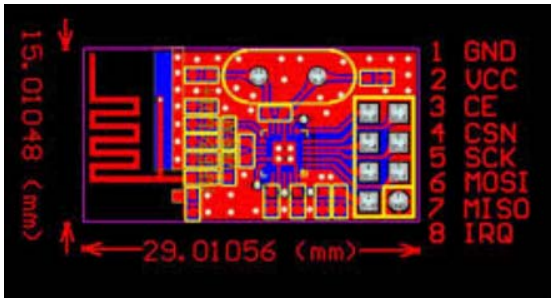


Fig 5. Conexionado NRF24L01

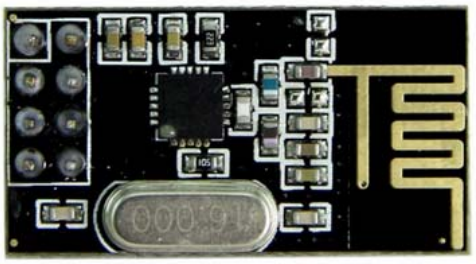


Fig 6. Imagen módulo transceptor

Pines	N°	Descripción
Vcc	1	Tensión de alimentación del módulo
Vss	2	GND
CE	3	Chip Enable
CSN	4	Chip Select del SPI (Negado)
SCK	5	Reloj del bus SPI
SDI	6	Entrada de datos del bus SPI
SDO	7	Salida de datos del bus SPI
IRQ	8	Salida interrupción (negado)

4.1.4 JOYSTICK

Un joystick es un elemento que transmite una señal de entrada para programas digitales. Este en concreto está contruido mediante un conjunto de resistencias variables.

Es una herramienta muy útil, que nos ayuda a mover en una determinada dirección a lo que queramos. El funcionamiento está basado en el movimiento en dos dimensiones de una palanca, este movimiento es captado por dos potenciometros (uno para cada dirección de movimiento, lateral o vertical) .

Este módulo de joystick cuenta con cinco pines los cuales se enumeran a continuación:

1. GND: Pin conectado a tierra.
2. +5V: pin de alimentación (5v).
3. VRx: pin de lectura del potenciómetro para el eje X
4. VRy: pin de lectura del potenciómetro para el eje Y
5. SW: es un pin adicional que se utiliza para un pulsador en la parte inferior del joystick

Fig 7. Imagen Joystick

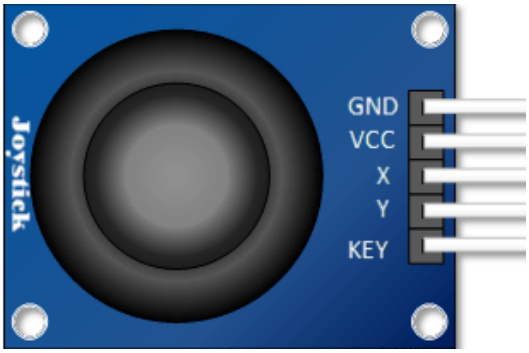


Fig 8. Pines de conexión Joystick

4.1.5 SLIDER

Este dispositivo, conocido como SLIDER, es una resistencia variable de 10kΩ, que variando la posición del deslizador, aumenta o disminuye el valor de esta de su valor mínimo, 0Ω, proporcionando resistencia nula al paso de la corriente, y 10kΩ, ofreciendo una resistencia máxima. Variando la posición del deslizador, en este proyecto se actúa sobre la velocidad a la que el vehículo se desplaza.

El potenciómetro variable consta de 2 salidas, OTA y OTB, alimentada cada una de ellas con su propio Vcc y GND. La información que transmite estas dos salidas necesita ser leída por uno de los pines del PIC que estemos utilizando, capaz de realizar la lectura Analógico/Digital, para el correcto uso de dicha información.

Fig 9. Slider

4.1.6 MOTORES

Los motores que se utilizan para este proyecto son motores sencillos de corriente continua, consumen unos 60mA cada uno y aceptan una tensión ente 3 y 6 voltios.

Se han utilizado dos motores de este tipo y un módulo Driver L298N (el cual se explicará más adelante), para poder controlar el sentido de giro de estos.

Fig 10. Motor eléctrico DC y rueda

4.1.7 SERVOS

La utilización de servos en este proyecto, por un lado, nos permite mantener una posición que indiquemos, siempre que esté dentro del rango de operación del propio dispositivo, en este caso, 180º. Por otro lado nos permite controlar la velocidad de giro, podemos hacer que antes de que se mueva a la siguiente posición espere un tiempo.

Se han elegido cuatro servos S3003 de la marca FUTUBA, con las siguientes características técnicas:

Detailed Specifications			
Control System:	+Pulse Width Control 1520usec Neutral	Current Drain (4.8V):	7.2mA/idle
Required Pulse:	3-5 Volt Peak to Peak Square Wave	Current Drain (6.0V):	8mA/idle
Operating Voltage:	4.8-6.0 Volts	Direction:	Counter Clockwise/Pulse Traveling 1520-1900usec
Operating Temperature Range:	-20 to +60 Degree C	Motor Type:	3 Pole Ferrite
Operating Speed (4.8V):	0.23sec/60 degrees at no load	Potentiometer Drive:	Indirect Drive
Operating Speed (6.0V):	0.19sec/60 degrees at no load	Bearing Type:	Plastic Bearing
Stall Torque (4.8V):	44 oz/in. (3.2kg.cm)	Gear Type:	All Nylon Gears
Stall Torque (6.0V):	56.8 oz/in. (4.1kg.cm)	Connector Wire Length:	12"
Operating Angle:	45 Deg. one side pulse traveling 400usec	Dimensions:	1.6" x 0.8"x 1.4" (41 x 20 x 36mm)
360 Modifiable:	Yes	Weight:	1.3oz. (37.2g)

TABLA 1. Especificaciones servomotor

La elección de este servo se ha basado en la fuerza de torque, ya que con una tensión de trabajo baja (4.8-6.0V) son capaces de girar las ruedas del vehículo 180º con gran facilidad, al contrario que los que se utilizaron al principio (servo SG90), que con una tensión de funcionamiento de 5V, el torque es de 1.8kg/cm.

4.1.8 PANTALLA LCD LMB204BFC

Para visualizar la tarea que se está ejecutando, el estado de los interruptores de la estación fija y los diferentes valores que se decidan mostrar en ella, se ha utilizado una pantalla LCD de 4x20 (cuatro filas de 20 caracteres cada una). La tensión de operación de esta pantalla en concreto, es de 4.7-5.3V.

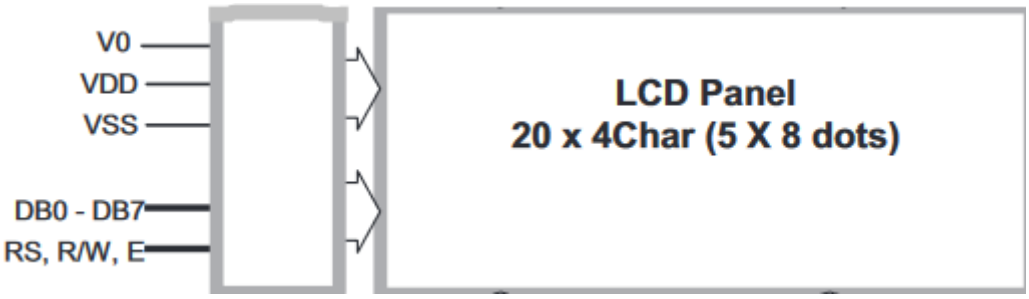


Fig 10. Conexiones pantalla LCD

V0	LCD Contrast Reference SUpply
VSS	Ground
VDD	Positive Power Supply
DB0	Bi-directional Tri-state Data Bus
..	
DB7	
R/W	Read/Write Control Bus R/W= HIGH → Read Mode Selected R/W=LOW →Write Mode Selected
E	Data Enable
RS	Register Select RS = HIGH: Transferring Display Data RS = LOW: Transferring Instruction Data

TABLA 2. Pines de conexión pantalla LCD

4.1.9 Módulo Controlador de Motores L298N

Este módulo basado en el chip L298N te permite controlar dos motores de corriente continua o un motor paso a paso bipolar de hasta 2 amperios.

El módulo cuenta con todos los componentes necesarios para funcionar sin necesidad de elementos adicionales, entre ellos diodos de protección y un regulador LM7805 que suministra 5V a la parte lógica del integrado L298N. Cuenta con jumpers de selección para habilitar cada una de las salidas del ódulo (A y B). La salida A está conformada por OUT1 y OUT2 y la salida B por OUT3 y OUT4. Los pines de habilitación son ENA y ENB respectivamente.

En la parte inferior se encuentran los pines de control del módulo, marcados como IN1, IN2, IN3 e IN4.

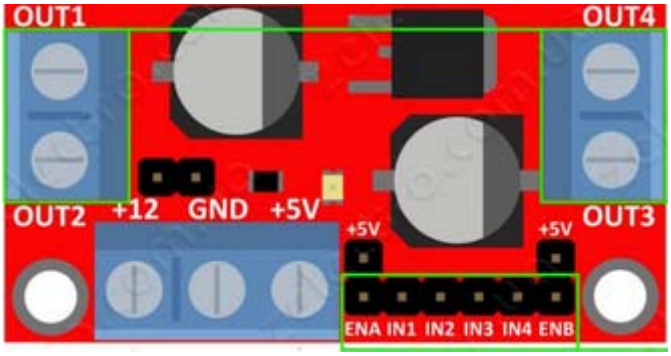


Fig 12. Módulo L298N

Este módulo se puede alimentar de 2 maneras gracias al regulador integrado LM7805.

Cuando el jumper de selección de 5V se encuentra activo, el módulo permite una alimentación de entre 6V a 12V CC. Como el regulador se encuentra activo, el pin marcado como +5V tendrá un voltaje de 5V CC. Este voltaje se puede usar para alimentar la parte de control del módulo ya sea un microcontrolador o un Arduino.

Cuando el jumper de selección de 5V se encuentra inactivo, el módulo permite una alimentación de entre 12V a 35V CC. Como el regulador no está funcionando, tendremos que conectar el pin de +5V a una tensión de 5V para alimentar la parte lógica del L298N. Usualmente esta tensión es la misma de la parte de control, ya sea un microcontrolador o Arduino.

4.1.10 SENSORES DE REFLEXIÓN CNY70

El CNY70 es un sensor óptico infrarrojo de corto alcance. Tiene una construcción compacta donde la fuente de emisión de luz y el detector están dispuestos en la misma dirección para detectar la presencia de un objeto utilizando el haz de IR de reflexión en el objeto.

Su uso más común es para construir robots seguidores de líneas. Contiene un emisor de radiación infrarroja (fotodiodo) y un receptor (fototransistor). El fotodiodo emite un haz de radiación infrarroja, el fototransistor recibe ese haz de luz cuando se refleja sobre alguna superficie u objeto. Dos sensores de este tipo se encuentran integrados en un módulo S-110 desarrollado por la empresa MSE (Fig. 14). En este proyecto se han empleado dos módulos de este tipo.

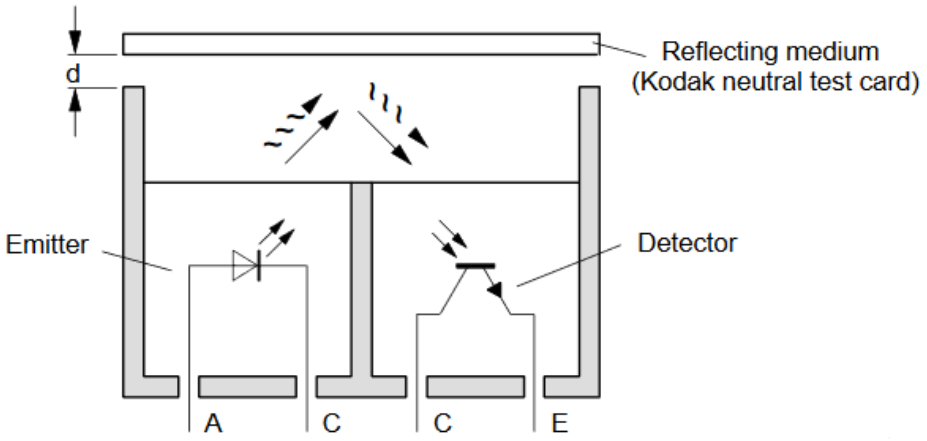


Fig 13. Funcionamiento sensor de reflexión

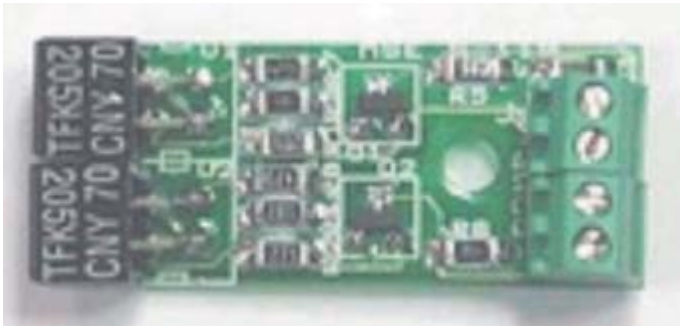


Fig 14. Módulo S-110

4.1.11 MÓDULO LM2596

El módulo LM2596 permite tener un voltaje regulado a partir de una fuente de alimentación con un voltaje mayor (en el caso de este proyecto la fuente de alimentación utilizada es de 12V DC, y se convierte a una tensión de 5V DC para alimentar la protoboard).

El módulo tinene la capacidad de obtener un voltaje de salida ajustable, gracias a la posibilidad de actuar sobre un potenciometro que regula el valor de la resistencia por la que pasa la tensión antes de llegar a la salida del dispositivo.

Ofrece una tensión de salida entre 1,5 y 35V DC (Ajustable), para una tensión de entrada de 4.5-40V DC. Soporta una corriente de salida máxima de 3 A.

Fig 15. Regulador de tensión LM2596

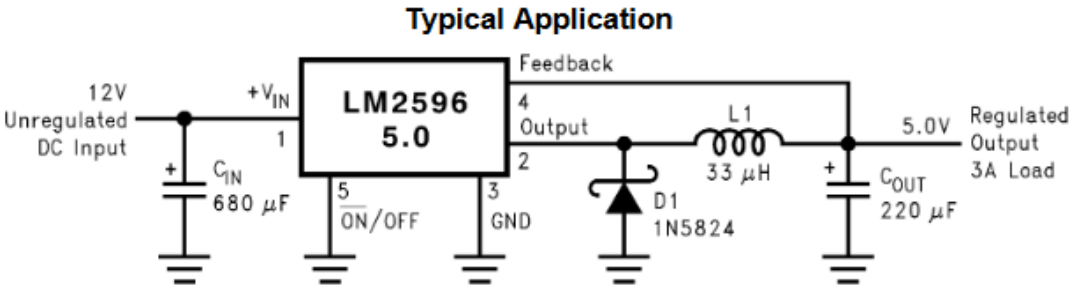


Fig 16. Circuito módulo regulador

4.1.12 FUENTE DE ALIMENTACIÓN

La fuente de alimentación utilizada para este proyecto es una batería ión-lítio recargable de corriente continua (DC 12680) de 12V con una capacidad de carga de 3800mAh. Esta fuente de alimentación ha resultado ser perfecta para el proyecto, ya que por su tamaño y peso puede ser equipada en el vehículo, alimentando perfectamente todos los dispositivos equipados.

Fig 17. Fuente de alimentación 12V

4.1.13 PROGRAMADOR VELLEMAN

Debido a el número de Pines que poseé el PIC 18F4620 (40) y a la imposibilidad de colocar un zócalo que adaptara el tamaño de este PIC en el programador USB PIC'School, se ha tenido que utilizar un programador alternativo, en concreto la *Tarjeta de Programación PIC K8076,* de la empresa *Velleman.*

Fig 18. Programador Valleman

Esta tarjeta de programación se conecta al ordenador a través del puerto serie en lugar de conexión USB. Una vez conectado al ordenador, se utiliza el programa Niple para hacer el programa a ejecutar y crear el archivo .HEX , el cual se compilará en el PIC a través del programa que viene asociado a la tarjeta, Programador PicProg2009. (Tanto el programa Niple como PicProg2009 se explicarán en detalle en la sección de Software).

4.2 SOFTWARE

4.2.1 NIPLE

Niple es un programa con un entorno visual e interactivo que facilita la programación de microcontroladores.

Con Niple, el usuario diseña un diagrama de flujo mediante pantallas gráficas e interactivas y el software traduce automáticamente el diagrama de flujo al correspondiente código ensamblador.

Este programa tiene una programación visual muy intuitiva, donde el usuario diseña el programa vinculando bloques que representan diferentes funciones. Así mismo, el programa incluye diferentes dispositivos ya implementados, en los cuales solo hay que decidir qué registros se utilizarán con el dispositivo utilizado y que pines del PIC controlaran tal dispositivo (pantallas LCD, motores paso a paso, servos, etc).

Hay que tener en cuenta también la gran cantidad de Microcontroladores con los que permite trabajar este Software, prácticamente permite la utilización de todos los PIC's de cada familia de Microcontroladores.

Durante la explicación del desarrollo de este proyecto, se explicará paso a paso las distintas funcionalidades del programa y como se han ido utilizando para conseguir el buen funcionamiento del programa diseñado.

4.2.2 PICKIT 2 V2.61

Este software es un ensamblador de PIC's, y ha sido utilizado en este proyecto cuando se programaban los PIC's 16F886 y 18F2620. El software detecta que se ha conectado el microcontrolador (conectado en el USB PIC'School) al ordenador utilizando el conector USB.

El software traduce el archivo "asm." generado por *Niple* con el programa que se ha diseñado, a un archivo "hex." el cual es grabado en el microcontrolador.

Este ensamblador se dejó de utilizar debido a que se cambió el tipo de microcontrolador utilizado a uno de mayor tamaño (18F4620), el cual no podía ser conectado en el USB PIC'School, por lo que se tuvo que buscar otro software ensamblador para poder grabar en el microcontrolador el programa diseñado, para su posterior montaje en el vehículo.

Este ensablador tiene una interfaz muy intuitiva, tal que así:

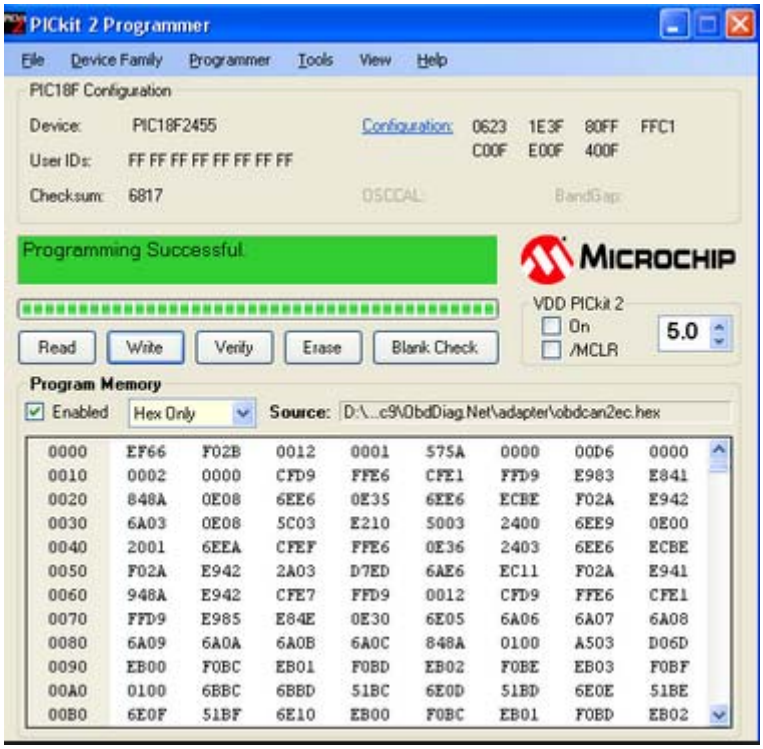


Fig 19. Software Ensamblador Microchip.

4.2.3 SOFTWARE PROGRAMADOR PICPROG 2009

Este ensamblador es el que se ha utilizado para poder programar el microcontrolador 18F4620 utilizando la *Tarjeta de Programación PIC K8076* (explicada previamente).

Poseé las mismas características básicas que el *PICkit 2* , graba en el microcontrolador seleccionado el programa previamente diseñado, guardado en un archivo “asm.” y traducido a un archivo “hex.” a la hora de ser ensablado.

Tiene varias funciones, a parte de grabar en el microcontrolador, te permite leer los datos que tenga este y guardarlos en el ordenador en un archivo “hex.” Así como borrar los datos que tenga este grabados y dejar el PIC vacío.

La interfaz de usuario es parecida a la del PICkit 2, como se muestra en la siguiente imagen:

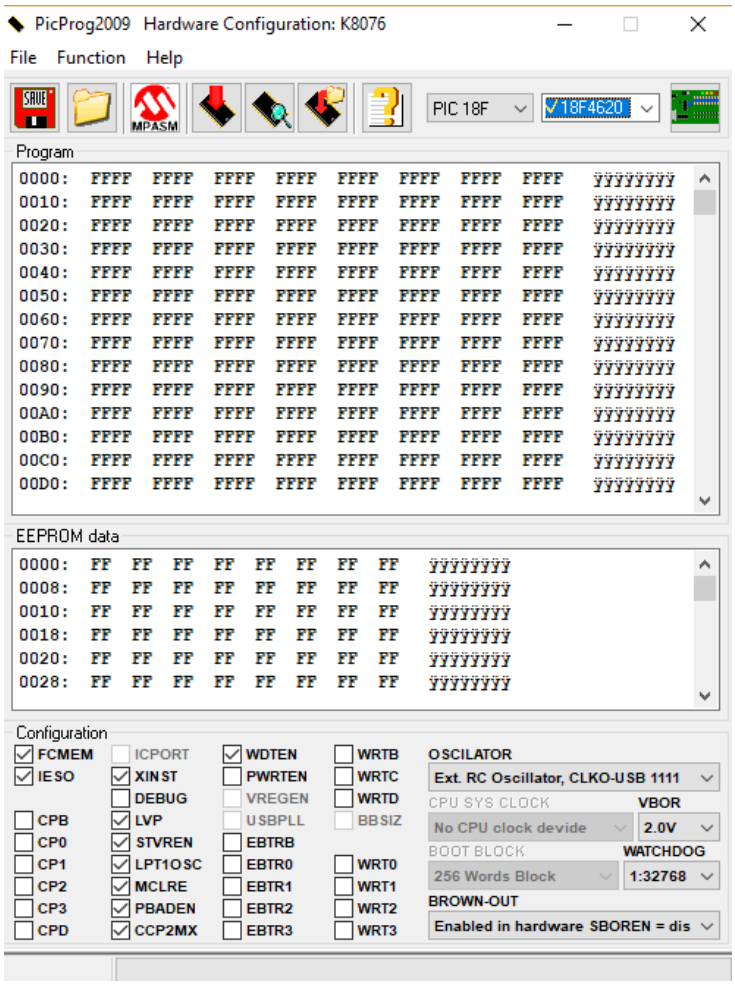
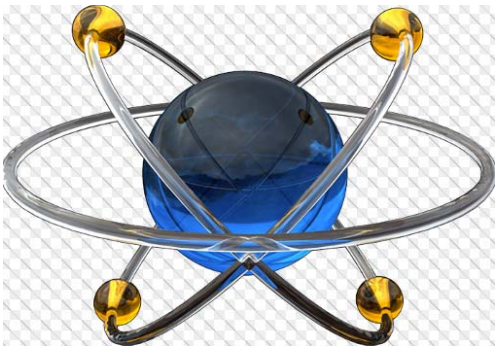


Fig 20. Software ensamblador Valleman

4.2.4 PROTEUS



Proteus es una aplicación para la ejecución de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño del esquema electrónico, programación del software, construcción de la placa de circuito impreso, simulación de todo el conjunto, depuración de errores y documentación.

En el mundo de la formación, Proteus se muestra como una herramienta magnífica porque permite al alumno realizar modificaciones tanto en el circuito como en el programa, experimentando y comprobando de forma inmediata los resultados y permitiéndole de esta forma aprender de forma práctica y sin riesgos de estropear materiales de elevado coste.

Para este proyecto, este programa se ha utilizado para el diseño de los circuitos presentados al final de este proyecto, así como el diseño de algunos de los elementos utilizados y que no venían de serie en la librería del software.

Más adelante, se explicará el proceso que se ha seguido para la elboración de estos circuitos y la gran capacidad de este programa para realizar cualquier tipo de tarea de diseño y simulación.

5. DESCRIPCIÓN DEL PROBLEMA

A la hora de la realización de este proyecto, se intenta simular el funcionamiento de los vehículos de riego conocidos como "*Pivots*" de tal manera que desempeñe el trabajo y cumpla con el diseño de estos, de forma que se asemeje lo máximo posible a ellos, teniendo en cuenta la limitación presupuestaria y que se va a realizar con los conocimientos adquiridos durante este Grado en Radioelectrónica.

5.1 COMPRENSIÓN DEL PROBLEMA

Para cumplir con el objetivo señalado en el punto anterior, se seguirán unas etapas específicas para el diseño y desarrollo del vehículo, que serán las siguientes:

1. En primer lugar, se buscará información sobre el funcionamiento de este tipo de vehículos, los tipos que hay y las particularidades de cada uno de ellos.

2. En segundo lugar, una vez estudiadas las características de estos tipos de vehículos y asumiendo las restricciónes existentes, se procederá a a decidir cuales de aquellas que se hayan visto, podrán ser desarrolladas en este proyecto, así como incluir otras nuevas para mejorar, en la medida de lo posible, el avanzado diseño que ya poseen estos vehículos.

3. En tercer lugar, se procederá a elegir los dispositivos que se utilizarán para el desarrollo de este proyecto, que ayudaran a asemejar lo máximo posible a la realidad el vehículo.

4. En cuarto lugar, se empezará con el diseño del proyecto, tanto a nivel de software como de forma física del vehículo. Se realizará paso a paso, sin saltar al siguiente hasta que no esté en perfecto funcionamiento el paso en el que se encuentre el proyecto, para seguir un orden de diseño adecuado.

5. Por último una vez realizado todo el diseño de Software y de la maqueta del vehículo, se procederá a la elaboración de los esquémas eléctricos diseñados y su presentación.

Como ya se ha dicho, el primer paso que se va a seguir para diseñar un vehículo que simule el funcionamiento de los llamados "Pivots" es el de recopilar la mayor información posible sobre ellos, para lo cual, se acude a una de las mayores empresas que se dedica al diseño, fabricación y desarrollo de estos tipos de vehículos de regadio, *VALLEY IRRIGATION* , empresa Americana lider en el sector de fabricación de estos tipos de vehículos para el regadio. En su página web (www.valleyirrigation.com) se puede encontrar toda la información necesaria para conocer los diferentes tipos de diseños que existen, según las necesidades de cada usuario, así como su funcionamiento, dispositivos que utiliza, navegación, etc.

Lo primero que hay que saber, es que tipos de vehículos de regadío existen y para qué situaciones se utiliza cada uno, así como sus características.

Se distinguen dos tipos de vehículos de diferente diseño, así mismo, dentro de cada uno de los dos grupos, estos varían en diferentes modelos que van variando desde las características más básicas, hasta las más innovadoras y modernas:

- ***PIVOTES CENTRALES :*** Recibe su nombre por su movimiento circular alrededor de un punto central, sobre el que pivota. Es uno de los sistemas más eficientes para regar y para utilizar fertilizantes líquidos sobre el cultivo.

La piramide del pivot, es la estructura central alrededor de la cual gira todo el sistema. Normalmente tiene cuatro patas que están fijadas a unos cimientos, pero también tienen la posibilidad de ser sistemas remolcables, para lo cual esta piramide dispone de ruedas que permiten su transporte fácilmente por el campo. En la piramide se encuentra tanto la toma de agua como la caja eléctrica que alimenta a toda la estructura, así como el panel de control que controla el arranque y parada del pivote.

Fig 21. Torre control pivote central

En la parte superior del pivot se encuentra el codo giratorio, el cual conecta la piramide con los tramos del vehículo. A través de el se bombea el agua a cada uno de ellos, así como el suministro de energía eléctrica. Cada tramo tiene una unidad motriz para mover el pivote alrededor del campo. Los tramos, arqueados, están compuestos por la tubería principal (que puede ser de diferentes diámetros), tirantes y estructuras en forma de V que dan mayor fortaleza y seguridad al conjunto. La unión entre tramos se realiza mediante enganches que permiten oscilaciones (laterales y verticales).

En cada unidad motriz hay una caja eléctrica. Sus componentes básicos son dos micros (de trabajo y de seguridad/alineación). El cable eléctrico que recorre todo el sistema, entrando y saliendo de cada caja eléctrica, lleva dos tipos de carga eléctrica, 460/380 voltios para mover los motores de las torres motrices de cada tramo y 120 voltios para control y maniobra.

El panel de control situado en la pirámide, aparte de controlar la puesta en marcha o parada del pivote, permite seleccionar la dirección de avance: adelante/atrás. Dependiendo de la velocidad que se elija para el sistema, se aplicará una cantidad de agua determinada. A mayor velocidad del vehículo, menor aplicación de agua por hextárea. Para incrementar el riego, el sistema deberá ir más despacio.

Así mismo, para controlar que el vehículo no se desvía de su trazado, en cada una de las torres intermedias hay una varilla de alineación que conecta la base de la caja eléctrica de la torre con el tramo siguiente. Al moverse la última torre, la varilla de la penúltima torre va girando, y al detectar desalineación, enciende y apaga el micro de trabajo, activando el motor de la torre (que se mueve hasta alinearse).

Como ya se ha dicho, aparte del micro de trabajo, también está el micro de seguridad, el cual, si por cualquier causa el sistema se sale demasiado de su alineación, el circuito de seguridad para el sistema. Con ello se evitan daños estructurales y posibles accidentes.

A continuación, se muestra una imagen de como sería una instalación completa de un sistema de regadió por pivot central, donde:

1. Panel de control del grupo electrógeno que alimenta la bomba de agua y todo el sistema eléctrico del pivote central.

2. Bomba vertical que adquiere el agua del pozo.

3. Tubería subterránea por la que se bombea el agua proveniente del pozo.

4. Cuadro de control del pivote central.

5. Codo giratorio.

6. Tramos del vehículo.

7. Punto de unión de cada tramo, en el que se encuentra cada torre de control con sus dos respectivos micros de trabajo y seguridad, así como motores y ruedas que mueven la estructura.

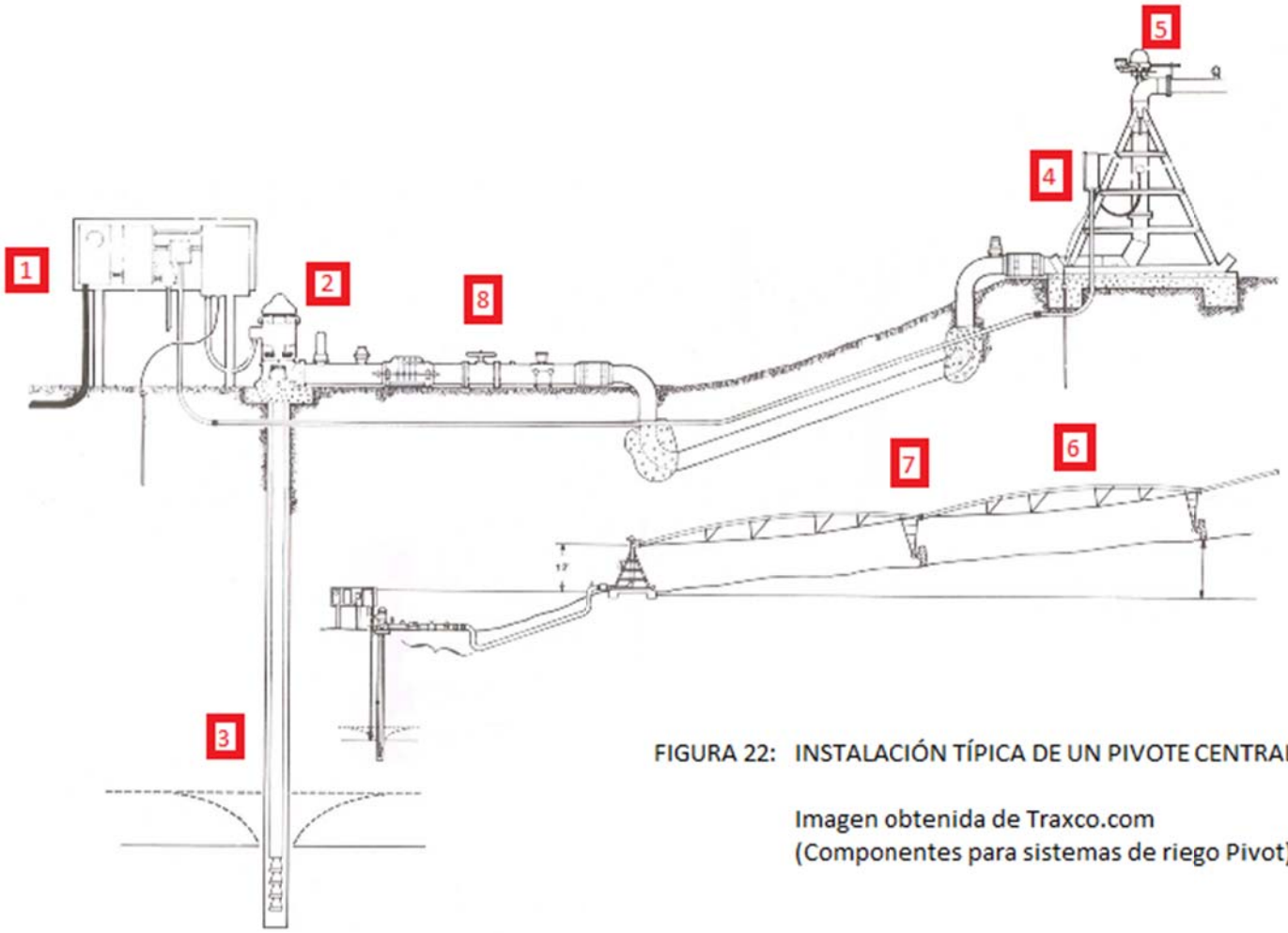


FIGURA 22: INSTALACIÓN TÍPICA DE UN PIVOTE CENTRAL

Imagen obtenida de Traxco.com
(Componentes para sistemas de riego Pivot)

- ***PIVOTES LATERALES:*** A diferencia del Pivot Central, este modelo de vehículo tiene la capacidad de desplazarse en su totalidad a lo largo de la extensión del terreno. Esto es gracias a que en uno de los dos extremos del vehículo, está instalado todo el mecanismo de control, suministro eléctrico y de agua que necesita el vehículo para su buen funcionamiento, en un soporte con cuatro ruedas que le permite desplazarse.

Tanto el suministro de agua como el suministro eléctrico de estos vehículos se puede realizar de dos formas:

La primera de ellas consiste en obtener el suministro eléctrico a través de un cable de conexión, de grandes dimensiones (de longitud) que se conecta desde la cabeza de control motorizada hasta un toma de corriente subterranea, colocada de forma paralela a lo largo del terreno. De esta manera se obtendría la energía electrica necesaria para mover el vehículo directamente de una fuente de transformación instalada por la compañía eléctrica. Así mismo, sucede lo mismo con el suministro de agua; la cabeza de control tiene a bordo un sistema de bombeo, al cual se le conecta una tubería de unas dimensiones concretas en funcion de la cantidad de agua que se precise, que se acopla a la boca de una tubería subterránea colocada de forma paralela a la toma de corriente subterránea, como se observa en la siguiente imagen:

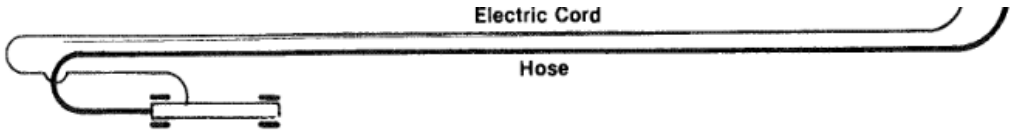


Fig 23. Alimentación con toma de corriente

En el momento que se realice el desplazamiento completo sobre el terreno por parte del vehículo, de un extremo a otro, tanto la tubería de agua como el cable de suministro eléctrico quedaría en la posición opuesta:

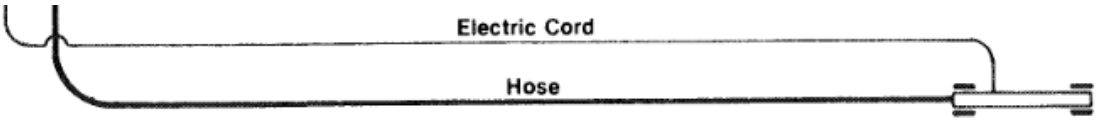


Fig 24. Alimentación con toma de corriente

Una vez en este punto, si el movimiento a realizar es simplemente de derecha a izquierda, y las dimensiones del terreno se cubren con las longitud del cable y tubería representado, no sería necesario desconectarlos y conectarlos a un segundo punto de toma de corriente y de agua. En caso contrario, si el terreno representado en este caso fuese el doble, se conectarían tanto la tubería de suministro de agua como el cable de

corriente al siguiente punto de suministro y se procedería a seguir con el desplazamiento.

Así mismo, para evitar la utilización de cable y tubería, existe la posibilidad tener a bordo de la cabeza de control, un tanque de gasoil que alimente el motor que desplaza el vehículo, así como una acequia con agua que se situe paralela al recorrido del vehículo, de la cual este obtenga el agua necesaria a través de la bomba ya instalada, pero con una tubería que está introducida en la acequia, obteniendo directamente el agua de ella, como se observa en la siguiente imagen:

Fig 25. Obtención de agua por canal

En cuanto al panel de control, es el cerebro de la máquina y permite controlar todas las operaciones. Con este panel se determina: la dirección de desplazamiento del pivote, su velocidad en el campo y el punto en el que se para o da marcha atrás automáticamente (si el equipo tiene estas funciones adicionales), controla automatizaciones que van incorporadas, gestiona la alineación del sistema y todo tio de seguridad, con orden de detener el sistema en caso necesario. Este panel recibe la información de los cofres de los tramos intermedios, del cofre penúltimo y del cofre último, los cuales se explican a continuación.

A diferencia del anterior, este vehículo tiene un sistema de movimiento que controla su desviación diferente al anterior, y se conoce como surco guía, y se controla a través la información que este transmite a través del cofre de tramo.

Fig 26. Sistema de guia por surco

Este sistema tiene la función de controlar los motores de cada torre por separado y mantener el sistema de forma alineada, proporcionando las funciones de seguridad para detener el sistema en caso de desvío de la alineación dentro de los límites establecidos en su configuración.

El panel de control recibe la información sobre el surco guía, si este percibe una desviación de la vara que recorre el surco, manda una orden inmediata a los cofres de tramos para indicarlos que existe una desviación del vehículo en su desplazamiento, por lo que en función de cual sea el tipo de desvio, actuen en consecuencia para volver a poner el sistema en una dirección correcta.

Una vez desarrollado los vehículos de sistema de riego en los que se va a basar este proyecto, se decide cuales de estas características van a simularse en el prototipo que se va a diseñar.

En primer lugar, se decide que tipo de movimiento va a realizar nuestro prototipo. Teniendo en cuenta la movilidad de los dos que se han explicado anteriormente, se decide simular el funcionamiento de un vehículo pivot de movimiento lateral, ya que tiene una mayor eficiencia y funcionalidad que el primero. Para darle nuestro propio toque de innovación, se decide que el panel de control que va en uno de los laterales de este vehículo, se diseñará de tal manera que sea ajeno al vehículo, es decir, se diseñará una estación fija que se comunique con el vehículo para controlar tanto la dirección del vehículo como la velocidad y diferentes opciones que se explicarán más adelante.

De esta manera, se evita la necesidad de tener que acudir al vehículo para su puesta en funcionamiento. Las funciones que se podrán utilizar en la estación fija serán:

- Funcionamiento Manual o Automático.
- Selección de dirección de movimiento, adelante/atrás (Manual).
- Selección de dirección de movimiento y posicionamiento servomotores (Manual).
- Reseteo del sistema en caso de cuelgue del mismo.
- Marcha/Paro de emergencia.
- Puesta en funcionamiento del sistema de riego (Simulado).
- Visualización de los datos enviados al vehículo.
- Control de la velocidad a la que se va a desplazar el vehículo. (Manual/Automático)

Para poder actuar con estas funciones previamente descritas sobre la maqueta del vehículo, será necesaria la programación/instalación de una de las partes más importantes del proyecto, la comunicación por radiofrecuencia.

6. MÓDULO COMUNICACIÓN NRF24L01

El módulo de comunicación por radiofrecuencia NRF24L01, es un transceptor fabricado por ‘’Nordic Semiconductors’’. En este circuito integrado se ha incorporado toda la lógica necesaria para establecer una comunicación inalámbrica bidireccional con acuse de recibo. La comunicación con el microcontrolador se realiza a través de un bus SPI.

‘’El BUS SPI es un estándar de comunicaciones, usado principalmente para la transferencia de información entre circuitos integrados en equipos electrónicos. Sus siglas significacan BUS INTERFAZ DE PERIFÉRICOS SERIE, es un estándar para controlar casi cualquier dispositivo electrónico digital que acepte el flujo de bits serie regulado por un reloj (comunicación sincrónica)**’’**

Las principales características del módulo NRF24L01 son:

- Bajo consumo.
- Frecuencia de trabajo de 2.4GHz.
- Potencia de emisión entre -18 y 0 dBm.
- Velocidad de transmisión entre 1 y 2 Mbps.
- 128 canales de transmisión seleccionables por el bus SPI.

Como interfaz dispone de cuatro pines accesibles para el bus SPI, dos pines más para el control del módulo y otros dos para la alimentación.

Con el objetivo de facilitar el manejo del módulo se han desarrollado unas librerias que simplifican y acortan el tiempo de desarrollo de cualquier aplicación inalámbrica con estos módulos.

La librería consta de seis funciones. A continuación se dará una breve descripción de cada una de ellas.

- **RF_CONFIG** : Confugra las entradas y salidas del microcontrolador, así como parámentros del módulo de radio.
- **RF_CONFIG_SPI** : Configura las entradas y salidas del microcontrolador, así como los parámetros necesarios para utilizar el bus SPI.
- **RF_ON** : Activa el módulo de radiofrecuencia en modo escucha.
- **RF_OFF**: Desactiva el módulo de radiofrecuencia y lo deja en modo de bajo consumo.
- **RF_SEND** : Envía una trama de datos (8 Bytes) a la dirección indicada.
- **RF_RECEIVE**: Comprueba si se ha producido una recepción y de ser así, recoge la trama.

De tal manera, el programa principal habiendo utilizado como programador el Software Niple, la parte de configuración del módulo NRF24L01, quedaría de la siguiente manera:

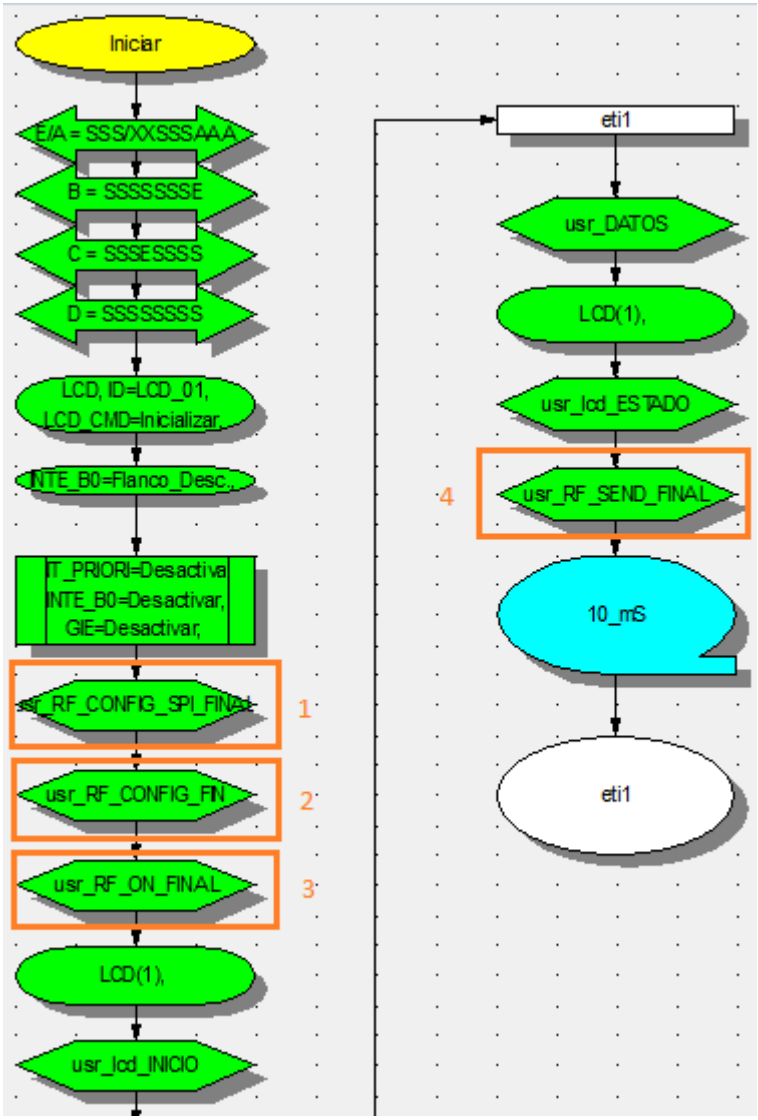


Fig 27. Programa mando de control

El código para la programación de el módulo, se ha obtenido de una librería publicada en internet, la cual está expresada en lenguaje ensamblador (nemónico), por lo que ha sido necesario, línea por línea de código, traducirlo a la manera de programación que utiliza niple, que es mucho más llevadera e intuitiva.

En la siguiente imagen se puede observar como es el código procedente de la librería encontrada para la programación del módulo RF_CONFIG_SPI y como quedaría una vez pasado el código al software niple:

```
RF_CONFIG_SPI

        ;Configuración I/O.
        bsf             STATUS,RP0
        bcf             STATUS,RP1
        bsf             SDI
        bcf             SDO
        bcf             SCK

        ;Configuración módulo comunicaciones.
        movlw   b'11000000'
        movwf   SSPSTAT
        clrf    SSPCON2
        bcf             STATUS,RP0
        movlw   b'00100000'
        movwf   SSPCON
        bcf             STATUS,RP1
        bcf             STATUS,RP0
        return
```

FIGURA 28. Código nemónico

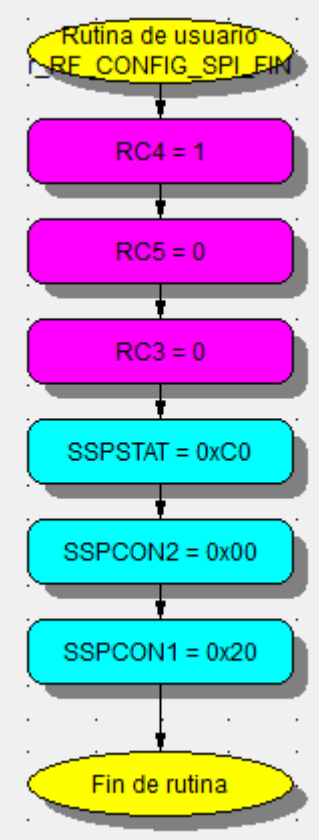


FIGURA 29. Niple

Al comienzo de la librería, se especifican una serie de variables a declarar. Estas serán utilizadas posteriormente en las subrutinas previamente nombradas, para el control del módulo NRF.

En la siguiente imagen se puede observar como vienen definidas en la librería:

```
;*      Variables SPI
BYTE_IN             EQU   0x6F
BYTE_OUT    EQU     0x6E

;*      Variables RF internas
RF_ESTADO   EQU     0x6D
DIR_REG             EQU   0x6C
DATO_REG    EQU     0x6B
RF_NO_HIGH  EQU 0x6A
RF_NO_LOW   EQU 0x69
INTER_RF    EQU     0x68
RF_STATUS   EQU     0x67
DATO_DIR    EQU     0x66
DATA_N_SND  EQU     0x65
DATA_N_RCV  EQU     0x64

;*      Variables RF configurables
RF_CHN              EQU   0x63
RF_DIR              EQU   0x62
RF_DATA_0   EQU     0x61
RF_DATA_1   EQU     0x60
RF_DATA_2   EQU     0x5F
RF_DATA_3   EQU     0x5E
RF_DATA_4   EQU     0x5D
RF_DATA_5   EQU     0x5C
RF_DATA_6   EQU     0x5B
RF_DATA_7   EQU     0x5A

;*      Variables para los Delays
PDel0       EQU     0x59
PDel1       EQU     0x58
PDel2       EQU     0x57

;*      Variable RF_STATUS
#define SNDOK                   0
#define ACK                     1
#define RCVOK                   2
#define RCVNW                   3
```

FIGURA 30. Variables Nemónico

En esta parte del nemónico, se observan las diferentes variables a introducir, cada una de 8 bits (1ª columna), lugar de la memoria en el que crearlo (3ª columna) y valor predeterminado de algunas variables (4ª columna).

Así pues, en la parte final, se puede observar una variable en concreto, RF_STATUS, en la cual se indica que hay que dar nombre a los cuatro primeros bits de esta variable, ya que serán los que nos informen de la situación de la comunicación por el módulo de radio

La variable RF_STATUS, como ya se ha explicado anteriormente, es una variable de solo lectura, la cual informa de la situación de la comunicación por el módulo de radio. De sus 8 bits, se utilizan cuatro de llos, que son los siguientes:

- **RCVNW**: Muestra si todavía quedan datos por leer.
  1 = Quedan tramas de datos por leer en la pila del módulo de radio.
  0 = Tras la última lectura, la pila de datos del módulo quedó vacía. No hay mensajes en espera

- **RCVOK**: Informa que se han recibido datos correctamente y están accesibles para ser tratados.
  1 = Recepción correcta.
  0 = No se han recibido datos o la información recibida es corrupta.

- **ACK**: Muestra si se ha recibido el ACK (confirmación) del receptor tras una transmisión.
  1 = El receptor ha confirmado que ha recibido los datos correctamente.
  0 = No se ha recibido la confirmación del receptor.. Puede ser causa de que no haya recibido la señal o de que tenga la pila llena y no pueda almacenar más mensajes.

- **SNDOK**: Muestra si el último envio de datos se ha realizado.
  1 = El módulo de radio ha enviado los datos. Este bit no indica que alguien lo haya escuchado.
  0 = No ha sido posible enviar los datos. Puede ser debido a un fallo en la comunicación con el módulo de radio.

De tal manera que, a la hora de declarar esta variable, se accede a ella y se editan los nombres de sus cuatro primeros bits (0, 1, 2 y 3):

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Sin uso				RCVNW	RCVOK	ACK	SNDOK
-	-	-	-				

Fig 31. Bits del registro RF_STATUS

A continuación, se explicará detalladamente el funcionamiento de cada una de las seis funciones principales:

o RF_CONFIG

Esta función configura el transceptor, como el canal a utilizar, la velocidad de transmisión, la potencia de emisión, pines del PICm etc.

Dentro de esta subrutina principal, se encuentras las variables de entrada RF_DIR y RF_CHN:

- **RF_DIR**: Es la dirección del dispositivo. Debe ser un valor entre 0x01 y 0XFE.
- **RF_CHN**: Canal a utilizar en la comunicación. Debe ser un valor entre 0x00 y 0x7F (128 canales).

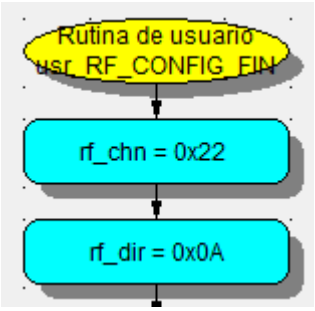


Fig 3. Selección canal y dirección dispositivo

El canal debe ser común a todos los módulos que van a participar en la comunicación. El usuario puede elegir cualquier canal de los 128 disponibles. Sin embargo, si en el entorno existe más de una comunicación entre módulos en diferentes canales, es conveniente dejar un espaciamento de 2 entre los canales a utilizar para evitar interferencias, dejando así 32 canales útiles. Otra cuestión a tener en cuenta es la existencia de otras tecnologías que utilizan la banda ISM 2.4GHz (Wifi, Bluetooth, etc.) ya que también pueden causar interferencias en alguno de los canales, de tal manera que:

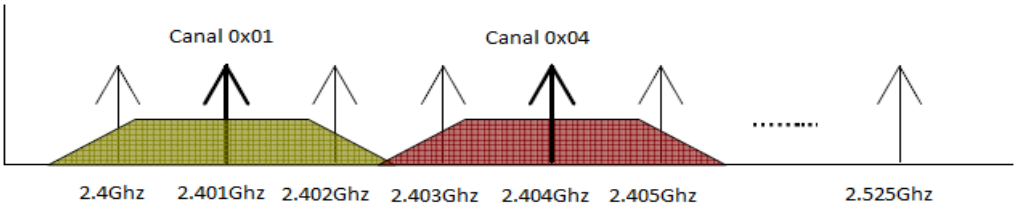


Fig 33. Canales disponibles en la banda de 2,4GHz

- **RF_CONFIG_SPI**

En esa subrutina se configuran los parámetros del BUS SPI. La velocidad del SPI no debe superar los 8 Mhz por lo que la utilización de esta función queda limitada a microcontrolaores PIC con una frecuencia menor de 32Mhz.

Así mismo, en la subrutina se configuran los diferentes parámetros del BUS SPI y los pines del PIC.

PIN RF
SCK
SDI
SDO

Fig 34. Pines de control BUS SPI

- SCK → Reloj del bus SPI
- SDI →Entrada de datos del BUS SPI
- SDO → Salida de datos del BUS SPI

o RF_ON

Esta rutina activa el módulo de radio en modo escucha para poder recibir datos o realizar envios de datos.

Tras la llamada a esta rutina el módulo va a necesitar 2,5ms para estar listo para el funcionamiento.

o RF_OFF

Esta rutina desactiva el módulo de radio dejándolo en modo de bajo consumo. No borra la configuración establecida.

Esta subrutina en concreto, no se encuentra implementada en el proyecto, debedio a que no resultaba práctico para la puesta en funcionamiento de la maqueta del vehículo. Se le añadirá esta opción como mejora final, una vez esté todo en correcto funcionamiento.

o RF_SEND

Esta subrutina, junto con RF_RECEIVE son la clave de la programación del módulo, son las encargadas de enviar y recibir las tramas de datos, por lo que más desarrollo tienen con tema de programación.

Esta subrutina envía 8 Bytes de datos a la dirección indicada informando de la correcta recepción en el destinatario. Tras su ejecución el dispositivo volverá al modo de escucha.

RF_DIR	Dirección a la que se quiere enviar los datos (1 byte).
RFO_DATA_0 – RFO_DATA_7	Variables que van a ser transmitidas (8 bytes).

Fig 35. Datos a configurar en RF_SEND

Además de configurar la dirección a la que se envian los datos configurados y transmitir estos últimos, también se configuran las dos variables responsables de confirmar si se ha realizado una buena transmisión de datos, ACK y SNDOK.

o RF_RECEIVE

Esta rutina se encarga de comprobar si se ha producido una recepción y de ser así, devuelve los datos recibidos. Asi mismo, informa si quedan datos sin leer en la FIFO de recepción del módulo.

Cuando se reciba una trama se debe hacer una comprobación del bit RCVNW de la variable RF_STATUS y si está activo se debe llamar a la función RF_RECEIVE de nuevo tras tratar los datos.

Al no utilizar interrupciones, la probabilidad de pérdida de paquetes, con tráfico elevado, es moderada. Es aconsejable utilizarla sólo en entornos con pocos dispositivos y/o poco tráfico de datos. También se puede solucionar este problema haciendo que los emisores reenvíen la misma trama hasta que la comunicación haya sido correcta, pero en entornos con mucho tráfico las colisiones crecen exponencialmente aumentando considerablemente los tiempos de envío.

Por este motivo previamente explicado, se ha implementado la interrupción por RB0 en el módulo NRF24L01 para este proyecto. Así se impide una recepción continua de tramas de información, evitando saturación del módulo. Por lo que la subrutina RF_RECEIVE se ubica dentro de la interrupción RBO, en la cual, el PIC solo accederá en el momento que reciba una nueva trama de datos procedentes del transceptor que está actuando de transmisor.

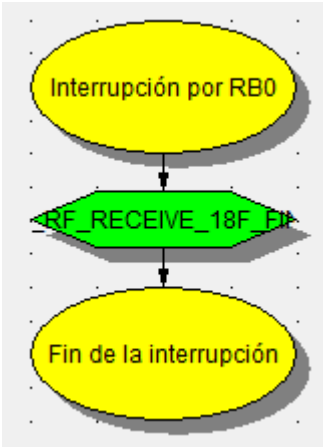


Fig 36. Recepción por interrupción en RB0

7. PROGRAMACIÓN CONSOLA DE MANDO

Una vez explicadas las principales funciones con las que se controla el módulo NRF24L01, se procederá a explicar como ha sido el desarrollo de la estación fija del proyecto, tanto su programación como montaje, así como las complicaciones que se han ido encontrando a medida que se iba avanzando y las soluciones proporcionadas.

Una de las partes más importantes del proyecto es la comunicación mediante radiofrecuencia por parte de las dos maquetas (estación fija y vehículo).

Como ya se ha dicho previamente, el proyecto se va a desarrollar de tal manera que se irá realizando parte por parte, de lo más básico/necesario hasta lo más complejo, sin pasar de un paso a otro hasta que no funcione el anterior correctamente. De esta manera se asegura un desarrollo ordenado y con "copias de seguridad" (tener un punto de guardado con funcionamiento correcto a partir del cual seguir desarrollando el proyecto en caso de error).

La parte fundamental del proyecto es la comunicación entre los módulos de radiofrecuencia NRF24L01, ya que será utilizando estos con lo que, desde la maqueta fija, se podrán controlar las diferentes funciones que poseé la maqueta del vehículo.

Para poder empezar a trabajar con la comunicación entre los transceptores, en primer lugar se necesitaría disponer de dos "estaciones transceptoras", cada una de ellas con su propio módulo NRF24L01, así como con su propio PIC en el cual irá la programación necesaria para el funcionamiento del módulo.

Para ello, se utilizarán dos placas USB PIC'SCHOOL independientes, pero con los mismos elementos (PIC 16F886, NRF24L01, JOYSTICK, SLIDER e INTERRUPTOR). Al principio del proyecto, cuando se empezó la puesta en funcionamiento de la comunicación entre módulos, se utilizó un microcontrolador con una memoria menor, ya utilizado previamente en otro proyecto de la asignatura "*MICROCOPROCESADORES Y MICROCONTROLADORES*". Por lo que, como ya había una cierta familiarización con este PIC, se decidió utilizar para programar los módulos de radiofrecuencia, ya que el programa para que estos funcionen no requería un tamaño importante y el PIC lo soportaba correctamente.

Lo primero de todo, antes de comenzar a montar los distintos elementos y cables de conexión en la placa base, se procedió a la programación del NRF24L01 con el software Niple.

7.1. PASOS EN LA PROGRAMACION CON NIPLE. NRF24L01

7.1.1. Creación documento e inicialización de dispositivos

Para comenzar con un proyecto en Niple, se procede a crear un nuevo marco de trabajo, seleccionando el PIC con el que se va a trabajar, en función de las características del proyecto que se vaya a realizar, se seleccionará un PIC con unas características u otras, las cuales vienen especificadas para cada PIC diferente. En este caso, se seleccionará el PIC 16F886:

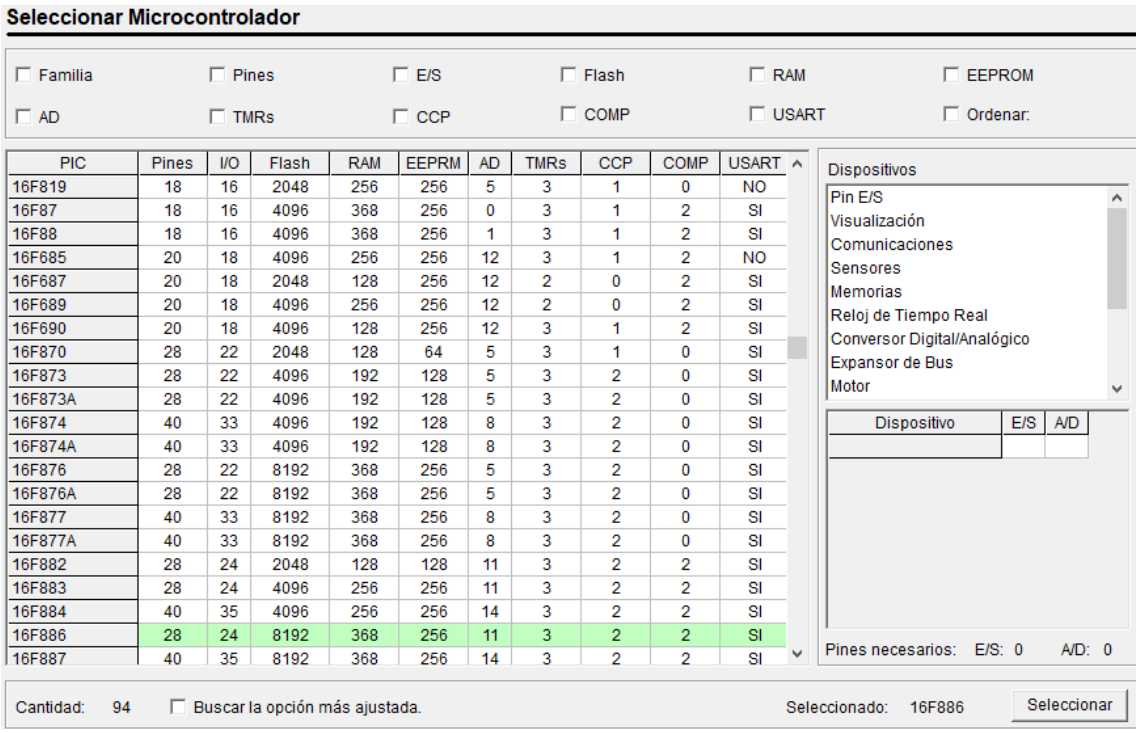


Fig 37. Menú de selección de PIC en niple

Una vez seleccionado el PIC a utilizar, el software te da la opción de adjuntar una breve descripción al documento creado, así como declarar el nombre del mismo y el lugar donde se va a guardar dentro del ordenador:

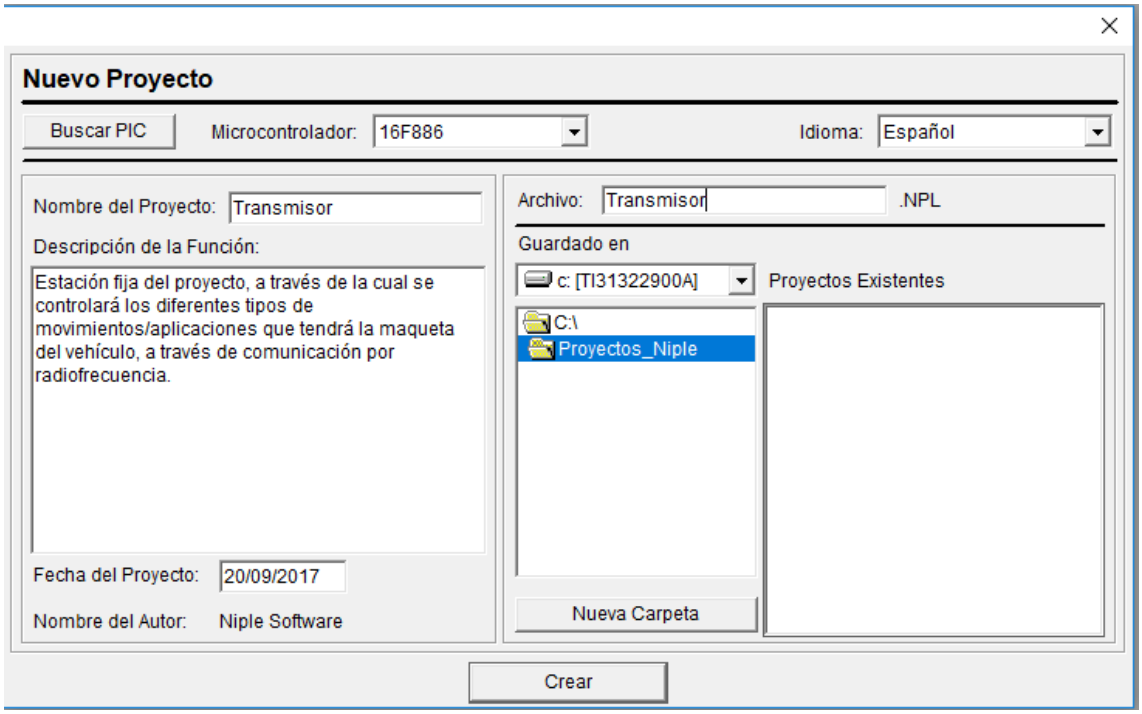


Fig 38. Descripción del proyecto y asignación de nombre

Ya creado el documento, el software muestra la pantalla de trabajo principal, sobre la que se va a ir desarrollando el programa que se desea. En esta pantalla, se puede distinguir la parte sobre la que se van a ir uniendo los bloques de instrucciones y el menú en el que se encuentran las diferentes instrucciones que se pueden incluir, así como la declaración de los puertos del PIC y la inicialización de los diferentes dispositivos que se incluyan en el proyecto.

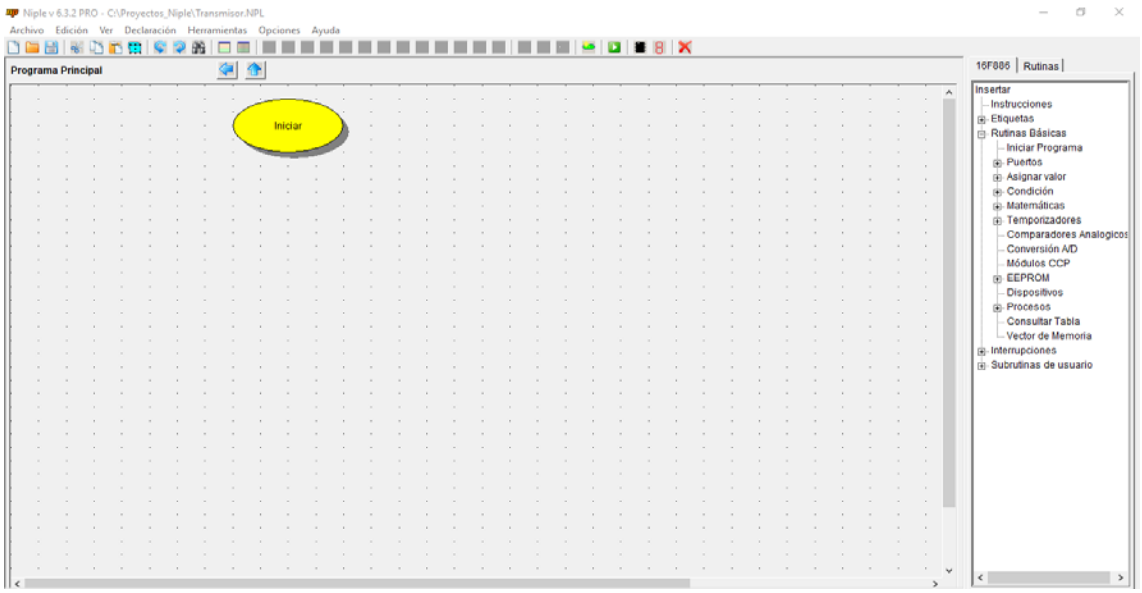


Fig 39. Programa incial

Como ya se ha dicho previamente, en el menú que se puede observar a la derecha de la "*FIGURA 39*" se encuentran las diferentes instrucciones que se pueden usar para el desarrollo del proyecto.

En todo proyecto que se utilice el software Niple, lo primero ha realizar es la declaración de los puertos del PIC seleccionado:

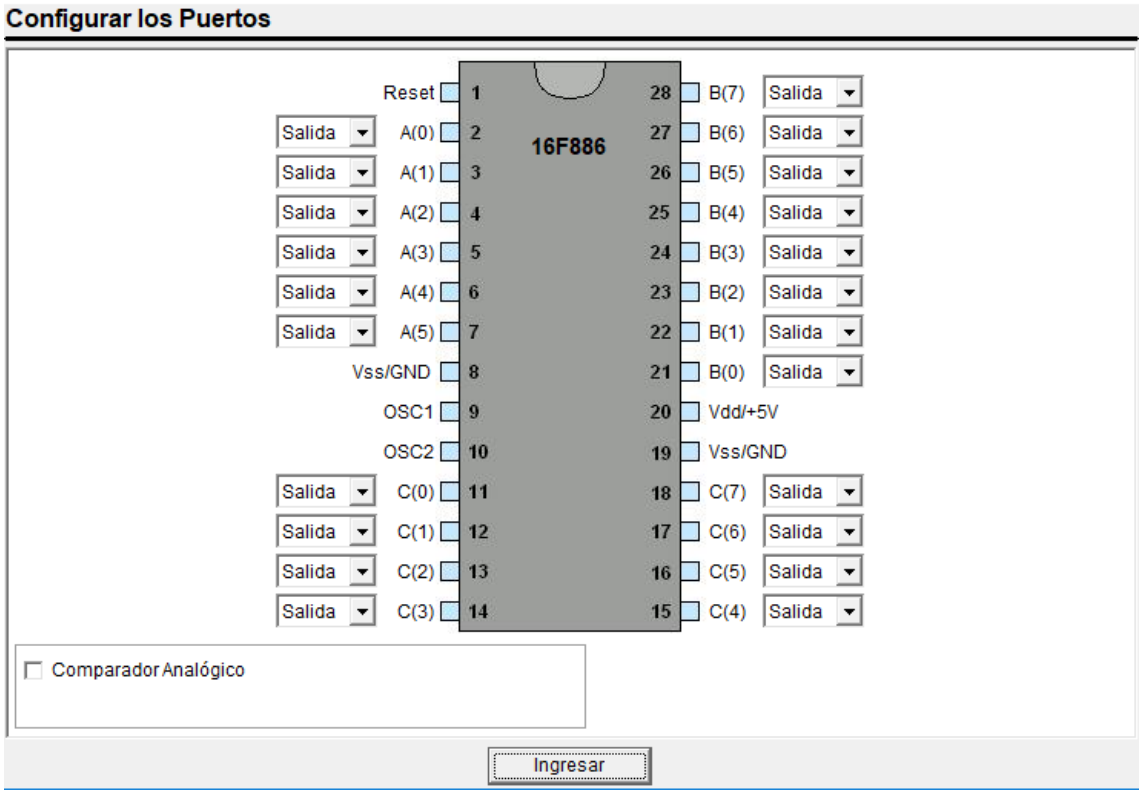


Fig 40. Configuración de los Puertos

En este apartado se puede seleccionar como va a funcionar cada patilla del PIC, en función de las necesidades que se tengan y de la posibilidad de utilizar la patilla para el fin que se desea, ya que, hay patillas que tienen funciones que otras no tienen, como por ejemplo, conversión Analógica/Digital:

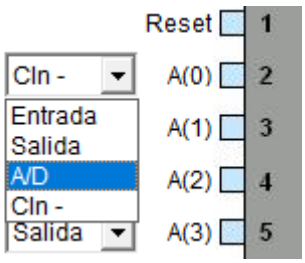


Fig 41. Funciones posibles de cada pin

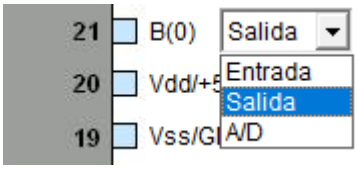


Fig 42. Funciones posibles de cada pin

Como se puede observar en las imágenes previas, las funciones que son capaces de desempeñar los pines “RA0” y “RB0” son diferentes, por lo que hay que saber organizarse en cuanto a la utilización de los pines del PIC. De esta manera, es recomendable, antes de empezar a hacer el proyecto, saber que dispositivos se van a utilizar en el proyecto, cuantos pines requiere la utilización de todos ellos y que tipo, ya que va a agilizar el proyecto considerablemente, y por supuesto, evitar la situación de haber elegido un PIC con insuficientes pines para el proyecto que se desea realizar, obligando a tener que elegir otro PIC y empezar desde cero el proyecto, a no ser que se elija un PIC de la misma familia, de lo cual ya se hablará más adelante.

En este caso, se han elegido los siguientes pines del PIC 16F886 para realizar un programa simple de comunicación bidireccional:

PANTALLA LCD : RB2 - RB3 … RB7
NRF24L01 : RB0 - RC1 - RC2 .. - RC5
POTENCIOMETRO : RA0

Una vez se tiene claro los pines que se van a utilizar, es necesario saber como va a funcionar cada uno, si como entrada, salida o conversor.

En el caso de la PANTALLA LCD, todos los pines funcionarán como **entrada**, ya que solo recibe la información a visualizar por parte del PIC.

En cuanto al módulo NRF24L01, hay variaciones en cuanto a como configurar los pines. En primero lugar, se debe conocer que función desempeña cada conector que poseé el dispositivo:

CE : Siglas en inglés de *CHIP ENABLE*,

CSN : Siglas en inglés de *CHIP SELECT*,

SCK: La señal de reloj es generada por el maestro y sincroniza la transferencia de datos, por lo que se configura como **salida**.

MOSI : Son las siglas en inglés de *MASTER OUT SLAVE IN,* por lo que este pin ha de ser configurado como **salida** en el PIC.

MISO : Son las siglas en inglés de *MASTER IN SALVE OUT*, por lo que este pin ha de ser configurado como **entrada** en el PIC.

IRQ : En el momento que se detecta la entrada de un paquete de datos, la interrupción se activa, por lo que ha de funcionar como **entrada**.

Finalmente, el potenciometro, que se encuentra incorporado a la placa del PIC’SCHOOL, transmite datos analógicos al PIC, por lo que el pin al que se conecte, deberá estar configurado como **entrada ANALÓGICO/DIGITAL**, para poder interpretar los datos en el PIC.

Una vez configurados los puertos con las necesidades de los dispositivos que se van a utilizar, hay que declarar e inicializar aquellos dispositivos que vengan incorporados en el software, en este caso, la pantalla LCD:

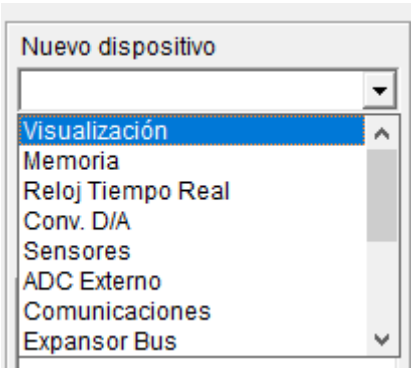


Fig 43. Declaración nuevo dispositivo

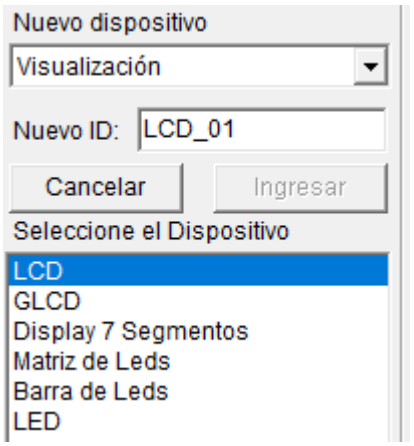


Fig 44. Declaración nueva LCD

Una vez seleccionado la familia del nuevo dispositivo, en este caso visualización, como se observa en la FIGURA 43, el software proporciona diferentes tipos de dispositivos a elegir; en este caso se seleccionará la pantalla LCD, ya que es la que viene incorporada en el PIC'SCHOOL.

Una vez checho click sobre el dispositivo LCD, se abrirá una ventana de configuración, donde se podrá decidir que pines del PIC controlarán la pantalla. Previamente se ha fijado que pines serán los escogidos, por lo que solo es necesario rellenar y aceptar.

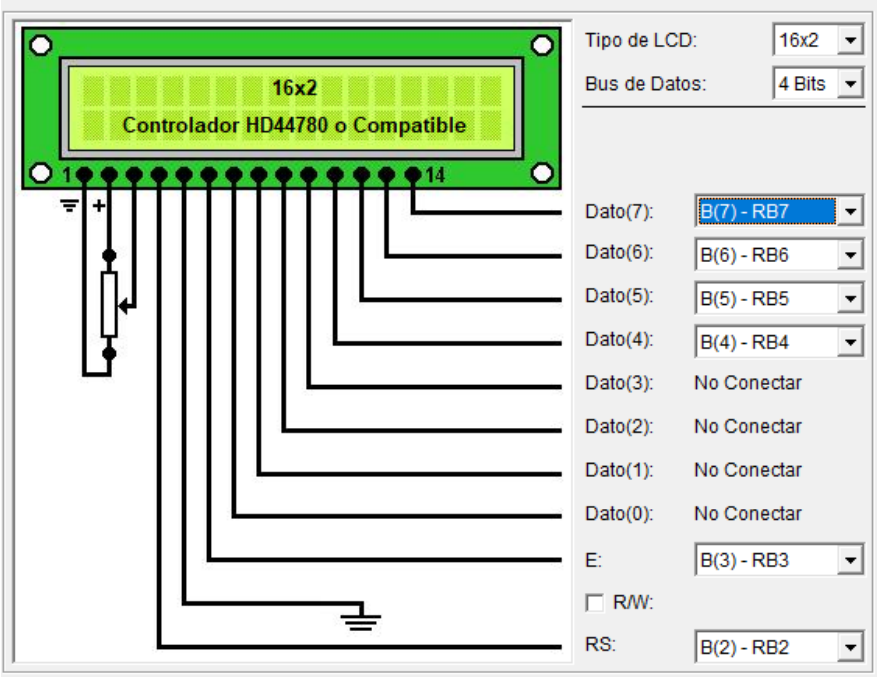


Fig 45. Configuración Pantalla LCD

Una vez creado el dispositivo, en la pantalla principal, se habrá creado un apartado en el menú de la derecha, dentro de la pestaña de dispositivos, un subíndice denominado “PANTALLA LCD”, el cual dará la posibilidad, pinchando en el, de inicializar el dispositivo, añadiendo una instrucción de inicializar al comienzo del programa, como se puede observar en las siguientes imágenes:

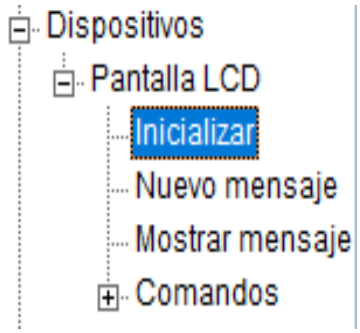


Fig 46. Inicialización LCD

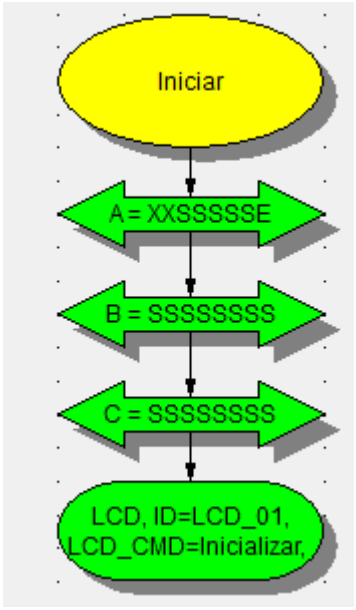


Fig 47. Inicialización en el programa

Inicializada ya la pantalla LCD, se procede a configurar el siguiente paso, que sería la interrupción por RB0 del módulo NRF24L01.

Al igual que la pantalla LCD, se necesario configurar y activar la interrupción, para ello, en el menú principal hay que seleccionar el apartado “INTERRUPCIONES”, donde en primer lugar, se deberá acceder al apartado “CONFIIGURAR INTERRUPCIÓN”.

En este apartado se configura la manera con la que el PIC va a tener que detectar la interrupción. En este caso, la interrupción va a ser detectado por el pin RB0 del Puerto B, por lo que habrá que seleccionar dentro de el apartado “PUERTO” la opción “INTERRUPCIÓN POR FLANCO EN RB0” (FIGURA 49)).

En las especificaciones del módulo de radiofrecuencia NRF24L01, especifica que la interrupción funciona como un flanco descendente, es decir, la patilla RB0 estará continuamente recibiendo una tensión continua mientras no esté recibiendo ningún paquete de datos, mientras que por el contrario, cuando reciba uno, esta señal decaerá, creando un flanco con distinto valor (pasa de “1” a “0” en valor digital). Esta explicación se puede entender mejór con la siguiente imagen (FIGURA 48):

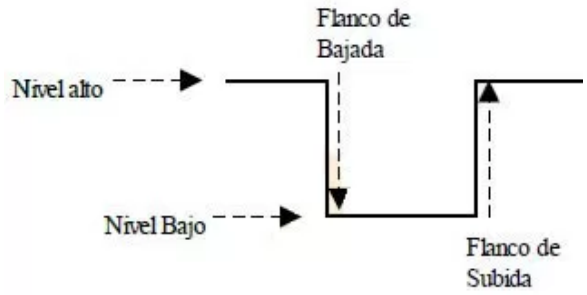


Fig 48. Cambio de estado en una señal digital

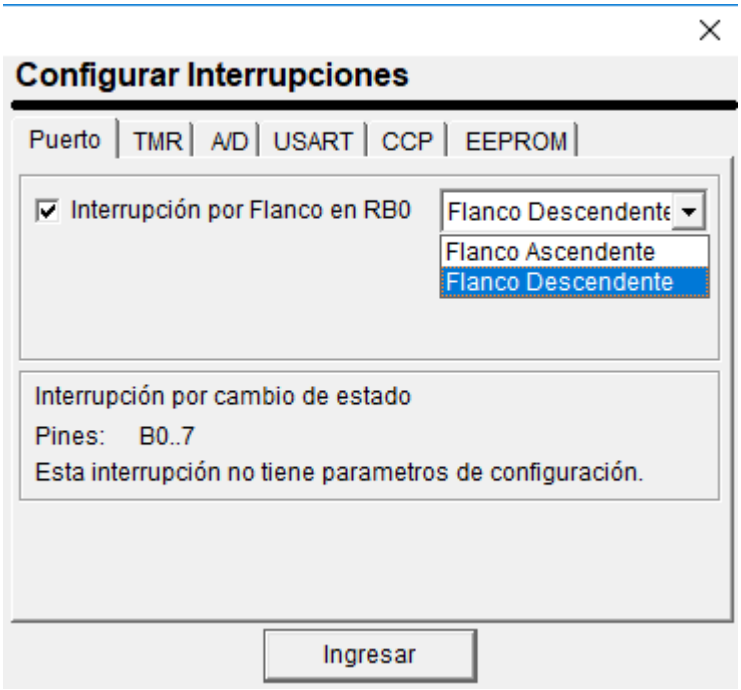


Fig 49. Configuración de interrupciones

Una vez seleccionada la interrupción por detección de flanco descendente en el pin RB0, el siguiente paso sería activar el PIC para que detecte las interrupciones, por lo que siguiendo unos pasos similares a los llevados a cabo con la pantalla LCD, se selecciona en el menú principal, en el menú de la interrupciones, la opción de "ACTIVAR/DESACTIVAR" donde después de seleccionarla, aparecerá un menú en el que permite activar la detección de interrupciones y el tipo que se requiere:

Fig 50.

7.1.2. Declaración de Registros, Bits y desarrollo de Subrutinas

Finalizada la primera parte de configuración de los puertos, así como la declaración de la pantalla LCD y la activación de la interrupción por flanco descendente en RB0, el siguiente paso es programar las diferentes subrutinas que van a permitir la comunicación entre módulos. Estas son las nombradas en la página 31.

Previamente, antes de programar las seis funciones principales, es necesario la declaración de las variables que aparecen en la primera página de la librería en nemónico, ya que se utilizarán durante la escritura de las instrucciones dentro de estas seis funciones.

DECLARACIÓN DE VARIABLES

El conjunto de variables a declarar es el siguiente:

```
;*****************
;*      VARIABLES   *
;*****************

;*      Variables SPI
BYTE_IN             EQU   0x6F
BYTE_OUT    EQU     0x6E

;*      Variables RF internas
RF_ESTADO   EQU     0x6D
DIR_REG             EQU   0x6C
DATO_REG    EQU     0x6B
RF_NO_HIGH  EQU 0x6A
RF_NO_LOW   EQU 0x69
INTER_RF    EQU     0x68
RF_STATUS   EQU     0x67
DATO_DIR    EQU     0x66
DATA_N_SND  EQU     0x65
DATA_N_RCV  EQU     0x64

;*      Variables RF configurables
RF_CHN              EQU   0x63
RF_DIR              EQU   0x62
RF_DATA_0   EQU     0x61
RF_DATA_1   EQU     0x60
RF_DATA_2   EQU     0x5F
RF_DATA_3   EQU     0x5E
RF_DATA_4   EQU     0x5D
RF_DATA_5   EQU     0x5C
RF_DATA_6   EQU     0x5B
RF_DATA_7   EQU     0x5A
```

FIg 51. Variables a declarar en la librería.

Para cada una de estas variables, es necesario crear un registro nuevo en el banco de memoria RAM correspondiente del PIC.

Para ello, el software Niple ofrece una forma rápida y cómoda de crearlos, así como de darles unos valores determinados, seleccionar el tamaño de registro,etc.

Para poder crear un nuevo registro, es necesario acceder al menú principal del proyecto existente, y dirigirse al menú de la parte superior, donde se encuentran una serie de herramientas. Una de ellas, señalada en la "FIGURA 52" va a permitir acceder al banco de registros:

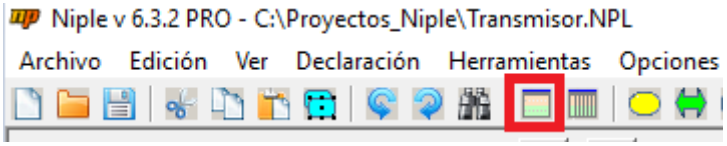


FIGURA 52

Una vez accedido al banco de registros, se puede observar el número de bancos que tiene el PIC seleccionado, con sus diferentes registros libres u ocupados. Los registros libres sobre los que se puede actuar son representados con un color verde claro, mientras que los registros ocupados sobre los que no se puede actuar son representados con un color naranja.

En la librería del módulo NRF24L01, cuando indica las variables que se necesitan, se puede observar que indican el banco y la posición dentro de este en el que tiene que ir cada variable (3ª columna), sin embargo este dato es indiferente, por lo que se crearan en los primeros espacios libres que se encuentren.

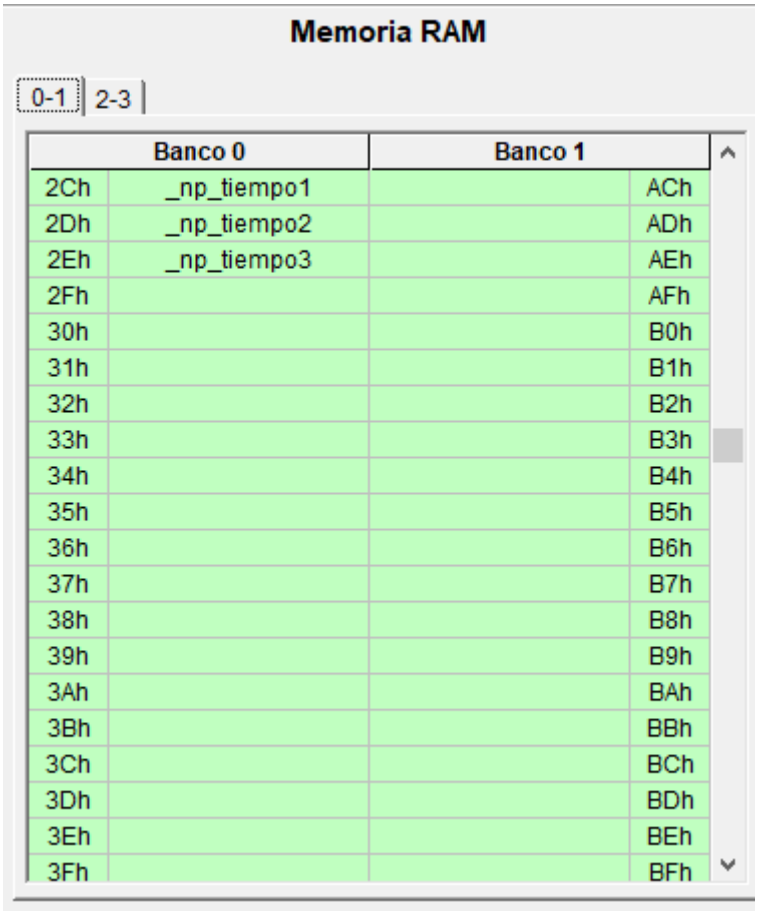


Fig 53. Declaración de registros en los bancos de memoria RAM

En el momento de seleccionar dos veces un registro vacío, se abrirá una ventana con la que interactuar, para llevar a cabo la creación del registro:

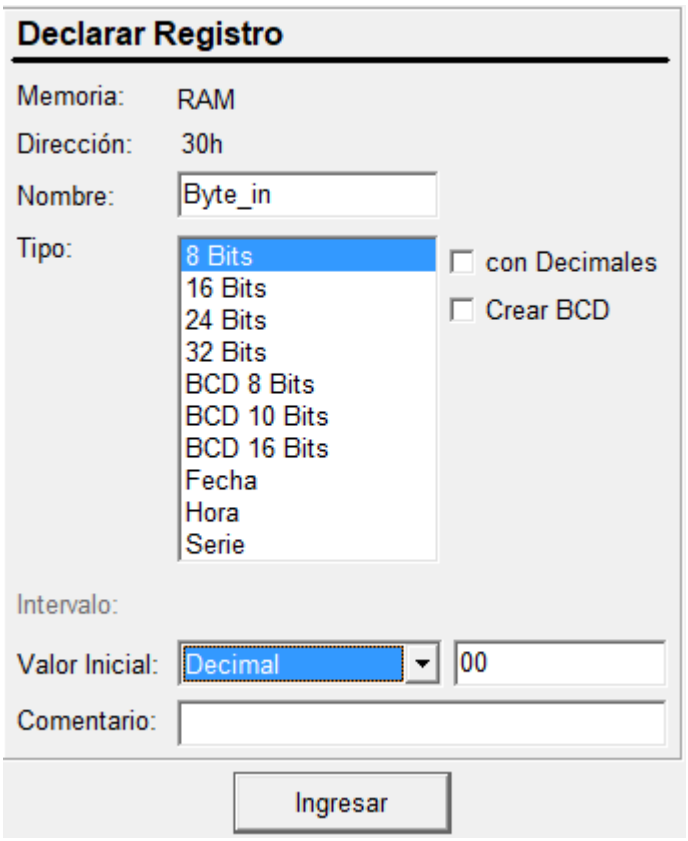


Fig 54. Declaración registro

En la "FIGURA 54" se puede observar las distintas posibilidades que se ofrecen. Las variables que se han de crear, deben de tener un tamaño de 8 bits, ya que la función RF_SEND envía tramas de datos de 8 bytes (RFO_DATA_0,….RFO_DATA_7).

En las variables en las cuales se indica que hay que incluir un valor inicial, este valor viene mostrado en su valor Hexadecimal, por lo que se seleccionaría este en lugar de decimal y se crearía el registro.

Este proceso hay que hacerlo con cada una de las variables indicadas en la librería, así como la declaración de los bits de la variable RF_STATUS.

Esto último, asignar un nombre a cada bit del registro RF_STATUS, ha de hacerse después de haberse declarado la variable. La herramienta que se encuentra a la derecha de la que se ha utilizado para declarar los registros, es la que permite acceder a los bits de estos.

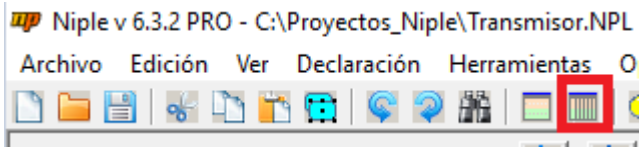
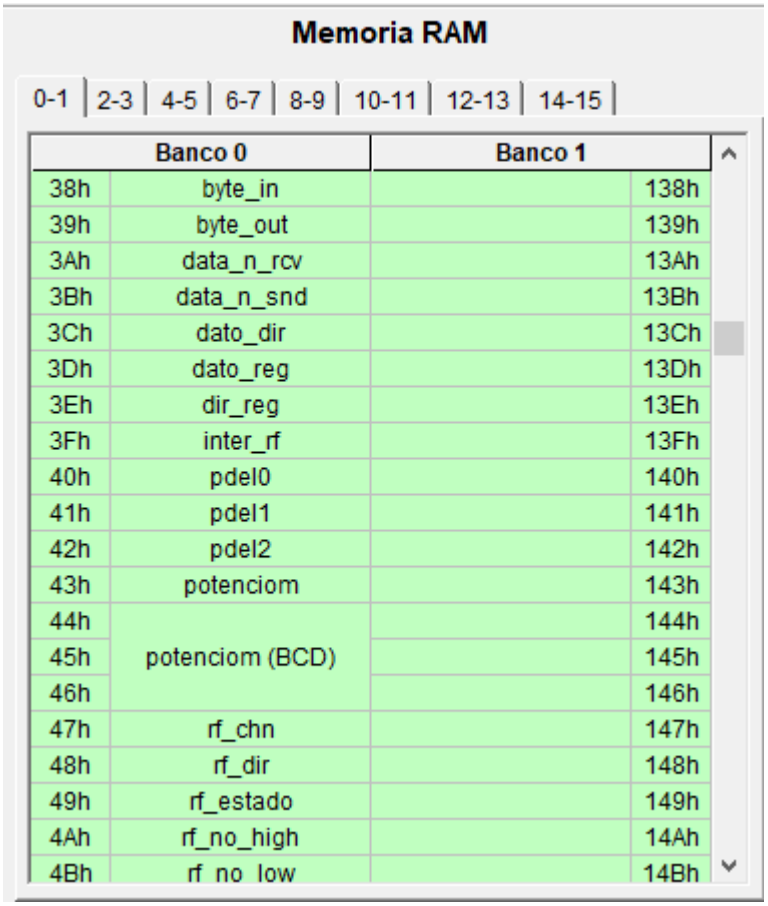


Fig 55.

Una vez accedido a esta ventana, se visualizan todos los registros previamente creados, seguidos de sus ocho bits (Bit 0… Bit 7). Solo es necesario buscar el registro RF_STATUS y darle el nombre que se indica en la librería a los bits señalados.

El resultado final de haber creado todos los registros indicados en la librería, da como resultado una visión del banco de registros como la que se puede observar a continuación:



Banco 0		Banco 1	
38h	byte_in		138h
39h	byte_out		139h
3Ah	data_n_rcv		13Ah
3Bh	data_n_snd		13Bh
3Ch	dato_dir		13Ch
3Dh	dato_reg		13Dh
3Eh	dir_reg		13Eh
3Fh	inter_rf		13Fh
40h	pdel0		140h
41h	pdel1		141h
42h	pdel2		142h
43h	potenciom		143h
44h	potenciom (BCD)		144h
45h			145h
46h			146h
47h	rf_chn		147h
48h	rf_dir		148h
49h	rf_estado		149h
4Ah	rf_no_high		14Ah
4Bh	rf_no_low		14Bh

Fig 56. Registros declarados

El siguiente paso a seguir, será crear las subrutinas principales (a partir de ahora se referirá a las seis funciones principales previamente nombradas como “subrutinas”).

El motivo principal de crear subrutinas, es la fácil organización del espacio de trabajo que estas ofrecen. Dentro de ellas, se pueden crear tanto pequeños como grandes ramas de instrucciones.

Para crear una subrutina el proceso a seguir es sencillo; en primer lugar es necesario estar en la página principal del proyecto y acudir al menú lateral derecho, donde se debe acceder al apartado “SUBRUTINAS DE USUARIO”, allí se encuentra la pestaña “NUEVA SUBRUTINA”.

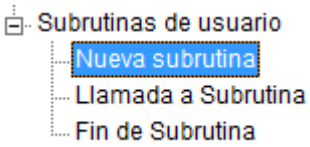


Fig 57. Creación de una nueva subrutina

Una vez accedido a esta pestaña, aparecerá una ventana emergente donde se le asignará un nombre a la subrutina, en este caso, RF_CONFIG_SPI. Así mismo, aparece un espacio de escritura para una posible descripción de esta, donde es recomendable escribir que es lo que se ejecuta dentro de la subrutina, por el simple hecho de orden y claridad :

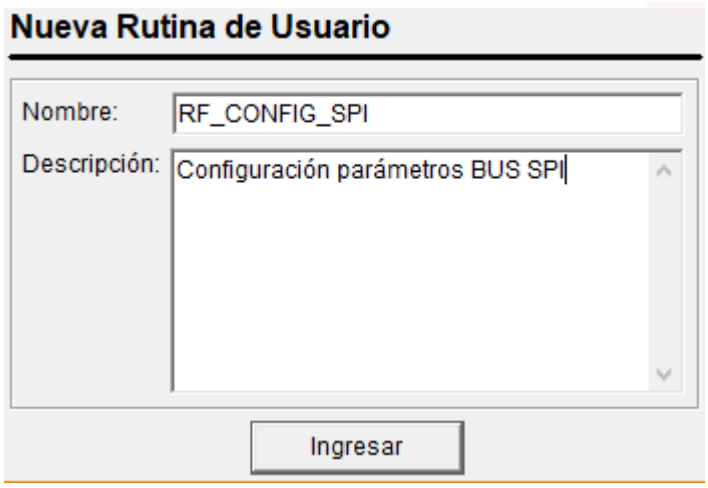


Fig 58.

Una vez creadas las seis subrutinas principales, es necesario desplazarlas al programa principal, por lo que se procede a “llamar a la subrutina”. Para esto, se accede al menú lateral que se había accedido previamente para la creación de estas, y se selecciona la pestaña “LLAMADA A SUBRUTINA” (FIGURA 59), donde se encontrarán todas las subrutinas creadas. Simplemente es necesario seleccionar la que nos interesa y presionar ingresar (FIGURA 60), en ese momento aparecerá en la vista de trabajo principal (FIGURA 61).

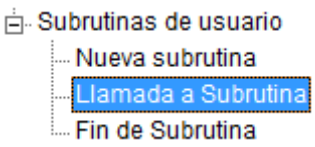


Fig 59.

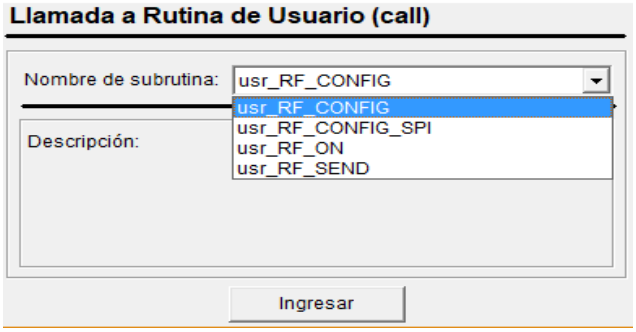


Fig 60. Llamada a una subrutina en particular

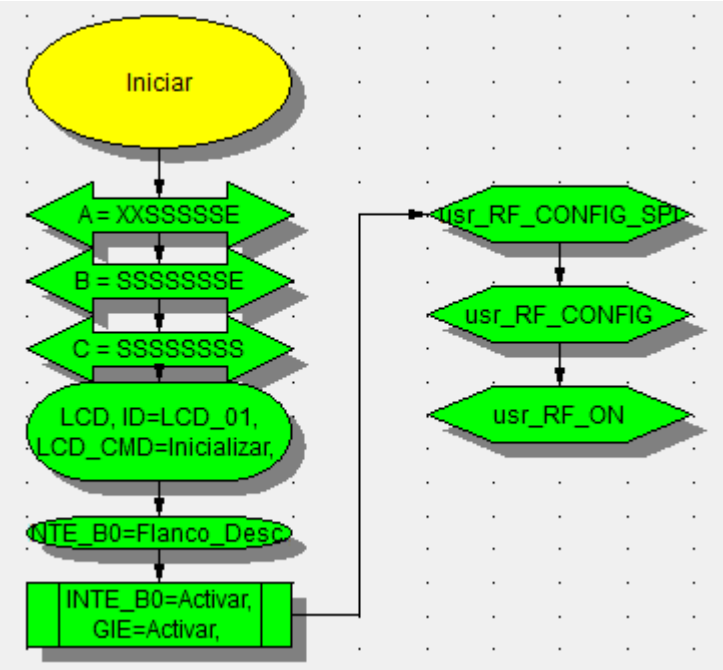


Fig 61. Programa principal con las nuevas subrutinas añadidas

Las subrutinas restantes que no aparecen en la FIGURA 61, se introducirán en el programa en su debido momento. En esta imagen se han introducido aquellas que son necesarias ejecutar cada vez que el dispositivo es encendido.

Las restantes (RF_SEND y RF_RECEIVE) se incluiran en los programas que se ejecuten en la estación fija y en la maqueta del vehículo, respectivamente.

Una de las ventajas de organizar un proyecto utilizando subrutinas, es que te permite exportar estas a un fichero creado en el ordenador. Esto significa que, si por algún casual, un proyecto diferente al que pertenecian, necesita ejecutar la misma subrutina, es posible importarla al nuevo proyecto, sin necesidad de tener que volver a rellenar el código que se requiere. Esto significa una gran ventaja para este proyecto, ya que debido a errores, alguna que otra vez va a resultar necesario empezar de cero el programa, y con esta posibilidad se hace de nuevo en muy poco tiempo.

Por eso mismo, en el momento que se hagan correctamente las subrutinas principales, comprobando que los módulos de radiofrecuencia se comunican correctamente, se hace una copia de seguridad de estas para que, si por algún casual, el archivo del proyecto ".npl" se daña, tener las subrutinas intactas y en buen funcionamiento.

Se desarrollará por partes la programación de cada una de las tres subrutinas mostradas en la FIGURA 61, así como las de RF_SEND y RF_RECEIVE, y se explicará como se protegen estas, exportandolas a un archivo seguro.

7.1.3. NEMÓNICO

Antes de empezar a explicar la programación de las diferentes subrutinas que poseé el proyecto, merece la pena incluir una pequeña introducción al lenguaje informático que se va a emplear para pasar las instrucciones de la librería al software Niple. A continuación se adjunta unas tablas con las instrucciones que se van a encontrar en la librería, así como su significado:

ADDWF	Suma el al registro (f), el valor del acumulador (w).
CLRF	Pone a "0" el registro (f).
CLRW	Pone a "0" el acumulador (w).
DECF	Disminuye en "1" el valor del registro (f).
INCF	Aumenta en "1" el valor del registro (f).
MOVF,0/1	Mueve el valor del registro (f) al acumulador (w) cuando es "0" y al mismo registro cuando es "1" para verificar lo escrito.
MOVWF	Mueve el valor que hay en el acumulador (w) al registro (f)
SUBWF	Resta al registro (f) el valor del acumulador (w) y este resultado lo guarda en un registro (f)

INSTRUCCIONES ORIENTADAS A REGISTROS

BCF, b	Pone a "0" el bit (b) del registro (f)
BSF, b	Pone a "1" el bit (b) del registro (f)
BTFSC, b	Salto a la siguiente instrucción si el bit (b) del registro (f) es un "0"
BTFSS, b	Salto a la siguiente instrucción si el bit (b) del registro (f) es un "1"

INSTRUCCIONES ORIENTADAS A BIT

ADDLW	Sumar al acumulador (w) un literal (L) y guardarlo en el acumulador (w).
CALL	Llamar a subrutina (k).
GOTO	Ir a etiqueta (k).
MOVLW	Mover un literal (L) al acumulador (w).
SUBLW	Restar al acumulador (w) un literal (L) y guardarlo en el acumulador (w).

INSTRUCCIONES CON LITERALES Y DE CONTROL

Una vez entrado en contacto con los comandos en mnemónico que se van a poder observar en el siguiente apartado del documento, comienza la parte de escritura del código de las subrutinas en Niple:

7.1.4 PROGRAMACIÓN SUBRUTINAS PRINCIPALES

7.1.4.1 RF_CONFIG_SPI

En primer lugar, se procederá a explicar el desarrollo de la subrutina conocida como RF_CONFIG_SPI en la cual se configura el módulo SPI del microcontrolador.

En ella se especifica como salida MISO y como entrada MOSI entre otros parámetros del protocolo SPI.

En primer lugar, se observan las instrucciones en mnemónico, se interpretan y se traduce a la simbología con la que trabaja niple:

RF_CONFIG_SPI

```
bsf     STATUS,RP0
bcf     STATUS,RP1
bsf     SDI
bcf     SDO
bcf     SCK

movlw   b'11000000'
movwf   SSPSTAT
clrf    SSPCON2
bcf     STATUS,RP0
movlw   b'00100000'
movwf   SSPCON
bcf     STATUS,RP1
bcf     STATUS,RP0
return
```

Fig 62. Instrucciones en nemónico

```
;*********************
;*      DEFINICIONES    *
;*********************
;*
;
        ;PORTB
#define IRQ         PORTB,0
#define CSN         PORTB,7
        ;PORTC
#define CE          PORTC,2
#define SCK         PORTC,3
#define SDI         PORTC,4
#define SDO         PORTC,5
```

Fig 63. Definiciones en nemónico

Las instrucciones a seguir para la elaboración de la primera subrutina son las que se muestran en la FIGURA 62, mientras que la FIGURA 63 muestra unas definiciones que se pueden encontrar al principio de la librería, donde se define el patillaje del PIC.

En primer lugar, se encuentras las instrucciones “BSF SDI, BCF SDO y BCF SCK” las cuales, como se ha visto previamente en las tablas de mnemónico, ponen a “1” o “0” el bit al que se refiera a continuación, en este caso SDI, SDO y SCK, los cuales, en esta ocasión, pone a “1”, “0” y “0” respectivamente, por lo que representado en Niple, quedaría de la siguiente manera:

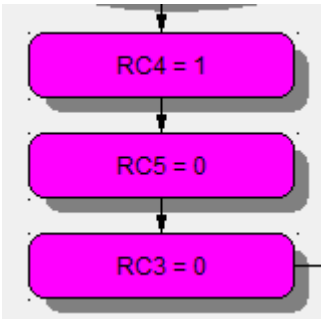


Fig 64. Asignación valor a bits

A continuación se puede observar instrucciones que actuan sobre los registros SSPSTAT, SSPCON2 y SSPCON, los cuales son registros fijos en el PIC, sobre los cuales no se puede actuar en el banco de registro.

Al registro SSPSTAT se le pretende asignar un valor a partir de un número binario previamente grabado en el acumulador:

```
Movlw b'11000000
Movwf SSPSTAT
```

El número binario consta de 8 bit y se va a proceder a traducir a hexadecimal:

1100	0000	Valor binario 4 bits
(12)	(0)	Valor decimal

En la representación Hexadecimal se utilizan tanto letras como números:

0 1 2 3 4 5 6 7 8 9 A B C D E F

Los diez primero dígitos (de 0 a 10) se representan como tal, pero a partir del número diez, se representan con letras, A, B, C, D, E, F, que representan al 10, 11, 12, 13, 14 y 15, respectivamente.

Por lo que el número binario 11000000 se traduciría a Hexadecimal como:

0xC0

La siguiente instrucción es CLRF SSPCON2, lo cual pondría con valor "0" el registro SSPCON2

La última instrucción es exactamente igual que la que se ha realizado previamente, grabar un número literal en el acumulador, y de este, al registro SSPCON.

```
Movlw b'00100000
Movwf SSPCON
```

Realizando la misma conversión previamente echa, el número binario quedaría tal que así:

0x20

Y así quedarían las tres instrucciones en Niple:

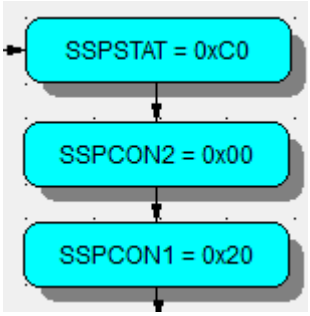


Fig 65.

De esta sencilla manera, con las instrucciones que se pueden encontrar en el menú del lateral derecho de Niple (FIGURA 66), se pueden ejecutar todas las instrucciones que poseé la librería, dependiendo si se va a actuar sobre un Registro, un

Bit, un Literal o si es una instrucción de control (FIGURA 67), pudiendo seleccionar ahí la acción a realizar, previamente señaladas en las tablas de Mnemónico en la página 53.

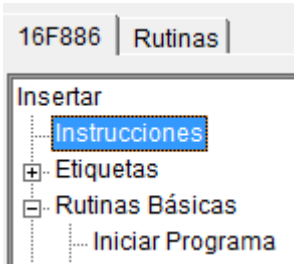


Fig 66 .

Fig 67.

Quedaría así acabada la primera subrutina principal del proyecto, de las más simples y rápidas de traducir a Niple que se pueden encontrar en la librería.

De esta manera, se va traduciendo toda la librería al software Niple, subrutina por subrutina, código por código, de tal manera que la explicación de las siguientes subrutinas, se ceñirá en el funcionamiento de estas en el Software Niple ya traducido, debido a que resulta poco práctico explicar línea por línea todo el código en mnemónico, por el espacio que ocuparía y por lo repetitivo que resultaría.

7.1.4.2. RF_CONFIG

Esta función se encarga de configurar el transceptor habilitando su propia dirección de escucha y canal entre otros parámetros.

Así pues, en esta subrutina se configuran dos de los valores que van a a ser esenciales para llevar a cabo la comunicación entre los módulos, RF_CHN y RF_DIR.

RF_CHN es el registro en el cual se escribirá el canal por el que se realizará la comunicación.

RF_DIR es de un solo byte. Indica el origen o destino del mensaje de 8 bytes de RF_DATA. En esta variable se escribirá la dirección del destinatario a la hora de realizar un envío. En el caso de la recepción, en esta variable se podrá ver la dirección de quien está enviando la información.

En la siguiente página, es encuentra el esquema completo de la subrutina, representado con los bloques de Niple.

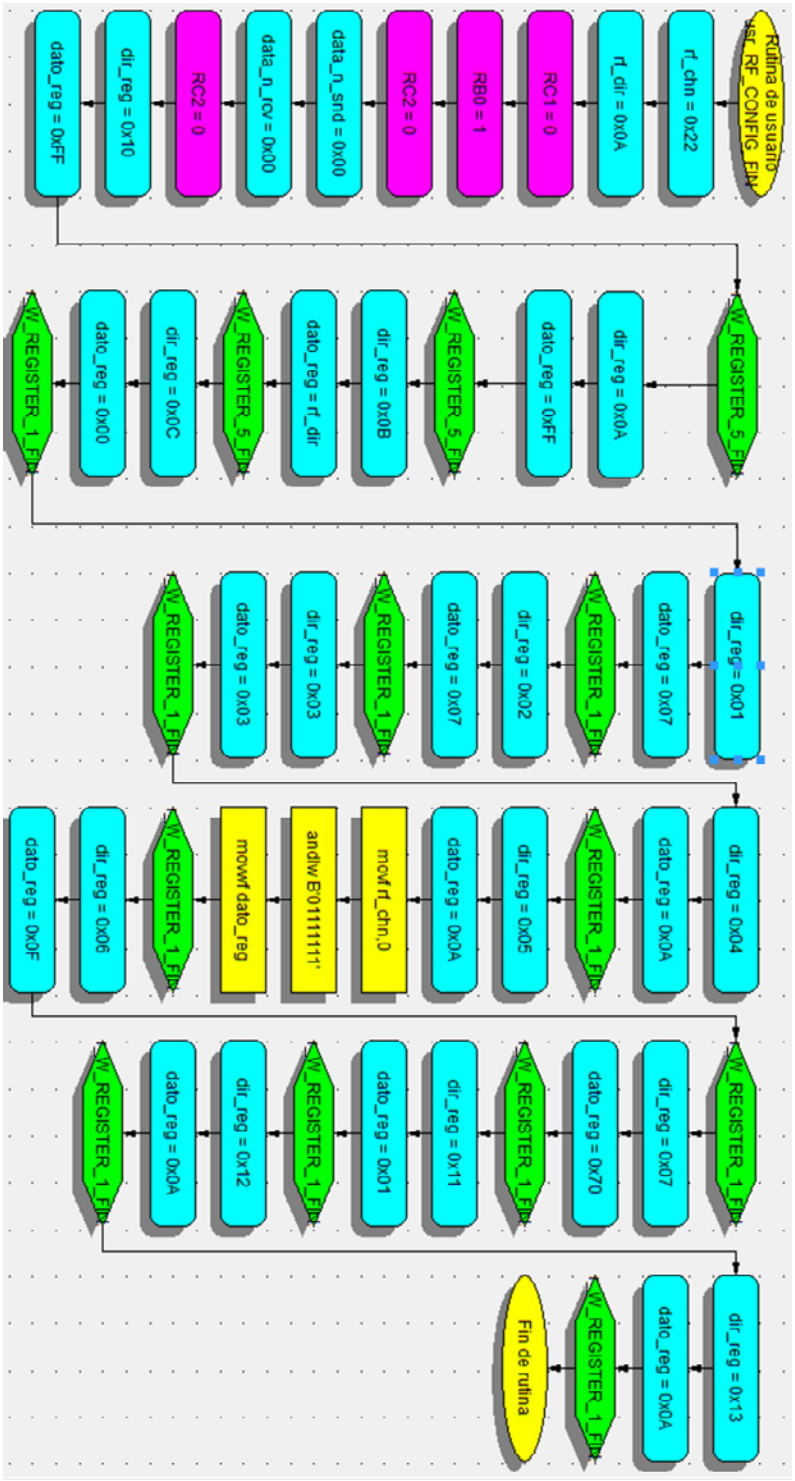


Fig 68. Esquema Niple RF_CONFIG

7.1.4.3. *RF_ON*

Esta rutina activa el módulo de radio en modo escucha para poder recibir los datos y/o realizar envíos de datos.

Es importante tener en cuenta que tras la llamada a esta rutina el módulo necesita 2,5ms para estar listo.

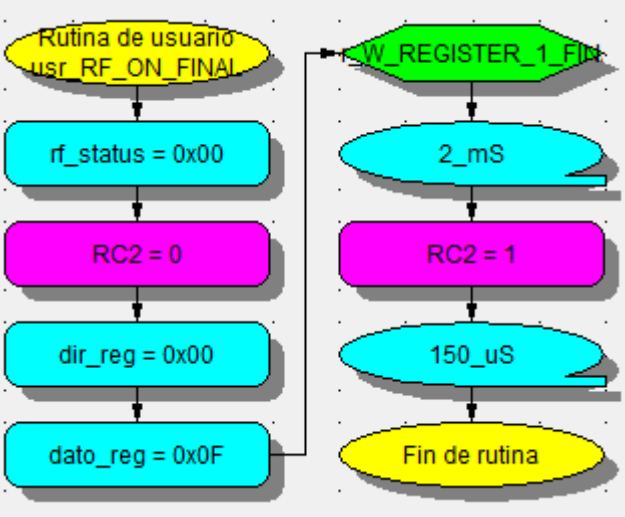


Fig 69. Esquema RF_ON

7.1.4.4. *RF_SEND*

Esta función envía 8 Bytes de datos (RFO_DATA_0 – RFO_DATA_7) a la dirección indicada (RF_DIR) informando de la correcta recepción en el destinatario (SNDOK y ACK). Tras su ejecución el dispositivo volverá al modo de escucha.

Como ya se explicó anteriormente, el registro “ACK” muestra si se ha recibido la confirmación del receptor tras la transmisión, así como el “SNDOK” muestra si el último envío de datos se ha realizado.

Es importante editar en esta subrutina RF_DIR, que se corresponde con el ID del equipo al cual se le quieren enviar los 8 Bytes de datos, ya que si el valor no coincide con el ID del destinatario, la trama de datos no le llegará.

A continuación se va a mostrar el esquema correspondiente a la subrutina RF_SEND dividida en tres partes, debido al tamaño del mismo:

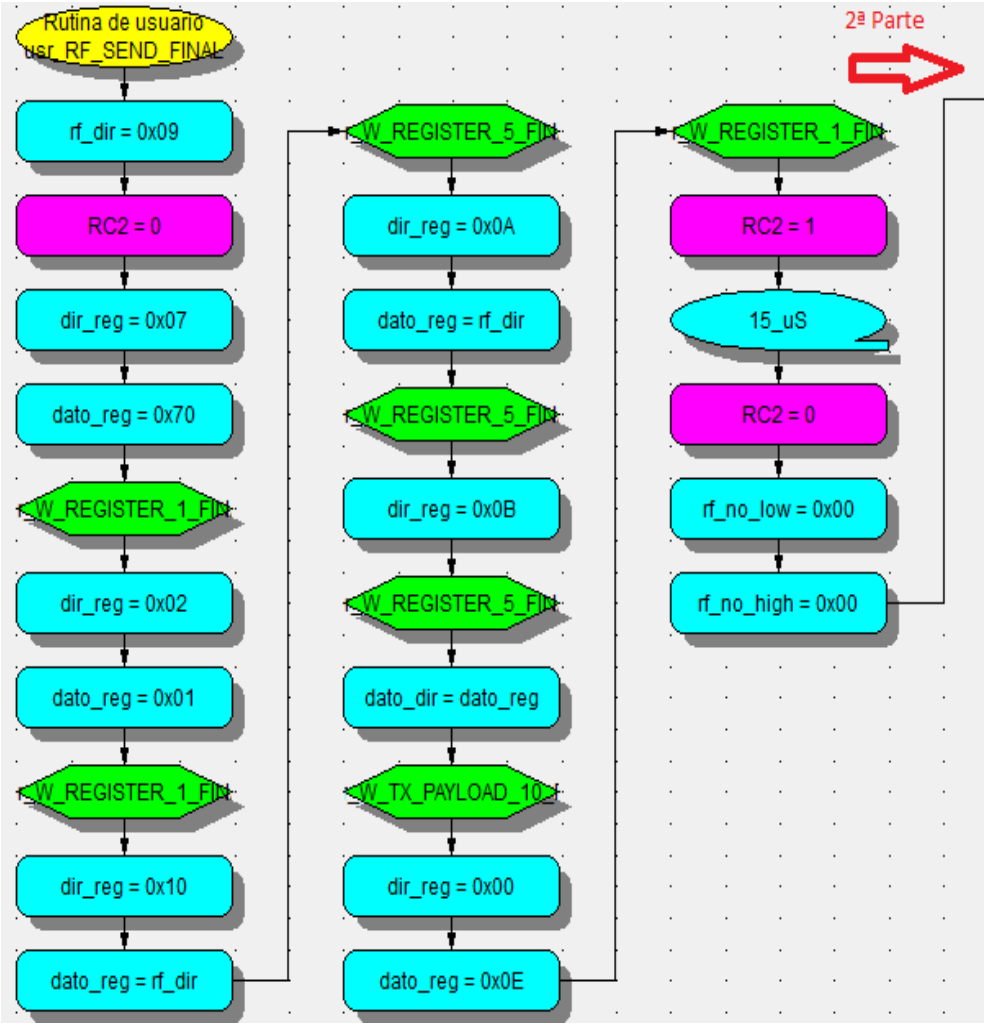


Fig 71. Esquema RF_SEND 1ªParte

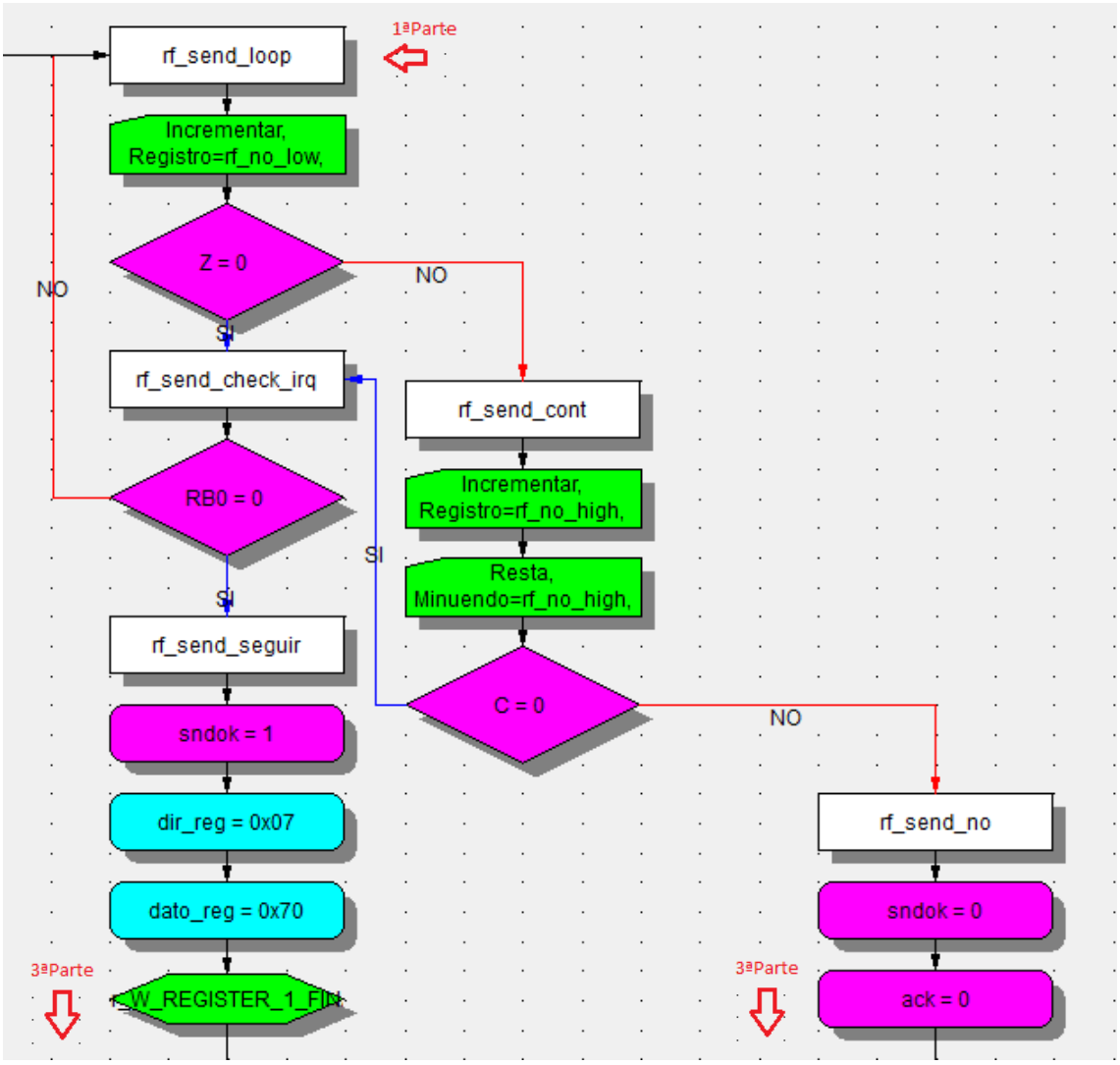


Fig 71. Esquema RF_SEND 2ªParte

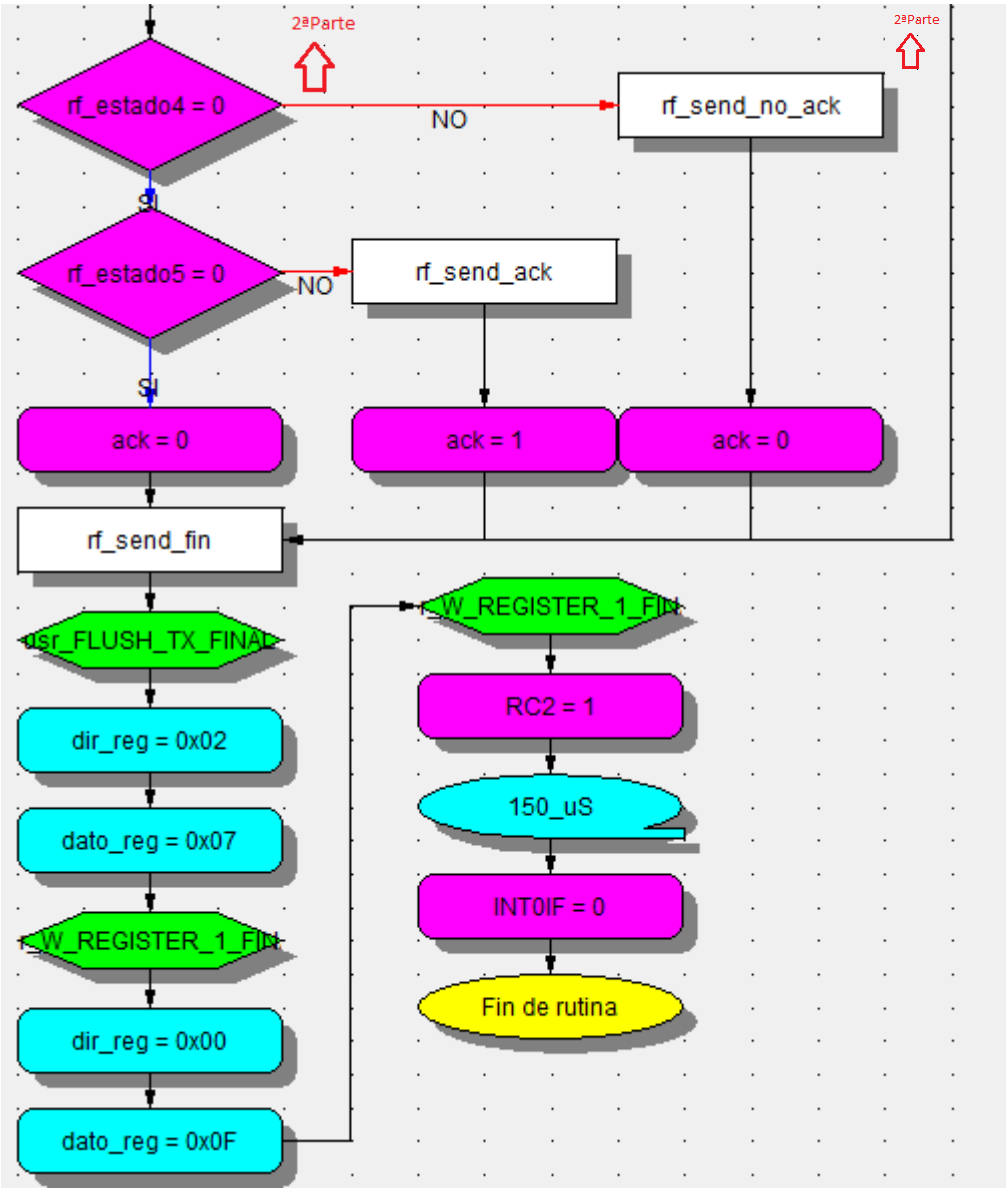


Fig 72. Esquema RF_SEND 3ªParte

7.1.4.5. *RF_RECEIVE*

Esta subrutina funciona por interrupción, por ello se encuentra ubicada dentro de la subrutina de la misma, por lo que hay que activar las interrupciones y configurarlas antes del desarrollo de esta subrutina.

Se encarga de comprobar si se ha producido una interrupción externa (RB0) y si es así, accede a la subrutina de interrupción, donde se encuentra RF_RECEIVE.

Cuando se reciba una trama se debe hacer una comprobación del bit RCVNW de la variable RF_STATUS y si está activo se debe llamar a la función RF_RECEIVE de nuevo tras tratas los datos. El transceptor tiene una pila de tres niveles, por lo que si no se llama a la función recibir antes de que se llene la pila, el dispositivo no podrá recibir más datos.

En esta subrutina se encuentran los registros de RF_DIR (en este caso muestra la ID del módulo transmisor) y los registros de 1 byte recibidos (RFI_DATA_0 – RFI_DATA_7), así como los bit correspondientes al registro de control RF_STATUS, los cuales son RCVOK(informa que se ha recibido los datos correctamente y están accesibles para ser tratados) y RCVNW (el cual muestra si todavía quedan datos por leer).

Como se ha explicado previamente, la rutina de recepción de ratos ha de colocarse dentro de la interrupción por RB0, como se observa en la siguiente imagen:

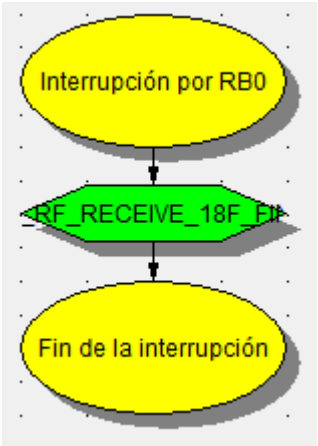


Fig 73. Recepción por interrupción en RB0

A continuación, se mostrará el esquema completo de está subrutina, el cual se va a dividir en varias partes debido a sus dimensiones en el plano:

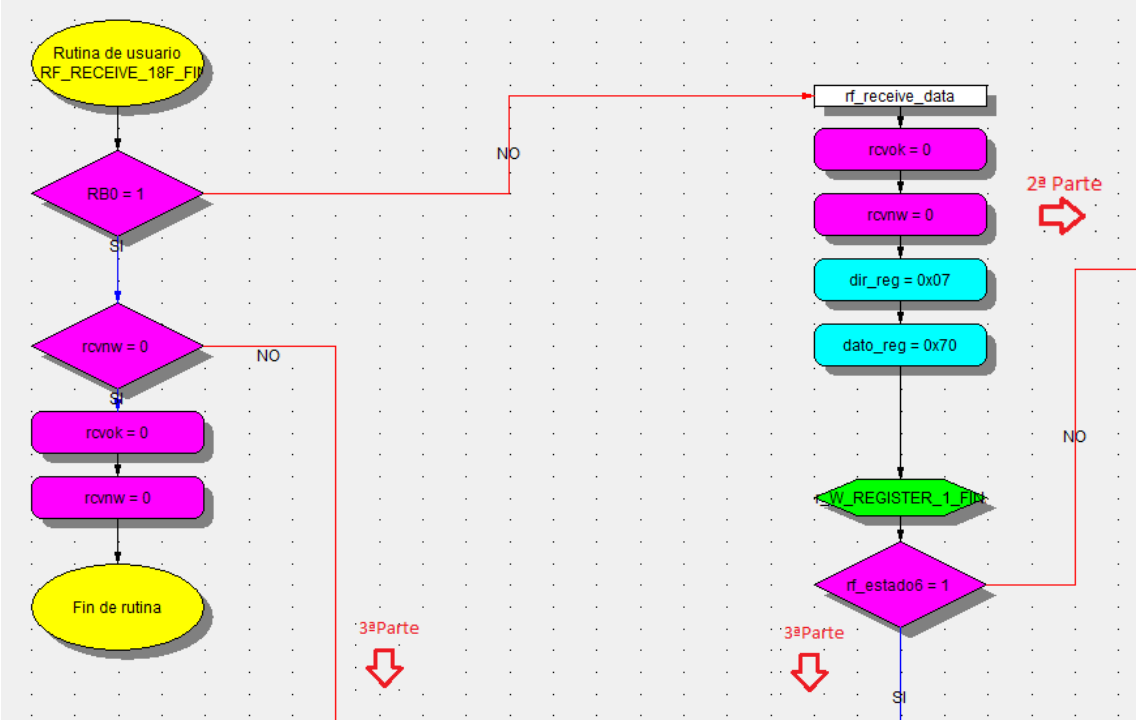


Fig 74. Esquema RF_RECEIVE 1ªParte

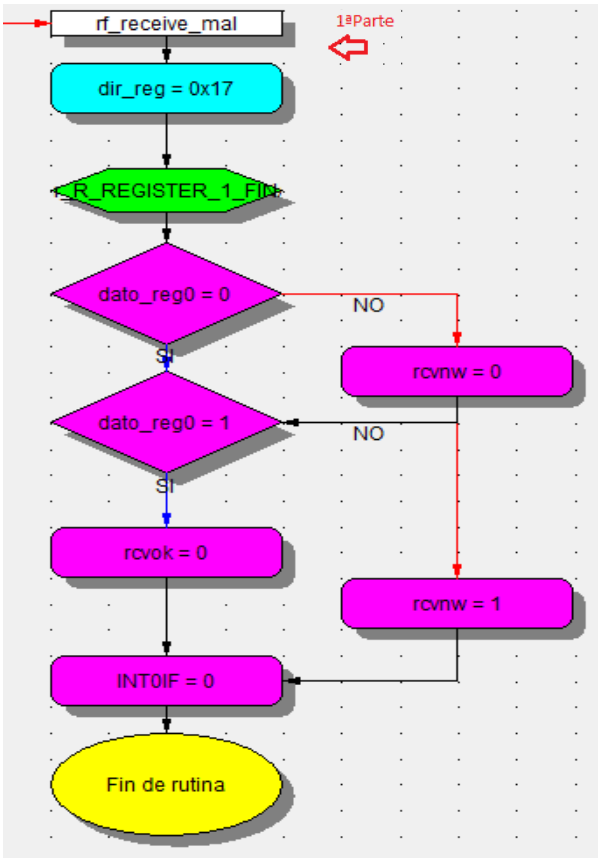


Fig 75. Esquema RF_RECEIVE 2ª Parte

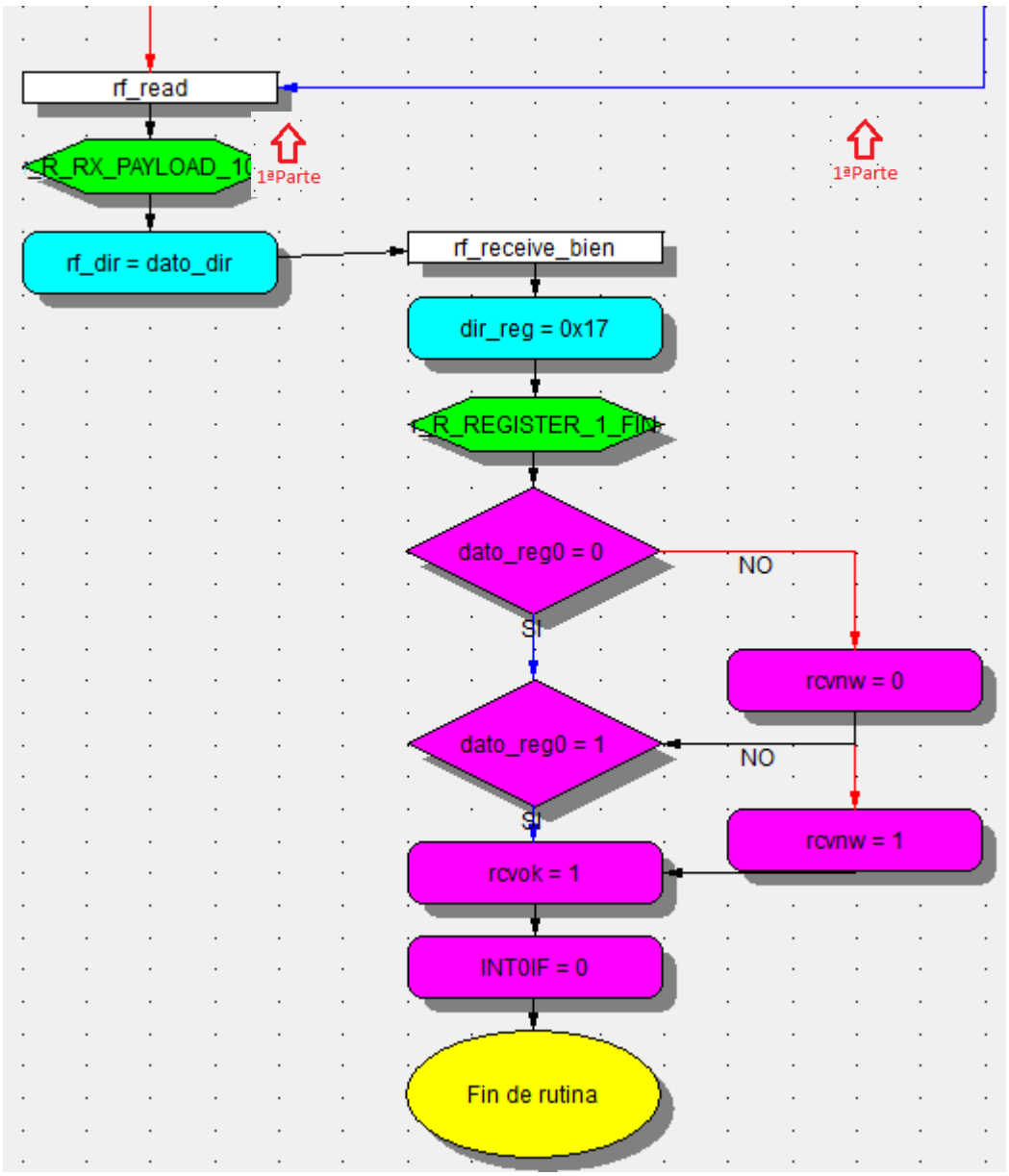


Fig 76. Esquema RF_RECEIVE 3ª Parte

7.1.4.6. *RF_OFF*

Esta subrutina desactiva el módulo de radio dejándolo en modo de bajo consumo sin borrar la configuración previamente establecida.

Esta subrutina no ha sido implantada en el proyecto, de momento, debido a la poca utilidad que tiene en el momento de pruebas de funcionamiento del proyecto, ya que simplemente se cortaba la corriente que alimentaba el sistema en el momento que no se quisiera continuar transmitiendo datos.

No se descarta, en un futuro, incluirla como tema de mejora del vehículo.

7.1.5. Comunicación TX/RX transmitiendo valores de un potenciómetro.

Una vez configurado los puertos, creada y activada la interrupción por flanco descendente de RB0, declarada la pantalla LCD e inicializada, declarado los registros que se indicaban en la librería del módulo NRF24L01 y desarrolladas las subrutinas necesarias para poder llevar a cabo una transmisión/recepción de información, es hora de transmitir datos, para lo cual se utilizará el valor obtenido de dos potenciómetros.

Antes de preparar el montaje en los dos entrenadores PIC'Kit SCHOOL, con los que se va a llevar a cabo la prueba, es necesario diseñar el programa en el software Niple.

Previamente, el desarrollo del programa, con todos los registros y subrutinas incluidas, se había quedado en la siguiente etapa:

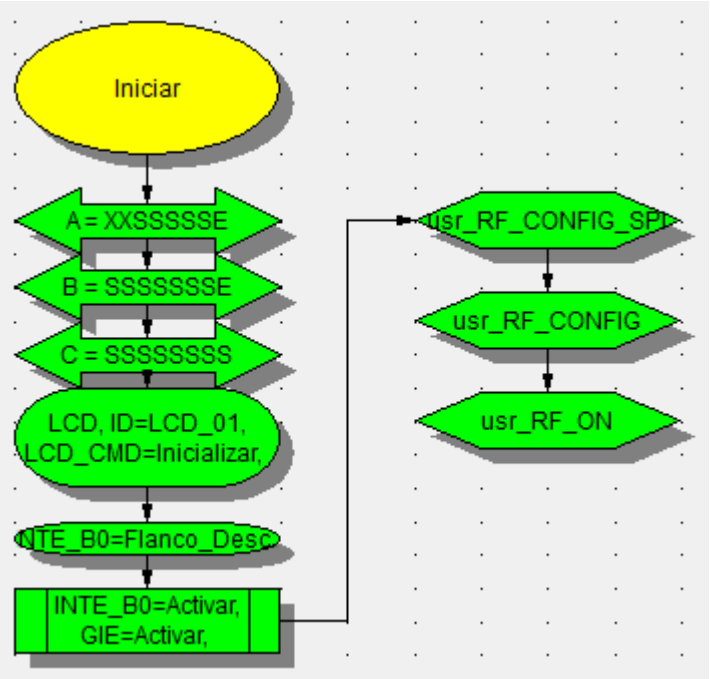


Fig 77. Programa con subrutinas principales implementadas

En primer lugar, se van a establecer los parametros de RF_CHN y RF_DIR en la subrutina RF_CONFIG, con lo que se pretende determinar el canal por el cual se va a realizar la comunicación entre módulos, así como la asignación de una ID a cada módulo, a través del registro RF_DIR.

Como se van a utilizar dos módulos, a partir de ahora se va a referir a cada uno de ellos como ‘’MÓDULO DERECHO’’ y “MÓDULO IZQUIERDO”.

Así pues, se procede a la selección del canal común de comunicación, el cual será el Canal 22 (0x22 → 2.422Ghz).

Una vez decidido cual será el canal de comunicación, se decide que ID tendrá cada uno de los módulos; el módulo izquierdo tendrá como ID el número Hexadecimal 0x09 y el módulo derecho el número Hexadecimal 0x0A.

De tal maneral, la subrutina RF_CONFIG donde se determina el valor de estos dos parámetros, quedaría de la siguiente manera:

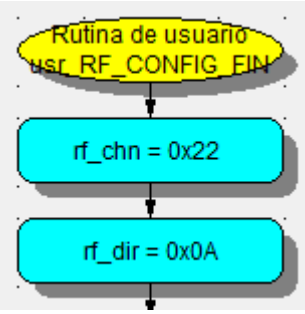


Fig 78. RF_CONFIG. Módulo Izq.

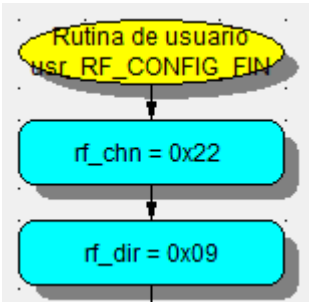


Fig79. RF_CONFIG. Módulo Dch.

Una vez configurada la subrutina RF_CONFIG, ya se tiene configurado el canal de comunicación y la ID de cada módulo. El siguiente paso será determinar en la subrutina RF_SEND, el destinatario de la transmisión:

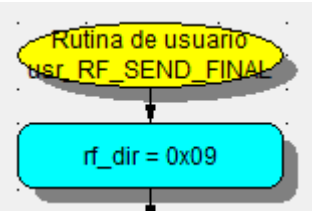


Fig 80. RF_SEND. Módulo Izq.

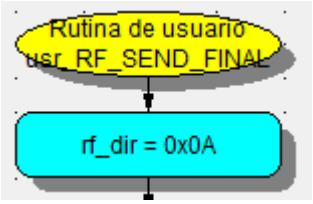


Fig 81. RF_SEND. Módulo Dch.

En la subrutina RF_RECEIVE no es necesario actuar sobre el valor de ningún registro, ya que en ella se recopilan los datos recibidos e identifica el ID del módulo transmisor.

Una vez configurado lo anterior, el siguiente paso sería incluir en el programa las instrucciones necesarias para recopilar del entrenador, donde se van a instalar todos los dispositivos, la información por parte del potenciómetro.

En primer lugar, es necesario configurar una entrada de uno de los puertos del PIC, como entrada Analógico/Digital, para que el programa que se grabará en el PIC, pueda interpretar la variación de tensión que será transmitida por el potenciómetro. Por lo que sencillamente, seleccionamos el pin RA0 del Puerto A:

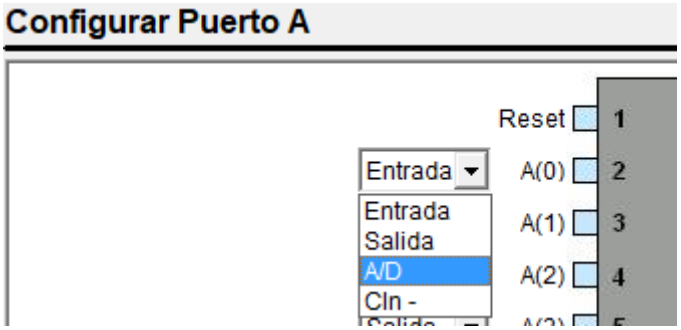


Fig 82

Seleccionado ya el pin del PIC que va a recavar la información transmitida por parte del potenciómetro, es necesario declarar un registro en el que se guarde esa información. Ese registro se llamará “Potenciómetro”:

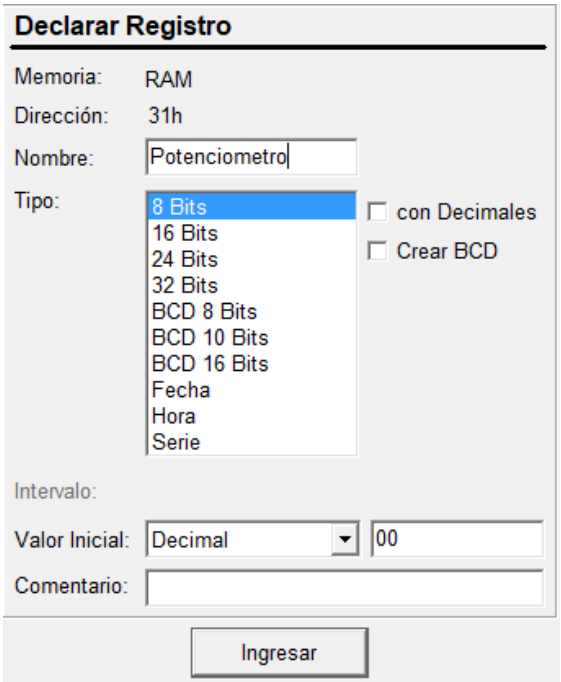


Fig 83

Como ya se ha dicho, la información que se vá a recibir es analógica y es necesario tratarla como datos digitales, por lo que se utiliza un conversor Analógico/Digital que poseé el Software Niple.

En el menú de la derecha de la pantalla principal, donde se encuentran todas las instrucciones con las que cuenta el software, se observa una pestaña con el nombre "CONVERSIÓN A/D". Una vez seleccionado, aparece una ventana emergente en la que se ha de seleccionar el Canal a Leer (Pin del PIC que se ha configurado como entrada A/D) y el registro en el que ha de guardarse el resultado (El registro seleccionado será el creado previamente con el nombre "Potenciometro"):

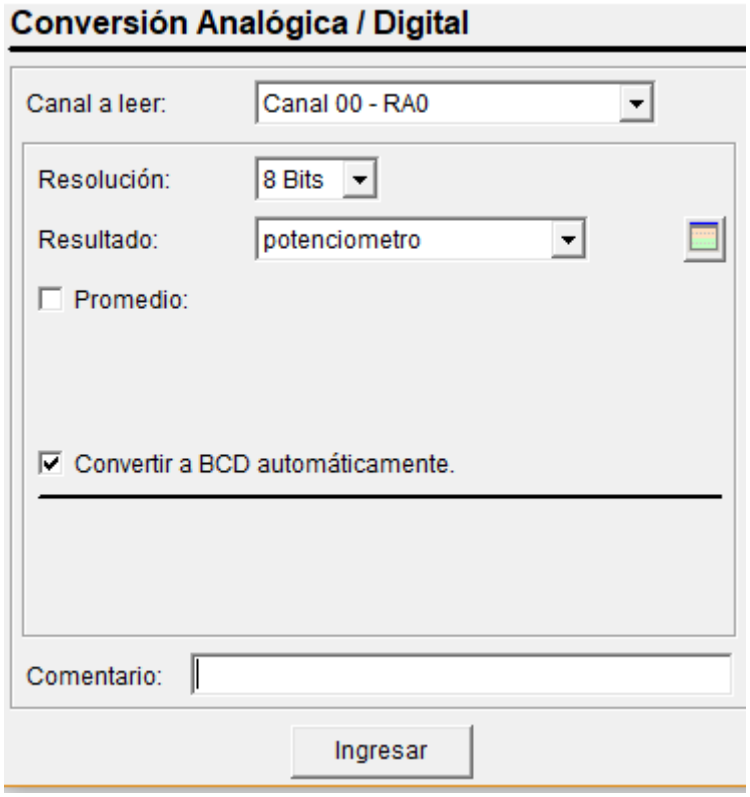


Fig 84. Configuración Conversión Analógica/Digital

En la imagen se puede observar que se ha seleccionado la opción "Convertir a BCD automáticamente". Esta opción ofrece repartir el valor que se lea a traves del canal seleccionado en tres registros diferentes, unidades, decenas y centenas. Estos registros los crea automáticamente, y serán "potenciometro_uni" , "potenciometro_dec" y "potenciometro_cen". Posteriormente se verá que estos tres registros son los que se utilizarán para poder visualizar los datos en la pantalla LCD, a la hora de programarla en el software.

Una vez pulsado el botón “Ingresar“, en el programa principal se puede observar que se incluye la función previamente descrita.

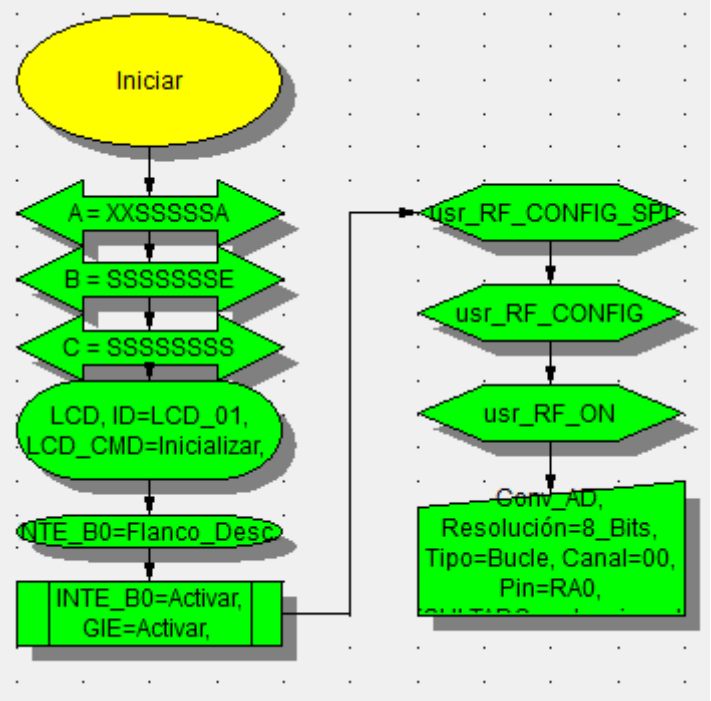


Fig 85

Ahora que ya se ha configurado el canal de comunicación a utilizar, los ID’s de los respectivos módulos y cómo obtener los datos que se pretenden enviar, es hora de asignar a uno de los 8 bytes que el módulo NRF24L01 envía, el valor obtenído por RA0 del potenciómetro.

En el momento de una comunicación, el módulo utiliza 16 bytes diferentes para el envío y recepción de datos. Para la transmisión, utiliza 8 (RFO_DATA_0 – RFO_DATA_7) y para la recepción otros 8 (RFI_DATA_0 – RFI_DATA_7). Los “RFO” son los datos que se envian “OUT” y los “RFI” los que se reciben “INPUT”.

Como el valor del potenciometro se recoge en un registro de 1 byte (8bits) solo es necesario uno de los registros de salida para enviar la información, por lo que tomamos, por ejemplo, RFO_DATA_0 para enviar la información y RFI_DATA_0 para recibirla.

Para enviar la información guardada en el registro “potenciometro” a través de el registro “RFO_DATA_0” simplemente se le asigna el valor del registro “potenciometro” al registro “RFO_DATA_0”, utilizando una de las funciones del menú lateral derecho:

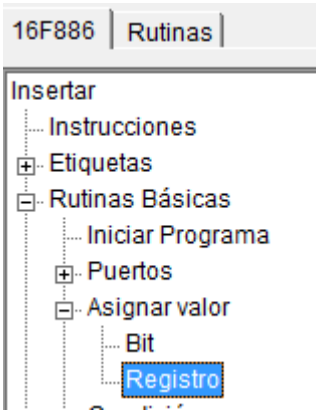


Fig 86

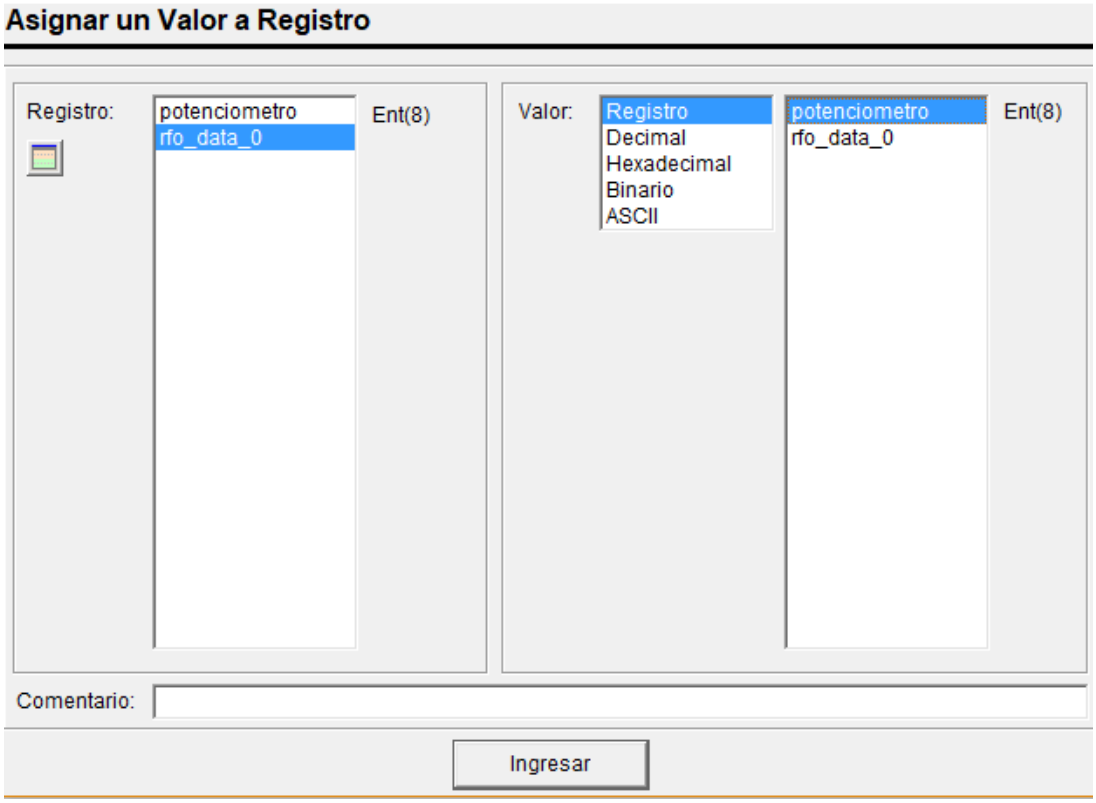


Fig 87. Asignación del valor de un registro, a otro registro

Una vez asignado el valor del potenciómetro al registro de salida RFO_DATA_0, el programa principal estaría acabado para poder transmitir los datos, por lo que se le añade, después de el último paso seguido, la subrutina de RF_SEND y un pequeño temporizador para tener una pequeña pausa (a nuestra percepción no se nota) entre transmisión y transmisión. Se termina el programa uniendo el último bloque de instrucciones con la lectura del pin RB0, para crear un bucle de lectura/transmisión continua.

Este programa se ha hecho para poder tener una transmisitir y recibir al mismo tiempo, por lo que en la subrutina creada con la interrupción en RB0, se encuentra la subrutina RF_RECEIVE.

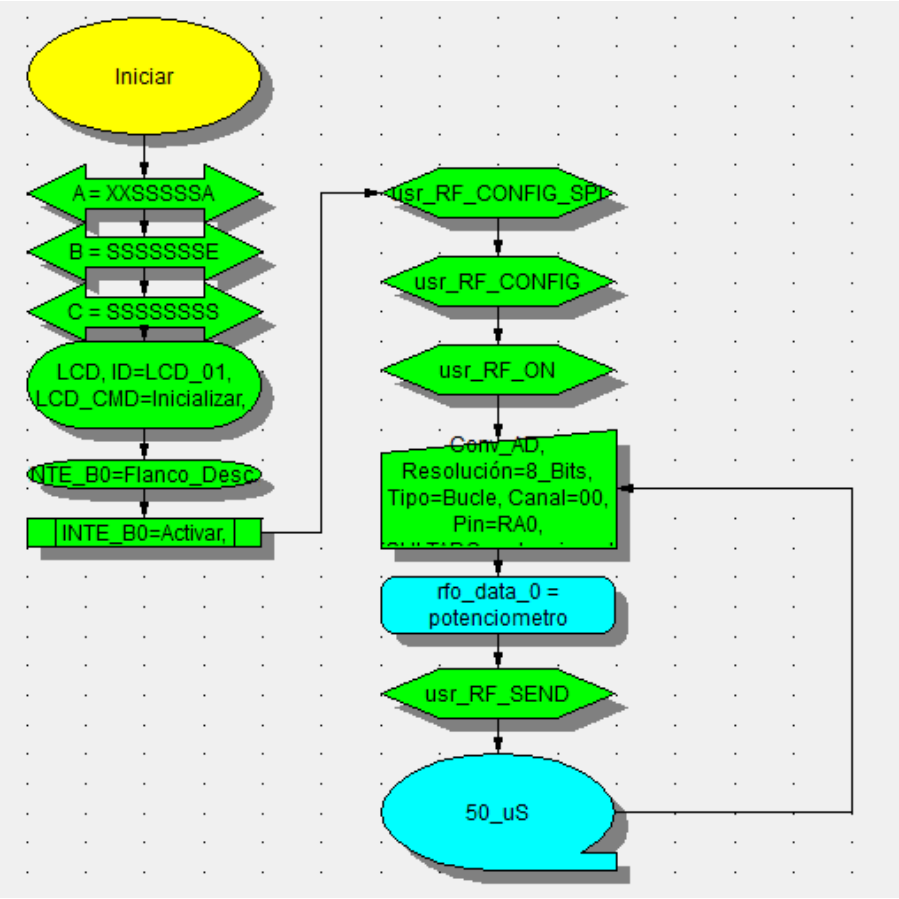


Fig 88. Programa para transmitir los datos de un potenciometro

Para poder visualizar los datos, tanto que se envian como que se reciben, es necesario crear un mensaje en la pantalla LCD previamente inicializada. Para ello, en el menú del lateral derecho, en el apartado de pantalla LCD, se selecciona la opción “Nuevo Mensaje “. Lo primero que hay que hacer, es darle un nombre al mensaje, en este caso “Valores” y acto seguido, elegir las dimensiones de la pantalla. El entrenador que se utiliza para esta comunicación (ambos) poseen integrada una pantalla LED de 16x2, por lo que es la que se selecciona.

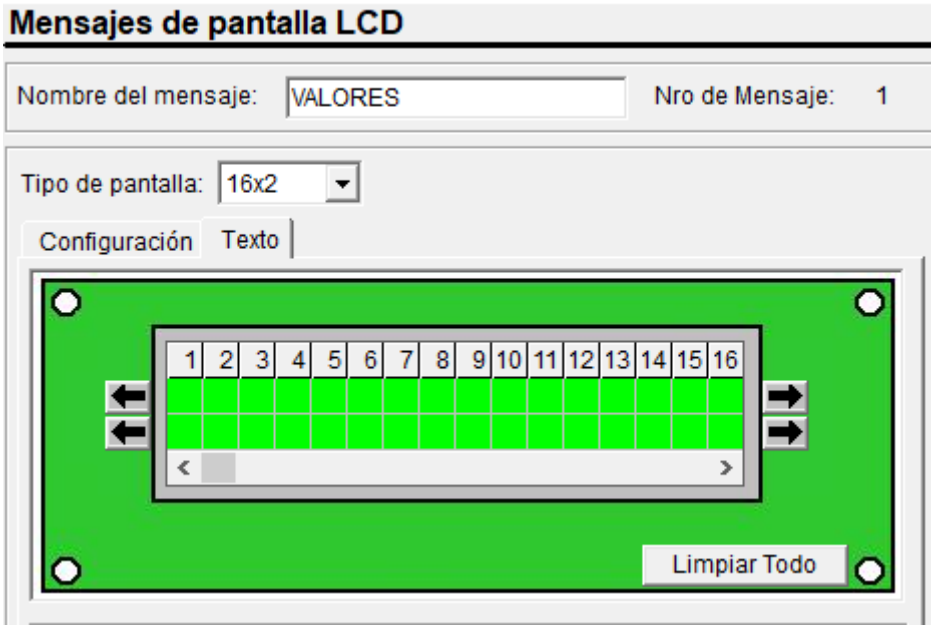


Fig 89. Declaración de mensaje en pantalla LCD

Como se observa en la imagen, hay dos filas de 16 caractéres cada una. Si están en verde, significa que están libres para introducir un caractér. En este caso, se introduciran dos indicaciones “Valor enviado” y “Valor recibido”, seguido de los registros (en BCD) de “RFO_DATA_0” para el primero y “RFI_DATA_0” para el segundo, quedando tal que así:

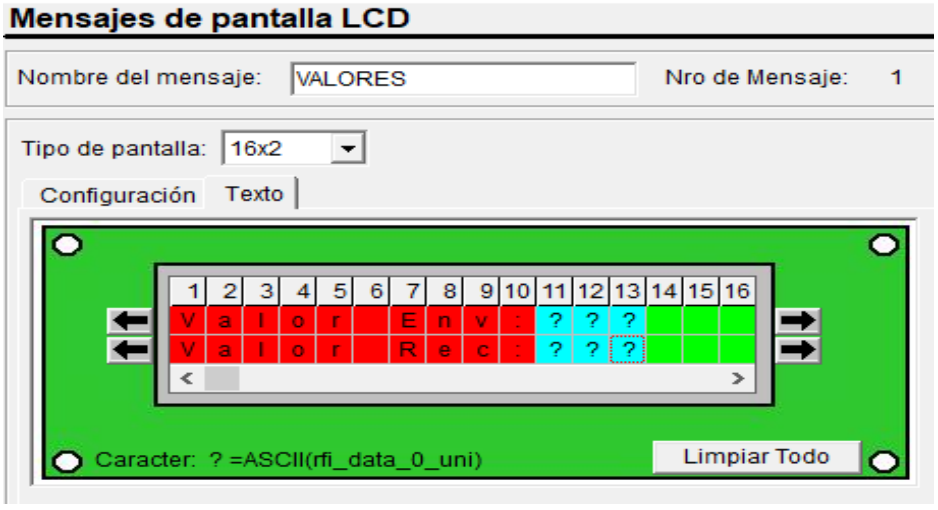


Fig 90. Visualización de los valores de los registros en la pantalla LCD

De manera, que el programa final para comprobar el funcionamiento de la comunicación entre los dos módulos quedaría de la siguiente manera:

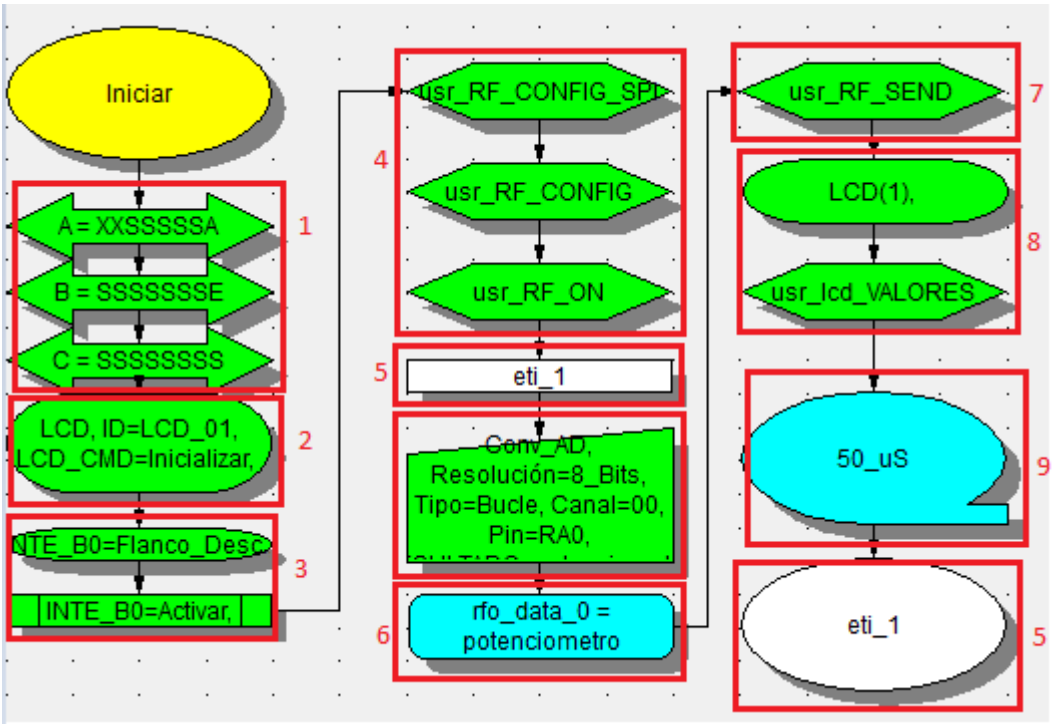


Fig 91. Programa final comunicación entre módulos

1 : Configuración de los Puertos del PIC

2: Inicialización pantalla LCD

3: Configuración y activación Interrupción

4: Rutinas de configuración del módulo NRF24L01

5: Etiqueta que te permite ir a otro punto del programa sin necesidad de nexos de unión.

6: Asignación de el valor de un registro a otro.

7: Rutina de envio de datos.

Hay que constatar, que este programa no se obtuvo con el primer intento de llevarlo a cabo, y no por el hecho de la complejidad del mismo, sino por la dificultad de traducir la librería en mnemónico del módulo NRF2L01 al software Niple.

Esta tarea ha sido la que más problemas ha dado, ya que a la hora de introducir los datos en el software, en el momento de introducir uno erróneamente, la comunicación no se realiza. Esto supone la revisión, subrutina por subrutina, de toda la librería traducida al software.

Así mismo, la importancia de configurar los puertos correctamente es esencial. En dos ocasiones se consideró que había un problema de código dentro del programa creado para la comunicación, cuando en realidad se trataba de una mala configuración de un PIN del PIC, el cual, debería haber sido configurado como entrada y no como salida, ya que se trataba del MOSI.

El lograr la comunicación entre los dos módulos ha resultado la parte más costosa del proyecto, hasta el momento.

Una vez conseguido el montaje final de la comunicación de los dos módulos, transmitiendose los valores de los respectivos potenciómetros, se procede a pasar al siguiente paso, transmitir, además del valor del potenciómetro, los valores de un JOYSTICK, con el cual se controlará la dirección del vehículo y la posición de los servomotores, para hacer girar las ruedas en una dirección u otra.

Fig 92. PIC'SCHOOL utilizados en las pruebas de funcionamiento

7.1.6. Transmisión de los datos del Joystick. Control motores y servomotores.

7.1.6.1. JOYSTICK

Antes de comenzar a programar un nuevo dispositivo, es necesario conocer el funcionamiento de este, por muy sencillo que sea.

El Joystick consta de 5 pines:

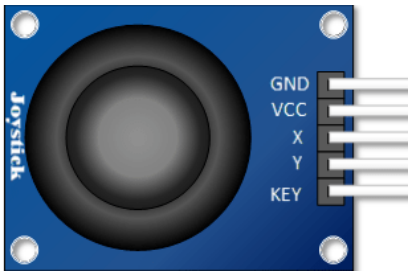


Fig 93. Joystick

GND y VCC, conectados a masa y tensión respectivamente, mientras que X e Y son las dos salidas de lass dos resistencias variables que van a proporcionar los valores, mientras que el último pin, KEY, es un pulsador, el cual no es utilizado en el proyecto.

Los pines X e Y funcionan de la siguiente manera:

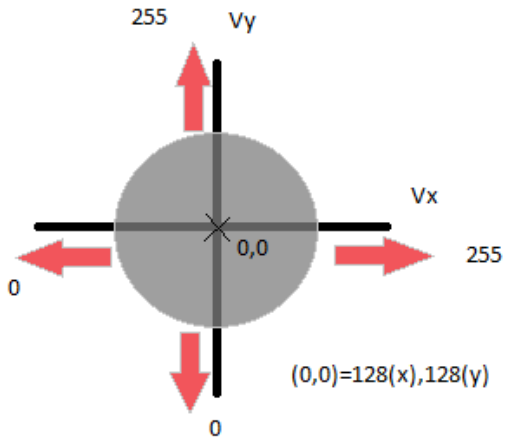


Fig 94. Valores Joystick en función de la posición

Para el EJE Y, el movimiento hacia la parte superior hace reducir la resistencia variable a “0” por lo que el valor que transmitirá será el máximo (255), mientras que si se desplaza a la parte inferior, la resistencia adquiere el valor máximo, por lo que el valor seríael mínimo (0).

En el EJE X, el movimiento lateral hacia la derecha reduce la resistencia variable a “0”, por lo que el valor que transmitirá será el máximo (255), mientras que si se desplaza a la parte lateral izquierda, la resistencia adquiere el valor máximo, por lo que el valor sería el mínimo (0).

En la posición de reposo, señalada en la “FIGURA 93” como las coordenadas (0,0), los valores de ambos ejes rondan el valor medio, 128, con pequeñas variaciones debido a la calidad del producto.

A partir de este momento, comprobado el funcionamiento de la transmisión/recepción de datos, se comenzará a especializar cada uno de los dos PIC’SCHOOL kit, en la tarea que van a desempeñar, uno como módulo receptor (Maqueta del vehículo) y otro como módulo emisor (Base fija de control). Como cada módulo dispondrá de su propio PIC, será necesario hacer dos programas de control con Niple, utilizando las subrutinas del NRF24L01 previamente echas como base del desarrollo de cada uno.

Será en la base fija donde se instalarán todos los dispositivos de control, y se comenzará, con este que se está desarrollando.

En primer lugar, será necesario crear unos registros para los datos que se obtengan del EJE X y EJE Y del Joystick:

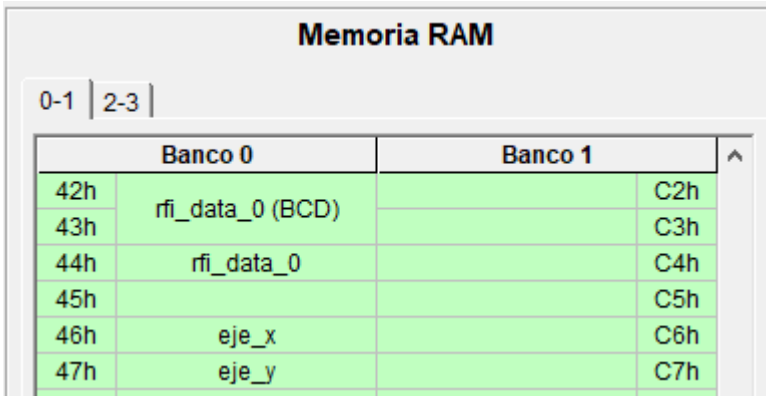


Fig 94. Banco de Memoria RAM

Al igual que se hizo con el potenciómetro, es necesario habilitar dos pines del Puerto A como entradas Analógica/Digital para poder transmitir los datos al PIC:

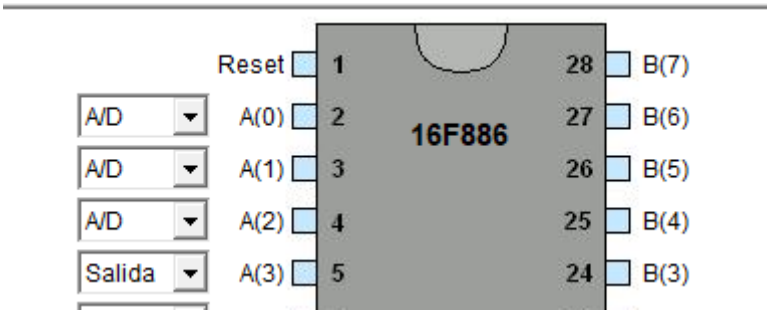


Fig 95.

Una configurado los dos pines a los que se conectarán las salidas del JOYSTICk (EJE X y EJE Y), será necesario configurar una lectura de los dos pines (RA1 y RA2) para poder guardar los valores recibidos en los registros recientemente creados (eje X y eje Y).

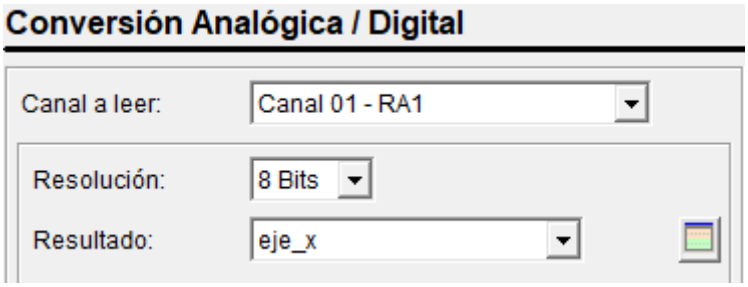


Fig 96. Conversión valor EJE_X

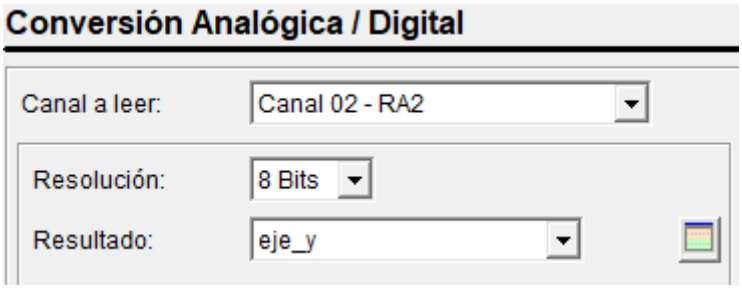


Fig 97. Conversión valor EJE_Y

Una vez seleccionado el canal de lectura y el registro donde se guardarán los datos (se selecciona la opción de "Convertir a BCD automáticamente" para la visualización de los datos en la pantalla), será necesario asignar el valor de estos dos registros a dos byte de transmisión (RFO_DATA_1 y RF0_DATA_2), para poder enviarlos al módulo receptor:

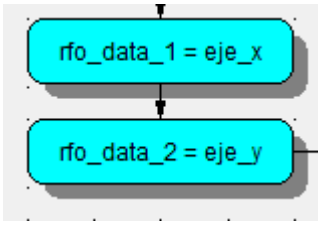


Fig 98. Asignación a los registros de salida el valor de los ejes

Para evitar un tamaño excesivo del programa principal, se decide crear una subrutina con el nombre de "VALORES" en la cual se incluirán toda la adquisición de datos por parte de los diferentes dispositivos, así como la interpretación de los datos que transmitan. Por ahora, dentro de esta, solo se encontrará el JOYSTICK, hasta que se añadan los demás:

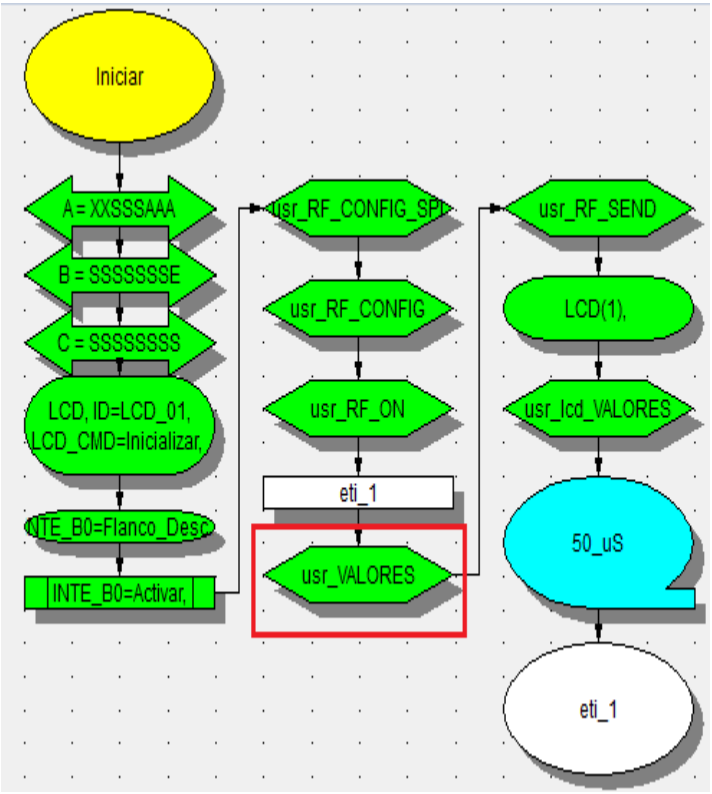


Fig 99. Introducción subrutina de recogida de datos

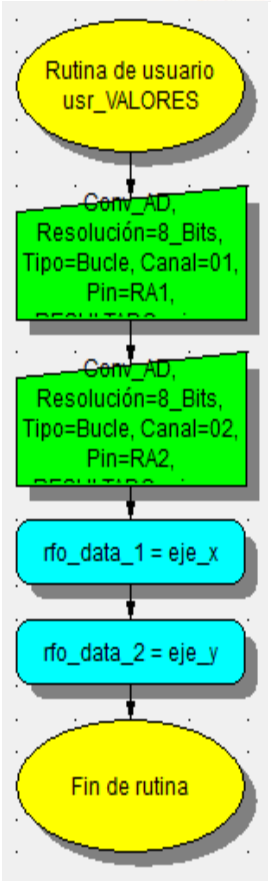


Fig 100. Subrutina usr_VALORES

Una vez hecho el programa para poder interpretar los datos que se reciben del Joystick, se configura la pantalla LCD para que muestren estos, y comprobar, que los valores que muestra la pantalla LCD del transmisor, son los mismos que muestra la del receptor, por lo que, al igual que en el caso del potenciómetro, asignamos los registros correspondientes a las casillas de la LCD:

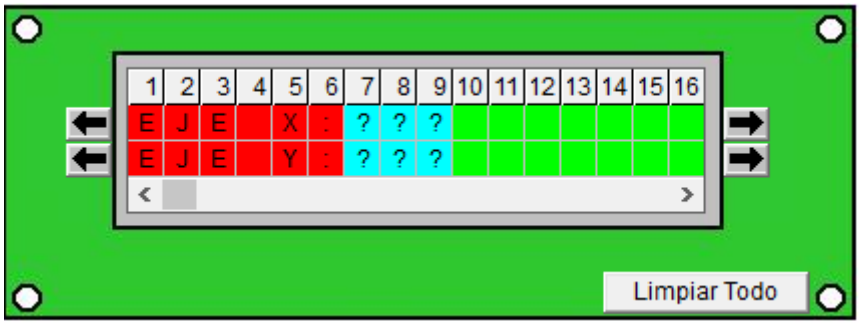


Fig 101. Pantalla LCD con el valor de los ejes

7.1.6.2. JOYSTICK. CONTROL MOTORES

Tras haber configurado y comprobado el buen funcionamiento del Joystick, el siguiente paso a seguir en el desarrollo del proyecto, es controlar, de manera simple, el movimiento de los motores que se van a utilizar con la maqueta del vehículo.

Para controlar los motores de la maqueta, es necesario utilizar el Driver L298, el cual permite gestionar dos motores de corriente continua y motores paso a paso de no más de 2 amperios. En la siguiente imagen se pueden observar las partes de este componente:

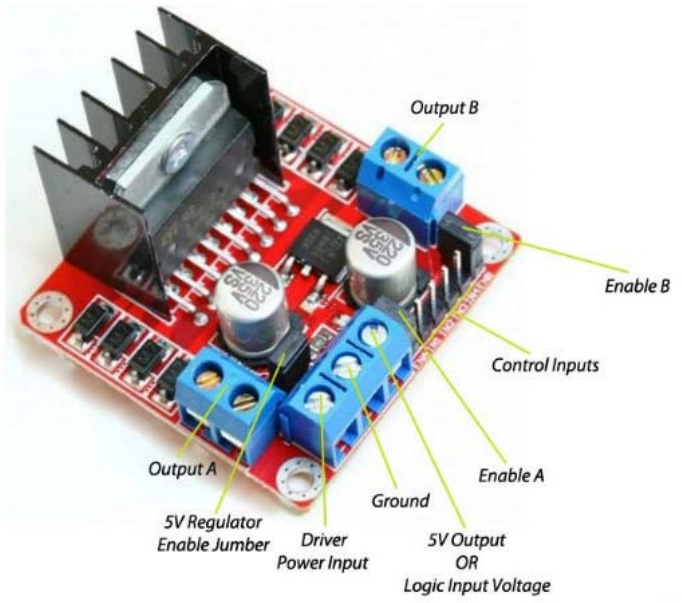


Fig 102. Módulo de control de los motores

Como se puede observar en la imagen, el módulo cuenta con lo siguiente:

8. Regulador de voltaje LM7805
9. Conectores Output A y Output B que son las salidas de los motores.
10. Control inputs, son los terminales de control, dos de ellos son los pines de habitilación de cada motor.
11. Jumper para decidir si se va a utilizar el regulador LM7805 o no.

Hay que tener presente el voltaje con el cual se va a alimentar, si bien el módulo tiene un regulador de tensión y es capaz de soportar tensiones de hasta 36V, también tiene un Jumper que permite decidir si se usa o no.

Si se va a alimentar el módulo con la misma tensión a la que se alimenta el PIC, no es necesario utilizar el regulador, ya que el Driver L298 tolera tensiones de entre 6 y 12V.

En el caso que se alimente el módulo con tensiones superiores a 12V, se ha de conectar el jumper para que tulice el regulador, de lo contrario se puede quemar en cuestión de segundos.

En el caso de este proyecto, se utilizará el jumper, debido a la tensión utilizada para controlar la maqueta del vehículo.

Cada motor, está conectado a una de las salidas del Driver (Motor Izq → Output A / Motor Dch → Output B) y son controlados a partir de los datos procedentes del PIC, que entran a través de las INPUTS (cada motor tiene dos entradas, para poder variar el sentido de giro de este).

Para seleccionar el tipo de movimiento que se va a realizar, sencillamente se hace una tabla, que posteriormente, se programará dentro del software del módulo receptor.

Los pines del PIC que controlaran la dirección de giro de los motores, serán los siguientes:

Motor Izquierdo : RD0 y RD1
Motor Derecho : RD2 y RD3

A ESTA ALTURA DEL PROYECTO, DEBIDO A LA CANTIDAD DE DISPOSITIVOS QUE SE IBAN A INTEGRAR EN EL PROGRAMA Y EL TAMAÑO QUE ESTÁ ADQUIRIENDO ESTE, SE DECIDIÓ BUSCAR UN PIC CON UNAS CARACTERÍSTICAS MÁS ÓPTIMAS (MÁS MEMORIA Y PUERTOS), POR LO QUE SE OPTÓ POR EL PIC 18F4620, UN PIC CON UNAS CARACTERÍSTICAS MUY SUPERIORES A LAS DEL 16F886 EN CUANTO A MEMORIA RAM, Y CON UN PUERTO MÁS A UTILIZAR PARA E/S (PUERTO D).

Una vez seleccionados los pines del PIC que controlarán la dirección de giro de los motores, se hace una tabla donde se le dan unos determinados valores a los pines en función de la dirección que se quiera:

PIN	RD0	RD1	RD2	RD3
Delante	1	0	0	1
Atrás	0	1	1	0
Cntrol In.	IN1	IN2	IN3	IN4

Ahora que se han establecido los valores que se le ha de dar los diferentes pines seleccionados para que los motores de muevan en una dirección u otra, la idea ha de ser trasladada al programa.

Este parte, como es la encargada de controlar los motores, se realizará en el programa que se destinará al PIC de la maqueta del vehículo.

Como previamente se ha explicado, la decisión del tipo de movimiento vendrá dada por los valores del Joystick. En este caso, de los dos Ejes de los que consta, el Eje Y se encargará de controlar la dirección del vehículo, mientras que el Eje X controlará la posición de los servomotores (Se explicará en el siguiente apartado).

El EJE X consta de dos tipos de movimientos; Un movimiento vertical hacia arriba, el cual moverá la maqueta hacia delante, y un movimiento vertical hacia abajo, moviendolo hacia atrás.

Cuando el EJE X, está en la posición vertical superior, toma el valor máximo, 255, y 0 cuando está en la posición contraria, por lo que se utilizarán estos valores para comenzar en la programación del movimiento de los motores.

En primer lugar, los datos de los dos EJES, se habían enviado, por parte del módulo transmisor, a través de los registros RFO_DATA_1 (EJE X) y RFO_DATA_2 (EJE Y), por lo que en el módulo receptor, estos datos serán guardados en los registros RFI_DATA_1 (EJE X) y RFI_DATA_2 (EJE Y).

Para tener todo organizado, se crearan dos registros en el PIC del módulo receptor, con los nombres de EJE X y EJE Y, a los que se le asignará los valores recibidos por el módulo transmisor, a cada uno el suyo respectivamente.

Una vez hecho esto, el software Niple dispone de una instrucción que se conoce como “CONDICIÓN” y será la que se utilizará para poder programar la dirección de giro.

Esta instrucción te permite condicionar la dirección del programa. Esto significa, que la elección de un camino u otro de este, irá precedida por el cumplimiento de una condición establecida por el usuario, en este caso, el valor recibido del EJE Y.

Como el valor del EJE Y para que los motores se muevan hacia delante ha de ser 255, se le asignará una condición de la siguiente manera:

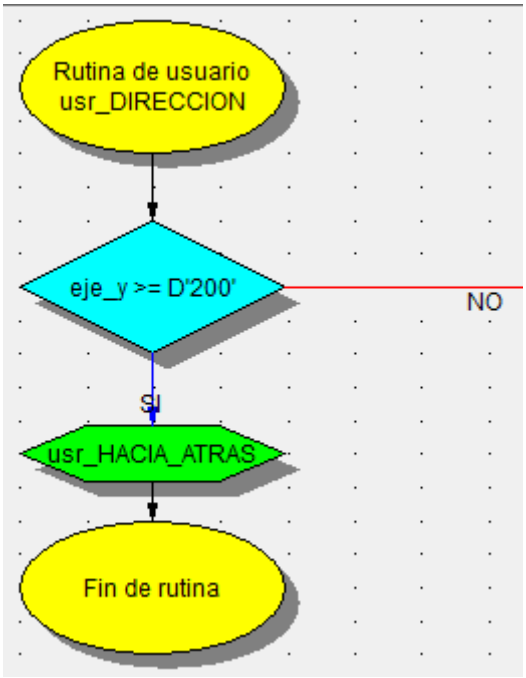


Fig 103. Condición para selección de movimiento

Si el valor del EJE Y es superior al decimal 200, el camino elegido por el PIC, será afirmativo, por lo que se dirigirá a realizar la subrutina denominada “HACIA ATRÁS”, por el contrario, si el valor es inferior al decimal 200, la condición dará un resultado negativo, por lo que tomará el camino opuesto:

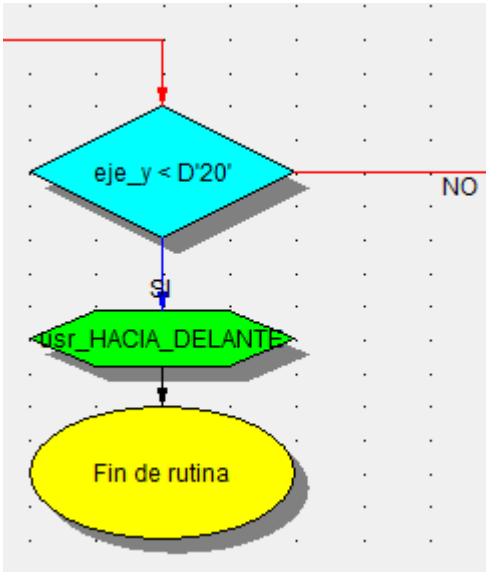


Fig 104. Condición para selección de movimiento.

Este camino opuesto es el que conduce hasta la siguiente condición, la cual es responsable de ejecutar el movimiento de los motores hacia delante, o por el contrario, tomar el otro camino de la condición.

Por lo que, en función de los valores recibidos del EJE Y, el vehículo realizará un movimiento u otro, si cumple con los valores establecidos para que ello pase (se han puesto cifras inferiores a los valores máximos debido a que en su posición máxima, el valor también oscila debido a la calidad del producto)

Si por el contrario, ninguna de estas dos condiciones se cumplen, el vehículo tenderá a no moverse en ninguna dirección, ya que se ha incluido una subrutina de "STOP", la cual asigna el valor "0" a cada bit de control de los dos motores.

La vista de la subrutina "DIRECCIÓN", así como la de "HACIA DELANTE", "HACIA ATRÁS" y "STOP" es la siguiente:

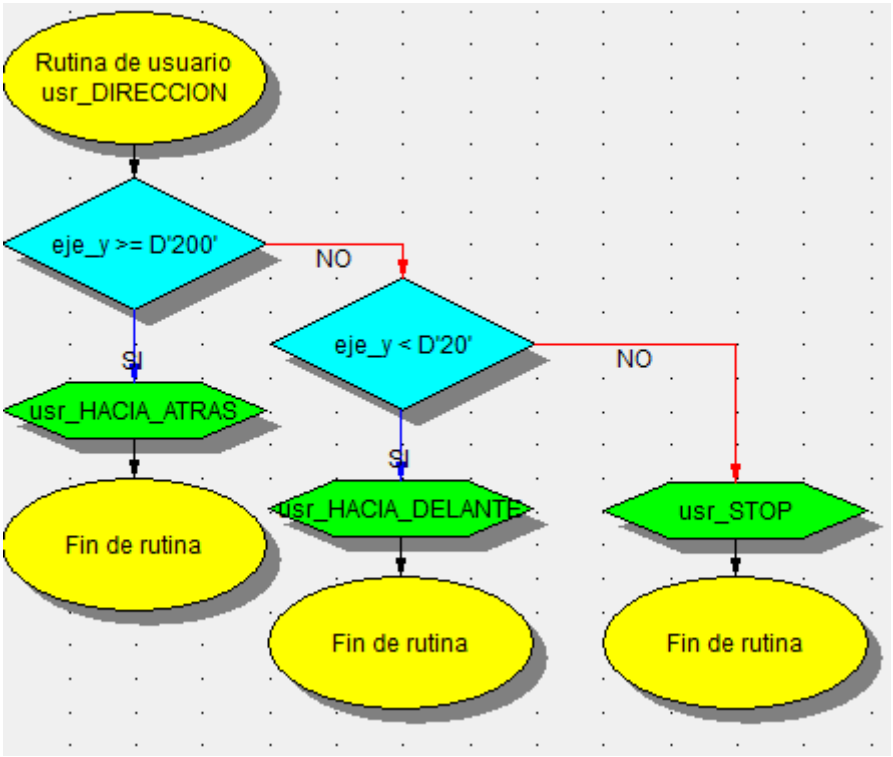
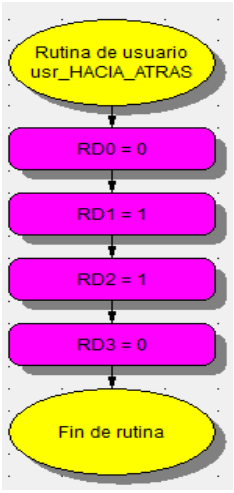
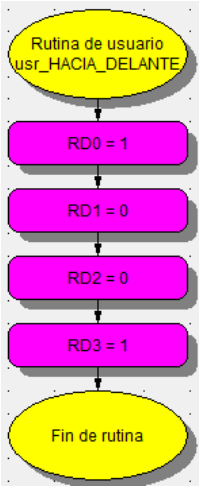


Fig 105. Rutina selección de dirección





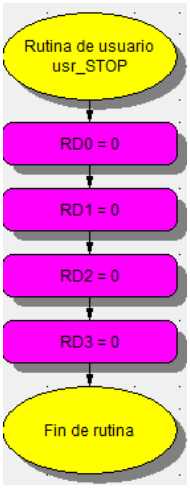


Fig 106. Hacia Atrás Fig 107. Hacia Delante Fig 108. STOP

Una vez conseguido el control de la dirección de giro de los motores, una de las características necesarias con la que ha de contar el vehículo, es el control de la velocidad de este, poder decidir si se mueve a una velocidad u otra.

Para esto, se utilizará un “SLIDER”. Este instrumento es una resistencia variable con el que se controlará la velocidad a la que se desplaza la maqueta, utilizando el valor máximo que ofrece (255) como velocidad máxima, y su valor mínimo (0) como velocidad mínima, en la que no hay desplazamiento.

El SLIDER consta de una salida, OTA, a través de la cual se obtendrá el valor en función del valor de la resistencia. Cuenta con un PIN VCC alimentado a con 5V y un PIN GND conectado a masa.

Al igual que con los dispositivos previos, los datos que va a proporcionar han de ser tratados a través de un PIN de lectura Analógico/Digital, por lo que se le asigna el PIN 2 del Puerto A (RA2). Los datos que se reciben, han de ser guardados en un registro nuevo, por lo que se crea un registro con nombre "SLIDER" de 1 byte de tamaño. Así mismo, el módulo emisor tendrá que enviar el valor de este al receptor, por lo que se le asigna uno de los ocho registros disponibles para transmitir datos, RFO_DATA_2, al que se le da el valor del registro "SLIDER".

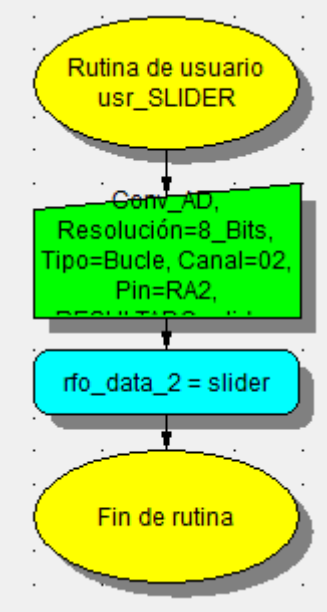


Fig 109. Conversión resistencia variable slider y asignación a un registro de salida

Configurado ya la obtención de datos por parte del SLIDER y el envío de estos a través del módulo NRF24L01, es necesario programar en el receptor una subrutina capaz de utilizar estos datos recibidos para el control de la velocidad del vehículo.

A continuación, se va a mostrar la subrutina realizada para el control de la velocidad del vehículo, finalizada, la cual se va a explicar por partes a continuación, así como en qué consiste su funcionamiento general:

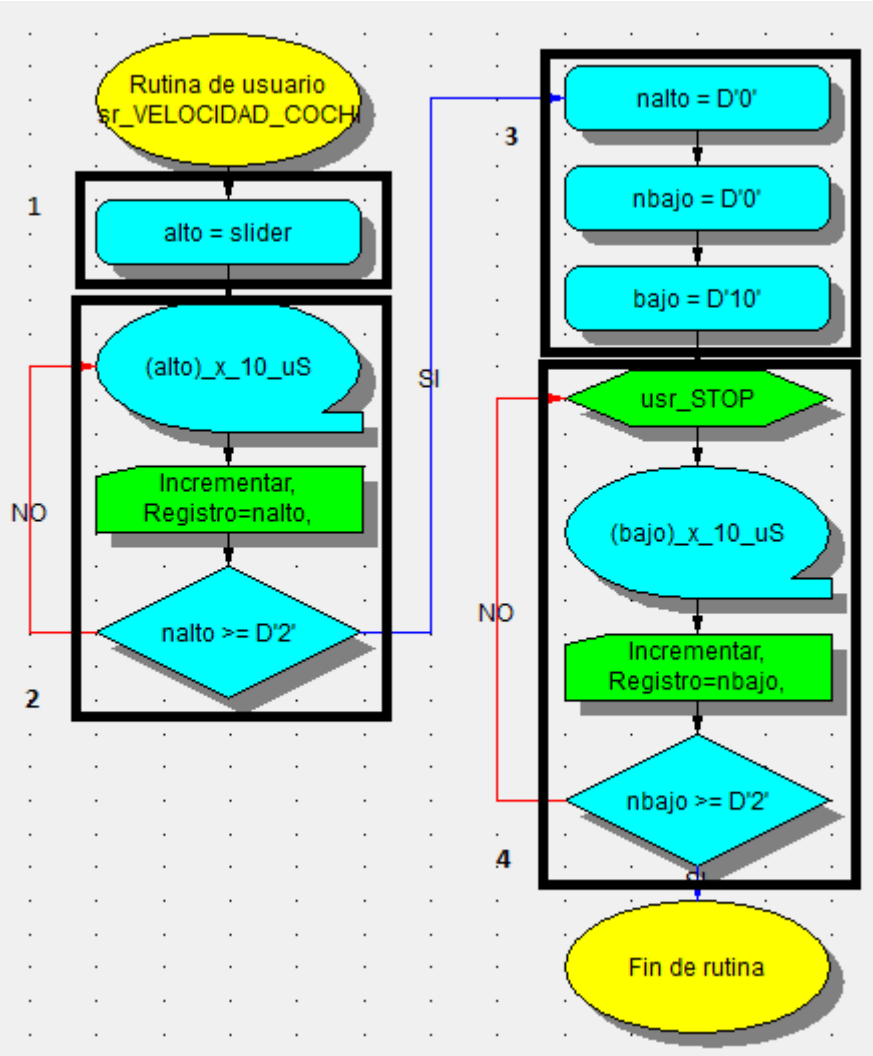


Fig 110. Rutina para el control de la velocidad del coche

1. ASIGNACIÓN VALOR SLIDER.

En primer lugar, se puede observar un registro, "ALTO" al cual se le ha asignado el valor del registro "SLIDER" (en el módulo receptor se creó otro registro SLIDER, al cual se le asignó el valor entrante del registro RFI_DATA_2, a través del cual se envió por parte del transmisor la información).

2 Y 4. CONTROL VELOCIDAD MOTORES.

Previamente, se ha visto como se le ha asignado el valor recibido del SLIDER, a un nuevo registro que se ha denominado "ALTO". Esto es debido a que se va a jugar con este registro, así como con su contrario "BAJO" para llegar a controlar la velocidad del vehículo.

La idea que se va a utilizar para conseguir controlar la velocidad, es la de una señal digital, que varía entre los valores "0" y "1". Los nuevos registros que se han creado, hacen la vez de estos valores, "ALTO" sería el valor "1" y "BAJO" el valor "0".

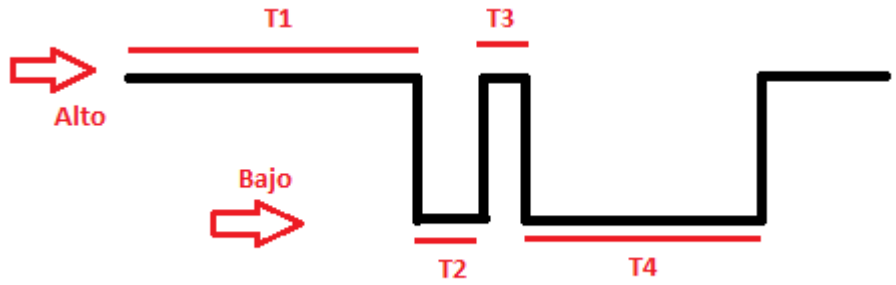


Fig 111. Funcionamiento del control de la velocidad de los motores

La velocidad del vehículo va a depender del tiempo que la señal se encuentre en la posición ALTO, cuanto más tiempo se encuentre en esta posición, más velocidad tendrá el vehículo. Como se puede observar en la FIGURA 111, la señal en el punto T1, tiene una duración más prolongada que en T3, por lo que sería un movimiento más rápido del vehículo. La longitud de estos valor dependerá de el valor que se le asigne al registro ALTO a través de la manipulación del SLIDER, se agrandará el tiempo que permaneza en ALTO la señal a mayor valor del SLIDER, y viceversa.

Por el contrario, el registro BAJO, controla el tiempo que los motores están en modo reposo. La mezcla del tiempo que los motores están funcionando (ALTO) y el tiempo que tienen la orden de detenerse (BAJO), da lugar al control de la velocidad del vehículo. El valor de este registro, BAJO, y el tiempo que transcurre hasta que se vuelve a pasar al estado de "ALTO", son valores fijos que se han de editar en el mismo programa, a diferencia de ALTO, que se controla su valor a través del SLIDER.

Como se muestra en la FIGURA 110, los recuadros 2 y 4 tienen una estructura exacta, con valores diferentes (el recuadro 4 tiene la subrutina "STOP" para parar los motores):

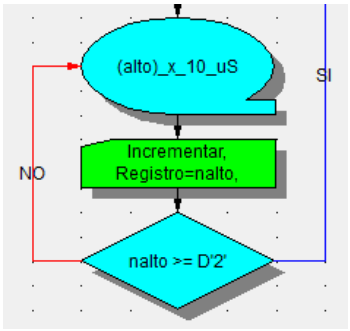


Fig 112. Ciclo de cuenta

Para poder jugar con los dos registros establecidos y con ello controlar la velocidad, se han creado dos bluces, con los que se controlará el tiempo que el programa pasará en el estado de "ALTO" y "BAJO".

Cada vez que se entra en el bucle, el programa se pasa un tiempo determinado, fijado por el usuario, detenido en un temporizador. Este temporizador utiliza el registro ALTO, en el cual tiene el valor del SLIDER, para multiplicarlo por un tiempo determinado, el cual, en este caso, se ha fijado de diez microsegundos (este tiempo se ha fijado de tal

manera tras diferentes pruebas y errores, hasta que se encontró el más adecuado para el control de la velocidad, de la manera más fina posible). Por ejemplo, en el caso de que el registro ALTO tuviese el valor de 100 (entre 0 y 255), el programa estaría parado en esa instrucción 100 x 10 = 1000 microsegundos = 1 milisegundo.

Para crear el bucle, se ha dispuesto de una condición, en la que se pregunta por el valor de un registro nuevo "NALTO" el cual, incrementa su valor en 1, cada vez que el programa pasa por la instrucción anterior:

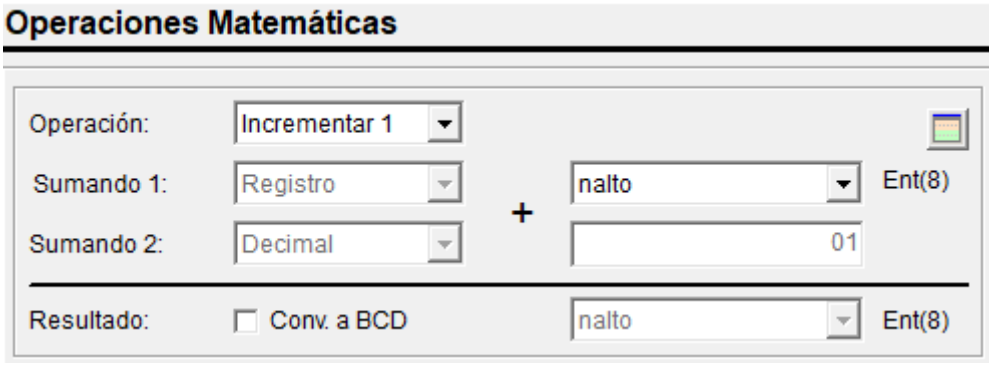


Fig 113. Incrementar el valor del registro "nalto"

Hasta que el bucle no se haya realizado las veces indicadas en la condición, no podra pasar a la siguiente instrucción. El valor del registro NALTO es determinante para el buen funcionamiento del control de la velocidad, ya que también afecta al tiempo que el programa esté en "ALTO" o "BAJO", por lo que se ha tenido que experimentar con diferentes valores hasta encontrar, junto con el valor del temporizador, el adecuado para un buen funcionamiento. Esto se hace en ambos bucles, con los registros correspondientes.

Como se ha indicado anteriormente, en el bucle donde se establece el tiempo que el programa pasa en "BAJO", está la subrutina "STOP" la cual pone los bits de control de los motores a 0, parandolos por un breve periodo de tiempo establecido por el usuario, según el valor de "BAJO" y "NBAJO". El tiempo que estos reciben la instrucción de detenerse, es imperceptible para el ser humano, microsegundos, por lo que en vez de pararse, se consigue el control de la velocidad.

El bloque 3 de la FIGURA 110 corresponde al reseteo de el valor de los registros correspondientes a la condición (para evitar el acarreo) y asignación de valor de "NBAJO"

7.1.6.3 JOYSTICK. CONTROL SERVOS

Una vez conseguido el control de la dirección de los motores, se procede a configurar el control de la posición de los servomotores situados en la parte superior de los motores, para poder girar el vehículo en una dirección u otra.

En un primer momento se decidió controlar dos servos de manera independiente, uno para cada motor, pero con la finalidad de utilizar los menos PINS posibles del PIC, por motivo de espacio disponible, se decidió unificar la señal para controlar ambos con una misma instrucción.

La primera tarea a realizar en cuanto a la programación de los servos con el software Niple, es declarar el dispositivo, al igual que se realizó con la pantalla LCD:

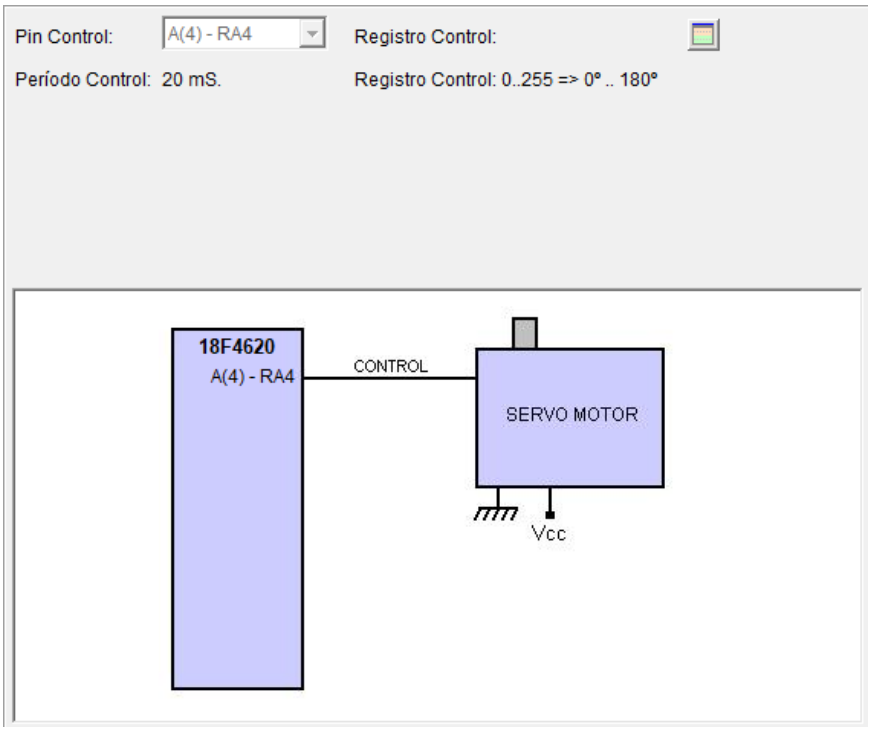


Fig 114. Declaración servomotor y su pin de control

La declaración del dispositivo requiere elegir cual será el Pin de Control, a través del cual se moverá el servo a una posición u otra.

La posición del servomotor dependerá del valor recibido del EJE X, el cual se recibe a través de RFI_DATA_0, y podrá variar de 0 a 255, pudiendo moverlo de su posición inicial de 90º a 180º cuando el valor recibido es 255 (incluyendo todo el rango el valores intermedio) o a 0º cuando el valor recibido es 0.

Una vez declarado el dispositivo, se crea una subrutina que se le da el nombre de “SERVO” donde se controlará el movimiento de este. Dentro de esta, en función de la posición del EJE X del Joystick, el servo deberá de realizar un movimiento u otro, por lo que la programación del movimiento se realizará con condiciones:

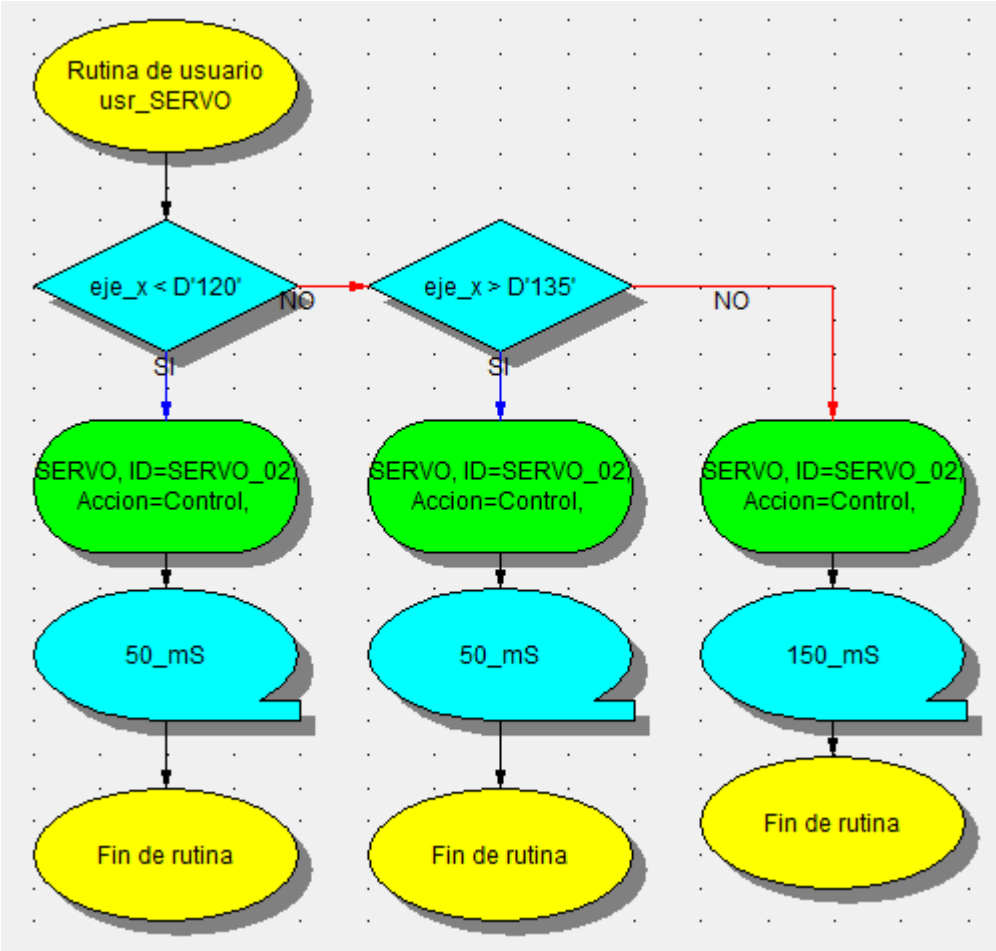


Fig 115. Posicionamiento del servo en función del valor de EJE_X

Debido a la calidad del Joystick, los valores del EJE X en resposo varian sin actuar sobre el. Poniendo como valor en resposo el decimal 127, se programa el movimiento de manera que el movimiento lateral izquierdo, de 90º a 0º, solo se realice en el momento que detecte una variación en el valor inferior al decimal 120, en caso contrario, se mantendrá en el punto de reposo de 127 (90º). Sucede por igual con el movimiento lateral derecho, de 90º a 180º.

Para evitar que el movimiento sea rápido y descontrolado (se requiere un cambio de posición lento para evitar desplazar la maqueta del vehículo) se incluyen unos temporizadores a continuación de la instrucción de mover el servo, para que lo realice con un movimiento escalonado.

8. PROGRAMACIÓN MAQUETA VEHÍCULO

La maqueta del vehículo que se está desarrollando, controlada a partir del módulo fijo, contará con la opción de elegir entre dos tipos de movimientos.

El primero, es el movimiento manual, el más simple, donde el usuario es el que decide el movimiento a realizar, dirección de desplazamiento, posición de los servomotores para realizar movimientos laterales, así como diferentes características añadidas tras la comprobación del buen funcionamiento de las funciones básicas previamente descritas.

El segundo, es el movimiento automático, el más complejo, en el cual, el usuario únicamente ha de accionar el interruptor que lo pone en funcionamiento, ya que todos los parámetros a controlar han sido establecidos en el software. Así mismo, las características añadidas con posterioridad, también pueden ser utilizadas durante el funcionamiento del movimiento automático.

8.1 MOVIMIENTO MANUAL

Para alternar entre un tipo de movimiento u otro, se ha incluido en el módulo transmisor, un interruptor.

Este interruptor funciona con una señal digital, cuando está en reposo, toma el valor de 0, y cuando se activa, toma el valor de 1. La siguiente subrutina "MANUAL_AUTOMAT" es la creada en el módulo transmisor:

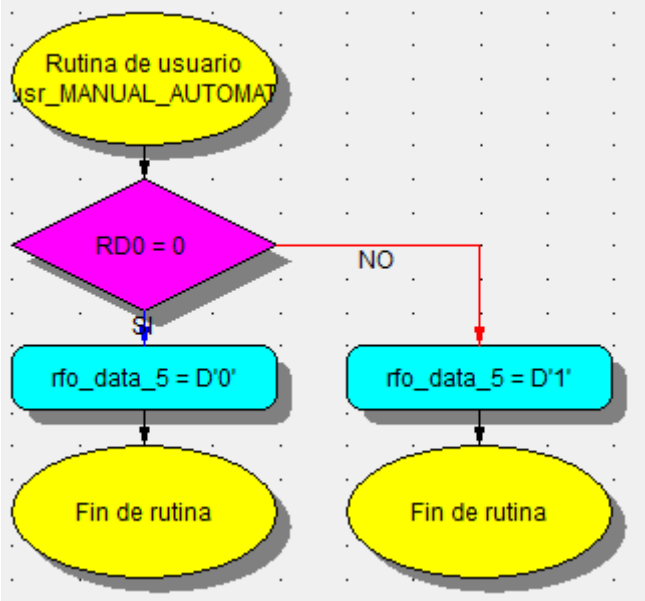


Fig 116. Selección Manual o Automático

El interruptor ha sido conectado al pin 0 del Puerto D del PIC. En función del valor de este, se le asigna un valor decimal de "0", si el interruptor está en el estado de

reposo, al registro RFO_DATA_5 (si el valor es “0”, el movimiento será manual), por el contrario, si el interruptor está accionado, se le asigna el valor decimal “1” a RFO_DATA_5 (si el valor es “1”, el movimiento será automático).

El registro de RFO_DATA_5 será enviado al módulo receptor (vehículo) y recibido en RFI_DATA_5:

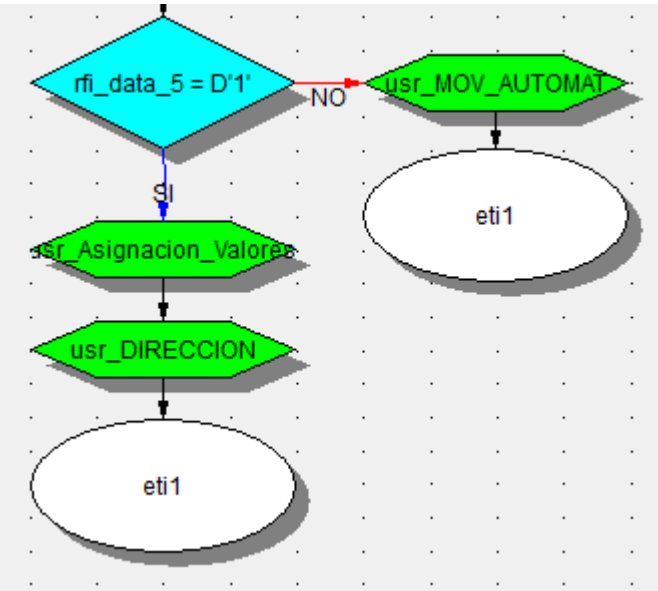


Fig 117. Movimiento automático o manual dependiendo del valor de RFO_DATA_5

Como se ha explicado previamente, dependiendo del valor recibido en RFI_DATA_5, se decidirá el tipo de movimiento a realizar. Por defecto, el interruptor va a estar en la posición “1”, seleccionado el movimiento manual.

Tras escoger el movimiento manual, antes de acceder a la subrutina “DIRECCIÓN”, dentro de la cual están las instrucciones para mover el vehículo, es necesario acceder a la subrutina “ASIGNACIÓN VALORES”.

Dentro de esta subrutina se encuentran las instrucciones que asignan los valores de los EJE X y EJE Y a sus respectivos registos, así como el SLIDER, que son los tres que se utilizan para el movimiento manual.

En el apartado anterior, se explicó la configuración establecida dentro de la subrutina “DIRECCIÓN” para determinar el movimiento de los motores en función del valor del EJE Y. A continuación, se explicará la parte del código que se le ha añadido para poder realizar un giro del vehículo, lo cual se consigue haciendo girar uno de los dos motores mientras que el otro permanece en estado de reposo.

Este movimiento lateral, se realiza a partir del valor que se reciba del EJE X (este eje es el utilizado para mover los servomotores, pero como el movimiento de estos no se realiza dentro de la subrutina de desplazamiento del vehículo, se utiliza para realizar el giro de este).

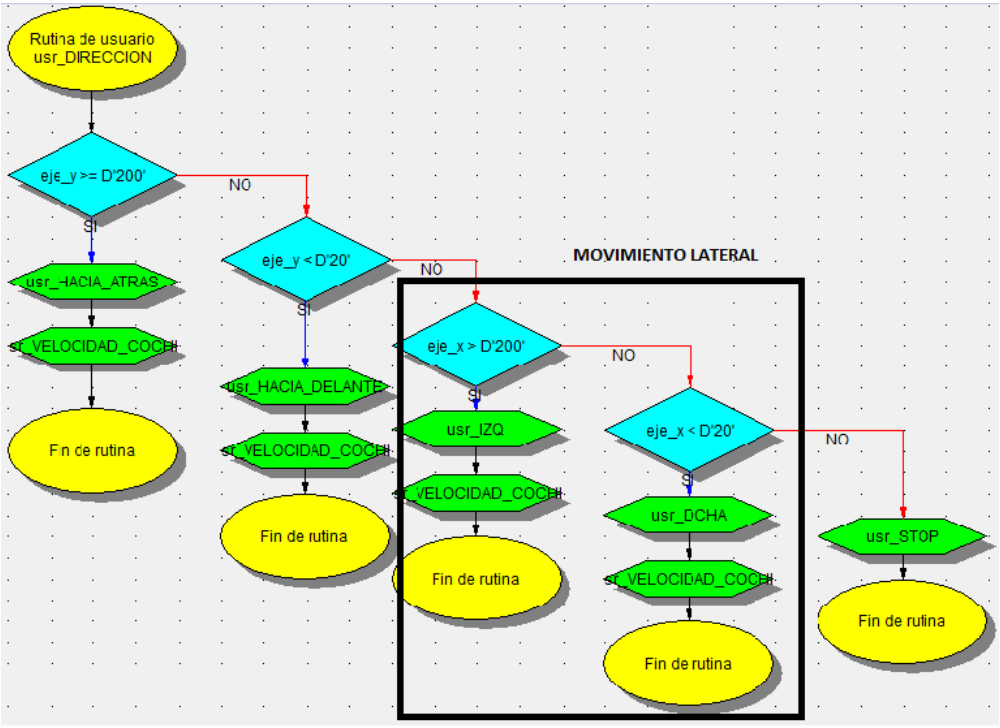


Fig 118. Movimiento lateral incluido en la subrutina “DIRECCION”

En el momento que el EJE X del Joystick, se desplace de manera lateral hacia la derecha, el valor del EJE X aumentará hasta llegar a 255 en su posición máxima, haciendo mover únicamente el motor izquierdo del vehículo para girar a la derecha, asignandole a los bit de control de los dos motores los valores que se observan en la figura 119.

Por el contrario, cuando el EJE X del Joystick, se desplace hacia la izquierda, el valor del EJE X disminuirá hasta llegar a 0 en su posición máxima, para mover únicamente el motor derecho del vehículo para girar a la izquierda, asignandole a los bit de control de los dos motores los valores que se observan en la figura 120.

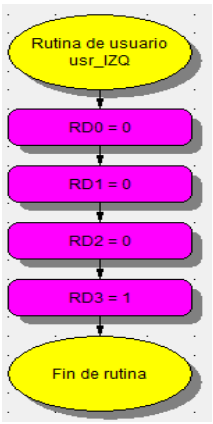


Fig 119. Mov Lat Dcha.

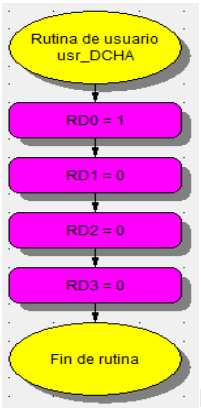


Fig 120. Mov Lat Izq.

El movimiento de los servomotores se realiza de manera independiente al movimiento de los motores, por lo que, al igual que se utiliza un interruptor para seleccionar el movimiento manual o automático, se ha configurado otro interruptor que selecciona entre movimiento de lo motores o movimiento de los servos:

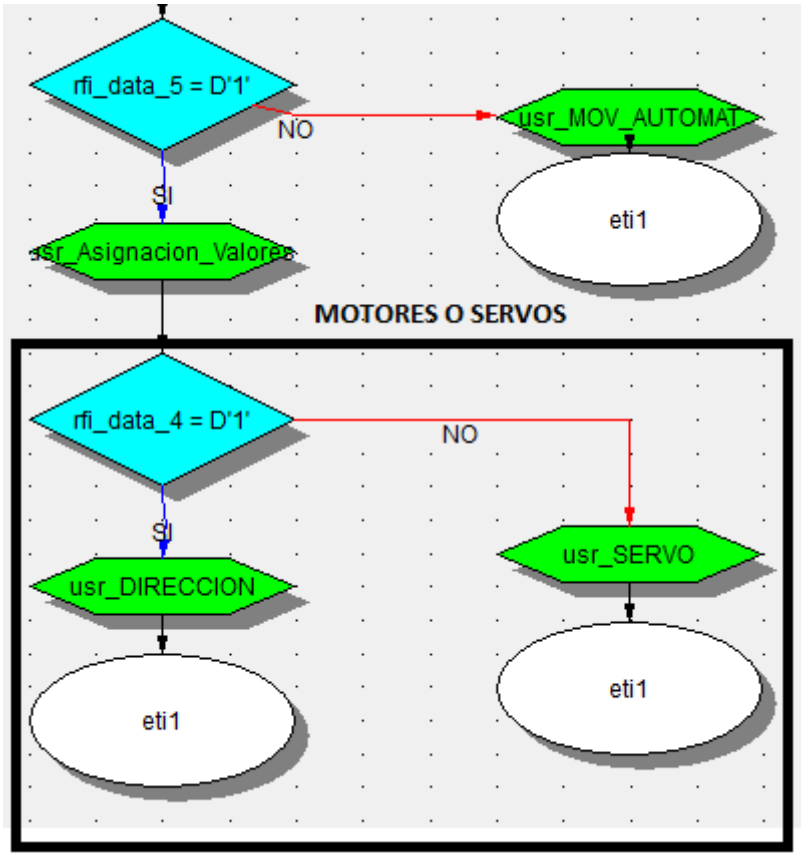


Fig 121. Selección movimiento de motores o movimiento servomotores

La condición en la que se lee el valor de RFI_DATA_4, es la encargada, una vez seleccionado el MOVIMIENTO MANUAL, de decidir si se va a proceder al movimiento de los motores o de la posición de los servos.

Un segundo interruptor es incluido en el programa del PIC del módulo transmisor:

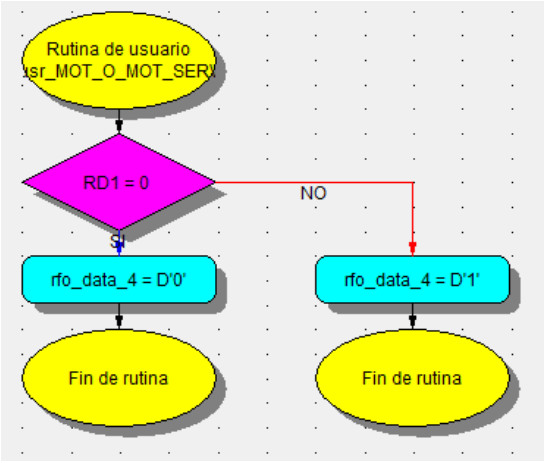


Fig 122. Interruptor el cual decide el movimiento de los motores o servos

El interruptor se ha conectado al PIN 1 del Puerto D del PIC del módulo transmisor. Si el interruptor está en su posición de reposo, tiene el valor digital “0”, por lo que en el registro RFO_DATA_4 se escribe el literal 0. Por el contrario, si se acciona el interruptor, este toma el valor digital “1”, escribiendo en el registro el valor literal “1”.

Dependiendo del valor escrito en el registro, el valor de RFI_DATA_4 del módulo receptor tendrá un valor u otro, la condición señalada en la FIGURA 121, cogerá el camino de mover los motores o mover los servos.

El movimiento de los servos se había configurado de tal manera que se podía realizar a la vez que el vehículo se desplazaba manualmente, pero cuando se comprobó el funcionamiento en el vehículo, se observó que a a la hora de iniciar el movimiento hacia delante (o hacia atrás) girando también los servomotores desde su posición de reposo, el movimiento se realizaba, pero de una forma inusual, a golpes. Es por esto, por lo que se decidió separar los dos movimientos utilizando un interruptor. Lo cual, en definitiva, es más práctico y da lugar a un movimiento de ambos dispositvos más natural, sin verse forzado.

8.2 *MOVIMIENTO AUTOMÁTICO*

Una vez explicado las características y diferentes opciones de las cuales consta el movimiento manual, es el momento de proceder a explicar el otro tipo de movimiento del que dispone el vehículo, el movimiento automático.

La idea principal de este movimiento radica en la capacidad del vehículo en desplazarse de manera autónoma, hacer un recorrido establecido por el usuario y detenerse al finalizarlo, corregir la dirección del mismo en caso de que se produza un desvío en el recorrido establecido y realizar movimientos verticales (hacia delante y hacia atrás) así como laterales (derecha e izquierda) controlando para ello los servos de manera autónoma.

En primer lugar, se hablará de los cuatro dispositivos que se utilizan para poder guiar el vehículo en su recorrido. Estos dispositivos son cuatro SENSORES DE REFLEXIÓN CNY70. Estos sensores están distribuidos de la siguiente manera en el vehículo:

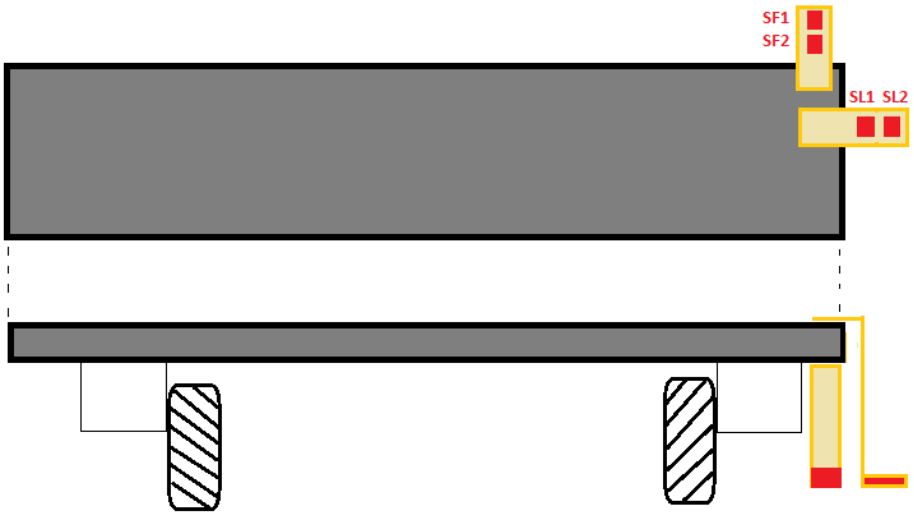


Fig 123. Posición Sensores de Reflexión CNY70

Para guiar el vehículo con estos sensores, se utilizará una cinta adhesiva negra. Esta cinta será colocada de manera vertical, para los movimientos hacia delante y hacia atrás, y de manera horizontal para los movimientos de izquierda o derecha.

En el momento que los sensores estén encima de la cinta, está reflectará la luz infraroja, la cual detectará el colector del sensor, encendiendo un LED indicando la detección de la cinta. En el momento que el sensor salga del grosor de la cinta, dejará de recibir la reflexión de la luz infraroja y el LED se apagará.

Es por esto último por lo que se ha colocado dos sensores en cada punto, para detectar si el vehículo desborda la cinta por la derecha o izquierda (en el movimiento vertical) o por la parte superior o inferior (movimiento horizontal).

8.2.1. *CORRECCIÓN DESVÍO MOVIMIENTO VERTICAL*

Los dos sensores de reflexión instalados en el lateral derecho del vehículo, están configurados en el PIC como entradas, en los Pines RA2 y RA3, ya que envían un “0” si no detectan la cinta negra y un “1” si la detectan.

La situación que se ha de dar para que el vehículo realice el movimiento vertical sin ningún tipo de desviación, es que ambos sensores, SL1 y SL2, detecten la cinta negra y transmitan al PIC el valor “1”. En el momento que esto no suceda, significará que se ha desviado a la derecha (SL1 = 1 , SL2 = 0) o a la izquierda (SL1 = 0 , SL2 = 1).

En la siguiente imagen, se observa la primera parte del programa del movimiento automático:

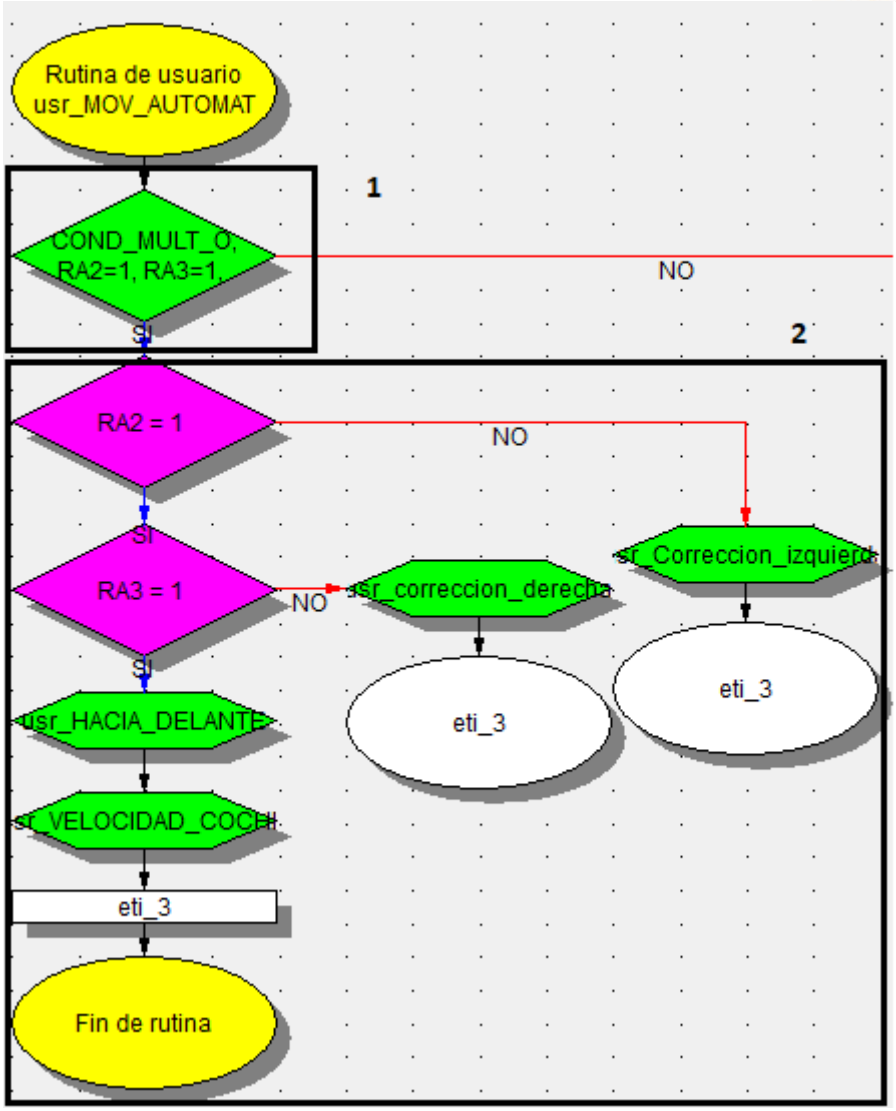


Fig 124. Movimiento Automático vertical (1) y correción por desvio (2)

En el primer recuadro de la FIGURA 124, se puede observar una condición, pero no como la condiciones utilizadas anteriormente. En este caso, se trata de una condición multiple, y tiene la característica de poder incluir varias condiciones a la vez para poder pasar, en caso de cumplirse todas o alguna de ellas (según se configure) a la siguiente instrucción del programa. Esto se configura en el software Niple de la siguiente manera:

Fig 125. Declaración de condición múltiple

Los datos a evaluar pueden ser Registros, Bits o Pines. Aquellos que se vayan a comparar se añaden a la lista, como en este caso puede verse en la FIGURA 125 (RA2 y RA3, los pines de los dos sensores de reflexión). Así pues, también se puede seleccionar el tipo de lógica de comparación, donde se puede elegir entre O (la cual no implica el cumplimiento de todas las condiciones adjuntas, sino que con el cumplimiento de una o varias de ellas, la condición es afirmativa y pasaría a la siguiente instrucción) e Y (en esta las condiciones adjuntas han de verse cumplidas en su totalidad, o la condición será negativa). En este caso, la elección del tipo de lógica de comparación sería O, ya que se pretende detectar si existe un desvío del vehículo o no, y con esto la dirección del desvío.

En el segundo recuadro del programa creado para el movimiento automático de la FIGURA 124, se observan dos condiciones, dirigidas al valor de los Pines RA2 y RA3, que funcionan como entradas de los sensores de reflexión.

En el momento que el valor de los dos es un “1”, significaría que el vehículo está centrado y puede continuar con el desplazamiento vertical, por lo que se procedería a la subrutina “HACIA DELANTE” y “CONTROL DE VELOCIDAD”. El programa seguirá realizando este movimiento hasta que detecte que uno de los dos sensores cambia de valor “1” a “0”, lo cual significará un desvío del vehículo.

Si el BIT que cambia de valor, es el RA2, significará que se ha desviado a la izquierda, por lo que habrá que corregir a la derecha. Para esto, se detiene el motor derecho y se pone en funcionamiento el izquierdo, hasta que el vehículo se centre y vuelva a ponerse el BIT RA2 a “1”.

Si por el contrario, el BIT que cambia de valor, es el RA3, significará que se ha desviado a la derecha, por lo que habrá que corregir a la izquierda. Para esto, se detiene el motor derecho y se pone en funcionamiento el izquierdo, hasta que el vehículo se centre y vuelva a ponerse el BIT RA3 a “1”.

El movimiento horizontal comienza en el momento que, durante el movimiento vertical, los sensores frontales, SF1 y SF2 (RD6 y RD7, respectivamente), pasan de transmitir el valor “0” al PIC, a transmitir el “1”, lo que significa que se han encontrado con la cinta negra que da comienzo al movimiento horizontal.

Fig 126. Movimiento Vertical

Si por algún casual, durante el desarrollo del movimiento vertical, o durante la incorporación al cambio de movimiento, se pierde la detección, por parte de los cuatro sensores de reflexión, la detección de la cinta negra, el vehículo por seguridad, se detendría, utilizando la subrutina “STOP”. Esto queda programado de la siguiente manera:

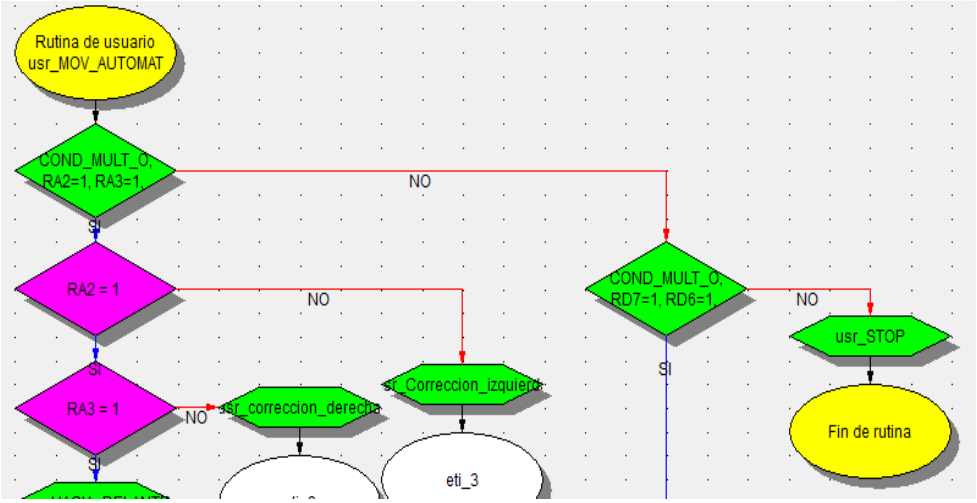


Fig 127. Selección Movimiento Frontal, Lateral o Parada del vehículo

Si la condición múltiple asociada a los BITS RD6 y RD7 de los sensores de reflexión frontales es afirmativa, significaría que se ha llegado al final del primer desplazamiento y ha de procederse a girar los servomotores a la posición de 0º, para realizar el desplazamiento lateral hasta el siguiente punto de riego.

Para el movimiento lateral, se ha decidido interrumpir el sistema de riego durante la realización de este.

El movimiento de los servomotores se ha programado en una subrutina independiente, con un tiempo de pausa significativo para asegurar el movimiento total de estos hasta la posición de giro de 0º.

En las figuras 127 y 128 se puede observar la parte del programa diseñada para este movimiento:

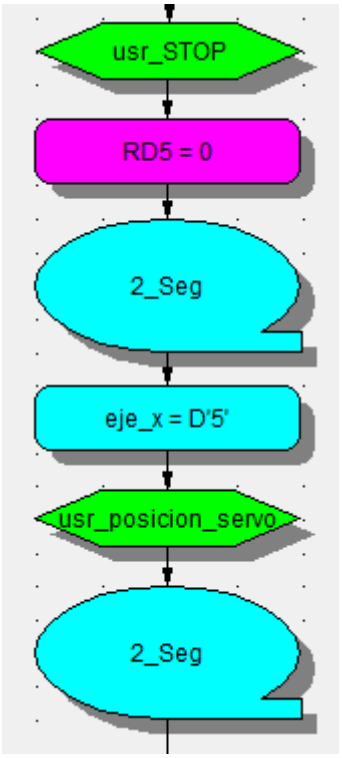


Fig 128. Posicionamiento Servo

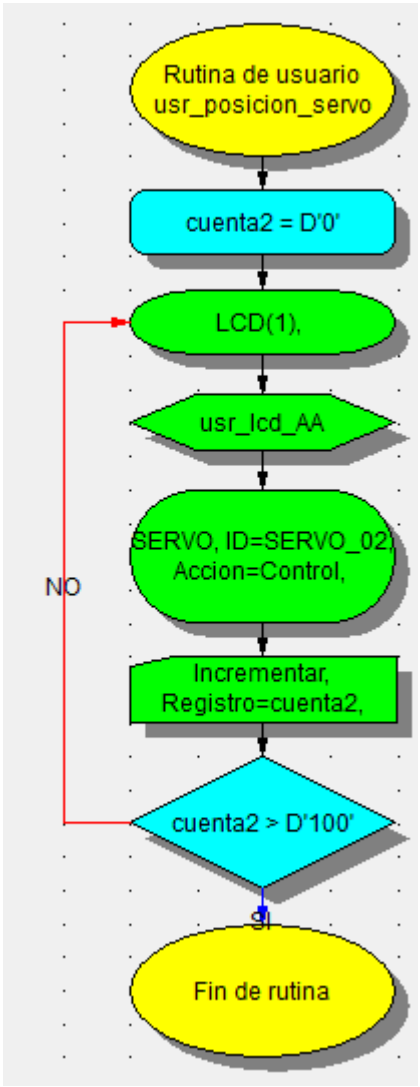


Fig 129. Bucle para el posicionamiento correcto

En la figura 128 se puede observar la parte principal de esta parte del programa, la cual, al haber detectado los valores digitales de "1" por parte de los sensores de reflexión frontales, detendría el movimiento del vehículo con la subrutina "STOP", así como el riego previamente en funcionamiento, asignandole al Pin de control de este, RD5, el valor de 0.

Una vez desactivado tanto el movimiento como el riego del vehículo, se realizaría una parada de dos segundos, antes de pasar a la siguiente instrucción. Así mismo, al finalizar la instrucción de movimiento del servo, se realizará otra parada de dos segundos. Esto se introducide para evitar un movimiento inmediato al finalizar el primer desplazamiento por parte de los servomotores y la puesta en marcha de los motores, lo cual, únicamente ocasionaría problemas.

Por lo que, como se ha explicado, la fase de posicionamiento de los servomotores se realizará paso a paso:

1º Parada del Vehículo
2º Pausa de dos segundos
3º Posicionamiento de los servomotores
4º Pausa de dos segundos
5º Movimiento del vehículo

En la figura 128, se puede observar como se le asigna el valor del decimal “5” al registro “EJE_X”. Como préviamente se ha expllicado, el “EJE_X” es el encargado de posicionar los servomotores en un punto u otro, y como se trata de un movimiento automático en el que no se ha de actuar sobre el JOYSTICK, se le asigna el valor al que se pretende mover los servomotores. Por motivos de calidad del producto se le ha asignado el valor de 5 en vez de 0, ya que con este último la posición de la rueda instalada en el servomotor quedaba un tanto ladeada, provocando el desvío del vehículo.

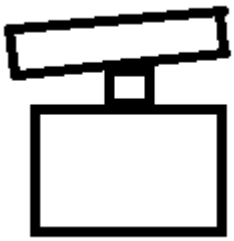


Fig 130. Eje_X = 0

Fig 131. Eje_X = 5

Una vez asignado el valor al que se pretende que se posicionen los servomotores, es momento de introducir la instrucción de control que los moverá, para ello, se ha creado una subrutina denominada “POSICIÓN_SERVO”, la cual, se puede observar en la figura 129.

Si el programa que se realiza, se encargase única y exclusivamente del posicionamiento del servomotor, con incluir la instrucción de control que viene dada por niple en el momento de la declaración de los servomotores, valdría para posicionarlo hasta la posción que se requiere, sin embargo, en este programa, el movimiento lo va a realizar de manera autónoma, y después, pasará a otra instrucción totalmente independiente. El paso de la instrucción de posicionamiento de los servomotores a la instrucción de movimiento del vehículo, sin ningún tipo de temporizador entre un paso y otro, impide que el posicionamiento de los servos se realice por completo, por lo que es necesario asegurarse de que este posicionamiento se realiza de manera adecuada.

Para conseguir el posicionamiento adecuado, se ha creado un bucle en el cual se manda constantemente la orden de posición a los servomotores. Como se puede observar en la figura 131, la subrutina consta de diferentes instrucciones:

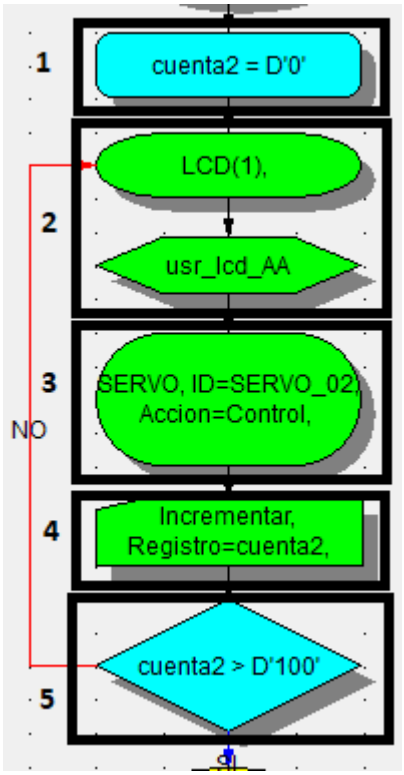


Fig 132. Subrutina Posicionamiento Servomotores

Este bucle se crea a partir del incremento de un registro previamente definido, “CUENTA2” (número 4). Este registro, cada vez que el PIC accede a esta subrutina, es puesto a “0” para evitar el acarreo(número 1), ya que si no se hiciese, como previamente ya a alcanzado el valor necesario para salir del bucle, saldría directamente de la subrutina. El bucle consiste en mandar constantemente la intrucción de posicionamiento de los servomotores (número 3) un número determinado de veces, en este caso 100 (número 5), para conseguir el desplazamiento completo. En el momento que el incremento del registro “CUENTA2” supera el valor 100, el bucle concluye, por lo que cuando CUENTA2 = 101, el programa pasaría a la siguiente instrucción, que sería esperar dos segundos antes de poner en movimiento el vehículo.

En la figura 132, se puede observar en el recuadro número 2, la configuración de un mensaje en la pantalla LCD. Esto se ha introducido debido a que, por motivos que se desconocen, el software Niple, requiere esta instrucción para poder realizar el bucle con el posicionamiento de los servomotores.

Una vez posicionados, como se ha explicado, el programa pasará a una instrucción de temporizado de dos segundos, antes de poner el vehículo en funcionamiento.

Una vez superado el tiempo establecido de espera, el PIC comenzaría el siguiente tipo de desplazamiento, el cual se ha programado de la siguiente manera:

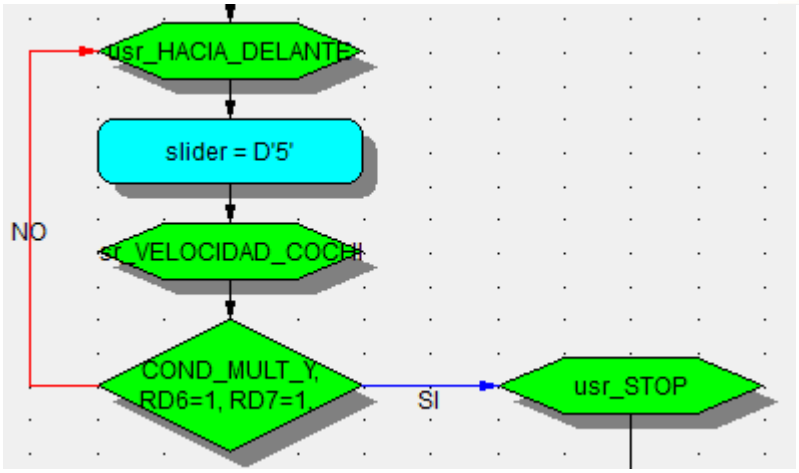


Fig 133. Posicionamiento servomotores a 0º.

Una vez están los servos en posición, se configura el desplazamiento del vehículo, lo cual, se realiza de manera que el vehículo se desplaza hacia delante (debido a que los servos están en la posición de 0º, si estuvieran en la posición de 180º, habría que desplazar el vehículo hacia atrás para conseguir el movimiento que se requiere), controlando la velocidad de este, asignandole un valor al registro "SLIDER", en función de la velocidad con la que se pretenda mover el vehículo (cuanto más pequeña mejor, para evitar salidas del camino).

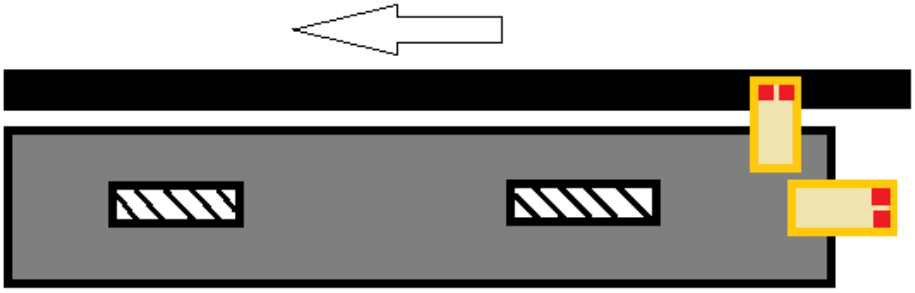


Fig 134. Posicionamiento sensor de reflexión frontal en el movimiento lateral

El vehículo se desplazará en esta dirección hasta que los sensores frontales, asociados a los Pines RD6 y RD7, dejen de detectar la línea negra, lo cual significará que los sensores de reflexión laterales, habrán detectado la línea que da lugar al tercer movimiento.

En ese momento, se realiza la misma acción que en el cambio del primer movimiento vertical al lateral, pero con la acción contraria. El vehículo se detendrá durante 2 segundos, posicionará los servomotores en 90º (EJE_X = 123 en valor digital), parará otro dos segundos, como se puede apreciar en el apartado 1 de la Figura 135 y dará comienzo el tercer movimiento, el cual es vertical pero con movimiento hacia atrás (apartado 3), hasta que deje de cumplirse la condición fijada (apartado 2).

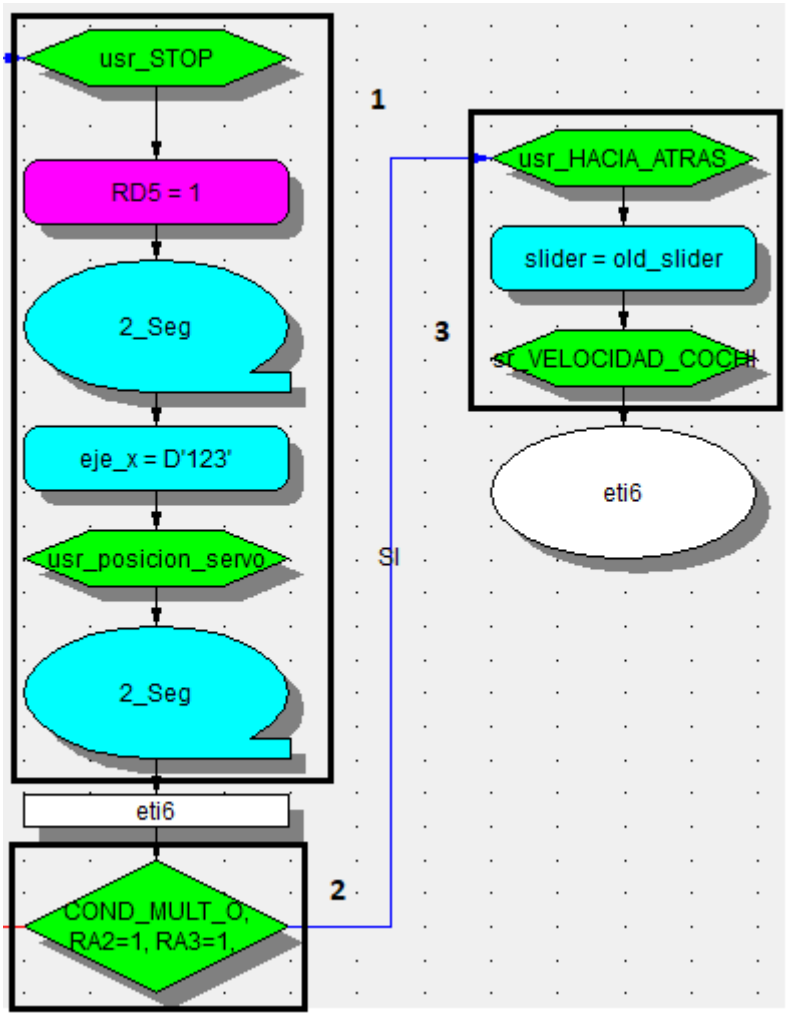


Fig 135. Movimiento vertical hacia atrás.

Fig 135. Movimiento Vehículo hacia atrás.

Una vez finalizado el movimiento, que acaba en el momento que los sensores de reflexión laterales no detectan la línea negra, es necesario volver a poner los servomotores a 0º, de la misma manera que ya se ha realizado y explicado previamente.

El siguiente movimiento presentó una complicación. Durante el movimiento lateral, los sensores de reflexión frontales, con los cuales se va a detener el vehículo en el punto que se situe la cinta negra para proceder al último movimiento, ha de pasar antes, por encima de la línea de guía de los sensores de reflexión laterales, lo cual, pararía el vehículo antes de tiempo, dando lugar a un error y a la parada del vehículo.

En la Figura 136 hay se puede observar gráficamente el problema para una mejor comprensión del mismo:

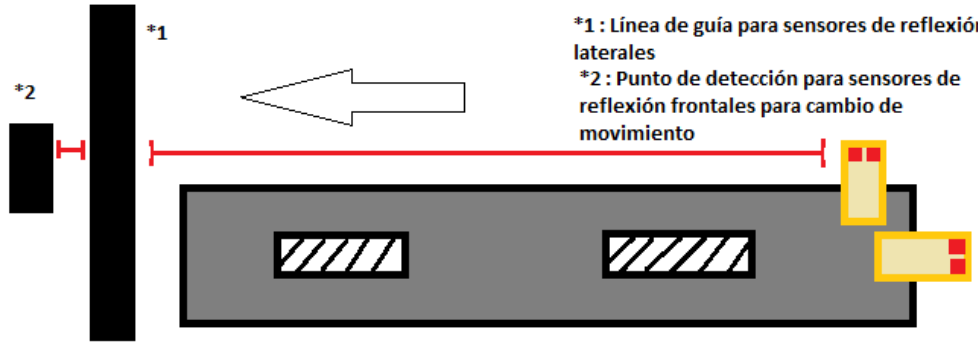


Fig 136. Problema en la programación del vehículo

Este problema se ha solucionado de tal manera que, en vez de considerar el recorrido en sí como uno solo, se ha dividido en tres:

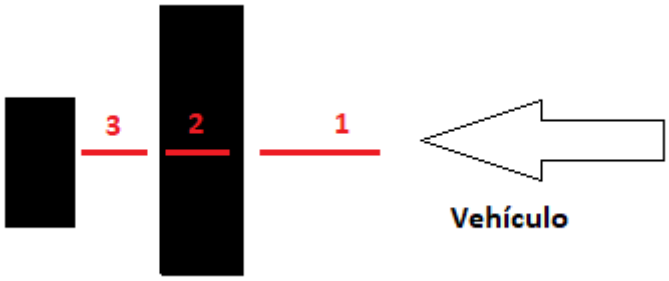


Fig 137. Solución propuesta al problema

En primer lugar, el vehículo se desplazaría hasta encontrarse con la línea guía de los sensores de reflexión laterales, para lo cual se configuraría la condición en el software niple de tal manera que se desplazase hasta que estos detecten la línea negra, como puede observarse en la FIGURA 137:

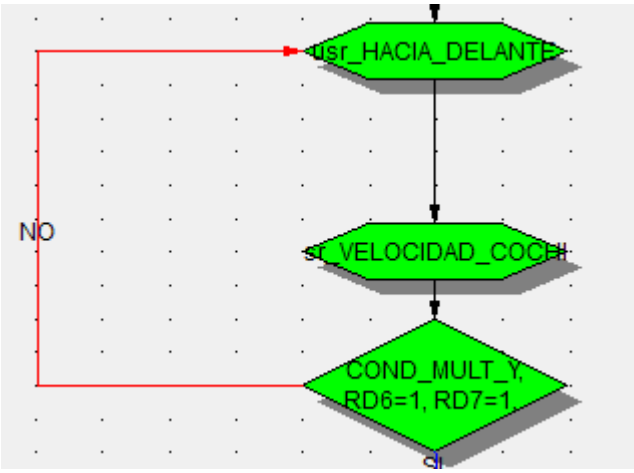


Fig 137. Primer tramo

En segundo lugar, el siguiente recorrido a realizar sería el grosor de la línea negra de guía. Para ello se configura el software de tal manera que, el vehículo se desplazará hasta el momento que los sensores de reflexión situados encima de la línea de guia, dejen de detectarla, como se observa en la FIGURA 138:

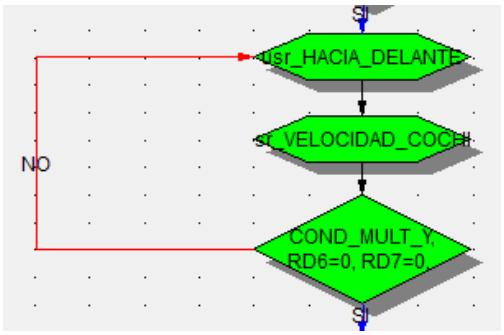


Fig 138. Segundo tramo

Al finalizar el segundo recorrido, el restante sería el que se ha de realizar hasta llegar a la línea que indica el cambio de tipo de movimiento, por lo que ha de desplazarse hasta que los sensores frontales detecten la línea negra.

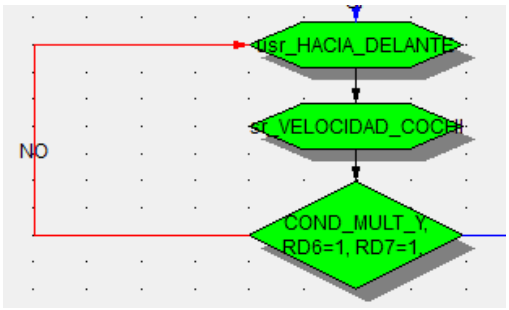


Fig 139. Tercer tramo

Al llegar al tercer punto del recorrido, en el momento que los sensores de reflexión frontales han detectado la segunda línea negra, comienza el último desplazamiento del módo automático.

El vehículo, ha de posicionar los servomotores de 0º a 90º, como ya se ha realizado en etapas anteriores y desplazarse verticalmente hacia delante, hasta alcanzar el final de la línea negra de guía. En el momento que los sensores de reflexión laterales, llegan al final de la línea negra, el desplazamiento programado habrá finalizado y el vehículo quedará inmovil hasta que el usuario actúe sobre el sistema.

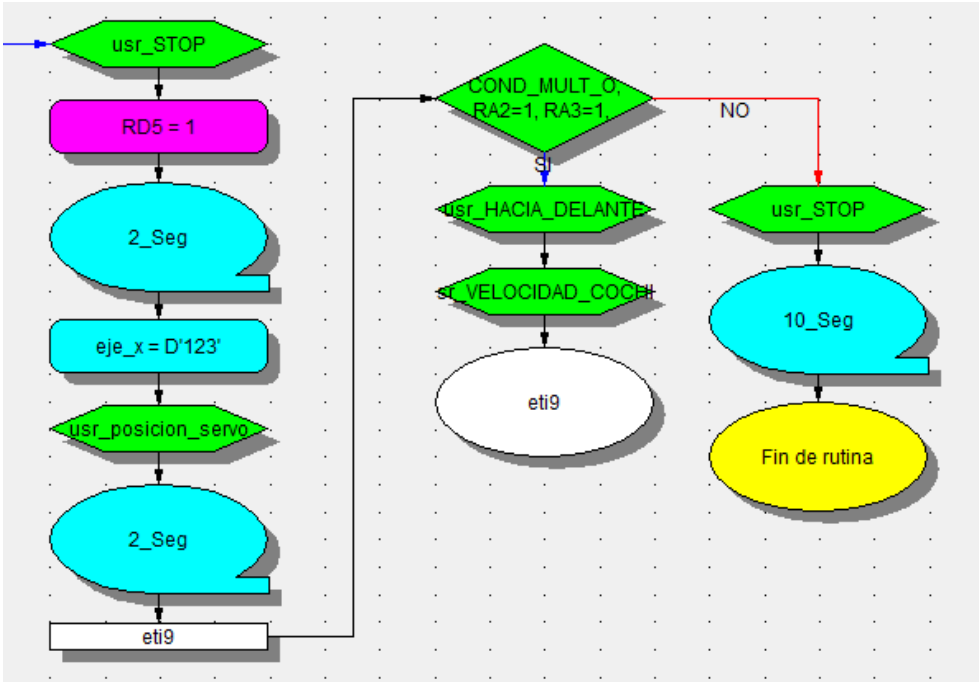


Fig 140. Parte final programación movimiento automático

8.2.2 SISTEMA DE SEGURIDAD POR SOFTWARE (PERRO GUARDÍAN)

Además de un pulsador de parada de emergencia situado en el mando de control para detener el vehículo en caso de error, se ha programado un sistema de protección que viene implementado dentro del microcontrolador, el cual se denomina PERRO GUARDIÁN (WDT).

El perro guardián, es un temporizador interno del PIC, el cual es programable. Su funcionamiento se basa en controlar cada un tiempo determinado (programable por el usuario en la opción de configuración del temporizador) si el microcontrolador recibe el paquete de datos transmitido por la estación de control.

La parte configurable del perro guardián, es el tiempo que tarda en comprobar la recepción de el paquete de datos. Esto se realiza cada vez que el temporizador rebosa (desde que empieza la cuenta con valor “0” hasta su valor máximo “255”).

Una vez que el Perro Guardián rebosa, existen dos posibilidades:

La primera opción sería que la comprobación de la recepción del paquete de datos sea afirmativa. En este caso, el funcionamiento del vehículo y del mando de control sería correcto, por lo que se debería de resetear el valor de la cuenta para que se vuelva a realizar hasta el momento que se desborde otra vez, volviendo a comprobar su funcionamiento.

La otra opción sería que la comprobación en el momento del desbordamiento mostrara que existe un problema, por parte del mando de control o del vehículo, causando un fallo en la recepción del paquete de datos, por lo que el perro guardían accedería a una interrupción, previamente creada, en la cual se encontraría la orden de parada del vehículo.

Fig 141. Configuración temporizado perro guardián

En la FIGURA 141, se puede observar la ventana de configuración del perro guardián, en la cual, el usuario ha de escoger intervalo de tiempo que tardará el registro en desbordarse, realizando la comprobación de recepción del paquete de datos. Aumentando el valor del apartado número “1”, se aumenta el tiempo que tarda el Perro Guardián en volver a realizar la comprobación. En este caso, se ha seleccionado un prescalado de “64”, lo cual implica que el Perro Guardián se desborde cada 528mS (0,5 segundos).

En el momento que la comprobación tras el desbordamiento, indica que ha surjido un error en alguno de los dos módulos, el Perro Guardian accedería a la "INTERRUPCIÓN POR WDT", en la cual se encontraría la instrucción de para de los motores del vehículo, por motivos de seguridad, como se indica en la FIGURA 142.

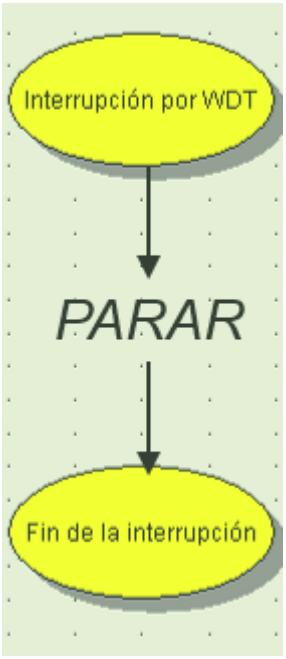


Fig 142. Interrupción por WDT

El programa del PIC se mantendría siempre dentro de la interrupción por WDT debido a que no se ha reseteado el registro del temporizador que se desborda, por lo que se accedería una y otra vez a la interrupción.

En el caso de que la recepción del paquete de datos sea afirmativa, será necesario resetear el valor del registro del temporizador a "0" para volver a empezar con la cuenta.

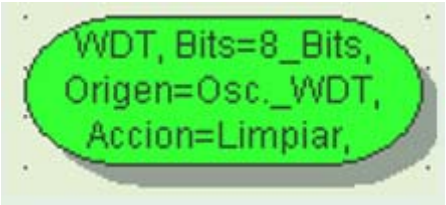


Fig 143. Instrucción para poner a "0" el contador.

9. DISEÑO MANDO DE CONTROL

Para el diseño de esta parte del proyecto, se han utilizado tres elementos indispensables a partir de los cuales se ha desarrollado la construcción de esta. Estos elementos se pueden observar en la siguiente imagen:

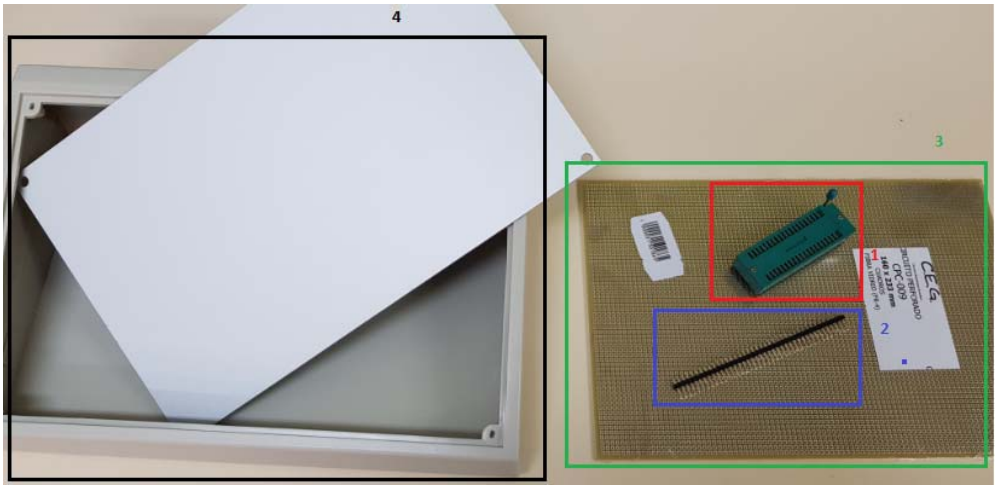


Fig 144. Elementos principales en la construcción del mando

El Zócalo de Conexión, con el número 1 (rojo) en la imagen, es el elemento al cual se le conecta el PIC 18F4620. Sirve de adaptador entre el circuito perforado, con el número 3 (verde) y el PIC a utilizar. Esto se utiliza para evitar tener que soldar el PIC al circuito, pudiendo así intercambiarlo por otro si este se estropea, o simplemente por facilidad a la hora de programarlo.

Así mismo, para poder conectar las entradas y salidas del PIC, así como la alimentación para los interruptores/pulsadores, el módulo regulador de tensión y la Pantalla LCD, se utiliza lo que se conoce como Regleta de Pines, con el número 2 (azul). Se han utilizado pines macho-macho y hembra-macho.

Para poder utilizar correctamente la regleta de pines, es necesario fijar estos al circuito perforado mediante el uso de la soldadura con estaño. La soldadura se ha de realizar por la parte inferior del circuito perforado, dejando en la parte superior los pines macho/hembra a utilizar.

Estos elementos irán en el interior de una caja metálica que simulara el mando de control del proyecto, la cual se puede observar en la imagen 144, con el número 4 (negro). En el interior de la caja irá todo el tema de conexionado de los elementos utilizados para el control del vehículo, mientras que en la parte superior de la caja, en la parte exterior de la tapa de esta, se instalarán estos elementos de control previamente explicados en el apartado “4.1 HARDWARE”.

A continuación se puede observar como ha quedado el montaje final interno del mando de control:

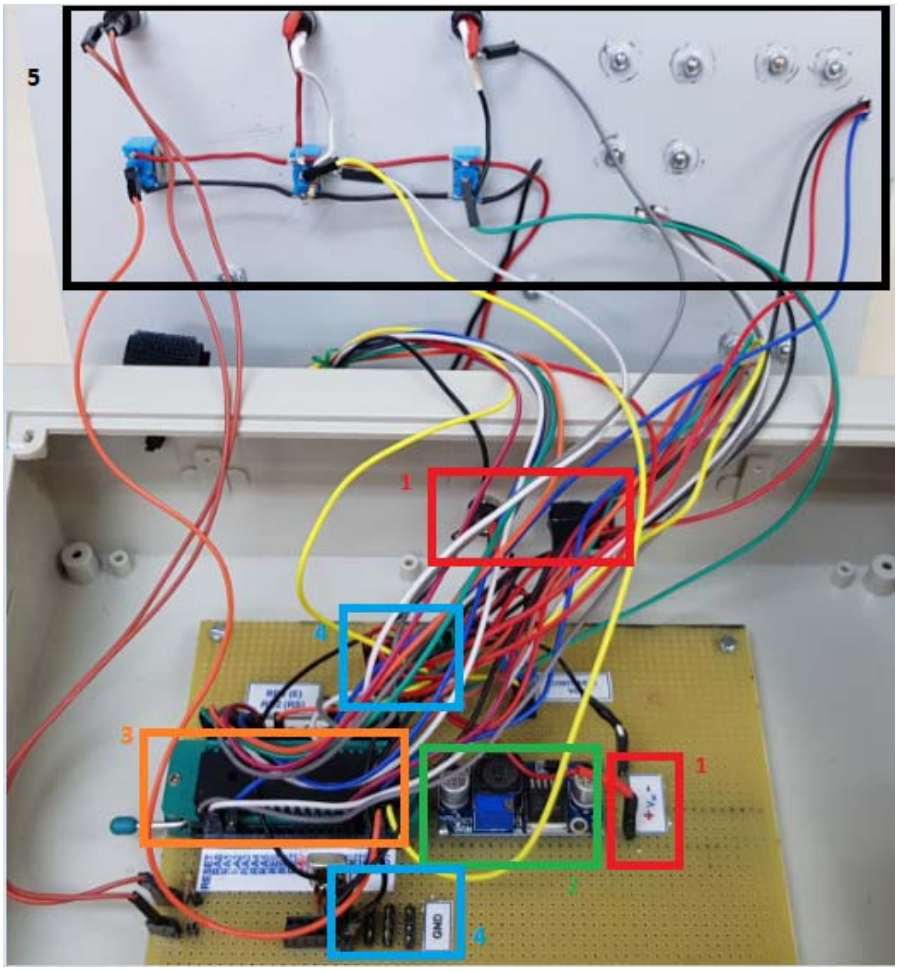


FIGURA 145. Conexionado interno del mando de control.

En primer lugar (número 1), se puede observar la alimentación externa que alimenta el circuito, la cual se conecta directamente al módulo regulador LM2596 (número 2), del cual, se obtiene en su salida GND y VCC (5V), donde se conectan todos los elementos utilizados (número 4, GND parte inferior, VCC 5V, parte superior).

En la parte exterior de la tapa de la caja, como ya se ha explicado, se encuentran los dispositivos de control del vehículo. Todo el cableado referido a la alimentación de los interruptores, pulsadores, Joystick, Slider y Pantalla LCD se conecta a la salida del módulo regulador para alimentarse a 5V.

La alimentación del mando de control se puede realizar mediante la utilización de diferente fuentes, según disponibilidad, por lo que no limita el funcionamiento del vehículo a la posesión de una fuente en concreto. La alimentación del mando de control, en este proyecto, se ha realizado variando entre la utilización de pilas de 9V, baterías de ión-litio de 12V, baterías recargables portátiles para teléfonos móviles.

Estas se conectan al mando de control utilizando dos conectores, un JACK SOLDADOR HEMBRA, utilizado en las diferentes baterías, y un JACK SOLDADOR MACHO instalado fijo en el mando, sobre el que se actúa para encenderlo o apagarlo utilizando un interruptor ON/OFF.

En las siguiente imagenes se pueden observar la conexión de la fuente de alimentación externa al mando de control, así la vista que tendrá el usuario a la hora de la manipulación del mando de control, una vez finalizado el montaje:

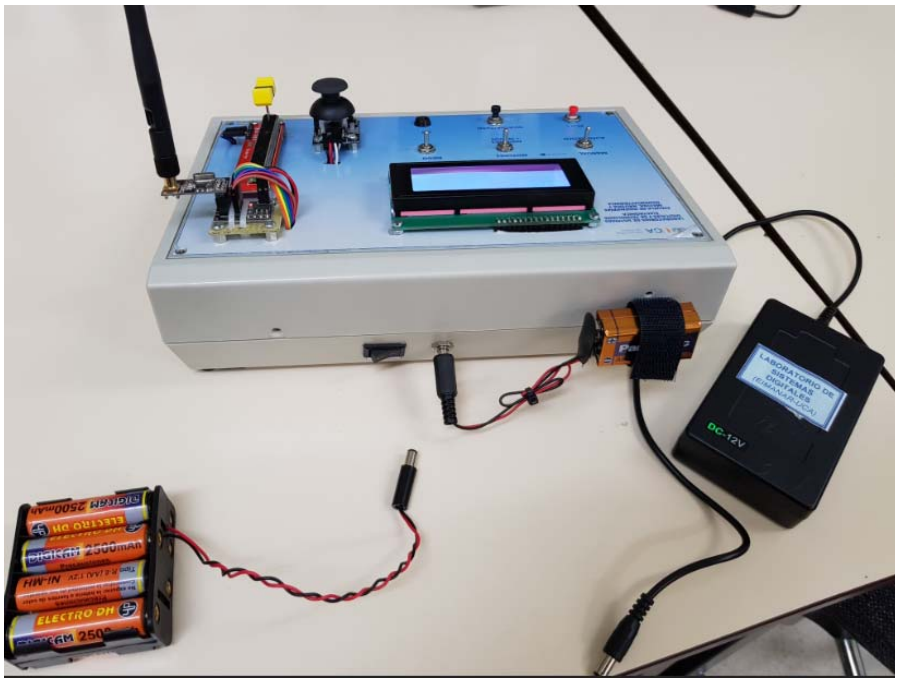


FIGURA 146. Fuentes de alimentación del mando de control.

FIGURA 147. Mando de control.

10. *DISEÑO VEHÍCULO*

Antes de comenzar con el desarrollo del diseño del vehículo, es obligatoria una introducción a los materiales utilizados para la construcción de este, y sobre ellos, destaca el utilizado para la estructura:

10.1 TOTEM MECHANICS

Totem es una empresa con sede en Lituania, la cual se dedica a la venta de piezas para la elaboración de cualquier tipo de estructura de tamaño reducido que se pueda imaginar.

Así mismo, a parte de vender las piezas para el diseño de estructuras propias, también venden el hardware requerido para poder controlarlas, ya programado o con la posibilidad de programación por parte del usuario.

En este proyecto, se ha adquirido un KIT de iniciación con las piezas básicas para construir, así como un KIT de herramientas necesarias para poder moldearlos al gusto y necesidad.

En la siguiente imagen se puede observar como se recibió el paquete con los elementos y herramientas adquiridos:

FIGURA 148. Materiales de construcción adquiridos

El inventario de estas piezas sería el siguiente:

Fig 149. Inventario piezas Totem.

Estas piezas han sido utilizadas para fabricar una maqueta de un Pivot Lateral de riego.

Estos vehículos, en tamaño real, llegan a alcanzar decenas de metros de longitud, con una magnífica estabilidad, debido a que este está dividido en segmentos, de aproximadamente 30 metros de longitud cada uno. Estos segmentos están hechos de acero, para dar una buena robustez al vehículo.

La maqueta, se ha desarrollado de tal manera que tenga las dimensiones de un único segmento del vehículo real, a escala, por supuesto, en concreto 1:33, por lo que el vehículo tiene un tamaño aproximado de 90cm de largo.

Para simular lo más fiel posible el diseño del vehículo real, se ha construido de tal manera que se asemeja a la forma de uno de los segmentos. La parte central, alargada y fina, se encuentra fijada a los dos extremos del vehículo, en los cuales están instalados tanto los motores como los servomotores que permiten el movimiento y direccionamiento del vehículo.

En la siguiente imagen se puede observar como es el estado final del vehículo:

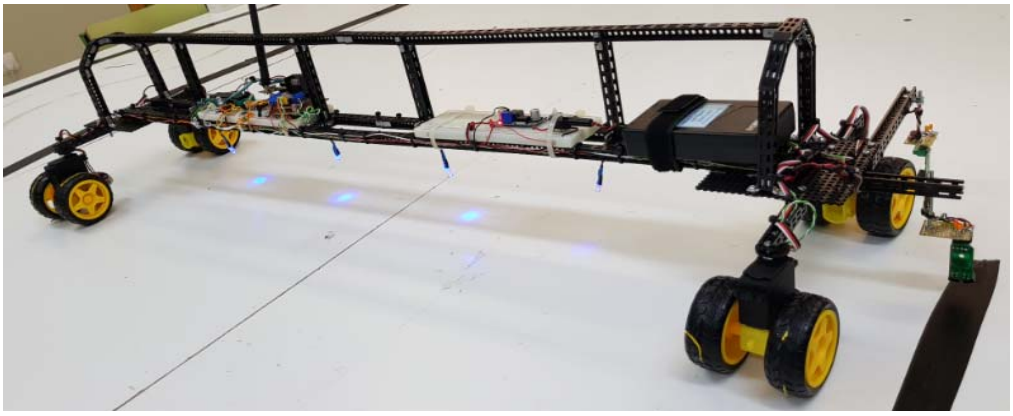


Fig 150. Maqueta Vehículo

En la maqueta, se han instalado dos protoboards. En las cuales se encuentras los diferentes elementos de control instalados.

En la primera, la más cercana a la fuente de alimentación, se ha instalado el regulador de tensión LM2596, el cual ofrece una tensión de salida de 5V, que alimentará la segunda protoboard.

En la segunda protoboard, la más alejada de la fuente de alimentación, se encuentra el PIC, el módulo NRF24L01 y el dispositivo de control de los motores L298N:

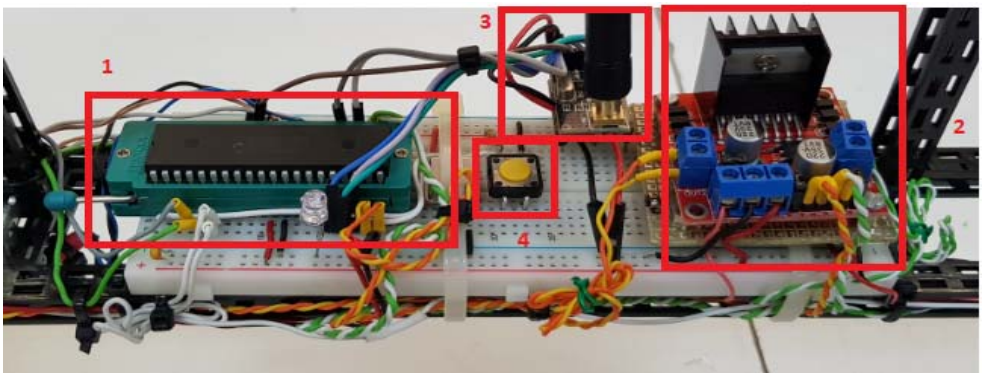


Fig 151. Protoboard de control del vehículo.

1. Zocalo Adaptador con PIC 18F4620.
2. Módulo controlador de motores L298N.
3. Módulo NRF24L01.
4. Pulsador de reseteo del PIC

La maqueta se desplaza con el movimiento de cuatro ruedas, situadas dos a cada lateral, de las cuales solo una de cada extremo poseé un motor DC para moverse, mientras que, en cuanto a servomotores se refiere, cada rueda poseé el suyo propio, ya que todas ellas han de posicionarse de igual manera si se quiere mover el vehículo en la dirección deseada.

Así mismo, la maqueta consta, en su lateral derecho, de cuatro sensores de reflexión, los dos frontales y los dos laterales, para el sistema de autoguiado del vehículo:

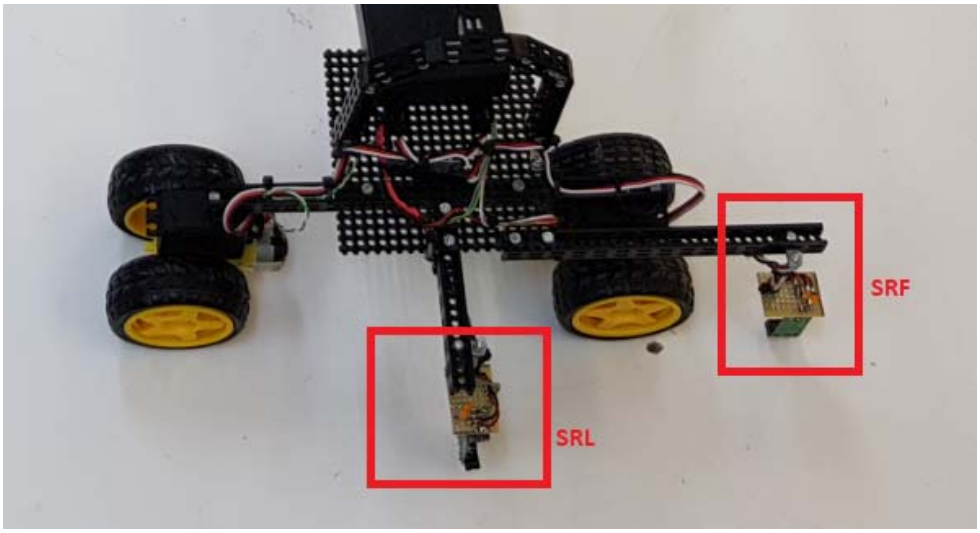


Fig 152. Sensores de reflexión en la maqueta

11. *CÁLCULOS REALIZADOS*

En este apartado se presentan los cálculos realizados para justificar la fuente de alimentación del mando de control y del vehículo, así como la velocidad de transmisión del transceptor

11.1 TABLAS DE CONSUMO

DESCRIPCIÓN	CONSUMO	UNIDADES	TOTAL	
PANTALLA LCD 20X4	75,00	1	75,00	mA
TRANSCEPTOR NRF24L01	15,00	1	15,00	mA
MICROCONTROLADOR 18F4620	200,00	1	200,00	mA
DIODO LED ROJO	20,00	1	20,00	mA
SLIDER	1	1	1,00	mA
JOYSTICK	1	2	2,00	mA
CONSUMO TOTAL MANDO A 5V DC			**313,00**	**mA**
POTENCIA TOTAL MÁXIMA			**1,565**	**W**
POTENCIA PILA 9V 565mA/H			**5,09**	**W**
EFICIENCIA TÍPICA FUENTE LM2596			**85,00**	**%**
POTENCIA REAL SUMINISTRADA POR FUENTE			**4,33**	**W**

TABLA 1. Consumo mando de control.

DESCRIPCIÓN	CONSUMO	UNIDADES	TOTAL	
MOTOR DC 5V	150,00	2	300,00	mA
SERVOMOTOR S3003	400,00	4	1.600,00	mA
MICROCONTROLADOR 18F4620	200,00	1	200,00	mA
DIODO LED AZUL	20,00	4	80,00	mA
TRANSCEPTOR NRF24L01	15,00	1	15,00	mA
CONTROLADOR MOTORES L298N	35,00	1	35,00	mA
SENSORES CNY70 S-110	35,00	2	70,00	mA
CONSUMO TOTAL VEHÍCULO A 5V DC			**2.300,00**	**mA**
POTENCIA TOTAL MÁXIMA			**11,50**	**W**
POTENCIA BATERIA 12V 3800mA/H			**45,60**	**W**
EFICIENCIA TÍPICA FUENTE LM2596			**85,00**	**%**
POTENCIA REAL SUMINISTRADA POR FUENTE			**38,76**	**W**

TABLA 2. Consumo maqueta vehículo.

11.2. CÁLCULO DE LA POTENCIA DE MANDO DE CONTROL

En corriente continua; $P = V \times I$, donde:

P = Potencia (W)

V = Tensión (V)

I = Corriente (A)

La Potencia (P) es la incógnita de la cual se pretende averiguar el valor. La Tensión (V), es el valor del voltaje al cual está alimentado cada dispositivo listado. La corriente (I), es el consumo por hora del dispositivo, viene reflejado en el datasheet como mA/h.

Por lo que:

- Pantalla LCD 20x4

$$P = 5V \times 0{,}075A = 0{,}375\ W$$

- Transceptor NRF24L01

$$P = 5V \times 0{,}015A = 0{,}075W$$

- Microcontrolador NRF24L01

$$P = 5V \times 0{,}2A = 1W$$

- Diodo LED Rojo

$$P = 5V \times 0{,}02A = 0{,}1W$$

- Joystick

$$P = 5V \times (0{,}001A \times 2) = 0{,}01W$$

- Slider

$$P = 5V \times 0{,}001A = 0{,}005W$$

POTENCIA TOTAL = 0,375 + 0,075 + 1 + 0,1 + 0,01 + 0,005= 1,565 W

A continuación, se calculará la potencia real suministrada por la fuente de 9V teniendo en cuenta la eficencia del módulo regulador de tensión LM2596, el cual es del 85% aproximadamente:

Potencia de la fuente = $9V \times 0{,}565A = 5{,}085W$

Calculando el 85% de este valor, se obtendrá el valor real de potencia suministrado por la fuente:

P. suministrada real = $0{,}85 \times 5{,}085W = 4{,}32W$

11.3. CALCULO DE LA POTENCIA DE LA MAQUETA DEL VEHÍCULO

- Motor DC 5V

$$P = 5V \times (0{,}15A \times 2) = 1{,}5W$$

- Servomotor S3003

$$P = 5V \times (0{,}4A \times 4) = 8W$$

- Microcontrolador 18F4620

$$P = 5V \times 0{,}2A = 1W$$

- Diodo LED azul riego

$$P = 5V \times (0{,}02A \times 4) = 0{,}4W$$

- Transceptor NRF24L01

$$P = 5V \times 0{,}015A = 0{,}075W$$

- Controlador motores L298N

$$P = 5V \times 0{,}035A = 0{,}175W$$

- Sensores CNY70 S-110

$$P = 5V \times (2 \times 0{,}035A) = 0{,}35W$$

POTENCIA TOTAL VEHÍCULO = 1,5 + 8 + 1 + 0,4 + 0,075 + 0,175 + 0,35 = 11,5W

Potencia de la fuente = $12V \times 3{,}8A = 45{,}6W$

Potencia real de la fuente = $0{,}85 \times 45{,}6 = 38{,}76W$

11.4. ELECCIÓN DE LA VELOCIDAD DE TRANSMISIÓN DEL TRANSCEPTOR

El transceptor NRF24L01 se puede configurar para tres velocidades de transmisión: 250Kbs, 1Mbs y 2Mbs. En este proyecto se va a emplear la velocidad de 250Kbs dado que es la idónea para las condiciones de trabajo.

Si se configura la velocidad de transmisión a 250Kbs, será equivalente a 31,25 KBs, y como cada paquete de datos incluye 8 Bytes, querrá decir que se podrán transmitir 3906 paquetes por segundo a esa velocidad.

En el apartado 7, se indica que desde el mando de control, se van a mandar paquetes de datos cada 10ms, es decir, 100 paquetes de datos por segundo. Por tanto, la velocidad de transmisión de 250Kbs será suficiente y apropiada para la transmisión de datos de este proyecto, ya que el alcance será el mayor.

12. PRESUPUESTO

DESCRIPCIÓN	PRECIO U.	UNIDADES	TOTAL	TOTAL CON IVA
MOTOR DC 5V	2,35	4	9,40	11,37
RUEDA PLÁSTICO	1,55	8	12,40	15,00
SERVOMOTOR S3003	6,00	4	24,00	29,04
MICROCONTROLADOR 18F4620	9,06	2	18,12	21,93
PANTALLA LCD 20X4	6,94	1	6,94	8,40
TRANSCEPTOR NRF24L01	6,79	2	13,58	16,43
ADAPTADOR NRF24L01	1,29	2	2,58	3,12
CONTROLADOR MOTORES L298N	3,22	1	3,22	3,90
FUENTE LM2596	1,82	2	3,64	4,40
PLACA PCB TALADRADA	8,00	1	8,00	9,68
CAJA PLASTICO MANDO	18,00	1	18,00	21,78
JOYSTICK	1,41	1	1,41	1,71
SLIDER	4,90	1	4,90	5,93
SISTEMA TOTEM MAKER	120,00	1	120,00	145,20
ZOCALO ZIF 40P	2,37	2	4,74	5,74
PROTOBOARD MB-102	2,89	2	5,78	6,99
CABLEADO Y CONEXIONADO	15,00	1	15,00	18,15
RESISTENCIAS 1/4W	0,10	7	0,70	0,85
CRISTAL 4MHZ	0,70	2	1,40	1,69
PULSADOR SWITCH	0,83	4	3,32	4,02
CONDENSADORES	0,20	6	1,20	1,45
INTERRUPTORES PALANCA	1,00	4	4,00	4,84
DIODOS LED VARIOS COLORES	0,83	5	4,15	5,02
POTENCIÓMETROS	1,10	1	1,10	1,33
MATERIAL FERRETERIA	8,00	1	8,00	9,68
TRANSDUCTORES MSE S-110	10,00	2	20,00	24,20
MANO DE OBRA (HORA)	23,20	15	348,00	421,08
PRESUPUESTO (€)			**663,58**	**802,93**

13. *CONCLUSIONES*

Se estima que el empleo de un sistema de posicionamiento global con las redes de satélites GPS y GLONASS, es el idoneo para definir el movimiento del vehículo en el modo automático propuesto en este trabajo, tal y como se emplea por ejemplo en la actualidad, en las máquinas agricolas. La utilización de estos dos sistemas de posicionamiento da lugar a una elevada precisión con margen de error centimétrico.

Dada la versatalidad de movimiento del sistema propuesto (360º), otra aplicación del mismo podría ser su empleo como AGV (Automated Guided Vehicles) en la estiba portuaria, tal y como se emplea en la actualidad en el puerto marítimo de Rotterdam.

Con la utilización de un entorno visual de programación de microcontroladores, como es el Niple, no es necesario tener un conocimiento profundo de la arquitectura del microcontrolador que implique la programación en bajo nivel. Esto se ha apreciado en este proyecto en la conversión de las librerías de programación del transceptor NRF24L01, de Nemónico a Visual.

La sencillez de uso del módulo transceptor NRF24L01 así como su economia, lo hace idóneo para su empleo en proyectos de sistemas basado en microcontrolador.

El sistema de riego propuesto, es idóneo para su empleo en fincas con parcelas no rectactangulares, en las que no se puede utilizar el sistema de pivote lateral.

El hecho de haber realizado una maqueta/pototipo proporciona un mayor conocimiento del sistema con una curva menor de aprendizaje. Así mismo se reduce el riesgo de fallos en la programación, ya que así puede observerse cual sería su funcionamiento en la realidad.

14. *BIBLIOGRAFÍA*

MICROCONTROLADORES PIC: LA CLAVE DEL DISEÑO [*AUTOR: JOSE MARÍA ANGULO USATEGUI ; EDITORIAL: THOMSON EDITORES*]

MICROCONTROLADORES PIC. TEORÍA Y PRÁCTICA [*AUTOR: MIKEL ETXEBARRIA; EDITORIAL: CREACIONES COPYRIGHT*]

MICROCONTROLADORES PIC. LA SOLUCIÓN EN UN CHIP [*AUTORES: E.MARTÍN CUENCA, J.M. ANGULO USATEGUI, I. ANGULO MARTÍNEZ ; EDITORIAL: S.A EDICIONES PARANINFO. THOMSON LEARNING*]

DISEÑO PRÁCTICO CON MICROCONTROLADORES [AUTORES: SUSANA ROMERO YESA, J.M. ANGULO USATEGUI, IGNACIO ANGULO ; EDITORIAL: S.A EDICIONES PARANINFO

https://www.traxco.es/blog/pivotes-de-riego/el-pivot-una-fabrica-de-lluvia

www.velleman.eu

https://hetpro-store.com/TUTORIALES/joystick-analogico-programado-con-arduino/

http://tmrh20.github.io/RF24/

http://www.microchip.com/

www.niplesoft.net

http://inven.es

www.prometec.net

http://inven.es/motores-y-servos/192-modulo-driver-pap-dual-bridge-dc-l298n-puente-h.html

http://www.es.co.th

http://www.micropik.com/PDF/SG90Servo.pdf

https://programarfacil.com/tutoriales/fragmentos/servomotor-con-arduino/

http://www.ukai.com/LMB204BFC-1-display-lcd-alfanumerico-matricial-20x4/

https://electronilab.co/

www.brikogeek.com

http://www.todomicro.com.ar/sensores

http://www.alldatasheet.com

https://www.vellemanusa.com

https://www.theengineeringprojects.com/2016/08/new-proteus-libraries-engineering-students.html

www.traxco.es

15. ESQUEMAS

AGRADECIMIENTOS

En primer lugar, agradecer a mi familia todo el apoyo y la confianza aportados durante todo el periodo universitario, sin el cual, no hubiera sido posible finalizarlo.

Igualmente, todo el apoyo recibido por parte de mi pareja, Montserrat Maestre Rey, la cual ha sabido aguantar todos y cada uno de los buenos y malos momentos que han sucedido a lo largo de este tiempo, apoyándome incondicionalmente.

Agradecer a los profesores y personal de los laboratorios de Sistemas Digitales y de Tecnología Electrónica del CASEM, adscritos al Departamento de Ingeniería en Automática, Electrónica, Arquitectura y Redes de Computadores, todos los conocimientos que han sabido transmitirme, no solo durante la realización de este proyecto, sino durante todo el grado. Así mismo, agradecerles los materiales y herramientas facilitados para la construcción de este.