### Taller de Proyecto II

2017

“CarCar”

Proyecto N° 10

Integrantes:

* Becerra Agustín - 053/2
* Buscaglia Joaquín - 238/7
* Molina Rodrigo - 404/3

1. **Propuesta Original del Proyecto**

**[Describir la propuesta original hecha a la cátedra en esta sección, sobre todo desde el punto de vista funcional y agregar la propuesta original completa como Apéndice A de este informe]**

El proyecto que se desea diseñar es un auto de tres ruedas en las que una de ellas será de dirección y tracción, y las otras dos serán libres. En la rueda principal, la dirección estará determinada por un servo y la tracción a través de un motor de corriente continua que serán gobernados por el micro-controlador. Además se desea dar la posibilidad al usuario de controlar un auto mediante un navegador web el cual se comunicara con el micro controlador montado en el auto a través de comunicación wifi.

1. **Correcciones/Cambios de la Propuesta**

**2.1. Indicadas por la Cátedra**

**[Tal como las recibieron, en el caso de haber aclaraciones, compaginarlas]**

En la presentación del proyecto hubo una inconsistencia entre la introducción y el objetivo principal. En la cual la introducción especificaba que el auto constaría de una rueda de dirección y dos de tracción, cuando el proyecto original indicado por la cátedra establecía que una de ellas sería tanto de tracción como de dirección.

* 1. **Definidas por el Avance/Disponibilidad**

**[Enunciarlas y explicar las razones correspondientes]**

**[Tener en cuenta que de lo que describan en la sección anterior y en ambas subsecciones de esta sección debería quedar claro el proyecto entregado]**

El principal aspecto que no fue considerado en la propuesta original fue el desarrollo del hardware que servirá como medio de conexión entre el micro-controlador con los diferentes componentes (servo, motor, fuente de alimentación, y módulo wifi). Para llevar a cabo la solución más prolija, se pensó en crear una placa PCB con las conexiones requeridas para cada uno de los módulos así como para suministrar una alimentación acorde a cada uno.

Además, en un principio se realizó las conexiones para el motor de corriente continua utilizando directamente uno de los pines del micro-controlador definidos como PWM, lo cual nos daba una velocidad final de la rueda muy pequeña, y que ante cualquier peso asociado no podría mantener el movimiento requerido, por lo que se debió implementar un puente H, el cual podrá variar el voltaje de alimentación del motor desde 0 a 12v con una variación de la señal de control producida por el micro-controlador desde 0 a 5v de valor medio.

1. **Descripción de Hardware y Conexiones**

**[Diagrama esquemático de conexiones – cableado. Asociado a cada esquemático de conexiones agregar una foto del sistema real en la que se vean las conexiones del esquemático tal como las implementaron]**

**[Debería mostrarse un esquemático general, describiendo todo el sistema, con su foto asociada, y un esquemático por cada subsistema, en particular el que involucre a la placa de desarrollo utilizada]**

**[Opcionalmente, se puede agregar un video en el que se describa todo lo anterior. Este video opcional no reemplaza lo anterior sino que lo complementa. Para los casos en que el sistema web no permita subir el video, incluir en este apartado el enlace de algún sistema de almacenamiento de videos (youtube, vimeo, etc.) o archivos (dropbox, Mega, etc.) desde donde se pueda ver y/o descargar, asegurándose que no sea necesario ningún tipo de información o tarea extra como clave, registro en el sistema, etc.]**

Para establecer las conexiones necesarias para el funcionamiento del sistema (Figura 1 y 2), se desarrolló una placa PCB para que haga de interface entre el micro-controlador y los módulos o dispositivos externos utilizados.

Dicha PCB fue desarrollada con la herramienta de diseño Eagle y consta de un puente H para controlar la velocidad y dirección de giro del motor CC, donde mediante una señal de control de 0 a 5v de valor medio producida por el micro-controlador, se debe generar una señal que valla de 0 a 9v para abarcar todo el rango de velocidades posibles para el giro del motor.

Por otro lado se establecieron los pines para las conexiones del servo y el módulo wifi, donde para el servo-motor se debió asignar tres pines de los cuales dos se utilizaran para alimentación y el restante para la señal de control que definirá la posición a la que debe dirigirse. Además, la alimentación será independiente de la utilizada para los demás dispositivos con el fin de no sobrecargar los pines del micro-controlador.

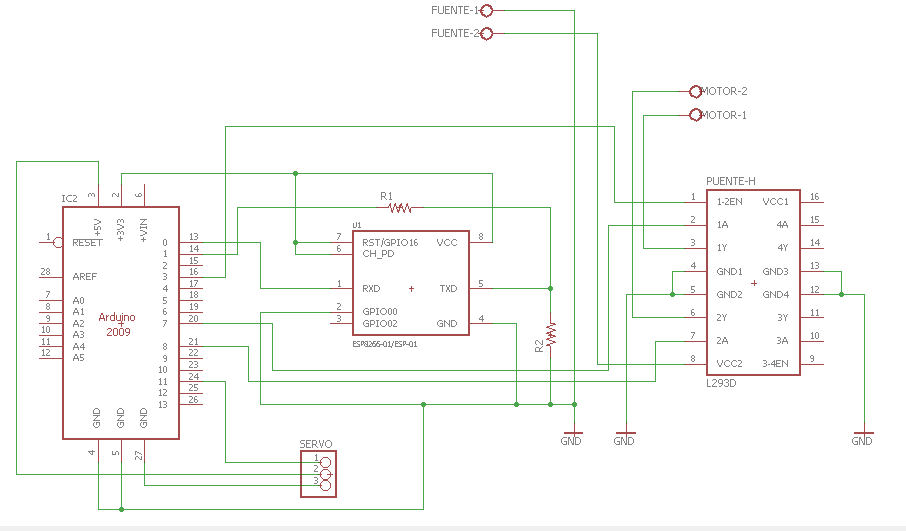


Figura 1.

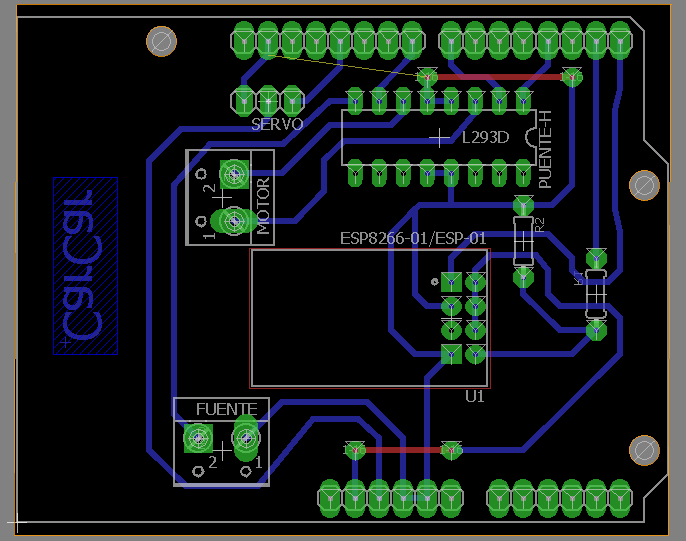


Figura 2

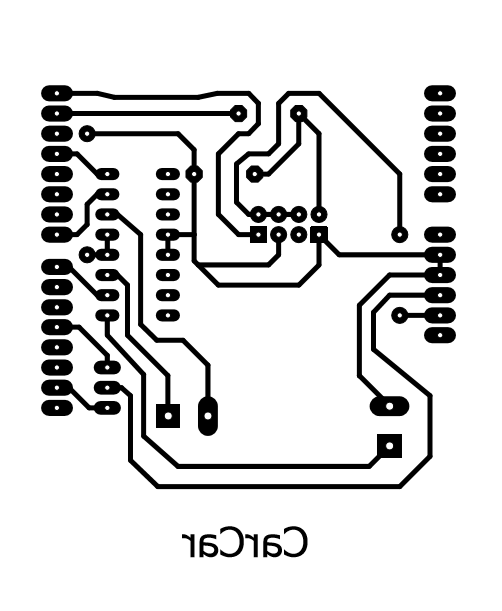


Figura 3. Gráfico de la placa PCB.

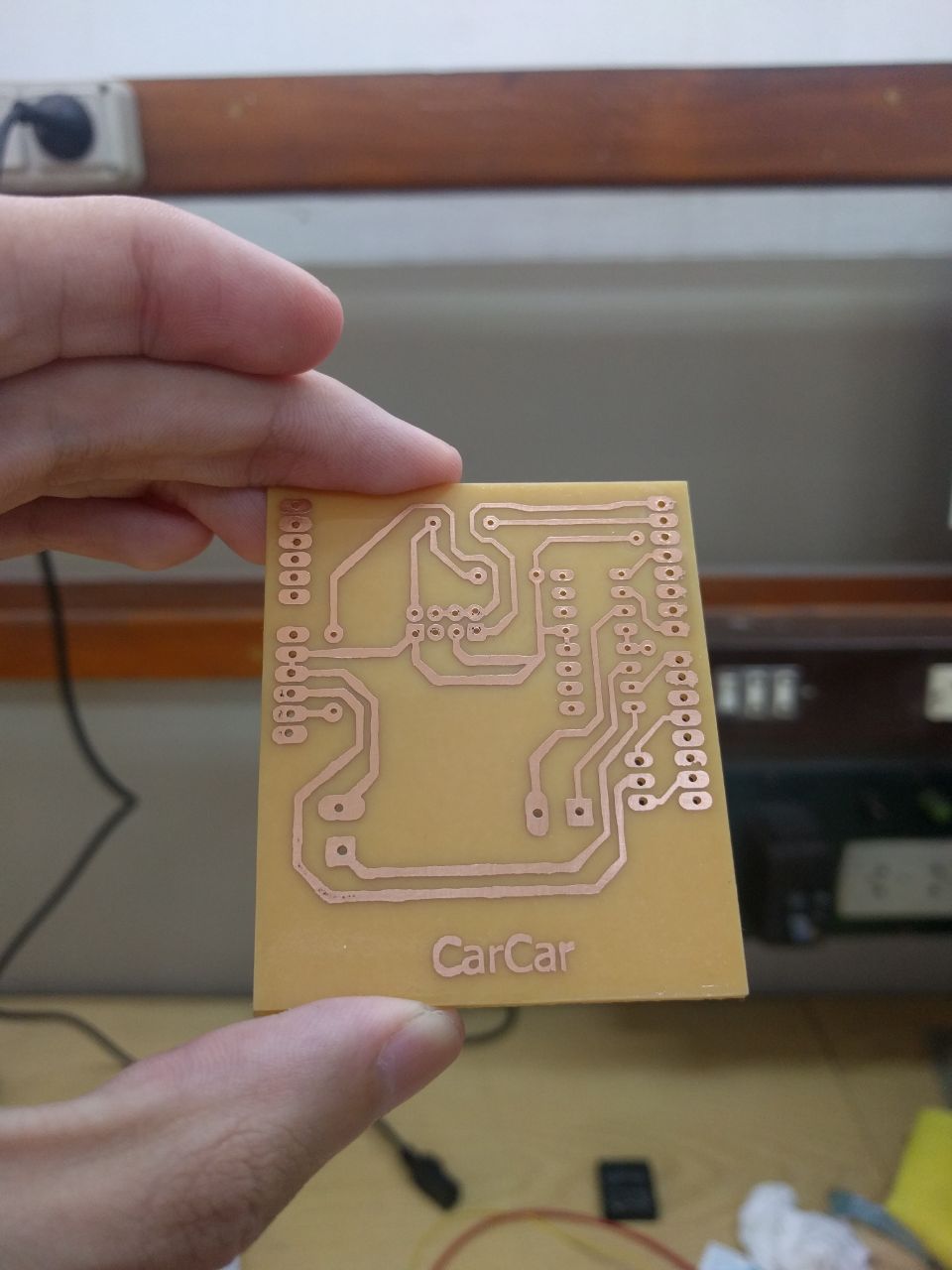


Figura 4. Placa PCB.

En el caso del puerto utilizado para el módulo wifi ESP8266, se realizaron las conexiones de alimentación utilizando el puerto del micro-controlador definido para este fin, aprovechando así la referencia a 3.3v y evitando la necesidad de generarla. Además, como el ESP tiene un voltaje de alimentación menor al del Arduino UNO, también los valores utilizados para la comunicación serie entre ellos serán diferentes, por lo que se debió utilizar un divisor de tensión en la línea de comunicación desde el Micro-controlador al módulo wifi.

Para poder variar la alimentación del motor CC de 0 a 9V utilizando un pin PWM del micro-controlador, se asoció puente H, descripto en la figura 2, a dos pines diferentes, donde si el primero se pone en alto, los transistores Q1 y Q4 permitirán el paso de corriente por ellos, por lo que el motor quedara conectado en directo (figura 3), y en caso de que el que este activo sea el pin 2 los transistores Q2 y Q3 son los que permitirán el paso de corriente y establecerán una conexión en forma inversa en los puertos del motor (Figura 4). Además, se debe tener en cuenta que nunca deberán estar el alto los dos pines conectados al puente H a la vez, ya que esto habilitaría el paso de corriente por todos los transistores a la vez, lo que se traduciría en un corto circuito en la fuente.

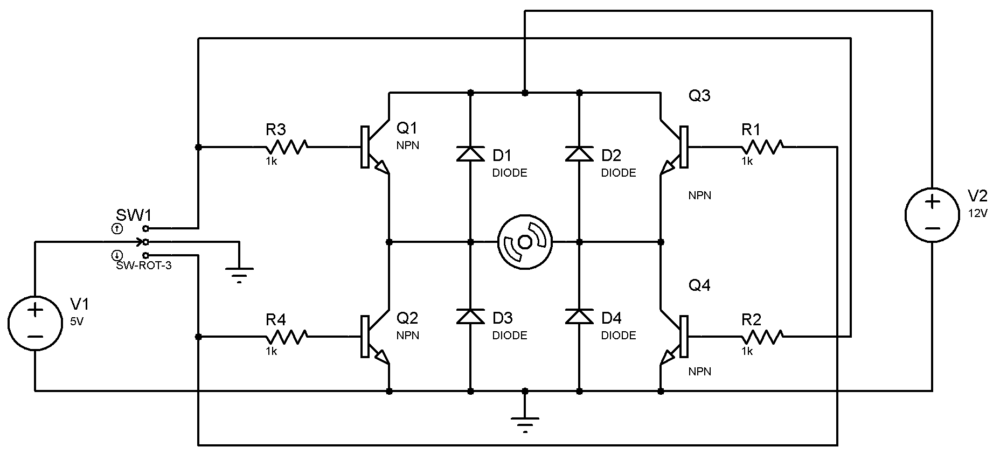


Figura 5.

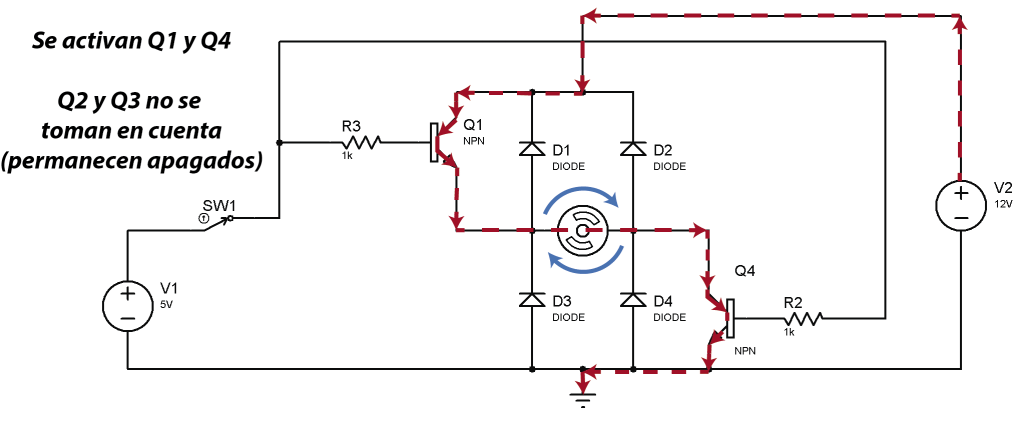


Figura 6.

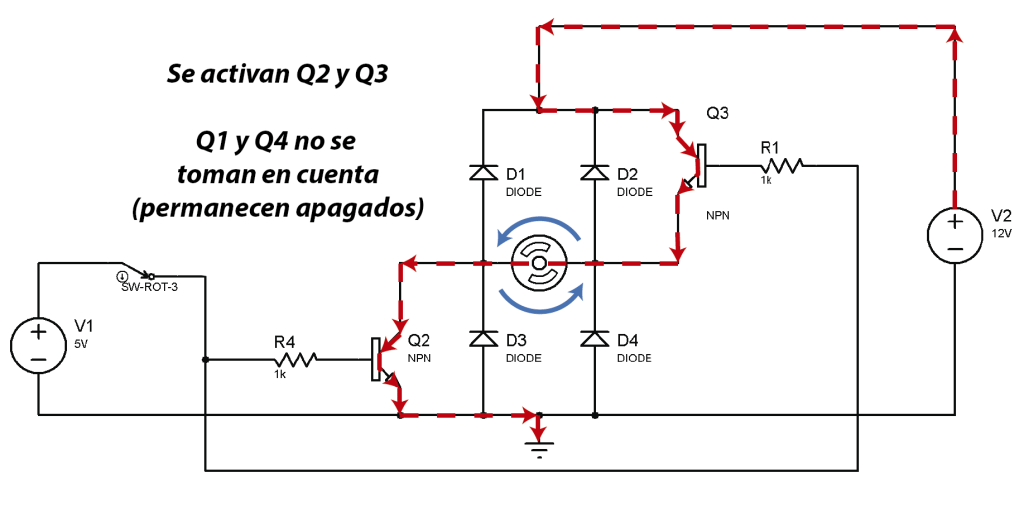


Figura 7.

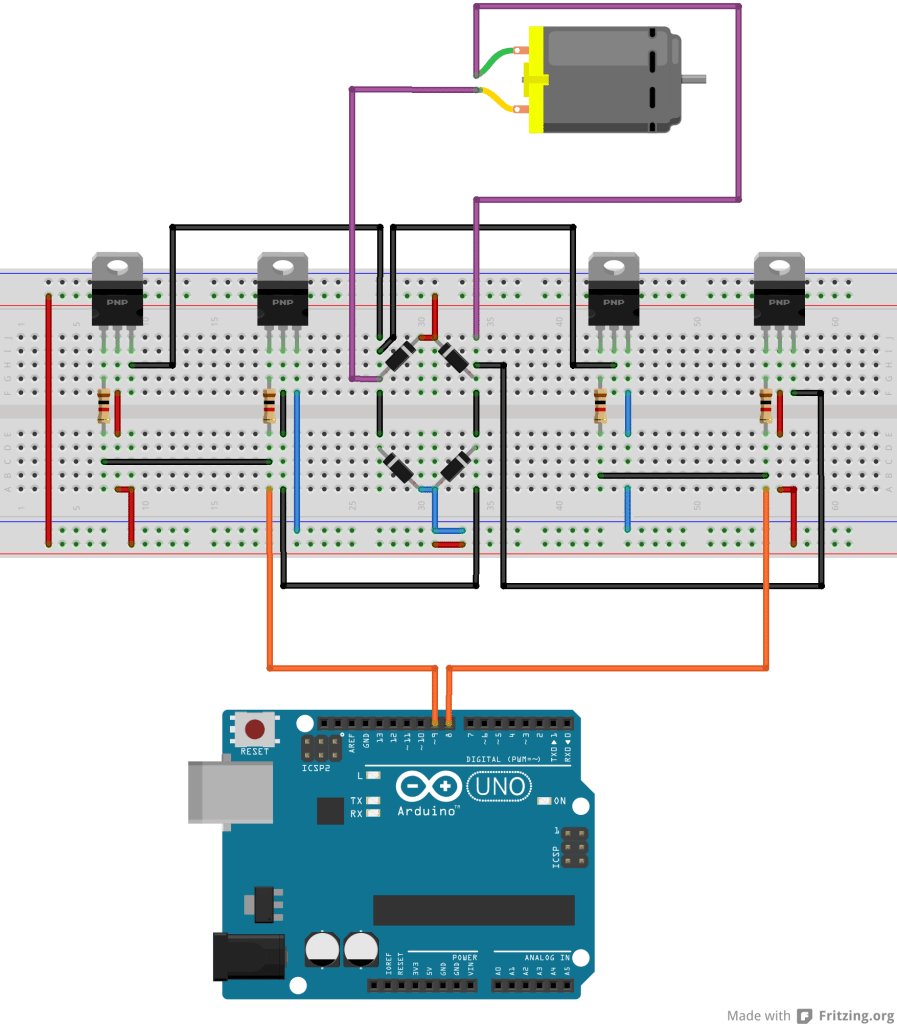


Figura 8.

En la figura de arriba (Figura 5) se muestra la conexión del puente H, el motor y los pines del Arduino UNO utilizados para el control del mismo, que fueron mencionados anteriormente utilizando una protoboar para probar su funcionamiento. A continuación, se deja detallada la ruta de donde se obtuvo las figuras ilustradas: <http://panamahitek.com/el-puente-h-invirtiendo-el-sentido-de-giro-de-un-motor-con-arduino/>

En nuestro caso no realizamos el circuito del puente H, sino que utilizamos el circuito integrado llamado L293D, **Figura N+3**, el cual posee 2 puentes H, nosotros utilizamos uno solo de ellos, 8 de 16 pines.

* Pin 1: Enable, es el que habilita y deshabilita el puente H, a este se le conecta la señal PWM para establecer el ciclo de trabajo y la tensión que se aplica en el motor.
* Pin 2 y 7: 1A y 2A, son pines digitales que se utilizan para control para manejar la dirección de giro del motor, dependiendo de cual se ponga en alto y cual en bajo
* Pin 3 y 6: 1Y y 2Y, son los pines de salida al motor
* Pin 4 y 5: GND
* Pin 8: la fuente externa con la que se alimenta al motor

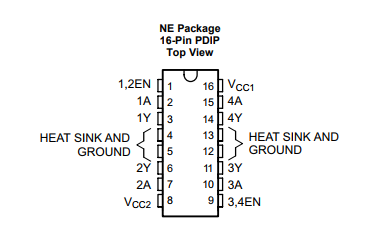


Figura 9. Puente H L293D

**Nuevo diagrama de conexiones**

Por recomendaciones de la cátedra, y teniendo en cuenta que el dispositivo no será producido en serie, se descartó el uso de la PCB y se realizaran las conexiones utilizando cables de electrónica provistos por el profesor. En primer lugar se debe proceder a la configuración del módulo wifi, para lo que será necesario la conexión de dicho modulo y el micro-controlador en modo de configuración (figura 10), para que se pueda guardar el sketch en la memoria flash del ESP8266. Dicho modo de conexión se caracteriza por tener el pin correspondiente al GPIO0 del modulo conectado a GND, y por las líneas de comunicación conectadas directamente entre Rx y Rx, y entre Tx y Tx, del módulo wifi y del micro-controlador respectivamente.

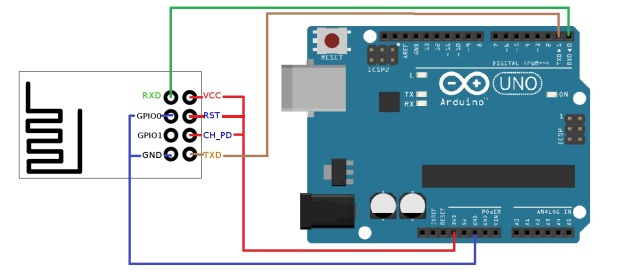


Figura 10.Conexión de configuración del módulo ESP8266 con Arduino Uno.

Una vez que el sketch que se ejecutara en el módulo wifi y la página web estén alojadas en el ESP8266, se procede a realizar las conexiones definitivas para el funcionamiento de los dispositivos en conjunto, donde se establecen las conexiones guiádos por el esquemático siguiente:

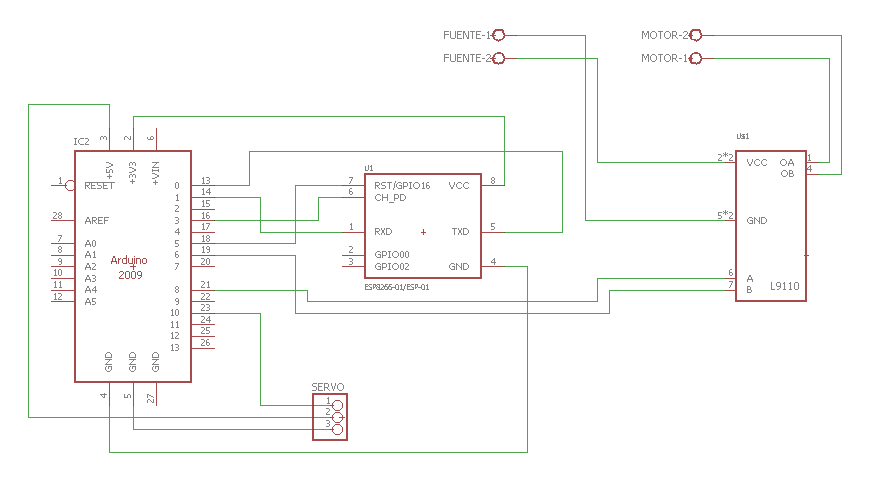


Figura 11. Diagrama esquemático de conexiones definitivas.

A continuación se mostraran los diferentes esquemas de las conexiones definitivas de los subsistemas integrados en el esquemático anterior, en primer lugar se especifica un grafico simbólico de la conexión del módulo wifi y el micro-controlador, donde se caracterizan la asociación del Rx de un dispotivo, con el Tx del otros (y viceversa), además de debe dejar el GPIO0 del ESP8266 al aire, para que este no entre en modo de configuración.

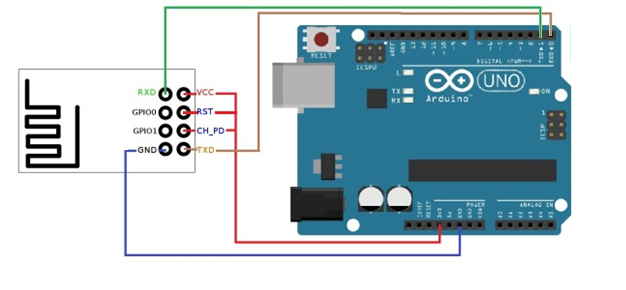


Figura 11.Conexión definitiva del módulo ESP8266 con Arduino Uno.

Con respecto a la conexión del motor, se deberá utilizar un módulo de puente H para aumentar el valor medio de tensión que llega al motor y así obtener una mayor velocidad final. En este caso se dispone del módulo L9110s, al cual se lo alimenta con una fuente de alimentación externa de 9V y se conecta los pines de control como indica la figura 12, donde uno de ellos indicará la dirección de movimiento del motor, y el otro será una señal PWM que establecerá el valor de velocidad final.

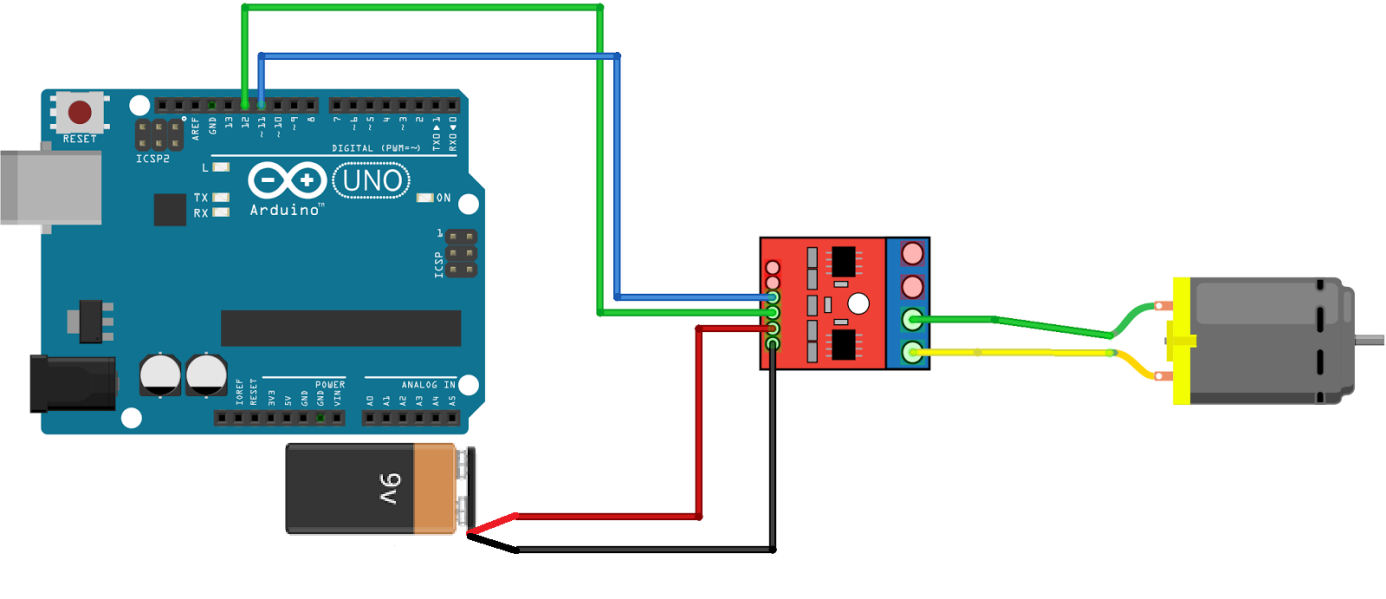


Figura 12. Conexión de motor con Arduino UNO a través de puente H L9110s.

Para terminar se puede ver la conexión del servo-motor con el micro-controlador utilizado, donde además de la alimentación se puede observar una línea de control, que asociara uno de los pines capaces de generar una señal PWM, a la entrada del control del dispositivo, la cual dependiendo de su ciclo de trabajo establecerá la posición final del mismo.

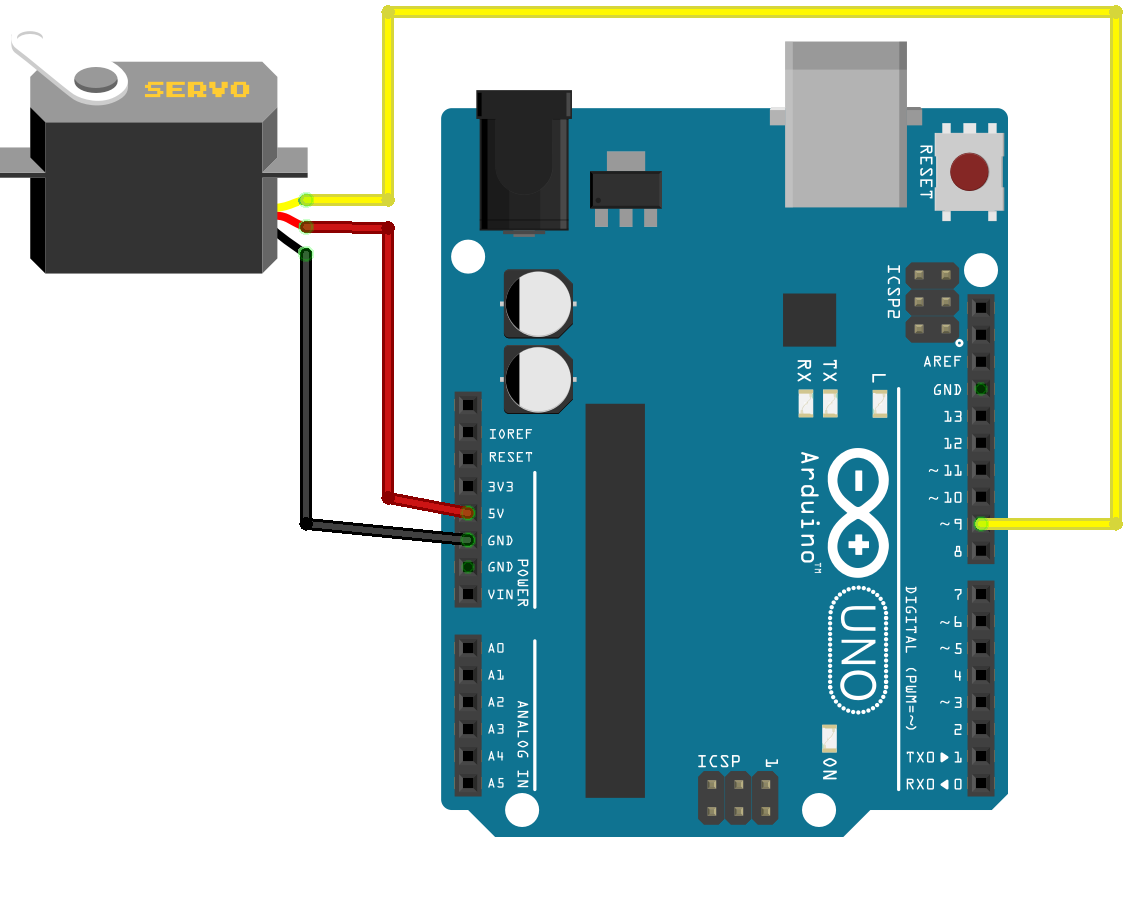


Figura 13. Conexión del servo-motor con Arduino UNO.

1. **Descripción Funcional**

**[Descripción funcional general, asociando las funciones a procesos, software, comunicaciones o hardware utilizado en el proyecto]**

**[Describir casos de uso o sucesiones de eventos que se manejan o monitorizan, en qué partes de hardware o software generan qué operaciones y las comunicaciones involucradas. Al menos una de las descripciones de casos de uso o sucesiones de eventos deben incluirse capturas de pantalla del sistema en funcionamiento y un video de la secuencia completa. Para los casos en que el sistema web no permita subir el video, incluir en este apartado el enlace de algún sistema de almacenamiento de videos (youtube, vimeo, etc.) o archivos (dropbox, Mega, etc.) desde donde se pueda ver y/o descargar, asegurándose que no sea necesario ningún tipo de información o tarea extra como clave, registro en el sistema, etc.]**

**[Identificar claramente el hardware utilizado y para qué tareas se utiliza, el software utilizado y qué resuelve y la o las comunicaciones involucradas. Detallar la funcionalidad específica del subsistema web con el usuario y el resto de subsistemas del proyecto]**

El auto que se quiere desarrollar consta de un Arduino uno, al que se le van a asociar un servo motor, un motor CC y un módulo wifi (esp8266), también se utilizara para la alimentación una batería de 9 volt, además para obtener y procesar los datos de tracción y dirección se deberá establecer una comunicación con el usuario mediante una página web que estará alojada en un servidor levantado en el módulo wifi.

Con respecto a la página web, presentará una interfaz para el usuario que constara de cinco botones donde dos serán para la dirección, que llebaran al servo a derecha o izquierda, otro de parada que al presionarse detendrá el movimiento del motor, independientemente de su sentido , y por ultimo otros dos para elección del sentido de giro del motor, definiendo la dirección en que se moverá el auto, y por consiguiente la velocidad del mismo, que dependerá de cuantas veces se presionen dichos botones, por ejemplo, si se presiona dos veces en el mismo sentido, sin haber presionado ninguna vez en el sentido contrario.

Los datos enviados desde la página web a través de un formulario utilizando el protocolo 802.11 (Wifi), que serán recibidos por su respectivo módulo y luego enviado al micro-controlador a través de una comunicación serie, donde se traducirá lo que el usuario quiera realizar de manera de generar las señales de control, tanto para el servo como para el motor.

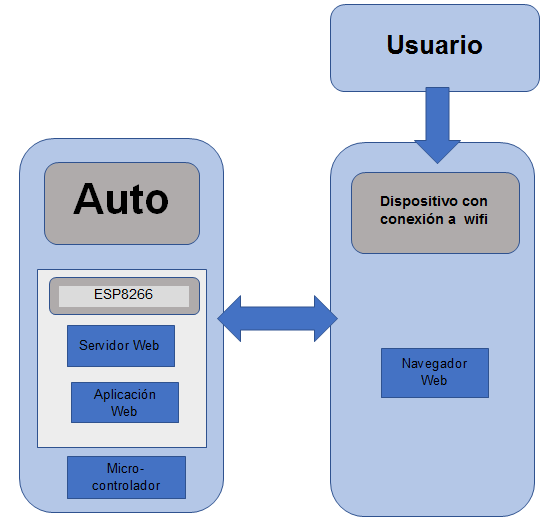


Figura 14. Diagrama funcional del sistema web.

Para generar las señal de control que indicara al servo (HOBBICO CS-51) su posición final se debe generar un PWM, donde dependiendo del ciclo de trabajo se especificara un valor entre 0 y 180 grados, que es el rango de posiciones posibles que puede adoptar el mismo.

Con respecto al control del motor, será necesaria la configuración de dos pines como salida, uno para generar una señal PWM que al variar su ciclo de trabajo modificará el valor medio entregado al módulo L9110 (puente H), por lo que se concluye que la velocidad aumente o disminuya. Y otro pin de control asociado al módulo que definirá el sentido de giro de motor, donde con un valor bajo se especifica que el auto vaya hacia adelante y con uno alto hacia atrás.

1. **Descripción del Software**

**[Descripción de las funciones/rutinas del sistema web, asociarlas a los casos de uso o sucesiones de eventos que se describieron en la sección anterior. Agregar todo el código del sistema web como Apéndice B de este informe (usar tamaño pequeño de letra, aunque legible, y de espaciado constante o “monoespaciado” como Courier).]**

**[Descripción de las funciones/rutinas del software que se ejecuta en la placa de desarrollo, asociarlas a los casos de uso o sucesiones de eventos que se describieron en la sección anterior. Agregar todo el código del software/sketch que se ejecuta en la placa de desarrollo como Apéndice C de este informe (usar tamaño pequeño de letra, aunque legible, y de espaciado constante o “monoespaciado” como Courier).]**

**[Una de las mejores opciones para la descripción del software es el pseudocódigo, en el que se describen conceptualmente las operaciones del código sin llegar a copiar el código fuente propiamente dicho]**

**Sketch modulo ESP8266**

Para la configuración del modulo wifi (ESP8266) como punto de acceso se debe cargar el esquema ”codigoWIFI.ino” que se adjunta en el apéndice B, para lo que es necesario asegurarse de tener seleccionada, en la IDE de arduino, la placa de desarrollo denominada “Generic ESP8266 Module”, además de encontrarse en el modo de conexión para configuración, del modulo wifi.

Este sketch es cargado en la memoria flash del ESP8266 y se encargará de establecer la red y asociar a ella la pagina web con la que el usuario interactuará, además de recolectar los datos de control asociados a dicha interaccion para enviarlos, mediante una comunicación serie (RS232), al micro-controlador. Para dicho objetivo fueron utilizadas y referenciadas las siguientes librerías:

#include <SoftwareSerial.h>

#include <WiFiClient.h>

#include <ESP8266WiFi.h>

#include <ESP8266WebServer.h>

#include "index.h"

Donde la primera contiene las funciones utilizadas para la comunicación serie del modulo con el microcontrolador, y las tres siguientes son usadas para establecer un punto de acceso, levantar un servidor y dar soporte para una pagina web. Ademas se incluye la referencia al documento que contendrá el código en lenguaje HTML que será interpretado por el navegador de los dispositivos que accedan a el.

Para operar necesitamos crear un objeto de la clase ESP8266WebServer cuya función es servir el contenido cuando accedamos con el navegador, definiendo como puerto a utilizar el numero 80.

ESP8266WebServer server(80);

Para terminar con la inicialización de variables globales definimos dos cadenas donde vamos a guardar el nombre de nuestra red WiFi y la contraseña para que el ESP8266 pueda conectarse:

const char\* ssid = "AutoTPII";

const char\* password = "12345678";

A continuación se define un procedimiento que se ejecutará cuando el usuario interactúe con la página web, el cual se encargará de realizar el intercambio de información con el dispotivo utilizado para controlar el auto, así como con el micro-controlador. Para esto en primer lugar el objeto servidor creado anteriormente envía el archivo que deberá ser interpretado por el navegador utilizando la función send() provista por la biblioteca ESP8266WebServer.h, la cual recibe tres argumentos, donde el primero se trata del código de estado HTTP, y se define en 200 que corresponde a la respuesta “OK”, en segundo lugar se especifica la URI(uniform resource identifier) o identificador del recurso uniforme, que hace referencia al tipo de dato que será enviado, y por último se envía una correspondencia al documento a mostrar, en este caso el principal o cabecera, el cual estará asociado al archivo index antes mencionado.

Luego se espera la respuesta del navegador utilizando la función arg(), la cual devuelve una cadena de caracteres correspondiente a la tecla presionada por el usuario del sistema, y recibe como argumento un string con el valor del campo enviado desde el formulario. Para terminar se envía el dato recibido, por el servidor web al arduino, mediante el puerto serie, utilizando la función write provista por SoftwareSerial.h.

void handleRoot() {

String s = MAIN\_page;

server.send(200, "text/html", s);

String dato = (server.arg("Direccion"));

Serial.write(dato[0]);

}

Por otro lado se define la función de inicialización en la que se debe configurar cada uno de los subsistemas utilizados, en primer lugar el objeto que modela el puerto serie que se comunicara con el micro-controlador, a la que se le setea una velocidad de transferencia de datos de 9600 baudios, mediante la utilización de la función begin, que a su vez asigna a esta instancia del objeto SoftwareSerial el control de los pines físicos establecidos para la comunicación.

En segundo lugar se utiliza la biblioteca ESP8266Wifi.h para la configuración del modulo ESP8266, donde enviando la referencia WIFI\_AP a la función mode() se establece el modo de funcionamiento del dispositivo como punto de acceso, para el cual tambien se debe definir el nombre y la contraseña de la red wifi creada, por lo que es necesario referenciar a la función softAP con las constantes definidas globalmente.

Para terminar con la inicialización se hace uso de la función on que asocia la página raíz del servidor montado en el sistema, con la función handleRoot definida anteriormente, que dara soporte a la pagina web. Y con un llamado de la instancia que modela el servidor a la función begin, se establece la red pudiendo ser accedida por cualquier dispositivo móvil en su área de trabajo.

void setup(void){

Serial.begin(9600);

WiFi.mode(WIFI\_AP);

WiFi.softAP(ssid, password);

server.on("/", handleRoot);

server.begin();

}

Solo falta definir la función loop() que será ejecutada por el modulo wifi mientras el sistema esté en funcionamiento, ésta solo se encargara de escuchar las conexiones entrantes, por lo que solo será necesario hacer uso de la función handleClient() por parte de nuestro objeto servidor, la cual verifica si se recibe una nueva solicitud HTTP y si detecta alguna ejecutara las funciones que se especificaron en la inicialización.

void loop(void){

server.handleClient();

}

**Sketch Arduino UNO**

El codigo ejecutado por el micro-controlador se debe encargar de recibir los datos enviados por el modulo wifi y traducirlos en señales de control para el motor o el servo según corresponda. Para el control del servo se declara la biblioteca Servo.h que viene incluida en la IDE de arduino, además se definen los pines de la placa de desarrollo utilizados para realizar las conexiones, y los valores que se deben enviar al servo para cada una de sus posibles posiciones. Por ultimo se declaran las variables globales utilizadas por el programa, donde además de las utilizadas para conservar el estado de los subsistemas, se define una para contener la instancia del servo que modelara dicho dispositivo y otra para la próxima instrucción a ejecutar.

#include <Servo.h>

#define MOTOR\_PIN 6

#define MOTOR\_CONTROL\_PIN 8

#define SERVO\_PIN 10

#define SERVO\_IZQUIERDA 70

#define SERVO\_CENTRO 90

#define SERVO\_DERECHA 110

Servo servo; // crea el objeto servo

int nuevoComando = 0;

int estadoDireccion = 0;

int estadoTraccion = 0;

int velocidad = 0;

Con respecto al proceso de inicialización se comienza llamando a la función Serial.begin() para iniciar el puerto serie para los pines por defecto (Tx=0, Rx=1), donde se establecerá la comunicación con el modulo wifi. Luego se fija el modo (OUTPUT) y se definen los pines para la señal de control de velocidad y dirección del motor, a los que se les setea cero y low respectivamente para iniciar el auto con el motor detenido y dispuesto a ir hacia adelante. Ahora solo falta configurar el servo, para lo que se debe llamar, con la instancia del dispositivo creada previamente, a la función attach, con el valor del pin definido para la señal de control, y luego usando write se envia la posición inicial en que se encontrara el dispositivo luego de la inicialización.

void setup() {

Serial.begin(9600);

pinMode(MOTOR\_PIN, OUTPUT);

pinMode(MOTOR\_CONTROL\_PIN, OUTPUT);

analogWrite(MOTOR\_PIN, velocidad);

digitalWrite(MOTOR\_CONTROL\_PIN, LOW);

servo.attach(SERVO\_PIN);

servo.write(SERVO\_CENTRO);

}

Por último el procedimiento loop(), que se ejecuta indefinidamente en el Arduino UNO, este basa su funcionamiento en realizar pooling sobre el puerto serie, utilizando la función available(), y esperando la existencia de un dato proveniente del módulo wifi el cual procesar. Una ves que el dato se encuentra en el buffer del puerto serie, está disponible para ser leído utilizando la función read(), por lo que se guarda en una variable global de tipo entera, para ser posteriormente evaluado por una estructura de control de selección (switch-case) y determinar la acción a realizar dependiendo del valor obtenido.

void loop() {

if (Serial.available() > 0) {

nuevoComando = Serial.read();

Serial.write(nuevoComando);

switch(nuevoComando){

case 'D':

...

break;

case 'I':

...

break;

case 'R':

...

analogWrite(MOTOR\_PIN, velocidad);

break;

case 'A':

...

analogWrite(MOTOR\_PIN, velocidad);

break;

case 'N':

estadoTraccion = 0;

velocidad = 0;

digitalWrite(MOTOR\_CONTROL\_PIN, LOW);

analogWrite(MOTOR\_PIN, velocidad);

break;

default :

break;

}

}

}

1. **Guía de Instalación Completa**

**[Siguiendo esta guía paso a paso debería ser posible reconstruir el proyecto completo casi sin conocimientos previos de los detalles involucrados. Esta guía es esencial no solamente para reconstruir el proyecto sino para su continuación/evolución]**

**a) Ambiente de desarrollo:**

**Debería documentarse desde la instalación de drivers necesarios hasta el software de desarrollo (IDE y/o lenguajes necesarios) y lo desarrollado (programas fuente), para que el proyecto se pueda reproducir, mantener, modificar, mejorar y/o agregar funcionalidad, etc. A partir de lo documentado en esta sección debería ser posible que otro grupo de trabajo retome el proyecto a partir de lo entregado. En el caso de los programas fuente, no incluir el listado, solamente la enumeración de lo desarrollado y en qué ambiente debería ser utilizado. En el caso de software que no sea de proveedores conocidos o que tengan alguna posibilidad de no mantenerse en los sitios web, descargarlos y adjuntarlos al presente informe. Arduino es un ejemplo de sitio que ha perdurado en el tiempo, con lo cual no sería necesario.**

Para descargar la última versión de la IDE de Arduino se debe ingresar a la siguiente dirección: https://www.arduino.cc/en/main/software y elegir la versión específica para su sistema operativo.

Una vez descargado se debe ejecutar el archivo y se podrá ver la siguiente ventana.

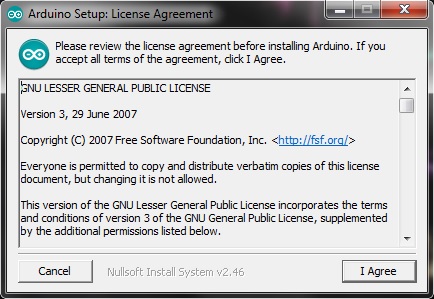


Figura 13.

Pulsamos en "I Agree". En este cuadro dejamos todas las opciones marcadas, pero hay que prestar especial atención a la opción de instalar los USB driver, ya que esto es muy importante para que la placa Arduino se pueda comunicar con la PC. Pulsamos en Next e Install.

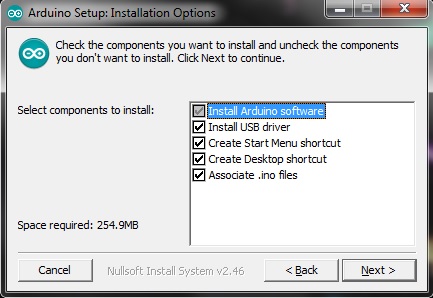


Figura 14.

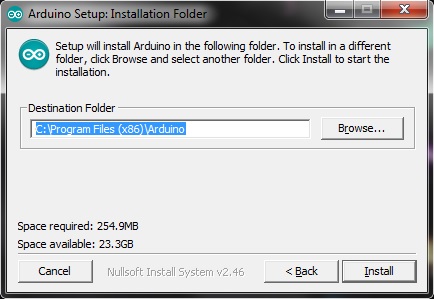


Figura 15.

Y esperaremos que termine de instalar (si pregunta si deseamos instalar el software Arduino USB le damos a instalar).

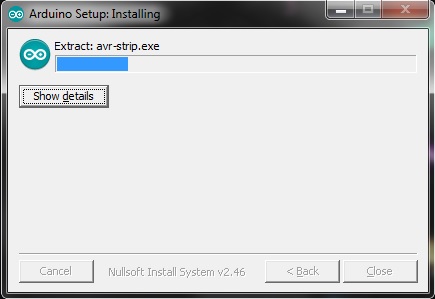


Figura 16.

Una vez terminado el proceso, hacemos clic en Close y ya tendremos el IDE instalado en nuestra PC.

Además fue necesaria la instalación de dos librerías adicionales a las que vienen por defecto con la IDE de Arduino (ESP8266WiFi.h y ESP8266WebServer.h) para poder configurar y utilizar el módulo ESP8266 sin recurrir al uso de comandos AT, dicha biblioteca se encuentra disponible en la siguiente direccion:

<https://github.com/esp8266/Arduino>

Debe descargar y descomprimir el repositorio, luego ingresar en la carpeta “libreries” (Arduino-master🡪libraries) y buscar las librerías a utilizar, en nuestro caso la ESP8266WIFI y la ESP8266WebServer.

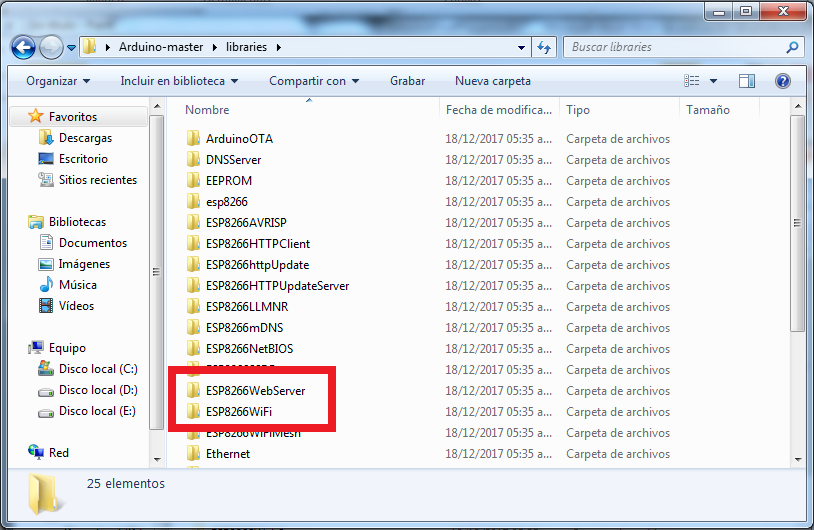


Figura . Bibliotecas utilizadas para modulo wifi.

Luego deben copiarse estos archivos en la carpeta de librería de la IDE de Arduino C:\Program Files (x86)\Arduino\libraries en nuestro sistema.

Por otro lado, hay que agregar al gestor de tarjetas de la IDE de Arduino la correspondiente al módulo esp8266, esto se lleva a cabo en el menú del programa, accediendo a Archivo 🡪 Preferencias, en el campo de Gestor de URLs Adicionales de tarjetas (figura 17) se debe copiar el siguiente enlace:

<http://arduino.esp8266.com/versions/2.3.0/package_esp8266com_index.json>

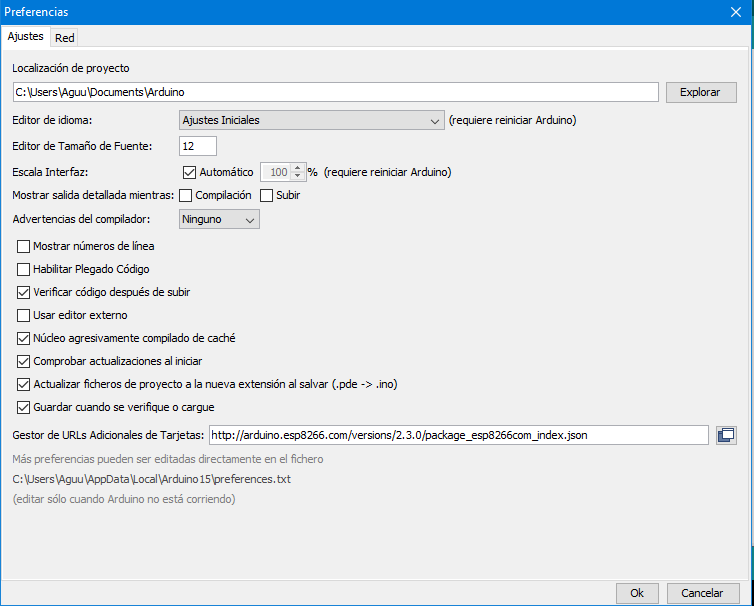


Figura 17.

Luego ir a la pestaña de herramientas, hacer click en la placa a utilizar, buscar la ESP8266 y dar click en instalar (figura 18), espere hasta que se termine el proceso y cierre la ventana.

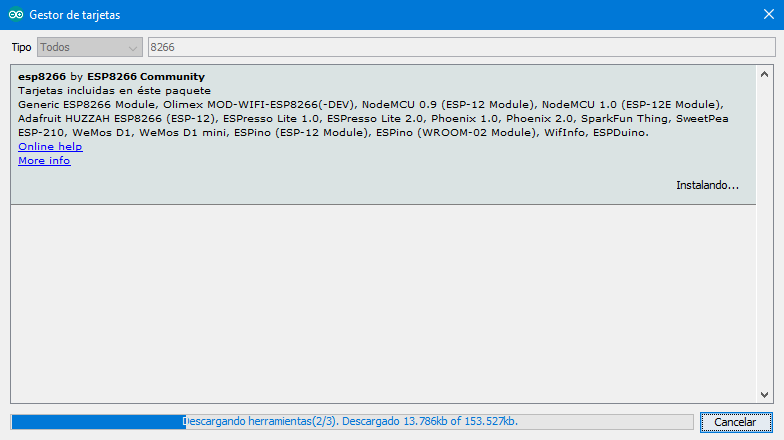


Figura 18.

**b) Copia/Instalación de Código:**

**Identificar qué archivos corresponden a qué hardware o subsistema (ej: web en la PC y sketch en Arduino). Cada una de estos archivos deben ser adjuntados al presente informe. Notar que el mismo software se ha incorporado en/como Apéndices, pero en este caso debe ser adjuntado (como directorio/s y/o archivo/s en .zip o .tar.gz) con el objetivo de ser reconstruido el sistema de desarrollo.**

Los sketch del Motor CC y el Servo-motor estarán alojados en el Arduino como también el sketch del módulo wifi, el cual una vez inicializado el micro-controlador configurara el modulo como punto de acceso a través de un puerto serie, alojando en el módulo (ESP8266) la página y el servidor web.

Estos mismo archivos estarán adjunto al informe a entregar a través de un archivo.zip llamado Sketches.zip.

**c) Guía de Compilación/Instalación/Upload de Ejecutable/s:**

**Indicar configuraciones y ubicaciones de los ejecutables y código del sistema en el caso del subsistema web y si hay alguna consideración o configuración especial para el código que se ejecute en la placa de desarrollo.**

Con respecto a la página web, ella estará definida en sentencias HTML y alojado en un archivo con extensión punto h, para ser correctamente interpretado por el

1. **Consideraciones Generales**

**El informe es la parte central de la documentación o de hecho contiene todo lo referido a la documentación del sistema.**

**El informe debería contener 3 Apéndices (A, B y C) tal como se ha especificado antes**

**La entrega del proyecto debería contener:**

**- Todo el hardware conectado y listo para usar**

**- El informe completo, con sus Apéndices incluidos. En el caso de los videos indicados antes, pueden estar adjuntos o directamente los enlaces desde donde bajarlos en las condiciones que se especificaron antes.**

**- El código fuente adjunto, listo para ser copiado, compilado y ejecutado en los subsistemas correspondientes.**

**Apéndice A**

# **Introducción**

El proyecto que se desea diseñar es un auto de tres ruedas en las que una de ellas será de dirección y las dos restantes de tracción, donde la dirección estará determinada por un servo y la tracción a través de dos motores de corriente continua que serán gobernados por el micro-controlador. Además se desea dar la posibilidad al usuario de controlar un auto mediante un navegador web el cual se comunicara con el micro controlador montado en el auto a través de comunicación wifi.

La problemática más importante que vamos a tener a la hora de desarrollar el proyecto radica en la comunicación vía wifi con una aplicación web, con la que se pretende interactuar con el auto a fin de controlar sus movimientos (velocidad y dirección). Establecer una correcta comunicación y definir un protocolo acorde a los fines establecidos (giros a derecha e izquierda, aceleración y frenado).

# **Objetivo**

**Objetivo Primario:**

El objetivo principal de este proyecto es llevar a cabo el desarrollo funcional de un auto que consta de 3 ruedas, de las cuales una de ellas es la encargada de la tracción y la dirección del mismo. El cual se podrá controlar a través de una página web que estará montada sobre el micro controlador del auto y podrá ser accedida vía wifi.

**Objetivo Secundario:**

El segundo objetivo propuesto es incluir 2 o 3 beacons (Módulos con comunicación bluetooth) para modelar checkpoints con los cuales censaremos el tiempo en el que el auto pase por el área de cobertura y así modelar un circuito de carreras.

# **Dispositivos a utilizar**

[Tabla con el detalle de los dispositivos, módulos y materiales que van a utilizar en el proyecto]

**Importante**: Incluir el presupuesto de todos los materiales a utilizar, como si fueran a ser adquiridos (es decir independientemente de que la cátedra los provea).

Materiales que se utilizaran para llevar a cabo el proyecto son los siguientes:

|  |  |
| --- | --- |
| **Materiales** | **Precio [ $ ]** |
| Kit de desarrollo Arduino | 700 |
| Auto de tres ruedas | 200 |
| Módulo Wifi | 150 |
| Servo | 100 |
| Motor de corriente continua x2 | 100 |
| Modulo Bluetooth | 150 |
| Beacons x 2 | 360 |

Presupuesto estimado total: $ 1.760

# **Identificación de partes**

## E/S del controlador con el exterior, excepto PC

## Comunicaciones con la PC

## Sistema web

1. El micro-controlador se comunicara con el exterior mediante un módulo wifi para su control, y mediante un módulo bluetooth para el testeo de su velocidad.
2. El micro-controlador se comunicara con una pc a través del modulo wifi

.

1. A través de una aplicación web que estará montada en una pc, el usuario podrá asignar dirección y velocidad a la que pretende que el auto se mueva.

El microcontrolador se conectará al servidor web montado en esa pc y actualizará la información del auto y recolectara los comandos a realizar.

Para realizar el servidor web utilizaremos Python con flask y para guardar la información que se transmite de un lado a otro utilizaremos una base de datos en mysql montada también sobre la pc.

**Apéndice B**

**CodigoArduino.ino**

#include <Servo.h> // libreria utilizada para controlar al servo

#define MOTOR\_PIN 6

#define MOTOR\_CONTROL\_PIN 8

#define SERVO\_PIN 10

#define SERVO\_IZQUIERDA 70

#define SERVO\_CENTRO 90

#define SERVO\_DERECHA 110

/\* Variales globales \*/

Servo servo;

int nuevoComando = 0;

int estadoDireccion = 0;

int estadoTraccion = 0;

int velocidad = 0;

void setup() {

Serial.begin(9600);

pinMode(MOTOR\_PIN, OUTPUT);

pinMode(MOTOR\_CONTROL\_PIN, OUTPUT);

digitalWrite(MOTOR\_CONTROL\_PIN, LOW);

analogWrite(MOTOR\_PIN, velocidad);

servo.attach(SERVO\_PIN);

servo.write(SERVO\_CENTRO);

}

void loop() {

// send data only when you receive data:

if (Serial.available() > 0) {

// read the incoming byte:

nuevoComando = Serial.read();

Serial.write(nuevoComando);

switch(nuevoComando){

case 'D':

if(estadoDireccion == -1){

servo.write(SERVO\_CENTRO);

estadoDireccion = 0;

}else{

servo.write(SERVO\_DERECHA);

estadoDireccion = 1;

}

break;

case 'I':

if(estadoDireccion == 1){

servo.write(SERVO\_CENTRO);

estadoDireccion = 0;

}else{

servo.write(SERVO\_IZQUIERDA);

estadoDireccion = -1;

}

break;

case 'R':

if(estadoTraccion == 0){

velocidad = 105;

estadoTraccion = -1;

digitalWrite(MOTOR\_CONTROL\_PIN, HIGH);

}else{

if(estadoTraccion == -1){

if(velocidad>75){

velocidad += -75;

}else{

velocidad = 0;

}

}else{

if(velocidad<150){

velocidad = 0;

estadoTraccion = 0;

digitalWrite(MOTOR\_CONTROL\_PIN, LOW);

}else{

velocidad += -75;

}

}

}

analogWrite(MOTOR\_PIN, velocidad);

break;

case 'A':

if(estadoTraccion == 0){

velocidad = 150;

estadoTraccion = 1;

digitalWrite(MOTOR\_CONTROL\_PIN, LOW);

}else{

if(estadoTraccion == 1){

if(velocidad<175){

velocidad += 75;

}else{

velocidad = 255;

}

}else{

if(velocidad>100){

velocidad = 0;

estadoTraccion = 0;

digitalWrite(MOTOR\_CONTROL\_PIN, LOW);

}else{

velocidad += 75;

}

}

}

analogWrite(MOTOR\_PIN, velocidad);

break;

case 'N':

estadoTraccion == 0;

velocidad = 0;

digitalWrite(MOTOR\_CONTROL\_PIN, LOW); //velocidad maxima en 255

analogWrite(MOTOR\_PIN, velocidad);

break;

default :

break;

}

}

}

**CodigoWIFI.ino**

#include <SoftwareSerial.h>

#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

#include "index.h"

//SSID y contraseña de la red

const char\* ssid = "AutoTallerII";

const char\* password = "12345678";

ESP8266WebServer server(80); //Server en puerto 80

//=======================================================================

// Esta rutina es ejecutada cuando el dispositivo navega en la pagina

//=======================================================================

void handleRoot() {

String s = MAIN\_page; //Lee el contenido HTML

server.send(200, "text/html", s); //Envia la pagina web

String dato = (server.arg("Data"));

Serial.write(dato[0]);

}

//===============================================================

// Configuración

//===============================================================

void setup(void){

Serial.begin(9600);

WiFi.mode(WIFI\_AP); //Configura el modulo wifi como access point

WiFi.softAP(ssid, password);

server.on("/", handleRoot); //Cuando se navega a la raiz de la pagina se ejecuta //se ejecuta la funcion handleRoot()

server.begin(); //Inicia el servidor

}

//===============================================================

// LOOP

//===============================================================

void loop(void){

server.handleClient();

}

**Apéndice C**

**¿?**