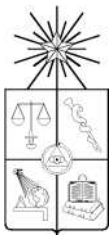


UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

DISEÑO E IMPLEMENTACIÓN DE VEHÍCULO AUTOBALANCEADO SOBRE DOS RUEDAS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA

LEONARDO FELIPE MORENO BUSTAMANTE

PROFESOR GUÍA:
MANUEL DUARTE MERMOUD

MIEMBROS DE LA COMISIÓN:
HÉCTOR AGUSTO ALEGRÍA
CLAUDIO ALARCÓN REYES

SANTIAGO DE CHILE
JUNIO 2009

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO
DE INGENIERO CIVIL ELECTRICISTA
POR: LEONARDO MORENO BUSTAMANTE
FECHA: 15/06/2009
PROF. GUÍA: SR. MANUEL DUARTE MERMOUD

"DISEÑO E IMPLEMENTACIÓN DE VEHÍCULO AUTOBALANCEADO SOBRE DOS RUEDAS"

Uno de los principales problemas que enfrentan muchas ciudades en el mundo son los atochamientos de tráfico y la polución producida por este hecho. Entre las distintas soluciones a este problema se cuenta una alternativa de transporte orientada a tramos cortos, cuyo funcionamiento se basa en mantener un equilibrio inestable con el pasajero parado sobre dos ruedas, del mismo modo en que las personas se mantienen erguidas y caminan, con las ventajas en términos de facilidad de conducción que esto significa.

Este trabajo presenta el desarrollo de un vehículo de esta clase, que corresponde al primero de este tipo de dispositivos en el país. Para ello, se realiza un análisis de las características dinámicas del sistema, de modo de obtener información relevante de las variables que se requiere medir para lograr un funcionamiento exitoso. Se abordan las etapas de diseño e implementación de las distintas partes involucradas, como la electrónica digital, la electrónica de potencia y los aspectos mecánicos del dispositivo. Se pone especial énfasis en la electrónica de control asociada a un vehículo de este tipo, en particular, en el diseño de una unidad de medición de inercia (IMU), encargada de entregar la estimación de la inclinación del vehículo, la cual es la principal variable para lograr la estabilidad deseada.

Para lograr un funcionamiento exitoso del prototipo, se debió primero realizar un ajuste fino de los distintos parámetros involucrados, como tiempos de muestreo y constantes de filtros, entre otros, junto con la implementación de funciones encargadas de detectar y corregir en tiempo real ciertas oscilaciones en la estimación de la unidad IMU observadas en las primeras pruebas. Se obtuvo un prototipo de 40 kg, capaz de alcanzar velocidades del orden de los 6 km/h con un controlador proporcional derivativo. Se le realizaron pruebas de validación con pasajeros inexpertos, los que tras un período de aprendizaje de 30 minutos lograban trasladarse en el vehículo sin mayores problemas, por lo que se concluye que es posible realizar una implementación exitosa de este tipo de vehículos con un costo cercano a una unidad comercial.

A mi familia, amigos, y a tí.

Agradecimientos

Debo partir agradeciendo a mi familia, a quienes están y a quienes ya partieron; no podría haber logrado nada sin el cariño y el apoyo que me han dado siempre. En especial, gracias a mis padres, quienes sin saberlo fueron parte importantísima de este proyecto: cuando las cosas iban mal y ni siquiera yo creía poder sacarlo adelante, ellos no dudaron jamás de mí y me dieron la confianza suficiente para partir de nuevo.

Gracias también a don Manuel Duarte, quien me aceptó con esta loca idea con que llegué un día y tuvo la infinita paciencia para guiarme en medio de las fallas, atrasos e innumerables problemas por los que pasó el proyecto; a Claudio, quien siempre estuvo cuando necesité su ayuda, gran parte de lo que aprendí fue gracias a él. Gracias a Ian, cuando las cosas fallaban siempre estaba ahí dando el dato clave para seguir adelante, o bien apurándome cuando me veía haciendo cualquier otra cosa.

Gracias a todos mis compañeros, amigos, que estuvieron siempre rondando, preguntando como iba *la moto*, *la bicicleta*, *el bicho*, etc. Gracias Diego, Yiyo, Nacho, Kurt, Rubén, Gabriel y tantos otros que se me quedan en el tintero. Mención especial para Gonzalo, el *piloto de pruebas oficial*, daba lo mismo si había que caerse o tirarse por pendientes, ahí estaba siempre apoyándome.

Finalmente, muchas gracias a tí, Colorina. Tu fuiste de las que más sufrió con mi memoria, ya que nunca tenía tiempo, siempre había un problema, pero ni reclamaste, todo lo contrario: me dabas ánimo para seguir; fuiste y seguirás siendo un pilar importantísimo en mi vida. Te adoro.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Una de las áreas de mayor auge dentro de la industria del transporte es aquella que busca solucionar los problemas generados por la gran cantidad de tráfico existente en las grandes urbes, como son la emisión de partículas contaminantes producida por los motores de combustión interna de los vehículos tradicionales y el colapso de las calles de dichas ciudades por congestión vehicular.

Entre las posibles alternativas de solución, la opción que más desarrollo ha tenido en el último tiempo ha sido dotar a los vehículos, ya sea en parte o en su totalidad, de sistemas eléctricos de tracción, impulsado a su vez por el alto e inestable costo del petróleo. Así, ya es posible ver en las ciudades vehículos pequeños (una o dos personas) que funcionan en base a un motor eléctrico, siempre orientados a tramos cortos por su menor autonomía respecto de la tecnología de combustión interna. En Chile, los vehículos eléctricos han penetrado muy fuertemente en el mercado de las motos, ya que da la opción a los consumidores de tener un vehículo pequeño y económico para trasladarse con facilidad en dentro de la ciudad.

Otra de las alternativas existentes para el transporte en tramos cortos es un vehículo basado en el control de un péndulo invertido, donde el usuario está de pie sobre dos ruedas paralelas. El

equilibrio inestable se mantiene por la acción de los motores. Si bien en principio es una opción algo rebuscada, esta clase de dispositivo tiene la particularidad de que ocupa prácticamente el mismo espacio que una persona de pie, además de tener una curva de aprendizaje de manejo muy rápida, ya que todo el equilibrio se realiza del mismo modo en que lo hace una persona al caminar. Así, no requiere un gran conocimiento previo; basta saber caminar para poder transportarse en este tipo de vehículos. Además, una de las grandes ventajas respecto de los vehículos tradicionales es su gran movilidad, ya que es posible obtener un radio de giro nulo girando ambas ruedas en direcciones opuestas, pudiendo así movilizarse fácilmente por rutas complejas que solamente pueden hacerse a pie.

De este modo, los vehículos de autobalanceo pueden absorber parte del tráfico de las ciudades, específicamente el asociado a los tramos cortos, con la ventaja de que ocupan practicamente el espacio de una persona de pie y con una movilidad muy similar. Así, pueden transitar por rutas para peatones sin causar inconvenientes, permitiendo descongestionar las calles y sin emisión de partículas contaminantes. Ejemplos de posibles aplicaciones son las ventas puerta a puerta, servicios de repartición, correos y vigilancia. En este trabajo se buscará implementar una alternativa de vehículo eléctrico a escala humana de este estilo, la cual corresponde al primero desarrollado en el país.

1.1. Antecedentes

Este tipo de vehículo tiene básicamente un único exponente, denominado Segway [2], el cual fue el primer vehículo comercial de esta categoría. A la fecha, existen además un par de trabajos similares entre sí, orientados al control de un vehículo inestable basados en ese primer diseño. A continuación se describirán brevemente estos diseños y sus características.

Figura 1.1: Segway *x2*.

1.1.1. Segway

En el año 2001, el desarrollador estadounidense Dean Kamen presentó un dispositivo de transporte revolucionario para la época, cuyo funcionamiento está basado en el equilibrio natural de las personas: el Segway PT[1].

La aparición de este nuevo medio de transporte basado en una premisa tan estudiada como lo es el control de un péndulo invertido, generó gran interés a nivel mundial ya que se trataba de una solución simple en términos constructivos, ecológica [1] al usar tracción eléctrica, y que eventualmente podía aportar fuertemente a la descongestión de las calles de grandes urbes al tener el potencial de absorber la demanda de transporte en tramos cortos.

Segway Inc. entrega diferentes versiones del dispositivo, existiendo 2 líneas principales de mo-

[1]Segway Personal Transporter

delos: la línea básica, llamada *i2*, y la línea todo terreno, llamada *x2* (Figura 1.1). En la Tabla 1.1 se muestran las características principales del vehículo para ambos modelos [2]:

Modelo	Segway *i2*	Segway *x2*
Peso	47.7 [kg]	54.4 [kg]
Tamaño Ruedas	19”	33”
Tamaño Base	48x63 [cm]	53x84 [cm]
Velocidad Máxima	20 [km/hr]	20 [km/hr]
Autonomía	38 [km]	19 [km]
Motores	Brushless	Brushless

Tabla 1.1: Datos Segway *i2* y *x2*.

Sin embargo, este vehículo no logró el éxito comercial esperado, principalmente por el alto costo de acceder a una unidad (alrededor de US$5.000 en EEUU). Así, dichos vehículos pasaron a aplicaciones específicas, siendo un popular reemplazo de los *caddies* en las canchas de golf, o en aplicaciones de empresas, como lo es el marketing o las rondas de guardias de seguridad. También logró penetrar en un grupo de usuarios de elite cuyo nivel de ingresos les permite acceder a este tipo de tecnologías, existiendo un cierto nivel de compromiso/fanatismo con el vehículo del estilo que generan los productos Apple en sus consumidores.

En el primer semestre del 2009, Segway Inc. presentó un diseño experimental en conjunto con General Motors, denominado proyecto P.U.M.A.[2][3]. Este proyecto, que se observa en la Figura 1.2, busca posicionar definitivamente este tipo de vehículos como una solución real de transporte urbano dotando al diseño original de ciertas características que lo hacen más amigable. En particular, este vehículo se basa en el diseño clásico, pero permite transportar dos pasajeros sentados, haciéndolo más parecido a un auto. El prototipo incluye ruedas extra, las que se utilizan en las detenciones del vehículo, pero en sus desplazamientos se sigue equilibrando dinámicamente sobre las dos ruedas centrales. Este vehículo puede, además, ser una gran oportunidad para los discapacitados, ya que permite desplazamientos naturales para una persona (inclinándose), pero esta vez sin el riesgo de caerse.

[2]Personal Urban Mobility and Accessibility

Figura 1.2: Segway P.U.M.A.

De este modo, a pesar del aparente poco éxito del vehículo, se generó una interesante área de desarrollo de vehículos personales que llamó la atención de muchos desarrolladores, generando una serie de prototipos que emulan el equilibrio humano basados en estructuras análogas al péndulo invertido, ampliando la definición del vehículo como un modelo en particular de una compañía a una categoría de medio de transporte.

1.1.2. Balancing Scooter - Trevor Blackwell

Construido el 2002 por el desarrollador estadounidense Trevor Blackwell y mejorado el año 2005, este vehículo [4] es la principal referencia al momento de buscar alternativas a la versión comercial Segway [2], sin ser este mismo un desarrollo con fines de lucro. El funcionamiento es análogo al Segway, pero con menores medidas de seguridad en cuanto a redundancia en el estado del vehículo (no utiliza más de un sensor por variable).

La primera versión realizada (Figura 1.3), si bien lograba un funcionamiento óptimo, no lograba igualar al Segway [2] (única referencia entonces), por lo que la versión 2 agregó una serie

Figura 1.3: Trevor Blackwell.

de mejoras, en particular agrandando el tamaño de las ruedas, logrando así velocidades de casi 25 km/h. En la Tabla 1.2 se pueden ver las características de ambas versiones realizadas:

Modelo	Versión 1	Versión 2
Peso	40.8 [kg]	31.75 [kg]
Tamaño Ruedas	14”	20”
Velocidad Máxima	14.5 [km/hr]	24 [km/hr]
Motores	CC con imánes	CC con imánes

Tabla 1.2: Características de las versiones de Trevor Blackwell.

1.1.3. JOE: A Mobile, Inverted Pendulum

Este desarrollo [5] consiste en una versión a escala radiocontrolada de un vehículo autobalanceado. En el trabajo [5] se presenta el desarrollo del modelo matemático del sistema y una descripción del sistema de control implementado. El vehículo mide 65 cm, pesa 12 kgs y es capaz de inclinarse hasta 30º sin caer. Su velocidad máxima es de 1.5 m/s.

1.1.4. The DIY Segway

Este trabajo [6] fue desarrollado por estudiantes de secundaria de EEUU asesorados por estudiantes del MIT [3] (ver Figura 1.4). Es un poco más pequeño que los otros desarrollos [2, 4, 7], su estructura es una de las más simples existentes, basada íntegramente en aluminio para reducir el peso final.

Utiliza como unidad de procesamiento una placa de desarrollo basada en un microprocesador PIC16F877. Lo interesante es que utiliza dos motores relativamente pequeños (50 W aproximadamente, peak de 337 W) siendo totalmente funcional, pero con restricciones de velocidad, masa e inclinación mayores a otras implementaciones.

1.1.5. The Two Wheel Deal

Esta versión [7] ha sido finalizada recientemente por 4 estudiantes de la Universidad de Purdue (ver Figura 1.5). En el trabajo [7] se muestra una implementación desde cero, desarrollando toda la electrónica necesaria para el proyecto. Se realiza además un estudio detallado de factibilidad de patentar el vehículo, intentando diferenciarlo de la competencia directa, el Segway [2].

[3] Massachusetts Institute of Technology

Figura 1.4: The DIY Segway.

1.1.6. Toyota Winglet PT

Esta versión [8] será la segunda versión comercial de este tipo de vehículos cuando salga al mercado el año 2010 (ver Figura 1.6). Desarrollado por Toyota, contará con 3 modelos de diferentes tamaños, donde destacan las versiones más pequeñas. Éstas se manejan únicamente con la inclinación de los pies. Sin embargo, no apuntan al mismo mercado que el Segway [2], ya que las velocidades que alcanza son muy bajas (del orden de los 6 km/h), lo que lo hace más adecuado para transportarse dentro de edificios.

Figura 1.5: The Two Wheel Deal.

1.2. Objetivos

Como se expuso, los vehículos de autobalanceo son una importante y creciente alternativa a diversos problemas asociados al tráfico en grandes ciudades, lo que hace interesante el desarrollo de este tipo de transporte. Así, este trabajo busca construir un prototipo de un vehículo autobalanceado desarrollando toda la electrónica necesaria. Además, este trabajo es el primero en el país acerca del tema, lo que permite abrir el camino a nuevos prototipos, los cuales aprovecharían la experiencia ganada en pos de mejores resultados, analizar distintas aplicaciones y/o variaciones al modelo, y poder aportar al desarrollo mundial de la industria automotriz.

Figura 1.6: Toyota Winglet TP.

1.2.1. Objetivos Generales

- Diseño y construcción de un vehículo de autobalanceo en tamaño real, capaz de transportar una persona.
- Lograr que el vehículo tenga curva de aprendizaje de manejo rápida

1.2.2. Objetivos Específicos

Dadas los objetivos generales del proyecto y las características del vehículo en cuestión, se definen los siguientes objetivos específicos:

- Implementación de un sistema de control electrónico y de sensores apropiados para el vehículo

- Diseño e implementación de la etapa de potencia para motores CC.
- Lograr un diseño mecánico y construcción robusta.

1.3. Estructura de la Memoria

Este documento está compuesto por 5 capítulos, los cuales se describen a continuación.

El Capítulo 1 presenta una introducción a los vehículos autobalanceados, mostrando de donde nacen y la necesidad que buscan cubrir. Se muestran las distintas implementaciones realizadas a la fecha de este tipo de vehículos, algunas comerciales y otras por mero afán investigativo. Asimismo, se presentan tanto los objetivos generales como específicos de este trabajo.

El Capítulo 2 entrega el modelo matemático simplificado del sistema, de forma de poder entender la dinámica del vehículo y buscar el modo de controlarlo. A partir de este modelo se obtienen los requerimientos técnicos que deberán ser cubiertos en el diseño del vehículo, en particular, las variables que se deben medir para obtener el estado del sistema.

En el Capítulo 3 se presenta en detalle el diseño de cada una de las etapas del proyecto, así como su implementación física. Este capítulo es el más técnico y amplio, ya que presenta información relativa al desarrollo de la electrónica necesaria, a las etapas de potencia y a los aspectos de construcción mecánica, entre otros. También se analizarán las distintas dificultades observadas en el desarrollo del proyecto y cómo se solucionaron.

El Capítulo 4 presenta un análisis de los resultados obtenidos de la implementación y puesta en marcha del vehículo, en base a las distintas pruebas realizadas. Además, se realiza un análisis de costos del proyecto completo.

Finalmente, el Capítulo 5 se analiza el trabajo realizado desde el punto de vista de los objetivos planteados inicialmente, obteniéndose las conclusiones pertinentes. Además, se postulan posibles mejoras y/o adiciones que podrían realizarse en futuros diseños.

Capítulo 2

Modelación y Requerimientos de Diseño

2.1. Modelo Matemático

Para la implementación del vehículo en cuestión se requiere estimar ciertas variables físicas del sistema de forma de poder definir parámetros constructivos y de diseño, como posibles materiales, energía necesaria, tipos de sensores, etcétera.

Como se aprecia en el modelo simplificado de las Figuras 2.1 y 2.2, el vehículo es básicamente un péndulo invertido, con la diferencia de que el agente que induce el movimiento (el motor) tiene una acción directa tanto en la ecuación para el movimiento lineal como para el movimiento angular. En el caso del péndulo invertido clásico, la acción sobre el movimiento angular es sólo consecuencia de la dinámica de los cuerpos.

Las variables y parámetros que se considerarán se detallan en la Tabla 2.1.

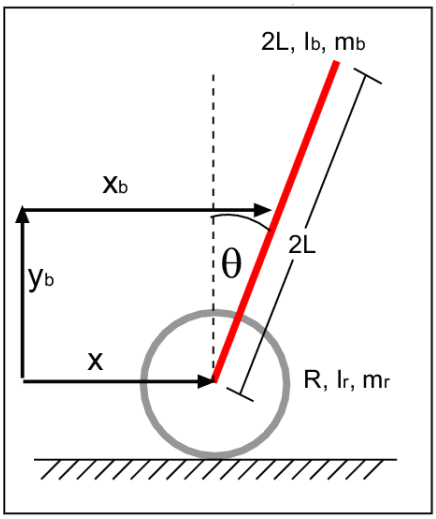


Figura 2.1: Modelo simplificado del sistema - Vista lateral.

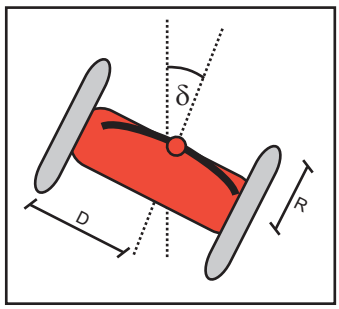


Figura 2.2: Modelo simplificado del sistema - Vista superior.

Tras analizar la física del sistema[1], el modelo matemático que representa el conportamiento dinámico del sistema está representado por las Ecuaciones 2.1.

[1]El detalle del desarrollo de ecuaciones se encuentra en el Apéndice A

Nombre	Tipo	Significado
θ	Variable	Inclinación del vehículo respecto de la gravedad
$\dot{\theta}$	Variable	Velocidad de giro respecto del eje
x	Variable	Posición del centro de las ruedas respecto de la horizontal
$\dot{x}$	Variable	Velocidad del centro de las ruedas respecto de la horizontal
x_b	Variable	Posición del centro de masas del sistema respecto de la horizontal
$\dot{x_b}$	Variable	Velocidad del centro de masas del sistema respecto de la horizontal
y_b	Variable	Posición del centro de masas del sistema respecto de la vertical
$\dot{y_b}$	Variable	Velocidad del centro de masas del sistema respecto de la vertical
$\dot{\delta}$	Variable	Velocidad angular de giro sobre el plano
L	Parámetro	Distancia del eje de las ruedas al CM
m_b	Parámetro	Masa de la barra (estructura y pasajero)
$I_{b\theta}$	Parámetro	Momento de inercia de la barra respecto del CM en inclinaciones
$I_{b\delta}$	Parámetro	Momento de inercia de la barra respecto del CM en giros
D	Parámetro	Distancia entre centro del vehículo y centro de una rueda
R	Parámetro	Radio de la rueda
m_r	Parámetro	Masa de la rueda
I_r	Parámetro	Momento de inercia de la rueda respecto del eje de giro
τ_{izq}, τ_{der}	Acción	Torques aplicados a la rueda izquierda y derecha

Tabla 2.1: Variables y parámetros del modelo simplificado.

$$\begin{aligned} I_{b\theta} \cdot \ddot{\theta} - L \cdot m_b \cdot \ddot{x} \cdot cos(\theta) + m_b \cdot L^2 \cdot \ddot{\theta} &= -\tau_{izq} - \tau_{der} \\ \left(m_b + 2m_r + 2\frac{I_r}{R^2}\right) \cdot \ddot{x} - m_b \cdot L \cdot \dot{\theta}^2 \cdot sin(\theta) + m_b \cdot L \cdot \ddot{\theta} \cdot cos(\theta) &= \frac{\tau_{izq} + \tau_{der}}{R} \\ I_{b\delta} \cdot \ddot{\delta} &= \frac{\tau_{izq} - \tau_{der}}{R} \cdot D \end{aligned} \tag{2.1}$$

De la observación de las ecuaciones 2.1, es fácil notar que existe una dependencia entre el movimiento lineal (representado por x) y la inclinación (representada por θ). Únicamente el giro del vehículo no tiene relación con el resto de las variables del sistema, siendo dependiente solo de la acción de los motores.

Al linealizar el sistema 2.1 en torno a $\theta = 0^o$, las ecuaciones quedan de la siguiente forma:

$$\begin{aligned} \ddot{\theta} &= a \cdot \theta + b \cdot (\tau_{izq} + \tau_{der}) \\ \ddot{x} &= c \cdot \theta + d \cdot (\tau_{izq} + \tau_{der}) \\ \ddot{\delta} &= e \cdot (\tau_{izq} - \tau_{der}) \end{aligned} \tag{2.2}$$

donde a, b, c, d y e son constantes que dependen de los parámetros del vehículo[2]. Visto de modo matricial, el sistema es el siguiente:

$$\frac{d}{dt}\begin{pmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \\ \delta \\ \dot{\delta} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & a & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \\ \delta \\ \dot{\delta} \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ d & d \\ 0 & 0 \\ b & b \\ 0 & 0 \\ e & e \end{pmatrix} \cdot \begin{pmatrix} \tau_{izq} \\ \tau_{der} \end{pmatrix} \tag{2.3}$$

Como se puede apreciar de las ecuaciones 2.2 y 2.3, la variable asociada a la inclinación (θ) dejó de depender de las otras variables del sistema (x, δ). Es decir, es posible diseñar un controlador simple para el ángulo θ y separadamente para el giro δ. No obstante, la dinámica del movimiento lineal x sí dependerá de la inclinación. De este modo, se puede considerar el problema (en torno al punto de operación) como dos problemas menores e independientes, los que controlan la inclinación y el giro, mientras que el movimiento lineal queda dependiente de la dinámica de dichos sistemas.

[2]Detalle de dichas constantes en el Apéndice A

2.2. Requerimientos

Con los objetivos, tanto generales como específicos del trabajo, más lo obtenido en la modelación del sistema, se definen ciertas funcionalidades específicas a implementar en cada una de las etapas que componen el vehículo, lo que se detalla a continuación:

2.2.1. Variables a medir

Debido a la característica MIMO[3] que presenta el sistema y el modo de funcionamiento del mismo, se requiere la medición o estimación en cada instante de las siguientes variables:

Inclinación / Velocidad de inclinación (θ, $\dot{\theta}$). Este parámetro es la principal variable a ser controlada, ya que de ella depende directamente el equilibrio del sistema.

Velocidad lineal ($\dot{x}$). Es la velocidad de traslación del vehículo, combinación de la velocidad de ambas ruedas.

Referencia de giro ($\dot{\delta}$). Se requiere para establecer la dirección y velocidad de rotación deseada del vehículo. Básicamente representará una diferencia de velocidad entre cada una de las ruedas.

Estado baterías. Para asegurar el correcto funcionamiento de todos los sistemas involucrados, se debe monitorear constantemente el estado de las baterías, principalmente para dar aviso en caso de una baja carga de éstas (y con ello, de la pérdida del control).

Corriente en motores. De modo de proteger la etapa de potencia del sistema, se debe monitorear la corriente circulante por ésta. Así, se le puede limitar en caso de sobrepasar límites de seguridad previamente establecidos.

[3]Multiple Input, Multiple Output

2.2.2. Restricciones

Como que el sistema a implementar debe ser totalmente funcional, se definen a continuación ciertas restricciones para las diferentes etapas involucradas.

Robustez mecánica. La estructura debe ser suficientemente robusta para soportar caídas sin sufrir mayores daños y protegiendo la electrónica a bordo. Además, no debe poseer elementos innecesarios que compliquen su construcción y/o posibles reparaciones.

Plataforma de control robusta. Las tarjetas electrónicas de control y sensores estarán expuestas a vibraciones mecánicas y a un medio ruidoso en términos electromagnéticos (debido principalmente a la presencia de motores eléctricos), por lo que el diseño debe considerar dichas variables.

2.3. Solución Planteada

Dados los requerimientos impuestos, el trabajo a realizar se divide en 4 etapas principales:

- Electrónica de Control y Sensores
- Electrónica de Potencia y Actuación
- Estrategia de Control
- Diseño Mecánico

A continuación, se describen las directrices de diseño que guían el desarrollo de cada una de estas etapas.

2.3.1. Electrónica de Control y Sensores

Esta etapa será la encargada de procesar toda la información relativa al vehículo, ya sea extrayendo información de los sensores, estimando las distintas variables, ejecutando las acciones de control o estableciendo una comunicación con el usuario.

El desarrollo de esta etapa se realizará tomando como experiencia previa el Proyecto de Instrumentación de Vehículos a Escala (PIVE) [9, 10, 11]. Este proyecto consiste en un sistema de instrumentación electrónico distribuido y flexible, cuyo funcionamiento separa en distintas placas la implementación del control y los distintos sensores existentes.

Así, se deberá diseñar una tarjeta electrónica de control central que sea capaz de sensar distintas variables, que posea diversos canales de comunicación, que controle los motores y que tenga capacidad de procesamiento suficiente para ejecutar cálculos del sistema de control implementado. Se optó por utilizar como unidad central de proceso un DSP[4] de la línea 280x, de la familia C2000 de Texas Instruments, la cual está orientada a control en tiempo real de sistemas de potencia [13]. Las principales características de esta línea de DSP se muestran en la Tabla 2.2.

Línea	280x Fixed-point Series
Clock	60/100 Mhz
Voltaje de Alimentación	3.3 V (I/O) y 1.8 V (Core)
Módulos de Comunicación Serial	SPI SCI CAN I2C
Módulos de Control	PWM
Conversores	ADC de 12 bits(hasta 80 ns)
Otros	Soporte para JTAG [20]

Tabla 2.2: Principales características línea F280x.

Respecto de los sensores, variables como el estado de las baterías o referencias de giro se miden fácilmente mediante señales análogas en los conversores ADC. La medición de la velocidad se debe

[4]Digital Signal Processor

realizar mediante un encoder adosado al eje de giro del motor (ver Figura 2.3) que entregue dos trenes de pulsos en cuadratura (QEPA y QEPB), de modo de poder medir directamente velocidad y sentido de giro.

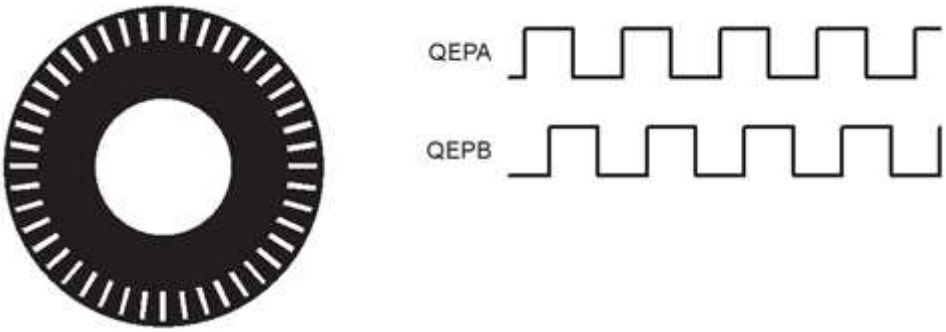


Figura 2.3: Disco de encoder y señal en cuadratura.

La inclinación requiere un tratamiento especial, ya que hay que considerar una serie de factores relativos a la naturaleza de los sensores asociados a dicha medición, lo que se analiza a continuación.

Estimación de la inclinación

Esta medición tiene una gran complejidad, ya que requiere el uso de 2 tipos de sensores: acelerómetros y giróscopos. Un acelerómetro, como dice su nombre, es un sensor capaz de detectar y cuantizar aceleraciones externas, lo cual se hace generalmente referenciando el valor respecto de la gravedad. Así, un acelerómetro en reposo en la superficie terrestre indica una aceleración de $-1 \cdot g$, mientras que aislado de cualquier fuente gravitacional, y manteniendo el estado de reposo, indicaría una medición nula. Este tipo de sensores se utiliza para una estimación directa de la inclinación tomando como hipótesis que la única aceleración existente es la gravedad, la cual se busca descomponer en los ejes de medición según el ángulo de inclinación, como se esquematiza en la Figura 2.4.

Usando el eje x del acelerómetro, cuya medición fue de gx, se puede deducir la inclinación θ

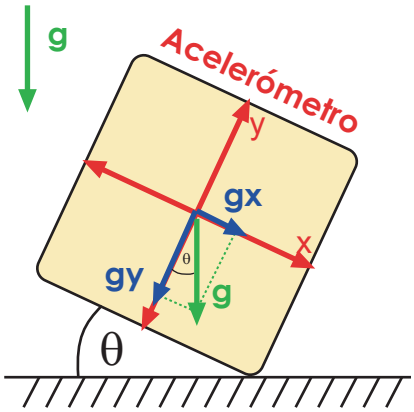


Figura 2.4: Mediciones de acelerómetro inclinado.

de la igualdad $g \cdot sin(\theta) = gx$, o bien usando el eje *y*, donde la igualdad sería $g \cdot cos(\theta) = gy$. Es importante notar que dependiendo del cuadrante en que se encuentre el ángulo se debe usar una u otra ecuación, ya que solo una permitirá establecer el signo de θ (en el ejemplo visto, la igualdad para el eje x permite determinar dicho signo). Cuando un acelerómetro se utiliza de este modo suele llamarse *inclinómetro*.

Esta estimación será tan exacta como lo sea el sensor respectivo **siempre que la única aceleración existente sea la gravedad**. De lo contrario, de existir aceleraciones anexas éstas aparecerán en las mediciones del sensor (ya que se trata de un sistema no inercial), dando así una solución distinta a la esperada de las igualdades ya vistas. El sistema en cuestión está expuesto a aceleraciones distintas a la gravedad, por su condición de vehículo en movimiento.

Otro modo de estimar la inclinación es mediante un giróscopo electrónico. Este sensor entrega una señal directa de la *velocidad de inclinación* ($\dot{\theta}$), es decir, de la derivada de la variable que se busca estimar. Tiene la gran ventaja respecto del acelerómetro que las aceleraciones externas no le afectan, por lo que suele usarse para estimar la inclinación mediante el uso de una condición inicial conocida e integración durante un período. Es decir, la inclinación se estima según $\theta_t = \theta_{t-1} + \dot{\theta} \cdot dt$, donde θ_{t-1} es la estimación anterior (o condición inicial) y *dt* corresponde al tiempo de muestreo utilizado. El problema más claro de esta solución es el error acumulado asociado a la integración numérica, que provoca que luego de transcurrido cierto tiempo de cómputo la estimación pierda credibilidad. En este tipo de aplicaciones este error es conocido como *drift*, ya que el valor asociado

a inclinación nula se desplazará algunos grados conforme pasa el tiempo de procesamiento.

Luego, ambos sensores presentan ventajas y desventajas. Sin embargo, estas son complementarias. Es decir, donde tiene problemas un tipo sensor, el otro no los tiene. Así, pueden usarse ambos sensores al unísono, donde el acelerómetro tiene una mejor estimación de largo plazo (las aceleraciones son de corta duración y no tiene error acumulado), mientras el giróscopo posee un mejor comportmiento en el corto plazo (no es afectado por las aceleraciones transientes). Así, una posible estimación para la inclinación que use ambos sensores está dada por la ecuación 2.4

$$\theta_t = (\theta_{t-1} + \dot{\theta} \cdot dt) \cdot \alpha + \theta_{acc} \cdot (1 - \alpha) \tag{2.4}$$

donde $\alpha \in [0;1]$ determina el porcentaje de creencia que se le asigna a cada estimación (ya sea con giróscopo o con acelerómetro), y su valor se debe determinar según el comportamiento observado, θ_{acc} y $\dot{\theta}$ representan las mediciones del acelerómetro y el giróscopo, respectivamente. Este tipo de estimación mediante acelerómetros y giróscopos es la base de una IMU[5], principal componente en los sistemas que buscan estimar su orientación e inclinación. Este tipo de unidades son utilizadas en una amplia variedad de aplicaciones, como lo son aviones, helicópteros, satélites, barcos e incluso misiles, entre otras posibles aplicaciones. Las complejidad de la unidad IMU está dada principalmente por la cantidad de giróscopos y acelerómetros que esta posea, los cuales dependiendo de su orientacion pueden determinar orientación en distintos ejes. Así, la placa de sensores que se debe diseñar tendrá (entre otros tipos de sensores) una unidad IMU básica, consistente en un acelerómetro y un giróscopo, permitiendo determinar orientación respecto de un solo eje (en este caso, respecto del eje de los motores).

[5]Inertial Measurement Unit

2.3.2. Etapa de Potencia y Motores

Esta etapa considera el análisis y la elección de los motores y el desarrollo de la electrónica de potencia capaz de controlarlos. Por facilidad en la implementación de esta etapa, se opta por utilizar motores de corriente continua. Este tipo de motor, sin embargo, está en desuso debido a las ventajas que presentan los motores sincrónicos, por lo que se debe importarlos. Aprovechando este hecho, se escoge un modelo de motor que ya ha sido utilizado en aplicaciones de este tipo [4] [7]. El motor seleccionado es el modelo NPC-T64 [17], de la empresa NPC-Robotics, en EEUU, cuyas principales características se muestran en la Tabla 2.3.

Modelo	NPC-T64
Voltaje	24 V
Caja Reductora	20:1
Velocidad nominal	230 RPM
Potencia	0.7 HP
Corriente nominal	21 A
Corriente máxima	110 A (trabado)
Torque máximo	93 Nm

Tabla 2.3: Datos motor NPC-T64.

Le elección del motor permite determinar parámetros generales de diseño en términos estructurales y de electrónica de potencia. El control de un motor de este tipo se realizará mediante un circuito tipo *puente H* [19] basado en MOSFETS que soporten los voltajes y corrientes necesarios. La alimentación será de 48 V, en base a 4 baterías de 12 V. El control del voltaje entregado al motor será mediante señales PWM[6] que excitan el semiconductor, de modo de regular la potencia traspasada al motor según el ciclo de trabajo asignado a dichas señales.

[6]Pulse Width Modulation

2.3.3. Diseño Mecánico

Los requerimientos asociados a esta etapa básicamente buscan proteger la electrónica del vehículo mediante la robustez de la estructura. Dicha estructura será principalmente una caja en base a un esqueleto de acero que cumpla dicha función. En esta caja irá almacenada, además de la electrónica, los motores y las baterías. Esta estructura deberá estar contenida completamente dentro del radio de las ruedas, de modo de asegurar que ante cualquier caída esta no será golpeada, sino que el impacto será absorbido por la parte superior del vehículo, lejana a la electrónica de control.

2.3.4. Estrategia de Control

En este trabajo no se implementará un controlador muy complejo, sino que se pondrá el énfasis en la instrumentación, por lo que el controlador deberá ser sencillo. Basado en la experiencia de otros trabajos [4, 7, 5] y en las características propias del sistema observadas durante su modelamiento, se implementará un controlador proporcional-derivativo (PD) para el subsistema asociado únicamente a la inclinación, un controlador proporcional asociado al giro y el movimiento lineal se dejará en lazo abierto y dependerá de las acciones de control de los dos subsistemas anteriores.

En detalle, para el diseño del controlador PD es necesario notar (como se dijo anteriormente) que en las ecuaciones linealizadas (ver ecuaciones 2.2) el ángulo de inclinación θ es independiente de las otras variables del sistema, por lo que se puede diseñar un controlador sobre esta única variable y analizar el comportamiento en lazo abierto de la velocidad lineal $\dot{x}$ (que sí depende de la inclinación θ). La velocidad de giro $\dot{\delta}$ se controla con un lazo independiente.

Sin embargo, no es posible separar el sistema relativo a θ y a δ directamente, ya que las acciones de control τ_{izq} y τ_{der} dependen de cada motor (ver ecuaciones 2.2). Para solucionar ésto, en vez de trabajar directamente con τ_{izq} y τ_{der}, se utilizarán *torques virtuales* τ_θ y τ_δ asociados a cada subsistema y sin injerencia sobre el otro (es decir, τ_θ únicamente actúa sobre la inclinación, por

ejemplo). Las acciones de control reales y virtuales se relacionan según la ecuación 2.5.

$$\begin{aligned} \tau_\theta &= \tau_{izq} + \tau_{der} \\ \tau_\delta &= \tau_{izq} - \tau_{der} \end{aligned} \tag{2.5}$$

Visto de modo matricial, dicha relación se muestra en la ecuación 2.6. La matriz M se denomina *matriz de desacople* [5], ya que permite desacoplar definitivamente los distintos subsistemas involucrados. Además, su inversa permite calcular las acciones de control de cada motor una vez calculadas las acciones de control virtuales.

$$\begin{pmatrix} \tau_\theta \\ \tau_\delta \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}}_{M} \cdot \begin{pmatrix} \tau_{izq} \\ \tau_{der} \end{pmatrix} \tag{2.6}$$

De este modo, el sistema a controlar queda representado en la ecuación 2.7.

$$\begin{aligned} \ddot{\theta} &= a \cdot \theta + b \cdot \tau_\theta \\ \ddot{x} &= c \cdot \theta + d \cdot \tau_\theta \\ \ddot{\delta} &= e \cdot \tau_\delta \end{aligned} \tag{2.7}$$

El controlador planteado tiene la particularidad de que el pasajero no controla el sistema mediante la referencia, la cual siempre es 0^o (es decir, el sistema siempre busca estar vertical), sino

que lo hace mediante una perturbación ocasionada al inclinarse en uno u otro sentido, tal como se observa en la Figura 2.5.

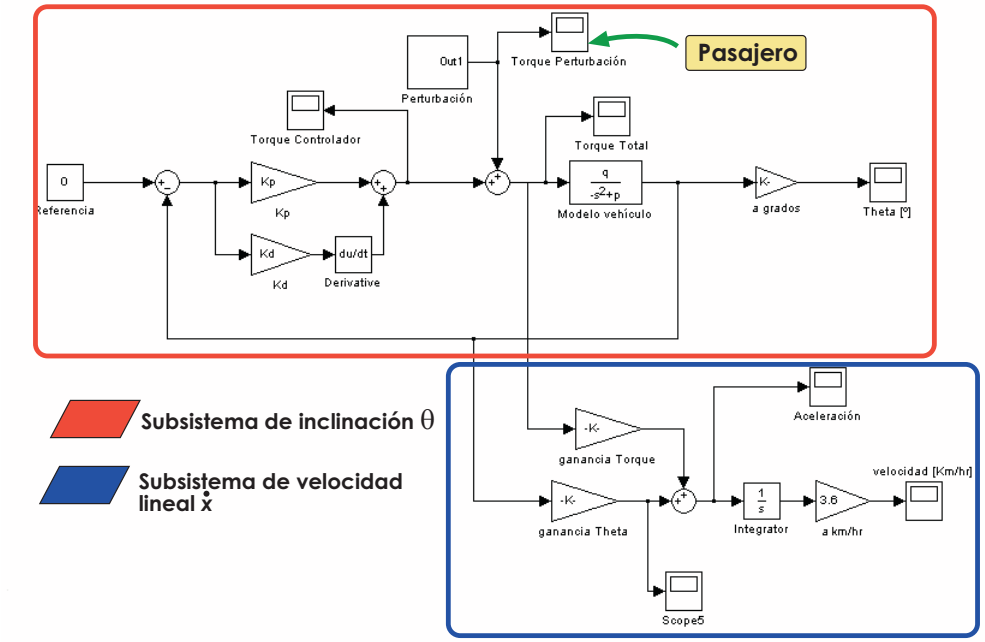


Figura 2.5: Subsistemas de inclinación θ y velocidad $\dot{x}$.

El control de la inclinación es el más importante en este sistema, ya que se trata de mantener el vehículo en un equilibrio inestable. La ecuación para el giro es suficientemente simple (de primer orden para la velocidad $\dot{\delta}$) como para implementar un controlador proporcional (ver Figura 2.6). La velocidad lineal queda en lazo abierto y se determina como consecuencia del control de inclinación, es decir, si el ángulo varía respecto de 0^o, el controlador de inclinación necesariamente debe mover el vehículo en dicha dirección, aumentando la velocidad.

Asignando valores aproximados a los parámetros del modelo se realizaron algunas simulaciones implementando el controlador descrito bajo diferentes condiciones, las cuales demostraron que es factible controlar el sistema. Además, debe considerarse que el conductor también ejerce cierto control sobre el equilibrio (no es peso muerto), lo cual no está incorporado en dichas simulaciones, por lo que es aún más factible el funcionamiento exitoso del vehículo.

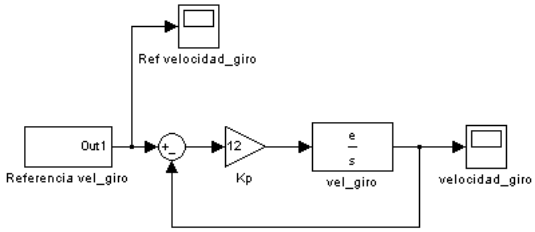


Figura 2.6: Subsistema de giro $\dot{\delta}$.

Para obtener los valores de las constantes del controlador (K_p y K_d) se tomaron requerimientos en el dominio del tiempo. En particular, se busca una respuesta al escalón unitario con una sobreoscilación menor al 10% (buscando no alejarse del rango de linealización) y un tiempo de estabilización t_s (tiempo en entrar a la banda del $\pm 5\%$ del valor deseado) variable entre 0.5 y 3 segundos. Se realizaron algunas pruebas del controlador descrito con distintos tiempos de estabilización, lo que se refleja en la Tabla 2.4.

$t_s[s]$	K_p	K_d	$\theta_{max}[^o]$	τ_{max}	$\dot{x}_{max}[\frac{km}{h}]$
0.5	-4278.0	-436.9	0.4	30	0.8
1.0	-1278.0	-218.4	1.5	35	3.0
1.5	-722.3	-145.6	3.5	45	7.0
2.0	**-527.9**	**-109.2**	**6.4**	**60**	**12.0**
2.5	-4379.0	-87.4	10.0	75	20.0
3.0	-388.9	-72.8	15.0	90	28.0

Tabla 2.4: Características para distintos tiempos de estabilización.

De la observación de la Tabla 2.4 es fácil notar que existe una relación inversa entre el torque máximo para controlar el sistema y la velocidad lograda. Así, se eligen las constantes de modo tal de obtener 12 km/hr, pero en un rango de torque aceptable para los motores (ver Tabla 2.3).

Las curvas obtenidas mediante simulaciones para dichas constantes elegidas se muestran en la Figura 2.7.

De esta simulación se puede observar que el controlador logra mantener estable el sistema.

En la Figura 2.7a se aprecia que al acelerar el vehículo se inclina hasta los 6º aproximadamente, donde el torque virtual τ_θ se grafica en la Figura 2.7b, mostrando la misma tendencia de la curva anterior (inclinación). En las figuras 2.7c y 2.7d se observa el resultado del sistema asociado al giro del vehículo y el respectivo torque virtual τ_δ. Finalmente en las Figuras 2.7e y 2.7f se muestra la velocidad $\dot{x}$ que alcanza el vehículo (cerca de 12 km/h) y los torques efectivos sobre los motores. En este caso, cada motor debe entregar cerca de 30 [Nm], con algunos peaks de 40 [Nm] asociados a los pulsos que indican el giro del vehículo (referencia que en la práctica será suavizada). Este valor se encuentra dentro del rango estimado según las curvas características entregadas por el fabricante del motor elegido, por lo que el sistema tiene grandes posibilidades de ser controlado eficazmente.

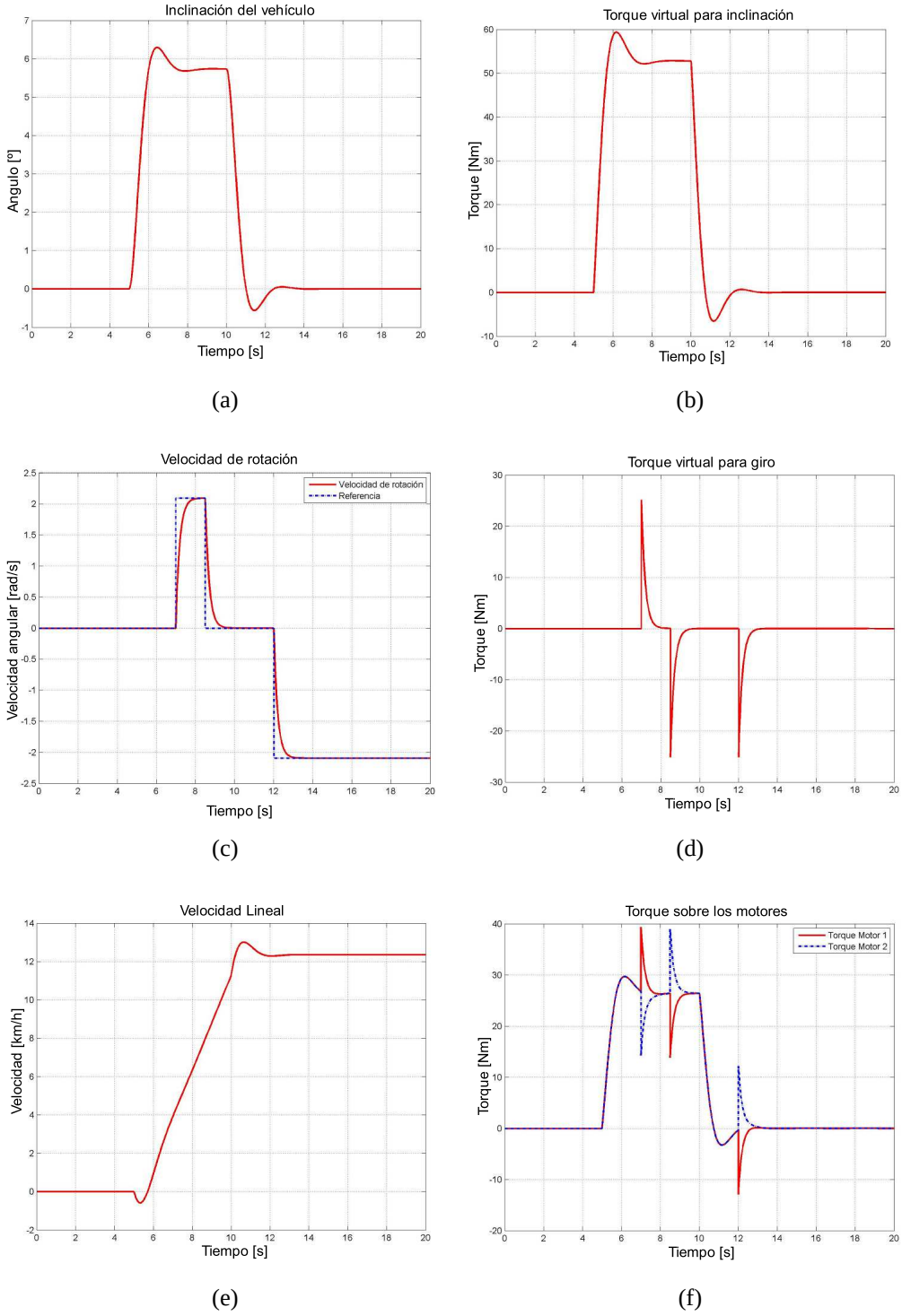


Figura 2.7: Curvas obtenidas en simulaciones para las variables involucradas.

Capítulo 3

Diseño e Implementación

En este capítulo se presenta el detalle técnico de la implementación realizada para cada una de las etapas del vehículo descritas en el capítulo anterior. Se analizan también las dificultades que se presentaron en la implementación y como se solucionaron. Para el diseño de las distintas tarjetas electrónicas se utilizó el software de diseño Orcad 15.7. Para la programación de los microcontroladores involucrados se utilizaron los software Code Composer Studio v3.3 (para el DSP) y PIC-C (para microcontroladores PIC). Finalmente, para el diseño mecánico de piezas se utilizó el software Solid Edge v19.

3.1. Módulo de Control y Sensores

Esta tarjeta, como se mencionó en el capítulo anterior, está basada en un DSP de la línea F280x, los que están orientados a control. Considerando las distintas necesidades de periféricos que requiere la aplicación, se elige el modelo TMS320F2808 [12], cuyas características principales se muestran en la Tabla 3.1. Este dispositivo tiene suficiente capacidad para comunicarse con distintos tipos

de sensores y generar las señales PWM[1] necesarias para el control de los motores sin entorpecer el cálculo de las acciones de control requeridas. Así, se aprovecharon al máximo los recursos de este microprocesador, bajo la restricción de no aumentar excesivamente la complejidad del diseño de la tarjeta electrónica.

Modelo	TMS320F2808
Clock	100 Mhz
Periféricos	16 salidas PWM 4 módulos Capture 2 módulos eQEP 16 ADC, 12 bits (160 ns) 4 buses SPI 2 puertos SCI 4 buses CAN 1 bus I2C

Tabla 3.1: Principales características DSP TMS320F2808.

La Figura 3.1 muestra esquemáticamente los distintos módulos que se habilitaron en la placa.

En resumen, los principales bloques de esta plataforma son:

- DSP
- IMU
- Módulo de Comunicaciones
- Conversores Análogo-Digital (ADC)
- Módulo de PWM
- Encoders
- Alimentación

[1] Pulse Width Modulation

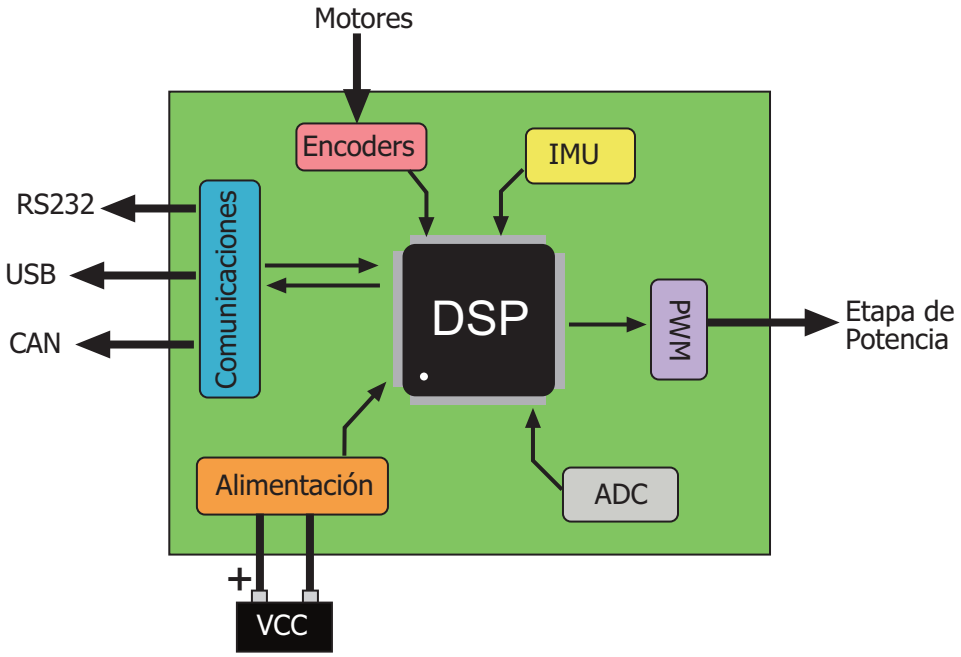


Figura 3.1: Diagrama de bloques de la placa de control.

3.1.1. DSP TMS320F2808

Como se mencionó anteriormente, el DSP elegido pertenece a la familia C2000 de la empresa Texas Instruments, específicamente el modelo TMS320F2808 [12]. Este dispositivo (uno de los de mayores prestaciones en dicha línea) tiene variados módulos integrados que le permiten establecer distintos tipos de comunicaciones con otros dispositivos, tales como conversores análogo-digital y módulos PWM, entre otros. Esto le permite realizar una gran cantidad de acciones sin sobrecargar la CPU, remitiéndose ésta a realizar distintos tipos de cálculos y a reconfigurar de vez en cuando alguno de dichos módulos, de ser necesario.

De este modo, un DSP de esta línea es, como su nombre lo dice, un *procesador digital de señales* con características de microcontrolador, lo que le permite realizar cálculos de alta complejidad manteniendo una comunicación directa con el exterior.

Este dispositivo es parte de la familia de DSP, con arquitectura de tipo Harvard [18], con punto

fijo. Esto se debe considerar al momento de trabajar con este microcontrolador, ya que se puede perder precisión en algunos cálculos de variables cuya parte decimal sea indispensable si no se toman precauciones al momento de programar.

Programación

Este dispositivo permite reprogramarse por diversos medios. Sin embargo, en este caso solamente se utilizarán dos de ellos. El primero y más simple es mediante el puerto serie (SCI-A), para lo cual basta configurar un pin del DSP mediante un jumper, de modo de seleccionar si este buscará iniciar el programa desde la memoria FLASH del dispositivo (funcionamiento normal) o bien quedará a la espera de recibir datos mediante el puerto serie, para luego proceder a la reprogramación (borrado, programación y verificación).

La segunda opción, más compleja pero mucho más versátil que la anterior, es mediante el uso de la interfaz JTAG[2] [20] del DSP. Esta interfaz no sirve únicamente para reprogramar, sino que permite depurar el código programado directamente en el DSP, permitiendo encontrar más rápidamente posibles errores, los cuales son muy difíciles de detectar sin una herramienta de este tipo. Para poder utilizar esta interfaz se requiere un dispositivo JTAG que permita conectar el DSP con un computador con el software necesario para realizar dichas tareas. En este caso, se dispone de un JTAG modelo XDS560USB de la compañía Spectrum Digital y de una versión de prueba del software Code Composer Studio v3.3, de Texas Instruments, software en el cual se realiza la edición y compilación del código.

[2] Joint Test Action Group

3.1.2. IMU

La unidad IMU (Inertial Measurement Unit) es el sensor de mayor importancia en el diseño del sistema, ya que es la encargada de obtener la información relativa a la inclinación del vehículo. Como se discutió en la Sección 2.3.1, debe implementarse una IMU básica con un acelerómetro (inclinómetro) y un giróscopo. Como esta unidad se encontrará ubicada en un ambiente electromagnéticamente ruidoso, se deben privilegiar los sensores con interfaz digital. Así, el acelerómetro utilizado será un ADIS16203 [16] (ver Figura 3.2) de la empresa Analog Devices, el cual tiene la ventaja de entregar directamente el valor de la inclinación, ahorrando los cálculos de conversión respectivos. Algunas características importantes de este sensor se muestran en la Tabla 3.2.

Modelo	ADIS16203
Voltaje	3.3 V
Interfaz	SPI
Rango	360^o
Resolución	14 bits, 0.025^o/bit
Empaquetado	LGA

Tabla 3.2: Datos del inclinómetro ADIS16203.

Este sensor, basado en la tecnología *MEMS*[3], tiene la ventaja de que al utilizar una interfaz digital SPI es altamente programable. La medición de inclinación puede entregarse en distintos formatos (0-360^o, o bien +/-180^o), setearse el ángulo 0^o en distintas posiciones, determinar tiempos de muestreo, generar alarmas para distintas condiciones, integra también un sensor de temperatura y conversores digital-análogo, entre otras características. Si bien todas estas características pueden ser útiles, en este caso se utilizará únicamente la medición del ángulo, para lo cual se debe ajustar el sensor en términos de calibrar el cero y el tiempo de muestreo.

Respecto del giróscopo, se había seleccionado en un inicio el modelo ADIS16250[23], también de Analog Devices, el cual posee características similares al inclinómetro seleccionado, siendo posible ajustar, además, el rango de velocidades de giro aceptadas. Es decir, los mismos 14 bits con que cuenta este sensor pueden ser utilizados en rangos de 320 ^{o}/s hasta 80 ^{o}/s. Sin embargo, dada la posible aplicación de una IMU en misiles o sistemas similares, EEUU limita la exportación

[3]Microelectromechanical Systems

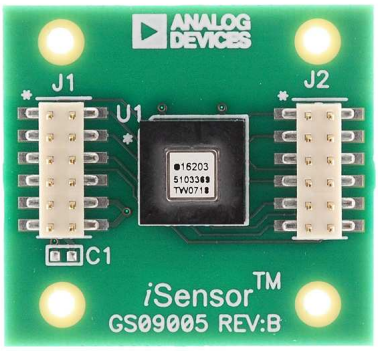


Figura 3.2: Inclinómetro ADIS16203.

de giróscopos electrónicos a ciertos países solamente, entre los cuales Chile no está incluido. Por este motivo, hubo que buscar una solución a nivel nacional. Para esta aplicación, la única alternativa viable es el sensor IDG-300 [24] (ver Figura 3.3), de la empresa Invensense. Este sensor es un giróscopo de 2 ejes que entrega su medición mediante un voltaje análogo. Sus principales características se muestran en la Tabla 3.3.

Modelo	IDG-300
Voltaje	3.3 V
Interfaz	Análoga
Rango	500 ^{o}/s
Resolución	2 mV ^{o}/s
Empaquetado	PCB

Tabla 3.3: Datos del giróscopo IDG-300.

Si bien cumple su función, este giróscopo tiene la desventaja de que es más susceptible al ruido debido a su salida análoga, además de la gran sensibilidad al giro que posee (2 mV por cada ^{o}/s). A todo esto se suma que el chip tiene un error de diseño que hace que el valor análogo para la situacion estacionaria (Zero Rate Output) no es constante en el tiempo (error distinto al *drift* mencionado en la Sección 2.3.1), lo cual hace más dificil determinar si una medición dada corresponde a una cierta velocidad de giro. El fabricante recomienda determinar por otros medios el momento en que el dispositivo no está girando, para tomar la medición y asignar a dicho valor la referencia para el caso estacionario. En este caso, esto se puede realizar mediante el uso de los datos del inclinómetro,

pero con la consiguiente complejidad en el algoritmo.

Figura 3.3: Giróscopo IDG-300.

3.1.3. Comunicaciones

El esquema de comunicaciones cuenta con 2 puertos SCI[14], los cuales pueden ser fácilmente utilizados para comunicarse con un computador, utilizando el protocolo RS232 y un conversor de datos adecuado, mediante un transductor que convierta los niveles de voltaje, o bien con otro microcontrolador. Para mayor flexibilidad, uno de dichos puertos tiene la opción de usarse alternativamente, mediante un chip conversor FT232R[15], como un puerto USB, pudiendo establecer una comunicación directa con computadores sin necesidad de utilizar el transceiver, tal como se muestra en la Figura 3.4.

Anexo a las comunicaciones basadas en los puertos SCI, la placa tiene disponible un bus SPI [21] y un bus CAN [22], de modo de ofrecer la posibilidad de conectar otro tipo de dispositivos, como sensores digitales, memorias (a través del bus SPI) o bien para integrarla en un esquema de conectividad tipo CAN, como sería posible hacerlo con la plataforma PCIVE [9] por ejemplo.

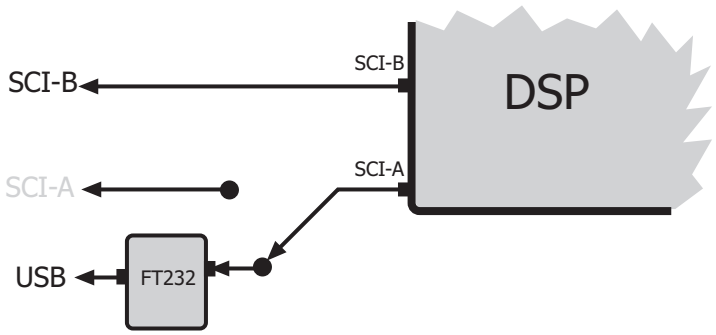


Figura 3.4: Esquema de Comunicaciones

3.1.4. Conversores Análogo-Digital (ADC)

Este bloque es el que se utilizará para medir las distintas variables del sistema que posean una señal análoga como salida. El DSP elegido tiene 2 módulos ADC [25] de 8 canales cada uno. Sin embargo, solo se dejó disponible uno de dichos módulos. Así, se tienen 8 canales ADC de 12 bits cada uno, con un valor máximo de 3 V (representa el valor digital 4095) y un mínimo de 0 V (0 digital). Este módulo debió ser uno de los primeros en ser ruteados en PCB, ya que las señales deben estar lo más próximas al DSP posible y aisladas de cualquer señal de alta frecuencia, de modo que el valor obtenido sea fiel.

Esté módulo se utiliza para obtener el valor de las siguientes variables:

- Giróscopo IDG-300, eje X
- Giróscopo IDG-300, eje Y
- Sensor de corriente motor izquierdo
- Sensor de corriente motor derecho
- Estado de baterías

Para monitorear la corriente en los motores se utilizan sensores de efecto Hall, modelo LEM LTS 15-NP [34]. Para el estado de las baterías, se utiliza simplemente un divisor de tensión protegido por un diodo zener de 3.3 V.

3.1.5. Módulo de PWM

Esta etapa se encarga de entregar 4 señales PWM por cada motor, haciendo un total de 8 señales PWM de salida. Esto es directo desde el DSP, sin embargo este dispositivo trabaja en niveles de 3.3 V, mientras que los drivers de los mosfets que utiliza la etapa de potencia requieren de 15 V para su encendido, por lo que se hace necesario una etapa que eleve en tensión dichas señales.

Para dicha labor se utilizan dos inversores en cascada: un 74LS06[26] y un CD4049[27]. El 74LS06 es un inversor que trabaja en niveles de 0-5 V (por lo que los 3.3 V del DSP los toma como un ’1’ lógico) cuya salida es *open collector*. De este modo, las señales PWM ya se pueden replicar en la salida de este chip en el nivel de tensión deseado (15 V). Luego, el CD4049 (alimentado en 15 V) invierte y acondiciona nuevamente la señal de modo de dejarla lista para ser entregada a la etapa de potencia (es decir, a los mosfets).

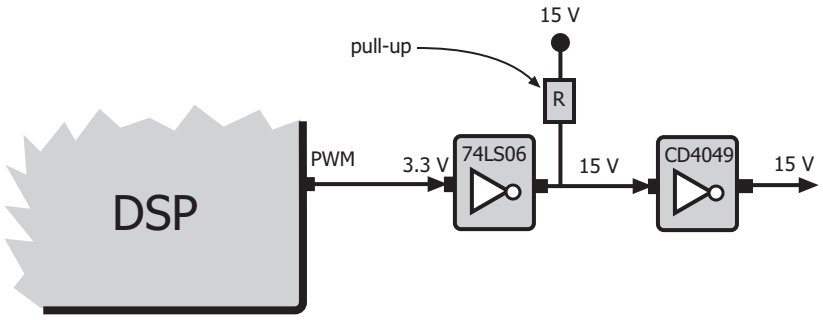


Figura 3.5: Esquema de Elevacion de Tensión para PWM.

3.1.6. Encoders

Los encoders son básicamente 2 trenes de pulsos desfasados en 90º (en cuadratura). El DSP tiene integrado un bloque encargado de manejar este tipo de señales: el módulo eQEP[4] [28]. Este módulo es capaz de calcular la posición (absoluta o relativa), la velocidad y el sentido de giro del encoder en cuestión. No es necesario utilizar circutos de interfaz, por lo que los pines correspondientes del DSP quedan ruteados directamente a las salidas. Dichos encoders se construyeron en base a switches ópticos H22A1 [35], que se muestra en la Figura 3.6.

Figura 3.6: Switch óptico H22A1.

3.1.7. Alimentación

La alimentación de la placa es un tema esencial, ya que es la etapa encargada de llevar la energía a los distintos dispositivos existentes en ella. Se requieren 4 niveles de tensión distintos: 1.8 V (core del DSP), 3.3 V (I/O del DSP, sensores), 5 V y 15 V (buffers de PWM). De modo de estandarizar el diseño, la alimentación principal se realiza en niveles de 5 V, a partir de los cuales deben obtenerse los 1.8 V y 3.3 V que requiere el DSP. Esto se realiza a través del chip TPS767D318[29], recomendado por Texas Instruments para este DSP, el que entrega directamente

[4]Enhanced Quadrature Encoder Pulse

los niveles de tensión requeridos. Este dispositivo es un regulador lineal de baja caída capaz de entregar 1 A en cada nivel de tensión.

Respecto de las tensiones de 5 V y 15 V, éstas deben ser entregadas desde fuera. Sin embargo, existe la opción (mediante un jumper) de obtener los 5 V a través de la conectividad USB dada a la placa. Así, es posible trabajar con ella fácilmente manteniéndola conectada a un computador.

3.1.8. Implementación

La placa se diseñó con el software Orcad 15.7. El ruteo se realizó por bloques, ubicando en primer lugar los módulos más sensibles a la distancia al DSP y a la complejidad de sus líneas. Así, se dio prioridad al módulo ADC (por su sensibilidad al ruido), al inclinómetro (por ser uno de los sensores principales), al módulo de alimentación (cuidando en particular la alimentación del DSP) y al conector JTAG (para permitir un efectivo *debugging* del DSP). Además, se mantuvo el módulo ADC lo más aislado posible de las salidas PWM, las cuales inducen una fuerte componente de ruido debido a la alta frecuencia a la que trabaja dicho módulo.

Finalmente, esto dio como resultado una placa de 100 x 70 mm, la que se muestra en la Figura 3.7.

Este diseño tiene 2 problemas menores, pero que deben ser resueltos antes del montaje de componentes: se cruza tanto una línea de 3.3 V como la línea RX del puerto SCI-A con tierra. Para solucionarlo, basta cortar la pista en el primer caso (quedando aislado un condensador), en cambio, en el segundo caso, además de cortar la pista, ésta se debió conectar por encima con un cable, de modo de mantener establecida la comunicación serial.

Para el ensamble de los distintos componentes se optó por montar por bloques, de modo de poder probar el sistema a medida que se avanzaba. Para efectuar esto la primera etapa en montar-

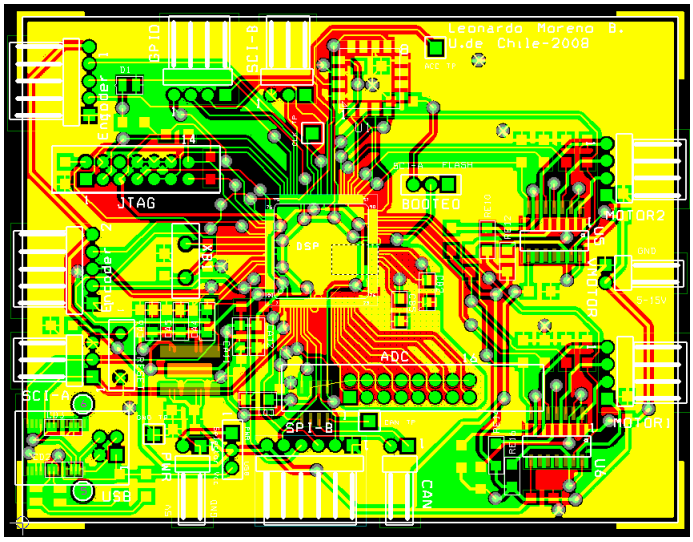


Figura 3.7: Ruteo de placa de control en software CAD.

se debía ser la alimentación, de modo de verificar que los niveles de tensión fuesen los correctos. A continuación se montó la etapa conversora RS232 a USB [15], luego el bloque asociado a las señales PWM, terminando con el montaje del DSP y el resto de los componentes (excepto el inclinómetro y el giróscopo). Una vez comprobado el correcto funcionamiento del DSP, se montaron los sensores faltantes, completando la implementación en hardware de la IMU (ver Figura 3.8).

Es importante notar (Figura 3.8) que el giróscopo utilizado debía ser montado perpendicular a la placa, de modo que la medición en uno de sus ejes coincidiera con el eje de medición del acelerómetro.

3.2. Electrónica de Potencia

El diseño de esta etapa es relativamente simple, ya que se están utilizando motores de corriente continua como actuadores. Esta etapa implementa un circuito tipo *puente H* [19], el cual mediante señales PWM permite modular el voltaje promedio en el motor y el sentido de giro de éste depen-

Figura 3.8: Placa de control y sensores.

diendo de cuales mosfets se encienden (y de como lo hacen). En la Figura 3.9, se puede ver que tanto el interruptor I_{1A} como el I_{1B} se encuentran encendidos, lo que permite que la corriente fluya a través de ellos y de la armadura del motor. Para cambiar el sentido de giro, basta apagar I_{1A} y I_{1B}, para luego encender I_{2A} y I_{2B}.

Para controlar la tensión en el motor, se varía el ciclo de trabajo del interruptor superior, manteniendo el correspondiente abajo encendido con un ciclo de trabajo muy alto, pero nunca del 100 % (para evitar tenerlo conduciendo todo el tiempo). En la Figura 3.10 se aprecia un detalle de las señales en los respectivos interruptores.

Para la implementación de dichos interruptores se utilizarán mosfets IRFP2907Z, cuyas principales características se muestran en la Tabla 3.4.

Cada interruptor del puente H consistirá de dos mosfets en paralelo, los cuales recibiran las mismas señales de encendido, dando así dando una capacidad teórica al puente H de 180 A, superando holgadamente los 100 A máximos que consume el motor [17]. Para encender cada uno de

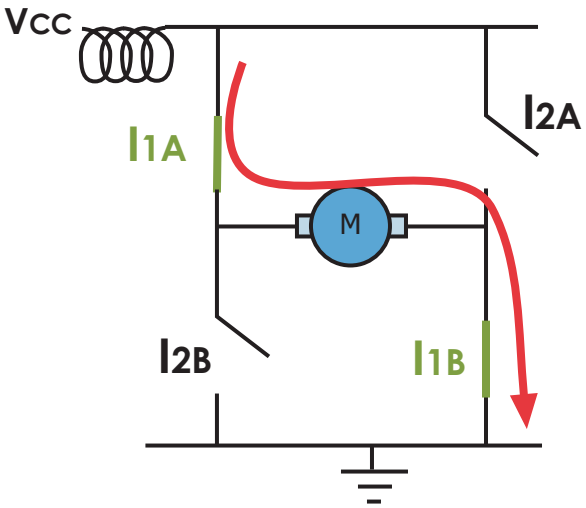


Figura 3.9: Esquema de funcionamiento puente H.

Modelo	IRFP2907Z
V_{DS}	75 V
R_{DS}	4.5 mΩ
Corriente Máxima	90 A

Tabla 3.4: Datos IRFP2907Z.

los mosfets se utilizan circuitos integrados especializados en ello. Para el encendido de los mosfets superiores (los que quedan levantados de tierra) se utiliza el chip IR2117 [31], el cual se encarga de sensar la tensión en la fuente de los correspondientes mosfets de modo de aplicar el voltaje exacto para encender o apagar el interruptor, según sea el caso. Este chip tiene una serie de componentes externos necesarios para su correcto funcionamiento, cuyos parámetros dependen de variables como el voltaje en que se trabaja, la capacitancia de los mosfets o la frecuencia de las señales PWM. Para determinar dichos valores se debe utilizar una nota de aplicación disponible en el sitio web de Interntional Rectifier [32]. Para los mosfets inferiores se utiliza un driver para mosfet modelo ICL7667 [33], de la compañía Intersil.

Se incluyó, además, un banco de condensadores de 2400 uF, un inductor de 1 mH como filtro de corriente y snubbers basados en TVS[5] como protección por sobretensión, además de disipadores

[5]Transient Voltage Suppressor

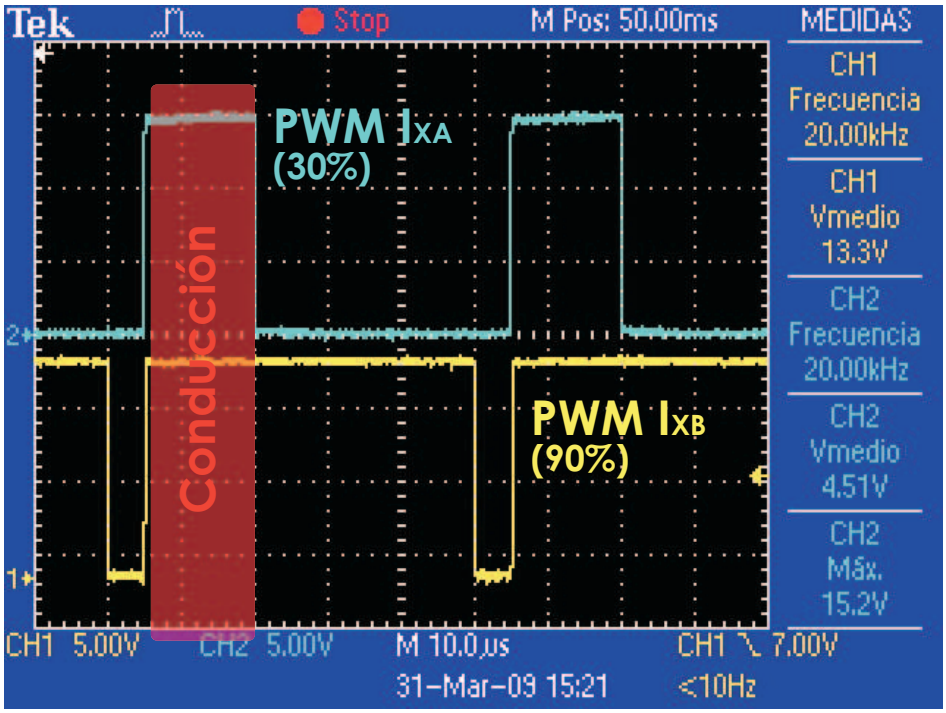


Figura 3.10: Señales PWM aplicadas al puente H.

en cada mosfet para prevenir sobrecalentamientos.

3.2.1. Implementación

El ruteo de la placa es bastante simple, solamente hay que considerar el tamaño de los disipadores que se instalarán en los mosfets, además de rutear las líneas lo suficientemente gruesas como para que no se aprecie un efecto inductivo en ellas. Así, se obtuvo una placa de 170x130 mm, la que se aprecia en la Figura 3.11.

Esta placa requiere, entonces, una tensión para el motor de 48 V como máximo, 4 señales PWM (una por cada interruptor) y una alimentación de 15 V. Como ya se dijo, un tema importante en esta etapa es el mantener una inductancia baja en las líneas, de modo de evitar sobretensiones asociadas

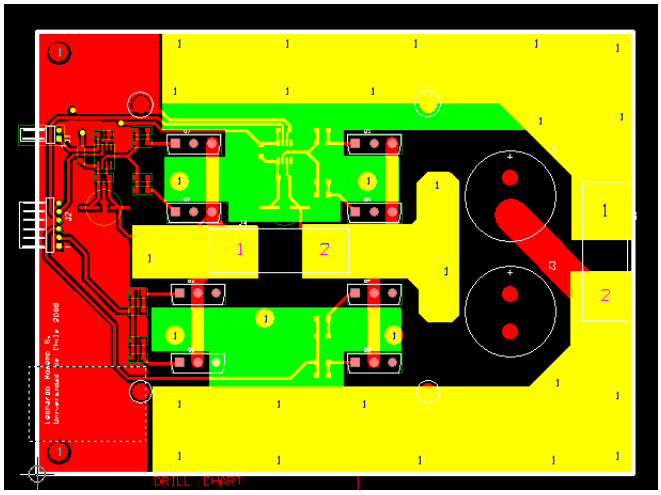


Figura 3.11: Ruteo de puente H.

al corte de la corriente[6]. Para ello, las pistas se reforzaron con trozos sólidos de cobre (2.5 mm) unidos mediante pernos M5. Para evitar movimientos en los mosfets, se rellenaron algunos espacios con silicona.

A pesar de las precauciones tomadas, en un inicio se quemaron bastantes mosfets por sobretensión. Por este motivo se instalaron los diodos TVS (los cuales no estaban contemplados en el diseño original). Estos semiconductores [39] son básicamente diodos zener capaces de absorber altas cantidades de energía por períodos cortos (cerca de 1500 W). Así, se instalaron en los terminales de cada motor como se aprecia en la Figura 3.12, lo que permite bloquear tensiones que superen los 63 V y que estén bajo los -5 V. Esta solución permitió finalmente llevar a cabo la etapa de pruebas de un modo satisfactorio.

Para efectos de montaje, se instaló toda la circuitería asociada al encendido de los mosfets, de modo de verificar que dichas señales llegasen sin ningún problema. Luego, se revisó cada mosfet por separado antes de instalarlo. Finalmente, se observó el funcionamiento de cada interruptor por separado en la placa ya montada (Figura 3.13), para luego hacer una prueba completa con una carga pequeña y con el motor.

[6]ya que en una inductancia, $V = L \cdot \frac{di}{dt}$, por lo que un corte abrupto en la corriente provoca un impulso en la tensión

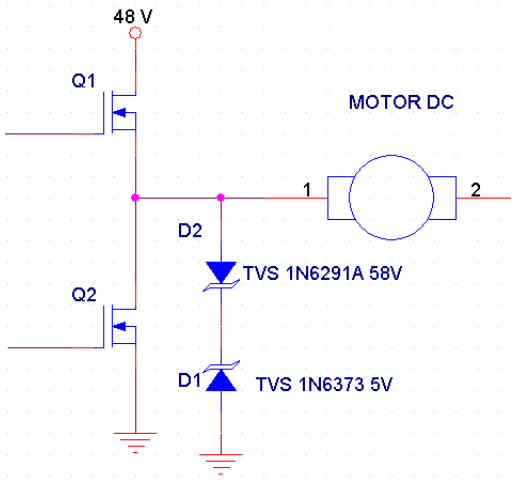


Figura 3.12: Esquema de conexión de TVS.

3.3. Electrónica Anexa

Además de la electrónica de control y de la electrónica de potencia, el sistema requiere cierta electrónica anexa que preste servicios a los bloques principales. En particular, se requiere de una fuente que entregue los niveles de tensión requeridos y de una interfaz con el usuario.

3.3.1. Fuente Switching

La fuente del sistema debe tener los siguientes requisitos básicos:

- Niveles de tensión: 5 V y 15 V
- Corriente: 2 A
- Alimentación: 24 V

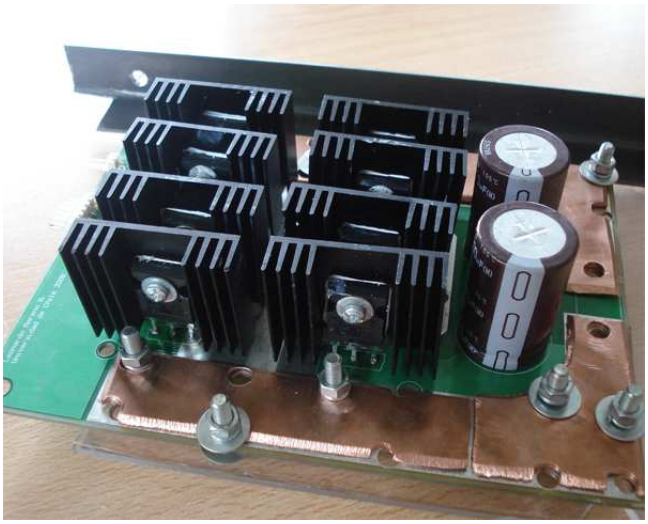


Figura 3.13: Puente H.

En un inicio el sistema utilizaba una fuente basada en reguladores lineales de la familia 78xx. Sin embargo, esto trajo consigo varios problemas, principalmente en términos de disipación de calor, ya que en total la electrónica consume alrededor de 1 A, por lo que la potencia que se debe disipar está dada por $P_{dis} = 1[A] \cdot (24 - 15)[V] = 9[W]$. El calentamiento excesivo de estos reguladores empeoraba la calidad de los niveles de tensión. Por este motivo, se requería una fuente más eficiente.

La solución se implementó en base al circuto integrado LM2576 [36], de National Semiconductor. Este chip permite diseñar una fuente switching de una eficiencia μ = 70 % y hasta 3 A con tan sólo unos pocos componentes externos. Fue necesario diseñar y fabricar un par de bobinas y tomar ciertas precauciones en el ruteo (todo esto especificado en el datasheet [36]). En el diseño se utilizó la versión regulable del chip para obtener los 15 V (alimentado desde 24 V) y, en cascada a este, la versión fija en 5 V. En total, la fuente entrega hasta 3 A, en 3 salidas en 5 V y 3 en 15 V. La Figura 3.14 muestra la fuente ya construida.

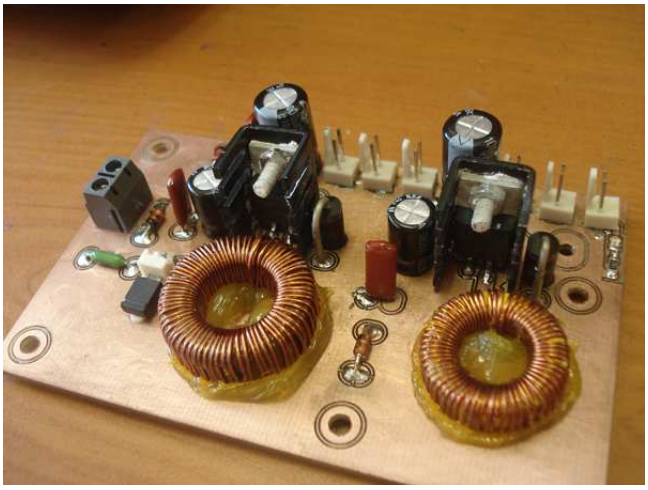


Figura 3.14: Fuente Switching.

3.3.2. Interfaz usuario

Para poder mostrar información del estado del vehículo al usuario, se diseñó una placa independiente a la electrónica de control. Esta placa está basada en un micocontrolador PIC18LF252 [37], de Microchip, el cual tiene la particularidad de usar un rango más amplio de voltaje que la versión simple PIC18F252, permitiendo trabajarlo en 3.3 V, al igual que el DSP. De este modo, no hay que cambiar niveles de tensión para comunicar ambos microcontroladores.

Las funciones de esta interfaz son las siguientes:

- Recibir información de las variables del sistema desde el DSP
- Mostrar la información actualizada al usuario mediante una pantalla LCD
- Obtener el valor de la referencia de giro del sistema desde el volante de dirección y enviar dicha variable al DSP.

La placa diseñada cumple con todos estos requisitos (Figura 3.15), manteniendo un enlace

constante con el DSP mediante el puerto RS232, actualizando cada 100 ms la información de la pantalla LCD mediante la implementación del protocolo Hitachi HD44780 [38] y leyendo a través de sus puertos ADC un potenciómetro adosado al volante a modo de encoder de giro.

Figura 3.15: Pantalla LCD.

La ventaja de utilizar un microcontrolador para este fin es que no es un simple enlace entre la placa de control y la pantalla LCD, sino que permite realizar cierto procesamiento de datos que simplifique el trabajo del DSP, por ejemplo, preocupándose en este nivel del formato de datos o de realizar ciertas comprobaciones del encoder. Esta interfaz fue una etapa fundamental en la etapa de pruebas, ya que permitía mantener información del estado del sistema visible incluso cuando el vehículo estaba en movimiento (y por ende, lejano a un PC).

La programación del microcontrolador asociado se realiza sin necesidad de remover el chip, mediante el protocolo ICSP[7] [40]. Para dicho efecto, la placa dispone de un conector asociado, a través del cual se comunica con el puerto serie del PC con un programador casero.

3.4. Diseño Mecánico

Como se estableció previamente, el diseño mecánico se realizó buscando la mayor simpleza posible y siempre con la premisa de que la estructura debiese ser robusta. Para ello, la caja principal

[7] In-Circuit Serial Programming

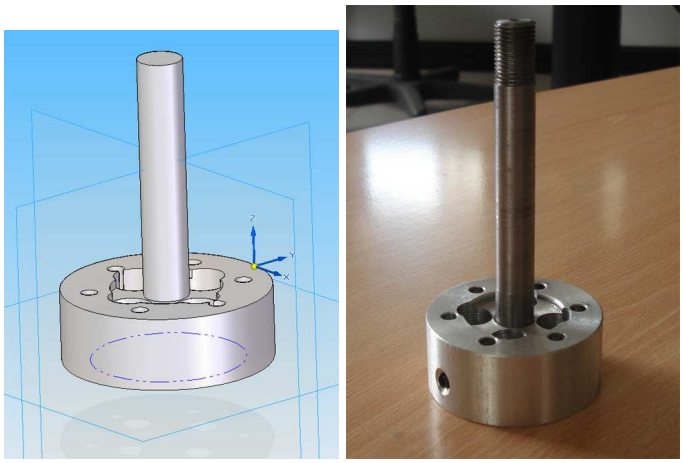
debe estar totalmente contenida en el radio de las ruedas de modo que no se golpee ante una eventual caída. Así, la elección de las ruedas resultaba vital. Se optó por ruedas de BMX de 20” de diámetro, con masas Shimano para freno de disco (ver Figura 3.16). El hecho de que fuesen masas para freno de disco era para disponer de los orificios destinados a los pernos de los discos para lograr el acople del motor con las ruedas.

Figura 3.16: Ruedas utilizadas.

Fue necesario, entonces, diseñar una pieza que permitiera acoplar la masa de la rueda con el eje del motor. Para ello, se diseñó y construyó una pieza en base a duraluminio (ver Figura 3.17) que fuese capaz de ajustarse al eje del motor y al mismo tiempo a pernos ubicados en los orificios para freno de disco de la masa de la rueda (detalle en Figura 3.18), los cuales ejercen únicamente la torsión. El movimiento axial de la rueda se evita mediante un eje de acero endurecido que la atraviesa y posee una tuerca con seguro plástico en su extremo.

La estructura se fabricó en base a un esqueleto de pletina de acero soldado (Figura 3.19). La dirección se realizó también en base a duraluminio, adaptando un juego de dirección de bicicletas de modo de obtener el medio de giro del vehículo.

La electrónica, tanto de control como de potencia, cuelga del piso del vehículo, quedando justo



(a) (b)

Figura 3.17: Acople de ruedas con eje de motor.

entre los dos motores. Las 4 baterías de gel UCG-1812 [41], de 12 V y 18 Ah, se ubican a los costados de los motores, sujetas con cinturones de velcro remachados a la estructura.

3.5. Esquema de Control

El esquema de control implementado sufrió varios ajustes según avanzaba la etapa de pruebas, en particular respecto de la parte propocional, alejándose así del controlador diseñado para pasar a un controlador más empírico. En efecto, a la parte proporcional del controlador se le asoció un offset, ya que existe un rango de bajos ciclos de trabajo donde el vehículo no sale del reposo, así, al sumar por defecto un valor se puede garantizar el sacar rápidamente al vehículo del estado de reposo. Además, se hizo necesario trabajar por tramos, ya que muchas veces la respuesta que funcionaba bien en torno a 0^o no lo hacía del mismo modo en velocidad, o bien el giro tuvo que ser limitado en ciertos ángulos, ya que se obtenían giros demasiado bruscos cuando el vehículo ya estaba en movimiento. Respecto de la parte derivativa, se tuvo que limitar bastante, ya que debido a los problemas comentados en la Sección 3.1.2 acerca de la unidad IMU, el sensor tendía a dispararse dadas ciertas condiciones, lo que podía provocar que el pasajero fuese lanzado violentamente del vehículo.

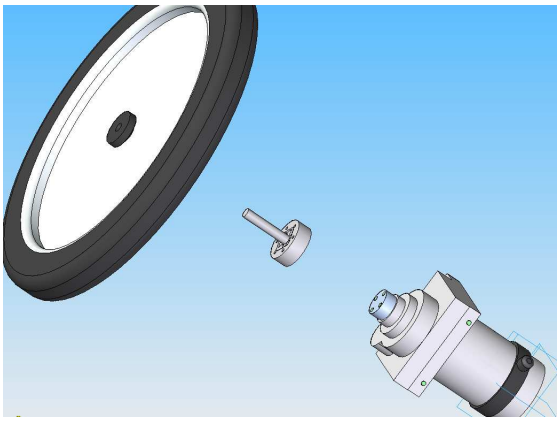


Figura 3.18: Acoplamiento ruedas y motor.

(a) (b)

Figura 3.19: Diseño y estructura del prototipo.

Capítulo 4

Resultados y Análisis

Como resultado del trabajo se obtuvo, en primer lugar, un vehículo funcional, capaz de trasladar a una persona (con cierto tiempo de entrenamiento) sin problemas de un lugar a otro en tramos cortos. Las características principales del prototipo logrado se muestran el la Tabla 4.1.

Peso	40 [kg]
Velocidad Máxima	6 [km/hr]
Ángulo máximo	10^o
Ruedas	BMX Aro 20”
Autonomía	mayor a 5 horas
Ancho	0.9 [m]
Alto	1.45 [m]
Largo	0.5 [m]
Distancia del chasis al suelo	0.1 [m]
Radio de Giro	0 [mt]

Tabla 4.1: Principales características del prototipo implementado.

Además, la electrónica desarrollada alcanzó un cierto grado de madurez que permitió finalmente realizar pruebas sin temor a que se quemaran componentes, lo que significó un rápido avance en las pruebas una vez que éstas se dieron sin problemas. Así, es fácil tomar esta plataforma e intentar

implementar nuevas estrategias de control. Sin embargo, se deben considerar diversos detalles, principalmente asociados a la unidad IMU implementada, los cuales se analizan a continuación.

4.1. Sensores

En términos de adquisición de datos no hubo mayores problemas, pero se hace necesario analizar el caso de la unidad IMU implementada, la cual requiere un cierto nivel de procesamiento para obtener la estimación buscada. El principal problema radicó en las dificultades en la utilización del giróscopo. Este sensor, como se comentó en la Sección 2.3.1, se utiliza para realizar una corrección en el corto plazo de la estimación del ángulo (instante en que el acelerómetro no entrega una buena estimación). Sin embargo, dados los problemas que posee el sensor IDG-300, en términos de que se hace necesaria una corrección de éste **mediante el acelerómetro (mismo sensor que corrige)**, provoca que dicha corrección sea a veces errónea, haciendo creer al sistema que este se está moviendo cuando en realidad no lo está (muchas veces a grandes velocidades) y viceversa. Esto provoca que el sistema inicie una oscilación que muchas veces puede ser muy brusca. Esto ocurre en dos ocasiones: cuando las ganancias del controlador se setean lo suficientemente grandes como para que el sistema experimente un pequeño tirón (el cual inicia la pérdida de sentido en las mediciones del giróscopo), o bien cuando el sistema se encuentra manejado por alguien en fase de aprendizaje (independientemente de las ganancias), ya que el usuario inexperto muchas veces se bajará bruscamente, iniciando él mismo la oscilación, la cual a veces es controlable (manteniendo el sistema lo mas quieto posible debiese recuperarse), pero esto no siempre ocurre.

Dada la naturaleza recursiva del problema es que el ajuste de parámetros del controlador debe ser muy fino, ya que el salirse de ciertos rangos traerá consigo sobreoscilaciones producto de la pérdida de validez de las estimaciones, en lugar de sobreoscilaciones asociadas al comportamiento esperado de un sistema común. Para corregir esto, o bien se debe afinar el método de detección de velocidad cero del sistema (de modo de detectar efectivamente cuando el vehículo no está inclinándose) o bien, se debe conseguir un giróscopo que no tenga dicho problema, lo cual necesariamente pasa por lograr importarlo. En ese caso, lo ideal es traer el sensor con que se diseñó originalmente la IMU, el ADIS16250 [23], lo que permitiría evitar correcciones y utilizar los datos casi directa-

mente.

4.2. Estabilidad y Control

Como se analizó en la Sección 2.3.4, el sistema es controlable considerando al pasajero como una barra rígida. Sin embargo, esto no es así, ya que la persona inconscientemente realiza un control que ayuda al sistema a mantenerse erguido. Este hecho es el que logra que el sistema funcione relativamente bien, ya que dado que las ganancias no pueden ser muy grandes por los motivos ya expuestos en la Sección 4.1, para iniciar el movimiento hay que llegar a una inclinación mayor a la deseable en un funcionamiento óptimo. Por esto, el pasajero debe contribuir al equilibrio del sistema haciendo en parte equilibrio, en particular cuando se está aprendiendo a utilizar el vehículo, ya que las reacciones del pasajero son excesivas. Una vez obtenido cierto dominio, ya es más fácil controlar el vehículo y guiarlo en la dirección y velocidad deseadas, pero siempre intentando mantener la estabilidad (es decir, el prototipo no es capaz de estabilizarse por sí mismo).

4.3. Análisis de Costos

En esta sección se realizará un análisis de los costos asociados al desarrollo del proyecto. Para realizar dicho análisis se considerarán 3 puntos escenciales:

- Costo de materiales
- Mano de obra
- Arriendo de taller mecánico

A modo de información se debe considerar además el valor de una unidad comercial de este tipo de vehículos. Dado que la única referencia en este sentido es el Segway[2], se utilizará el valor de una unidad *i2*, la cual es la unidad más básica existente. En Chile, al día 24 de abril del 2009, adquirir una unidad de este tipo cuesta $4.300.000 (IVA incluido) aproximadamente.

Al analizar los costos de materiales (ver Tabla 4.2), se incluyen los costos por internación y aduana en el caso de los elementos importados, los cuales ascendieron a $506.665.

Electronica de Control	$388.970
Elementos Mecánicos	$121.173
Electrónica de Potencia	$396.918
Motores y Baterías	$736.544
Otros	$19.875
Total Materiales	$1.663.480

Tabla 4.2: Costo de materiales.

Para calcular el costo del uso de taller mecánico, se considera un valor promedio de $8.000/hr. Dado que se utilizó por 160 hrs en total, el costo por este ítem asciende a $1.280.000.

Finalmente, para el costo de mano de obra se considera un sueldo de ingeniero de $800.000 mensuales[1]. Así, la hora-ingeniero tiene un valor de $4.545. Considerando un trabajo de 8 meses a medio tiempo (4 horas diarías), el costo de mano de obra del proyecto es de $3.200.000. Así, los costos totales del proyecto se muestran en la Tabla 4.3.

Costo Materiales	$1.663.480
Uso de Taller	$1.280.000
Mano de Obra	$3.200.000
Total Proyecto	$6.143.480

Tabla 4.3: Costo total del proyecto.

De la Tabla 4.3 se tiene, entonces, que el costo total del proyecto fue de $6.143.480.Este valor es casi dos millones de pesos mayor que adquirir una unidad Segway *i2* [2], sin embargo, hay que

[1]Fuente: www.futurolaboral.cl

considerar dos puntos importantes: en este resumen de costos no cuentan las economías de escala (construir una única unidad es más costoso que en una línea de producción) y, lo más importante, que este es un primer prototipo, es decir, el costo incluye una fuerte componente de diseño de que no existía previamente, es decir, hay un costo por la experiencia adquirida (*know how*). Es fácil ver que dicho costo se traduce principalmente en mano de obra (lo más costoso), la cual en este caso incluye muchas horas-ingeniero, las que ya no se considerarían en su totalidad en el caso de la implementación de una segunda o tercera unidad. En efecto, al realizar el mismo ejercicio para una segunda unidad, es decir, los costos por materiales no varían (no hay economías de escala), pero si se considera un 50 % de tiempo de desarrollo respecto de la primera unidad (tanto en mano de obra como uso de taller) y con el valor de horas-hombre de técnico[2] ($3.125) en vez de ingeniero, se obtienen los datos de la Tabla 4.4.

Costo Materiales	$1.663.480
Uso de Taller	$640.000
Mano de Obra	$1.100.000
Total Proyecto	$3.403.480

Tabla 4.4: Costo total de una segunda unidad.

Así, se puede ver que no es descabellada la idea de construir este tipo de vehículos en el país de modo competitivo. Sin embargo, hay que tener en consideración que se realizaron aproximaciones muy gruesas para obtener estos valores. En estricto rigor, si se piensa en realizar versiones comerciales de este vehículo debiese realizarse un estudio de costos más acabado en conjunto con un estudio de mercado.

[2]Tomando un sueldo de $550.000 mensual. Fuente: www.futurolaboral.cl

Capítulo 5

Conclusiones

En este trabajo se presentó toda la fase de diseño e implementación de un vehículo eléctrico de autobalanceo, incluyendo instrumentación, aspectos mecánicos, ajuste de controladores, etc. Este tipo de vehículos tiene prácticamente un único exponente con el Segway [2] y sus diferentes versiones, el cual sin embargo no ha tenido el éxito esperado debido principalmente a los altos costos en que se debe incurrir para adquirir una unidad, más aún si se está en un país alejado de los principales centros de producción de dichos vehículos.

El vehículo implementado en este proyecto logró ser funcional en el sentido de que una persona puede transportarse sin mayores problemas de un lugar a otro, evitando obstáculos, sobre superficies irregulares e incluso cruzando calles. Además, todo se realizó manteniéndose dentro de los costos esperados, lo que muestra que puede ser factible realizar este tipo de diseños en Chile y lograr que funcionen. Sin embargo, la curva de aprendizaje para la conducción del vehículo no fue la esperada, ya que se buscaba obtener resultados similares al Segway [2], el que requiere de unos 5 minutos de práctica para manejarlo con comodidad. En este caso, se requiere de alrededor de 30 minutos de práctica poder mantener el equilibrio sin problemas, lo cual si bien es más de lo deseado, está dentro de rangos razonables.

Gran parte de los problemas asociados al equilibrio del sistema están asociados al desarrollo de la unidad IMU, en particular, al giróscopo. Este sensor requiere una constante corrección, la cual muchas veces no funcionó del todo bien. Esto provoca que el sistema tienda a oscilar cuando el pasajero aún no se acostumbra al manejo del vehículo. Sin embargo, esto sucede sólo en ciertos casos, siendo el resto del tiempo la unidad IMU diseñada funcional dentro del rango de funcionamento designado (evitar cambios bruscos). Así, bastaría mejorar la calidad (y cantidad) de sensores disponibles para poder implementar una unidad IMU de mejor comportamiento, la cual no solo podría mejorar el funcionamiento de este tipo de vehículos, sino que también sería aplicable a otras implementaciones que requieran estimar la orientación de algún sistema, teniendo grandes aplicaciones en el campo de la aeronáutica y en el área militar.

Respecto del control implementado, si bien tenía una cierta base teórica, terminó siendo un control más bien empírico, dado por ajustes de parámetros y curvas de comportamiento según las pruebas realizadas. Sin embargo, queda implementada una instrumentación bastante completa del estado del vehículo que permite realizar controles más complejos que puedan obtener mejores resultados en términos de manejo.

En términos de autonomía se puede decir que el vehículo nunca quedó a medio camino por falta de baterías. Se hicieron muchas pruebas, incluso de más de una hora seguida varias veces en un día, y el comportamiento no cambió. Las pruebas que se hicieron para determinar la autonomía del vehículo no lograron descargar las baterías lo suficiente como para hacer inviable el manejo. Por ende, el prototipo debería ser capaz de soportar sin problemas un viaje de ida y vuelta dentro de un radio cercano (alrededor de 2 a 3 km), como se buscó originalmente con el desarrollo de este tipo de vehículos.

En conclusión, se cumplió el principal objetivo de este trabajo, que era lograr un vehículo funcional capaz de transportar a una persona (ver Figura 5.1). Sin embargo, hay mucho desarrollo por delante para futuras versiones de modo de lograr una curva de aprendizaje de conducción más rápida, es decir, el prototipo actual no es capaz de transportar a cualquier persona de inmediato, ya que se requiere cierto entrenamiento y valentía en los inicios, a pesar de que en términos de metodología de conducción sigue siendo mucho más simple que cualquer otro vehículo existente, ya que no tiene comandos ni palancas de cambio que se deban manejar.

Figura 5.1: Prototipo en funcionamiento.

Además, este proyecto mostró que es factible desarrollar este tipo de medio de transporte en el país, ya que técnicamente fue posible obtener una unidad funcional, y los costos en que se incurrió no se alejaron demasiado del costo de una unidad comercial, considerando que es una única unidad y que se tuvo que absorber el costo de ser la primera implementación de este tipo en Chile. Así, en el caso de intentar seguir avanzando en el desarrollo de estos vehículos, se reducen notablemente las dificultades y, de ser posible importar mayor y mejor eléctrónica, se abaratan costos y se gana en calidad.

5.1. Trabajo Futuro

Para futuras versiones y/o correcciones de este vehículo, existen varios detalles que se deberían corregir o bien adiciones que se pueden hacer. En primer lugar se encuentra el conseguir giróscopos de mejor calidad de modo de poder mejorar la estimación que la unidad IMU entrega sobre la inclinación del vehículo. Se debería ver la posibilidad de utilizar otro tipo de motores, como los brushless, ya que si bien el control es un poco más complejo, tienen ventajas en términos de tamaño, eficiencia y costos.

En términos de control, resultaría interesante buscar alguna alternativa que se ajuste mejor al tipo de sistema, como implementar un controlador difuso o adaptable.

Respecto de la estimación en la unidad IMU, sería interesante ver la posibilidad de aplicar un filtro de Kalman de modo de obtener la estimación óptima.

También se debería cambiar el método en que se determina el giro deseado, ya que si bien el volante instalado es bastante intuitivo, es peligroso ya que en una eventual caída, en caso de no soltar uno de los dos lados, el vehículo tenderá a girar sin control.

Bibliografía

[1] Segway Inc, *"Energy Credits Spread"* [en línea], http://www.segway.com/downloads/pdfs/Energy_Credits_Spread.pdf [consulta: 23 de mayo del 2008]

[2] Segway Inc, *"Explore Models"* [en línea], http://www.segway.com/individual/models/index.php [consulta: 27 de mayo del 2008]

[3] Segway Inc, *"P.U.M.A. - Segway Advanced Development"* [en línea], http://www.segway.com/puma/ [consulta: 20 de abril del 2009]

[4] Blackwell, T., *"Building a Balancing Scooter"* [en línea], http://www.tlb.org/scooter.html [consulta: 23 de mayo del 2008]

[5] Grasser, F., D'Arrigo, A., Colombi, S. y Rufer, A. C, *"JOE: a mobile, Inverted Pendulum"* , IEEE Transactions on Industrial Electronics, Vol. 49, No 1, 2002.

[6] MIT, *"The DIY Segway"* [en línea] http://web.mit.edu/first/segway/ [consulta: 24 de mayo del 2008]

[7] Dudash, P., Eakins, G., Geier, E. y Gries, J, *"The Two Wheel Deal"* [en línea], http://cobweb.ecn.purdue.edu/~477grp12/index.html [consulta: 24 de mayo del 2008]

[8] Toyota, *"Toyota Develops Personal Transport Assistance Robot Winglet"* [en línea], http://www.toyota.co.jp/en/news/08/0801_1.html [consulta: 20 de febrero del 2009]

[9] Alarcón, C. y Ramírez, J., *"Proyecto de Instrumentación de Vehículos a Escala"* [en línea], http://automatica.li2.uchile.cl/webpage/PIVE.html [consulta: 25 de mayo del 2008]

[10] Alarcón, C., *"Diseño e implementación de interfaz de programación para plataforma orientada al control de vehículos autónomos"*, Memoria de Ingeniero Civil Electricista, Santiago: Departamento de Ingeniería Eléctrica, Universidad de Chile, 2006.

[11] Ramírez, J., *"Diseño y construcción de plataformas reprogramables de comunicación de sensores vía protocolo CAN, aplicado al control de vehículos autónomos"*, Memoria de Ingeniero Civil Electricista, Santiago: Departamento de Ingeniería Eléctrica, Universidad de Chile, 2006.

[12] Texas Instruments, *"TMS320F2809, F2808, F2806, F2802, F2801, C2802, C2801, and F2801x DSPs (Rev. J)"* [en línea], http://focus.ti.com/lit/ds/symlink/tms320f2808.pdf [consulta: 27 de mayo del 2008]

[13] Texas Instruments, *"C2000 Real-Time Microcontrollers"* [en línea], http://focus.ti.com/lit/ml/sprb176e/sprb176e.pdf [consulta: 1 de junio del 2009]

[14] Texas Instruments, *"TMS320x280x, 2801x, 2804x Serial Communication Interface (SCI) Reference Guide"* [en línea], http://www.ti.com/litv/pdf/sprufk7 [consulta: 24 de abril del 2009]

[15] Future Technology Devices International Ltd., *"FT232R USB UART IC Datasheet Version 2.01"* [en línea], http://www.ftdichip.com/Documents/DataSheets/DS_FT232R.pdf [consulta: 24 de abril del 2009]

[16] Analog Devices, *"Analog Devices: Programmable 360° Inclinometer - ADIS1603"* [en línea], http://www.analog.com/UploadedFiles/Data_Sheets/ADIS16203.pdf [consulta: 26 de mayo del 2008]

[17] NPC Robotics, *"NPC-T64: Product Information"* [en línea], http://www.npcrobotics.com/products/viewprod.asp?prod=42&cat=20&mode=gfx [consulta: 27 de mayo del 2008]

[18] V. Madisetti D. B Williams, *"The Digital Signal Processing Handbook"*, CRC Press, 1998, p. 77-4.

[19] Rashid, Muhammad H., *"Electrónica de Potencia - Circuitos, Dispositivos y Aplicaciones"*, Pearson Educación, 2004, ISBN:970-26-0532-6, p. 669-670, .

[20] IEEE Std. 1149.1-2001, *"Test Access Port and Boundary-Scan Architecture"*, IEEE, 2001.

[21] Texas Instruments, *"TMS320x280x, 2801x, 2804x Serial Peripheral Interface (SPI) Reference Guide"* [en línea], http://www.ti.com/litv/pdf/sprug72 [consulta: 24 de abril del 2009]

[22] Texas Instruments, *"TMS320x280x, 2801x DSP Enhanced Controller Area Network (eCAN) User's Guide"* [en línea], http://www.ti.com/litv/pdf/sprueu0 [consulta: 24 de abril del 2009]

[23] Analog Devices, *"ADIS16250: Programmable Low Power Gyroscope"* [en línea], http://www.analog.com/static/imported-files/data_sheets/ADIS16250_16255.pdf [consulta: 26 de mayo del 2008]

[24] Invensense, *"IDG-300 Integrated Dual-Axis Gyroscope"* [en línea], http://www.invensense.com/shared/pdf/DS_IDG300.pdf [consulta: 26 de mayo del 2008]

[25] Texas Instruments, *"TMS320x280x 2801x, 2804x Analog-to-Digital Converter(ADC) Module Reference Guide (Rev. B)"* [en línea], http://www.ti.com/litv/pdf/spru716b [consulta: 24 de abril del 2009]

[26] Texas Instruments, *"SN54LS06, SN74LS06, SN74LS16 (Rev. E)"* [en línea], http://www.ti.com/lit/gpn/sn74ls06 [consulta: 26 de abril del 2009]

[27] Fairchild Semiconductor, *"CD4049UBC Hex Inverting Buffer"* [en línea], http://www.fairchildsemi.com/ds/CD/CD4049UBC.pdf [consulta: 26 de abril del 2009]

[28] Texas Instruments, *"TMS320x280x, 2801x, 2804x Enhanced Quadrature Encoder Pulse Reference Guide (Rev. D)"* [en línea], http://www.ti.com/litv/pdf/spru790d [consulta: 26 de abril del 2009]

[29] Texas Instruments, *"Dual-Output Low-Dropout Voltage Regulators (Rev. H)"* [en línea], http://www.ti.com/lit/gpn/tps767d318 [consulta: 26 de abril del 2009]

[30] International Rectifier, *"IRFP2907Z datasheet"* [en línea], www.irf.com/product-info/datasheets/data/irfp2907z.pdf [consulta: 26 de abril del 2009]

[31] International Rectifier, *"IR2117(S)-IR2118(S) (PbF) revO.p65"* [en línea], www.irf.com/product-info/datasheets/data/ir2117.pdf [consulta: 26 de abril del 2009]

[32] International Rectifier, *"Application Note AN-978 - HV Floating MOS-Gate Driver ICs"* [en línea], http://www.irf.com/technical-info/appnotes/an-978.pdf [consulta: 26 de abril del 2009]

[33] Intersil, *"ICL7667 Dual Power MOSFET Driver"* [en línea], http://www.intersil.com/data/fn/fn2853.pdf [consulta: 26 de abril del 2009]

[34] LEM Holding SA, *"Current Transducer LTS 15-NP"* [en línea], http://www.lem.com/docs/products/lts%2015-np%20e.pdf [consulta: 12 de mayo del 2009]

[35] Fairchild Semiconductors, *"H22A1 Slotted Optical Switch"* [en línea], http://www.fairchildsemi.com/ds/H2%2FH22A1.pdf [consulta: 12 de mayo del 2009]

[36] National Semiconductors, *"LM2576/LM2576HV Series - SIMPLE SWITCHER 3A Step-Down Voltage Regulator"* [en línea], http://www.national.com/ds/LM/LM2576.pdf [consulta: 12 de mayo del 2009]

[37] Microchip, *"PIC18F252 Datasheet"* [en línea], http://ww1.microchip.com/downloads/en/DeviceDoc/39564c.pdf [consulta: 6 de febrero del 2009]

[38] Hitachi, *"Hitachi HD44780U"* [en línea], http://www.sparkfun.com/datasheets/LCD/HD44780.pdf [consulta: 6 de febrero del 2009]

[39] Vishay, *"1.5KE6.8 thru 1.5KE540A, 1N6267 thru 1N6303"* [en línea], http://www.vishay.com/docs/88301/15ke.pdf [consulta: 11 de marzo del 2009]

[40] Microchip, *"In-Circuit Serial Programming (ICSP) Guide"* [en línea], http://ww1.microchip.com/downloads/en/DeviceDoc/30277d.pdf [consulta: 22 de diciembre del 2008]

[41] Ultracell, *"Ultracell UCG-1812"* [en línea], http://www.ultracell.co.uk/UCG%2018-12.pdf [consulta: 11 de marzo del 2009]

Apéndice A

Modelación del sistema

El sistema completo se modela como una barra montada sobre un eje que une 2 ruedas desde sus centros. Se considerarán los mismos parámetros que en la Tabla 2.1, pero además, se agrega la definición de las restricciones que aparecen en el desarrollo de ecuaciones (Tabla A.1).

Nombre	Tipo	Significado
H_{izq}, H_{der}	Restricción	Fuerzas horizontales de contacto entre el chasis y las ruedas
V_{izq}, V_{der}	Restricción	Fuerzas verticales de contacto entre el chasis y las ruedas
F_{Rizq}, F_{Rder}	Restricción	Fuerzas de roce entre el suelo y las ruedas
N_{izq}, N_{der}	Restricción	Fuerzas normales de contacto entre el suelo y las ruedas

Tabla A.1: Restricciones del modelo simplificado

Se considerará además que las ruedas no resbalan. Así, el modelo se obtiene en base a los diagramas de cuerpo libre de la Figura A.1.

Las ecuaciones correspondientes a las ruedas izquierda y derecha son, respectivamente, A.1 y A.2.

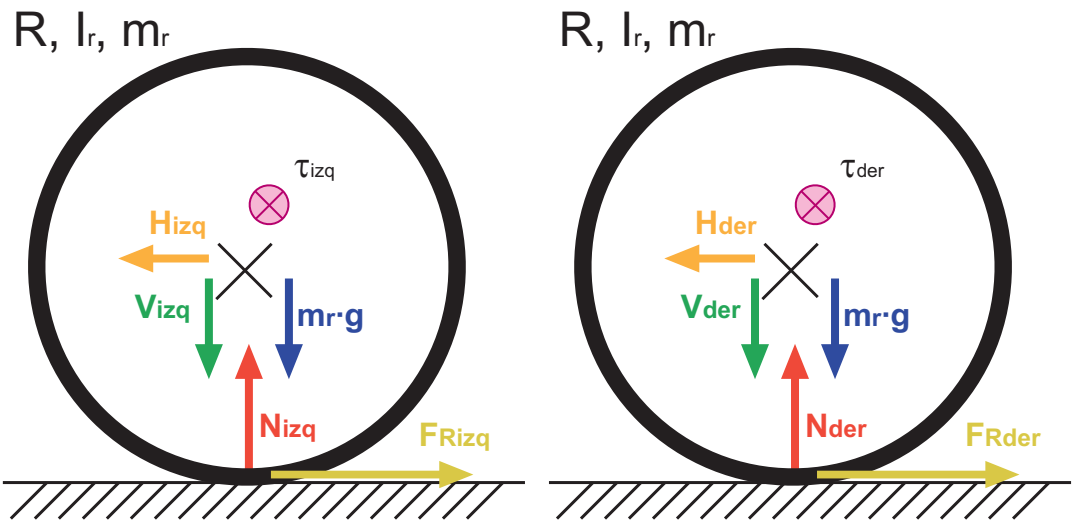


Figura A.1: Diagramas de cuerpo libre para las ruedas

$$\begin{aligned} m_r \cdot \ddot{x_{izq}} &= -H_{izq} + F_{Rizq} \\ m_r \cdot \ddot{y_{izq}} &= N_{izq} - V_{izq} - m_r \cdot g = 0 \\ I_r \cdot \frac{\ddot{x_{izq}}}{R} &= \tau_{izq} - R \cdot F_{Rizq} \end{aligned} \tag{A.1}$$

$$\begin{aligned} m_r \cdot \ddot{x_{der}} &= -H_{der} + F_{Rder} \\ m_r \cdot \ddot{y_{der}} &= N_{der} - V_{der} - m_r \cdot g = 0 \\ I_r \cdot \frac{\ddot{x_{der}}}{R} &= \tau_{der} - R \cdot F_{Rder} \end{aligned} \tag{A.2}$$

Sin embargo, dado que el sistema de referencia está fijo al centro del vehículo, se tiene que $x_{izq} = x_{der} = x$, $y_{izq} = y_{der} = y$. Así, de ahora en adelante se quitará la referencia a la rueda correspondiente.

Para el análisis del DCL del chasis, se consideran además de la Figura A.1, las Figuras 2.1 y 2.2. Las ecuaciones correspondientes son las siguientes:

$$\begin{aligned}
m_b \cdot \ddot{x_b} &= H_{izq} + H_{der} \\
m_b \cdot \ddot{y_b} &= V_{izq} + V_{der} - m_b \cdot g \\
I_{b\theta} \cdot \ddot{\theta} &= L \cdot (V_{izq} + V_{der}) \cdot sin(\theta) - L \cdot (H_{izq} + H_{der}) \cdot cos(\theta) - \tau_{izq} - \tau_{der} \\
I_{b\delta} \cdot \ddot{\delta} &= (H_{izq} - H_{der}) \cdot D
\end{aligned} \tag{A.3}$$

donde las dos últimas ecuaciones describen los movimientos angulares involucrados, ya sea la inclinación del vehículo o el giro de éste. Además, es necesario considerar la relación entre la posición del eje (x,y) y del centro de masas del chasis (x_b,y_b), como se muestra en las siguientes ecuaciones:

$$\begin{aligned}
x_b &= L \cdot sin(\theta) + x & y_b &= L \cdot cos(\theta) \\
\dot{x_b} &= L \cdot \dot{\theta} \cdot cos(\theta) + \dot{x} & \dot{y_b} &= -L \cdot \dot{\theta} \cdot sin(\theta) \\
\ddot{x_b} &= -L \cdot \dot{\theta}^2 \cdot sin(\theta) + L \cdot \ddot{\theta} \cdot cos(\theta) + \ddot{x} & \ddot{y_b} &= -L \cdot \dot{\theta}^2 \cdot cos(\theta) - L \cdot \ddot{\theta} \cdot sin(\theta)
\end{aligned} \tag{A.4}$$

Dadas estas consideraciones, las ecuaciones del chasis A.3 quedan de la siguiente forma:

$$
\begin{aligned}
& m_b \cdot \ddot{x} - m_b \cdot L \cdot \dot{\theta}^2 \cdot sin(\theta) + m_b \cdot L \cdot \ddot{theta} \cdot cos(\theta) = H_{izq} + H_{der} \\
& - m_b \cdot L \cdot \dot{\theta}^2 \cdot cos(\theta) - m_b \cdot L \cdot \ddot{theta} \cdot sin(\theta) = V_{izq} + V_{der} - m_b \cdot g \\
& I_{b\theta} \cdot \ddot{\theta} = L \cdot (V_{izq} + V_{der}) \cdot sin(\theta) - L \cdot (H_{izq} + H_{der}) \cdot cos(\theta) - \tau_{izq} - \tau_{d}er \\
& I_{b\delta} \cdot \ddot{\delta} = (H_{izq} - H_{der}) \cdot D
\end{aligned}
\tag{A.5}
$$

Combinando adecuadamente estas 10 ecuaciones (A.1, A.2 y A.5) se obtienen las siguientes 4 ecuaciones dinámicas del sistema:

$$
\begin{aligned}
I_{b\theta} \cdot \ddot{\theta} - L \cdot m_b \cdot g \cdot sin(\theta) + L \cdot m_b \cdot \ddot{x} \cdot cos(\theta) + m_b \cdot L^2 \cdot \ddot{\theta} &= -\tau_{izq} - \tau_{der} \\
(m_n + 2m_r + 2\frac{I_r}{R^2}) \cdot \ddot{x} - m_b \cdot L \cdot \dot{\theta}^2 \cdot sin(\theta) + m_b \cdot L \cdot \ddot{\theta} \cdot cos(\theta) &= \frac{\tau_{izq} + \tau_{der}}{R} \\
I_{b\delta} \cdot \ddot{\delta} &= D \cdot \frac{\tau_{izq} - \tau_{der}}{R}
\end{aligned}
\tag{A.6}
$$

Linealizando el sistema en torno 0^o, se obtiene lo siguiente:

$$
\begin{aligned}
(I_{b\theta} + m_b \cdot L^2) \cdot \ddot{\theta} - L \cdot m_b \cdot g \cdot \theta + L \cdot m_b \cdot \ddot{x} &= -\tau_{izq} - \tau_{der} \\
(m_b + 2m_r + 2\frac{I_r}{R^2}) \cdot \ddot{x} + m_b \cdot L \cdot \ddot{\theta} &= \frac{\tau_{izq} + \tau_{der}}{R} \\
I_{b\delta} \cdot \ddot{\delta} &= D \cdot \frac{\tau_{izq} - \tau_{der}}{R}
\end{aligned}
\tag{A.7}
$$

Finalmente, combinando adecuadamente las ecuaciones A.7, se obtiene:

$$\begin{aligned}
\ddot{\theta} &= a \cdot \theta + b \cdot (\tau_{izq} + \tau_{der}) \\
\ddot{x} &= c \cdot \theta + d \cdot (\tau_{izq} + \tau_{der}) \\
\ddot{\delta} &= e \cdot (\tau_{izq} - \tau_{der})
\end{aligned} \tag{A.8}$$

donde los valores de las constantes están dados por:

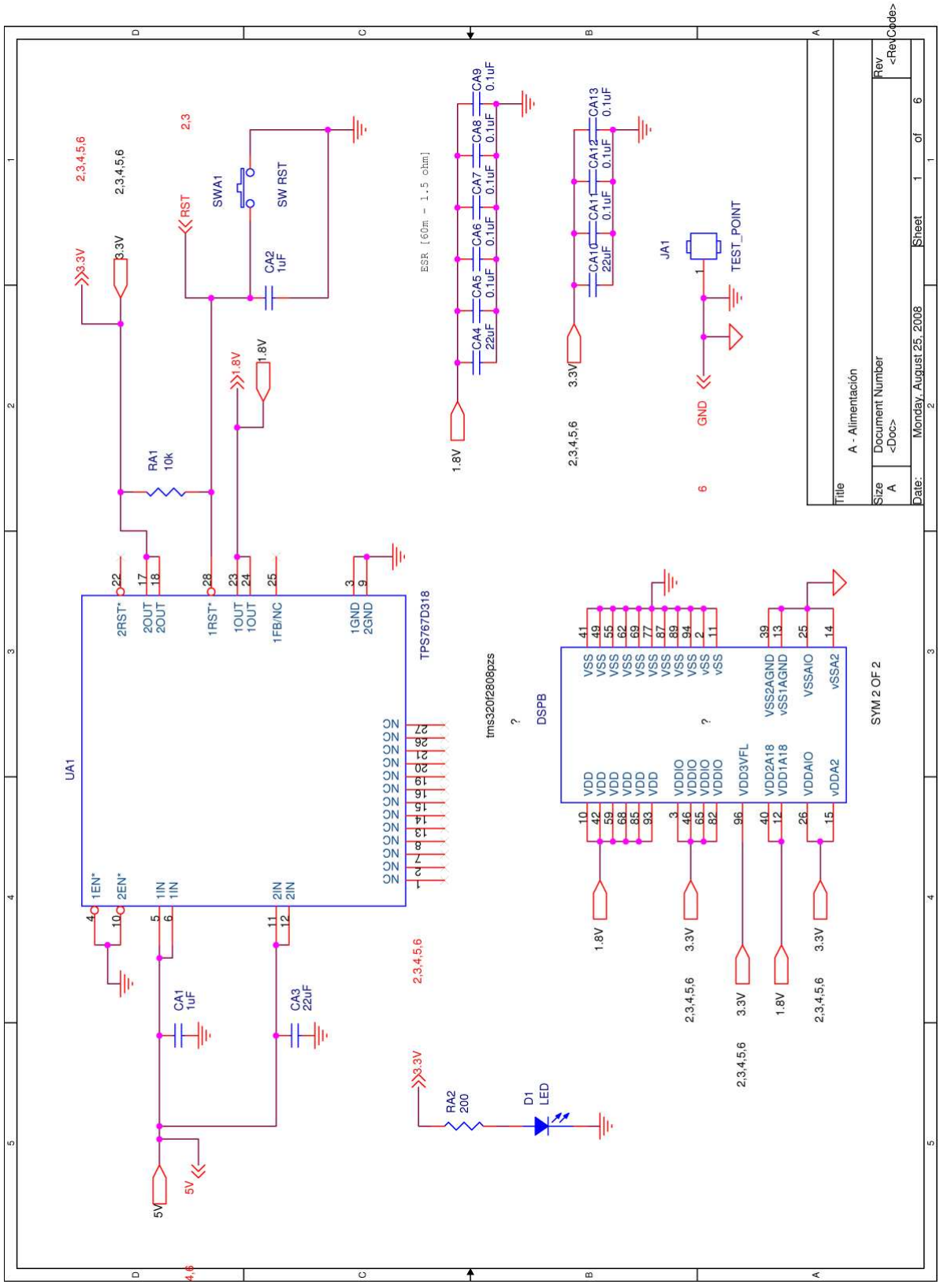
$$\begin{aligned}
a &= \frac{(m_b + 2m_r + 2\frac{I_r}{R^2}) \cdot L \cdot m_b \cdot g}{(m_b + 2m_r + 2\frac{I_r}{R^2}) \cdot (I_{b\theta} + m_b \cdot L^2) - (L \cdot m_b)^2} \\
b &= \frac{(m_b + 2m_r + 2\frac{I_r}{R^2}) + fracL \cdot m_b R}{(m_b + 2m_r + 2\frac{I_r}{R^2}) \cdot (I_{b\theta} + m_b \cdot L^2) - (L \cdot m_b)^2} \\
c &= \frac{-(L \cdot m_b)^2 \cdot g}{(m_b + 2m_r + 2\frac{I_r}{R^2}) \cdot (I_{b\theta} + m_b \cdot L^2) - (L \cdot m_b)^2} \\
d &= \frac{L \cdot m_b + \frac{(m_b + 2m_r + 2\frac{I_r}{R^2})}{R}}{(m_b + 2m_r + 2\frac{I_r}{R^2}) \cdot (I_{b\theta} + m_b \cdot L^2) - (L \cdot m_b)^2} \\
e &= \frac{D}{R \cdot I_{b\delta}}
\end{aligned}$$

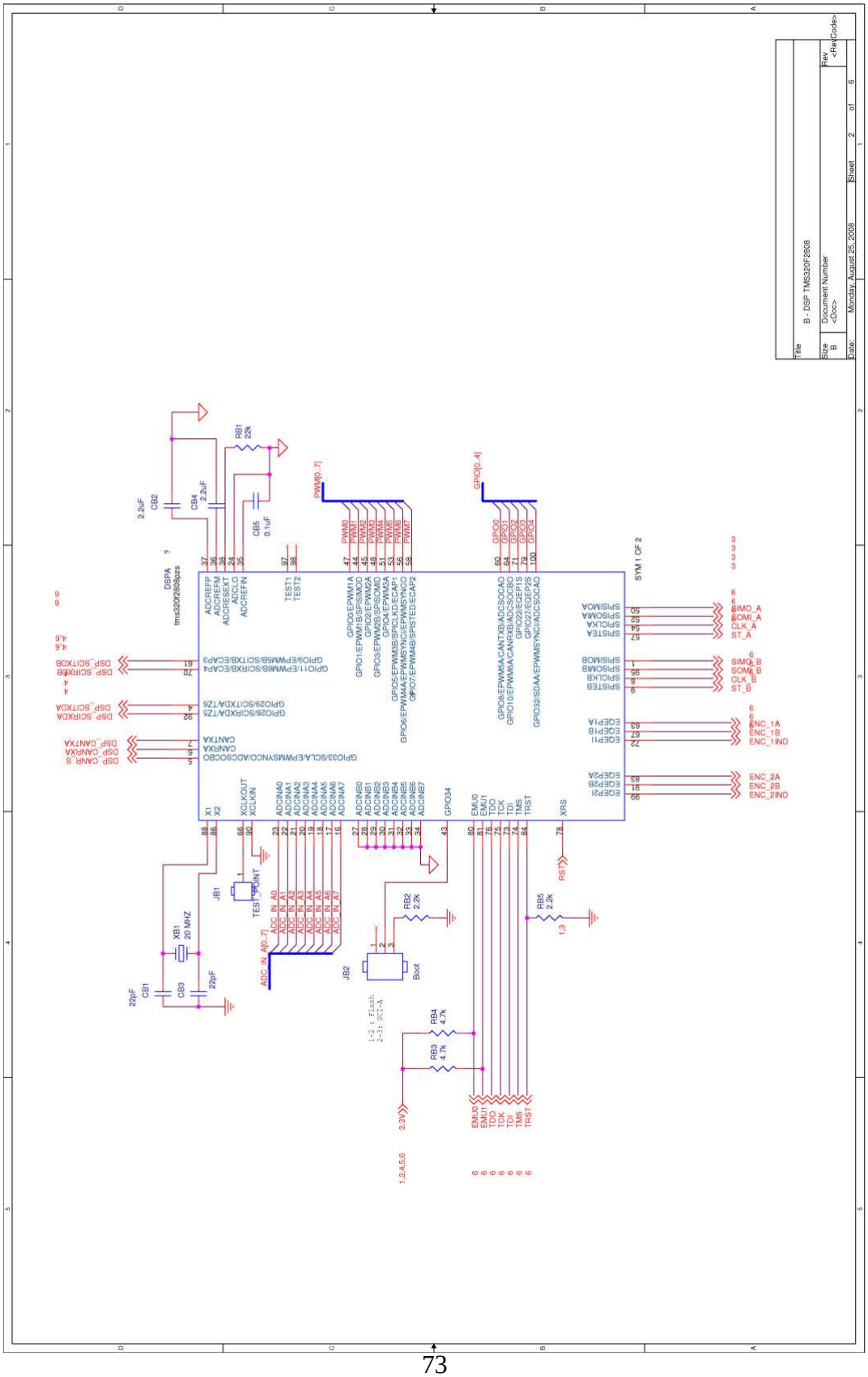
Apéndice B

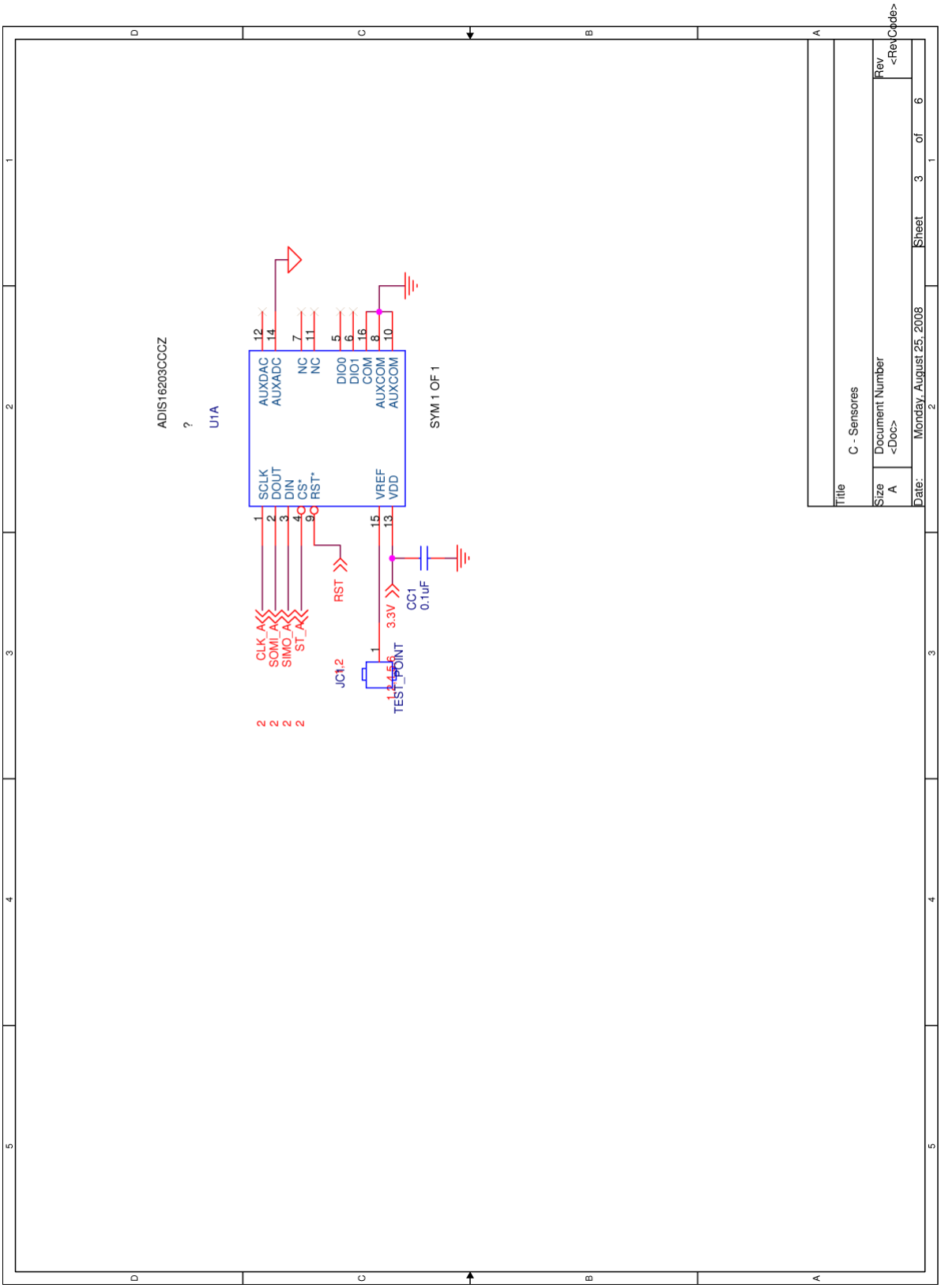
Esquemáticos

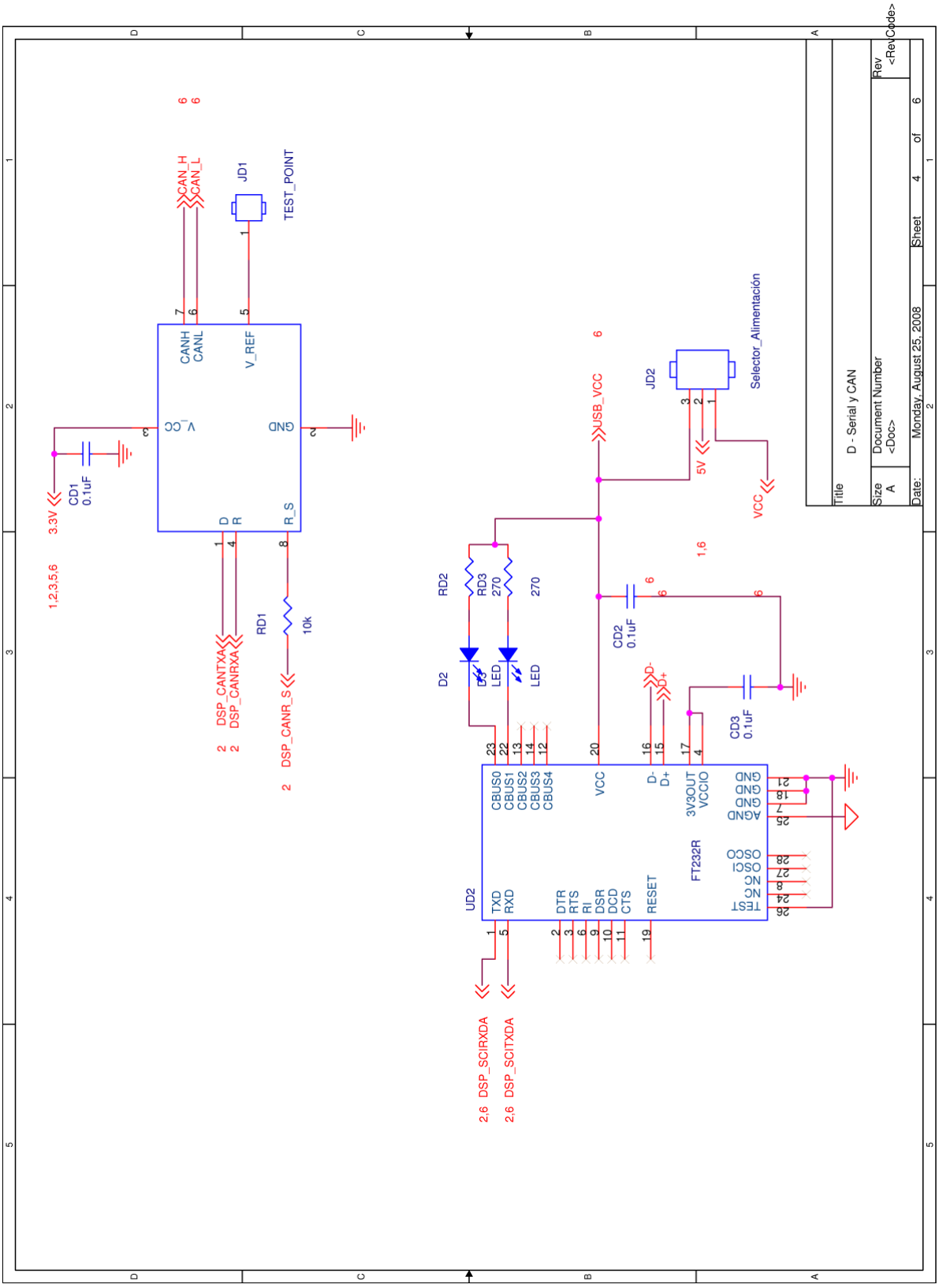
En este apéndice se incluyen los esquemáticos de las siguientes PCB:

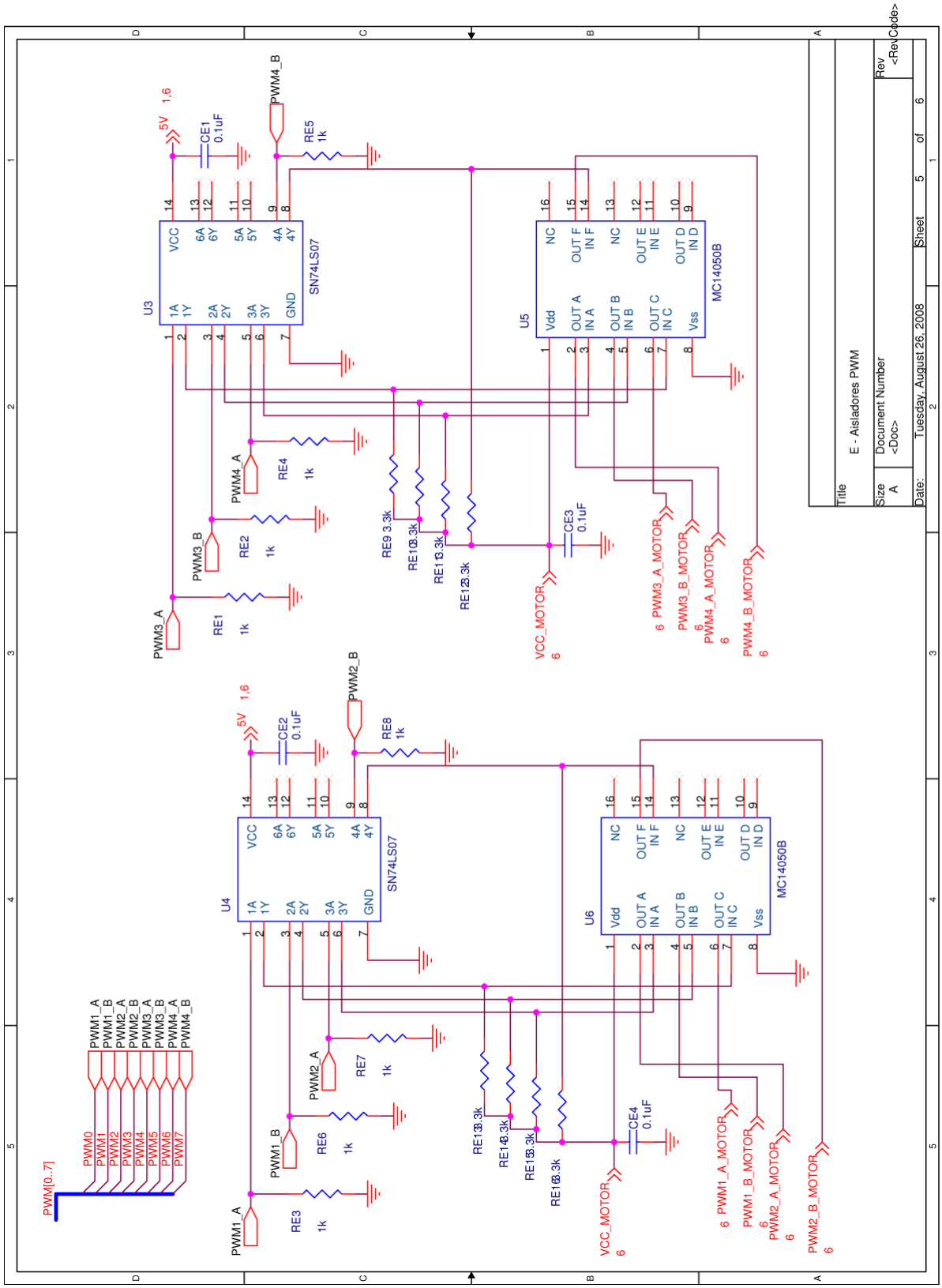
- Placa de control y sensores
- Puente H
- Fuente switching
- Placa de interfaz de usuario.

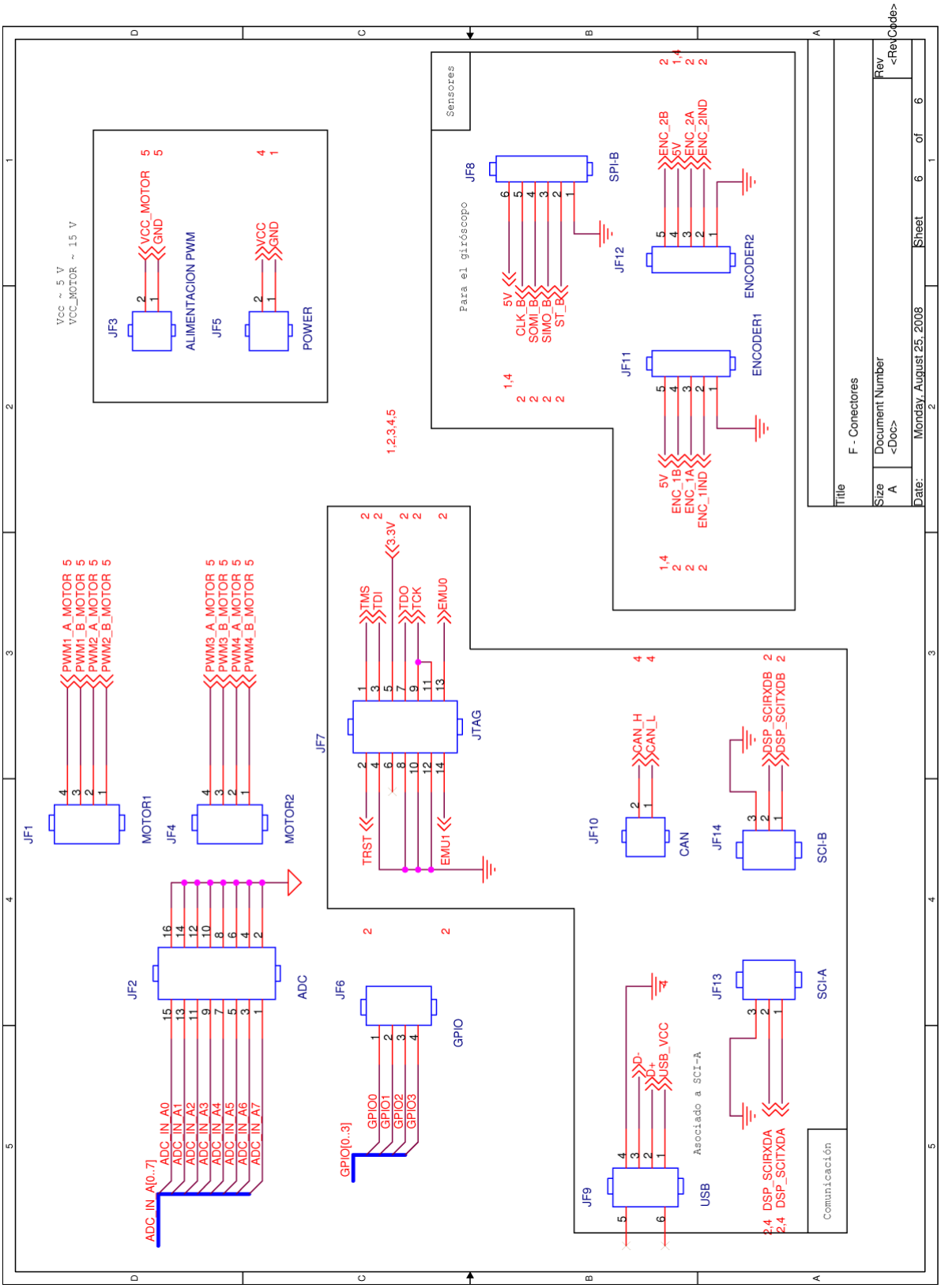

UA1
TPS767D318
1EN*
2EN*
1IN
1IN
2IN
2IN
2RST*
2OUT
2OUT
1RST*
1OUT
1OUT
1FB/NC
1GND
2GND
CA1
1uF
CA3
22uF
RA1
10k
SWA1
SW RST
CA2
1uF
RST
3.3V
1.8V
5V
ESR [60m - 1.5 ohm]
CA4
22uF
CA5
0.1uF
CA6
0.1uF
CA7
0.1uF
CA8
0.1uF
CA9
0.1uF
CA10
22uF
CA11
0.1uF
CA12
0.1uF
CA13
0.1uF
JA1
TEST_POINT
GND
tms320f2808pzs
DSPB
SYM 2 OF 2
VDD
VSS
VDDIO
VDD3VFL
VSS2AGND
VSS1AGND
VDD2A18
VDD1A18
VSSAIO
VDDAIO
vSSA2
vDDA2
RA2
200
D1
LED
Title
A - Alimentación
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Monday, August 25, 2008
Sheet
1
of
6

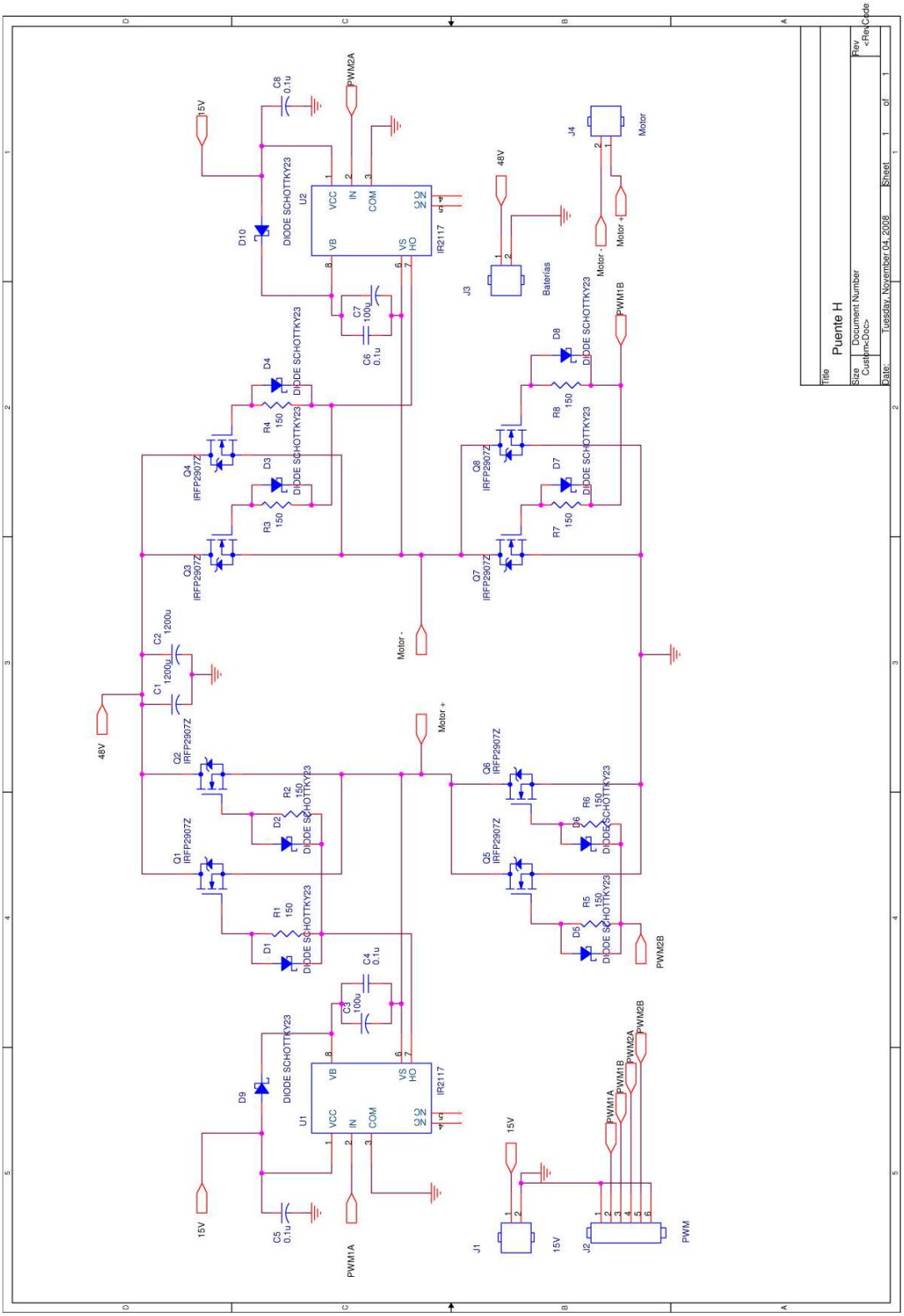

Title
B - DSP TMS320F2808
Size
B
Document Number
<Doc>
Rev
<RevCode>
Date:
Monday, August 25, 2008
Sheet
2
of
6
DSPA
tms320f2808pzs
SYM 1 OF 2
PWM[0..7]
GPIO[0..4]
RB1
22k
CB2
2.2uF
CB4
2.2uF
CB5
0.1uF
RB2
2.2k
RB3
4.7K
RB4
4.7K
RB5
2.2k
XB1
20 MHZ
CB1
22pF
CB3
22pF
JB1
TEST_POINT
JB2
Boot
RST
3.3V
SIMO_A
SOMI_A
CLK_A
ST_A
SIMO_B
SOMI_B
CLK_B
ST_B
ENC_1A
ENC_1B
ENC_1IND
ENC_2A
ENC_2B
ENC_2IND
EMU0
EMU1
TDO
TCK
TDI
TMS
TRST

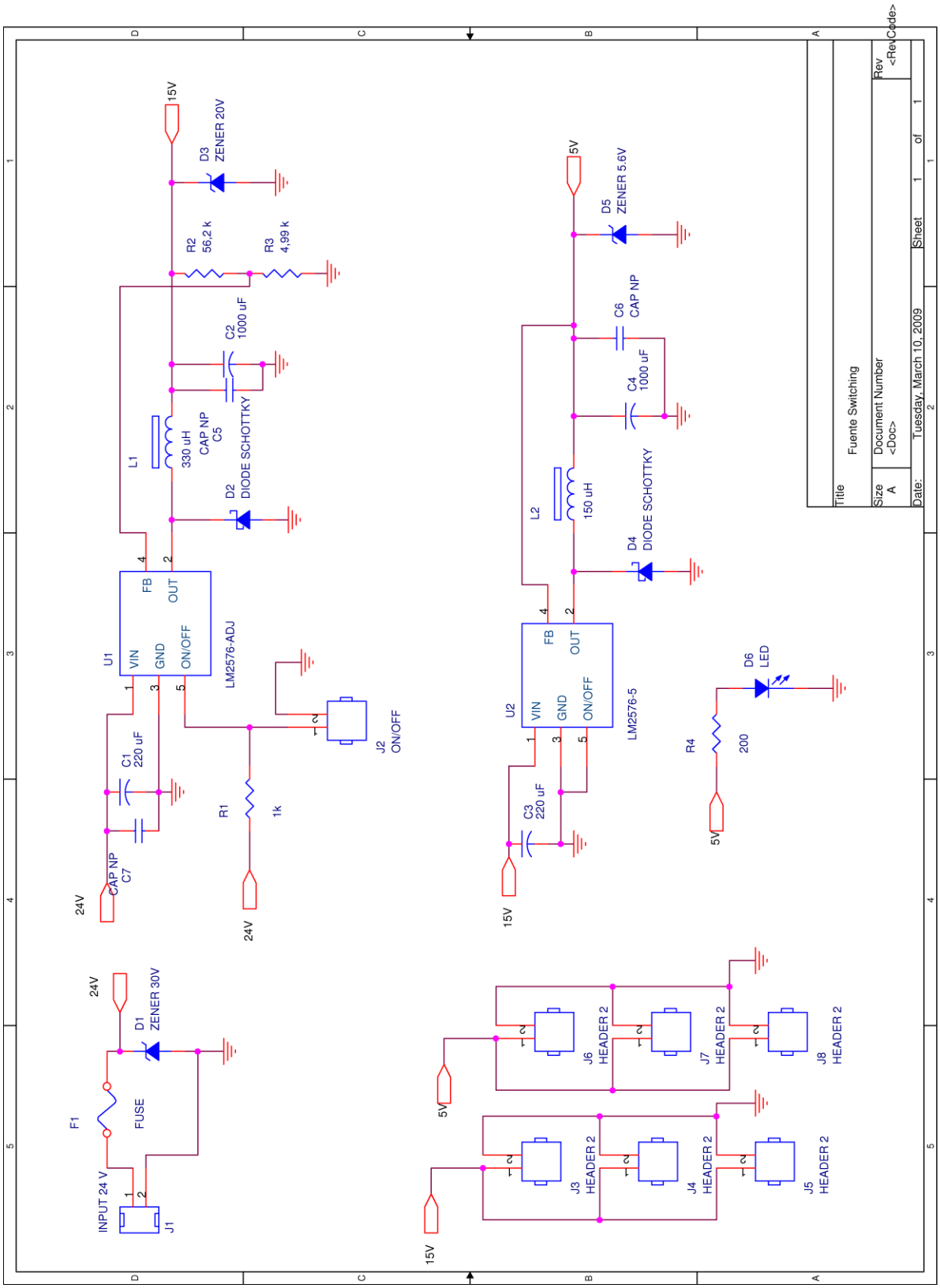

ADIS16203CCCZ
?
U1A
AUXDAC
AUXADC
NC
NC
DIO0
DIO1
COM
AUXCOM
AUXCOM
SCLK
DOUT
DIN
CS*
RST*
VREF
VDD
SYM 1 OF 1
CLK_A
SOMI_A
SIMO_A
ST_A
RST
3.3V
CC1
0.1uF
JC1.2
TEST_POINT
Title
C - Sensores
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Monday, August 25, 2008
Sheet
3
of
6

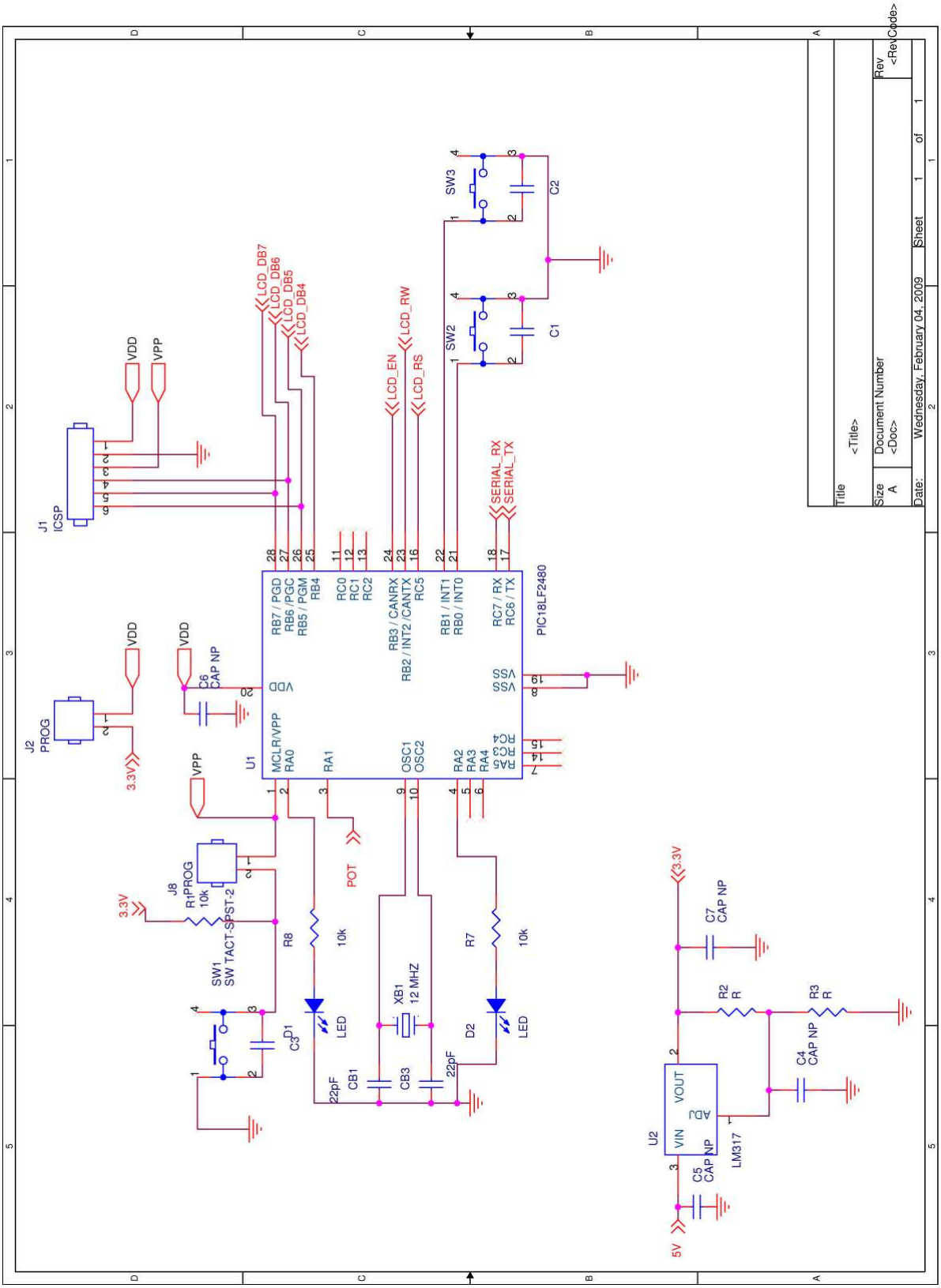

Title
D - Serial y CAN
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Monday, August 25, 2008
Sheet
4
of
6
CAN_H
CAN_L
JD1
TEST_POINT
CANH
CANL
V_REF
V_CC
GND
D
R
R_S
CD1
0.1uF
3.3V
RD1
10k
DSP_CANTXA
DSP_CANRXA
DSP_CANR_S
RD2
RD3
270
D2
LED
USB_VCC
JD2
Selector_Alimentación
5V
VCC
CD2
0.1uF
CD3
0.1uF
UD2
FT232R
TXD
RXD
RESET
DTR
RTS
RI
DSR
DCD
CTS
CBUS0
CBUS1
CBUS2
CBUS3
CBUS4
3V3OUT
VCCIO
AGND
GND
TEST
OSCO
OSCI
NC
DSP_SCIRXDA
DSP_SCITXDA

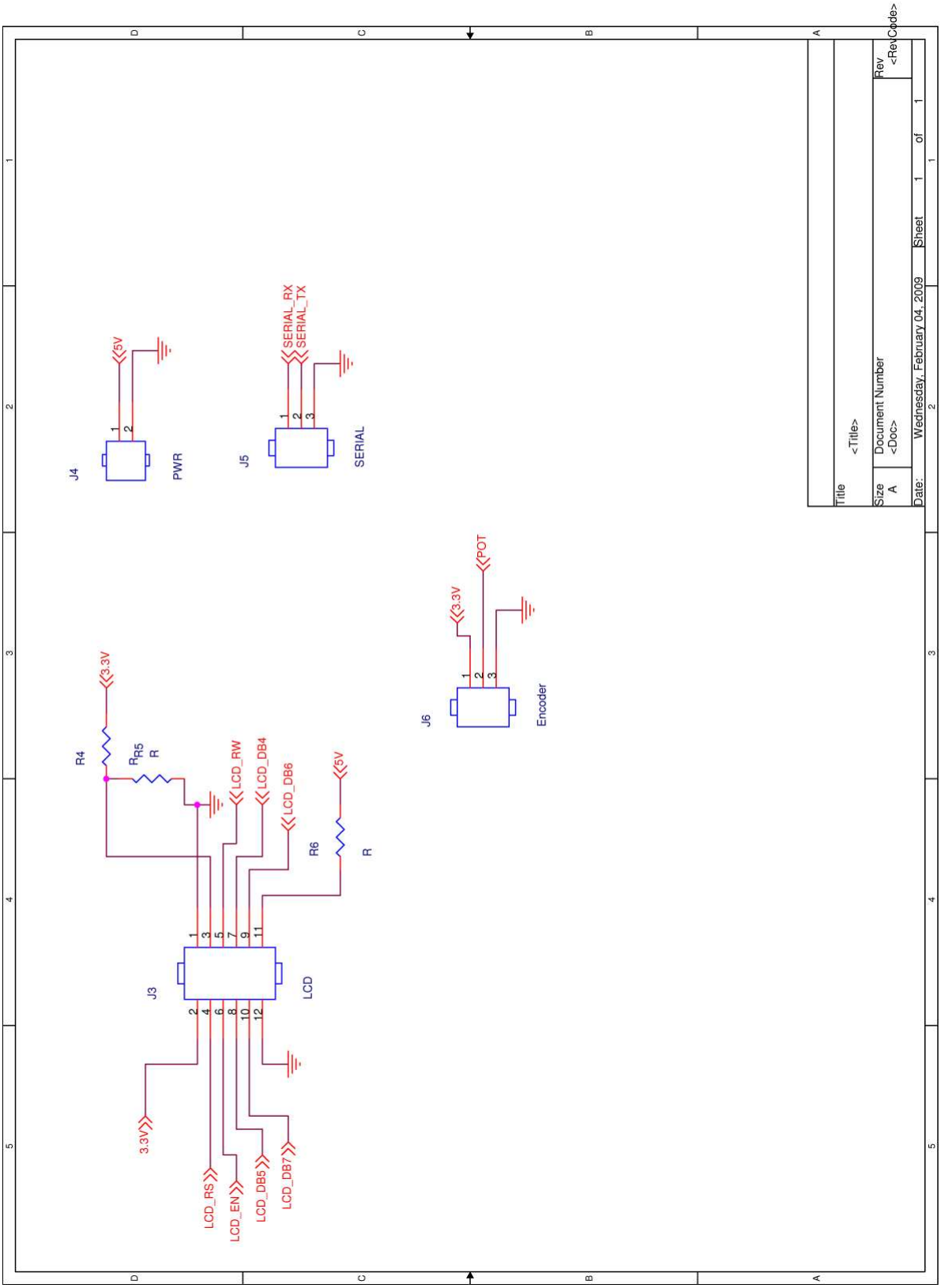

U3
SN74LS07
U4
SN74LS07
U5
MC14050B
U6
MC14050B
5V
CE1 0.1uF
CE2 0.1uF
CE3 0.1uF
CE4 0.1uF
RE1 1k
RE2 1k
RE3 1k
RE4 1k
RE5 1k
RE6 1k
RE7 1k
RE8 1k
RE9 3.3k
RE10 3.3k
RE11 3.3k
RE12 3.3k
RE13 3.3k
RE14 3.3k
RE15 3.3k
RE16 3.3k
PWM1_A
PWM1_B
PWM2_A
PWM2_B
PWM3_A
PWM3_B
PWM4_A
PWM4_B
PWM[0..7]
PWM0
PWM1
PWM2
PWM3
PWM4
PWM5
PWM6
PWM7
VCC_MOTOR
PWM1_A_MOTOR
PWM1_B_MOTOR
PWM2_A_MOTOR
PWM2_B_MOTOR
PWM3_A_MOTOR
PWM3_B_MOTOR
PWM4_A_MOTOR
PWM4_B_MOTOR
Title
E - Aisladores PWM
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Tuesday, August 26, 2008
Sheet
5
of
6


Vcc ~ 5 V
VCC_MOTOR ~ 15 V
JF3
VCC_MOTOR
GND
ALIMENTACION PWM
JF5
VCC
GND
POWER
Sensores
Para el giróscopo
JF8
SPI-B
JF12
ENC_2B
5V
ENC_2A
ENC_2IND
ENCODER2
JF11
ENCODER1
5V
ENC_1B
ENC_1A
ENC_1IND
1,2,3,4,5
JF1
MOTOR1
PWM1_A_MOTOR
PWM1_B_MOTOR
PWM2_A_MOTOR
PWM2_B_MOTOR
JF4
MOTOR2
PWM3_A_MOTOR
PWM3_B_MOTOR
PWM4_A_MOTOR
PWM4_B_MOTOR
JF7
JTAG
TMS
TDI
3.3V
TDO
TCK
EMU0
TRST
EMU1
JF10
CAN
CAN_H
CAN_L
JF14
SCI-B
DSP_SCIRXDB
DSP_SCITXDB
JF2
ADC
ADC_IN_A0
ADC_IN_A1
ADC_IN_A2
ADC_IN_A3
ADC_IN_A4
ADC_IN_A5
ADC_IN_A6
ADC_IN_A7
ADC_IN_A[0..7]
JF6
GPIO
GPIO0
GPIO1
GPIO2
GPIO3
GPIO[0..3]
JF9
USB
D-
D+
USB_VCC
Asociado a SCI-A
JF13
SCI-A
DSP_SCIRXDA
DSP_SCITXDA
Comunicación
Title
F - Conectores
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Monday, August 25, 2008
Sheet
6
of
6


Puente H
Document Number
<Doc>
Rev
<RevCode>
Tuesday, November 04, 2008
Sheet 1 of 1
Size
Custom
Date:
Title
U1
U2
IR2117
DIODE SCHOTTKY23
IRFP2907Z
Motor +
Motor -
Baterias
PWM
PWM1A
PWM1B
PWM2A
PWM2B
48V
15V


15V
D3
ZENER 20V
R2
56,2 k
R3
4,99 k
C2
1000 uF
L1
330 uH
CAP NP
C5
D2
DIODE SCHOTTKY
U1
VIN
GND
ON/OFF
FB
OUT
LM2576-ADJ
C1
220 uF
CAP NP
C7
24V
R1
1k
J2
ON/OFF
24V
5V
D5
ZENER 5.6V
C6
CAP NP
C4
1000 uF
L2
150 uH
D4
DIODE SCHOTTKY
U2
LM2576-5
C3
220 uF
15V
D6
LED
R4
200
5V
24V
D1
ZENER 30V
F1
FUSE
INPUT 24 V
J1
5V
J6
HEADER 2
J7
HEADER 2
J8
HEADER 2
J3
HEADER 2
J4
HEADER 2
J5
HEADER 2
15V
Title
Fuente Switching
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Tuesday, March 10, 2009
Sheet
1
of
1


Title
<Title>
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Wednesday, February 04, 2009
Sheet
1
of
1
J1
ICSP
J2
PROG
J8
PROG
SW1
SW TACT-SPST-2
SW2
SW3
U1
PIC18LF2480
MCLR/VPP
RA0
RA1
RA2
RA3
RA4
RB7 / PGD
RB6 / PGC
RB5 / PGM
RB4
RC0
RC1
RC2
RB3 / CANRX
RB2 / INT2 / CANTX
RC5
RB1 / INT1
RB0 / INT0
RC7 / RX
RC6 / TX
OSC1
OSC2
VDD
VSS
U2
LM317
VIN
VOUT
ADJ
XB1
12 MHZ
D1
D2
LED
R1
10k
R2
R
R3
R
R7
10k
R8
10k
POT
C1
C2
C3
C4
CAP NP
C5
CAP NP
C6
CAP NP
C7
CAP NP
CB1
22pF
CB3
22pF
LCD_DB7
LCD_DB6
LCD_DB5
LCD_DB4
LCD_EN
LCD_RW
LCD_RS
SERIAL_RX
SERIAL_TX
VDD
VPP
3.3V
5V

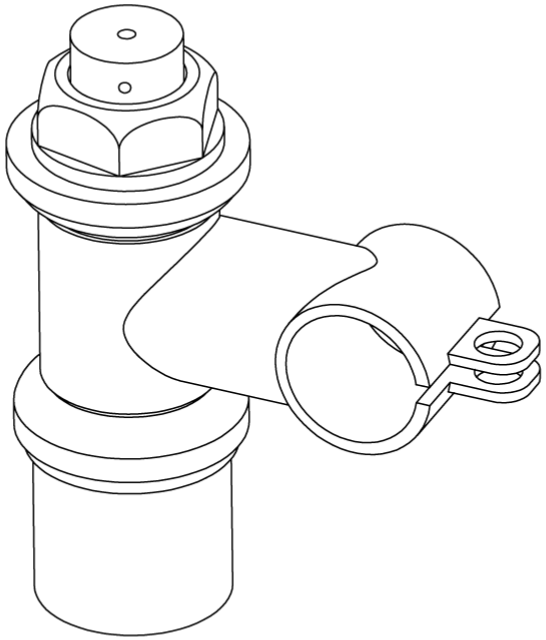

J4
PWR
5V
J5
SERIAL
SERIAL_RX
SERIAL_TX
J6
Encoder
POT
3.3V
J3
LCD
R4
R5
R
R6
R
LCD_RW
LCD_DB4
LCD_DB6
LCD_RS
LCD_EN
LCD_DB5
LCD_DB7
Title
<Title>
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Wednesday, February 04, 2009
Sheet
1
of
1

Apéndice C

Planos Mecánicos

En este apéndice se presentan los planos mecánicos de las principales piezas que fueron fabricadas en el desarrollo del proyecto.

REVISION HISTORY			
REV.	DESCRIPTION	DATE	APPROVED



	NAME	DATE
DRAWN	Administrador	06/18/09
CHECKED		
ENG APPR.		
MGR. APPR.		

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
ANGLES ±X.X°
2 PL ±.XXX 3 PL ±.XXXX

SOLID EDGE
UGS - The PLM Company

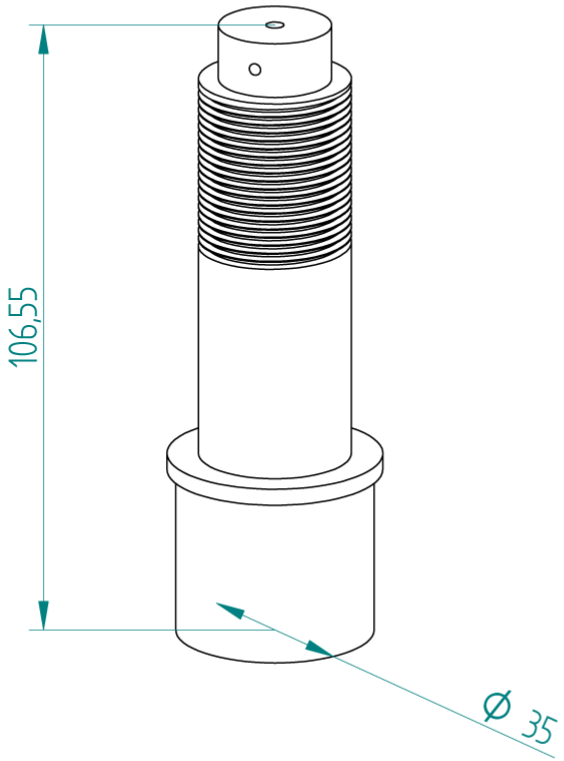
TITLE: JUEGO DE DIRECCION

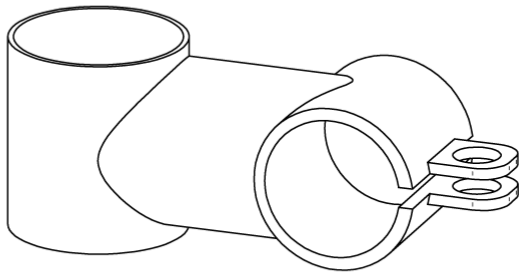
SIZE: A2 | DWG NO | REV

FILE NAME: PLANO FINAL.dft

SCALE: | WEIGHT: | SHEET 1 OF 7

REVISION HISTORY			
REV.	DESCRIPTION	DATE	APPROVED





	NAME	DATE
DRAWN	Administrador	06/18/09
CHECKED		
ENG APPR.		
MGR. APPR.		

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
ANGLES ±X.X°
2 PL ±.XXX 3 PL ±.XXXX

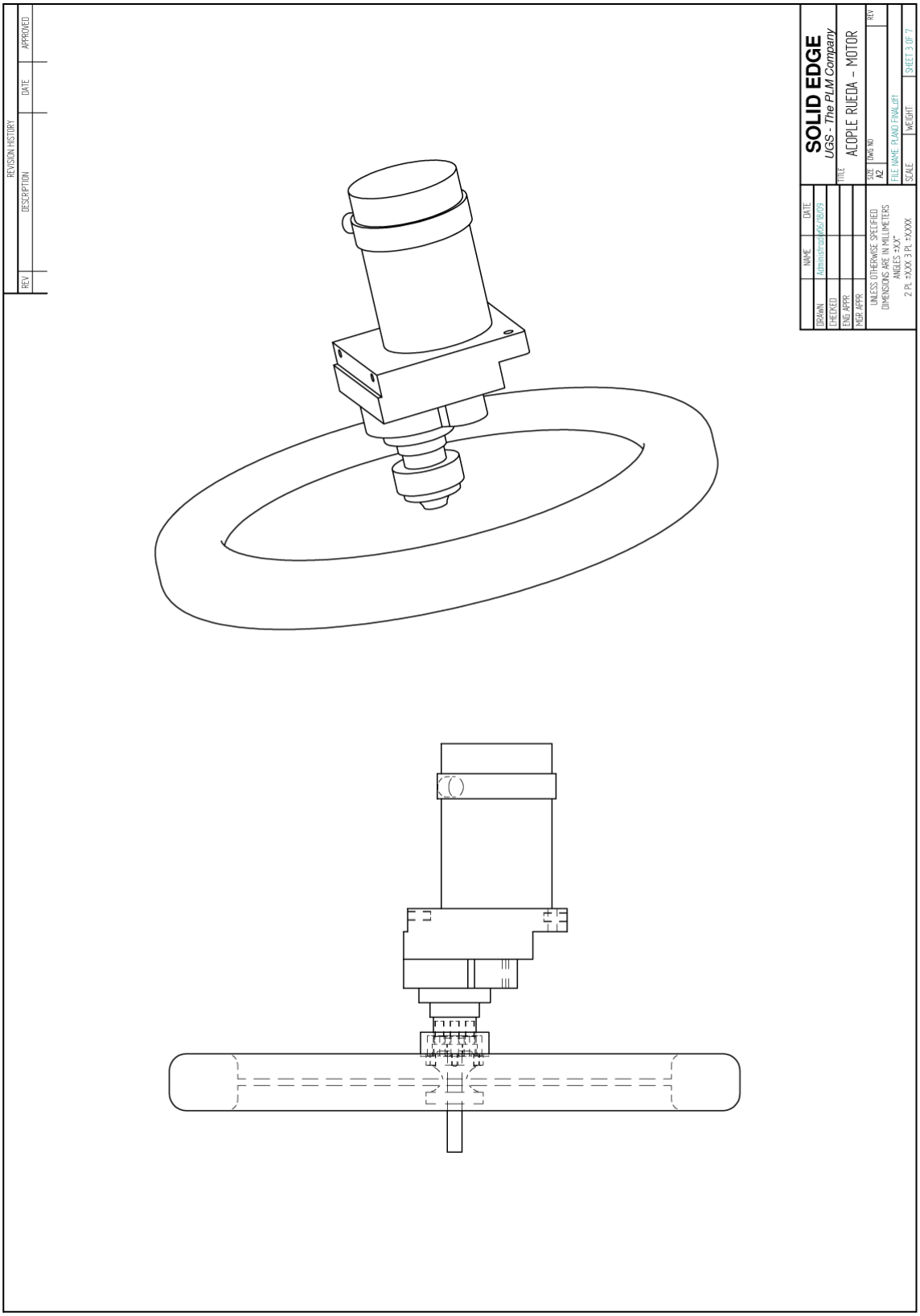
SOLID EDGE
UGS - The PLM Company

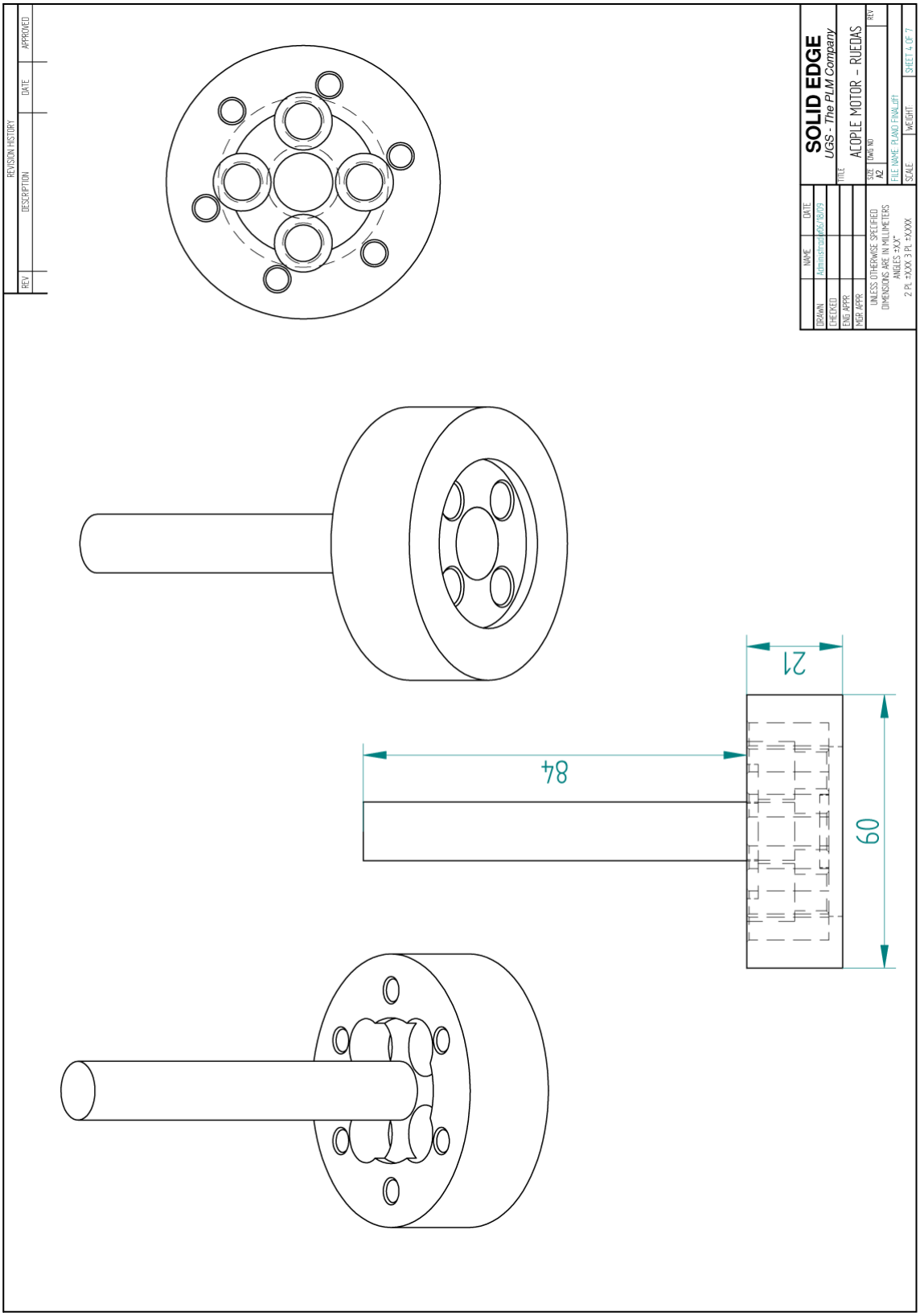
TITLE: JUEGO DE DIRECCION

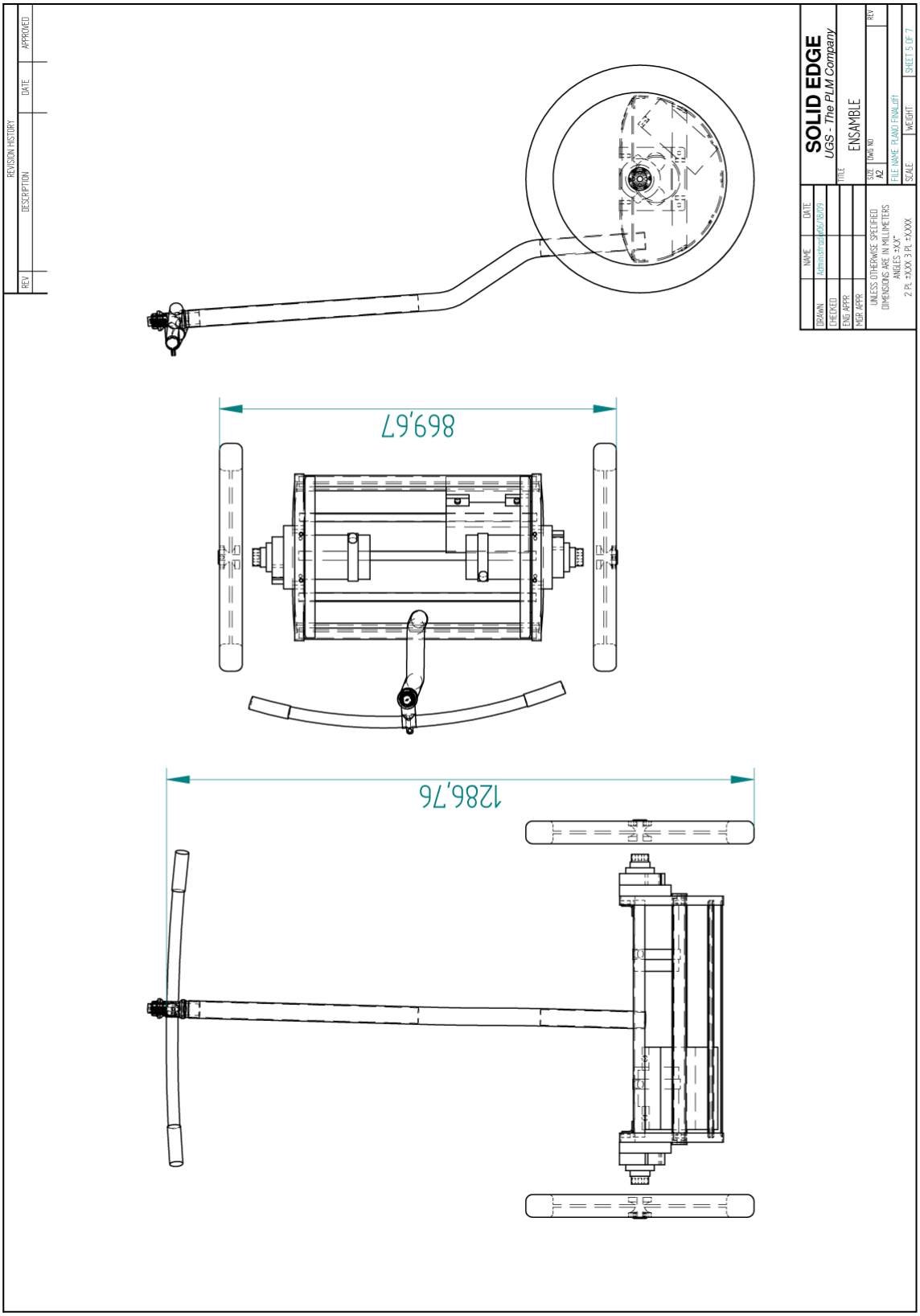
SIZE: A2 | DWG NO | REV

FILE NAME: PLANO FINAL.dft

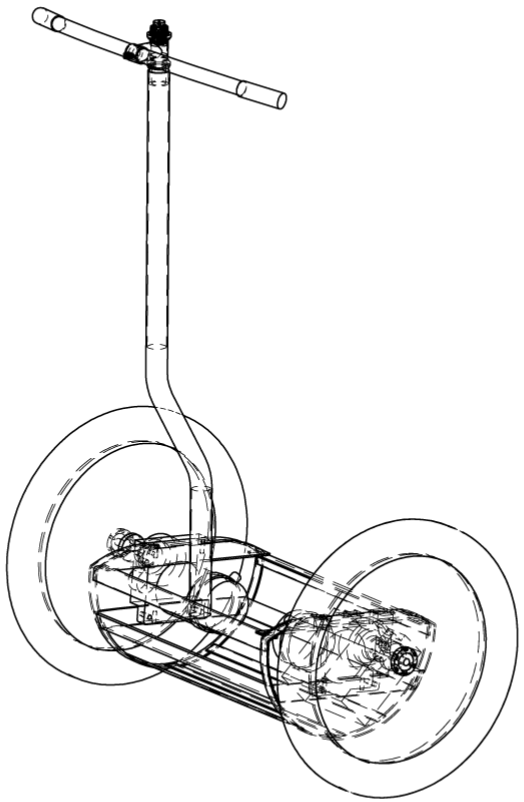
SCALE: | WEIGHT: | SHEET 2 OF 7


REVISION HISTORY
REV
DESCRIPTION
DATE
APPROVED
SOLID EDGE
UGS - The PLM Company
NAME
DATE
DRAWN
Administrador
6/6/2013
CHECKED
ENG APPR.
MGR APPR.
TITLE
ACOPLE RUEDA – MOTOR
SIZE
A2
DWG NO
REV
UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
ANGLES ±X.X°
2 PL ±X.XX 3 PL ±X.XXX
FILE NAME PLANO FINAL.dft
SCALE
WEIGHT
SHEET 3 OF 7


REVISION HISTORY
REV
DESCRIPTION
DATE
APPROVED
SOLID EDGE
UGS - The PLM Company
NAME
DATE
DRAWN
Administrador
06/10/13
CHECKED
ENG APPR.
MGR APPR.
TITLE
ACOPLE MOTOR – RUEDAS
UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
ANGLES ±X.X°
2 PL ±X.XX 3 PL ±X.XXX
SIZE
A2
DWG NO
REV
FILE NAME PLANO FINAL.dft
SCALE
WEIGHT
SHEET 1 OF 7
21
84
60


REVISION HISTORY
REV
DESCRIPTION
DATE
APPROVED
SOLID EDGE
UGS - The PLM Company
NAME
DATE
DRAWN
Administrateur
06/10/2013
CHECKED
ENG APPR.
MGR APPR.
TITLE
ENSAMBLE
SIZE
A2
DWG NO
REV
UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
ANGLES ±X.X°
2 PL ±X.XX 3 PL ±X.XXX
FILE NAME PLANO FINAL.iPT
SCALE
WEIGHT
SHEET 5 OF 7
869,67
1286,76

REVISION HISTORY			
REV.	DESCRIPTION	DATE	APPROVED



	NAME	DATE
DRAWN	Administrator	06/18/09
CHECKED		
ENG APPR.		
MGR. APPR.		

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
ANGLES ±X.X°
2 PL ±.XXX 3 PL ±.XXXX

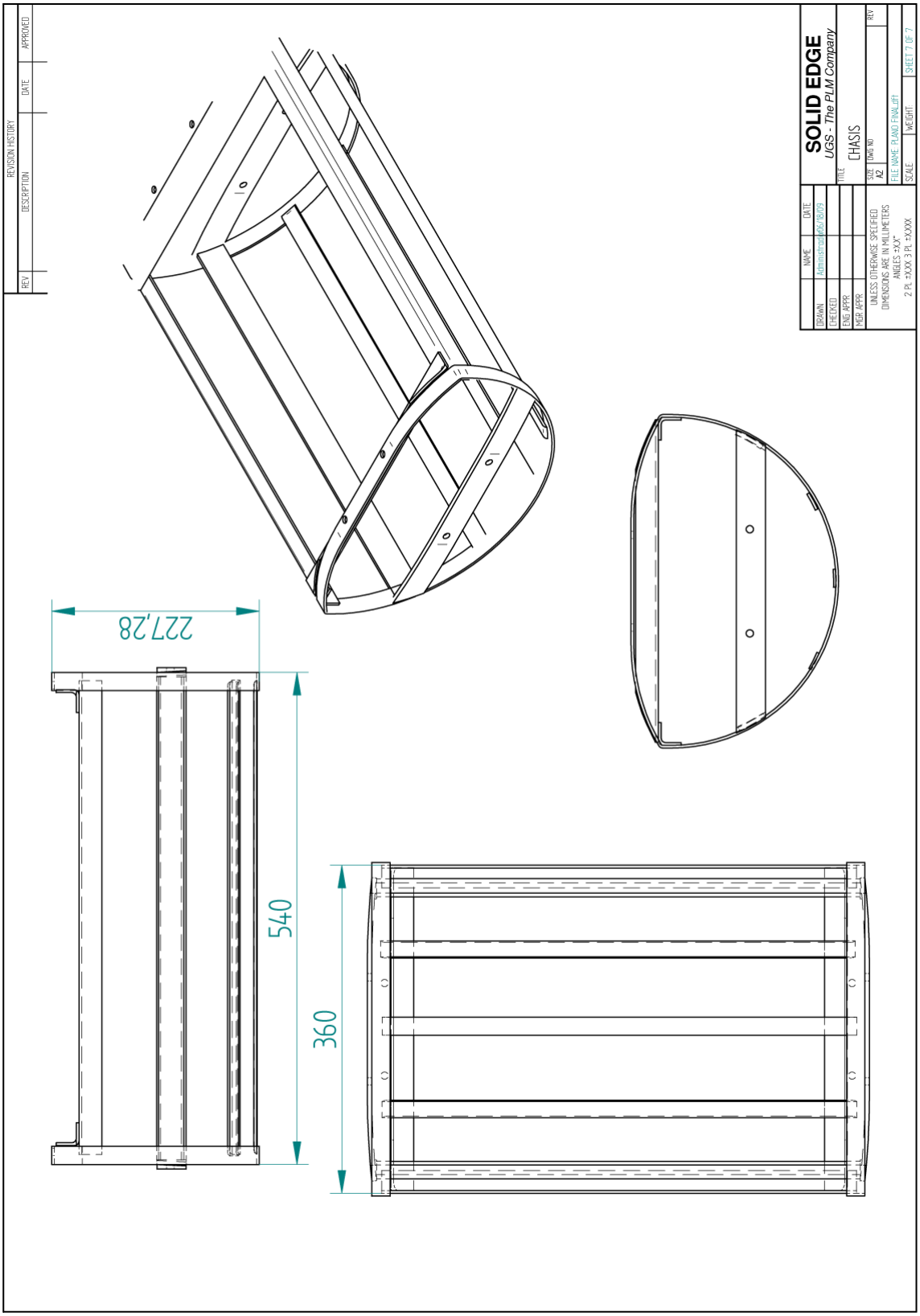
SOLID EDGE
UGS - The PLM Company

TITLE: ENSAMBLE

SIZE: A2 | DWG NO | REV

FILE NAME: PLANO FINAL.dft

SCALE: | WEIGHT: | SHEET 6 OF 7


227,28
540
360
REVISION HISTORY
REV
DESCRIPTION
DATE
APPROVED
SOLID EDGE
UGS - The PLM Company
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MGR APPR.
Administrateur 06/10/2015
UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
ANGLES ±X.X°
2 PL ±X.XX 3 PL ±X.XXX
TITLE
CHASIS
SIZE
A2
DWG NO
REV
FILE NAME PLANO FINAL.dft
SCALE
WEIGHT
SHEET 7 OF 7

Apéndice D

Códigos

Es este apéndice se encuentran los códigos principales que se programaron tanto en la plataforma de DSP como en la placa de interfaz de usuario. Para los códigos completos, refiérase al DVD adjunto.

D.1. Códigos programados en DSP (*main.c*)

```
#include "DSP280x_Device.h"     // Definiciones generales del dispositivo.
#include "DSP280x_Examples.h"       // Definiciones para el DSP
#include "pwm.h"                            // funciones para PWM
#include "spi.h"
#include "sci.h"

/*      Máscaras para datos desde el inclinómetro */
#define MASK14      0x3FFF
#define MASK12      0x0FFF

```

```
/* Registros para peticiones al inclinómetro */
#define SUPPLY_OUT 0x03FF
#define TEMP_OUT 0x0BFF
#define INCL_OUT 0x0DFF
#define INCL_180_OUT 0x0FFF

#define N_MUESTRAS_TILT 11
#define MAX_AUMENTO 40       // Límite de aumento de ciclo de trabajo cada 10 ms, 4%
#define OFFSET 20            // OFFSET para el ángulo, 0.5º
#define CENTRO_VOLANTE 109   // Valor del manillar centrado

#define ADC_MODCLK 0x4
#define CURRENT_MAX 1500             // Equivalente a 40 A

/*Variables relativas a la inclinación*/
int tilt, dtilt;
int tilt_acum[N_MUESTRAS_TILT];
int       dtilt_acum[N_MUESTRAS_TILT];       // Vector de derivadas numéricas
int puntero;
long suma_tilt;       //Almacena la suma de angulos

/* Variables del resto del estado del sistema */
int m_izq;       // Ciclo de trabajo del motor izquierdo (porcentual)
int m_der;       // Ciclo de trabajo del motor derecho (porcentual)
int giro;       // Giro del manubrio, entre -86 y 85
Uint16 bat;             // Estado de la batería, 12 bits (0 y 4095)
int corriente_izq, corriente_der;       // Valores de corriente
int offset_corriente_izq, offset_corriente_der;       //Valores para corriente nula
/* Contadores de tiempo que hay sobrecorriente */
Uint16 cont_sobrecorriente_izq, cont_sobrecorriente_der;

/* Variables asociadas al giróscopo */
Uint16 gyro_sensor;       // Valor leído directamente desde el ADC
Uint16 gyro_zro;       // Valor para Zero Rate Output
int gyro_efectivo;       // Velor efectivo del gyro (gyro_sensor-gyro_zro)

/* Contador de milisegundos */
Uint16 ms;

/* Variable para envío de información por SCI */
```

```
51  char *msg;
52  int i;
53
54  /*Flags*/
55  int flag_timer0;
56  int flag_update_LCD;
57  int flag_rs232;
58  int flag_caida;       // se activa si se supera el ángulo límite
59  int flag_corrientes;
60
61  /* Funciones a utilizar */
62  interrupt void sciaRxFifoIsr(void);              // Asociada a la interrupción serial
63  interrupt void cpu_timer0_isr(void);       // Asociada a la interrupción del timer0
64  int getTilt();                                   // Encargada de la petición por SPI de la inclinación
65  void updateLCD();                                  // Envío del estado y petición de giro
66  void getPWM(int *m_izq, int *m_der, int tilt, int giro, Uint16 bat);
67  void getADC();
68
69  Uint16 getGyro_ZRO(void);
70
71  /* Funciones menores */
72  void calibrar_incl(void);
73  void delay(void);
74
75  void main(void){
76
77      /* == Paso 1. Inicializar Control de sistema: ==*/
78      InitSysCtrl();
79
80      EALLOW;
81      SysCtrlRegs.HISPCP.all = ADC_MODCLK;
82      EDIS;
83
84      /*== Paso 2. Inicializar GPIO ==*/
85
86      // Configura GPIO para ePWM1, ePWM2, ePWM3 y ePWM4
87      InitEPwm1Gpio();       // Motor izquierdo, adelante
88      InitEPwm2Gpio();       // Motor izquierdo, atrás
89      InitEPwm3Gpio();       // Motor derecho, adelante
90      InitEPwm4Gpio();       // Motor derecho, atrás
```

```
91
92          //      Configura GPIO para SPI-A
93          InitSpiaGpio();
94
95          //      Configura GPIO para SCI-A
96          InitSciaGpio();
97
98          /*== Paso 3. Desactivar todas las interrupciones ==*/
99          DINT;
100         InitPieCtrl();
101         IER = 0x0000;
102         IFR = 0x0000;
103         InitPieVectTable();
104
105         EALLOW;
106         PieVectTable.TINT0 = &cpu_timer0_isr;
107         PieVectTable.SCIRXINTA = &sciaRxFifoIsr;
108         EDIS;
109
110         /*==  Paso 4. Inicializar todos los periféricos: ==*/
111         // Inicializa el puerto ADC
112         InitAdc();
113
114         // Inicializa ePWM1, ePWM2, ePWM3, ePWM4
115         init_pwm();
116
117         // Inicializa SPI A
118         init_spi_fifo();
119         init_spi();
120
121         // Inicializa SCI-A
122         init_scia_fifo();
123         init_scia();
124
125         //Inicializa CPU Timers
126         InitCpuTimers();
127
128         // Interrupción c/10 ms
129         ConfigCpuTimer(&CpuTimer0, 100, 10000);       //10 ms
130
```

```
131         // Se copian funciones a RAM.
132         MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd, &RamfuncsRunStart);
133
134         /*==Paso 5. Código específico, habilitar interrupciones.==*/
135
136         EALLOW;
137       // Habilitar CPU INT1 que está conectada a CPU-Timer 0
138         // Habilitar CPU INT9 que está conectada a interrupciones de SCI-A.
139         IER |= M_INT1 | M_INT9;
140
141         PieCtrlRegs.PIEIER1.bit.INTx7 = 1;       //Habilitar TINT0 en PIE; Grupo 1 interrupcion 7
142         PieCtrlRegs.PIEIER9.bit.INTx1 = 1;       // PIE Group 9, INT1 (SCI-A RX)
143         EDIS;
144
145         // Habilitar interrupciones globales.
146         EINT;   // Habilitar Global interrupt INTM
147         ERTM;   // Habilitar Global realtime interrupt DBGM,
148
149       /*== Paso 6. Ciclo principal. ==*/
150
151         /* Inicialización de variables */
152         ms = 0;
153         i = 0;
154
155         /* Motores apagados, inclinación nula, no hay giro, baterías a full */
156         m_izq = 0;
157         m_der = 0;
158         tilt = 0;
159         dtilt = 0;
160         giro = 0;
161         bat = 100;
162         gyro_sensor=1800;
163
164         corriente_izq=0;
165         corriente_der=0;
166         offset_corriente_izq=0;
167         offset_corriente_der=0;
168
169         /* Se inicializa el vector de ángulos */
170         for( i=0; i < N_MUESTRAS_TILT; i++){
```

```
171             dtilt_acum[i] = 0;
172             tilt_acum[i] = 0;
173         }
174         suma_tilt = 0;
175         puntero = 0;
176 
177         /* Ninguna interrupción se ha activado */
178         flag_timer0 = 0;
179         flag_update_LCD = 0;
180         flag_rs232 = 0;
181 
182         /* No se ha caído ni hay sobrecorriente*/
183         flag_caida = 0;
184         flag_corrientes = 0;
185 
186         cont_sobrecorriente_izq = 0;
187         cont_sobrecorriente_der = 0;
188 
189         /* Se inicia la conversión del ADC */
190         AdcRegs.ADCTRL2.all = 0x2000;
191         /* Se inicializan las variables del ADC */
192         /* Se lee varias veces el ADC para estabilizar los datos */
193         for( i = 0; i < 10000; i++){
194             while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {}
195             gyro_sensor = 0.99*gyro_sensor + 0.01*(AdcRegs.ADCRESULT0>>4);
196             while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {}
197             offset_corriente_izq = (offset_corriente_izq)*0.9 + 0.1*((int)(AdcRegs.ADCRESULT1>>4));
198             while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {}
199             offset_corriente_der = (offset_corriente_der)*0.9 + 0.1*((int)(AdcRegs.ADCRESULT2>>4));
200             while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {}
201             bat = 0.9*bat + 0.1*(AdcRegs.ADCRESULT3>>4);
202         }
203 
204         /* Se ajusta el cero del giróscopo */
205         gyro_zro = gyro_sensor;//getGyro_ZRO();
206         gyro_efectivo = 0;
207 
208         for( i = 0; i < N_MUESTRAS_TILT+111; i++)
209             getTilt();        // Para limpiar los valores basura
210 
```

```
211         StartCpuTimer0();
212
213         /* Ciclo infinito */
214         while(1){
215             /* Interrupción del Timer0 */
216             if(flag_timer0){              // cada 10 ms...
217                 tilt = getTilt();       // Obtengo el ángulo
218
219                 /* Obtiene los ciclos de trabajo necesarios según las variables del estado */
220                 getPWM(&m_izq, &m_der, tilt, giro, bat);
221
222                 /* Actualiza los ciclos de trabajo solicitados */
223                 update_pwm(&m_izq, &m_der);
224
225                 /* Obtiene los valores de gyro, corrientes y bat */
226                 getADC();
227
228                 ms = ms + 10;
229
230                 //Cada 100 ms actualiza la información de LCD
231                 if(ms >= 100) flag_update_LCD = 1;
232
233                 /* Limpia el flag de la interrupción */
234                 flag_timer0 = 0;
235             }
236
237             /* Actualizar información pantalla LCD */
238             if(flag_update_LCD){
239                 updateLCD();
240                 ms = 0;
241                 flag_update_LCD = 0;
242             }
243
244             /* Interrupción puerto serie */
245             if(flag_rs232)
246                 flag_rs232 = 0;
247
248         }       //Fin While
249 }
250
```

```
int getTilt(){
	int aux;
	/* Petición de dato y lectura de SPI */
	spi_xmit(INCL_OUT);
	  while(SpiaRegs.SPIFFRX.bit.RXFFST !=1) { }
	aux = SpiaRegs.SPIRXBUF;
	aux = MASK14 & aux;

	if(aux > 7200){	// si es mayor a 180º, lo leo negativo
		aux = aux - 14400;
	}
	aux = -aux;
	aux = aux - OFFSET;

	/* Calculo de la derivada numérica */
	if(puntero == 0) dtilt_acum[puntero] = aux - tilt_acum[N_MUESTRAS_TILT - 1];
	else dtilt_acum[puntero] = aux - tilt_acum[puntero - 1];
	dtilt = dtilt*0.8 + 0.2*dtilt_acum[puntero];

	/* Se actualiza y guarda el dato */
	suma_tilt = suma_tilt + aux - tilt_acum[puntero];
	tilt_acum[puntero++] = aux;
	aux = suma_tilt/N_MUESTRAS_TILT;
	if(puntero == N_MUESTRAS_TILT) puntero = 0;

	/* Si el valor actual de gyro es "normal", se utiliza */
	if(abs(gyro_efectivo) < 60)
		aux = (tilt + gyro_efectivo*14.65*0.01)*0.9 + 0.1*aux;
	else
		aux = tilt*0.8 + 0.2*aux; //09-01
	return aux;
}

void getPWM(int *m_izq, int *m_der, int tilt, int giro, Uint16 bat){
	int m_izq_new, m_der_new;
	int delta_pwm, signo, aux_gyro, signo_gyro;

	m_izq_new = 0;
	m_der_new = 0;
```

```
291         signo = 1;
292         if(tilt < 0) signo = -1;
293         signo_gyro = 1;
294         if(gyro_efectivo < 0) signo_gyro = -1;
295
296         /* Si ha caído, pero el ángulo es cercano a cero, enciende nuevamente */
297         if(flag_caida == 1 && abs(tilt) < 120)
298               flag_caida = 0;
299
300         /* Parte derivativa, aporta hasta un 5.7% de ciclo de trabajo por motor */
301         if(abs(gyro_efectivo) > 3 && abs(gyro_efectivo) < 60)
302               aux_gyro = gyro_efectivo - signo_gyro*3;
303         else if(abs(gyro_efectivo) > 60)
304               aux_gyro = signo_gyro*57;
305         else if(abs(gyro_efectivo) <= 3 || abs(gyro_efectivo) > 90)
306               aux_gyro = 0;
307
308         if(flag_caida == 0){
309               m_izq_new += aux_gyro;
310               m_der_new += aux_gyro;
311         }
312
313         /* Volante, giro : [-68; 68] */
314         if(abs(tilt) < 60){
315               m_izq_new += giro*2.06; // *124/60
316               m_der_new += -giro*2.06;
317         }
318         else if(signo*giro > 0){       //signos iguales
319               m_izq_new += giro*3/6*(-abs(tilt)+240)/180; // *30/60
320               m_der_new += -giro*15/6*(-abs(tilt)+240)/180; // *150/60
321         }
322         else if(signo*giro < 0){       //signos distintos
323               m_izq_new += giro*15/6*(-abs(tilt)+240)/180; // *150/60
324               m_der_new += -giro*3/6*(-abs(tilt)+240)/180; // *30/60
325         }
326
327         /* Parte proporcional */
328         /* [1.5º; 6º] */
329         if(abs(tilt) < 240 && abs(tilt) > 60 && flag_caida == 0){
330               m_izq_new += signo*100 + tilt/2.5;//2.6
```

```
331             m_der_new += signo*100 + tilt/2.5;//2.6                    19.6
332         }
333         /* [6º; 12º] */
334         else if(abs(tilt) < 480 && abs(tilt) > 60 && flag_caida == 0){
335             m_izq_new += signo*(0.0319*tilt)*(0.0319*tilt) - 0.09166*tilt + signo*158;
336             m_der_new += signo*(0.0319*tilt)*(0.0319*tilt) - 0.09166*tilt + signo*158;
337         }
338
339         /* Saturación, [12º; 15º]*/
340         else if(abs(tilt) > 480 && flag_caida == 0){
341             m_izq_new += signo*350;
342             m_der_new += signo*350;
343         }
344
345         /* Si el ángulo es mayor que 15º, caída */
346         if(abs(tilt) > 600){
347             m_izq_new = 0;
348             m_der_new = 0;
349             flag_caida = 1;
350         }
351
352         /* Una vez calculados los ciclos de trabajo, se limitan a un avance de 2% máximo c/10 ms */
353         /* si es que no hay sobrecorriente, sino, se baja el ciclo de trabajo a la mitad */
354         if(flag_corrientes == 0){
355             /* Motor izquierdo */
356             delta_pwm = m_izq_new - *m_izq;
357             signo = 1;
358             if(delta_pwm < 0) signo = -1;
359
360             if(abs(delta_pwm) > MAX_AUMENTO) delta_pwm = signo*MAX_AUMENTO;
361             *m_izq += delta_pwm;
362
363             /* Motor derecho */
364             delta_pwm = m_der_new - *m_der;
365             // Si es un aumento, signo positivo
366             signo = 1;
367             if(delta_pwm < 0) signo = -1;
368
369             if(abs(delta_pwm) > MAX_AUMENTO) delta_pwm = signo*MAX_AUMENTO;
370             *m_der += delta_pwm;
```

```
371         }
372         else{
373               *m_izq = *m_izq/2;
374               *m_der = *m_der/2;
375         }
376   }
377
378   void getADC(){
379         Uint16 aux;
380
381         while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {}
382         aux = (AdcRegs.ADCRESULT0>>4);
383
384         gyro_sensor = gyro_sensor*0.9 + 0.1*aux;
385         if(getGyro_ZRO() == 1) gyro_zro = gyro_zro*0.5 + gyro_sensor*0.5;
386         gyro_efectivo = 0.5*gyro_efectivo + 0.5*((int)gyro_sensor - (int)gyro_zro);
387
388         while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {}
389         aux = ((AdcRegs.ADCRESULT1>>4));
390         corriente_izq = corriente_izq*0.95 + 0.05*((int)aux - offset_corriente_izq);
391
392         while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {}
393         aux = ((AdcRegs.ADCRESULT2>>4));
394         corriente_der = corriente_der*0.95 + 0.05*((int)aux - offset_corriente_der);
395
396         while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {}
397           bat = 0.95*bat + 0.05*(AdcRegs.ADCRESULT3>>4);
398
399         AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1;
400
401         /* Si la corriente ya bajó, se apaga el flag */
402         if(abs(corriente_izq) < CURRENT_MAX-200) cont_sobrecorriente_izq = 0;
403         if(abs(corriente_der) < CURRENT_MAX-200) cont_sobrecorriente_der = 0;
404         if(cont_sobrecorriente_izq == 0 && cont_sobrecorriente_der == 0)
405               flag_corrientes = 0;
406
407         /* Si se sobrepasa la corriente, aumenta el contador */
408         /* Si es por más de 50 ms, enciende el flag */
409         if(abs(corriente_izq) > CURRENT_MAX) cont_sobrecorriente_izq++;;
410         if(cont_sobrecorriente_izq > 5)       flag_corrientes = 1;
```

```
411
412         if(abs(corriente_der) > CURRENT_MAX) cont_sobrecorriente_der++;
413         if(cont_sobrecorriente_der > 5)       flag_corrientes = 1;
414     }
415
416     void updateLCD(){
417         int aux;
418
419         /* Envío de tilt */
420         aux =  tilt;
421         if(tilt > 1023) aux = 1023;
422         if(tilt < -1024) aux = -1024;
423         aux = aux/8; //se ajusta al rango [-128,127]
424
425         if(tilt < 0) aux = aux | 0x80;
426         scia_xmit('t');
427         scia_xmit(aux);       // se ajusta a los 8 bits de SCI
428         scia_xmit(0x0D);
429
430         /* Envio de giróscopo */
431         aux = gyro_efectivo;//gyro_sensor;
432         aux = (aux>>8);       // se corre 8 bits (2^8=256) para tomar los MSB
433         scia_xmit('g');
434         scia_xmit(aux);       //MSB
435         scia_xmit(gyro_efectivo/*gyro_sensor*/);       //LSB
436         scia_xmit(0x0D);
437
438         /* Envio de motor izquierdo */
439         aux = m_izq/10;
440         if(aux < 0) aux = aux | 0x80;
441         scia_xmit('i');
442         scia_xmit(aux);       // se ajusta a los 8 bits de SCI
443         scia_xmit(0x0D);
444
445         /* Envio de motor derecho */
446         aux = m_der/10;
447         if(aux < 0) aux = aux | 0x80;
448         scia_xmit('d');
449         scia_xmit(aux);       // se ajusta a los 8 bits de SCI
450         scia_xmit(0x0D);
```

```
451
452         /* Envio de corriente motor izquierdo */
453         aux = abs(corriente_izq);
454         if(aux > 1023) aux = 1023;
455         aux = aux/4;        // se corre 2 bits (2^2) para tomar los 8 primeros bits
456         scia_xmit('l');
457         scia_xmit(aux);       // se ajusta a los 8 bits de SCI
458         scia_xmit(0x0D);
459
460         /* Envio de corriente motor derecho */
461         aux = abs(corriente_der);
462         if(aux > 1023) aux = 1023;
463         aux = aux/4/*0+413/4*/;       // se corre 2 bits (2^2) para tomar los 8 primeros bits
464         scia_xmit('r');
465         scia_xmit(aux);       // se ajusta a los 8 bits de SCI
466         scia_xmit(0x0D);
467
468         /* Envio de estado de baterias */
469         aux = bat;
470         aux = aux/4/*0+413/4*/;       // se corre 2 bits (2^2) para tomar los 8 primeros bits
471         scia_xmit('b');
472         scia_xmit(aux);       // se ajusta a los 8 bits de SCI
473         scia_xmit(0x0D);
474
475   }
476
477   /* Interrupción RS232 */
478   interrupt void sciaRxFifoIsr(void){
479         Uint16 aux;
480         aux = SciaRegs.SCIRXBUF.all;        // Read data
481         if(aux != 255) giro = 0.5*((int)CENTRO_VOLANTE - (int)aux) + 0.5*giro;
482
483         SciaRegs.SCIFFRX.bit.RXFFOVRCLR = 1;   // Clear Overflow flag
484         SciaRegs.SCIFFRX.bit.RXFFINTCLR = 1;   // Clear Interrupt flag
485         SciaRegs.SCIFFRX.all;
486         PieCtrlRegs.PIEACK.all |= PIEACK_GROUP9;
487         flag_rs232 = 1;
488   }
489
490   /*Interrupción cada 10 milisegundos*/
```

```
491 interrupt void cpu_timer0_isr(void){
492         CpuTimer0.InterruptCount++;
493          // Acknowledge this interrupt to receive more interrupts from group 1
494         PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
495         flag_timer0 = 1;
496 }
497
498 /* Entrega '1' si la medición del acelerómetro indica */
499 /* que el giróscopo DEBIESE marcar 0 º/sec */
500 Uint16 getGyro_ZRO(){
501         int tilt_0, tilt_1, i;
502         int punt;
503         long diff, dtilt_suma;
504
505         dtilt_suma = 0;
506         /* Valor más antiguo */
507         tilt_0 = tilt_acum[puntero];
508
509         /* Valor más nuevo */
510         punt = puntero -1;
511         if(punt < 0) punt = N_MUESTRAS_TILT-1;
512         tilt_1 = tilt_acum[punt];
513
514         diff = (tilt_0 - tilt_1);
515
516         for(i = 0; i < N_MUESTRAS_TILT; i++)
517                 dtilt_suma += dtilt_acum[i];
518
519         dtilt_suma = dtilt_suma/N_MUESTRAS_TILT;
520
521
522         if(abs(diff) < 5 && abs(dtilt_suma) < 5 ) return 1;
523         else return 0;
524 }
```

D.2. Códigos programados en PIC (*main_lcd.c*)

```
#include <18F252.h>
#device ADC=10

#fuses HS,NOWDT,PUT,NOLVP
#use delay(clock=16000000)
#include "flex_lcd.c"
#include <stdlib.h>

#use rs232(baud=19200, xmit=PIN_C6, rcv=PIN_C7, stream = DSP)

/* Pines para Leds */
#define LED_V PIN_A0
#define LED_A PIN_A2

/* Valor ASCII para el signo de grado y de porcentaje*/
#define ASCII_GRADO 223
#define ASCII_PORC 37


// CONSTANTES
int const lenbuff=50; // Longitud de buffer

// VARIABLES EN RAM
int xbuff = 0x00;      // Índice: siguiente char en cbuff
char cbuff[lenbuff]; // Buffer
char txt[50];
char rcvchar = 0x00;    // último carácter recibido
int1 flag_rs232 = 0;    // Flag para indicar comando disponible
int1 flag_int_rs232 = 0;    // Flag para interrupción serial
int1 flag_txt = 0;      // Si se activa, se mantiene impreso en LCD el último texto enviado
int1 flag_timer = 0;    //Flag para timer
int1 flag_button1 = 0;  //Flag para boton 1
int1 flag_button2 = 0;  //Flag para botón 2
int1 estado = 0;        // Flag para verificar petición de botón
int1 seguro = 0;
int1 flag_aux = 0;

```

```
//0 para mostrar angulo, bateria y velocidad, 1 para mostrar ciclos de trabajo y corrientes
int1 flag_ventana = 0;

signed int tilt = 0;     // Angulo actual
signed long gyro = 0;    // Giróscopo, 12 bits
signed int velocidad = 0;
int bat = 100;    // Estado Baterías
int encoder = 0;
signed int m_izq = 0;
signed int m_der = 0;
float corriente_izq = 0.0;
float corriente_der = 0.0;

// Declaración de Funciones

void inicbuff(void);        // Borra buffer
void addcbuff(char c);       // añade carácter recibido al buffer
void actualiza_datos(void); // Procesa comando
unsigned int get_encoder(void);

/* valor encoders */
int get_encoder(void){
        signed int16 aux;
        aux = read_adc();
        aux = (aux < 400) ? 0 : aux - 400;
        aux = (aux > 255) ? 255 : aux;
        return (signed int) aux;
}


// INTERRUPCIONES
#int_rda
void serial_isr() {    // Interrupción recepción serie USART
        rcvchar=0x00;        // Inicializo carácter recibido
        if(kbhit()){         // Si hay algo pendiente de recibir ...
                rcvchar=getc();    // lo descargo y ...
                addcbuff(rcvchar); // lo añado al buffer y ...
        }
        if(flag_aux == 0){
                flag_aux = 1;
```

```
78                  inicbuff(); // Borro buffer.
79          }
80  }
81
82  /* Interrupción del Timer0, cada 65 ms */
83  /* Actualiza la información del LCD y envía */
84  /* datos del encoder al DSP.                */
85  #int_RTCC
86  void RTCC_isr(){
87          flag_timer = 1;
88  }
89
90  /* Interrupción botón 1 (izquierda)*/
91  #int_ext1
92  void ext1_isr(){
93          flag_button1 = 1;
94          flag_ventana = 0;
95  }
96
97  /* Interrupción botón 2 (derecha)*/
98  #int_ext
99  void ext_isr(){
100         flag_button2 = 1;
101         flag_ventana = 1;
102 }
103
104 void main(){
105         float aux;
106         delay_ms(500);
107         // TIMER0: Clock Interno 16 bits, Presescaler 16
108         // para una RTCC cada 65,536 milisegundos
109         setup_counters(RTCC_INTERNAL, RTCC_DIV_8);
110         setup_timer_1(T1_DISABLED);
111         setup_timer_2(T2_DISABLED,0,1);
112
113         enable_interrupts(INT_RTCC);     // Habilito Interrupción RTCC
114         enable_interrupts(INT_EXT1);     // Interrupción para Botón 1
115         enable_interrupts(INT_EXT);      // Interrupción para Botón 2
116         enable_interrupts(int_rda);      // Interrupción serial
117         enable_interrupts(global);       // Habilito Interrupciones
```

```
118
119         ext_int_edge(1, L_TO_H);        /* Botones se activan al soltar */
120         ext_int_edge(0, L_TO_H);
121
122         port_b_pullups(TRUE);
123
124         /* Setea los conversores AD */
125         setup_adc(ADC_CLOCK_INTERNAL); //configura el conversor
126         set_adc_channel(1);
127         delay_ms(50);
128
129         /* Enciende LED's para debugging */
130         output_high(LED_V);
131         output_low(LED_A);
132
133         inicbuff(); // Borra buffer al inicio
134         lcd_init();  // Inicializa LCD
135
136         delay_ms(2000);
137         flag_ventana = 0;
138         aux = 0;
139
140         while(1){
141               /* Interrupción serial */
142               if(flag_rs232)
143                     actualiza_datos(); // Si hay comando pendiente
144
145               /* Timer cada 65,536 ms */
146               if(flag_timer){
147                     encoder = get_encoder();
148                     putchar(encoder);
149                     aux = ((float) tilt)*25.6/128;
150                     if(flag_ventana == 0)
151                           printf(lcd_putc,"\fInc:%3.1f%c Bt:%u%c\nGy:%ld V:%d km/h",
152                                 aux, ASCII_GRADO, bat, ASCII_PORC, gyro, velocidad);
153                     else
154                           printf(lcd_putc,"\fIz:%d%c %2.1f [A]\nDe:%d%c %2.1f [A]",
155                                 m_izq, ASCII_PORC, corriente_izq, m_der, ASCII_PORC, corriente_der);
156                     flag_timer = 0;
157               }
```

```
158         }//Fin del While
159
160     }
161
162     void inicbuff(void){ // Inicia a \0 cbuff ------------------
163         int i;
164         for(i=0;i<lenbuff;i++)        // Bucle que pone a 0 todos los
165             cbuff[i]=0x00;                          // caracteres en el buffer
166       xbuff=0x00;                                       // Inicializo el índice de siguiente carácter
167
168     }
169
170     void addcbuff(char c){ // Añade a cbuff
171         switch(c){
172             case 0x0D:                              // Enter -> Habilita Flag para procesar
173                 flag_rs232=1;                   // Comando en Main
174                 break;
175             default:
176                 cbuff[xbuff++]=c;        // Añade carácter recibido al Buffer
177         }
178     }
179
180     void actualiza_datos(void){
181         int i;
182         long aux;
183         char arg[lenbuff]; // Argumento de comando (si lo tiene)
184
185         if(cbuff[0]=='i')
186             output_toggle(LED_V);
187
188         flag_rs232=0; // Desactivo flag de comando pendiente.
189
190         for(i=0;i<lenbuff;i++) // Bucle que pone a 0 todos los
191             arg[i]=0x00; // caracteres en el argumento
192
193         /* Valor de inclinación */
194         if(cbuff[0]=='t') // Recibo valor de inclinación
195             tilt = cbuff[1];
196
197     /* Recibo valor del giróscopo */
```

```
198         else if(cbuff[0]==’g’){ // Recibo giróscopo
199             gyro = 0x0000;
200             aux = cbuff[1];
201             gyro = gyro | (aux<<8);  // MSB
202             gyro = gyro | cbuff[2];
203         }
204
205         /* Recibo valor de estado de las baterías */
206         else if(cbuff[0]==’b’) // Recibo estado de las baterías
207             bat = cbuff[1];
208
209         else if(cbuff[0]==’v’) // Recibo valor de la velocidad
210             velocidad = cbuff[1];
211
212         /* Valor de ciclo de trabajo del motor izquierdo */
213         else if(cbuff[0]==’i’)
214             m_izq = cbuff[1];
215
216         /* Valor de ciclo de trabajo del motor derecho */
217         else if(cbuff[0]==’d’)
218             m_der = cbuff[1];
219
220         /* Valor de corriente del motor izquierdo */
221         else if(cbuff[0]==’l’)
222             corriente_izq = ((float)cbuff[1])*0.1;
223
224         /* Valor de corriente del motor derecho */
225             else if(cbuff[0]==’r’)
226                 corriente_der = ((float)cbuff[1]*0.1);
227
228         inicbuff(); // Borro buffer.
229     }
```

Apéndice E

Material desarrollado

El DVD adjunto contiene el material desarrollado en este proyecto.

- Esquemáticos de las distintas tarjetas electrónicas
- Códigos de las distintas plataformas desarrolladas
- Archivos CAD del diseño mecánico del vehículo
- Videos de las distintas pruebas relizadas