
**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**Facultad de Ingeniería Eléctrica y Computación**

# Control de robot oruga con controlador Orangután SV-328 mediante joystick utilizando el Kit AVR Butterfly en interfaz inalámbrica por radio frecuencia

**TESINA DE SEMINARIO**

Previo a la obtención del Título de:

**INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES**

Presentado por:

Carlos Luis Gutiérrez Pincay
César Miguel Robalino Ponce

GUAYAQUIL – ECUADOR

2011

# A G R A D E C I M I E N T O

A Dios, por ser la guía de nuestro camino y fuente de esperanza, por colmarnos de bendiciones y brindarnos aliento y esperanza.

A nuestros padres, por su apoyo y consejos para nuestro diario vivir, por sus deseos de vernos triunfantes y por inculcarnos metas y sueños.

Al Ing. Carlos Valdivieso por su guía a través del desarrollo de nuestro proyecto, por su ejemplo de profesionalismo.

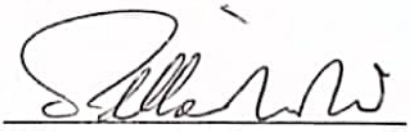
# DEDICATORIA

A nuestros padres, pilar fundamental de nuestras vidas, presentes durante todo este tiempo de aprendizaje y formación profesional.

## TRIBUNAL DE SUSTENTACIÓN

ING. CARLOS VALDIVIESO A.

**PROF. DEL SEMINARIO DE GRADUACIÓN**



ING. HUGO VILLAVICENCIO V.

**DELEGADO DEL DECANO**

## DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesina, nos corresponde exclusivamente; y el patrimonio intelectual de la misma, a la Escuela Superior Politécnica del Litoral".

(Reglamento de Graduación de la ESPOL)

Carlos Luis Gutiérrez Pincay

César Miguel Robalino Ponce

# RESUMEN

El proyecto consiste en el uso de una comunicación inalámbrica para la transmisión de datos enviados desde el joystick del KIT AVR BUTTERFLY, al controlador Orangután SV-328 el cual facilita el control del robot oruga, haciendo uso de las herramientas de software AVR Studio y Proteus.

El proyecto fue desarrollado en lenguaje C haciendo uso del AVR Studio con su compilador GCC, el cual permite la compilación no solo de lenguaje C sino también de C++.Cabe recalcar que al ser un software libre y trabajar bajo el entorno de AVR no estamos limitados a un tamaño mínimo de generación de código y la mayoría de funciones se encuentran a disposición para ser modificados.

Para nuestro caso, la interfaz inalámbrica se obtiene haciendo uso de los módulos RF HM-TR434 y HM-TR434 los cuales modulan y demodulan la señal correspondiente. De ellos depende la distancia máxima a la cual se pueden encontrar sus módulos

El kit AVR Butterfly el cual funcionara como transmisor de las señales obtenidas desde el joystick que posee. Al presionar algún botón del joystick se presentara el nombre de la instrucción en el LCD y se genera la comunicación serial con la interfaz inalámbrica. Por otro lado el módulo Orangután SV-328 que recibe la señal a través de la interfaz inalámbrica RF procesa los datos y genera el movimiento de los motores del robot oruga permitiendo así un desplazamiento.

# INDICE GENERAL

# INDICE DE FIGURAS

# INTRODUCCION

Para el desarrollo del mismo se hace uso del chasis RP5, el cual consta de dos grupos de engranajes que se conectan a las dos motores permitiendo el movimiento de los ejes que contienen el disco por el cual se mueve la banda de caucho que permite que tenga mayor tracción el robot.

La alimentación correspondiente del controlador Orangután SV- 328 depende de un grupo de 6 Baterías AA y posee la conexión correspondiente a los motores del chasis.

La alimentación del Kit AVR Butterfly se hace a base de un tipo de batería que le suministra la suficiente corriente para su debido funcionamiento. La configuración de sus puertos se realiza mediante la correspondiente programación.

Los capítulos se encuentran estructurados de la siguiente manera:

En el capítulo uno encontraremos una descripción detallada de los antecedentes del proyecto, una breve descripción del problema, aplicaciones que se podrían implementar y comparativas con otros kits de robots que utilicen orugas para movilizarse.

En el capítulo dos se detalla las herramientas utilizadas para el proyecto como son los módulos Orangután y Butterfly, el robot Oruga RP5, los módulos usados para la transmisión inalámbrica y las herramientas de software como son AVR Studio y Proteus.

En el Capítulo tres se describe la idea principal del proyecto, la forma en la cual se fue realizando la unión de las diversas partes para su completo funcionamiento y las funciones utilizadas en ambos módulos para simplificar la programación y conseguir una aplicación más eficiente la cual realice las funciones esperadas.

Y por último en el capítulo cuatro se describen los diagramas esquemáticos del Orangután SV-328, el Kit AVR Butterfly. Y las mediciones correspondientes de las pruebas efectuadas para la configuración correspondiente del robot oruga.

# CAPÍTULO 1

## 1 DESCRIPCIÓN GENERAL DEL PROYECTO

### 1.1 Antecedentes

La Robótica, desde sus inicios hasta la actualidad, ha evolucionado y experimentado cambios que, gracias a la tecnología, en los últimos años ha permitido optimizar el control de robots por medio de controladores programables facilitando la utilización de motores que necesitan un mayor torque, debido a que poseen drivers que satisfagan las condiciones requeridas y de eso depende que modelo se selecciona de mejor manera.

Para el caso de los robots orugas que en general se construyen uniendo, mediante una cadena que rodea las llantas, las ruedas delanteras y traseras, cuyo fin es aumentar la superficie de contacto con el suelo y conseguir una mayor tracción. Entre sus ventajas se cuenta con que permiten rebasar mayores obstáculos que solamente usando ruedas e incluso subir escaleras. Cabe recalcar que el RP5 tiene un caucho que funciona como cadena.

El presente proyecto se realizó base del módulo Orangután SV-328 el cual permita controlar robot pequeños para nuestro caso el robot oruga debido a que su regulador permite suministrar un mayor nivel de corriente (3A), permitiendo así un mejor funcionamiento de los controladores de los motores debido a que posee dos puertos que suministran desde 1 a 3 Amperios por canal, la alimentación del módulo puede ser de 6.7 a 13V.

A su vez contamos con el joystick que se encuentra incorporado en el Kit AVR Butterfly, el cual puede operar en 5 direcciones posibles incluyendo la función central. Para que el robot se desplace según los movimientos del joystick necesitaremos solo 4 direcciones, quedando libre la posición central que podrá ser usada en alguna otra función del robot.

## 1.2 DESCRIPCIÓN GENERAL DEL PROYECTO

El proyecto consiste en la implementación de un sistema que permita el control de un robot oruga con controlador Orangután SV-328, inalámbricamente dependiendo de las instrucciones enviadas por el joystick del Kit AVR Butterfly.

EL Orangután SV-328 es un módulo programable el cual nos permite entre varias de sus aplicaciones controlar leds, LCD y motores debido a un integrado que permite suministrar la corriente requerida sin dañar al microcontrolador ni demás componentes.

El kit AVR Butterfly permite entre sus diferentes aplicaciones la transmisión de datos a través del estándar RS-232. Las señales del joystick serán enviadas a través de comunicación serial a un módulo de transmisión RF.

Dicha señal será recibida por un módulo receptor de RF acoplado al módulo Orangután para, con dichas instrucciones, realizar el control del robot oruga RP5.

## 1.3 APLICACIONES

A pesar de que el robot oruga RP5 es de un tamaño pequeño puede realizar varias acciones que dan a notar una considerable robustez con respecto a sus dimensiones físicas entre las cuales podemos citar:

- **Rescate y Búsqueda**

En lo que concierne a rescate y búsqueda para nuestro caso debido a las limitaciones del módulo Orangután SV-328 para lo que sería remolcar algún objeto y como esto afectaría la suministración de corriente por canal.

- **Exploración**

Para realizar una exploración en un lugar de difícil acceso, pero el principal problema sería el tipo de módulos RF usados, y la visibilidad que tenga ya que el robot no posee una cámara que nos permita ver el tramo de camino.

- **Robot War**

Al usarlo de esta manera sería inofensivo debido a que no posee armas o cosas con que defenderse de un ataque pero por su tracción, los obstáculos no son problemas.

## 1.4 PROYECTOS SIMILARES

### 1.4.1 Control de robot TRI TRAC mediante radio control con joystick de Play Station 2

TRI-TRAC es un robot todo terreno con orugas que puede ser usado mediante radio control o mediante control autónomo. Debido a su oruga el robot puede trabajar tanto en interior como exterior. El kit incluye:

- Parte mecánica con motores y oruga.
- Circuito NextStep con procesador ATCOM.
- Circuito de control de motores.
- Mando a distancia.

El precio en el mercado de este kit es de 363,04 €.

Fig. 1.4.1: Robot TRI TRAC con control PS2.

### 1.4.2 Detector de obstáculos y graficación de posición de obstáculos

El proyecto consiste en la detección de obstáculos de manera que puedan ser evitados y enviar mediante comunicación inalámbrica la posición del obstáculo para mostrarla a través del PC en el programa Stamplot.

Para la elaboración del proyecto se usa un robot XBOT el cual cuenta con servomotores, un sensor ultrasónico y un módulo inalámbrico. El control del robot será realizado a través del microcontrolador PIC16F876.

Al funcionar el robot detecta todo obstáculo a una distancia de 10 cm. En ese momento, por programación, el robot gira 90º hacia la izquierda, envía su posición a través del módulo inalámbrico. Al no encontrar obstáculo continuará su recorrido actualizando cada ciclo de programa su posición. El receptor consiste en otro PIC16F876 que tiene el módulo de recepción inalámbrica en donde cada dato que es recibido es enviado al Stamplot para la graficación.

Fig. 1.4.2.1: Robot XBOT

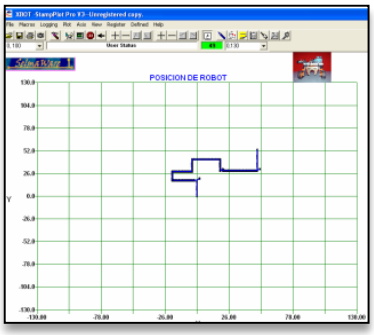



Fig. 1.4.2.2: Graficación de posición en Stamplot

### 1.4.3 Control de microbot SIKO con tarjeta CT6811

SIKO es un robot oruga de bajo costo que puede ser utilizado por aquellas personas que se inician en el mundo de la robótica. El esqueleto está armado con una plancha de aluminio y cuenta además con servomotores modificados para poder funcionar como motores de corriente continua normal y poder rotar 360º. Se tiene como finalidad obtener un robot de bajo costo capaz de movilizarse a través de superficies no lisas y con pequeños obstáculos.

Para el control del robot se utiliza la tarjeta CT6811, basada en el microcontrolador 68HC11 de Motorola y la tarjeta CT293+ para el control de los motores y sensores de infrarrojo, aunque se puede usar otro microcontrolador.

SIKO es un robot abierto, es decir que tanto los esquemas como la documentación de su estructura mecánica están disponibles y su elaboración y modificación están permitidas.

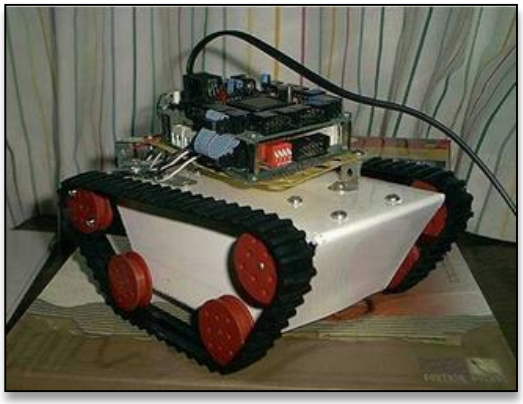


Fig. 1.4.3.1: Microbot SIKO

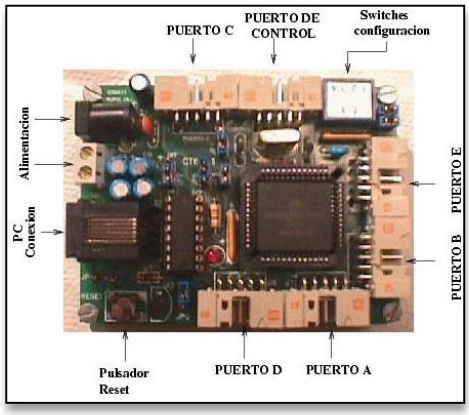



Fig. 1.4.3.2: Tarjeta CT6811

# CAPÍTULO 2

## FUNDAMENTO TEÓRICO

### 2.1 AVR BUTTERFLY EVALUATION KIT

El kit de evaluación del AVR Butterfly está configurado para demostrar los beneficios de usar microcontroladores ATMEL.

El AVR Butterfly contiene un microcontrolador ATMega169, el cual va a realizar el comando de las diferentes funciones de las que es capaz éste kit.

En el kit contamos con una resistencia LDR con la cual podemos realizar mediciones de intensidad lumínica y de ese modo desarrollar programas los cuales trabajen en función de la iluminación de un determinado lugar como por ejemplo buscar la ubicación de una fuente lumínica y mostrar en la LCD el valor correspondiente a dicha iluminación.

También cuenta con un sensor de temperatura NTC con el cual podemos desarrollar programas que actúen en función de la temperatura leída a través del termistor e ingresada al microcontrolador a través de su convertidor ADC de 10 bits de resolución.

El AVR Butterfly cuenta con un joystick de cuatro posiciones (Arriba, Abajo, Izquierda, Derecha) y una posición central. En el programa demo podemos hacer uso de las diferentes funciones precargadas en el kit a

través de dicho joystick. Una de esas funciones es la de reproducir una melodía a través del buzzer piezoeléctrico con el que cuenta el kit.

El AVR Butterfly permite cargar programas desarrollados por un usuario gracias a su puerto ISP y el uso de un software desarrollado para esta familia de micros como lo son el AVR Studio y el WinAVR. El programador Pololu USB AVR Programmer se conecta al PC a través de un puerto USB lo cual permite que cualquier PC con puerto USB pueda ser usada para realizar la grabación del programa.

Hay que mencionar además que este módulo cuenta con un set de instrucciones ya programadas las cuales hacen que el uso de las ventajas de este kit sea bastante amplio. Entre las aplicaciones precargadas se cuenta con un Bootloader el cual permite realizar la programación del módulo a través de una conexión con el puerto serial del computador.

El kit cuenta también con la opción de realizar una comunicación RS-232 gracias a la funcionalidad del ATMega169 y a un convertidor de nivel que el kit posee. Añadiendo un modelo de transmisión RF podemos realizar comunicación inalámbrica lo cual hace atractivo el uso del AVR Butterfly para la elaboración de esta clase de proyectos.

Su alimentación se realiza a través de una batería tipo botón de 3 V la cual proporciona la energía necesaria para su funcionamiento a razón de 600 mAh.

Gracias a su facilidad de manejo y a todas las funciones que incorpora incluyendo la opción de conectar otros módulos vamos a utilizar el AVR Butterfly para enviar comandos vía RF al módulo Orangután SV-328.

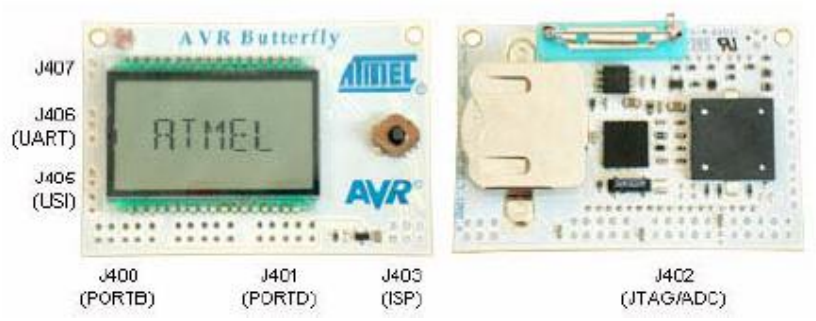



Fig. 2.1: Módulo AVR Butterfly

## 2.2 Orangután SV-328

El módulo Orangután SV-328 es una herramienta que permite el control de varios periféricos incluyendo leds, motores, LCD. Su tamaño (2,15'' x 1,9'') permite que sea adaptable en diferentes placas de modo que no ocupa mucho espacio y da una amplia funcionalidad.

El Orangután posee una pantalla LCD de 8 caracteres por 2 líneas lo cual permite mostrar a través de mensajes las funciones que se pueden realizar. El contraste de la LCD puede ser variado a través de un potenciómetro en la tarjeta en caso de que el contraste actual de la LCD no sea el deseado.

El módulo posee en su interior el microcontrolador ATMega328 el cual se encarga de comandar las diferentes funciones disponibles y además el reprogramado del microcontrolador.

El Orangután SV-328 es una versión mejorada de un anterior modelo de Orangután. Entre dichas mejoras podemos mencionar el amplio rango de voltaje de alimentación ya que el módulo puede ser alimentado por voltajes que van desde 6.7V hasta 13V.

El control del voltaje se hace a través de un regulador de 5V el cual permite manejar valores de corriente hasta de 3 A como máximo lo cual

es usado para el control de motores a través de un driver conectado a los pines que generan la señal PWM que puede alcanzar un valor máximo de 80 KHz. La modulación de ancho de pulso la conseguimos gracias a los temporizadores TMR0 y TMR1.

La tarjeta presenta un grupo de 8 pines de entrada/salida de los cuales 6 pueden ser configurados como entradas analógicas los cuales con ayuda de dos entradas adicionales accesibles en la misma tarjeta, ADC6 y ADC7, nos provee de 8 entradas para usar la conversión analógica a digital la cual es de una resolución de 10 bits.

Además cuenta con dos hileras adicionales de conectores los cuales pueden suministrar un voltaje de 5V (hilera central) y un punto de referencia GND (hilera externa) con lo cual se pueden adaptar diferentes sensores al módulo a través de sus pines de entrada/salida y conseguir la polarización de los mismos sin necesidad de fuentes de alimentación externas.

El Orangután SV-328 presenta un conjunto de PUSH-BUTTON para la selección de opciones según la programación que se cargue en el mismo lo cual permite realizar aplicaciones las cuales cuenten con un menú el cual puede ser controlado por el usuario a través de dichos botones con lo cual se consigue una mejor demostración de las capacidades del módulo.

El Orangután SV-328 puede ser programado a través de un programador SPI. Varios programadores pueden ser usados y para este caso vamos a usar el programador Pololu USB AVR Programmer que va a ser conectado al puerto ISP del Orangután a través de un cable de seis hilos.

Entre los programas a disposición para poder desarrollar una aplicación podemos mencionar el AVR Studio y el WinAVR. En nuestro caso usaremos el GCC (GNU C Compiler) del AVR Studio para desarrollar nuestro programa. Cabe mencionar que es existe una librería dedicada al

control del Orangután SV-328 de modo que el desarrollo de aplicaciones usando este módulo es muy amplio.

Gracias al microcontrolador que posee se puede realizar una comunicación RS-232 para recibir información a través de otro módulo, característica que será aprovechada para la realización del proyecto. Es importante mencionar que el Orangután SV-328 no posee un convertidor de nivel de RS-232 a TTL por lo cual no debe conectarse dispositivos cuyos valores de tensiones sean mayores a 5V para evitar un posible daño en el módulo.

A través de un módulo RF el Orangután SV-328 recibirá instrucciones enviadas por el AVR Butterfly para poder controlar los motores del robot oruga RP5 y dado que la salida del módulo receptor (HM-TR434) es de nivel TTL no es necesario añadir una interfaz que convierta el nivel de voltaje a uno que pueda ser soportado por el módulo Orangután.

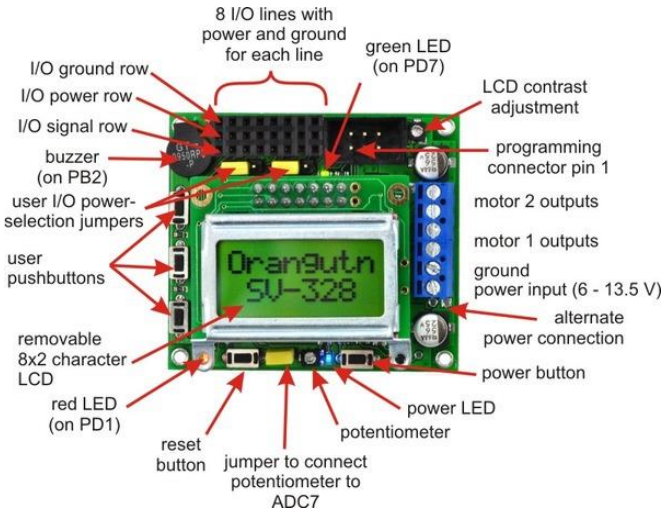



Fig. 2.2: Módulo Orangután SV-328

## 2.3 CHASIS RP5

El chasis RP5 es una plataforma móvil para un robot que usa una oruga para desplazarse por terrenos difíciles, rotar. Cuenta con una base para seis baterías AA y dos motores DC para lograr el movimiento del robot.

Las ruedas del robot oruga conectada a los motores poseen una banda de plástico semejante al de la oruga de los Caterpillar lo cual permite que su campo de desplazamiento sea bastante amplio, logrando incluso escalar por pendientes inclinadas donde el único limitante es la fricción de la superficie. La velocidad máxima que puede ser alcanzada por el RP5 es de 15 cm/s a 7.2V.

Dado que se trata de un chasis con motores se pueden adaptar varios módulos para el control del movimiento del RP5.

El módulo Orangután SV-328 o el Baby Orangután funcionan muy bien con este chasis y permiten un fácil control del mismo. Hay que tener en cuenta que debido al control de los motores se van a manejar niveles de corriente considerables para un circuito comandado por un microcontrolador, por lo tanto se debe evitar que durante el funcionamiento de la oruga ésta no llegué a quedar trabada, caso contrario se puede tener una mayor exigencia de corriente con el posterior daño de la tarjeta de control.

Dentro de los recursos disponibles para el uso del Orangután se cuenta con una librería escrita en lenguaje C con diferentes funciones para controlar el robot.

Gracias a dichas funciones es posible realizar el control de los motores variando su velocidad desde un valor mínimo igual a cero hasta un valor máximo de 255 los cuales son interpretados por el programa para configurar el ciclo de trabajo en la modulación PWM.

Para obtener un giro de motor en dirección opuesta solo basta con incluir el signo menos (-) delante del valor de configuración de la velocidad de los motores con lo cual el microcontrolador comprende que se desea realizar el giro en la otra dirección y a través del driver para motores TB6612FNG se consigue el movimiento de los mismos.

El robot Oruga RP5 no cuenta con circuitos electrónicos internos. Solamente cuenta con 2 motores DC y un conjunto de capacitores los cuales son usados durante el funcionamiento de cada motor.

Dentro del kit del robot se cuenta con una llave hexagonal para realizar la revisión de los motores en caso de un funcionamiento inesperado y de una pequeña bolsa de grasa para realizar el mantenimiento de los engranajes en caso de que las llantas presenten dificultad en moverse debido a la fricción con los engranes.

Fig. 2.3: Robot oruga RP5

## 2.4 AVR STUDIO 4

AVR Studio es un ambiente de desarrollo para escribir y simular aplicaciones para los microcontroladores de la familia ATMEL.

Provee herramientas de manejo, editor de código fuente, que puede ser en lenguaje ensamblador o lenguaje C. Permite visualizar los diferentes bancos que posee el microcontrolador que estemos programando y además facilita manipular esos valores durante la ejecución del programa para observar los cambios generados por los mismos.

El IDE (IntegratedDevelopmentEnvironment) soporta todas las herramientas de ATMEL requeridas para la arquitectura AVR de 8 bits.

El lenguaje ensamblador nos permite manipular con mayor detalle los valores de los registros que posee en microcontrolador que estemos utilizando. Al mismo tiempo posee herramientas que permiten ver los valores que toman las diferentes variables que hayamos declarado a lo largo del desarrollo del programa.

El lenguaje GCC (GNU C Compiler) permite una programación de alto nivel de modo que consigamos desarrollar aplicaciones de mayor grado de desarrollo.

AVR Studio 4 posee además la opción de programación de los microcontroladores ATMEL mediante interfaz JTAG, ISP y Serial, delas cuales serán usadas las dos últimas para realizar la programación de los módulos: la programación Serial para programar el AVR Butterfly mediante la opción de Bootloader y la programación ISP para el módulo Orangután SV-328. Cabe mencionar que el Buttefly también permite cargar una aplicación mediante su puerto ISP de 6 pines con la diferencia que se debe conectar además una polarización externa de 5V y retirar la batería.

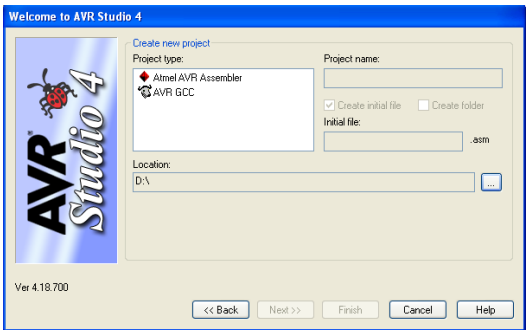



Fig. 2.4.1: Pantalla de inicio del AVR Studio 4

AVR Studio dispone de varios métodos para poder programar los microcontroladores de la familia ATMEL, de las cuales serán usados para la realización de nuestro proyecto las siguientes opciones:

La programación ISP, a la cual se accede mediante la opción AVRISP, permite grabar el microcontrolador tanto del Orangután SV-328 como del AVR Butterfly. Se hace uso del Pololu USB AVRProgrammer el cual se conecta al puerto ISP de los módulos a través de un cable de 6 líneas.

Una vez conectado el módulo a programar el software permite la lectura de la firma del módulo para detectar su estado, lo cual permite comprobar la correcta conexión entre el computador y el módulo. Una vez verificada la conexión se procede al proceso de grabación del microcontrolador en el cual se realiza en primera instancia el borrado del programa que ya contenía antes de cargar la nueva aplicación y finalmente realiza la verificación de la aplicación cargada. Una vez terminado el proceso abandona el modo de programación para que sea seguro desconectar el módulo sin causarle algún daño.

Cabe destacar que para realizar la programación del AVR Butterfly se debe conectar, además del programador, una alimentación externa de 5V y retirar la pila que energiza el módulo. Al encender el Butterfly con la alimentación externa éste debería encender sin ningún problema lo cual sería muestra de que la conexión de la polarización externa ha sido

correctamente realizada y podemos continuar con el proceso de grabación normalmente.

Fig. 2.4.2: Pololu USB AVR Programmer

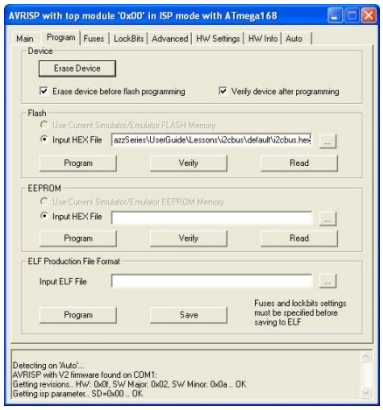



Fig. 2.4.3 : Interfaz de programación con puerto ISP

La programación serial a la cual se accede mediante la opción AVRProg hace uso de la aplicación de Bootloader precargada en el Butterfly para su programación. En este caso se utiliza el puerto serial del computador conectado al puerto UART del AVR Butterfly el cual posee un convertidor de nivel lo cual realiza la conversión de nivel de voltaje de RS-232 a un voltaje que es soportado por el módulo.

Para realizar el proceso de grabado se debe presionar el botón del Joystick en su posición central y en ese momento hacer click sobre la opción Tools ->AVRProg. Si se ha realizado la acción correctamente aparecerá la ventana de programación, caso contrario se mostrará un mensaje de error. Se debe notar que durante el proceso de programación el AVR Butterfly no muestra nada en su pantalla.

Al momento de realizar la programación el software realizar el borrado de la aplicación anteriormente cargada para programar la que nosotros hayamos seleccionado y luego realiza la verificación antes de abandonar

el modo de programación. Antes de desconectar el AVR Butterfly se debe hacer click sobre el botón EXIT del AVRProg para que se desconecte por completo y no vaya a causar algún daño al módulo por mala desconexión.

A pesar que al cargar una nueva aplicación al Butterfly se realiza el borrado del programa anterior la opción de Bootloader no se borra de modo que el módulo puede ser grabado varias veces por este mismo método.

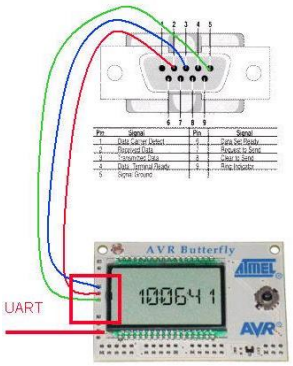



Fig: 2.4.4: Conexión del AVR Butterfly con puerto serial

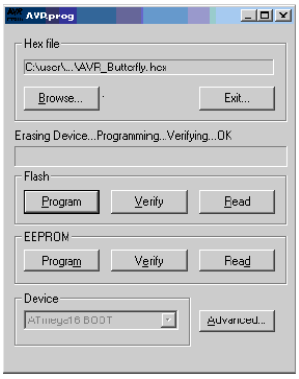



Fig. 2.4.5: Interfaz de programación con puerto serial

## 2.5 PROTEUS 7. 7

Proteus 7.7 es un software de simulación y diseño de PCB distribuido por LabcenterElectronics, que permite analizar el comportamiento de los circuitos, previo a la implementación física del circuito.

Provee de herramientas de edición, a su vez de librerías que contienen componentes análogos y familias de microcontroladores entre ellas Microchip, Motorola, ARM, AVR, y microprocesadores de la familia ATMEL, Motorola entre otros.

Una de las ventajas para el desarrollo del proyecto es que se hizo uso de los ejemplos para ATMEL en el cual encontramos el Kit AVR Butterfly funcionando con su programa demo, con lo cual no fue necesario el diseño del esquemático correspondiente.

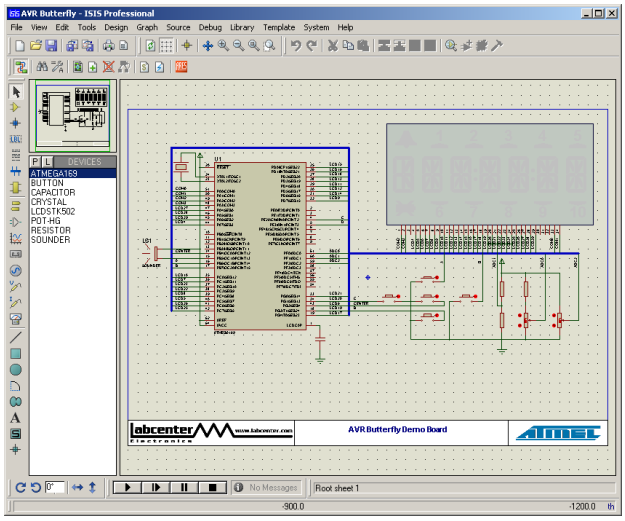



Fig. 2.5: Proteus 7.7

## 2.6 TRANSRECEIVER HM-TR 434

El módulo transreceiverHM-TR 434, que convierte la trama generada en el transmisor a radio frecuencia a través de una modulación FSK, en un rango de frecuencia de 433.92 MHZ, o seleccionable dependiendo de la configuración que se le asigne la frecuencia de operación.

Es un módulo transmisor y receptor inalámbrico pequeño que se puede alimentar con una fuente de 3-5V, y una tasa de velocidad desde300-19200bps, posee una antena que permite evitar interferencia que ocasionaría si se usara un cable.

Entre otras de sus ventajas es que posee un alcance mayor a 300m en áreas abiertas. Y si algún otro transmisor está operando a la misma frecuencia se puede variar la frecuencia para evitar interferencias.

El módulo HM-TR 434 se lo puede adquirir para operar directamente en lógica TTL y en lógica RS-232, que la única diferencia en el circuito es que tiene implementado el MAX232.

Para realizar la conexión correspondiente para un microcontrolador se lo conecta en forma cruzada. Es decir:

Si se requiere implementar el transmisor se conecta el Tx del transmisor al DRX del módulo RF, y para la realización correspondiente del receptor se conecta el Rx al DTX del módulo. Este tipo de comunicación es en una sola vía o Half Duplex.

Cabe recalcar que al habilitar transmisión y recepción en ambos módulos se realiza una comunicación Full- Duplex,

Otra de las características que posee es que permite el control de los módulos haciendo uso del pin enable el cual habilita el uso de ellos. Un ejemplo clásico es que si el microcontrolador transmite los datos y el

enable no está activo no se realiza la modulación correspondiente y no se transmite el dato.

Fig.2.6: TransreceiverHM-TR 434

# CAPÍTULO 3

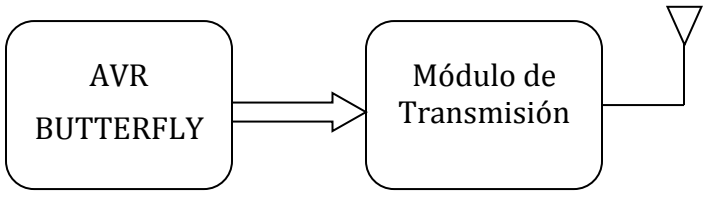
## DISEÑO E IMPLEMENTACION DEL SISTEMA

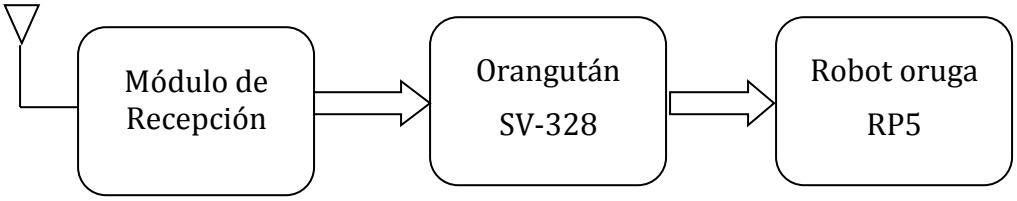
### 3.1 DIAGRAMA DE BLOQUES

El diagrama de bloques del sistema se encuentra constituido por la etapa del transmisor y la etapa del receptor.

La etapa de transmisión está constituida por el Kit AVR Butterfly que envía las tramas correspondientes generadas al presionar los botones del joystick. La señal ingresa al módulo de transmisión de RF, que se encarga de modular la señal para ser transmitida.

En la etapa de recepción se encuentra el módulo receptor que demodula la señal para convertirla en tramas que por medio del USART del orangután son procesadas generando de esa manera algún cambio en los motores del robot RP5.






## 3.2 TRANSMISOR

### 3.2.1 AVR BUTTERFLY

Mediante el uso del Joystick del AVR Butterfly se usa las interrupciones generadas por las mismas por cambio de estado en lo cual se procede a enviar una trama por medio de la comunicación USART presente en el ATMega169 la cual contiene un carácter que indica la opción que ha sido realizada por el usuario de la siguiente manera:

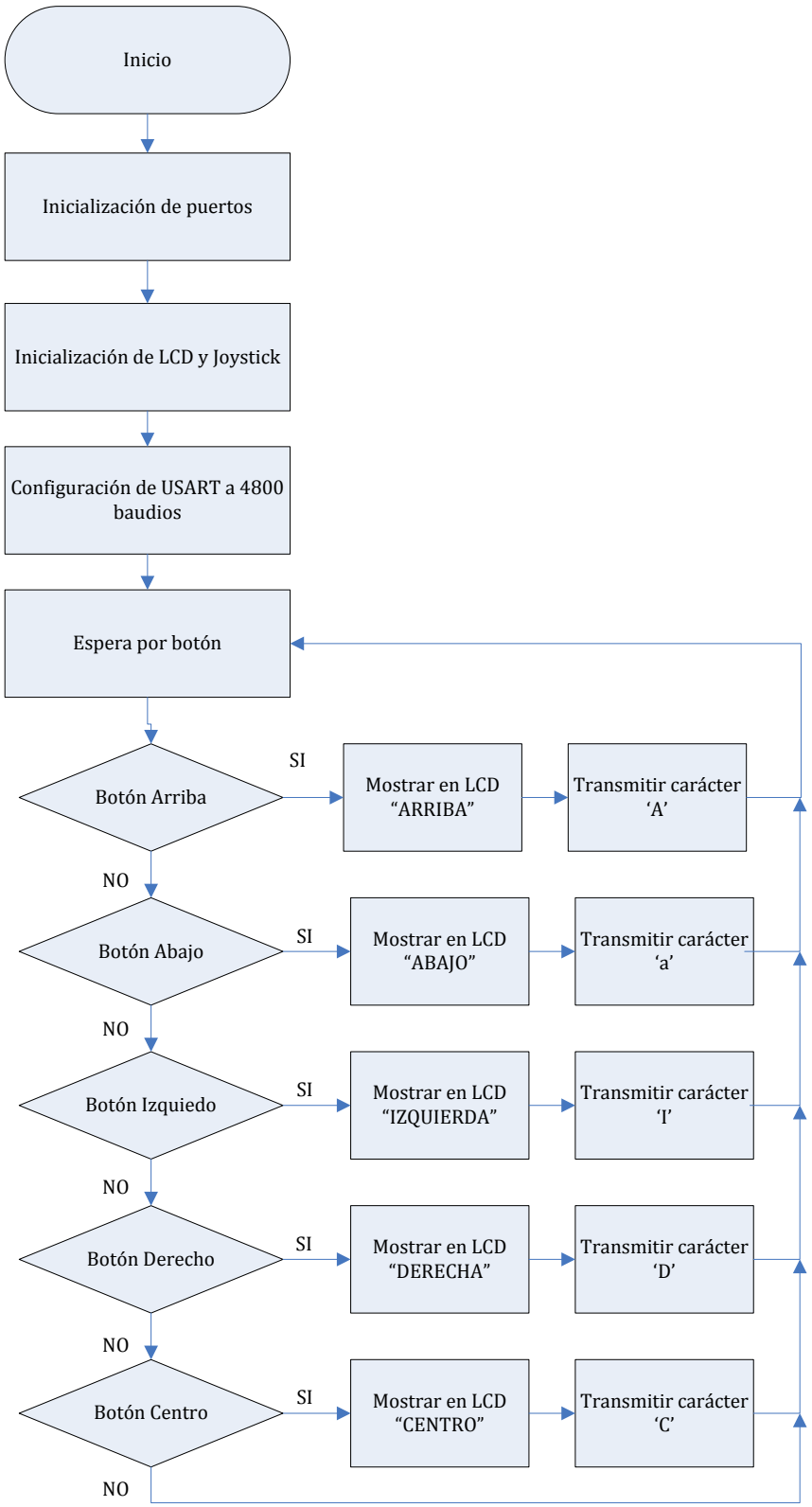
- A: Arriba
- a: Abajo
- D: Derecha
- I: Izquierda
- C: Centro

La trama a transmitir está compuesta de 8 bits y se le añade un bit de inicio y un bit de parada para indicarle al receptor tanto el inicio como final de la transmisión de una trama.

En la programación se usa el valor que viene configurado por defecto en el microcontrolador el cual es de 8 MHz y a su vez mediante el uso de un divisor de frecuencia se consigue obtener una frecuencia de 1 MHz la cual es usada para calcular el valor que se debe cargar en los registros de configuración de la transmisión USART para conseguir una velocidad de 4800 baudios.

Al mismo tiempo se hace uso del LCD presente en el AVR Butterfly para mostrar por pantalla la instrucción que ha sido elegida por el controlador del AVR Butterfly.

### 3.2.1.2 DIAGRAMA DE FLUJO DEL TRANSMISOR



### 3.2.1.3 CÓDIGO DEL AVR BUTTERFLY

```
//Declaración de constantes
#define  Centro    0
#define  Arriba    1
#define  Abajo     2
#define  Izquierda 3
#define  Derecha   4
#define  Otros     5

//Libreias a usar
#include<avr/io.h>
#include<avr/interrupt.h>
#include<avr/pgmspace.h>
#include<avr/delay.h>
#include<inttypes.h>

//Librerias a usar que no pertenecen al entorno
#include"mydefs.h"
#include"LCD_functions.h"
#include"LCD_driver.h"
#include"button.h"
#include"usart.h"

//Prototipo de funciones
int Obtener_Boton(void);

//Programa principal
int main(void)
{
        //Declaración de variable
int input;

        //Se habilita las interrupciones globales
        sei();

        //Se muestra un mensaje a través del LCD
PGM_P statetext = PSTR("AVR BUTTERFLY");

// Disable Analog Comparator (power save)
ACSR = (1<<ACD);

// Disable Digital input on PF0-2 (power save)
DIDR0 = (7<<ADC0D);

// Enable pullups
        PORTB = (15<<PB0);
        PORTE = (15<<PE4);
```

```
        // Initialize pin change interrupt on joystick
        Button_Init();

        // initialize the LCD
        LCD_Init();

        // set Clock Prescaler Change Enable
        CLKPR = (1<<CLKPCE);
// set prescaler = 8, Inter RC 8Mhz / 8 = 1Mhz
        CLKPR = (0<<CLKPS1) | (1<<CLKPS0);

        //Configuración del USART a 4800 baudios
        USART_Init(103.08);

        if (statetext)
        {
    LCD_puts_f(statetext, 1);
    LCD_Colon(0);
    statetext = NULL;
        }

        //Lazo infinito
while (1)
  {
if (statetext)
        {
                LCD_puts_f(statetext, 1);
                LCD_Colon(0);
                statetext = NULL;
        }

        //Se espera a que sea presionado un botón
        input = Obtener_Boton();

        //Se realiza una determinada acción según el botón presionado
        switch (input)
                {
        case Centro:
                statetext = PSTR("CENTRO");
                Usart_Tx('C');
        break;

        case Derecha:
        statetext = PSTR("DERECHA");
                Usart_Tx('D');
        break;

        case Izquierda:
                statetext = PSTR("IZQUIERDA");
        Usart_Tx('I');
```

```
		break;

		case Arriba:
			statetext = PSTR("ARRIBA");
			Usart_Tx('A');
		break;

		case Abajo:
			statetext = PSTR("ABAJO");
			Usart_Tx('a');
		break;

		default:
		break;
		}
}

return 0;
}

/*
Función que retorna un valor entero correspondiente al botón
presionado en el JoyStick
*/
int Obtener_Boton(void)
{
		int Temp1, Temp2;

		//PB4-->O  Centro
		//PB6-->A  Arriba
		//PB7-->B  Abajo
		//PE2-->C  Izquierda
		//PE3-->D  Derecha

		//Centro
		Temp1=(PINB) & 0b00010000;
		if(Temp1==0b00000000)
		{
			sei();
			return Centro;
		}

		//Arriba
		Temp1=PINB & 0b01000000;
		if(Temp1==0b00000000)
		{
			sei();
			return Arriba;
		}
```

```
        //Abajo
        Temp1=PINB & 0b10000000;
        if(Temp1==0b00000000)
        {
                sei();
                return Abajo;
        }

        //Izquierda
        Temp1=PINE & 0b00000100;
        if(Temp1==0b00000000)
        {
                sei();
                return Izquierda;
        }

        //Derecha
        Temp1=PINE & 0b00001000;
        if(Temp1==0b00000000)
        {
                sei();
                return Derecha;
        }

        sei();
        return Otros;
}
```

### 3.2.2 RECEPTOR

#### 3.2.2.1 MÓDULO ORANGUTAN SV-328

El módulo Orangután SV-328 es el encargado de recibir las instrucciones, mediante radiofrecuencia, del AVR Butterfly las cuales son usadas para realizar las funciones elegidas por el operador y hacer funcionar al robot oruga según las mismas.

Las instrucciones captadas por el módulo receptor son enviadas al pin de Recepción del microcontrolador ATMega328P. Dicha recepción está configurada para recibir datos a una tasa de 4800 baudios para que los datos del transmisor sean interpretados correctamente.

Las instrucciones van a actuar directamente sobre los motores del robot RP5 mediante PWM (modulación por ancho de pulso). Dichos valores están comprendidos entre -255 y 255, siendo ambas las máximas velocidad según el sentido de rotación.

Al igual que en el AVR Butterfly las instrucciones a realizar serán mostradas a través del LCD presente en el módulo Orangután para una mejor apreciación de las ordenes que va a realizar.

Dado que los niveles de corriente que normalmente suministran los microcontroladores se encuentran comprendidos en el orden de miliamperios se hace el uso de driver para motores TB6612FNG capaz de suministrar un valor máximo de 3 A por canal lo cual es posible gracias al arreglo de 6 baterías que usa el Orangután.

Dado que la demanda de corriente será mayor en función de la dificultad de movimiento de las orugas no se debe permitir que el robot quede atascado, caso contrario por el exceso de corriente demandada se pueden dañar los elementos del módulo.

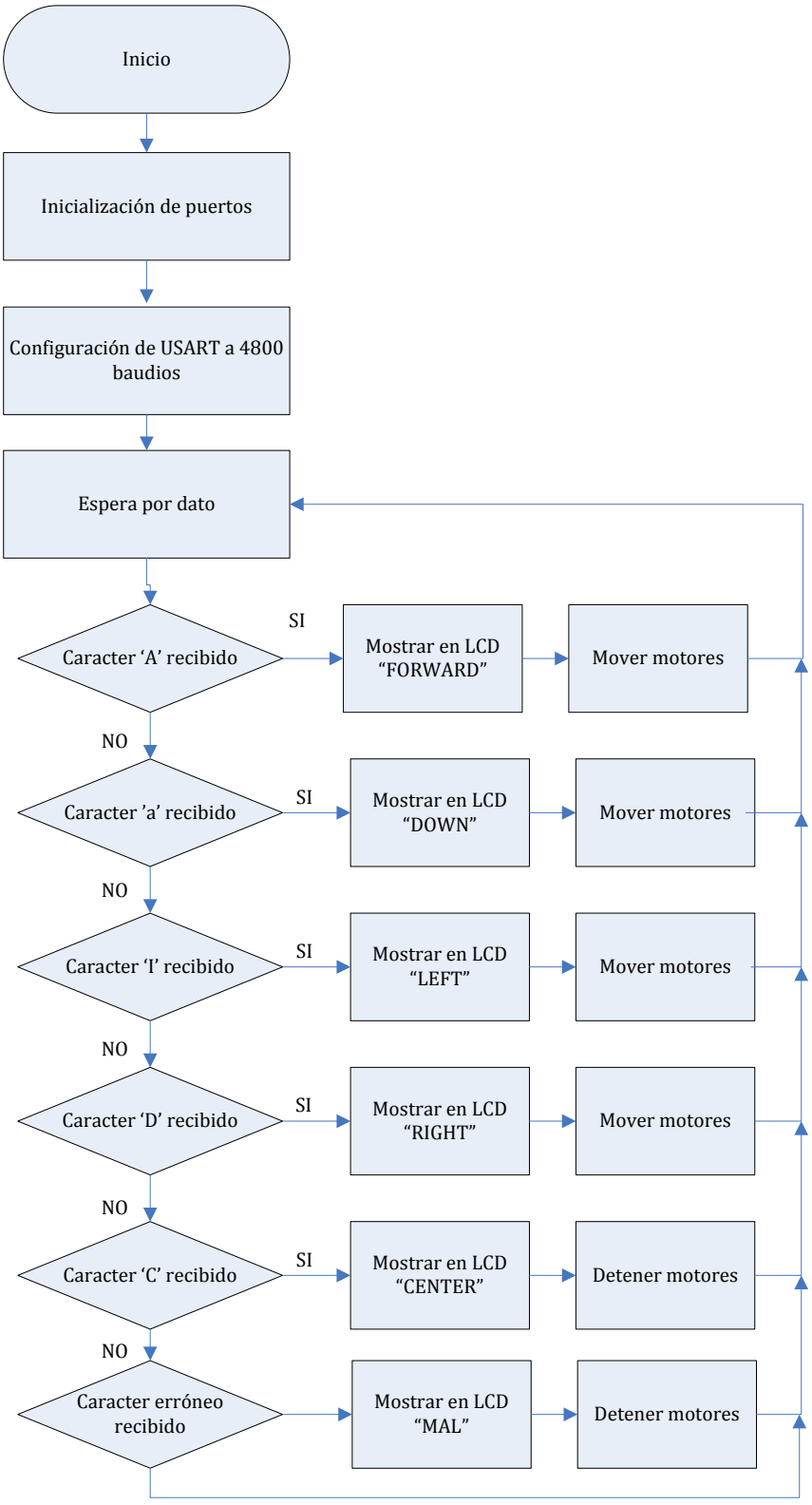
#### 3.2.2.2 ROBOT ORUGA RP5

El robot oruga RP5 es un chasis sin ninguna electrónica en su interior al que, mediante el uso de diferentes módulos, se pueden controlar los motores DC que posee.

Dichos motores poseen un grupo de engranaje que van acoplados en su extremo final a unas llantas que están cubiertas por una banda de caucho que le permite obtener una mayor tracción y por ende movilizarse por terrenos que no son del todo lisos o que presentan una determinada inclinación, siendo la única limitación la fricción de la oruga con respecto a su superficie.

El módulo oruga cuenta con una base para seis pilas AA las cuales son las encargadas de alimentar tanto al circuito de control como a los motores, motivo por el cual se debe contar con unas baterías capaces de suministrar el nivel de corriente necesario, caso contrario al ser menor la corriente suministrada a la requerida el módulo Orangután se reinicia.

### 3.2.2.3 DIAGRAMA DE FLUJO DEL RECEPTOR



### 3.2.2.4 CÓDIGO DEL ORANGUTÁN SV-328

```
//Declaración de constantes
#define F_CPU 20000000UL
#define BAUD 4800
#define MYUBRR F_CPU/16/BAUD-1
#define PIND_MASK ((1<<PIND0)|(1<<PIND1))

//Librerias a usar
#include<avr/pgmspace.h>
#include<pololu/orangutan.h>

//Prototipo de procedimientos
void USART_Init( unsignedint );
unsignedchar ReceiveByte (void);

//Programa principal
int main ()
{
	//Declaración de variable
	int i;

	//Seteo de puertos
	DDRD = 0xFE;
	PORTD |= PIND_MASK;
	DDRB = 0x08; // set PORTD for output
	PORTB = 0x00; // set LEDs off

	//Configuración de USART a 2400 baudios
	USART_Init(520);

	//Lazo infinito en espera de caracter recibido
		while(1)
		{
			//Se muestra en pantalla un mensaje que indica que
			//se está a la espera de recibir un caracter
			clear();
			print("WAITING");
			lcd_goto_xy(0, 1);
			print("ORDER");
			delay_ms(500);

			i = ReceiveByte();

			//Se toma la decisión en función del caracter
			//recibido. El arranque del motor se lo realiza
			//en dos tiempos para no realizar una demanda súbita
			//de potencia
```

```
//ADELANTE
if(i == 'A')
{
        clear();
        print("FORWARD");
        set_motors(-50,50);
        delay_ms(1000);
        set_motors(-100,100);
        delay_ms(1000);

}

//ATRÁS
elseif (i == 'a')
{
        clear();
        print("BACK");
        set_motors(50,-50);
        delay_ms(1000);
        set_motors(100,-100);
        delay_ms(1000);

}

//IZQUIERDA
elseif (i == 'I')
{

        clear();
        print("LEFT");
        set_motors(50,50);
        delay_ms(1000);
        set_motors(100,100);
        delay_ms(1000);


}

//DERECHA
elseif (i == 'D')
{

        clear();
        print("RIGHT");
        set_motors(-50, -50);
        delay_ms(1000);
        set_motors(-100, -100);
        delay_ms(1000);
}
```

```
                        //ALTO
                        elseif (i == 'C')
                        {
                                clear();
                                print("Stop");
                                set_motors(0,0);
                                delay_ms(250);
                        }
                }
        return 0;
}


//Implementación de procedimiento

/*
        Se configuran los registros para la transmisión por USART
        de manera que el dato recibido indica el BAUDRATE al cual
        se va a trabajar
*/

void USART_Init(unsignedint baudrate)
{
// Set baud rate
  UBRR0H = (unsignedchar)(baudrate>>8);
UBRR0L = (unsignedchar)baudrate;
        //UCSR0A = (0<<U2X0);

// Enable receiver and transmitter
UCSR0B = (1<<RXEN0)|(1<<TXEN0);

// Async. mode, 8N1
UCSR0C = (1<<USBS0)|(3<<UCSZ00);
}

//Implementación de función
/*
        Función que espera a la recepción de un caracter y luego
        retorna el dato recibido al programa principal
*/
unsignedchar ReceiveByte (void)
{
/* Wait for incomming data */
while (!(UCSR0A & (1 << RXC0)));

/* Return the data */
return UDR0;
}
```

# CAPÍTULO 4

## 4 SIMULACIÓN Y PRUEBAS DEL PROYECTO

En este capítulo se presentan las simulaciones correspondientes del transmisor y del receptor que conforman el proyecto que se desarrolló, a su vez se presentan pruebas que fueron realizadas para la optimización de la transmisión RF.

### 4.1 SIMULACIÓN DEL TRANSMISOR

El transmisor está constituido por el ATMEGA169 el LCD STK502 y los PUSHBUTTONS que representan el joystick del KIT AVR BUTTERFLY.

Al presionar algunos de los pushbuttons se escribe en el LCD la instrucción que se ejecutó, y se envía el dato al módulo transmisor HM-TR434 el cual se encarga de realizar la conversión del dato a una señal RF correspondiente.

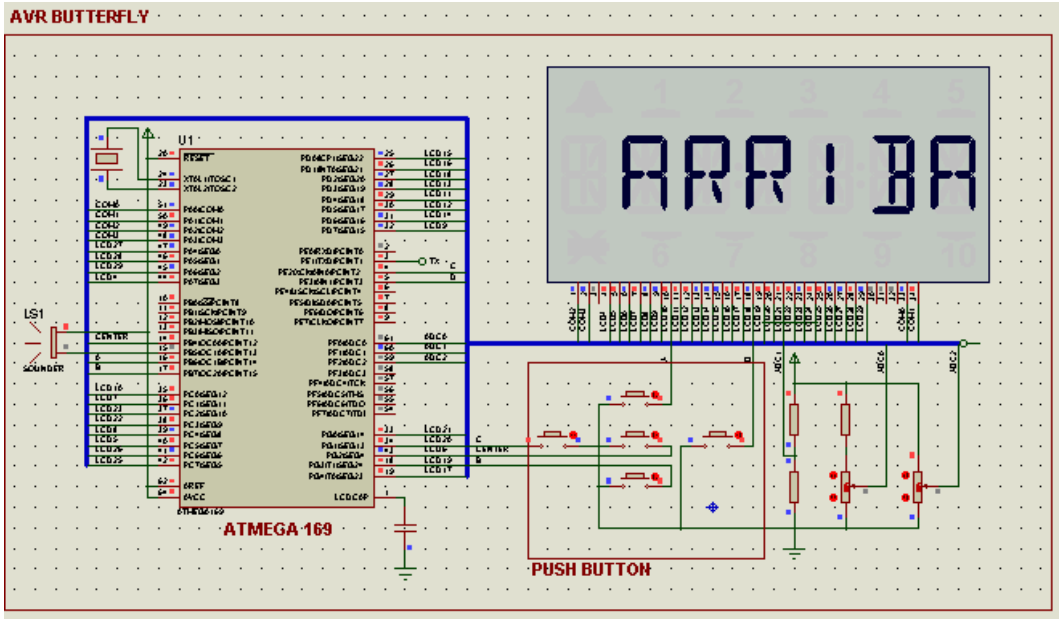



Fig. 4.1.1: Transmisor mostrando instrucción ha sido ejecutada.

Para observar que carácter enviado se usó la herramienta de Proteus Virtual Terminal con el cual se observa que carácter se envía y se configura a la velocidad de transmisión de datos.

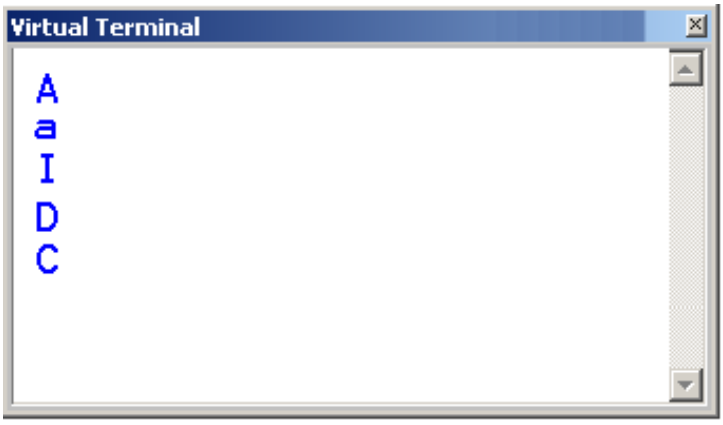



Fig. 4.1.2 Visualización de los caracteres de transmisión.

## 4.2 SIMULACIÓN DEL RECEPTOR

El receptor está constituido por el ATMEGA 328P, el LCD DE 16 x2 ya que en las librerías de proteus no se encuentra un LCD de 8x2, el cual está configurado para trabajar con 8 bits de datos. Para el caso del proyecto en el LCDse muestran solo 8 caracteres. A su vez para realizar la simulación de los motores se usó el driver L293D, como reemplazo del driver TB6612FNG que se encuentra implementado en el Orangután.

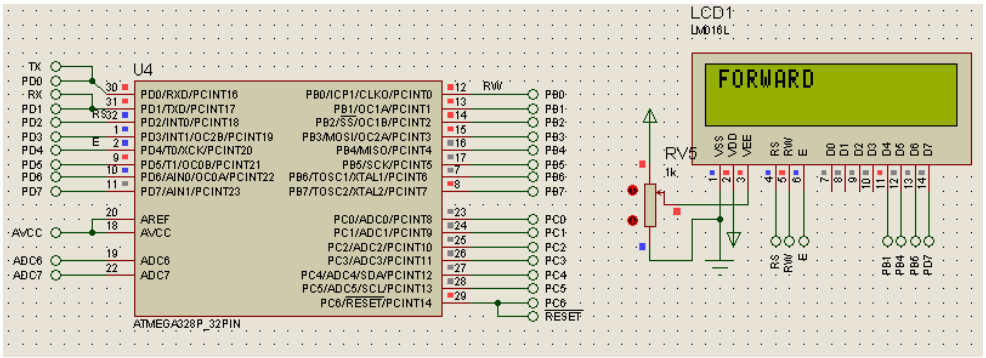



Fig. 4.2.1: Receptor ejecutando instrucción que recibe.

Después de que el dato es recibido el Orangután ejecuta el movimiento de los motores, debido a que para la conexión entre el microcontrolador y los motores se necesita de un driver para controlarlos, en la simulación se usó un L293D que suministra la corriente necesaria para una simulación eficiente.

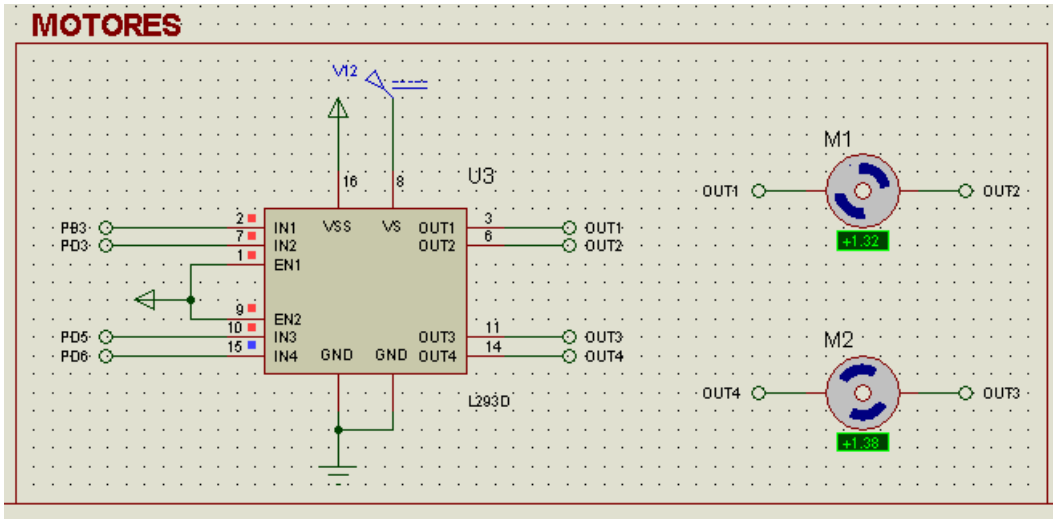



Fig. 4.2.2: Motores ejecutando el desplazamiento hacia adelante.

Para que el robot se mueva hacia atrás al recibir el carácter específico los motores giran en el sentido contrario permitiendo de esa manera que el robot retroceda.

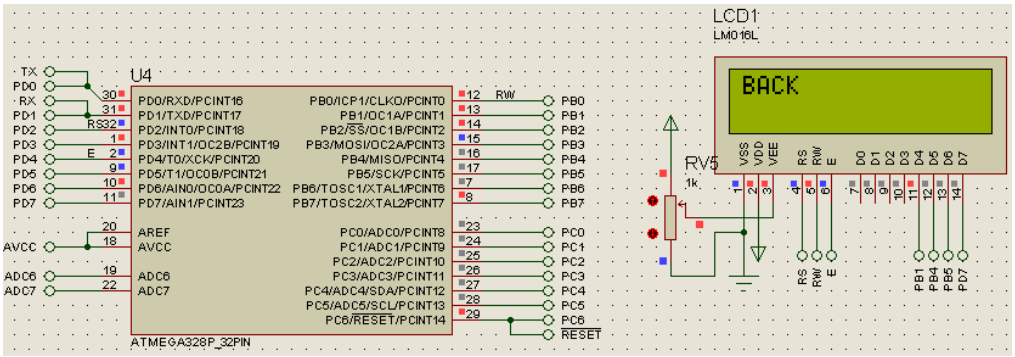



Fig. 4.2.3: Receptor mostrando la instrucción hacia atrás.

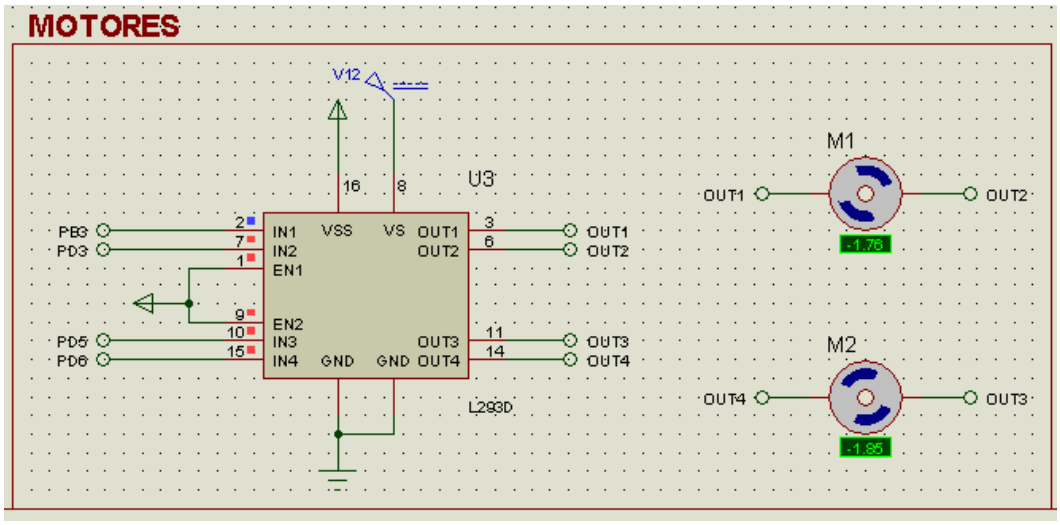



Fig. 4.2.3: Motores ejecutando el desplazamiento hacia atrás.

Para realizar los giros correspondientes se debe presionar la dirección a la cual se desea girar presionando la correspondiente botonera, y el giro consiste en girar un motor en sentido horario y el otro motor en sentido anti horario.

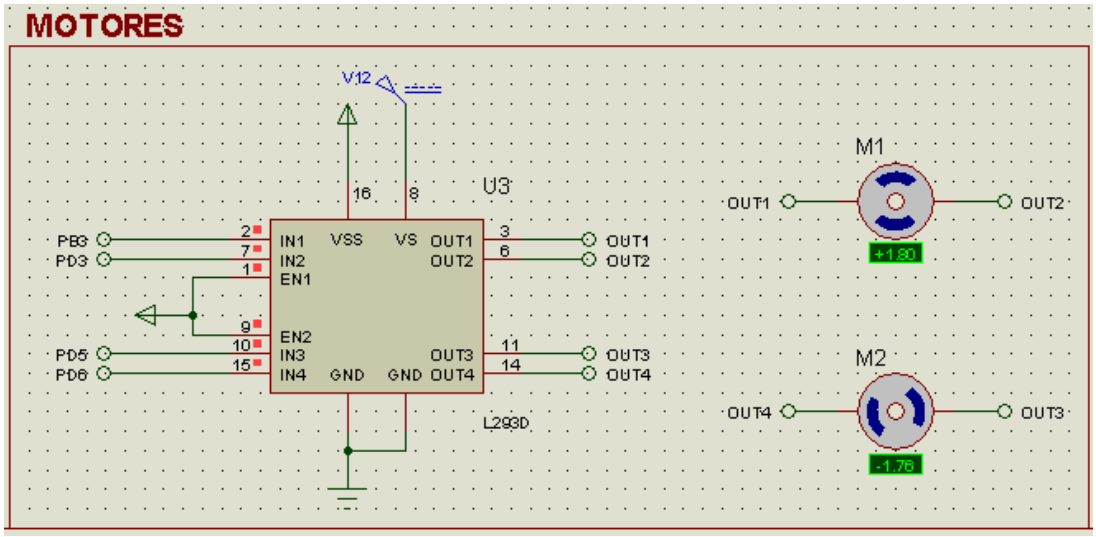



Fig. 4.2.4: Motores ejecutando el giro hacia la derecha.

Por último el botón central del joystick permite parar el movimiento de los motores.

## 4.3 RESULTADOS DE LA SIMULACIÓN

De los datos de simulación se pudo obtener una perspectiva del funcionamiento del proyecto, debido a que en cada una de las simulaciones se obtuvo el dato correspondiente desde el mensaje que se muestra en el LCD indicando que botón había sido presionado y el correspondiente carácter que se podía apreciar en el Virtual Terminal.

Para simular la comunicación se hizo uso del COMPIM que, al ser una interfaz con un puerto serial físico asignándole dos puertos COM virtuales, se crea un canal de comunicación.

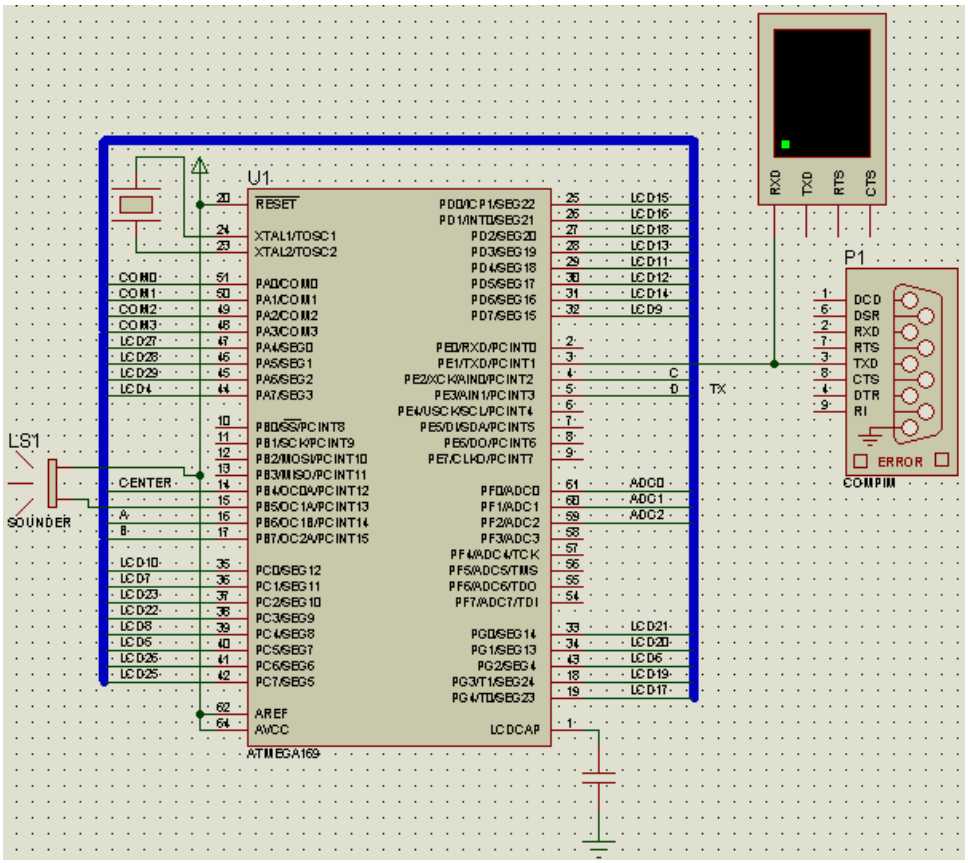



Fig 4.3.1 Esquema del transmisor usando COMPIM

## 4.4 PRUEBAS DEL PROYECTO

Debido a que en la simulación solo se observa el comportamiento virtual del circuito, y al interpretar los datos obtenidos y comprobarlos físicamente.

Al realizar la correspondiente conexión del Kit AVR Butterfly con el computador para comprobar que los caracteres asignados a cada movimiento del joystick sean enviados y se puedan observar en el Virtual Terminal del PC.

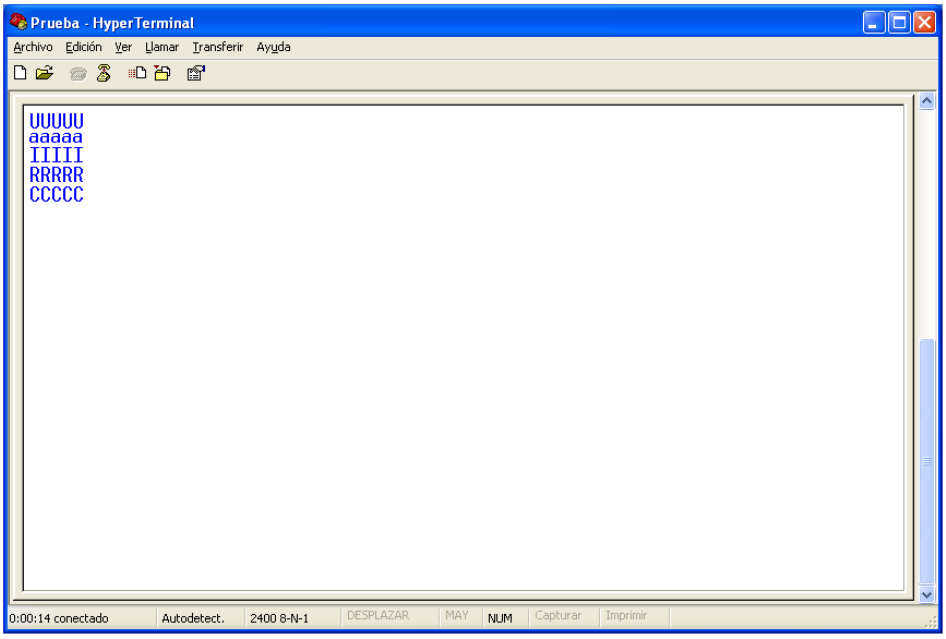



Fig. 4.4.1 Hyper Terminal mostrando caracteres recibidos.

## 4.5 IMÁGENES DEL PROYECTO

En la etapa del transmisor podemos observar el módulo AVR Butterfly como parte central del proyecto que está conectado a una tarjeta la cual se encarga de transformar el nivel de la señal RS232 del Butterfly a un nivel TTL para que pueda ser transmitido a través del módulo HM-TR 434. A su vez dicha tarjeta posee una etapa de regulación de voltaje para alimentar tanto al MAX232 como al módulo RF.

En la etapa del receptor podemos observar el módulo Orangután SV-328 conectado a una pequeña tarjeta la cual sirve de base para el módulo HM-TR 434 y la cual es alimentada a través de las salidas de voltaje del Orangután. Ambas tarjetas van montadas sobre el robot oruga RP5.


Fig. 4.5.1: Imagen de la etapa de transmisión

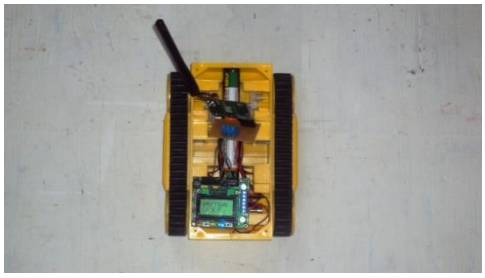


Fig. 4.5.2: Imagen de la etapa de recepción

Fig. 4.5.3: Robot oruga subiendo un obstáculo

Fig. 4.5.4: Robot oruga girando en superficie inclinada

# CONCLUSIONES

1. Con respecto al diseño del proyecto, se puede observar los útil que son los módulos desarrollados en base a microcontroladores ya que los mismos pueden realizar una gran gamma de funciones mediante el control de periféricos según la programación que éste contenga.
2. El uso de drivers capaces de manejar niveles de corriente que un microcontrolador no puede suministrar facilita mucho el desarrollo de proyectos de control de motores como son el control de un robot ya que todas las instrucciones son realizadas por el microcontrolador y el mismo se encarga de enviar las señales a los diferentes drivers para que puedan suministrar el nivel de potencia necesario para dicha labor.
3. El uso de módulos de radiofrecuencia permite realizar un control de manera remota de dispositivos de interés los cuales de modo que se evita tener que realizar dicho control a través de cableado.
4. Se puede observar que el uso de motores como las mismas alimentaciones del circuito, al ser de diferentes niveles de voltaje, causan la presencia de un nivel de ruido el cual puede afectar la recepción y generar comportamientos inesperados.
5. Verificar el nivel de voltaje de la fuente a utilizar para energizar los módulos antes de realizar la programación, caso contrario existe la posibilidad de que los módulos queden inutilizables.
6. Diseñar un circuito de alimentación apropiado al momento de realizar una transmisión inalámbrica de modo que el transmisor tenga la potencia suficiente para realizar dicha acción y que no consuma la energía del módulo de modo que acorte la duración de su funcionamiento.
7. Observar que los niveles de voltaje de los datos a recibir sean aceptados por el transmisor, caso contario se puede conseguir una

transmisión errónea o el transmisor podría no interpretar los datos y por ende no realizar la transmisión.

8. Configurar la velocidad de transmisión de datos tanto en el transmisor como en el receptor de modo que puedan ser interpretados de manera correcta en el destino y realizar las acciones que se desean.

9. Los datos transmitidos mediante radio frecuencia presentan interferencias debido al medio de transmisión, pero el nivel de recepción dependerán de los módulos ya que dentro de sus especificaciones se encuentran parámetros como tipo de modulación y la ganancia que poseen los mismos.

# RECOMENDACIONES

1. Con respecto al diseño del proyecto, una de las recomendaciones más importantes es realizar la adecuada búsqueda de información y conocer tanto las ventajas como limitaciones que poseen los elementos a usar para tomar las debidas precauciones en su manejo.
2. Al realizar una transmisión inalámbrica revisar que tanto el voltaje de entrada como el dato que se está recibiendo posean una configuración que se encarguen de disminuir o eliminar el ruido presente de modo que el funcionamiento de los dispositivos sea el adecuado.
3. Al trabajar con motores se recomienda no realizar un trabajo a velocidades altas partiendo desde el reposo porque eso genera un gran consumo de potencia durante el arranque de modo que es recomendable ir aumentando la velocidad de manera gradual para un uso más eficiente de la energía.
4. Tener presente que al realizar la conexión de circuitos que trabajen a diferentes niveles de voltaje se debe realizar una correcta referencia entre los circuitos a través de GND de modo que se eviten lecturas erróneas por el uso de diferentes referencias o en el peor de los casos un daño de los módulos y elementos utilizados.
5. Asegurarse que los niveles de energía de los módulos son los adecuados antes de realizar la programación caso contrario se podría afectar a los mismos.
6. Para la programación del AVR Butterfly es recomendable que se use el programador AVR Pololu, para evitar programarlo mediante el puerto serial haciendo uso del bootloader, ya que se podría afectar el joystick del kit avr butterfly.

# ANEXOS

# ANEXO I

# ESPECIFICACIONES DEL MICROCONTROLADOR ATMEGA169

## Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 130 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz
  - On-Chip 2-cycle Multiplier
- Non-volatile Program and Data Memories
  - 16K bytes of In-System Self-Programmable Flash
    Endurance: 10,000 Write/Erase Cycles
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - 512 bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 1K byte Internal SRAM
  - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - 4 x 25 Segment LCD Driver
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Universal Serial Interface with Start Condition Detector
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Five Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, and Standby
- I/O and Packages
  - 53 Programmable I/O Lines
  - 64-lead TQFP and 64-pad QFN/MLF
- Speed Grade:
  - ATmega169V: 0 - 4 MHz @ 1.8 - 5.5V, 0 - 8 MHz @ 2.7 - 5.5V
  - ATmega169: 0 - 8 MHz @ 2.7 - 5.5V, 0 - 16 MHz @ 4.5 - 5.5V
- Temperature range:
  - -40°C to 85°C Industrial
- Ultra-Low Power Consumption
  - Active Mode:
    1 MHz, 1.8V: 350µA
    32 kHz, 1.8V: 20µA (including Oscillator)
    32 kHz, 1.8V: 40µA (including Oscillator and LCD)
  - Power-down Mode:
    0.1µA at 1.8V

# 8-bit AVR® Microcontroller with 16K Bytes In-System Programmable Flash

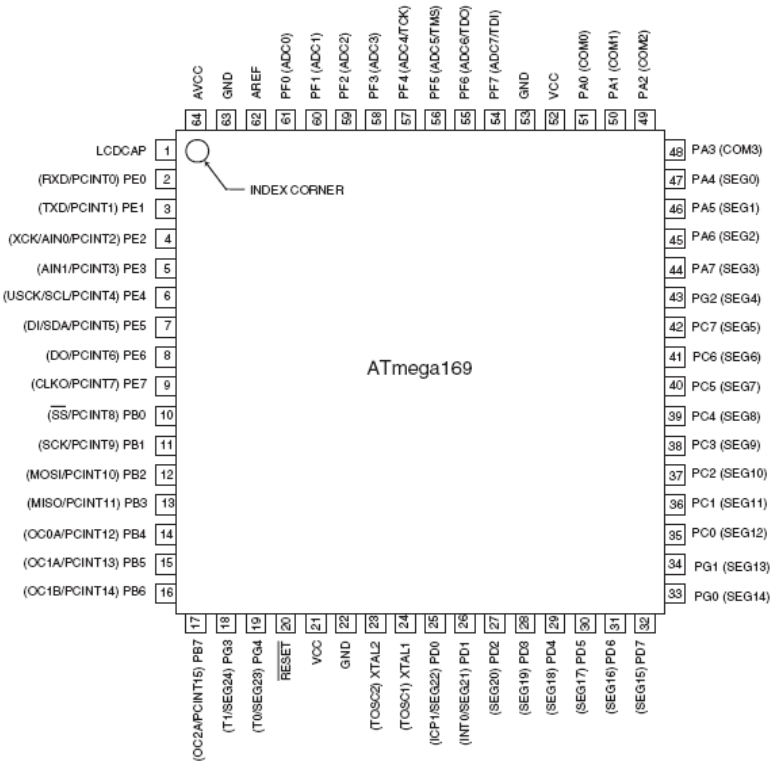
## ATmega169V
## ATmega169

Notice:
Not recommended in new designs.

2514P–AVR–07/06

## Pin Configurations

**Figure 1.** Pinout ATmega169




Note: The large center pad underneath the QFN/MLF packages is made of metal and internally connected to GND. It should be soldered or glued to the board to ensure good mechanical stability. If the center pad is left unconnected, the package might loosen from the board.

## Disclaimer

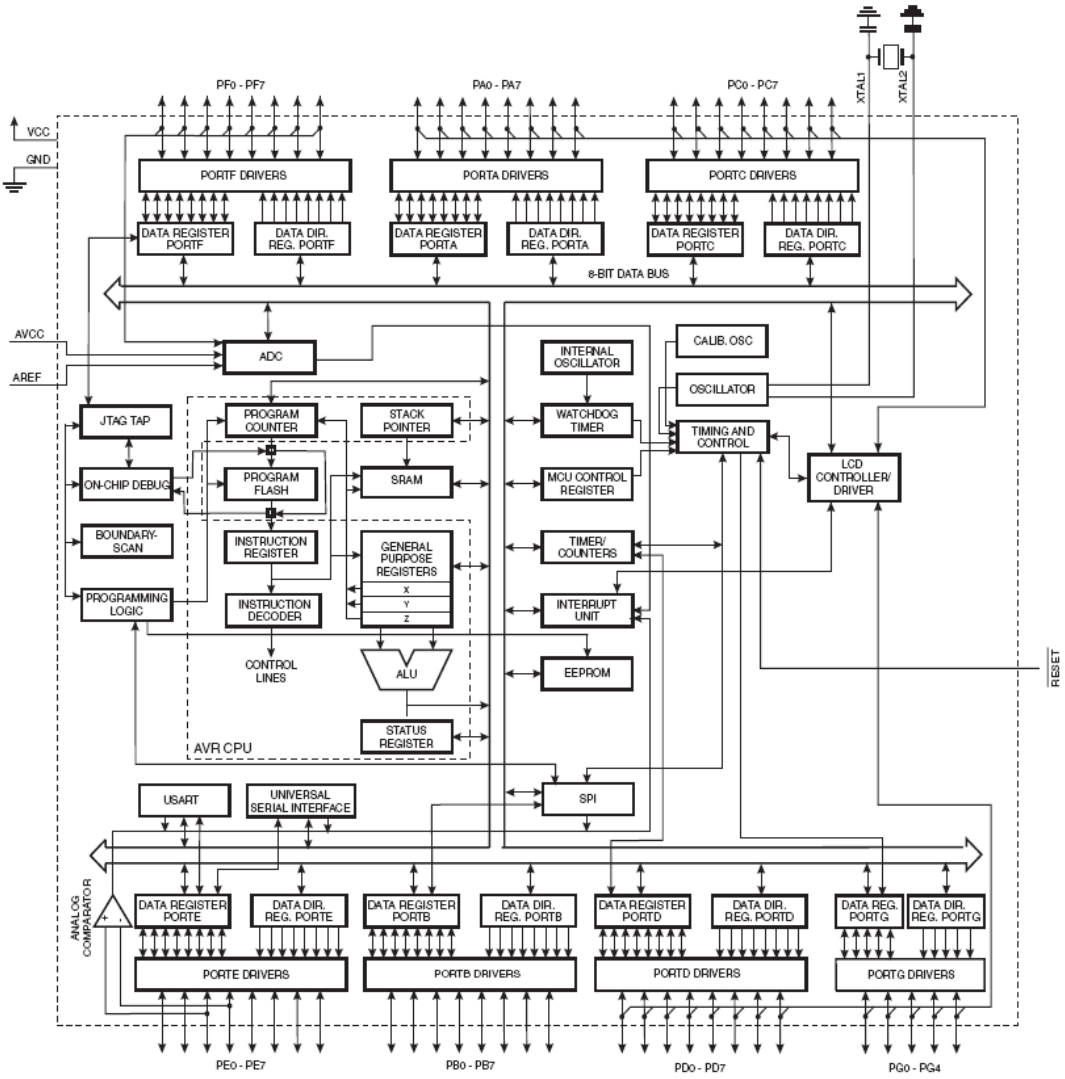
Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

## Overview

The ATmega169 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega169 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

## Block Diagram

**Figure 2.** Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega169 provides the following features: 16K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512 bytes EEPROM, 1K byte SRAM, 54 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, On-chip Debugging support and programming, a complete On-chip LCD controller with internal step-up voltage, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, Universal Serial Interface with Start Condition Detector, an 8-channel, 10-bit ADC, a programmable Watchdog Timer with internal Oscillator, an SPI serial port, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer and the LCD controller continues to run, allowing the user to maintain a timer base and operate the LCD display while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer, LCD controller and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption.

The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega169 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega169 AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

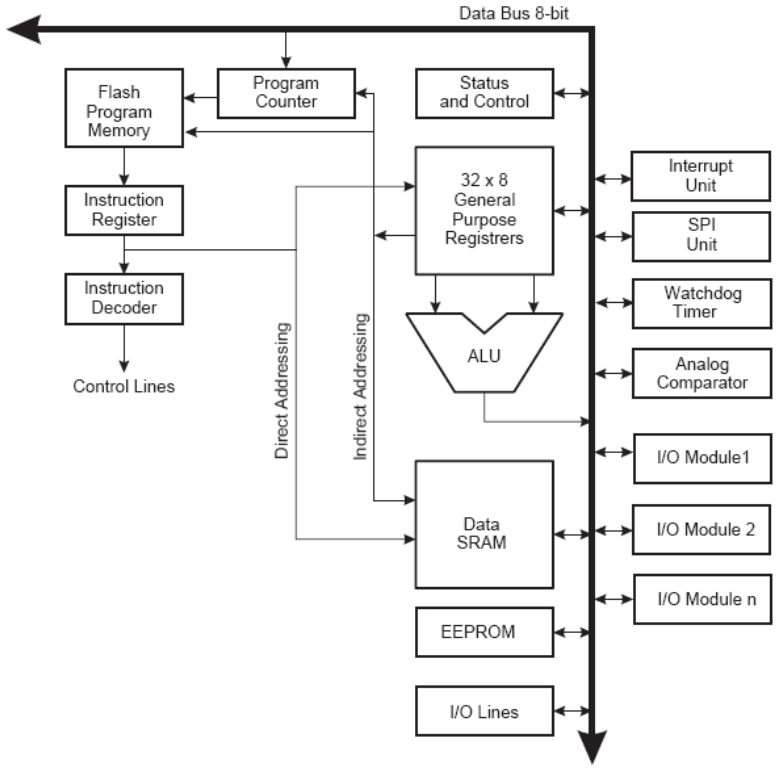
# AVR CPU Core

## Introduction

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

## Architectural Overview

**Figure 3.** Block Diagram of the AVR Architecture



# ANEXO II

# HOJA DE ESPECIFICACIONES DEL MICROCONTROLADOR

# ATMEGA328P

## Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash progam memory (ATmega48PA/88PA/168PA/328P)
  - 256/512/512/1K Bytes EEPROM (ATmega48PA/88PA/168PA/328P)
  - 512/1K/1K/2K Bytes Internal SRAM (ATmega48PA/88PA/168PA/328P)
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C[1]
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
    - Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
    - Temperature Measurement
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Byte-oriented 2-wire Serial Interface (Philips $I^2C$ compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
  - 1.8 - 5.5V for ATmega48PA/88PA/168PA/328P
- Temperature Range:
  - -40°C to 85°C
- Speed Grade:
  - 0 - 20 MHz @ 1.8 - 5.5V
- Low Power Consumption at 1 MHz, 1.8V, 25°C for ATmega48PA/88PA/168PA/328P:
  - Active Mode: 0.2 mA
  - Power-down Mode: 0.1 µA
  - Power-save Mode: 0.75 µA (Including 32 kHz RTC)

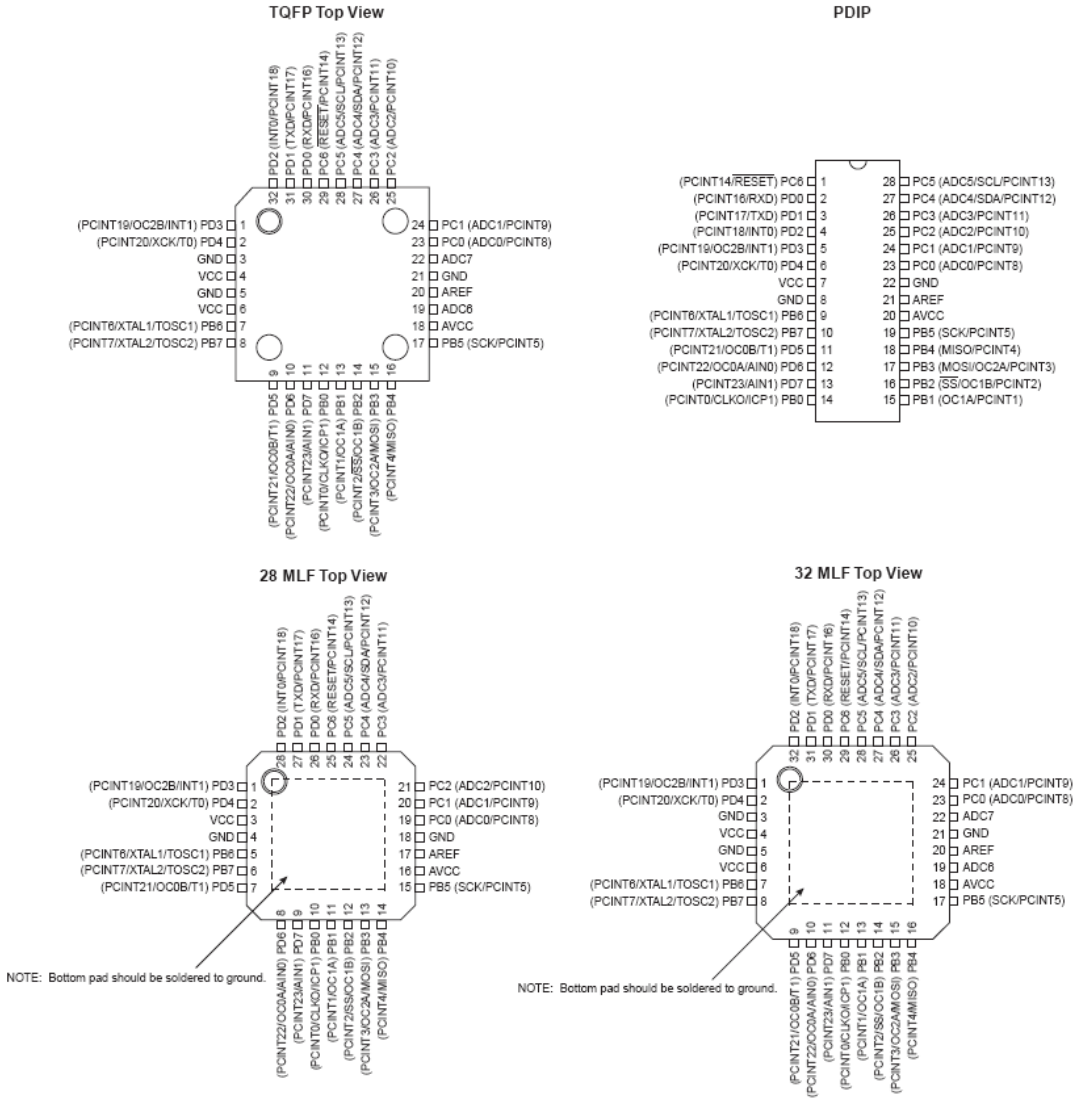
# 8-bit AVR® Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash

## ATmega48PA
## ATmega88PA
## ATmega168PA
## ATmega328P

Rev. 8161D–AVR–10/09

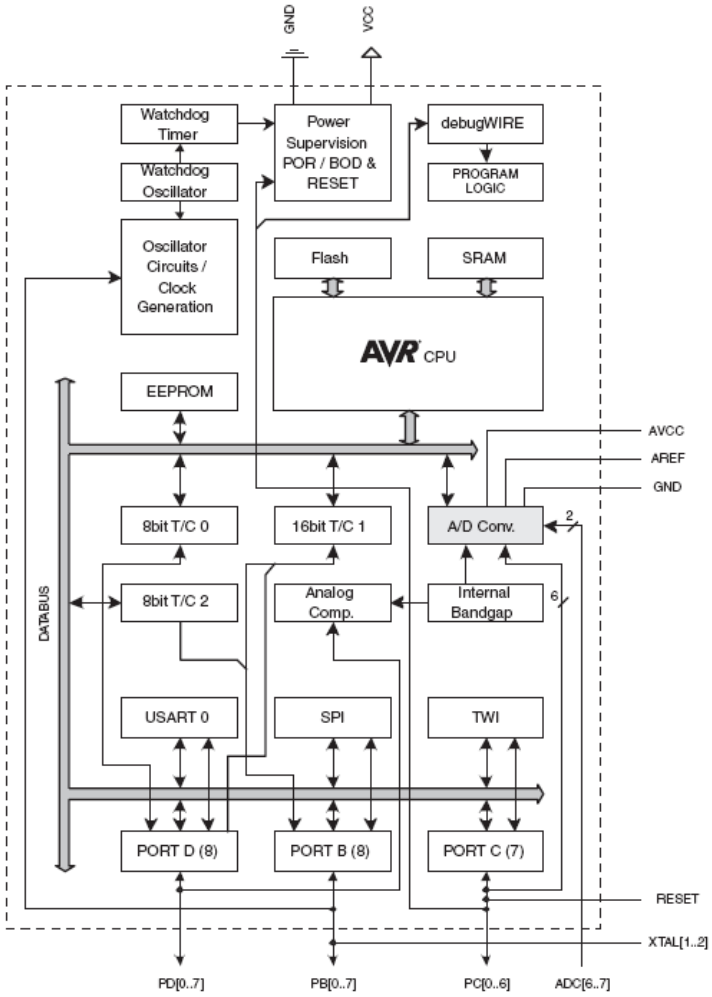
# 1. Pin Configurations

**Figure 1-1.** Pinout ATmega48PA/88PA/168PA/328P



## 2.1 Block Diagram

**Figure 2-1.** Block Diagram




The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting

architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega48PA/88PA/168PA/328P provides the following features: 4K/8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 256/512/512/1K bytes EEPROM, 512/1K/1K/2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable Watchdog Timer with internal Oscillator, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega48PA/88PA/168PA/328P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega48PA/88PA/168PA/328P AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

## 2.2 Comparison Between ATmega48PA, ATmega88PA, ATmega168PA and ATmega328P

The ATmega48PA, ATmega88PA, ATmega168PA and ATmega328P differ only in memory sizes, boot loader support, and interrupt vector sizes. Table 2-1 summarizes the different memory and interrupt vector sizes for the three devices.

**Table 2-1.** Memory Size Summary

| Device | Flash | EEPROM | RAM | Interrupt Vector Size |
|---|---|---|---|---|
| ATmega48PA | 4K Bytes | 256 Bytes | 512 Bytes | 1 instruction word/vector |
| ATmega88PA | 8K Bytes | 512 Bytes | 1K Bytes | 1 instruction word/vector |
| ATmega168PA | 16K Bytes | 512 Bytes | 1K Bytes | 2 instruction words/vector |
| ATmega328P | 32K Bytes | 1K Bytes | 2K Bytes | 2 instruction words/vector |

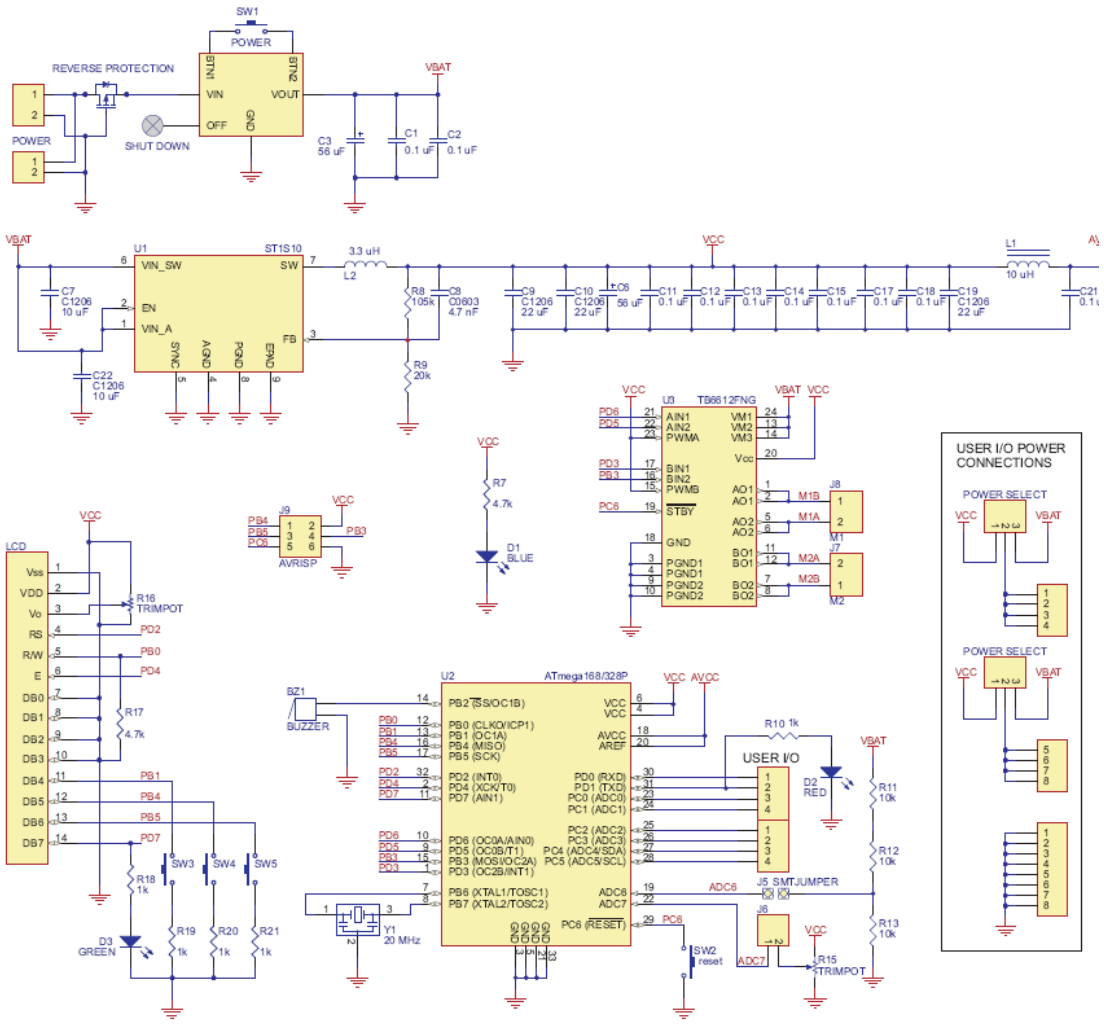
ATmega88PA, ATmega168PA and ATmega328P support a real Read-While-Write Self-Programming mechanism. There is a separate Boot Loader Section, and the SPM instruction can only execute from there. In ATmega48PA, there is no Read-While-Write support and no separate Boot Loader Section. The SPM instruction can execute from the entire Flash.

# ANEXO III

# DIAGRAMA ESQUEMÁTICO DEL MÓDULO

# ORANGUTÁN SV-328

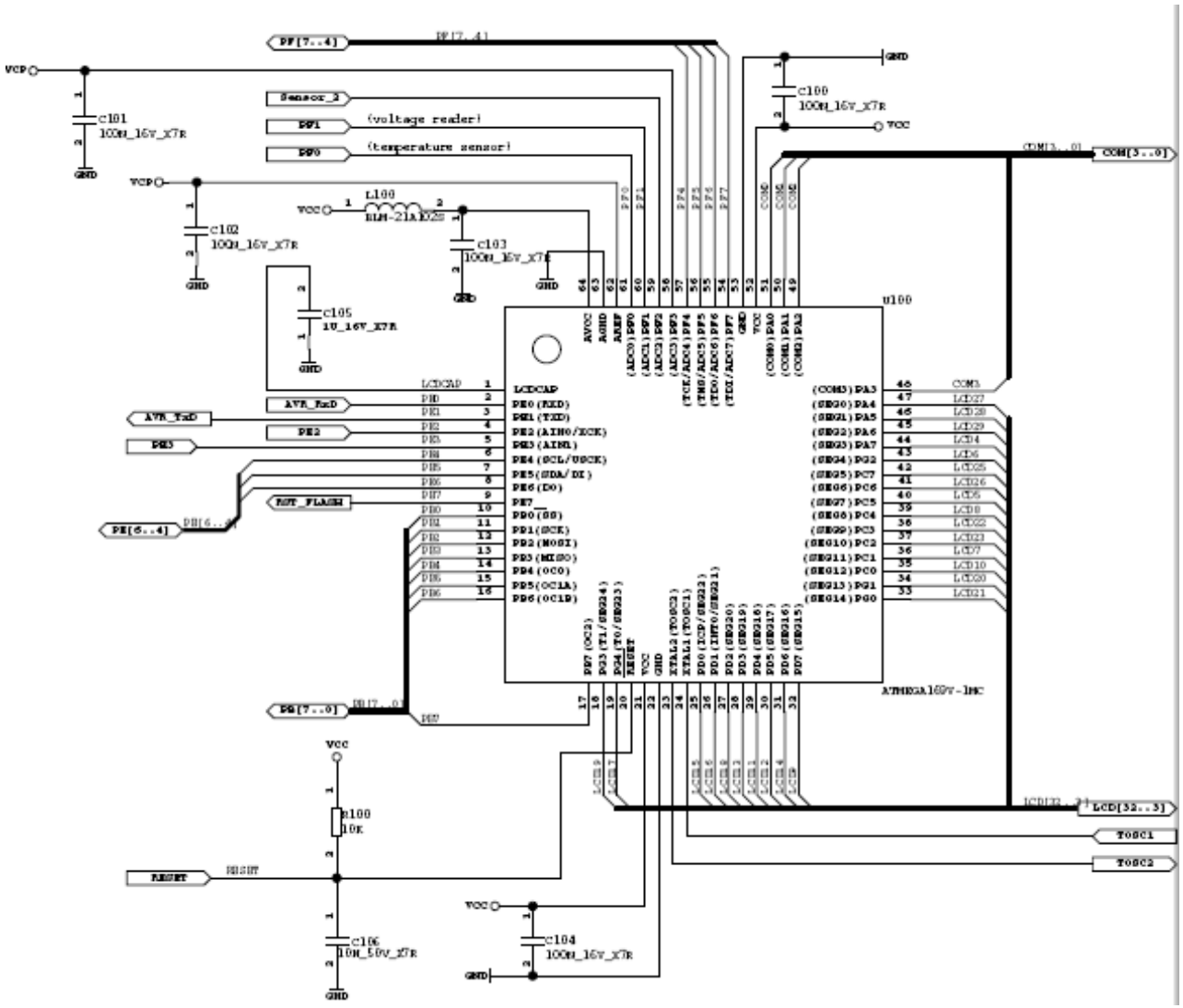
# Orangutan SV-168/SV-328 Schematic Diagram

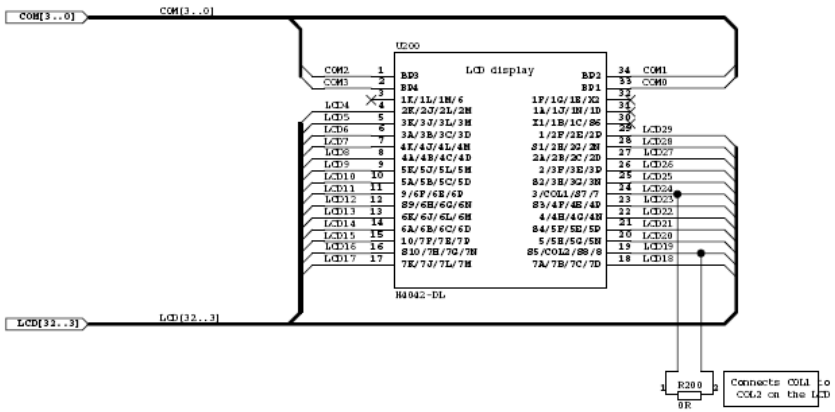


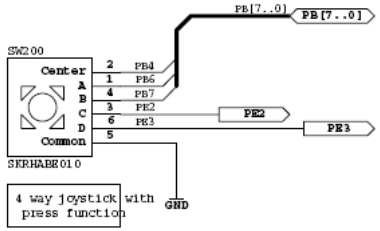
# ANEXO IV

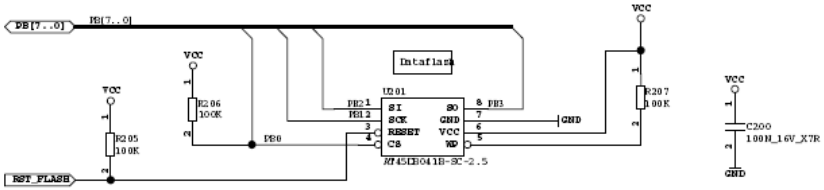
# DIAGRAMA ESUQEMÁTICO DEL MÓDULO

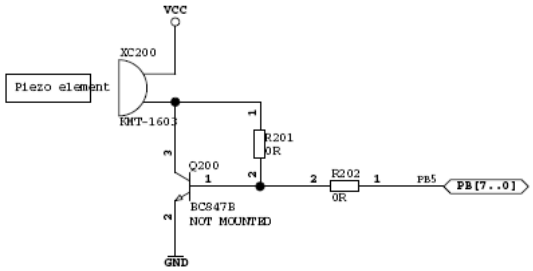
# AVR BUTTERFLY

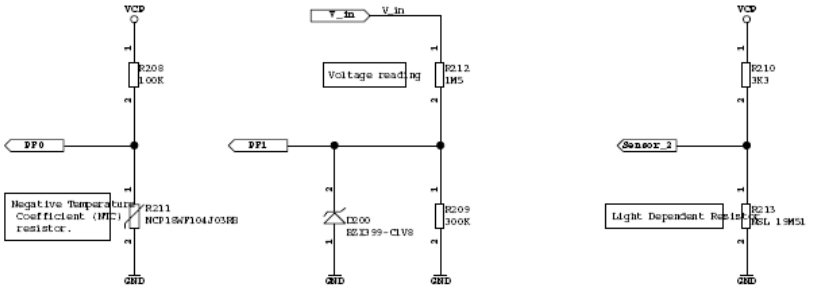

PF[7..4]
Sensor_2
PF1
(voltage reader)
PF0
(temperature sensor)
VCP
VCC
GND
C101
100n_16V_X7R
C102
100n_16V_X7R
C103
100n_16V_X7R
C100
100n_16V_X7R
L100
BLM-21A102S
C105
1U_16V_X7R
U100
ATMEGA169V-1MC
LCDCAP
PE0(RXD)
PE1(TXD)
PE2(AIN0/XCK)
PE3(AIN1)
PE4(SCL/USCK)
PE5(SDA/DI)
PE6(DO)
PE7
PB0(SS)
PB1(SCK)
PB2(MOSI)
PB3(MISO)
PB4(OC0)
PB5(OC1A)
PB6(OC1B)
PB7(OC2)
PG3(T1/SEG24)
PG4(T0/SEG23)
RESET
XTAL2(TOSC2)
XTAL1(TOSC1)
PD0(ICP/SEG22)
PD1(INT0/SEG21)
PD2(SEG20)
PD3(SEG19)
PD4(SEG18)
PD5(SEG17)
PD6(SEG16)
PD7(SEG15)
(COM3)PA3
(SEG20)PA4
(SEG21)PA5
(SEG22)PA6
(SEG23)PA7
(SEG24)PG2
(SEG25)PC7
(SEG26)PC6
(SEG27)PC5
(SEG28)PC4
(SEG29)PC3
(SEG30)PC2
(SEG31)PC1
(SEG32)PC0
(SEG33)PG1
(SEG34)PG0
COM[3..0]
LCD[32..3]
TOSC1
TOSC2
AVR_TxD
AVR_RxD
PE2
PE3
PE[6..4]
PB[6..4]
FLT_FLASH
PB[7..0]
R100
10K
RESET
C106
10n_50V_X7R
C104
100n_16V_X7R

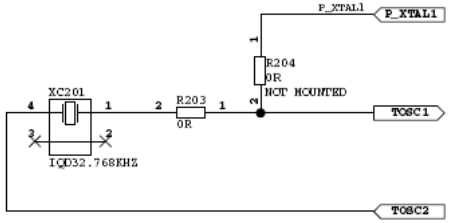

COM[3..0]
COM[3..0]
U200
LCD display
COM2
COM3
LCD4
LCD5
LCD6
LCD7
LCD8
LCD9
LCD10
LCD11
LCD12
LCD13
LCD14
LCD15
LCD16
LCD17
BD3
BD4
1K/1L/1M/6
2K/2J/2L/2M
3K/3J/3L/3M
3A/3B/3C/3D
4K/4J/4L/4M
4A/4B/4C/4D
5K/5J/5L/5M
5A/5B/5C/5D
9/6P/6E/6Q
89/6B/6Q/6N
6K/6J/6L/6M
6A/6B/6C/6D
10/7P/7E/7Q
810/7B/7Q/7N
7K/7J/7L/7M
BD2
BD1
1P/1Q/1E/X2
1A/1C/1N/1D
I1/1B/1C/86
1/2P/2E/2D
81/2E/2Q/2N
2A/2B/2C/2D
2/3P/3E/3D
82/3E/3Q/3N
3/COL1/87/7
83/4P/4E/4D
4/4E/4Q/4N
84/5P/5E/5D
5/5E/5Q/5N
85/COL2/86/8
7A/7E/7C/7D
COM1
COM0
LCD29
LCD28
LCD27
LCD26
LCD25
LCD24
LCD23
LCD22
LCD21
LCD20
LCD19
LCD18
M4042-DL
LCD[32..3]
LCD[32..3]
R200
0R
Connects COL1 to
COL2 on the LCD

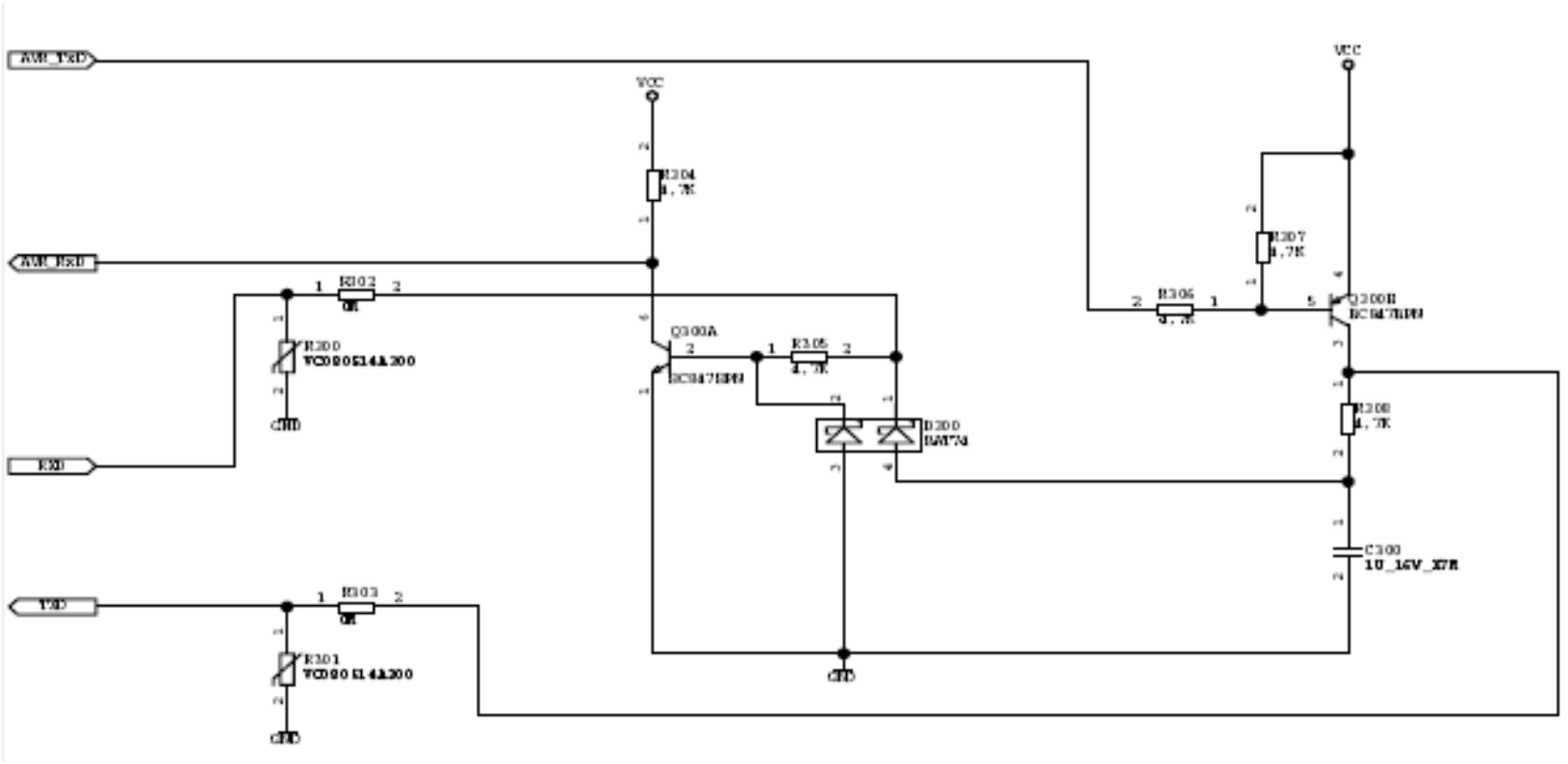

PB[7..0]
PB[7..0]
SW200
Center
A
B
C
D
Common
PB4
PB6
PB7
PE2
PE3
PE2
PE3
SKRHABE010
4 way joystick with
press function
GND

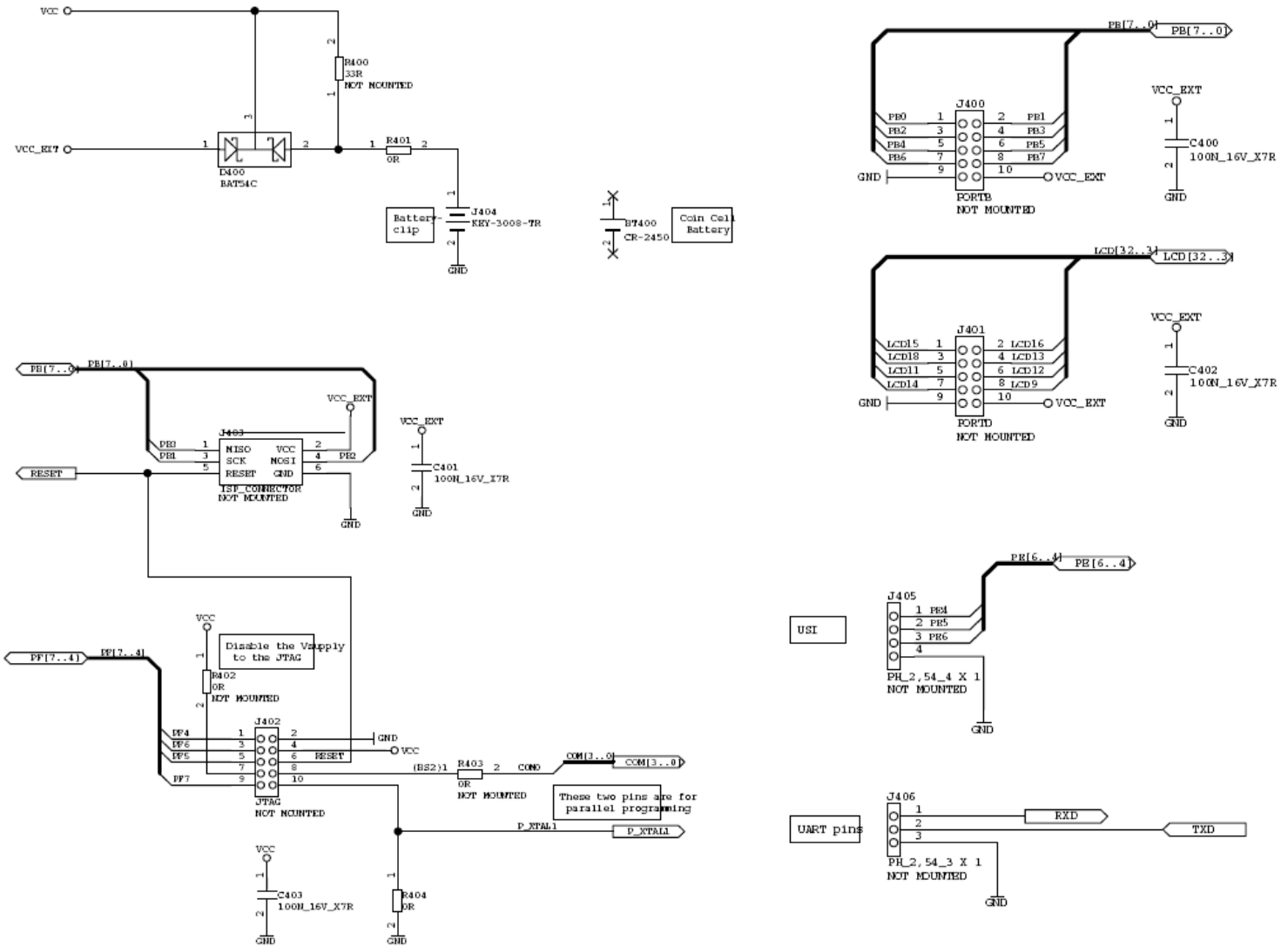

PB[7..0]
PB[7..0]
Intaflash
VCC
VCC
VCC
R206
100K
R205
100K
RST_FLASH
PB0
U201
PE2
PE3
SI
SCK
RESET
CS
SO
GND
VCC
WP
GND
AT45DB041B-SC-2.5
R207
100K
VCC
C200
100N_16V_X7R
GND


VCC
Piezo element
XC200
KHT-1603
R201
0R
Q200
BC847B
NOT MOUNTED
R202
0R
PB5
PB[7..0]
GND


VCP
R208
100K
PF0
Negative Temperature
Coefficient (NTC)
resistor.
R211
NCP15WF104J03RB
GND
V_in
V_in
Voltage reading
R212
1M5
PF1
D200
BZX399-C1V8
R209
300K
GND
VCP
R210
3K3
Sensor_2
Light Dependent Resistor
R213
NSL 19M51
GND


P_XTAL1
P_XTAL1
R204
0R
NOT MOUNTED
XC201
R203
0R
TOSC1
IQD32.768KHZ
TOSC2


AVR_TXD
AVR_RXD
RXD
TXD
VCC
R304
4.7K
R302
0R
R200
VC0805S14A200
GND
Q300A
2
1
R305
4.7K
D300
R303
0R
R301
VC0805S14A200
GND
R306
R307
4.7K
Q300B
R308
4.7K
C300
10_16V_X7R


VCC
R400
33R
NOT MOUNTED
VCC_EXT
D400
BAT54C
R401
0R
J404
KEY-3008-7R
Battery clip
GND
B7400
CR-2450
Coin Cell Battery
PB[7..0]
J400
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PORTB
NOT MOUNTED
C400
100N_16V_X7R
LCD[32..3]
J401
LCD15
LCD16
LCD18
LCD13
LCD11
LCD12
LCD14
LCD9
PORTD
C402
100N_16V_X7R
J403
MISO
SCK
RESET
MOSI
ISP_CONNECTOR
C401
100N_16V_X7R
PF[7..4]
Disable the Vapply to the JTAG
R402
J402
PF4
PF5
PF6
PF7
JTAG
R403
COM[3..0]
These two pins are for parallel programming
P_XTAL1
C403
R404
USI
J405
PH_2,54_4 X 1
UART pins
J406
RXD
TXD
PH_2,54_3 X 1

# ANEXO V

# HOJA DE DATOS DEL DRIVER

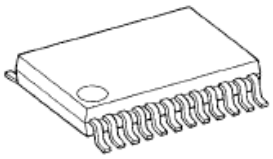
# TB6612FNG

Toshiba Bi-CD Integrated Circuit Silicon Monolithic

# TB6612FNG

## Driver IC for Dual DC motor

TB6612FNG is a driver IC for DC motor with output transistor in LD MOS structure with low ON-resistor. Two input signals, IN1 and IN2, can choose one of four modes such as CW, CCW, short brake, and stop mode.



SSOP24-P-300-0.65A

質量: 0.14 g (標準)

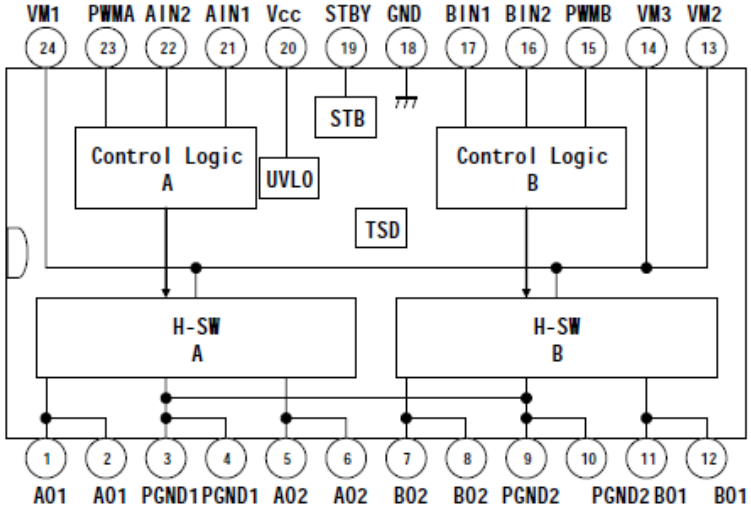
## Features

- Power supply voltage ; VM=15V (Max.)
- Output current ; Iout=1.2A(ave) / 3.2A (peak)
- Output low ON resistor ; 0.5Ω (upper+lower Typ. @VM≧ 5V)
- Standby (Power save) system
- CW/CCW/short brake/stop function modes
- Built-in thermal shutdown circuit and low voltage detecting circuit
- Small faced package (SSOP24 : 0.65mm Lead pitch)
- Response to Pb free packaging

* This product has a MOS structure and is sensitive to electrostatic discharge. When handling this product, ensure that the environment is protected against electrostatic discharge by using an earth strap, a conductive mat and an ionizer. Ensure also that the ambient temperature and relative humidity are maintained at reasonable levels.

The TB6612FNG is a Pb-free product.
The following conditions apply to solderability:
*Solderability
1. Use of Sn-37Pb solder bath
   *solder bath temperature = 230°C
   *dipping time = 5 seconds
   *number of times = once
   *use of R-type flux
2. Use of Sn-3.0Ag-0.5Cu solder bath
   *solder bath temperature = 245°C
   *dipping time = 5 seconds

## Block Diagram




## Pin Functions

| Pin NO. | Symbol | I/O | Remarks |
|---|---|---|---|
| 1 | AO1 | O | chA output1 |
| 2 | AO1 | | |
| 3 | PGND1 | — | Power GND 1 |
| 4 | PGND1 | | |
| 5 | AO2 | O | chA output2 |
| 6 | AO2 | | |
| 7 | BO2 | O | chB output2 |
| 8 | BO2 | | |
| 9 | PGND2 | — | Power GND 2 |
| 10 | PGND2 | | |
| 11 | BO1 | O | chB output1 |
| 12 | BO1 | | |
| 13 | VM2 | — | Motor supply (2.5V～13.5V) |
| 14 | VM3 | | |
| 15 | PWMB | I | chB PWM input / 200kΩ pull-down at internal |
| 16 | BIN2 | I | chB input2 / 200kΩ pull-down at internal |
| 17 | BIN1 | I | chB input1 / 200kΩ pull-down at internal |
| 18 | GND | — | Small signal GND |
| 19 | STBY | I | "L"=standby / 200kΩ pull-down at internal |
| 20 | Vcc | — | Small signal supply (2.7V～5.5V) |
| 21 | AIN1 | I | chA input1 / 200kΩ pull-down at internal |
| 22 | AIN2 | I | chA input2 / 200kΩ pull-down at internal |
| 23 | PWMA | I | chA PWM input / 200kΩ pull-down at internal |
| 24 | VM1 | — | Motor supply (2.5V～13.5V) |

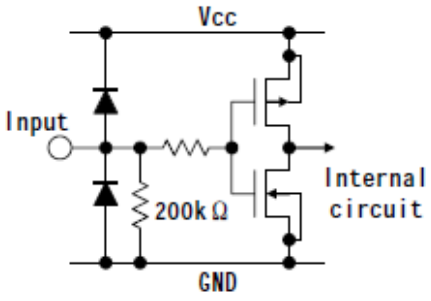
## Absolute Maximum Ratings (Ta = 25°C)

| Characteristics | Symbol | Rating | Unit | Remarks |
|---|---|---|---|---|
| Supply voltage | VM | 15 | V | |
| | Vcc | 6 | | |
| Input voltage | VIN | -0.2～6 | V | IN1, IN2, STBY, PWM pins |
| Output voltage | Vout | 15 | V | O1, O2 pins |
| Output current | Iout | 1.2 | A | Per 1ch |
| | Iout (peak) | 2 | | tw=20ms Continuous pulse, Duty≦20% |
| | | 3.2 | | tw=10ms Single pulse |
| Power dissipation | PD | 0.78 | W | IC only |
| | | 0.89 | | 50×50 t=1.6(mm) Cu≧40% in PCB mounting |
| | | 1.36 | | 76.2×114.3 t=1.6(mm) Cu≧30% in PCB monting |
| Operating temperature | Topr | -20～85 | ℃ | |
| Storage temperature | Tstg | -55～150 | ℃ | |

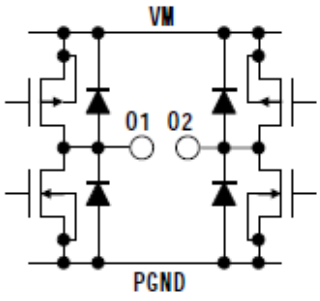
## Operating Range (Ta=-20～85℃)

| Characteristics | Symbol | Min | Typ. | Max | Unit | Remarks |
|---|---|---|---|---|---|---|
| Supply voltage | Vcc | 2.7 | 3 | 5.5 | V | |
| | VM | 4.5 | 5 | 13.5 | V | |
| Output current (H-SW) | Iout | --- | --- | 1.0 | A | VM≧5V |
| | | --- | --- | 0.4 | | 5V>VM≧4.5V |
| Switching frequency | fPWM | --- | --- | 100 | kHz | |

Input pin; IN1, IN2, PWM, STBY




Output pin; O1, O2

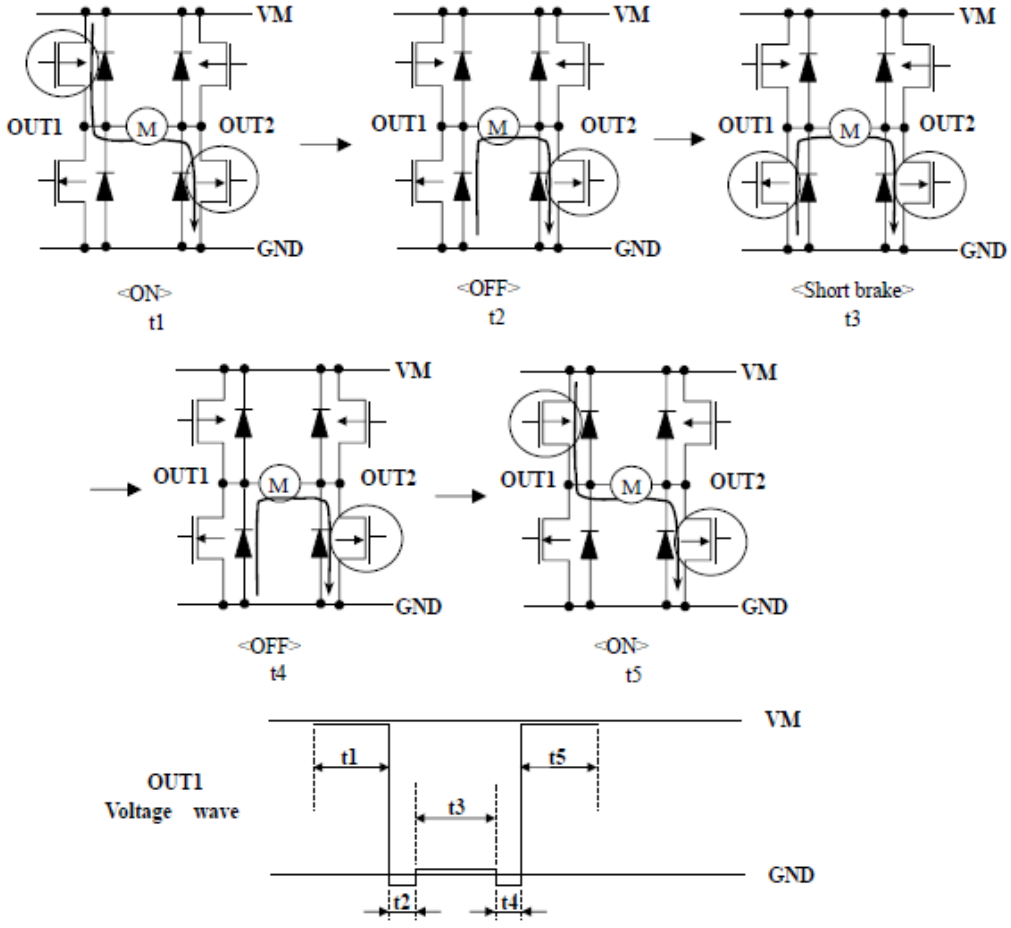


## H-SW Control Function

| Input | | | | Output | | |
|---|---|---|---|---|---|---|
| IN1 | IN2 | PWM | STBY | OUT1 | OUT2 | Mode |
| H | H | H/L | H | L | L | Short brake |
| L | H | H | H | L | H | CCW |
| | | L | H | L | L | Short brake |
| H | L | H | H | H | L | CW |
| | | L | H | L | L | Short brake |
| L | L | H | H | OFF (High impedance) | | Stop |
| H/L | H/L | H/L | L | OFF (High impedance) | | Standby |

## H-SW Operating Description

・To prevent penetrating current, dead time t2 and t4 is provided in switching to each mode in the IC.



# ANEXO VI

# HOJA DE DATOS DEL

# HM-TR 434

# HM-TR Series UHF Wireless Transparent Data Transceiver

## General

The HM-TR series UHF wireless transparent data transceiver, developed by Hope Microelectronics Co. Ltd, is designed for applications that need wireless data transmission. It features high data rate, longer transmission distance, programmable frequencies, configurable UART formats and low sleep current make it ideal choice. The communication protocol is self controlled and completely transparent to users. The module can be embedded to your existing design so that low cost high performance wireless data communication can be utilized easily.

## Features

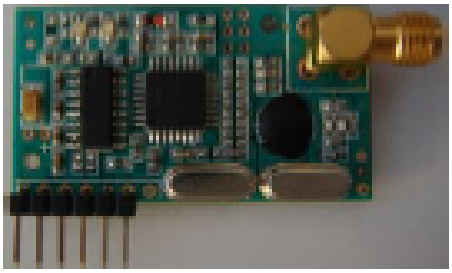
1. FSK (Frequency Shift Keying) modulation, high interference immunity
2. 2-way half-duplex communication
3. 315/433/868/915MHz ISM band, globally license free.
4. Programmable frequencies, allowing be used in FDMA (Frequency Division Multiple Access) applications
5. Self controlled RF to UART protocol translation, reliable and easy to use.
6. Configurable UART format, with data rate from 300~19200bps
7. Using ENABLE pin to control duty-cycle to satisfy different application requirements
8. High performance, long transmission range. =300m in open area
9. Standard UART interface, with TTL or RS232 logic level available
10. Compact size, standard 0.1" pinch SIP connector and SMA antenna socket
11. No RF tuning needed in application

## Application Areas

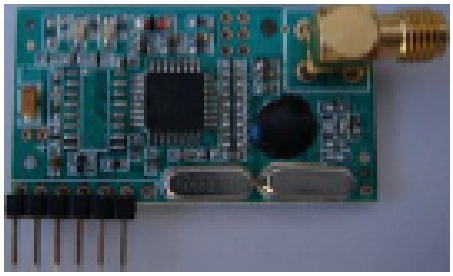
1. Remote control, remote measurement system
2. Wireless metering
3. Access control
4. Identity discrimination
5. Data collection
6. IT home appliance
7. Smart house products
8. Data store and forward repeater

## Overview and Pin assignment

HM-TR/232

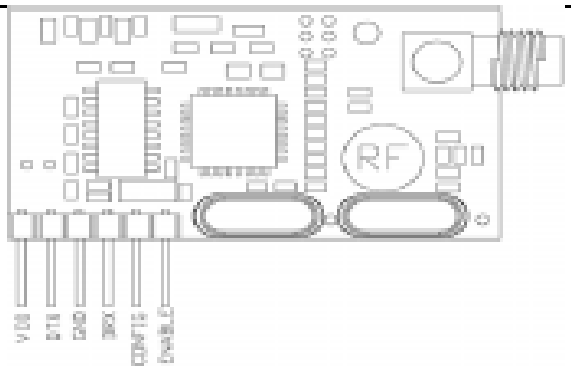


HM-TR/TTL



# HOPE RF

## HM-TR v2.2




| Pin | name | note |
|---|---|---|
| 1 | VDD | Power supply |
| 2 | DTX | Data output from module |
| 3 | GND | Ground |
| 4 | DRX | Data input to module |
| 5 | CONFIG | If this pin is high at power on, module will enter configure mode, while it communicates if set low |
| 6 | ENABLE | If this pin is low in normal mode, the module will enter sleep mode immediately. Assert high will awaken |

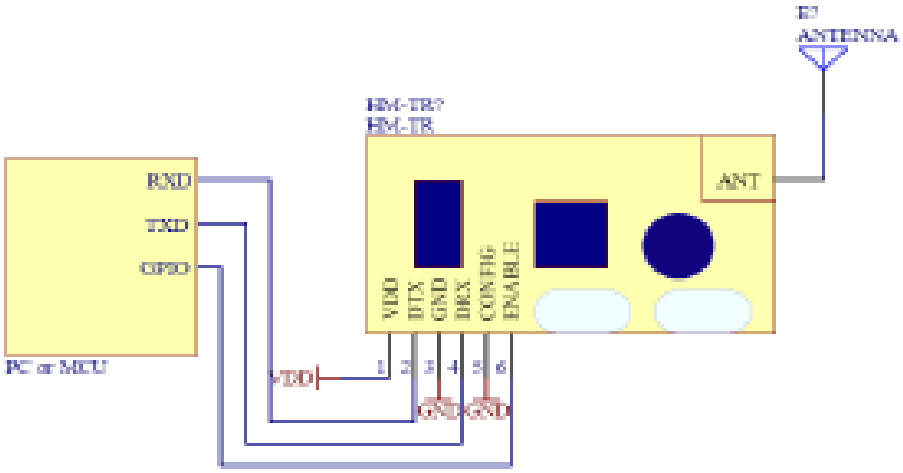
### Parameters

| parameter | condition | min | typical | max | |
|---|---|---|---|---|---|
| Power supply | | 4.5 | 5.0 | 5.0 | V |
| Operate temperature | | -35 | 25 | 80 | ℃ |
| Operate frequency | | | | | MHz |
| | HM-TR433 | 430.24 | 434 | 439.75 | |
| | HM-TR868 | 860.48 | 869 | 879.51 | |
| | HM-TR915 | 900.72 | 915 | 929.27 | |
| | | | | | |
| Max output power | | | | | dBm |
| | HM-TR433 | 3 | 5 | | |
| | HM-TR868 | -2 | 0 | | |
| | HM-TR915 | -2 | 0 | | |
| Transmitting power | | Pmax-21 | Pmax | Pmax | dBm |
| Receive sensitivity | | | | | dBm |
| | HM-TR433 | | -105 | -100 | |
| | HM-TR868 | | -102 | -95 | |
| | HM-TR915 | | -102 | -95 | |
| | | | | | |
| TX current | | | | | mA |
| | HM-TR433 | | | 26 | |
| | HM-TR868 | | | 28.5 | |
| | HM-TR915 | | | 30 | |
| | | | | | |
| RX current | | | | | mA |
| | HM-TR433 | | | 15 | |
| | HM-TR868 | | | 16 | |
| | HM-TR915 | | | 17 | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **continued** | | | | | |
| Sleep current | | | | | uA |
| | HM-TR433/TTL | | | 1 | |
| | HM-TR868/TTL | | | 1 | |
| | HM-TR915/TTL | | | 1 | |
| | | | | | |
| Reference distance | | | | | m |
| | HM-TR433/TTL | | | 330 | |
| | HM-TR868/TTL | | | 220 | |
| | HM-TR915/TTL | | | 230 | |
| | | | | | |
| Modulate deviation | | 15 | | 240 | kHz |
| Receiver bandwidth | | 67 | | 400 | kHz |
| UART data rate | | 300 | 9600 | 19200 | bps |
| UART data bits | | 5 | 8 | 9 | bit |
| UART parity check | | None | Odd | Even | |
| UART stop bits | | 1 | 1 | 2 | bit |
| ANT connector | | | | | SMA female |
| Module size | | | | | 24×43mm |

## Quick Setup

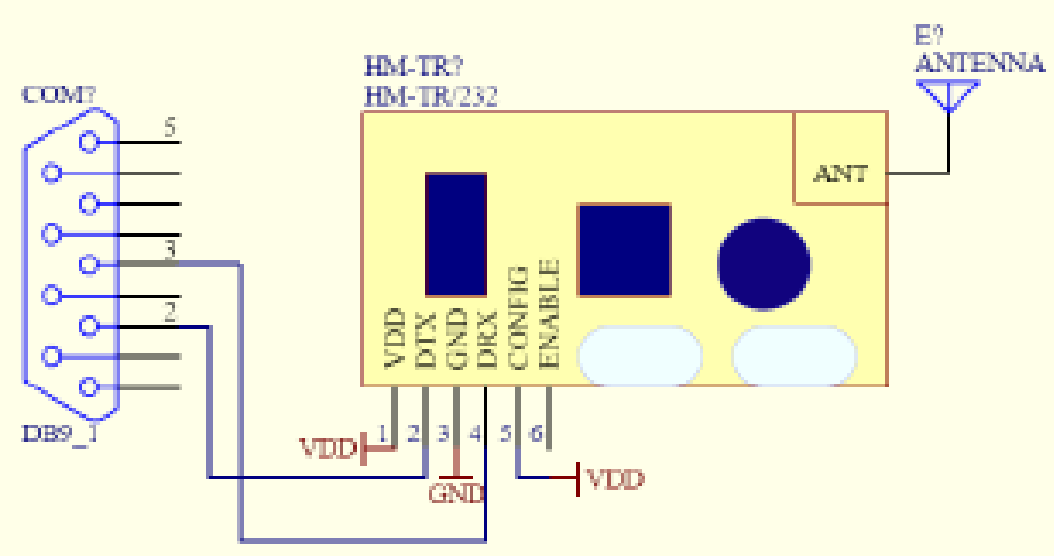
Connect HM-TR/232 to the RS232 connector of serial fitted PC or connect HM-TR/TTL to MCU(micro controller unit)'s UART directly, apply power supply, both RED and GREEN status LED will blink 3 times to indicate it is ready for your application. If CONFIG pin is low at power on, module will enters normal mode for data transmission, or CONFIG is high the module enters configure mode to setup work parameters.



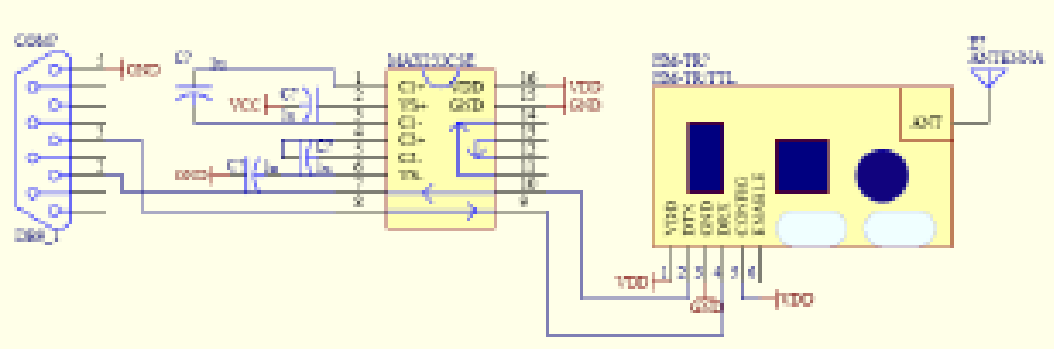

Note: MCU=Micro Controller Unit, PC=Personal Computer, GPIO=General Purpose Input/Output

### Normal mode connection

In normal mode, the ENABLE pin controls the module work or sleep, module will enter sleep mode as soon as the pin is low level.



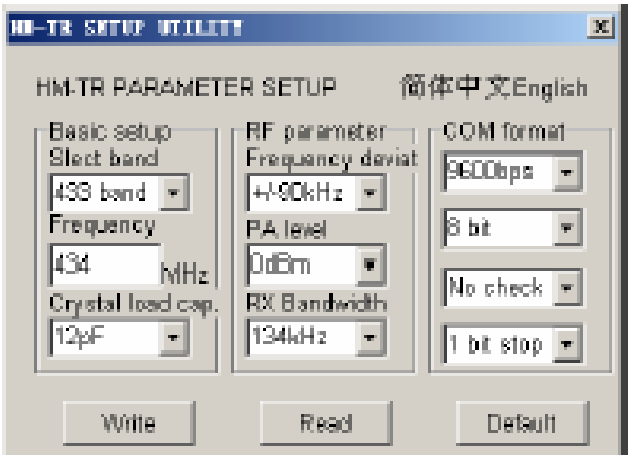

**Configure mode connection (HM-TR/232)**




**Configure mode connection (HM-TR/TTL)**

In configure mode, the module work parameters can be setup via the HM-TR setup utility, with the communication format between module and computer fixed at: 9600, 8, N, 1. see below:

## HM-TR SETUP UTILITY



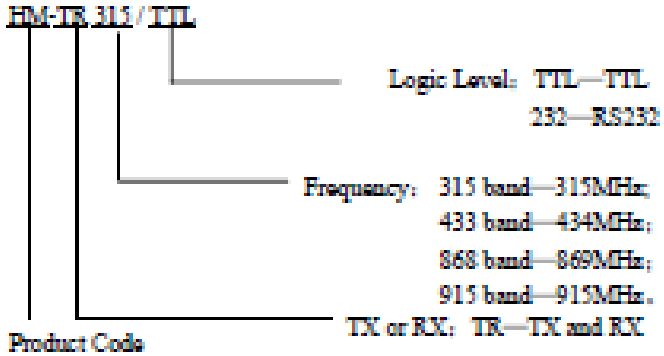

"Read" button: Read the parameters the module currently use;
"Write" button: Write new configuration to module;
"Default" button: Recover module parameters as default value

## Ordering Information

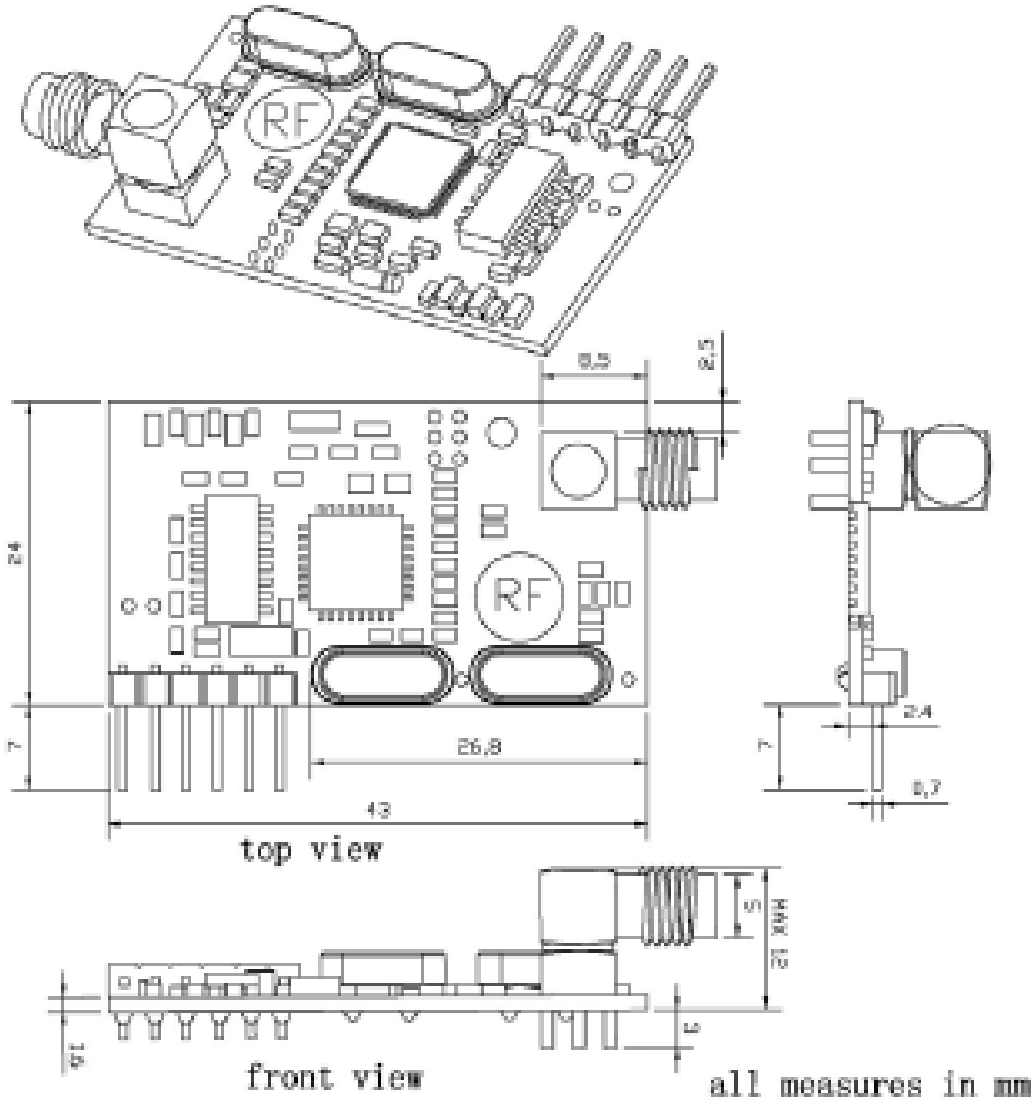
| Model | Logic Level |
|---|---|
| HM-TRxxxx/TTL | TTL |
| HM-TRxxxx/232 | RS232 |

232 versions usually are used in PC or equivalent, **TTL** version suited for simple 5V MCU system.

## Module Naming Rule



**Mechanical outline:**



# BIBLIOGRAFÍA

[1]. Pololu, Robotics & Electronics, Orangutan SV-328 Robot Controller, Specifications,

http://www.pololu.com/catalog/product/1227/specs

Fecha de consulta: 17/01/11.

[2]. Pololu, Robotics & Electronics, Orangutan SV-328 Robot Controller, Resources, http://www.pololu.com/catalog/product/1227/resources

Fecha de consulta: 20/01/11.

[3]. Pardue Joe, Smiley Micros.com, *C programming for Microcontrollers*, *Featuring ATMEL's AVR Butterfly and the Free WinAVR Compiler,*edición 2005, http://www.smileymicros.com/

Fecha de consulta: 10/01/11.

[4]. Pololu, Robotics & Electronics, Pololu Orangutan SV-xx8 and LV-xx8 User's Guide, http://www.pololu.com/docs/0J27

Fecha de consulta: 23/01/11.

[5]. Toshiba, TB6612FNG,
http://www.pololu.com/file/0J86/TB6612FNG.pdf

Fecha de consulta: 25/01/11.

[6]. Pololu, Robotics & Electronics , Pololu USB AVR Programmer User's Guide, http://www.pololu.com/docs/0J36

Fecha de consulta: 05/02/11.

[7]. Atmel, AVR Butterfly, http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3146

Fecha de consulta: 05/02/11.

[8]. Martin THOMAS, AVR-Projects, http://gandalf.arubi.uni-kl.de/avr_projects/

Fecha de consulta: 07/02/11.

[9]. INTPLUS, Robot TRI TRAC control PS2, http://www.superrobotica.com/S300168.htm

Fecha de consulta: 10/02/11.

[10]. José A. Pichardo Gallardo, Microbot SIKO, http://www.iearobotics.com/proyectos/siko/Microbot_Siko.html

Fecha de consulta: 10/02/11.

[12]. ATMEL, ATMEGA169 datasheet, www.atmel.com/dyn/resources/prod_documents/doc2514.pdf

Fecha de consulta: 31/01/11.

[13]. ATMEL, ATMEGA328P datasheet,
www.atmel.com/dyn/resources/prod_documents/doc8161.pdf

Fecha de consulta: 04/02/11.