



**TRABAJO DE FIN DE GRADO**

**INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA**

---

**BRAZO PROTÉSICO CON  
MÚSCULO NEUMÁTICO Y  
SENSOR ELÁSTICO**

---

**AUTOR: SAÚL RODRÍGUEZ HERNÁNDEZ  
TUTOR: JONAY TOMÁS TOLEDO CARRILLO**

**ESCUELA SUPERIOR DE INGENIERÍA Y TECNOLOGÍA  
LA LAGUNA, SEPTIEMBRE 2021**

## **Agradecimientos**

Quiero agradecer a mi tutor Jonay Toledo por haberme dado la oportunidad de realizar este proyecto, por aceptar mis ideas y ayudarme a enfocarlas. También por no rendirse conmigo, pese a que no haya podido avanzar todo lo que queríamos en un principio.

Me gustaría agradecer también a mi familia y amigos por apoyarme durante estos 5 años de universidad, escuchándome y animándome pese a mis agobios y problemas que surgían por el camino. Sin ellos este trabajo no habría salido a flote. Quiero recordar también a mis abuelos, quienes estoy seguro de que siguen apoyándome y de que se sienten orgullosos de su nieto.

Quiero hacer una mención especial a mi pareja, quien no sólo me ha ayudado a centrarme y a relajarme, sino que también ha contribuido corrigiendo la redacción. Además de ayudarme a expresar las ideas del proyecto en inglés.

También quiero recordar el gran apoyo que han sido mis compañeros de carrera, con los que he compartido momentos duros, largos días de estudio y más de un momento divertido.

Por último, quiero dar las gracias al profesorado del grado de electrónica por haberme enseñado los conocimientos que me han servido para la ejecución de este proyecto, y que, algún día, me permitirán convertirme en un profesional.

Muchas gracias a todos los que han estado conmigo estos años.

## Resumen

En este proyecto hemos investigado la viabilidad de la neumática en la robótica para la realización de prótesis biónicas al hacer uso de un actuador al que se le conoce como músculo neumático.

La principal diferencia entre el uso del músculo neumático y el clásico pistón neumático es la capacidad de carga y el rango de retracción o extensión que manejan. Por un lado, el pistón nos ofrece una capacidad de carga media y mayor rango de movimiento que el músculo. Por otro lado, el músculo es mucho más flexible, y tiene una gran capacidad de carga en relación a su bajo peso, pero tiene una capacidad de retracción bastante menor, del 24% de su extensión normal. Haciendo una valoración de las ventajas e inconvenientes, decidimos utilizar el músculo por su flexibilidad, por lo que soportaría mejor los movimientos de las demás articulaciones, y por su capacidad de carga superior a otros actuadores.

La prótesis que elaboramos es un brazo, aunque en este caso solo trabajaremos con el codo para evitar la coordinación entre los músculos, ya que por el tipo de actuador y de sensor, el modelo es poco preciso. Este hecho será importante remediarlo, debido a que la prótesis la utilizará una persona, y lo ideal es que no haya grandes diferencias con el manejo de un brazo humano.

El funcionamiento del actuador consistirá en dos electroválvulas “todo o nada”, que procurarán la entrada y salida de aire al músculo, por lo que el control en el código también será del tipo “todo o nada”. Concretamente, trabajaremos con 3 intervalos para el control del actuador: cuando el brazo tenga que comprimirse, por lo que deberemos introducir aire; cuando tenga que relajarse, por lo que expulsaremos aire; y cuando esté dentro de un margen de la posición requerida.

Para determinar la posición del codo usaremos como sensor una resistencia elástica, la cual es un tipo de goma que, al estirarse, reduce su resistencia eléctrica. Una de las bondades de este sensor es su alta elasticidad, ya que nos permite asignarle un rango de movimiento amplio, además de ser económico y resistir bien los movimientos de las articulaciones. El principal problema es su baja precisión, puesto que los valores pueden oscilar sin siquiera haber cambiado la extensión del elástico.

## **Abstract**

The aim of this project is to investigate the feasibility of pneumatics in robotics for the realisation of bionic prostheses by using an actuator known as pneumatic muscle.

The main difference between the use of the pneumatic muscle and the classic pneumatic piston is the load capacity and the range of retraction or extension that they can handle. On the one hand, the piston offers a medium load capacity and a greater range of movement than the muscle. On the other hand, the muscle is much more flexible than the piston, and has a high load capacity in relation to its low weight but has a much lower retraction capacity as for a 24% of its normal extension. Assessing the advantages and disadvantages, it was decided to use the muscle due to its flexibility, and would therefore better support the movements of the other joints, and also due to its higher load capacity than other actuators.

Although in this case, we will only work with the elbow, the prosthesis developed is an arm. This is so in order to avoid coordination between the muscles, as due to the type of actuator and sensor used, the prototype is not very precise. It will be important to solve this, as the prosthesis will be used by a person, and there should not be major differences with the operation of a human arm.

The functionality of the actuator will consist of two "all or nothing" solenoid valves, which will provide air in and out of the muscle, so the control in the code will also be of the "all or nothing" type. Specifically, it will run with three intervals for the actuator control: when the arm must compress, so the air will have to be introduced; when it must relax, so the air will be expelled; and when it is within a range of the required position.

To determine the position of the elbow, an elastic resistor will be used as a sensor, which is a type of rubber that, when stretched, reduces its electrical resistance. One of the advantages of this sensor is its high elasticity, which allows us to assign it a wide range of movement, as well as being cheap and resistant to joint movements. The main problem is its low accuracy, as the values can fluctuate without even changing the elastic extension.



## Índice

<b>Introducción</b>	<b>6</b>
1.1. Antecedentes	6
1.2. Objetivo	7
<b>Marco teórico</b>	<b>7</b>
2.1. Fuentes de energía	7
2.1.1. Actuadores eléctricos	7
2.1.2. Actuadores hidráulicos	8
2.1.3. Actuadores neumáticos	8
2.2. Sistemas neumáticos	9
2.2.1. Elementos de los sistemas neumáticos y su funcionamiento	9
2.2.2. Sensores y control neumático	10
2.3. Músculo neumático	11
2.4. Sensor de estiramiento flexible	12
<b>Herramientas utilizadas</b>	<b>13</b>
3.1. Protoboard	13
3.2. Arduino UNO	13
3.3. Sensor elástico	14
3.4. Músculo neumático	15
3.5. Electroválvulas	15
3.6. Compresor	16
3.7. Multímetro	16
3.8. Fuente de alimentación	17
3.9. Arduino IDE	17
3.10. LTspice XVII	18
3.11. Octave	18
3.12. Paquete Office y Google Drive	18
<b>Implementación</b>	<b>19</b>
4.1. Diseño del músculo	19
4.2. Sistema de sensado de la prolongación	20
4.3. Elección de sensores	21
4.4. Circuito de adaptación de señales	22
4.5. Adecuación del sensor y modelo de Steinhart-Hart	26
4.6. Adecuación de la lectura del sensor a ángulos, teorema del coseno	34
4.7. Sistema de control	35
<b>Elaboración del modelo experimental y decisiones de diseño</b>	<b>37</b>
<b>Diseño de código</b>	<b>39</b>
<b>Conclusión</b>	<b>43</b>
<b>Bibliografía</b>	<b>44</b>

<b>Anexos</b>	<b>46</b>
9.1. Lectura de datos del sensor	46
9.2. Diseño de circuito para la adecuación de la señal en LTspice	46
9.3. Código de Octave para el cálculo y graficación del modelo de Steinhart-Hart	46
9.4. Código de Arduino	46
9.5. Datasheets	46
9.5.1. Transistor	46
9.5.2. Amplificador operacional	46
9.5.3. Arduino	46

## **1. Introducción**

Las prótesis son una tecnología que ha tenido un gran impacto en la vida de aquellas personas que han sufrido la pérdida de alguno de sus miembros. Se cree que las primeras prótesis tenían un sentido más bien estético, aunque funcional, y que su origen se remonta al antiguo Egipto. Desde entonces, las prótesis se han ido desarrollando, a la vez que han ido variando en su tipología a lo largo de los siglos.

A lo largo de la historia de la humanidad, podemos encontrar una gran cantidad de prótesis: desde prótesis principalmente estéticas, sin articulaciones, ni mecanismos y, por ende, bastante limitadas; pasando por algunas más complejas y manejables que permitían sujetar objetos o movilizar sus articulaciones; hasta llegar a la modernidad, donde existen articulaciones que se mueven gracias a motores eléctricos o a otros tipos de actuadores y mecanismos. No obstante, la humanidad siempre da un paso más y hemos conseguido ser capaces de integrar dichas prótesis en nuestro sistema nervioso mediante sensores electromiográficos, capaces de captar las señales nerviosas de los músculos para que la prótesis se mueva acorde a ello. Incluso se ha logrado que el usuario tenga un sentido del tacto básico, pudiendo percibir la dureza de los materiales gracias a sensores de fuerza, contacto y temperatura, que mandarán las señales al cerebro del portador.

Este trabajo se centrará en las prótesis de brazo y en cómo se podría utilizar la neumática para conseguir mayor potencia para dichas prótesis.

El interés por este proyecto surge debido a mi afición por la robótica, concretamente la robótica para prótesis y la biónica. Por este motivo, decidimos investigar e innovar con la neumática dentro del mundo de las prótesis, utilizando como novedad un actuador llamado músculo neumático. Esta decisión se debe a que la neumática, a parte de ser una alternativa más económica, proporciona una gran capacidad de carga, además de su capacidad ignífuga. No obstante, su mayor defecto es su escasa estabilidad debido a la compresibilidad del gas y una respuesta más lenta en comparación con el caso eléctrico.

### **1.1. Antecedentes**

La realización de este trabajo se apoya en dos proyectos de brazo con músculos neumáticos de BioMakers Industries y de Hacksmith Industries. Estos dos proyectos han sido documentados en vídeo y se pueden encontrar en la plataforma de Youtube.

En el primero de los dos canales se explica cómo elaborar un músculo neumático, especificando los materiales necesarios para su elaboración y alimentación de aire comprimido. Además de esto, se comentan algunos consejos para su correcta implementación a una estructura.

Por su parte, el segundo proyecto se basa en la elaboración de un exoesqueleto de brazo para aumentar la capacidad de carga. En este proyecto se realizan pruebas con distintos materiales para la realización del músculo, explicando las diferencias entre cada

uno de ellos. Utilizaremos los resultados que han obtenido para escoger el material de nuestro músculo.

## **1.2. Objetivo**

El objetivo del proyecto consiste en comprobar la funcionalidad y viabilidad del brazo protésico que hemos explicado anteriormente. Para ello, utilizaremos el músculo neumático como actuador, y el sensor deberá de ser capaz de medir la posición de este con exactitud.

Se ha realizado un modelo de codo experimental compuesto por el músculo, el sensor elástico y dos electroválvulas para controlar la articulación. Sin embargo, debido a la complejidad de la implementación de un sensor electromiográfico, en este proyecto no se abordará la posibilidad de que sea utilizado por una persona.

## **2. Marco teórico**

En primer lugar, explicaremos varios conceptos de las prótesis modernas: la energía que utilizan, los actuadores que producen el movimiento, y los sensores que nos permitirán controlar todo el proceso.

### **2.1. Fuentes de energía**

Ya hemos visto que las prótesis robóticas tienen una tipología muy variada, donde se incluyen los tipos de actuadores y la energía que usan. En este apartado vamos a ver algunos de ellos.

#### **2.1.1. Actuadores eléctricos**

Los actuadores eléctricos son los más empleados en el mercado actual. Esto se debe a que las baterías son bastante ligeras y que proporciona mayor autonomía que los demás actuadores. Además de esto, los actuadores eléctricos son de los más precisos y con mayor capacidad de control que existen actualmente.

Los primeros actuadores que podemos ver son los motores eléctricos, los cuales generan movimientos rotatorios, que nos permiten conseguir los movimientos de las articulaciones fácilmente. Normalmente, se utilizan los motores de corriente continua, ya que los brazos protésicos suelen funcionar con baterías portables, para las que es conveniente un bajo consumo energético. Entre los motores de corriente continua destaca el servomotor, que es un tipo de motor con la capacidad de ubicarse y mantenerse estable en cualquier posición dentro de su rango de operación. El servomotor se compone de un motor y un circuito de control, con un sistema de regulación que puede controlarse tanto por posición como por velocidad.

Otro actuador que cabe destacar en este apartado son los músculos eléctricos, también llamados elastómeros dieléctricos, que se intercalan entre dos electrodos. Al aplicar un cierto voltaje, surge una presión electrostática debida a las fuerzas de Coulomb que actúan entre los electrodos. Los electrodos apretarán la película elastomérica, haciendo que esta se expanda, en un orden de entre el 10% y el 35% de su tamaño original. Esta expansión debida a la aplicación de un cierto voltaje se puede aprovechar para imitar a las fibras musculares, consiguiendo un tipo de actuador ideal para una prótesis. Al ser eléctrico tiene una respuesta rápida y se puede controlar directamente por señales eléctricas. El material utilizado está formado por una superposición de gran cantidad de capas de elastómeros y electrodos de nanotubos de carbono, lo que hace que sean músculos ligeros y que no necesiten de altas tensiones para funcionar.

### **2.1.2. Actuadores hidráulicos**

Los actuadores hidráulicos funcionan mediante fluidos a presión, necesitando de un sistema de bombeo que complica la autonomía, pero pueden proporcionar movimientos lineales y suaves, además de que proporcionan una gran potencia. Algunas de sus desventajas son la posibilidad de fugas de líquido, la necesidad de muchos componentes para realizar los movimientos y el ruido que genera en comparación a otros tipos de actuadores.

Los actuadores hidráulicos más comunes son los cilindros, tanto de simple como de doble efecto. Ambos proporcionan un movimiento lineal debido a la presión hidráulica sobre el émbolo. La diferencia entre ambos radica en que los de simple efecto tienen un retorno mecánico, ya sea por un muelle interno, o alguna fuerza externa; mientras que los de doble efecto realizan el retorno también por la presión hidráulica en el sentido contrario.

Existen también los actuadores hidráulicos rotativos, que generan un giro continuado, o, por el contrario, un giro determinado a un número de grados, que puede llegar hasta los 360 grados. Su funcionamiento es parecido al de una turbina: el fluido a presión empujará las palas, lo que provocará el giro del eje.

Por último destacaremos los motores hidráulicos, con la misma función que cualquier otro motor, generando un movimiento rotatorio continuo gracias a la presión de un fluido.

### **2.1.3. Actuadores neumáticos**

Como su nombre indica, los actuadores neumáticos funcionan con gases comprimidos, siendo el aire el más utilizado. Se caracterizan por ser seguros, económicos y rápidos tanto en el arranque como al parar.

Los actuadores neumáticos más conocidos son los pistones neumáticos, también llamados cilindros, los cuales también se dividen en de simple y de doble efecto, como en el caso de los hidráulicos.

Otros actuadores neumáticos son los actuadores de giro limitado, que nos permiten un desplazamiento angular en desplazamientos de menos de una vuelta completa.

Asimismo, también existen los motores neumáticos, que proporcionan un movimiento rotatorio constante. Su principal ventaja es que alcanzan velocidades angulares bastante altas.

Por último, nombraremos al actuador que utilizaremos para la realización de este proyecto: el músculo neumático. Como vimos con el elastómero dieléctrico, su funcionamiento es como el de un músculo natural, solo que utiliza el aire comprimido para realizar la compresión. Hablaremos más en detalle de este actuador en el apartado 2.3.

## **2.2. Sistemas neumáticos**

Los sistemas neumáticos son aquellos que emplean el aire (o algún gas) comprimido como medio de transmisión de la energía para hacer funcionar distintos mecanismos. Estos sistemas se rigen por la ley de los gases ideales, que dicta que los fluidos gaseosos se pueden comprimir por medio de una fuerza, manteniendo dicha compresión para luego devolver esa energía acumulada al permitirle expandirse. Esta ley se describe en la siguiente ecuación:

$$P \cdot V = n \cdot R \cdot T$$

Dónde P es la presión absoluta, V el volumen de gas, n moles de gas, R la constante universal de los gases ideales y T la temperatura absoluta.

### **2.2.1. Elementos de los sistemas neumáticos y su funcionamiento**

Un sistema neumático básico comprende varios elementos: un compresor, un tanque o depósito, un secador con filtro, las válvulas, y los actuadores finales.

El compresor es nuestra fuente de energía o generador del sistema. Absorbe aire de la atmósfera, aumentando su presión reduciendo el volumen en el que se encuentra, hasta llegar a la presión de consigna. El control del compresor se realiza mediante un manómetro como sensor del modelo de control. El manómetro es un instrumento de medida para la presión de fluidos en recipientes cerrados.

Lo siguiente es el depósito, donde se almacena y se enfría el aire a alta presión. El depósito posee varios elementos de medida para controlar las condiciones del aire contenido.

El filtro acondiciona el aire previo a introducirlo al circuito neumático, secando o eliminando la humedad de este.

Las válvulas son, en cierto modo, los elementos activadores de los actuadores, pero que también nos permiten dirigir el aire dentro del circuito, impidiendo o permitiendo su paso. Normalmente son controladas electrónicamente. Además de permitirnos dirigir el flujo del aire, nos permiten regular el caudal e incluso la presión de éste si así fuese necesario.

Para explicar los actuadores utilizaremos como ejemplo el de un pistón neumático. Un pistón está compuesto por un tubo, una tapa posterior, una tapa anterior con cojinete y aro rascador. El diámetro interno del tubo marcará la fuerza y el consumo del pistón, ya que, a mayor diámetro, se ejercerá mayor fuerza. En el caso de la presión del aire, a mayor compresión, también provocará mayor fuerza. Esto se ve en la siguiente fórmula:

$$F = P \cdot A$$

Dónde F es la fuerza que ejerce el pistón, P la presión del aire y A la sección interna del tubo.

En la práctica hay que tener en cuenta los rozamientos del pistón, por lo que la fuerza real que ejerce el pistón será distinta a la calculada en teoría con esta fórmula.

La longitud del tubo determinará la longitud de trabajo útil, ya que el émbolo tendrá la misma longitud que el cilindro. No se debe exceder la longitud de la carrera, puesto que el esfuerzo mecánico del vástago provoca riesgo de pandeo, además de ser necesarios diámetros superiores a los normales.

La velocidad del émbolo depende de la fuerza en oposición, de la presión del aire, de la longitud del tubo, de la sección entre los elementos de mando y trabajo, y del caudal que circula por los elementos de mando. También influye la velocidad de amortiguación de los finales de carrera. La velocidad del émbolo puede regularse con algunos tipos de válvulas.

El consumo de aire, que influye también en el consumo energético y los costes de funcionamiento, se calcula según la potencia que consumen los actuadores. El cálculo de consumo de aire se puede hallar con la siguiente fórmula del caudal:

$$Q = 2 \cdot (S \cdot n \cdot q)$$

Dónde Q es el caudal nominal (NI/min) (NI se refiere a newton litro), S la carrera (cm), n las carreras por minuto, y q el consumo por carrera.

### **2.2.2. Sensores y control neumático**

Los sistemas neumáticos utilizan distintos tipos de sensores para poder controlar la posición de los actuadores, como, por ejemplo, los finales de carrera. El problema de este tipo de sensores es que son “todo o nada”, o dicho de otro modo, no podemos saber la posición continuada del actuador, sino cuando está extendido del todo o retraído completamente. Uno de los sensores más utilizados son los de posicionamiento de tipo

magnético. La tecnología que utilizan es sin contacto, para poder detectar la posición del pistón en su cilindro, pudiendo utilizarlo para cualquier posición puntual, pero no para un control continuado del émbolo. Para el sensado, no hay necesidad de remover los mecanismos adicionales del pistón, ya que el émbolo es ferromagnético y el cilindro de aluminio, lo que facilita su implementación.

### 2.3. Músculo neumático

El músculo neumático (Figura 3.4.) es un actuador alimentado por aire a presión, que utilizaremos para simular el bíceps del brazo. Está compuesto por un tubo de goma de látex que se infla con la presión del aire, lo que lo expandirá a lo ancho. Este tubo estará rodeado por un flexible de fontanería (Figura 3.4.1.), que, debido al trenzado de los hilos metálicos, sostendrá al tubo flexible, no solo evitando que se estire, sino consiguiendo que se comprima con fuerza, como lo haría un músculo orgánico.



Figura 3.4. Músculo neumático



Figura 3.4.1. Flexible de acero inoxidable trenzado

La goma de látex (Figuras 2.4.2. y 2.4.3.) nos resultó difícil de encontrar, pero, finalmente, la obtuvimos de tiras elásticas para gimnasia, que resultaron ser tubulares, permitiéndonos tener suficiente con una sola goma para dos músculos, en caso de necesitar repuestos.





*Figura 2.4.2. Goma látex / Figura 2.4.3. Tira elástica de gimnasia*

El flexible de fontanería que utilizamos es de acero inoxidable trenzado, pero encontramos distintos materiales que consiguen variar las propiedades del músculo, como la fibra de carbono, que nos da un poco más de fuerza, pero menor contracción. Las dimensiones del flexible también provocan cambios en las propiedades, por ejemplo, un flexible de un diámetro de 3/8 de pulgada tiene una retracción del 32% y es capaz de levantar 47.63 Kg, mientras que un diámetro superior, de una pulgada, tiene una retracción del 25% y una carga de 92.99 Kg. Con esto podemos suponer que el diámetro está relacionado con la contracción y con la carga, de manera que un diámetro más fino nos dará un rango de movilidad mayor, pero menos carga. Por el contrario, un flexible más grueso nos proporcionará mayor fuerza, pero menor capacidad de carga.

#### **2.4. Sensor de estiramiento flexible**

Para llevar a cabo el control del músculo neumático usaremos una resistencia elástica (Figura 2.4.), llamada sensor de estiramiento flexible por los fabricantes, que estará ubicada para recoger la variación de la posición del músculo, estirándose y contrayéndose a la par que este. Según informa el fabricante, esta resistencia está fabricada en caucho impregnado en carbono, tiene un diámetro de 2mm y su resistencia aumenta entre 140-160 $\Omega$  por centímetro que la estiremos. Los enganches de este sensor se harán mediante dos chinchetas, ya que el modelo se realiza en madera, lo que nos permitirá cambiar la posición del sensor en caso de ser necesario.



Figura 2.4. Sensor de estiramiento flexible

### 3. Herramientas utilizadas

#### 3.1. Protoboard

Es una placa de pruebas para el diseño de circuitos electrónicos sin la necesidad de soldar los componentes. Tiene dos pares líneas de alimentación en cada extremo, y dos conjuntos de filas que constituyen nodos con 5 conexiones. A continuación se puede ver una imagen de las conexiones en la protoboard (Figura 3.1).

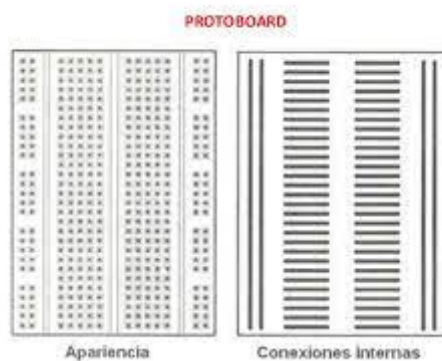


Figura 3.1. Esquema de conexiones de la protoboard

#### 3.2. Arduino UNO

Se trata de una placa (Figura 3.2.) que se utiliza para el desarrollo de proyectos de electrónica sencilla. Está compuesta por un microcontrolador ATmega328P. Gracias a su conexión por USB, la placa no necesita de una fuente externa. Asimismo, el lenguaje de programación es sencillo, semejante al C++. La compañía también distribuye multitud de sensores y actuadores considerablemente económicos para la elaboración de proyectos.



Figura 3.2. Placa Arduino UNO

Adjunto las características técnicas:

Microcontrolador	ATmega328P
Voltaje de operación	5 V
Voltaje de alimentación (recomendado)	7-12 V
Voltaje de alimentación (límite)	6-20 V
Pines E/S digitales	14 (de las cuales 6 proveen de salida PWM)
Pines PWM E/S digitales	6
Pines de entrada analógicos	6
Corriente por pin de CC de E/S	20 mA
Corriente por pin de 3.3V de CC	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

### 3.3. Sensor elástico

Es un tipo de goma resistiva que, en función de su elongación, aumenta su resistencia eléctrica. Esta resistencia aumenta entre 140-160 $\Omega$  por centímetro de estiramiento. Está fabricada en caucho impregnado en carbono, y tiene un diámetro de 2mm. Viene de fábrica con dos pinzas cocodrilo para su sujeción, pero nosotros utilizamos unas chinchetas debido al modelo experimental que hemos elaborado.

### 3.4. Músculo neumático

El músculo neumático es un sustituto del clásico émbolo o pistón neumático, siendo capaz de levantar pesos de 92.99 Kg. Consiste en un tubo de goma de látex, un flexible de fontanería de acero inoxidable trenzado, unas abrazaderas y cinta teflón para sellar los extremos, además de una conexión para controlar la entrada y salida de aire. El tubo elástico se coloca dentro del flexible, sellando los laterales, colocando la entrada para la conexión del tubo del compresor en uno de esos extremos. Su funcionamiento consiste en introducir aire al tubo elástico y que este se hinche, obligando al flexible a expandirse. Debido al trenzado se comprime el músculo, acortándolo a lo largo.

Este tipo de músculo se retrae solo hasta un 30% de su longitud, dependiendo de los diámetros del tubo y el flexible. La presión del compresor creemos que no deberá sobrepasar los 4 bares. La contracción va de los 25 cm a los 19 cm. Uno de los beneficios de este músculo es que, aunque la goma estalle por la presión, el flexible metálico evitará el daño, además de ser fácilmente sustituible y de bajo coste.

### 3.5. Electroválvulas

Una electroválvula (Figura 3.5.) es una válvula controlada de manera eléctrica, que permite o no el paso de un fluido por un conducto. Para ello, se sirve de un solenoide que se activará con el paso de corriente, abriendo el paso del aire a presión, por ello este tipo de válvulas se les nombra como normalmente cerradas. Las electroválvulas que utilizamos funcionan a 12V, para lo que necesitaremos de una fuente de alimentación que nos los proporcione.



Figura 3.5. Electroválvula “todo o nada”

### 3.6. Compresor

El compresor (Figura 3.6.) es una máquina que aumenta la presión de un fluido compresible, en nuestro caso el aire. La compresión ocurre por un intercambio energético entre la máquina y el aire, transmitiendo el trabajo al fluido en forma de presión y energía cinética. Los compresores pueden tener incorporada una cámara de aire para almacenar el aire a presión y permitir trabajar sin necesidad de salida de aire constante, pudiendo apagar el compresor para ahorrar energía. Los compresores sin cámara de aire han de trabajar con salida constante de aire y, en caso de no tenerla, corren el riesgo de estallar por la presión, por lo que hay que ser cuidadosos con estas máquinas. Muchos de los compresores traen instrumentos de medida, llamados manómetros, capaces de medir la presión del fluido dentro de un recipiente cerrado, por lo que nos permiten ajustar la presión del fluido a presión.



*Figura 3.6. Compresor neumático*

### 3.7. Multímetro

Un multímetro (Figura 3.7.) es un dispositivo electrónico portátil que permite llevar a cabo la toma de medidas de diferentes variables eléctricas en circuitos eléctricos, como lo son la resistencia, el voltaje y la intensidad de la corriente, además de las magnitudes pasivas. Es capaz de medir variables en corriente continua y alterna. Permite, además, medir valores para distintas escalas, que hay que cambiar según las dimensiones que vayamos a medir.



Figura 3.7. Multímetro

### 3.8. Fuente de alimentación

La fuente de alimentación (Figura 3.8.) es un dispositivo que proporciona una tensión y corriente modulables para un circuito u otro dispositivo. En el caso de la fuente de laboratorio que utilizamos, mantiene una salida de corriente o tensión estable, con poco ruido, para asegurar el correcto funcionamiento y la toma de mediciones más fiables. Nuestra fuente trabaja con corriente alterna, y la salida que proporciona es de corriente continua.



Figura 3.8. Fuente de alimentación

### 3.9. Arduino IDE

Es el programa de control de la placa Arduino, el cual se compone de una serie de herramientas para el desarrollo de código. El lenguaje del programa es simple y está construido sobre el lenguaje C++.

La elaboración de los *scripts* consta de dos funciones principales:

-*Void Setup*- como su nombre indica, es la función de inicio de arduino que solo se ejecutará una vez al iniciar el programa. Usualmente se utiliza para la inicialización de variables, la calibración de algunos sensores, etc.

-*Void Loop*- es una función de bucle infinito, y se inicia al acabar la función *Setup*. Aquí se escribe el programa principal, de forma que, una vez iniciado, repita la ejecución que se detalla en este apartado.

Fuera de estas funciones se pueden declarar variables globales, incluir librerías e incluso definir otras funciones.

Por último, cabe destacar que es de licencia libre, y que existe una web llamada [arduino.cc](http://arduino.cc) donde encontramos un gran respaldo de la comunidad e información sobre los distintos comandos, librerías y funciones del lenguaje.

### **3.10. LTspice XVII**

Este es un programa para la simulación y elaboración de circuitos analógicos. Incluye un visualizador de señales, el cual nos permite apreciar los valores y formas de las ondas en voltaje respecto al tiempo.

El programa fue creado y distribuido por la empresa Analog Devices, y es del tipo *freeware*, es decir, que el usuario no debe pagar por su uso.

El programa de base cuenta con distintos tipos de componentes electrónicos, pero, además, permite la adición de modelos de componentes que otras empresas diseñan para su prueba en LTspice. Asimismo, los valores de las componentes pueden modificarse en cualquier momento, pero requiere de resetear el visor para apreciar el cambio en las señales.

### **3.11. Octave**

Es un programa de *software* libre con licencia bajo el nombre de GNU General Public License (GPL). Se describe como un lenguaje de programación científico, con gran capacidad de cálculo matemático, con capacidad de elaboración de gráficos 2D/3D y con diferentes herramientas para mejorar su visualización. Es compatible con muchos *scripts* de Matlab y es funcional en los sistemas operativos de GNU/Linux, macOS, BSD y Microsoft Windows. La interfaz del programa permite la visualización de las variables y consta de una consola en la que podemos iniciar las funciones elaboradas en cualquier momento.

### **3.12. Paquete Office y Google Drive**

Entre los distintos programas que ofrece el paquete Office, los que utilicé para la elaboración de este proyecto fueron Excel y Word:

-Excel es un programa para gestionar datos y permite cálculos estadísticos con los mismos. La visualización de los datos es en tablas y gráficas que se elaboran a partir de la información que introducimos de manera sencilla.

-Word es un programa de edición de texto con el cual redactamos este trabajo.

También utilicé la herramienta de Google Drive de documentos, para compartir la redacción con el tutor y facilitar la corrección.

## **4. Implementación**

Nuestro principal objetivo es utilizar el músculo neumático como actuador para nuestro brazo protésico, por lo que lo primero que haremos es diseñar y realizar las pruebas pertinentes.

### **4.1. Diseño del músculo**

Como se comentó anteriormente, tomamos como referencia principal los modelos de músculo que encontramos en los trabajos de BioMakers Industries y de Hacksmith Industries, los cuales han documentado su trabajo en vídeo para la plataforma de Youtube. En estos vídeos se explican los materiales necesarios para su elaboración, el montaje, y el funcionamiento.

Para elaborarlo utilizaremos un tubo de goma de látex, ya que este material es capaz de inflarse con el aire a presión. Lo colocaremos dentro de un flexible de acero inoxidable trenzado, el cual permitirá la compresión del músculo al inflarse el tubo de goma en su interior. Esta compresión se debe al trenzado del flexible, que consigue aprovechar el esfuerzo de expansión del tubo, para acortar el músculo a lo largo. En los extremos colocaremos conectores neumáticos para permitir la entrada y salida de aire, y los sellaremos con cinta teflón. Con un par de abrazaderas sujetamos en torno a los conectores el flexible. En nuestro proyecto decidimos utilizar una sola entrada de aire, con lo que el extremo sin conector lo sellamos doblando el tubo y el flexible sobre sí mismos, sellándolos con una abrazadera. Queremos destacar que, *a posteriori*, colocar una anilla en este extremo nos permitiría un mejor punto de sujeción en el modelo del codo.

La elección del material del flexible y de su diámetro se deben a los resultados de las pruebas realizadas en el proyecto de Hacksmith Industries. En la documentación de su proyecto vemos una comparativa inicial de los materiales y dimensiones de los flexibles del músculo. Para realizar esta comparativa utilizan un banco de pruebas donde conectarán los músculos con sus distintos recubrimientos. Uno de los extremos del músculo se conectará al extremo corto de una palanca y en el extremo largo de esta se ubicará un sensor de fuerza que indicará en el ordenador las medidas tomadas. Además, se tomarán los datos de la retracción del músculo. Queremos recalcar que nosotros no hemos hecho las pruebas, hemos tomado directamente los resultados del proyecto de Hacksmith Industries.



El primer material de prueba es una manga de polietileno de  $\frac{3}{8}$  de pulgada de diámetro. Este modelo puede cargar con 11,34 Kg y alcanza una retracción del 15%. Con este material no podríamos imitar un músculo natural, debido a su baja capacidad de carga y que, en comparación, la retracción es bastante pequeña.

El segundo músculo es de poliéster, con un diámetro de  $\frac{1}{4}$  de pulgada. En este caso la fuerza útil es de 2,27 Kg, menor que el primer modelo, y la retracción es de 18%, la cual sigue siendo baja para lo que buscamos.

En tercer lugar emplean acero inoxidable trenzado, con un diámetro de  $\frac{3}{8}$  pulgadas. La capacidad de carga aumenta considerablemente, llegando a los 47,63 Kg. Además, permite una retracción del 32%, siendo características mucho más próximas a las de un músculo orgánico.

El siguiente es del mismo material que el anterior, pero, en este caso, utilizan un flexible de 1 pulgada. Este duplica la carga anterior, siendo capaz de levantar 92,99 Kg. La retracción es algo menor, de un 25%, pero son valores manejables para utilizarlo como actuador de nuestra prótesis.

Por último, prueban un recubrimiento de fibra de carbono de 1 pulgada de diámetro. Este material es más problemático para trabajar, debido a que puede provocar quemaduras en el usuario si no se utiliza la protección adecuada, por lo que se deben llevar guantes para manejarlo. Este modelo tiene la mayor capacidad de carga, de unos 95,25 Kg, pero su retracción es la menor, de solo un 10%.

Nuestra valoración tras ver los resultados del vídeo fue probar con el revestimiento de acero inoxidable, ya que ambos diámetros conseguían una retracción mayor que la del resto de materiales, y, además, nos permiten trabajar con pesos próximos a los que manejaría un músculo humano.

## **4.2. Sistema de sensado de la prolongación**

El siguiente problema que nos surgió fue cómo medir de manera continuada la prolongación del músculo. Lo primero que hicimos fue probar el propio músculo que habíamos fabricado, para saber su extensión máxima y mínima. El actuador lo diseñamos para que tuviera las dimensiones de un bíceps humano. Decidimos empezar por este músculo, ya que el movimiento del codo es de los más sencillos de un brazo. Por ello, la longitud del músculo completamente estirado es de unos 25.5 cm y, al alimentarlo mediante el aire del compresor, se redujo a unos 16.5 cm, lo que se corresponde con una retracción del 24%, próxima a los valores obtenidos en los resultados comentados en el apartado anterior.

Una vez tuvimos dimensionado nuestro actuador, decidimos crear un modelo experimental para realizar las pruebas. Como dijimos antes, el músculo representa al bíceps. Esta decisión se debe al movimiento más sencillo de este músculo en comparación

a otros. Por ello, decidimos replicar un codo mediante el uso de dos tablillas unidas por una bisagra.

Ya que se trata de un movimiento lineal de elongación, los sensores que podríamos utilizar deberán ser capaces de captar ese cambio de posición o de contracción, relacionándolo con un cambio en su resistividad o alguna otra variable eléctrica. Un potenciómetro es un buen ejemplo de ello, pero la medición se realizaría desde una cierta distancia, midiendo el ángulo de giro de una articulación, o mediante un mecanismo para medir el desplazamiento lineal. Otra opción más común en neumática son los sensores de posicionamiento magnéticos, que acompañan al émbolo en su trayectoria. Esto se debe a que el cilindro está fabricado en aluminio, siendo el émbolo de materiales ferromagnéticos, por lo que la implementación es muy sencilla. Lo ideal sería algún tipo de galga o similar que pueda ir pegada al músculo, aunque el problema surge debido a que el músculo es de acero, y podría alterar la corriente eléctrica. Por último, mi tutor propuso utilizar lo que se denomina como sensor de estiramiento flexible, el cual es una resistencia elástica que modifica su resistividad según la tensión de tracción a la que se someta.

### **4.3. Elección de sensores**

Con nuestro modelo experimental diseñado, decidimos buscar un sensor adecuado para este. Como hemos dicho, el modelo consiste en un codo, con dos tablillas de madera en las que podremos colocar los elementos del brazo y variar su posición cómodamente. Además, la madera es mala conductora eléctrica, por lo que no tendremos problemas de variación de corriente por la estructura del brazo.

El tutor propuso utilizar un tipo de resistencia variable en función de la elongación de esta, la cual llamaremos resistencia elástica, elastómero resistivo, o viceversa. Como se mencionó anteriormente, el valor de la resistencia varía según el esfuerzo de tracción al que es sometido, lo que podemos traducir en la elongación que sufre la resistencia. Sabiendo esto, podemos utilizarla en la misma ubicación que el actuador para captar la elongación o posición del músculo.

A la hora de colocarla, decidimos que compartiera posición con el músculo, por lo que, para evitar contacto con el flexible de acero, la colocamos en un lateral de las tablillas, a la misma altura que los extremos del actuador. Las uniones serían con chinchetas, para asegurar la correcta unión con el posterior sistema de control, además de para facilitar las pruebas y cambios de ubicación.

En teoría, este sensor debería permitirnos seguir el trazado completo del actuador, pero comprobamos que es muy dependiente de las condiciones ambiente. Nos dimos cuenta de este hecho conforme trabajamos con él, ya que los valores cambiaban debido a la temperatura y posiblemente a otras características del medio. Por ello, veremos que más adelante realizaremos una toma de medidas para comprobar la regularidad del sensor, además de para encontrar una función que describa el comportamiento de este.

#### 4.4. Circuito de adaptación de señales

A continuación necesitamos ser capaces de recibir la señal del sensor para obtener la información de posición del músculo. Por eso, diseñamos un circuito que dé como salida una relación en voltaje con la prolongación del elastómero en centímetros. El proceso de diseño lo realizamos en el programa LTspice, que nos permite simular los circuitos y medir los valores de salida que necesitamos. Antes de empezar a diseñarlo, tomamos las medidas de resistencia máxima (de aproximadamente  $1200\Omega$ ) y mínima (de aproximadamente  $400\Omega$ ) del sensor elástico, las cuales coinciden con las posiciones del modelo del codo de máxima y mínima extensión.

Para empezar utilizamos una conexión en puente de Wheatstone (Figura 4.4.1.), la cual nos permite saber el valor de una resistencia desconocida, que, en nuestro caso, es el elastómero.

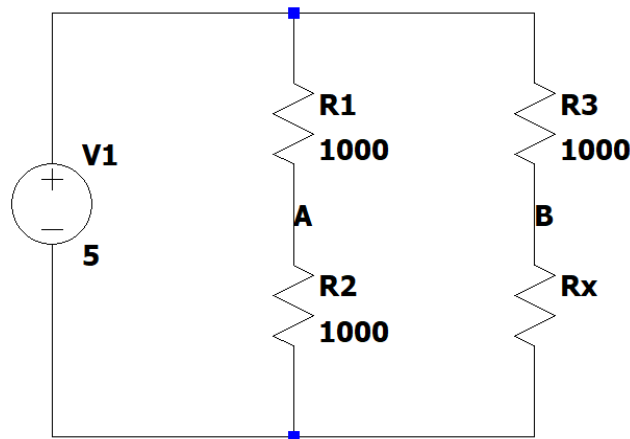


Figura 4.4.1. Puente de Wheatstone

En este circuito situamos a  $R_x$  como la resistencia de valor desconocido, en este caso, nuestro sensor. Las otras 3 resistencias  $R_1$ ,  $R_2$ ,  $R_3$  tienen el mismo valor (en nuestro puente de Wheatstone será de  $1000\Omega$ ). La fuente alimentará las resistencias con 5V, que es el voltaje de alimentación de la placa Arduino que usaremos más adelante. Normalmente, se utiliza una resistencia ajustable en el lugar de  $R_2$  para fijar el punto de equilibrio. Si la relación entre las resistencias  $R_1/R_2$  es igual a la de  $R_3/R_x$ , el voltaje entre los puntos A y B será nulo, por lo que podemos deducir que  $R_x$  será igual a  $R_2$ . Nosotros utilizaremos este circuito fijando  $R_2$ , por lo que, al variar el valor de resistivo de nuestro sensor, la salida entre los puntos A y B estará en función de la prolongación del elastómero, y, por tanto, del músculo.

La salida del puente nos da valores entre los  $-0,3V$  y los  $0,9V$ , coincidiendo con la máxima extensión y la máxima compresión respectivamente. Se podría trabajar con este rango de voltaje, no obstante, tenemos poca resolución a la hora de conectarlo al Arduino. Arduino es capaz de captar entradas de señales analógicas entre los 0 y los 5V, coincidiendo con una conversión digital entre los 0 y los 1024 bits. Como vemos, sería conveniente ampliar el rango para aprovechar el mayor número de *bits*, con lo que

aumentamos la resolución del sensor. Además, es conveniente desplazar el valor mínimo de tensión, ya que Arduino no captará las tensiones negativas, cortándolas a los 0V.

La solución que hemos propuesto para la adecuación de la señal consiste en desplazar los valores a un valor medio de 2,5V, con lo que tendríamos 2,5V por arriba y por abajo en los que amplificar la señal para aumentar la resolución.

Para realizar estos cambios hemos utilizado dos amplificadores operacionales (Figura 4.4.4.). El primero está en configuración de seguidor de tensión (Figura 4.4.2.) para conseguir un voltaje constante de 2,5V, el cual consideramos como el punto medio de extensión del músculo.

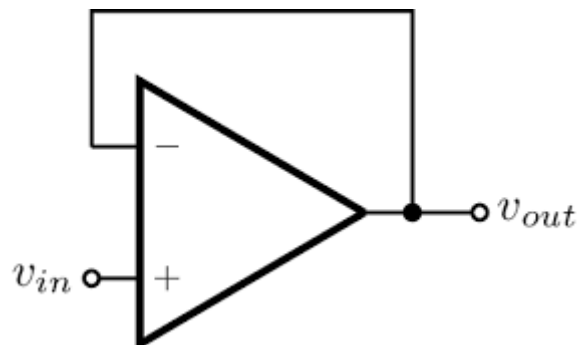


Figura 4.4.2. Amplificador en configuración de seguidor de tensión

La entrada positiva de este amplificador se conecta con un partidor de tensión que divide a la mitad la tensión de alimentación de 5V de Arduino. El segundo amplificador está en configuración no inversora, con su entrada positiva, donde normalmente se conecta a tierra, conectada a los 2,5V de salida del primer amplificador, desplazando la salida al valor medio que buscamos. La salida negativa se conecta a otro partidor de tensión entre la una resistencia de 850Ω y el sensor elastómero. La amplificación de este operacional se calcula por la siguiente fórmula (Figura 4.4.3.):

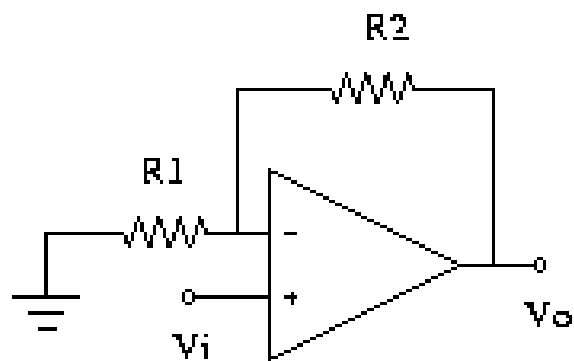


Figura 4.4.3. Amplificador en configuración no inversora (esquema de ejemplo)

$$V_o/V_i = \frac{R_1 + R_2}{R_1}$$

dónde R1 es la primera resistencia por la izquierda, y R2 la resistencia que une la entrada positiva y la salida del operacional. La amplificación corresponde con el voltaje de salida entre el voltaje de entrada, como podemos ver en la fórmula anteriormente descrita.

A continuación podemos ver el circuito, donde R1 es nuestro sensor, y la amplificación de nuestro segundo operacional es de:

$$A = \frac{1000\Omega + 2500\Omega}{1000\Omega} = 3.5$$

Por lo tanto, nuestro circuito conseguirá aumentar el rango de nuestro sensor hasta 3.5 veces más, teniendo en cuenta que antes el rango de medida era de 1.2V en total.

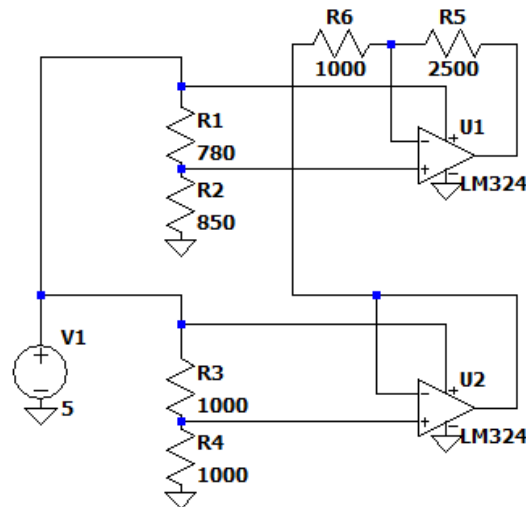


Figura 4.4.4. Circuito de adecuación de la señal

Con el circuito ya preparado, vamos a realizar una simulación de la señal a la salida del segundo amplificador, tomando diferentes valores del sensor dentro del rango del modelo del codo. En la siguiente gráfica (Figura 4.4.5.), por orden podemos ver en verde cuando el sensor tiene un valor de 400Ω, en azul cuando mide 600Ω, en rojo para los 780Ω, y en turquesa para los 1200Ω. En la figura 4.4.6. podemos apreciar los circuitos con la resistencia sensora modificada.

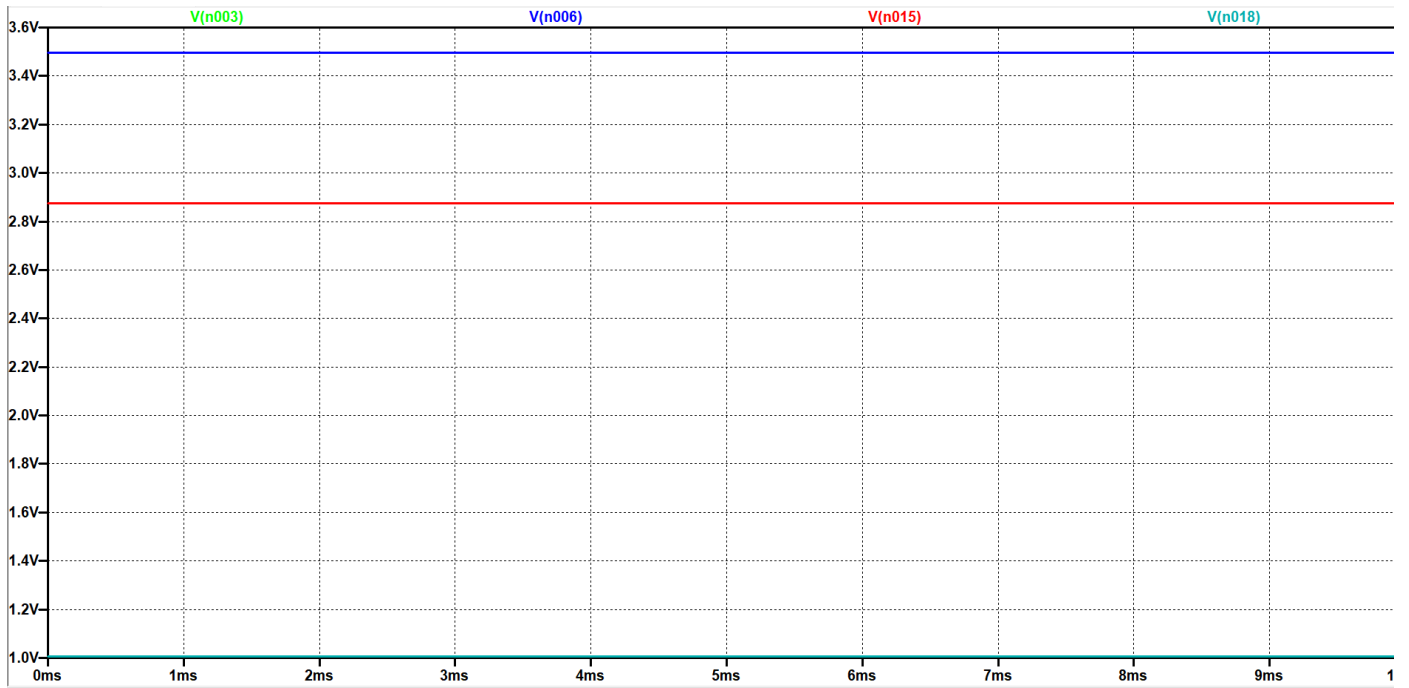


Figura 4.4.5. Señal de salida según los valores resistivos del sensor

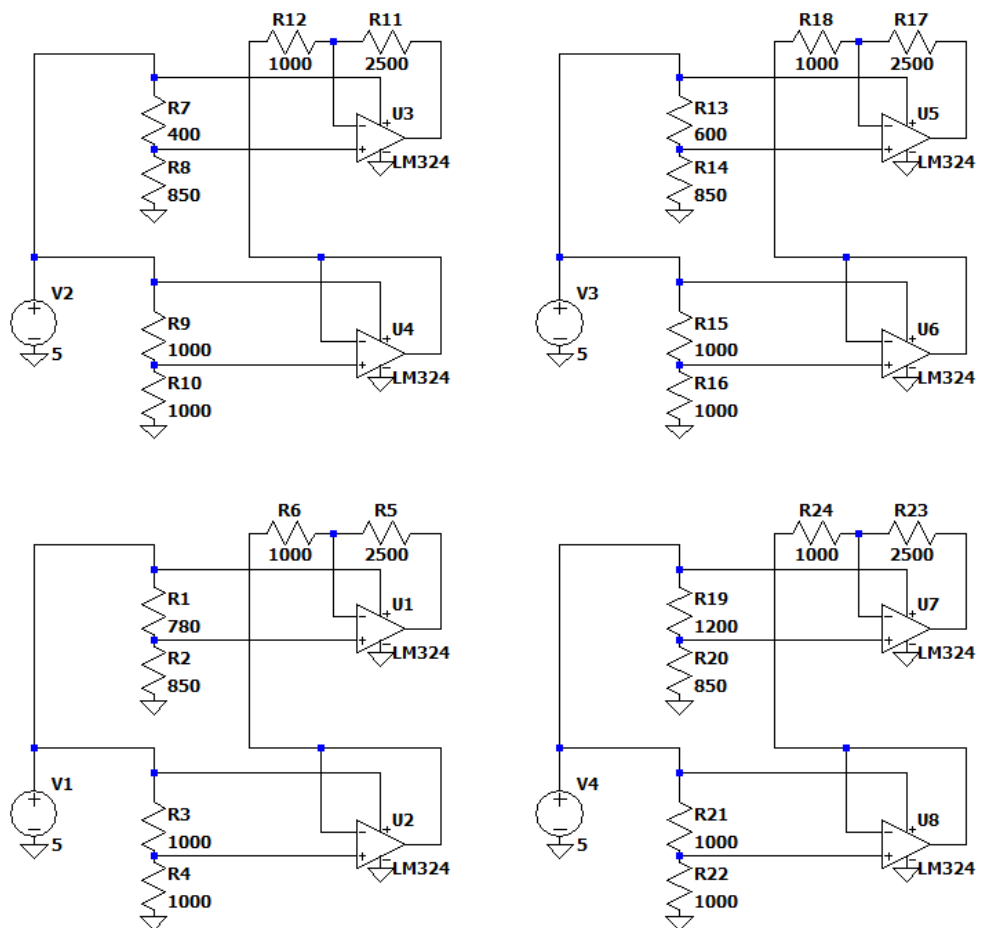


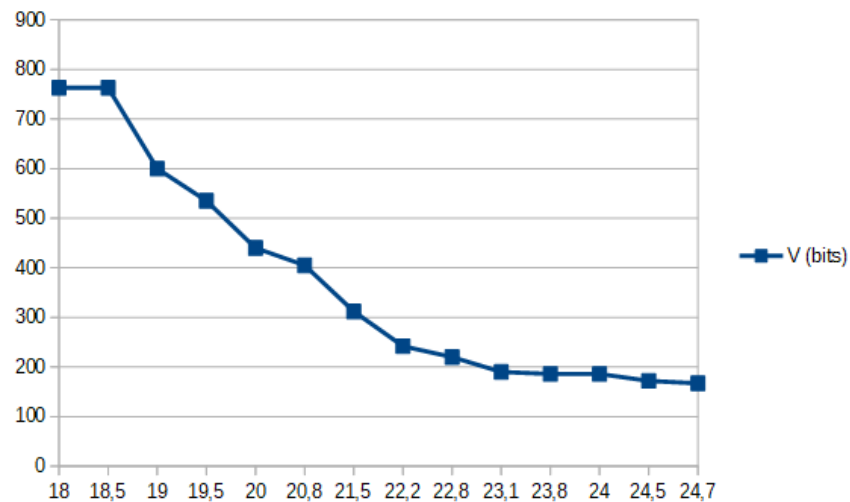
Figura 4.4.6. Circuitos de prueba con el rango de valores del sensor

#### 4.5. Adecuación del sensor y modelo de Steinhart-Hart

Como se mencionó anteriormente, nuestro sensor muestra ciertas irregularidades debido a las características del entorno, entre ellas la temperatura. Esto es debido a que el sensor de estiramiento flexible está fabricado en caucho impregnado en carbono, que, como la mayoría de gomas, se deforman debido a cambios de temperatura. Para comprobar la regularidad, decidimos realizar una toma de medidas durante los días, apuntando el voltaje de salida del circuito en función de la elongación del sensor, además de la hora y temperatura del momento de la toma de datos. En la siguiente tabla de Excel (Figura 4.5.1. a 4.5.3.), podemos ver algunas mediciones para ejemplificar:

L (cm)	Codo (°)	V (bits)		L (cm)	Codo (°)	V (bits)
18	43,31	763		18	43,31	763
18,5	51,87	763		18,5	51,87	704
19	59,72	600		19	59,72	551
19,5	67,17	535	20/05/21	19,5	67,17	465
20	74,42	440		20	74,42	405
20,8	85,90	405		20,5	81,58	363
21,5	96,16	312		21	88,79	320
22,2	106,98	242		21,5	96,16	244
22,8	117,07	220		22	103,80	210
23,1	122,56	190		22,5	111,90	176
23,8	137,43	186		23	120,69	153
24	142,57	186		23,5	130,60	140
24,5	160,15	172		24	142,57	131
24,7	180,00	167		24,5	160,15	136

Figura 4.5.1.A Tablas de medidas 20/05



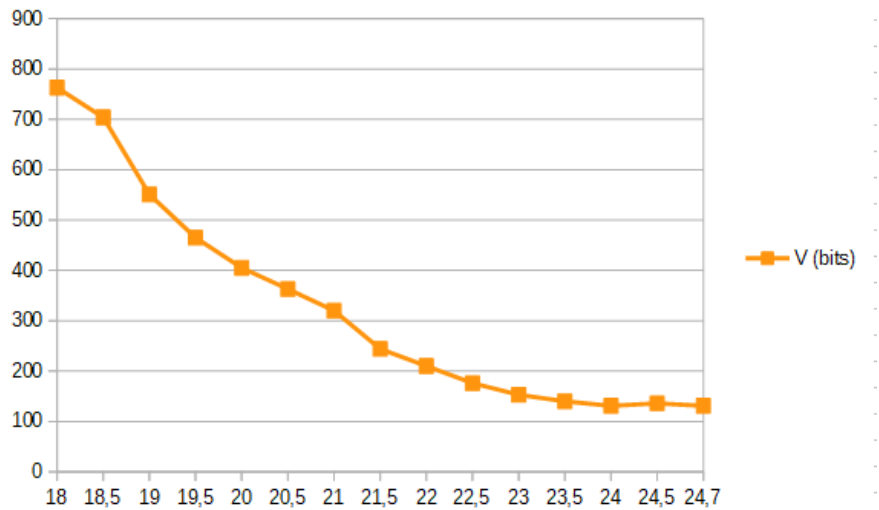


Figura 4.5.1.B Gráficas de voltaje respecto elongación 20/05

			21/05/21			
L (cm)	Codo (°)	V (bits)	19:15:00	L (cm)	Codo (°)	V (bits)
18	43,31	763	21° aprox	18	43,31	764
18,5	51,87	640		18,5	51,87	700
19	59,72	580		19	59,72	616
19,5	67,17	440		19,5	67,17	490
20	74,42	410		20	74,42	461
20,5	81,58	330		20,5	81,58	381
21	88,79	306		21	88,79	336
21,5	96,16	260		21,5	96,16	295
22	103,80	204		22	103,80	257
22,5	111,90	195		22,5	111,90	251
23	120,69	183		23	120,69	198
23,5	130,60	156		23,5	130,60	196
24	142,57	160		24	142,57	168
24,5	160,15	160		24,5	160,15	156
24,7	180,00	155		24,7	180,00	140

Figura 4.5.2.A Tablas de medidas 21/05



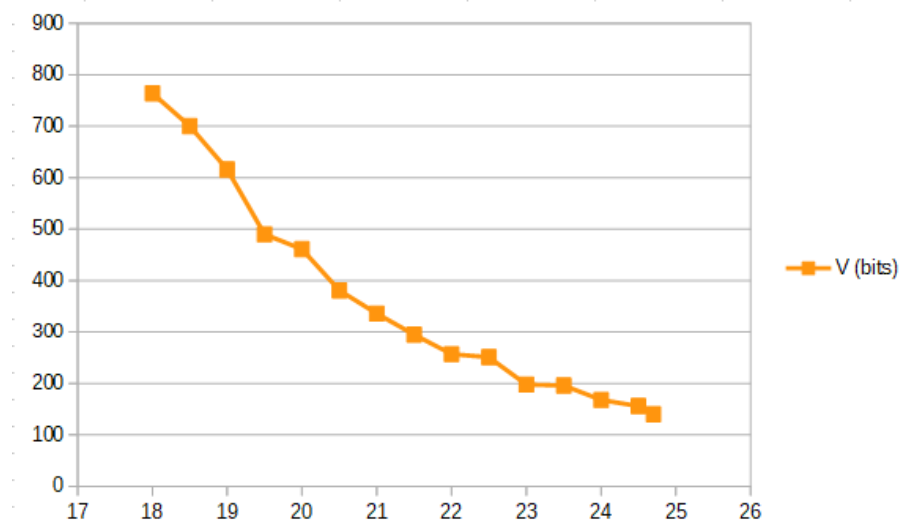
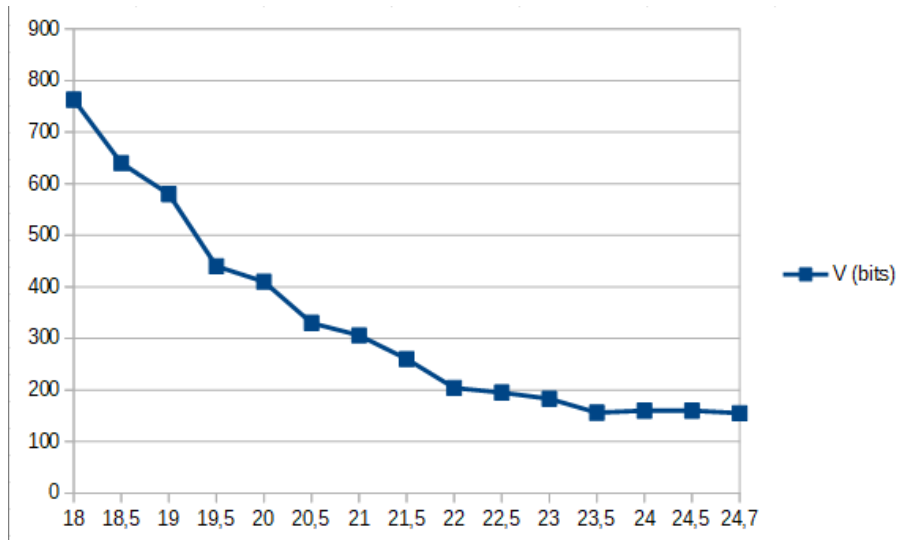


Figura 4.5.2.B Gráficas de voltaje respecto elongación 21/05

			24/05/21			
L (cm)	Codo (°)	V (bits)	20:40:00	L (cm)	Codo (°)	V (bits)
18	43,31	760	23°	18	43,31	762
18,5	51,87	756		18,5	51,87	762
19	59,72	634		19	59,72	711
19,5	67,17	528		19,5	67,17	604
20	74,42	456		20	74,42	456
20,5	81,58	387		20,5	81,58	395
21	88,79	326		21	88,79	334
21,5	96,16	318		21,5	96,16	302
22	103,80	269		22	103,80	264
22,5	111,90	251		22,5	111,90	233
23	120,69	250		23	120,69	224
23,5	130,60	207		23,5	130,60	201
24	142,57	182		24	142,57	196
24,5	160,15	179		24,5	160,15	181
24,7	180,00	177		24,7	180,00	173

Figura 4.5.3.A Tablas de medidas 24/05

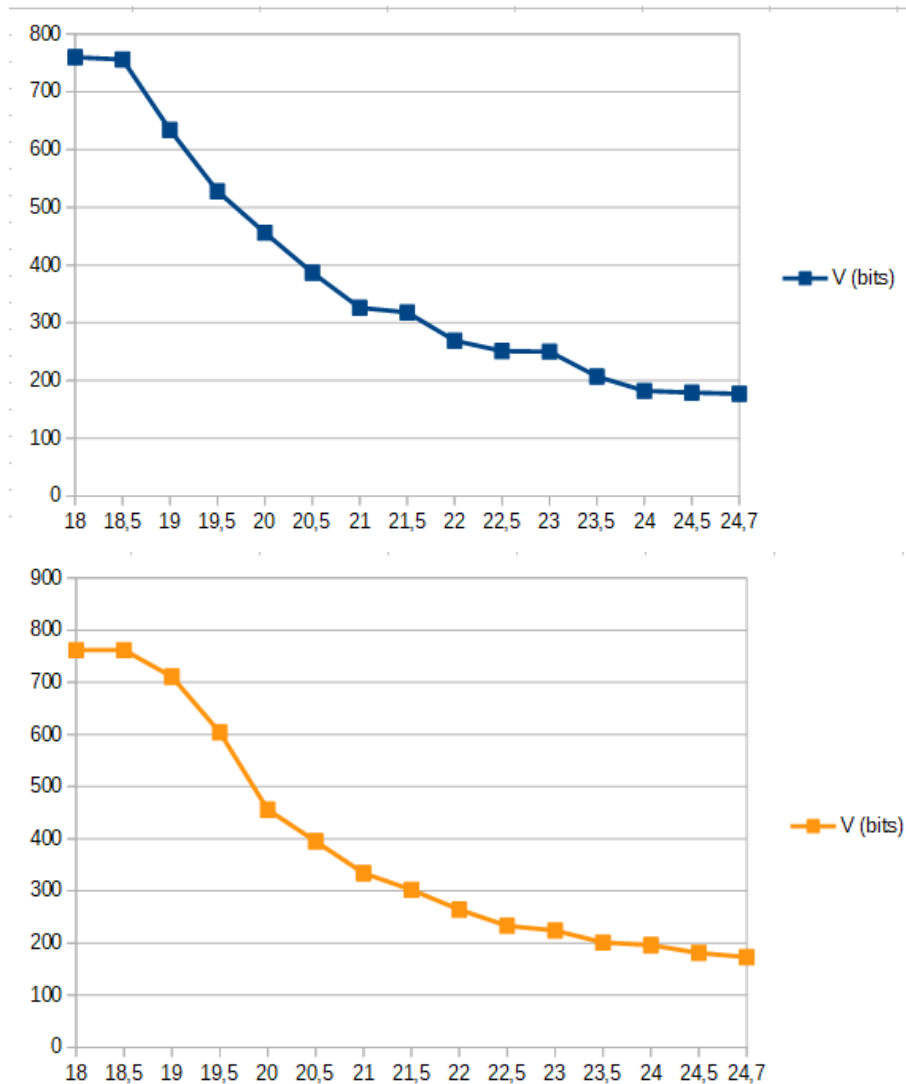


Figura 4.5.3.B Gráficas de voltaje respecto elongación 24/05

Como podemos apreciar en las gráficas, el sensor es bastante irregular, incluso durante las tomas de medida realizadas el mismo día y a la misma hora. Debido a esto, decidimos efectuar dos arreglos: realizar una calibración previa al uso del brazo y definir una función para determinar la evolución del movimiento del brazo.

Para definir una función, decidimos utilizar Octave para realizar la comparación entre las gráficas de los datos que hemos tomado y la gráfica generada por la ecuación que describimos. Lo primero fue realizar la representación de las gráficas de datos de voltaje de salida del circuito respecto a la longitud del sensor, para hacernos una idea de que tipo de ecuación tiene un trazado similar. Nos dimos cuenta de que el comportamiento era muy similar al de un termistor. Los termistores son un tipo de resistencia que varía en resistividad según la temperatura y, concretamente, están elaborados por un tipo de semiconductor.

La ecuación de Steinhart-Hart describe el modelo de la resistencia de un semiconductor a diferentes temperaturas, por lo que normalmente se utiliza para definir la función de los termopares. La ecuación es de tercer grado y se describe de la siguiente manera:

$$\frac{1}{T} = A + B \ln R + C (\ln R)^3$$

dónde T es la temperatura en grados kelvin, R es la resistencia para una temperatura determinada en ohmios, y A, B y C son los denominados coeficientes de Steinhart-Hart, que dependen del termistor y el rango de temperatura de interés.

Para calcular los coeficientes de Steinhart-Hart necesitamos conocer tres puntos de operación del modelo, es decir, tres valores de resistencia con sus respectivos valores de temperatura relacionados. Esto se describe de manera matricial:

$$\begin{bmatrix} 1 & \ln R_1 & \ln^3 R_1 \\ 1 & \ln R_2 & \ln^3 R_2 \\ 1 & \ln R_3 & \ln^3 R_3 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \frac{1}{T_1} \\ \frac{1}{T_2} \\ \frac{1}{T_3} \end{bmatrix}$$

Esto se puede expresar con el siguiente desarrollo de ecuaciones, donde T1, T2 y T3 son los datos de temperatura, y R1, R2 y R3 los datos de la resistencia:

$$L_1 = \ln R_1, L_2 = \ln R_2, L_3 = \ln R_3$$

$$Y_1 = \frac{1}{T_1}, Y_2 = \frac{1}{T_2}, Y_3 = \frac{1}{T_3}$$

$$\gamma_2 = \frac{Y_2 - Y_1}{L_2 - L_1}, \gamma_3 = \frac{Y_3 - Y_1}{L_3 - L_1}$$

$$C = \left( \frac{\gamma_3 - \gamma_2}{L_3 - L_2} \right) (L_1 + L_2 + L_3)^{-1}$$

$$B = \gamma_2 - C (L_1^2 + L_1 L_2 + L_2^2)$$

$$A = Y_1 - (B + L_1^2 C) L_1$$

Con los coeficientes de Steinhart-Hart definidos, podremos describir la ecuación de Steinhart-Hart que mencionamos al principio de este apartado.

También es posible utilizar la inversa de la ecuación para conocer la resistencia de un semiconductor a una temperatura dada, pero, como hemos dicho antes, para utilizar la ecuación de Steinhart-Hart es necesario definir sus coeficientes, los cuales son los que describen el sistema:

$$R = \exp(\sqrt[3]{y - x/2} - \sqrt[3]{y + x/2})$$

dónde

$$x = \frac{1}{C} \left( A - \frac{1}{T} \right)$$

$$y = \sqrt{\left(\frac{B}{3C}\right)^3 + \frac{x^2}{4}}$$

Cabe destacar que suele simplificarse la ecuación de Steinhart-Hart, reescribiéndola como una exponencial de primer orden:

$$R(T) \approx ae^{-bT} + c$$

Para definir nuestro modelo de Steinhart-Hart para nuestro sensor, tendremos que sustituir en las ecuaciones la resistencia ( $\Omega$ ) y la temperatura ( $^{\circ}\text{C}$ ), que normalmente son las medidas de un termistor, por el voltaje (en bits) y la elongación (cm) del sensor elastómero, respectivamente:

$$\frac{1}{L} = A + B \ln V + C(\ln V)^3$$

Una vez descrito el modelo de Steinhart-Hart, podemos proseguir con las comprobaciones en Octave. Realizamos el cálculo de la función mediante las muestras de datos que habíamos tomado. Con ello, podemos comparar las gráficas de los datos, con las gráficas del cálculo teórico por la fórmula, para lo que describimos el siguiente código (Figura 4.5.4.):

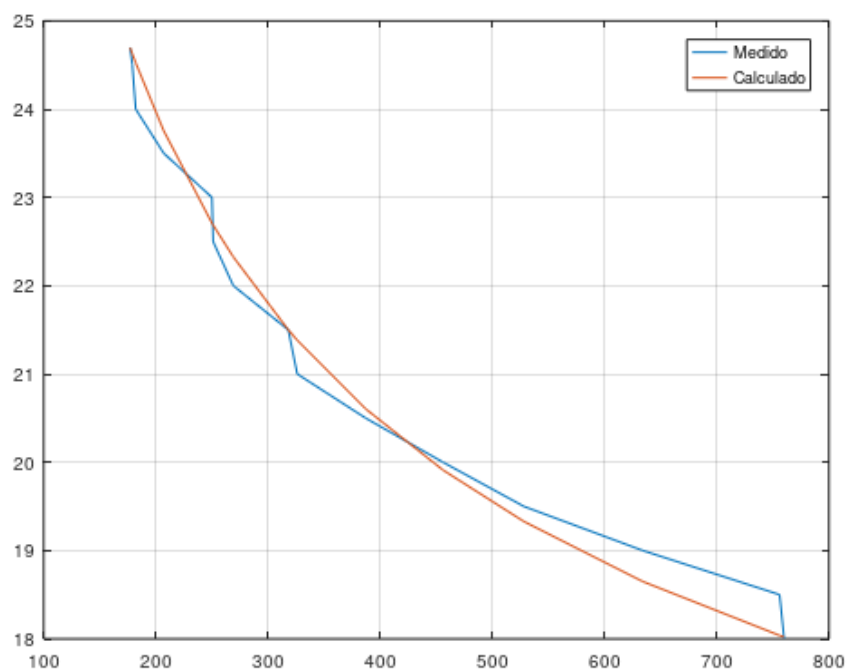
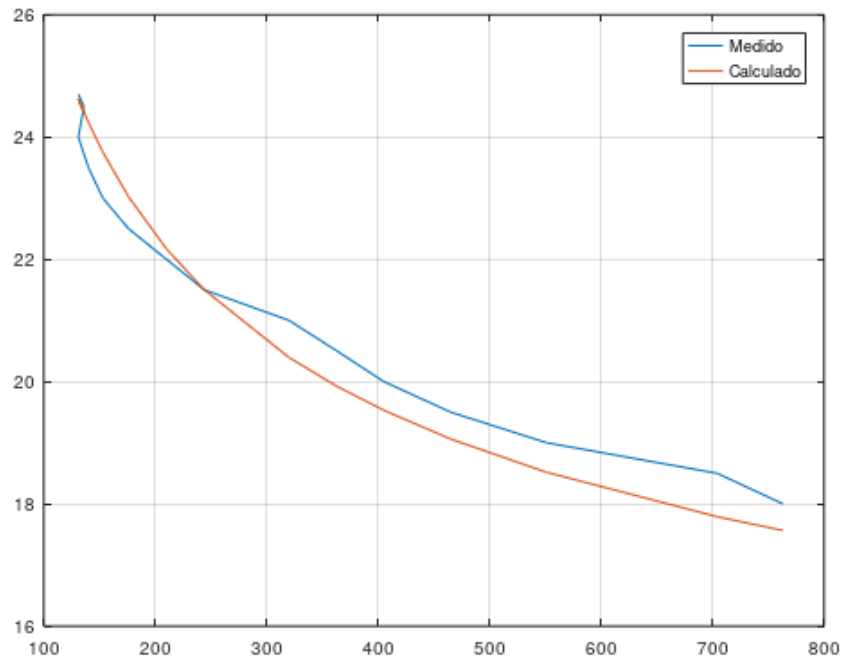
```

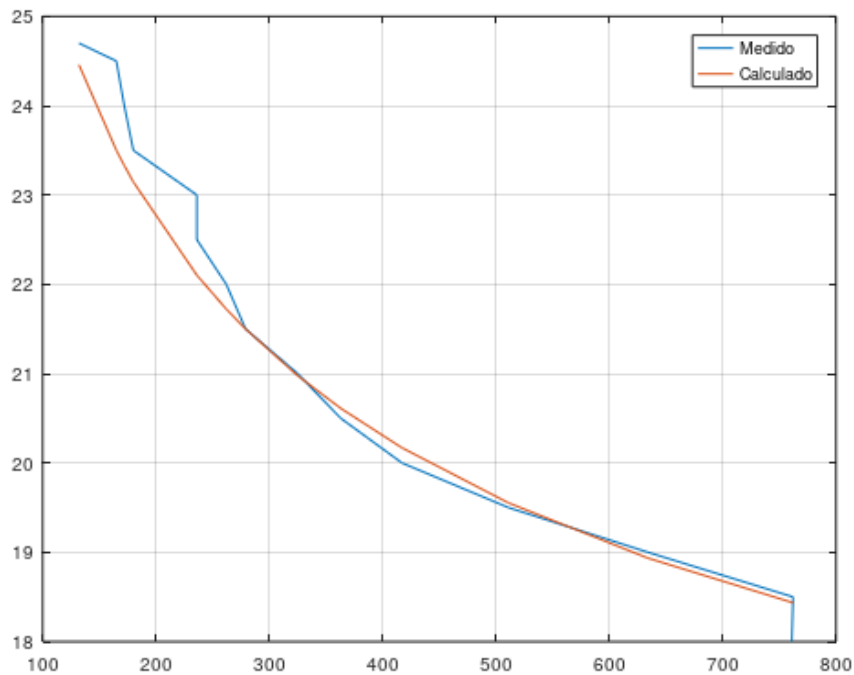
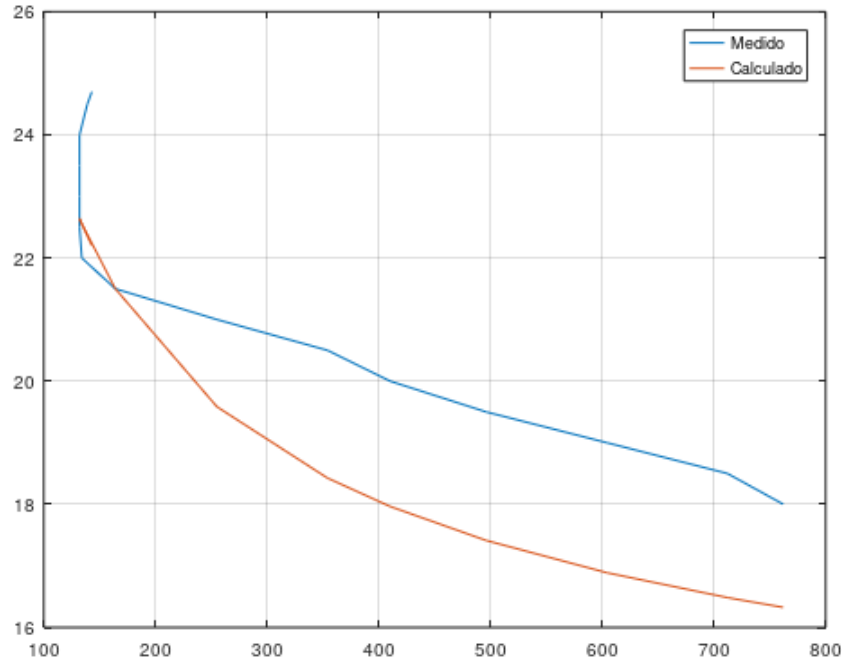
248 T = [T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2];
249 R = [R2 R3 R4 R5 R6 R7 R8 R9 R10 R11 R12 R13 R14];
250
251 for i=1:13,
252
253     ADV=i
254
255     L1=log(R(1,i))
256     L2=log(R(8,i))
257     L3=log(R(15,i))
258     Y1=1/T(1,i);
259     Y2=1/T(8,i);
260     Y3=1/T(15,i);
261
262     ganma2=(Y2-Y1)/(L2-L1)
263     ganma3=(Y3-Y1)/(L3-L1)
264
265     C=((ganma3-ganma2)/(L2-L1))/(L1+L2+L3)
266
267
268     B=ganma2-C*(L1^2)+L1*L2+(L2^2)
269     A=Y1-(B+(L1^2)*C)*L1
270     t = A + B* log(R(8,i)) + C*(log(R(8,i)^3));
271     diff = T(8,i) - 1/t
272
273
274     for j=1:15,
275         t = A + B* log(R(j,i)) + C*(log(R(j,i)^3));
276         calT(j) = 1/t + diff;
277
278     end
279
280     figure(i)
281     plot(R(:,i),T(:,i),R(:,i),calT)
282     legend('Medido','Calculado');
283
284 end
285
286

```

Figura 4.5.4. Código para el cálculo y graficación del modelo de Steinhart-Hart en Octave

Donde los matrices T y R corresponden a las tomas de medida de los valores de elongación y tensión. Estas se representarán en 13 gráficas distintas junto con la gráfica calculada de tensión para los mismos puntos de elongación. Algunos ejemplos (Figuras 4.5.5. a 4.5.9.) representativos son los siguientes:





Figuras 4.5.5. a 4.5.9. Gráficas de la toma de medidas frente al cálculo por Steinhart-Hart

Las gráficas calculadas que hemos visto se ajustan a los datos que hemos tomado del sensor, aunque debemos destacar el caso de la tercera imagen que hemos mostrado, en la que sí que hay una variación importante de los valores obtenidos en la medida. Creemos que no se trata más que de un error en la toma de valores, ya que en las otras 12 gráficas el comportamiento es muy próximo a la función de Steinhart-Hart. Nuestra decisión fue utilizar este modelo, para el cual tendremos que realizar una calibración previa a cada inicio del uso del brazo.

La calibración que hemos nombrado consistirá en la lectura de 3 parejas de valores de tensión (en *bits*) por ciertas elongaciones (en *cm*) determinadas, concretamente 16.5, 22,

25.5, para tomar los dos puntos extremos y uno intermedio, lo que favorece una buena representación de la función. La forma de realizar la calibración se explicará en el apartado 6.

#### 4.6. Adecuación de la lectura del sensor a ángulos, teorema del coseno

La lectura que obtenemos tras el circuito de adecuación de la señal y la modelización de Steinhart-Hart, hace referencia a la longitud del sistema respecto al voltaje de entrada a Arduino. Esto quiere decir que, cuando queramos saber en qué posición se encuentra el codo, lo que nos mostrará en pantalla será la longitud del sensor, no el ángulo de apertura del brazo, que es más intuitivo a la hora de visualizar su correcto funcionamiento. Por ello, hemos aplicado la trigonometría al modelo experimental (Figura 4.6.1.) para relacionar la longitud del sensor con el ángulo del codo.



Figura 4.6.1. Esquema del modelo experimental acotado

En esta primera imagen podemos ver que el antebrazo estará unido con el sensor elástico (H) a 20,7 cm del codo (C). El brazo estará unido con el sensor a 5 cm del codo.

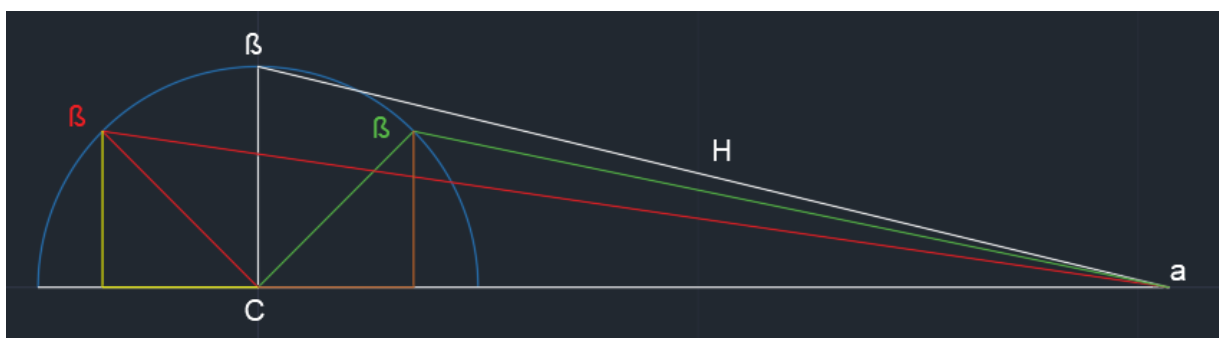


Figura 4.6.2. Esquema del modelo experimental sin cotas

Como hemos dicho antes, la medida C equivale al ángulo del codo, y la H es la elongación del elastómero.  $\alpha$  y  $\beta$  son los ángulos que se forman entre el antebrazo y el sensor, y el brazo y el sensor, respectivamente. En el dibujo podemos ver las distintas posiciones que tomará el brazo respecto del antebrazo, siendo el semicírculo azul, el trazo del brazo, y las líneas en rojo y verde dos progresiones distintas de dicho trazado como ejemplo.

Utilizando trigonometría sobre el modelo, podemos ver que el teorema del coseno resuelve la relación entre la longitud de H, y el ángulo del codo (C).

$$H^2 = A^2 + B^2 - 2AB \cos(C)$$

Sustituyendo las dimensiones del brazo (A) y del antebrazo (B), y despejando el ángulo del codo:

$$H^2 = 5^2 + 20.5^2 - 205 \cos(C)$$

$$C = \frac{\arccos(5^2 + 20.7^2 - H^2)}{207}$$

Con esta fórmula, podemos deducir cómo evolucionará la posición del codo (C en grados) según la elongación (H en cm) del elastómero, y viceversa.

#### 4.7. Sistema de control

Lo que nos queda para empezar a trabajar con el modelo experimental es realizar un programa para el control del brazo. Lo primero será definir nuestro actuador, el cual es el músculo neumático, y el sensor que nos aporte la información de la posición de este, es decir, el sensor elastómero.

Nuestro actuador no funciona por sí mismo, sino que utiliza dos electroválvulas, una para la entrada de aire y otra para la salida. Estas válvulas funcionan a 12V, por lo que para poder controlarlas desde Arduino, el cual tiene una alimentación de 5V, necesitaremos utilizar unos transistores para activar una alimentación externa con la señal de control. Para esta alimentación utilizaremos una fuente externa de laboratorio. El circuito es bastante simple (Figura 4.7.1), y utilizaremos uno para cada electroválvula:

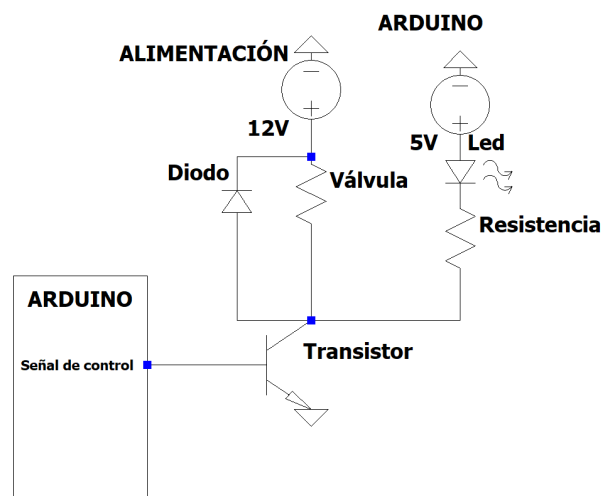


Figura 4.7.1. Circuito de control para las electroválvulas



Como vemos en la imagen, la alimentación de 12V se conectará a la válvula, y se hará uso de un diodo para evitar picos de corriente, puesto que las cargas inductivas pueden provocarlos. Estos elementos se conectarán al colector del transistor. Al colector también se conectará un led junto con una resistencia en serie (es necesaria la resistencia para evitar que el led se queme por la tensión), alimentado con el voltaje de 5V de Arduino, para poder comprobar de manera visual qué válvula está activa. El emisor estará directamente conectado a tierra. A la base del transistor se conecta, con una resistencia para reducir la tensión, la señal digital de control de Arduino, que cuando sea "0" no dejará pasar la corriente entre colector y emisor, y cuando sea "1" cerrará el circuito, activando la electroválvula. La tierra será común entre la fuente y el Arduino, por lo que conectaremos ambas al circuito.

Como se ha mencionado, la señal de control será una salida digital de Arduino, ya que las electroválvulas son del tipo "todo o nada". Este tipo de actuador nos lleva a utilizar un control "todo o nada", ya que las válvulas solo tendrán dos estados posibles: dejar pasar el aire o sacarlo, según cuál se active (la activación de una válvula implica la desactivación de la otra; esto es para evitar pérdidas de aire). Debido a esta situación, no tiene sentido un control más gradual y complejo, porque las válvulas no posibilitan caudales ajustables. Además, el único tipo de control viable para un actuador digital, es el "todo o nada".

El control "todo o nada" consiste en que, cuando la medición esté por debajo de la consigna, se aplique la acción del actuador (no de manera gradual), y cuando esté por encima, detener la acción del actuador (o viceversa, según el efecto en la medición del actuador).

Este tipo de control no provoca infinitas activaciones del actuador, debido a que la respuesta del sistema no es instantánea, además de que el actuador tiene cierta inercia. También es recomendable tener un intervalo como consigna en lugar de un valor concreto, ya que si se supera o se baja del valor, se provocará muchas activaciones del actuador, que puede provocar una avería. Al utilizar un intervalo de valores aceptables en torno a dicha consigna, lo que también denominamos histéresis, conseguimos un mayor tiempo de descanso del actuador. Esto funciona de la siguiente manera: estamos por debajo del máximo valor del intervalo de consigna y activamos el actuador para alcanzarlo; una vez alcanzado, se desactiva el actuador, por lo que la medición bajará; si se alcanza el mínimo del intervalo se activará el actuador de nuevo. Como vemos, el objetivo es estar dentro del intervalo, lo que logra menos activaciones/desactivaciones del actuador.

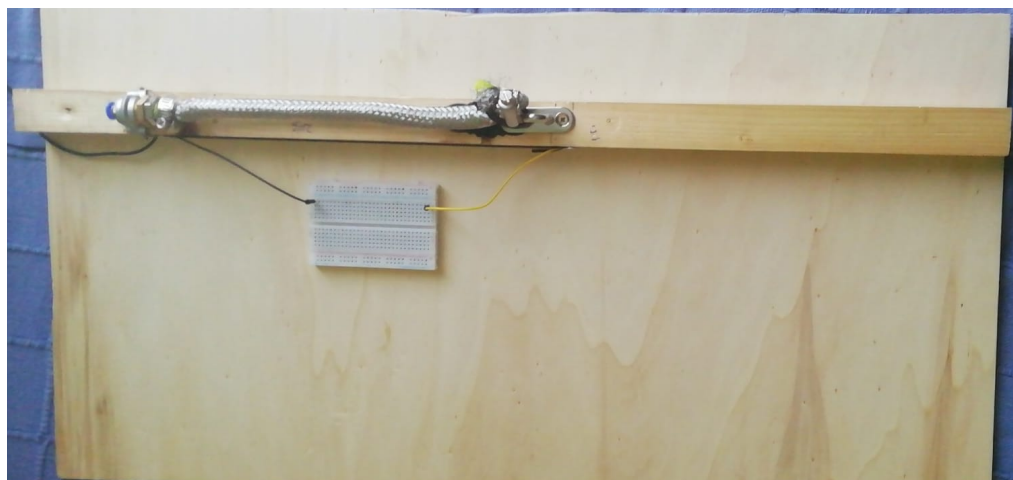
El principal problema de este tipo de control es que se introduce un mayor rango de oscilación del sistema, lo que se refleja en que el brazo pueda quedar en una consigna en la que haga algo que podríamos describir como "rebotes" en torno a dicho punto. Esto se debe a los múltiples encendidos de las válvulas, lo que afecta al tiempo de vida útil de estas. Además, no es posible modular la salida respecto al error de la entrada del sensor, ya que no podemos regular la salida, solo activar o desactivar los actuadores.

Para terminar, queremos destacar que la sencillez de este tipo de control nos permite implementarlo, fácilmente, en una placa Arduino. Concretamente, utilizamos la función *if* para generar 3 condicionales según estemos por debajo, dentro o por encima del intervalo de consigna, entrando a cada bucle únicamente si la lectura del sensor analógico

se encuentra dentro del intervalo. Este intervalo lo calculamos en el *script*, según la consigna que introducimos por el teclado al programa, y podemos modificar en el código, según veamos oscilaciones, las dimensiones del intervalo.

## 5. Elaboración del modelo experimental y decisiones de diseño

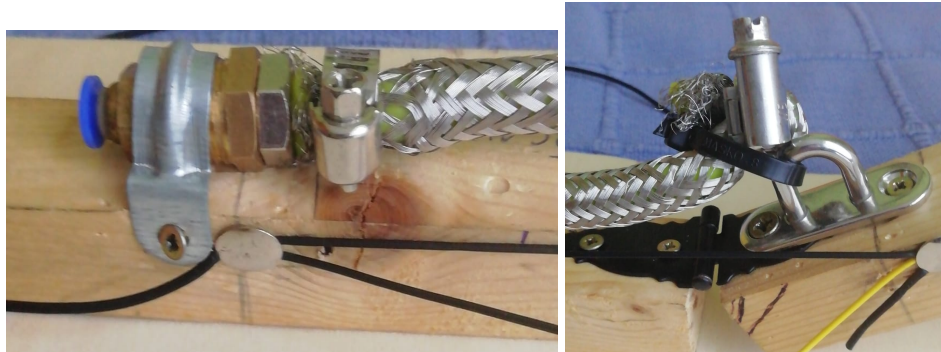
Tomamos en consideración fabricar un modelo experimental del brazo, pero, para realizar las primeras pruebas, y debido a la complejidad del proyecto, tomamos como inicio la articulación del codo, ya que nos permite un amplio rango de movimiento para las pruebas, y podemos trabajar con un único músculo. La presentación (Figura 5.1.) de este se hará sobre una plancha de madera, en la que colocaremos tanto el brazo como el circuito, y las electroválvulas.



*Figura 5.1. Presentación del modelo experimental*

El modelo se basa en dos tablillas de madera, ya que este material permite que cambiemos de posiciones las piezas sin mucha dificultad, además de que no dejará pasar la electricidad entre los componentes que coloquemos. Ambas tablillas se unen con una bisagra, imitando a un codo con el brazo y el antebrazo. El músculo está colocado siguiendo la posición del bíceps, lo que nos permite, además de imitar a un brazo humano, aumentar el tramo de giro del codo. Esto es importante, ya que, como hemos dicho, el músculo se retrae en un 24% de su extensión base, por lo que colocarlo más alejado del eje de giro provocaría un ángulo de desplazamiento mucho menor. Además, el músculo queda más pegado a la estructura, quedando un brazo con mayor manejabilidad.

El problema viene a la hora de controlarlo, puesto que al ser neumático es más complicado de ajustar a posiciones exactas, y ajustar el brazo a un ángulo concreto implica retracciones muy pequeñas y mantenidas del músculo. Las uniones del músculo serán dos: una fija en el antebrazo, donde estará ubicada la entrada y salida del aire; y la unión en el brazo será móvil (Figuras 5.2. y 5.3.) para permitir el correcto movimiento del codo.



*Figura 5.2. Unión fija del músculo / Figura 5.3. Unión móvil del músculo*

La resistencia elástica, que, como hemos dicho, es nuestro sensor, está sujeta en puntos próximos a las uniones del músculo, pero por un lateral para evitar el contacto. Esto sigue la misma lógica que con el músculo, pues el elastómero también tiene un límite elástico y colocarlo más alejado del eje de giro podría provocar que se rompiera. Con el sensor también nos surge un problema: los ángulos de mayor extensión provocan una variación muy pequeña de la resistencia, y, con ello, de la tensión de control que llega al Arduino. Esto complica que el brazo sea muy preciso. Para remediar este problema, desplazamos ligeramente la unión con el brazo (refiriéndonos a la parte donde se encuentra el cúbito y el radio), consiguiendo que los ángulos de mayor extensión tengan una variación en la longitud del elastómero más suave. Este desplazamiento ha sido de 1 cm, debido a que al probar valores más alejados el elastómero se tensaba demasiado, llegando a romperse.

Como podemos ver en la imagen siguiente (Figura 5.4.), es un modelo a escala real, con la idea en un inicio de añadir al sistema un sensor electromiográfico, el cual nos permitiría controlar el músculo con los impulsos nerviosos, pero no pudimos abarcar tanto en este proyecto.



*Figura 5.4. Vista del modelo de codo*

También pensamos en utilizar como alimentación neumática cartuchos de CO2 comprimido (Figura 5.5.), pero, debido a la necesidad de realizar pruebas y al costo de estos, trabajaremos con un compresor con cámara de aire.



Figura 5.5. Cartuchos de CO2 de 16 gramos

## 6. Diseño de código

El programa se centra en el control de las dos electroválvulas que provocarán la extensión o compresión del músculo, mediante la información del sensor elastómero. Como hemos explicado en el apartado sobre el *software* de Arduino, el código se divide en dos funciones principales: *void setup* y *void loop*. También definiremos las variables globales y llamaremos a las librerías fuera de estas dos funciones principales (Figura 6.1.).

```
//librerías
#include <stdio.h>
#include <math.h>

//entradas y salidas analógicas
int posM = A0;
int lim = 0;

//entradas y salidas digitales
#define evco 2
#define evrel 3

//declaración de variables
float consigna;
float comax;
float comin;
float pres = 10;
float calibE;
float calibM;
float calibC;
float RM = 22;
float RE = 25.5;
float RC = 16.5;
float L1, L2, L3, Y1, Y2, Y3, GAN2, GAN3, A, B, C;
float T;
float t;
float diff;
float x = 5;
float y = 20.7;
```

Figura 6.1. Declaración de variables y librerías

En la función *setup* está descrita la calibración del sensor (Figura 6.2.), para lo cual tendremos que ajustar en ciertas posiciones el codo manualmente, tomando los valores del

sensor en voltaje de 3 puntos determinados (extensión completa de 16.5cm, punto intermedio de 22cm, compresión completa de 25.5cm). La calibración se lleva a cabo manualmente debido a la dificultad de colocar el brazo en un punto medio concreto, cosa que no sucede con los puntos extremos que son límites físicos del brazo. Utilizaremos 3 variables para guardar los valores de calibración (calibE, calibM y calibC), los cuales guardarán los valores tras introducir cualquier carácter por teclado. Cuando se haya guardado el valor de medida, se nos indicará mediante unos leds, para que podamos colocar el brazo en la siguiente posición. Además, en el *setup* asignamos los *pins* de la placa donde se conectarán las electroválvulas, siendo evco la válvula de entrada de aire al músculo (compresión), y evrel la válvula de salida (relajación).

```

void setup() {
  // put your setup code here, to run once:
  pinMode(evco, OUTPUT);
  pinMode(evrel, OUTPUT);

  //calibracion:
  Serial.println("Lleve al brazo a la posición de compresión");
  Serial.println("Pulse una tecla cuando lo haya colocado");
  Serial.read();
  while(Serial.available() <= 0);
  Serial.read();
  calibC = analogRead(A0);

  Serial.println("Lleve al brazo a la posición de extensión");
  Serial.println("Pulse una tecla cuando lo haya colocado");
  Serial.read();
  while(Serial.available() <= 0);
  Serial.read();
  calibE = analogRead(A0);

  Serial.println("Lleve al brazo a la posición media");
  Serial.println("Pulse una tecla cuando lo haya colocado");
  Serial.read();
  while(Serial.available() <= 0);
  Serial.read();
  calibM = analogRead(A0);

  Serial.println("Iniciando");
  Serial.print("Voltaje contraído");
  Serial.println(calibC);
  Serial.print("Voltaje medio");
  Serial.println(calibM);
}

```

*Figura 6.2. Código de calibración manual del sensor*

Con esto, podemos definir la función Steinhart-Hart (Figura 6.3.) que describe la relación entre elongación del elastómero y el cambio de voltaje que genera este. Con ello, haremos el cálculo de la función con las 3 posiciones que hemos definido previamente, y con el valor en voltios del sensor en cada una de dichas posiciones.

```

L1 = log(calibC);
L2 = log(calibM);
L3 = log(calibE);

Y1 = 1/RC;
Y2 = 1/RM;
Y3 = 1/RE;

Serial.print("L1: ");
Serial.println(L1, 6);
Serial.print("L2: ");
Serial.println(L2, 6);
Serial.print("L3: ");
Serial.println(L3, 6);

GAN2 = (Y2 - Y1)/(L2 - L1);
GAN3 = (Y3 - Y1)/(L3 - L1);

Serial.print("G2: ");
Serial.println(GAN2, 6);
Serial.print("G3: ");
Serial.println(GAN3, 6);

C = ((GAN3 - GAN2)/(L2 - L1))/(L1+L2+L3);
B = GAN2 - C*(pow(L1,2) + L1*L2 + pow(L2,2));
A = Y1 - (B + pow(L1,2)*C)*L1;

t = 1/(A + B*log(calibM) + C*pow(log(calibM),3));
diff = RM - (1/t);
}

```

Figura 6.3. Cálculo del modelo de Steinhart-Hart

Con todo lo realizado en la función *setup*, estamos preparados para entrar al ciclo de funcionamiento del *void loop*. En este ciclo, definiremos el control “todo o nada” que hemos comentado anteriormente. Lo primero es definir la variable con la que tomamos la medida del sensor por cada ciclo del código. Con esto podremos calcular la posición del codo, según la medida del sensor, gracias a la función de Steinhart-Hart, además de que calcularemos el ángulo del codo para facilitar la lectura de la posición, cosa que explicamos en el apartado 4.6.

```

t = 1/(A + B*log(calibM) + C*pow(log(calibM),3));
diff = RM - (1/t);
}

void loop() {
// put your main code here, to run repeatedly:
lim = analogRead(A0);

T = 1/(A + B*log(lim) + C*pow(log(lim),3));

grados = (acos((pow(x, 2) + pow(y, 2) - pow(T, 2))/(2*x*y)))*(180/pi);

```

Figura 6.4. Lectura del sensor y cálculo de la posición

Tras esto, entramos en el control del músculo, para lo que definiremos el punto al que queremos que se mueva el brazo o consigna (lo introducimos por teclado), y en torno a él un intervalo en el que la posición será correcta, ya que el brazo difícilmente estará en la posición exacta. Este intervalo lo hemos ajustado durante las pruebas realizadas para ajustar la oscilación del brazo.

```
//pedir consigna
Serial.println("Introduzca el valor de consigna: ");

if(Serial.available()>0){
  consigna = Serial.parseFloat();
  Serial.print("La consigna es: ");
  Serial.println(consigna);
}

comax = consigna + pres;
comin = consigna - pres;
```

*Figura 6.5. Solicitud de consigna y asignación del intervalo*

Con esto haremos 3 funciones *if*: uno para cuando el sensor detecte que está por debajo del valor mínimo del intervalo de consigna (para lo que se activa la válvula de compresión y desactiva la de relajación) (Figura 6.6.1.), otro para cuando se encuentre por encima del valor máximo del intervalo (para lo que se activa la válvula de relajación y desactiva la de compresión) (Figura 6.6.2.), y un último para cuando esté dentro del intervalo (para lo que se desactivan ambas válvulas) (Figura 6.6.3.).

```
//ascendente
if(lim < comin){
  digitalWrite(evco, HIGH);
  digitalWrite(evrel, LOW);

  Serial.println("asc");
  Serial.print("consigna: ");
  Serial.println(consigna);
  Serial.print("posición: ");
  Serial.println(lim);
  Serial.print("posición (cm): ");
  Serial.println(T);
  Serial.print("grados (°): ");
  Serial.println(grados);
}
```

*Figura 6.6.1. Por debajo del intervalo de consigna*

```

//descendente
if(lim > comax){
    digitalWrite(evco, LOW);
    digitalWrite(evrel, HIGH);

    Serial.println("desc");
    Serial.print("consigna: ");
    Serial.println(consigna);
    Serial.print("voltaje en bits: ");
    Serial.println(lim);
    Serial.print("posición (cm): ");
    Serial.println(T);
    Serial.print("grados (°): ");
    Serial.println(grados);
}

```

*Figura 6.6.2. Por encima del intervalo de consigna*

```

//en el intervalo
if((comin <= lim)&&(lim <= comax)){
    digitalWrite(evco, LOW);
    digitalWrite(evrel, LOW);

    Serial.println("en el intervalo");
    Serial.print("consigna: ");
    Serial.println(consigna);
    Serial.print("posición: ");
    Serial.println(lim);
    Serial.print("posición (cm): ");
    Serial.println(T);
    Serial.print("grados (°): ");
    Serial.println(grados);
}

```

*Figura 6.6.3. Dentro del intervalo de consigna*

## 7. Conclusión

En conclusión, la neumática es una herramienta muy útil, pero llevar a cabo un control preciso de esta es bastante complejo. Esto se debe a la naturaleza de los fluidos gaseosos, los cuales son compresibles y generan en los actuadores ciertas oscilaciones, siendo al final la causa de ser más lentos e imprecisos que otros tipos. El uso de actuadores neumáticos para la robótica y las prótesis no presenta grandes ventajas frente a los clásicos actuadores eléctricos, más precisos y rápidos en la respuesta.

Por otro lado, la resistencia elástica es un tipo de sensor poco regular y, posiblemente, esto sea un punto de mejora en el futuro desarrollo de esta prótesis. Asimismo, existe la posibilidad de que pueda llegar a utilizarse con otro tipo de calibración y adecuación de la señal.

También queríamos añadir que las electroválvulas son bastante propensas a averiarse, debido a que, al ajustar el brazo a la posición de consigna, se producen múltiples activaciones y desactivaciones. Creemos que esto es provocado por la oscilación producida por el aire comprimido y el control “todo o nada”.



Para finalizar, nos gustaría que este proyecto pueda desarrollarse mucho más, realizando una estructura más compleja e implementando todos los músculos necesarios para imitar un brazo humano real. Esto llevaría al control simultáneo de los músculos, además de la posibilidad de trabajar con un sensor electromiográfico para que el usuario lo pueda manejar como un brazo normal. Además de esto, se debería comprobar la autonomía del brazo con el uso de cartuchos de CO<sub>2</sub>, ya que no se podría utilizar el brazo conectado a un compresor de aire en todas las situaciones.

## 8. Bibliografía

[1] Motherboard. *The Mind-Controlled Bionic Arm With a Sense of Touch*. (18 de agosto de 2016). [Vídeo en línea]. Disponible en:  
[https://www.youtube.com/watch?v=F\\_brnKz\\_2tl&ab\\_channel=Motherboard](https://www.youtube.com/watch?v=F_brnKz_2tl&ab_channel=Motherboard)

[2] Technology Vision. *Fabrican un músculo artificial que puede revolucionar la robótica suave*. (12 de agosto de 2016). [Vídeo en línea]. Disponible en:  
[https://www.youtube.com/watch?v=6cDNZcucvIw&ab\\_channel=TechnologyVision](https://www.youtube.com/watch?v=6cDNZcucvIw&ab_channel=TechnologyVision)

[3] V. Lobato Ríos, "Modelo Flexible de Movimiento de Torso, Brazo, Antebrazo y Muñeca", Tesis doctoral, Instituto Nacional de Astrofísica, Óptica y Electrónica, Puebla, México, 2016. [En línea]. Disponible en:  
<https://inaoe.repositorioinstitucional.mx/jspui/bitstream/1009/834/1/LobatoRV.pdf>

[4] Fisioterapia en Movimiento. *ANATOMÍA Y MÚSCULOS DE BRAZO Y ANTEBRAZO*. (15 de julio de 2017). [Vídeo en línea]. Disponible en:  
[https://www.youtube.com/watch?v=3\\_Y0iGG4Cfw&ab\\_channel=FisioterapiaenMovimiento](https://www.youtube.com/watch?v=3_Y0iGG4Cfw&ab_channel=FisioterapiaenMovimiento)

[5] "Servomotor". Wikipedia. <https://es.wikipedia.org/wiki/Servomotor>

[6] L. Llamas. "TEORÍA DE CONTROL EN ARDUINO: CONTROL TODO O NADA CON HISTÉRESIS". Luisllamas.  
<https://www.luisllamas.es/control-todo-o-nada-con-histeresis-en-arduino/>

[7] "Teorema del coseno". Wikipedia. [https://es.wikipedia.org/wiki/Teorema\\_del\\_coseno](https://es.wikipedia.org/wiki/Teorema_del_coseno)

[8] "Compresor (máquina)". Wikipedia.  
[https://es.wikipedia.org/wiki/Compresor\\_\(m%C3%A1quina\)](https://es.wikipedia.org/wiki/Compresor_(m%C3%A1quina))

[9] "Steinhart–Hart equation". Wikipedia.  
[https://en.wikipedia.org/wiki/Steinhart%E2%80%93Hart\\_equation](https://en.wikipedia.org/wiki/Steinhart%E2%80%93Hart_equation)

[10] H. Hervin y S. Kirchner. "Arduino and Thermistors – The Secret to Accurate Room Temperature". Ametherm.  
<https://www.ametherm.com/blog/thermistor/arduino-and-thermistors>

- [11] J. Hrisko. "Arduino Thermistor Theory, Calibration, and Experiment". Makersportal. <https://makersportal.com/blog/2019/1/15/arduino-thermistor-theory-calibration-and-experiment>
- [12] "Elastómeros dieléctricos". Wikipedia. [https://es.wikipedia.org/wiki/Elast%C3%B3meros\\_diel%C3%A9ctricos](https://es.wikipedia.org/wiki/Elast%C3%B3meros_diel%C3%A9ctricos)
- [13] "Actuadores Neumáticos". Chemik. <https://www.chemik.es/actuadores-neumaticos/>
- [14] J.C. Díaz Montes y J.M. Dorador González, "Cánones por uso de la Infraestructura Ferroviaria Española", presentada en XV CONGRESO INTERNACIONAL ANUAL DE LA SOMIM, OBREGÓN, SONORA. MÉXICO, 23-25 sep., 2009. [En línea]. Disponible en: [http://somim.org.mx/memorias/memorias2009/pdfs/A1/A1\\_216.pdf](http://somim.org.mx/memorias/memorias2009/pdfs/A1/A1_216.pdf)
- [15] Electromecanic. "Definición y tipos de actuadores hidráulicos". Automantenimiento. <https://automantenimiento.net/hidraulica/definicion-y-tipos-de-actuadores-hidraulicos/>
- [16] "Neumática". Wikipedia. <https://es.wikipedia.org/wiki/Neum%C3%A1tica>
- [17] "Manómetro". Wikipedia. <https://es.wikipedia.org/wiki/Man%C3%B3metro>
- [18] "Puente de Wheatstone". Wikipedia. [https://es.wikipedia.org/wiki/Puente\\_de\\_Wheatstone](https://es.wikipedia.org/wiki/Puente_de_Wheatstone)
- [19] F. Roussillon. "Conoce tu Sistema Neumático: Detección de Posición Continua". Parker. <http://blog.parker.com/mx/conoce-tu-sistema-neum%C3%A1tico%3A-detecci%C3%B3n-de-posici%C3%B3n-continua>
- [20] M.J. Escalera Tornero y A.J. Rodríguez. "Actuadores Neumáticos". uhu. Fernández <http://www.uhu.es/rafael.sanchez/ingenieriamaquinas/carpetaapuntes.htm/Trabajos%20IM%202009-10/Manuel%20Jesus%20Escalera-Antonio%20Rodriguez-Actuadores%20Neumaticos.pdf>
- [21] "Ventajas y desventajas de los actuadores". M.parrict. <http://m.parrict.com/info/advantages-and-disadvantages-of-actuators-43037225.html>
- [22] BioMakers Industries. *¿MUSCULO ARTIFICIAL? Real Iron Man 2019*. (10 de octubre de 2019). [Vídeo en línea]. Disponible en: [https://www.youtube.com/watch?v=ISuBA--x1g4&list=PLRwhDo1gHiOD5nBXIONutXSpO2EdcZ2dN&index=9&ab\\_channel=BioMakersIndustries](https://www.youtube.com/watch?v=ISuBA--x1g4&list=PLRwhDo1gHiOD5nBXIONutXSpO2EdcZ2dN&index=9&ab_channel=BioMakersIndustries)
- [23] Hacksmith Industries. *BIONIC ARM = MAXIMUM STRENGTH (Crysis Nanosuit IRL)*. (2 de julio de 2020). [Vídeo en línea]. Disponible en: [https://www.youtube.com/watch?v=L0esNg2ys\\_s&list=PLRwhDo1gHiOD5nBXIONutXSpO2EdcZ2dN&index=2&t=330s&ab\\_channel=HacksmithIndustries](https://www.youtube.com/watch?v=L0esNg2ys_s&list=PLRwhDo1gHiOD5nBXIONutXSpO2EdcZ2dN&index=2&t=330s&ab_channel=HacksmithIndustries)

## **9. Anexos**

### **9.1. Lectura de datos del sensor**

### **9.2. Diseño de circuito para la adecuación de la señal en LTspice**

### **9.3. Código de Octave para el cálculo y graficación del modelo de Steinhart-Hart**

### **9.4. Código de Arduino**

### **9.5. Datasheets**

#### **9.5.1. Transistor**

#### **9.5.2. Amplificador operacional**

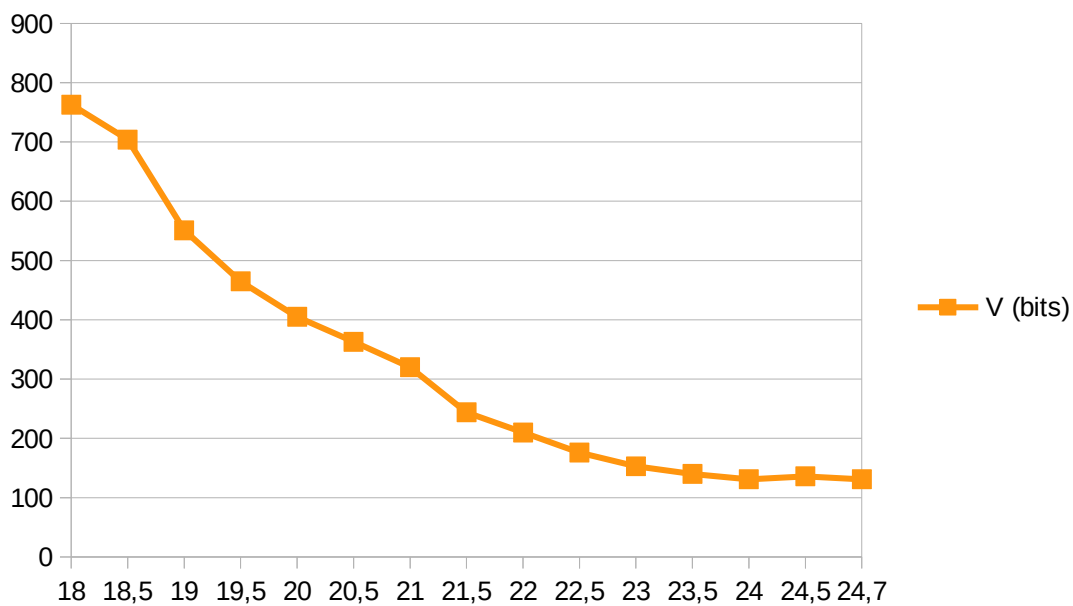
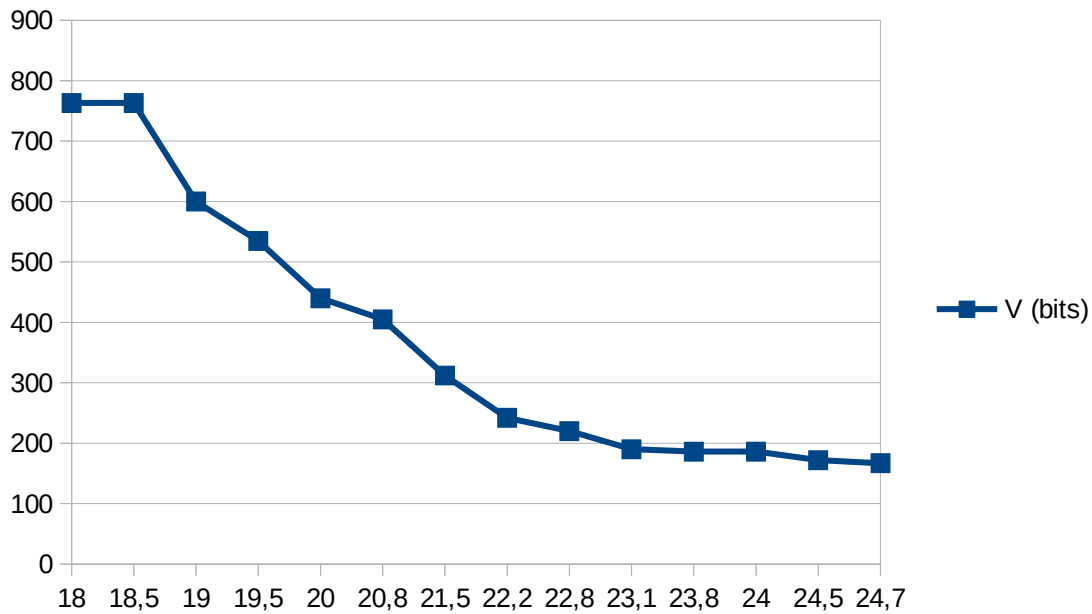
#### **9.5.3. Arduino**

Hoja1

L (cm)	Codo (°)	V (bits)
18	43,31	763
18,5	51,87	763
19	59,72	600
19,5	67,17	535
20	74,42	440
20,8	85,90	405
21,5	96,16	312
22,2	106,98	242
22,8	117,07	220
23,1	122,56	190
23,8	137,43	186
24	142,57	186
24,5	160,15	172
24,7	180,00	167

20/05/21

L (cm)	Codo (°)	V (bits)
18	43,31	763
18,5	51,87	704
19	59,72	551
19,5	67,17	465
20	74,42	405
20,5	81,58	363
21	88,79	320
21,5	96,16	244
22	103,80	210
22,5	111,90	176
23	120,69	153
23,5	130,60	140
24	142,57	131
24,5	160,15	136
24,7	180,00	131

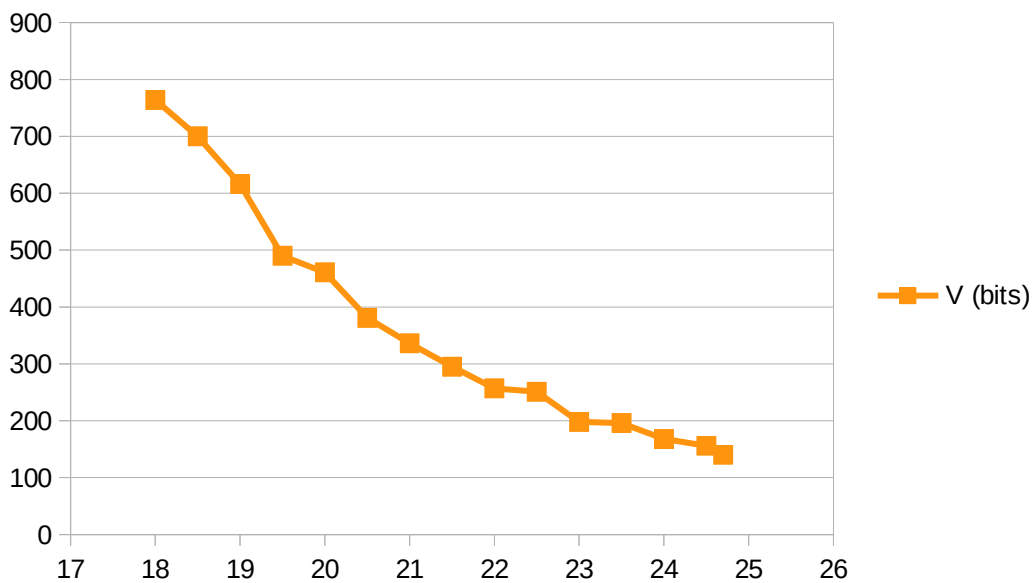
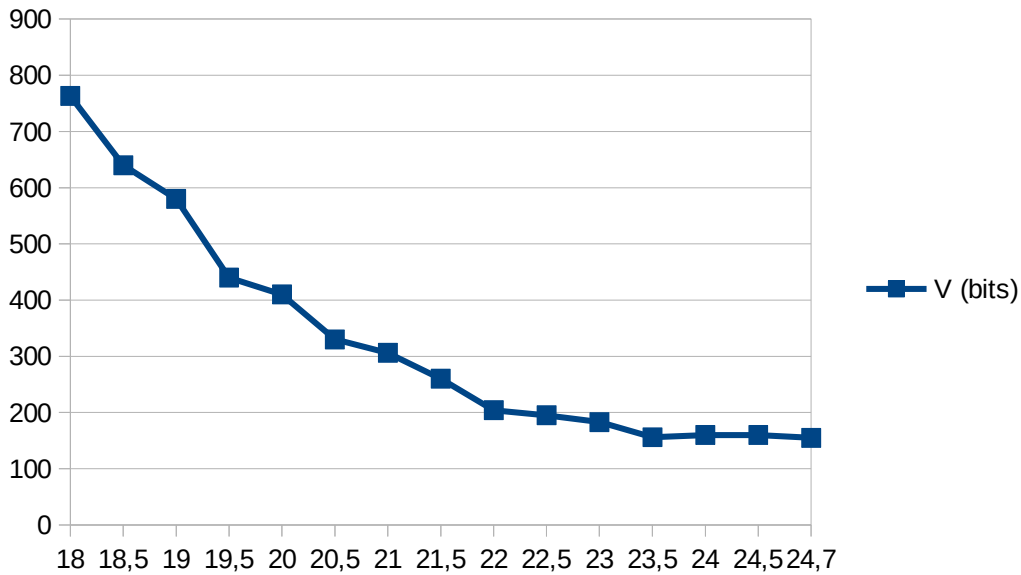


Hoja1

21/05/21  
19:15:00  
21° aprox

L (cm)	Codo (°)	V (bits)
18	43,31	763
18,5	51,87	640
19	59,72	580
19,5	67,17	440
20	74,42	410
20,5	81,58	330
21	88,79	306
21,5	96,16	260
22	103,80	204
22,5	111,90	195
23	120,69	183
23,5	130,60	156
24	142,57	160
24,5	160,15	160
24,7	180,00	155

L (cm)	Codo (°)	V (bits)
18	43,31	764
18,5	51,87	700
19	59,72	616
19,5	67,17	490
20	74,42	461
20,5	81,58	381
21	88,79	336
21,5	96,16	295
22	103,80	257
22,5	111,90	251
23	120,69	198
23,5	130,60	196
24	142,57	168
24,5	160,15	156
24,7	180,00	140

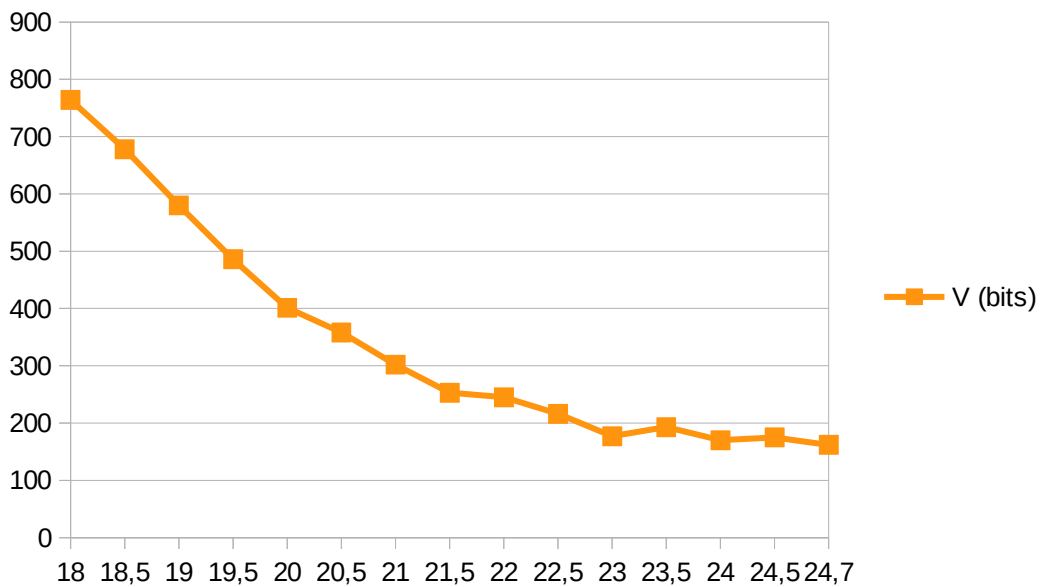
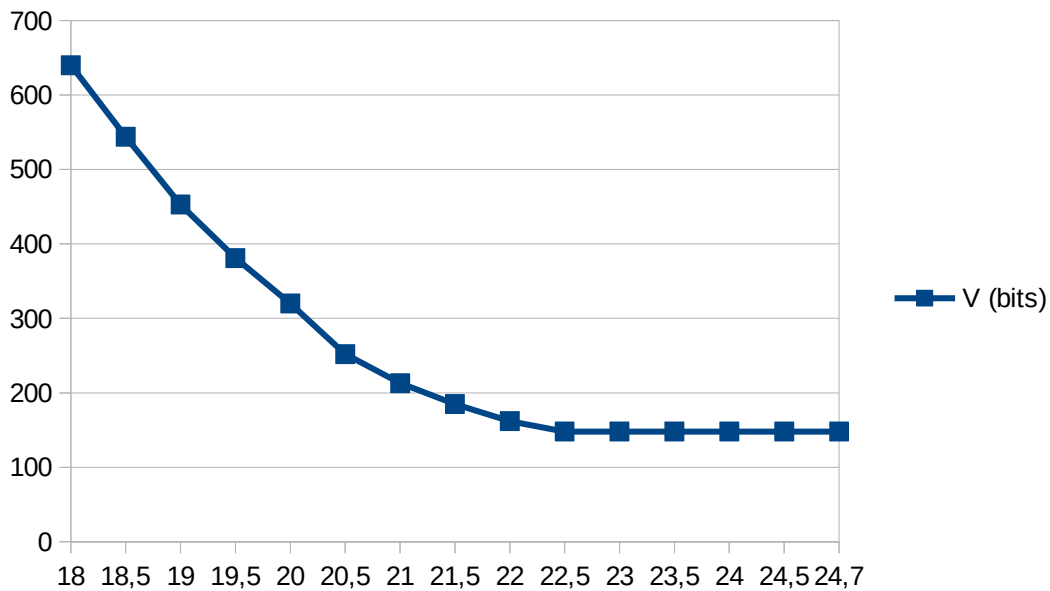


Hoja1

22/05/21  
17:25:00  
22°

L (cm)	Codo (°)	V (bits)
18	43,31	640
18,5	51,87	544
19	59,72	453
19,5	67,17	381
20	74,42	320
20,5	81,58	252
21	88,79	213
21,5	96,16	185
22	103,80	162
22,5	111,90	148
23	120,69	148
23,5	130,60	148
24	142,57	148
24,5	160,15	148
24,7	180,00	148

L (cm)	Codo (°)	V (bits)
18	43,31	764
18,5	51,87	678
19	59,72	580
19,5	67,17	486
20	74,42	401
20,5	81,58	358
21	88,79	302
21,5	96,16	253
22	103,80	245
22,5	111,90	216
23	120,69	177
23,5	130,60	193
24	142,57	170
24,5	160,15	175
24,7	180,00	162

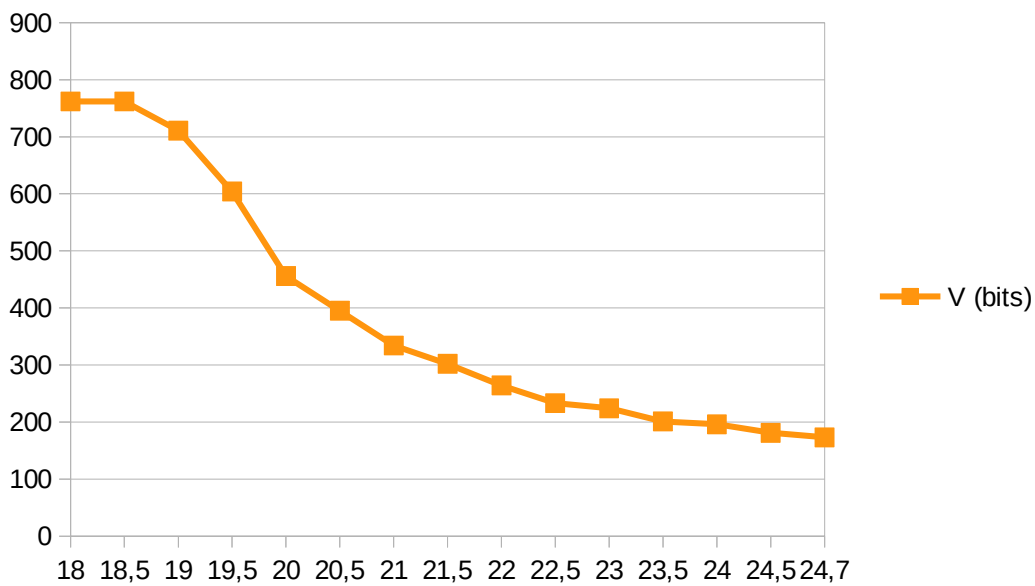
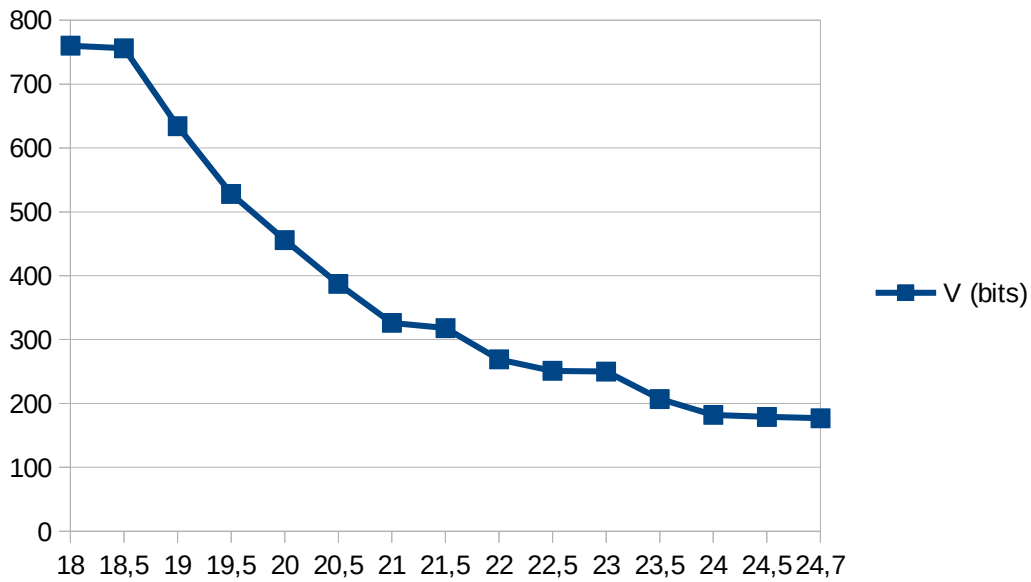


Hoja1

24/05/21  
20:40:00  
23°

L (cm)	Codo (°)	V (bits)
18	43,31	760
18,5	51,87	756
19	59,72	634
19,5	67,17	528
20	74,42	456
20,5	81,58	387
21	88,79	326
21,5	96,16	318
22	103,80	269
22,5	111,90	251
23	120,69	250
23,5	130,60	207
24	142,57	182
24,5	160,15	179
24,7	180,00	177

L (cm)	Codo (°)	V (bits)
18	43,31	762
18,5	51,87	762
19	59,72	711
19,5	67,17	604
20	74,42	456
20,5	81,58	395
21	88,79	334
21,5	96,16	302
22	103,80	264
22,5	111,90	233
23	120,69	224
23,5	130,60	201
24	142,57	196
24,5	160,15	181
24,7	180,00	173

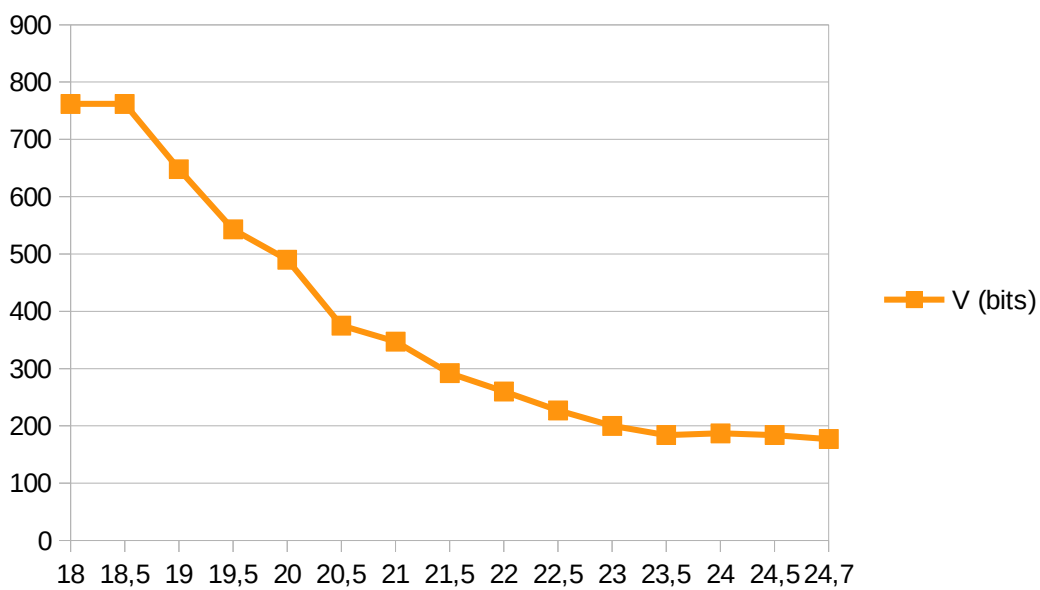
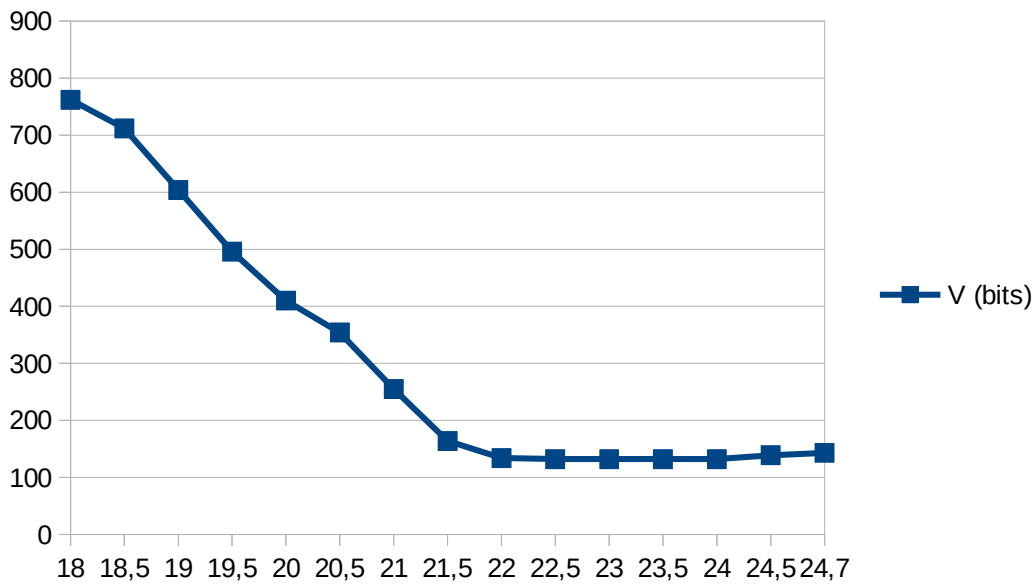


Hoja1

25/05/21  
21:15:00  
22°

L (cm)	Codo (°)	V (bits)
18	43,31	762
18,5	51,87	712
19	59,72	604
19,5	67,17	496
20	74,42	410
20,5	81,58	354
21	88,79	255
21,5	96,16	164
22	103,80	134
22,5	111,90	132
23	120,69	132
23,5	130,60	132
24	142,57	132
24,5	160,15	139
24,7	180,00	143

L (cm)	Codo (°)	V (bits)
18	43,31	762
18,5	51,87	762
19	59,72	648
19,5	67,17	543
20	74,42	490
20,5	81,58	375
21	88,79	347
21,5	96,16	292
22	103,80	260
22,5	111,90	227
23	120,69	200
23,5	130,60	184
24	142,57	187
24,5	160,15	184
24,7	180,00	177



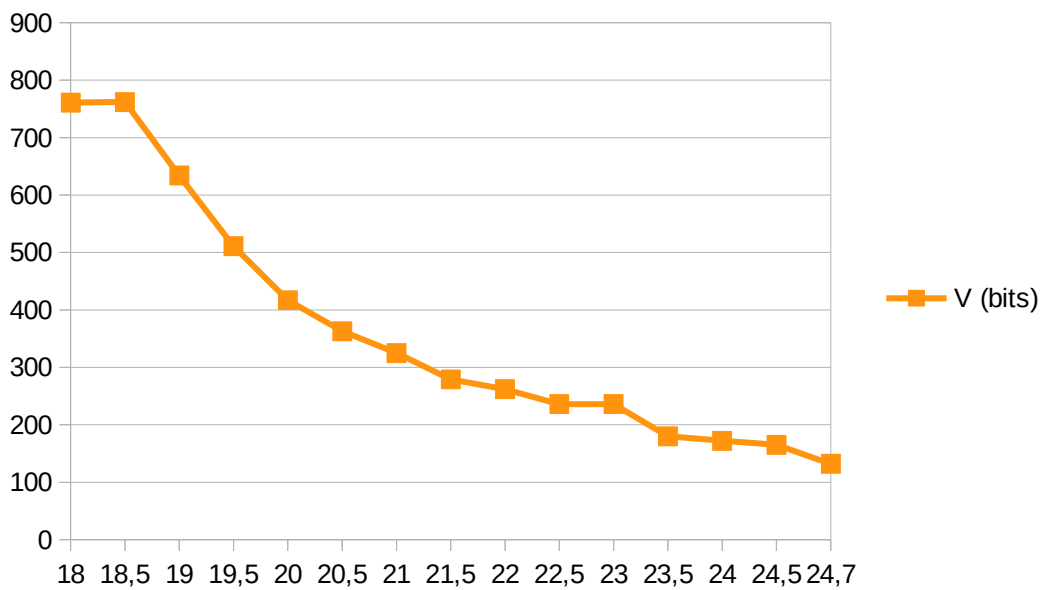
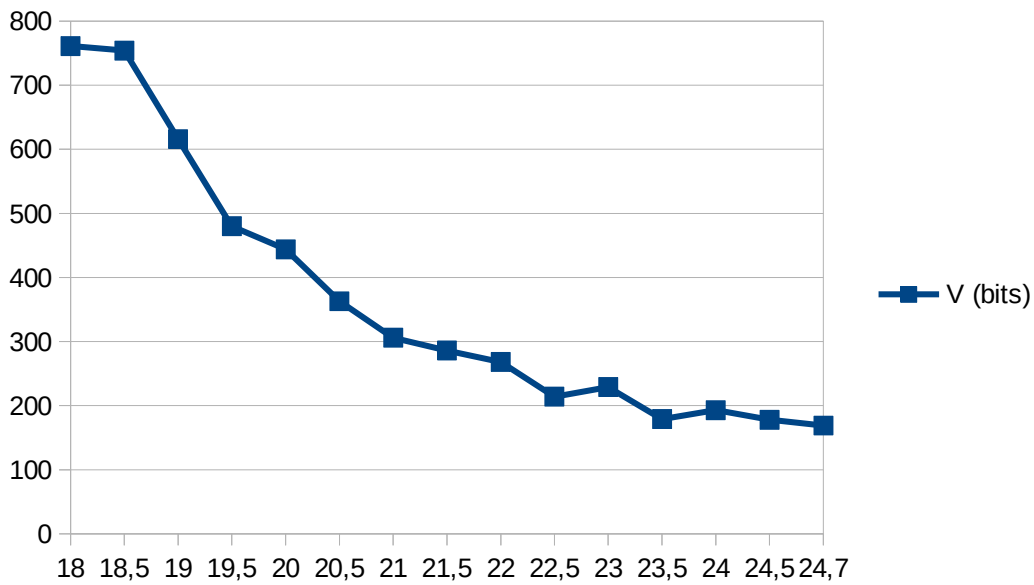


Hoja1

26/05/21  
21:20:00  
22°

L (cm)	Codo (°)	V (bits)
18	43,31	761
18,5	51,87	754
19	59,72	616
19,5	67,17	480
20	74,42	444
20,5	81,58	363
21	88,79	306
21,5	96,16	286
22	103,80	268
22,5	111,90	214
23	120,69	229
23,5	130,60	179
24	142,57	193
24,5	160,15	178
24,7	180,00	169

L (cm)	Codo (°)	V (bits)
18	43,31	761
18,5	51,87	762
19	59,72	634
19,5	67,17	511
20	74,42	417
20,5	81,58	363
21	88,79	325
21,5	96,16	279
22	103,80	262
22,5	111,90	236
23	120,69	236
23,5	130,60	180
24	142,57	172
24,5	160,15	165
24,7	180,00	132

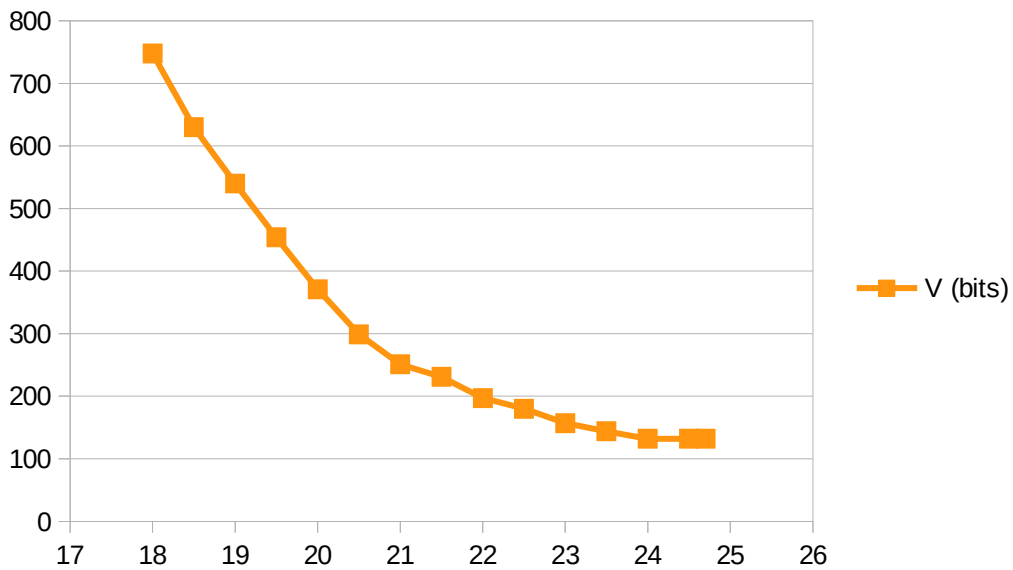
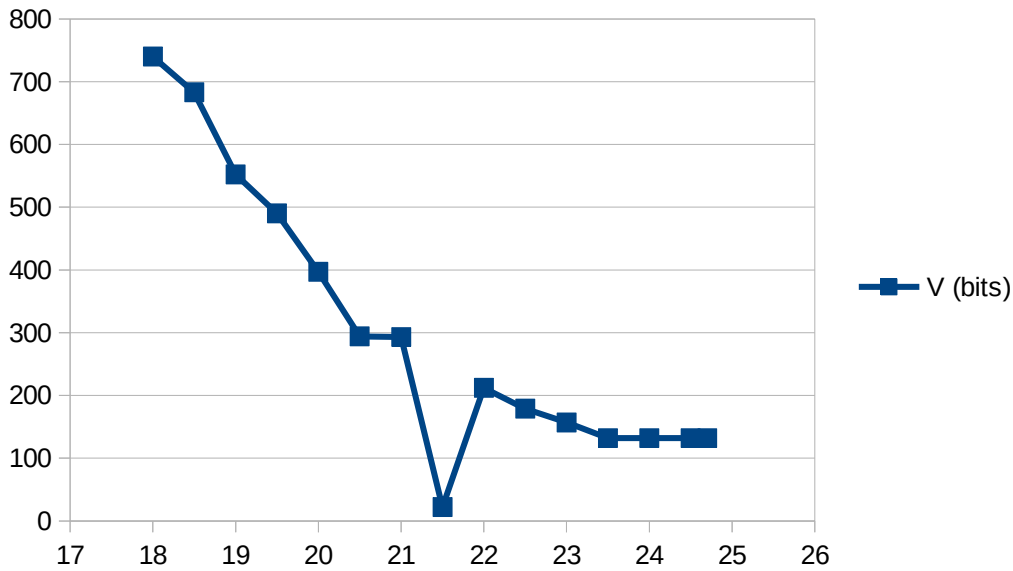


Hoja1

28/05/21  
15:00:00  
22°

L (cm)	Codo (°)	V (bits)
18	43,31	740
18,5	51,87	683
19	59,72	552
19,5	67,17	490
20	74,42	397
20,5	81,58	294
21	88,79	293
21,5	96,16	22
22	103,80	212
22,5	111,90	179
23	120,69	157
23,5	130,60	132
24	142,57	132
24,5	160,15	132
24,7	180,00	132

L (cm)	Codo (°)	V (bits)
18	43,31	748
18,5	51,87	630
19	59,72	540
19,5	67,17	454
20	74,42	371
20,5	81,58	299
21	88,79	251
21,5	96,16	231
22	103,80	197
22,5	111,90	180
23	120,69	157
23,5	130,60	144
24	142,57	132
24,5	160,15	132
24,7	180,00	132

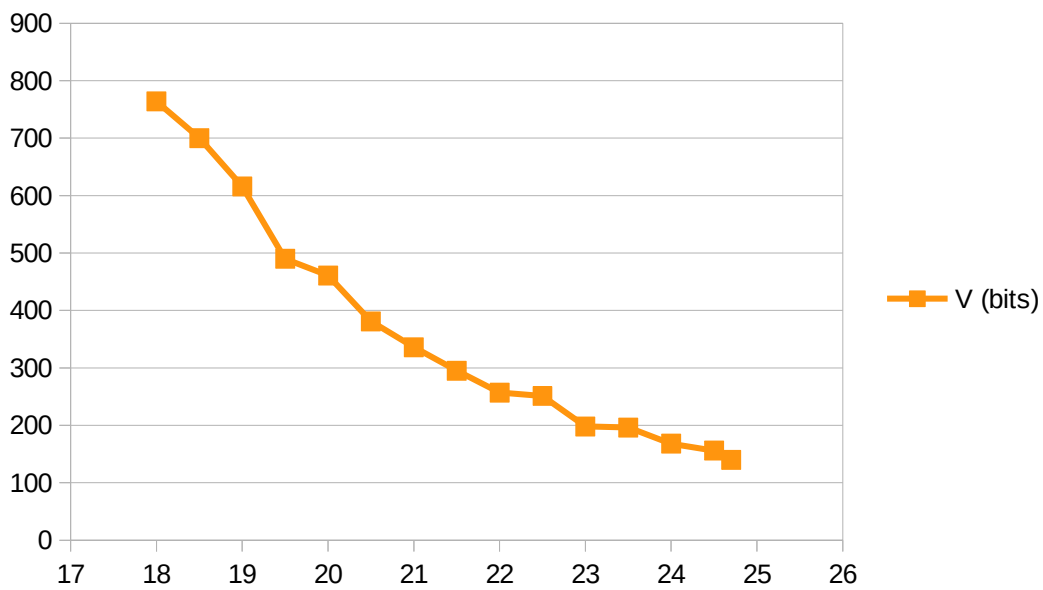
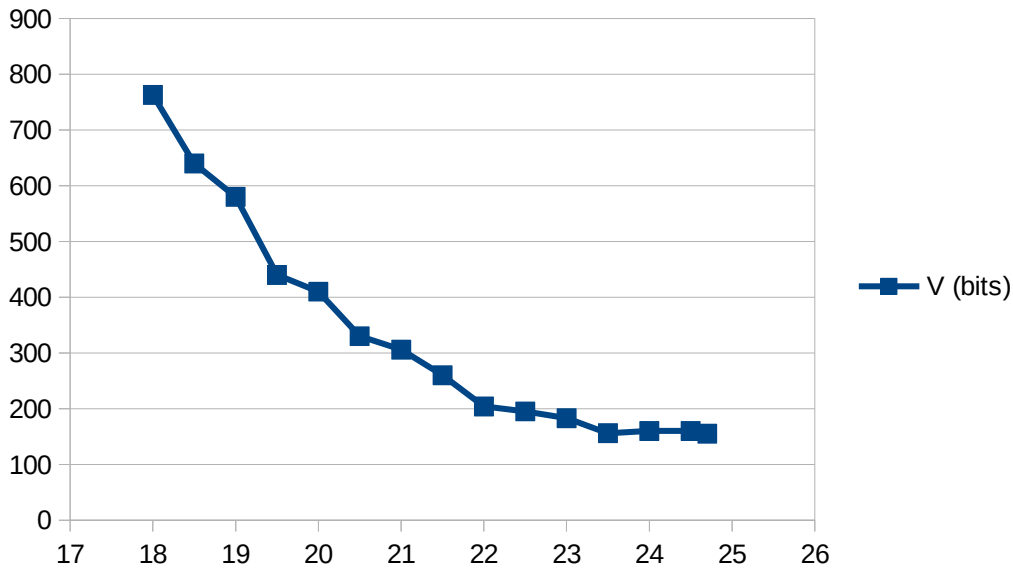


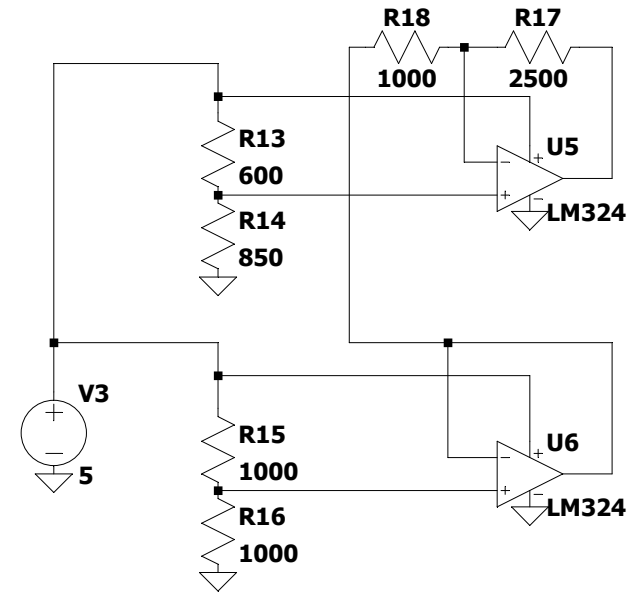
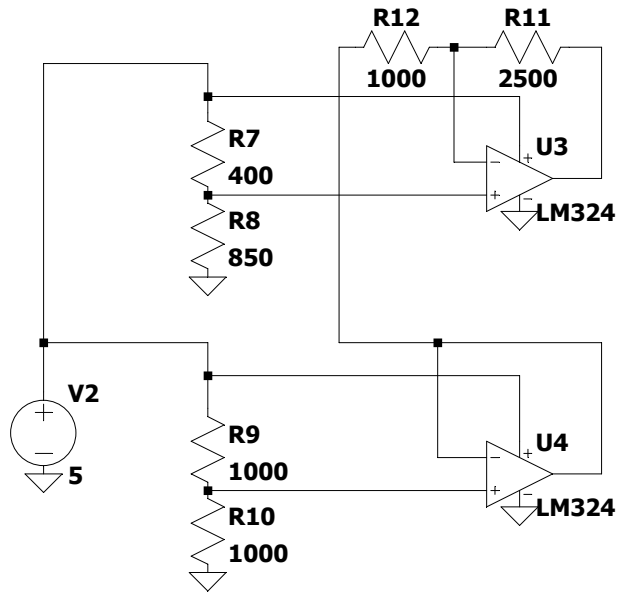
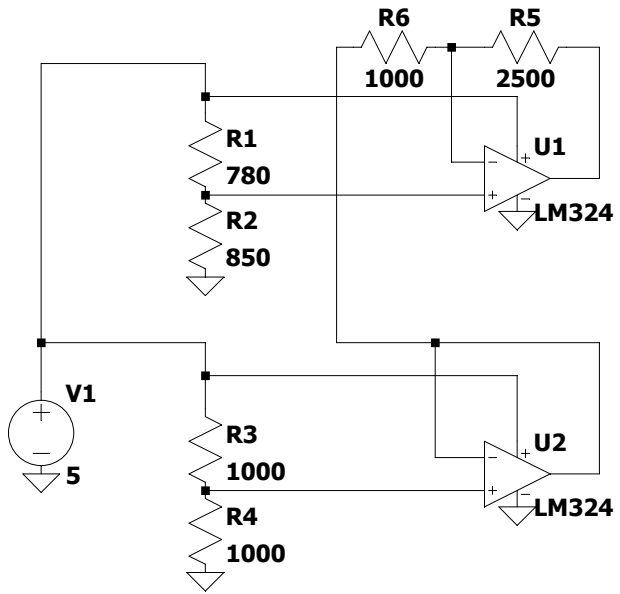
Hoja1

10/08/21  
14:00:00  
21° aprox

L (cm)	Codo (°)	V (bits)
18	43,31	763
18,5	51,87	640
19	59,72	580
19,5	67,17	440
20	74,42	410
20,5	81,58	330
21	88,79	306
21,5	96,16	260
22	103,80	204
22,5	111,90	195
23	120,69	183
23,5	130,60	156
24	142,57	160
24,5	160,15	160
24,7	180,00	155

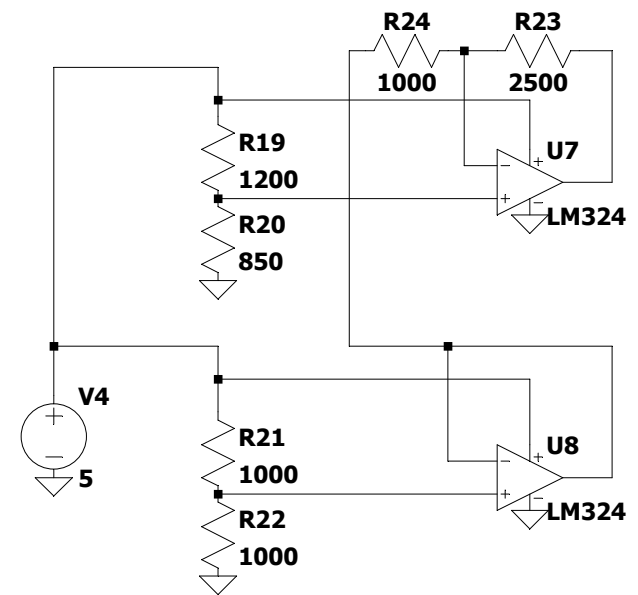
L (cm)	Codo (°)	V (bits)
18	43,31	764
18,5	51,87	700
19	59,72	616
19,5	67,17	490
20	74,42	461
20,5	81,58	381
21	88,79	336
21,5	96,16	295
22	103,80	257
22,5	111,90	251
23	120,69	198
23,5	130,60	196
24	142,57	168
24,5	160,15	156
24,7	180,00	140





.tran 10m

.include LM324.txt



```
1 close all
2 clear
3 pkg load optim
4
5 T2 = [18
6 18.5
7 19
8 19.5
9 20
10 20.5
11 21
12 21.5
13 22
14 22.5
15 23
16 23.5
17 24
18 24.5
19 24.7
20 ];
21
22 R2 = [763
23 704
24 551
25 465
26 405
27 363
28 320
29 244
30 210
31 176
32 153
33 140
34 131
35 136
36 131
37 ];
38
39 R3 = [763
40 640
41 580
42 440
43 410
44 330
45 306
46 260
47 204
48 195
49 183
50 156
51 160
52 160
53 155
54 ];
55
56 R4 = [764
57 700
58 616
59 490
60 461
61 381
62 336
63 295
64 257
65 251
66 198
67 196
68 168
69 156
70 140
71 ];
72
73 R5 = [640
74 544
75 453
```

```
76 381
77 320
78 252
79 213
80 185
81 162
82 148
83 148
84 148
85 148
86 148
87 148
88 ];
89
90 R6 = [764
91 678
92 580
93 486
94 401
95 358
96 302
97 253
98 245
99 216
100 177
101 193
102 170
103 175
104 162
105 ];
106
107 R7 = [760
108 756
109 634
110 528
111 456
112 387
113 326
114 318
115 269
116 251
117 250
118 207
119 182
120 179
121 177
122 ];
123
124 R8 = [762
125 762
126 711
127 604
128 456
129 395
130 334
131 302
132 264
133 233
134 224
135 201
136 196
137 181
138 173
139 ];
140
141 R9 = [762
142 712
143 604
144 496
145 410
146 354
147 255
148 164
149 134
150 132
```

```
151 132
152 132
153 132
154 139
155 143
156 ];
157
158 R10 = [762
159 762
160 648
161 543
162 490
163 375
164 347
165 292
166 260
167 227
168 200
169 184
170 187
171 184
172 177
173 ];
174
175 R11 = [761
176 754
177 616
178 480
179 444
180 363
181 306
182 286
183 268
184 214
185 229
186 179
187 193
188 178
189 169
190 ];
191
192 R12 = [761
193 762
194 634
195 511
196 417
197 363
198 325
199 279
200 262
201 236
202 236
203 180
204 172
205 165
206 132
207 ];
208
209 R13 = [740
210 683
211 552
212 490
213 397
214 294
215 293
216 228
217 212
218 179
219 157
220 132
221 132
222 132
223 132
224 ];
225
```

```

226 R14 = [748
227 630
228 540
229 454
230 371
231 299
232 251
233 231
234 197
235 180
236 157
237 144
238 132
239 132
240 132
241 ];
242
243 T = [T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2];
244 R = [R2 R3 R4 R5 R6 R7 R8 R9 R10 R11 R12 R13 R14];
245
246 for i=1:13,
247
248     ADV=i
249
250     L1=log(R(1,i))
251     L2=log(R(8,i))
252     L3=log(R(15,i))
253     Y1=1/T(1,i);
254     Y2=1/T(8,i);
255     Y3=1/T(15,i);
256
257     ganma2=(Y2-Y1)/(L2-L1)
258     ganma3=(Y3-Y1)/(L3-L1)
259
260     C=( (ganma3-ganma2)/(L2-L1))/(L1+L2+L3)
261
262
263     B=ganma2-C*((L1^2)+L1*L2+(L2^2))
264     A=Y1-(B+(L1^2)*C)*L1
265     t = A + B*log(R(8,i)) + C*(log(R(8,i)^3));
266     diff = T(8,i) - 1/t
267
268
269     for j=1:15,
270         t = A + B*log(R(j,i)) + C*(log(R(j,i)^3));
271         calT(j) = 1/t + diff;
272
273     end
274
275     figure(i)
276     plot(R(:,i),T(:,i),R(:,i),calT)
277     legend('Medido','Calculado');
278
279 end
280
281
282
283

```



```

/**Controlador todo o nada: músculo neumático**
**Sául Rguez Hdez**
*/

//librerias
#include <stdio.h>
#include <math.h>

//entradas y salidas analógicas
int posM = A0;
int lim = 0;

//entradas y salidas digitales
#define evco 2
#define evrel 3

//declaración de variables
float consigna;
float comax;
float comin;
float pres = 10;
float calibE;
float calibM;
float calibC;
float RM = 22;
float RE = 25.5;
float RC = 16.5;
float L1, L2, L3, Y1, Y2, Y3, GAN2, GAN3, A, B, C;
float T;
float t;
float diff;
float x = 5;
float y = 20.7;
float grados;
const float pi=3.1416;

void setup() {
  // put your setup code here, to run once:
  pinMode(evco, OUTPUT);
  pinMode(evrel, OUTPUT);

  Serial.begin(9600);

  //calibracion:
  Serial.println("Lleve al brazo a la posición de compresión");
  Serial.println("Pulse una tecla cuando lo haya colocado");
  Serial.read();
  while(Serial.available() <= 0);

```

```

Serial.read();
calibC = analogRead(A0);

Serial.println("Lleve al brazo a la posición de extensión");
Serial.println("Pulse una tecla cuando lo haya colocado");
Serial.read();
while(Serial.available() <= 0);
Serial.read();
calibE = analogRead(A0);

Serial.println("Lleve al brazo a la posición media");
Serial.println("Pulse una tecla cuando lo haya colocado");
Serial.read();
while(Serial.available() <= 0);
Serial.read();
calibM = analogRead(A0);

Serial.println("Iniciando");
Serial.print("Voltaje contraído");
Serial.println(calibC);
Serial.print("Voltaje medio");
Serial.println(calibM);
Serial.print("Voltaje extendido");
Serial.println(calibE);

L1 = log(calibC);
L2 = log(calibM);
L3 = log(calibE);

Y1 = 1/RC;
Y2 = 1/RM;
Y3 = 1/RE;

Serial.print("L1: ");
Serial.println(L1, 6);
Serial.print("L2: ");
Serial.println(L2, 6);
Serial.print("L3: ");
Serial.println(L3, 6);

GAN2 = (Y2 - Y1)/(L2 - L1);
GAN3 = (Y3 - Y1)/(L3 - L1);

Serial.print("G2: ");
Serial.println(GAN2, 6);
Serial.print("G3: ");
Serial.println(GAN3, 6);

C = ((GAN3 - GAN2)/(L2 - L1))/(L1+L2+L3);

```

```
B = GAN2 - C*(pow(L1,2) + L1*L2 + pow(L2,2));
```

```
A = Y1 - (B + pow(L1,2)*C)*L1;
```

```
Serial.print("A: ");
```

```
Serial.println(A, 6);
```

```
Serial.print("B: ");
```

```
Serial.println(B, 6);
```

```
Serial.print("C: ");
```

```
Serial.println(C, 6);
```

```
t = 1/(A + B*log(calibM) + C*pow(log(calibM),3));
```

```
diff = RM - (1/t);
```

```
}
```

```
void loop() {
```

```
  // put your main code here, to run repeatedly:
```

```
  lim = analogRead(A0);
```

```
  T = 1/(A + B*log(lim) + C*pow(log(lim),3));
```

```
  grados = (acos((pow(x, 2) + pow(y, 2) - pow(T, 2))/(2*x*y)))*(180/pi);
```

```
  //pedir consigna
```

```
  Serial.println("Introduzca el valor de consigna: ");
```

```
  if(Serial.available()>0){
```

```
    consigna = Serial.parseFloat();
```

```
    Serial.print("La consigna es: ");
```

```
    Serial.println(consigna);
```

```
  }
```

```
  comax = consigna + pres;
```

```
  comin = consigna - pres;
```

```
  //ascendente
```

```
  if(lim < comin){
```

```
    digitalWrite(evco, HIGH);
```

```
    digitalWrite(evrel, LOW);
```

```
    Serial.println("asc");
```

```
    Serial.print("consigna: ");
```

```
    Serial.println(consigna);
```

```
    Serial.print("posición: ");
```

```
    Serial.println(lim);
```

```
    Serial.print("posición (cm): ");
```

```
    Serial.println(T);
```

```
    Serial.print("grados (°): ");
```

```
    Serial.println(grados);
```

```
}

//descendente
if(lim > comax){
  digitalWrite(evco, LOW);
  digitalWrite(evrel, HIGH);

  Serial.println("desc");
  Serial.print("consigna: ");
  Serial.println(consigna);
  Serial.print("voltaje en bits: ");
  Serial.println(lim);
  Serial.print("posición (cm): ");
  Serial.println(T);
  Serial.print("grados (°): ");
  Serial.println(grados);
}

//en el intervalo
if((comin <= lim)&&(lim <= comax)){
  digitalWrite(evco, LOW);
  digitalWrite(evrel, LOW);

  Serial.println("en el intervalo");
  Serial.print("consigna: ");
  Serial.println(consigna);
  Serial.print("posición: ");
  Serial.println(lim);
  Serial.print("posición (cm): ");
  Serial.println(T);
  Serial.print("grados (°): ");
  Serial.println(grados);
}
}
```

## Complementary low voltage transistor

### Features

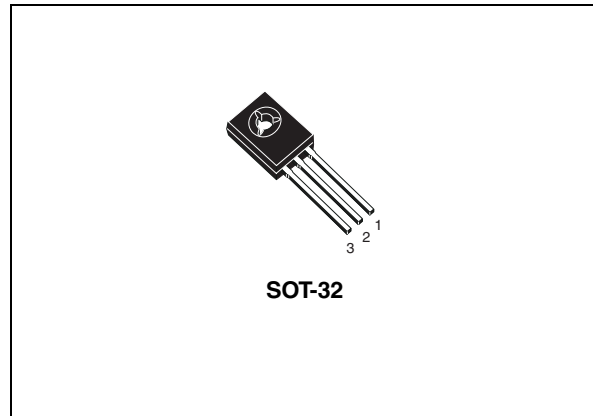
- Products are pre-selected in DC current gain

### Application

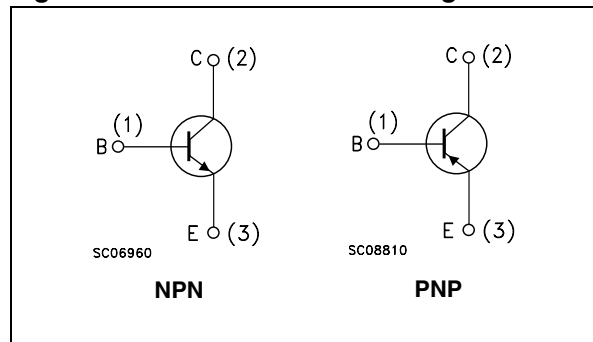
- General purpose

### Description

These epitaxial planar transistors are mounted in the SOT-32 plastic package. They are designed for audio amplifiers and drivers utilizing complementary or quasi-complementary circuits. The NPN types are the BD135 and BD139, and the complementary PNP types are the BD136 and BD140.



**Figure 1. Internal schematic diagram**



**Table 1. Device summary**

Order codes	Marking	Package	Packaging
BD135	BD135	SOT-32	Tube
BD135-16	BD135-16		
BD136	BD136		
BD136-16	BD136-16		
BD139	BD139		
BD139-10	BD139-10		
BD139-16	BD139-16		
BD140	BD140		
BD140-10	BD140-10		
BD140-16	BD140-16		

# Contents

<b>1</b>	<b>Electrical ratings</b> .....	<b>3</b>
<b>2</b>	<b>Electrical characteristics</b> .....	<b>4</b>
	2.1 Electrical characteristics (curves) .....	5
<b>3</b>	<b>Package mechanical data</b> .....	<b>6</b>
<b>4</b>	<b>Revision history</b> .....	<b>8</b>

# 1 Electrical ratings

**Table 2. Absolute maximum ratings**

Symbol	Parameter	Value				Unit
		NPN		PNP		
		BD135	BD139	BD136	BD140	
$V_{CBO}$	Collector-base voltage ( $I_E = 0$ )	45	80	-45	-80	V
$V_{CEO}$	Collector-emitter voltage ( $I_B = 0$ )	45	80	-45	-80	V
$V_{EBO}$	Emitter-base voltage ( $I_C = 0$ )	5		-5		V
$I_C$	Collector current	1.5		-1.5		A
$I_{CM}$	Collector peak current	3		-3		A
$I_B$	Base current	0.5		-0.5		A
$P_{TOT}$	Total dissipation at $T_c \leq 25\text{ °C}$	12.5				W
$P_{TOT}$	Total dissipation at $T_{amb} \leq 25\text{ °C}$	1.25				W
$T_{stg}$	Storage temperature	-65 to 150				°C
$T_j$	Max. operating junction temperature	150				°C

**Table 3. Thermal data**

Symbol	Parameter	Max value	Unit
$R_{thj-case}$	Thermal resistance junction-case	10	°C/W
$R_{thj-amb}$	Thermal resistance junction-ambient	100	°C/W

## 2 Electrical characteristics

( $T_{\text{case}} = 25\text{ °C}$  unless otherwise specified)

**Table 4. On/off states**

Symbol	Parameter	Polarity	Test conditions	Value			Unit
				Min.	Typ.	Max.	
$I_{\text{CBO}}$	Collector cut-off current ( $I_{\text{E}}=0$ )	NPN	$V_{\text{CB}} = 30\text{ V}$ $V_{\text{CB}} = 30\text{ V}, T_{\text{C}} = 125\text{ °C}$			0.1 10	$\mu\text{A}$ $\mu\text{A}$
		PNP	$V_{\text{CB}} = -30\text{ V}$ $V_{\text{CB}} = -30\text{ V}, T_{\text{C}} = 125\text{ °C}$			-0.1 -10	$\mu\text{A}$ $\mu\text{A}$
$I_{\text{EBO}}$	Emitter cut-off current ( $I_{\text{C}}=0$ )	NPN	$V_{\text{EB}} = 5\text{ V}$			10	$\mu\text{A}$
		PNP	$V_{\text{EB}} = -5\text{ V}$			-10	$\mu\text{A}$
$V_{\text{CEO(sus)}}^{(1)}$	Collector-emitter sustaining voltage ( $I_{\text{B}}=0$ )	NPN	$I_{\text{C}} = 30\text{ mA}$ BD135 BD139	45 80			V V
		PNP	$I_{\text{C}} = -30\text{ mA}$ BD136 BD140	-45 -80			V V
$V_{\text{CE(sat)}}^{(1)}$	Collector-emitter saturation voltage	NPN	$I_{\text{C}} = 0.5\text{ A}, I_{\text{B}} = 0.05\text{ A}$			0.5	V
		PNP	$I_{\text{C}} = -0.5\text{ A}, I_{\text{B}} = -0.05\text{ A}$			-0.5	V
$V_{\text{BE}}^{(1)}$	Base-emitter voltage	NPN	$I_{\text{C}} = 0.5\text{ A}, V_{\text{CE}} = 2\text{ V}$			1	V
		PNP	$I_{\text{C}} = -0.5\text{ A}, V_{\text{CE}} = -2\text{ V}$			-1	V
$h_{\text{FE}}^{(1)}$	DC current gain	NPN	$I_{\text{C}} = 5\text{ mA}, V_{\text{CE}} = 2\text{ V}$ $I_{\text{C}} = 150\text{ mA}, V_{\text{CE}} = 2\text{ V}$ $I_{\text{C}} = 0.5\text{ A}, V_{\text{CE}} = 2\text{ V}$	25 40 25		250	
		PNP	$I_{\text{C}} = -5\text{ mA}, V_{\text{CE}} = -2\text{ V}$ $I_{\text{C}} = -150\text{ mA}, V_{\text{CE}} = -2\text{ V}$ $I_{\text{C}} = -0.5\text{ A}, V_{\text{CE}} = -2\text{ V}$	25 40 25		250	
$h_{\text{FE}}^{(1)}$	$h_{\text{FE}}$ groups	NPN	$I_{\text{C}} = 150\text{ mA}, V_{\text{CE}} = 2\text{ V}$ BD139-10 BD135-16/BD139-16	63 100		160 250	
		PNP	$I_{\text{C}} = -150\text{ mA}, V_{\text{CE}} = -2\text{ V}$ BD140-10 BD136-16/BD140-16	63 100		160 250	

1. Pulsed: pulse duration = 300  $\mu\text{s}$ , duty cycle 1.5%



## 2.1 Electrical characteristics (curves)

Figure 2. Safe operating area

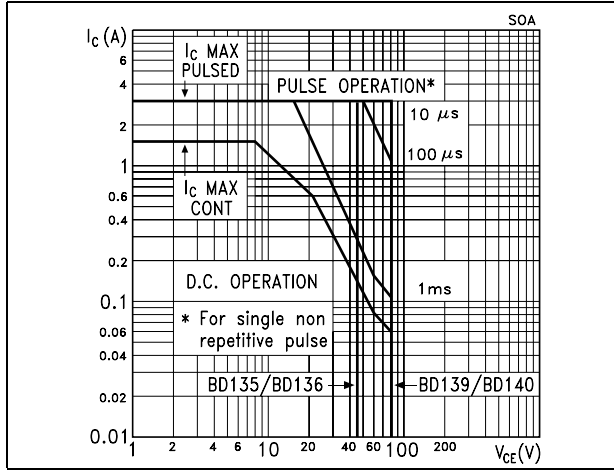
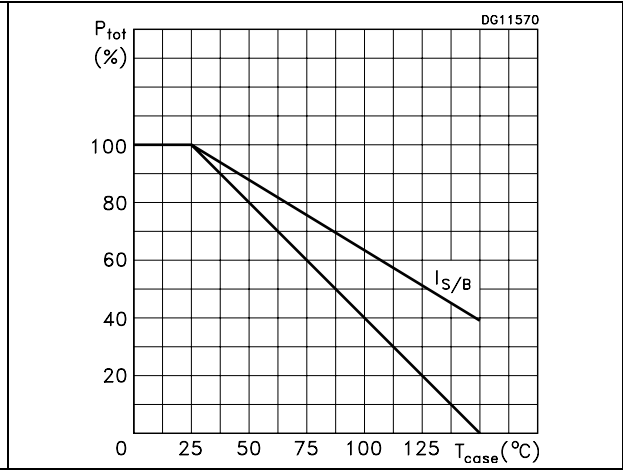


Figure 3. Derating

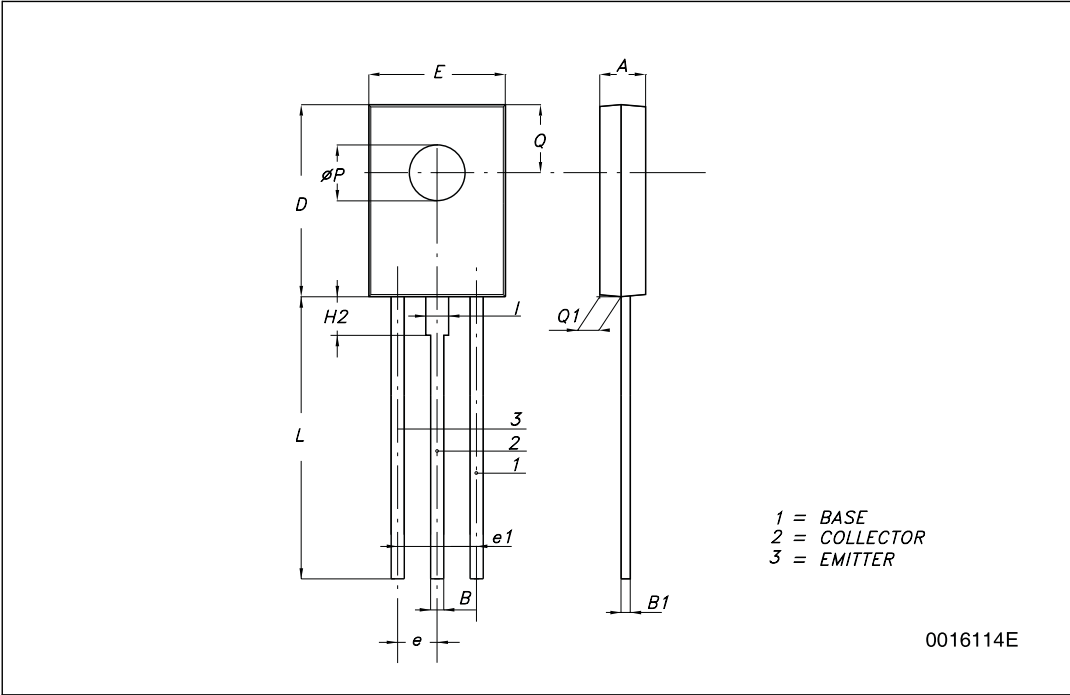


### 3 Package mechanical data

In order to meet environmental requirements, ST offers these devices in ECOPACK® packages. These packages have a lead-free second level interconnect. The category of second level interconnect is marked on the package and on the inner box label, in compliance with JEDEC Standard JESD97. The maximum ratings related to soldering conditions are also marked on the inner box label. ECOPACK is an ST trademark. ECOPACK specifications are available at: [www.st.com](http://www.st.com)

**SOT-32 (TO-126) MECHANICAL DATA**

DIM.	mm.		
	MIN.	TYP	MAX.
A	2.4		2.9
B	0.64		0.88
B1	0.39		0.63
D	10.5		11.05
E	7.4		7.8
e	2.04	2.29	2.54
e1	4.07	4.58	5.08
L	15.3		16
P	2.9		3.2
Q		3.8	
Q1	1		1.52
H2		2.15	
I		1.27	



## 4 Revision history

**Table 5. Document revision history**

Date	Revision	Changes
16-Sep-2001	4	
22-May-2008	5	Mechanical data has been updated.

**Please Read Carefully:**

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

**UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.**

**UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER'S OWN RISK.**

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2008 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

[www.st.com](http://www.st.com)



# Dual Low Power Operational Amplifiers

Utilizing the circuit designs perfected for recently introduced Quad Operational Amplifiers, these dual operational amplifiers feature 1) low power drain, 2) a common mode input voltage range extending to ground/ $V_{EE}$ , 3) single supply or split supply operation and 4) pinouts compatible with the popular MC1558 dual operational amplifier. The LM158 series is equivalent to one-half of an LM124.

These amplifiers have several distinct advantages over standard operational amplifier types in single supply applications. They can operate at supply voltages as low as 3.0 V or as high as 32 V, with quiescent currents about one-fifth of those associated with the MC1741 (on a per amplifier basis). The common mode input range includes the negative supply, thereby eliminating the necessity for external biasing components in many applications. The output voltage range also includes the negative power supply voltage.

- Short Circuit Protected Outputs
- True Differential Input Stage
- Single Supply Operation: 3.0 V to 32 V
- Low Input Bias Currents
- Internally Compensated
- Common Mode Range Extends to Negative Supply
- Single and Split Supply Operation
- Similar Performance to the Popular MC1558
- ESD Clamps on the Inputs Increase Ruggedness of the Device without Affecting Operation

## MAXIMUM RATINGS ( $T_A = +25^\circ\text{C}$ , unless otherwise noted.)

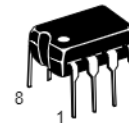
Rating	Symbol	LM258 LM358	LM2904 LM2904V	Unit
Power Supply Voltages Single Supply Split Supplies	$V_{CC}$ $V_{CC}, V_{EE}$	32 $\pm 16$	26 $\pm 13$	Vdc
Input Differential Voltage Range (Note 1)	$V_{IDR}$	$\pm 32$	$\pm 26$	Vdc
Input Common Mode Voltage Range (Note 2)	$V_{ICR}$	-0.3 to 32	-0.3 to 26	Vdc
Output Short Circuit Duration	$t_{SC}$	Continuous		
Junction Temperature	$T_J$	150		$^\circ\text{C}$
Storage Temperature Range	$T_{stg}$	-55 to +125		$^\circ\text{C}$
Operating Ambient Temperature Range	$T_A$			$^\circ\text{C}$
LM258		-25 to +85	-	
LM358		0 to +70	-	
LM2904		-	-40 to +105	
LM2904V		-	-40 to +125	

- NOTES:** 1. Split Power Supplies.  
2. For Supply Voltages less than 32 V for the LM258/358 and 26 V for the LM2904, the absolute maximum input voltage is equal to the supply voltage.

# LM358, LM258, LM2904, LM2904V

## DUAL DIFFERENTIAL INPUT OPERATIONAL AMPLIFIERS

### SEMICONDUCTOR TECHNICAL DATA

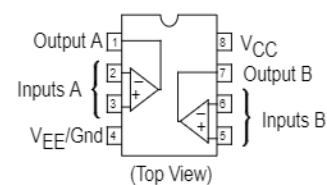


**N SUFFIX**  
PLASTIC PACKAGE  
CASE 626



**D SUFFIX**  
PLASTIC PACKAGE  
CASE 751  
(SO-8)

## PIN CONNECTIONS



## ORDERING INFORMATION

Device	Operating Temperature Range	Package
LM2904D	$T_A = -40^\circ$ to $+105^\circ\text{C}$	SO-8
LM2904N		Plastic DIP
LM2904VD	$T_A = -40^\circ$ to $+125^\circ\text{C}$	SO-8
LM2904VN		Plastic DIP
LM258D	$T_A = -25^\circ$ to $+85^\circ\text{C}$	SO-8
LM258N		Plastic DIP
LM358D	$T_A = 0^\circ$ to $+70^\circ\text{C}$	SO-8
LM358N		Plastic DIP

# LM358, LM258, LM2904, LM2904V

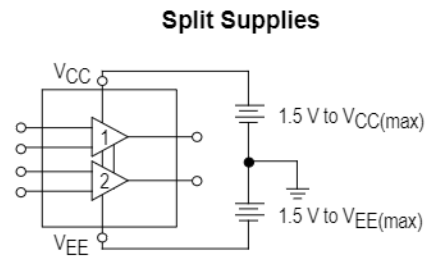
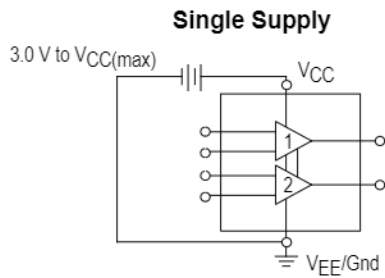
## ELECTRICAL CHARACTERISTICS (V<sub>CC</sub> = 5.0 V, V<sub>EE</sub> = Gnd, T<sub>A</sub> = 25°C, unless otherwise noted.)

Characteristic	Symbol	LM258			LM358			LM2904			LM2904V			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Input Offset Voltage V <sub>CC</sub> = 5.0 V to 30 V (26 V for LM2904, V), V <sub>IC</sub> = 0 V to V <sub>CC</sub> -1.7 V, V <sub>O</sub> ≈ 1.4 V, R <sub>S</sub> = 0 Ω T <sub>A</sub> = 25°C T <sub>A</sub> = T <sub>high</sub> (Note 1) T <sub>A</sub> = T <sub>low</sub> (Note 1)	V <sub>IO</sub>	-	2.0	5.0	-	2.0	7.0	-	2.0	7.0	-	-	-	mV
Average Temperature Coefficient of Input Offset Voltage T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub> (Note 1)	ΔV <sub>IO</sub> /ΔT	-	7.0	-	-	7.0	-	-	7.0	-	-	7.0	-	μV/°C
Input Offset Current T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub> (Note 1)	I <sub>IO</sub>	-	3.0	30	-	5.0	50	-	5.0	50	-	5.0	50	nA
Input Bias Current T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub> (Note 1)	I <sub>IB</sub>	-	-45	-150	-	-45	-250	-	-45	-250	-	-45	-250	nA
Average Temperature Coefficient of Input Offset Current T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub> (Note 1)	ΔI <sub>IO</sub> /ΔT	-	10	-	-	10	-	-	10	-	-	10	-	pA/°C
Input Common Mode Voltage Range (Note 2), V <sub>CC</sub> = 30 V (26 V for LM2904, V) V <sub>CC</sub> = 30 V (26 V for LM2904, V), T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub>	V <sub>ICR</sub>	0	-	28.3	0	-	28.3	0	-	24.3	0	-	24.3	V
Differential Input Voltage Range	V <sub>IDR</sub>	-	-	V <sub>CC</sub>	-	-	V <sub>CC</sub>	-	-	V <sub>CC</sub>	-	-	V <sub>CC</sub>	V
Large Signal Open Loop Voltage Gain R <sub>L</sub> = 2.0 kΩ, V <sub>CC</sub> = 15 V, For Large V <sub>O</sub> Swing, T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub> (Note 1)	A <sub>VOL</sub>	50	100	-	25	100	-	25	100	-	25	100	-	V/mV
Channel Separation 1.0 kHz ≤ f ≤ 20 kHz, Input Referenced	CS	-	-120	-	-	-120	-	-	-120	-	-	-120	-	dB
Common Mode Rejection R <sub>S</sub> ≤ 10 kΩ	CMR	70	85	-	65	70	-	50	70	-	50	70	-	dB
Power Supply Rejection	PSR	65	100	-	65	100	-	50	100	-	50	100	-	dB
Output Voltage—High Limit (T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub> ) (Note 1) V <sub>CC</sub> = 5.0 V, R <sub>L</sub> = 2.0 kΩ, T <sub>A</sub> = 25°C V <sub>CC</sub> = 30 V (26 V for LM2904, V), R <sub>L</sub> = 2.0 kΩ V <sub>CC</sub> = 30 V (26 V for LM2904, V), R <sub>L</sub> = 10 kΩ	V <sub>OH</sub>	3.3	3.5	-	3.3	3.5	-	3.3	3.5	-	3.3	3.5	-	V
Output Voltage—Low Limit V <sub>CC</sub> = 5.0 V, R <sub>L</sub> = 10 kΩ, T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub> (Note 1)	V <sub>OL</sub>	-	5.0	20	-	5.0	20	-	5.0	20	-	5.0	20	mV
Output Source Current V <sub>ID</sub> = +1.0 V, V <sub>CC</sub> = 15 V	I <sub>O+</sub>	20	40	-	20	40	-	20	40	-	20	40	-	mA
Output Sink Current V <sub>ID</sub> = -1.0 V, V <sub>CC</sub> = 15 V V <sub>ID</sub> = -1.0 V, V <sub>O</sub> = 200 mV	I <sub>O-</sub>	10	20	-	10	20	-	10	20	-	10	20	-	mA
Output Short Circuit to Ground (Note 3)	I <sub>SC</sub>	-	40	60	-	40	60	-	40	60	-	40	60	mA
Power Supply Current (T <sub>A</sub> = T <sub>high</sub> to T <sub>low</sub> ) (Note 1) V <sub>CC</sub> = 30 V (26 V for LM2904, V), V <sub>O</sub> = 0 V, R <sub>L</sub> = ∞ V <sub>CC</sub> = 5 V, V <sub>O</sub> = 0 V, R <sub>L</sub> = ∞	I <sub>CC</sub>	-	1.5	3.0	-	1.5	3.0	-	1.5	3.0	-	1.5	3.0	mA
		-	0.7	1.2	-	0.7	1.2	-	0.7	1.2	-	0.7	1.2	mA

NOTES: 1. T<sub>low</sub> = -40°C for LM2904  
           = -40°C for LM2904V  
           = -25°C for LM258  
           = 0°C for LM358  
           T<sub>high</sub> = +105°C for LM2904  
                 = +125°C for LM2904V  
                 = +85°C for LM258  
                 = +70°C for LM358

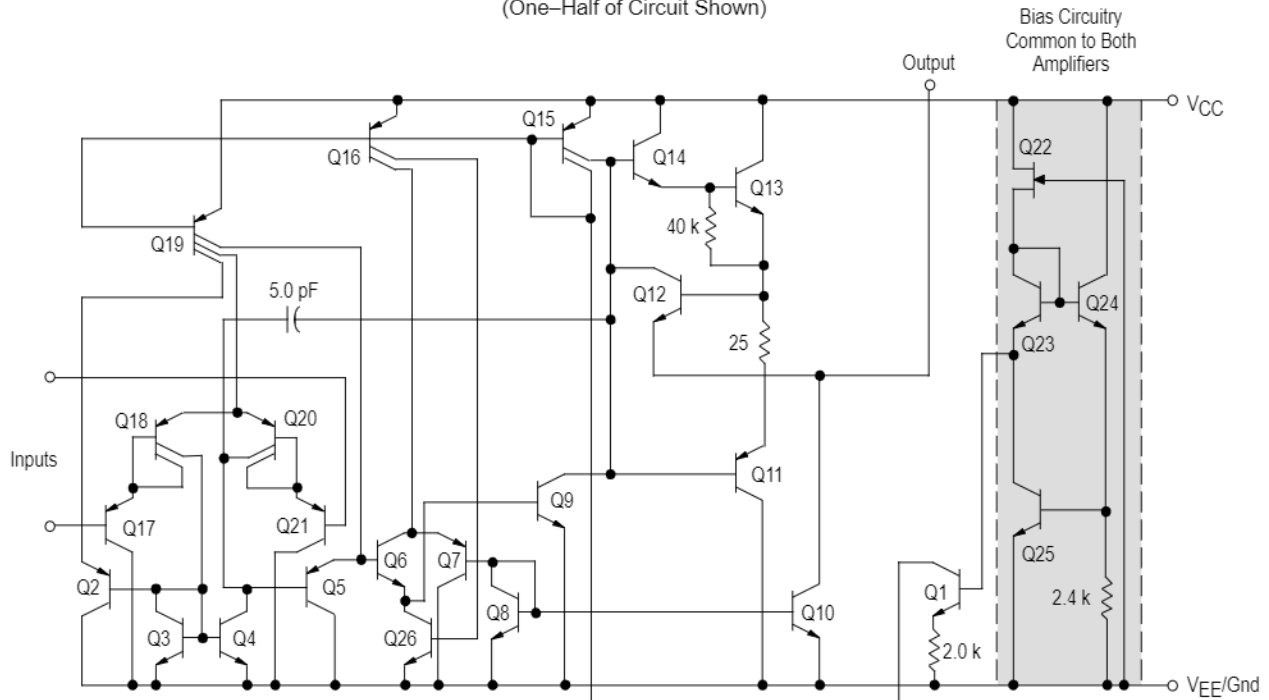
2. The input common mode voltage or either input signal voltage should not be allowed to go negative by more than 0.3 V. The upper end of the common mode voltage range is V<sub>CC</sub> -1.7 V.  
 3. Short circuits from the output to V<sub>CC</sub> can cause excessive heating and eventual destruction. Destructive dissipation can result from simultaneous shorts on all amplifiers.

# LM358, LM258, LM2904, LM2904V



## Representative Schematic Diagram

(One-Half of Circuit Shown)



## CIRCUIT DESCRIPTION

The LM258 series is made using two internally compensated, two-stage operational amplifiers. The first stage of each consists of differential input devices Q20 and Q18 with input buffer transistors Q21 and Q17 and the differential to single ended converter Q3 and Q4. The first stage performs not only the first stage gain function but also performs the level shifting and transconductance reduction functions. By reducing the transconductance, a smaller compensation capacitor (only 5.0 pF) can be employed, thus saving chip area. The transconductance reduction is accomplished by splitting the collectors of Q20 and Q18. Another feature of this input stage is that the input common mode range can include the negative supply or ground, in single supply operation, without saturating either the input devices or the differential to single-ended converter. The second stage consists of a standard current source load amplifier stage.

Each amplifier is biased from an internal-voltage regulator which has a low temperature coefficient thus giving each amplifier good temperature characteristics as well as excellent power supply rejection.

## Large Signal Voltage Follower Response

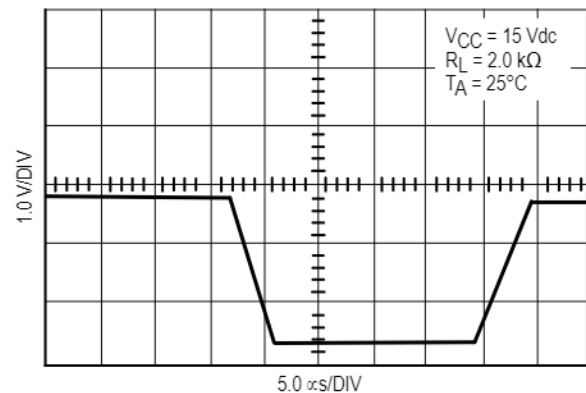




Figure 1. Input Voltage Range

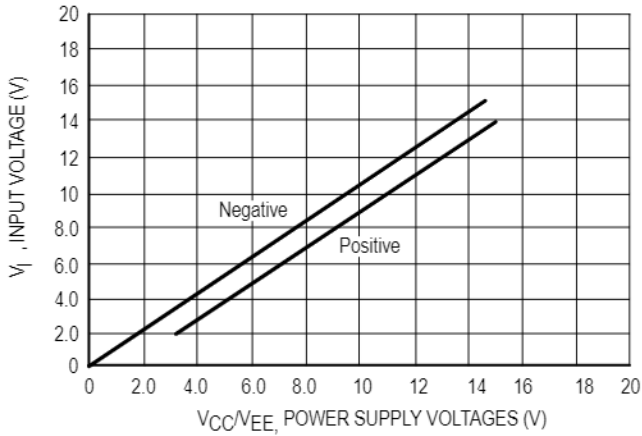


Figure 2. Large-Signal Open Loop Voltage Gain

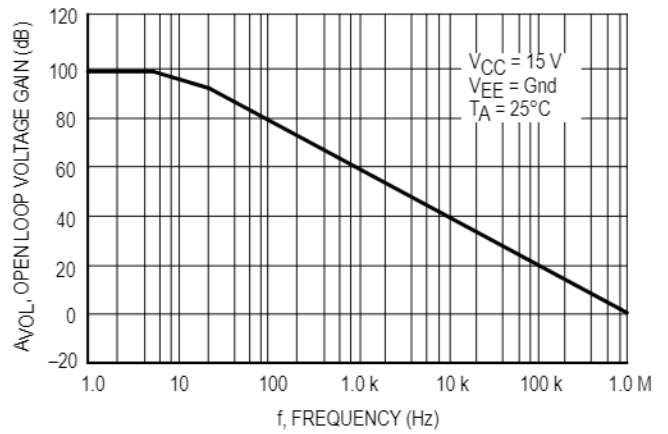


Figure 3. Large-Signal Frequency Response

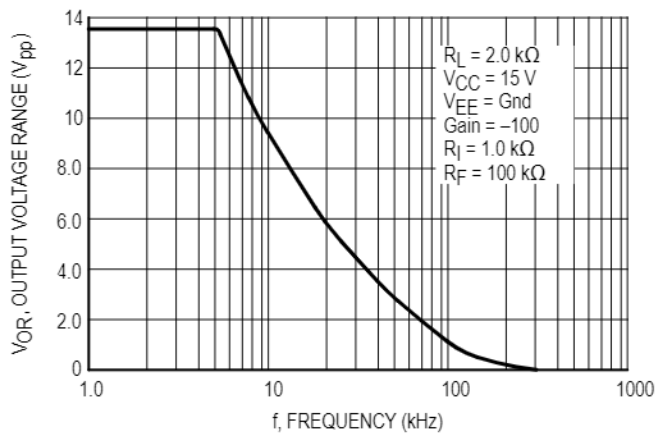


Figure 4. Small Signal Voltage Follower Pulse Response (Noninverting)

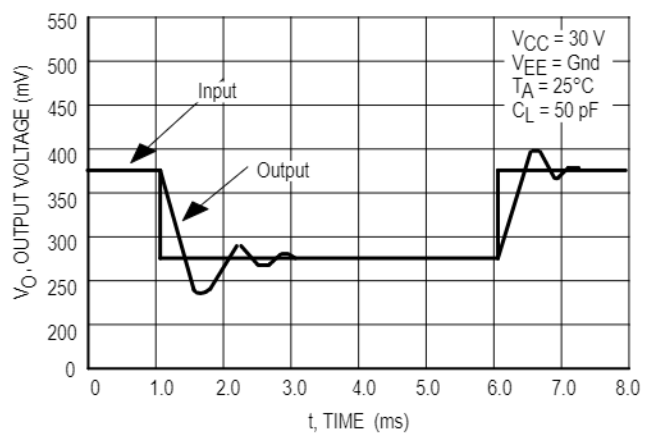


Figure 5. Power Supply Current versus Power Supply Voltage

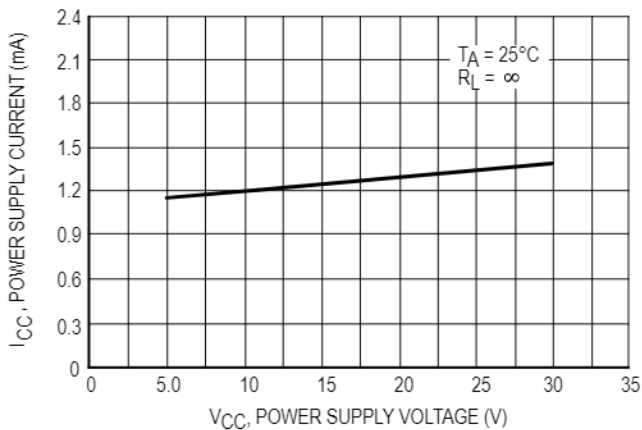
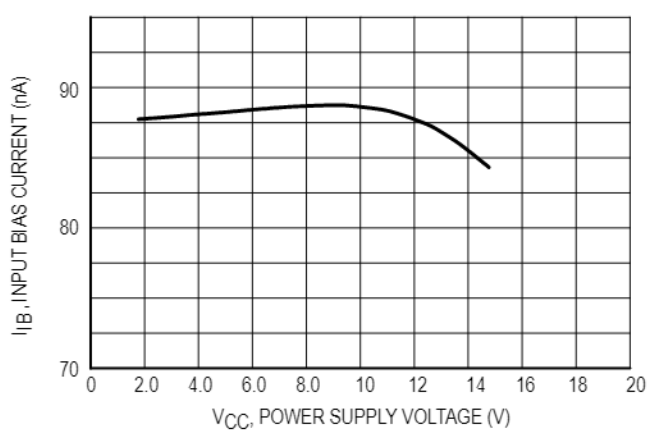


Figure 6. Input Bias Current versus Supply Voltage



# LM358, LM258, LM2904, LM2904V

Figure 7. Voltage Reference

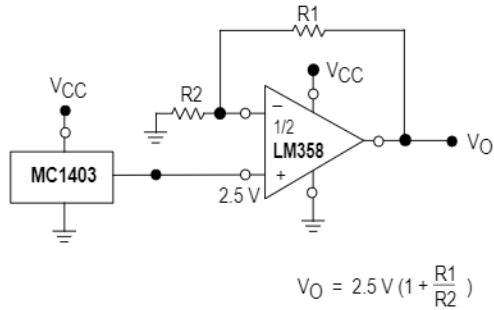


Figure 8. Wien Bridge Oscillator

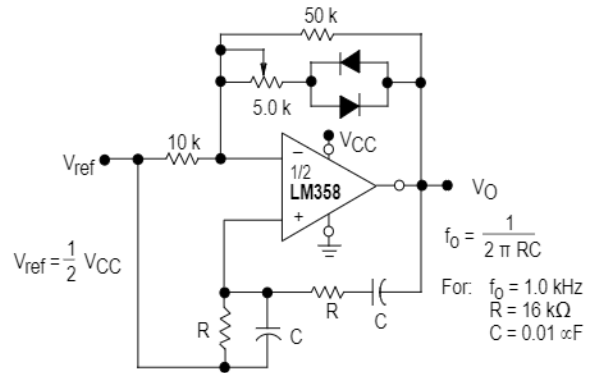


Figure 9. High Impedance Differential Amplifier

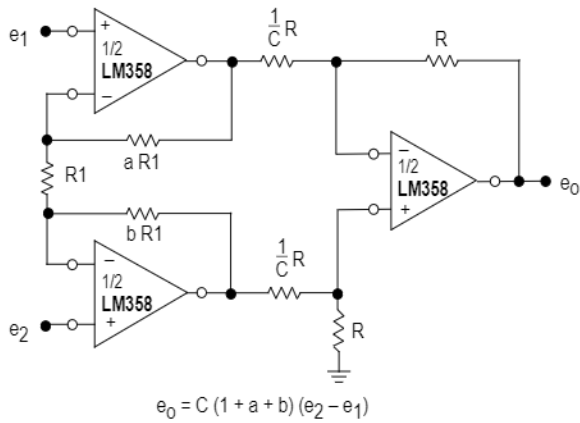


Figure 10. Comparator with Hysteresis

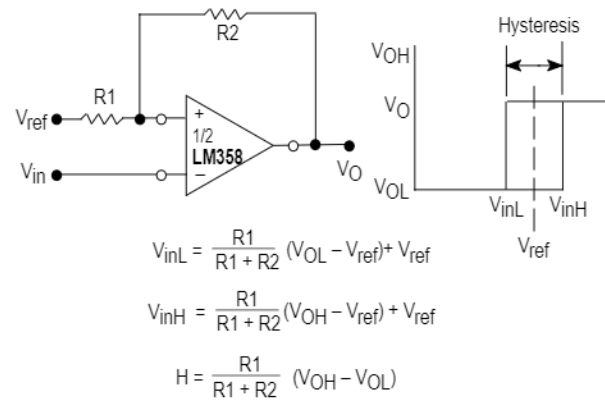


Figure 11. Bi-Quad Filter

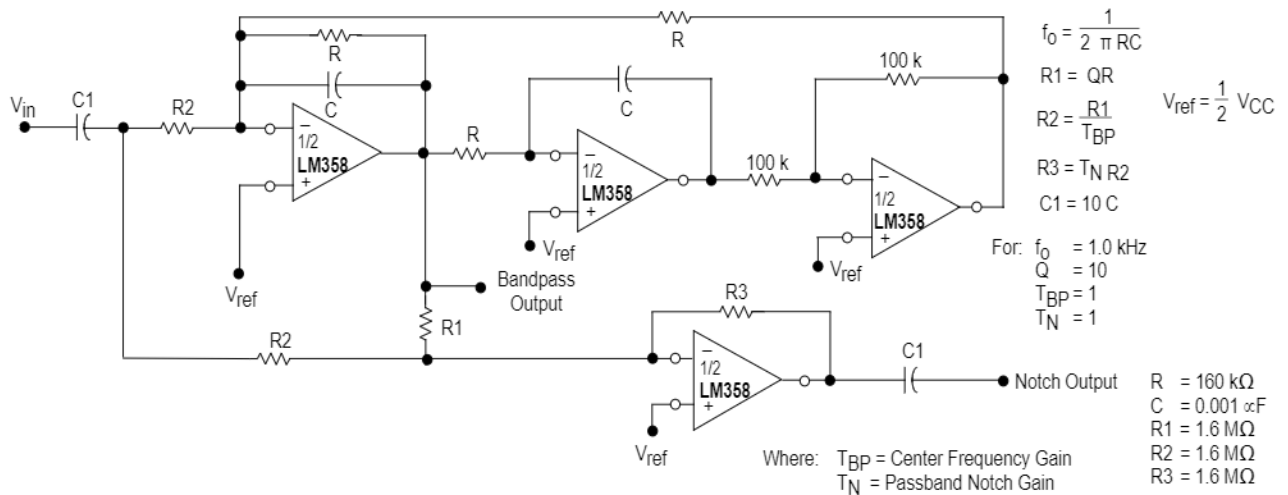
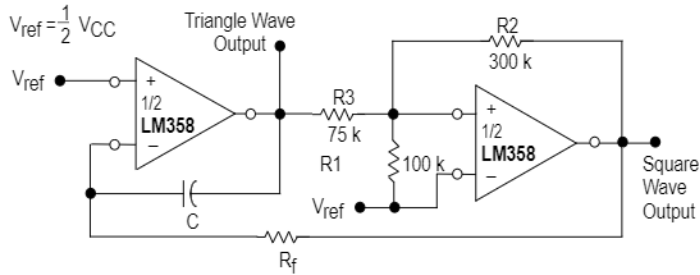
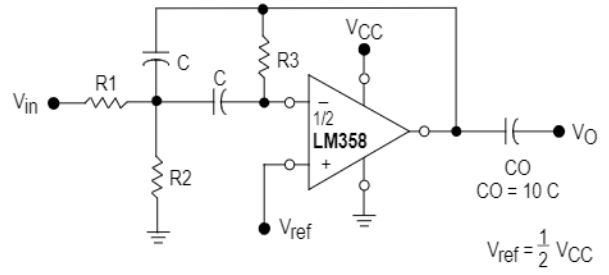


Figure 12. Function Generator



$$f = \frac{R1 + R_C}{4 C R_f R1} \quad \text{if, } R3 = \frac{R2 R1}{R2 + R1}$$

Figure 13. Multiple Feedback Bandpass Filter



Given:  $f_0$  = center frequency  
 $A(f_0)$  = gain at center frequency

Choose value  $f_0, C$

$$\text{Then: } R3 = \frac{Q}{\pi f_0 C}$$

$$R1 = \frac{R3}{2 A(f_0)}$$

$$R2 = \frac{R1 R3}{4 Q^2 R1 - R3}$$


For less than 10% error from operational amplifier.  $\frac{Q_0 f_0}{BW} < 0.1$

Where  $f_0$  and BW are expressed in Hz.

If source impedance varies, filter may be preceded with voltage follower buffer to stabilize filter parameters.



# LM358, LM258, LM2904, LM2904V

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

**How to reach us:**

**USA/EUROPE/Locations Not Listed:** Motorola Literature Distribution;  
P.O. Box 20912; Phoenix, Arizona 85036. 1-800-441-2447 or 602-303-5454

**MFAX:** RMFAX0@email.sps.mot.com – TOUCHTONE 602-244-6609  
**INTERNET:** <http://Design-NET.com>

**JAPAN:** Nippon Motorola Ltd.; Tatsumi-SPD-JLDC, 6F Seibu-Butsuryu-Center,  
3-14-2 Tatsumi Koto-Ku, Tokyo 135, Japan. 03-81-3521-8315

**ASIA/PACIFIC:** Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,  
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298



LM358/D



## Features

- High Performance, Low Power AVR<sup>®</sup> 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 program memory
  - 256/512/512/1K Bytes EEPROM
  - 512/1K/1K/2K Bytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C<sup>(1)</sup>
  - 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<sup>2</sup>C 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
- Temperature Range:
  - -40°C to 85°C
- Speed Grade:
  - 0 - 4 MHz@1.8 - 5.5V, 0 - 10 MHz@2.7 - 5.5.V, 0 - 20 MHz @ 4.5 - 5.5V
- Power Consumption at 1 MHz, 1.8V, 25°C
  - Active Mode: 0.2 mA
  - Power-down Mode: 0.1 µA
  - Power-save Mode: 0.75 µA (Including 32 kHz RTC)



**8-bit AVR<sup>®</sup>  
Microcontroller  
with 4/8/16/32K  
Bytes In-System  
Programmable  
Flash**

**ATmega48A  
ATmega48PA  
ATmega88A  
ATmega88PA  
ATmega168A  
ATmega168PA  
ATmega328  
ATmega328P**

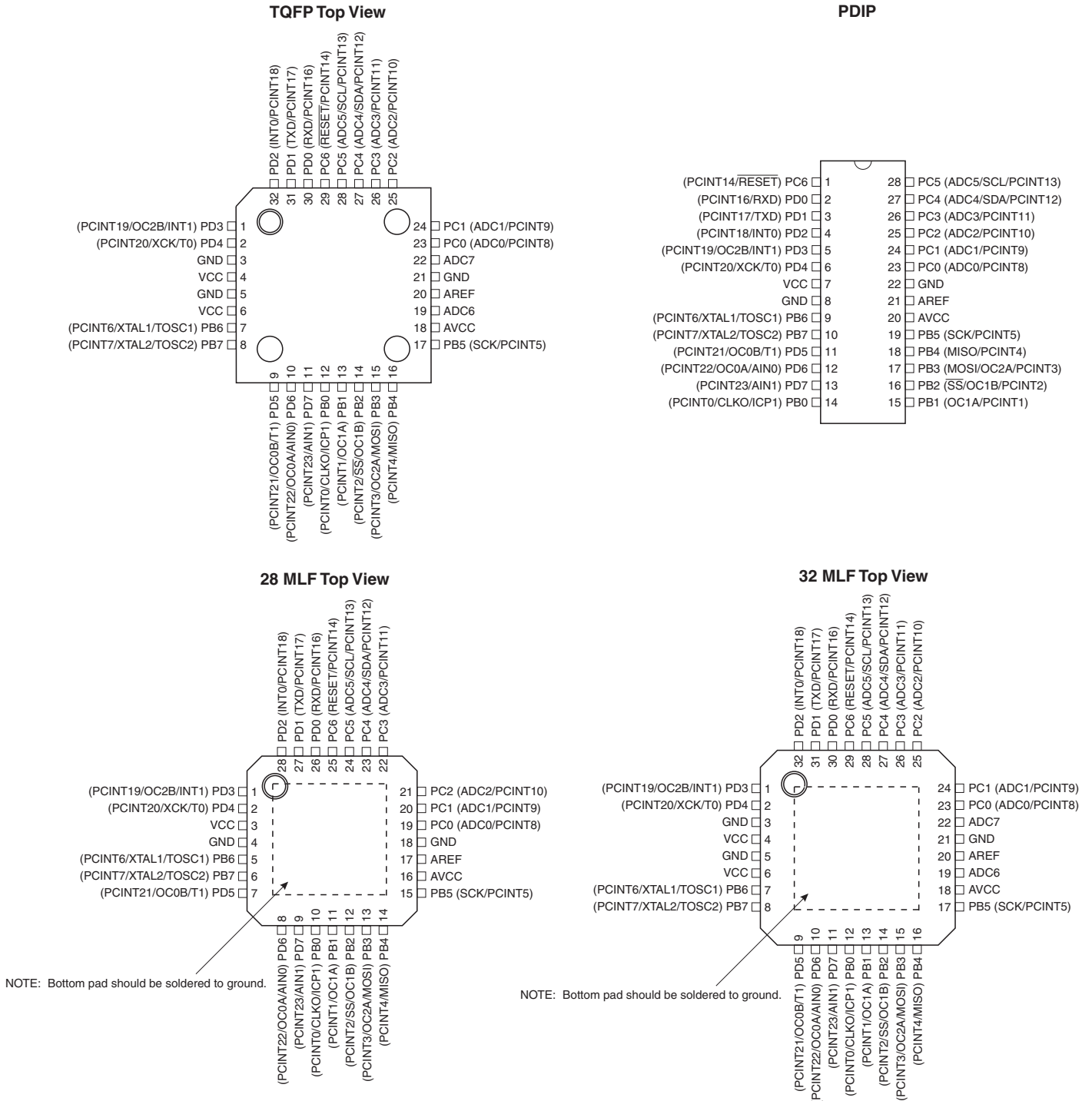
**Summary**

Rev. 8271BS-AVR-04/10



## 1. Pin Configurations

Figure 1-1. Pinout ATmega48A/48PA/88A/88PA/168A/168PA/328/328P



## 1.1 Pin Descriptions

### 1.1.1 VCC

Digital supply voltage.

### 1.1.2 GND

Ground.

### 1.1.3 Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7...6 is used as TOSC2...1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

The various special features of Port B are elaborated in [and "System Clock and Clock Options" on page 26.](#)

### 1.1.4 Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5...0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

### 1.1.5 PC6/ $\overline{\text{RESET}}$

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. The minimum pulse length is given in [Table 28-12 on page 323](#). Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated in ["Alternate Functions of Port C" on page 86.](#)

### 1.1.6 Port D (PD7:0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.



The various special features of Port D are elaborated in ["Alternate Functions of Port D"](#) on page 89.

## 1.1.7 $AV_{CC}$

$AV_{CC}$  is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. It should be externally connected to  $V_{CC}$ , even if the ADC is not used. If the ADC is used, it should be connected to  $V_{CC}$  through a low-pass filter. Note that PC6...4 use digital supply voltage,  $V_{CC}$ .

## 1.1.8 AREF

AREF is the analog reference pin for the A/D Converter.

## 1.1.9 ADC7:6 (TQFP and QFN/MLF Package Only)

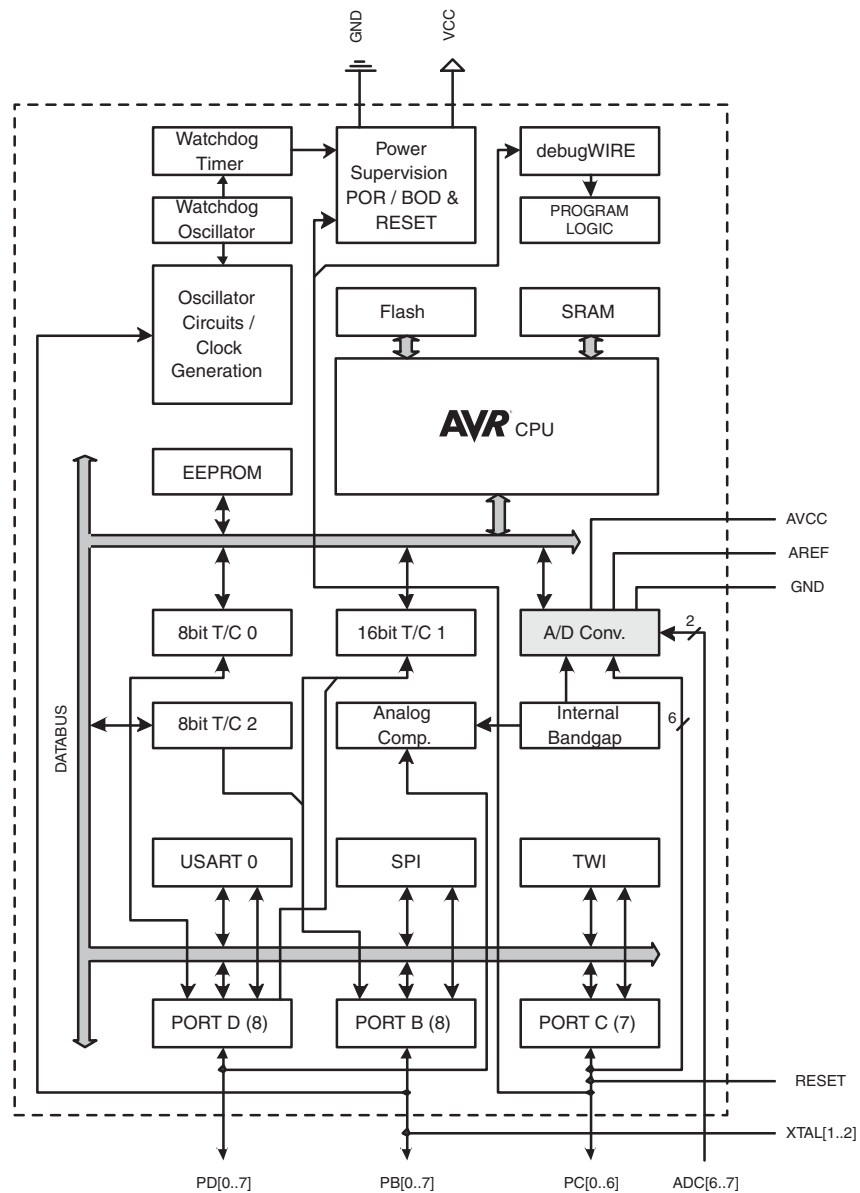
In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

## 2. Overview

The ATmega48A/48PA/88A/88PA/168A/168PA/328/328P 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 ATmega48A/48PA/88A/88PA/168A/168PA/328/328P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

### 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 ATmega48A/48PA/88A/88PA/168A/168PA/328/328P 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 ATmega48A/48PA/88A/88PA/168A/168PA/328/328P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega48A/48PA/88A/88PA/168A/168PA/328/328P 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 Processors

The ATmega48A/48PA/88A/88PA/168A/168PA/328/328P 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 devices.

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

Device	Flash	EEPROM	RAM	Interrupt Vector Size
ATmega48A	4K Bytes	256 Bytes	512 Bytes	1 instruction word/vector
ATmega48PA	4K Bytes	256 Bytes	512 Bytes	1 instruction word/vector
ATmega88A	8K Bytes	512 Bytes	1K Bytes	1 instruction word/vector
ATmega88PA	8K Bytes	512 Bytes	1K Bytes	1 instruction word/vector
ATmega168A	16K Bytes	512 Bytes	1K Bytes	2 instruction words/vector

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

Device	Flash	EEPROM	RAM	Interrupt Vector Size
ATmega168PA	16K Bytes	512 Bytes	1K Bytes	2 instruction words/vector
ATmega328	32K Bytes	1K Bytes	2K Bytes	2 instruction words/vector
ATmega328P	32K Bytes	1K Bytes	2K Bytes	2 instruction words/vector

ATmega48A/48PA/88A/88PA/168A/168PA/328/328P 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 ATmega 48A/48PA there is no Read-While-Write support and no separate Boot Loader Section. The SPM instruction can execute from the entire Flash.

## 3. Resources

A comprehensive set of development tools, application notes and datasheets are available for download on <http://www.atmel.com/avr>.

## 4. Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0xFF)	Reserved	–	–	–	–	–	–	–	–	
(0xFE)	Reserved	–	–	–	–	–	–	–	–	
(0xFD)	Reserved	–	–	–	–	–	–	–	–	
(0xFC)	Reserved	–	–	–	–	–	–	–	–	
(0xFB)	Reserved	–	–	–	–	–	–	–	–	
(0xFA)	Reserved	–	–	–	–	–	–	–	–	
(0xF9)	Reserved	–	–	–	–	–	–	–	–	
(0xF8)	Reserved	–	–	–	–	–	–	–	–	
(0xF7)	Reserved	–	–	–	–	–	–	–	–	
(0xF6)	Reserved	–	–	–	–	–	–	–	–	
(0xF5)	Reserved	–	–	–	–	–	–	–	–	
(0xF4)	Reserved	–	–	–	–	–	–	–	–	
(0xF3)	Reserved	–	–	–	–	–	–	–	–	
(0xF2)	Reserved	–	–	–	–	–	–	–	–	
(0xF1)	Reserved	–	–	–	–	–	–	–	–	
(0xF0)	Reserved	–	–	–	–	–	–	–	–	
(0xEF)	Reserved	–	–	–	–	–	–	–	–	
(0xEE)	Reserved	–	–	–	–	–	–	–	–	
(0xED)	Reserved	–	–	–	–	–	–	–	–	
(0xEC)	Reserved	–	–	–	–	–	–	–	–	
(0xEB)	Reserved	–	–	–	–	–	–	–	–	
(0xEA)	Reserved	–	–	–	–	–	–	–	–	
(0xE9)	Reserved	–	–	–	–	–	–	–	–	
(0xE8)	Reserved	–	–	–	–	–	–	–	–	
(0xE7)	Reserved	–	–	–	–	–	–	–	–	
(0xE6)	Reserved	–	–	–	–	–	–	–	–	
(0xE5)	Reserved	–	–	–	–	–	–	–	–	
(0xE4)	Reserved	–	–	–	–	–	–	–	–	
(0xE3)	Reserved	–	–	–	–	–	–	–	–	
(0xE2)	Reserved	–	–	–	–	–	–	–	–	
(0xE1)	Reserved	–	–	–	–	–	–	–	–	
(0xE0)	Reserved	–	–	–	–	–	–	–	–	
(0xDF)	Reserved	–	–	–	–	–	–	–	–	
(0xDE)	Reserved	–	–	–	–	–	–	–	–	
(0xDD)	Reserved	–	–	–	–	–	–	–	–	
(0xDC)	Reserved	–	–	–	–	–	–	–	–	
(0xDB)	Reserved	–	–	–	–	–	–	–	–	
(0xDA)	Reserved	–	–	–	–	–	–	–	–	
(0xD9)	Reserved	–	–	–	–	–	–	–	–	
(0xD8)	Reserved	–	–	–	–	–	–	–	–	
(0xD7)	Reserved	–	–	–	–	–	–	–	–	
(0xD6)	Reserved	–	–	–	–	–	–	–	–	
(0xD5)	Reserved	–	–	–	–	–	–	–	–	
(0xD4)	Reserved	–	–	–	–	–	–	–	–	
(0xD3)	Reserved	–	–	–	–	–	–	–	–	
(0xD2)	Reserved	–	–	–	–	–	–	–	–	
(0xD1)	Reserved	–	–	–	–	–	–	–	–	
(0xD0)	Reserved	–	–	–	–	–	–	–	–	
(0xCF)	Reserved	–	–	–	–	–	–	–	–	
(0xCE)	Reserved	–	–	–	–	–	–	–	–	
(0xCD)	Reserved	–	–	–	–	–	–	–	–	
(0xCC)	Reserved	–	–	–	–	–	–	–	–	
(0xCB)	Reserved	–	–	–	–	–	–	–	–	
(0xCA)	Reserved	–	–	–	–	–	–	–	–	
(0xC9)	Reserved	–	–	–	–	–	–	–	–	
(0xC8)	Reserved	–	–	–	–	–	–	–	–	
(0xC7)	Reserved	–	–	–	–	–	–	–	–	
(0xC6)	UDR0	USART I/O Data Register								196
(0xC5)	UBRR0H					USART Baud Rate Register High				200
(0xC4)	UBRR0L	USART Baud Rate Register Low								200
(0xC3)	Reserved	–	–	–	–	–	–	–	–	
(0xC2)	UCSROC	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01 /UDORD0	UCSZ00 /UCPHA0	UCPOL0	198/213

# ATmega48A/48PA/88A/88PA/168A/168PA/328/328P

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0xC1)	UCSR0B	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80	197
(0xC0)	UCSR0A	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0	196
(0xBF)	Reserved	–	–	–	–	–	–	–	–	
(0xBE)	Reserved	–	–	–	–	–	–	–	–	
(0xBD)	TWAMR	TWAM6	TWAM5	TWAM4	TWAM3	TWAM2	TWAM1	TWAM0	–	245
(0xBC)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE	242
(0xBB)	TWDR	2-wire Serial Interface Data Register								244
(0xBA)	TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	245
(0xB9)	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0	244
(0xB8)	TWBR	2-wire Serial Interface Bit Rate Register								242
(0xB7)	Reserved	–	–	–	–	–	–	–	–	
(0xB6)	ASSR	–	EXCLK	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB	165
(0xB5)	Reserved	–	–	–	–	–	–	–	–	
(0xB4)	OCR2B	Timer/Counter2 Output Compare Register B								163
(0xB3)	OCR2A	Timer/Counter2 Output Compare Register A								163
(0xB2)	TCNT2	Timer/Counter2 (8-bit)								163
(0xB1)	TCCR2B	FOC2A	FOC2B	–	–	WGM22	CS22	CS21	CS20	162
(0xB0)	TCCR2A	COM2A1	COM2A0	COM2B1	COM2B0	–	–	WGM21	WGM20	159
(0xAF)	Reserved	–	–	–	–	–	–	–	–	
(0xAE)	Reserved	–	–	–	–	–	–	–	–	
(0xAD)	Reserved	–	–	–	–	–	–	–	–	
(0xAC)	Reserved	–	–	–	–	–	–	–	–	
(0xAB)	Reserved	–	–	–	–	–	–	–	–	
(0xAA)	Reserved	–	–	–	–	–	–	–	–	
(0xA9)	Reserved	–	–	–	–	–	–	–	–	
(0xA8)	Reserved	–	–	–	–	–	–	–	–	
(0xA7)	Reserved	–	–	–	–	–	–	–	–	
(0xA6)	Reserved	–	–	–	–	–	–	–	–	
(0xA5)	Reserved	–	–	–	–	–	–	–	–	
(0xA4)	Reserved	–	–	–	–	–	–	–	–	
(0xA3)	Reserved	–	–	–	–	–	–	–	–	
(0xA2)	Reserved	–	–	–	–	–	–	–	–	
(0xA1)	Reserved	–	–	–	–	–	–	–	–	
(0xA0)	Reserved	–	–	–	–	–	–	–	–	
(0x9F)	Reserved	–	–	–	–	–	–	–	–	
(0x9E)	Reserved	–	–	–	–	–	–	–	–	
(0x9D)	Reserved	–	–	–	–	–	–	–	–	
(0x9C)	Reserved	–	–	–	–	–	–	–	–	
(0x9B)	Reserved	–	–	–	–	–	–	–	–	
(0x9A)	Reserved	–	–	–	–	–	–	–	–	
(0x99)	Reserved	–	–	–	–	–	–	–	–	
(0x98)	Reserved	–	–	–	–	–	–	–	–	
(0x97)	Reserved	–	–	–	–	–	–	–	–	
(0x96)	Reserved	–	–	–	–	–	–	–	–	
(0x95)	Reserved	–	–	–	–	–	–	–	–	
(0x94)	Reserved	–	–	–	–	–	–	–	–	
(0x93)	Reserved	–	–	–	–	–	–	–	–	
(0x92)	Reserved	–	–	–	–	–	–	–	–	
(0x91)	Reserved	–	–	–	–	–	–	–	–	
(0x90)	Reserved	–	–	–	–	–	–	–	–	
(0x8F)	Reserved	–	–	–	–	–	–	–	–	
(0x8E)	Reserved	–	–	–	–	–	–	–	–	
(0x8D)	Reserved	–	–	–	–	–	–	–	–	
(0x8C)	Reserved	–	–	–	–	–	–	–	–	
(0x8B)	OCR1BH	Timer/Counter1 - Output Compare Register B High Byte								139
(0x8A)	OCR1BL	Timer/Counter1 - Output Compare Register B Low Byte								139
(0x89)	OCR1AH	Timer/Counter1 - Output Compare Register A High Byte								139
(0x88)	OCR1AL	Timer/Counter1 - Output Compare Register A Low Byte								139
(0x87)	ICR1H	Timer/Counter1 - Input Capture Register High Byte								139
(0x86)	ICR1L	Timer/Counter1 - Input Capture Register Low Byte								139
(0x85)	TCNT1H	Timer/Counter1 - Counter Register High Byte								139
(0x84)	TCNT1L	Timer/Counter1 - Counter Register Low Byte								139
(0x83)	Reserved	–	–	–	–	–	–	–	–	
(0x82)	TCCR1C	FOC1A	FOC1B	–	–	–	–	–	–	138
(0x81)	TCCR1B	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	137
(0x80)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	135

# ATmega48A/48PA/88A/88PA/168A/168PA/328/328P

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0x7F)	DIDR1	–	–	–	–	–	–	AIN1D	AIN0D	250
(0x7E)	DIDR0	–	–	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	267
(0x7D)	Reserved	–	–	–	–	–	–	–	–	
(0x7C)	ADMUX	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	263
(0x7B)	ADCSRB	–	ACME	–	–	–	ADTS2	ADTS1	ADTS0	266
(0x7A)	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	264
(0x79)	ADCH	ADC Data Register High byte								266
(0x78)	ADCL	ADC Data Register Low byte								266
(0x77)	Reserved	–	–	–	–	–	–	–	–	
(0x76)	Reserved	–	–	–	–	–	–	–	–	
(0x75)	Reserved	–	–	–	–	–	–	–	–	
(0x74)	Reserved	–	–	–	–	–	–	–	–	
(0x73)	Reserved	–	–	–	–	–	–	–	–	
(0x72)	Reserved	–	–	–	–	–	–	–	–	
(0x71)	Reserved	–	–	–	–	–	–	–	–	
(0x70)	TIMSK2	–	–	–	–	–	OCIE2B	OCIE2A	TOIE2	164
(0x6F)	TIMSK1	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	140
(0x6E)	TIMSK0	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	112
(0x6D)	PCMSK2	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16	75
(0x6C)	PCMSK1	–	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8	75
(0x6B)	PCMSK0	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	75
(0x6A)	Reserved	–	–	–	–	–	–	–	–	
(0x69)	EICRA	–	–	–	–	ISC11	ISC10	ISC01	ISC00	72
(0x68)	PCICR	–	–	–	–	–	PCIE2	PCIE1	PCIE0	
(0x67)	Reserved	–	–	–	–	–	–	–	–	
(0x66)	OSCCAL	Oscillator Calibration Register								37
(0x65)	Reserved	–	–	–	–	–	–	–	–	
(0x64)	PRR	PRTW1	PRTIM2	PRTIM0	–	PRTIM1	PRSPI	PRUSART0	PRADC	42
(0x63)	Reserved	–	–	–	–	–	–	–	–	
(0x62)	Reserved	–	–	–	–	–	–	–	–	
(0x61)	CLKPR	CLKPCE	–	–	–	CLKPS3	CLKPS2	CLKPS1	CLKPS0	37
(0x60)	WDTCR	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0	55
0x3F (0x5F)	SREG	I	T	H	S	V	N	Z	C	9
0x3E (0x5E)	SPH	–	–	–	–	–	(SP10) <sup>5</sup>	SP9	SP8	12
0x3D (0x5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	12
0x3C (0x5C)	Reserved	–	–	–	–	–	–	–	–	
0x3B (0x5B)	Reserved	–	–	–	–	–	–	–	–	
0x3A (0x5A)	Reserved	–	–	–	–	–	–	–	–	
0x39 (0x59)	Reserved	–	–	–	–	–	–	–	–	
0x38 (0x58)	Reserved	–	–	–	–	–	–	–	–	
0x37 (0x57)	SPMCSR	SPMIE	(RWWSB) <sup>5</sup>	–	(RWWSR) <sup>5</sup>	BLBSET	PGWRT	PGERS	SELFPRGEN	294
0x36 (0x56)	Reserved	–	–	–	–	–	–	–	–	
0x35 (0x55)	MCUCR	–	BODS <sup>(6)</sup>	BODSE <sup>(6)</sup>	PUD	–	–	IVSEL	IVCE	45/69/93
0x34 (0x54)	MCUSR	–	–	–	–	WDRF	BORF	EXTRF	PORF	55
0x33 (0x53)	SMCR	–	–	–	–	SM2	SM1	SM0	SE	40
0x32 (0x52)	Reserved	–	–	–	–	–	–	–	–	
0x31 (0x51)	Reserved	–	–	–	–	–	–	–	–	
0x30 (0x50)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	248
0x2F (0x4F)	Reserved	–	–	–	–	–	–	–	–	
0x2E (0x4E)	SPDR	SPI Data Register								176
0x2D (0x4D)	SPSR	SPIF	WCOL	–	–	–	–	–	SPI2X	175
0x2C (0x4C)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	174
0x2B (0x4B)	GPIOR2	General Purpose I/O Register 2								25
0x2A (0x4A)	GPIOR1	General Purpose I/O Register 1								25
0x29 (0x49)	Reserved	–	–	–	–	–	–	–	–	
0x28 (0x48)	OCR0B	Timer/Counter0 Output Compare Register B								
0x27 (0x47)	OCR0A	Timer/Counter0 Output Compare Register A								
0x26 (0x46)	TCNT0	Timer/Counter0 (8-bit)								
0x25 (0x45)	TCCR0B	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	
0x24 (0x44)	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	
0x23 (0x43)	GTCCR	TSM	–	–	–	–	–	PSRASY	PSRSYNC	144/166
0x22 (0x42)	EEARH	(EEPROM Address Register High Byte) <sup>5</sup>								21
0x21 (0x41)	EEARL	EEPROM Address Register Low Byte								21
0x20 (0x40)	EEDR	EEPROM Data Register								21
0x1F (0x3F)	EEDR	–	–	EEP1	EEP0	EERIE	EEMPE	EEPE	EERE	21
0x1E (0x3E)	GPIOR0	General Purpose I/O Register 0								25



# ATmega48A/48PA/88A/88PA/168A/168PA/328/328P

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
0x1D (0x3D)	EIMSK	–	–	–	–	–	–	INT1	INT0	73
0x1C (0x3C)	EIFR	–	–	–	–	–	–	INTF1	INTF0	73
0x1B (0x3B)	PCIFR	–	–	–	–	–	PCIF2	PCIF1	PCIF0	
0x1A (0x3A)	Reserved	–	–	–	–	–	–	–	–	
0x19 (0x39)	Reserved	–	–	–	–	–	–	–	–	
0x18 (0x38)	Reserved	–	–	–	–	–	–	–	–	
0x17 (0x37)	TIFR2	–	–	–	–	–	OCF2B	OCF2A	TOV2	164
0x16 (0x36)	TIFR1	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1	140
0x15 (0x35)	TIFR0	–	–	–	–	–	OCF0B	OCF0A	TOV0	
0x14 (0x34)	Reserved	–	–	–	–	–	–	–	–	
0x13 (0x33)	Reserved	–	–	–	–	–	–	–	–	
0x12 (0x32)	Reserved	–	–	–	–	–	–	–	–	
0x11 (0x31)	Reserved	–	–	–	–	–	–	–	–	
0x10 (0x30)	Reserved	–	–	–	–	–	–	–	–	
0x0F (0x2F)	Reserved	–	–	–	–	–	–	–	–	
0x0E (0x2E)	Reserved	–	–	–	–	–	–	–	–	
0x0D (0x2D)	Reserved	–	–	–	–	–	–	–	–	
0x0C (0x2C)	Reserved	–	–	–	–	–	–	–	–	
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	94
0x0A (0x2A)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	94
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	94
0x08 (0x28)	PORTC	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	93
0x07 (0x27)	DDRC	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	93
0x06 (0x26)	PINC	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	93
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	93
0x04 (0x24)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	93
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	93
0x02 (0x22)	Reserved	–	–	–	–	–	–	–	–	
0x01 (0x21)	Reserved	–	–	–	–	–	–	–	–	
0x0 (0x20)	Reserved	–	–	–	–	–	–	–	–	

- Note:
1. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.
  2. I/O Registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions.
  3. Some of the Status Flags are cleared by writing a logical one to them. Note that, unlike most other AVRs, the CBI and SBI instructions will only operate on the specified bit, and can therefore be used on registers containing such Status Flags. The CBI and SBI instructions work with registers 0x00 to 0x1F only.
  4. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these addresses. The ATmega48A/48PA/88A/88PA/168A/168PA/328/328P is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.
  5. Only valid for ATmega88A/88PA/168A/168PA/328/328P.
  6. BODS and BODSE only available for picoPower devices ATmega48PA/88PA/168PA/328P



## 5. Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clocks
<b>ARITHMETIC AND LOGIC INSTRUCTIONS</b>					
ADD	Rd, Rr	Add two Registers	$Rd \leftarrow Rd + Rr$	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry two Registers	$Rd \leftarrow Rd + Rr + C$	Z,C,N,V,H	1
ADIW	Rd,K	Add Immediate to Word	$Rdh:Rdl \leftarrow Rdh:Rdl + K$	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	$Rd \leftarrow Rd - Rr$	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	$Rd \leftarrow Rd - K$	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry two Registers	$Rd \leftarrow Rd - Rr - C$	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	$Rd \leftarrow Rd - K - C$	Z,C,N,V,H	1
SBIW	Rd,K	Subtract Immediate from Word	$Rdh:Rdl \leftarrow Rdh:Rdl - K$	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	$Rd \leftarrow Rd \bullet Rr$	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	$Rd \leftarrow Rd \bullet K$	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	$Rd \leftarrow Rd \vee Rr$	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	$Rd \leftarrow Rd \vee K$	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	$Rd \leftarrow Rd \oplus Rr$	Z,N,V	1
COM	Rd	One's Complement	$Rd \leftarrow 0xFF - Rd$	Z,C,N,V	1
NEG	Rd	Two's Complement	$Rd \leftarrow 0x00 - Rd$	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	$Rd \leftarrow Rd \vee K$	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	$Rd \leftarrow Rd \bullet (0xFF - K)$	Z,N,V	1
INC	Rd	Increment	$Rd \leftarrow Rd + 1$	Z,N,V	1
DEC	Rd	Decrement	$Rd \leftarrow Rd - 1$	Z,N,V	1
TST	Rd	Test for Zero or Minus	$Rd \leftarrow Rd \bullet Rd$	Z,N,V	1
CLR	Rd	Clear Register	$Rd \leftarrow Rd \oplus Rd$	Z,N,V	1
SER	Rd	Set Register	$Rd \leftarrow 0xFF$	None	1
MUL	Rd, Rr	Multiply Unsigned	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
MULS	Rd, Rr	Multiply Signed	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	$R1:R0 \leftarrow (Rd \times Rr) \lll 1$	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	$R1:R0 \leftarrow (Rd \times Rr) \lll 1$	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	$R1:R0 \leftarrow (Rd \times Rr) \lll 1$	Z,C	2
<b>BRANCH INSTRUCTIONS</b>					
RJMP	k	Relative Jump	$PC \leftarrow PC + k + 1$	None	2
IJMP		Indirect Jump to (Z)	$PC \leftarrow Z$	None	2
JMP <sup>(1)</sup>	k	Direct Jump	$PC \leftarrow k$	None	3
RCALL	k	Relative Subroutine Call	$PC \leftarrow PC + k + 1$	None	3
ICALL		Indirect Call to (Z)	$PC \leftarrow Z$	None	3
CALL <sup>(1)</sup>	k	Direct Subroutine Call	$PC \leftarrow k$	None	4
RET		Subroutine Return	$PC \leftarrow STACK$	None	4
RETI		Interrupt Return	$PC \leftarrow STACK$	I	4
CPSE	Rd,Rr	Compare, Skip if Equal	if $(Rd = Rr)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
CP	Rd,Rr	Compare	$Rd - Rr$	Z, N, V, C, H	1
CPC	Rd,Rr	Compare with Carry	$Rd - Rr - C$	Z, N, V, C, H	1
CPI	Rd,K	Compare Register with Immediate	$Rd - K$	Z, N, V, C, H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if $(Rr(b)=0)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
SBRS	Rr, b	Skip if Bit in Register is Set	if $(Rr(b)=1)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if $(P(b)=0)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if $(P(b)=1)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if $(SREG(s) = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if $(SREG(s) = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BREQ	k	Branch if Equal	if $(Z = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRNE	k	Branch if Not Equal	if $(Z = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRCS	k	Branch if Carry Set	if $(C = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRCC	k	Branch if Carry Cleared	if $(C = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRSH	k	Branch if Same or Higher	if $(C = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRLO	k	Branch if Lower	if $(C = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRMI	k	Branch if Minus	if $(N = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRPL	k	Branch if Plus	if $(N = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if $(N \oplus V = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if $(N \oplus V = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if $(H = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if $(H = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRTS	k	Branch if T Flag Set	if $(T = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRTC	k	Branch if T Flag Cleared	if $(T = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if $(V = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if $(V = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2

# ATmega48A/48PA/88A/88PA/168A/168PA/328/328P

Mnemonics	Operands	Description	Operation	Flags	#Clocks
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
<b>BIT AND BIT-TEST INSTRUCTIONS</b>					
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	Rotate Left Through Carry	Rd(0) ← C, Rd(n+1) ← Rd(n), C ← Rd(7)	Z,C,N,V	1
ROR	Rd	Rotate Right Through Carry	Rd(7) ← C, Rd(n) ← Rd(n+1), C ← Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0..6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3..0) ← Rd(7..4), Rd(7..4) ← Rd(3..0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Twos Complement Overflow.	V ← 1	V	1
CLV		Clear Twos Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half Carry Flag in SREG	H ← 1	H	1
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
<b>DATA TRANSFER INSTRUCTIONS</b>					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, -X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, -Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd, Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	-X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	-Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q, Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q, Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2



# ATmega48A/48PA/88A/88PA/168A/168PA/328/328P

Mnemonics	Operands	Description	Operation	Flags	#Clocks
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2
<b>MCU CONTROL INSTRUCTIONS</b>					
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	1
WDR		Watchdog Reset	(see specific descr. for WDR/timer)	None	1
BREAK		Break	For On-chip Debug Only	None	N/A

Note: 1. These instructions are only available in ATmega168PA and ATmega328P.

## 6. Ordering Information

### 6.1 ATmega48A

Speed (MHz)	Power Supply	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20 <sup>(3)</sup>	1.8 - 5.5	ATmega48A-AU ATmega48A-AUR <sup>(5)</sup> ATmega48A-MMH <sup>(4)</sup> ATmega48A-MMHR <sup>(4)(5)</sup> ATmega48A-MU ATmega48A-MUR <sup>(5)</sup> ATmega48A-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. See "Speed Grades" on page 321.
  4. NiPdAu Lead Finish.
  5. Tape & Reel.

Package Type	
<b>32A</b>	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
<b>28M1</b>	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>32M1-A</b>	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>28P3</b>	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)

## 6.2 ATmega48PA

Speed (MHz)	Power Supply	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20 <sup>(3)</sup>	1.8 - 5.5	ATmega48PA-AU ATmega48PA-AUR <sup>(5)</sup> ATmega48PA-MMH <sup>(4)</sup> ATmega48PA-MMHR <sup>(4)(5)</sup> ATmega48PA-MU ATmega48PA-MUR <sup>(5)</sup> ATmega48PA-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. See ["Speed Grades" on page 321](#).
  4. NiPdAu Lead Finish.
  5. Tape & Reel.

Package Type	
<b>32A</b>	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
<b>28M1</b>	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>32M1-A</b>	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>28P3</b>	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)

## 6.3 ATmega88A

Speed (MHz)	Power Supply	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20 <sup>(3)</sup>	1.8 - 5.5	ATmega88A-AU ATmega88A-AUR <sup>(5)</sup> ATmega88A-MMH <sup>(4)</sup> ATmega88A-MMHR <sup>(4)(5)</sup> ATmega88A-MU ATmega88A-MUR <sup>(5)</sup> ATmega88A-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. See ["Speed Grades" on page 321](#).
  4. NiPdAu Lead Finish.
  5. Tape & Reel.

Package Type	
<b>32A</b>	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
<b>28M1</b>	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>32M1-A</b>	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>28P3</b>	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)

## 6.4 ATmega88PA

Speed (MHz)	Power Supply (V)	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20 <sup>(3)</sup>	1.8 - 5.5	ATmega88PA-AU ATmega88PA-AUR <sup>(5)</sup> ATmega88PA-MMH <sup>(4)</sup> ATmega88PA-MMHR <sup>(4)(5)</sup> ATmega88PA-MU ATmega88PA-MUR <sup>(5)</sup> ATmega88PA-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. See "[Speed Grades](#)" on page 321.
  4. NiPdAu Lead Finish.
  5. Tape & Reel.

Package Type	
<b>32A</b>	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
<b>28M1</b>	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>32M1-A</b>	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>28P3</b>	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)

## 6.5 ATmega168A

Speed (MHz) <sup>(3)</sup>	Power Supply (V)	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20	1.8 - 5.5	ATmega168A-AU ATmega168A-AUR <sup>(5)</sup> ATmega168A-MMH <sup>(4)</sup> ATmega168A-MMHR <sup>(4)(5)</sup> ATmega168A-MU ATmega168A-MUR <sup>(5)</sup> ATmega168A-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. See ["Speed Grades" on page 321](#)
  4. NiPdAu Lead Finish.
  5. Tape & Reel.

Package Type	
<b>32A</b>	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
<b>28M1</b>	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>32M1-A</b>	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>28P3</b>	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)



## 6.6 ATmega168PA

Speed (MHz) <sup>(3)</sup>	Power Supply (V)	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20	1.8 - 5.5	ATmega168PA-AU ATmega168PA-AUR <sup>(5)</sup> ATmega168PA-MMH <sup>(4)</sup> ATmega168PA-MMHR <sup>(4)(5)</sup> ATmega168PA-MU ATmega168PA-MUR <sup>(5)</sup> ATmega168PA-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. See ["Speed Grades" on page 321](#).
  4. NiPdAu Lead Finish.
  5. Tape & Reel.

Package Type	
<b>32A</b>	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
<b>28M1</b>	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>32M1-A</b>	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
<b>28P3</b>	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)

## 6.7 ATmega328

Speed (MHz)	Power Supply (V)	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20 <sup>(3)</sup>	1.8 - 5.5	ATmega328-AU ATmega328-AUR <sup>(4)</sup> ATmega328-MU ATmega328-MUR <sup>(4)</sup> ATmega328-PU	32A 32A 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. See [Figure 28-1 on page 321](#).
  4. Tape & Reel

Package Type	
<b>32A</b>	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
<b>28P3</b>	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
<b>32M1-A</b>	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)



## 6.8 ATmega328P

Speed (MHz)	Power Supply	Ordering Code <sup>(2)</sup>	Package <sup>(1)</sup>	Operational Range
20 <sup>(3)</sup>	1.8 - 5.5	ATmega328P-AU ATmega328P-AUR <sup>(4)</sup> ATmega328P-MU ATmega328P-MUR <sup>(4)</sup> ATmega328P-PU	32A 32A 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

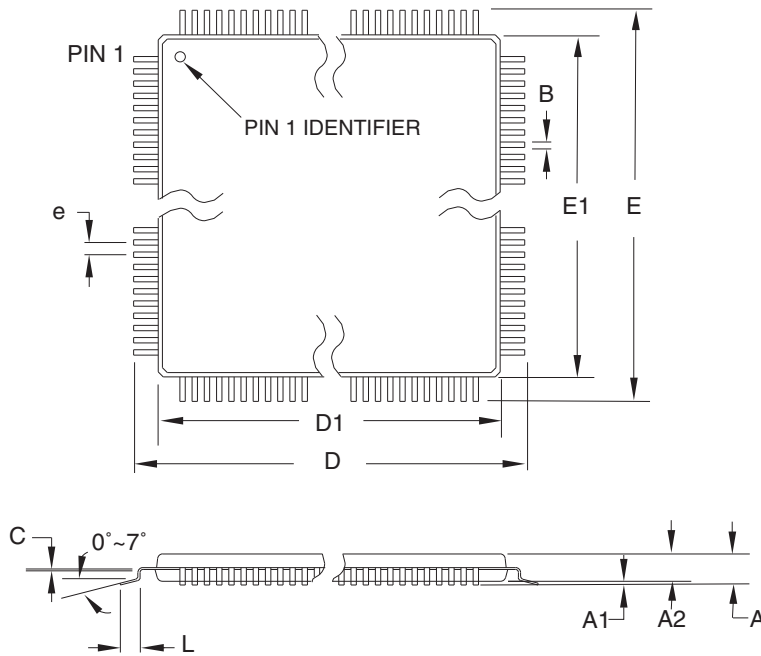
- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. See [Figure 28-1 on page 321](#).
  4. Tape & Reel.

Package Type	
<b>32A</b>	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
<b>28P3</b>	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
<b>32M1-A</b>	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)



## 7. Packaging Information

### 7.1 32A



**COMMON DIMENSIONS**  
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	8.75	9.00	9.25	
D1	6.90	7.00	7.10	Note 2
E	8.75	9.00	9.25	
E1	6.90	7.00	7.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

- Notes:
1. This package conforms to JEDEC reference MS-026, Variation ABA.
  2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
  3. Lead coplanarity is 0.10 mm maximum.

10/5/2001



2325 Orchard Parkway  
San Jose, CA 95131

**TITLE**

**32A**, 32-lead, 7 x 7 mm Body Size, 1.0 mm Body Thickness,  
0.8 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)

**DRAWING NO.**

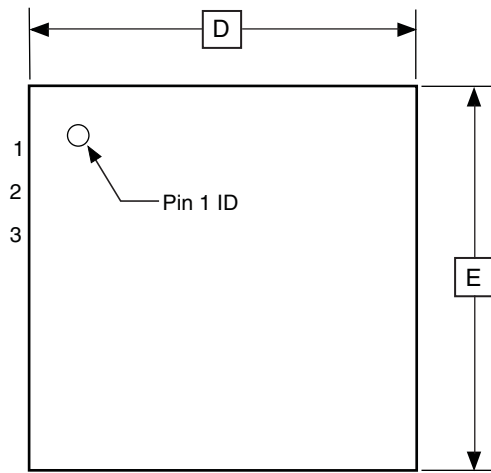
32A

**REV.**

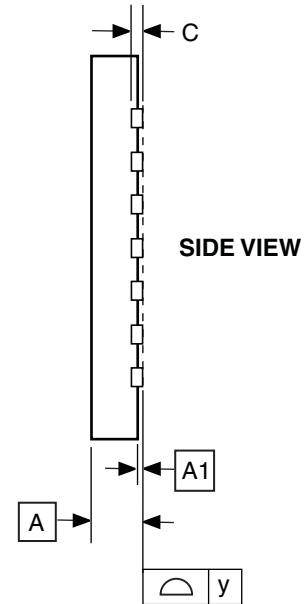
B



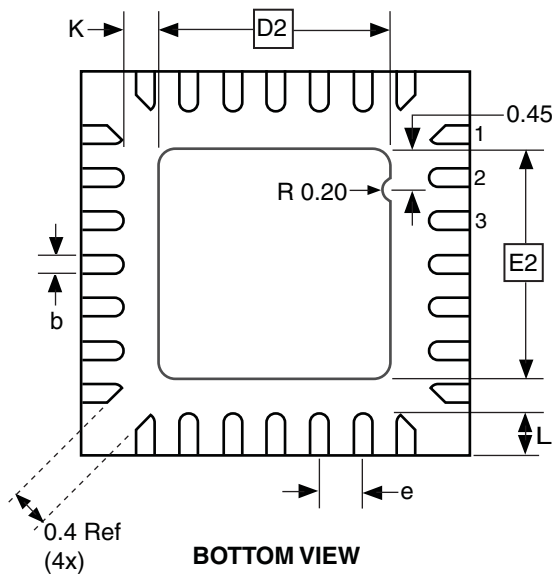
## 7.2 28M1



TOP VIEW



SIDE VIEW



BOTTOM VIEW

COMMON DIMENSIONS  
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.90	1.00	
A1	0.00	0.02	0.05	
b	0.17	0.22	0.27	
C	0.20 REF			
D	3.95	4.00	4.05	
D2	2.35	2.40	2.45	
E	3.95	4.00	4.05	
E2	2.35	2.40	2.45	
e	0.45			
L	0.35	0.40	0.45	
y	0.00	-	0.08	
K	0.20	-	-	

Note: The terminal #1 ID is a Laser-marked Feature.

10/24/08



Package Drawing Contact:  
packagedrawings@atmel.com

**TITLE**  
28M1, 28-pad, 4 x 4 x 1.0 mm Body, Lead Pitch 0.45 mm,  
2.4 x 2.4 mm Exposed Pad, Thermally Enhanced  
Plastic Very Thin Quad Flat No Lead Package (VQFN)

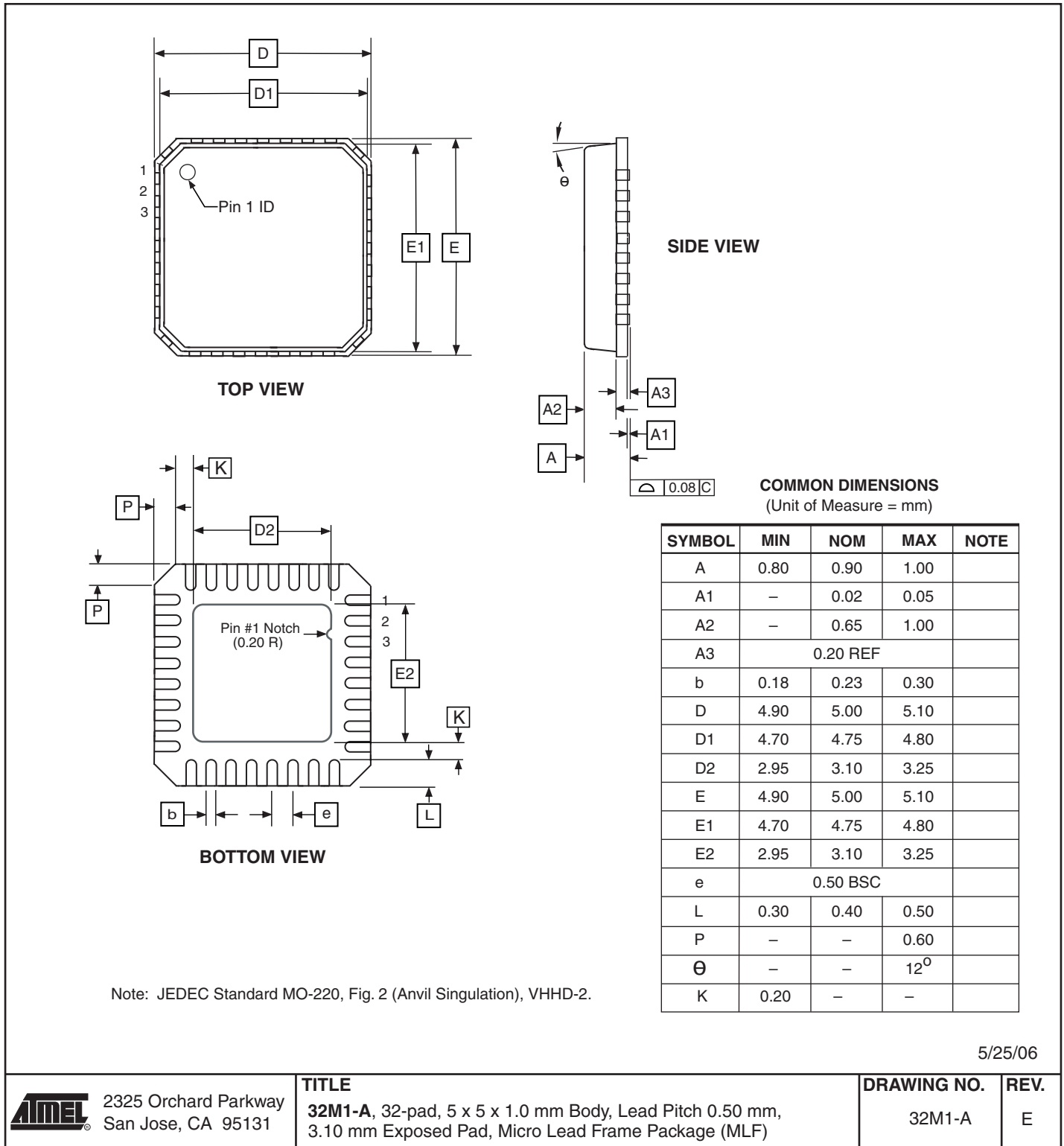
**GPC**  
ZBV

**DRAWING NO.**  
28M1

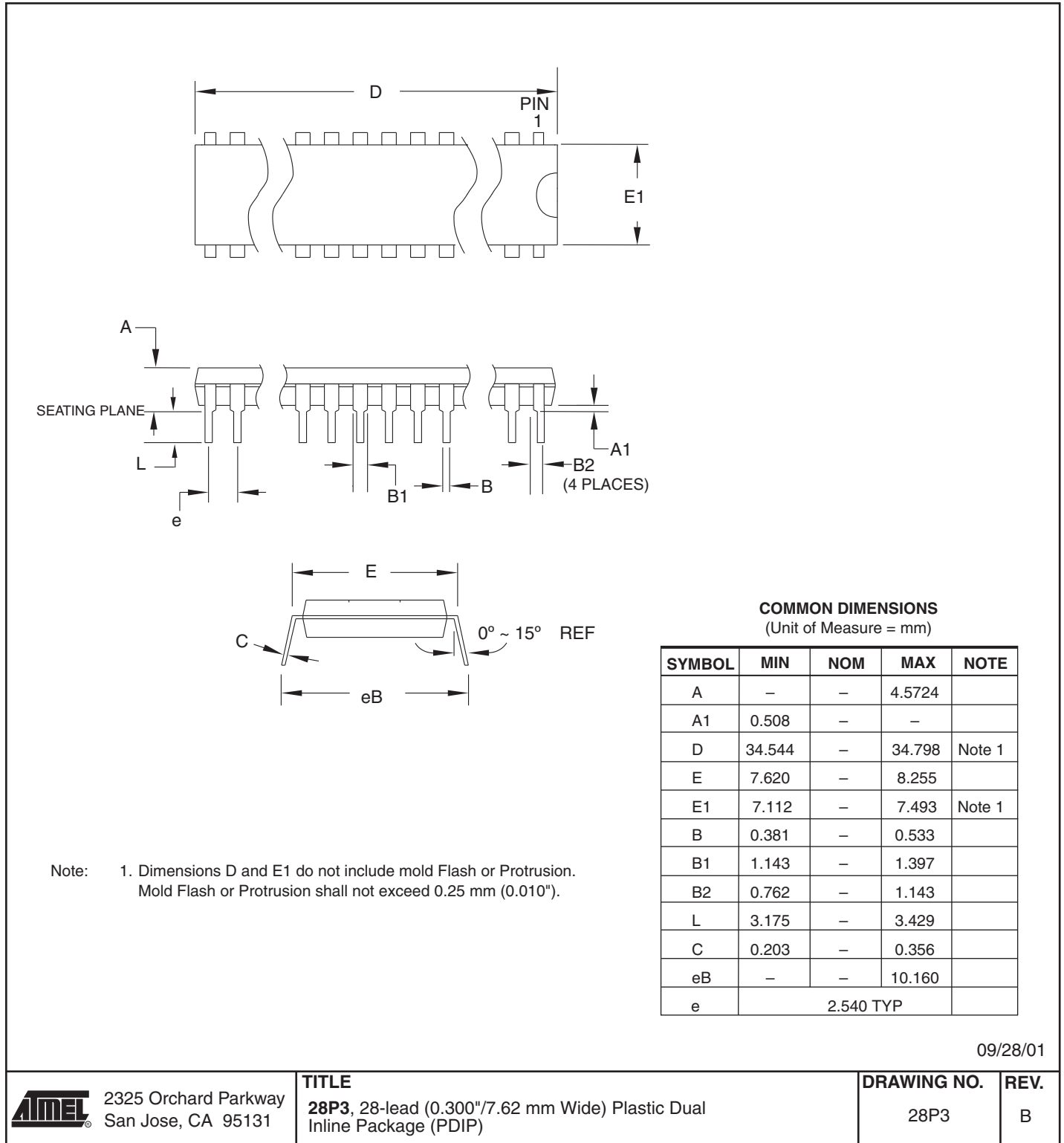
**REV.**  
B



## 7.3 32M1-A



## 7.4 28P3



## 8. Errata

### 8.1 Errata ATmega48A

The revision letter in this section refers to the revision of the ATmega48A device.

#### 8.1.1 Rev. D

- **Analog MUX can be turned off when setting ACME bit**

1. **Analog MUX can be turned off when setting ACME bit**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSR is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

**Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

### 8.2 Errata ATmega48PA

The revision letter in this section refers to the revision of the ATmega48PA device.

#### 8.2.1 Rev. D

- **Analog MUX can be turned off when setting ACME bit**

1. **Analog MUX can be turned off when setting ACME bit**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSR is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

**Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

### 8.3 Errata ATmega88A

The revision letter in this section refers to the revision of the ATmega88A device.

#### 8.3.1 Rev. F

- **Analog MUX can be turned off when setting ACME bit**

1. **Analog MUX can be turned off when setting ACME bit**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSR is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

**Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.



## 8.4 Errata ATmega88PA

The revision letter in this section refers to the revision of the ATmega88PA device.

### 8.4.1 Rev. F

- **Analog MUX can be turned off when setting ACME bit**

1. **Analog MUX can be turned off when setting ACME bit**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSRB is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

**Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

## 8.5 Errata ATmega168A

The revision letter in this section refers to the revision of the ATmega168A device.

### 8.5.1 Rev. E

- **Analog MUX can be turned off when setting ACME bit**

1. **Analog MUX can be turned off when setting ACME bit**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSRB is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

**Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

## 8.6 Errata ATmega168PA

The revision letter in this section refers to the revision of the ATmega168PA device.

### 8.6.1 Rev E

- **Analog MUX can be turned off when setting ACME bit**

1. **Analog MUX can be turned off when setting ACME bit**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSRB is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

**Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

## 8.7 Errata ATmega328

The revision letter in this section refers to the revision of the ATmega328 device.

### 8.7.1 Rev D

- **Analog MUX can be turned off when setting ACME bit**

#### 1. **Analog MUX can be turned off when setting ACME bit**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSRB is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

#### **Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

### 8.7.2 Rev C

Not sampled.

### 8.7.3 Rev B

- **Analog MUX can be turned off when setting ACME bit**
- **Unstable 32 kHz Oscillator**

#### 1. **Unstable 32 kHz Oscillator**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSRB is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

#### **Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

#### 2. **Unstable 32 kHz Oscillator**

The 32 kHz oscillator does not work as system clock. The 32 kHz oscillator used as asynchronous timer is inaccurate.

#### **Problem Fix/ Workaround**

None.

### 8.7.4 Rev A

- **Analog MUX can be turned off when setting ACME bit**
- **Unstable 32 kHz Oscillator**

#### 1. **Unstable 32 kHz Oscillator**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSRB is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

#### **Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

#### 2. **Unstable 32 kHz Oscillator**

The 32 kHz oscillator does not work as system clock. The 32 kHz oscillator used as asynchronous timer is inaccurate.

#### **Problem Fix/ Workaround**

None.

## 8.8 Errata ATmega328P

The revision letter in this section refers to the revision of the ATmega328P device.

### 8.8.1 Rev D

- **Analog MUX can be turned off when setting ACME bit**

#### 1. **Analog MUX can be turned off when setting ACME bit**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSRB is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

#### **Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

### 8.8.2 Rev C

Not sampled.

### 8.8.3 Rev B

- **Analog MUX can be turned off when setting ACME bit**
- **Unstable 32 kHz Oscillator**

#### 1. **Unstable 32 kHz Oscillator**

If the ACME (Analog Comparator Multiplexer Enabled) bit in ADCSRB is set while MUX3 in ADMUX is '1' (ADMUX[3:0]=1xxx), all MUX'es are turned off until the ACME bit is cleared.

#### **Problem Fix/Workaround**

Clear the MUX3 bit before setting the ACME bit.

#### 2. **Unstable 32 kHz Oscillator**

The 32 kHz oscillator does not work as system clock. The 32 kHz oscillator used as asynchronous timer is inaccurate.

#### **Problem Fix/ Workaround**

None.

### 8.8.4 Rev A

- **Unstable 32 kHz Oscillator**

#### 1. **Unstable 32 kHz Oscillator**

The 32 kHz oscillator does not work as system clock. The 32 kHz oscillator used as asynchronous timer is inaccurate.

#### **Problem Fix/ Workaround**

None.

## 9. Datasheet Revision History

Please note that the referring page numbers in this section are referred to this document. The referring revision in this section are referring to the document revision.

### 9.1 Rev. 8271B-04/10

1. Updated [Table 8-8](#) with correct value for timer oscillator at xtal2/tos2
2. Corrected use of SBIS instructions in assembly code examples.
3. Corrected BOD and BODSE bits to R/W in [Section 9.11.2 on page 45](#), [Section 11.5 on page 69](#) and [Section 13.4 on page 93](#)
4. Figures for bandgap characterization added, [Figure 29-34 on page 349](#), [Figure 29-81 on page 374](#), [Figure 29-128 on page 399](#), [Figure 29-175 on page 424](#), [Figure 29-222 on page 449](#), [Figure 29-269 on page 474](#), [Figure 29-316 on page 499](#) and [Figure 29-363 on page 523](#).
5. Updated ["Packaging Information" on page 546](#) by replacing 28M1 with a correct corresponding package.

### 9.2 Rev. 8271A-12/09

1. New datasheet 8271 with merged information for ATmega48PA, ATmega88PA, ATmega168PA and ATmega48A, ATmega88A and ATmega168A. Also included information on ATmega328 and ATmega328P
2. Changes done:
  - New devices added: ATmega48A/ATmega88A/ATmega168A and ATmega328
  - Updated Feature Description
  - Updated [Table 2-1 on page 6](#)
  - Added note for BOD Disable on [page 40](#).
  - Added note on BOD and BODSE in ["MCUCR – MCU Control Register" on page 93](#) and ["Register Description" on page 294](#)
  - Added limitation information for the application ["Boot Loader Support – Read-While-Write Self-Programming" on page 279](#)
  - Added limitation information for ["Program And Data Memory Lock Bits" on page 296](#)
  - Added specified DC characteristics per processor
  - Added typical characteristics per processor
  - Removed exception information in ["Address Match Unit" on page 223](#).



## Headquarters

---

**Atmel Corporation**  
2325 Orchard Parkway  
San Jose, CA 95131  
USA  
Tel: 1(408) 441-0311  
Fax: 1(408) 487-2600

## International

---

**Atmel Asia**  
Unit 1-5 & 16, 19/F  
BEA Tower, Millennium City 5  
418 Kwun Tong Road  
Kwun Tong, Kowloon  
Hong Kong  
Tel: (852) 2245-6100  
Fax: (852) 2722-1369

**Atmel Europe**  
Le Krebs  
8, Rue Jean-Pierre Timbaud  
BP 309  
78054 Saint-Quentin-en-  
Yvelines Cedex  
France  
Tel: (33) 1-30-60-70-00  
Fax: (33) 1-30-60-71-11

**Atmel Japan**  
9F, Tonetsu Shinkawa Bldg.  
1-24-8 Shinkawa  
Chuo-ku, Tokyo 104-0033  
Japan  
Tel: (81) 3-3523-3551  
Fax: (81) 3-3523-7581

## Product Contact

---

**Web Site**  
[www.atmel.com](http://www.atmel.com)

**Technical Support**  
[avr@atmel.com](mailto:avr@atmel.com)

**Sales Contact**  
[www.atmel.com/contacts](http://www.atmel.com/contacts)

**Literature Requests**  
[www.atmel.com/literature](http://www.atmel.com/literature)

---

**Disclaimer:** The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. **EXCEPT AS SET FORTH IN ATMEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATMEL'S WEB SITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.** Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel's products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

© 2010 Atmel Corporation. All rights reserved. Atmel®, Atmel logo and combinations thereof, AVR®, AVR® logo and others are registered trademarks or trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others.