CENTRO UNIVERSITÁRIO UNIFACVEST

CURSO DE ENGENHARIA ELÉTRICA

RUDSON JOÃO BITENCOURTE

APLICAÇÃO DA ROBÓTICA AUXILIANDO DEFICIENTES VISUAIS

LAGES
2018

RUDSON JOÃO BITENCOURTE

APLICAÇÃO DA ROBÓTICA AUXILIANDO DEFICIENTES VISUAIS

Trabalho de conclusão de curso apresentado ao Centro Universitário UNIFACVEST como parte dos requisitos para obtenção do grau de Bacharel em Engenharia Elétrica.

Orientador: Prof. Eng. Msc. Silvio Moraes de Oliveira

Coorientador: Prof. Msc. João Francisco Frank Gil

LAGES
2018

Monografia apresentada ao Centro Universitário Facvest – UNIFACVEST, como requisito necessário para a obtenção do título de Bacharel em Engenharia Elétrica.

Rudson João Bitencourte
NOME DO ALUNO

Aplicação da Robótica Auxiliando Deficientes Visuais
TÍTULO DO TRABALHO

BANCA EXAMINADORA:

Msc. Silvio Moraes de Oliveira
Titulação e nome do Orientador(a)

Franciéli Lima de Jesus, Dra.
Titulação e nome do Avaliador (a).

Titulação e nome do Avaliador (a).

Franciéli Lima de Jesus, Dra.
Coordenador (a) Prof. (a). Titulação e nome da Coordenador (a).

Lages, 13 de dezembro de 2018.

DEDICATÓRIA

Dedico este trabalho a todas as pessoas que participaram direta e indiretamente da minha formação acadêmica. Em especial dedico a Deus por ter me proporcionado a vida, a minha mãe e irmãos, acreditarem na minha capacidade, minha noiva Isabella pela compreensão e auxílio que recebi. Ao professor orientador Silvio Moraes de Oliveira e o professor coorientador João Francisco Frank Gil e a professora Nathielle Waldrigues Branco por acreditar e me auxiliar nessa jornada.

"Lute com determinação, abrace a vida com paixão, perca com classe e vença com ousadia, porque o mundo pertence a quem se atreve e a vida é muito para ser insignificante ".
Augusto Branco

AGRADECIMENTOS

Primeiramente agradeço a Deus pela saúde e por ter chegado até nesse momento tão importante na minha vida. Agradeço a minha mãe Ida pelo amor, carinho e confiança depositada em minha capacidade. A minha noiva Isabella pelo apoio, compreensão, atenção e auxílio em todos os momentos difíceis desta caminhada.

Aos meus irmãos Ramon e Robson, pelo apoio, amizade e momentos que dividimos juntos, meu muito obrigado. Aos meus colegas de faculdade que estiveram ao meu lado nesta longa jornada contribuindo com muitas informações para o desenvolvimento de projeto, agradeço pelo apoio, convivência e amizade nesses cinco anos.

Ao professor orientador Silvio Moraes de Oliveira, professor coorientador João Francisco Frank Gil e a professora Nathielle Waldrigues Branco pelo apoio, por acreditar e incentivar a realização deste trabalho, meus sinceros agradecimentos. A todas as pessoas que contribuíram diretamente e indiretamente para a realização deste trabalho, meu muito obrigado.

LISTA DE FIGURA

LISTA DE QUADRO

LISTA DE SIGLAS E ABREVIATURAS

2WD – Modelo de chassi

Ah – Ampere hora, unidade de medida de corrente elétrica por hora

C / C ++ - Linguagem de programação

CC – Corrente continua

cm – Centímetro, unidade de medida

GPS - *Global Positioning System*

IDE – Ambiente de desenvolvimento de programação

IN – Pino de entrada da ponte H

K – Kilo, equivale 1000 vezes, na unidade de base 10

KHz – Kilo Hertz, unidade de frequência

MA – Pino de controle de velocidade do motor A

MB – Pino de controle de velocidade do motor B

ONG – Organização não governamental

PWM - *Pulse Width Modulation*

USB - *Universal Serial Bus*

V - Volts, unidade de medida de tensão elétrica

RESUMO

Este trabalho tem como objetivo incentivar as pessoas com deficiência visual a obter uma maior segurança em sua locomoção, reduzir o índice de colisões contra obstáculos e aumentar a autonomia de se movimentar. Por meio do desenvolvimento de um robô autônomo, na área da engenharia, foram utilizado a placa arduino *ATMEGA 2560* e sensores ultrassônicos para detecção dos obstáculos frontal, desnível e aéreo. Esses sensores são programados para detectar os obstáculos, sendo capaz de calcular a distância entre o obstáculo e o deficiente visual, avisando-o sobre a sua proximidade. Caso a distância for menor que a distância programada do robô, é emitido um aviso ao usuário, que vai ser transmitido via cabo para o controle de mão, este alerta é executado através de motores *vibracall* (motores que fazem vibrar o ponto de sensibilização do usuário). Os sensores aéreo e frontal estão fixados em um servo motor onde é realizado o rastreamento em um ângulo de 0° à 180° pra detecção de obstáculos. Portanto com a execução do protótipo, pretende-se diminuir as colisões dos deficientes visuais, para que eles possam ter uma vida mais acessível e sem riscos de acidentes.

Palavras-Chave: Deficiência Visual, Rastreamento, Colisão contra Obstáculos.

ABSTRACT

This work aims to encourage people with visual disabilities to obtain greater security in your mobility aid, reduce the rate of collisions against obstacles and increase the autonomy of movement. Through the development of a robot as, in the area of engineering, were used the arduino Board ATMEGA 2560 and Ultrasonic sensors for detecting obstacles front, and air gap. These sensors are programmed to detect obstacles, being able to calculate the distance between the obstacle and the visually impaired, warning him about your proximity. If the distance is less than the distance programmed robot, is issued a warning to the user, which will be broadcast via cable to the hand control, this alert runs through vibracall engines (engines that make vibrate the point user awareness). Air sensors and front are set on a servo motor where the trace to an angle of 0° to 180° for obstacle detection. So by running the prototype is designed to lessen collisions of the visually impaired, so that they can have a life more accessible and without risk of accidents.

Key - words: Visual impairment, Tracing, Collision against Obstacles.

SUMÁRIO

1 INTRODUÇÃO

Monteiro, Silva e Lopes (2014), afirmam que no cotidiano podemos reparar-se com os deficientes visuais, que são desafiados a encararem as circunstâncias do acesso ao mundo, ações rotineiras, como uma caminhada ou corrida torna-se difícil.

De acordo com Nunes e Lomonaco (2010), a cegueira é uma deficiência visual, no qual restringi a percepção de conhecimentos do mundo visível - a visão. Portanto ocorre duas formas de deficiência visual: a cegueira e a baixa visão.

Através dos dados obtidos por Amirialian (1997), no início de 1970, o conhecimento sobre a deficiência visual passou a ser conceituada como acuidade visual para verificar as formas de entendimento do indivíduo: se o sujeito compreende o universo através do toque, cheiro, percepção de movimentos, peso, resistência e estímulos, etc., deste modo o indivíduo é tido como cego, entretanto se tiver insuficiência visual, mas se mesmo assim houver utilização do remanescente visual de modo suficiente, então, é classificado como uma pessoa com baixa visão.

Desta forma possibilita a utilização de meios ópticos (lupas de aumento, óculos binoculares e monoculares, lentes de contato para obter uma melhor condição de visão, etc.); podendo se utilizar medidas de apoio (com a utilização do sistema de escrita tátil, ou seja, o braile ou letra comum). Em situações de baixa visão, auxílios ópticos podem ser empregados para aumentar a condição visual. Já com o sujeito cego, não ocorre a situação descrita, pois é necessário fazer com que o conhecimento visual chegue até o indivíduo por outras maneiras. Para isto, outros meios sensoriais podem ser utilizados, como o toque com as mãos e a escuta.

Dentre os obstáculos de deslocamento para os deficientes visuais, podemos citar alguns exemplos como: O desnível das calçadas e ruas : O impedimentos de acesso como arvores, bancos de praça, pessoas, postes, e/ou outros itens que impeçam a sua movimentação; A falta de entendimento e consciência dos indivíduos que rotineiramente estacionam seus veículos em calçadas ; Em locais fechados que não se obtêm conhecimento da localização dos moveis, objetos e paredes ; Em locais que transitam muitas pessoas, é difícil prever uma colisão.

> No Brasil, dentre os tipos de deficiências investigadas pela PNS 2013, a deficiência visual foi a mais representativa na população, com proporção de 3,6%. Esse indicador foi mais elevado na Região Sul (5,9%). As pessoas de 60 anos ou mais de idade apresentaram proporção superior (11,5%) à observada nos demais grupos de idade. No País, 6,6% das pessoas com deficiência visual faziam uso de algum recurso para auxiliar a locomoção, como bengala articulada ou cão-guia. Todas as Grandes Regiões obtiveram estimativas assemelhadas estatisticamente à observada para o Brasil. (IBGE,2015, p.28).

A delimitação do deficiente visual não acontece somente devido à ausência da visão, mas sim pela falta de possibilidade para conhecer diversas situações. Há décadas, diversas pesquisas incluíram novas tecnologias que possibilitaram favorecer a vida dos deficientes visuais, visando a locomoção, proteção, conhecimento através da educação, entre outros recursos. A robótica, mecânica, eletrônica/ elétrica e a ciência da computação estão cada vez mais frequentes na vida dos cegos para ajudar em objetivos terapêuticos ou de aprendizagem, tanto no progresso como na reabilitação Burlamaqui et al. (2017)

Bersch (2008) relata em sua obra que no Brasil e em diversos países, existem várias formas de tecnologia assistiva, no qual vêm sendo expandida por várias empresas e institutos de pesquisa com o intuito de ajudar essas pessoas na execução de suas funções do dia- dia. Dentre as inovações mais empregadas atualmente, localizamos dispositivos de reprodução em braile, aumentadores e leitores de tela de computadores, bengalas que identificam obstáculos, passarelas com caminhos exclusivos para a locomoção de deficientes visuais em passeios ou ruas/avenidas e aparelhos de controle de ambiente, no qual o foco principal destas tecnologias é garantir aos cegos a mobilidade e o acesso.

Através das soluções tecnológicos existentes na área da engenharia elétrica/ eletrônica, mecânica, automação e computação, sugere-se desenvolver um sistema autônomo, no qual terá atividade de detectar obstáculos, através de sensores ultrassônicos, no qual auxiliara os deficientes visuais em sua locomoção e acessibilidade, tendo o intuito de minimizar a colisão, acidentes desses indivíduos. Por meio deste robô, os deficientes visuais serão capazes de se movimentar com melhor eficiência. Deste modo, este trabalho pretende demonstrar a construção de um protótipo para auxílio dos deficientes visuais.

1.1 Justificativa

Estudar a robótica e as dificuldades enfrentadas pelos deficientes visuais, é importante para que se possa compreender as suas principais dificuldades e limitações em relação a sua locomoção, desde as tarefas mais simples do dia a dia como caminhar, atravessar a rua, desviar de obstáculos, como galhos de árvores, degraus, irregularidades nas ruas e calçadas, entre outros.

Esta pesquisa identificará os principais obstáculos enfrentados pelos deficientes visuais e os meios auxiliares mais utilizados, tendo como intuito o desenvolvimento de um robô autônomo que tem como principal função, detectar os obstáculos enfrentados por esse grupo de pessoas.

Para a comunidade científica esta pesquisa será inovadora, pois este estudo propõe a acessibilidade dos deficientes visuais através da utilização da robótica, em realizar as suas atividades rotineiras.

Os deficientes visuais podem usufruir de diversos sistemas de locomoção, no qual interagem e auxiliam no seu dia a dia, os meios mais utilizados são a bengala, o cão guia e alguns sistemas de reconhecimento de voz.

Porém estes meios requerem um alto custo e exigem muito tempo de treinamento, como no caso do cão guia, alguns métodos também não são muito eficientes e seu auxilio é bem limitado. Sendo que na prática muitos desses deficientes visuais não possuem condições financeiras de adquirir meios de locomoção mais eficientes.

Portanto o presente trabalho, além dos reforços acadêmicos e científicos gerados em função das pesquisas que foram realizadas, tem importante relevância, pois cria um dispositivo (protótipo) funcional e prático, no qual se mostra viável por possuir um baixo custo, sendo seu treinamento mais facilitado e seu método de locomoção mais eficiente.

Além disso este estudo tem grande importância para a faculdade pois é uma aplicação tecnológica na área da robótica, sendo de grande relevância para o curso e para os alunos, uma vez que terei a oportunidade de expor esta pesquisa aos demais membros da comunidade universitária no qual estaremos cada vez mais conectados com os assuntos atuais da engenharia.

1.2 Objetivos

1.2.1 Objetivo geral

Apresentar o desenvolvimento e a construção de um protótipo de um robô autônomo para auxílio dos deficientes visuais.

1.2.2 Objetivo específico

Analisar os principais obstáculos enfrentados pelos deficientes visuais;
Comparar o protótipo com outros meios auxiliares, assim como projetar a estrutura do robô;
Identificar as principais vantagens de utilizar a robótica no dia a dia;
Identificar a precisão dos sensores e controle de mão;
Analisar a interação de homem e máquina;
Realizar os testes (Demonstração da movimentação do protótipo);

1.3 Metodologia

O presente estudo foi desenvolvido com base em pesquisa exploratória, culminando no desenvolvimento de um protótipo, onde houve aplicação de uma das metodologias estudadas.

A primeira etapa realizada no desenvolvimento do protótipo, foi analisar as dificuldades dos deficientes visuais, através de pesquisas em artigos, dissertações e outros projetos na mesma área de aplicação, para ter um melhor entendimento das necessidades que eles teriam no seu cotidiano, e ao mesmo momento, visualizar como o robô autônomo iria auxiliar nestas necessidades.

A principal ideia é que o robô seja um auxílio para o usuário com mais eficiência que o próprio cão e até mesmo a bengala. Para isso, o protótipo contém dois motores de corrente continua individual em cada roda, três sensores ultrassônicos para detecção de obstáculos acima, a frente e ao chão. O usuário é informado sobre obstáculos, via cabo, em um controle de mão, que tem motores *vibracall*, conforme as respostas dos sensores ultrassônicos, permitindo ao usuário identificar onde estará o obstáculo.

Foram utilizados: um servo motor para monitoramento do ângulo de 0° a 180°; duas (2) baterias Duracell 9 volts para alimentação do robô; uma ponte H do modelo L298N utilizada para inversão de giro dos motores CC’s; *jumpers* para realizar a comunicação elétrica dos

componentes; e um chassi acrílico. Através do chassi, foi possível fixar todos os componentes conforme o projeto desenvolvido no *Solid Works*. Foi elaborado um *driver* para efetuar o acionamento dos motores *vibracall*, devido a limitação de tensão de alimentação dos motores.

Foi projetada uma caixa em impressora 3D, com intuito de armazenar o driver e os motores *vibracall*.

A linguagem utilizada foi C / C++ dentro do ambiente de programação plataforma *IDE* arduino para compilação do programa no qual enviara para o microcontrolador ATMega 2560.

O presente trabalho está dividido em 6 capítulos, sendo que, além deste capitulo introdutório, temos:

- No Capítulo 2, foram apresentados alguns conceitos, sobre a robótica, sua origem e classificação, em seguida foi exposto o conceito de deficiência visual, a aplicação da robótica para os deficientes visuais;

- No Capítulo 3, foi realizada uma discussão sobre os algoritmos, com explicação do princípio de programação, no qual foram apresentados a linguagem C / C ++ e o software que foi utilizado;

- O Capítulo 4, é dedicado à apresentação dos componentes do trabalho em si, sendo descrito de forma detalhada, os seguintes itens: Placa arduino mega, Ponte H L298N, sensores ultrassônicos, chassi, servo motor, cabos, pilhas/baterias e controle de mão;

- No Capítulo 5, foi apresentado todo o sistema proposto, assim como a visão da programação do robô que será detalhado em forma de fluxograma, em seguida será descrito a montagem dos componentes no chassi, será demonstrado através de ilustrações os princípios de funcionamento do robô.

- No Capítulo 6, foram apresentadas as considerações finais referentes a pesquisa e algumas expectativas para futuros projetos.

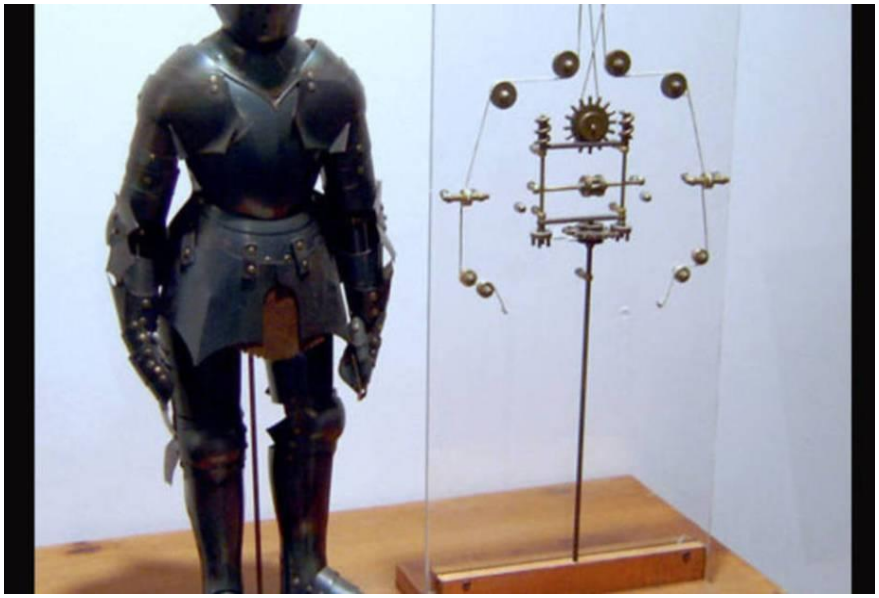
2 FUNDAMENTAÇÃO TEÓRICA E APLICAÇÕES DA ROBÓTICA

2.1 Origem da robótica

Segundo Ribeiro (2005) a expressão robô está no nosso dia a dia, mas é válido nos perguntamos o que é um robô e quais são as suas primícias. No início os homens sonhavam com implantação de máquinas que os auxiliasse em suas atividades. Um dos primeiros robôs foi o relógio de água, inventado no ano 250 a.c. por Ctesibius de Alexandria, um matemático, engenheiro e inventor grego.

Foresti (2006) relata em sua obra que em 1495 que Leonardo Da Vinci projetou cavaleiro mecânico onde era capaz de fazer os movimentos de sentar-se, mexer seus braços, mover sua cabeça e o maxilar. Com essa ideia de Leonardo Da Vinci foi criado o primeiro robô humanoide da história. Portanto muitos robôs surgiam na época mas a maioria serviria apenas como inspiração eram apenas obras de ficção e bem poucos desses robôs podiam ser construído.

Figura 1 - Robô humanoide criado por Leonardo Da Vinci



Fonte: Revista Exame (2012).

Leonardo da Vinci, um dos mais importantes pintores do renascimento, proporcionou grandes competências na execução e construção de máquinas, onde estão em diversos acontecimentos está presente na atualidade. Leonardo da Vinci deixou esboços de estruturas mecânicas humanoides, de máquinas voadoras, recursos de propulsão de helicópteros, de máquinas engenhosas para sondar canais em rios (RIBEIRO, 2005).

2.2 Classificação dos robôs

2.3 Robôs na medicina

A aplicação da robótica na medicina é algo bem recente, mas vem se mostrando uma aplicação bem promissora, começando a se tornar algo indispensável. São mais utilizados em cirurgias porque se obtém mais velocidade, precisão, repetição, confiabilidade. Por exemplo, um braço robótico que sustenta um endoscópio com uma câmera durante uma cirurgia não sofre nenhuma fadiga, não importa quanto tempo vai durar a cirurgia, não apresentará tremores e realizará o seu trabalho adequadamente (LISBOA, 2010).

O uso de robôs não está restrito apenas a isso. Sua eficiência tem sido demonstrada em outras áreas como na distribuição de remédios dentro de hospitais, robôs para reabilitação, para simulação de traumas e cirurgias, etc.

Figura 2 - Robôs Sendo Aplicados na Medicina

Fonte: Lisboa (2010)

2.4 Robótica móvel

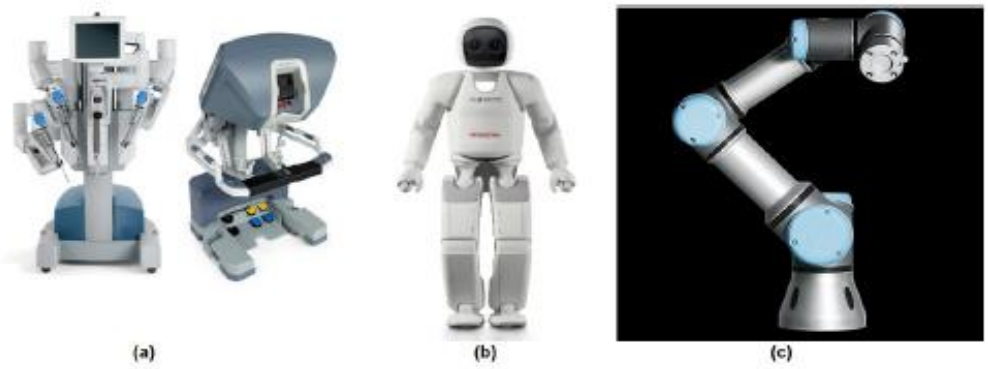
Atualmente a robótica chega com grandes avanços que fazem com que ela se apresente como algo cotidiano nas vidas dessa geração. Na época atual, os robôs são utilizados em infinitas situações que abrangem desde a indústria até a área médica. A sua aplicação se dá pela segurança e qualidade oferecidos por essa tecnologia. A Figura 3 ilustra alguns desses robôs e suas áreas de atuação.

Com inclusão dos robôs ao setor industrial, segundo Secchi (2008), sempre ofereceu um excelente compromisso entre produtividade e flexibilidade, uma qualidade uniforme dos produtos, uma sistematização dos processos e critérios. Dessa maneira, destacam-se o aumento

da produtividade, alta flexibilidade, qualidade e melhoria na segurança como consequência da robotização.

As últimas décadas houve um avanço muito significativo na área de robótica, muito em função dos novos recursos de *hardware* e *software* desenvolvidos. Em conteúdo de *hardware* os computadores e dispositivos embarcados vêm sendo cada vez mais minúsculos, tendo seus custos reduzidos e sua capacidade de processamento bem mais alta. Ainda, eles estão sendo mais robustos e precisos, consumindo menos energia e alcançando maior autonomia (ROMERO, et al 2014).

Figura 3 - Robôs modernos existentes no mercado atual (a) Robô cirurgião Da Vinci; (b) Robô Asimo da Honda; (c) Braço robótico UR3 da empresa Universal Robots.



Fonte: Umezaki (2017)

De acordo com Romero et al (2014), a criação de ***softwares*** se reflete no desenvolvimento de programas de controle capazes de auxiliar na tomada de decisão, processamento de imagens, reconhecimento de voz e muitas outras funções que aliadas à evolução dos atuadores, possibilitam a sofisticação e capacidades dos robôs.

2.5 Robôs industriais

A robótica é uma área da tecnologia que engloba elétrica/eletrônica, mecânica e computação, que atualmente trata de sistemas compostos por máquinas e partes mecânicas automáticas e controladas por circuitos integrados, tornando sistemas mecânicos motorizados, controlados manualmente ou automaticamente por circuitos elétricos. A aplicação dos robôs nas empresas e indústrias, tem obtido êxito um modo geral, em questões levantadas sobre a redução de custos, aumento de produtividade e os vários problemas trabalhistas com funcionários (OTTONI, 2010).

Quando um robô é na realidade, uma ferramenta de trabalho e ajuda a preservar o ser humano, como robôs bombeiros, submarinos, cirurgiões, entre outros tipos. O robô pode auxiliar a reintegrar algum profissional que teve parte de suas capacidades motoras reduzidas devido a doença ou acidente e, a partir utilização da ferramenta robótica, ser reintegrado ao mercado. Além disto, estas alternativas permitem que seja preservada a vida do operador (OTTONI, 2010).

Figura 4 - Robôs industriais

Fonte: Souza (2018)

Num contexto industrial podemos definir a automação como uma área da tecnologia que se ocupa de sistemas mecânicos, eletrônicos e à base de computadores na operação e controle de produção. Na automação industrial existem 3 tipos de automação onde são:

- Fixa: Volume de produção muito elevado.
- Programável: Volume de produção relativamente baixo e há uma variedade de produtos a serem fabricados.
- Flexível: Médio volume.

Um dos países que teve um grande investimento na área de robotização foi o Japão, um exemplo foi a Toyota. Na década de 1980 teve grandes avanços obtidos graças às indústrias automobilísticas investiram na automação de suas linhas de montagem. Os braços robóticos industriais (ou manipuladores robóticos) modernos aumentaram sua capacidade e desempenho com o uso de microprocessadores e linguagens de programação mais avançadas.

2.6 Robótica aplicada à deficiência visual

2.6.1 Considerações gerais

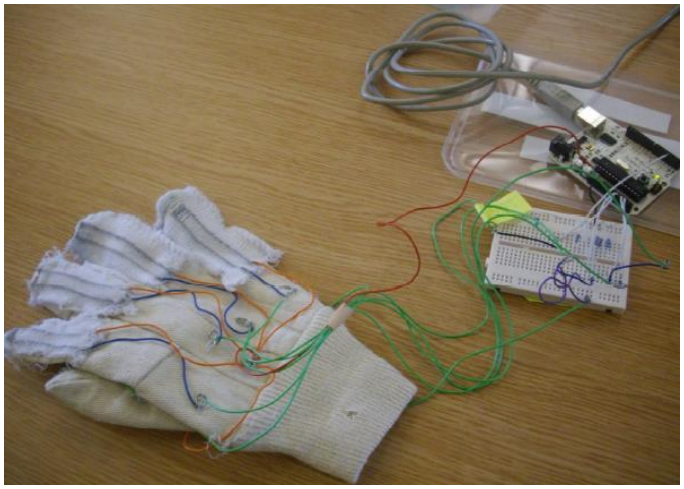
A deficiência visual é a diminuição ou perda da capacidade visual onde não se tem uma melhora ou correção através do uso de lentes, de tratamento clínico ou cirúrgico como apresenta o Instituto Benjamin Constant (CONSTANT, 2018).

Existem vários tipos de auxílio à pessoas com deficiência visual, serão mostrados a seguir, alguns tipos de robótica aplicada, onde será comentado com um breve resumo descritivo e imagens de cada uma delas.

2.6.2 Luva Sensorial

É uma luva que contem sensores térmicos aparelhados uniformemente na mão do usuário, conforme a figura 5. Estes sensores são estimulados com o aumento da temperatura fazendo com que um atuador ligado a luva vibre mais intensamente. Com a utilização desta luva, será possível evitar que um deficiente visual exponha a sua pele a altas temperaturas, como exemplo, retirar uma panela do fogo, uma forma do forno elétrico, onde o deficiente identifica o quão quente está o objeto manipulado e elimina a preocupação com manipulação de partes muito quentes (FILHO; VASCONCELOS; MOREIRA, 2010).

Figura 5 - Protótipo da luva sensorial

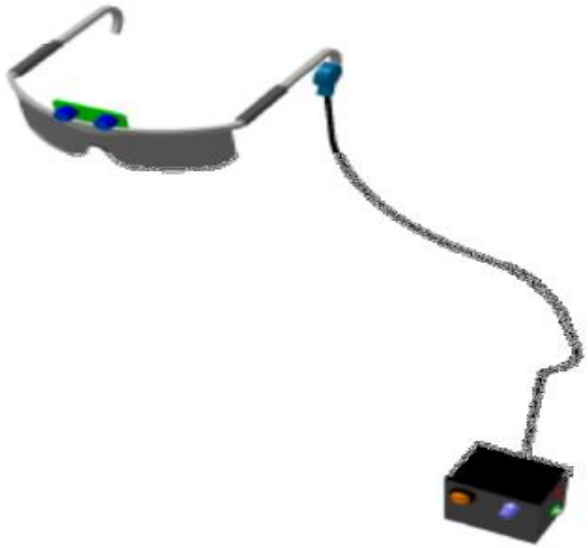


Fonte: Filho, Vasconcelos e Moreira (2010).

2.6.3 Óculos Sonar

Esse protótipo na figura 6, utiliza um sensor ultrassônico bem no centro dos óculos, que realiza a interface das informações no ambiente, possibilitando a detecção dos obstáculos a uma distância de 0,20 metros a 1,50 metros. Para iteração com o usuário, utilizou-se um pequeno alto-falante com um potenciômetro que quanto mais próximo do obstáculo, maior é a frequência do sinal sonoro emitido em um fone de ouvido, que o deficiente visual estará usando (GONZATTO et al., 2009).

Figura 6 - Óculos sonar



Fonte: Gonzatto et al. (2009)

2.6.4 Bengala Eletrônica

É uma bengala eletrônica baseada em sensores ultrassônicos para detectar obstáculos que estão ao seu redor e determinar a distância. Esta bengala pode identificar obstáculos a uma distância de 0,15 metros a 6,45 metros, a resposta chega até o usuário em forma de vibração, onde através do tato o deficiente visual vai interpretar, logo que a vibração vai acontecendo, é possível distinguir a distância do obstáculo (ALESSI; PASSOS; RIBEIRO, 2010).

3 ALGORITMO E PROGRAMAÇÃO

3.1 Considerações gerais

Quando expressamos a palavra lógica de programação, é utilizado vários termos como pseudocódigo, algoritmos, fluxograma, entre outras linguagens. No instante que uma atividade estabelecida passa a ser efetuada por máquinas, inclusivamente por computadores, ao invés de ser executada pelo homem, estamos realizando um processo de automação (FISCINA; BORGES, 2012).

Para que a automação de uma determinada ação tenha êxito, é fundamental que o equipamento que vá realizar o procedimento obrigatoriamente teve ter alguns requisitos para desempenhar todas as etapas que o envolve com precisão e no menor espaço de tempo possível. A identificação dos passos a serem seguidos e suas regras é dado o nome de algoritmo, ou seja, para que o computador possa executar uma determinada função, será essencial que seja detalhado passo a passo, através de uma forma clara e sucinta pela máquina para ser utilizado no que chamamos de programa (FISCINA; BORGES, 2012).

De acordo com Forbellone e Eberspacher (2005), o algoritmo tem como desígnio principal de estudar a lógica de programação para construir algoritmos coerentes. Algoritmo é um conjunto de passos que um programa deve seguir, onde no fim dessa lógica vai proceder a um objetivo final, conforme que precisarmos especificar uma sequência de passos, é necessário utilizar ordens, ou seja ‘pensar com ordem’, pois isso visa utiliza a lógica de programação.

Um algoritmo é feito temos que especificar ações claras e distintas, que a partir de um estado inicial, após um determinado tempo finito, produzem um estado final previsto e bem sucedido. Com isso significa que o algoritmo fixa um padrão de comportamento a ser seguido e executando a cada linha de programa, e com vista a alcançar o resultado final, a solução de um problema, onde sempre garante que quando for executado, sob a mesma condições, vai produzir o mesmo resultado desejado (FORBELLONE; EBERSPACHER, 2005).

3.2 Linguagem c / c ++

A programação em C++ é uma linguagem de baixo nível e dirigida a objeto. No caso essa linguagem ser de baixo nível ser igual e compatível com linguagem C pura, a linguagem C++ pode gerar programas muito rápidos e competentes. Geralmente é usada para criação de jogos, software de gráficos, controle de hardware e outras aplicações. Como essa linguagem é

orientada a objeto, C++ tem o poder e a agilidade de escrever programas em grande escala. C++ é uma das linguagens de programação mais populares para todos os tipos de programas (DAVIS, 2016),

A maior parte dos programas que é utilizado nos computadores todos os dias são escritos em C++ (ou seu subconjunto, a linguagem C). A principal finalidade de programar em C++ e escrever uma série de comandos que possam ser resumidos em um programa de linguagem de máquina, que realmente vai fazer o que queremos de resultado final. Essa conversão e chamada de compilação, essa é a função do compilador. O código de máquina que o usuário escreve tem que ser combinado com configurações e instruções que se desfazem algumas rotinas padrões das bibliotecas em um processo chamado ligação (DAVIS, 2016).

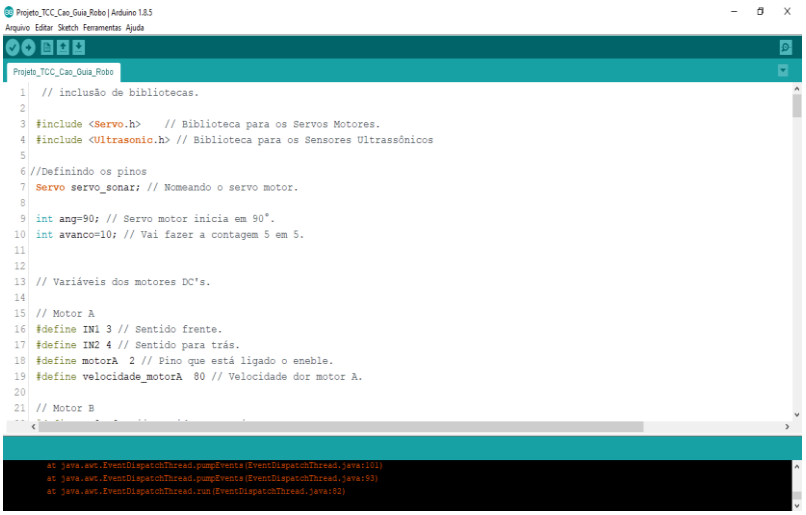
3.3 Hardware e software utilizados

Arduíno é uma plataforma de prototipagem eletrônica de hardware livre, projetada com um microcontrolador com suporte de entrada/saída embutido, uma linguagem de programação padrão é C/C++ versão 1.8.5, ela foi criada na cidade Ivrea, Itália em 2005 pelo italiano Massimo Banzi e outros colaboradores para auxiliar no ensino de eletrônica para estudantes.

O objetivo principal de Massimo Banzi foi criar uma plataforma de baixo custo, para que os estudantes pudessem desenvolver seus protótipos com o menor custo possível. Outro ponto interessante do projeto, foi a proposta de criar uma plataforma de código aberto, disponível para a comunidade e teve como intuito de interagir em projetos educacionais, de forma a ter um baixo custo de outros sistemas eletrônicos de prototipagem realizados naquela época na Itália (GOMES, 2016).

A figura 7 mostra a plataforma de edição de programção.

Figura 7 - Plataforma IDE Arduino



Fonte: Autor

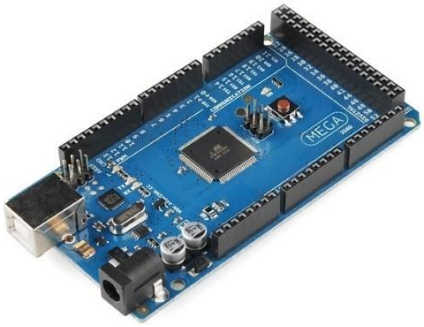
O Arduino *IDE* é uma aplicação multiplataforma escrita em Java derivada dos projetos *Processing e Wiring.* Tem um esquema para introduzir a programação para pessoas leigas não familiarizadas com o desenvolvimento de *software.* Inclui um editor de código com recursos de realce de sintaxe, parênteses correspondentes e endentação automática, sendo capaz de compilar e carregar programas para a placa com um único clique, essa troca de informação entre máquina e placa é via cabo *USB* (GOMES, 2016).

3.4 Componentes

3.5 Placa arduino mega 2560

A placa Arduino Mega 2560 é um microcontrolador da plataforma Arduino que possui soluções bem diferenciados para prototipos e projetos mais elaborados. O microcontrolador na figura 8 foi utilizado no desenvolvimento do protótipo foi o ATmega2560, composto por 54 pinos de entradas e saídas digitais onde 15 destes podem ser utilizados como saídas *PWM.* Possui 16 entradas analógicas, 4 portas de comunicação serial. Além disso ela tem uma disponibilidade quantidade de memória bem maior, sendo uma ótima opção para projetos que necessitem de muitos pinos de entradas e saídas além de memória de programa com maior habilidade para execução dos programas (SOUZA, 2018).

Figura 8 - Placa Arduino Mega 2560



Fonte: Souza (2018)

Alimentação da placa pode ser feita por cabo *USB* ou até mesmo pela conexão por pino *Jack*. O valor de tensão da fonte externa deve estar entre os limites 6V a 20V, porém pode ser alimentada com uma tensão abaixo de 7V, a tensão de funcionamento interno da placa, que no Arduino MEGA 2560 é de 5V, pode ficar instável e quando alimentada com tensão acima de 12V, o regulador de tensão da placa pode sobreaquecer e danificar a placa. Dessa forma, é recomendado para tensões de fonte externa valores de 7V. a 12V (SOUZA, 2018).

3.6 Ponte H modelo L298N

A maioria das aplicações na robótica é necessária a utilização de motores de corrente continua em diversos tipos de locomoção de robôs, movimentação de braços mecânicos, etc. Os motores DC são cargas indutivas que em geral, demandam uma quantidade de corrente superior à que as portas da placa Arduino conseguem disponibilizar (CARDOSO,2018).

Sendo assim, não devemos ligar estes motores diretamente nas portas do Arduino pois se o motor demandar uma corrente acima de 40mA nas portas digitais (máxima fornecida pelo Arduino) pode queimar a porta e danificar a placa. Para solucionar o problema da alta corrente podemos usar transistores, na figura 9 mostra a ponte H, aonde é possível ter um controle nos sentidos de giro do motor, função que não se faz possível usando apenas um transistor já que para inverter o sentido de giro devemos inverter a polaridade da alimentação do motor.

Para resolver nosso problema utilizamos um famoso circuito conhecido como Ponte H que nada mais é que um arranjo de 4 transistores. Este circuito é uma elegante solução por ser

capaz de acionar simultaneamente dois motores controlando não apenas seus sentidos, como também suas velocidades. Além de seu uso ser simples no arduino (CARDOSO, 2018).

Figura 9 - Ponte H modelo L298N

Fonte: Cardoso (2018)

Dados da ponte H:

De acordo com Thomsen (2018) segue abaixo os dados da ponte H L298N.

- **6-35V**: Porta para alimentação da placa com tensão entre 6 a 35V.
- **Ativa 5V**: Quando jumpeado, a placa utilizará o regulador de tensão integrado para fornecer 5v (na porta 5v) quando a porta 6-35V estiver sendo alimentada por uma tensão entre 6 e 35V. Neste caso, não se deve alimentar a porta 5V pois pode danificar os componentes. A tensão fornecida na porta 5V pode ser usada para alimentar o Arduino, por exemplo.
- **5v**: Em casos de não haver fonte de alimentação com mais de 6V podemos alimentar a placa com 5V por esta porta.
- **Ativa MA**: Quando jumpeado aciona o motor A com velocidade máxima. Para controlar a velocidade do motor A basta remover o jumper e alimentar o pino com uma tensão entre 0 e 5v, onde 0V é a velocidade mínima (parado) e 5V a velocidade máxima.
- **Ativa MB**: Quando jumpeado aciona o motor B com velocidade máxima. Para controlar a velocidade do motor B basta remover o jumper e alimentar o pino com uma tensão entre 0 e 5v, onde 0V é a velocidade mínima (parado) e 5V a velocidade máxima.
- **IN1 e IN2:** são utilizados para controlar o sentido do motor A;
- **IN3 e IN4**: são utilizados para controlar o sentido do motor B;

3.7 Sensor ultrassônico HC – SR04

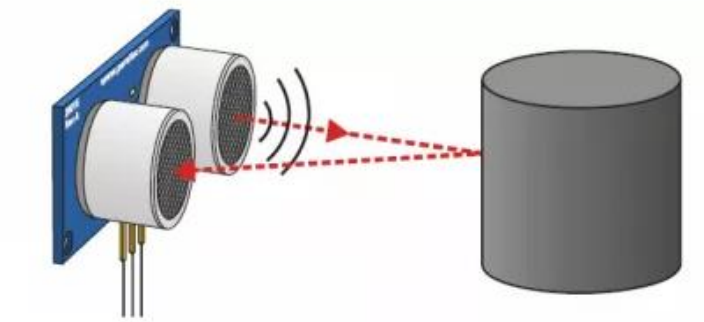
Este sensor ultrassônico geralmente são utilizados em aplicações onde se deseja medir distância ou evitar colisões, como na robótica móvel e de reabilitação. A figura 10 abaixo ilustra o sensor que foi utilizado no protótipo.

Figura 10 - Sensor Ultrassônico

Fonte: Cardoso (2018)

Para Cardoso (2018) o sensor emite um pequeno pulso sonoro de alta frequência que se propaga na velocidade do som. A partir do momento que este pulso atingir um objeto, um sinal de eco será refletido para o sensor, a distância entre o sensor e o objeto pode então ser calculada caso saibamos o tempo entre a emissão e a recepção do sinal, além da velocidade do som no meio em questão. A figura 11 abaixo vai explicar o processo de detecção de obstáculos.

Figura 11 - Sensor ultrassônico detectando obstáculo

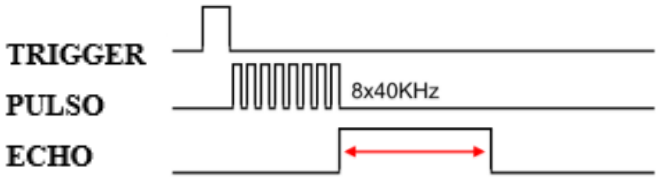


Fonte: Cardoso (2018)

Na figura 12 mostras o procedimento de leitura do pino *trig*, ele funciona como gatilho no sensor ultrassom, porém recebe um pulso de 5V por pelo menos 10 microssegundos. Isso fará com que o sensor emita 8 pulsos ultrassônicos em 40kHz e o pino *echo*, que funcionará como nosso cronômetro, vai para 5V, iniciando, o sensor fica esperando as ondas refletidas que foi detectado do objeto. Assim que uma onda refletida for detectada, o pino e*cho*, que estava em 5V, será alterado para 0V.

Desta forma, o período que o pino *echo* fica em 5V é igual ao tempo que a onda emitida leva para ir até o obstáculo e voltar (THOMSEN, 2018).

Figura 12 - Amostragem da frequência



Fonte: Thomsen (2018)

3.8 Chassi 2WD

Este Kit Chassi 2WD é destinado para montagem de um chassi para aplicações robóticas e trabalhos educacionais. Na figura 13 mostra o chassi que é feito de acrílico e acompanha uma película adesiva para proteção arranhões. Possui dois motores CC e também possui caixa de redução (1:48) que são conectados independentemente em cada roda, e mais uma roda livre (universal) para dar sustentação, esta roda fica na parte traseira do chassi (FILIPE FLOP, 2018).

Figura 13 - Chassi 2WD

Fonte – Filipe Flop (2018)

3.9 Servo motor tower pro MG 946R

Na figura 14 ilustra o servo motor MG 946R possui um torque muito elevado, além disso ele tem um novo sistema de controle, com isso ele torna mais preciso em seus movimentos. Sua engrenagem interna e eixo são de alumínio deixando mais resistente e leve. Também são atualizados para melhorar a largura de banda morta e centralização (TOWER PRO, 2018).

Figura 14 - Servo Motor MG 946R

Fonte: Tower Pro (2018)

Dados do fabricante Tower Pro:

- Peso: 55g.
- Dimensão: 40,7 × 19,7 × 42,9 mm.
- Torque de parada: 10,5 kg / cm (4,8 v); 13 kg / cm (6 v).
- Velocidade de operação: 0.20seg / 60degree (4.8v); 0.17sec / 60degree (6.0v).
- Tensão de funcionamento: 4.8-6.6 v.
- Faixa de temperatura: 0-55g.
- Plug Servo: JR (se encaixa JR e Futaba).
- Tipo de Engrenagem: Metal Gear.
- Largura da banda morta: 1us.
- Fonte de Alimentação: Através de Adaptador Externo.
- comprimento do fio servo: 32cm.
- Empate atual em marcha lenta 10MA.
- Sem corrente de operação de carga desenhar 170MA.
- Corrente de empate 1200MA.

3.10 Baterias

A bateria é um dispositivo que contêm uma energia armazenada, porém converte a energia química contida em seus materiais ativos, diretamente em energia elétrica, necessária para funcionamento do conjunto, por meio de uma reação eletroquímica de oxidação e redução. A reação química, que ocorre internamente, envolve a transferência de elétrons dos materiais que se oxidam para os materiais que se reduzem através de um circuito elétrico (MICHELINI, 2017).

A bateria utilizada foi de 9 Volts, uma para parte de comando de controle, na qual é o microcontrolador que vai executar todos os parâmetros programados. A outra alimentação de 9 volts é para parte de potência, ou seja, os motores de 5 Volts. A figura 15 mostra a bateria que foi utilizado no projeto.

Figura 15 - Bateria de 9Volts

Fonte: Fabricante Duracell (2018)

3.11 Cabos

Os cabos utilizados foram os cabos *jumpers* macho/macho, macho/fêmea e fêmea/fêmea. A utilização destes cabos teve como objetivo de ligar as partes elétricas e montar todo o circuito desde os sensores ultrassônicos, alimentação, servo motor e motores. Logo abaixo, a figura 16 irá mostrar os cabos que foram utilizados no projeto.

Figura 16 - (A) Cabos macho/macho (B) Cabos macho/fêmea (C) Cabos fêmea/fêmea

Fonte: Cardoso (2018)

Outro tipo de cabo utilizado, foi o para comunicação entre robô e controle de mão. Foram utilizados cabo de rede de internet 8 vias para este fim.

Figura 17 - Cabos de rede

Fonte: Mercado Livre (2018)

3.12 Controle de mão

3.12.1 Construção do driver

Para construir o circuito foi utilizado o software *protheus*, esse circuito vai fazer o acionamento dos motores *vibracall.*

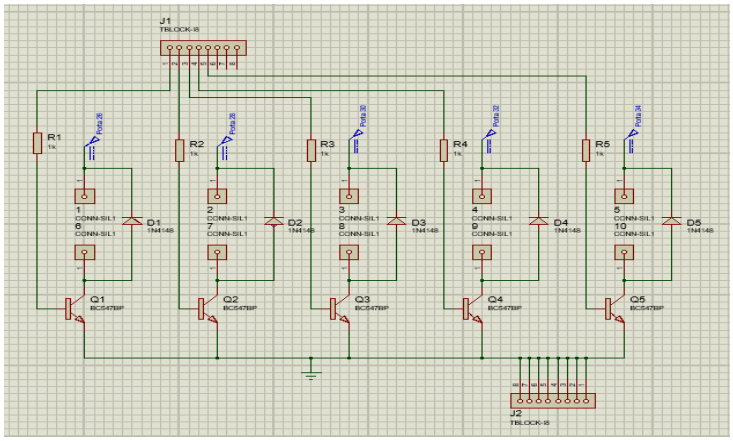
Componentes utilizados:

- 5 transistores BC 547;
- 5 diodo 1N4148;
- 5 resistores de 1k;
- 8 conectores KRE 2 vias;
- 10 cabos para fazer a ligação para os motores;
- Placa de fenolite;

O princípio de funcionamento do circuito abaixo, vai ser alimentado com 5V, para ligar os motores, e para fazer o chaveamento do transistor vai ser enviado um pulso de 5V da porta digital da arduino. Para não ocorrer a queima do transistor foi colocado em paralelo com o motor o diodo 1N4148, isso devido a corrente de retorno após de feito o acionamento, e foi

colocado um resistor de 1k ohms na entrada da chave para ter uma queda de tensão. Na figura 18 mostra o circuito projetado no *software Protheus.*

Figura 18 - Circuito projetado no software Protheus

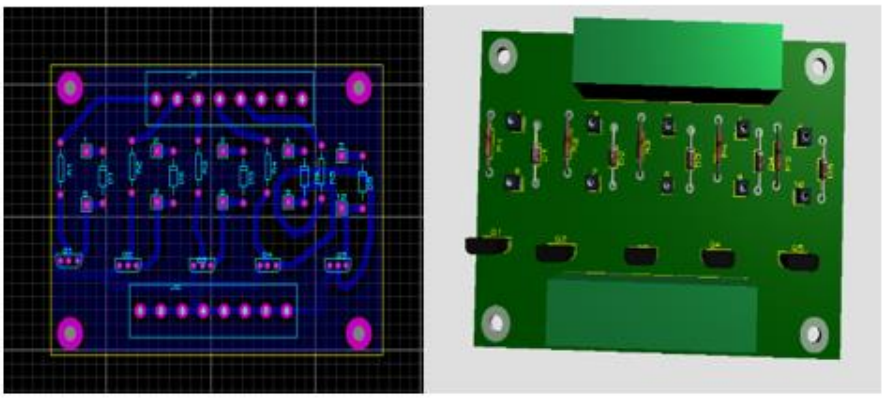


Fonte: Autor

3.12.2 Confecção da placa

Então a primeira etapa foi calcular o tamanho da placa e fazer a montagem do circuito, conforme a figura 19 abaixo.

Figura 19 - Montagem do circuito no software Protheus



Fonte: Autor

Após ter feito a montagem do circuito no *software*, foi realizado a impressão do circuito em uma impressora de jato de tinta, tem que ser esse tipo de impressora porque vai ser realizado um processo de transferência de imagem da folha para a placa de fenolite, a figura 20 irá mostrar o processo de transferência de imagem.

Sabendo-se o tamanho da placa então realizou-se o corte de 5 x 8 cm da placa de fenolite, e em seguida foi posicionado o circuito impresso em cima da placa cortada nas dimensões desejada para que possa fazer o processo de transferência de imagem por calor, foi utilizado o ferro de passar roupa aplicando sobre a superfície em torno de 20 minutos.

Figura 20 - Processo de Transferência de Imagem

Fonte: Autor

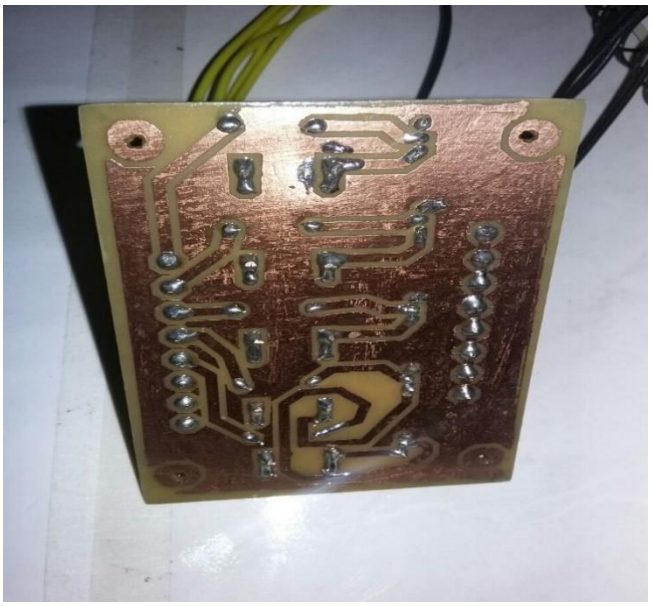
E seguida a placa de fenolite foi colocada de baixo de uma torneira de água corrente para fazer a remoção do papel, isso tudo com cuidado para que não destrua a transferência de imagem, depois da retirada do papel foi realizado o processo de corrosão da placa, foi utilizado percloreto de ferro. Em seguida foi fazer a furação dos locais dos componentes que irão ficar na placa, será mostrado na figura 21 abaixo.

Figura 21 - Furação da Placa

Fonte: Autor

O próximo processo que foi realizado foi de soldar os componentes na placa, e fazer teste de continuidade para verificar se não houve encontros das trilhas do circuito, a figura 22 ilustra o procedimento de soldagem.

Figura 22 - Placa soldada



Fonte: Autor

3.12.3 Confecção do controle e posicionamento dos motores vibracall

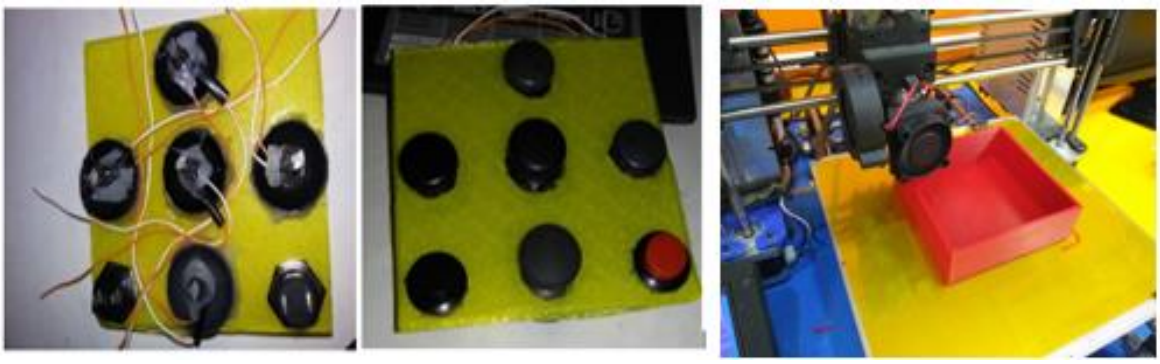
Foi feito o desenho do controle no software *solid works*, para determinar o tamanho do controle que irá ficar na mão do deficiente visual, os motores *vibracall* vão ficar fixados de baixo das capas de borracha, essas capas vão ser coladas com cola quente na parte superior do controle, portanto quando o motor receber o sinal de liga, terá a vibração nessas capas de borracha. Além disso o controle terá dois botões um para comando de virar à direita e outro pra virar à esquerda. Na figura 23 mostra as capas de borracha que foram utilizadas no controle de mão.

Figura 23 - Capas de Borracha

Fonte: Autor

Com as dimensões conforme o projeto no *solid Works*, foi feito uma caixa para colocar a placa do circuito de acionamento dos motores *vibracall*, botões e as capas de borrachas. Os motores *vibracall* foram colados com cola quente conforme a figura 24 abaixo. As conexões dos motores foram feitas com cabo de rede e soldados com estanho para ficar bem presos aos terminais. Porém o recorte dos furos na tampa da caixa, foi realizada com o ferro de solda para ficar bem adequados na colocação das capas de borrachas.

Figura 24- Colagem das borrachas, motores vibracall e fabricação da caixa na impressora 3D



Fonte: Autor

4 SISTEMA PROPOSTO

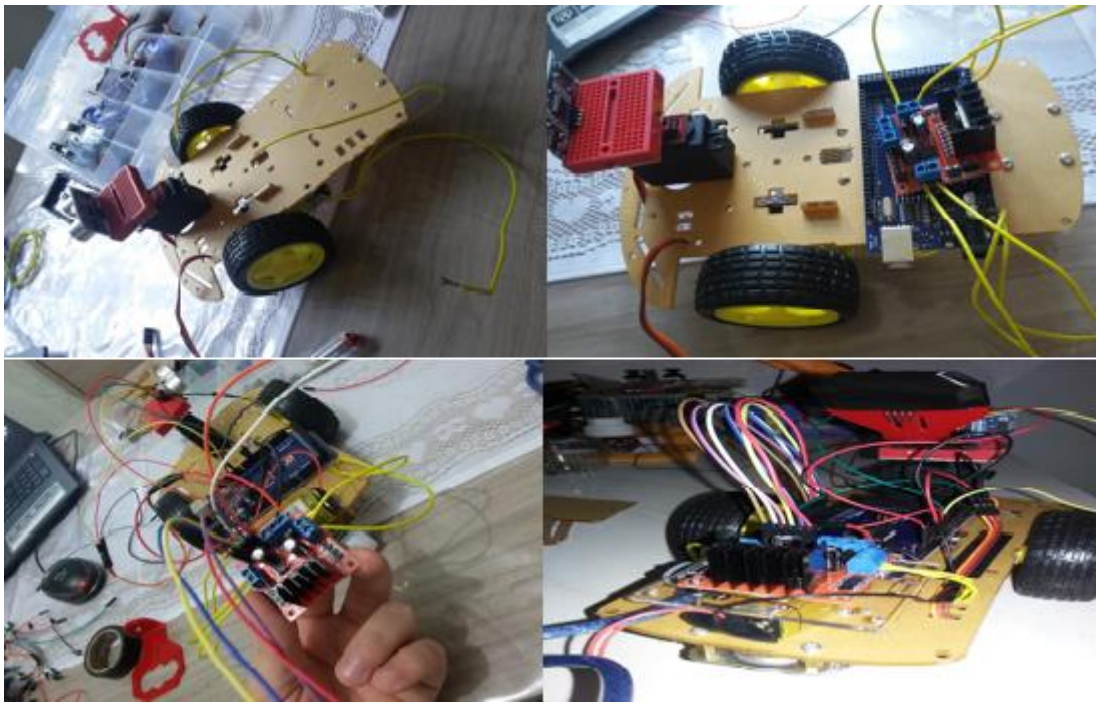
4.1 Visão da programação do robô

A programação do robô foi desenvolvida para proporcionar às pessoas com deficiência visual em ter uma maior segurança em relação a sua movimentação em ambientes que contenham muitos obstáculos. A principal ideia desse robô autônomo, é ele monitorar a área no perímetro de 0° à 180° na parte frontal e superior, no qual será possível detectar os obstáculos presentes ao seu redor, através de sensor ultrassônico específicos e outro sensor, só pra identificar desníveis. A partir dos sensores ultrassônicos o usuário será capaz de calcular a distância entre o obstáculo e o deficiente visual, receberá um alerta sobre a sua proximidade dos objetos.

4.2 Montagem do chassi e sensores

Essa etapa de montagem foi realizada conforme o projeto feito no *software Solid Works*, foi verificado os lugares que os sensores iriam ficar, posição da placa de controle, ponte H e bateria da alimentação do robô. A figura 25 abaixo mostra as primeiras versões de montagem desse protótipo.

Figura 25 - Montagem dos componentes



Fonte: Autor

4.3 Instalações e programação do protótipo

Inicialmente, foi necessário realizar as instalações das bibliotecas no *IDE* arduino, para que seja possível possa utilizar os sensores ultrassônicos e o servo motor. Em seguida foram feitas as declarações da variáveis e definição dos pinos das portas digitais do microcontrolador. Também foram realizados pesquisas de princípio de funcionamento dos sensores. Para ter os parâmetros iniciais da programação do protótipo, foi preciso deixar todos os sensores nas posições corretas e alinhar o servo motor ao ponto 0°.

4.3.1 Princípio de funcionamento

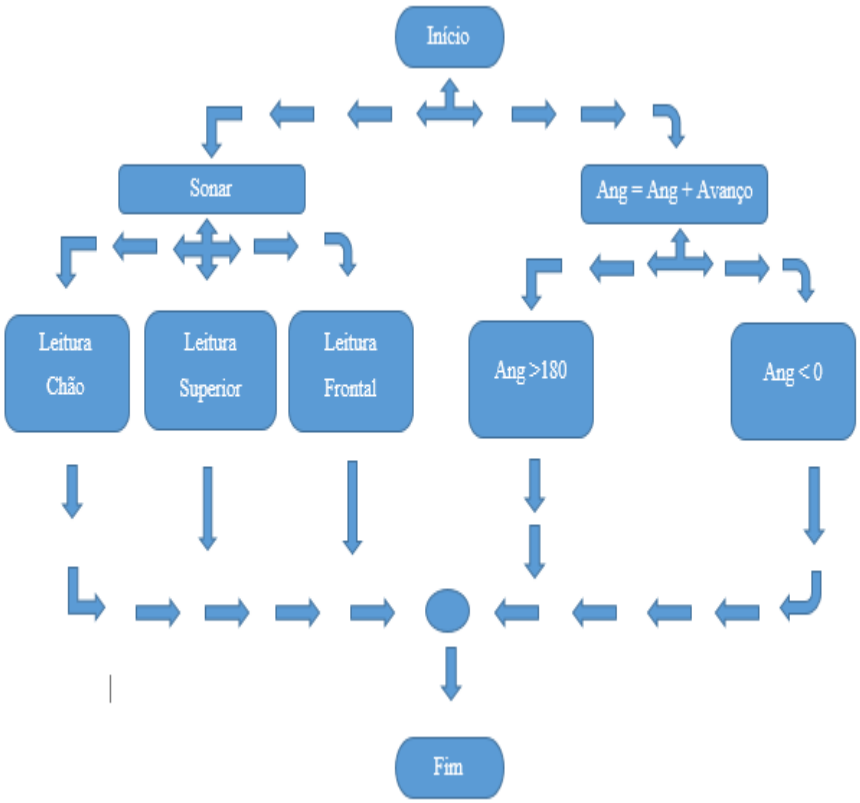
Será explicado cada laço de programação dos sensores e movimentos que o robô irá se comportar.

4.3.1.1 Laço principal da arduino (loop)

O Fluxograma abaixo da figura 26 irá mostrar o laço principal do microcontrolador, ou seja, a sua varredura nesse laço vai mostrar o comportamento dos sensores ultrassônicos e a movimentação do servo motor. A explicação de cada laço, será apresentada em seguida com mais detalhes conforme a programação realizada em C / C++. A execução dos laços será conforme atuação dos sensores, porém cada sensor tem sua forma de fazer a leitura e comparação, conforme a sua resposta vai acarretar a um movimento do robô autônomo.

Essa decisão do robô, resumir-se conforme foi planejado e programado dentro do ambiente *Sketch* Arduino *IDE,* vai ser em base e conforme os testes durante toda a execução do projeto.

Figura 26 - Fluxograma do laço Loop



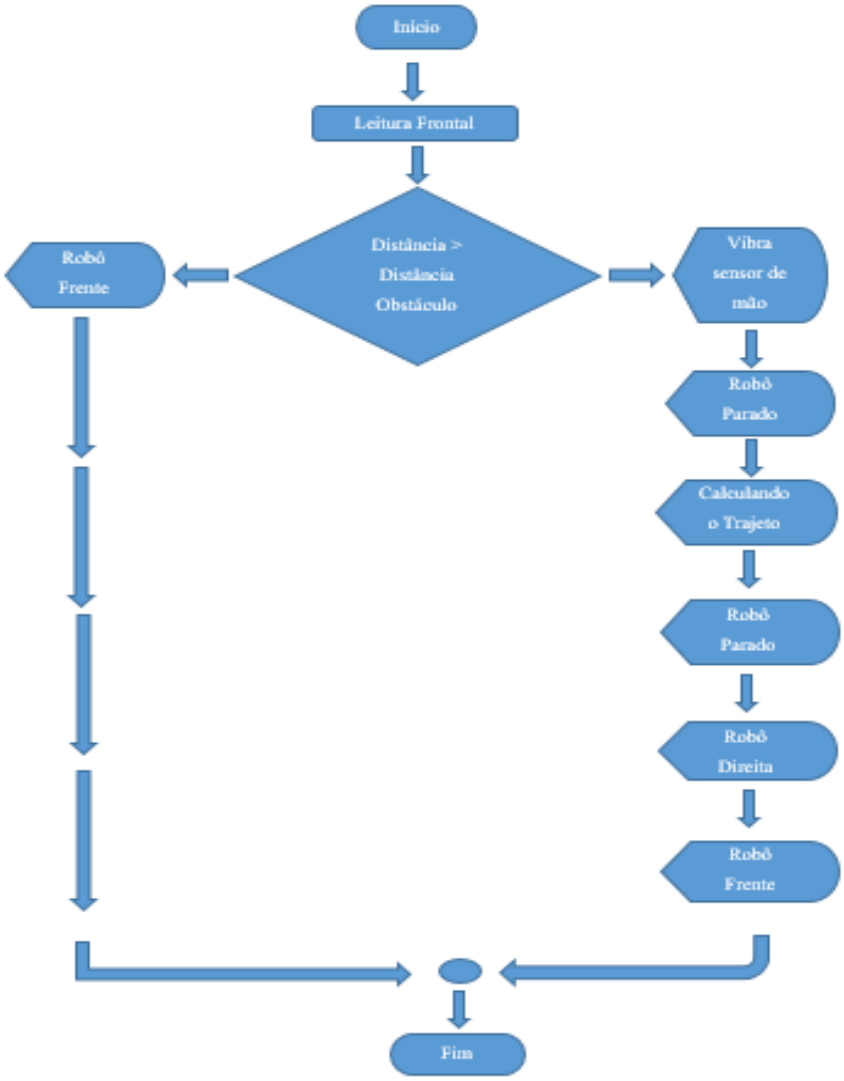
Fonte: Autor

4.3.1.2 Laço de detecção pelo sensor frontal

Para a programação que foi criada para o protótipo para detecção de obstáculos frontais, um fluxograma, apresentado a seguir será mostrado todos os passos do laço do sensor ultrassônico frontal. O laço do sensor faz a seguinte condição, foi declarado uma variável com o nome distância, onde essa variável vai ser igual a leitura do sensor frontal, porém foi feito uma comparação se a distância medida pelo sensor é maior ou igual a distância programada que foi selecionada por 40 centímetro do obstáculo.

Essa distância é segura para que o sensor possa reconhecer o limitação do trajeto e mandar o robô parar. Então na hora que o sensor detectar alguma coisa em seu caminho, vai comandar o robô parado, dar um sinal vibratório via cabo para um motor *vibracall* de 3 V, portanto o usuário vai reconhecer esse sinal, porque vai ter um ponto especifico no controle de mão que será atuado. Com esse sinal emitido para o deficiente visual irá saber aonde está o obstáculo, no perímetro de 0° até 180° como foi citado no início do trabalho, vai ter um servo motor fazendo essa monitoração nessa área. Afigura 27 mostra rotina de programação.

Figura 27 - Fluxograma do laço do sensor frontal



Fonte: Autor

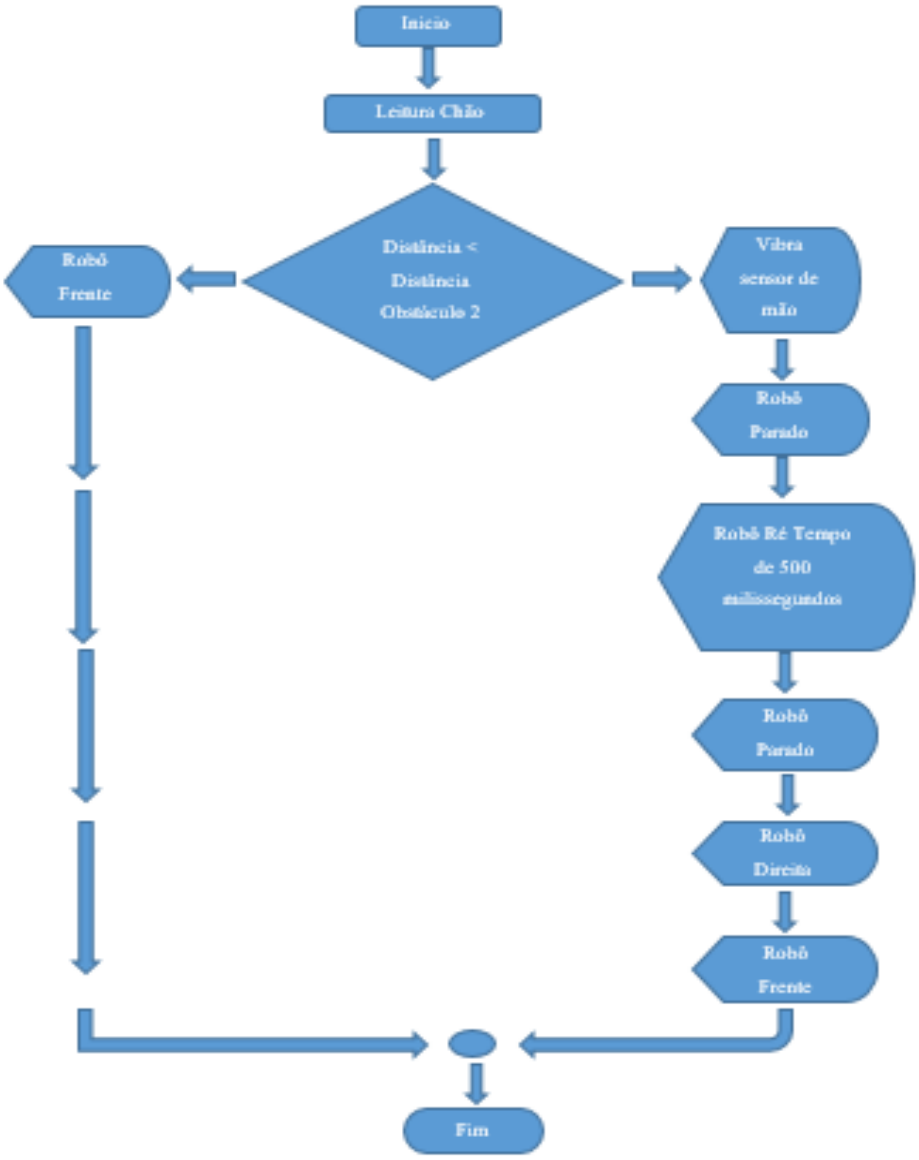
4.3.1.3 Laço de detecção pelo sensor chão

Neste laço vai ser explicado o sensor ultrassônico que foi direcionado para o chão, que vai detectar todas as imperfeições, degraus e buraco. O princípio de funcionamento é o mesmo que os outros sensores ultrassônicos, ou seja, foi declarado uma variável inteira distância 2 que é igual a leitura do sensor chão, porém faz a comparação com a distância 2 com a distância selecionada no programa que foi inicialmente ajustada em 8 centímetros.

Qualquer coisa que for menor que esse valor, o deficiente visual vai receber um sinal vibratório em outro ponto especifico no controle de mão, em seguida o robô vai parar, dar uma ré de um tempo de 500 milissegundos, para novamente vai se posicionar para direita e vai retornar seu caminho que é pra frente. Se acaso contrário dessa condição o robô vai seguir andando para frente.

O fluxograma abaixo na figura 28 mostra de forma mais clara esse laço.

Figura 28 - Fluxograma do laço do sensor de chão



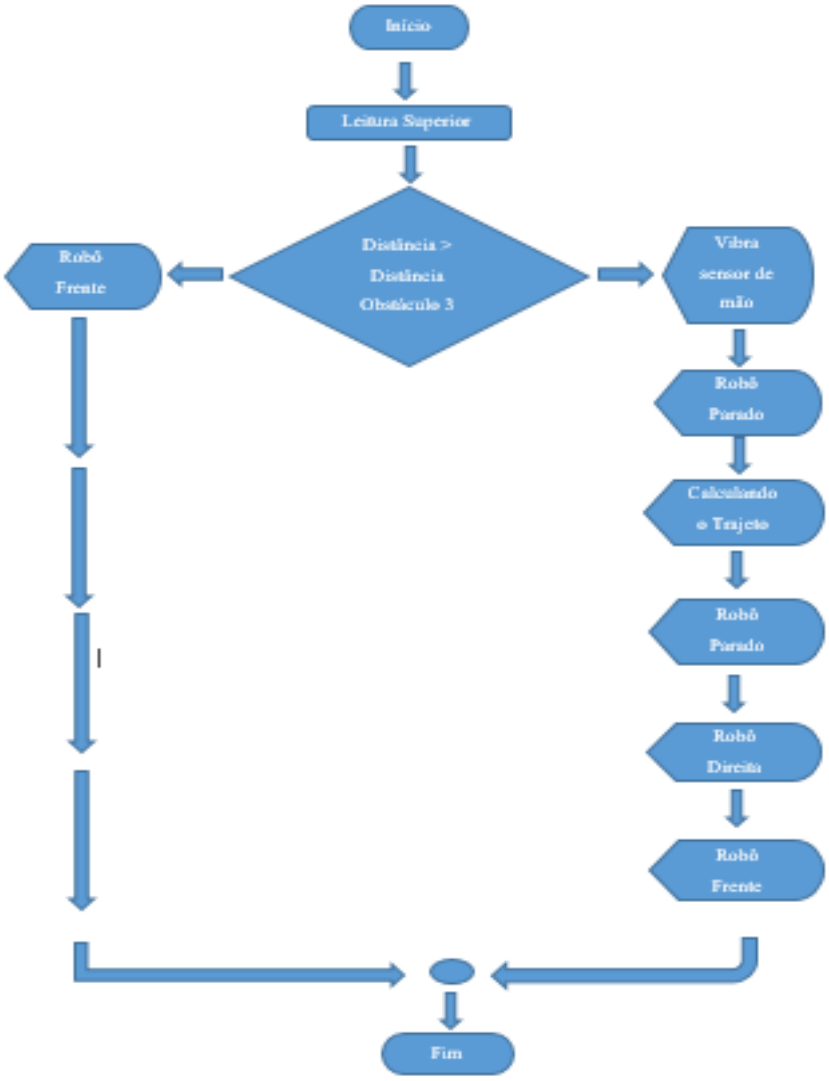
Fonte: Autor

4.3.1.4 Laço de detecção pelo sensor superior

A programação desse sensor superior teve como objetivo reconhecer obstáculos em uma distância de 1,40 cm, dentro da área que o servo motor ficará monitorando. As declarações de variáveis desse laço, foi declarado uma variável inteira com o nome de distância 3, logo em seguida foi realizado uma comparação com a leitura do sensor ultrassônico superior.

Foi dado a seguinte condição, se a distância 3 for maior que a leitura do sensor, robô anda para frente, caso contrário o usuário receberá um sinal de vibração no controle de mão, vai parar o robô em seguida calcula o trajeto, vira a direita e segue para frente. A figura 29 explica a rotina do sensor superior.

Figura 29 - Fluxograma do laço do sensor superior

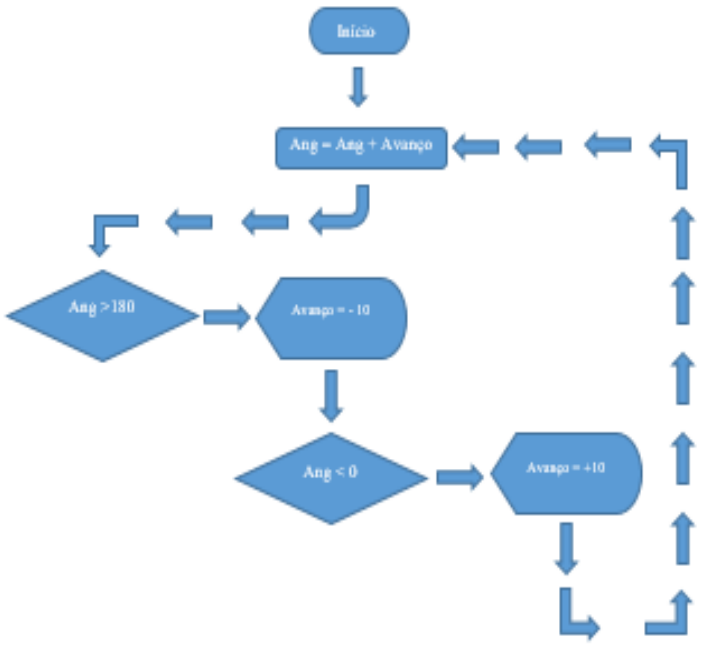


Fonte: Autor

4.3.1.5 Laço de funcionamento do servo motor

Esse laço, na figura 30 explica, como o servo motor vai se comportar e fazer o seu movimento de 0° até 180°, ele ficará em repetição. O fluxograma abaixo vai apresentar como foi programado o servo motor.

Figura 30 - Fluxograma do servo motor



Fonte: Autor

4.4 Inicialização do robô

Para que o robô ligue, foram colocadas duas chaves de liga e desliga, uma da alimentação da placa e a outra dos motores, as duas alimentações terão uma bateria de 9V para cada aplicação, o deficiente visual terá que ligar essas chaves para que robô possa fazer a sua inicialização. No controle que irá ficar na sua mão ele vai ter a opção de virar à direita ou à esquerda, a velocidade do robô é controlado pelo pulso *PWM* do microcontrolador, esse valor está em 120, o usuário terá um botão de parar o seu cão robô. Na figura 31 abaixo mostra o protótipo.

Figura 31 - (A) Indicação da chave de liga e desliga placa arduino, (B) Indicação da chave liga/desliga ponte H

(A) **(B)**

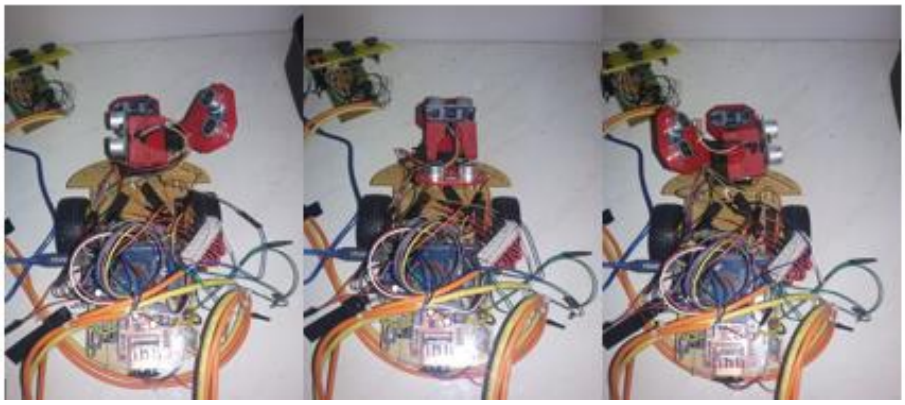
Fonte: Autor

Após o usuário ligar essas duas chaves o robô irá fazer a sua inicialização, no qual, o programa que está sendo executado no microcontrolador vai fazer todos os parâmetros, determinados conforme seu funcionamento exige e como foi projetado. O deficiente visual deverá pegar o controle de mão nesse estante, para que o robô possa guia – lo.

Se o deficiente visual querer virar à direita ou a esquerda ele vai ter dois botões pra fazer essa função durante a sua locomoção e em seguida o robô vai dar um sinal vibratório para orientar o usuário indicando qual o lado o robô virou.

Após ligar os dois botões, o robô tem a sua inicialização, no qual o servo motor se posiciona em 90°, em seguida ele vai para 0° e fica rastreando até 180°, a figura 32 abaixo vai mostrar essas posições.

Figura 32 (A) Servo motor em 180° (B) Servo motor em 90° (C) Servo motor em 0°



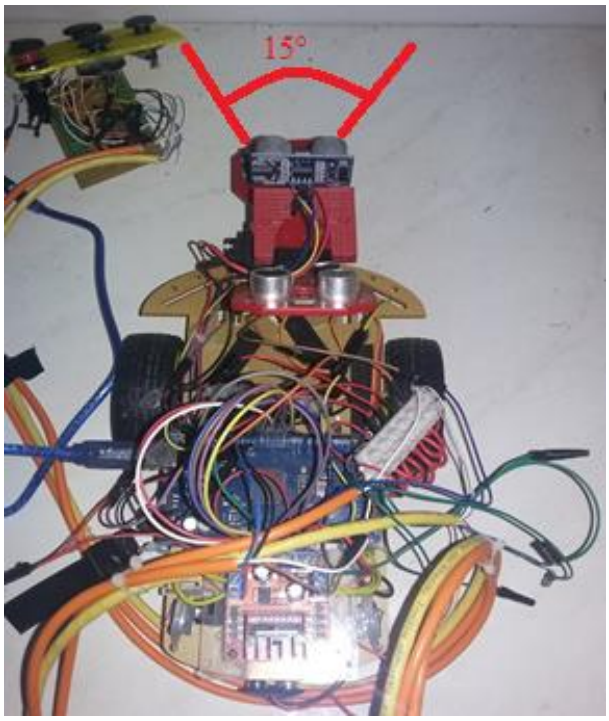
(A) **(B)** **(C)**

Fonte: Autor

4.5 Restrição de leitura dos sensores ultrassônicos

Os sensores ultrassônicos tem uma limitação do ângulo no seu emissor de sinal, esse ângulos possui 15°, por isso que foi utilizado um servo motor para eliminar esse problema, a figura 33 irá mostrar a posição do servo motor.

Figura 33 Ângulo de alcance do Sensor Ultrassônico



Fonte: Autor

4.6 Descrição de custos do protótipo

Quadro 1: Ilustração do custo para realizar o projeto

Item	Unidade	Custo Unitário	Total
Chassi 2WD	1	R$ 71,90	R$ 71,90
Servo Motor	1	R$ 42,90	R$ 42,90
Cabos Jumper	2	R$ 15,90	R$ 31,80
Arduino MEGA	1	R$ 99,00	R$ 99,00
Sensor Ultrassônico	3	R$ 13,90	R$ 41,70
Botões sem Renteção	2	R$ 8,00	R$ 16,00
Placa de Fenolite	1	R$ 3,00	R$ 3,00
Percloreto de Ferro	1	R$ 15,00	R$ 15,00
Suporte de Sensor	2	R$ 11,50	R$ 23,00
Baterias	3	R$ 21,90	R$ 65,70
Chave com Rentenção	2	R$ 2,00	R$ 4,00
Conectores para Bateria de 9 V	2	R$ 1,50	R$ 3,00
Caixa de Polímero	1	-	-
Resistores	5	R$ 0,10	R$ 0,50
Transitor	5	R$ 4,76	R$ 23,80
Diodo Zener	5	R$ 3,60	R$ 18,00
Capas de Borracha	5	-	
Conexão de Derivação	2	R$ 8,50	R$ 17,00
Cola Quente	1	R$ 2,00	R$ 2,00
Motor Vibracall	5	-	-
Protoboard	1	R$ 6,00	R$ 6,00
TOTAL			**R$ 484,30**

Fonte: Autor

5 CONSIDERAÇÕES FINAIS

Neste estudo foi desenvolvido um protótipo que permite a interação homem/ máquina, no qual o robô é autônomo, porém o usuário terá a possibilidade de alterar o seu percurso.

Durante a execução do projeto foi possível analisar algumas restrições, no qual os sensores ultrassônicos possuem uma limitação em seu ângulo de envio de som que é de um ângulo de 15°, através de algumas pesquisas realizadas foi proposto a utilização de um servo motor, para monitorar entre o ângulo de 0° até 180°.

A velocidade do motor em comparação com a atuação dos sensores ultrassônicos é desprezível, ou seja, a rotação dos motores não irá interferir no reconhecimento dos obstáculos, o robô vai conseguir detectar, tomar uma ação de parada, e analisar a rotina de programação a ser executada.

Atualmente para adquirir um cão guia, o deficiente visual necessita preencher alguns requisitos específicos e ser capaz de promover os cuidados necessários ao animal, o solicitante deve se cadastrar em uma extensa fila de espera através de ongs focadas em treinamentos de cães guias, o processo é lento e o adestramento desses animais é bastante rigoroso e levam em torno de 2 anos.

Hoje no Brasil existem poucos cães guias em atividade, e o número de deficientes visuais é altíssimo quando comparada com a quantidade de cães em atuação.

Em relação ao cão o protótipo irá detectar obstáculos mais específicos, principalmente obstáculos aéreos, quanto ao treinamento, o usuário será instruído a manusear o robô, que possui uma forma simplificada de utilização, tornando sua locomoção mais segura e eficaz.

Para trabalhos futuros nota- se a necessidade de rodas adequadas e que possam ser utilizadas em diversos ambientes, faz se essencial a criação de estrutura externa apresentável, com estrutura resistente, que não ocorra corrosão e que seja impermeável, a comunicação entre robô e usuário ser via remota, isto é, que não seja via cabo, também é necessário um botão para que o robô pare, em relação as baterias, o ideal é que sejam baterias de chumbo ácido de 7,3 V 1,3 Ah.

Para que o robô seja exemplar é imprescindível a instalação de um módulo *GPS* que seja compatível com o microcontrolador arduino.

Sendo assim o protótipo está em fase de testes e adaptações, que devem ser aprimoradas e colocadas em prática.

6 REFERÊNCIAS BIBLIOGRÁFICAS

ALESSI, A.; PASSOS, I.; RIBEIRO, J. **DOMÓTICA: Bengala Eletrônica via Sensor de Ultrasom.** Brasil: Curitiba, 2010.

Amiralian, M. L. T. M. (1997). Compreendendo o cego: uma visão psicanalítica da cegueira por meio de desenhos-estórias. São Paulo: Casa do Psicólogo.

BASSALO, José Maria Filardo. **CURIOSIDADES DA FÍSICA.** Disponível em: <http://www.seara.ufc.br/folclore/folclore507.pdf>. Acesso em: 18 out. 2018.

BERSCH, Rita. **Introdução à tecnologia assitiva:** CEDI centro especializado em desenvolvimento infantil. Porto Alegre:, 2017. 20 p. Disponível em: <http://www.assistiva.com.br/Introducao_Tecnologia_Assistiva.pdf>. Acesso em: 18 mar. 2018.

BURLAMAQUI, Aquiles Medeiros Filgueira et al. CardBot - Assistive Technology for Visually Impaired in Educational Robotics: Experiments and Results. **Ieee Latin America Transactions**, Rio Grande do Norte, v. 15, n. 3, p.517-527, 1 mar. 2017. Disponível em: <ieeexplore.ieee.org/document/7867603>. Acesso em: 18 mar. 2018.

CARDOSO, Daniel. **Driver motor com Ponte H L298N – Controlando Motor DC com Arduino**. Disponível em: <https://portal.vidadesilicio.com.br/driver-motor-com-ponte-h-l298n/>. Acesso em: 14 mar. 2018.

CONSTANT, Instituto Benjamin. **Os Conceitos de Deficiência.** 2018. Disponível em: <http://www.ibc.gov.br/?itemid=396>. Acesso em: 27 maio 2018.

CRAIG, John J. **Robótica.** 3. ed. São Paulo: Pearson, 2012. 369 p. Heloisa Coimbra de Souza.

DAVIS, Stephen R. **C++ Para Leigos.** 7. ed. Rio de Janeiro: Alta Books, 2016. 447 p.

FISCINA, Fabrizio Leandro Fonsêca; BORGES, Marcio Vieira. **Licenciatura em Ciência da Computação:** Algoritmo. Salvador: Eduneb, 2012. 96 p.

FILIPE, Flop. **Kit Chassi 2WD Robô para Arduino.** 2018. Disponível em: <https://www.filipeflop.com/produto/kit-chassi-2wd-robo-para-arduino/#tab-description>. Acesso em: 25 maio 2018.

FILHO, J.; VASCONCELOS, F.; MOREIRA, A. **Uso de Robótica Assistiva no Auxílio de Pessoas com Deficiências Visuais.** Disponível em < http://connepi.ifal.edu.br/ocs/index.php/connepi/CONNEPI2010/paper/viewFile/220/193 > Acesso em: 30 set. 2018.

FORBELLONE, André Luiz Villar; EBERSPACHER, Henri Frederico. **Lógica de Programação a Construção de Algoritmos e Estrutura de Dados.** 3. ed. São Paulo: Pearson Prentice Hall, 2005. 220 p.

FORESTI, Henrique Braga. **Desenvolvimento de um robô bípede autônomo.** 2006. 139 f. Dissertação (Mestrado) - Curso de Engenharia Mecânica, Centro Tecnológico, Universidade Federal de Pernambuco, Recife, 2006.

GALIZA, André; GONÇALVES, Daniel; ALMEIDA, Isa. **PILHAS E BATERIAS:** Estudo da capacidade disponível para pilhas recarregáveis. 2014. 24 f. Dissertação (Mestrado) - Curso de Engenharia Eletrotécnica e de Computadores, Feup, Universidade do Porto, Porto, 2014. Cap. 1.

GOMES, Tássio José Gonçalves. **INTRODUÇÃO A PLATAFORMA ARDUINO.** 2016. Disponível em: <https://tassiogoncalves.com.br/wp-content/uploads/2016/08/Minicurso-de-Introdução-a-Plataforma-Arduino-2016.pdf>. Acesso em: 26 maio 2018.

GONZATTO, A.; SANTOS, C.; MELO, F.; RODRIGUES, G.; FARIA,J. **Óculos sonar para deficientes Visuais.** Disponível em < http://www.inicepg.univap.br/cd/INIC_2009/anais/arquivos/RE_0948_0818_01.pdf > acesso em 30 set. 2018.

IBGE, Coordenação de Trabalho e Rendimento. **Pesquisa nacional de saúde:2013:** Ciclos de vida Brasil e grandes regiões. Rio de Janeiro: Fiocruz, 2015. 93 p. Disponível em: <https://biblioteca.ibge.gov.br/visualizacao/livros/liv94522.pdf>. Acesso em: 18 mar. 2018.

LISBOA, Bárbara Texeira. **ROBÓTICA E MEDICINA.** 2010. Disponível em: <https://ufsj.edu.br/portal2-repositorio/File/orcv/materialdeestudo_roboticaemedicina.pdf>. Acesso em: 10 jun. 2018.

MERCADO, Livre. **Cabo 8 vias.** Disponível em: <https://produto.mercadolivre.com.br/MLB-713353905-cabo-de-rede-8-vias-4-pares-lan-cat-5e-caixa-300-metros--_JM>. Acesso em: 30 out. 2018.

MICHELINI, Aldo. **BATERIAS RECARREGÁVEIS:** Para Equipamentos Portáteis. São Paulo: S.t.a. – Sistemas e Tecnologia Aplicada Ind. Com. Ltda., 2017. 173 p.

MONTEIRO, André L.; SILVA, Eduardo M.; LOPES, Igor C. F. LOCALIZAÇÃO PARA DEFICIENTES VISUAIS UTILIZANDO UM GADGET PARA ORIENTAÇÃO. **Revista Innover,** Fortaleza- CE, v. 1, n. 4, p.93-105, 4 de dez 2014. Disponível em: <www2.ifma.edu.br/revistainnover/Nova/index.php/inicio/article/download/55/35>. Acesso em: 17 mar. 2018.

NUNES, Sylvia; LOMÔNACO, José Fernando Bitencourt. O aluno cego: preconceitos e potencialidades. **Revista Semestral da Associação Brasileira de Psicologia Escolar e Educacional**, São Paulo, v. 14, n. 1, p.55-64, jan-jun 2010. Disponível em: <http://www.scielo.br/pdf/pee/v14n1/v14n1a06>. Acesso em: 17 mar. 2018.

OTTONI, André Luiz Carvalho. **Introdução à Robótica.** 2010. Disponível em: <https://ufsj.edu.br/portal2-repositorio/File/orcv/materialdeestudo_introducaoarobotica.pdf>. Acesso em: 01 out. 2018.

RIBEIRO, Maria Isabel. **Uma Viagem ao Mundo dos Robots:** Ciclo de Colóquios Despertar para a Ciência. 2005. Disponível em: <http://users.isr.ist.utl.pt/~mir/pub/ViagemRobots-IsabelRibeiro05.pdf>. Acesso em: 20 mar. 2018.

RODRIGUES, Renato; GONÇALVES, José Correia. **Procedimentos de Metodologia Científica.** 8. ed. Lages: Papervest, 2017. 195 p.

ROMERO, Roseli Aparecida F.; PRESTES, Edson; OSÓRIO, Fernando; WOLF, Denis. 2014. **Robótica Móvel.** Rio de Janeiro: Grupo Editora Nacional, 2014. p. 303.

RUIC, G. **A história dos Robôs em imagens,** 6 abr 2012 Disponível em < https://exame.abril.com.br/ciencia/a-historia-dos-robos-em-imagens >. Acesso: 20 mar.2018.

SECCHI, Humberto A. 2008. **Uma Introdução aos Robôs Móveis.** Cynthia Netto de ALMEIDA e Felipe Nascimento MARTINS. [S.l.] : s.n., 2008. p. 81. Título original: **Una Introducción a los Robots Móviles**. Disponível em: < https:// www.aadeca.org/pdf/CP.../monografia_robot_movil.pdf > Acesso: 10 de jun. de 2018.

SECCHI, Humberto A. 2008**. Uma Introdução aos Robôs Móveis**. [trad.] Cynthia Netto de ALMEIDA e Felipe Nascimento MARTINS. [S.l.] : s.n., 2008. p. 81. Título original: Una Introducción a los Robots Móviles.

SOUZA, Fábio. **Arduino Mega 2560.** Disponível em: <https://www.embarcados.com.br/arduino-mega-2560/>. Acesso em: 14 de mar. de 2018.

SOUZA, J. A. M. Felippe de. **Introdução aos robôs.** Disponível em: <http://webx.ubi.pt/~felippe/texts5/robotica_cap1.pdf>. Acesso em: 01 out. 2018.

STA, Sistemas e Tecnologias Aplicadas. **A história das baterias.** Disponível em: <http://www.sta-eletronica.com.br/artigos/a-historia-das-baterias>. Acesso em: 21 out. 2018.

THOMSEN, Adilson. **Como conectar o Sensor Ultrassônico HC-SR04 ao Arduino.** Disponível em: <https://www.filipeflop.com/blog/sensor-ultrassonico-hc-sr04-ao-arduino/>. Acesso em: 14 mar. 2018.

TOWER Pro. **Servo Motor MG 946R.** 2018. Disponível em: <http://www.towerpro.com.tw/product/mg946r/>. Acesso em: 26 maio 2018.

UMEZAKI, Gustavo Corrêa. **PROJETO E IMPLEMENTAÇÃO DE ROBÔ MÓVEL OMNIDIRECIONAL APLICADO A ENXAME DE ROBÔS.** 2017. 73 f. TCC (Graduação) - Curso de Engenharia Mecatrônica, Universidade Federal de Uberlândia, Uberlândia, 2017. Cap. 2.

Apêndice A

Código fonte do protótipo em linguagem C/C++.

Esse código é que vai permitir que o robô faça todas as suas funções, e análise todas as condições programadas, para que o usuário possa utilizar com segurança e possa orienta –ló no seu dia a dia.

```
// inclusão de bibliotecas.

#include <Servo.h>     // Biblioteca para os Servos Motores.
#include <Ultrasonic.h> // Biblioteca para os Sensores Ultrassônicos

//Definindo os pinos
Servo servo_sonar; // Nome do servo motor.

int ang = 90; // Servo motor inicia em 90°.
int avanco = 10; // Vai fazer a contagem 10 em 10.

// Variável do Botão Direita.
int botaoD = 36;   // escolha do pino do botão da direita.
int estado = 0;

// Variável do Botão Direita.
int botaoE = 38;   // escolha do pino do botão da esquerda.
int estado1 = 0;

// Variáveis dos motores DC's.

// Motor A
#define IN1 3 // Sentido frente.
#define IN2 4 // Sentido para trás.
#define motorA  2 // Pino que está ligado o eneble.
```

```
#define velocidade_motorA  120 // Velocidade dor motor A.

// Motor B
#define IN3  6   // Sentido para trás.
#define IN4  7   // Sentido frente.
#define motorB 5  //Pino que está ligado o eneble.
#define velocidade_motorB  120// Velocidade do motor B.

// Variáveis do Sensor Ultrassônico Frontal.
#define trigPin A0 //Pino TRIG do sensor no pino analógico A0.
#define echoPin A1 //Pino ECHO do sensor no pino analógico A1.

// Variáveis do Sensor Ultrassônico de Chão.
#define trigPin2 A6 //Pino TRIG do sensor no pino analógico A6.
#define echoPin2 A7 //Pino ECHO do sensor no pino analógico A7.

// Variáveis do Sensor Ultrassônico Superior.
#define trigPin3 A8 //Pino TRIG do sensor no pino analógico A8.
#define echoPin3 A12 // Pino Echo dos sensor no pino analógico A12.

// Variáveis dos Motores Vibracais
#define motor1_sensor_frente 26 // Pino Digital Que Está Ligado Motor Vibracall Número 1.
#define motor2_sensor_buraco 28 // Pino Digital Que Está Ligado Motor Vibracall Número 2.
#define motor3_sensor_superior 30 // Pino Digital Que Está Ligado Motor Vibracall Número
3.
#define motor4_sensor_direita 32 // Pino Digital Que Está Ligado Motor Vibracall Número 4.
#define motor5_sensor_esquerda 34 // Pino Digital Que Está Ligado Motor Vibracall Número
5.

// Variáveis do Sensor Ultrassônico Frontal.
++long duracao;
long distancia_cm = 0;
int minimumRange = 5; //tempo de resposta do sensor.
int maximumRange = 200;
```

```
int distancia_obstaculo = 5; // Distância que o sensor vai detectar obstáculo de frente.

// Variáveis do Sensor Ultrassônico Chão.
long duracao2;
long distancia_cm2 = 0;
int minimumRange2 = 5; //tempo de resposta do sensor.
int maximumRange2 = 200;
int distancia_obstaculo2 = 15; // Distância que o sensor vai detectar obstáculo de chão.

// Variáveis do Sensor Ultrassônico Superior.

long duracao3;
long distancia_cm3 = 0;
int minimumRange3 = 5; //tempo de resposta do sensor.
int maximumRange3 = 200;
int distancia_obstaculo3 = 5; // Distância que o sensor vai detectar obstáculo Superior

// Executando inicialização no arduino.
void setup() {

  Serial.begin(9600);
  // Serial.println("teste");
  servo_sonar.attach(9);  // Servo motor ligado no pino digital 9.

  // Entradas dos Botôes.
  pinMode(botaoD, INPUT);
  pinMode(botaoE, INPUT);

  // Entrada e Saídas dos Motores DC's.
  pinMode(IN1, OUTPUT);   // Define o pino IN1 como saída.
  pinMode(IN2, OUTPUT);   // Define o pino IN2 como saída.
  pinMode(IN3, OUTPUT);   // Define o pino IN3 como saída.
  pinMode(IN4, OUTPUT);   // Define o pino IN4 como saída.
```

```
  // Entradas e Saídas dos Sensores Ultrassônicos.
  pinMode(trigPin, OUTPUT); //Define o pino TRIG como saída.
  pinMode(echoPin, INPUT);  //Define o pino ECHO como entrada.

  pinMode(trigPin2, OUTPUT); //Define o pino TRIG como saída.
  pinMode(echoPin2, INPUT);  //Define o pino ECHO como entrada.

  pinMode(trigPin3, OUTPUT); //Define o pino TRIG como saída.
  pinMode(echoPin3, INPUT);  //Define o pino ECHO como entrada.

  // Entrada e Saídas dos Motores Vibracais.
  pinMode(motor1_sensor_frente, OUTPUT);
  pinMode(motor2_sensor_buraco, OUTPUT);
  pinMode(motor3_sensor_superior, OUTPUT);
  pinMode(motor4_sensor_direita, OUTPUT);
  pinMode(motor5_sensor_esquerda, OUTPUT);

}

// Função para ler e calcular a distância do sensor ultrassônico Frontal.

int  leitura_frontal() {
  digitalWrite(trigPin, LOW); //não envia som.
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH); //envia som.
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW); //não envia o som e espera o retorno do som enviado.
  duracao = pulseIn(echoPin, HIGH); //Captura a duração em tempo do retorno do som.
  distancia_cm = duracao / 58; //Calcula a distância.
  delay(30);
  Serial.println(distancia_cm);
  return distancia_cm;    // Retorna a distância.

}
```

```
// Função para ler e calcular a distância do sensor ultrassônico Chão.

int  leitura_chao() {
  digitalWrite(trigPin2, LOW); //não envia som
  delayMicroseconds(2);
  digitalWrite(trigPin2, HIGH); //envia som
  delayMicroseconds(10);
  digitalWrite(trigPin2, LOW); //não envia o som e espera o retorno do som enviado
  duracao2 = pulseIn(echoPin2, HIGH); //Captura a duração em tempo do retorno do som.
  distancia_cm2 = duracao2 / 58; //Calcula a distância
  Serial.println(distancia_cm2);
  delay(30);
  return distancia_cm2;  // Retorna a distância
}

// Função para ler e calcular a distância do sensor ultrassônico Superior.

int  leitura_superior() {
  digitalWrite(trigPin3, LOW); //não envia som.
  delayMicroseconds(2);
  digitalWrite(trigPin3, HIGH); //envia som.
  delayMicroseconds(10);
  digitalWrite(trigPin3, LOW); //não envia o som e espera o retorno do som enviado.
  duracao3 = pulseIn(echoPin3, HIGH); //Captura a duração em tempo do retorno do som.
  distancia_cm3 = duracao3 / 58; //Calcula a distância.
  delay(30);
  Serial.println(distancia_cm3);
  return distancia_cm3;    // Retorna a distância.

}

// Função principal do Arduino.
void loop() {
```

```
  sonar();
  botao_esquerda();
  botao_direita();
  ang = ang + avanco;
  if (ang > 180) avanco = -10; // Vai fazer a contagem 10 em 10.
  if (ang < 0) avanco = 10;  // Vai fazer a contagem 10 em 10.
  servo_sonar.write(ang);
  Serial.println(ang);
}

void sonar() {

  analogWrite( motorA, velocidade_motorA);
  analogWrite( motorB, velocidade_motorB);

  // Laço para detectar obstáculos frontal.

  int distancia = leitura_frontal(); //Vai fazer comparação da distância do obstáculo.
  if (distancia > distancia_obstaculo) {  // Se a distância for maior que 30 cm.
    robo_frente(); //Robô anda para frente.

  }

  else {
    vibra1(); // Liga Motor vibra Número 1.
    digitalWrite (motor1_sensor_frente, LOW); // Desliga Motor Vibra Número1.
    Serial.println("FRONTAL");
    robo_parado();  // Robô parado.
    calculando_trajeto(); // Verificação dos obstáculos no alcance de 0° à 180°.
    robo_direita();
    robo_frente(); // Robo anda para frente.

  }
```

```
  //Laço para detectar obstáculos de chão.

  int distancia2 = leitura_chao(); // Vai fazer comparação da distancia do obstáculo de Chão.
  if (distancia2 < distancia_obstaculo2) {  // Se a distância for maior que 8 cm.
    robo_frente();

  }
  else {
    vibra2(); // Liga Motor vibra Número 2.
    Serial.println("BURACO");
    digitalWrite(motor2_sensor_buraco, LOW);

    robo_parado(); //Robô anda parado.
    robo_da_re();  // Robô da ré por 2 segundos.
    delay(500);
    robo_parado(); //Robô anda parado.
    robo_direita(); // Robô vira a direita.
    robo_frente();  // Robô anda para frente.

  }

  //Laço para detectar obstáculos Superior.

  int distancia3 = leitura_superior(); //Vai fazer comparação da distancia do obstáculo de
Superior.
  if (distancia3 > distancia_obstaculo3) { // Se a distância for maior que 1.40 cm.
    robo_frente();

  }
  else {
    Serial.println("SUPERIOR");
    vibra3(); // Liga Motor vibra Número 3.
    digitalWrite(motor3_sensor_superior, LOW);
    robo_parado();  // Robô parado.
```

```
    calculando_trajeto(); // Verificação dos obstáculos no alcance de 0° à 180°.
    robo_direita();
    robo_frente(); // Robo anda para frente.

  }

}
// Laço que o robo anda pra frente.
void robo_frente() {

  digitalWrite (IN1, HIGH);
  digitalWrite (IN4, HIGH);
}

// Laço que o robo fica parado.
void robo_parado() {
  digitalWrite (IN1, LOW);
  digitalWrite (IN2, LOW);
  digitalWrite (IN3, LOW);
  digitalWrite (IN4, LOW);

  delay(2000);
}

// Laço que o robo anda para atrás.
void robo_da_re() {

  digitalWrite (IN1, LOW);
  digitalWrite (IN2, HIGH);
  digitalWrite (IN3, HIGH);
  digitalWrite (IN4, LOW);

}
```

```
// Laço que o robo vira a direita.
void robo_direita() {

  digitalWrite (IN1, LOW);
  digitalWrite (IN4, HIGH);
  delay(100);
}

// Laço que o robo vira a esquerda.
void robo_esquerda() {

  digitalWrite (IN1, HIGH);
  digitalWrite (IN4, LOW);

}

// Função em que o robo verifica os obstáculos conforme o movimento do servo motor no alcance de 0° à 180°.
void calculando_trajeto () {

  if (ang < 90 && ang > 0) { // Se o obstáculo estiver entre o ângulo de 0° à 90°, robo vai virar a esquerda e retorna o seu caminho.
    robo_esquerda();
    delay(500);
  }
  else  if (ang < 180 && ang > 90) { // Se o obstáculo estiver entre o ângulo de 180° à 90°, robo vai virar a direita e retorna o seu caminho.
    robo_direita();
    delay(500);
  }

  //sonar();
}
```

```
// Laço do Sensor Vibracall Número 1.
void vibra1() {
  digitalWrite (motor1_sensor_frente, HIGH);
  delay(2000);
}

// Laço do Sensor Vibracall Número 2.
void vibra2() {
  digitalWrite (motor2_sensor_buraco, HIGH);
  delay(2000);
}

// Laço do Sensor Vibracall Número 3.
void vibra3() {
  digitalWrite (motor3_sensor_superior, HIGH);
  delay(2000);
}

// Laço do Sensor Vibracall Número 4.
void vibra4() {
  digitalWrite (motor4_sensor_direita, HIGH);
  delay(2000);
}

// Laço do Sensor Vibracall Número 5.
void vibra5() {
  digitalWrite (motor5_sensor_esquerda, HIGH);
  delay(2000);
}

// Laço do Botão da direita.

void botao_direita() {
  estado = digitalRead(botaoD); // ler o valor de entrada
```

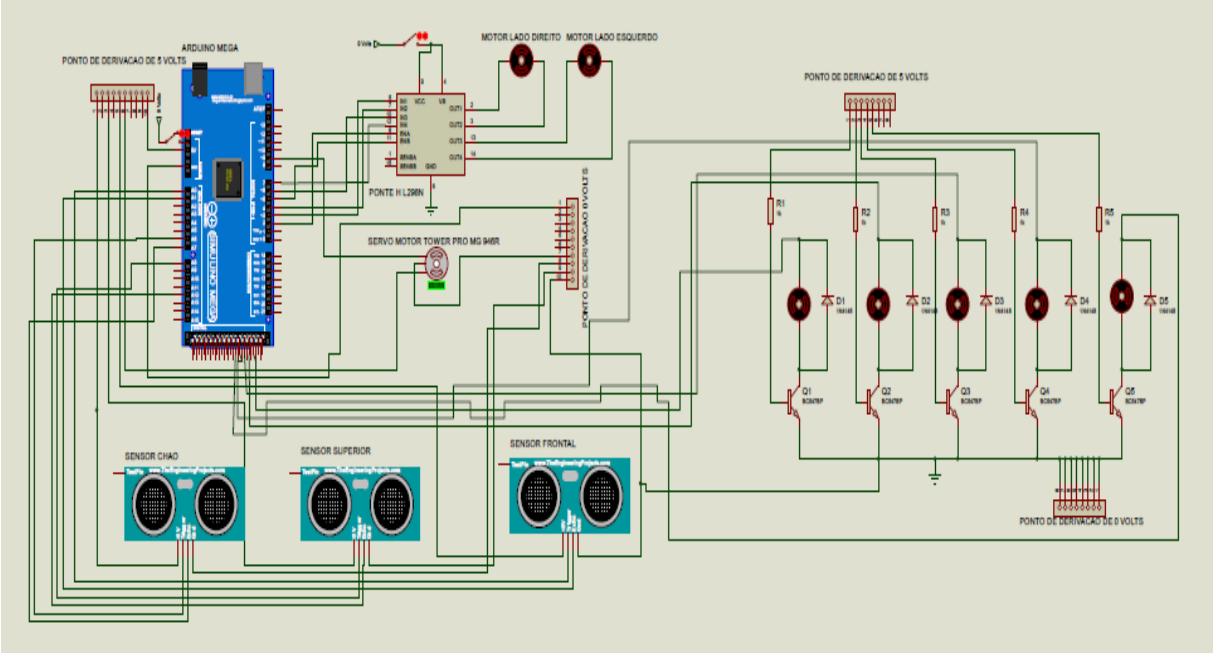
```
  if (estado == HIGH) {         // verificar se a entrada é HIGH (interruptor livre)
    vibra4(); // Liga Motor vibra Número 4.
    digitalWrite(motor4_sensor_direita, LOW);
    robo_direita();
  }
  else {
    robo_frente();
  }
}

// Laço do Botão da Esquerda.

void botao_esquerda() {
  estado1 = digitalRead(botaoE); // ler o valor de entrada
  if (estado1 == HIGH) {
    vibra5(); // Liga Motor vibra Número 5.
    digitalWrite(motor5_sensor_esquerda, LOW);
    robo_esquerda();
  }
  else {
    robo_frente();
  }
}
```

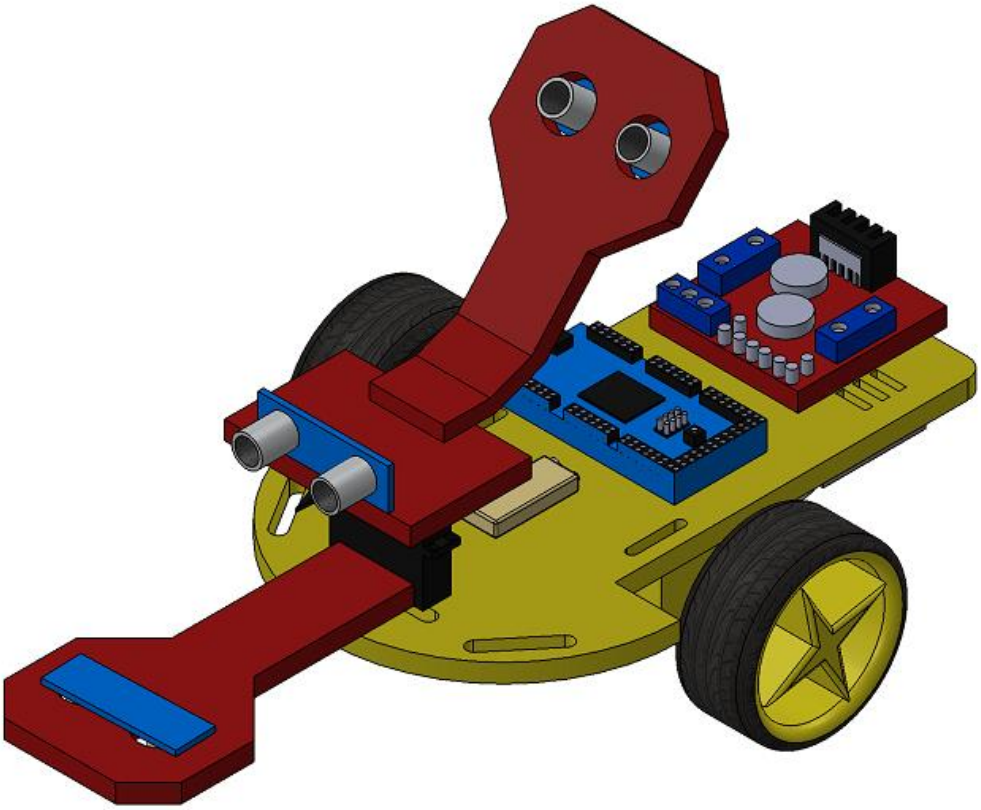
Apêndice B

Diagrama do circuito elétrico do robô.



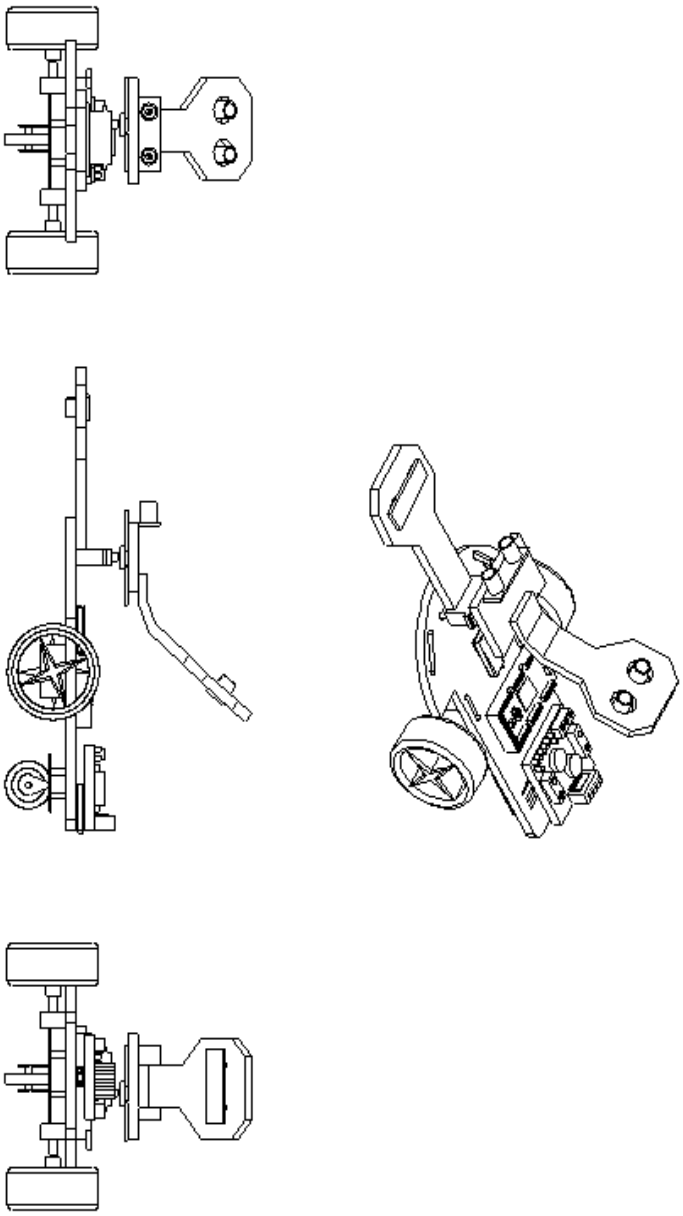
Apêndice C

Projeto do robô desenvolvido no software Solid Works.

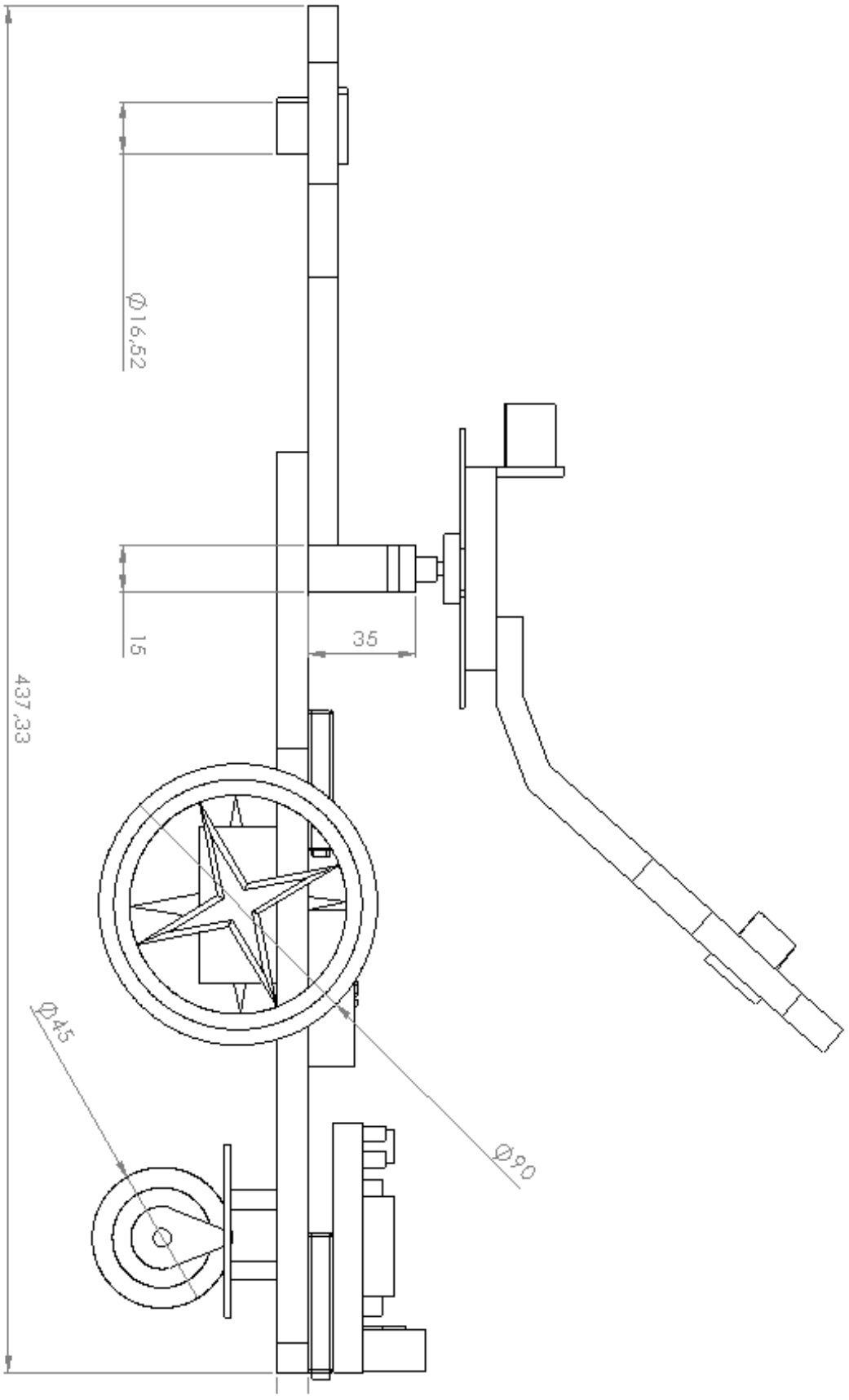


Apêndice D

Desenho do protótipo em vistas.

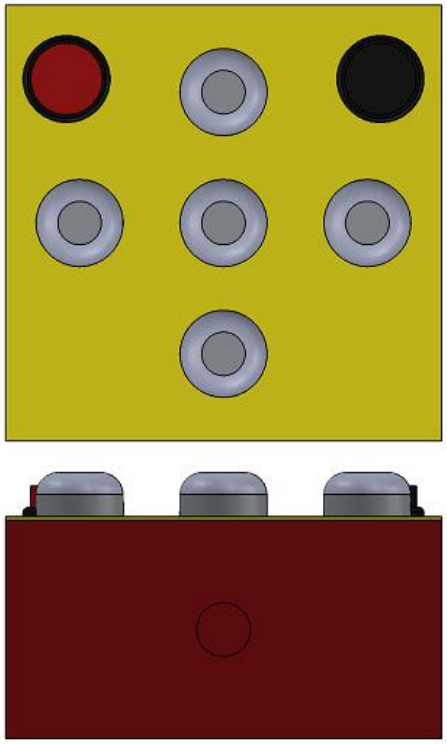


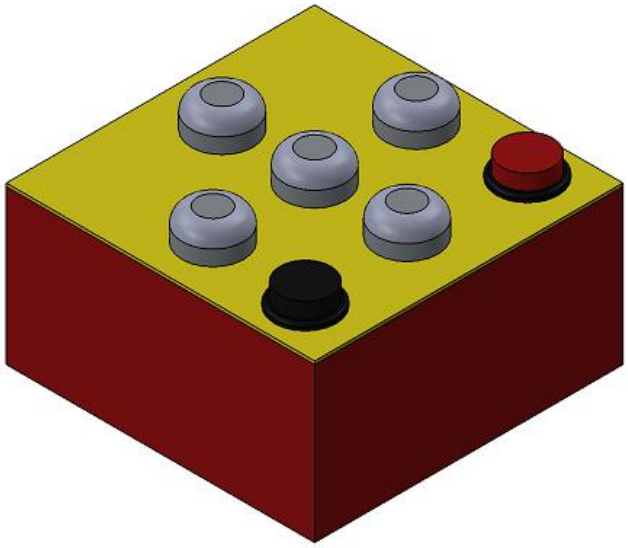
Apêndice E



Apêndice F

Estrutura da caixa e tampa, do controle de mão feito no Solid Works, a sua construção foi realizada na impressora 3D.





ANEXOS

Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 135 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz
  - On-Chip 2-cycle Multiplier
- Non-volatile Program and Data Memories
  - 64K/128K/256K Bytes of In-System Self-Programmable Flash
    Endurance: 10,000 Write/Erase Cycles
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - 4K Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 8K Bytes Internal SRAM
  - Up to 64K Bytes Optional External Memory Space
  - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - Four 16-bit Timer/Counter with Separate Prescaler, Compare- and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four 8-bit PWM Channels
  - Six/Twelve PWM Channels with Programmable Resolution from 2 to 16 Bits (ATmega1281/2561, ATmega640/1280/2560)
  - Output Compare Modulator
  - 8/16-channel, 10-bit ADC
  - Two/Four Programmable Serial USART (ATmega1281/2561,ATmega640/1280/2560)
  - Master/Slave SPI Serial Interface
  - Byte Oriented 2-wire Serial Interface
  - 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
  - 51/86 Programmable I/O Lines (ATmega1281/2561, ATmega640/1280/2560)
  - 64-lead (ATmega1281/2561)
  - 100-lead (ATmega640/1280/2560)
  - 100-lead TQFP (64-lead TQFP Option)
- Temperature Range:
  - -40°C to 85°C Industrial
- Speed Grade:
  - ATmega1281/2561V/ATmega640/1280/2560V:
    0 - 4 MHz @ 1.8 - 5.5V, 0 - 8 MHz @ 2.7 - 5.5V
  - ATmega640/1280/1281/2560/2561:
    0 - 8 MHz @ 2.7 - 5.5V, 0 - 16 MHz @ 4.5 - 5.5V

8-bit AVR® Microcontroller with 256K Bytes In-System Programmable Flash

ATmega1281/2561/V ATmega640/1280/2560/V

Advance Information

2549A–AVR–03/05

Pin Configurations

Figure 1. Pinout ATmega640/1280/2560

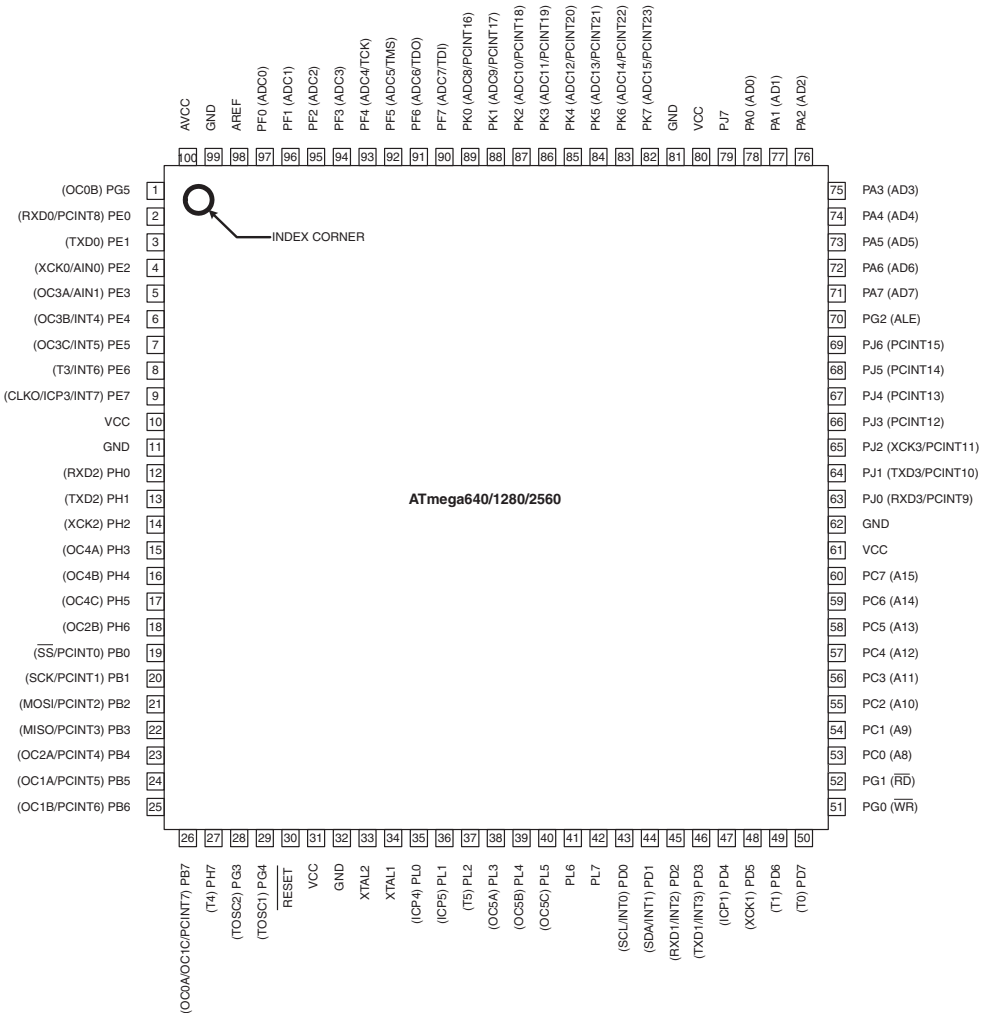
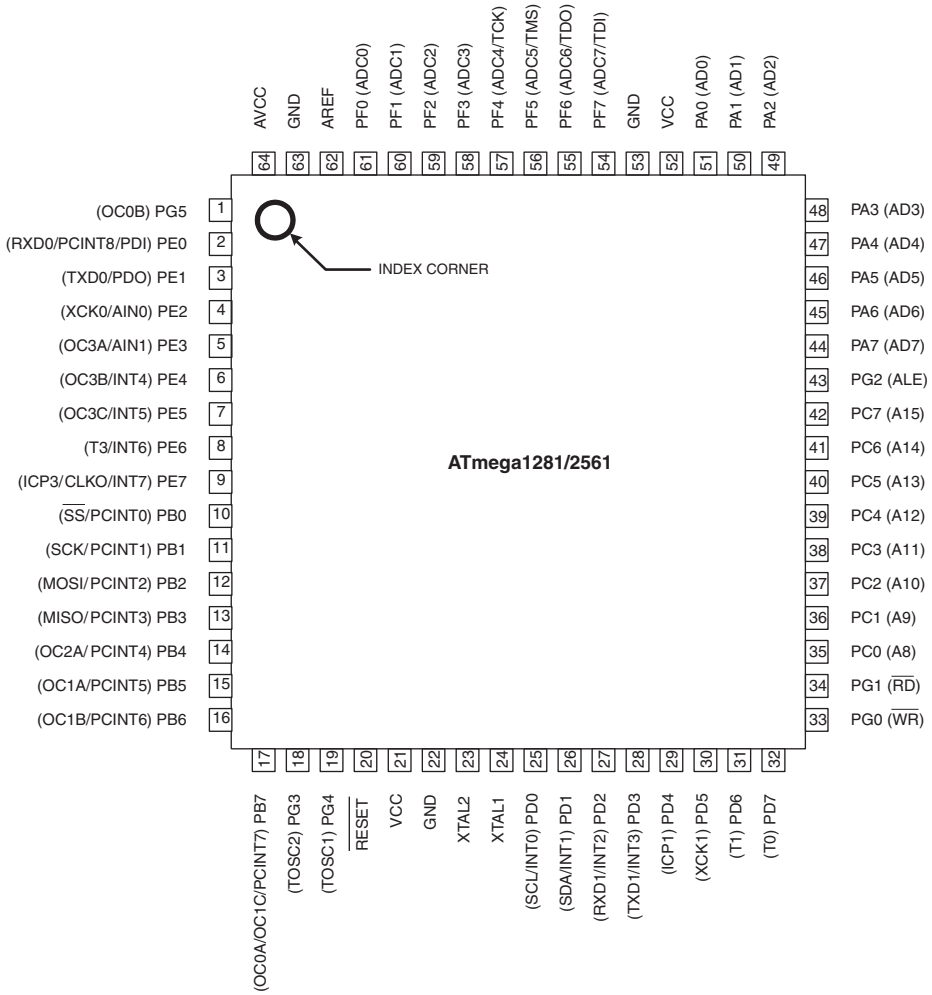


Figure 2. Pinout ATmega1281/2561



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

Disclaimer

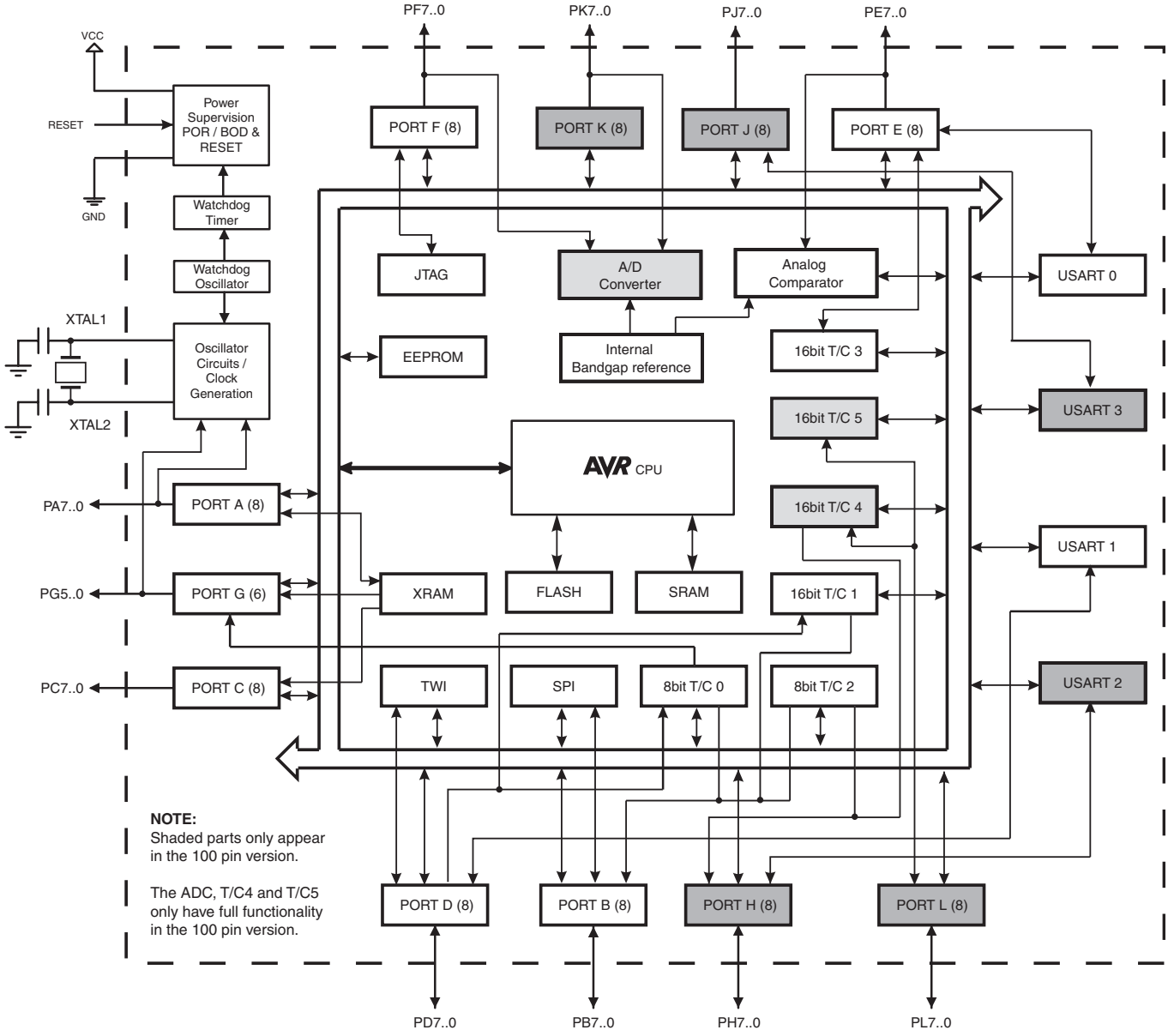
Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

Overview

The ATmega640/1280/1281/2560/2561 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 ATmega640/1280/1281/2560/2561 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 3. 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 ATmega640/1280/1281/2560/2561 provides the following features: 64K/128K/256K bytes of In-System Programmable Flash with Read-While-Write capabilities, 4K bytes EEPROM, 8K bytes SRAM, 54/86 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), six flexible Timer/Counters with compare modes and PWM, 4 USARTs, a byte oriented 2-wire Serial Interface, a 16-channel, 10-bit ADC with optional differential input stage with programmable gain, programmable Watchdog Timer with Internal Oscillator, an SPI serial port, IEEE std. 1149.1 compliant JTAG test interface, also used for accessing the On-chip Debug system and programming and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer 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. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high-density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile 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 ATmega640/1280/1281/2560/2561 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega640/1280/1281/2560/2561 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.

Comparison Between ATmega1281/2561 and ATmega640/1280/2560

Each device in the ATmega640/1280/1281/2560/2561 family differs only in memory size and number of pins. Table 1 summarizes the different configurations for the six devices.

Table 1. Configuration Summary

Device	Flash	EEPROM	RAM	General Purpose I/O pins	16 bits resolution PWM channels	Serial USARTs	ADC Channels
ATmega640	64KB	4KB	8KB	86	12	4	16
ATmega1280	128KB	4KB	8KB	86	12	4	16
ATmega1281	128KB	4KB	8KB	54	6	2	8
ATmega2560	256KB	4KB	8KB	86	12	4	16
ATmega2561	256KB	4KB	8KB	54	6	2	8

Pin Descriptions

VCC Digital supply voltage.

GND Ground.

Port A (PA7..PA0) Port A is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port A also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 88.

Port B (PB7..PB0) 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.

Port B has better driving capabilities than the other ports.

Port B also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 89.

Port C (PC7..PC0) Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C 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.

Port C also serves the functions of special features of the ATmega640/1280/1281/2560/2561 as listed on page 92.

Port D (PD7..PD0) 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.

Port D also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 94.

Port E (PE7..PE0)

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

Port E also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 96.

Port F (PF7..PF0)

Port F serves as analog inputs to the A/D Converter.

Port F also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port F output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port F pins that are externally pulled low will source current if the pull-up resistors are activated. The Port F pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PF7(TDI), PF5(TMS), and PF4(TCK) will be activated even if a reset occurs.

Port F also serves the functions of the JTAG interface.

Port G (PG5..PG0)

Port G is a 6-bit I/O port with internal pull-up resistors (selected for each bit). The Port G output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port G pins that are externally pulled low will source current if the pull-up resistors are activated. The Port G pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port G also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 102.

Port H (PH7..PH0)

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

Port H also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 104.

Port J (PJ7..PJ0)

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

Port J also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 106.

Port K (PK7..PK0)

Port K serves as analog inputs to the A/D Converter.

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

Port K also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 108.

Port L (PL7..PL0)

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

Port L also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 110.

$\overline{\text{RESET}}$

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 23 on page 58. Shorter pulses are not guaranteed to generate a reset.

XTAL1

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting Oscillator amplifier.

AVCC

AVCC is the supply voltage pin for Port F and the A/D Converter. 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.

AREF

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

About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

These code examples assume that the part specific header file is included before compilation. For I/O registers located in extended I/O map, "IN", "OUT", "SBIS", "SBIC", "CBI", and "SBI" instructions must be replaced with instructions that allow access to extended I/O. Typically "LDS" and "STS" combined with "SBRS", "SBRC", "SBR", and "CBR".

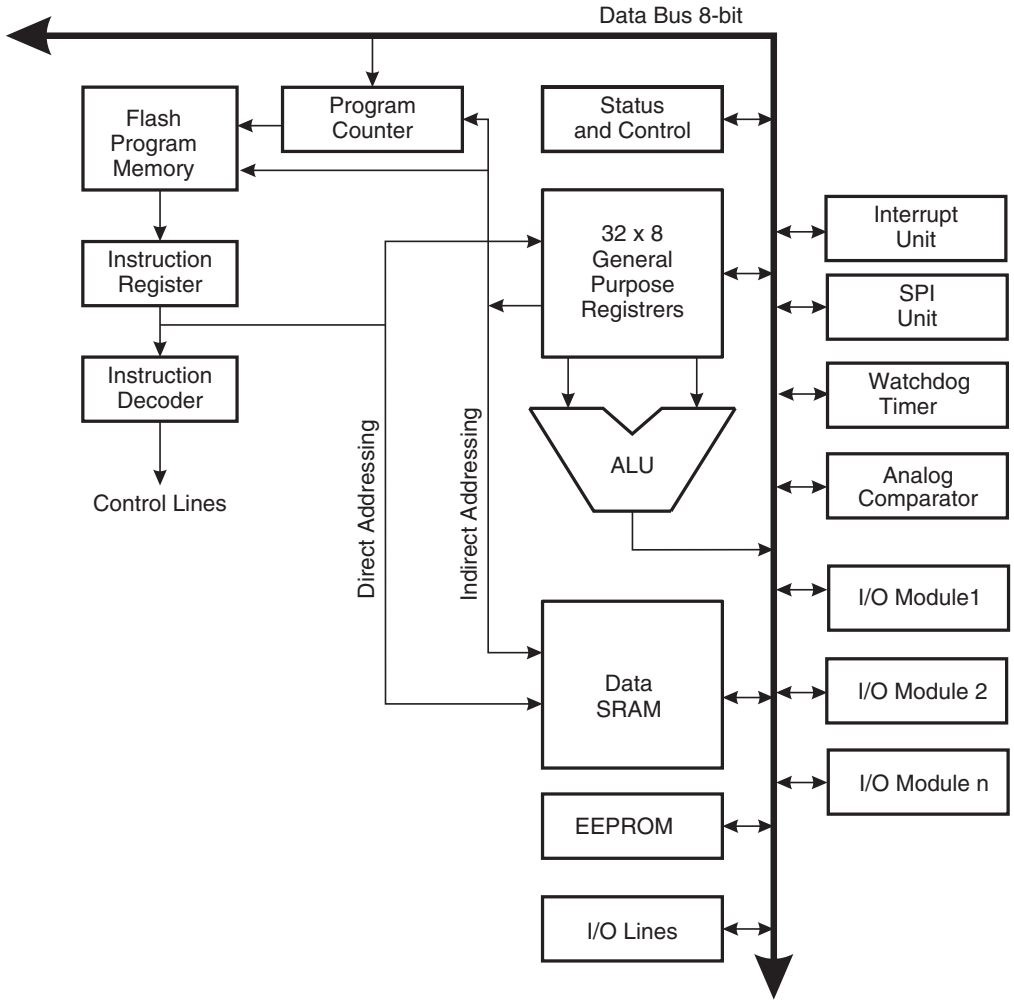
AVR CPU Core

Introduction

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

Architectural Overview

Figure 4. Block Diagram of the AVR Architecture



In order to maximize performance and parallelism, the AVR uses a Harvard architecture – with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory.

The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File,

the operation is executed, and the result is stored back in the Register File – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

Program Flash memory space is divided in two sections, the Boot Program section and the Application Program section. Both sections have dedicated Lock bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot Program section.

During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the Reset routine (before subroutines or interrupts are executed). The Stack Pointer (SP) is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional Global Interrupt Enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority.

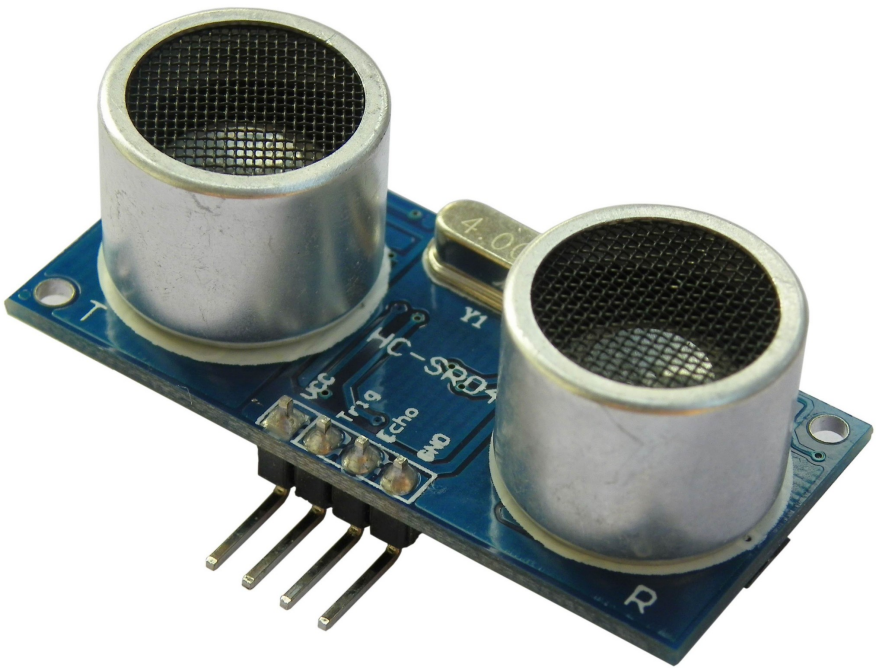
The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the Register File, 0x20 - 0x5F. In addition, the ATmega640/1280/1281/2560/2561 has Extended I/O space from 0x60 - 0x1FF in SRAM where only the ST/STS/STD and LD/LDS/LDD instructions can be used.

ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the “Instruction Set” section for a detailed description.

Status Register

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the Status Register is updated after all ALU operations, as specified in the Instruction Set Reference. This will in many cases



User's Manual

V1.0

May 2013

Index

1.0 INTRODUCTION

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats or dolphins do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 400 cm or 1" to 13 feet. It operation is not affected by sunlight or black material like Sharp rangefinders are (although acoustically soft materials like cloth can be difficult to detect). It comes complete with ultrasonic transmitter and receiver module.

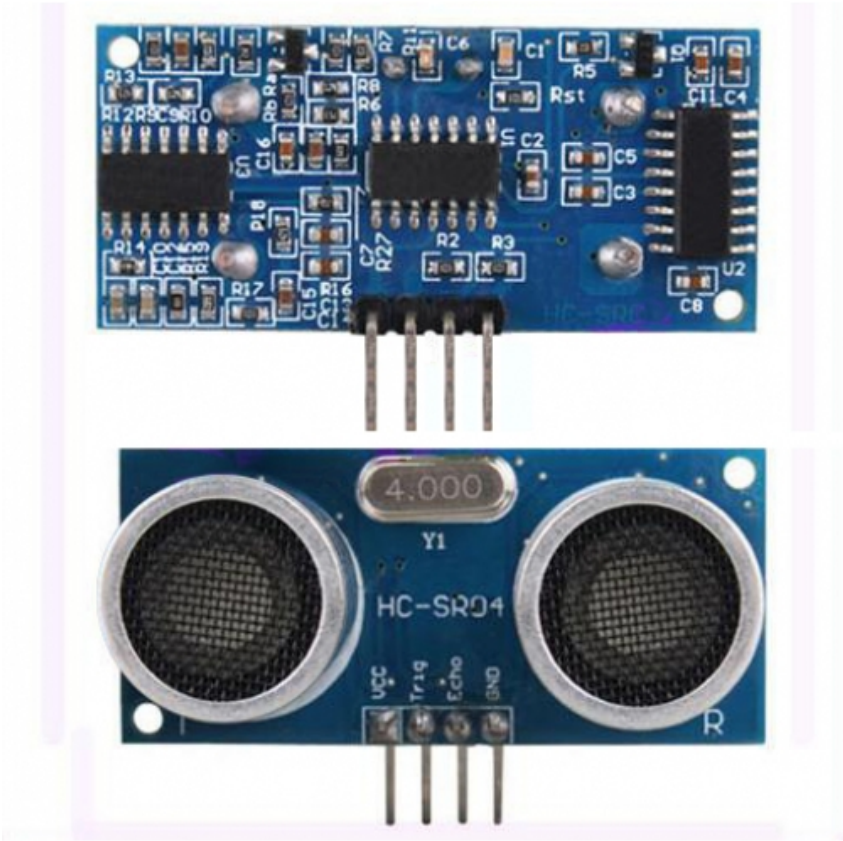
Features:

- Power Supply :+5V DC
- Quiescent Current : <2mA
- Working Currnt: 15mA
- Effectual Angle: <15°
- Ranging Distance : 2cm – 400 cm/1" - 13ft
- Resolution : 0.3 cm
- Measuring Angle: 30 degree
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm

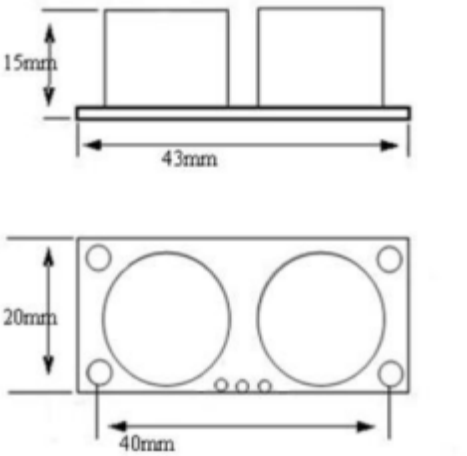
2.0 PACKING LIST

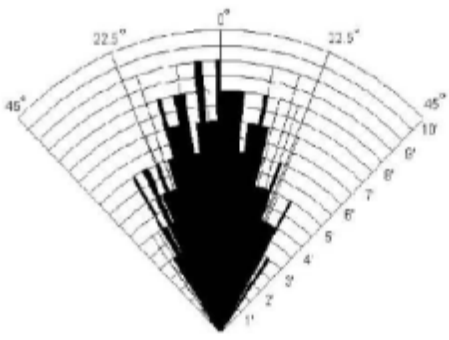
1. 1 x HC-SR04 module

3.0 PRODUCT LAYOUT



VCC = +5VDC
Trig = Trigger input of Sensor
Echo = Echo output of Sensor
GND = GND





Practical test of performance,
Best in 30 degree angle

4.0 PRODUCT SPECIFICATION AND LIMITATIONS

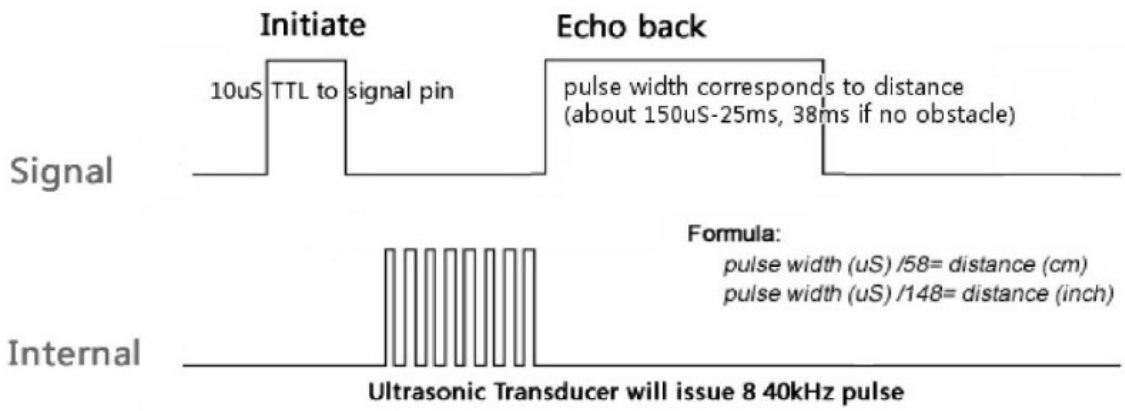
Parameter	Min	Typ.	Max	Unit
Operating Voltage	4.50	5.0	5.5	V
Quiescent Current	1.5	2	2.5	mA
Working Current	10	15	20	mA
Ultrasonic Frequency	-	40	-	kHz

5.0 OPERATION

The timing diagram of HC-SR04 is shown. To start measurement, Trig of SR04 must receive a pulse of high (5V) for at least 10us, this will initiate the sensor will transmit out 8 cycle of ultrasonic burst at 40kHz and wait for the reflected ultrasonic burst. When the sensor detected ultrasonic from receiver, it will set the Echo pin to high (5V) and delay for a period (width) which proportion to distance. To obtain the distance, measure the width (Ton) of Echo pin.

Time = Width of Echo pulse, in uS (micro second)

- Distance in centimeters = Time / 58
- Distance in inches = Time / 148
- Or you can utilize the speed of sound, which is 340m/s

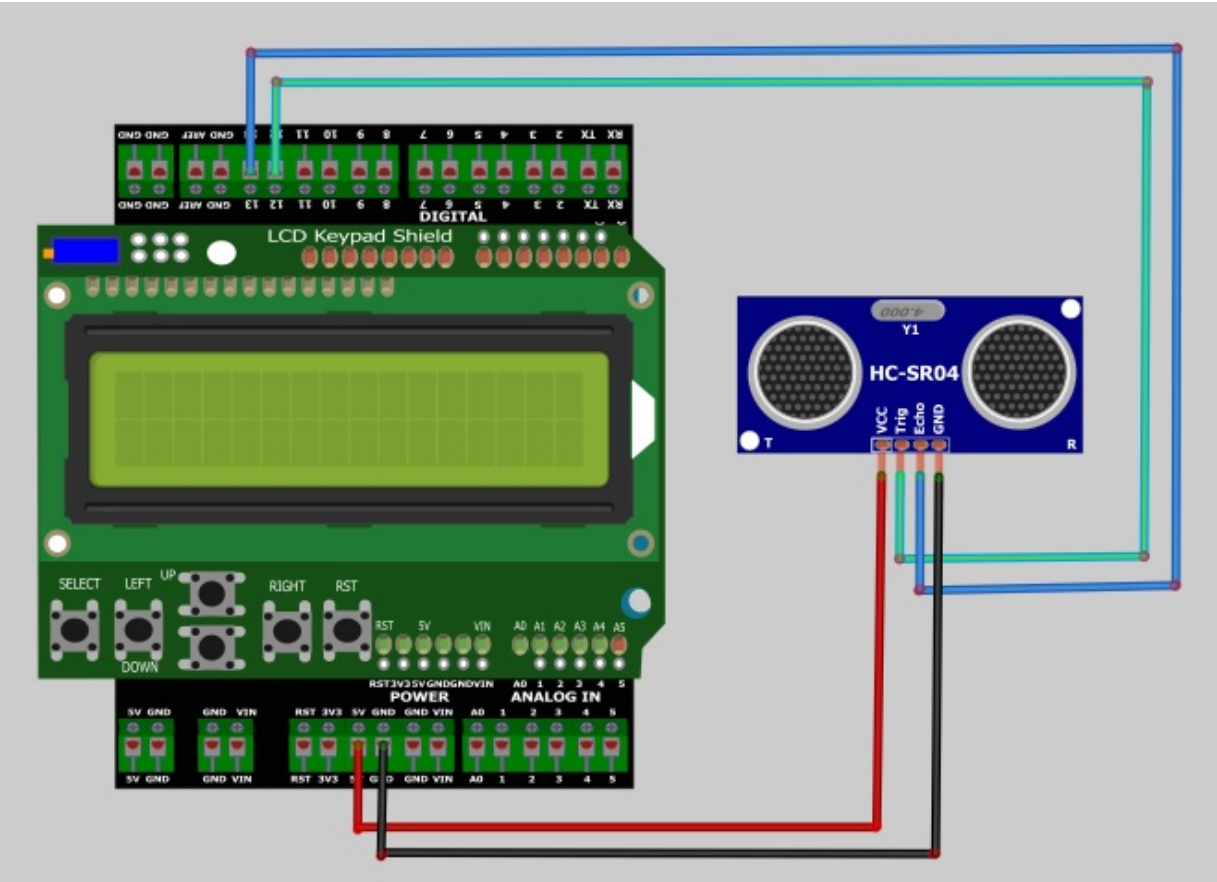


Note:

- Please connect the GND pin first before supplying power to VCC.
- Please make sure the surface of object to be detect should have at least 0.5 meter^2 better performance.

6.0 HARDWARE INTERFACE

Here is example connection for Ultrasonic Ranging module to Arduino UNO board. It can be interface with any microcontroller with digital input such as PIC, SK40C, SK28A, SKds40A, Arduino series.



7.0 EXAMPLE CODE

This is example code Ultrasonic Ranging module. Please download the complete code at the product page.

```
#include "Ultrasonic.h"
#include <LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
Ultrasonic ultrasonic(12,13);

void setup() {
lcd.begin(16, 2);
lcd.setCursor(0, 0);
lcd.print("HC-SR4 testing..");
delay(1000);

}

void loop()
{
  //lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print(ultrasonic.Ranging(CM));
  lcd.print("cm  ");

  delay(100);
}
```

8.0 WARRANTY

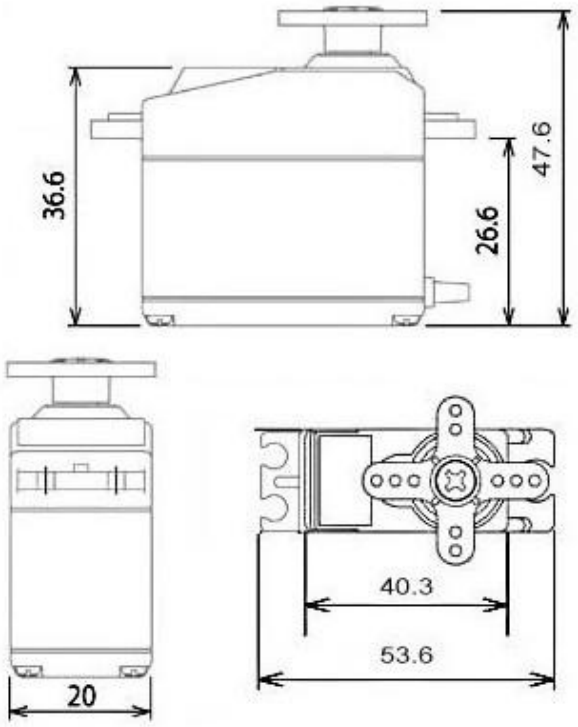
- Product warranty is valid for 6 months.
- Warranty only applies to manufacturing defect.
- Damaged caused by miss-use is not covered under warranty
- Warranty does not cover freight cost for both ways.

Prepared by
Cytron Technologies Sdn. Bhd.
19, Jalan Kebudayaan 1A,
Taman Universiti,
81300 Skudai,
Johor, Malaysia.

Tel: *+607-521 3178*
Fax: *+607-521 1861*

URL: www.cytron.com.my
Email: support@cytron.com.my
sales@cytron.com.my

MG996R High Torque
Metal Gear Dual Ball Bearing Servo



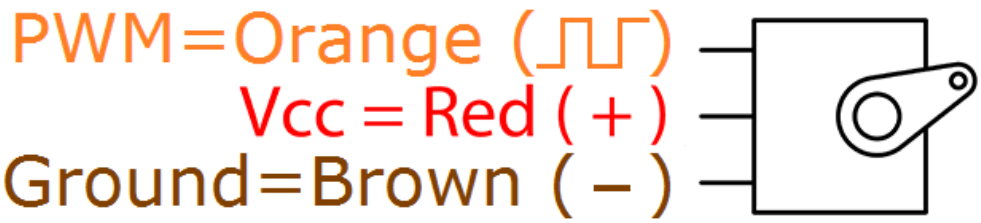
This High-Torque MG996R Digital Servo features metal gearing resulting in extra high 10kg stalling torque in a tiny package. The MG996R is essentially an upgraded version of the famous MG995 servo, and features upgraded shock-proofing and a redesigned PCB and IC control system that make it much more accurate than its predecessor. The gearing and motor have also been upgraded to improve dead bandwith and centering. The unit comes complete with 30cm wire and 3 pin 'S' type female header connector that fits most receivers, including Futaba, JR, GWS, Cirrus, Blue Bird, Blue Arrow, Corona, Berg, Spektrum and Hitec.

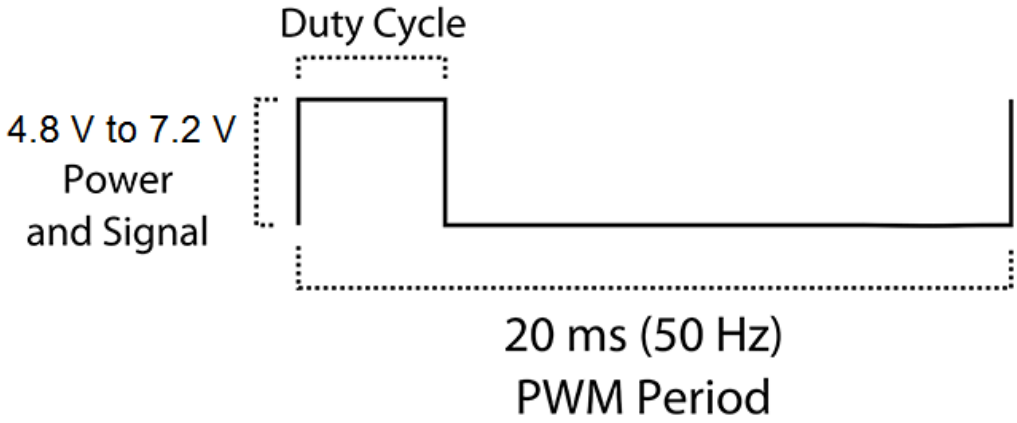
This high-torque standard servo can rotate approximately 120 degrees (60 in each direction). You can use any servo code, hardware or library to control these servos, so it's great for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. The MG996R Metal Gear Servo also comes with a selection of arms and hardware to get you set up nice and fast!

Specifications

- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.
- Stall torque: 9.4 kgf·cm (4.8 V), 11 kgf·cm (6 V)
- Operating speed: 0.17 s/60º (4.8 V), 0.14 s/60º (6 V)

- Operating voltage: 4.8 V a 7.2 V
- Running Current 500 mA – 900 mA (6V)
- Stall Current 2.5 A (6V)
- Dead band width: 5 µs
- Stable and shock proof double ball bearing design
- Temperature range: 0 ºC – 55 ºC

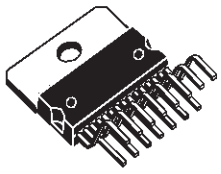




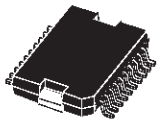
L298

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)



Multiwatt15



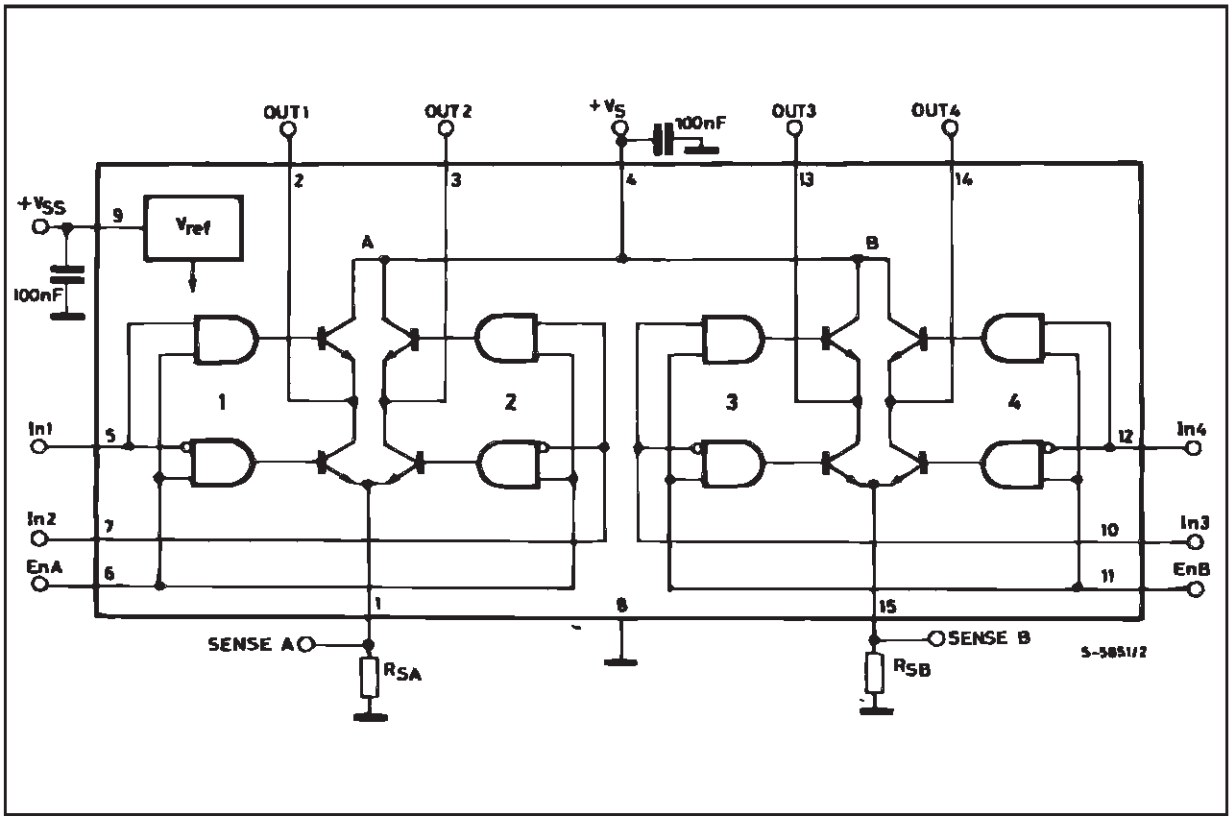
PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

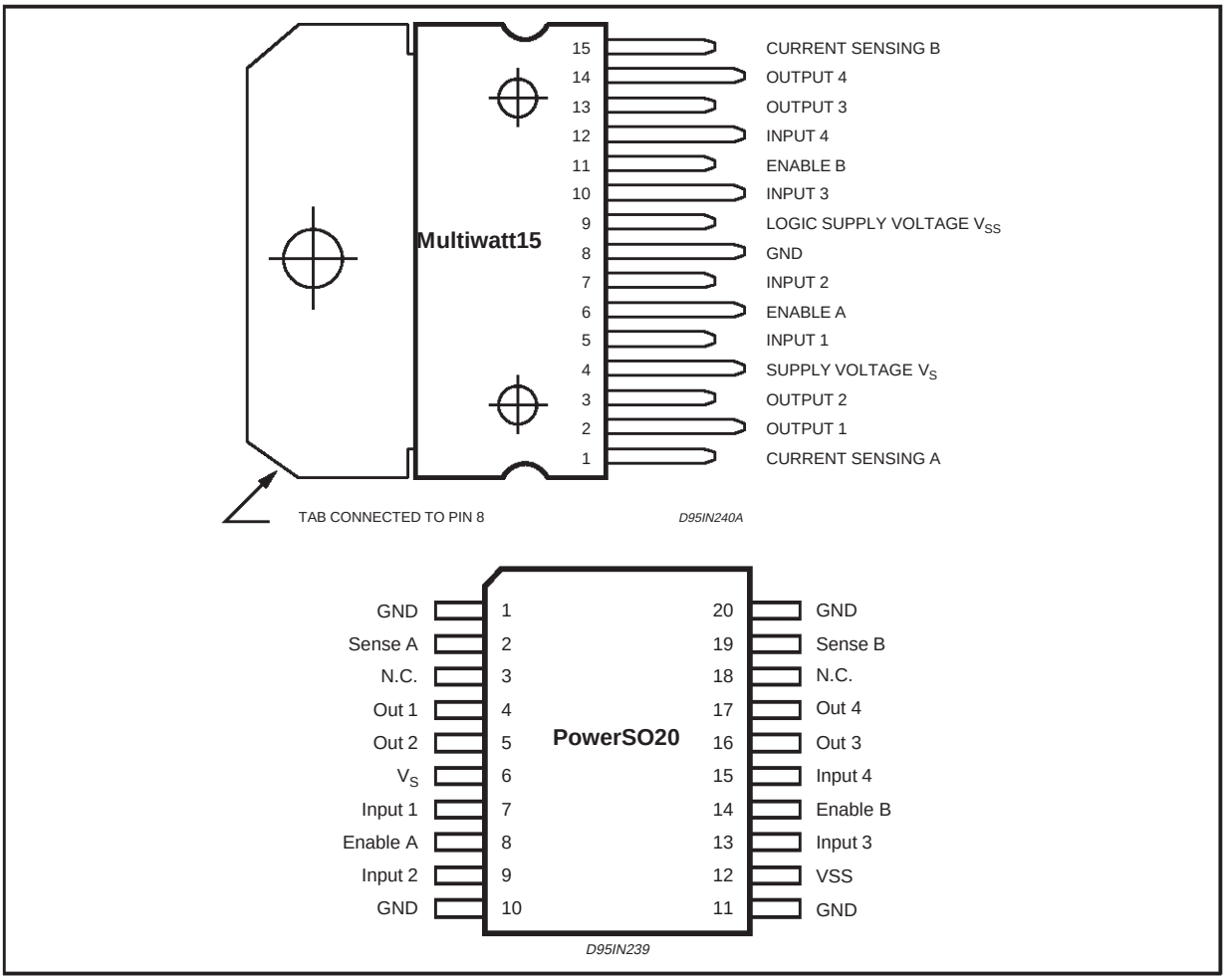
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_I, V_{en}	Input and Enable Voltage	–0.3 to 7	V
I_O	Peak Output Current (each Channel) – Non Repetitive (t = 100µs) –Repetitive (80% on –20% off; t_{on} = 10ms) –DC Operation	 3 2.5 2	 A A A
V_{sens}	Sensing Voltage	–1 to 2.3	V
P_{tot}	Total Power Dissipation (T_{case} = 75°C)	25	W
T_{op}	Junction Operating Temperature	–25 to 130	°C
T_{stg}, T_j	Storage and Junction Temperature	–40 to 150	°C

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	Max.	–	3	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	°C/W

(*) Mounted on aluminum substrate

PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V_S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	VSS	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition	V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0 V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0 V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L V_i = X			6	mA
V_{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V_{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		VSS	V
I_{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L			–10	µA
I_{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	Vi = H ≤ V_{SS} –0.6V		30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)		2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L			–10	µA
I_{en} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{SS} –0.6V		30	100	µA
$V_{CEsat(H)}$	Source Saturation Voltage	I_L = 1A I_L = 2A	0.95	1.35 2	1.7 2.7	V V
$V_{CEsat(L)}$	Sink Saturation Voltage	I_L = 1A (5) I_L = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V_{CEsat}	Total Drop	I_L = 1A (5) I_L = 2A (5)	1.80		3.2 4.9	V V
V_{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T_1 (V_i)	Source Current Turn-off Delay	0.5 V_i to 0.9 I_L (2); (4)		1.5		µs
T_2 (V_i)	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		0.2		µs
T_3 (V_i)	Source Current Turn-on Delay	0.5 V_i to 0.1 I_L (2); (4)		2		µs
T_4 (V_i)	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.7		µs
T_5 (V_i)	Sink Current Turn-off Delay	0.5 V_i to 0.9 I_L (3); (4)		0.7		µs
T_6 (V_i)	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.25		µs
T_7 (V_i)	Sink Current Turn-on Delay	0.5 V_i to 0.9 I_L (3); (4)		1.6		µs
T_8 (V_i)	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.2		µs
fc (V_i)	Commutation Frequency	I_L = 2A		25	40	KHz
T_1 (V_{en})	Source Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (2); (4)		3		µs
T_2 (V_{en})	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		1		µs
T_3 (V_{en})	Source Current Turn-on Delay	0.5 V_{en} to 0.1 I_L (2); (4)		0.3		µs
T_4 (V_{en})	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.4		µs
T_5 (V_{en})	Sink Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (3); (4)		2.2		µs
T_6 (V_{en})	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.35		µs
T_7 (V_{en})	Sink Current Turn-on Delay	0.5 V_{en} to 0.9 I_L (3); (4)		0.25		µs
T_8 (V_{en})	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.1		µs

1) 1)Sensing voltage can be –1 V for t ≤ 50 µsec; in steady state V_{sens} min ≥ –0.5 V.
2) See fig. 2.
3) See fig. 4.
4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current.

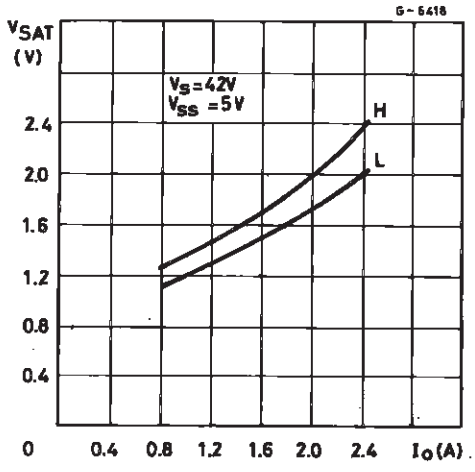
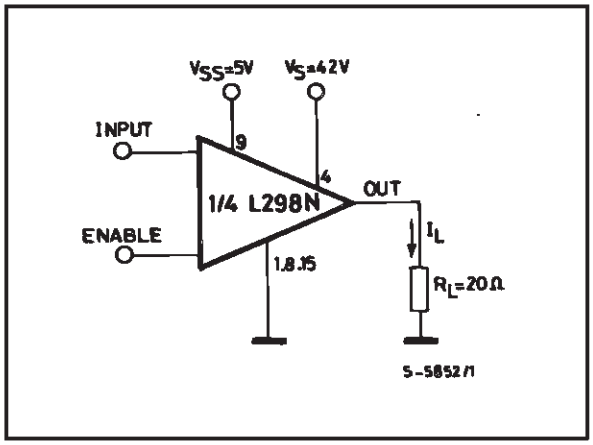


Figure 2 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = H

Figure 3 : Source Current Delay Times vs. Input or Enable Switching.

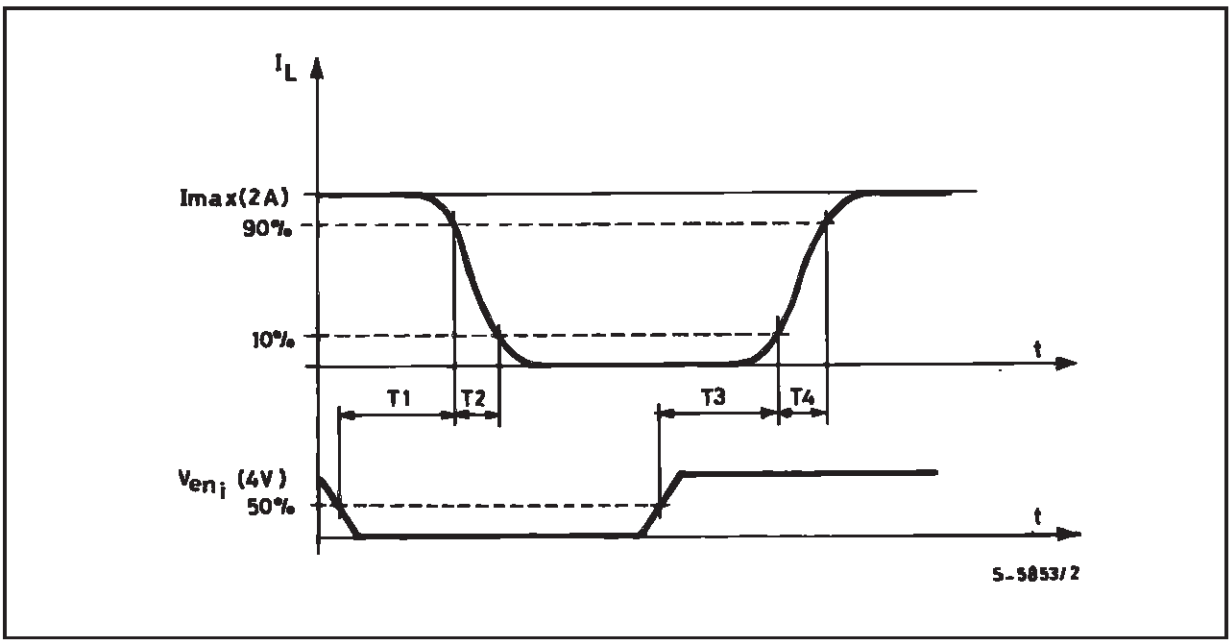
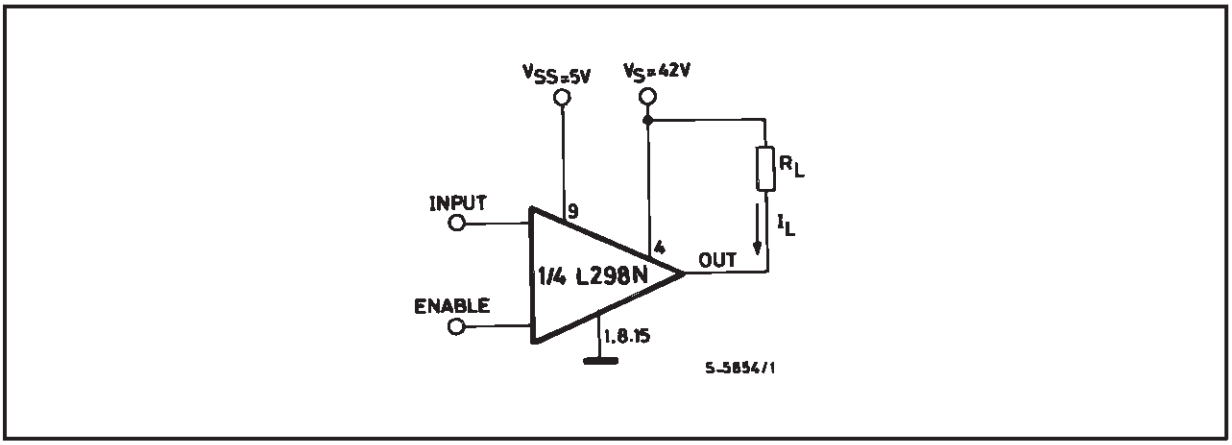


Figure 4 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = L

Figure 5 : Sink Current Delay Times vs. Input 0 V Enable Switching.

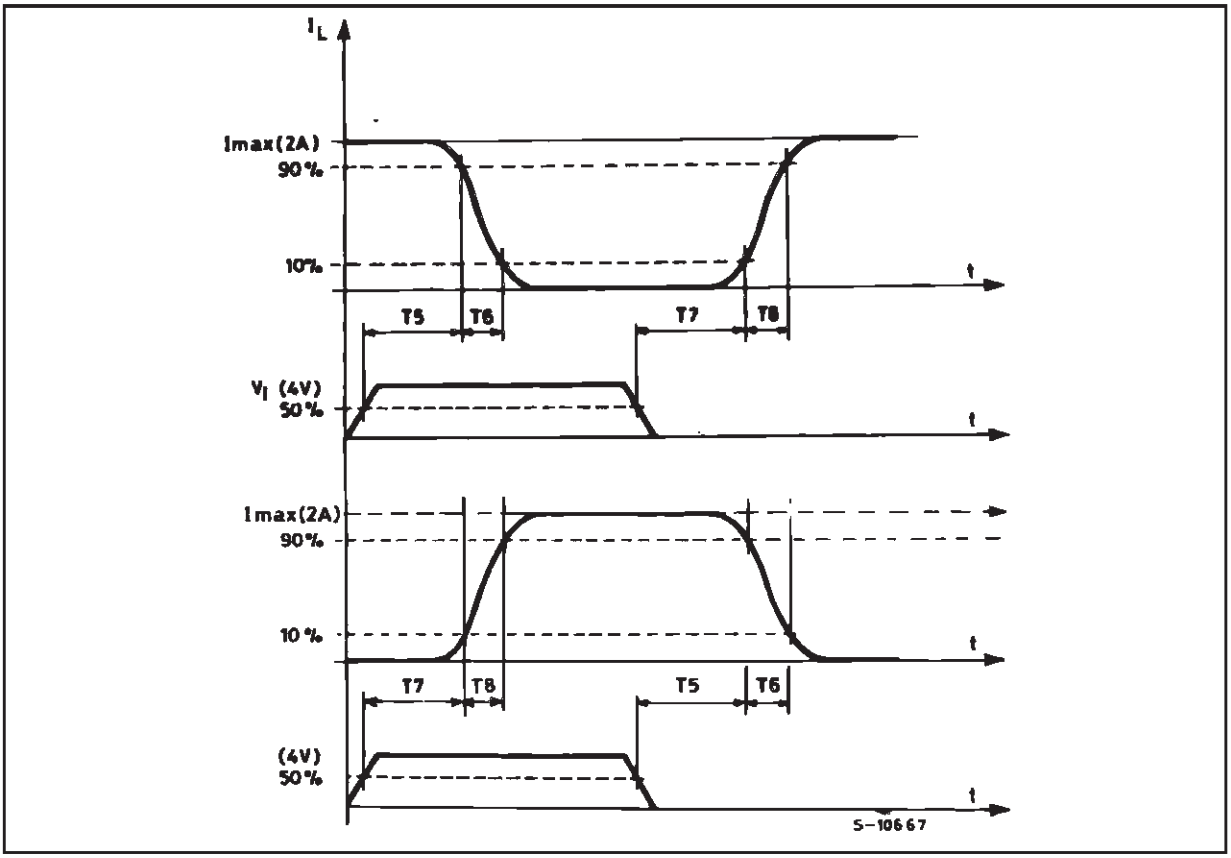
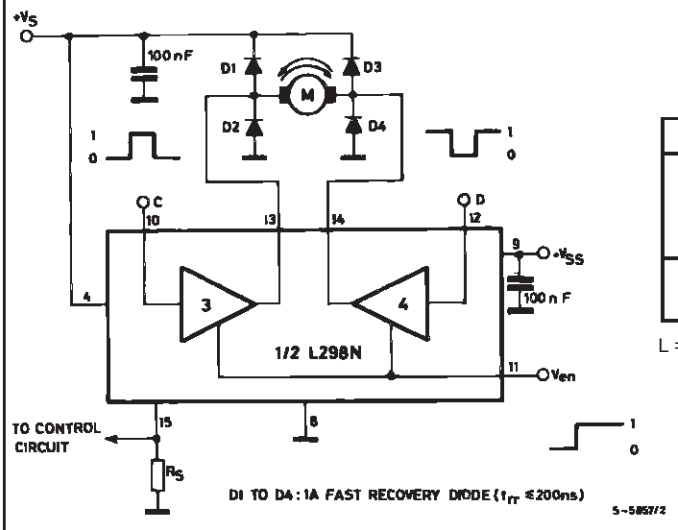


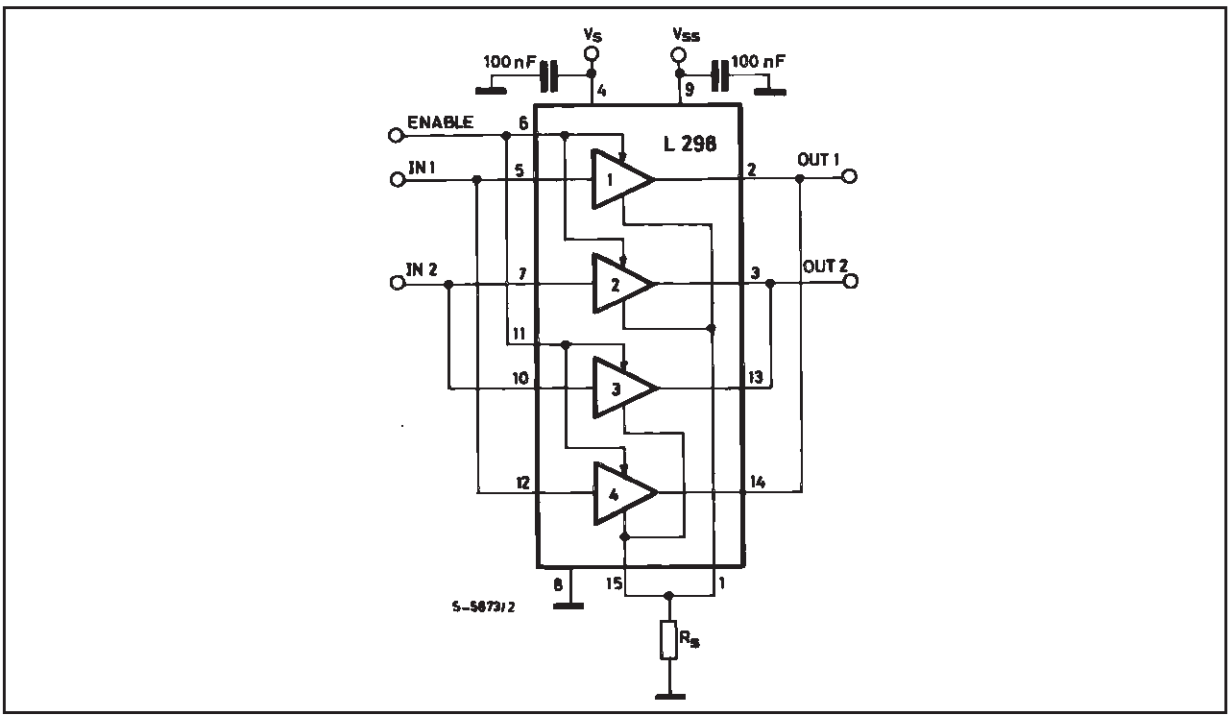
Figure 6 : Bidirectional DC Motor Control.



Inputs		Function
V_{en} = H	C = H ; D = L	Forward
	C = L ; D = H	Reverse
	C = D	Fast Motor Stop
V_{en} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't care

Figure 7 : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



APPLICATION INFORMATION (Refer to the block diagram)

1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A ; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differenzial mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output : an external resistor (R_{SA} ; R_{SB}.) allows to detect the intensity of this current.

1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are In1 ; In2 ; EnA and In3 ; In4 ; EnB. The In inputs set the bridge state when The En input is high ; a low state of the En input inhibits the bridge. All the inputs are TTL compatible.

2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both Vs and Vss, to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of Vs that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes D1 to D4 is made by four fast recovery elements (trr $\leq$ 200 nsec) that must be chosen of a VF as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped ; Shottky diodes would be preferred.

This solution can drive until 3 Amps In DC operation and until 3.5 Amps of a repetitive peak current.

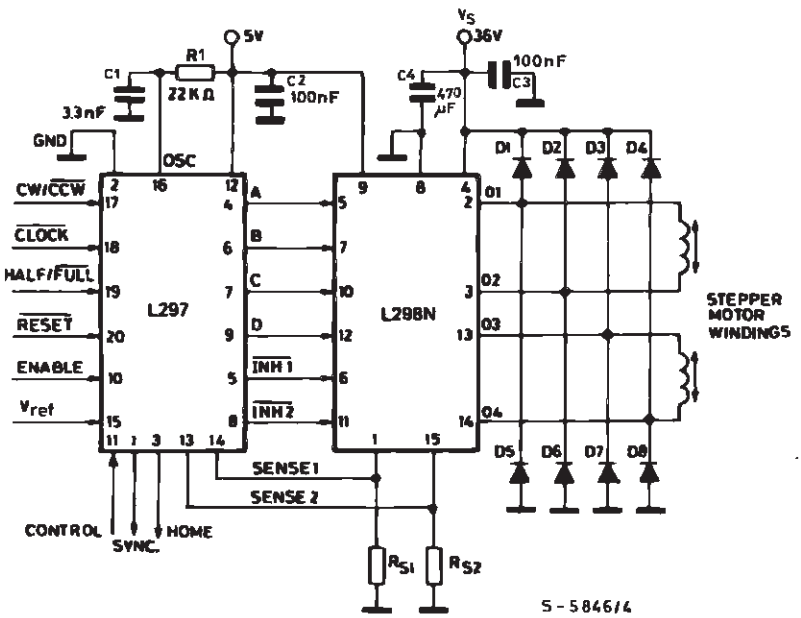
On Fig 8 it is shown the driving of a two phase bipolar stepper motor ; the needed signals to drive the inputs of the L298 are generated, in this example, from the IC L297.

Fig 9 shows an example of P.C.B. designed for the application of Fig 8.

Fig 10 shows a second two phase bipolar stepper motor control circuit where the current is controlled by the I.C. L6506.

Figure 8 : Two Phase Bipolar Stepper Motor Circuit.

This circuit drives bipolar stepper motors with winding currents up to 2 A. The diodes are fast 2 A types.



$R_{S1} = R_{S2} = 0.5\ \Omega$

D1 to D8 = 2 A Fast diodes { $V_F \leq 1.2$ V @ I = 2 A
trr ≤ 200 ns

Figure 9 : Suggested Printed Circuit Board Layout for the Circuit of fig. 8 (1:1 scale).

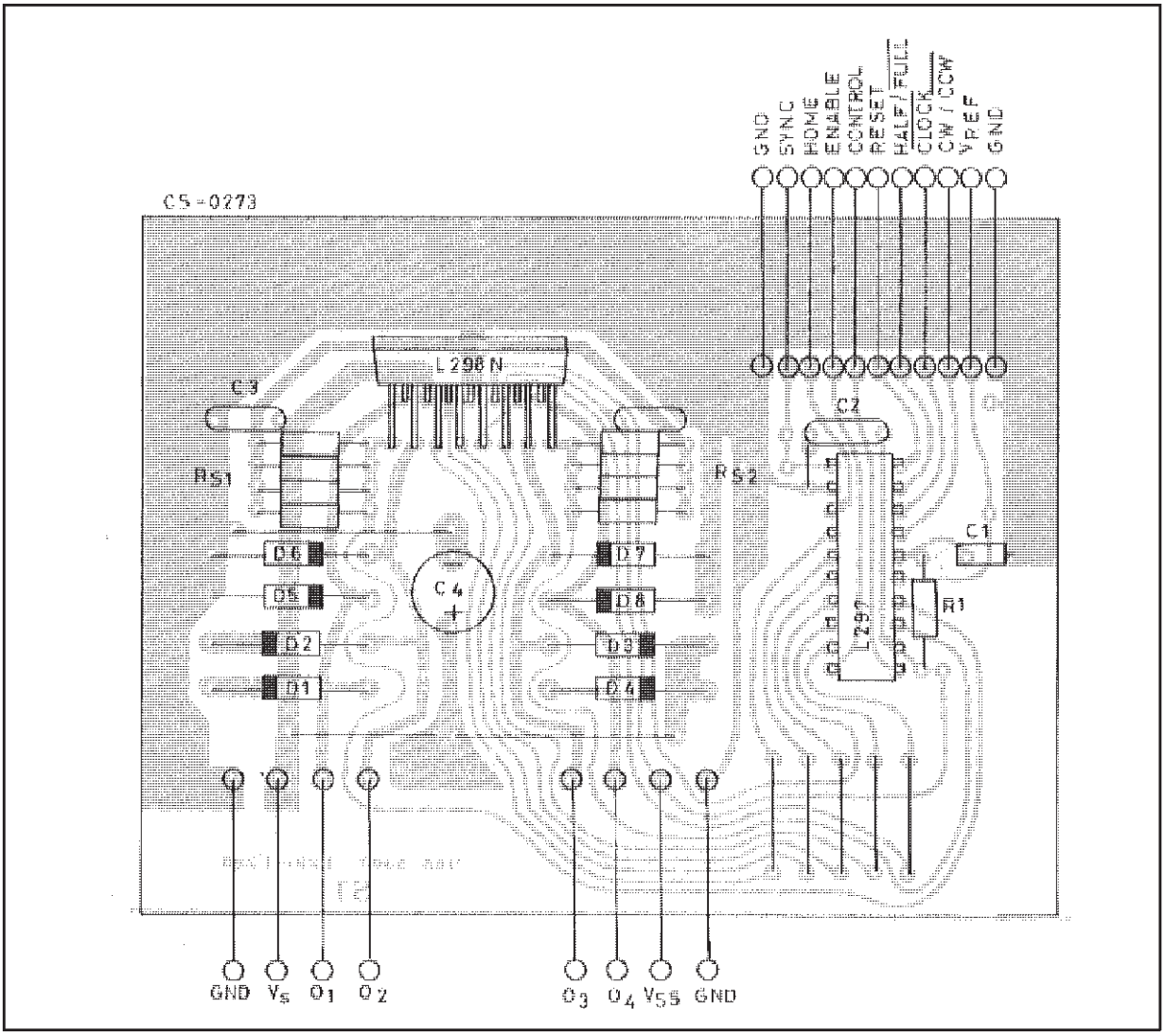
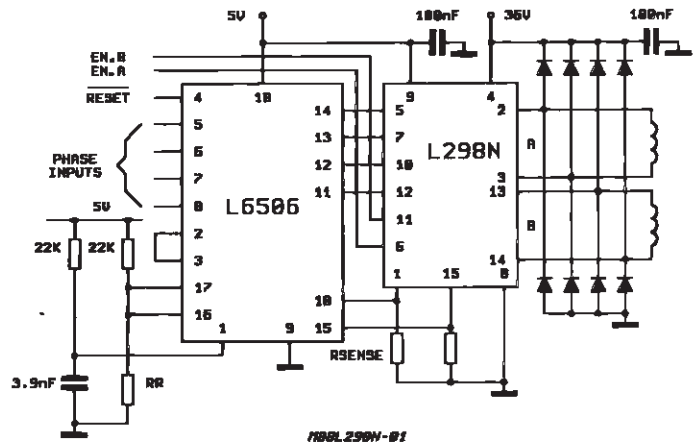


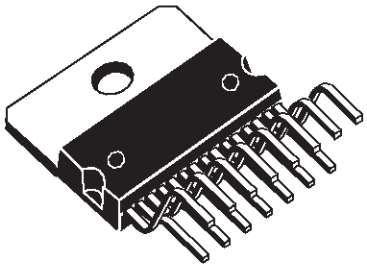
Figure 10 : Two Phase Bipolar Stepper Motor Control Circuit by Using the Current Controller L6506.



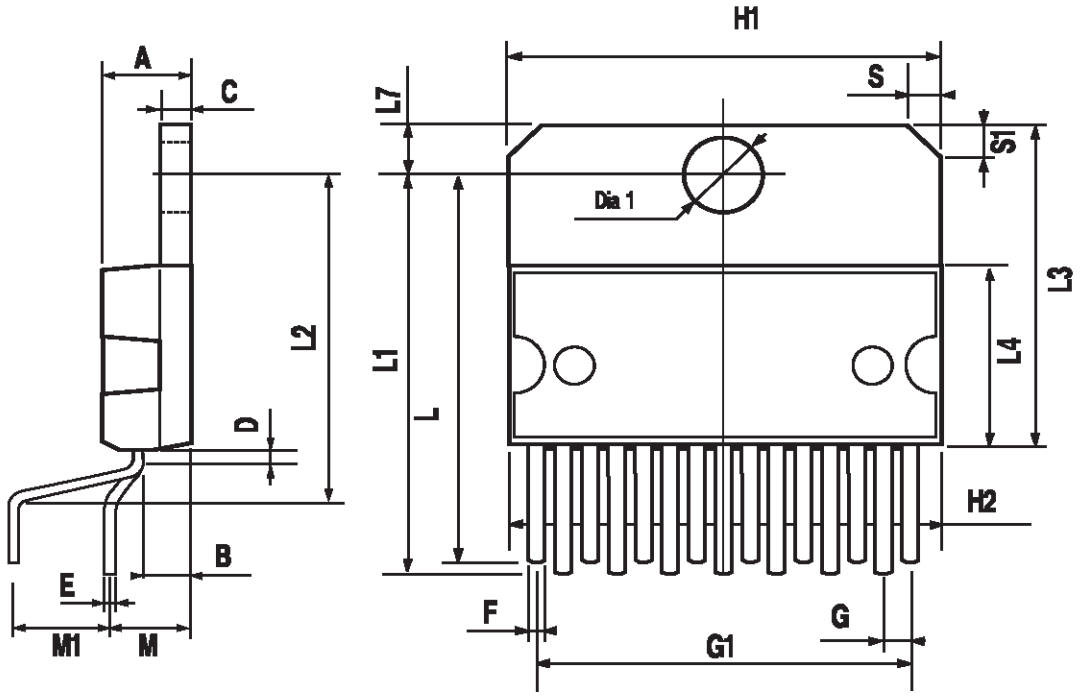
R_R and R_{sense} depend from the load current

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
D		1			0.039	
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.02	1.27	1.52	0.040	0.050	0.060
G1	17.53	17.78	18.03	0.690	0.700	0.710
H1	19.6			0.772		
H2			20.2			0.795
L	21.9	22.2	22.5	0.862	0.874	0.886
L1	21.7	22.1	22.5	0.854	0.870	0.886
L2	17.65		18.1	0.695		0.713
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L7	2.65		2.9	0.104		0.114
M	4.25	4.55	4.85	0.167	0.179	0.191
M1	4.63	5.08	5.53	0.182	0.200	0.218
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA

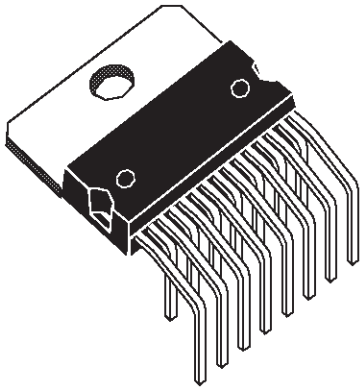


Multiwatt15 V

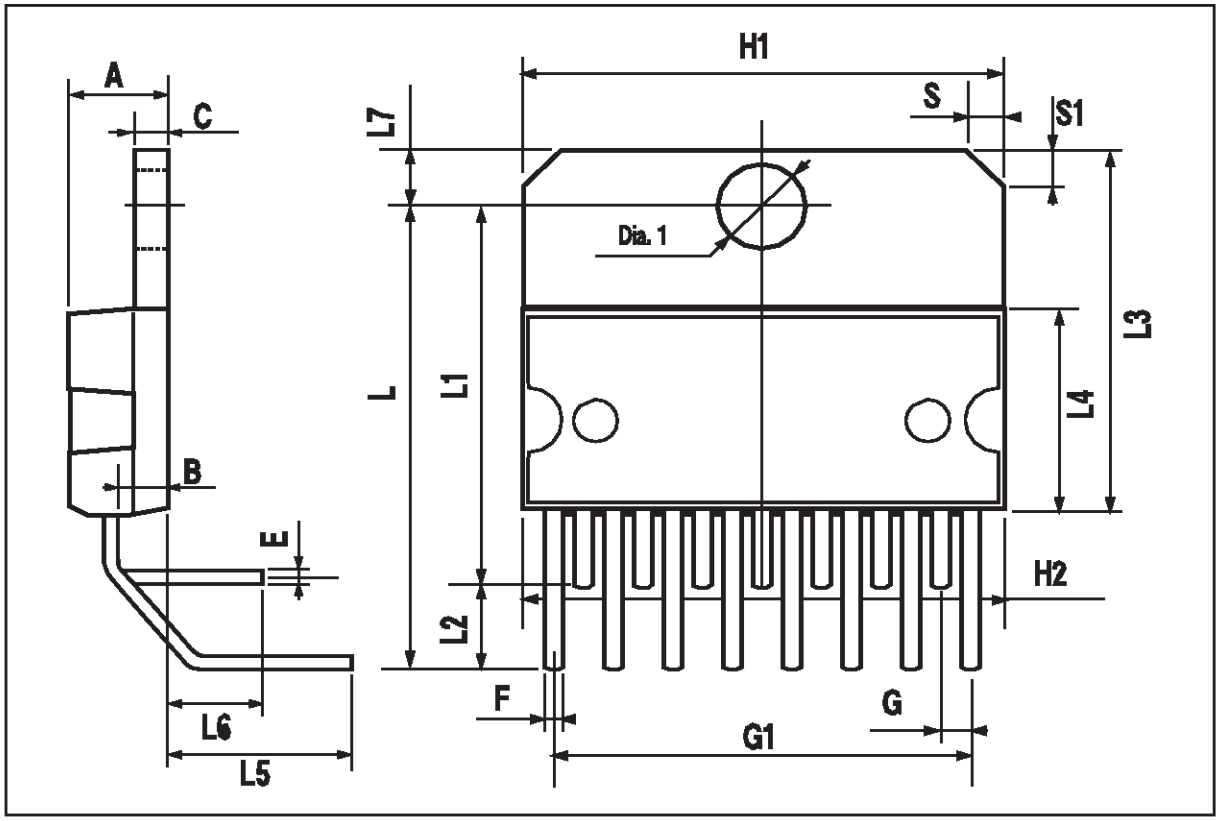


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.14	1.27	1.4	0.045	0.050	0.055
G1	17.57	17.78	17.91	0.692	0.700	0.705
H1	19.6			0.772		
H2			20.2			0.795
L		20.57			0.810	
L1		18.03			0.710	
L2		2.54			0.100	
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L5		5.28			0.208	
L6		2.38			0.094	
L7	2.65		2.9	0.104		0.114
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA



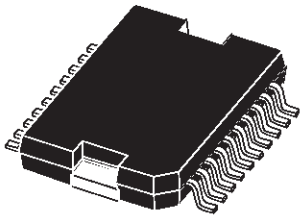
Multiwatt15 H



DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			3.6			0.142
a1	0.1		0.3	0.004		0.012
a2			3.3			0.130
a3	0		0.1	0.000		0.004
b	0.4		0.53	0.016		0.021
c	0.23		0.32	0.009		0.013
D (1)	15.8		16	0.622		0.630
D1	9.4		9.8	0.370		0.386
E	13.9		14.5	0.547		0.570
e		1.27			0.050	
e3		11.43			0.450	
E1 (1)	10.9		11.1	0.429		0.437
E2			2.9			0.114
E3	5.8		6.2	0.228		0.244
G	0		0.1	0.000		0.004
H	15.5		15.9	0.610		0.626
h			1.1			0.043
L	0.8		1.1	0.031		0.043
N	10° (max.)					
S	8° (max.)					
T		10			0.394	

(1) "D and F" do not include mold flash or protrusions.
- Mold flash or protrusions shall not exceed 0.15 mm (0.006").
- Critical dimensions: "E", "G" and "a3"

OUTLINE AND MECHANICAL DATA



JEDEC MO-166

PowerSO20