**UNIVERSIDADE ESTADUAL DE CAMPINAS**

**FACULDADE DE ENGENHARIA MECÂNICA**

# Desenvolvimento de um Robô Autônomo Móvel Versátil utilizando Arquitetura Subsumption

Autor : **Justo Emilio Alvarez Jácobo**

Orientador: **Prof. Dr. Pablo Siqueira Meirelles**

71/01

**UNIVERSIDADE ESTADUAL DE CAMPINAS**
**FACULDADE DE ENGENHARIA MECÂNICA**
**DEPARTAMENTO DE MECÂNICA COMPUTACIONAL**

# Desenvolvimento de um Robô Autônomo Móvel Versátil utilizando Arquitetura Subsumption

Autor : **Justo Emilio Alvarez Jácobo**
Orientador: **Prof. Dr. Pablo Siqueira Meirelles**

Curso: Engenharia Mecânica.
Área de concentração: Mecânica dos Sólidos e Projeto Mecânico

Dissertação de mestrado apresentada à comissão de Pós Graduação da Faculdade de Engenharia Mecânica, como requisito para obtenção do título de Mestre em Engenharia Mecânica.

Campinas, 2001
S.P. - Brasil

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

AL86d

Alvarez Jácobo, Justo Emilio.
Desenvolvimento de um robô autônomo móvel versátil utilizando arquitetura subsumption / Justo Emilio Alvarez Jácobo.--Campinas, SP: [s.n.], 2001.

Orientador: Pablo Siqueira Meirelles.
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Robótica. 2. Robôs - Sistemas de controle. 3. Robôs móveis. 4. Controladores programáveis. I. Meirelles, Pablo Siqueira. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título.

**UNIVERSIDADE ESTADUAL DE CAMPINAS**

**FACULDADE DE ENGENHARIA MECÂNICA**

**DEPARTAMENTO DE MECÂNICA COMPUTACIONAL**

**DISSERTAÇÃO DE MESTRADO**

# Desenvolvimento de um Robô Autônomo Móvel Versátil, utilizando Arquitetura Subsumption


Autor : **Justo Emilio Alvarez Jácobo**

Orientador: **Prof. Dr. Pablo Siqueira Meirelles**


**Prof. Dr. Pablo Siqueira Meirelles, Presidente**
**FEM/UNICAMP**

**Prof. Dr. Franco Giuseppe Dedini**
**FEM/UNICAMP**

**Prof. Dr. Akebo Yamakami**
**FEEC/UNICAMP**

Campinas, 24 de Agosto de 2001

## Dedicatória:

Dedico este trabalho a minha querida mãe, como gratidão por todo o esforço, dedicação, e amor que teve para conosco e a meu querido sobrinho como símbolo da esperança de um futuro melhor.

## Agradecimentos

Em primeiro lugar, agradeço a Deus, pela benção da inteligência, saúde, amizade, e principalmente por jamais me ter deixado sozinho na procura dos meus sonhos.

Um agradecimento muito especial aos meus pais, a minha irmã Isabelita, aos meus tios Elma e Anizio, e a Paola pelo apoio, incentivo e compreensão na realização deste trabalho.

Meu profundo agradecimento ao Prof. Pablo Siqueria por esta oportunidade, e acima de tudo pela paciência que teve para comigo ao longo destes anos. Obrigado por todo o tempo que me dispensou.

Aos meus amigos conterrâneos, Blanca, Marco, John, Carlitos Salinas, pela grande amizade que me brindaram e pelos valiosos momentos que tivemos juntos, os quais me ajudaram a superar as saudades da pátria.

Aos meus amigos de Campinas, Kelly, Sílvia, Egnilson, André, Fábio e Antonio, por me suportar durante todo este tempo.

Ao Dr. Roberto Tavares pela ajuda com o microcontrolador PIC.

A todos os meus colegas do DMC que de alguma forma contribuíram para o meu trabalho.

E finalmente para o Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pela ajuda financeira para a realização deste trabalho.

*"Ama Sua, Ama Llulla, Ama Quella."*

*"No seas ladrón, no seas mentiroso, no seas perezoso."*

*"Não seja ladrão, não seja mentiroso, não seja preguiçoso."*

*Leis principais do Império Inca*

## Resumo

ALVAREZ JÁCOBO, Justo Emilio, *Desenvolvimento de um Robô Autônomo Móvel Versátil utilizando Arquitetura Subsumption*, Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2001. 139 p.. Dissertação (Mestrado).

Este trabalho apresenta o projeto, implementação e construção do robô autônomo móvel **RAM-I**, o qual tem autonomia energética e de controle. A locomoção é feita através de motores de passo, controlados pelo módulo Controlador dos Motores, baseados no microcontrolador PIC16F84. O módulo Principal, recebe os sinais dos sensores externos e os processa conforme a estratégia de controle implementada. Esta é baseada na Arquitetura *Subsumption*, tomando as decisões que serão enviadas pela interface para o módulo Controlador dos Motores, sem necessidade de intervenção externa. A alimentação é feita por meio de baterias embarcadas. Será apresentado o projeto do Sistema, descrição das partes, implementação da estratégia de controle, modelagem do robô e por ultimo os resultados experimentais obtidos com o protótipo construído em laboratório.

*Palavras Chave*

Robótica, Controle, Arquitetura *Subsumption,* Microcontroladores.

## Abstract

ALVAREZ JÁCOBO, Justo Emilio, *Development of a Versatile Mobile Autonomous Robot using Subsumption Architecture*, Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2001. 139 p.. Dissertação (Mestrado).

This work presents the design, implementation and construction of a mobile autonomous robot **RAM-I**, which has autonomous energy supply and control. The locomotion is done through stepper motors, controlled by the module Controller of the Motors based on microcontroller PIC16F84. The Main module, also based in the PIC, receives the inputs signals from external sensors and processes according with the control strategy implemented, based on the Architecture of Subsumption, taking decisions that will be sent to the interface for the Motor Controller, without the necessity of external intervention. The power will be supplied by internal batteries. It will be presented the design of the system, description of the parts, implementation of the control strategy and modeling, as well as the experimental results obtained with the prototype constructed in the laboratory.

*Key Words*

Robotic, Control, Architecture Subsumption, Microcontroller.

# Índice

# Lista de Figuras

# Lista de Tabelas

# Nomenclatura

## *Letras Latinas*

**x**- Abcissa no plano cartesiano.

**y**- Ordenada no plano cartesiano.

**I**- Sistema de coordenadas cartesianas de referencia.

**X**- Sistema de coordenadas cartesianas móvel solidária como robô.

**P**- Ponto no robô onde o Sistema móvel de referência está assentado.

**R**- Matriz de rotação de coordenadas.

**l**- Distância entre o centro da roda e a origem do sistema móvel no robô.

**r**- raio da roda.

**d**- Excentricidade nas rodas orientadas fora do centro.

**N**- Número de rodas.

**L**- Méia distância entre as rodas medida sobre o eixo destas.

## *Letras Gregas*

$\theta$ - Posição angular do robô em relação ao sistema de referencia inercial.

$\beta$ - Ângulo de rotação das rodas orientáveis.

$\varphi$ - Ângulo de rotação das rodas em torno do seu centro.

$\xi$ - Postura cartesiana do robô.

$\eta_1$ - Velocidade longitudinal do robô.

$\eta_2$ - Velocidade angular do robô.

$\lambda$ - Matriz que relaciona as variáveis de controle internas com as velocidades das rodas.

$\alpha$ - Posição angular do centro da roda em relação ao chassi do robô.

$\delta_m$ - Índice de Mobilidade do robô.

$\delta_s$ - Índice de Dirigibilidade do robô.

$\gamma$ - Ângulo que determina a direção de não deslizamento nas rodas suecas.

$\sum$ - Matriz cujas colmas formam uma base no espaço nulo da matriz das restrições cinemáticas.

$\phi$ - Trajetória.

## *Subscritos*

**c**- Rodas orientáveis no centro.

**fc**- Rodas orientáveis fora do centro.

**sw**- Rodas suecas.

# Capítulo 1

# Introdução e Revisão Bibliográfica

## 1.1 Introdução

Nos últimos anos observou-se um aumento no grau de consciência com relação ao potencial de aplicações dos robôs autônomos móveis, em especial em tarefas perigosas, sujas ou desagradáveis. As aplicações propostas para robótica avançada parecem cobrir a quase totalidade da atividade humana. Estas incluem: luta contra o fogo, salvamento de emergência, prevenção de desastres, patrulha de segurança, limpeza industrial e doméstica, serviço doméstico, manipulação de pacientes, operações de procura em ambientes distantes e/ou inóspitos, etc.

Uma caraterística comum destas aplicações, ao invés do robô industrial, é a exigência de um veículo móvel ou plataforma, que sirva como base para qualquer tarefa exigida. A exigência comum para tal veículo móvel ou plataforma, é a habilidade para navegar de uma posição conhecida a uma nova localização, evitar obstáculos e se posicionar na tarefa a ser realizada. Isto é possível com o uso de um sistema de sensores, o qual deve adquirir os dados que descrevem o ambiente e passá-los ao sistema de computador do robô, que fará os cálculos necessários para que o sistema piloto do robô controle os movimentos.

O desenvolvimento de robôs autônomos constitui uma área atual de pesquisa em robótica. Tais robôs aceitarão ordens para a execução de tarefas com um elevado grau de dificuldade e as cumprirão sem a intervenção humana adicional (Latombe, 1996). Na entrada das ordens será

especificado o que o usuário quer que seja realizado e não uma explicação do que deve ser feito para realizá-las (Andrade, 2001).

Ao colocar o robô como substituto do homem, deve-se dotá-lo de autonomia, para que ele possas trabalhar conjuntamente com as demais máquinas. Esta autonomia deve ser: mecânica, de controle, e se é possível, energética. Esta premissa tem sido uma das principais motivações para a pesquisa do robô autônomo móvel.

Nos últimos anos, também parece haver uma desafortunada divergência na pesquisa da robótica autônoma móvel, entre teoria e prática. Relatórios publicados nesta área podem ser divididos em duas categorias: trabalho teórico com pequena ou nenhuma verificação experimental (a excepção de simulação), e resultados experimentais de sistemas implementados com pequena ou nenhuma fundamentação teórica formal. É raro encontrar um postulado teórico formal verificado (ou refutado) por experiências reproduzidas em um robô real (Gat, 1995).

Esta divergência é devida ao fato de que os robôs autônomos móveis têm que interagir com ambientes complexos, que especificamente não foram criados para o robô. Estas interações são extremamente difíceis de modelar, pois são governadas por um enorme número de variáveis independentes. As formulações teóricas ignoram habitualmente fatores como: custo computacional, ruído nos sensores, interações mecânicas, fricção e deformação da superfície.

Temos ainda, que os robôs móveis com rodas e os robôs móveis a pernas, juntamente com os satélites, pertencem a uma classe de sistema mecânico denominada de sistemas não-holonômicos, que se caraterizam por ter restrições cinemáticas não integráveis. Os algoritmos de controle e planejamento de movimento de tais sistemas requerem, portanto, uma classe diferente de procedimentos do que aqueles empregados no controle de manipuladores mecânicos estacionários, exemplo claro de um sistema holonômico (Victorino, 1998).

## 1.2 Breve Histórico.

Desde os tempos mais remotos, o homem tem sentido a necessidade de criar mecanismos que o auxiliem no seu trabalho. Esta capacidade de criação do homem se remonta aos tempos dos antigos egípcios, aproximadamente no ano 3 000 A.C., os quais haviam construído relógios de água e figuras articuladas para seus cultos.

O desenvolvimento da arte mecânica alcançou notável intensidade entre os anos 1400 e 1700. Nas obras dos artistas e experimentalistas destes anos ganha corpo uma nova apreciação sobre o trabalho: a função do saber técnico.

O mundo deve aos trabalhos de Leonardo da Vinci, Galileu Galilei, Giordano Bruno, Francisco Baco, Pascal, Descarte e Diderot, entre outros, os quais marcam a divisão entre o antigo e o moderno, a valorização do saber técnico. Ao não mais desprezar o saber técnico, eles criaram uma nova concepção de técnica e ciência, fator importante na concepção de novas idéias e de progressos científicos.

Esses homens não foram apenas acadêmicos, trabalhavam nas oficinas e ateliês, não desprezando a prática. Por exemplo: Descartes nos últimos anos da sua vida, elaborou o projeto de uma grande escola de artes e ofícios, que teria a tarefa de estabelecer elo entre os trabalhos dos cientistas, dos artesãos e dos técnicos. A escola idealizada por Descartes, era composta por várias grandes salas para os artesãos; para cada grupo de ofício uma sala. Cada sala tinha um laboratório provido de todos os instrumentos mecânicos necessários ou úteis às artes que ali deveriam ser ensinadas; o número de professores ou mestres era igual ao das artes ensinadas. Esses professores deveriam ser especialistas em física e matemática para responder a todas as perguntas dos artesãos, e assim explicar e esclarecer dúvidas, para realizar novas descobertas sobre as artes, (Viera, 1996).

Todos esses novos métodos da ciência experimental que provocaram a revolução técnica/científica do século XVII, abriram caminho à grande transformação dos meios de produção, a chamada Revolução Industrial.

A primeira fase desta Revolução Industrial acontece quando as escolas de artes e ofícios tornam-se independentes do estado, e a fase definitiva dessa revolução acontece com o surgimento da fábrica, quando James Watt desenvolve o governador para a máquina de vapor. O surgimento da fábrica traz consigo a divisão dos processos de manufatura, e em particular o trabalho cooperativo.

A Revolução Industrial, iniciada na Inglaterra no final do século XVIII , trouxe como fator sociológico o surgimento das fábricas, a criação e desenvolvimento das grandes cidades industriais e consequentemente, o êxodo rural.

Se vemos o avanço na área de robótica, temos por exemplo, autômatos construídos em 1770 que escreviam, desenhavam e tocavam instrumentos musicais. Até então, a utilização de tais criações era para entretenimento. (Rezende, 1992).

A Segunda Revolução Industrial (Era da modernização) idealizada como saída econômica para a grande depressão americana, iniciou-se nos anos trinta do século passado com a automação que Henry Ford realizou em sua fábrica de automóveis, baseada na organização estruturada do processo. Nesta época surge o conceito de autonomia.

A terceira Revolução Industrial (Era de informação), ainda está em processo. Idealizada como saída econômica para a grande crise do petróleo, nela se vê a robotização do processo produtivo.

O primeiro a utilizar a palavra "*robô*" foi o tetrólogo tcheco Karel Capek na sua publicação "Os Robôs Universais de Roussum", cujo significado é servidão ou trabalho forçado.

A partir de 1930, surgem na ficção os chamados "robôs amigáveis" idealizados pelo escritor Isaac Asimov. Tais robôs eram máquinas bem projetadas, cuja construção e programação eram baseadas em princípios denominados As Três Leis da Robótica: (Rezende, 1992 )

1. Um robô não pode ferir um ser humano ou, por inação, permitir que um humano seja ferido.
2. Um robô deve obedecer às ordens dada por humanos, exceto quando isto conflitar com a Primeira Lei.
3. Um robô deve proteger sua própria existência a menos que isto conflite com a Primeira ou Segunda Lei.

A partir da Revolução Industrial, vários esforços foram empreendidos no sentido de criar mecanismos que facilitem a vida do homem, ou seja, mecanismos que façam pelo homem os trabalhos rotineiros, pesados e cansativos.

O desenvolvimento destes mecanismos a princípio foi difícil. Com os computadores digitais, conseguiu-se avanços significativos. As incontáveis aplicações fizeram com que mais recursos fossem voltados para esta área, fazendo desta forma com que os pesquisadores começassem a se interessar por ela.

Ultimamente, além das pesquisas na área tecnológica, que estão voltadas mais ao desenvolvimento de robôs de aplicação prática, como os industriais, mais ligados à relação custo/benefício e produtividade; outra área que vêm despertando interesse dos pesquisadores é a área científica. Nesta área é enfatizada a percepção sensorial, controle motor, e comportamento inteligente.

Tal como ocorreu com os computadores, os robôs foram classificados de acordo com o seu surgimento:

- **Primeira Geração: Robôs Seqüenciais:** São manipuladores automáticos a ciclos ou cadência de operadores preestabelecidas, controlados em malha aberta. Têm de 2 a 4 graus de liberdade e só executam tarefas simples como por exemplo, os de carga e descarga de prensas e máquinas ferramenta.

- **Segunda Geração: Robôs a Ciclos Programáveis:** São robôs um pouco mais sofisticados, possuem de 4 a 8 graus de liberdade e em função do modo de programação podem ser divididos em: robôs programáveis por aprendizagem ou robôs "Play-Back" e robôs programáveis por linguagem.

- **Terceira Geração: Robôs Inteligentes:** São robôs capazes de se adaptar às modificações do ambiente mediante sistemas de controle, percepção, comunicação e decisão e são capazes de executar tarefas via "interações" com o meio ambiente.

É importante notar que a maioria dos autores referem-se à Terceira Geração, como aqueles capazes de operar autonomamente num ambiente desconhecido. É assim que temos como comum referência: Robôs Autônomos, Robôs Autônomos Inteligentes e Robôs Móveis Autônomos, todos eles da Terceira Geração, hoje dedicados a fazer tarefas especializadas em ambientes desestruturados (Rezende, 1992).

Os robôs móveis possuem três formas básicas de locomoção, podendo utilizar apenas uma, como também, uma associação dessas configurações. Estas formas são: dispositivo articulado de rotação (rodas), corpos articulados e pernas. A forma de locomoção do robô deve levar em conta a finalidade, o tipo de terreno em que o robô opera, fonte de alimentação e autonomia energética.

Vendo as diferenças entre os robôs antes citados, verifica-se que os robôs com pernas apresentam duas vantagens: a mobilidade em terrenos irregulares e a suspensão ativa que proporciona um transporte mais estável.

## 1.3 Pontos Importantes em Robótica

Para melhor entender este trabalho, torna-se relevante levar em consideração algumas definições, que, ainda que muito conhecidas, serão muito usadas nesta pesquisa, como é o caso da definição do robô:

### 1.3.1 Definição do Robô

- **Definição da JIRA (Japan Industrial Robot Association):** O robô é definido como um sistema mecânico que possui movimentos flexíveis análogos aos movimentos orgânicos, e combina esses movimentos com funções inteligentes e ações semelhantes as do humano. Neste contexto, função inteligente significa o seguinte: decisão, reconhecimento, adaptação ou aprendizagem.

- **Definição do RIA (Robot Institute of America):** O robô é definido como um dispositivo mecânico programável para execução de algumas tarefas de manipulação ou locomoção sob controle automático. Robô industrial é um manipulador multifuncional e reprogramável projetado para movimentar materiais, peças e ferramentas ou dispositivos especiais, conforme programação prévia, de modo a executar uma variedade de tarefas.

### 1.3.2 Robótica Móvel Autônoma

Também é necessária a introdução de uma definição para robô móvel. Muir (1988), define um robô móvel como: "um robô capaz de se locomover sobre uma superfície somente através da atuação de rodas montadas no robô e em contato com a superfície" (Victorino, 1998). As rodas permitem um deslocamento relativo entre o seu eixo e a superfície sobre a qual se espera ter um único ponto de contato com rolamento puro.

Um robô móvel atual é equipado com sensores (visão, infravermelho, sonar, tato, toque, sistemas de navegação inercial, etc.) que permitem a percepção do meio ambiente, total ou parcialmente desconhecido, e é dotado de capacidade de decisão, que lhe permite cumprir uma tarefa sem intervenção humana. Quando tais sensores estão disponíveis no robô, pode-se fazer uso da interação do robô com o ambiente, fazendo-o perceber e construir o modelo do ambiente no qual o movimento se desenvolve e depois decidir as ações a serem tomadas para a realização da tarefa.

Os processos de percepção e decisão podem ser considerados de três formas:

- estratégia reativa: o comportamento do robô é determinado pelos estímulos exteriores vindos do ambiente através dos sensores.
- estratégia deliberada: as informações do ambiente são previamente processadas e a tarefa e o comportamento do robô são, da mesma forma, previamente determinadas.
- estratégia híbrida: junção das duas anteriores.

## 1.4 Robôs Móveis

Um dos primeiros robôs móveis de relevância experimental foi o robô SHAKEY, do Instituto de Pesquisa de Stanford, entre as décadas de 60 e 70. Esse robô foi desenvolvido com o objetivo de estudar as aplicações de inteligência artificial e estudar os processos para controle em tempo real de robôs que interagissem com ambientes complexos (Rezende,1992). O robô CART do mesmo instituto de pesquisa foi desenvolvido para estudos de visão estéreo.

Estes sistemas usavam computadores externos e operavam, na maioria das vezes, em ambientes estáticos, ou seja, em ambientes especialmente projetados para eles. No caso do SHAKEY, os experimentos incluíam a alteração do ambiente após muitas horas de execução, sem que o robô percebesse o efeito do ato dinâmico, a menos que tal mudança fosse relevante para a subtarefa que ele estivesse fazendo.

Apesar das simplificações, eles operavam muito lentamente. Levavam muito tempo construindo modelos do mundo e usavam pouco tempo de computação para planejar e agir.

Na época, a dificuldade de levar adiante as aplicações ao mundo real e a falta de instrumentação eficiente (computadores, sensores) retardaram o avanço da pesquisa para estes robôs. Houve falta de perspectivas palpáveis por parte dos pesquisadores, como por parte dos financiadores que queriam resultados imediatos. Desta forma, houve a ascensão dos robôs manipuladores de aplicação imediata e de resultados palpáveis a curto prazo. Com a tecnologia dos microprocessadores houve uma retomada do interesse pelos robôs móveis.

O Laboratório de Robôs Móveis, no Instituto de Robótica da Universidade de Carnegie-Mellon, tem desde 1982 projetado, construído e avaliado quatro protótipos de robôs móveis:

O robô PLUTO, com três rodas independentemente associadas, foi desenvolvido para o estudo dos problemas de controle e estabilização.

O Robô NEPTUNE que é um triciclo equipado com duas câmeras e 24 sensores de ultra-som distribuídos em sua volta.

O robô ILONATOR possui quatro rodas unidirecionais montadas simetricamente nos quatro vértices de um retângulo, resultando um movimento com três graus de liberdade. Um microprocessador embarcado faz a interface com os atuadores das rodas que possuem sensores (encoders ópticos) e o robô necessita de um computador para a comunicação dos comandos de alto-nível.

O robô TERRAGATOR é dotado de um motor à gasolina, ele se desloca a curtas distâncias em estradas e se orienta visualmente pelas bordas da pista.

Na França os robôs móveis mais representativos são:

O HILARE do *Laboratorie d'Automatique et d'Analyses des Systemes* (LAAS) que é utilizado para pesquisas sobre programação de robôs móveis e planejamento de movimento.

O robô móvel do projeto ICARE, do INRIA/Sophia Antipolis, que serve como plataforma experimental para o estudo de processamento de imagens em 3D.

Alguns sistemas mais representativos foram desenvolvidos no *Artificial Intelligence Lab. Massachusetts Institute of Technology* (MIT), estes são:

O robô PEBBLES, um robô móvel que usa uma câmera para evitar obstáculos em ambientes acidentados e desconhecidos.

Os microrobôs ANTS foram projetados para explorar idéias sobre a ação de cooperação entre robôs. Esse projeto tem como objetivos estender os limites da micro-robótica integrando muitos sensores e atuadores em um espaço pequeno, formando dessa forma uma comunidade robótica estruturada a partir de seus indivíduos. O projeto se inspirou na colônia de formigas.

O SOLEITE, um robô móvel de 30 gramas de massa, completamente autônomo e alimentado por energia solar. A energia coletada pelo painel é armazenada em um capacitor de 1F para a alimentação do robô. Tal robô serve também aos propósitos de exploração espacial.

Um dos locais mais representativos do estudo da robótica móvel no Brasil é o Laboratório de Robótica e Visão, do Instituto de Automação no Centro Tecnológico para a Informática LRV/IA/CTI, em Campinas-S.P. Os projetos se relacionam não apenas com robôs móveis, como também com sistemas não-holonômicos como veículos aéreos autônomos (dirigível e helicóptero). Os projetos robóticos e robôs do CTI são:

O projeto LEOPARD, que procura estudar a questão do controle de robôs móveis atuando de forma cooperativa, inclui os módulos de sensoriamento por visão computacional, estratégia de controle e comunicação por telemetria.

O robô NOMAD (Nomadic Technologies Company), de origem americana, é uma plataforma para estudos experimentais e possui sensores de proximidade de infravermelho e de ultra-som. Possui também dois anéis em sua base com sensores de tato e é equipado de uma câmera que faz parte do sistema de sensoriamento por visão. A navegação do robô é ainda auxiliada por uma bússola digital. O robô é utilizado em desenvolvimento de "softwares" de controle, percepção e navegação.

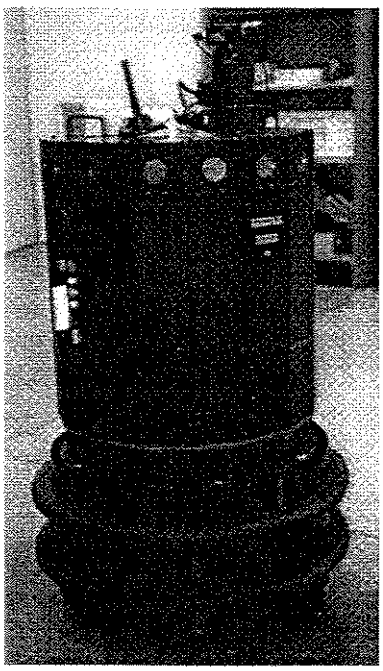


Figura 1.1: O Robô NOMAD.

O Robô KHEPERA, de origem suíça, trata-se de um robô móvel miniaturizado de seção circular com raio de 2,5 cm e que se move através de duas rodas fixas impulsadas por motores de corrente contínua independentes. Dispõe de módulos suplementares de visão, telemetria e garra mecânica. Em sua circunferência existem oito sensores de infravermelho que detectam a proximidade do obstáculo. É utilizado no estudo de controle e estabilização de sistemas não-holonômicos.

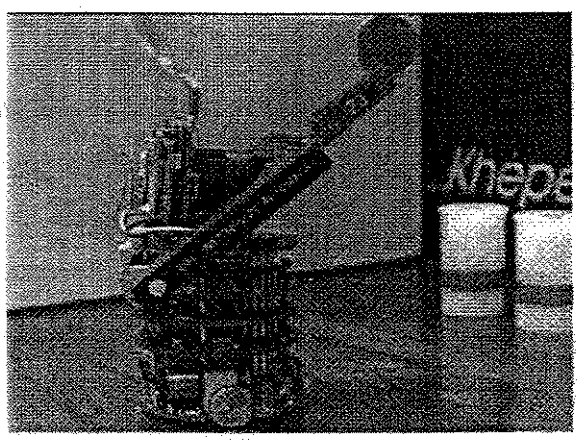


Figura 1.2 : O Robô Khepera

É importante mencionar também que o Laboratório de Automação e Sistemas da Escola Politécnica de São Paulo dispõe de um AGV (Autonomous Guide Vehicle). O ARIEL, uma base experimental construída no mesmo laboratório, possui duas rodas motrizes acionadas por motores de corrente contínua, além de possuir um computador a bordo para operações de controle e interfaceamento.

O laboratório de robótica da Universidade Federal de Minas Gerais começa a ser montado com a aquisição de um robô NOMAD, sendo o segundo no Brasil.

### 1.4.1 Aplicações

Terrorismo, envelhecimento da população, mudanças de clima, e escassez de recursos, são alguns dos desafios do mundo de hoje. Mas as máquinas inteligentes ajudarão a resolver estes e outros problemas sociais, ambientais e de defesa. Nestes dias já é possível fabricar robôs que servem quase para qualquer operação, seja de limpeza, operações cirúrgicas a distancia, cortar a grama, fazer o chá, etc. O único limite é a nossa imaginação. Existem micromáquinas (colocados num relógio de pulso, por exemplo) capazes de obter dados de temperatura, pressão, pulso, etc. Um computador recolherá estes dados e diagnosticará em tempo real, acionando outros robôs, que por exemplo estejam carregados de pílulas.

No futuro, robôs móveis com biosensores poderão ajudar aos soldados frente às armas químicas, identificando a presença e a natureza dos gases, alertando assim aos humanos. Robôs móveis voadores poderão monitorar a poluição no meio ambiente.

A disponibilidade de nova tecnologia permite a formação de novas idéias nas áreas de inteligência artificial, robótica, micro-máquinas e materiais inteligentes. Os sistemas de inteligência para robôs autônomos móveis permitem agora, comportamentos simples quase ao nível de insetos. Estes avanços em "software", junto com nova tecnologia para a fabricação de micro-sensores e micro-atuadores, oferecem a oportunidade de termos no futuro robôs pequenos, de baixo custo, e potencialmente de muita utilidade para tarefas como a cirurgia.

Como exemplo de um futuro próximo, no Laboratório de Robótica Móvel do MIT, novos enfoques na inteligência artificial aplicados aos robôs móveis autônomos, fazem com que eles explorem, caminhem, interajam com pessoas e aprendam a coordenar vários comportamentos internos. Este tipo de sistema de controle, conhecido como Arquitetura *Subsumption*, mostra-se como uma rede distribuída em camadas com capacidade aumentada de transmissão de mensagens, permitindo uma estreita relação entre a percepção e a ação que se deve ter no mundo dinâmico de um robô móvel (Flynn, 1996).

O SQUIRT, o menor robô construído com esta arquitetura, mede 2,5 cm, tem a bordo um computador, dois microfones, um sensor de luz, motor e baterias. O programa que permite que ele possa ocultar-se na escuridão ou possa aproximar-se de um determinado som, é de 1,300 bytes. Está dotado de um motor DC, que só lhe permite um grau de liberdade.

Seria possível imaginar que o computador, os sensores e as baterias coubessem num espaço reduzido como o de SQUIRT. Ainda assim, este mesmo espaço não permitiria incluir motores suficientes que permitissem ao robô uma maior destreza.

São necessários pequenos motores, compactos, de transmissão direta, de custo razoável e cujo torque útil possa se adaptar à carga. Um motor com estas qualidades é o motor piezo - elétrico ultra-sônico. Estes motores podem funcionar sem engrenagens, com sistema de transmissão direta, que pode reduzir de forma considerável o peso do robô móvel.

Como exemplo da utilização destes motores tem-se: um carro que leva uma câmera; este mede aproximadamente um centímetro cúbico e conta com mais movimentos que o Squirt. Micro-Trepadores; as caraterísticas do piezo-motor (alto torque e baixa velocidade), permitem que possa ser usado de forma direta para acionar cada uma das seis pernas deste robô. Micro-Submergível; um robô submarino, de 2,5 cm de comprimento, incorpora um micro motor piezo-elétrico para propulsão.

O robô CLEO é mostra do desenvolvimento nesta área. Este é um robô autônomo altamente integrado, com mais sensores e atuadores que o Squirt, sendo do mesmo tamanho. Contém 3

motores, 17 sensores, um computador, uma garra, bateria, e um conversor DC-AC. O sistema de inteligência gera comportamentos de formiga e é implementado com a arquitetura de assunção e lógica percepção-ação. O cérebro de CLEO foi implementado em um microprocessador MOTOROLA 68HC11E2, com 256 bytes de RAM e 2K de memória, somente para leitura (EEPROM).

Nos laboratórios do *Sandia's intelligent System and Robotic Center* (ISCR) tem sido desenvolvidos robôs e máquinas inteligentes (RIM's) com flexibilidade pare executar múltiplas tarefas. Estes serão apresentados a seguir (Sandia Technology, 2000):

O robô REATLER (*Robotic All Terrain Lunar Exploration Rover*) pode cooperar com outros robôs sentinelas para proteger um perímetro. Este robô foi desenvolvido para uma missão à lua. Está equipado com um CHIP de computador Intel, um sistema receptor de posicionamento global, dois rádios, uma bússola e sensores de inclinação, câmera de vídeo, etc.

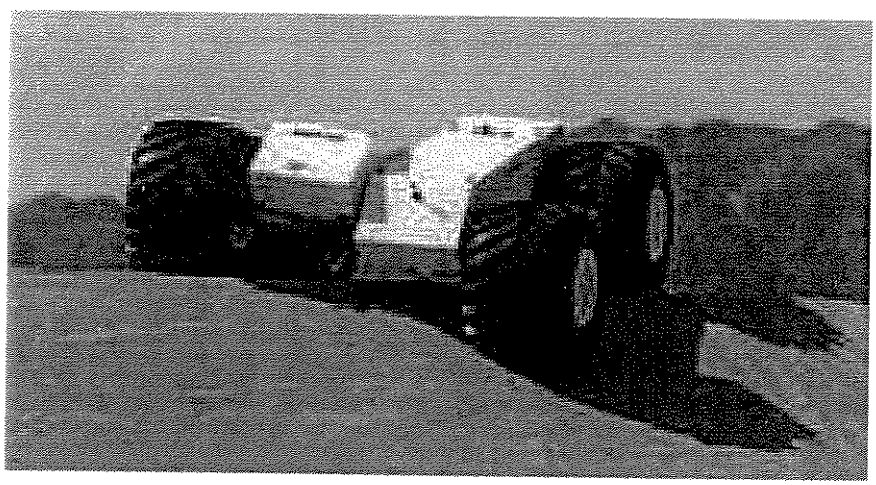


Figura 1.3 : O Robô REATLER

O robô FIRE ANT é um exemplo de um sistema móvel autônomo destrutor de blindados, usa um projetil do tipo explosivo (EFP). Um detetor de movimento por vídeo proporciona o alvo a seguir. A plataforma é teleoperada a partir de um ponto de observação. Na versão inicial, o robô é destruído quando chega ao alvo. Nas versões novas, os robôs não são destruídos, usando novos projetis mais letais (EFPs).

O robô MARV (*Miniature Autonomous Robotic Vehicle*) prepara o caminho para pequenos veículos com mobilidade, inteligência, navegação e comunicação, e apresenta ainda a capacidade de comportamento cooperativo. Tendo só uma polegada cúbica (16 centímetros cúbicos), o minirobô MARV carrega fonte, sensores, um computador e os controles necessários na mesma placa, para localizar e seguir fios que levam predeterminados sinais de rádio.

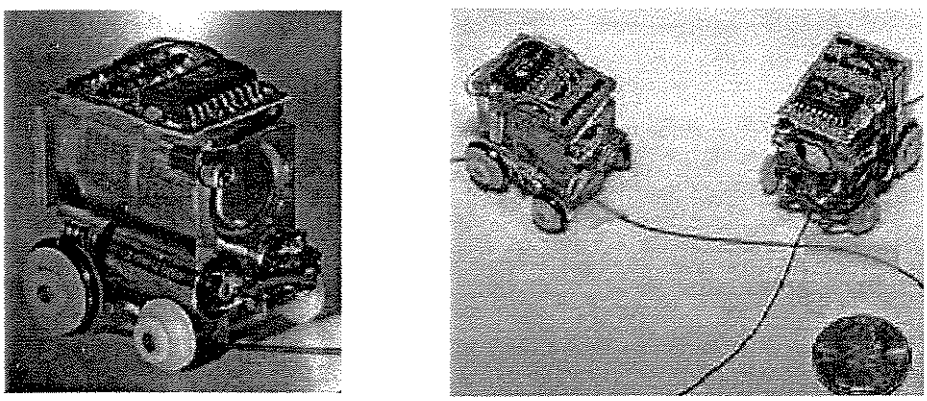


Figura 1.4 : O Robô MARV

O MINI-ROBOT, Com 4 centímetros cúbicos de volume e pesando menos de 28 gramas, possui três pilhas de relógio como fonte, rodas a tração, um processador de 8K ROM, sensor de temperatura e dois motores. As melhorias que estão sendo consideradas no futuro incluem uma câmera em miniatura, microfone, dispositivo de comunicação e um microsensor químico. É um produto do *Laboratories Directed Research and Development* (LDRD) continuando um trabalho iniciado no *Sandia's Intelligent Systems Sensors & Controls Department.*

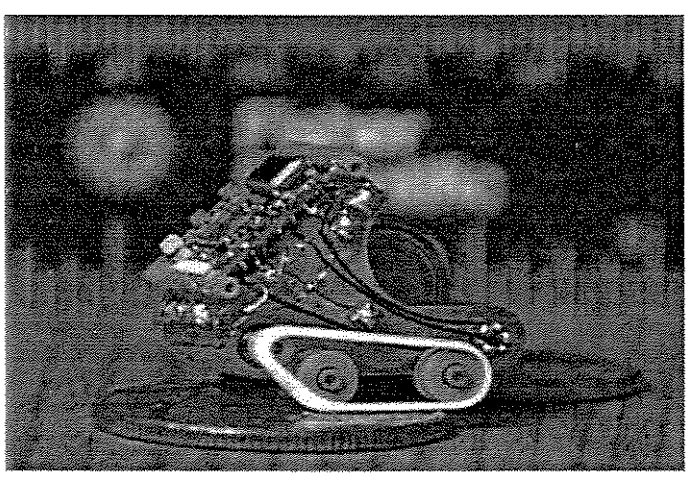


Figura 1.5 : MINI-ROBOT girando numa moeda de dez centavos.

## 1.5 Objetivos deste Trabalho

Como apresentado neste capítulo, existe um enorme potencial e interesse na aplicação dos robôs autônomos móveis, pois a sua utilização abarca a maioria das áreas da vida humana. Alem de existir este potencial na utilização deste tipo de robôs, na área acadêmica existem poucos trabalhos dedicados à verificação de dados teóricos com dados experimentais adquiridos com um robô real, e não por simulações.

Na necessidade de formar um elo entre o teórico e o prático, e assim poder proporcionar aos futuros trabalhos na área da robótica autônoma móvel uma ferramenta ("plataforma de testes") que facilite a validação experimental, é que surge a necessidade de desenvolver um robô autônomo móvel com características apropriadas para a experimentação, e também fornecer ao futuro pesquisador as ferramentas que o ajudem a sobrelevar os diferentes problemas na utilização deste robô.

O objetivo principal deste trabalho é o desenvolvimento de um robô autônomo móvel, chamado **RAM-I** utilizando a Arquitetura *Subsumption*, para ser usado como base de testes de futuros trabalhos na área da robótica autônoma móvel.

Para a realização deste trabalho, implementação de um robô autônomo móvel, foi necessário seguir vários passos: escolha de um microcontrolador adequado às necessidades do projeto, conhecimento da linguagem de programação, escolha dos módulos de "hardware", seleção dos motores, dos tipos de sensores a ser usados, baterias, desenho da plataforma móvel, montagem do conjunto, e testes finais.

## 1.6 Desenvolvimento de um Robô Autônomo Inteligente: RAM-I.

Para este trabalho foi escolhido desenvolver um robô de plataforma retangular sobre três rodas, sendo duas delas motrizes e a terceira de apoio.

Para que servisse de exemplo, foi decidido que o robô deveria ter a habilidade de vagar num plano, evitando obstáculos e depressões. Para atender a estas especificações foi necessário

escolher dimensões, motorização, sensores, processador, estratégia de controle, etc, sempre atendendo ao objetivo maior, que é o de deixar uma plataforma flexível o suficiente para que possa ser reaproveitada para outros trabalhos.

### 1.6.1 Considerações Iniciais

A necessidade de aplicação da Inteligência Artificial em robótica não é nova. Uma das grandes dificuldades que se tem, está na complexidade de uma definição rigorosa da palavra "inteligência", para definir se um robô é ou não "inteligente". Talvez seja mais prudente, aceitar que o ser humano detenha a exclusividade de exibir o que o consenso geral considera inteligente, e através da identificação de alguns de seus comportamentos mais simples, mas que o consenso geral os tenha como inteligentes, buscar o desenvolvimento de sistemas artificiais que sejam capazes de simular estes comportamentos (Rezende, 1992). Robôs Móveis Autônomos Inteligentes será a denominação adotada para robôs que simulam comportamentos que o consenso geral classifica como inteligentes.

Em termos de comportamento (por exemplo o retorno a um determinado lugar, no que se refere a desviar, ultrapassar obstáculos e alcançar um objetivo), não há maior diferença entre ser humano, formiga e autômato, podendo exibir um mesmo tipo de comportamento. Por que então, somente apenas a um desses objetos se atribui inteligência? Pode–se argumentar que ao ser humano, e não aos demais, são atribuídas façanhas como pensamento, auto-conciência, etc.

No caso dos homens, existem pessoas que têm maior habilidade para aprender certas tarefas, estas pessoas são chamadas de "mais inteligentes". Será que a inteligência pode ser mensurada? Então por que não classificar a inteligência da formiga e do autômato como uma espécie de inteligência inferior à do ser humano, mas ainda assim inteligência?.

Neste trabalho se apresenta um robô autônomo móvel considerado inteligente por apresentar comportamentos que, quando observados por um ser humano, demostram características de um nível de inteligência inferior.

## 1.6.2 Interface sistema-ambiente

Pode-se estabelecer um conjunto mínimo de condições intrínsecas (denominadas **Potenciais Singulares de Comportamento**) para que um *sistema* possa ser considerado Artificial Inteligente, e um conjunto mínimo de condições, intrínsecas ao *ambiente*, para que possa ser considerado favorável ao referido sistema.

No exemplo do trajeto de retorno realizado por uma formiga ao formigueiro, ilustra-se um *sistema* (formiga) perfeitamente adaptado a seu *ambiente*. Algumas propriedades ou hipóteses de potencial que permitem ao *sistema* realizar o trajeto são (Rezende, 1992):

- **P1 - Capacidade de internalização de objetivos:** A formiga possui objetivos próprios, isto é, internalizados. Ela quer retornar ao formigueiro.

- **P2 - Capacidade de sentir e perceber seu meio ambiente:** A formiga é capaz de sentir e perceber no seu ambiente externo impedimentos à sua caminhada em direção ao formigueiro.

- **P3 - Capacidade de modificar parcialmente os objetivos internalizados:** Ao perceber uma situação inusitada, a formiga modifica parcialmente seus objetivos internalizados, porém a medida que a dificuldade passe a formiga retorna a priorizar o objetivo anterior.

- **P4 - Potencial para movimento intencional:** Observa-se que todas as decisões são tomadas pela formiga e é ela quem dispara um macro processo de ativação de movimentos coordenados, intencionalmente, o que evidencia uma capacidade ou potencial para movimento intencional.

Alguns autores colocam esta subdivisão como níveis de inteligência artificial. Existem outros potenciais intrínsecos, mas os quatro citados são suficientes para que um robô ou um *sistema* possa ser considerado inteligente. Na ausência de alguma das capacidades, a formiga

não teria capacidade por si só, de retornar ao formigueiro, o que indica que este sistema tem um certo nível de inteligência.

As propriedades que fornecem condições para que o *ambiente* se constitua favorável a um referido sistema, são:

- **C1 - Permite sua auto-movimentação:** O sistema tem que ser favorável para a própria movimentação.

- **C2 - Possuir energia alcançável pelo sistema:** Há necessidade que o ambiente possua energia, e que o sistema possa sozinho ser capaz de alcançar tal energia.

- **C3 - Não possuir intempéries fatais ao sistema:** O ambiente não precisa possuir substâncias nocivas ao sistema. Por exemplo no caso da formiga, não ter substancias tóxicas que a mate.

Um autômato apesar de ser completamente diferente, a nível microscópico ("hardware"), do que uma formiga, poderia simular o simples comportamento de uma formiga retornando ao formigueiro. Para isso o autômato teria que contemplar minimamente os quatro potenciais acima citados e o ambiente no qual está desenvolvendo-se o comportamento, ainda teria que lhe ser favorável nas condições acima citadas. Com estas caraterísticas (condições) pode-se "assegurar" a correta adaptação de um *sistema* a um ambiente que lhe seja favorável.

## 1.7 Organização deste Trabalho

A dissertação está organizada de acordo com os seguintes capítulos:

- O Capítulo 2 está dividido em duas partes: Na primeira parte são apresentados os microcontroladores em geral, critérios para a escolha do microcontrolador, descrição e funcionamento dos microcontroladores PIC (em particular o PIC16F84), e uma breve introdução ao Assembler do microcontrolador utilizado. Na segunda parte descreve-se os módulos, sensores e atuadores utilizados para a implementação do robô **RAM-I**.

- No capítulo 3, é feita uma apresentação do princípio de percepção-ação e da Arquitetura *Subsumption*, desenvolvida no laboratório de inteligência artificial do MIT (Massachussets Institute of Technology), usadas na implementação de robôs autônomos móveis, alguns exemplos da implementação desta arquitetura em alguns robôs, e a descrição do modelo aplicado ao robô **RAM-I.**

- No capítulo 4, é feita uma modelagem matemática dos robôs móveis com rodas, incluindo a descrição dos tipos de rodas, restrições de movimento, e os modelos de estado derivados do modelamento das rodas. No inicio trata-se de generalizar o modelamento para robôs móveis com rodas mostrando vários exemplos. Depois é feita a modelagem para o robô **RAM-I**, a qual ajudará aos futuros usuários deste robô.

- No Capítulo 5, são apresentados e discutidos os resultados, mostrando-se primeiramente o robô implementado, e logo a seguir é mostrado o robô na execução de algumas tarefas num ambiente de trabalho.

- Finalmente, o Capítulo 6 apresenta as conclusões e sugestões para trabalhos futuros.

# Capítulo 2

# Concepção de um Robô Autônomo Móvel

## 2.1 Introdução

Neste capitulo serão apresentados os materiais e métodos utilizados para o desenvolvimento do presente trabalho. O **RAM-I** é uma base móvel inteligente, implementada com módulos de "hardware" baseados no microcontrolador PIC 16F84 da série PIC da MICROCHIP. Na primeira parte se descreve os principais tipos de microcontroladores, critérios para a escolha do microcontrolador, descrição e funcionamento dos microcontroladores PIC (em particular o PIC16F84, escolhido para o **RAM-I**), e se faz uma introdução à programação em "Assembler" e às principais ferramentas para a programação do PIC.

Na Segunda parte, descreve-se os módulos utilizados para a implementação do **RAM-I**. O primeiro módulo é o ***Sb Step Control***, que é o encarregado de controlar os motores de passo, fazendo com que eles obedeçam os comandos simples: velocidade de giro, inversão de giro, parar os motores, etc. O segundo módulo é o ***Sb 9902-2***, e este é o denominado "cérebro" do robô, por conter o microcontrolador PIC programado com a arquitetura de controle do mesmo.

A última parte deste capitulo descreve os atuadores (motores de passo) e sensores empregados no **RAM-I**.

## 2.2 "Hardware" Computacional

Os circuitos que conformam o controle do **RAM-I** estão baseados no microcontrolador PIC16F84 da empresa MICROCHIP. Este tipo de dispositivo têm varias vantagens sobre o "hardware" lógico: versatilidade, baixo consumo de potência, tamanho, etc. É importante notar que os microcontroladores e os microprocessadores introduzem uma ferramenta significativa à solução do problema de controle dos robôs: o "software". O comportamento do robô baseado em "software", poderá ser modificado simplesmente com a implementação de um novo programa sem necessidade de alterações físicas. Todas estas vantagens foram alcançadas com a utilização do microcontrolador PIC 16F84, no presente trabalho.

É importante ter em mente que o "hardware" determina o potencial final de um robô, mas o que permite realizar este potencial é o trabalho do "software" (Jones, 1993).

## 2.3 Projetos com Microcontroladores

Em poucas palavras pode-se dizer que um microcontrolador é um "pequeno" componente (dispositivo) eletrônico, dotado de uma "inteligência" programável, utilizado no controle de processos lógicos (Souza, 2000).

Controle de processos pode ser definido como o controle de periféricos tais como: "leds", botões, "displays" de segmentos, "displays" de cristal liquido (LCD), resistências, reles, sensores (pressão, temperatura, etc.), motores de passo, servo motores e outros periféricos. É usada também a palavra controle lógico pois a operação do sistema baseia-se nas ações lógicas que devem ser executadas, dependendo do estado dos periféricos de entrada e/ou saída.

O controlador é programável pois a lógica da operação é estruturada na forma de um programa e gravada dentro do componente. Depois disso, toda vez que o microcontrolador for alimentado, o programa interno será executado. Quanto à "inteligência" do componente, podemos associa-la à Unidade Lógica Aritmética (ULA, em inglês ALU), que é a encarregada de

executar as operações matemáticas e lógicas. Quanto mais poderosa a ULA do componente, maior a sua capacidade de processar informações.

O microprocessador é dito "pequeno" pois em uma única pastilha de silício encapsulada (popularmente chamada de CHIP ou CI) se tem todos os componentes necessários para que o controle de um processo seja possível.

O microcontrolador está provido internamente de memória de programa, memória de dados, portas de entrada e/ou saída paralela, "timers" (ou contadores de tempo), contadores, comunicação serial, PWMs, conversores analógico-digital, etc. Esta é uma das caraterísticas fundamentais que diferencia os microcontroladores dos microprocessadores, pois estes últimos, apesar de possuírem uma ULA muito mais rápida e poderosa, não possuem todos estes recursos em uma única pastilha. Uma outra definição de microcontrolador é a de ser um microprocessador acrescido de endereçadores de I/Os e de memória, memória de programa (PROM, EPROM, EEPROM), memória RAM, tudo isto em uma única pastilha (CI).

Atualmente muitos equipamentos de uso diário, tais como: eletrodomésticos, videocassetes, alarmes, celulares e brinquedos, entre outros, utilizam microcontroladores para a execução de suas funções básicas.

As principais vantagens no uso de microcontroladores são a diminuição do custo e a diminuição de "hardware". As vezes o uso dos microcontroladores não é viável já que não é recomendável o uso destes para rotinas simples, pois seria bem mais barato utilizar componentes convencionais (sistemas analógicos).

Basicamente todos os microcontroladores de baixo custo, no mercado atual, são dispostos de forma semelhante. As principais diferenças entre eles são as seguintes:

- Disposição do "hardware" interno: memória (RAM, FLASH, SPI, UART, etc..).
- Capacidade de memória de programa (EPROM, OTP, FLASH).
- Modo de gravação.
- Mnemônicos do "Assembler".

- E, principalmente: "O apoio dado pelo fabricante/distribuidor no início do projeto", fornecendo ferramentas, amostras, programas exemplo, apoio técnico, etc. Tudo isso é chamado de "Kit de Desenvolvimento".

A escolha correta do microcontrolador para um determinado projeto torna-se difícil dado que existe no mercado atual uma vasta variedade de tipos, famílias e fabricantes de microcontroladores, as quais devem ser bem estudadas antes de se iniciar o projeto. Devem ser tomadas em conta as seguintes considerações:

- Qual é a capacidade do microcontrolador que se está precisando?. Que trabalho ele vai realizar?.
- Número de I/Os necessários para o projeto.
- "Hardware" interno disponível, por exemplo se possui uma UART ("Universal Assyncrhonous Receiver and Transmiter") interna para rotinas de comunicação serial.
- Se requer WATCHDOG ("cão de guarda").
- Facilidade de aquisição no mercado.
- Custo do "Kit de Desenvolvimento".
- Custo final do produto.

Após analisar estas considerações e outras que se apresentem devido à particularidade do projeto, é possível decidir-se pelo microcontrolador mais apropriado, mas só se terá certeza de ter atingido o objetivo principal do projeto (versatilidade), se dispor de um projeto que possa utilizar mais de um microcontrolador de famílias ou fabricantes diferentes, para evitar o inconveniente de ser informado que o microcontrolador está saindo de linha.

Existem vários fabricantes de microcontroladores: Microchip, Intel, Motorola, National, Atmel, Siemens, Hitashi, TI, Toshiba, são alguns dos mais conhecidos.

A Tabela 2.1 mostra um resumo das famílias de microcontroladores do fabricante MOTOROLA.

Tabela 2.1: Famílias e tipos de microcontroladores fabricados pela MOTOROLA

| Microcontrolador | Família | Tipo |
|---|---|---|
| 8 bits | M68HC05 | Microcontroladores de 8 bits 68HC05 |
| 8 bits | M68HC08 | Microcontroladores de 8 bits 68HC08 |
| 8 bits | M68HC11 | Microcontroladores de 8 bits 68HC11 |
| 16 bits | M68HC12 | Família de produtos 68HC12 |
| 16 bits | M68HC16 | Família de microcontroladores 68HC16 |
| 32 bits | M68300 | Microcontroladores de 32 bits 683XX |
| 32 bits | MPC500 | Microcontroladores de 32 bits |
| 32 bits | MCORE | Microcontroladores de 32 bits MCORE |

A Tabela 2.2 mostra um resumo das famílias e tipos de microcontroladores do fabricante INTEL. Além dos mostrados na tabela existem microcontroladores específicos para a linha automotiva.

Tabela 2.2: Famílias e tipos de microcontroladores fabricados pela INTEL

| Microcontrolador | Família | Tipo |
|---|---|---|
| MCS®51/151/251 | MCS® 51 | Clássico (8 bits) |
| | | De alta velocidade |
| | | Memória RAM expandida |
| | | Para ambientes especiais |
| MCS®51/151/251 | MCS® 151 | 8XC151AS / SB |
| MCS®51/151/251 | MCS® 251 | 8XC251AS / SB / SP / SQ |
| | | 8XC251TA / TB / TP / TQ |
| MCS®96/296 | MCS®96-HSIO* | 8XC196KB |
| | | 8XC196KC |
| | | 8XC196KD |
| MCS®96/296 | MCS®96-EPA** | 8XC196KR |
| | | 8XC196KT |
| | | 8XC196NT |
| | | 8XC196NP |
| | | 8XL196NP |
| | | 80C196NU |
| | | 80C196EA |
| MCS®96/296 | MCS®96 | 8XC196MC/MH |
| | | 8XC196MD |
| MCS®96/296 | Família de produtos CAN*** | (Controller Area Network) |

*High-Speed Input/Output, **Event Processor Array, ***Controller Area Network.

Um dos principais produtos da National Semiconductor é o microcontrolador da série COP8 (Processador Orientado ao Controle de 8 bits) bem difundido no Brasil. A Tabela 2.3 mostra um resumo das famílias de microcontroladores COP8 e seus diferentes tipos.

Tabela 2.3: Famílias e tipos de microcontroladores COP8 fabricados pela National Semiconductor.

| Família | Caraterísticas |
|---|---|
| **Família básica em máscara** | **Dividido na Temperatura, Tamanho de memória e outras Caraterísticas (não são reprogramáveis).** |
| **Família com características em máscara** | **Orientadas a comunicações, não são reprogramáveis.** |
| **Família OTP** | **Versão que permite um única programação.** |
| **Família S** | **Família COP8AS** |
| | **Família COP8SG** |
| | **Família COP8AC** |
| | **Família COP8SB/ COP8CB** |

A MICROCHIP inicia suas operações no Brasil pelos anos 90 através da sua parceira, a empresa Aplicações Eletrónicas Artimar Ltda, representante de várias empresas americanas no setor da eletroeletrônica. Apresenta-se na Tabela 2.4 um resumo das famílias de microcontroladores da série PIC (Peripheral Interface Controller).

Tabela 2.4: Famílias de microcontroladores PIC fabricados pela MICROCHIP.

| Microcontrolador | Família | Caraterística |
|---|---|---|
| 8 bits | PIC12C5XX | 8 pinos |
| 8 bits | PIC12CEXX | 8 pinos com memória de dados EEPROM |
| 8 bits | PIC12C67X | 8 pinos com conversor A/D |
| 8 bits | PIC12CE67X | 8 pinos com conversor A/D e memória de dados EEPROM |
| 8 bits | PIC16C5X & PIC16HV540 | EPROM/ROM |
| 8 bits | PIC16C55X | EPROM |
| 8 bits | PIC16C6X | |
| 8 bits | PIC16C62X | 18 pinos com EPROM |
| 8 bits | PIC16C64X & PIC16C66X | Com EPROM e comparador analógico |
| 8 bits | PIC16CE62X | Com comparador analógico, memória de dados EEPROM |
| 8 bits | PIC16C7X | Com conversor A/D |
| 8 bits | PIC16C71X | 18 pinos com conversor A/D |
| 8 bits | PIC16C43X | 18/20 pinos com bus LIN |
| 8 bits | PIC16C78X | Com conversor programável A/D, D/A, amplificador operacional. |
| 8 bits | PIC16C7XX | Com conversor A/D, de USB, PS/2 e aplicações seriais do dispositivo. |
| 8 bits | PIC16C77X | 28/40 pinos com conversor A/D de 12 bits |
| 8 bits | PIC16F87X | 28/40 pinos, FLASH, com conversor A/D de 10 bits |
| 8 bits | PIC16X8X | 18 pinos, FLASH/EEPROM |
| 8 bits | PIC16F7X | 28/40 pinos, FLASH |
| 8 bits | PIC17C4X | EEPROM/ROM, de alta performance |
| 8 bits | PIC17C7XX | De arquitetura realçada |
| 8 bits | PIC18F0XX | 8 pinos, FLASH, EEPROM, A/D e PWN |
| | PIC18FXX2 | Com proteção de código, EEPROM, modo SLEEP |
| | PIC18FXX8 | Filtros, duas máscaras, buffers 2RX |
| | PIC18FX32 | 28/40 pinos, FLASH, EEPROM, com modo baixo de potência, com conversor A/D de 10 bits |
| | PIC18FXX31 | 28/40 pinos, para o controle de motores, EEPROM, com modo baixo de potência, com conversor A/D de 10 bits |
| | PIC18FXX5 | 28/40 pinos, FLASH, com conversor A/D de 10 bits |
| | PIC18FXX30 | 18/20 pinos, para o controle de motores, EEPROM, com modulo PWM de 3 fases |

Depois de estudar a imensa variedade de microcontroladores e além disso as características principais dos mesmos, tem-se de reduzir os candidatos a aqueles que são oferecidos no mercado nacional. O preço do microcontrolador, como foi dito anteriormente, tem pouca influencia na escolha pois geralmente o dispositivo (CHIP) é de baixo custo. Um dos fatores que pesam mais na hora da escolha é o apoio do fabricante no inicio do projeto: informações, cursos, programas exemplo, custo do "hardware" de gravação (gravador). Geralmente o fabricante oferece no seu "site" o "software" necessário para o desenvolvimento do projeto, mas nem todo "software" é sem custo. Aqueles microcontroladores que estejam mais difundidos com certeza serão os primeiros escolhidos.

O PIC16F84, bastante difundido no Brasil, é um microcontrolador da família PIC16X8X com memória de dados e programa tipo FLASH (reprogramável), com 18 pinos, tamanho standard de qualquer CI de 18 pinos que facilita o desenho do "hardware". Ele tem 13 portas de entrada ou saída, suficientes para o projeto. Possui um "set" de 35 instruções, ótimo para ser usado no aprendizado e desenvolvimento (prototipagem) de programas de controle de motores. O "Kit de Desenvolvimento" não é de custo elevado, com muita informação, e exemplos da sua utilização. Além disto, o PIC pertence a um família grande de microcontroladores, os que poderiam satisfazer futuras necessidades, sem precisar de mudanças significativas na programação. Por todas estas razões o PIC16F84A foi o microcontrolador escolhido para ser usado na implementação do **RAM-I**.

## 2.4 Microcontroladores PIC da MICROCHIP

O PIC é um microcontrolador que teve suas origens no ano de 1965, quando a companhia GI (General Instruments), formou a Divisão de Microeletrônica. Na década 70 a GI criou um dos primeiros processadores de 16 bits chamado CP16000. Este tinha uma certa deficiência no processamento de entradas/saídas. Então a GI projetou e construiu um Controlador de Interface Periférica (em inglês Peripheral Interface Controller, ou PIC). Ele tinha que ser bem rápido e possuir um conjunto de instruções muito pequeno. O CP16000 não teve muito sucesso, porem o PIC evoluiu para o PIC16C5X. Com o tempo esta divisão se transformaria na Arizona Microchip Technology desenvolvendo várias versões do PIC, sendo a versão com EEPROM (Electrical

Erasable PROM, EPROM apagável eletricamente) a mais indicada para o desenvolvimento de aplicações (Escola Virtual Solbet, 2000).

O dispositivo mais avançado disponível hoje da série PIC e que ao mesmo tempo reúne uma série de características adequadas ao uso do mesmo em pequenas series de produtos, facilitando o estudo dos microcontroladores, é o PIC16F84

Os microcontroladores PIC apresentam uma estrutura de maquina interna do tipo Harvard, enquanto grande parte dos microcontroladores tradicionais apresenta uma arquitetura do tipo Von-Neumann. A diferença está na forma como os dados e o programa são processados pelo microcontrolador. Na arquitetura tradicional, tipo Von-Neumann, existe apenas um barramento (bus) interno (geralmente de 8 bits), por onde passam as instruções e os dados. Na arquitetura Harvard existem dois barramentos internos, sendo um de dados e outro de instruções. No PIC , o barramento de dados é de 8 bits e o de instruções pode ser de 12, 14 ou 16 bits, dependendo do microcontrolador, permitindo que enquanto uma instrução é executada outra seja "buscada" (lida) da memória, o que torna o processamento mais rápido. Outra vantagem desta arquitetura está no barramento de instruções, que sendo maior do que 8 bits, já inclui o dado e o local onde ela vai operar (quando for necessário), o que possibilita que apenas uma posição de memória seja utilizada por instrução, economizando assim muita memória do programa.

Os PICs utilizam uma tecnologia chamada RISC, Reduced Instruction Set Computer (Computador com "set" de instruções reduzido), possuindo cerca de 35 instruções (o número correto varia de acordo com o microcontrolador), muito menos do que os microcontroladores convencionais que chegam a possuir mais de 100 instruções. Isto torna o aprendizado muito mais fácil e dinâmico, mas requer também maior habilidade do programador para construir funções que não possuam instruções diretas.

### 2.4.1 Estrutura Interna

No diagrama de blocos da Figura 2.1 (Souza, 2000) podem ser visualizadas as diversas partes que compõem o microcontrolador PIC16F84. Nela se observa claramente como o ULA

(Unidade Lógica Aritmética, em inglês ALU) está diretamente ligada ao registrador W ("Work reg"). No canto superior esquerdo se tem a memória do programa ("Program Memory"), saindo deste bloco temos o barramento de 14 bits ("Program Bus 14"). Ao meio se encontra a memória de dados (RAM) com seu barramento de 8 bits ("Data bus 8"). Ao lado direito o primeiro periférico, a EEPROM (memória de dados não volátil). Outros periféricos podem ser também observados, tais como: o timer (TMRO) e as portas paralelas (I/O Ports). Observe-se que ao timer está ligado um pino do microcontrolador (RA4/TOCK); este pino, em conjunto com o timer, forma o contador externo. No centro está o registrador STATUS (STATUS reg). Algumas informações importantes sobre as operações aritméticas da ULA ficam armazenadas neste registrador. Na parte superior se têm o contador de linhas de programa (Program Counter) e a pilha de 8 níveis (Stack). Entre todos os periféricos a comunicação é feita por meio de um barramento de 8 vias. Alem destes dispositivos existem os circuitos internos de reset, osciladores e o Watch dog Timer (WDT).

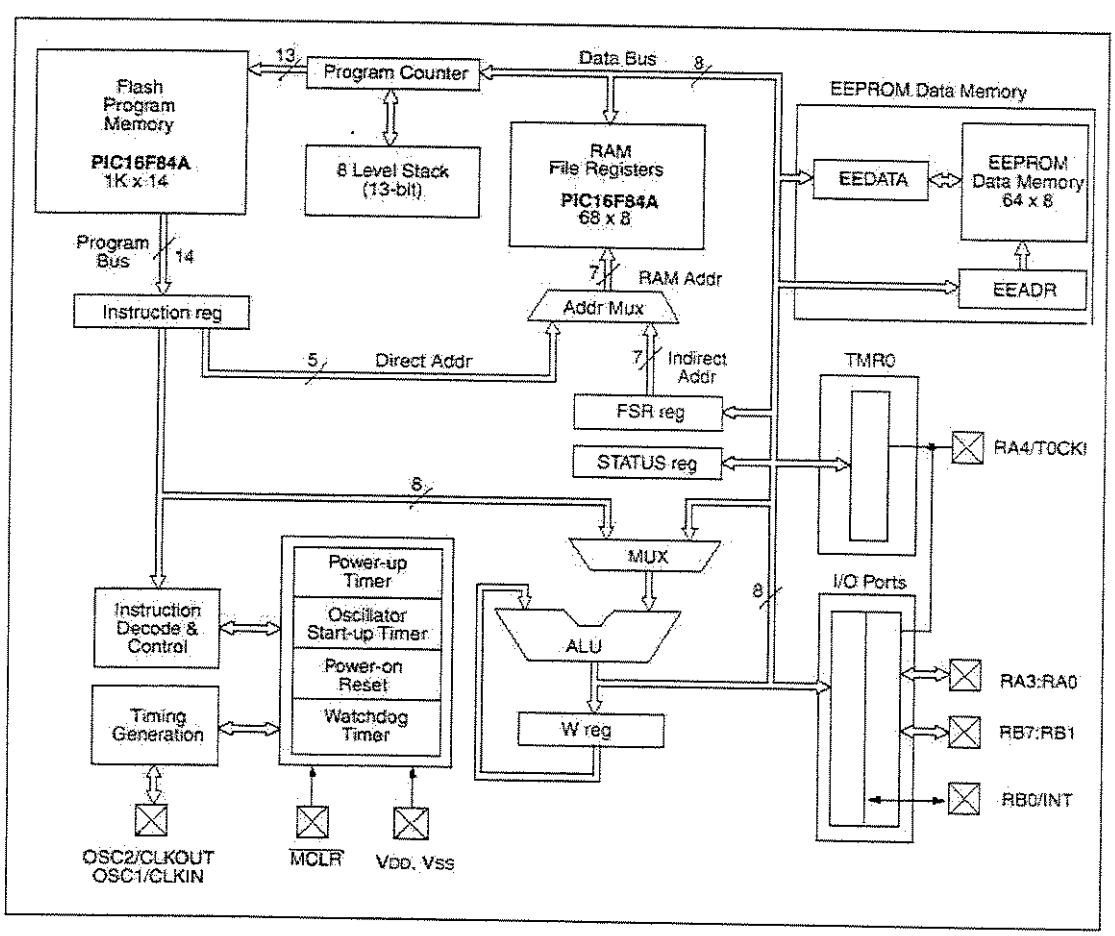



Figura 2.1: Diagrama interno do PIC 16F84.

Na figura 2.2 é mostrado o diagrama de blocos simplificado do microcontrolador. O PC (contador de programa) é onde está armazenado o endereço da instrução que será executada. Após cada instrução o valor do PC é alterado para passar a indicar o endereço da próxima instrução. No caso da instrução CALL e no atendimento das interrupções o microcontrolador deve manter a informação do endereço de retorno, para saber aonde ir quando voltar da subrotina. Este endereço de retorno é armazenado no STACK. No caso do PIC16F84 é permitido o alinhamento de até oito subrotinas ou interrupções. A ALU interpreta e executa os programas armazenados na área de programas. A ALU trabalha em estreito relacionamento com o registro W, onde grande parte das operações são realizadas. Os demais blocos selecionados são os responsáveis pela sincronização das operações dentro do microcontrolador.

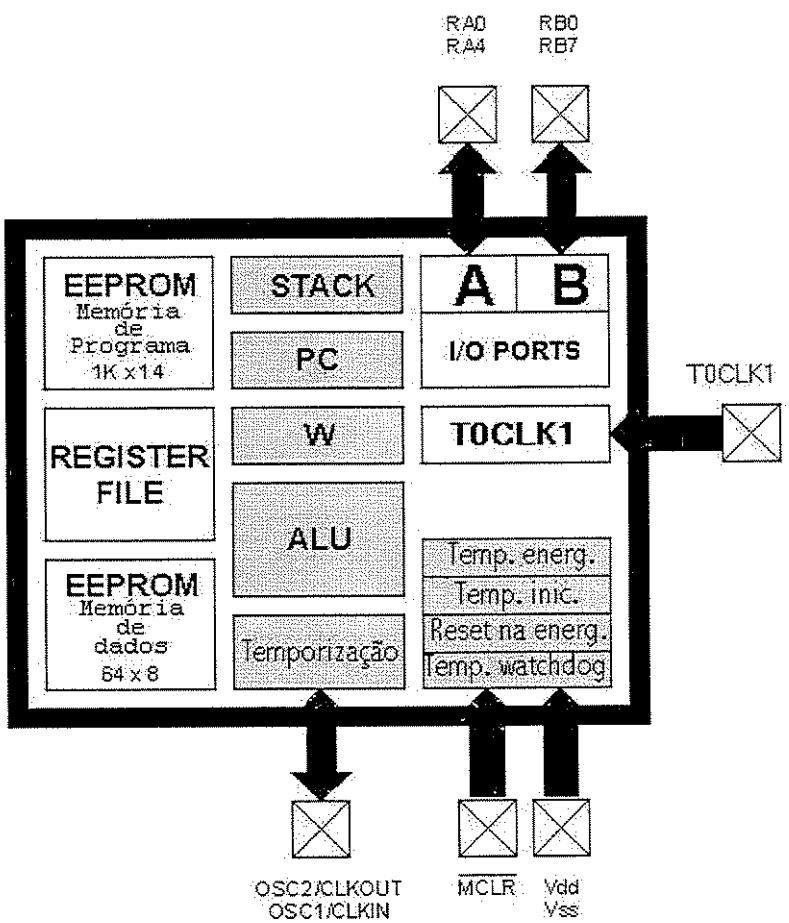



Figura 2.2: Diagrama em blocos simplificado do PIC.

Nos microcontroladores PIC, o sinal de relógio ("clock") é internamente dividido por quatro. Por tanto, para um clock externo de 4MHz, se tem um clock interno de 1MHz, e consequentemente cada ciclo de maquina dura 1 μs (Souza, 2000).

A divisão do clock por quatro forma as fases Q1, Q2, Q3 e Q4. O contador de programa é incrementado automaticamente na fase Q1 do ciclo da maquina, então a instrução seguinte é buscada da memória de programa e armazenada no registrador de instruções no ciclo Q4. Ela é codificada e executada no próximo ciclo, no intermedio de Q1 ate Q4. Esta caraterística de buscar a informação num ciclo da maquina e executa-la no próximo é conhecida como PIPELINE, ela permite que quase todas as instruções sejam executadas em apenas um ciclo, gastando assim 1μs (para um clock de 4MHz), e tornando o sistema muito mais rápido. No caso de instruções que geram "saltos" no contador de programas (Program Counter), como chamadas de rotinas e retornos, quando são executadas o PIPELINE deve primeiramente ser limpo para depois ser carregado novamente com o endereço correto, consumindo para isso dois ciclos de máquina. O conceito de PIPELINE só é possível graças à arquitetura "Harvard".

Como será visto na parte da programação, o entendimento da duração das instruções será usado para definir os tempos de espera das múltiplas partes do programa. A figura 2.3 mostra as divisões do ciclo nas 4 fases (Q1 a Q4) e o funcionamento do PIC.

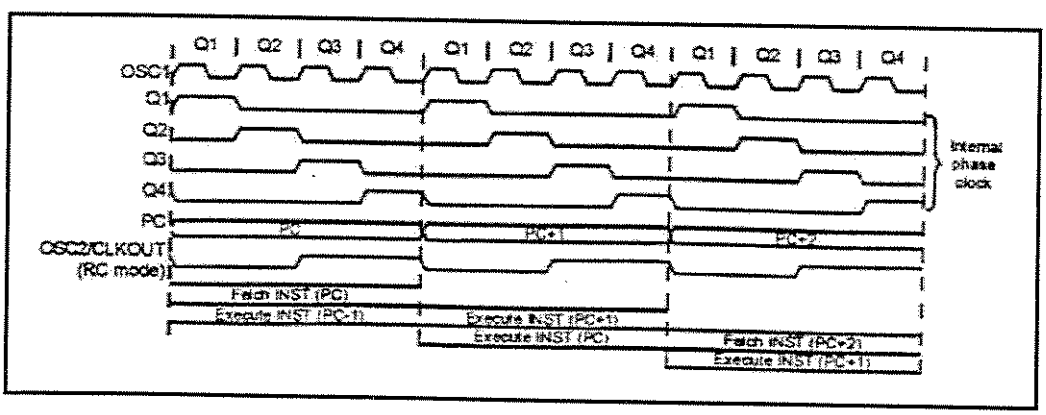



Figura 2.3: Esquema dos ciclos de máquina no PIC.

O PIC possui barramentos diferenciados para as memórias de programa e de dados. Estas memórias são completamente separadas. No caso do PIC16F84, utilizado no presente trabalho, existe uma terceira memória: a memória não volátil EEPROM.

Devido à estruturação tipo Harvard do PIC, a memória de programa pode ser de 12, 14 ou 16 bits. O tamanho desta memória também varia de modelo para modelo. Na maioria dos modelos essa memória é do tipo EPROM, que só pode ser gravada uma vez para PIC's normais, ou gravada varias vezes no caso de PIC's janelados (que podem ser apagados por meio de luz ultravioleta). Existe ainda modelos que possuem a memória de programa do tipo EEPROM, que pode ser gravada varias vezes sem necessidade de apagar a gravação anterior. Estes PIC's são muito mais fáceis de trabalhar e são usados no desenvolvimento de sistemas.

Quando o programa é desviado para o começo de uma rotina por meio de uma instrução correta, o endereço seguinte ao ponto que estava sendo rodado é armazenado na pilha (STACK, local separado da memória de programação), para que, ao fim da rotina, o programa possa retornar. O tamanho da pilha também varia de acordo com o modelo do PIC.

A memória de dados do sistema é a RAM, que é utilizada para guardar todas as variáveis e registradores utilizados pelo programa. Esta memória armazena dados de 8 bits e é volátil , ou seja que quando o PIC for desligado ela será automaticamente perdida. Ela está dividida em dois grupos: registadores especiais e registradores de uso geral (Souza, 2000).

Como já foi dito anteriormente, alguns modelos do PIC possuem ainda uma terceira memória que também pode ser utilizada pelo usuário para guardar dados. Entretanto, diferente da memória de dados, esta é uma EEPROM, isto é uma memória não volátil, que consegue manter as informações mesmo sem alimentação. Os modelos que não possuem esta memória internamente podem utilizar este recurso por intermedio de uma memória EEPROM externa interligada ao microcontrolador por I/O's, mas neste caso precisa-se também implementar rotinas para possibilitar a escrita e leitura de dados.

## 2.4.2 O PIC 16F84

Na escolha do microcontrolador procurou-se um modelo que fosse versátil, compacto e poderoso. Assim chegou-se à série PIC16F84 da MICROCHIP, por ser um dispositivo computacional extremamente adequado para experimentação e para pequenos projetos.

O PIC16F84 possui as seguintes caraterísticas básicas:

- Microcontrolador de 18 pinos, o que facilita a montagem de "hardware" experimental;
- 13 portas configuráveis como entradas ou saídas.
- 4 interrupções disponíveis (TMRO, Externa, Mudança de estado e EEPROM)
- Memória de programação EEPROM FLASH, que permite a gravação do programa diversas vezes, com até 1 000 000 de ciclos de apagamento e escrita, com retenção garantida por mais de 40 anos. Esta gravação pode ser feita sem a desmontagem do CHIP do "hardware".
- Memória EEPROM (não volátil) interna
- Via de programação com 14 bits e 35 instruções.
- Sistema de proteção de código na EEPROM. (Impossibilita que outras pessoas leiam seu código).

Uma grande vantagem da família PIC é que todos os modelos possuem um conjunto ("set") de instruções bem parecido, Uma outra vantagem é o fato de manter muitas semelhanças entre suas caraterísticas básicas, o que torna mais fácil a migração para um outro modelo da família PIC.

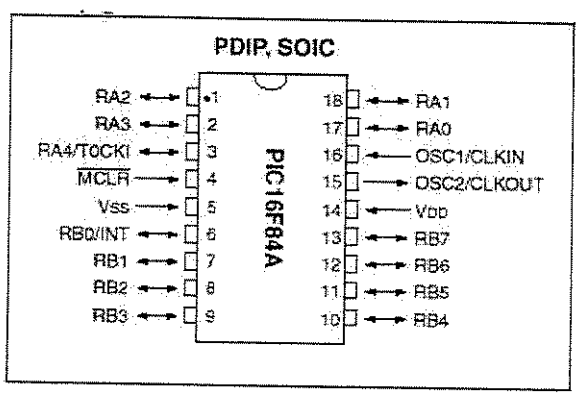



Figura 2.4: O PIC 16F84.

O PIC 16F84 possui um total de 13 I/Os separados em dois grupos denominados PORTAS. Desta forma temos a porta A (PORTA) e a porta B (PORTB). O PORTA possui 5 pinos que podem ser configurados como entradas ou saídas e seus nomes são definidos como RA0, RA1, RA2, RA3 e RA4. O pino referente ao RA4 pode ser configurado também para incrementar o contador TMR0. O PORTB possui 8 pinos configuráveis como entrada ou saída, sendo seus nomes RB0 a RB7. O RB0 pode ser utilizado também para gerar a interrupção externa , assim como os pinos RB4, RB5, RB6 e RB7 podem gerar a interrupção por mudança de estado.

Para que o microcontrolador possa funcionar é preciso ligar ao pino 5 o GND (Vss) e ao pino 14 os +5Vcc (Vdd) de tensão de alimentação nominal. O valor da tensão de alimentação depende do modelo estudado. No caso do PIC 16F84 ele vai de 2.0 a 6.0 Vcc.

O pino 4, denominado MCLR (Master Clear externo), é quem recebendo um nível baixo (GND) reseta o programa e assim paralisa o processo. Ao ser colocado em nível alto (+5Vcc) a execução do programa será retomada do ponto inicial.

O oscilador deve ser ligado aos pinos 16 e 15 respetivamente. O PIC pode operar com 4 tipos de osciladores. Estas configurações são necessárias para que o circuito opere com uma grande faixa de freqüências (32 KHz a 10 MHz), e com diferentes dispositivos osciladores.

- Osciladores a cristal de baixa potência (LP) de 32 KHz a 200 KHz.
- Osciladores a cristal/ressonadores cerâmicos convencionais (XT) de 2MHz a 4MHz.
- Osciladores a cristal/ressonadores cerâmicos de alta freqüência (HS) de 8MHz a 10 MHz.
- Osciladores a ressistor/capacitor (RC)

A escolha do tipo de oscilador depende das necessidades de nossa aplicação. Assim, se as temporizações não são críticas o Oscilador RC dá o menor custo. Um modo interessante de usar o PIC é com um oscilador externo. Embora utilize um maior número de componentes, esta opção permite um sincronismo entre vários PICs alimentados com o mesmo clock.

Para os modos LP, XT, e HS o circuito básico é sempre o mesmo, com variações nos componentes em função da freqüência de trabalho e uso de cristais ou ressonadores cerâmicos. A Tabela 2.5 apresenta os valores recomendados para os componentes e a figura 2.5 mostra o circuito básico externo do oscilador no PIC 16F84 (Microchip Technology Inc., 1998).

Tabela 2.5: Valores recomendados para os capacitores do oscilador

| Modo | Freqüência | C1 | C2 |
|---|---|---|---|
| XT | 455 KHz(ressonador) | 47-100pF | 47-100pF |
| XT | 2.0 MHz(ressonador) | 15-33pF | 15-33pF |
| XT | 4.0 MHz(ressonador) | 15-33pF | 15-33pF |
| HS | 8.0 MHz(ressonador) | 15-33pF | 15-33pF |
| HS | 10 MHz(ressonador) | 15-33pF | 15-33pF |
| LP | 32 KHz(cristal) | 68-100pF | 68-100pF |
| LP | 200 KHz(cristal) | 15-33pF | 15-33pF |
| XT | 100 KHz(cristal) | 100-150pF | 100-150p |
| XT | 2.0 MHz(cristal) | 15-33pF | 15-33pF |
| XT | 4.0 MHz(cristal) | 15-33pF | 15-33pF |
| HS | 4.0 MHz(cristal) | 15-33pF | 15-33pF |
| HS | 10 MHz(cristal) | 15-33pF | 15-33pF |

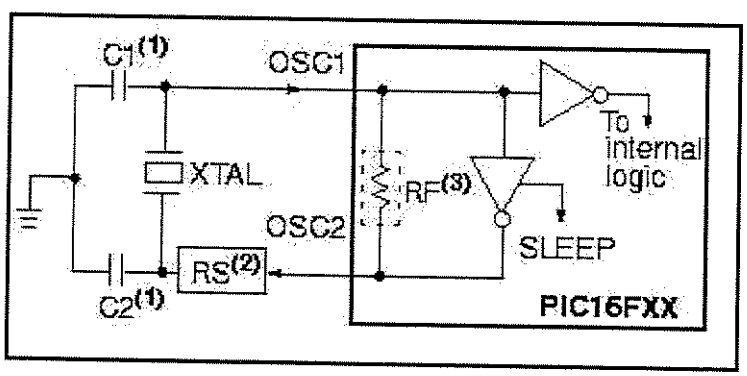



Figura 2.5: Circuito do oscilador do PIC 16F84.

Para o caso de Osciladores RC, apresentados na figura 2.6, a freqüência de operação será dependente dos valores de R e C, da fonte de alimentação, da temperatura ambiente e do circuito integrado utilizado. Este circuito não é usado senão para sistemas onde o custo e não a

estabilidade da freqüência de oscilação for o mais importante. Os valores do resistor devem de estar entre 100 KΩ e 3KΩ e o capacitor deve tomar valores entre 20 pF e 330 pF,

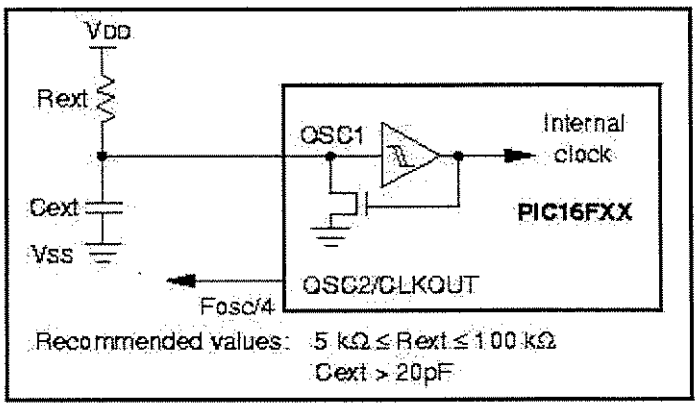



Figura 2.6: Circuito para oscilador RC do PIC 16F84.

Tabela 2.6: Caraterísticas elétricas e outras do PIC16F84

| CARATERÍSTICAS ELÉTRICAS E OUTRAS DO PIC16F84 | |
|---|---|
| **Temperatura de trabalho** | **-55°C até +125°C** |
| **Temperatura de armazenamento** | **-65°C até +150°C** |
| **Tensão de trabalho** | **2.0V a 6.0V** |
| **Voltagem máxima no pino Vdd (em relação ao Vss)** | **-0.3V até 7.5V** |
| **Voltagem máxima no pino MCLR (em relação ao Vss)** | **-0.3V até 14.0V** |
| **Voltagem máxima nos demais pinos (em relação ao Vss)** | **-0.6V até (Vdd+0.6V)** |
| **Dissipação máxima de energia** | **800 mW** |
| **Corrente máxima de saída no pino Vss** | **150 mA** |
| **Corrente máxima de entrada no pino Vdd** | **100 mA** |
| **Corrente máxima de entrada de um pino (quando em Vss)** | **25 mA** |
| **Corrente máxima de saída de um pino (quando em Vdd)** | **20 mA** |
| **Corrente máxima de entrada do PORTA** | **80 mA** |
| **Corrente máxima de saída do PORTA** | **50 mA** |
| **Corrente máxima de entrada do PORTB** | **150 mA** |
| **Corrente máxima de saída do PORTB** | **100 mA** |

Os mapas das memórias são mostrados, tanto da memória de programa, figura 2.7 como da memória de dados, figura 2.8.

### 2.4.3 Os Registradores Especiais

Tem-se falado sobre as possíveis configurações que podem ser feitas para definir as portas como entradas ou saídas, ativar as interrupções, ativar a contagem do TMRO, etc. O PIC possui uma serie de registradores especiais que são denominados SRF (Special Function Registers), os quais servem para guardar a configuração e o estado de funcionamento atual da máquina. Os SRFs são de 8 bits de comprimento e o seu endereço esta alocado na área da memória de dados. Na Tabela 2.7 são apresentados os registradores especiais.

Tabela 2.7: Endereços dos registradores do PIC16F84

| Nome | Endereço |
|---|---|
| EEADR | 09h |
| EECON1 | 88h |
| EECON2 | 89h |
| EEDATA | 08h |
| FSR | 04h/84h |
| INDF | 00h/80h |
| INTCON | 0Bh/8Bh |
| OPTION | 81h |
| PCL | 02h/82h |
| PCLATH | $0^{A}$h/8Ah |
| PORTA | 05h |
| PORTB | 06h |
| STATUS | 03h/83h |
| TMR0 | 01h |
| TRISA | 85h |
| TRISB | 86h |

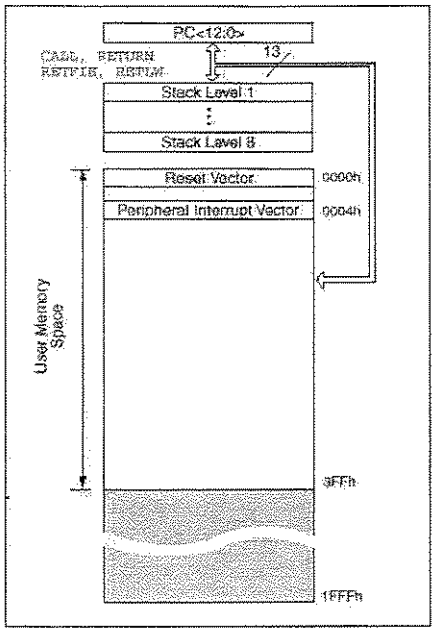



Figura 2.7: Mapa da memória de programa e STACK do PIC 16F84A.

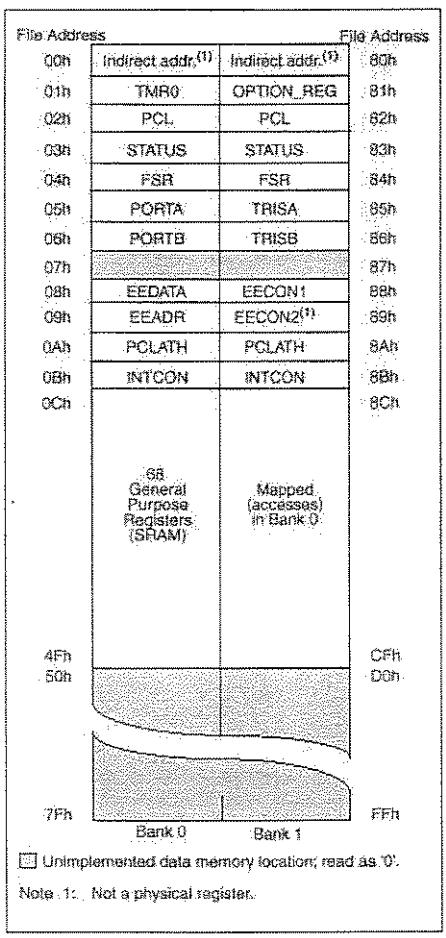



Figura 2.8: Mapa da memória de dados do PIC 16F84A.

### 2.4.4 "Set" de Instruções

Para facilitar a leitura dos programas e também a iniciação na programação dos microcontroladores PICs da MICROCHIP é conveniente entender corretamente os termos utilizados na construção dos nomes das instruções e seus argumentos.

- **Work**: Trata-se de um registrador temporário para as operações da ULA. No "assembler" ele é conhecido como **W**, e é um registro acumulador.

- **File**: Refere-se a um registrador (posição de memória) de nome de argumento **"f"**. Utiliza-se a letra **F** para representa-lo na instrução.

- **Literal**: Ele é um número qualquer que pode ser escrito, na forma: decimal (escrito no assembler como **.XX** ou **D'XX'**), hexadecimal (escrito no assembler como **0xXX** ou **H'XX'**) ou binário (escrito no assembler como **B'XXXXXXXX'**), de nome **"k"**. Utiliza-se a letra **L** para representa-lo na instrução.

- **Destino**: Este é o local onde deve ser armazenado o resultado da operação, este pode ser **F** ou **W**.

- **Bit**: Refere-se a um bit especifico dentro de um byte, o nome de argumento dele é **"b"**. Utiliza-se a letra **B** para representa-lo na instrução.

- **Teste**: Quando se requer testar o estado de um bit, quer dizer descobrir se ele e ZERO ou UM. Utiliza-se a letra **T** para representa-lo na instrução.

- **Skip**: É utilizada para criar desvios na programação, pulando ou não a próxima linha dada uma condição. Utiliza-se a letra **S** para representa-lo na instrução.

- **Set**: Refere-se ao ato de setar um bit, isto é torna-lo equivalente a UM. Utiliza-se a letra **S** para representa-lo na instrução.

- **Clear**: Refere-se ao ato de limpar um bit, isto é torna-lo equivalente a ZERO. Utiliza-se a letra **C** para representa-lo na instrução.

- **Zero**: Algumas instruções podem gerar desvios se o resultado da operação efetuada for ZERO. Utiliza-se a letra **Z** na instrução para indicar tal condição.

Existem vários outros termos, mas eles são específicos das ações realizadas pelas instruções, e são em parte auto explicados: ADD é soma, AND é "E" lógica, etc.

Como foi dito neste capítulo, o PIC 16F84 possui 35 instruções. Elas podem ser organizadas em quatro grupos conforme as suas aplicações: Operações com registradores, Tabela 2.8, Operações com literais, Tabela 2.9, Operações com bits, Tabela 2.10, Controles, Tabela 2.11.

Tabela 2.8: Resumo das Instruções do PIC, Operações com registradores.

| Instrução | Arg. | Descrição | Bit do STATUS afetado |
|---|---|---|---|
| ADDWF | f,d | Soma W e f, guardando o resultado em d. | C, DC, Z, |
| ANDWF | f,d | Lógica "E" entre W e f guardando o resultado em d | Z |
| CRLF | f | Limpa f | Z |
| COMF | f,d | Calcula o complemento de f, guardando-o em d | Z |
| DECF | f,d | Decrementa f, guardando o resultado em d. | Z |
| DECFSZ | f,d | Decrementa f, guardando o resultado em d e pula a próxima linha se o resultado for zero. | Nenhum |
| INCF | f,d | Incrementa f, guardando o resultado em d. | Z |
| INCFSZ | f,d | Incrementa f, guardando o resultado em d e pula a próxima linha se o resultado for zero. | Nenhum |
| IORWF | f,d | Lógica "OU" entre W e f guardando o resultado em d | Z |
| MOVF | f,d | Move f para d (copia) | Nenhum |
| MOVWF | f | Move W para f (copia) | Nenhum |
| RLF | f,d | Rotaciona f em 1 bit para a esquerda. | C |
| RRF | f,d | Rotaciona f em 1 bit para a direita. | C |
| SUBWF | f,d | Subtrai W de f (f-W), guardando o resultado em d. | Nenhum |
| SWAPF | f,d | Executa uma inversão entre as partes alta e baixa de f, guardando o resultado em d. | Z |
| XORWF | f,d | Lógica "OU exclusivo" entre W e f guardando o resultado em d. | Z |

Tabela 2.9: Resumo das Instruções do PIC, Operações com literais.

| Instrução | Arg. | Descrição | Bit do STATUS afetado |
|---|---|---|---|
| ADDLW | k | Soma k e W, guardando o resultado em W. | C, DC, Z |
| ANDLW | k | Lógica "E" entre k e W guardando o resultado em W | Z |
| IORLW | k | Lógica "OU" entre k e W guardando o resultado em W. | Z |
| MOVLW | k | Move k para W. | Nenhum |
| SUBLW | k | Subtrai W de k (k-W), guardando o resultado em W. | C, DC, Z |
| XORLW | k | Lógica "OU exclusivo" entre k e W guardando o resultado em W. | Z |

Tabela 2.10: Resumo das Instruções do PIC, Operações com bits.

| Instrução | Arg. | Descrição | Bit do STATUS afetado |
|---|---|---|---|
| BCF | f,b | Impõe ZERO ao bit b do registrador f. | Nenhum |
| BSF | f,b | Impõe UM ao bit b do registrador f. | Nenhum |
| BTFSC | f,b | Testa o bit b do registrador f, e pula a próxima linha se ele for ZERO. | Nenhum |
| BTFSS | f,b | Testa o bit b do registrador f, e pula a próxima linha se ele for UM. | Nenhum |

Tabela 2.11: Resumo das Instruções do PIC,Controles.

| Instrução | Arg. | Descrição | Bit do STATUS afetado |
|---|---|---|---|
| CLRW | - | Limpa W. | Z |
| NOP | - | Gasta um ciclo de máquina sem fazer nada. | Nenhum |
| CALL | R | Executa a rotina R. | |
| CLRWDT | - | Limpa o registrador WDT para não acontecer o reset. | /TO,/PD |
| GOTO | R | Desvia para o ponto R, mudando o PC | Nenhum |
| RETFIE | - | Retorna de uma instrução. | Nenhum |
| RETLW | K | Retorna de uma rotina com k em W. | Nenhum |
| RETURN | - | Retorna de uma rotina, sem afetar W. | Nenhum |
| SLEEP | - | Coloca o PIC em modo sleep (dormido) para economia de energia. | /TO,/PD |

### 2.4.5 Introdução à Ferramenta MPLAB

O MPLAB é um programa para computador que roda sobre a plataforma Windows, e é usado como ambiente de desenvolvimento de programas para microcontroladores PIC da MICROCHIP. É uma ferramenta muito poderosa. Ele junta no mesmo ambiente o gerenciamento de projetos, compilação, simulação, emulação e gravação do CHIP. O manual para esta ferramenta de desenvolvimento está no site da MICROCHIP. Mesmo assim, é apresentada esta ferramenta devido à grande utilidade que ela teve no projeto.

Ao iniciar o MPLAB se tem acesso ao ambiente de trabalho global. Trata-se de uma área para abertura das janelas de trabalho, com um menu superior. Abaixo dele se têm uma barra de ferramentas, com diversos icones relativos a funções especificas. O primeiro ícone do lado esquerdo muda as barras de ferramentas disponíveis. Na parte inferior está a barra de status (figura 2.9).

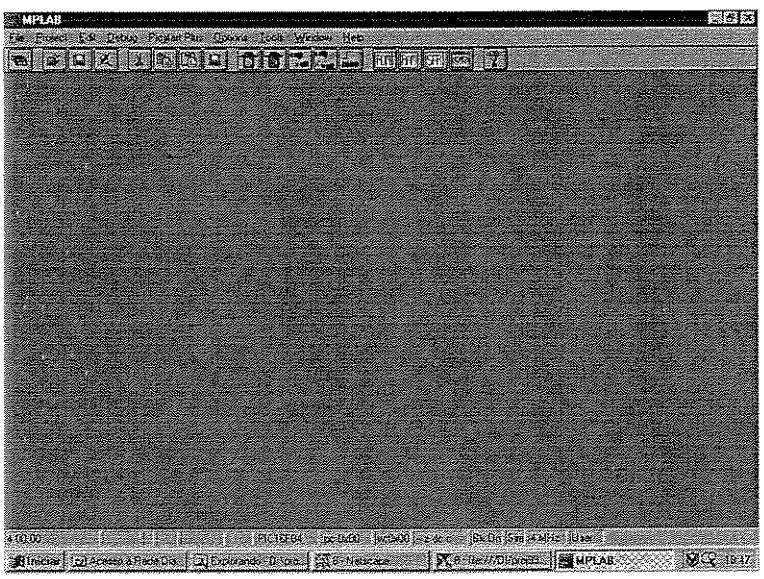



Figura 2.9: Tela inicial do ambiente de trabalho MPLAB.

Para poder trabalhar neste ambiente não basta o arquivo de código fonte. É necessário ter outras informações para que o sistema possa ser compilado e executado. O MPLAB usa para isto o conceito de "projeto". Define-se o conceito de projeto como o arquivo que guarda todas as informações necessárias ao sistema em desenvolvimento. Para que o MPLAB possa funcionar corretamente tem que ser aberto o projeto correspondente.

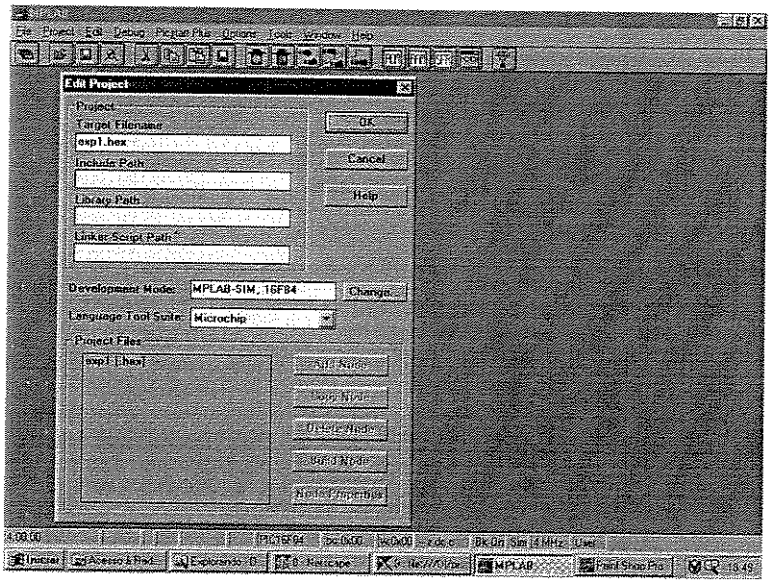



Figura 2.10: Tela de composição do projeto no MPLAB.

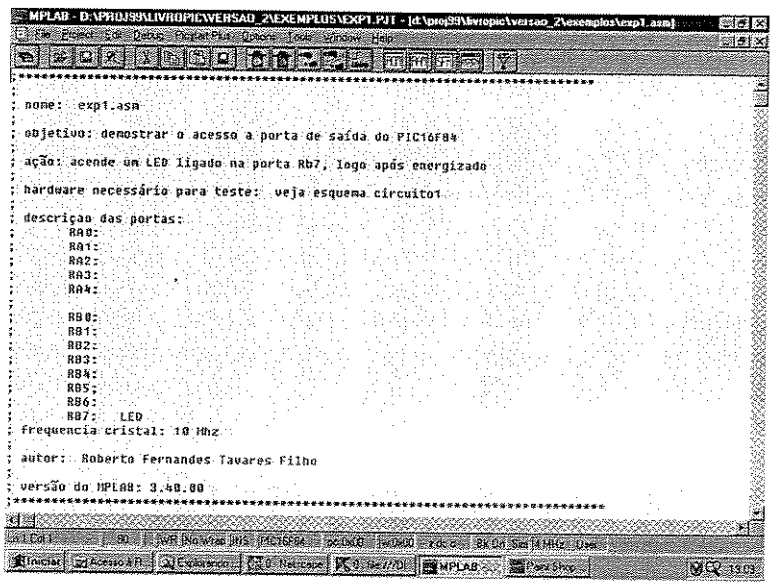



Figura 2.11: Ambiente de trabalho completo do MPLAB.

### 2.4.6 Programador *Sb Microlab*

O módulo de desenvolvimento *Sb Microlab* visa permitir ao experimentador ou integrador de sistemas, a construção de um ambiente para programação e experimentação com o microcontrolador PIC16F84. Com este módulo, um computador pessoal (PC) e o "software"

adequado, o projetista estará equipado para o desenvolvimento de circuitos baseados neste microcontrolador.

O desenvolvimento de uma aplicação baseada em microcontroladores envolve duas fases: Projeto do "hardware" do sistema e Projeto do "software" do sistema. Na implementação do "software" é usado, como já foi dito, o "software" MPLAB, disponível gratuitamente no "site" do fabricante do PIC, em http://www.microchip.com

Para a gravação do programa no PIC alvo e teste do sistema (parte do Projeto do "hardware" do sistema) o *Sb Microlab* é a ferramenta ideal, pois ele permite a monitoração de saídas e possui led's para o teste do programa do PIC16F84 alvo.

O sistema se interliga a um PC através de um cabo serial com conector DB9 fêmea. O protocolo de "hardware" da RS232-C é completado na placa de circuito impresso.

O *Sb Microlab* opera conjuntamente com o "software" *Sbprog40* que deve ser executado sob o sistema operacional Windows95 ou superior, previamente instalado no computador. O *Sbprog40* transferirá um arquivo **.hex** de 8bits, padrão de saída dos programas assembler, para a memória de programas do PIC16F84. Este arquivo **.hex** é gerado pelo assembler do MPLAB ou outro assembler semelhante. No Anexo A.2 se têm o diagrama eletrônico do módulo e na figura 2.12 se mostra a disposição dos componentes

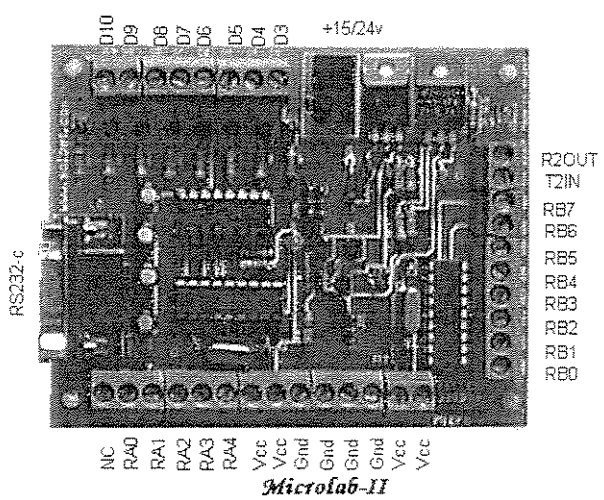



Figura 2.12: Placa do módulo *Sb Microlab* para a programação do PIC16F84.

O *SbProg40* é um "software" simples que permite gravar na memória de programas do PIC16F84 um programa qualquer, desde que este já esteja compilado num arquivo **.hex.** Na figura 2.13 é mostrada a janela de iniciação do *SbProg40.*

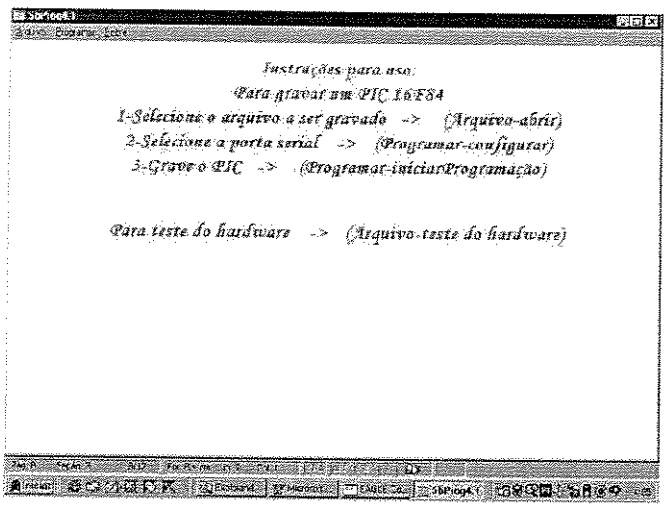



Figura 2.13: Tela de apresentação do *Sb Prog40.*

## 2.5 Controle em Tempo Real

Até agora foram apresentados os fundamentos do "software" do microcontrolador, e também, como eles são "suportados" pelo "hardware". A seguir será mostrado como devem ser unidos o "software" e o "hardware", construindo blocos com determinadas estratégias para se obter o controle em tempo real no robô.

Existem três estratégias para escrever um "software" que possa responder a estímulos externos em tempo real. ***"Polling"*** (sondagem) é o método onde "loops" do "software" continuamente verificam os pinos de entrada, mantendo ocupado o microcontrolador, mesmo que nenhum evento externo esteja acontecendo. O seguinte método, o ***"Interrupt driven"***, é mais eficiente. Neste método o evento externo cria um sinal que direciona o microcontrolador a pospor qualquer tarefa que esteja realizando e responder ao evento/estimulo imediatamente. A terceira estratégia, ***"Input Capture"***, pode ser usada se o microcontrolador possui um "hardware"

especial, conhecido como registrador de captura de entrada. O processador nunca é interrompido, (Jones, 1993).

Estos conceitos serão explicados a seguir com ajuda de alguns exemplos.

- ***Polling*:** Requer monitoramento de uma entrada e determina uma ação imediatamente após o evento/estimulo de interesse ter acontecido. Por exemplo, medir o tempo de propagação de um pulso emitido por um sonar. Esta tarefa é importante para medir a diferença entre o tempo em que o pulso foi enviado e seu tempo de retorno. Na implementação de uma subrotina assume-se que a saída do receptor de ultra-som está conectada a uma porta (PORTA0), a qual passa ao nível ALTO quando um eco é detectado. Na inicialização do pulso (som) do sonar um registro determinado é zerado para começar a contagem do tempo de espera. Será este registro o encarregado de armazenar o valor dos ciclos gastos na espera do eco do sonar. Esta quantidade de ciclos será usado, por exemplo, na determinação da distancia a um obstáculo. Para que o programa seja o suficientemente robusto, na subrotina tem que ser considerada uma saída eventual do "loop" (tempo limite de espera pelo eco), no caso do eco não voltar.

- ***Interrupts*:** Na estratégia anterior o microcontrolador fica esperando por um evento acontecer. Nesta estratégia o microcontrolador toma uma determinada ação (executa instruções) quando um evento/estimulo atual ocorre (isto é chamado de interrupção). Uma interrupção é um evento/estimulo que provoca automaticamente uma resposta no microcontrolador. O código que responde ao evento é chamado de "rotina de interrupção". Após a execução da rotina de interrupção o microcontrolador volta à linha onde parou antes da interrupção e continua com a execução do programa principal. O microcontrolador pode fazer isso graças ao fato de ter armazenado o endereço da linha em execução do programa no STACK.

- ***Input Capture:*** O trabalho que foi necessário nas duas estratégias anteriores é desnecessário quando temos microcontroladores que facilitem a captura de entradas, e assim o cálculo do tempo de retorno do pulso do sonar, sem precisar de rotinas de interrupção. O próprio "hardware" pode capturar o tempo quando o PORTA0 adquire um nível ALTO.

## 2.6 Módulo *Sb Step Control*

O módulo ***Sb Step Control*** permite o controle remoto, via interface serial, de até dois motores de passo do tipo unipolar, com cinco ou seis fios. Cada bobina pode ser alimentada com até 12 V, 500 mA por enrolamento e permite controlar a velocidade e a direção de giro dos dois motores de passo através de um computador.

A alimentação do módulo deve de ser realizada com uma fonte de 12Vcc, com capacidade de 1 Ampère. O protocolo de comunicação utiliza apenas caracteres ASCII, de modo que podem ser realizados testes utilizando apenas um emulador de terminais, como o hiperterminal do Windows, ou qualquer outro similar.

O microcontrolador utilizado é o PIC16F84, de modo que o firmware pode ser atualizado sem necessidade da compra de outro microcontrolador.

### 2.6.1 Caraterísticas do controlador

Este módulo possui as seguintes características básicas:

1. Consegue controlar dois motores de passo de quatro bobinas cada um, a uma tensão máxima de 12V e uma corrente máxima de 500 mA por bobina.
2. Alimentação de 12 V, 1 A.
3. Comunicação serial RS232-C, 1200, 8, N, 1 (1200 bauds, 8 bits de dados, sem paridade e 1 *Stop Bit*).
4. Protocolo de comunicação com comandos para controle individual dos motores. Teste de conexão.
5. Possui um microcontrolador PIC16F84

Na Tabela 2.12 são mostrados os comandos usados no protocolo de comunicação para o controle dos motores de passo. Estes são reconhecidos após o recebimento de um caracter <CR>

Tabela 2.12: Comandos usados pelo módulo *Sb Step Control* para o controle dos motores.

| Comando | Parâmetro | Ação |
|---|---|---|
| A | NNN qualquer valor entre 1 e 255* | Estabelece a duração do passo do motor 0 |
| B | NNN qualquer valor entre 1 e 255* | Estabelece a duração do passo do motor 1 |
| C | Nenhum | Liga ambos motores |
| D | Nenhum | Desliga ambos motores |
| E | Nenhum | Motor ZERO no sentido horário |
| F | Nenhum | Motor ZERO no sentido antihorário (default) |
| G | Nenhum | Motor UM no sentido horário |
| H | Nenhum | Motor UM no sentido antihorário (default) |
| I | Nenhum | Solicita o envio do pronpt para o PC controle |
| J | Nenhum | Liga motor ZERO |
| K | Nenhum | Desliga motor ZERO |
| L | Nenhum | Liga motor UM |
| M | Nenhum | Desliga motor UM |

* Cada unidade do parâmetro eqüivale a 4 milisegundos de duração.

### 2.6.2 Descrição do controlador *Sb Step Control*

No esquema do circuito eletrônico do *Sb Step Control*, (Anexo A.3) pode-se observar três componentes principais. Estes são os circuitos integrados MAX232CPE (Maxim Integrated Products, 2000), ULN2803A (SGS Thomson Microelectronics, 1997)e o PIC16F84 (Microchip Technology Inc., 1998).

Para que o módulo possa se comunicar com outros módulos ou PCs possui uma interface do tipo RS232-C. utilizando o integrado MAX232CPE.

O ULN2803A é um CI com oito transistores do tipo Darlington, saída de corrente de até 500 mA funcionando como driver dos motores de passo.

O 78L05 é um regulador da tensão, e é o encarregado de fornecer os 5V necessários para a alimentação dos CI (National Semiconductor Corporation, 2000).

No ***Sb Step Control*** existem um total de cinco conectores. O conector JP2 para a alimentação +12V e GND. JP3 e JP4 são os conectores dos motores de passo com quatro bobinas do tipo unipolar com cinco ou seis fios. O conector JP1 esta interligado à PORTA do PIC bits 4, 3 e 2 (estas portas não são utilizadas e estão livres para o uso), e um conector tipo DB9 fêmea, para a interface. Ver a figura 2.14.

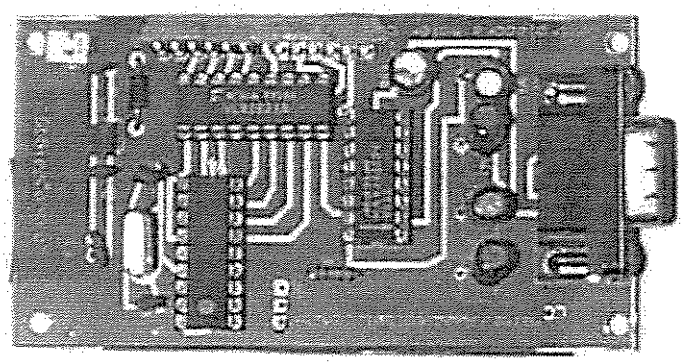


Figura 2.14: Placa do módulo ***Sb Step Control*** para controlar motores de passo.

## 2.7 Módulo *Sb 9902_2*

O módulo ***Sb 9902_2*** é um "hardware" usado para a construção de equipamentos dedicados com processamento digital. Esta baseado no PIC16F84-10P, operando com um cristal de 10MHz. As portas RB0 a RB7 e RA2 a RA5 estão disponíveis. Este módulo incorpora uma interface RS232-C padrão, com um conector DB9. A alimentação pode ser realizada com qualquer tensão continua entre 8 e 25V, ocorrendo a estabilização na própria placa. A programação poderia ser realizada na própria placa, sem necessidade da retirada do circuito.

Adicionalmente o módulo ***Sb 9902_2*** permite construir um sistema microprocessado baseado no PIC16F84 e um conversor analógico-digital (A/D). Também pode se implementar uma interface RS232-C, com um protocolo de comunicação que utilize apenas caracteres ASCII, de modo a permitir a ligação a computadores ou display´s inteligentes. Um exemplo de utilização do módulo esta na aquisição remota de dados.

### *2.7.1* Caraterísticas do Módulo *Sb 9902_2*

Este módulo possui as seguintes características básicas:

1. CPU PIC16F84 operando com um cristal de 10MHz.
2. 9 bits de I/O digital.
3. Entrada analógica 0-5v, oito bits.
4. Comunicação serial RS232-C.
5. Alimentação com tensões entre 8-25V.
6. Portas do PIC livres para o uso respectivo.

### 2.7.2 Descrição do Módulo *Sb 9902_2*

No esquema do circuito eletrônico do módulo ***Sb 9902_2***, (Anexo A.4) pode-se observar três componentes principais, estes são os circuitos integrados: MAX232CPE (Maxim Integrated Products, 2000), PIC16F84 (Microchip Technology Inc., 1998) e ADC0831 (National Semiconductor Corporation, 1999).

O integrado MAX232CPE é utilizado para a comunicação com outros módulos ou PCs (interface do tipo RS232-C).

O ADC0931 é um conversor analógico-digital (A/D) com I/Os em serie, desenhado para permitir uma fácil interface com alguns microcontroladores.

Também existe na placa um outro componente, o 78L05, que é um regulador de voltagem. Ele é o encarregado de fornecer os 5V necessários para a alimentação dos CI.

No ***Sb9902-2*** existe um total de quatro conectores: Um conector tipo DB9 fêmea, para a interface serial. Se for necessário utilizar um conector de 25 pinos, e possível utilizar um "plug" conversor (ou adaptador). Consultar no Anexo A.5 o *SB Fast Card.*

O conector JP1 serve para a gravação "*in circuit*" do microcontrolador PIC16F84. Isto quer dizer que o PIC, como já foi dito anteriormente, pode ser gravado no mesmo módulo, sem precisar ser retirado da placa. Na Tabela 2.13 é mostrada a configuração deste conector.

Tabela 2.13: Pinos do conector JP1 para a gravação "*in circuit*" do módulo ***Sb Step Control.***

| Pino do conector | Função |
|---|---|
| JP1-1 | Vcc |
| JP1-2 | RB7 |
| JP1-3 | RB6 |
| JP1-4 | /MCLR |
| JP1-5 | GND |

O conector JP2 serve para isolar a fonte de alimentação durante o procedimento de gravação "*in circuit*", na mesma placa do módulo.

O conector JP3 serve para o interfaceamento do módulo ***Sb 9902-2*** com os diversos dispositivos a ser utilizados, Tabela 2.14.

Tabela 2.14: Pinos do conector JP3 para o interfaceamento do módulo ***Sb Step Control.***

| Pino do conector | Função |
|---|---|
| JP3-1 | Vcc (8-25V) |
| JP3-3 : JP3-5 | Entrada analógica |
| JP3-7 | Livre |
| JP1-9 | Livre |
| JP3-11 : JP3-25 | RB7 : RB0 |
| JP3-27 : JP3-33 | RA4 : RA2 |
| JP3-2 : JP3-34 | GND |

Na figura 2.15 se mostra a placa do circuito impresso do módulo ***Sb 9902-2***.

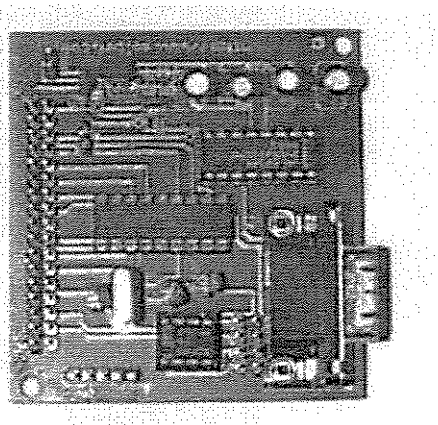


Figura 2.15: Placa do módulo ***Sb 9902-2,*** usado como "cérebro" do robô.

## 2.8 Implementação da Interface RS232-C

Uma das funções de grande importância no projeto de sistemas de microcontroladores se refere à comunicação. Um dos sistemas de comunicação mais utilizados é a comunicação serial padrão RS232-C. Embora o PIC16F84 não possua a função de comunicação serial implementada por "hardware", a implementação baseada em "software" é relativamente simples.

A interface RS232-C possui níveis de tensão diferentes dos que freqüentemente são usados em lógica digital. O nível lógico ZERO é representado por um sinal de 12V, e o nível lógico UM por um sinal de –12V. Para realizar esta conversão de níveis a maneira mais pratica é o uso de um integrado especifico. O CI usado nos dois módulos é o MAX232CPE (Maxim Integrated Products, 2000).

A RS232-C é uma interface assíncrona, ou seja, o dado é transmitido sem um sinal de sincronismo associado a cada bit. O sincronismo é obtido a partir da detecção de um bit inicial, chamado de *Start Bit.* A partir do *Start Bit* seguem-se de 5 a 8 bits de dados, um bit opcional de paridade e um *Stop Bit* que na realidade poderia ocupar o espaço de 1, 1.5, ou 2 bits.

Um ponto importante na especificação de um enlace serial é a velocidade de transferência de dados. A velocidade de uma interface serial é dada normalmente em bits por segundo, ou bauds. As velocidades padronizadas são de 300, 600, 1200, 2400, 4800, 9600 e 19200 bauds. Por isso na literatura normalmente se terá especificações do tipo 2400, N, 8, 1 o que significa 2400

bauds (bits por segundo), sem paridade, oito bits de dados e 1 *Stop Bit*. No código do programa que implementa tanto o Controlador dos Motores ***Sb Step Control*** como o Módulo ***Sb 9902_2*** opera-se com as seguintes especificações: 1200, N, 8, 1, permitindo que os dois módulos se comuniquem entre si.

Com estas especificações um **byte** de dados (8 bits) transmitido, é "envolvido" por um bit inicial em 0 (ZERO *Start Bit*), e um bit final em 1 (UM *Stop Bit*). Se não houverem bytes adicionais a transmitir a linha deve ficar em nível lógico UM. É importante ter presente que na interface RS232-C, o nível lógico UM corresponde à -12V. O byte é transmitido com o **bit 0** (bit menos significativo do byte) em primeiro lugar.

A duração de um bit depende da velocidade com a qual esteja-se transmitindo, conforme se mostra na Tabela 2.15. Este tempo de duração é importante na implementação da subrotina encarregada da transmissão de dados, nos programas principais dos PICs. Se deve procurar amostrar o bit bem no meio do intervalo, tendo assim segurança de estar amostrando um bit corretamente.

Tabela 2.15: Tempo de duração de um bit em relação as velocidades da interface RS232-C

| Velocidade | Duração |
|---|---|
| 300 | 3.33 ms. |
| 600 | 1.67 ms. |
| 1200 | 833 µs. |
| 2400 | 417 µs. |
| 4800 | 208 µs. |
| 9600 | 104 µs. |
| 19200 | 52.1 µs. |

Gerar um protocolo RS232-C é muito simples. Se gera em primeiro lugar um *Start Bit*, para isto se faz um bit de uma porta, por exemplo o PORTA0, igual a ZERO. Se espera por um tempo equivalente ao tempo de duração de um bit (dado pela tabela anterior). A partir deste instante se envia o byte desejado, bit a bit, lembrando sempre de enviar o menos significativo em primeiro

lugar, esperando sempre que cada bit enviado tenha a duração adequada. Por ultimo enviamos um bit com UM.

No programa implementado no **RAM-I** se gera o intervalo de um bit (duração do tempo de um bit) através de um "loop" de "software". Ver no Anexo A.1 os programas implementados. Nesta contagem do tempo, é também importante ter presente os ciclos gastos pelos comandos do PIC, como já foi visto no inicio deste capitulo, na parte da apresentação do PIC.

## 2.9 Os Sensores

Uma das principais características dos robôs móveis considerados inteligentes é o uso de sensores externos que os tornam capazes de operar em ambientes desestruturados. Tais sensores permitem ao robô autônomo extrair informações do ambiente levando-o a reagir às mudanças do mesmo de forma inteligente (Rezende, 1992).

A entrada de informações em um sistema inteligente ocorre sempre por meio de sensores. A informação dos sensores é sempre enviada ao módulo de processamento sensorial do sistema inteligente, que transformará os sinais oriundos dos sensores em informação útil ao sistema (Suárez, 2000).

Sistemas com pouca ou nenhuma capacidade sensorial externa, geralmente são limitados a operações de sequência fixa em ambientes altamente estruturados e não podem fornecer nenhum grau de autonomia substancial ou adaptabilidade (Rezende, 1992). Assim sendo, pode-se perceber que um robô sem sensores não é capaz de lidar devidamente com mudanças em seu ambiente.

Existem em robótica diversos modos de classificar os sensores (Ferreira, 1991), como por exemplo em sensores de percepção interna e de percepção externa ao robô. Outra classificação dos sensores para robótica móvel os divide em: sensores de luz (fotoresistores, detectores de proximidade com infravermelho), sensores de força (microchaves, resistências sensíveis à força), sensores de som (microfones, soar), sensores de posição e orientação ("encoders", giroscopios,

sensores de inclinação, bússola), sensores "internos" (nível de bateria, corrente, temperatura) (Jones, 1993).

Os sensores de percepção externa encontrados na robótica podem ser classificados da seguinte forma (GSI, 2001): sensores de toque, sensores de distâncias, sensores de proximidade e sensores de visão.

### 2.9.1 Sensores de Toque

Sensores de toque (tato) são dispositivos que indicam o contato entre eles próprios e algum outro objeto. Normalmente são usados em robôs industriais ou manipuladores. A informação de toque pode ser usada, por exemplo, para a localização e reconhecimento de objetos, indicando ainda a magnitude da força de contato entre os dois. Um dos objetivos de se usar sensores deste tipo é identificar e controlar diretamente a interação entre o robô e o ambiente onde ele se encontra.

### 2.9.2 Sensores de Distância

São dispositivos que medem a distância entre um ponto de referência (normalmente outro sensor) e os objetos no campo de atuação do mesmo. Tais sensores são usados na navegação de robôs e no desvio de obstáculos, onde a sua utilização consiste em estimar as distâncias aos objetos mais próximos, em aplicações onde a localização desses objetos é necessária. Existem várias técnicas para efetuar os cálculos para determinar essa distância: o método da triangulação, a abordagem da luz estruturada e a técnica dos ultra-sons. Os sensores de distância mais utilizados nos robôs móveis autônomos inteligentes (Rezende, 1992) são os sensores de ultra-som.

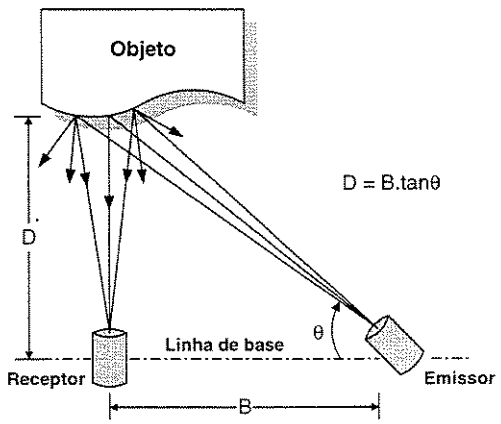



Figura 2.16: Exemplo de um sensor de distância.

### 2.9.3 Sensores de Proximidade

Os sensores de distância discutidos anteriormente estimam a distância entre o sensor e um determinado objeto. Os sensores de proximidade, por outro lado, têm geralmente uma saída binária que indica a presença de um objeto a uma distância pré-definida. Os sensores de proximidades mais utilizados nos robôs móveis autônomos inteligentes (Rezende, 1992) são sensores de infravermelho. Em algumas aplicações é suficiente determinar a presença (ou ausência) de um objeto em uma certa posição no lugar de medir a distância ao objeto (Bradshaw, 1990). Embora este tipo de sensor não devolva a distância entre ele e um objeto qualquer, eles podem identificar se algo está presente ou não, dentro do seu cone de detecção (Jones, 1993).

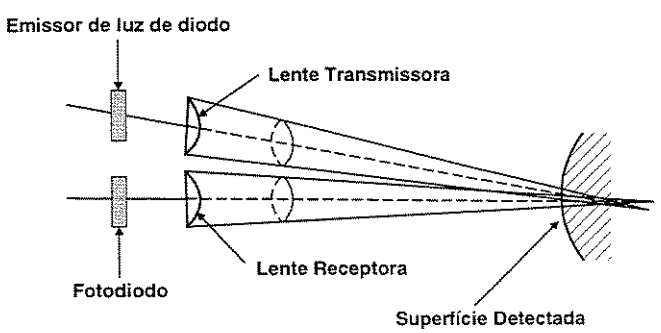



Figura 2.17: Exemplo de um sensor de proximidade.

### 2.9.4 Sensores de Visão

O uso destes sensores tem despertado o interesse de muitos pesquisadores e é atualmente uma das áreas de pesquisa mais ativas (Han, 1994) (Ohya, 1998). O avanço das pesquisas em sistemas de visão tornou possível o surgimento de robôs que interagem com o ambiente, recebendo informações de diversas formas para processá-las e, em função delas, executar determinadas tarefas. A recepção e obtenção de informações do meio é feita pelos "olhos" do robô, ou seja, por uma câmara de televisão e/ou algum outro sensor juntamente com o equipamento de digitalização e "hardware/software" necessários para o interfaceamento entre os mesmos.

Pode-se concluir que uma exigência fundamental para a solução do problema de navegação (Bradshaw, 1990) é a existência de um sistema sensorial, permitindo a aquisição dos dados necessários para uma descrição do ambiente do robô móvel em detalhes suficientes para o controle do seu movimento.

## 2.10 Motores

Um motor elétrico converte a energia elétrica em mecânica. Existem uma ampla variedade de motores para uso em robótica: motores eletromagnéticos a corrente contínua (DC), motores eletromagnéticos a corrente alternada (AC) e as variações de cada um (Jones, 1993).

Em adição aos motores electromagnéticos DC e AC, existe um outro tipo de motor que não é eletromagnético. Este é o motor ultra-sônico piezoeléctrico, que tem como uma das suas caraterísticas girar a baixa velocidade com um maior torque.

Os motores eletrostáticos são micromáquinas que trabalham com o principio de atração de cargas. Em pequenas escalas forças eletrostáticas podem ser relativamente fortes.

SMA ("Shape Memory Alloy"), ligas com memória de forma, também são usadas como atuadores para robótica.

Os motores DC são usados na robótica móvel porque a fonte de energia neles é tipicamente uma bateria DC. Dentro deste tipo de motores se encontram os motores de passo e os "servo motores". Este termo é usado numa variedade de contextos. O contexto empregado neste trabalho é o de um servo motor DC (motor de precisão, no qual se pode controlar a sua velocidade e/ou posição, usualmente caraterizado por possuir três conexões).

O motor de passo é um dispositivo usado para converter pulsos elétricos num movimento mecânico rotacional discreto, (Thomson Airpax, 1995).

Em linhas gerais o motor de passo é um atuador do tipo digital. Na operação mais típica, a cada pulso recebido da unidade de controle, as correntes em suas fases são chaveadas e o rotor do motor avança ou recua um passo. Eles exibem três estágios: parados, ativados com o rotor travado (bobinas energizadas) ou girando em etapas.

No que se refere ao seu funcionamento, os motores de passo podem ser comparados aos síncronos, ou seja, um campo rotativo (neste caso gerado pela eletrônica de controle) faz girar um rotor magnético (figura 2.18). Tais motores foram subdivididos de acordo com a forma em que é gerado o campo rotativo (enrolamento bipolar ou unipolar no estator) e do material empregado na confecção do rotor. Os mais usados são os unipolares, geralmente de quatro bobinas. Neles, cada fase consiste de um enrolamento com derivação central, ou mesmos de dois enrolamentos separados, de forma que o campo magnético possa ser invertido sem a necessidade de se inverter o sentido da corrente.

Os motores de passo possuem algumas caraterísticas próprias que os diferencia dos motores comuns (Braga, 1999):

- Os motores de passo funcionam como dispositivos posicionadores, pois podem parar numa posição perfeitamente controlada.
- Os motores de passo também funcionam como motores de velocidade perfeitamente controlada, sendo energizados numa determinada ordem.

- Eles podem ter o seu eixo posicionado em um ângulo proporcional ao número de impulsos na entrada.
- Os erros que ocorrem no posicionamento de seu eixo são muito pequenos e não são cumulativos. Um motor de passo pode ser posicionado com a precisão de 1 milésimo de radiano.
- O controle sem realimentação ("open loop") é possível devido ao uso de sinais digitais para esta finalidade.
- As respostas à partida, à parada e à reversão são muito rápidas.

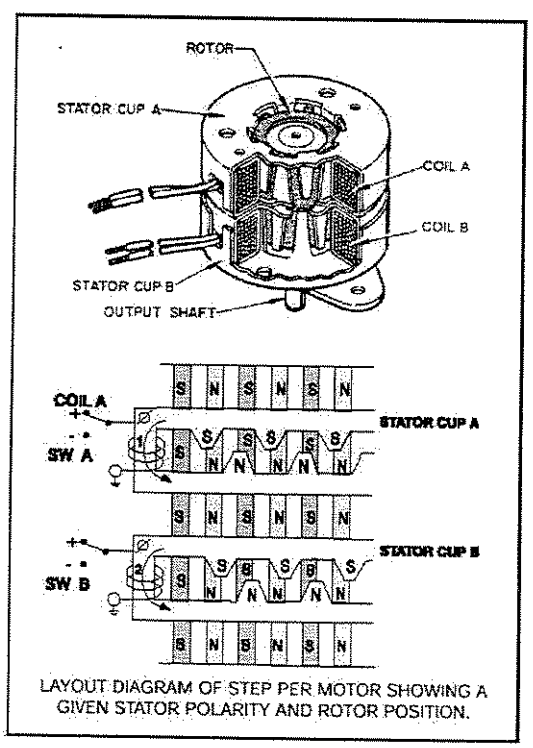



Figura 2.18: Motor de passo magnético permanente.

O ângulo do passo do motor é variável (desde 0.72° até 90°), alguns ângulos de passo padronizados são: 3.6° (100 ppv), 7.5° (48 ppv), 15° (24 ppv), 18° (20 ppv).

O torque produzido por um motor de passo específico depende de vários fatores: duração do passo, corrente aplicada ao enrolamento e o projeto do "drive".

A curva caraterística Torque-Velocidade do motor de passo é a chave para a seleção correta do tipo de motor e o melhor método de controle. Na figura 2.19 se tem a curva caraterística do motor de passo Thomson Airpax 57L048B L/R Stepper.

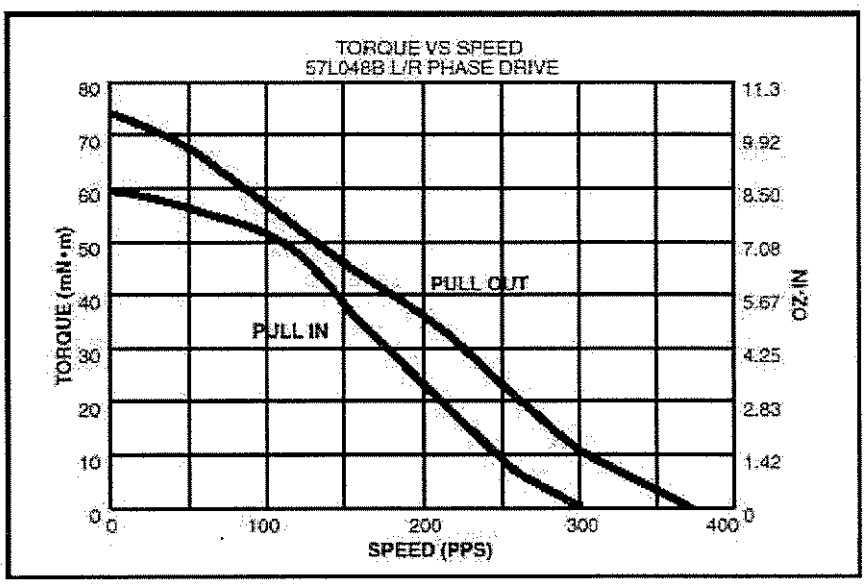



Figura 2.19: Curva caraterística Torque/Velocidade do motor de passo 57L048B L/R.

Os motores de passo são formados por quatro bobinas que devem ser excitadas com uma certa ordem, de acordo com o posicionamento desejado. O modo de acionamento normal é uma seqüência de quatro passos mostrada na figura 2.20.

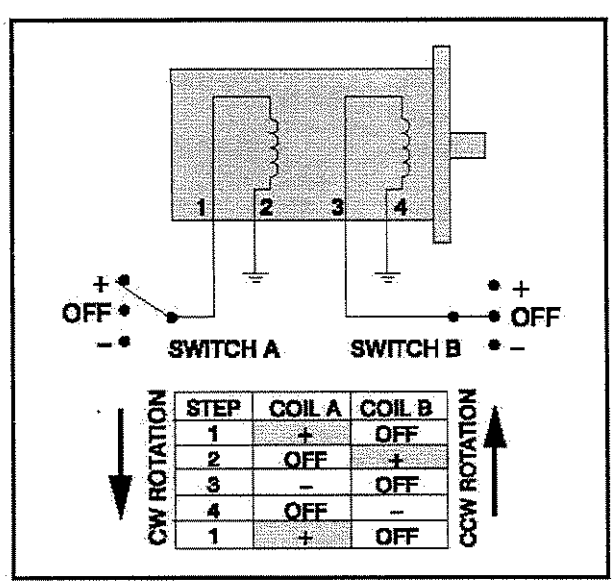



Figura 2.20: Esquema da seqüência de ligação de 4 passos.

Na figura 2.21 se mostra os modos de ligação unipolar e bipolar e o esquema da seqüência de ligação dos motores de passo unipolares e bipolares.

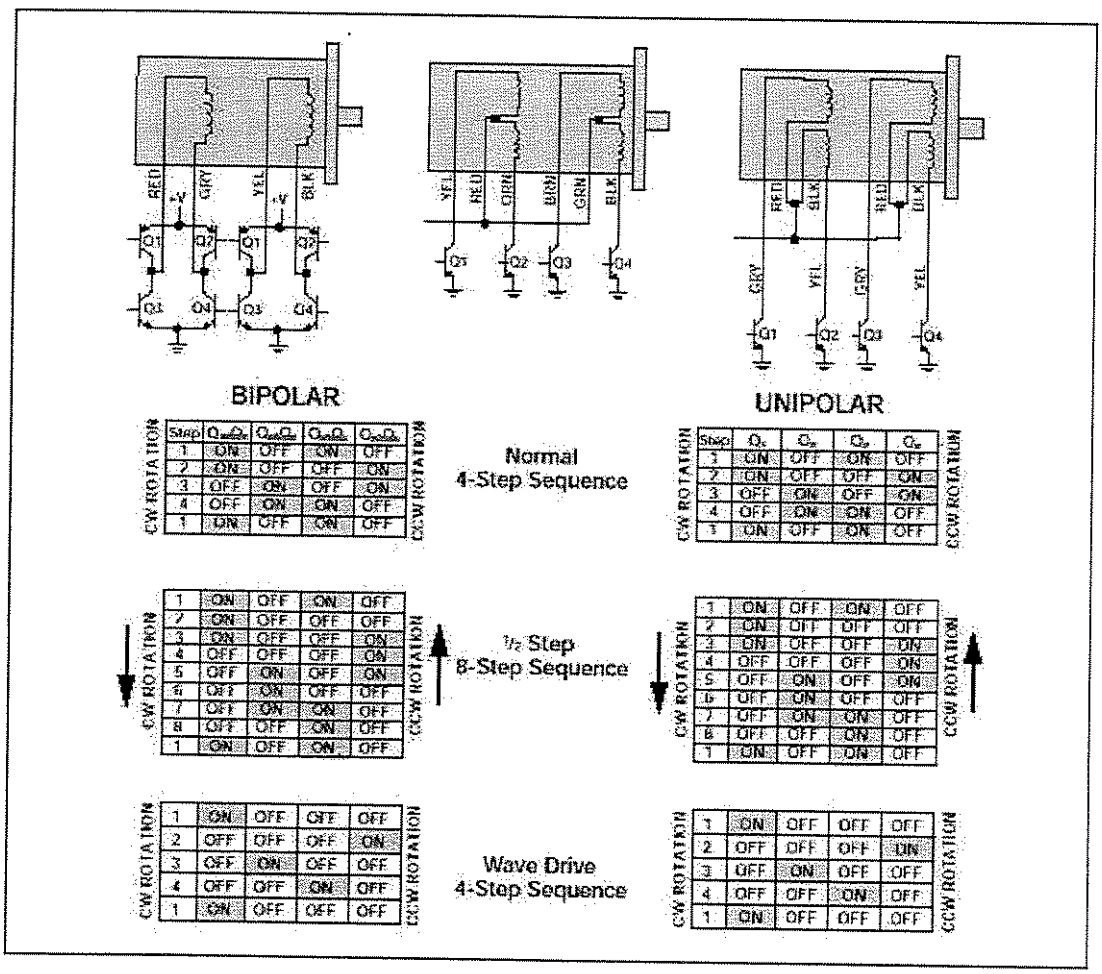



Figura 2.21: Esquema da seqüência de ligação de motores de passo unipolares e bipolares.

No robô **RAM-I** foi utilizado o motor de passo do fabricante TAMAGAWA SEIKI CO LTD. De 12V, $0.16^{A}/\phi$ (Amperios por Fase), 75Ω, 100 S/R (ppv, passos por volta) da série TS3012N5005. Estes motores são correntemente encontrados em periféricos de computadores, se caraterizam por ter uma ampla versatilidade, e a possibilidade de aplicação de controle em malha aberta. A velocidade e posição são diretamente controladas pela freqüência e número de pulsos enviados. Tais características, aliadas à disponibilidade de alguns destes motores, determinou a utilização dos mesmos no robô. No Anexo A.6 se tem os dados técnicos de uma série de motores de passo semelhante à utilizada no **RAM-I.**

## 2.11 Baterias

Para um robô móvel ser dito autônomo (em energia), é necessário que possua uma fonte de alimentação (potência) que seja capaz de armazenar energia suficiente para permitir ao robô executar um trabalho útil, e que seja embarcada no próprio robô.

As baterias são uma solução comum ao problemas de armazenamento de energia em robôs móveis. Uma bateria converte energia química em energia elétrica. São apresentados, a seguir algumas caraterísticas importantes (Jones, 1993):

- **Recarregáveis:** Uma unidade que gera energia elétrica através de uma reação química e que não pode ser recarregada é uma célula primaria. Uma que pode ser recarregada é dita de secundária ou bateria de armazenamento.
- **Densidade de Energia (Wh/kg):** É a máxima quantidade de energia por unidade de massa.
- **Capacidade (Amp-hora ou mA-h):** É a energia armazenada na célula. Ela é o produto da Densidade de Energia vezes a massa da bateria.
- **Voltagem (V):** Esta é uma característica particular da reação química na bateria, dependendo do estado de carga da célula.
- **Resistência Interna (Ω):** Quando se faz um corto circuito, a corrente fornecida por uma bateria é limitada pela resistência interna da mesma.
- **Taxa de descarga (mA):** É a taxa na que uma bateria é descarregada. A taxa máxima de descarga é limitada pela resistência interna da bateria.
- **Vida útil:** As baterias perdem carga mesmo quando não estiverem conectadas a uma carga externa. A vida útil é uma medida de quão rapidamente esto ocorre.
- **Dependência da temperatura:** Muitas das propriedades, em particular a capacidade disponível e a vida útil, são afetadas pela temperatura.

Uma bateria ideal tem que ter uma alta Densidade de Energia, manter uma voltagem constante e possuir uma baixa resistência interna. A informação mostrada na Tabela 2.16 pode servir como uma guia para a escolha segundo uma determinada aplicação.

Tabela 2.16: Comparação de características para a seleção dos tipos e tamanhos de baterias.

| Bateria | Recarrega | Densidade Energia (Whr/Kgr) | Voltagem da célula (V) | Capacidade (mAh) | Resistência Interna (Ω) | Comentários |
|---|---|---|---|---|---|---|
| Alkaline | Não | 130 | 1.5 | AA-1400<br>C-4500<br>D-10000 | 0.1 | Bateria primaria. |
| Lead-Acid | Sim | 40 | 2.0 | C-1.2 a 120Ah | 0.006 | Variedade de tamanhos |
| Lithium | Não | 300 | 3.0 | A-1800<br>C-5000<br>D-14000 | 0.3 | Densidade de energia boa, alto custo. |
| Mercury | Não | 120 | 1.35 | Coin-190 | 10 | |
| NiCd | Sim | 38 | 1.2 | AA-1400<br>C-1800<br>D-4000 | 0.009 | Resistência interna baixa. |
| NiMH | Sim | 57 | 1.3 | AA-1100<br>4/3A-2300 | | Densidade de Energia boa. |
| Silver | Não | 130 | 1.6 | Coin-180 | 10 | |
| Zinc-Air | Não | 310 | 1.4 | | | Alta densidade de energia, tamanhos limitados |
| Carbon-Zinc | Não | 75 | 1.5 | D-600 | | Barato mas obsoleto |

Para o robô **RAM-I,** foi escolhido um "pack" de baterias de NiCd, das seguintes características: 38Whr/kg, 1.2V p/cela, 500 mAh, de baixa resistência interna, bateria disponível no mercado, assim como um recarregador especial adaptado aos 12V do "pack" de baterias.

# Capítulo 3

# Arquitetura Subsumption

## 3.1 Introdução

Nos anos oitenta surgiu um grupo de pesquisadores, cujo nome de maior destaque é Rodney Brooks, pesquisador do Artificial Intelligence Lab. do Massachussets Institute of Technology (MIT), que acreditaram que o nível de inteligência humana é muito complexa e pouco entendida para ser decomposta corretamente (Brooks, 1991). O grupo propôs uma arquitetura totalmente diferente para agentes inteligentes autônomos (Maes, 1990), dando uma outra alternativa ao problema de planejamento e modelagem do sistema de controle de robôs autônomos móveis. Claro está que esta arquitetura poderia ser usada também para outras aplicações.

Neste capítulo será feita uma apresentação do princípio de percepção-ação e da Arquitetura *Subsumption*, utilizados na implementação de robôs autônomos móveis. Serão também apresentados alguns exemplos de implementação desta arquitetura, e a descrição do modelo aplicado ao robô **RAM-I.**

## 3.2 Percepção-Ação

O princípio de percepção-ação implementado utilizando a Arquitetura *Subsumption* rejeita a representação do conhecimento como base fundamental de Sistemas Artificiais inteligentes. A idéia se baseia no princípio de que quando a inteligência é abordada de uma maneira incremental, isto é, por camadas adicionais, em que cada uma delas é produtora de uma atividade ou

comportamento específico completo (no sentido de que a atividade produzida por uma camada é independente das produzidas por outras camadas), e com estrita ênfase na interface dos sistema com seu ambiente real através da percepção (do ambiente pelo sistema) e da ação (do sistema em relação ao ambiente), a ênfase da representação do conhecimento desaparece. Em outras palavras, o sistema percebe modificações no ambiente e reage à esta percepção, (Brooks, 1991).

## 3.3 Divisão por Níveis Funcionais

Na abordagem tradicional, para se construir sistemas de controle para robôs autônomos móveis, é feita a divisão do controle em unidades funcionais (figura 3.1). Cada unidade funcional está intimamente ligada aos outros níveis vizinhos, de tal forma que o sistema tem que ser projetado de uma forma completa, pois um nível sozinho não é capaz de realizar nenhuma atividade. Para se adicionar uma nova atividade, é preciso alterar todo o projeto.

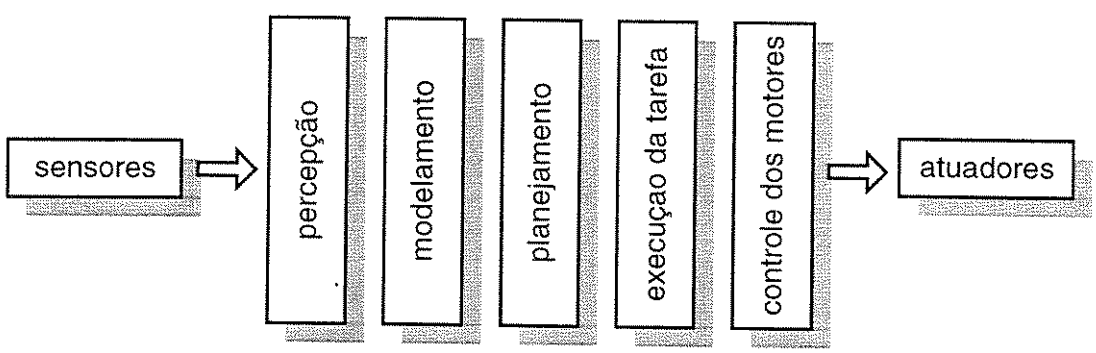



Figura 3.1: Divisão tradicional do sistema de controle em módulos funcionais.

## 3.4 Divisão por Níveis de Atividades

A Arquitetura *Subsumption* difere notavelmente da maneira tradicional de dividir o controle em módulos funcionais: percepção, planejamento, modelamento, etc. Ao invés disso organiza o agente móvel como um conjunto de camadas que representam uma tarefa ou comportamento completo (figura 3.2).

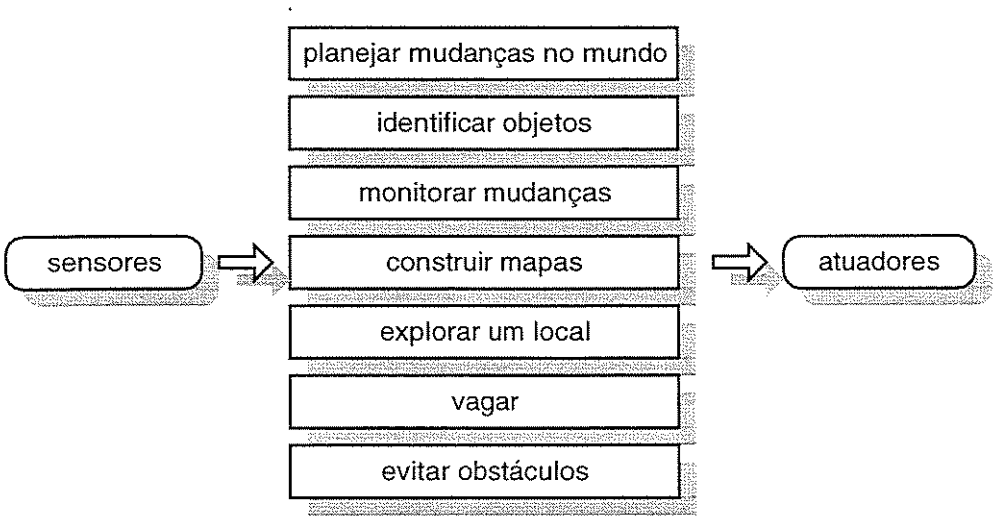



Figura 3.2: Exemplo de divisão do sistema em atividades.

A divisão em camadas de atividades permite acrescentar um comportamento quando este for necessário, construindo a nova camada e interfaceando-a com o sistema. A idéia é construir um sistema autônomo completo, muito simples e testá-lo no mundo real (ver figura 3.3).

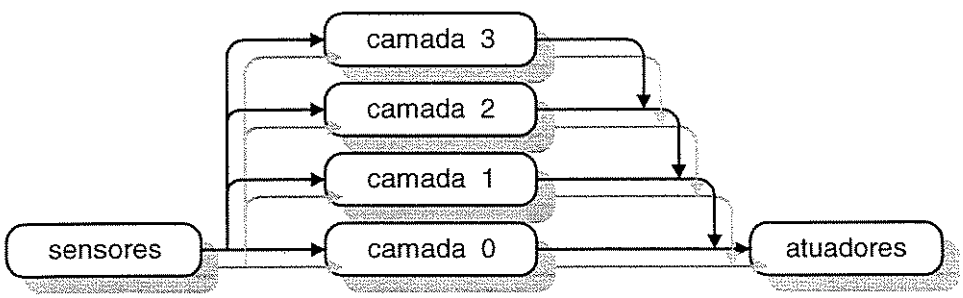



Figura 3.3: Arquitetura *Subsumption*.

O termo Arquitetura *Subsumption* tem origem no fato de que o nível mais elevado assume o papel das camadas inferiores, quando uma determinada configuração dos sensores indica uma situação propicia para sua atuação. O nível acima suprime o fluxo de dados da camada inferior. O sistema pode ser separado em qualquer nível e as camadas inferiores continuarão formando um sistema operacional completo (Rezende, 1992)

Com relação à utilização da metodologia de decompor o sistema em camadas de comportamentos, pode-se inferir o seguinte:

1. Atividades simples dos níveis inferiores podem provocar reações rápidas, devido a que as camadas não utilizam representações complexas do mundo. O sistema está baseado no principio de percepção-ação. A idéia é sentir o ambiente freqüentemente e assim ter uma idéia atualizada do que acontece.

2. Dado que se tem múltiplas atividades paralelas, existe uma menor possibilidade de que qualquer mudança de estado do ambiente provoque um colapso total do sistema. No máximo, incapacitaria algumas camadas do sistema.

3. Cada camada de atividade pode ser imaginada como tendo seu próprio propósito implícito. A idéia principal é usar o mundo como seu próprio modelo e, continuamente, adaptar cada objetivo às condições atuais do ambiente real.

4. A função do robô está implícita em suas camadas.

## *3.5* Exemplos da Implementação da Arquitetura *Subsumption*

Três exemplos de implementação da Arquitetura *Subsumption* serão descritos. A primeira foi usada por Rodney Brooks em um robô móvel do Labotary AI no MIT, conhecido como ALLEN (Brooks, 1986). A Segunda foi realizada por Sridhar Mehadevan e Jonathan Connel em um robô chamado OBELIX do IBM T. J Watson Reserach Center (Rezende, 1992). A terceira foi feita por Marcos Rezende F. no robô MINEIRIN do Centro de Ciências Exatas e Tecnologia da Universidade Federal de Uberlandia (Rezende, 1992). A pesar delas serem implementações da mesma Arquitetura existem notáveis diferenças.

### 3.5.1 Implementação da Arquitetura *Subsumption* no Robô ALLEN.

Rodney A. Brooks desenvolveu vários robôs móveis utilizando a metodologia da decomposição em camadas de atividades, obtendo comportamentos e composição incremental através da depuração do mundo real.

Cada camada, na Arquitetura *Subsumption,* é composta de uma rede de topologia fixa de máquinas de estado finito simples. Cada máquina de estado finito tem uma quantidade de estados, um ou dois registros internos e acesso às máquinas computacionais simples que podem fazer cálculos, como por exemplo soma de vetores.

Máquinas de estado finito são circuitos lógicos seqüenciais nos quais as saídas não dependem apenas das entradas presentes, mas também até certo ponto, da história das entradas. Cada estágio através do qual uma máquina de estado finito avança é chamado um estado.

As máquinas de estado finito são assíncronas, enviando e recebendo mensagens de tamanho fixo por meio de fios (mensagens de 1 bit para robôs pequenos e de 24 bits para os maiores). Estes fios que transportam as mensagens poderão ser virtuais (simulados por "software") ou físicos (Rezende, 1992).

Não há um controle central. As máquinas de estado finito são ativadas pelos dados das mensagens que recebem dos sensores. A mudança de um estado, na máquina de estado finito, ocorre quando chega uma mensagem ou quando o tempo determinado para este estado termina. As máquinas de estado finito têm acesso aos conteúdos das mensagens e podem dar respostas. As camadas são combinadas através de mecanismos chamados supressão e inibição, conforme ilustra a figura 3.4

Tanto na supressão como na inibição, quando uma nova camada é adicionada, ocorre uma ligação entre seus fios e os das camadas existentes. Uma constante de tempo pré-definida é associada a cada ligação. No caso da supressão, a ligação ocorre na entrada da máquina de estado finito. Se uma mensagem chega no fio supressor, ela é direcionada para a porta de entrada da maquina de estado finito, como se estivesse chegando pelo fio que conduz o sinal de entrada. Qualquer nova mensagem sobre o fio existente é suprimida, rejeitada, pelo período de tempo especificado pela constante de tempo. Um módulo para implementar camadas de comportamento tem entradas e saídas. Sinais de entrada podem ser suprimidos e substituídos pelo sinal supressor. Sinais de saída podem ser inibidos.

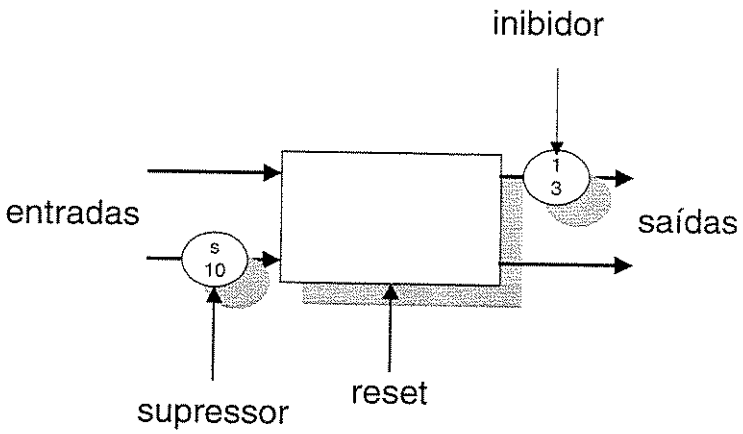



Figura 3.4: Modulo para implementar camadas de comportamento com entradas e saídas.

O robô ALLEN tem três camadas. Possui um conjunto de doze sonares ultra-sônicos, os quais dão uma medida de profundidade a cada segundo. Na figura 3.5 é ilustrado uma leitura dos sensores do ALLEN. Pode-se observar que algumas radiações de ultra-som não retornam ao emissor, indicando a não existência de obstáculos na proximidade do robô.

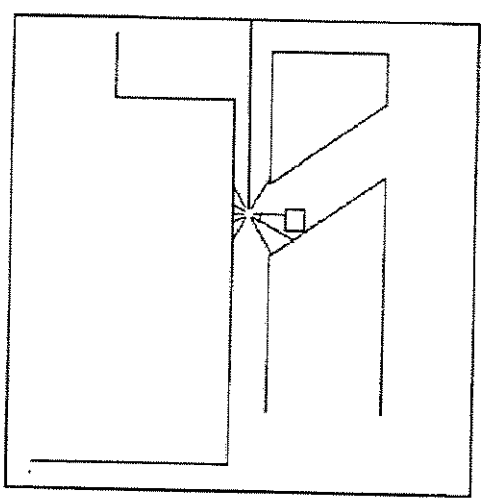


Figura 3.5: Robô ALLEN recebendo a leitura dos doze sensores de ultra-som.

A seguir descreve-se o funcionamento das três camadas utilizadas no robô ALLEN:

A camada de nível mais baixo é a CAMADA ZERO. Esta implementa um comportamento que faz o robô evitar a colisão com obstáculos. Ele consegue evitar obstáculos tanto parados como em movimento.

A máquina de estado finito chamada de SONAR converte, a cada segundo, a leitura dos sensores de ultra-som em coordenas polares, e cria um mapa instantâneo. Esta leitura é transmitida às máquinas de estado finito COLISÃO e REPULSÃO. COLISÃO verifica se tem algo fixo adiante. Se houver , envia uma mensagem de parada à máquina de estado finito PARA FRENTE. Em simultâneo REPULSÃO calcula uma força repulsiva sobre o robô que é enviada para a máquina de estado finito FUGIR, quem indica a direção e o sentido para a máquina de estado finito VIRAR, a qual faz o robô virar na direção oposta ao do objeto repulsivo. Finalmente a máquina PARA FRENTE comanda de novo o robô até receber uma mensagem de parada.

Esta camada permite ao robô ALLEN ficar no meio de uma sala vazia, afastar-se logo se alguém vier na sua direção, e parar se for em direção a um obstáculo muito próximo, cumprindo assim o seu objetivo de evitar bater ou ser atingidos por objetos. Esta camada está representada na figura 3.6.

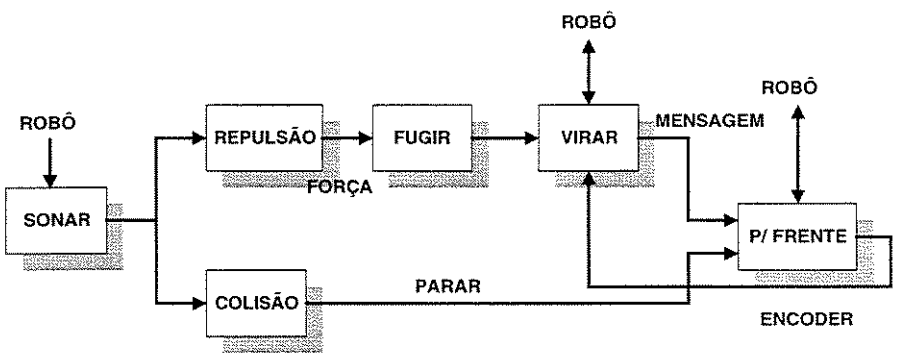



Figura 3.6: Camada ZERO do robô móvel ALLEN.

A próxima camada, CAMADA UM, faz o robô vagar quando não está evitando obstáculos. A máquina de estado finito VAGAR gera uma direção randômica para o robô a cada dez minutos. A máquina EVITAR recebe esta direção como uma força atrativa que é adicionada à força repulsiva fornecida pela máquina de estado finito REPULSÃO. O resultado desta soma serve para suprimir o resultado do nível mais baixo, forçando o robô a ir em uma direção próxima à fornecida pela máquina VAGAR, porem ao mesmo tempo evitando obstáculos (figura 3.7).

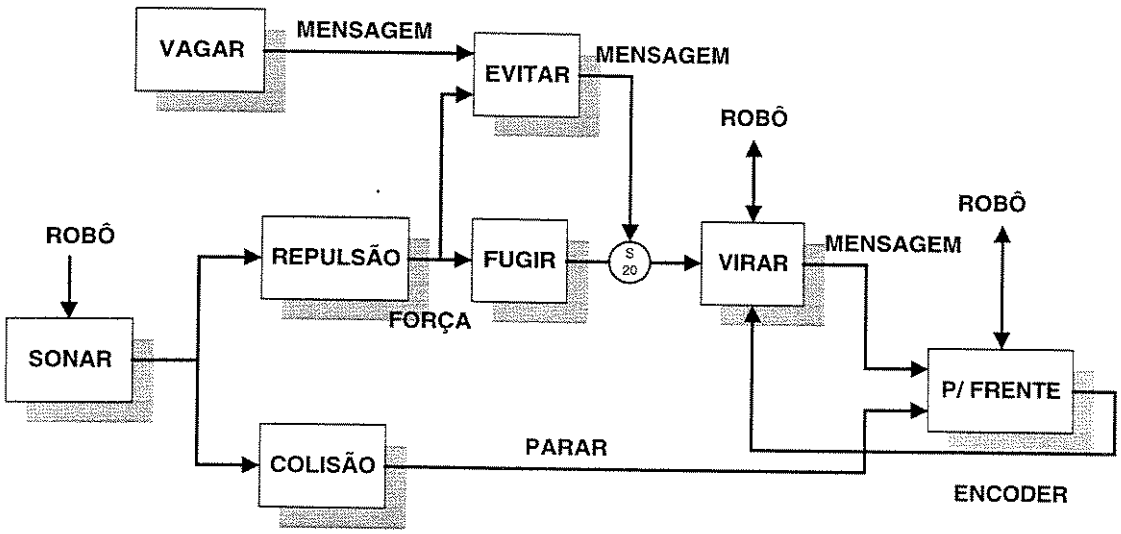



Figura 3.7: Camada ZERO e camada UM do robô móvel ALLEN.

A figura 3.8 é uma representação do mundo em duas dimensões usada para simular o comportamento do robô quando está sendo controlado pelas camadas ZERO e UM.

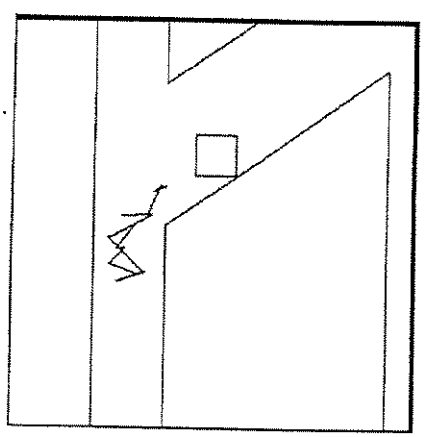


Figura 3.8: Trajetória do robô quando controlado pelas camadas ZERO (evitar obstáculos) e UM (vaguear).

A terceira camada, CAMADA DOIS, incrementa a capacidade de fazer o robô explorar. A máquina de estado finito QUANDO-OLHAR, percebe quando o robô não está ocupado movimentando-se e ativa a máquina DESCOBRIR, descobridora de espaço livre, e inibe o comportamento VAGUEAR. Quando um caminho é observado, é enviado à máquina PLANEJAMENTO que planeja a trajetória enviando uma direção para a máquina EVITAR. Desta forma o nível mais baixo continua agindo. Isto pode levar o robô a uma direção diferente

da planejada. Para que o robô persiga o objetivo planejado o caminho real do robô é monitorado pela máquina INTEIRAR, que envia estimativas atualizadas à máquina PLANEJAMENTO. Esta máquina age de maneira diferente, forçando o robô para a direção desejada e assim compensando os desvios causados pelos obstáculos, realizados pela camada inferior.

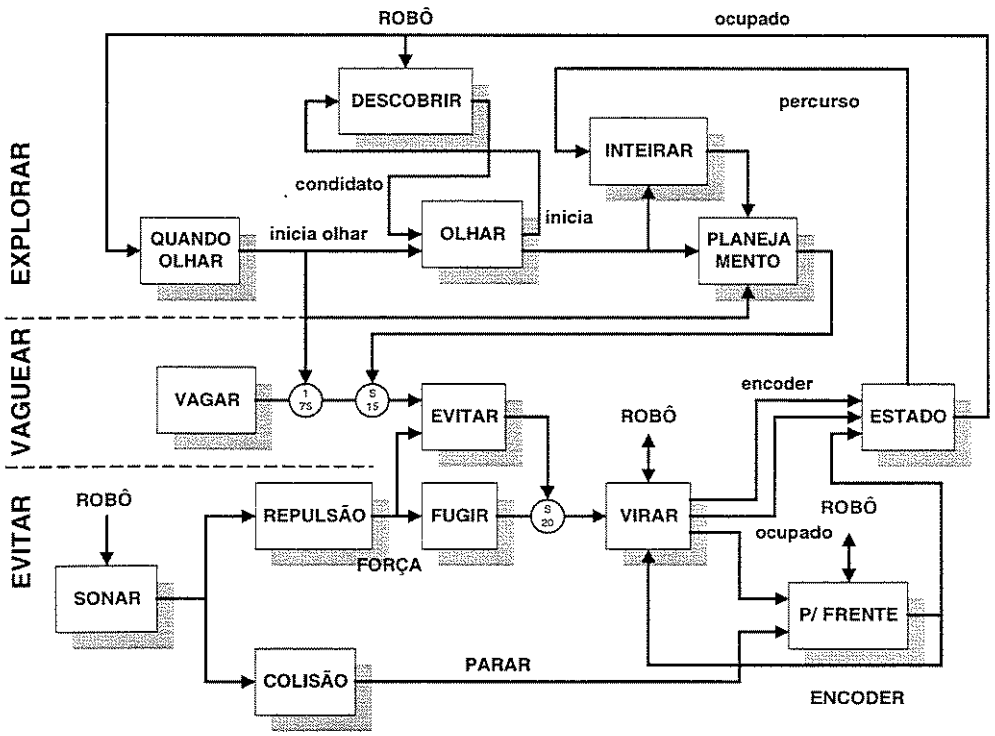



Figura 3.9: Conjunto das três camadas utilizadas no robô ALLEN.

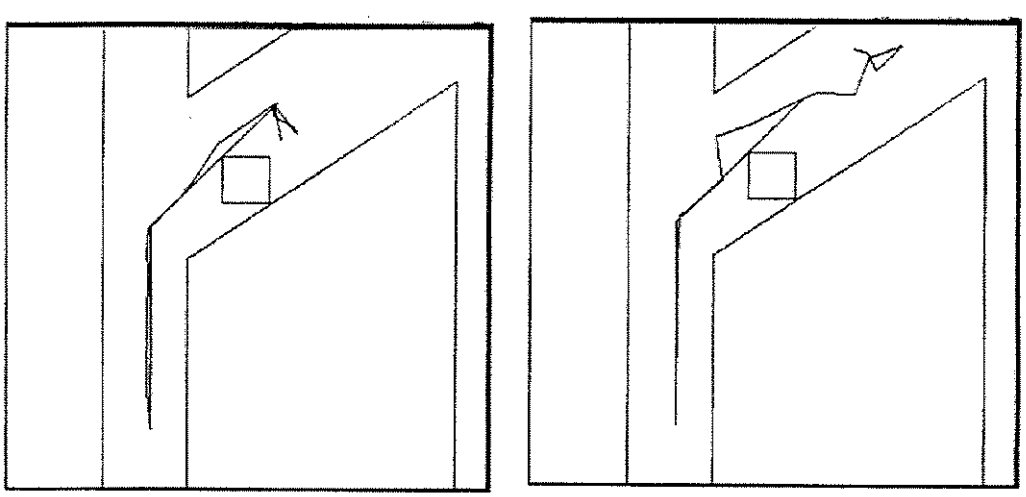


Figura 3.10: Trajetórias do robô ALLEN controlado pelas três camadas ZERO (evitar obstáculos), UM (vaguear) e DOIS (explorar).

### 3.5.2 Implementação da Arquitetura *Subsumption* no Robô OBELIX.

Manhadevan e Connell implementaram esta Arquitetura no robô OBELIX. Como é mostrado na figura 3.11, uma Arquitetura do estilo *Subsumption* opera através da divisão do sistema de controle central em um número de pequenos processos paralelos e concorrentes. Cada um desses comportamentos usa um subconjunto de dados avaliados pelos sensores para controlar os parâmetros de saída. Uma rede de fios arbitrária ou um bloco de controle de prioridade combina os resultados individuais dando o comando aos atuadores.

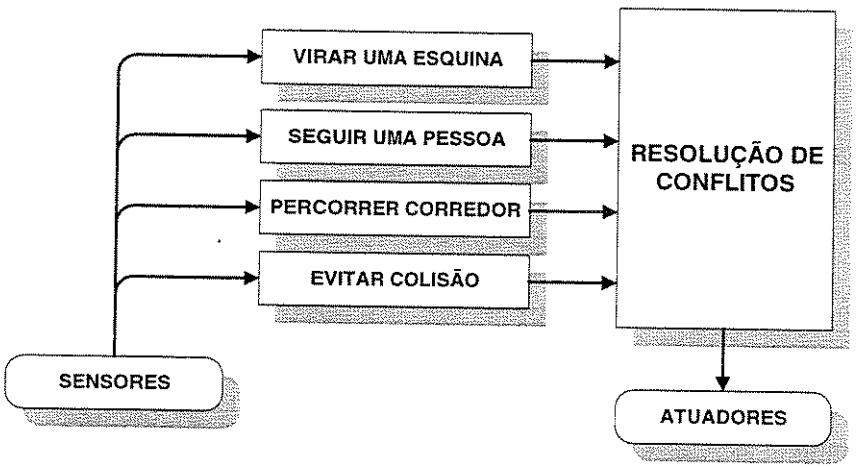



Figura 3.11: Arquitetura *Subsumption* composta de caminhos paralelos e uma rede de resolução de conflitos.

Nesta implementação cada camada de controle consiste de um módulo, que gera um comportamento específico no robô. Um módulo tem dois componentes internos: O bloco PLANO DE AÇÃO diz o que fazer com uma informação sensorial atualizada, e o bloco PREDICADO DE APLICABILIDADE diz quando fazer, cancelando o plano quando seus avisos são questionáveis ou sem sentido (Rezende, 1992).

Para se construir um sistema de controle usando a Arquitetura *Subsumption*, as tarefas globais são divididas primeiramente em um numero de subtarefas. Depois, para cada subtarefa, dispositivos convenientes são projetados com planos de geração de ação e condições de aplicabilidade. Então as ordens de prioridades dos comportamentos são definidas permitindo ao sistema resolver algum conflito que possa existir entre as camadas.

O robô OBELIX conta com oito sensores, voltados sempre na direção do movimento. O número de movimentos foi limitado a cinco, o robô pode ir para frente, virar para direita ou esquerda com duas possibilidades de ângulo (22° e 45°). Ele nunca para ou se movimenta para traz. Os principais sensores são de ultra-som, quatro na frente e dois nas laterais, e um infravermelho na frente. Também possui um sensor para detectar o nível de corrente dos seus motores, porque a sua tarefa de nível mais elevado, empurrar caixas, exige o contato com obstáculos que podem imobilizá-lo.

A principal tarefa do OBELIX é empurrar caixas, o que pode ser dividido em três subtarefas:

1. Encontrar uma caixa.
2. Empurrar uma caixa através de uma sala
3. Livrar-se de situações embaraçosas, como por exemplo empurrar uma caixa para um canto ou tentar empurrar um objeto fixo como uma parede.

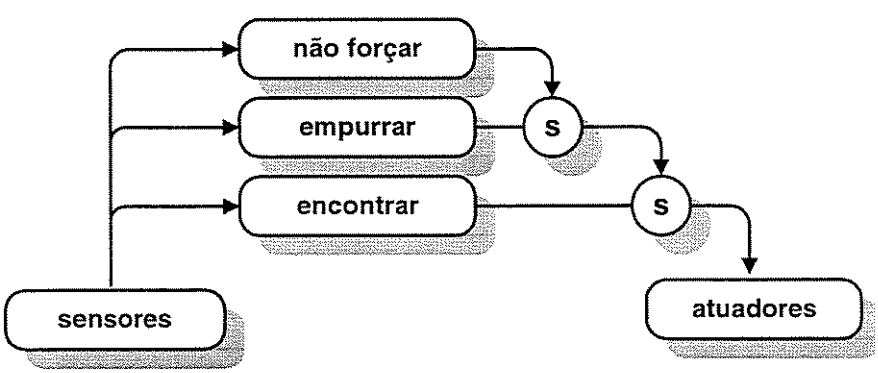



Figura 3.12: Camadas no sistema de controle do robô OBELIX.

Da figura 3.12 pode se inferir que o comportamento NÃO-FORÇAR é o de prioridade mais elevada. Se o robô está em uma situação embaraçosa é importante tirar o robô desta situação de perigo.

O comportamento EMPURRAR tem a próxima prioridade. Se ele esta empurrando uma caixa então ele continuará enquanto puder.

Por último, a tarefa ENCONTRAR é a de menor prioridade. Se ele estiver sem perigo e livre de caixa, percorre o espaço procurando uma caixa.

Uma forma interessante de obter a ordem de prioridades nos comportamentos é examinar a seqüência em que elas se tornarão aplicáveis. No inicio o robô precisa encontrar uma caixa, menor ordem de prioridade, para depois empurrá-la. Finalmente, se encontrar um obstáculo na frente, ele precisa parar de empurrar para não danificar-se, maior ordem de prioridade.

### 3.5.3 Implementação da Arquitetura *Subsumption* no Robô MINEIRIN

Rezende implementou esta Arquitetura no robô MINEIRIN. Neste robô são montadas três camadas (figura 3.13):

1. EVITAR obstáculos.
2. VAGAR sem direção fixa.
3. ENCONTRAR objetivo para reenergização.

O comportamento EVITAR é o de prioridade mis elevada. Se o robô está para colidir com um obstáculo ele deve evitá-lo. Quando o robô não está evitando nenhum obstáculo ele fica parado, se alguém tenta se aproximar dele, ele se afasta. Por períodos de tempo programados de 30 em 30 segundos o robô verifica se não está ativa a camada EVITAR obstáculos e, então, gera um movimento aleatório para o robô ficar VAGUEANDO no local.

A camada ENCONTRAR, quando percebe que as baterias do robô estão descarregadas, ativa um procedimento que tenta encontrar o local de recarga. Este comportamento gera direções randômicas, até que o sensor de infravermelho indique a direção correta que o robô deve seguir. Sua potencialidade se verifica no caso de encontrar um obstáculo. Ele tentará evita-lo e continuar o seu percurso em direção ao alvo.

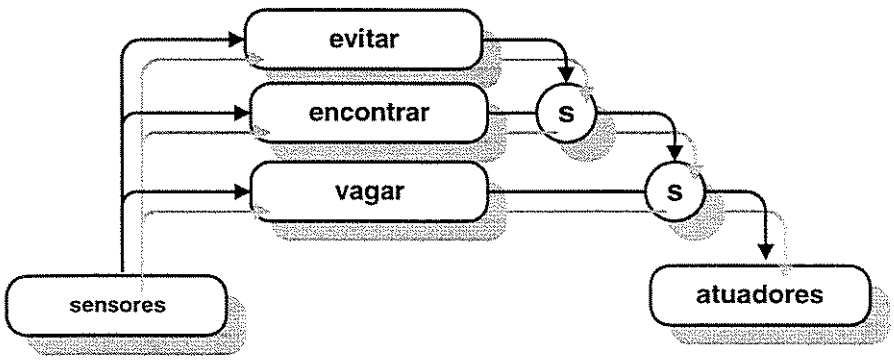



Figura 3.13: Camadas no sistema de controle do robô MINEIRIN.

### 3.5.4 Descrição das Camadas Implementadas no Robô RAM-I

O robô **RAM-I,** construído no Departamento de Mecânica Computacional, e que será descrito nos seguintes capítulos, teve a sua estratégia de controle implementada usando Arquitetura *Subsumption* a fim de testar a sua capacidade. Ele apresenta um comportamento simples com duas tarefas específicas. A de VAGAR num determinado *ambiente* tendo para este fim que EVITAR os possíveis obstáculos que impedem o seu livre deslocamento. Neste robô são montadas duas camadas (figura 3.14):

1. EVITAR obstáculos.
2. VAGAR sem direção fixa.

O comportamento EVITAR é o de prioridade mis elevada. Se o robô está para colidir com um obstáculo ele deve evitá-lo. Quando o robô não está evitando nenhum obstáculo está ativa a camada VAGAR, a qual gera uma trajetória retilínea do robô no local.

Quando ligado, o robô é comandado pela camada de prioridade mais baixa, VAGAR, e começa a andar em linha reta. Neste estado, ele não é afetado por eventuais perturbações que alterem a sua direção, como por exemplo escorregamento das rodas. O robô permanece neste estado até que a camada de nível mas elevado, EVITAR, é ativada pela presença de um obstáculo.

Quando a camada EVITAR é ativada, o robô identifica se o obstáculo é um objeto ou uma depressão, e a sua localização (direita ou esquerda). Em qualquer um destes casos o robô efetua uma manobra para evitar o obstáculo, retrocedendo e girando para o lado oposto ao do obstáculo encontrado.

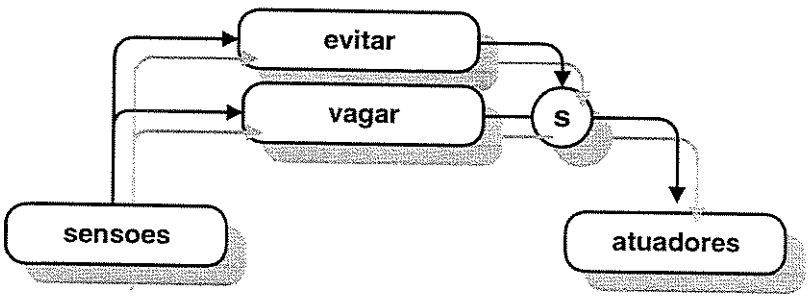



Figura 3.14: Camadas no sistema de controle do robô RAM-I

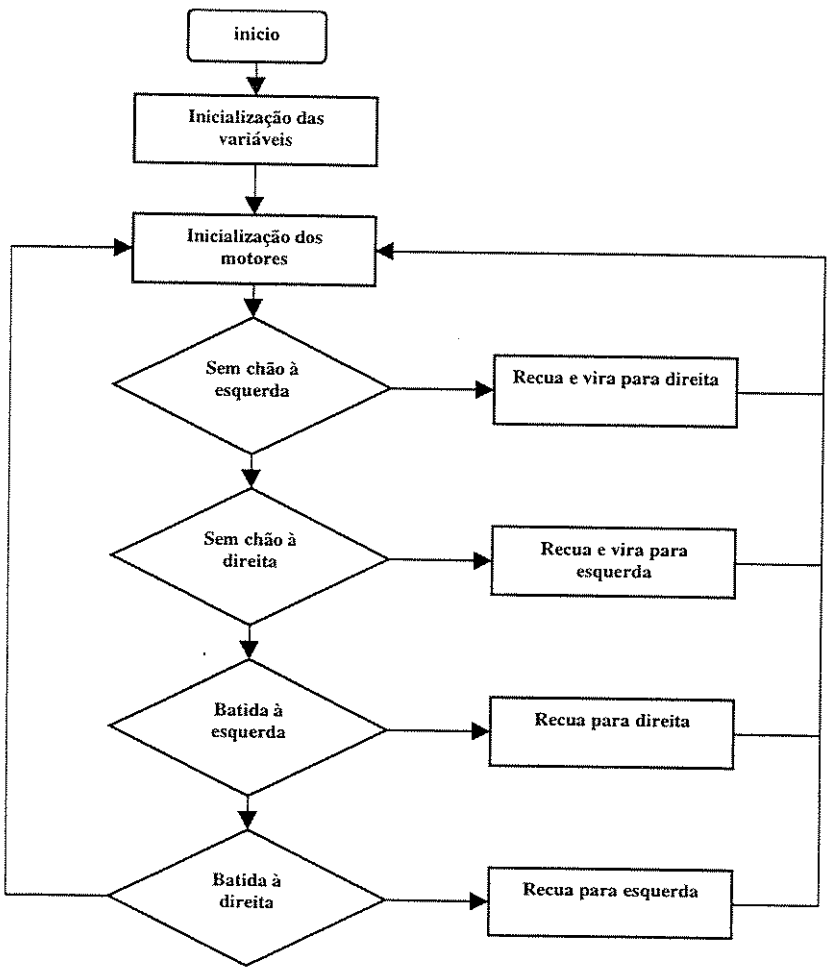



Figura 3.15: Diagrama de fluxo do programa implementado no robô RAM-I

# Capítulo 4

# Modelagem de Robôs Móveis

## 4.1 Introdução

Este capítulo será dividido em duas partes: A primeira parte abordará a modelagem matemática dos robôs móveis com rodas, incluindo a descrição dos tipos de rodas, restrições de movimento e os modelos de estado derivados do modelamento das rodas. Inicialmente trata-se de generalizar o modelamento para robôs móveis com rodas mostrando vários exemplos, e depois é feita uma aplicação particular para o caso do robô móvel **RAM-I.** Na Segunda parte é feita uma introdução das definições chaves e teoremas de não-holonomicidade. Esta é feita porque os robôs móveis sobre rodas são sistemas mecânicos não-holonômicos.

Atingiu-se atualmente um progresso significativo no planejamento do movimento de robôs móveis com rodas (carros). Estes veículos têm muitas aplicações potenciais, mas o seu planejamento é dificultoso pois são objeto de restrições de rolamento, as quais limitam as possíveis direções do movimento. Eles não podem mover-se lateralmente, mas podem ir para a frente ou para trás em trajetórias curvas, a fim de girar. Manobras complicadas podem ser requeridas para se mover de uma configuração a uma outra próxima. Veículos dirigidos a leme como navios e aviões são objeto de restrições similares. Tais restrições, que limitam a possível direção do movimento num ponto, mas que podem ser "desfeitas" por manobras locais , são chamadas de não-holonômicas.

## 4.2 Modelo Cinemático de uma Base Móvel

### 4.2.1 A Postura do Robô

Os robôs em estudo são considerados compostos de corpos rígidos, equipados com rodas indeformáveis e se movimentam em um plano horizontal. Na figura 4.1 vemos a posição do robô numa base inercial $\{0,I_1,I_2\}$ fixa no plano de movimento e um ponto de referência $P$, onde um referencial móvel $\{X_1,X_2\}$ é fixado.

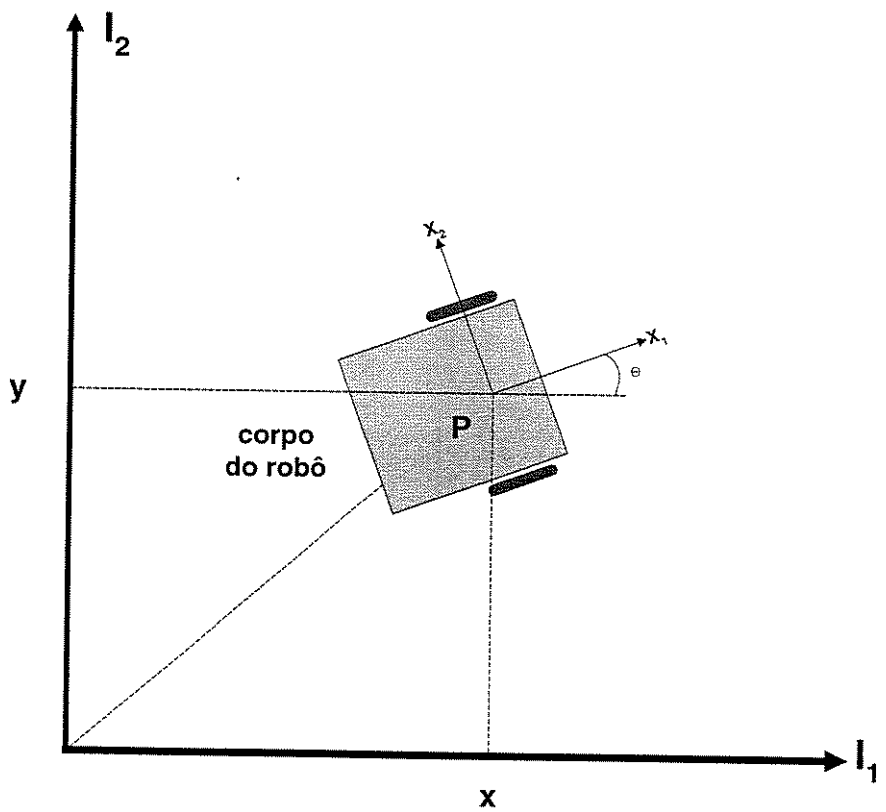



Figura 4.1: Definição da postura do robô móvel.

Nesta figura:

- $x$, $y$ são as coordenadas do ponto **P** de referência na base inercial, isto é,

$$\overline{OP} = x\overline{I_1} + y\overline{I_2}$$

- $\theta$ é a orientação da base $\{P, X_1, X_2\}$ em relação à base inercial $\{0,I_1,I_2\}$.

Pode-se então definir um vetor $\xi$ que descreve a postura do robô.

$$\xi = \begin{Bmatrix} x \\ y \\ \theta \end{Bmatrix} \tag{4.1}$$

e a matriz ortogonal de rotação:

$$R(\theta) = \begin{bmatrix} \cos\theta & \text{sen}\,\theta & 0 \\ -\text{sen}\,\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

Esta matriz transforma (rotaciona) uma coordenada escrita no sistema inercial $A_I = \{x, y, 0\}$, para o sistema móvel solidario à base móvel

$$A_m = R(\theta) A_I .$$

### 4.2.2 Descrição das Rodas

O robô implementado se desloca sobre rodas. Por essa razão o comportamento das mesmas têm um papel importante no modelo. A seguir é apresentada a descrição de alguns tipos de rodas e as suas caraterísticas.

Considera-se que as rodas, durante o movimento, permanecem no plano vertical e giram em torno do seu eixo horizontal, que pode permanecer fixo ou variar em relação ao chassi do robô. A seguir, descreve-se as duas classes de rodas segundo Campion (1996): as convencionais e as suecas. Considere-se que o contato dessas com o chão, se dá através de um único ponto no plano.

Para as rodas convencionais, o contato entre a roda e o chão é suposto para satisfazer o rolamento puro sem escorregamento: isto significa que a velocidade do ponto de contato é igual a zero. Por tanto, as suas componentes, a velocidade paralela e a velocidade ortogonal ao plano da roda, são iguais a zero.

Para as rodas suecas, só uma componente da velocidade do ponto de contato da roda com o chão é considerada igual a zero ao longo do movimento. A direção desta componente é arbitraria, mas é fixa com respeito à orientação da roda.

➢ **Rodas Convencionais.**

Como já foi dito, par as rodas convencionais não existe deslizamento no ponto de contato entre a roda e o chão satisfazendo a condição de rolamento puro.

As rodas convencionais podem ser classificadas em três tipos:

1. **Rodas Fixas:** O centro da roda, denotado por $A$, é um ponto fixo no chassi do robô (figura 4.2). A posição de $A$ na base $\{X_1,X_2\}$ é descrita através das coordenadas polares $\overline{PA}=l$ e $\alpha$. A orientação do plano da roda em relação a $\overline{PA}$ é representado por um ângulo fixo $\beta$, a rotação da roda em relação ao seu eixo horizontal é denotado por $\varphi(t)$, e o raio da roda é $r$. A posição da roda no plano é então caracterizada pelas constantes $\alpha$, $\beta$, $l$, $r$, e pelo ângulo de rotação em torno do seu próprio eixo $\varphi(t)$. Com esta descrição, as componentes de velocidade nos pontos de contato são denotados através das seguintes restrições.

   - Ao longo do plano da roda

$$[-sin(\alpha+\beta) \quad \cos(\alpha+\beta) \quad l\cos\beta]R(\theta)\dot{\xi}+r\dot{\varphi}=0 \tag{4.3}$$

   - Ortogonal ao plano da roda

$$[\cos(\alpha+\beta) \quad sin(\alpha+\beta) \quad lsin\beta]R(\theta)\dot{\xi}=0 \tag{4.4}$$

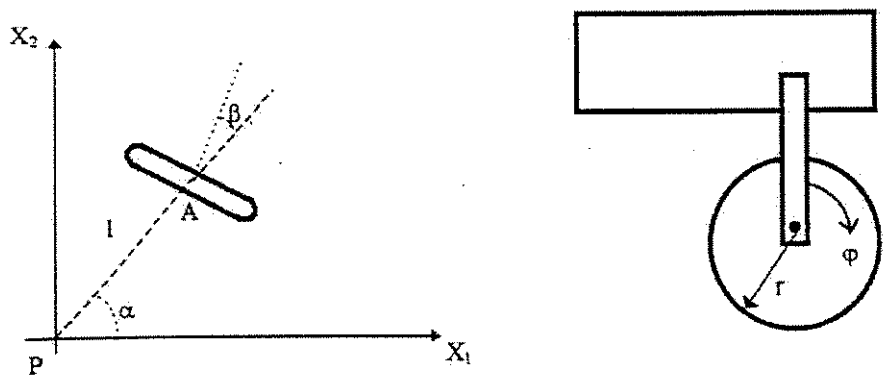



Figura 4.2: Rodas convencionais fixas e orientadas no centro

2. **Rodas Orientadas no Centro:** Seu movimento em relação ao sistema de referência é uma rotação em torno de um eixo vertical passando através do seu centro, A construção é a mesma que para as rodas fixas (figura 4.2), porém, o ângulo $\beta(t)$, não é mas constante. Ele pode ser alterado ao longo do tempo. A posição da roda é caracterizada pelas constantes $\alpha$, $l$, $r$ e seu movimento em relação ao sistema de referência móvel por dois ângulos variantes com o tempo $\varphi(t)$, $\beta(t)$. As equações cinemáticas de restrição ao movimento possuem a mesma forma que para as rodas fixas:

   - Ao longo do plano da roda

$$[-sin(\alpha+\beta) \quad \cos(\alpha+\beta) \quad l\cos\beta]R(\theta)\dot{\xi}+r\dot{\varphi}=0 \tag{4.5}$$

   - Ortogonal ao plano da roda

$$[\cos(\alpha+\beta) \quad sin(\alpha+\beta) \quad lsin\beta]R(\theta)\dot{\xi}=0 \tag{4.6}$$

3. **Rodas Orientadas Fora do Centro:** Possuem orientação em relação ao chassi do robô, mas a rotação do plano da roda se dá em torno do eixo vertical que não passa pelo seu centro *B*. O eixo da roda está conectado ao chassi do robô pela haste rígida *AB*, a qual pode girar em torno do eixo vertical que passa por *A*. A distância entre o centro da roda e o eixo vertical que passa por *A* é expressa por *d* (figura 4.3). A posição da roda é descrita por quatro constantes $\alpha$ , *l*, *r* e *d*, e seu movimento em relação ao sistema de referência por dois ângulos variantes com o tempo, $\varphi(t)$ **e** $\beta(t)$. As equações cinemáticas, para tais rodas, são as seguintes:

- Ao longo do plano da roda

$$[-sin(\alpha+\beta) \quad \cos(\alpha+\beta) \quad l\cos\beta]R(\theta)\dot{\xi}+r\dot{\varphi}=0 \tag{4.7}$$

- Ortogonal ao plano da roda

$$[\cos(\alpha+\beta) \quad sin(\alpha+\beta) \quad d+lsin\beta]R(\theta)\dot{\xi}+d\dot{\beta}=0 \tag{4.8}$$

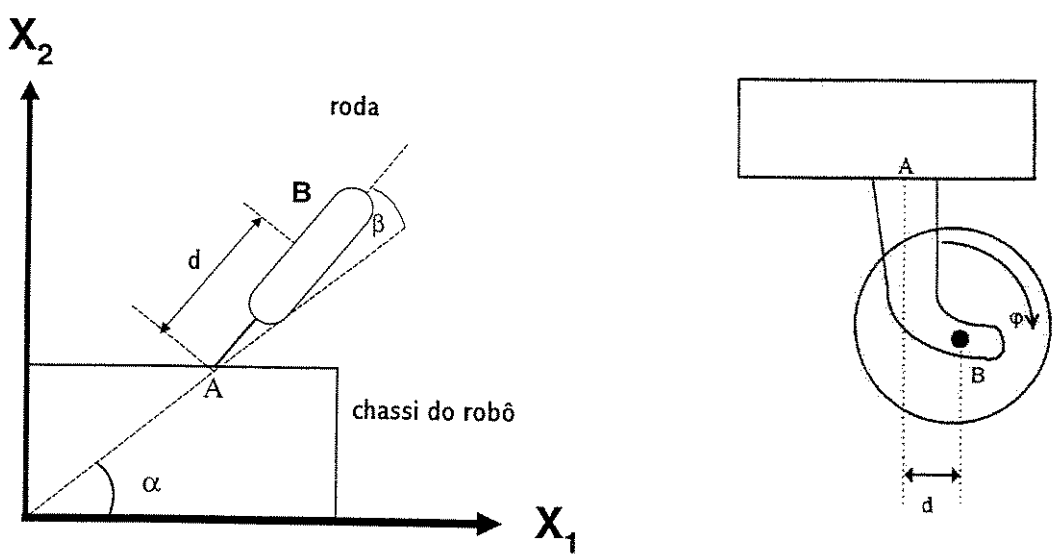



Figura 4.3: Rodas convencionais orientadas fora do centro.

➢ **Rodas Suecas**

Neste caso somente uma das componentes da velocidade do ponto de contato da roda com o chão é considerada igual a zero, diferentemente das rodas convencionais onde todas as componentes da velocidade do ponto de contato são consideradas iguais a zero, ao longo do movimento. A posição da roda com relação ao sistema de referência móvel é descrito como para as rodas convencionais fixas, por três parâmetros constantes, $\alpha$, $\beta$, $l$. Só um parâmetro mais é requerido para caracterizar a direção em relação ao plano da roda, segundo a qual a componente da velocidade no ponto de contato é zero, e será o ângulo $\gamma$. A equação de restrição cinemática é escrita da seguinte forma.

- Ao longo da direção representada por $\gamma$ na figura 4.4

$$\left[-sin(\alpha+\beta+\gamma) \quad \cos(\alpha+\beta+\gamma) \quad l\cos(\beta+\gamma)\right]R(\theta)\dot{\xi}+r\cos\gamma\varphi=0 \tag{4.9}$$

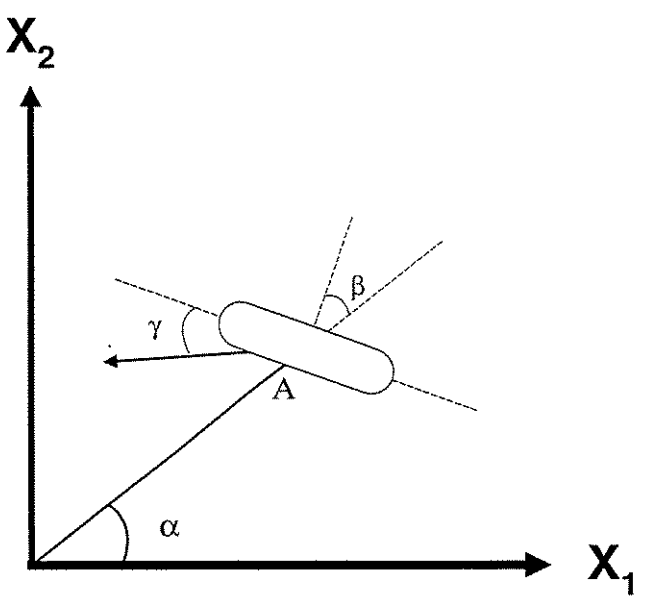



Figura 4.4: Rodas Suecas.

Tabela 4.1: Restrições cinemáticas para cada classe de rodas

| | |
|---|---|
| **Rodas Fixas** | |
| **Ao longo do plano da roda:** | |
| $[-sin(\alpha+\beta) \quad \cos(\alpha+\beta) \quad l\cos\beta]R(\theta)\dot{\xi}+r\dot{\varphi}=0$ | **(4.3)** |
| **Ortogonal ao plano da roda:** | |
| $[\cos(\alpha+\beta) \quad sin(\alpha+\beta) \quad lsin\beta]R(\theta)\dot{\xi}=0$ | **(4.4)** |
| **Rodas Orientadas no centro** | |
| **Ao longo do plano da roda:** | |
| $[-sin(\alpha+\beta) \quad \cos(\alpha+\beta) \quad l\cos\beta]R(\theta)\dot{\xi}+r\dot{\varphi}=0$ | **(4.5)** |
| **Ortogonal ao plano da roda:** | |
| $[\cos(\alpha+\beta) \quad sin(\alpha+\beta) \quad lsin\beta]R(\theta)\dot{\xi}=0$ | **(4.6)** |
| **Rodas Orientadas fora do Centro** | |
| **Ao longo do plano da roda:** | |
| $[-sin(\alpha+\beta) \quad \cos(\alpha+\beta) \quad l\cos\beta]R(\theta)\dot{\xi}+r\dot{\varphi}=0$ | **(4.7)** |
| **Ortogonal ao plano da roda:** | |
| $[\cos(\alpha+\beta) \quad sin(\alpha+\beta) \quad d+lsin\beta]R(\theta)\dot{\xi}+d\dot{\beta}=0$ | **(4.8)** |
| **Rodas Suecas** | |
| **Ao longo da direção representada por $\gamma$:** | |
| $[-sin(\alpha+\beta+\gamma) \quad \cos(\alpha+\beta+\gamma) \quad l\cos(\beta+\gamma)]R(\theta)\dot{\xi}+r\cos\gamma\varphi=0$ | **(4.9)** |

### 4.2.3 Mobilidade e Classificação dos Robôs Móveis.

Um robô móvel pode ser equipado com $N$ rodas podendo ter $N_f$ rodas fixas, $N_c$ rodas convencionais orientadas no centro, $N_{fc}$ orientadas fora do centro, e $N_{sw}$ rodas suecas.

$$N = N_f + N_c + N_{fc} + N_{sw}$$

A configuração do robô no plano cartesiano é completamente descrita pelos seguintes vetores de coordenadas.

Postura $\xi = \begin{Bmatrix} x \\ y \\ \theta \end{Bmatrix}$.

Coordenada angular $\beta_c(t)$ e $\beta_{fc}(t)$

Coordenadas de rotação $\varphi = \begin{Bmatrix} \varphi_f(t) \\ \varphi_c(t) \\ \varphi_{fc}(t) \\ \varphi_{sw}(t) \end{Bmatrix}$, ângulos de rotação das rodas em torno do centro.

Os sub-índices indicam respectivamente: *f* - roda fixa; *c* – orientada no centro; *fc* – orientada fora do centro; *sw* - roda sueca.

O conjunto $\{\xi, \beta_c(t), \beta_{fc}(t), \varphi\}$ é chamado de coordenadas de configuração, sendo o número de coordenadas de configuração igual a:

$$N_f + 2N_c + 2N_{fc} + N_{sw} + 3$$

As equações de restrições cinemáticas (Tabela 4.1), podem ser resumidas em uma forma matricial compacta:

$$J_1(\beta_c, \beta_{fc})R(\theta)\dot{\xi} + J_2\dot{\varphi} = 0 \tag{4.10}$$

$$C_1(\beta_c, \beta_{fc})R(\theta)\dot{\xi} + C_2\dot{\varphi} = 0 \tag{4.11}$$

Com as seguintes definições:

$$J_1(\beta_c\beta_{fc}) = \begin{Bmatrix} J_{1f} \\ J_{1f}(\beta_c) \\ J_{1fc}(\beta_{fc}) \\ J_{1sw} \end{Bmatrix};\ J_2 = diag(r, r\cos\gamma).$$

$J_{1f}, J_{1c}, J_{fc}, J_{sw}$ são matrizes com dimensões ($N_f$ x 3), ($N_c$ x 3), ($N_{fc}$ x 3), ($N_{sw}$ x 3), respectivamente, cuja forma vem diretamente das equações das restrições ao longo do plano da roda (4,3), (4,5), (4,7), (4,9). $J_2$ é uma matriz *(N* x *N)* cuja diagonal são os raios das rodas, exceto para o raio das rodas suecas o qual é multiplicado por $\cos\gamma$.

$$C_1(\beta_c \beta_{fc}) = \begin{Bmatrix} C_{1f} \\ C_{1f}(\beta_c) \\ C_{1fc}(\beta_{fc}) \end{Bmatrix};\ C_2 = \begin{Bmatrix} 0 \\ 0 \\ C_{2fc} \end{Bmatrix}$$

$C_{1f}, C_{1c}, C_{1fc}$ são matrizes com dimensões ($N_f$ x 3), ($N_c$ x 3), ($N_{fc}$ x 3), cuja forma vem diretamente das restrições na direção ortogonal ao plano da roda. $C_{2fc}$ é uma matriz diagonal constante com elementos iguais a $d$, parâmetro das $N_{fc}$ rodas orientadas fora do centro.

Pode-se introduzir a seguinte suposição com respeito à configuração das roda suecas:

A1:Para cada roda sueca: $\gamma \neq \frac{\pi}{2}$. O valor de $\gamma = \frac{\pi}{2}$ corresponderá a direção da componente zero da velocidade que é ortogonal ao plano da roda. Tal roda seria sujeita a uma restrição idêntica à restrição de não escorregamento das rodas convencionais, daí o beneficio de implementar rodas suecas.

Considerando as $(N_f + N_c)$ primeiras equações de (4.11), pode-se escrevê-las explicitamente como:

$$C_{1f} R(\theta) \dot{\xi} = 0 \tag{4.12}$$

$$C_{1c}(\beta_c) R(\theta) \dot{\xi} = 0 \tag{4.13}$$

Essas equações implicam que o vetor $R(\theta)\dot{\xi}$ pertence ao espaço nulo da matriz $C_1^*(\beta_c)$ definida como:

$$C_1^*(\beta_c) = \begin{Bmatrix} C_{1f} \\ C_{1c}(\beta_c) \end{Bmatrix} \tag{4.14}$$

$$R(\theta)\dot{\xi} \in N\left[C_1^*(\beta_c)\right] \tag{4.15}$$

Sabe-se, assim, que o posto (*rank*) da matriz $\left|C_1^*(\beta_c)\right| \leq 3$ e depende do projeto do robô móvel. Se $\left|C_1^*(\beta_c)\right| = 3$ então $R(\theta)\dot{\xi} = 0$ e qualquer movimento no plano é impossível. Uma interpretação geométrica das equações (4.12) e (4.13) é que em cada instante o movimento do robô pode ser visto como uma rotação em torno do centro instantâneo de rotação (CIR) cuja posição relativa ao sistema de referência móvel pode ser variável com o tempo. Em cada instante também o vetor de velocidade de qualquer ponto no chassi do robô é perpendicular à linha reta que une este ponto ao CIR. Esto é ilustrado na figura 4.5 e é equivalente à condição de posto $[C_1^*(\beta_c)] \leq 2$.

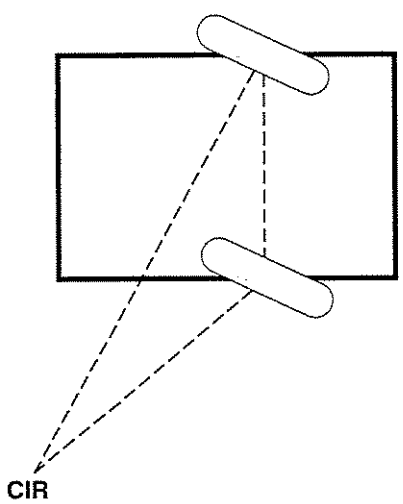



Figura 4.5: Centro instantâneo de rotação (CIR)

O grau de mobilidade para um robô móvel pode então ser definido da seguinte forma

$$\delta_m = \dim N[C_1^*(\beta_c)] = 3 - posto[C_1^*(\beta_c)]$$

O $posto[C_{1c}(\beta_c)]$ é o número de rodas convencionais orientadas no centro que podem ser guiadas independentemente para dirigir o robô. Este número é chamado de grau de dirigibilidade $\delta_s$:

$$\delta_s = posto[C_{1c}(\beta_c)]$$

Pode-se agrupar os robôs móveis não singulares ou não-degenerados em cinco classes diferentes de interesse prático, de acordo com o grau de mobilidade $\delta_m$ e de dirigibilidade $\delta_s$. As classes são separadas de acordo com o par $(\delta_m, \delta_s)$.

Tabela 4.2: Classes de robôs móveis de acordo com $(\delta_m, \delta_s)$

| $\delta_m$ | 3 | 2 | 2 | 1 | 1 |
|---|---|---|---|---|---|
| $\delta_s$ | 0 | 0 | 1 | 1 | 2 |

Essas classes são formadas devido às seguintes condições sobre $\delta_m$ e $\delta_s$:

- O grau de mobilidade $\delta_m$ satisfaz a inequação

$$1 \leq \delta_m \leq 3 \tag{4.16}$$

Os limites indicam o número máximo de graus de liberdade que o robô pode ter no plano, as translações nas direciones $x$, $y$ e a rotação $\theta$, e como mínimo um movimento possível.

- O grau de dirigibilidade $\delta_s$ satisfaz a inequação

$$1 \leq \delta_s \leq 2 \tag{4.17}$$

O limite superior aplica-se para os robôs sem rodas fixas $(N_f = 0)$, e o inferior corresponde aos robôs sem rodas convencionais orientadas no centro $(N_c = 0)$.

- Alem disso satisfaz a inequação

$$2 \leq \delta_m + \delta_s \leq 3 \tag{4.18}$$

Não existem classes onde $\delta_m + \delta_s = 1$, pois corresponderiam a uma rotação em torno do *CIR* (Centro Instantâneo de Rotação) fixo. Também não se admitem classes onde $\delta_m > 2$, e $\delta_s = 2$, para que o robô não apresente uma estrutura singular.

Serão apresentados a seguir exemplos de robôs móveis para ilustrar os tipos de estruturas não singulares. Será centrada a atenção em robôs móveis com três rodas (Campion, 1996).

- Tipo (3,0): Robô omnidirecional com rodas suecas, O robô considerado tem três rodas localizadas nos vértices do chassi, o qual tem a forma de um triângulo equilátero.

| RODAS | α | β | γ | *L* |
|---|---|---|---|---|
| **sueca 1** | π/3 | **0** | **0** | **L** |
| **sueca 2** | π | **0** | **0** | **L** |
| **sueca 3** | 5π/3 | **0** | **0** | **L** |

Suas restrições tomam a forma (4.10) onde:

$$J_1 = [J_{1sw}] = \begin{pmatrix} -\frac{\sqrt{3}}{2} & \frac{1}{2} & L \\ 0 & -1 & L \\ \frac{\sqrt{3}}{2} & \frac{1}{2} & L \end{pmatrix}, \quad J_2 = diag(r)$$

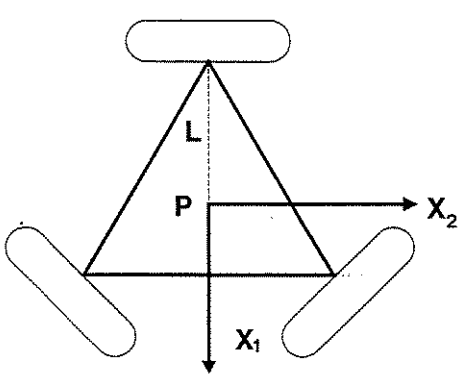



Figura 4.6: Robô Omnidirecional tipo (3,0), três rodas suecas

- Tipo (3,0): Robô omnidirecional com rodas orientadas fora do centro, O robô considerado tem três rodas convencionais orientadas fora do centro, como mostra a figura 4.7. As restrições tomam a forma (4.10) e (4.11) onde:

$$J_1 = \left[J_{1fc}(\beta_{fc})\right] = \begin{pmatrix} -sin\beta_{fc1} & \cos\beta_{fc1} & L\cos\beta_{fc1} \\ sin\beta_{fc2} & -\cos\beta_{fc2} & L\cos\beta_{fc2} \\ \cos\beta_{fc3} & sin\beta_{fc3} & L\cos\beta_{fc3} \end{pmatrix}, \; J_2 = diag(r)$$

$$C_1 = \left[C_{1fc}(\beta_{fc})\right] = \begin{pmatrix} \cos\beta_{fc1} & sin\beta_{fc1} & d + Lsin\beta_{fc1} \\ -\cos\beta_{fc2} & -sin\beta_{fc2} & d + Lsin\beta_{fc2} \\ sin\beta_{fc3} & -\cos\beta_{fc3} & d + Lsin\beta_{fc3} \end{pmatrix}, \; C_2 = \left[C_{2fc}\right] = diag(d)$$

| RODAS | α | β | L |
|---|---|---|---|
| 1fc | 0 | - | L |
| 2fc | π | - | L |
| 3fc | 3π/2 | - | L |

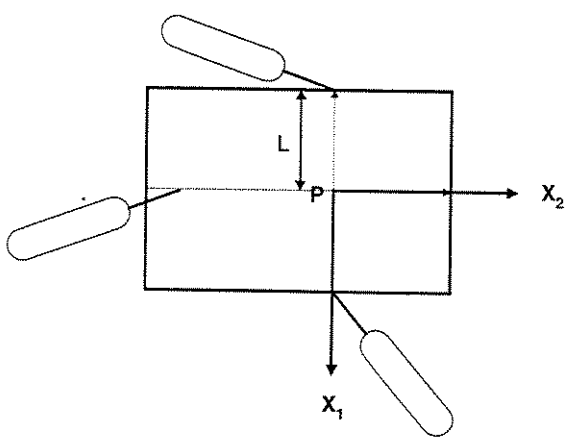



Figura 4.7: Robô omnidirecional, tipo (3,0), com três rodas orientadas fora do centro.

- Tipo (2,0): Robô com duas rodas convencionais fixas no mesmo eixo e uma roda convencional orientada fora do centro, como mostra a figura 4.8. As restrições tomam a forma (4.10) e (4.11) onde:

$$J_1 = \begin{bmatrix} J_{1f} \\ J_{1fc} \end{bmatrix} = \begin{pmatrix} 0 & 1 & L \\ 0 & -1 & L \\ \cos\beta_{fc3} & sin\beta_{fc3} & L\cos\beta_{fc3} \end{pmatrix}, \; J_2 = diag(r)$$

$$C_1 = \begin{pmatrix} C_{1f} \\ C_{1fc} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ sin\beta_{fc3} & -\cos\beta_{fc3} & d + Lsin\beta_{fc3} \end{pmatrix}, \; C_2 = \begin{pmatrix} 0 \\ 0 \\ d \end{pmatrix}$$

| RODAS | α | β | L |
|---|---|---|---|
| 1f | 0 | 0 | L |
| 2f | π | 0 | L |
| 3fc | 3π/2 | - | L |

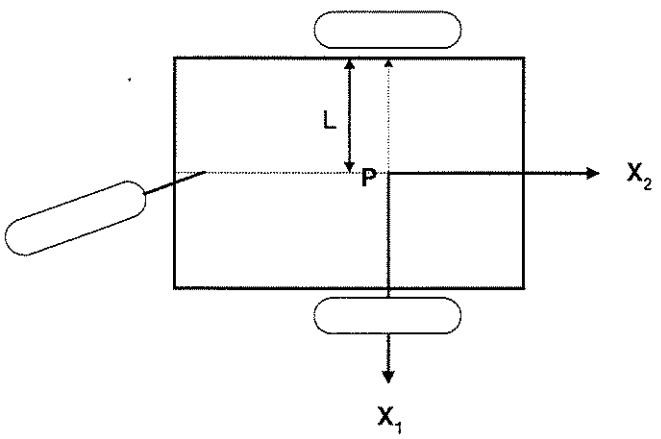



Figura 4.8: Robô tipo (2,0).

- Tipo (2,1): Robô com uma roda orientada no centro e duas rodas orientadas fora do centro, como mostra a figura 4.9. As restrições tomam a forma (4.10) e (4.11) onde:

$$J_1 = \begin{bmatrix} J_{1c}(\beta_{c1}) \\ J_{1fc}(\beta_{fc2}, \beta_{fc3}) \end{bmatrix} = \begin{pmatrix} sin\beta_{c1} & \cos\beta_{c1} & l\cos\beta_{c1} \\ sin\left(\beta_{fc2} + \frac{\pi}{4}\right) & -\cos\left(\beta_{fc2} + \frac{\pi}{4}\right) & l\cos\beta_{fc2} \\ -sin\left(\beta_{fc3} - \frac{\pi}{4}\right) & sin\left(\beta_{fc3} - \frac{\pi}{4}\right) & l\cos\beta_{fc3} \end{pmatrix}, \; J_2 = diag(r)$$

$$C_1 = \begin{bmatrix} C_{1c}(\beta_{c1}) \\ C_{1fc}(\beta_{fc2}, \beta_{fc3}) \end{bmatrix} = \begin{pmatrix} \cos\beta_{c1} & sin\beta_{c1} & lsin\beta_{c1} \\ \cos\left(\beta_{fc2} + \frac{\pi}{4}\right) & -sin\left(\beta_{fc2} + \frac{\pi}{4}\right) & d + lsin\beta_{fc2} \\ -\cos\left(\beta_{fc3} - \frac{\pi}{4}\right) & sin\left(\beta_{fc3} - \frac{\pi}{4}\right) & d + lsin\beta_{fc3} \end{pmatrix},$$

$$C_2 = \begin{pmatrix} 0 \\ diag(d) \end{pmatrix}$$

| RODAS | α | β | L |
|---|---|---|---|
| 1c | 0 | - | 0 |
| 2fc | 5π/4 | - | $\sqrt{2}$ L |
| 3fc | 7π/4 | - | $\sqrt{2}$ L |

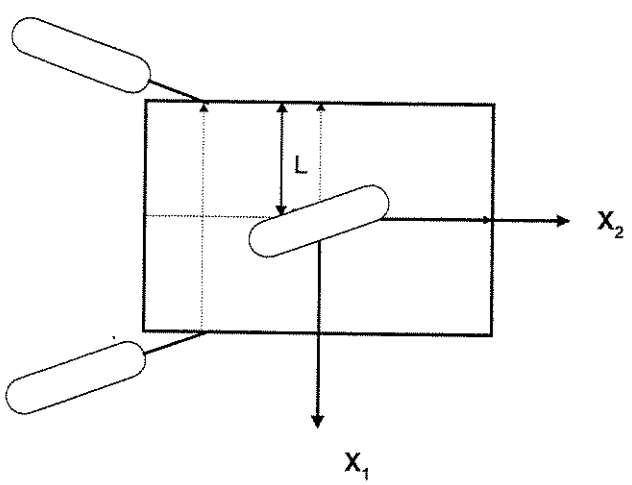



Figura 4.9: Robô tipo (2,1).

- Tipo (1,1): Robô com duas rodas fixas no mesmo eixo e uma roda orientada no centro, por exemplo triciclos para crianças, como mostra a figura 4.10. As restrições tomam a forma (4.10) e (4.11) onde:

$$J_1 = \begin{bmatrix} J_{1f} \\ J_{1c} \end{bmatrix} = \begin{pmatrix} 0 & 1 & L \\ 0 & -1 & L \\ \cos\beta_{c3} & sin\beta_{c3} & L\cos\beta_{c3} \end{pmatrix}, \; J_2 = diag(r)$$

$$C_1 = \begin{pmatrix} C_{1f} \\ C_{1c} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ sin\beta_{c3} & -\cos\beta_{c3} & d + Lsin\beta_{c3} \end{pmatrix}, \; C_2 = 0$$

| RODAS | α | β | L |
|---|---|---|---|
| 1f | 0 | 0 | L |
| 2f | π | 0 | L |
| 3c | 3π/2 | - | L |

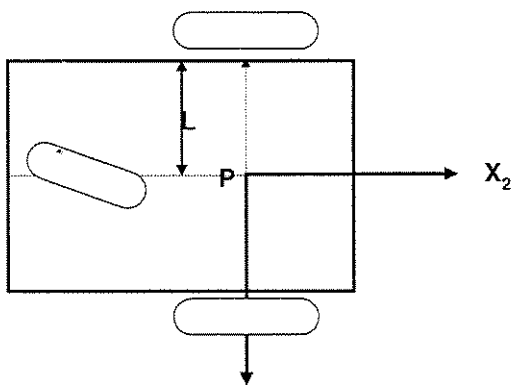



Figura 4.10: Robô tipo (1,1).

- Tipo (1,2): Robô com duas rodas orientadas no centro e uma roda orientada fora do centro, como mostra a figura 4.11. As restrições tomam a forma (4.10) e (4.11) onde:

$$J_1 = \begin{pmatrix} J_{1c}(\beta_{c1}, \beta_{c2}) \\ J_{1fc}(\beta_{fc3}) \end{pmatrix} = \begin{pmatrix} -sin\beta_{c1} & \cos\beta_{c1} & L\cos\beta_{c1} \\ sin\beta_{c2} & -\cos\beta_{c2} & L\cos\beta_{c2} \\ \cos\beta_{fc3} & sin\beta_{fc3} & L\cos\beta_{fc3} \end{pmatrix}, \; J_2 = diag(r)$$

$$C_1 = \begin{pmatrix} C_{1c}(\beta_{c1}, \beta_{c2}) \\ C_{1fc}(\beta_{fc3}) \end{pmatrix} = \begin{pmatrix} \cos\beta_{c1} & sin\beta_{c1} & Lsin\beta_{c1} \\ -\cos\beta_{c2} & -sin\beta_{c2} & Lsin\beta_{c2} \\ sin\beta_{fc3} & -\cos\beta_{fc3} & Lsin\beta_{fc3} \end{pmatrix}, \; C_2 = \begin{pmatrix} 0 \\ C_{2fc} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ d \end{pmatrix}$$

| RODAS | α | β | L |
|---|---|---|---|
| 1fc | 0 | - | L |
| 2fc | π | - | L |
| 3fc | 3π/2 | - | L |

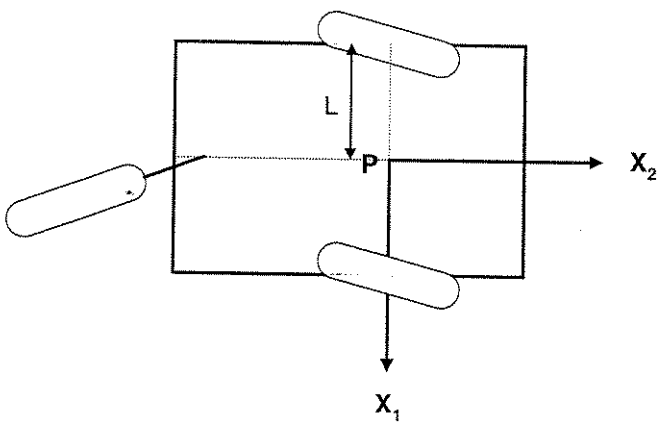



Figura 4.11: Robô tipo (1,2).

### 4.2.4 Modelo Cinemático de Postura

São modelos matemáticos de estado que dão uma descrição geral dos robôs móveis, permitindo que se discuta as propriedades de manobra do robô. Possibilitam uma análise do robô dentro do contexto de sistemas não-holonômicos.

Permite inferir sobre conceitos de Mobilidade, Dirigibilidade e Manobrabilidade do robô móvel

Como é descrito pela equação (4.15), $R(\theta)\dot{\xi} \in N\left[C_1^*(\beta_c)\right]$, conclui-se que para qualquer instante de tempo $t$, existe um vetor variante no tempo $\eta(t)$, entrada de controle, tal que:

$$\dot{\xi} = R^T(\theta)\Sigma(\beta_c)\eta \tag{4.19}$$

A dimensão do vetor $\eta(t)$ é o grau de mobilidade $\delta_m$ do robô. Quando o robô não tem rodas orientáveis ao centro ( $\delta_s = 0$ ou $N_c = 0$ ) a matriz $\Sigma$ é constante e:

$$\dot{\xi} = R^T(\theta)\Sigma\eta \tag{4.20}$$

De outra forma o modelo pode ser descrito pela equação 3.19 com:

$$\dot{\beta}_c = \zeta \tag{4.21}$$

As expressões acima podem ser consideradas como a representação no espaço de estados do sistema (chamado modelo cinemático de postura), onde a coordenada de postura $\xi$ e (possivelmente) a coordenada angular $\beta_c$ são variáveis de estado, enquanto $\eta$ e $\zeta$ são consideradas como as entradas de controle.

Esta interpretação deve entretanto ser tomada com cuidado. Em um sistema físico as verdadeiras entradas de controle são os torques dos motores, fornecidos pelos motores embarcados: O modelo cinemático do espaço de estados é de fato só um subsistema do modelo dinâmico geral

O modelo cinemático de postura escrito de uma forma compacta fica:

$$\dot{z} = B(z)u$$

Se $N_c = 0$

$$z = \xi;$$
$$B(z) = R^T(\theta)\Sigma$$
$$u = \eta$$

Se $N_c \neq 0$

$$z = \begin{Bmatrix} \xi \\ \beta_c \end{Bmatrix}$$

$$B(z) = \begin{bmatrix} R^T(\theta)\Sigma & 0 \\ 0 & 1 \end{bmatrix}$$

$$u = \begin{Bmatrix} \eta \\ \varsigma \end{Bmatrix}$$

Para cada classe de robô móvel entre as definidas na Tabela 4.3, existe um modelo cinemático de postura diferente. Os robôs da mesma classe podem ser representados pelo mesmo modelo, variando a posição do ponto P e do referencial $(X_1,X_2)$.

Para cada robô móvel com rodas pertencente à mesma classe, o modelo cinemático de postura é genérico. Para um robô móvel com rodas é sempre possível selecionar o ponto de referencia P e a base $(X_1,X_2)$ solidário ao chassi do robô de tal modo que o modelo cinemático de postura tome exatamente a forma correspondente ao tipo de robô dado na Tabela 4.3.

- Para o tipo de robô (3,0) o ponto de referência P e a base $(X_1,X_2)$ podem ser escolhidos arbitrariamente.

- Para o tipo de robô (2,0) o ponto de referência P é escolhido como um ponto do eixo das rodas fixas, com $X_1$ alinhado ao longo deste eixo, ver exemplo da figura 4.8.

- Para o tipo de robô (2,1), selecionamos uma das rodas centradas e P é escolhido como o centro de esta roda, com $(X_1,X_2)$ escolhido arbitrariamente, ver exemplo na figura 4.9.

- Para o tipo de robô (1,1) se seleciona uma das rodas centradas, P é a interseção da linha perpendicular que passa pelo centro da roda centrada selecionada, com o eixo comum das rodas fixas. L e a longitude de esta perpendicular. Ver o exemplo da figura 4.10.

- Para o tipo de robô (1,2) se seleciona duas rodas orientáveis centradas. P é escolhido como o ponto médio da distancia entre os centros destas duas rodas, com $X_I$ alinhado ao longo da linha que junta estes centros: $L$ é a distancia media entre estes centros. Ver o exemplo da figura 4.11.

Tabela 4:3: Modelos cinemáticos de postura genéricos para robôs móveis com rodas.

| **TIPO** | $z$ | $\Sigma(\beta_c)$ *ou* $\Sigma$ | **Modelo cinemático de postura** $\dot{z} = B(z)u$ |
|---|---|---|---|
| **(3,0)** | $x$<br>$y$<br>$\theta$ | **Matriz Identidade** | $\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}\cos\theta & -sin\theta & 0\\ sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}\eta_1\\ \eta_2\\ \eta_3\end{pmatrix}$ |
| **(2,0)** | $x$<br>$y$<br>$\theta$ | $\begin{pmatrix}0 & 0\\ 1 & 0\\ 0 & 1\end{pmatrix}$ | $\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}-sin\theta & 0\\ \cos\theta & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}\eta_1\\ \eta_2\end{pmatrix}$ |
| **(2,1)** | $x$<br>$y$<br>$\theta$<br>$\beta_{c1}$ | $\begin{pmatrix}-sin\beta_{c1} & 0\\ \cos\beta_{c1} & 0\\ 0 & 1\end{pmatrix}$ | $\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\\ \dot{\beta}_{c1}\end{pmatrix} = \begin{pmatrix}-sin(\theta+\beta_{c1}) & 0 & 0\\ \cos(\theta+\beta_{c1}) & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}\eta_1\\ \eta_2\\ \zeta_1\end{pmatrix}$ |
| **(1,1)** | $x$<br>$y$<br>$\theta$<br>$\beta_{c3}$ | $\begin{pmatrix}0\\ Lsin\beta_{c3}\\ \cos\beta_{c3}\end{pmatrix}$ | $\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\\ \dot{\beta}_{c3}\end{pmatrix} = \begin{pmatrix}-Lsin\theta sin\beta_{c3} & 0\\ L\cos\theta sin\beta_{c3} & 0\\ \cos\beta_{c3} & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}\eta_1\\ \zeta_1\end{pmatrix}$ |
| **(1,2)** | $x$<br>$y$<br>$\theta$<br>$\beta_{c1}$<br>$\beta_{c2}$ | $\begin{pmatrix}-2Lsin\beta_{c1}sin\beta_{c2}\\ Lsin(\beta_{c1}+\beta_{c2})\\ sin(\beta_{c2}-\beta_{c1})\end{pmatrix}$ | $\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\\ \dot{\beta}_{c1}\\ \dot{\beta}_{c2}\end{pmatrix} = \begin{pmatrix}-L(sin\beta_{c1}sin(\theta+\beta_{c2}) & 0 & 0\\ +sin\beta_{c2}sin(\theta+\beta_{c1})) & & \\ L(sin\beta_{c1}\cos(\theta+\beta_{c2}) & 0 & 0\\ +sin\beta_{c2}\cos(\theta+\beta_{c1})) & & \\ sin(\beta_{c2}-\beta_{c1}) & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}\eta_1\\ \zeta_1\\ \zeta_2\end{pmatrix}$ |

O *grau de mobilidade* $\delta_m$ é o primeiro critério de manobrabilidade, mas não expressa a totalidade dos graus de liberdade que podem ser manipulados pelas entradas de controle $\eta$ e $\zeta$, mas somente as diretamente manipuladas pelas entradas $\eta$.

O *grau de manobrabilidade* de um robô móvel pode ser definido como o número de graus de liberdade que podem ser diretamente manipulados com as entradas de controle $\eta$ e $\zeta$, instantaneamente, a partir da configuração do robô em um dado momento, sem redirecionar qualquer de suas rodas, e está definido pela seguinte equação:

$$\delta_M = \delta_m + \delta_s$$

com o grau de dirigibilidade $\delta_s$ adicionando os graus de liberdade manipulados pela entrada $\zeta$.

A manobrabilidade de um Robô móvel a rodas depende de $\delta_M$, mas também do modo como está integrado $(\delta_m, \delta_s)$. Portanto dois índices são necessários para caraterizar esta manobrabilidade: $\delta_M$ e $\delta_m$ ou equivalentemente $\delta_m$ e $\delta_s$. Dois robôs com o mesmo $\delta_M$ mas com diferente $\delta_m$ não são equivalentes.

### 4.2.5 Modelo Cinemático de Configuração

O modelo cinemático referente à postura, que se viu no item anterior, não envolve as velocidades angulares e de rotação $\dot{\beta}_{fc}$ e $\dot{\varphi}$. O modelo ciemático de configuração relaciona tais variáveis de estado.

Das equações (4.10) e (4.11) temos:

$$\dot{\varphi} = -J_2^{-1} J_1(\beta_c, \beta_{fc}) R(\theta) \dot{\xi}$$

$$\dot{\beta}_{fc} = -C_{2fc}^{-1} C_{fc1}(\beta_{fc}) R(\theta) \dot{\xi}$$

Combinando estas equações com o modelo cinemático de postura (4.19) e (4.21), as equações de estado para $\dot{\beta}_{fc}$ e $\dot{\varphi}$ são:

$$\dot{\varphi} = E(\beta_c, \beta_{fc})\Sigma(\beta_c)\eta \tag{4.22}$$

$$\dot{\beta}_{fc} = D(\beta_{fc})\Sigma(\beta_c)\eta \tag{4.23}$$

Onde:

$$D(\beta_{fc}) = -C_{2fc}^{-1}C_{1fc}(\beta_{fc})$$

$$E(\beta_c, \beta_{fc}) = -J_2^{-1}J_1(\beta_c, \beta_{fc})$$

Observe-se também que essas matrizes satisfazem as seguintes equações:

$$J_1(\beta_c, \beta_{fc}) + J_2E(\beta_c, \beta_{fc}) = 0 \tag{4.24}$$

$$C_{1fc}(\beta_{fc}) + C_{2fc}D(\beta_{fc}) = 0 \tag{4.25}$$

O modelo cinemático de configuração do robô se escreve a partir da definição de $q$ como vetor de coordenadas de configuração

$$q = \begin{Bmatrix} \xi \\ \beta_c \\ \beta_{fc} \\ \varphi \end{Bmatrix}$$

A evolução das coordenadas de configuração pode ser descrita pela seguinte equação compacta, resultado das equações (4.19), (4.21), (4.22) e (4.23), chamado de modelo cinemático de configuração (Campion, 1996):

$$\dot{q} = S(q)u \tag{4.26}$$

Onde

$$S(q) = \begin{bmatrix} R^T(\theta)\Sigma(\beta_C) & 0 \\ 0 & I \\ D(\beta_c)\Sigma(\beta_c) & 0 \\ E(\beta_c, \beta_{fc})\Sigma(\beta_c) & 0 \end{bmatrix}, \text{ e } u = \begin{Bmatrix} \eta \\ \zeta \end{Bmatrix} \tag{4.27}$$

## 4.3 O Modelo do Robô RAM-I

Como o principal objetivo deste trabalho é deixar uma plataforma de testes para outras pesquisas, será apresentado nesta seção o modelamento que se aplica ao robô construído, o que poderia ser útil para futuros usuários.

### 4.3.1 Descrição do Robô RAM-I

O RAM-I é um robô móvel de base retangular de 0,18 m por 0,16 m. Possui duas rodas convencionais fixas no chassi com eixos colineares, paralelos ao lado menor, com raio de 0,035 m, e uma roda orientada fora do centro como apoio, sem nenhum controle, a qual não será tomada em conta para o seu modelo cinemático. O controle de direção é realizado através da variação da velocidade de rotação das rodas fixas (motrizes). A figura 4.12 esquematiza o chassi do robô.

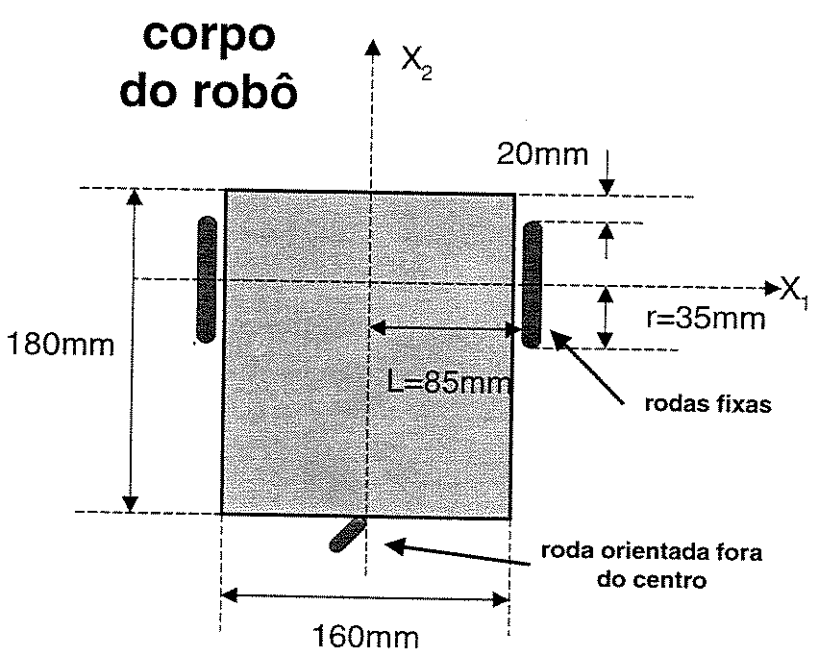



Figura 4.12: O chassi do robô **RAM-I**.

### 4.3.2 Modelo Cinemático de Postura do RAM-I.

Considera-se o robô como livre, movendo-se numa superfície lisa, e um ponto fixo **P** dentro do robô. Pode-se modelar este robô como se fosse um objeto retangular movendo-se no espaço, como mostrado na figura 4.13. Num espaço livre pode-se direcionar o robô a uma posição com uma orientação. Portanto o espaço de configuração tem três dimensões, duas de translação e uma de rotação. Esta configuração foi mostrada no começo por $\xi(x,y,\theta)$, onde $x$ e $y$ são as coordenadas do ponto médio **P,** entre as duas rodas motrizes, na base inercial e $\theta \in [0,2\pi]$ é o ângulo entre o eixo $x$ da base inercial e o eixo principal do robô, variante durante o movimento. Consequentemente, o movimento é restringido pela equação (Latombe, 1996):

$$\dot{y}\cos\theta - \dot{x}sin\theta = 0$$

Esta restrição não-holonômica diz que o robô somente pode mover-se na direção normal ao eixo das rodas motrizes, satisfazendo as condições de rolamento puro e não escorregamento (Fierro, 1998).

Da Tabela 4.3, o modelo cinemático de postura para os robôs moveis pertencentes á classe (2,0) tipo "char" à qual pertence o **RAM-I** é:

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{Bmatrix} = \begin{bmatrix} -sin\theta & 0 \\ \cos\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} \eta_1 \\ \eta_2 \end{Bmatrix}$$

Esta é a equação cinemática de movimento em termos de velocidade linear e velocidade angular, onde $|\eta_1| \leq V_{max}$ e $|\eta_2| \leq W_{max}$. $V_{max}$, $W_{max}$ são as velocidades maximas linear e ângular do ponto **P** que se deseja controlar.

Uma restrição não-holonômica é uma equação não integrável que envolve os parâmetros de configuração e as suas derivadas (parâmetros de velocidade). Tais restrições não reduzem o espaço de configuração atingível pelo robô, mas reduzem a dimensão do espaço dos possíveis

movimentos diferencias (por exemplo o espaço das direções da velocidade) em alguma dada configuração (Latombe, 1996).

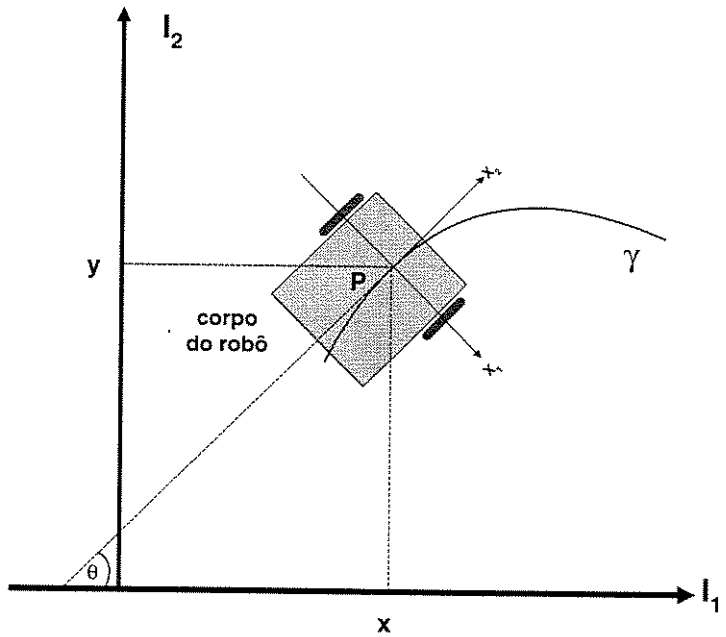



Figura 4.13: Coordenadas do robô móvel **RAM-I**.

### 4.3.3 Modelo Cinemático de Configuração do RAM-I

Varias configurações de motorização para robôs do tipo (2,0), "char", que usam dois motores, são possíveis para este tipo de robô, (Campion, 1996):

1. Dois motores de rotação nas rodas 1 e 2;
2. Um motor para a orientação da roda 3, e um para a rotação da roda 2 (ou 1), providenciando $d > L\frac{\sqrt{2}}{2}$;
3. Dois motores (orientação e rotação) na roda 3 orientada fora do centro, providenciando $d < L$.

onde: as rodas 1 e 2 são rodas convencionais fixas, a roda 3 é uma roda convencional orientada fora do centro, $L$ é a distância média (ponto **P**) entre as rodas convencionais fixas, $d$ é a distância

entre o centro da roda convencional orientada fora do centro e o eixo vertical que passa pelo ponto de união do conjunto com o chassi (figura 4.3)

Dado que no **RAM-I** se adotou uma configuração de motorização com dois motores de passo, os quais controlam as rodas fixas independentemente, e não se exerce nenhum controle na roda orientada fora do centro, esta roda segue a rota determinada pelas rodas fixas, comportando-se como uma roda passiva (escrava). Pode-se assumir que o movimento desta roda pode ser ignorado na dinâmica do robô móvel (Yang, 1999). Esta roda não será levada em conta para o modelamento do robô.

Tabela 4.4: Parâmetros do modelo tipo "Char" do robô **RAM-I**

| RODAS | L | $\alpha$ | R | $\beta$ | $\varphi$ |
|---|---|---|---|---|---|
| **Roda 1** | **L** | **0** | **r** | **0** | $\varphi_1$ |
| **Roda 2** | **L** | $\pi$ | **r** | **0** | $\varphi_2$ |

onde $\alpha$ , r, l, são constantes; $\beta$ e $\varphi$ são variáveis com o tempo; L= 0.085m e r= 0.035m.

As restrições cinemáticas têm a forma (4.10) e (4.11)

$$J_1 = \begin{Bmatrix} J_{1f} \\ J_{1fc} \end{Bmatrix} = \begin{bmatrix} 0 & 1 & L \\ 0 & -1 & L \end{bmatrix}$$

$$J_2 = \begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix}$$

$$C_1 = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$$

$$C_2 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

Temos o modelo cinemático de configuração genérico, equações (4.26) e (4.27):

$$\dot{q} = S(q)u$$

$$S(q) = \begin{bmatrix} R^T(\theta)\Sigma(\beta_C) & 0 \\ 0 & I \\ D(\beta_c)\Sigma(\beta_c) & 0 \\ E(\beta_c, \beta_{fc})\Sigma(\beta_c) & 0 \end{bmatrix}, \text{ e } \quad u = \begin{Bmatrix} \eta \\ \varsigma \end{Bmatrix}$$

O vetor de estado $q$ é definido como:

$$q = \begin{Bmatrix} \xi \\ \beta_c \\ \beta_{fc} \\ \varphi \end{Bmatrix}$$

Como não será considerada a roda de apoio, as expressões são reduzidas para:

$$\begin{Bmatrix} \xi \\ \varphi \end{Bmatrix} = \begin{bmatrix} R^T(\theta)\Sigma(\beta_c) & 0 \\ 0 & I \\ E\Sigma & 0 \end{bmatrix} \begin{Bmatrix} \eta_1 \\ \eta_2 \end{Bmatrix} \tag{4.28}$$

onde :

$$E = J_2^{-1} J_1 = \begin{bmatrix} -1/r & 0 \\ 0 & -1/r \end{bmatrix} \begin{bmatrix} 0 & 1 & -L \\ 0 & -1 & L \end{bmatrix} = \begin{bmatrix} 0 & -1/r & -L/r \\ 0 & 1/r & -L/r \end{bmatrix}$$

$$E\Sigma = \begin{bmatrix} 0 & -1/r & -L/r \\ 0 & 1/r & -L/r \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -1/r & -L/r \\ 1/r & -L/r \end{bmatrix}$$

Agora se tem

$$\dot{q} = S(q)u$$

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{Bmatrix} = \begin{bmatrix} -sin\theta & 0 \\ \cos\theta & 0 \\ 0 & 1 \\ -1/r & -L/r \\ 1/r & -L/r \end{bmatrix} \begin{Bmatrix} \eta_1 \\ \eta_2 \end{Bmatrix} \tag{4.29}$$

onde $\eta_1$ é a velocidade longitudinal e $\eta_2$ é a velocidade de rotação do robô em relação ao referencial inercial ; $\dot{\varphi}_1$ e $\dot{\varphi}_2$ são as velocidades de rotação das rodas.

É apresentado um outro modelo cinemático de configuração para o robô (2,0) que foi obtido pela parametrização do movimento do robô em relação à trajetória realizada. Este modelo é escrito em relação à base móvel e diferente do convencional onde a referência é o sistema inercial (Victorino, 1998).

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{Bmatrix} = \begin{bmatrix} -1 & y \\ 0 & -(c+x) \\ 0 & 1 \\ \lambda^{-1} \end{bmatrix} \begin{Bmatrix} \eta_1 \\ \eta_2 \end{Bmatrix} \tag{4.30}$$

onde $c$ é a distância de um ponto fixo no chassi do robô, que se pretende controlar, à origem do sistema de referencia móvel solidário ao robô.

Neste modelo, o vetor de entradas é relacionado às velocidades das rodas da seguinte forma:

$$\begin{Bmatrix} \eta_1 \\ \eta_2 \end{Bmatrix} = \lambda \begin{Bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{Bmatrix}$$

$$\lambda = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{2L} & -\frac{r}{2L} \end{bmatrix}$$

Admitindo a matriz $\lambda$ como não singular, pode-se trabalhar com qualquer parâmetro $(\eta_1, \eta_2)$ ou $(\dot{\varphi}_1, \dot{\varphi}_2)$ como entradas de controle.

Usando a mesma matriz $\lambda$, uma outra forma de se escrever o modelo cinemático de configuração se dá quando as velocidades das rodas $(\dot{\varphi}_1, \dot{\varphi}_2)$ são consideradas como entradas de controle.

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{Bmatrix} = \begin{bmatrix} -\frac{r}{2}\sin\theta & -\frac{r}{2}\sin\theta \\ \frac{r}{2}\cos\theta & \frac{r}{2}\cos\theta \\ \frac{r}{2L} & -\frac{r}{2L} \end{bmatrix} \begin{Bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{Bmatrix} \tag{4.31}$$

Existe uma relação finita entre $\theta$ e a posição angular das rodas $\dot{\varphi}_1$ e $\dot{\varphi}_2$:

$$\theta = \frac{r}{2L}(\varphi_1 - \varphi_2) + \text{Const.}$$

A derivada desta relação inicial é dita holonômica. As duas primeiras relações são não-holonômicas, pois nenhuma relação finita pode ser obtida através delas, elas não sendo integráveis.

Como comentário sobre os vários tipo de restrições importantes para veículos que o projetista em robótica móvel terá de lidar (Triggs, 1993) se tem:

- **Restrições de configuração**: Restringem a possível configuração estática do veículo, delimitando regiões de trabalho (obstáculos, objetos físicos).

- **Restrições dinâmicas**: Restringem os valores das quantidades diferencias tais como velocidade, aceleração e curvatura do trajeto.

- **Restrições do tipo Integral**: Limitam valores integrais tais como: tempo de execução, consumo de recursos para uma determinada tarefa.

- **Restrições de Planejamento**: São significativos quando há interação com objetos em movimento no ambiente de trabalho do veículo.

- **Restrições Diferenciais** : São igualdades com restrições de velocidade ou de manobras, por exemplo o estacionamento de um veículo. As principais restrições diferenciáveis são holonômicas ou integráveis e não-holonômicas ou não integráveis.

Os robôs móveis com rodas pertencem, em geral, aos sistemas mecânicos ditos não holonômicos ou não integráveis. É possível saber "apriori" quando uma restrição ou um sistema de equações é não integrável. Uma forma de verificar a integrabilidade de sistemas é denominado de teorema de Frobenius, (Victorino, 1998).

***"O sistema*** $\dot{x} = B(x)u$ ***é completamente integrável se:***

***Posto (B2....Bn)=Posto (B1 B2 ...Bn[Bi,Bj])"***

onde *Bi, i=1...n,* são colunas de $B(x)$ e $[Bi,Bj] = \nabla BjBi - \nabla BiBj$, operador matemático chamado de colchete de Lie ("Lie Bracket").

O entendimento do teorema ficará claro com uma aplicação ao modelo de estado do Robô móvel, onde se constatará que o mesmo não é integrável e assim não-holonômico.

O modelo cinemático de postura do robô **RAM-I** é:

$$\dot{z} = B(z)u$$

$$z = \begin{Bmatrix} x \\ y \\ \theta \end{Bmatrix}; B(z) = \begin{bmatrix} -\sin\theta & 0 \\ \cos\theta & 0 \\ 0 & 1 \end{bmatrix}; u = \begin{Bmatrix} \eta_1 \\ \eta_2 \end{Bmatrix}$$

$$B_1 = \begin{Bmatrix} -sin\theta \\ \cos\theta \\ 0 \end{Bmatrix}; B_2 = \begin{Bmatrix} 0 \\ 0 \\ 1 \end{Bmatrix}$$

$$B_3 = [B_1, B_2] = (\nabla B_2)B_1 - (\nabla B_1)B_2$$

$$B_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} -sin\theta \\ \cos\theta \\ 0 \end{Bmatrix} - \begin{bmatrix} 0 & 0 & -\cos\theta \\ 0 & 0 & -sin\theta \\ 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} 0 \\ 0 \\ 1 \end{Bmatrix}$$

$$B_3 = 0 - \begin{Bmatrix} -\cos\theta \\ sin\theta \\ 0 \end{Bmatrix} = \begin{Bmatrix} \cos\theta \\ -sin\theta \\ 0 \end{Bmatrix}$$

Então

$$Posto\left\{\begin{bmatrix} -sin\theta & 0 & \cos\theta \\ \cos\theta & 0 & -sin\theta \\ 1 & 1 & 0 \end{bmatrix}\right\} = 3 > Posto\left\{\begin{bmatrix} -sin\theta & 0 \\ \cos\theta & 0 \\ 0 & 1 \end{bmatrix}\right\}$$

Portanto, pode-se concluir pelo modelo de Frobenius que o modelo de estado do Robô **RAM-I** é não-holonômico.

# Capítulo 5

# Resultados e Conclusões

Neste capítulo será apresentado o robô **RAM-I** resultado do trabalho desenvolvido durante esta dissertação de mestrado. Ele servirá como plataforma de testes dos futuros trabalhos na área da robótica móvel autônoma. Será apresentado o robô em algumas configurações especiais aqui denominadas tarefas. Neste capítulo também se faz um resumo das conclusões que foram optidas no transcurso do desenvolvimento do **RAM-I**. Finalmente se apresentarão algumas propostas para trabalhos futuros.

O robô **RAM-I** foi construído no *Departamento de Mecânica Computacional (DMC)* da Faculdade de Engenharia Mecânica da Universidade Estadual de Campinas.

## 5.1 Robô Autônomo Móvel Inteligente: RAM-I

O RAM-I é um robô autônomo móvel inteligente desenvolvido utilizando a Arquitetura *Subsumption* e o principio percepção-ação, mostrado na figura 5.1. Está conformado por uma base retangular de 0,18 m por 0,16 m, possui duas rodas convencionais fixadas ao chassi com os eixos colineares, paralelos ao lado menor, com raio de 0,035 m, e uma roda orientada fora do centro como apoio. O mecanismo de motorização está conformado por dois motores de passo TAMAGAWA SEIKI acoplados diretamente às rodas fixas. O controle de direção é realizado através da variação da velocidade de rotação das rodas fixas (motrizes) e/ou a inversão de giro dos motores.

Este robô foi construído utilizando dois módulos baseados no microcontrolador PIC16F84, o ***Sb Step Control***, controlador dos motores, e o ***Sb 9902_2*** com o programa do sistema de controle total do robô. A este módulo estão conectados quatro sensores de toque, dois na frente (detecção de obstáculos) e os outros dois na parte inferior (figura 5.3) na procura de depressões e assim sentir a eventual "perda" de chão e se proteger contra quedas.

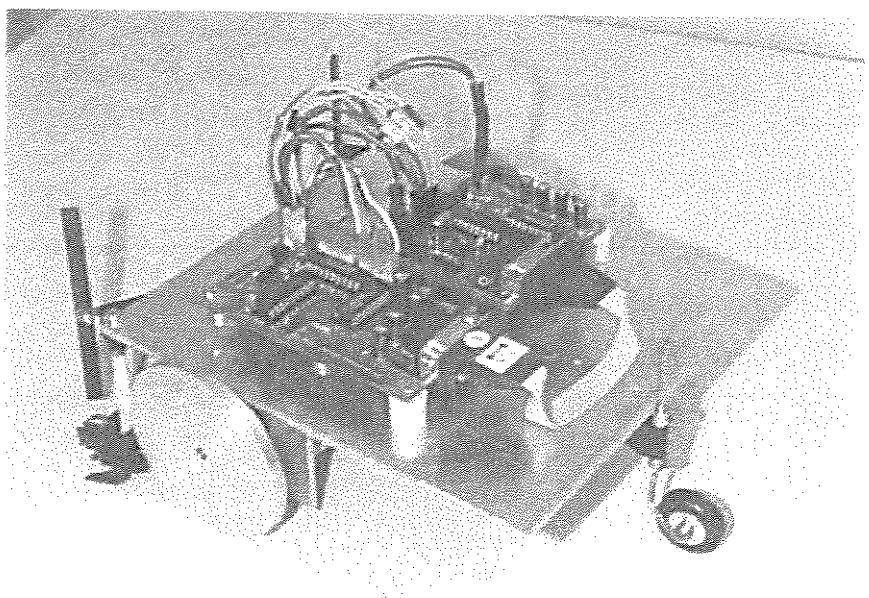


Figura 5.1: Robô **RAM-I** .

O sistema de alimentação esta conformado por um "pack" de baterias de 12V. recarregáveis, o qual fornece ao robô a autonomia energética necessária à realização das suas tarefas programadas.

### 5.1.1 Características do Robô RAM-I

- Dimensões físicas:
  - Altura: 10 cm;
  - Base: 17.5 cm por 18 cm;
  - Peso: 2 Kg;

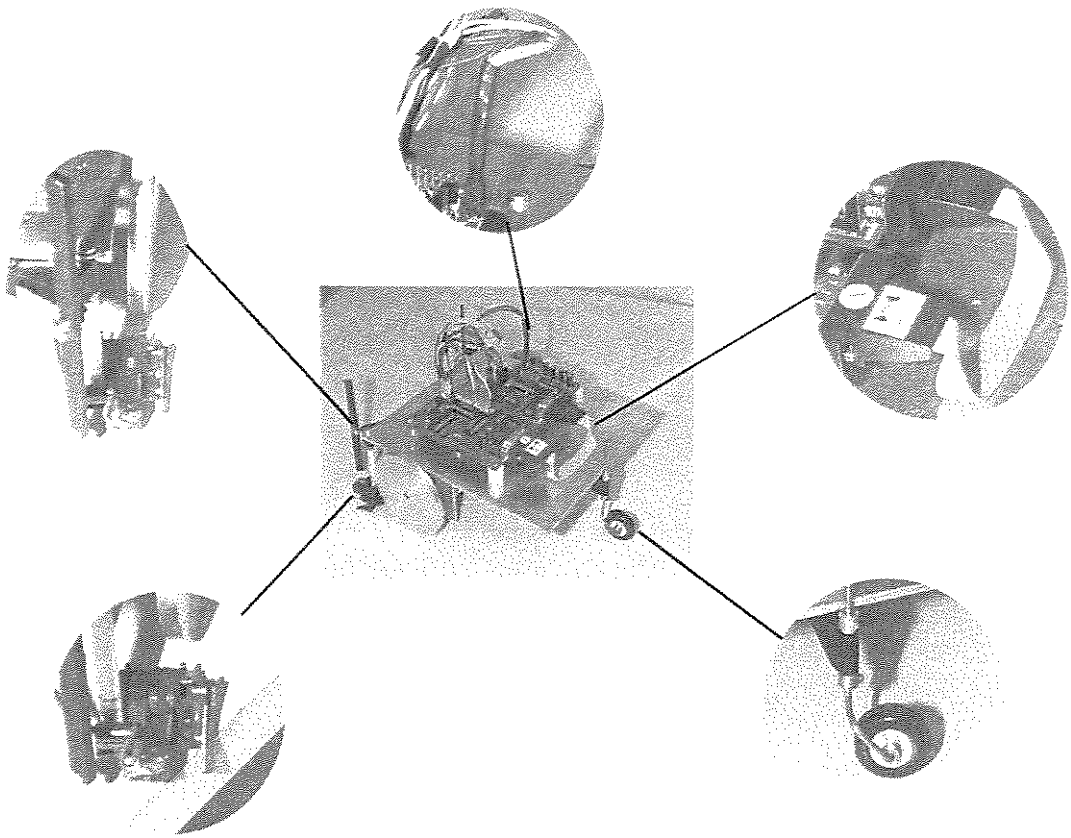


Figura 5.2: Detalhes do Robô **RAM-I** .

- Cinemática:
  - Direção: através do controle da velocidade de rotação das rodas;
  - Redução: 1:1;
  - Velocidade 3.3 m/min
  - Motores de passo: 3,6° por passo (100 ppv);

- Energia:
  - Baterias: "pack" de baterias recarregáveis de NiCd, 38 Wh/Kg, 12 V, 500mA;
  - Autonomia: 1 hora em funcionamento continuo;

- Sensores:
  - Sensor de toque para obstáculos: duas chaves liga desliga na forma de "bigode";
  - Sensor de depressões: dois sensores do tipo "fim de curso" com molas;

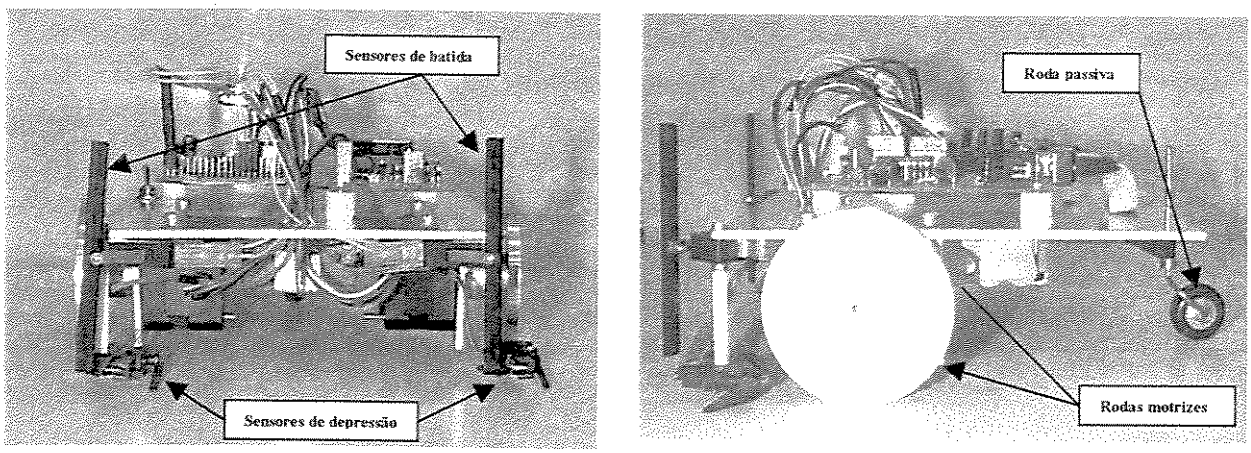



Figura 5.3: Vistas de frente e de lado do Robô **RAM-I**: motores, sensores de toque e rodas.

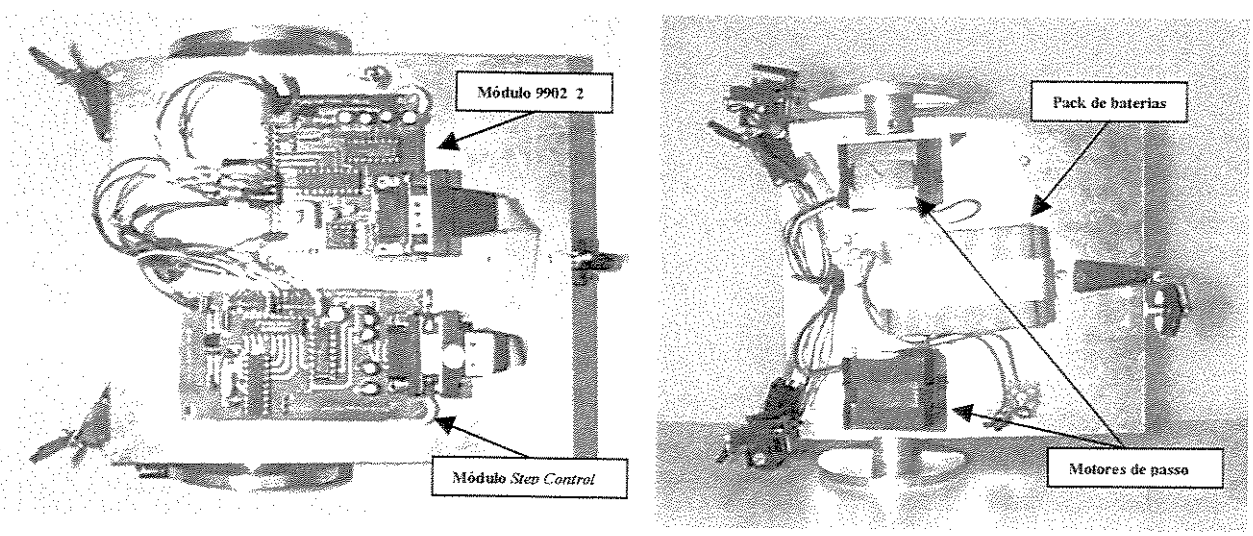



Figura 5.4: Vistas superior e inferior do Robô **RAM-I**: módulos, motores, sensores, bateria.

- Controladores:
  - Baseados no microcontrolador PIC16F84 da MICROCHIP;
  - Programação em "Assembler" da MICROCHIP;
  - Comunicação entre os módulos via cabo (interface RS232-C);
  - Controle em "tempo real";
  - Módulo ***Sb 9902_2*** com capacidade para receber até 13 entradas de sinal dos sensores externos.
  - Restrições na capacidade da memória RAM do microcontrolador PIC16F84.

Como exemplo, foi programada uma tarefa simples, descrita a seguir. Esta tarefa poderá ser alterada de acordo com as necessidades de futuros trabalhos.

A arquitetura de controle foi idealizada em base à Arquitetura *Subsumption* atendendo os requerimentos das tarefas implementadas VAGAR e EVITAR, descritas no capitulo 3 como camadas implementadas no robô **RAM-I**.

## 5.2 Testes e Resultados com o Robô RAM-I

Para verificar a capacidade do robô de servir como uma plataforma de testes e assim cumprir as tarefas implementadas, um conjunto de testes foram propostos:

1. Vagar num ambiente estruturado.
2. Desviar obstáculos que impedem o livre transcurso do robô.
3. Afastar-se do perigo de perder o chão.

### 5.2.1 Vagar Livremente num Ambiente Estruturado e desviar Obstáculos

Nestes primeiros testes o robô **RAM-I** foi colocado no centro de um ambiente determinado e deixado vagar livremente. O ambiente é apropriado ao uso do robô, formado por uma superfície lisa, sem grandes depressões e com alguns obstáculos.

Nesta caso a camada VAGAR comanda o robô fazendo com que ele se desloque pelo ambiente em trajetórias retas. Nesta estado ele não é afetado por eventuais perturbações que alterem a sua direção. O robô permanece neste estado até que a camada de nível mais elevado, EVITAR, seja ativada pela presença de um obstáculo.

Quando a camada EVITAR é ativada, o robô identifica se o sensor que a ativou é um sensor de depressão (neste ambiente só existem obstáculos fixos). Depois desta verificação, o robô verifica se o obstáculo está na frente-esquerda ou na frente-direita, para determinar a manobra adequada. Se a batida for pela direita, o robô recua e vira para esquerda, se for pela esquerda o robô recua e vira para direita, diminuindo assim a possibilidade de bater de novo com o mesmo obstáculo. A manobra de recuar e virar pode ser compreendido melhor pelos comandos: inverter motores, diminuir velocidade da roda do lado da batida (esquerda ou direita), após certo tempo, continuar com o trajeto.

Verificou-se também nestes testes, a "reação" do robô quando os dois sensores de obstáculos da frente são ativados ao mesmo tempo. O robô age de acordo com a programação das camadas implementadas. A manobra de nível mais elevado no robô, é a manobra de recuar e virar para direita. Sendo ativados os dois sensores ao mesmo tempo, o robô sempre virará e recuará primeiro pela direita até a direção indicada pela estratégia implementada para o sensor de obstáculo.

### 5.2.2 Afastar-se do Perigo, Perder o Chão.

Os robôs autônomos móveis tem que se desenvolver em ambientes diversos, desde laboratórios de pesquisa a terrenos inóspitos em planetas distantes, e assim enfrentar perigos iminentes apresentados pelo próprio ambiente. Tais perigos põem em risco a "vida" do robô, perigos como por exemplo a "perda" do chão ou a proximidade de alguma borda que indique o termino da superfície. Um exemplo típico deste perigo está na queda do robô pelas escadas de um laboratório.

O robô **RAM-I** está provido de dois sensores chamados de "sensores de depressão" localizados na frente das duas rodas motrizes, de modo a evitar a que o robô possa cair numa depressão.

Neste segundo teste o robô **RAM-I** foi colocado no centro de um ambiente estruturado e deixado vagar livremente. O ambiente é um ambiente apropriado ao uso do robô, com uma superfície lisa (mesa).

Para a realização deste teste foi preciso montar o ambiente simulando eventuais obstáculos (paredes) de um lado e do outro bordas que indicam o fim da superfície do ambiente.

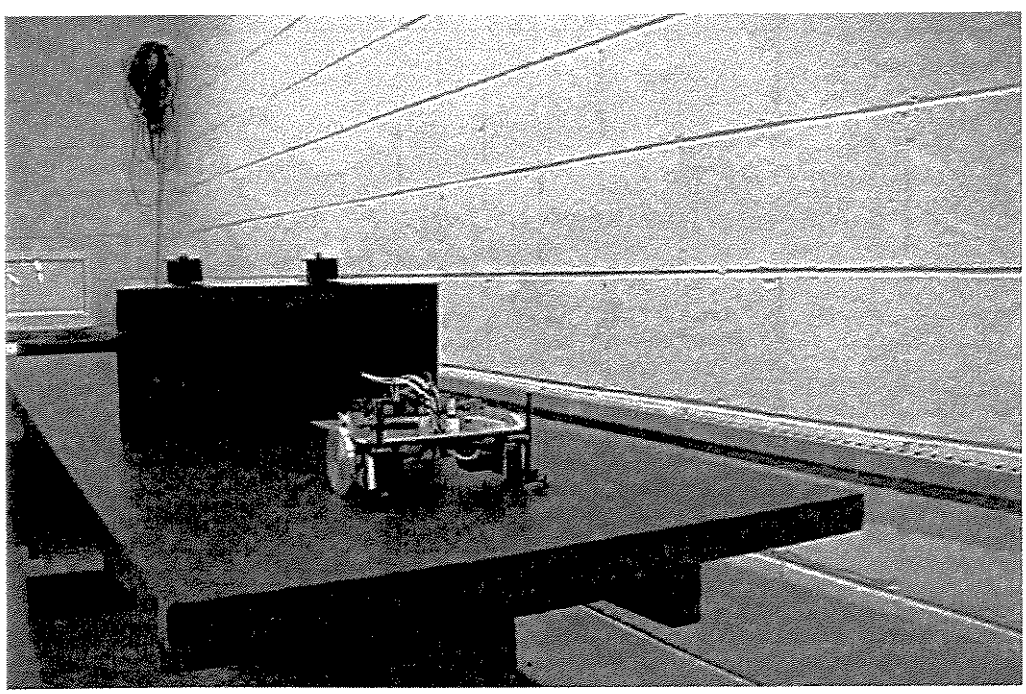


Figura 5.5: Robô autônomo móvel **RAM-I** no ambiente de testes.

Uma vez ligado, a camada VAGAR comanda o robô fazendo com que ele se desloque pelo ambiente em trajetórias retas. Neste estado, ele não é afetado por eventuais perturbações que alterem a sua direção. O robô permanece neste estado até que a camada de nível mais elevado, EVITAR, seja ativada pela presença de um obstáculo ou uma depressão.

Quando a camada EVITAR é ativada, o robô identifica se o sensor que fio ativado é um sensor de depressão. O robô depois verifica se a depressão está na frente-esquerda ou na frente-direita, e determina a manobra adequada. Se a batida for à direita, o robô recua e vira para esquerda, se for à esquerda o robô recua e vira para direita, diminuindo assim a possibilidade de se encontrar de novo com a mesma "depressão". A manobra de recuar e virar é composta de: inverter motores, diminuir velocidade da roda do lado do sensor ativado (esquerda ou direita), após algum tempo, continuar com o trajeto. A direção que o robô toma é diferente da que tomaria se fosse simplesmente um obstáculo na frente. Esta nova direção procura afastar o robô da "depressão" por esta ser um perigo à segurança do robô.

Verificou-se nestes testes, a "reação" do robô quando os dois sensores de depressão são ativados ao mesmo tempo. O robô age de acordo com a programação das camadas implementadas. A manobra de nível mais elevado no robô, é a manobra de recuar e virar para direita. Sendo ativados os dois sensores ao mesmo tempo o robô sempre virará e recuará primeiro para a direita.

Verificou-se também nestes testes, a "reação" do robô quando dois tipos de sensores diferentes são ativados ao mesmo tempo, sensores de depressão e sensores de obstáculos. O robô age de acordo com a programação das camadas implementadas. A manobra de nível mais elevado no robô, é a manobra ativada pelos sensores de depressão. O robô então procura se afastar do perigo iminente de não ter chão na frente. Sendo ativados os sensores de depressão e de obstáculo ao mesmo tempo, o robô sempre virará e recuará na direção indicada pelo sensor de depressão.

## 5.3 Conclusões

O objetivo proposto neste trabalho foi alcançado: implementar um robô autônomo móvel inteligente utilizando a Arquitetura de controle *Subsumption,* para ser usado como base de testes em futuros trabalhos na área da robótica móvel autônoma.

Foi estudado o problema da validação de experimentos com robôs autônomos móveis, sendo proposto a implementação destes para uma "verdadeira" validação dos resultados.

A implementação do projeto é simples e flexível, de modo a permitir ao robô **RAM-I** se converter numa plataforma de testes para novos componentes: motores, sensores, controladores, baterias.

Para a implementação escolheu-se dois módulos (módulos de "Hardware") baseados no microcontrolador PIC16F84, um deles para o controle dos motores de passo e o outro como "cérebro". A comunicação entre eles é feita através de um cabo (interface RS232-C).

No robô **RAM-I,** dois comportamentos bem definidos foram implementados: EVITAR e VAGAR. A ordem nas camadas foi estabelecida como sendo a de mais alta prioridade a EVITAR, para cada camada um predicado de aplicabilidade foi definido utilizando como base a configuração dos sensores.

Pode se definir a Arquitetura *Subsumption,* como sendo a divisão do sistema de controle do robô em vários processos pequenos paralelos e concorrentes chamados de camadas, os quais simulam certo tipo de "comportamento". Eles avaliam os dados dos sensores para assim controlar os parâmetros de saída. Esta Arquitetura demostrou ser uma abordagem prática e eficiente para se construir robôs autônomos móveis inteligentes.

Existem estratégias para escrever "software" que possa responder a estímulos externos em tempo real. O robô **RAM-I**, possui um controle em tempo real utilizando um método onde

"loops" de "software" continuamente verificam as entradas do microcontrolador (sensoriamento por sondeio), chamado de "polling".

O robô autônomo móvel **RAM-I**, por sua configuração, pertence aos Sistemas Mecânicos ditos de não-holonômicos, caraterizados pelas restrições cinemáticas não integráveis. Logo, eles não podem ser estabilizados com as mesmas leis de controle com realimentação de estados.

## 5.4 Trabalhos futuros

No robô **RAM-I** poderiam ser introduzidas muitas modificações, sendo este o seu objetivo. A seguir, serão apresentadas algumas das propostas sugeridas para futuros trabalhos:

- Implementação de um novo módulo para o "cérebro" com um outro microcontrolador PIC da MICROCHIP que facilite a aquisição de dados, maior capacidade de memória, número maior de portas disponíveis.

- Utilização de outros tipos de sensores: ultrasom, infravermelho, ou mesmo uma câmera digital como auxílio à teleoperação, ou mesmo para a visualização do funcionamento do sistema.

- Implementação de um sistema de comunicação via rádio.

- Utilização de outros tipos de motores como por exemplo motor ultra-sónico.

- Uso de outras Arquiteturas de Controle baseados em diferentes teorias, como por exempto redes neurais, lógica Fuzzy.

- Implementação de um "software" (Linguagem C) para a interação entre um computador e o robô autônomo móvel, no caso de testes experimentais que requeiram o uso de maior capacidade de memória. Esta interação poderia ser obtida via cabo, radio, ou por sinal infravermelho.

- Desenvolvimento de um programa de simulação para o robô autônomo móvel, visando a simulação dos comportamentos, navegação, controle de trajetória.

- Acoplar novos dispositivos, como por exemplo braços manipuladores, sobre a plataforma do **RAM-I.**

- Construção de um novo protótipo utilizando tecnologia das micromáquinas. Este seria um mini robô.

# Referências Bibliográficas


Aihara, C. K. *Projeto e Implantação de Plataforma Didática Aplicada ao Ensino e Pesquisa em Automação*. Campinas - SP: Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2000, 92 p. Dissertação (Mestrado)

Andrade, José F. A. de; Mendeleck, A.; Zampieri, Douglas E. Geração de Trajetórias para Robôs Móveis Autônomos Usando Redes Neurais Artificiais. In: V Congresso Brasileiro de Redes Neurais, Rio de Janeiro - RJ, Brasil. *Anais...* abril 2001. p. 571-576

Bernardes, L.H. Microcontrolador National COP8. Saber Eletrônica, São Paulo, v. 1, n. 308, p 4-11, setembro 1998.

Bradshaw, A. Sensors for Mobile Robots. *Measurement + Control,* v. 23, p. 48-52, march 1990

Braga, N.C. Motores de passo. Saber Eletrônica, São Paulo, v. 1, n. 323, p 4-6, dezembro 1999.

Braga, N.C. Controlando motores de passo. Saber Eletrônica, São Paulo, v. 1, n. 314, p 48-53, março 1999.

Brooks, R. A. Intelligence Without Reason. In: IJCAI-91, Morgan Kaufmann, San Mateo CA USA. *Proceedings...*, p. 569-595, 1991.

Brooks, R. A. A Robust Layered Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation,* v. 2, n.1, p. 14-23; Março 1986.

Brooks, R. A., Intelligence Without Representation, *Artificial Intelligence Journal*, p. 139 – 159, 1991.

Campion, G., et al. Structural properiesanda classification of kinematic and dynamic models of wheeled mobile robots, *IEEE Trans. robotic and automation,* v. 12, n.1, p. 47-62; 1996.

Costa, S. E. *Estudo e Implementação de Sistema de Controle de direção para Veículo Auto-guiado.* Campinas - SP: Faculdade de Engenharia Elétrica, UNICAMP, 1993. 125p. Dissertação (Mestrado)

Craig, J. *Introduction to robotics: mechanics and control.* U.S.A: Addison-Weslwy Publishing Company, 1986. 450 p.

Doughty, S. *Mechanics of machines.* New York: J. Wiley, 1988. 467p.

Escola Virtual Solbet v.2.1, PIC Básico: Solbet Ltda. Treinamento Profissional, [2000]. 1 CD-ROM. Windows 95-98 NT.

Ferreira, Edson de P. *Robótica Básica.* Rio de Janeiro: Versão Preliminar Publicada para a V Escola Brasileiro - Argentina de Informática, janeiro 1991.

Fierro, R; Lewis, F. L. Control of a Nonholonomic Mobile Robot Using Neural networks. *IEEE Transactions on Neural Networks*, v. 9, n. 4, july 1998.

Flynn, Anita M. et al. Cirujia del Mañana: Micromotores y Microrobots. **Cirujia del Mañana,** 1996. (Tomorrow´s Surgery, January 29, 1995) Disponível na Internet. http://wwwfacmed.unam.mix/bmnd/testocompleto/laparosc/flynn.pdf 13 de dezembro de 1999.

Gat, E. Towards principled experimental study of autonomous mobile robots. *Kluwer Academic Publishers,* v. 2, p. 179-189, Março 1995.

GSI, DIN - UEM. *Grupo de Sistemas Inteligentes. Departamento de Informática - Universidade Estadual de Maringá.* Fevereiro de 2001. Material descrito na homepage. Disponível na Internet: http://www.din.uem.br/~ia/1024x768/index.html

Han, Min-Hong; Rhee, Sang-Yong. Navigation Control for a Mobile Robot. *Journal of Robotic Systems,* v. 11, n. 3, p. 169-179, 1994

Jetto, Leopoldo; Longhi, Sauro; Venturini, Giuseppe. Development and Experimental Validation of na Adaptive Extended Kalman Filter for the Localization of Mobile Robot. *IEEE Transactions on Robotics and Automation,* v. 15, n. 2, p. 219-228, April 1999

Jones, Joseph L.; Flynn, Anita M. *Mobile Robots – Inspiration to Implementation.* Wellesley, Massachusetts: A. K. Peters Ltd., 1993, 345 p.

Laumond, J. P. *Robot Motion Planning and Control.* Lectures Notes in Control and Information Sciences 229, Springer, 1998, 343 p. Disponível na Internet: http://www.laas.fr/~jpl/book.html

Latombe, Jean-Claude. *Robot Motion Planning.* Massachusetts: Kluwer Academic Publishers, 1996, 651 p.

Machado, J. A. T. ; Rodrigues, C.M.B.; Silva, F. M. Sistemas Robóticos de Locomoção Quadrúpede e Hexápode. *Revista Robótica e Automatização,* n. 30, fevereiro 1998.

Maes, P., Brooks R. A. Learning to Coordinate Behaviors, *AAAI Journal*, p. 796--802 Agosto 1990.

Marto, A. G. *Motores Ultra-sônicos: Princípios de Funcionamento e Características.* Guaratinguetá: Faculdade de Engenharia de Guaratinguetá, Universidade Estadual Paulista, 1997. 130 p. Dissertação (Mestrado)

Maxim Integrated Products, *MAXIM +5V-Powered, Multichannel RS-232 Drivers/Receivers*, May 2000, Disponível em: www.electromelec.co.uk/max232cpe.htm. Acesso em: 7 Julho 2001.

Microchip Technology Inc., *PIC16F84A*. September 1998, Disponível em http://www.microchip.com/0/lit/pline/picmicro/families/16f8x/devices/16f84a/index.htm Acesso em: Julho 2000.

Microchip Technology Inc., *MPLAB, IDE, SIMULATOR, EDITOR*: User`s Guie, U.S.A., 2000. 284 p.

Muir, P. F, *Modelling and control of wheeled mobile robots.* Pittsburgh, PA, USA: Department of Electric and Computer Engineering and Robotics Institute, Carnegie Mellon University, 1998. 335 p. These (PhD.)

National Semiconductor Corporation, *ADC0831/ADC0832/ADC0834/ADC0838 8-Bit A/D Converters with Multiplexer Options.* August 1999, Disponível em: http://national.com/pf/AD/ADC0831.html. Acesso em: 8 Julho 2001.

National Semiconductor Corporation, *LM78LXX Series 3-Terminal Positive Regulators.* November 2000, Disponível em: http://national.com/pf/LM/LM78L05.html. Acesso em: 8 Julho 2001.

Yang, Jung-Min; Kim, Jong-Hwan. Slidind Mode Control for Trajectory Tracking of Nonholonomic Wheeled Mobile Robots. *IEEE Transactions on Robotics and Automation,* v. 15, p. 578-587, 1999

Ohya, Akihisa; Kosaka, Akio; Kak, Avinash. Vision-Based Navigation by a Mobile Robot with Obstacle Avoidance Using Single-Camara Vision and Ultrasonic Sensing. *IEEE Transactions on Robotics and Automation,* v. 14, n. 6, p. 969-978, 1998

Passeto, F. *Desenvolvimento de uma Aplicativo para Simulação e Controle de Manipuladores Robóticos com ênfase em Aplicações Didáticas.* Campinas - SP: Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2000, 102 p. Dissertação (Mestrado)

Rezende, Marcos F. de. *Desenvolvimento de um Robô Móvel Autônomo Inteligente Utilizando a Arquitetura de Assunção.* Uberlândia - MG: Centro de Ciências Exatas e Tecnologia, Universidade Federal de Uberlândia, 1992. 102 p. Dissertação (Mestrado)

Sandia Technology, *A Quarterly Research & Development Journal,* v. 2, n. 2, 21 p. summer 2000.

Segenreich, Solly.; Vargas, Juan. Uma nova Tecnologia para acionamento de motores de passo operando em micropasso. Congresso Nacional de Engenharia Mecânica - CONEM 2000, Natal, Anais do CONEM 2000., 2000, v.CDROM

SGS Thomson Microelectronics. *ULN2801A-ULN2802A-ULN2803A-ULN2804A-ULN2805A*, September 1997, Disponível em: http://us.st.com/stonline/books/pdf/docs/1536.pdf. Acesso em: 8 Julho 2001.

Souza, D. *Desbravando o PIC.* São Paulo: Érica Ltda, 2000, 200 p.

Solbet Microcontroladores e Robótica Ltda. *Sb Microlab V. 2.0*: manual de operações e manutenção, São Paulo, 2000. 16 p.

Solbet Microcontroladores e Robótica Ltda. *Sb StepControl V. 1.0*: manual de operações e manutenção, São Paulo, 2000. 12 p.

Solbet Microcontroladores e Robótica Ltda. *KITs 9901_1 & 9901_2 V. 1.1*: manual de montagem e operação, São Paulo, 2000. 14 p.

Spong, M. *Robot Dynamics and Control.* New York: John Wiley & Sons, 1989. 336 p.

Suárez, Lizet L. *Conhecimento Sensorial - Uma Análise Segundo a Perspectiva da Semiótica Computacional.* Campinas - SP: Faculdade de Engenharia Elétrica e de Computação, UNICAMP, 2000. 144 p. Dissertação (Mestrado)

Takiguchi, J. et al. Study of autonomous mobile system in outdoor enviroment. In: IEEE International Conference On Robotics Andautomation, 3, 1998, Leuvem. *Proceeding...* Leuven IEEE 1998, p. 2416-2421.

Triggs, Bills *Motion Planning for Nonholonomic Vehicles: Na Introduction*, 1993. Disponível na Internet. http://www.inrialpes.fr/movi/people/Triggs/abstracts.html#Triggs:nonhol93 12 de março de 2001.

Thomson Airpax. Product Selection and engineering Guide: catálogo. New York 1995. 41 p.

Tourino, Sérgio; Álvares, Alberto. Desenvolvimento de um robô móvel autônomo teleoperado via internet. Congresso Nacional de Engenharia Mecânica - CONEM 2000, Natal, Anais do CONEM 2000., 2000, v. CDROM

Victorino, A. C. *Controle de Trajetória e estabilização de Robôs Móveis Não-Holonômicos.* Campinas-SP: Faculdade de Engenharia Mecânica, UNICAMP, 1998. Dissertação (Mestrado)

Viera, A. *Desenvolvimento e Implementação de controle de Trajetória Continua em Robô Industrial de Alto Desempenho.* Campinas - SP : Faculdade de Engenharia Elétrica, UNICAMP, 1996. Tese (Doutorado).

# Anexos

## A.1: Programa

```
;*****************************************************************************************
;
;nome:sensafaz10.asm
;
;objetivo: implementar (programar) as tarefas do cérebro
;
;ação: Este programa contem a Arquitetura de Controle implementado no "cérebro" para um robô
;      com dois sensores de toque e dois sensores de depressão na frente.
;      O robô recua e vira para esquerda quando acionado o sensor direito, quando acionado o
;      sensor direito recua e vira para esquerda
;      A estratégia do robô quando acionados os sensores de depressão é de recuar e dar
;      volta para esquerda ou para direita conforme a estratégia para depois voltar a
;      caminhar livremente.
;
;descrição das portas
;RA0:
;RA1:saida RS232-C
;RA2:
;RA3:
;RA4:
;
;RB0:
;RB1:
;RB2:
;RB3:
;RB4: entrada do sensor esquerdo
;RB5: entrada do sensor direito
;RB6: entrada do sensor de depressao esquerdo
;RB7: entrada do sensor de depressao direito
;freqüência do cristal: 10 MHz
;autor: Justo Emilio Alvarez Jácobo
;versão do MPLAB: 3.40.00
;sinal RS232-C, 1200,N,8,1.
;
;*****************************************************************************************
;
;              sensor_depre_esq     sensor_ depre _dir
;
;          sensor_esq                    sensor_dir
;                        ***************
;                      * *             * *
;       roda esq       * *             * *     roda dir
;       (motor0)       * *             * *     (motor1)
;                      * *             * *
;                        *             *
;                        *             *
;                        *             *
;                        *             *
;                        *             *
;                        ***************
```

```
;
;*****************************************************************************************
; configuração do Assembler e microcontrolador
;*****************************************************************************************

                LIST P=PIC16F84 ;seleciona o tipo de controlador
                #include <p16F84.inc> ;definições do processador

;*****************************************************************************************
; configuração da CPU
; CP(ON,OFF)proteção do softwarePWRTE (ON,OFF) temporizador alimentação
; WDT(ON,OFF) temporizador watchdog OSC (HS,LP, RC, XT) seleção oscilador
;*****************************************************************************************

                __CONFIG _CP_OFF  & _WDT_OFF & _PWRTE_ON & _HS_OSC

;*****************************************************************************************
; definições
;*****************************************************************************************

#define BANK0 BCF STATUS,RP0
#define BANK1 BSF STATUS,RP0
#define SAIDA_RS PORTA,1

#define SENSOR_ESQ PORTB,4          ;lado esquerdo do robô
                                    ;flags para informar o estado sensor
                                    ;0 -> livre
                                    ;1 -> obstáculo na frente
#define SENSOR_DIR PORTB,5          ;lado direito do robô
                                    ;flags para informar o estado sensor
                                    ;0 -> livre
                                    ;1 -> obstáculo na frente
#define SENSOR_DEPRE_ESQ PORTB,6    ;frente esquerdo do robo
                                    ;flags para informar o estado sensor
                                    ;0 -> livre
                                    ;1 -> depressao na frente
#define SENSOR_DEPRE_DIR PORTB,7    ;frente direito do robo
                                    ;flags para informar o estado sensor
                                    ;0 -> livre
                                    ;1 -> depressao na frente

RESET       EQU  0X0000 ; ponto de reset da CPU
INT         EQU  0X0004 ; ponto de desvio de interrupção da CPU
VELO_INIC   EQU .100

;#define DEBUG ; válido para simulação
;*****************************************************************************************
; declaração das variáveis
;*****************************************************************************************
      cblock 0x0c
            w_temp
            status_temp
            nIndex
            valorTx
            cont1
            cont2
            cont3
            cont4
            cont5
            cont6
            int_cont1
            int_cont2
            int_cont3
            int_cont4 ;contador usado para incrementar tempo entre as tarefas
            sensores
            temp
            d; lixeira
      endc
;*****************************************************************************************
; declaração das macros
;*****************************************************************************************
```

```
; não há macros neste programa

;*****************************************************************************************
; inicio do código
;*****************************************************************************************

        org RESET
        goto INICIO

;*****************************************************************************************
; inicio da interrupção
;*****************************************************************************************


;           ------------------------------------------------------------
            org INT
;           ------------------------------------------------------------
;           SEM INTERRUPCOES
;           ------------------------------------------------------------
            RETFIE

;*****************************************************************************************
; rotinas e subrotinas
;*****************************************************************************************
;-----------------------------------------------------------------------------------------
;  SUBROTINAS DE INICIALIZACAO DOS MOTORES
;-----------------------------------------------------------------------------------------
; inicializa os motores numa outra velocidade diferente do default dado pelo Step Control
;-----------------------------------------------------------------------------------------
inic_motores    ;default do robô

        movlw  'D'    ;para os motores para enviar as novas velocidades ao mesmo
        call  envia   ;tempo e assim o motor 1 não fique retrasado

        call inic_motor0
        call inic_motor1

        movlw 'F'     ;M0 no sentido antihorário
        call  envia
        movlw 'G'     ;M1 no sentido horário
        call  envia

        movlw 'C'     ;ativa os motores de novo
        call  envia

        return        ;(ultimo return) da função inic_motores

inic_motor0
        movlw  'A'
        call envia2
        return        ;do inic_motor0

inic_motor1
        movlw  'B'
        call envia2
        return        ;do inic_motor1

;-----------------------------------------------------------------------------------------
; subprograma encarregado de enviar a constante da vel_inic (neste caso 10)
;-----------------------------------------------------------------------------------------
envia2
        call    tx        ;transmite o primeiro comando que tenho no W
        clrf    nIndex
loop2   movf    nIndex,W
        call    tabela2
        movwf   temp
        movf    temp,F
        btfsc   STATUS,Z  ;se fim da mensagem
        return            ;se terminou de enviar a mensagem volta @@@
        call    tx
        incf    nIndex,F
```

```
        goto    loop2
;-------------------------------------------------------------------------------------------
tabela2                     ;valor da velocidade default inicial dos motores (neste caso 10)
        addwf PCL,F
        retlw '1'
        retlw '0'
        retlw 0x0d
        retlw 0x0a
        retlw 0

;-------------------------------------------------------------------------------------------
; SUBROTINAS DE GIRO DO ROBO
;-------------------------------------------------------------------------------------------

recua_dir
        movlw  'D'     ;para os motores
        call  envia
        movlw  'E'     ;M0 sentido H (horário)
        call  envia
        movlw  'H'     ;M1 sentido AH (antihorário)
        call  envia
        movlw  'C'     ;liga motores
        call  envia
        call espera12
        movlw  'F'     ;M0 no sentido AH para virar no mesmo eixo a direita
        call  envia
        call espera12
        call inic_motores
        return         ;retorno de recua_dir
;-------------------------------------------------------------------------------------------

recua_esq
        movlw  'D'     ;para os motores
        call  envia
        movlw  'E'     ;M0 sentido H
        call  envia
        movlw  'H'     ;M1 sentido AH
        call  envia
        movlw  'C'     ;liga motores
        call  envia
        call espera12
        movlw  'G'     ;M1 no sentido H  para virar no mesmo eixo a esquerda
        call  envia
        call espera12
        call inic_motores
        return         ;retorno de recua_esq
;-------------------------------------------------------------------------------------------

afasta_por_dir
        movlw  'D'     ;para os motores
        call  envia
        movlw  'E'     ;M0 sentido H
        call  envia
        movlw  'H'     ;M1 sentido AH
        call  envia
        movlw  'C'     ;liga motores
        call  envia
        call espera12
        movlw  'F'     ;M0 no sentido AH  para virar no mesmo eixo a direita
        call  envia
        call  espera12
        call  espera12
        call  inic_motores
        return  ;retorno de afasta_por_dir
;-------------------------------------------------------------------------------------------

afasta_por_esq
        movlw  'D'     ;para os motores
        call  envia
        movlw  'E'     ;M0 sentido H
        call  envia
```

```
        movlw  'H'      ;M1 sentido AH
        call   envia
        movlw  'C'     ;liga motores
        call   envia
        call espera12
        movlw  'G'     ;M1 no sentido H  para virar no mesmo eixo por esquerda
        call   envia
        call   espera12
        call   espera12
        call   inic_motores
        return  ;retorno de afasta_por_esq
;-------------------------------------------------------------------------------

parada
        movlw  'D'     ;para os motores
        call   envia
        movlw  'E'     ;M0 sentido horario
        call   envia
        movlw  'H'     ;M1 sentido antihorario
        call   envia
        movlw  'C'     ;liga os motores
        call   envia
        call   espera12
        call   inic_motores
        return         ; retorno de parada

;-------------------------------------------------------------------------------
; subprograma encarregado de enviar a velocidade de retardo do motor que gira a menor velo
; exemplo: quando este recúe (100), o outro motor estará na velocidade default (10)
; este subprograma não está sendo usado, o robô gira no mesmo eixo
;-------------------------------------------------------------------------------

envia3
        call    tx          ;transmite o primeiro comando que eu tenho no W
        clrf    nIndex
loop3   movf    nIndex,W
        call    tabela3
        movwf   temp
        movf    temp,F
        btfsc   STATUS,Z    ;se fim de mensagem
        return              ;se terminou de enviar a mensagem volta @@@
        call    tx
        incf    nIndex,F
        goto    loop3
;-------------------------------------------------------------------------------
tabela3   ;valor da velocidade do motor que esta recuando a menor velocidade (100)
        addwf PCL,F
        retlw '1'
        retlw '0'
        retlw '0'
        retlw 0x0d
        retlw 0x0a
        retlw 0

;-------------------------------------------------------------------------------
; SUBROTINAS DE ENVIO DE COMANDOS PARA O STEP CONTROL
;-------------------------------------------------------------------------------
envia
        call    tx        ; transmite o primeiro comando que eu tenho no W
        clrf    nIndex
loop1   movf    nIndex,W
        call    tabela1
        movwf   temp
        movf    temp,F
        btfsc   STATUS,Z ;se fim de mesagem
        return           ;se terminou de enviar a mensagem, volta @@@
        call    tx
        incf    nIndex,F
        goto loop1

;-------------------------------------------------------------------------------
```

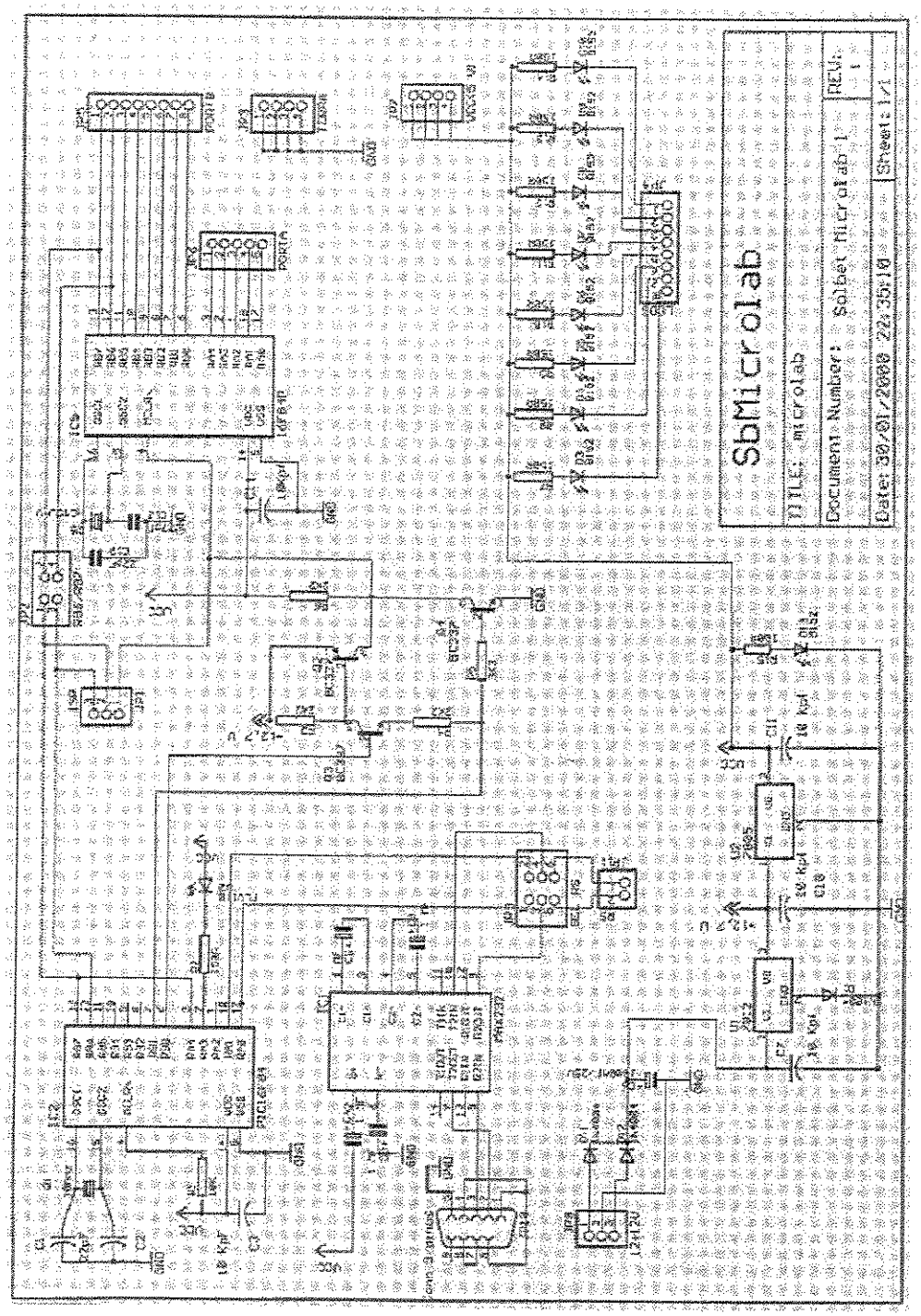
```
tabela1
        addwf PCL,F
        retlw 0x0d
        retlw 0x0a
        retlw 0
;-------------------------------------------------------------------------------------------
tx      movwf   valorTx
        bcf     SAIDA_RS   ;start bit
        call    delayBit
        call    tx1
        call    delayBit
        call    tx1
        call    delayBit
        call    tx1
        call    delayBit
        call    tx1
        call    delayBit
        call    tx1
        call    delayBit
        call    tx1
        call    delayBit
        call    tx1
        call    delayBit
        call    tx1
        call    delayBit
        call    tx1
        call    delayBit
        bsf     SAIDA_RS
        call    delayBit ;stop bit
        return
;-------------------------------------------------------------------------------------------
tx1     rrf     valorTx,F
        btfsc   STATUS,C    ;se carry=1, transmite bit em 1
        goto    bit1
        bcf     SAIDA_RS
        return
bit1    bsf     SAIDA_RS
        return

;-------------------------------------------------------------------------------------------
; ESPERA PELO TEMPO DE UM BIT A 1200 BAUDS
;-------------------------------------------------------------------------------------------

#ifdef DEBUG
delayBit
        return
#else
delayBit
;       Dados para o tempo de amostragem do sinal de transmissão
;       ---------------------------------------------------
        movlw d'13'
        movwf cont1
        movlw d'51'
        movwf cont2
;       ---------------------------------------------------
        movf cont1,W
        movwf int_cont1 ;contador interno
esp1    movf cont2,W
        movwf int_cont2
esp2    decfsz int_cont2,F
        goto esp2
        decfsz int_cont1,F
        goto esp1
        return
#endif

;-------------------------------------------------------------------------------------------
; TEMPOS DE ESPERA  ENTRE AS TAREFAS
;-------------------------------------------------------------------------------------------
espera12
        movlw d'26'
        movwf cont5
esp11   decfsz cont5,F
```

```
        goto esp12
        return
esp12   call espera
        goto esp11
;-------------------------------------------------------------------------------------------
espera
        bsf  SAIDA_RS    ;nível default = 1
        movlw d'255'
        movwf cont3
        movlw d'255'
        movwf cont4
;-------------------------------------------------------------------------------------------
;espera
        movf cont3,W
        movwf int_cont3 ;contador interno
esp3    movf cont4,W
        movwf int_cont4
esp4    decfsz int_cont4,F
        goto esp4
        decfsz int_cont3,F
        goto esp3
        return

;*******************************************************************************************
; INICIO DO PROGRAMA
;*******************************************************************************************
INICIO
        BANK1
        movlw   B'11111111'  ; porta RB4, programada como  sensor
        movwf   (TRISB & 0x07F)
;       7 -> 1  RB7 ENTRADA DO SENSOR DE DEPRESSAO DIREITO
;       6 -> 1  RB6 ENTRADA DO SENSOR DE DEPRESSAO ESQUERDO
;       5 -> 1  RB5 ENTRADA DO SENSOR DIREITO
;       4 -> 1  RB4 ENTRADA DO SENSOR ESQUERDO
;       3 -> 1  RB3 CONFIGURADO PARA ENTRADA
;       2 -> 1  RB2 CONFIGURADO PARA ENTRADA
;       1 -> 1  RB1 CONFIGURADO PARA ENTRADA
;       0 -> 1  RB0 CONFIGURADO PARA ENTRADA
        movlw   B'11111101' ; programa porta RA1 programada como  saida
        movwf   (TRISA & 0x07F)
;       7 -> 1
;       6 -> 1
;       5 -> 1
;       4 -> 1
;       3 -> 1
;       2 -> 1
;       1 -> 0  RA1 CONFIGURADO PARA SAIDA
;       0 -> 1
        BANK0
;*******************************************************************************************
; INICIALIZAÇÃO DAS VARIÁVEIS
;*******************************************************************************************
; nada
;*******************************************************************************************
; ROTINA PRINCIPAL
;*******************************************************************************************

main
        bsf   SAIDA_RS  ;nivel default = 1 saida da comunicacao serial
        call inic_motores
;       movlw 'i'
;       call envia
       ;-----------------------------------------------------------------------------------
       ;LEITURA DOS SENSORES UM A UM
       ;-----------------------------------------------------------------------------------

main1
        movf PORTB,w
        movwf sensores
        btfss SENSOR_DEPRE_ESQ ;interruptor pressionado (NF)?
        goto sim_depre_esq ; sim esta =0 (depressão na esquerda, cuidado!)
```

```
        btfss SENSOR_DEPRE_DIR;interruptor pressionado (NF)?
        goto sim_depre_dir ; sim esta =0 (depressão na direita, cuidado!)
        btfsc SENSOR_ESQ   ;interruptor pressionado (NF)?
        goto sim_obs_esq   ; sim esta =1 (obstáculo)
        btfsc SENSOR_DIR   ; não esta =0 interruptor pressionado (NF)? pula se e 0
        goto sim_obs_dir   ; sim esta =1 (obstáculo)

goto main1

;-------------------------------------------------------------------------------------------------

simesta_esq
        call recua_dir
        goto main1

simesta_dir
        call recua_esq
        goto main1

sim_depre_esq
        call afasta_por_dir
;       movlw  'D'
;       call envia
;       movlw  'I'
;       call envia
        goto main1

sim_depre_dir
        call afasta_por_esq
;       movlw  'E'
;       call envia
;       movlw  'S'
;       call envia
        goto main1
;-------------------------------------------------------------------------------------------------
        END
;*************************************************************************************************
```

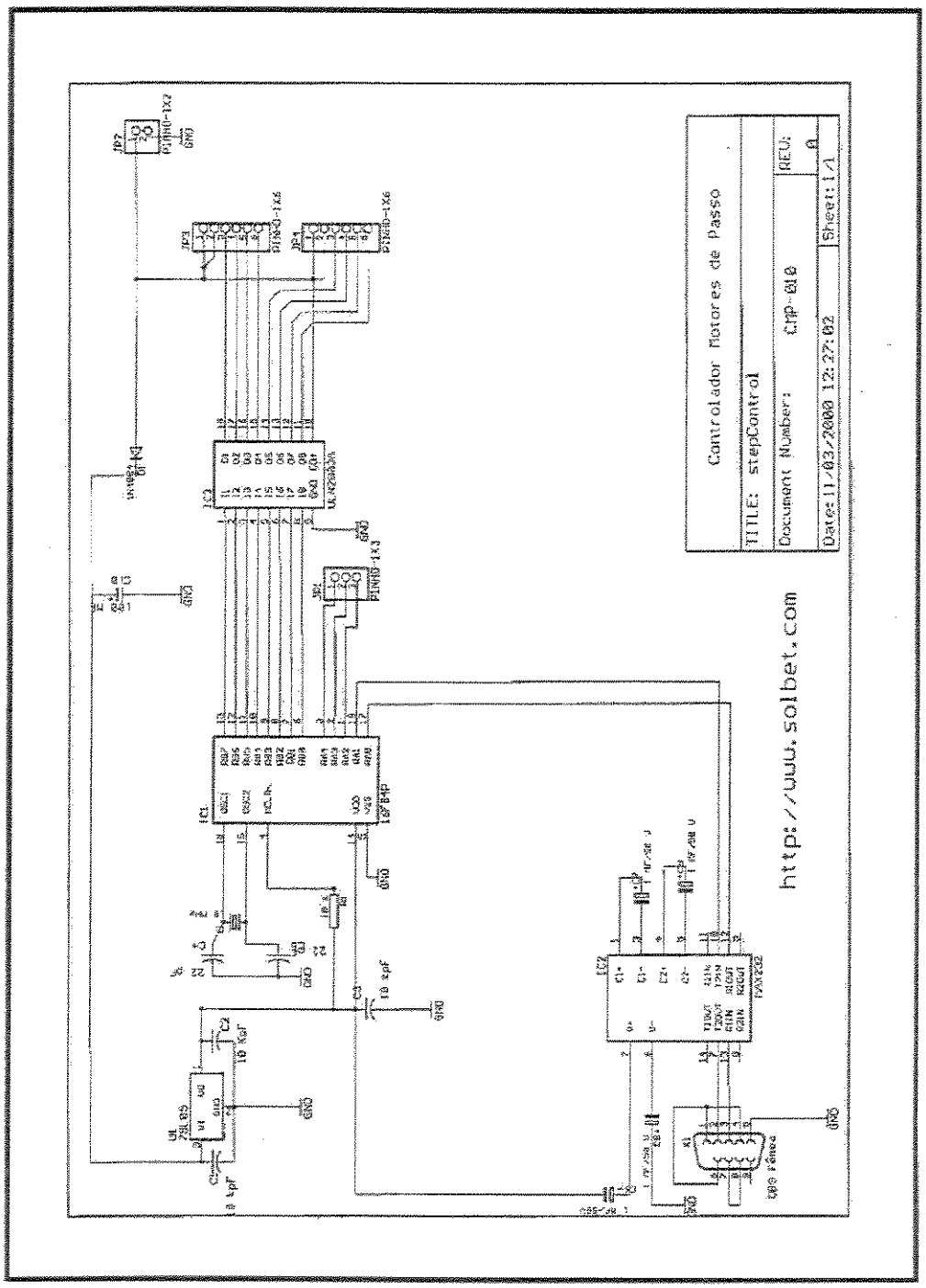
# A.2: Diagrama eletrônico do *SbMicrolab*

Cortesia da Solbet Microcontroladores e Robótica http://www.solbet.com



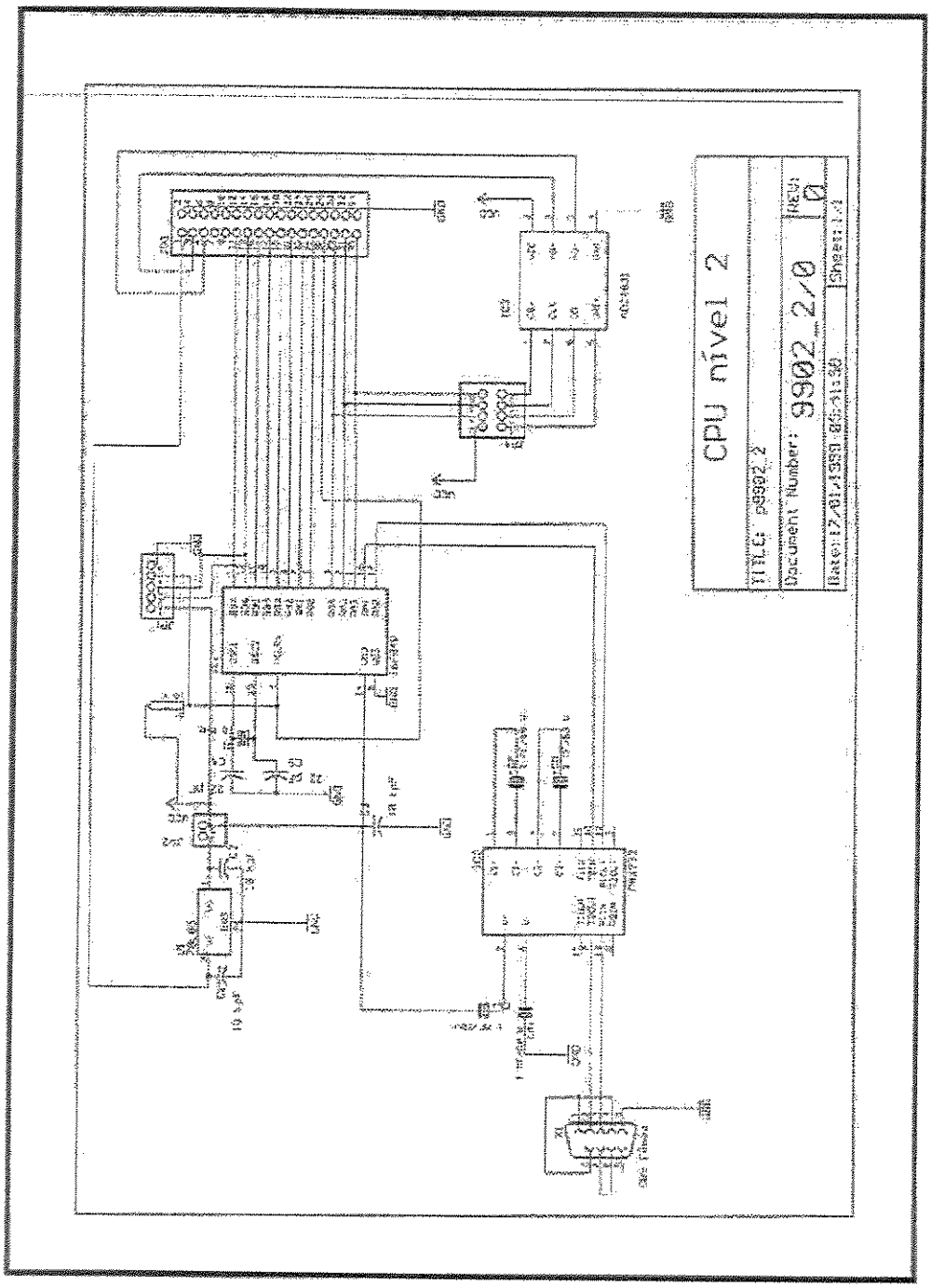
# A.3: Diagrama eletrônico do *Sb Step Control*

Cortesia da Solbet Microcontroladores e Robótica http://www.solbet.com



## A.4: Diagrama eletrônico do Módulo *9902_2*

Cortesia da Solbet Microcontroladores e Robótica http://www.solbet.com CGC00.946.876/0001-25 Insc. Estadual: 244.559.200.110



# A.5: Portas de Comunicação PC *Sb Fast Card*

Cortesia da Solbet Microcontroladores e Robótica http://www.solbet.com

**Sb Fast Card** ™

PORTAS DE COMUNICAÇÃO PC

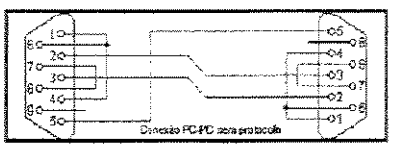
Conector Saída Serial DB25- PC

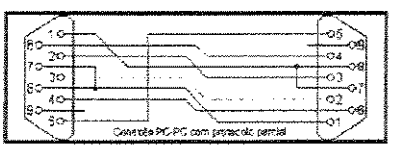
| Pino | Função | Sentido |
|---|---|---|
| Pino 1 | Terra proteção | Entrada |
| Pino 2 | TX (dado transmitido) | Saída |
| Pino 3 | RX(dado recebido) | Entrada |
| Pino 4 | RTS (Request To Send) | Saída |
| Pino 5 | CTS (Clear To Send) | Entrada |
| Pino 6 | DSR (Data Set Ready) | Entrada |
| Pino 7 | Terra do sinal | |
| Pino 8 | DCD (detetor de sinal de linha) | Entrada |
| Pino 20 | DTR (Data Terminal Ready) | Saída |
| Pino 22 | Indicador de chamada | Entrada |

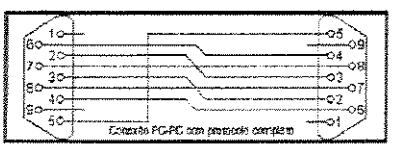
Conector Saída Serial DB9- PC

| | | |
|---|---|---|
| Pino 1 | DCD (detetor de sinal de linha) | Entrada |
| Pino 2 | RX (dado recebido) | Entrada |
| Pino 3 | TX (dado transmitido) | Saída |
| Pino 4 | DTR (Data Termina. Ready) | Saída |
| Pino 5 | GND (terra do sinal) | |
| Pino 6 | DSR (Data Set Ready) | Entrada |
| Pino 7 | RTS (Request To Send) | Saída |
| Pino 8 | CTS (Clear To Send) | Entrada |
| Pino 9 | Indicador de chamada | Entrada |

O conector do PC é do tipo macho

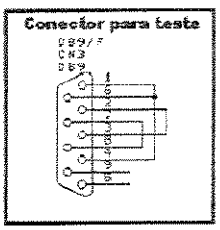



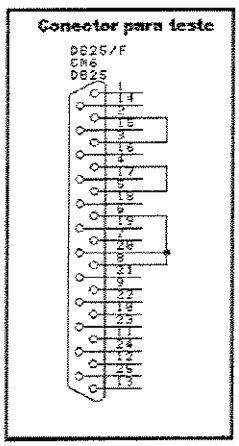



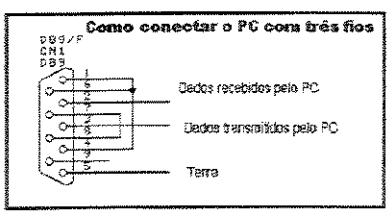

Comprimento max de cabo

| Velocidade (bauds) | distância (metros) |
|---|---|
| 19200 | 16 |
| 9600 | 165 |
| 4800 | 330 |

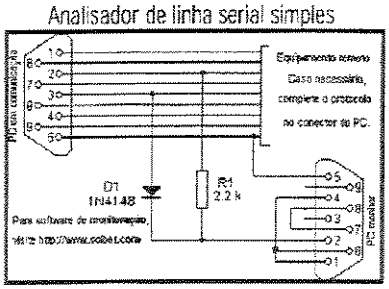
| 2400 | 990 |
|---|---|




Analisador de linha serial simples

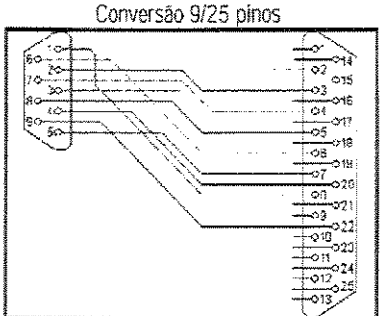


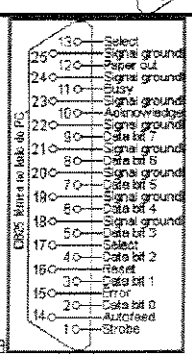

Protocolo RS232-C

Relação nível lógico/tensão

| 0 (baixo) | 1 (alto) |
|---|---|
| +3 a + 12V | -3 a -12 V |

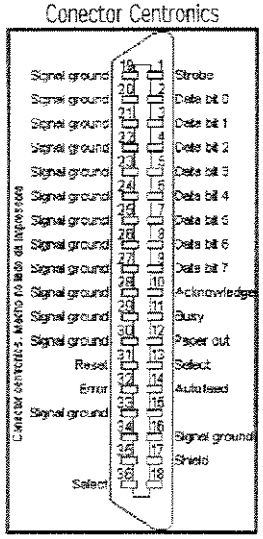
Conversão 9/25 pinos






Interface Paralela

Conector Centronics



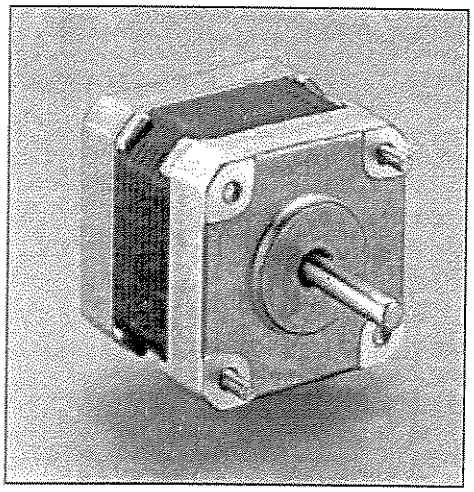

Protocolo Hexa (INHX8M)

| : | NN | EE | EE | TT | HH.. | ..HH | CC |
|---|---|---|---|---|---|---|---|

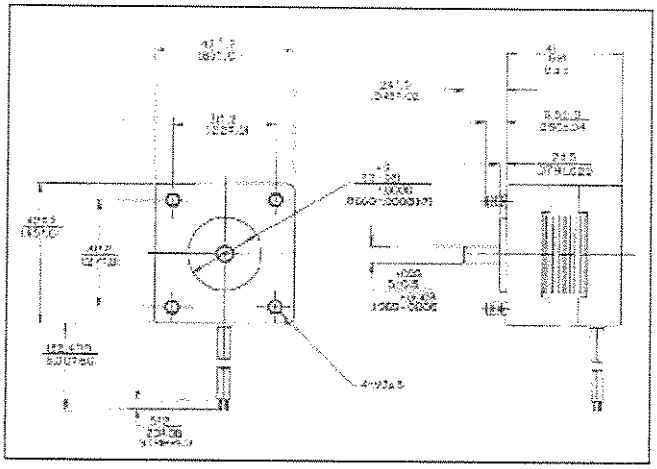
NN = número de bytes de dados
EEEE: Endereço de início de carga
TT= tipo 1=e último registro
HH .. HH = bytes de dados
CC = checksum (complemento de dois da soma de todos bytes precedentes)

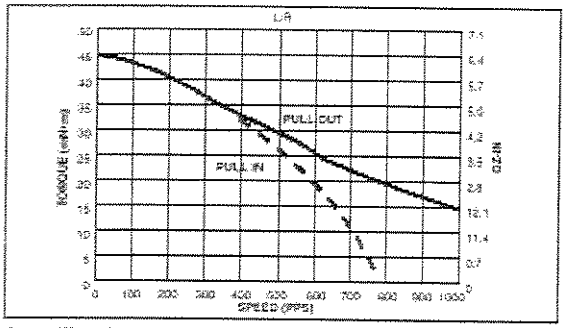
# A.6: Series 4SQ Stepper Motors 1.8 ° *Thomson Airpax Mechatronics*

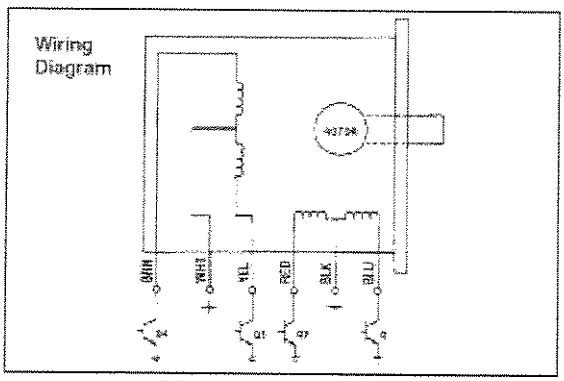
Thomson Airpax Mechatronics, Product Selection and Engineering Guide, 1995 (Series 4SQ Stepper Motors 1.8 °)



Dimensions: mm/in






NOTE: Refer to page 7 for unipolar switching sequence

Specifications

| Part Number | 4SQ - 120B34S |
|---|---|
| DC Operating Voltage | 12 |
| Res. per Winding Ω | 5 |
| Ind. per Winding mH | 26 |
| Holding Torque mN×m/oz-in | 65/9.2 |
| Rotor Moment of Inertia g×m² | $1.9 \times 10^{-6}$ |
| Detent Torque mN×m/oz-in | 8.5/01.2 |
| Step Angle | 1.8 ° |
| Step Angle Tolerance | ±5% |
| Steps per Rev. | 200 |
| Max. Radial Load* kg/lb | 4/8.8 |
| Max. Axial Load kg/lb | 3/7.6 |
| Max. Temp. Rise | 55 °C |
| Ambient Temp Range<br>Operating<br>Storage | <br>-20 °C to +50 °C<br>-20 °C to +80 °C |
| Bearing Type | Ball, Double Shielded |
| Insulation Res. at 500Vdc | 50 megohms |
| Dielectric Withstanding Voltage | 500 Vac for 60 Sec |
| Weight g/oz | 195/7 |

NOTE: Unless otherwise indicated all values shown are typical. Other windings available on special order. Consult Thomson Airpax for availability of motors with 3.6 °step angle.

* Measured with 2 grooves energized. ** Measured at 10mm from mounting plate surface.

Catalog P/N Construction

Example 1. Single Shaft Extension

4SQ — 060 — BA — 34 — S
- S — Single Shaft Extension
- 34 — 34 mm/1.3 Length
- BA — Aluminum End Bells
- 060 — 6 Volts
- 4SQ — Basic Motor

Example 2. Double Shaft Extension

4SQ — 120 — BF — 34 — W
- W — Double Shaft Extension
- 34 — 34 mm/1.3 Length
- BF — Steel End Bells
- 120 — 12 Volts
- 4SQ — Basic Motor