Robô ADR-3

Na edição nº 19 da Revista, foi publicado o Robô Explorador que utilizava como cérebro um microcontrolador PIC. Um sucesso! O nível de aprovação entre nossos leitores foi muito grande. Aliado a um baixo custo (do microcontrolador), o leitor ainda teve a oportunidade de estudar um pouco mais sobre microcontroladores PIC e sua aplicação na Robótica. Pensando nisso, desenvolvemos o Robô ADR-3. Este novo robô também utiliza um microcontrolador PIC e proporcionará uma montagem diferente e interessante, além de apresentar novos recursos do microcontrolador PIC, aumentando ainda mais os conhecimentos daqueles que se propuserem a "experimentar" mais esta montagem.

A proposta para este novo robô exigia que o mesmo fosse do tipo "interativo" com o usuário. Já publicamos outros robôs deste tipo (interativo) nesta Revista como, por exemplo, o Robô IR na edição 11 (**figura 1**), o Robô RF na edição 15 (**figura 2**), o Robô Lixobô na edição 17 (**figura 3**) e Robô Luzbô na edição 20 **(figura 4)**. Cada qual com seu tipo de "interatividade". Para esta nova proposta, necessitávamos algo novo. E o autor conseguiu. Criou um robô que pode ser programado!

Sim, é isso mesmo! O Robô ADR-3 permite que o usuário programe seus movimentos, que são:

• Andar para frente;

• Andar para trás;

• Desviar para direita;

• Desviar para esquerda;

• Tocar buzzer.

Apesar da simplicidade dos comandos presentes, muitos outros podem ser implementados no futuro pelo próprio leitor, desde que o mesmo conheça um pouco sobre a linguagem de programação utilizada (Assembly).

Todo o movimento do robô é medido em “passos”. O leitor pode inserir até 64 comandos diferentes e consecutivos para movimentar o robô. Sendo assim, ele poderá programar uma série de movimentos no robô para que o mesmo execute uma determinada tarefa, como se deslocar por uma sala sem colidir com os obstáculos, ou ainda “entrar e sair” de um labirinto. Temos certeza que nosso leitor já tem em mente outras maneiras de utilizar o Robô ADR-3.

As características técnicas do Robô ADR-3 são as seguintes:

• **Cérebro**: microcontrolador PIC16F877 Microchip®;

• **Motores:** dois motores de passo 12 V – tipo unipolar;

• **Entrada de dados (programa):** através de teclado matricial com 20 teclas;

• **Visualização dos dados (programa):** por display LCD 16x2 do tipo paralelo;

• **Chassi:** construído com caixas plásticas comerciais e tubos de PVC;

• **Outras “interações”:** LEDs nas laterais informando o sentido do movimento e buzzer;

• **Alimentação:** bateria GEL 12 V / 1,3 A (pode-se utilizar 8 pilhas grandes também).

Após a entrada dos dados, os mesmos são gravados na EEPROM (memória não volátil) do microcontrolador. Assim, mesmo que o robô seja desligado, o programa continuará disponível para execução (tecla RUN).

Um estudo no programa do robô poderá ajudar o leitor interessado em aprender mais sobre a linguagem "Assembly" dos microcontroladores PIC. Além dos controles habituais, já discutidos em nossa série sobre microcontroladores PIC como: display, teclado matricial e motores de passo, ele receberá dicas importantes sobre o uso da interrupção do timer, memória EEPROM do microcontrolador, e inclusive algumas dicas sobre "expansões" para o número de pinos de I/O.

O CIRCUITO

Como dito anteriormente, utilizamos o microcontrolador PIC16F877 como cérebro para nosso robô. Este microcontrolador possui as seguintes características:

• Memória FLASH de programa com 8 kbytes;

• 368 bytes de RAM;

• 256 bytes de EEPROM;

• 32 pinos de I/O com dreno de corrente na ordem de 20 mA;

• 10 conversores analógicos (AD) de 10 bits;

• Dois "timers" de 8 bits e um de 16 bits;

• Dois canais CCP (Capture, Compare and PWM);

• Um canal USART para comunicação RS-232;

• Comunicação SPI ou I2C;

• Várias opções para oscilador:

• RC;

• Cristal ou ressonador;

• Cristal de alta freqüência (acima de 10 MHz);

• Clock externo.

• Watch Dog Timer (cão de guarda) interno;

• Controle de "Power-on Reset" e "Power-up Timer";

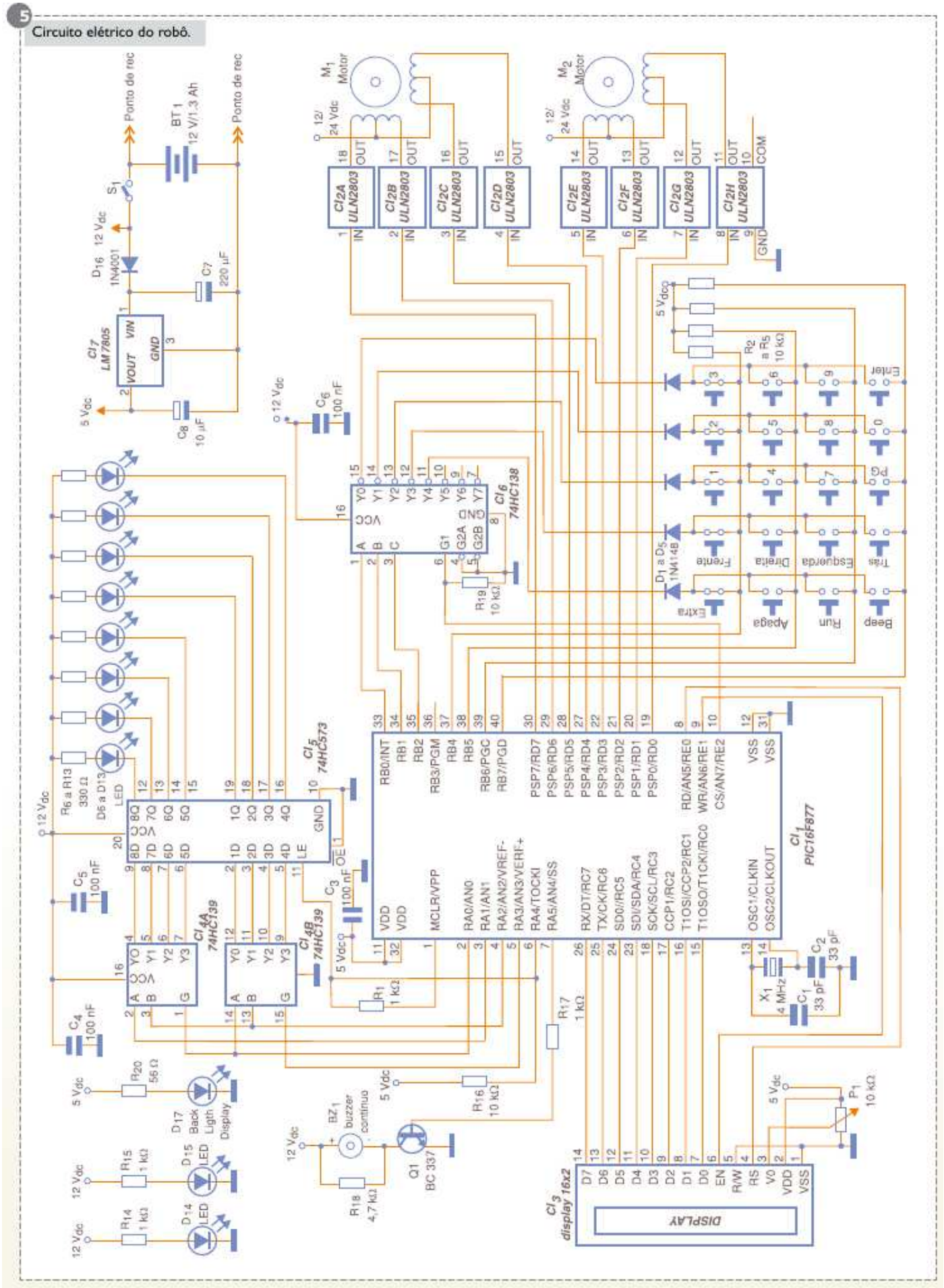
• Proteção de código contra cópias;

• Código de instruções reduzido (35 instruções);

• Encapsulamento DIP com 40 pinos;
• etc.

Para aqueles que desejam mais informações sobre este microcontrolador, aconselhamos o download do data sheet no site do fabricante, **www.microchip.com.**

Na **figura 5**, inserimos o circuito elétrico do robô ADR-3. Todo o controle dos periféricos é feito por CI1 (microcontrolador PIC16F877). Um detalhe importante sobre o circuito são os Cis "auxiliares" utilizados para, "expandir" o número de I/Os do microcontrolador.



Note que para controlar todos os periféricos seriam necessários 36 pinos de I/O: oito para os motores, dez para o display, um para o buzzer, oito para os LEDs e nove para o teclado. O microcontrolador só possui 32 pinos de I/O.

Sendo assim, alguns recursos para “ampliar” esse número foram usados. O primeiro deles diz respeito ao controle dos LEDs sinalizadores (D6 a D13) que indicam o sentido do movimento.

Precisávamos de oito pinos de I/O do microcontrolador para esse controle e utilizamos apenas cinco. Para isso, contamos com CI4 e CI5. CI4 é um duplo “decoder/demultiplex” de 4 linhas (entrada binária, saída decimal) e CI5 um “latch” de 8 bits. A entrada de ambas as partes do 74HC139 (entrada A e B) é controlada pelos mesmos pinos de I/O do microcontrolador e sua seleção por pinos independentes.

Vamos explicar melhor. Sempre que inserimos um valor binário nas entradas de CI4 (A e B), a saída correspondente a este valor é levada ao nível lógico “0”. Por exemplo, se desejamos fazer a saída “Y2” igual a nível lógico “0”, basta inserir nas entradas A e B os valores “0” e “1” lógico (2 binário), respectivamente. Para que este valor seja inserido em uma ou outra parte do CI, basta habilitar o pino “G” da parte desejada com um nível lógico “0”. Neste momento temos o valor desejado na saída do 74HC139 (parte selecionada).

A cada alteração do 74HC139 é necessário “inserir” o dado nos LEDs. Essa é a função de CI5, um “latch” de oito bits. Para isso basta ativar o pino de transferência de dados da entrada para a saída LE (pino 11) do CI5. Feito isso, o dado permanecerá no “buffer” e o LED ficará aceso, mesmo que o 74HC139 seja desabilitado.

Isso pode parecer um pouco estranho para o leitor, mas para compreender melhor basta conferir as **tabelas 1 e 2**. Estas tabelas contêm os estados possíveis do CI4 e do CI5 para os pinos de saída de acordo com as suas entradas (tabela verdade). Nelas estão representadas as seguintes possibilidades:

T1 Tabela-verdade para o 74HC139.

Entradas			Saídas			
/E	A	B	/Y0	/Y1	/Y2	/Y3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	1	0	1	0	1	1
0	0	1	1	1	0	1
0	1	1	1	1	1	0

T2 Tabela-verdade para o 74HC573.

Entradas			Saídas
/OE	LE	D	Q
1	X	X	Z
0	1	1	1
0	1	0	0
0	0	X	Q anterior

0 – nível lógico baixo (GND)

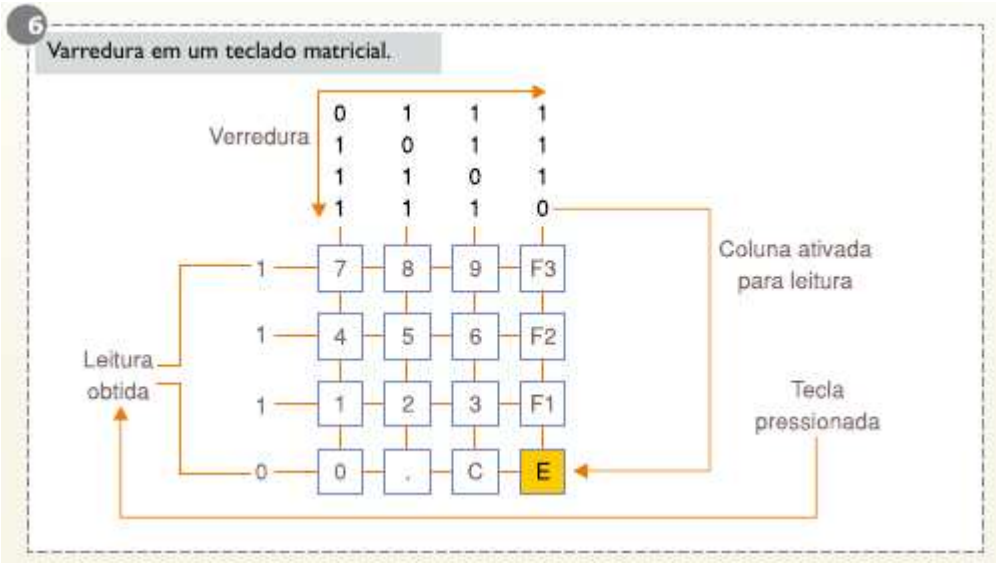
1 – nível lógico alto (5 VDC)

X – nível lógico pode assumir “0” ou “1”

Q anterior – estado anterior / - ativo no nível baixo

CI6 é um outro CI utilizado para “ampliar” os pinos de I/O do microcontrolador. Este CI amplia o número de colunas do teclado matricial. Para operar um teclado matricial, precisamos utilizar o método da varredura. Veja a **figura 6.**

6 Varredura em um teclado matricial.



Para saber qual tecla foi pressionada, temos que inserir um valor lógico "0" em uma coluna, mantendo todas as outras em nível lógico "1", e verificar o estado das linhas. Se uma delas indicar o estado "0", é sinal de que temos uma tecla pressionada. Como sabemos em qual coluna foi inserido o nível lógico "0" e temos agora em qual linha o mesmo apareceu, podemos determinar através de uma tabela interna ao programa qual tecla foi pressionada. Se nenhum valor lógico igual a zero aparecer nas linhas, é sinal que nada foi pressionado e podemos passar para a coluna seguinte e assim sucessivamente até a última. Caso nenhuma tecla seja pressionada, fazemos novamente uma varredura (novo ciclo), até que uma tecla seja pressionada.

O funcionamento do 74HC138 (CI6) é exatamente o mesmo do 74HC139 (CI4), explicado anteriormente. A diferença é que não há a divisão de duas partes no CI6. A **tabela 3** mostra a "tabela-verdade" do 74HC138. Observando-a, podemos verificar que alterando o estado de quatro pinos apenas (E3, A, B e C), podemos alterar oito saídas e todas com o estado lógico desejado para a coluna do teclado (nível lógico "0"). Um outro detalhe importante é que utilizamos cinco colunas para nosso teclado e o leitor tem a possibilidade de empregar outras três (Y5, Y6 e Y7), o que aumentaria o número de teclas para 32, caso deseje ampliar o número de comandos e recursos do robô.

T3 Tabela-verdade para o 74HC138.

Entradas						Saídas							
/E1	/E2	E3	A	B	C	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
1	X	X	X	X	X	1	1	1	1	1	1	1	1
X	1	X	X	X	X	1	1	1	1	1	1	1	1
X	X	0	X	X	X	1	1	1	1	1	1	1	1
0	0	1	0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	0	1	0	1	1	1	1	1	1
0	0	1	0	1	0	1	1	0	1	1	1	1	1
0	0	1	1	1	0	1	1	1	0	1	1	1	1
0	0	1	0	0	1	1	1	1	1	0	1	1	1
0	0	1	1	0	1	1	1	1	1	1	0	1	1
0	0	1	0	1	1	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	0

CI3 é um display de cristal líquido tipo caractere com 16 colunas e duas linhas (16 x 2) paralelo (sem circuito de comunicação serial), com ou sem "back ligth" (em nosso protótipo utilizamos um com "back ligth"). Este tipo de display é facilmente encontrado em lojas especializadas na venda de componentes eletrônicos.

CI2 é um “line driver” capaz de operar tensões de até 50 V sob uma corrente máxima de 500 mA por porta. O ULN2803 foi desenhado para trabalhar com tensões de entrada compatíveis com as linhas TTL e CMOS (5 V a 15 V). Este CI controla os motores de passo que devem ser, obrigatoriamente, do tipo unipolar (cinco ou seis fios). Os tipos bipolares (quatro fios) não servem neste circuito!

Importante:

Em muitos outros projetos apresentados aqui na Revista, utilizamos muitas vezes displays LCDs do tipo serial, que necessitavam de apenas um único pino para comunicação com um microcontrolador. O LCD requerido no presente projeto não é desses. LCDs seriais não servem para o Robô ADR-3.

O leitor deve estar atento para o código deste CI. Trata-se do ULN2803. Ele possui oito portas no total contra sete do ULN2003. O uso deste último deve ser evitado no circuito do ADR-3.

Além dos circuitos “ampliadores” (auxiliares), drives e display, temos um buzzer de 12 V do tipo contínuo (com oscilador interno), dois LEDs auxiliares de 10 mm e um circuito regulador para 5 VDC montado com CI7, D16, C7 e C8. O cristal X1 mais os capacitores C1 e C2 formam o circuito oscilador, necessário para o microcontrolador.

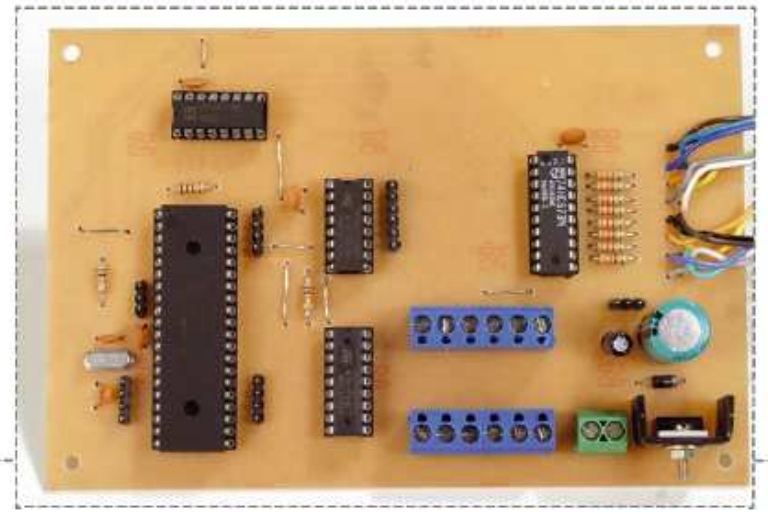
Mais à frente, quando for explicado o funcionamento do programa, o leitor poderá compreender melhor o funcionamento do circuito. Lembre-se que em um circuito microcontrolado a análise do “programa” e “circuito” deve ser, preferencialmente, feita em conjunto.

A MONTAGEM

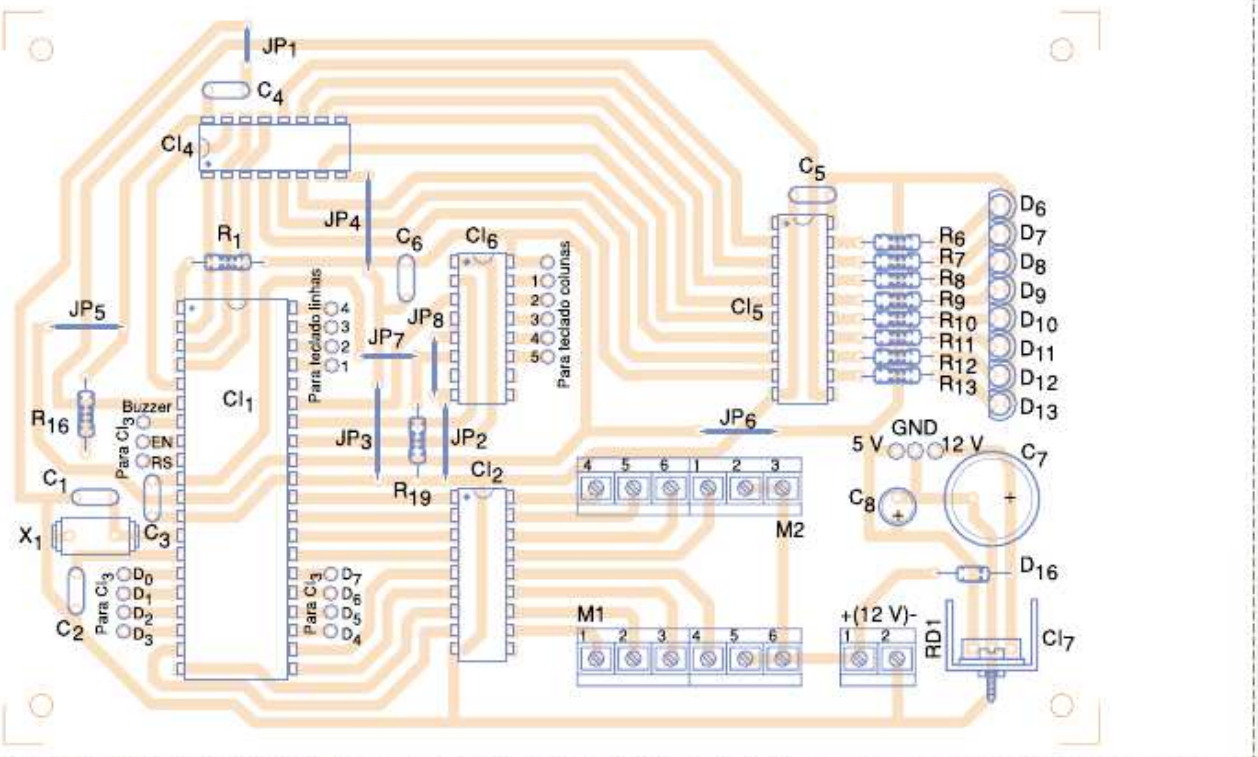
Dividimos nossa montagem em “Montagem Eletrônica” e “Montagem Mecânica”. Assim o leitor poderá obter detalhes importantes sobre cada uma.

Montagem eletrônica

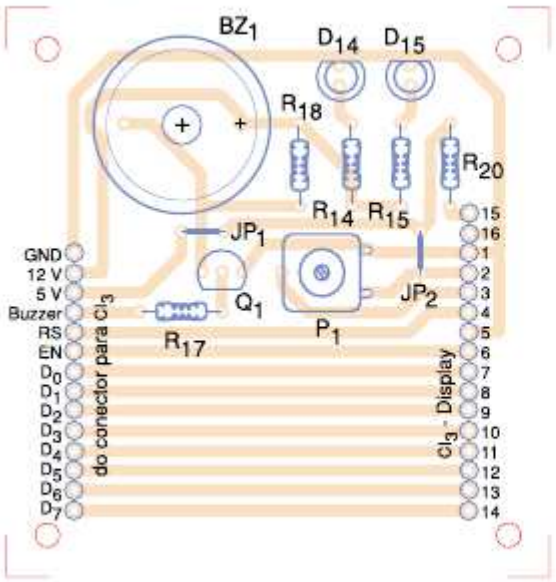
Nas ***figuras 7 a 9*** o leitor tem os layouts das placas desenvolvidas e utilizadas em nosso protótipo. Temos três placas, a saber:

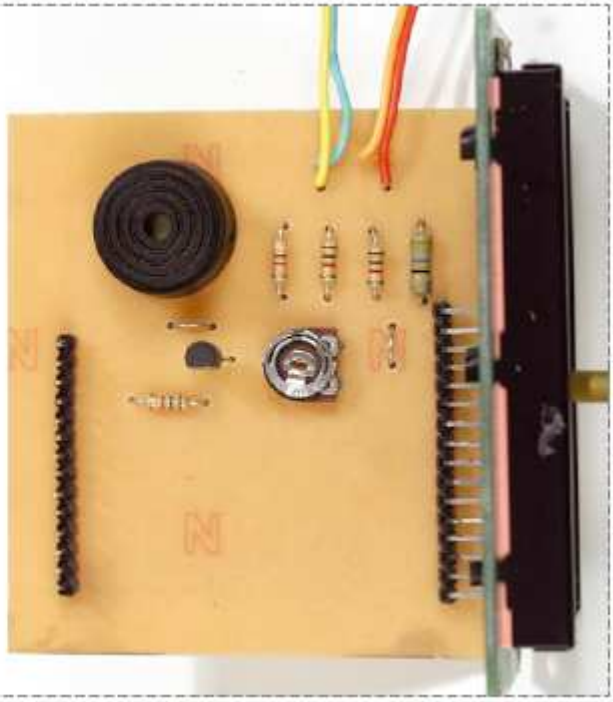


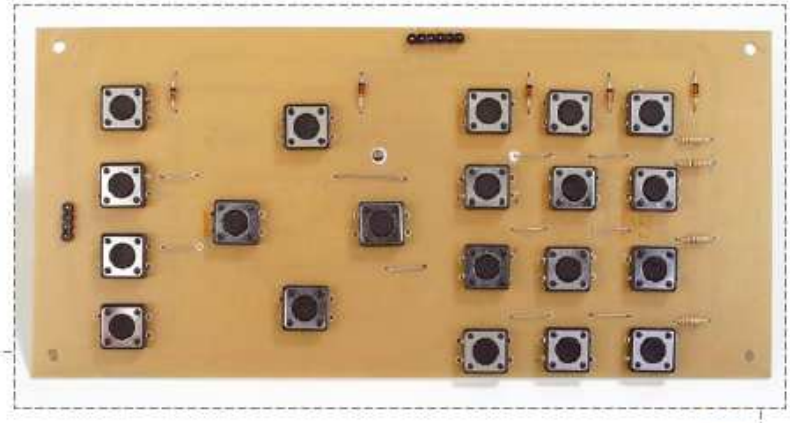
7 *Layout* da placa principal.



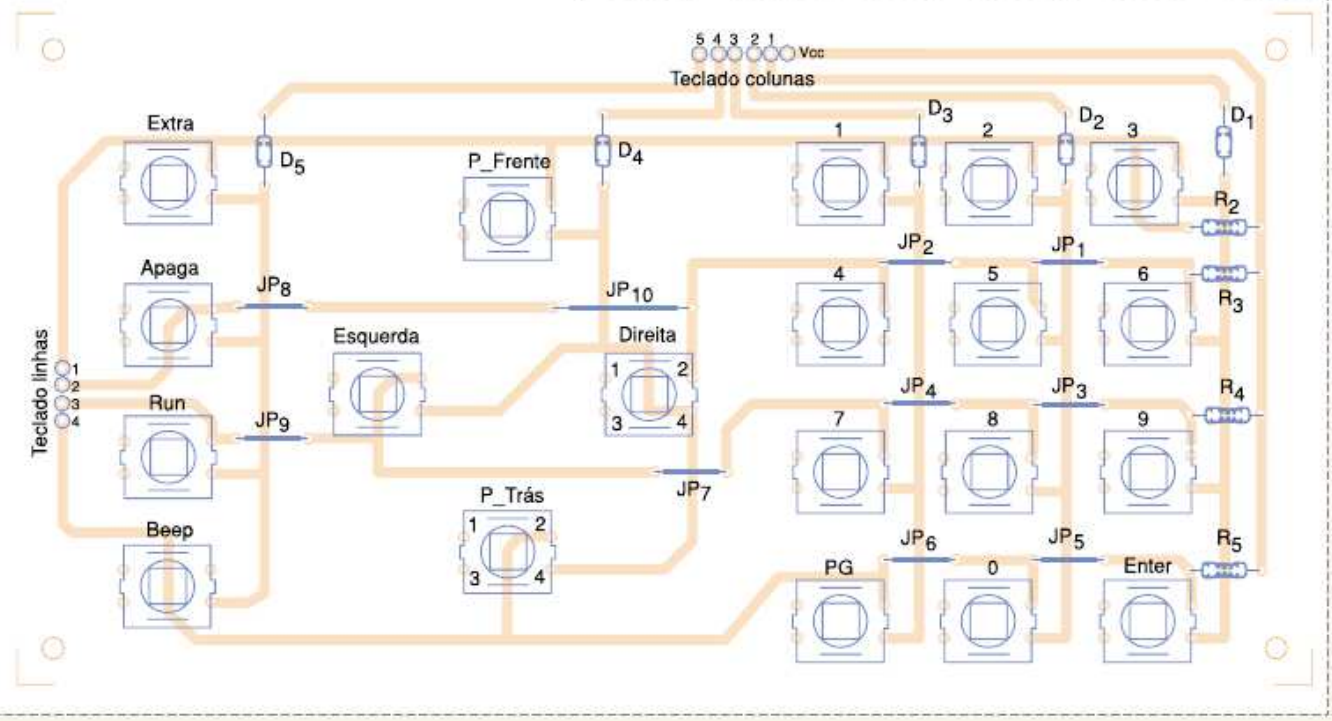
8 *Layout* da placa display.







9 *Layout* da placa teclado.



• **placa principal:** nesta placa fica o microcontrolador, CIs auxiliares, LEDs e controle do motor.

• **placa display:** placa que conterá o display, LEDs extras e buzzer.

• **placa teclado:** placa para montagem do teclado matricial.

A montagem em três placas permite a separação dos itens de acordo com o desenho do chassi, demonstrado mais à frente. Um outro ponto é que assim, não temos uma placa "extragrande" com todos os itens (muito mais difícil para ser confeccionada).

Ao finalizar a confecção das placas, verifique todas com muita atenção. Note que existem muitas trilhas, ilhas, jumpers e outros. São placas complexas e é aconselhável uma boa verificação após a sua confecção. Não se apresse em soldar os componentes. Verifique todas as ligações, possíveis curtos em ilhas e trilhas, furações, enfim tudo!

Comece por soldar os componentes como capacitores cerâmicos, o cristal e resistores. São componentes não polarizados e não requerem muito cuidado com possíveis inversões. Mesmo assim, tenha bastante atenção para seus valores. Qualquer troca poderá provocar o não funcionamento do circuito e quiçá a queima deste posteriormente, quando o mesmo for ligado.

Solde em seguida os soquetes dos CIs. Não solde nenhum CI diretamente na placa. O uso de soquetes para os mesmos ajudará em futuras manutenções ou talvez nos seus reaproveitamentos em outros circuitos (testes). A utilização de um soquete para CI1 (microcontrolador) é obrigatória.

Ao soldar os componentes polarizados como transistores, diodos, LEDs, capacitores eletrolíticos, buzzer, etc., tenha cuidado para não invertê-los. Isso, além de provocar o não funcionamento do circuito (ou parte dele), poderá causar a queima parcial ou total do mesmo.

A interligação entre as placas, LEDs, chave liga/desliga, ponto de recarga e motores deve ser feita com fios. O uso de barra de pinos e terminais de conexão, como feito em nosso protótipo, é livre. O leitor que não tiver este material em sua bancada, poderá soldar diretamente os fios às placas. Lembre-se apenas de tomar o máximo cuidado para não inverter nada! Aqui as inversões também podem ser fatais!

Nos desenhos das placas, todos os pontos de interligações foram ilustrados para não deixar qualquer dúvida. Ligue-os exatamente conforme descrito e, em caso de dúvidas, recorra ao circuito elétrico. O comprimento destas ligações deve ser tal que permita a instalação das placas no chassi de forma "confortável".

O uso de terminais para conectar a bateria é muito aconselhável (**figura 10**). Evite soldar os fios diretamente nos terminais da mesma. O calor poderá danificar o GEL interno estragando assim a bateria.

É extremamente aconselhável, ao finalizar esta etapa (montagem eletrônica), que todas as ligações sejam revistas. Use um multímetro para tal. Não insira nenhum CI ainda. Use os pontos nos soquetes para verificar as ligações descritas no esquema elétrico.

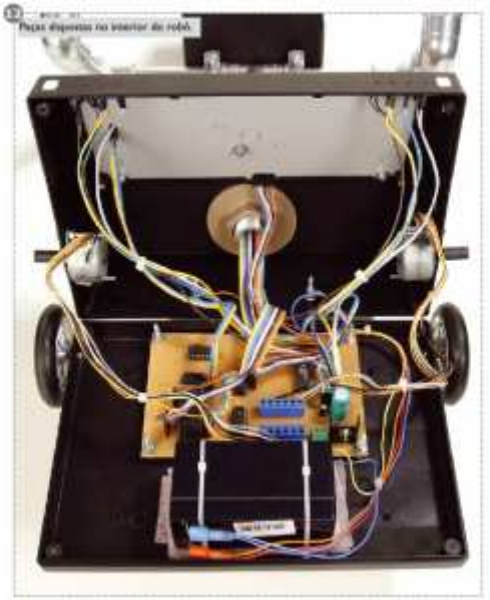
Montagem mecânica

Na **figura 11**, o leitor encontrará detalhes das peças utilizadas no chassi do nosso protótipo. Note que usamos caixas comerciais Patola, canos e curvas de PVC de ¾ de polegada (tubos empregados em encanamento d'água).

A "cabeça" do robô é uma caixa plástica Patola código PB215, a "base" uma outra caixa da Patola código PB900. O corpo do robô é composto de pedaços de cano PVC, curvas de 90° e uma "flange" na base.

Em nosso site o leitor encontrará desenhos com as medidas que facilitarão a furação no caso da escolha da montagem de acordo com nosso protótipo, além das medidas dos tubos de PVC utilizados.

Na **figura 12**, o leitor poderá ver como disponibilizamos a placa principal e outros itens internamente em nosso protótipo.



Na **figura 13**, o leitor poderá ver no detalhe que providenciamos pontos ligados internamente a bateria para a recarga da mesma. Nessa figura é possível notar seu detalhe, pelo lado interno do robô. Caso o leitor utilize pilhas para alimentar o robô, poderá omitir estes pontos.

A “garra” do robô foi feita com um pedaço de cano PVC. Um corte foi feito no mesmo para simular uma garra. A união da “garra” ao “antebraço” foi feita com massa tipo adesivo “epóxi”.

Os LEDs utilizados como “olhos” do robô são do tipo redondo com 10 mm de diâmetro, também conhecidos no mercado como “ledões”. O leitor poderá usar qualquer um que julgar interessante.

As rodas empregadas são do tipo para aeromodelos com três polegadas de diâmetro. A roda “boba” (terceira roda) é do tipo comercial e sua escolha deve atender a uma única exigência: manter o robô em equilíbrio.

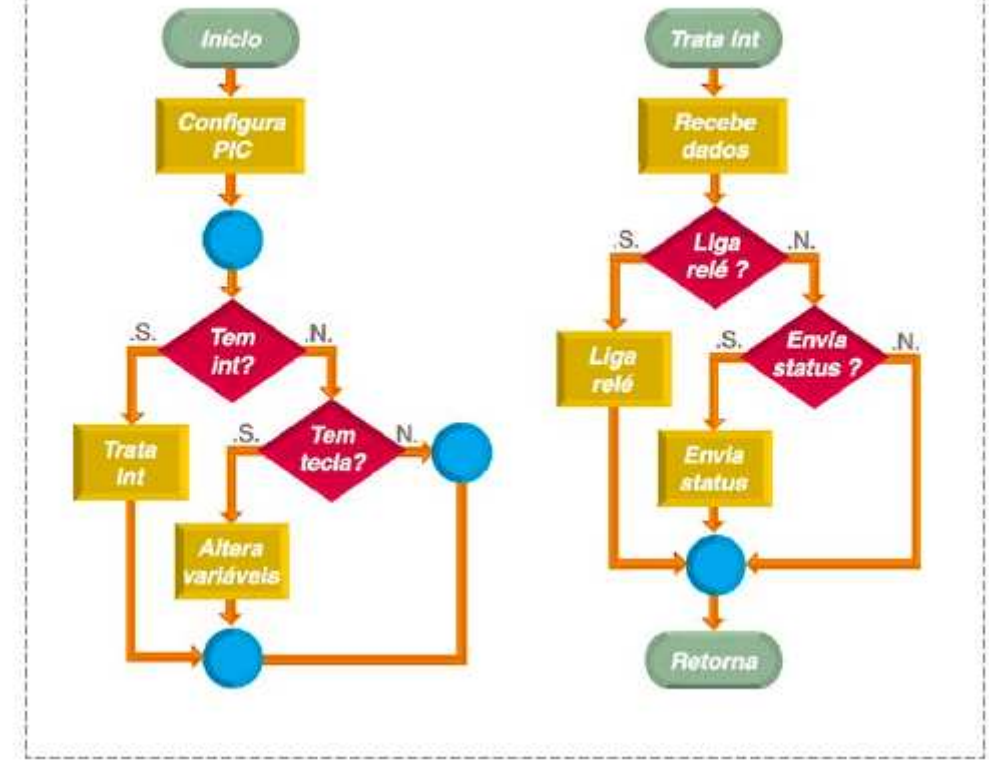
O acabamento do robô foi feito com tinta “spray” na cor metálica para os tubos e curvas de PVC e também para os cubos das rodas e na cor cinza para as tampas dos gabinetes.

O PROGRAMA

Aconselhamos que o leitor faça o “download” do programa (código-fonte) em nosso site, para acompanhar melhor o que será explicado a seguir.

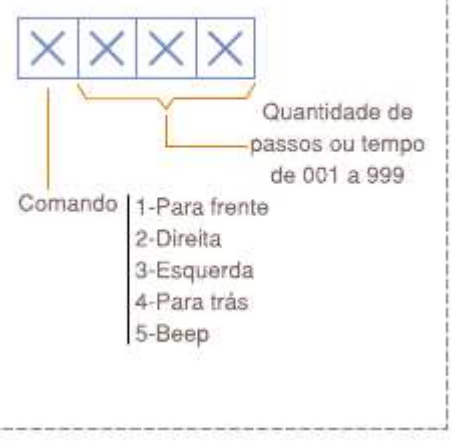
Na **figura 14** temos o fluxograma do programa ROBO_ADR3.ASM. O programa inicia o microcontrolador (periféricos) e variáveis. Em seguida, um “laço eterno” é mantido para verificar a presença de uma tecla ou movimentar o robô. Note que existe um controle através de “flags” (bits) em variáveis que determina se há ou não um programa em execução e, portanto, se os motores devem ou não ser colocados em movimento.

14 Fluxograma do programa.



Se um programa estiver na memória e a tecla “RUN” for pressionada, o robô “lerá” os bytes na EEPROM referentes ao comando (4 no total) e o executará. O protocolo estabelecido para os comandos está descrito na **figura 15.**

15 Protocolo para os comandos.



O primeiro byte define o tipo de comando (são cinco os comandos possíveis), e os três últimos bytes o valor dos passos ou tempo desejado para o beep. Assim é possível determinar a direção e a quantidade de passos desejada, ou ainda se o robô deve ou não tocar o beep e por quanto tempo (de 1ms a 999ms).

Um detalhe importante a respeito do programa é o uso da interrupção do TIMER0 do microcontrolador. Ele é utilizado para acender os LEDs na direção em que o robô estiver seguindo. A **figura 16** demonstra isso. O uso da interrupção do timer permite que mais de um processo seja executado ao mesmo tempo, sem a preocupação do controle de tempo "manual".

A gravação e leitura da EEPROM são feitas durante a programação do robô e a execução do programa, respectivamente. O programa termina sempre quando o contador de comando chega a 64 ou ainda quando um valor "FFH" é encontrado na EEPROM.

Os motores são controlados manualmente e o tempo entre os passos pode ser alterado, aumentando ou diminuindo a velocidade de deslocamento do mesmo. O controle do teclado é feito através do método da varredura. O controle do LCD é efetuado de forma direta, com comunicação de 8 bits. Todos estes periféricos foram discutidos na série sobre microcontroladores PIC publicada entre a edição nº 6 e edição nº 13. Aconselhamos a todos, dentro do possível, o estudo desta série, pois muitos conceitos básicos foram discutidos através dela.

Importante

A movimentação do robô depende da presença de um programa (um único comando já representa um programa para o robô) e também do "pressionar" da tecla RUN.

O programa tem mais de 2.500 linhas de código Assembly. O leitor deve ter em mente que se trata de uma linguagem não muito intuitiva e amigável. A compreensão do programa na sua totalidade não poderá ser feita em "minutos". Será necessária alguma experiência com a linguagem e uma boa dose de paciência perseverança para alcançar o objetivo Aconselhamos o estudo do programa, dividindo-o em sub-rotinas. Para ajudar ainda mais, o programa foi ricamente comentado pelo autor.

TESTE E USO

Para testar o robô, o leitor não precisará ter a "montagem mecânica" pronta. Todos os testes poderão ser feitos na bancada, apenas com as placas interligadas, LEDs e motores conectados, além da alimentação (bateria).

Importante

Não entraremos em detalhes sobre a gravação do microcontrolador neste artigo. Julgamos que o leitor que se propôs a montar este circuito tem a experiência necessária com microcontroladores e conhece as operações envolvidas no processo de gravação dos mesmos. Para os leitores sem experiência no assunto, aconselhamos a leitura da série sobre "Microcontroladores PIC" do autor Márcio José Soares, nas edições de nº 6 a nº 13.

Antes de alimentar o circuito, refaça visualmente uma inspeção em todas as ligações, posições dos CIs e outras. É preferível perder alguns minutos agora a perder horas tentando encontrar um possível erro na montagem.

Grave no microcontrolador o programa ROBO_ADR3.HEX, disponível em nosso site. Instale o microcontrolador no circuito, conecte a alimentação (bateria de 12 V ou conjunto de oito pilhas em série) e ligue o robô.

Ao ser ligado, o robô emite um beep de um segundo e mostra no display **"Robô ADR-3"**, posicionado no centro. Este é o momento para verificar a regulagem de P1. Este "trimpot" regula o contraste do display. Gire-o em ambos os sentidos até obter o melhor contraste possível.

Agora entre com um programa qualquer. Os passos são os seguintes:

• **Tecle "PG" (programa):** após isso um beep será ouvido e o robô solicitará o comando;

• **Tecle o comando desejado:** teclas para frente, para trás, direita, esquerda e beep;

• **Entre com o número de passos ou tempo:** se desejar alterar o comando, durante a sua entrada basta teclar "A" (anula) e o comando será anulado.

• **Tecle "E" (enter):** após entrar com os passos ou tempo desejado. Um novo beep será ouvido e o robô avisará que o comando foi gravado.

Neste momento o ciclo se repetirá, e um novo comando será pedido, até que a memória de programa fique cheia (64 comandos) ou o usuário decida que já tem os comandos necessários na memória.

Para sair do modo "programa", basta teclar novamente "PG".

Importante

Ao entrar com um valor durante a "programação" dos movimentos, ou mesmo do tempo do buzzer, serão solicitados três dígitos. Os valores possíveis vão de 001 a 999. Caso o leitor deseje entrar com um valor como 50, por exemplo, deverá digitar 050 e não 50 ou o programa interpretará o valor como 500.

Agora o leitor tem um programa na memória pronto para a sua execução. Para rodá-lo, basta teclar "Run" e o robô será colocado em movimento (de acordo com o programa inserido pelo leitor).

Este também é um ótimo momento para finalizar a configuração do robô. Verifique se o movimento dos motores está de acordo com o sentido. Caso contrário, mude a posição dos motores, fios, etc. Lembre-se que o comando "desviar a esquerda e direita" diz respeito à "esquerda e direita" do robô e não do "espectador".

Os LEDs sinalizadores também podem ser posicionados no chassi de acordo com o efeito desejado (já explicado) ou outro qualquer outro que o leitor julgar mais interessante.

Se algo no teclado lhe parecer estranho como teclas trocadas, por exemplo, bastará verificar atentamente a posição das ligações entre linhas e colunas das placas principal e de teclado. Caso as mesmas tenham sido invertidas, o teclado não funcionará corretamente.

Com tudo testado, basta inserir as placas no chassi montado e divertir-se criando programas para a movimentação em sua sala, casa, escola, feiras e demonstrações, etc.

AJUDA COM PROBLEMAS

A seguir, listamos algumas dicas para possíveis problemas que possam surgir durante os testes do robô:

Liguei o robô e nenhum beep foi ouvido!

Possíveis soluções:

• Verifique se a alimentação está presente;

• Cheque se o microcontrolador foi gravado corretamente;

• Observe se o beep não foi ligado de maneira invertida (pinos);

• Verifique se o beep possui oscilador interno, conforme solicitado;

• Cheque se Q1 também não foi ligado de forma invertida;

• Confirme se a ligação entre a placa principal e display está correta.

Liguei o robô, ouvi o beep, mas nada foi mostrado no display!

Possíveis soluções:

• Verifique se a alimentação está presente no display (pinos 1 e 2);

• Cheque se a ligação entre a placa principal e display está correta;

• Faça ajustes em P1 até que algo seja mostrado no display;

• Confirme se o display está de acordo com o solicitado.

Liguei o robô, ouvi o beep, li a mensagem inicial no display, mas o teclado parece não funcionar corretamente. A tecla PG não funciona!

Possíveis soluções:

• Verifique as ligações entre a placa principal e a placa teclado.

Os motores parecem não girar de acordo com os comandos na memória!

Possíveis soluções:

• Troque as ligações de posição motor 1 com motor 2 se a inversão for entre os comandos “para frente” e “para trás”.

• Mude de posição os fios dos motores (fases) se a inversão for entre os comandos “direita” e “esquerda”.

• Reveja todas as ligações dos motores, até que as condições sejam satisfeitas.

Meu robô funciona bem, mas por pouco tempo! Usei pilhas e parecem não durar muito!

Possíveis soluções:

• Troque a alimentação com pilhas por uma com bateria de 12 V;

• Verifique se os motores não estão drenando corrente excessiva (até 500 mA é aceitável). Troque-os se este for o caso;

• Cheque o ajuste dado entre a roda e o eixo do motor. Quanto mais pressionado estiver, maior será o esforço do motor. O ajuste deve ser tal que permita o contato, mas sem pressão excessiva. A pressão deve apenas garantir o movimento, sem a perda de “passos”.

CONCLUSÃO

O robô apresentado neste artigo usou recursos já conhecidos dos nossos leitores, mas também acrescentou novos conhecimentos, principalmente na programação dos microcontroladores PIC Microchip®. Acreditamos que tanto o leitor com experiência no assunto como também o inexperiente poderão tirar proveito do circuito, seja montando o robô, seja aproveitando “partes” do mesmo em outros projetos. Esperamos que todos tenham gostado! Boa montagem e até a próxima!

Lista de material:

Semicondutores

CI_1 – PIC16F877 – microcontrolador Microchip®
CI_2 – ULN2803 – driver para motor de passo
CI_3 – Display de cristal liquido (LCD) paralelo 16x2 com "Back Light"
CI_4 – 74HC139
CI_5 – 74HC573
CI_6 – 74HC138
CI_7 – LM7805 – regulador de 5VDC

Q_1 – BC337 – transistor NPN
D_1 a **D_5** – 1N4148 – diodo de sinal
D_6 a **D_{13}** – LED comum redondo ∅5 mm
D_{14}, **D_{15}** – LED redondo ∅10 mm
D_{16} – 1N4001 – diodo retificador
D_{17} – "Back Light" do LCD

Resistores (dissipação de 1/8W, salvo indicação contrária)

R_1 – 1 kΩ (marrom, preto, vermelho)
R_2 a **R_5** – 10 kΩ (marrom, preto, laranja)
R_6 a **R_{13}** – 330 Ω (laranja, laranja, marrom)
R_{14}, **R_{15}** - 1 kΩ (marrom, preto, vermelho)
R_{16} - 10 kΩ (marrom, preto, laranja)
R_{17} - 1 kΩ (marrom, preto, vermelho)
R_{18} – 4,7 kΩ (amarelo, violeta, vermelho)
R_{19} - 10 kΩ (marrom, preto, laranja)
R_{20} – 56 Ω (verde, azul, preto) – 1/4W

Capacitores

C_1, **C_2** – 33 pF – cerâmico
C3 a **C6** – 100 nF – cerâmico
C_7 – 220 µF/25V – eletrolítico
C_8 – 10 µF/16V – eletrolítico

Diversos

BZ_1 – Buzzer (*beep*) com oscilador interno
P_1 – 10 kΩ - trimpot mini horizontal
X_1 – 4 MHz – cristal oscilador
M_1, **M_2** – motores de passo, bobina de 12 V, unipolar
S_1 – chave liga/desliga
BT_1 – bateria 12V/1,3 A GEL (ou 8 pilhas)

20 teclas mini para montagem em placa (veja texto e figuras)

1 – soquete para CI 40 pinos
1 – soquete para CI 20 pinos
2 – soquetes para CI 16 pinos
1 – soquete para CI 18 pinos
1 – radiador de calor TO-220

1,5 - metros de cano PVC ¾ de polegada (usado em tubulação d'água)
4 – "cotovelos" (curvas) 90° para cano PVC ¾ de polegada (p/ cola)
1 – flange para cano PVC ¾ de polegada (p/ cola)
1 – "T" para cano PVC ¾ de polegada (p/ cola)
1 – tubo pequeno de cola para cano PVC

Barra de pinos, 1 metro de cabo *flat* com 16 vias, placas de circuito impresso, conectores parafusáveis para os motores, solda, gabinetes para chassi, duas rodas para aeromodelos, parafusos para montagem, conectores diversos, tinta *spray* para acabamento, etc

Fonte: http://www.mecatronicaatual.com.br