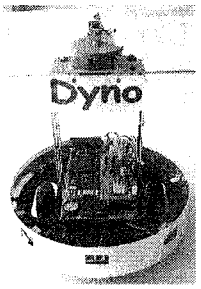


Dyno: Um Robot com Hardware Reconfigurável

Arnaldo Oliveira, Andreia Melo

I. INTRODUÇÃO

O robot Dyno foi desenvolvido tendo por base o regulamento e as especificações técnicas fornecidas pela organização do concurso MicroRato. No seu circuito de controlo foram utilizados um microcontrolador da família 8051 e uma Field Programmable Gate Array (FPGA). Sendo um dispositivo lógico programável, a FPGA permitiu por um lado concentrar num único circuito integrado funções normalmente dispersas por vários (ex. *latch* e descodificação de endereços) e por outro desempenhar funções repetitivas de forma eficiente sem a intervenção do microcontrolador (ex. controlo PWM dos motores). Adicionalmente, permitiu que a partição da implementação de algumas funções entre o hardware e o software fosse efectuada após a execução do circuito impresso. As próximas secções descrevem de forma sucinta a estrutura do hardware e do software do robot Dyno.

II. HARDWARE

O hardware fornecido pela organização para construção deste robot consistiu numa base de madeira, motores, rodas, sensores de infravermelhos (IV) para detecção de obstáculos, do farol e do chão da área de chegada. Como placa principal de controlo foi utilizada uma XS40 desenvolvida pela XESS Corporation contendo os seguintes componentes: um microcontrolador 80C154 da Intel, uma FPGA XC4005XL da Xilinx, 32 Kbytes de RAM, 256 Kbits de EEPROM para configuração da FPGA e interface paralelo (Centronics) para programação. Especificamente para aplicação no robot foram desenvolvidas as seguintes placas:

- Extensão para a XS40 constituída por 16 entradas digitais, 16 saídas digitais, 8 entradas analógicas, 32 Kbytes de RAM protegida por um supervisor de tensão e uma pilha de *backup*;
- Interface para ligação dos sensores e motores contendo circuitos de comutação, *drivers* e conectores.

A fig. 1 ilustra a disposição das placas na parte superior da base do robot. À direita temos a placa de controlo, em baixo a placa de expansão e à esquerda a de interface.

A fig. 2 mostra de forma simplificada a estrutura do hardware de controlo do robot. O circuito de supervisão, juntamente com a pilha de *backup*, garantem através do controlo das linhas de alimentação e de *enable* da memória de programa que o seu conteúdo é preservado nos casos de descida significativa ou falha da tensão de alimentação.

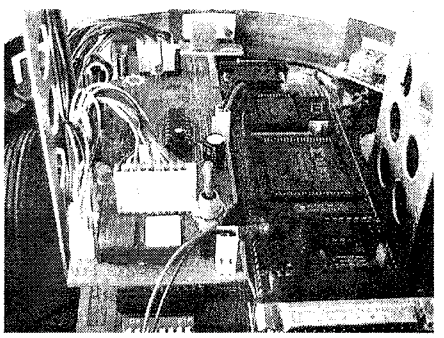


Fig. 1 – Placas de controlo, extensão e interface.

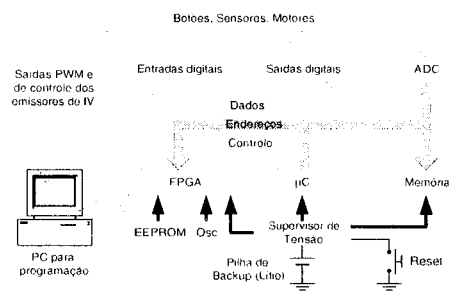


Fig. 2 – Estrutura do hardware de controlo do robot.

A FPGA é utilizada nas seguintes funções (fig. 3):

- Descodificação e latch de endereços;
- Geração dos sinais de controlo para os portos de entrada/saída;
- Geração dos sinais PWM para os motores;
- Obtenção das diversas frequências de relógio do circuito.

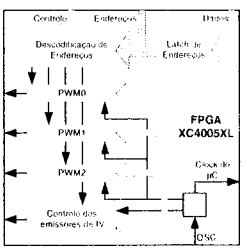
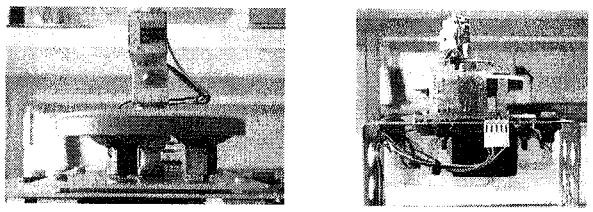


Fig. 3 – Componentes implementados na FPGA.

Para o projecto do circuito na FPGA foi utilizada a ferramenta *Xilinx Foundation Express* para captura do esquemático, implementação e geração do ficheiro de configuração. A programação do robot consiste no carregamento desse ficheiro na FPGA e na transferência do programa para a memória de código do microcontrolador, sendo ambas as operações efectuadas através do porto paralelo de um PC. Os sinais de controlo PWM dos motores e dos leds emissores de IV usam pinos dedicados da FPGA,

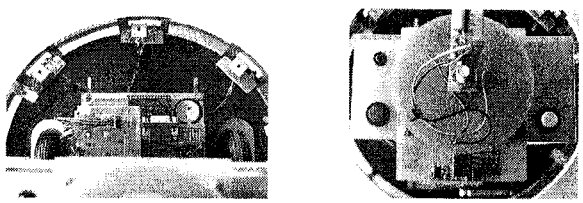
sendo gerados por esta sem a intervenção do microcontrolador. Este interage com a FPGA somente para especificar um novo valor para a velocidade dos motores ou para alterar o estado de activação dos leds de IV. Foi também desenvolvido um sensor de farol rotativo que possui a vantagem de detectar o farol independentemente da posição do robot. Este sensor é constituído por um motor DC e um sensor IV sensível ao sinal do farol, montado sobre uma base rotativa. A ligação eléctrica entre as partes fixa e móvel deste sensor é realizada através de escovas de grafite que deslizam sobre pistas de cobre circulares (fig. 4a)). Para detectar a passagem do sensor por um ponto de referência, foi usado um par emissor-receptor de IV semelhante ao sensor de chão (fig. 4b)).



a) b)

Fig. 4 Sensor de farol: a) sensor de IV e escovas; b) detector de posição e motor.

Na parte inferior do robot encontram-se montados os motores, as rodas, o suporte da bateria e o sensor de chão. Os sensores para detecção de obstáculos encontram-se montados sobre a base do robot, tal como é apresentado na fig. 5a). A sua disposição permite cobrir razoavelmente toda a parte da frente do robot. No entanto, devido à inexistência de sensores na traseira não são executados movimentos para a rectaguarda, somente rotações sobre si próprio. Os botões de Início, Paragem, o Led de Fim e o sensor de farol encontram-se na parte superior do robot (fig. 5b)).



a) b)

Fig. 5 – Vista superior do robot: a) Localização dos sensores de detecção de obstáculos; b) Led de Fim, botões de Início e Paragem e sensor de farol.

III. SOFTWARE

A arquitectura do software desenvolvido para controlo do robot encontra-se representada na fig. 6. A linguagem de programação escolhida foi o C tendo sido utilizado um compilador para a família 8051 da Intel. As acções realizadas pelo robot resultam da arbitragem das acções sugeridas individualmente por cada um dos seguintes três comportamentos: **detecção de armadilhas**, **detecção de obstáculos** e **ir para o farol**. Cada um destes comportamentos possui uma prioridade diferente na determinação da acção a realizar pelo robot. O comportamento "Detecção de obstáculos" possui maior prioridade do que o "Ir para o farol" porque a movimentação do robot no labirinto até à área de chegada deve ser efectuada sem colisões. Por outro lado, o comportamento "Detecção de armadilhas" é o que possui maior prioridade porque em determinadas situações, as acções sugeridas pelo comportamento de detecção de obstáculos podem levar o robot a cair numa armadilha. Esta situação ocorre quando, devido ao ambiente que o rodeia, as ordens emitidas por um comportamento fazem com que o robot execute movimentos oscilatórios sem sair da posição em que se encontra. De notar que esta atribuição de prioridades não constitui perigo de colisão, uma vez que no caso de ser detectada uma armadilha, o robot inicia um movimento rotativo sobre si próprio até encontrar uma zona suficientemente livre por onde possa escapar. De forma sucinta, os módulos representados na fig. 6 possuem as seguintes funções:

- **Entrada/Saída** – disponibiliza macros e funções para leitura/escrita dos portos de entrada/saída existentes na placa de expansão e na FPGA;
- **Sensores** – contém funções para ler todos os sensores do robot colocando os valores em variáveis globais acessíveis aos restantes módulos do programa;
- **Motores** – efectua a gestão dos três motores do robot. Devido à variação da velocidade dos motores responsáveis pelo deslocamento do robot, a sua aceleração é limitada, o que contribui por um lado para a suavização dos movimentos do robot e por outro para um aumento da duração das baterias. Esta limitação é transparente ao utilizador deste módulo. No caso do motor do sensor de farol não é feita qualquer limitação da aceleração, uma vez que roda com velocidade constante;
- **Temporizador** – este módulo é utilizado para estabelecer o tempo do ciclo de processamento do robot. O principal motivo da sua existência é a limitação da aceleração dos motores de forma determinística. O valor que foi utilizado para este tempo, por nos parecer razoável, foi 50ms. Com o microcontrolador a operar a 6MHz, o tempo efectivo de processamento era de 15ms;
- **Gestão de comportamentos** – este módulo é controlado pelo temporizador, efectuando de forma sequencial a activação da leitura dos sensores, a execução dos comportamentos, a invocação do árbitro e a actualização da velocidade dos motores. A sua utilização facilita a alteração do número de comportamentos do robot;
- **Comportamentos (detecção de armadilhas, detecção de obstáculos, ir para o farol)** – cada um destes três módulos recolhe informação dos sensores relevantes, determina a acção a sugerir ao árbitro e coloca-a numa variável global no módulo de gestão de comportamentos para posterior processamento;
- **Árbitro** – este módulo, com base nas acções determinadas pelos vários comportamentos, calcula a acção que o robot deve efectuar de acordo com o esquema de prioridades atrás descrito. Se um comportamento de maior prioridade (ex. detecção de obstáculos) impuser um desvio mas não especificar a direcção, o árbitro recorre à acção sugerida por um comportamento de menor prioridade e extrai-lhe a direcção por forma a tomar a melhor decisão possível;

- **Principal** – é responsável pela inicialização do robot entrando num ciclo "infinito" quando for pressionado o botão de Início. A partir daí o processamento passa a ser controlado pelos módulos de temporização e gestão de comportamentos até que seja detectada uma condição de paragem, como por exemplo, a actuação do botão de Paragem ou a conclusão da prova.

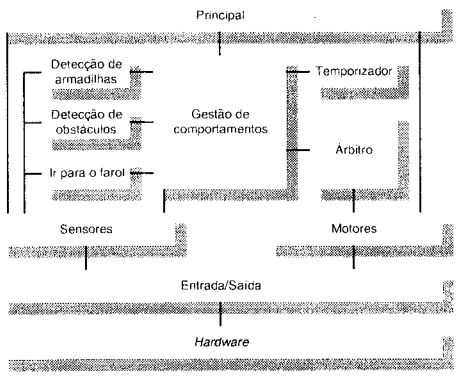


Fig. 6 – Arquitectura do software de controlo.