PROJETO DE UM ROBÔ AUTÔNOMO SEGUIDOR DE ECTOPLASMA

Leonardo Amorim de Araújo

Programa de Graduação em Engenharia Eletrônica, Faculdade Gama
Universidade de Brasília
Gama, DF, Brasil
email: leonardoaraujodf@gmail.com

RESUMO

Este documento apresenta como foi realizado o projeto de um robô autônomo seguidor de ectoplasma (substância fluídica que os fantasmas deixam aonde passam) cuja função é seguir, encontrar e capturar fantasmas. Este mostra de forma detalhada como foi realizado o projeto e o teste experimental. O microcontrolador utilizado foi o MSP430[1] modelo g2553.

1. INTRODUÇÃO

A análise do ectoplasma pode ser realizada de diversos níveis, seja utilizando dispositivos com hardware programável, como FPGAs, ou via software através de um microcontrolador. A vantagem de se utilizar uma FPGA neste caso é que o robô ganha muito desempenho e velocidade de processamento, pois o projetista cria um hardware dedicado para a tarefa especificada, utilizando a metodologia das máquinas de estado ou Projeto RTL[2], por exemplo. Via de regra, parece ser a solução mais interessante, porém, como já é da natureza do problema, os sensores possuem saída analógica, ou seja, fornecem tensões que variam em uma faixa (0 à 3V, no caso da família CMOS). Sendo assim, seria necessário que a FPGA possuísse internamente um conversor A/D que pudesse converter os dados analógicos através da amostragem e quantização. Apesar de ser possível, esta tarefa torna o projeto extremamente árduo de se realizar dentro do tempo requerido, e é nesse aspecto que um microcontrolador faz mais sentido. O microcontrolador MSP430 utilizado neste projeto possui um conversor A/D de 10 bits[1], que lê valores em uma faixa de 0 à 3V, tendo portanto uma faixa de $3V/1024\,\text{valores} = 3\text{mV}$ em cada passo lido. Esta precisão grande torna-o útil para a análise de dados analógicos que possuem variação bem pequena de tensão (que é o caso do ectoplasma dos fantasmas).

Com os valores obtidos de cada sensor à uma taxa de amostragem relevante para o ataque e destruição do fantasma, é possível programar o microcontrolador através da linguagem C para que este possa setar ou não as saídas e realizar as tarefas especificadas pelo problema, tudo isto dentro do tempo de amostragem do problema.

O microcontrolador MSP430 já possui diversas definições prontas para serem utilizadas caso seja necessário um conversor A/D, um relógio (clock) bem exato e além disso comunicação serial (caso, por exemplo, fosse necessário saber por GPS onde o robô se encontra no momento, ou por WIFI).

2. EXPERIMENTO

Para mostrar que é possível realizar a proposta requerida, os sensores de ectoplasma foram simulados utilizando LDRs, que são compatíveis pois estes variam sua resistência elétrica conforme a incidência de luz. As saídas para acionamento dos motores esquerdo, direito e da arma de prótons, foram simuladas utilizados LEDS.

2.1. Fluxograma do Projeto

Como o projeto foi realizado com um microcontrolador, não é relevante usar máquina de estados pois as ações não são realizadas após pulsos de clock, e sim um fluxograma que apresenta as ações tomadas.

2.2. Conversão Analógico-Digital

A figura 1 apresenta como foi realizada a conversão analógico-digital usando o MSP. No código apresentado na referência, as funções *Setup_ADC* e *Read_ADC()* tem o objetivo de realizar a configuração inicial do Conversor A/D e a leitura dos valores analógicos nos pinos P1.0, P1.1, P1.2 e P1.3, respectivamente. Os valores convertidos serão armazenados nas variáveis *ADC[0]*, *ADC[1]*, *ADC[2]* e *ADC[3]*. Posteriormente, na função *Ler_Sensores()*, os valores destas variáveis serão atualizadas para que os dados obtidos de tensão sejam tratados, de forma que serão atribuídos os valores lidos às variáveis:

- *SENSOR_FRONTAL = ADC[3]*
- *SENSOR_TRASEIRO = ADC[2]*
- *SENSOR_ESQUERDO = ADC[1]*

- *SENSOR_DIREITO = ADC[0]*

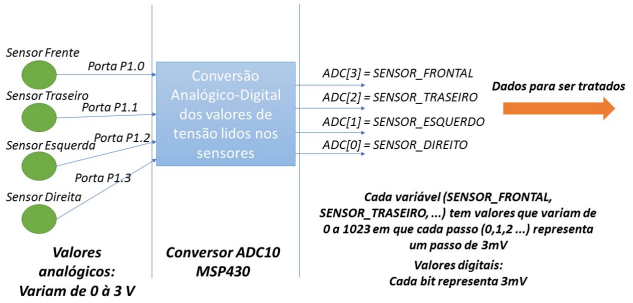

Fig. 1. Fluxograma mostrando como foi realizado a conversão analógico-digital

2.3. Ações a serem realizadas após a conversão

A figura 2 apresenta o fluxograma das ações tomadas pelo microcontrolador após a conversão analógico-digital ser realizada. Na função *Ler_Sensores()*, todas estas ações são realizadas. É importante destacar que as saídas estão declaradas nas seguintes portas do microcontrolador:

- P1.4 - MOTOR ESQUERDO
- P1.5 - MOTOR DIREITO
- P1.6 - ARMA

onde o valor lógico 1 representa acionamento e valor lógico 0, desligamento. A função *Ler_Sensores()* segue o seguinte procedimento:

- Caso a média dos valores dos sensores lidos seja menor que 20, ou seja, menor que 20*3 mV = 60 mV, isto quer dizer que a queda de tensão nos LDRs foi grande, significando que a incidência de luz está alta sobre os sensores, e em todos ao mesmo tempo. Desta forma, é possível que um fantasma esteja muito próximo do robô e deve-se atacar. Chama-se a função *Selecao_Saida()* que seleciona qual saída será acionada. Nesta hora, o robô parará, a arma será acionada, será esperado 1 segundo para o disparo, e mais 9 segundos para que a arma de prótons possa estabilizar, conforme requerido pela proposta de projeto. Após isto, será realiza mais uma leitura dos sensores, e será verificado se ainda há existência do fantasma, realizando o mesmo procedimento em loop descrito.
- Caso o valor do sensor frontal seja maior que a média dos valores dos sensores lidos mais um parâmetro de média, que é definido pelo projetista (por exemplo, existem ectoplasmas que são difíceis de se detectar, então o parâmetro de média deve ser baixo, já para outros fantasmas, aumenta-se o valor desta constante) o robô deve seguir em frente, pois o sensor frontal detecta ectoplasma à certa distância.
- Caso o valor do sensor esquerdo seja maior que a média dos valores dos sensores lidos mais o parâmetro de média, o robô deve virar a esquerda, desligando o motor da direita.
- Caso o valor dos sensores direito ou traseiro sejam maiores que média dos valores lidos mais o parâmetro de média, o robo deve virar aà direita, desligando o motor da esquerda. Veja que, quando o sensor traseiro estiver acionado, já que o robô não pode mover para trás, é necessário que este gire 180º, virando à direita totalmente, e assim o sensor frontal será acionado, e desta forma, este seguirá em frente. Neste ponto, pode existir uma falha, pois o fantasma se for rápido, pode enganar o robô, devendo-se posteriormente criar uma arma que seja utilizada com servo-motores, por exemplo.
- Se nenhum sensor for acionado, ou seja, não existir presença de ectoplasma, o robô irá seguir reto. Os projetistas podem posteriormente mudar esta condição.

Em cada caso, após o acionamento dos motores, espera 100 ms para se realizar uma nova conversão. Isto é interessante para estabilidade do motor, pois ele pode começar a tremer muito, caso os valores mudem com rapidez (no caso do acionamento da arma, espera-se 10 segundos, utilizado a função *Atraso()* que utiliza o TimerA do MSP430).

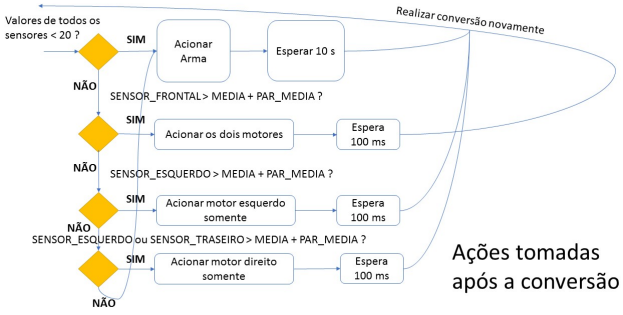

Fig. 2. Fluxograma mostrando as ações tomadas após a conversão

2.4. Material Utilizado

A tabela 1 apresenta todos os dispositivos e componentes utilizados no projeto.

Tabela 1. Materiais Utilizados no Experimento

Quantidade	Material
4	Resistores de de $10,00 \pm 0,50$ kΩ (1/4 W)
3	Resistores de $1,00 \pm 0,05$ kΩ (1/4 W)
4	Resistores Dependentes de Luz - LDRs
1	Mini-Protoboard
3	LEDs
1	Microcontrolador MSP 430 modelo G2553

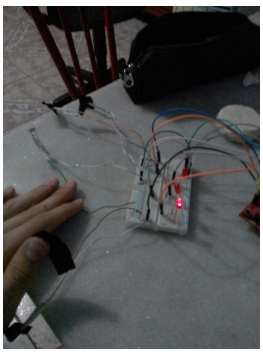

Fig. 3. Sensor Esquerdo sem fita-isolante, motor direito acionado - Representação com LEDs

3. RESULTADOS

Como nesta simulação foram utilizados LDRs, a luz ambiente interfere na variação de tensão do resistores. Logo em cada resistor, foi colocado um pedaço de fita isolante para que estes se comportem como se estivessem "na escuridão", e a tensão que será lida será próxima do máximo, ou seja, 3V. A figura 3 apresenta a simulação utilizando LEDs para representar os motores e a arma. Neste caso, quando o LDR que representa o sensor esquerdo, que fica na porta P1.2 ficou sem a fita isolante, o motor esquerdo foi acionado (LED Aceso), ou seja, a saída P1.4 foi acionada no MSP. Além disso, veja que, os dois outros LEDS ficaram desligados, e estes representam a arma e o motor direito. Já na figura 4, pode-se visualizar que ao deixar o LDR que representa o sensor direito (ligado à porta P1.3) ficou sem fita isolante, e desta forma o motor direito foi acionado (porta P1.5), e o os outros dois LEDs ficaram desligados. Da mesma forma, quando o sensor traseiro (ligado à porta P1.1) ficou sem fita isolante, somente este LED ficou aceso. Na figura 5, pode-se visualizar que quando o LDR que representa o sensor frontal foi acionado (porta P1.0), os LEDs que representam os dois motores foram acionados (Portas P1.4 e P1.5). E por fim, na figura 6, pode-se visualizar que, quando todos os LDRs estão sem fita isolante, o que representa que todos os sensores estão acionados, o LED que representa a arma dispara. Pode-se verificar também que o tempo de acionamento foi realmente de 1 segundo, e o MSP esperou 10 segundos para realizar outro disparo (acionou o led).

Fig. 4. Sensor Direito sem fita-isolante, motor direito acionado. O mesmo ocorre se o sensor traseiro é acionado (fica sem fita isolante) - Representação com LEDs

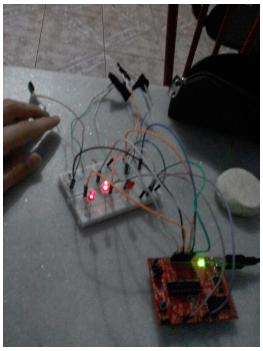

Fig. 5. Sensor Frente sem fita-isolante, motor direito e esquerdo acionado - Representação com LEDs

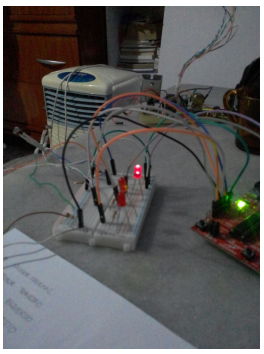

Fig. 6. Todos os Sensores Acionados, Arma disparada - Representação com LEDs

4. DISCUSSÃO E CONCLUSÕES

Pode-se verificar que nesta proposta, é possível realizar o projeto do robô através do microcontrolador MSP430 e que este consegue com facilidade atender aos requisitos da proposta em questão, conseguindo inclusive, incorporar mais protótipos, como um módulo GPS ou WIFI para descobrir a localização do robô. Verificou-se que, com os LDRs utilizados como sensores, os motores e a arma foram acionadas praticamente com precisão. Apesar disto, algumas pontos interessantes podem ser melhorados, como o fato de que o robô pode se mover pra frente e para trás, evitando que este tenha que girar 180^o para acompanhar um fantasma que aciona o sensor traseiro. O projeto com microcontrolador pode ser interessante para um futuro protótipo utilizando FPGA, pois pode-se utilizar a mesma lógica para criação de um hardware[2] que realiza as tarefas que o MSP realiza.

5. REFERENCIAS

[1] J. H. Davies, *MSP430 microcontroller basics*. Oxford: Newnes, 2008.

[2] F. Vahid, *Digital design, with RTL design, VHDL, and Verilog*. Hoboken, NJ: Wiley, 2011.