Subconjunto de Instruções do MIPS para Suporte à Robótica

Vicente S. Cruz, Henrique C. Freitas, Philippe O. A. Navaux
Grupo de Processamento Paralelo e Distribuído
Instituto de Informática
Universidade Federal do Rio Grande do Sul
{vscruz,hcfreitas,navaux}@inf.ufrgs.br

Resumo

This paper will present a proposal of MIPS Instruction Set Architecture (ISA), through inclusion of new instructions that supports robot control. They will be implemented using the architecture description language ArchC. Finally, will be shown that there are advantages to include these instructions in the MIPS, and the results will show a favorable performance for MIPS-robot ISA.

1. Introdução

Os robôs vêm substituindo cada vez mais o homem em diversas atividades, muitas delas bastante arriscadas [1]. Com isso, é necessário que o robô otimize o tempo que dispõem para poder realizar suas atividades. Como as pesquisas realizadas nessa área, para atingir esse objetivo, demandam respostas rápidas, certos detalhes físicos podem ser omitidos, a um primeiro momento do projeto. Dessa forma, esse artigo tem por objetivo utilizar a ferramenta ArchC [2] para expandir o conjunto de instruções do MIPS, de tal forma que auxilie os robôs na realização dos cálculos necessários para efetuar seus movimentos, de forma rápida e eficiente. Essa expansão será realizada em cima da descrição do MIPS de 32 bits monociclo para ArchC [2], implementada pela equipe desenvolvedora dessa ADL, e será chamada de MIPS_Robot.

Após o projeto das novas instruções, será feito uma comparação do tempo de execução, acessos à memória e acessos ao banco de registradores dessas duas arquiteturas (MIPS e MIPS_Robot). Será usado como aplicação a realização dos cálculos necessários para movimentar o braço de um robô.

2. Trabalhos Relacionados

Fu e Wilson Jr. [3], assim como Paliouras [4], propõem novos chips que realizam as operações de seno e cosseno de forma otimizada. Em [3] as operações são realizadas utilizando uma variação do algoritmo de CORDIC [5], gerando circuitos menores e mais velozes. Já em [4] também é realizada essas operações, porém usando um método baseado na técnica de interpolação de polinômios de segunda ordem, obtendo um ganho de 40% de redução do espaço utilizado na memória.

A proposta deste artigo é a simulação não apenas das instruções seno e cosseno, mas também instruções que utilizem o resultado dessas operações para efetuar os cálculos de movimento dos braços e incluí-los no chip proposto.

3. Proposta do ISA

Baseado nas análises, abaixo apresentadas, definiram-se sete novas instruções que serão inseridas na arquitetura MIPS. São elas: seno, cosseno, produto escalar, translação, rotação em X, rotação em Y e rotação em Z, e seus mnemônicos são: SIN, COS, SCL, TRS, RTX, RTY e RTZ.

As instruções de seno e cosseno são necessárias, para auxiliar nas operações de rotação. Já a operação do produto escalar é necessária por ser útil na determinação da orientação de um objeto. Por fim, as operações de translação, rotação em X, Y e Z são as principais para manipular os robôs propriamente ditos.

Seguindo a especificação do formato das instruções do MIPS [6], definiu-se as novas instruções com sendo Tipo-R (Registrador):

SIN *rt, rs*	COS *rt, rs*	SCL *rd, rs, rt*	TRS *rd, rs, rt*
RTX *rd, rs, rt*	RTY *rd, rs, rt*	RTZ *rd, rs, rt*	

As operações de SENO e COSSENO são realizadas através da chamada da função sin() e cos() da biblioteca <math.h>. O Produto Escalar é realizado pela soma dos produtos dos componentes dos vetores informados em rs e rt. Já a operação da instrução

Translação é definida pela soma pontual entre o ponto dado e os compontentes do vetor. Por fim, segue a operação de RTX, (para RTY e RTZ, as operações são bastante semelhantes):

RB_float[*rd*] = RB_float[*temp*];
RB_float[*rd+1*] = RB_float[*rt*]*RB_float[*temp*+1] -
RB_float[*rs*]*RB_float[*temp*+2];
RB_float[*rd+2*] = RB_float[*rs*]*RB_float[*temp*+1] +
RB_float[*rt*]*RB_float[*temp*+2];

Onde *rs* é o seno de um ângulo e *rt* é o cosseno, *rd* possui o ponto inicial antes da operação e o final após a execuçao da mesma. *Temp* armazena temporariamente o ponto de origem e RB_float é explicado a seguir.

As novas instruções geram valores em ponto-flutuante (PF), mas a descrição do MIPS trata apenas de valores inteiros. Dessa forma, criou-se o banco de registradores RB_float, que armazena valores em PF.

4. Resultados

Para verificar a eficácia das novas instruções, foi desenvolvida uma aplicação que realiza os cálculos necessários que um braço de um robô partindo do ponto inicial de coordenadas [2 3 4], translade 3 unidades na direção dos 3 eixos, rotacione 3 radianos em torno de X e depois rotacione 2 radianos em torno de Z. Foram feitas duas implementações, em assembly, dessa aplicação: uma para o MIPS e outra para o MIPS_Robot, ambos rodando sob a mesma freqüência (50 MHz). Foram analisados os tempos de execução, quantidade de instruções executadas, quantidade de acessos à memória, quantidade de acesso aos registradores (inteiros e PF).

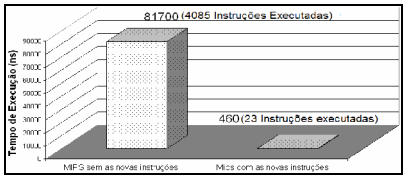

Figura 1 – Tempo de execução

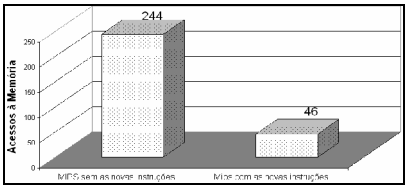

Figura 2 – Acessos à memória

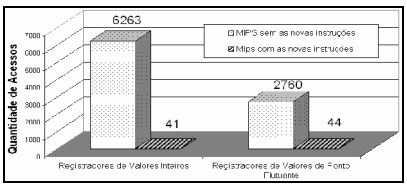

Figura 3 – Acessos aos registradores

Pela análise dos gráficos, nota-se o quanto é vantajoso incluir as novas instruções no ISA do MIPS monociclo. Em relação ao Tempo de execução (Figura 1), o MIPS-Robot executou em apenas 0,56% do tempo total de execução do MIPS. Já em termos de acesso a memória (Figura 2), o MIPS-Robot acessou-a apenas 18,85% das vezes em que o MIPS acessou-as. Por fim, o total de acesso ao Banco de Registradores (Figura 3) do MIPS-Robot em relação ao MIPS foi de 0,65% para os registradores de valores inteiros, e de 1.59% para os registradores de valores em PF. Por fim, é importante observar que a simulação não ocasiona nenhum overhead para a obtenção desses resultados.

5. Conclusões

Os testes mostraram que há uma vantagem significativa ao se incluir as novas instruções no processador. No entanto, é importante ressaltar que os resultados foram obtidos sob uma mesma freqüência de simulação, o que ocasionou na utilização de área e dissipação maiores.

Como trabalhos futuros, planeja-se incluir essas novas instruções no MIPS com *pipeline* e verificar se as vantagens obtidas nesse artigo também se refletem nessa nova proposta. Novos estudos também serão realizados para verificar os impactos decorrentes deste novo projeto, principalmente em relação à área ocupada, consumo de potência e freqüência limite de operação. Estes novos resultados podem influenciar na porcentagem de ganho de desempenho do MIPS-Robot.

Referências

[1] K. Yokoi, et al., *Application of Humanoid Robots for Teleoperation of Construction Machines*, IEEE/RSJ IROS Workshop on Exploration towards Humanoid Robots Applications, 2001.

[2] S.Rigo, et al., *ArchC: A SystemC-Based Architecture Description Language*, International Symposium on Computer Architecture and High Performance Processing, pp.66-73, October 2004.

[3] D. Fu and A. N. Willson Jr, *A High-Speed Processor for Digital Sine/Cosine Generation and Angle Rotation*, Conference on Signals, Systems & Computers, 1998.

[4] V. Paliouras, et al., *A floating-point processor for fast and accurate sine/cosine evaluation*, IEEE Trans. on Circuits and Syst. – II., vol. 47, pp.441-451, May 2000.

[5] R. Andraka, *A survey of CORDIC algorithms for FPGA based computers*, Sixth Inter. Symp. on Field Programmable Gate Arrays, pp. 191-200, February 1998.

[6] D. A. Patterson, J. L. Hennessy, *Organização e Projeto de Computadores: A Interface Hardware/Software*, Editora Campus, 3ª Edição, 2005.