

Com o progresso científico-tecnológico da sociedade em que vivemos, o uso de sistemas embarcados (*embedded*) tem sido cada vez mais freqüente. Maneiras de diminuir o consumo de potência e o custo, bem como aumentar o desempenho, têm sido pesquisadas. Infelizmente, há poucas maneiras de melhorar todos os três parâmetros. É difícil encontrar, inclusive, maneiras de melhorar apenas um deles sem afetar negativamente, algum outro. A técnica que o artigo propõe consegue melhorar os três parâmetros.

Basicamente, a idéia é encontrar o conjunto de instruções mais freqüentes no código e colocá-las em um banco de registradores especial (IRF). Além do novo banco de registradores, foi adicionado uma tabela de constantes (IMM). A IMM (de entradas de 32 bits) contém as constantes mais presentes no código e são utilizadas para evitar o problema de ser necessário duas entradas para instruções que diferem apenas pelo imediato. A utilização desta técnica acaba compactando o código, aumentando a eficiência do estágio de *fetch* do pipeline do processador.

Para ilustrar o funcionamento da técnica, os autores do artigo apresentam uma versão modificada do processador MIPS. As alterações mais drásticas na ABI foram a inclusão de um novo formato de instrução e a modificação do formato das instruções: o campo *shamt* das instruções de formato R é utilizado, agora, para referenciar uma instrução na IRF (a entrada zero da IRF é um *nop* que não é executado). Além disso, o campo *imm* das instruções do tipo I foi encolhido em 5 bits (para resolver este problema, o campo *imm* da instrução *lui* agora carrega constantes de 21 bits). Assim, com estas alterações é possível buscar duas instruções por *fetch*. Este tipo de compactação é chamado *loosely packed*.

Mas a modificação mais drástica foi a inclusão de novas instruções que permitem até cinco referências ao IRF. Estas instruções, cujo formato é chamado T, são ditas *tightly packed*. É possível haver até cinco referências ao IRF em instruções do tipo T, ou 4 instruções e um parâmetro (um parâmetro é uma referência à IMM), ou ainda 3 instruções e dois parâmetros. O uso desta tabela para representação de constantes, ao invés de campos de immediatos nas próprias instruções representam uma melhora significativa no total de instruções que são colocadas no IRF.

Para aumentar ainda mais a probabilidade de utilização da IRF, os autores propõe o uso de *positional registers*. Estes registradores não são referências aos registradores físicos da arquitetura, mas correspondem aos registradores utilizados em instruções anteriores à instrução atual.

Por fim, as instruções de desvio incondicional não podem ser facilmente compactadas, dado que o *offset* utilizado pode utilizar potencialmente todos os bits. Para solucionar isso, a técnica codifica os desvios cujo destino é representável em 5 bits (*offsets* variando de -16 a 15) como um desvio condicional que compara um registrador com ele mesmo, o que possibilita o uso de instrução do IRF com parâmetros.

Para fazer a geração de código para este processador alterado, é necessário um compilador especial. Este compilador deve detectar quais as instrução são candidatas a serem colocadas no IRF e fazer a alocação de registradores para estas instruções. A determinação de candidatos à alocação pode ser feita estaticamente ou com informação de *profiling*. Obviamente, a alocação utilizando informação sobre a execução do programa é mais eficiente. Entretanto, a alocação estática pode obter resultados razoáveis com o uso de heurísticas (e.g., favorecendo instruções de laços). O artigo apresenta o algoritmo utilizado para fazer a otimização.

Os resultados obtidos por esta técnica se mostraram deveras animadores. Devido à natureza das modificações, dificilmente o código utilizado nesta nova arquitetura seria maior que um código para a arquitetura simples. A técnica também reduziu o consumo de potência do processador (o que era esperado). O mais impressionante, contudo, foi a diminuição do tempo de execução de alguns dos programas do benchmark (provavelmente devido ao fato de ser necessário menos acesso à memória).

Os autores fazem ainda considerações sobre o aumento / diminuição do IRF. Em geral, o desempenho aumenta para IRF's maiores. Entretanto, a partir de 512 registradores de instrução ocorre uma degradação da performance da aplicação já que torna-se impossível representar mais que 2 instruções em uma instrução.

Para ambientes multi-processo, é possível utilizar uma IRF por processo, sendo necessário apenas o carregamento dos registradores com as instruções de cada processo. Entretanto é necessário salvar (e restaurar) os especificadores dos registradores posicionais.