2004

# Code compression based on operand-factorization for VLIW processors

Montserrat Ros
*University of Queensland*, montse@uow.edu.au

Peter Sutton
*University of Queensland*

## Publication Details

M. Ros & P. Sutton, "Code Compression Based on Operand-Factorization for VLIW Processors," in Proceedings of the Data Compression Conference, 2004, p. 559.

# Code compression based on operand-factorization for VLIW processors

**Abstract**
Code compression techniques have been devised as a way of battling large code. Code compression algorithms usually require specific techniques to maintain the integrity of the program and ensure its functionality. RISC processors have been the main focus for code compression techniques but VLIW (Very Long Instruction Word) processors are now being considered in this area.

**Keywords**
code, vliw, processors, factorization, operand, compression

**Disciplines**
Engineering | Science and Technology Studies

**Publication Details**
M. Ros & P. Sutton, "Code Compression Based on Operand-Factorization for VLIW Processors," in Proceedings of the Data Compression Conference, 2004, p. 559.

# Code Compression Based on Operand-Factorization for VLIW Processors

Montserrat Ros, Peter Sutton
School of Information Technology and Electrical Engineering
The University of Queensland
St Lucia QLD 4072
{ros,p.sutton}@itee.uq.edu.au

Code compression techniques have been devised as a way of battling large code. Code compression algorithms usually require specific techniques to maintain the integrity of the program and ensure its functionality. RISC processors have been the main focus for code compression techniques but VLIW (Very Long Instruction Word) processors are now being considered in this area.

This paper presents three code compression algorithms; the first based on dictionary methods for entire instructions, the second based on instructions factorized into op-code/operand pairs and the third based on operand factorization applied to instruction words. Codewords in all methods are byte-aligned. The first encoding scheme used is a dictionary compression method which compresses the original program by replacing every instruction with its corresponding reference into the dictionary.

The second compression algorithm exploits the facts that 1) the entire op-code space is usually not used and 2) operand patterns are often common between different instructions. This algorithm first classifies instructions into one of several instruction classes. Then, two dictionaries are generated for each instruction class and the instructions are compressed using these.

In an attempt to parallelize the previous compression scheme, the third compression scheme groups instructions into 8-instruction words and the corresponding instruction/operand bits from all 8 instructions are grouped together to form dictionary entries. In this scheme, we are faced with multiple dictionaries to choose a dictionary word from. The class selector bits for each of the 8 instructions are added to the codeword.

The MediaBench Suite was chosen as an appropriate set of benchmark programs, compiled for maximum code optimization on the TI TMS320C6x and then compressed. Compression Ratios (defined as the ratio of compressed code to uncompressed code) of 81.5%, 68.3% and 84.7% are reported for the three compression schemes.

Instruction Factorization was found to be the most efficient compression scheme, though decompression is done sequentially. Although this technique may be very advantageous to single-issue processors, it is detrimental to the time cost of the decompression of instruction words for VLIW processors, particularly when many instructions are scheduled for simultaneous execution. Operand Factorization across instruction-words allows decompression to be parallelized for instructions in the same instruction word, however this is at a cost to compression ratio.

Further work can be done on the application of other compression algorithms to the separated instruction/operand streams and investigation into the efficiency of these schemes on other platforms, particularly other VLIW processors.