ABSTRACT

EFFICIENT EXECUTION OF COMPRESSED PROGRAMS

by

Charles Robert Lefurgy

Chair: Trevor Mudge

Code compression is the technique of using data compression to reduce the program memory size for memory-limited, embedded computers. For system-on-a-chip designs, this reduces the system die area which lowers die cost. After compilation, the binary (native code) program is compressed and stored in the embedded system. At runtime, the compressed program is incrementally decompressed and executed. While compressed programs have better code density, their performance is typically lower because additional effort is required to decompress the instruction stream. This dissertation presents methods to improve the performance of compressed programs.

Decompression overhead can be minimized by using special-purpose hardware. This dissertation analyzes IBM's CodePack decompression algorithm and proposes optimizations for it. The optimized decompressor can often execute compressed programs faster than the original native program. The performance benefit of using fewer memory transactions to fetch compressed instructions surpasses the small decompression overhead. Therefore, code compression improves performance as well as code density.

The decompression hardware can be largely replaced with software. The benefits of software decompression are greater design flexibility, reduced hardware complexity, reduced die area, and reduced cost. However, software decompression is much slower than hardware decompression. On a 5-stage pipelined embedded processor with a 4KB instruction cache, CodePack programs execute 1.3 to 27.0 times slower than native programs and reduce program memory die area (instruction cache and main memory) by 26% to 41%. This dissertation proposes instruction set support to enable efficient software-managed decompression. In addition, it explores two software optimizations, hybrid programs and memoization, to improve the execution time of compressed programs by reducing the compression. Hybrid programs contain both native and compressed code to reduce the number of times the decompressor is invoked. Memoization is a dynamic optimization that caches recent decompression results to also avoid invoking the decompressor. Opti-

mized compressed programs that reduce die area 10% to 33% execute only 1.00 to 1.22 times slower than native code. In addition, loop-oriented (multimedia) programs are nearly as fast as native code.