Indo-Iranian Journal of Scientific Research (IIJSR)
(Peer Reviewed Quarterly International Journal) Volume 2, Issue 4, Pages 31-36, Oct-Dec 2018

A Study on Algorithm for Compression and Decompression of Embedded Codes using Xilinx

M.Mohana Soundarya[1], Dr.S.Jayachitra[2]

[1]*PG Scholar, Department of ECE, Vivekanandha College of Engineering for Women, Namakkal, India.*
[2]*Professor, Department of ECE, Vivekanandha College of Engineering for Women, Namakkal, India.*

Article Received: 29 May 2018 *Article Accepted: 22 September 2018* *Article Published: 06 October 2018*

ABSTRACT

Engineers should take into account performance, power consumption, and value once coming up with embedded digital systems; conjointly memory may be a key think about such systems. Code compression may be a technique employed in embedded systems to cut back the memory usage. Bit Mask-based code compression may be a modified version of dictionary-based code compression. The essential purpose of Bit Mask is to record mismatched values and their positions to compress a bigger range of directions; it may be used solely or incorporated with the reference instructions to rewrite the code words. I introduce two formulas. The primary is lexicon primarily based formula to limit the code word length of high frequency directions and second is novel lexicon choice formula to cut back the compression magnitude relation. This methodology will improve the performance of the decompression engine while not poignant the compression ratio (CR).

Keywords: Compression ratio, Dictionary based code compression (DCC).

1. INTRODUCTION

Embedded systems have become an essential part of everyday life, and are widely used worldwide. Embedded systems must be cost effective, and memory occupies a substantial portion of the entire system. The complexity and performance requirements for embedded programs grow rapidly. Thus, reducing the code size and providing a simple decompression engine are both challenges when applying code compression to embedded systems. Dictionary-based code compression (DCC) is commonly used in embedded systems, because it can achieve an efficient CR, possess a relatively simple decoding hardware, and provide a higher decompression bandwidth than the code compression by applying lossless data compression methods. In this paper, various steps in the code compression process were combined into a new algorithm to improve the compression performance (including the CR) with a smaller hardware overhead. Based on the BitMask code compression (BCC) algorithm , a small separated dictionary is proposed to restrict the code word length of high-frequency instructions, and a novel dictionary selection algorithm is proposed to achieve more satisfactory instruction selection, which in turn may reduce the average CR.

2. RELATED WORKS

Wei Jhih Wang and Chang Hong Lin (2016) introduced a Code Compression for Embedded Systems Using Separated Dictionaries. In this paper a small separated dictionary and variable mask numbers were used with the Bitmask algorithm to reduce the code word length of high frequency instructions. In addition a novel dictionary selection algorithm was proposed to increase the mismatch rates. The fully separated dictionary method was used to improve the performance of decompression engine without affecting the compression ratio (CR). Based on the experimental results, the proposed method can achieve a 7.5% improvement in the CR with nearly no hardware overhead.

Wanderson Roger Azevedo Dias (2013) introduced A New Code Compression Algorithm and its Decompressor in FPGA-Based Hardware. In this paper a new method of code compression for embedded systems called as Compressed Code using Huffman based Multi-Level Dictionary (CC-MLD). This method applies two compression techniques and it uses the Huffman code compression algorithm. A single dictionary is divided into two levels and it is shared by both techniques. This method reduces code size up to 30.6%. The performance of the simulations using applications from MIBench and it had used four embedded processors (ARM, MIPS, PowerPC and SPARC).

Sreejth K Menon (2012) introduced An Efficient Tool chain for Analyzing Tradeoffs of Code Compression Schemes in Embedded Processors. The code compression has emerged as a promising solution to meet the challenges of rapidly increasing application size and of scare memory resources in embedded systems. In past two decades a large number of compression algorithms have been proposed and implemented to overall code density on a wide variety of architectures. The problems are addressed with an efficient tool chain capable of analyzing different code compression schemes and evaluating the tradeoffs. The tool chain consists of a frontend framework that works with different compression or decompression schemes and a backend with high level synthesis and logic synthesis tools.

Xiaoke Qin (2011) introduced a Decoding Aware Compression of FPGA Bit streams. This paper proposes a novel decode aware compression technique to improve both compression and decompression and efficiencies. The three major contributions of this paper are: 1) smart placement of compressed bit streams that can significantly decrease the overhead of decompression engine; 2) selection of profitable parameters for bit stream compression and 3) efficient combination of bitmask based compression and run length encoding of repetitive patterns. The proposed technique outperforms the existing compression approaches by 15%, while the decompression hardware for variable length coding is capable of operating at the speed closest to the best known field programmable gate array based decoder for fixed length coding.

Usha S. Mehta and Kankar S. Dasgupta (2010) introduced the Hamming Distance Based Reordering and Column wise Bit Stuffing with Difference Vector. In this paper a new scheme named Hamming Distance Based Reordering and Column wise Bit Stuffing with Difference Vector (HDR-CBS- DV), which can be used with any run length based code technique for better compression ratio. Four techniques have been applied in this scheme: Selection of first vector, Hamming Distance Based Reordering, Column wise Bit Stuffing and Difference Vector. Instead of directly applying any known run length code like Golomb, Frequency Directed Run Length (FDR), Extended FDR (EFDR), Modified FDR (MDFR) or Shifted Alternate FDR (SAFDR) to given test set, if the proposed scheme to test set prior to applying the run length base code, the compression obtained is improved drastically. For few cases, it requires an extra XOR gate and feedback path only. The proposed scheme can be easily integrated into the existing industrial flow.

Jayasanthi Ranjith (2010) introduced a VLSI Implementation of Single Bit Control System Processor with Efficient Code Density. This paper describes the efficient code density in System on Chip(SOC) and mixed signal applications. Advanced process technology enables hundreds of million transistors to be integrated onto a single chip, making SOC designs more feasible than ever. Such SOC chips usually consist of various components such as analog to digital (ADCs) and digital to analog (DACs) converters, Phase Locked Loops (PLLs), random logic memory, processors and so on. Delta Sigma modulator is frequently used in the conversion of signals from analog to digital form into bit streams at very high sampling rate. It may also reduce the power consumption, single memory consumes a significant amount of control system processors power.

Talal Bonny and Jorg Henkel (2009) introduced LICT: Left- uncompressed Instructions Compression Technique to improve the Decoding the Performance of VLIW Processors. In this paper the performance of decoding compressed instructions improved by using novel compression technique (LICT: Left-uncompressed Instruction Technique) which can be used in conjunction with any compression algorithm. Furthermore, we adapt a new code compression approach called Burrows-Wheeler (BW) which has been used before in data compression. It significantly reduces the code size compared to state-of- the-art approaches for VLIW processors. Using the LICT in conjunction with the BW algorithm improves the performance explicitly (2.5x) with little impact on the compression ratio (only 3% compression ratio loss).

Xiaoke Qin and Prabhat Mishra (2009) introduced an Efficient Placement of Compressed Code for Parallel Decompression. This paper combines the advantages of both approaches by introducing a novel bit stream placement method. The contribution of this paper is a novel code placement technique to enable parallel decompression without sacrificing the compression efficiency. The proposed technique splits a single bit stream (instruction binary) fetched from memory into multiple bit streams, which are then fed into different decoders. As a result, multiple slow-decoders can work simultaneously to produce the effect of high decode bandwidth.

Talal Bonny (2008) introduced the Efficient Code Compression for Embedded Processors. In this paper a novel, hardware-supported approach. Besides the code, also the lookup tables (LUTs) are compressed, that can become significant in size if the application is large and/or high compression is desired. The scheme optimizes the number and size of generated LUTs to improve the compression ratio. To show the efficiency of the approach there are two compression schemes applied: “dictionary-based” and “statistical”. An average compression ratio of 48% (already including the overhead of the LUTs). Thereby, the scheme is orthogonal to approaches that take particularities of a certain instruction set architecture into account. The evaluations using a representative set of applications and have applied it to three major embedded processor architectures, namely ARM, MIPS, and PowerPC.

Seok-Won Seong (2007) introduced An Efficient Code Compression Technique using Application Aware Bitmask and Dictionary Selection Methods. Memory plays a crucial role in designing embedded systems. A larger memory

can accommodate more and large applications but increases cost, area, as well as energy requirements. Code compression techniques address this problem by reducing the size of the applications. While early work on bitmask-based compression has proposed several promising ideas, many challenges remain in applying them to embedded system design. This paper makes two important contributions to address these challenges by developing application-specific bitmask selection and bitmask-aware dictionary selection techniques. These techniques are applied for code compression of TI and MediaBench applications to demonstrate the usefulness of our approach.

TABLE 1: Comparison between block matching methods

SL. NO	PAPER TITLE	ADVANTAGES	DRAWBACKS
1	An Efficient Code Compression Technique using Application-Aware Bitmask and Dictionary Selection Methods	Improve code compression efficiency without introducing any decompression overhead.	It is very difficult to determine the optimal mask set prior to compression.
2	Efficient Code Compression for Embedded Processors	Improves further important design parameters like power consumption and performance.	One of the major problems is that the tables can become large in size.
3	Efficient Placement of Compressed Code for Parallel Decompression	It reduces the code size and thereby improves overall area, power and performance	A larger memory implies increased area (cost) and higher power/energy requirements
4	LICT: Left-uncompressed Instructions Compression Technique to improve the Decoding Performance of VLIW Processors	Improves the performance ratio with only a slight impact on the compression ratio.	The system performance penalty because of the extra time required to decode the compressed instructions during run time.
5	Hamming Distance Based Reordering and Column wise Bit Stuffing with Difference Vector	Smaller coding tables, occupy less area and have larger decompression throughput.	The processor will not consume a fetch packet every clock cycle.
6	VLSI Implementation of Single Bit Control System Processor with Efficient Code Density	Compression schemes results in very high performance control systems	Required larger memory
7	Decoding Aware Compression of FPGA Bitstreams	Improves the communication bandwidth and thereby decreases the reconfiguration time.	Slow decompression or fast decompression at the cost of compression efficiency.

8	An Efficient Tool Chain for Analysing Tradeoffs of Code compression schemes in Embedded Processors	Ability to achieve fast address resolution at decompression time.	Strategy becomes difficult due to the unavailability of flexible tools or frameworks in the field of code compression
9	A New Code Compression Algorithm and its decompression in FPGA Based Hardware	Increasing system performance and reducing energy consumption	Need for high priority for projects in embedded systems to achieve efficiency
10	Code Compression for Embedded Systems Using Separated Dictionaries	Reduce the codeword length of high frequency instructions	All binary instructions are compressed offline and decompressed as required during execution.

3. CONCLUSION

An improved DCC algorithm is proposed in this paper. The fully separated dictionary architecture was used to improve the performance of the decoder, and this architecture is better suitable to decompress instruction in parallel to increase the decompression bandwidth per cycle. Not only the compression ratio but also the performance, zipper effect, power consumption and communication bandwidth between the memory and caches should be analyzed. We can replace the PLL with Delay Locked Loop, in will reduce the power losses and memory size. DLL based dictionary search algorithm increases the memory utilization and increases the operating speed of the processor.

REFERENCES

[1] Wei Jhih Wang and Chang Hong Lin ”Code Compression for Embedded Systems Using Separated Dictionaries”, in IEEE Transactions o very large scale integration (VLSI) systems 1 063-8210 © 2016 IEEE

[2] Wanderson Roger Azevedo Dias “A New Code Compression Algorithm and its Decompressor in FPGA-Based Hardware” 978-1-4799-1132-5/13/$31.00 ©2013 IEEE

[3] Sreejith K Menon “An Efficient Tool-chain for Analyzing Tradeoffs of Code Compression Schemes in Embedded Processors” 2012 IEEE International Conference on Embedded and Real-Time Computing Systems and Applications

[4] Xiaoke Qin and Prabhat Mishra “Decoding-Aware Compression of FPGA Bitstreams” IEEE Transactions On Very Large Scale Integration (VLSI) Systems, VOL. 19, NO. 3, MARCH 2011

[5] Usha S. Mehta “Hamming Distance Based Reordering and Columnwise Bit Stuffing with Difference Vector: A Better Scheme for Test Data Compression with Run Length Based Codes” 2010 23rd International Conference on VLSI Design

[6] Jayasanthi Ranjith “VLSI Implemetation of Single Bit Control System Processor with Efficient Code Density” 978-1-4244-7770-8/10/$26.00 ©2010 IEEE

[7] Talal Bonny and Jörg Henkel "LICT: Left-uncompressed Instructions Compression Technique to Improve the Decoding Performance of VLIW Processors" DAC'09, July 26-31, 2009, San Francisco, California, USA Copyright 2009 ACM 978-1-60558-497-3/09/07

[8] Xiaoke Qin and Prabhat Mishra "Efficient Placement of Compressed Code for Parallel Decompression" 2009 22nd International Conference on VLSI Design

[9] Talal Bonny and Jörg Henkel Efficient Code Compression for Embedded Processors IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 16, NO. 12, DECEMBER 2008

[10] S.-W. Seong and P. Mishra, "An efficient code compression technique using application-aware bitmask and dictionary selection methods," in *Proc. DATE*, 2007, pp. 1–6.