Piotr DZIURZAŃSKI, Tomasz MĄKA
POLITECHNIKA SZCZECIŃSKA, WYDZIAŁ INFORMATYKI

Networks on Chip paradigm for iLBC Speech Decoder

Dr inż. Piotr DZIURZAŃSKI

Ukończył studia na Wydziale Informatyki Politechniki Szczecińskiej w 2000 r. Na tym samym wydziale obronił pracę doktorską w 2003 r. Obecnie pracuje na stanowisku adiunkta w Instytucie Architektury Komputerów i Telekomunikacji WI PS. Jest członkiem organizacji IEEE i ACM. Jego zainteresowania to kosynteza sprzętowo-programowa, synteza wysokiego poziomu, synteza logiczna i formalna weryfikacja.

e-mail: pdziurzaski@wi.ps.pl

Dr inż. Tomasz MĄKA

W roku 2000 ukończył studia na Wydziale Informatyki Politechniki Szczecińskiej, natomiast w roku 2005 obronił pracę doktorską na tym samym wydziale. Jego zainteresowania obejmują zagadnienia związane ze sprzętowymi realizacjami systemów cyfrowego przetwarzania sygnałów oraz technikami przetwarzania sygnałów akustycznych.

e-mail: tmaka@wi.ps.pl

Abstract

In modern hardware multimedia solutions, such as set-top boxes, there exists a trend of implementing numerous codecs in a single device. In this paper, we describe our attitude to mapping a set of decoder stages to a regular mesh structure, which consists of two techniques for decreasing number of cores and assigning IP blocks to NoC nodes.

Keywords: Network on Chip, Speech decoding, iLBC, in-chip transfers.

Wykorzystanie paradygmatu sieci wewnątrzukładowych do dekodera mowy iLBC

Streszczenie

We współczesnych rozwiązaniach sprzętowych, takich jak STB (ang. set-top box), można zauważyć trend implementowania wielu kodeków w pojedynczym urządzeniu. W niniejszym artykule zostało opisane podejście odwzorowania zbioru funkcjonalnie niezależnych etapów dekodera iLBC do regularnej struktury sprzętowej, na które składają się dwa algorytmy pozwalające zmniejszyć liczbę bloków i przypisać te bloki do węzłów sieci wewnątrzukładowej.

Słowa kluczowe: sieci wewnątrzukładowe, dekodowanie mowy, iLBC, transfery wewnątrzukładowe.

1. Introduction

Contemporary hardware solutions in a multimedia domain, such as encoders and decoders, often utilize System on Chip paradigm [5]. However, in order to maintain the performance growth in the future SoCs, it is desirable to apply a new paradigm that allows interconnect scaling performance connecting functional blocks (cores or intellectual-property, IP, blocks) synthesized from high-level descriptions.

The new on-chip interconnect approach that attracts most attention of researchers is the packet-based Network-on-Chip (NoC) paradigm [2]. This technique of communication in distributed systems allows to decrease the contention level in a chip with a large number of cores, where traditional on-chip buses are considered as the main obstacle in the SoC efficiency. The performance of both dedicated wiring and buses have been reported as inferior to NoCs due to its worse reusability, scalability, problems of wiring latency and noise [2]. According to [3], interconnections in multimedia SoCs play a vital role due to the requirement of transferring a large amount of temporal video data between numerous IP blocks. These trends are observed, among others, in set-top box designing approaches [3]. As there exist numerous codecs in the audio, speech, and video coding applications, it is essential to implement as many codecs as possible in a single chip to increase the chip flexibility and reduce power dissipation as well as the time to market and non-recurring engineering costs.

A typical speech codec is comprised of a number of stages that perform autonomous operations. Consequently, an algorithmic structure of these codecs is suitable to a mesh NoC architecture due to the facts that (i) the stages implementations are characterized with coarse grain parallelism, i.e., the stages communicate after executing large portions of code, (ii) the stages of codecs are selected in such the way that the transfer between them are quite balanced. These properties follow the fact that codec algorithms are usually planed with hardware implementation in mind [6].

2. Algorithm

As the number of cores is determined with the number of routers and wires implementable in a target device, it is desirable to merge codec blocks to obtain the number of cores that fits in the chip. Since the transfer in a single core is much faster than the asynchronous transfer between different cores [2], it seems important to merge blocks that transfer the maximum number of bytes between them. This operation is to be repeated until the desired number of cores is obtained. This greedy approach is of computational complexity $O(n^2)$, where n is the initial number of cores.

Having obtained the appropriate number of cores, we have to determine the mapping of the cores into the mesh structure leading to improved performance of the NoC structure. In order to find the favorable permutation, we determine the number of used connections (C), the total transfer (T), the mean value of transfer per a connection (M), and a standard deviation of the transfer on each connection (D). The analysis of the influence of these parameters on NoC performance is described in the sequel of this paper. Consequently, the criteria used for selecting the appropriate permutation is of the form of $\alpha \cdot T + \beta \cdot C + \gamma \cdot M + \delta \cdot D$, where α, β, γ and δ are coefficients whose sum is equal to 1 and whose values are chosen by an engineer in order to obtain the NoC that follows the design constraints. For example, using high value of the δ parameter may results in the mapping that is characterized with balanced transfers, whereas the number of used connections C and the length of routes between cores may be far from being minimal. However, due to balanced transfers, the probability of contention is lower than when we minimize, e.g., the total transfer value or the number of used links.

With our first algorithm, we generate all the core permutations and determine the flow for each connection. When a core is to send a package into a core that is not its neighbor, it crosses a series of links and this transfer increases the traffic on all these links. In order to route the packages, we use the well-known XY routing algorithm that firstly route a flit according to X axis as long as the X coordinate is equal to the X coordinate of the destination core, and then the flit is routed vertically. It was proved in [4] that this algorithm is deadlock-free.

The algorithm for determining the transfer for each permutation is considerable computational complexity $O(n! \cdot n^2)$, where n is the number of cores. However, our experiments, described in the next section, showed that its speed is acceptable for values of n used in practice.

3. Case study

In order to verify the above provided approach, we have chosen the widely-used iLBC decoder [1], i.e., an Internet low bit rate narrow-band speech codec dedicated to VoIP solutions, such as Skype, and other applications for real time communications. The main property of this codec that results in suitability for Internet telephony is its robust packet loss concealment. The iLBC codec offers two modes with 13,33 kbps (frame 30 ms, length 400 bits) and 15.2 kbps (frame 20 ms, 304 bits) bit rates, using 8 kHz sample rate.

The structure of the iLBC decoder is depicted in Fig. 1, where eight modules are singled out. These modules are described in, e.g., [1].

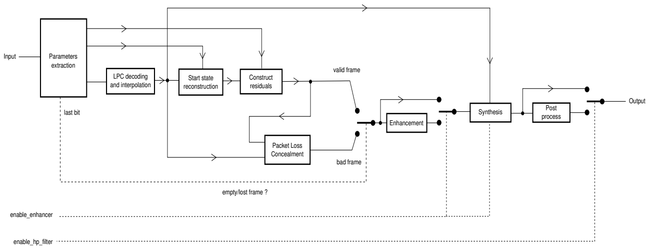

Fig. 1. Block diagram of iLBC decoder
Rys. 1. Diagram blokowy dekodera iLBC

We have implemented the algorithms, described in Section 2, for generating an exhaustive set of permutations with the number of used connections, the total transfer, the mean value of transfer per a connection, and a standard deviation of the transfer on each connection. We fixed the position of the 0th core in the left-down corner of the mesh as we treat it as the source core that is to be connected with the external environment.

Three examples of permutations (out of 5040) and their parameters are presented in table 1. In this table, the first and the last permutations are of the highest and the lowest value of the standard deviation, respectively, whereas the remaining one is in the middle between them.

Tab. 1. Selected core permutations and their characteristics
Tab. 1. Wybrane permutacje rdzeni i ich charakterystyki

Permutation	Total transfer (T) [bits]	Used connections (C)	Mean (M) [bits]	Standard deviation (D) [bits]
0-3-7-6-1-5-4-2	132832	18	5534	2685.42
0-6-5-7-1-3-4-2	127200	17	5300	4524.91
0-2-5-3-1-4-6-7	112288	12	4678	9017.33

We have built a histogram for the standard deviation and observed that there is no single permutation whose standard deviation is lower than 3000 bits. Then, only 12% of the permutations have the standard deviation between 3000 and 3600 bits. These permutations are favorable as they lead to decreasing the contention level. On the other hand, the standard deviations of 6.8% of the permutations are over 6600 bits. These permutations are to be avoided due to the fact that a certain number of their connections transfer huge amount of data and they become bottlenecks.

The histogram of the number of used connections (out of 24 possible) resembles the Gauss distribution; the lowest value is equal to 10 (for 5 permutations), the highest – 21 (for 4 permutations), whereas the maximum value of the used connections, equal to 16, was reached for 1184 permutations.

The number of total connections is important for the final realization as it determines the number of the connections that can be removed. Notice, that every connection requires the number of lines equal to the length of the flit increased by two for the request and acknowledge signals. Thus, selecting the permutation with a lower number of used connections may lead to a better routability while implemented in FPGA devices. However, the lower is the number of used connections, the higher is the standard deviation value and, consequently, the higher level of contention.

In Fig. 2 we illustrated the connections and the transfers for the NoC mesh when the three permutations from table 1 are chosen.

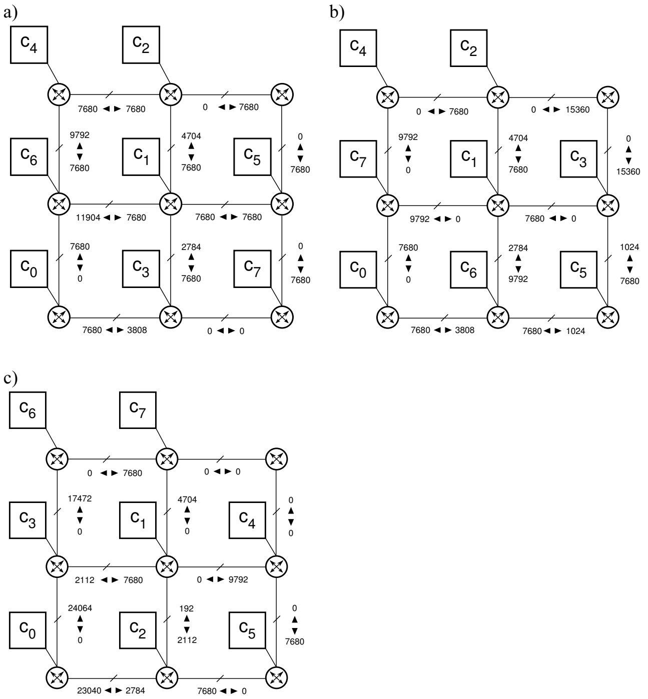

Fig. 2. Connections and transfers for the NoC mesh with permutations from Tab. 1
Rys. 2. Połączenia i transmisje w NoC z permutacjami z Tab. 1

Despite large computational complexity, the implementation of the algorithm works effectively for 3x3 mesh architecture and finds the proper solution in few seconds.

4. Conclusion

In this paper, we have described an approach for mapping a set of autonomous modules into a regular NoC structure. We provided two algorithms for decreasing number of cores and assigning codec IP blocks into NoC mesh nodes. Taking into account the total transfer, the number of used connections, the mean value of transfer per connection, and standard deviation, we discovered that the number of cores is characterized with the Gauss distribution, and that only a small portion of the permutations has low standard deviation values. We provide some discussion on the influence of the above mentioned parameters into NoC implementation properties. The results of the paper may be used as guidance for an engineer working on codecs implementation in reprogrammable circuits.

5. References

[1] S.V. Andersen, et al., iLBC - A linear predictive coder with robustness to packet losses, IEEE Speech Coding Workshop, pp. 23-25, 2002.

[2] T. Bjerregaard, S. Mahadevan, A Survey of Research and Practices of Network-on-Chip, ACM Computing Surveys (CSUR), vol. 38, Article 1, 2006.

[3] K. Goossend, et al., Interconnect and memory organization in SoCs for advanced set-top boxes and TV – evaluation, analysis, and trends. In J. Nurmi, et al., (eds.), Interconnect Centric Design for Advanced SoCs and NoCs, chapter 15, pp. 399-423, Kluwer, 2004.

[4] M. Li, Q.A. Zeng, W.B. Jone, DyXY: a proximity congestion-aware deadlock-free dynamic routing method for network on chip, 43rd ACM IEEE Design Automation Conference, pp. 849-852, 2006.

[5] F. Remond, P. Bricaud, Set Top Box SoC Design Methodology at STMicroelectronics, Design, Automation, and Test in Europe Conference and Exhibition (DATE'03), pp. 220-223, 2003.

[6] A. S. Spanias, Speech Coding: A Tutorial Review, Proceeding of the IEEE, vol. 82, no. 10, pp. 1541-1582, 1994.

Artykuł recenzowany