Titan II : An IPComp Processor for 10Gbit/sec networks

Ioannis Papaefstathiou
Institute of Computer Science, Foundation of Research and Technology, Hellas
Vassilika Vouton, P.O. Box 1375, GR 71110, Crete, GREECE

Abstract

As it has already been proved, link layer compression is very effective when used in packet networks. In particular, IP payload compression is especially useful when encryption is applied to the network packets. Encrypting the IP packets causes the data to be random in nature, rendering compression at lower protocol layers. As a result, and since it is believed encryption will be applied to the vast majority of IP networks in the near future, we claim that an IP packet compressor will be required for taking full advantage of the capabilities of the future networks. However one of the major problems with such network compression schemes, is that there should exist hardware modules capable of compressing the network streams up to the speed of the state-of-the-art links. In this paper we present such a hardware compressor/decompressor core that can work at speeds up to 10Gb/sec, it is fairly inexpensive and can very easily be plugged into an existing network node without causing any side effects. The presented design can be easily incorporated in a network System-on-a-Chip(Soc).

1 Introduction

An important application of packet networks is in the interconnections of private LANs. In this capacity a packet network protocol sees common carrying data over WAN links. In these networks, data is often encrypted and it is widely supported that the vast majority of such networks in the future will use IP layer encryption. IP payload compression is especially useful when encryption is applied to IP datagrams. Since, encrypting the IP datagram causes the data to be random in nature, compression at lower protocol layers (e.g., PPP Compression Control Protocol) becomes ineffective. In general, if both compression and encryption are required, compression MUST be applied before encryption . As a result, IP payload compression is especially useful and thus a specific protocol for compression of IP packets has been proposed [2]. As it is shown in [1], if a standard dictionary based algorithm (i.e. a Lempel-Ziv one) is applied to the network streams arriving at/departing from the gateways of such networks, the increase in the effective bandwidth of network varies from 50% to 150%.

This paper describes the Titan II which is a device that implements the above compression mechanism at up to 10Gb/sec rates. It implements a completely novel design in order to achieve reliability, high speed and low latency. In the next sections the hardware architecture and implementation of this device is outlined.

2 Core Hardware Architecture

As in every dictionary based compression device, the core comprises of the dictionary and the comparison circuits around it. Therefore, the speed of the device depends heavily on the memory throughput and the comparisons' latency. In order to accelerate the compression core the following techniques were used:

- 256-stage pipeline.
- 16 comparisons in parallel at each pipeline stage.
- Memory repetition (100% more memory used) for higher memory throughput.

The Titan II was implemented on a UMC 0.18um CMOS technology with a worst case 2 input NAND gate delay of 0.25ns and a worst case memory latency of 2.4ns.

In the next paragraphs only the compression unit is described because the decompression one is much simpler, since there is no need for comparisons between the input data and the one stored in the dictionary.

The compression unit implements a typical Lempel-Ziv algorithm which is applied only to the payloads of the IP packets. Its block diagram is shown in Figure 1. In general, the compressibility table determines if a flow should be compressed or if it should bypass the main unit and the header memory, and the merge and bypass circuits ensure the cells will be formed and according to the network protocol's specifications.

As stated above, the core circuit is organized in a 256-stage pipeline. In Figure 2 a pipeline stage is demonstrated.

Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI'03)
0-7695-1904-0/03 $17.00 © 2003 IEEE

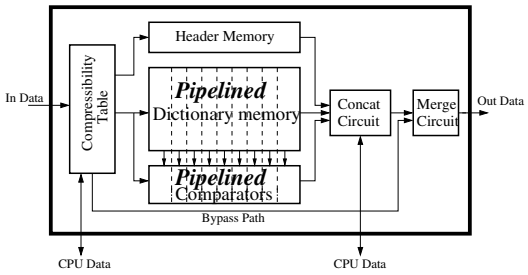

Figure 1: Compression Unit Block Diagram

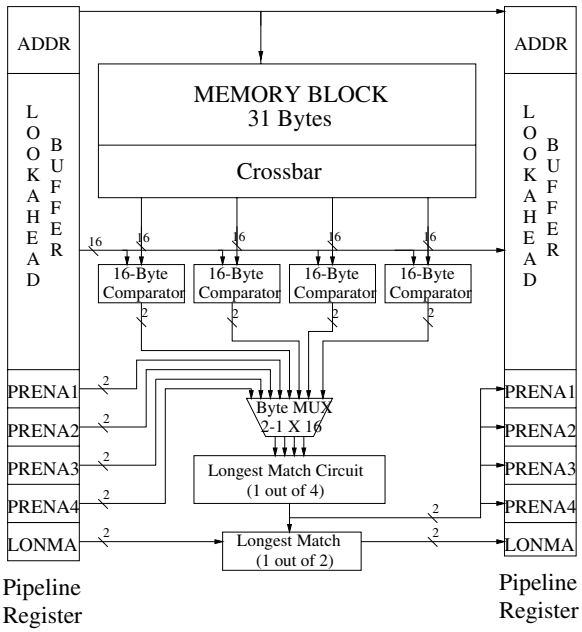

Figure 2: Block diagram of a Pipeline stage. Wire widths are in **Bytes**

It consists of a memory bank of 31 bytes for each dictionary[1], a "crossbar" so as to be able to route each 16-byte quantity to a specific comparator and 64 comparators that are used 4 times each at every clock cycle. The inputs of each stage are (a) the 16-byte long lookahead buffer, (b) the 15-bit address of the dictionary that should be used, (c) register LONMA which specifies what is the longest match up to the last point of the pipeline and what is the dictionary address of the first byte of this match, and (d) the four PRENA registers which specify the 4 longest matches found in the last pipeline stage and the addresses of these match. The outputs of each stage are (a) the unchanged 16-byte lookahead buffer, (b) the also unchanged 15-bit address fields, (c) the possibly altered LONMA register, and (d) the new PRENA registers which specify the 4 longest matches found on this stage.

The reasoning behind the size of the memory is as follows: The algorithm implemented has a longest possible match of 16. Thus, taking 16 subsequent bytes, all their possible matches are included in these 16 bytes and the next 15 subsequent ones. So, it is guaranteed that all the matches of the first 16 bytes are included in the 31 bytes stored in the memory.

Since the main objective has been to minimize the time for the comparisons of the 16 byte lookahead buffer with every single byte in the dictionary, parallelism is also used. In every pipeline stage there are 64 byte-comparators each used 4 times in each major cycle. Therefore, all of the 256 comparisons needed are done in each major cycle.

By using all the above speedup factors the compressor can process data at a constant speed of more than 1Gb/sec (1.17Gb/sec) introducing a latency of 256 clock cycles or a couple of mean packet times. This is a significant improvement over the current network compressors the fastest such device can work up to a speed of several hundred Mb/sec and its latency is up to several payload times.

In order for this design to be used in Multi Gigabit network the only alteration needed is the following: Instead of having 64 comparators in each pipeline stage 256 are needed, so as all the necessary comparisons can be done at the same time. In this case the latency of each pipeline stage is 3.5ns and thus a clock rate of 275MHz can be used. As a result, every single compressor can process data at a rate more than 2Gb/sec (2.23Gb/sec). So, by plugging 5 of these compressors in parallel, a 10Gb/sec total throughput can be achieved.

3 Conclusions

The device presented here addresses some of the most important implementation issues of a network compression scheme, i.e high throughput and low latency. It can compress network streams at speeds up to 10Gb/sec with a latency of just 2 packet times. Therefore, it is claimed that it can be used with the state-of-the-art high speed networks. The presented core is about to be fabricated as part of a sophisticated network processor.

References

[1] I. Papaefstathiou " *Compressing ATM streams* . IEEE Data Compression Conference 1999 (DCC'99), Utah, 29-31 March 1999.

[2] A. Shacham, R. Monsour, R. Pereira and M. Thomas, "IP Payload Compression Protocol (IPComp)", RFC-2393, December 1999.

[1] Since the address is 15-bit wide up to 32K dictionaries can be supported. However the number of the actual dictionaries will depend on the cost requirements the device should satisfy.