A FPGA Implementation of Gzip compression

Bisheng Huang Tsinghua University
Xinyu Niu Imperial College London
Wayne Luk Imperial College London

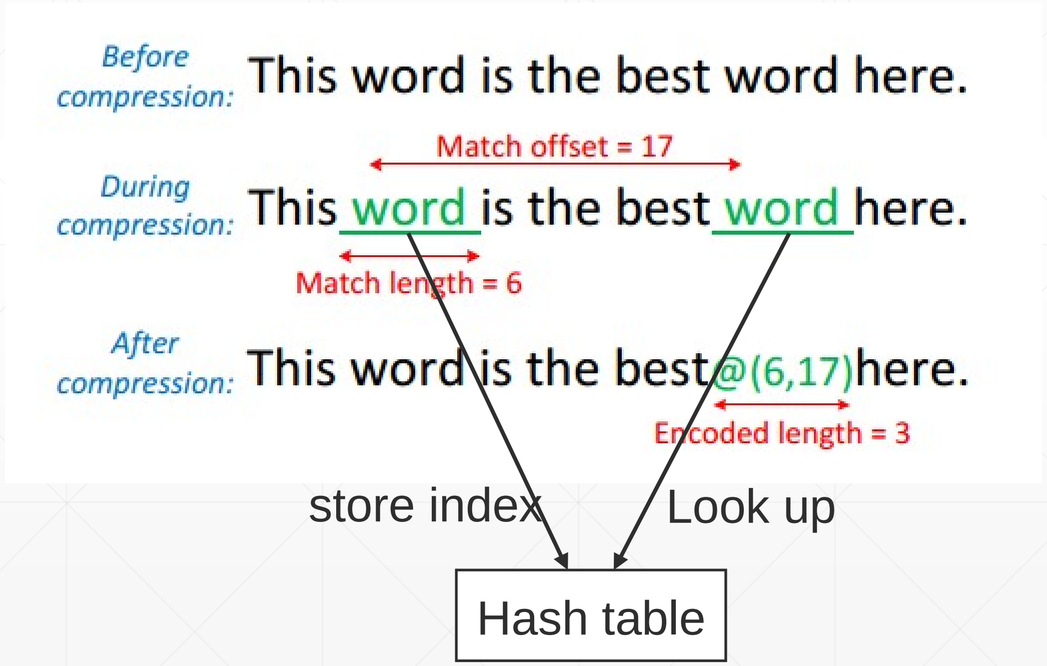
28, August 2015

Background

- Applications such as databases and web servers have to process hug e amounts of data. They have a critical need for lossless data compres sion.
- Hardware implementation of lossless data compression is important for optimizing the capacity/cost/power of storage devices.
- Goal: Use the DFE to optimize the speed and quality of the compressi on process

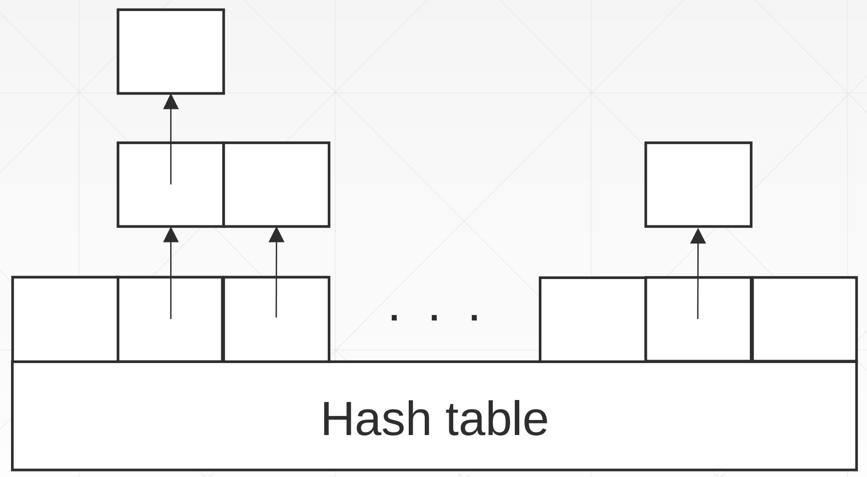
Gzip compression in software

- Based on the widely used standard to perform lossless compression - DEFLATE.
- The key is to find a reference to a duplicated string in previous contexts.
- Keep track of
  - History data window of certain size (e.g. 4kb)
  - Hash table to store indices



The Challenges in hardware

- The CPU uses a link list to maintain the hash table
- The number of elements in each bucket is dynamic

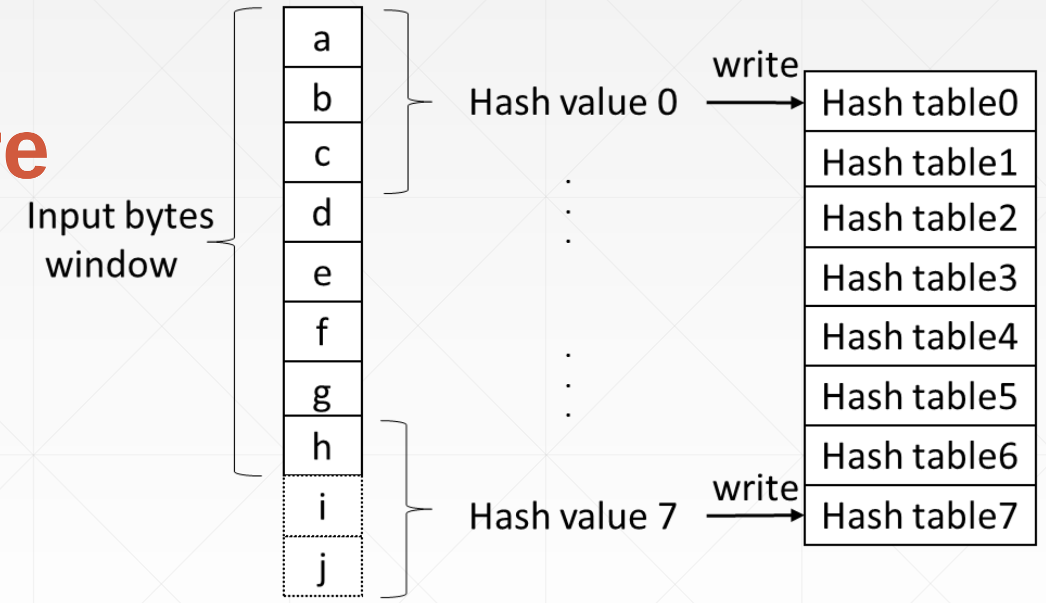


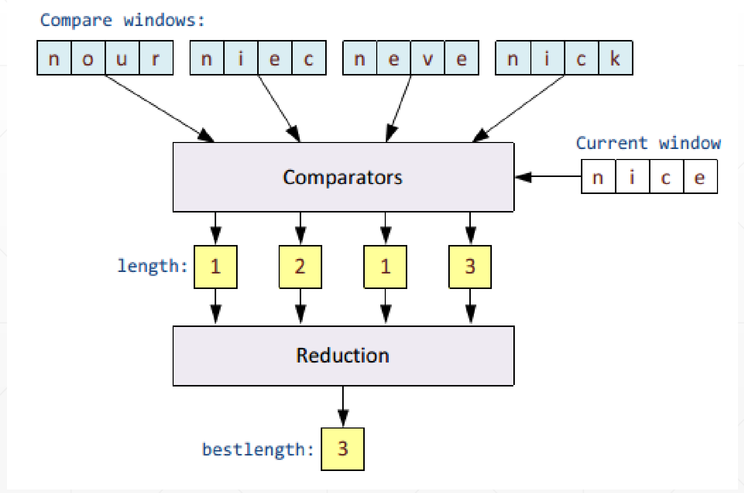
- In hardware, it’s not easy to keep the data structures and be pipelined at the sam e time
- Referred to some prior implementations

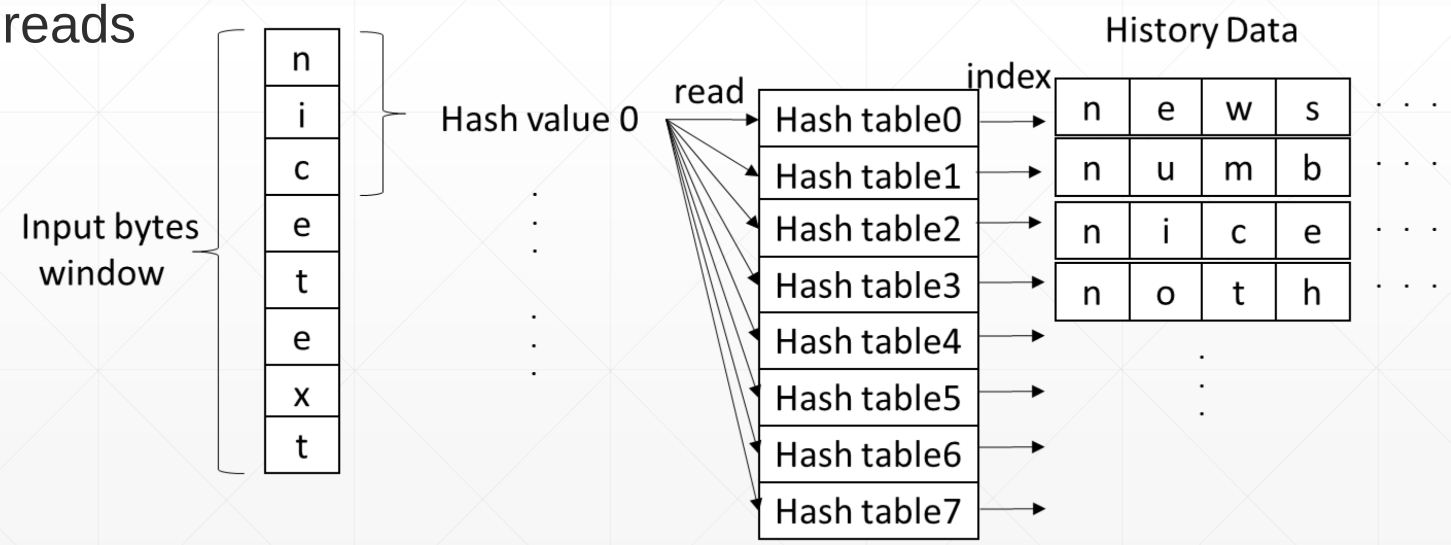
How we put it into hardware

- Hashing architectures
  - Multiple hash tables
  - For each table in one cycle,

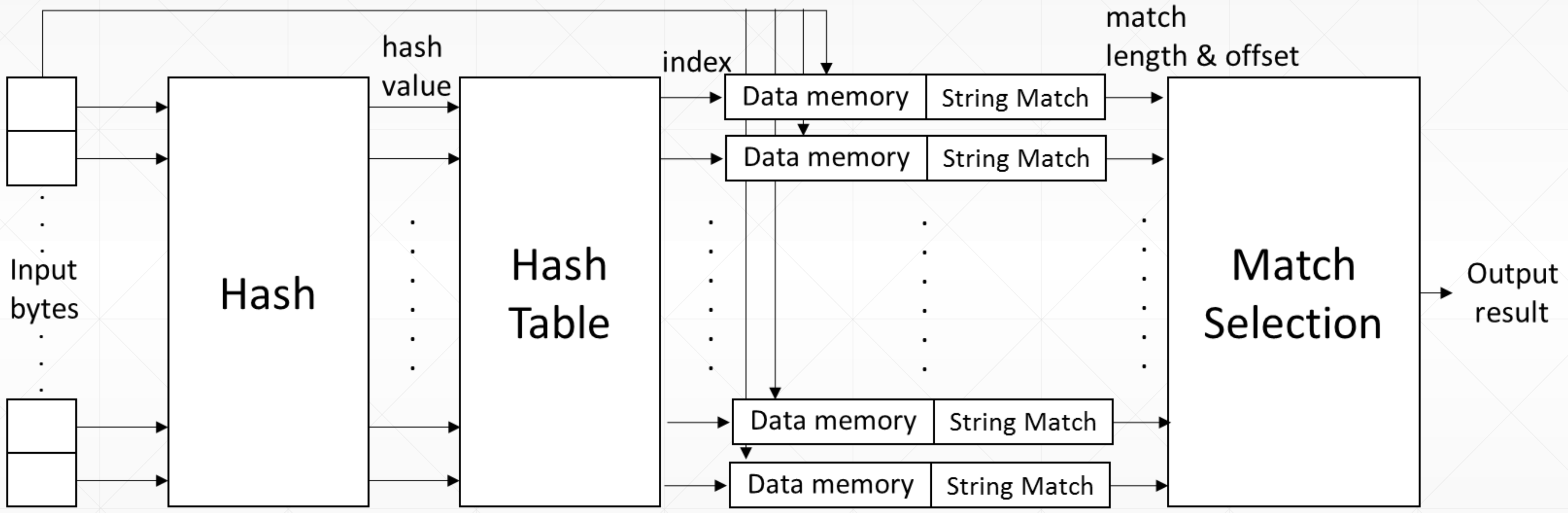
    single writes and multiple reads







The overall architecture of the design



Performance

- The standard benchmarks - Calgary Corpus

$$Compression\ Ratio = \frac{Uncompressed\ file\ size}{Compressed\ file\ size}$$

Versions	Throughput	Clock Frequency	Ratio	Resources Usage
Read One Index_ One Match	100MB/s	100MHz	1.88	21% Logic, and 25% Mem
Read Eight Indices_ Eight Match	1.6GB/s	200MHz	1.88	97% Logic and 51% Mem

- The problem is the logic resource usage: the use of cross bar when matching.
- The solution may be avoid using crossbar.

Comparison with prior implementations

- For throughput improvement,
  - Increase the input window from 8 to 16 . We have the potentials to reach **3.2GB/s**.
- For compress ratio improvement,
  - Add Huffman encoding.
  - Parameters like data memory size and hash size can be tuned.

	Throughput	Ratio
Altera(OpenCL)	2.94 GB/s	2.18
IBM(Verilog)	3 GB/s	-
Intel(CPU)	338MB/s	2.17
Our design	1.6 GB/s	1.88

Results when considering file IO

- The results in the above do not consider disk IO on actual file system.
- Single core CPU with -O2
- Disk IO bounded at ~30MB/s
- PCI-LMEM bounded at ~110MB/s

	Throughput	Ratio	Speed up
DFE	30.9 MB/s	1.88	-
CPU (same hashing strategy)	9.4 MB/s	1.90	~3x
CPU	2.5 MB/s	2.35	~20x

Review

- Comparable performance should be achieved after dealing with the cross bar problem, and tuning the parameters.

- Personal experience:
  - Time: two months
  - Get to know about the research area and the advance developme nt tools on FPGA.

References

- Fowers, Jeremy ; Kim, Joo-Young ; Burger, Doug ; Hauck, Scott; *A Scalable High-Bandwidth Architecture for Lossless Compression on FPGAs*. IEEE International Symposium on Field-Programmable Custom Computing Machines, 2015.
- Abdelfattah, M. S.; Hagiescu, A. & Singh; *Gzip on a chip: high performance lossless data compression on FPGAs using OpenCL.* Proceedings of the International Workshop on OpenCL, 2014.
- Jian Ouyang, Hong Luo, Zilong Wang, Jiazi Tian, Chenghui Liu, Kehua Sheng. *FPGA Implementation of GZIP Compression and Decompression for IDC Services*. Int. Conf. on Field-Programmable Technology, 2010.