ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Correction Predictive Architecture for Error Correction Code with Low Complexity

Chaganti Varchaswi
M.Tech (VLSI&ES),
Department of ECE,
Devineni Venkataramana & Dr Himasekhar
MIC College of Technology.

M.Adi Seshaiah
Assistant Professor,
Department of ECE,
Devineni Venkataramana & Dr Himasekhar
MIC College of Technology.

ABSTRACT:

In this paper. A new architecture for matching the data protected with an error-correcting code (ECC) is proposed in brief to reduce latency and complexity. To further reduce the latency and complexity, in addition, a new butterfly-formed weight accumulator (BWA) is proposed for the efficient computation of the Hamming distance. Grounded on the BWA, the proposed architecture examines whether the incoming data matches the stored data, and if not it aims to locate the erroneous bit and they are corrected. The empirical evaluation proves that the proposed methodology discovers the best service for reliability issues of memory and to reduce the area of BWA compressors are used to replace the adders .xilinx tool is used with Verilog language.

INTRODUCTION:

The data comparison usually resides in the critical path of the components that are devised to increase the system performance, e.g., caches and TLBs, whose outputs determine the flow of the succeeding operations in a pipeline. The circuit, therefore, must be designed to have as low latency as possible, or the components will be disqualified from serving as accelerators and the overall performance of the whole system would be severely deteriorated.As recent computers employ errorcorrecting codes (ECCs) to protect data and improve reliability , complicated decoding procedure, which must precede the data comparison, elongates the critical path and exacerbates the complexity overhead. Thus, it becomes much harder to meet the above design constraints.Despite the need for sophisticated designs as described, the works that cope with the problem are not widely known in the literature since it has been usually treated within industries for their products. however,triggered the attraction of more and more attentions from the academic field.

The most recent solution for the matching problem is the direct compare method , which encodes the incoming data and then compares it with the retrieved data that has been encoded as well. Therefore, the method eliminates the complex decoding from the critical path. In performing the comparison, the method does not examine whether the retrieved data is exactly the same as the incoming data. Instead, it checks if the retrieved data resides in the error correctable range of the codeword corresponding to the incoming data.BWA is used to compute hamming distance between 2 code words such as X and Y. BWA is constructed by a set of HA. The matching or fault of the received code words is computed based on hamming distance computation method.

RELATED WORK:

Architecture for Computing the Hamming Distance

The proposed architecture grounded on the datapath design is shown in Fig. 1. It contains multiple butterfly-formed weight accumulators (BWAs) proposed to improve the latency and complexity of the Hamming distance computation.

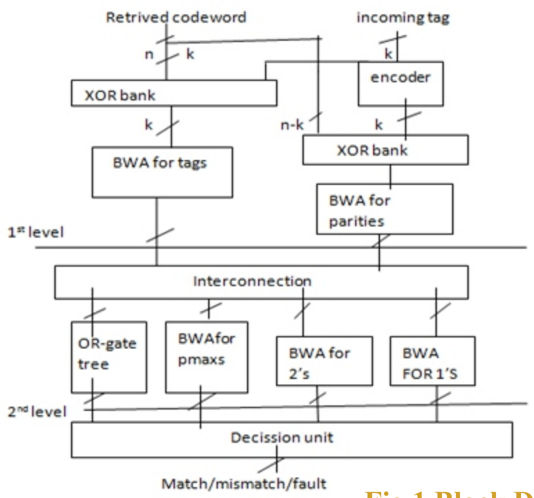

Fig 1 Block Diagram

The basic function of the BWA is to count the number of 1's among its input bits. It consists of multiple stages of HAs as shown in Fig. 3.5, where each output bit of a HA is associated with a weight. The HAs in a stage are connected in a butterfly form so as to accumulate the carry bits and the sum bits of the upper stage separately. In other words, both inputs of a HA in a stage, except the first stage, are either carry bits or sum bits computed in the upper stage. This connection method leads to a property that if an output bit of a HA is set, the number of 1's among the bits in the paths reaching the HA is equal to the weight of the output bit. In Fig. 2, for example, if the carry bit of the gray-colored HA is set, the number of 1's among the associated input bits, i.e., A, B, C, and D, is 2. At the last stage , the number of 1's among the input bits, d, can be calculated as

d= 8I + 4 (J + K + M) + 2 (L + N + O) + P.

Since what we need is not the precise Hamming distance but the range it belongs to, it is possible to simplify the circuit. When rmax = 1, for example, two or more than two 1's among the input bits can be regarded as the same case that falls in the fourth range. In that case, we can replace several HAs with a simple OR-gate tree as shown in Fig. 6(b). This is an advantage over the SA that resorts to the compulsory saturation expressed . Note that in Fig. 6, there is no overlap between any pair of two carry-bit lines or any pair of two sum-bit lines. As the overlaps exist only between carry-bit lines and sum-bit lines, it is not hard to resolve overlaps in the contemporary technology that provides multiple routing layers no matter how many bits a BWA takes.

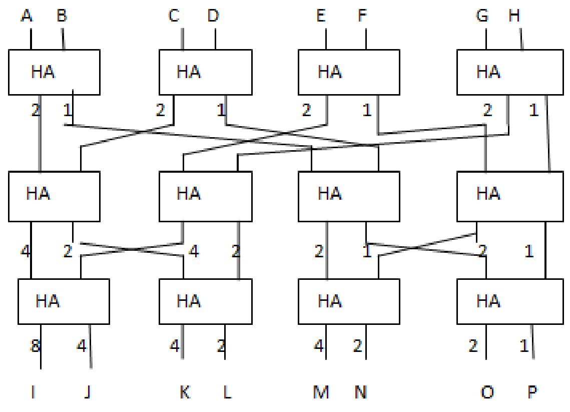

Fig 2 BWA HA Architecture

Each XOR stage in Fig. 2generates the bitwise difference vector for either data bits or parity bits, and the following processing elements count the number of 1's in the vector, i.e., the Hamming distance. Each BWA at the first level is in the revised form shown in Fig. 2, and generates an output from the OR-gate tree and several weight bits from the HA trees.

In the interconnection, such outputs are fed into their associated processing elements at the second level. The output of the OR-gate tree is connected to the subsequent OR-gate tree at the second level, and the remaining weight bits are connected to the second level BWAs according to their weights.

More precisely, the bits of weight w are connected to the BWA responsible for w-weight inputs. Each BWA at the second level is associated with a weight of a power of two that is less than or equal to Pmax, where Pmax is the largest power of two that is not greater than rmax + 1.

As the weight bits associated with the fourth range are all ORed in the revised BWAs, there is no need to deal with the powers of two that are larger than Pmax.

PROPOSED ARCHITECTURE:

To reduce further latency BWA compressors are replaced with BWA adders.

Compressor design:

Adder compressors have been used to implement arithmetic and digital signal processing (DSP) circuits for low power and high performance applications . Compressors are also used in multiplier architectures. Multipliers are structured into three functions: partial-product generation, partial-product accumulation and final addition.

The main source of power, delay and area came from the partial-product accumulation stage . Compressors usually implement this stage because they contribute to the reduction of the partial products (reducing the number of adders at the final stage) and also contribute to reduce the critical path which is important to maintain the circuit's performance.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

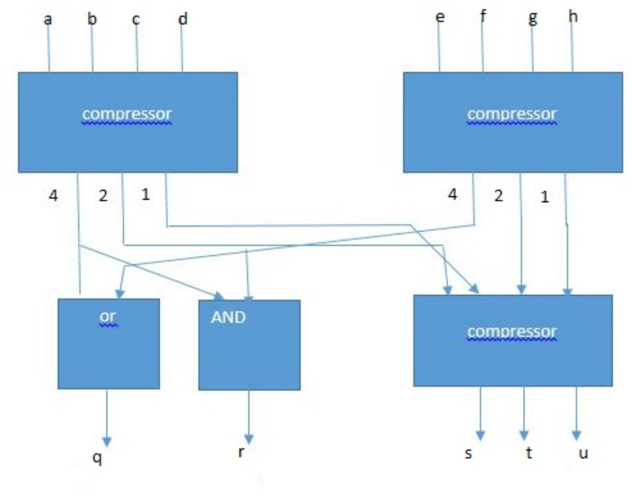

FIG3 : BWA COMPRESSOR ARCHITECTURE

In the above fig shows BWA compressor architecture. Its calculate no.of 1's depending on given inputs. number of 1's shows number of errors.

SIMULATION RESULTS:

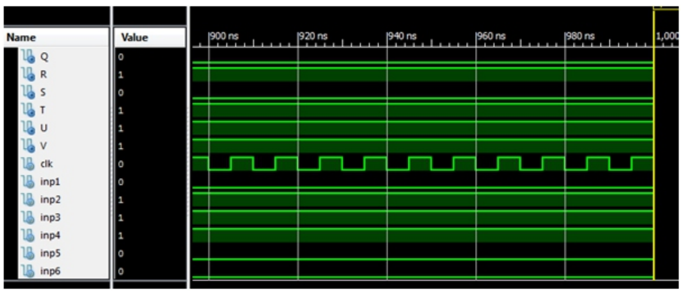

Fig 4 Output Waveform.

The above fig 4 show the output waveform of BWA architecture.

Conclusion:

To reduce the hardware complexity and latency, a new architecture has been presented for matching the data protected with an ECC.To reduce the latency, the comparison of the data is parallelized with the encoding process that generates the parity information. The parallel operations are enabled based on the fact that the systematic codeword has separate fields for the data and parity. In addition, an efficient processing architecture has been presented to further minimize the latency and complexity. Therefore a reasonable reduction in area is achieved with the proposed design.

References:

1.J. Chang, M. Huang, J. Shoemaker, J. Benoit, S.-L. Chen, W. Chen,S. Chiu, R. Ganesan, G. Leong, V. Lukka, S. Rusu, and D. Srivastava,"The 65-nm 16-MB shared on-die L3 cache for the dual-core Intel xeonprocessor 7100 series," IEEE J. Solid-State Circuits, vol. 42, no. 4,pp. 846–852, Apr. 2007.

2. J. D. Warnock, Y.-H. Chan, S. M. Carey, H. Wen, P. J. Meaney, G. Gerwig,H. H. Smith, Y. H. Chan, J. Davis, P. Bunce, A. Pelella, D. Rodko,P. Patel, T. Strach, D. Malone, F. Malgioglio, J. Neves, D. L. Rude,and W. V. Huott "Circuit and physical design implementation of the microprocessor chip for the zEnterprise system," IEEE J. Solid-StateCircuits, vol. 47, no. 1, pp. 151–163, Jan. 2012.

3.H. Ando, Y. Yoshida, A. Inoue, I. Sugiyama, T. Asakawa, K. Morita,T. Muta, and T.Motokurumada, S. Okada, H. Yamashita, and Y. Satsukawa,"A 1.3 GHz fifth generation SPARC64 microprocessor," in IEEE ISSCC. Dig. Tech. Papers, Feb. 2003, pp. 246–247.

4. M. Tremblay and S. Chaudhry, "A third-generation 65nm 16-core32-thread plus 32-scout-thread CMT SPARC processor," in ISSCC. Dig.Tech. Papers, Feb. 2008, pp. 82–83.

5. AMD Inc. (2010). Family 10h AMD Opteron Processor Product Data Sheet, Sunnyvale, CA, USA [Online].

6.W. Wu, D. Somasekhar, and S.-L. Lu, "Direct compare of information coded with error-correcting codes," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 20, no. 11, pp. 2147–2151,Nov. 2012.

7.S. Lin and D. J. Costello, Error Control Coding: Fundamentals and Applications, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall,2004.

8. Y. Lee, H. Yoo, and I.-C. Park, "6.4Gb/s multi-threaded BCH encoder and decoder for multi-channel SSD controllers," in ISSCC Dig. Tech.Papers, 2012, pp. 426–427.