Proceedings of the 5th WSEAS International Conference on Applied Computer Science, Hangzhou, China, April 16-18, 2006 (pp118-128)

NEW HARDWARE ARCHITECTURE FOR BIT-COUNTING

Ahmed Dalalah
Computer Science Dept.
Jordan University of Science and Technology

Sami Baba
Computer Science Dept.
Applied Science Private University

Abdallah Tubaishat
College of Information Systems
Zayed University

Abstract

Bit-counting implementations are used to count the number of 1's in a given computer word. There are several techniques to implement bit-counting operation. These techniques are either so algorithms or specialized hardware techniques. The hardware implementations requi hardware supported in the processor or associated math co-processor. The perform hardware-supported bit-counting was found to be superior to most software implementati serial shifting). In this paper, a new hardware implementation of bit-counting routine is p reduces the number of logic gates and the delay in comparison with existing impleme performance of the proposed hardware bit-counting implementations is further inve evaluated.

Key words: Bit-Counting, Bit-Parallelism, Counters, Popcount, Redundant Co

1. INTRODUCTION

Bit-counting is a well-known computer operation used to determine the number of "1's" in a given computer word [Berkovich, et al.,2000] [El-Qawasmeh, et al., 2000]. Some applications, like those related to coding theory procedures, must be supported by specialized high-speed, processor-implemented bit-counting functions that may rely on dedicated processor facilities including hardware. Increasing attention to the issue is evident from the numerous patents directed to different hardware architectures, dedicated for bit-counting ([Ashkenazi, 1996], [Hossain, 2004] [Balmer, 1994] and [Morris, 1998]). Currently, many applications use the bit-counting operation. Among these applications are: Information retrieval, file comparison problems and genetic algorithms. For example, information retrieval systems may represent search results in the form of a single bit attribute matrix representing hundreds-of-thousands of documents indicating whether each document satisfies one or more search criteria. In this case, bit-counting is used to determine the number of documents satisfying the search criteria. Likewise, file comparison routines may be used to compare files having large numbers of elements. In this case, a counter of matching elements may be required to find an overall match metric. In addition, genetic algorithms use the bit-counting operation in many of its algorithms [Goldberg, et al., 1992].

Bit-counting instructions are sometimes referred to as "population counts" and are abbreviated by "popcount". For simplicity of reference, the term "popcount" as used herein will refer to hardware implementations of a word bit count instruction, e.g., "popcnt" on Compaq alpha machines.

Popcount techniques are classified into two categories. The first category is software based techniques that span a wide range of algorithms. These algorithms include: a)- Serial shifting, b)- Table lookup, c)- Arithmetic logic counting, d)- Emulated popcount, e)- Hamming distance bit vertical counter, and f)- Frequency Division (see [El-Qawasmeh, et al., 2000], [Gutman, 2000] and [Reingold, et al.,1977]). The second category is hardware implementations, which is the scope of this paper. Hardware implementations used a special circuitry for "popcount" function.

In this paper, a new proposed hardware circuitry to do the bit-counting is suggested. The purpose of the proposed design is to reduce the number of logic gates and the delay in comparison with current designs. The suggested implementation is compared and evaluated with other implementations.

The organization of this paper is as follows. Section 2 describes current hardware implementation for bit-counting. Section 3 describes our proposed hardware. Section 4 is performance analysis. Section 5 is a discussion and section 6 is the conclusion.

2. HARDWARE IMPLEMENATATIONS

Popcount is a built-in function that was first introduced by *Cray* Company and then added by other companies such as *Texas Instruments* in TMS320C62^{TM} DSP processors and *HP* in Alpha EV6 processor ([Texas Instruments, 2004] [HP, 2005]). This function allows the programmer to access directly hardware and machine instructions. The same function can be implemented using software rather than hardware by adding it to the definitions of the programming languages. However, most programming languages lack this, and therefore, the programmer has to write the necessary code to implement this function.

Some patents of hardware implementation of popcount have been found in the literature. Ashkenazi [Ashkenazi, 1996] developed a circuit that performs the popcount based on Carry Save Adders (**CSA**) (see Fig. 1). The circuit accepts a word of 16-bit, and outputs 5-bit as a number of set bits, the circuit is comprised of three levels of **CSA**s, the outputs are forwarded to a fourth level that gives the final number of "1's" in the given word.

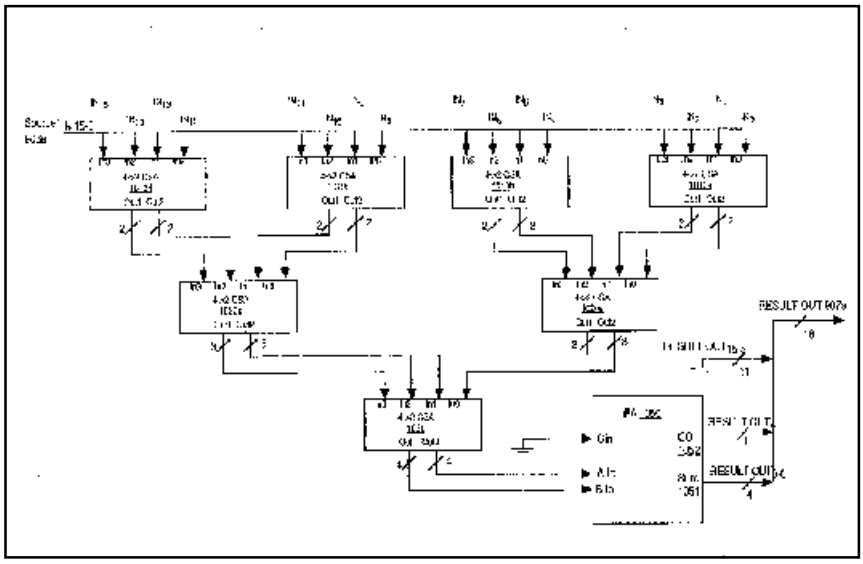

Figure 1: Ashkenazi Circuit [Ashkenazi, 1996]

Another invention proposed by Hossain [Hossain, 2004], which also based on **CSA**s. The circuit of this patent accepts 16-bit input and gives 5-bit output, the circuit is comprised of two levels of **CSA**s and a third level of a 4-bit adder (see Fig. 2).

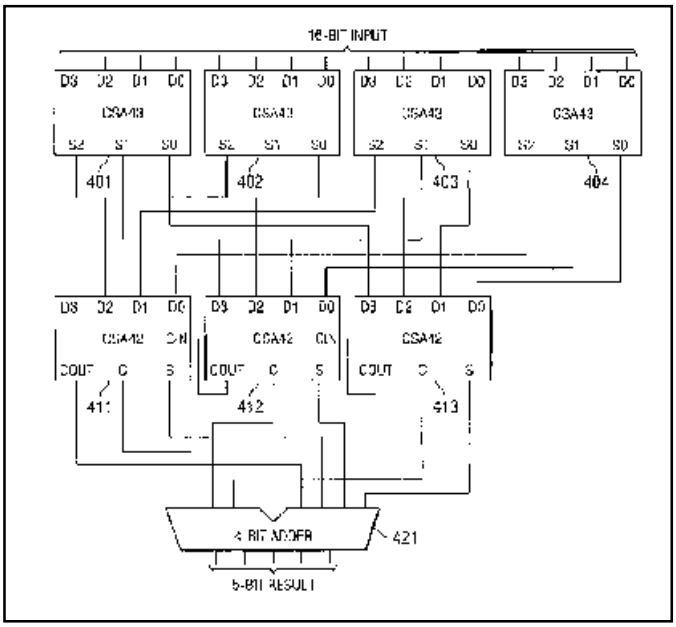

Figure 2: Hossain Circuit [Hossain, 2004]

Both of previous circuits are designed for inputs of 16 bits. This paper will propose a competitor architecture that can be applied to words of 16 bits or higher. The proposed architecture is evaluated and compared with other implementations.

3. SUGGESTED TECHNIQUE

This section will propose a new circuit for bit-counting. The proposed implementation to count the number of "1's" in a word of 8-bit is based on components of Half Adders (**HA**), and Full Adders (**FA**). A word of 8-bit is the input to the proposed circuit, and 4-bit output has been made to count number of "1's" which ranges from 0 to 8. Fig. 3 illustrates the proposed circuit to count number of "1's" in a word of 8 bits.

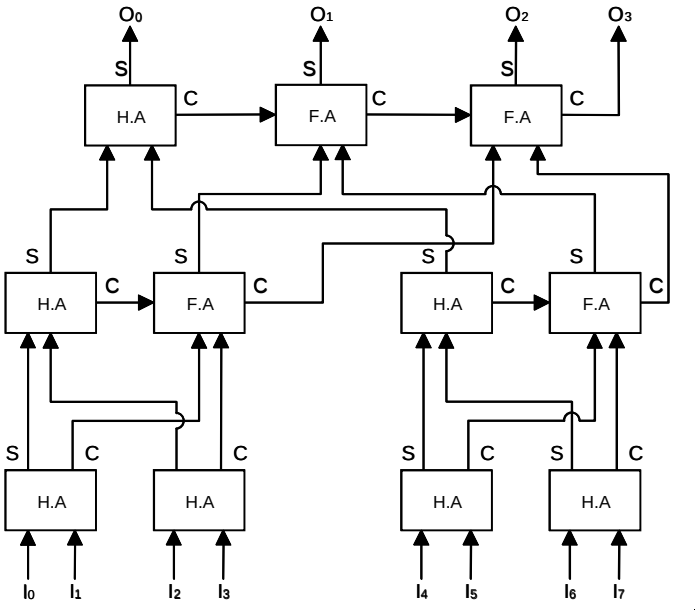

Figure 3: The proposed circuit of 8 inputs

The proposed design in Figure 3 consists of 3 layers. The bottom layer contains 4 **HA**s where each HA consists of (1 **XOR** ,1 **AND)** gates. The total number of logic gates in the bottom layer will be 4 * 2 = 8 gates. The middle layer consists of 2 **FA** and 2 **HA**. Since each **FA** consists of 5 logic gates and each **HA** consists of 2 gates, then the total number of gates in the middle layer is 5*2 + 2 * 2 = 14. The top layer will contain 5 * 2 + 2 * 1 = 12 logic gate. Thus, the overall summation of logic gates for figure No. 3 is 8 + 14 + 12 = 34 logic gates.

The above design can be further enhanced by two improvements. The first improvement will be applied to the second layer in Figure 3 by replacing it. This layer which consists of two **FA**s and two **HA**s will be replaced by four **HA**s and two **OR** gates as can be seen in Figure 4. Note that second layer in Figure 3 consists of 14 gates. The replacement of this layer by four **HA**s and 2 **OR** gates reduces the total number of gates to 4 * 2 + 2 = 10 gates. In other words, we are reducing the number of logic gates by 4 logic gates. This improvement takes advantage from the observation that the truth table of two variables has many cases that will not occur at all. For example, suppose that in Figure 3 the value of (I_0, I_1) is 11, then the value of the output from the **HA** will be 10. Note that it is not possible for the **HA** to have an output of 11. Note that in Figure 4, the leftmost component is **HA** rather than **FA**. This will not affect the result since the carry to the **HA** is always zero. This modification will reduce the number of circuitry in our design. The modified proposed design will be as follows:

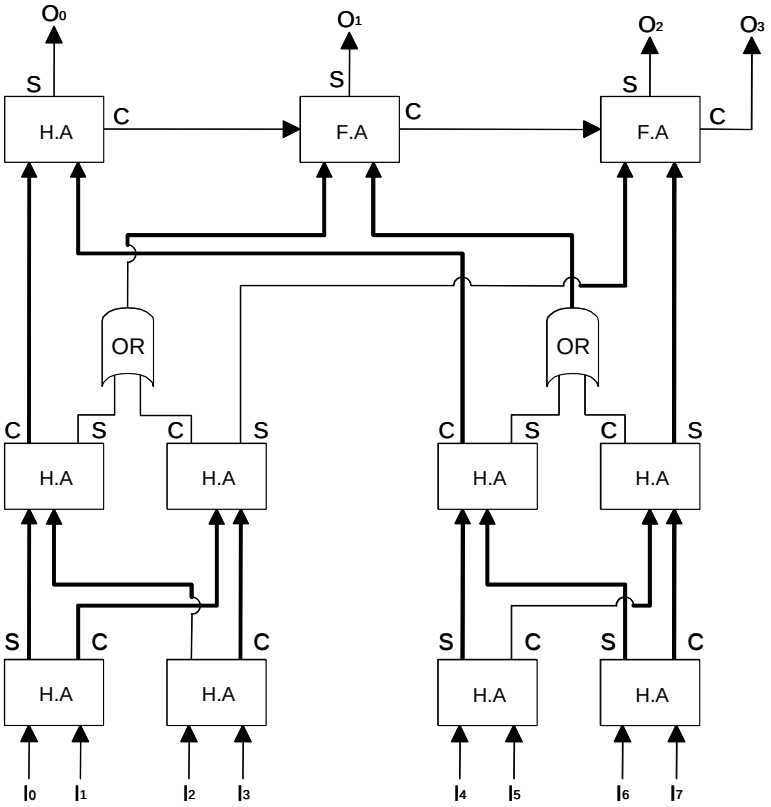

Figure 4: The modified proposed design of 8 inputs.

The total number of logic gates for Figure No. 4 is as follows : 5 * 2 **FA** + 2 * 9 **HA** + 2 **OR** = 30 logic gates. The design of Figure 4 is for an input of 8-bit. However, we can duplicate the number of input bits (i.e., 16-bit) by duplicating the circuit above and adding another layer which consists of three **FA**s and one **HA** to get an output of 5 bits, as shown below in Figure 5.

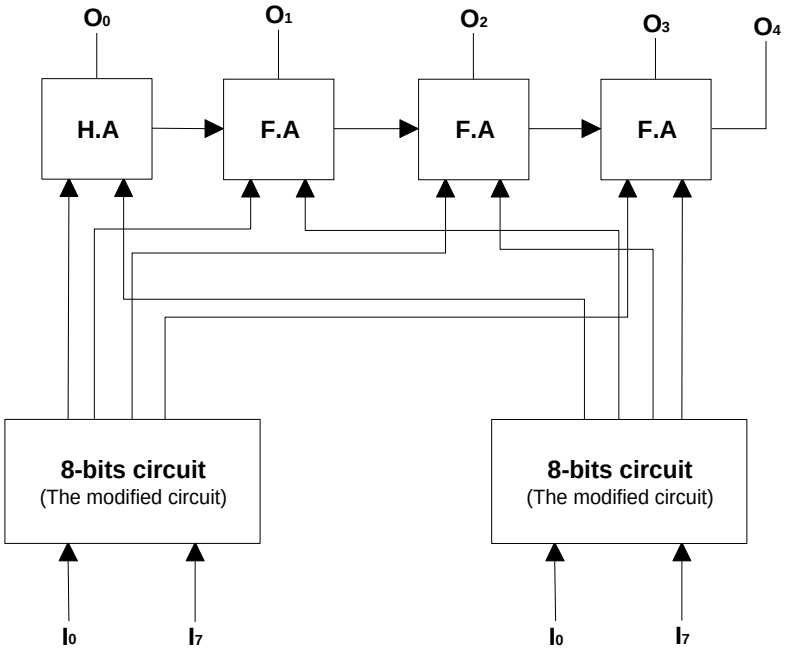

Figure 5: An extended version that counts "1's" in a word of 16-bit

To compute the total number of logic gates in Figure 5, the bottom layer has two 8-bit circuits where each circuit consist of 30 logic gates as was calculated previously. Thus, the total number of logic gates for the bottom layer will be 30 * 2 = 60 gates. Adding the top layer which is 17 (2 * 1 **HA** + 5 * 3 **FA)** logic gates makes the overall total equal to 77 logic gates. This number 77 will be used to fill the last cell in the last column of Table No. (2). An expansion of the circuit in figure 5 could be done to handle word length of 32, 64, or 128 bits, the same way that has been done when expanding 8-bits to 16-bits. For example, to design a circuit of 32 bits, duplicate the number of input bits (i.e., 16-bit) by duplicating the circuit above and adding another layer which consists of four **FA**s and one **HA** to get an output of 5 bits. We can also work in this direction to expand it to 64, 128 bit and so on.

4. PERFORMANCE ANALYSIS

The authors used the Hardware Description Language (HDL) to verify the performance (speed and gate count) of the proposed bit-counting architecture. We used the Verilog codes to implement these four architectures and use the Design Compiler (DC) to synthesize them. Table No. (1) shows the experimental results.

Table No. (1): The computations of Core area and the critical path using Verilog HDL

Comparison Item / Architecture	Core Area	Critical Path
Ashkenazi	1430 μm^2	2.93 ns
Hossain	1430 μm^2	2.25 ns
Proposed Design	1455 μm^2	1.04 ns
Modified Proposed Design	1346 μm^2	1.04 ns

Where the process is TSMC .18 µm CMOS and the wire load is equal to 10.

Table No. (1) shows that the critical path in our method is much better than the other two designs. The computation was done using Verilog HDL.

Another comparison between different implementations was done based on the total number of logic gates used by each implementation. We did comparisons based on an input of 16-bits. In table 2, a list of the total number of logic gates used in Figure 1, Figure 2, and Figure 5 using modified proposed design is shown is shown below.

Table No. (2): Total number of gates used by different approaches

Bit-counting Circuit of size 16-bit input	**No. of FAs**	**Total No. of gates in the FAs**	**Extra components**	**Total No. of gates in the extra Components**	**Total No. of gates**
Ashkenazi design	7 **CSA**s * 2 **FA**= 14	14 **FA***5 = 70	Special circuit of 8-bit input	4 **FA**s * 5 = 20	
		70		20	90
Hossain design	(5 **CSA**s Type 1) * 2 + (2 CSAs Type 2) * 2 = 14	14 **FA***5 = 70	4-bit adder and 2 **HA** from **CSA** Type 2	4 **FA**s * 5 = 20 + 2 * 2	
		70		24	94
Proposed design	11	11 **FA***5= 55	15 **HA**s	15 **HA** *2 = 30	

		55		30	85
Modified proposed design	7	7 **FA** *5 = 35	19 **HA**s + 4 **OR**s	19 **HA** *2 + 4 = 42	
		35		42	77

The design of Ashkenazi used circuits based on **CSA**s as a basic component [Ashkenazi, 1996]. Each **CSA** is composed of two **FA**s as can be seen in Figure 8. Since Ashkenazi design contains 7 **CSA**s where each one consists of 2 **FA**s, then the total number of **FA**s in Ashkenazi implementation is 14 **FA**s. However, each **FA** consists of 5 logic gates. This means that the **CSA** contains a total of at least 14 * 5 = 70 logic gates. The extra components also contains 20 logic gates. Thus, the overall total of Ashkenazi design is 90 logic gates.

To understand the second row of Table No. (2), which belong to the design of Hossain, we should notice the existence of two types of **CSA**s in Figure No. (2). Type 1: (4 input, 3 output) which consists of 2 **FA**s. Type 2: (5 inputs, 3 outputs) which consists of 2 **FA**s and 1 **HA**. Thefore, the total number of gates in the **CSA**s is equal to 7 **CSA**s * 2 **FA**s = 14 **FA**. These 14 **FA**s contain 14 * 5 = 70 logic gate. Adding the extra 2 **HA**s (resulted from Type 2 CSA) where each one consists of two gates, the extra components which consists of 4 **FA**s brings the total to 94 gates.

As seen from table No. (2), the modified proposed circuit is a good competitor for the two above mentioned hardware inventions, since it has less number of gates and the same functionality as the other two patent's circuitry that gives the number of "1s" in a computer word.

The suggested design may be used for any number of inputs rather than 16 bits. For example, we can duplicate our circuit to count words of lengths 32 by adding an extra level of 4 **FA**s and one **HA**, while this will not work with the above-mentioned inventions, since they are restricted to 16-bit length only and duplicating the circuits will not work, since it give a not accurate number of outputs. The design of 32 inputs can also be used to construct implementations for 64 inputs by using 2 blocks of it and some extra **FA**s and one **HA.** In fact, we investigate the number of necessary gates for a certain number of inputs using modified proposed circuit and we got the results that are listed in Figure No. 6.

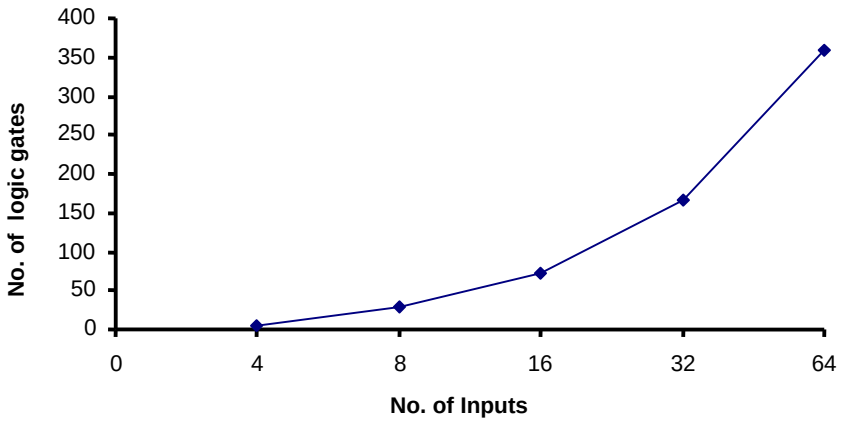

Figure 6: Number of logic gates related to the number of inputs

As an example, if we have a number of inputs equal to 64 bits, then we need around 380 logic gates for this design. Figure 6 shows that our design is scalable.

The other factor, which was investigated, was the maximum delay in the design which is the longest path. The longest path is the number of required unit gate delay in the design. Figure 7, shows the longest path of Figure 3 and Figure 4 shaded.

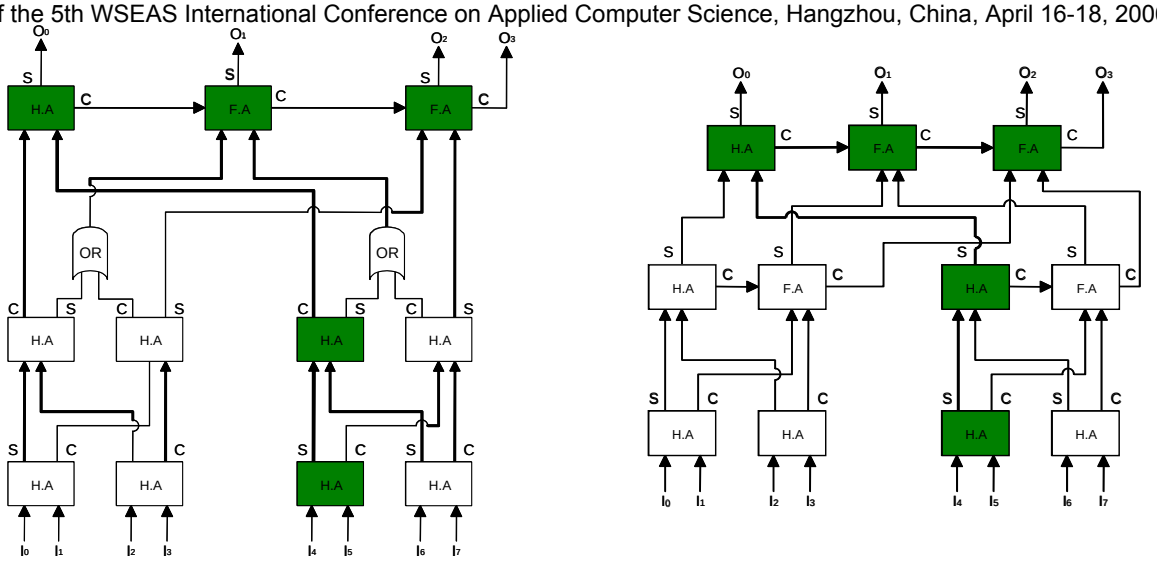

Figure 7: Longest path in Figure 3 and Figure 4.

As we can see from the previous figures, Ashkenazi design requires four layers [Ashkenazi, 1996], Hossain design requires three layers [Hossain, 2004] and our design requires four layers. These results listed in table (3). In addition, we found the number of logic delay that are needed for the whole popcount operation. Note that it is possible for a layer to have more than one sequential operation, which lead to more delays in the execution time. The results of each design is listed in table (3)

Table No. (3): logic gate delay for 16 bit input

Popcount hardware	**No. of delay gates**
Ashkenazi design	30
Hossain design	36
Proposed design	23
Modified Proposed design	23

When we filled the number of sequential operations, we have looked at each design carefully. Each **CSA** consists of two **FA**s as can be seen in Figure 6. Each **FA** consists of 5 gates, where the sequential delay goes through 3 gates. Therefore, the cost of each **CSA** with 4 inputs is to go through six logic gates. Hence, for Ashkenazi design, the number of sequential operations will be as follows: 6+6+6+3*4= 30. For Hossain design, it will be 6+6+6+6+3*4=24. For our design, the number of sequential operations is 23 for the proposed design and the modified proposed design respectively.

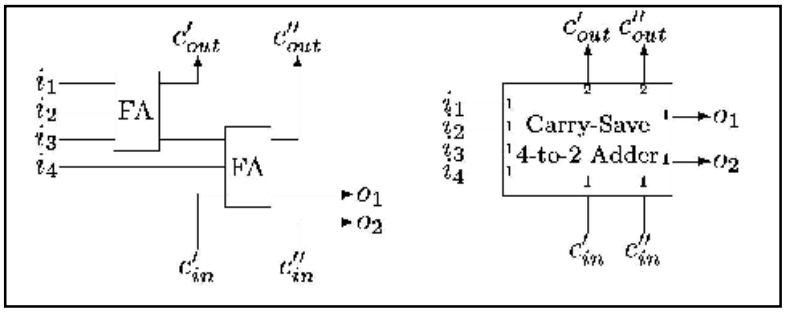

Figure 8: The internal design of the Carry Save Adder

5. DISCUSSION

It is very clear that the proposed modified design requires fewer circuits and less delay than other patents. Note that the number of layers does not give exact indication of the number of unit gate delay unless further analysis of the components of each layer is known.

The overall popcount performance is a product of the machine, operating system, implementation, and compiler design. These factors are related to each other and they may affect the execution time when the popcount runs on real machines. The authors recommend to add the this instruction to all hardware implementations. The addition of the proposed circuitry will encourage different programming languages to make the "popcount" function part of their language or include it in a library if we want to avoid the changes in the definition of the language.

6. CONCLUSIONS

Bit-counting used in various applications such as information retrieval, file processing and coding theory [El-Qawasmeh, 2001]. In information retrieval, the number of "1's" in an array of binary words presents the characteristic function of a set of retrieved values. This can be used to decide whether to constrain or broaden the search criteria to ensure selection of the desired items. The comparison operation between two given files in terms of Hamming distance can also employ popcount operation.

In this paper, we have presented a hardware technique for counting the number of "1's" that is faster than other existing implementation. The speed varies depending on the nature of the numbers. Merging the hardware popcount implementation with software techniques further improves performance.

The proposed design has many advantages. These include regular design, less number of gates, which imply fewer interconnections, and less number of charges. All in all, it will be cheaper than other competitors will.

REFERENCES

1- Ashkenazi, Y., "Method and Apparatus for Performing a Population Count Operation", U.S. Patent No. 5,541,865, issued July 30, 1996.

2- Balmer, K., "Ones Counting Circuit, Utilizing a Matrix of Interconnected Half-Adders, for Counting the Number of Ones in a Binary String of Image Data" U.S. Patent 5,339,447, Aug. 16, 1994.

3- Berkovich, S., El-Qawasmeh, E., Lapir, G., Mack, M., and Zincke, C. "Organization of Near Matching in Bit Attribute Matrix Applied to Associative Access Methods In Information Retrieval," Proc. of the 16th IASTED International Conference Applied Informatics, Garmisch-Partenkirchen, Germany: 62-64, 1998.

4- Berkovich, S., Lapir, G., and Mack, M. "A Bit-Counting Algorithm Using the Frequency Division Principle," Software: Practice and Experience Vol. 30, No. 14, pp. 1531-1540, 2000.

5- El-Qawasmeh, E. "Performance Investigation of Bit-Counting Algorithms with a Speedup to Lookup Table," Journal of Research and Practice in Information Technology, Australia, Vol. 32, No. 3/4, pp. 215-230, Jun., 2001.

6- El-Qawasmeh, E. and Hemidi, I. "Performance Investigation of Hamming Distance Bit Vertical Counter Applied to Associative Access Methods in Information Retrieval," Journal of the American Society for Information Science, USA, Vol. 51, No.5, pp. 427-432, 2000.

7- Goldberg, D., Deb, k., and Clark, J. “Genetic Algorithms, Noise, and the Sizing of Populations,'' Complex Systems, Vol. 6, pp. 333-362, 1992.

8- Gutman, R. “Bit-Parallelism,” Dr. Dobb’s Journal, Vol. 25, No. 9, pp.133-134, 2000.

9- Hossain, R U.S. Patent No. 6,729,168 entitled ''Circuit for Determining the Number of Logical One Values on a Data Bus'', issued May 4, 2004.

10- HP website. http://www.hp.com. Visited on June 19, 2005.

11- Kornerup, P., Reviewing 4 to-2 Adders for Multi-Operand Addition Danish National Scientific Foundation, grant no. 9801811.

12- Morris, D., “Computer Hardware Instruction and Method for Computing Population Counts”, U.S. Patent 5,717,616, issued Feb. 10, 1998.

13- Reingold, E., Nievergeit, J. and Deo, N. Combinatorial Algorithms: Theory and Practice. Englewood Cliffs, New Jersey 07632, Prentice Hall, 1997.

14- Stelling, P., Martel, C., Oklobdzija, V, and Ravi, R. “Optimal Circuits for Parallel Multipliers,” IEEE Trans. on Computers, Vo;. C47, No. 3, pp. 273-285, March, 1998.

15- Texas Instruments website, http://dspvillage.ti.com/. Visited on June 19,2005.

Proceedings of the 5th WSEAS International Conference on Applied Computer Science, Hangzhou, China, April 16-18, 2006 (pp118-128)