THE BIT PARTITIONED MULTIOPERAND ADDITION CIRCUIT FOR A CO-PROCESSOR

Haji Khan Soomro, M.M. Khushk & Anwaruddin Ujjan
Institute of Physics & Technology
University of Sindh, Sindh, Pakistan

Abstract

The use of Co-processors is becoming common in microprocessor based computer systems. Co-processors are used primarily to improve arithmetic handling capability of the main processor. The use of multioperand addition circuit based on Bit Partitioned technique can further enhance the performance of a processor to handle large brusts of data.

This paper describes the methodology and design used in the Bit Partitioned addition circuit which was implemented in the Lab.

Introduction

There are several approaches which had been implemented in arithmetic processor of a computer to deal with addition of multiple operands. One method is accumulation of the sum through repeated addition. In this case single adder continues to receive sequentially each number and the process is carried on until all numbers are exhausted. This solution is in fact very simple and requires less hardware but takes large execution time. Another approach handles the addition of numbers by pairs and then addition of resulting sums by pairs and repetition of that process until the final sum is reached. Although this method needs extra registers to store the intermediate sums for a while but there is no significant speed up in the process of addition (1).

Another widely used method is the carry save technique in which carry propagation is saved until all additions are completed and then a final cycle is taken to include the previously saved carries. This method is useful for limited number of

operands because their increase results in more carry save steps. As a consequence, saving of carries and then their retrieval contributes significantly to the time delay (2,3).

Under such corcusnstances, the use of Bit Partitioned circuit is more efficient and offers modularity

The Principle of Bit Partitioned addition technique

In this scheme the operands to be added are divided into suitable column wise groups called partitions which are added separately but simultaneously. The result so obtained out of each portioned block is then passed on to a fast carry Look Ahead circuit to form the final sum. This process can be illustrated by an example to demonstrate addition of 5 numbers each with 8 bits:

$A_7A_6A_5A_4A_3A_2A_1A_0$ = 0 0 |101| 1 1 1

$B_7B_6B_5B_4B_3B_2B_1B_0$ = 1 1 |110| 1 0 1

$C_7C_6C_5C_4C_3C_2C_1C_0$ = 1 0 |111| 1 0 0

$D_7D_6D_5D_4D_3D_2D_1D_0$ = 0 1 |000| 1 1 0

$E_7E_6E_5E_4E_3E_2E_1E_0$ = 1 0 |101| 1 0 1

Each column has 5 bits and there are eight columns in all. Groups of adjacent columns of any convenient size can be made. If it is decided that each group be of three columns then the partition size becomes equal to three, except the last with two columns only. Each group of columns is processed separately. Consider the addition of right most partition as reproduced below:

$A_2A_1A_0$ = 1 1 1

$B_2B_1B_0$ = 1 0 1

$C_2C_1C_0$ = 1 0 0

$D_2D_1D_0$ = 1 1 0

$E_2E_1E_0$ = 1 0 1

Imagine that left most column is brought for addition. The sum of bits in it is $(101)_2$ or decimal 5 as there are five 1's. Then next column sum is $(010)_2$ or decimal

2. The sum of bits in the last column is $(011)_2$ or decimal 3. The results of three columns can be got easily if the sum of each column is rightly positioned and then added together as shown below:

```
1 0 1        Sum from left most column
  0 1 0      Sum from middle column
    0 1 1    Sum from right most column
+
-----------
  1 1 0 1 1
```

In the same way next two partitions can be added. The sum obtained from the middle partition is 10111 and that from left most partition is 1000. Finally one cycle of operation is required to get the overall result if each partition sum is appropriately positioned and added:

```
01000        Sum from left most partition
 10111       Sum from middle partition
  11011      Sum from right most partition
----------
1011010011
```

The above binary number represents the equivalent decimal number 723. This is the correct sum of all the five 8 bit numbers and can be verified by adding their decimal equivalents: $173 + 70 + 188 + 245 + 47 = 723$.

Digital Design Considerations

The Bit Partitioned adder modules were designed and implemented in the Lab. with partition size of 3 confined to addition of four numbers. The working of the circuit can be easily understood by refering to the block diagram shown in fig.1.

The sections of circuit are: The Shift register, decoder, ROM, network of adders, shifting circuit and clocking circuit. The shift register is used to transmit a four bit column to 1 of 16 decoder. Depending upon the setup of bits in the column, the corresponding output line of the decoder will fall to low state. As a result, the ROM will produce the sum of column bits, which is restored in the ROM, and will be

triggered to the three data outputs of ROM. These data outputs are received by 4 bits parallel addition and shifting circuit to appropriately shift, position and save them at proper clock transition.

The same clock transition allows the shift register to pass on second column of 4 bits and the process is carried as before. Then third column of bits is processed. Finally the sum will appear at the outputs: $S_4S_3S_2S_1S_0$.

The ROM for column addition

The ROM used in this case consists of sixteen 3 bit locations each requiring separate address line to bring the contents of a particular location at the output of data lines.

The 16×3 ROM has been implemented using semiconductor diodes. The 4 bit column may vary between all bits being "0000" to all bits being "1111" or any intermediate setup as shown in the table of figure 2. These 4 bits are treated as address of the location containing the required sum.

The realisation of ROM table is shown in the circuit of figure 3. The presence of a diode in each position represents a logic 0 and its absence as logic 1. In this way 29 diodes are required.

By careful inspection it appears that the complete right most column of diodes can be eliminated by feeding the output of corresponding data line to an inverter. Then only one diode is required at location "1111". In this way 14 diodes can be saved as shown in fig.4.

In figure 1, the decoder is used to decode the setup of bits at its inputs and then sends the active low signal at the corresponding output line which activates the relevant ROM location to trigger its contents on three data lines.

4 Bit addition and Shifting Network

The function of this module is to receive, shift and save all the partial sums from ROM column adder and finally it should produce the appropriate sum. The operation of this circuit can be explained when coupled with other units as shown in figure 5. Initially all flip-flops should be cleared. First column of bits is then applied to ROM which produces corresponding output at C_2C_1 and Sum. The sum bit is the

L.S.B which directly goes to S_0 and also appears at input of flip-flop FF1. C1 is connected to half adder H.A.1, while C_2 is applied to full adder F.A.1. Now a clock transition is required to load the data present at input of each flip-flop. In the same way second and third columns of bits are applied and the result will appear at $S_4S_3S_2S_1S_0$ outputs.

The Complete Circuit

The detailed circuit is shown in figure 6. It includes the clocking circuit. The required full adders and half adders are constructed by using logic gates.

A set of four D flip flops has been used which are sensitive to leading edge of clock transition. As such the positive edge of the clock should be very sharp and narrow so that false triggering should not happen. For this purpose, in the clocking circuit, one of the outputs has been obtained from "BOUNCE LESS" Circuit so that multiple transitions are eliminated. This signal is then passed through an inverter and an AND gate. The other input A to the AND gate comes from the inverter. The output of the AND gate should stay high only when its both inputs are high. Thus there is a brief interval during which both inputs to the AND gate are high producing an instant high signal which will fall back after 10ns. This signal works as an excellent positive clock transition for all the D-flip flops.

Conclusion

There are different methods in practice to carry on multioperand additions. A common drawback in them is the increase in execution time with the increase in number of operands. The Bit Partitioned technique is different altogether. When the number of operands increases the length of each column of a partition increases but this does not contribute to increase in the time. This is in fact very attractive feature when execution speed has the prime importance.

However, the partition size is related to the time and its effect is prominent only for long operands with 32 or 64 bits where the designer has to choose between the partition size and the number of required modules.

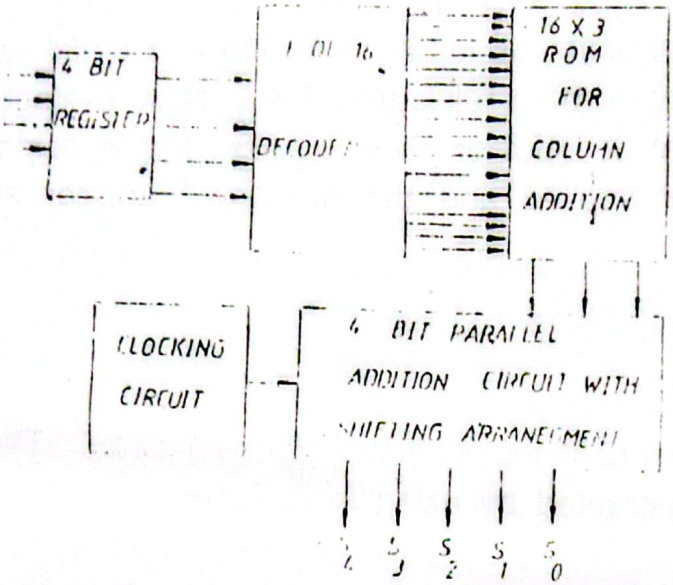


FIG.1 Shows the Block Diagram for Multoperand 8 Bit Partioned addition Circuit.

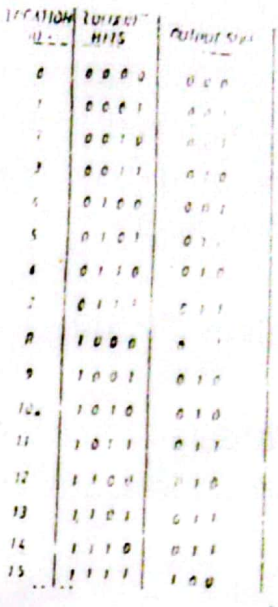


FIG.2 ROM Table for Column Addition.

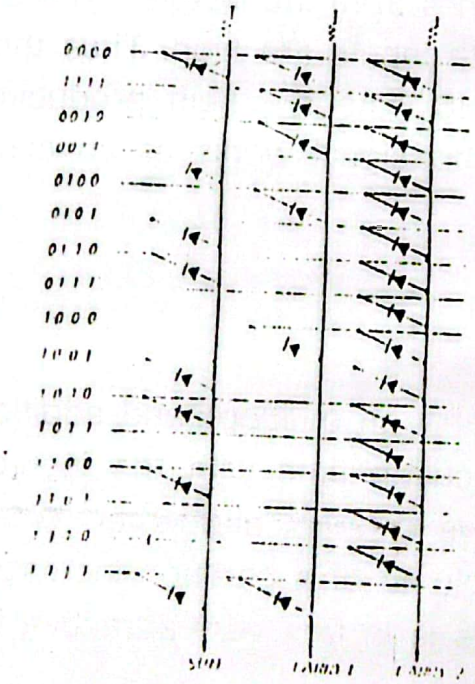
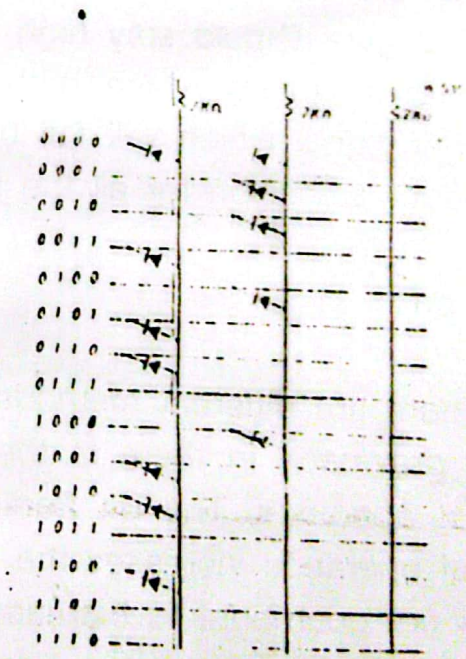


FIG.3 16×7 ROM needs 29 Diodes.



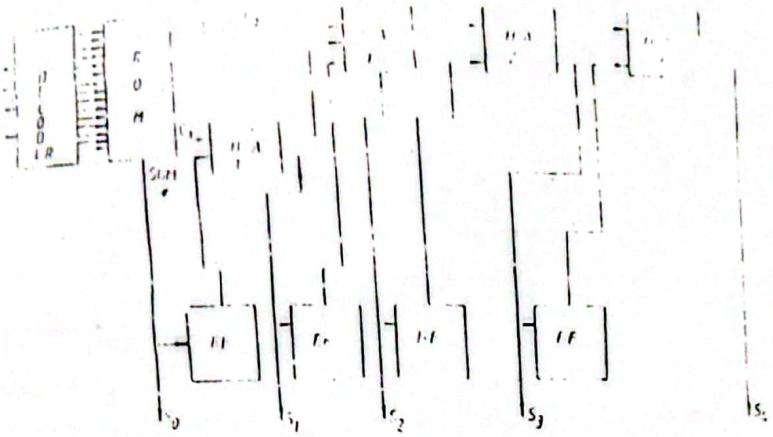


FIG.5 Addition & Shifting Circuit

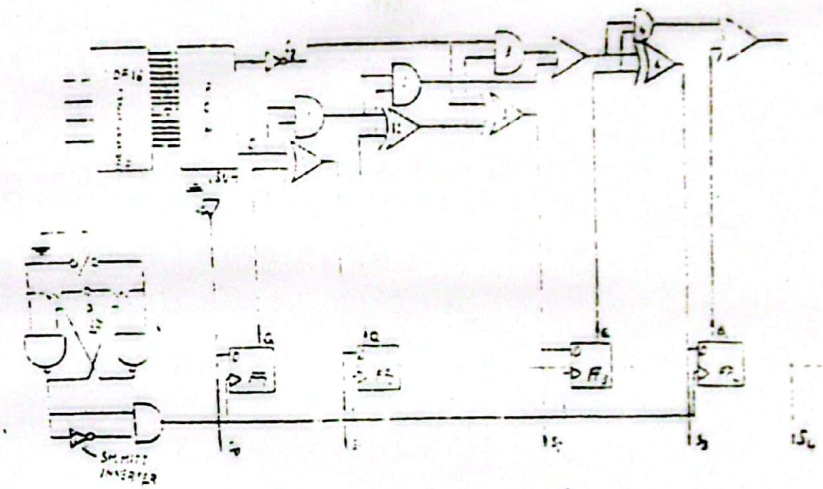


FIG.6 Multioperand Bit Partioned addition circuit.

References

1. **Crowley D., and Amaratunga (1986).** *"Pipe lined carry look Ahead adders"*. electronic letters, Vol **22**. 661–662.

2. **Young, H. and Allen C., (1986).** *"Recursive addition and its parameterisation in VLSI"*. IEEE proceedings, Vol. **133**, 256–263.

3. **Bruce and Pomerance (1980).** *"Fast carry logic for digital computers"*. Papers in Electrical Engg. and Computer Science, Hutchinson & Ross Inc., 43–46.