***King Fahd University of Petroleum and Minerals***

***College of Computer Science and Engineering***

***Computer Engineering Department***

**COE 202: Digital Logic Design (3-0-3)**

**Term 131 (Fall 2013)**

**Major Exam II**

**Saturday November 30, 2013**

**Time: 120 minutes, Total Pages: 12**

**Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_**

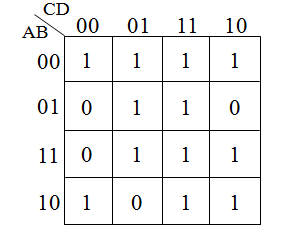
**Notes:**

* Do not open the exam book until instructed
* **Calculators are not allowed** (*basic, advanced, cell phones, etc*.)
* Answer all questions
* All steps must be shown
* Any assumptions made must be clearly stated

|  |  |  |
| --- | --- | --- |
| **Question** | **Maximum Points** | **Your Points** |
| **1** | **14** |  |
| **2** | **8** |  |
| **3** | **20** |  |
| **4** | **12** |  |
| **5** | **15** |  |
| **6** | **16** |  |
| **Total** | **85** |  |

**Question 1. (14 points)**

For the following Boolean function shown in the K-map:



# F(A, B, C, D)=m(0, 1, 2, 3, 5, 7, 8, 10, 11, 13, 14, 15)

# Identify all possible *prime implicants* of F and indicate which of these is essential.

# Simplify the Boolean function F into a minimal sum-of-products expression.

# Simplify the Boolean function F into a minimal product-of-sums expression.

**Question 2. (8 Points)**

It is required to design a circuit to compute the equation Z=2\*X-Y, where X and Y are two n-bit unsigned numbers. The circuit can be designed in a modular manner where it is designed for one bit and replicated *n* times. A 1-bit circuit block diagram is given below:



The meaning of the values of Bi and Ci is given in the table below:

|  |  |  |
| --- | --- | --- |
| Bi | Ci | Meaning |
| 0 | 0 | There is no carry or borrow |
| 0 | 1 | There is a carry of 1 |
| 1 | 0 | There is a borrow of 1 |
| 1 | 1 | This condition does not occur |

For example, if Xi=1 and Yi=1, then we should have Zi=1, Bi+1=0 and Ci+1=0. If Xi=0 and Yi=1, then we should have Zi=1, Bi+1=1 and Ci+1=0.

# The figure below shows how a 4-bit Z=2\*X-Y circuit is implemented using 4 copies of the basic 1-bit cell.



# Derive the truth table for the basic one-bit cell. You do not need to derive the equations for the circuit.

**Question 3. (20 Points)**

a. Fill in all blank cells in the two tables below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Binary | Equivalent decimal value with the binary interpreted as: | | | | |
| Unsigned number | Signed-magnitude number | Signed-1’s complement number | Signed-2’s complement number | BCD number |
| 10000000 |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Decimal | Binary representation in 8 bits: | | |
| Signed-magnitude notation | Signed-1’s complement notation | Signed-2’s complement notation |
| - 75 |  |  |  |

b. Using 2’s-complement signed arithmetic in 5 bits, do the following operations **in binary**. Show all your work, and:

- Verify that you get the expected decimal results.

- Check for overflow and mark clearly any overflow occurrences.

|  |  |
| --- | --- |
| (i)  00111  - 10101 | (ii)  10110  - 10011 |
| (iii)  (+6)  + (-11) | (iv)  (-9)  - (+7) |

c. Consider the signed 2’s complement arithmetic operation A - B in 6 bits. With B = 101100, the largest value allowed for A in order to avoid the occurrence of overflow is (\_\_\_\_\_\_\_\_\_)2.

**Question 4. (12 Points)**

1. **(4 points)** Considering the following circuit, provide a minimized **SOP** expression of *F*(*A*, *B*, *C*, *D*).



1. **(4 points)** **Using only NAND gates**, redraw the following circuit to show a multi-level **NAND** circuit. Only the **true** form of each input variable is available.



1. **(4 points)** Implement using a 4-to-1 MUX. Show how you obtained your solution, and properly label all input and output lines.

**Question 5. (15 Points)**

A Triple adder **circuit** adds three *n-*bit numbers a, b, and d. The triple adder circuit consists of *n*-stages of the single bit circuit slice shown in Fig. (i) (called 3-Add). The *ith* stage receives 5 inputs 3 of which are the *ith* bits of a, b, and d and the other two are carry-in inputs C0*i* and C1*i*. It has 3 outputs; one sum bit (S*i*) and two carry-out bits C0*i+1* and C1*i+1*.

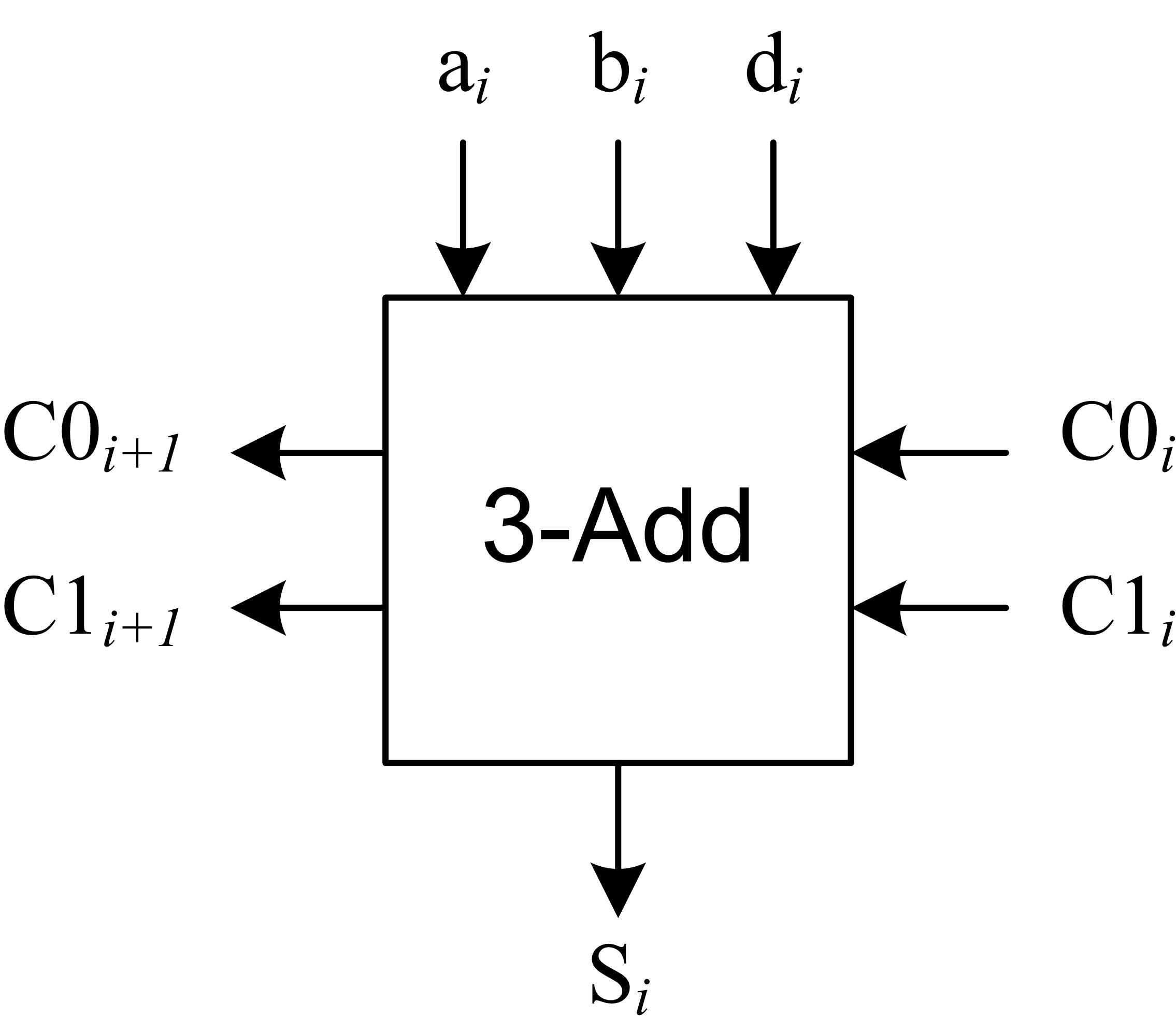


Fig (i)

Fig. (ii) shows the *n-*bit Triple adder circuit

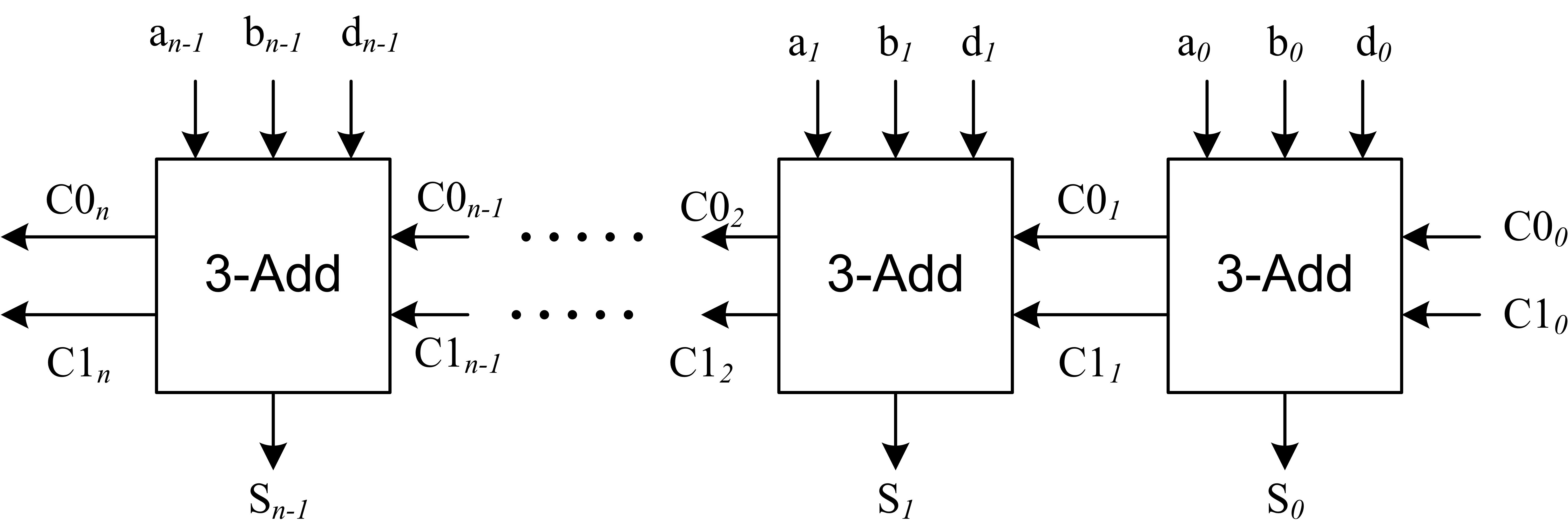


Fig (ii)

**Fig. (iii) shows an example for such addition.**

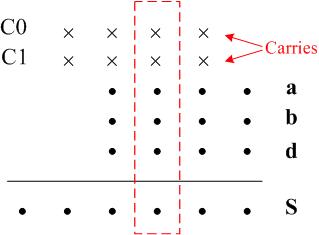
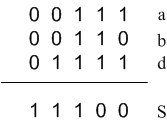
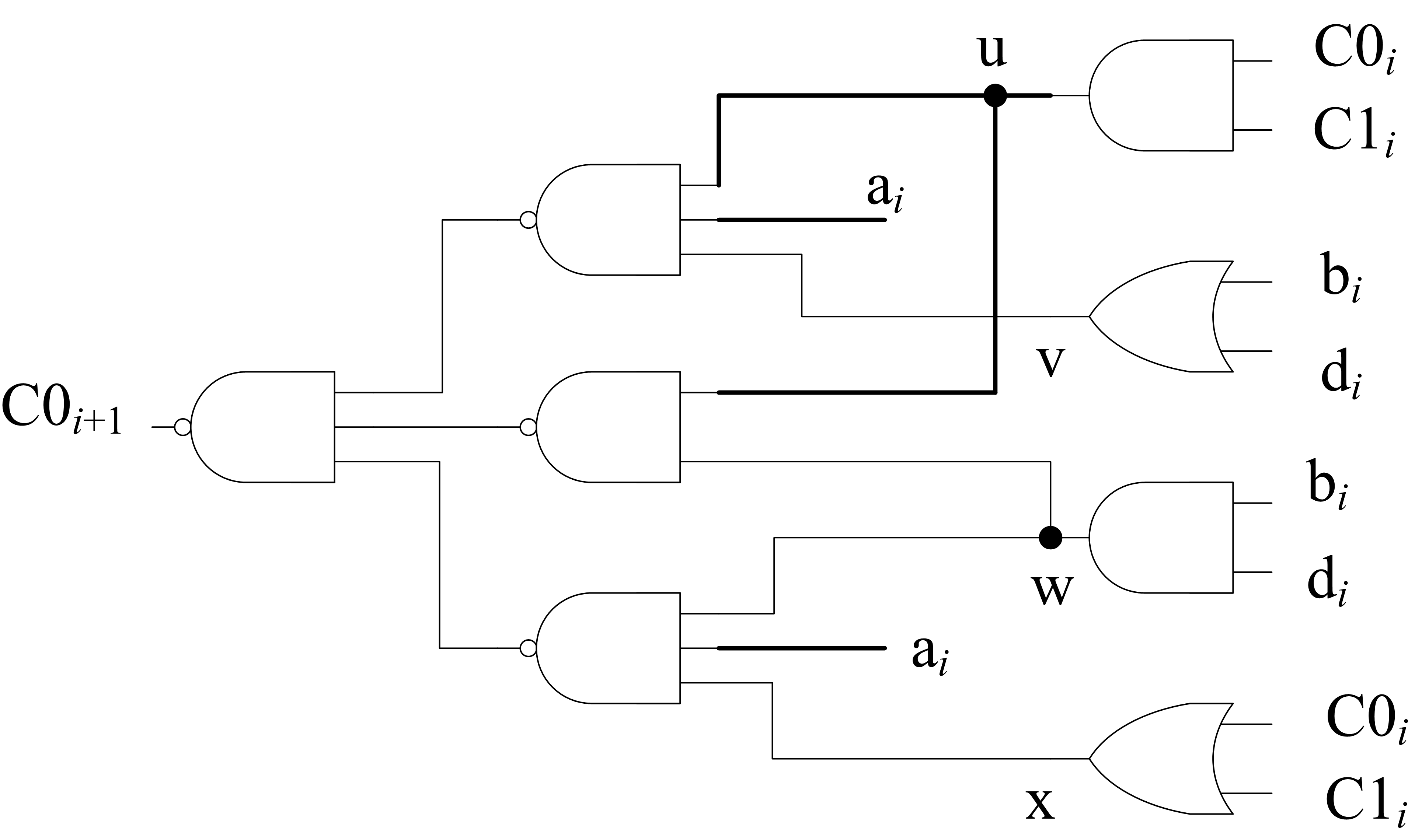
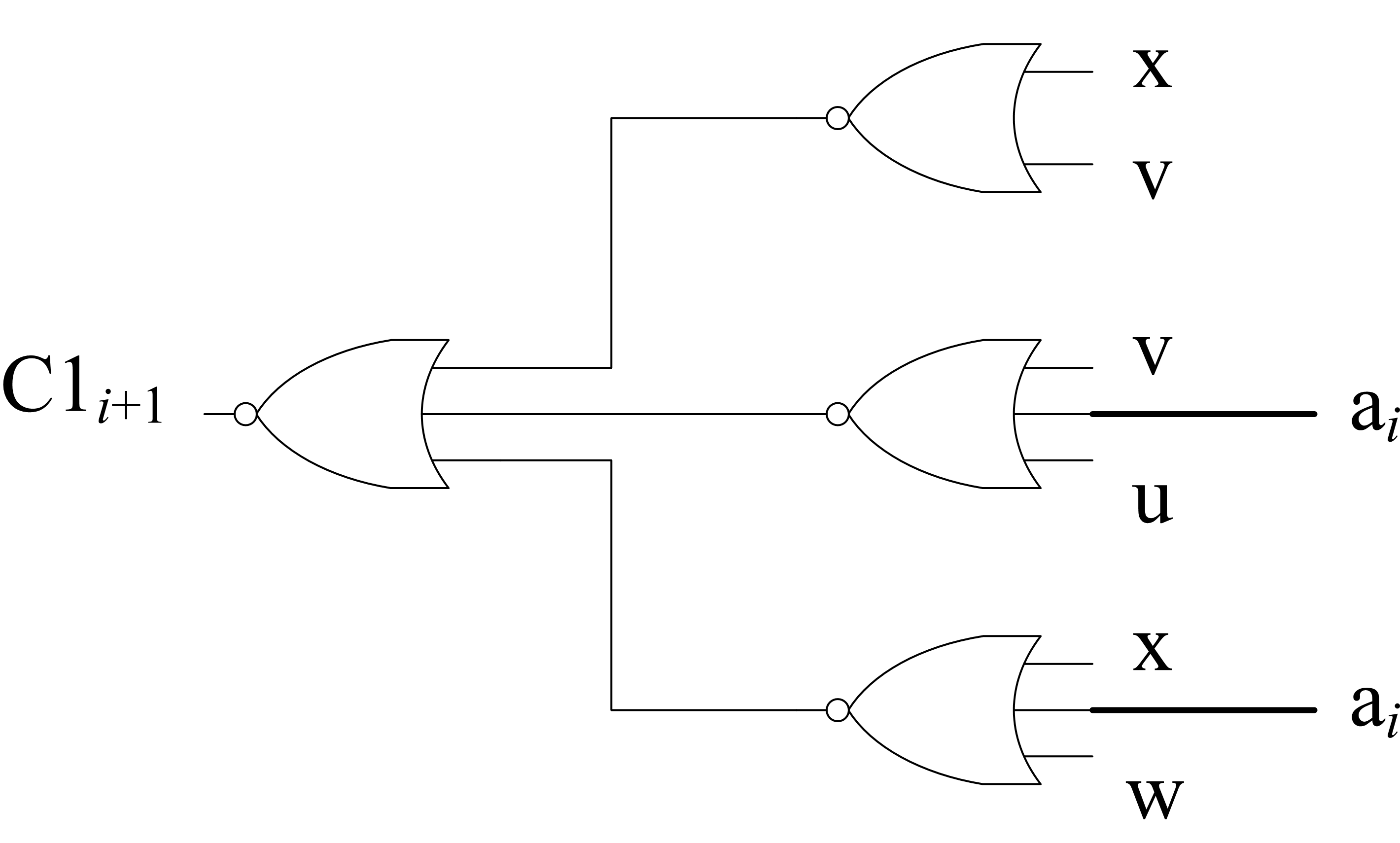
 

Fig. (iii)

**T**he circuits used togenerate the output carry bits are shown in Fig. (i**v**)**. a**nswer the following:

**Fig. (iv)**

|  |  |
| --- | --- |
| Gate | Delay |
| **AND , NAND** | **1 ns** |
| **OR , NOR** | **3 ns** |
| **XOR** | **4 ns** |
| **Table (i)** | |

1. Using the gate propagation delays of Table (i)**,** what is the carry propagation delay *per single stage* **for both of the output carries (C0 and C1)?** (5 Points)
2. For the *n-*bit triple adder circuit of Fig. (ii), assuming a 12ns delay from the *ith* input carries to the *ith* sum signal (S*i*), calculate the worst case delay to generate the *n*-bit sum (S*n-1* S*n-2…..* S*1* S*0*) of the three *n*-bit operands. (5 Points)
3. The *ith* output sum bit is given by , select one of the following logic implementations of S*i* to yield the fastest *n*-bit triple adder. *You must Label the 5-inputs of this circuit* (as a*i*, b*i*, d*i*, C0*i*, C1*i*) and **justify** your answer. (5 Points)

|  |  |  |
| --- | --- | --- |
| Sa.gif  (a) | Sb.gif  (b) | Sc.gif  (c) |

**Question 6. (16 Points)**

|  |  |
| --- | --- |
| **X** | **Y** |
| 0 | 0 |
| 1 | 3 |
| 2 | 6 |
| 3 | 1 |
| 4 | 4 |
| 5 | 7 |
| 6 | 2 |
| 7 | 5 |

1. Design a circuit that has a three-bit input X and three-bit output Y. Both X and Y represent the integers 0 to 7 (i.e., ). Using a *single* decoder and a *single* encoder of appropriate sizes, show how can you build a circuit that performs the function **[Y = 3X mod 8]**. Make sure you *label all signals*. The truth table for this circuit is shown in decimal notation. **[4 pts]**
2. Construct a 16-to-1 multiplexer using the minimum number of 4-to-1 multiplexers. **[5 pts]**

|  |  |
| --- | --- |
| **S1S0** | **O** |
| 00 | A+B |
| 01 | A-B |
| 10 | A+1 |
| 11 | A-1 |

1. Using *only* MSI parts, design a circuit that takes two 4-bit binary numbers A = A3A2A1A0 and B = B3B2B1B0 together with a 2-bit selection input S = S1S0. The circuit produces a 5-bit output O = O4O3O2O1O0 according to the shown table:

*Clearly label* all inputs and outputs of the MSI parts. **[7 pts]**