Name and Std ID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Lab Section:\_\_\_\_\_

Date:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**PRELAB:**

**Q1.** Add the following numbers then write them in decimal:

|  |  |  |
| --- | --- | --- |
| **Binary numbers to add**  **a3 a2 a1 a0 + b3 b2 b1 b0** | **Binary result**  **C0 S3 S2 S1 S0** | **Decimal conversion**  **N2 N1**  **(X3 X2 X1 X0) (X3 X2 X1 X0)** |
| 1001 + 0111 | 10000 | 16 |
| 1011 + 1001 |  |  |
| 1110 + 0101 |  |  |
| 0010 + 1110 |  |  |
| 1101 + 1011 |  |  |

**Q2.** Consider the five-bit binary result (C0, S3, S2, S1, S0) representation in the table above. We would like to represent each combination as its equivalent in two decimal digits, each of which can be represented in binary as shown in the following table. Finish filling in the following truth table.

**Binary Coded Decimal Converter**

**N2X3 N2X2 N2X1 N2X0**

**C0**

**S3**

**S2**

**S1**

**S0**

**N1X3 N1X2 N1X1 N1X0**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C0** | **S3** | **S2** | **S1** | **S0** | **Decimal** | | **N2X3** | **N2X2** | **N2X1** | **N2X0** | **N1X3** | **N1X2** | **N1X1** | **N1X0** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 9 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 2 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 3 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 | 4 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 5 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 | 6 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |  |  |  |  |  |  |  |  |  |  |
| 1 | 0 | 0 | 1 | 0 |  |  |  |  |  |  |  |  |  |  |
| 1 | 0 | 0 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |
| 1 | 0 | 1 | 0 | 0 |  |  |  |  |  |  |  |  |  |  |
| 1 | 0 | 1 | 0 | 1 |  |  |  |  |  |  |  |  |  |  |
| 1 | 0 | 1 | 1 | 0 |  |  |  |  |  |  |  |  |  |  |
| 1 | 0 | 1 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 0 | 0 | 0 |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 0 | 0 | 1 |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 0 | 1 | 0 |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 0 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 1 | 0 | 0 |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 1 | 0 | 1 |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 1 | 1 | 0 |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 1 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |

**Q3.** Find the logic expressions for N2X3, N2X2, N2X1, N2X0, N1X3, N1X2, N1X1, and N1X0 as a function of C0, S3, S2, S1 and S0:

N2X3 =

N2X2 =

N2X1 =

N2X0 =

N1X3 =

N1X2 =

N1X1 =

N1X0 =

**Q4.** Write the verilog code for the Binary Coded Decimal Converter from **Section 3.3** using the assign statement.

*Example:*

***module***

***input …***

***output …***

***assign …***

***endmodule***

TA Initials: \_\_\_\_\_\_\_\_\_

**LAB:**

Hardware demonstrates a good design. TA Initials: \_\_\_\_\_\_\_\_\_