1

2

3 Detailed design steps with K-Maps

3.1 Design Steps

1. Here we have to perform four types of arithmetic operations (Add, Negation, Add with carry, Increment) and two types of logical operations(logical AND , logical XOR).

2. For the adder,the first input X_i is either A_i or $\overline{A_i}$ or A_iB_i or $A_i \oplus B_i$.We used a 4×1 multiplexer for each bit. The MUX has S_{x_1} and S_{x_0} as selection bits.The selection bits are controlled by control bits(c_{s_2},c_{s_1},c_{s_0}).The details is shown in the truth table and k-Map.

3. For the adder,the second input Y_i is either B_i or 0. We used a 2×1 multiplexer for each bit. The mux has S_{y_0} as its selection bit.The selection bit is controlled by control bits(c_{s_2},c_{s_1},c_{s_0}).The details in shown in the truth table and k-Map.

4. The input carry(C_{in}) of the adder IC is either 0 or 1.It should be zero for negation, add with carry, and increment operations.This is also controlled by control bits(c_{s_2},c_{s_1},c_{s_0}).The details is shown in the truth table and k-Map.

5. In the arithmetic unit, the adder adds A, B with $C_{in} = 0$ and $C_{in} = 1$ for Add, Add with Carry operations respectively. The multiplexers provide Y_i as a 0 for increment operation.During negation,X_i=$\overline{A_i}$ and Y_i=0 and C_{in}=1.

6. During logical operations,X_i is transferred as Y_i=0 and C_{in}=0.

7. Zero flag, ZF is computed by adding the 4 output bits using 3 OR gates and then inverting $O_0 + O_1 + O_2 + O_3$ by 1 XOR gate($X \oplus 1 = \overline{X}$).

8. The carry flag(C) is directly obtained from C_{out} of the parallel adder

9. The sign flag(SF) is obtained from the MSB of the sum(S_3).

10. For the overflow flag,we needed C_{out} and C_3. C_{out} is directly accessible from the Adder IC. C_3 is calculated as below:

$$
\begin{gathered}
S_3 = X_3 \oplus Y_3 \oplus C_3 \\
S_3 \oplus C_3 = X_3 \oplus Y_3 \\
C_3 = X_3 \oplus Y_3 \oplus S_3 \\
\text{OF} = C_{out} \oplus C_3 \\
\text{OF} = C_{out} \oplus X_3 \oplus Y_3 \oplus S_3
\end{gathered}
$$

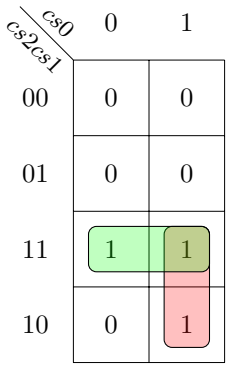
3.2 K-maps

We will be following Table **??** to construct the K-maps for selection bits of multiplexers and C_{in}.

3.2.1 K-maps for S_{x_1} and S_{x_0}

The IC of parallel adder takes X_i and Y_i and C_{in} as input.We need X_i as A_i or its complement or its logical changes with B.This values are received as X_i(output of the multiplexer) and the kmap for selection bits(S_{x_1} and S_{x_0}) of the multiplexer are as follows:

3.2.2 K-map for S_{x_1}

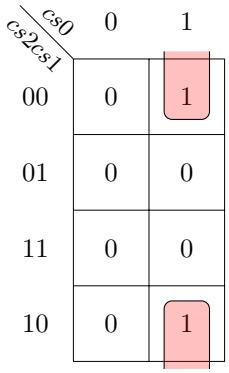


We can easily express S_{x_1} as sum of minterms.

$$S_{x_1} = c_{s_2} c_{s_1} + c_{s_2} c_{s_0}$$

$$S_{x_1} = c_{s_2}(c_{s_1} + c_{s_0})$$

3.2.3 K-map for S_{x_0}

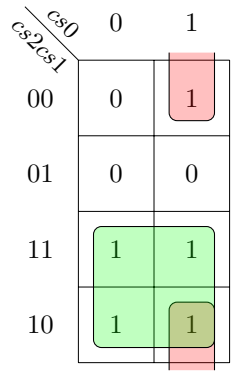


So, there are two minterms.

$$S_{x_0} = \overline{c_{s_1}} c_{s_0}$$

3.2.4 K-map for S_{y_0}

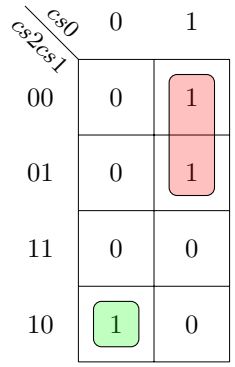
S_{y_0} is the selection bit for the multiplexer that selects B and 0 as input of a 2*1 MUX .



$$S_{y_0} == c_{s_2} + \overline{c_{s_1}} c_{s_0}$$

3.2.5 K-map for C_{in}

It is the input carry bit of the adder used inside arithmetic unit.



$$C_{in} = \overline{c_{s_2}} c_{s_0} + c_{s_2} \overline{c_{s_1}} . \overline{c_{s_0}}$$

$$C_{in} = \overline{c_{s_2}} c_{s_0} + c_{s_2} . \overline{c_{s_1} + c_{s_0}}$$