6 Arithmetic Circuits

In computers, arithmetic computations such as binary addition and subtraction are done in arithmetic logic unit (ALU) that consists of logic gates and flip-flops. Logic gates perform the arithmetic operation while the flip-flops (i.e. register and accumulator) are used as temporary memory storage (something like a scratch pad that we use to perform mathematical computation). We will look at adder and subtractor circuits in this chapter.

6.1 Half adder

Consider adding two bits, A_0 and B_0 to give sum Σ_0 and carry-out, C_1. Table 6.1 shows the possible combinations that can take place.

Table 6.1: Half adder combination

A_0	B_0	C_1	Σ
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

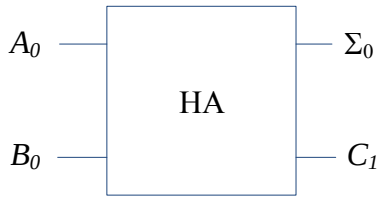

Figure 6.1: Half adder (HA) symbol.

Using K-maps as shown in Figure 6.2, we can obtain the logic expressions for Σ_0 and C_1. It can be seen that for Σ_0, it is not possible to simplify the expression as no looping is possible and the expression is

$$\Sigma_0 = A_0\overline{B_0} + \overline{A_0}B_0.$$

Since this is XOR expression (see Section 3.5), it can also be expressed as

$$\Sigma_0 = A_0 \oplus B_0.$$

Similarly, the expression for C_1 is

$$C_1 = A_0 B_0 .$$

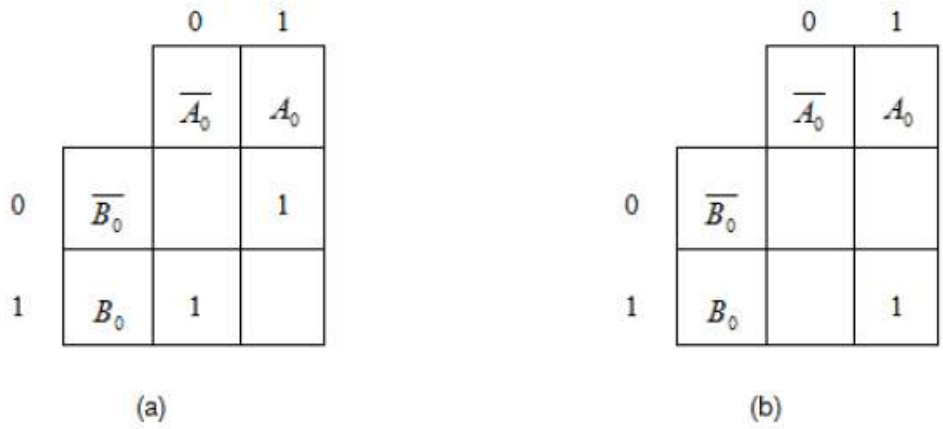

Figure 6.2: Half adder K-maps for (a) Σ_0 (b) C_1.

The half-adder logic circuit is shown in Figure 6.3.

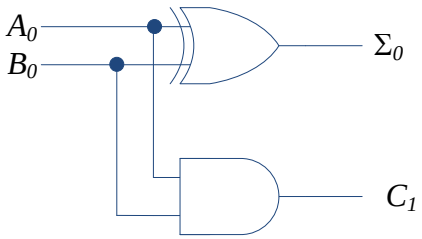

Figure 6.3: Half adder logic circuit.

6.2 Full adder

Very often when adding two bits, A_0 and B_0 to give sum Σ_0 and carry-out C_1, there can be another input, carry-in C_0 resulting from addition of previous bits. The possible combinations for a full adder are shown in Table 6.2 where it can be seen that the three binary inputs, A_0, B_0 and C_0 add to give the two binary outputs, Σ_0 and C_1. Full adder symbol is shown in Figure 6.4.

Table 6.2: Full adder combinations

A_0	B_0	C_0	C_1	Σ
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

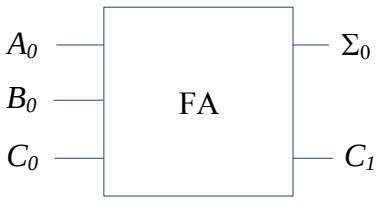

Figure 6.4: Full adder (FA) symbol.

K-maps for the two full adder outputs are shown in Figure 6.5. For Σ_0, no looping is possible and the expression is

$$\Sigma_0 = \overline{A_0}\,\overline{B_0}C_0 + \overline{A_0}B_0\overline{C_0} + A_0B_0C_0 + A_0\overline{B_0}\,\overline{C_0}$$

$$\Sigma_0 = C_0(\overline{A_0}\,\overline{B_0} + A_0B_0) + \overline{C_0}(\overline{A_0}B_0 + A_0\overline{B_0})$$

which can also be expressed in simpler form using XOR and XNOR expressions as $(\overline{A_0}B_0 + A_0\overline{B_0}) = A_0 \oplus B_0$ and $(\overline{A_0}\,\overline{B_0} + A_0B_0) = \overline{A_0 \oplus B_0}$ to give

$$\Sigma_0 = C_o(\overline{A_0 \oplus B_0}) + \overline{C_0}(A_0 \oplus B_0)\,.$$

We can actually simplify this further by allowing $X = A_0 \oplus B_0$:

$$\Sigma_0 = C_0\overline{X} + \overline{C_0}X\,.$$

Further simplification can be made using an XOR expression to give

$$\Sigma_0 = X \oplus C_0$$

$$\Sigma_0 = A_0 \oplus B_0 \oplus C_0$$

For C_1, three pair loops are possible resulting in

$$C_1 = A_0B_0 + A_0C_0 + B_0C_o\,.$$

		00	01	11	10
		$\overline{A_0}\ \overline{B_0}$	$\overline{A_0}\ B_0$	$A_0\ B_0$	$A_0\ \overline{B_0}$
0	$\overline{C_0}$		1		1
1	C_0	1		1	

(a)

		00	01	11	10
		$\overline{A_0}\ \overline{B_0}$	$\overline{A_0}\ B_0$	$A_0\ B_0$	$A_0\ \overline{B_0}$
0	$\overline{C_0}$			1	
1	C_0		1	1	1

(b)

Figure 6.5: Full adder K-maps for (a) Σ_0 (b) C_1.

The full adder circuitry is shown in Figure 6.6.

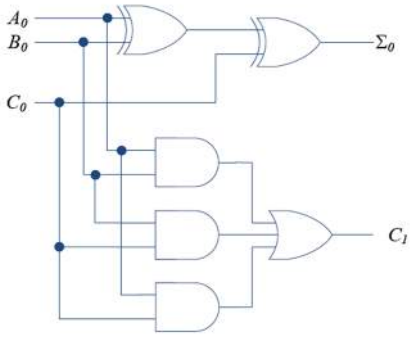

Figure 6.6: Full adder logic circuitry.

It should be obvious that a half-adder can be constructed using a full adder by setting C_0=0. This is illustrated in Figure 6.7.

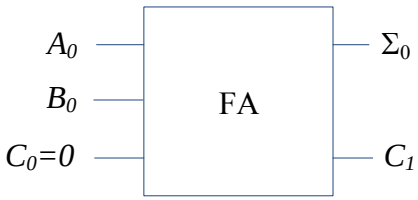

Figure 6.7: Half adder design using full adder.

6.3 Parallel adder

Usually, addition is done on a number of bits using a parallel adder that consists of several full adders as shown in Figure 6.8 for addition of two 3 bit numbers.

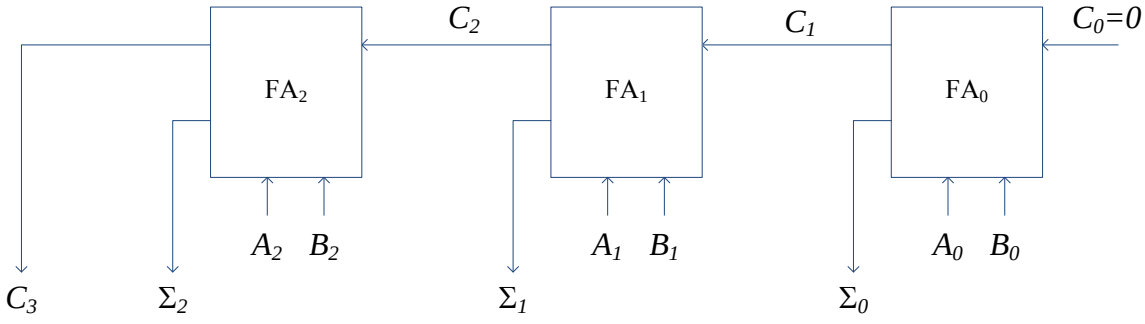

Figure 6.8: Parallel adder layout for addition of two 3 bit numbers.

As an example, consider adding A = 1 1 1 with B = 1 0 1 as depicted in Figure 6.9 to give sum = 1 0 0 and final carry of 1.

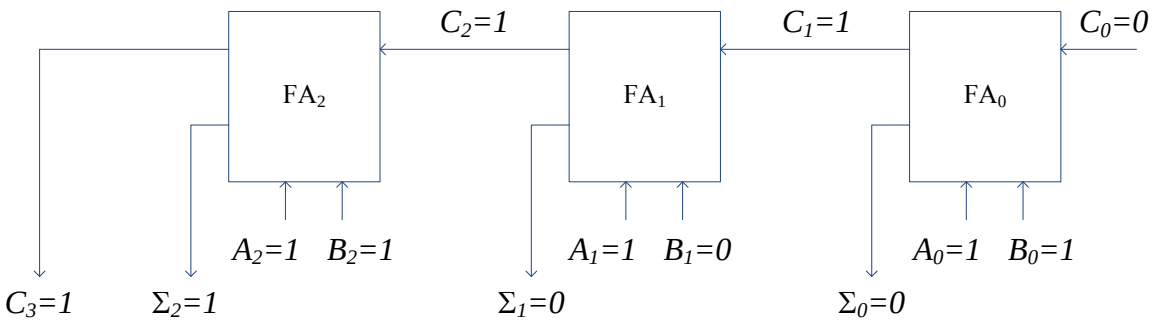

Figure 6.9: Parallel addition example of two 3 bit numbers.

6.4 Parallel addition using integrated circuits

Parallel adders in integrated circuits (IC) form are available such as the four bit TTL 74LS283 as shown in Figure 6.10 (with pin configurations). Such adders can be cascaded to add more bits. For example, two 74LS283 ICs can be used to add two 8 bit numbers as illustrated in Figure 6.11 (pin layout has been modified for ease of understanding, the actual layout is as shown in Figure 6.10). The two numbers: A_0, A_1, A_2, A_3, A_4, A_5, A_7 and B_0, B_1, B_2, B_3, B_4, B_5, B_7 are added together with carry input C_0 to give sum S_0, S_1, S_2, S_3, S_4, S_5, S_7 and carry out C_8. The carry out from the first IC, C_4 is passed as the carry input to the second IC.

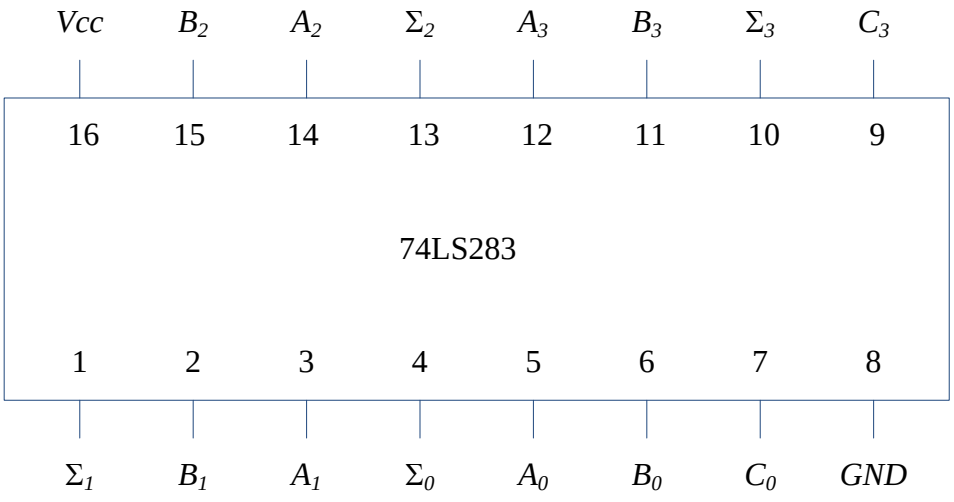

Figure 6.10: Four bit adder IC, 74LS283 showing pin configurations.

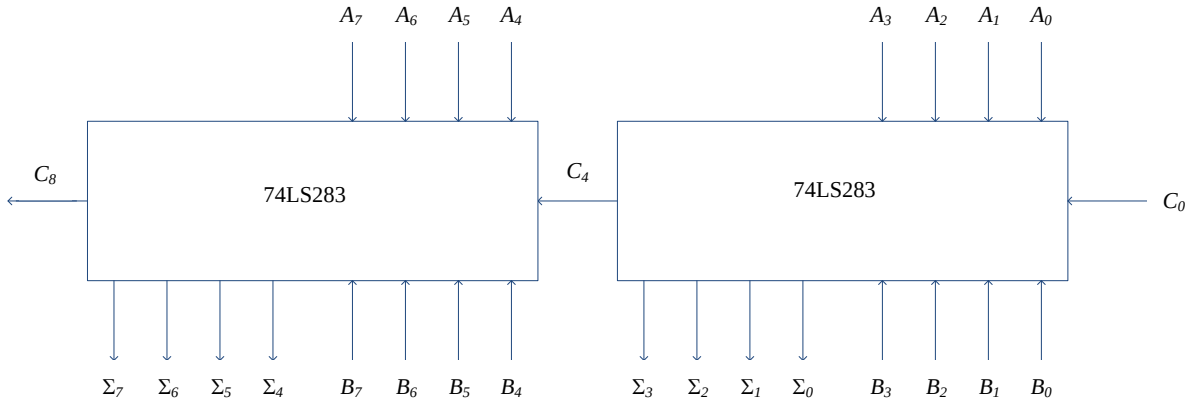

Figure 6.11: Cascading two 74LS283 to add 8 bit numbers.

6.5 Parallel subtraction

Consider a simple subtraction problem: 6 - 4 = 2. In binary, this will be 0110 – 0100 = 0010. Subtraction in binary can be performed through addition by converting the number to be subtracted (i.e. the subtrahend) to 2's complement form and adding to the minued[9].

9 In the example, 6 – 2 = 4, 6 is the minued and 2 is the subtrahend.

6.5.1 2's complement

A binary number can be converted to 2's complement simply by performing 1's complement (i.e. inverting) each bit and then adding 1 to the inverted bits. Any carry from this operation should be discarded. For example, 2's complement of 4 in binary is

4 in binary → 0100

1's complement of 4 → 1011

2's complement of 4 → 1100

Now, 6 - 4 can be represented in binary as shown in Figure 6.12. The carry is discarded to give the correct answer of 2.

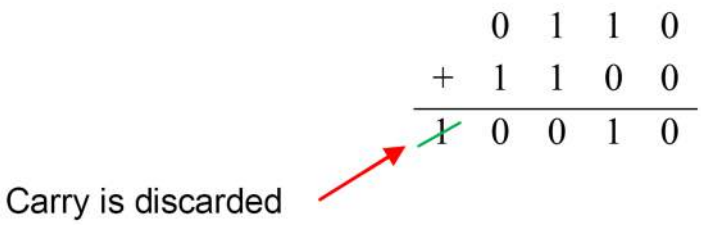

Figure 6.12: Subtracting two numbers using 2's complement method for subtrahend.

It should be obvious that an adder can also function as subtractor with additional gates. For example, the full adder shown in Figure 6.4 can be used to design a subtractor by inverting B_0 and setting C_0=1 (both these actions will result in 2's complement form for B_0) as shown in Figure 6.13. Similar to parallel adders, parallel subtractors can be designed using several full adders as shown in Figure 6.14.

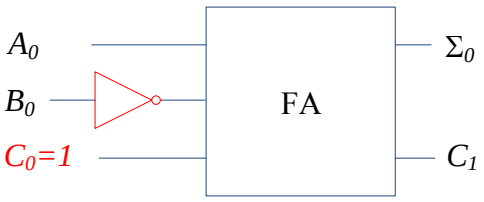

Figure 6.13: A full adder used as subtractor.

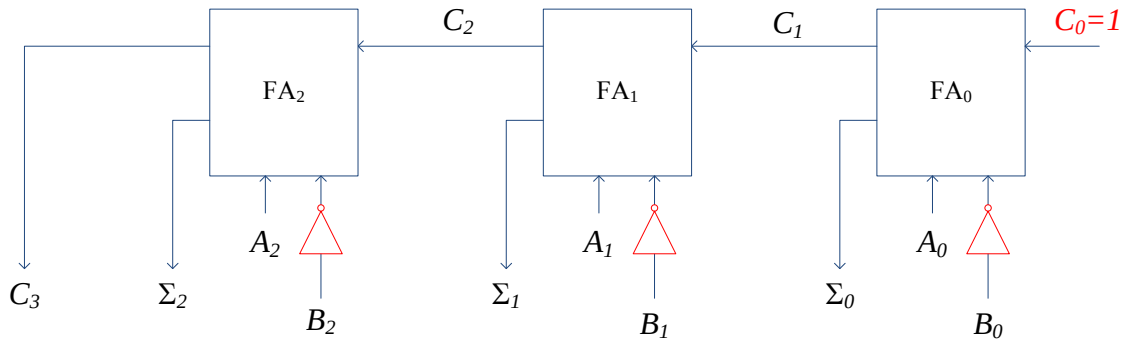

Figure 6.14: Designing a parallel subtractor using several full adders.

Using the example in Figure 6.12, 74LS283 can be modified to act as subtractor as shown in Figure 6.15. The minued is represented by A_0, A_1, A_2, A_3 and the inverters convert the subtrahend (B_0, B_1, B_2, B_3) to 1's complement and C_0 is set to 1 to convert this 1's complement number to 2's complement. The outputs (Σ_0, Σ_1, Σ_2, Σ_3) denote the correct answer as 4 and the carry out, $C_4 = 1$ is discarded.

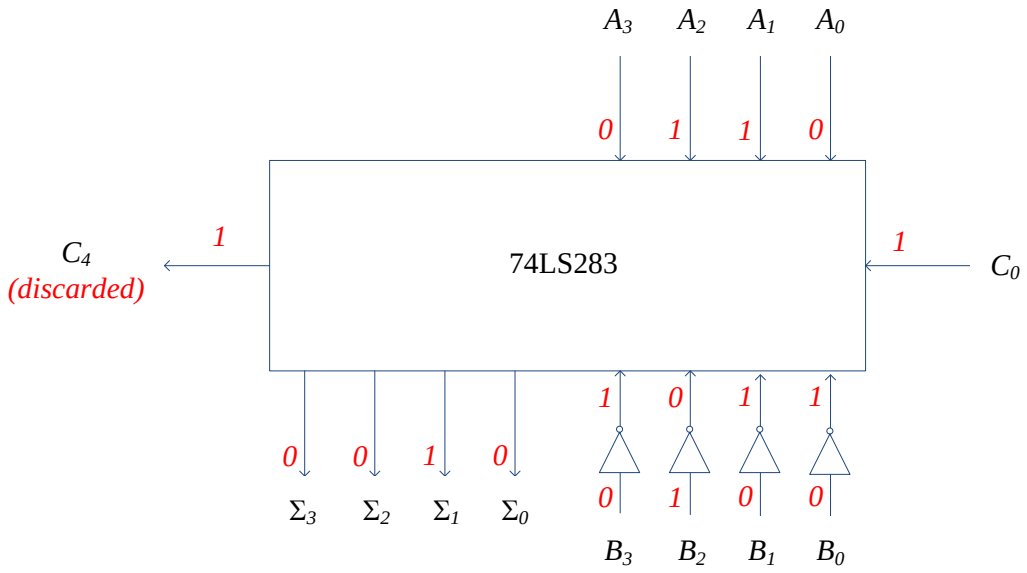

Figure 6.15: Using 74LS283 as subtractor.

6.5.2 Dual adder/subtractor

Replacing the inverters in Figure 6.15 with XOR gates will result in a dual mode adder/subtractor circuit. This is illustrated in Figure 6.16. When the control input is 1, the circuit acts as a subtractor and when the control input is 0, it acts as an adder. For example, when B_0=1 and control input=1 (during subtraction), the output of XOR is 0, i.e. the XOR gate acts as an inverter to give 1's complement and C_0=1 to give 2's complement. When B_0=1 and control input=0 (during addition), C_0=0 and the output of XOR is 1, i.e. the XOR gate acts just as a buffer without changing the logic value.

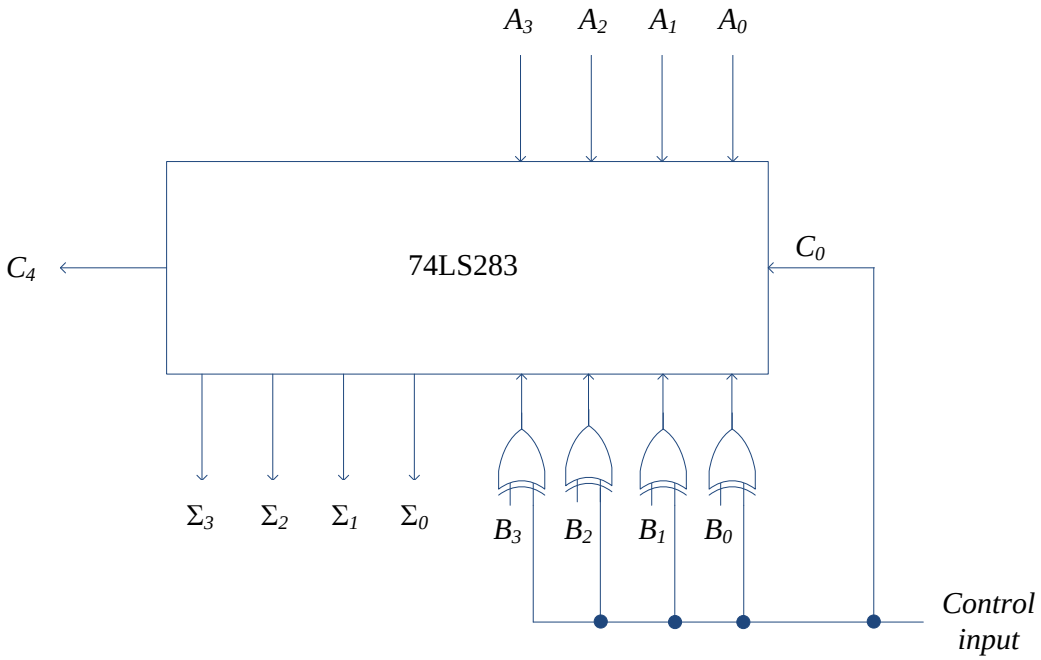

Figure 6.16: Using 74LS283 in dual mode: adder/subtractor.