Chapter 7 Binary Add

Lab Using only contacts and coils, add two integer numbers found in two integer numbers. Counts may be 8 bit, 16 bit or 32 bit in length.

Binary Addition

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 carry 1
1 + 0 + carry = 0 carry 1
1 + 1 + carry = 1 carry 1

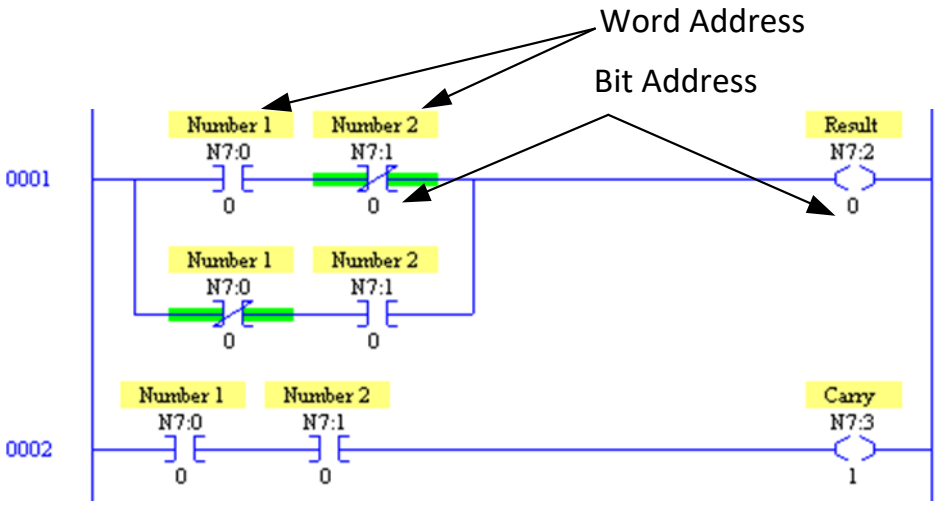
These are the rules for binary addition.

To see binary addition at work:

Carry				1	1	1	1				
Number 1	0	1	0	0	1	1	0	1	1	0	0
+ Number 2	0	1	0	1	1	0	1	1	0	1	0
Results	1	0	1	0	1	0	0	0	1	1	0

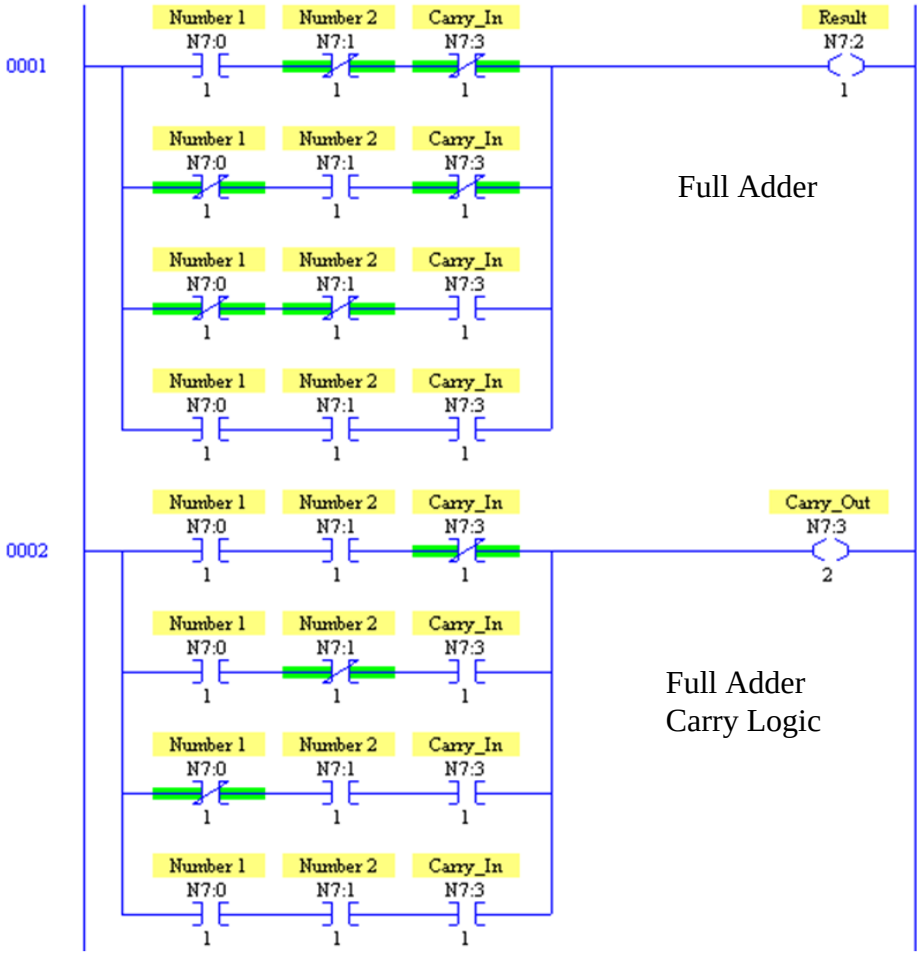
Binary addition may take place in ladder logic. Instructions are provided to carry out this function (ADD), but it is worthwhile to examine the process of binary addition using ladder logic. In Figs. 8-35 and 8-36, logic to add two numbers using only combinational logic is shown.

Since Bit 0 does not have a *carry_in*, half-adder logic may be employed but only for this bit. It can be seen that half-adder logic is simpler than full-add logic by comparing Fig. 8-35 (Half-Adder) to Fig. 8-36 (Full Adder).



Half Adder Logic

The number in location N7:0 is added to the number in N7:1. The result is stored in location N7:2. The carry is located in N7:3. The same locations are used for remaining bits of the word shown in Fig. 8-36. Full adder logic for each remaining bit from 1 to 15 is required. The logic must be duplicated for each bit. Carry_In is from the prior bit. The Carry_In for bit 1 is found in Carry_Out of bit 0.



It is worth noting that to actually build this logic requires a great deal of time unless the Copy-Paste function is employed. Once the logic is built, changing the bit numbers in the logic is all that is required for succeeding bits from 2 to 15.

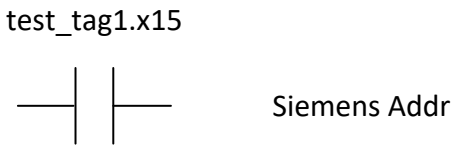
Summary of Addressing Individual Bits

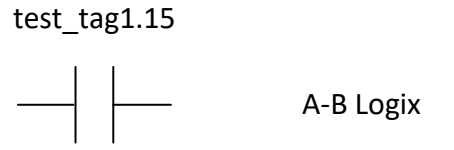
For a 16 bit integer, we have:

Bit	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Siemens M	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Siemens Addr	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Logix	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Data Slice Last Look

With the above table, we can assign contacts or coils referencing specific bits in the word test_tag1. These examples show addressing for the most significant or sign bit.





Binary Subtraction

Lab Using only contacts and coils, subtract one integer number from another integer.

Binary Subtraction

To perform binary subtraction, the easiest method is to find the 2's complement of the second number and then add the two numbers together. To find the 2's complement, invert all the bits (1's complement and add 1).

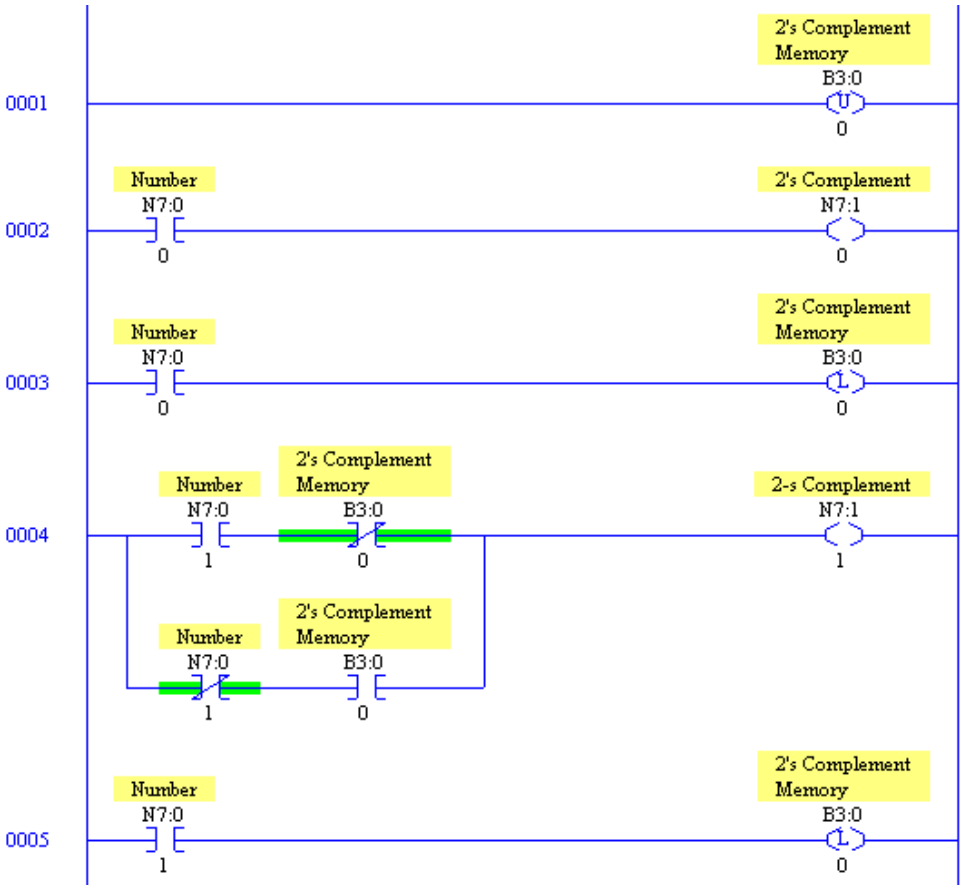
To find the 2's complement:

number	0	1	0	0	1	1	0	1	0	1	1
1's complement	1	0	1	1	0	0	1	0	1	0	0
+1											1
2's complement	1	0	1	1	0	0	1	0	1	0	1

Then add the 2's complement to the first number.

A second method of finding the 2's complement requires the use of a memory bit. The rule requires that bits from the original number be copied to the 2's complement number starting at the right-most bit. The rule applies until a "1" is encountered. The first "1" is copied but a memory bit is set after which the bits are "flipped". Try this rule. It works and may be employed using ladder logic and a Latch bit to quickly find the 2's complement of a number.

Again, logic must be added to complete the function using rungs similar to rungs 4 and 5 of this figure but using bits 2 through15.



Rung 1,2, 3
Bit 0 Logic

Rung 4, 5
Bit 1 Logic which must be repeated for Bits 2-15