Technical Note PLC-EOTKN102U-EN

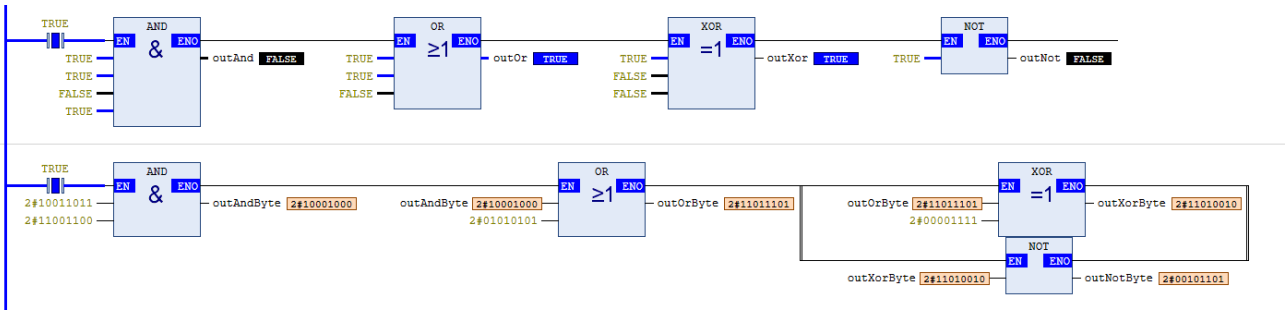
AC500 V3 PLC – Boolean operators

AND, OR, XOR & NOT Operators and Examples

The Boolean operators, also known as Bitstring operators are the most used operators. The operators AND, OR and XOR are used to combine two signals. NOT is inverting the signal. Mainly, Boolean variables are used with these operators. In addition, BYTE, WORD, DWORD or LWORD variables can be used with these operators.

- The AND operator returns TRUE, if all input signals are TRUE.
- The OR operator returns TRUE, if at least one of the input signals is TRUE.
- The XOR operator returns TRUE, if exactly one input signal is TRUE.
- The NOT operator inverts the input signal.

Examples:



The screenshot above shows a simple logic in ladder diagram. The top row shows logic with Boolean values. The bottom row a logic with byte values. By default, the AND, OR & XOR operators have two inputs. By right clicking on a block additional input can be added to the operators.

Starting on top left. The AND block has one FALSE input therefore the output is FALSE. The OR block has at least one (two TRUE inputs in this example) TRUE input. The output is TRUE.

The XOR has exactly one TRUE input and the others are all false. The output is TRUE. The NOT is inverting the TRUE input to a FALSE output.

The BYTE values can be used with these operators as well. Each single bit of the inputs is considered. Where both AND inputs signals have a 1, the output will be 1. The OR is returning a 1 in all positions where at least one input signal has a 1. XOR is doing the same where exactly one input has a 1. The NOT operator is inverting the signal so all 1 become 0 and all 0 become 1.

The same logic as shown in the screenshot above can be found below in structured text below.

```
//Boolean
outAnd FALSE := TRUE AND TRUE AND FALSE AND TRUE;
outOr TRUE  := TRUE OR TRUE AND FALSE;
outXor TRUE := TRUE XOR FALSE XOR FALSE;
outNot FALSE := NOT TRUE;

//Byte
outAndByte 2#10001000 := 2#10011011 AND 2#11001100;
outOrByte 2#11011101  := outAndByte 2#10001000 OR 2#01010101;
outXorByte 2#11010010 := outOrByte 2#11011101 XOR 2#00001111;
outNotByte 2#00101101 := NOT outXorByte 2#11010010 ;RETURN
```

```
//Boolean
outAnd    := TRUE AND TRUE AND FALSE AND TRUE;
outOr     := TRUE OR TRUE AND FALSE;
outXor    := TRUE XOR FALSE XOR FALSE;
outNot    := NOT TRUE;

//Byte
outAndByte        := 2#10011011 AND 2#11001100;
outOrByte         := outAndByte OR 2#01010101;
outXorByte        := outOrByte XOR 2#00001111;
outNotByte        := NOT outXorByte;
```