Siemens S7-1200

CPU 1212C AC/DC/Relay

Math Operations

- Understanding Data types
- Various Math operations
- Exercise Example

Code and Compile
Learning Made Easy
www.codeandcompile.com

Understanding various Data Types (1) – S7 1200

Data Types

The data type define the properties of the data, for example, the representation of the contents and the valid memory areas.

the user program, you can use predefined data type or also data types that you have defined.

Pre-defined Data Type

Data Type	Length (bits)	Range of values	Examples	Address Example
BOOL	1	2#0 or 2#1	2#0	I0.0, Q0.0, M0.0
BYTE	8	-128 ~ +127 or 0 ~ 255	2#00001111	MB0
WORD	16	0 to 65535	61680, W#61680	MW0
DWORD	32	0 to 4294967295	15793935, DW#15793935	MD0
SINT	8	-128 to 127	+44, SINT#+44	MB0
USINT	8	0 to 255	78, USINT#78	MB0
INT	16	-32768 to 32767	+3785, INT#+3785	MW0
UINT	16	0 to 65535	65295, UINT#65295	MW0
DINT	32	-2147483648 to +2147483647	125790, DINT#125790,	MD0
UDINT	32	0 to 4294967295	4042322160, UDINT#4042322160	MD0

Understanding various Data Types (2) – S7 1200

Pre-defined Data Type

Operands of the data type **REAL have a length of 32 bits** and are used to represent floating-point numbers**. An operand of the REAL data type consists of the following three components:**

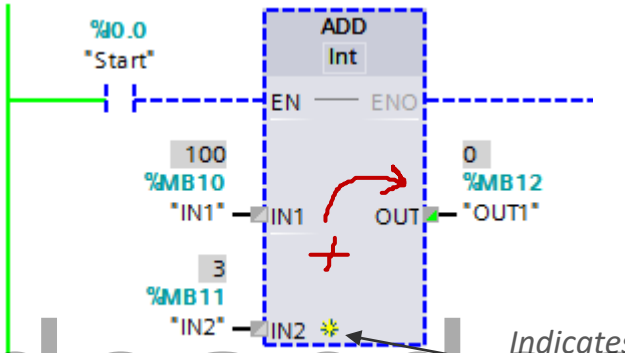
- **Sign**: The sign is determined by the signal state of bit 31. The bit 31 assume the value **"0" (positive)** or **"1" (negative).**
- **8-bit exponents to basis 2**: The exponent is increased by a constant (base, +127), so that it has a value range of **0 to 255**.
- **23-bit mantissa**: Only the fraction part of the mantissa is shown. The integer part of the mantissa is always 1.

Data Type	Length (bits)	Format	Range of values	Examples	Address Example
REAL	32	Floating-point numbers according to IEEE754	-3.402823e+38 to -1.175495e-38	1.0e-5	MD0
REAL	32	Floating-point numbers	+1.175495e-38 to +3.402823e+38	1.0	MD0

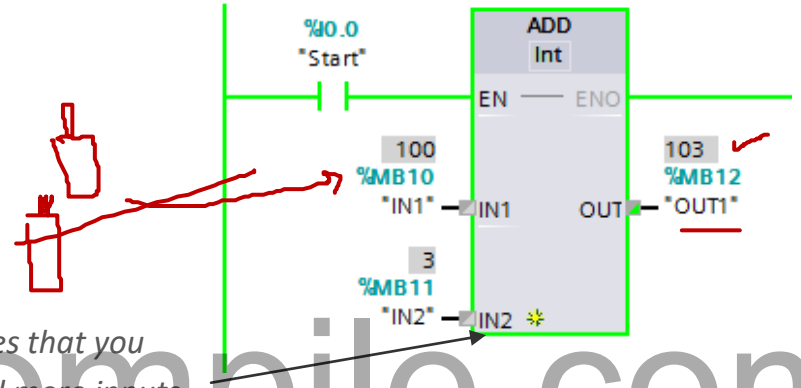
Math Operations in LAD

ADDITION – *for addition of two or more inputs*

Status of **OUT1** before addition



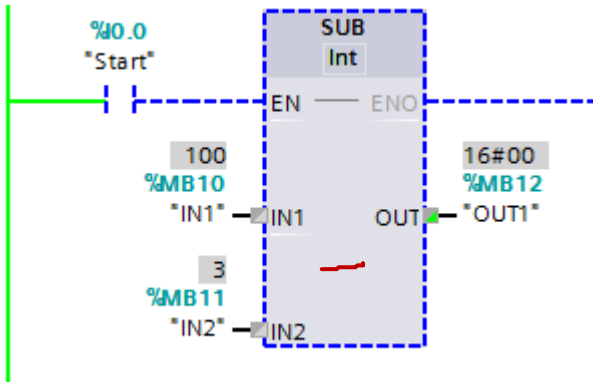
Status of **OUT1** after addition



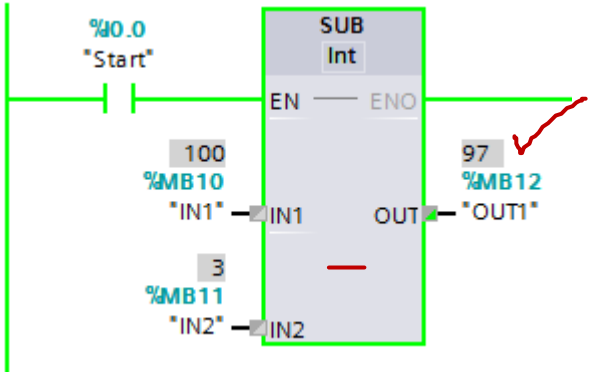
Indicates that you can add more inputs

SUBTRACT *for subtraction of two inputs*

Status of **OUT1** before subtraction



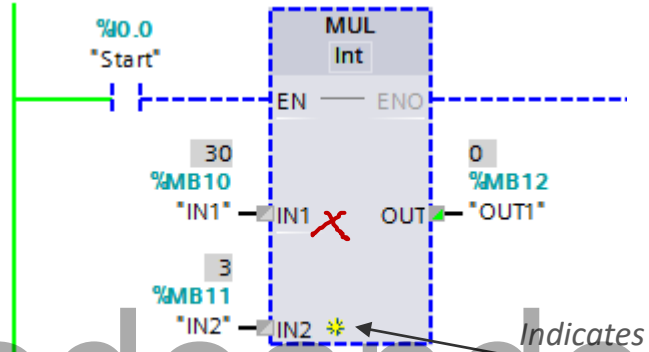
Status of **OUT1** after subtraction



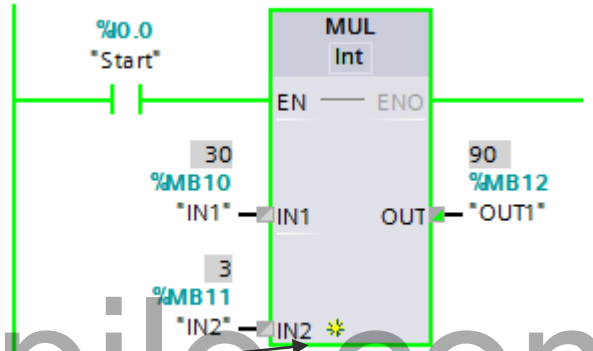
Math Operations in LAD

MULTIPLICATION- *for multiplication of two or more inputs*

Status of **OUT1** before multiplication



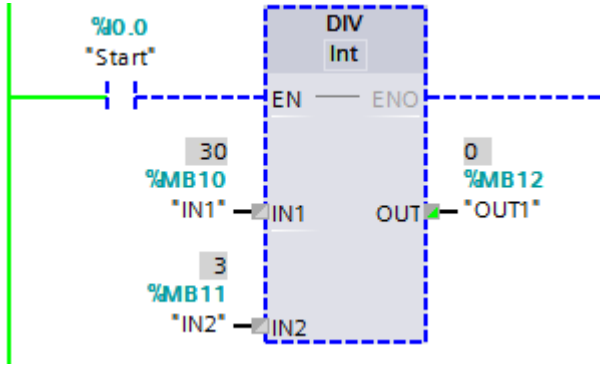
Status of **OUT1** after multiplication



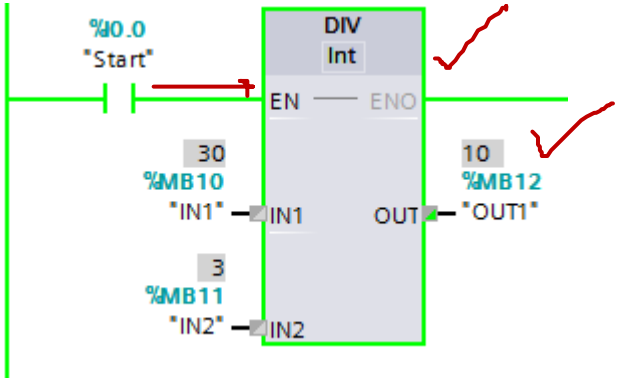
Indicates that you can add more inputs

DIVISION *for division of two inputs*

Status of **OUT1** before Division



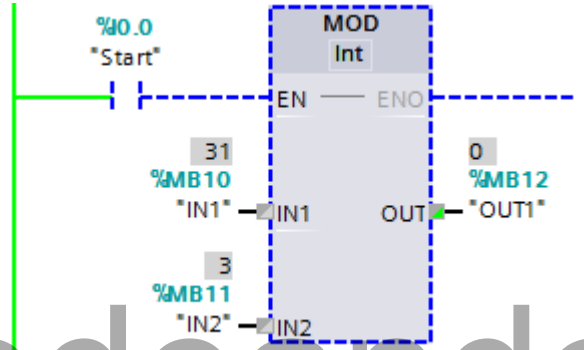
Status of **OUT1** after Division



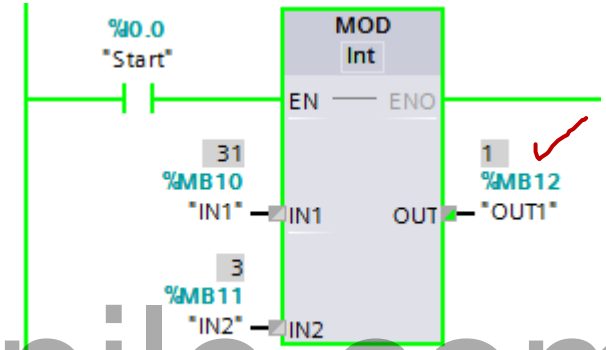
Math Operations in LAD

MOD – *return remainder of division*

Status of **OUT1** before MOD

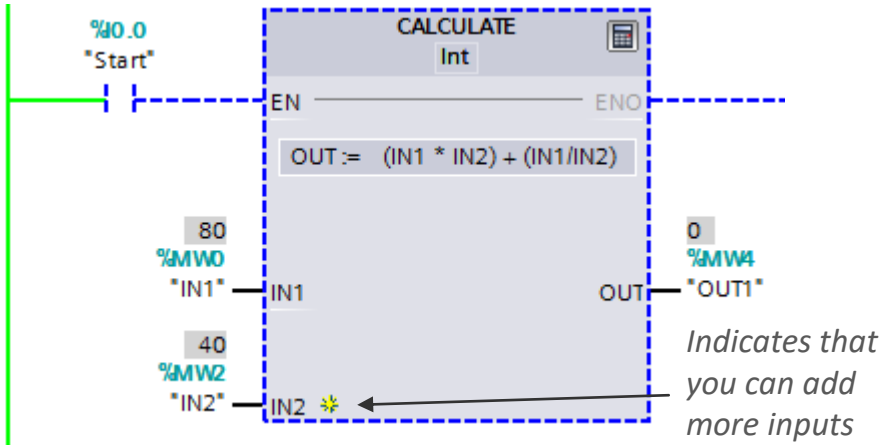


Status of **OUT1** after MOD

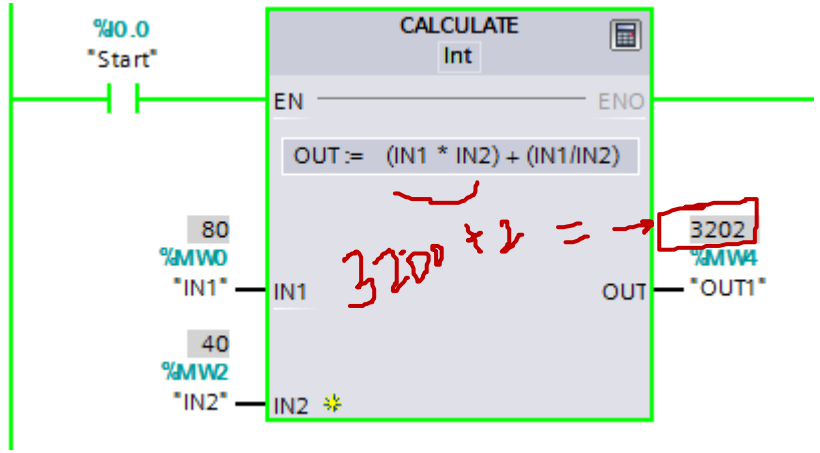


CALCULATE – *to calculate the pre-defined equation*

Status of **OUT1** before calculation



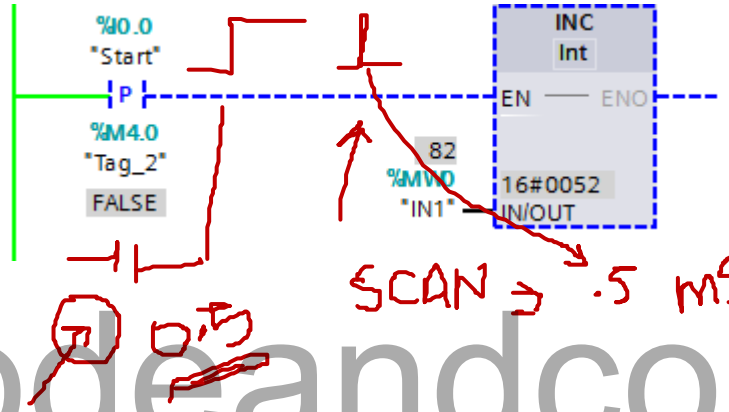
Status of **OUT1** after calculation



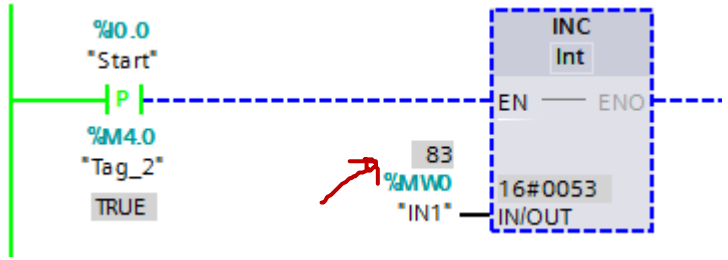
Math Operations in LAD

INC – *used to increment data by +1*

Status of **IN1** before increment

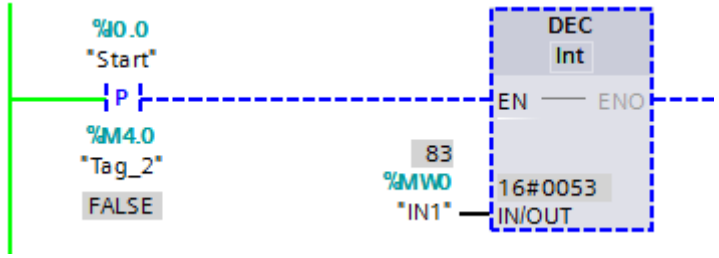


Status of **IN1** after increment

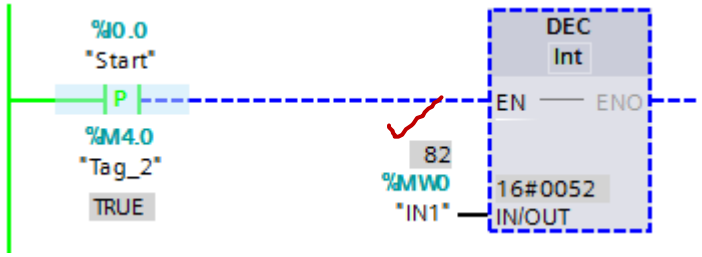


DEC - *used to decrement data by +1*

Status of **IN1** before decrement



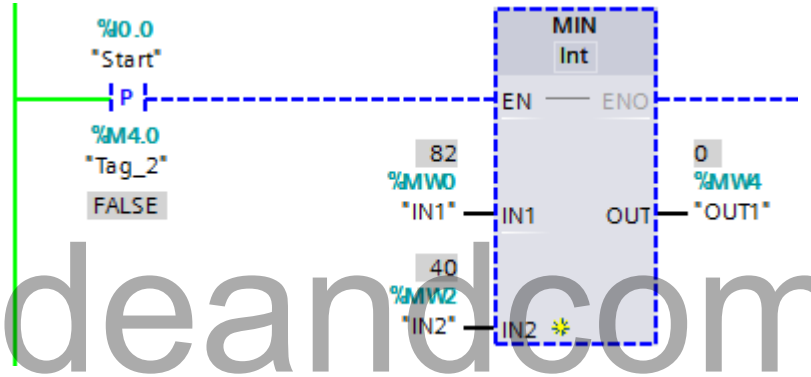
Status of **IN1** after decrement



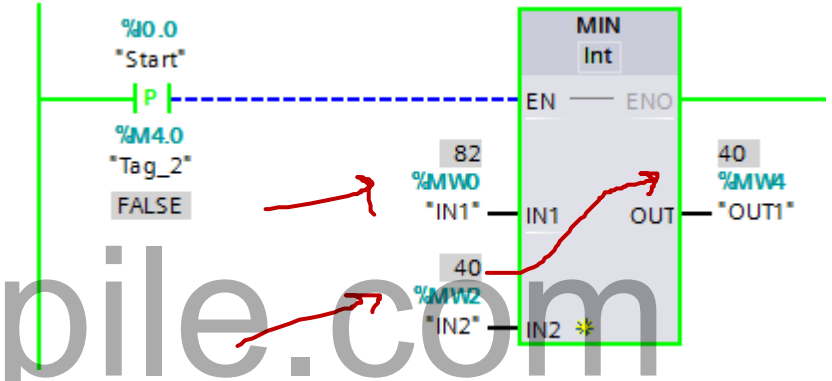
Math Operations in LAD

GET MINIMUM – compares the value at the input and write the lowest value to the output

Status of **OUT1** before execution

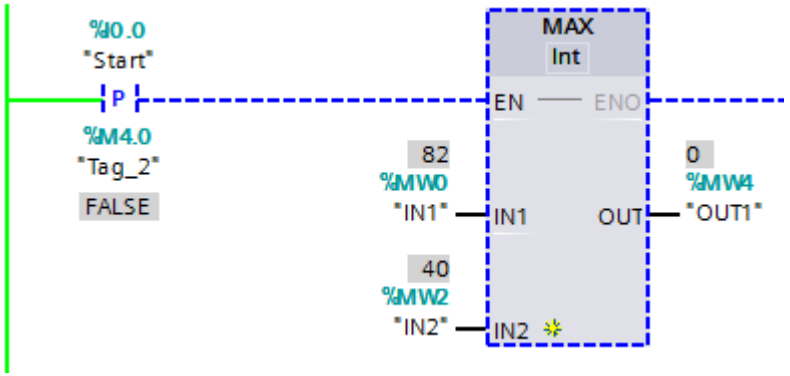


Status of **IN1** after execution

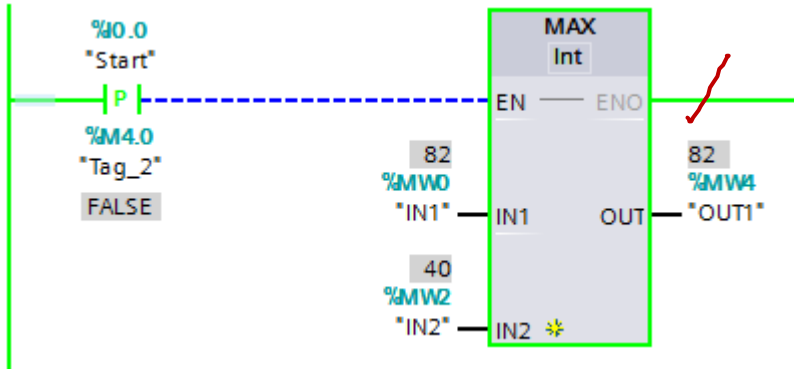


GET MAXIMUM - compares the value at the input and write the highest value to the output

Status of **OUT1** before execution



Status of **IN1** after execution



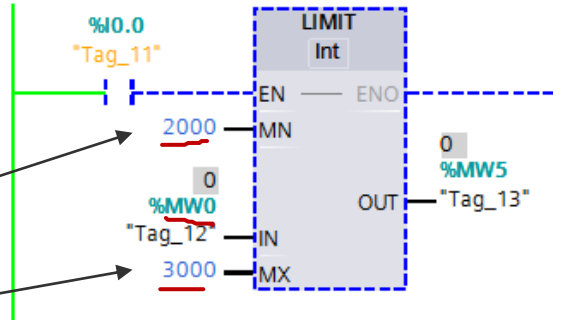
Math Operations in LAD

SET LIMIT – limit the value at the input to the values at MN and MX input.

Status of **OUT** before execution

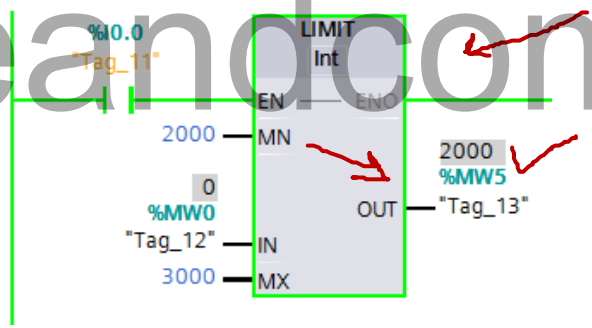
Min limit

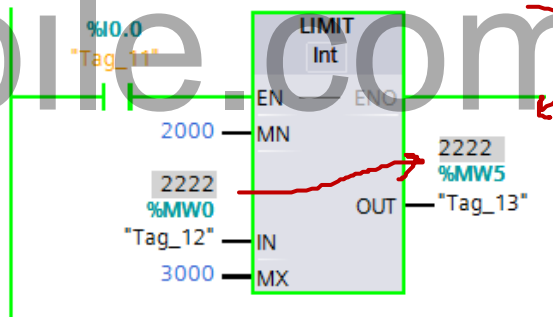
Max limit



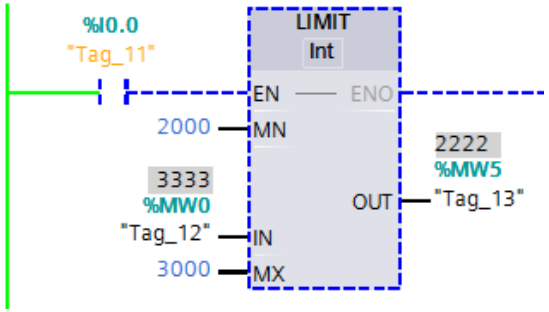
Output

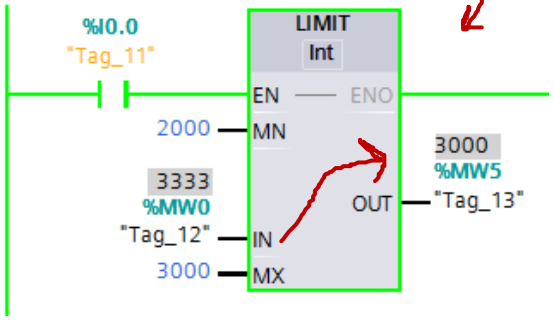
Status of **OUT** when the input is **<** minimum limit or **between min** and **max**





Status of **OUT** when the input is **>** maximum limit





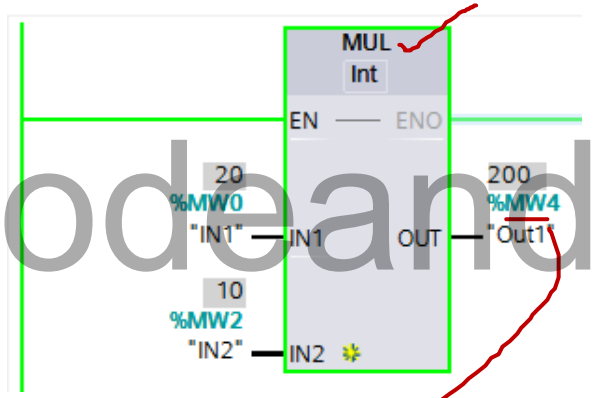
Exercise Example:

Make a **ladder logic** for the following equations: (Use INT as data type)

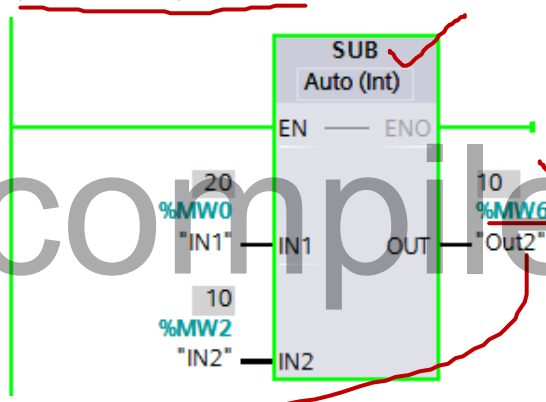
OUT5 = (IN1*IN2) + (IN1-IN2) – (IN1/IN2) – *without using calculate instruction*

Take the following constant and find result: **IN1 = 20; IN2 = 10**

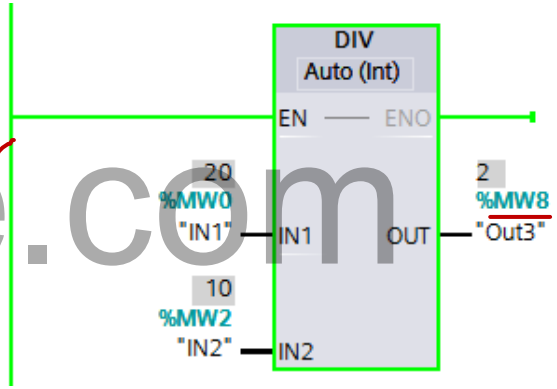
(IN1*IN2) = OUT1



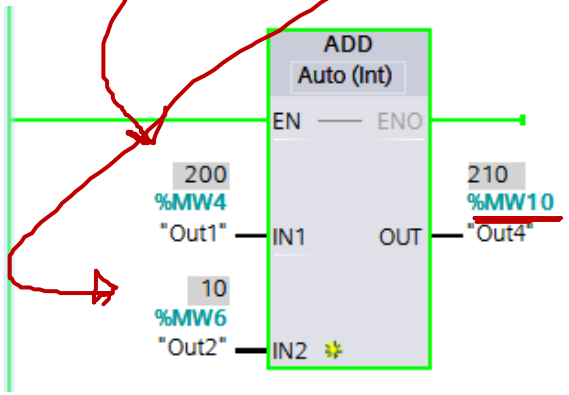
(IN1-IN2) = OUT2



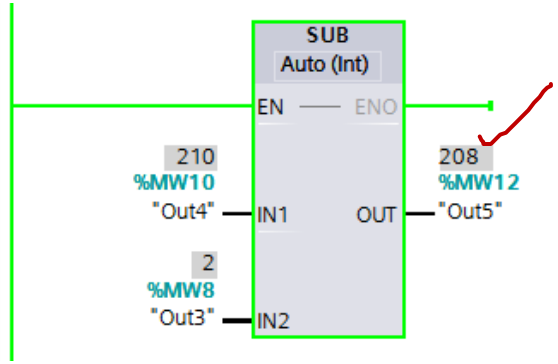
(IN1/IN2) = OUT3



(IN1*IN2) + (IN1-IN2) = OUT1 +OUT2 = OUT4



(IN1*IN2) + (IN1-IN2) – (IN1/IN2) = OUT4- OUT3 = OUT5



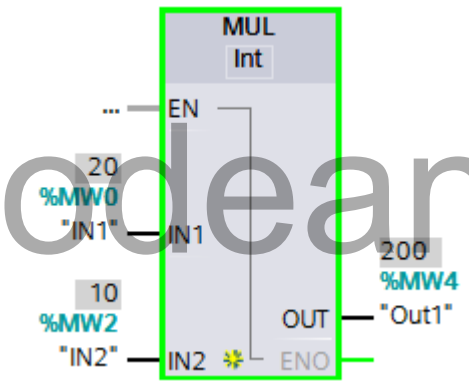
Exercise Example:

Make a **FBD logic** for the following equations: (Use INT as data type)

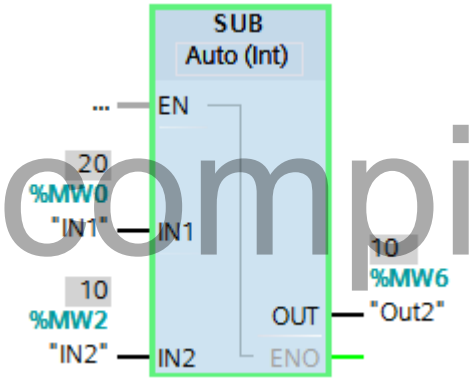
OUT5 = (IN1*IN2) + (IN1-IN2) – (IN1/IN2) – ***without using calculate instruction***

Take the following constant and find result: **IN1 = 20; IN2 = 10**

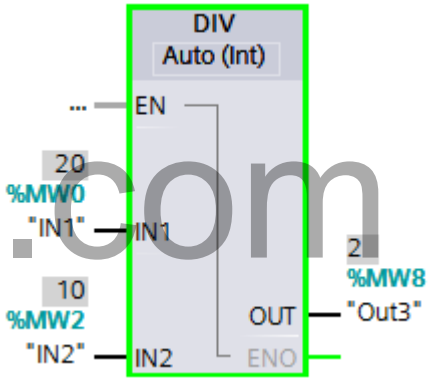
(IN1*IN2) = OUT1



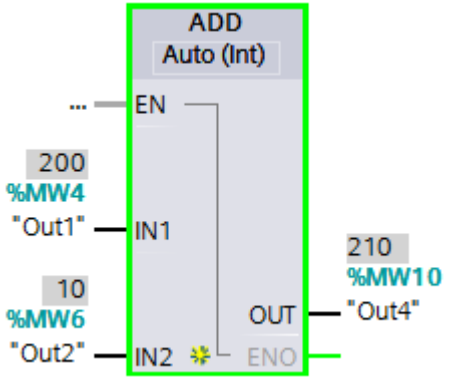
(IN1-IN2) = OUT2



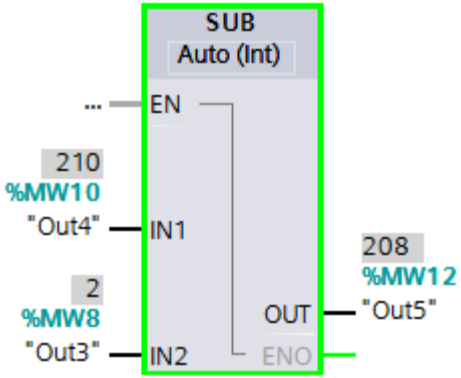
(IN1/IN2) = OUT3



(IN1*IN2) + (IN1-IN2) = OUT1 +OUT2 = OUT4



(IN1*IN2) + (IN1-IN2) – (IN1/IN2) = OUT4- OUT3 = OUT5



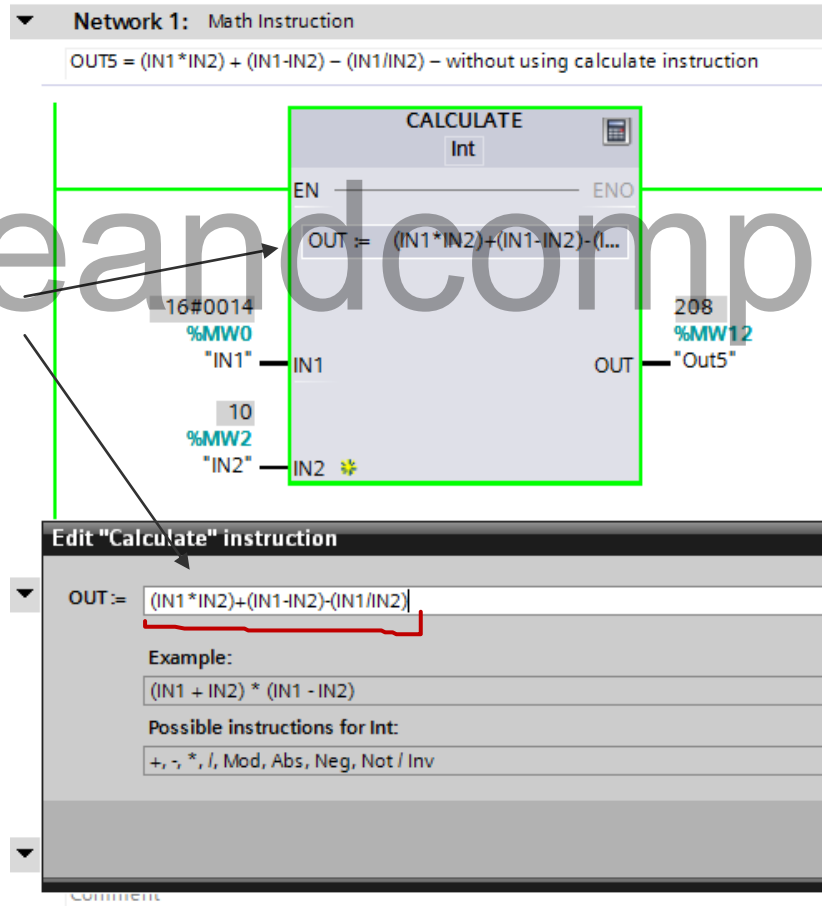
Exercise Example:

Make a **ladder logic** for the following equations: (Use INT as data type)

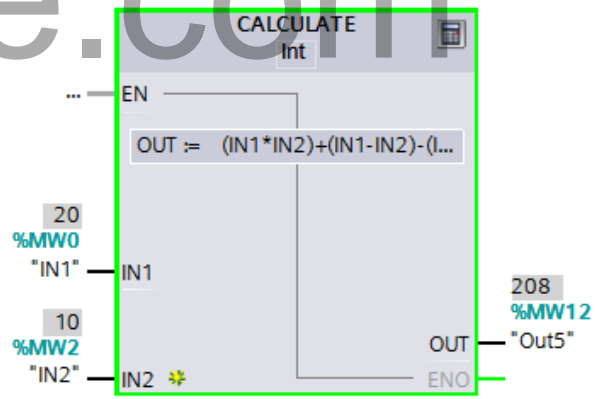
OUT5 = (IN1*IN2) + (IN1-IN2) – (IN1/IN2) –***using calculate instruction***

Take the following constant and find result: **IN1 = 20; IN2 = 10**

Pre-defined Equation



Equivalent FBD Logic



Several other maths instruction you can use!

- SQR = Square
- SQRT = Square root
- LN = Natural Logarithm
- EXP= Exponential Value
- SIN = Sine value
- COS = Cosine value
- TAN = Tangent value
- ASIN = Arcsine value
- ACOS = Arccosine value
- ATAN = Arctangent value
- FRAC = Return Fraction
- EXPT = Exponentiate

Thank you

Get copy of this presentation in the course!

Code and Compile
Learning Made Easy
www.codeandcompile.com