Programming Languages

Structured Text (ST)

Elmo
Motion Control

Programming Languages Structured Text (ST)

- Comments
  - Begin with “(*”
  - End with “*)”
  - Anywhere in the program
  - Several lines
  - Cannot be nested

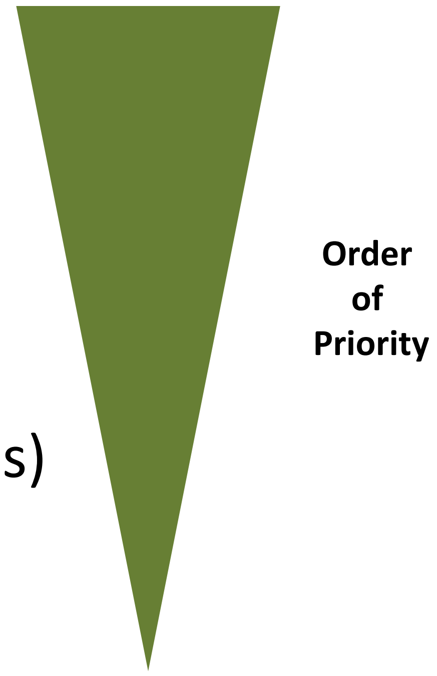
```
(* My comment *)
a := d + e;

(* A comment can also
   be on several lines  *)
b := d * e;

c := d - e;   (* My comment *)
```

Programming Languages Structured Text (ST)

- Expressions
  - Each statement describes an action and may include evaluation of complex expressions.
  - An expression is evaluated:
    - From the left to the right
    - According to the default priority order of operators
    - The default priority can be changed using parenthesis
  - Arguments of an expression can be:
    - Declared variables
    - Constant expressions
    - Function calls

Programming Languages Structured Text (ST)

- Operators
  - - (…)   NOT (…)
  - ** (power)
  - * /
  - + -
  - < > <= >= <> = (comparisons)
  - AND (you can use “&”)
  - OR
  - XOR



Programming Languages
Structured Text (ST)

- IF / THEN / ELSE / ELSIF / END_IF
  - Conditional execution of statements.
  - One or several ELSIF are allowed.

```
IF a = b THEN
      c := 0;
ELSIF a < b THEN
      c := 1;
ELSE
      c := -1;
END_IF;
```

Programming Languages Structured Text (ST)

- CASE / OF / ELSE / END_CASE
  - Switch between enumerated statements, according to the result of an expression.
  - The selector can be any integer or a STRING.

```
CASE iChoice OF
   0:
    MyString := 'Nothing';
   1 .. 6,9:
      MyString := 'First case';
   7,10:
      MyString := 'Second case';
   ELSE
      MyString := 'Other case';
END_CASE;
```

Elmo Motion Control

Programming Languages Structured Text (ST)

- WHILE / DO / END_WHILE
  - Condition is evaluated before the statements.
  - Warnings:
    - Loop instructions may lead to infinite loops that block the target cycle.
    - Never test the state of an input using this condition, because the input will not refresh before the next cycle.

```
iCount := 0;
WHILE iCount < 100 DO
  iCount := iCount +1;
  MyVar := MyVar + 1;
END_WHILE;
```

Elmo Motion Control

Programming Languages Structured Text (ST)

- REPEAT / UNTIL / END_REPEAT
  - Repeat a list of statements.
  - Condition is evaluated after the statements.
  - Warning:
    - Loop instructions may lead to infinite loops that block the target cycle.
    - Never test the state of an input using this condition, because the input will not refresh before the next cycle.

```
iCount := 0;
REPEAT
  MyVar := MyVar + 1;
  iCount := iCount + 1;
UNTIL iCount < 100 END_REPEAT;
```

Elmo Motion Control

Programming Languages
Structured Text (ST)

- FOR / TO / BY / END_FOR
    - Iteration of statement execution.
    - The “BY” statement can be omitted, the default value is 1.
    - Warning:
        - Loop instructions may lead to infinite loops that block the target cycle.
        - Never test the state of an input using this condition, because the input will not refresh before the next cycle.

```
FOR iCount := 0 TO 100 BY 2 DO
      MyVar := MyVar + 1;
END_FOR;
```

Elmo Motion Control

Programming Languages Structured Text (ST)

- Function
  - To call a function in ST:
    - Enter its name, followed by the input parameters written between parenthesis and separated by comas.
    - The function call may be inserted into any complex expression.
    - A function call can be used as an input parameter of another function.

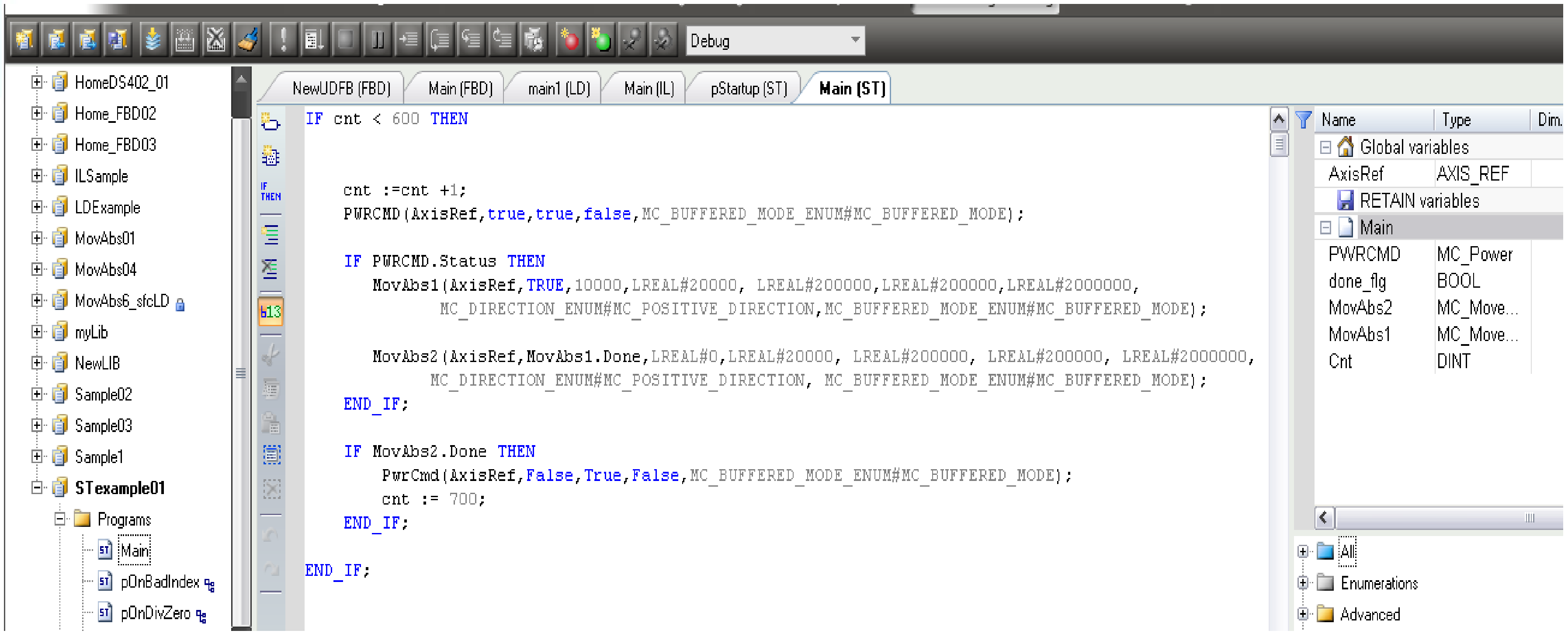
```
a := MAX(b, c);

d := MAX(5, RAND(20));
```

Programming Languages Structured Text (ST)

- Function Block
  - To call a function block in ST:
    - Declare an instance of the function block.
    - Use the instance name as instructed, followed by the input parameters written between parenthesis and separated by commas.
    - The outputs of the function block are stored in the instance.

```
MyCTU(CU, RESET, PV);    (* FBlock call *)

Q := MyCTU.Q;            (* Get output *)
CV := MyCTU.CV;          (* Get output *)
```

Programming Languages Structured Text (ST)



```
IF cnt < 600 THEN


    cnt :=cnt +1;
    PWRCMD(AxisRef,true,true,false,MC_BUFFERED_MODE_ENUM#MC_BUFFERED_MODE);

    IF PWRCMD.Status THEN
       MovAbs1(AxisRef,TRUE,10000,LREAL#20000, LREAL#200000,LREAL#200000,LREAL#2000000,
              MC_DIRECTION_ENUM#MC_POSITIVE_DIRECTION,MC_BUFFERED_MODE_ENUM#MC_BUFFERED_MODE);

       MovAbs2(AxisRef,MovAbs1.Done,LREAL#0,LREAL#20000, LREAL#200000, LREAL#200000, LREAL#2000000,
             MC_DIRECTION_ENUM#MC_POSITIVE_DIRECTION, MC_BUFFERED_MODE_ENUM#MC_BUFFERED_MODE);
    END_IF;

    IF MovAbs2.Done THEN
        PwrCmd(AxisRef,False,True,False,MC_BUFFERED_MODE_ENUM#MC_BUFFERED_MODE);
        cnt := 700;
    END_IF;

END_IF;
```

Name	Type	Dim.
Global variables		
AxisRef	AXIS_REF	
RETAIN variables		
Main		
PWRCMD	MC_Power	
done_flg	BOOL	
MovAbs2	MC_Move...	
MovAbs1	MC_Move...	
Cnt	DINT	

Inspiring Motion

Since ***1988***

Thank You!

www.elmomc.com

Elmo
Motion Control