Structured Text
a high level language

Eelco van der Wal

Managing Director PLCopen

www.plcopen.org

Operators in ST

Symbol	**Operation**
(expression)	Parenthesization
identifier(argument list)	Function evaluation
Examples:	
LN(A), MAX(X,Y), etc.	
**	Exponentiation
-	Negation
NOT	Complement

Operators in ST

*	Multiply
/	Divide
MOD	Modulo
+	Add
-	Subtract
< , > , <= , >=	Comparison
=	Equality
<>	Inequality

Operators in ST

&	**Boolean AND**
AND	**Boolean AND**
XOR	**Boolean Exclusive OR**
OR	**Boolean OR**

assignment

```
A := B;
```

assignment

A := B;

Datatype to Datatype

INT to INT

or

Analog_Channel_Configuration to Analog_Channel_Configuration

assignment

A := B;

INT to INT

or

Analog_Channel_Configuration to Analog_Channel_Configuration

CV := CV+1;

C := SIN(X);

Controlled repetition via iteration or selection

- **IF .. THEN .. ELSE**
- **CASE**
- **FOR**
- **WHILE …**
- **REPEAT UNTIL**

IF .. THEN .. ELSE

IF **condition= true**

THEN **action1**

ELSE **action2** **(condition not true)**

IF raining THEN Stay_In ELSE Go_Out

IF .. THEN .. ELSE

```
D := B*B - 4*A*C ;
  IF D < 0.0       THEN       NROOTS := 0 ;
  ELSIF            D = 0.0 THEN
    NROOTS := 1 ;
    X1 := - B/(2.0*A) ;
  ELSE
    NROOTS := 2 ;
    X1 := (- B + SQRT(D))/(2.0*A) ;
    X2 := (- B - SQRT(D))/(2.0*A) ;
  END_IF ;
```

CASE : repetitive IF

```
CASE selector OF
        selector_value :    action

   ELSE  …..
END_CASE;
```

CASE : repetitive IF

```
TW := BCD_TO_INT(THUMBWHEEL);
      TW_ERROR := 0;
CASE TW OF
      1,5:    DISPLAY := OVEN_TEMP;
      2:      DISPLAY := MOTOR_SPEED;
      3:      DISPLAY := GROSS - TARE;
      4,6..10: DISPLAY := STATUS(TW - 4);
ELSE  DISPLAY := 0 ;
      TW_ERROR := 1;
END_CASE;
  QW100 := INT_TO_BCD(DISPLAY);
```

FOR

```
SUM := 0 ;

FOR   begin_value        TO     end_value    DO

END_FOR ;
```

FOR

```
SUM := 0 ;

FOR   I := 1   TO     3       DO

    FOR J := 1 TO 2 DO
        SUM := SUM + J ;
    END_FOR ;
  SUM := SUM + I ;
END_FOR ;
```

FOR

```
J := 101 ;

FOR I := 1 TO 100  BY    2       DO
        IF                WORDS[I] = 'KEY'   THEN
        J := I ;
        EXIT ;
        END_IF ;
END_FOR ;
```

WHILE

```
J := 1;


WHILE      J <= 100 & WORDS[J] <> 'KEY' DO
    J := J+2 ;
END_WHILE ;
```

REPEAT … UNTIL

```
J := -1 ;

  REPEAT
        J := J+2 ;
  UNTIL J = 101 OR WORDS[J] = 'KEY'
  END_REPEAT ;
```

EXIT and RETURN

- The EXIT statement shall be used to terminate iterations before the termination condition is satisfied.

```
SUM := 0 ;
      FOR I := 1 TO 3 DO
                FOR J := 1 TO 2 DO
                IF FLAG THEN EXIT ; END_IF
               SUM := SUM + J ;
               END_FOR ;
      SUM := SUM + I ;
      END_FOR ;
```