**FUNCTION\_BLOCK fbConvertUnixTime**

**VAR\_INPUT**

dtInput: DT;

siTimezoneOffset: SINT; // Timezone offset in hours + or -

**END\_VAR**

**VAR\_OUTPUT**

usiSeconds: USINT;

usiMinutes: USINT;

usiHours: USINT;

iYear: INT;

usiMonth: USINT;

usiDay: USINT;

xIsDST: BOOL; // dst active?

usiDOW: USINT; // day of week 0-6 with 0= sunday

**END\_VAR**

**VAR**

iTimeZone: INT; // timezone is a signed int, given in seconds

iDstOffset: INT := 3600; // daylight saving time is a signed int, given in seconds

usiPreviousSunday: USINT;

a: DINT;

b: DINT;

c: DINT;

d: DINT;

e: DINT;

f: DINT;

t: DINT;

h: DINT;

j: DINT;

k: DINT;

m: DINT;

y: DINT;

**END\_VAR**

t:= DT\_TO\_UDINT(dtInput);

iTimeZone:= siTimezoneOffset\*3600; // convert timezone offset from hours to seconds

t := t + iTimeZone; // compensate for timezone

IF xIsDST THEN // compensate for daylight saving time

t := t + iDstOffset;

END\_IF

**// Retrieve hours, minutes AND seconds**

usiSeconds := TO\_USINT(t MOD 60);

t := t / 60;

usiMinutes := TO\_USINT(t MOD 60);

t := t / 60;

usiHours := TO\_USINT(t MOD 24);

t := t / 24;

**//Convert Unix time to date**

a := (((4 \* t) + 102032) / 146097) + 15;

b := t + 2442113 + a - (a / 4);

c := ((20 \* b) - 2442) / 7305;

d := b - (365 \* c) - (c / 4);

e := d \* 1000 / 30601;

f := d - (e \* 30) - (e \* 601 / 1000);

**//January and February are counted as months 13 and 14 of the previous year**

IF (e <= 13) THEN

c := c - 4716;

e := e - 1;

ELSE

c := c - 4715;

e := e - 13;

END\_IF

**// Retrieve year, month and day**

iYear := TO\_INT(c);

usiMonth := TO\_USINT(e);

usiDay := TO\_USINT(f);

**// Calculate day of week**

**// January and February are counted as months 13 and 14 of the previous year**

IF (usiMonth <= 2) THEN

m := usiMonth + 12;

y := iYear - 1;

ELSE

m := usiMonth;

y := iYear;

END\_IF

j := y / 100; // j is the century

k := y MOD 100; // k the year of the century

h := usiDay + (26 \* (m + 1) / 10) + k + (k / 4) + (5 \* j) + (j / 4); // Compute h using Zeller's congruence

h := ((h + 5) MOD 7) + 1; // Return the day of the week

**// Convert DOW 1-7 till 0-6, week starts with Sunday at day 0**

IF h < 7 THEN

usiDOW := TO\_USINT(h);

ELSE usiDOW := 0;

END\_IF

**// Compensate DST, active from last Sunday in march till last Sunday in October**

**// The last Sunday in March and October is always on day 25,26,27,28,29,30,31**

usiPreviousSunday := usiDay - usiDOW;

IF usiMonth < 3 OR usiMonth > 10 THEN

xIsDST := FALSE;

ELSIF usiMonth > 3 AND usiMonth < 10 THEN

xIsDST := TRUE;

ELSIF usiMonth = 3 AND usiPreviousSunday >= 25 THEN

xIsDST := TRUE;

ELSIF usiMonth = 10 AND usiPreviousSunday < 25 THEN

xIsDST := TRUE;

ELSE xIsDST := FALSE;

END\_IF