# Stuktureret tekst pdf edition

# Globale variable

VAR  
 dirX : BOOL;  
 dirY : BOOL;  
 dirZ : BOOL;  
 enabX : BOOL;  
 enabY : BOOL;  
 enabZ : BOOL;  
 input : ARRAY[0..65535] OF USINT;  
 posX : UDINT;  
 posY : UDINT;  
 posZ : UDINT;  
 reset : BOOL;  
 stepX : BOOL;  
 stepY : BOOL;  
 stepZ : BOOL;  
 switch\_X : BOOL;  
 quit : BOOL;  
 sharpen : UINT;  
 status : USINT;  
 testEnab : BOOL;  
 z\_move\_down : BOOL;  
 z\_move\_up : BOOL;  
 y\_move\_right : BOOL;  
 y\_move\_left : BOOL;  
 x\_move\_right : BOOL;  
 x\_move\_left : BOOL;  
 switch\_Z : BOOL;  
 switch\_Y : BOOL;  
 newLineX : UDINT;  
 ValueZ : UINT;  
 penLength : REAL;  
 placeInArray : UDINT;  
 placement : UINT;  
 newLineHelper : UDINT;  
END\_VAR

# Draw

Locale variable

VAR  
 check : UDINT;  
 sharpenderLen : UINT;  
 timer : TON;  
 i : UDINT;  
 tempX : UDINT;  
 activator : BOOL;  
 finder : BOOL;  
 tempPlace : UINT;  
END\_VAR

Program

PROGRAM \_INIT  
   
 placeInArray := 0;   
 i := 0;  
 sharpen := 1;  
 activator := FALSE;   
 testEnab:= FALSE;  
 penLength := 0;  
   
   
   
END\_PROGRAM  
  
PROGRAM \_CYCLIC  
 ValueZ := REAL\_TO\_INT(penLength\*\*1.5);  
 sharpenderLen := REAL\_TO\_INT(45000/penLength);  
   
 IF (reset = FALSE AND testEnab = FALSE AND penLength > 5 ) THEN  
  
   
 IF switch\_X = FALSE OR switch\_Y = FALSE OR switch\_Z = FALSE THEN  
 enabX := TRUE;  
 enabY := TRUE;  
 enabZ := TRUE;  
 END\_IF   
   
 IF (enabX = FALSE) THEN  
 status:= 2;  
 END\_IF  
   
 IF (i = 30) THEN  
 IF placement = 78 THEN  
 newLineHelper := newLineHelper - 17 ;  
 posX := posX + 17;  
 ELSIF placement = 21041 THEN  
 FOR check := placeInArray TO finder BY check +1 DO  
 tempPlace := input[check];  
 IF tempPlace = 21041 THEN  
 finder := TRUE;  
 ELSIF tempPlace = 20016 OR tempPlace = 20017 THEN  
 placeInArray := check -1;  
 newLineHelper := newLineHelper + 17;  
 posX := posX - 17;  
 finder := TRUE;  
 END\_IF  
 END\_FOR   
 END\_IF  
 placeInArray := placeInArray + 1;  
 i := 0;  
 END\_IF   
 placement := input[placeInArray];  
   
 // here the robot will move to the shapener, sharpen the pencil and move back to were it came from  
 IF (sharpen >= 30000) THEN   
 IF (posZ >= 300 AND tempX > 0 AND activator = FALSE) THEN   
 dirZ := FALSE;  
 stepZ := NOT stepZ;  
 IF (stepZ = FALSE) THEN  
 posZ := posZ -1;  
 END\_IF  
   
 ELSIF (tempX > 0 AND posZ <= 300 AND activator = FALSE)THEN  
 dirX := FALSE;  
 stepX := NOT stepX;  
 IF (stepX = FALSE) THEN  
 tempX := tempX -1;  
 END\_IF  
   
 ELSIF (tempX = 0 AND posZ <= 1500 + sharpenderLen AND activator = FALSE) THEN  
 dirZ := TRUE;  
 stepZ := NOT stepZ;  
 IF (stepZ = FALSE) THEN  
 posZ := posZ +1;  
 END\_IF  
   
 ELSIF (tempX = 0 AND posZ >= 1500 + sharpenderLen AND activator = FALSE) THEN   
 timer.IN := TRUE;  
 timer.PT := T#6000ms;  
 IF timer.Q THEN  
 activator := TRUE;  
 END\_IF   
   
 ELSIF (tempX = 0 AND posZ > 300 AND activator = TRUE) THEN  
 dirZ := FALSE;  
 stepZ := NOT stepZ;  
 IF (stepZ = FALSE) THEN  
 posZ := posZ - 1;  
 END\_IF  
   
 ELSIF (tempX < posX AND posZ = 300 AND activator = TRUE) THEN   
 dirX := TRUE;  
 stepX := NOT stepX;  
 IF (stepX = FALSE) THEN  
 tempX := tempX +1;  
 END\_IF  
   
 ELSIF (tempX = posX AND posZ = 300 AND activator = TRUE) THEN   
 sharpen := 0;  
 activator := FALSE;  
 timer.IN := FALSE;  
   
 END\_IF  
   
   
 ELSE  
 // here the robot will draw  
 IF (placement = 68 AND i < 30) THEN  
 IF (posZ < (3200-ValueZ)) THEN   
 IF dirZ THEN   
 stepZ := NOT stepZ;  
 IF (stepZ = FALSE) THEN  
 posZ := posZ +1;  
 END\_IF  
 ELSE   
 dirZ := TRUE;  
 END\_IF  
   
 ELSE  
 IF dirX THEN  
 stepX := NOT stepX;  
 i := i + 1;  
 IF (stepX = FALSE) THEN  
 posX := posX + 1;  
 newLineX := newLineX +1;  
 sharpen := sharpen + 1;  
 tempX := posX;  
 END\_IF  
 ELSE  
 dirX := TRUE;  
 END\_IF   
   
 END\_IF   
   
 //dont draw  
 ELSIF placement = 85 AND i < 30 THEN  
 IF (posZ > (2900-ValueZ)) THEN // !!!!!!  
 IF dirZ = FALSE THEN   
 stepZ := NOT stepZ;  
 IF (stepZ = FALSE) THEN  
 posZ := posZ -1;  
 END\_IF  
 ELSE  
 dirZ := FALSE;  
 END\_IF   
 ELSE  
 IF dirX THEN  
 stepX := NOT stepX;  
 i := i + 1;  
 IF (stepX = FALSE) THEN  
 posX := posX +1;  
 newLineX := newLineX +1;  
 END\_IF  
 ELSE  
 dirX := TRUE;  
 END\_IF  
   
 END\_IF   
   
   
 ELSIF (placement = 78) THEN  
 IF (posZ > (2900-ValueZ) )THEN   
 IF dirZ = FALSE THEN  
 stepZ := NOT stepZ;  
 IF (stepZ = FALSE) THEN  
 posZ := posZ -1;  
 END\_IF  
 ELSE  
 dirZ := FALSE;  
 END\_IF  
   
 ELSIF (posZ <= (2900-ValueZ)) AND (newLineX >= (newLineHelper)) THEN  
 IF dirX = FALSE THEN  
 stepX := NOT stepX;  
 IF (stepX = FALSE) THEN  
 posX := posX - 1;  
 newLineX := newLineX -1;  
 END\_IF  
 ELSE  
 dirX := FALSE;  
 END\_IF  
   
 ELSIF ((posZ <= (2900-ValueZ)) AND (newLineX <= newLineHelper) AND (i < 30)) THEN  
 IF dirY THEN  
 stepY := NOT stepY;  
 i := i +1;  
 IF (stepY = FALSE) THEN  
 posX := posX - 1;  
 newLineX := newLineX -1;  
 END\_IF  
 ELSE  
 dirY := TRUE;  
 END\_IF  
   
 END\_IF  
   
 ELSIF placement = 81 THEN  
   
 quit := TRUE;  
 END\_IF   
  
 END\_IF  
 END\_IF   
 timer();  
END\_PROGRAM

# Reset

Lokale variable

VAR  
 resetX : USINT;  
 resetY : USINT;  
 resetZ : USINT;  
 timer : TON;  
END\_VAR

Program

PROGRAM \_INIT  
 reset := TRUE;  
 resetX := 0;  
 resetY := 0;  
 resetZ := 0;  
  
 quit := FALSE;  
 status := 1;  
 testEnab := TRUE;  
END\_PROGRAM  
  
PROGRAM \_CYCLIC  
  
//   
 IF (reset = TRUE AND testEnab = FALSE) THEN  
 IF switch\_X = TRUE AND resetX = 0 THEN // Resets the x-axis   
 enabX:= FALSE;  
 dirX:= FALSE;  
 stepX := NOT stepX;  
 ELSIF switch\_X = FALSE THEN   
 resetX := 1;  
 dirX:= TRUE;  
 stepX := NOT stepX;  
 END\_IF   
   
 IF switch\_Y = TRUE AND resetY = 0 THEN // Resets the y-axis  
 enabY:= FALSE;  
 dirY:= FALSE;  
 stepY := NOT stepY;  
 ELSIF switch\_Y = FALSE THEN  
 resetY := 1;   
 dirY:= TRUE;  
 stepY := NOT stepY;  
 END\_IF  
   
 IF switch\_Z = TRUE AND resetZ = 0 THEN // Resets the z-axis  
 enabZ:= FALSE;  
 dirZ:= FALSE;  
 stepZ := NOT stepZ;  
 ELSIF switch\_Z = FALSE THEN  
 resetZ := 1;  
 dirZ:= TRUE;  
 stepZ := NOT stepZ;   
 END\_IF  
   
// sets the position of the variables posX, posY and posZ  
 timer.IN := TRUE;  
 timer.PT := T#100ms;  
 IF timer.Q THEN   
 IF resetX = 1 AND resetY = 1 AND resetZ = 1 THEN  
 posX := 0;  
 posY := 0;  
 posZ := 0;  
 resetX := 2;  
 END\_IF   
 timer.IN := FALSE;  
 END\_IF  
   
   
 // if start-button pressed, then go to start position   
 IF quit = FALSE AND resetX = 2 THEN  
 IF posX < 1900 THEN  
 dirX := TRUE;  
 stepX := NOT stepX;  
 IF (stepX = FALSE) THEN  
 posX := posX +1;  
 END\_IF  
 ELSE  
 reset:= FALSE;  
 resetX := 0;   
 resetY := 0;   
 resetZ := 0;   
 newLineX := 2294967295;  
 newLineHelper := 2294967295;  
 END\_IF  
 END\_IF  
 END\_IF   
   
   
   
   
   
 // move x-axis left   
 IF (x\_move\_left AND testEnab) THEN   
 stepX := NOT stepX;  
 dirX := TRUE;  
 IF stepX = FALSE THEN  
 posX := posX +1;  
 END\_IF  
 END\_IF  
   
 // move x-axis right  
 IF (x\_move\_right AND testEnab) THEN   
 stepX := NOT stepX;  
 dirX := FALSE;  
 IF stepX = FALSE THEN  
 posX := posX -1;  
 END\_IF  
 END\_IF  
   
 // move y-axis left  
 IF (y\_move\_left AND testEnab) THEN   
 stepY := NOT stepY;  
 dirY := TRUE;  
 IF stepY = FALSE THEN  
 posY := posY +1;  
 END\_IF  
 END\_IF   
   
 // move y-axis right   
 IF (y\_move\_right AND testEnab) THEN   
 stepY := NOT stepY;  
 dirY := FALSE;  
 IF stepY = FALSE THEN  
 posY := posY -1;  
 END\_IF  
 END\_IF  
   
 // move z-axis up   
 IF (z\_move\_up AND testEnab) THEN   
 stepZ := NOT stepZ;  
 dirZ := FALSE;  
 IF stepZ = FALSE THEN  
 posZ := posZ -1;  
 END\_IF  
 END\_IF  
   
 // move z-axis down   
 IF (z\_move\_down AND testEnab) THEN   
 stepZ := NOT stepZ;  
 dirZ := TRUE;  
 IF stepZ = FALSE THEN  
 posZ := posZ +1;  
 END\_IF   
 END\_IF  
   
   
   
   
   
   
   
 timer();  
END\_PROGRAM

# EmergencyStop

Lokale variable

# VAR emergencyStop : BOOL; reset\_quit : BOOL; END\_VAR

Program   
PROGRAM \_INIT  
   
 reset\_quit := FALSE;  
 emergencyStop := FALSE;  
   
END\_PROGRAM  
  
PROGRAM \_CYCLIC  
 IF (emergencyStop) THEN  
 enabX := TRUE;  
 enabY := TRUE;  
 enabZ := TRUE;  
 status := 0;  
   
 END\_IF  
   
 IF (reset\_quit) THEN  
 reset := TRUE;  
 quit := TRUE;  
 status := 1;  
 reset\_quit := FALSE;  
 placeInArray := 0;  
   
 END\_IF  
   
   
   
END\_PROGRAM

# TCP

Lokale variable

VAR  
 reciveData : ARRAY[0..199999] OF BOOL;  
 tcp2 : TcpServer;  
 tcp1 : TcpOpen;  
 state : UINT;  
 tcp3 : TcpRecv;  
 tcp4 : TcpSend;  
 tcp5 : TcpClose;  
END\_VAR

Program

PROGRAM \_INIT  
 state := 1;  
   
END\_PROGRAM  
  
PROGRAM \_CYCLIC  
 CASE state OF  
 1:  
 tcp1.enable := TRUE;  
 tcp1.port := 12345;  
 tcp1.options := tcpOPT\_REUSEADDR;  
 tcp1();  
 IF tcp1.status = 0 THEN  
 state := 10;  
 END\_IF   
   
 10:  
 tcp2.enable := TRUE;  
 tcp2.ident := tcp1.ident;  
 tcp2();  
 IF tcp2.status = 0 THEN  
 state := 20;  
 END\_IF   
   
 20:  
 tcp3.enable := TRUE;  
 tcp3.ident := tcp2.identclnt;  
 tcp3.pData := ADR(input);  
 tcp3.datamax := SIZEOF (input);  
 tcp3();  
 IF tcp3.status = 0 THEN  
  
 state := 30;  
   
 END\_IF   
  
 30:  
 tcp4.enable := TRUE;  
 tcp4.ident := tcp2.identclnt;  
 tcp4.pData := ADR(input);  
 tcp4.datalen := brsstrlen(ADR(input));  
 tcp4();  
 IF tcp4.status = 0 THEN  
 state := 40;  
 END\_IF   
   
 40:  
 tcp5.enable := TRUE;  
 tcp5.ident := tcp2.identclnt;  
 tcp5();  
 IF tcp5.status = 0 THEN  
 tcp2.enable := FALSE;  
 state := 10;  
   
 END\_IF   
 END\_CASE  
  
END\_PROGRAM  
  
PROGRAM \_EXIT  
 tcp5.enable := TRUE;  
 tcp5.ident := tcp2.identclnt;  
 tcp5();  
   
 tcp5.enable := TRUE;  
 tcp5.ident := tcp1.ident;  
 tcp5();  
   
END\_PROGRAM