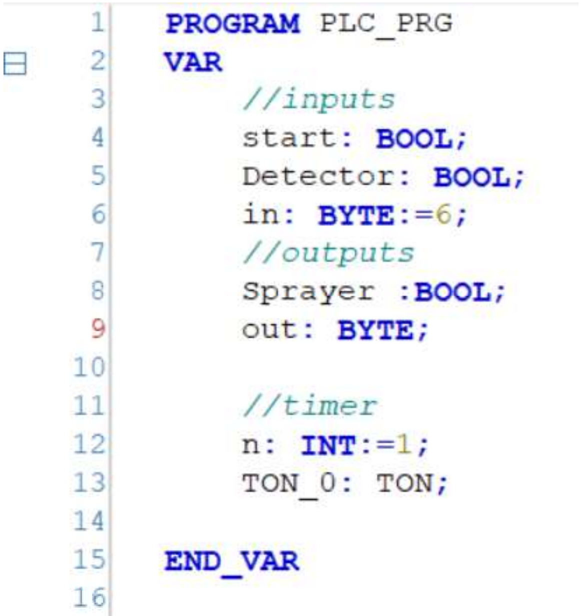
Exercise 7

Task 1

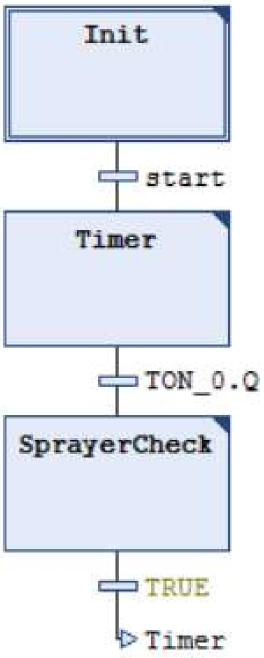
The system has an "**Inital-step**" where the sprayer output in set to off ("0"). The transistion from initial step to timer step is start. So to "start" the system one has to press a "button".
"**Timer-step**" has a timer to simulate a running system and a shift right register. The transistion from timer step to SprayerCheck step is TON_0.Q which "1"(TRUE) every 3 seconds when the timer resets.
"**SprayerCheck-step**", checks whether the output from the shift register out.5 is high. If it is high, the sprayer output will go high.
After the system is started, the states will go from Timer step to SprayerCheck step, and back to timer and soforth continuously.

Inputs/outputs etc.

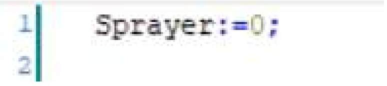


```
PROGRAM PLC_PRG
VAR
    //inputs
    start: BOOL;
    Detector: BOOL;
    in: BYTE:=6;
    //outputs
    Sprayer :BOOL;
    out: BYTE;

    //timer
    n: INT:=1;
    TON_0: TON;

END_VAR

```

SFC(block diagram)

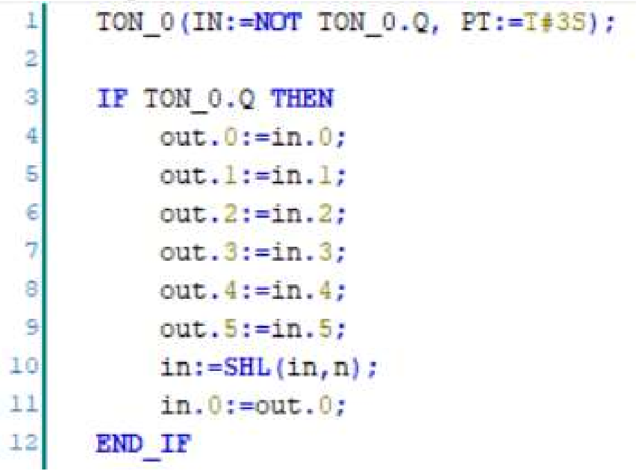


Initial step



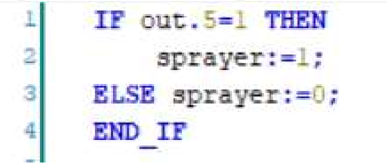
```
Sprayer:=0;

```

Timer step



```
TON_0(IN:=NOT TON_0.Q, PT:=T#3S);

IF TON_0.Q THEN
    out.0:=in.0;
    out.1:=in.1;
    out.2:=in.2;
    out.3:=in.3;
    out.4:=in.4;
    out.5:=in.5;
    in:=SHL(in,n);
    in.0:=out.0;
END_IF
```

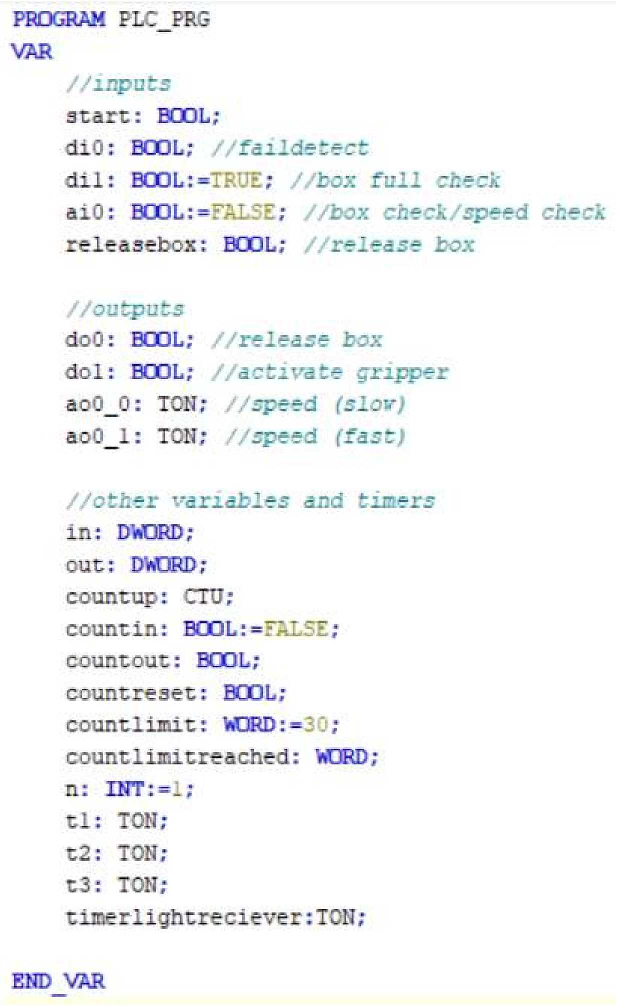
SprayerCheck step



```
IF out.5=1 THEN
    sprayer:=1;
ELSE sprayer:=0;
END_IF
```

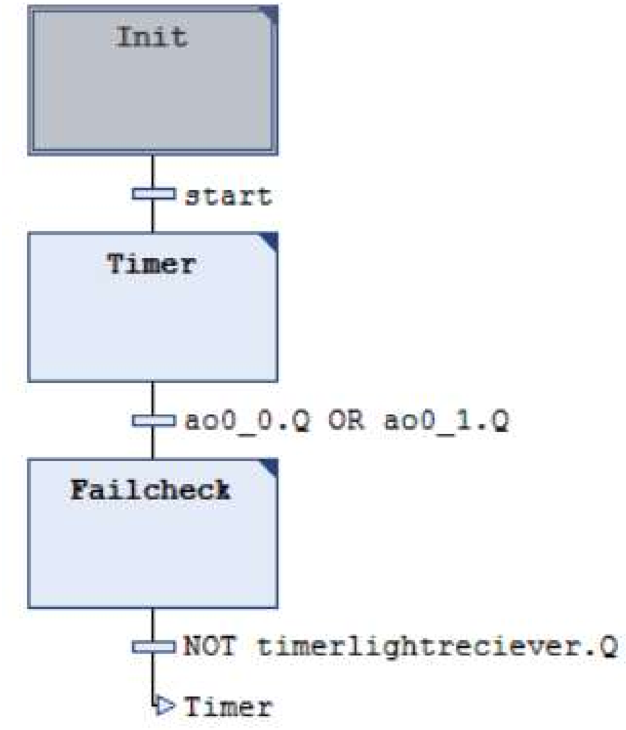
Task 2

Inputs/outputs etc.



```
PROGRAM PLC_PRG
VAR
    //inputs
    start: BOOL;
    di0: BOOL; //faildetect
    di1: BOOL:=TRUE; //box full check
    ai0: BOOL:=FALSE; //box check/speed check
    releasebox: BOOL; //release box

    //outputs
    do0: BOOL; //release box
    do1: BOOL; //activate gripper
    ao0_0: TON; //speed (slow)
    ao0_1: TON; //speed (fast)

    //other variables and timers
    in: DWORD;
    out: DWORD;
    countup: CTU;
    countin: BOOL:=FALSE;
    countout: BOOL;
    countreset: BOOL;
    countlimit: WORD:=30;
    countlimitreached: WORD;
    n: INT:=1;
    t1: TON;
    t2: TON;
    t3: TON;
    timerlightreciever:TON;

END_VAR
```

SFC(block diagram)



Description of the program sequence

I added a initial state just to secure that the program doesn't just run without pressing "start". The transition from **Timer-step** to **Failcheck-step** is the output for the speedcontrol timers. And lastly the transition from **Failcheck-step** to **Timer-step** is "NOT timerlightreciever.Q". The timerlightreciever.Q output only goes high when the output box is full. This ensures that the system stops when the output box is full.

Calculations for speed control

$$Lengthbelt := 8.4\,\mathrm{m} = 8.4\ \mathrm{m}$$

$$Speedfast := 70\,\frac{\mathrm{cm}}{\mathrm{s}} = 70\ \frac{\mathrm{cm}}{\mathrm{s}}$$

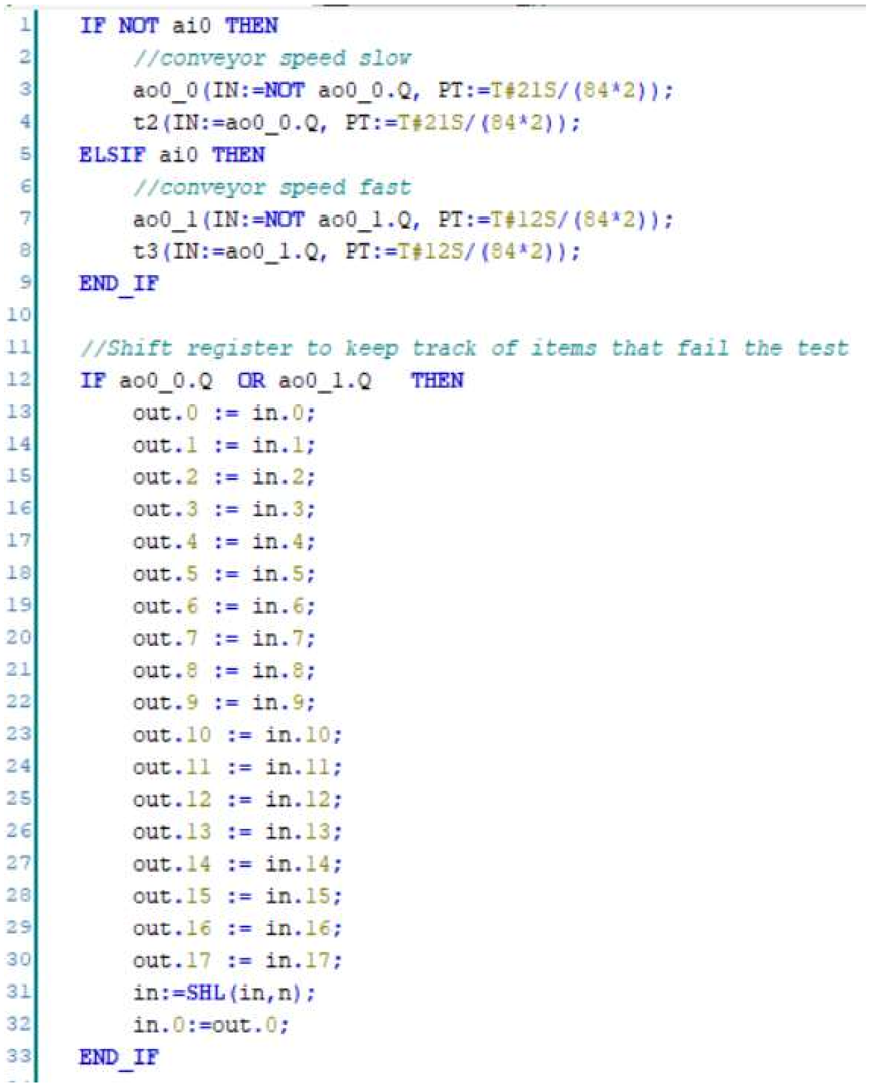
$$SpeedSlow := 40\,\frac{\mathrm{cm}}{\mathrm{s}} = 40\ \frac{\mathrm{cm}}{\mathrm{s}}$$

The time it takes for a box to go from one end of the conveyor to the other depending on the speed.

$$ReleasetimeFast := \frac{Lengthbelt}{Speedfast} = 12.00000000\ \mathrm{s}$$

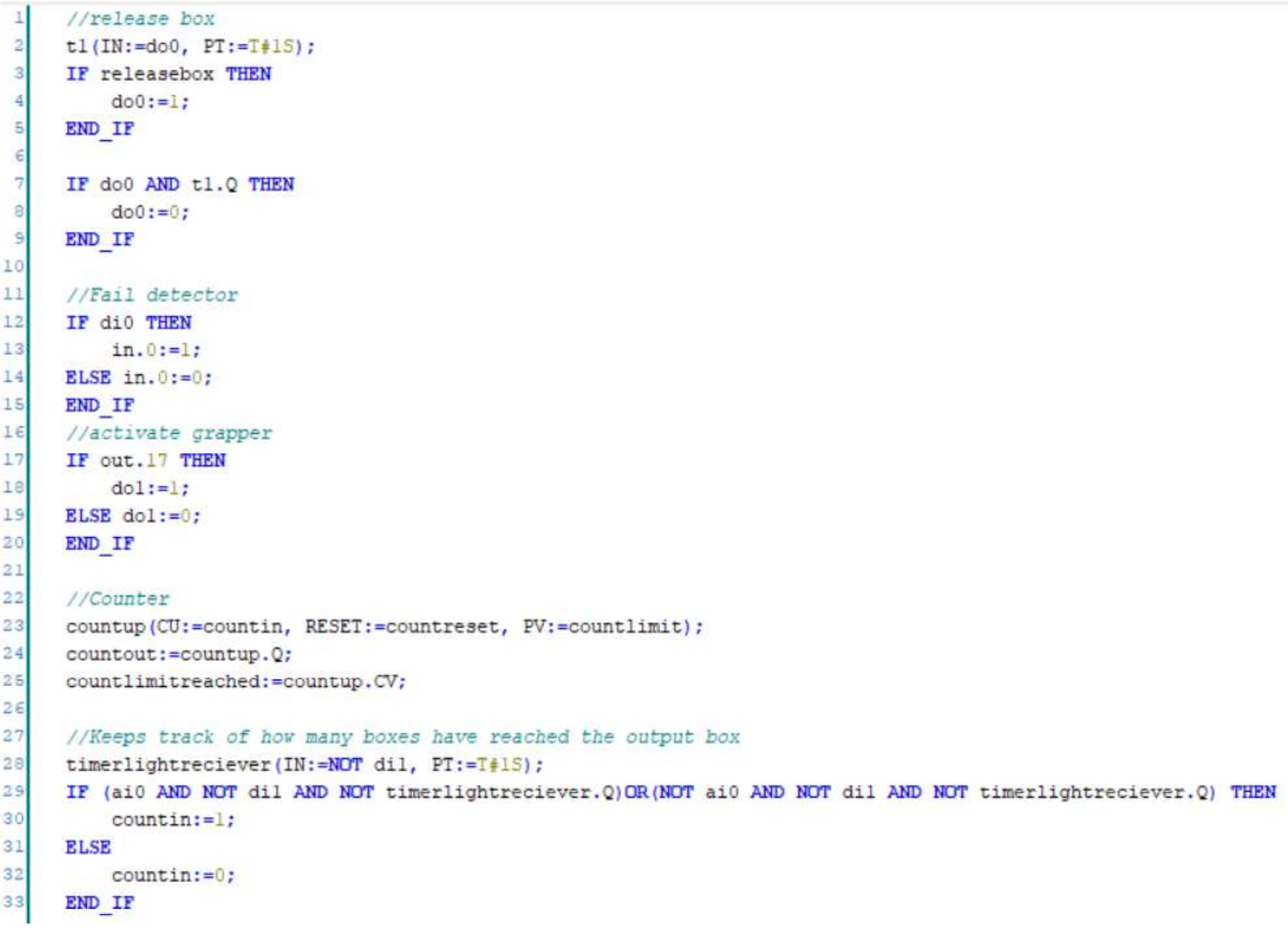
$$RelasetimeSlow := \frac{Lengthbelt}{SpeedSlow} = 21.00000000\ \mathrm{s}$$

In "**Timer-step**" the shift left register keeps track of how far the "failed box" has moved from "fail-detector" to "gripper". There are 18 shifts, from 0 to 17. each representing 10cm on the conveyor belt. This is ensured by dividing the time it takes for the box to go from one end to the other by 84 increments, which is the lowest common demoniator for 21 and 12, and also the length of the conveyorbelt (840cm) divided by 10. So this determines the fast/slow speed of the timers to control the shift register of the conveyorbelt for controlling the grapper.



```
IF NOT ai0 THEN
    //conveyor speed slow
    ao0_0(IN:=NOT ao0_0.Q, PT:=T#21S/(84*2));
    t2(IN:=ao0_0.Q, PT:=T#21S/(84*2));
ELSIF ai0 THEN
    //conveyor speed fast
    ao0_1(IN:=NOT ao0_1.Q, PT:=T#12S/(84*2));
    t3(IN:=ao0_1.Q, PT:=T#12S/(84*2));
END_IF

//Shift register to keep track of items that fail the test
IF ao0_0.Q  OR ao0_1.Q   THEN
    out.0 := in.0;
    out.1 := in.1;
    out.2 := in.2;
    out.3 := in.3;
    out.4 := in.4;
    out.5 := in.5;
    out.6 := in.6;
    out.7 := in.7;
    out.8 := in.8;
    out.9 := in.9;
    out.10 := in.10;
    out.11 := in.11;
    out.12 := in.12;
    out.13 := in.13;
    out.14 := in.14;
    out.15 := in.15;
    out.16 := in.16;
    out.17 := in.17;
    in:=SHL(in,n);
    in.0:=out.0;
END_IF
```

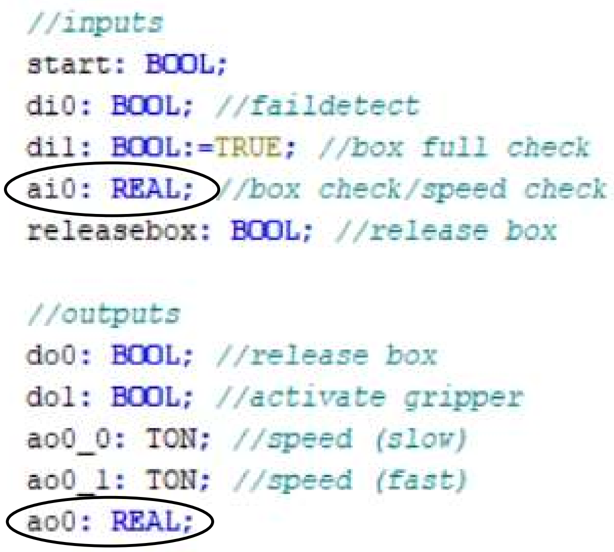
"**Failcheck-step**" controls that the boxes are released with appropriate distances between them by using a timer to keep the signal on for exactly 1 second and then turn it off.
If the input signal from the fail detector is high the input to the shift register "in.0" goes high and 18 shifts later the grapper is activated (do1 goes high).
To keep track of how many items have reached the output box, the sensor di1 is used. If di1 goes low, the counter counts up. The timerlightreciever is added so that di1 can be used as a counter and to check if the output box is full.



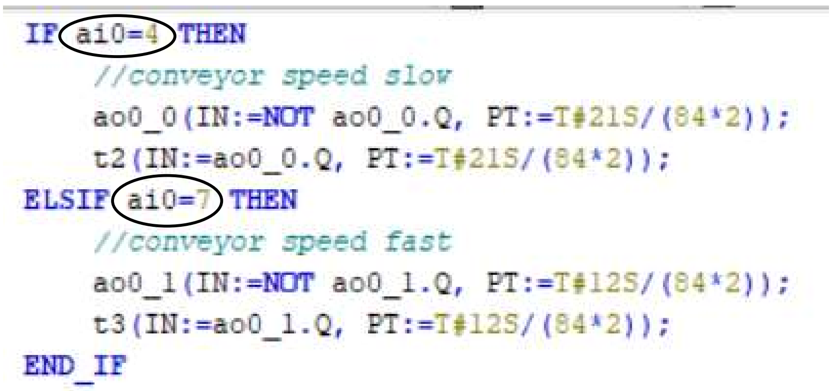
```
//release box
t1(IN:=do0, PT:=T#1S);
IF releasebox THEN
    do0:=1;
END_IF

IF do0 AND t1.Q THEN
    do0:=0;
END_IF

//Fail detector
IF di0 THEN
    in.0:=1;
ELSE in.0:=0;
END_IF
//activate grapper
IF out.17 THEN
    do1:=1;
ELSE do1:=0;
END_IF

//Counter
countup(CU:=countin, RESET:=countreset, PV:=countlimit);
countout:=countup.Q;
countlimitreached:=countup.CV;

//Keeps track of how many boxes have reached the output box
timerlightreciever(IN:=NOT di1, PT:=T#1S);
IF (ai0 AND NOT di1 AND NOT timerlightreciever.Q)OR(NOT ai0 AND NOT di1 AND NOT timerlightreciever.Q) THEN
    countin:=1;
ELSE
    countin:=0;
END_IF
```

Additions and corrections to exercise 7 Task 2 hand-in

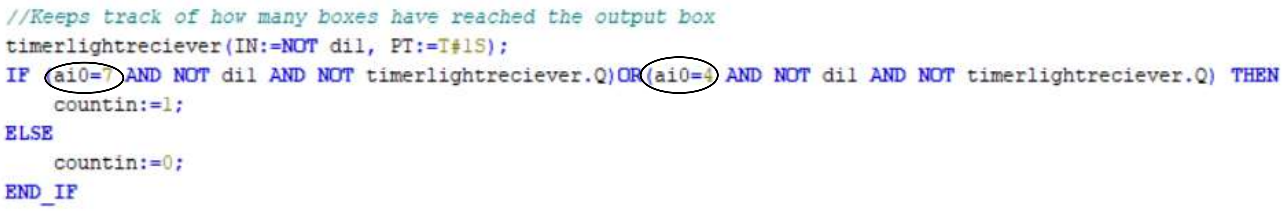
I have changed the analog input ai0 from BOOL to REAL to account for the values given from the sensor (4V and 7V). The output ao0 has been added, and i assume that the value of ao0 is the same as the value of ai0. Now the shift register shifts at a low speed if ai0 is equal to 4 and at high speed if ai0 is equal to 7.



```
//inputs
start: BOOL;
di0: BOOL; //faildetect
di1: BOOL:=TRUE; //box full check
ai0: REAL; //box check/speed check
releasebox: BOOL; //release box

//outputs
do0: BOOL; //release box
do1: BOOL; //activate gripper
ao0_0: TON; //speed (slow)
ao0_1: TON; //speed (fast)
ao0: REAL;
```



```
IF ai0=4 THEN
    //conveyor speed slow
    ao0_0(IN:=NOT ao0_0.Q, PT:=T#21S/(84*2));
    t2(IN:=ao0_0.Q, PT:=T#21S/(84*2));
ELSIF ai0=7 THEN
    //conveyor speed fast
    ao0_1(IN:=NOT ao0_1.Q, PT:=T#12S/(84*2));
    t3(IN:=ao0_1.Q, PT:=T#12S/(84*2));
END_IF
```



```
//Keeps track of how many boxes have reached the output box
timerlightreciever(IN:=NOT di1, PT:=T#1S);
IF (ai0=7 AND NOT di1 AND NOT timerlightreciever.Q)OR(ai0=4 AND NOT di1 AND NOT timerlightreciever.Q) THEN
    countin:=1;
ELSE
    countin:=0;
END_IF
```