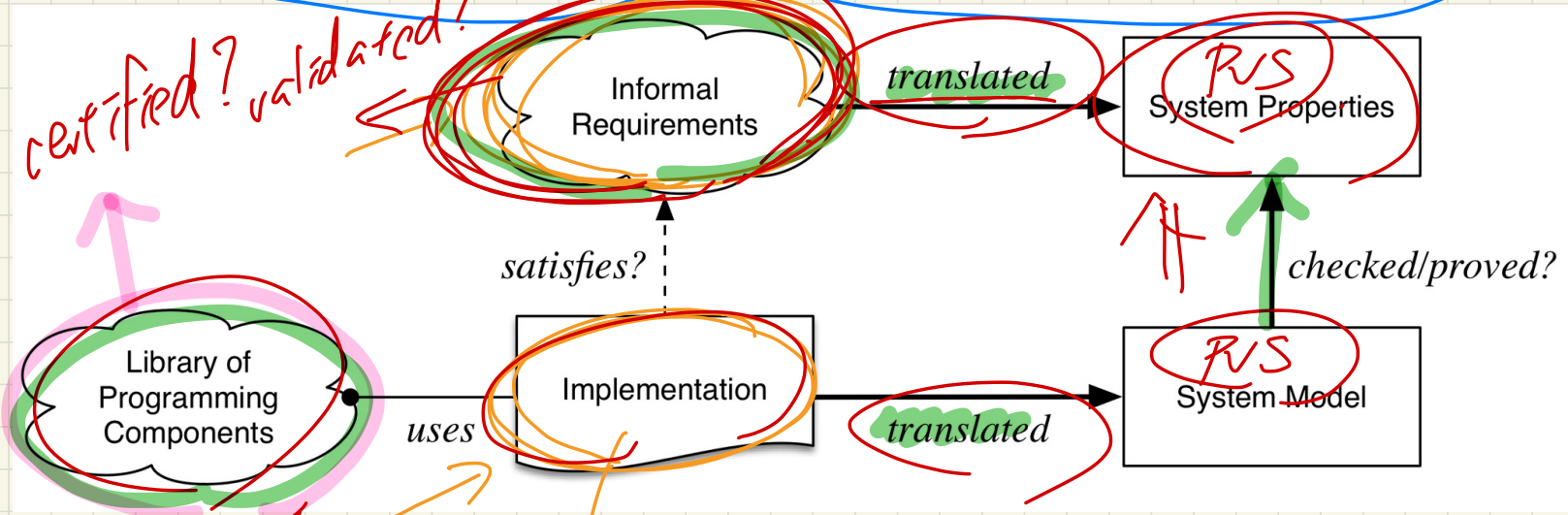
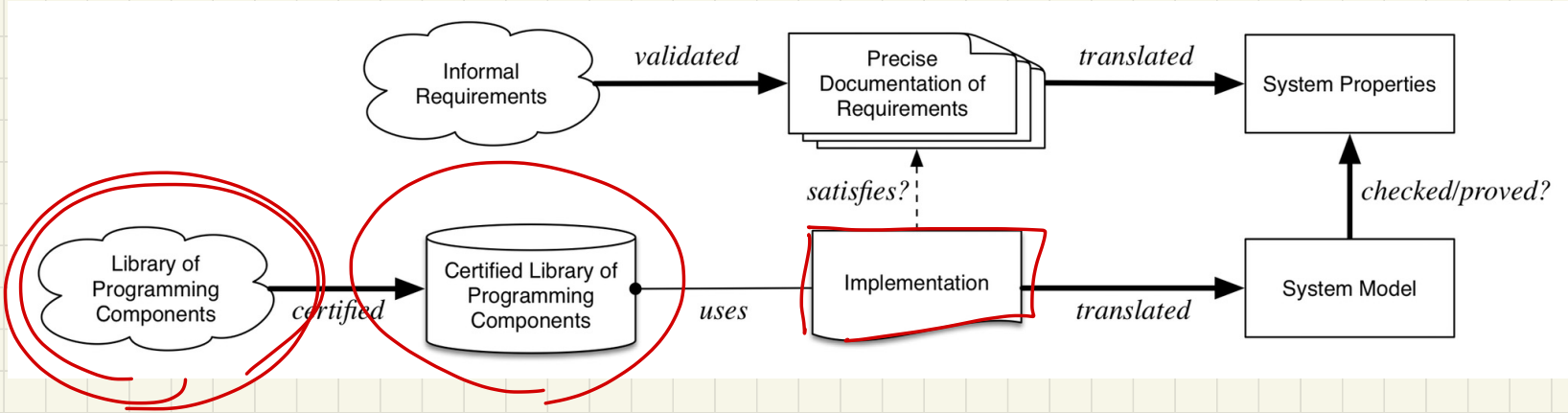
Guest Lecture

Tuesday October 22

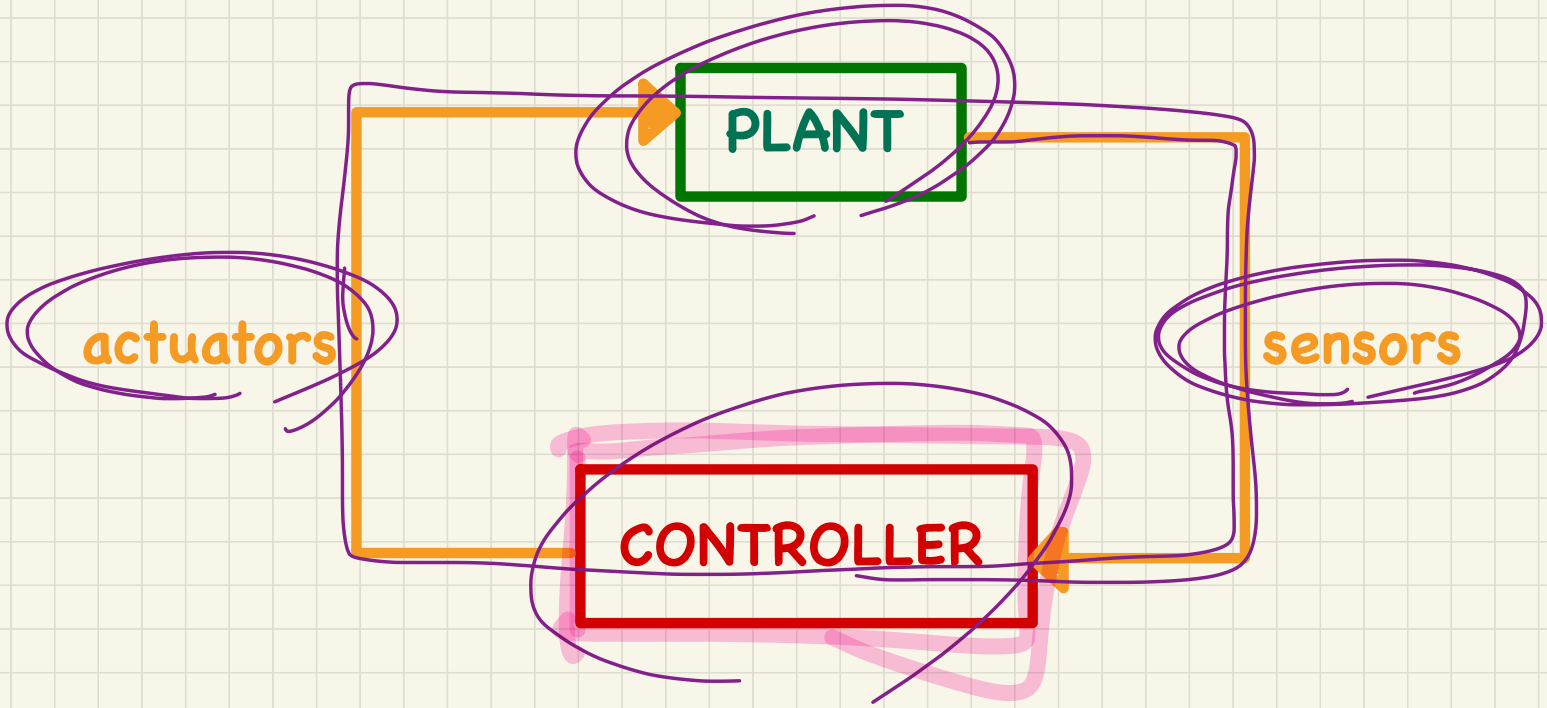
# Building the product right? Building the right product?



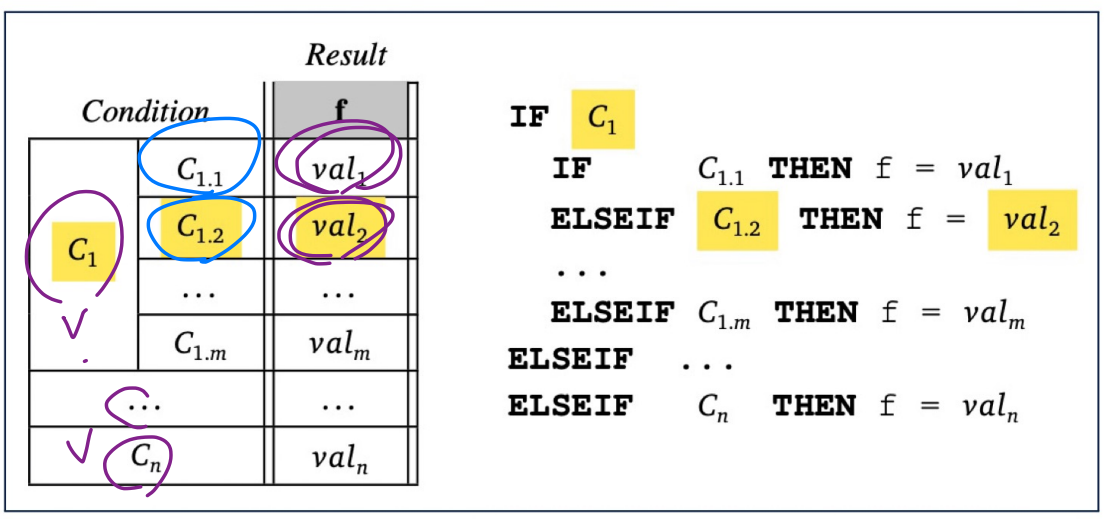
# Building the right product right?



# Cyber-Physical Systems: Plant, Sensors, Controller, Actuators



# Function Tables



| Condition | | Result f |
|---|---|---|
| $C_1$ | $C_{1.1}$ | $val_1$ |
| | $C_{1.2}$ | $val_2$ |
| | ... | ... |
| | $C_{1.m}$ | $val_m$ |
| ... | | ... |
| $C_n$ | | $val_n$ |

```
IF C1
   IF      C1.1 THEN f = val1
   ELSEIF  C1.2 THEN f = val2
   ...
   ELSEIF  C1.m THEN f = val_m
ELSEIF ...
ELSEIF     Cn   THEN f = val_n
```

$x = 3$.

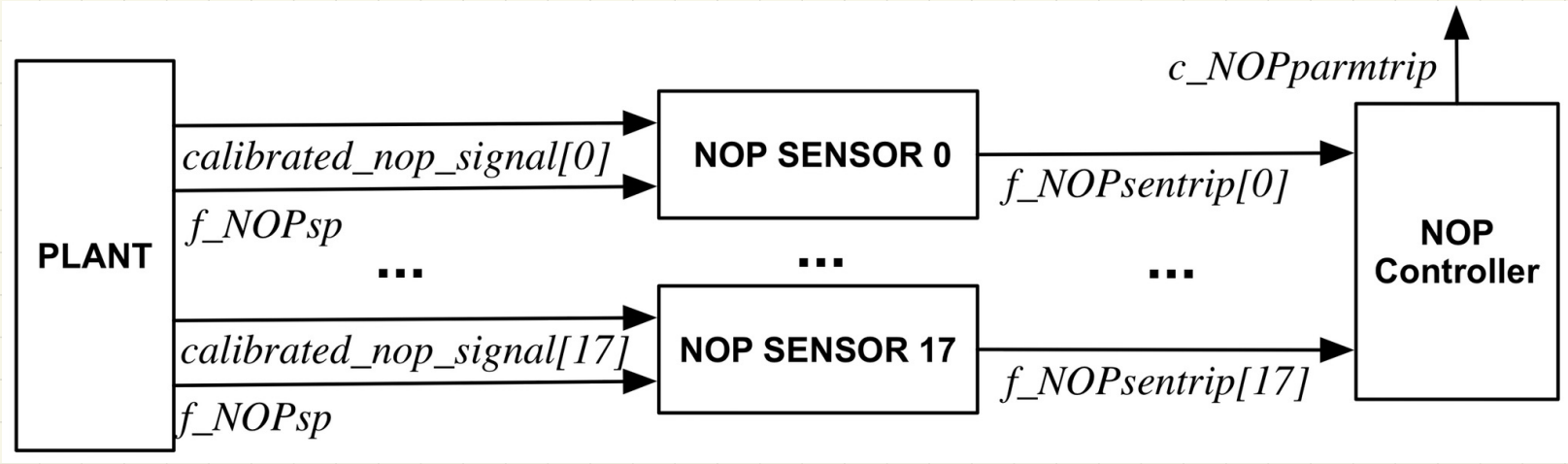
| | Result |
|---|---|
| $x \geq 0$ | V1 |
| $x \leq 0$ | V2 |

$\neg(C_{1.1} \wedge C_{1.2})$

Proof obligation of Completeness?

Proof obligation of Disjointness?

# Example CPS: Neutron OverPower Unit of Darlington SDS



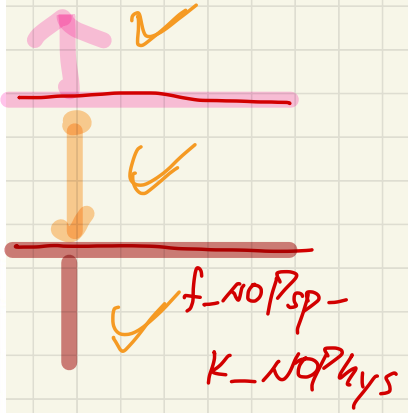
# Formalizing Requirements of NOP using Function Tables

| Condition | Result c_NOPparmtrip |
|---|---|
| $\exists i \in 0..17 \bullet f\_NOPsentrip[i] = e\_Trip$ | e_Trip |
| $\forall i \in 0..17 \bullet f\_NOPsentrip[i] = e\_NotTrip$ | e_NotTrip |

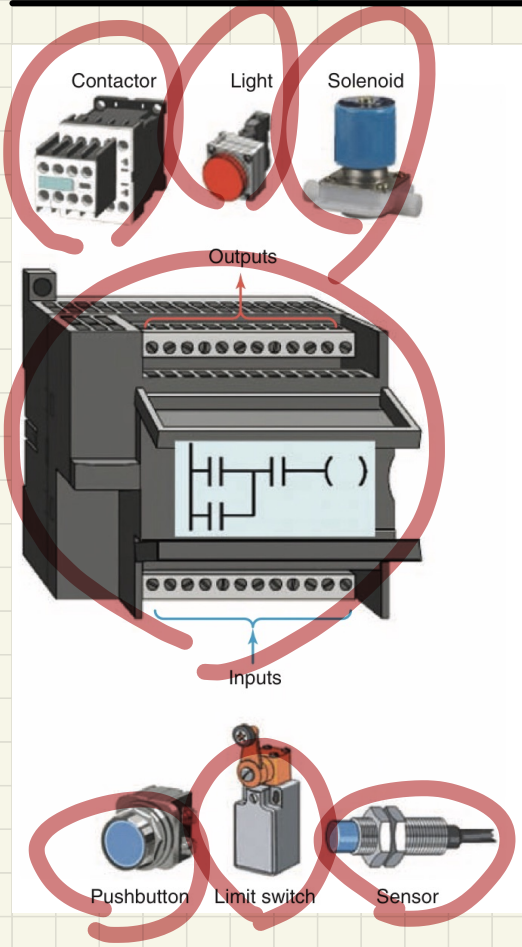
**Table:** NOP Controller

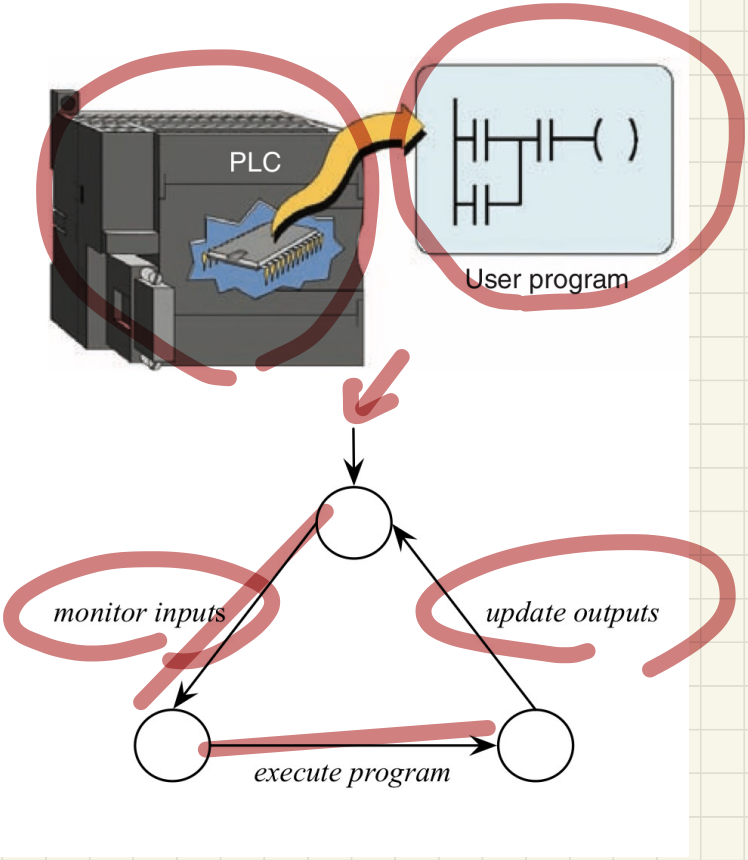
| Condition | Result f_NOPsentrip[i] |
|---|---|
| $calibrated\_nop\_signal[i] \geq f\_NOPsp$ | e_Trip |
| $f\_NOPsp - k\_NOPhys < calibrated\_nop\_signal[i] < f\_NOPsp$ | $(f\_NOPsentrip[i])_{-1}$ |
| $calibrated\_nop\_signal[i] \leq f\_NOPsp - k\_NOPhys$ | e_NotTrip |

**Table:** NOP sensor $i$, $i \in 0..17$ (monitoring $calibrated\_nop\_signal[i]$)

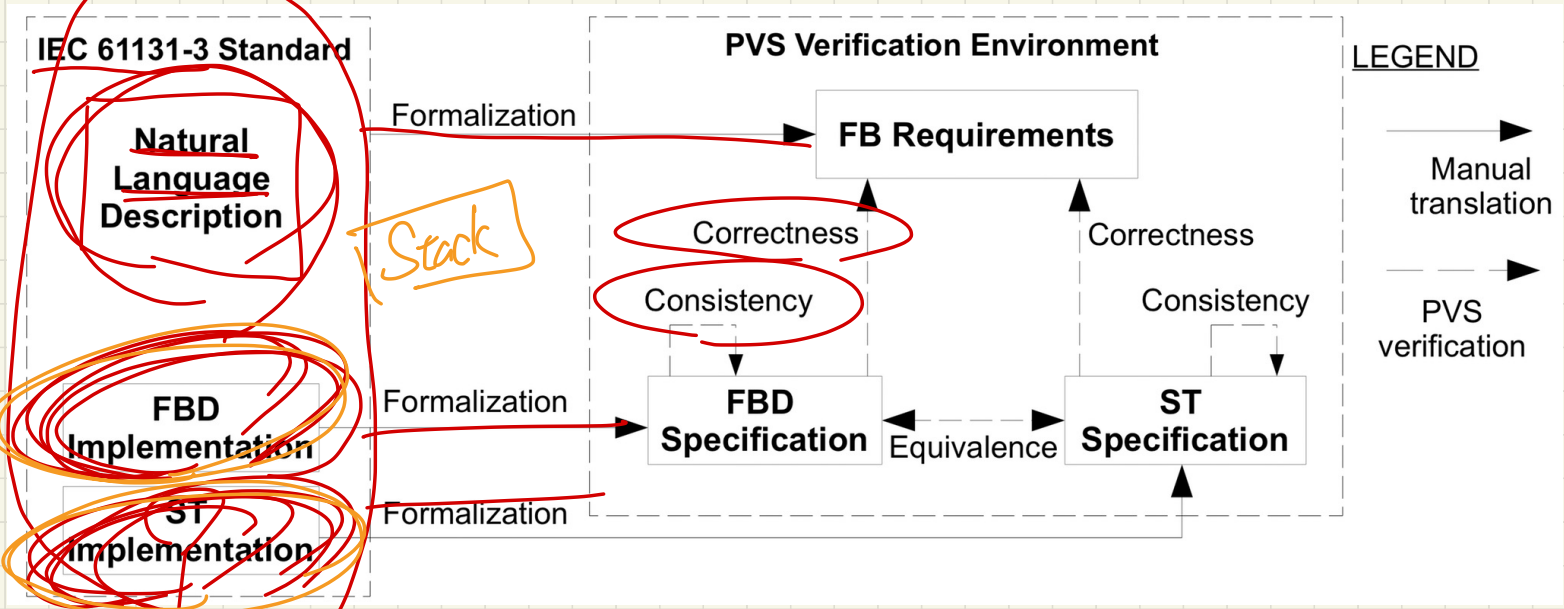


# PLC as a Cyclic Executive

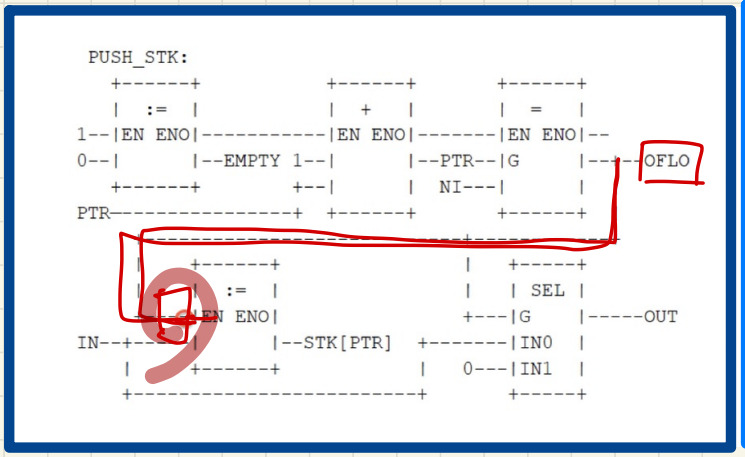


# Certifying IEC 61131-3 Function Block Library



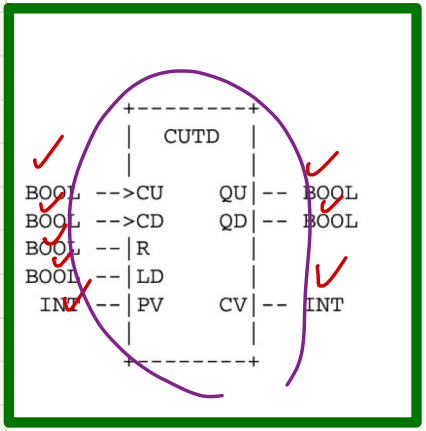
# IEC 61131-3 Annex F: Stack of Integers




```
ELSIF PUSH & NOT OFLO THEN
   EMPTY := 0; PTR := PTR+1; OFLO := (PTR = NI);
   IF NOT OFLO THEN OUT := IN ; STK[PTR] := IN;
   ELSE OUT := 0;
   END_IF ;
END_IF ;
```

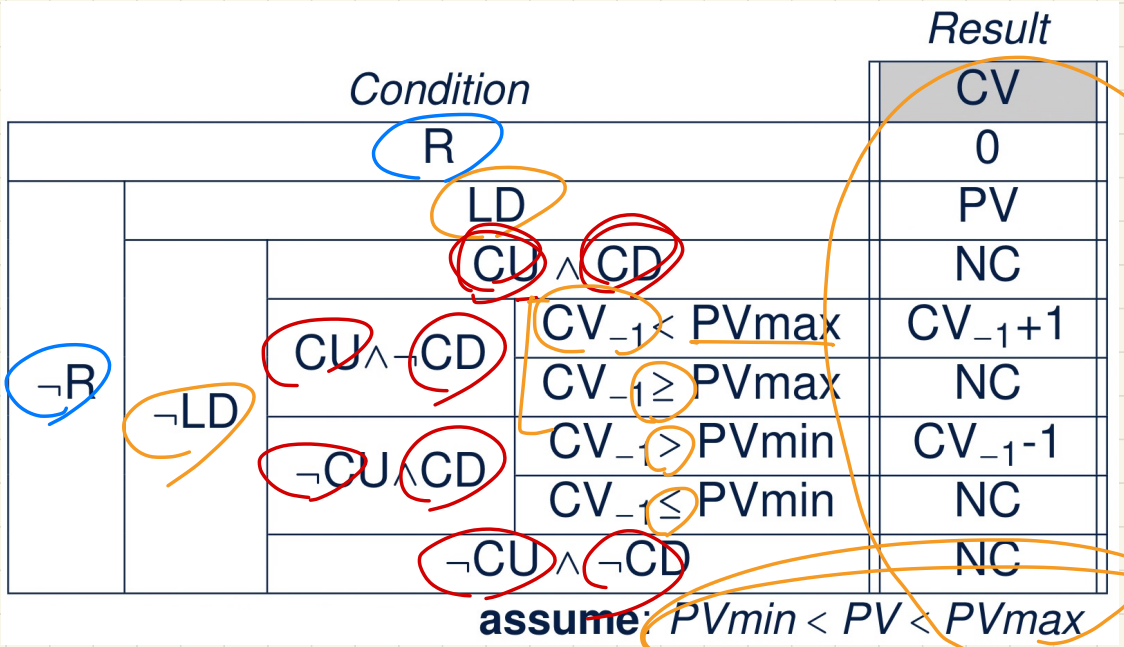
if OFLO THEN OUT := IN

# IEC 61131-3 Annex F: Counters




```
FUNCTION_BLOCK CTUD
  VAR_INPUT
    CU, CD : BOOL R_EDGE; (* Value to be counted up/down *)
    R      : BOOL         (* Reset *)
    LD     : BOOL         (* Load value flag *)
    PV     : INT          (* Preset value *)
  END_VAR
  VAR_OUTPUT
    QU : BOOL (* Compare CV with PV for up counter *)
    QD : BOOL (* Compare CV with 0 for down counter *)
    CV : INT  (* Current counted value *)
  END_VAR
  IF R THEN CV := 0 ;
  ELSIF LD THEN CV := PV ;
  ELSE
    IF NOT (CU AND CD) THEN
      IF CU AND (CV < PVmax)
      THEN CV := CV + 1 ;
      ELSIF CD AND (CV > PVmin)
      THEN CV := CV - 1 ;
      END IF ;
    END IF ;
  END IF ;
  QU := (CV >= PV) ;
  QD := (CV <= 0) ;
END_FUNCTION_BLOCK
```

# Formalizing Requirements of Counters using Function Tables



| Condition | | | | Result |
|---|---|---|---|---|
| | | | | CV |
| R | | | | 0 |
| ¬R | LD | | | PV |
| | ¬LD | CU ∧ CD | | NC |
| | | CU ∧ ¬CD | $CV_{-1} < PVmax$ | $CV_{-1}+1$ |
| | | | $CV_{-1} \geq PVmax$ | NC |
| | | ¬CU ∧ CD | $CV_{-1} > PVmin$ | $CV_{-1}-1$ |
| | | | $CV_{-1} \leq PVmin$ | NC |
| | | ¬CU ∧ ¬CD | | NC |

**assume**: $PVmin < PV < PVmax$

Proof obligation of Completeness?

Proof obligation of Disjointness?