APC7
more switching, more elements and more inventory control

Sigurd Skogestad
30 Sep. 2025

Change of active constraints. Four cases

MV-MV switching (because MV may saturate)

- Need many MVs to cover whole steady-state range
- Use only one MV at a time
- Three options:
  - A1. Split range control,
  - A2. Different setpoints,
  - A3. Valve position control (VPC)

CV-CV switching (because we may reach new CV constraint)

- Must select between CVs
- «Only» option: Many controllers with **selector**

Already covered

Now: MV-CV switching

Simple MV-CV switching: CV can be given up when reach MV saturation

- This means we followed «input saturation rule»
- Don't need to do anything

Complex MV-CV switching: CV cannot be given up (need to «repair loops»)

- Must combine MV-MV switching (three options) with CV-CV switching (selector)

Adriana Reyes-Lua and Sigurd Skogestad, Systematic Design of Active Constraint Switching Using Classical Advanced Control Structures, Ind.Eng.Chem.Res, 2020

Note: we are here assuming that the constraints are not conflicting so that switching is possible

Simple MV-CV switching

- When MV (u) saturates, we can give up the CV (y).
  - Don’t need to do anything, except having anti-windup in the controller
- This is because we have followed the

Input saturation rule: “Pair a MV that may saturate with a CV that can be given up (when the MV saturates)”

- Many examples (that it works is not always so obvious!)
  - 1. Driving as fast as possible to the airport
  - 2. Heating of cabin in the winter
  - 3. Anti-surge control

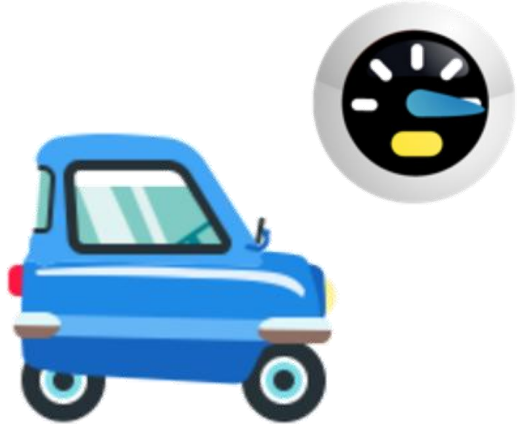
NTNU

Optimization with PI-controller

max y

s.t. $y \le y^{max}$

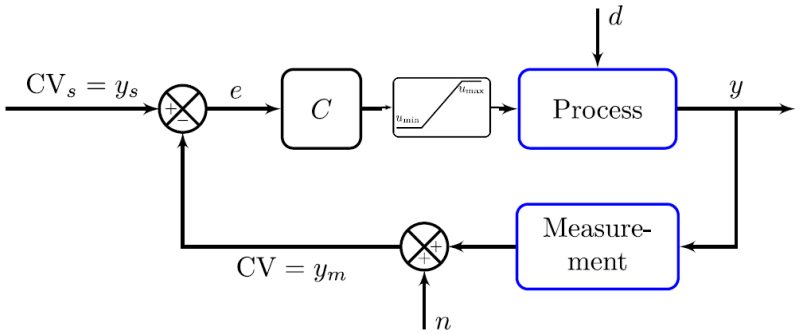
$u \le u^{max}$



Example: Drive as fast as possible to airport (u=power, y=speed, y^{max} = 110 km/h)

- **Optimal solution has two active constraint regions:**
  1. $y = y^{max}$ → **speed limit**
  2. $u = u^{max}$ → **max power**

- **Solved with PI-controller**
  - $y^{sp} = y^{max}$
  - Anti-windup: I-action is off when $u=u^{max}$



s.t. = subject to

y = CV = controlled variable

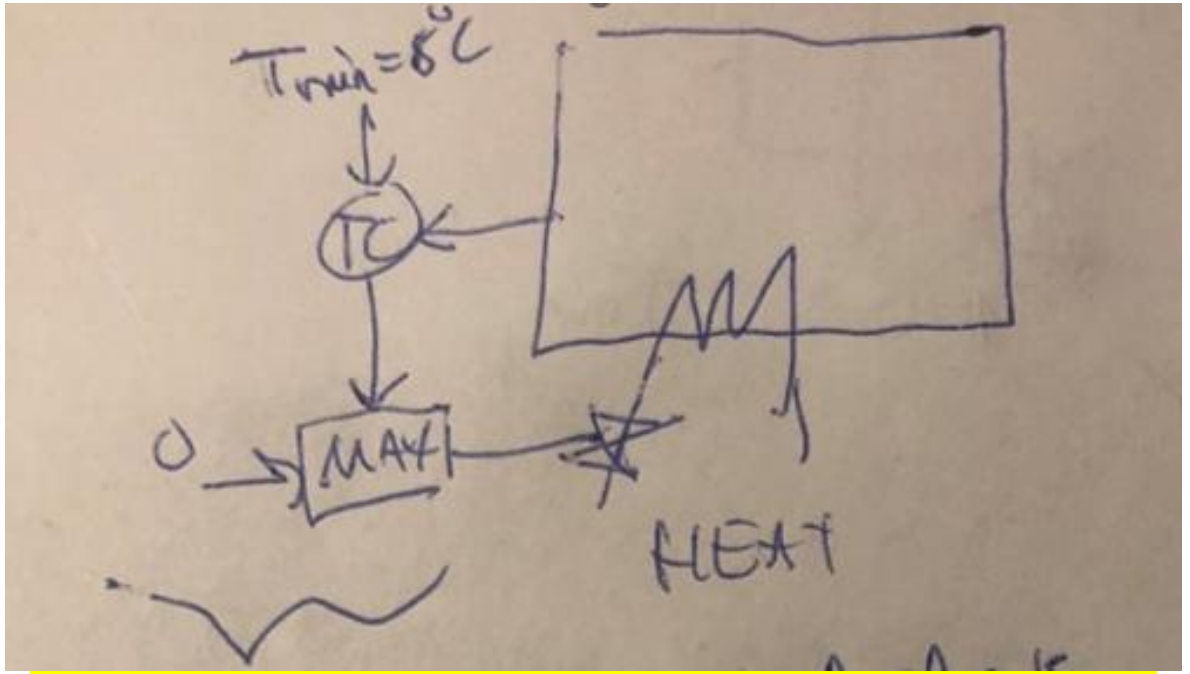
Avoid freezing in cabin

Minimize u (heating), subject to

$$T \geq T_{min}$$
$$u \geq 0$$

Keep CV=T>T_{min} = 8C in cabin in winter by using MV=heating

If it's hot outside (>8C), then the heat will go to zero (MV=Q=0), but this does not matter as the constraint is over-satisfied.



- Actually, no selector required, because MV=z has a «built-in» max-selector at z=0.

Example 3, «simple» MV-CV switching

Anti-surge control (= min-constraint on F)

Minimize u (recycle), subject to

$$F \geq F_{min}$$
$$u = z \geq 0$$

Keep minimum flow F_{min} for pump or compressor using recycle valve.

If the flow F_0 (and thus F) becomes large then the recycle valve will close (MV=0), but this does not matter as the constraint $F \geq F_{min}$ is over-satisfied.

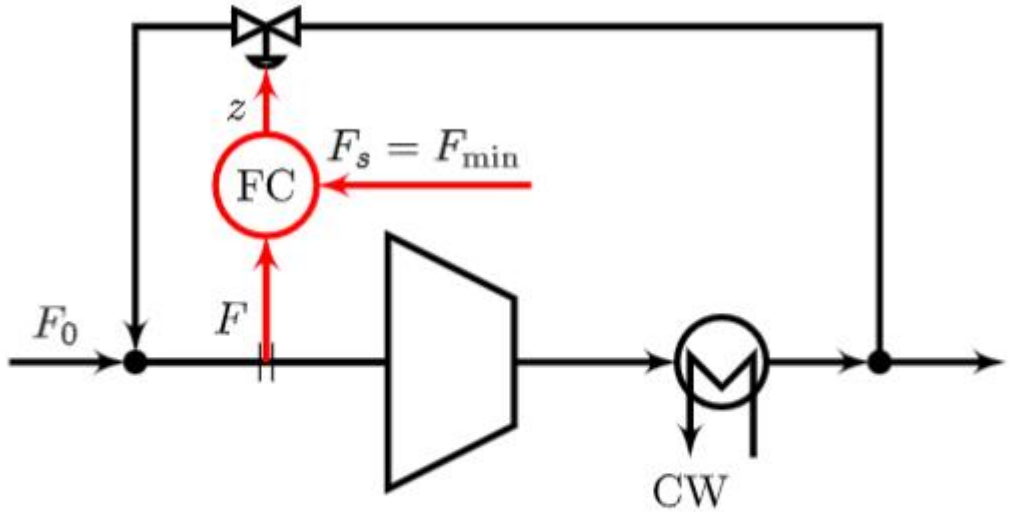


Fig. 32. Flowsheet of anti-surge control of compressor or pump (CW = cooling water). This is an example of simple MV-CV switching: When MV=z (valve position) reaches its minimum constraint ($z = 0$) we can stop controlling CV=F at $F_s = F_{min}$, that is, we do not need to do anything except for adding anti-windup to the controller. Note that

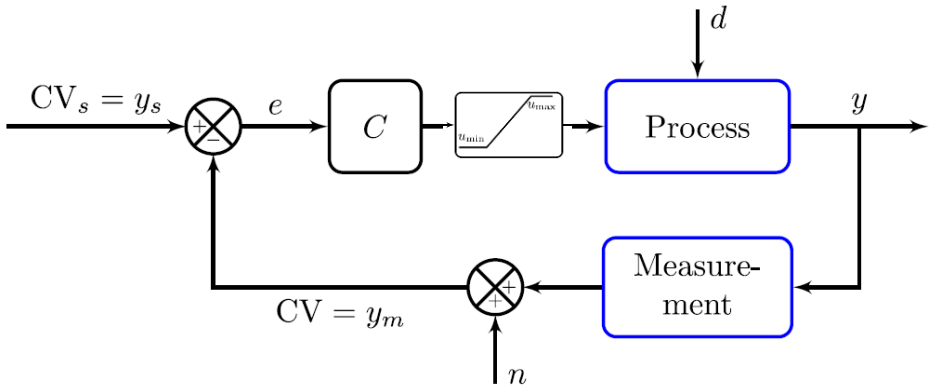
- No selector is required, because MV=z has a «built-in» max-selector at z=0.

Summary: Simple MV-CV switching

- When MV (u) saturates, we can give up the CV (y).
  - Don't need to do anything, except having anti-windup in the controller
- This is because we have followed the

Input saturation rule: "Pair a MV that may saturate with a CV that can be given up (when the MV saturates)"

- Many examples (that it works is not always so obvious!)
  - 1. Driving as fast as possible to the airport
    - u=power $\leq u_{max}$
    - y = speed $\leq y_{max}$
    - $y_s = y_{max}$ = 90 km/h
    - "If we reach max power (u=umax), we must give up controlling y"
  - 2. Heating of cabin in the winter
    - u=power ≥ 0
    - y = temperature $\geq$ 8C
    - $y_s = y_{min}$ = 8C
    - "If we reach min. power (u=0), then it is hot outside - and there is no need to control y"
  - 3. Anti-surge control
    - u =recycle ≥ 0,
    - y = flowrate $\geq y_{min}$
    - $y_s = y_{min}$
    - "If we reach min. recycle (u=0), then the feedrate is larger than y_{min} - and there is no need to control y"



Example: Compressor with max-constraint on F_0
(in addition to the min-constraint on F)

Minimize u (recycle), subject to

$u = z \geq 0$ (satisfied by large u=z, «built in»)

$CV_1 = F \geq F_{min}$ (satisfied by large u=z)

$CV_2 = F_0 \leq F_{0,max}$ (satisfied by large u=z)

- Both CV-constraints are satisfied by a large z
  ⇒ Max-selector for CV-CV
- When we reach MV-constraint (z=0) both CV-constraints are oversatisfied
  ⇒ Simple MV-CV switching

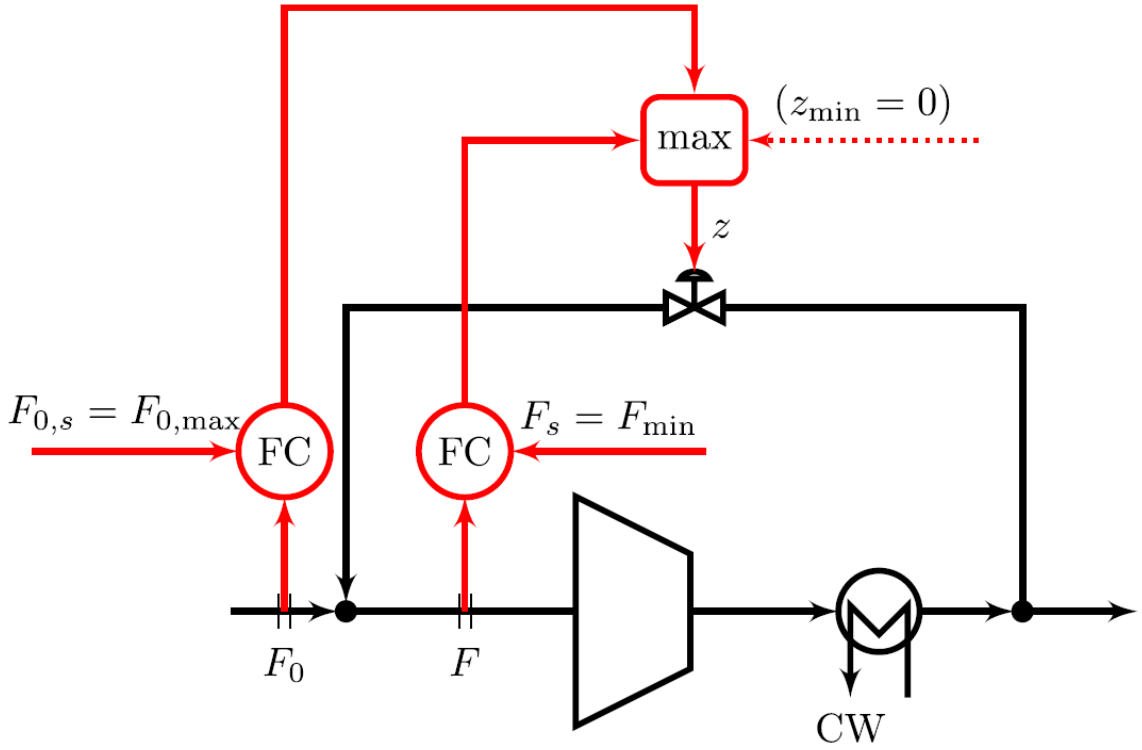
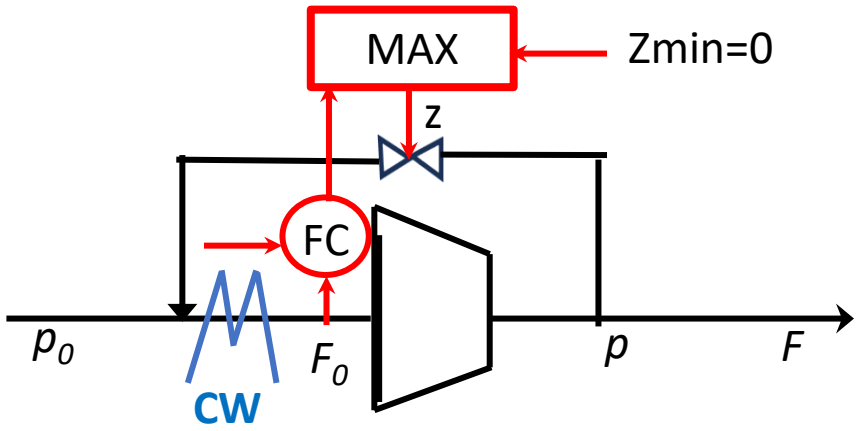


Fig. 33. Anti-surge compressor control with two CV constraints. This is an example of simple MV-CV-CV switching. MV = z, $CV_1 = F$, $CV_2 = F_0$ (all potentially active constraints).

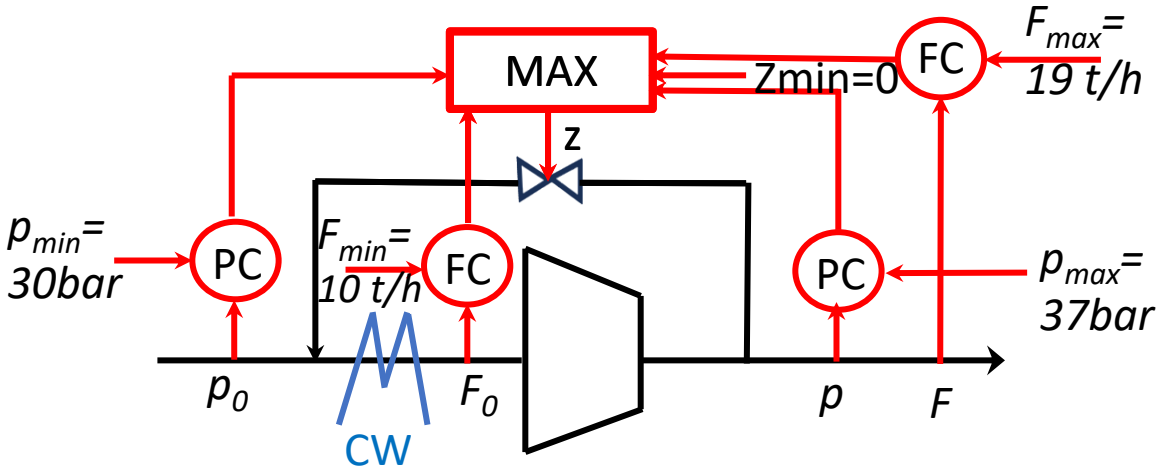
QUIZ Compressor control



Suggest a solution which achieves

- $p < p_{max} = 37$ bar *(max delivery pressure)*
- $P_0 > p_{min} = 30$ bar *(min. suction pressure)*
- $F < F_{max} = 19$ t/h *(max. production rate)*
- $F_0 > F_{min} = 10$ t/h *(min. through compressor to avoid surge)*

SOLUTION



All these 4 constraints are satisfied by a large z
-> MAX-selector

Finally: Complex MV-CV switching

- Didn’t follow input saturation rule
- This requires a repairing of loops
- Need to combine MV-MV switching with CV-CV-switching

- The CV-CV switching always uses a selector
- As usual, there are three alternatives for the MV-MV switching:
  1. Split range control (block /\): Has problems because limits may change
  2. Several controllers with different setpoints (often the best for MV-CV switching)
  3. Valve position control (Gives «long loop» but avoids repairing).

  +4. Shinskey alternative (not covered here)

Furnace control : Cannot give up control of $y_1=T_1$. What to do?

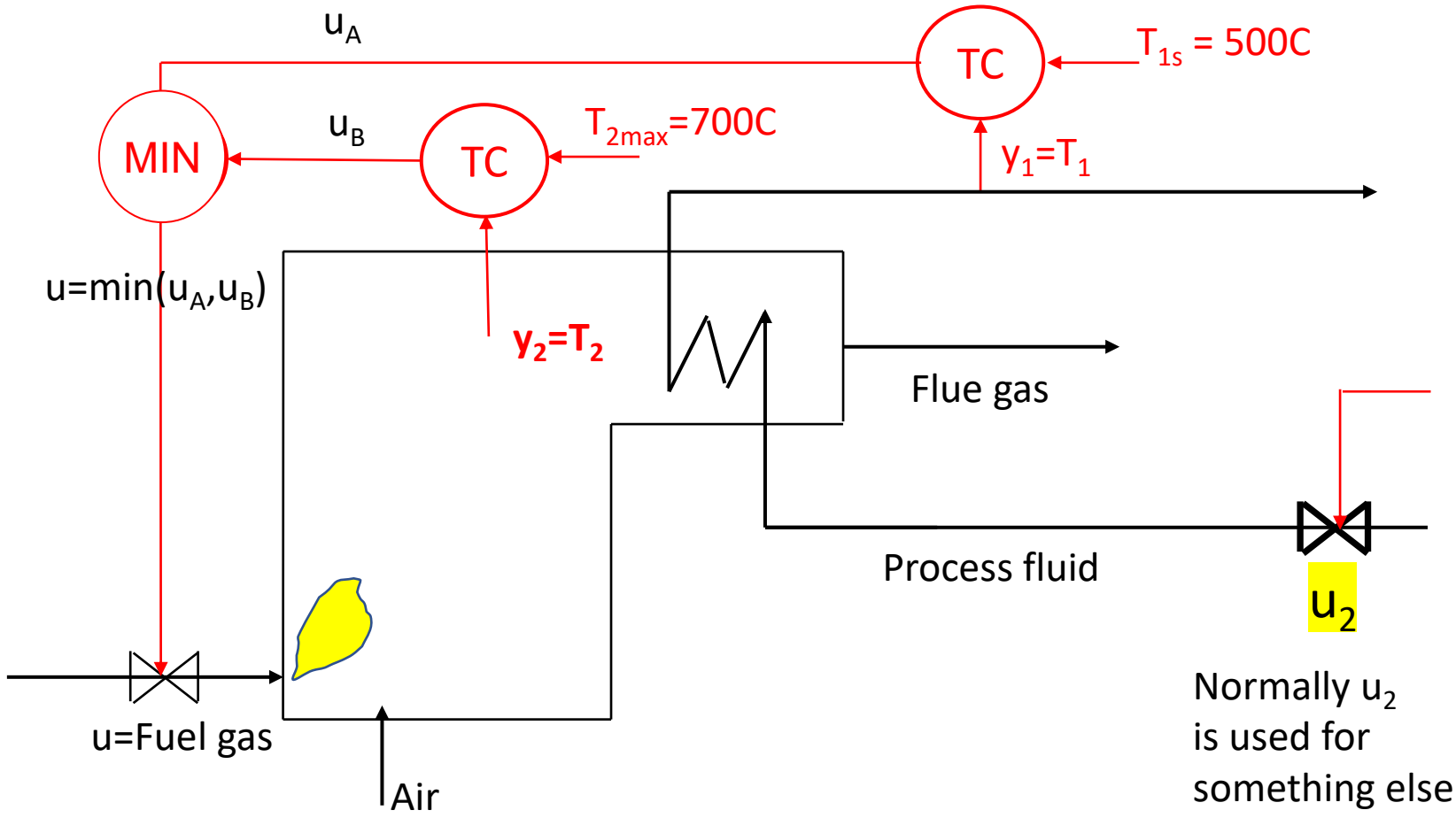
Inputs (MV)

u = Fuel gas flowrate

u_2 = Process flowrate

Output (CV)

y_1 = process temperature T_1 (with desired setpoint)



Cannot give up controlling T_1

Solution: Cut back on process feed (u_2) when T_1 drops too low

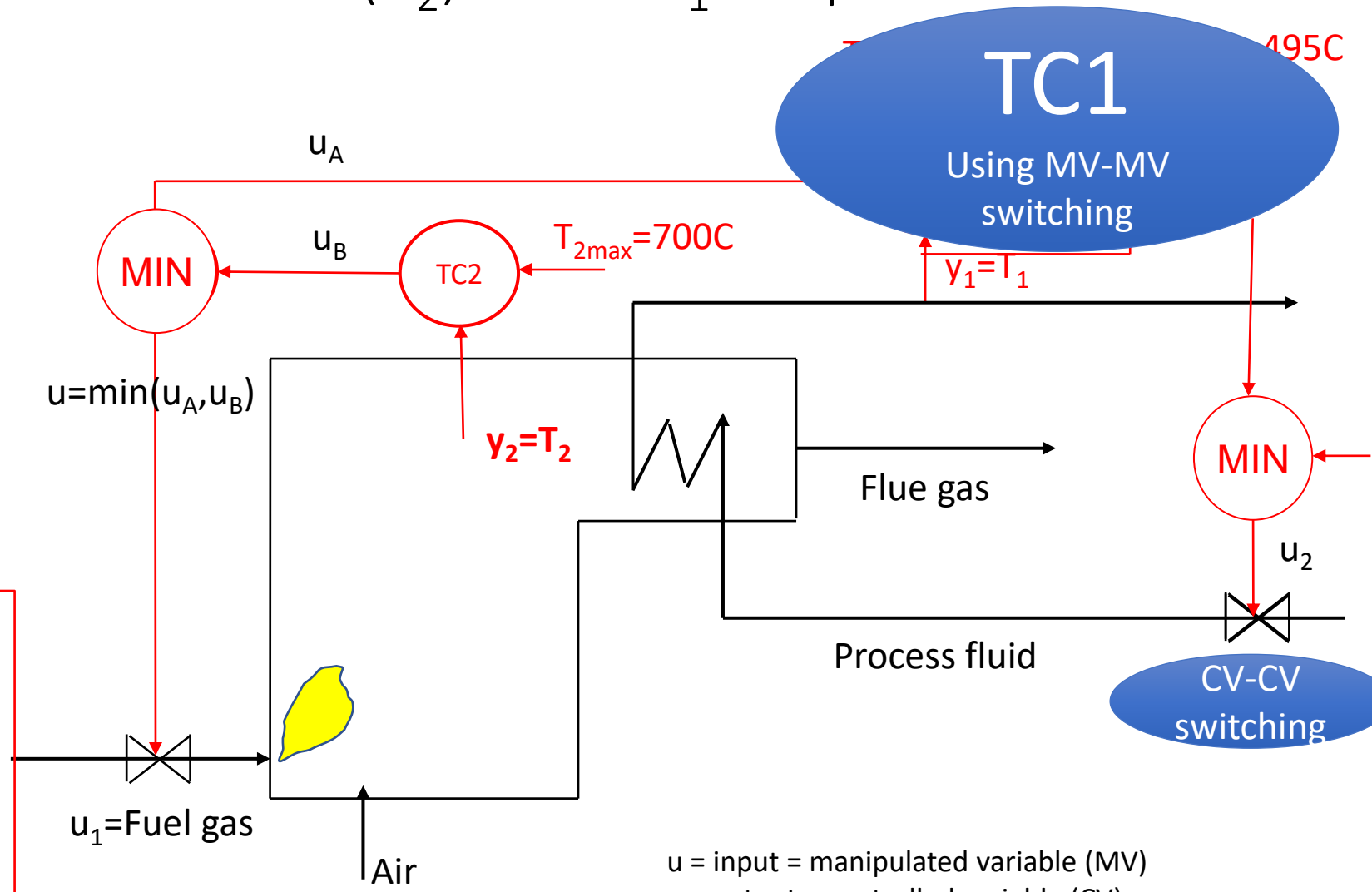
Inputs (MV)

u = Fuel gas flowrate

u_2 = Process flowrate

Output (CV)

y_1 = process temperature (with desired setpoint)



Note: Standard Split Range Control (Alt. 1) is not good here for MV-MV swiitching.

Because could be two reasons for too little fuel

- Fuel is cut back by override (safety)
- Fuel at max,

So don't know limit for MV1 to use in SRC-block.

Also, Alt. 3 (VPC) is not really an alternative as u1 is not saturating but overridden. What we could do instead, is to let u_B from TC2 go to the min-selector for u_2 (with a backoff, e.g., T_{2max}=690C) but this would involve a long loop through TC1 and u_A and is not recommended.

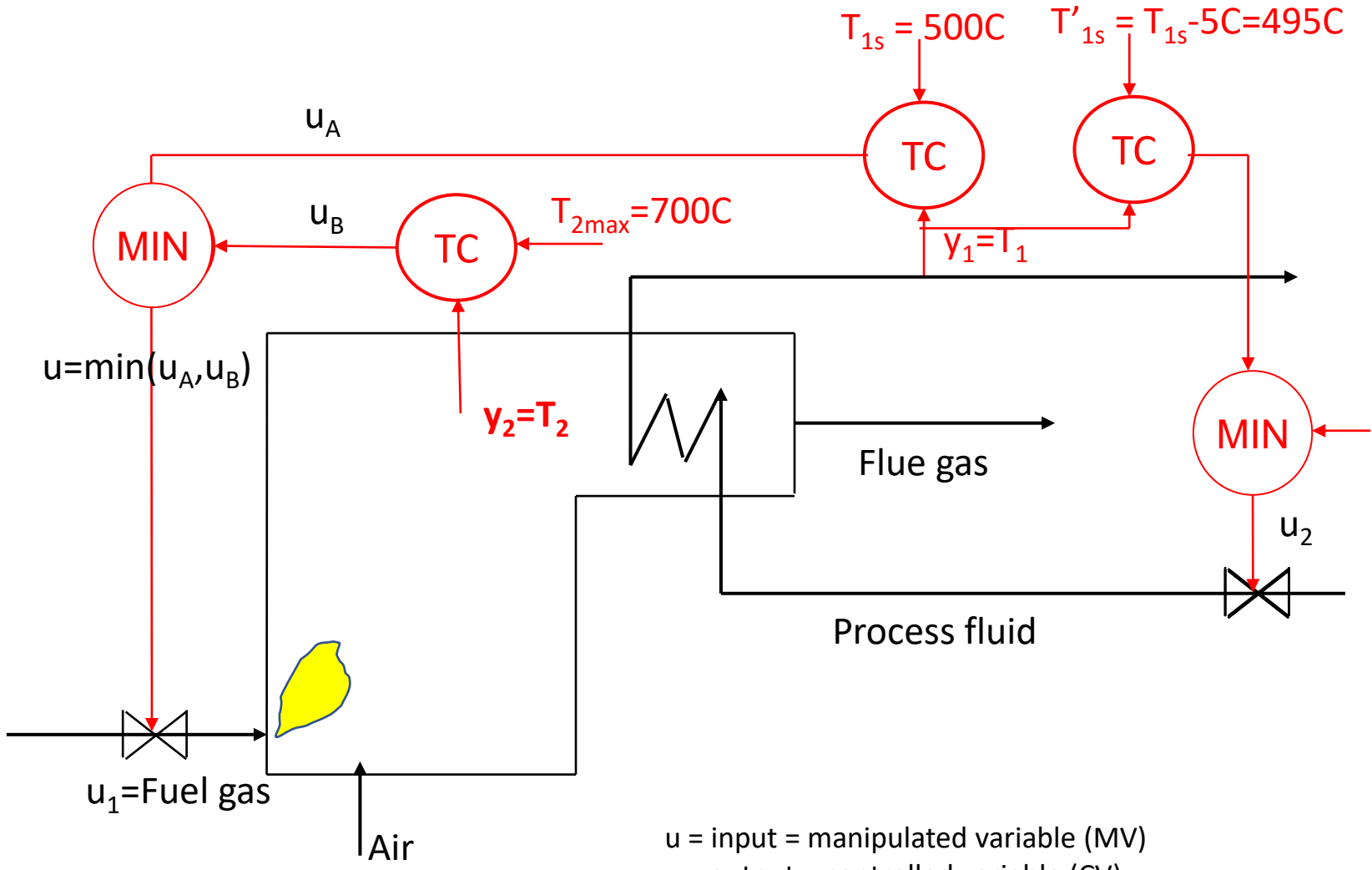
u = input = manipulated variable (MV)

y = output = controlled variable (CV)

Use Alt. 2: Split parallell control (two setpoints)

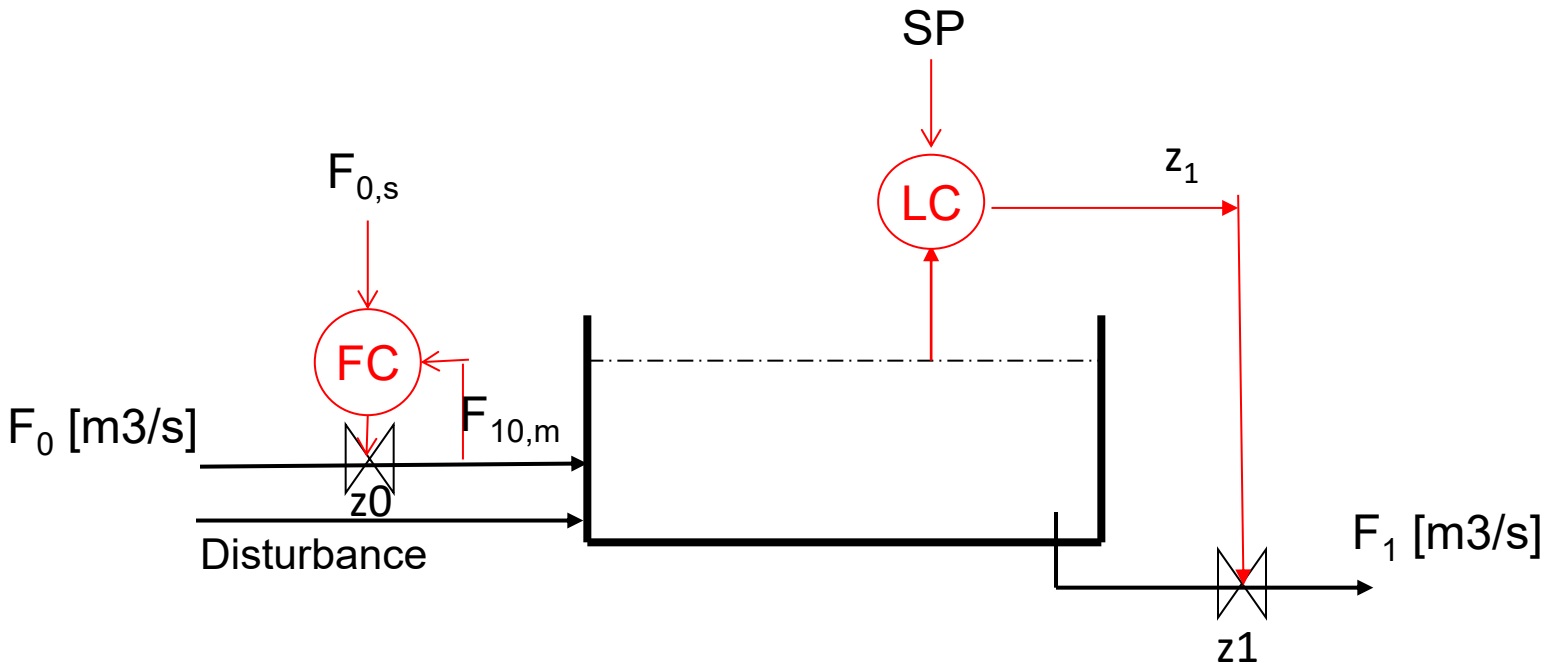
Inputs (MV)
- u = Fuel gas flowrate
- u2 = Process flowrate

Output (CV)
- y_1 = process temperature (with desired setpoint)



u = input = manipulated variable (MV)
y = output = controlled variable (CV)

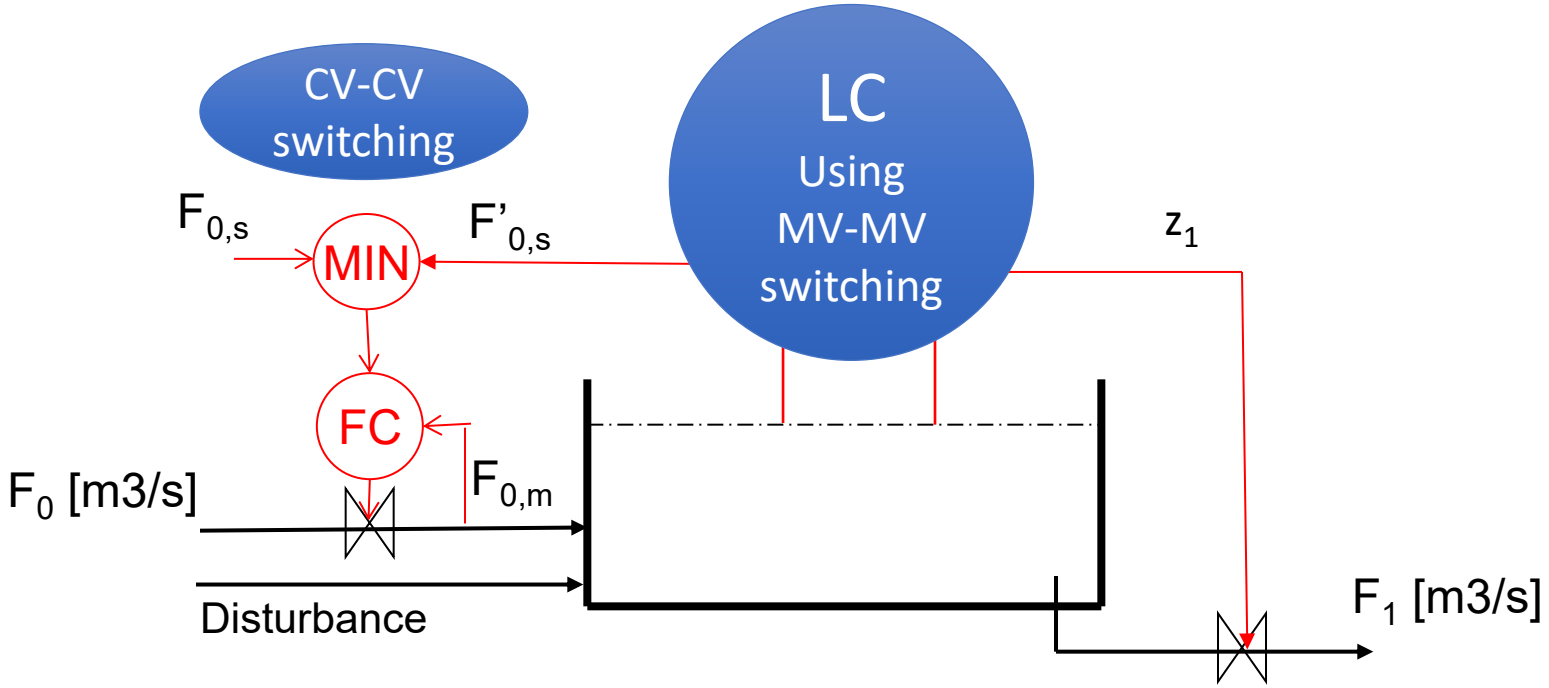
Example : Level control



TPM at feed, so level control is with outflow

Problem: Lose level control if we feed too much so outflow valve saturate at fully open (z1=100%)

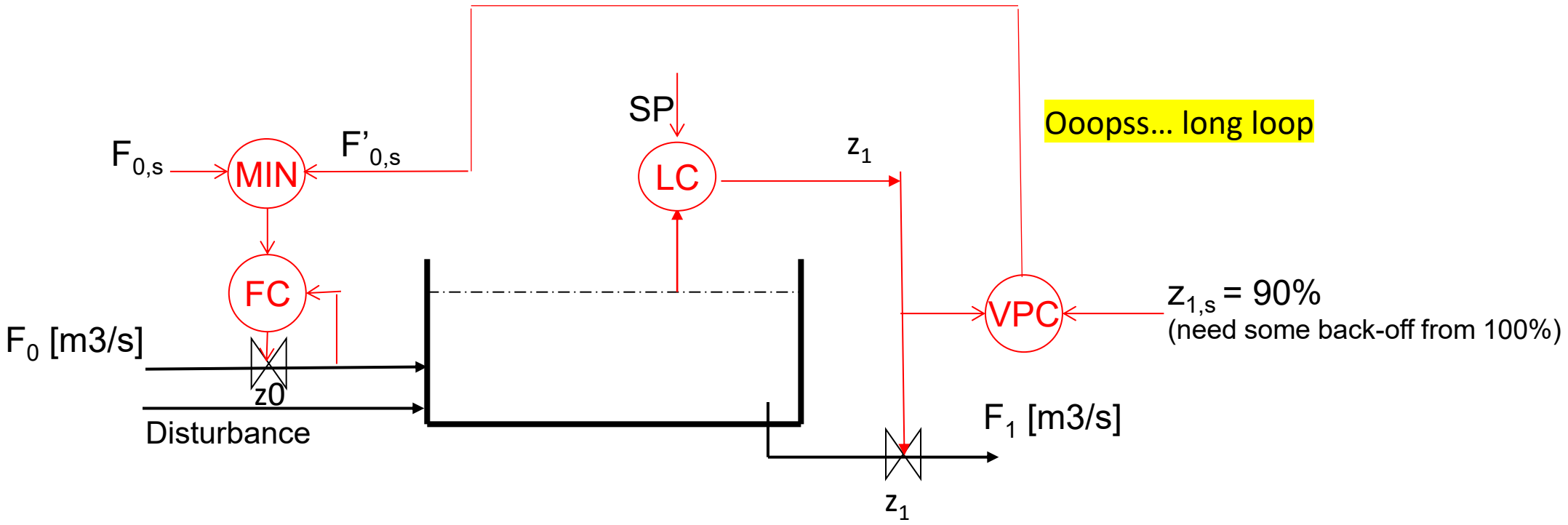
Solution: Complex MV-CV switching (bidirectional inventory control)



Three alternatives for MV-MV switching

1. Split range control (problem since F_{0s} varies).
2. Split parallel control
3. VPC

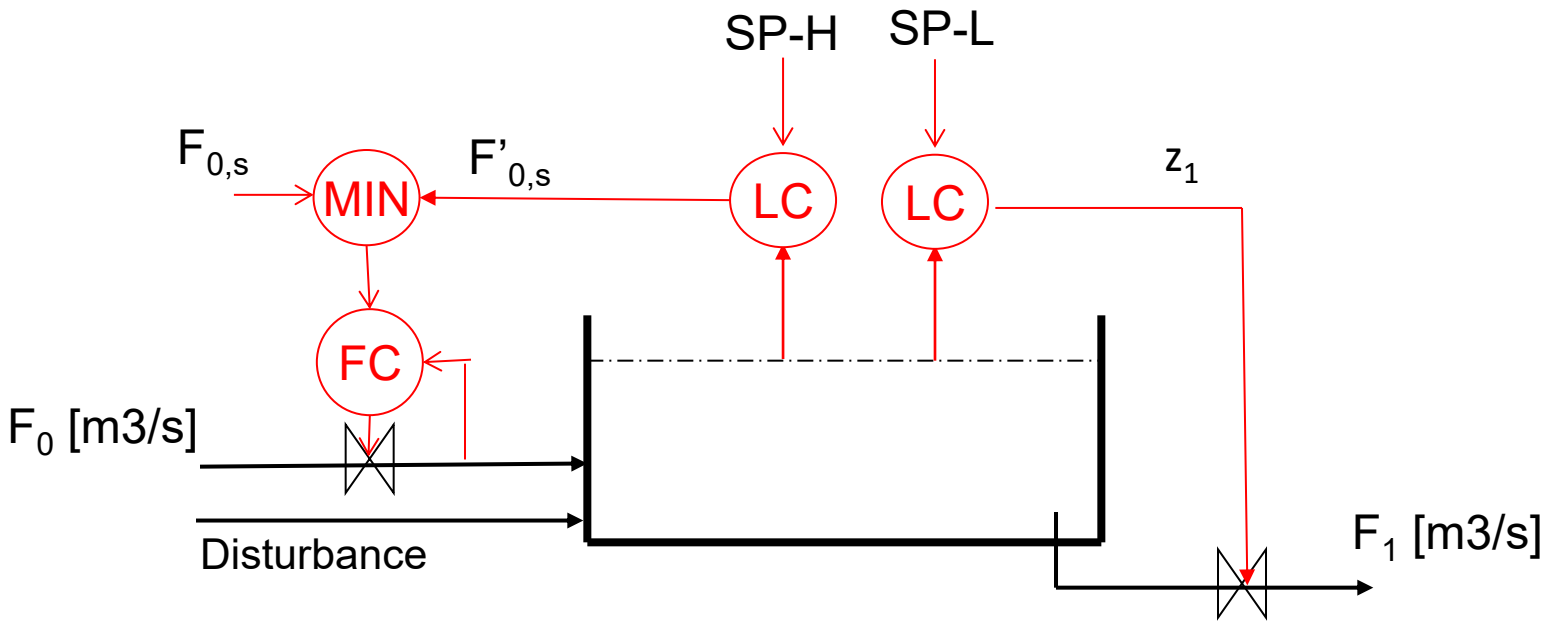
Alt. 3 MV-MV switching: VPC



VPC: “reduce inflow (F_0) if outflow valve (z_1) approaches fully open”

Complex MV-CV switching
Bidirectional inventory control

Alt. 2 MV-MV switching: Split parallel control
(recommended)



SP-L = low level setpoint = 50 % (or 20%)
SP-H = high level setpoint = 60 % (or 80%)

When F1 saturates (100%) we cannot keep SP-L, so level rises until we reach SP-H and F0 takes over

In addition: Use of two setpoints is good for using buffer dynamically!!

Implementing optimal operation by switching

- Most people think
  - You need a detailed nonlinear model and an on-line optimizer (RTO) if you want to optimize the process
  - You need a dynamic model and model predictive control (MPC) if you want to handle constraints
  - The alternative is Machine Learning
- No! In many cases you just need to measure the constraints and use PID control
  - «Conventional advanced regulatory control (ARC)»
- How can this be possible?
  - Because optimal operation is usually at constraints
  - Feedback with PID-controllers can be used to identify and control the active constraints
  - For unconstrained degrees of freedom, one often have «self-optimizing» variables

- This fact is not well known, even to control professors
  - Because most industrial ARC-applications seem *ad hoc*
  - Few systematic design methods exists

- Today ARC and MPC are in parallel universes
  - Both are needed in the control engineer's toolbox

more elements

E8. Anti-windup for the integral mode

Without anti-windup:

$$u(t) = K_c e(t) + K_c \tau_D \frac{de(t)}{dt} + \overbrace{\underbrace{\frac{K_c}{\tau_I}\int_{t_0}^{t} e(t')dt' + u_0}_{\text{bias}=b}}^{u_I} \qquad \text{(C.1)}$$

Appendix C.6.1. Simple anti-windup schemes

Many industrial anti-windup schemes exist. The simplest is to limit u in (C.1) to be within specified bounds (by updating u_0), or to limit the bias $b = u_0 + u_I$ to be within specified bounds (also by updating u_0). These two options have the advantage that one does not need a measurement of the actual applied input value ($\tilde{u}$), and for most loops these simple anti-windup approaches suffice (Smith, 2010) (page 21).

Appendix C.6.2. Anti-windup using external reset

A better and also common anti-windup scheme is "external reset" (e.g., Wade (2004) Smith (2010)) which originates from Shinskey. This scheme is found in most industrial control systems and it uses the "trick" of realizing

Appendix C.6.3. Recommended: Anti-windup with tracking

The "external reset" solution is a special case of the further improved "tracking" scheme in Figure 7 which is recommended by Åström & Hägglund (1988). The tracking scheme (sometimes referred to as the "back-calculation" scheme (Åström & Hägglund, 2006)) has a very useful additional design parameter, namely the tracking time constant τ_T, which tells how fast the controller output u tracks the actual applied value $\tilde{u}$. This makes it possible to handle more general cases in a good way, e.g., switching of CVs. In the simpler "exter-

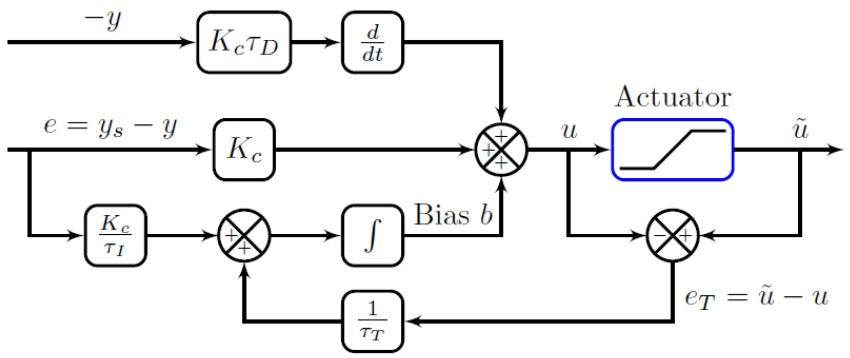


Figure 7: Recommended PID-controller implementation with anti-windup using tracking of the actual controller output ($\tilde{u}$), and without D-action on the setpoint. (Åström & Hägglund, 1988).

With anti-windup using tracking:

$$u(t) = K_c e(t) + K_c \tau_D \frac{de(t)}{dt} + \overbrace{\underbrace{\int_{\hat{t}=t_0}^{t} \left(\frac{K_c}{\tau_I} e(\hat{t}) + \frac{1}{\tau_T} e_T(\hat{t}) \right) d\hat{t} + u_0}_{\text{bias}=b}}^{u_I(t)} \qquad \text{(C.14)}$$

to choose the tracking time equal to the integral time ($\tau_T = \tau_I$). With this value, we get at steady state that the output from the integral part (u_I) is such that the bias b is equal to the constraint value, $b = u_{lim}$. To derive this, note that with

Anti-windup with cascade control

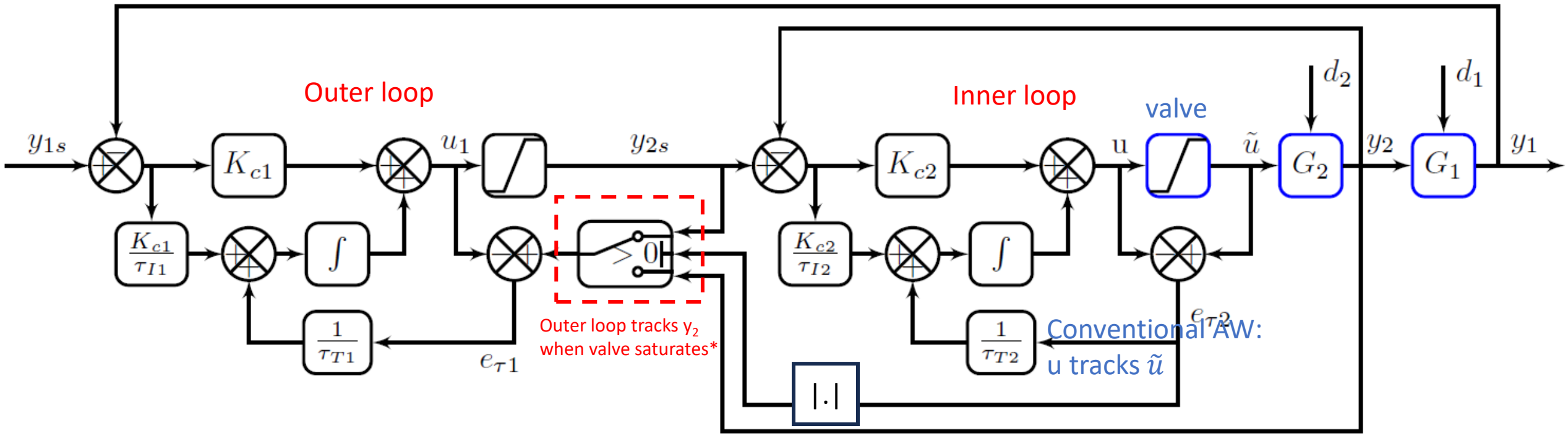


Figure 25: Cascade control with anti windup using the industrial switching approach (Leal et al., 2021).

* This selector makes sure we use anti windup in the outer loop (and track y_2) only when the inner loop (u) is saturating

E9. Two degrees-of-freedom control

- One degree-of freedom control: Controller uses $e=y_s-y$
- Two degrees-of-freedom control: y and y_s used differently.
  - For example, no derivative action on setpoint or beta-factor.
  - More generally, setpoint filter F_s and measurement filter F.

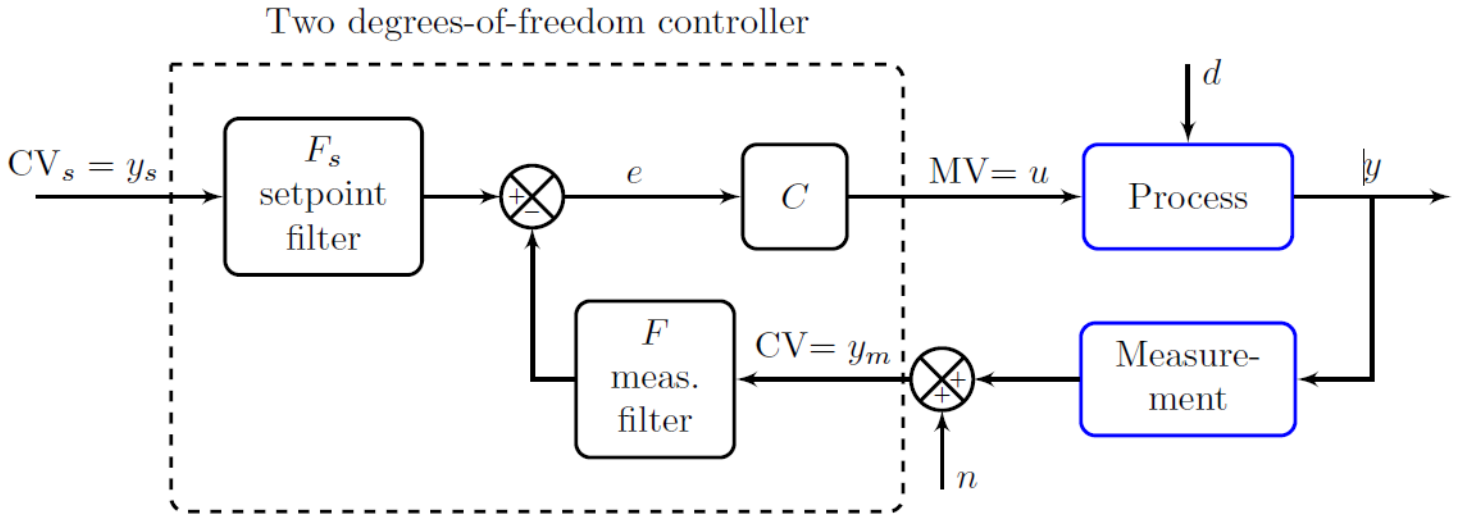


Figure A.41: Two degrees-of-freedom control system with setpoint filter F_s and measurement filter F. All blocks are possibly nonlinear.

Design of setpoint prefilter

- Let T(s) = gc/(1+gc) be the setpoint response with feedback only
- Then choose $F_s(s)$ such that $T_s(s) = F_s(s)*T(s)$ where T_s is the desired setpoint response.
- Very simple!
- It works well as shown for PI-control of integrating process on the next page (in spite of using the time delay approximation $e^{-\theta s} = 1-\theta s$).

Setpoint filter to avoid overshoot with PI-control for integrating process

Consider a first-order process with a large tau (tau=infinity will be integrating).

G(s) = k exp(-theta*s)/(tau*s+1)

It is be controlled by a SIMC PI-controller

c(s) = Kc (taul*s+1)/(taul*s) with Kc = 1/k' (1/(tauc+theta))

where k'= k/tau and tauc is the closed-loop time constant.

Assume that we want to get the same setpoint response as with the original IMC-controller, which has taul = tau (this response is nice, see red curve next page).
For the IMC-controller, we get gc = e-theta*s/(tauc+theta)s. We evaluate T=gc/1+gc, where we for the term 1+gc we use the approximation e-theta*s=1-theta*s. This gives

TIMC=y/ys = e-theta*s / (tauc*s+1)

Let us now use a different value for taul (for example, taul=4*(tauc+theta) according to SIMC-rule), which gives an overshoot in the setpoint response (red curve next page).
We evaluate T=gc/(1+gc) and for the term 1+gc we again make use of the approximation e-theta*s=1-theta*s. We find that closed-loop response is (without a prefilter)

T=y/ys'=gc/(1+gc) = (taul*s + 1) e-theta*s / x(s)

where

x(s) = taul*tauc*s^2 + [(taul/tau)*(theta+tauc) + taul - theta]*s + 1

For an integrating process (with taul=infinity) we get x(s) = taul*tauc*s^2 + [taul - theta]*s + 1.
With a prefilter fs(s), the setpoint response is Tf= fs(s) * T. So to make (the approximation of) Tf equal to TIMC we must select

fs(s) = x(s) / (tauc*s+1)*(taul*s+1)

Note that we for the case taul=tau get x(s)=(tauc*s+1)(taul*s+1) and fs(s)=1 (as excpected).

Example 1. For the almost-integrating process in Figure 3 (tau=30, theta=1) in the SIMC-paper, we get for the SIMC-tunings (taul=8, tauc=theta=1)

x(s) = 8s^2 + 7.53 s + 1 = (6.25s+1)(1.28s+1)

So the setpoint filter is

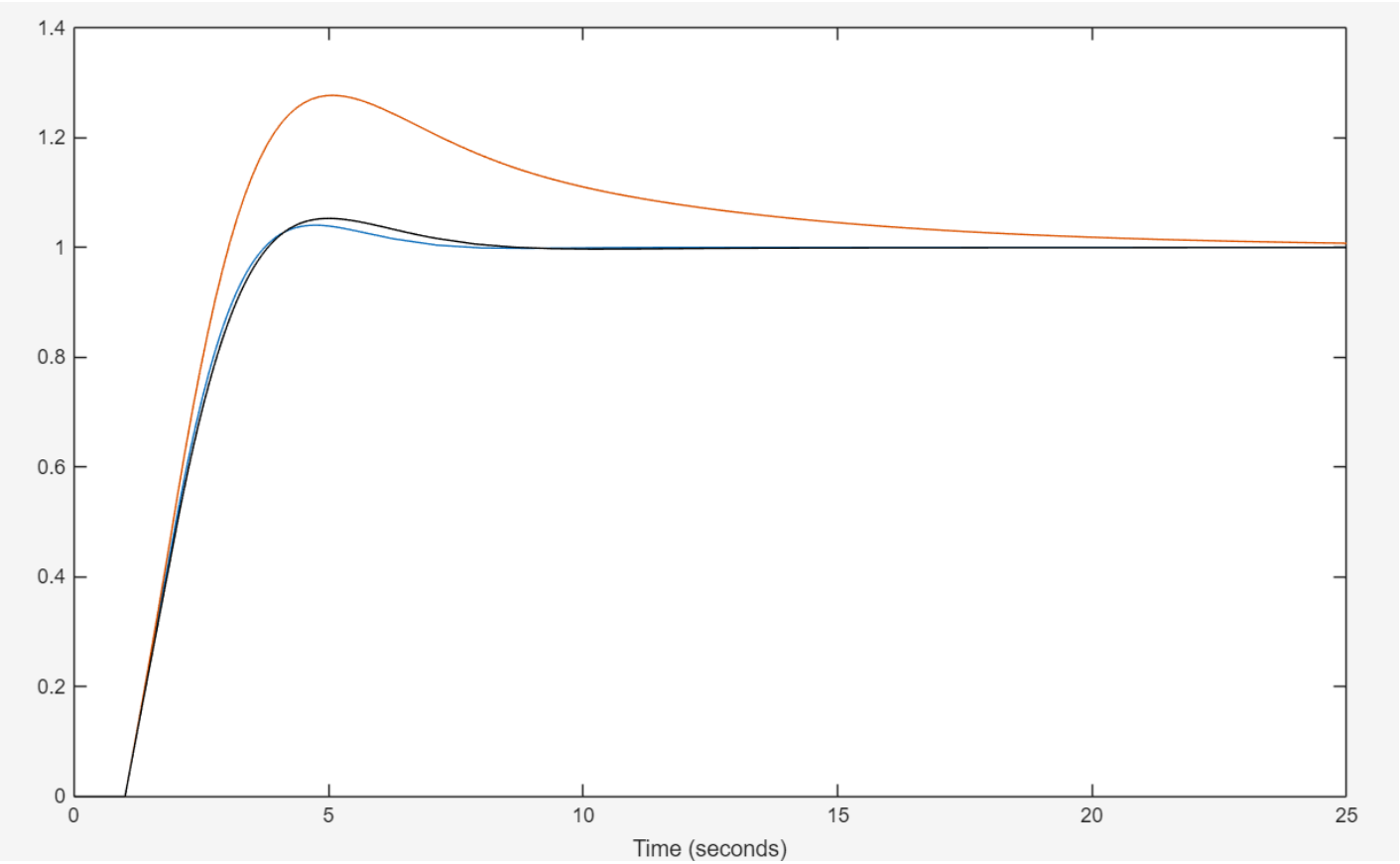
fs(s) = (6.25s+1)(1.28s+1)/(8s+1)(s+1) = (8s^2+7.53s+1)/(8s^2+9s+1)

This prefilter makes the setpoint response for taul=8 almost identical to the one shown with taul=30 (Fig.3). (I checked with Matlab and it works like a charm!).

Example 2. The next page shows the response for an integrating process. Notice that the filter Fs(s) works great, as the blue and black curves are almost the same-

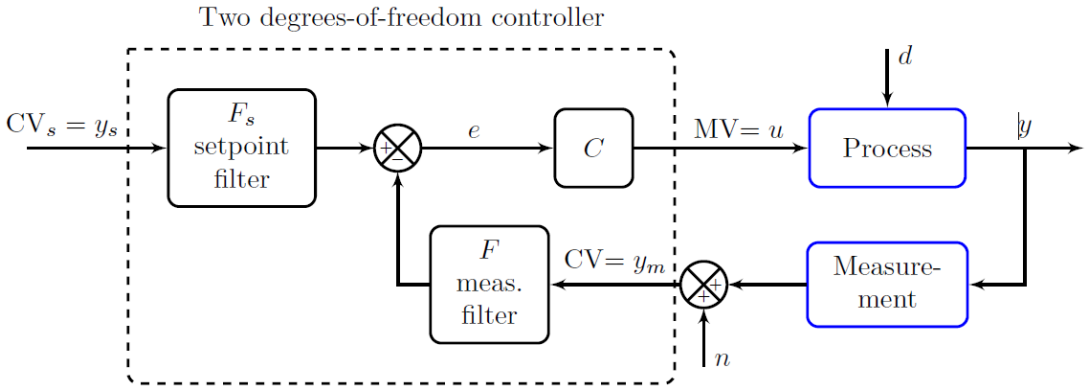
https://skoge.folk.ntnu.no/publications/2003/tuningPID/ see comment 13 (from 2004, but I think unpublished)

Example. Integrating process, g(s)=exp(-s)/s. Setpoint response with PI-control with taui=infty (blue), taui=8 (SIMC, red) and taui=8 with prefilter (black)




```
s=tf('s')
ks=1, tau=9999, theta=1;
G = ks*exp(-theta*s)/(s+1/tau)
tauc=theta
Kc=1/(ks*(tauc+theta)) % SIMC
taui1=tau
taui2=4*(tauc+theta)
C1 = Kc*(1+ 1/(taui1*s))
L1 = G*C1
T1=L1/(1+L1)
C2 = Kc*(1+ 1/(taui2*s))
L2 = G*C2
T2=L2/(1+L2)
tauI=taui2
Fs = (tauI*tauc*s^2 + [(tauI/tau)*(theta+tauc) + tauI - theta]*s + 1)/ ((tauc*s+1)*(tauI*s+1))
Tf2= Fs*T2
figure(1), step(T1,T2,Tf2,'black')
```

Measurement filter F(s)



- Very common, especially with noisy measurements
  - Used also alone (without F_s)
  - Most common: First-order filter

$$F(s) = \frac{1}{\tau_F s + 1}$$

Recommended: $\tau_F \leq \frac{\tau_c}{2}$ (preferably smaller, typically $\tau_F = 0.1\ \tau_c$)

- τ_c: Closed-loop time constant (SIMC)

Here τ_F is the measurement filter time constant, and the inverse ($\omega_F = 1/\tau_F$) is known as the cutoff frequency. However, one should be careful about selecting a too large filter time constant τ_F as it acts as a effective delay as seen from the controller C.

King (2011) (page xii) writes in this respect: "Many engineers are guilty of installing excessive filtering to deal with noisy measurements. Often implemented only to make trends look better they introduce additional lag and can have a detrimental impact on controller performance." To reduce the effective delay (lag) introduced by filtering, Sigifredo Nino (personal email communication, 30 March 2023), who has extensive industrial experience, suggests using a second-order Butterworth filter,

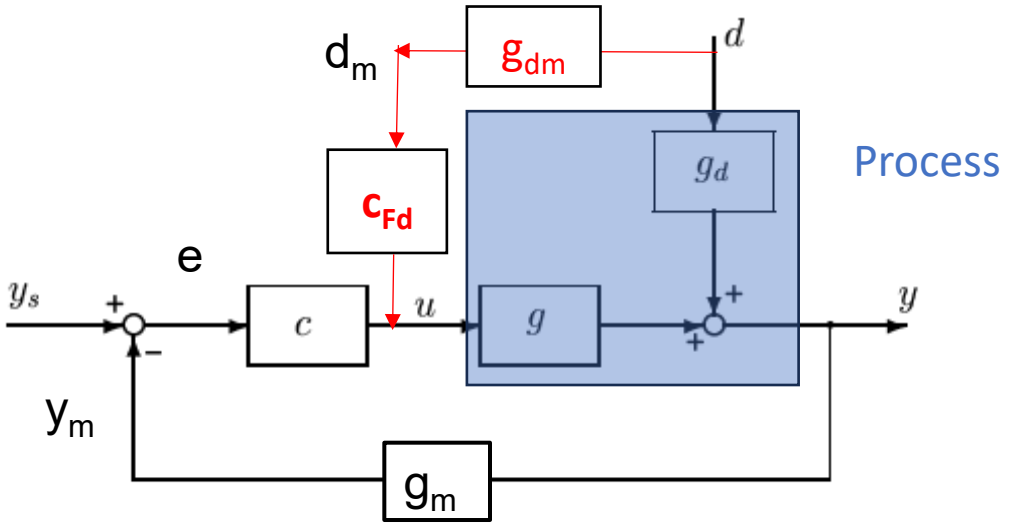
$$F(s) = \frac{1}{\tau_F^2 s^2 + 1.414\tau_F s + 1} \qquad \text{(A.4)}$$

E10. Gain scheduling

- Very popular for PID within EE and ME, e.g., airplanes, automotive.
- Controller (PID) tunings change as a given function of the scheduling variable, e.g.,
  - disturbance d
  - process input u
  - process output y
  - setpoint y_s
  - control error $e=y_s-y$

A little on feedforward control (E11)

Feedforward control: Measure disturbance (d)



Block diagram of feedforward control

c = Feedback controller

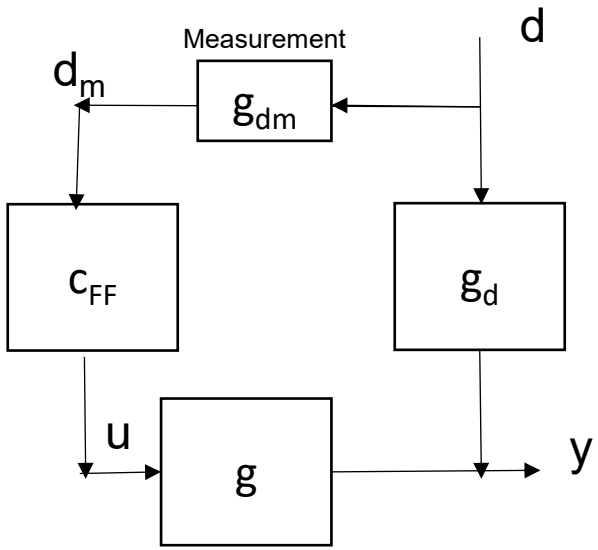
c_{Fd} = Feedforward controller.

Ideal, inverts process g: $\boldsymbol{c_{Fd} = g^{-1} g_d\, g_{dm}^{-1}}$

Usually: Add feedforward when feedback alone is not good enough, for example, because of measurement delay in g_m

Details Feedforward control

- Model: $y = g\ u + g_d\ d$
- Measured disturbance: $d_m = g_{dm}\ d$
- Feedforward controller: $u = c_{FF}\ d_m$
- Get $y = (g\ c_{FF}\ g_{dm} + g_d)\ d$
- Ideal feedforward:
  - $y = 0 \Rightarrow c_{FF,\,ideal} = -g^{-1}\ g_d\ g_{dm}^{-1} = -\frac{g_d}{g_{dm}\ g}$
- In practice: $c_{FF}(s)$ must be realizable
  - Order pole polynomial ≥ order zero polynomial
  - No prediction allowed (θ cannot be negative)
  - Must avoid that c_{FF} has too high gain to avoid (to avoid aggressive input changes)
- Common simplification: $c_{FF} = k$ (static gain)
- General. Approximate $c_{FF,\,ideal}$ as :



$$c_{FF}(s) = k\frac{(T_1 s + 1)\ \ldots}{(\tau_1 s + 1)(\tau_2 s + 1)\ \ldots} e^{-\theta s}$$

where we must have at least as many τ's as T's

Example feedforward

$$y = gu + g_{d1}d_1 + g_{d2}d_2$$

Feedforward control: $u = c_{FF}d_m$
Ideal feedforward controller: $c_{FF} = -\frac{g_d}{g_{dm}g}$

Example (assume perfect measurements, $g_{dm} = 1$):
$g(s) = \frac{e^{-s}}{s(20s+1)}$
$g_{d1}(s) = \frac{1}{s}$
$g_{d2}(s) = \frac{e^{-s}}{s(20s+1)}$

Disturbance 1:
Ideal: $c_{FF1} = -(20s+1)e^s$ (has prediction + has more zeros than poles)
Actual: $c_{FF1} = -1 \cdot \frac{20s+1}{\tau s+1}$ where τ is tuning parameter
(smaller τ gives better control, but requires more input usage).

Comment: In the simulation we use $\tau = 2$ which is quite aggressive; $\tau = 20$ would give $c_{FF1} = -1$.

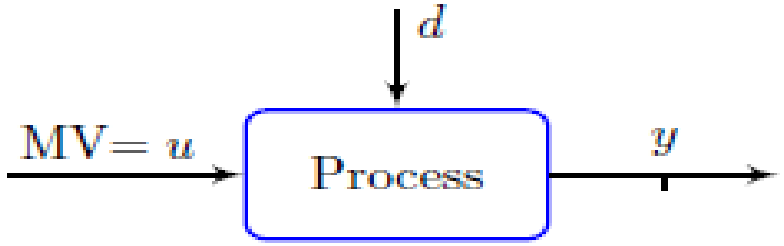
Disturbance 2:
Ideal: $c_{FF2} = -1$
Actual: $c_{FF2} = -1$

«Chicken factor»

Comment: In practice, one often sets the feedforward gain about 80% of the theoretical, that is, $c_{FF2} = -0.8$. This is to avoid that the feedforward controller overreacts, which may confuse the operators. It also makes the feedforward action more robust.

What is best? Feedback or feedforward?

Example: Feedback vs. feedforward for setpoint control of uncertain process



y = G(s) u

$$G(s) = \frac{k}{\tau s + 1}, \quad k = 3,\ \tau = 6 \qquad \text{(B.2)}$$

Desired response : $y = \frac{1}{\tau_c s + 1} y_s = \frac{1}{4s + 1} y_s$

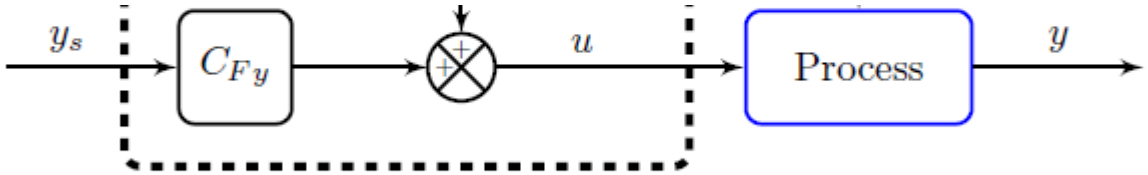


Figure A.42: Block diagram of feedforward control system with linear combination of feedforward from measured disturbance (d) and setpoint (y_s) (E14).

Feedforward solution. We use feedforward from the setpoint (Fig. A.42):

$$u = C_{Fy}(s) y_s$$

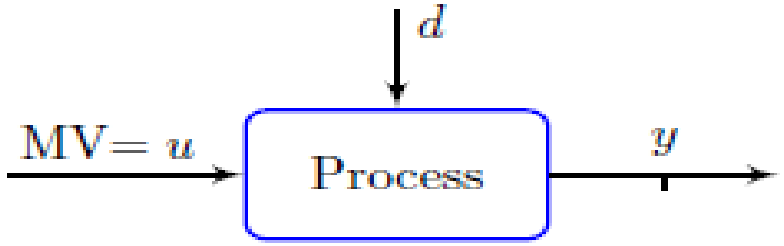
where we choose

$$C_{Fy}(s) = \frac{1}{\tau_c s + 1} G(s)^{-1} = \frac{1}{k} \frac{\tau s + 1}{\tau_c s + 1} = \frac{1}{3} \frac{6s + 1}{4s + 1} \qquad \text{(B.3)}$$

The output response becomes as desired,

$$y = \frac{1}{4s + 1} y_s \qquad \text{(B.4)}$$

Example: Feedback vs. feedforward for setpoint control of uncertain process



y = G(s) u

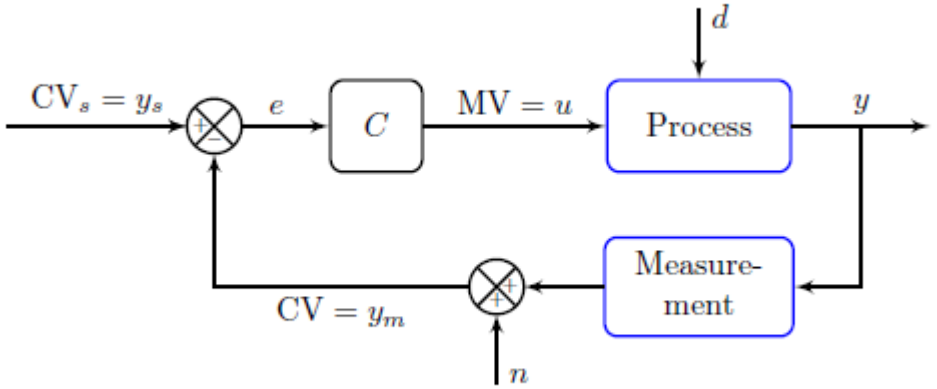


Figure 3: Block diagram of common "one degree-of-freedom" negative feedback control system.

$$G(s) = \frac{k}{\tau s + 1}, \quad k = 3,\ \tau = 6 \qquad \text{(B.2)}$$

Desired response : $y = \frac{1}{\tau_c s + 1} y_s = \frac{1}{4s + 1} y_s$

Feedback solution. We use a one degree-of-freedom feedback controller (Fig. 3) acting on the error signal $e = y_s - y$:

$$u = C(s)(y_s - y)$$

We choose a PI-controller with $K_c = 0.5$ and $\tau_I = \tau = 6$ (using the SIMC PI-rule with $\tau_c = 4$, see Appendix C.2):

$$C(s) = K_c \left(1 + \frac{1}{\tau_I s}\right) = 0.5 \frac{6s + 1}{6s} \qquad \text{(B.5)}$$

Note that we have selected $\tau_I = \tau = 6$, which implies that the zero dynamics in the PI-controller C, cancel the pole dynamics of the process G. The closed-loop response becomes as desired:

$$y = \frac{1}{\tau_c s + 1} y_s = \frac{1}{4s + 1} y_s \qquad \text{(B.6)}$$

Proof. $y = T(s) y_s$ where $T = L/(1 + L)$ and $L = GC = kK_c/(\tau_I s) = 0.25/s$. So $T = \frac{0.25/s}{1 + 0.25/s} = \frac{1}{4s+1}$.

Thus, we have two fundamentally different solutions that give the same nominal response, both in terms of the process input $u(t)$ (not shown) and the process output $y(t)$ (black solid curve in Fig. B.43).

- But what happens if the process changes?
  - Consider a gain change so that the model is wrong
    - Process gain from k=3 to k'=4.5

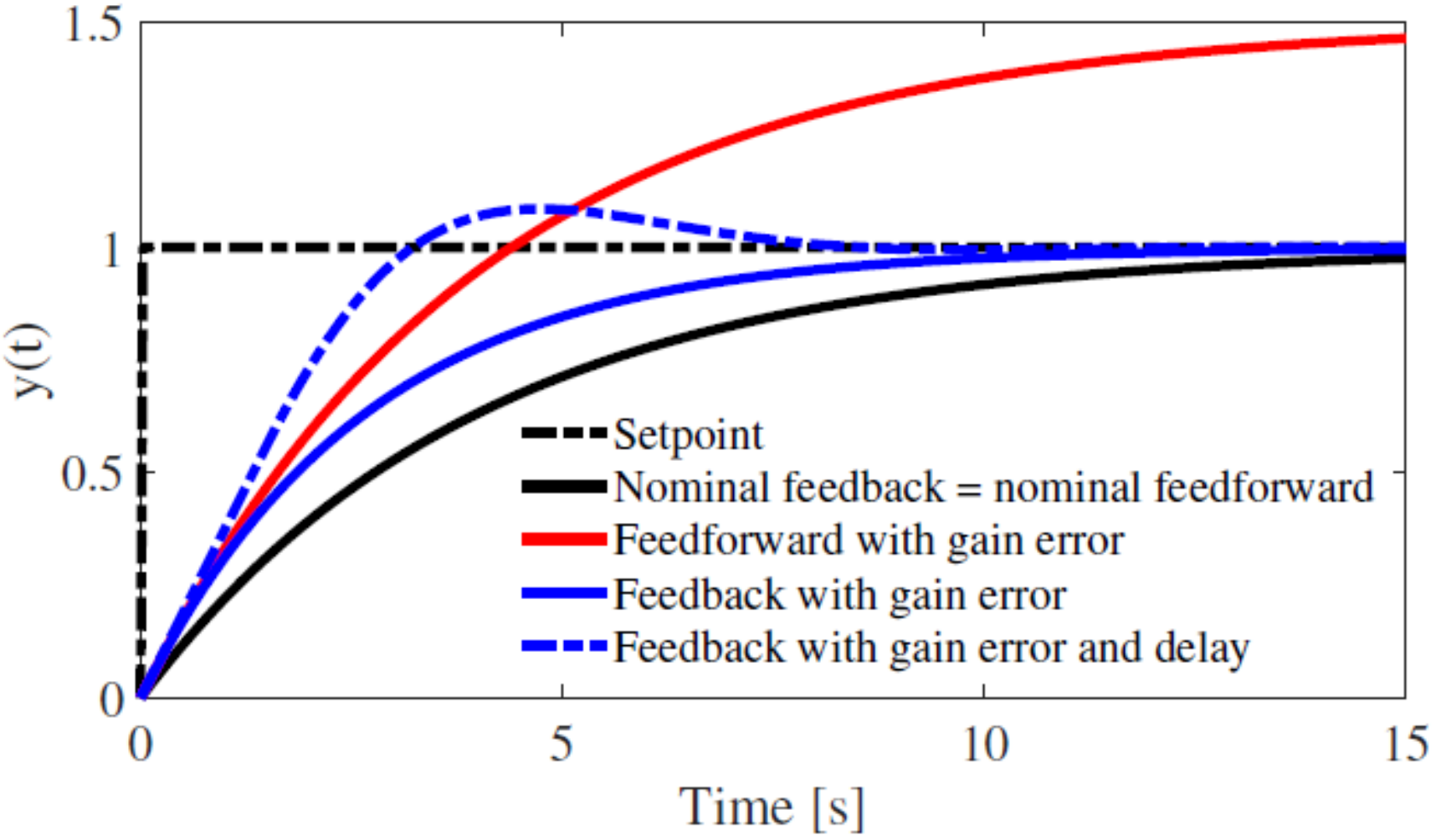
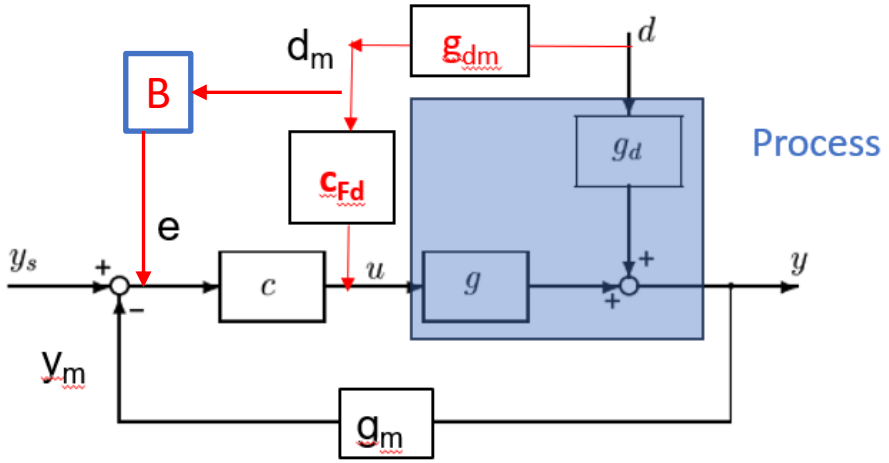


Figure B.43: Setpoint response for process (B.2) demonstrating the advantage of feedback control for handling model error.

Gain error (feedback and feedforward): From k=3 to k'=4.5

Time delay (feedback): From $\theta = 0$ *to* $\theta = 1.5$

Disturbances: Avoid fighting of feedforward and feedback (with B)



We want to choose B such that C_{fd} (FF) and c (FB) can be designed independently!!

- Problem: If feedforward is not perfect (typically because of a delay in g), feedback may try to correct for temporary deviations in y (which feedforward will handle, but it needs a little time).
- To avoid this fighting between feedforward (c_{Fd}) and feedback (c), we want the transfer function (with feedforward included) from d to the feedback controller input (e) to be zero*. So we want
- B*gdm*d – gm*(gd+g*cFd*gdm)*d= 0

Here gm includes a possible measurement filter F. We get

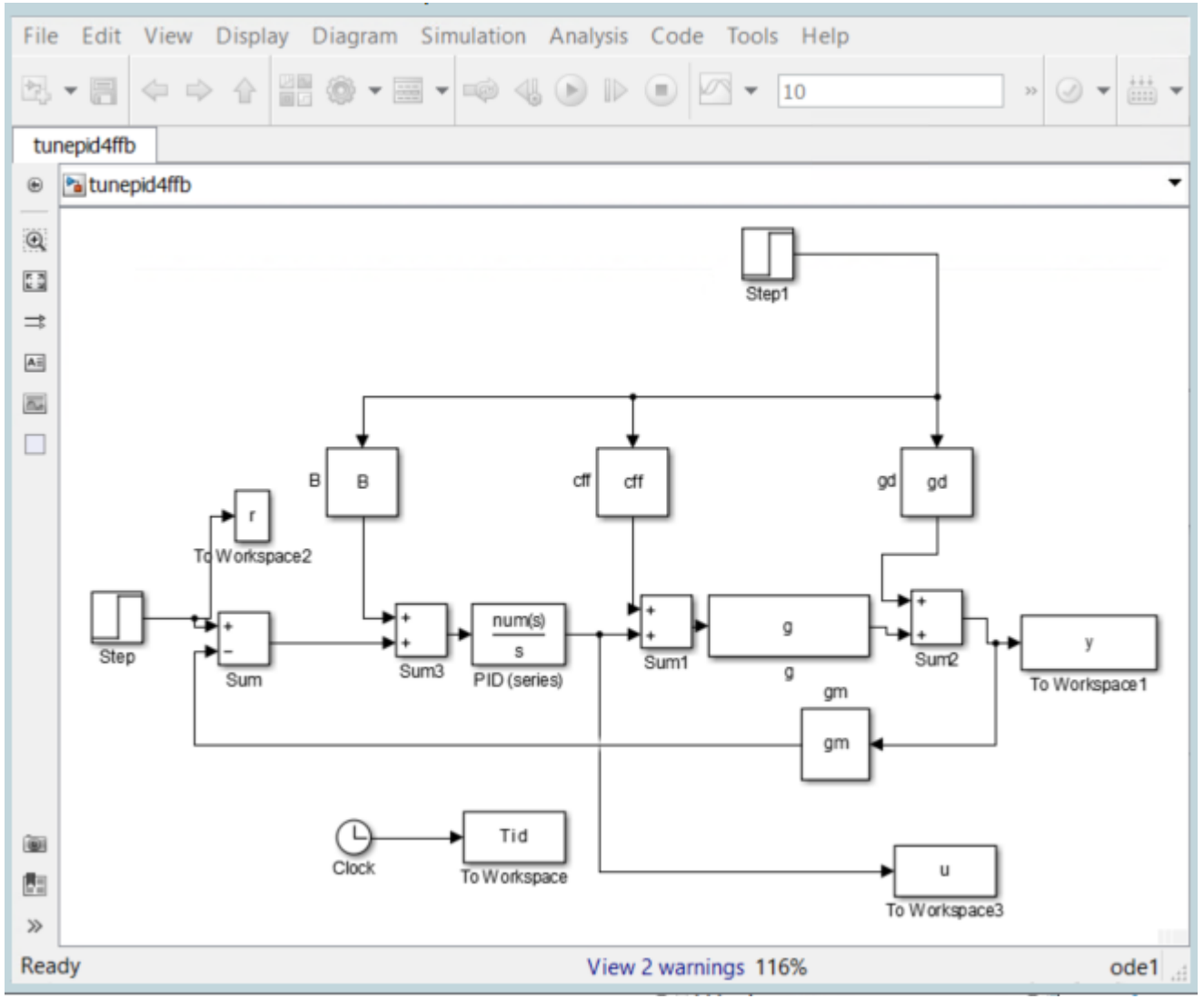
$$B = (g_d+g_{dm}*c_{Fd}*g)*g_m / g_{dm}.$$

This is usually realizable unless gdm has a large delay.
Note that $(g_d+g_{dm}*c_{Fd}*g)$ is the expected response from d to y with feedforward.
With perfect feedforward it will be 0 and we get B=0.

* This idea was originally proposed by Lang and Ham (1955) for the case with combined feedback and feedforward from setpoints. The paper is referred to from D'azzo and Houpis in their 2nd edition from 1966, but not in the third from 1988. Åstrøm and Hägglund also discuss this structure in great detail in their PID-book from 2006. Also see Guzman and Hägglund (2021) they use H=B) who refer to Brosilow and Joseph (2002).

Lang, G., and J. M. Ham. "Conditional feedback systems-A new approach to feedback control." *Transactions of the American Institute of Electrical Engineers, Part II: Applications and Industry* 74.3 (1955): 152-161.

- g=exp(-s), gd=1, gm=1
- Cff=-1
- Pure I-controller with Ki=0.5 (SIMC)
- B = (cff*g*gdm + gd)*gm = 1 - exp(-s)



E11. Simple static estimators

- Inferential element
- Soft sensor
  - Linear: May use SVD (PLS)
  - Nonlinear: May use neural networks

E12. Linear decoupling

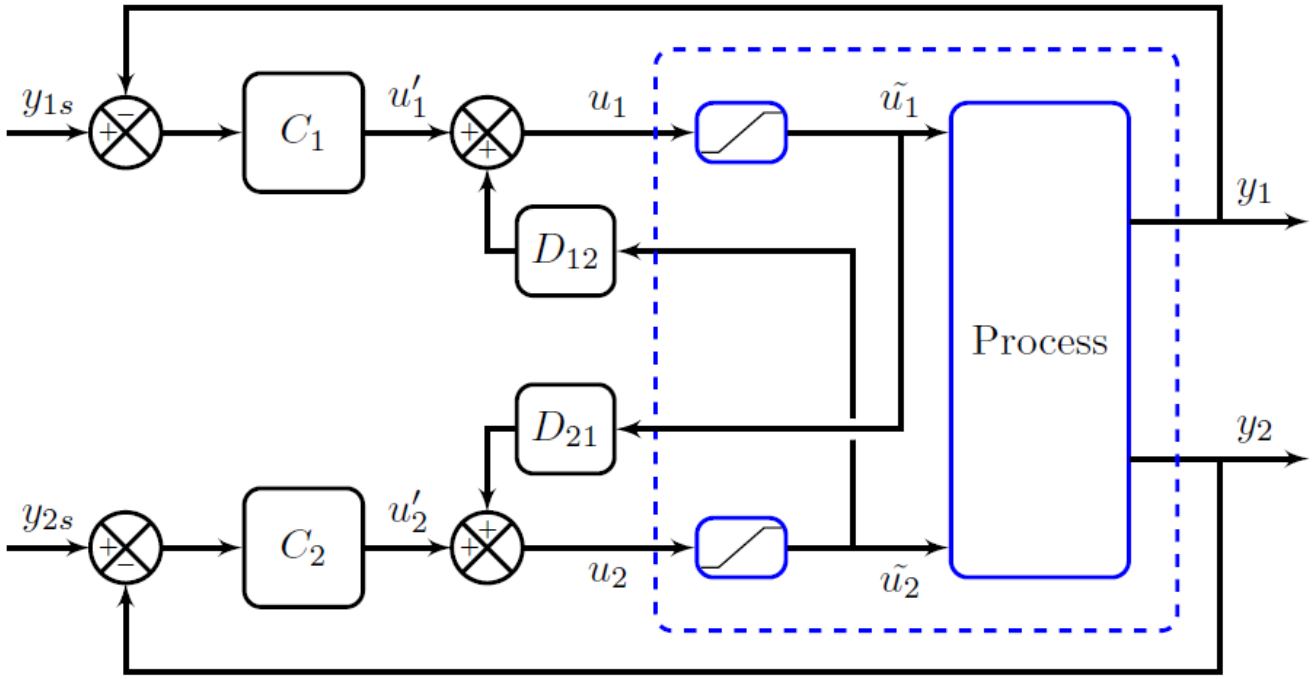


Figure 26: Linear decoupling with feedback (reverse) implementation of Shinskey (1979)

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} \qquad D_{12} = -\frac{G_{12}}{G_{11}}, \qquad D_{21} = -\frac{G_{21}}{G_{22}}$$

To make the decoupling elements realizable, we need a larger (effective) delay in the off-diagonal elements than in the diagonal elements of G. This means that the "pair close" rule should be followed also when using decoupling. An alternative is to use static decoupling or partial (one-way) decoupling.

Note that Figure 26 uses the feedback decoupling scheme of Shinskey (1979) which is called inverted decoupling (Wade, 1997). Compared with the to the more common "feedforward" scheme (where the input to the decoupling elements is u' rather than $\tilde{u}$), the feedback decoupling scheme in Figure 26 has the following nice features (Shinskey, 1979):

1. With inverted decoupling, the model from the controller outputs (u') to the process outputs (y) becomes (assuming no model error) $y_1 = G_{11}u'_1$ and $y_2 = G_{22}u'_2$. Thus, the system, as seen from the controllers C_1 and C_2, is in addition to being decoupled (as expected), also identical to the original process (without decoupling). This simplifies both controller design and switching between manual and auto mode. In other words, the tuning of C_1 and C_2 can be based on the open loop models (G_{11} and G_{22}).

2. The inverted decoupling works also for cases with input saturation, because the actual inputs ($\tilde{u}$) are used as inputs to the decoupling elements.

Note that there is potential problem with internal instability with the inverted implementation because of the positive feedback loop $D_{12}D_{21}$ around the two decoupling elements. However, this will not be a problem if we can follow the "pair close" pairing rule. In terms of the relative gain array (RGA), we should avoid pairing on negative RGA-elements. To avoid the stability problem (and also for other reasons, for example, to avoid sensitivity to model uncertainty for strongly coupled processes) one may use one-way decoupling where one of the decoupling elements is zero. For example, if tight control of y_2 is not important, one may select $D_{21} = 0$.

The scheme in Figure 26 can easily be extended to 3x3 systems and higher

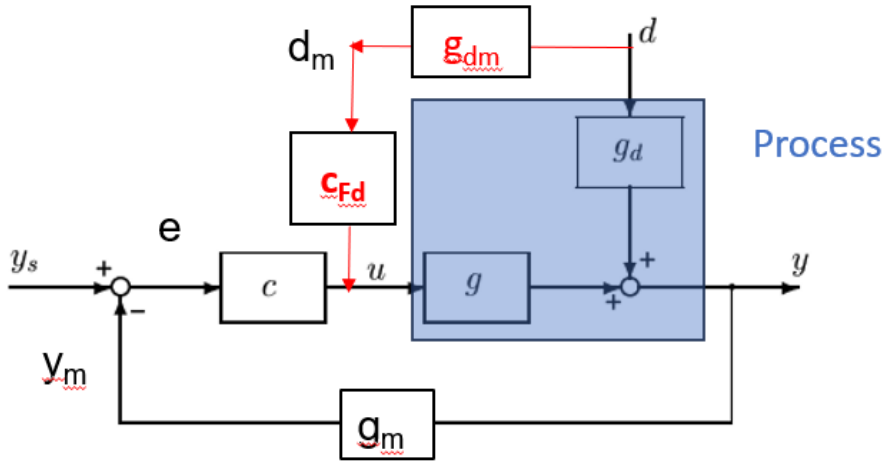
E13. Linearization elements

- Typically, logarithm or nonlinear feedforward blocks
- General approach: See Input transformations

E14. Calculation block based on transformed input (SEE LATER)

Linear feedforward, $u = C_{Fd} d_m + c\, e$

If perfect measurements: $g_m=1$, $g_{dm}=1$



Ideal Feedforward (with no feedback, c=0): Want y=0.
In the linear case
y = gd*d + g*u = (gd + g*cFd*gm) d
To get y=0, the ideal dynamic feedforward controller (if realizable) inverts the process:

$$Linear: \quad c_{Fd,ideal}(s) = g^{-1} g_d\, g_{dm}^{-1}$$

Annual Reviews in Control 56 (2023) 100903

Nonlinear static feedforward based on input transformations (with setpoint v0 from feedback):

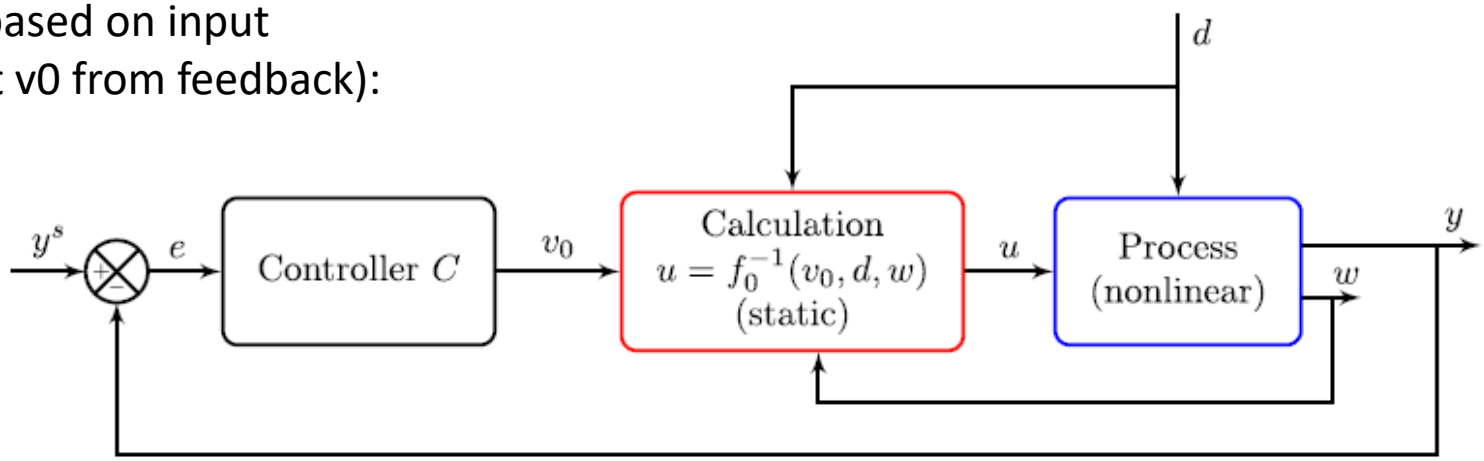
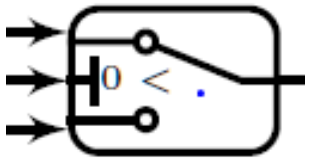


Fig. 29. Feedforward, decoupling and linearization (red calculation block) using transformed inputs $v_0 = f_0(u, d, w)$ based on static model $y = f_0(u, d, w)$. In the ideal case with no model error, the transformed system from v_0 to y (as seen from the controller C) becomes $y = I v_0$ at steady state.
d = measured disturbance
w = measured process state variable.

Main problem feedforward: Sensitive to model error and changes (nonlinearity)

Additional standard elements

- E16. Simple nonlinear static elements
  - Multpliction
  - Division (avoid or at least be careful)
  - Square root
  - Dead zone
  - Dead band
  - Limiter (saturation element) 
  - On/off
- E17. Simple linear dynamic elements
  - Lead-lag filter
  - Time delay
  - … more…
- E18. Standard logic elements
  - If, then, else
  - Example: Select depending on sign of another signal: 

What about the Smith Predictor? Forget it!

Note that the Smith Predictor (Smith, 1957) is not included in the list of 18 control elements given in the Introduction, although it is a standard element in most industrial control systems to improve the control performance for processes with time delay. The reason why it is not included, is that PID control is usually a better solution, even for processes with a large time delay (Grimholt & Skogestad, 2018b; Ingimundarson & Hägglund, 2002). The exception is cases where the true time delay is known very accurately. There has been a myth that PID control works poorly for processes with delay, but this is not true (Grimholt & Skogestad, 2018b). The origin for the myth is probably that the Ziegler–Nichols PID tuning rules happen to work poorly for static processes with delay.

The Smith Predictor is based on using the process model in a predictive fashion, similar to how the model is used in internal model control (IMC) and model predictive control (MPC). With no model uncertainty this works well. However, if tuned a bit aggressively to get good nominal performance, the Smith Predictor (and thus also IMC and MPC) can be extremely sensitive to changes in the time delay, and even a *smaller* time delay can cause instability. When this sensitivity is taken into account, a PID controller is a better choice for first-order plus delay processes (Grimholt & Skogestad, 2018b).

A smart invention: Cross-limiting control

Industry also makes use of other smart solutions, which do not follow from the standard structures presented in this paper.

One example is cross-limiting control for combustion, where the objective is to mix air (A) and fuel (F) in a given ratio, but during dynamic transients, when there will be deviations from the given ratio, one should make sure that there is always excess of air. The scheme in Fig. 39 with a crossing min- and max-selector achieves this. It is widely used in industry and is mentioned in many industrial books (e.g., Liptak (1973), Nagy (1992) and Wade (2004)). The setpoint for the ratio, $(F_F/F_A)_s$, could be set by a feedback controller (not shown) which controls, for example, the remaining oxygen after the combustion.

The selectors in Fig. 39 are used to handle the dynamic (transient) case, so this is a somewhat rare case where the selectors are not performing a steady state CV-CV switch.

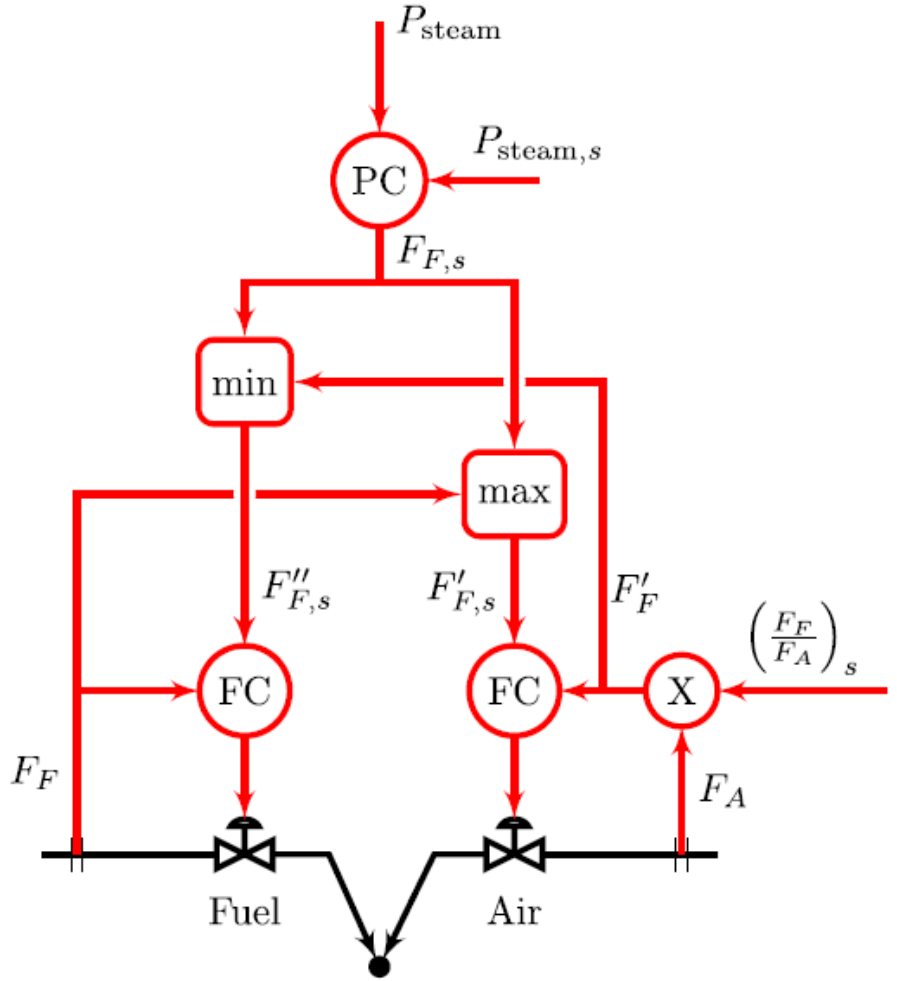


Fig. 39. Cross-limiting control for combustion where air (A) should always be in excess to fuel (F).

Standard advanced control elements studied in this paper.

Control element	Main use	Inputs	Outputs
E1. Cascade control Figs. 9 and 10	Linearization and local disturbance rejection	Outer master controller: • CV_{1s}-CV Inner controller: • CV_{2s}-CV_2	Outer master controller: • CV_{2s} Inner controller: • MV
E2. Ratio control Fig. 11	Feedforward or decoupling without model (assumes that scaling property holds)	• R (desired ratio) • DV or MV_1	• MV = R · DV, or • MV_2 = R · MV_1
E3. VPC on extra dynamic input Fig. 12	Use extra dynamic input MV_1 to improve dynamic response (because MV_2 alone is not acceptable). MV_1 setpoint is unconstrained (mid-range) and controlled all the time	• MV_{1s} - MV_1	• MV_2
E4. Selector Figs. 17, 18 and 19	CV-CV switching: Many CVs (CV_1, CV_2, …) controlled by one MV	• MV_1 • MV_2, … (generated by separate controllers for CV_1, CV_2, …)	• MV = max/min (MV_1, MV_2, …)
E5. Split-range control Figs. 21 and 23	MV-MV switching: One CV controlled by sequence of MVs (using only one controller)	• CV_s-CV	• MV_1 • MV_2, …
E6. Separate controllers with different setpoints Fig. 22	MV-MV switching: One CV controlled by sequence of MVs (using individual controllers with different setpoints)	• CV_{s1} – CV • CV_{s2} - CV …	• MV_1 • MV_2 …
E7. VPC on main steady-state input Fig. 24	MV-MV switching: One CV controlled by main MV_1 with use of extra MV_2 to avoid saturation of MV_1. MV_1 setpoint is close to constraint and only controlled when needed	• MV_{1s} - MV_1	• MV_2
E9. Two degrees-of-freedom feedback controller Fig. A.41	Treat setpoint (CV_s) and measurement (CV) differently in controller C	• CV_s • CV	• MV
E11. Feedforward control Fig. A.42	Reduce effect of disturbance (using model from DV and MV to CV)	• DV	• MV
E12. Decoupling element Fig. 26	Reduce interactions (using model from MV_1 and MV_2 to CV)	• MV_1 • MV_2	• MV_2 • MV_1
E14. Calculation block based on transformed input Fig. 27	Static nonlinear feedforward, decoupling and linearization based on nonlinear model from MV, DV and w to CVv	• Transformed input = feedback trim (v) • DV (d) • Extra meas. (w)	• MV (u)

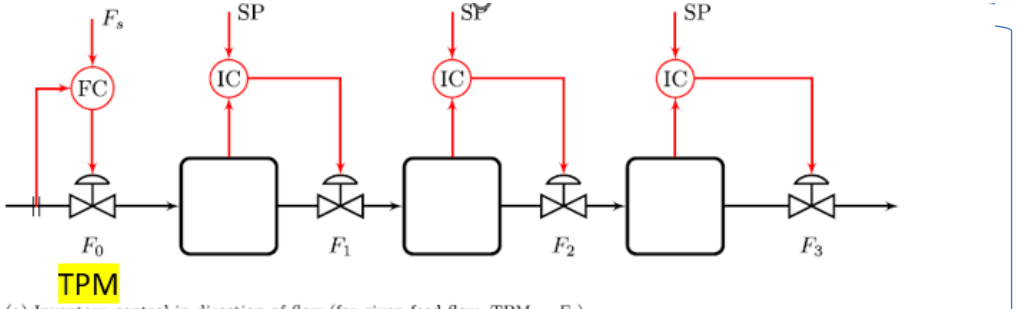
Comment on need for rules

- The human brain (at least mine) has problems in analyzing even quite simple cases
- Two «simple» cases are:
  - choice of max- and min- selectors
  - how to get consistent inventory control
- I frequently need to og back to the «selector rules» or the «radiation rule» to get this right.

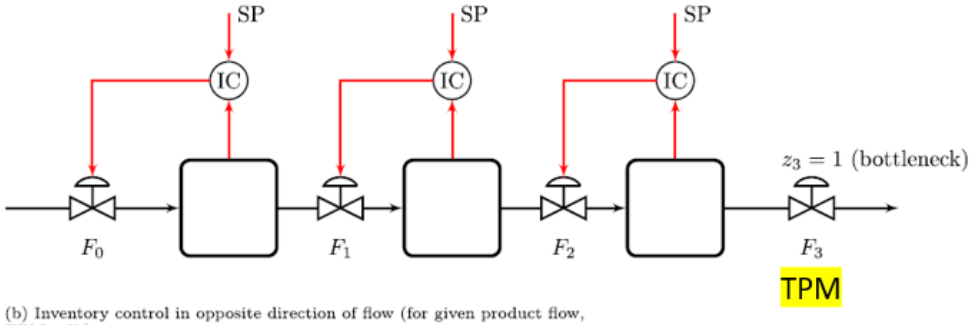
More on inventory control

Inventory control for units in series

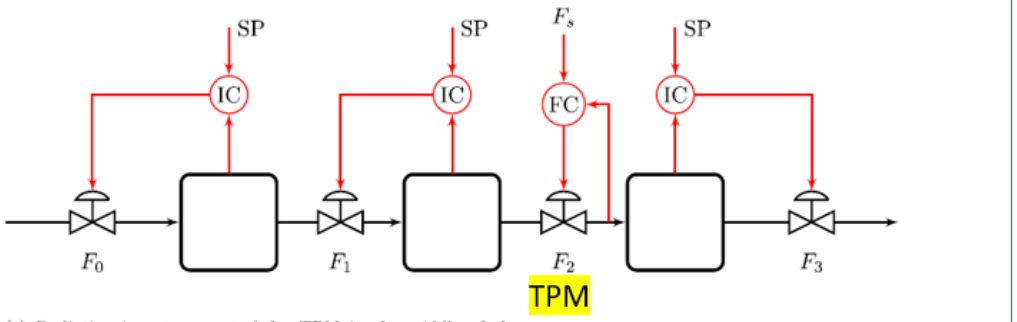
Radiating rule:
Inventory control should be "radiating" around a given flow (TPM).



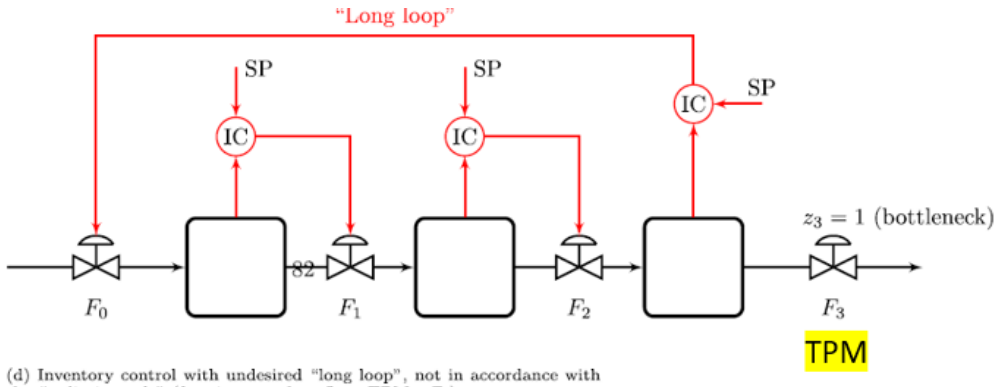
(a) Inventory control in direction of flow (for given feed flow, TPM = F_0)



(b) Inventory control in opposite direction of flow (for given product flow, TPM= F_3)



(c) Radiating inventory control for TPM in the middle of the process (shown for TPM = F_2)

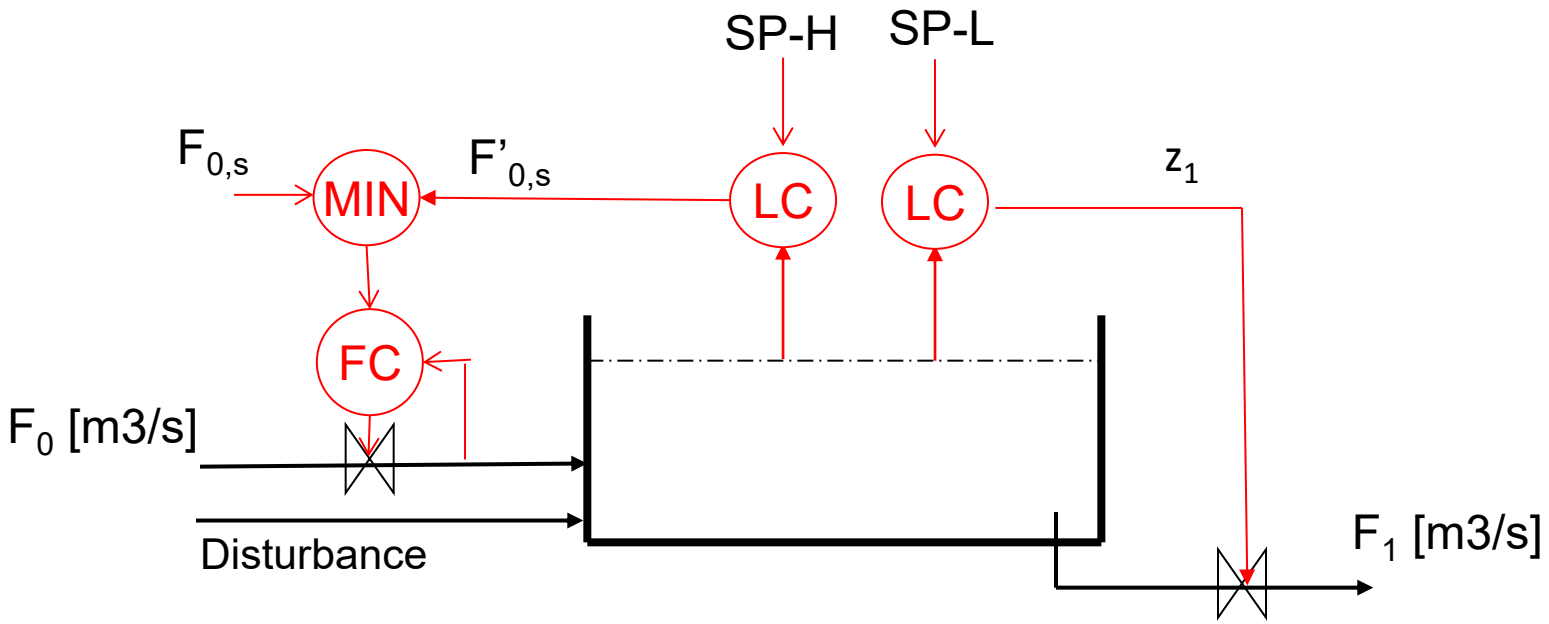


(d) Inventory control with undesired "long loop", not in accordance with the "radiation rule" (for given product flow, TPM= F_3)

Follows radiation rule

Does NOT follow radiation rule

Recall: Bidirectional inventory control using Split parallel control



SP-L = low level setpoint = 50 % (or 20%)
SP-H = high level setpoint = 60 % (or 80%)

Use of two setpoints is good for using buffer dynamically!!

Generalization of bidirectional inventory control

Reconfigures TPM automatically with optimal buffer management!!

Maximize throughput:
$F_s=\infty$

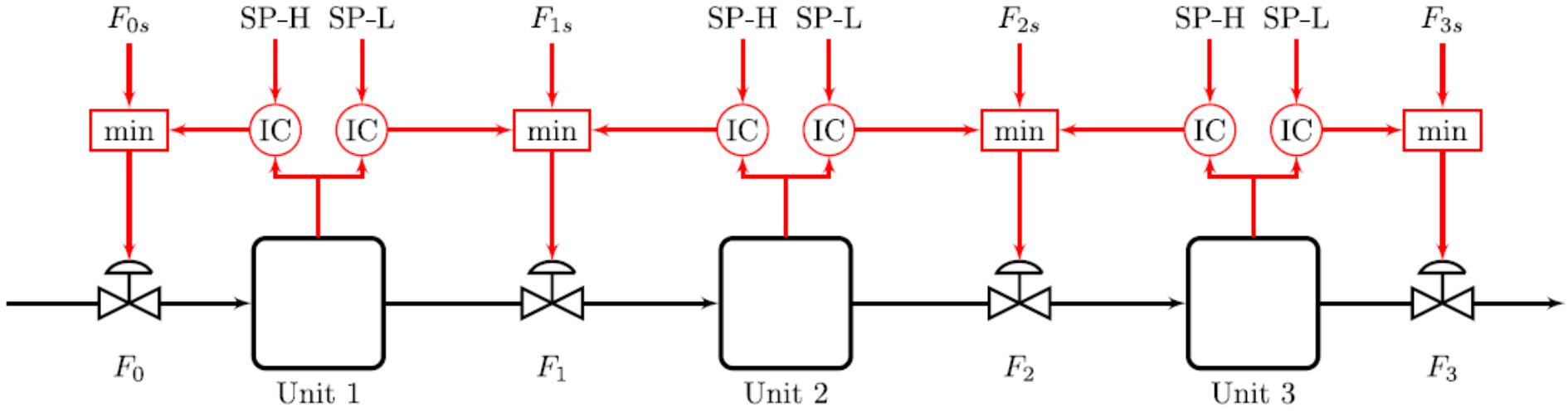


Fig. 36. Bidirectional inventory control scheme for automatic reconfiguration of loops (in accordance with the radiation rule) and maximizing throughput. Shinskey (1981) Zotică et al. (2022).
SP-H and SP-L are high and low inventory setpoints, with typical values 90% and 10%.
Strictly speaking, with setpoints on (maximum) flows ($F_{i,s}$), the four valves should have slave flow controllers (not shown). However, one may instead have setpoints on valve positions (replace $F_{i,s}$ by $z_{i,s}$), and then flow controllers are not needed.

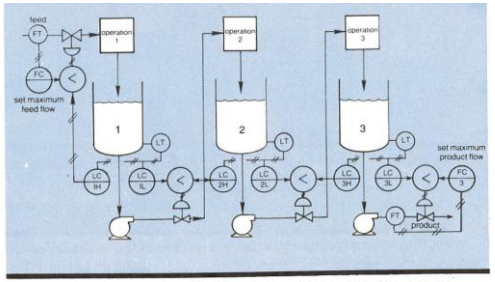


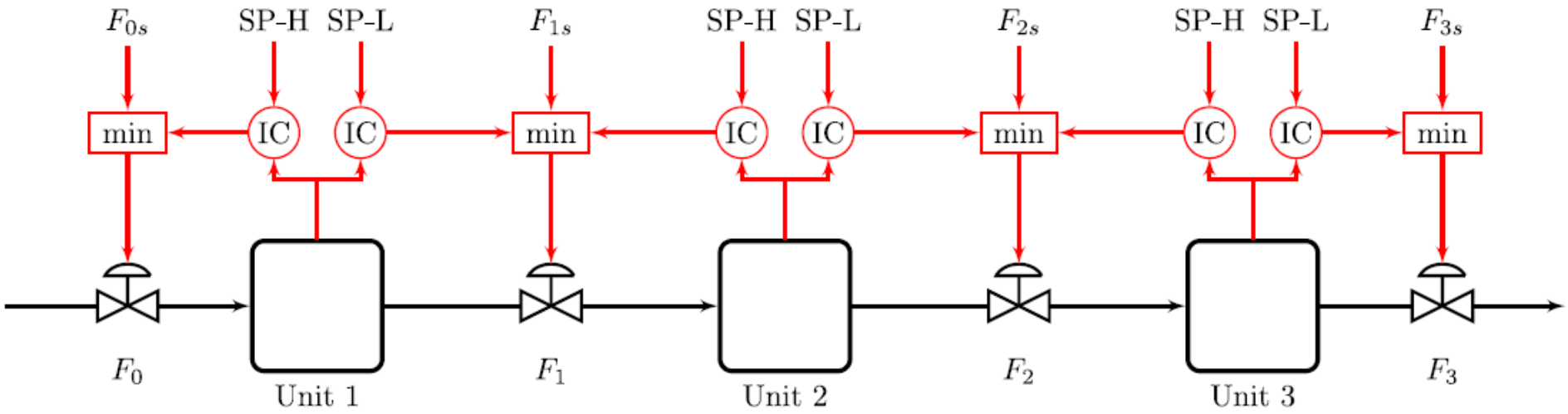
Fig. 3-7. Production rate can be set at either end of the process or constrained at any intermediate point without loss of inventory control.

F.G. Shinskey, «Controlling multivariable processes», ISA, 1981, Ch.3

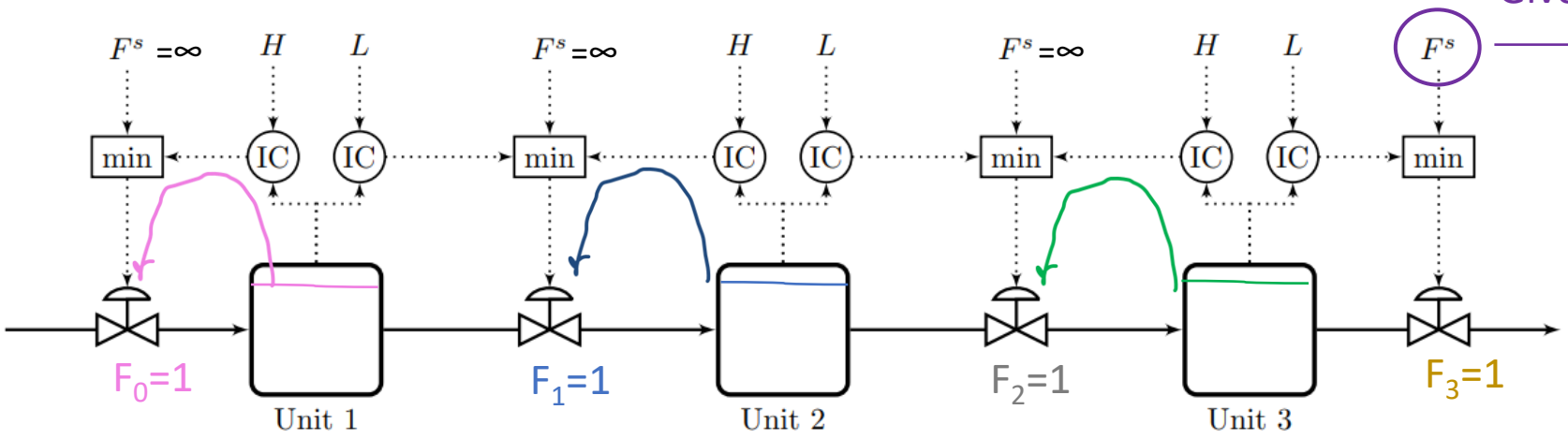
Cristina Zotica, Krister Forsman, Sigurd Skogestad ,»Bidirectional inventory control with optimal use of intermediate storage», Computers and chemical engineering, 2022

Bidirectional inventory control

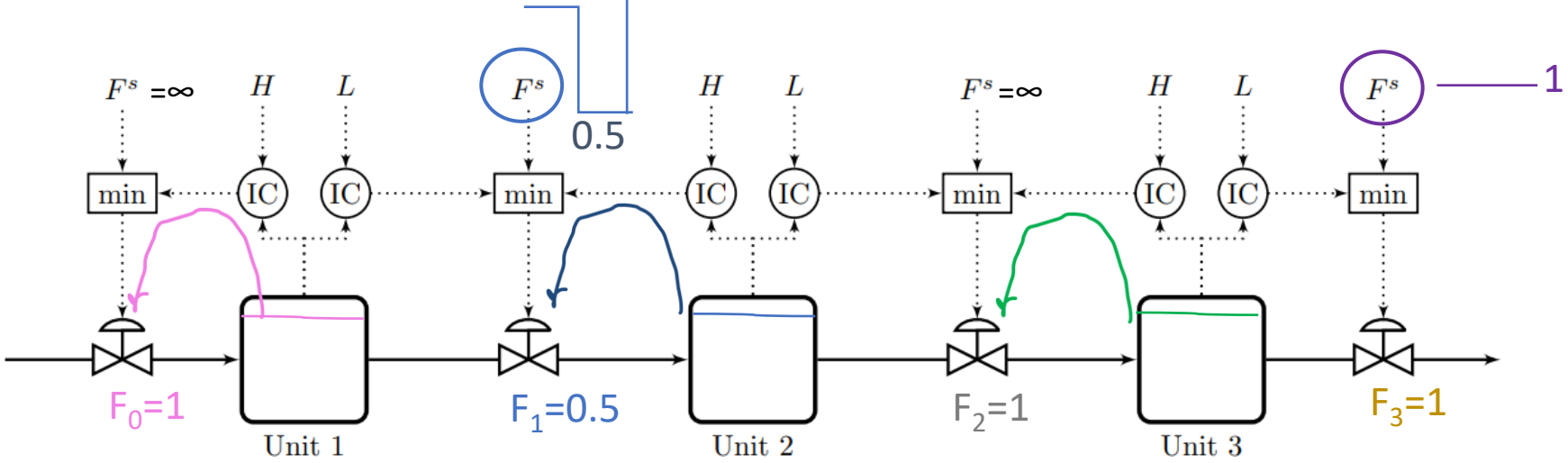
Reconfigures TPM automatically with optimal buffer management!!



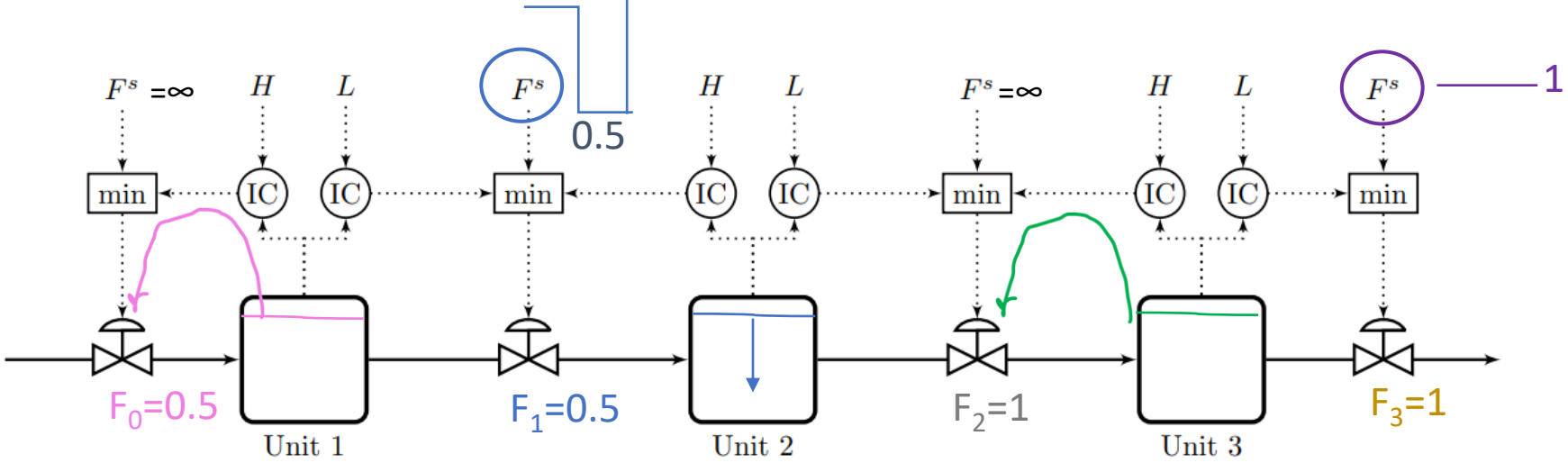
*Shinskey: "*Production rate can be set at either end of the process or constrained at any intermediate point without loss of inventory control…… The tank capacities are used for buffering between operations, delaying the transmission of upsets in either direction. Momentary upsets in one operation might not interfere with adjacent operations at all. "



All levels are high (SP-H)

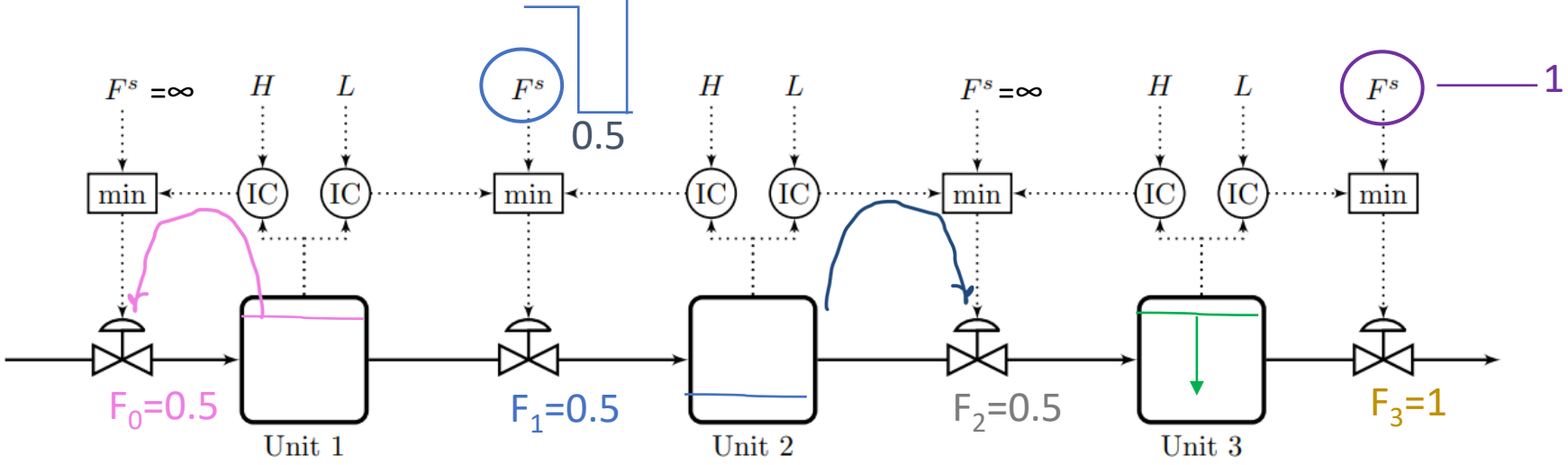


Temporary reduction in flow F1 (feed to unit 2)

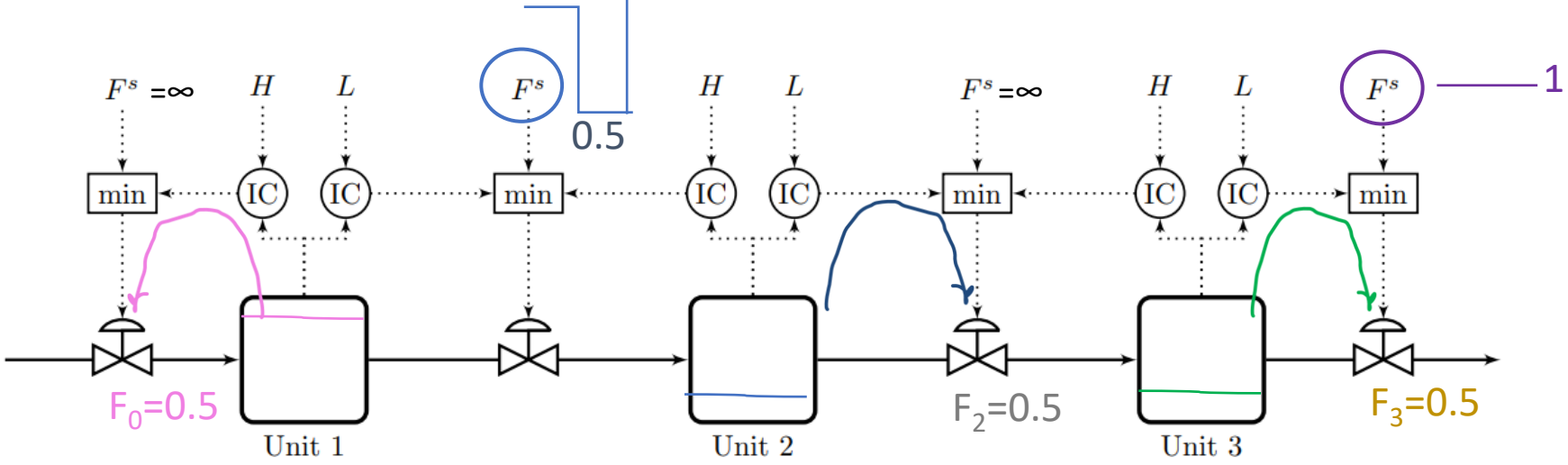


Level 1 constant:
Reduction in feed to unit 1

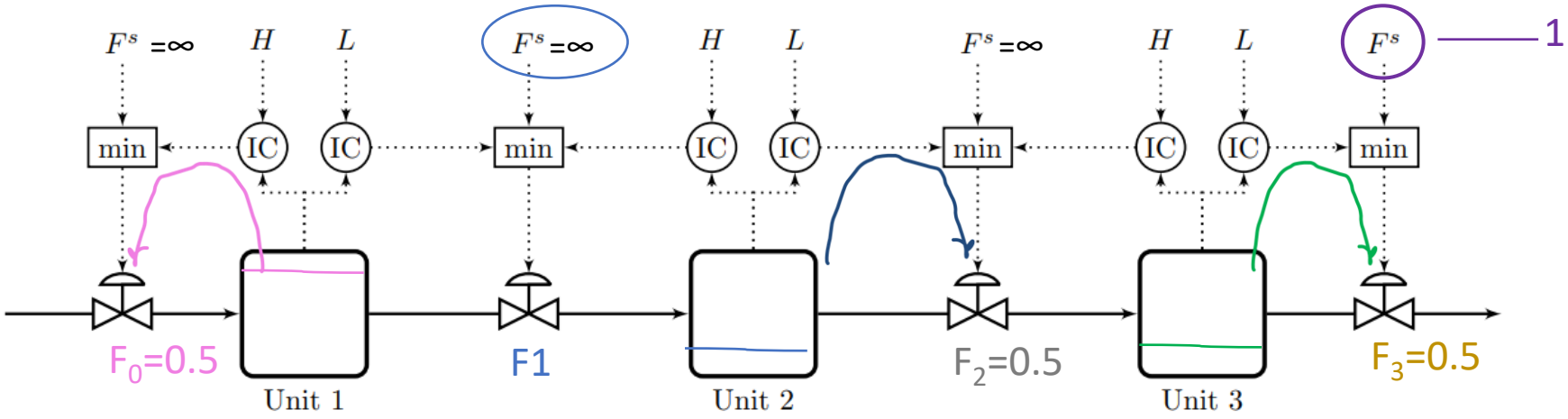
Level in unit 2 drops



Level 2 reaches SP-L:
Flow reduction
moves to unit 3



Flow reduction reaches product after some time



Temporary flow
reduction in F1 is over.
Get z_1=1 (fully open).

System recovers:
Temporary need $F_0 > 1$

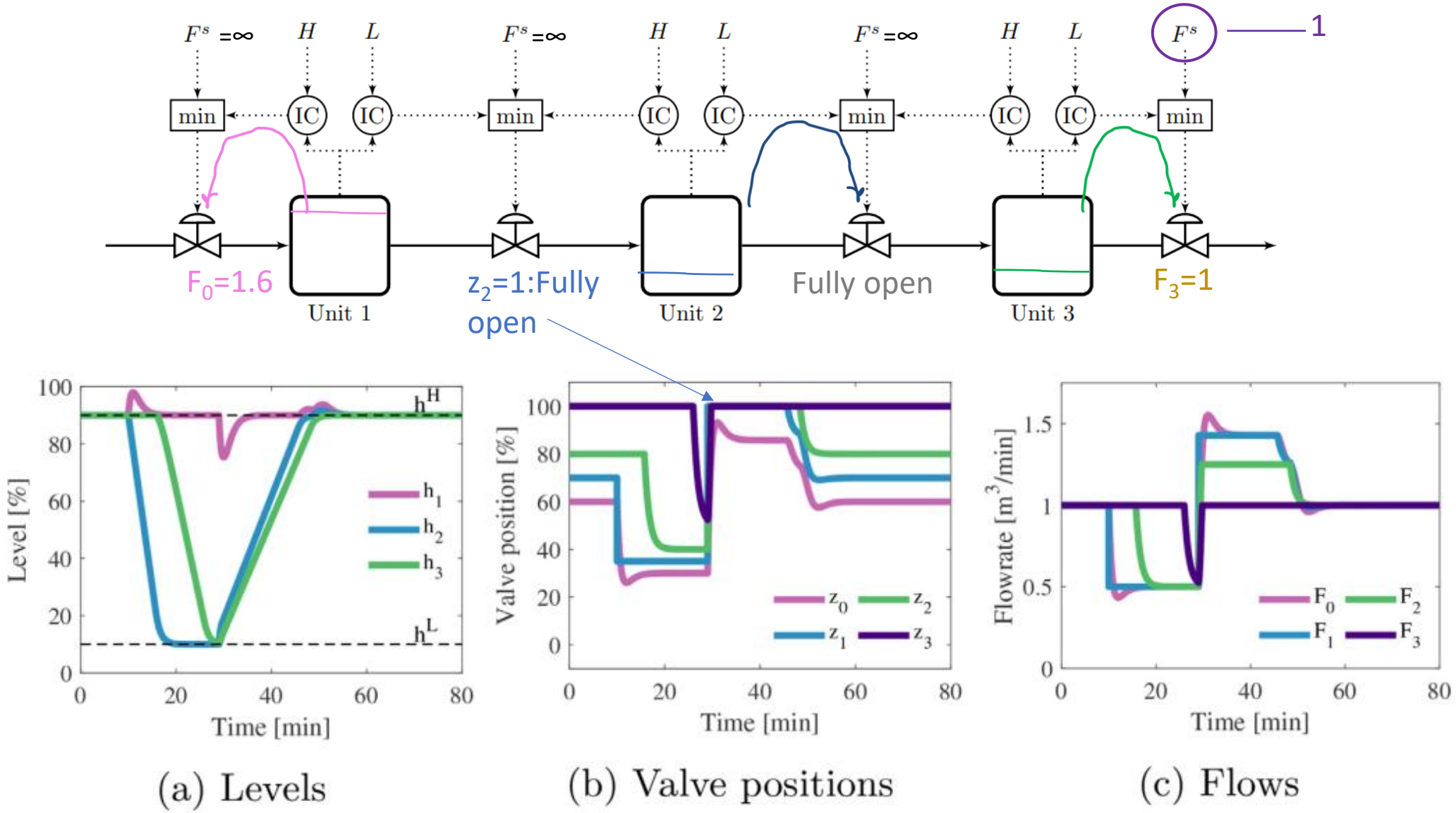


Fig. 13. Simulation of a temporary (19 min) bottleneck in flowrate F_1 for the proposed control structure in Fig. 10. The TPM is initially at the product (F_3).

Final steady state
All levels are high (SP-H)

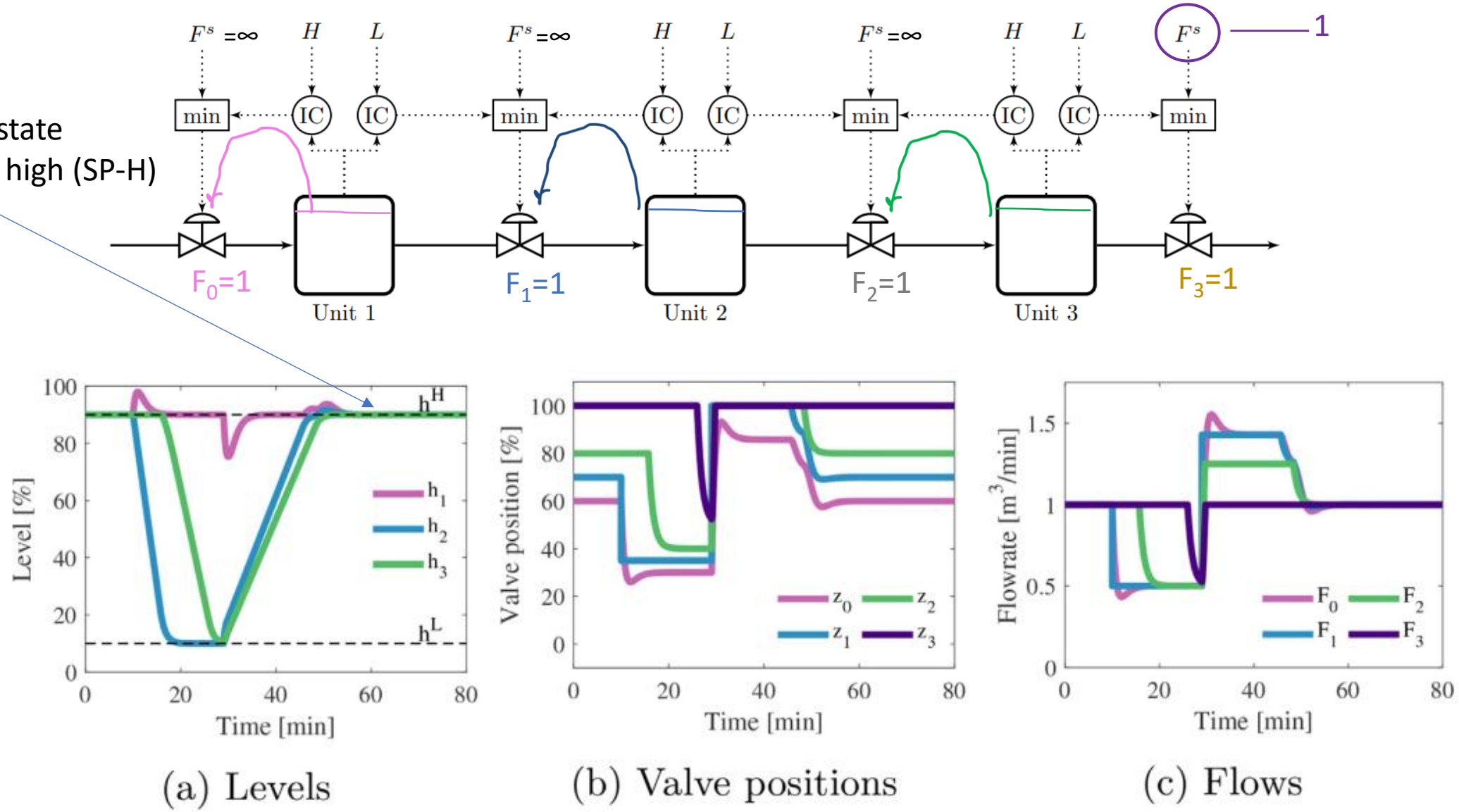
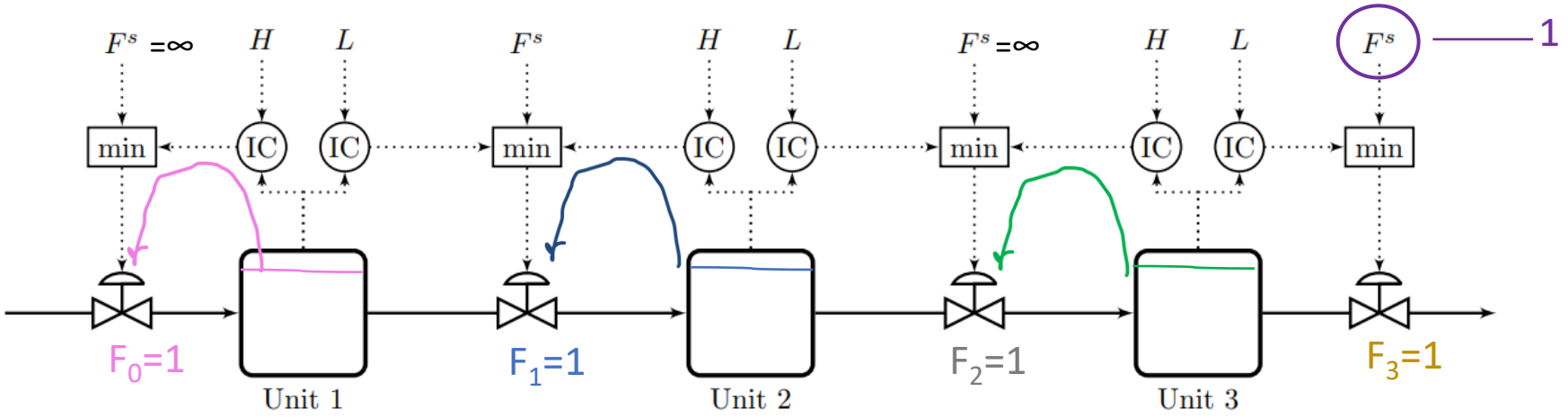


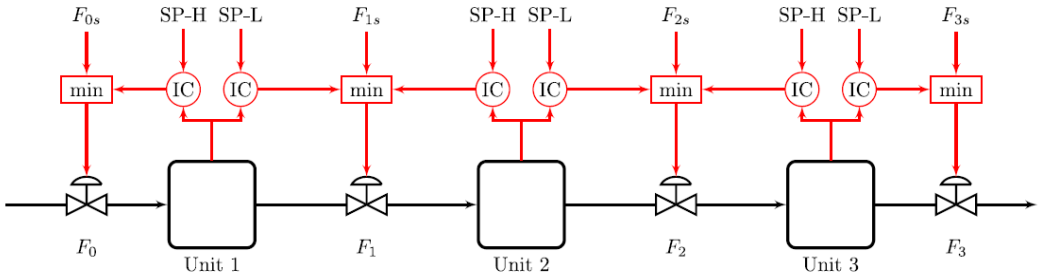
Fig. 13. Simulation of a temporary (19 min) bottleneck in flowrate F_1 for the proposed control structure in Fig. 10. The TPM is initially at the product (F_3).



Challenge: Can MPC be made to do his? Optimally reconfigure loops and find optimal buffer?

YES. Use «trick»/insight of unachievable high setpoints on all flows

Comments on Bidirectional inventory control



- It's almost like magic (meaning that it's difficult to understand what is actually happening)
- It both moves the TPM optimally and gives optimal levels.
- It is more like an invention.
- One cannot generally except to be able to solve complex problems without coordination, but this is a special case.

Extension . **Bidirectional inventory control with minimum flow for F_2**

Max flow: $F_s = \infty$
L = 10%,
M_L = 40%,
M_H = 60%
H = 90%.

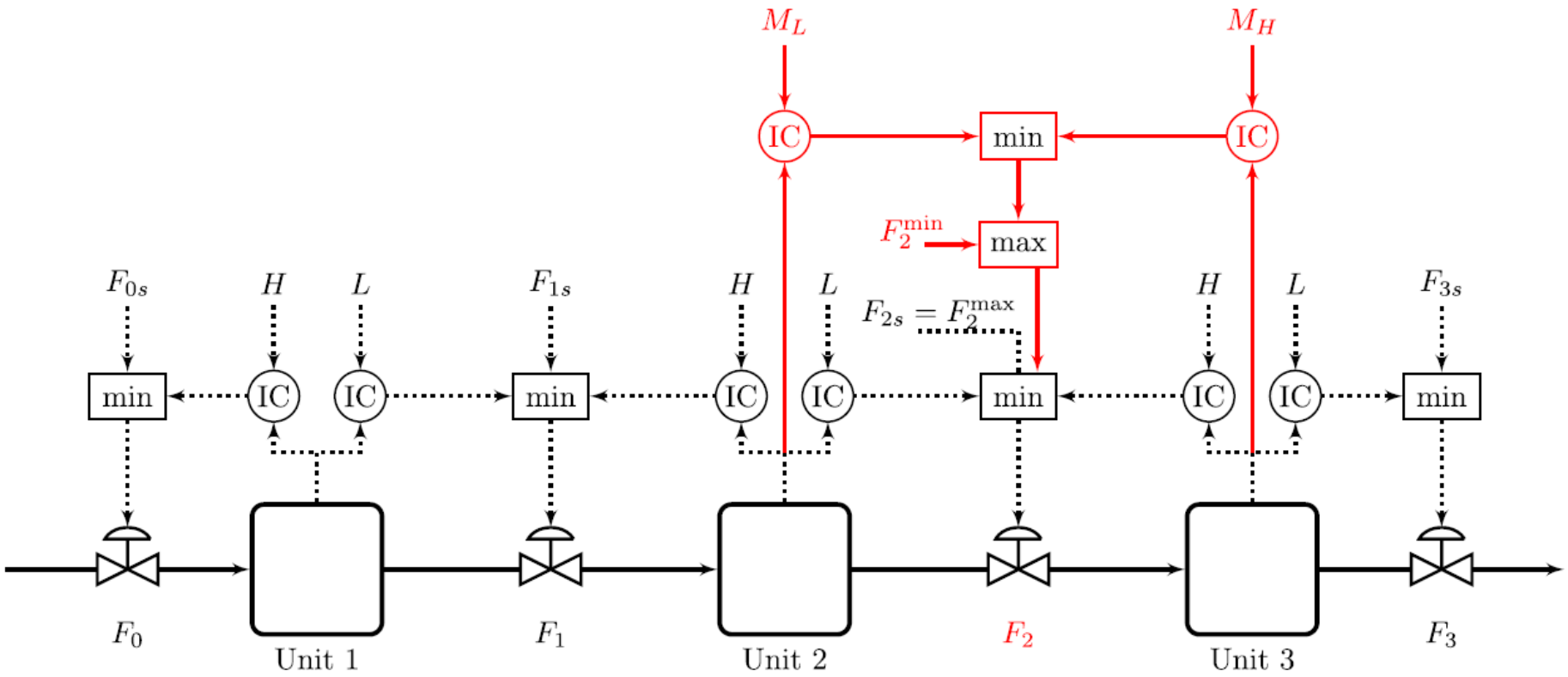


Fig. 37. Bidirectional inventory control scheme for maximizing throughput (dashed black lines) while attempting to satisfy minimum flow constraint on F_2 (red lines). H, L, M_L and M_H are inventory setpoints.

The control structure in Fig. 37 may easily be dismissed as being too complicated so MPC should be used instead. At first this seems reasonable, but a closer analysis shows that MPC may not be able to solve the problem (Bernardino & Skogestad, 2023).[8] Besides, is the control structure in Fig. 37 really that complicated? Of course, it is a matter of how much time one is willing to put into understanding and studying such structures. Traditionally, people in academia have dismissed almost any industrial structure with selectors to be ad hoc and difficult to understand, but this view should be challenged.

Industrial example (Perstorp)

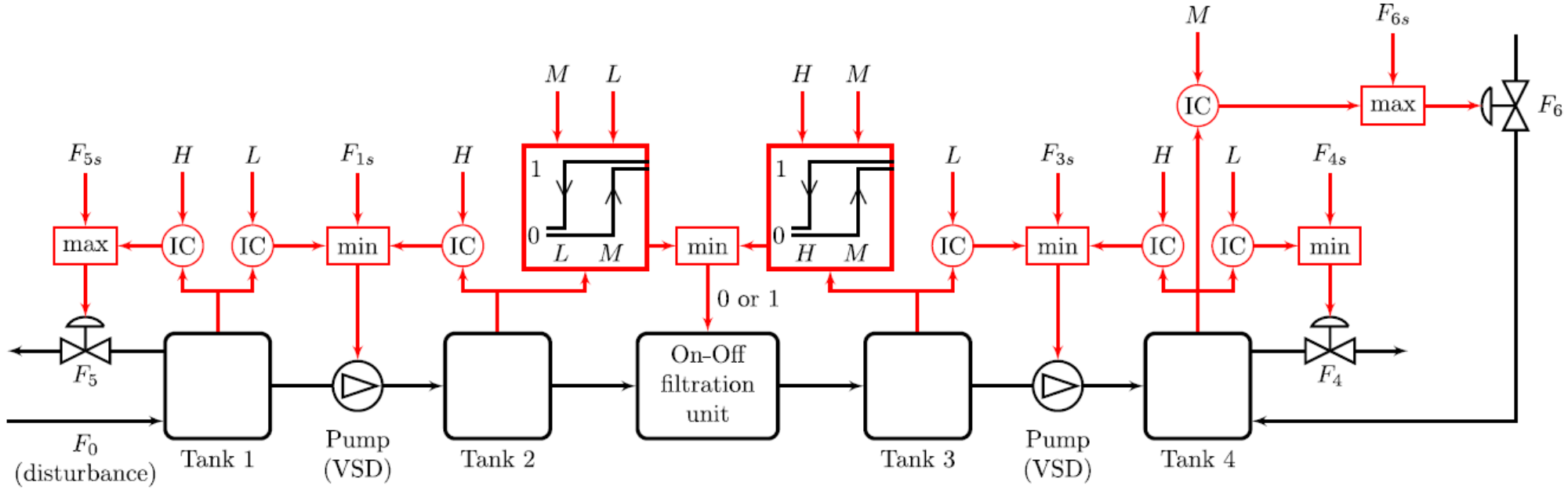
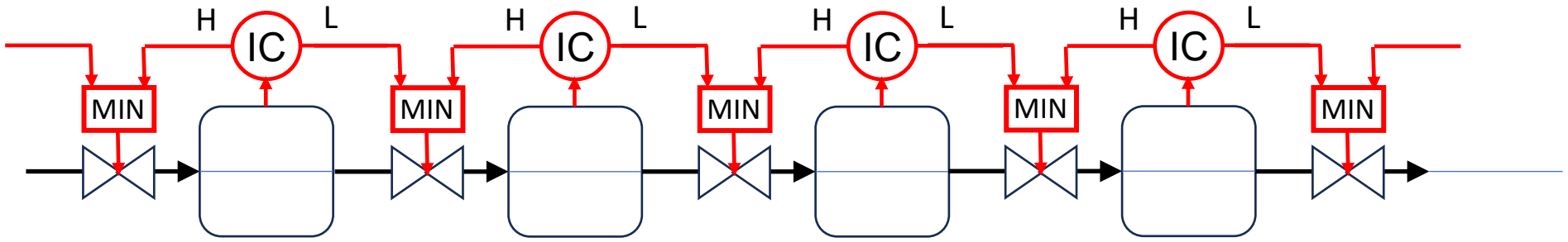
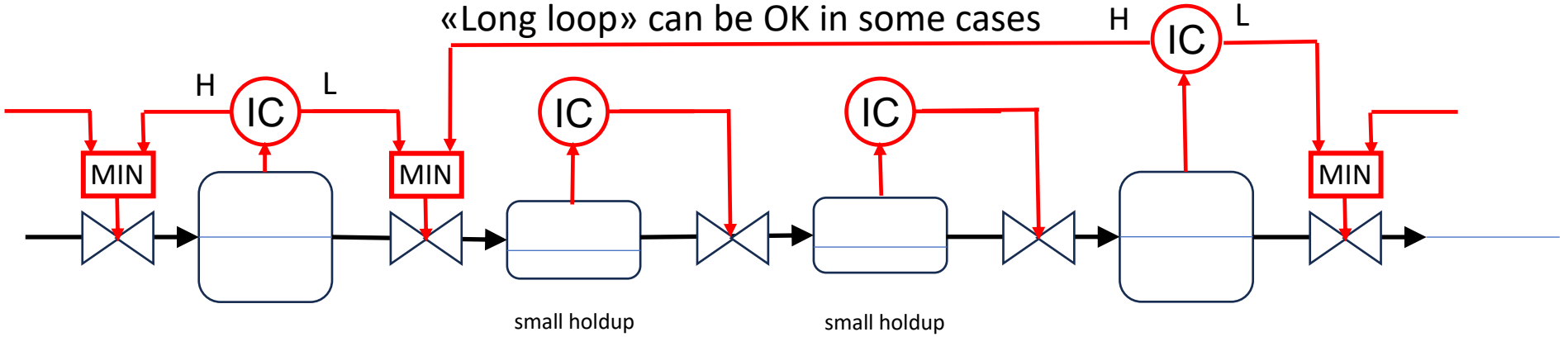


Fig. 38. Bidirectional inventory control structure for industrial plant with on/off (1/0) control of filtration unit.
H, L and M are inventory setpoints with typical values 90%, 10% and 50%.
If it is desirable to set a flowrate (F_s) somewhere in the system, then flow controllers must be added at this location.

Don’t need bidirectional control on all units





Level control

1. Pairing: Use in- or outflow
   - Radiation rule
2. Tuning: Tight or slow («averaging») level control?
   - Averaging (slow) is good to dampen flow disturbances
     - But is this really so important?
   - «Floating» (uncontrolled level) is good for isolating process parts
     - This may be achieved with bidirectional inventory control.
     - But requires tight control when we reach max- or min level

Bidirectional control for recycle processes

Bidirectional control for plants with recycle

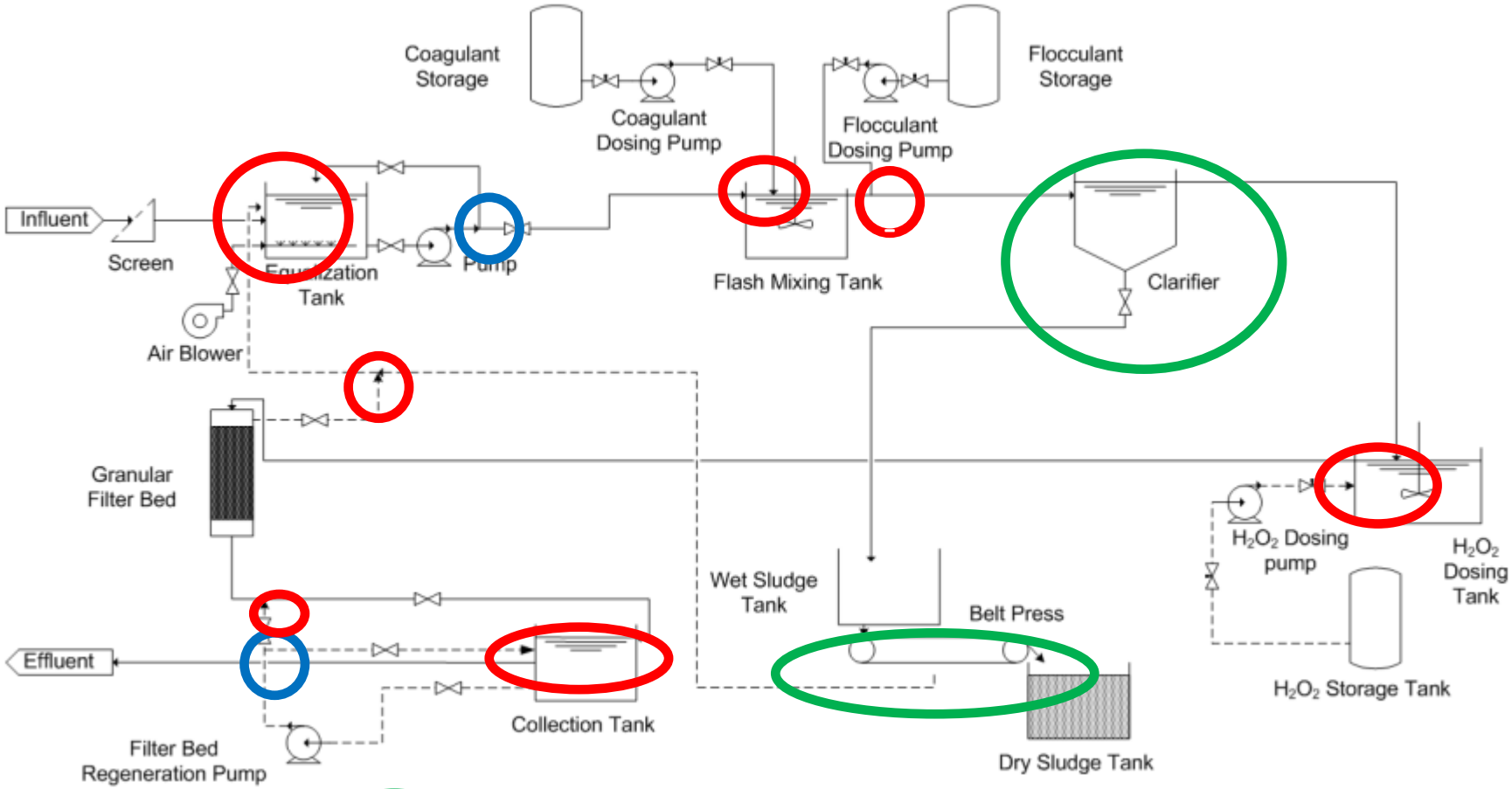
Recycle combines split and mixing.

Two cases of split:

- «Fixed split» (Separator). No extra control DOF
- «Adjustable split» (stream split, Extra control DOF)

DOF = degree of freedom

Example



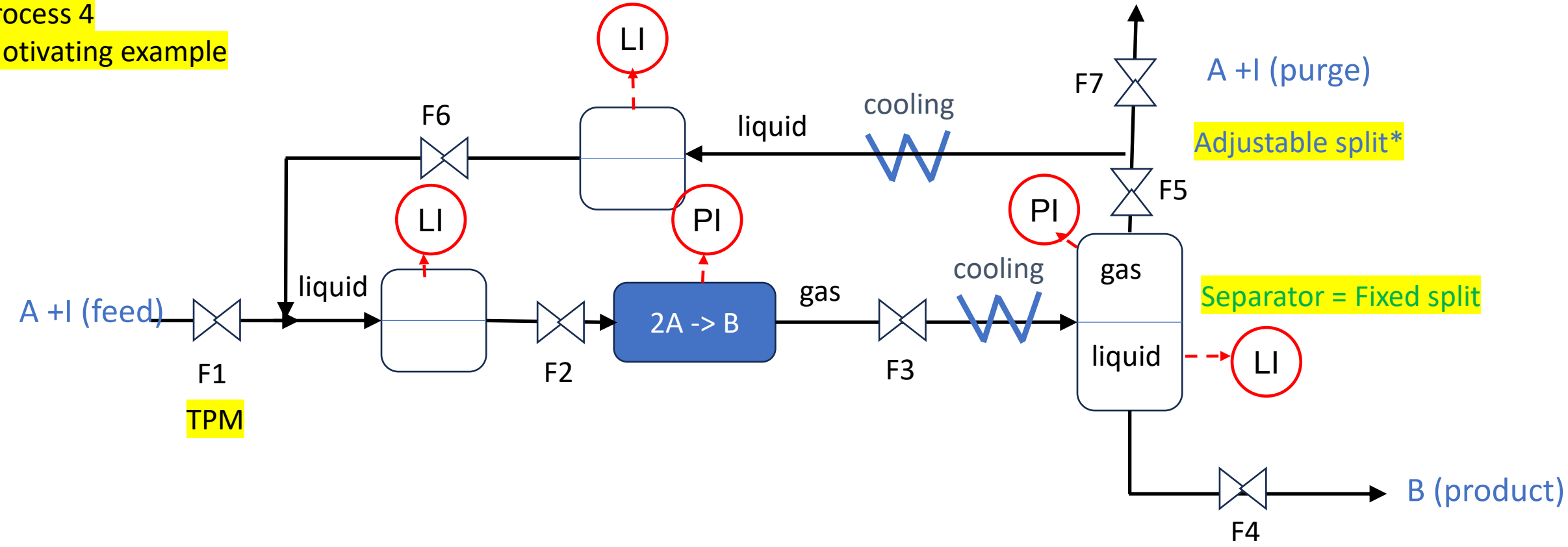
Mixing
(use ratio control for feeds)

Separator
(«fixed» split)

Adjustable split
(same composition)

Chemical Engineering Research Bulletin 14(2010) 65-68 / Rahman and Kabir

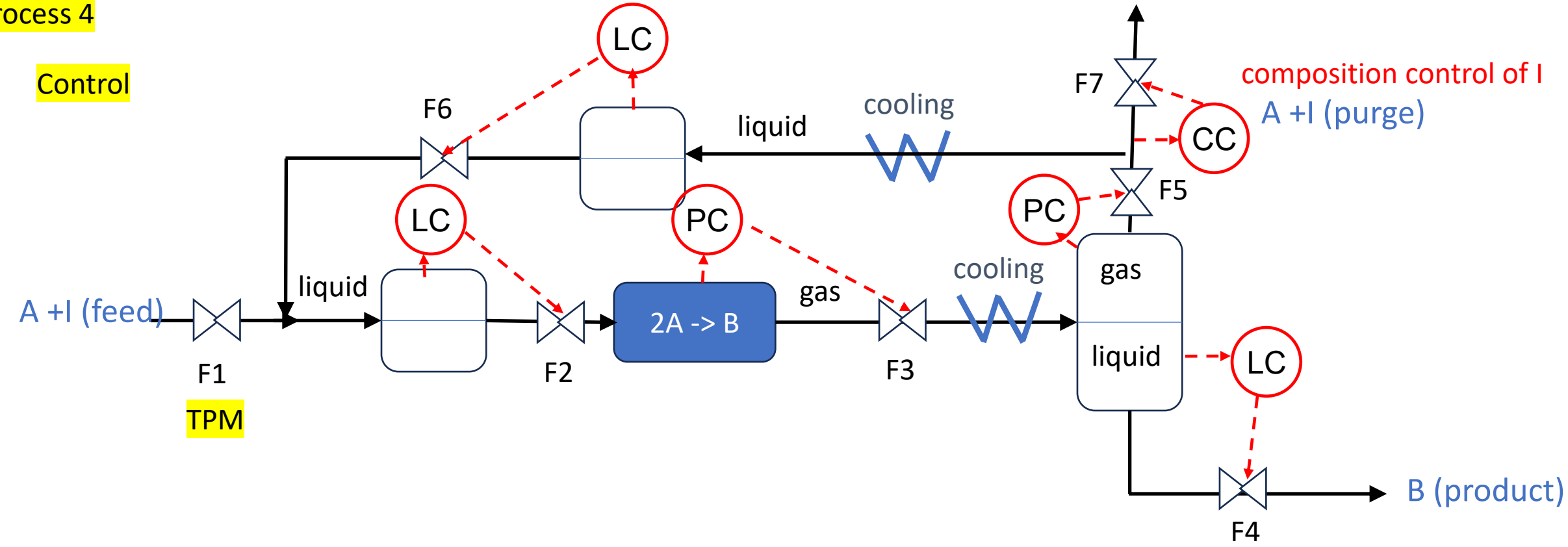
Process 4
Motivating example



*Adjustable split gives extra DOF.
BUT: I do not normally recommend two valves in series (like here) unless purge is very small.
So would normally move F5 to the recycle.

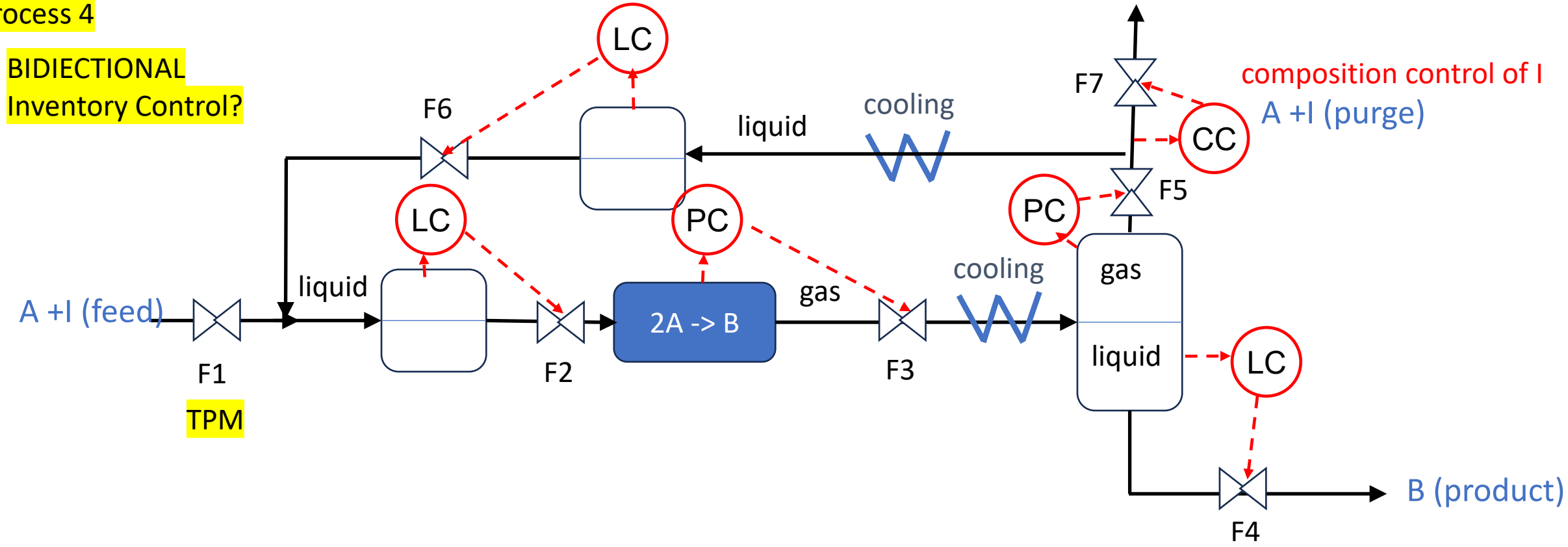
Process 4

Control



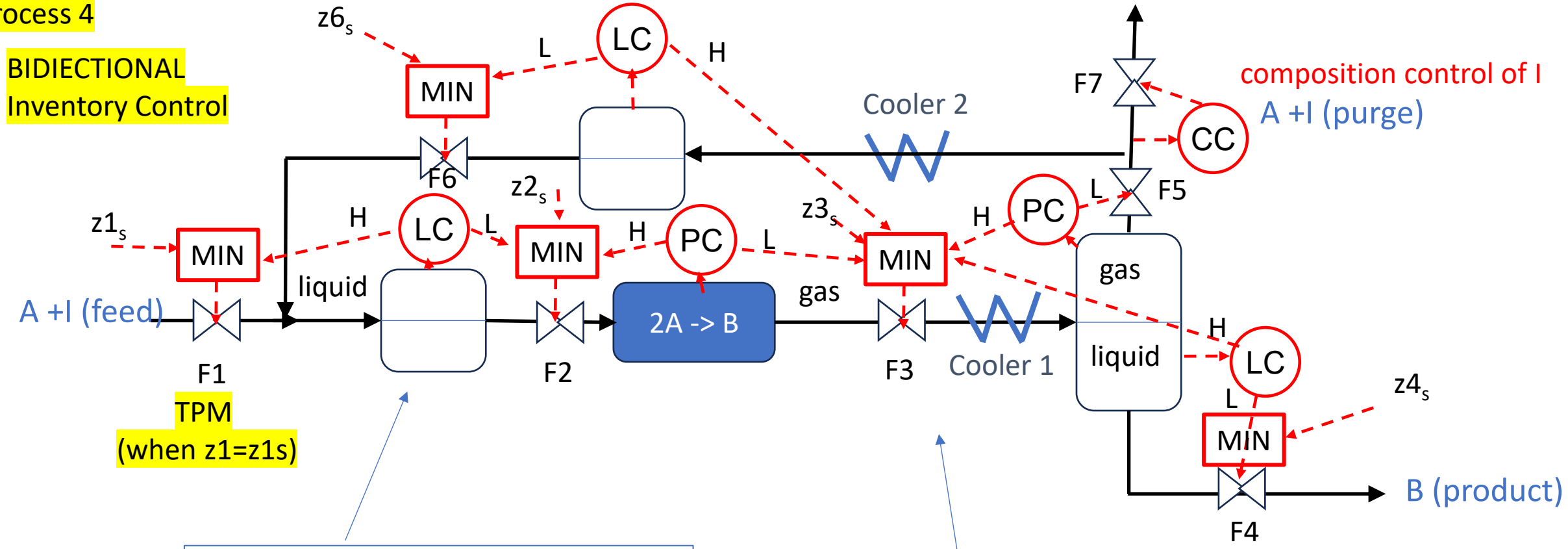
Process 4

BIDIECTIONAL
Inventory Control?



Process 4

BIDIECTIONAL
Inventory Control



This LC is two controllers which both control level.

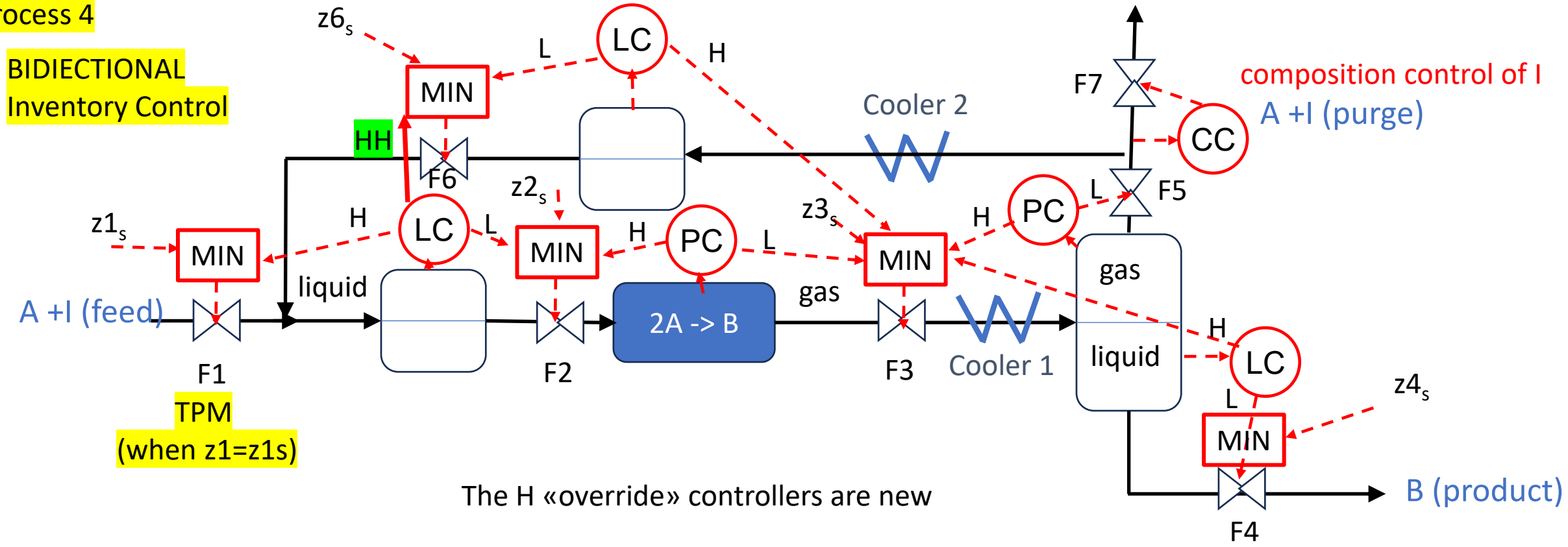
- The one with outflow F2 as the MV has a Low level setpoint
- The one with inflow F1 as the MV has a High level setpoint

3 H-setpoints go to this MIN-selector:
Reduce F3 if

1. too high pressure (cooler 2 max),
2. too much gas (high level) (F6 limiting)
3. too much liquid (high level) (F4 limiting)

Process 4

BIDIECTIONAL
Inventory Control



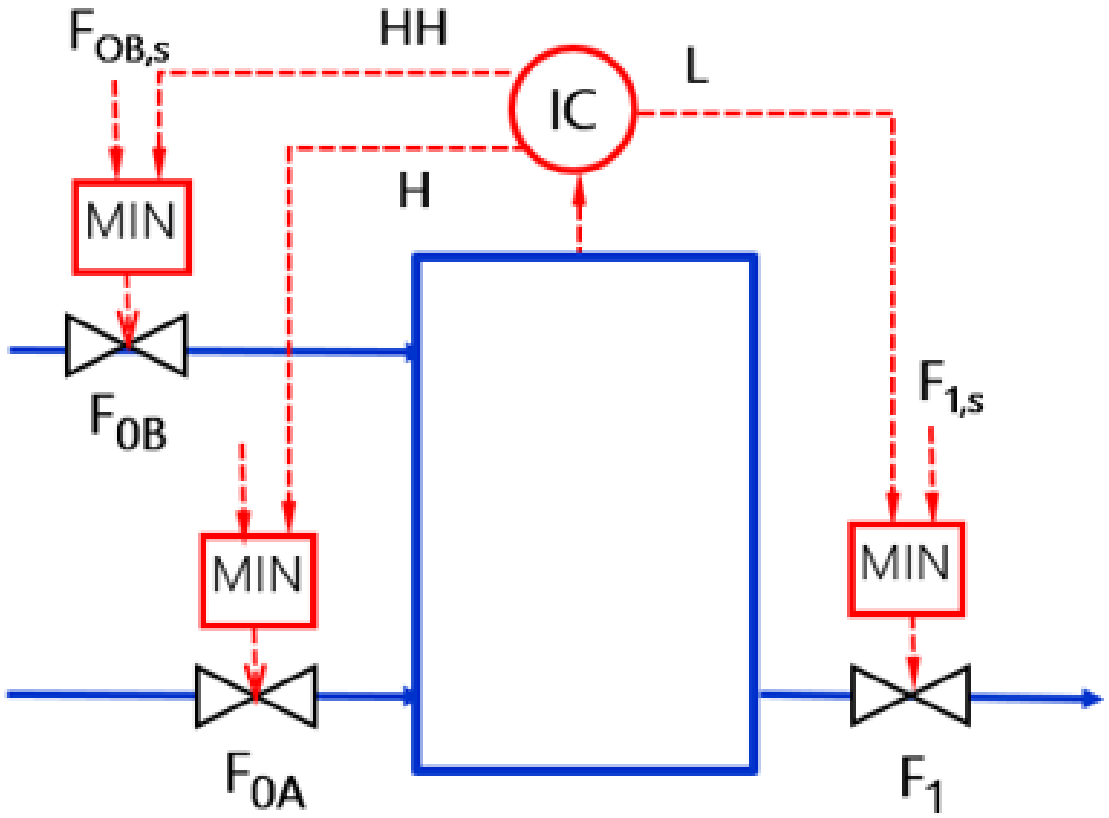
The H «override» controllers are new

If the Recycle is very large (F6 >> F1) then it may be necessary to add a HH-override to protect tank 1 from overflowing (it will only have a dynamic effect)

Bidrectional inventory control

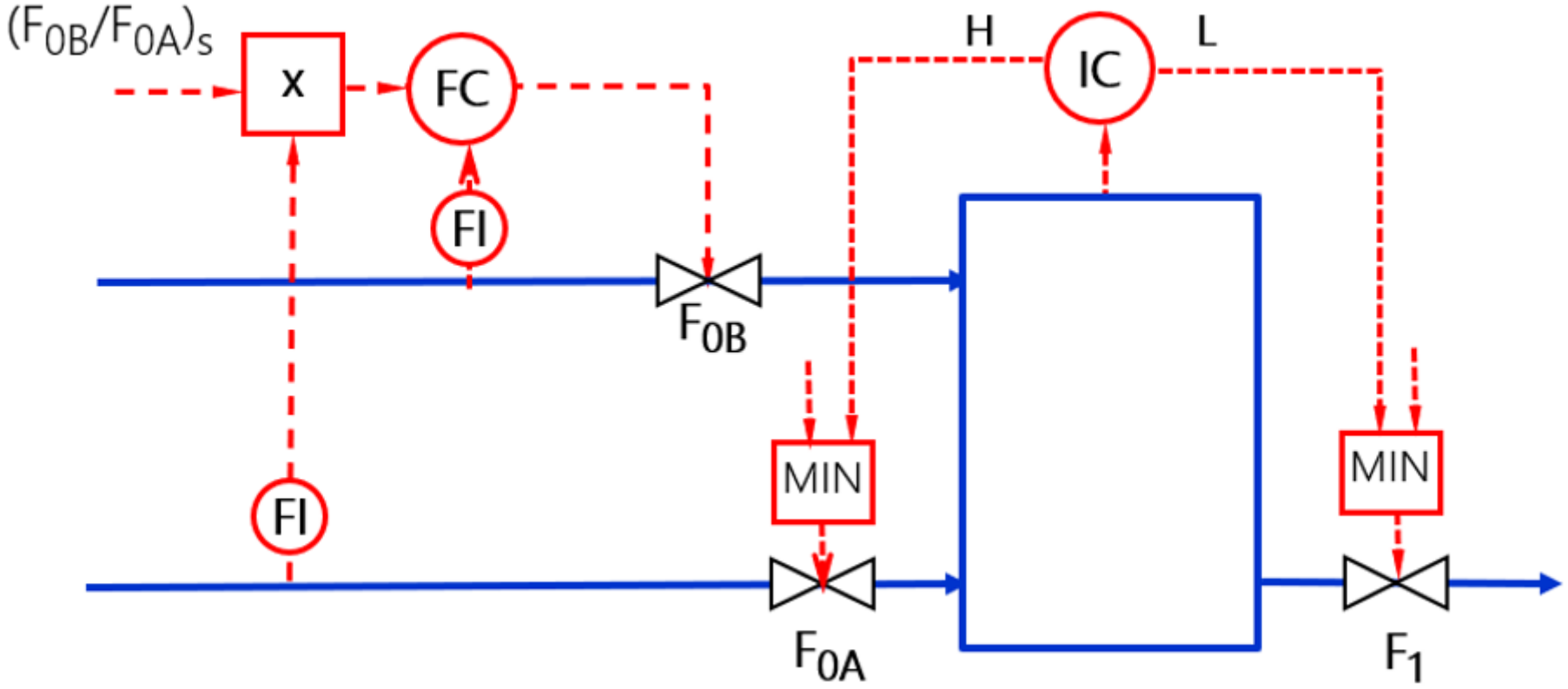
Mixing = Stream merging (junction)

HH=90%
H=80%
L=20%



Bidirectional inventory control

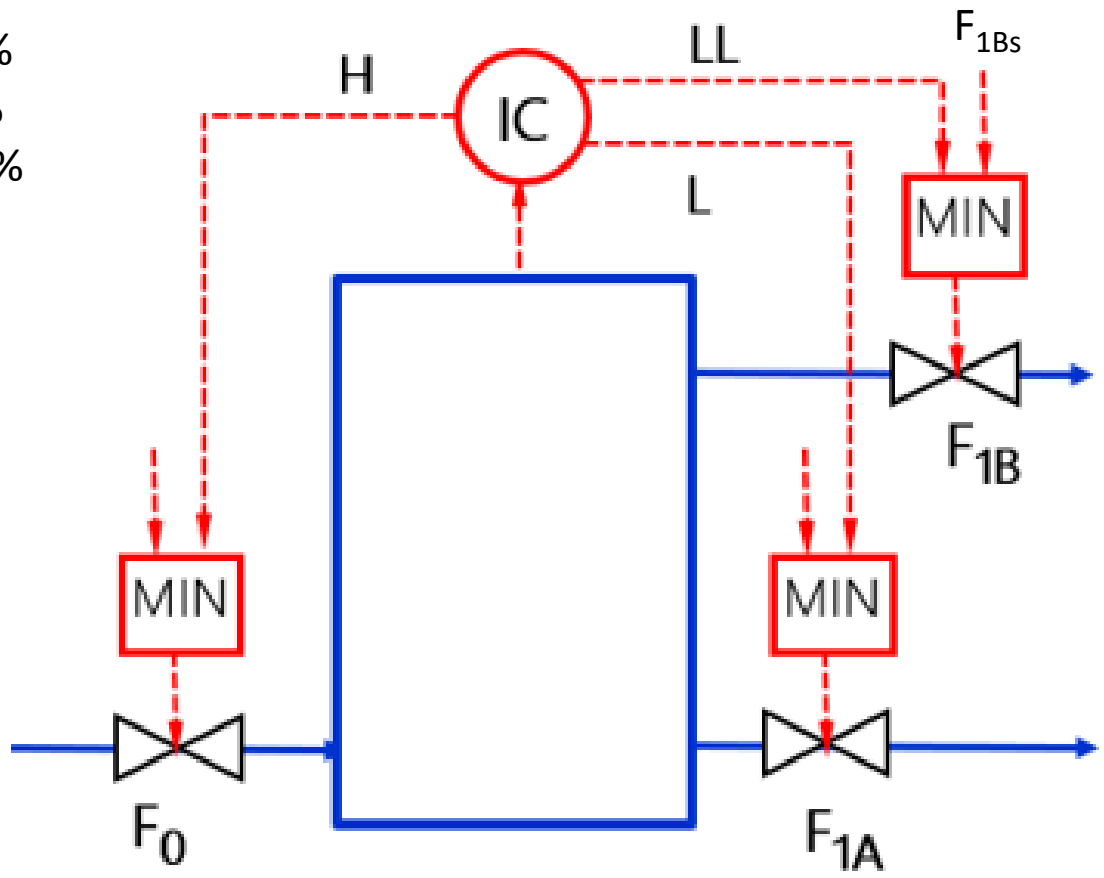
Mixing with ratio control



Bidrectional inventory control

Adjustable split (same composition for F_{1A} and F_{1B})

H=80%
L=20%
LL=10%

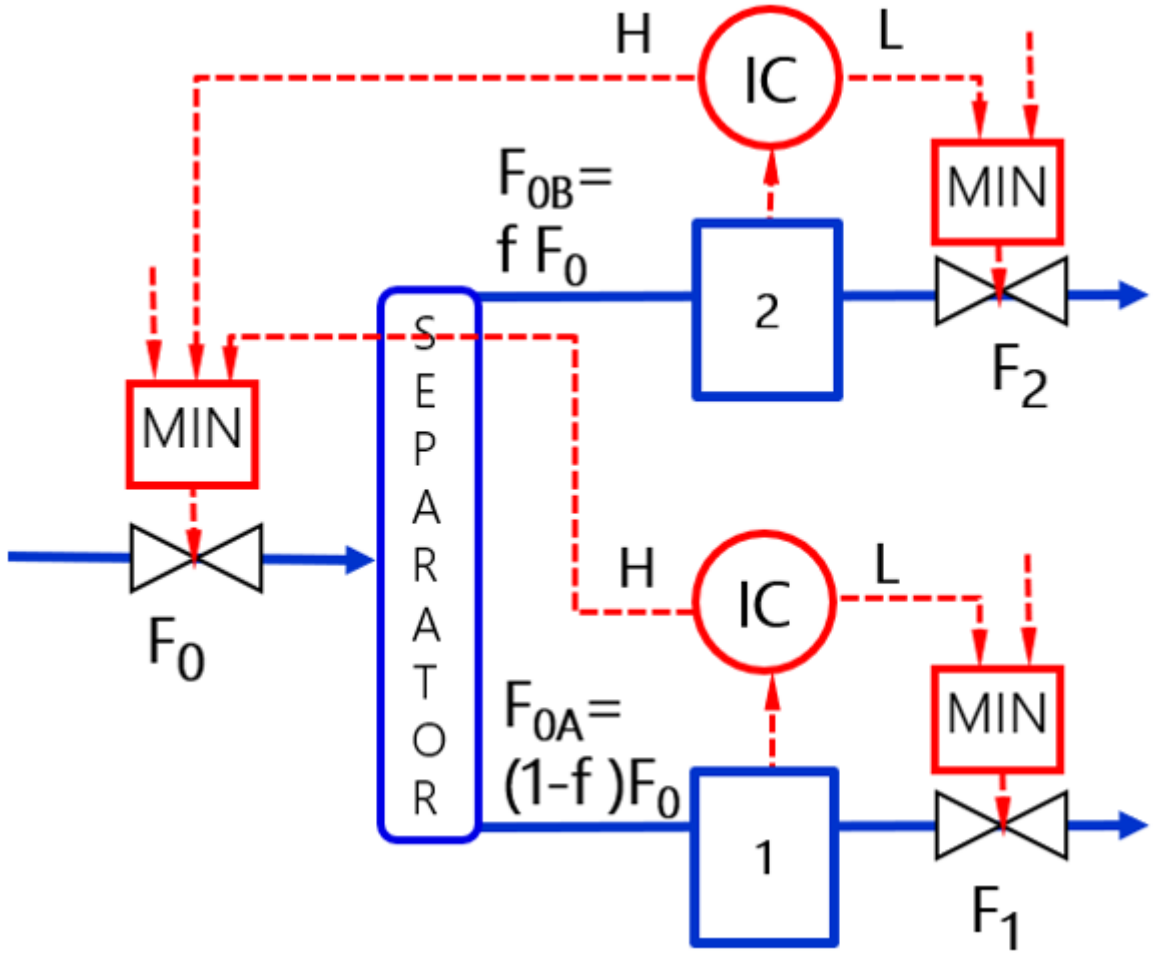


- F_{1Bs} is extra DOF and is normally set.
- But if F_0 is too high and F_{1A} closes (0%), we add LL override

Bidrectional inventory control

Fixed split fraction (separator, different compositions

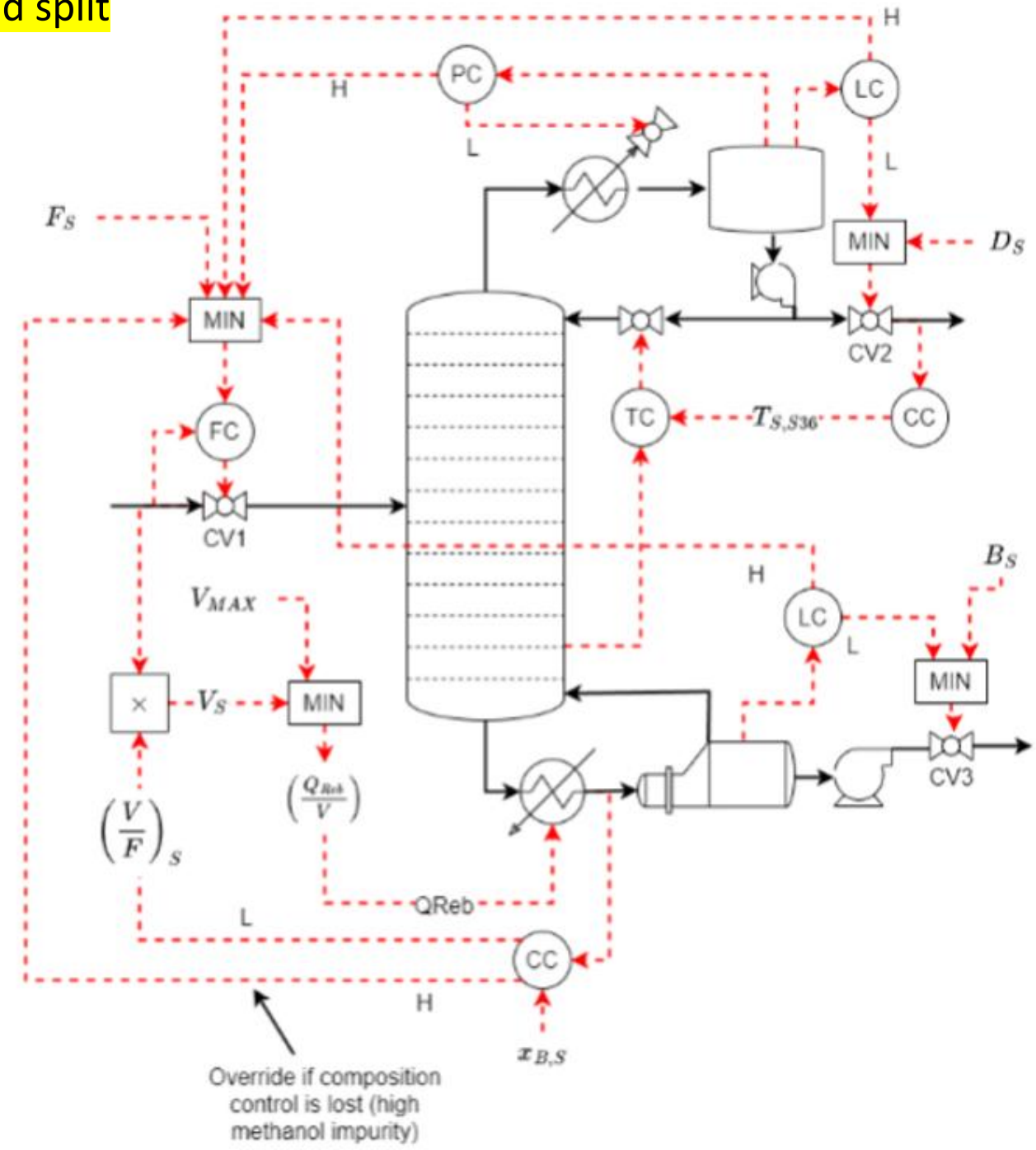
H=80%
L=20%



- For: Distillation, cyclone, filter, crystallizer, phase separator,

Bidirectional inventory control with fixed split Distillation

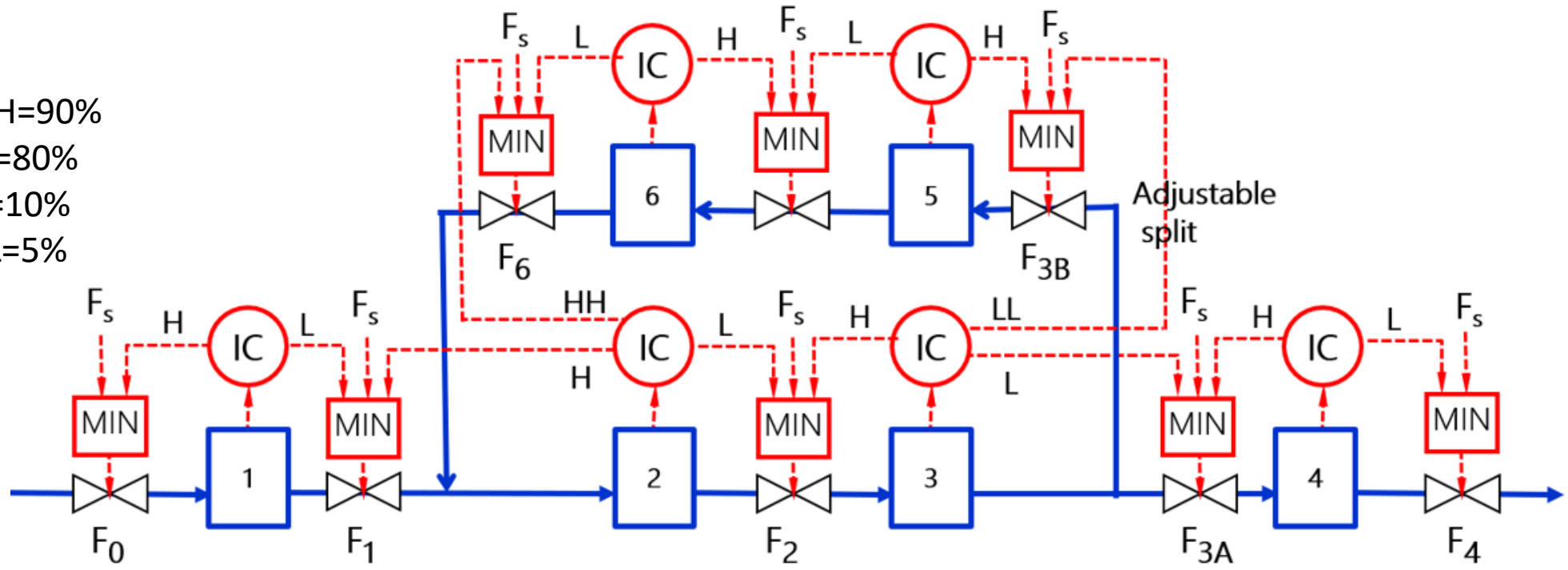
4 H-overrides go back to feed:
LC(top), LC (btm), PC, CC (btm)



Recycle plants

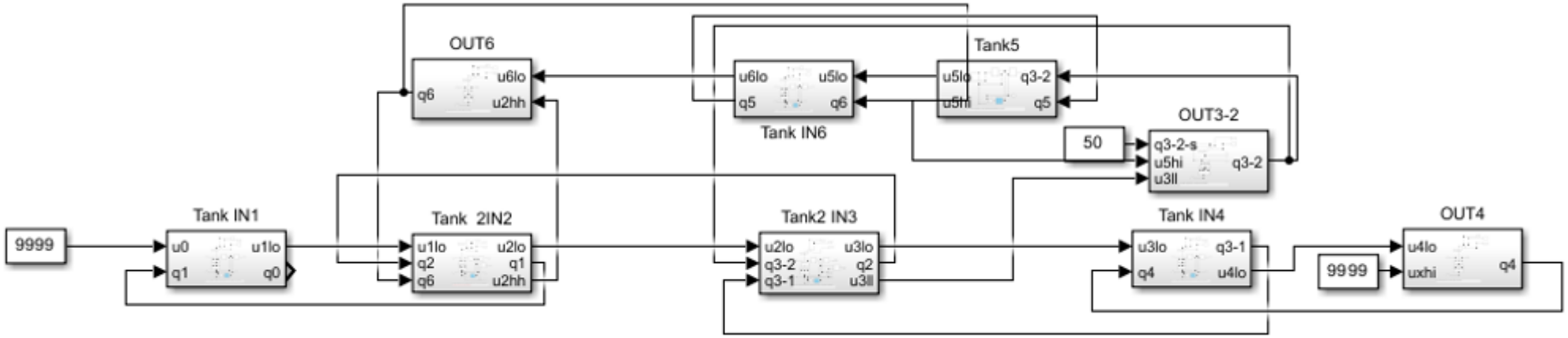
Recycle example with adjustable split (location flexible)

HH=90%
H=80%
L=10%
LL=5%



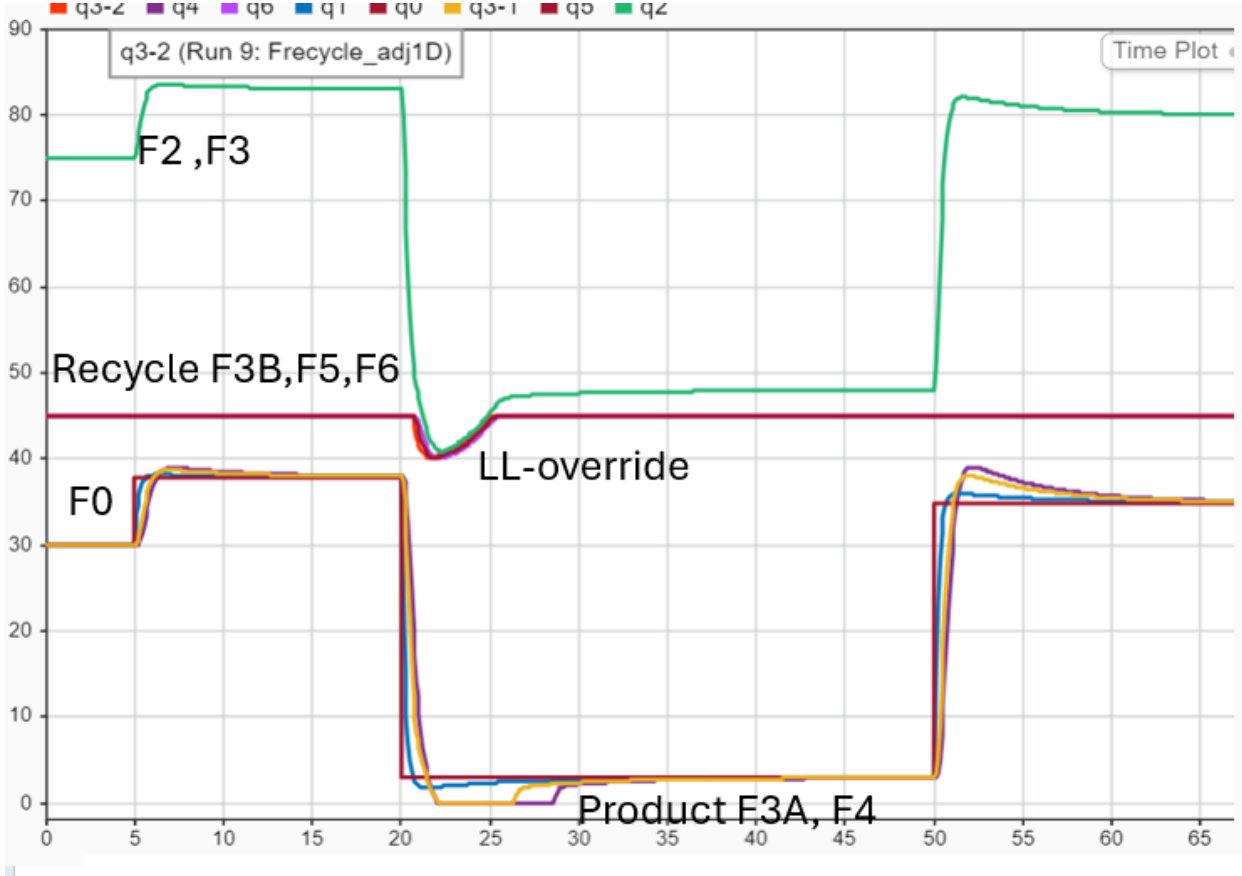
Can set flow (Fs) two places in this network (the system will automatically figure it out)
For consistency the overrides (HH and LL) are on the recycle branch

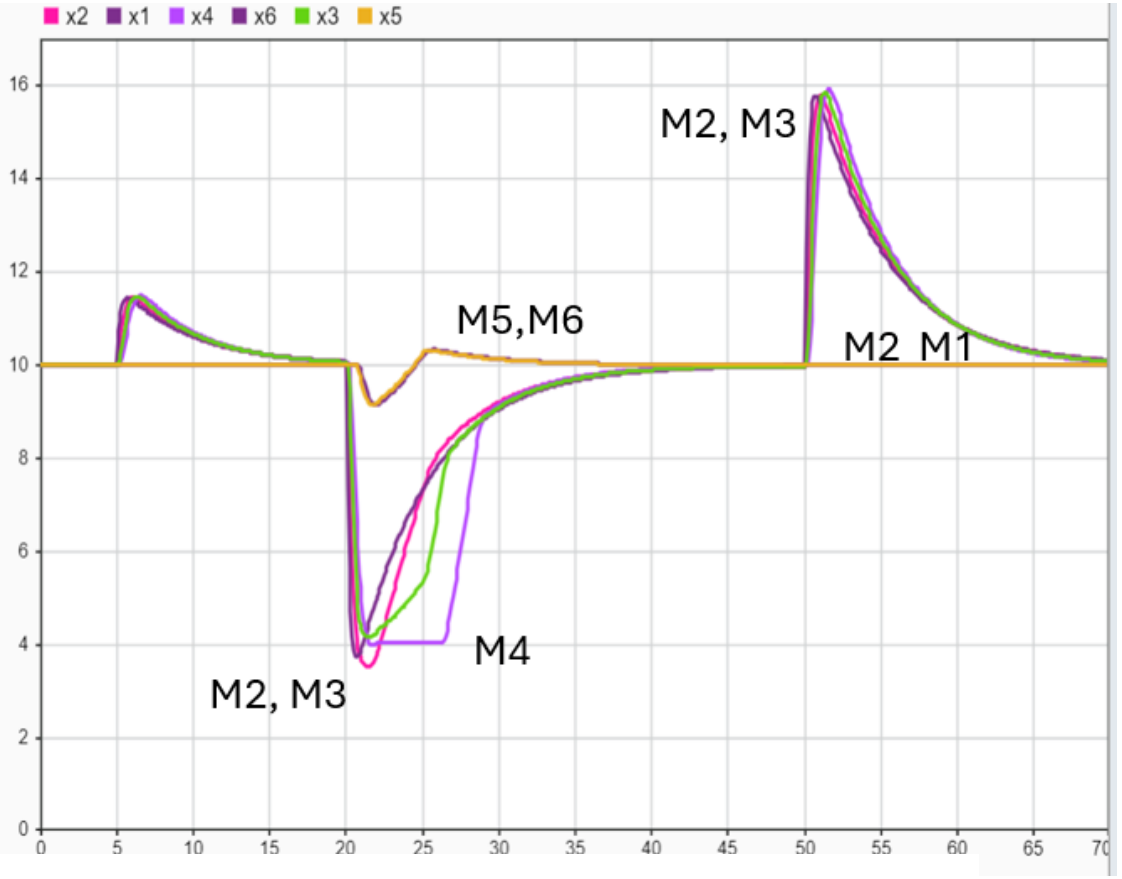
Simulations with Matlab

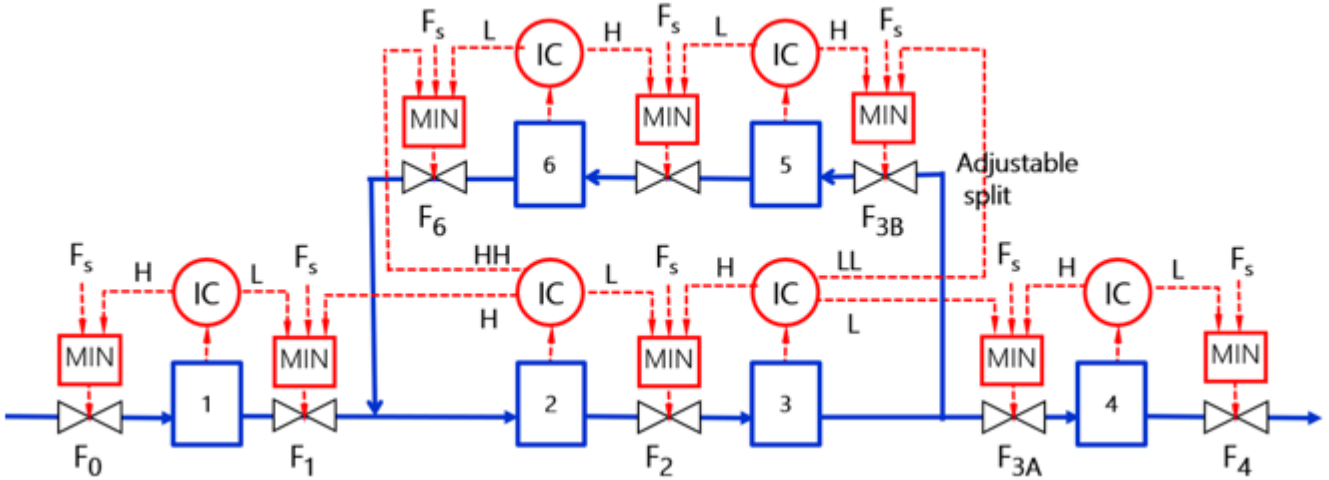


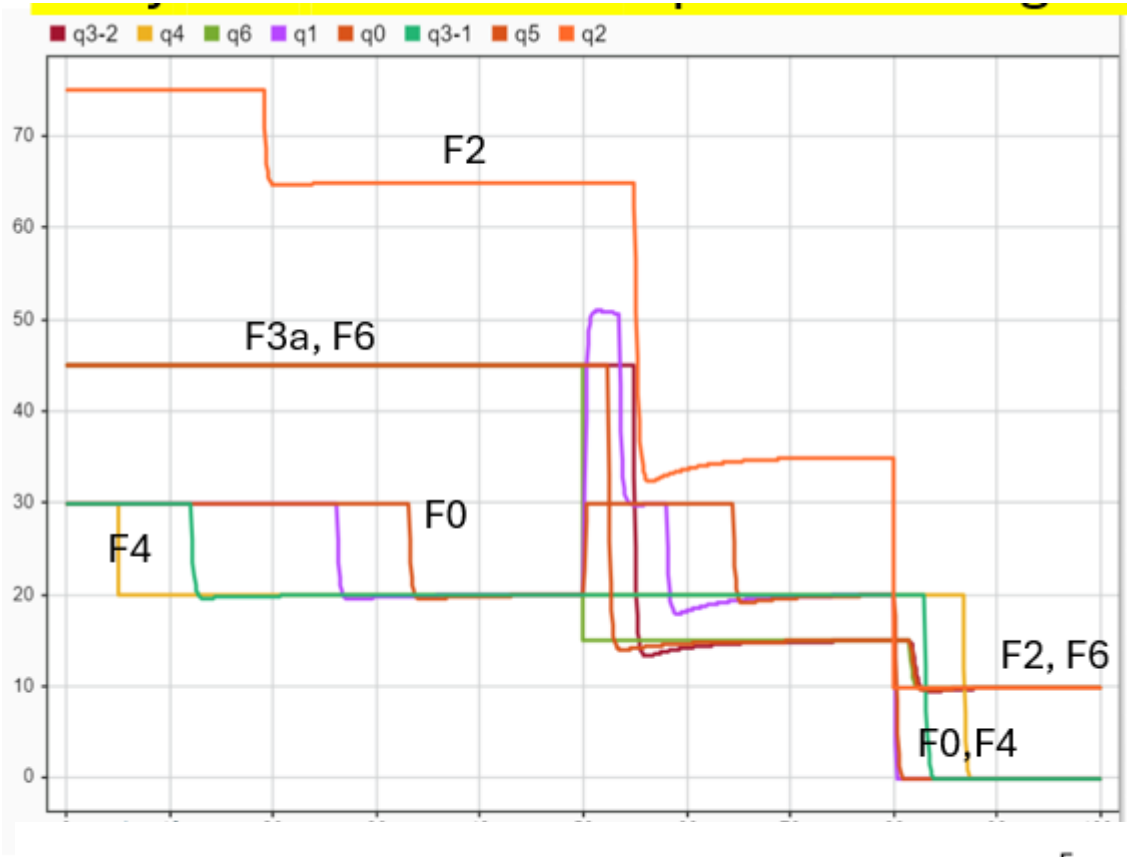
All 14 level controllers (6 H, 6 L, 1 HH, 1 LL):

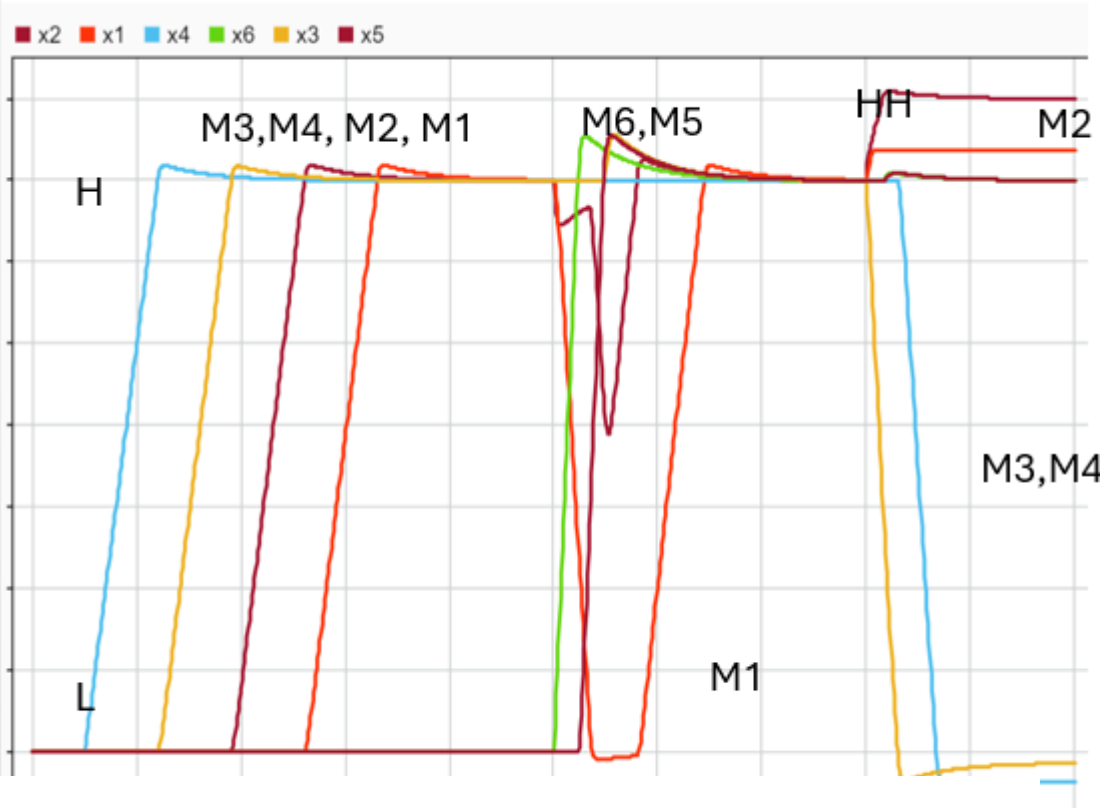
- Kc = 5 %/%
- Integral time, $\tau_I = 5\ \tau$ (where $\tau = M_{max}/F_{max}$ = redidence time =10 min)
  - No oscillations for single tank since Kc $\tau_I = 25\ \tau > 4\ \tau$
- Tracking time for anti-windup, $\tau_T = \tau_I$

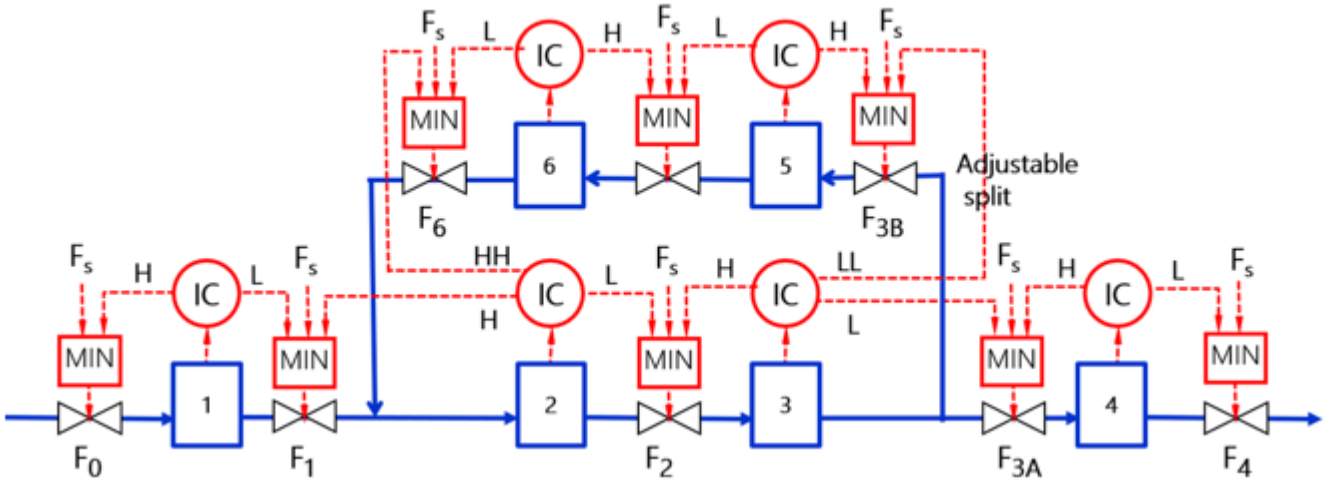

q3-2 (Run 9: Frecycle_adj1D)
Time Plot
F2 ,F3
Recycle F3B,F5,F6
LL-override
F0
Product F3A, F4


x2 x1 x4 x6 x3 x5
M2, M3
M5,M6
M2 M1
M4
M2, M3


IC
MIN
H
L
HH
LL
Adjustable split
F0
F1
F2
F3A
F3B
F4
F6

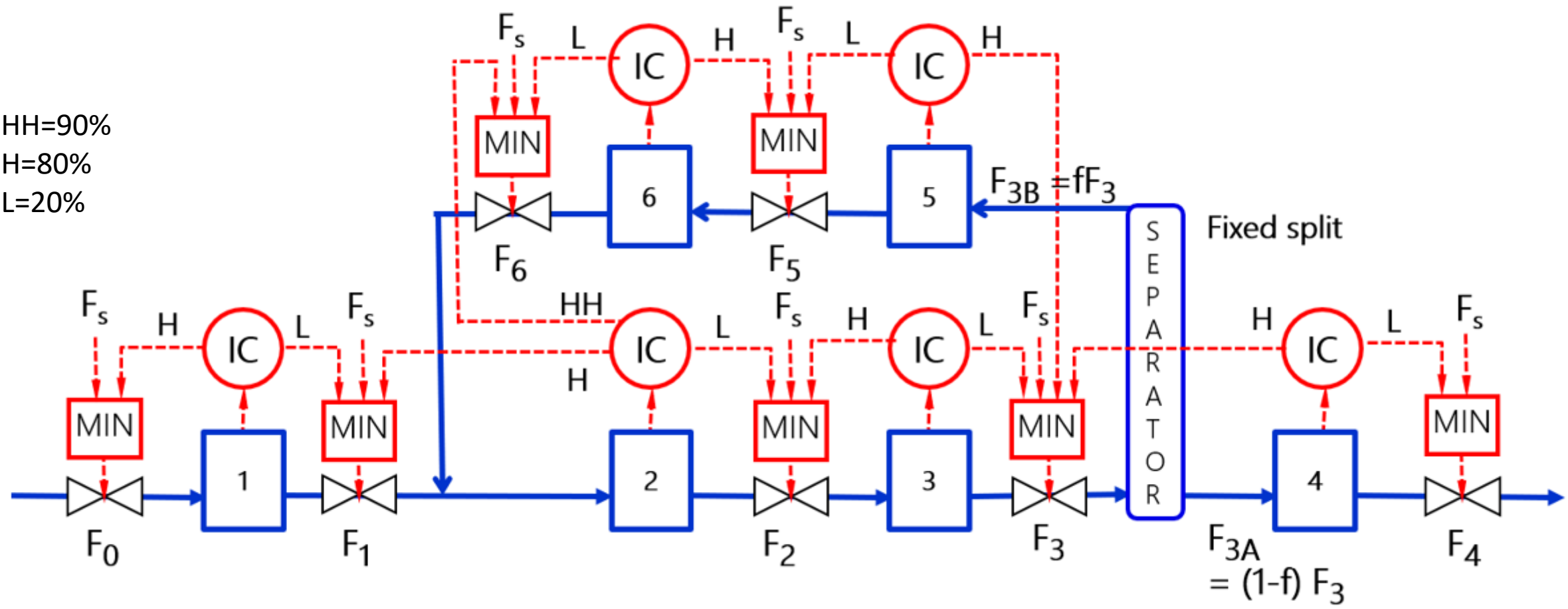

q3-2
q4
q6
q1
q0
q3-1
q5
q2
70
60
50
40
30
20
10
0
F2
F3a, F6
F0
F4
F2, F6
F0,F4


x2
x1
x4
x6
x3
x5
M3,M4, M2, M1
M6,M5
HH
M2
H
M3,M4
L
M1
100

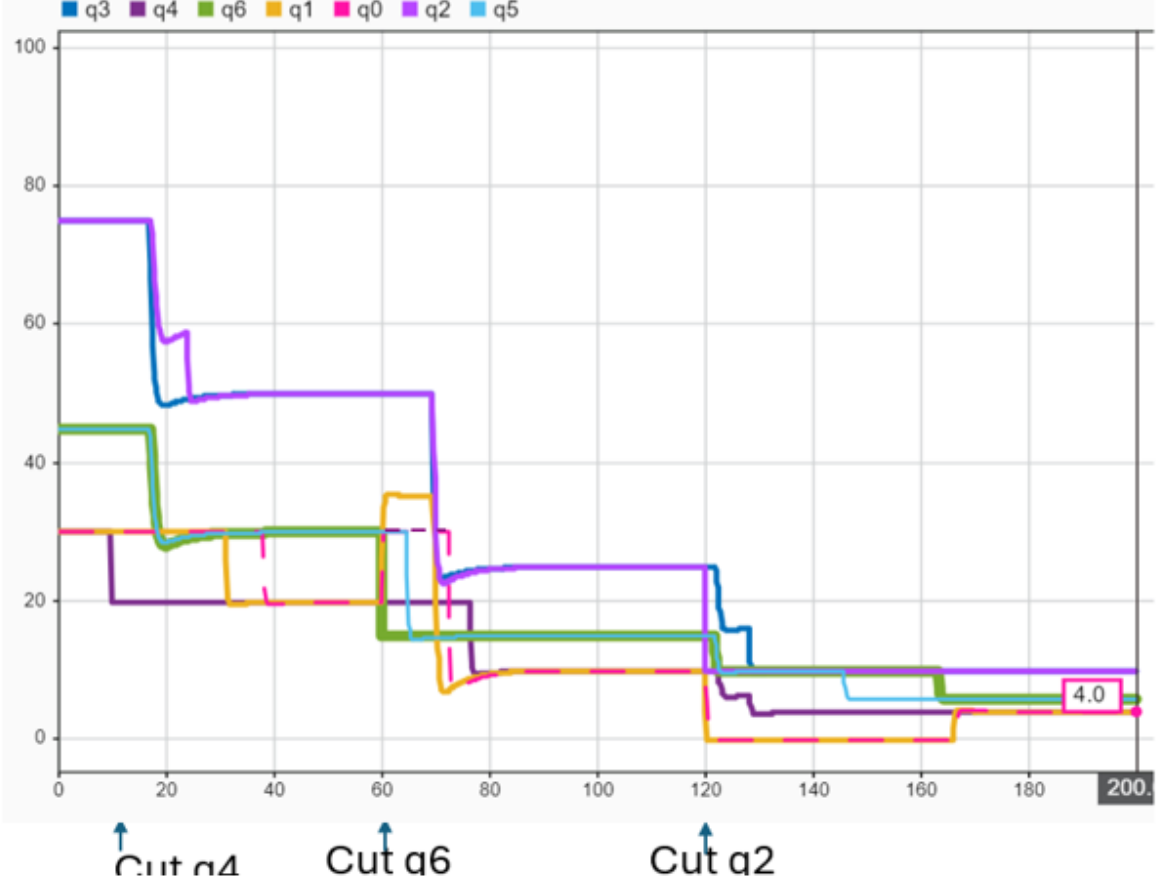

IC
MIN
H
L
HH
LL
Adjustable split
1
2
3
4
5
6
F0
F1
F2
F3A
F3B
F4
F6

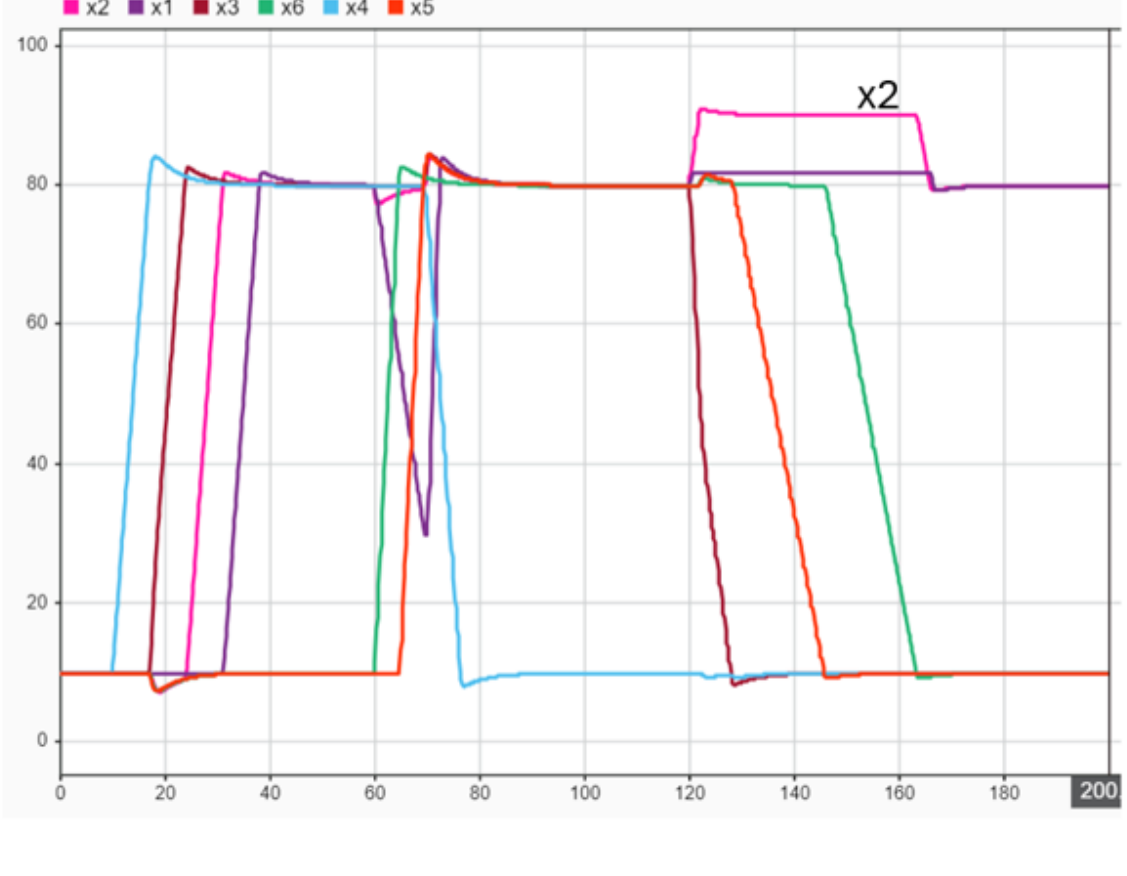
Recycle example with separator (fixed split)

HH=90%
H=80%
L=20%

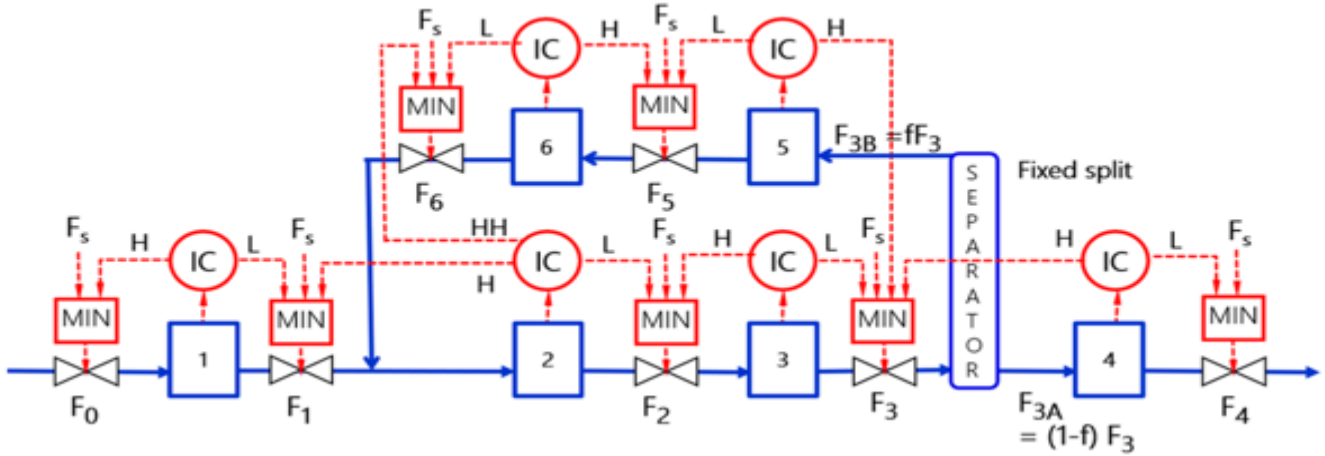


Can set flow (Fs) only one place in this network (f is assumed fixed)





Cut q4 Cut q6 Cut q2



Example bidirectional inventory control

(It uses split-parallel control, but note here that we don't use the inventory for bottleneck isolation)

AIChE

Economic Plantwide Control of the Ethyl Benzene Process

Rahul Jagtap, Ashok S Pathak, and Nitin Kaistha
Dept. of Chemical Engineering, Indian Institute of Technology Kanpur, Kanpur 208016,Uttar Pradesh, India

DOI 10.1002/aic.13964
Published online December 10, 2012 in Wiley Online Library (wileyonlinelibrary.com).

A1: Benzene
A2: Ethylene
B: Ethylbenzene (product)
C: Diethylbenzene (undersired)
A1+A2 →B
B + A2 → C
C + A1 → 2B

LS = MIN

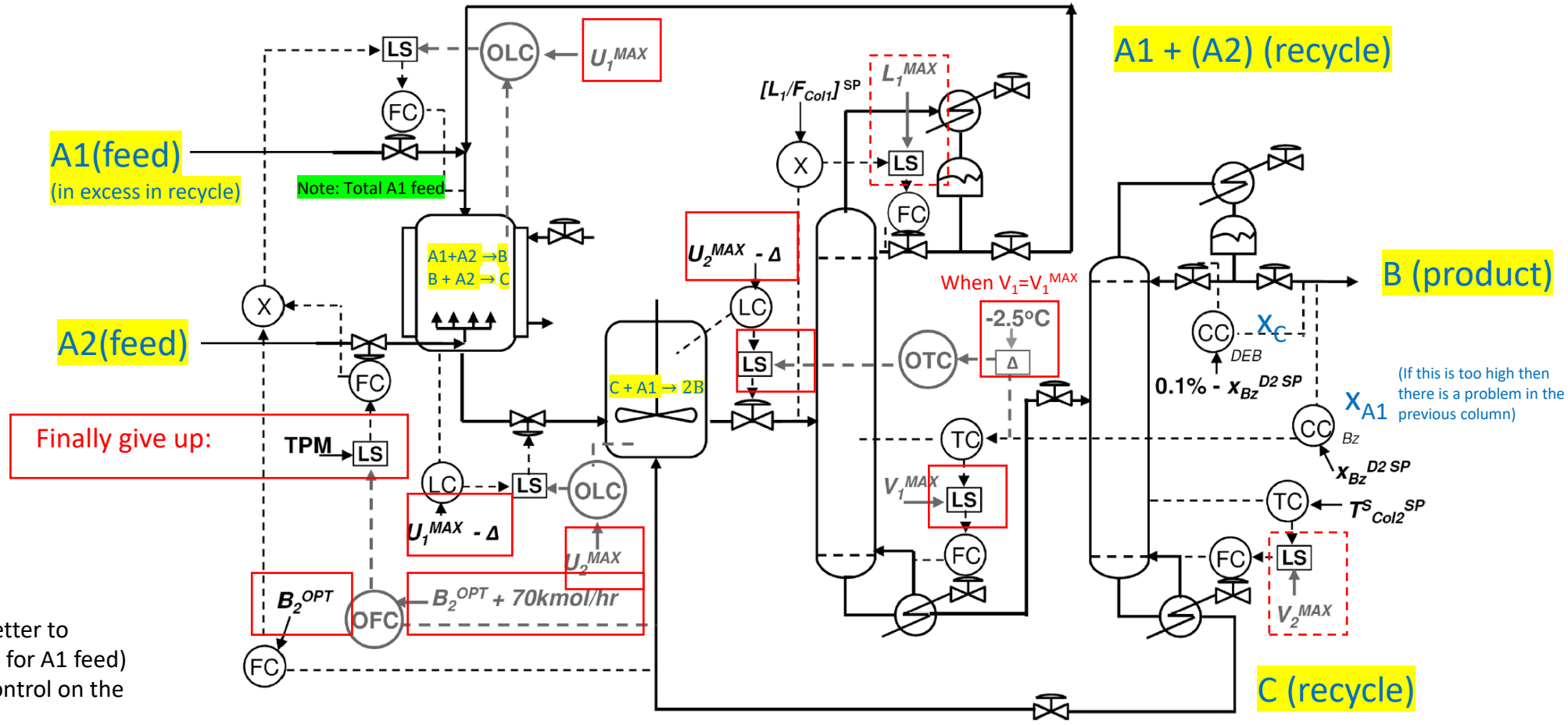


Figure 7. CS2 with overrides for handling equipment capacity constraints.

Comment 22 May 2025: Better to «dual» ratio control (for FC for A1 feed) so you always have ratio control on the fast time scale

Example bidirectional inventory control

AIChE

Economic Plantwide Control of the Ethyl Benzene Process

Rahul Jagtap, Ashok S Pathak, and Nitin Kaistha
Dept. of Chemical Engineering, Indian Institute of Technology Kanpur, Kanpur 208016,Uttar Pradesh, India

DOI 10.1002/aic.13964
Published online December 10, 2012 in Wiley Online Library (wileyonlinelibrary.com).

A1: Benzene
A2: Ethylene
B: Ethylbenzene (product)
C: Diethylbenzene (undersired)
A1+A2 →B
B + A2 → C
C + A1 → 2B

LS = MIN

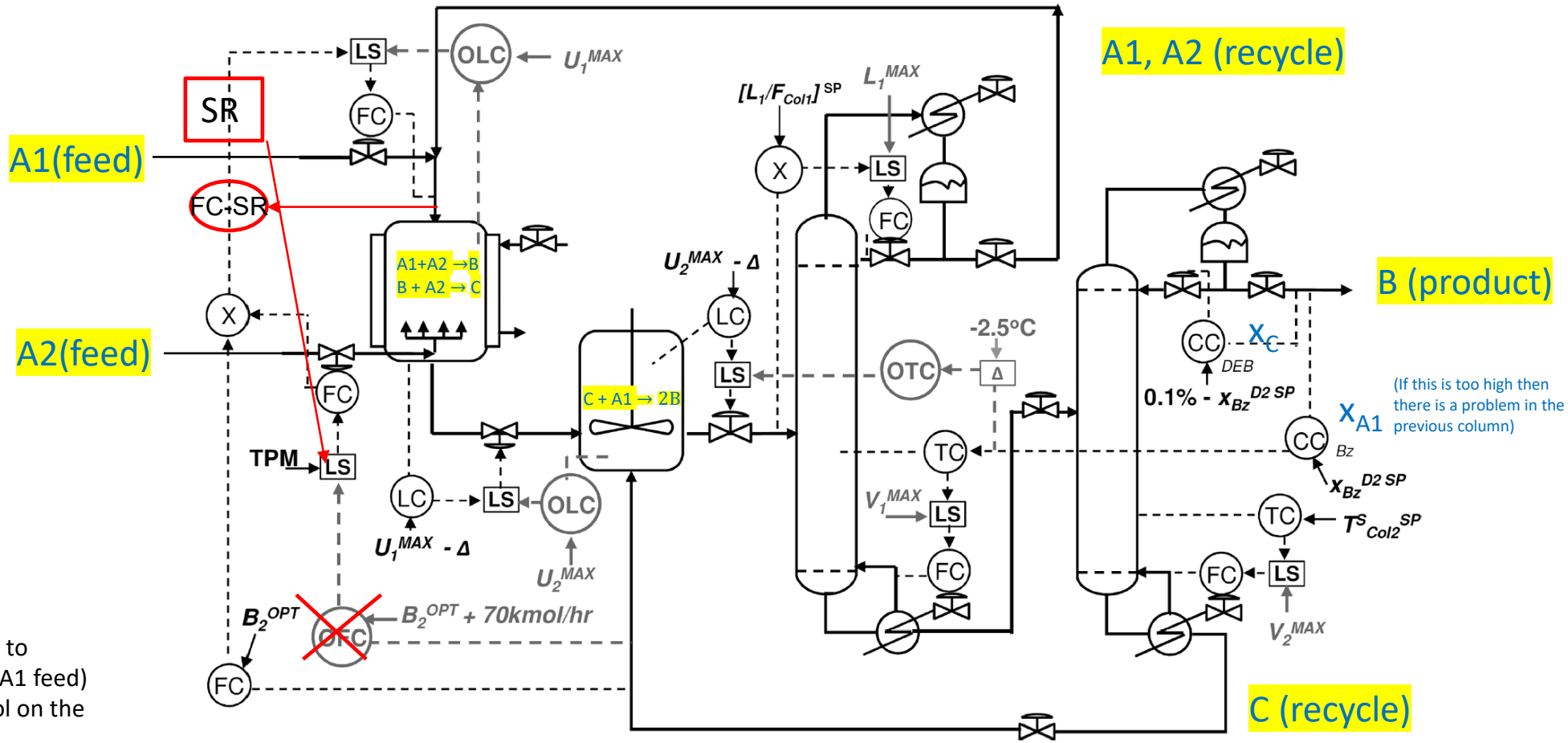
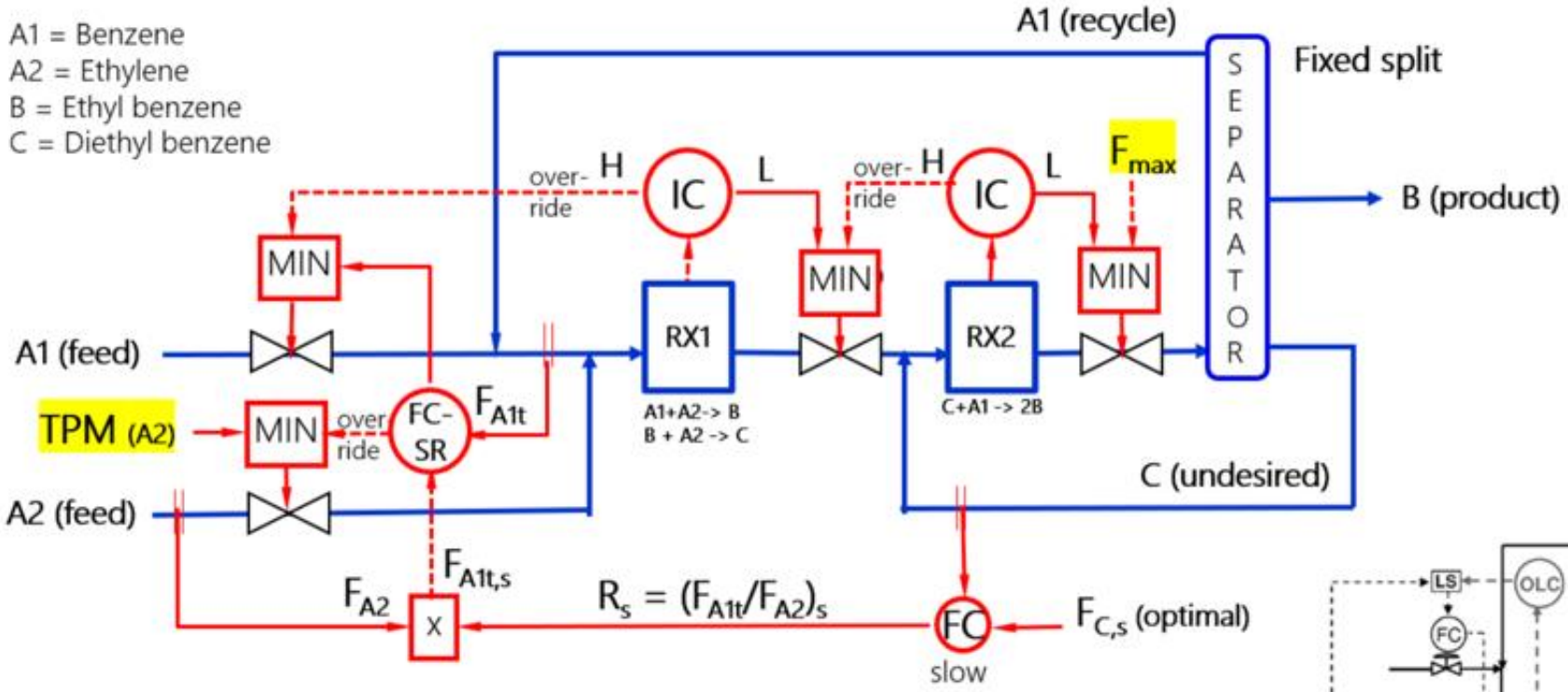


Figure 7. CS2 with overrides for handling equipment capacity constraints.

Comment 22 May 2025: Better to «dual» ratio control (for FC for A1 feed) so you always have ratio control on the fast time scale .
BUT get two FCs in series ? See new FC-SR. Should be OK (see next slide)

A1 = Benzene
A2 = Ethylene
B = Ethyl benzene
C = Diethyl benzene



nax is reduced when we reach Vmax in separator (column), and then overrides are activated.
itio control is maintained as a fast loop under all conditions. The FC for C=DEB is slow.

Economic Plantwide Control of the Ethyl Benzene Process

Rahul Jagtap, Ashok S Pathak, and Nitin Kaistha
Dept. of Chemical Engineering, Indian Institute of Technology Kanpur, Kanpur 208016,Uttar Pradesh, India

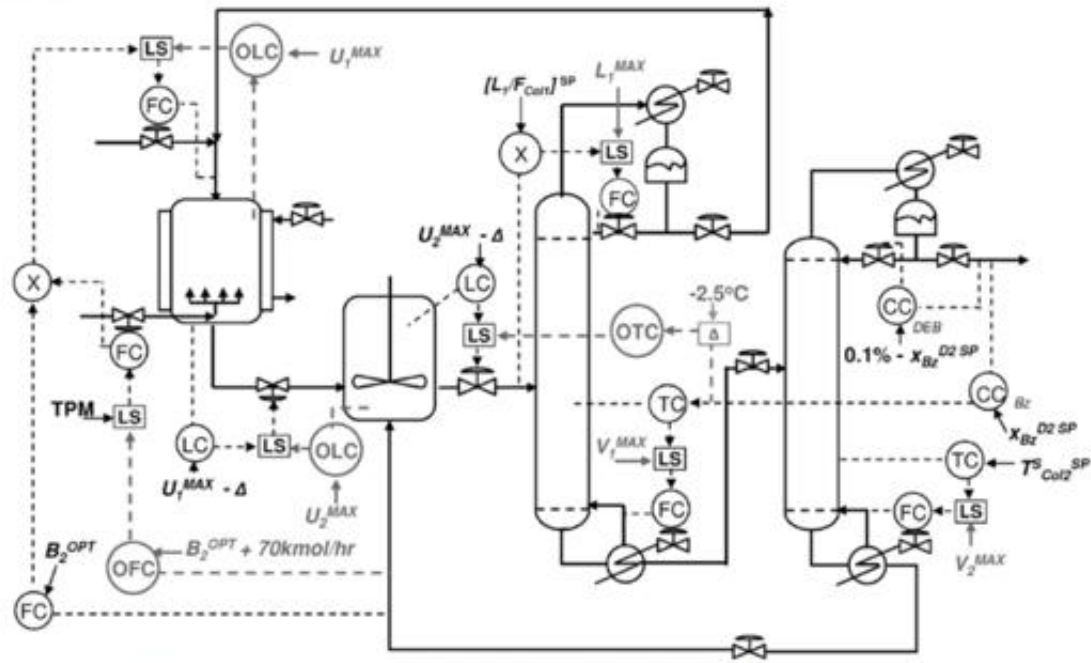


Figure 7. CS2 with overrides for handling equipment capacity constraints.

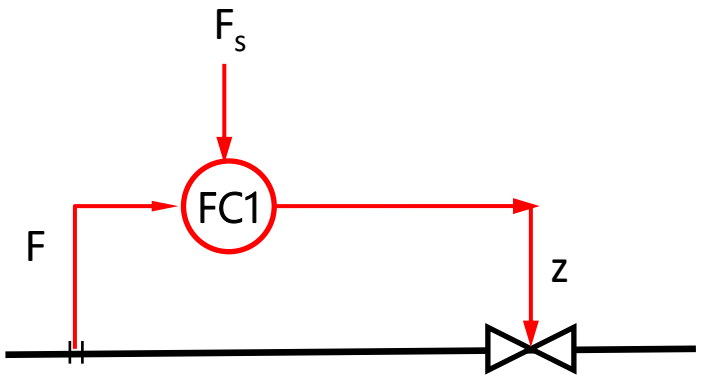
Implementing optimal operation Summary

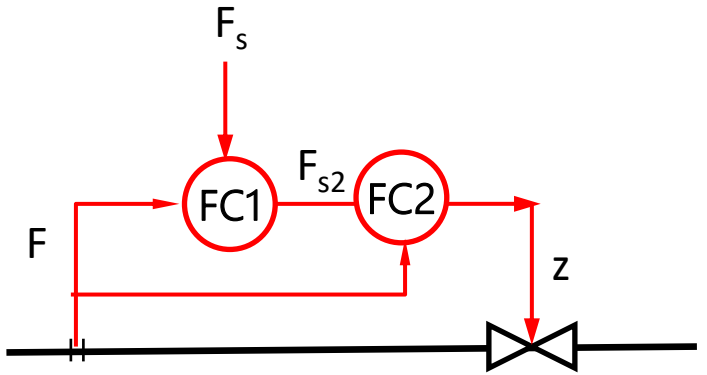
- Most people think
  - You need a detailed nonlinear model and an on-line optimizer (RTO) if you want to optimize the process
  - You need a dynamic model and model predictive control (MPC) if you want to handle constraints
  - The alternative is Machine Learning
- No! In many cases you just need to measure the constraints and use PID control
  - «Conventional advanced regulatory control (ARC)»
- How can this be possible?
  - Because optimal operation is usually at constraints
  - Feedback with PID-controllers can be used to identify and control the active constraints
  - For unconstrained degrees of freedom, one often have «self-optimizing» variables

- This fact is not well known, even to control professors
  - Because most ARC-applications are *ad hoc*
  - Few systematic design methods exists

- Today ARC and MPC are in parallel universes
  - Both are needed in the control engineer's toolbox

Can we have two controllers (here FCs) in cascade (controlling the same variable)? (Yes)





Assume static linear valve with gain 1:

F=z (process).

Only dynamics come from I-controller FC1:

$$z = \frac{1}{\tau_{I1} s} (F_s - F)$$

Combine:

$$F = \frac{1}{\tau_I 1 s + 1} F_s$$

Two I-controllers for F (assume FC2 is fast):

$$F_{s2} = \frac{1}{\tau_{I1} s} (F_s - F); \quad z = \frac{1}{\tau_{I2} s} (F_{s2} - F)$$

Use F=z and eliminate F_{s2}:

$$F = \frac{1}{\tau_{I1}\, \tau_{I2} s^2 + \tau_{I1} s + 1} F_s$$

If $\tau_{I1} \gg \tau_{I2}$:

$$F \approx \frac{1}{(\tau_{I1}\, s + 1)(\tau_{I2}\, s + 1)} F_s$$

OK, so it's possible… but necessarily a good idea

Comments on Inventory control (level, pressure)

- All inventories (level, pressure) must be regulated by
  - Controller, or
  - “self-regulated” (e.g., overflow for level, open valve for pressure)
  - Exception closed system: Must leave one inventory (level) uncontrolled

- Usually only one TPM
  - To get consistent mass balance: Can only fix same flow once
  - But there are exceptions
    - Multiple feeds (they are then usually set in ratio to the “main” TPM)
    - Recycle systems often have a flow that can be set freely

- Rule for maximizing production for cases where we cannot rearrange inventory loops (that is, we don’t use bidirectional inventory control): Locate TPM at expected bottleneck
  - Otherwise you will need a “long loop” and you get loss in production because of backoff from constraint

Summary: Systematic design of advanced regulatory control (ARC) system

- First design simple control system for nominal operation
  - With single-loop PID control we need to make pairing between inputs (MVs) and outputs (CVs):
  - Should try to follow two rules
    1. «Pair close rule» (for dynamics).
    2. «Input saturation rule»:

Then: design of switching schemes

- Make a list of possible new contraints that may be encountered (because of disturbances, parameter changes, price changes)
- Reach constraint on new CV
  - Simplest: Find an unused input (simple MV-CV switching)
  - Otherwise: CV-CV switching using selector (may involve giving up a CV-constraint or a self-optimizing CV)
- Reach constraint on MV (which is used to control a CV)
  - Simplest (If we followed input saturation rule):
    - Can give ip controlling the CV (Simple MV-CV switching)
    - Don't ned to do anything
  - Otherwise (if we cannot give up controlling CV)
    - Simplest: Find an unused input
      - MV-MV switching
    - Otherwise: Pair with a MV that already controls another CV
      - Complex MV-CV switching
      - Must combine MV-MV and CV-CV switching
- Is this always possible? No, pairing inputs and outputs may be impossible with many constraints.
- May then instead use RTO or feedback-RTO
- Maybe MPC?

Here is a summary of some additional insights from this paper:

- If the industrial solution has a selector (sometimes realized using a saturation element, especially for the cascade implementation) then generally there is a CV constraint involved. Most likely, the selector is performing a steady-state CV-CV switch (E4), although there may be exceptions as seen in the cross-limiting example below.
  - A CV-CV switch can be realized in two ways, either with two (or more) independent controllers with a selector on the MV (Fig. 17), or as a cascade implementation with a selector on the CV setpoint (Fig. 19).
  - If there are several selectors (max and min) in series then we know that the constraints are potentially conflicting and that the highest priority constraint should be at the end (Fig. 18).

- If the industrial solution has a valve position controller (VPC) then there may be two quite different problems that it is addressing (see E3 and E7 in Table 1), and it may not be immediately clear which.
  1. If we have an extra MV for dynamic reasons (E3; Fig. 12) then the two controllers (and MVs) are used all the time. The MV manipulated by the VPC (MV_1 in Fig. 12) is then used on the longer time scale, whereas the MV linked to the CV (MV_2 in Fig. 12) is used for dynamic reasons (fast control). Here, an alternative is to use parallel control (Fig. 13).
  2. There is also another possibility, namely, when the VPC makes use of an extra MV to avoid that the primary MV saturates at steady-state (E7; Fig. 24). This is then a case where the VPC is used for MV-MV switching and the VPC is only active part of the time.

- For MV-MV switching there are three alternatives.
  1. A common solution is split range control (E5; Fig. 21) which is usually easy to identify.
  2. Another common solution is multiple controllers with different setpoints (E6; Fig. 23). It may be a bit more difficult to identify.
  3. Finally, there is VPC (E7), as just discussed, which is probably the least common solution for MV-MV switching

One should have all these three alternatives in mind when choosing the best solution for MV-MV switching, as there is not one alternative which is best for all problems (see Section 5.1 for details).

Example adaptive cruise control: CV-CV switch followed by MV-MV switch

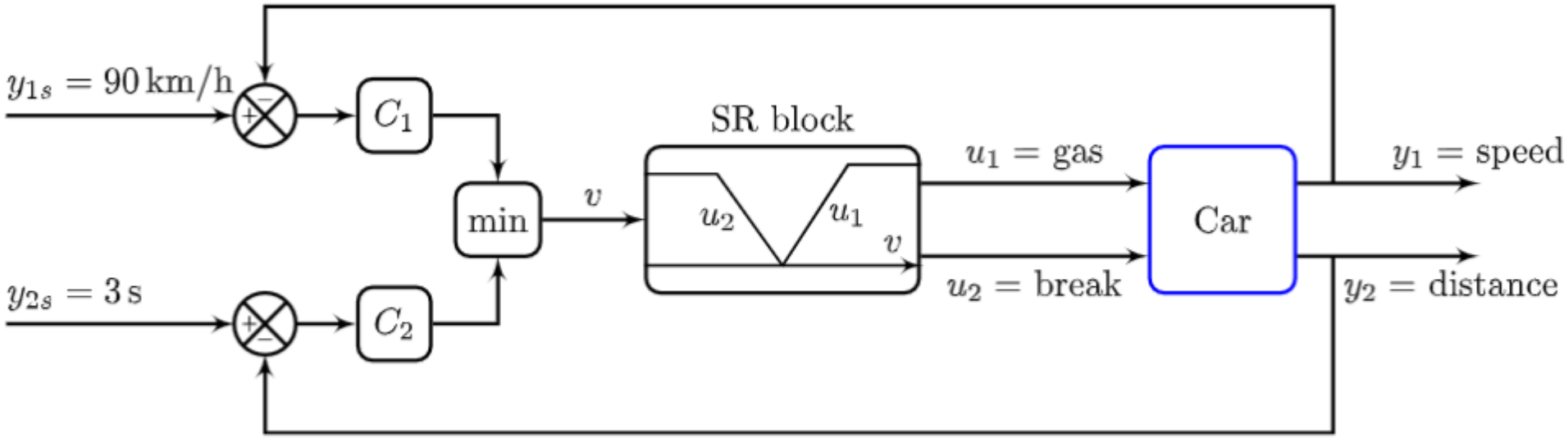


Fig. 31. Adaptive cruise control with selector and split range control.

Note: This is not Complex MV-CV switching, because then the order would be opposite.