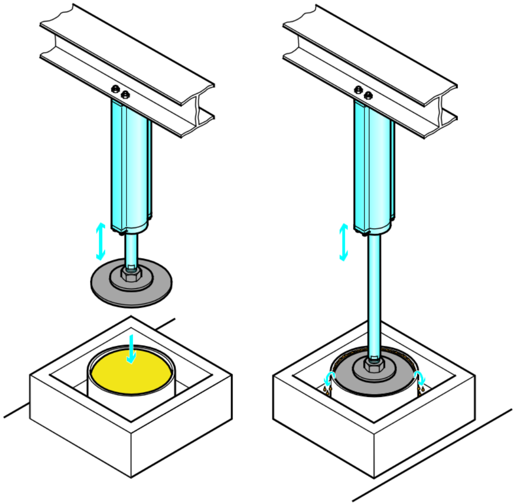
Logic control of electro-pneumatic systems

Kjartan Halvorsen

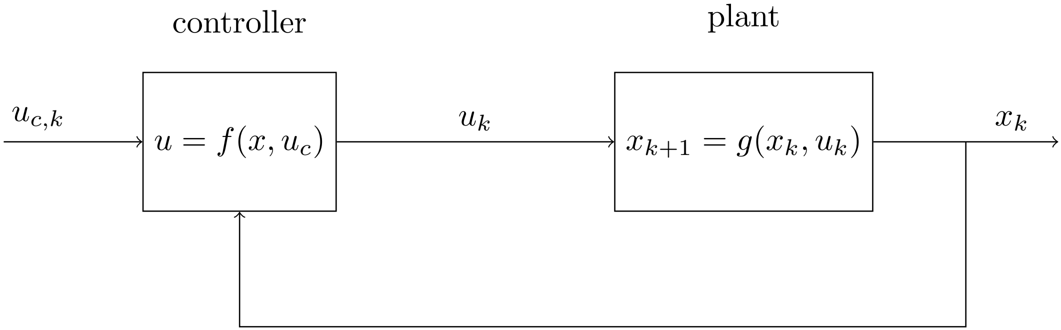
May 14, 2020

Cheese pressing example, sequence A+A-



From FESTO Didactic

A logic control loop

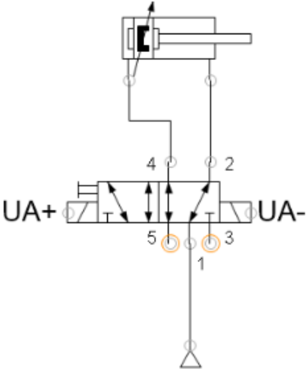


Cheese pressing example - Variables

Activating solenoid UA+ extends the cylinder, activating UA- retracts the cylinder.

State variable

$$x = \begin{cases} 0 & \text{Cylinder retracted} \\ 1 & \text{Cylinder extended} \end{cases}$$



Control signal

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix},$$

with

$$u_1 = \begin{cases} 0 & \text{Don't activate UA+} \\ 1 & \text{Activate UA+} \end{cases}$$

$$u_2 = \begin{cases} 0 & \text{Don't activate UA-} \\ 1 & \text{Activate UA-} \end{cases}$$

Command signal

$$u_c = \begin{cases} 0 & \text{Button unpushed} \\ 1 & \text{Button pushed} \end{cases}.$$

Cheese pressing example - Plant dynamics and control law

Activating solenoid UA+ extends the cylinder, activating UA- retracts the cylinder.

Plant dynamics $x_{k+1} = g(x_k, u_k)$

		state	
$u_{1,k}$	$u_{2,k}$	x_k	x_{k+1}
0	0	0	0
0	1	0	0
1	0	0	1
1	1	0	0
0	0	1	1
0	1	1	0
1	0	1	1
1	1	1	1

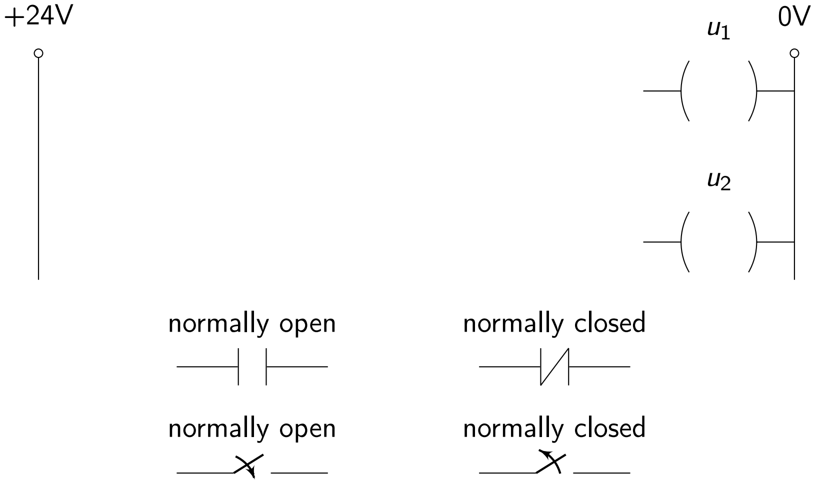
Control law $u_k = f(x, u_c)$

x	u_c	u_1	u_2
0	0	0	0
1	0	0	1
0	1	1	0
1	1	0	1

$$u_1 =$$

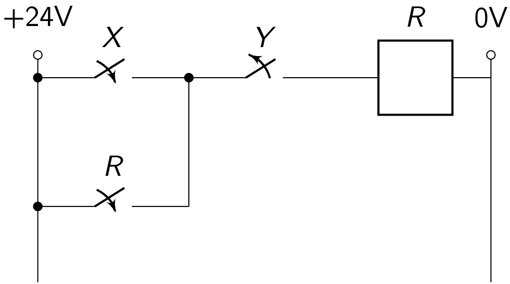
$$u_2 =$$

Cheese pressing example - implementing the control law



Intermezzo - An electrical circuit with memory

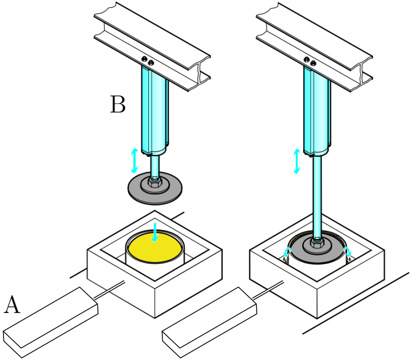
Latching circuit

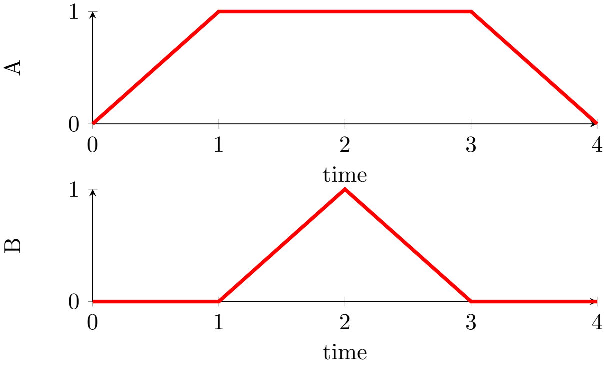


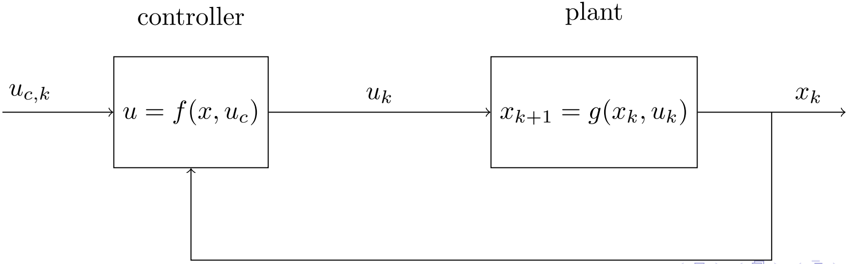
Truth table

X	Y	R_k	R_{k+1}
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

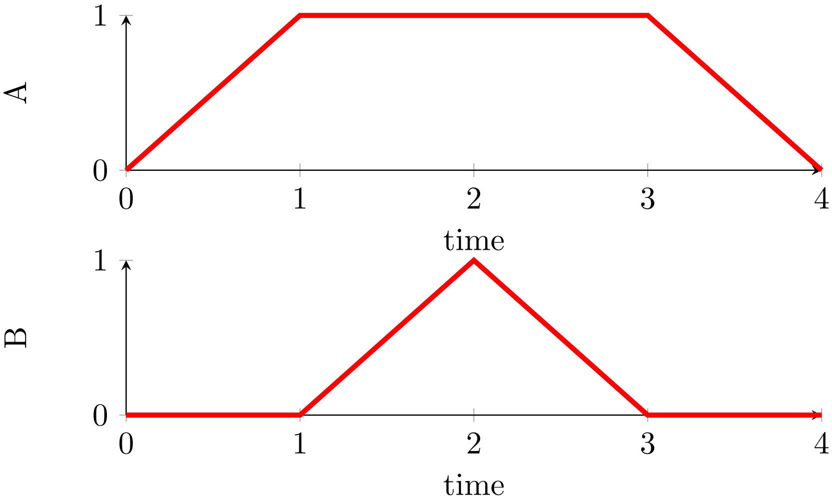
The lab assignment



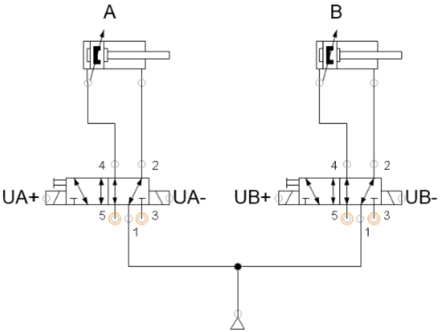


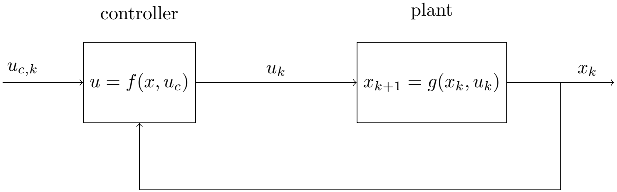


Implementing the sequence A+B+B-A-



Implementing the sequence A+B+B-A-, control signal





Control signal

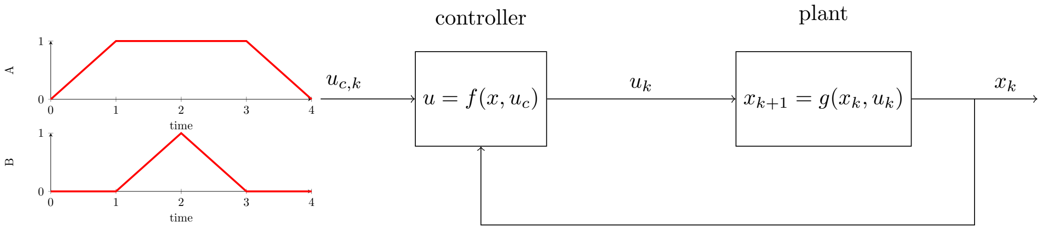
$$u = \begin{bmatrix} u_A+ & u_A- & u_B+ & u_B- \end{bmatrix}^T,$$

with

$$u_A+ = \begin{cases} 0 & \text{Solenoid extending A is not activated} \\ 1 & \text{Solenoid extending A is activated} \end{cases}$$

etc.

Implementing the sequence A+B+B-A-, state variables



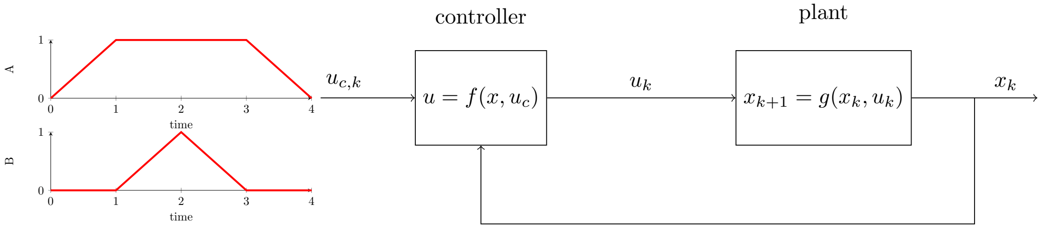
State variables (naive)

$$x = \begin{bmatrix} x_A & x_B \end{bmatrix}^T,$$

with

$$x_{\{A,B\}} = \begin{cases} 0 & \text{Cylinder \{A,B\} retracted} \\ 1 & \text{Cylinder \{A,B\} extended} \end{cases}$$

Implementing the sequence A+B+B-A-, control law

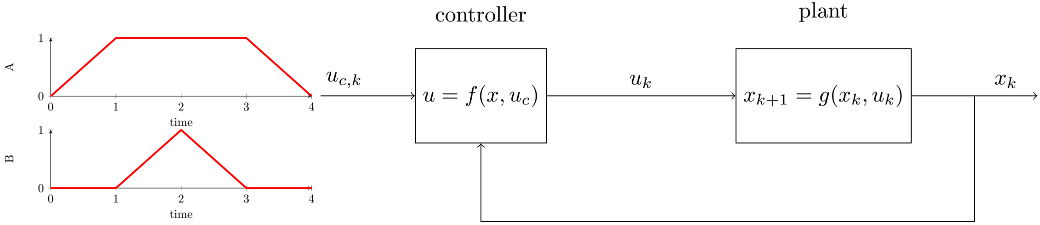


Control law (problematic)

Ignoring input signal u_c (no start/stop buttons). Movement should be cyclic

x_A	x_B	u_A+	u_A-	u_B+	u_B-
0	0				
1	0				
1	1				
0	1				

Implementing the sequence A+B+B-A-, control law



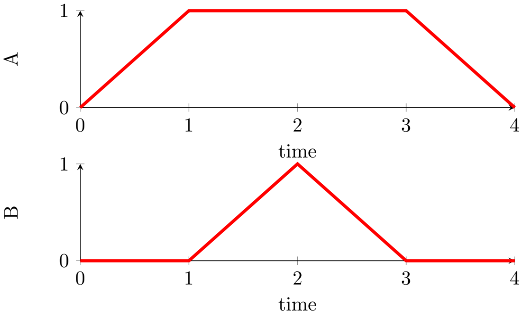
Control law (problematic)

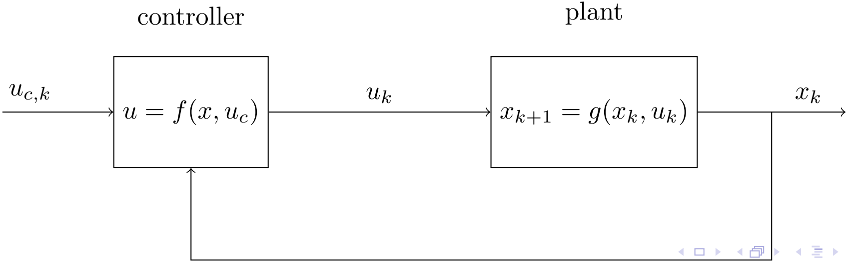
Ignoring input signal u_c. Movement should be cyclic

x_A	x_B	u_A+	u_A-	u_B+	u_B-
0	0	1	0	0	0
1	0	0	1 or 0	0 or 1	0
1	1	0	0	0	1
(0)	(1)	0	0	0	1

Implementing the sequence A+B+B-A-, the problem

The correct control signal (action) is not uniquely defined by the position of the cylinders

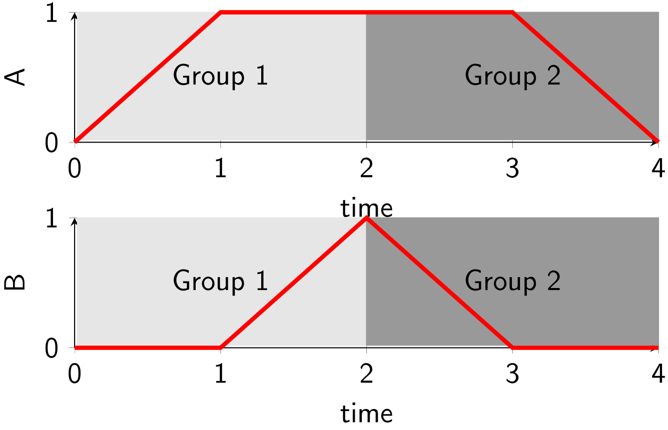




Implementing the sequence A+B+|B-A-

Dividing the sequence into groups (a.k.a. cascade method)

$$\underbrace{A+B+}_{\text{Group 1}}\,|\,\underbrace{B\text{-}A\text{-}}_{\text{Group 2}}$$



Implementing the sequence A+B+|B-A-, state variables

State variables (better)

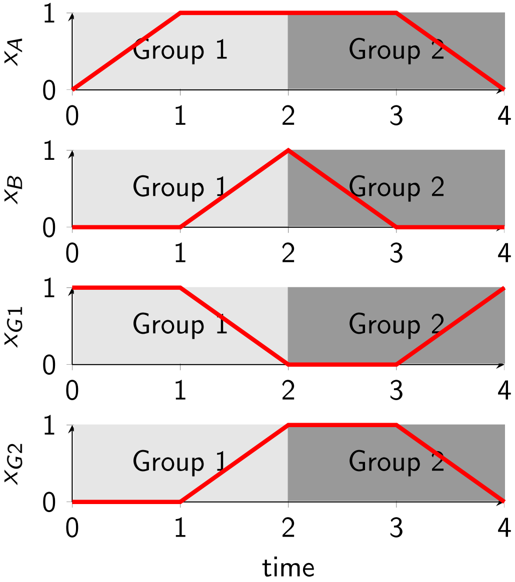
$$x = \begin{bmatrix} x_A & x_B & x_{G1} & x_{G2} \end{bmatrix}^T,$$

with

$$x_{\{A,B\}} = \begin{cases} 0 & \text{Cylinder \{A,B\} retracted} \\ 1 & \text{Cylinder \{A,B\} extended} \end{cases}$$

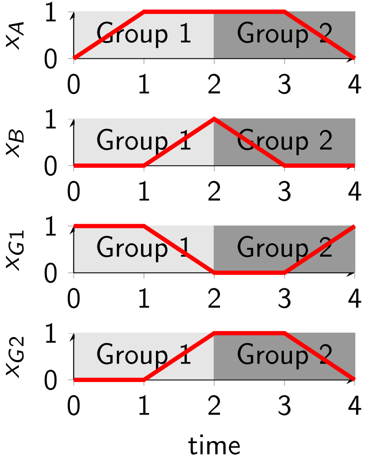
$$x_{Gi} = \begin{cases} 0 & \text{Group } i \text{ not active} \\ 1 & \text{Group } i \text{ active} \end{cases}$$

State transitions



Implementing the sequence A+B+|B-A-, control law

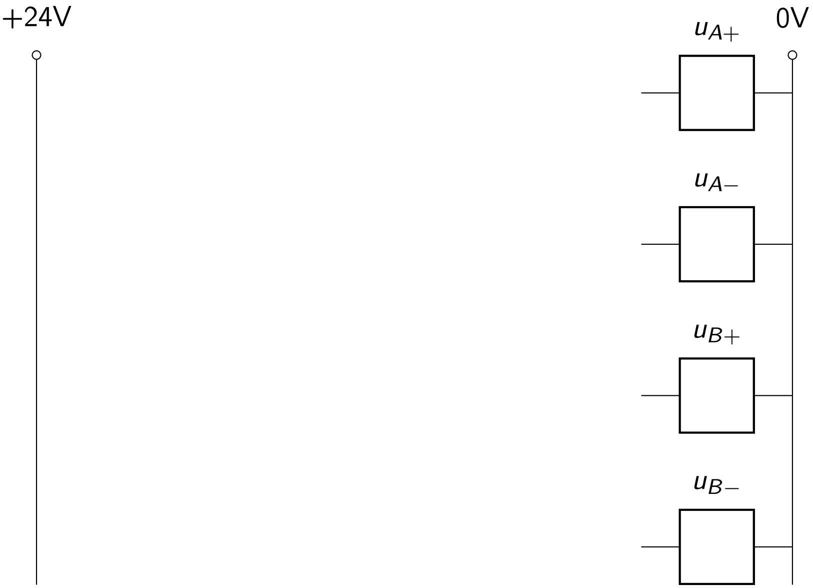
State transitions



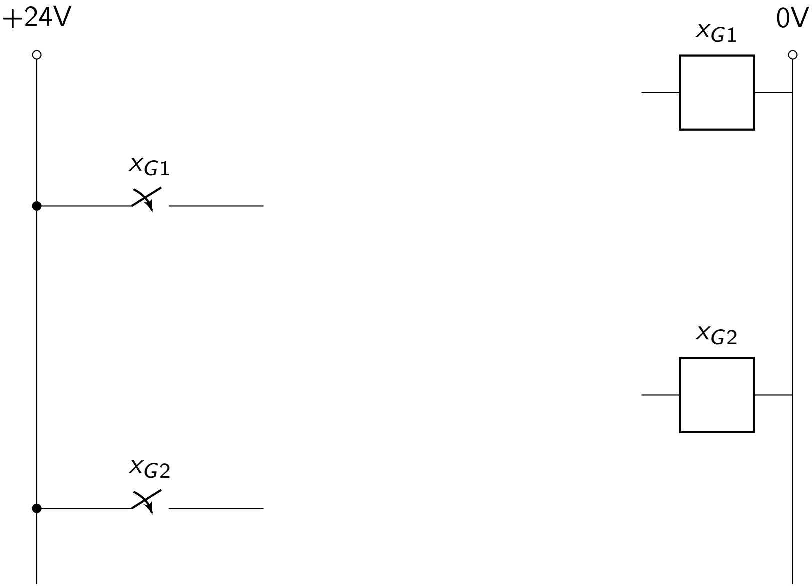
Control law

x_A	x_B	x_{G1}	x_{G2}	u_A+	u_A-	u_B+	u_B-
0	0	1	0				
1	0	1	0				
1	1	0	1				
1	0	0	1				

Implementing the control law



Implementing the group transitions



Implementing the proximity sensor circuit