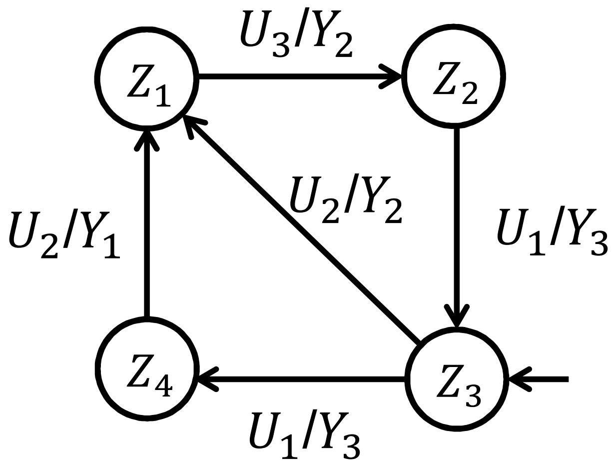
Logic Control

Prof. Dr. Ping Zhang

WS 2017/2018

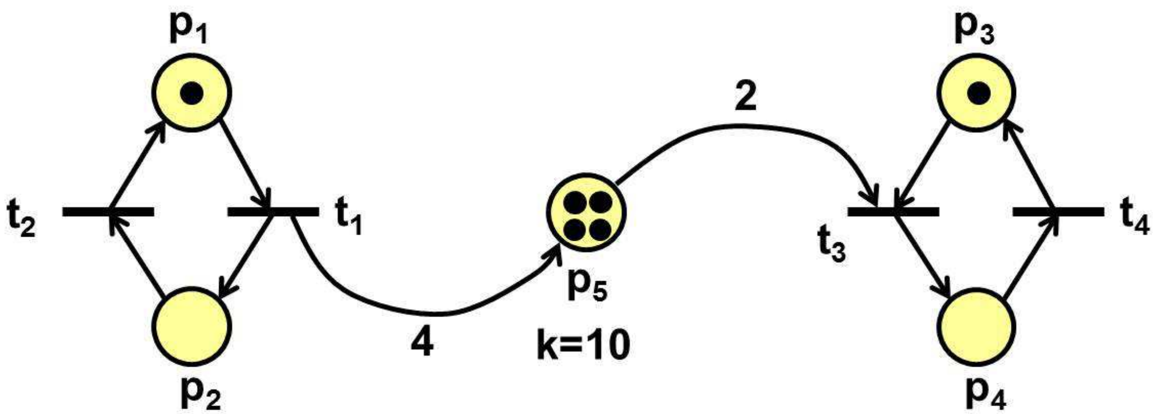
Overview of the course

- **Introduction**
- **Modeling of logic control systems**
  - **Boolean algebra**
  - **Finite state automata**
  - **Petri nets**, SIPN
- Analysis of logic control systems
- Design of logic control systems
- Verification and validation
- Online diagnosis of logic control systems
- Implementation of logic control systems
  - PLC
  - Programming languages (IEC 61131-3)
  - Automatic code generation
- Distributed control (optional)

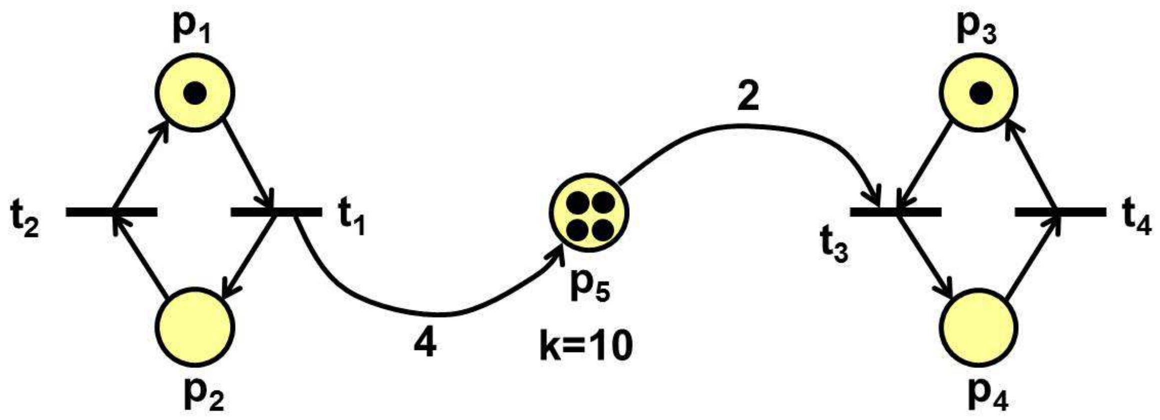


Petri nets (PN)

- Proposed by Carl Petri in 1961
- Petri nets are suitable for the description of **dynamic systems with discrete signals**, especially **concurrent processes**.



Petri nets

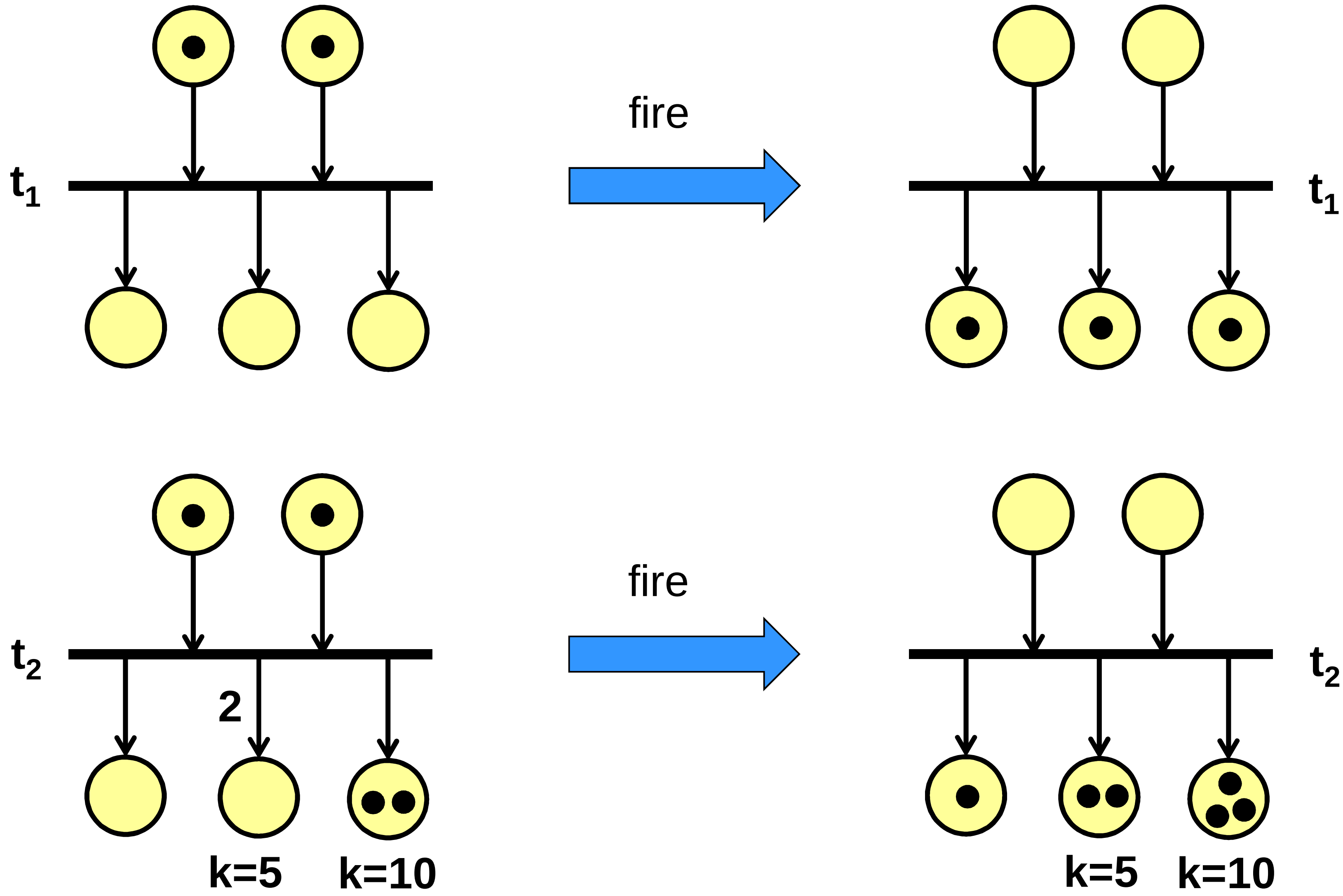


- One of the basic forms of Petri nets: **place transition net**
- Two types of nodes in a place transition net: **places** and **transitions**.
- **Directed arcs**: either from a place to a transition or from a transition to a place.
- Each transition has **pre-place(s)** and **post-place(s)**.
- Each place contains a number of **tokens**. The maximal number of tokens that can be put in one place is called the **capacity** of the place.
- The distribution of tokens in the petri net is called the **marking**.

Petri nets

- Tokens are moved by the **firing of transitions**. → system dynamics
- If a transition fires, then tokens will be removed from all its pre-places and all its post-places will receives tokens. The number of tokens that are removed from / added to one place is decided by the **weight** of the directed arc that is connected to that place.
- **Firing conditions** of a transition (i.e. the transition is activated / enabled):
  - Each **pre-place** of the transition has enough tokens.
  - Each **post-place** of the transition has enough capacity to receive the token.
- By the firing of transitions, the marking may change.
- Interpretation from the control perspective: **A marking corresponds to a state of the dynamic system**.
- If several transitions are enabled at the same time, it is assumed that these transitions can only fire individually and successively, but not simultaneously.

Petri nets



Petri nets

In summary, a place transition net is characterized by

$$N = (P, T, F, k, w, m_0)$$

$P = \{p_1, p_2, \cdots, p_{n_s}\}$: the finite set of **places**

$T = \{t_1, t_2, \cdots, t_{n_t}\}$: the finite set of **transitions**

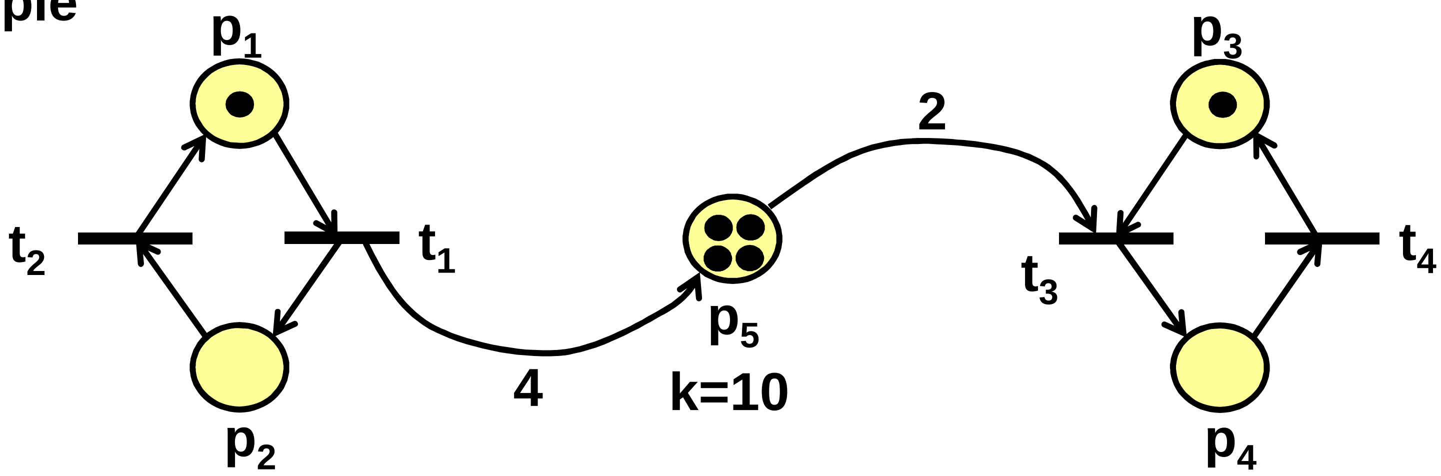
$F \subseteq (P \times T) \cup (T \times P)$: the set of **directed arcs** (**flow relation**) from places to transitions or from transitions to places.

k: the **capacity** of the places.

w: the **weight** of the arcs, which shows how many tokens should be taken away from the pre-places or how many token should be added to the post-places, if a transition fires..

m_0: the number of tokens in each place at the initial state (called **initial marking**)

Example



$P = \{p_1, p_2, p_3, p_4, p_5\}$, $T = \{t_1, t_2, t_3, t_4\}$,

$F = \{(p_1, t_1), (p_2, t_2), (p_3, t_3), (p_5, t_3), (p_4, t_4),$

$(t_1, p_2), (t_1, p_5), (t_2, p_1), (t_3, p_4), (t_4, p_3)\}$

$k(p_1) = 1, k(p_2) = 1, k(p_3) = 1, k(p_4) = 1, k(p_5) = 10$

$w(p_1, t_1) = 1, w(p_2, t_2) = 1, w(p_3, t_3) = 1, w(p_5, t_3) = 2, w(p_4, t_4) = 1,$

$w(t_1, p_2) = 1, w(t_1, p_5) = 4, w(t_2, p_1) = 1, w(t_3, p_4) = 1, w(t_4, p_3) = 1.$

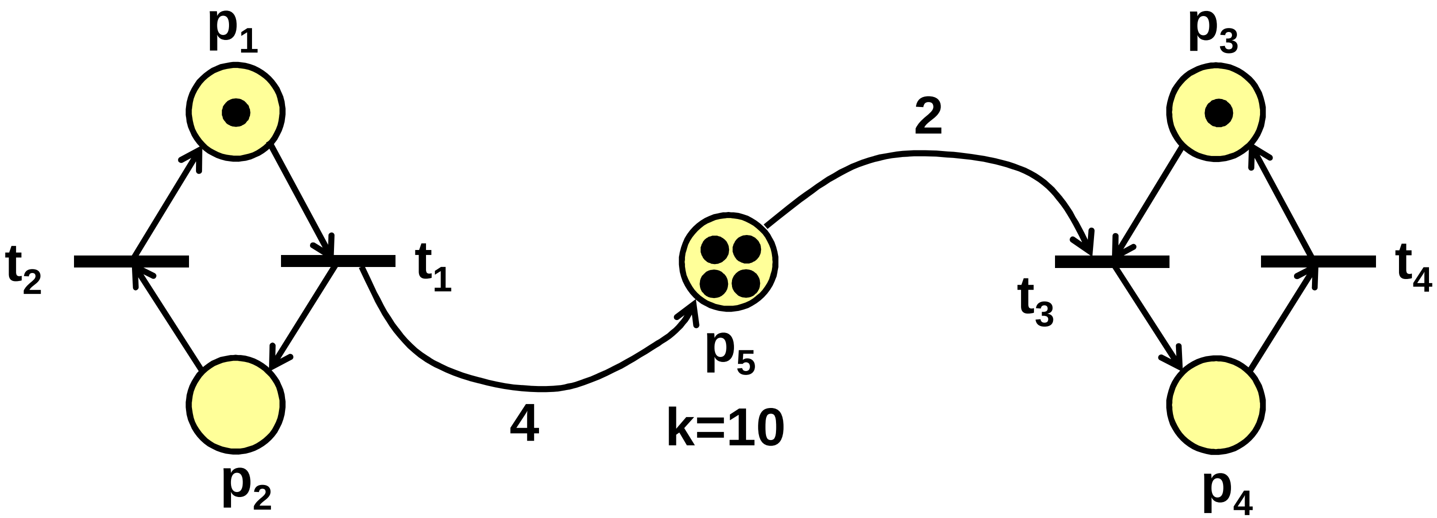
$m_0 = [1 \;\; 0 \;\; 1 \;\; 0 \;\; 4]'$

Petri nets

- If the transition $\mathbf{t_1}$ fires, then 1 token will be removed from its pre-place $\mathbf{p_1}$, simultaneously the post-place $\mathbf{p_2}$ receives 1 token and $\mathbf{p_5}$ receives 4 token.
- The conditions for $\mathbf{t_1}$ to fire are: (1) its pre-place $\mathbf{p_1}$ has at least 1 token, (2) its post-place $\mathbf{p_2}$ has no token (i.e. unmarked) and its post-place $\mathbf{p_5}$ has at most 6 tokens.
- In the above example, $\mathbf{t_1}$ and $\mathbf{t_3}$ can fire, $\mathbf{t_2}$ and $\mathbf{t_4}$ can not fire.

Petri nets

A Petri net can be described by its **incidence matrix**.



$$N = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{array} \begin{array}{c} \begin{array}{cccc} t_1 & t_2 & t_3 & t_4 \end{array} \\ \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \\ 4 & 0 & -2 & 0 \end{bmatrix} \end{array}$$

Petri nets

The incidence matrix can be used to describe the change of the marking.

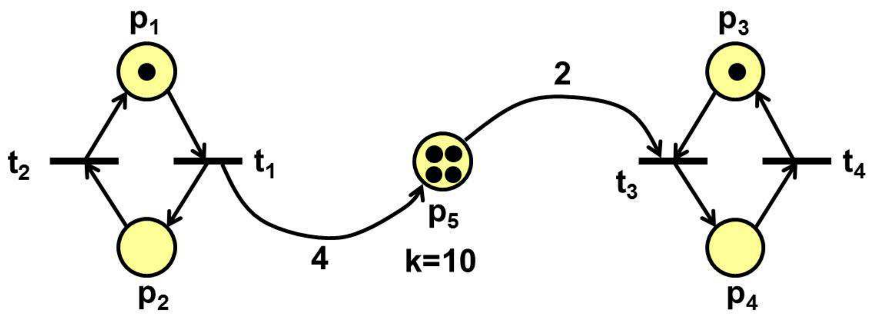
$$m = m_0 + Nq$$

m:	the current marking
m_0:	the initial making
N:	the incidence matrix
q:	the firing count vector

A tool for the **algebraic analysis** of petri net characteristics

Petri nets

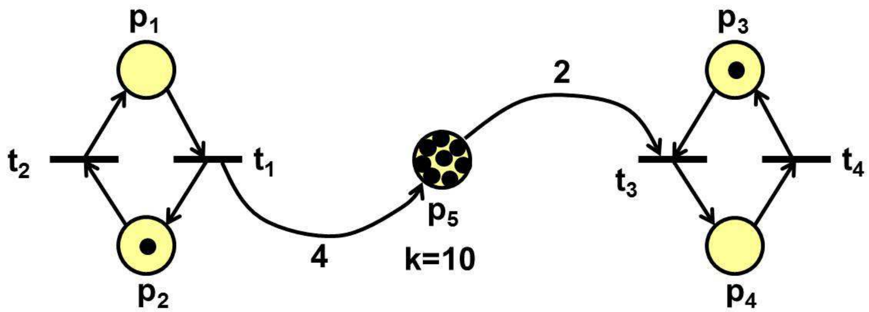
Example



$$m_0 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 4 \end{bmatrix}$$

t$_1$ fires once

$$q = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



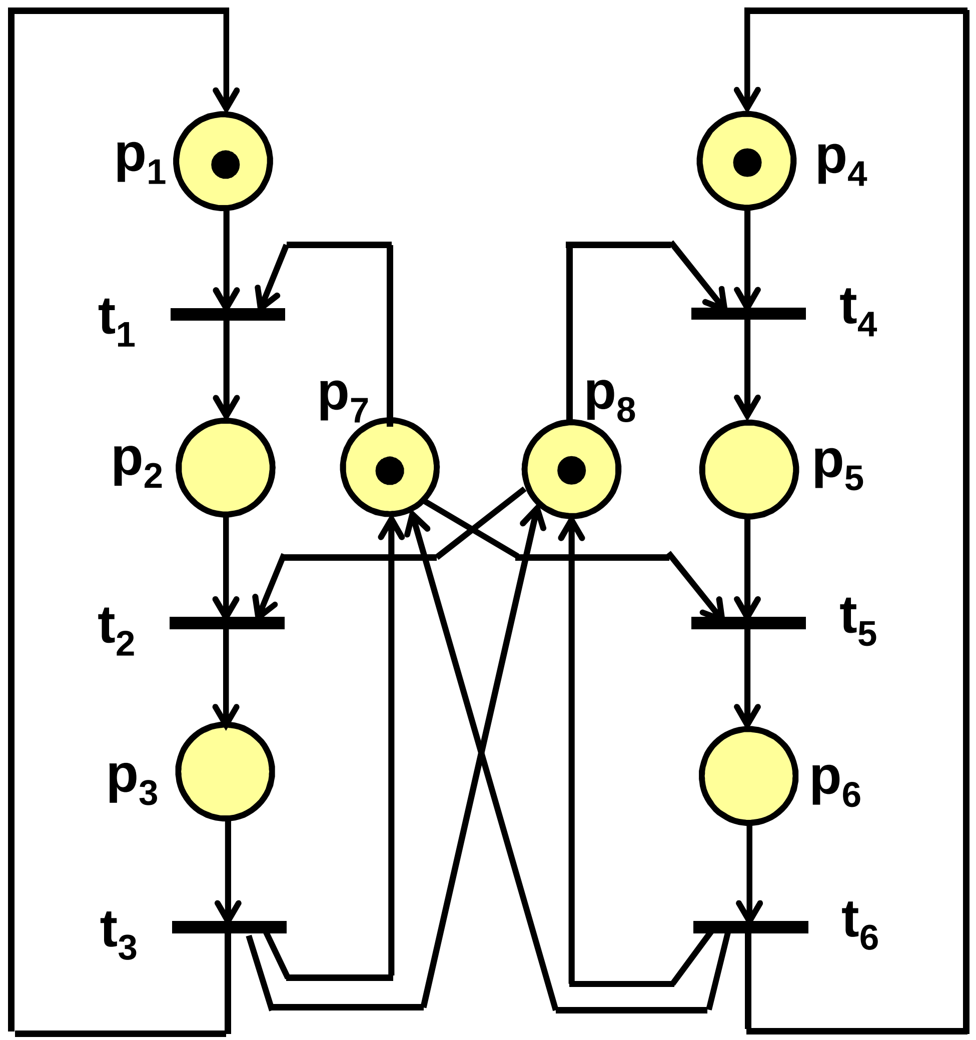
$$m = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 8 \end{bmatrix} = m_0 + Nq$$

Petri nets

Condition event nets:

- A special kind of place transition nets, in which the **capacity** of all places is 1 and the **weight** of all arcs is 1.
- The places in a condition event net is either **marked** or **unmarked**.
- A transition in a condition event net can fire, if all the **pre-places** of this transition are **marked** and all the **post-places** of this transition are **unmarked**.
- If a transition is fired, then all the pre-places of this transition become unmarked and all the post-places of this transition become marked.
- The pre-places represent the conditions, so that the transition can fire (i.e. the event happens).

Example

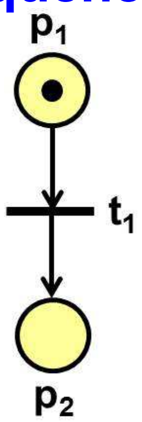


1. CE net?
2. Incidence matrix?
3. Which transition(s) can fire now?

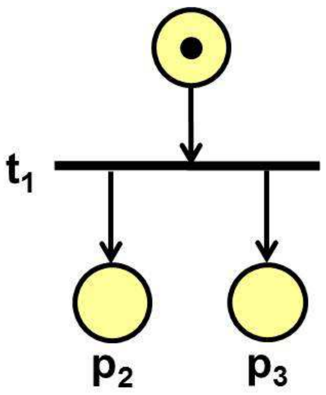
Petri nets

Basic structures in Petri nets

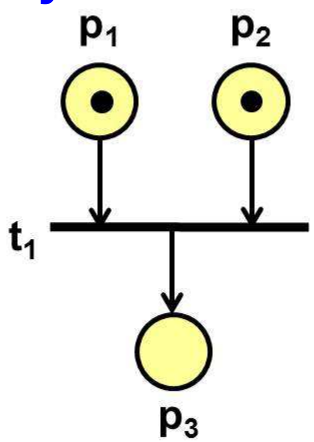
Sequence



Splitting



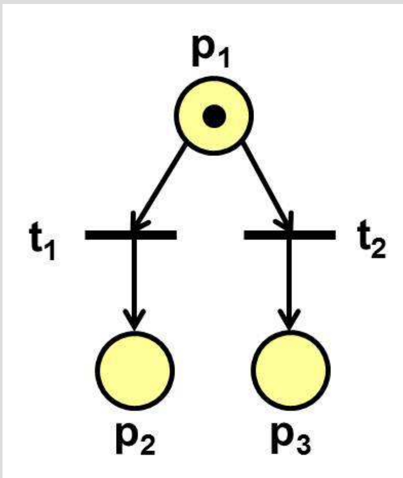
Synchronization



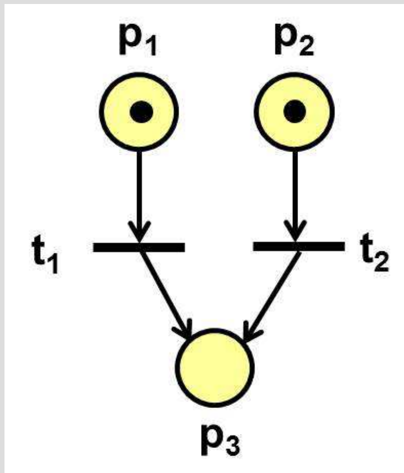
Conflict:

Both transitions are ready to fire. The firing of one transition will disable the other transition.

Choice



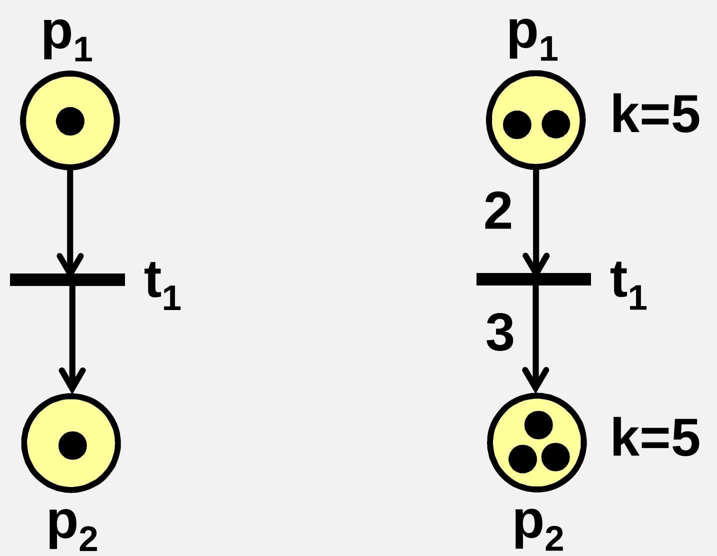
Merging



Contact in petri nets

Contact:

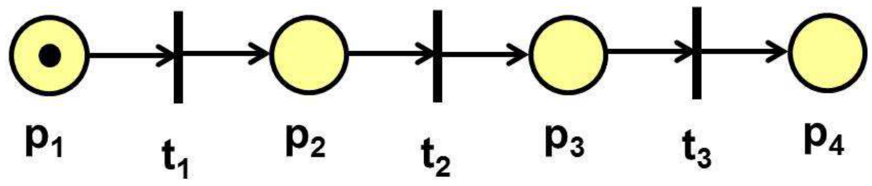
The transition can not fire. The pre-places have enough tokens, but the post-places don‘t have enough capacity.



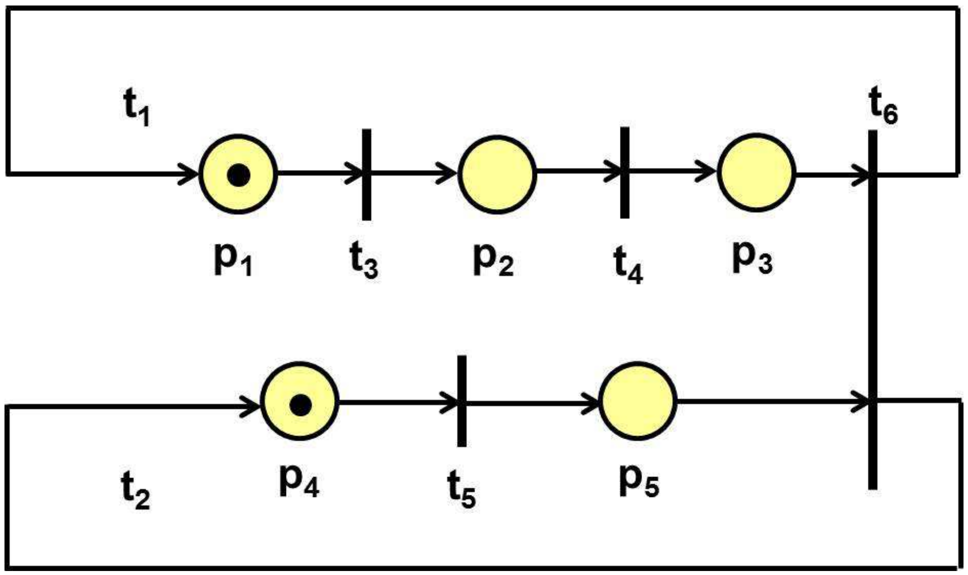
Petri nets

Typical examples of petri nets

Sequential execution

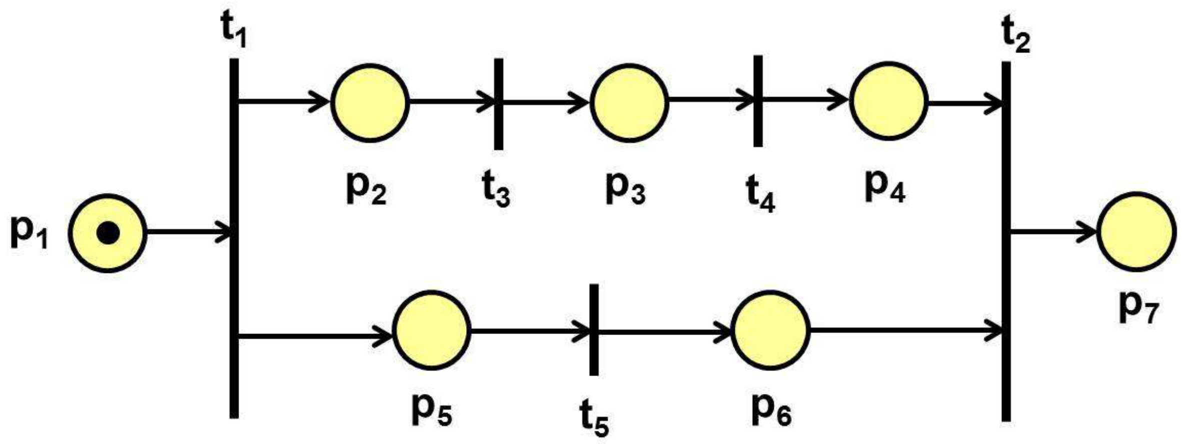


Synchronisation

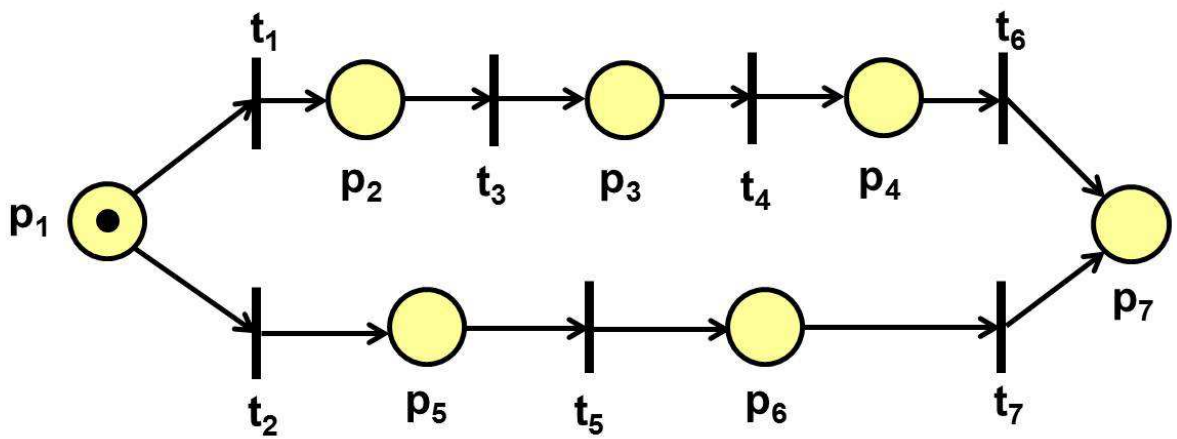


Petri nets

Parallel process

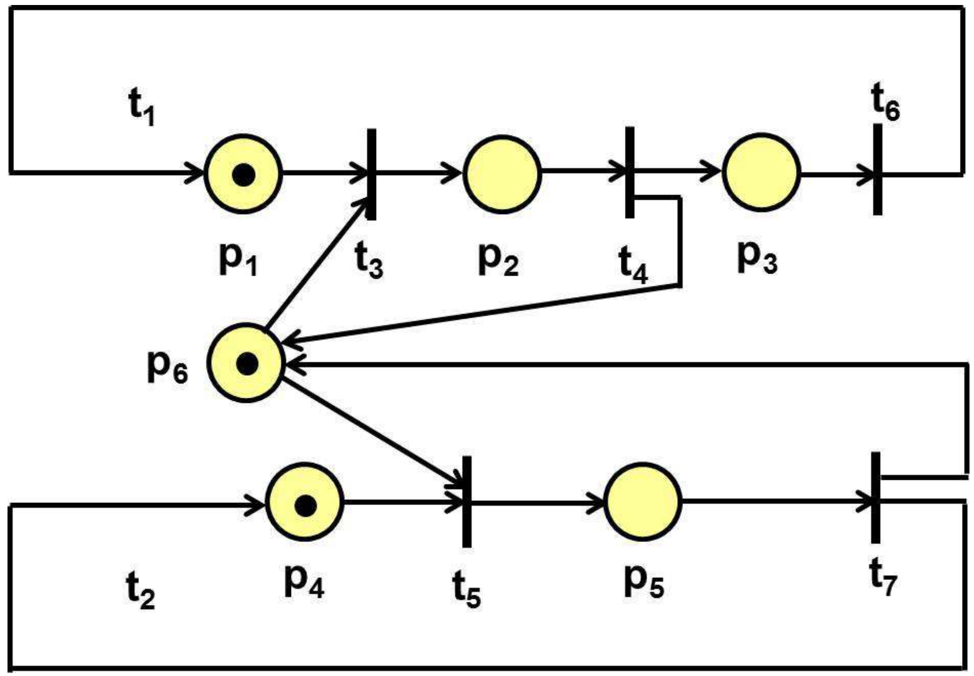


Alternative process



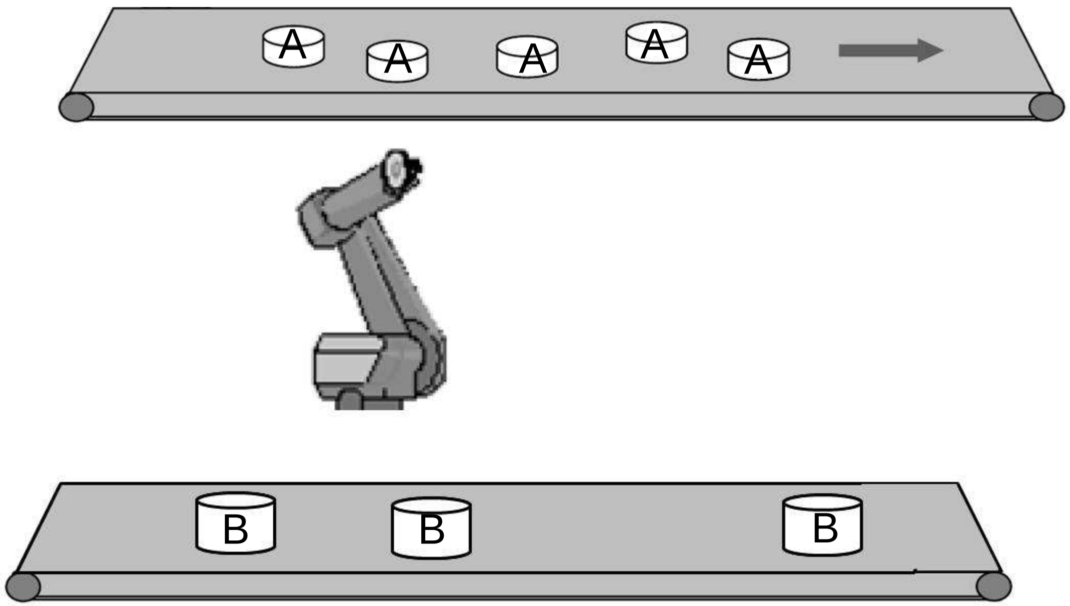
Petri nets

Shared resource



Modelling by PN: Shared robot

Example 1: Flexible production process with shared robot

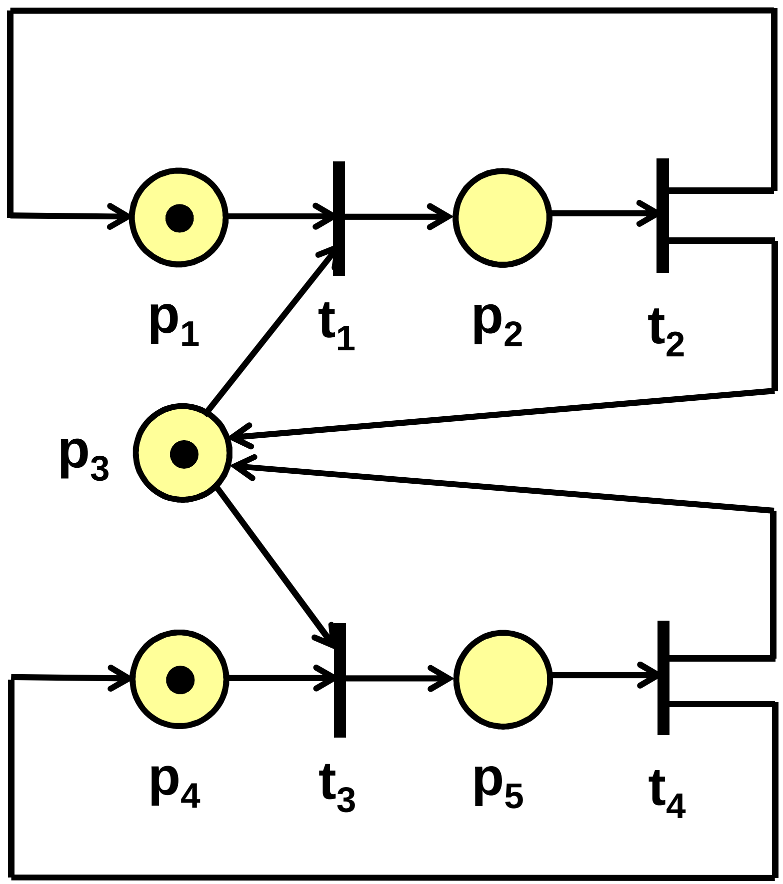


The robot serves two parallel production lines.

The production line handles, respectively, workpiece type A and workpiece type B.

Modelling by PN: Shared robot

Production line I:

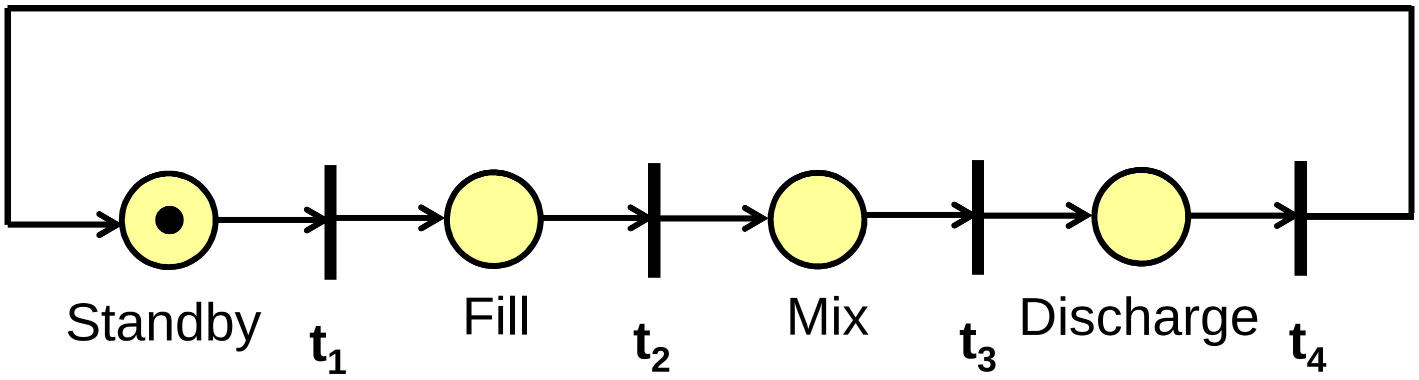


p_1: Workpiece A is not being handled.
p_2: Workpiece A is being handled.
p_3: The robot is free.
p_4: Workpiece B is not being handled.
p_5: Workpiece B is being handled.

Production line II:

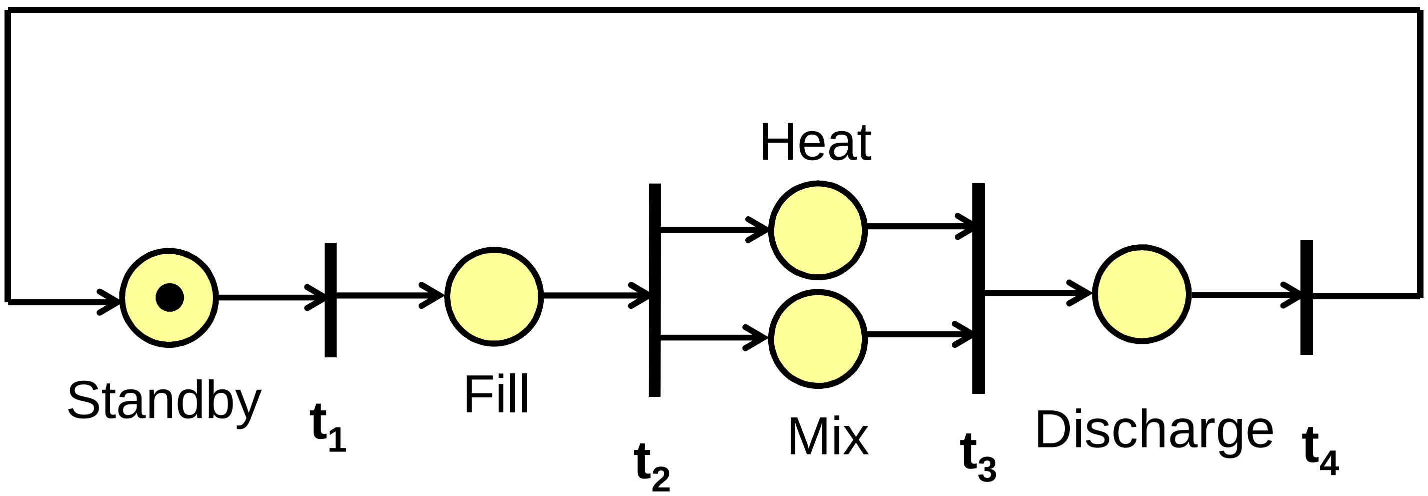
Modelling by PN: further examples

Example 2: Mixing tank



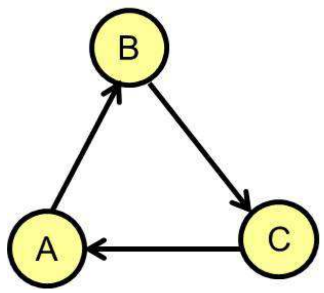
Modelling by PN: further examples

Example 3: Heated mixing tank

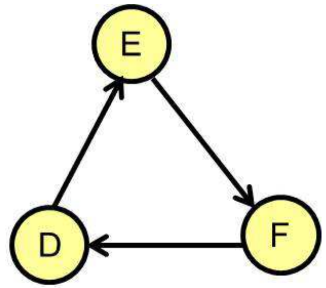


FSA vs. PN

Assume that there are two production line, which are independent and asynchron.

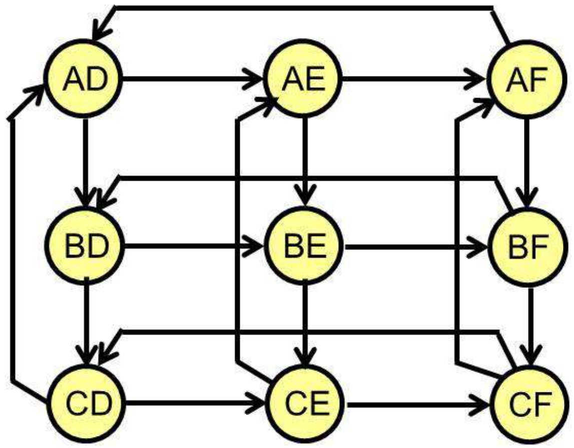


production line 1

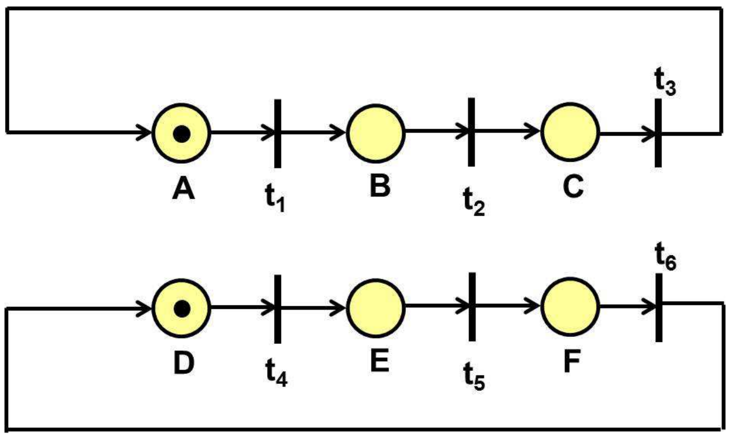


production line 2

FSA description
of the whole system

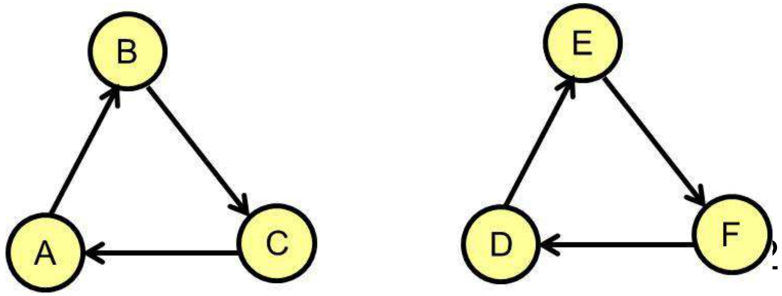


PN description
of the whole system

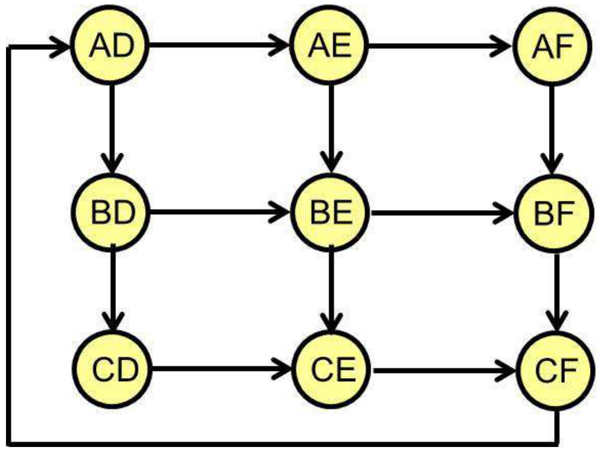


FSA vs. PN

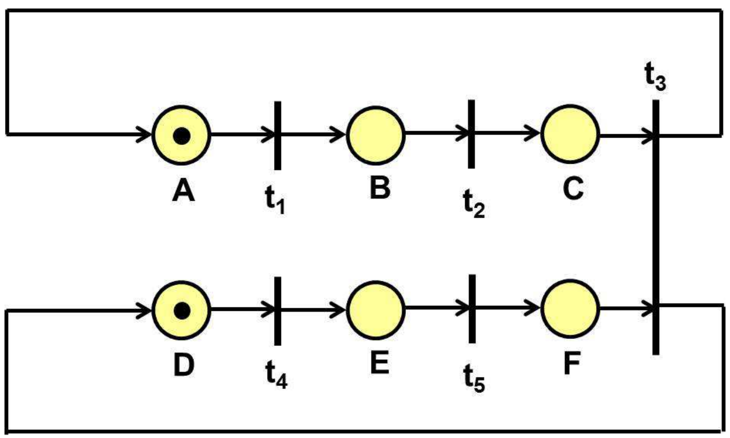
Assume that there are two production line, which are independent and the state transitions C→A and F→D must happen simultaneously.



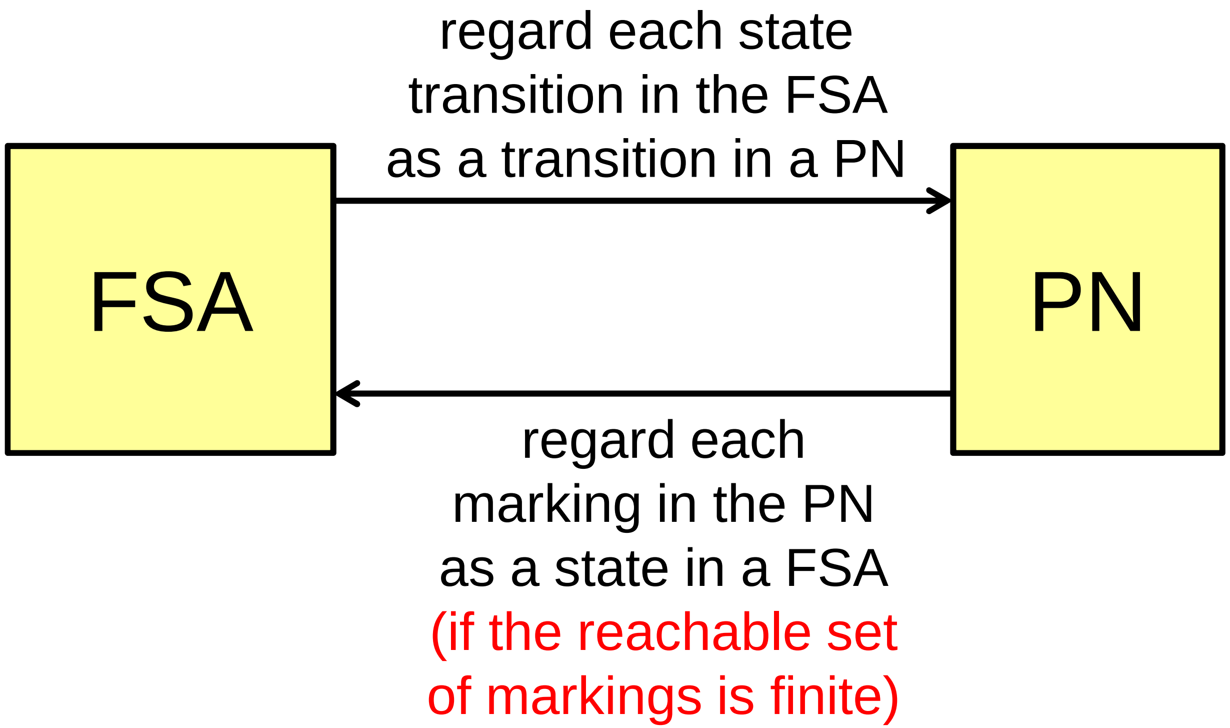
FSA description
of the whole system



PN description
of the whole system



FSA vs. PN



The FSA and the PN are complementary modeling approaches!

Summary of Petri nets

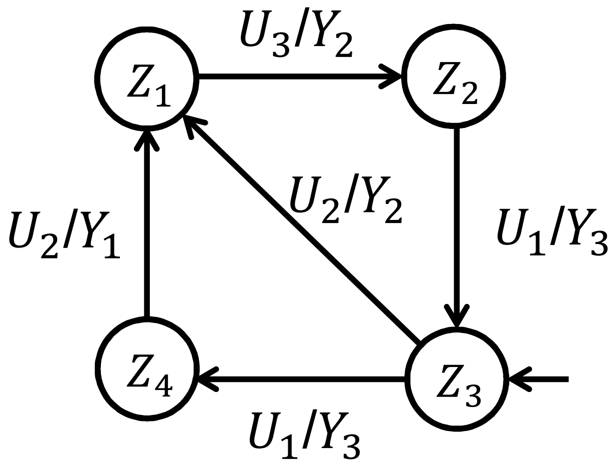
- Petri nets are convenient for the description of concurrent processes.
- Distributed information structure
- However, in the basic form the input and output signals have not been considered.

↓

Signal interpreted Petri nets (SIPN)

Overview of the course

- **Introduction**
- **Modeling of logic control systems**
  - **Boolean algebra**
  - **Finite state automata**
  - **Petri nets, SIPN**

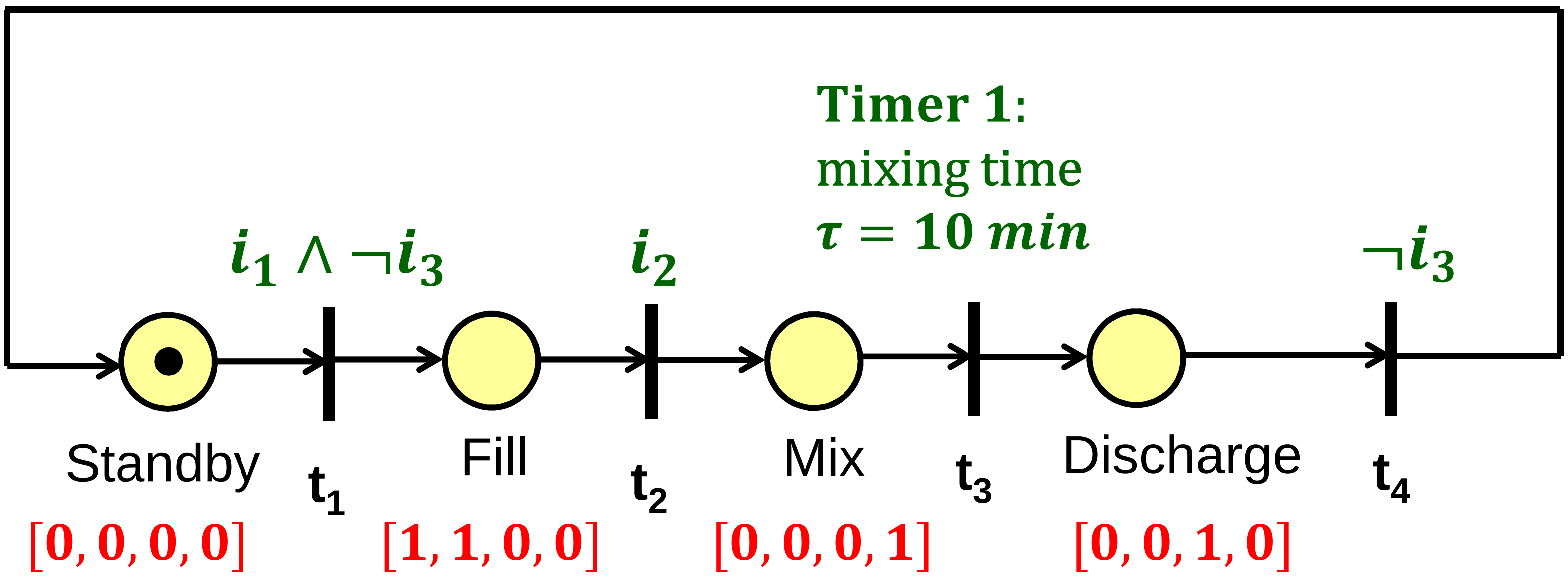


- Analysis of logic control systems
- Design of logic control systems
- Verification and validation
- Online diagnosis of logic control systems
- Implementation of logic control systems
  - PLC
  - Programming languages (IEC 61131-3)
  - Automatic code generation
- Distributed control (optional)

Signal interpreted Petri nets (SIPN)

- The SIPN is introduced to describe the **interaction** of the petri nets with the environment.
- **Input and output signals** are taken into account.
- Defined based on the **condition event nets** (CE nets).

Signal interpreted Petri nets (SIPN)



- Every **transition** is associated with a **firing condition** (for instance, Boolean function of input signals, timers).
- Every **place** is associated with an **output** vector (for instance, signals sent to actuators or other controllers, information to operators, etc).

Signal interpreted Petri nets (SIPN)

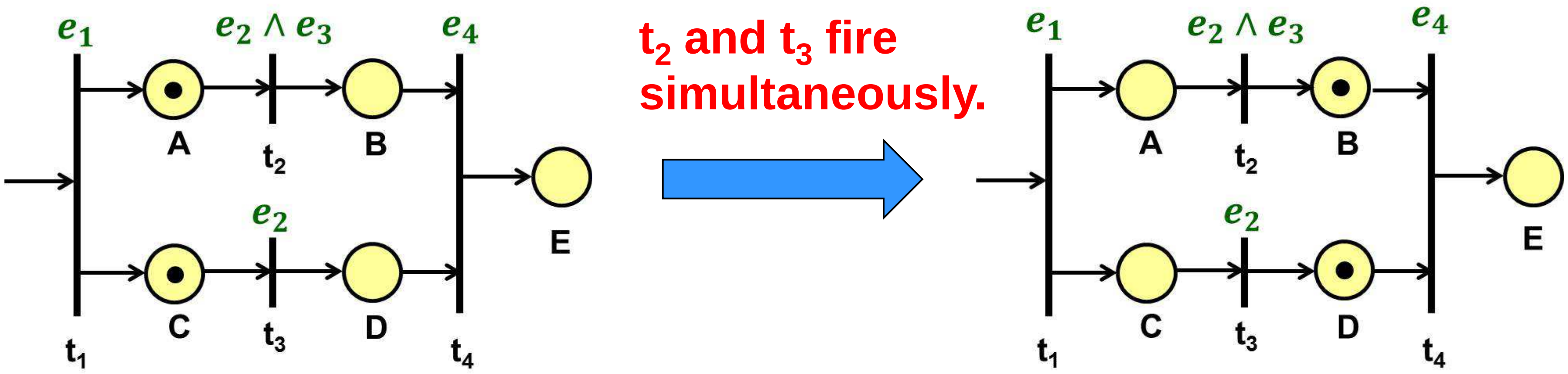
- A transition in an SIPN is **enabled**, if
  - all the **pre-places** of this transition are **marked**,
  - all the **post-places** of this transition are **unmarked**.
- An enabled transition fires **immediately**, as soon as the **firing condition** of this transition is **fulfilled**.
- If a transition is fired, then all the pre-places of this transition become unmarked and all the post-places of this transition become marked.
- The current **output** signals are calculated based on the current **marking** of the SIPN.

Signal interpreted Petri nets (SIPN)

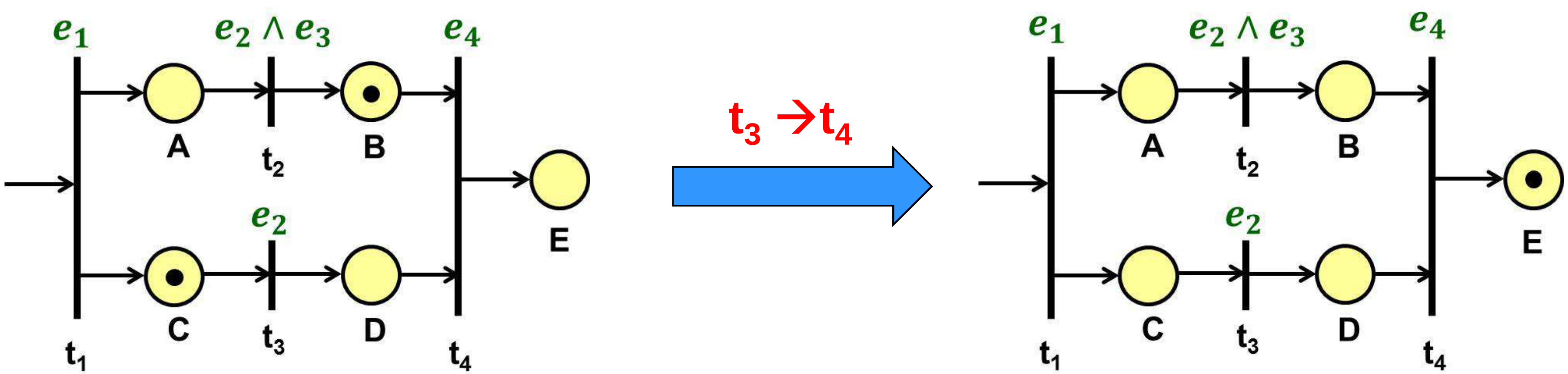
- About firing of transitions:
  - Firing of transitions takes no time.
  - If several transitions in the SIPN can fire simultaneously, they fire **simultaneously**.
  - The firing process is continued until a **stable marking** is reached. The transient marking(s) will not be shown in the reachability graph.

Signal interpreted Petri nets (SIPN)

Example 1: Input $e^T = [0 \;\; 1 \;\; 1 \;\; 0]$



Example 2: Input $e^T = [0 \;\; 1 \;\; 0 \;\; 1]$



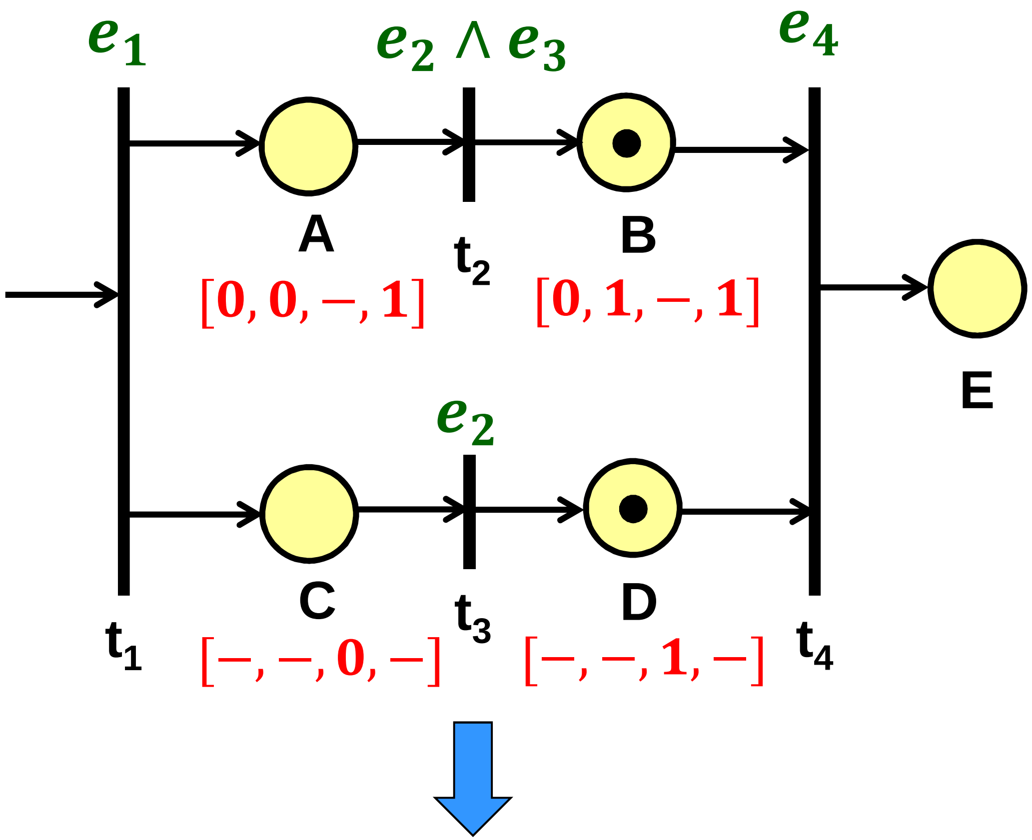
Signal interpreted Petri nets (SIPN)

- About **output signals**:
  - The output signals are calculated, after a new **stable marking** is reached.
  - The outputs of **the marked places** in the marking $\boldsymbol{m}$ are united together according to the **product operation** defined below:

$$\Omega(m) = \prod_{\substack{i\,, \\ p_i\, is\, marked}} \omega(p_i)$$

$\boldsymbol{x \cdot y}$	**1**	**0**	**-**	**c**
1	1	c	1	c
0	c	0	0	c
-	1	0	-	c
c	c	c	c	c

Example 3:



$$\Omega(m) = \omega(p_B) \cdot \omega(p_D) = [\,0, 1, -, 1] \cdot [-, -, 1, -] = [0,1,1,1]$$

Signal interpreted Petri nets (SIPN)

The product operation used in the output function is defined by

$$\Omega(m) = \prod_{\substack{i\,, \\ p_i \text{ is marked}}} \omega(p_i)$$

- Commutativity: $\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{y} \cdot \boldsymbol{x}$
- Idempotent: $\boldsymbol{x} \cdot \boldsymbol{x} = \boldsymbol{x}$
- Contradition: $\mathbf{c} \coloneqq 1 \cdot 0$
- „one“ element -: $- \cdot \boldsymbol{x} = \boldsymbol{x}$
- „zero“ element **c**: $\mathbf{c} \cdot \boldsymbol{x} = \mathbf{c}$

$x \cdot y$	**1**	**0**	**-**	**c**
1	1	c	1	c
0	c	0	0	c
-	1	0	-	c
c	c	c	c	c

Signal interpreted Petri nets (SIPN)

In summary, an SIPN is characterized by

$$SIPN = (P, T, F, m_0, I, O, \varphi, \omega)$$

I: the finite set of **input signals**.

O: the finite set of **output signals**, $I \cap O = \phi$.

φ: a mapping associating every **transition** with a **firing condition**.

ω: a mapping associating every **place** with an **output vector**,

$p_i \rightarrow \{0,1,-\}^{|O|},\ \mathrm{i} = 1,2,\cdots,n_p.$