

Állapotgépek, időzítők és számlálók

Programozható irányítóberendezések
és szenzorrendszerek

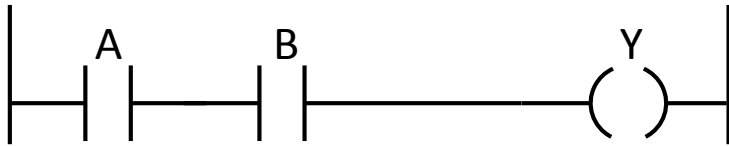
KOVÁCS Gábor
gkovacs@iit.bme.hu

A létradiagram kiértékelése


- A PLC ciklikus működésű
- A programvégrehajtás fázisában a teljes kód feldolgozásra kerül
- Minden egyes ciklusban a teljes létradiagram kiértékelésre kerül



Létrásor = Logikai függvény

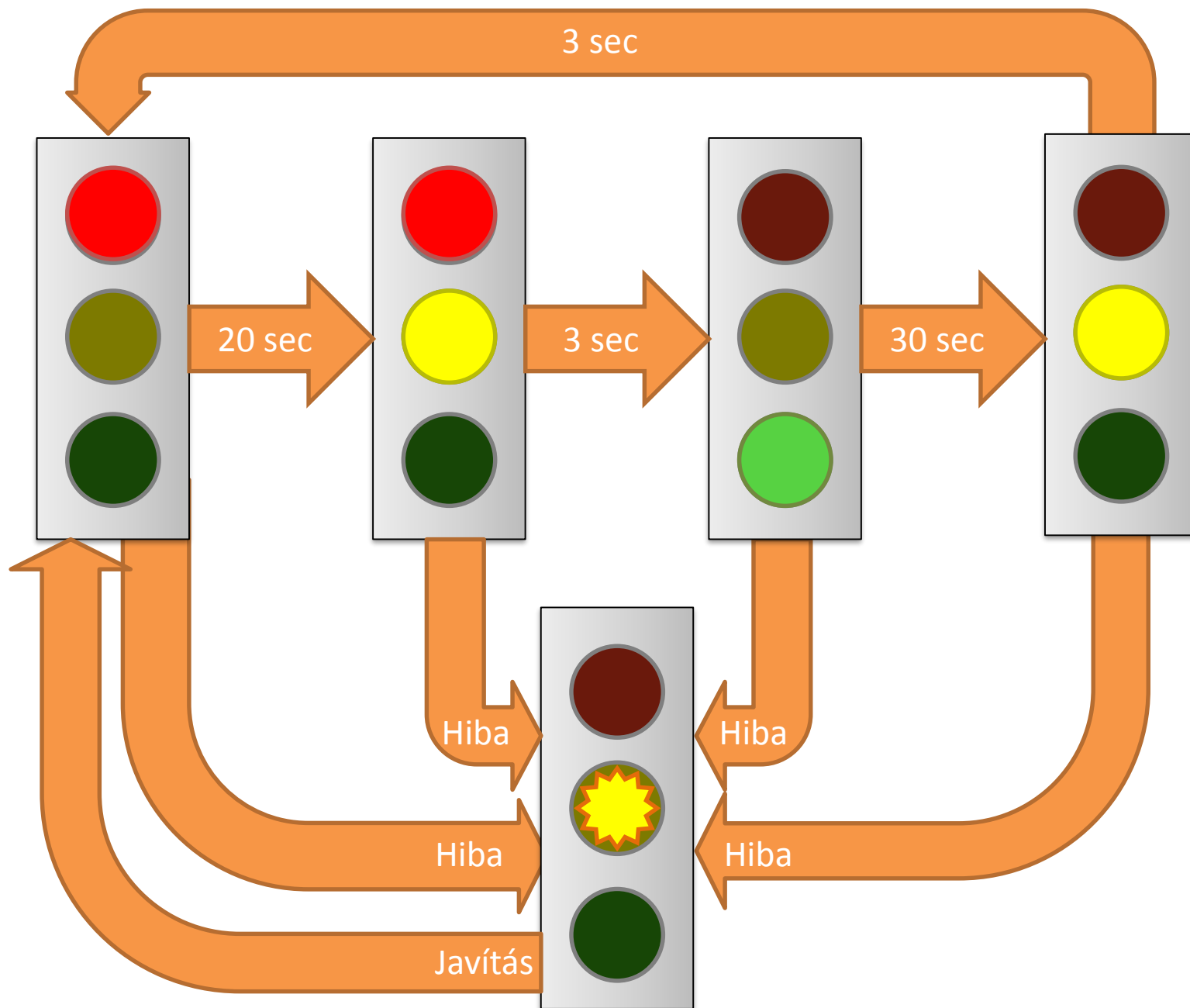


IF (A=1) AND (B=1)
THEN Y=1 

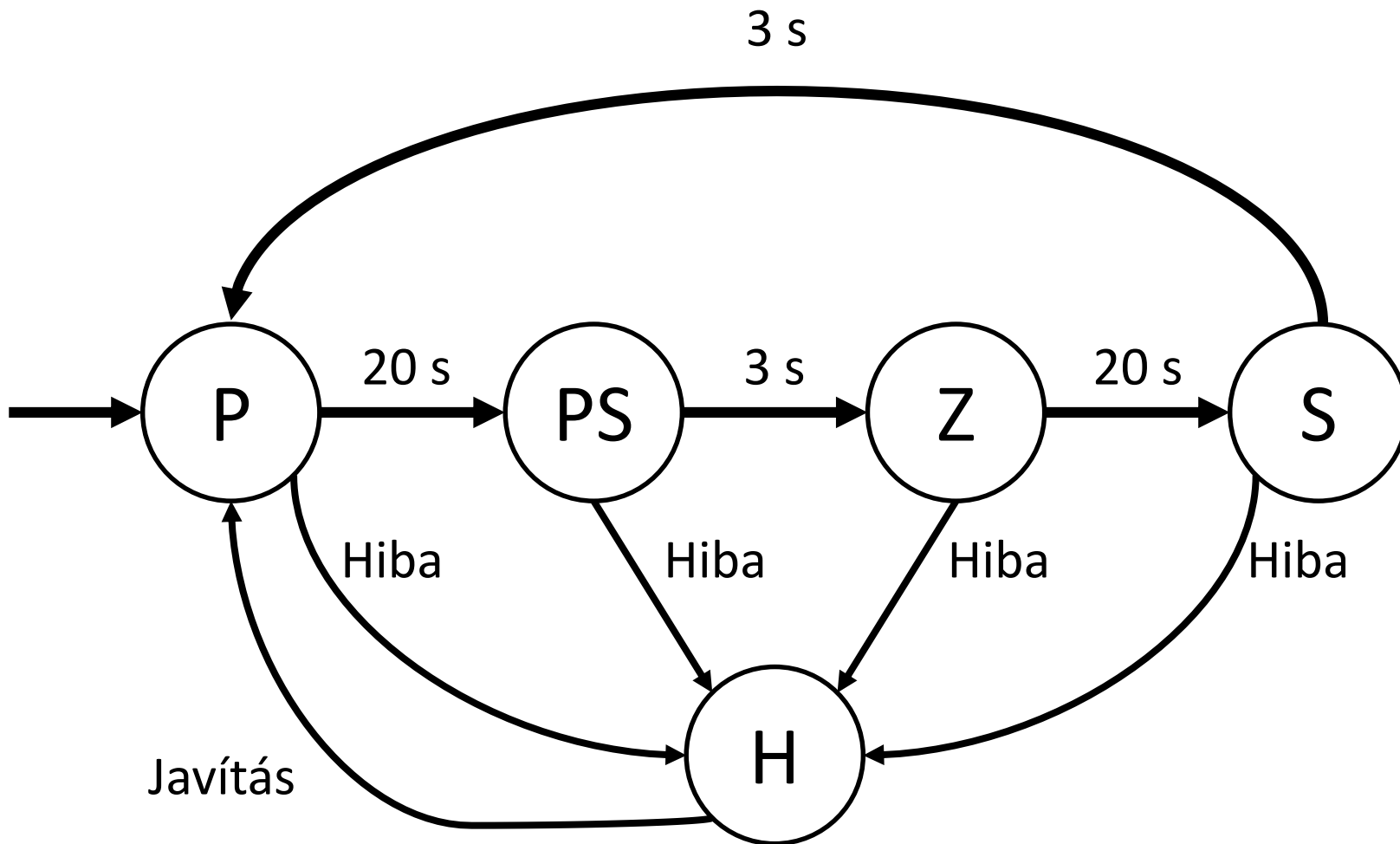
IF (A=1) AND (B=1)
THEN Y=1
ELSE Y=0 

Jelzőlámpa



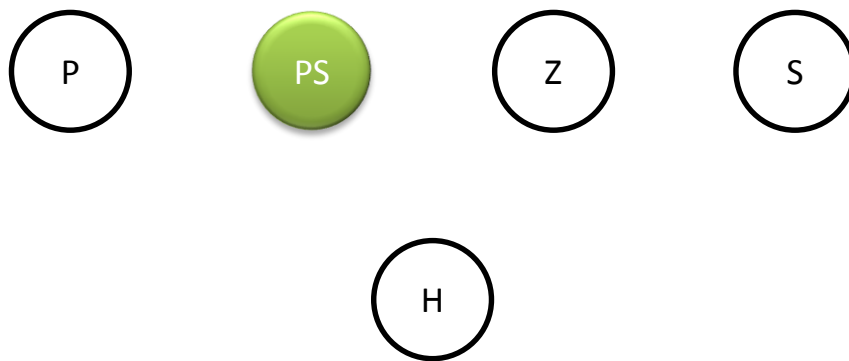


Jelzőlámpa



Állapotok

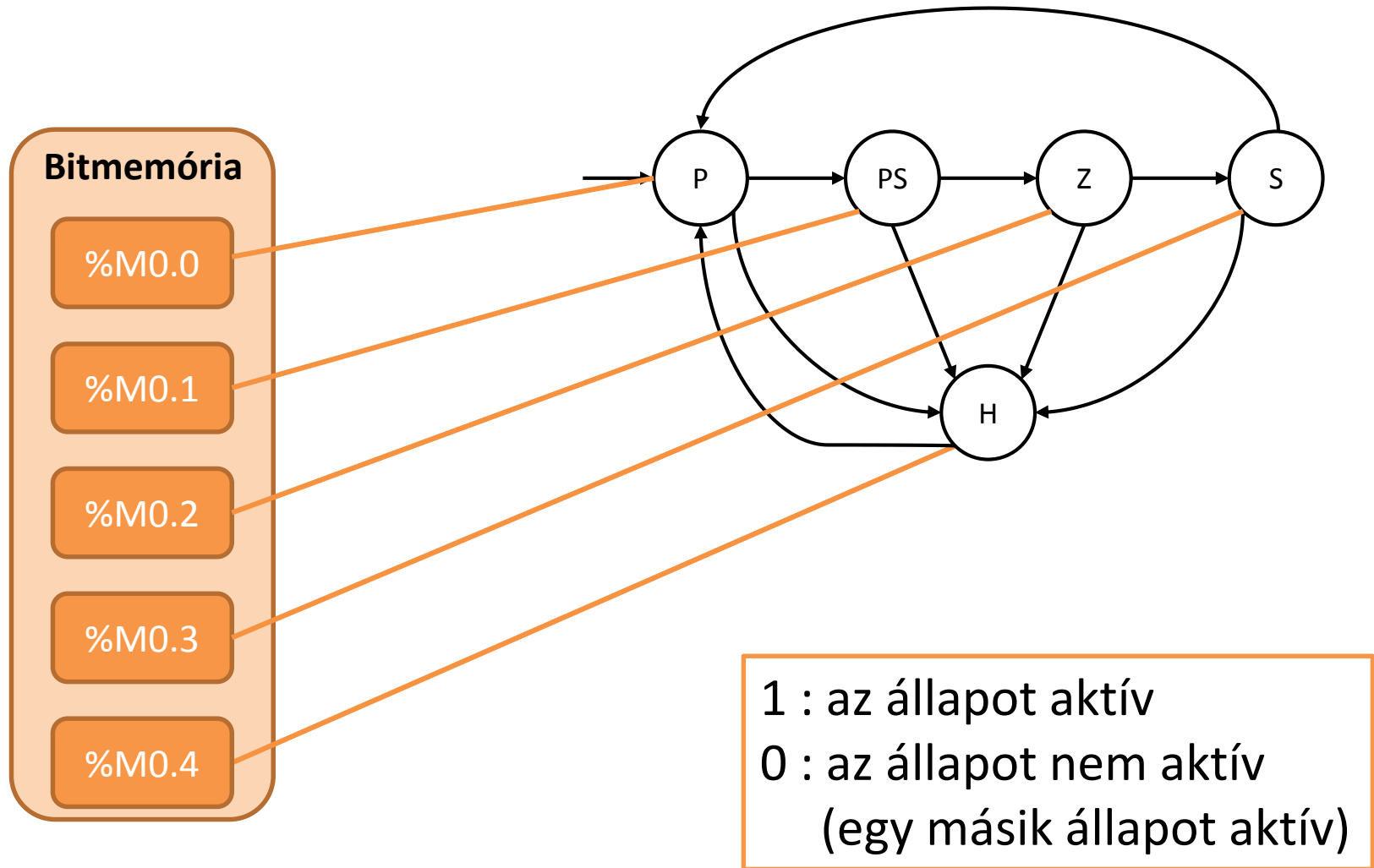
- A rendszer működésének egy fázisát vagy stádiumát reprezentálja
- Egyszerre csak egy állapot lehet aktív egy rendszeren belül (aktuális állapot)



Állapotok reprezentációja

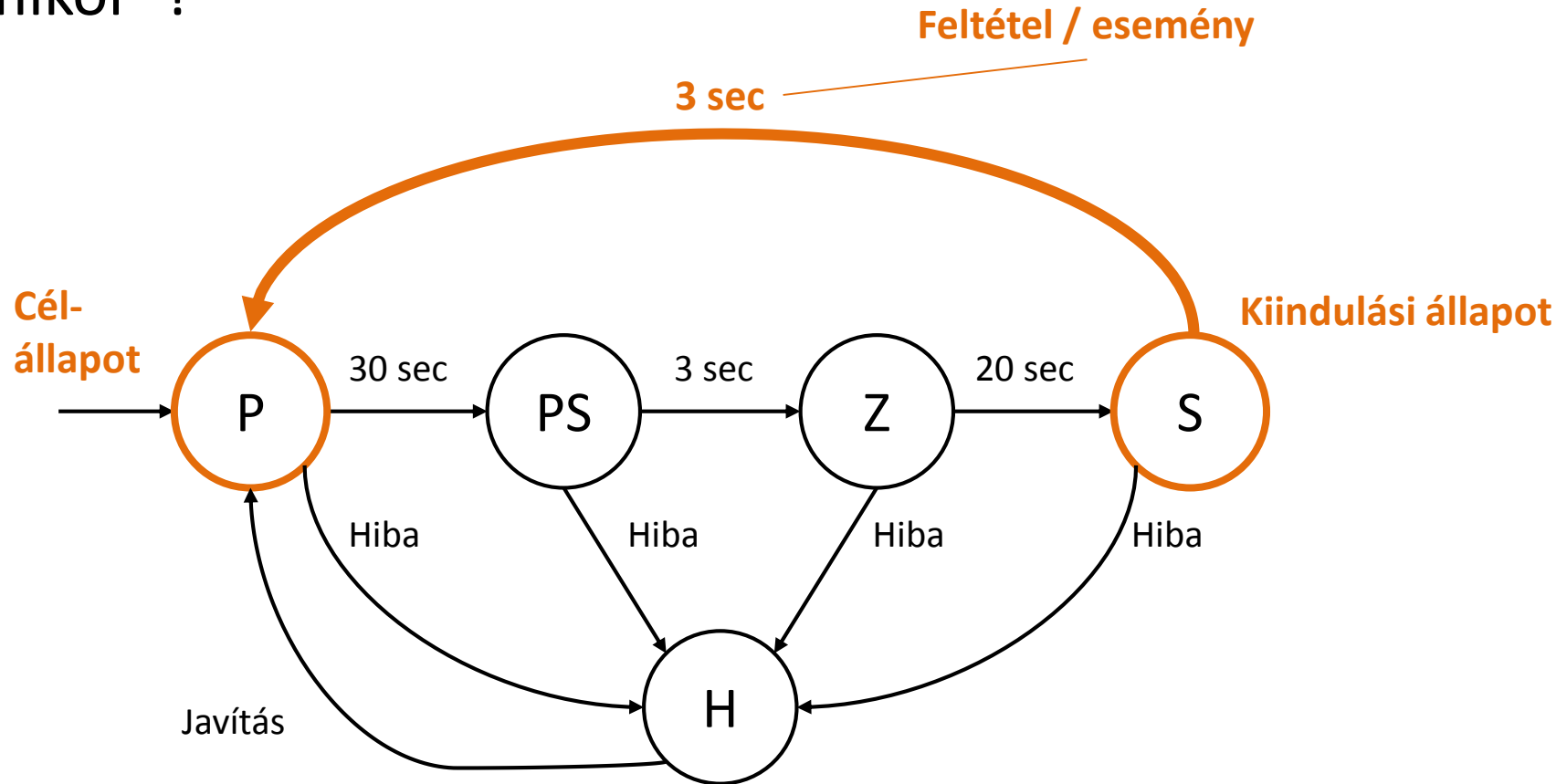
- **Bitregiszter minden egyes állapothoz**
 - Egy állapothoz rendelt bit aktív értéke jelzi, hogy az adott állapot az aktuális állapot
 - Egyszerre csak egy ilyen bit lehet aktív
 - Előny: egyszerű működés
 - Hátrány: sok regiszterre van szükség (állapotok számával megegyező)
- **Egyetlen szó-regiszter**
 - Az aktuális állapot sorszámát (azonosítóját) tartalmazza
 - Előny: egyetlen regiszterre van szükség
 - Hátrány: egy állapot aktív mivoltának ellenőrzése komparálást igényel

Állapotok reprezentációja



Átmenetek

- „Az S állapotban vagyok. Merre menjek tovább és mikor”?

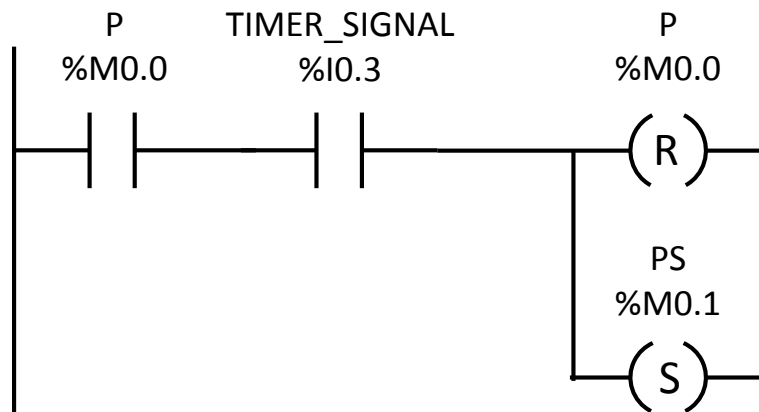
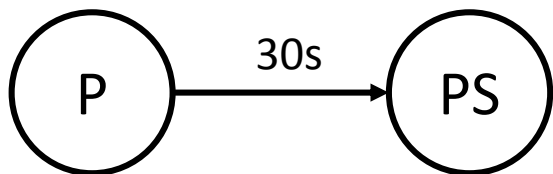
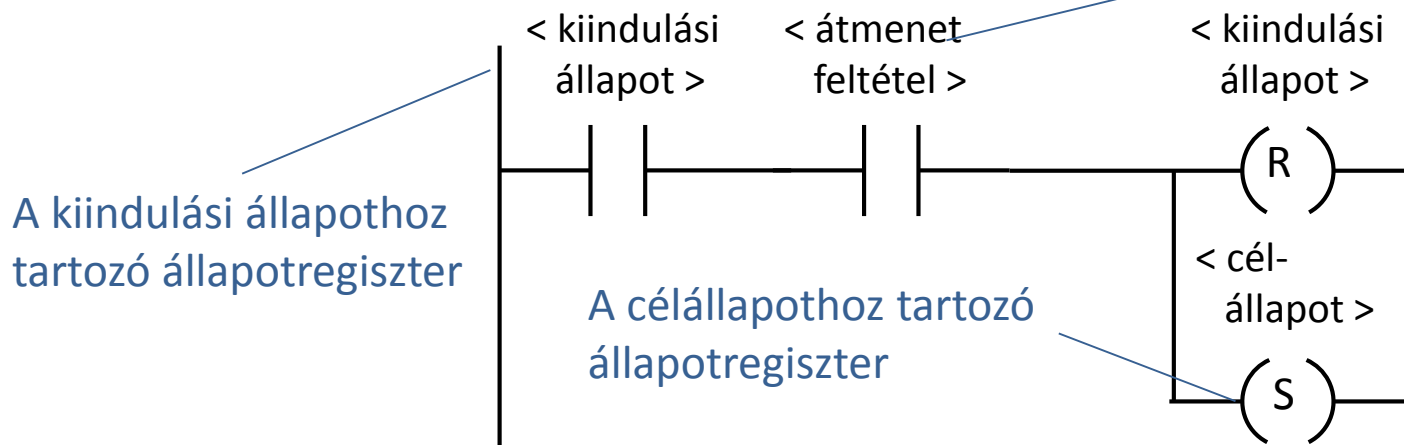


Átmenetek

- Az állapotátmenetet akkor kell végrehajtani, ha
 - kiindulási állapota aktív
- **ÉS**
- az átmenet feltétele igaz
- Egy átmenet végrehajtása során
 - az aktuális állapot regiszterét 0-ba kell állítani
 - a célállapot regiszterét 1-be kell állítani

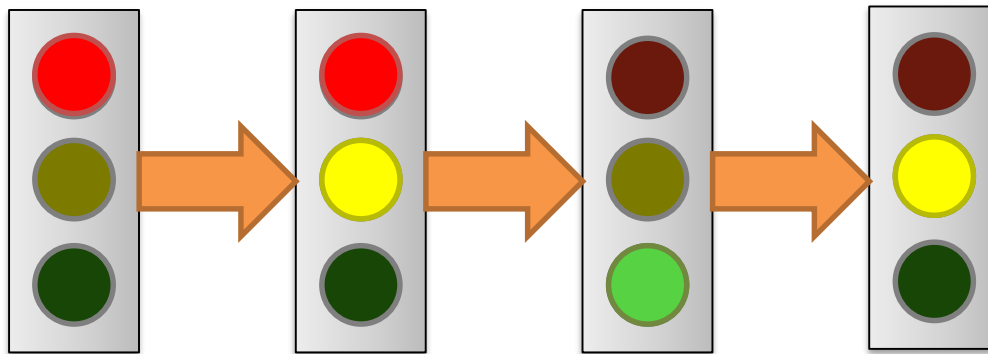
Átmenetek megvalósítása

Átmenet feltétele (tetszőleges logikai függvény)



Kimenetek

- Mealy automata: a kimenetek értéke az aktuális állapottól és a bemenetek aktuális értékétől függ
- Moore automata: a kimenetek értéke csak az aktuális állapottól függ



Kimenetek

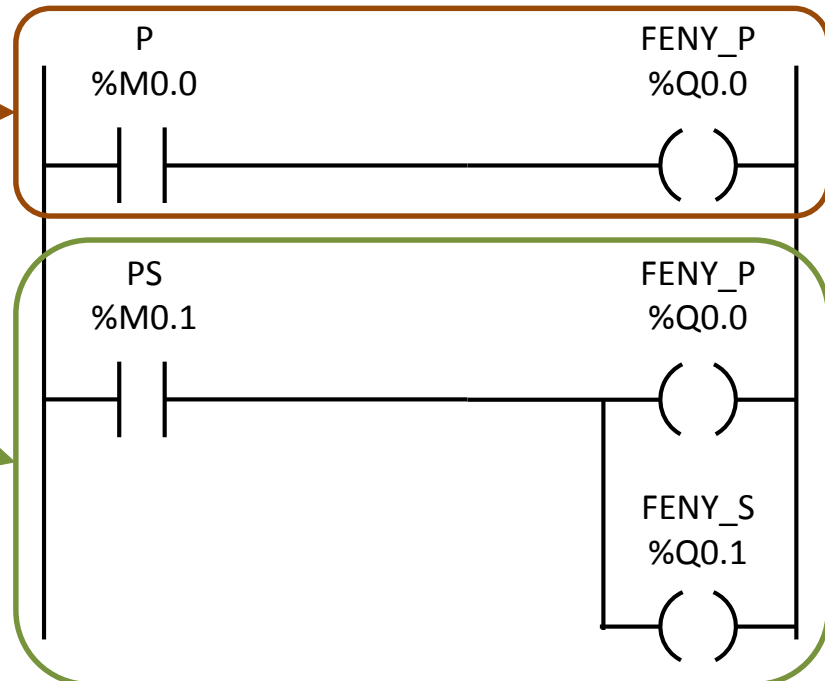
- Minden állapotban a kimenetek valamely kombinációja aktív
- A kimeneti leképezés egy táblázattal is megadható

Állapot	P	PS	Z	S	H
Piros fény	1	1	0	0	0
Sárga fény	0	1	0	1	0
Zöld fény	0	0	1	0	0
Villogó sárga	0	0	0	0	1

A kimeneti leképezés megvalósítása

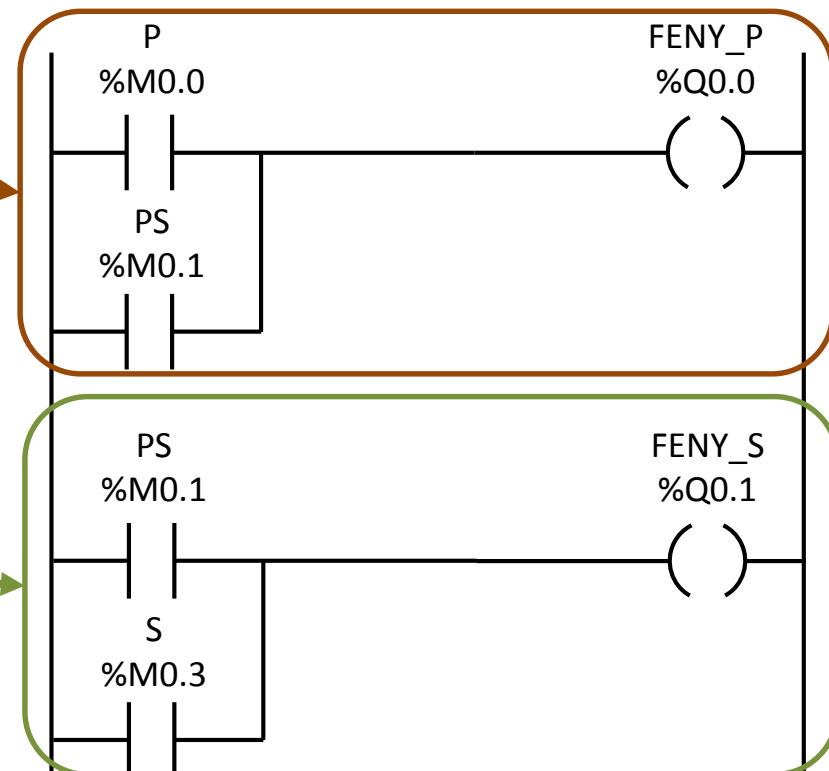
Állapot	P	PS	Z	S	H
Piros fény	1	1	0	0	0
Sárga fény	0	1	0	1	0
Zöld fény	0	0	1	0	0
Villogó sárga	0	0	0	0	1

**Ezt a megvalósítást
választva a P állapotban
egyik fény sem világít!**



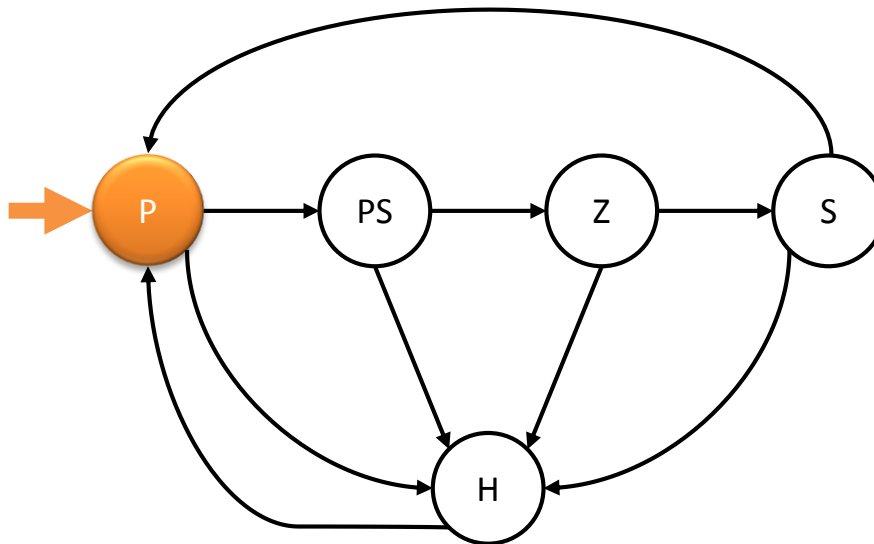
A kimeneti leképezés megvalósítása

Állapot	P	PS	Z	S	H
Piros fény	1	1	0	0	0
Sárga fény	0	1	0	1	0
Zöld fény	0	0	1	0	0
Villogó sárga	0	0	0	0	1



A kezdeti állapot

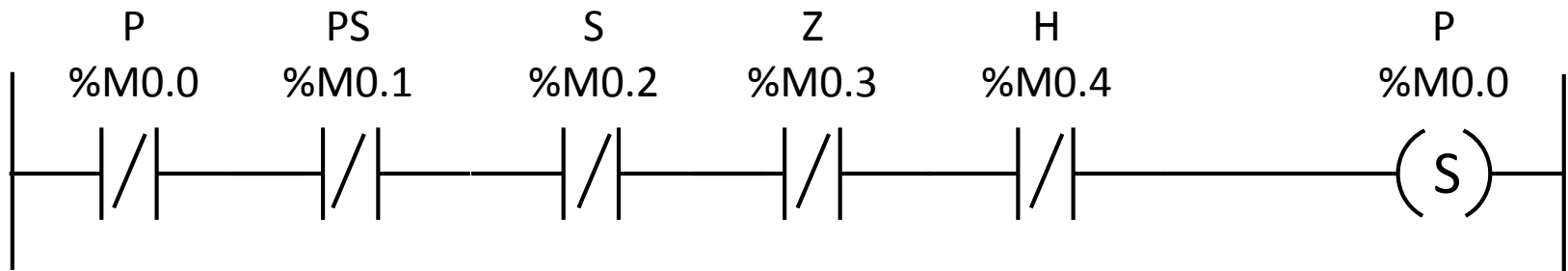
- Melyik állapot lesz aktív, amikor az állapotgépet „bekapcsoljuk”?



Az állapotgép inicializálása

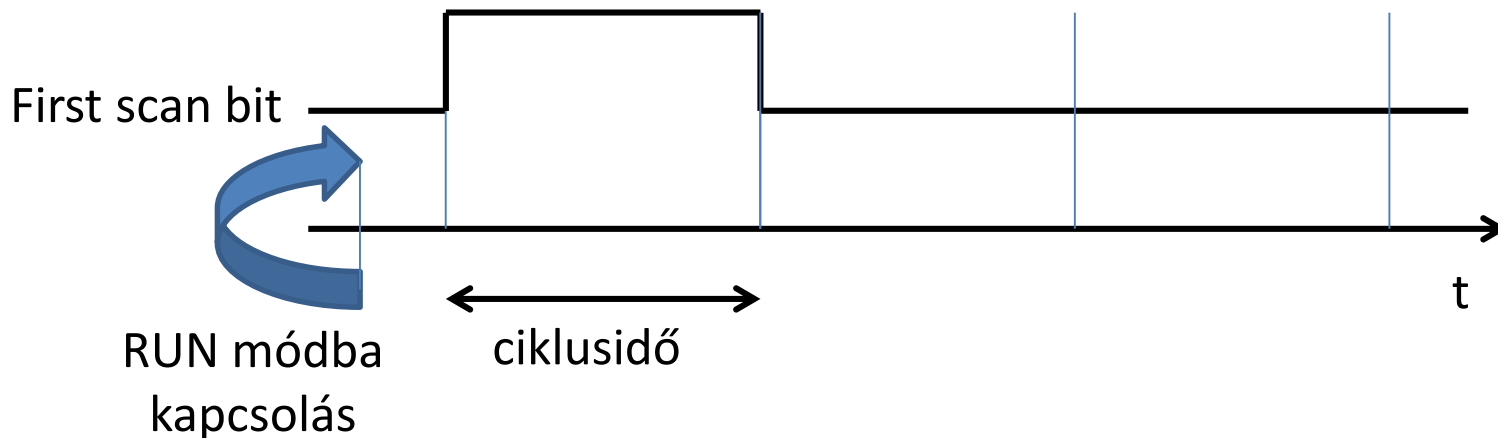
- Elv: ha nincsen aktuális állapot, akkor a kezdeti állapotot állítsuk be aktuálisnak
- Általános megoldás
 - Minden PLC-típuson működik
 - Semmit sem kell tudnunk a belső státuszregiszterekről

Állapotgép inicializálása

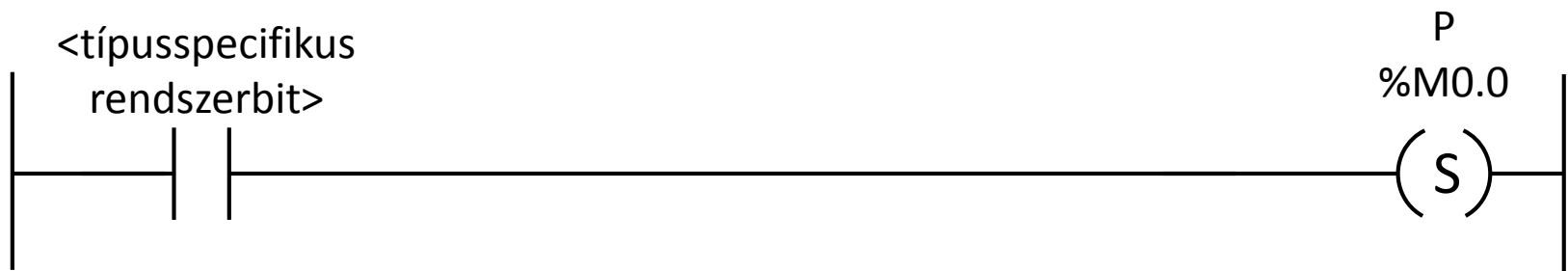


Típus-specifikus megoldás

- A PLC státuszinformációinak használata
- A PLC-k egy biten jelzik, ha az adott ciklus a RUN módba állítás utáni első ciklus (*first scan*)
 - Schneider TWIDO: %S13 rendszerbit
 - Siemens S7-1200: %MBx bájt 0. bitje

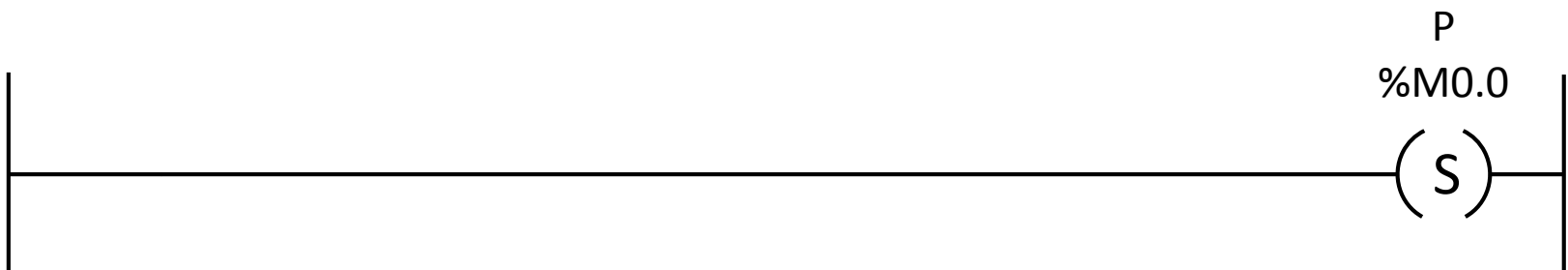


Állapotgép inicializálása



Típuspecifikus megoldás

- Számos környezetben definiálható olyan programblokk, ami kizárólag az első ciklusban fut le, a felhasználói program más részeinek kiértékelése előtt (pl. Siemens OB100)
- Ebben a kezdeti állapot regisztere feltétel nélkül állítható be



Formálisan

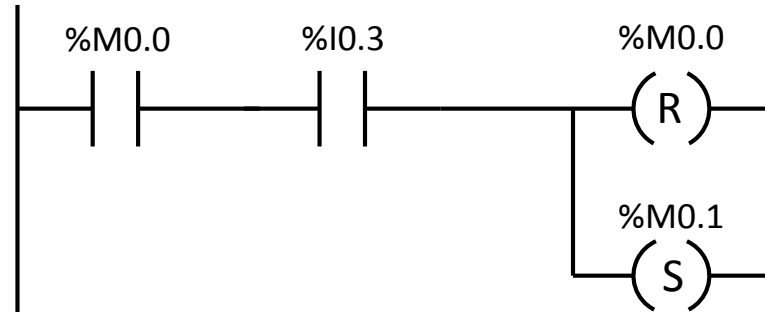
- A Moore automata egy hatos: $C = (Q, \Lambda, \Omega, \rho, \varphi, q_0)$
 - Q : állapotok halmaza
 - Λ : bemeneti szimbólumok (feltételek és események) halmaza
 - Ω : kimeneti szimbólumok halmaza
 - $\rho: Q \times \Sigma \rightarrow Q$: állapotátmeneti függvény
 - $\varphi: Q \rightarrow \Omega$: kimeneti leképezés
 - q_0 : kezdeti állapot

Praktikusan: állapotgép =

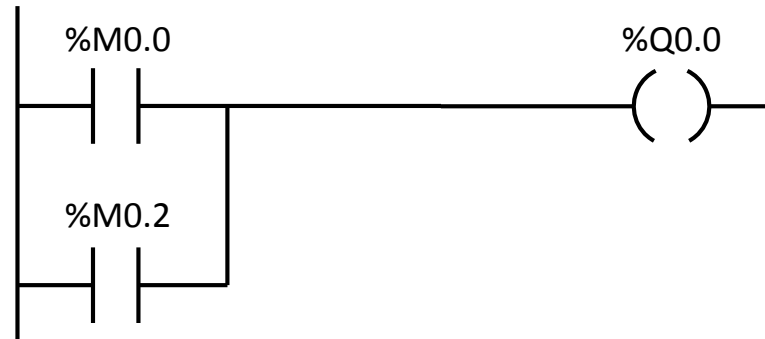
állapotok

%M0.0, %M0.1, %M0.2...

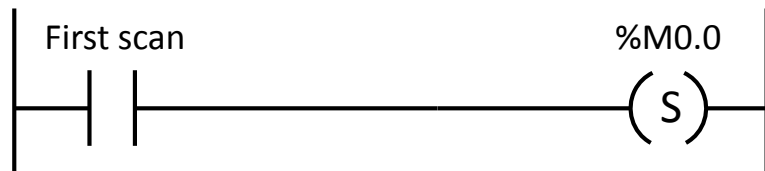
+ átmenetek



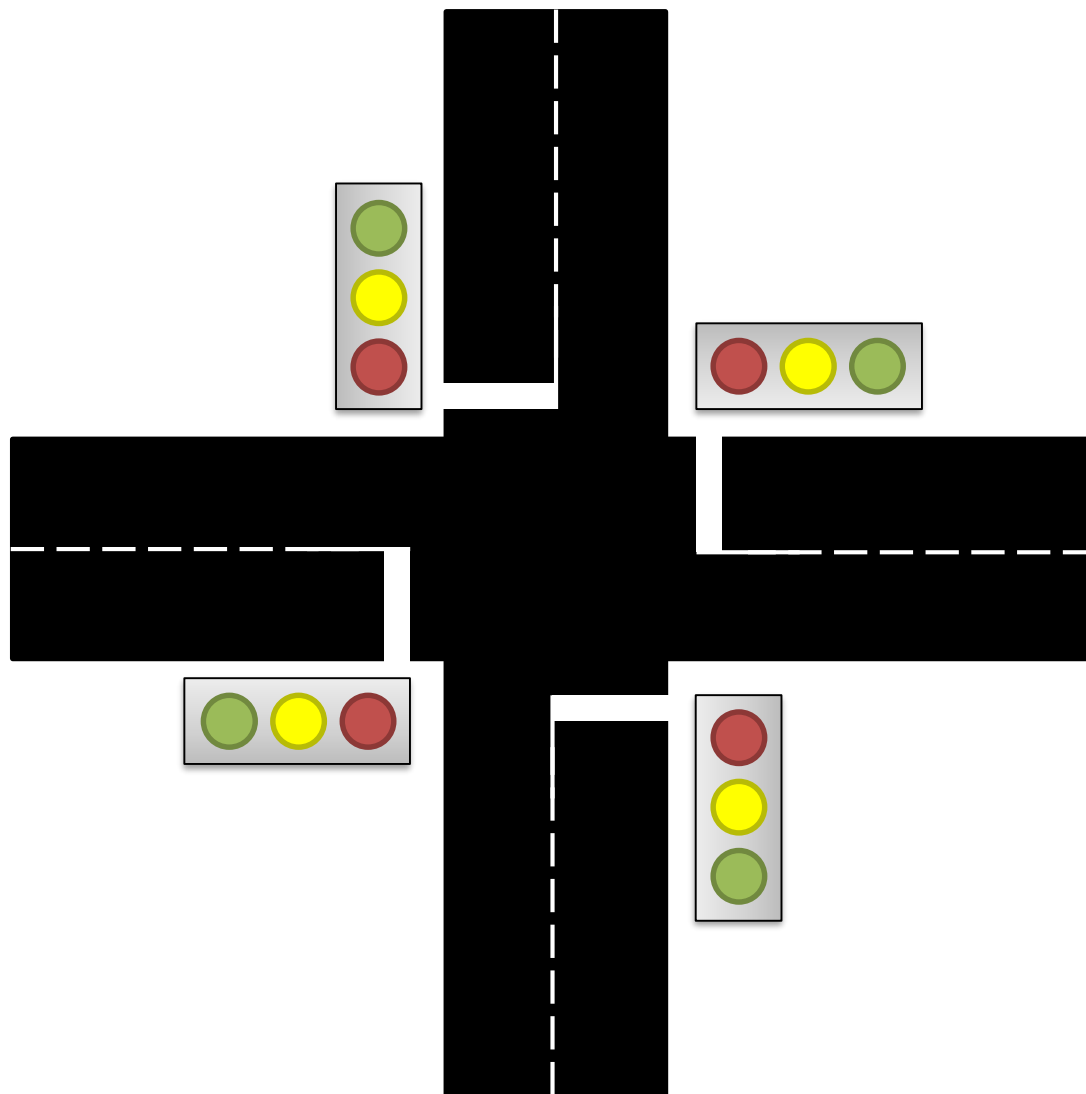
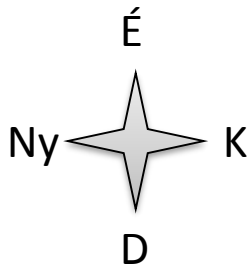
+ kimeneti leképezés



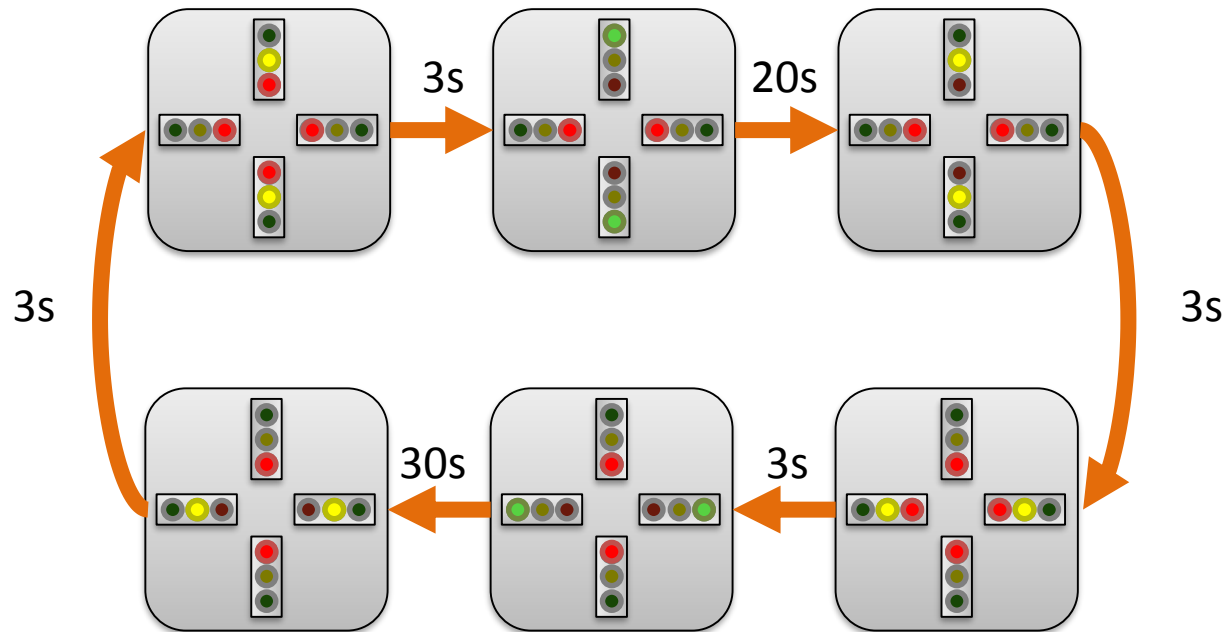
+ kezdeti állapot



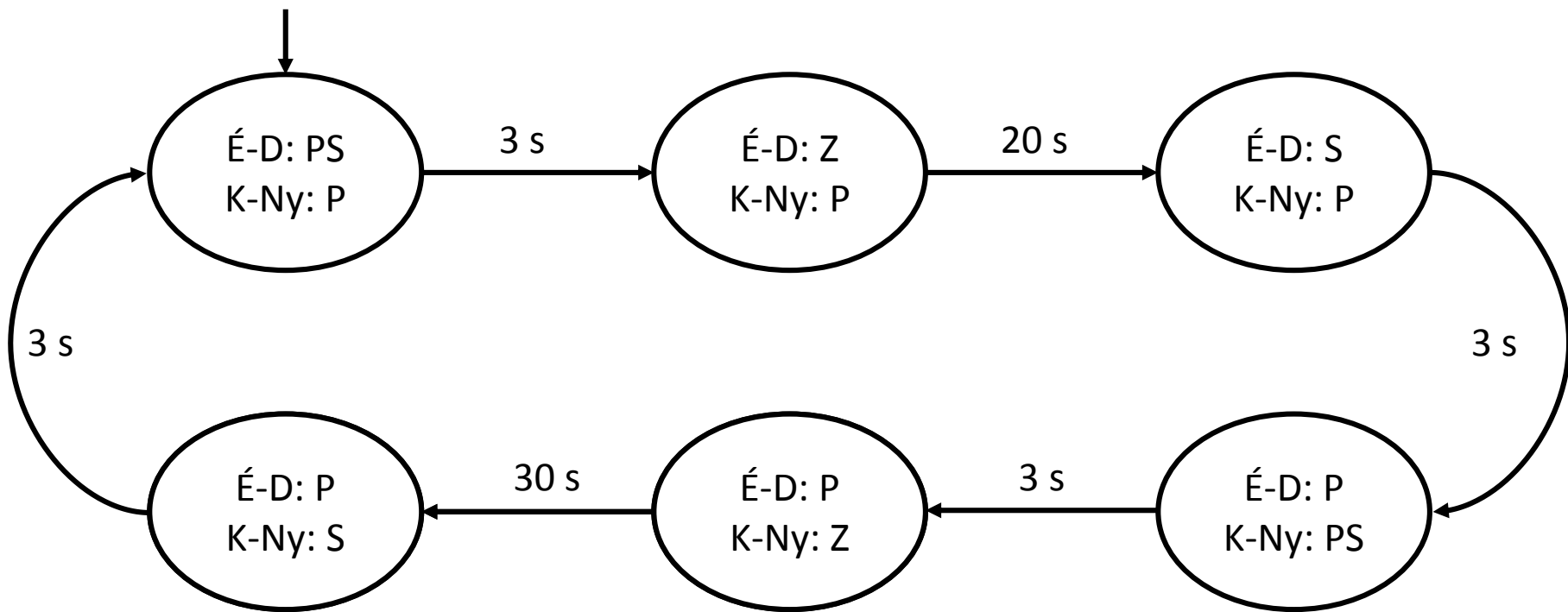
Jelzőlámpás kereszteződés



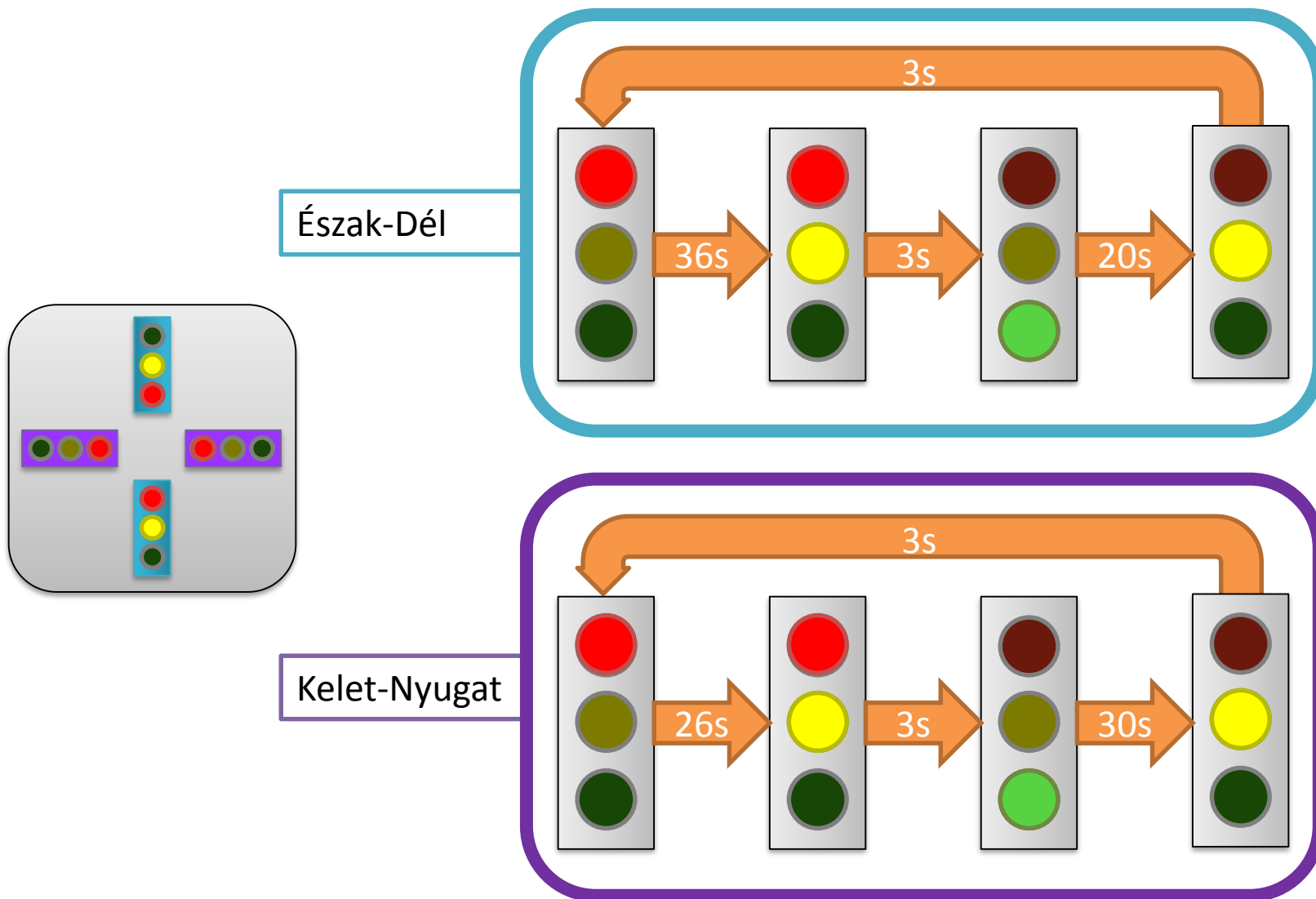
Jelzőlámpás kereszteződés



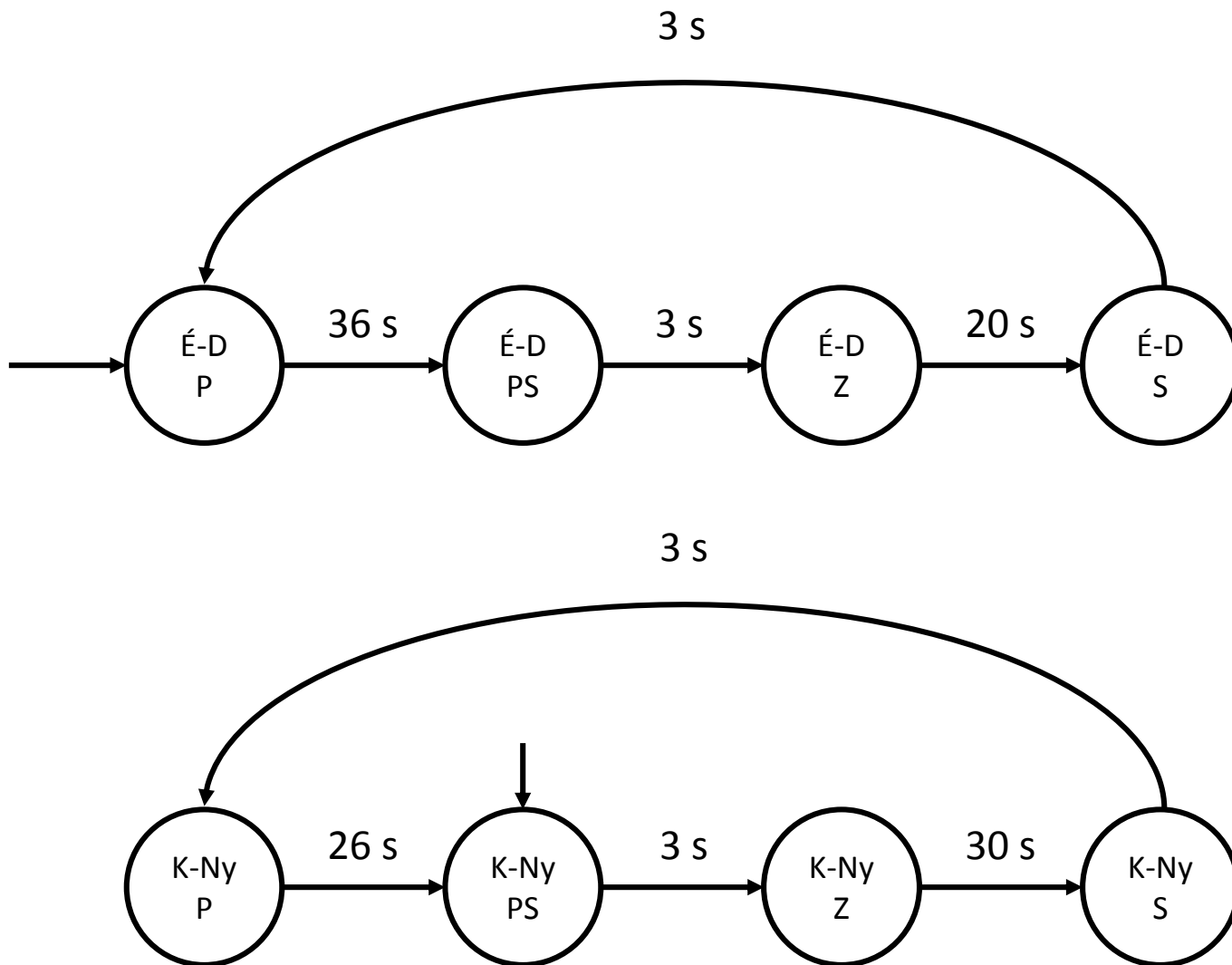
Jelzőlámpás kereszteződés – 1. megoldás



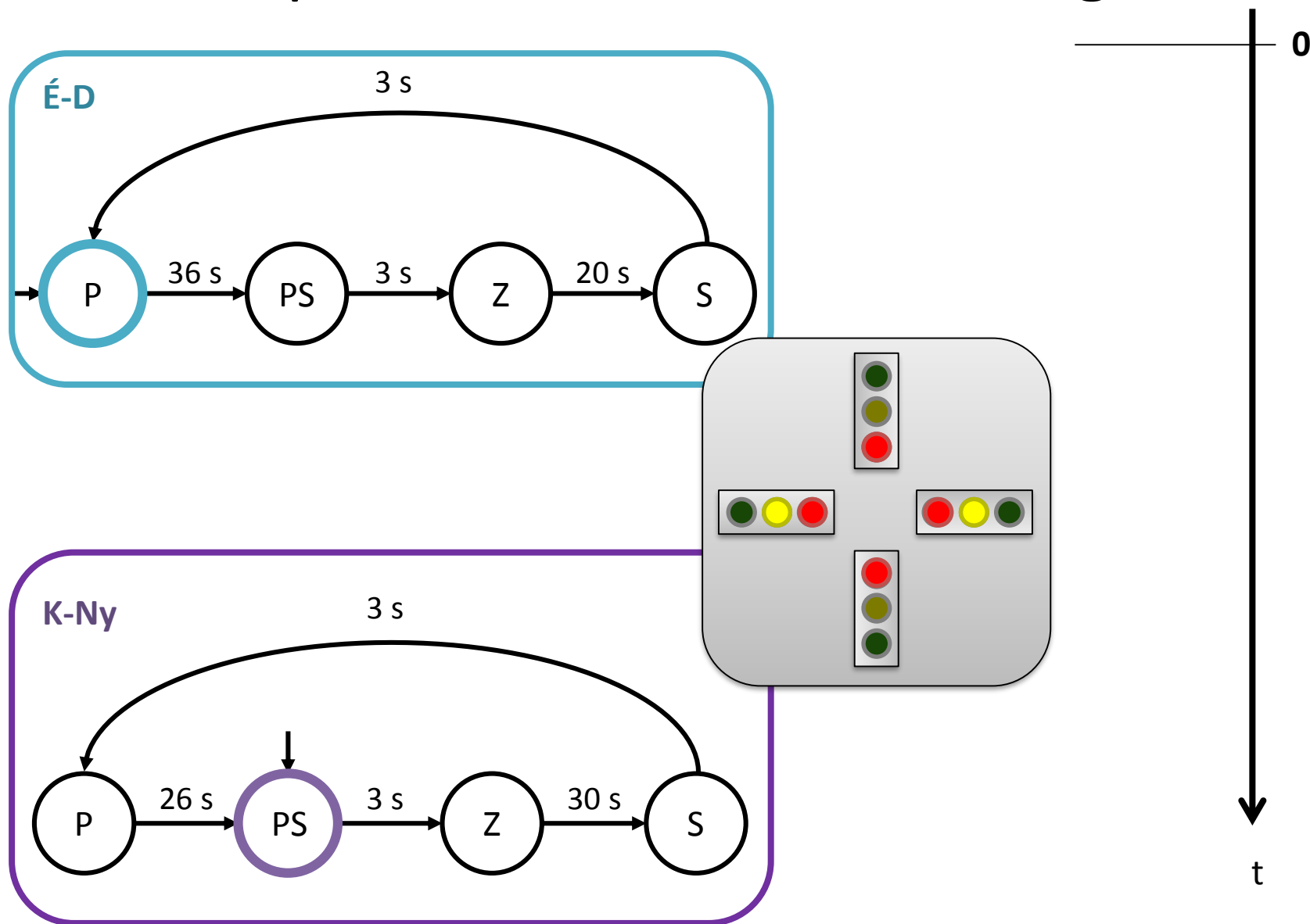
Jelzőlámpás kereszteződés – 2. megoldás



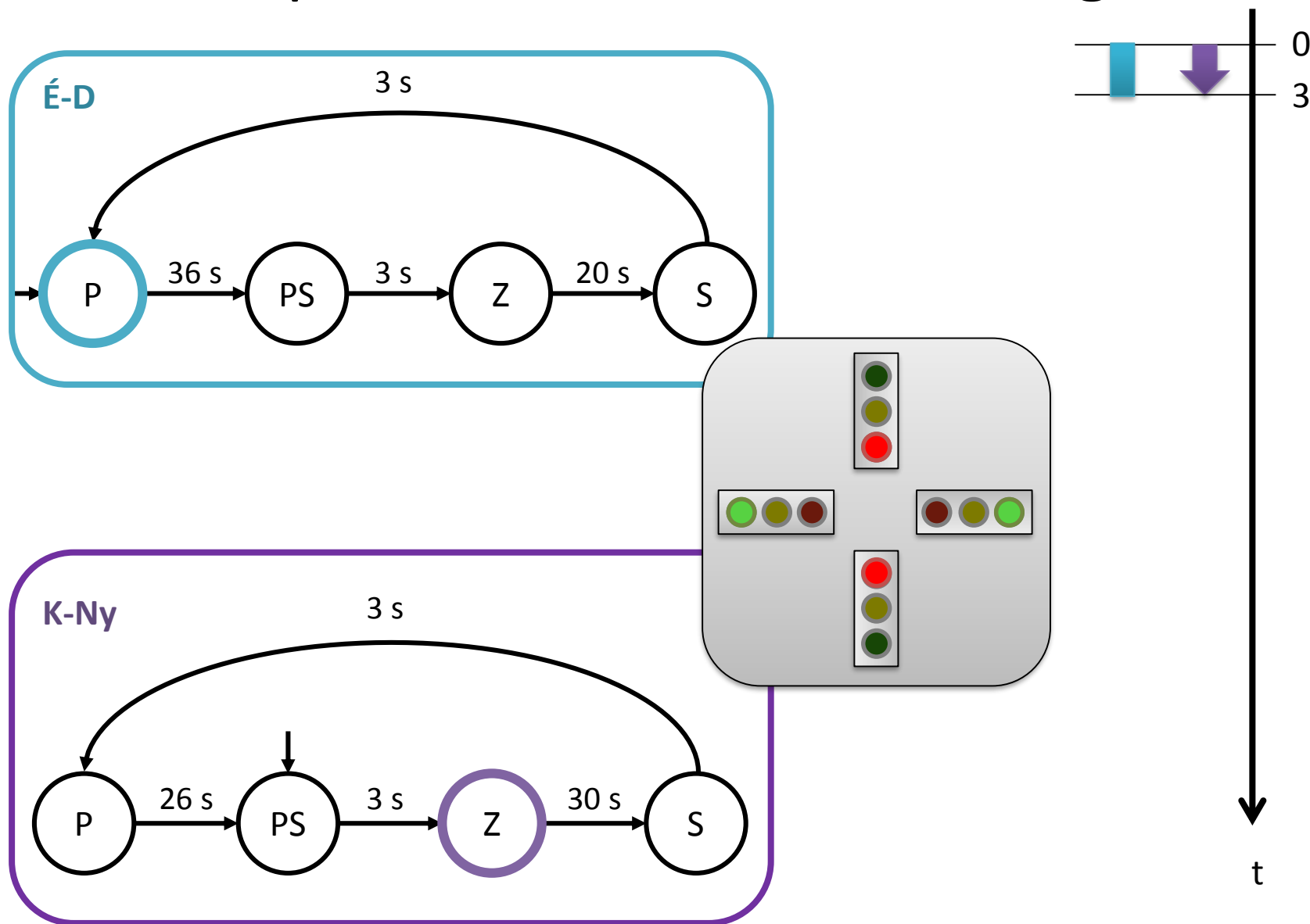
Jelzőlámpás kereszteződés – 2. megoldás



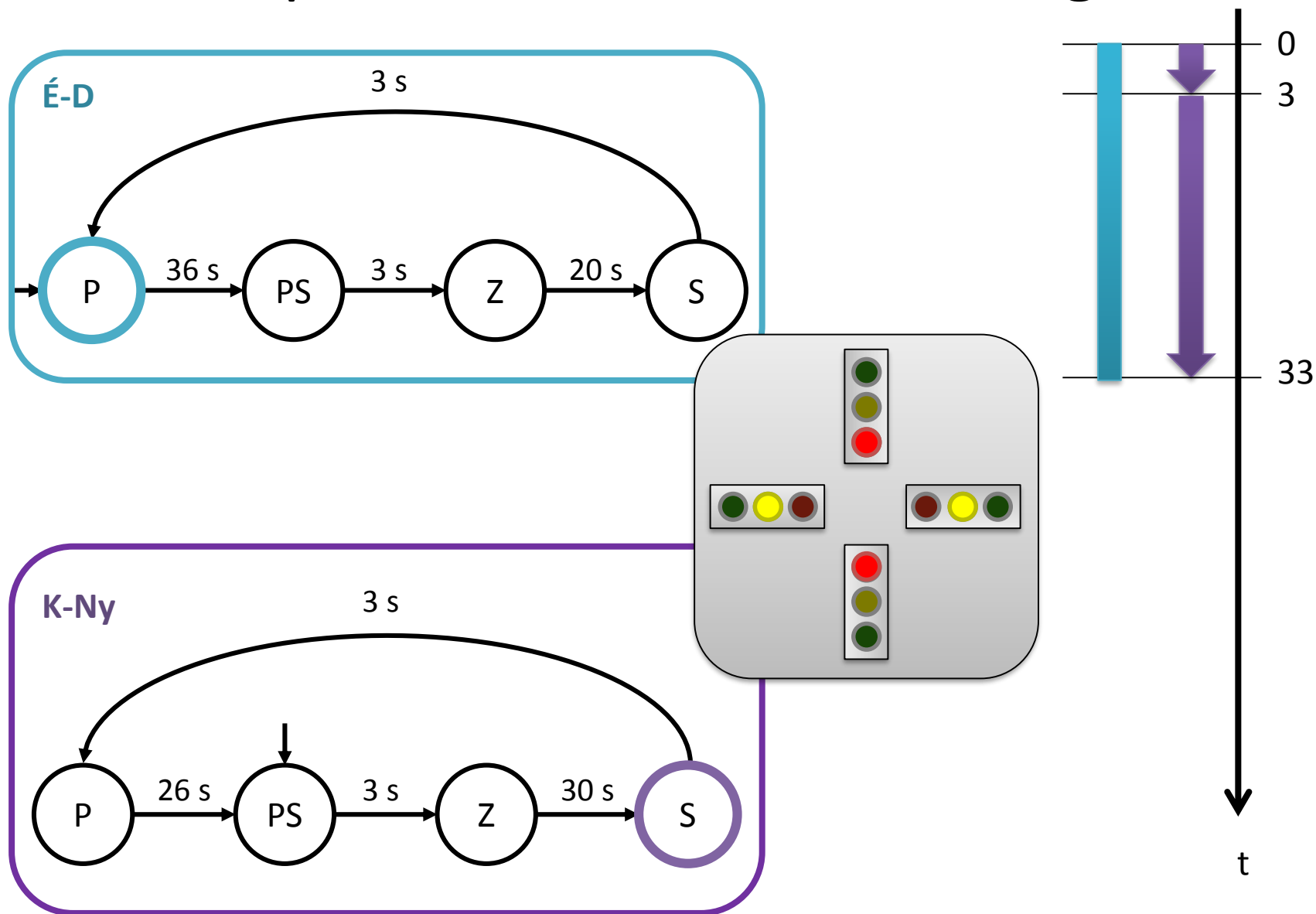
Jelzőlámpás kereszteződés – 2. megoldás



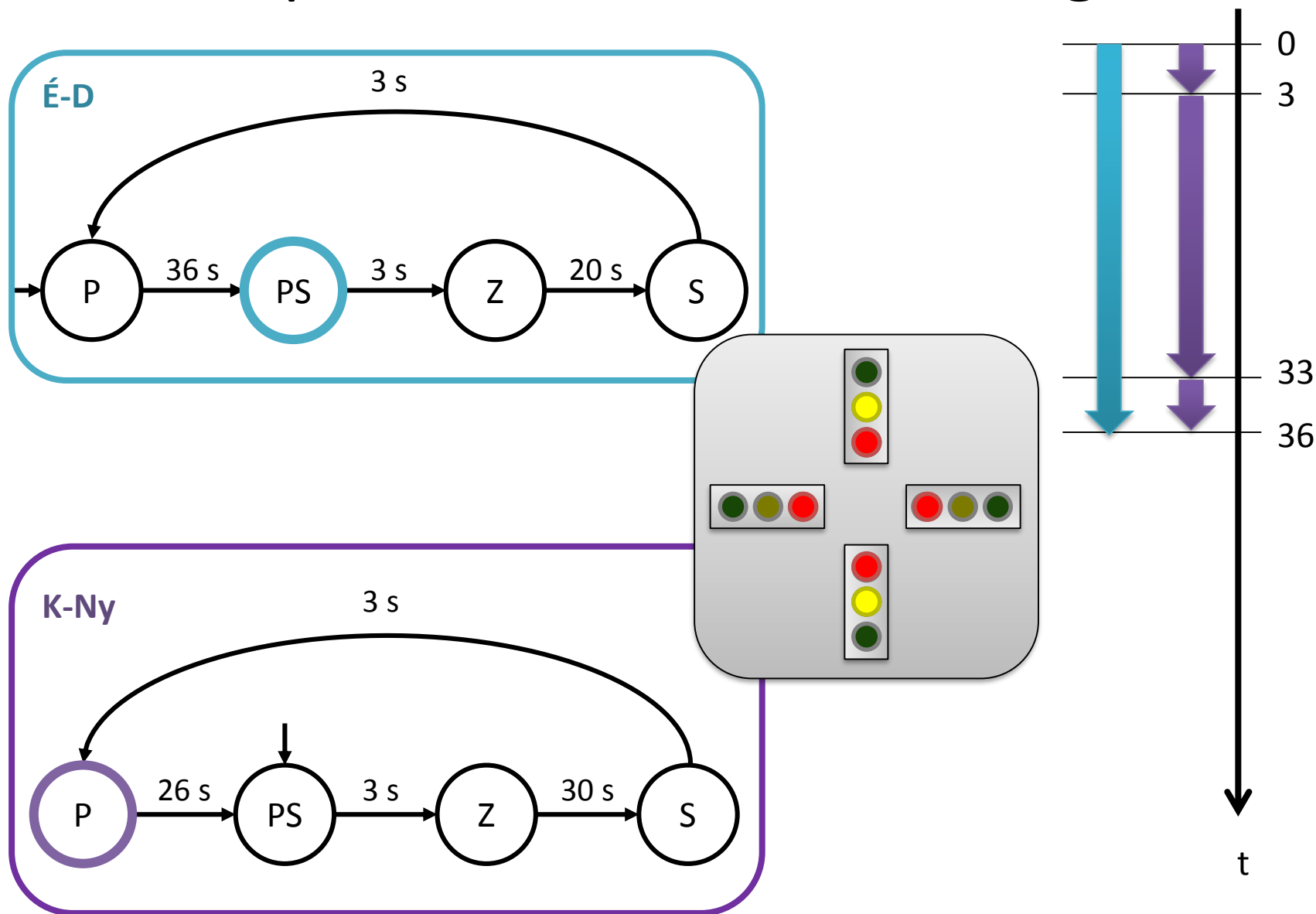
Jelzőlámpás kereszteződés – 2. megoldás



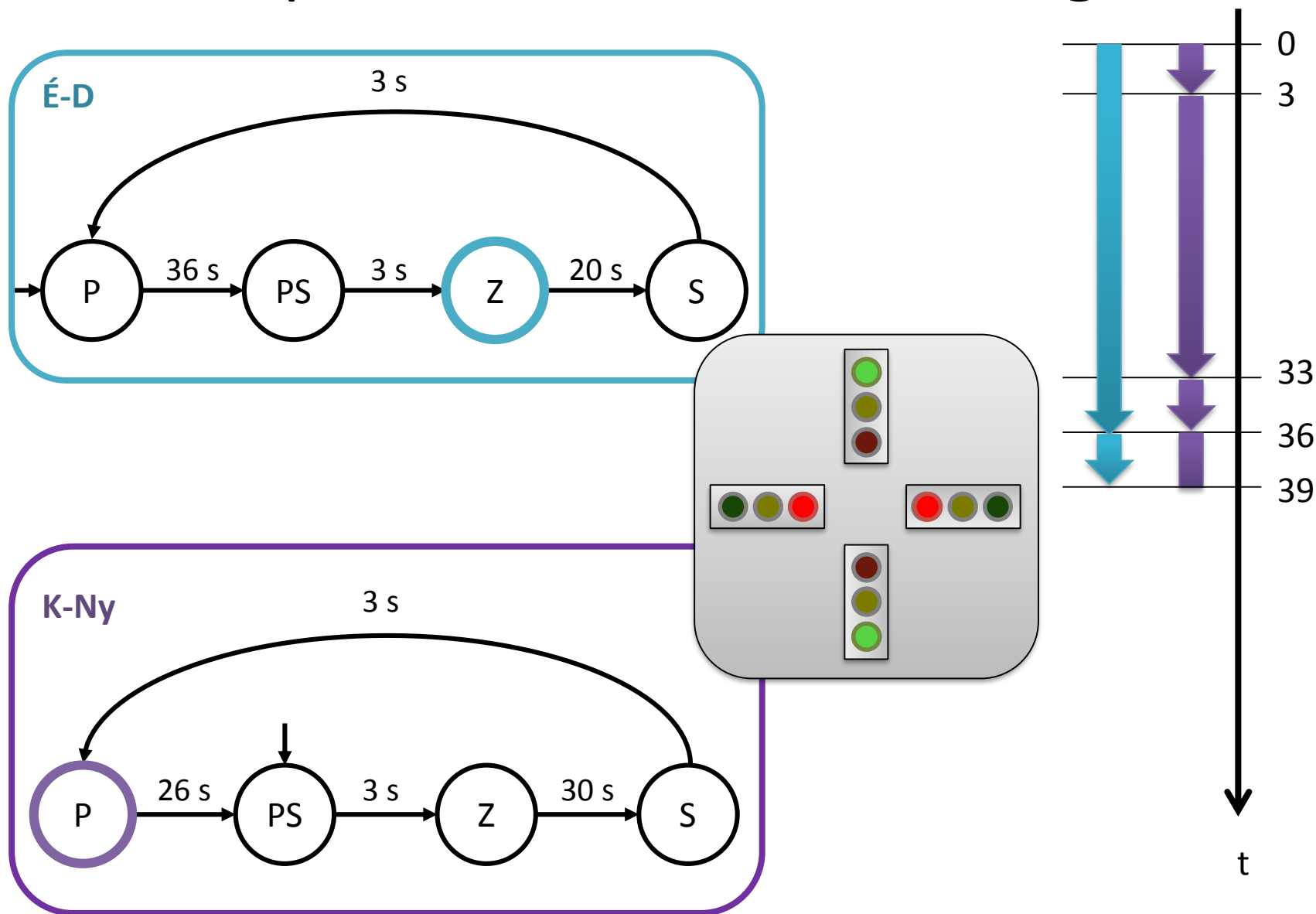
Jelzőlámpás kereszteződés – 2. megoldás



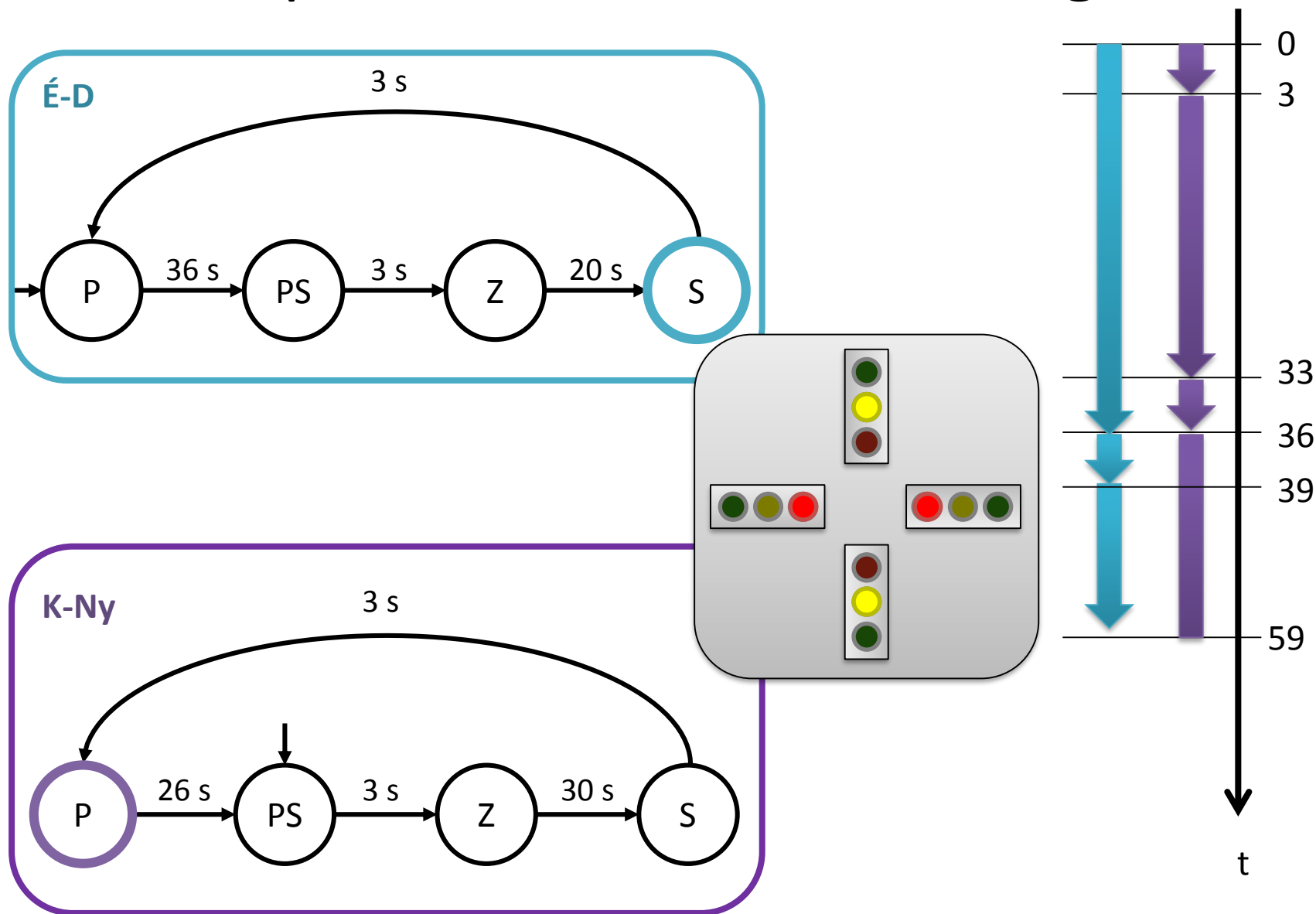
Jelzőlámpás kereszteződés – 2. megoldás



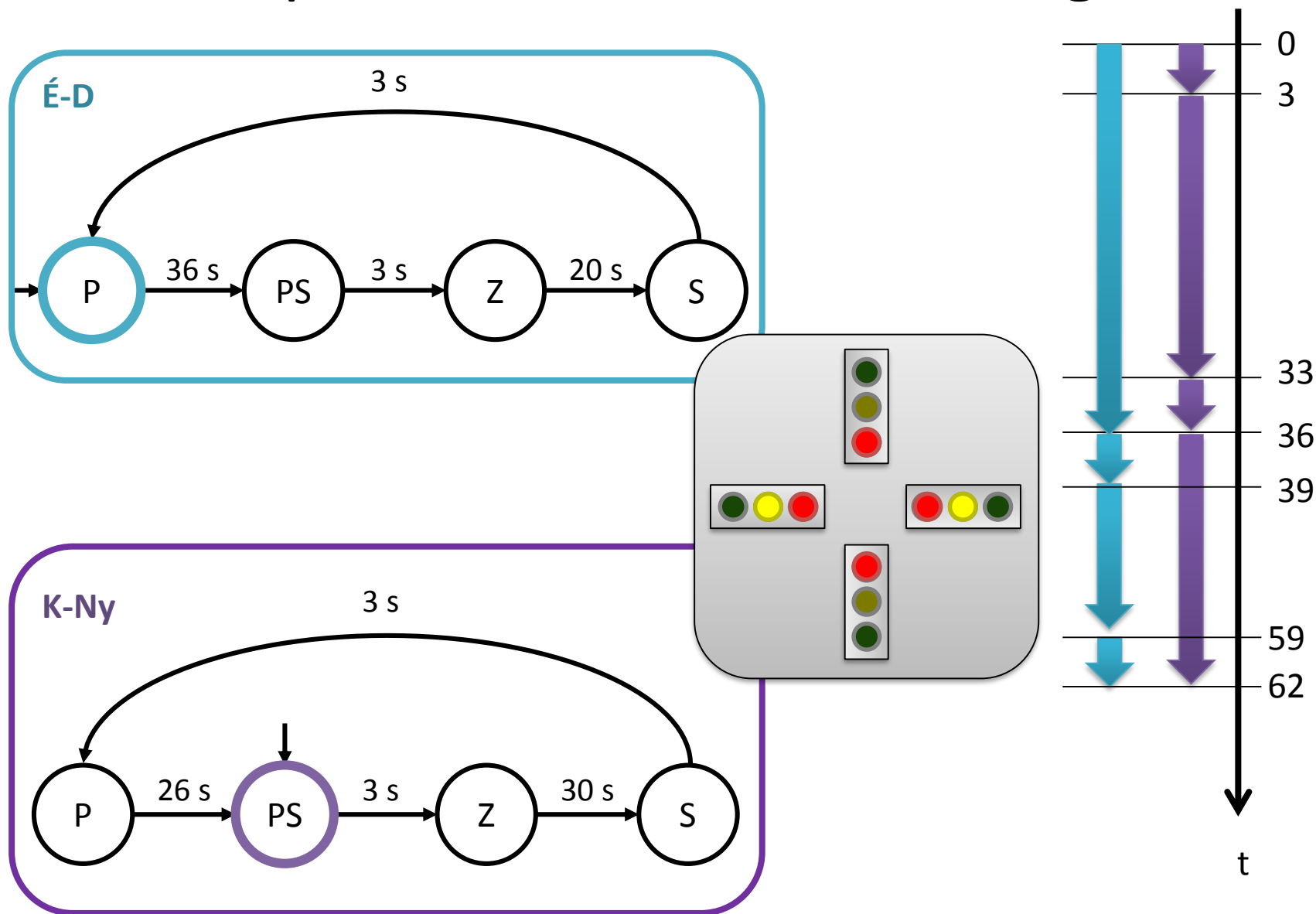
Jelzőlámpás kereszteződés – 2. megoldás



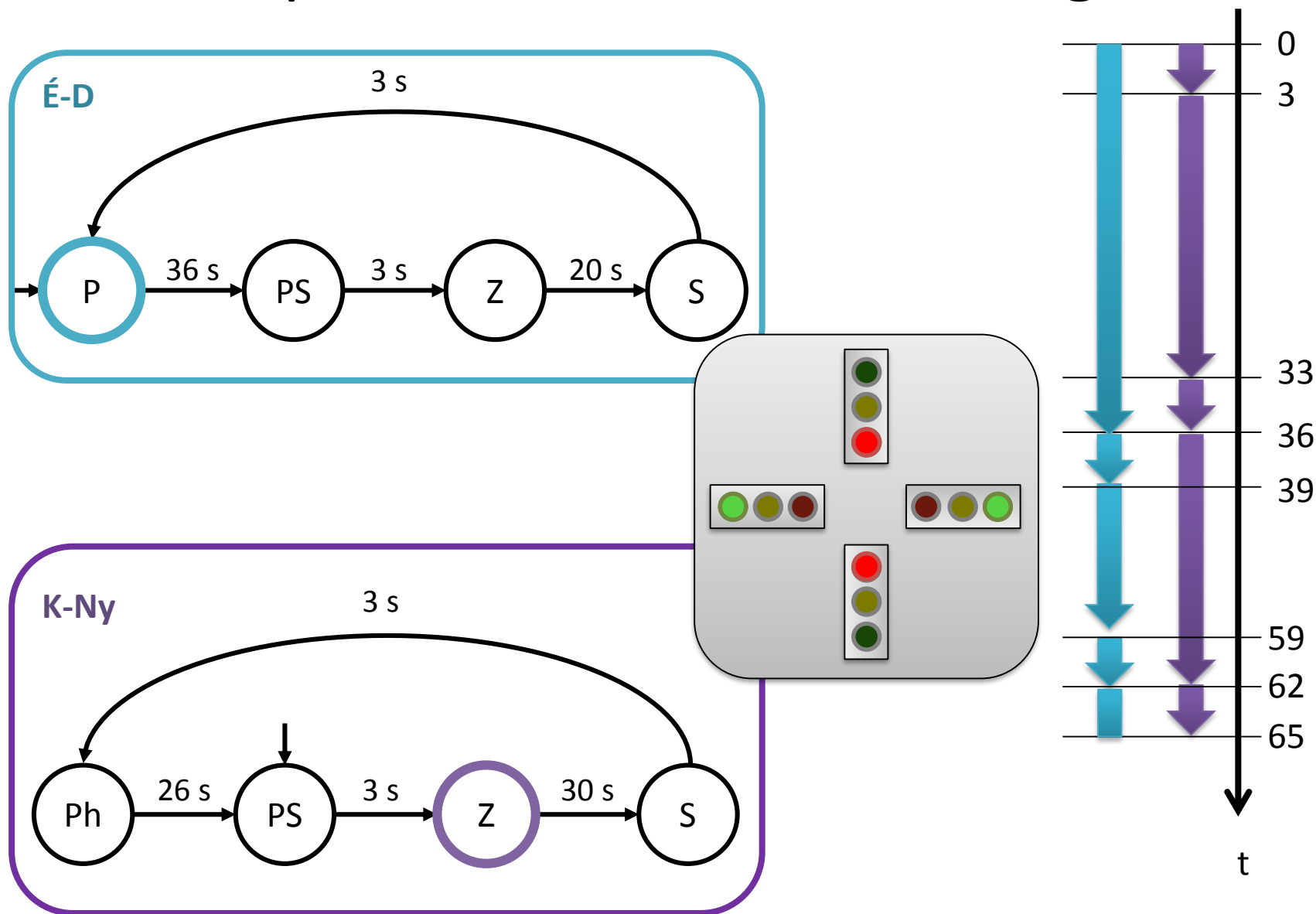
Jelzőlámpás kereszteződés – 2. megoldás



Jelzőlámpás kereszteződés – 2. megoldás



Jelzőlámpás kereszteződés – 2. megoldás



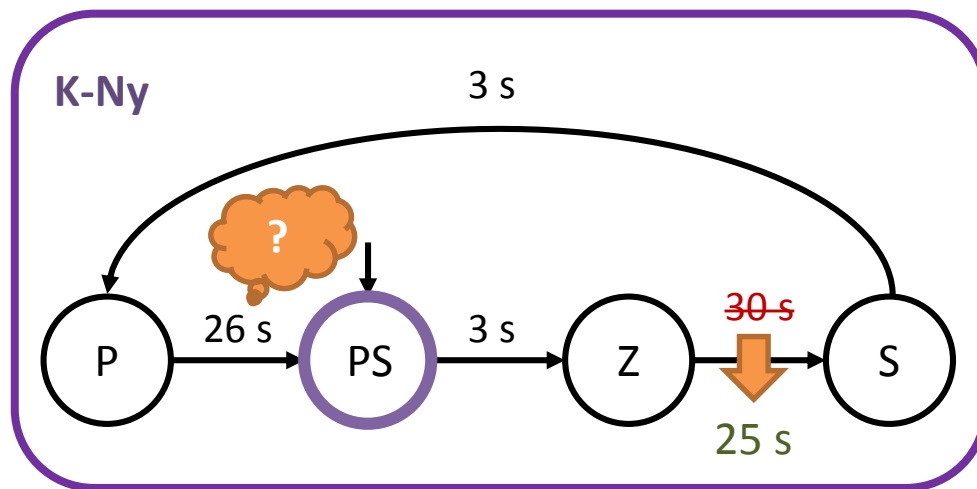
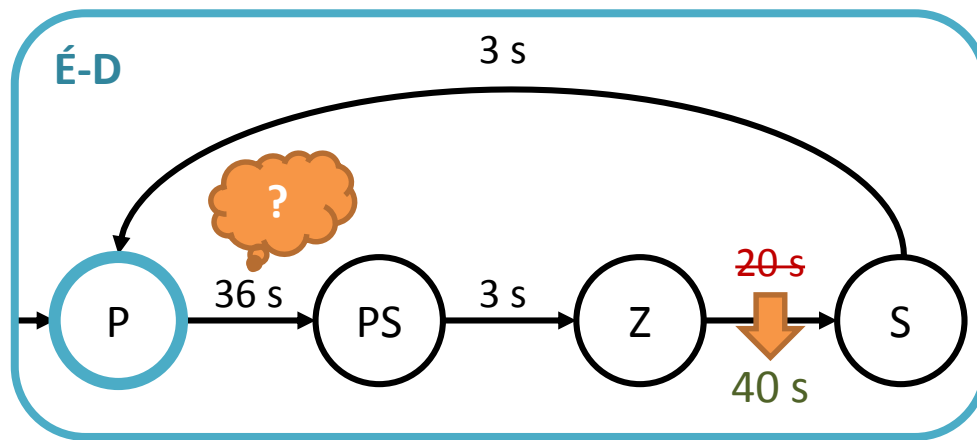
Párhuzamos állapotgépek megvalósítása

- Definiáljunk állapotregisztereket minden állapotgépnek
- Valósítsuk meg az állapotgépek átmeneteit ugyanabban a létradiagramban
- Valósítsuk meg az állapotgépek kimeneti leképezéseit ugyanabban a létradiagramban
- Indításkor állítsuk be minden állapotgép kezdeti állapotát

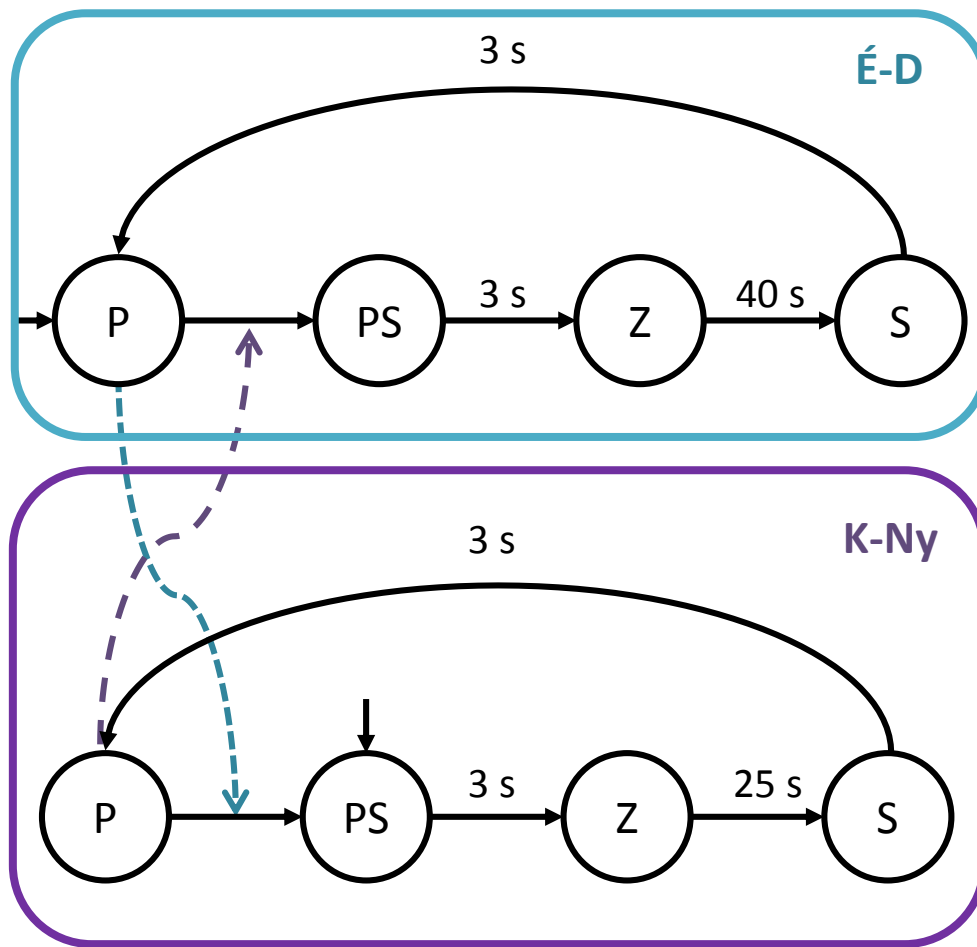
Párhuzamos állapotgépek megvalósítása

- Egy állapotgép átmenete nem állíthatja egy másik állapotgép állapotregisztereit
- Közös kimenetek esetén a kimeneti leképezésben a különböző állapotgépek állapotait egy létrisorban kell szerepeltetni

Párhuzamos állapotgépek szinkronizálása

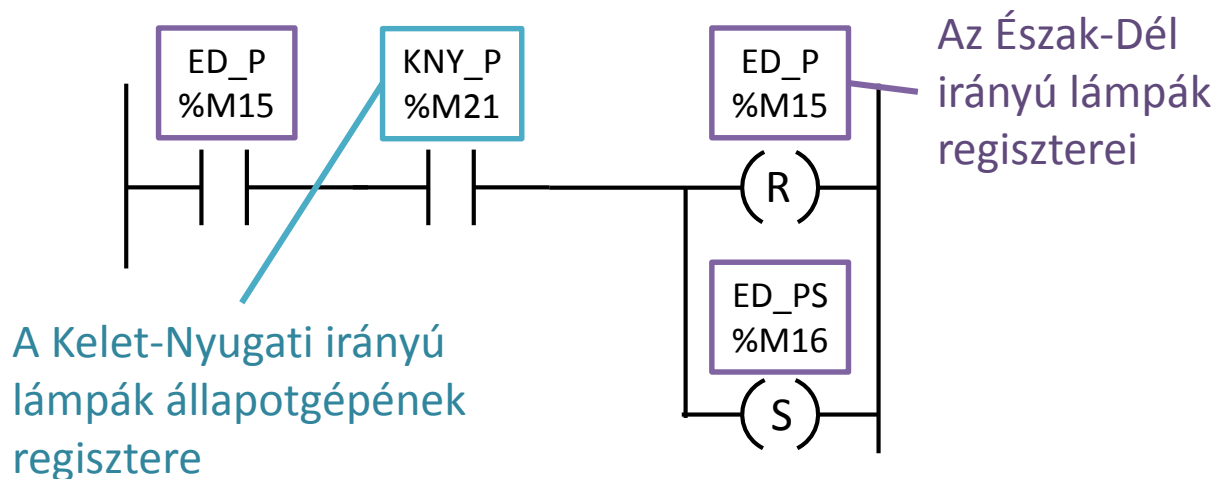


Párhuzamos állapotgépek szinkronizálása

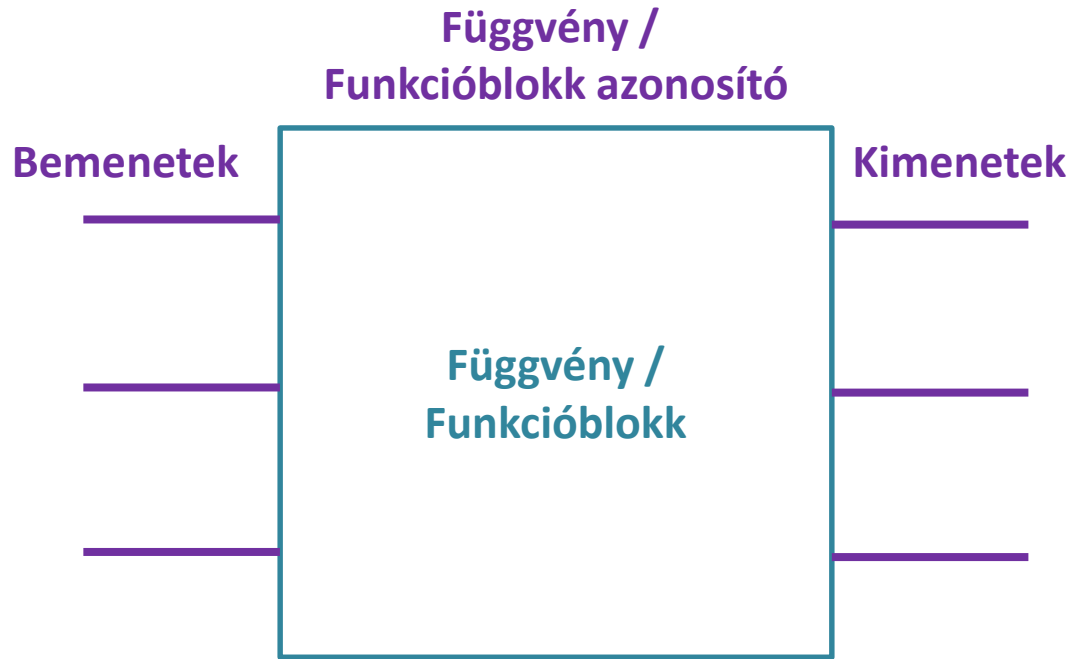


Párhuzamos állapotgépek szinkronizálása

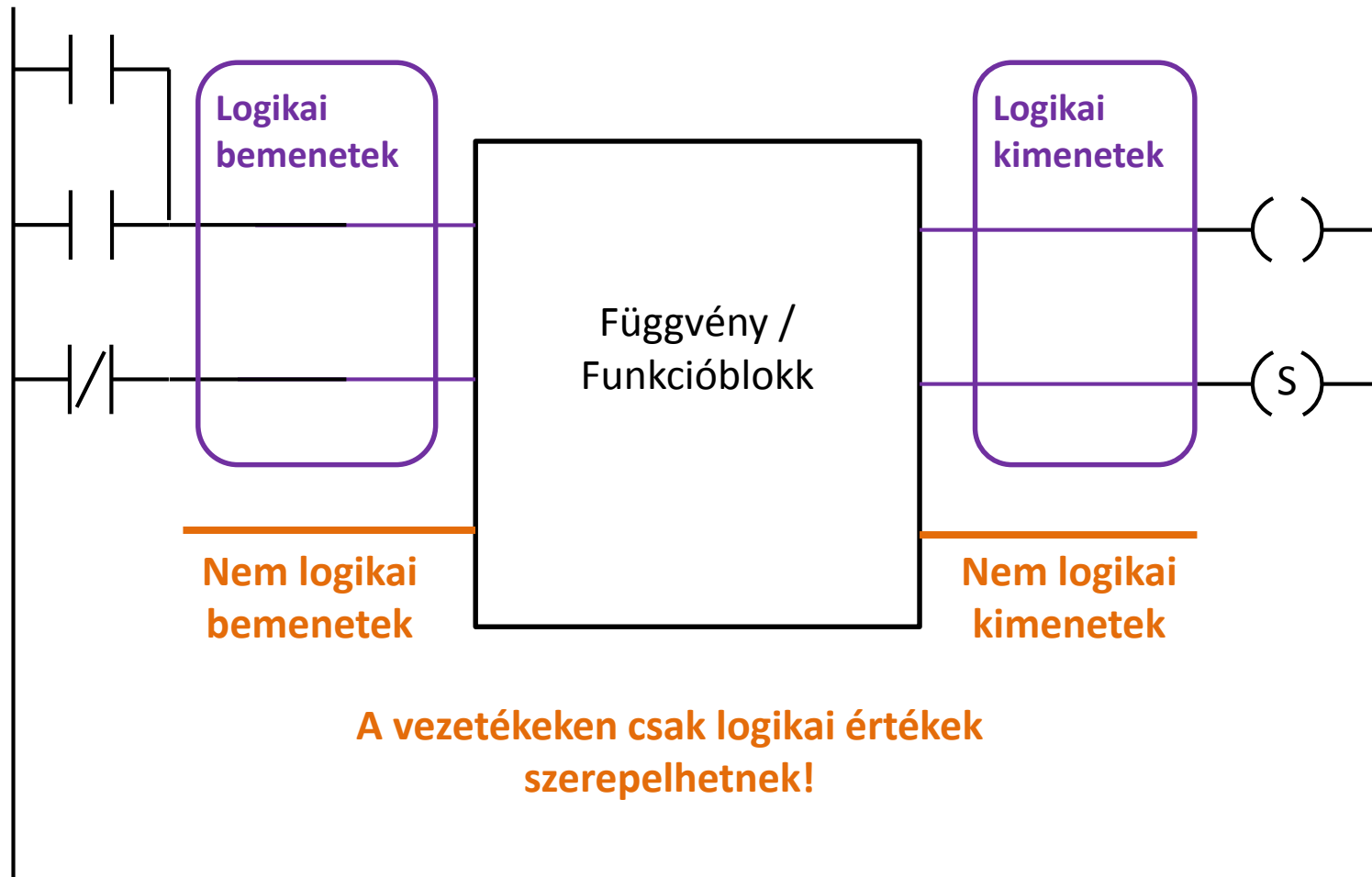
- Egy állapotgépe átmenetének feltételeként használható egy másik állapotgép állapotregisztere
- Egy állapotgép állapotváltása egy másik állapotgép állapotátmenetét okozhatja



Funkcióblokkok és függvény-blokkok a létradiagramban



Funkcióblokkok és függvény-blokkok a létradiagramban



Funkcióblokkok / Függvény-blokkok

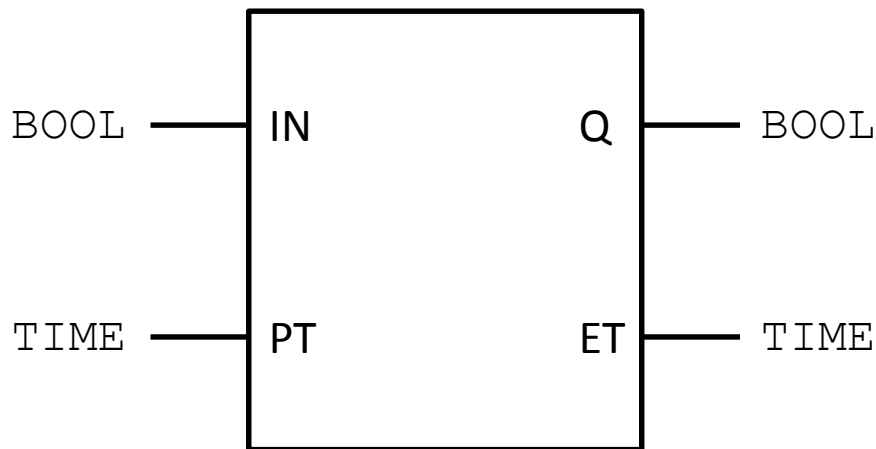
- Kész funkcióblokkok
 - Szabványos (IEC-61131)
 - Gyártóspecifikus
- Felhasználói blokkok

Időzítők



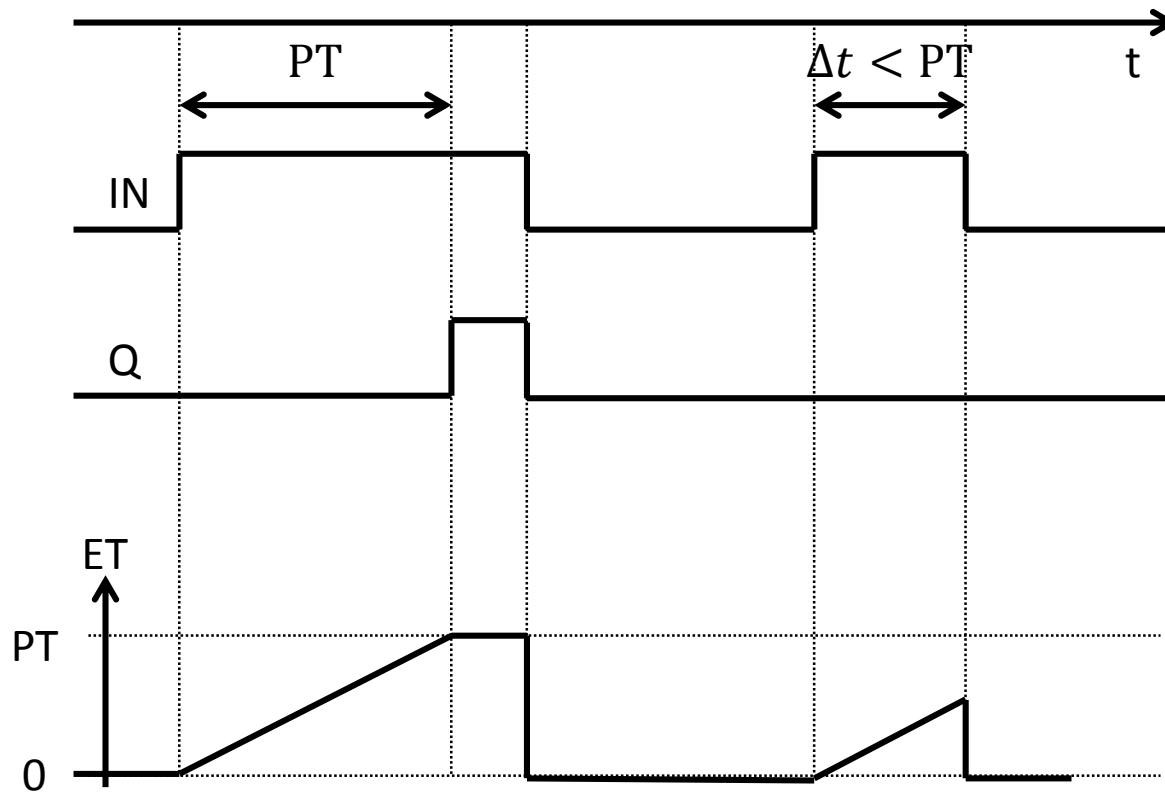
Időzítők

Szabványos megvalósítás

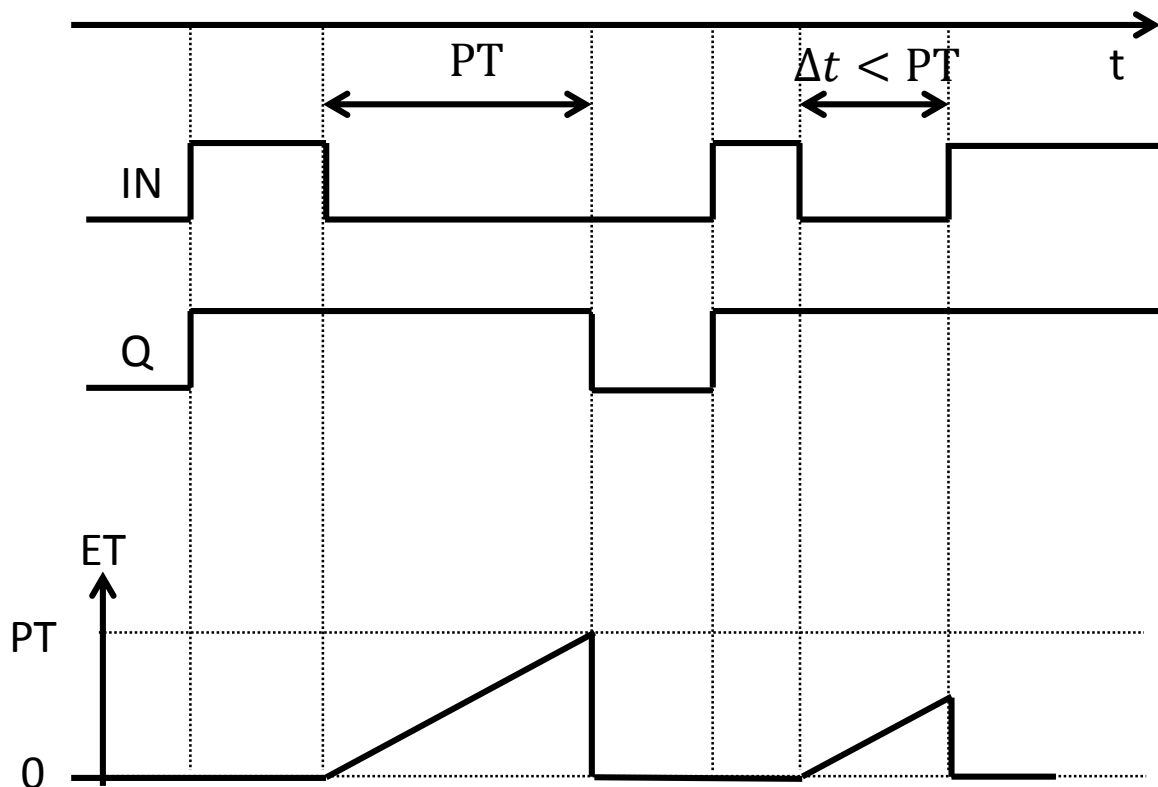


- IN: Számláló bemenet (*timer input*)
- PT: Késleltetés (*timer preset*)
- Q: Számláló kimenet (*timer output*)
- ET: Eltelt idő (*elapsed time*)

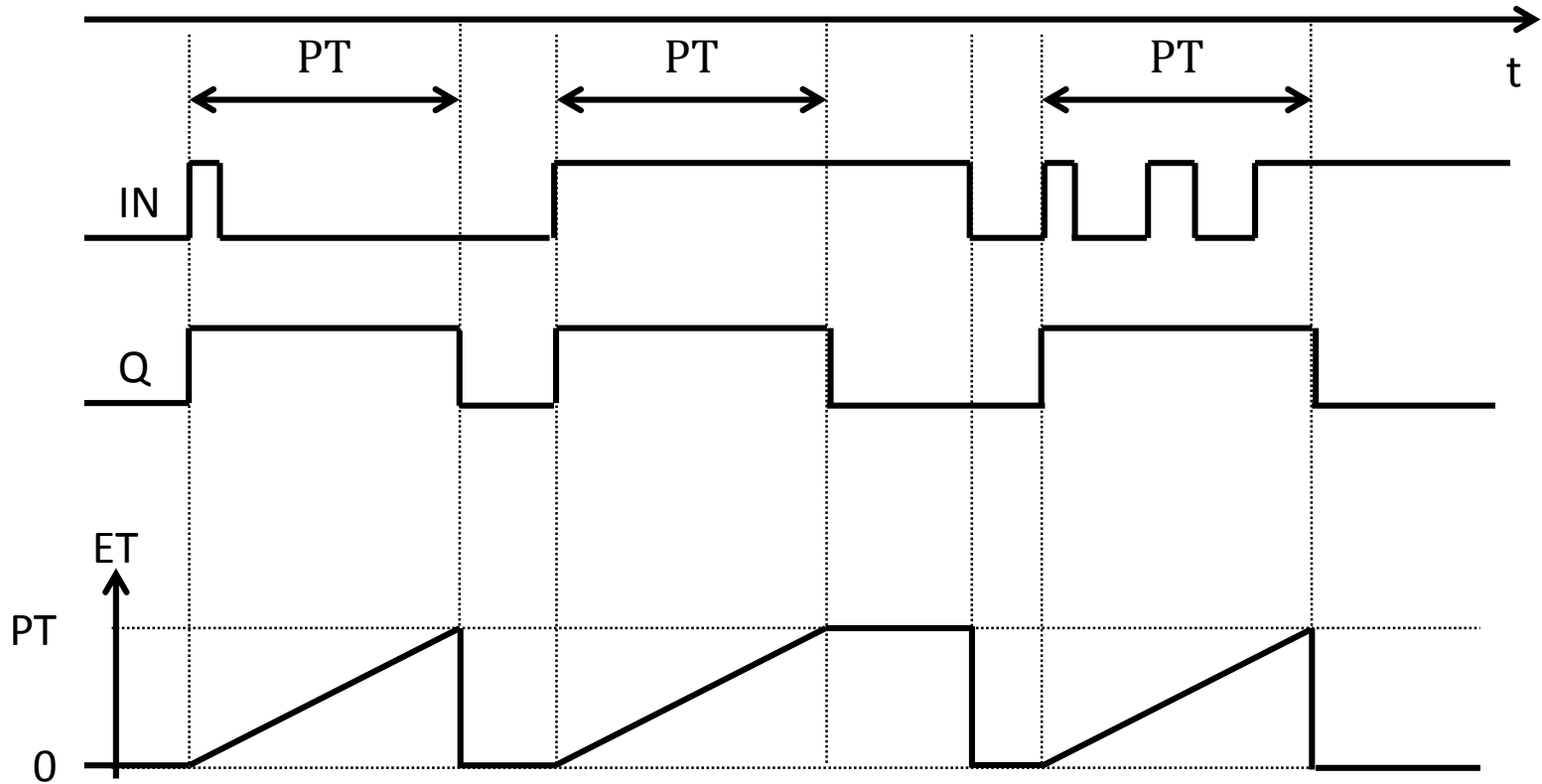
Bekapcsolás-időzítő (*On-delay timer, TON*)



Kikapcsolás-időzítő (*Off-delay timer, TOF*)



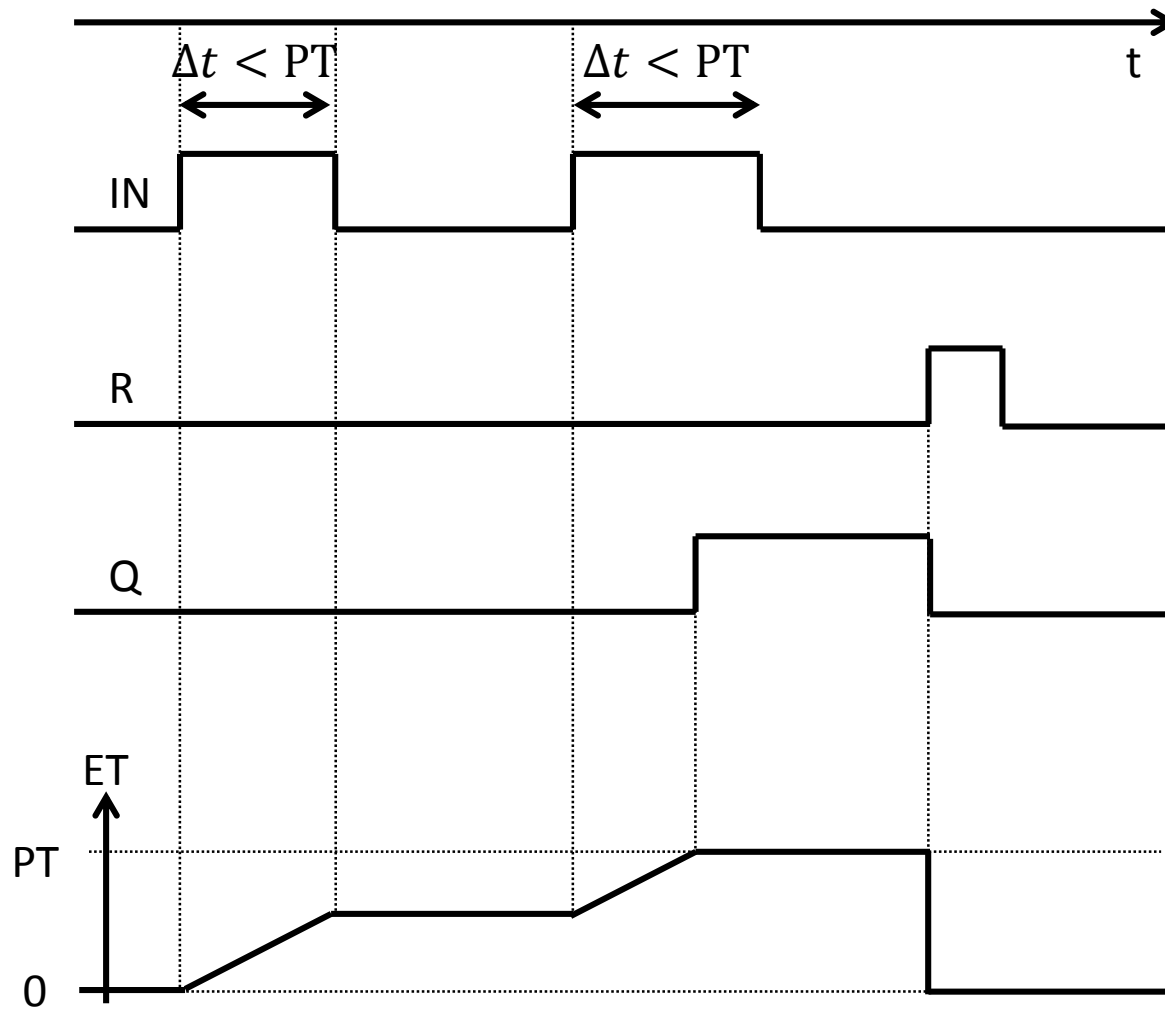
Impulzus időzítő (*Pulse timer, TP*)



Retentív időzítők

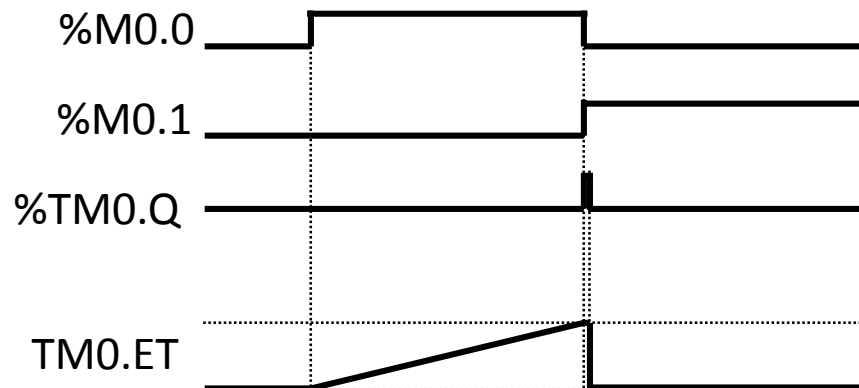
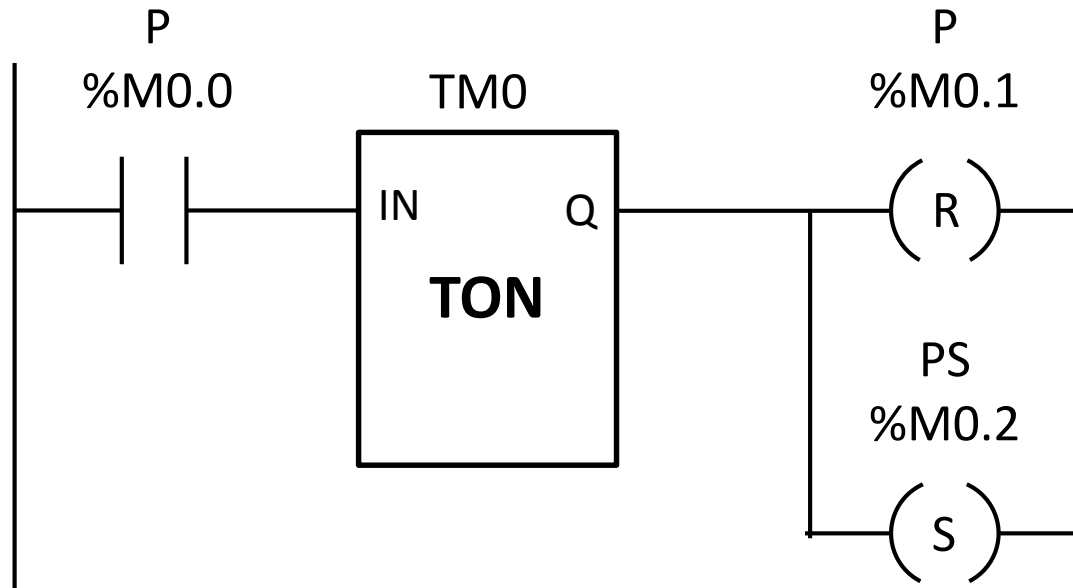
- A retentív TON/TOF időzítők belső számlálóját a bemenet lefutó/felfutó éle nem nullázza
- A bemenet összegzett 1/0 állapotban töltött idejét méri
- Nullázásra külön Reset bemenet
- Nem szabványos, de sok fejlesztőkörnyezetben elérhető

Retentív TON időzítő



Állapotgépek időzítése

Vissza a jelzőlámpához

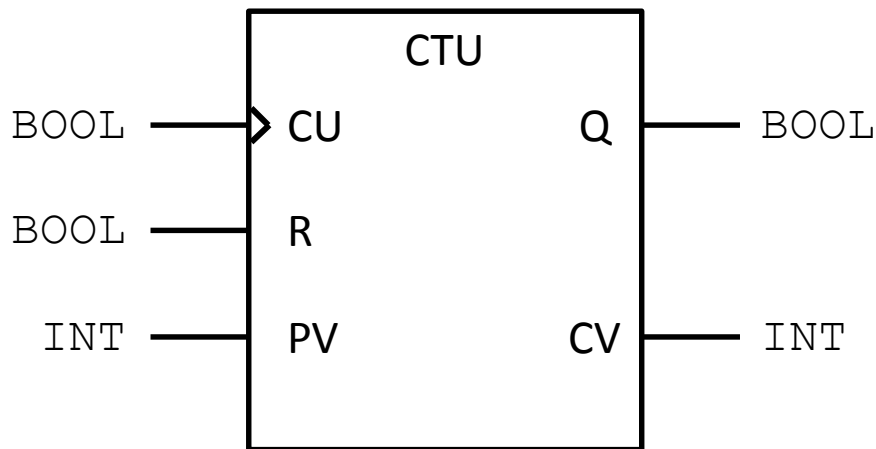


Számlálók



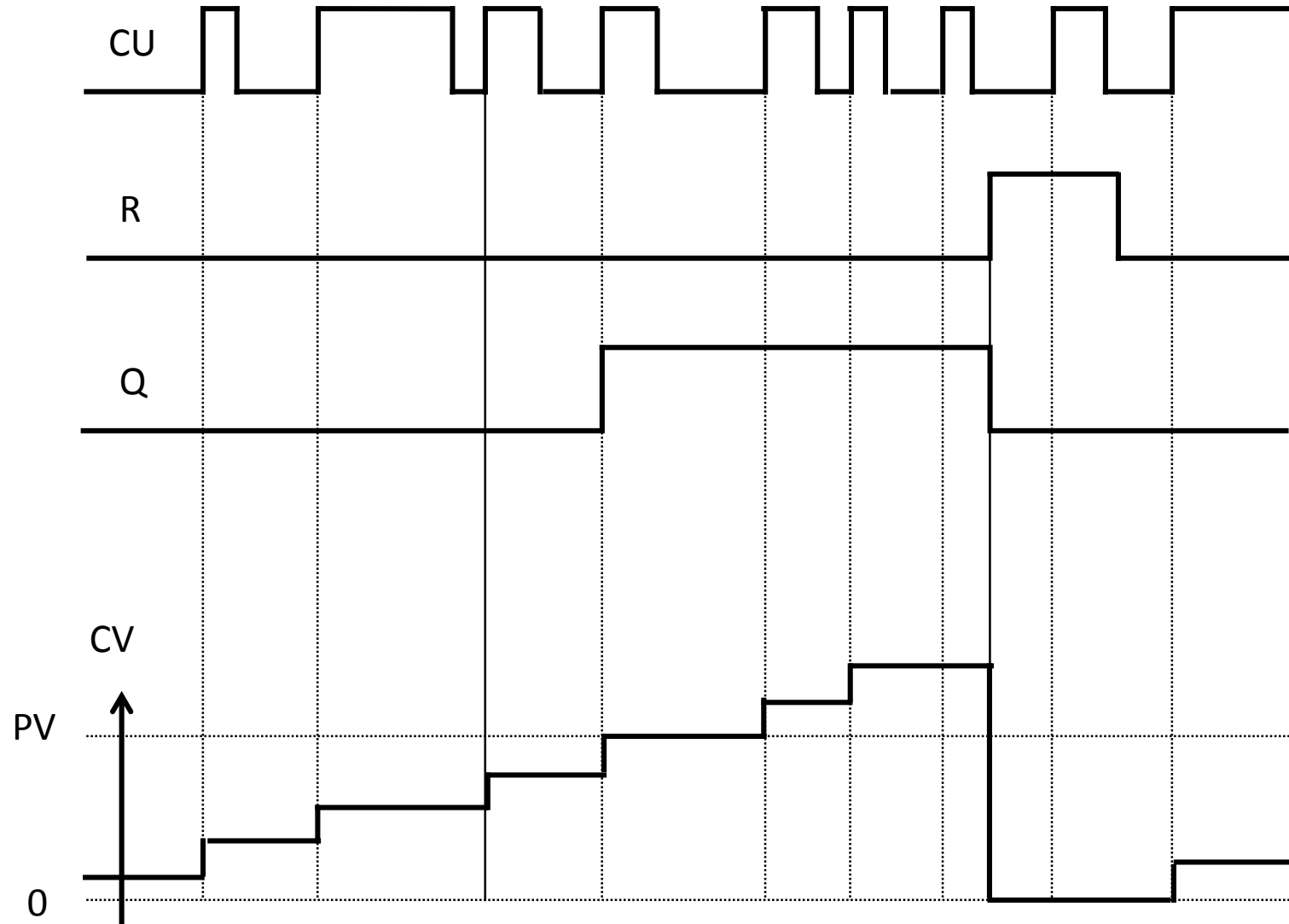
Felfelé számláló (CTU)

Szabványos megvalósítás



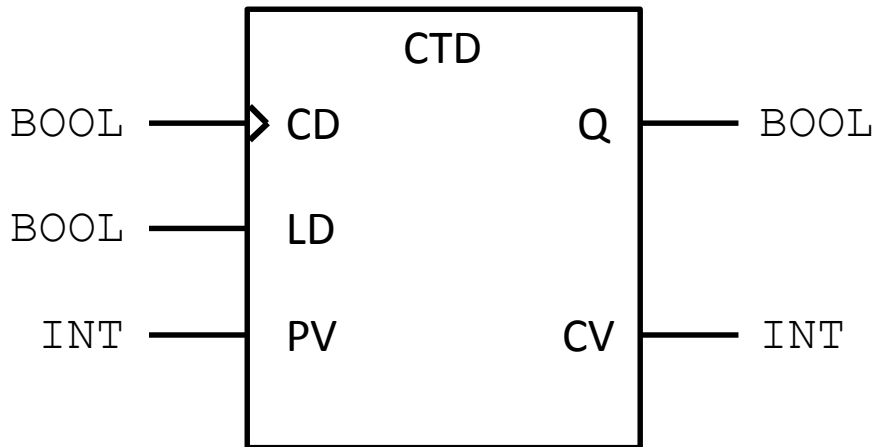
- CU: **Élérzékeny** számláló bemenet (*counting input*)
- R: Számláló nullázása (*reset*) – $CV := 0$
- PV: Célérték (*preset value*)
- Q: Státusz kimenet (*status output*): elérte-e a számláló a küszöbértéket? $Q = (CV \geq PV)$
- CV: Számláló regiszter értéke (*counter value*)

Felfelé számláló (CTU)



Lefelé számláló (CTD)

Szabványos megvalósítás



CD: **Élérzékeny** számláló bemenet (*counting input*)

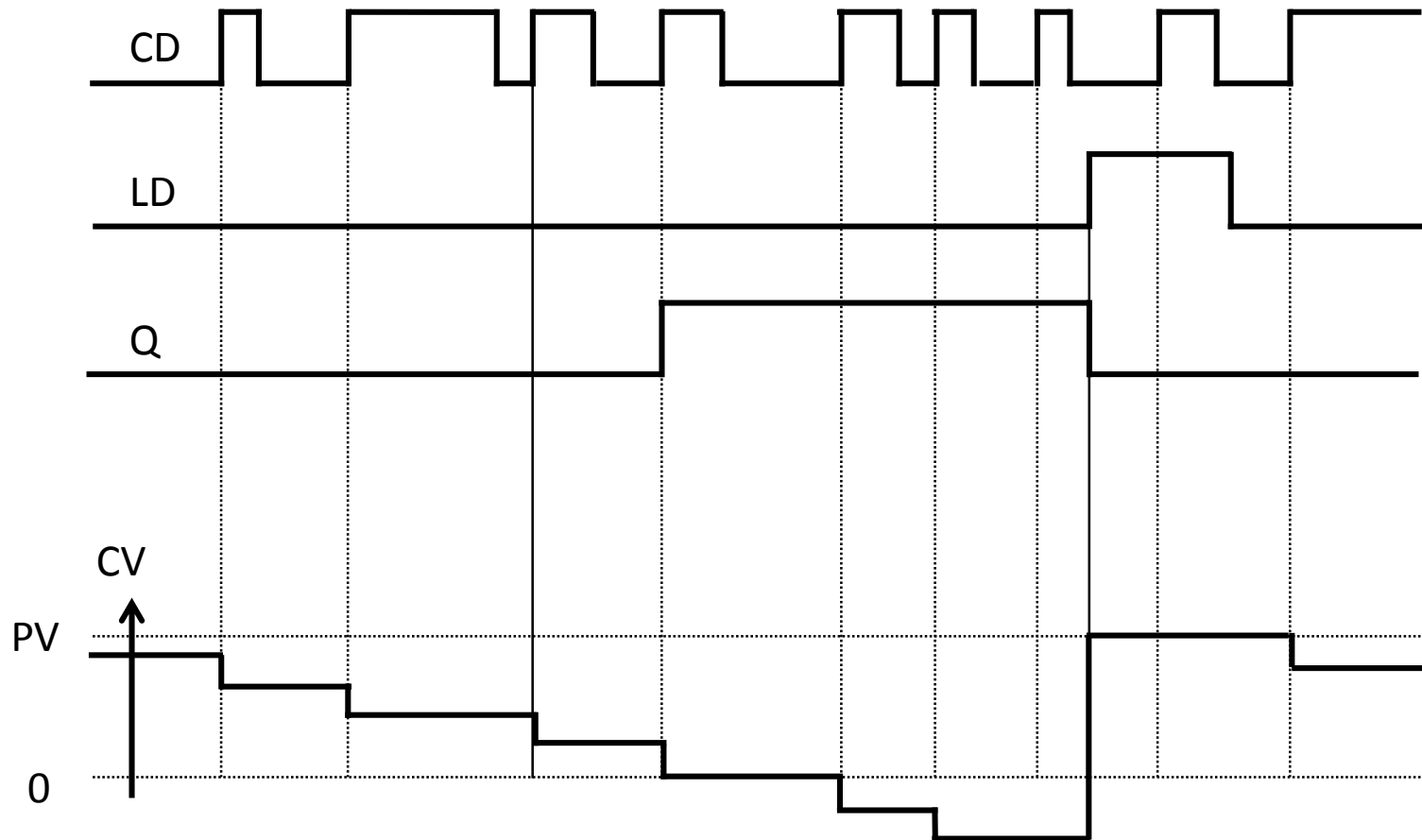
LD: Kezdőérték betöltése a számláló regiszterbe (*load*) - $CV := PV$

PV: Kezdőérték (*preset value*)

Q: Státusz kimenet (*status output*): elérte-e a számláló a nullát?
 $Q = (CV \leq 0)$

CV: Számláló érték (*counter value*)

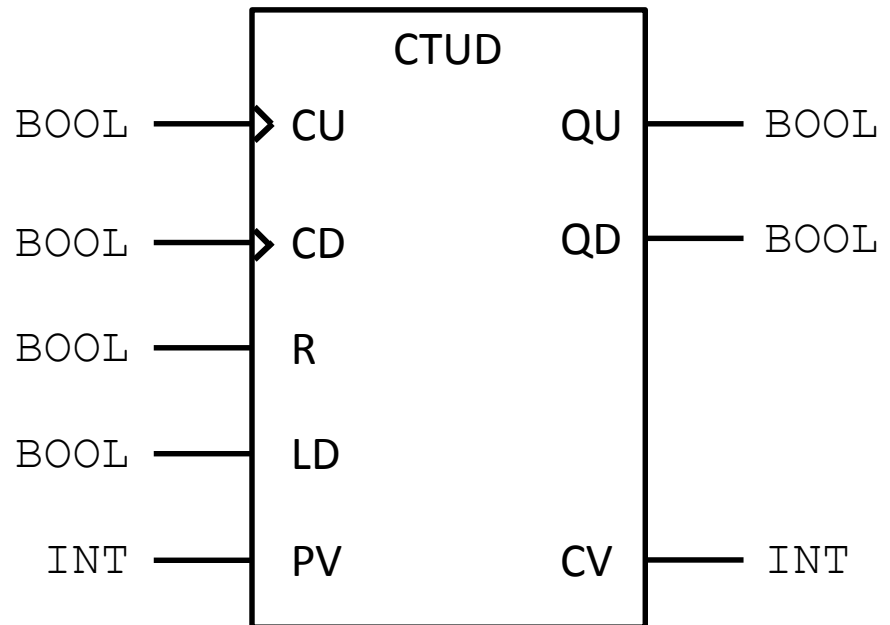
Lefelé számláló (CTD)



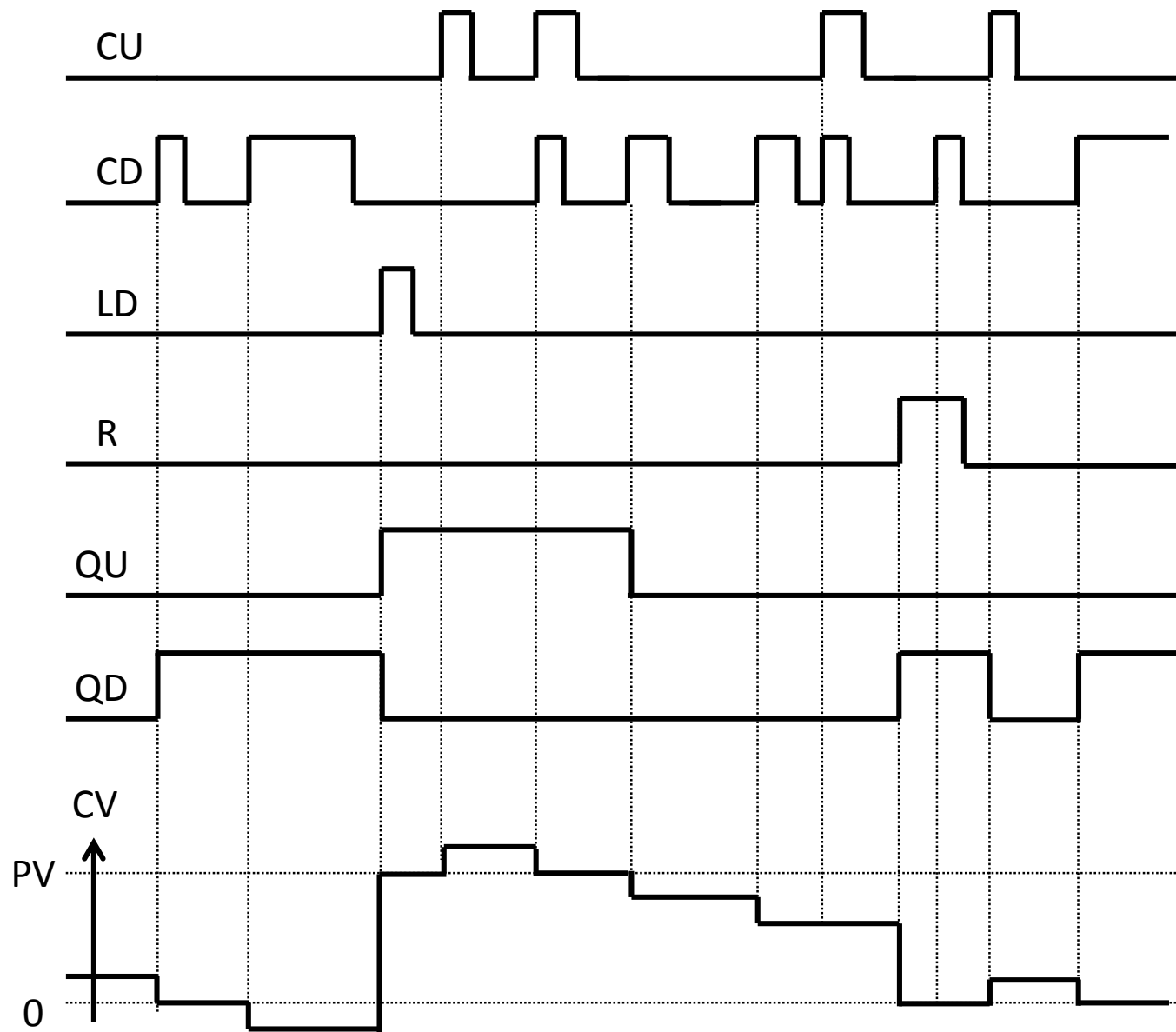
Fel- és lefelé számláló (CTUD)

Szabványos megvalósítás

- CU: **Élérzékeny** felfelé számláló bemenet (*count up input*)
- CD: **Élérzékeny** lefelé számláló bemenet (*count down input*)
- R: Számláló nullázása (*reset*) - $CV := 0$
- LD: Kezdő/célérték betöltése (*load*) - $CV := PV$
- PV: Kezdő/célérték (*preset value*)
- QU: Felfelé számláló státusz
 $QU = (CV \geq PV)$
- QD: Lefelé számláló státusz
 $QD = (CV \leq 0)$
- CV: Számlálóérték
(*counter value*)



Fel- és lefelé számláló (CTUD)



A bitműveleteken túl

Műveletek szavakon, hosszúszavakon...

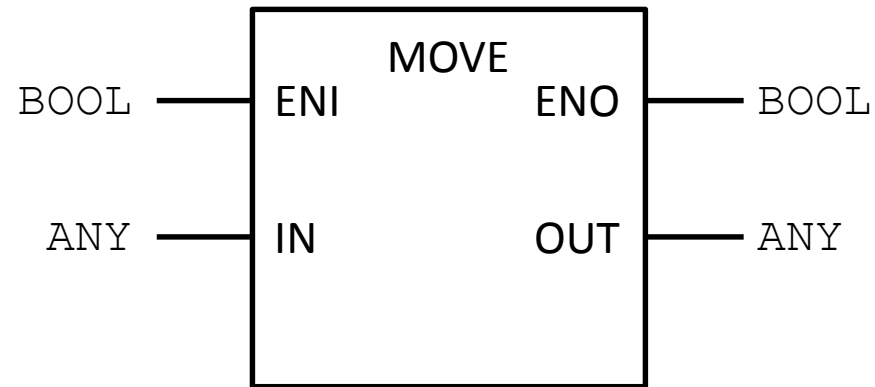
- Összehasonlítás
- Értékadás
- Aritmetikai és logikai műveletek

Engedélyező be- és kimenetek

- A létradiagram vezetékein csak logikai értékek jelenhetnek meg
- Hogyan integrálható egy nem logikai ki- és bemenetekkel rendelkező blokk (pl. összeadó)?
- ENI / ENO pár
 - Szabványos blokkoknál mindenképpen megvan
 - ENI: *Enable Input*
 - A művelet csak akkor hajtódik végre, ha ENI=1
 - ENO beállítása
 - Alapértelmezésben $ENO := ENI$
 - Hiba esetén $ENO := 0$
 - Tetszőleges beállítás a blokkon belül

Értékadás

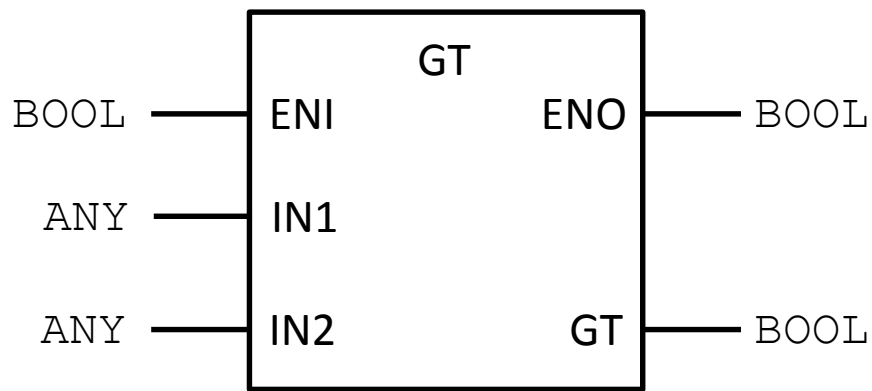
Szabványos megvalósítás



OUT := IN

Összehasonlítás

Szabványos megvalósítás

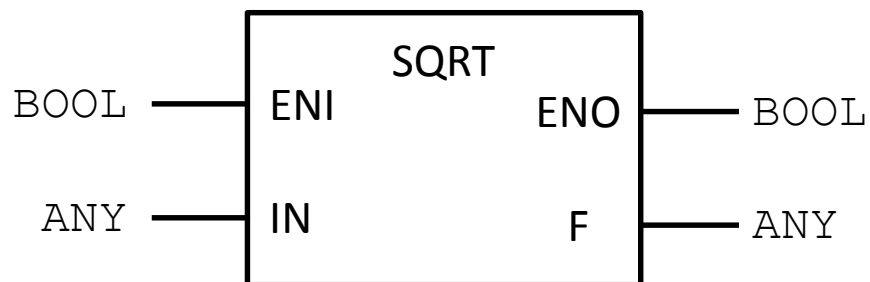


$GT = (IN1 > IN2)$

Mnemonic	Művelet
GT	>
GE	≥
LT	<
LE	≤
EQ	=
NEQ	≠

Aritmetikai műveletek

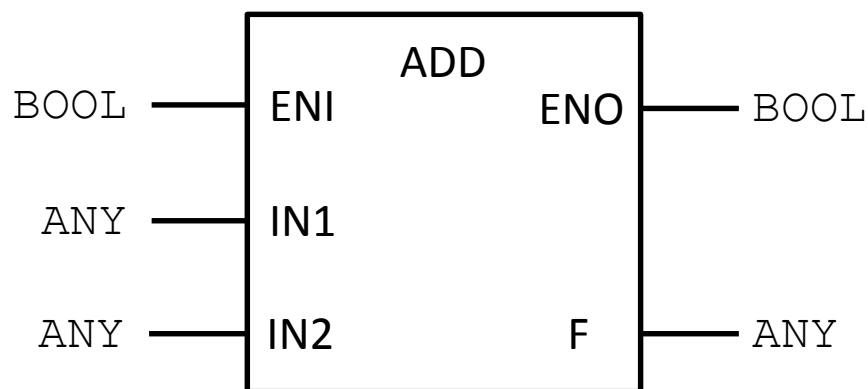
Szabványos megvalósítás



Mnemonic	Művelet
ABS	$F = IN $
SQRT	$F = \sqrt{IN}$
LN	$F = \ln IN$
EXP	$F = e^{IN}$
LOG	$F = \log_{10} IN$
SIN	$F = \sin IN$
COS	$F = \cos IN$
TAN	$F = \tan IN$
ASIN	$F = \sin^{-1} IN$
ACOS	$F = \cos^{-1} IN$
ATAN	$F = \tan^{-1} IN$

Többszörös aritmetikai műveletek

Szabványos megvalósítás



Mnemonic	Művelet
ADD	+
MUL	×
SUB	–
DIV	/
MOD	IN1 mod IN2
EXPT	$IN1^{IN2}$

További műveletvégző blokkok

- Konverziós műveletek
 - BCD – bináris
 - Word – Double Word
 - ...
- Szó eltolás és logikai műveletek
- Regiszterek (sorok)
 - LIFO
 - FIFO

Programszervezési utasítások

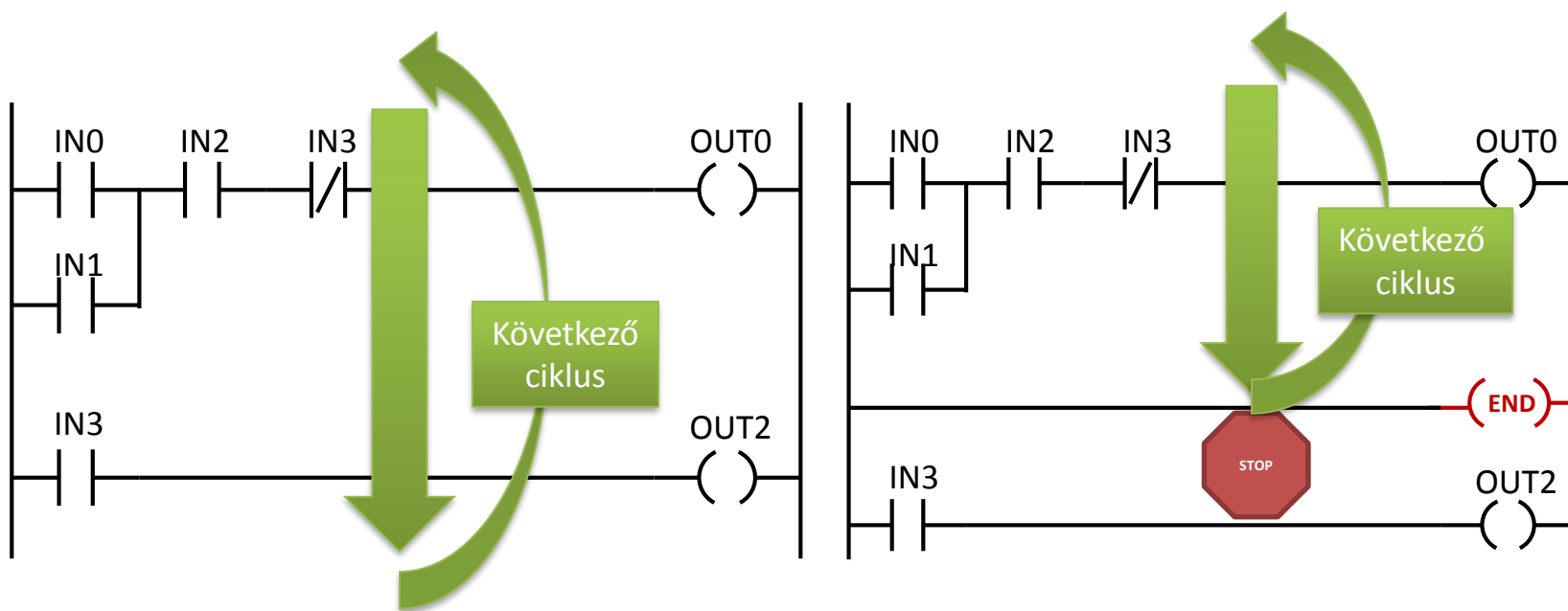
- Megszakított vagy nemlineáris programvégrehajtást tesz lehetővé
- A PLC-ciklusnak a programvégrehajtási fázisára hat
 - Nincs hatása a bemenetek olvasására és a kimenetek beállítására
 - A be- és kimeneti kép a szokásos módon kerül kezelésre



- **A programszervezési utasítások jelentősen ronthatják a ciklusidőt**
- **Hiba esetén túlléphető a maximális ciklusidő!**

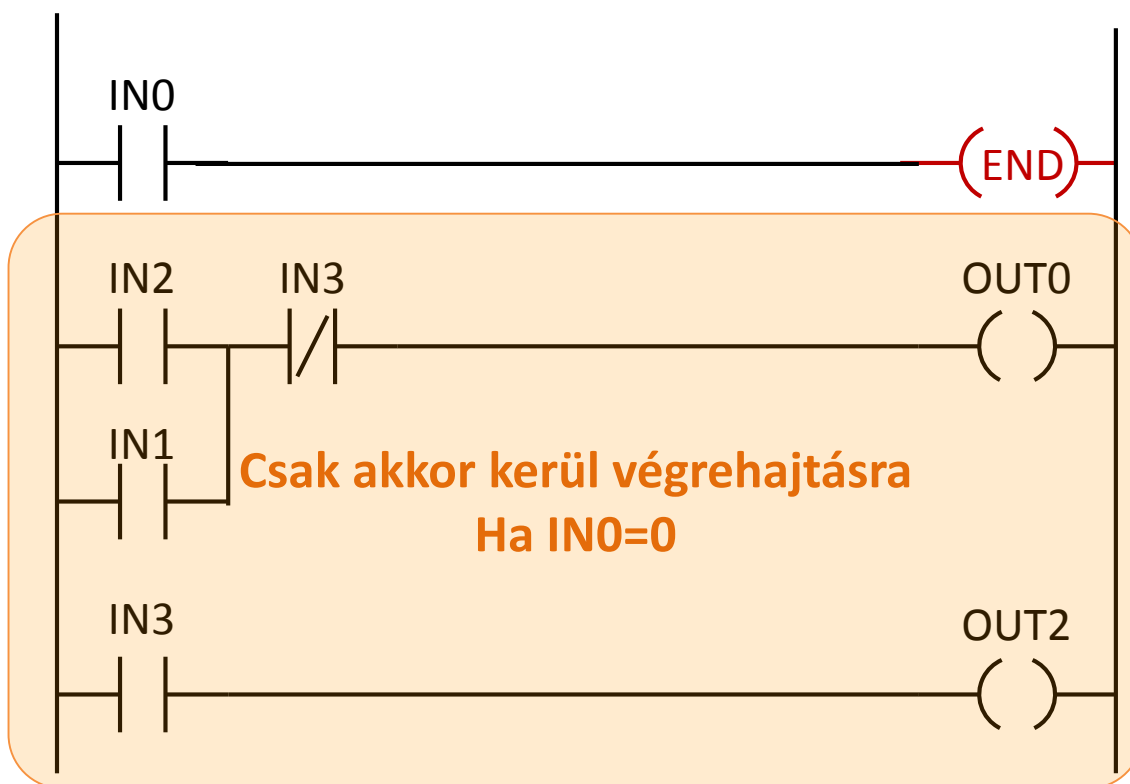
Programvégrehajtás leállítása

- Egy ciklusban a program végrehajtása megáll
 - az utolsó létrásor után
 - egy END tekercs hatására



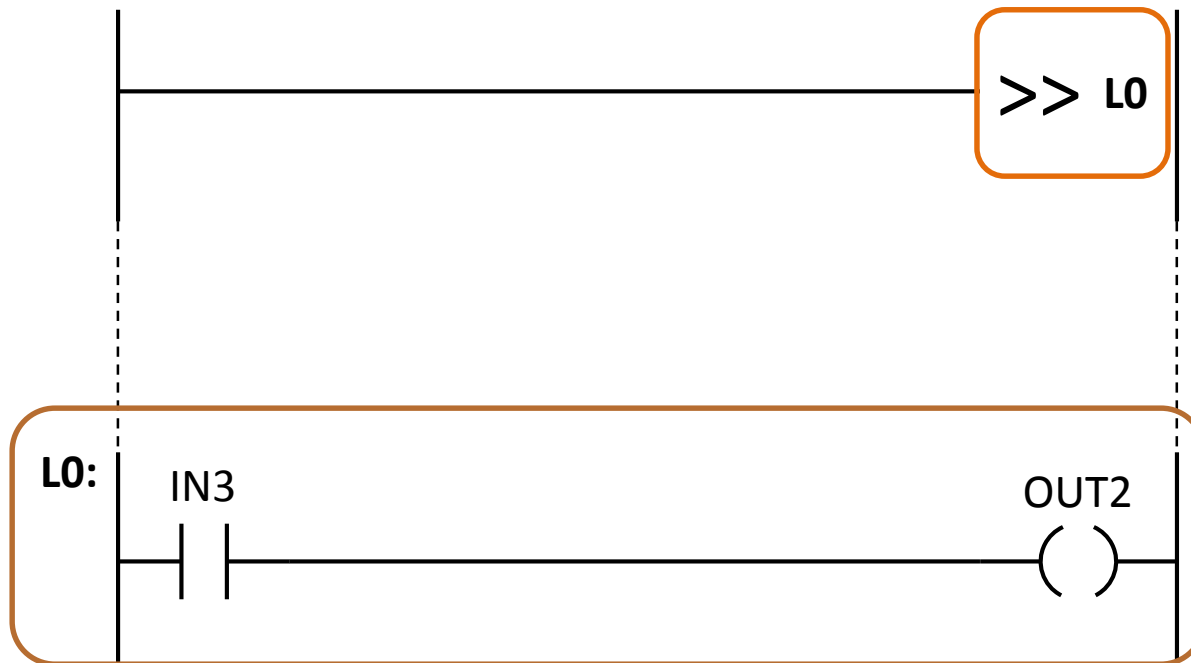
Feltételes leállítás

- A programvégrehajtás leállítása, ha egy feltétel teljesül
 - Csökkenti a ciklusidőt
 - Hibakeresésnél hasznos

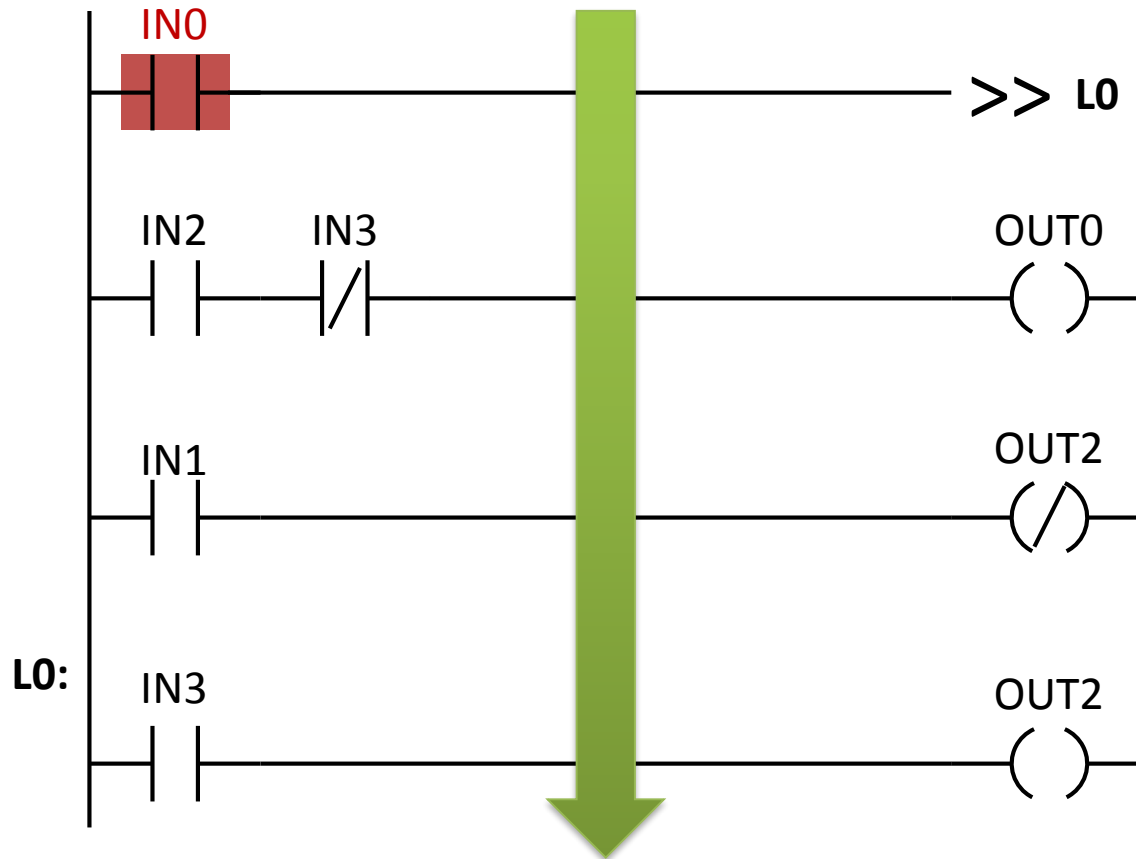


Ugró utasítások

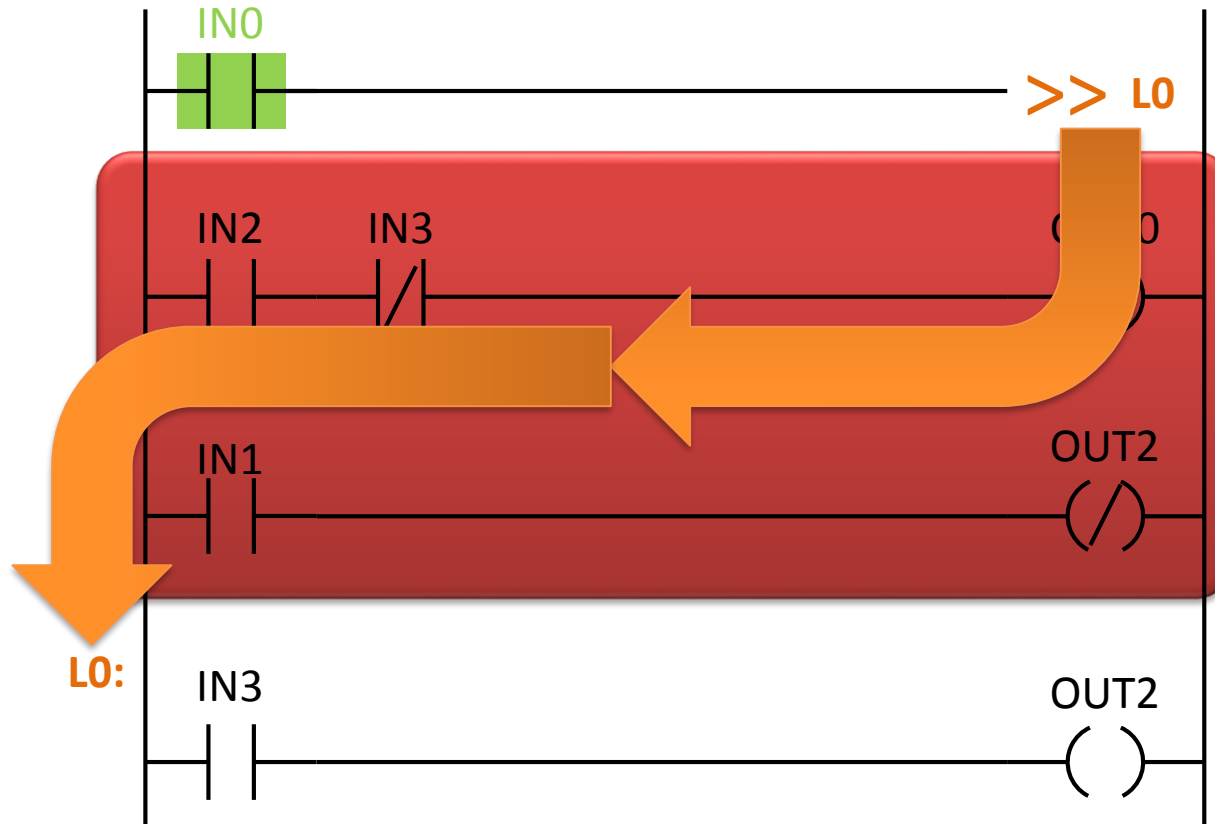
- A létrafokokhoz címkék (*label*) kapcsolhatók
- Ugró utasítás hatására a programvégrehajtás a megfelelő címkéjű létrasortól folytatódik



Ugró utasítások

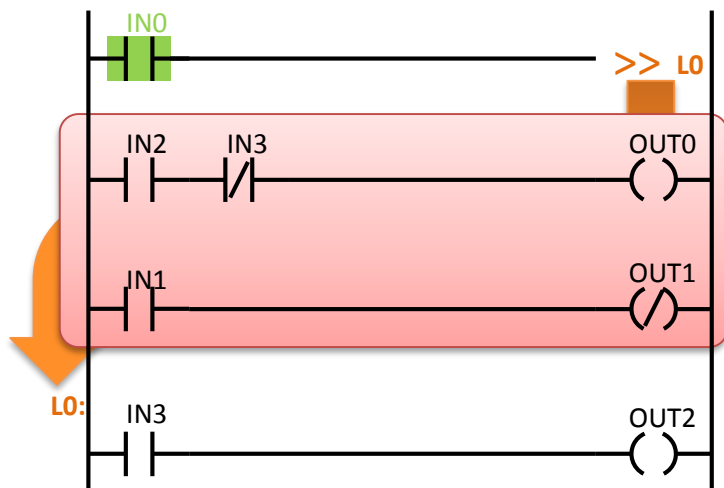


Ugró utasítások



Ugró utasítások hatása

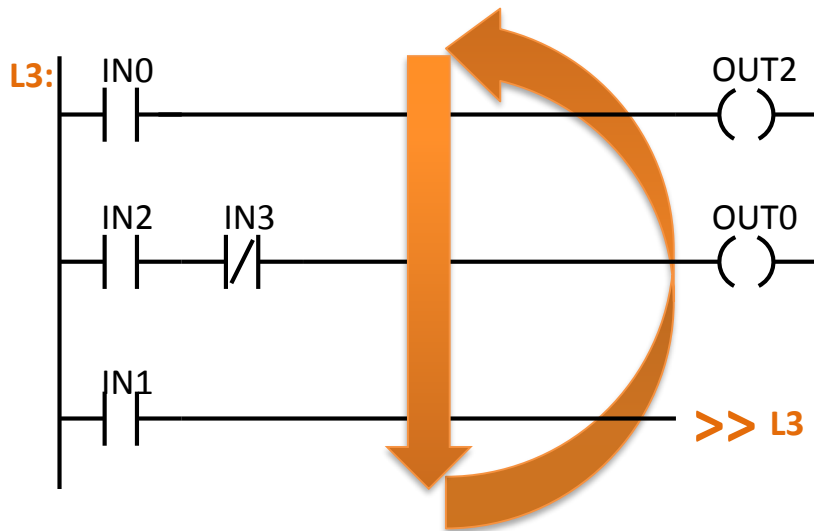
IN0	1	OUT0	1
IN1	1	OUT1	0
IN2	0	OUT2	1
IN3	0		



IN0	1	OUT0	1
IN1	1	OUT1	0
IN2	0	OUT2	0
IN3	0		

- Az ugró utasítás és a célcímke közötti létraszorok „kimaradnak”
- A kimaradó sorok logikai függvényei nem értékelődnek ki
- A kapcsolódó kimenetek nem kerülnek beállításra
- A tekercsek változói megőrzik előző értéküket

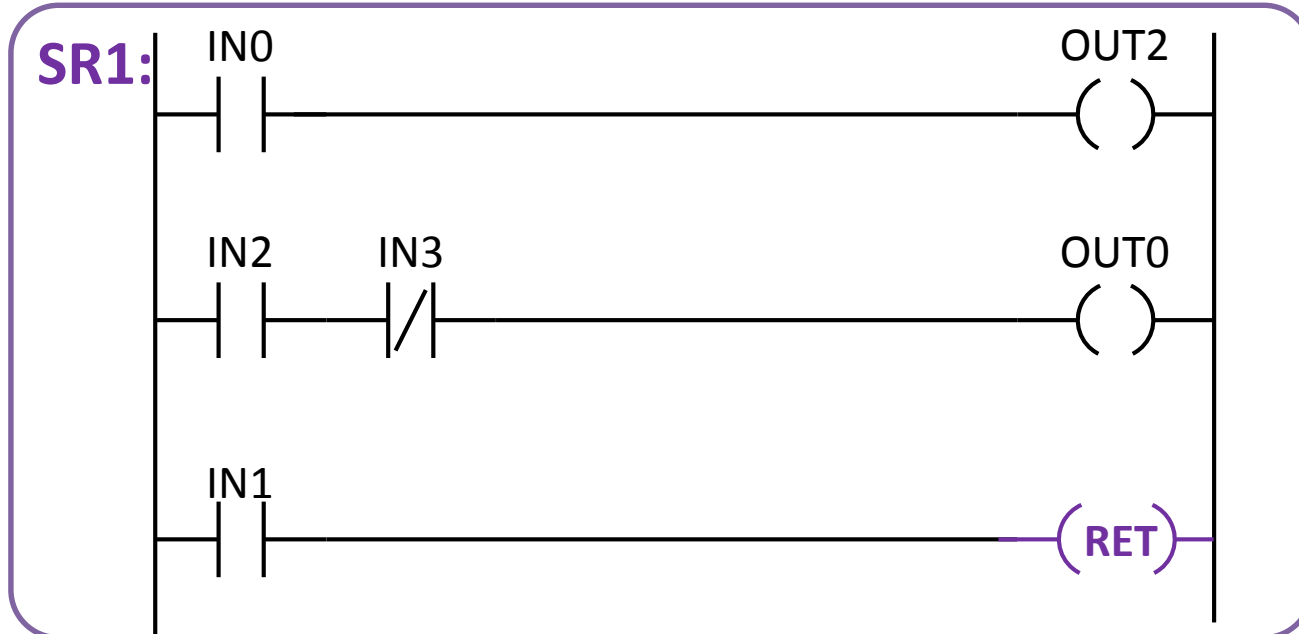
Visszaugrás



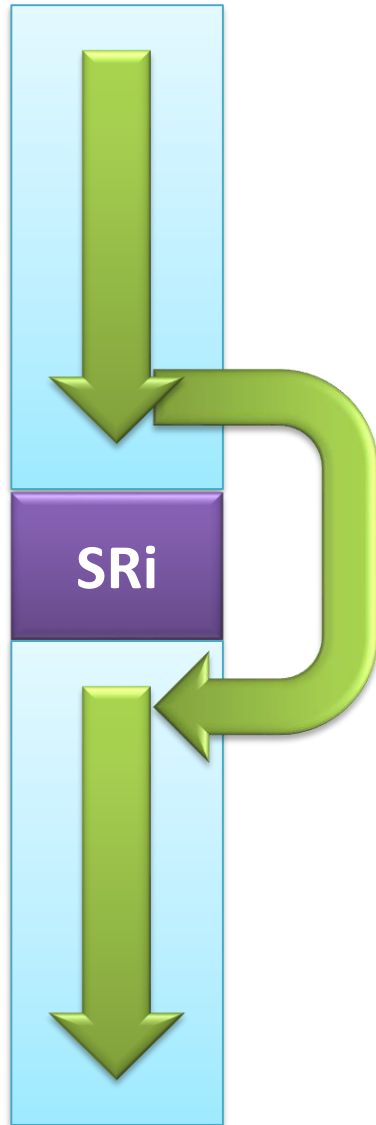
- Végtelen ciklus veszélye
- Kerülendő módszer
- Ha alkalmazzuk, legyünk nagyon óvatosak!

Szubrutinok

- Szubrutin: egymást követő létrások halmaza
- Kezdet: szubrutin címke
- Vége: RET (*return*) utasítás

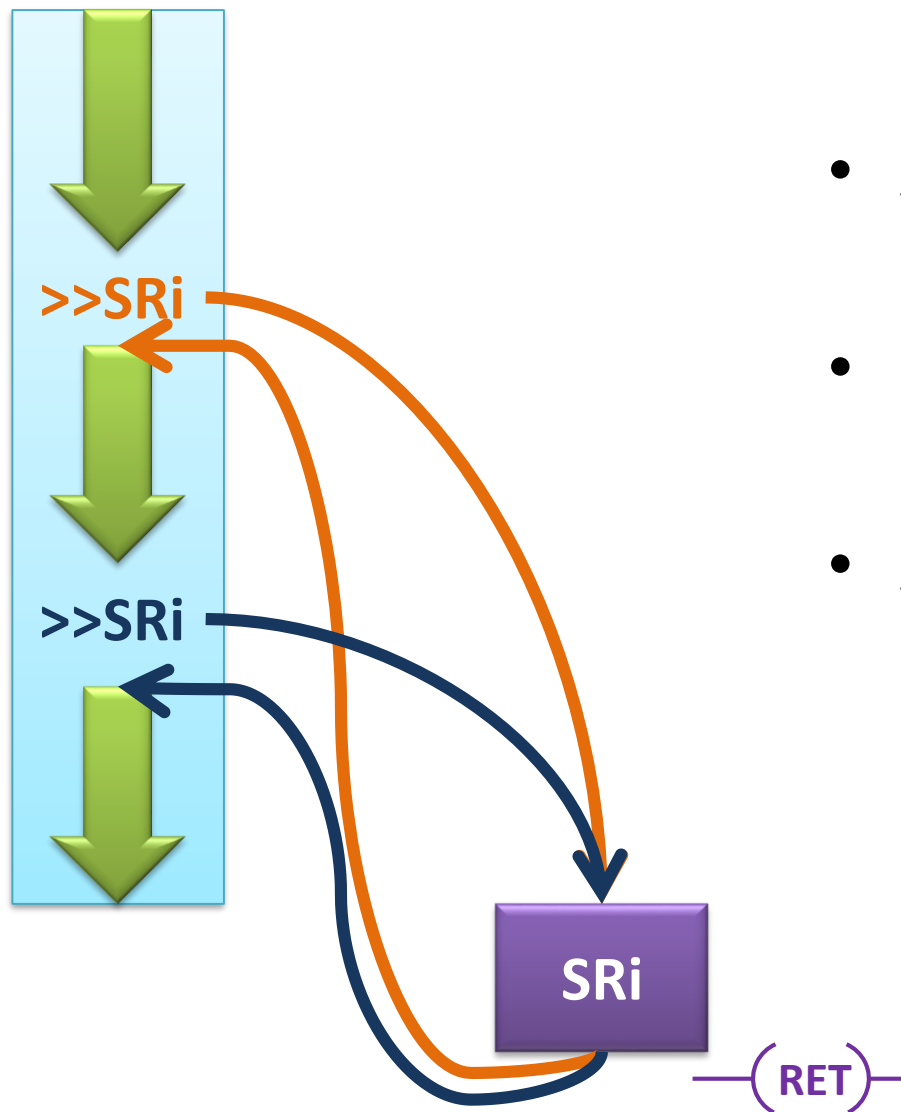


Szubrutinok



- Hívás hiányában a szubrutinhoz tartozó létrasorok nem kerülnek kiértékelésre
- *Best practice: a szubrutinokat a kód végére célszerű elhelyezni. Ugyan a működést ez nem befolyásolja, de a kód átláthatóbb lesz*

Szubrutinok



- A szubrutinok hívása az ugró utasításhoz hasonló
- Egy szubrutin több létrasorból is hívható
- A szubrutin lefutása után a végrehajtás a következő létrasortól folytatódik

Paraméterátadás

- A szubrutin nem függvény → nincs formális paraméterátadás
- Megoldás: memóriabitek vagy szavak használata, pl.
 - A paramétereket az %MW1 és %M12 regiszterekbe helyezzük a hívás előtt
 - A szubrutin az eredményt az %MW8 regiszterbe tölti