SYGNALIZACJA ŚWIETLNA NA SKRZYŻOWANIACH

Modelowanie przebiegu ruchu na skrzyżowaniach z sygnalizacją zmiennoczasową

VISSIM: *Symulacja skrzyżowań z sygnalizacją świetlną*

Zachowania kierowców na skrzyżowaniu:

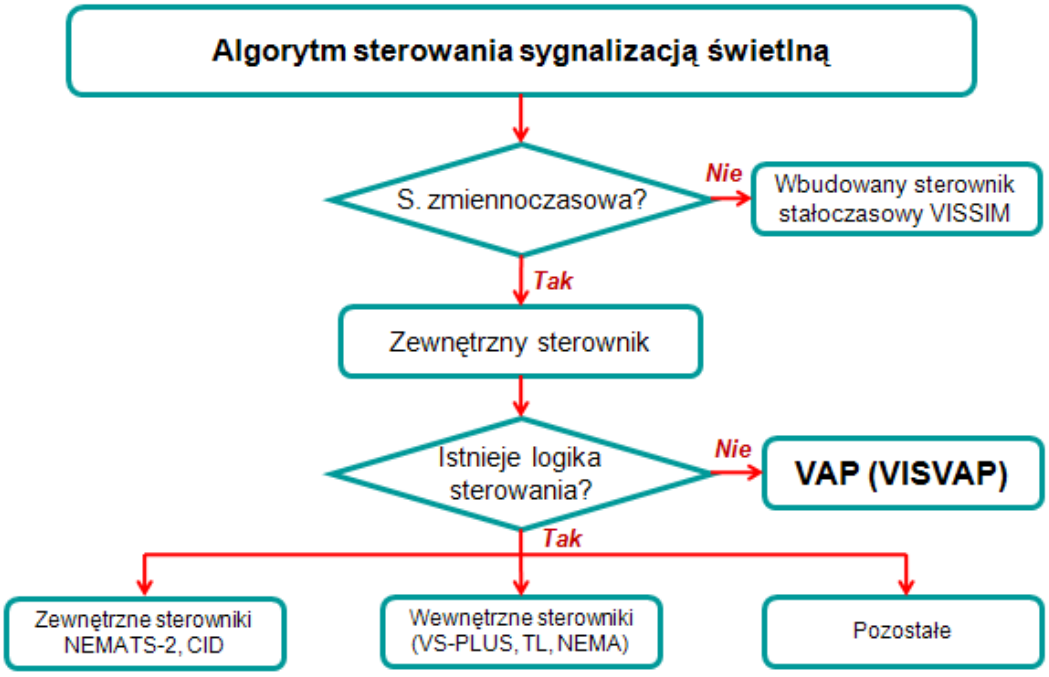
- pojazdy zatrzymują się 0,5 m przed sygnalizatorem (linią zatrzymania)
- sygnalizator musi wyświetlać sygnał czerwony
- przejazd podczas trwania sygnału żółtego – wyłącznie w przypadku braku możliwości zatrzymania
- modelowanie probabilistyczne przejazdu na żółtym świetle – moduł zachowania kierowców

Modelowanie sygnalizacji świetlnej:

- definiowanie grup sygnalizacyjnych (max 125 grup dla 1 sterownika)
- lokalizacja sygnalizatorów (sprzężonych z linią zatrzymania) wraz z przypisaniem do grup sygnalizacyjnych
- przypisanie sygnalizatorom odpowiednich typów pojazdów (np. sygnał dla tramwaju/autobusu)
- relacje kolizyjne muszą uwzględniać zasady pierwszeństwa

Detektory:

- element sieci umieszczany na pasie ruchu bez uwzględniania rzeczywistej formy detekcji (pętla indukcyjna, wideodetektor, przycisk zgłoszeniowy, blokada zwrotnicy, podczerwień, radio itp.), ale ze zdefiniowaną długością
- określa się typ pojazdów, które będą wykrywane przez detektor
- przypisuje się detektor do sygnalizacji (ale nie do grupy sygnałowej) i definiuje rodzaj detektora:
  - standardowy (uniwersalny)
  - obecności
  - zgłoszeń (impuls)
  - zgłoszeń KZ (meldunki)
- kilka detektorów może mieć jeden numer – sterownik potraktuje je jako jeden detektor
- możliwe stosowanie „wygładzania wykładniczego" – wyrównywania zajętości detektora:
  - obliczanie stopnia zajętości detektora w ciągu ostatnich t sekund
  - płynna wartość średnia z pomierzonych ostatnich kroków



VAP: *Język programowania*

Logika sterowania:

1. Tworzenie własnej logiki sterowania w oparciu o język programowania – logika opisana w zbiorze tekstowym
2. Podczas symulacji VAP przetwarza program i tworzy instrukcje sterowania sygnalizacją świetlną – odpowiedź „sterownika" na stan wzbudzenia detektorów podczas symulacji
3. Prosty język programowania:
   - w obliczeniach operatorów arytmetycznych i logicznych stosuje się reguły kolejności działań
   - klasyczne funkcje warunkowe i język logiki zdań (if - *jeżeli*, Else – *jeżeli nie*, i, lub)
   - etykiety skoku do danej linii algorytmu(GOTO, WAIT_AT)
   - dwa rodzaje stałych:
     a. systemowe (odwołują się do ust. parametrów, np. grup sygn.)
     b. definiowane (własne)

Przykład:

```
S00Z001:  IF T_free( 3 ) = 1 THEN
S01Z002:     cycSecond := 1
          ELSE
S00Z002:     cycSecond := cycSecond + 1
          END;
S00Z004:  SetT( cycSecond );
S00Z005:     IF Stage_active( 1 ) THEN
S01Z005:        IF Gmin_1 OR Gmin_2 THEN
                        GOTO PROG_ENDE
                  ELSE
S01Z006:             IF Call_BUS THEN
S03Z006:                Is( 1 , 2 );
                        GOTO PROG_ENDE
```

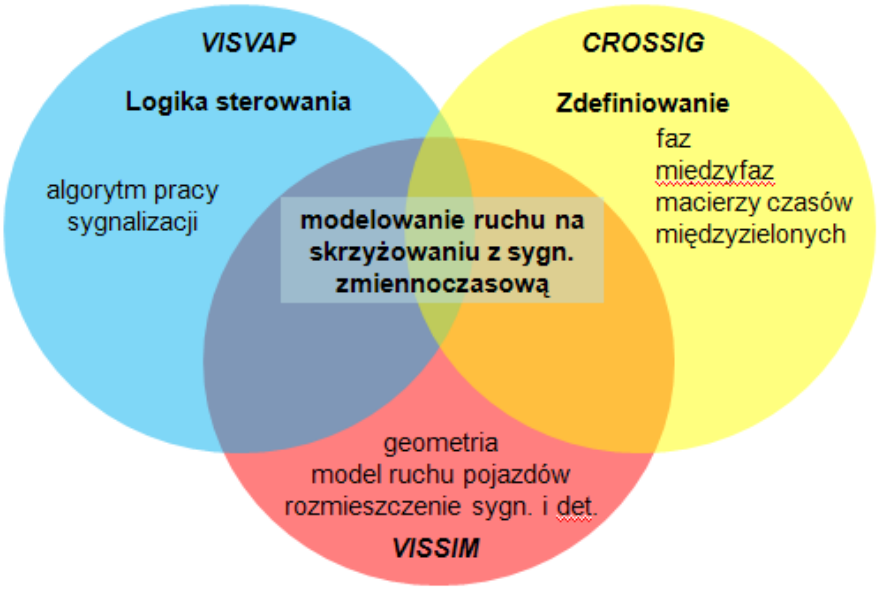
S00Z001	- etykiety miejsca
IF	- funkcje logiczne
Is(1 , 2)	- wyrażenia systemowe
Gmin_1	- wyrażenia definiowane

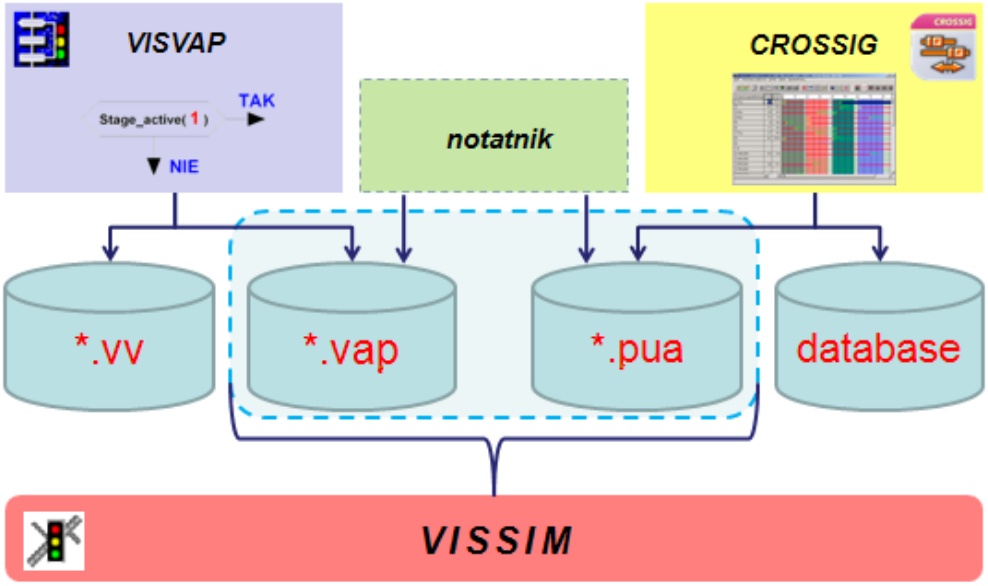
Wyrażenia w nawiasach są numerami faz, grup sygnałowych czy detektorów zgodnie z numeracją w programie Vissim, lub też innymi elementami zależnie od wykorzystanej funkcji.

Moduł VisVAP: *Sterowanie zmiennoczasowe*

Logika sterowania:

1. Wspomaga tworzenie logiki sterowania wykorzystującej język VAP
2. Tworzenie i edytowanie programu logiki w postaci schematu blokowego
3. Łatwe zastosowanie – nie wymaga znajomości języka programowania
4. Współpracuje z innymi programami wspierającymi pracę projektanta:



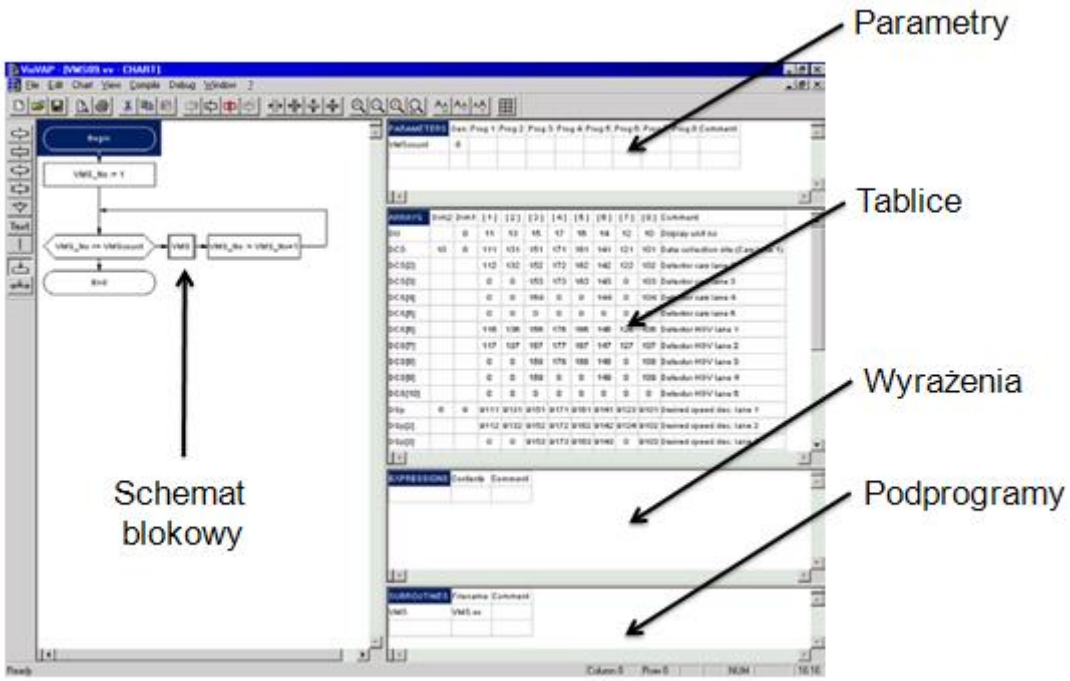


Do pełnego za modelowania pracy sygnalizacji zmiennoczasowej potrzebne są:

a) plik *.vap zawierający algorytm sterowania, zmienne i zdefiniowane parametry (VAP lub VisVAP)
b) plik *.pua zawierający macierz czasów międzyzielonych, zdefiniowanie faz i przejść międzyfazowych (np. Crossig)
c) plik vap.dll pozwalający na interpretację sporządzonej logiki sterowania

Możliwe jest wykorzystanie edytorów tekstu do tworzenia plików .vap i .pua.

Okno programu:



Parametry: definiowane parametry z przypisaniem wartości (mogą być różnicowane dla każdego programu sygnalizacji oddzielnie) wykorzystywane w logice sterowania, np. wartości min i max sygnału zielonego, luki czasowej, itp.

Wyrażenia: własne wyrażenia tworzone poprzez łączenie wyrażeń systemowych językiem logiki zdań (I/LUB, NIE)

Podprogramy: wyrażenia będące programami logiki zapisanymi w innym pliku vap

Symbole blokowe:

Symbole blokowe połączone za pomocą linii łączących 

Symbole blokowe dzielą się na pięć typów:

 końcowe – początek i koniec bloku (zawsze)

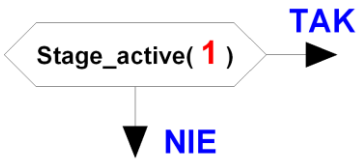
 działania

 warunkowe

 wywołania podprogramu

 podziału strony

Z bloku warunku wyjście w prawo następuje po spełnieniu warunku, w dół w przeciwnym wypadku:



W przypisywaniu wartości w bloku założeń korzystamy z ” := ” („takie że”)

Operatory łączące wyrażenia:

Logiczne „NIE”	NOT , NICHT, ~ , \ , !
Logiczne „i”	AND , UND , && , & , ^
Logiczne „lub”	OR , ODER , v , V

Działania	+ - * /
Porównań	= <> < > <= =>
Modulo	% , \

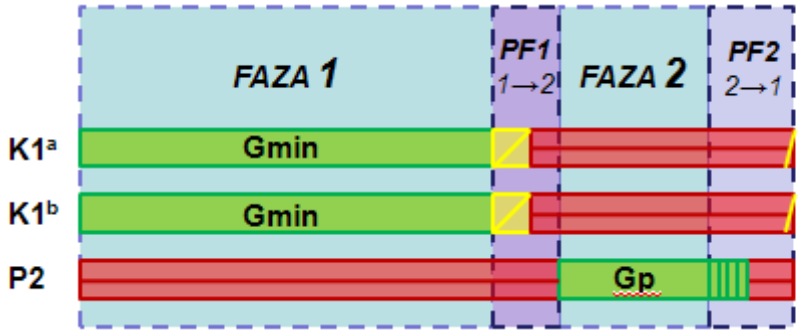
ĆWICZENIE 1: *Sygnalizacja wzbudzana na przejściu dla pieszych*

Przeprowadzić symulację przejścia dla pieszych z wzbudzaną sygnalizacją świetlną na drodze 1x2.

Sygnalizacja wzbudzana dla pieszych:

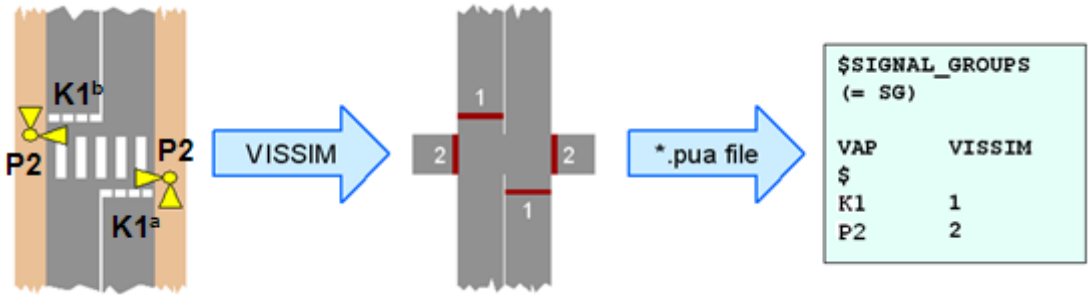
1. Informacje o zgłoszeniach zbierane wyłącznie z detektorów dla pieszych.
2. W przypadku braku zgłoszeń sygnał zielony dla pojazdów może być wydłużany bez ograniczeń.
3. W przypadku ciągłych wzbudzeń program sygnalizacji musi zapewniać przepustowość dla pojazdów – dobranie Gmin o wystarczającej długości.

Program sygnalizacji oparty o fazy i przejścia międzyfazowe:



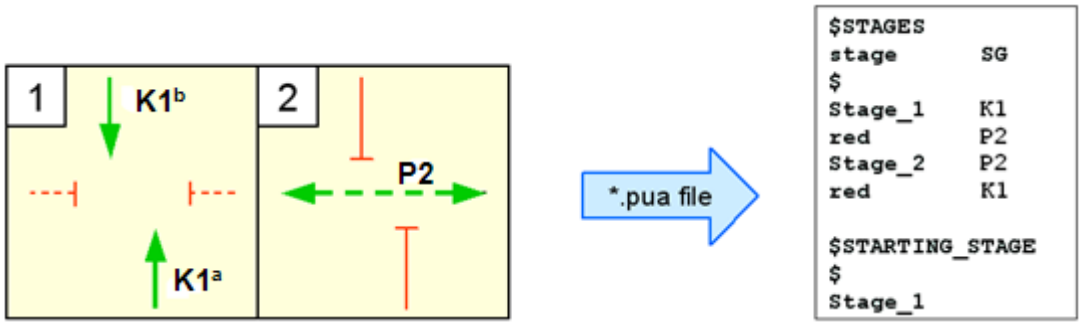
GRUPY KOŃCZĄCE \ GRUPY ROZPOCZYNAJĄCE	E/D	K1a	K1b	P2
K1a				5,0
K1b				5,0
P2	4,0	4,0		

Powiązanie grup sygnałowych w programie Crossig, VisVap i Vissim w pliku pua:



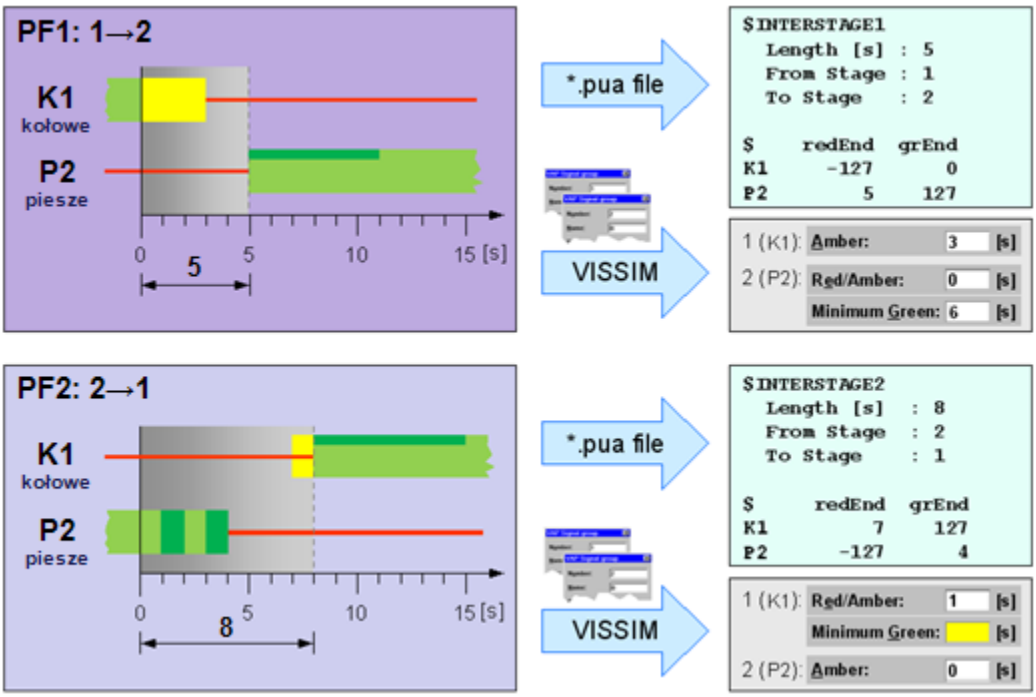
(ponieważ czasy międzyzielone w grupie K1a i K1b są identyczne, scalono je do jednej grupy sygnalizacyjnej K1)

Powiązanie układu faz w programie Crossig, VisVap i Vissim w pliku pua:



Grupa sygnałowa	Signal group (SG)	Signalgruppen (Sg)
Faza ruchu	Stage	Phase
Przejście międzyfazowe	Interstage (Is)	Phasenübergang (PÜ)
Długość przejścia międzyfazowego	Interstage duration	Übergangsdauer

Zdefiniowanie przejść międzyfazowych w pliku pua:

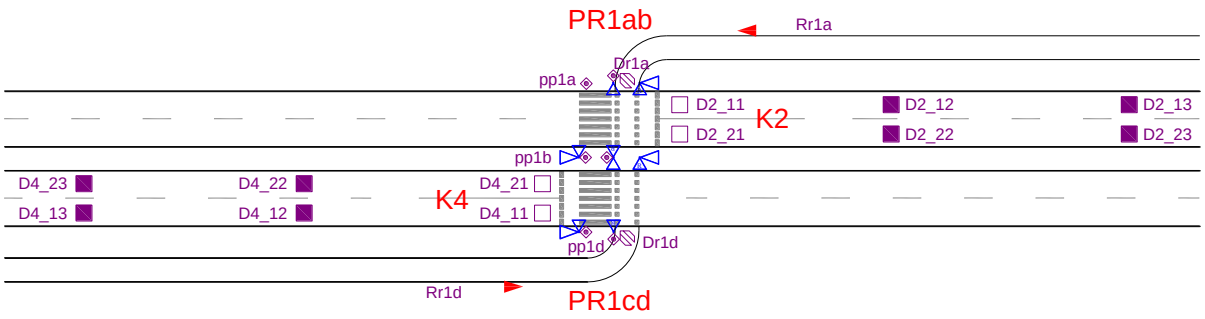


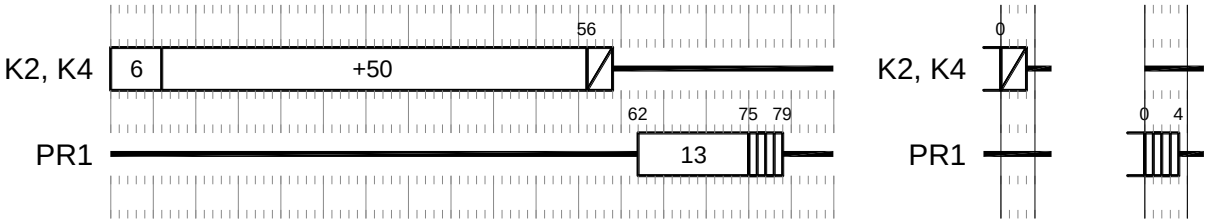
Schemat blokowy algorytmu (do samodzielnego wykonania):

Schemat blokowy:	*Parametry:*	
	.....	.....
	.....	.....
	.....	.....
	Wyrażenia:	
	.....	.....
	.....	.....
	.....	.....

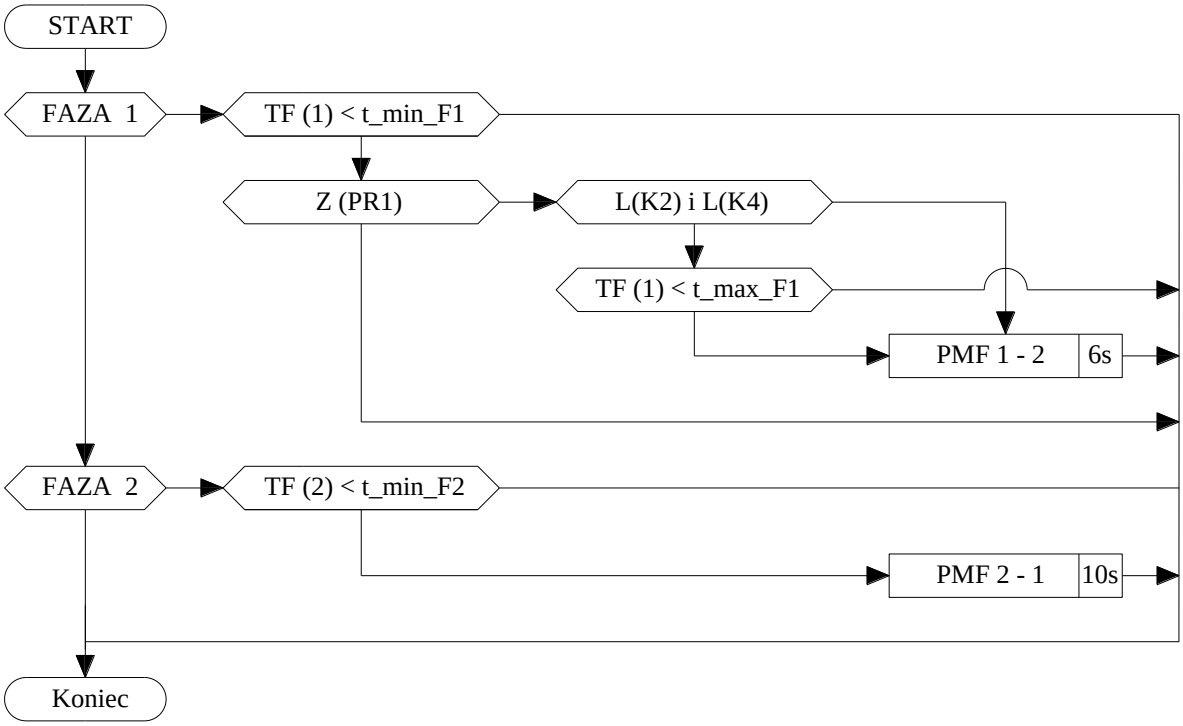
ĆWICZENIE 2: *Sygnalizacja wzbudzana na przejściu dla pieszych*

Przeprowadzić symulację przejścia dla pieszych z wzbudzaną sygnalizacją świetlną na drodze G2x2 o ustalonym programie sygnalizacji świetlnej i algorytmie sterowania ruchem.





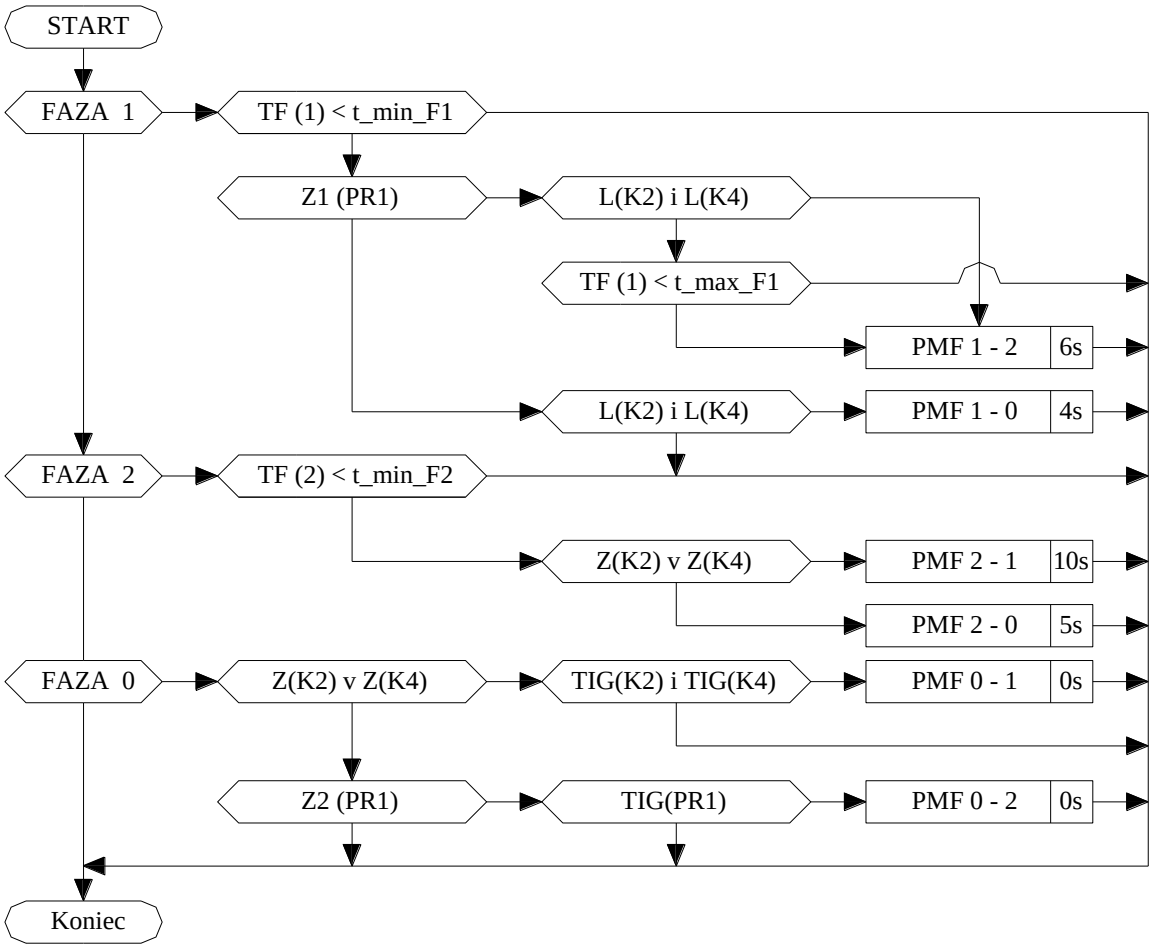
Algorytm 1 – faza preferowana 1 (ruch kołowy)



Minimum trwania fazy		Maksimum trwania fazy	
t_min_F1	16s	t_max_F1	56s
t_min_F2	13s	-	-

Warunek przerwania	
L (K2)	L (D2_11) ≥ 3.5s ∧ L(D2_21) ≥ 3.5s
L (K4)	L (D4_11) ≥ 3.5s ∧ L(D4_21) ≥ 3.5s
Warunek zgłoszenia	
Z (PR1)	Z (pp1a) ∨ Z(pp1b) ∨ Z(pp1c) ∨ Z (pp1d) ∨ Z(Dr1a) ∨ Z(Dr1d)

Algorytm 2 – faza all red



Warunek zgłoszenia	
Z1 (PR1)	Z (pp1a) ∨ Z(pp1b) ∨ Z(pp1c) ∨ Z (pp1d) ∨ Z(Dr1a) ∨ Z(Dr1d) v Z(Rr1a) v Z(Rr1d)
Z2 (PR1)	Z (pp1a) ∨ Z(pp1b) ∨ Z(pp1c) ∨ Z (pp1d) ∨ Z(Dr1a) ∨ Z(Dr1d)
Z (K2)	Z(D2_11) v Z(D2_12) v Z(D2_13) v Z(D2_21) v Z(D2_22) v Z(D2_23)
Z (K4)	Z(D4_11) v Z(D4_12) v Z(D4_13) v Z(D4_21) v Z(D4_22) v Z(D4_23)

	Warunek przerwania
L (K2)	dla TF(1) < 16s (czas fazy dłuższy bądź równy 16s) L (D2_11) ≥ 4.0s ∧ L(D2_12) ≥ 3.5s ∧ L(D2_13) ≥ 3.0s L (D2_21) ≥ 4.0s ∧ L(D2_22) ≥ 3.5s ∧ L(D2_23) ≥ 3.0s dla TF(1) ≥ 16s (czas fazy dłuższy bądź równy 16s) L(D2_12) ≥ 3.5s ∧ L(D2_13) ≥ 3.0s L(D2_22) ≥ 3.5s ∧ L(D2_23) ≥ 3.0s
L (K4)	dla TF(1) < 16s (czas fazy krótszy od 16s) L (D4_11) ≥ 4.0s ∧ L(D4_12) ≥ 3.5s ∧ L(D4_13) ≥ 3.0s L (D4_21) ≥ 4.0s ∧ L(D4_22) ≥ 3.5s ∧ L(D4_23) ≥ 3.0s dla TF(1) ≥ 16s (czas fazy dłuższy bądź równy 16s) L(D4_12) ≥ 3.5s ∧ L(D4_13) ≥ 3.0s L(D4_22) ≥ 3.5s ∧ L(D4_23) ≥ 3.0s

ĆWICZENIE 3: *Skrzyżowanie ze zmiennoczasową sygnalizację świetlną*

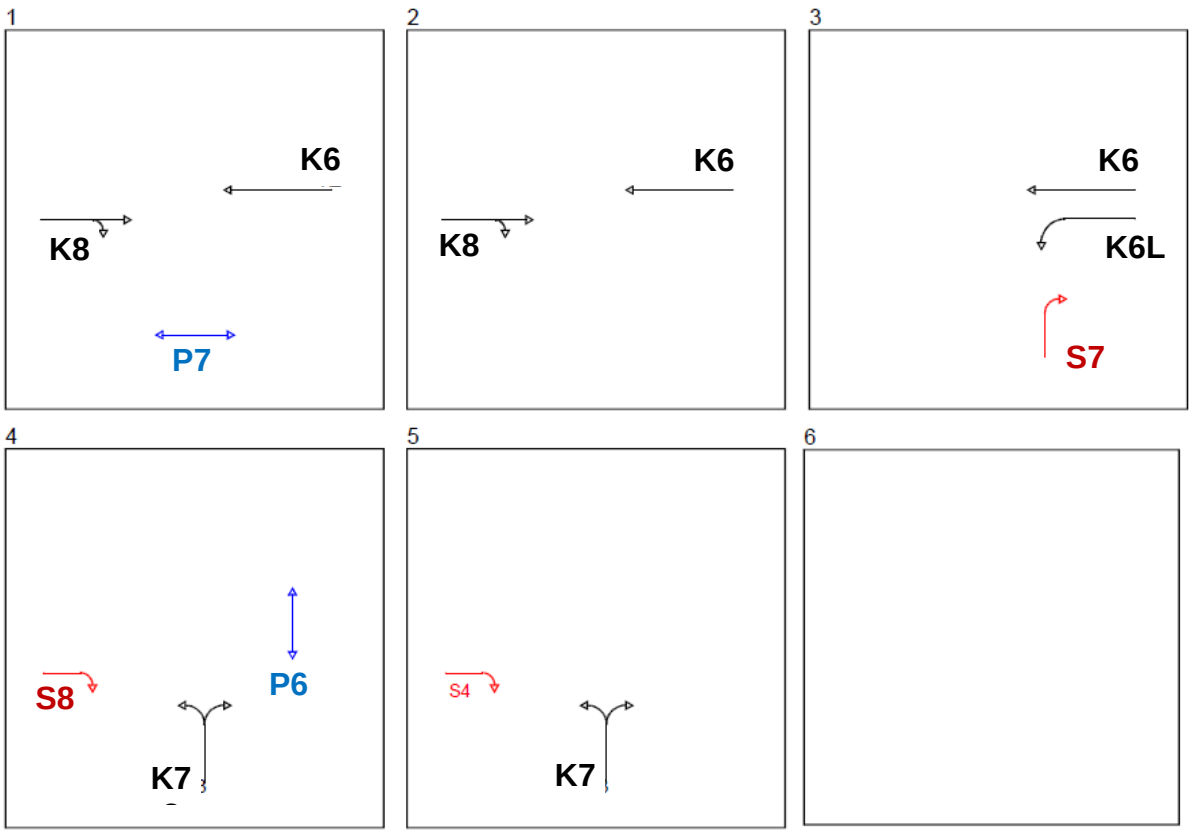
Dla skrzyżowania trzywlotowego i danych:

- geometrii i organizacji ruchu,
- natężeń ruchu,
- bazowego programu sygnalizacji świetlnej,
- układu detekcji

opracować w programie VisVAP algorytm sterowania ruchem w postaci schematu blokowego a następnie przeprowadzić symulację ruchu w programie PTV Vissim.

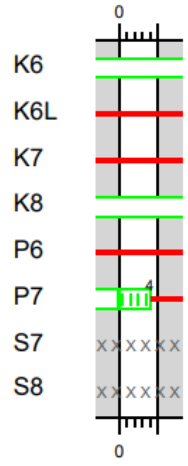
Układ faz ruchu:

Faza 1: kierunek K6 i K8, przejście P7
Faza 2: kierunek K6 i K8,
Faza 3: kierunek K6L, K6, zielona strzałka S7,
Faza 4: kierunek K7 i przejście P6 oraz zielona strzałka S8 (w cieniu K7),
Faza 5: kierunek K7 i strzałka S8 (w cieniu)
Faza 6: faza wszędzie czerwone

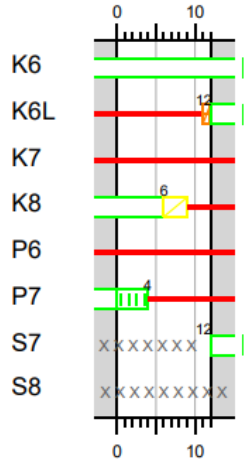


Przejścia międzyfazowe:

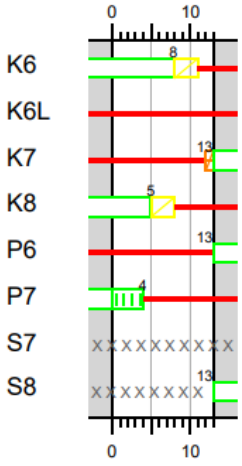
1-->2



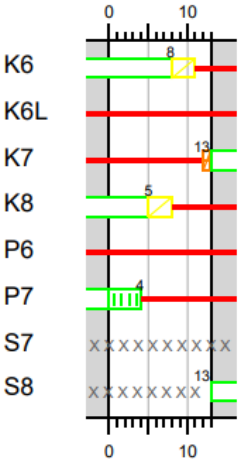
1-->3



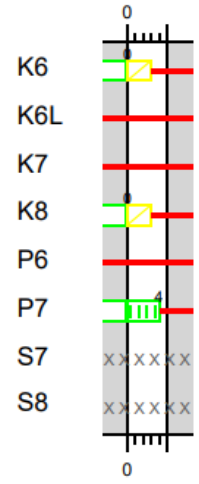
1-->4



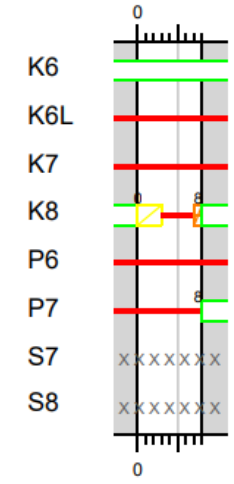
1-->5



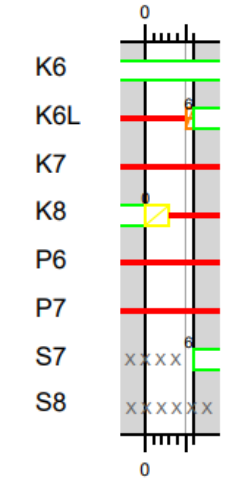
1-->6



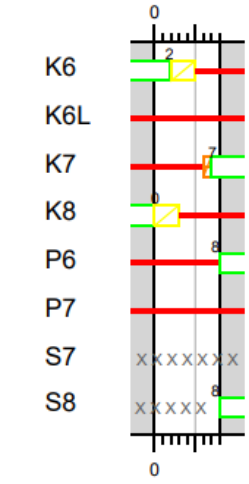
2-->1



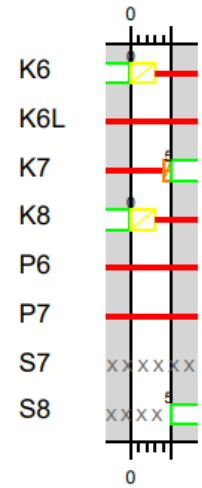
2-->3



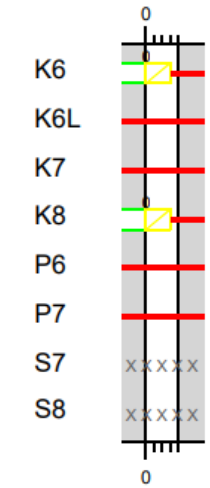
2-->4



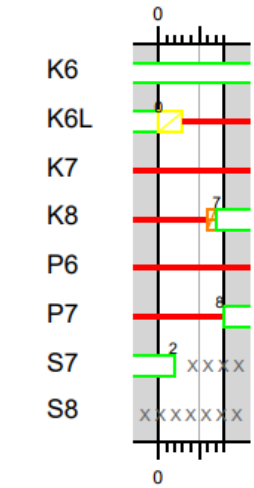
2-->5



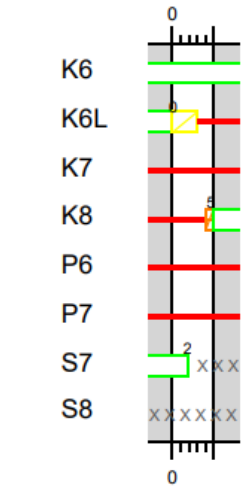
2-->6



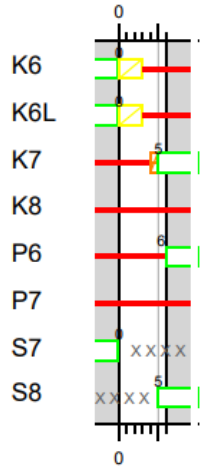
3-->1



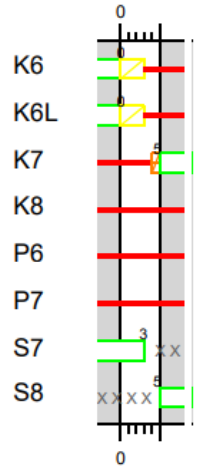
3-->2



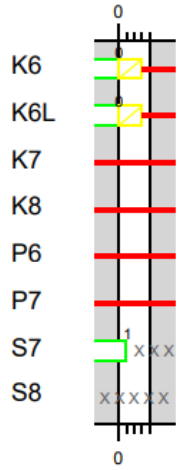
3-->4



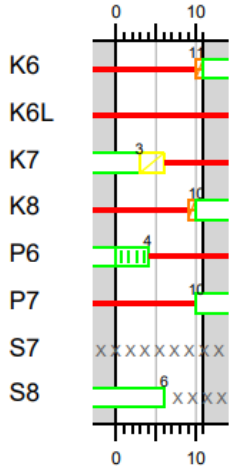
3-->5



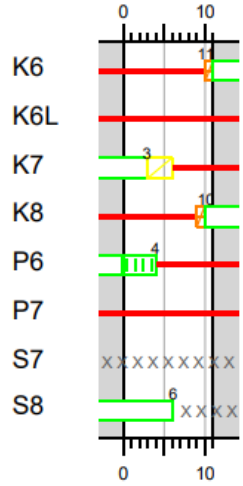
3-->6



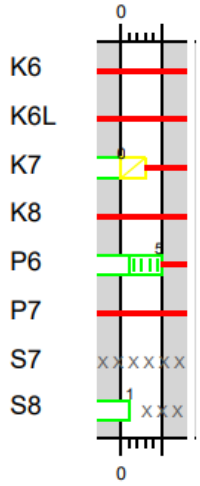
4-->1



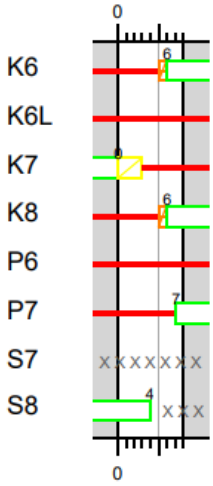
4-->2



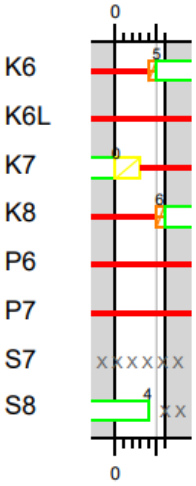
4-->6



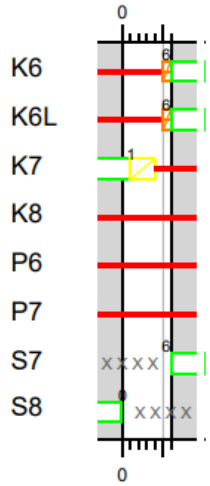
5-->1



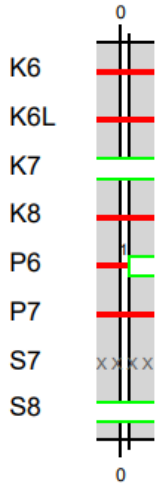
5-->2



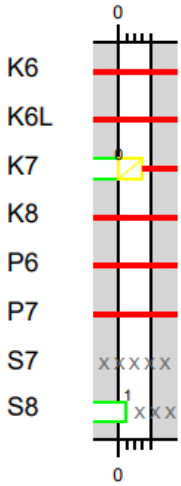
5-->3



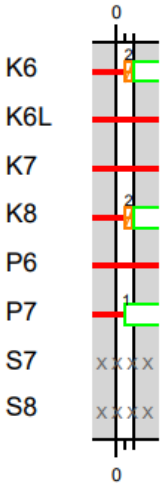
5-->4



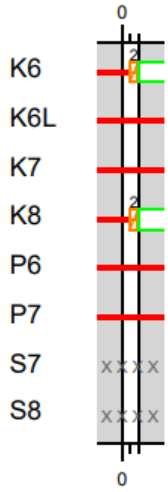
5-->6



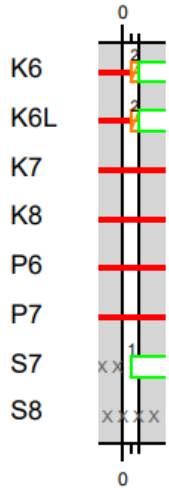
6-->1



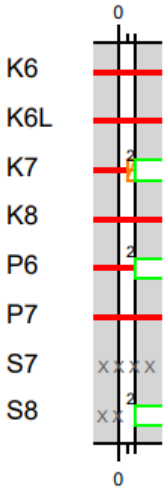
6-->2


0
K6
K6L
K7
K8
P6
P7
S7
S8
0

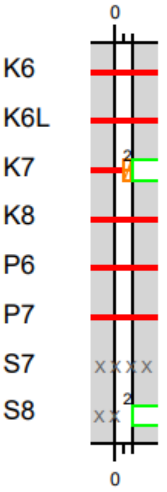
6-->3


0
K6
K6L
K7
K8
P6
P7
S7
S8
0

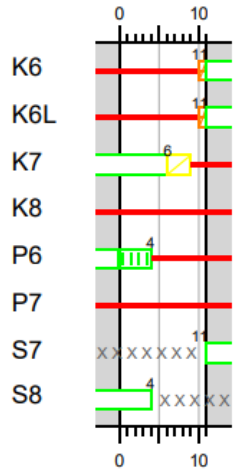
6-->4


0
K6
K6L
K7
K8
P6
P7
S7
S8
0

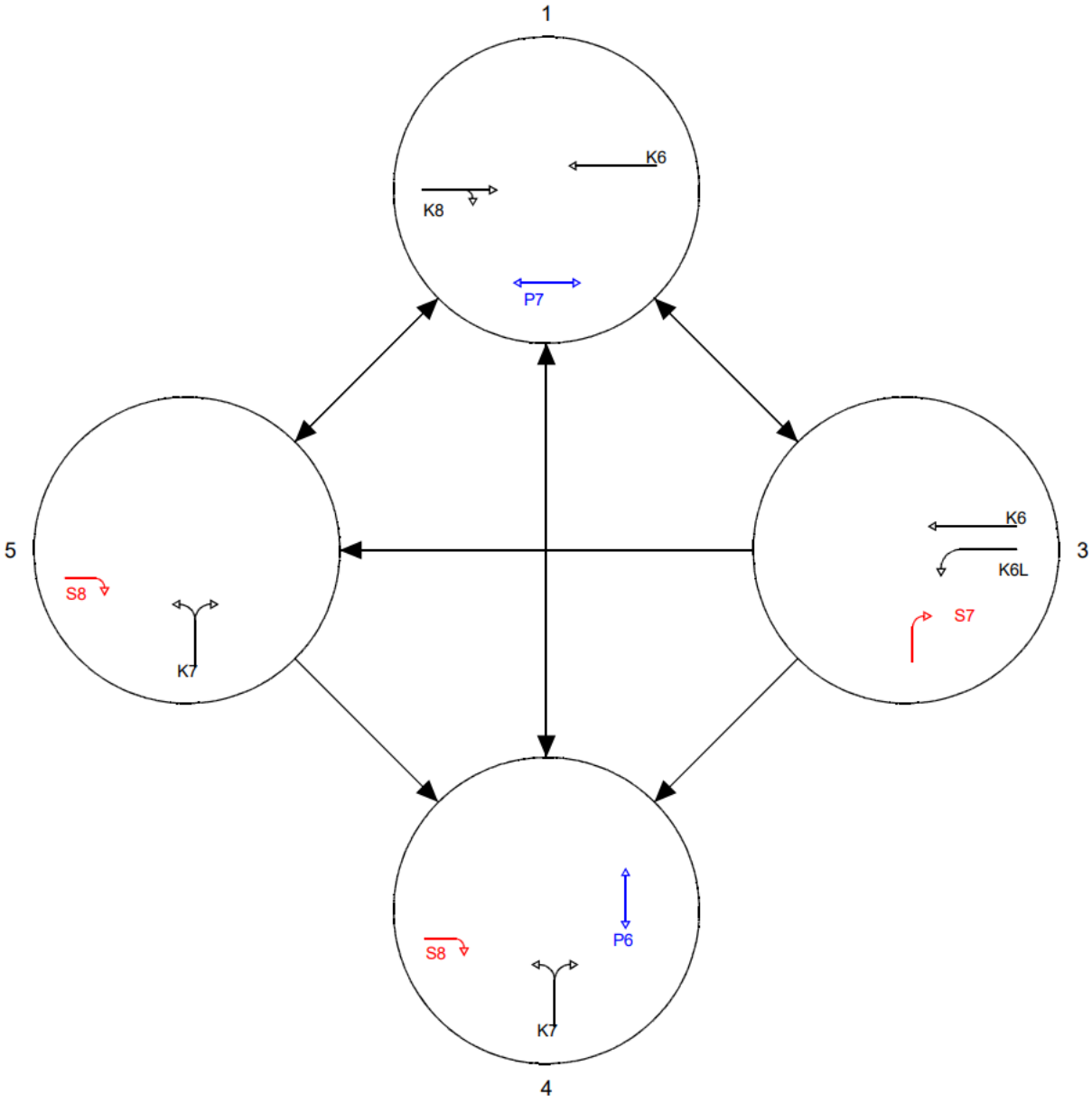
6-->5

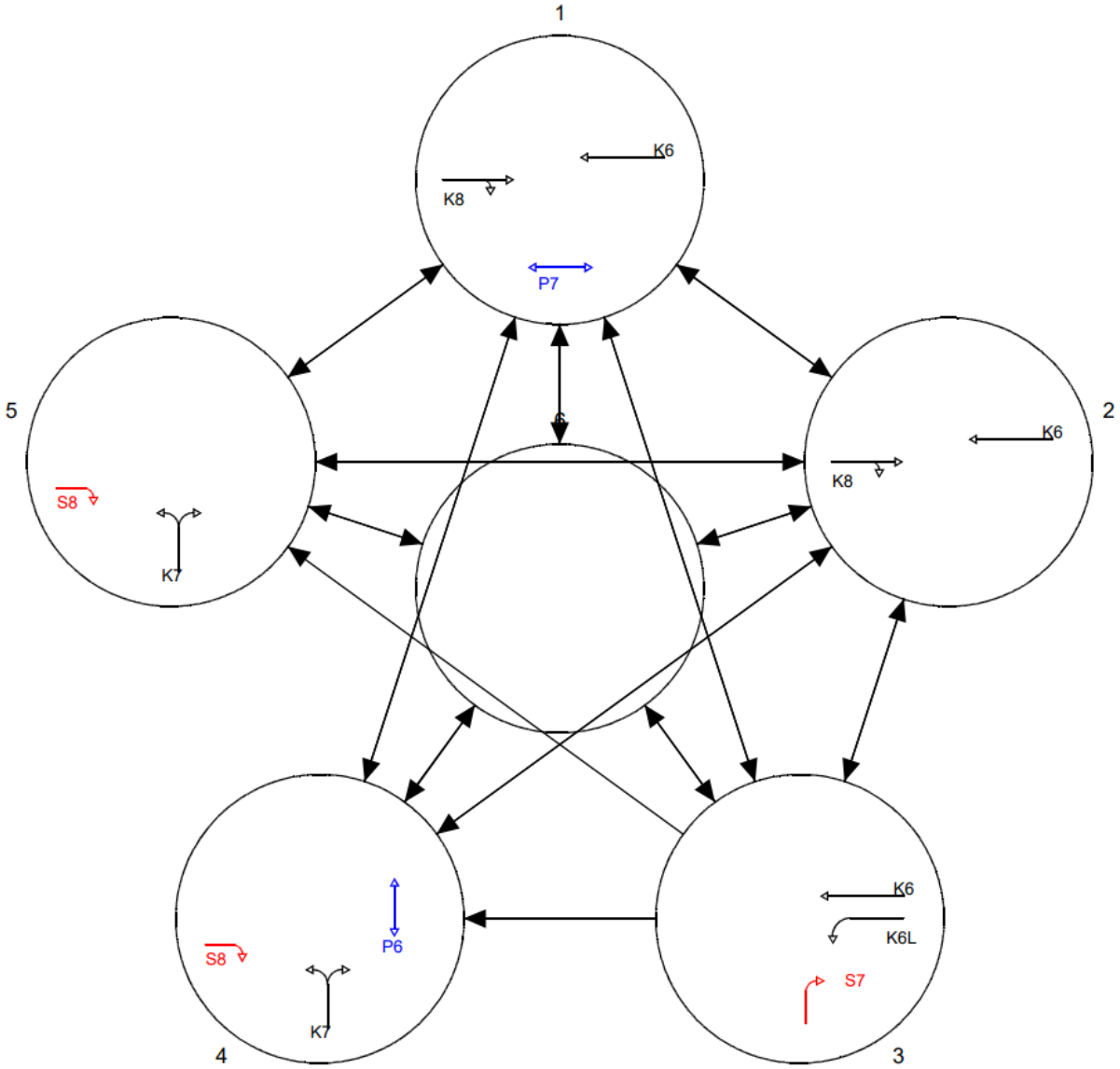

0
K6
K6L
K7
K8
P6
P7
S7
S8
0

4-->3


0
10
K6
K6L
K7
K8
P6
P7
S7
S8
0
10

Przykładowe układy faz ruchu:




1
K6
K8
P7
5
S8
K7
2
K6
K8
4
P6
S8
K7
3
K6
K6L
S7