POLITECHNIKA GDAŃSKA

# Systemy Czasu Rzeczywistego (SCR)

## Wykład 10 i 11: Modelowanie systemu sterującego z wykorzystaniem modelu maszyny stanowej, przybornik StateFlow Matlab/Simulink – szybkie wprowadzenie

WYDZIAŁ ELEKTROTECHNIKI I AUTOMATYKI – KATEDRA INŻYNIERII SYSTEMÓW STEROWANIA
Kierunek: Automatyka i Robotyka
Studia stacjonarne I stopnia: rok II, semestr IV
dr inż. Tomasz Rutkowski
2017

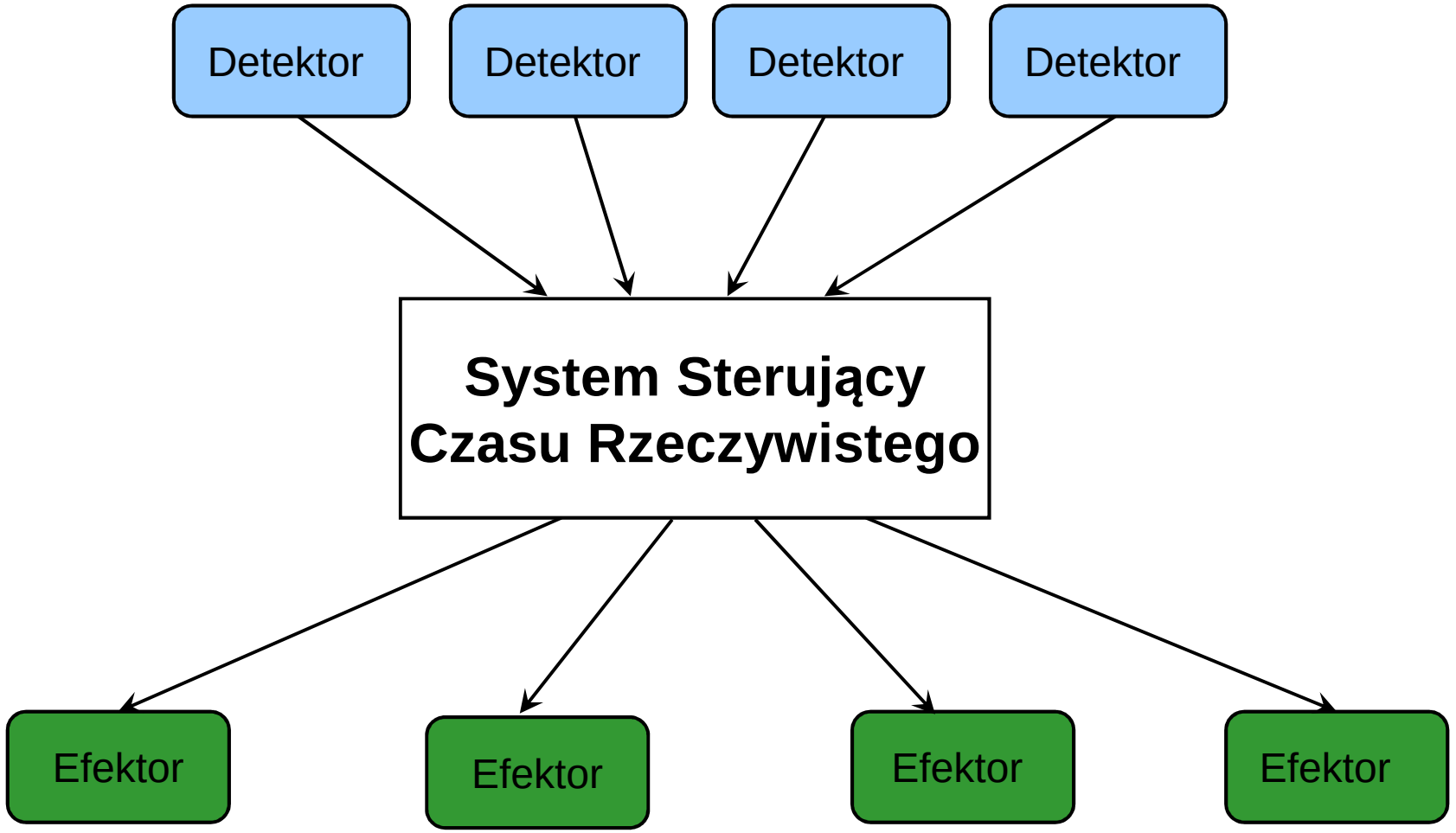
# *System Czasu Rzeczywistego - model „bodziec-reakcja”*

# System Czasu Rzeczywistego - model „bodziec-reakcja (odpowiedź)”

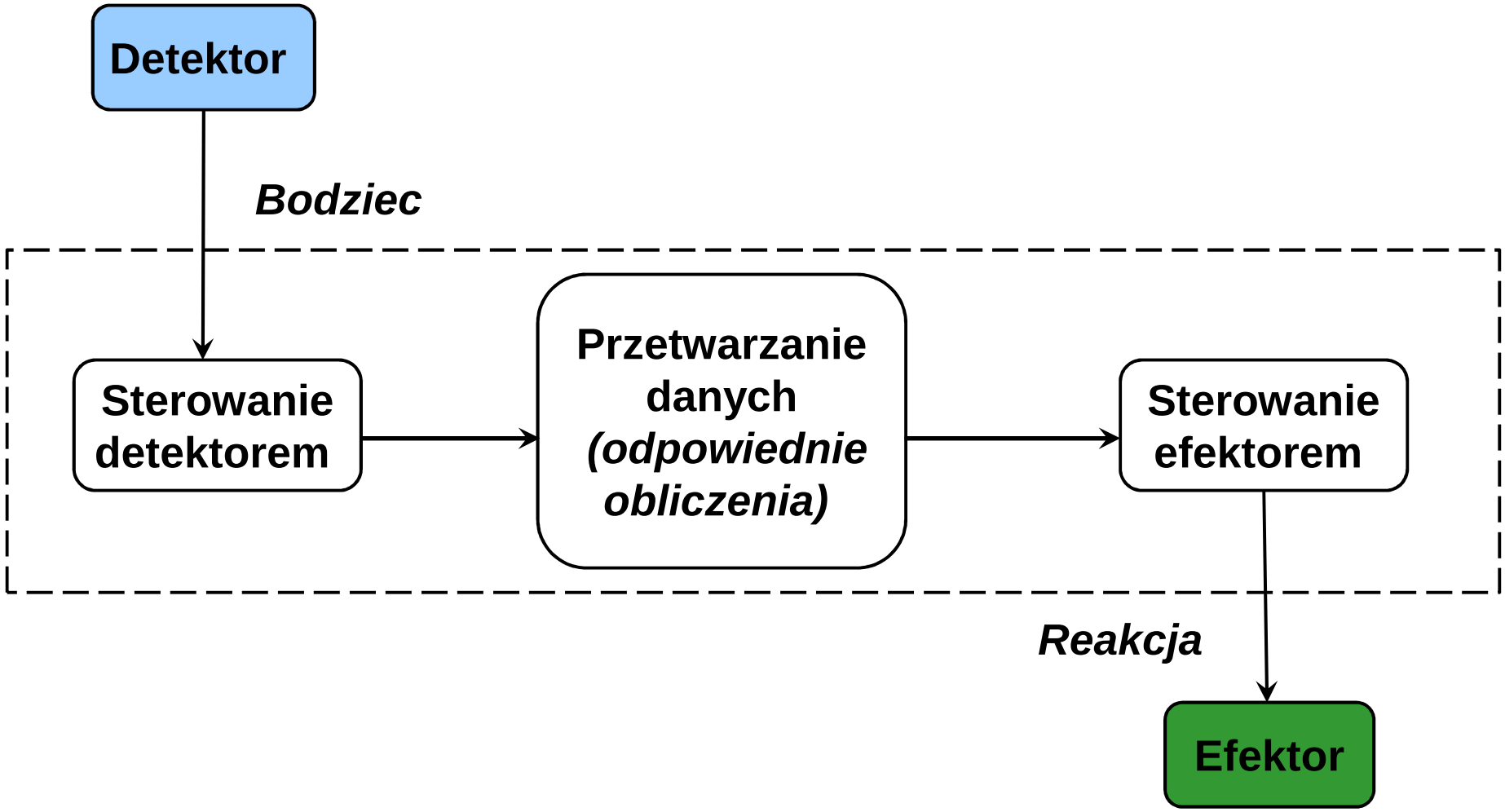
- Na podstawie bodźca otrzymanego z otoczenia, system musi wygenerować odpowiednią reakcję (odpowiedź systemu)
- Bodźce mogą mieć charakter synchroniczny, asynchroniczny lub mieszany (co ma bezpośredni wpływ na wybór „typu” systemu sterującego)

- Zachowanie systemu czasu rzeczywistego można zdefiniować poprzez:
  - zidentyfikowanie i zestawienie bodźców, które może otrzymać
  - odpowiednie odpowiedzi (reakcje) skojarzone z danymi bodźcami
  - czasu, w którym należy przetworzyć dane bodźce

# System Czasu Rzeczywistego - model „bodziec-reakcja (odpowiedź)”



# System Czasu Rzeczywistego - model „bodziec-reakcja (odpowiedź)”



# System Czasu Rzeczywistego - model „bodziec-reakcja (odpowiedź)”

- Z każdym rodzajem detektora kojarzy się proces sterowania detektorem
- Procesy przetwarzania danych wyznaczają oczekiwaną odpowiedź na bodźce otrzymane przez system
- Procesy sterowania efektorami zarządzają działaniem efektorów

- Model w przedstawionej postaci umożliwia szybkie odbieranie danych od detektorów (zanim będą gotowe następne dane wejściowe), późniejsze ich przetwarzanie i oddziaływanie na proces przez efektory

# System Czasu Rzeczywistego - podstawowe wymogi projektowe

- Część procesu projektowania systemu polega na podjęciu decyzji, które funkcje systemu będą zaimplementowane przez oprogramowanie, a które przez sprzęt (np.  te związane z ograniczeniami czasowymi)

- Proces projektowania systemu może więc zarówno obejmować projektowanie odpowiedniego sprzętu jak i projektowanie oprogramowania czasu rzeczywistego

# System Czasu Rzeczywistego - podstawowe kroki procesu projektowania

1. Zidentyfikować bodźce, które system musi przetwarzać oraz skojarzone z nimi reakcje

2. Dla każdego bodźca i związanej z nim reakcji zidentyfikować wymagania czasowe, które dotyczą przetwarzania zarówno bodźca, jak i reakcji

3. Pogrupować bodźce i reakcje w kilku procesach współbieżnych

# System Czasu Rzeczywistego - podstawowe kroki procesu projektowania

4. Dla każdego bodźca i reakcji zaprojektować algorytmy do przeprowadzania koniecznych obliczeń (*projekty algorytmów często trzeba opracować we wczesnej fazie procesu, aby poznać ilość przetwarzania i czas potrzebny do wykonania niezbędnych obliczeń*)

5. Zaprojektować system szeregujący, który zapewni, że procesy będą uruchamiane w odpowiednich chwilach, tak by spełnić ograniczenia czasowe

6. Zintegrować system pod kontrolą modułu wykonawczego

# *Modelowanie systemu sterującego z wykorzystaniem*

# *modelu maszyny stanowej (automat skończony)*

# Model maszyny stanowej

- Służy do opisywania zachowania systemu dynamicznego, gdy reaguje on na wewnętrzne lub zewnętrzne zdarzenia (bodźce)
- Model maszyny stanowej zawiera skończoną liczbę stanów i skończoną liczbę przejść pomiędzy tymi stanami, które następują w wyniku odpowiednich zdarzeń
- Model przedstawiany w postaci diagramu obrazuje maszynę stanową w postaci odpowiedniego grafu złożonego
- Modele maszyn stanowych są integralna częścią metod projektowania systemów czasu rzeczywistego (głównie modelowanie zachowania systemów interaktywnych typu bodziec-reakcja)

# Model maszyny stanowej

- Stan obiektu – zestaw wszystkich wartości atrybutów (działań, akcji) oraz aktualnych powiązań danego obiektu z innymi obiektami, zmianę aktualnego stanu na inny może spowodować zajście pewnego zdarzenia.
- Zdarzenie – to specyfikacja zjawiska, które zachodzi w czasie i przestrzeni  w maszynie stanowej, zjawiska które jest związane z wystąpieniem bodźca, który może uruchomić przejście między odpowiednimi stanami. Zdarzeniami mogą być sygnały, wywołania, upływ czasu czy zmiana stanu.
- Przejście – to związek między dwoma stanami, wskazujący, że obiekt znajdujący się w pierwszym stanie i wykonywujący pewne akcje, przejdzie do drugiego stanu i wykonywania związanych z nim akcji, zawsze gdy zajdzie określone zdarzenie i będą spełnione określone warunki.

# Model maszyny stanowej

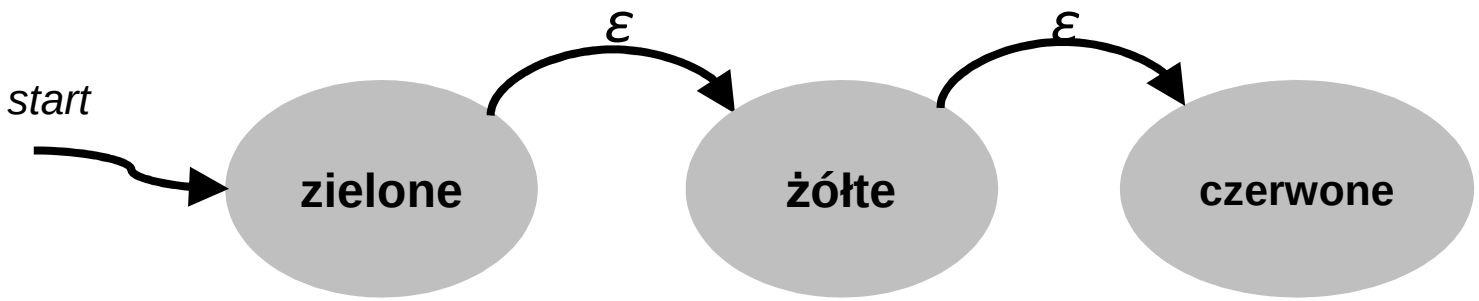
- Automaty skończone reagują na określone sygnały zewnętrzne reprezentowane przez litery alfabetu $\sum$, pewnego skończonego zbioru symboli: np. {0,1} czy, {ON, OFF}

- Automatem skończonym nazywamy system składający się z pięciu następujących elementów ($\sum$, S, $s_0$, $\delta$, F):
  - $\sum$ to alfabet automatu
  - S to skończony, niepusty zbiór wszystkich stanów automatu
  - $s_0$ to stan początkowy, taki że: $s_0 \in S$
  - $\delta$ to funkcja przejścia, taka że: $\delta$: S x $\sum \rightarrow$ S
  - F to skończony, niepusty zbiór stanów końcowych automatu

# Model maszyny stanowej

## Reprezentacja maszyny stanowe:

- graf działania prostego sygnalizatora świetlnego

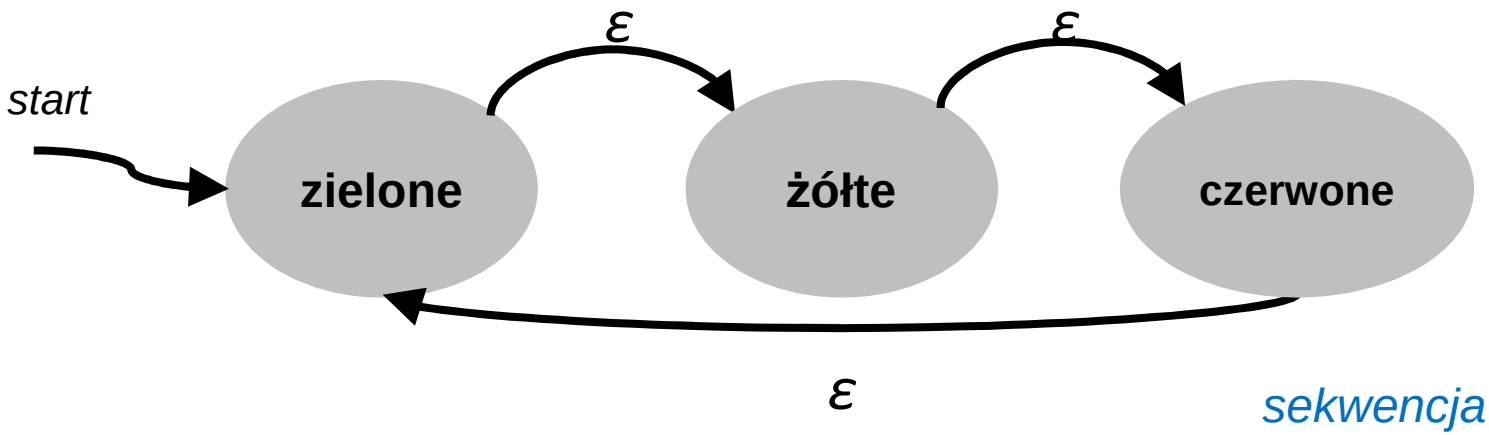



Σ = [ε] , np. zdarzenie -> upływ czasu
S = [zielone, żółte, czerwone]
$S_0$ = [zielone]
δ = [zielone/ε→żółte, żółte/ε→czerwone]
F = [czerwone]

# Model maszyny stanowej

Reprezentacja maszyny stanowe:

- graf działania prostego sygnalizatora świetlnego

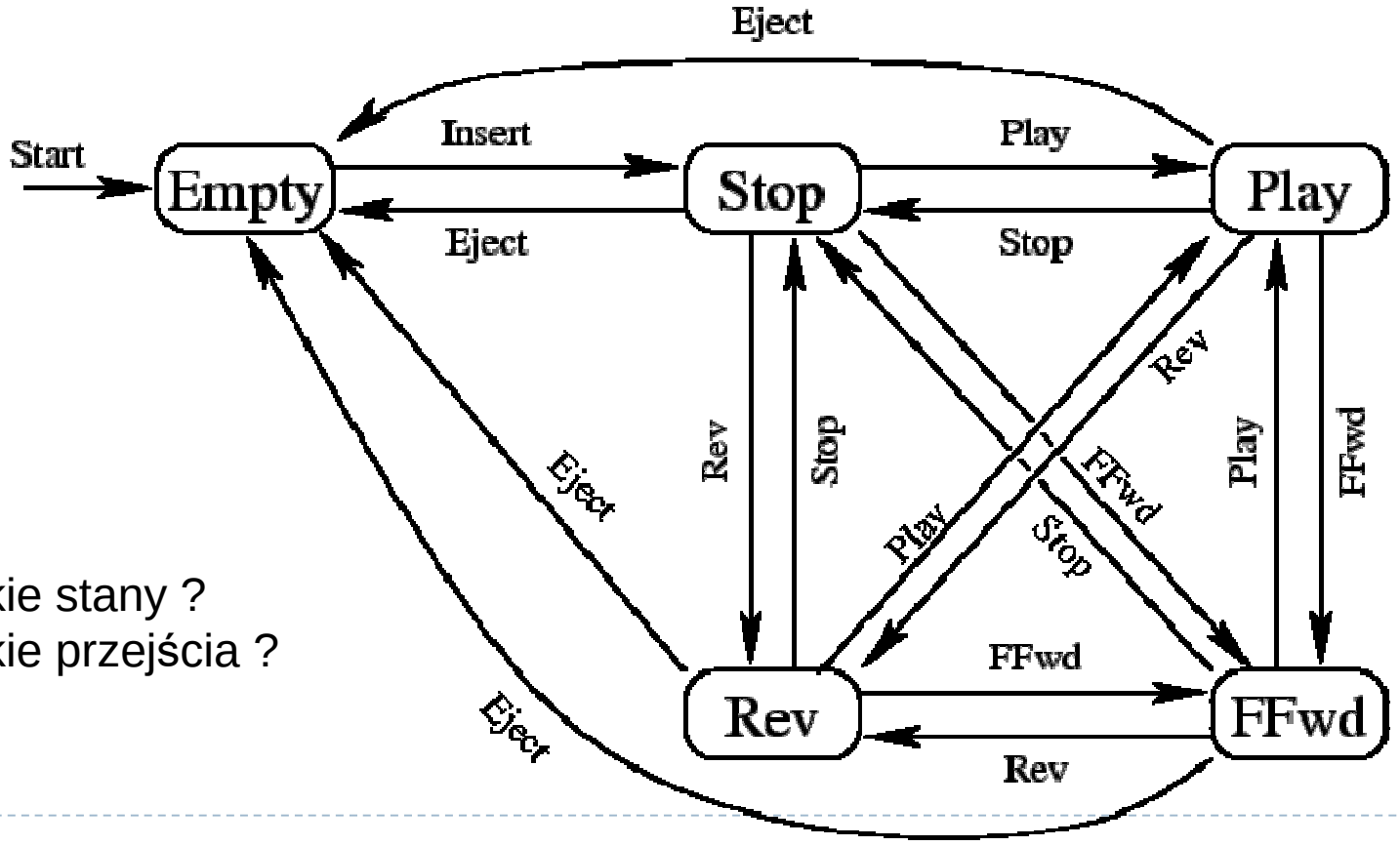



δ = [zielone/ε→żółte, żółte/ε→czerwone, czerwone/ε→zielone]

# Model maszyny stanowej

## Reprezentacja maszyny stanowej:

- graf działania prostego odtwarzacza video

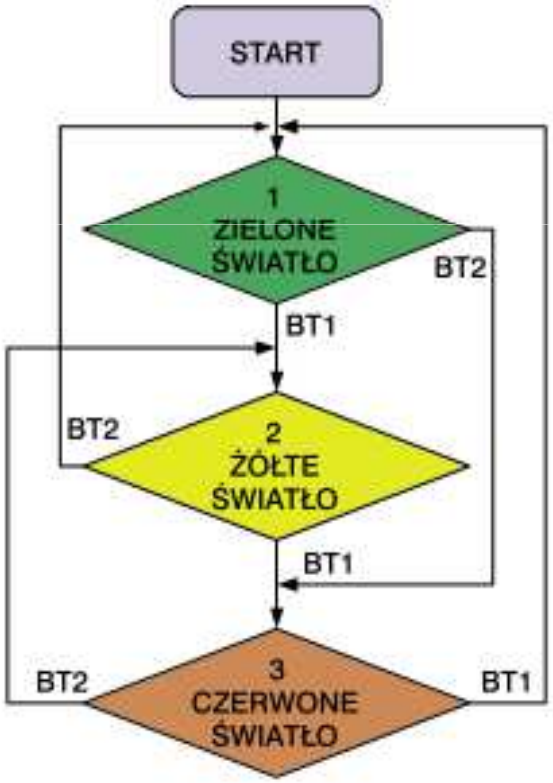



jakie stany ?
jakie przejścia ?

# *Przykład 1*

# Model maszyny stanowej - przykład układu przełącznika kolorowych świateł

Klasyczny algorytm blokowy

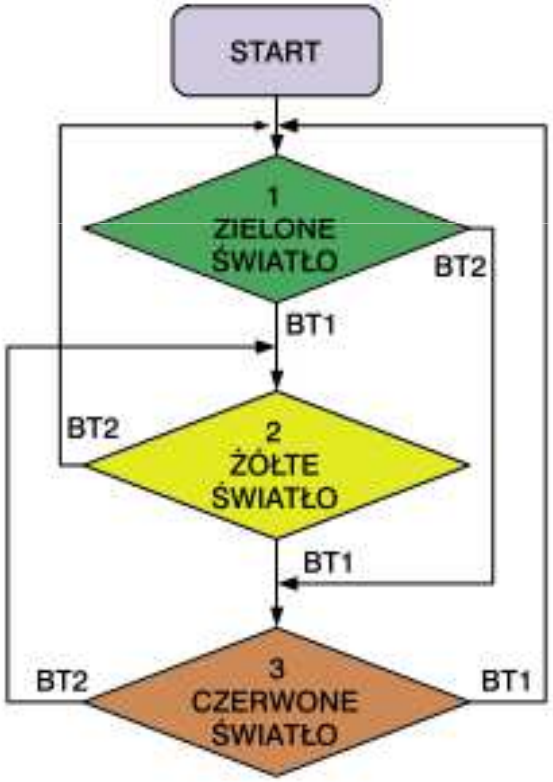



BT1 – Zdarzenie przyciśnięcie przycisku „w przód”
BT2 – Zdarzenie przyciśnięcie przycisku „w tył”

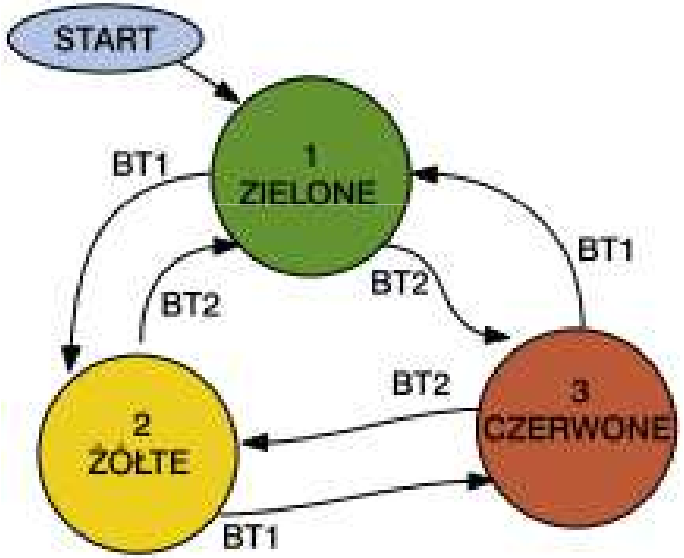
*Źródło: [2]*

# Model maszyny stanowej - przykład układu przełącznika kolorowych świateł

Klasyczny algorytm blokowy




Automat skończony




BT1 – Zdarzenie przyciśnięcie przycisku „w przód”
BT2 – Zdarzenie przyciśnięcie przycisku „w tył”

*Źródło: [2]*

# Model maszyny stanowej - przykład układu przełącznika kolorowych świateł

Stany i warunki przejścia

| Stan | Warunki |
|---|---|
| 1 | BT1->2, BT2->3 |
| 2 | BT1->3, BT2->1 |
| 3 | BT1->1, BT2->2 |

Macierz przejść

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | | BT1 | BT2 |
| 2 | BT2 | | BT1 |
| 3 | BT1 | BT2 | |

*Źródło: [2]*

# Model maszyny stanowej

*Przykładowa realizacja w języku C*

```
#define ZIELONE  1
#define ZOLTE    2
#define CZERWONE 3

#define BT1      1
#define BT2      2

void swiatlo(int s)
{
...
}

char pobierz_przycisk()
{
...
}

int main()
{
  int stan=ZIELONE; // poczatkowy stan
  char przycisk;
  while(1) {
    przycisk=pobierz_przycisk();
    if (przycisk==BT1){
      switch(stan){
        case ZIELONE: stan=ZOLTE; break;
        case ZOLTE: stan=CZERWONE; break;
        case CZERWONE: stan=ZIELONE; break;
      }
    }
    if (przycisk==BT2){
      switch(stan){
        case ZIELONE: stan=CZERWONE; break;
        case ZOLTE: stan=ZIELONE; break;
        case CZERWONE: stan=ZOLTE; break;
      }
    }
    swiatlo(stan);
  }
  return 0;
}
```

# *Przykład 2*

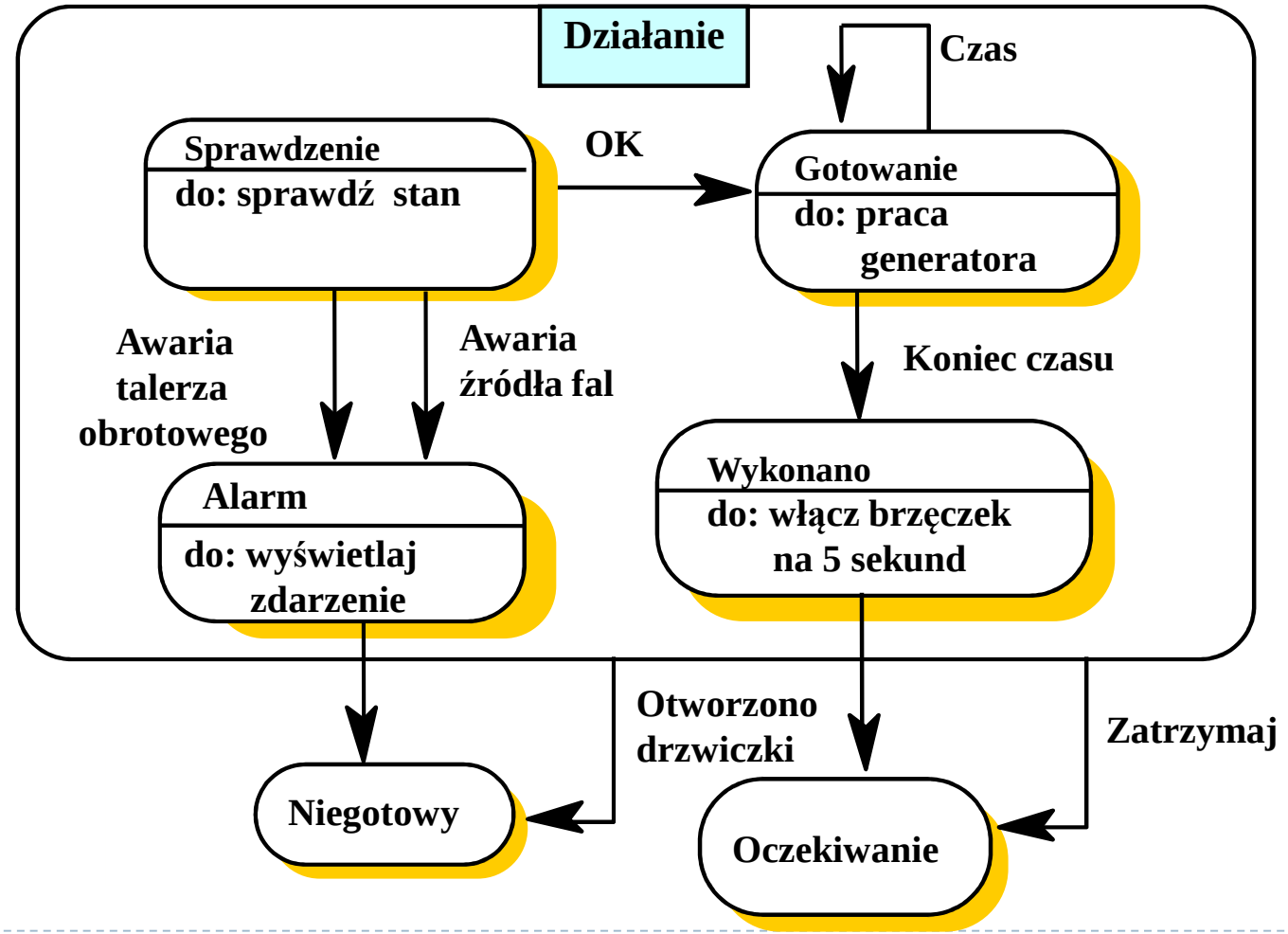
# Model maszyny stanowej - opis stanów prostej kuchenki mikrofalowej

| Stan | Opis |
|---|---|
| **Oczekiwanie** | Kuchenka czeka na dane wejściowe. Wyświetlacz pokazuje aktualną godzinę. |
| **Połowa mocy** | Moc kuchenki ustawiono na 500 watów. Wyświetlacz pokazuje napis „Połowa mocy”. |
| **Pełna moc** | Moc kuchenki ustawiono na 1000 watów. Wyświetlacz pokazuje napis „Pełna moc”. |
| **Ustawienie czasu** | Czas trwania gotowania jest ustawiany na wartość wprowadzoną przez użytkownika. Wyświetlacz pokazuje wybrany czas gotowania i jest aktualizowany w miarę wprowadzania danych. |
| **Niegotowy** | Kuchenka nie może działać ze względów bezpieczeństwa. Wewnętrzne światło kuchenki jest włączone. Wyświetlacz pokazuje napis „Niegotowy”. |
| **Gotowy** | Kuchenka jest gotowa do działania. Wewnętrzne światło jest wyłączone. Wyświetlacz pokazuje napis „Gotowy”. |
| **Działanie** | Kuchenka działa. Wewnętrzne światło jest włączone. Wyświetlacz odlicza zmniejszający się czas pozostały do zakończenia gotowania. Po zakończeniu brzęczyk włącza się na 5 sekund. Światło kuchenki jest wtedy włączone. Wyświetlacz pokazuje napis „Gotowanie zakończone” w czasie działania brzęczka. |

*Źródło: [3]*

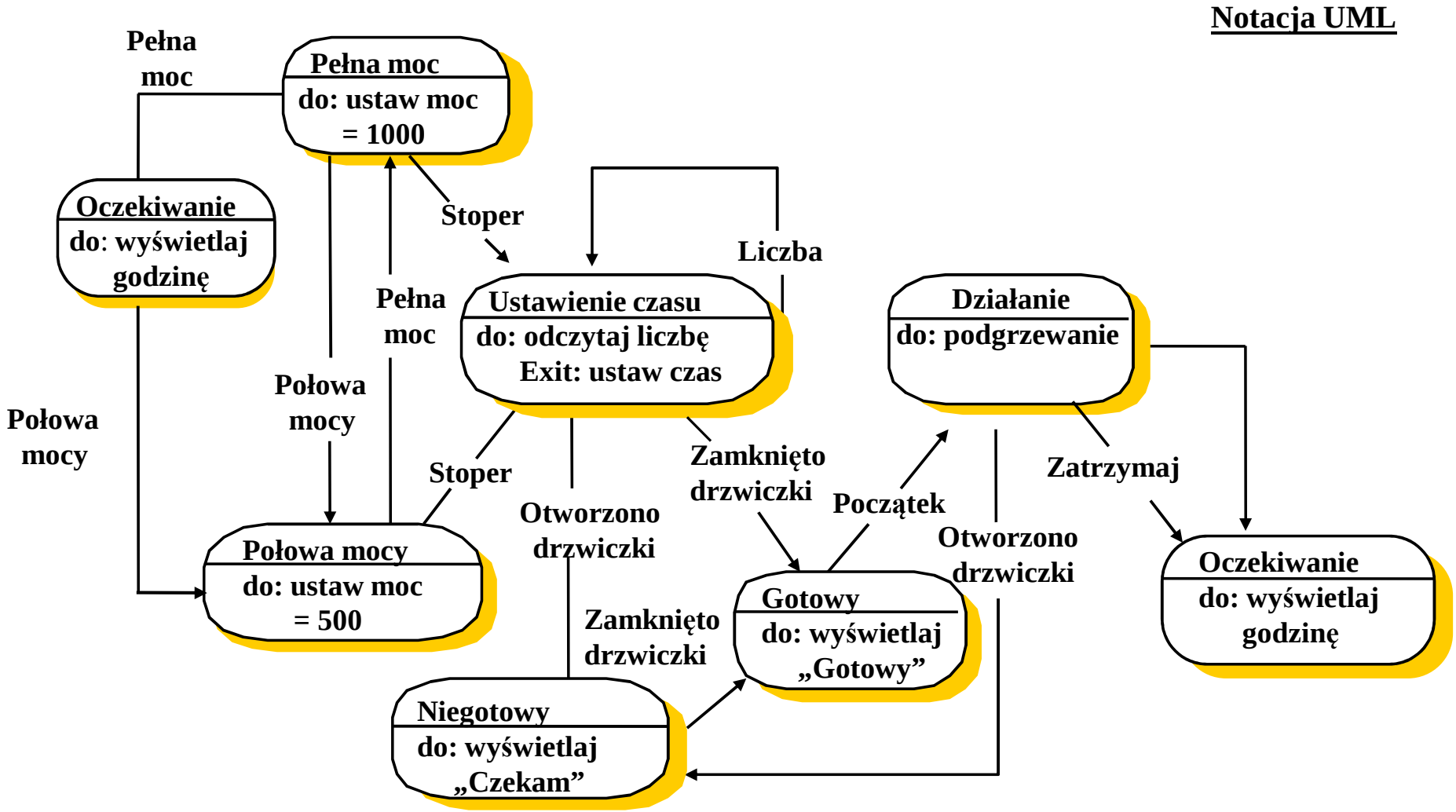
# Model maszyny stanowej - opis działania prostej kuchenki mikrofalowej

**Notacja UML**




*Źródło: [3]*

# Model maszyny stanowej - prosta kuchenka mikrofalowa




*Źródło: [3]*

# Model maszyny stanowej - opis bodźców dla prostej kuchenki mikrofalowej

| Bodziec | Opis |
|---|---|
| Połowa mocy | Użytkownik nacisnął przycisk "Połowa mocy" |
| Pełna moc | Użytkownik nacisnął przycisk "Pełna moc" |
| Stoper | Użytkownik nacisnął przycisk "Stoper" |
| Liczba | Użytkownik nacisnął przycisk z liczbą |
| Otworzono drzwiczki | Przełącznik drzwiowy jest niezamknięty |
| Zamknięto drzwiczki | Przełącznik drzwiowy jest zamknięty |
| Początek | Użytkownik nacisnął przycisk "Początek" |
| Zatrzymaj | Użytkownik nacisnął przycisk "Zatrzymaj" |

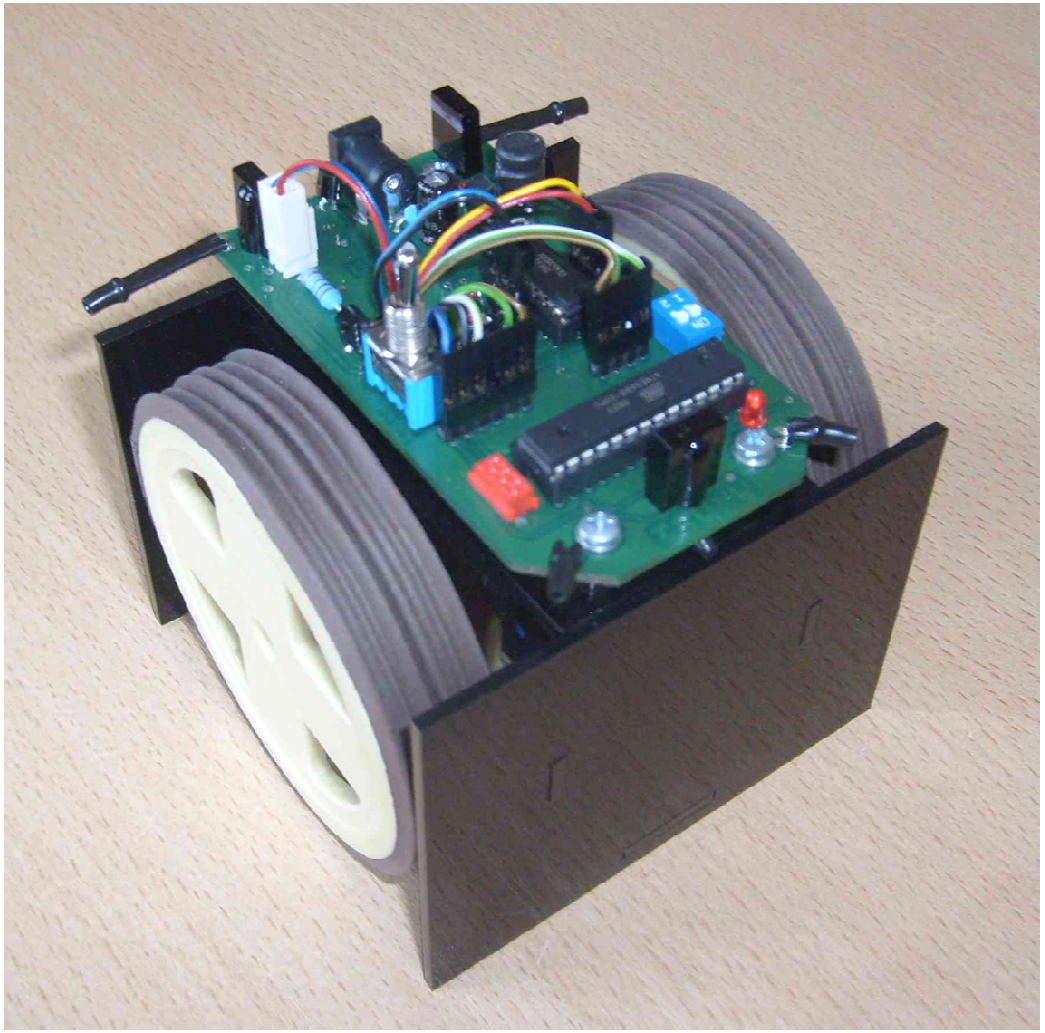
*Źródło: [3]*

# Model maszyny stanowej - opis reakcji dla prostej kuchenki mikrofalowej

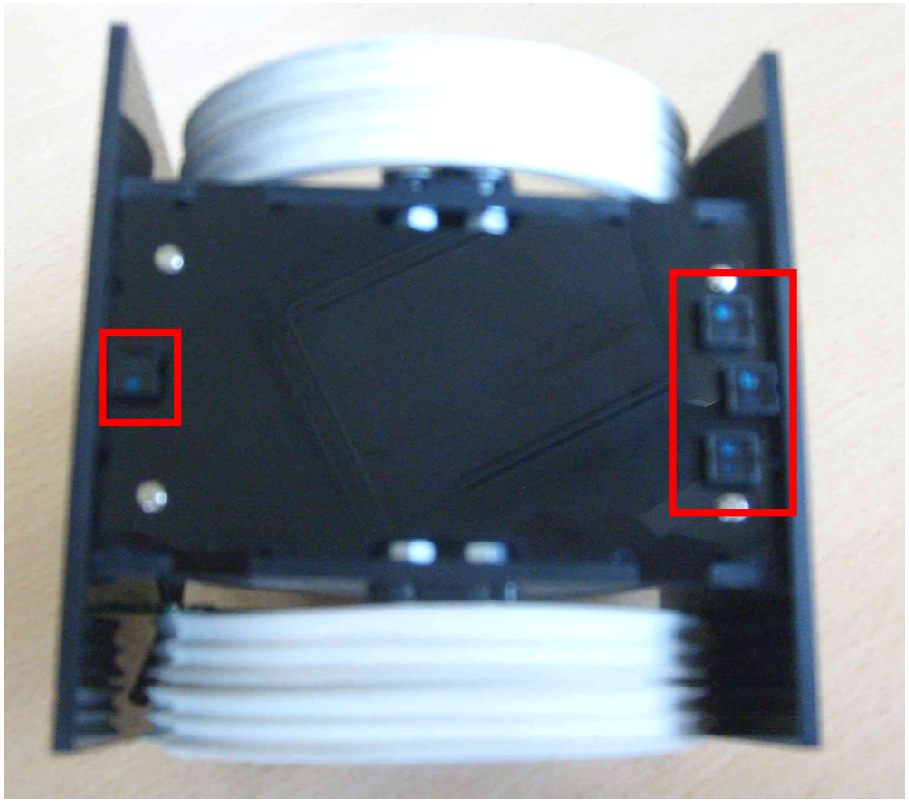
| Reakcja | Opis |
|---|---|
| ? | ? |
| … | … |

# *Przykład 3*

# „Prosty wojownik mini-sumo”

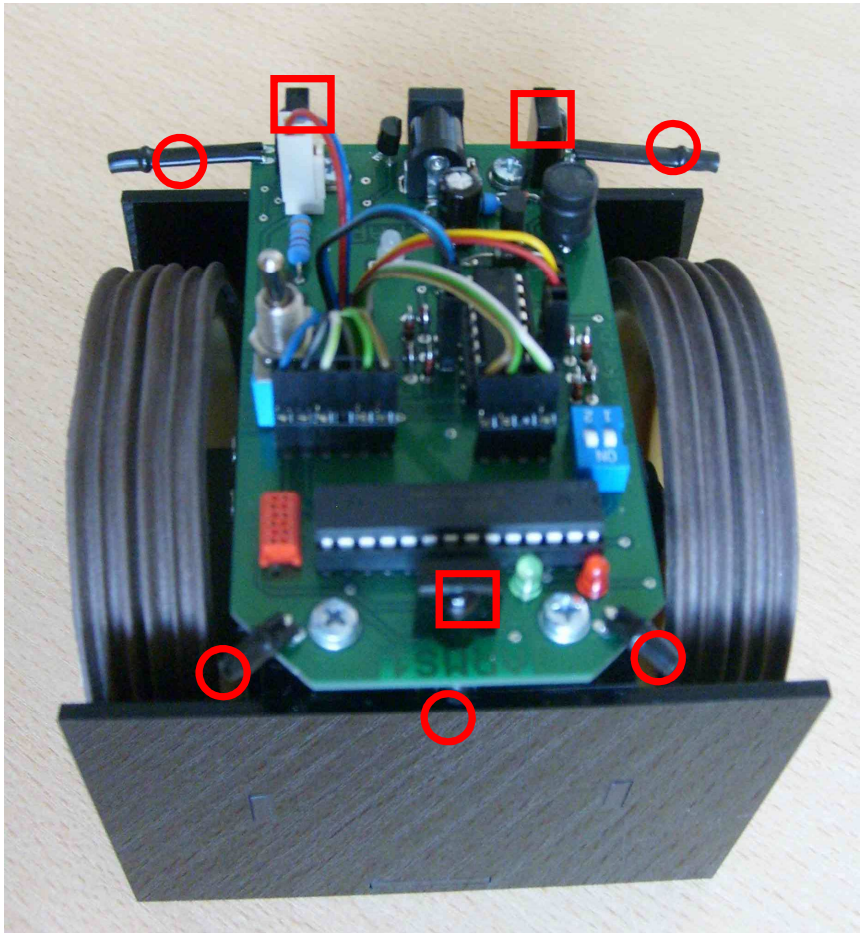


# „Prosty wojownik mini-sumo”



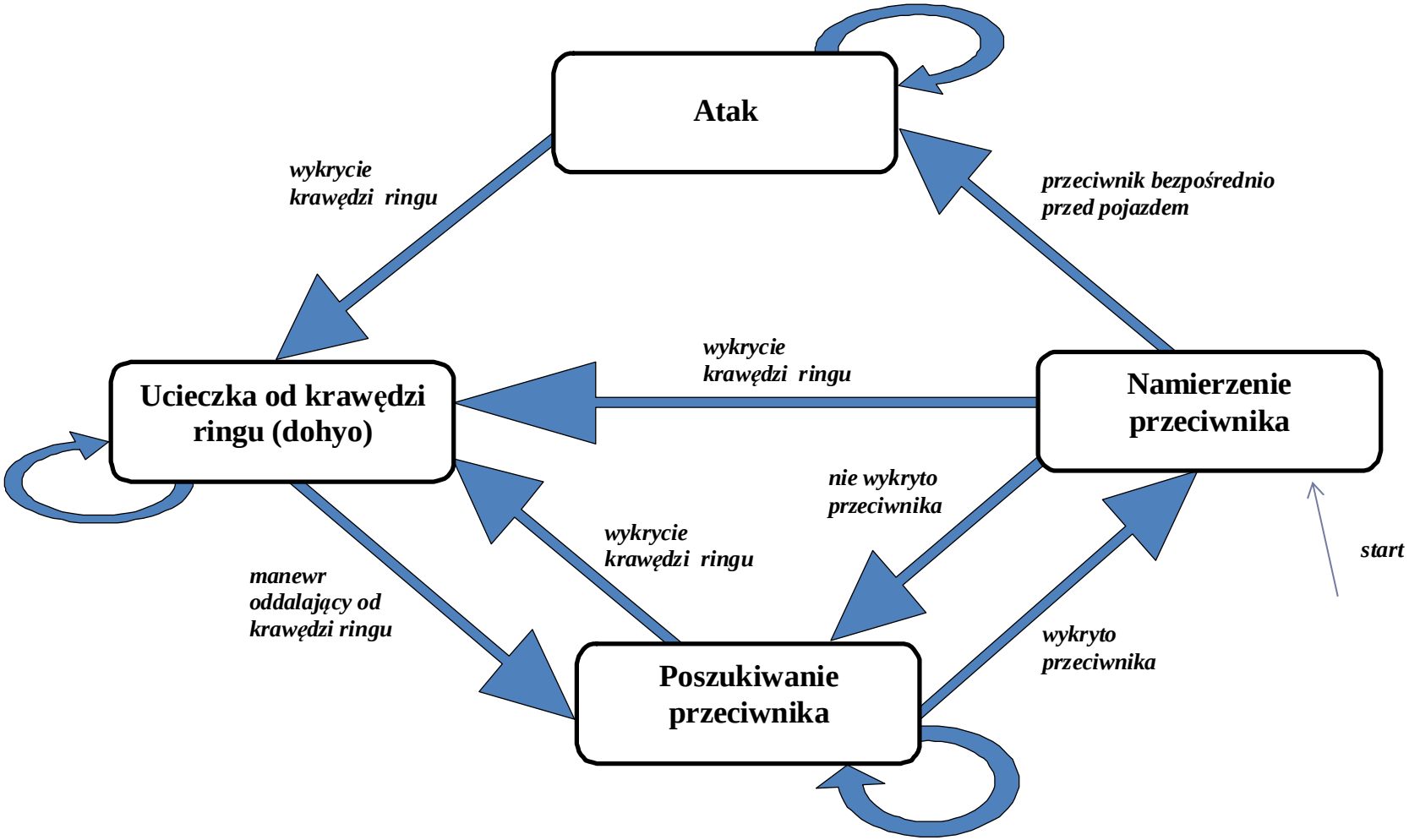
Zestaw czujników
w postaci modułów odbiciowych
wykrywających białą linię
*(z czujnikiem podczerwieni)*

Zestaw nadajników ○ i odbiorników □ podczerwieni pełniący funkcję czujników obecności przeciwnika



Zastanówmy się jakie są:
- bodźce ?
- detektory?
- efektory?
- reakcje?

# Model maszyny stanowej - „prosty wojownik mini-sumo”



# Model maszyny stanowej - opis stanów „prostego wojownika mini-sumo”

| STAN | OPIS | PROPOZYCJE AKCJI (REAKCJI) |
|---|---|---|
| **Ucieczka od krawędzi ringu (dohyo)** | Stan w który wchodzi pojazd gdy jeden z czujników sygnalizuje wykrycie krawędzi ringu (dohyo) – czujnik białej linii.<br>Pojazd powinien podjąć akcje związaną z utrzymaniem się na ringu, np.: skierowanie pojazdu do środka ringu. | •jeżeli wykryto krawędź ringu z tyłu pojazdu:<br>•jazda do przodu<br>•przejście do stanu „**Poszukiwania przeciwnika**”<br>•jeżeli wykryto krawędź ringu z tyłu pojazdu:<br>•wyznaczenie kierunku obrotu pojazdu (w lewo czy w prawo)<br>•wykonanie pełnego obrotu o 180° (zawracanie)<br>•przejście do stanu „**Poszukiwania przeciwnika**” |
| **Poszukiwanie przeciwnika** | Stan w którym podejmowane są akcje mające na celu zlokalizowanie przez pojazd przeciwnika na ringu.<br>Pojazd powinien podjąć akcje związaną ze skanowaniem swojego otoczenia w celu wykrycia przeciwnika (sygnał z radaru) np.: jazda przed siebie z cykliczną zmianą kierunku czy jazda po łukach. | •realizacja dostępnych algorytmów pozwalających wykrycie przeciwnika:<br>•jazda przed siebie z cykliczną zmianą kierunku ruchu<br>•jazda po łukach<br>•jeżeli wykryto krawędź ringu to przejdź do stanu „**Ucieczki od krawędzi ringu**”<br>•jeżeli wykryto przeciwnika w bezpośrednim otoczeniu pojazdu to przejdź do stanu „**Namierzania przeciwnika**” |
| **Namierzenie przeciwnika** *(start)* | Stan w którym pojazd wykrył przeciwnika w swoim otoczeniu (sygnał z radaru).<br>Pojazd powinien podjąć akcje mające na celu zwrócenie się przodem do przeciwnika. | •jeżeli przeciwnik zlokalizowany z prawej strony to obrót w prawo<br>•jeżeli przeciwnik zlokalizowany z lewej strony to obrót w lewo<br>•jeżeli przeciwnik zlokalizowany z tyłu to pełen obrotu o 180°<br>•jeżeli przeciwnik zlokalizowany bezpośrednio przed pojazdem to przejdź do stanu „**Atak**”<br>•jeżeli wykryto krawędź ringu to przejdź do stanu „**Ucieczki od krawędzi ringu**”<br>•w przeciwnym wypadku przejdź do stanu „**Poszukiwania przeciwnika**” |
| **Atak** | Stan w którym przeciwnik został zlokalizowany i znajduje się bezpośrednio z przodu pojazdu (sygnał z radaru).<br>Pojazd powinien ruszyć przed siebie z pełną mocą silników by wypchnąć przeciwnika z ringu. | •jazda do przodu z pełną mocą silników<br>•jeżeli wykryto krawędź ringu to przejdź do stanu „**Ucieczki od krawędzi ringu**” |

# Model maszyny stanowej

## - przykład fragmentu programu w C dla „prostego wojownika mini-sumo”

```
0035    while(1){
0036
0037        // Sygnal z radaru ze przeciwnik centralnie z przodu
0038        if (Radar_przod() ){
0039            // "atak"
0040            // ---- tu odpowiednie linie kodu ----
0041        }else {
0042            // Sygnal z radaru ze przeciwnik z przodu z prawej strony
0043            if (Radar_przod_prawo() ){
0044                // "zakrec w prawo"
0045                // ---- tu odpowiednie linie kodu ----
0046            }else{
0047                // Sygnal z radaru ze przeciwnik z przodu z lewej strony
0048                if (Radar_przod_lewo() ){
0049                    // "zakrec w lewo"
0050                    // ---- tu odpowiednie linie kodu ----
0051                }else{
0052                    // Sygnal z radaru ze przeciwnik z boku z prawej strony
0053                    if (Radar_prawo() ) {
0054                        // "zakrec w prawo"
0055                        // ---- tu odpowiednie linie kodu ----
0056                    }else{
0057                        // Sygnal z radaru ze przeciwnik z boku z lewej strony
0058                        if (Radar_lewo() ){
0059                            // "zakrec w lewo"
0060                            // ---- tu odpowiednie linie kodu ----
0061                        }else {
0062                            // Brak sygnalu z radaru
0063                            // Poszukiwanie przeciwnika na dohyo (jazda po lini prostej lub po luku
0064                            // w lewa strone lub prawa strone by zlokalizowac przeciwnika
0065                            // ---- tu odpowiednie linie kodu ----
0066    } } } } }
```

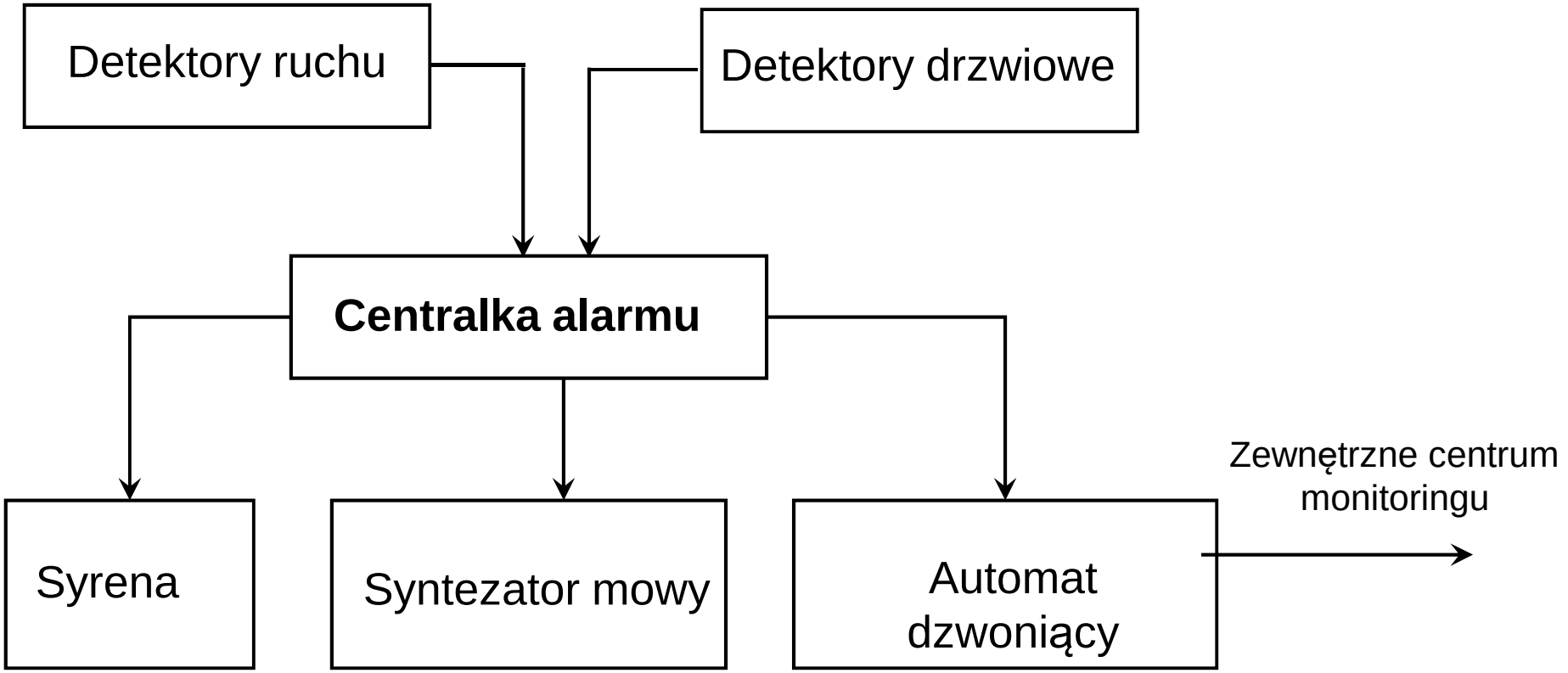
**Atak**

**Namierzanie przeciwnika**

**Poszukiwanie przeciwnika**

# *Przykład 4*

# Modelowanie systemu - prosty system antywłamaniowy




*Źródło: [3]*

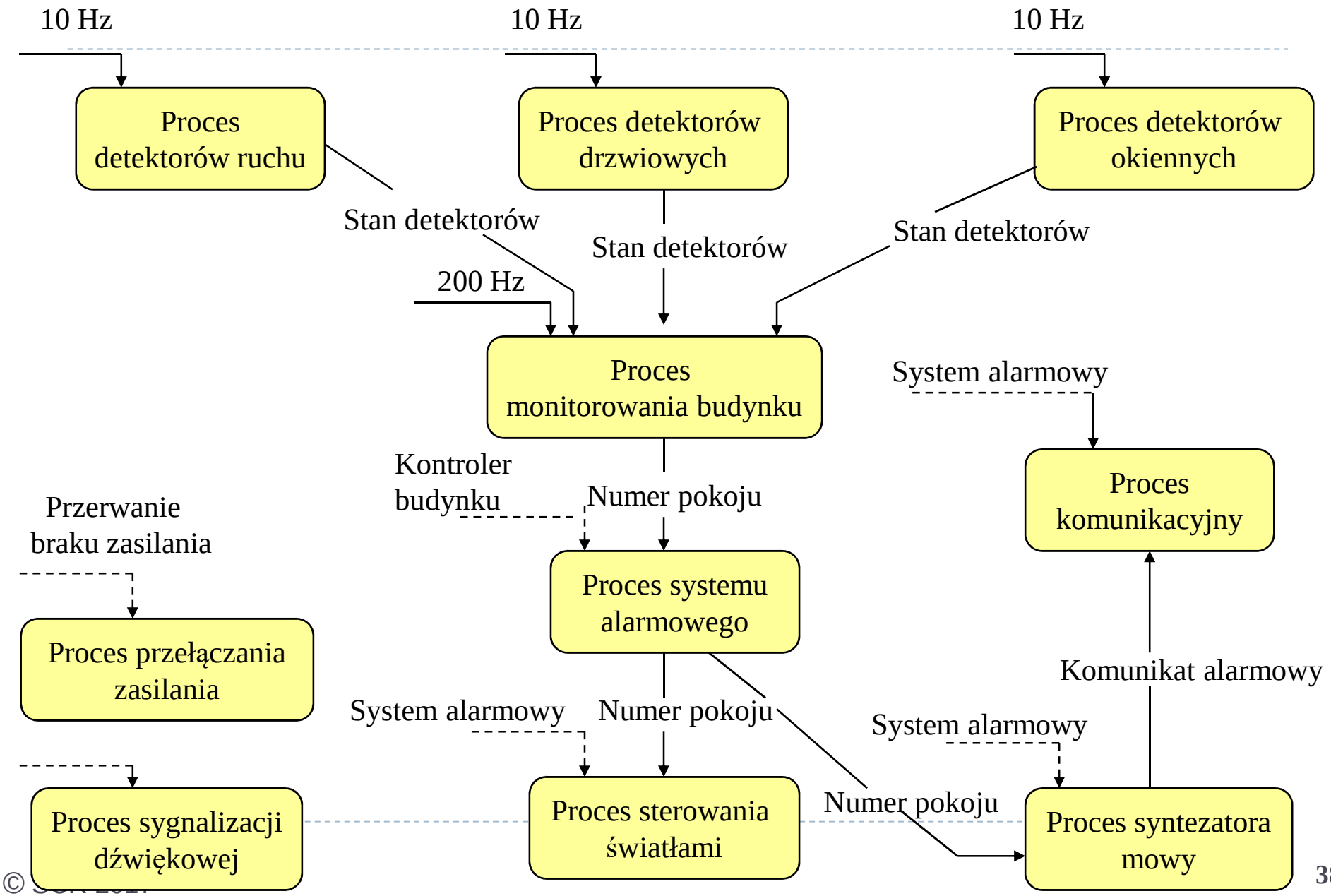
# Model maszyny stanowej - bodźce/reakcje „prostego systemu antywłamaniowego”

| Bodziec/reakcja | Wymagania czasowe |
|---|---|
| Przerwanie zanik zasilania | Przełączenie na zasilanie zapasowe musi być ukończone po najwyżej 50ms |
| Alarm drzwiowy | Każdy detektor drzwiowy musi być odpytywany co najmniej dwa razy na sekundę |
| Alarm okienny | Każdy detektor okienny musi być odpytywany co najmniej dwa razy na sekundę |
| Detektor ruchu | Każdy detektor ruchu powinien być odpytywany co najmniej dwa razy na sekundę |
| Sygnał dźwiękowy | Sygnał dźwiękowy musi być włączony po upływie najwyżej 1 sekundy od alarmu wywołanego przez detektor |
| Włączenie świateł | Światła powinny być włączone po upływie najwyżej 1/2 sekundy od alarmu wywołanego przez detektor |
| Komunikacja | Wezwanie policji przez telefon należy rozpocząć po upływie najwyżej 2 sekund od alarmu wywołanego przez detektor |
| Syntezator mowy | Komunikat z syntezatora powinien być dostępny po upływie najwyżej 4 sekund od alarmu wywołanego przez detektor |

*Źródło: [3]*

# Architektura procesowa „prostego systemu antywłamaniowego”

*Źródło: [3]*

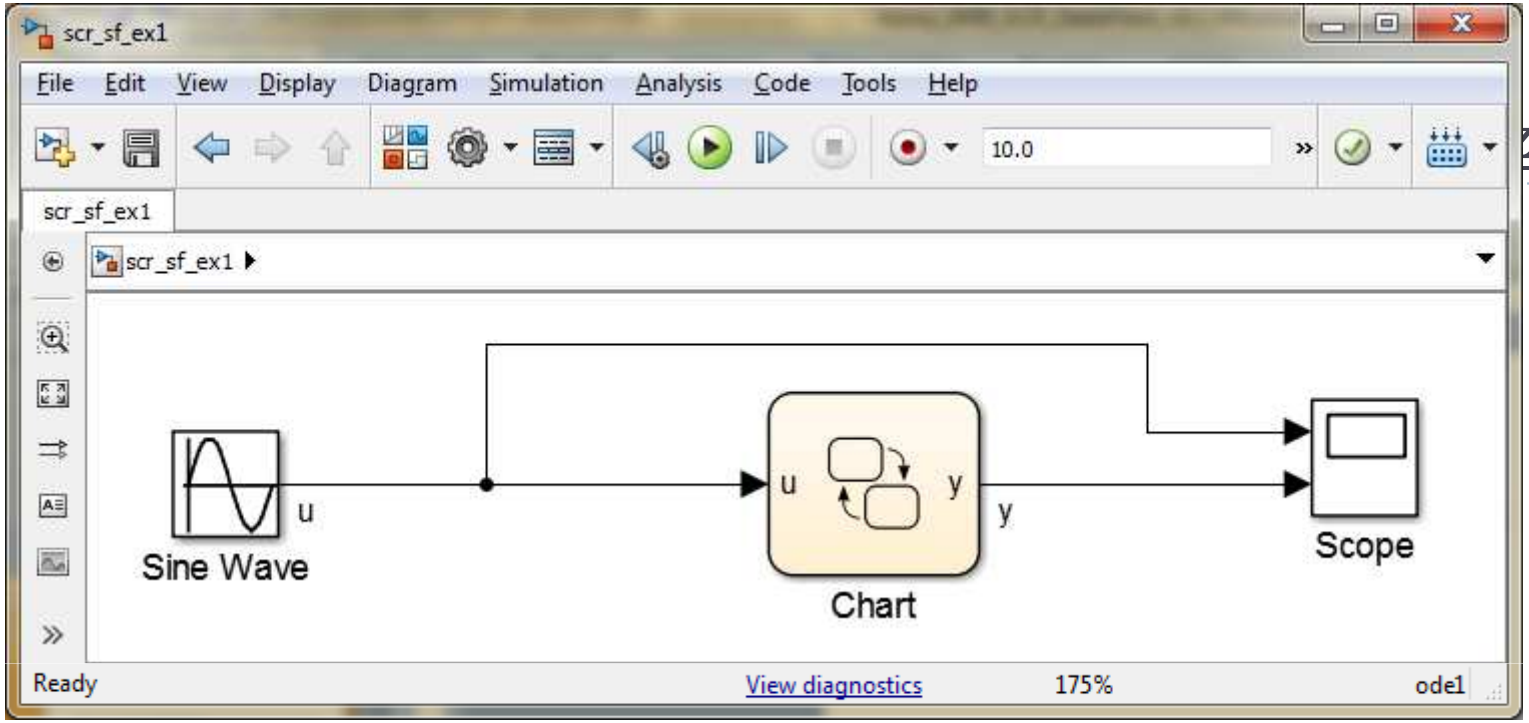


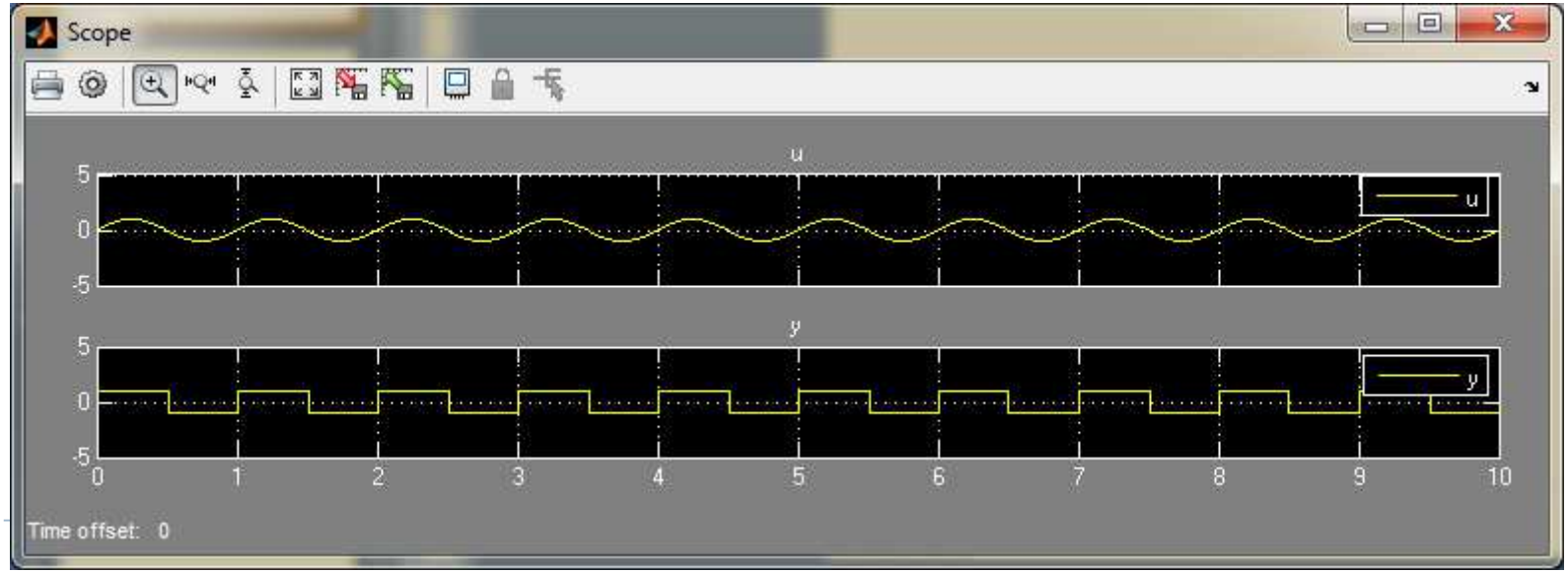
# *Modelowanie maszyny stanowej z wykorzystaniem przybornika StateFlow Matlaba/Simulinka*

# *Przykład 1*

*- prosty model -*

# zykład 1

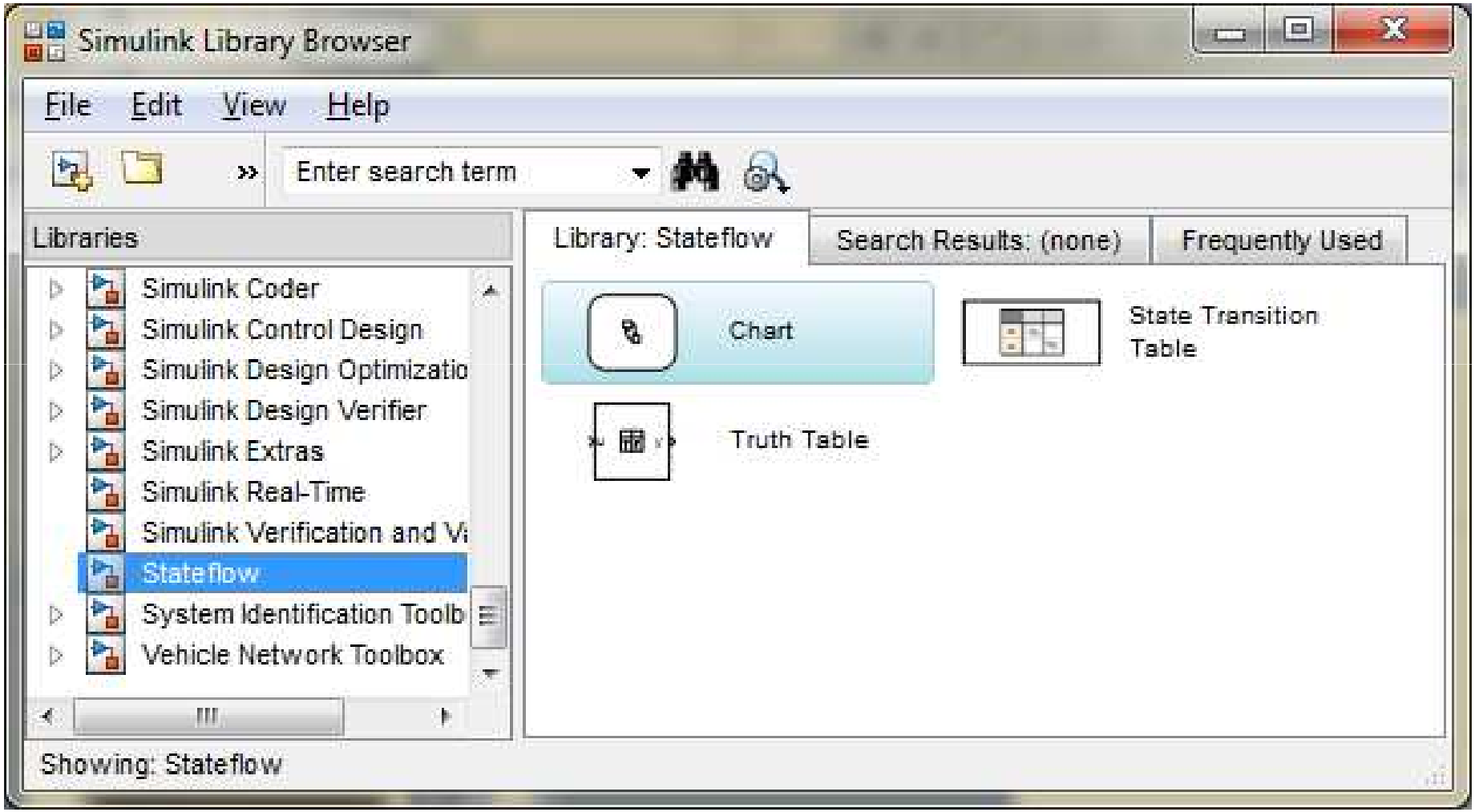


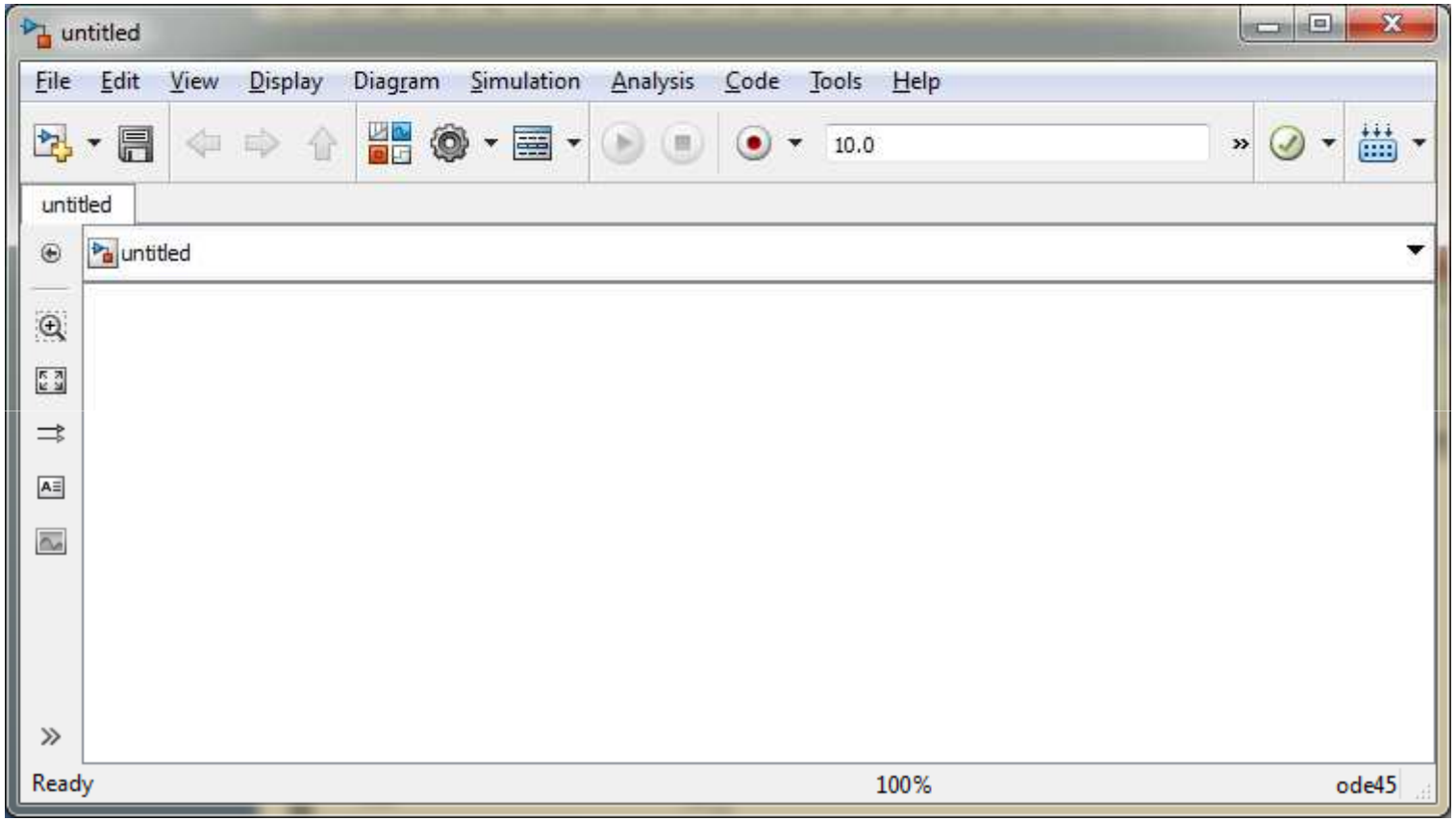
# Przykład 1

- Stan obiektu – ???
- Zdarzenie – ???
- Przejście – ???

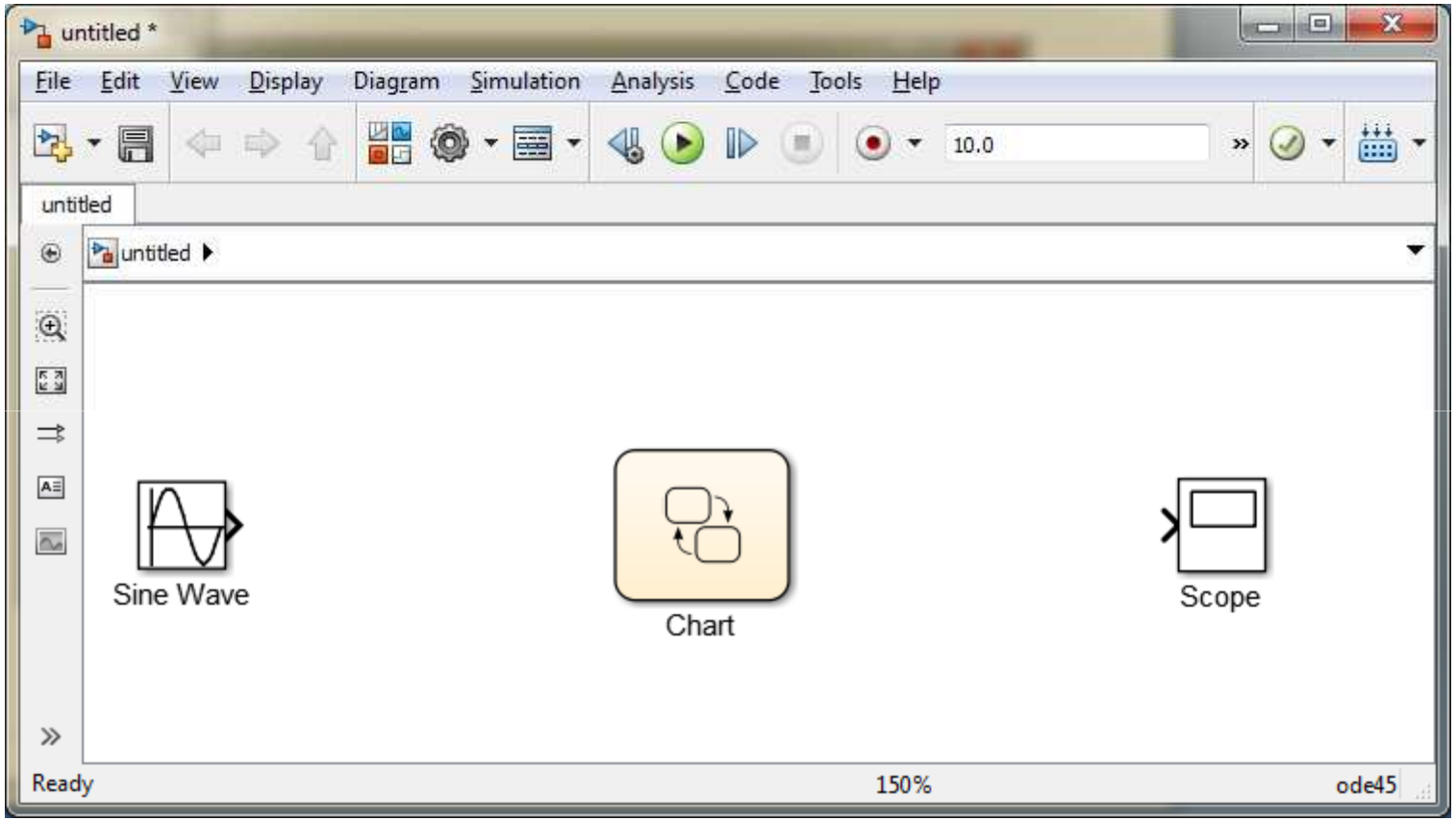
# Przykład 1



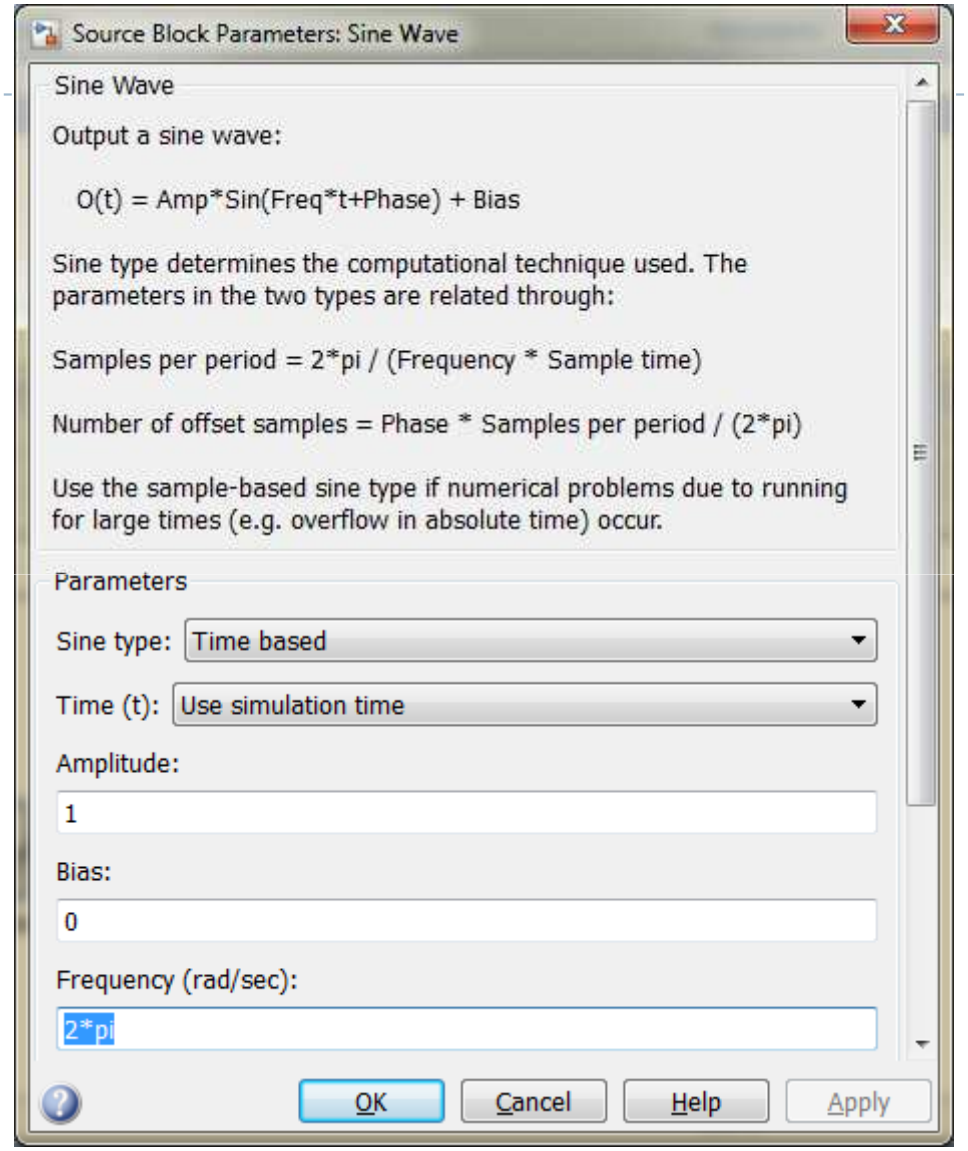
# Przykład 1



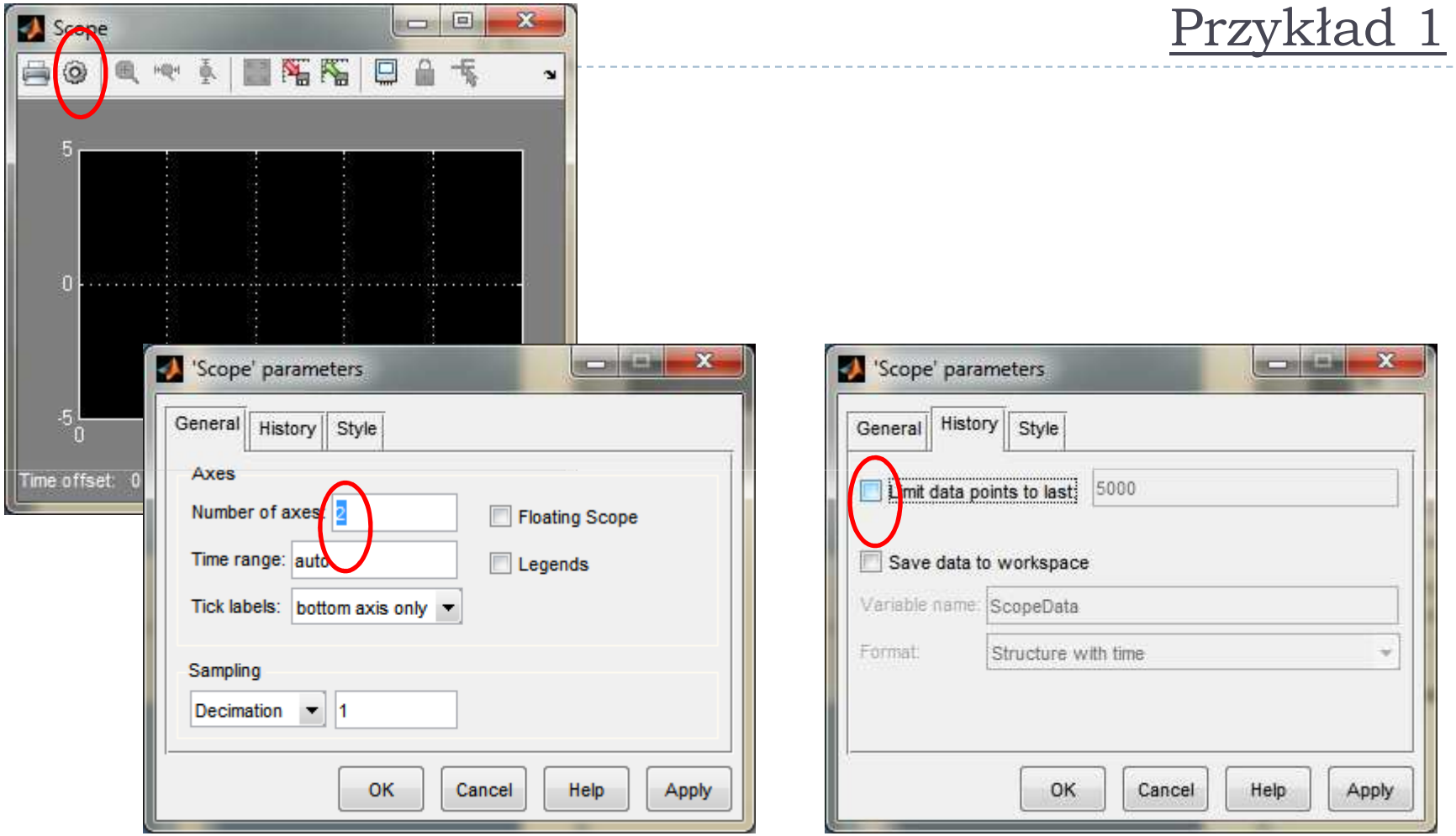
# Przykład 1



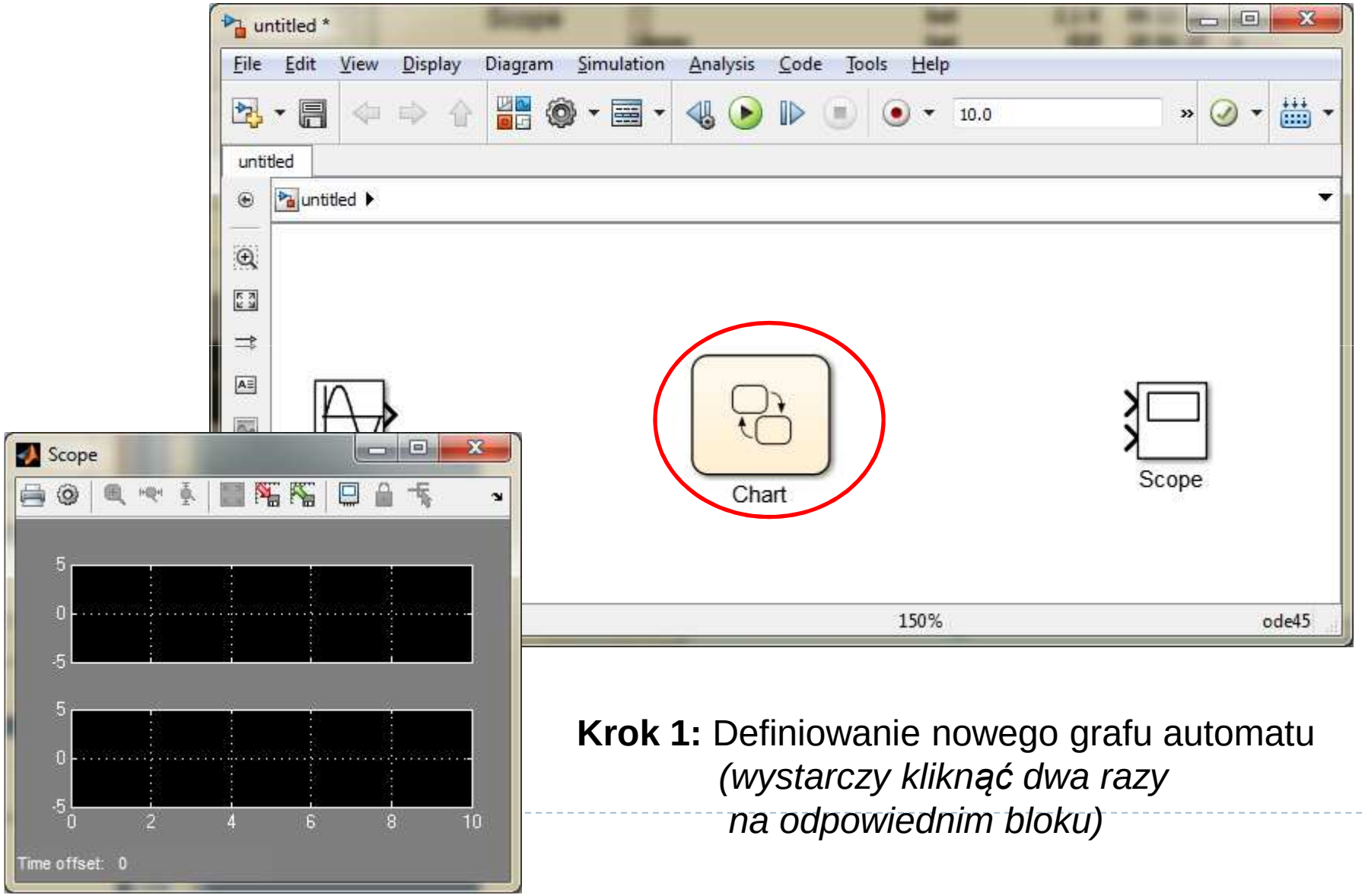
# Przykład 1



# Przykład 1

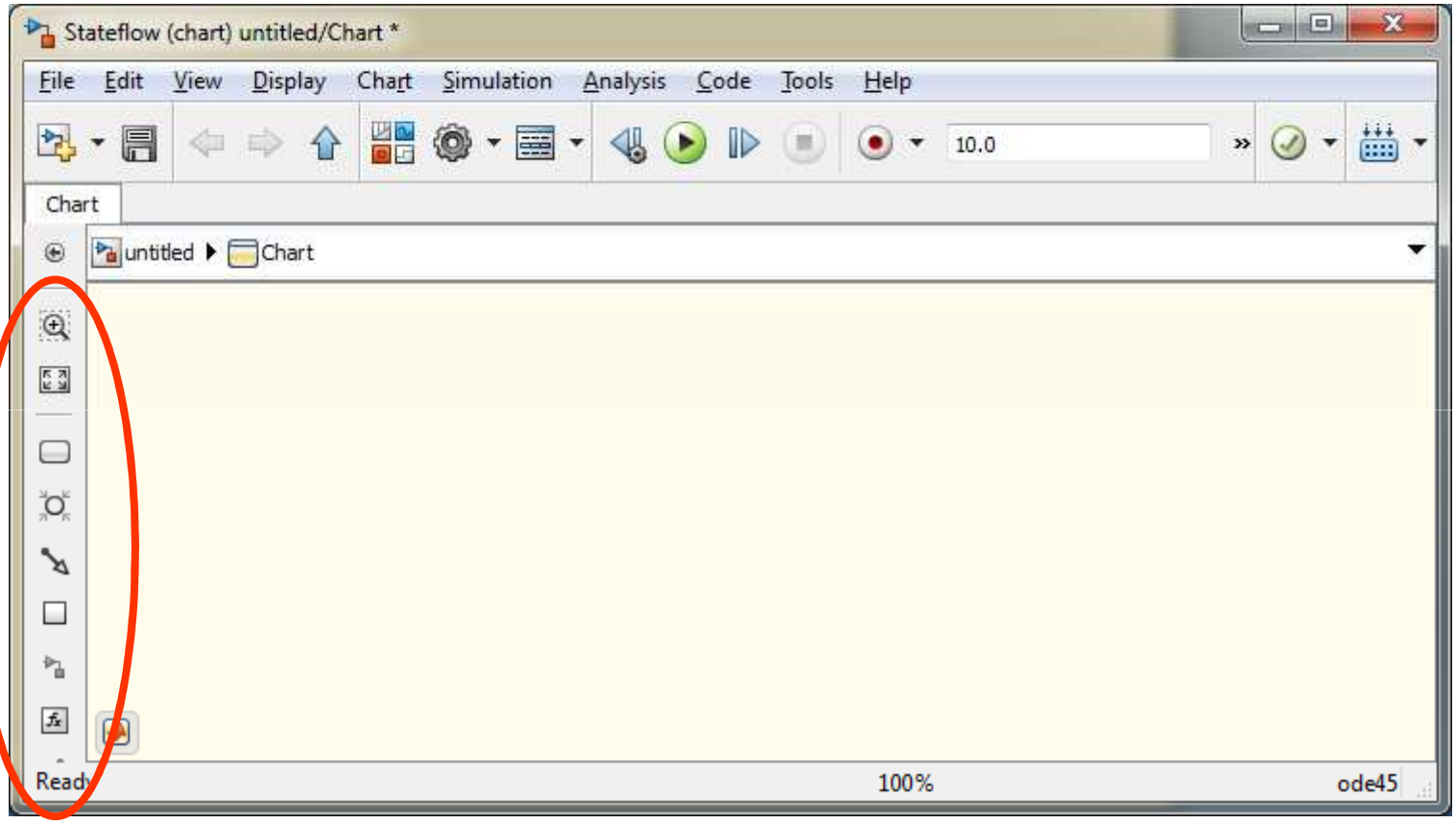


# Przykład 1



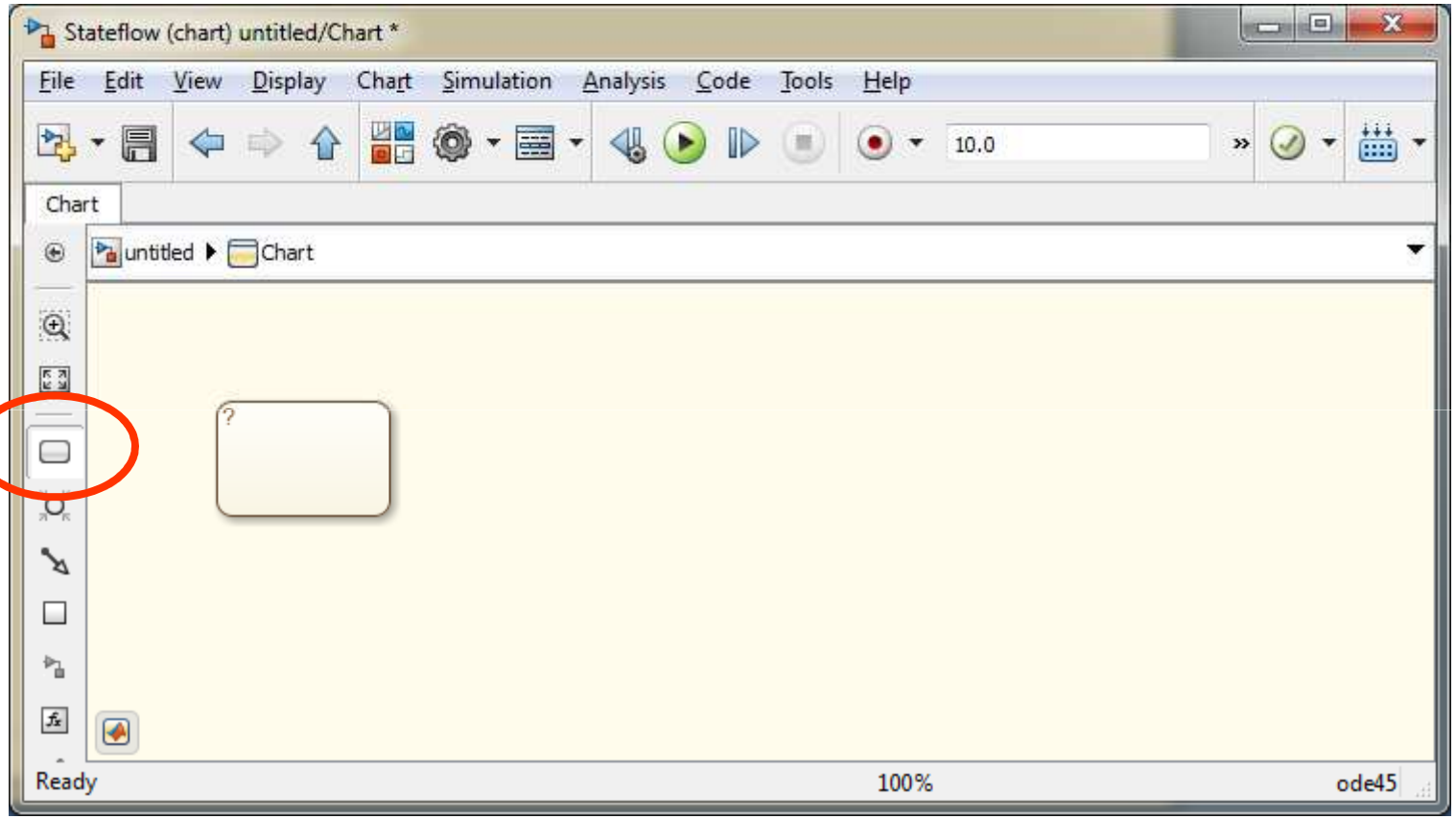

**Krok 1:** Definiowanie nowego grafu automatu
*(wystarczy kliknąć dwa razy*
*na odpowiednim bloku)*

# Przykład 1



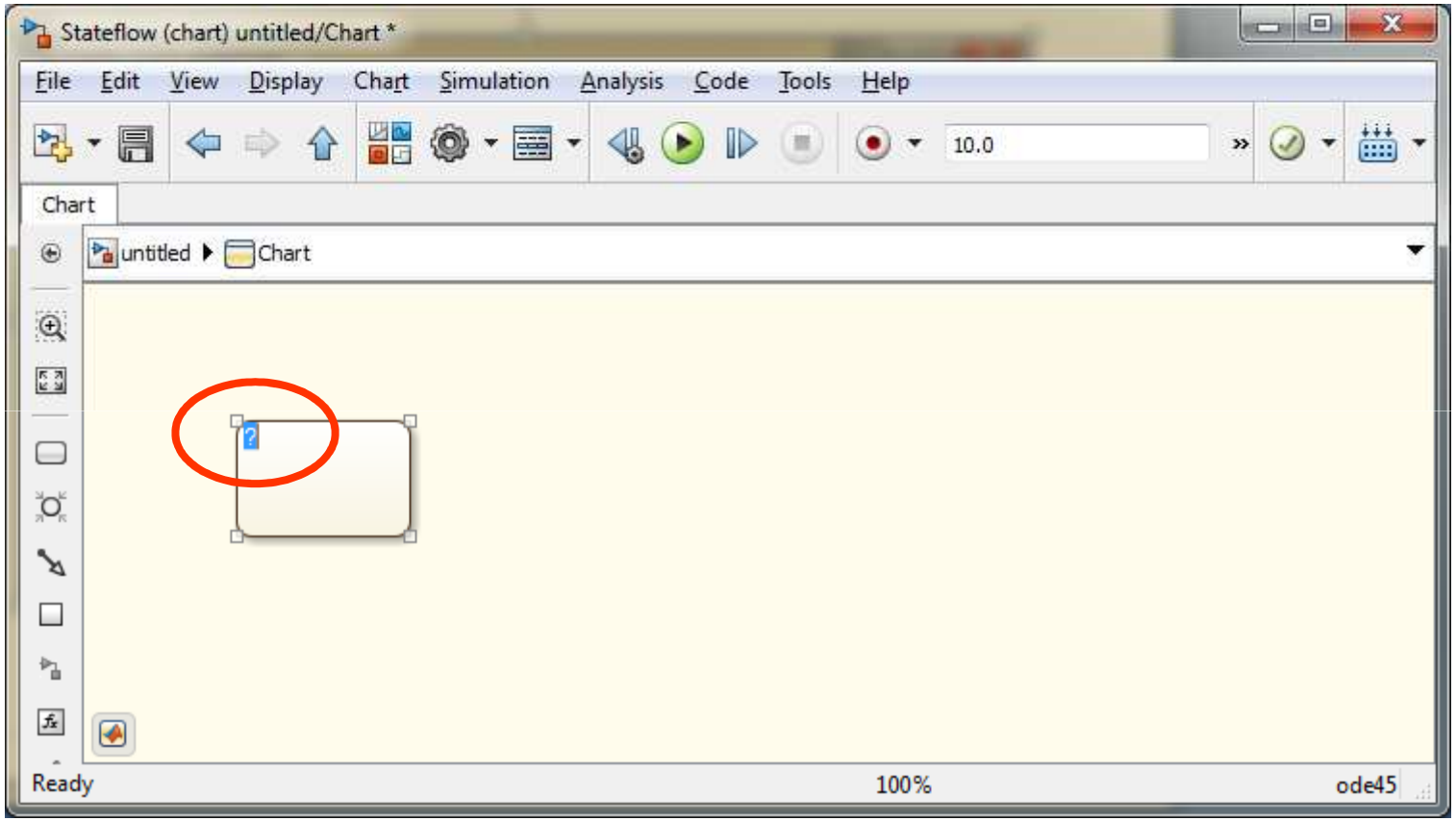

*Wykorzystując dostępne narzędzia edytora można budować grafy odpowiednich automatów skończonych*

# Przykład 1



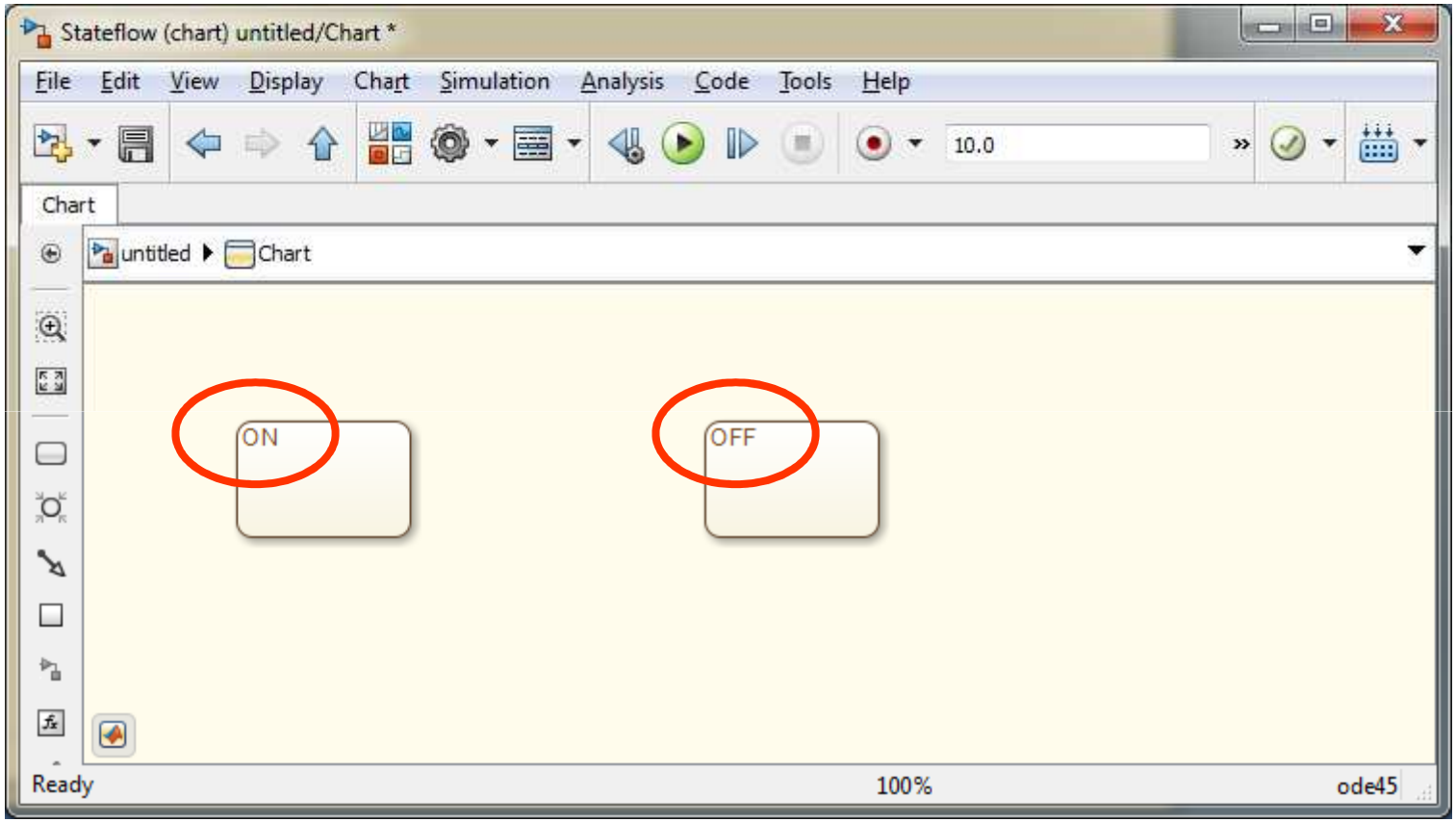

**Krok 2:** Definiowanie nowego stanu

# Przykład 1



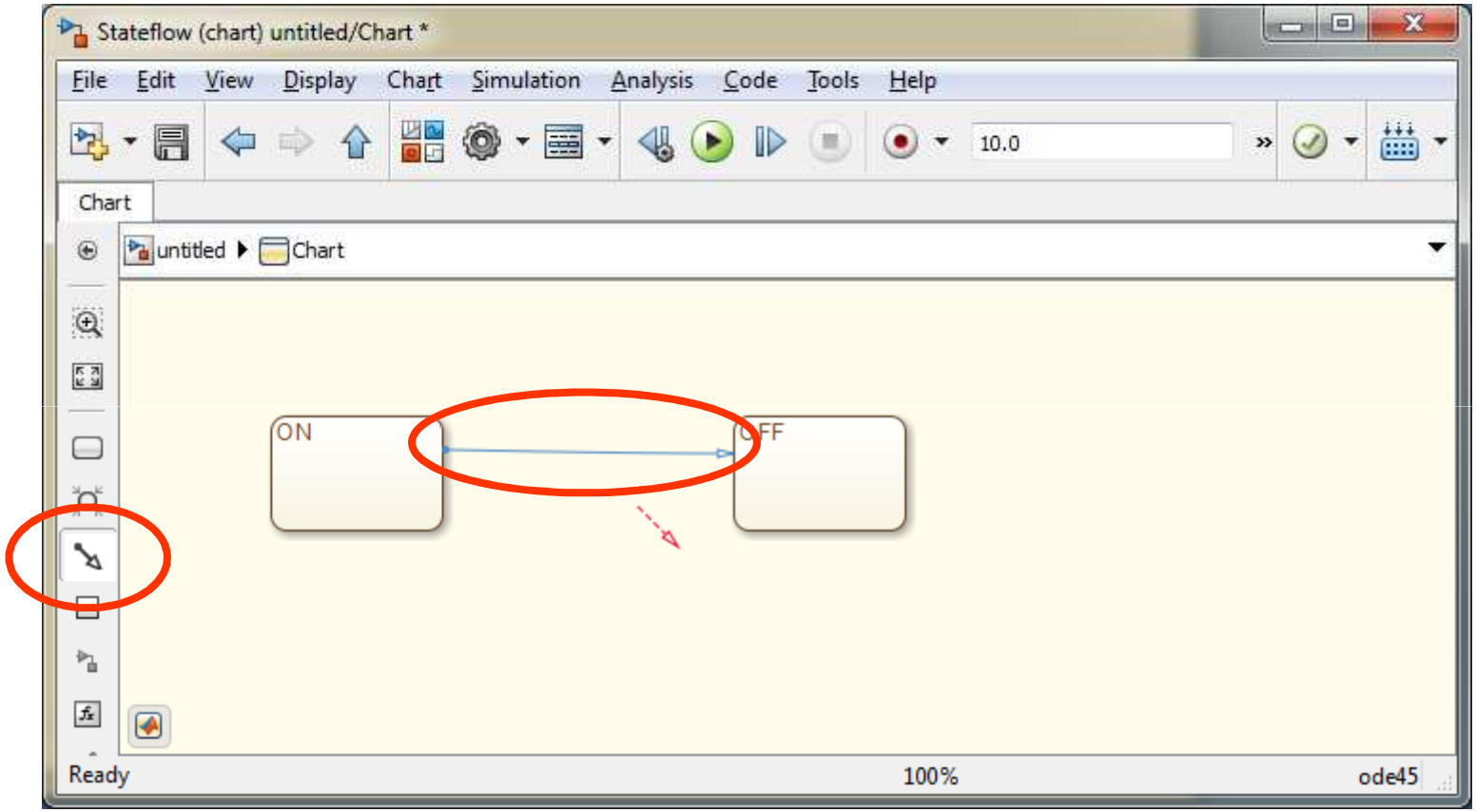

**Krok 2:** Definiowanie nowego stanu – *nadanie odpowiedniej nazwy powiązanej np. z działaniem systemu, które będzie bezpośrednio zdefiniowane w tym stanie*

# Przykład 1



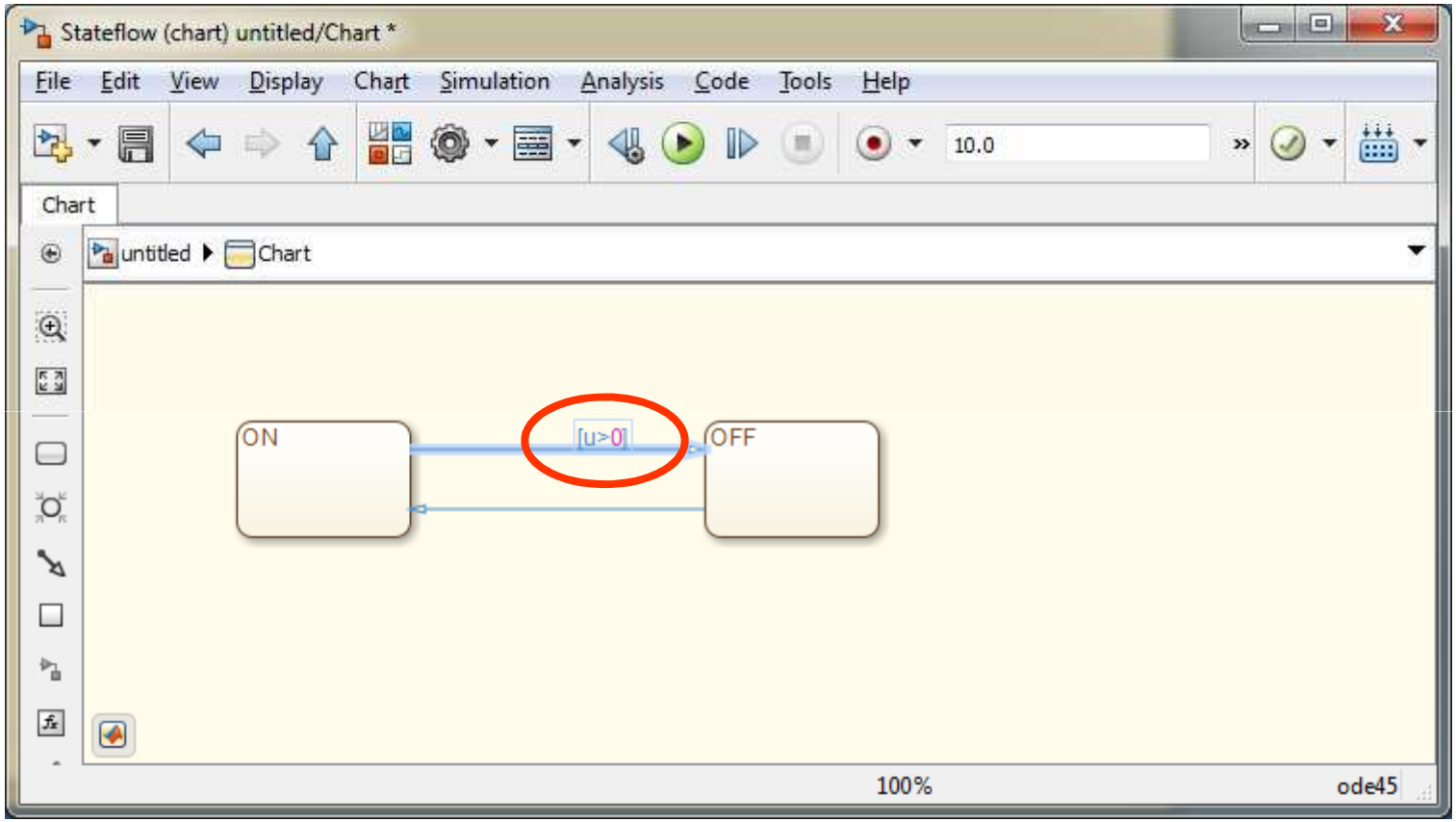

**Krok 2:** Definiowanie nowego stanu – *odpowiednim stanom nadano nazwy ON i OFF, które symbolicznie wskazują działania w każdym z tych stanów*

# Przykład 1



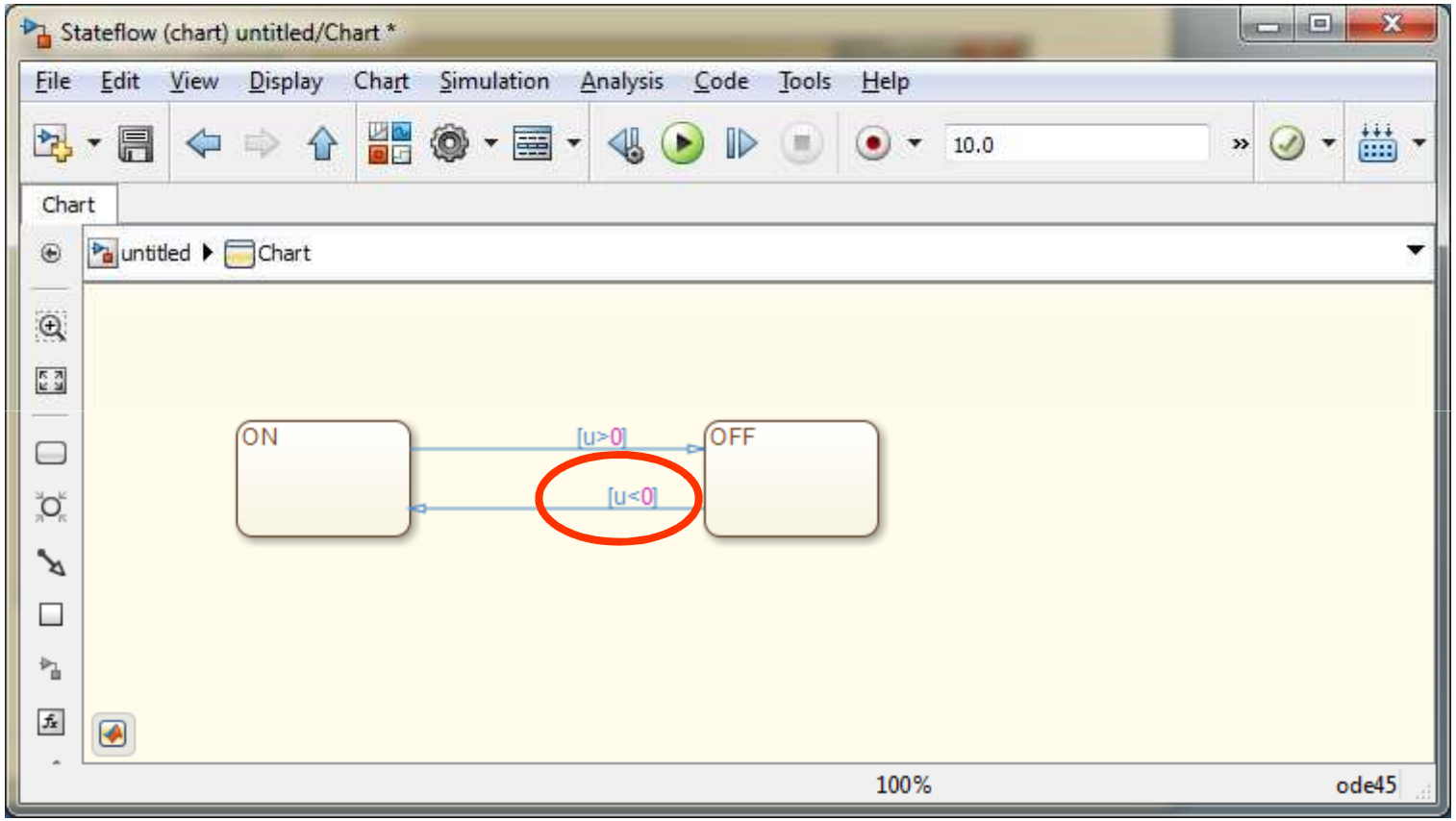

**Krok 2:** Definiowanie przejść pomiędzy stanami – *pierwsze przejście (strzałka) wskazuje przejście ze stanu ON do stanu OFF, należy jeszcze zdefiniować drugie przejście (poprowadzić strzałkę) pomiędzy stanem OFF a stanem ON*

# Przykład 1



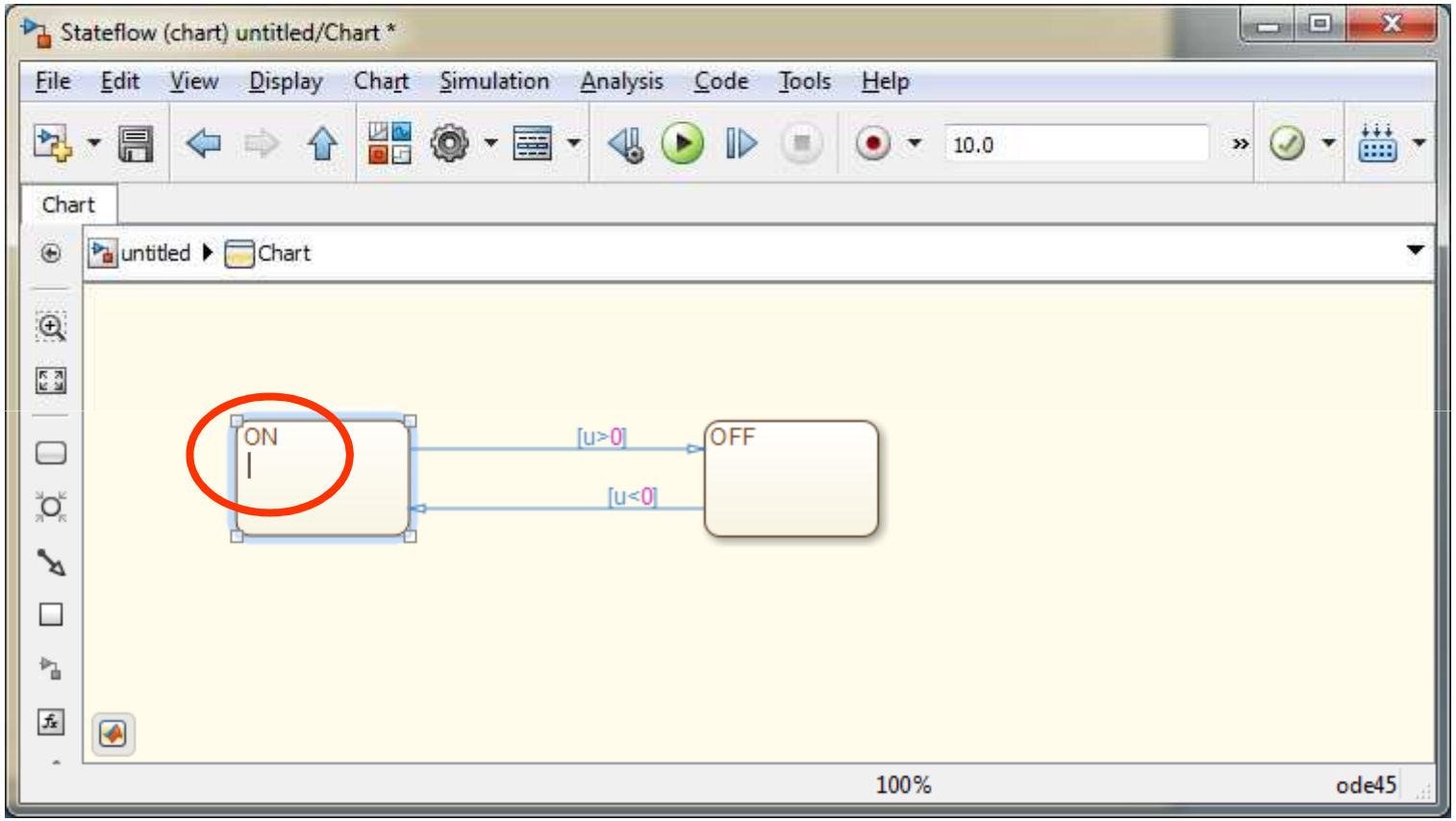

**Krok 3:** Definiowanie warunków przejść pomiędzy stanami – *klikając bezpośrednio na dane przejście można bezpośrednio zdefiniować warunek (wpisać), który je umożliwia (kliknięcie na daną strzałkę); w tym przypadku wpisano u>0 zakłada się że dalej zmienną u zdefiniuje się jako zmienną wejściową*

# Przykład 1



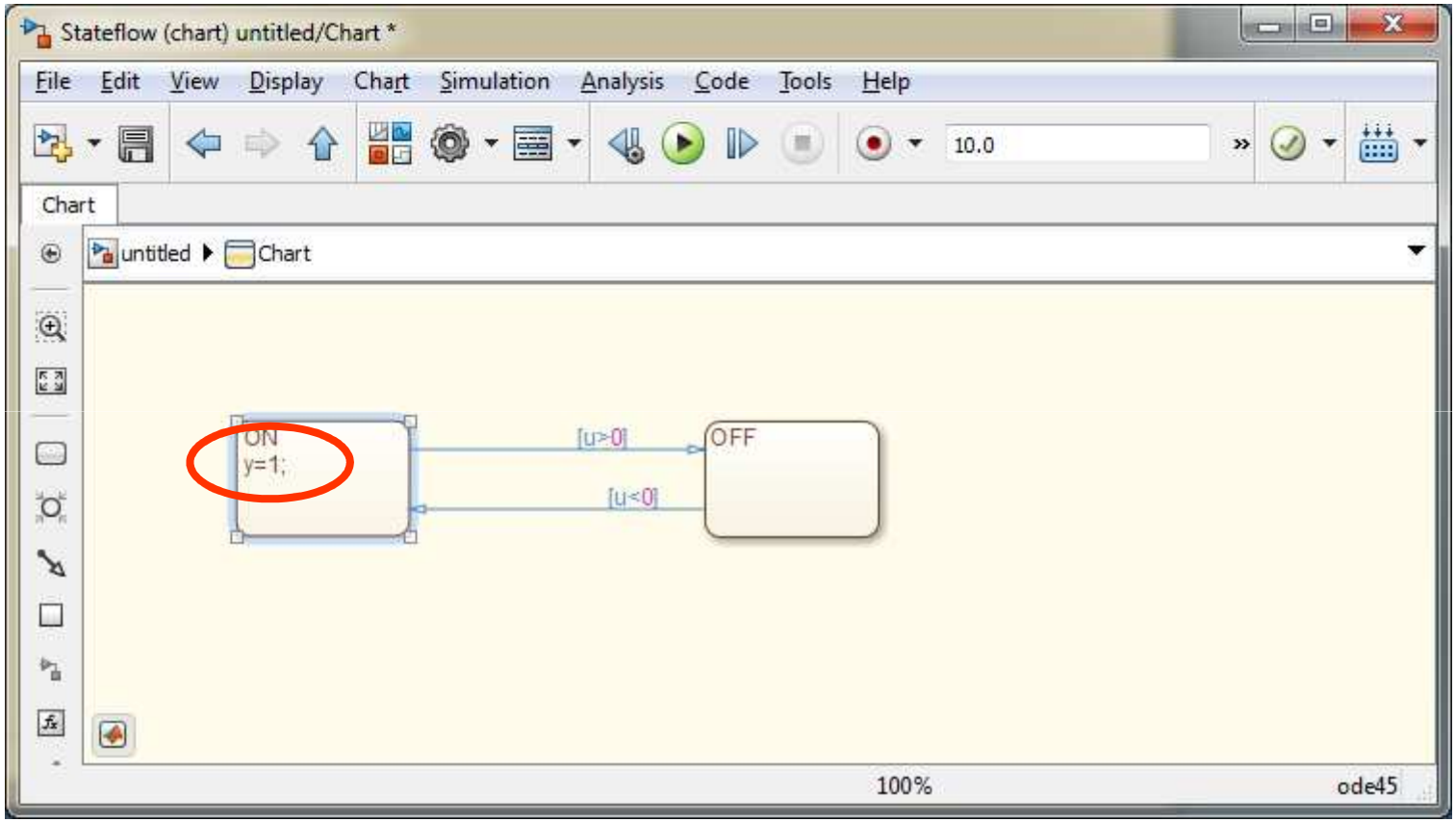

**Krok 3:** Definiowanie warunków przejść pomiędzy stanami – *klikając bezpośrednio na drugie przejście definiuje się kolejny warunek; w tym przypadku wpisano u<0 również zakłada się że zmienną u zdefiniuje się jako zmienną wejściową*

# Przykład 1



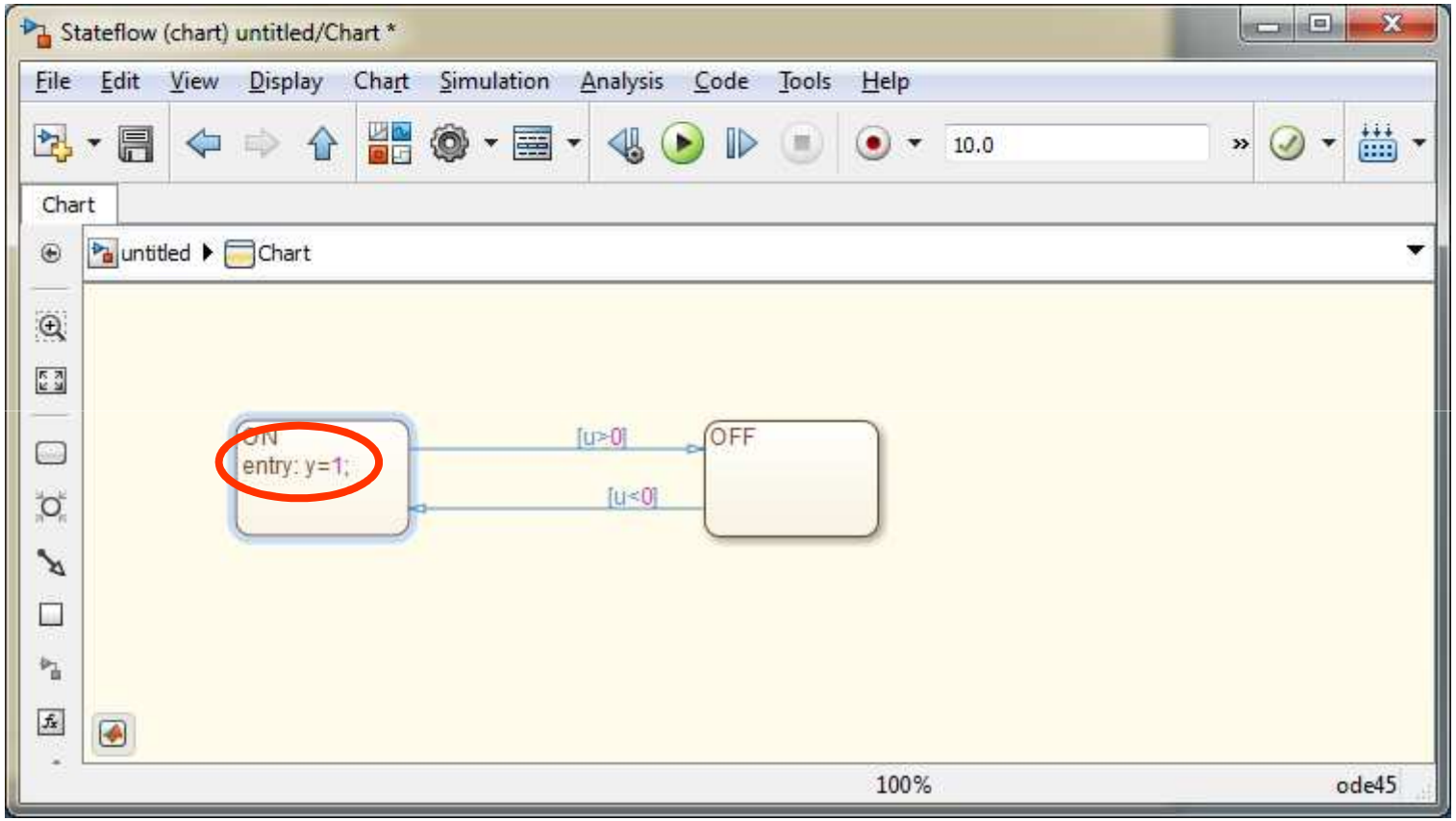

**Krok 4:** Definiowanie stanu – *ustawiając kursor za nazwą stanu i wciskając enter, można zdefiniować zestaw działań w danym stanie (powiązań z innymi obiektami)*

# Przykład 1



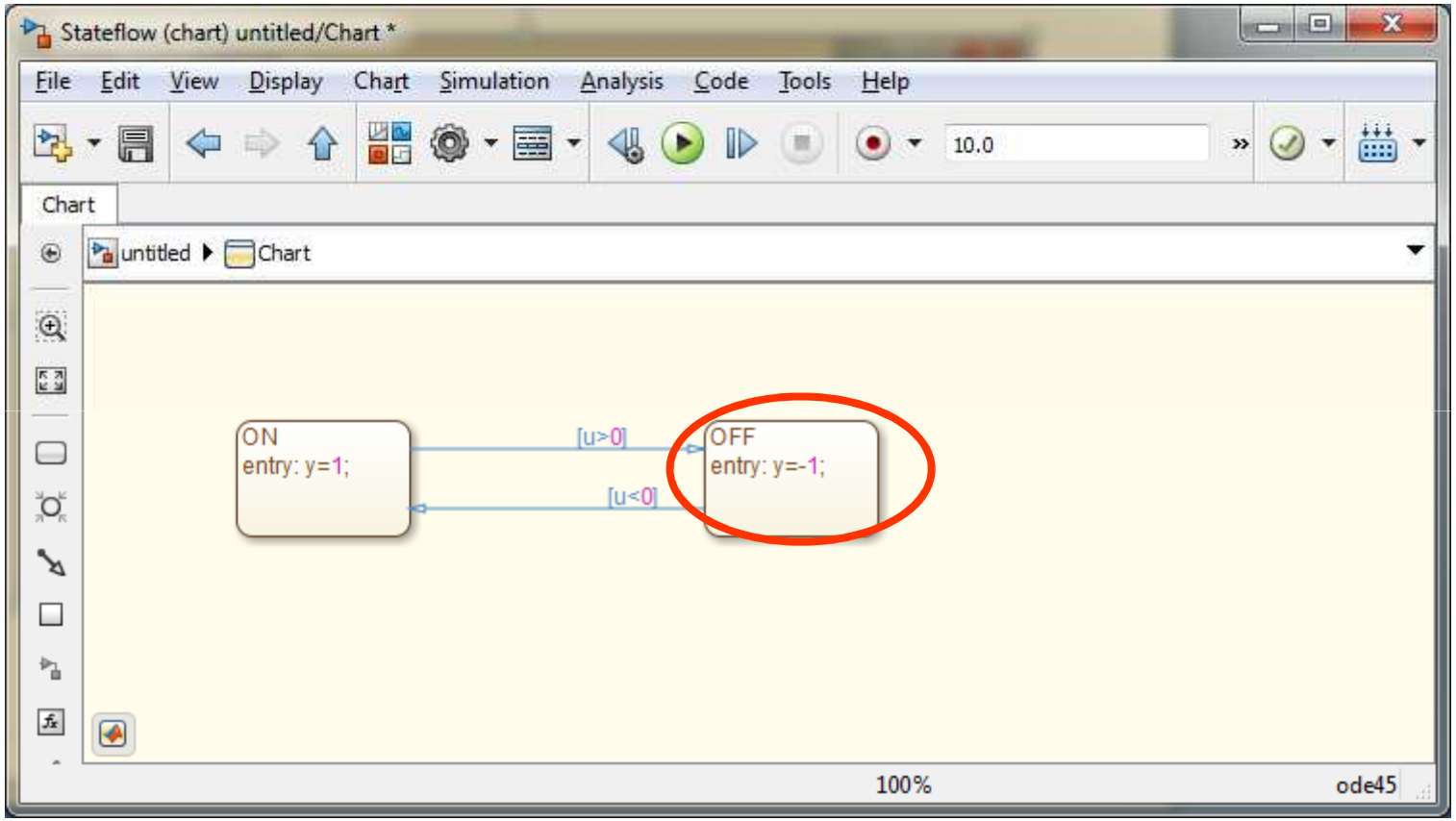

**Krok 4:** Definiowanie stanu ON – *ustawiając kursor za nazwą stanu i wciskając enter, można zdefiniować zestaw działań w danym stanie; w tym przypadku wpisano y=1; (koniec linii kończy się średnikiem), następnie wciśnięto enter, zakłada się że y to zmienna wyjściowa, która zostanie zdefiniowana dalej)*

# Przykład 1



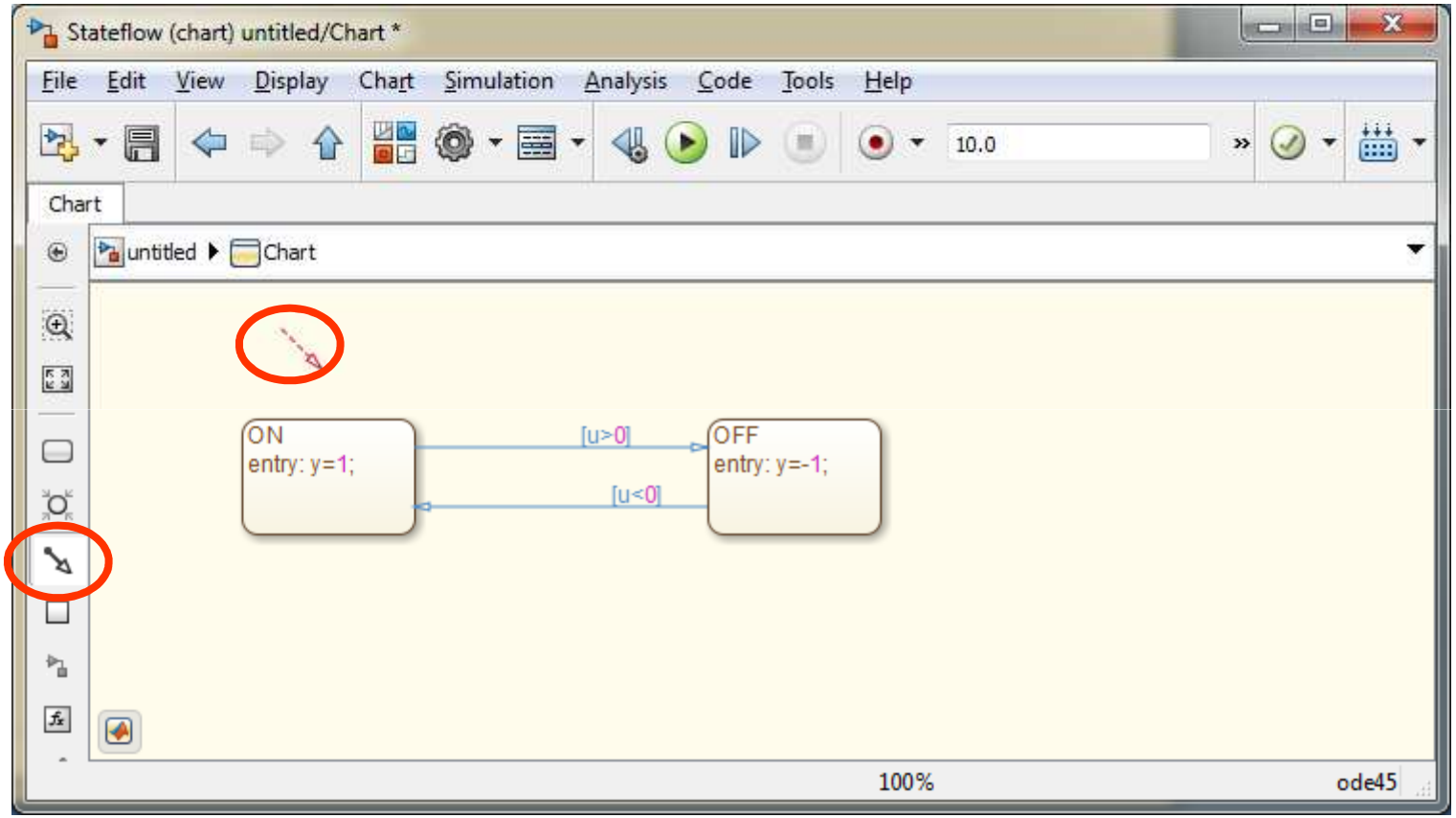

**Krok 4:** Definiowanie stanu ON – *przyciśniecie enter (z poprzedniego kroku) powoduje dodanie wpisu 'entry:' oznaczające że w tym stanie zmienna y przyjmie wartość 1*

# Przykład 1



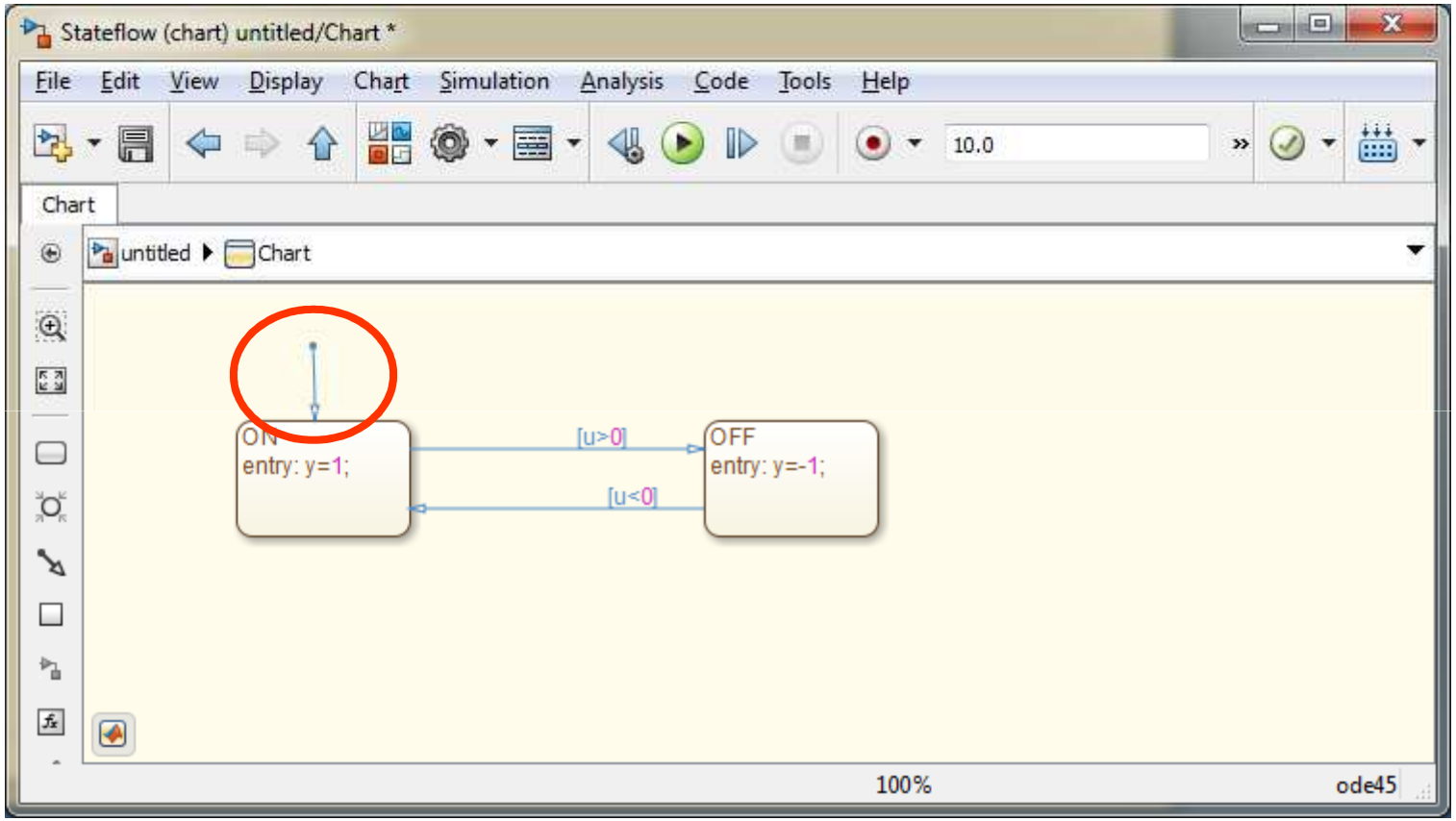

**Krok 5:** Definiowanie stanu OFF – w podobny sposób do poprzedniego kroku, definiuje się stan OFF, w tym przypadku wpisuje się y=-1

# Przykład 1



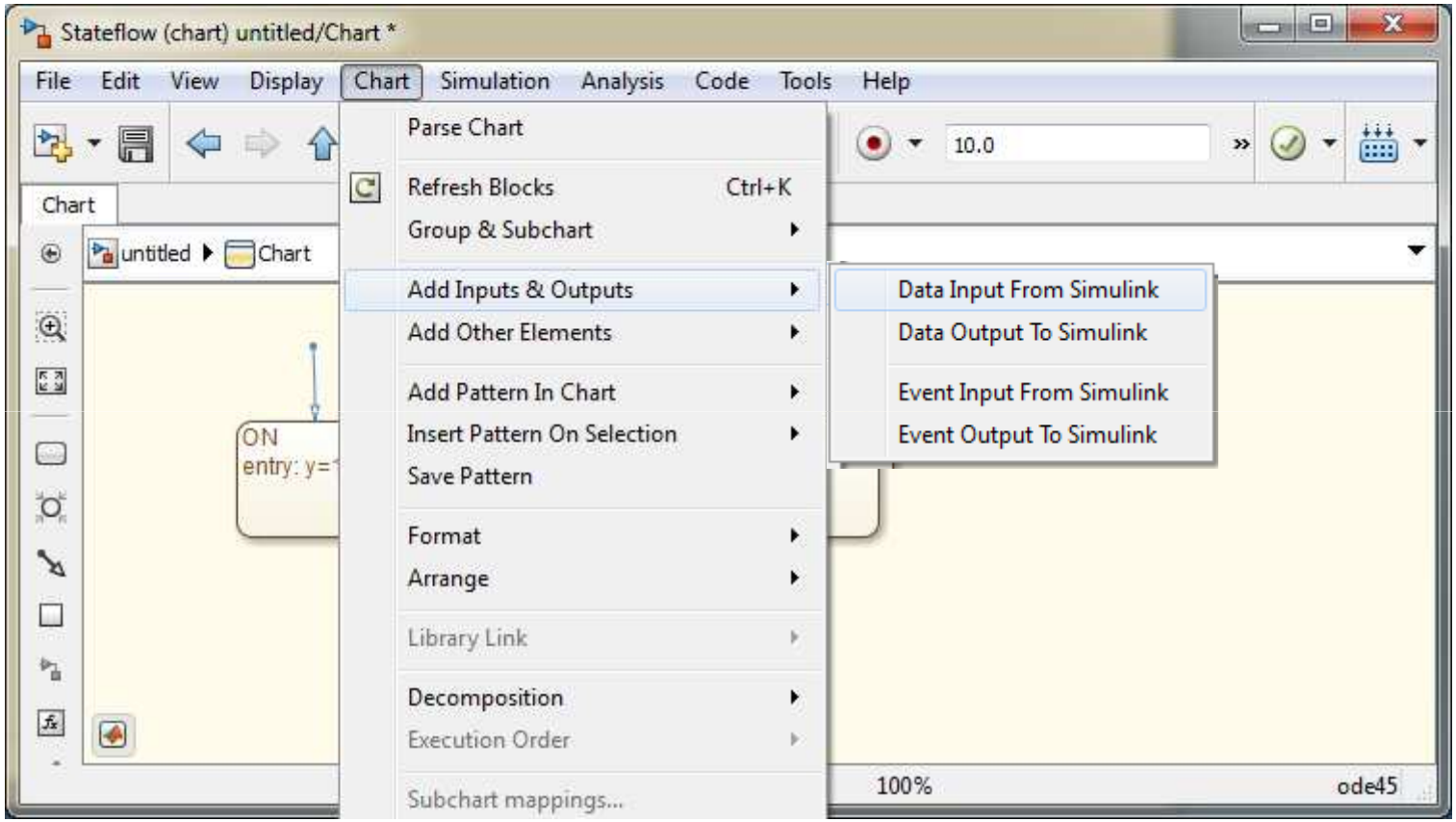

**Krok 6:** Definiowanie stanu początkowego – *zakłada się że stanem początkowym będzie stan ON*

# Przykład 1



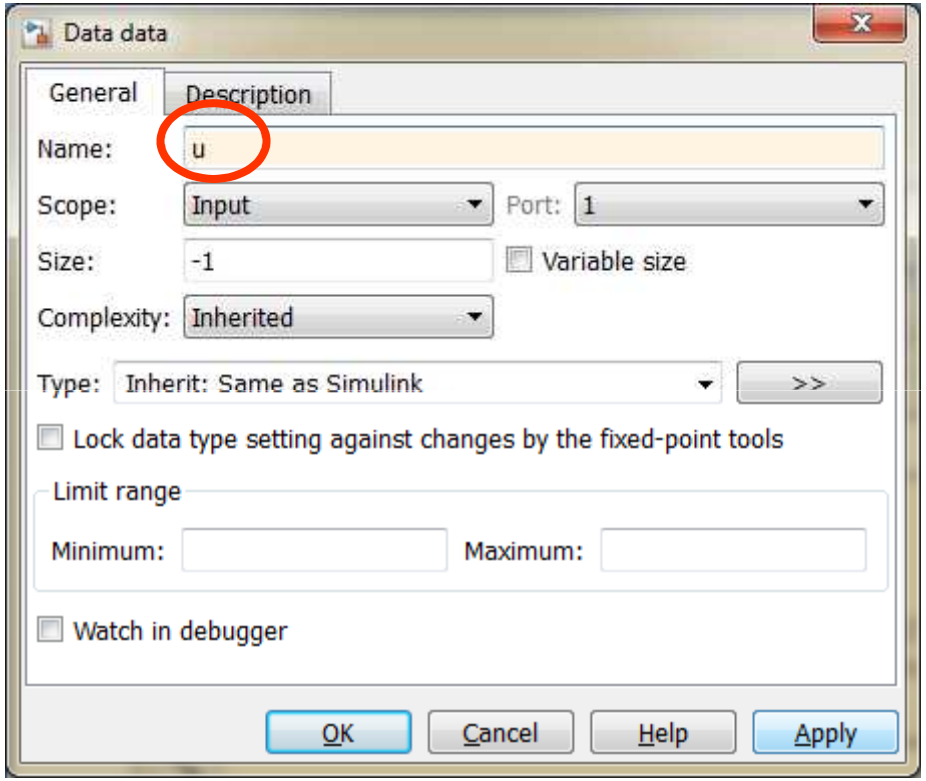

**Krok 6:** Definiowanie stanu początkowego – *zakłada się że stanem początkowym będzie stan ON*

# Przykład 1



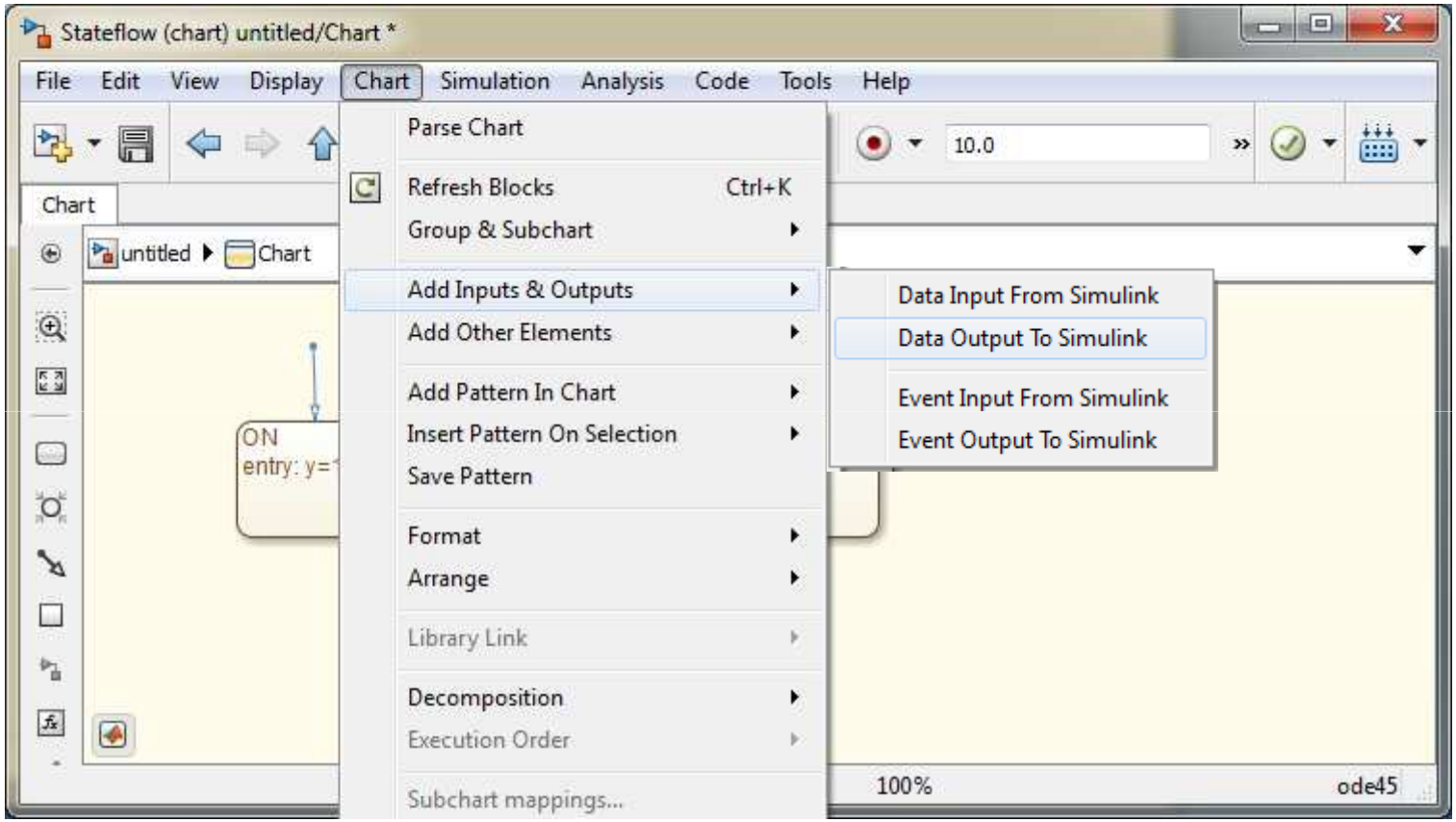

**Krok 7:** Definiowanie „wejścia” do automatu z poziomu Simulinka

# Przykład 1



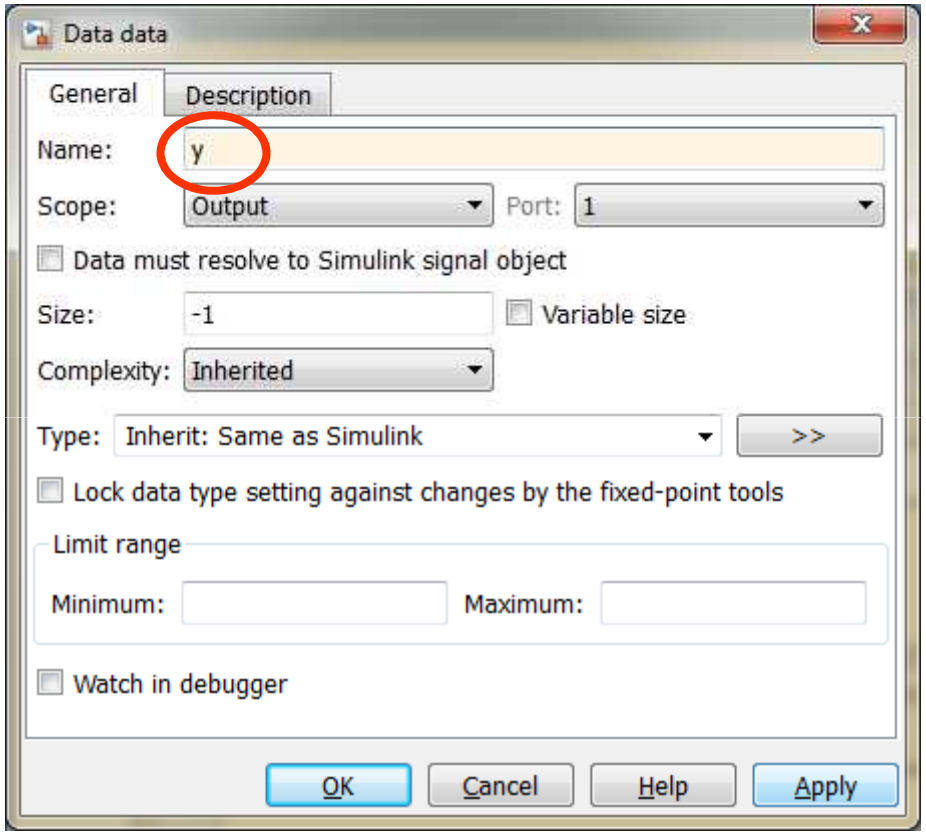

**Krok 7:** Definiowanie „wejścia” do automatu z poziomu Simulinka – *definicja zmiennej u*

# Przykład 1



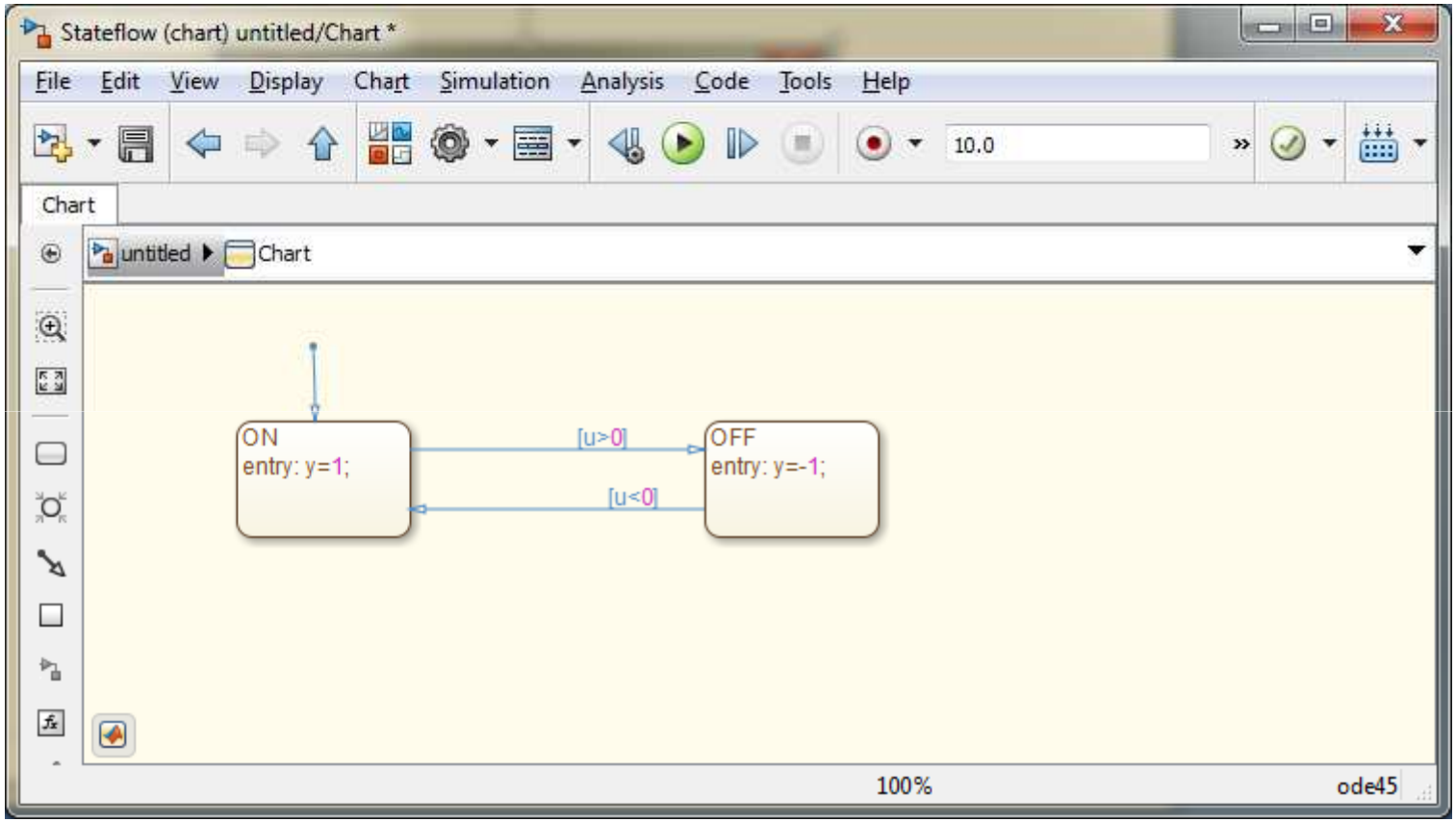

**Krok 8:** Definiowanie „wyjścia” z automatu „do” Simulinka

# Przykład 1



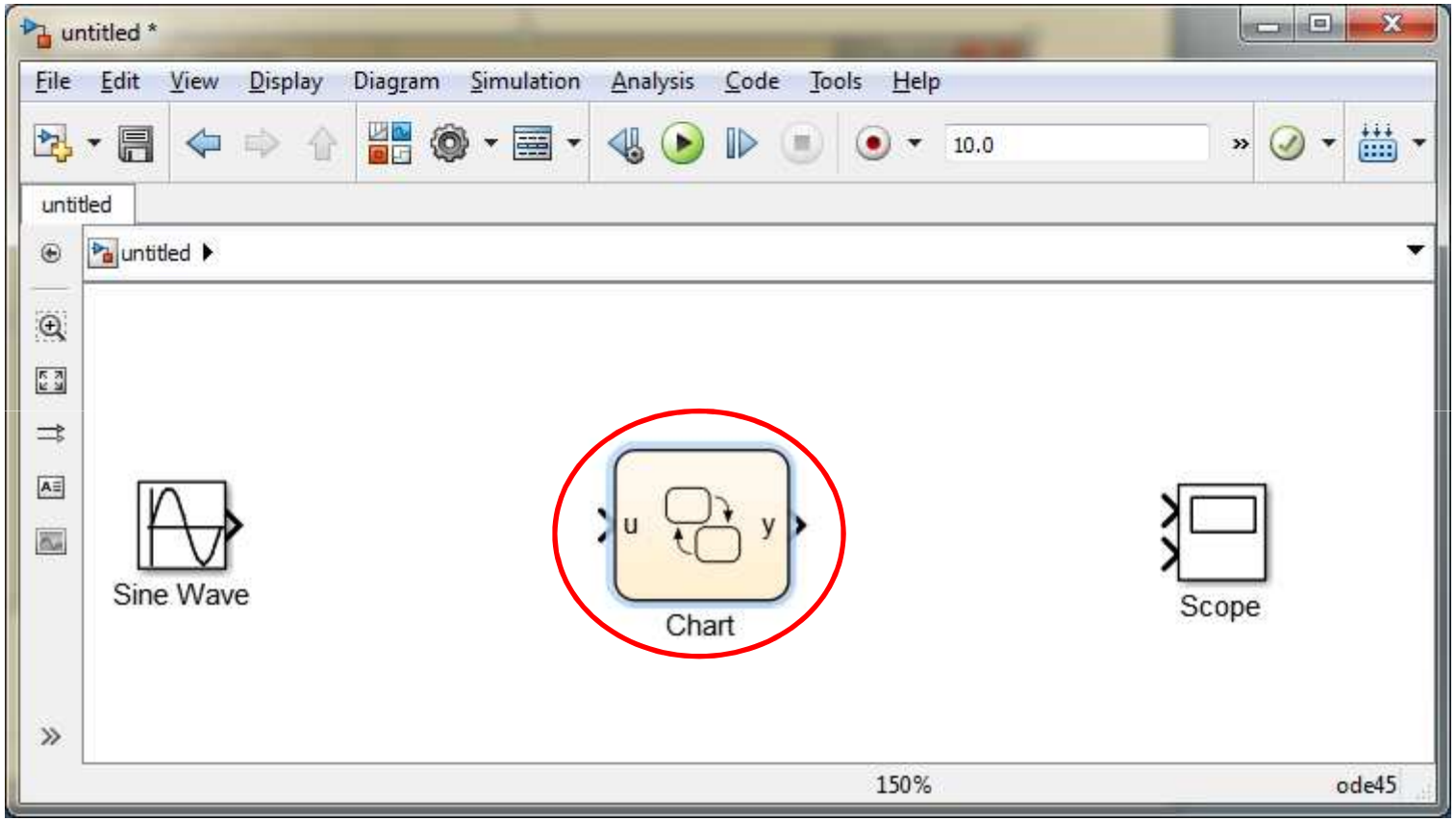

**Krok 8:** Definiowanie „wyjścia” z automatu „do” Simulinka – *definicja zmiennej y*

# Przykład 1



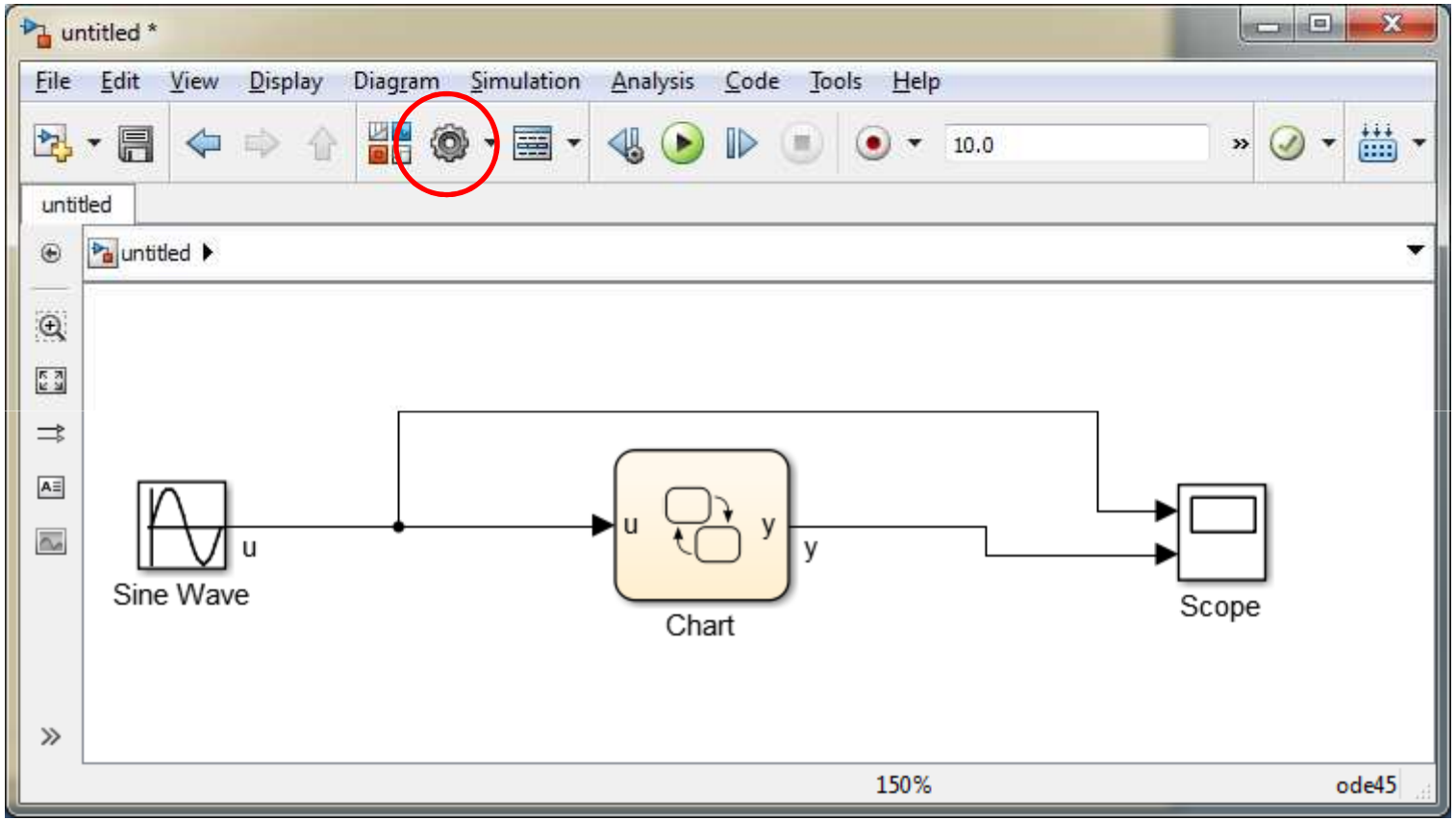

**Krok 9:** Koniec definicji automatu – *wracamy „na poziom" Simulinka*

# Przykład 1



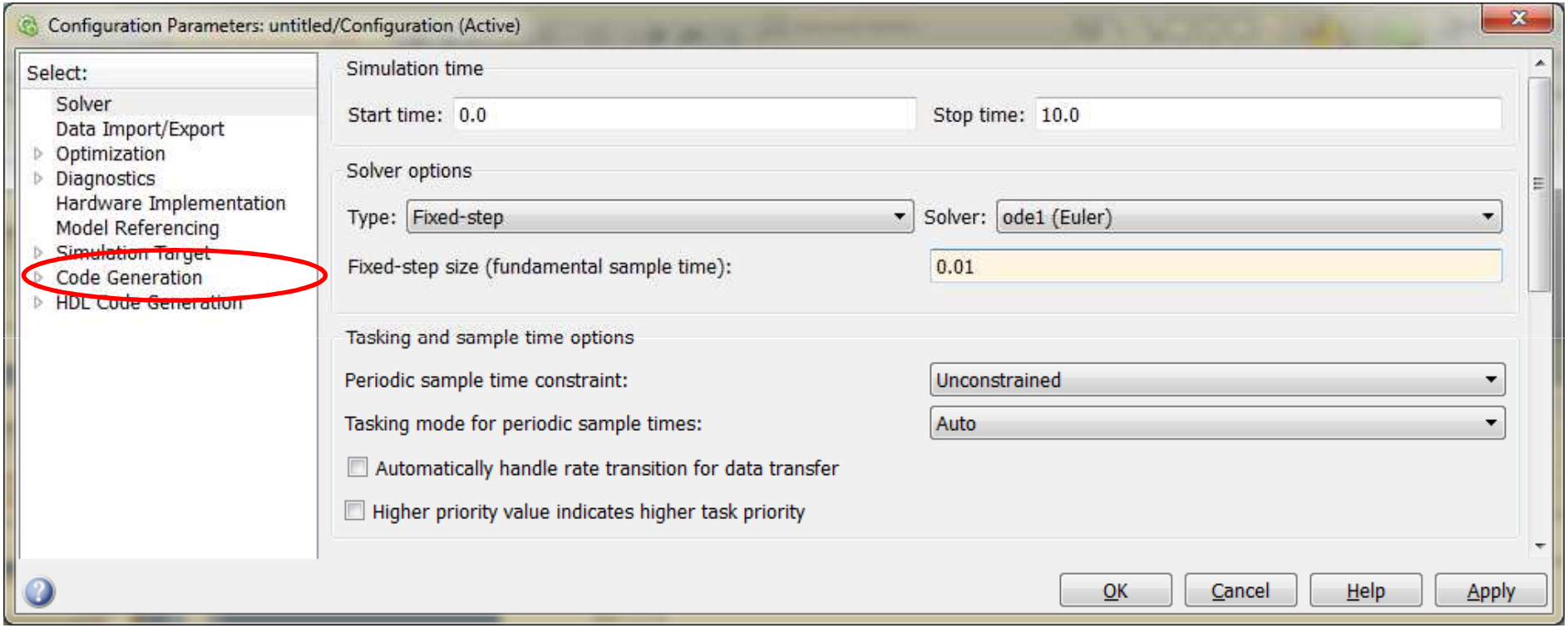

**Krok 10:** Blok grafu automatu „ma” teraz wejście i wyjście, które zostaną wykorzystane do połączeń z pozostałymi blokami Simulinka

# Przykład 1



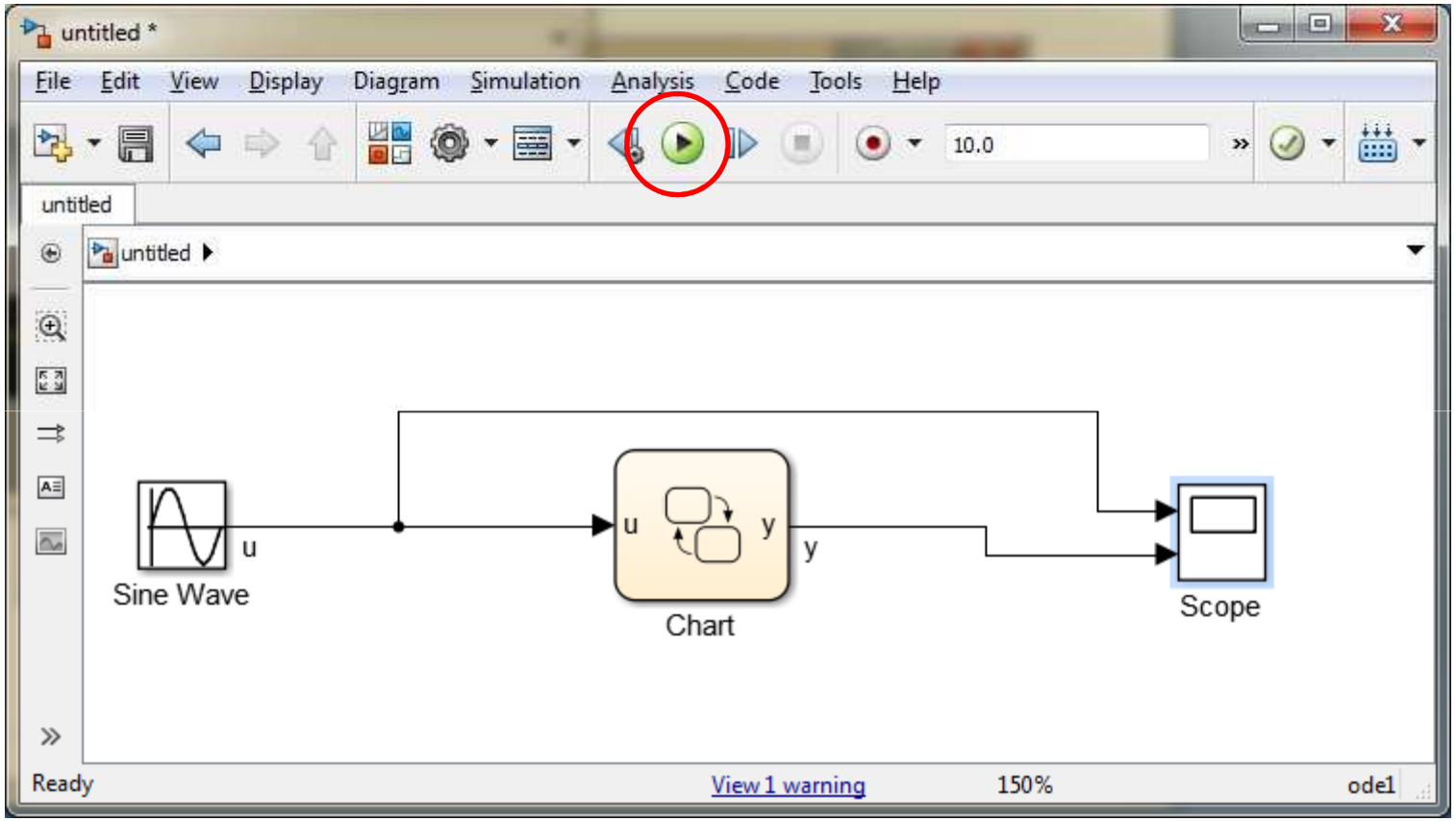

**Krok 11:** Ustawienie parametrów symulacji

# Przykład 1



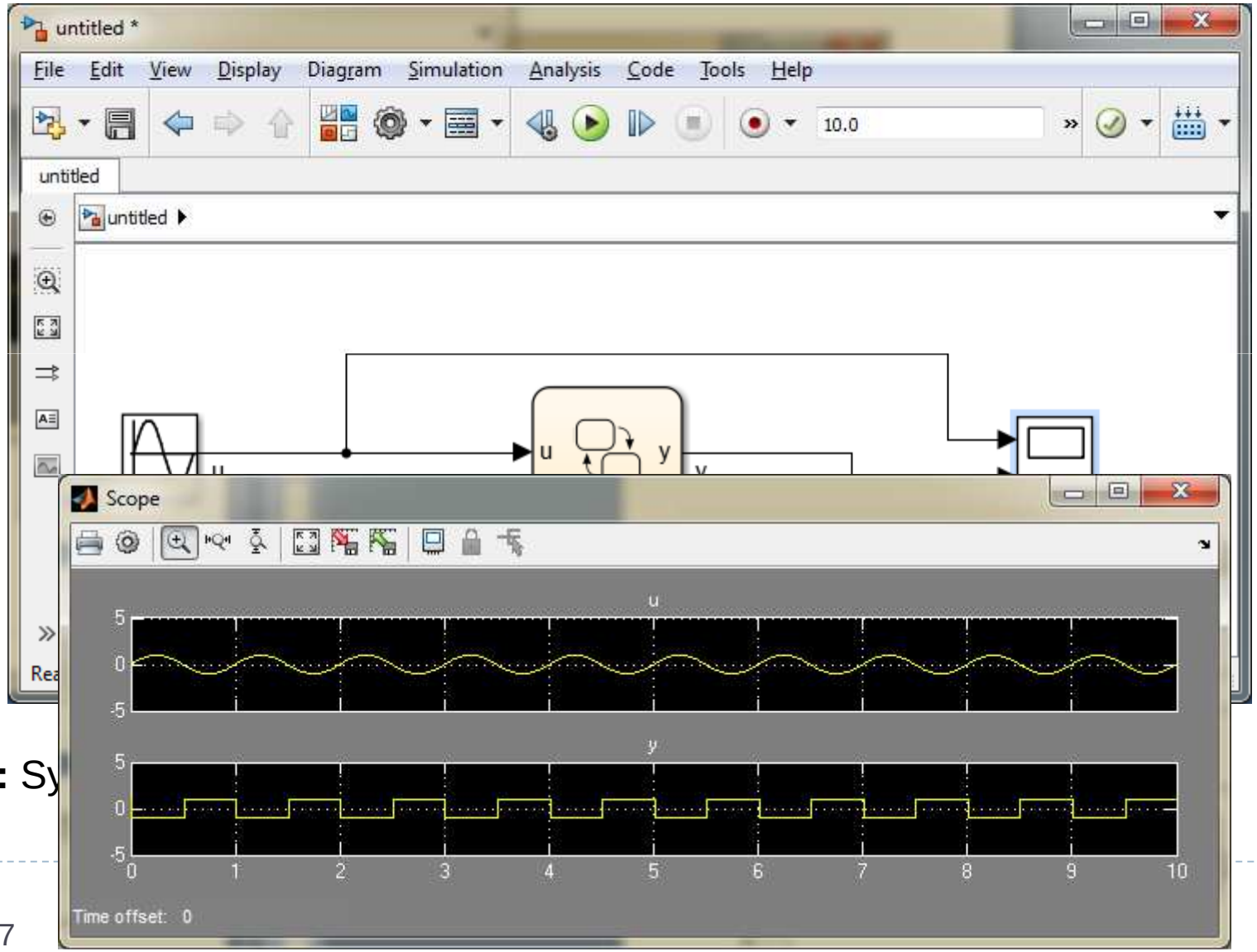

**Krok 11:** Ustawienie parametrów symulacji – *ustawienie odpowiednie solvera, w przypadku symulacji w czasie rzeczywistym, należy wybrać odpowiedni „kompilator" w „Code Generation" np. związany z przybornikiem Simulink Desktop Real-Time*

# Przykład 1




**Krok 12:** Symulacja działania zaprojektowanego automatu i analiza rezultatów

# Przykład 1




**Krok 12:** Sy

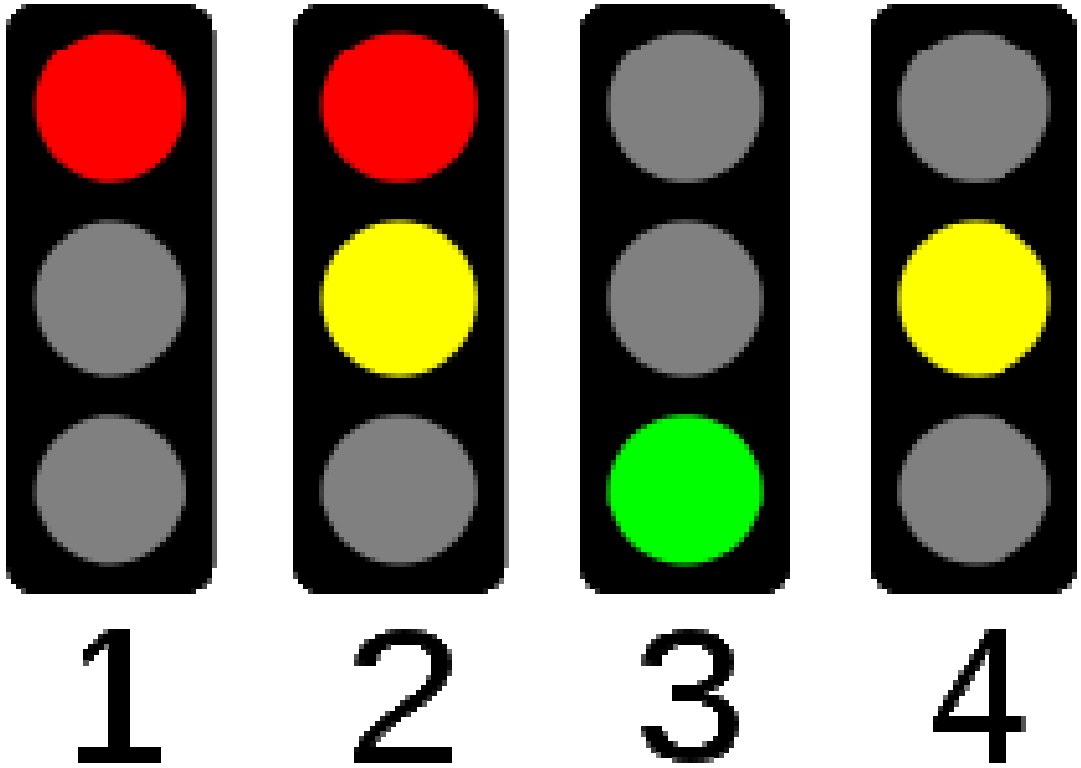
# DEMO

## Symulacja w trybie NORMAL
## Symulacja w czasie rzeczywistym SLDRT

# *Przykład 2*

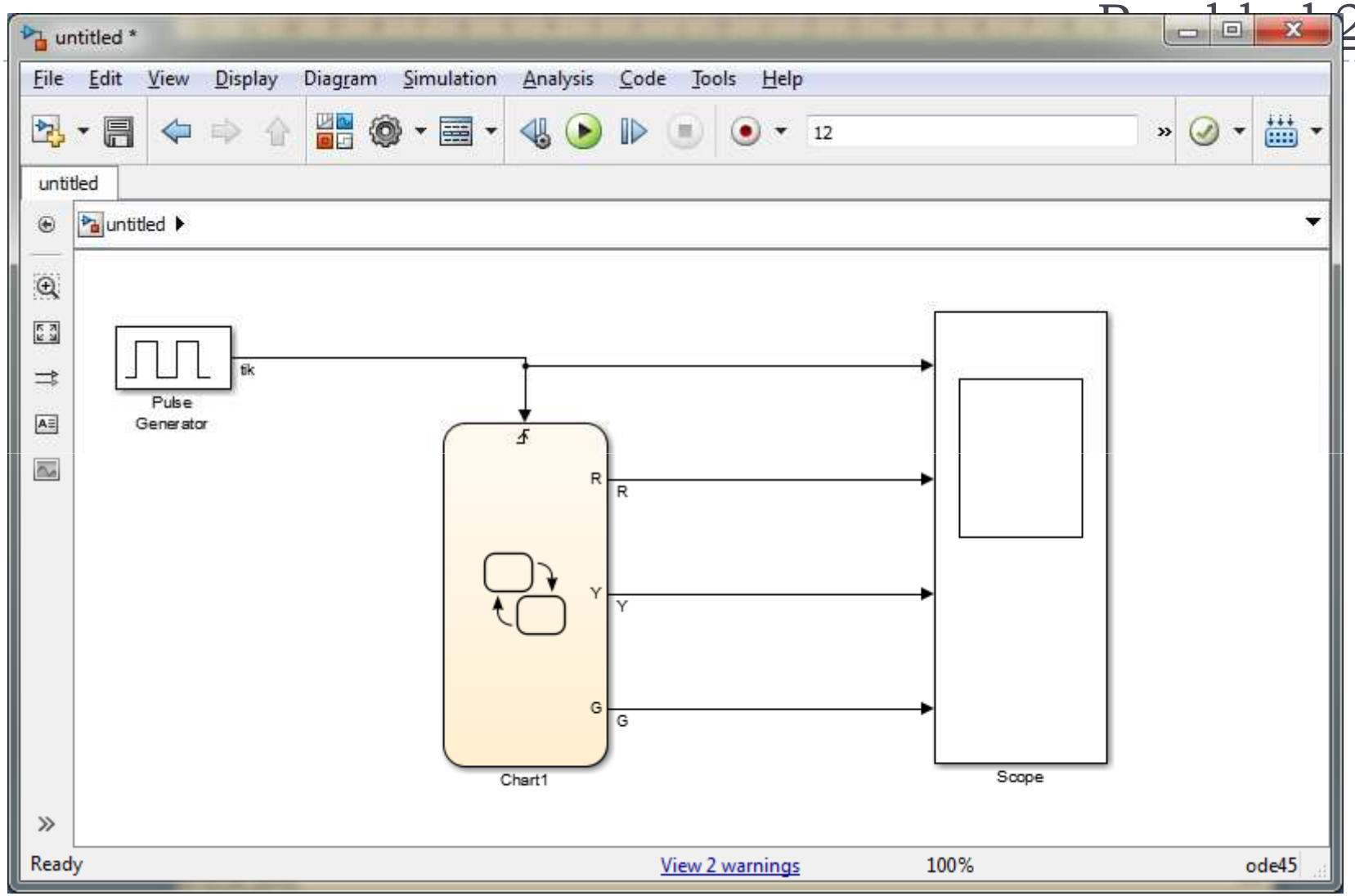
## *- układ sekwencyjny: prosty sygnalizator świetlny-*

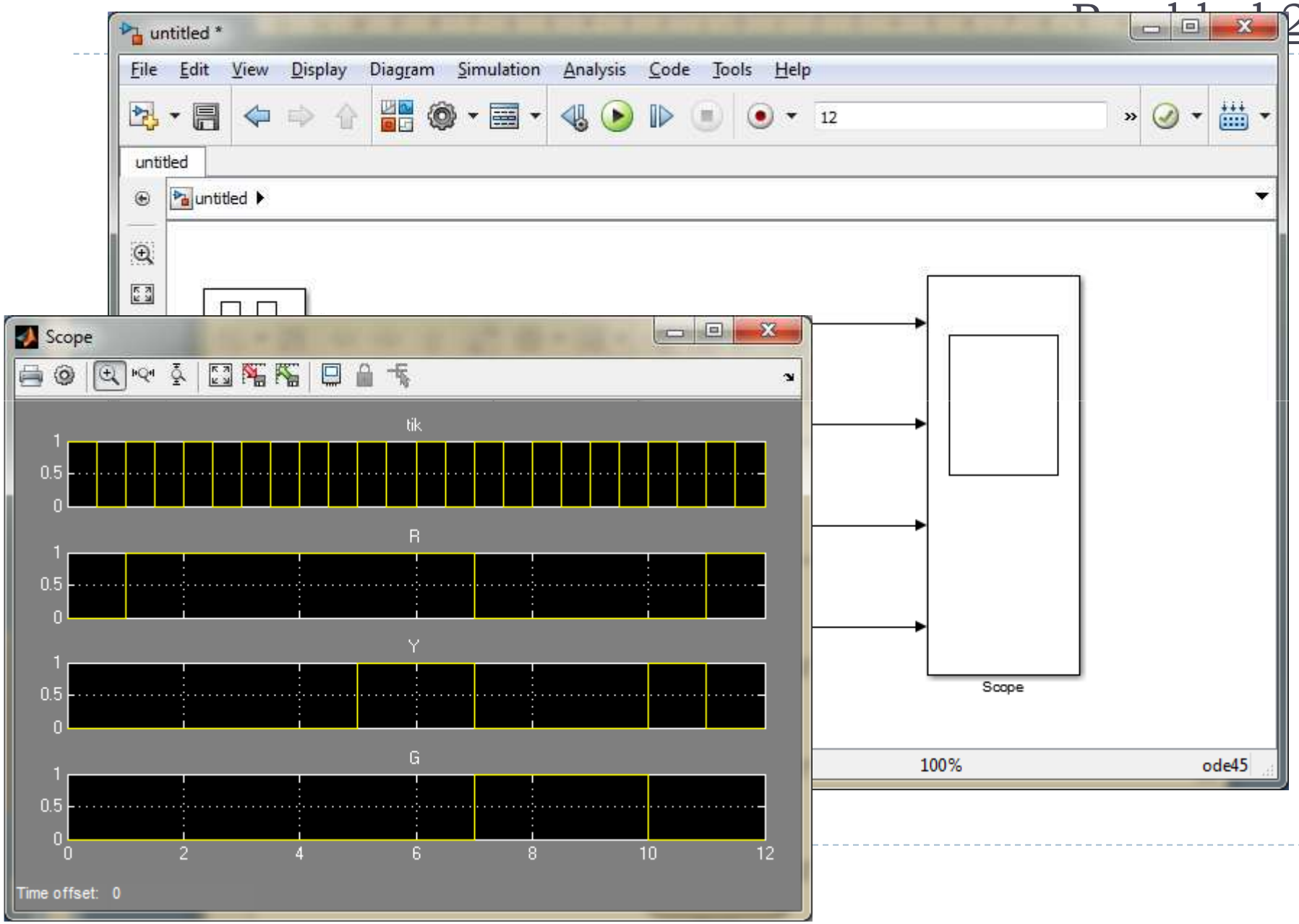
# Przykład 2

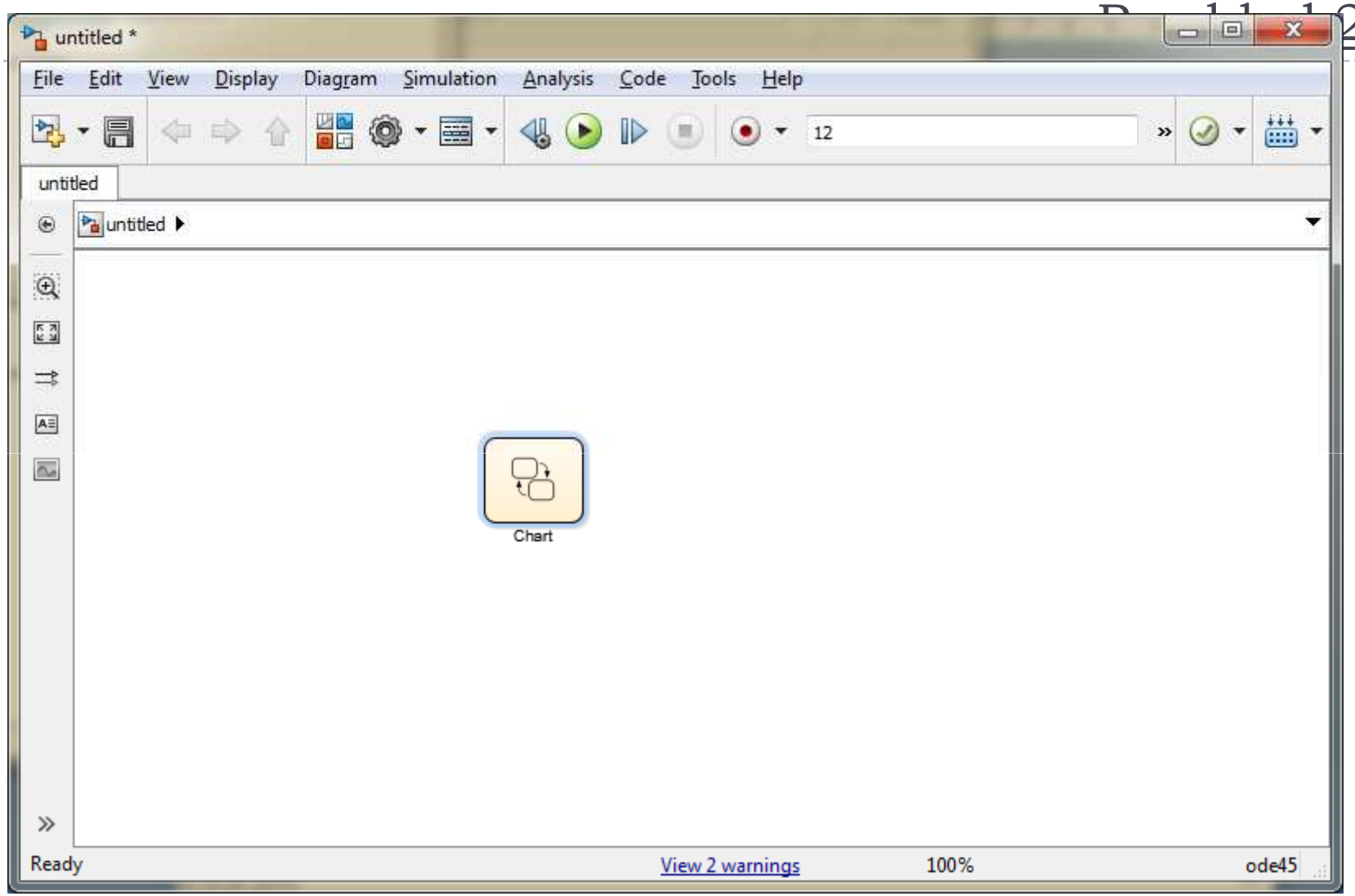


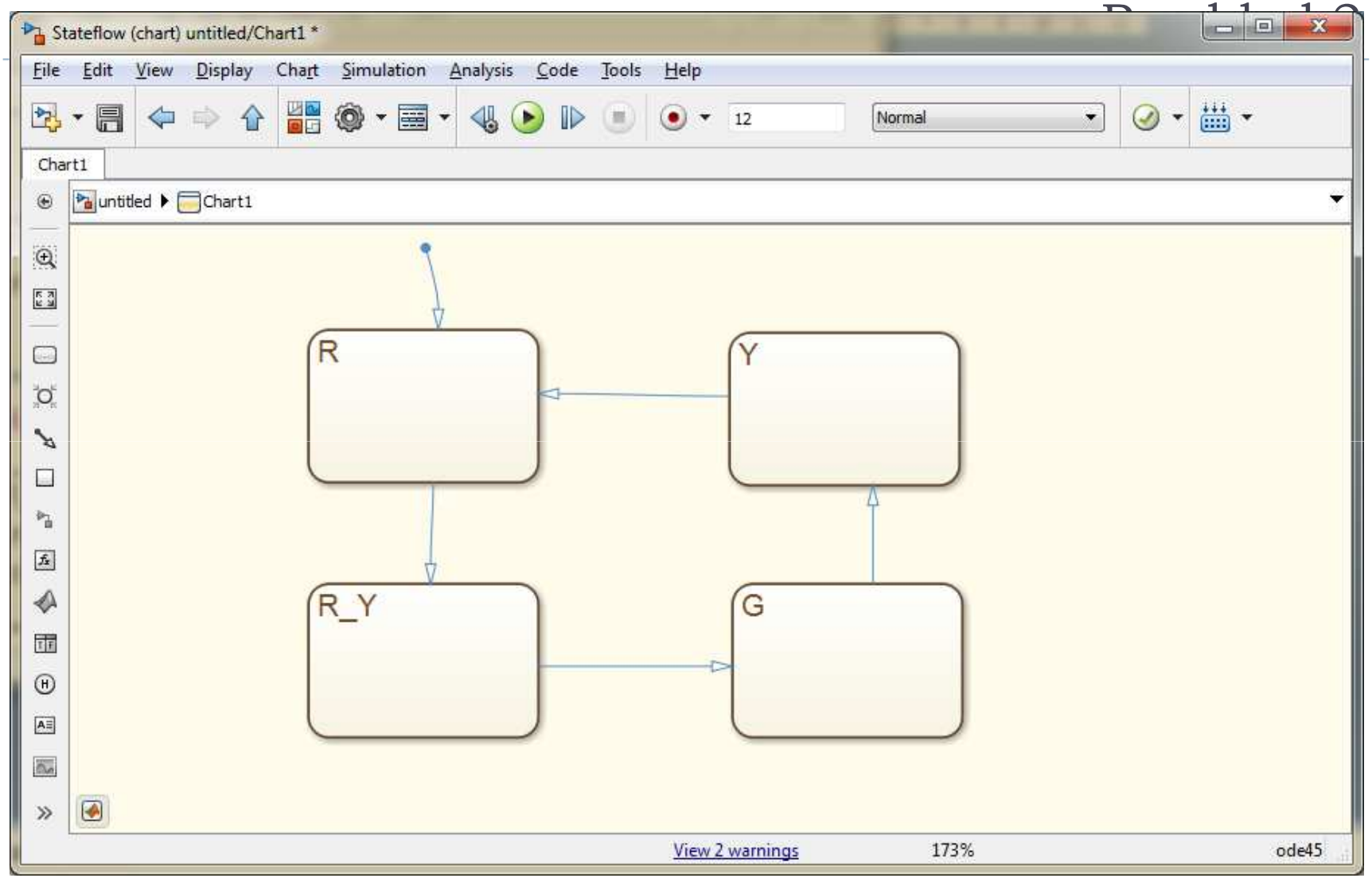
# Przykład 2

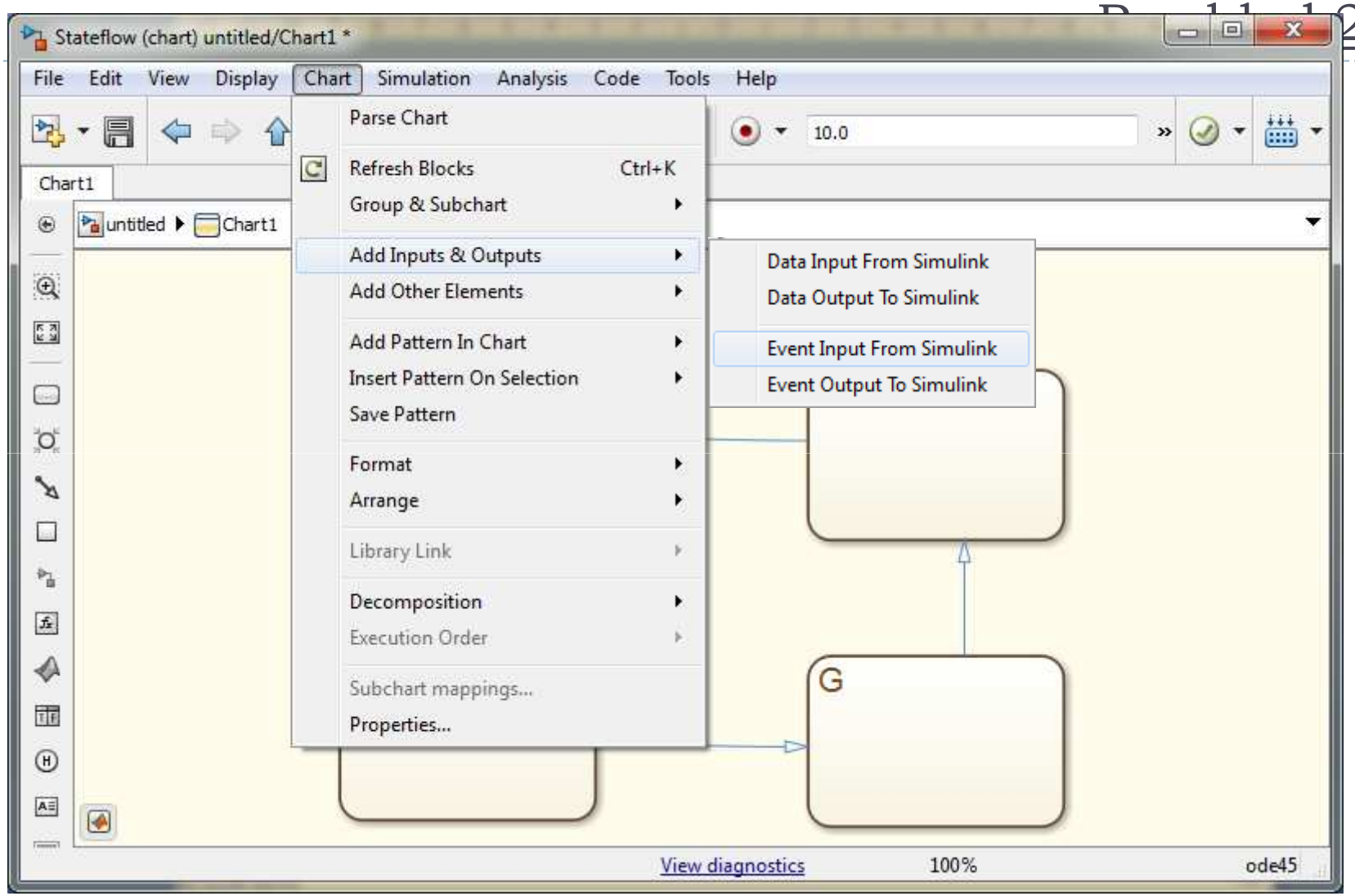
- Stan obiektu – ???
- Zdarzenie – ???
- Przejście – ???


untitled *
File Edit View Display Diagram Simulation Analysis Code Tools Help
12
untitled
untitled
tik
Pulse Generator
R
R
Y
Y
G
G
Chart1
Scope
Ready
View 2 warnings
100%
ode45


untitled *
File Edit View Display Diagram Simulation Analysis Code Tools Help
12
untitled
untitled
Scope
tik
R
Y
G
1
0.5
0
0 2 4 6 8 10 12
Time offset: 0
Scope
100%
ode45

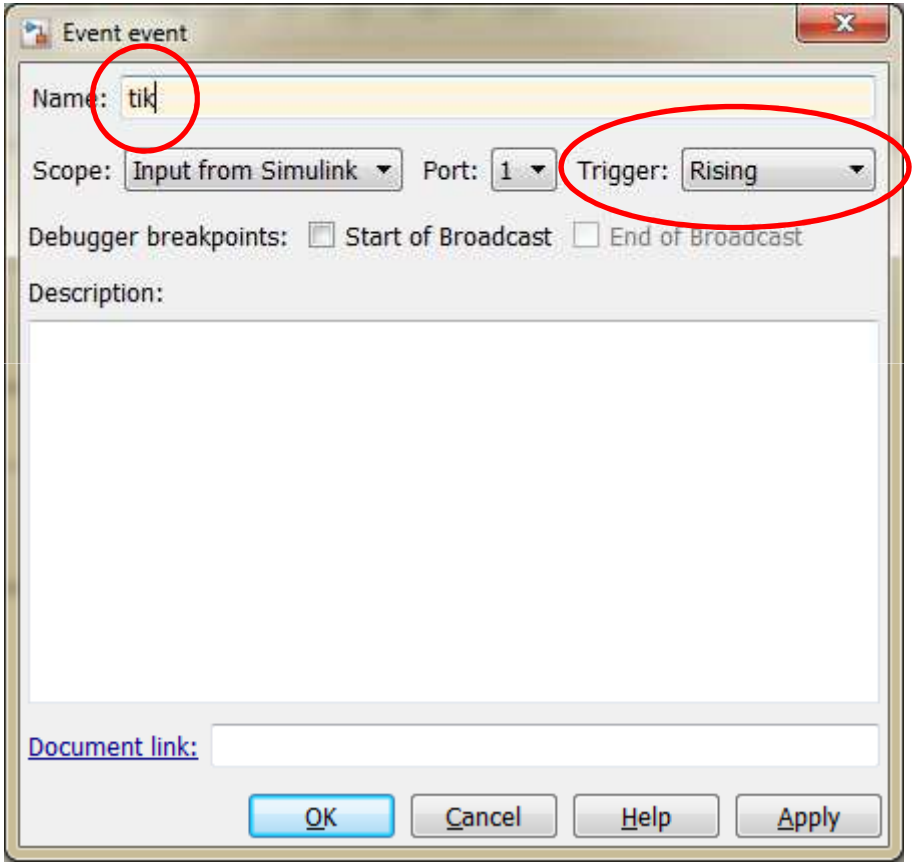

untitled *
File Edit View Display Diagram Simulation Analysis Code Tools Help
12
untitled
untitled
Chart
Ready
View 2 warnings
100%
ode45


Stateflow (chart) untitled/Chart1 *
File Edit View Display Chart Simulation Analysis Code Tools Help
12
Normal
Chart1
untitled
Chart1
R
Y
R_Y
G
View 2 warnings
173%
ode45

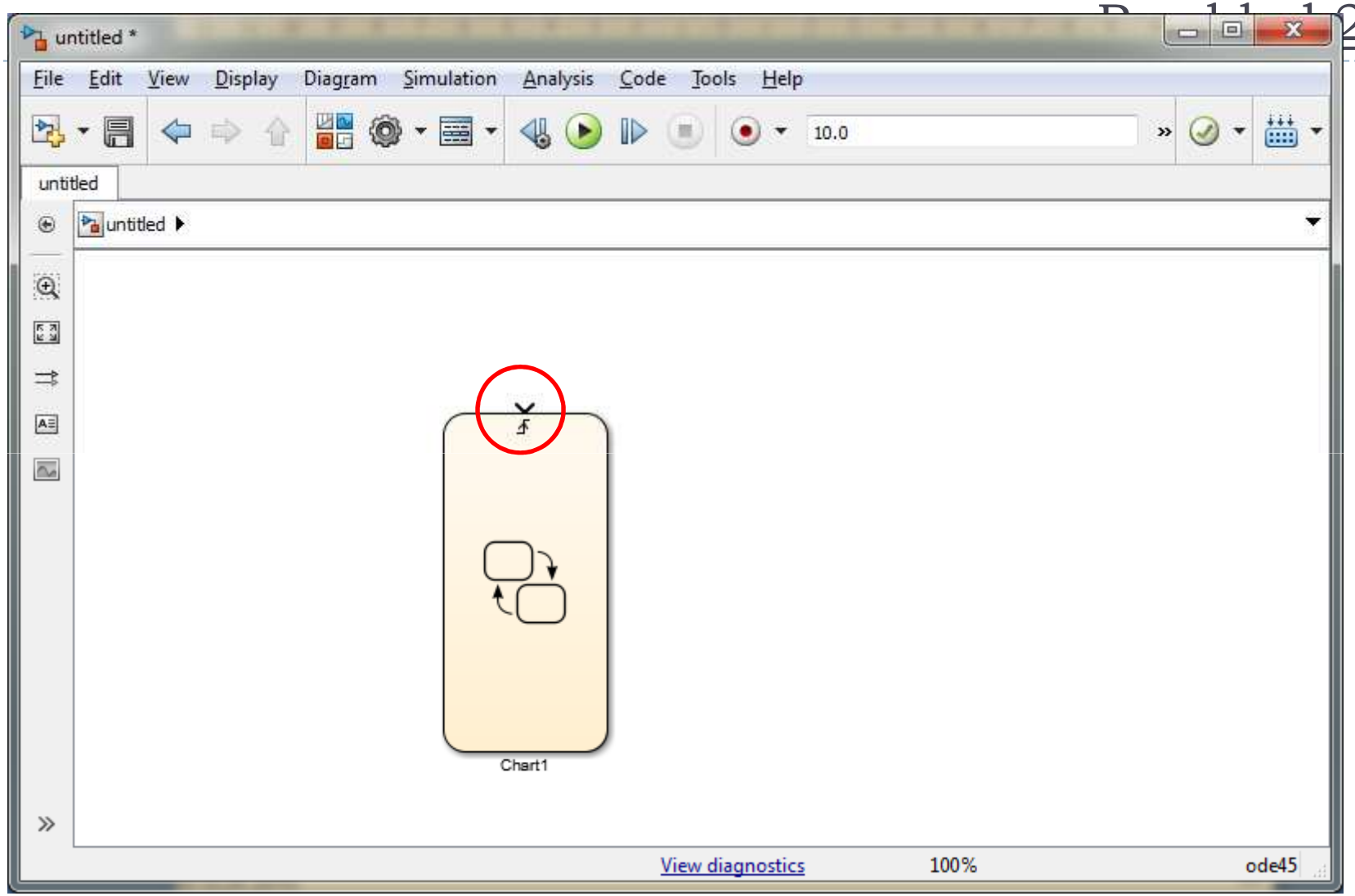


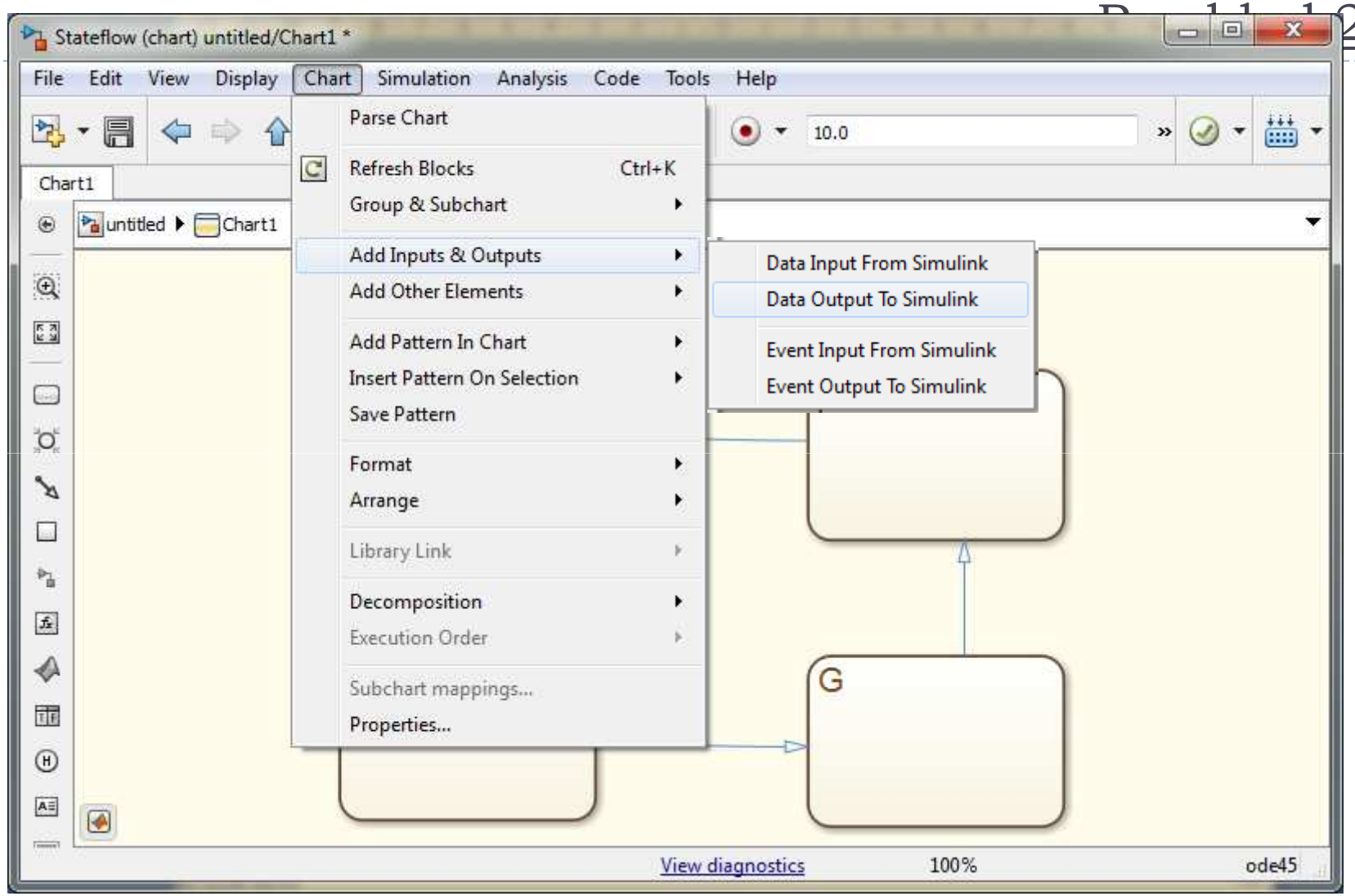

*Definiowanie wejścia związanego z generatorem czasu*

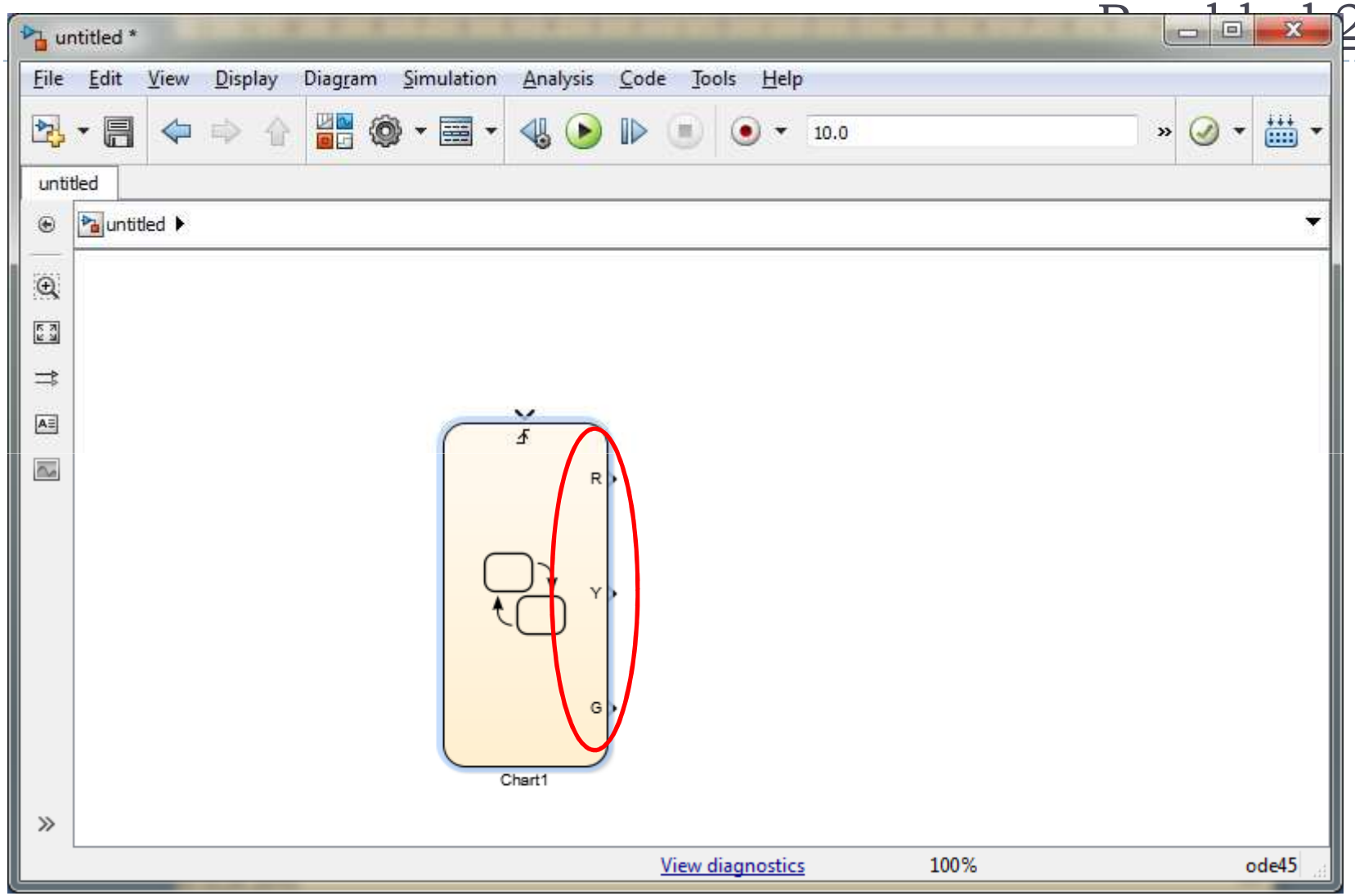
# Przykład 2

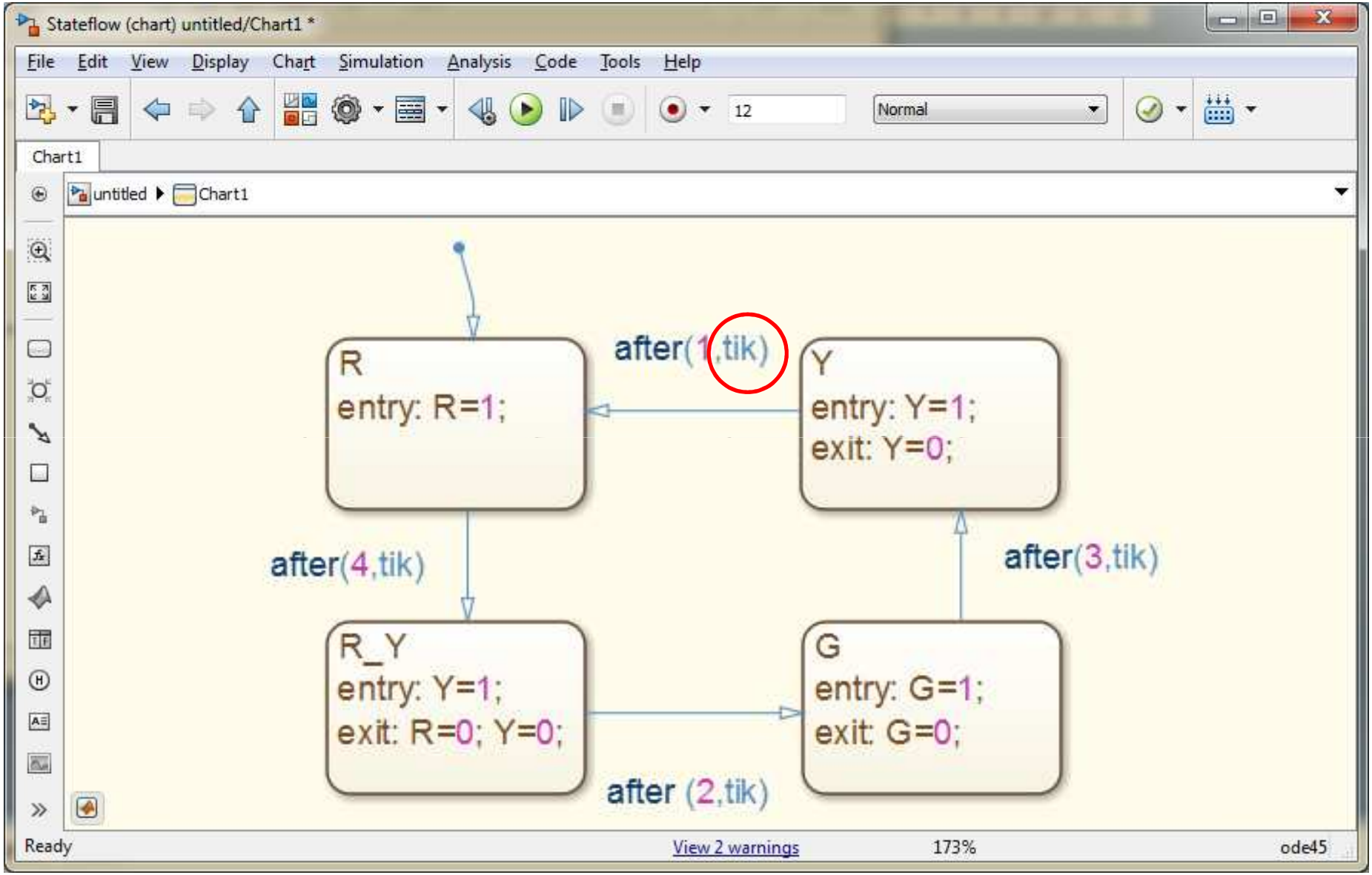


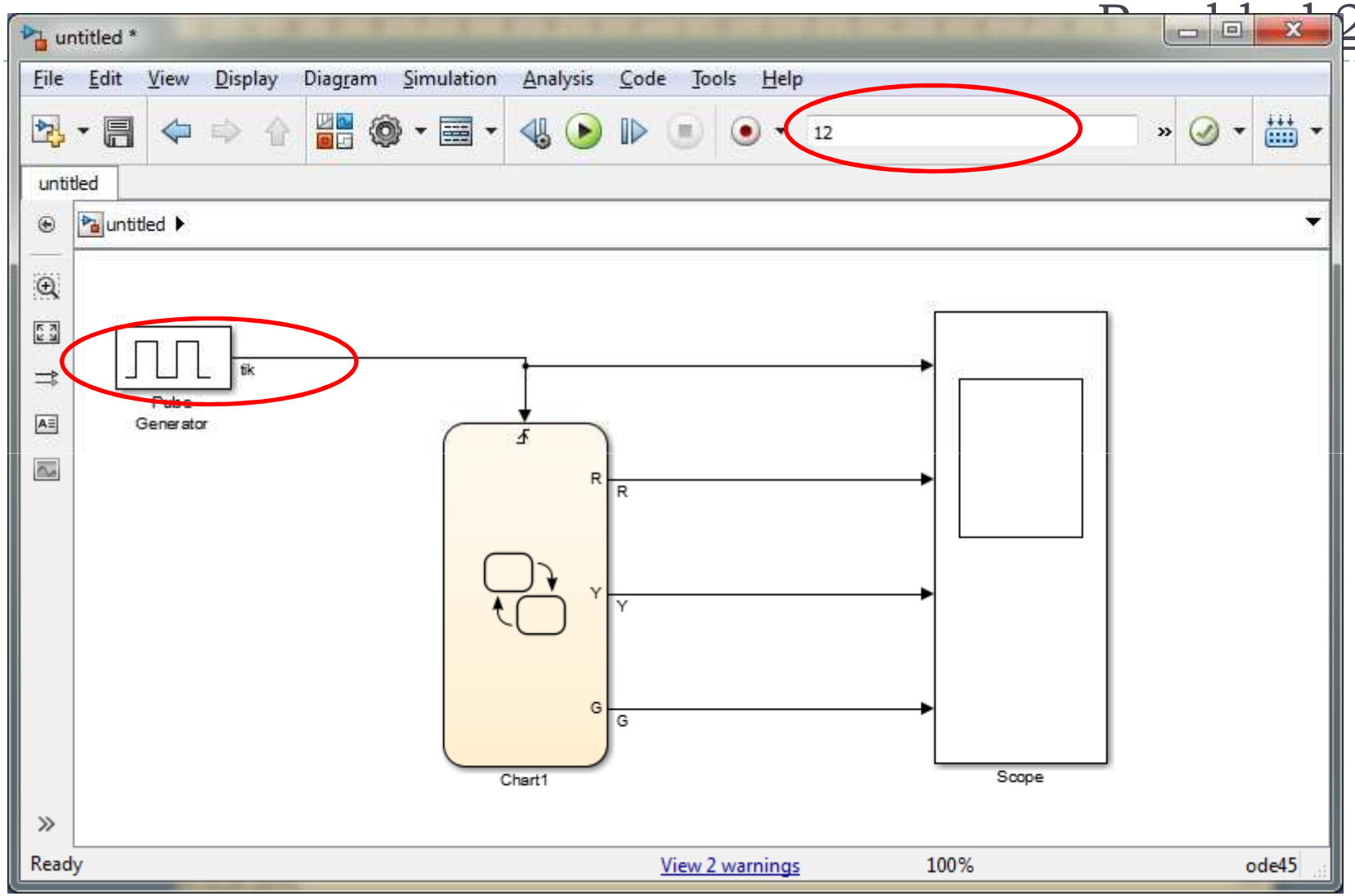

*Definiowanie wejścia związanego z generatorem czasu*


untitled *
File Edit View Display Diagram Simulation Analysis Code Tools Help
10.0
untitled
untitled
Chart1
View diagnostics
100%
ode45

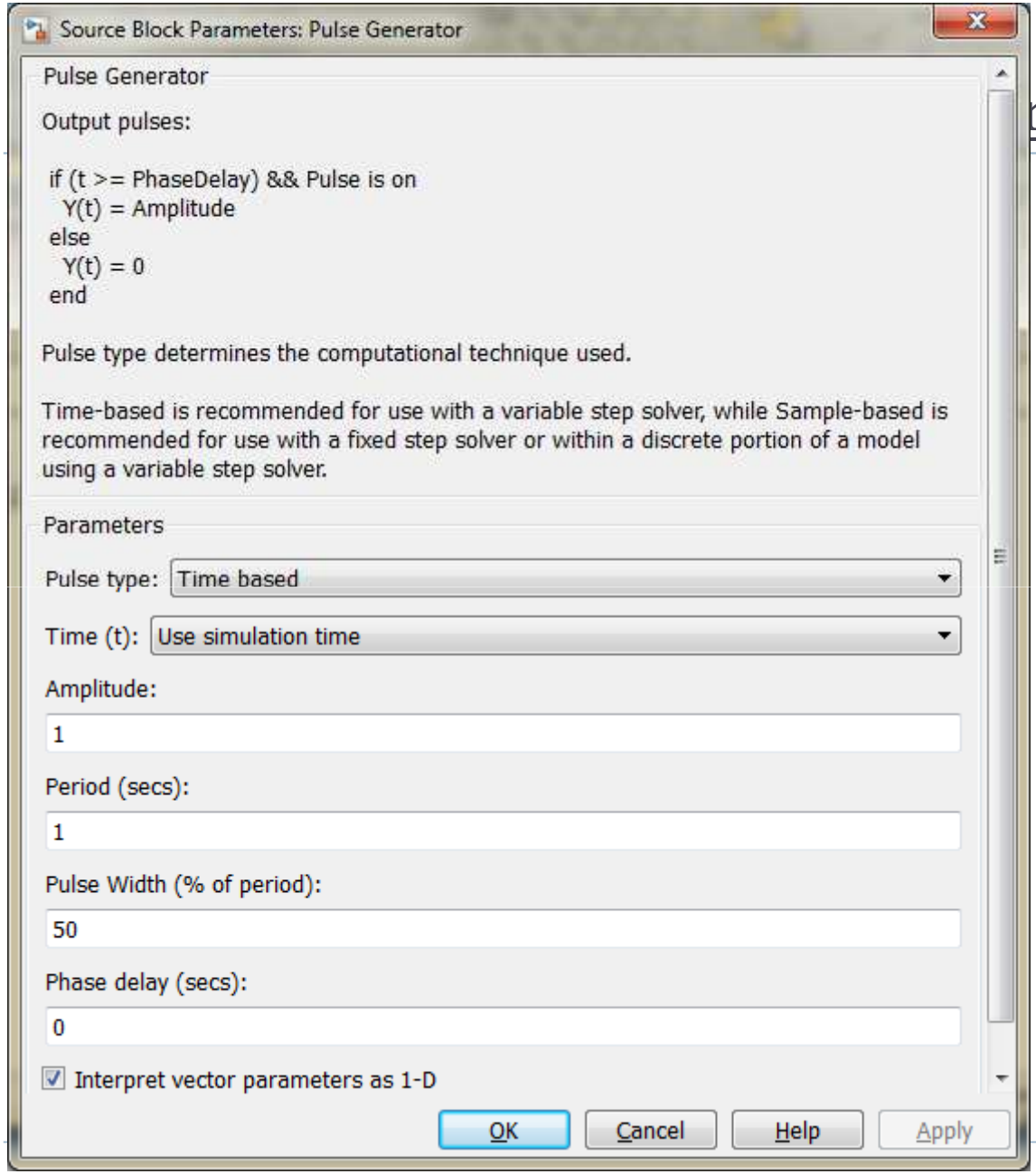

Stateflow (chart) untitled/Chart1 *
File
Edit
View
Display
Chart
Simulation
Analysis
Code
Tools
Help
10.0
Chart1
untitled
Chart1
Parse Chart
Refresh Blocks
Ctrl+K
Group & Subchart
Add Inputs & Outputs
Add Other Elements
Add Pattern In Chart
Insert Pattern On Selection
Save Pattern
Format
Arrange
Library Link
Decomposition
Execution Order
Subchart mappings...
Properties...
Data Input From Simulink
Data Output To Simulink
Event Input From Simulink
Event Output To Simulink
G
View diagnostics
100%
ode45


untitled *
File Edit View Display Diagram Simulation Analysis Code Tools Help
10.0
untitled
untitled
R
Y
G
Chart1
View diagnostics
100%
ode45

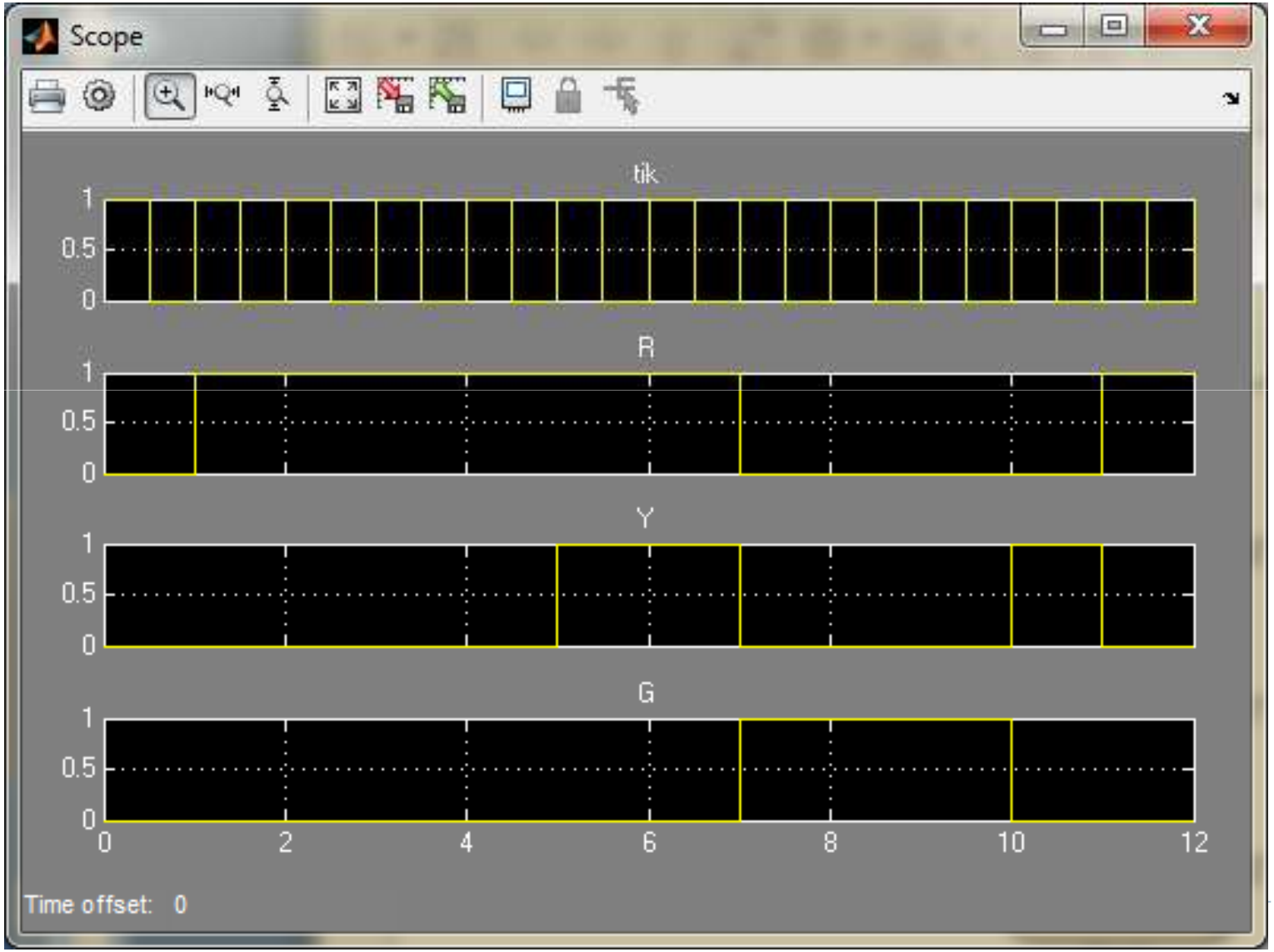

Stateflow (chart) untitled/Chart1 *
File Edit View Display Chart Simulation Analysis Code Tools Help
12
Normal
Chart1
untitled
Chart1
R
entry: R=1;
after(1,tik)
Y
entry: Y=1;
exit: Y=0;
after(4,tik)
after(3,tik)
R_Y
entry: Y=1;
exit: R=0; Y=0;
G
entry: G=1;
exit: G=0;
after (2,tik)
Ready
View 2 warnings
173%
ode45

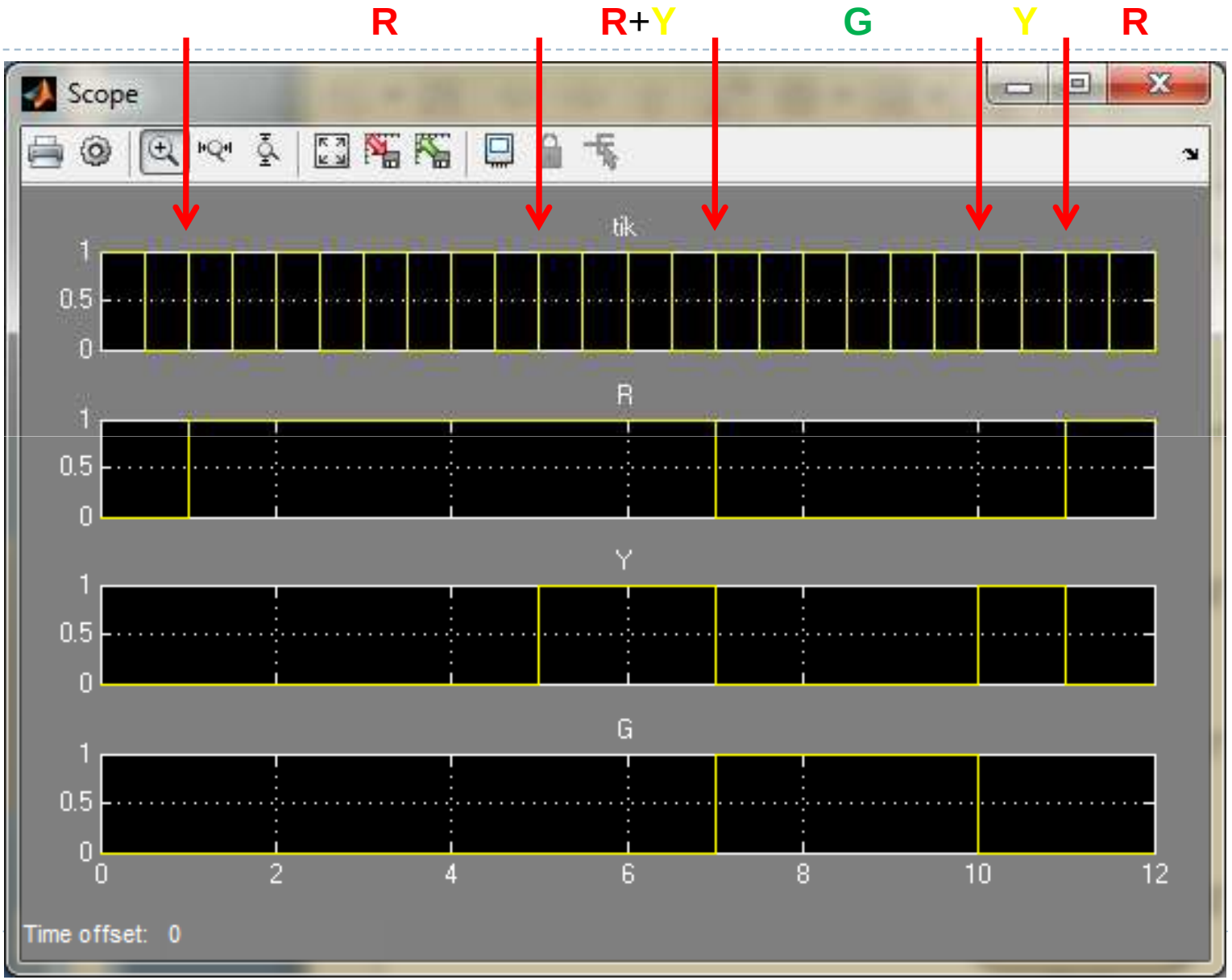

untitled *
File Edit View Display Diagram Simulation Analysis Code Tools Help
12
untitled
untitled
tik
Pulse
Generator
R
R
Y
Y
G
G
Chart1
Scope
Ready
View 2 warnings
100%
ode45

# rzykład 2



# Przykład 2

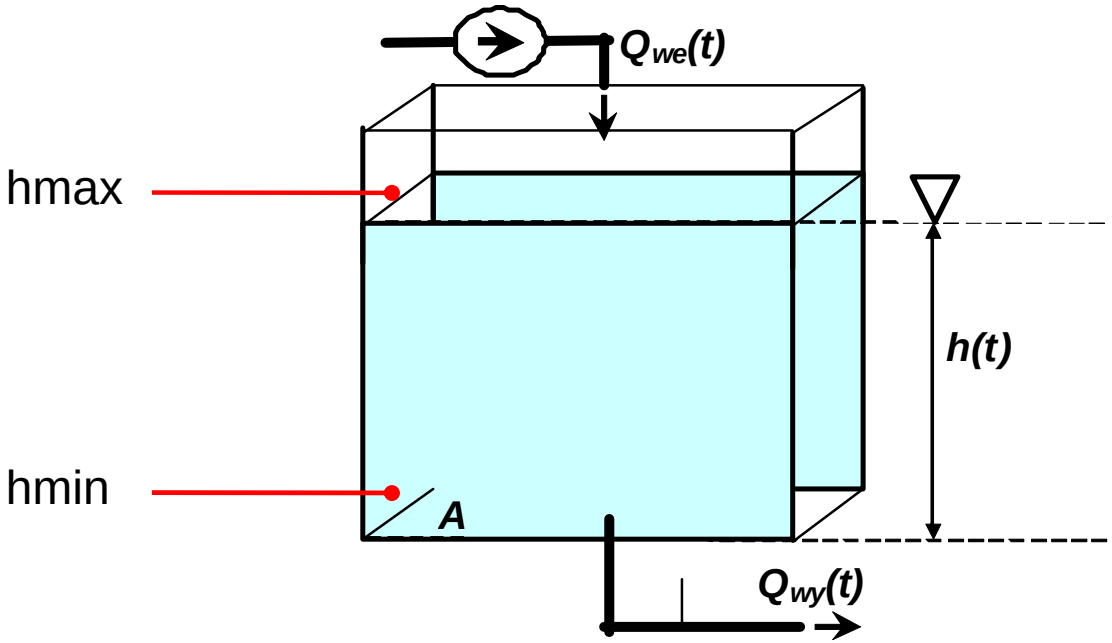




# *Przykład 3*

## *- dwustanowy regulator poziomu medium w zbiorniku*

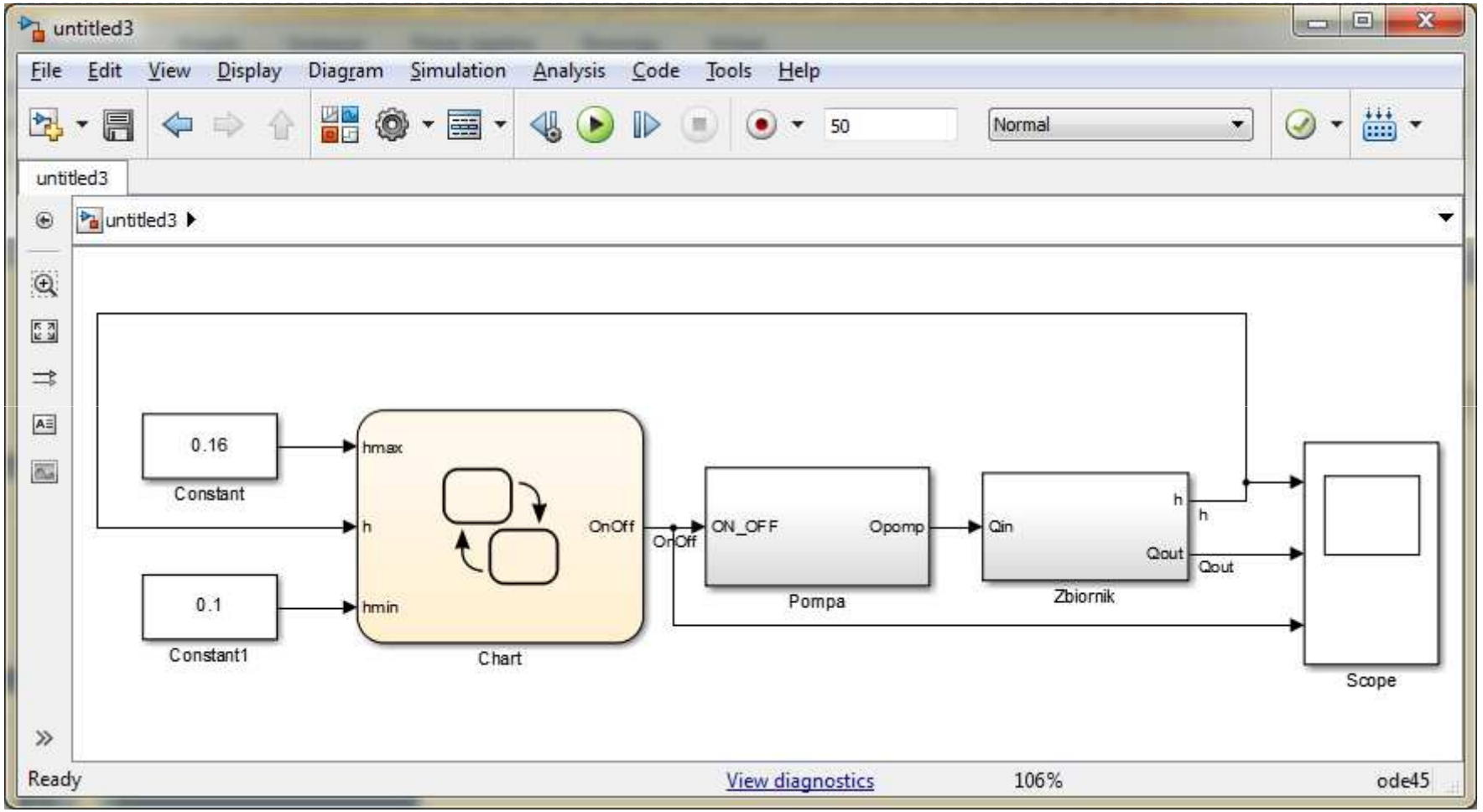
# Przykład 3



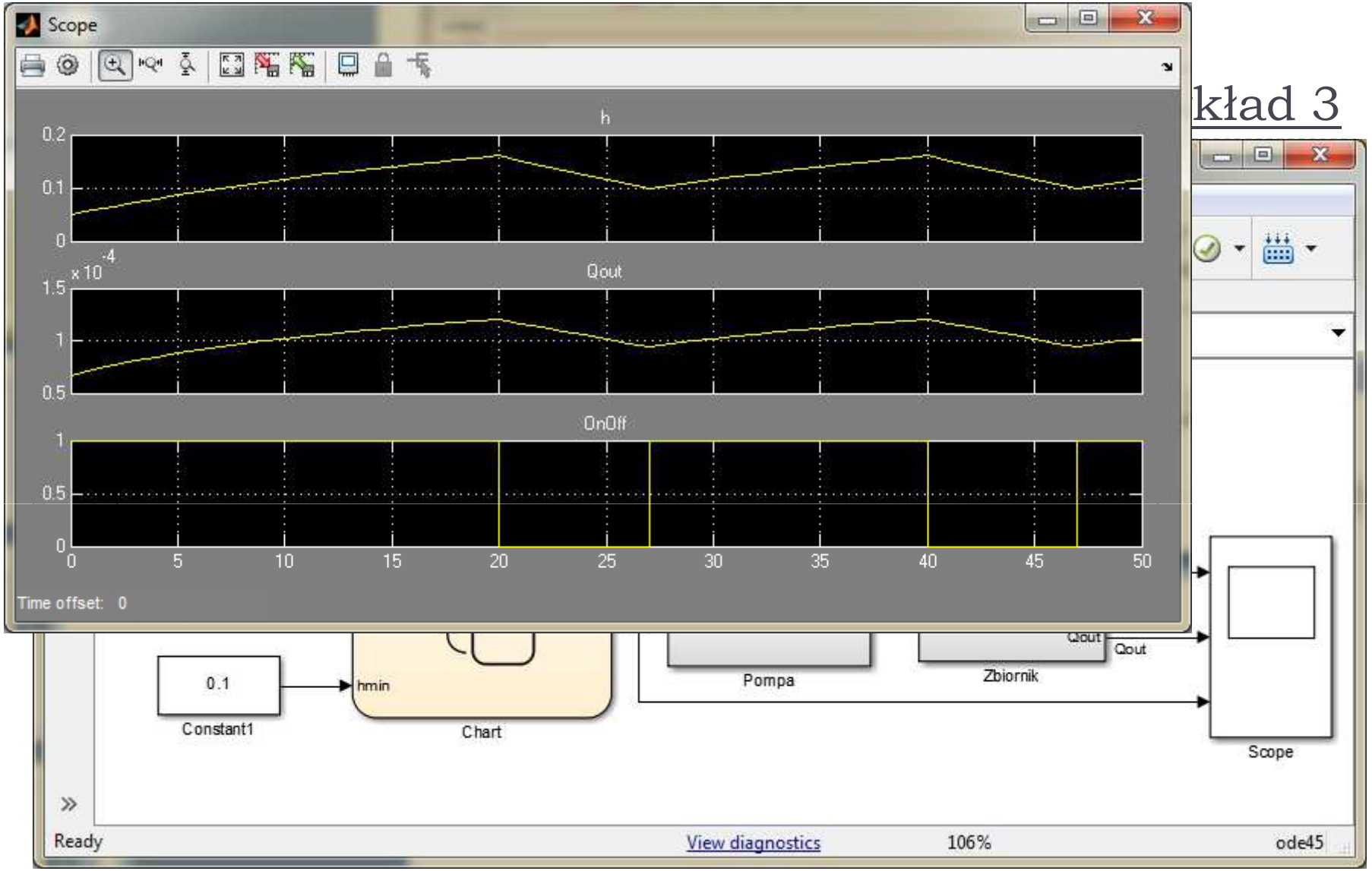
# Przykład 3

- Stan obiektu – ???
- Zdarzenie – ???
- Przejście – ???

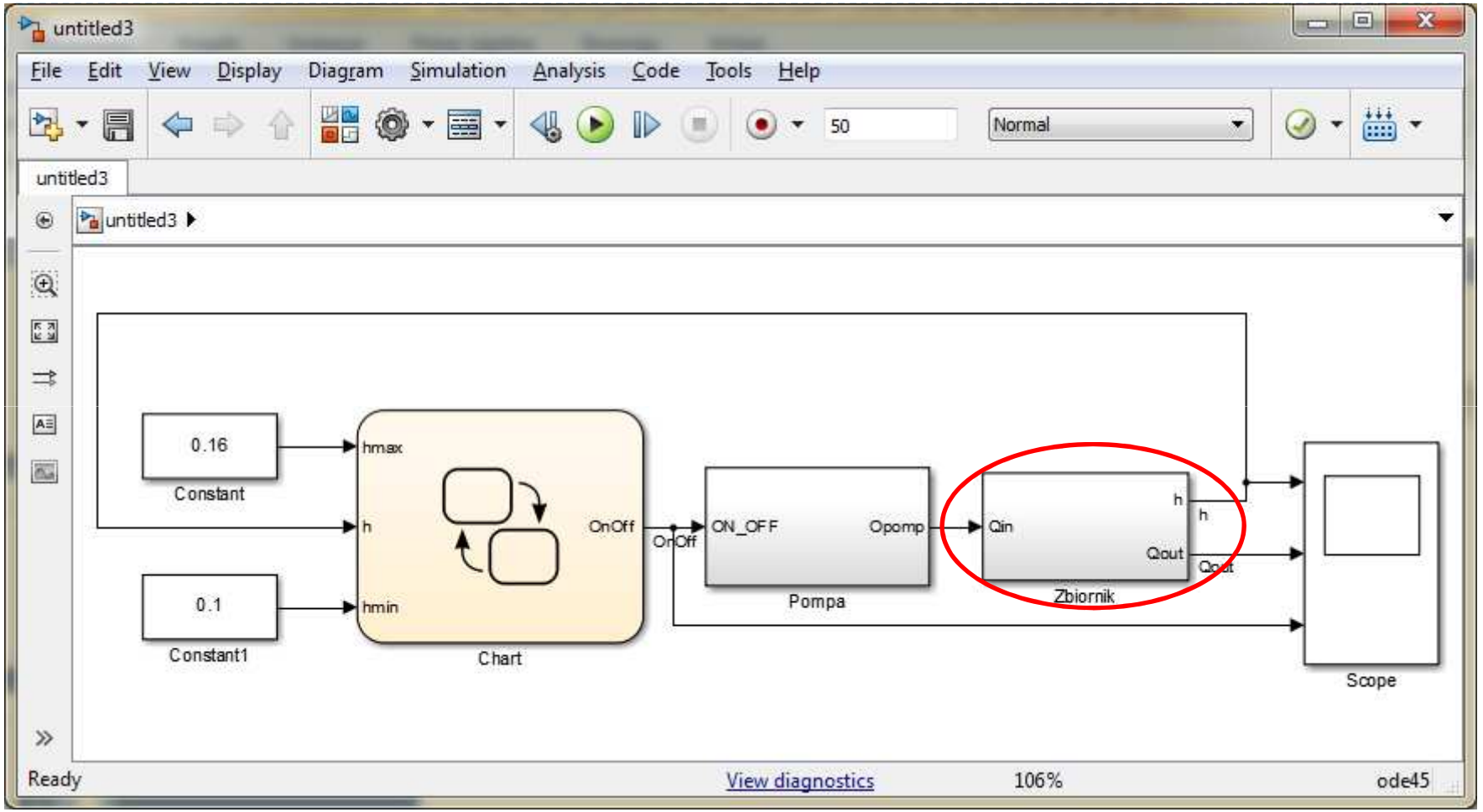
# Przykład 3



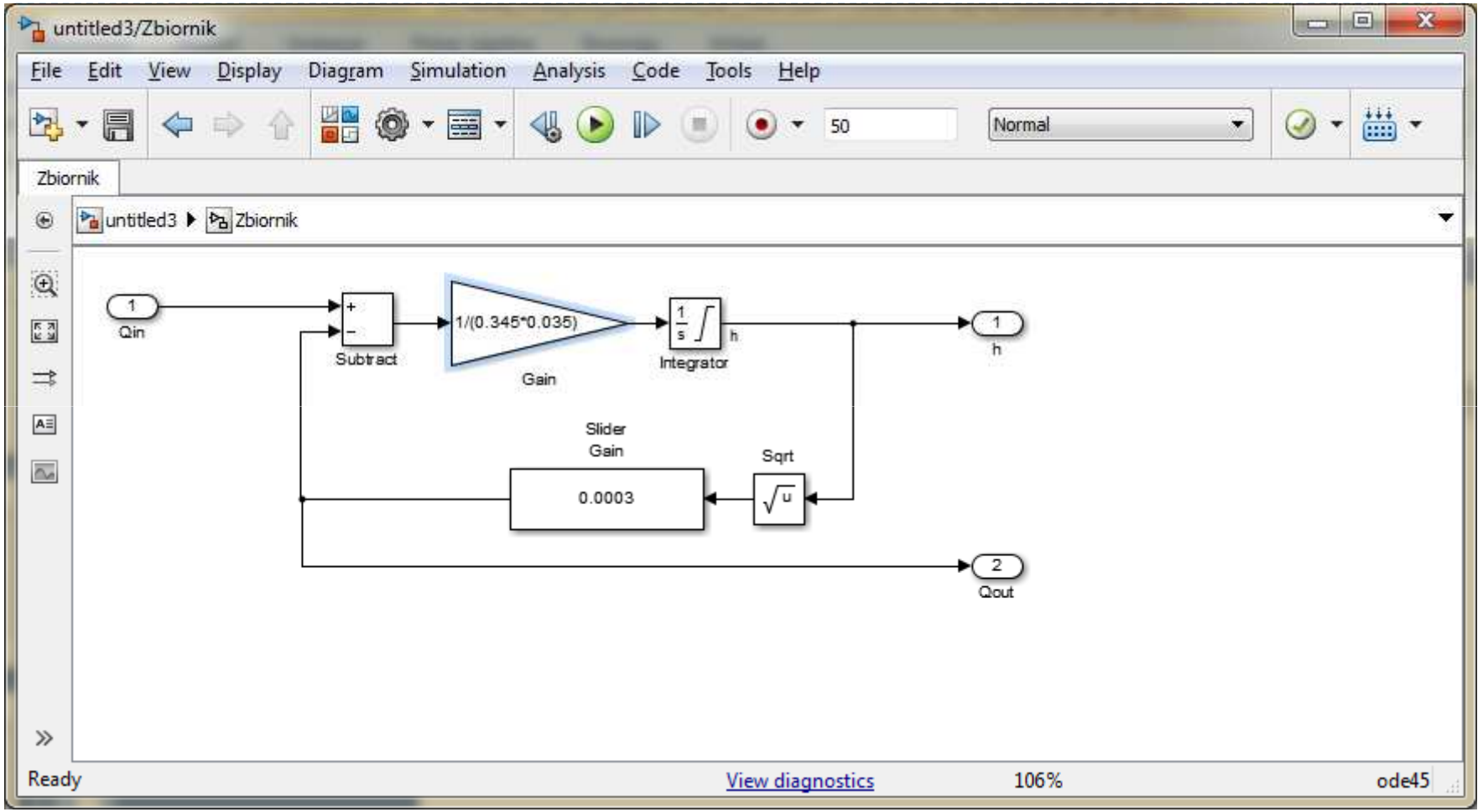
# kład 3



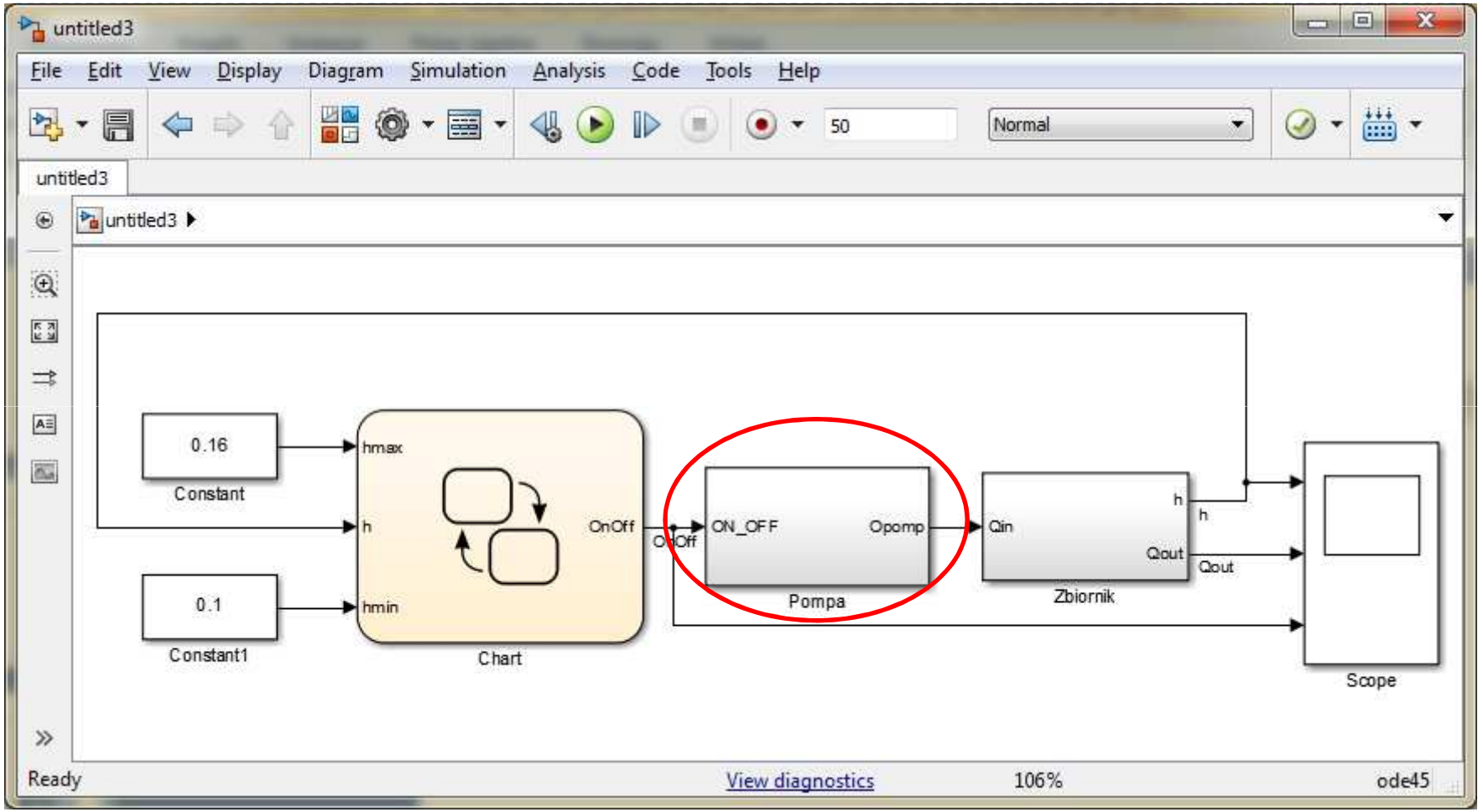
# Przykład 3



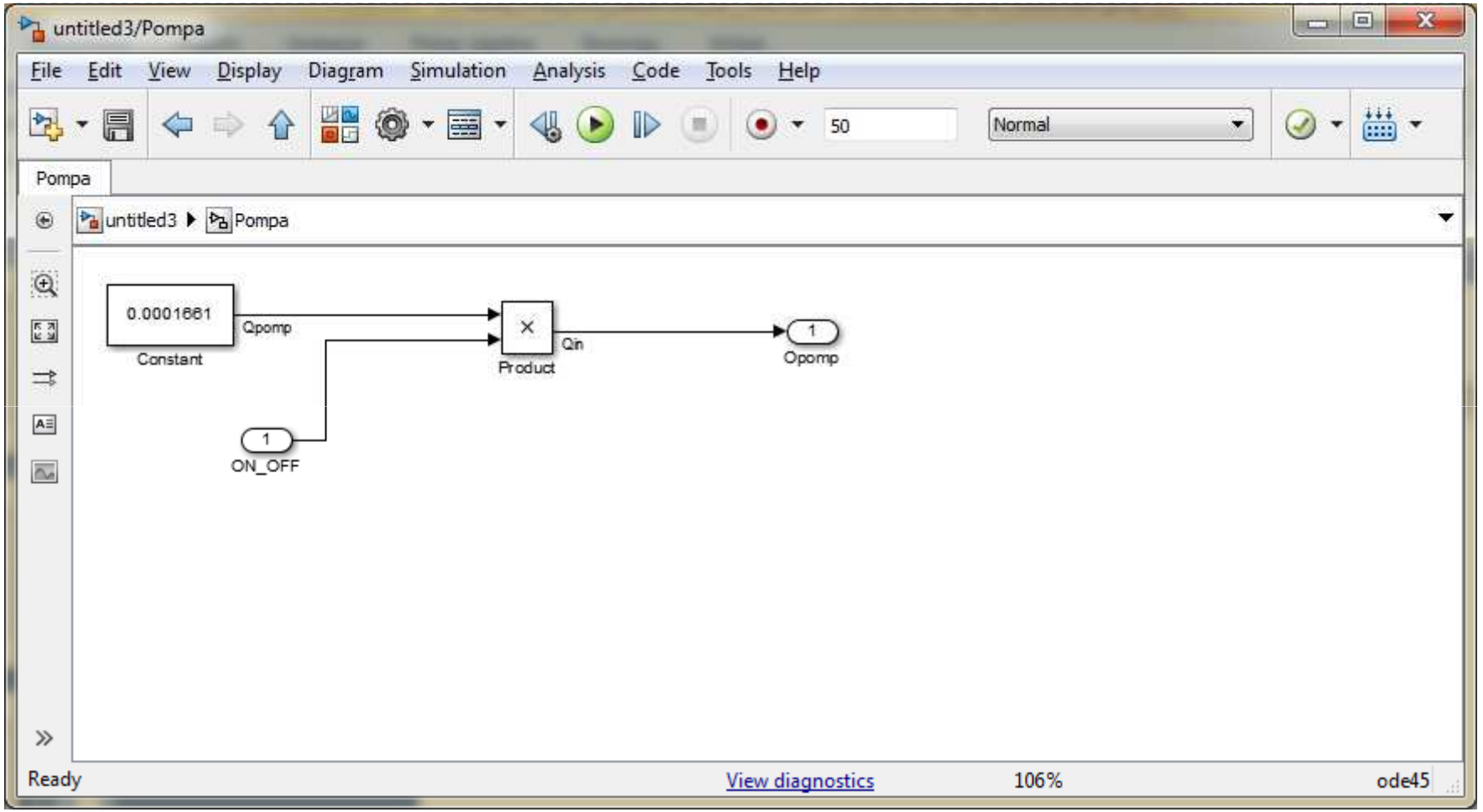
# Przykład 3



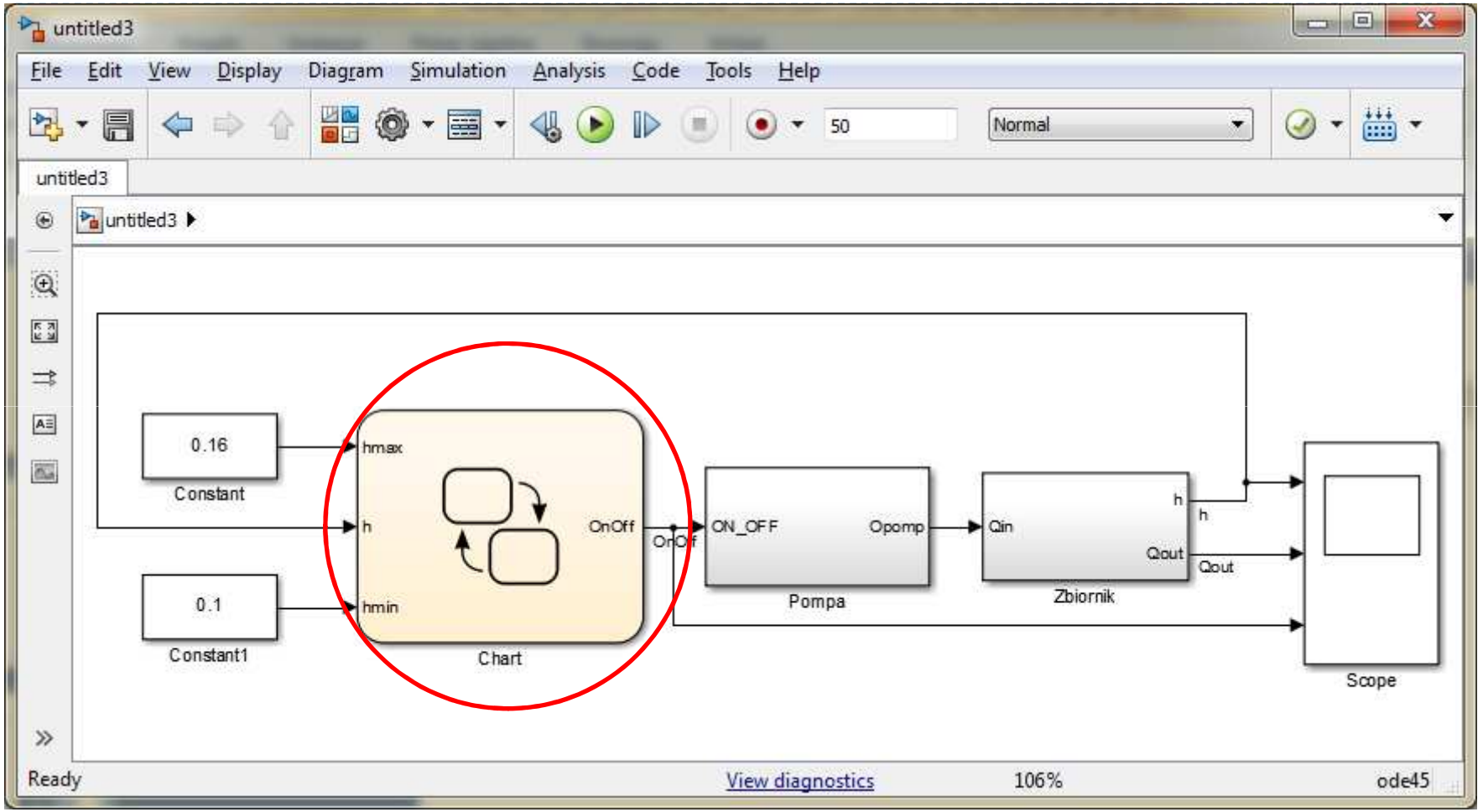
# Przykład 3



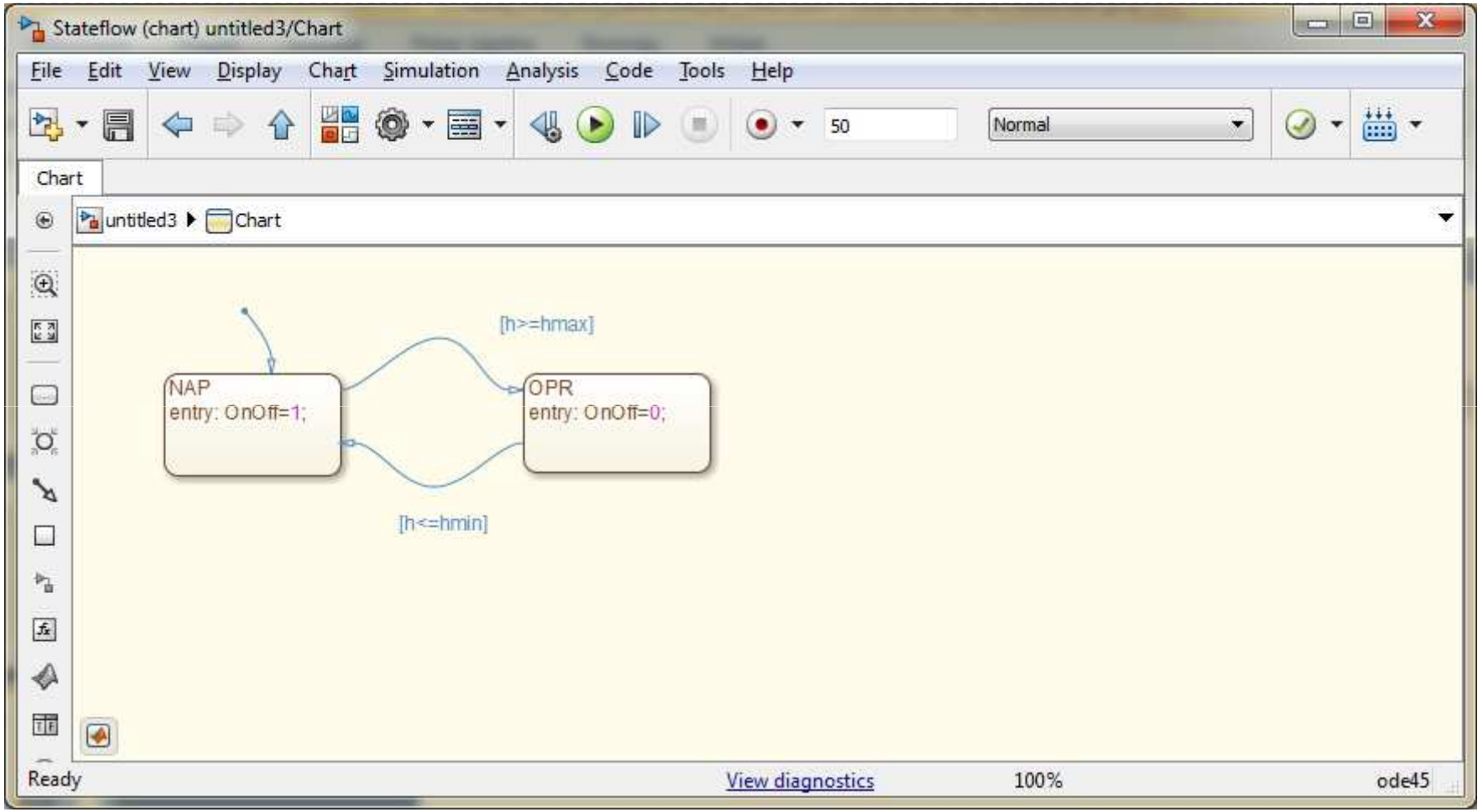
# Przykład 3



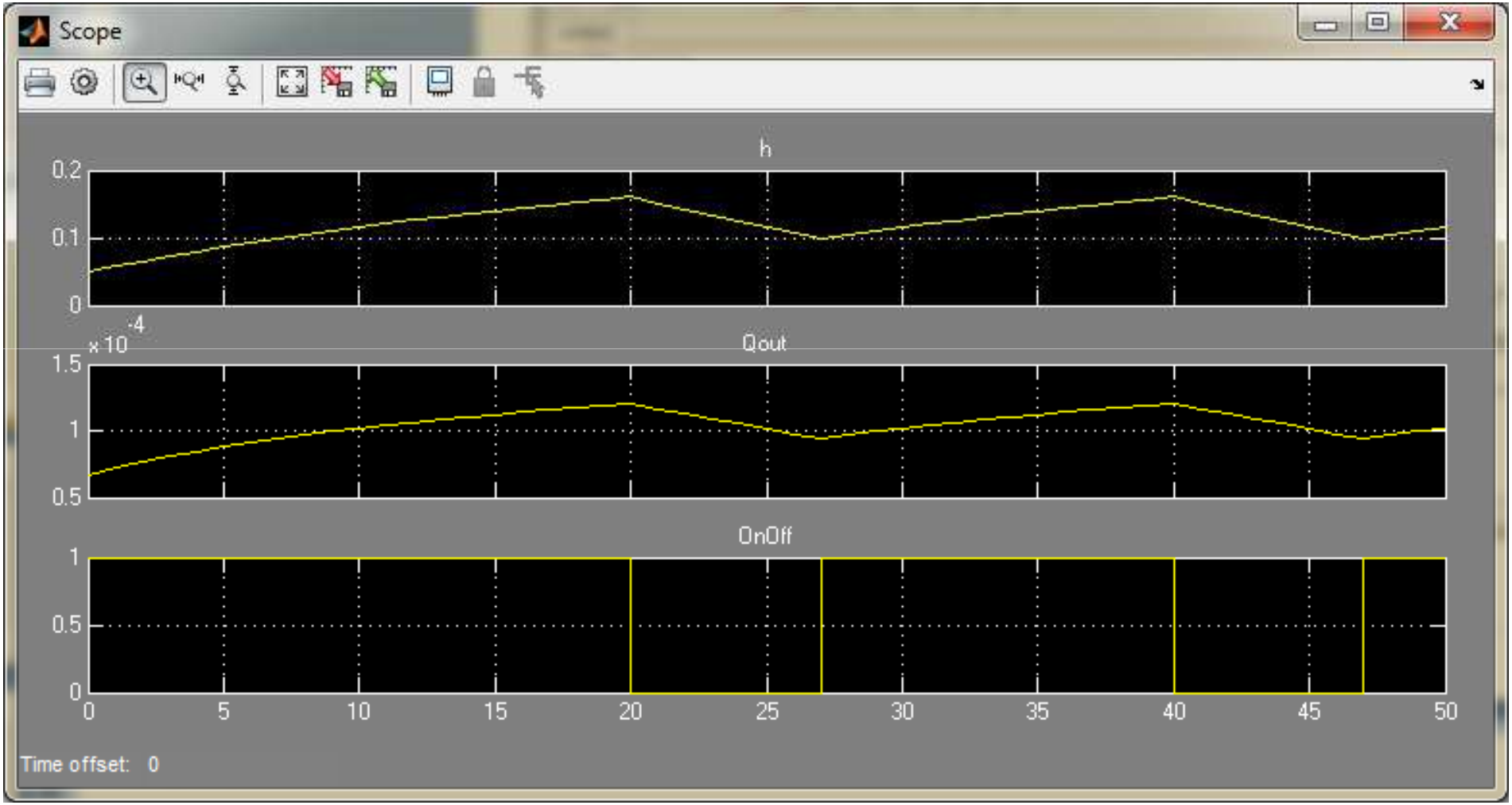
# Przykład 3



# Przykład 3



# Przykład 3



Bibliografia:

[1] T. Szmuc, G.Motet (2000). Specyfikacja i projektowanie oprogramowania systemów czasu rzeczywistego. Uczelniane Wydawnictwa Naukowo-Dydaktyczne, Kraków.

[2] Elektronika praktyczna 1/2010.

[3] I. Sommervill (2003). Inżynieria oprogramowania. WNT.

[4] http://www.mathworks.com

*Dziękuję za uwagę !!!*