Slajdy do przedmiotu: Języki modelowania i symulacji

Materiały przygotowali:

dr inż. Bogdan Pankiewicz

dr inż. Marek Wójcikowski

Gdańsk, styczeń 2015

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Część I - PSPICE

Czyli jak wykonać symulację elektryczną obwodu przy użyciu oprogramowania PSPICE

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 1. Wstęp

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE - Literatura

Literatura podstawowa:

1. J. Izydorczyk, „PSpice komputerowa symulacja układów elektronicznych”, Helion, 1993.
2. Cadence Design Systems, „PSPICE reference guide”, Cadence PCB Design Systems, 2001. Dokumentacja w wersji elektronicznej dostępna w laboratorium w katalogu C:\Cadence\PSD_14.0\doc.

Literatura uzupełniająca:

1. P. Antognetti, Massobrio, „Semiconductor Device Modeling with SPICE”, McGraw-Hill, 1988.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Historia powstania

- SPICE – ang. Simulation Program with Integrated Circuit Emphasis, ze względu na koszty produkcji układów scalonych bardzo ważne stała się ich wstępna symulacja, jeszcze przed procesem produkcji i stąd powstała potrzeba symulacji.
- Pierwsza wersja symulatora powstaje w 1973r na Unversity of California, Berkeley i jest sponsorowana przez United States Department of Defense.
- Następne wersje są kontynuacją, powstają również wersje komercyjne, SPICE natomiast staje się oprogramowaniem typu Public Domain.

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Historia powstania c.d.

- Obecnie mamy dostępne również inne symulatory obwodów elektrycznych: Eldo, Spectre, HSpice, PSpice, Saber i inne.
- PSPICE - Personal Simulation Program with Integrated Circuit Emphasis, wytworzony inicjalnie prze firmę MicroSim.
- Następnie wykupiony przez OrCad a następnie przez
- Cadence Design Systems.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Darmowe wersje symulatorów

- LT SPICE dostępny: http://www.linear.com/designtools/software/#Spice
- Berkeley SPICE 3fg, dostępny: http://bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE/
- Wersja DEMO PSPICE, dostępna: http://www.cadence.com/products/orcad/pages/downloads.aspx

Dostęp do ww. stron sprawdzony w dniu 15.01.2015.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 2. Składnia ogólna pliku PSPICE

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Składnia ogólna pliku opisu układu


```
Układ 1
R1 1 0 10k
C1 1 0 1f
* to jest komentarz
.op
.end
```

Pierwsza linia - tytuł obwodu, dozwolona dowolna zawartość tej linii.

Kolejne linie zaczynające się od litery – lista połączeniowa obwodu.

Linie zaczynające się „*” – komentarz trwający do końca linii.

Linie zaczynające się kropką stanowią polecenia symulatora.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Ogólne zasady opisu

- typowe rozszerzenie nazwy pliku to „*.cir”, inne rozszerzenia mogą powodować problemy w działaniu oprogramowania,
- należy unikać polskich znaków i spacji zarówno w nazwach plików jak i w opisie wewnątrz pliku SPICE,
- składnia SPICE jest nieczuła na wielkość liter,
- w pierwszej linii podaje się zwięzły opis badanego układu,
- ostatnia linia pliku powinna zawierać polecenie **`.end`**, jeśli plik zawiera linie za **`.end`** traktowane są one jako kolejny obwód do analizy,
- puste linie są pomijane,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Ogólne zasady opisu c.d.

- linie rozpoczynające się literą są opisem elementu zależnego od tego jaka jest to litera np.: R- rezystor, L- cewka, C- kondensator i.t.d.,
- linie rozpoczynające się kropką są poleceniami, parametrami lub dodatkowym sterowaniem pracą symulatora,
- linie rozpoczynające się gwiazdką „ ***” są komentarzem trwającym do końca linii,
- linie w których występuje średnik ; – od pozycji średnika do końca linii też są traktowane jako komentarz,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Ogólne zasady opisu c.d.

- długie linie (typowo ponad 80 znaków) można przenieść do kolejnej linii opisu rozpoczynając ją znakiem „+”

Przykład:

```
R1 1 0 22k
```

można zapisać:

```
R1 1 0
+ 22k
```

PSPICE – Ogólne zasady opisu c.d.

- węzeł masy ma w SPICE zawsze nazwę „**0**” (zero),
- węzeł ten **musi** wystąpić w badanym układzie,
- każdy z węzłów musi mieć przynajmniej 2 połączenia do elementów – w przeciwnym przypadku węzeł będzie typu „floating” i zablokuje możliwość symulacji,
- niektóre symulatory samodzielnie usuwają węzły „floating” i umożliwiają symulacje generując jedynie ostrzeżenie o potencjalnym problemie,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- również węzły, które mają 2 połączenia ale wyłącznie zrealizowane poprzez kondensatory są typu „floating”,
- niedozwolone jest równoległe łączenie idealnych źródeł napięciowych,
- niedozwolone jest szeregowe łączenie idealnych źródeł prądowych.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 3. Jednostki, wartości, temperatura w PSPICE

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

PSPICE – Jednostki

- Większość używanych jednostek to jednostki S.I. (tylko pojedyncze przypadki w niektórych modelach elementów są spoza układu S.I.)
- można stosować przyrostki wartości:

f	p	n	u	m	k	meg	g	t	mil
F	P	N	U	M	K	MEG	G	T	MIL
10^{-15}	10^{-12}	10^{-9}	10^{-6}	10^{-3}	10^{3}	10^{6}	10^{9}	10^{12}	$25.4*10^{-6}$

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Wartości

Formaty podawania wartości parametrów elementów:

- zwykły, np.: `123.23`, - punktem dziesiętnym jest kropka, można nie podawać zera przed wartością np. `0.123=.123`,
- naukowy, np.: `1.2323e-5`,
- z użyciem przyrostka, np.: `1.2323m`,
- z dodatkowym podaniem jednostki, np.: `1.2323e-3V` lub `1.2323mV`,
- uwaga na jednostkę F gdyż jest to przyrostek oznaczający 10^{-15} więc:

  `1.2323e-6F nie jest równe 1.2323uF`

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Wartości c.d.

- wartości można również podawać pośrednio poprzez wykorzystanie wyrażenia umieszczonego w nawiasie klamrowym, np.: **`R1 1 0 {10k*2/23}`**,
- wartości wyliczane w nawiasie klamrowym mogą wykorzystywać bardziej złożone zależności poprzez zastosowanie operatorów, funkcji oraz możliwość umieszczenia w wyrażeniu parametrów, np.:

```
.param mult=10
R1 1 0 {mult*2*sin(mult)*1K}
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Temperatura w PSPICE

Polecenie **.TEMP** ustala wartość temperatury elementów, dla której wykonywane są analizy. Zakłada się, że wszystkie parametry modeli były pomierzone dla temperatury nominalnej oznaczonej parametrem **TNOM** (domyślnie 27°C), wartość **TNOM** można ustawić poleceniem **.OPTIONS**.

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Temperatura w PSPICE, c. d.

- Temperatura w SPICE ma jednostkę **°C**.
- Temperatura nominalna **TNOM** jest ustawiona do wartości domyślnej równej **TNOM=27°C**, wartość domyślnej temperatury nominalnej można modyfikować poleceniem **.options**.
- Temperatura ogólna symulacji jest ustawiana poleceniem **.temp=new_value** lub jako zmiana parametru o nazwie **TEMP** , domyślna wartość temperatury jest równa **27°C**.
- Bieżąca temperatura elementu może być ustawiona na kilka sposobów: parametr globalny **TEMP**, parametr modelu **T_ABS** oraz parametr modelu **T_REL_GLOBAL**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Temperatura w PSPICE c.d.

Temperatura poszczególnych elementów:

- dla elementu bez modelu lub dla elementu dla którego w modelu nie podano parametrów **T_ABS** lub **T_REL_GLOBAL** temperatura jest równa parametrowi **TEMP**,
- dla elementu dla którego w modelu podano parametr **T_ABS** temperatura jest równa temu parametrowi,
- dla elementu dla którego w modelu podano parametr **T_REL_GLOBAL** temperatura jest równa **TEMP + T_REL_GLOBAL**,
- dla elementu dla którego model jest typu AKO i podano parametr **T_REL_LOCAL** temperatura jest równa **TEMP** z modelu odniesienia **+ T_REL_GLOBAL**,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Symulacje dla różnych temperatur:

- polecenie **`.TEMP=40`** lub równoważne **`.TEMP 40`** ustala temperaturę elementów używaną w czasie symulacji (niektóre elementy mogą mieć temperaturę modyfikowaną innymi poleceniami),
- polecenie **`.TEMP <lista temperatur>`** powoduje powielenie aktualnych symulacji dla każdej z temperatur występujących na liście,
- przykłady:

```
.temp 10  .temp=10
.temp 10 20 30
.step temp list 10 20 30
```


Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

PSPICE

Rozdział 4. Elementy z dwoma wyprowadzeniami

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Zasady stosowanej składni w plikach symulacyjnych PSPICE:

- nawias trójkątny **< >** – obowiązkowy parametr,
- nawias kwadratowy **[ ]** – parametr opcjonalny,
- Pionowa kreska **|** rozdzielająca elementy wpisu – oznacza, że tylko jeden element z rozdzielonych symbolem **|** jest dopuszczalny.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Elementy z dwoma wyprowadzeniami – uwagi ogólne:

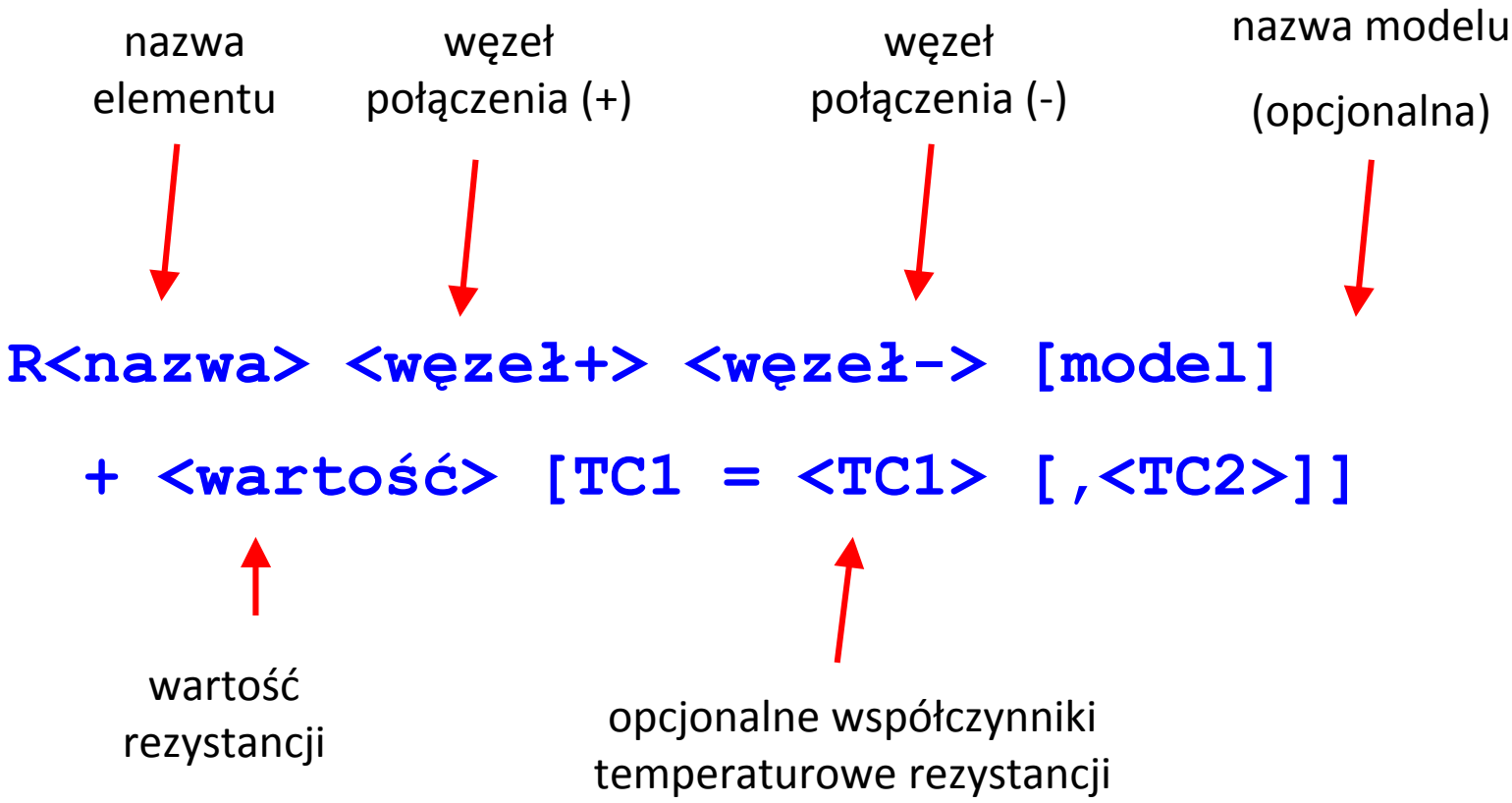
- zasada definiowania jest jak poniżej:

`Rnazwa węzeł+ węzeł- <wartość lub model>`

- napięcie mierzy się pomiędzy **`węzeł+`** a **`węzeł-`** ,
- jako prąd dodatni uważa się prąd wpływający do wyprowadzenia **`węzeł+`** .

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

PSPICE – Rezystor



W przypadku użycia modelu rzeczywista wartość rezystancji jest modyfikowana zgodnie z równaniem modelu!

PSPICE – Rezystor c.d.

Przykłady wstawienia rezystora do badanego obwodu:

`Rload out 0 10k` – rezystor o nazwie **`load`** włączony pomiędzy węzły **`out`** oraz **`0`** i wartości **`10kOhm`** ,

`R123 wypr12 12 my_res 22k` – rezystor o nazwie **`123`** włączony pomiędzy węzły **`wypr12`** oraz **`12`** , modelu **`my_res`** i wartości inicjalnej rezystancji równej **`22kOhm`** ,

W tym przypadku, aby doszło do prawidłowej analizy musi być powołany model rezystora o nazwie **`my_res`** model.

Przykład deklaracji modelu rezystora:

```
.MODEL my_res RES (R=3 TC1=0.01 TC2=0.1)
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Przykłady wyliczenia wartości rezystancji wstawionego elementu:

```
Rload out 0 10k    -> r=10k

Rload out 0 10k TC1=0.01 TC2=0.001 ->

r=10k*(1+TC1*(T-Tnom)+TC2*(T-Tnom)^2)
```

gdzie: **T** - temperatura symulacji,
domyślnie = 27°C,
można ją ustawić na inną wartość np. poleceniem **.temp=10**
Tnom=27°C – temperatura nominalna

PSPICE – Rezystor c.d.


```
Rload out 0 my_res 10k
.MODEL my_res RES (R=2 TC1=0.001 TC2=0.0001)
 ->
r = 10k*R*(1+TC1*(T-Tnom)+TC2*(T-Tnom)^2)
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

```
.MODEL <nazwa_modelu> [AKO: <nazwa_modelu_odniesienia>]
+ <typ_modelu>
+ ([<nazwa_parametru> = <wartość> [specyfikacja_tolerancji]]
+ [T_MEASURED=<wartość>] [[T_ABS=<wartość>] |
+ [T_REL_GLOBAL=<wartość>] | [T_REL_LOCAL=<wartość>]])
```

Gdzie:

<nazwa_modelu> - nazwa modelu nadana przez użytkownika,

[AKO: <nazwa_modelu_odniesienia>] - nazwa modelu odniesienia

<typ_modelu> - typ modelu, jedna z wartości z listy dostępnych modeli, modele dostępne to m.in. : **CAP, CORE, D, GASFET, IND, ISWITCH, LPNP, NIGBT, NJF, NMOS, NPN, PJF, PMOS, PNP, RES, TRN, VSWITCH**,

[<nazwa_parametru> = <wartość> [specyfikacja_tolerancji] - lista wartości parametrów modelu wraz z ewentualnymi tolerancjami

PSPICE – Modele elementów, inf. ogólne c.d.


```
.MODEL <nazwa_modelu> [AKO: <nazwa_modelu_odniesienia>]
+ <typ_modelu>
+ ([<nazwa_parametru> = <wartość> [specyfikacja_tolerancji]]
+ [T_MEASURED=<wartość>] [[T_ABS=<wartość>] |
+ [T_REL_GLOBAL=<wartość>] | [T_REL_LOCAL=<wartość>]])
```

Gdzie:

[T_MEASURED=<value>] - temperatura dla którego dokonano pomiaru parametrów modelu, jeśli podana nadpisuje wartość TNOM dla danego elementu,

[[T_ABS=<value>] | [T_REL_GLOBAL=<value>] | [T_REL_LOCAL=<value>]] – parametry modyfikujące temperaturę danego elementu w stosunku do wartości ogólnych.

Parametry: **T_MEASURED, T_ABS, T_REL_GLOBAL, T_REL_LOCAL** mają identyczne znaczenie dla wszystkich modeli i nie będą dalej dla szczegółowych modeli powtarzane przy ich omawianiu.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Model rezystora

Parametr modelu	Opis	Jednostka	Wartość domyślna
R	mnożnik		1.0
TC1	współczynnik temperaturowy liniowy	$1/^0C$	0.0
TC2	współczynnik temperaturowy kwadratowy	$1/^0C^2$	0.0
TCE	współczynnik temperaturowy ekspotencjalny	$\%/^0C$	0.0

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Model rezystora c.d.

Jeśli rezystor ma model i podano parametr TCE wówczas jego rezystancja jest równa:

$$r =< \text{wartość} > * R * 1.01^{TCE(T-TNOM)}$$

Jeśli rezystor ma model i parametr TCE nie jest podany wówczas jego rezystancja jest równa:

$$r =< \text{wartość} > * R * \left(1 + TC1(T - TNOM) + TC2(T - TNOM)^2\right)$$

Szum rezystora to szum biały o gęstości widmowej mocy szumów równej:

$$i^2 = 4kT / r$$

Gdzie: *k* jest stałą Boltzmana a *T* temperaturą bezwzględną (wyjątkowo w tym miejscu w Kelwinach)

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Kondensator

nazwa elementu

węzeł połączenia (+)

węzeł połączenia (-)

nazwa modelu (opcjonalna)

```
C<nazwa> <węzeł+> <węzeł-> [model]
+ <wartość> [IC=<wartość_początkowa>]
```

wartość pojemności

opcjonalna wartość napięcia początkowego na kondensatorze

W przypadku użycia modelu rzeczywista wartość pojemności jest modyfikowana zgodnie z równaniem modelu!

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Kondensator c. d.

Parametr modelu	Opis	Jednostka	Wartość domyślna
C	mnożnik		1.0
TC1	współczynnik temperaturowy liniowy	$1/^0C$	0.0
TC2	współczynnik temperaturowy kwadratowy	$1/^0C^2$	0.0
VC1	współczynnik napięciowy liniowy	1/V	0.0
VC2	współczynnik napięciowy kwadratowy	$1/V^2$	0.0

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Równanie modelu kondensatora:

$$c =< \text{wartość} > *C*\left(1+TC1(T-TNOM)+TC2(T-TNOM)^2\right)$$
$$*\left(1+VC1*V+VC2*V^2\right)$$

Gdzie: V jest napięciem występującym na okładkach kondensatora.

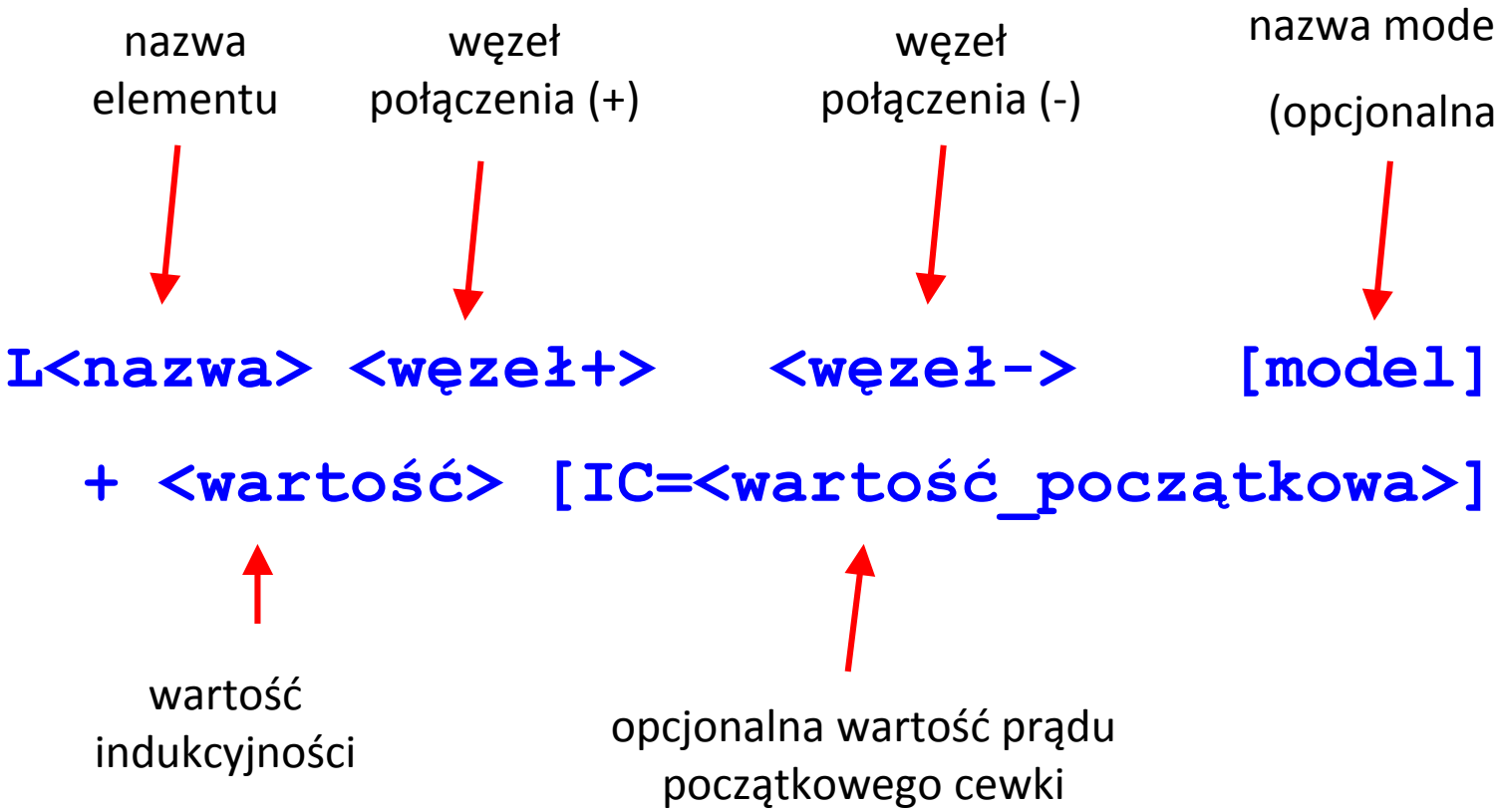
Kondensator jest elementem bezszumnym.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Cewka



W przypadku użycia modelu rzeczywista wartość indukcyjności jest modyfikowana zgodnie z równaniem modelu!

PSPICE – Cewka c. d.

Parametr modelu	Opis	Jednostka	Wartość domyślna
L	mnożnik		1.0
TC1	współczynnik temperaturowy liniowy	$1/^0C$	0.0
TC1	współczynnik temperaturowy kwadratowy	$1/^0C^2$	0.0
IL1	współczynnik prądowy liniowy	1/A	0.0
IL2	współczynnik prądowy kwadratowy	$1/A^2$	0.0

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Równanie modelu cewki:

$$l =< \text{wartość} > * L * \left(1 + TC1(T - TNOM) + TC2(T - TNOM)^2\right) * \left(1 + IL1 * I + IL2 * I^2\right)$$

Gdzie: ***I*** jest prądem wpływającym do cewki.

Cewka jest elementem bezszumnym.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Niezależne źródło napięciowe i prądowe


```
V<nazwa> <węzeł+> <węzeł->
+ [ [DC] <wartość> ]
+ [ AC <wartość_amplitudy>
+ [wartość_fazy]]
+ [specyfikacja_transient]
```

Źródło napięciowe i prądowe jest definiowane identycznie, jedyną różnicą jest litera definicji **V** lub **I**.

Przykłady:

```
IBIAS 13 0 DC 2.3mA AC 1
VPULSE 1 0 PULSE(-1mA 1mA 2ns 2ns 2ns 50ns 100ns)
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Niezależne źródło V, I, c. d.

<węzeł+> <węzeł-> - węzły połączeniowe,

[[DC] <wartość>] – wydajność źródła dla analizy stałoprądowej (DC),

[AC <wartość_amplitudy>

+[wartość_fazy]] – parametry źródła dla analizy małosygnałowej częstotliwościowej (AC), amplituda w V lub A oraz opcjonalnie faza w stopniach,

[specyfikacja_transient] - specyfikacja przebiegu czasowego dla analizy czasowej (TRAN).

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Niezależne źródło V, I, c. d.

- można podać dowolną ilość specyfikacji analiz – nawet zero, wówczas zastaną zastosowane wartości domyślne równe zero dla każdej analizy t.j. AC, DC i TRAN,
- słowo kluczowe DC jest opcjonalne i jego występowanie nic nie zmienia,
- dla analizy AC faza jest w stopniach,
- dla analizy TRAN tylko jeden typ specyfikacji może być podany,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

[specyfikacja_transient] – typ przebiegu dla analizy czasowej (**TRAN**) wraz z listą wartości jego parametrów,

Typ przebieg może przyjęć jedną z następujących wartości:

- **EXP** – dla przebiegu ekspotencjalnego,
- **PULSE** – dla przebiegu prostokątnego,
- **PWL** – dla przebiegu w postaci połączonych odcinków,
- **SFFM** – dla przebiegu harmonicznego zmodulowanego częstotliwościowo,
- **SIN** – dla przebiegu sinusoidalnego

PSPICE – Specyfikacja czasowa typu EXP

Format ogólny:

```
EXP (<v1> <v2> [td1] [tc1] [td2] [tc2])
```

Przykład:

```
VA 1 0 EXP(1 2 1 .2 3 .3)
```

Przy deklarowaniu pobudzenia EXP jak i każdego innego czasowego można w nawiasie podać mniej parametrów, wówczas, pozostałe (nie podane) przyjmują wartości domyślne.

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Specyfikacja czasowa typu EXP, c. d.

Parametr	Opis	Jednostka	Wartość domyślna
<v1>	Wartość początkowa	V lub A	brak
<v2>	Wartość szczytowa		brak
[td1]	Opóźnienie do zbocza narastającego		0
[tc1]	Stała czasowa zbocza narastającego		TSTEP
[td2]	Opóźnienie do zbocza narastającego		td1 + TSTEP
[tc2]	Stała czasowa zbocza narastającego		TSTEP

Parametry **TSTEP** i **TSTOP** są parametrami ustalanymi przy wywoływaniu analizy **TRAN**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Specyfikacja czasowa typu EXP, c. d.

Dla czasu od 0 do **td1**:

$$v = v1$$

Dla czasu od **td1** do **td2**:

$$v = v1 + \left(v2 - v1\right)\left(1 - e^{-\left(TIME - td1\right)/tc1}\right)$$

Dla czasu od **td2** do **TSTOP**:

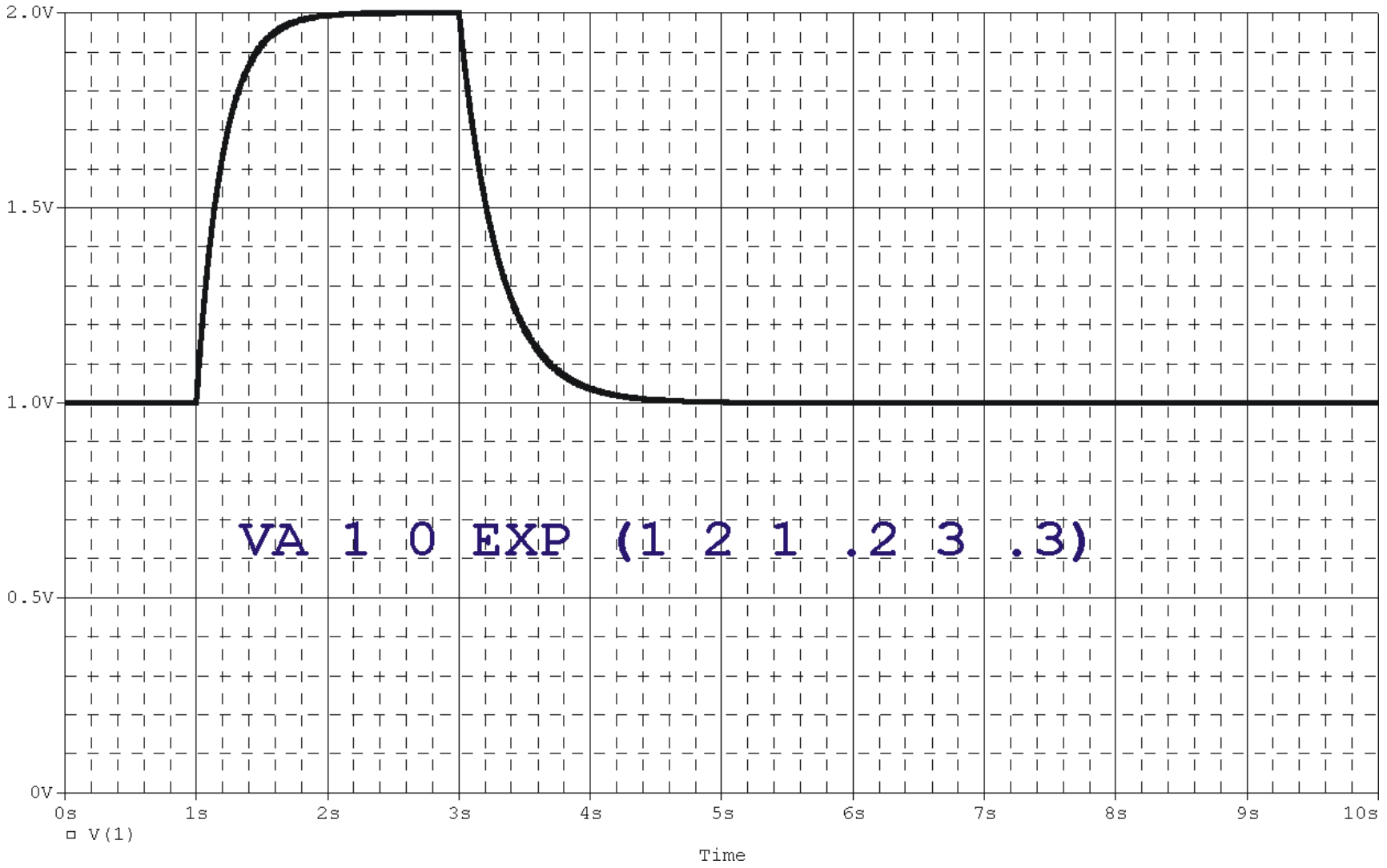
$$v = v1 + \left(v2 - v1\right)\left[\left(1 - e^{-\left(TIME - td1\right)/tc1}\right) - \left(1 - e^{-\left(TIME - td2\right)/tc2}\right)\right]$$

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Specyfikacja czasowa typu EXP, c. d.



PSPICE – Specyfikacja czasowa typu PULSE

Format ogólny:

```
PULSE (<v1> <v2> [td] [tr] [tf] [pw] [per])
```

Przykład:

```
Vpul 1 0 PULSE(1A 3A 2sec .15sec .14sec .7sec
+ 2sec)
```

Pobudzenie czasowe **PULSE** daje przebieg o wartości początkowej **v1**, który trwa przez **td** sekund a następnie zmienia się liniowo do wartości **v2** w czasie **tr** sekund i pozostaje stałe przez czas **pw** sekund aby ponownie liniowo zmienić wartość do poziomu **v1** w czasie **tf** sekund, następnie pozostaje w stanie **v1** przez czas **[per-(tr+pw+tf)]** sekund a następnie cykl jest powtarzany co **per** sekund z wyłączeniem odcinka początkowego **td**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Specyfikacja czasowa typu PULSE, c. d.

Parametr	Opis	Jednostka	Wartość domyślna
`<v1>`	Wartość początkowa	V lub A	brak
`<v2>`	Wartość końcowa	V lub A	brak
`[td]`	Czas opóźnienia	sec	0
`[tr]`	Czas narastania	sec	TSTEP
`[tf]`	Czas opadania	sec	TSTEP
`[pw]`	Czas trwania	sec	TSTOP
`[per]`	Okres przebiegu	sec	TSTOP

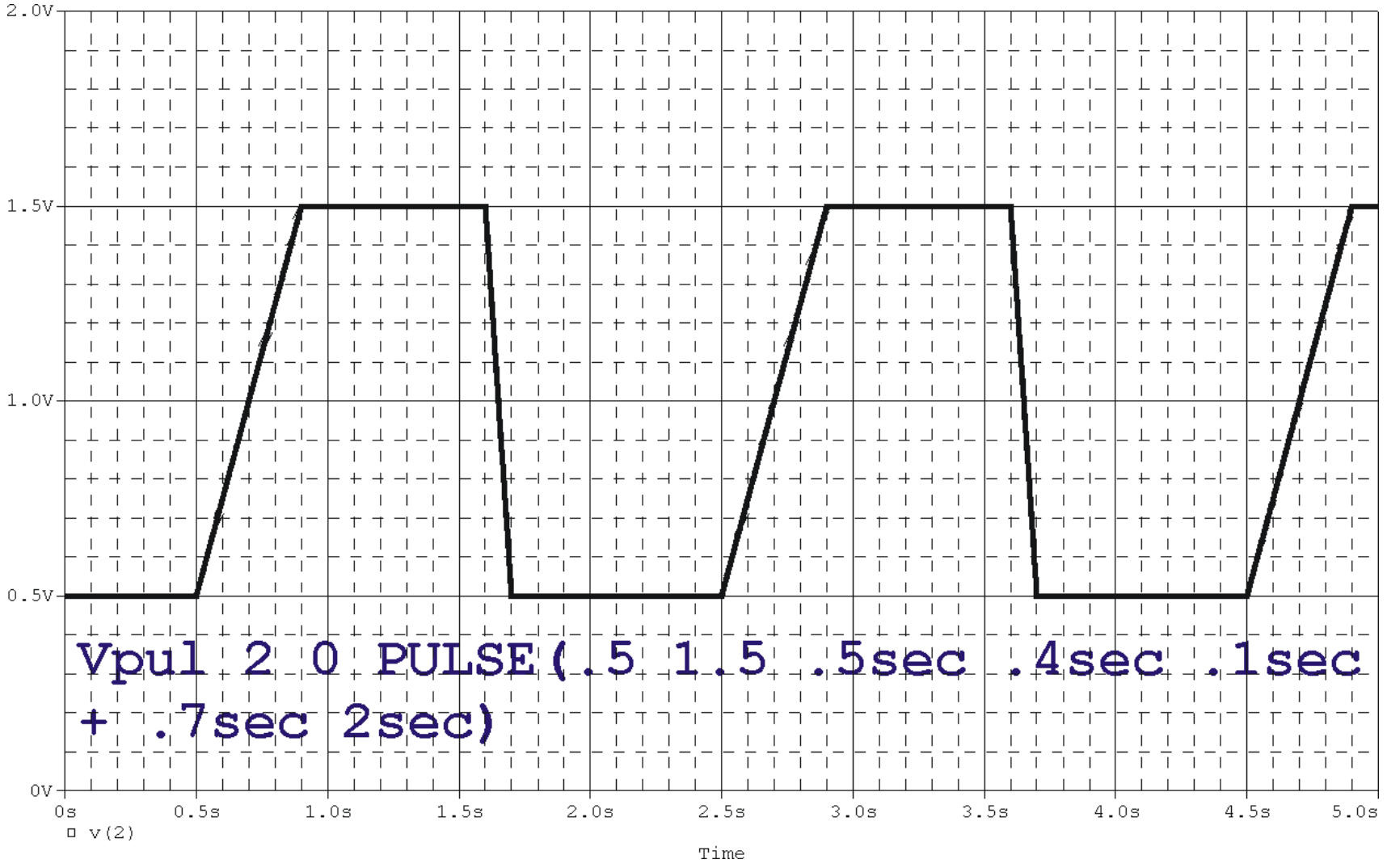
Parametry **`TSTEP`** i **`TSTOP`** są parametrami ustalanymi przy wywoływaniu analizy **`TRAN`**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Specyfikacja czasowa typu PULSE, c. d.



PSPICE – Specyfikacja czasowa typu SIN

Format ogólny:

```
SIN (<voff> <vampl> [freq] [td] [df] [phase])
```

Przykład:

```
Vsin 1 0 5 SIN(2 2 5Hz 1sec 1 30)
```

Pobudzenie czasowe **SIN** jest tłumionym przebiegiem sinusoidalnym.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

PSPICE – Specyfikacja czasowa typu SIN, c. d.

Parametr	Opis	Jednostka	Wartość domyślna
<voff>	Wartość przesunięcia	**V lub A**	**brak**
<vamp>	Amplituda	**V lub A**	**brak**
[freq]	Częstotliwość przebiegu	**Hz**	**1/TSTOP**
[td]	Czas opóźnienia	**sec**	**0**
[df]	Współczynnik tłumienia	**1/sec**	**0**
[phase]	Faza początkowa	**stopnie**	**0**

Parametry **TSTEP** i **TSTOP** są parametrami ustalanymi przy wywoływaniu analizy **TRAN**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Dla czasu od 0 do **td**:

$$v = voff + vampl * \sin\left(2\pi * phase / 360\right)$$

Dla czasu od **td** do **TSTOP**:

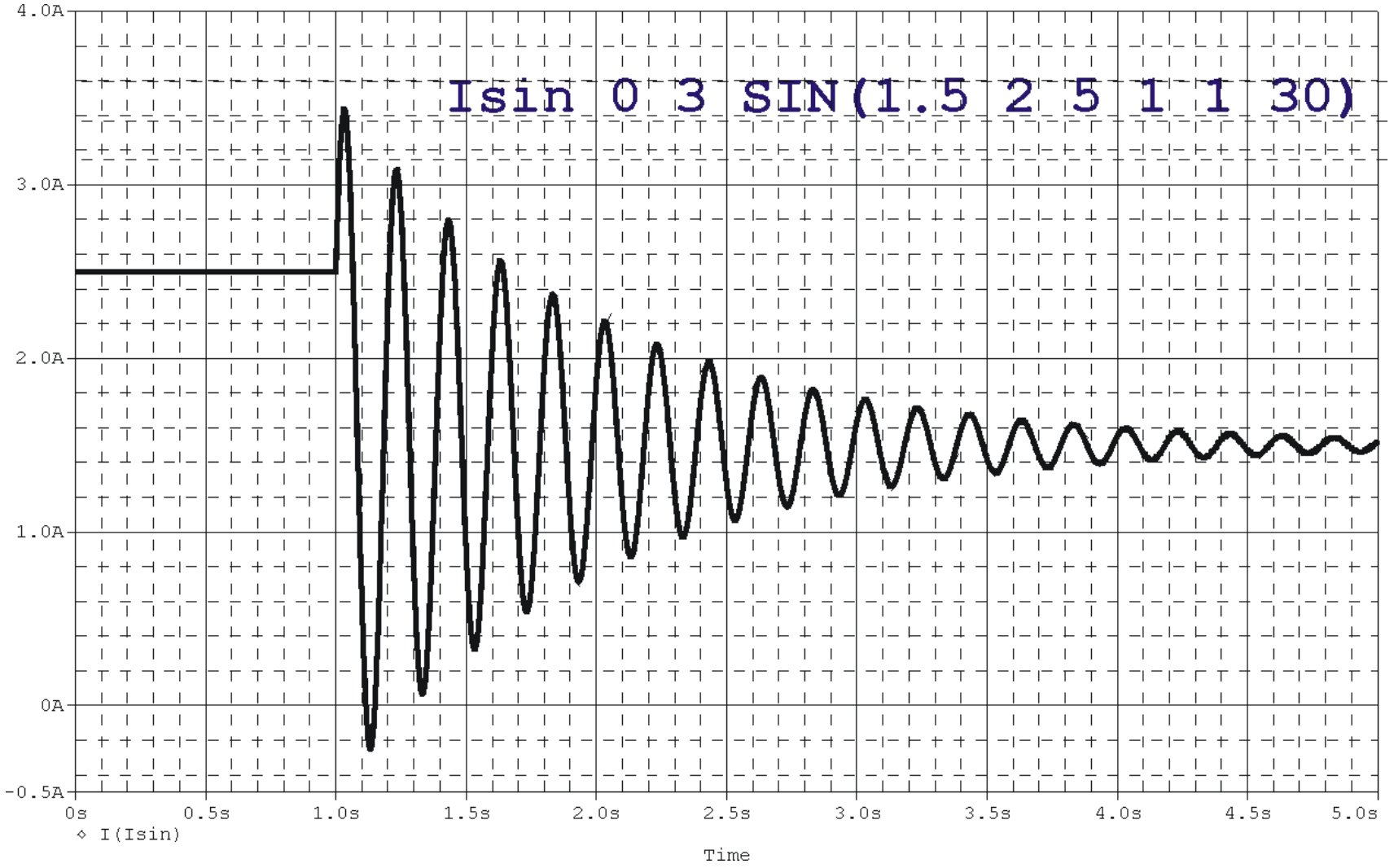
$$v = voff + vampl \cdot \sin\left(2\pi\left(freq \cdot (TIME - td) + phase / 360\right)\right) \cdot e^{(TIME - td) \cdot df}$$

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Specyfikacja czasowa typu SIN, c. d.



PSPICE

Rozdział 5. Wyprowadzenie wyników symulacji

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia PSPICE służące do określania miejsca umieszczania wyników:

- **`.PROBE`** - plecenie przekazywania wyników do postprocesora graficznego,
- **`.PRINT`** - polecenie wydruku wyników do pliku wyjściowego „**`*.out`**”,
- **`.PLOT`** - polecenie wykreślenia wykresu tekstowego (ze znaków ASCII) w pliku wyjściowym „**`*.out`**”.

PSPICE – Sterowanie wyjściem, .PROBE

Format ogólny:

```
.PROBE[/CSDF][zmienna_wyjściowa]
```

Przykłady:

```
.PROBE
```

```
.PROBE V(3) V(2,3) V(R1) I(VIN) I(R2) IB(Q13)
+VBE(Q13)
```

```
.PROBE/CSDF
```

W pierwszym przykładzie wszystkie napięcia i prądy układowe zostaną zapisane do pliku „**`*.dat`**”. W drugim przykładzie zapisane zostaną tylko wybrane sygnały. W trzecim przykładzie zapisane zostaną wszystkie sygnały do pliku w formacie **`CSDF`**. Format ten daje większe pliki ale umożliwia przenoszenie ich do innych programów.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Sterowanie wyjściem, .PRINT

Format ogólny:

```
.PRINT <rodzaj_analizy> [zmienna_wyjściowa]
```

Przykłady:

```
.PRINT DC  V(3)  V(2,3) V(R1) I(VIN)

.PRINT NOISE INOISE ONOISE DB(INOISE)
+DB(ONOISE)
```

W pierwszym przykładzie wyszczególnione sygnały dla analizy **.DC** zostaną umieszczone w pliku „***.out**”. W drugim przykładzie zapisane zostaną wybrane sygnały dla analizy **.NOISE**.

Można wyszczególnić tylko jeden rodzaj analizy ale można podać kilka poleceń **.PRINT**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Sterowanie wyjściem, .PLOT

Format ogólny:

```
.PLOT <typ_analizy> [zmienna_wyjściowa]
+ ( [<dolny_limit>,<górny_limit>] )
```

Przykład:

```
.PLOT TRAN V(3) V(2,3) (0,5V) ID(M2) I(VCC) (-50mA,50mA)
```

Można wyszczególnić tylko jeden rodzaj analizy ale można podać kilka poleceń **.PLOT**. Można wyszczególnić max. 8 zmiennych dla jednego polecenia. Limity, jeśli są podane, oznaczają zakres osi Y dla wszystkich zmiennych stosowanych na tej osi.

Polecenie **.PLOT** lepiej jest zastąpić poleceniem **.PROBE** a wynik oglądać w postprocesorze graficznym.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Właściwości polecenia **.PLOT**:

- zakresy, jeśli są stosowane dotyczą wszystkich zmiennych tego samego rodzaju,
- dla analizy **AC** limity nie są stosowane,
- dla analizy **AC** oś Y jest zawsze logarytmiczna.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 6. Dostępne analizy obwodów

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

ANALIZY PODSTAWOWE

- analiza stałoprądowa (**DC**),
- analiza częstotliwościowa (**AC**),
- analiza czasowa (**TRAN**).

ANALIZY POCHODNE

- Dla każdej z powyższych analiz istnieją analizy pochodne tzn. bazujące na analizach podstawowych, np.:
- **DC -> TF**
- **AC -> NOISE**
- **TRAN -> FOUR**

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

PSPICE – Analiza stałoprądowa .DC

Analiza **.DC** wykonuje sparametryzowane obliczenie punktu pracy układu. Elementy inercyjne są pomijane (C -> rozwacie, L-> zwarcie), elementy nieliniowe zachowują nieliniowy charakter. Argumentem tej analizy jest zmieniany parametr (pierwszy lub oba) a wynikiem rozpływ stałych prądów w układzie oraz wartości stałych napięć w węzłach.

Analiza **.DC** występuje w 4 formach przemiatania parametrów analizy: **LIN**, **OCT**, **DEC** oraz **LIST**. Analiza może być zagnieżdżona tzn. zmiana parametru pierwszego jest wykonywana w pętli wewnętrznej i następuje częściej niż parametru drugiego.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza stałoprądowa .DC, c. d.

Format ogólny dla przemiatania liniowego (LIN):

```
.DC [LIN] <nazwa przemiatanej zmiennej>
+ <start> <koniec> <wartość inkrementu>
```

+ [specyfikacja zagnieżdżenia] - (takie same zasady jak dla podstawowego przemiatania)

Przykłady:

```
.DC VIN -.25  .25  .05
.DC LIN I2  5mA  -2mA  0.1mA
.DC VCE 0V 10V .5V  IB 0mA 1mA 50uA
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza stałoprądowa .DC, c. d.

Przemiatanymi zmiennymi mogą być:

Zmienna	Zapis	Znaczenie
Źródło	Nazwa niezależnego źródła napięciowego lub prądowego.	Podczas przemiatania zmienia się wydajność źródła.
Parametr modelu	Typ i nazwa modelu oraz nazwa parametru w nawiasie.	Zmieniany jest wyszczególniony parametr modelu. Nie można zmieniać niektórych parametrów modeli, są to: W i L dla tran. MOS, oraz parametry temperaturowe.
Temperatura	Słowo **TEMP**	Zmieniana jest bieżąca temperatura symulacji.
Parametr globalny	Słowo kluczowe **PARAM** i nazwa parametru.	Zmieniana jest wartość parametru. Wszystkie wartości zależne od tego parametru są wyliczane na nowo.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza stałoprądowa .DC, c. d.

Format ogólny dla przemiatania logarytmicznego (LOG i OCT):

```
.DC [DEC lub OCT] <nazwa przemiatanej zmiennej>
+ <start> <koniec> <liczba pkt. na dekadę lub oktawę>
[specyfikacja zagnieżdżenia]
```

- (takie same zasady jak dla podstawowego przemiatania)

Przykład:

```
.DC DEC NPN QFAST(IS) 1E-18 1E-14  5
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza stałoprądowa .DC, c. d.

Format ogólny dla przemiatania LIST:

```
.DC <nazwa przemiatanej zmiennej> [LIST]
+ <lista wartości>
```

+ [specyfikacja zagnieżdżenia] - (takie same zasady jak dla podstawowego przemiatania)

Przykład:

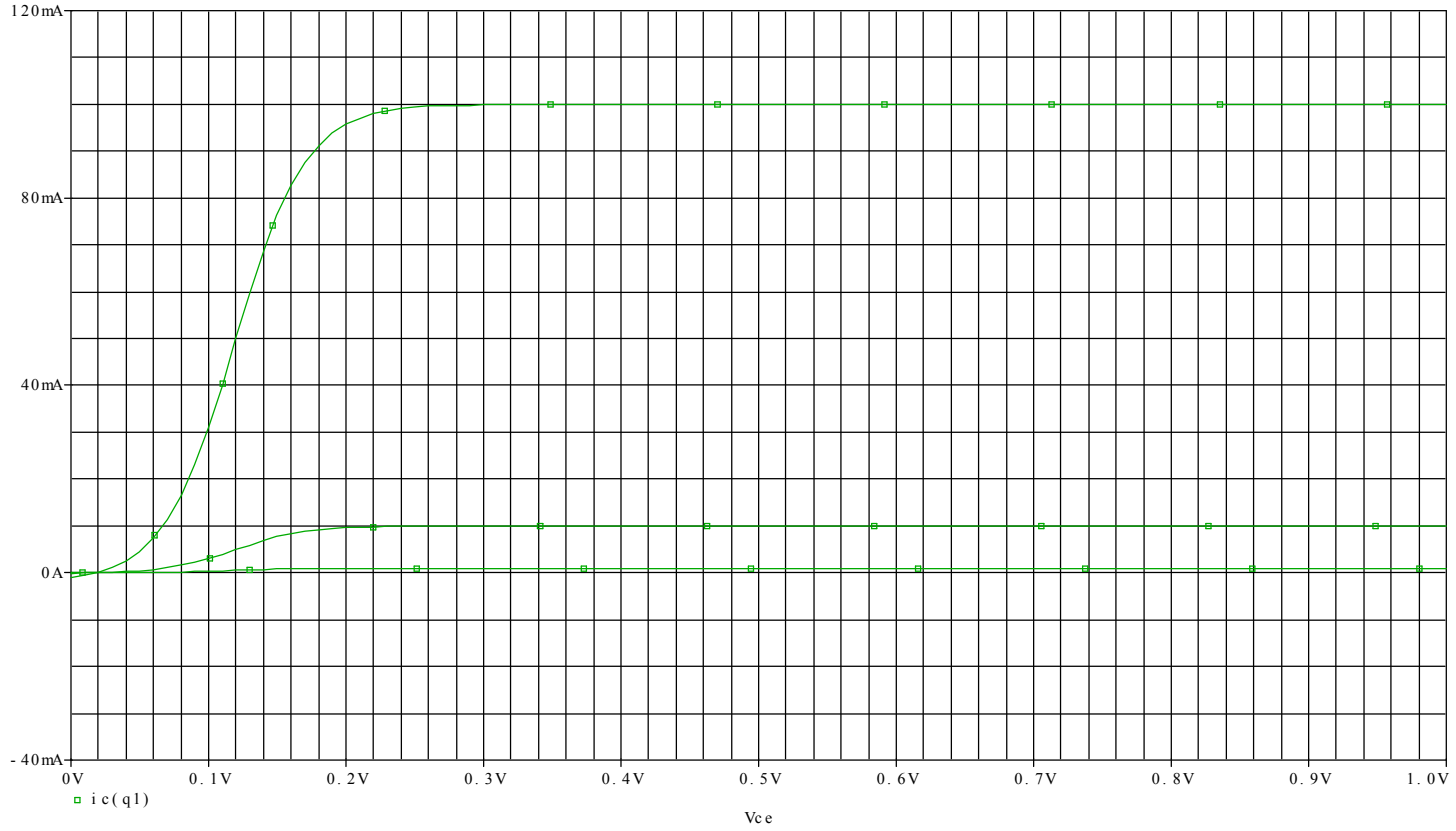
```
.DC VDD LIST 2 3 4 5 6
.dc Vce list 1 2 3 4 5 6 Ib list 10u 100u 1m
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza stałoprądowa .DC, c. d.




```
Qexp
Ib 0 B 1u
Vce c 0 1
Q1 c b 0 qnpn
.model qnpn npn ;(BF=200)
.dc Vce 0 10 .1 Ib list 10u 100u 1m
.probe
.end
```

PSPICE – Analiza stałoprądowa .OP

Format ogólny:

```
.op
```

Analiza wyznacza punkt pracy a szczegółowe wyniki umieszcza w pliku wyjściowym „`.out`”. Bez polecenia `.op` w pliku wyjściowym podawane są tylko wartości napięć węzłowych.

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza małosygnałowa .AC

Format ogólny:

```
.AC <typ przemiatania> <liczba punktów>
+ <częstotliwość początkowa>
+ <częstotliwość końcowa>
```

Przykłady:

`.ac lin 201 1k 2k` - 201 punków częstotliwościowych

`.ac dec 50 100Hz 1GHz` - 50 punków na każdą dekadę

`.ac oct 30 1 20` - 30 punków na każdą oktawę

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza małosygnałowa .AC, c. d.

Właściwości analizy:

- jest to analiza liniowa,
- najpierw obliczany jest punkt pracy,
- następnie linearyzowane są wszystkie elementy nieliniowe (zastępowane modelami małosygnałowymi),
- następnie obliczana jest charakterystyka częstotliwościowa,
- wejściami do układu są wszystkie niezależne źródła napięciowe i prądowe z niezerową specyfikacją AC
- wszystkie obliczone wielkości są liczbami zespolonymi.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza czasowa .TRAN

Format ogólny:

`.TRAN [/OP] <krok_wydruku> <czas_analizy>`

`+[nie_drukować [krok_obliczeniowy]][SKIPBP]`

Przykłady:

`.TRAN 1ns 100ns` - analiza do 100ns z krokiem wydruku 1ns

`.TRAN 1ns 100ns 0ns .1ns SKIPBP` – analiza do 100ns z krokiem obliczeniowym 0,1ns bez obliczania punktu pracy

Opcja **`/OP`** powoduje umieszczenie szczegółowych danych dotyczących punktu pracy w pliku wyjściowym „**`*.out`**”.

Opcja **`SKIPBP`** powoduje pominięcie obliczania punktu pracy, gdy użyte punkt pracy jest ustalony za pomocą wartości **`IC`** w deklaracjach kondensatorów i cewkach.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Właściwości analizy .TRAN:

- oblicza zachowanie się układu w czasie, argumentem dla analizy jest czas,
- analiza zawsze rozpoczyna się w czasie równym 0,
- analiza kończy się w czasie **`czas_analizy`**,
- przed rozpoczęciem analizy obliczany jest punkt pracy, który może być inny niż dla AC bo mogą być inne wartości specyfikacji pobudzeń,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza czasowa .TRAN, c. d.

- domyślny krok analizy dobierany jest w zależności od aktywności układu (zmian w układzie),
- domyślna wartość kroku analizy wynosi **`czas_analizy/50`**, ale w przypadku gdy nie ma w układzie elementów inercyjnych krok analizy jest równy **`krok_wydruku`**,
- analiza **`.TRAN`** ustawia wartości globalnych zmiennych **`TSTEP`** oraz **`TSTOP`** , które to są następnie używane do wyznaczania niektórych wartości domyślnych,
- **`TSTEP = krok_wydruku`**
- **`TSTOP = czas_analizy`**

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- dla parametru **krok_obliczeniowy** można zastosować funkcję:
  SCHEDULE(time1,val1,time2,valn...timen,valn),
- funkcja ta określa jaki krok obliczeniowy ma być użyty w kolejnych przedziałach czasowych,
- znaczenie jest następujące: od czasu **time1** użyj wartości **val1**, następnie od czasu **time2** użyj wartości **val2** i.t.d.

Przykład użycia funkcji:

```
.TRAN 1ns 90ns 0ns
+ {SCHEDULE(0,1ns,25ns,.1ns)}
```

PSPICE – Analiza Fouriera .FOUR

Format ogólny:

```
.FOUR <częstotliwość> [liczba_harmonicznych]
+ <zmienna_wyjściowa>
```

Przykłady:

`.FOUR 10kHz 20 v([inp])` - dekompozycja sygnału wokół częstotliwości podstawowej 10kHz

```
.FOUR 20 10 v([out1],[out2])
```

Analiza **`.FOUR`** wykonuje rozłożenie wyniku analizy **`.TRAN`** na składniki harmoniczne i umieszczenie wyników obliczeń w pliku wyjściowym „**`*.out`**”.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Właściwości analizy **`.FOUR`**:

- dla analizy **`.FOUR`** brane są wyniki analizy **`.TRAN`** z czasu od **`TSTOP-1/częstotliwość`** do końca analizy czasowej,
- analiza czasowa musi trwać conajmniej 1 okres częstotliwości rozkładu,
- jeśli nie podano wartości parametru **`liczba_harmonicznych`** obliczana jest składowa stała, składnik podstawowy oraz harmoniczne od 2 do 9 tej.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Przykład wykonania analizy .FOUR, polecenie:

`.FOUR 10kHz v([out])` Wyniki z pliku „*.out”:

```
 ****     FOURIER ANALYSIS                    TEMPERATURE =   27.000 DEG C
*******************************************************************************
DC COMPONENT =  -5.386184E-03
```

HARMONIC NO	FREQUENCY (HZ)	FOURIER COMPONENT	NORMALIZED COMPONENT	PHASE (DEG)	NORMALIZED PHASE (DEG)
1	1.000E+04	2.944E-01	1.000E+00	1.794E+02	0.000E+00
2	2.000E+04	7.813E-03	2.654E-02	8.750E+01	-2.714E+02
3	3.000E+04	1.255E-05	4.263E-05	-4.447E+01	-5.828E+02
4	4.000E+04	1.622E-05	5.511E-05	1.654E+02	-5.524E+02
5	5.000E+04	1.183E-05	4.019E-05	-1.667E+02	-1.064E+03
6	6.000E+04	7.385E-06	2.508E-05	-1.705E+02	-1.247E+03
7	7.000E+04	7.439E-06	2.527E-05	1.788E+02	-1.077E+03
8	8.000E+04	7.327E-06	2.489E-05	-1.716E+02	-1.607E+03
9	9.000E+04	5.683E-06	1.931E-05	-1.650E+02	-1.780E+03

```
     TOTAL HARMONIC DISTORTION =   2.654097E+00 PERCENT
```

PSPICE – Analiza .TF (funkcja przenoszenia)

Format ogólny:

`.TF <zmienna_wyjściowa> <źródło_wejściowe>`

Przykład:

`.TF V([out]) vin`

Analiza **`.TF`** jest pochodną analizy **`.DC`** i wyznacza stałoprądową, małosygnałową funkcję przenoszenia wyznaczoną wokół punktu pracy. Jako **`zmienna_wyjściowa`** mogą być użyte napięcia i prądy obwodu. Wyznaczane jest wzmocnienie liczone od **`źródło_wejściowe`** do **`zmienna_wyjściowa`** oraz rezystancje wyjściowa i wejściowa. Wyniki analizy **`.TF`** są umieszczane wyłącznie w pliku wyjściowym „**`*.out`**”.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza .TF (funkcja przenoszenia), c. d.

Polecenie:

```
.TF v([c]) va
```

Wynik z pliku „*.out”:

```
****     SMALL-SIGNAL CHARACTERISTICS

      V(c)/Va =  9.083E-02

      INPUT RESISTANCE AT Va =  1.100E+01

      OUTPUT RESISTANCE AT V(c) =  9.083E-01
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza wrażliwościowa .SENS

Format ogólny:

```
.SENS <zmienna_wyjściowa>
```

Przykład:

```
.SENS V([out]) v(2,1)
```

Analiza **.SENS** jest pochodną analizy **.DC** i wyznacza stałoprądowe czułości napięć i prądów w układzie na zmiany wszystkich wartości elementów i parametrów modeli. Jako **zmienna_wyjściowa** mogą być użyte napięcia węzłowe i prądy źródeł napięciowych. Wyniki analizy **.SENS** są umieszczane wyłącznie w pliku wyjściowym „***.out**”.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza wrażliwościowa .SENS, c. d.

Przykład zlecenia analizy wrażliwościowej, polecenie:

`.sens v([c])` Wyniki z pliku „**`*.out`**”:

```
DC SENSITIVITIES OF OUTPUT V(c)

          ELEMENT          ELEMENT          ELEMENT          NORMALIZED
           NAME             VALUE        SENSITIVITY      SENSITIVITY
                                        (VOLTS/UNIT)  (VOLTS/PERCENT)
           Rs             4.000E+03        0.000E+00        0.000E+00
           R1             1.000E+04        2.298E-04        2.298E-02
           Rc             4.000E+03       -9.724E-04       -3.890E-02
           RL             4.000E+03        0.000E+00        0.000E+00
           Vcc            1.000E+01        5.427E-01        5.427E-02
           Vs             0.000E+00        0.000E+00        0.000E+00
Q1
           RB             0.000E+00        0.000E+00        0.000E+00
           RC             0.000E+00        0.000E+00        0.000E+00
           BF             2.000E+02       -2.068E-04       -4.136E-04 .....
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza szumowa .NOISE

Format ogólny:

```
.NOISE V(<węzeł> [,<węzeł>])
+<nazwa_źródła_wejściowego> [wartość_interwału]
```

Przykłady:

```
.NOISE V([out]) vin 2
.NOSIE V(2,1) Iin
```

Analiza **.NOISE** jest pochodną analizy **.AC** i wyznacza wartości szumu pojawiające się na **V(<węzeł> [,<węzeł>])** oraz wartość szumu odniesioną do wejścia widzianego jako **nazwa_źródła_wejściowego**. Dla wykonania analizy **.NOISE** musi w układzie występować analiza **.AC**. Parametr **wartość_interwału** określa co który krok analizy **.AC** wykonywana jest analiza **.NOISE**, domyślnie parametr ten jest równy 1.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Właściwości analizy **`.NOISE`**:

- wyjściem dla analizy szumowej jest zawsze napięcie,
- jeśli wyjście podane jest dla dwóch węzłów po przecinku (np. **`v(1,2)`**) oznacza to, że jako wyjście brana jest różnica napięć występująca między tymi węzłami,
- **`nazwa_źródła_wejściowego`** oznacza nazwę niezależnego źródła napięciowego lub prądowego,
- symulator wyznacza: szum każdego elementu układu przeniesiony do wyjścia oraz szum całkowity na wejściu i wyjściu,

- wartość szumu na wyjściu wyznaczana jest dla każdej częstotliwości analizy i podawana jako gęstość widmowa napięcia skutecznego (RMS) w jednostkach **[V/Hz$^{1/2}$]**,
- wyznaczony jest szum całkowity na wyjściu, wzmocnienie od wejścia do wyjścia i szum odniesiony do wejścia (poprzez podzielenie),
- dla napięciowego źródła wejściowego jednostką szumu odniesioną do wejścia jest **[V/Hz$^{1/2}$]**,

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

PSPICE – Analiza szumowa .NOISE, c. d.

- dla prądowego źródła wejściowego jednostką szumu odniesioną do wejścia jest **[A/Hz$^{1/2}$]**,
- aby obliczyć wartość skuteczną napięcia szumów na wejściu lub wyjściu należy zastosować odpowiednią całkę:

$$V_{NOISE\_RMS} = \sqrt{\int_{f_{START}}^{f_{STOP}} V_{NOISE}^2(out)\,df}$$

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 7. Ustalenie punktu pracy i operacje na plikach

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Polecenie .IC, ustalenie punktu pracy

Format ogólny:

```
.IC < V(<węzeł> [,<węzeł>])=<wartość> >
```

```
.IC <I(<cewka>)=<wartość>>
```

Przykłady:

```
.IC V(2)=3.4 V(102)=0 V(3)=-1V I(L1)=2uAmp
```

```
.IC V(InPlus,InMinus)=1e-3 V(100,133)=5.0V
```

Polecenie ustala punkt pracy dla analizy **.AC** i analizy **.TRAN**. Do wyliczenia punktu pracy, wyspecyfikowane w poleceniu węzły są przyłączone do źródła napięciowego o rezystancji szeregowej równej 0.0002Ω i wartości podanej w poleceniu **.IC**. Po wyliczeniu punktu pracy, wyspecyfikowane węzły są odłączane od źródeł napięciowych.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Właściwości polecenia **`.IC`**:

- w przypadku równoczesnego wystąpienia polecenia **`.IC`** i **`.NODESET`** dla tego samego węzła, używane jest polecenie **`.IC`**,
- nie można ustalać niezerowej wartości napięcia dla cewek, gdyż dla wyliczania punktu pracy są one zwarciami,
- dla cewek można natomiast ustalić niezerowy prąd.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Polecenie .NODESET, inicjalny pkt. pracy

Format ogólny:

```
.NODESET < V(<węzeł> [,<węzeł>])=<wartość> >

.NODESET <I(<cewka>)=<wartość>>
```

Przykłady:

```
.NODESET V(2)=3.4 V(102)=0 V(3)=-1V I(L1)=2uAmp

.NODESET V(InPlus,InMinus)=1e-3 V(100,133)=5.0V
```

Polecenie pomaga w ustaleniu punktu pracy poprzez podanie początkowych wartości napięć węzłowych i prądów cewek – **które służą tylko jako punkt wyjściowy do obliczenia właściwego punktu pracy**. Można podać wszystkie napięcia węzłowe lub tylko niektóre. Można również podawać napięcie pomiędzy węzłami.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Operacje na plikach dostępne w PSPICE:

- **.INC** – plecenie włączenia pliku tekstowego,
- **.LIB** – plecenie wczytania biblioteki,
- **.SAVEBIAS** – zapisanie punktu polaryzacji,
- **.LOADBIAS** – załadowanie punktu polaryzacji.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

PSPICE – Operacje na plikach, polecenie .INC

Format ogólny:

```
.INC <nazwa_pliku>
```

Przykłady:

```
.INC "SETUP.CIR"

.INC SETUP.CIR

.INC "C:\LIB\VCO.CIR"
```

We wczytywanym pliku należy zachować składnię zgodną ze standardem SPICE. Plik wczytywany nie powinien mieć linii tytułu (chyba, że będzie zakomentowana). Wczytywanie może być zagnieżdżone do 4 poziomów.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Operacje na plikach, polecenie .LIB

Format ogólny:

```
.LIB [nazwa_pliku]
```

Przykłady:

```
.LIB

.LIB linear.lib

.LIB "C:\lib\bipolar.lib"
```

W przypadku braku podania nazwy pliku wczytywana jest biblioteka ”nom.lib”. Plik ten zawiera odniesienia do elementów ze standardowej biblioteki ORCAD. Z plikiem bibliotecznym związany jest plik indeksowy – wykorzystywany do szybkiego odnalezienia modelu w dużych plikach. Plik indeksowy tworzony jest automatycznie przez symulator.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

W pliku .lib można umieszczać jedynie:

- linie z komentarzem,
- polecenia **.MODEL** z definicjami parametrów elementów,
- polecenia definicji podukładów (**.SUBCKT**, **.ENDS**),
- definicje parametrów **.PARAM**,
- definicje funkcji **.FUNC**,
- polecenia **.LIB**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Operacje na plikach, polecenie .SAVEBIAS

Format ogólny:

```
.SAVEBIAS <"nazwa_pliku"> <[OP] [TRAN] [DC]> [NOSUBCKT]
+ [TIME=<wartość> [REPEAT]] [TEMP=<wartość>]
+ [STEP=<wartość>] [MCRUN=<wartość>] [DC=<wartość>]
+ [DC1=<wartość>] [DC2=<wartość>]
```

Przykłady:

`.SAVEBIAS "OPPOINT" OP` – zapisanie punktu pracy (AC i OP)

`.SAVEBIAS "TRANDATA.BSP" TRAN NOSUBCKT TIME=10u` – zapisanie punktu z analizy czasowej z czasu najbliższego 10us (ale nie mniejszego niż 10us).

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – .SAVEBIAS, c. d.

Przykłady c.d.:

`.SAVEBIAS "SAVETRAN.BSP" TRAN TIME=5n REPEAT` – zapisanie punktu co każde 5ns dla przebiegu obliczeniowego, poprzednie wyniki są nadpisywane.

`.SAVEBIAS "SAVEDC.BSP" DC MCRUN=3 DC1=3.5 DC2=100` – zapisanie punktu dla analizy **`.DC`** jeśli równocześnie spełnione są warunki: pierwsze przemiatanie DC równe 3.5, drugie przemiatanie DC równe 100 oraz 3 przebieg analizy Monte Carlo. Jeśli jest używane tylko jedno przemiatanie stałoprądowe można zamiast **`DC1`** użyć wyrażenia **`DC`**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Argumenty i opcje polecenia:

- nazwa pliku musi być w podwójnym cudzysłowie,
- **NOSUBCKT** – nie zapisuje napięć węzłowych i prądów cewek dla podukładów,
- **TIME=<wartość> [REPEAT]** używane dla punktów czasowych zapisów dla analizy czasowej, jeśli użyte jest **REPEAT** następuje nadpisywanie wyników więc tylko ostatnie wartości są dostępne,
- **TEMP=<wartość>** - definiuje temperatury dla których dokonywany jest zapis, i krok **STEP=<wartość>**,
- **MCRUN=<wartość>** - definiuje numer analizy Monte Carlo lub Worst Case dla której dokonywany jest zapis,

Argumenty i opcje polecenia c.d.:

- **`[DC=<wartość>] [DC1=<wartość>] [DC2=<wartość>]`**

– używa się do specyfikacji kroku analizy DC, który ma być zapisany.

Uwagi:

- jeśli nie użyto **`REPEAT`** zapis dokonywany jest dla wartości równej lub większej niż **`TIME`**, jeśli użyto **`REPEAT`** tylko ostatnio zapisany punkt istnieje bo poprzednie są nadpisywane,
- dla niezagnieżdżonych analiz stałoprądowych należy używać **`DC`** zamiast **`DC1`** i **`DC2`**,
- w pliku zapisywane są: ***napięcia węzłowe i prądy cewek***.

PSPICE – Polecenie .LOADBIAS

Format ogólny:

```
.LOADBIAS <"nazwa_pliku">
```

Przykłady:

```
.LOADBIAS "OPPOINT.OP"
```

Nazwa pliku musi być w cudzysłowie. Wczytywane są przybliżone wartości napięć węzłowych i prądów cewek. Wczytywany plik musi zawierać polecenie **.NODESET**. Zawartość pliku można wygenerować poleceniem **.SAVEBIAS** lub wykonać ręcznie.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Uwagi do stosowania poleceń .SAVEBIAS/.LOADBIAS:

- polecenia mogą być użyte do skrócenia czasu symulacji (punktu pracy) dla dużych układów,
- można też użyć polecenia do ustalenia stanu układów pamięciowych np. przerzutników,
- oba polecenia, jeśli użyte w jednym pliku wejściowym, nie mogą odnosić się do tego samego pliku,
- można użyć w.w. poleceń również w przypadku problemów ze zbieżnością obliczeń.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

PSPICE

Rozdział 8. Wstawienie niektórych elementów półprzewodnikowych

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Wstawienie diody półprzewodnikowej

Format ogólny:

```
D<nazwa> <węzeł+> <węzeł-> <nazwa_modelu>
[powierzchnia]
```

Przykłady:

```
DCLAMP 14  0 DMOD
```

```
D13 15 17 SWITCH 1.5
```

Opis modelu:

```
.MODEL <nazwa_modelu> D [parametry_modelu]
```

Anoda diody to **węzeł+** natomiast katoda to **węzeł-**. Parametr **powierzchnia** skaluje niektóre parametry modelu diody, jego domyślna wartość wynosi 1.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Wstawienie tranzystora bipolarnego

Format ogólny:

```
Q<nazwa> <kolektor> <baza> <emiter> [podłoże]
+<nazwa_modelu> [powierzchnia]
```

Przykłady:

```
Q1  14  2  13  PNPNOM

Q13  15  3   0  1  NPNSTRONG  1.5

Q7  VC  5  12  [SUB]  LATPNPDCLAMP  14   0  DMOD
```

Opis modelu:

```
.MODEL <nazwa_modelu> NPN [parametry_modelu]

.MODEL <nazwa_modelu> PNP [parametry_modelu]

.MODEL <nazwa_modelu> LPNP [parametry_modelu]
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Argumenty modelu:

`podłoże` - opcjonalny węzeł podłączenia podłoża umieszczany w nawiasach [], dla bocznych tranzystorów PNP

`powierzchnia` – względna powierzchnia elementu, skaluje niektóre parametry modelu, jego domyślna wartość wynosi 1.

PSPICE – Wstawienie tranzystora MOS

Format ogólny:

```
M<nazwa> <dren> <bramka> <źródło> <podłoże> <model>
+ [L=<value>] [W=<value>]
+ [AD=<value>] [AS=<value>]
+ [PD=<value>] [PS=<value>]
+ [NRD=<value>] [NRS=<value>]
+ [NRG=<value>] [NRB=<value>]
+ [M=<value>] [N=<value>]
```

Przykłady:

```
M16 17 3 0 0 PSTRONG M=2
M28 0 2 100 100 NWEAK L=33u W=12u
+ AD=288p AS=288p PD=60u PS=60u NRD=14 NRS=24 NRG=10
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Wstawienie tranzystora MOS, c. d.

Opis modelu:

```
.MODEL <nazwa_modelu> NMOS [parametry_modelu]
.MODEL <nazwa_modelu> PMOS [parametry_modelu]
```

Argumenty i opcje:

L, W – długość i szerokość kanału tranzystora, parametry te są modyfikowane i w modelu obliczana jest efektywna długość kanału. Można te parametry podać również w definicji modelu **.MODEL** lub poprzez parametry poleceniem **.OPTIONS**. Wartość podana przy wywołani elementu wypiera wartość w **.MODEL**, która to wypiera wartość z polecenia **.OPTIONS**. Jeśli nie podano **W** lub **L** wartość domyślna wynosi 100um.

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Wstawienie tranzystora MOS, c. d.

Argumenty i opcje c.d.:

AS, AD – powierzchnia obszaru źródła i drenu, wartość domyślna może być ustawiona w **.OPTIONS**, jeśli nie została ustawiona to wartość domyślna wynosi 0,

PS, PD – obwód obszaru źródła i drenu, wartość domyślna wynosi 0,

NRS, NRD, NRG, NRB – mnożniki rezystancji szeregowych wyprowadzeń tranzystora odpowiednio dla: źródła, drenu, bramki i podłoża, wartość domyślna wynosi 0, mnożony parametr to RSH,

M(NP) – mnożnik ilości równolegle połączonych tranzystorów MOS, wartość domyślna wynosi 1, **NP** oznacza to samo co **M**,

N(NS) – mnożnik długości kanału tranzystora, ważny tylko dla modeli MOS LEVEL=5, wartość domyślna wynosi 1, **NS** oznacza to samo co **N**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Wstawienie tranzystora MOS, c. d.

PSPICE obsługuje 7 poziomów modeli MOS:

- **LEVEL=1** – model Shiman-Hodges,
- **LEVEL=2** – model geometryczny,
- **LEVEL=3** – model półempiryczny,
- **LEVEL=4** – model BSIM,
- **LEVEL=5** – model EKV,
- **LEVEL=6** – model BSIM3 v2.0,
- **LEVEL=7** – model BSIM3 v3.1.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 9. Źródła sterowane

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Źródła sterowane napięciem, E i G

Format ogólny – podstawowe źródło sterowane:

```
E<nazwa> <węzeł+> <węzeł-> <węzeł_sterujący+>
+<węzeł_sterujący-> <wzmocnienie>
```

Przykłady:

```
E1 1 2 3 4 1
Gq8 3 4 5 6 10uS
```

E oznacza źródło napięciowe sterowane napięciem, **G** oznacza źródło prądowe sterowane napięciem. Dla **E** wzmocnienie jest w [V/V] dla **G** wzmocnienie wyrażone jest w [S].

PSPICE – Źródła sterowane napięciem, c. d.

Oprócz typowego źródła sterowanego można użyć następujących specyfikacji:

- **POLY** – deklaracja wielomianu sterującego,
- **VALUE** – deklaracja wyrażenia arytmetycznego sterującego wydajnością źródła,
- **TABLE** – deklaracja funkcji tabelarycznej,
- **LAPLACE** – deklaracja transformaty Laplace'a,
- **CHEBYSHEV** – deklaracja wielomianu Czebyszewa,
- **FREQ** – deklaracja funkcji częstotliwościowej.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe"*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Źródła sterowane napięciem, c. d.

Przykłady rozszerzone:

```
Ea 3 2 1 181 1.0

Eb 11 0 POLY(1) 12  0 0 230

Ec 100 101 POLY(2) 1 0  2 0  0.0 1.6 0.4 0.007

Epierwiastek   5   0 VALUE = {15V*SQRT(V(11,12))}

Et out 0 TABLE {V(A,B)} = (0,0) (15,3)

Elap out 0 LAPLACE {V(10)} = {10/(10+0.01*s)}

Elp out 0 FREQ {V(10)}=(0,0,0)(15kHz, 0,0)(16kHz -50,

+ 0) DELAY=33ms

Elp2 out 0 CHEBYSHEV {V(10)} = LP 900 1.0k .1dB 50dB
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Źródła sterowane prądem, H i F

Format ogólny – podstawowe źródło sterowane:

```
H<nazwa> <węzeł+> <węzeł->
+ <nazwa_napięciowego_źródła_sterującego>
<wzmocnienie>
```

Przykłady:

```
H1 1 2 vin 10
```

```
FSENSE 1 2 VSENSE 10.0
```

H oznacza źródło napięciowe sterowane prądem, **F** oznacza źródło prądowe sterowane prądem. Dla **H** wzmocnienie jest w [Ohm] dla **F** wzmocnienie wyrażone jest w [A/A].

PSPICE – Źródła sterowane prądem, c. d.

- w typowej sytuacji wydajność wyjściowa jest równa prądowi źródła sterującego pomnożonemu przez wzmocnienie,
- oprócz typowego źródła sterowanego można użyć specyfikacji **`POLY`** – deklaracja wielomianu sterującego,
- w takim przypadku wydajność określa wyrażenie wielomianu – szczegóły w instrukcji PSPICE.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 10. Opis hierarchiczny: podukłady i ich wstawienie

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja podukładu

Format ogólny:

```
.SUBCKT <nazwa> [węzły]
+ [OPTIONAL: < <węzeł_opcjonalny> = <węzeł_domyślny>>]
+ [PARAMS: < <nazwa_parametru> = <wartość>>]
+ [TEXT: < <nazwa> = <wartość_tekstowa>>]
...
.ENDS
```

Przykład:

```
.SUBCKT OPAMP 10 12 11 21 22
...
.ENDS
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja podukładu, c. d.

Przykłady c.d.:

```
.SUBCKT FILTR WE WY PARAMS: Fo=10kHz,
+ Pasmo=15kHz
...
.ENDS
.SUBCKT PLD I1 I2 I3 O1
+   PARAMS: M=0 IO=0
+   TEXT: FILE="PRG.JED"
...
.ENDS
.SUBCKT  LS7400 Y A B
+   OPTIONAL: DPWR=$N_PWR DGND=$N_DGND
+   PARAMS: M=0 IO=0
...
.ENDS
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja podukładu, c. d.

Argumenty i opcje:

[węzły] - lista węzłów (formalnych) łączących podukład z otoczeniem (może być pusta),

OPTIONAL – umożliwia specyfikację opcjonalnych węzłów połączeniowych,

PARAMS – umożliwia specyfikację opcjonalnych parametrów przekazywanych do podukładu,

TEXT – umożliwia specyfikację opcjonalnych parametrów tekstowych przekazywanych do podukładu (używane tylko w symulacji układów cyfrowych),

.ENDS – oznacza koniec definicji podukładu.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja podukładu, c. d.

Podukłady i ich używanie:

- przywołanie podukładu następuje poprzez użycie litery **X** jako typu wstawianego komponentu,
- przy powoływaniu podukładu liczba węzłów musi być taka sama jak w deklaracji,
- przywołanie podukładu powoduje zamianę nazw formalnych węzłów na nazwy aktualne (takie jak w przywołaniu),
- nie używa się węzła „0” na liście węzłów, ten węzeł jest globalną masą zawsze widzianą zarówno w układzie głównym jak i w podukładzie,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja podukładu, c. d.

- węzły opcjonalne mogą ale nie muszą być specyfikowane przy wywołaniu podukładu, wówczas użyte zostaną połączenia domyślne,
- węzły opcjonalne są bardzo przydatne do podłączania zasilania np. w bramkach cyfrowych,
- przywołanie (**`X`**) podukładu może być zagnieżdżone,
- deklaracja (**`.SUBCKT`**, **`.ENDS`**) podukładu nie może być zagnieżdżona,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja podukładu, c. d.

Wewnątrz definicji podukładu (czyli pomiędzy **.SUBCKT** i **.ENDS**) można używać:

- wstawiania elementów,
- polecenia **.IC** (inicjalny punkt pracy),
- polecenia **.NODESET** (przybliżony punkt pracy),
- polecenia **.MODEL** (definicja modelu),
- polecenia **.PARAM** (deklaracja parametru),
- polecenia **.FUNC** (deklaracje funkcji).

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja podukładu, c. d.

Podukłady, właściwości :

- modele, parametry i funkcje zdefiniowane w podukładzie są dostępne tylko w tym podukładzie,
- modele, parametry i funkcje zdefiniowane w głównym układzie są dostępne w tym układzie i wszystkich podukładach,
- nazwy węzłów, elementów i modeli są lokalne w podukładach i takie same nazwy mogą być używane w głównym układzie,
- nazwy węzłów, elementów w podukładach są widziane jako np. **`X1.Q13`** (element **`Q13`** w podukładzie **`X1`**) lub **`X4.123`** (węzeł **`123`** w podukładzie **`X4`**).

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08
KAPITAŁ LUDZKI

PSPICE – Wstawienie podukładu

Format ogólny:

```
X<nazwa> [węzły] <nazwa_podukładu> [PARAMS:
+ <<nazwa_parametu> = <wartość>>]
+ [TEXT: < <nazwa_par_text> = <text>>]
```

Przykłady:

```
Xwtornik WE WY VCC VEE WZMACNIACZ_12
Xfiltr  out  0  FILTR PARAMS: Fo=10kHz
```

Musi być taka sama liczba węzłów w układzie wstawianym jak w zdeklarowanym. Wstawianie podukładów może być zagnieżdżone ale nie może być zapętlone. Jeśli użyto opcji **`PARAMS:`** wówczas wyszczególnione wartości parametrów zostaną zamienione z wartości w definicji podukładu do wartości bieżących. Opcja **`TEXT:`** jest używana w symulacji cyfrowej.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08
KAPITAŁ LUDZKI

PSPICE – Wstawienie podukładu, c. d.

Przykłady zaawansowane wstawienia podukładu, najpierw jego deklaracja:

```
.SUBCKT  LS7400 Y A B
+  OPTIONAL: DPWR=$N_PWR DGND=$N_DGND
+  PARAMS: M=0 IO=0
...
.ENDS
```

Przykłady wstawienia podukładu:

X1 WY WE1 WE2 LS7400 - użyte są domyślne węzły połączenia do zasilania **$N_PWR** oraz **$N_DGND**,

X2 WY WE1 WE2 LS7400 PWR GND LS7400 - użyte są węzły połączenia do zasilania **PWR** oraz **GND**,

X3 WY WE1 WE2 LS7400 PWR LS7400 - użyty jest jeden i pierwszy węzeł opcjonalny połączenia do zasilania **PWR** oraz domyślny **$N_DGND**,

X4 WY WE1 WE2 LS7400 $N_PWR GND LS7400 – jeśli trzeba użyć drugiego węzła opcjonalnego wówczas należy wstawić również ten pierwszy.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

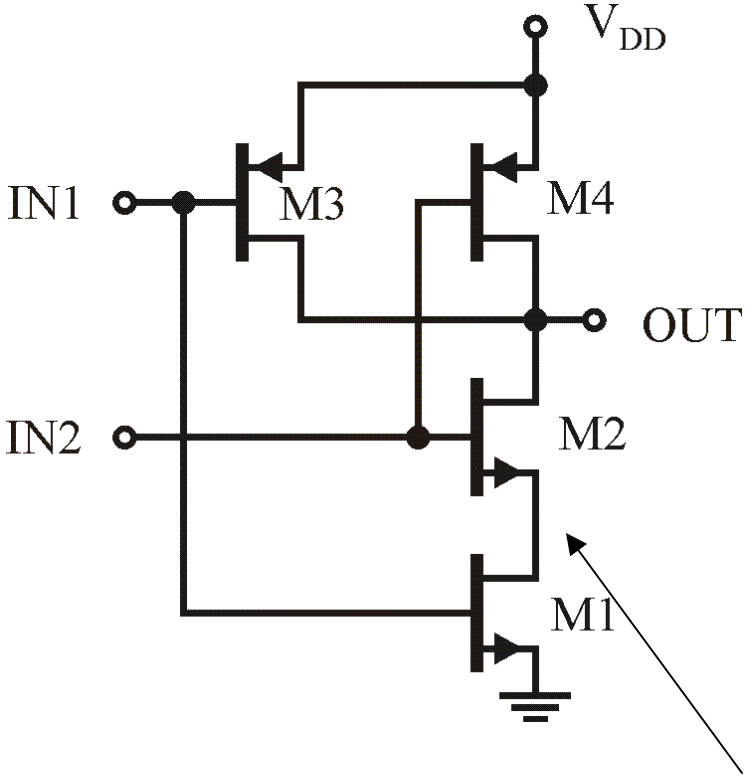
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Przykład – 2 x bramka NAND

Opis bramki umieszczony pomiędzy słowami kluczowymi **.sub** i **.ends** .

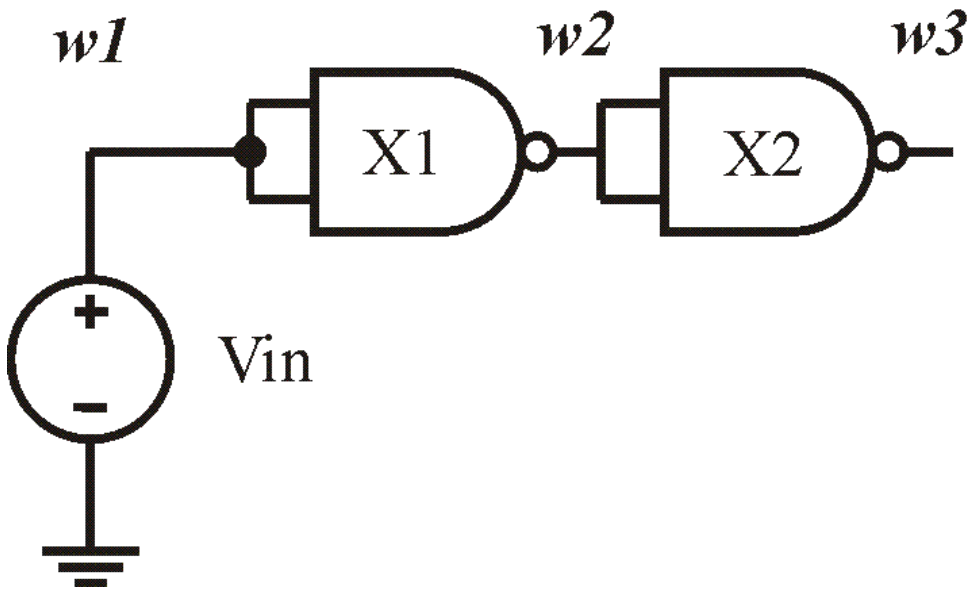
```
*Zdefiniowanie bramki NAND
*     NAZWA     węzły połączeniowe
*       |       ||||||||||||||||
.sub nand        in1 in2 out vdd
M1   10 in1    0    0 nfet W=0.9u L=0.6u
M2 out in2   10    0 nfet W=0.9u L=0.6u
M3 out in1 vdd vdd pfet W=0.9u L=0.6u
M4 out in2 vdd vdd pfet W=0.9u L=0.6u
.ends
```



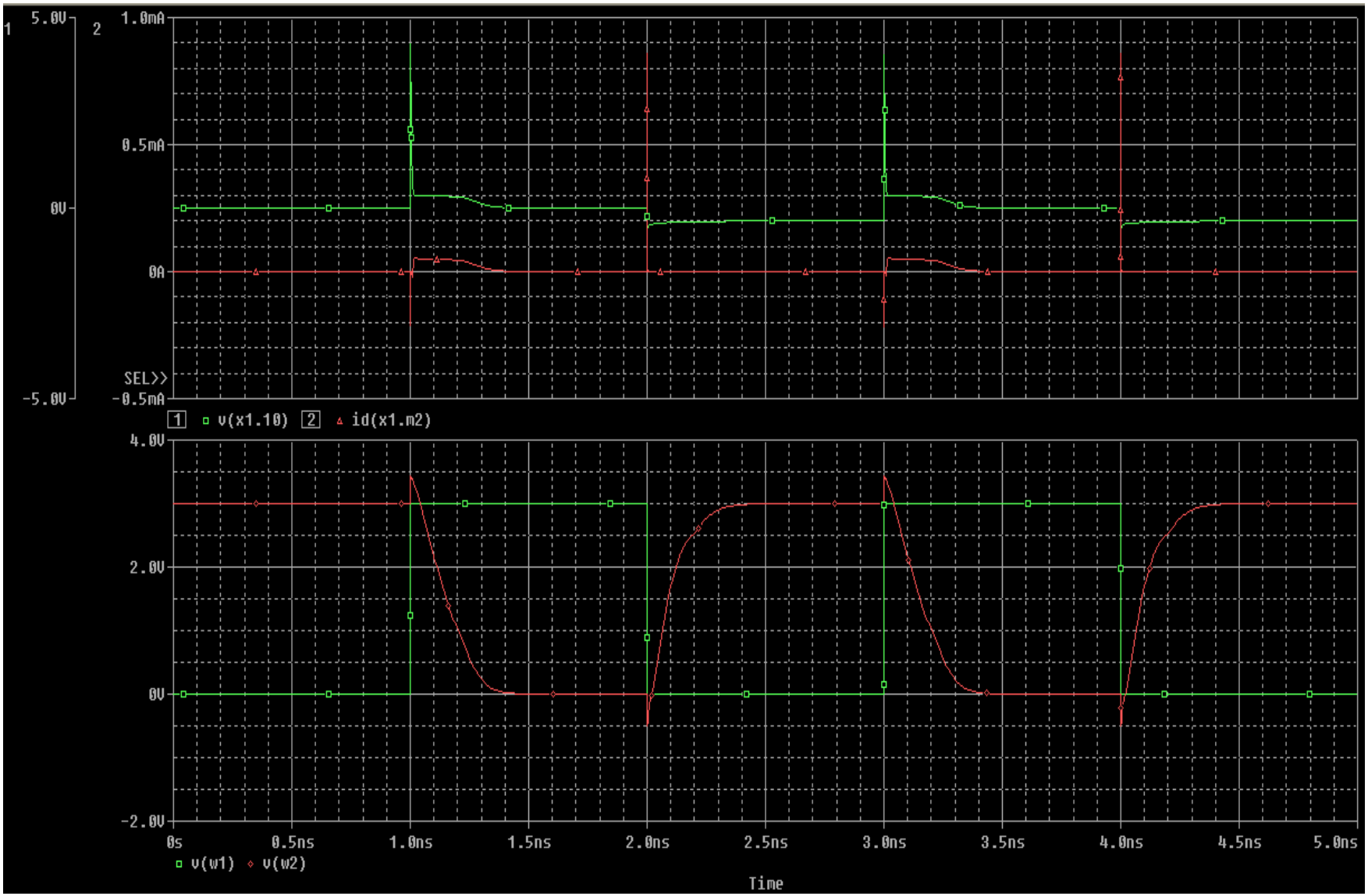
Węzeł wewnętrzny nazwany „**10**”

Pełny plik symulacyjny

```
*Zasilanie
Vdd vdd 0 3
*Sygnał wejściowy
Vin w1 0 dc 0 pulse(0 3 1n 1p 1p 1n 2n )
* Badany układ 2 bramek jako inwerterów
*  węzły połączeniowe    NAZWA
*  ||||||||||||||||      |
X1 w1   w1   w2   vdd    nand
X2 w2   w2   w3   vdd    nand
*Zdefiniowanie bramki NAND
*    NAZWA    węzły połączeniowe
*      |      ||||||||||||||||
.sub nand      in1 in2 out vdd
M1  10 in1   0   0 nfet W=0.9u L=0.6u
M2 out in2  10   0 nfet W=0.9u L=0.6u
M3 out in1 vdd vdd pfet W=0.9u L=0.6u
M4 out in2 vdd vdd pfet W=0.9u L=0.6u
.ends
*Analizy
.tran 5p 5n 0n 5p
.probe
.inc scn05mos59jmagic.lib
.end
```



PSPICE – Wstawienie podukładu – wyniki sym.



UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 11. Parametry i analiza parametryczna, operatory i funkcje

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja parametru .PARAM

Format ogólny:

```
.PARAM < <nazwa> = <wartość> >
.PARAM < <nazwa> = { <wyrażenie> } >
```

Przykłady:

```
.PARAM VCC = 12V, VEE = -12V
.PARAM BANDWIDTH = {100kHz/3}
.PARAM PI = 3.14159, TWO_PI = {2*3.14159}
.PARAM VNUM = {2*TWO_PI}
```

Nazwa nie może być nazwą predefiniowaną (**TEMP**, **VT**, **GMIN**,...). Wartość musi być podana za pomocą stałej lub wyrażenia. W wyrażeniu można używać stałych lub innych parametrów.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Deklaracja parametru .PARAM, c. d.

Właściwości i używanie parametrów:

- kolejność deklaracji parametrów nie ma znaczenia,
- można definiować parametry w podukładzie, będą one widoczne wówczas tylko w tym podukładzie,
- parametry można używać w miejsce stałych z następującymi wyjątkami: współczynniki **TC1** i **TC12** rezystorów w linii ich deklaracji, wartości specyfikacji **PWL** niezależnych źródeł napięciowych i prądowych, współczynniki **POLY** dla elementów **E**, **G**, **H**, **F**, nazwy węzłów układu, wartości parametrów analiz **TRAN**, **AC** i innych.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Format ogólny:

```
.STEP [LIN] <zmienna> <początek> <koniec> <inkrement>

.STEP [DEC|OCT] <zmienna> <początek> <koniec>
+<liczba_punktów_na_dekadę/oktawę>

.STEP <zmienna> LIST <lista wartości>
```

Polecenie **.STEP** wykonuje przemiatanie zmiennej dla wszystkich rodzajów analiz zleconych do wykonania w układzie. W plikach wyjściowych zapisywane są wyniki dla każdej wartości przemiatanej zmiennej. Przemiatanie może być wykonane liniowo (parametr **LIN**), logarytmicznie (parametr **OCT** lub **LOG**) lub można podać listę wartości (parametr **LIST**). Parametr **LIN** jest domyślny i można go nie podawać. Po użyciu parametru **LIST**, lista wartości musi być narastająca lub opadająca.

PSPICE – Analiza parametryczna .STEP, c. d.

Przykłady:

`.STEP VCE 0V 10V .1V` - liniowe przemiatanie VCE co 0.1V od 0V do 10V

`.STEP LIN Ibias 5mA -2mA 0.1mA` - liniowe przemiatanie Ibias

`.STEP RES RModel(R) 0.9 1.1 .02` - liniowe przemiatanie parametru R modelu RModel typu RES od wartości 0.9 do 1.1 co 0.02

`.STEP DEC NPN Qslow(IS) 1E-18 1E-14 10` - logarytmiczne przemiatanie parametru IS modelu Qslow typu NPN od wartości 1e-18 do 1e-14 z 10 punktami na każdą dekadę

`.STEP TEMP LIST 0 20 27 60 80 100` – lista temperatur pracy układu

`.STEP PARAM Fo 19.5kHz 20.5kHz 100Hz` - liniowe przemiatanie parametru CenterFreq.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opis	Zmienna	Znaczenie
Źródło	Nazwa niezależnego źródła napięciowego lub prądowego.	Podczas przemiatania zmienia się wydajność źródła.
Parametr modelu	Typ i nazwa modelu oraz nazwa parametru w nawiasie.	Zmieniany jest wyszczególniony parametr modelu.
Temperatura	Słowo **TEMP**	Zmieniana jest bieżąca temperatura symulacji.
Parametr globalny	Słowo kluczowe **PARAM** i nazwa parametru.	Zmieniana jest wartość parametru. Wszystkie wartości zależne od tego parametru są wyliczane na nowo.

PSPICE – Deklaracja funkcji .FUNC

Format ogólny:

```
.FUNC <nazwa>([argumenty]) {<ciało_funkcji>}
```

Przykłady:

```
.FUNC E(x) {exp(x)}

.FUNC DECAY(CNST) {E(-CNST*TIME)}
```

Polecenie **.FUNC** nie może redefiniować istniejących funkcji ani funkcji predefiniowanych w PSPICE. Argumenty funkcji nie mogą być węzłami układu. Ciało funkcji definiuje działanie deklarowanej funkcji i musi być wcześniej zdefiniowane (np. E w przykładach powyżej jest wykorzystywane w funkcji DECAY). Liczba argumentów definicji funkcji musi być równa liczbie w wywołaniu funkcji. Maksymalna liczba argumentów wynosi 10 minimalna 0. W ciele funkcji można używać parametrów, czasu **TIME**, innych funkcji (predefiniowanych i własnych), operatorów i zmiennej Laplacea ***s***.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Operatory

Operator	Działanie	Uwagi
+	dodawanie	
-	odejmowanie	
*	mnożenie	
/	dzielenie	
**	potęgowanie	
~	negacja	operator logiczny wykorzystywany w funkcji IF
|	OR	operator logiczny wykorzystywany w funkcji IF
^	XOR	operator logiczny wykorzystywany w funkcji IF
&	AND	operator logiczny wykorzystywany w funkcji IF

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Operatory, c. d.

Operator	Działanie	Uwagi
= =	równość	operator relacji wykorzystywany w funkcji IF
! =	nierówność	operator relacji wykorzystywany w funkcji IF
>	większe	operator relacji wykorzystywany w funkcji IF
>=	większe lub równe	operator relacji wykorzystywany w funkcji IF
<	mniejsze	operator relacji wykorzystywany w funkcji IF
<=	mniejsze lub równe	operator relacji wykorzystywany w funkcji IF

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Funkcje predefiniowane

Funkcja	Działanie	Uwagi
ABS(x)	$\|x\|$	
SQRT(x)	$x^{1/2}$	
EXP(x)	e^x	
LOG(x)	ln(x)	podstawą logarytmu jest e
LOG10(x)	log(x)	podstawą logarytmu jest 10
PWR(x,y)	$\|x\|^y$	
PWRS(x,y)	$+\|x\|^y$ $-\|x\|^y$	dla x>0 dla x<0
SIN(x)	sin(x)	gdzie x jest w radianach
ASIN(x)	$\sin^{-1}(x)$	gdzie wynik jest w radianach

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Funkcje predefiniowane, c. d.

Funkcja	Działanie	Uwagi
SINH(x)	sinh(x)	gdzie x jest w radianach
COS(x)	cos(x)	gdzie x jest w radianach
ACOS(x)	$\cos^{-1}(x)$	gdzie wynik jest w radianach
COSH(x)	cosh(x)	gdzie x jest w radianach
TAN(x)	tan(x)	gdzie x jest w radianach
ATAN(x) ARCTAN(x)	$\tan^{-1}(x)$	gdzie wynik jest w radianach
ATAN2(x,y)	$\tan^{-1}(y/x)$	gdzie wynik jest w radianach
TANH(x)	tanh(x)	gdzie x jest w radianach
M(x)	moduł	to samo co ABS(x) używane tylko w wyrażeniach Laplace'a

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Funkcje predefiniowane, c. d.

Funkcja	Działanie	Uwagi
P(x)	faza(x)	w stopniach używane tylko w wyrażeniach Laplace'a
R(x)	część rzeczywista z x	używane tylko w wyrażeniach Laplace'a
IMG(x)	część urojona x	używane tylko w analizach AC
DDT(x)	różniczka x względem czasu	używane tylko w analizach TRAN
SDT(x)	całka x względem czasu	używane tylko w analizach TRAN
TABLE(x_1,y_1,...)	wartość y z x	gdzie punkty $x_n y_n$ są połączone ze sobą liniami prostymi

PSPICE – Funkcje predefiniowane, c. d.

Funkcja	Działanie	Uwagi
MIN(x,y)	minimum z x i y	
MAX(x,y)	maksimum z x i y	
LIMIT(x,min,max)	min max x	jeśli x<min jeśli x>max x w pozostałych przypadkach
SGN(x)	+1 0 -1	jeśli x>0 jeśli x=0 jeśli x<0
STP(x)	1 0	jeśli x>0 0 w pozostałych przypadkach
IF(t,x,y)	x jeśli t jest prawdą y w przeciwnym przypadku	gdzie t jest operatorem logicznym lub relacyjnym z tabeli operatorowej

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE

Rozdział 12. Analiza Monte Carlo

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza Monte Carlo .MC

Format ogólny:

```
.MC <liczba_analiz> <typ_analizy> <zmienna_wyjściowa>
+ <funkcja> [OPCJE] [SEED=wartość]
```

Przykłady:

```
.MC 50 DC IC(Q7) YMAX LIST
.MC 20 AC VP(13,5) YMAX LIST OUTPUT ALL
```

Przy analizie MC zmianie podlega wartość parametru modelu dla którego podano wartość DEV lub LOT (lub oba). Pierwszy przebieg wykonywany jest dla wartości nominalnych wszystkich elementów, kolejne przebiegi wykonywane są dla wartości licznych zgodnie ze specyfikacjami DEV i LOT podanymi w modelach.

`liczba_analiz` - liczba wykonywanych analiz, dla drukowanych wyników górny limit wynosi 2000 dla pliku „**`*.out`**”, dla PROBE 400,

`typ_analizy` - specyfikuje typ analizy dla której będzie wykonywane Monte Carlo, ta analiza musi być zdefiniowana szczegółowo do analizy poza poleceniem MC,

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Format ogólny c.d.:

`zmienna_wyjściowa` – zmienna wyjściowa specyfikowana jak dla polecenia .PRINT

`funkcja` – specyfikuje operacje wykonywane na **`zmienna_wyjściowa`** i redukuje wynik do pojedynczej wartości, można wpisać jedną z następujących specyfikacji:

`YMAX` - znajduje wartość bezwzględną największej różnicy z każdym wykresie w stosunku do wykresu nominalnego,

`MAX` - znajduje wartość maksymalną w każdym każdym wykresie,

`MIN` - znajduje wartość minimalną w każdym wykresie,

`RISE_EDGE(val)` – znajduje pierwszy przypadek przekroczenia wykresu ponad wartość **`val`**, wykres musi mieć jeden lub więcej punktów poniżej **`val`** , wartość wyjściowa będzie pierwszym punktem wykresu przekraczającym tą wartość,

Format ogólny c.d.:

`FALL_EDGE(val)` – znajduje pierwszy przypadek przekroczenia wykresu poniżej wartości **`val`**, wykres musi mieć jeden lub więcej punktów powyżej **`val`** , wartość wyjściowa będzie pierwszym punktem wykresu przekraczającym tą wartość,

`<funkcja> [opcje]` nie mają znaczenia dla danych generowanych do postprocesora graficznego PROBE, mają one zastosowanie do pliku „**`*.out`**”. Wyjątkiem jest opcja **`<typ_wyjścia>`**.

`OPCJE`: lista możliwych do użycia opcji:

`LIST` - na początku każdego nowego przebiegu listuje bieżące parametry modeli,

Format ogólny c.d.:

OPCJE: lista możliwych do użycia opcji c.d.:

OUTPUT – specyfikuje które parametry są wpisywane do plików wyjściowych, jeśli nic nie podano tylko przebiegi nominale są przekazywane do wyjścia, przykłady:
OUTPUT ALL - wszystkie przebiegi są zapisywane do wyjścia,

OUTPUT FIRST N - N pierwszych symulacji przekazywane jest do wyjścia,
OUTPUT EVERY N - co N-ty przebieg jest kierowany do wyjścia,

SEED=n - definiuje wartość inicjalizującą generatora liczb losowych, wartość **n** musi być z przedziału 1 – 32767.

PSPICE – Analiza .MC specyfikacja tolerancji

Specyfikację tolerancji umieszcza się w modelu bezpośrednio za parametrem którego ma dotyczyć, format ogólny jest następujący:

```
[DEV [track&dist] <val> [%]]
[LOT [track&dist] <val> [%]]
```

gdzie:

DEV - indywidualne tolerancje parametru, każdy element ma inną wartość w zakresie dopuszczalnej tolerancji,

LOT - wspólne tolerancje parametrów identyczne co do wartości dla wszystkich elementów opisanych danym modelem,

Jeśli występuje znak **%** wówczas wyspecyfikowana tolerancja jest w procentach wartości nominalnej, w przeciwnym wypadku tolerancja jest w jednostkach jak dany parametr.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza .MC specyfikacja tolerancji, c. d.

track&dist – specyfikuje numer użytego generatora i dystrybucję (jeżeli nie ma być użyta domyślna UNIFORM), **track** podaje się jako numer w zakresie od 0 do 9 a **dist** jako jedna z wartości: UNIFORM, GAUS lub zdefiniowaną przez użytkownika.

UNIFORM – generuje równomiernie rozłożone wartości w zakresie podanych wartości, dystrybucja domyślna, dystrybucję domyślną na inną można zmienić poleceniem **.options**

GAUSS – generuje wartości w zakresie ±3sigma zgodnie z dystrybucją Gaussa, wartość specyfikowana w LOT/DEV jest równa 1sigma,

Przykłady specyfikacji tolerancji:

```
.MODEL CMOD CAP (C=1 DEV 5%)

.MODEL DLOAD D (IS=1E-9 DEV .5% LOT 10%)

.MODEL RTRACK RES (R=1 DEV/GAUSS 1% LOT/UNIFORM 5%)
```

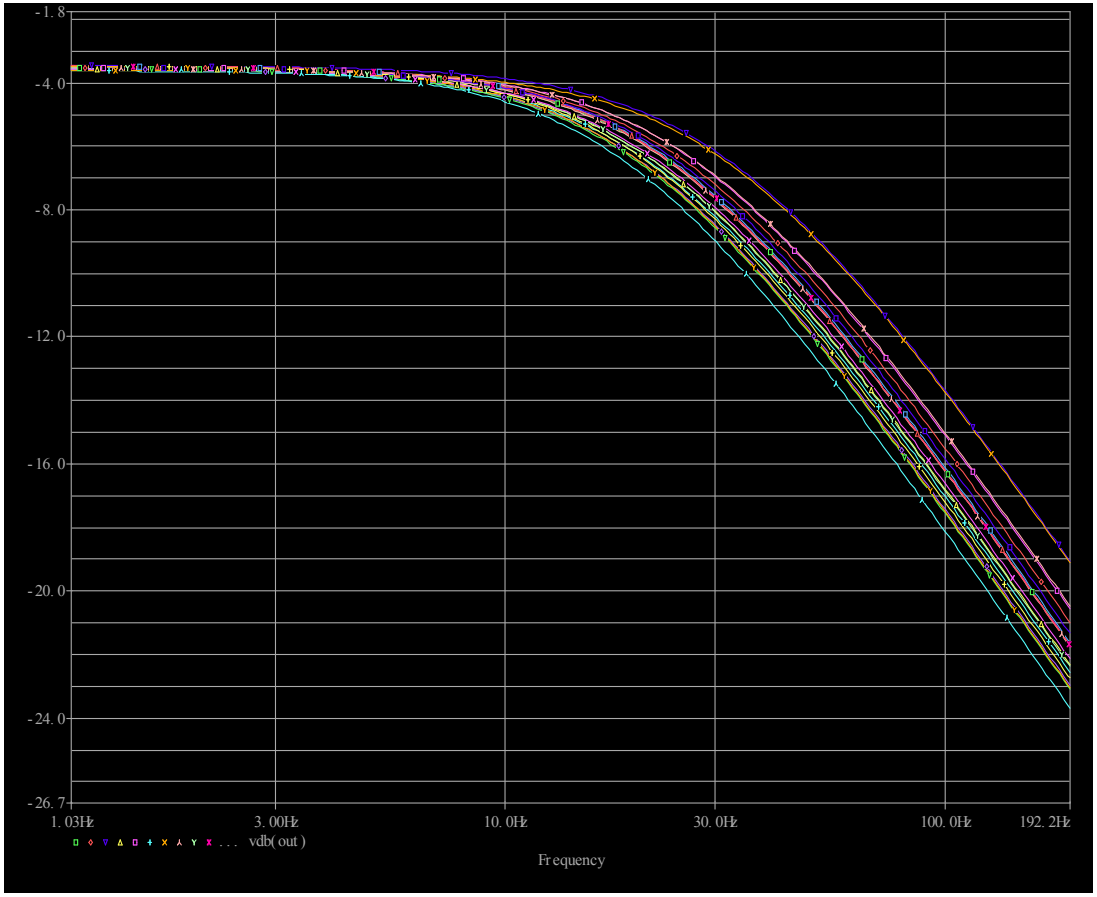
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Analiza .MC, prosty przykład


```
Monte Carlo analysis
Vin in 0 dc 0 ac 1
R1 in out rmod 10k
R2 out 0 rmod 20k
C1 out 0 cmod 1u
.dc Vin 0 10 .01
.ac dec 100 1 100k
.MC 20 AC V([out]) YMAX LIST
+ OUTPUT ALL
.model cmod cap (C=1 dev 1%
+ lot/gauss 10% vc1=0.01 lot=10%)
.model rmod res (R=1 dev=2%
+ lot=15% )
.probe
.end
```



W powyższym przykładzie wykonana jest 20 krotna analiza AC przy zmianie parametrów modeli losowanych w zadanych granicach.

PSPICE

Rozdział 13. Przykład symulacji wzmacniacza z tranzystorem bipolarnym w konfiguracji WE

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

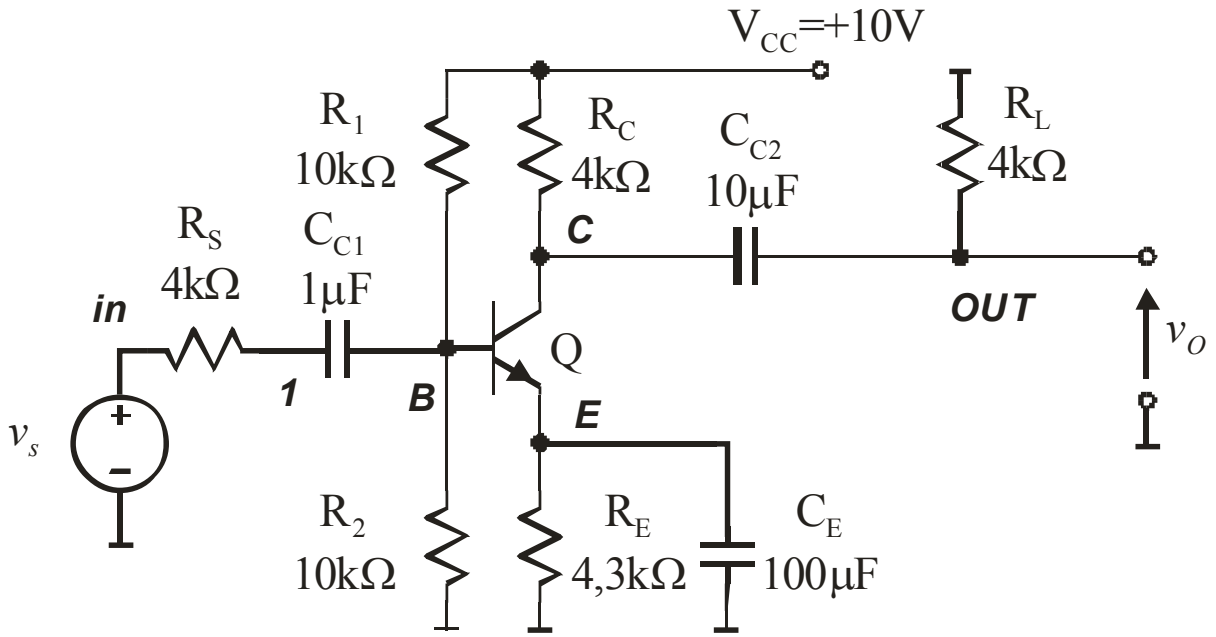
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Przykład symulacji wzmacniacza WE

Dane dla trazystora Q:
f_t=400MHz
C_μ=2pF
V_T =25mV (pot. termiczny)
r_O = ∞
β =200
V_{BE}=0,7V

Należy obliczyć:
- punkt pracy tranzystora Q ,
- wzmocnienie v_O/v_S w środku pasma przenoszenia,
- dolną częstotliwość graniczną f_{L3dB},
- górną częstotliwość graniczną f_{H3dB}.



Obliczenia ręczne:

V_B=5V, V_E=4,3V, V_C=6V, V_{CE}=1,7V, I_C=1mA gm=40mS,
R_{IN}=2,5k, V_S/V_{OUT}=-30,8V/V, f_{3dBL}=72,4Hz , f_{3dBH}=602,3kHz

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

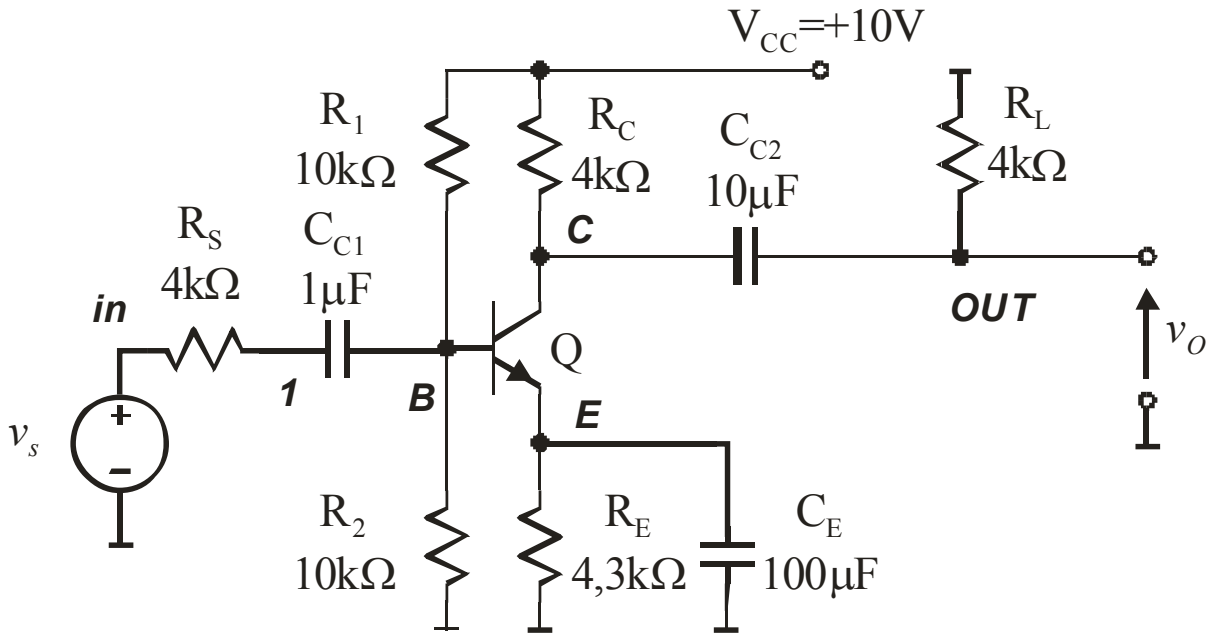
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Przykład symulacji wzmacniacza, c. d.

Dane dla trazystora Q:
f_t=400MHz
C_μ=2pF
V_T =25mV (pot. termiczny)
r_O = ∞
β =200
V_{BE}=0,7V

Należy obliczyć:
- punkt pracy tranzystora Q ,
- wzmocnienie v_o/v_s w środku pasma przenoszenia,
- dolną częstotliwość graniczną f_{L3dB},
- górną częstotliwość graniczną f_{H3dB}.




```
Przyklad 2 wzmacniacz BJT
Vcc vcc 0 dc 10
Vs in 0 dc 0 ac 1 sin(0 10mV 10kHz)
Rs in 1 4k
Cc1 1 B 1u
R1 vcc B 10k
R2 B 0 10k
Rc vcc C 4k
Re E 0 4.3k
Ce E 0 100u
Cc2 C out 10u
RL out 0 4k
Q1 C B E qnpn
.model qnpn npn (BF=200)
Cu C B 2p
Cpi B E 13.9p
.op
.dc Vcc 5 15 .1
.ac dec 100 1 .1g
.tran .001m .3m 0 .001m
.probe
.end
```

Wybrane fragmenty pliku wyjściowego:

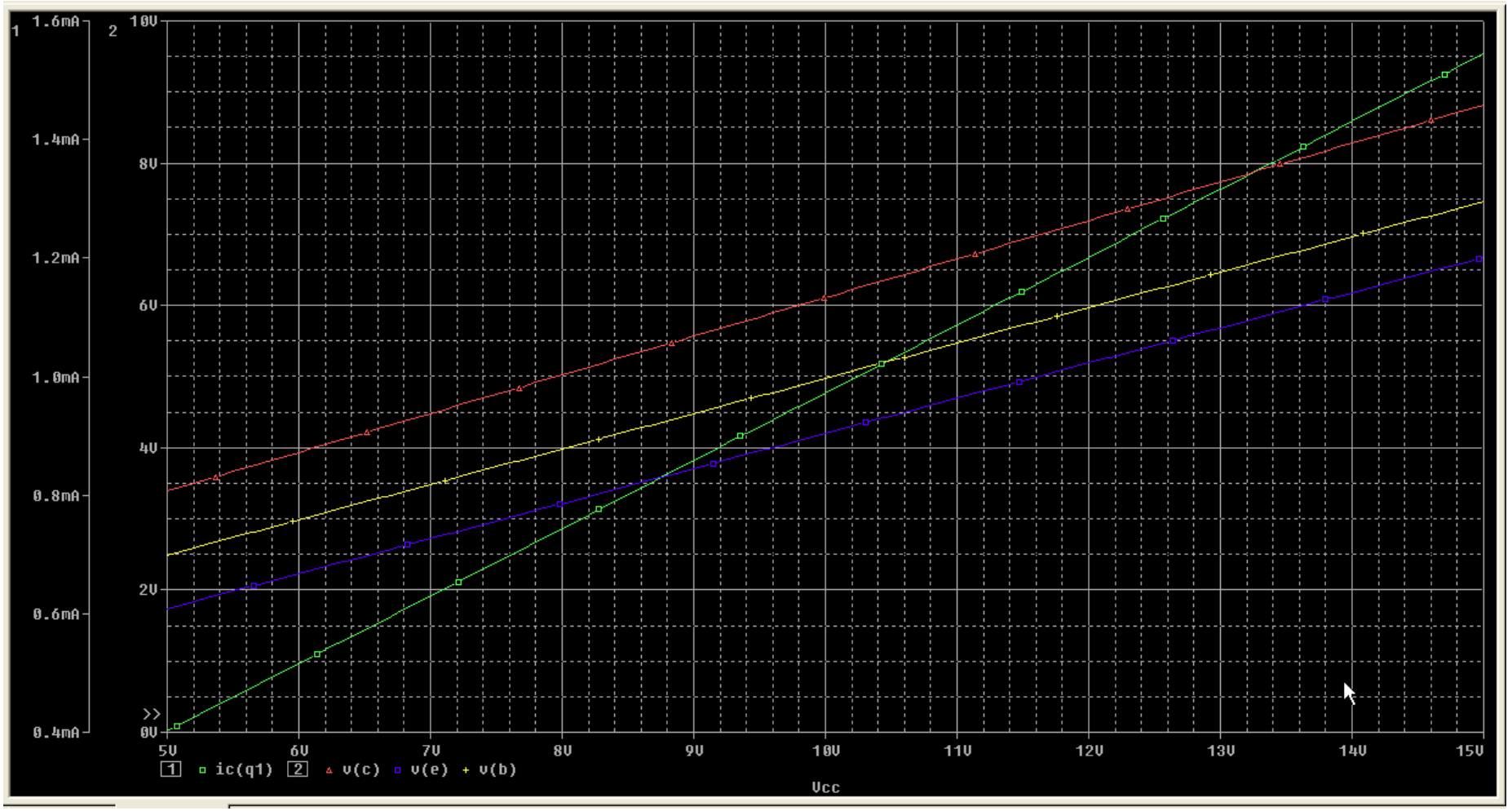
```
 NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE
(    1)    0.0000  (    B)    4.9757  (    C)    6.1104  (    E)    4.2022
(   in)    0.0000  (  out)    0.0000  (  vcc)   10.0000
```

```
**** BIPOLAR JUNCTION TRANSISTORS

NAME          Q1
MODEL         qnpn
IB            4.86E-06
IC            9.72E-04
VBE           7.73E-01
VBC          -1.13E+00
VCE           1.91E+00
BETADC        2.00E+02
GM            3.76E-02
RPI           5.32E+03
RX            0.00E+00
RO            1.00E+12
```

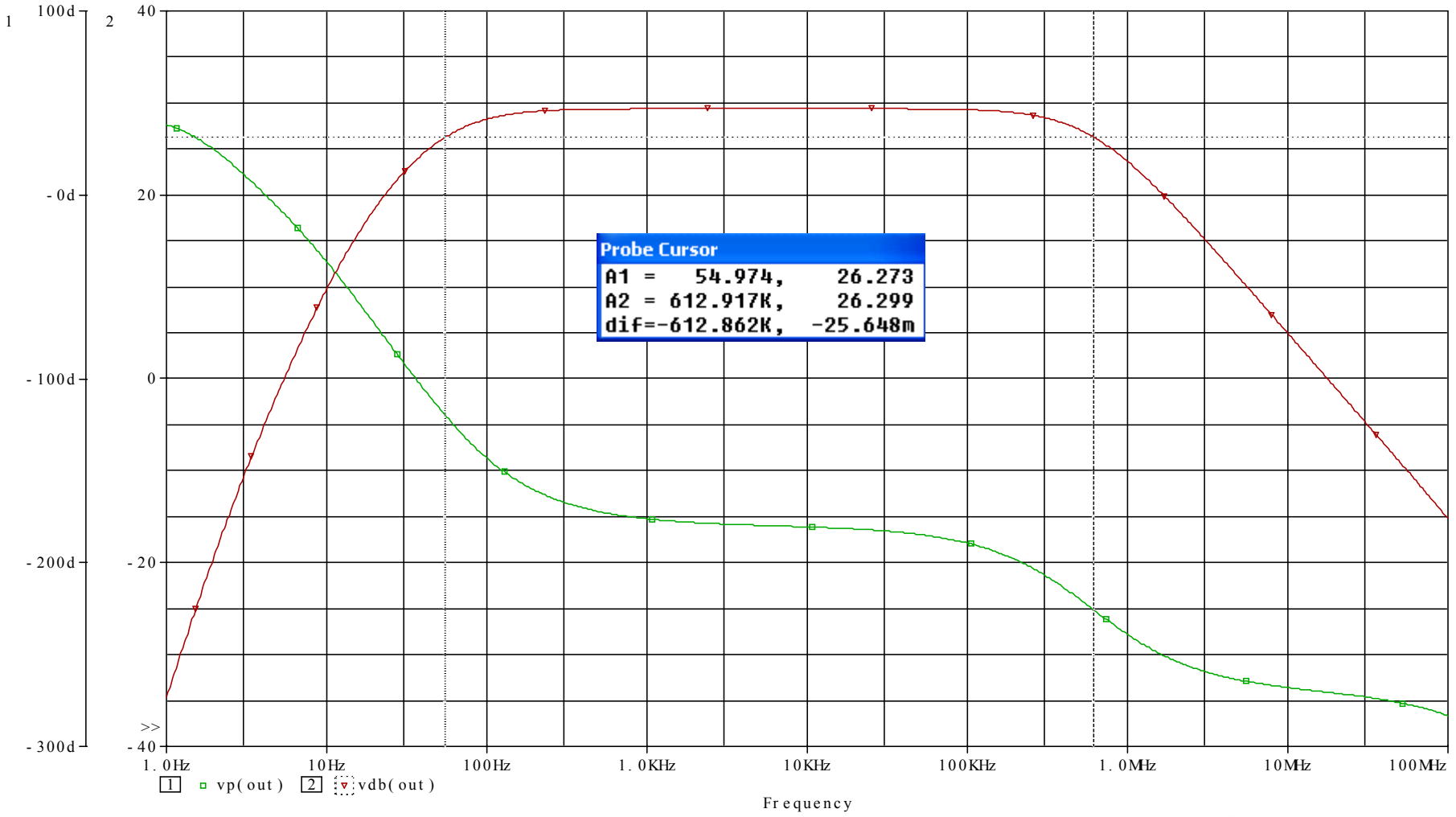
```
   qnpn
                NPN
          IS  100.000000E-18
          BF  200
          NF    1
          BR    1
          NR    1
          CN    2.42
           D     .87
```

PSPICE – Przykład symulacji wzmacniacza, c. d.

Wybrane charakterystyki analizy .DC:



Wybrane charakterystyki analizy .AC:



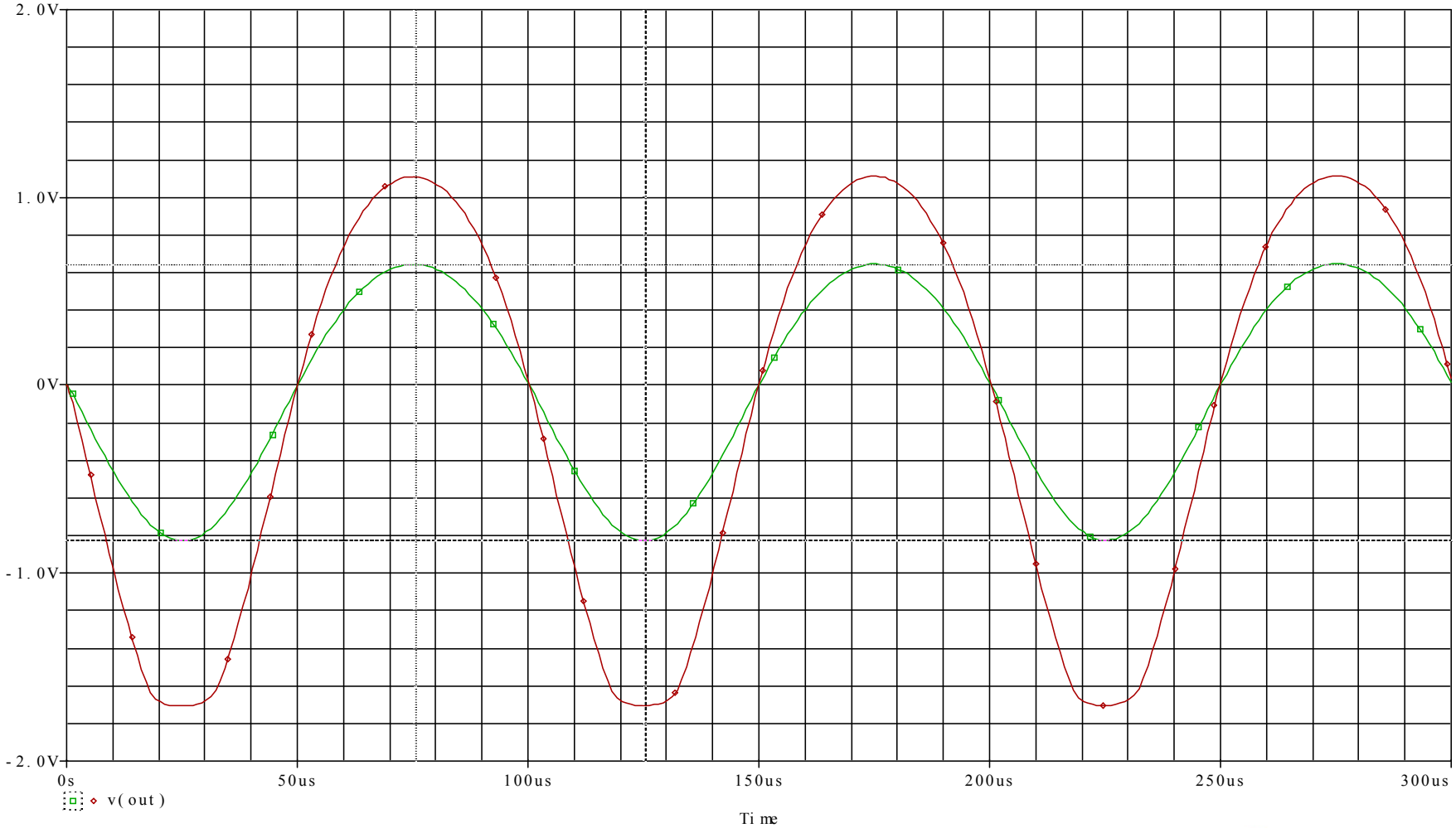
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

PSPICE – Przykład symulacji wzmacniacza, c. d.

Wyniki analizy .TRAN dla amplitudy V_S =25mV i 50mV:



UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Część II - Verilog

Modelowanie, symulacja i projektowanie układów cyfrowych z wykorzystaniem języka

HDL - Verilog

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI
CZŁOWIEK – NAJLEPSZA INWESTYCJA!

Verilog

Rozdział 1. Wstęp

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Znaczenie języków HDL

- Projekty mogą być opisane na bardzo wysokim poziomie abstrakcji.
- Sprawdzenie działania układu może nastąpić na bardzo wczesnym etapie projektowania.
- Opis tekstowy wraz z komentarzem dużo łatwiej zrozumieć i śledzić.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Zastosowania języków HDL:

- Wprowadzanie projektu do systemów automatycznej syntezy i projektowania.
- Opisu układu (netlista).
- Modelowanie i symulacja układów cyfrowych.
- Weryfikacja projektu (*test-bench*).
- Sporządzanie dokumentacji projektu.

Verilog – Historia

- 1984: firma Gateway Design Automation zaczęła sprzedaż symulatora cyfrowego "Verilog-XL". Firma do celów symulacji utworzyła język"Verilog".
- 1989: Gateway Design Automation zostało kupione przez Cadence Design Systems. Verilog-XL wraz z językiem Verilog stał się najbardziej popularnym symulatorem układów ASIC. Kilka firm kupiło licencję na Verilog, m.in. Synopsys, Logic Automation, LMSI.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Historia, c. d.

- 1990: Cadence udostępnił Verilog jako "*public domain*". Cadence dalej sprzedaje Verilog-XL. Powstała organizacja Open Verilog International (OVI) aby kontrolować i promować język Verilog.
- 1993: OVI wydała "Verilog HDL 2.0" – kilka drobnych poprawek do Veriloga. OVI poprosiła IEEE aby uczynić Verilog standardem IEEE.
- 1995: IEEE wydała standard "IEEE 1364-1995".
- 2001: IEEE wydała poprawkę "IEEE 1364-2001" Verilog-2001 standard.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Popularność języka

- Łatwy do nauczenia i używania.
- W składni podobny do języka C.
- Umożliwia opis układu na różnych poziomach abstrakcji jednocześnie
- Większość popularnych systemów syntezy logicznej posiada implementację Verilogu.
- Większość producentów układów ASIC dostarcza biblioteki do symulacji po syntezie logicznej.
- Programming Language Interface (PLI) umożliwia dopisywanie w języku C procedur współdziałających z wewnętrznymi strukturami Verilog - dopasowanie symulatora do indywidualnych potrzeb.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog

- *Poziom kluczy*
- Poziom bramek logicznych
- Poziom rejestrów (*Dataflow*)
- Poziom behawioralny

VHDL

- Poziom strukturalny
- Poziom przesłań międzyrejestrowych RTL
- Abstrakcyjny poziom behawioralny

Verilog

Rozdział 2. Modelowanie hierarchiczne

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Modelowanie w postaci hierarchicznej:

- Złożenie większego projektu z podprojektów, które są następnie:
- Składne z bloków o niższej hierarchii, i.t.d.

Możliwe metodologie projektowania:

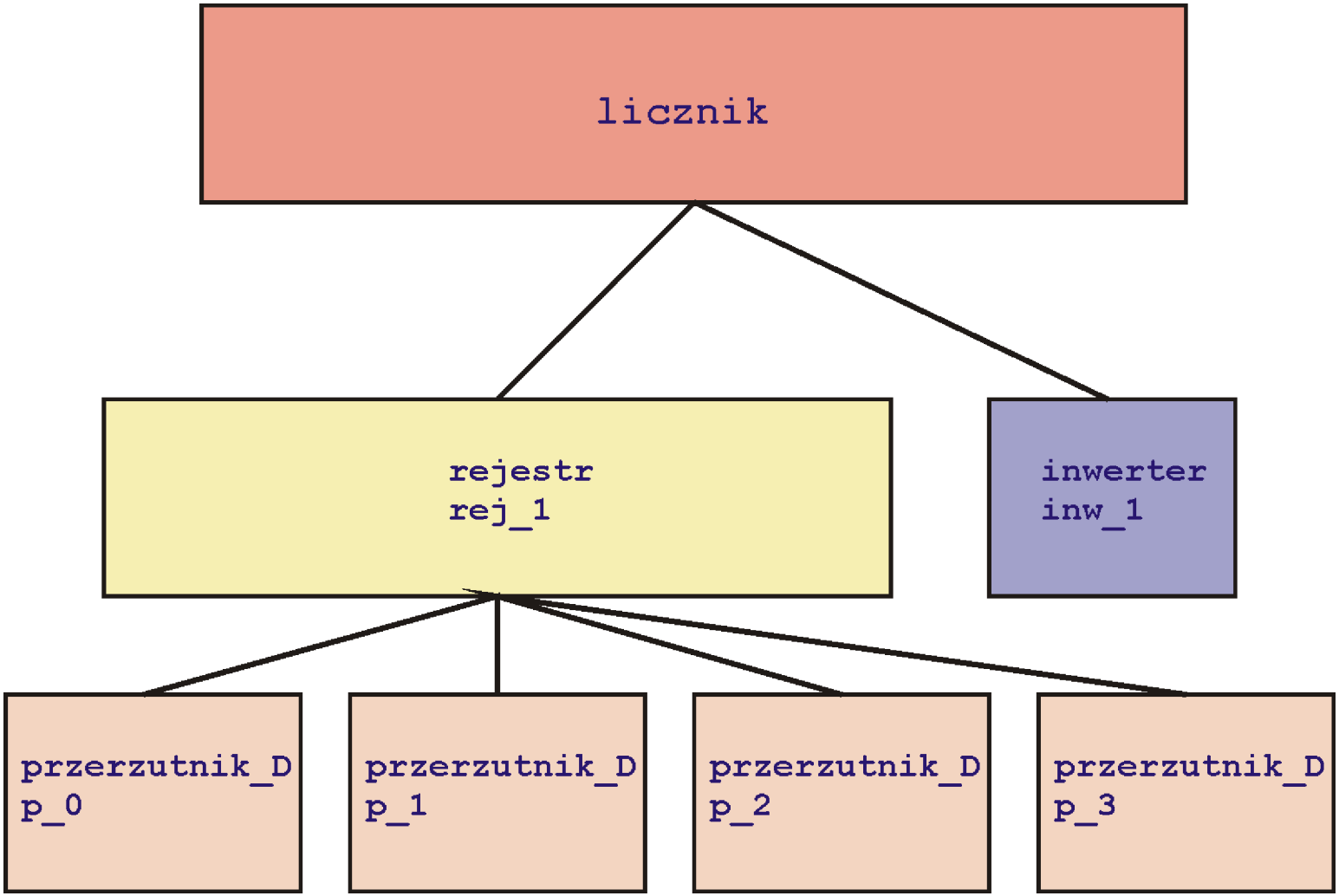
- Top – Down
- Bottom – Up

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Przykład - licznik Johnsona, założenia:

- Układ synchroniczny z asynchronicznym resetem.
- Licznik o 4 bitach.
- Przyjęta hierarchia projektu:
  - licznik -> rejestr -> przerzutnik
  - licznik -> inwerter

Verilog – Przykład c. d.

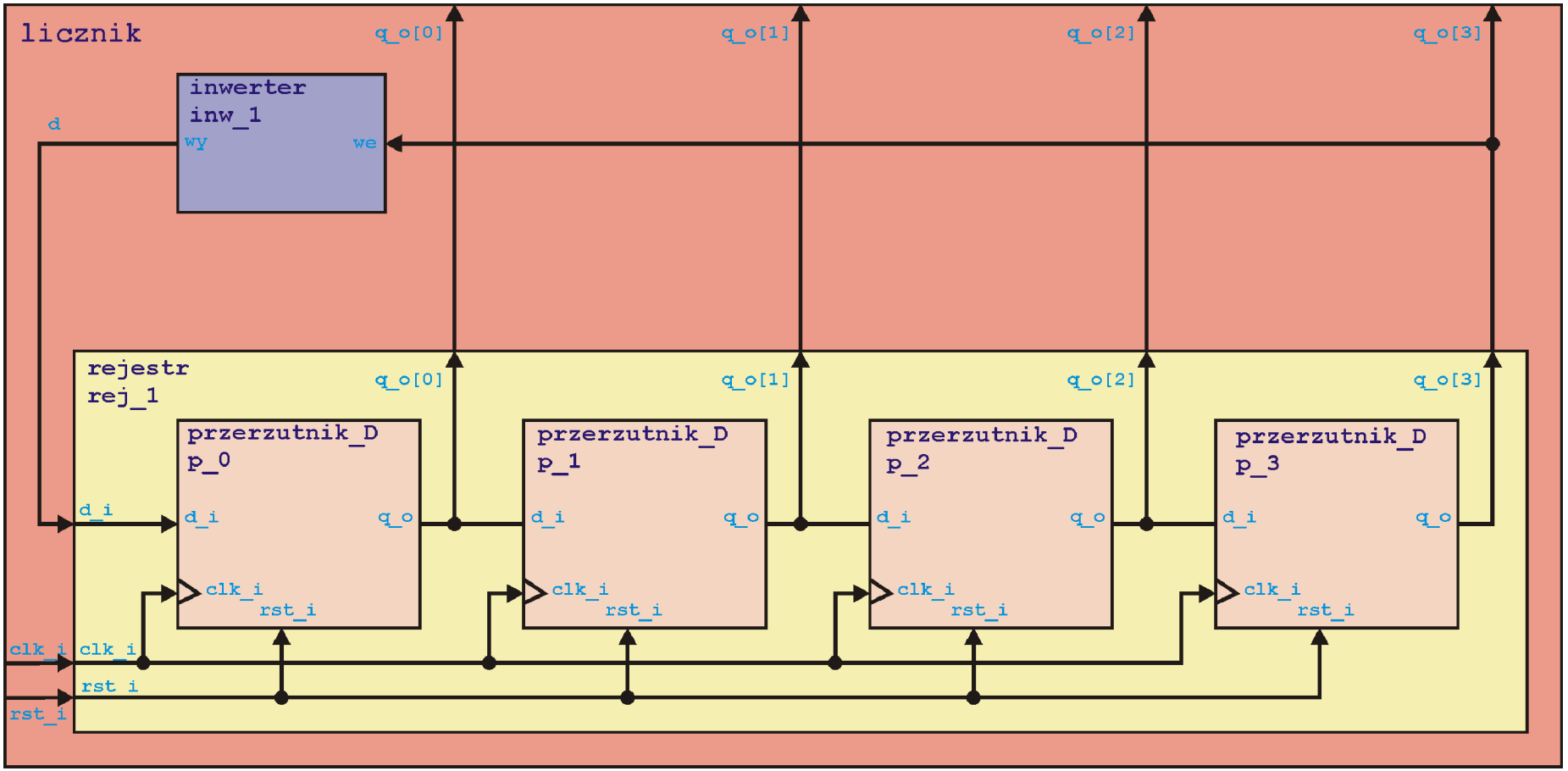


UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Przykład c. d.



Verilog – Moduł

```
module <nazwa_modułu> (<lista_końcówek_modułu>);
...
<wnętrze modułu>
...
...
endmodule
```

Nie wolno zagnieżdżać definicji modułów.

Moduł tylko opisuje działanie bloku układu oraz definiuje jego końcówki. Aby wykorzystać taki blok, należy go powołać do istnienia.

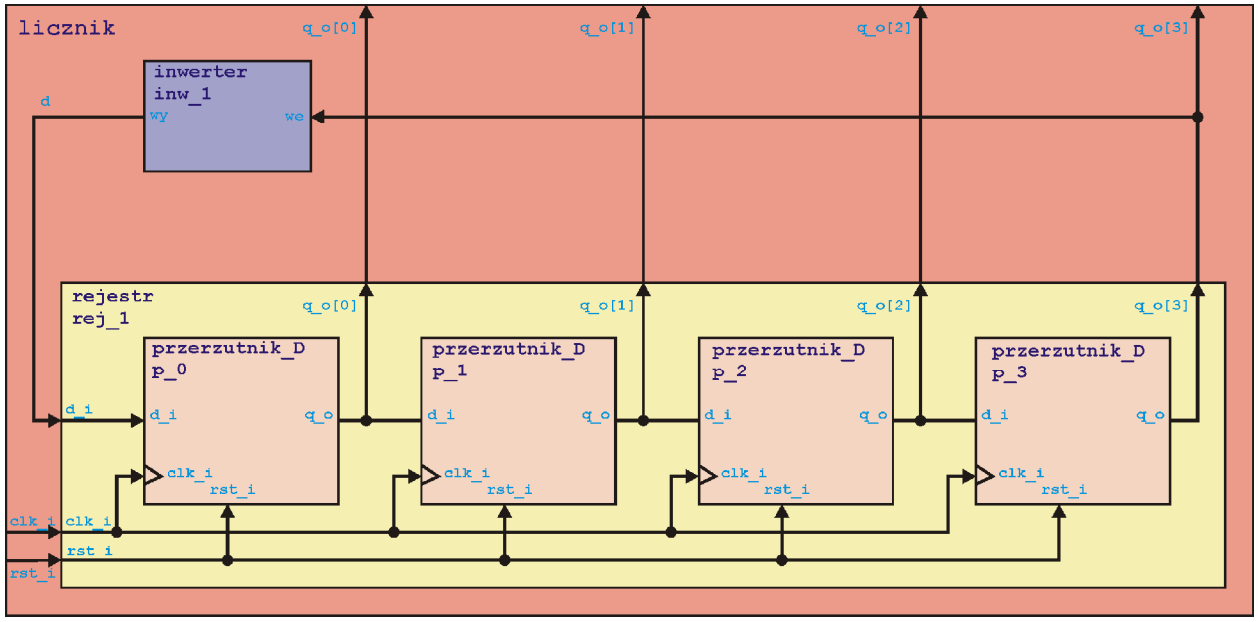
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

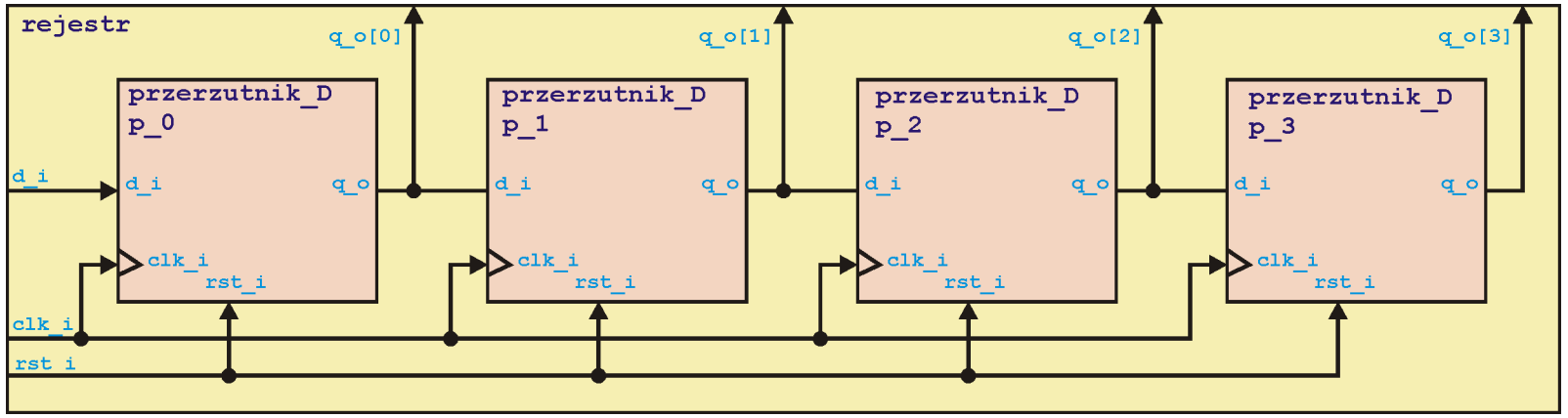
Verilog – Przykład c. d.


```
module licznik(q_o, clk_i, rst_i);
        output [3:0] q_o;
        input clk_i, rst_i;
        wire d;
        rejestr rej_1(q_o, d, clk_i, rst_i);
        inwerter inw_1(d, q_o[3]);
endmodule
```



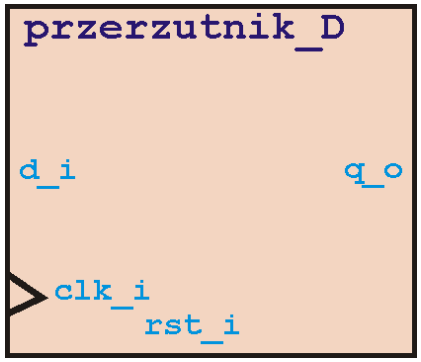
Verilog – Przykład c. d.


```
module rejestr(q_o, d_i, clk_i, rst_i);
        output [3:0] q_o;
        input d_i, clk_i, rst_i;
        przerzutnik_D p_0(q_o[0], d_i, clk_i, rst_i);
        przerzutnik_D p_1(q_o[1], q_o[0], clk_i, rst_i);
        przerzutnik_D p_2(q_o[2], q_o[1], clk_i, rst_i);
        przerzutnik_D p_3(q_o[3], q_o[2], clk_i, rst_i);
endmodule
```



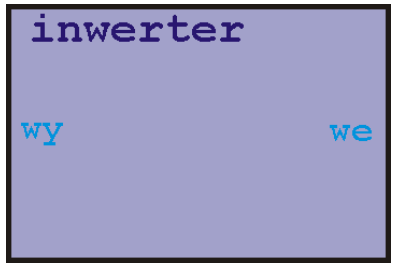
Verilog – Przykład c. d.


```
module przerzutnik_D(q_o, d_i, clk_i, rst_i);
        output reg q_o;
        input d_i, clk_i, rst_i;
        always@(posedge clk_i or posedge rst_i)
        begin
                if (rst_i==1)
                        q_o = 0;
                else
                        q_o = d_i;
        end
endmodule
```



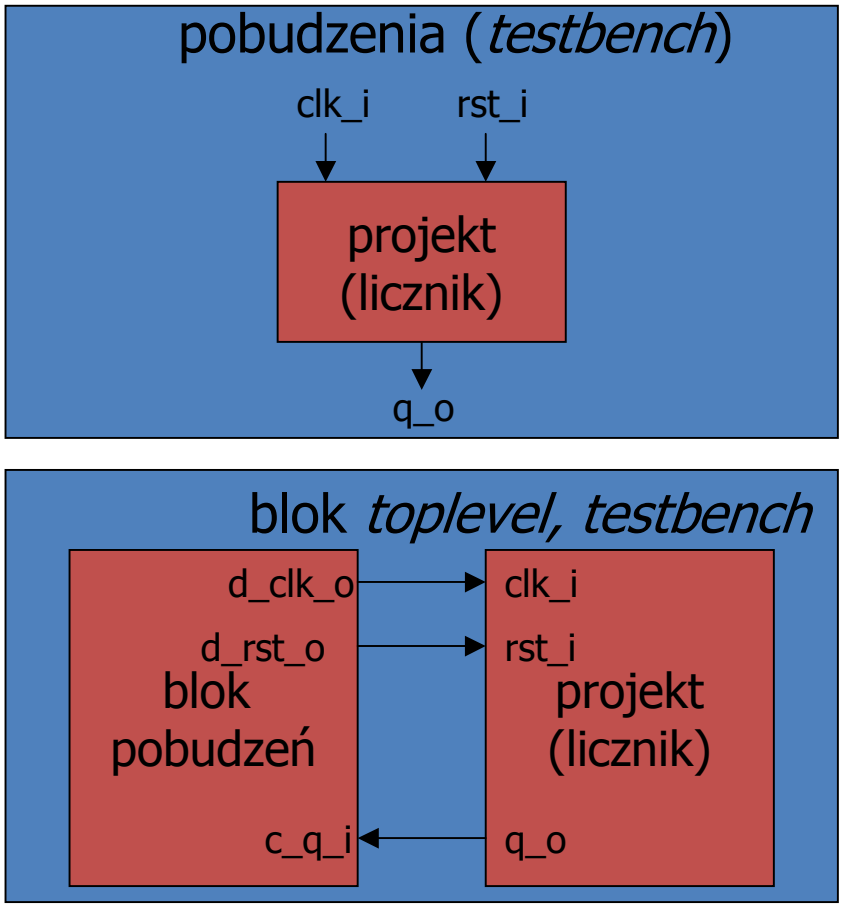
Verilog – Przykład c. d.

```
module inwerter(wy,we);
        output wy;
        input we;

        not n1(wy,we);
endmodule
```



Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Verilog – Przykład, symulacja



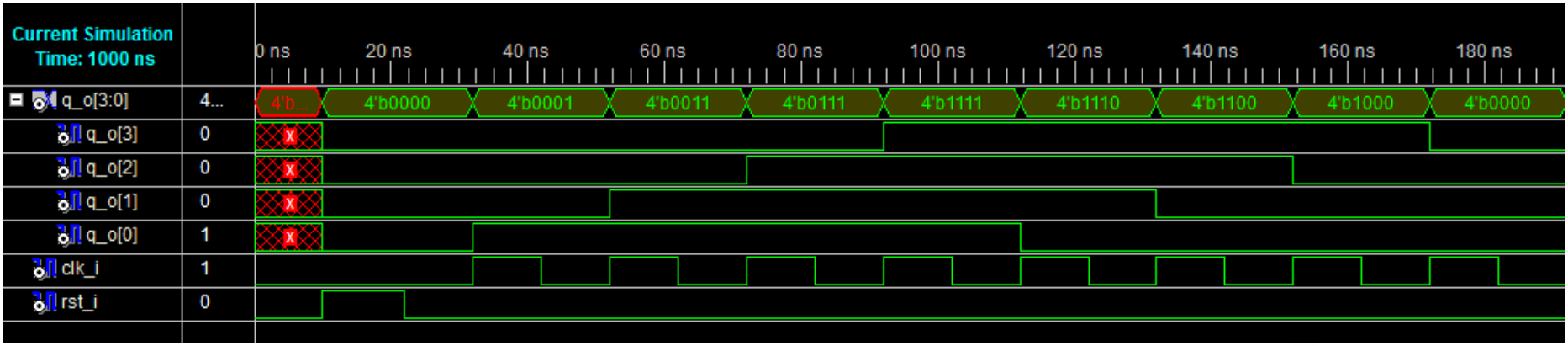
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Przykład, symulacja c. d.


```
`timescale 1ns / 1ps
module testbench;
    reg clk_i;
    reg rst_i;
    wire [3:0] q_o;
    // wstawienie testowanego bloku (UUT)
    licznik uut (q_o, clk_i,rst_i);
    initial
       begin
          clk_i = 0;
          rst_i = 0;
          # 10
          rst_i = 1;
          #12
          rst_i = 0;
          forever clk_i = #10 !clk_i;
       end
endmodule
```



Verilog

Rozdział 3. Podstawowe pojęcia

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Verilog rozróżnia **małe** i **DUŻE** litery!
- Słowa kluczowe piszemy **małymi** literami
- Biała spacja (spacja, tab., koniec linii) jest ignorowana (wyjątek: łańcuchy)
- Komentarze:

```
// komentarz

/* komentarz komentarz komentarz
 komentarz komentarz komentarz */
```


Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- **Operatory:**

```
a = ~b;          // 1 argument
a = b && c;      // 2 argumenty
a = b ? c : d;   // 3 argumenty
```

- **Liczby:**

<rozmiar>'<podstawa><liczba>

<rozmiar> = liczbą dziesiętną, ilość bitów w reprezentacji liczby.

<podstawa> = dziesiętna 'd lub 'D, szesnastkowa 'h lub 'H, binarna 'b lub 'B, ósemkowa 'o lub 'O

```
4'b1111 // to jest 4-bitowa liczba dwójkowa
12'habc // to jest 12-bitowa liczba szesnastkowa
16'd255 // to jest 16-bitowa liczba dziesiętna
```


Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- **Liczby (cd.):**
  - Liczby o nie określonej podstawie = dziesiętne.
  - Liczby bez określonego rozmiaru = 32-bitowe (zależne też od komputera).

```
23456 // liczba dziesiętna
'hc3  // 32-bitowa liczba szesnastkowa
'o21  // 32-bitowa liczba ósemkowa
```

  - **x** oznacza stan nieokreślony
  - **z** oznacza stan wysokiej impedancji

- **Liczby (cd.):**

```
12'h13x // 12-bitowa liczba szesnastkowa,
        // 4 najmniej znaczące bity są nieokreślone
6'hx    // 6-bitowa liczba szesnastkowa
32'bz   // 32-bitowa liczba o wysokiej impedancji
```

  - **x** lub **z** oznacza 4 bity w reprezentacji szesnastkowej, trzy bity w reprezentacji ósemkowej i jeden bit w dwójkowej.
  - Jeśli najbardziej znaczący bit w liczbie to 0, **z** lub **x**, to najbardziej znaczące bity są automatycznie wypełniane tą wartością. Jeśli najbardziej znaczącym bitem jest 1 to pozostałe najbardziej znaczące bity są uzupełniane zerami.

```
4'b0 = 0000
4'bz = zzzz
4'bx = xxxx
4'b1 = 0001
```

- **Liczby ujemne:**
  - Liczby ujemne określa się poprzez dodanie znaku minus przed określeniem rozmiaru liczby. Dodanie minusa przed liczbą spowoduje jej zapisanie w formacie „uzupełnienie do dwóch”

```
-6'd3 // 6-bitowa liczba ujemna przechowywana
      // jako uzupełnienie dwójkowe.
```

  - W liczbach, aby zwiększyć ich czytelność, można stosować znak podkreślenia (byle nie na początku liczby).
  - Znak zapytania w liczbie jest odpowiednikiem stanu wysokiej impedancji **z**.

- **Różne przykłady liczb *integer*:**

Liczba		*Reprezentacja*
1	//=	00000000000000000000000000000001
8'hF0	//=	11110000
6'b10_1100	//=	101100
'hF	//=	00000000000000000000000000001111
6'hE7	//=	100111
6'hF	//=	001111
16'bz	//=	zzzzzzzzzzzzzzzz
8'bx	//=	xxxxxxxx

- **Łańcuchy:**
  - Łańcuch musi być zdefiniowany w jednej linii - nie dopuszcza się znaków CR w łańcuchu.

- **Identyfikatory:**
  - Można stosować znaki alfanumeryczne, znak podkreślenia, oraz **$**.
  - Nie mogą się zaczynać od **$** ani od cyfry
  - Identyfikatory zaczynające się od \:

```
\a+b-c
\**my_name**
```


Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- **Zestaw wartości**

Wartość	Znaczenie w układzie cyfrowym
`0`	logiczne 0, fałsz
`1`	logiczna jedynka, prawda
`x`	wartość nieokreślona
`z`	stan wysokiej impedancji

Jeśli dwa sygnały mają równe siły, to na przewodzie jest sygnał nieokreślony **x**

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- **Siła sygnału**

Nazwa siły sygnału	Typ	Waga siły sygnału
`supply`	driving	najsilniejszy
`strong`	driving	
`pull`	driving	
`large`	storage	
`weak`	driving	
`medium`	storage	
`small`	storage	
`highz`	high impedance	najsłabszy

Jeśli dwa sygnały mają równe siły, to na przewodzie jest sygnał nieokreślony **x**

- **Sieci:**
  - połączenia pomiędzy elementami układu.
  - standardowo przyjmują one szerokość 1-bitową.
  - standardową domyślną wartością jest **z**.

```
wire a;
wire b, c;
wire d=1'b0 // sieć d ma wartość 0
            // podczas deklaracji
```


Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Verilog – Typy danych c. d.

- Rejestry:
  - reprezentują elementy pamiętające = zmienne, które pamiętają ostatnio zapisaną wartość.
  - wartość domyślna: **z**

```
reg reset; // <- deklaracja zmiennej reset
initial    // <- będzie wyjaśnione później
begin
   reset=1'b1; // ustaw reset=1
   #100 reset=1'b0; // po 100 jednostkach czasu
                    // ustaw reset=0
end
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Typy danych c. d.

- Wektory:
  - Sieci (**wire**) i rejestry (**reg**) mogą być wektorami:

```
wire a; // skalar
wire [7:0] bus; // 8-bitowa magistrala
wire [31:0] busA, busB, busC;
reg [0:40] addr;
```

  - [**max**:**min**] lub [**min**:**max**] = ale zawsze lewa liczba oznacza najbardziej znaczący bit.
  - Użycie części wektorów:

```
busA[7] // 7-my bit wektora busA
bus[2:0] // trzy najmniej znaczące bity wektora
         // (nie wolno: bus[0:2] bo wcześniej
         // zdefiniowano: wire [7:0] bus)
```

Verilog – Typy danych c. d.

- Liczby całkowite:

```
integer licznik;
initial
  licznik = -1;
```

- Liczby rzeczywiste:

```
real dana;
initial
begin
  dana=4e10;
end
```

Kiedy wartość rzeczywista jest przypisywana do całkowitej, jest ona zaokrąglana do najbliższej wartości całkowitej

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- **Tablice:**
  - **reg**, **integer**, **time**. Nie wolno dla **real**!
  - Dostęp do elementu macierzy:

    <nazwa_tablicy>[<indeks>]
  - Nie wolno definiować tablic wielowymiarowych.

```
integer count[0:7]; // tablica 8 liczników
reg bool[31:0]; // tabl. 32, jednobitowych rejestrów
time chk_point[1:1000];
reg [4:0] port_id[0:7]; // tablica 8 zmiennych,
                        // każda ma 5 bitów
count[5] // 5-ty element tablicy count
port_id[3] // 3-ci element zmiennej (5-bitowej)
```

  - Nie można odwoływać się do pojedynczych bitów tablicy!

- **Tablice (cd.):**
  - Nie należy mylić tablic z wektorami!
    - Wektor to pojedynczy element o szerokości n-bitów.
    - Tablica to zestaw wielu elementów o szerokości 1 lub n bitów.
- **Pamięci:**
  - Pamięci = tablice rejestrów.
  - Każdy element tablicy to słowo.
  - Każde słowo może mieć długość 1 lub więcej bitów.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- **Łańcuchy:**
  - Przechowywane są w zmiennych typu **reg**.
  - Szerokość rejestru musi być wystarczająca do przechowania łańcucha.
  - Każdy znak w łańcuchu zajmuje 8 bitów.
  - Jeśli łańcuch jest krótszy, Verilog uzupełnia bity z lewej strony zerami.
  - Jeśli łańcuch jest za długi, to Verilog obcina lewą część łańcucha.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- **Łańcuchy (cd.):**

```
reg [8*12:1] string_value; // zmienna o szer. 18 bajtów
initial
   string_value="Halo Verilog";
```

  - W łańcuchu można umieszczać znaki specjalne poprzedzone znakiem \:

```
\n      =      nowa linia
\t      =      tabulator
%%      =      %
\\      =      \
\"      =      "
\ooo    =      znak zapisany 1-3 cyfr ósemkowych
```

Zadania systemowe:

- Wywołanie zadania poprzez: $<słowo_kluczowe>
- wyświetlanie na ekranie w konsoli programu;
- monitorowanie wartości sieci;
- pauzowanie;
- zatrzymywanie programu.

- Wyświetlanie w konsoli:

```
$display(p1,p2,p3,...,pn) - wyświetla zmienną lub
   łańcuch na ekranie.
```


Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Zadania systemowe, kilka przykładów:

```
$display("Halo Verilog");
-- Halo Verilog

$display($time);
-- 1485

$display(„W czasie %d adres na szynie jest %h", $time, addr);
-- W czasie 1381 adres na szynie jest 1000a

$display("ID portu jest równe %b", port);
-- ID portu jest równe 1101

$display("Ten ciąg jest z hierarchii %m ");
-- Ten ciąg jest z hierarchii licznik.pl

$display("To jest wieloliniowy \n ciąg ze znakiem %% ");
-- To jest wieloliniowy
-- ciąg ze znakiem %
```

Verilog – Zadania systemowe c. d.

- $display na koniec wstawia znak nowej linii.
- Skłania jest podobna do printf w C:

%d lub %D	wyświetl zmienną w postaci dziesiętnej
%b lub %B	wyświetl zmienną w postaci binarnej
%s lub %S	wyświetl zmienną jako łańcuch
%h lub %H	wyświetl zmienną w postaci szesnastkowej
%c lub %C	wyświetl zmienną w postaci znaku ASCII
%m lub %M	wyświetl nazwę hierarchiczną (nie potrzebny argument)
%v lub %V	wyświetl siłę sygnału
%o lub %O	wyświetl zmienną w postaci ósemkowej
%t lub %T	wyświetl zmienną w postaci czasu
%e lub %E	wyświetl zmienną w postaci naukowej (np. 3.45e6)
%f lub %F	wyświetl zmienną w postaci zmiennoprzecinkowej
%g lub %G	wyświetl zmienną w postaci naukowej lub zmiennoprzecinkowej, w zależności od tego, która jest prostsza.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Zadanie systemowe $monitor:

- Monitorowanie zmiennych - po każdej zmianie sygnału będzie drukowana jego wartość.

```
$monitor(p1, p2, p3, ..., pn);
```

- Parametry = jak w poleceniu **$display**. Różnicą w stosunku do **$display** jest, że **$monitor** wystarczy podać tylko raz.
- Jeśli występuje więcej niż jedno polecenie **$monitor**, to aktywne jest tylko ostatnie.

- **$monitoron;** - załącza monitorowanie,
- **$monitoroff;** - wyłącza monitorowanie.

- Pauzowanie i koniec symulacji:
  - **$stop;** - zatrzymuje symulację i przełącza symulator w tryb interaktywny.
  - **$finish;** - kończy symulację.

Verilog – Dyrektywy kompilatora

- `**define** definiuje makro tekstowe. Verilog zastępuje każde wystąpienie `<nazwa_makra> określonym tekstem.

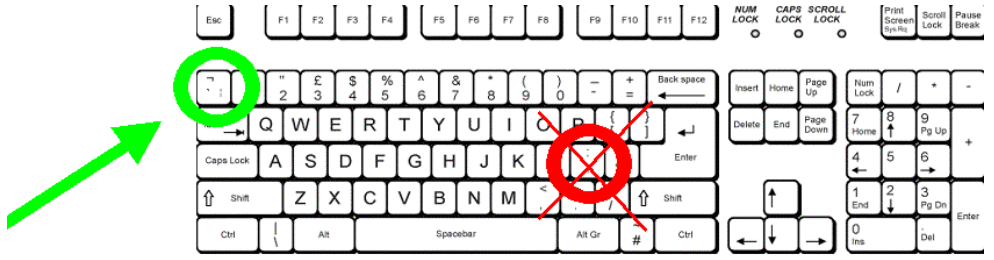
```
`define FALSE 1'b0
`define WORD_SIZE 32
`define S $stop;
`define WORD_REG reg [31:0]
```

- Wykorzystanie makra:

```
assign out = `FALSE;
```

- `**include** - pozwala dołączyć dowolny plik:

```
`include header.v
```



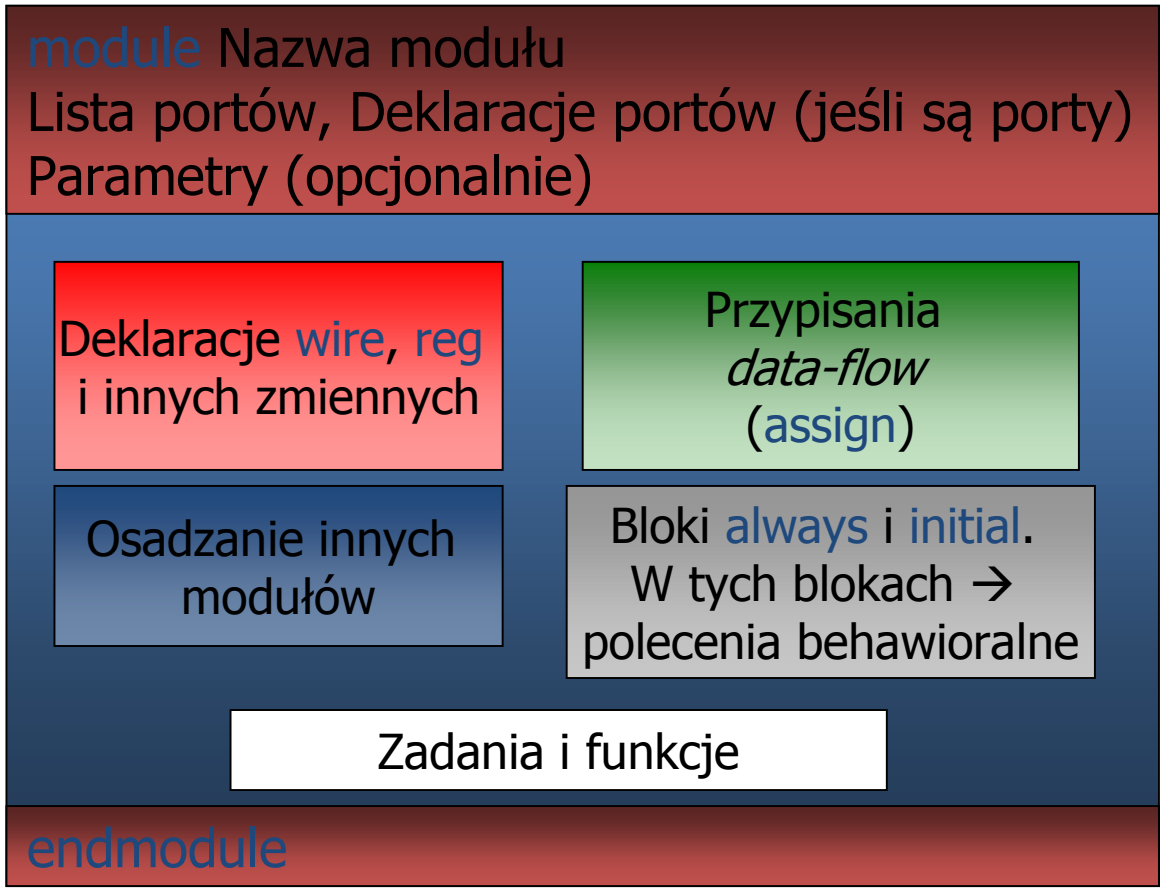
Verilog

Rozdział 4. Moduły i porty

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

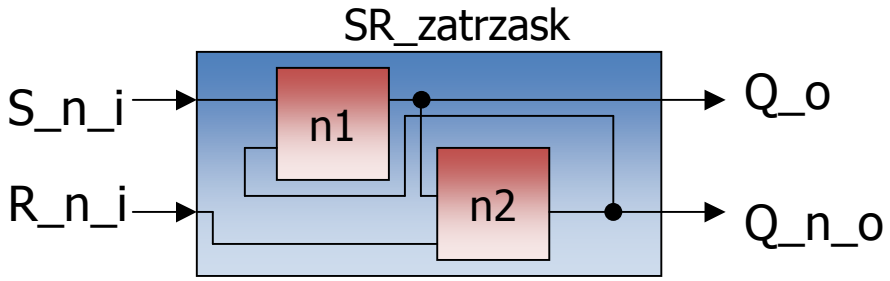
KAPITAŁ LUDZKI


module Nazwa modułu
Lista portów, Deklaracje portów (jeśli są porty)
Parametry (opcjonalnie)
Deklaracje wire, reg i innych zmiennych
Przypisania *data-flow* (assign)
Osadzanie innych modułów
Bloki always i initial. W tych blokach → polecenia behawioralne
Zadania i funkcje
endmodule

Verilog – Moduły i porty c. d.

```
// nazwa modułu i lista portów
module SR_zatrzask(Q_o, Q_n_o, S_n_i, R_n_i);

  // deklaracje portów
  output Q_o, Q_n_o;
  input S_n_i, R_n_i;

  // osadzanie innych modułów/obiektów:
  nand n1(Q_o, S_n_i, Q_n_o);
  nand n2(Q_n_o, R_n_i, Q_o);

  // słowo kluczowe endmodule
endmodule
```

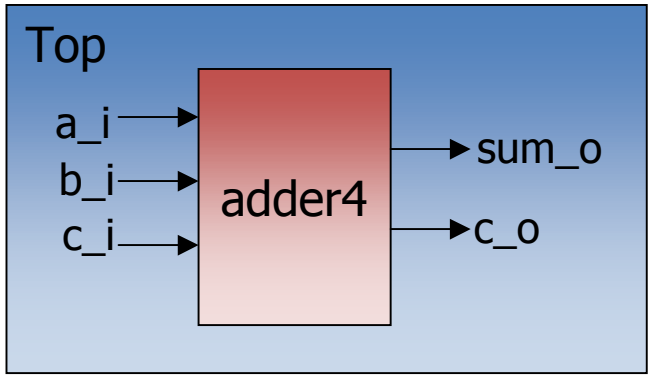
średnik



brak średnika

- Lista portów
  - Jeśli moduł nie wymienia sygnałów z otoczeniem, lista portów nie jest potrzebna:

```
module adder4(sum_o, c_o, a_i, b_i, c_i);
module Top;
```



- Wszystkie porty muszą zostać zadeklarowane jako **input**, **output** lub **inout**.
- Deklaracje te niejawnie inicjują port jako **wire**. Zazwyczaj jest to wystarczające dla portów typu **input** i **inout**.
- Porty typu **output** można dodatkowo zdeklarować jako **reg**, aby mogły pamiętać wartość.

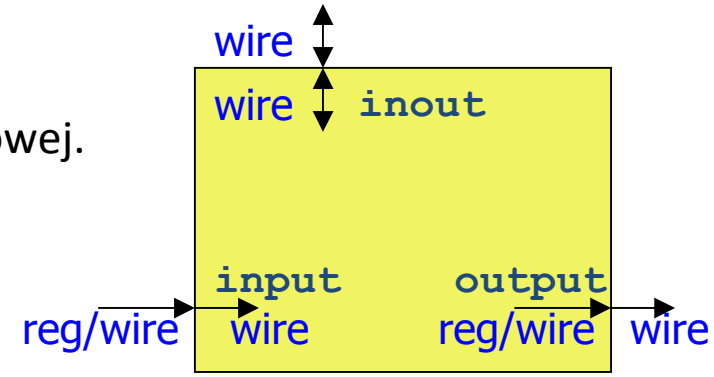
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

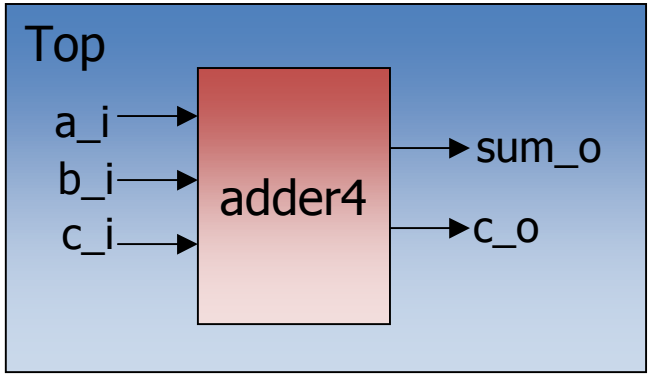
Verilog – Moduły i porty c. d.

- Reguły łączenia portów
  - **input** – wewnętrznie **wire**, zewnętrznie można łączyć z **reg** i **wire**.
  - **output** - wewnętrznie **reg** lub **wire**, zewnętrznie można podłączyć do **wire**. Nie można do **reg**!
  - **inout** - wewnętrznie **wire**, zewnętrznie muszą być podłączone do sieci typu **wire**.
- Dopasowanie szerokości
  - Można łączyć magistrale o różnej szerokości bitowej.

    Kompilator wygeneruje tylko ostrzeżenie.
- Nie podłączone porty
  - Można zostawiać porty nie podłączone.

```
adder4 fa0(SUM, ,A, B, C_IN);
// port c_o jest nie podłączony
module adder4(sum_o, c_o, a_i, b_i, c_i);
```

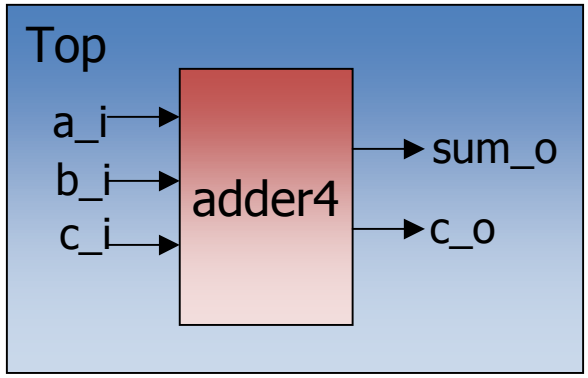




- Podłączanie portów do sygnałów zewnętrznych (dwie metody - nie mogą być stosowane jednocześnie):
  - Lista uporządkowana:

    porty w tej samej kolejności jak w definicji:

```
adder4 fa1 (SUM, C_OUT, A, B, C_IN);
```



  - Łączenie poprzez podanie nazwy:

```
adder4 fa2(.c_o(C_OUT), .sum_o(SUM), .b_i(B),
   .c_i(C_IN), .a_i(A) );
```

nazwa portu
(sygnał wewnętrzny modułu)

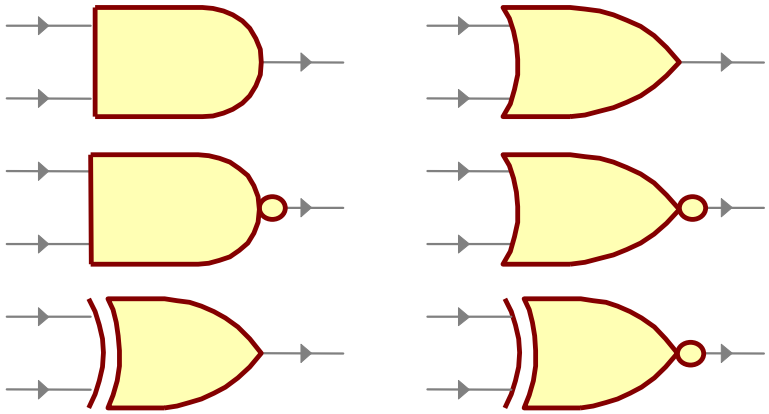
do czego podłączamy
(sygnał zewnętrzny względem modułu)

Verilog
Rozdział 5. Projektowanie na poziomie bramek logicznych

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

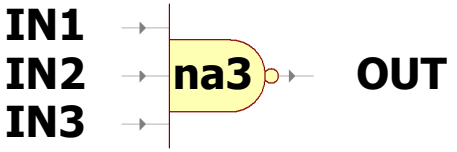
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI



- Dostępne bramki:
  - **and**, **nand**, **or**, **nor**, **xor**, **xnor**
- Bramki o większej ilości wejść = podanie większej ilości wejść w liście portów.

```
wire OUT, IN1, IN2, IN3;

and a1(OUT, IN1, IN2);
nand na3(OUT, IN1, IN2, IN3);
and (OUT, IN1, IN2); //można nie podawać nazwy bramki
```

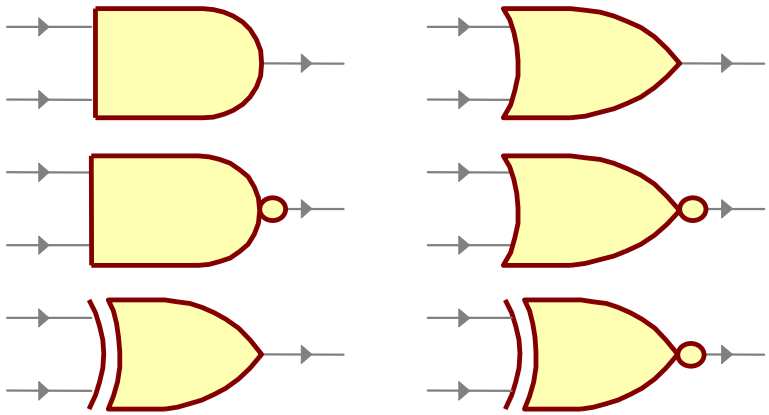


- Bramki **and**, **nand**, **or**, **nor**, **xor**, **xnor** (cd.)

and	0	1	x	z
0	0	0	0	0
1	0	1	x	x
x	0	x	x	x
z	0	x	x	x

or	0	1	x	z
0	0	1	x	x
1	1	1	1	1
x	x	1	x	x
z	x	1	x	x

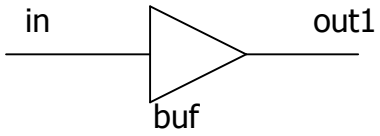
xor	0	1	x	z
0	0	1	x	x
1	1	0	x	x
x	x	x	x	x
z	x	x	x	x

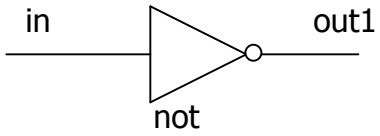


Verilog – Bramki logiczne c. d.

- Bramki typu **buf** / **not**
  - Mogą one mieć jedno **wejście** i wiele **wyjść**

```
buf b1(OUT1, IN);
not n1(OUT1,IN);
buf b2(OUT1, OUT2, IN); // wiele wyjść
not (OUT1, IN); // nie trzeba podawać nazwy
```





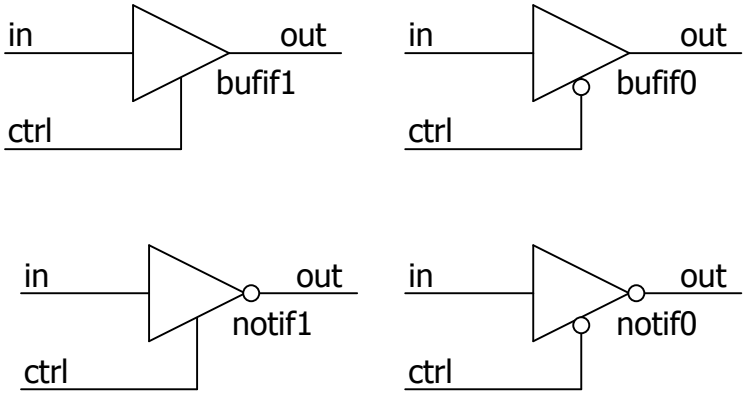
buf:

in	out
0	0
1	1
x	x
z	x

not

in	out
0	1
1	0
x	x
z	x

- Bramki typu **bufif** i **notif**:



Bufif1		ctrl			
		0	1	x	z
	0	z	0	L	L
In	1	z	1	H	H
	x	z	x	x	x
	z	z	x	x	x

Notif1		ctrl			
		0	1	x	z
	0	z	1	H	H
in	1	z	0	L	L
	x	z	x	x	x
	z	z	x	x	x

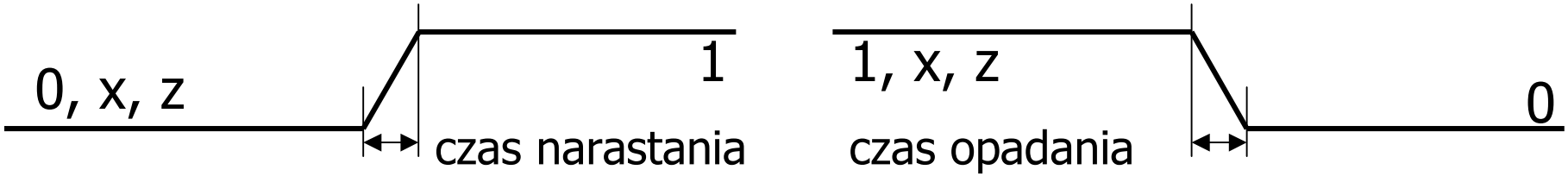
Bufif0		ctrl			
		0	1	x	z
	0	0	z	L	L
In	1	1	z	H	H
	x	x	z	x	x
	z	x	z	x	x

notif0		ctrl			
		0	1	x	z
	0	1	z	H	H
in	1	0	z	L	L
	x	x	z	x	x
	z	x	z	x	x

(oznaczenia: L=0,**z**, H=1,**z**)

```
bufif1 b1 (out, in, ctrl);
bufif0 b0 (out, in, ctrl);
notif1 n1 (out, in, ctrl);
notif0 n0 (out, in, ctrl);
```

Verilog – Opóźnienia w bramkach



- **Czas narastania** = czas w jakim wyjście osiągnie stan **1** z dowolnego innego stanu.
- **Czas opadania** = czas w jakim wyjście osiągnie stan **0** z dowolnego innego stanu.
- **Czas wyłączenia** = czas w jakim wyjście osiągnie stan wysokiej impedancji **z** z dowolnego innego stanu.
- Jeśli wyjście zmienia się do wartości **x**, to dzieje się to po czasie najkrótszym z trzech czasów opisanych powyżej.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Opóźnienia w bramkach c. d.

- Możliwości określenia czasów opóźnień:
  - 1 wartość = czas wszystkich trzech przejść;
  - 2 wartości = czasy narastania i opadania, czas wyłączenia = mniejszej z tych liczb.
  - 3 wartości = czas narastania, opadania i wyłączenia.
- Jeśli nie podano żadnej wartości, wszystkie trzy czasy są równe zeru.

```
// taki sam czas dla trzech rodzajów przejść:
and #(4) a1(out, i1,i2);
// określenie czasu narastania i opadania:
and #(3, 2) a2(out, i1,i2);
// określenie trzech czasów:
bufif0 #(3, 2, 5) b1(out, in, control);
```

narastania opadania wyłączania

Verilog – Opóźnienia w bramkach c. d.

- Wartości minimalne, typowe i maksymalne:

```
// jedno opóźnienie:
and #(4:5:6) a1(out, i1,i2);
```

Min. Typ. Max.

```
// dwa opóźnienia:
and #(3:4:5, 5:6:7) a2(out, i1, i2);
```

Min. Typ. Max. Min. Typ. Max.

```
// trzy opóźnienia:
and #(2:3:4, 3:4:5, 4:5:6) a3(out, i1, i2);
```

Min. Typ. Max. Min. Typ. Max. Min. Typ. Max.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

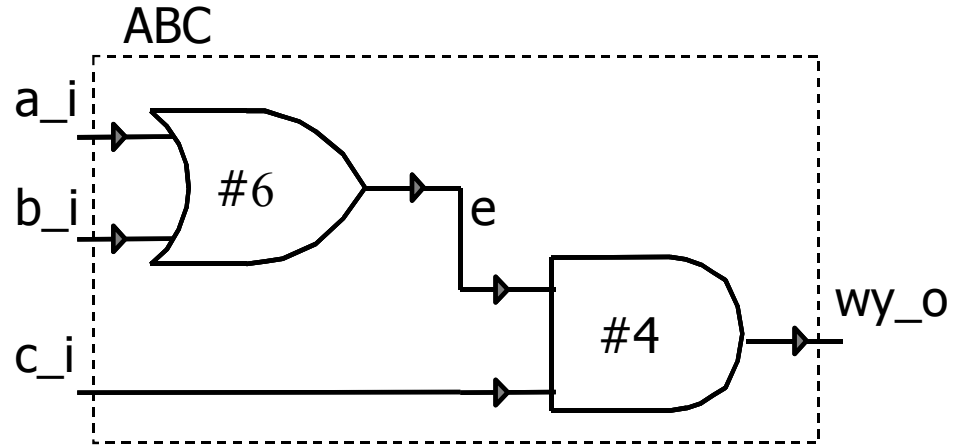
KAPITAŁ LUDZKI

Verilog – Opóźnienia w bramkach przykład

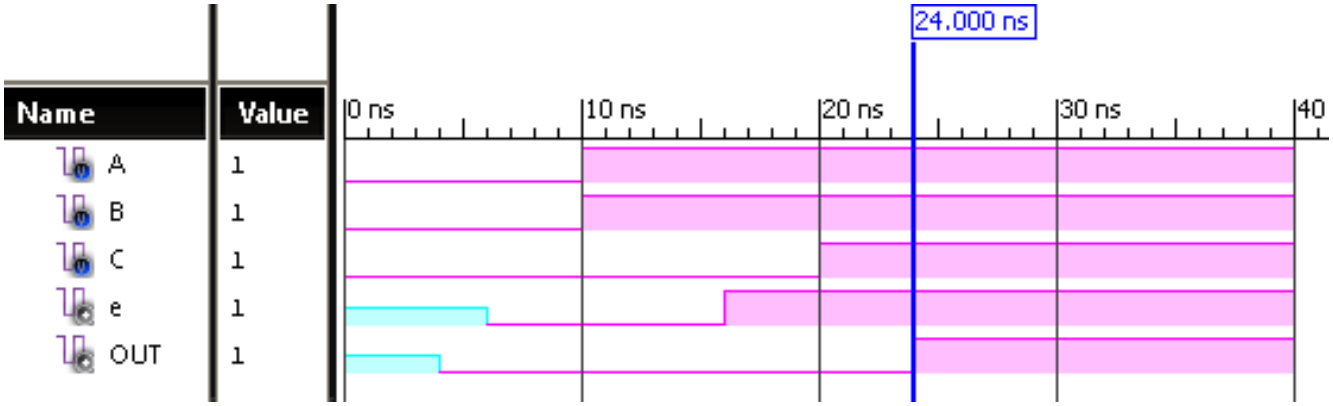
- Moduł **ABC** ma realizować funkcję logiczną:

wy_o = (a_i + b_i) * c_i

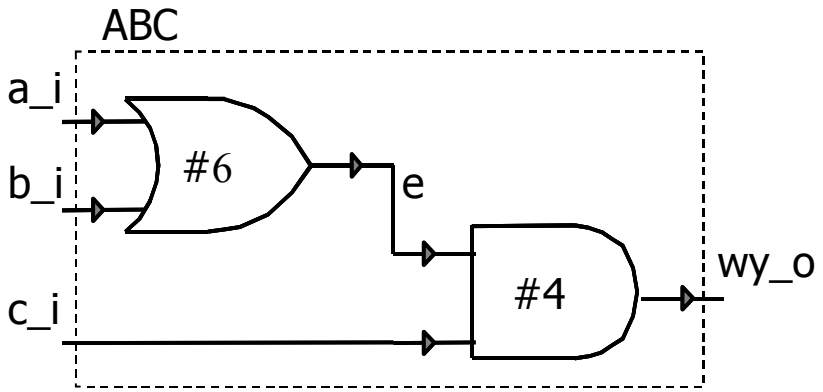
```
// definicja modułu ABC:
module ABC(wy_o, a_i, b_i, c_i);
  // deklaracje portów wej/wyj
output wy_o;
input a_i, b_i, c_i;
// sieć wewnętrzna:
wire e;
// bramki:
or #(6) a1(e, a_i, b_i);
and #(4) o1(wy_o, e, c_i);
endmodule
```



Verilog – Opóźnienia w bramkach przykład c. d.




```
module stimulus;
// deklaracja
// zmiennych:
reg A, B, C;
wire OUT;
ABC abc1(OUT, A, B, C); // osadzenie modułu ABC:
// pobudzanie wejść:
initial
begin
   A= 1'b0; B= 1'b0; C= 1'b0;
  #10 A= 1'b1; B= 1'b1; C= 1'b0;
  #10 A= 1'b1; B= 1'b1; C= 1'b1;
  #20 $finish;
end
endmodule
```



Verilog

Rozdział 6. Modelowanie na poziomie rejestrów (Dataflow)

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Przypisanie ciągłe:

- Sterowanie sieci wartością logiczną.
- Zastępuje bramki.
- Składnia:

  `assign` [<siła sygnału>][<opóźnienie>] <lista przypisań>;
- Definiowanie siły sygnału - opcjonalne (domyślnie **strong1** i **strong0**)
- Definiowanie opóźnienia – opcjonalnie (podaje się je jak w bramkach).

Cechy przypisania ciągłego:

- Cechy przypisania ciągłego:
- Lewa strona = sieć (skalar lub wektor), albo konkatenacja sieci skalarnych lub wektorowych.
- Przypisanie ciągłe jest zawsze aktywne.
- Wyrażenie obliczane jest zawsze, gdy któraś ze zmiennych z prawej strony zmieni swą wartość.
- Wynik jest natychmiast przesyłany do sieci określonej z lewej strony.
- Wyrażenia z prawej strony mogą być rejestrami, sieciami lub wywołaniami funkcji (skalary lub wektory).

Przykłady przypisania ciągłego:

```
assign c = a & b;
assign addr[15:0] = addr1_bits[15:0] ^
  addr2_bits[15:0];
// nawiasy klamrowe oznaczają
// łączenie (konkatenacja)
assign {c_out, sum[3:0]} = a[3:0] +
  b[3:0] + c_in;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Niejawne przypisanie ciągłe:

- Zamiast najpierw deklarować sieć a potem wykonywać do niej operację przypisania:

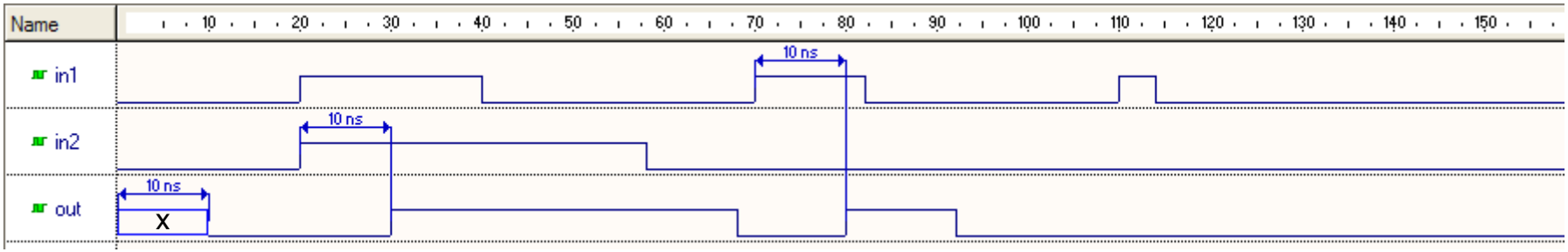
```
wire c;
assign c = a & b;
```

- można to zrobić krócej:

```
wire c = a & b;
```

Opóźnienia w przypisaniu ciągłym:

```
assign #10 c = a | b;
```



Uwaga: Impulsy krótsze niż 10 jedn. czasowych nie przechodzą na wyjście.

Verilog – Przypisanie ciągłe c. d.

- Niejawne określenie przypisania

```
wire out;
assign #10 c = a & b;
```

  - równoważne określenie:

```
wire #10 c = b & a;
```

- Określenie opóźnienia sieci
  - Można zdefiniować opóźnienie dla całej sieci - każda zmiana sygnału na sieci pojawi się z tym opóźnieniem.
  - Opis:

```
wire # 10 c;
assign c = a & b;
```

  - jest równoważny następującemu:

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Wyrażenia i operatory

Typ	symbol	operacja	ilość operandów
arytm.	*	mnożenie	2
	/	dzielenie	2
	+	dodawanie	2
	-	odejmowanie	2
	%	modulo	2
log.	!	negacja logiczna	1
	&&	logiczne and	2
	||	logiczne or	2
relacje	>	większe niż	2
	<	mniejsze niż	2
	>=	większe lub równe	2
	<=	mniejsze lub równe	2
równość	==	równość	2
	!=	różność	2
	===	równość warunkowa	2
	!==	różność warunkowa	2

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Wyrażenia i operatory c. d.

Typ	symbol	operacja	ilość operandów
bitowe	~	negacja bitowa	1
	&	and bitowe	2
	|	or bitowe	2
	^	xor bitowe	2
	^~ lub ~^	xnor bitowe	2
redukujące	&	redukujące and	1
	~&	redukujące nand	1
	|	redukujące or	1
	~|	redukujące nor	1
	^	redukujące xor	1
	^~ lub ~^	redukujące xnor	1
przesunięcie	>>	przesunięcie w prawo	2
	<<	przesunięcie w lewo	2
łączenie	{}	łączenie	dowolna
replikacja	{{}}	replikacja	dowolna
warunek	?:	warunek	3

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Operatory dwuargumentowe:

```
A=8'b00110101;     B=8'b00000010;
A*B    // mnożenie, wynik: 8'b01101010
A/B    // dzielenie, wynik: 8'b00011010
       // (część ułamkowa jest ucięta)
A+B    // dodawanie, wynik: 8'b00110111
A-B    // odejmowanie, wynik: 8'b00110011
```

Jeśli jakiś argument ma wartość **x**, to wynik też będzie miał wartość **x**.

- Operatory dwuargumentowe (cd.):
  - Operator *modulo* zwraca resztę z dzielenia dwóch liczb (działa podobnie jak w C).

```
15%4   // wynik: 3
15%5   // wynik: 0
-9%2   // wynik: -1 (znak z pierwszego argumentu)
9%-2   // wynik: 1 (znak z pierwszego argumentu)
```

- Operatory jednoargumentowe

```
-5  // MINUS 5
+6  // PLUS 6
```

  – Zaleca się **nie** używać ujemnych liczb w specyfikacji **<sss>'<podstawa> <nnn>**, gdyż może to prowadzić do złych wyników (liczby w tej postaci są reprezentowane jako dopełnienie do dwóch).

```
-'d10 / 5 // co oznacza: (2^32-10)/5
```

- Operatory logiczne:
  - && - logiczne **and**
  - || - logiczne **or**
  - ! - logiczne **not**
  - Operatory logiczne zwracają zawsze 1-bitową wartość: **0** (fałsz), **1** (prawda) lub **x** (niejednoznaczność).
  - Jeśli argument **nie jest** równy zero, jest traktowany jako **1** logiczna (prawda).
  - Jeśli argument **jest** równy zero, to jest traktowany jako **0** logiczne (fałsz).
  - Jeśli jakiś bit argumentu jest równy **z** lub **x**, to cały argument traktowany jest jako **x** (niejednoznaczność).

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Operatory logiczne (cd.)

```
A=2; B=0;
A&&B // wynik: 0
A||B // wynik: 1
!A   // wynik: 0
!B   // wynik: 1

A=2'bz0; B=2'b01
A&&B // wynik: x
(a==2)&&(b==3) // wynik: 1 jeśli a=2
               // oraz b=3, w przeciwnym
               // wypadku wynik: 0
```

Verilog – Wyrażenia i operatory c. d.

- Operatory porównania
  - Jeśli w wyrażeniu użyto operatorów porównania, wyrażenie zwraca wartość logiczne **1**, logiczne **0** lub **x**.

```
// A=5, B=4
// X=4'b1011, Y=4'b1110, Z=4'b0xxx
A<=B // wynik: 0
A>0  // wynik: 1
Y>=X // wynik: 1
Y<Z  // wynik: x
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Operatory równości:

wyrażenie	**opis**	**możliwe wartości**
a == b	a jest równe b. Wynik nieznany, gdy a lub b mają wartość **x** lub **z**.	**0,1,x**
a != b	a jest różne od b. Wynik nieznany, gdy a lub b mają wartość **x** lub **z**.	**0,1,x**
a === b	a jest równe b, włączając **x** i **z**	**0,1**
a !== b	a jest różne od b, włączając **x** i **z**	**0,1**

Operatory **===** i **!==** porównują bit po bicie i biorą pod uwagę również zgodność wartości **x** oraz **z**.

Verilog – Wyrażenia i operatory c. d.

- Operatory bitowe

~	**negacja** bit po bicie
&	**and** bit po bicie
|	**or** bit po bicie
^	**xor** bit po bicie
^~ lub ~^	**xnor** bit po bicie

and bitowe

	0	1	x
0	0	0	0
1	0	1	x
x	0	x	x

xnor bitowe

	0	1	x
0	1	0	x
1	0	1	x
x	x	x	x

or bitowe

	0	1	x
0	0	1	x
1	1	1	1
x	x	1	x

negacja

0	1
1	0
x	x

xor bitowe

	0	1	x
0	0	1	x
1	1	0	x
x	x	x	x

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Operatory redukujące
  - Operatory redukujące: **& (= and), ~& (= nand), | (= or), ~| (= nor), ^ (= xor), ^~ , ~^ (= xnor)** wymagają tylko jednego argumentu.
  - Operują na bitach wykonując odpowiednie operacje według tabel jak dla operatorów bitowych, działają od prawej do lewej.

```
// X=4'b1110
&X // 1&0&1&0 = 1'b0
|X // 1|0|1|0 = 1'b1
^X // 1^0^1^0 = 1'b1
```

  - Redukcja **xor** lub **xnor** może być wykorzystana przy obliczaniu ilości bitów parzystych lub nieparzystych w wektorze.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Operatory przesuwania
  - >> - przesunięcie w prawo
  - << - przesunięcie w lewo
  - Nowe pozycje są wypełniane zerami (ostatni bit nie przechodzi na początek!).

```
Y=X<<3 //przesuń X 3 bity w lewo
```


Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Verilog – Wyrażenia i operatory c. d.

- Operator łączenia
  - Umożliwia łączenie poszczególnych argumentów o znanych rozmiarach. Argumentami mogą być skalary (sieci, rejestry), wektory (sieci, rejestry), części wektorów (zakres bitów) i stałe o znanym rozmiarze.

```
// A=1'b1, B=3'b01, C=3'b11, D=3'b011
Y={B, C}
                    // wynik: Y=6'b0111

Y={A, B, C, D, 4'b1111}
                  // wynik: Y=12'b101110111111
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Wyrażenia i operatory c. d.

- Operator replikacji
  - Wielokrotne łączenie tego samego argumentu może być wyrażone za pomocą stałej oraz elementu powtarzanego w nawiasach klamrowych:

```
reg A;
reg [1:0] B, C;
reg [2:0] D;
A= 1'b1; B=2'b01; C=2'b10;
Y={4{A}}; // wynik: Y = 4'b1111
Y={ {4{A}}, {2{B}} }; // wynik: Y=8'b11110101
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Operator warunkowy
  - wymaga podania trzech argumentów:

    warunek ? wyrażenie_true : wyrażenie_false;
  - Najpierw obliczany jest warunek - jeśli wynik obliczeń jest **1**, to wykonywane jest **wyrażenie_true**, w przeciwnym wypadku **wyrażenie_false**. Jeśli wynik wyrażenia jest **x**, to obliczane są **wyrażenie_true** oraz **wyrażenie_false**, a następnie są porównywane bit po bicie. Jeśli bity się nie zgadzają, to w wyniku na tym miejscu będzie wartość bitu **x**.
  - Operator ten przypomina w pracy multiplekser lub polecenie *if-then-else*. Wykorzystuje się go często do przypisań warunkowych.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Verilog – Wyrażenia i operatory c. d.

- Operator warunkowy

```
// model bufora trójstanowego
assign data_bus = tristate ? 16'bz : data;

// model multipleksera 2-do-1
assign mux = addr ? A1 : A0;
```

– Operatory warunkowe mogą być zagnieżdżane:

```
assign out = (A==4) ? (addr ? A1 : A0) :
(ctr ? x : y);
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

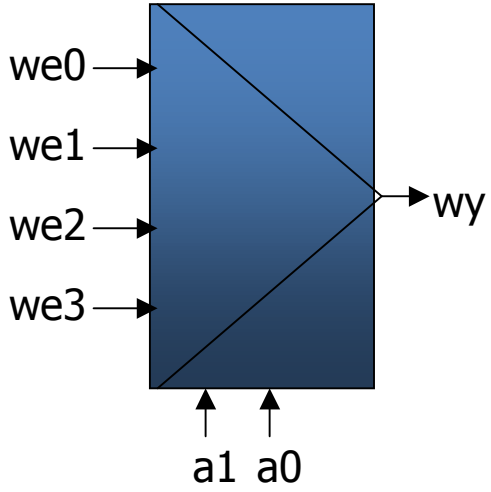
Priorytety operatorów:

Operator	Symbole	Priorytet
jednoargumentowe: mnożenie, dzielenie, modulo	+ - ! ~ * / %	najwyższy
dodawanie, odejmowanie przesunięcie	+ - << >>	
porównanie równość	< <= > >= == != === !==	
redukcja logiczne	& ~& ^ ^~ | ~| && ||	
Warunkowe	?:	najniższy

Verilog – Wyrażenia i operatory, przykład

- Multiplekser 4-do-1 (Sposób 1: równania logiczne)

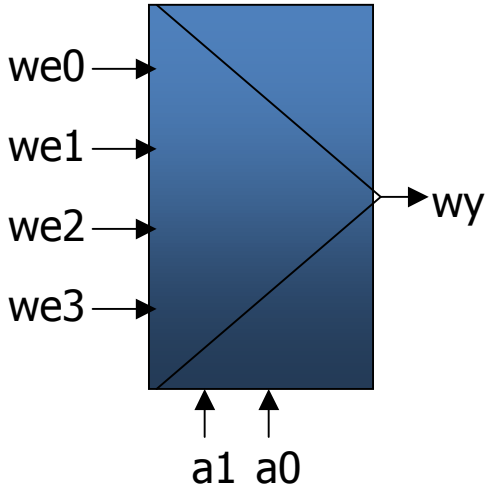
```
module mpx4_1 (wy, we0, we1, we2, we3, a1, a0);
   // Deklaracja portów:
   output wy;
   input we0,we1,we2,we3;
   input a1,a0;
   // Równania logiczne:
   assign wy =  ( a1 &  a0 & we3) |
                ( a1 & ~a0 & we2) |
                (~a1 &  a0 & we1) |
                (~a1 & ~a0 & we0);
endmodule
```



Verilog – Wyrażenia i operatory, przykład c. d.

- Multiplekser 4-do-1 (Sposób 2: operator warunkowy)

```
module mpx4_1 (wy, we0, we1, we2, we3, a1, a0);
  // Deklaracja portów:
  output wy;
  input we0,we1,we2,we3;
  input a1,a0;

  // Wykorzystanie
  // zagnieżdżonych
  // operatorów
  // warunkowych:
  assign owy = a1 ? (a0 ? we3 : we2) :
                    (a0 ? we1 : we0) ;
endmodule
```



Verilog

Rozdział 7. Modelowanie na poziomie behawioralnym

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Bloki **always** i **initial**
  - Polecenia w Verilog'u wykonywane są jednocześnie.
  - Każdy z bloków **always** i **initial** reprezentują osobny ciąg poleceń,
  - Rozpoczynają się w czasie **0**,
  - Nie mogą być zagnieżdżane.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Blok **initial**
  - Wykonuje się tylko jeden raz podczas symulacji, począwszy od czasu **0**.
  - Działanie każdego bloku **initial** kończy się niezależnie od pozostałych.
  - Instrukcje wewnątrz bloku muszą być zgrupowane za pomocą **begin** i **end**.
  - Jeśli w bloku występuje tylko jedno polecenie, grupowanie jest zbędne (podobnie jak w Pascalu *begin* i *end*, a w C { }).

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

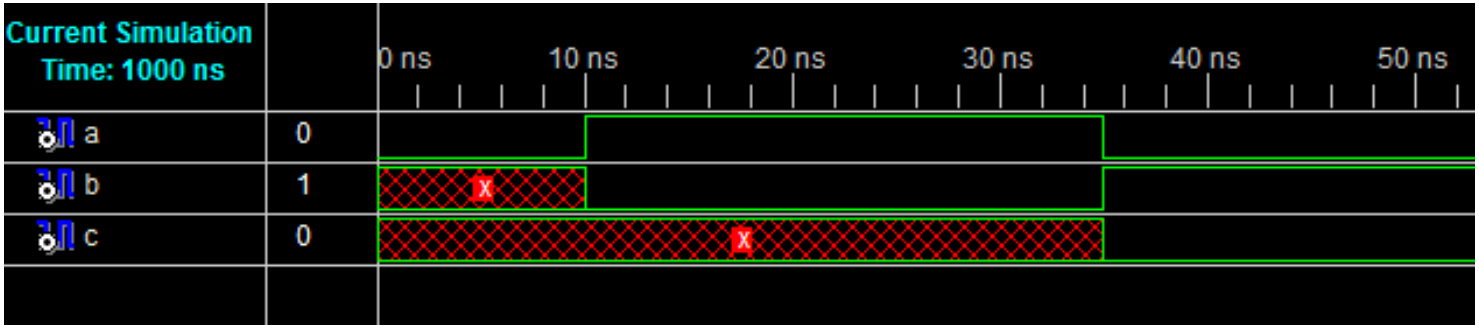
KAPITAŁ LUDZKI

- Blok **initial** (cd.):
  - Polecenia wykonywane są po kolei.
  - Opóźnienie czasowe **#<czas>** oznacza opóźnienie względem **aktualnego** czasu symulacji.
  - Bloki **initial** wykorzystuje się m.in. do inicjalizacji, monitorowania.

Verilog – Procedury strukturalne c. d.

- Blok **initial** (cd.):

```
`timescale 1ns / 1ps
module symulacja;
   reg a,b,c;
   initial
       a = 0;
   initial
   begin
       #10 a=1; b=0;
       #25 a=0; b=1; c=0;
   end
endmodule
```



- Blok **always**
  - Wszystkie bloki **always** rozpoczynają działanie równocześnie w czasie **0**.
  - Są od siebie niezależne.
  - Wykonują się w sposób ciągły (po zakończeniu ostatniej instrukcji wykonuje się pierwsza).

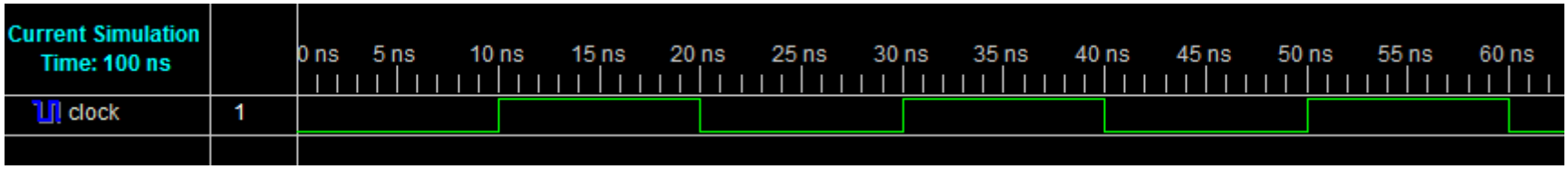
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Blok **always** - przykład

```
module clock_generator;
       reg clock;

       initial // = inicjalizacja zegara w czasie 0
         clock = 1'b0;

       always //przełączanie zegara co pół okresu:
         #10 clock = ~clock;

       initial
         #100 $finish;
endmodule
```



Verilog – Przypisanie proceduralne

- Uaktualnia wartości zmiennych typu **reg**, **integer**, **real** lub **time**.
- Wartości zmiennych nie ulegną zmianie, aż do następnego przypisania proceduralnego.
- Przypisanie odbywa się jednorazowo - w odróżnieniu od przypisania **assign**, które działa w sposób ciągły.
- **<lvalue> = <wyrażenie>** lub **<lvalue> <= <wyrażenie>**
- Lewa część przypisania może być:
  - zmienną typu **reg**, **integer**, **real**, **time**;
  - wybranym bitem z powyższych zmiennych;
  - zestawem bitów;
  - połączeniem powyższych elementów.
- Wyróżnia się dwa typy przypisań: **blocking** i **nonblocking**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

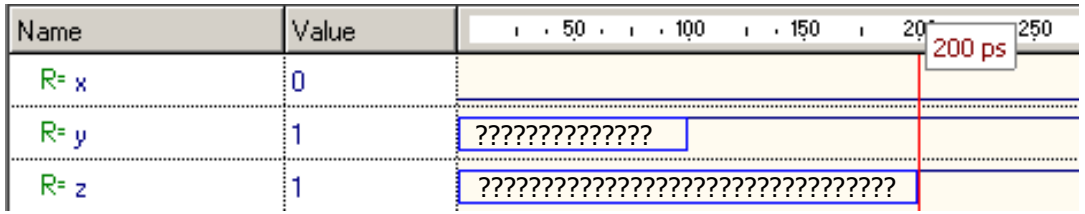
- Przypisanie typu **blocking (=)**
  - Wykonywane według kolejności ich umieszczenia w programie.
  - Nie wstrzymują one działania innych, równolegle działających, bloków.
  - Przypomnienie: Wszystkie wyrażenia behawioralne muszą być w bloku **initial** lub **always**.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Verilog – Przypisanie proceduralne c. d.

- Przypisanie typu **blocking (=)** (cd.)

```
module test;
  reg x, y, z;
  initial
    begin
      x = 1'b0;
      y = #100 1'b1;
      z = #100 1'b1;
    end
endmodule
```



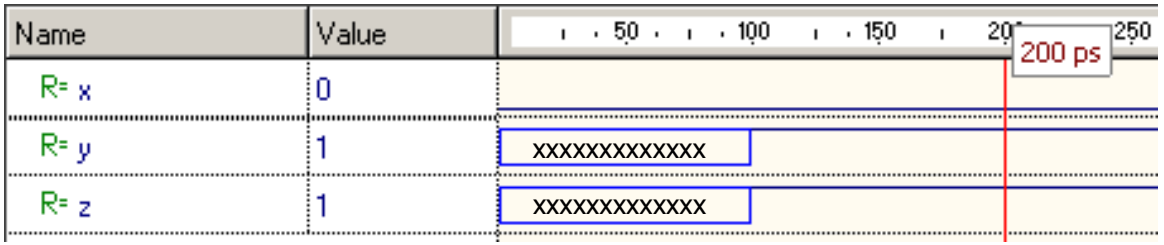
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- Przypisanie typu **nonblocking (<=)**
  - Pozwala na wykonanie przypisań bez wstrzymywania działania pozostałych instrukcji.
  - Symbol **<=** jest taki sam, jak operator warunkowy "mniejsze lub równe" - znaczenie rozstrzygane jest w zależności od kontekstu.
  - Wszystkie przypisania typu **nonblocking** będą wykonane równocześnie - tj. zostaną zaplanowane do wykonania po odpowiednim opóźnieniu. Zazwyczaj symulator wykonuje wszystkie przypisania **nonblocking** na końcu danego kroku czasowego.

Verilog – Przypisanie proceduralne c. d.

- Przypisanie **nonblocking (<=)** - przykład

```
module test;
  reg x, y, z;
  initial
    begin
      x <= 1'b0;
      y <= #100 1'b1;
      z <= #100 1'b1;
    end
endmodule
```



Verilog – Przypisanie proceduralne c. d.

- Różnica pomiędzy przypisaniami typu **blocking** i **nonblocking**.

```
//Przykład I: przypisania blocking
always @(posedge clock)
   x = y;

always @(posedge clock)
   y = x;
```

HAZARD! **x** i **y** będą miały taką samą wartość.

```
//Przykład II: przypisania nonblocking
always @(posedge clock)
   x <= y;

always @(posedge clock)
   y <= x;
```

Zamiana **x** i **y** odbędzie się prawidłowo.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

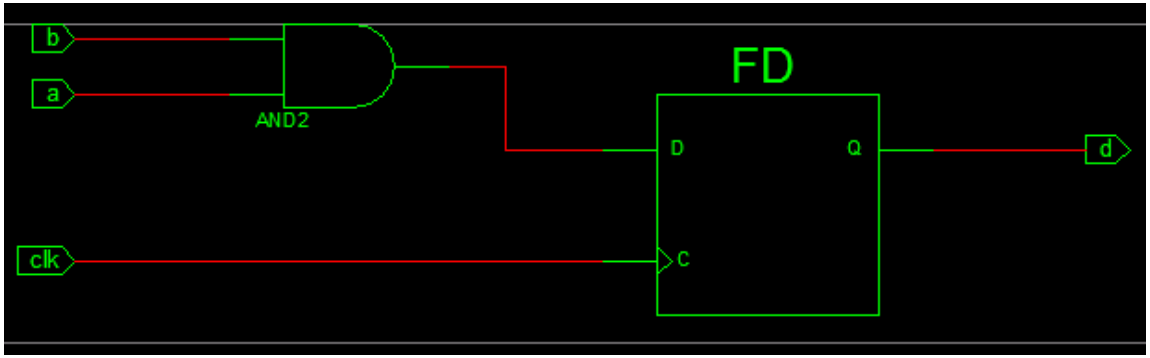
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Przypisanie proceduralne c. d.

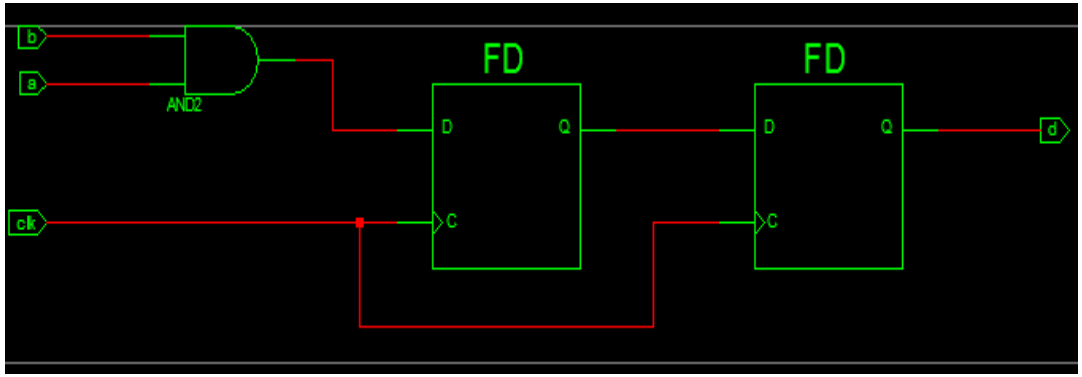
- Różnica pomiędzy przypisaniami typu **blocking** i **nonblocking (cd)**.

```
module test(clk, a, b, d);
   input a;
   input b;
   input clk;
   output d;

   reg c;
   reg d;

   always @(posedge clk)
       begin
              c = a & b;
              d = c;
       end
endmodule
```



- Różnica pomiędzy przypisaniami typu **blocking** i **nonblocking (cd)**.

```
module test(clk, a, b, d);
   input a;
   input b;
   input clk;
   output d;

   reg c;
   reg d;

   always @(posedge clk)
       begin
              c <= a & b;
              d <= c;
       end
endmodule
```



- Opóźnienia
  - **#** = czas pomiędzy „napotkaniem” wyrażenia przez symulator a jego wykonaniem.
  - **# <NUMBER>**
  - **# <identyfikator>**
  - **#(<min>:<typ>:<max>)**
  - Dla wyrażeń proceduralnych wyróżnia się trzy typy opóźnień:
    - zwykłe,
    - wewnętrzne,
    - zerowe.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opóźnienie zwykłe:

```
parameter opoznienie = 15;
parameter op_2 = 3;
reg a, b, c, d, e;
initial
begin
   a=0; // brak opóźnienia
   #8 b=1; // opóźnienie o 8 jedn.czasowych
   #opoznienie c=0; // opóźnienie określone parametrem
   #(opoznienie+op_2) d=1; // opóźn.określ.wyrażeniem
   #d e=e+1 // opóźnienie określone zmienną
   #(4:5:6) c=0 // mimnalne, typowe i maksymalne
end
```

– Dla grupy **begin-end**, opóźnienie jest odliczane zawsze względem czasu zakończenia poprzedniego polecenia w bloku.

- Opóźnienie wewnętrzne
  - opóźnienie przypisane do prawej części przypisania.

```
reg a, b, c; // definicja rejestrów

//opóźnienia wewnętrzne:
initial
begin
   a=0;
   b=0;
   c = #15 x+z; // pobierz wartość a i b w czasie=0,
                // oblicz a+b i poczekaj 15 jedn.
                // czasu a następnie zapisz wynik do c.
end
```

- Opóźnienie wewnętrzne (cd.)

```
// odpowiednik z wykorzystaniem opóźnień zwykłych oraz
// zmiennych tymczasowych
initial
begin
   a=0; b=0;
   temp_ab = a+b;
   #15 c=temp_ab; // pobierz wart. a+b w czasie bieżącym
                  // i zapisz w zmiennej tymczasowej.
end
```

- Opóźnienie zerowe
  - Gdy wyrażenia proceduralne w różnych blokach **always/initial** będą wykonane w tym samym kroku czasu => kolejność ich wykonania nie jest określona.
  - Opóźnienie zerowe = dana instrukcja wykona się jako ostatnia w danym kroku czasu.
  - Jeśli będzie więcej takich instrukcji, to będą one wykonane jako ostatnie, ale kolejność między nimi nie jest określona.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Opóźnienie zerowe (cd.)

```
initial
begin
    a = 0;
    b = 0;
end

initial
begin
    #0 a = 1;
    #0 b = 1;
end
```

  - Pod koniec czasu **0** zmienne **x** i **y** będą miały wartość **1**.

Pytanie

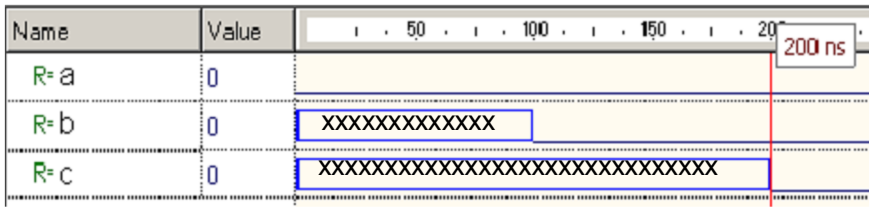
- Dane są dwa moduły:

```
module test;
  reg a, b, c;
  initial
    begin
      a = 1'b0;
      b = #100 a;
      c = #100 a;
    end
endmodule
```

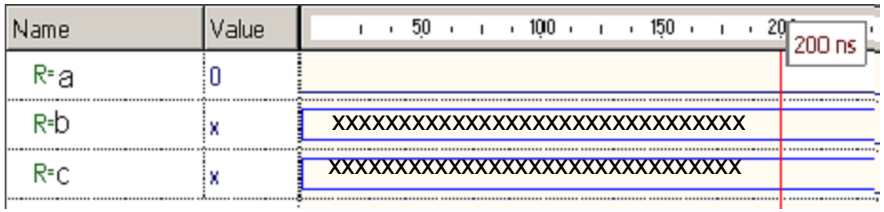
```
module test;
  reg a, b, c;
  initial
    begin
      a <= 1'b0;
      b <= #100 a;
      c <= #100 a;
    end
endmodule
```

- Jaki będzie wynik ich działania?

Odpowiedź




```
module test;
  reg a, b, c;
  initial
    begin
       a = 1'b0;
       b = #100 a;
       c = #100 a;
    end
endmodule
```




```
module test;
  reg a, b, c;
  initial
    begin
       a <= 1'b0;
       b <= #100 a;
       c <= #100 a;
    end
endmodule
```

- Sterowanie przy użyciu zdarzeń
  - Zdarzenie = zmiana wartości rejestru lub sieci.
  - Zdarzenia mogą być wykorzystane do wyzwolenia wykonania bloku instrukcji.
  - Są 4 typy sterowania zdarzeniem:
    - zwykłe sterowanie przy użyciu zdarzeń,
    - sterowanie przy użyciu nazwanych zdarzeń,
    - sterowanie przy użyciu zdarzeń typu OR,
    - sterowanie poziomem sygnału.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Sterowanie wykonaniem instrukcji c. d.

- Zwykłe sterowanie przy użyciu zdarzeń
  - Do sterowania zdarzeniami służy symbol @.
  - Instrukcje mogą reagować na:
    - zmianę wartości sygnału,
    - rosnące (**posedge**) zbocze,
    - opadające (**negedge**) zbocze.

```
@clock q = d; // poczekaj na zmianę clock i przypisz q=d
@(posedge clock) q = d; // q=d gdy clock zmienia się:
                        // 0->1,0->x,0->z,x->1,z->1
@(negedge clock) q = d; // q=d gdy clock zmienia się:
                        // 1->0,1->x,1->z,x->0,z->0
q=@(posedge clock) d;   // d jest oblicz.natychmiast,
                        // a wart.jest przypis.do q
                        // podczas rosn.zbocza clock.
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Sterowanie przy użyciu nazwanych zdarzeń
  - Wyrażenie nazywa się słowem kluczowym **event**.
  - Wyzwalanie zdarzenia: **->**.
  - Wykrywanie wyzwolenia zdarzenia: **@**.

```
event dane_gotowe;              // definicja zdarzenia

always @(posedge clk_i)     // wyzwolenie zdarzenia
begin
   if(ostatni_pakiet)     // jeśli to ostatni pakiet
     -> dane_gotowe;        // to wyzwól zdarzenie
end

always @(dane_gotowe)      // oczekuj na zdarzenie
  dane_wyjsciowe= dane_wejsciowe;
```

- Sterowanie przy użyciu zdarzeń typu OR
  - wyzwolenie może nastąpić poprzez jeden z kilku sygnałów:

```
// zatrzask wyzwalany poziomem z asynchr.resetem
always @(reset_i or zegar_i or dane_i)
  // czekaj na zmianę sygnału
  // reset_i, zegar_i lub dane_i
begin
  if (reset_i) //jeśli reset_i ma stan wysoki,
               // ustaw q_o=0
      q_o = 0;
  else if (zegar_i) //jeśli zegar_i ma stan wysoki
      q_o = dane_i;
end
```

Verilog – Sterowanie wykonaniem instrukcji c. d.

- Sterowanie <u>poziomem sygnału</u>
  - **wait** - oczekiwanie na określoną wartość sygnału.

```
always
   wait (licznik_en) #15 licznik = licznik + 1;
```

  - **licznik_en** jest monitorowane cały czas.
  - Jeśli **licznik_en** = 0, to wyrażenie nie jest wykonywane.
  - Jeśli **licznik_en** = 1 to **licznik = licznik + 1** zostanie wykonane po 15 jednostkach czasu i będzie powtarzane co 15 jednostek, dopóki wartość **licznik_en** ma wartość **1**.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Wyrażenie warunkowe „if”

- typ I - brak **else**:

```
if (<wyrażenie>) polecenie_true;
```

- typ II - jedno **else**:

```
if (<wyrażenie>) polecenie_true;  else
  polecenie_false;
```

- typ III – skomplikowane:

```
if (<wyrażenie1>) polecenie_true1;
else if (<wyrażenie2>) polecenie_true2;
else if (<wyrażenie3>) polecenie_true3;
else polecenie_domyślne;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Wyrażenie wyboru „case”

- Wyrażenie **case**

```
case (wyrażenie)
  alternatywa_1: polecenie1;
  alternatywa_2: polecenie2;
  alternatywa_3: polecenie3;
  ...
  ...
  default: polecenie_domyślne;
endcase
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Wyrażenie wyboru „case” c. d.

- Wyrażenie **case** (cd.)
  - Pierwsze dopasowanie spowoduje uruchomienie odpowiedniego polecenia.
- Wartości porównywane są bit po bicie, także wartości **x** i **z** są brane pod uwagę.
  - Jeśli nie zgadza się ilość bitów, to krótsze wyrażenie uzupełniane jest zerami, aby długość była identyczna.
  - Dopuszcza się oddzielenie przecinkiem kilku alternatyw.
  - Wielokrotne użycie części **default** nie jest dozwolone.
  - Wyrażenia **case** mogą być zagnieżdżone.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Przykład wyrażenia **case**:

```
reg [1:0] alu_ctrl;
...
...
case (alu_ctrl)
  2'd0 : wy = we1 - we2;
  2'd1 : wy = we1 + we2;
  default : $display(„Błędne polecenie");
endcase
```

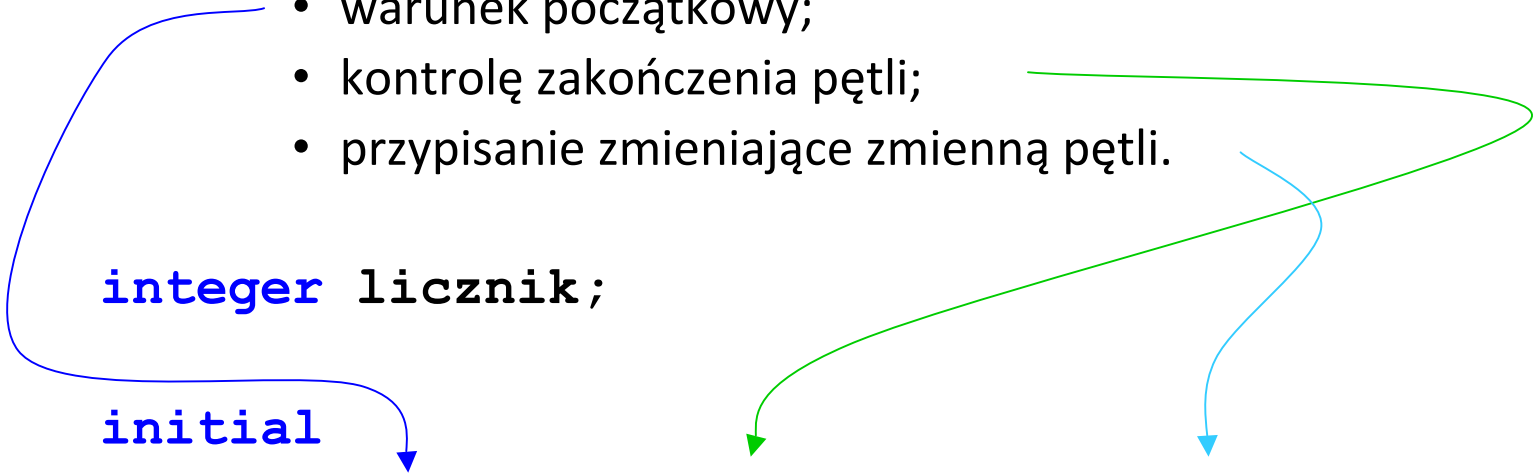
- Wyrażenie **casex** i **casez**
  - Są dwie modyfikacje wyrażenia case:
    - **casez** traktuje wszystkie wartości **z** jako nieznaczące. Wszystkie bity o wartości **z** można też zapisać za pomocą **?**.
    - **casex** traktuje wszystkie wartości **x** i **z** jako nieznaczące.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- Pętla **while**
  - Uruchamia się dopóki **wyrażenie** jest prawdziwe.
  - Jeśli **wyrażenie** jest fałszywe od początku, to pętla **while** nie uruchomi się w ogóle.

```
while (licznik < 20)
begin
   $display("Licznik jest równy: %d", licznik);
    licznik = licznik + 1;
end
```

- Pętla **for**
  - Pętla for zawiera trzy części:
    - warunek początkowy;
    - kontrolę zakończenia pętli;
    - przypisanie zmieniające zmienną pętli.



```
integer licznik;

initial
   for (licznik=0; licznik<20; licznik=licznik +1)
       $display("Licznik jest równy: %d",licznik);
```

Verilog – Pętle c. d.

- Pętla **repeat**
  - Wykonuje polecenie określoną ilość razy.
  - Musi zawierać liczbę powtórzeń, (stała, zmienna lub sygnał).
  - Ilość powtórzeń jest obliczana tylko raz na początku uruchamiania polecenia **repeat**.

```
repeat(20)
begin
   $diplay("Licznik jest równy: %d",licznik);
   licznik=licznik+1;
end
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Pętle c. d.

- Pętla **forever**
  - Wykonuje się aż do napotkania polecenia **$finish**.
  - Jest równoważna pętli **while(1)**.
  - Pętlę **forever** najczęściej używa się wraz z poleceniami kontrolującymi czas symulacji.
  - Gdyby ich nie było, pętla wykonywała by się w nieskończoność, wstrzymując pozostałe polecenia.

```
parameter okres = 20;
initial
begin
   clk_i = 1'b0;
   forever #(okres/2) clk_i = ~clk_i;
end
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Typy bloków: **sekwencyjne** i **równoległe**.
- Bloki **sekwencyjne**
  - Zgrupowanie poleceń w blok: **begin** i **end**.
  - Polecenia wykonywane są w kolejności, w jakiej są tam umieszczone.
  - Następne polecenie jest wykonywane, gdy zakończy się poprzednie (z wyjątkiem przypisań **nonblocking** w wewnętrznym opóźnieniem).
  - Jeśli określone jest opóźnienie lub kontrola wydarzeniem, jest ono określane względem czasu zakończenia poprzedniego polecenia w bloku.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- Bloki **równoległe**
  - Zgrupowanie poleceń w blok: **fork** i **join**.
  - Wyrażenia wykonywane są jednocześnie;
  - Kolejność wykonywania poleceń jest określona poprzez opóźnienia lub wydarzenia związane z konkretnymi poleceniami.
  - Opóźnienia liczone są względem czasu wejścia do bloku.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Bloki **równoległe** - przykład

```
reg a, b, c, d;

initial
fork
   a=0;              // wykonuje się w czasie=0
   #10 b=1;          // wykonuje się w czasie=10
   #20 c = a & b;    // wykonuje się w czasie=20
   #30 d = a | b;    // wykonuje się w czasie=30
join
```

  - Należy uważać na zjawisko hazardu, mogące wystąpić w bloku równoległym.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Bloki sekwencyjne i równoległe c. d.

- Cechy specjalne bloków
  - Bloki mogą być zagnieżdżane.
  - Można mieszać bloki sekwencyjne i równoległe.

```
// zagnieżdżanie bloków
initial
begin
   a = 0;
   fork
       #10 b = 1;
       #20 c = a & b;
   join
   #30 d = a | b;
end
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Bloki z nazwą
  - Bloki mogą mieć nadane nazwy.
  - Można deklarować zmienne lokalne dla bloków z nazwą.
  - Bloki z nazwą są częścią hierarchii projektu. Zmienne bloków mogą być dostępne przy wykorzystaniu pełnej nazwy hierarchicznej.
  - Bloki z nazwą mogą być dezaktywowane, tj. ich wykonywanie może być zatrzymane.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Bloki z nazwą - przykład

```
//bloki z nazwą:
module testbench;
initial
begin: blok_1  // blok sekwencyjny blok_1
  integer i;  // i jest zmienną statyczną i lokalną
              // dostęp do niej: testbench.blok_1.i
  ...
end

initial
fork: blok_2 // blok równoległy nazwany blok_2
  reg i;     // i jest zmienną statyczną i lokalną
  ...        // dostęp do niej: testbench.blok_2.i
join
```

- Dezaktywacja bloków z nazwą
  - Słowo **disable** powoduje zakończenie wykonywania poleceń bloku lub zadania.
  - W ten sposób można wychodzić z pętli, obsługiwać błędy, itp.
  - Wyłączenie bloku powoduje przejęcie kontroli nad programem przez polecenia występujące bezpośrednio za blokiem.
  - Wyłączenie bloku jest podobne do polecenia **break** w C.
  - Różnicą jest, że **break** powoduje przerwanie aktualnej pętli, a polecenie **disable** może przerwać wykonywanie dowolnego bloku.
  - Polecenie **disable** nie może kończyć działania funkcji (tylko zadanie).

Verilog – Bloki sekwencyjne i równoległe c. d.

- Przykład zastosowania polecenia **disable**:

```
module test;
  integer a, i;
  initial
    begin : blok_testowy1
      a = 1; // - ta instrukcja się wykona,
      disable blok_testowy1;
      a = 2; // - ta instrukcja nigdy się nie wykona!
    end
  initial
    begin : blok_testowy2
      i = 0;
      forever
        begin
          if (i==a) disable blok_testowy2;
          #1 i = i + 1;
        end
    end
endmodule
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog

Rozdział 8. Zadania i funkcje

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Cechy zadania:

- może uruchamiać inne zadania i funkcje,
- może wykonywać się przez czas niezerowy,
- może zawierać opóźnienia, wydarzenia i inne struktury kontroli czasowej,
- może mieć zero lub więcej argumentów typu **`input`**, **`output`** lub **`inout`**,
- nie zwraca wartości, ale może zwracać wartości poprzez argumenty typu **`output`** lub **`inout`**.

Cechy funkcji:

- może uruchamiać inną funkcję, nie może uruchamiać zadania,
- czas wykonywania funkcji jest zawsze zerowy,
- nie może zawierać opóźnień, wydarzeń ani innych struktur kontroli czasowej,
- musi posiadać przynajmniej jeden argument typu **`input`**,
- zawsze zwraca pojedynczą wartość.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Cechy wspólne zadań i funkcji:

- Muszą być zdefiniowane w module
- Są lokalne dla danego modułu.
- Mogą zawierać lokalne zmienne, rejestry, zdarzenia itp.
- Nie mogą zawierać zmiennych typu **`wire`**.
- Zawierają tylko wyrażenia behawioralne.
- Nie zawierają bloków **`initial`**/**`always`**, ale często są wywoływane przez te bloki.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Funkcje:

- Deklaruje się za pomocą słów kluczowych **`function`** i **`endfunction`**.
- Funkcje wykorzystuje się, gdy:
  - nie ma potrzeby opisu opóźnień lub zdarzeń,
  - zwracana jest dokładnie jedną wartość;
  - jest przynajmniej jeden argument wejściowy.

Deklaracja funkcji i jej wywołanie:

- Gdy zdeklarowano funkcję, automatycznie tworzony jest rejestr o takiej samej nazwie jak nazwa funkcji.
- Wartość funkcji jest zwracana w ten sposób, że ustawiany jest wspomniany rejestr.
- Funkcja nie może wywoływać innych zadań.
- Funkcja może wywoływać inne funkcje.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Przykład:

```
module tb;
reg [11:0] a;
reg [5:0] b;
// definicja funkcji sqr (zwraca 12-bit.wart.)
function[11:0] sqr;
    input[5:0] value;
    begin
          sqr = value * value;
    end
endfunction

always @(b)
   // wywołanie funkcji sqr
   a = sqr(b);

endmodule
```

Deklaracja zadania:

- Zadania deklaruje się za pomocą słów kluczowych **`task`** i **`endtask`**.
- Zadania wykorzystuje się, gdy:
  - potrzebne jest opóźnienie lub zdarzenie w procedurze;
  - procedura posiada zero lub więcej argumentów wyjściowych;
  - procedura nie posiada argumentów wejściowych.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Verilog – Zadania i funkcje c. d.

- Deklaracja zadania:

```
task <nazwa_zadania>;
  <deklaracja>*
  <polecenia>
endtask
```

  - Deklaracje: **input**, **output** lub **inout**.
  - Zmienne typu **input** i **inout** są przekazywane do zadania.
  - Zmienne typu **output** i **inout** są zwracane z powrotem po zakończeniu zadania.
  - Zadanie może wywoływać inne zadania lub funkcje.
  - Zadanie może operować na zmiennej typu **reg** zdefiniowanej w module.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Przykład deklaracji i wykorzystania zadania:

```
module tb;
reg [11:0] b;
reg [5:0] a;
// definicja zadania sqr
task sqr;
    input [5:0] value;
    output [11:0] sqr;
    begin
        sqr = value * value;
    end
endtask

always @(a)
   // wywołanie zadania sqr
   sqr(b,a);

endmodule
```

Verilog

Rozdział 9. Techniki modelowania

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Proceduralne przypisanie ciągłe

- Przypisanie proceduralne przypisuje wartość do rejestru. Rejestr zachowuje daną wartość, aż do następnego wpisu.
- Proceduralne przypisanie ciągłe - zmienne lub wyrażenia są w sposób ciągły podawane do rejestru przez określony okres czasu.
- Dwa typy przypisań
  - Typ 1: **assign** i **deassign**
  - Typ 2: **force** i **release**

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Typ 1: **`assign`** i **`deassign`**

- Lewa cześć przypisania może być tylko rejestrem lub połączeniem rejestrów.
- Nie może być częścią bitów z sieci ani też macierzą rejestrów.
- Przypisanie ciągłe ma większy priorytet niż zwykłe przypisania proceduralne.
- Przykład zastosowania:

```
always @(preset)
if (preset) assign q = 1'b1;
else deassign q;
```

Typ 2: `force` i `release`

- Pozwalają zapisywać wartości do rejestrów i sieci.
- Polecenia te wykorzystywane są głównie przy debuggingu.
- Rejestry:
  - **force** ma większy priorytet niż przypisania proceduralne czy ciągłe.
  - Po **release** wartość będzie pamiętana, ale będzie już można ją zmienić.
- Sieci:
  - Przypisanie **force** nadpisuje jakiekolwiek inne wartości.
  - Po **release** następuje powrót do normalnej wartości wymuszanej na sieci.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Proceduralne przypisanie ciągłe c. d.

Przykład:

```
always @(preset)
 if (preset)
   begin
      force FF.q = preset;
   end
 else
  begin
    release FF.q;
  end
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog

Rozdział 10. Parametry i skala czasu

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Parametry umożliwiają ustawienie pewnych wartości w modułach podczas ich powoływania do życia.
- Są dwie możliwości podawania parametrów:
  - poprzez polecenie **`defparam`**
  - podczas powoływania modułu

Polecenie `defparam`:

- Polecenie **`defparam`** umożliwia ustawienie parametrów w dowolnym module.
- W module może być wiele poleceń **`defparam`**.

```
module par_ex;
   parameter bin = 0;
   initial
       $display("Ten moduł ma parametr bin = %d",bin);
endmodule

module main;
   defparam EX1.bin = 1, EX2.bin = 2;
   par_ex EX1();
   par_ex EX2();
endmodule
```

Verilog – Parametry c. d.

- Ustawienie parametrów podczas powoływania do życia

```
module top;
        par_ex #(1) EX1();
        par_ex #(2) EX2();
endmodule
```

- W przypadku kilku parametrów, należy je wymienić po przecinku, w takiej kolejności, w jakiej zostały one zdefiniowane w definicji modułu:

```
par_example #(1,4,5) EX3();
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Skala czasu

- W jednym module czas w *ns* a w innym w *ms*.
- Verilog umożliwia definicję jednostki czasu dla modułu.

```
`timescale <jednostka_czasu>/<dokładność>
```

  - **<jednostka_czasu>** = jednostka czasu dla czasu symulacji oraz opóźnień.
  - **<dokładność>** = z jaką dokładnością zaokrąglane są opóźnienia podczas symulacji.
  - Do określenia obydwu tych wartości można wykorzystać tylko trzy wartości: 1, 10 i 100.
  - Przykłady:

```
`timescale 100 ns / 1 ns
`timescale 1 us / 10ns
```

Verilog
Rozdział 11. Verilog 2001, nowe elementy języka

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenie **generate** czyli wielokrotne osadzenie komponentu, przykład:

```
`define MAX_I 5
wire [1:`MAX_I] a, b, c, sum;
genvar i;
generate
   for(i=0; i<`MAX_I; i=i+1)
       begin:add_bit
          wire net1,net2,net3;
          xor g1 ( net1, a[i], b[i]);
          xor g2 (sum[i],net1, c[i]);
          and g3 ( net2, a[i], b[i]);
          and g4 ( net3, net1, c[i]);
          or g5 (c[i+1], net2, net3);
       end
endgenerate
```

Verilog – Polecenie generate c. d.

Dla polecenia **generate** można stosować osadzanie warunkowe, przykład:

```
generate
  if((a_width < 8) || (b_width < 8))
    CLA_multiplier #(a_width, b_width) u1 (a, b, product);
  else
    WALLACE_multiplier #(a_width, b_width) u1 (a, b, product);
endgenerate
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Nowy sposób indeksowania wektorów

```
[base +: width] //dodatni offset
[base -: width] //ujemny  offset
```

Przykład:

```
wire [7:0] dataN = word[byte_num*8 -: 8];
```

  - **base** może być zmienne
  - **width** musi być stałe

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Tablice wielowymiarowe

```
//1-wymiarowa tablica 8-bitowych zmiennych reg
//(dozwolone w Verilog-1995 oraz Verilog-2001)
reg [7:0] array1 [0:255];
wire [7:0] out1 = array1[address];
//3-wymiarowa tablica 8-bitowych zmiennych wire
//(nowość w Verilog-2001)
wire [7:0] array3 [0:255][0:255][0:15];
wire [7:0] out3 = array3[addr1][addr2][addr3];
```

Verilog – Tablice wielowymiarowe c. d.

- Indeksowanie zakresów w tablicach

```
reg [31:0] array2 [0:255][0:15];

wire [7:0] out2 = array2[100][7][31:24];
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Operacje na liczbach **signed**
  - **reg** oraz **net** mogą być zadeklarowane jako **signed**
  - Funkcja może zwracać wartości **signed**
  - Liczby **integer** mogą być zadeklarowane jako **signed**
  - Możliwa jest konwersja **signed** ←→ **unsigned**
  - Dodano operatory przesunięcia arytmetycznego

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Operacje na liczbach ze znakiem c. d.

- Operacje na liczbach **signed** (cd.)

```
reg signed [63:0] data;
wire signed [7:0] vector;
input signed [31:0] a;
function signed [128:0] alu;
```

```
16'hC501 //16-bitowa liczba hex unsigned
16'shC501 //16-bitowa liczba hex signed
```

```
reg [63:0] a; //wartość unsigned
always @(a) begin
  result1 = a / 2; //operacja na wartościach unsigned
  result2 = $signed(a) / 2;// operacja na wart. signed
end
```

```
// D=8'b10100011
D >> 3 //logical shift → 8'b00010100
D >>> 3 //arithmetic shift → 8'b11110100
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Inne zmiany

- Operator potęgowania **
- Domyślne reagowanie na wszystkie zdarzenia

```
always @* //dla układów kombinacyjnych
  if (sel)
    y = a;
  else
    y = b;
```

- Wymienianie sygnałów po przecinku:

```
always @(a or b or c or d or sel)
// jest równoważne:
always @(a, b, c, d, sel)
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Verilog – Inne zmiany c. d.

- Połączenie deklaracji portu z typem danych
  - Nie trzeba osobno deklarować **input** a potem **reg**

```
module mux8 (y, a, b, en);
  output reg [7:0] y;
  input wire [7:0] a, b;
  input wire en;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Deklaracja portów podobna do ANSI-C

```
module mux8 (output reg [7:0] y,
             input wire [7:0] a,
             input wire [7:0] b,
             input wire en );
```

Część III - VHDL

Modelowanie, symulacja i projektowanie układów cyfrowych z wykorzystaniem języka VHDL

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Plan prezentacji
  - Pojęcia podstawowe
    - Typy danych
    - Biblioteki
    - Stałe, sygnały
    - Osadzanie komponentów
    - Polecenie **generate**
  - Poziom przesłań międzyrejestrowych RTL
    - Współbieżne przypisanie proste
    - Operatory
    - Opóźnienia
    - Procesy
    - Zmienne
  - Poziom behawioralny
    - Wyrażenia **if**, **case**, **loop**
    - Podstawowe rodzaje procesów
    - Opóźnienia typu **wait**
  - Funkcje i procedury

VHDL - Literatura

- A. Yalamanchili, „Introductory VHDL: From Simulation to Synthesis”, Prentice-Hall, 2001.
- K. Skahill, „Język VHDL, Projektowanie programowalnych układów logicznych”, WNT, 2001.
- D. Pellerin, D. Taylor, „VHDL Made Easy!”, Prentice Hall, 1997.
- D. E. Ott, T. J. Wilderotter, „A Designer’s Guide to VHDL Synthesis”, Kluwer Academic Publishers, 1994.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Język VHDL

Pojęcia podstawowe

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Geneza powstania języka

- VHDL = **V**ery High Speed Integrated Circuits **H**ardware **D**escription **L**anguage,
- Prace badawcze 1983-1985, IBM, Intermetics oraz Texas Instruments, sponsorowane przez Departament Obrony USA.
- 1987: standard IEEE 1076-1987
- 1993: znowelizowano standard 1076-1993.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Geneza powstania języka (cd.)

- Rozszerzenia języka VHDL:
  - **1076.1** – VHDL-AMS
  - **1076.2** – **math_real**, **math_complex**
  - **1076.3** – **numeric_bit**, **numeric_std**
  - **1076.4** – **vital**
  - **1164** – **std_logic**

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Poziomy opisu

Verilog

- *Poziom kluczy*
- Poziom bramek logicznych
- Poziom rejestrów (Dataflow)
- Poziom behawioralny

VHDL:

- Poziom strukturalny
- Poziom przesłań międzyrejestrowych RTL
- Abstrakcyjny poziom behawioralny

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- **Nie** są rozróżniane duże i małe litery

  `ENTITY` = `EntitY`

  `TRUE = true`

  ale: `'Z' ≠ 'z'`

- Komentarz:

```
RESET : in std_logic; -- Zerowanie o szerokości
                      -- dwu mikrosekund
```

- Identyfikatory:
  - nazwa musi zaczynać się od litery,
  - można używać podkreślenia (_)
  - przerwy (spacje) nie są dozwolone w nazwach.

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- Słowa zarezerwowane

```
abs, access, after, alias, all, and,
architecture, array, assert, attribute, begin,
block, body, buffer, bus,  case, component ,
configuration, constant , disconnect, downto,
else, elsif, end, entity, exit, file, for,
function, generate, generic, guarded, if, in,
inout, is, label, library, linkage, loop, map,
mod, nand, new, next, nor, not, null, of, on,
open, or, other, out, package, port, procedure,
process, range, record, register, rem, report,
return, select, severity, signal, subtype, then,
to, transport, type, unit, until, use, variable,
wait, when, while, with, xor
```

Składnia języka i typy danych (cd.)

- Literały całkowite:
  - podstawa dziesiętna: **`170, 1_7_0, 10#170#`**
  - podstawa dwójkowa: **`2#1010_1010#`**
  - podstawa ósemkowa: **`8#252#`**
  - podstawa szesnastkowa: **`16#AA#`**
- Literały znakowe: **`'a', 'A'`**
- Symbole: **`ZIELONY STAN0`**
- Stałe łańcuchowe: **`"abc", "ABC"`**

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Składnia języka i typy danych (cd.)

- Literały łańcuchowo-bitowe:

```
"1001_1001"
B"1001_1001"
O"167"
X"abC"
```

- Literały fizyczne:

```
1.5 ns
2 kOhm
60 Hz
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Typ wyliczeniowy

```
type <Nazwa_typu> is ( <wartość_1>, <wartość_2>, ...,
   <wartość_n>);
```

Przykłady:

```
type t_KOLOR is (NIEBIESKI, ZIELONY, CZERWONY);
type t_MOJTYP is ('0', '1', 'U', 'Z');

variable x_v : t_KOLOR;
signal a: t_MOJTYP;

x_v := NIEBIESKI;
a <= 'Z';
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Składnia języka i typy danych (cd.)

- Typ całkowity
  - Wbudowany, nienazwany, typ całkowity; zakres od:

    $-(2^{31}-1)$ do $2^{31}-1$ (tj. -2 147 483 647 ... 2 147 483 647).
  - Definicja własnego typu całkowitego:

```
type <Nazwa_typu> is range <Zakres_integer>
```

Przykłady:

```
type integer is range -2147483647 to 2147483647;
type PROCENT is range -100 to 100;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

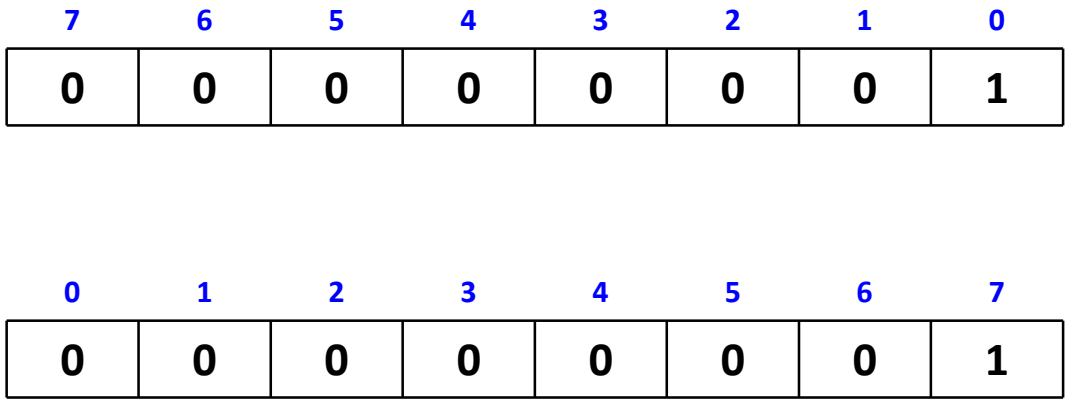
Składnia języka i typy danych (cd.)

- Typy tablicowe - Ograniczony typ tablicowy

```
type <Nazwa_typu_tablicowego> is array (<Zakres_całkowity>) of
   <Typ_elementu_tablicy>
```

  – Przykład:

```
type t_A is array (7 downto 0) of std_logic;
type t_B is array (0 to 7)     of std_logic;
signal reg_a : t_A;
signal reg_b : t_B;
```



- Typy tablicowe - Nieograniczony typ tablicowy

```
type <Nazwa_typu_tablicowego> is array
   (<Nazwa_typu_całkowitego> range <>) of
   <Typ_elementu_tablicy>
```

  - Przykłady:

```
type bit_vector is array (integer range <>) of bit;
type std_logic_vector is array (integer range <>) of
   std_logic;
...
signal moj_wektor1 : std_logic_vector (5 downto -5);
signal mój_wektor2 : std_logic_vector (1 to 11);
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Typy tablicowe - Dostęp do elementów tablicy

```
<identyfikator_tablicy>(<wyrażenie>)
```

  - Przykłady:

```
tab(6);
tab(x);
tab(3 to 6);
```

- Stałe tablicowe

```
type WEKTOR4 is array (1 to 4) of bit;
signal X: WEKTOR4;
X(1) <= '1'; -- Podstawienie poszczególnych elementów
X(2) <= '1';
X(3) <= '0';
X(4) <= '0';
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Stałe tablicowe (cd.)

```
X <= ('1', '1', '0', '0');
X <= (2 => '1', 3 => '0', 1 => '1', 4 => '0');
X <= ('1', '1', others => '0');
X <= (3 => '0', 4 => '0', others => '1');
X <= (3 to 4 => '0', 2 downto 1 => '1');
X <= (3 to 4 => '0', others => '1');
X <= (others => '1');
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Atrybuty tablic

```
signal moj_wektor : std_logic_vector (5 downto -5);
```

Wyrażenie zawierające atrybut	Wartość
`moj_wektor'left`	`5`
`moj_wektor'right`	`-5`
`moj_wektor'high`	`5`
`moj_wektor'low`	`-5`
`moj_wektor'length`	`11`
`moj_wektor'range`	`(5 downto -5)`
`moj_wektor'reverse_range`	`(-5 to 5)`

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Składnia języka i typy danych (cd.)

- Tablice wielowymiarowe
  - Syntezowalne tablice dwuwymiarowe :

```
type t_BAJT   is array (7 downto 0) of std_logic;
type t_WEKTOR is array (3 downto 0) of t_BAJT;
type t_WEKTOR is array (3 downto 0) of std_logic_vector(7
   downto 0);
```

  - Tablice wielowymiarowe (niesyntezowalne):

```
type t_MULTI is array ( 7 downto 0, 255 downto 0) of std_logic;
```

- Typ łańcuchowy string

```
signal NAPIS : string (1 to 8);
...
NAPIS <= "Kowalski";
```

- Typ *bit_vector*

```
signal DATA : bit_vector(7 downto 0);
...
DATA <=  "1011_1100";
DATA <= B"1011_1100";
DATA <= O"5274";
DATA <= X"ABC";
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Rekordy

```
type BYTE_AND_IX is
record
  BYTE: BYTE_VEC;
  IX: integer range 0 to 8;
end record;

. . .
X.BYTE <= "11110000";
X.IX <= 2;
DATA <= X.BYTE;
NUM <= X.IX;
Z <= X;
```

- Typ rzeczywisty

```
variable x_v : real;
...
x_v := 1.234;
```

- Typ fizyczny

```
wait for 10 ns;
```

Typy predefiniowane - przykładowa zawartość pakietu standardowego **standard**:

```
package standard is
  type boolean is (FALSE, TRUE);
  type bit is ('0', '1');
  type integer is range -2147483647 to 2147483647;
  subtype natural is integer range 0 to 2147483647;
  subtype positive is integer range 1 to 2147483647;
  type character is ( <tu wymieniono wszystkie 128
                       znaków ASCII> );
  type string is array (positive range <>) of character;
  type bit_vector is array (natural range <>) of bit;
end standard;
```

- Podtypy

  Deklaracja podtypu:

```
type bit_vector is array (integer range <>) of bit;
subtype WEKTOR4 is bit_vector(0 to 3);

type MÓJ_TYP is ('0', '1', 'U', 'Z');
subtype MÓJ_TYP2 is MÓJ_TYP range '0' to '1';
```

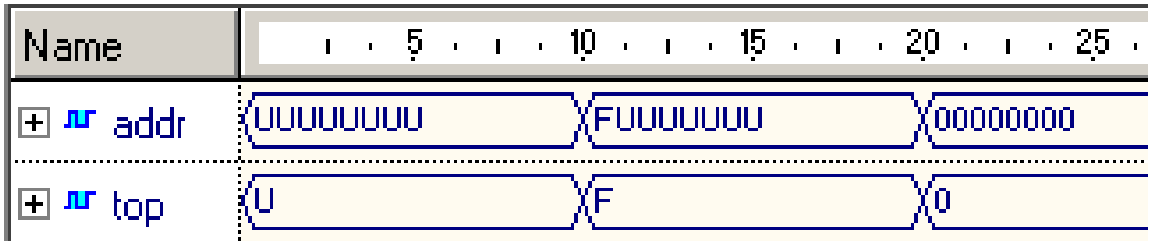
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Aliasy

```
signal addr : std_logic_vector(31 downto 0);
alias top: std_logic_vector (3 downto 0) is addr (31 downto
   28);
...
  wait for 10 ns;
  top <= "1111";
  wait for 10 ns;
  addr <= (others => '0');
```



- Konwersja typów

```
type t_T1 is range 0 to 255;
signal a1 : t_T1;
signal a2 : integer range 0 to 255;
if a1 = a2 then -- =błąd składniowy (type mismatch,
...             --operator not defined for such
                --operands).

-- JAK ROZWIĄZAĆ TAKI PROBLEM?
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI CZŁOWIEK – NAJLEPSZA INWESTYCJA!

- Konwersja typów (cd.)

```
-- ROZWIĄZANIE PROBLEMU:
type t_T1 is range 0 to 255;
signal a1 : t_T1;
signal a2 : integer range 0 to 255;

if a1 = t_T1(a2) then -- nie ma błędu!
...
```

Biblioteki

```
library <Nazwa_biblioteki>;
use <Nazwa_biblioteki>.<Nazwa_pakietu>.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
```


```
library IEEE;
use IEEE.std_logic_1164.my_func
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Struktura bibliotek

- Biblioteka **std**:
  - Pakiet **standard**
  - Pakiet **textio**
- Biblioteka **IEEE**:
  - Pakiet **std_logic_1164**
  - Pakiet **numeric_std**
  - Pakiet **numeric_bit**
  - Pakiet **std_logic_arith**
  - Pakiet **std_logic_unsigned**
  - Pakiet **std_logic_signed**
  - Pakiet **std_logic_textio**

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Pakiet **standard** – typy:
  - **bit**
  - **boolean**
  - **integer**
  - **character**
  - **real**
  - **time**
  - **string**
  - **bit_vector**

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Biblioteka IEEE

- Pakiet **std_logic_1164**:

```
library IEEE;
use IEEE.std_logic_1164.all;
```

- **U** = stan niezainicjowany (*uninitialized*)
- **X** = stan nieznany (*unknown*)
- **0** = stan logiczny '0'
- **1** = stan logiczny '1'
- **Z** = stan wysokiej impedancji dla sygnału trzystanowego (*tri-state*)
- **W** = stan nieznany dla sygnału niskoobciążalnego (*weak unknown*)
- **L** = stan niskiej rezystancji (dla wyjść typu otwarty emiter) (*weak* '0')
- **H** = stan wysokiej rezystancji (dla wyjść typu otwarty kolektor) (*weak* '1')
- **-** = stan nieistotny (*don't care*)

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Biblioteka IEEE (cd.)

- Pakiet **std_logic_1164** – funkcja arbitrażowa:

	'U'	'X'	'0'	'1'	'Z'	'W'	'L'	'H'	'-'
'U'	'U'	'U'	'U'	'U'	'U'	'U'	'U'	'U'	'U'
'X'	'U'	'X'	'X'	'X'	'X'	'X'	'X'	'X'	'X'
'0'	'U'	'X'	'0'	'X'	'0'	'0'	'0'	'0'	'X'
'1'	'U'	'X'	'X'	'1'	'1'	'1'	'1'	'1'	'X'
'Z'	'U'	'X'	'0'	'1'	'Z'	'W'	'L'	'H'	'X'
'W'	'U'	'X'	'0'	'1'	'W'	'W'	'W'	'W'	'X'
'L'	'U'	'X'	'0'	'1'	'L'	'W'	'L'	'W'	'X'
'H'	'U'	'X'	'0'	'1'	'H'	'W'	'W'	'H'	'X'
'-'	'U'	'X'	'X'	'X'	'X'	'X'	'X'	'X'	'X

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Biblioteka IEEE (cd.)

- Pakiet **std_logic_arith** (cd.)

```
library IEEE;
use IEEE.std_logic_arith.all;

type unsigned is array (natural range <>) of std_logic;
type signed is array (natural range <>) of std_logic;
```

ARG1	op	ARG2	UNSIGNED	SIGNED	bit_vector
"000"	=	"000"	true	true	true
"00"	=	"000"	true	true	false
"100"	=	"0100"	true	false	false
"000"	<	"000"	false	false	false
"00"	<	"000"	false	false	true
"100"	<	"0100"	false	true	false

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

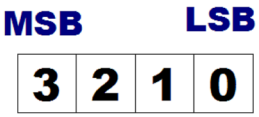
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Biblioteka IEEE (cd.)

- Pakiet **std_logic_arith** (cd.)


```
variable X: unsigned(1 to 8);
-- 8-bitowa liczba,
-- X(X'left) = X(1) - najbardziej znaczący bit
```




```
signal Y: unsigned(3 downto 0)
-- 4-bitowa liczba
-- Y(Y'left) = Y(3) - najbardziej znaczący bit
```

Biblioteka IEEE (cd.)

- Pakiet **std_logic_arith** (cd.)

```
signed'("0101") -- +5
signed'("1011") -- -5
```

  - Funkcje konwersji:

```
integer
unsigned
signed
std_ulogic
```

```
conv_integer(?) → integer;
conv_unsigned(?;SIZE) → unsigned;
conv_signed(?;SIZE) → signed;
conv_std_logic_vector(?; SIZE) → std_logic_vector
```

SIZE = integer

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Biblioteka IEEE (cd.)

- Pakiet **std_logic_arith** (cd.)

```
process
  variable a_v, b_v, res_v: unsigned (7 downto 0);
  variable sum_v: unsigned (8 downto 0);
  variable carry_v: std_logic;
begin
  sum_v := conv_unsigned(a_v, 9) + b_v;
  res_v := sum_v(7 downto 0);
  carry_v := sum_v(8);
end process;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Biblioteka IEEE (cd.)

- Pakiet **std_logic_arith** (cd.)

```
function shl(ARG: unsigned;
             COUNT: unsigned) return unsigned;
function shl(ARG: signed;
             COUNT: unsigned) return signed;
function shr(ARG: unsigned;
             COUNT: unsigned) return unsigned;
function shr(ARG: signed;
             COUNT: unsigned) return signed;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Biblioteka IEEE (cd.)

- Pakiet **std_logic_unsigned**
  - Umożliwia wykonywanie operacji
    - arytmetycznych,
    - konwersji
    - porównywania

    dla wartości typu **std_logic** traktowanych jako liczby całkowite bez znaku (**unsigned**).

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Pakiet **std_logic_signed**
  - Umożliwia wykonywanie operacji:
    - arytmetycznych,
    - konwersji
    - porównywania

  dla wartości typu **std_logic** traktowanych jako liczby całkowite ze znakiem (**signed**) zakodowanych jako uzupełnienie do 2.

- Tworzenie własnego pakietu

```
package mój_pakiet is
...
-- deklaracje typów, stałych,
-- deklaracje funkcji i procedur
...
end package mój_pakiet;

package body mój_pakiet is
...
-- deklaracje i definicje (implementacje)
-- funkcji i procedur
...
end package mój_pakiet;
```

Język VHDL

Poziom strukturalny

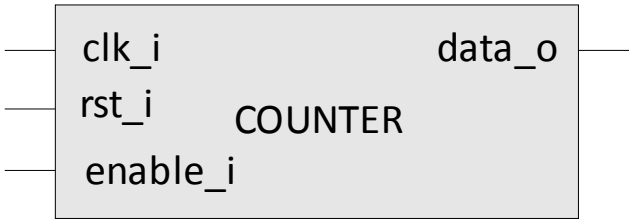
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Jednostki projektowe (**entity**)


```
entity COUNTER is
  port(clk_i  : in bit;
       rst_i  : in bit;
     enable_i : in bit;
       data_o : out bit_vector(31 downto 0)
       );
end entity;
-- lub: end COUNTER;
-- lub: end entity COUNTER;
```



UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Architektury jednostek (**architecture**)


```
architecture <Nazwa_architektury> of <Nazwa_entity> is
    <Deklaracje>
        ......
begin
    <Polecenia_współbieżne>
            ......
end architecture;
-- lub end <Nazwa_architektury>
-- lub end architecture <Nazwa_architektury>
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Stałe

- Stałe można definiować w
  - blokach **entity**,
  - architekturach,
  - pakietach,
  - procesach,
  - podprogramach.
- Stałe zadeklarowane w architekturze są widziane tylko wewnątrz architektury.

```
constant WIDTH: integer := 8;
constant ROM: ROM_TYPE := ("0000", "0011", "1100");
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Sygnały

- Do czego służą sygnały?
- Cechy sygnałów
- Miejsca deklaracji sygnałów

```
signal A, B: std_logic;
signal COUNT: integer := 1;
```

Osadzanie komponentów

- Deklaracja komponentu:

```
component <Nazwa_komponentu>
  port (<Deklaracje_portów>);
end component;
```

- Przykład deklaracji komponentu – bramki **nand2**:

```
component nand2 is
  port (a : in bit;
        b : in bit;
        y : out bit);
end component;
```

```
entity NAND2
  port (A, B : in bit,
        Y    : out bit);
end entity;

architecture beh of NAND2 is
...
begin
...
end architecture;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

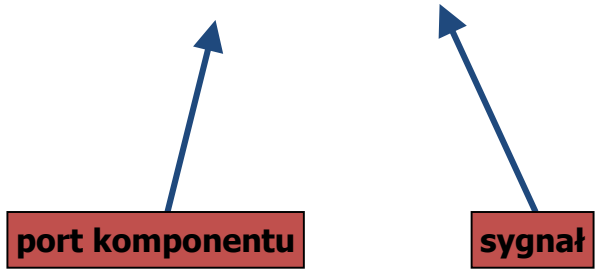
Osadzanie komponentów (cd.)

- Mapowanie portów
  - Sposób uproszczony (kolejności końcówek jak w deklaracji):

```
B1 : nand2 port map (in1, in2, out1);
```

  - Sposób II (kolejność jest nieistotna):

```
B1 : nand2
  port map (y => out1, a => in1, b => in2);
```



Osadzanie komponentów (cd.)

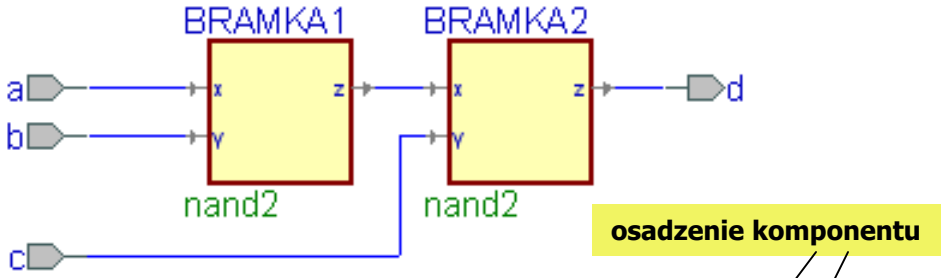

```
entity main is port (
     a    : in bit;
     b    : in bit;
     c    : in bit;
     d    : out bit
     );
end entity;

architecture beh of main is
  signal internal : bit;
  component nand2 is
    port (x, y : in bit;
          z    : out bit);
  end component;
begin
  BRAMKA1: nand2 port map (x => a, y => b, z => internal);

  BRAMKA2: nand2 port map (x => internal, y => c, z => d);
end architecture;
```

```
entity nand2 is
  port (x, y : in bit;
        z    : out bit);
end entity;

architecture beh of nand2 is
begin
  z <= x and y;
end architecture;
```

deklaracja komponentu

część deklaracyjna

część współbieżna



osadzenie komponentu

Polecenie **generate**

- Polecenie **for-generate**
- Polecenie **if-generate**

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

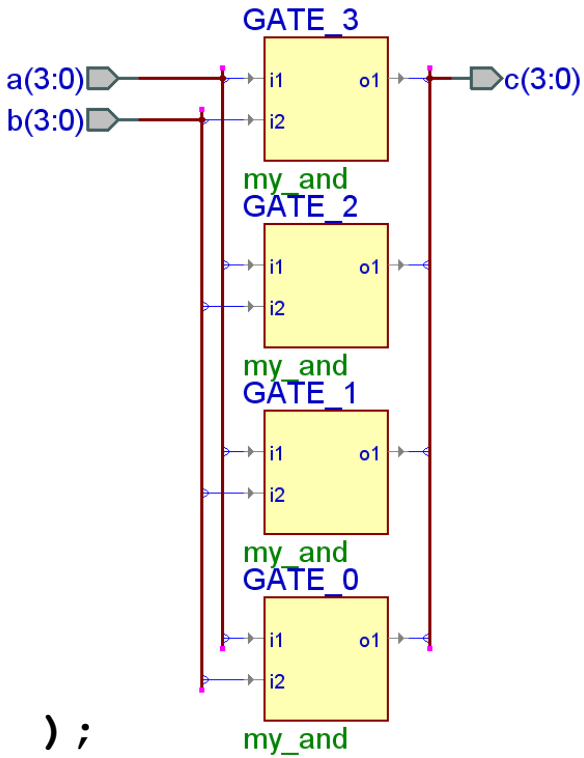
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

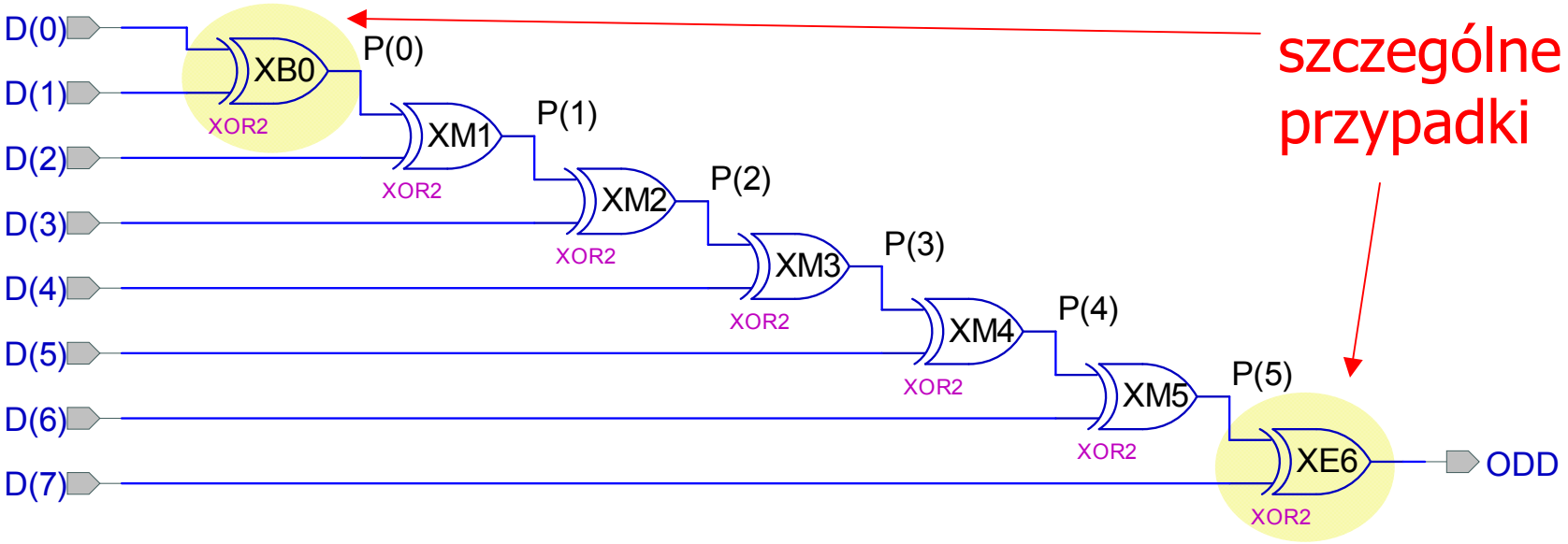
Polecenie **generate** (cd.)

- Polecenie **for-generate**

```
entity test is
  port (a, b : in  bit_vector(3 downto 0);
    c : out bit_vector(3 downto 0) );
end entity;

architecture rtl of test is
  component my_and
    port (
      i1 : in  bit;
      i2 : in  bit;
      o1 : out bit);
  end component;
begin
  AND_BLOCK: for i in 0 to 3 generate
    GATE : my_and port map (a(i), b(i), c(i) );
  end generate;
end architecture;
```



- Polecenie **if-generate**



UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

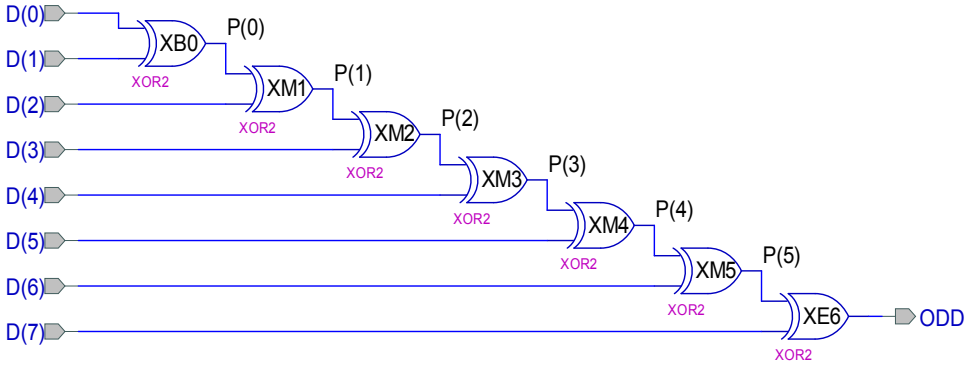
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenie **generate** (cd.)

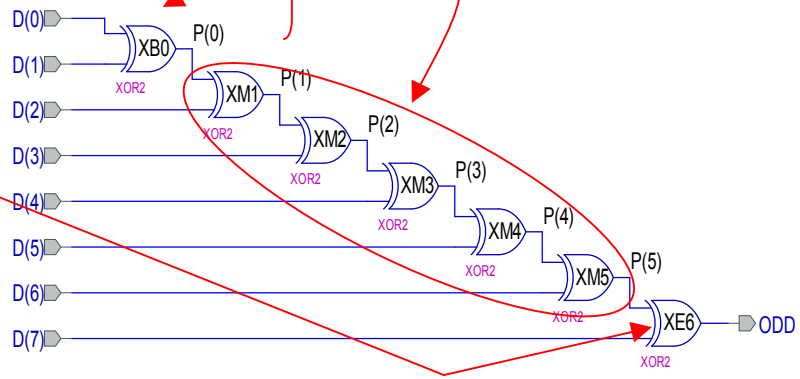
- Polecenie **if-generate**

```
entity parity8 is
  port (d : in  bit_vector(0 to 7);
    odd : out bit );
end entity;

architecture rtl of parity8 is
  component xor2
    port (a, b : in  bit;
      y : out bit);
  end component;
  signal p: bit_vector(d'low to d'high - 2);
...
```



Polecenie **generate** (cd.)

```
begin
  G: for i in d'low to d'high-1  generate
    G_BEG: if i=d'low generate
      XB: xor2 port map (a => d(d'low),
        b => d(d'low+1),
        y => p(d'low) );
    end generate;
    G_MIG: if i > d'low and i < d'high-1 generate
      XM: xor2 port map (a => p(i-1),
        b => d(i+1),
        y => p(i) );
    end generate;
    G_END: if i = d'high-1 generate
      XE: xor2 port map (a=>p(i-1),
        b=>d(i+1),
        y => odd );
    end generate;
  end generate;
end architecture;
```



Parametry bloku **entity** (**generic**)

- Przykład:

```
entity ANDX is
  generic (W : positive);
  port( A : in  bit_vector(W-1 downto 0);
        B : in  bit_vector(W-1 downto 0);
        Y : out bit_vector(W-1 downto 0) );
end entity;

architecture RTL of ANDX is

begin
  G: for I in W-1 downto 0 generate
    Y(I) <= A(I) and B(I);
  end generate;
end architecture;
```

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

Parametry bloku **entity** (**generic**) (cd.)


```
entity AND8 is
  port (A8 : in  bit_vector(7 downto 0);
        B8 : in  bit_vector(7 downto 0);
        Y8 : out bit_vector(7 downto 0) );
end entity;

architecture RTL of AND8 is
  component ANDX
    generic (W : positive);
    port( A : in  bit_vector(W-1 downto 0);
          B : in  bit_vector(W-1 downto 0);
          Y : out bit_vector(W-1 downto 0) );
  end component;
begin
  A1: ANDX
        generic map (W => 8)
        port map (A => A8, B => B8, Y => Y8);
end architecture;
```

deklaracja komponentu

osadzenie modułu **ANDX** z parametrem W=8

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Język VHDL

Poziom przesłań międzyrejestrowych RTL

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Współbieżne przypisanie proste

- Składnia:

```
<Odbiorca> <= <Wyrażenie>;
```

- Przykład:

```
Y <= A and B;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Współbieżne przypisanie proste (cd.)


```
signal S : bit_vector(1 to 4);
signal A, B, C, D : bit;
...
S <= ('1', '0', '1', '0');
-- przykłady:
(A, B, C, D) <= S;
-- A=1 B=0 C=1 D=0
-- lub
(A, B, C, D) <= bit_vector'('1', '0', '1', '0');
-- A=1 B=0 C=1 D=0
-- lub
(3=>A, 2=>B, 4=>C, 1=>D) <= S;
-- tj.: (D,B,A,C) <= (1,0,1,0) czyli A=1 B=0 C=0 D=1
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Współbieżne przypisanie warunkowe **when ... else**

- Składnia przypisania:

```
<Odbiorca> <= <Wyrażenie1> when <Warunek1> else
              <Wyrażenie2> when <Warunek2> else
              ...
              <WyrażenieN>;
```

- Przykład:

```
Y <= A when SELECT_A = '1' else
     B when SELECT_B = '1' else
     C;

Y <= A when ENABLE = '1' else 'Z';
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Przypisanie współbieżne **select** ... **when**

- Składnia przypisania **select**:

```
with <Wyrażenie_wyboru> select
     <Odbiorca> <= <Wyrażenie1> when <Wybór1>,
                   <Wyrażenie2> when <Wybór2>,
                   ...
                   <WyrażenieN> when <WybórN>;
```

- Przykład:

```
signal A, B, C, D, Y : bit;
signal SEL: bit_vector(1 downto 0);
...
with SEL select
  Y <= A when "00",
       B when "01",
       C when "10",
       D when others;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Operatory logiczne

- Dostępne operatory:

```
and
or
nand
nor
xor
xnor
not
```

Grupowanie operatorów za pomocą nawiasów:

```
A and B and C and D
```

ale:

```
(A and B) or C
A and (B or C)
```

- Operatory porównania:

```
"101011" < "1011" -- warunek spełniony
"101" < "101000" -- warunek spełniony
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

- Operatory dodawania i konkatenacji:

```
C <= A + B;
signal A : bit_vector(3 downto 0);
signal B, C : bit_vector(1 downto 0);
A <= not B & not C; -- tablica & tablica
```

Inne operatory

- mnożenia *
- dzielenia **/**
- **mod**
- **rem**
- **abs**
- potęgowania **.

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opóźnienia

- Opóźnienie inercyjne
- Opóźnienia typu **transport**

```
Y1 <= inertial  not A after 7 ns;
Y2 <= transport not A after 7 ns;
```

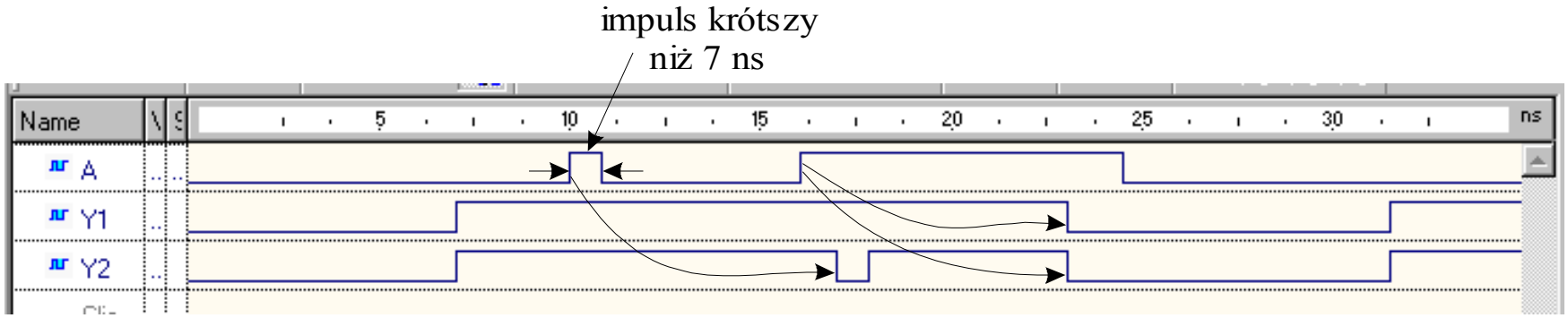
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opóźnienia (cd.)

```
Y1 <= inertial  not A after 7 ns;
Y2 <= transport not A after 7 ns;
```




```
Y1<= reject 3 ns inertial not A after 7 ns;
```

Opóźnienia (cd.)

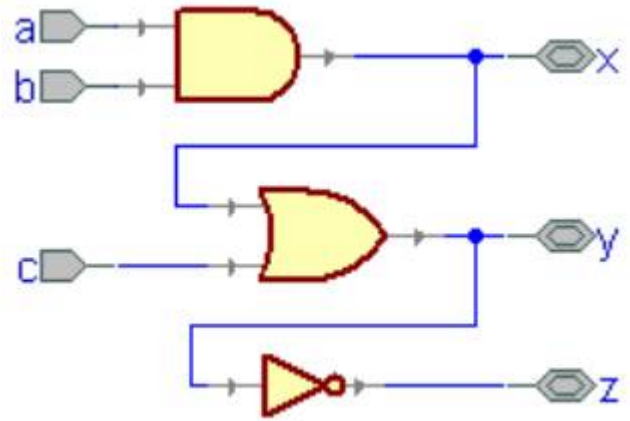

```
Y <= inertial A after 7 ns when SEL="00" else
              B after 11 ns when SEL="01" else
              C when SEL="10" else
              D ;

with SEL select
  Y <= transport A after 4 ns when "00",
                 B after 3 ns when "01",
                 C after 6 ns when "10",
                 D when others;
```

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

- Przykład 1 działania symulatora

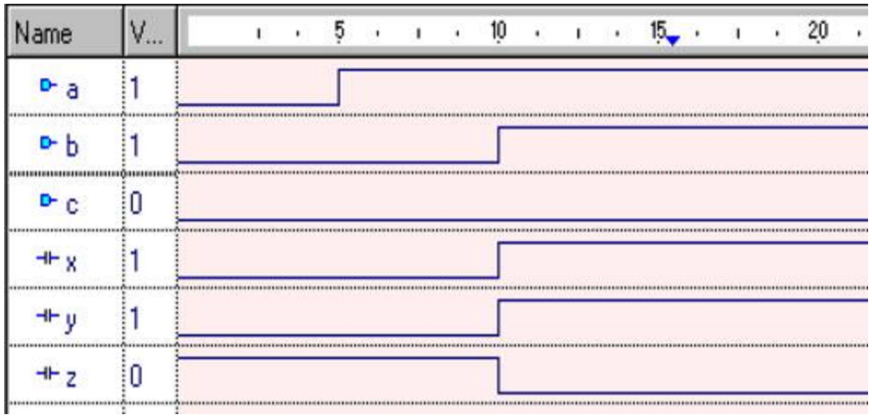
(uproszczony, sygnały typu **bit**)

```
entity test is port (
        a, b, c : in bit;
        x, y, z : buffer bit);
end entity;

architecture beh of test is

begin
   z <= not y;
   y <= c or x;
   x <= a and b;
end architecture;
```

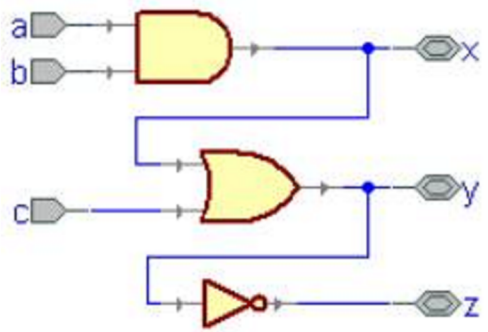


Operacje współbieżne oraz czasowe (cd.)

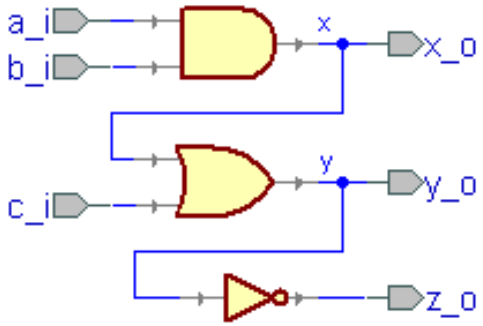
- Przykład 1 działania symulatora (cd.)



Time	Delta	a	b	c	x	y	z
0.000	0	0	0	0	0	0	0
0.000	1	0	0	0	0	0	1
5.000 ns	0	1	0	0	0	0	1
10.000 ns	0	1	1	0	0	0	1
10.000 ns	1	1	1	0	1	0	1
10.000 ns	2	1	1	0	1	1	1
10.000 ns	3	1	1	0	1	1	0



Operacje współbieżne oraz czasowe (cd.)



- Przykład 2 działania symulatora

```
library ieee;
use ieee.std_logic_1164.all;

entity tst is
  port(
    a_i : in std_logic;
    b_i : in std_logic;
    c_i : in std_logic;
    x_o : out std_logic;
    y_o : out std_logic;
    z_o : out std_logic
    );
end entity;
```

```
architecture tst of tst is
  signal x: std_logic;
  signal y: std_logic;
begin
  x <= a_i and b_i;
  y <= c_i or x;
  z_o <= not y;

  x_o <= x;
  y_o <= y;
end architecture;
```

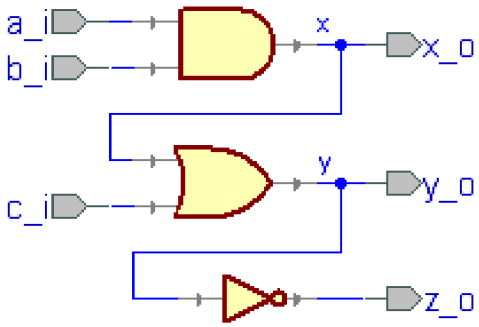
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

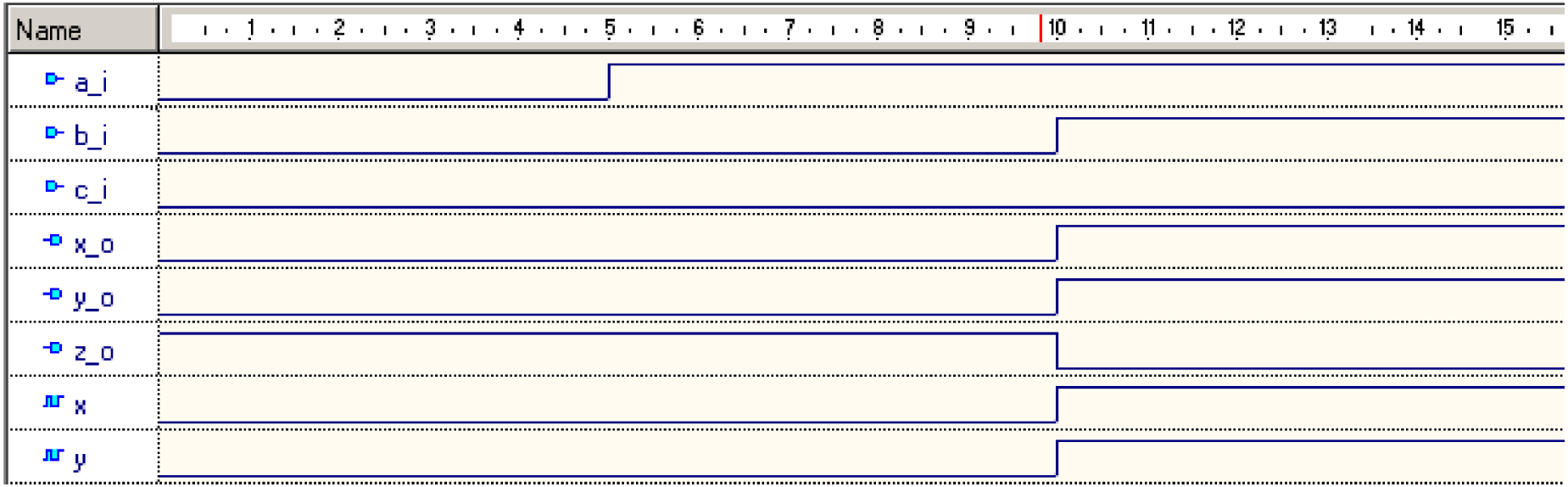
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Operacje współbieżne oraz czasowe (cd.)

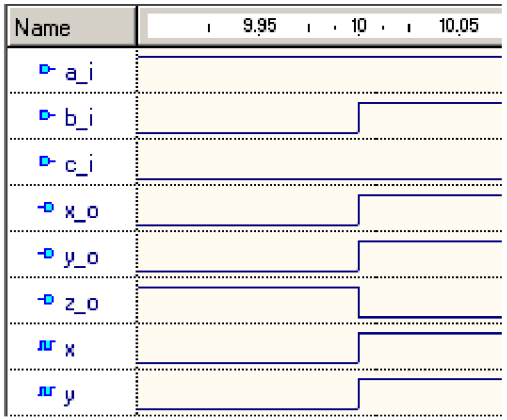
- Przykład 2 działania symulatora (cd.)



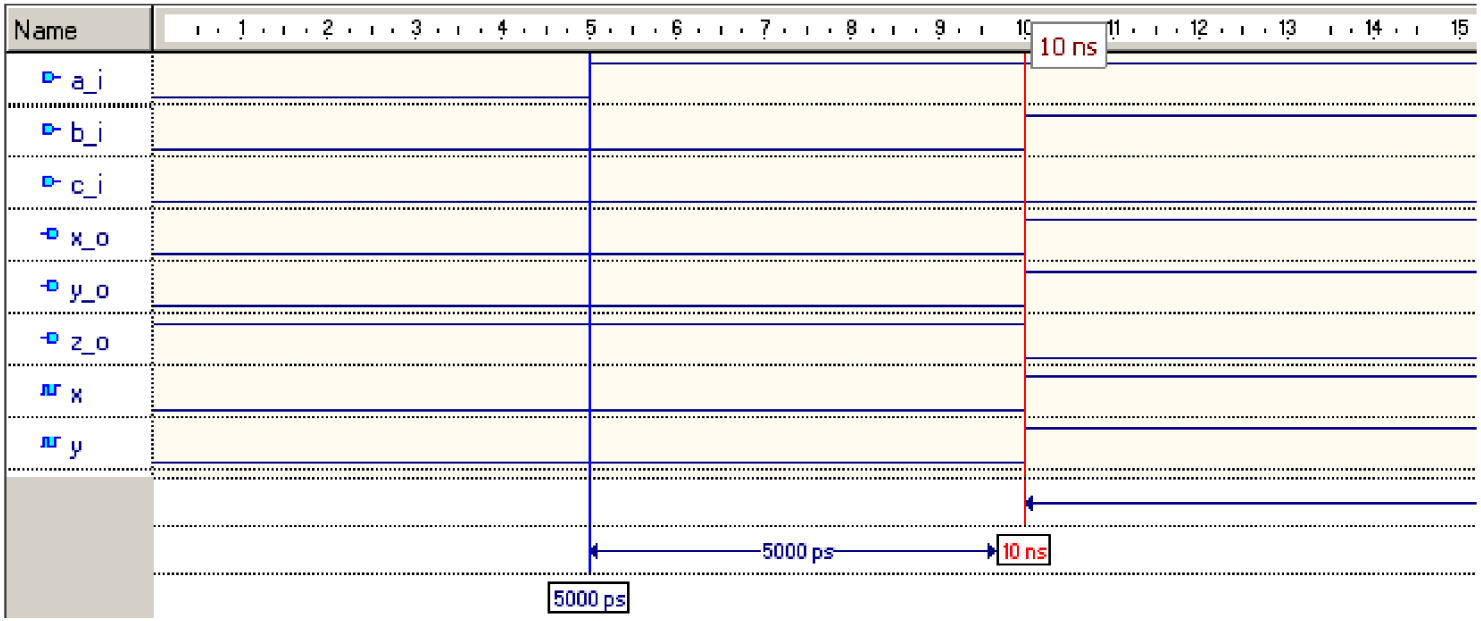


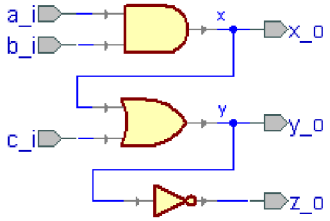
Operacje współbieżne oraz czasowe (cd.)

- Przykład 2 działania symulatora (cd.)



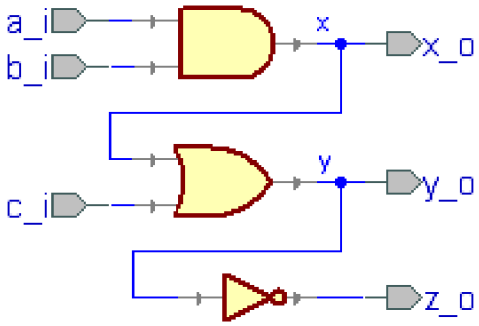
powiększenie





Operacje współbieżne oraz czasowe (cd.)

- Przykład 2 działania symulatora (cd.)



Time	Delta	a_i	b_i	c_i	x	y	x_o	y_o	z_o
0.000	0	0	0	0	U	U	U	U	U
0.000	1	0	0	0	0	U	U	U	U
0.000	2	0	0	0	0	0	0	U	U
0.000	3	0	0	0	0	0	0	0	1
5.000 ns	0	1	0	0	0	0	0	0	1
10.000 ns	0	1	1	0	0	0	0	0	1
10.000 ns	1	1	1	0	1	0	0	0	1
10.000 ns	2	1	1	0	1	1	1	0	1
10.000 ns	3	1	1	0	1	1	1	1	0

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Instrukcje współbieżne i sekwencyjne

- Instrukcje współbieżne:
  - przypisania wartości sygnału (**<=**),
  - kontroli (**assert**),
  - procesu (**process**),
  - procedury,
  - funkcji,
  - blokowa (**block**),
  - wstawiania składnika (**component**),
  - powielania (**generate**).

- Instrukcje sekwencyjne.
  - oczekiwania,
  - przypisania wartości sygnału,
  - przypisania wartości zmiennej,
  - kontroli (**assert**),
  - raportu,
  - wywołania procedury,
  - instrukcje warunkowe (**if**, **case**),
  - instrukcje pętli,
  - instrukcja wyjścia,
  - instrukcja powrotu,
  - instrukcja pusta.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Procesy

- Proces - wykonywanie instrukcji jest sekwencyjne.

```
<nazwa_procesu> : process (<lista_czułości>)
  <część_deklaracyjna>
begin
  <część_sekwencyjna>
end process;
```

- Proces jest traktowany jako jedno polecenie współbieżne.
- Jeśli w architekturze zdefiniowano kilka procesów, wszystkie one wykonują się współbieżnie względem siebie, a także niezależnie od siebie.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Zmienne

- Zmienne

```
variable  a_v, b_v : std_logic;
variable count_v : integer := 1;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Przypisanie sekwencyjne

- Przypisanie sekwencyjne dla sygnału:

```
<sygnał> <= <wyrażenie>;
```

- Przypisanie sekwencyjne dla zmiennej:

```
<zmienna> := <wyrażenie>;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Przypisanie sekwencyjne (cd.)


```
signal S : bit_vector(1 to 4);
signal A, B, C, D : bit;
...
S <= ('1', '0', '1', '0');
-- przykłady:
(A, B, C, D) <= S;
-- A=1 B=0 C=1 D=0
-- lub
(A, B, C, D) <= bit_vector'('1', '0', '1', '0');
-- A=1 B=0 C=1 D=0
-- lub
(3=>A, 2=>B, 4=>C, 1=>D) <= S;
-- tj.: (D,B,A,C) <= (1,0,1,0) czyli A=1 B=0 C=0 D=1
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Różnice pomiędzy sygnałem i zmienną


```
process ...
  ...
  data <= (others => '0');
  ...
  if ... then
    data(6) <= '1';
  end if;
  ...
end process;
```

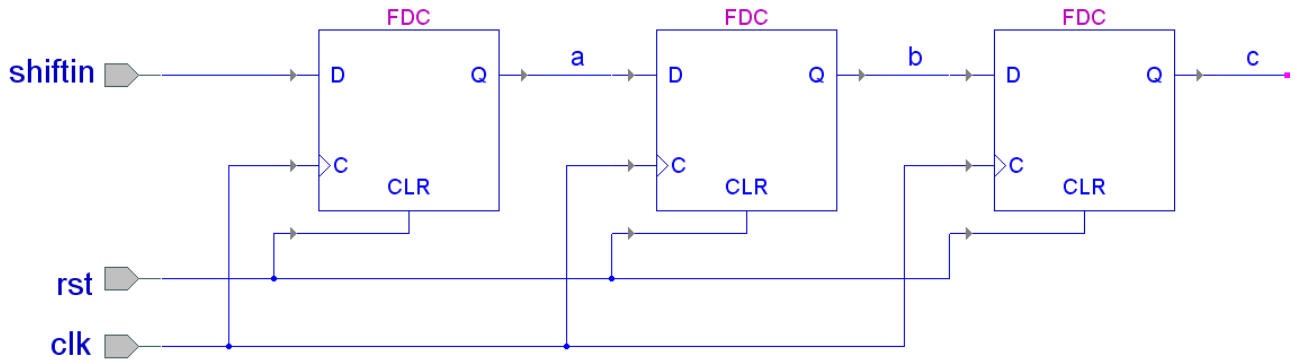
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

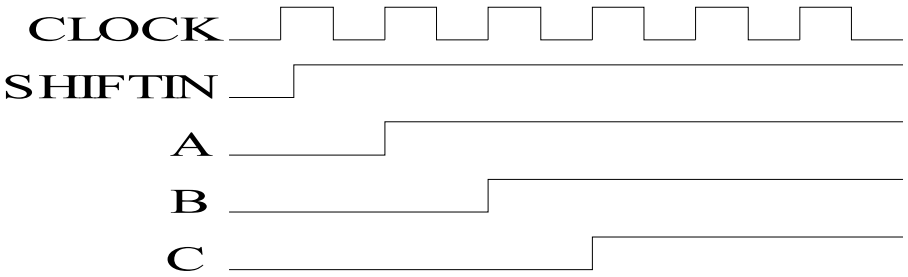
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Różnice pomiędzy sygnałem i zmienną (cd.)

```
SHIFTREG : process(clk,rst)
begin
  if rst='1' then
    a <= '0';
    b <= '0';
    c <= '0';
  elsif rising_edge(clk)
then
    a <= shiftin;
    b <= a;
    c <= b;
  end if;
end process;
```





Różnice pomiędzy sygnałem i zmienną (cd.)

– Następujący kod dałby identyczne rezultaty:

```
...
elsif rising_edge(clk) then
    c <= b;
    b <= a;
    a <= shiftin;
end if;
end process;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Język VHDL

Poziom behawioralny

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Wyrażenie warunkowe **if**


```
-- wersja I:
if <warunek> then
  <akcja>
end if;

-- wersja II:
if <warunek> then
  <akcja1>
else
  <akcja2>
end if;
```

```
-- wersja III:
if <warunek1> then
  <akcja1>
elsif <warunek2> then
  <akcja2>
end if;

-- wersja IV:
if <warunek1> then
  <akcja1>
elsif <warunek2> then
  <akcja2>
else
  <akcja3>
end if;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Wyrażenie warunkowe **case**


```
case SigA is
  when "00" => Output <= DataA;
               State  <= S1;
  when "01" => Output <= DataB;
               State  <= S2;
  when "10" => Output <= DataC;
               State  <= S3;
  when "11" => Output <= DataD;
               State  <= S1;
end case;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia pętli **loop**

- Polecenie **loop**
- Polecenie **while…loop**
- Polecenie **for…loop**
- Polecenie **next**
- Polecenie **exit**

- Polecenie **loop**

```
<etykieta>: loop
  <polecenia sekwencyjne>
end loop;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia pętli **loop** (cd.)

- Polecenie **while...loop**

```
<etykieta>: while <warunek> loop
  <polecenia sekwencyjne>
end loop;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia pętli **loop** (cd.)

- Polecenie **for...loop**

```
<etykieta>: for <identyfikator> in <zakres> loop
  <polecenia sekwencyjne>
end loop;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia pętli **loop** (cd.)

- Przypisanie dla wszystkich elementów tablicy:

```
for I in A'range loop
  A(I) <= not B(I);
end loop;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia pętli **loop** (cd.)

- Przykład typowego błędu:

```
entity test is port (
        a8_i : in std_logic_vector(7 downto 0);
        x_o  : out std_logic_vector);
end entity;

architecture beh of test is
   signal x : std_logic;
begin
   x_o <= x;
   process (a8_i)
   begin
        x <= '1';
        for i in 7 downto 0 loop
                x <= a8_i(i) and x;
        end loop;
   end process;
end architecture;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia pętli **loop** (cd.)

- Rozwiązanie problemu:

```
architecture beh of test is
   signal x : std_logic;
begin
   x_o <= x;
   process (a8)
   variable tmp_v: std_logic;
   begin
        tmp_v := '1';
        for i in 7 downto 0 loop
               tmp_v := a8_i(i) and tmp_v;
        end loop;
        x <= tmp_v;
   end process;
end architecture;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia pętli **loop** (cd.)

- Polecenie **next**

```
next <etykieta> when <warunek>;

next;

pętla_1: for I in 1 to 8 loop
  k := 0;
  pętla_2: loop
    B(k) := '0';
    next pętla_1 when warunek_1;
    k := k + 1;
  end loop pętla_2;
end loop pętla_1;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Polecenia pętli **loop** (cd.)

- Polecenie **exit**

```
exit <etykieta> when <warunek>;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Podstawowe rodzaje procesów

- Procesy synchroniczne
- Procesy kombinacyjne
- Procesy zatrzaskiwane

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Podstawowe rodzaje procesów (cd.)

- Procesy synchroniczne

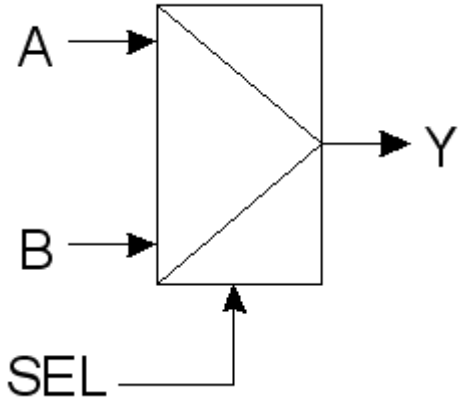
```
process (clk_i)
begin
  if rising_edge(clk_i) then
    reg8 <= data8;
  end if;
end process;

process (clk_i, rst_i)
begin
  if rst_i='1' then
    reg8 <= "00000000"; --wyczyść rejestr reg8
  elsif rising_edge(clk_i) then
    reg8 <= data8;
  end if;
end process;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

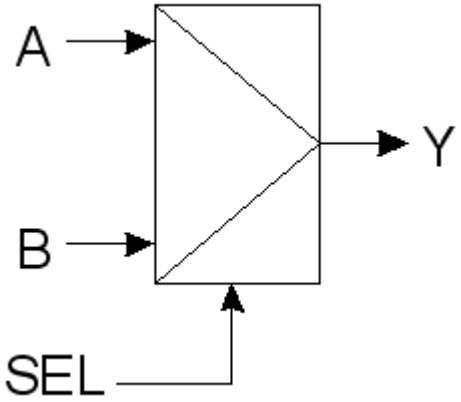
- Procesy kombinacyjne
  - Logika kombinacyjna może zostać opisana:
    - poza procesem,
    - poprzez proces kombinacyjny.



Podstawowe rodzaje procesów (cd.)

- Procesy kombinacyjne (cd.)

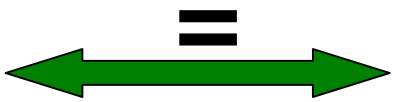
```
process(a, b, sel) -- multiplekser 2 na 1
begin
  if sel='0' then
    y<=a;
  else
    y<=b;
  end if;
end process;
```



Podstawowe rodzaje procesów (cd.)

- Procesy kombinacyjne (cd.)
  - Należy zauważyć, że współbieżne przypisanie ciągłe, np.:

```
architecture beh of test is
begin

  y <= a and b;

end architecture;
```



  - jest równoważne następującemu procesowi kombinacyjnemu:

```
architecture beh of test is
begin
  process (a, b)
  begin
    y <= a and b;
  end process;
end architecture;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Podstawowe rodzaje procesów (cd.)

- Procesy zatrzaskiwane

```
process(enable, rst_i, data8)
begin
   if rst_i='1' then --zerowanie zatrzasku
        latch8 <= "00000000";
   elsif enable='1' then --zatrzask aktywny
        latch8=data8;
   end if;
end process;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opóźnienia typu **wait**

- Polecenie **wait for**
- Polecenie **wait until**
- Polecenie **wait on**
- Polecenie **wait**

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opóźnienia typu **wait** (cd.)

- Polecenie **wait for**

```
wait for <czas>;

wait for 10 ns;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opóźnienia typu **wait** (cd.)

- Polecenie **wait until**

```
wait until <warunek_sygnału>;

wait until ENABLE='1';
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

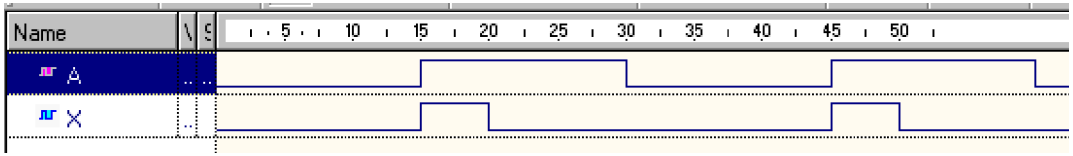
Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opóźnienia typu **wait** (cd.)

- Polecenie **wait until** (cd.)

```
architecture RTL of TEST is
signal A, X: bit;
begin
   process
   begin
         X <= '0';
         wait until A='1';
         X <= '1';
         wait for 5 ns;
   end process;
end architecture;
```



UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Opóźnienia typu **wait** (cd.)

- Polecenie **wait on**

```
wait on <lista_czułości>;

wait on S1, S2;
```

- Polecenie **wait**

```
wait;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Język VHDL

Funkcje i procedury

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Funkcje

```
function rising_edge(signal CLK: std_logic) return boolean is
--
-- część deklaracyjna
--
begin
  --
  -- ciało funkcji
  --
  return (VALUE);
end function rising_edge;
```

- Przykład wywołania funkcji

```
rising_edge(ENABLE);
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Funkcje (cd.)

- Funkcje i procedury mogą być zagnieżdżone.
- Wyrażenia **wait** nie są dopuszczalne w ciele funkcji (w procedurach są!) i dlatego funkcje są wykonywane w zerowym czasie.
- Z tego powodu wyrażenie **wait** nie może znajdować się również w procedurze wywoływanej przez funkcję.
- Funkcje mogą być wywoływane rekurencyjnie.
- Można definiować funkcje o nazwach zarezerwowanych dla operatorów lub innych funkcji – nazywa się to przeciążaniem operatora lub funkcji.

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Przykład funkcji


```
function      to_bitvector(IN_VAL:std_logic_vector)      return
   bit_vector is
variable RET_VAL:bit_vector(IN_VAL'range);
begin
   for I in IN_VAL'range loop
       case IN_VAL(I) is
             when '0' => RET_VAL(i) := '0';
             when '1' => RET_VAL(i) := '1';
             when others => RET_VAL(i) := '0';
       end case;
   end loop;
   return (RET_VAL);
end function to_bitvector;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Procedury

- Interfejs procedury:

```
procedure READ_V1D (variable FNAME: in text; V: out
   std_logic_vector);
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Procedury (cd.)


```
procedure QDR_read (
        read_addr: in std_logic_vector(7 downto 0);
        signal read_phase  : in std_logic;
        signal write_phase : in std_logic;
        signal QDR_rps_n_i : out std_logic;
        signal QDR_a_read: out std_logic_vector(23 downto 0)) is
begin
        QDR_rps_n_i<='Z';
        QDR_a_read  <= (others => 'Z');
        wait until read_phase='1';
        QDR_rps_n_i<='0';
        QDR_a_read  <= (others => '0');
        QDR_a_read(7 downto 0) <=  read_addr;
        wait until read_phase='0';
        QDR_rps_n_i <= 'Z';
        QDR_a_read  <= (others => 'Z');
end procedure;
```

UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI

Procedury (cd.)


```
process
...
begin
   ...
   for i in 0 to 3 loop
     QDR_read (address_v, read_phase, write_phase,
               QDR_rps_n_i, QDR_a_read);
     address_v := address_v + 1;
     random_delay(seed_v);
   end loop;
   ...
end process;
```

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*, współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego. Nr umowy UDA – POKL.04.01.01-00-236/08

KAPITAŁ LUDZKI