Laboratorij za načrtovanje integriranih vezij

Univerza *v Ljubljani*
Fakulteta *za elektrotehniko*

Digitalni Elektronski Sistemi

Osnove jezika VHDL

2. del: proces

Procesno okolje

- Znotraj procesnega okolja so sekvenčni stavki
  - vrstni red stavkov je pomemben
  - med sekvenčne stavke spada pogojni (if) stavek
  - signalu lahko priredimo vrednost na vač mestih, dejansko se izvrši le zadnja prireditev

```
primerjava: process (a, b)
begin
   enako <= '0';
   if a=b then
      enako <= '1';
   end if;
end process;
```

Zgradba procesnega okolja

```
oznaka: process (seznam signalov)
begin
.....
end process;
```

- Z oznako poimenujemo del vezja, ki ga predstavlja proces
- Simulacija procesa se izvrši ob spremembi enega izmed signalov iz seznama
  - pri opisu kombinacijskih vezij moramo navesti vse signale, ki predstavljajo vhode v proces

Pogojni stavek (if)

- Ob izpolnjenem pogoju se izvrši en ali več stavkov, ki so zapisani za "then"
- Opcija: če pogoj ni izpolnjen se izvršijo stavki za "else"
- Pogojni stavek zaključimo z "end if "

```
if pogoj then
   stavki(1);
else
   stavki(2);
end if;
```

```
if pogoj1 then
   stavki(1);
elsif pogoj2 then
   stavki(2);
else
   stavki(3);
end if;
```

Primerjava s sočasnimi stavki

- Pogojni stavek spada med sekvenčne stavke
  - uporabljamo ga lahko le znotraj procesa
- Pogojni prireditveni stavek je alterativa med sočasnimi stavki

```
p_max: process(a, b)
begin
  if a>b then
     max <= a;
  else
     max <= b;
  end if;
end process;
```

=

```
max <= a when a>b else b;
```

Primerjava (2)

```
bin2bcd: process(bin)
begin
  if bin>9 then
     enice <= bin – “1010”;
     desetice <= ‘1’;
  else
     enice <= bin;
     desetice <= ‘0’;
  end if;
end process;
```

=

```
enice <= bin - ”1010”  when bin>9
                 else bin;

desetice <= ‘1’ when bin>9
                      else ‘0’;
```

- V pogojnem stavku imamo lahko več prireditev

Zaporedje pogojev: prioriteta

```
p: process(int0, int1, int2)
begin
  if int0='1' then
     v <= "01"
  elsif int1='1' then
     v <= "10";
  elsif int2='1' then
     v <= "11";
  else
     v <= "00";
  end if;
end process;
```

=

```
v <="01"  when int0='1' else
      "10"  when int1='1' else
      "11"  when int2='1' else
      "00";
```

int0 ima prednost pred
int1, ki ima prednost pred
int2 ...

- Zaporedni pogoji se ovrednotijo s prioriteto

Sekvenčni izbirni stavek (case)

- Odločamo se glede na vrednost enega signala
- Izbirni stavek nima prioritete

```
case ime_signala is
    when vrednost1 =>
       stavki(1);
    when vrednost2 =>
       stavki(2);
    ...
    when others  =>
       stavki(n);
  end case;
```

```
decod: process(digit)
begin
  case digit is
    when “00” =>
       display <= “00111111”;
    when “01” =>
       display <= “00000110”;
    when others  =>
       display <= “11111001”;
  end case;
end process;
```

Primer: izbiralnik (multipleksor)

- Izbiralnik s štirimi vhodi (a, b, c in d)

```
m: process(mode,a,b,c,d)
begin
  if mode="00" then
     mux <= a;
  elsif mode="01" then
     mux <= b;
  elsif mode="10" then
     mux <= c;
  else
     mux <= d;
  end if;
end process;
```

```
m: process(mode,a,b,c,d)
begin
  case mode is
    when "00" =>
       mux <= a;
    when "01" =>
       mux <= b;
    when "10" =>
       mux <= c;
    when others  =>
       mux <= d;
  end case;
end process;
```

Sočasni izbirni stavek (with…select)

- Na podlagi izbire priredimo signalu različne vrednosti (ali izraze)

```
with izbira select
signal <= izraz(1) when vrednost1,
          izraz(2) when vrednost1,
          ...
          izraz(n) when others;
```

- Za razliko od case stavka lahko prirejamo vrednost le enemu signalu

```
with digit select
display <= “00111111” when “00”,
           “00000110”  when “01”,
           “11111001”  when others;
```

Sekvenčni stavki in sinteza vezja

- Prirejanje vrednosti signalom na več mestih
- Če signalu pod kakšnim pogojem ne določimo vrednosti, se bo ohranjala zadnja vrednost
  - dobimo sekvenčno vezje !

```
p: process(set_flag, clear_flag)
begin
  if set_flag='1' then
     flag <= '1';
  elsif clear_flag='1' then
     flag <= '0';
  end if;
end process;
```

Naredili smo asinhroni zapah za signal flag !

Opis kombinacijskih vezij

- Definiramo vrednosti pri vseh pogojih:
  - vrednost priredimo pri vsakem "if" in "else" ali
  - vrednost priredimo na začetku procesa in jo spremenimo v "if" stavkih

```
primerjava: process (a, b)
begin
   if a=b then
      enako <= '1';
   else
      enako <= '0';
   end if;
end process;
```

```
primerjava: process (a, b)
begin
   enako <= '0';
   if a=b then
      enako <= '1';
   end if;
end process;
```