Voorbeeld: Een hotelschakelaar

In deze sectie zal een voorbeeld worden gegeven van een inwerkopdracht. We zullen voor het gegeven probleem een toestandsdiagram opstellen, vervolgens de hieruit afgeleide VHDL gedragsbeschrijving invoeren, daarna deze simuleren en synthetiseren, en als laatste de layout maken. Voordat dit voorbeeld wordt doorlopen verdient het aanbeveling de VHDL appendix en de Linux appendix van de Studentenhandleiding Ontwerppracticum te lezen.

Gevraagd: Ontwerp een schakeling die een lamp aanstuurt met 3 toetsen(s1, s2, s3). Wanneer 1 of meer toetsen worden ingedrukt, gaat de lamp aan of uit, afhankelijk van de huidige toestand. Een 'overrule' toets(ov) zorgt dat de lamp in zijn huidige toestand blijft. Maak daartoe de schaking voor de black-box van figuur 1.

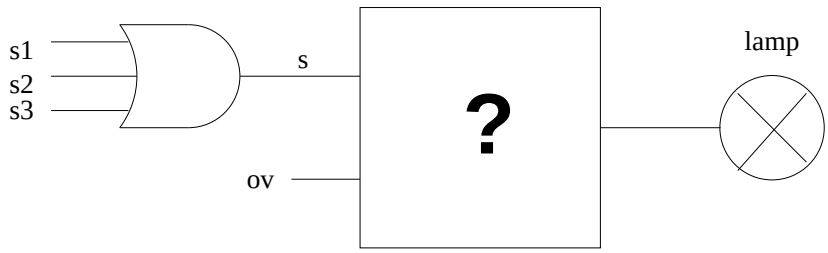

Figure 1: Ontwerpopdracht voor een hotelschakelaar

Uitwerking:
Het toestandsdiagram van figuur 2 kan voor de schakeling worden opgesteld.

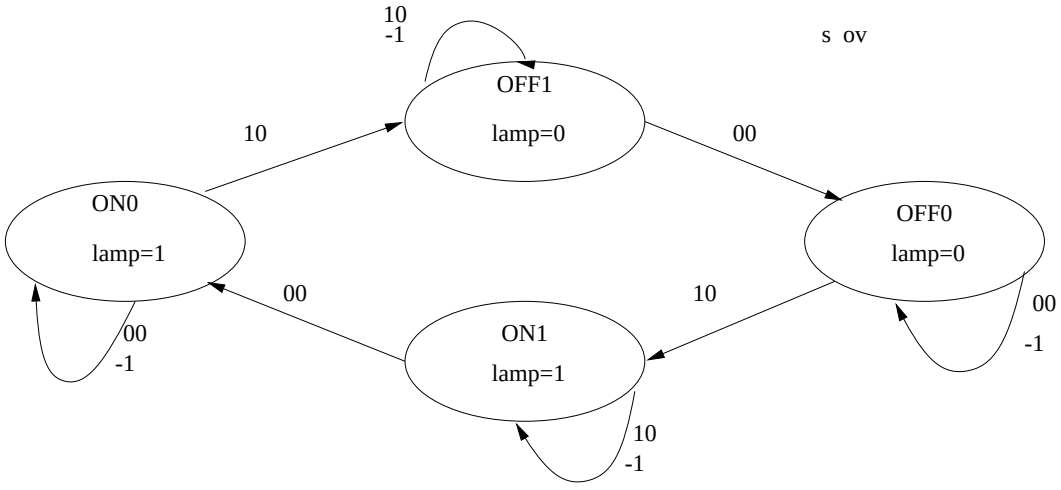

Figure 2: Toestandsdiagram voor de hotelschakelaar

Zorg eerst dat alle paden naar te gebruiken programmatuur zijn gezet door eenmalig het commando op_init uit te voeren in een terminal window (zie de Linux appendix).

Start nu het programma *GoWithTheFlow* op door op het desbetreffende icon op de DeskTop te

klikken. Dit zal de interface zoals in figuur 3 te zien geven.

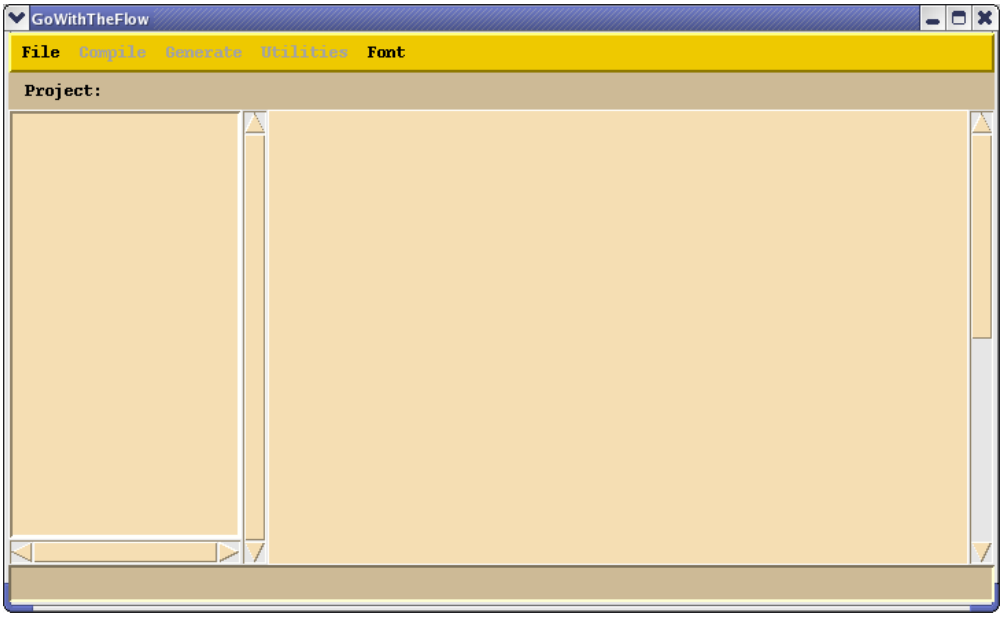

Figure 3: Het programma *GoWithTheFlow* na opstarten

Maak dan een project directory aan m.b.v. het commando File → New project. Een window zoals in figuur 4 zal verschijnen. Specificeer in het vak achter Selection de naam van een nog niet bestaande directory, en klik op OK.

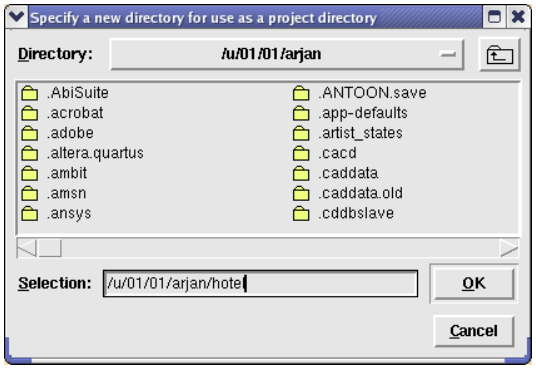

Figure 4: Het aanmaken van een project directory

Voer dan allereerst de entity beschrijving voor hotel in door op File → New entity te klikken en het daarop verschijnende widget in te vullen zoals in figuur 5. Gebruik de tab toets om naar de volgende kolom te springen.

Na op OK geklikt te hebben verschijnt de bijbehorende VHDL beschrijving in een new window, zie figuur 6. Klik op Compile. Na eerst bevestigend te hebben geantwoord op de vraag of de beschrijving moet worden weggeschreven zal deze worden opgeborgen in de database van *GoWithTheFlow* en vervolgens worden gecompileerd.

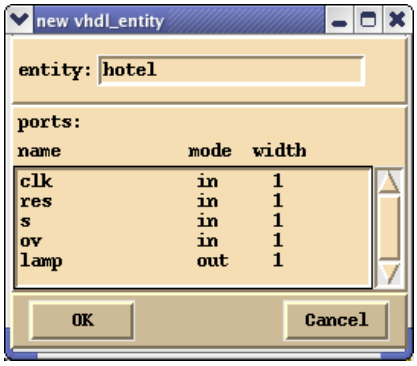

Figure 5: Het invoeren van een nieuwe entity

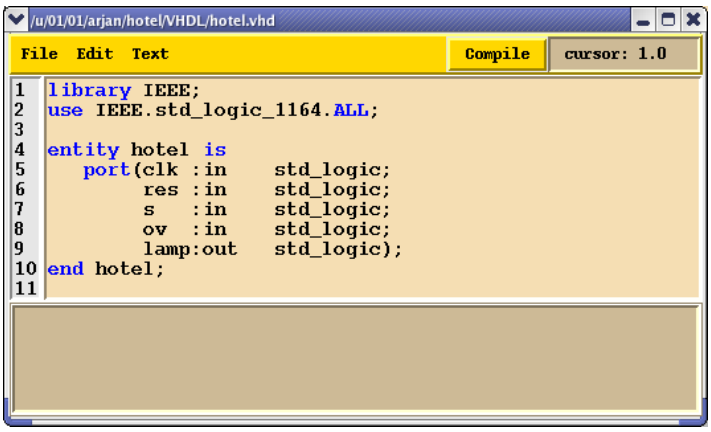

Figure 6: De VHDL code voor de hotel entity

Klik vervolgens op de rechthoek ”hotel” en selecteer Add Architecture om de volgende gedragsbeschrijving in te voeren. Merk op dat (conform wat in VHDL appendix is beschreven) het eerste process statement (lbl1) een beschrijving bevat van de toestandsregisters, en het tweede process statement (lbl2) een beschrijving van de combinatorische logica die aan de hand van de huidige toestand en de ingangssignalen de volgende toestand en de uitganssignalen berekent.

```
library IEEE;
use IEEE.std_logic_1164.ALL;

architecture behaviour of hotel is
   type lamp_state is (OFF0, OFF1, ON0, ON1);
   signal state, new_state: lamp_state;
begin
   lbl1: process (clk)
   begin
      if (clk'event and clk = '1') then
         if res = '1' then
            state <= OFF0;
         else
```

```
            state <= new_state;
         end if;
      end if;
   end process;
   lbl2: process(state, s, ov)
   begin
      case state is
         when OFF0 =>
            lamp <= '0';
            if (s = '1') and (ov = '0') then
                new_state <= ON1;
            else
                new_state <= OFF0;
            end if;
         when ON1 =>
            lamp <= '1';
            if (s = '0') and (ov = '0') then
                new_state <= ON0;
            else
                new_state <= ON1;
            end if;
         when ON0 =>
            lamp <= '1';
            if (s = '1') and (ov = '0') then
                new_state <= OFF1;
            else
                new_state <= ON0;
            end if;
         when OFF1 =>
            lamp <= '0';
            if (s = '0') and (ov = '0') then
                new_state <= OFF0;
            else
                new_state <= OFF1;
            end if;
      end case;
   end process;
end behaviour;
```

Klik op Compile. Het programma zal vragen om de beschrijving eerst weg te schrijven. Antwoord bevestigend, en wanneer er geen fouten in de beschrijving zitten zal vervolgens een rechthoek "behaviour" onder het blokje hotel verschijnen ten teken dat de beschrijving is opgeborgen in de database van *GoWithTheFlow*. Wanneer er wel compilatie fouten zijn, zullen deze fouten getoond worden en zullen deze eerst verbeterd moeten worden.

Ga vervolgens op de rechthoek behaviour staan, klik met de linker muisknop en selecteer Add Configuration. Er zal nu een window verschijnen met een naam voor de VHDL configuratie file. Klik

op OK, de VHDL beschrijving voor de configuratie file wordt getoond, schrijf die vervolgens weg en compileer die.

Maak nu op dezelfde wijze ook een entity beschrijving, een gedragsbeschrijving en een configuratie file aan voor een testbench voor de hotel schakeling. Noem de entity "hotel_tb" (het is niet nodig om terminals te specificeren voor deze testbench entity) en maak een behaviour beschrijving als volgt:

```
library IEEE;
use IEEE.std_logic_1164.ALL;

architecture behaviour of hotel_tb is
   component hotel
      port (clk  : in  std_logic;
            res  : in  std_logic;
            s    : in  std_logic;
            ov   : in  std_logic;
            lamp : out std_logic);
   end component;
   signal clk: std_logic;
   signal res: std_logic;
   signal s: std_logic;
   signal ov: std_logic;
   signal lamp: std_logic;
begin
   lbl1: hotel port map (clk, res, s, ov, lamp);
   clk <= '1' after 0 ns,
          '0' after 100 ns when clk /= '0' else '1' after 100 ns;
   res <= '1' after 0 ns,
          '0' after 200 ns;
   s <= '0' after 0 ns,
        '1' after 600 ns,
        '0' after 1000 ns,
        '1' after 1400 ns,
        '0' after 1800 ns,
        '1' after 2200 ns,
        '0' after 2600 ns,
        '1' after 3000 ns,
        '0' after 3400 ns,
        '1' after 3800 ns;
   ov <= '0' after 0 ns,
         '1' after 1800 ns,
         '0' after 2600 ns;
end behaviour;
```

Het main window van *GoWithTheFlow* zou nu een beeld moeten geven zoals in figuur 7.

Start nu een VHDL simulatie door met de linker muisknop op hotel_tb_behaviour_cfg te klikken en

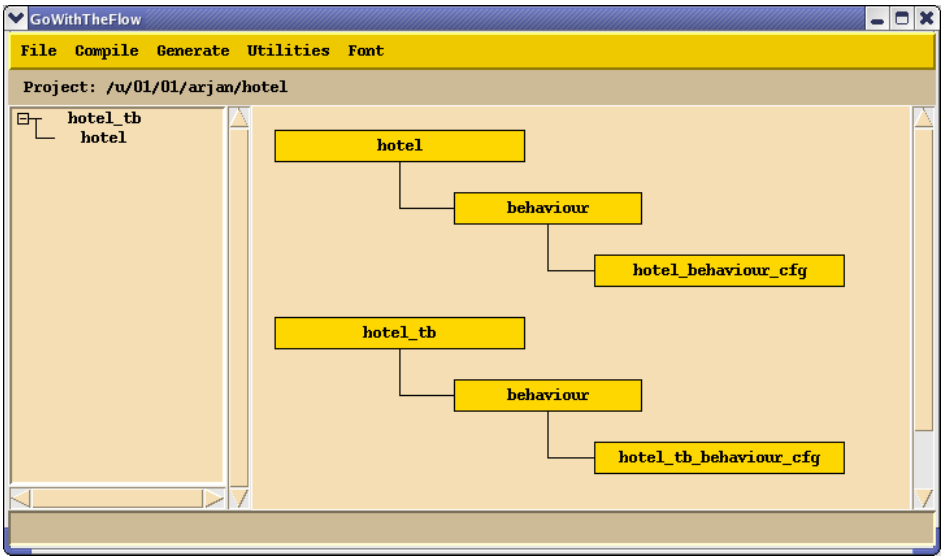

Figure 7: *GoWithTheFlow* na toevoegen behaviour and testbench

simulate te selecteren. Na een simulatie over bijvoorbeeld 4000 ns zal het wave window van ModelSim een resultaat geven zoals in figuur 8. Om zo dadelijk ook een switch-level simulatie (een

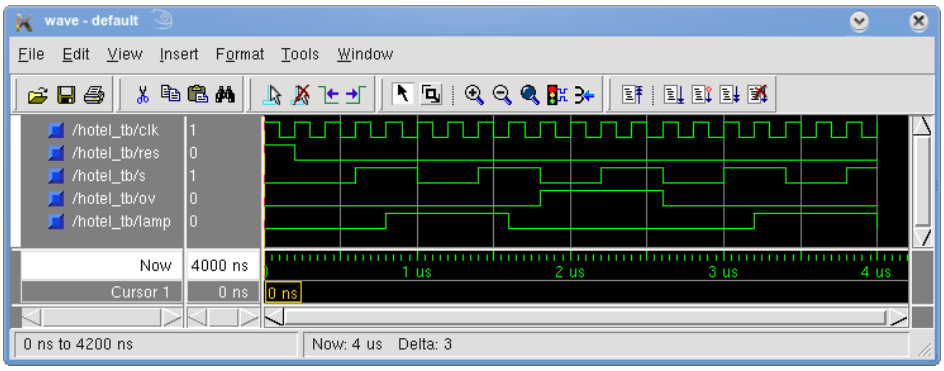

Figure 8: Resultaat van een of VHDL simulatie met ModelSim

simulatie op transistor niveau) voor de gemaakte layout te kunnen verrichten zullen we nu een .lst file aanmaken waarin informatie over de in en uitgangssignalen van de simulatie staat. Selecteer daartoe in het Tools menu van het wave window het commando Make_list_file. Selecteer vervolgens de hotel behaviour beschrijving en klik op OK. Schrijf vervolgens de file hotel.lst weg.

Vervolgens zullen we een synthese stap doen voor de behaviour beschrijving van de hotel entity. Daarbij wordt de gedragsbeschrijving omgezet in een netwerkbeschrijving bestaande uit cellen uit de Sea-of-Gates celbibliotheek. Klik daartoe met de linker muisknop op de configuratie rechthoek van hotel en selecteer synthesize. Het synthesizer window zal nu verschijnen. Klik makeSyntScript en daarna Synthesize. Eventueel kan op Show circuit worden geklikt om een schema van het gesynthetiseerde circuit te zien. Klik vervolgens op Compile om de VHDL beschrijving te compileren en Parse_sls om een SLS circuit beschrijving in the backend (OCEAN/NELSIS) library te plaatsen. Er zullen nu rechthoeken voor de gesynthetiseerde VHDL beschrijving en de gesynthetiseerde circuit

beschrijving zichtbaar worden in *GoWithTheFlow*.

De gesynthetiseerde VHDL beschrijving kan nu ook gesimuleerd worden met de eerder aangemaakte testbench. Klik daarvoor op de behaviour beschrijving van de testbench en voeg een nieuwe configuratie toe. Geef de nieuwe configuratie een naam die verwijst naar de synthese, bijvoorbeeld hotel_tb_behaviour_syn_cfg. Er zal nu gevraagd worden welke versie van hotel gekozen moet worden voor de nieuwe configuratie van hotel_tb. Selecteer de gesynthetiseerde versie en klik OK. Voor de nieuwe configuratie kan nu een simulatie worden opgestart.

Vanuit de gesynthetiseerde circuit beschrijving kan verder een layout gemaakt worden. Klik daartoe op de rechthoek circuit en selecteer Place & route. Het programma *seadali* zal nu gestart worden. Ga naar het menu automatic tools en voer achtereenvolgens de stappen plaatsen (m.b.v. *madonna*) en bedraden (m.b.v. *trout*) uit. Na afloop zal een layout zoals in figuur 9 te zien zijn. Schrijf vervolgens de layout weg via het database menu (ga terug naar het hoofdmenu via -return-) en verlaat de layout editor.

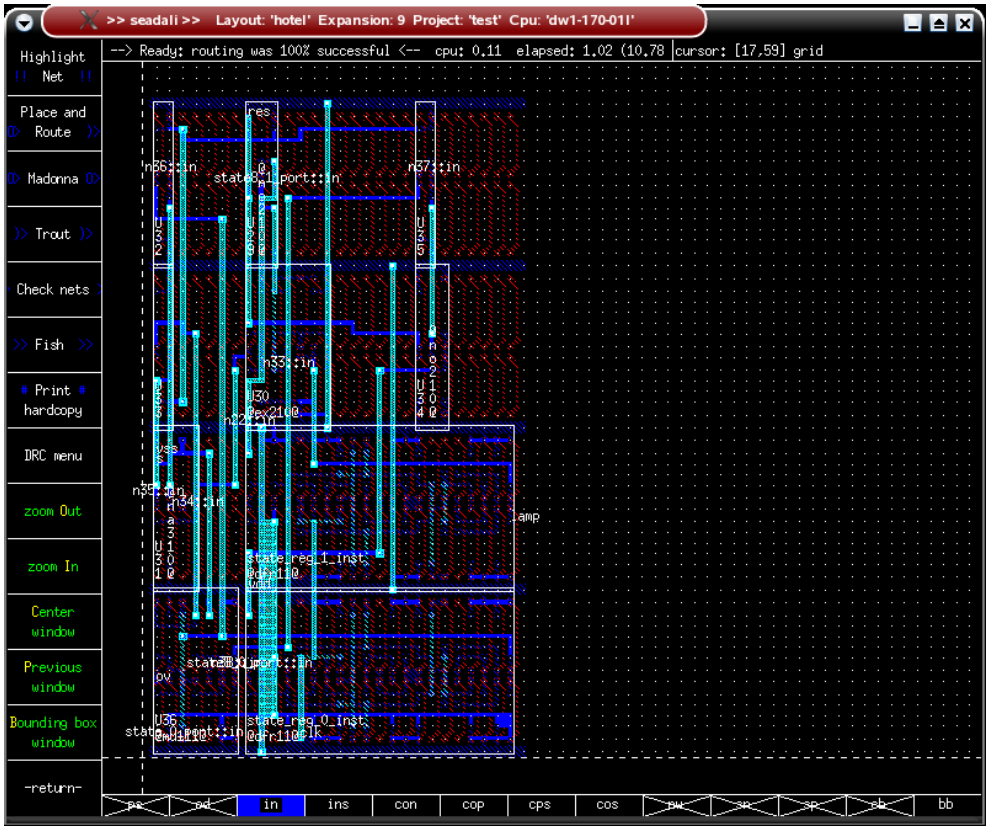

Figure 9: Een layout voor de hotel schakeling

Een alternatief voor Place & route m.b.v. *seadali* bestaat uit het eerst plaatsen van de componenten m.b.v. de *row placer*. Selecteer daartoe op de rechthoek circuit de optie Run row placer. M.b.v. deze placer kan over het algemeen een betere (compactere) plaatsing verkregen worden dan m.b.v. *seadali*. Deze placer kan echter alleen overweg met componenten uit de cellen bibliotheek (en dus niet met hierarchische ontwerpen waarbij eerder door u ontworpen layouts als component worden gebruikt) Verder zal de bedrading altijd nog met *trout* in *seadali* moeten worden gedaan.

De layout kan nu op 2 manieren gecontroleerd worden. De eerste manier bestaat uit het extraheren van een VHDL beschrijving uit de layout en deze met ModelSim te simuleren. Klik daartoe met de linker muisknop op de layout rechthoek en selecteer Extract vhdl. In het nieuwe window, klik Get, vervolgens Write en daarna Compile. Voeg vervolgens een configuratie toe op de manier zoals al eerder gedaan. Maak nu voor de behaviour beschrijving van de testbench voor hotel opnieuw een nieuwe configuratie waarbij de geextraheerde versie van hotel geselecteerd wordt en op deze manier kan de geextraheerde VHDL beschrijving gesimuleerd worden. Het main window van *GoWithTheFlow* zal nu een beeld zoals in figuur 10.

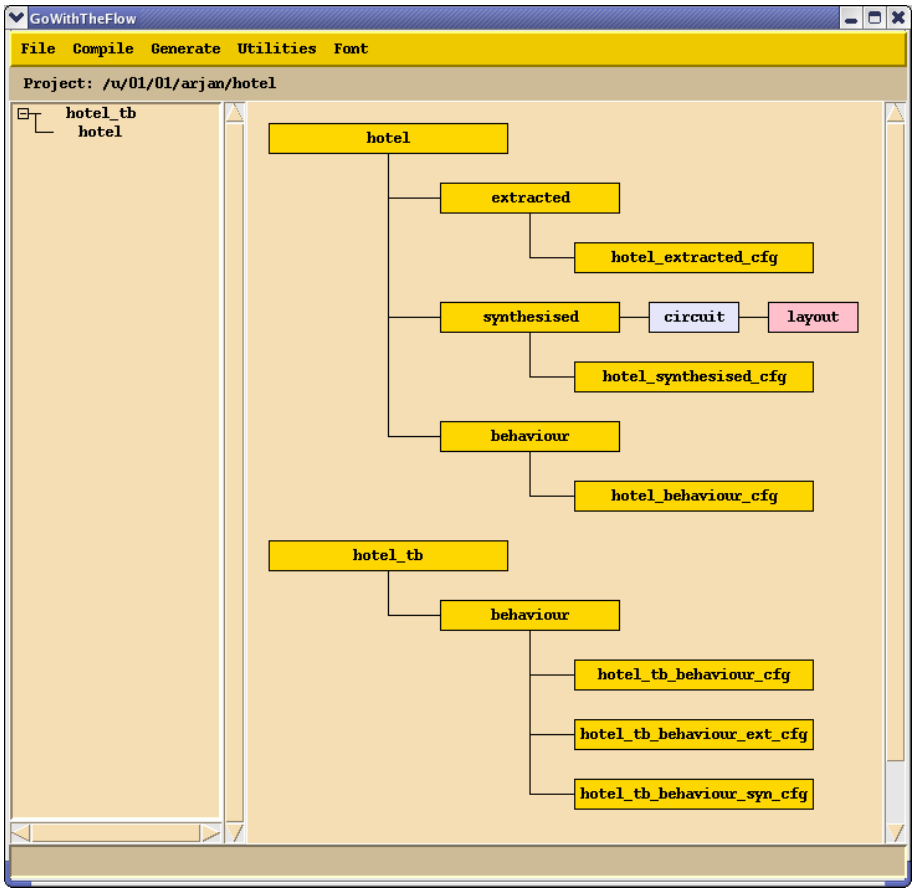

Figure 10: Het main-window van het programma *GoWithTheFlow* na alle stappen voor het hotel voorbeeld te hebben doorlopen

Een nog wat nauwkeurige manier van verificatie bestaat uit het simuleren van een uit de layout geextraheerde transistor beschrijving. Daartoe moet eerst een geschikte commando file voor de switch-level simulator gemaakt worden. Klik op Generate → Command file en klik vervolgens File → Generate from en selecteer de eerder gemaakte file hotel.lst. Schrijf de gegenereerde commando file weg met File → Write als hotel.cmd. Om te simuleren, klik op de layout rechthoek en selecteer Simulate. Na op Doit geklikt te hebben zal allereerst een extractie plaatsvinden en vervolgens een switch-level simulatie. Klik na afloop op ShowResult om een resultaat te zien zoals in figuur 11. Na inzoomen kunnen hier ook de vertragingstijden voor het uitgangssignaal lamp worden waargenomen.

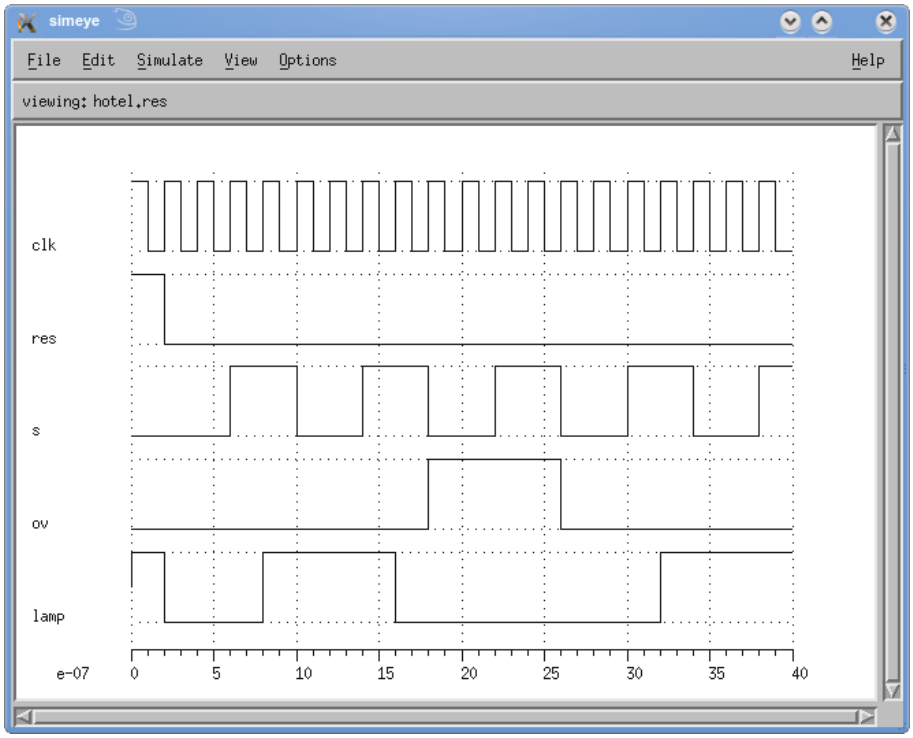

Figure 11: Switch-level simulatie resultaat

Hoewel het in dit geval duidelijk te zien is dat de resultaten van de simulatie op transistor niveau overeenkomen met de resultaten van de oorspronkelijke VHDL simulatie, kunnen we beide simulatie resultaten ook nog door het programma *GoWithTheFlow* laten vergelijken. We moeten daartoe eerst een simulatie referentie file (.ref) file aanmaken vanuit de eerder aangemaakte .lst file. Klik daarvoor op in *GoWithTheFlow* op Generate → Reference file. In het nieuwe window dat verschijnt klik op File → Generate from en selecteer hotel.lst. Klik vervolgens op File → Write om hotel.ref weg te schrijven en sluit het window af. Klik daarna op de knop Utilities → Compare in *GoWithTheFlow* om het compare window te laten verschijnen. Gebruik File → Read ref om hotel.ref in te lezen en File → Compare res om hotel.res in te lezen en deze te vergelijken met hotel.ref. De uitgangssignalen zullen hierbij vergeleken worden op tijdstippen vlak voor de volgende opgaande klokflank. Als het goed is zullen er alleen afwijkende uitgangssignalen optreden tijdens de initialisatie periode aan het begin van de simulatie (zie figuur 12)

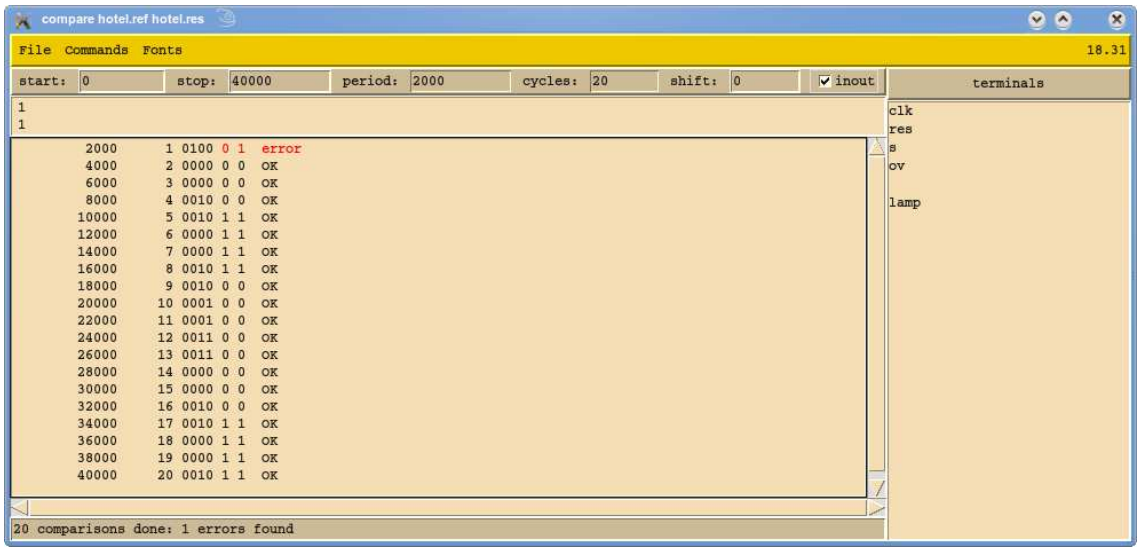

Figure 12: Het vergelijken van simulatie resultaten