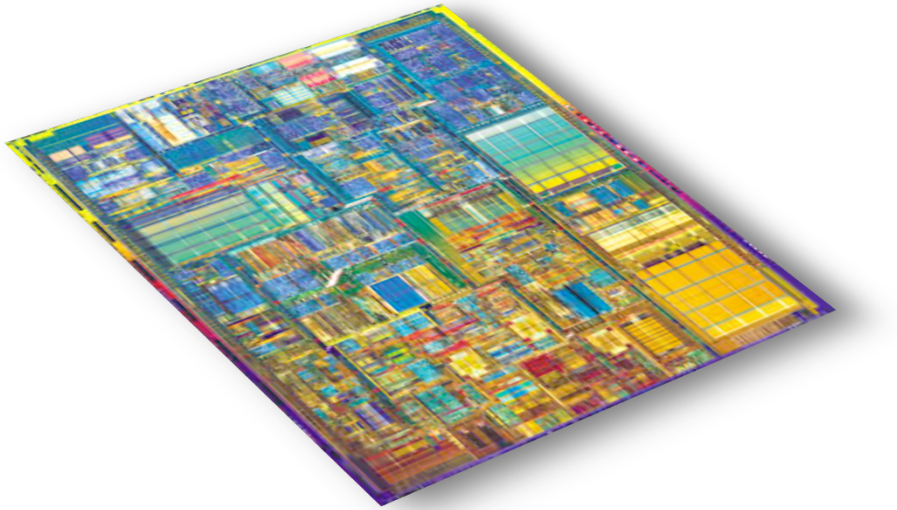
Laboratoire
Informatique
Robotique
Microélectronique
Montpellier

CIN Front-End Tutorial (script)



J.-M. Gallière - 2015

- In the SIMU work directory launch nclaunch :>nclaunch
- Select « Multiple Step »
- File > Set Design Directory…
- Click on « Create cds.lib File… », then « Save » « OK » « OK »
- Quit nclaunch : File > Exit

Before working with a simulation script it is necessary to launch "nclaunch" in graphical mode in order to configure your working directory

- Launch the script by :> source simu_scritp.tcl
- Before to launch the script again :> ncrm -message -library worklib

Files tree:

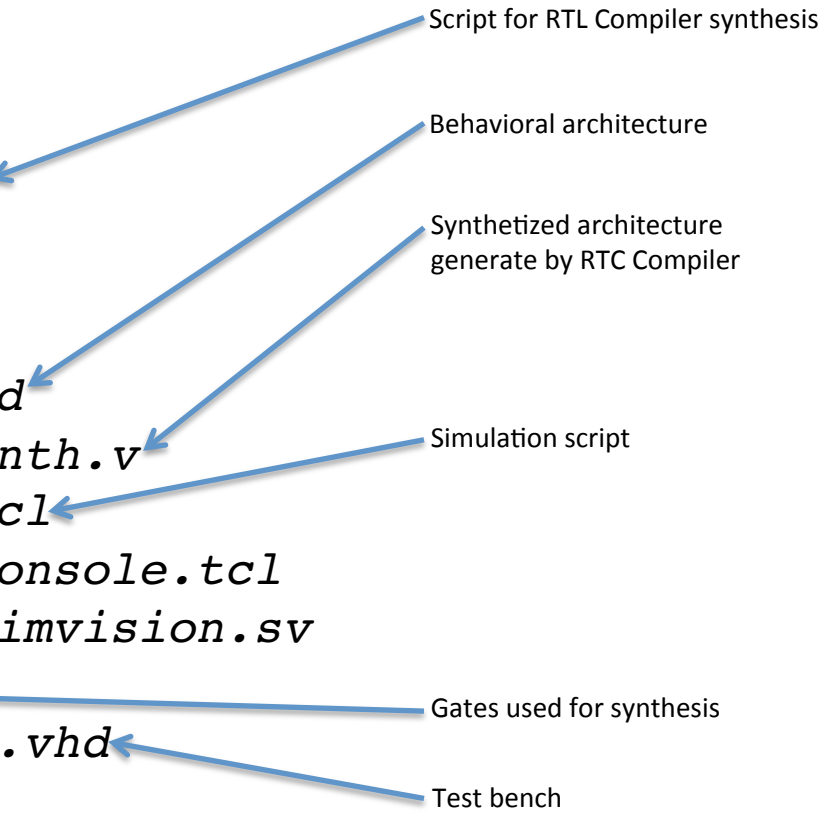
user_name/CIN/

RC/

rc_script.tcl

SIMU/

cds.lib
multiplier.vhd
multiplier_synth.v
simu_script.tcl
simu_script_console.tcl
simu_script_simvision.sv
slow.v
tb_multiplier.vhd



```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity multiplier is
port      (
          clk                 : in  std_logic;
          reset               : in  std_logic;
          num1, num2          : in  std_logic_vector(7 downto 0);
          product             : out std_logic_vector(15 downto 0)
          );
end entity;

architecture multiplier of multiplier is
signal num1_i : std_logic_vector(7 downto 0) ;
signal num2_i : std_logic_vector(7 downto 0) ;
begin
  process (clk, reset)
  variable reg : std_logic_vector(15 downto 0);
  variable add : std_logic_vector(8 downto 0);
          begin
           if reset = '1' then
                    num1_i<="00000000";
                    num2_i<="00000000";
                    product <= "0000000000000001";
           elsif rising_edge(clk) then
           num1_i<=num1;
           num2_i<=num2;
           reg:="00000000"&num2_i;
           for i in 1 to 8
            loop
              if reg(0)='1' then
              add:=('0'&num1_i)+('0'&reg(15 downto 8));
              reg:=add&reg(7 downto 1);
              else
              reg:='0'&reg(15 downto 1);
              end if;
            end loop;
       product<=reg;
            end if;
  end process;
end architecture;
```

Input architecture
multiplier.vhd

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity tb_multiplier is
end tb_multiplier;

architecture tb_multiplier of tb_multiplier is

component multiplier
port    (
        clk             : in  std_logic;
        reset           : in  std_logic;
        num1, num2      : in  std_logic_vector(7 downto 0);
        product         : out std_logic_vector(15 downto 0)
        );
end component;

signal  clk, reset : std_logic;
signal  num1, num2 : std_logic_vector(7 downto 0);
signal  product : std_logic_vector(15 downto 0);

constant periode : time := 6000 ps;

begin
uut : multiplier port map (
        clk => clk,
        reset => reset,
        num1   => num1,
        num2   => num2,
        product    => product
        );
```

The clock's period is set here

```
horloge : process
        begin
        clk <= '1';
        wait for periode/2 ;
        clk <= '0';
        wait for periode/2 ;
end process;

tb : process
begin
        num1 <= "00000000";
        num2 <= "00000000";
        reset <= '0';
        wait for 1 ns;
        reset <= '1';
        wait for 1 ns;
        reset <= '0';
        wait for 2 ns;
        wait for periode;
        num1 <= "10101010";
        num2 <= "01010101";
        wait for periode ;
        num1 <= "11111111";
        num2 <= "11111111";
        wait for periode ;
        num1 <= "00000010";
        num2 <= "00000011";
        wait for periode ;
        num1 <= "00001000";
        num2 <= "00000101";
        wait for periode ;
        num1 <= "01010000";
        num2 <= "00001001";
        wait;
end process;
end;
```

Input testbench *tb_multiplier.vhd*

Simulation script
simu_script.tcl

```
ncvhdl -work worklib -cdslib ./cds.lib -logfile ncvhdl.log -errormax 15 -update -v93 -linedebug -status ./multiplier.vhd

#ncvlog -work worklib -cdslib ./cds.lib -logfile ncvlog.log -errormax 15 -update -linedebug -status ./slow.v
#ncvlog -work worklib -cdslib ./cds.lib -logfile ncvlog.log -errormax 15 -update -linedebug -status ./multiplier_synth.v

ncvhdl -work worklib -cdslib ./cds.lib -logfile ncvhdl.log -errormax 15 -update -v93 -linedebug -status ./tb_multiplier.vhd

ncelab -work worklib -cdslib ./cds.lib -logfile ncelab.log -errormax 15 -access +wc -novitalaccl -status -v93 -timescale 1ns/1ps
                                        -coverage block -coverage expr -coverage toggle worklib.tb_multiplier

ncsim -gui -cdslib ./cds.lib -logfile ncsim.log -errormax 15 -covoverwrite -status worklib.tb_multiplier:tb_multiplier
                                        -input simu_script_console.tcl
```

The input architecture is chosen here
→ comment

The testbench is the same for each architecture

Simulation script console
simu_script_console.tcl

```
database -open -shm -into waves.shm waves -default
probe -create -database waves :clk :reset :num1 :num2 :uut.num1 :uut.num2 :product
simvision –input simu_script_simvision.sv
```

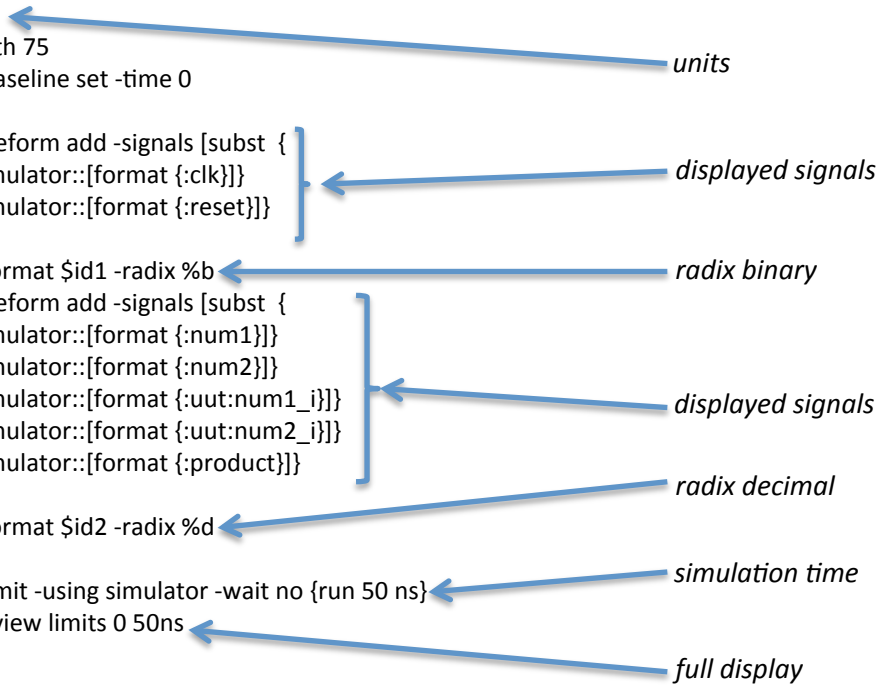
Next page…

cds.lib

```
define worklib ./INCA_libs/worklib
include $CDS_INST_DIR/tools/inca/files/cds.lib
```

Simulation script console
simu_script_simvision.sv

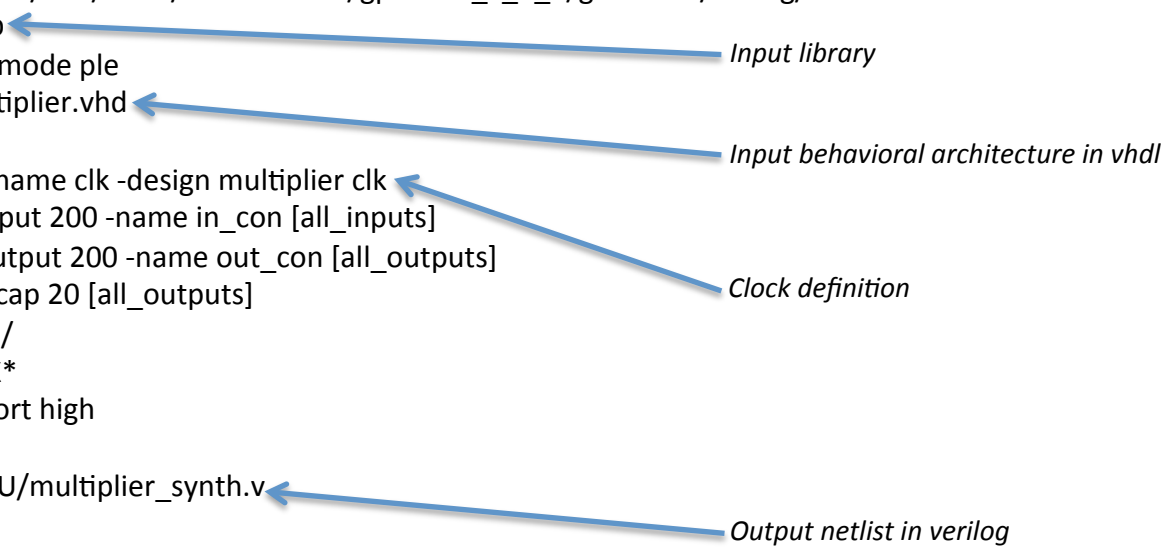
```
if {[catch {window new WaveWindow -name "Multiplier 8 bits" -geometry 1010x600+612+730}] != ""} {window geometry "Multiplier 8 bits" 1010x600+612+730}

window target "Multiplier 8 bits" on
waveform using {Multiplier 8 bits}
waveform sidebar visibility partial
waveform set \
   -primarycursor TimeA \
   -signalnames name \
   -signalwidth 175 \
   -units ns \
   -valuewidth 75
waveform baseline set -time 0

set id1 [waveform add -signals [subst  {
            {simulator::[format {:clk}]}
            {simulator::[format {:reset}]}
            } ]]
waveform format $id1 -radix %b
set id2 [waveform add -signals [subst  {
            {simulator::[format {:num1}]}
            {simulator::[format {:num2}]}
            {simulator::[format {:uut:num1_i}]}
            {simulator::[format {:uut:num2_i}]}
            {simulator::[format {:product}]}
            } ]]
waveform format $id2 -radix %d

console submit -using simulator -wait no {run 50 ns}
waveform xview limits 0 50ns
```



Synthesize with RTL Compiler

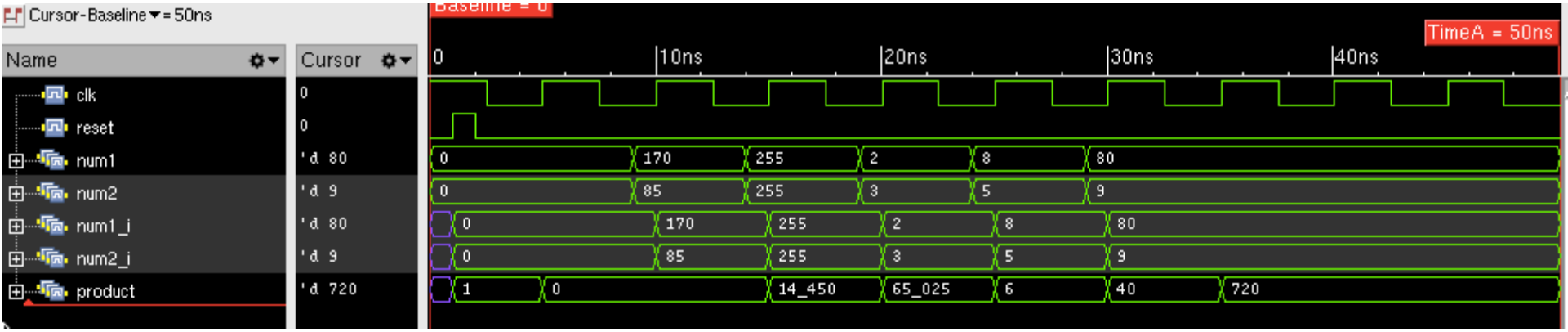
rc_script.tcl

```
set_attribute information_level 0
set_attribute lib_search_path /soft/DKits/CadencePDK/gpdk045_v_3_5/gsclib045/timing/
set_attribute library slow.lib
set_attribute interconnect_mode ple
read_hdl -vhdl ../SIMU/multiplier.vhd
elaborate
define_clock -period 6000 -name clk -design multiplier clk
external_delay -clock clk -input 200 -name in_con [all_inputs]
external_delay -clock clk -output 200 -name out_con [all_outputs]
set_attribute external_pin_cap 20 [all_outputs]
set_attribute tns_opto true /
set_attribute avoid true CLK*
synthesize -to_mapped -effort high
report timing -worst 1
write_hdl -mapped > ../SIMU/multiplier_synth.v
rm /designs/multiplier/
exit
```

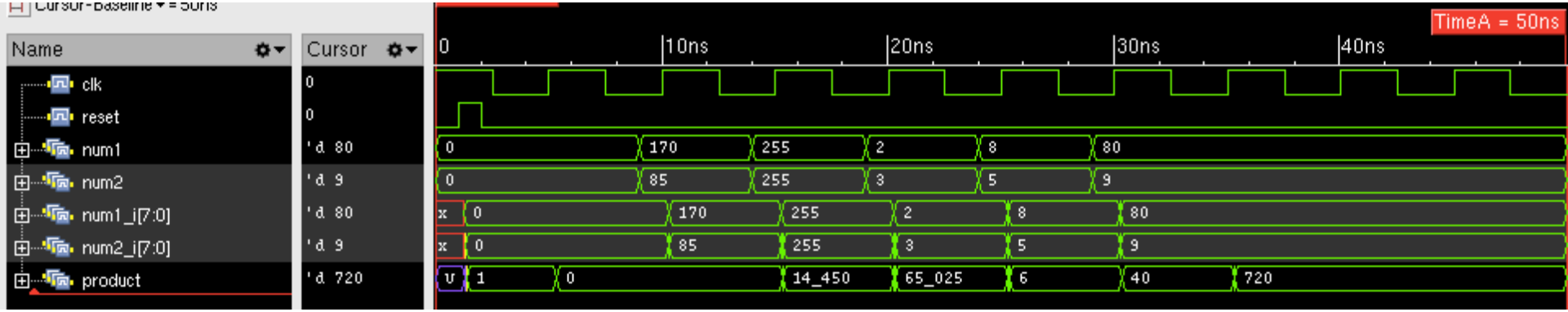


In the RC work directory launch the script by :> rc -nogui -files rc_script.tcl
The synthesize output file is copied in the SIMU work directory
You can modify the *simu_script.tcl* to take into account this file and the *slow.v* file (comment (#) the *multiplier.vhd* file)

Results obtained for behavioral architecture:



Results obtained for synthesized architecture(with delays):



Remark :
When no timing are includes in the technology description file (*slow.v* in the previous example), you must perform the simulation in the following way.

1) Create the SDF file at the end of the synthesis process (slide #7)

```
write_sdf  -edges check_edge   ../SIMU/multiplier.sdf
```

2) Create the *sdf.cmd* file :

```
COMPILED_SDF_FILE = "./multiplier.sdf.X",
SCOPE = :uut,
LOG_FILE = "sdf.log",
MTM_CONTROL = "TOOL_CONTROL";
```

3) Compile and Elaborate with the SDF file in the *simu_script.tcl*

```
(add the following line in the script)
ncsdfc -compile -cdslib ./cds.lib -output ./multiplier.sdf.X -status ./multiplier.sdf

(modify the following line in the script)
ncelab -SDF_CMD_FILE  ./sdf.cmd  -SDF_FILE ./multiplier.sdf.X  -MAXDELAYS
                -work worklib -cdslib ./cds.lib -logfile ncelab.log -errormax 15
                -access +wc -novitalaccl -status -v93 -timescale 1ns/1ps
                -coverage block -coverage expr -coverage toggle worklib.tb_multiplier
```