Lab 3 PLD state machine

EET 207

Repeat lab 2 but use VHDL to target the Altera MAX 3128 PLD.

Download from

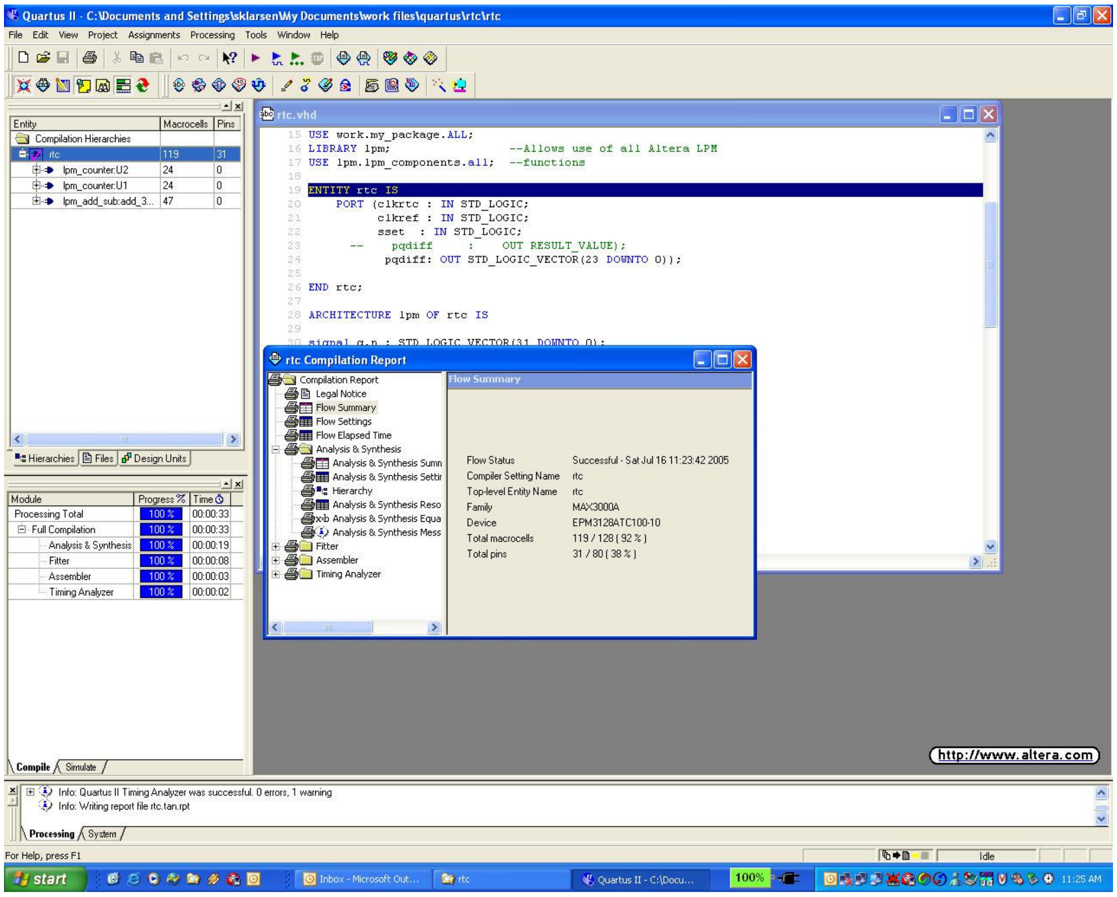
https://www.altera.com/support/software/download/sof-download_center.html

the Quartus II free web edition. (This is 227MB so a broadband connection is needed. If you do not have one, the instructor can provide a CD in the lab) Quartus II allows you to synthesize VHDL to target the MAX 3128 PLD as well as a logic simulator. Please include logic simulation in your lab report. This "free" software requires a license registration of your NIC. The installation walks you through the process to get the license.

The PLD board is available from the lab instructor. Please contact him to get this board if you are not going to be in the lab session. There is a parallel port cable that goes with this circuit board. If you do not have your own PC to use, you can use one of the lab computers.

As can be seen from the silkscreen on the boards, pins 24, 25, 27, 28 are switches 1,2,3,4 respectively. The LEDs can be controlled by segments A,B,C,D,E,F,G on pins 70, 71,72,75,76,77, 80. The board has a 32MHz clock coming in on pin 87.

With your VHDL file, a completed synthesis and place/route would result in a screenshot as shown below:



Quartus II - C:\Documents and Settings\sklarsen\My Documents\work files\quartus\rtc\rtc

Entity	Macrocells	Pins
Compilation Hierarchies		
rtc	119	31
lpm_counter:U2	24	0
lpm_counter:U1	24	0
lpm_add_sub:add_3...	47	0

Module	Progress %	Time
Processing Total	100 %	00:00:33
Full Compilation	100 %	00:00:33
Analysis & Synthesis	100 %	00:00:19
Fitter	100 %	00:00:08
Assembler	100 %	00:00:03
Timing Analyzer	100 %	00:00:02

rtc.vhd

```
USE work.my_package.ALL;
LIBRARY lpm;                    --Allows use of all Altera LPM
USE lpm.lpm_components.all;  --functions

ENTITY rtc IS
    PORT (clkrtc : IN STD_LOGIC;
          clkref : IN STD_LOGIC;
          sset   : IN STD_LOGIC;
      --    pqdiff     :    OUT RESULT_VALUE);
           pqdiff: OUT STD_LOGIC_VECTOR(23 DOWNTO 0));

END rtc;

ARCHITECTURE lpm OF rtc IS

signal q,n : STD_LOGIC_VECTOR(31 DOWNTO 0);
```

rtc Compilation Report — Flow Summary

Flow Status	Successful - Sat Jul 16 11:23:42 2005
Compiler Setting Name	rtc
Top-level Entity Name	rtc
Family	MAX3000A
Device	EPM3128ATC100-10
Total macrocells	119 / 128 (92 %)
Total pins	31 / 80 (38 %)

```
Info: Quartus II Timing Analyzer was successful. 0 errors, 1 warning
Info: Writing report file rtc.tan.rpt
```

Another sample design using this board is located on webct. It is a simple clock divider, but gives a sample complete design and a waveform stimulus.

A sample VHDL state machine is listed below and can be modified to realize the car buzzer state machine you did in lab 2. Replace or delete each text string noted with "__xxxxx" Email me with questions you have on text formatting.

```
--EET207 Example VHDL file
-- Your name and date

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
use IEEE.std_logic_unsigned.all;
USE ieee.std_logic_arith.all;
LIBRARY lpm;                  --Allows use of all Altera LPM
USE lpm.lpm_components.all;  --function

ENTITY __machine_name IS
        PORT(
                clk                           : IN    STD_LOGIC;
                reset                         : IN    STD_LOGIC;
                __input_name, __input_name    : IN    STD_LOGIC;
                __output_name, __output_name  : OUT   STD_LOGIC);
END __machine_name;
```

```
ARCHITECTURE a OF __machine_name IS
        TYPE STATE_TYPE IS (__state_name, __state_name, __state_name);
        SIGNAL state: STATE_TYPE;
BEGIN
        PROCESS (clk)
        BEGIN
                IF reset = '1' THEN
                        state <= _state_name;
                ELSIF clk'EVENT AND clk = '1' THEN
                        CASE state IS
                                WHEN __state_name =>
                                        IF __condition THEN
                                                state <= __state_name;
                                        END IF;

                                WHEN __state_name =>
                                        IF __condition THEN
                                                state <= __state_name;
                                        END IF;

                                WHEN __state_name =>
                                        IF __condition THEN
                                                state <= __state_name;
                                        END IF;

                        END CASE;
                END IF;
        END PROCESS;

        WITH state SELECT
                __output_name   <=      __output_value WHEN    __state_name,
                                                       __output_value WHEN    __state_name,
                                                       __output_value WHEN    __state_name;
END a;
```