ECSE-323
Digital System Design

Lab #4 – ***Sequential Circuit Design with VHDL*** Winter 2014

Introduction

In this lab you will learn how to use the Altera Quartus II FPGA design software to implement Counters described in VHDL.

You will design a year-month-day calendar counter and a module that converts Earth time and date to Mars time.

Learning Outcomes

After completing this lab you should know how to:

• Describe a sequential circuit with VHDL, using process block statements

Table of Contents

This lab consists of the following stages:

1. Design of an Years-Months-Days Counter .

Using VHDL, with a process block, design a counter circuit that counts Years, Months and Days on Earth. Do not use LPM components for this. The Days count should range from 1 to a maximum of 31, with the maximum depending on the current Month count. ***Include Leap years in your determination of the maximum day count.*** The Months counts should range from 1 to 12, and the Years count should range from 0 to 4000.

The Days counter should increment whenever a synchronous enable control input (*day_count_en*) goes high. The Months counter should increment whenever the Days count goes from its maximum back to 1. The Years counter should increment whenever the Months count goes from 12 back to 1.

The VHDL design entity should be named ***gNN_YMD_Counter*** (where NN is the number of your group) and should have the following inputs and outputs:

- **clock** (1-bit asynchronous input, should be connected to the master 50MHz clock)
- **reset** (1-bit asynchronous input, when high the counts are all set to zero)
- **day_count_en** (1-bit synchronous input, a pulse with a width of 1 master clock cycle)
- **load_enable** (1-bit synchronous input, if high the count values will be set to the values at the Y_Set, M_Set, and D_Set inputs)
- **Y_Set, M_Set, D_Set** (12-bit, 4-bit, 5-bit synchronous inputs, these values are copied to the count outputs when load_enable is high)
- **Years, Months, Days** (12-bit, 4-bit, 5-bit synchronous outputs, corresponding to the count values)

Show the VHDL description to the TA, explaining how the system is intended to work.

TIME CHECK

You should be this far at the end of your *first* 2-hour lab period!

2. Simulation of the YMD Counter .

Compile your YMD counter circuit, and do a functional simulation. To save on simulation time, you can use a fast pulse signal for the *day_count_en* input, say with a period of one microsecond (no need to wait for a whole day!). Run the simulation for at least three simulated years.

Show the simulation results to your TA.

TIME CHECK

You should be this far at the end of your *second* 2-hour lab period!

3. YMD Counter Testbed Implementation .

Using VHDL, design a ***testbed*** circuit for the YMD counter. Connect one pushbutton on the Altera board to the counter reset and the other to the counter load enable input. Display the Years, Months, and Days values one at a time on the four 7-segment LEDs on the DE1 board, using two dipswitches to select which count is being displayed.

Also use the dipswitches on the Altera board to input the load data for the counter.

To test the YMD counter use the one second pulse generator (either MPULSE or EPULSE) from lab 3 as the source for the *day_count_en* signal.

Show the VHDL description to the TA.

Compile the circuit and download the design to the Altera board.

Test the circuit by observing the proper count sequencing, and the proper loading of the count data. Demonstrate the functioning of your test-bed to the TA.

TIME CHECK

You should be this far at the end of your *third* 2-hour lab period!

4. Synchronizing the Earth and Mars Clocks.

Your system will have two time-clocks, the Mars Clock and the Earth Clock, running in parallel, being driven by the Epulse and Mpulse signals.

A problem that you now have to solve in this lab is how to synchronize these two clocks. To do this, your system should allow the user to set the proper Earth time, and then convert this Earth time to the associated Mars time and use that to set the Mars clock.

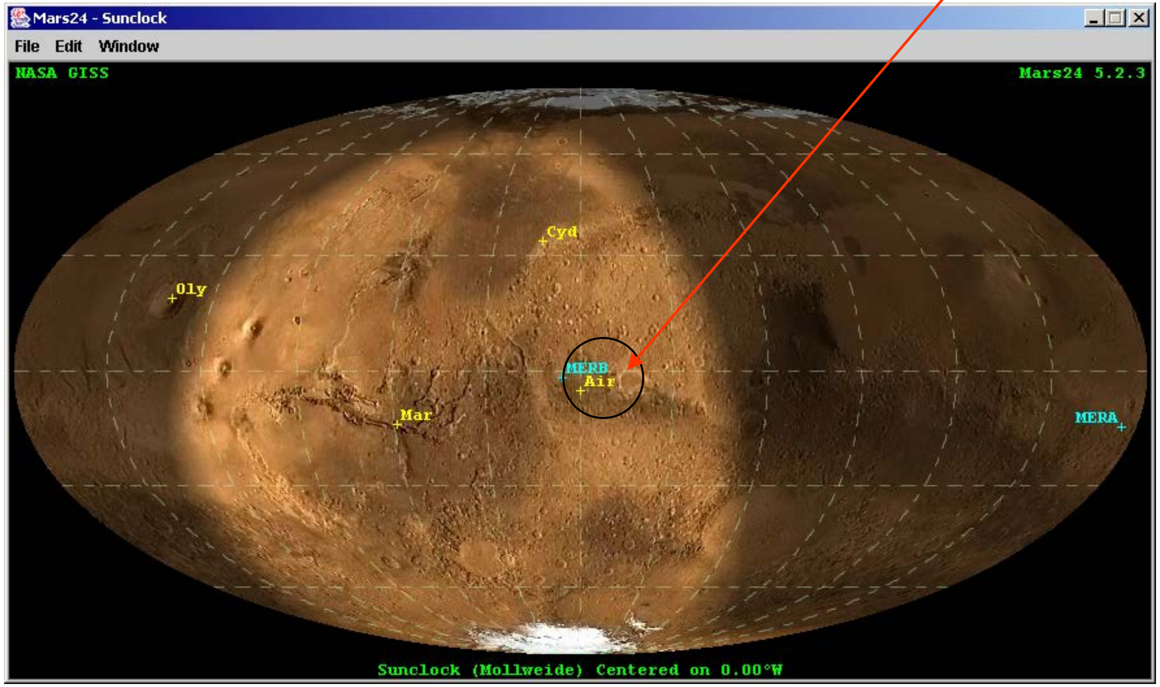
http://womenshistory.about.com/od/aviationspace/ig/Sally-Ride-Picture-Gallery/Sullivan---Ride-Before-Liftoff.htm

How do we do the conversion between Earth time and Mars time? It is trickier than you might think. One way is to find a date in time at which it is midnight (or noon, or some other reference time) at both the ***Prime Meridians*** on Mars and Earth, and then compute the time difference from that date.

Where is the Prime Meridian on Mars?

The prime meridian of Mars is defined by the location of the crater Airy-0. The Mean Solar Time (***MST***) on the Martian prime meridian is known as Mars Coordinated Time, or ***MTC***. On Earth the *Mean Solar Time* on the Prime (Greenwich) Meridian is known as *Universal Coordinated Time*, or ***UTC***.

On Mars, as on Earth, there are 24 time zones. Unlike on Earth, the Martian time zones are all equally spaced, at 15 degree intervals.



It turns out that midnight on the Prime Meridian of Mars coincided with noon on the Earth Prime Meridian on January 6, 2000 (on Mars it was actually 21 seconds before midnight). The midnights on the prime meridian also coincided on April 11, 1955.

Based on this, a research paper published by NASA gives the following conversion formula for the Earth time (expressed as a ***Julian day*** number) to the **MTC** on Mars (expressed as a day fraction in hours, from 0 to 23):

$$MTC = 24 * Frac\left[\frac{JD - 2451549.5}{1.02749125} - 0.00072\right]$$

Allison, M., and M. McEwen 2000. A post-Pathfinder evaluation of aerocentric solar coordinates with improved timing recipes for Mars seasonal/diurnal climate studies. *Planet. Space Sci.* **48**, 215-235
http://pubs.giss.nasa.gov/abstracts/2000/AllisonMcEwen.html

What is the Julian Date?

In the conversion formula given on the previous page, the quantity ***JD*** refers to the ***Julian Date*** on Earth.

From Wikipedia, the free encyclopedia (www.wikipedia.org):

*"The **Julian day** or Julian day number (JDN) is the number of days that have elapsed since 12 noon Greenwich Mean Time (UT or TT) on Monday, January 1, 4713 BC in the proleptic Julian Calendar. That day is counted as Julian day zero.*

…

*The **Julian Date** (JD) is the Julian day number plus the decimal fraction of the day that has elapsed since noon. The fraction of the day is found by converting the number of hours, minutes, and seconds after noon into the equivalent decimal fraction."*

To simplify our calculations a bit, let us define a ***modified Julian Date*** to be the (fractional) number of days since noon on January 6, 2000:

$$JD_{2000} = JD - 2451549.5$$

Computing the Julian Date.

Computing the (modified)Julian Date from a given time and date is not as simple as it may seem, since months and years are not all the same length. You could use a look-up table, but it would be very large since you would need one entry for every possible day that you would want to set the clock to.

A simpler approach is to make use of circuits that you already have - the ***gNN_YMD_Counter*** and ***gNN_HMS_Counter*** designed earlier. To compute JD_2000, initialize the *YMD_Counter* to the date January 6 2000, and the *HMS_Counter* to the time 00:00:00. Connect the *end_of_day* signal from the HMS counter to the *eod_clock* input of the YMD counter. Then, using a ***very fast*** *sec_clock* pulse, count up until the date and time reaches the desired values (e.g. March 17 2014, 13:45:00). As you count up, keep track of both the number of times the HMS counter is incremented (call this ***Nsecs***), and the number of times the YMD counter is incremented (call this ***Ndays***). Convert the value of *Nsecs* to a day fraction using the circuit you designed in lab 1 and call the result ***Dayfrac***. The modified Julian Date is then given by:

$$JD_{2000} = NDays.Dayfrac$$

The Conversion Formula (continued).

In terms of number of gates required, division operations are much more costly than multiplication operations. Thus, we should rewrite the conversion formula to require multiplication rather than division:

$$MTC = 24 * Frac\left[JD_{2000} * 0.973244297 - 0.00072\right]$$

It is up to you to determine how many bits you need to represent the various values in the above expression. You shouldn't need more than 32 bits for any value.

The last bit of business to take care of is to convert the value of MTC, which is between 0 and 24 (time on Mars in hours) to an Hours:Minutes:Seconds representation that you can load into your Mars HMD_counter.

Computing the Hours value is simple - just take the integral part of the value of MTC.

The Minutes value can be computed by multiplying the fractional part of MTC by 60, and keeping the integral part of the result.

The Seconds value can be computed by multiplying the fractional part of the previous result by 60 and keeping the integral part of this result.

5. Implementation of the MTC Computation .

Now put all these ideas together and write a VHDL description of a module, called *gNN_UTC_to_MTC*, which takes in an Earth time and date (UTC time on the prime meridian) expressed in Y:M:D:H:M:S, and generates the Mars time of day (MTC time on the prime meridian) expressed as in the H:M:S form needed to set the Mars clock.

Show the VHDL description of your circuit to the TA.

Once you have the UTC to MTC conversion circuit designed (using VHDL) it is important to simulate it, to make sure that it is working properly. Pay attention to the time needed by the various operations to complete. If the 50 MHz clock is too fast for an operation to complete, you will have to add in one or more wait states in your controller state diagram.

Show the results of the simulation to the TA and explain how your system works.

TIME CHECK

You should be this far at the end of your *fourth* 2-hour lab period!

8. Write-up of the Lab Report .

Write up two short reports, for the ***gNN_YMD_Counter*** and the ***gNN_UTC_to_MTC*** circuits that you designed in this lab.

The report must include the following items:

• A header listing the group number (and company name if you gave it one), the names and student numbers of each group member.
• A title, giving the name (e.g. ***gNN_YMD_Counter***) and function of the circuit.
• A description of the circuit's function, listing the inputs and outputs. Provide a pinout or symbol diagram.
•The VHDL description of the circuits.
• A discussion of how the circuit was tested, showing representative simulation plots and descriptions of the hardware tests.
• A summary of the FPGA resource utilization.

The report should be done in html or pdf (preferred), or in Microsoft Word, and uploaded to the myCourses site using the assignment 4 submission.

Make sure that you have uploaded ***all*** of the design files (e.g. .sof and .vhd files) used in your project.

The report is due at midnight, one week after the last day of the lab period, or Friday April 4.

Grade Sheet for Lab #4

Winter 2014.

Group Number:_______________.

Group Member Name:________________. Student Number:___________________.

Group Member Name:________________. Student Number:___________________.

Marks

1. The VHDL description of the YMD counter circuit
2. Functional simulation of the YMD counter circuit
3. The VHDL description of the YMD testbed
4. Testing of the YMD counter circuit
5. The VHDL description of the UTC to MTC conversion circuit
6. Functional simulation of the UTC to MTC conversion circuit

TA Signatures

Each part should be demonstrated to one of the TAs who will then give a grade and sign the grade sheet. Grades for each part will be either 0, 1, or 2. A mark of 2 will be given if everything is done correctly. A grade of 1 will be given if there are significant problems, but an attempt was made. A grade of 0 will be given for parts that were not done at all, or for which there is no TA signature.