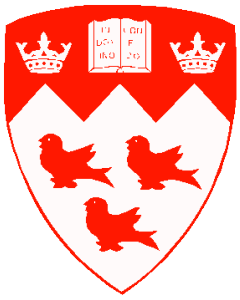


McGill

DIGITAL SYSTEM DESIGN (ECSE 323)

LAB 5: Earth-Mars Clock System

GROUP: 30
Habib Ahmed (260464679)
Bo Zheng (260481238)
11th April, 2014

Introduction

The purpose of this lab is to create a complete, standalone Mars Clock system on the Altera DE-1 board.

Description of the circuit

Over the course of 4 labs, we have developed individual design components that carry out certain tasks required by a complete clock system. In order to put these together and make them work coherently, it is crucial to use an interface controller which, in our case, is the Finite State Machine (FSM). The FSM along with some new design components make the system complete.

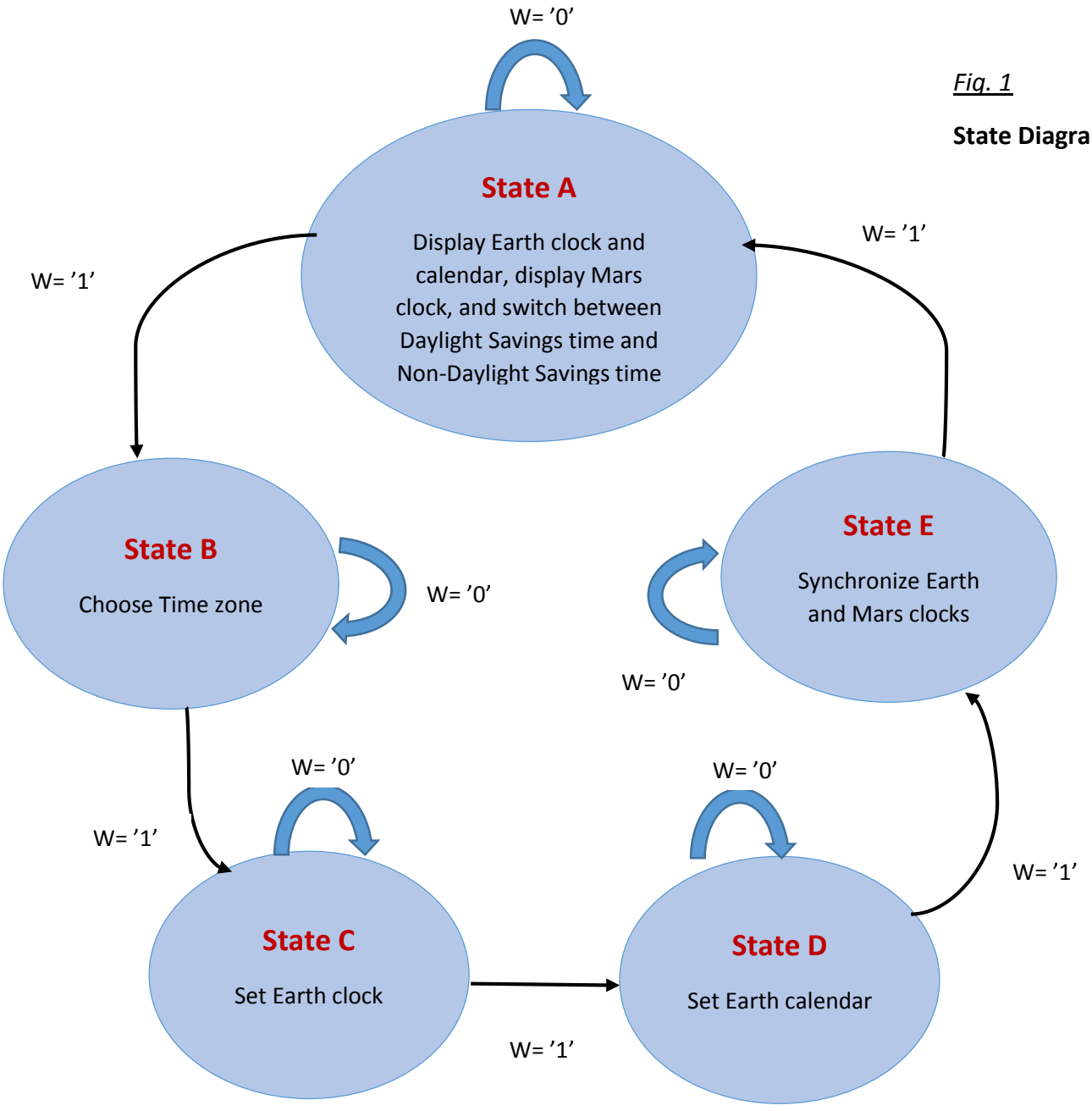


Fig. 1

State Diagram

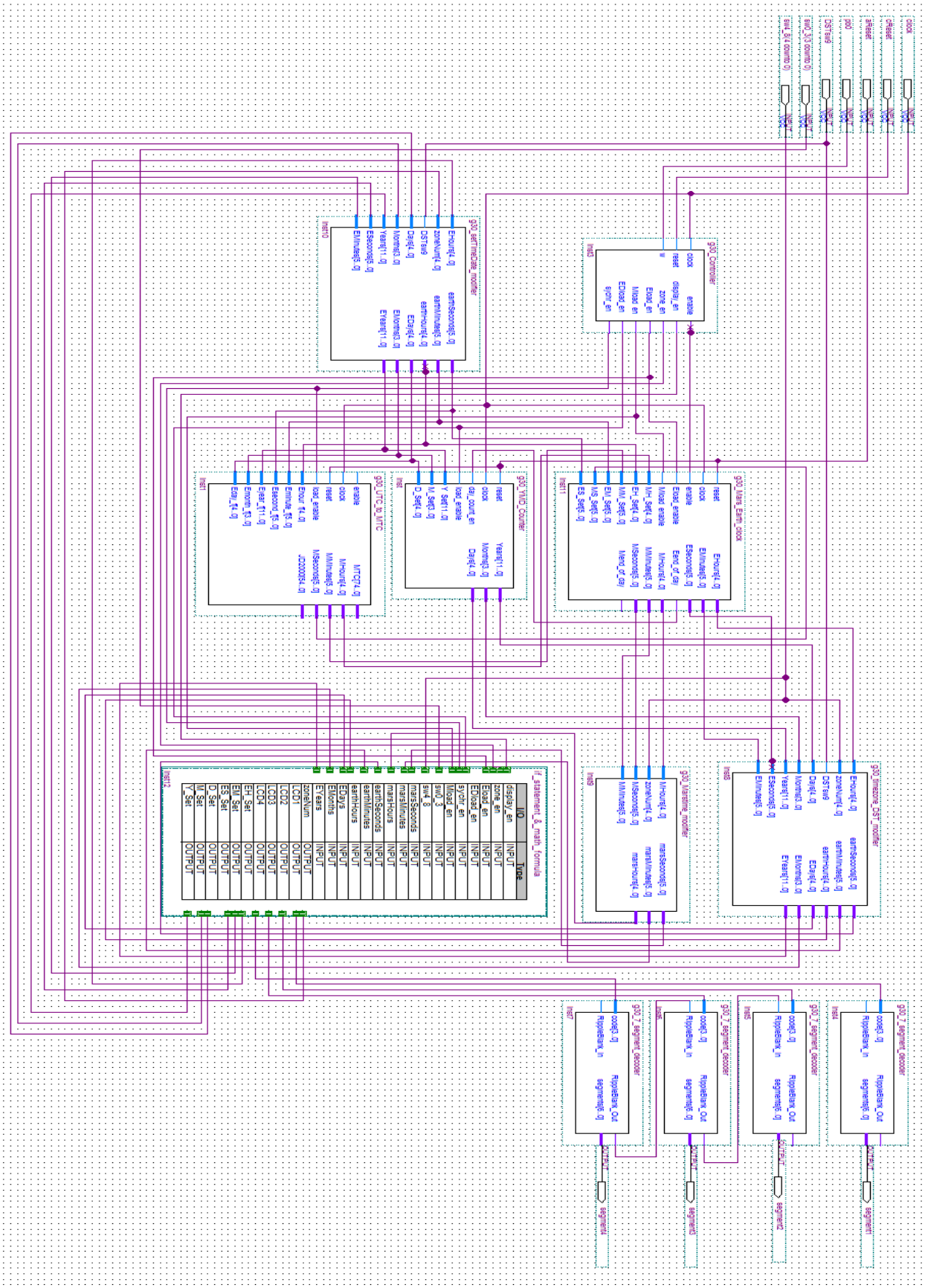


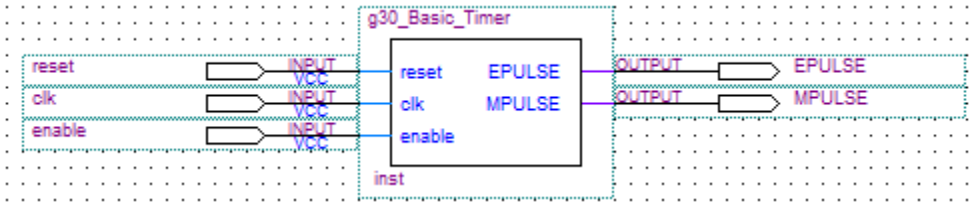
Fig. 2

Block Diagram

The overall system consist one controller block, 6 function blocks and 4 LCD decoder blocks. The working process starts from the controller mainly featured by a FSM. The FSM includes five states. With the input ph0, which is implemented by a pushbutton of the board, the transition among different states is controlled. To be specific, the first state produce an active display_en signal that is to display the Earth time, Mars time and Earth date. By pushing the button, the state enters in the second state which allows users to set the time zone by produces an active zone_en signal. In the third state, the Eload_en is active, which is input to the Mars_Earth_clock block to enable the setting-time function. The fourth state is to produce an active EDload_en signal, which enables the setting-date function. Also for the requirements from the zone_DST_time_modifier, the Eload_en is still high in this state. The next state produce an active synchr_en signal, which is the load enable signal to the UTC_to_MTC block, so in this state, a Mars setting time which be produced by synchronization with the current Earth setting time and this process needs about 5 seconds to be finished. Then, by pressing the pushbutton, it enters the last state, which produces an active Mload_en and then it will be input to the Mars_Earth_clock with the Mars setting time produced from the previous state.

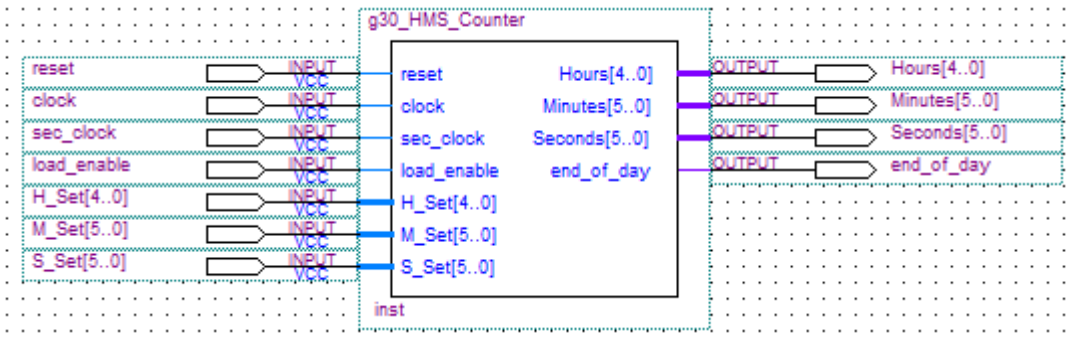
Listed below are all the individual design components that make up our system along with their individual pinout diagram.

g30_Basic_Timer:



This circuit is essentially a frequency divider circuit that makes use of the 50MHz clock on the Altera board. Each time the clock completes one cycle, i.e, every second, our clock increments by one thus allowing us to design a clock [see g30_HMS_Counter]. Slightly modifying our formula, we can increment our clock every ~1.02 seconds, thus allowing us to design a Mars clock as well. We cannot use the default frequency and need to make use of the divider circuit because the clock counts too fast and would not be visible otherwise.

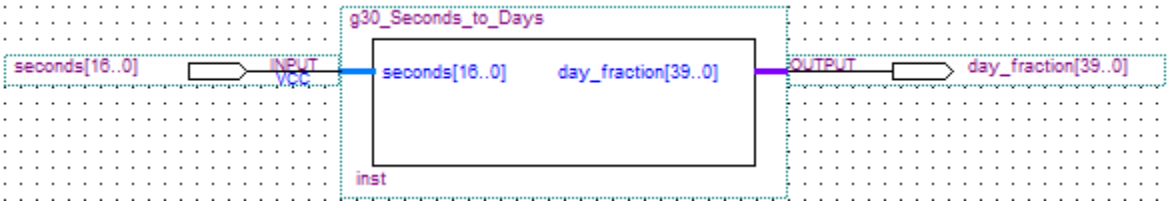
g30_HMS_Counter:



The diagram above shows all the inputs and the outputs of the circuit. The functions of this circuit are to set Hours, Minutes and Seconds as desired, increment Minutes when Seconds reach 59 and increment Hours when Minutes reach 59, display current time based on pulses from the *g30_Basic_Timer* circuit and output a signal "end_of_day" when our system has completed counting 1 whole day i.e, up to 23:59:59.

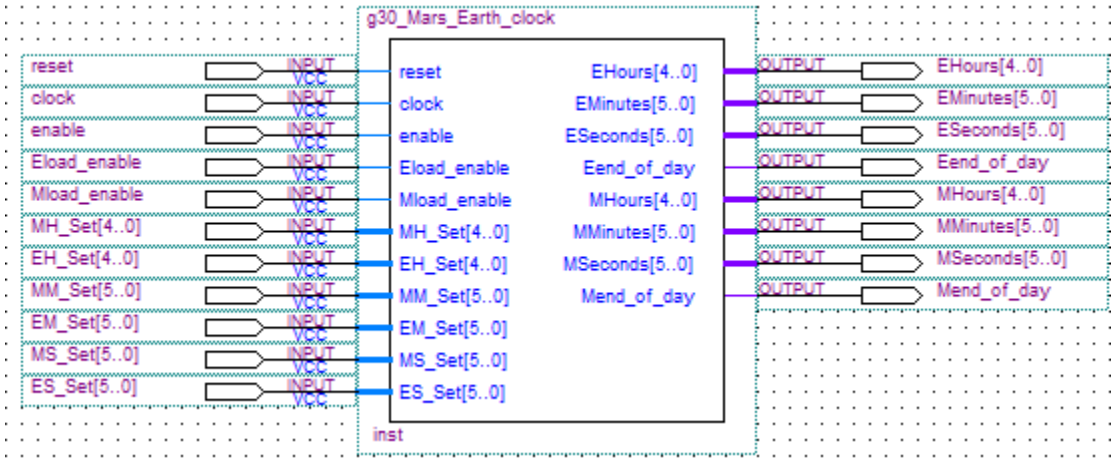
Using the Basic Timer circuit as a component, we can use this circuit to make the clocks for both Earth and Mars.

g30_Seconds_to_Days:



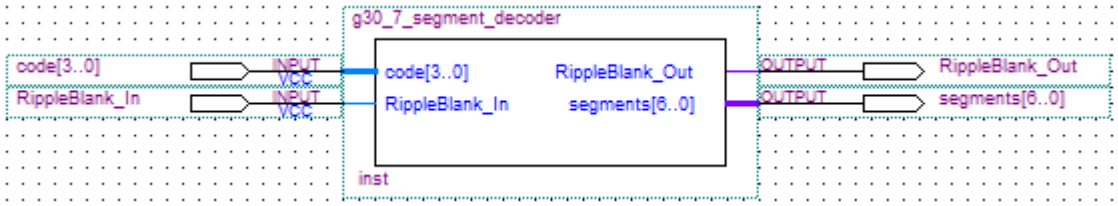
This circuit converts seconds into its fractional representation in days. This is used to Calculate JD2000.

g30_Mars_Earth_clock:



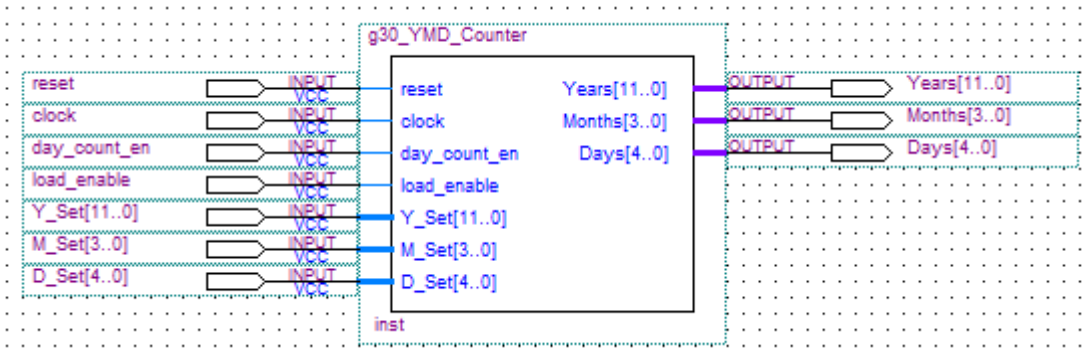
This circuit was used in the Testbed in Lab 3. The purpose of this circuit is to take as input an Earth clock and a Mars clock and give the output in Earth and Mars clocks after counting to a certain point.

g30_7_segment_decoder:



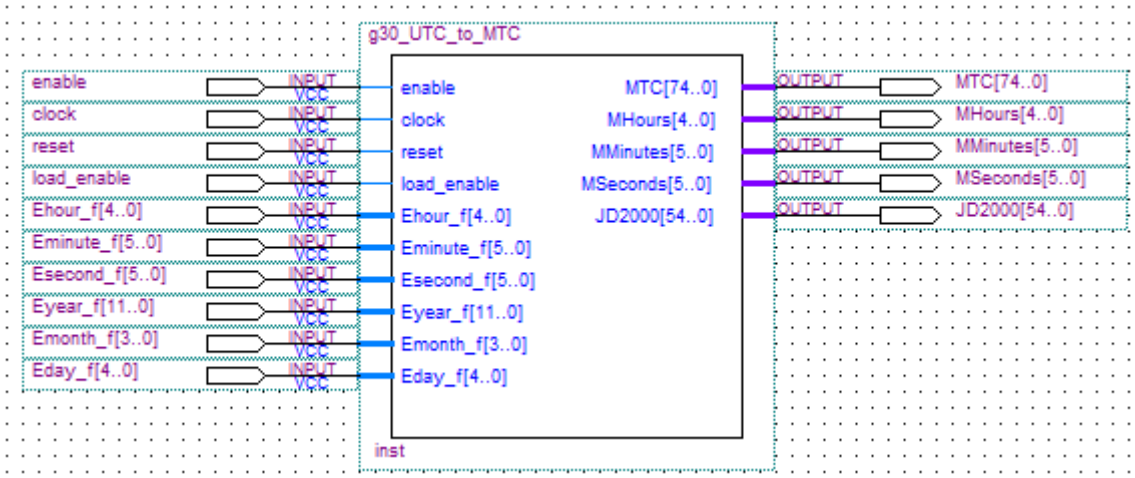
This circuit is used to display required numerical values on the four 7-segment displays on the board.

g30_YMD_Counter:



The objective of this circuit was to count the number of days, months as years as mentioned previously. The number of days would range from 0-31 depending on the current month, with leap years being taken into account. The number of months would be limited to 12 and the years count should range from 0 to 4000. In order to achieve our goals, we increment the "Days" counter whenever a synchronous enable control input (day_count_en) goes high. The month counter increments when the day counter reaches its maximum value and the years counter increments when the month counter reaches its maximum value. The "day_count_en" signal takes as input the value of the output "end_of_day" from the HMS_Counter circuit.

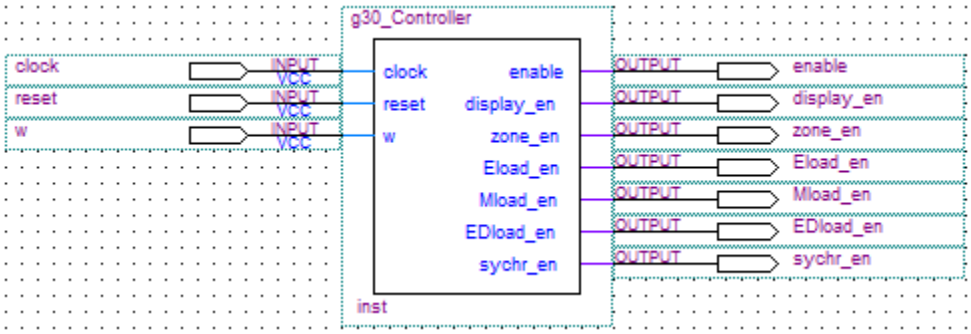
g30_UTC_to_MTC:



This circuit makes use of the g30_YMD_Counter, g30_HMS_Counter and the g30_Seconds_to_Days circuits to calculate the Julian Date, MTC. The MTC in turn is used to find out the time on Mars. To calculate the Julian Date, we used the YMD_Counter circuit to set the date to January 6, 2000 and the HMS_Counter circuit to set the time to 00:00:00. Then, the end_of_day signal from the HMS_Counter circuit is connected to the eod_clock input of the YMD_Counter circuit, thus enabling us to count everything from seconds to years in one circuit. On execution of our code, we kept track of the number of times the HMS and the YMD counters incremented their counters and called them Nsecs and Ndays respectively. Nsecs was later converted to a day fraction value called Dayfrac using our Seconds to Days circuit. Our Julian Date value and MTC were then calculated by formulas given in Lab 4. Once we calculated and found the value of MTC, the non-fractional part of it gives us the Hours in Mars, the non-fractional part

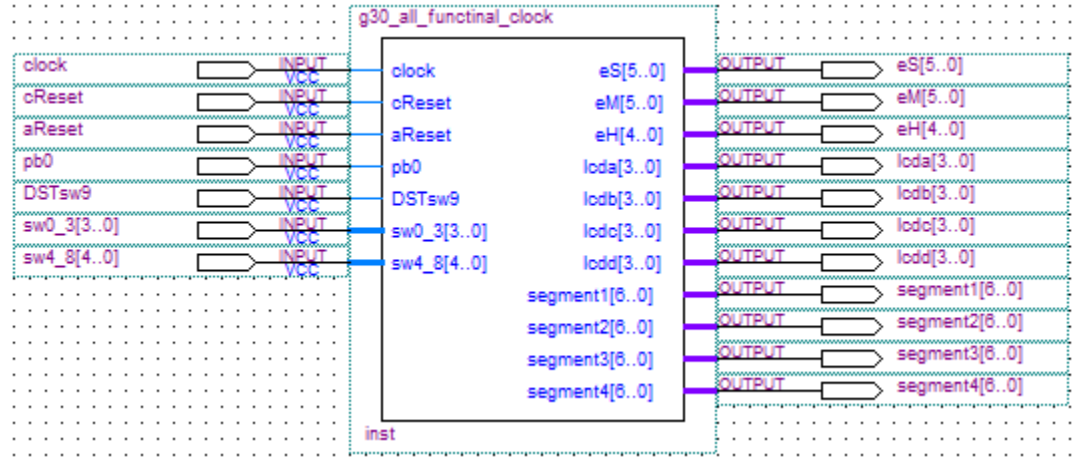
after multiplying the fractional part by 60 gives us the minutes and seconds is given by the non-fractional part when the remaining fractional part is multiplied by 60 again.

g30_Controller:



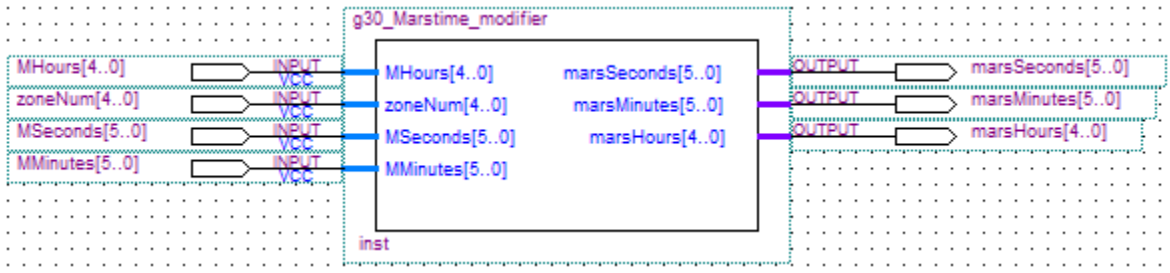
This circuit is essentially the FSM which includes 5 states. With the input implemented by a push button of the board, the transition among the states can be controlled. Also, outputs from each state will be sent to other blocks as enable signals.

g30_all_functional_clock:



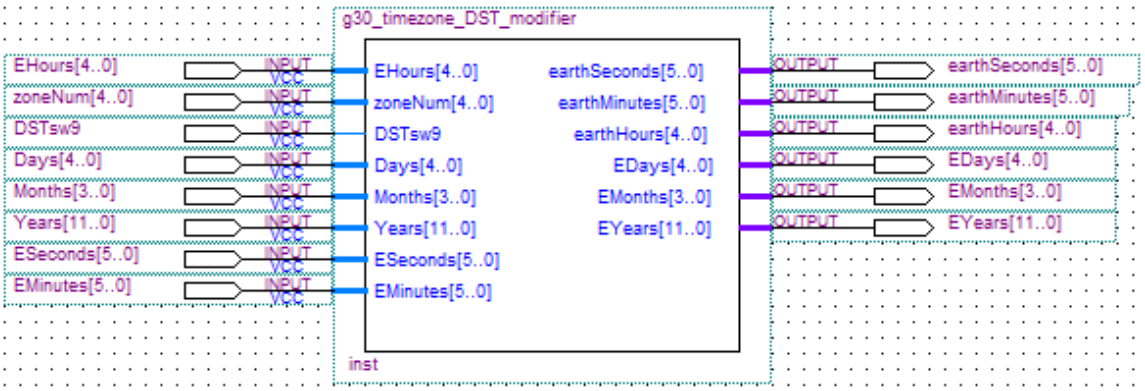
This circuit determines what to show on the 7-segment displays on the Altera board when we choose certain combinations of switches that correspond to a certain state. This contains all the VHDL components and this is the program that is run on the board.

g30_Marstime_modifier:



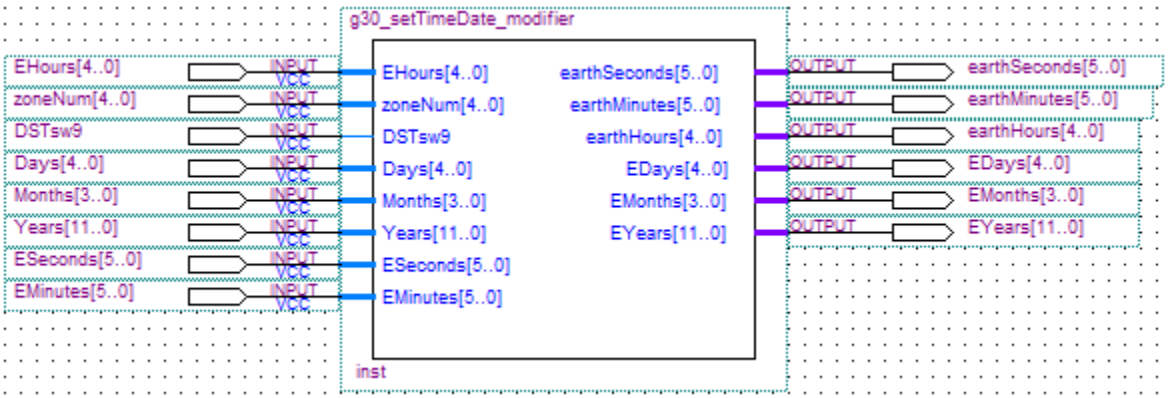
The Marstime_modifier block is also modified based on the timezone_DST_modifier. However, there is no DST and dates effects in this block. It computes the proper Mars time to display based on the time zone numbers. The outputs will be sent to if_statement block to obtain the inputs for the 7_segment_decoder block, which produced the LED segment codes.

g30_timezone_DST_modifier:



The timezone_DST_modifier is another important block made in this lab. It takes the earth time produced by the Mars_Earth_clock block as inputs. Based on the others inputs like the Days, Months, Years produced from the YMD_Counter, the zoneNum taken from the second state of the controller and the DST which is evaluated from the DST switch, the timezone_DST_modifier computes the desired earth time and earth date to display. On the other hand, it will not affect the UTC time and date values which are still involved in the Mars_Earth_clock block and the YMD_Counter block. These computed results are going to be sent to if_statement block and produce the inputs for LED decoder to obtain the segment codes for the LED display.

g30_setTimeDatemodifier:



The setTimeDate_modifier block is modified based on the timezone_DST_modifier. In the third and the fourth states of the FSM, this block receives the setting time and date from the switches of the board. Based on the zoneNum and DST values, it computes the equivalent UTC time and date setting values. These outputs will be taken as the inputs EH_Set, EM_Set, ES_Set, D_Set, M_Set and Y_Set by the Mars_Earth_clock block.

Description of the User Interface

This circuit is used to display required numerical values on the four 7-segment displays on the board. Due to our issues with designing the components and the FSM [see conclusion], we did not have a lot of time to make the system very user-friendly. In spite of this, our system turned out to be fairly straightforward. The instructions or user guide for our system is given:

I. Start the board using the red button.

II. Load the program on to the board and see that the clock starts running automatically. This is the default state, state **A** which displays the Earth and Mars clocks, the Earth calendar, as well as the option to switch between daylight savings and non-daylight savings time. For this state, switches SW0-SW3 and SW9 are valid. The switches as well as their respective results are given below.

Note: If values or switches are unlisted, then they are not valid.

SWITCH(ES) NAME/NUMBER	VALUE (1 = ON, 0 = OFF)	OUTPUT/RESULT
SW3 SW2 SW1 SW0	0000	Earth Second
SW3 SW2 SW1 SW0	0001	Earth Minute
SW3 SW2 SW1 SW0	0010	Earth Hour
SW3 SW2 SW1 SW0	0011	Earth Day
SW3 SW2 SW1 SW0	0100	Earth Month
SW3 SW2 SW1 SW0	0101	Earth Year
SW3 SW2 SW1 SW0	0110	Mars Second
SW3 SW2 SW1 SW0	0111	Mars Minute
SW3 SW2 SW1 SW0	1000	Mars Second
SW9	0	DST off
SW9	1	DST on

Table 1: Switches and results for State A

III. To move to the next state, press the **KEY0** button. This allows the program to go to state **B**, which allows us to choose the time zone. For this state, switches SW4-SW8 are valid. The switches as well as their respective results are given below.

Note: Time zone 0 signifies UTC and time zones greater than 11 signifies negative time zones. So, say time zone 14 is 24-14 = UTC-10.

SWITCH(ES) NAME/NUMBER	VALUE (1 = ON, 0 = OFF)	OUTPUT/RESULT
SW8 SW7 SW6 SW5 SW4	00000	Time zone 0
SW8 SW7 SW6 SW5 SW4	00001	Time zone 1
SW8 SW7 SW6 SW5 SW4	00010	Time zone 2
SW8 SW7 SW6 SW5 SW4	00011	Time zone 3
.	.	.
.	.	.
.	.	.
SW8 SW7 SW6 SW5 SW4	10111	Time zone 23

Table 2: Switches and results for State B

IV. To move to the next state, press the **KEY0** button. This allows the program to go to state **C**, which allows us to set the Earth clock. For this state, switches **SW0-SW5** and **SW7-SW8** are valid. The switches as well as their respective results are given below.

Note: For Seconds and Minutes, SW0-SW5 are valid whereas for Hours SW0-SW4 is valid. Also note that while setting values, the 7-segment displays might show values greater than 59. However, our VHDL code ensures that invalid values such as these are not accepted.

SWITCH(ES) NAME/NUMBER	VALUE (1 = ON, 0 = OFF)	OUTPUT/RESULT
SW8 SW7	00	Set Earth seconds
SW8 SW7	01	Set Earth minutes
SW8 SW7	11	Set Earth Hours
SW5 SW4 SW3 SW2 SW1 SW0	000000	Set value to 0
SW5 SW4 SW3 SW2 SW1 SW0	000001	Set value to 1
SW5 SW4 SW3 SW2 SW1 SW0	000010	Set value to 2
SW5 SW4 SW3 SW2 SW1 SW0	000011	Set value to 3
.	.	.
.	.	.
.	.	.
SW5 SW4 SW3 SW2 SW1 SW0	111011	Set value to 59

Table 3: Switches and results for State C

V. To move to the next state, press the **KEY0** button. This allows the program to go to state **D**, which allows us to set the Earth calendar. For this state, switches **SW0-SW5** and **SW7-SW8** are valid. The switches as well as their respective results are given below.

Note: For years, "20" is pre added to the first two (MSB) 7-segment LEDS. So if 14 is the value chosen using the switches, the year will be set to 2014. Also note that while setting values, the 7-segment displays might show invalid values (month =42). However, our VHDL code ensures that invalid values such as these are not accepted.

SWITCH(ES) NAME/NUMBER	VALUE (1 = ON, 0 = OFF)	OUTPUT/RESULT
SW8 SW7	00	Set Earth days
SW8 SW7	01	Set Earth months
SW8 SW7	11	Set Earth years
SW5 SW4 SW3 SW2 SW1 SW0	000000	Set value to 0
SW5 SW4 SW3 SW2 SW1 SW0	000001	Set value to 1
SW5 SW4 SW3 SW2 SW1 SW0	000010	Set value to 2
SW5 SW4 SW3 SW2 SW1 SW0	000011	Set value to 3
.	.	.
.	.	.
.	.	.

Table 4: Switches and results for State D

VI. To move to the next state, press the **KEY0** button. This allows the program to go to state **E**, which allows us to synchronise the Earth and Mars clocks. This state does not require the use of any switches.

VII. Press **KEY0** again to go back to state **A**.

How was the circuit tested:

We tested our circuit in two different ways, first by running a functional simulation on our main circuit, *g30_all_functional_clock.*

The simulation result was the following:

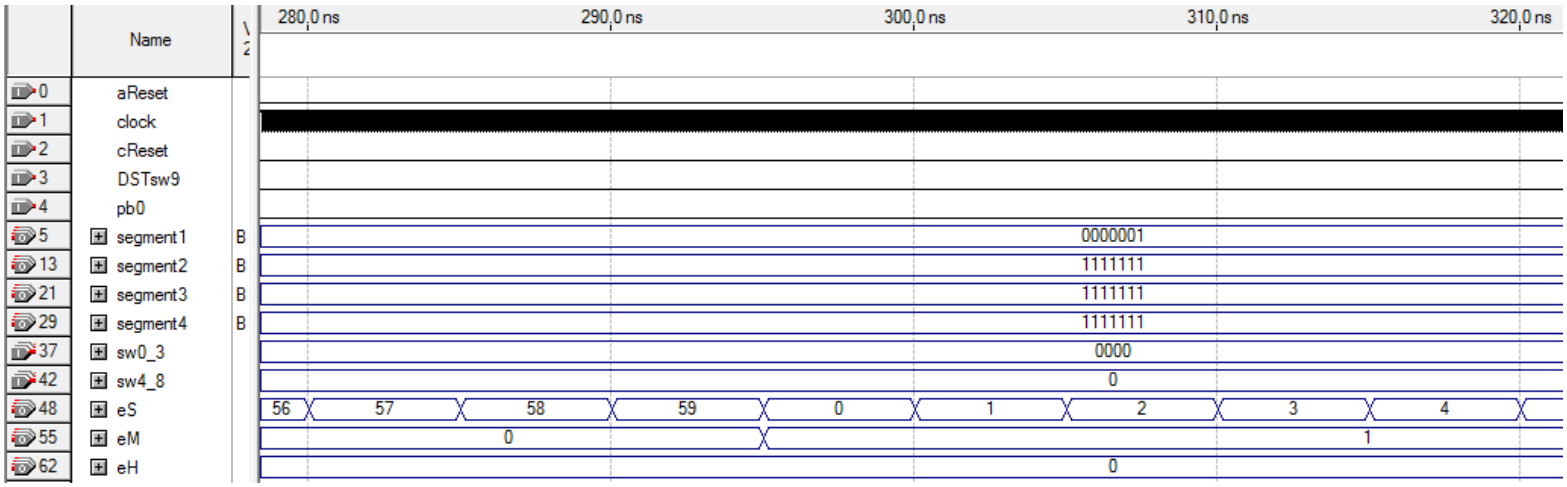


Fig. 3 **Vector Waveform**

The images suggests the test were successful.

Our next method of testing involved running our program on the Altera board extensively. Our test cases are given below:

Case 1

Test Objective: To see that the clock displays each of the required data on the 7-segment display.

Test Results: Our clock showed seconds, minutes, hours of both Earth and Mars as well as days, months and years for Earth as required by the lab objectives. The default state A showed these data. A more detailed guideline of viewing the data is given in the user interface description. Hence, our test was **successful**.

Case 2

Test Objective: Correctly set the Earth clock.

Test Results: We were asked by the TA to set our clock to the time shown on the computer and we were able to do so correctly. For a detailed description of setting the clock, please refer to the user interface description. This test was also **successful**.

Case 3

Test Objective: Correctly set the Earth calendar.

Test Results: The TA asked us to set the calendar to the current date and we were able to do so, correctly. For a detailed description of setting the calendar, please refer to the user interface description. Hence, this test was **successful**.

Case 4

Test Objective: Show that our program allowed to switch between regular and daylight savings time.

Test Results: Our toggle switch showed that the option to switch between regular and DST was available. For a detailed description selecting DST, please refer to the user interface description. This test was **successful**.

Case 5

Test Objective: Show that time zones were chosen correctly.

Test Results: The TA asked us to show that the hours changed accordingly when we choose the time zone to be 10(UTC+10). We reset our clock, chose the time zone to be 10 and showed the TA, on reverting back to state A, that the value of hours was set to 10. This test was **successful**.

Case 6

Test Objective: Show that the date changes when UTC + 0 is set to 00:00:00 and time zone is chosen to be UTC-2.

Test Results: We set the clock to be 00:00:00 and the date to be 11th April, 2014. Then, we chose the time zone to be 23 i.e, UTC-1. It was shown that the time is 23:00:00 on 10th April. This test was **successful**.

Case 7

Test Objective: Show that minutes incremented when seconds changed from 59 to 0

Test Results: Test was **successful**.

Case 8

Test Objective: Show that the Mars clock was indeed slower than the Earth clock.

Test Results: Toggling between Earth seconds and Mars seconds proved this test to be **successful**.

Case 9

Test Objective: Show that the Synchronise button works

Test Results: We set the Earth time and Calendar to the current time and date and pressed the synchronise button. However, there was no change in the Mars clock. Hence, this test was unsuccessful [see conclusion].

FPGA Resource Utilization Summary:

Flow Summary	
Flow Status	Successful - Fri Apr 11 19:02:09 2014
Quartus II 64-Bit Version	9.1 Build 350 03/24/2010 SP 2 SJ Full Version
Revision Name	g30_lab5
Top-level Entity Name	g30_all_functinal_clock
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Met timing requirements	No
Total logic elements	4,150 / 18,752 (22 %)
Total combinational functions	4,146 / 18,752 (22 %)
Dedicated logic registers	222 / 18,752 (1 %)
Total registers	222
Total pins	75 / 315 (24 %)
Total virtual pins	0
Total memory bits	512 / 239,616 (< 1 %)
Embedded Multiplier 9-bit elements	14 / 52 (27 %)
Total PLLs	0 / 4 (0 %)

Fig. 4

FPGA Summary

As we can see it took 22% of the total logic implements as well as 222 registers to implement our Earth-Mars clock system.

Conclusion:

Being unaccustomed to using FSMs, the first difficulty we faced was in determining states for our system. A lot of options were tried and tested to see how many states would be suitable for our program. Once we determined the number of states, it was difficult to choose the states, i.e, figure out what changes to be made at each state and which buttons to use to change them. By the time we came up with a fairly straightforward design, there was not enough time to refine the process. As a result, it was hard for the user to understand the working of our system. Had we been provided with more time, we would have utilized it to spend time giving light signals to tell the user which state he or she was on. Besides the design of the FSM, building the individual components did not turn out to be as difficult. Having made most of the components in previous labs, we made progress in designing the remaining design components fairly quickly. We were able to have a system that performed most of the functions set out to us in the project guidelines. The only task that we were unable to complete was the synchronisation of the Earth and Mars clocks. Given more time, that would be definitely be one of the other changes we would have made to our system. Other than this drawback, we believe our system worked really well and any more enhancements or changes would be unnecessary.