Digital Design I

Final Project

Digital Alarm Clock

Final Report

Presented to:

Dr. Mohamed Shalan

Done by:

Ali El-Said

Habiba Gamal

Noureldin Hashad

In this project we were required to design and implement a digital alarm clock. The design and implementation were done using a Basys2 FPGA board. The FPGA board has a seven segment display of 4 digits. The two digits on the left were used to display the hours and the two digits on the right were used to display minutes.

Objectives:

-The digital clock should have three modes:

- Clock mode (default mode)
- Adjust mode
- Stopwatch mode

-Transitions between the clock's states should be done using the push buttons on the FPGA board

-While in the default (clock) mode the display should display the current time with the LED off

-Pressing 'B1' button for two seconds should take the user to the adjust mode to adjust the current time or the alarm time depending on the state the user is in.

-Pressing 'B1' displays the alarm time and releasing it displays the current time.

-When the current time equals the alarm time the LED on the FPGA board flashes and an alarming sound is produced using the breadboard.

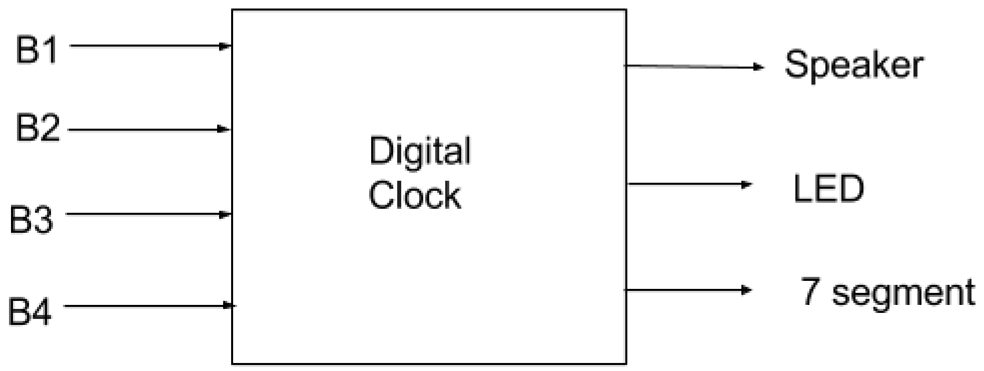


Figure 1: Black box for digital clock

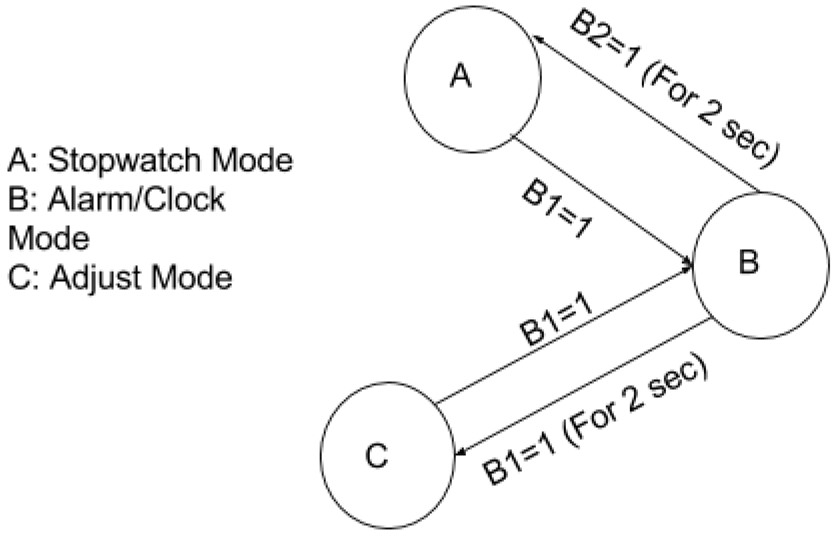
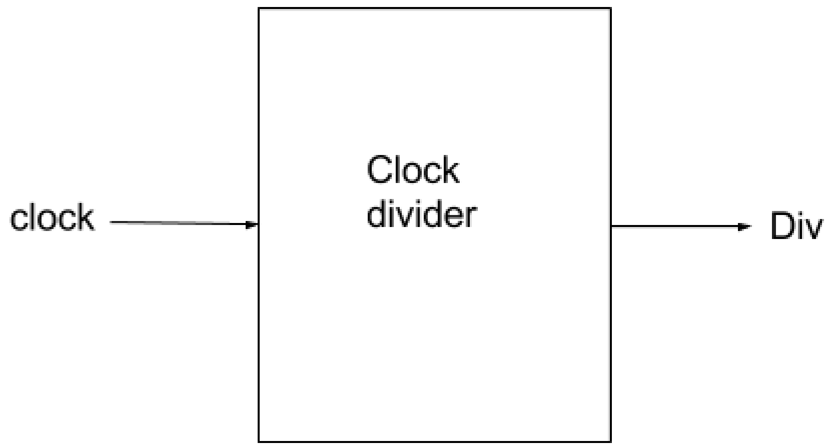
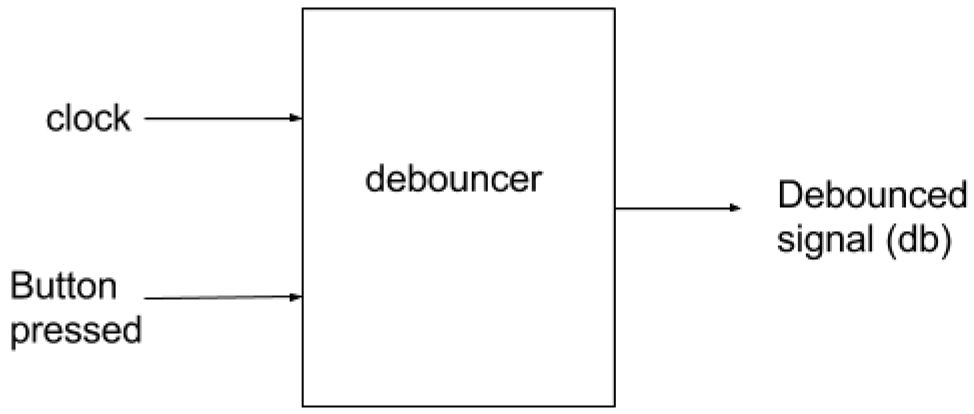
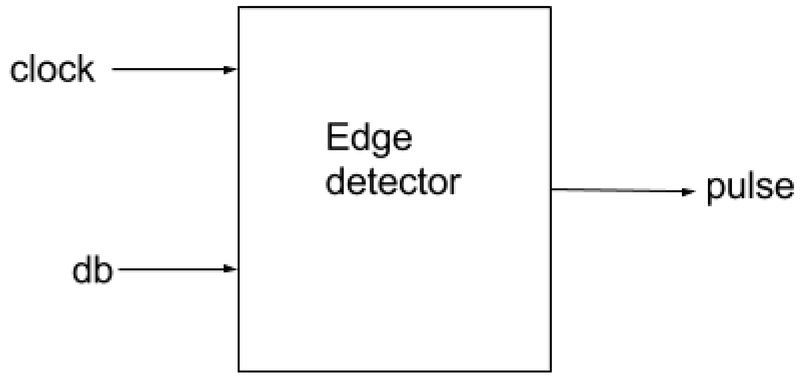


Figure 2: State Diagram for Digital Clock

Needed modules in implementation:





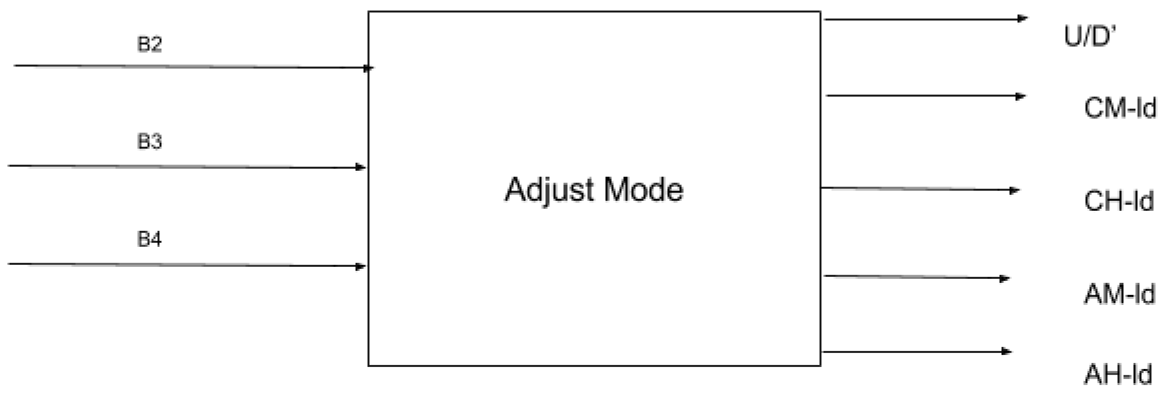


1) <u>Time Mode:</u>

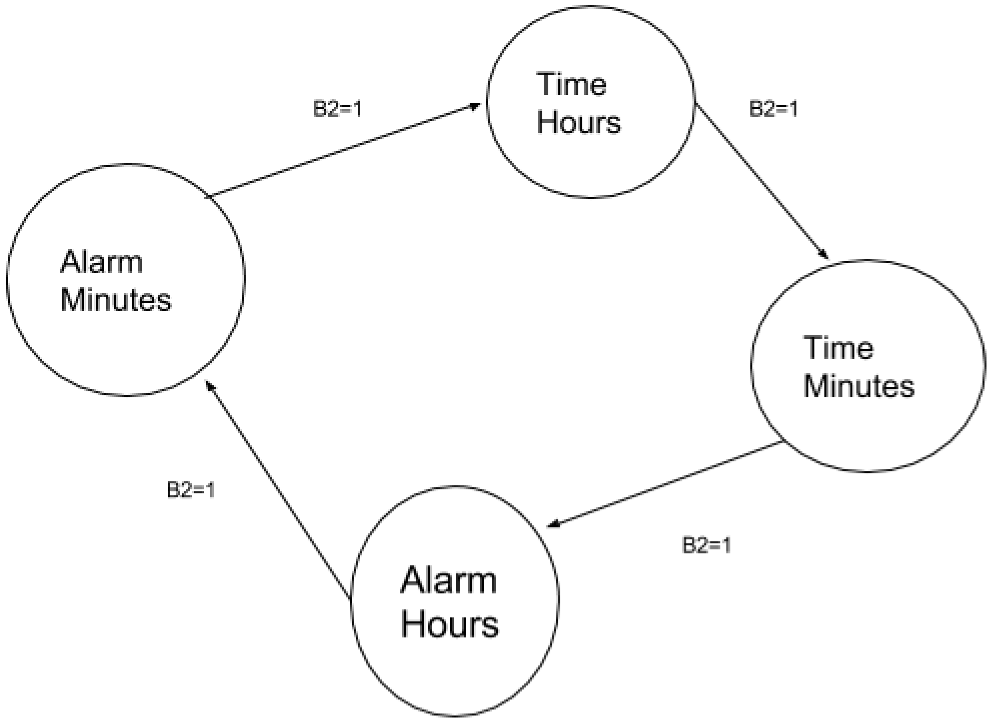
This module aims to count the minutes from 00:00 to 11:59 in 12-hours system or to 23:59 in 24 hours system. The inputs to the alarm mode are clk, reset, hours12_24, dataIN12, dataIN24.

The clk is the input clock of the FPGA (50 MHz), hours12_24 is the selection line. If it is 1 the clock is operating in 12 hours mode, else it it is 24 hours mode. The module contains 1 min counter that counts till 59. However, there are two separate counters for the hours in 12 hour system and for hours in 24 hour system. Both counters are enabled by the same enable, so they are both operating at the same time. However, the input hours12_24 is the selection of the multiplexer that chooses the input that goes to the binaryToBCD module. This module outputs the tens, ones and hundreds(ignored in our design), ready to be inputted into the seven segment module. These 7 bits are outputted from the module. The module has a ring counter module that enables one of the digits of the seven segment display. A multiplexer chooses which digit from the first hour digit, second hour digit, first minute digit, second minute digit & the selection line is the enable outputted from the ring counter. As for the dataIN12 and dataIN24 are used to load the counters for the 12 hours and 24 hours, respectively, with the current time of the clock(or the time that changed through the adjust mode). Other outputs include dataOUT_12 and dataOUT_24, which are the current time in both 12 hours system and 24 hours system, respectively.

2) Adjust Mode:



This module changes the current time or the alarm time based on the buttons pressed by the user. The inputs to this module are clk, B2, B3, B4, reset, enIN, dataAin_12, dataAin_24, dataC12_in, dataC24_in and hours 12_24. Again, hours12_24 chooses whether the 12 hour or the 24 hour counter will be displayed. Similar to the time mode, there is only one counter for the mins (one for the clock time and the other for the alarm time), but two separate counters for the hours, one for 12-hour system, and the other for the 24-hour system (modulo 12 and modulo 24), two for the clock time and two for the alarm time. Debouncers and edge detectors are used to handle the button presses. dataA_12 and dataA_24 are the inputs to the alarm counters for loading, modulo 12 and modulo 24 counters, respectively. dataC12_in and dataC24_in are are the inputs to the current time counters for loading, modulo 12 and modulo 24 counters, respectively. The outputs of the desired hour system is inputted to binaryToBCD whose output is inputted to the seven segment display. Ring counter enables one of the digits of the seven segment display which acts as the selection line of the multiplexor that chooses which digit to display from alarm hours digit1, alarm hours digit 2, alarm mins digit1, alarm mins digit 2, current hours digit 1, current hours digit 2, current mins digit 1, current mins digit 2 . The outputs from the module are the enables of the seven segment display, the 7 bits of the output of the 7 segments, dataA12_out is the 12-hour alarm time, dataA24_out is the 24-hour time of the alarm, dataC12_out is the 12-hour current time and dataC24_out is the 24-hour current time.



3) <u>Stopwatch Mode:</u>

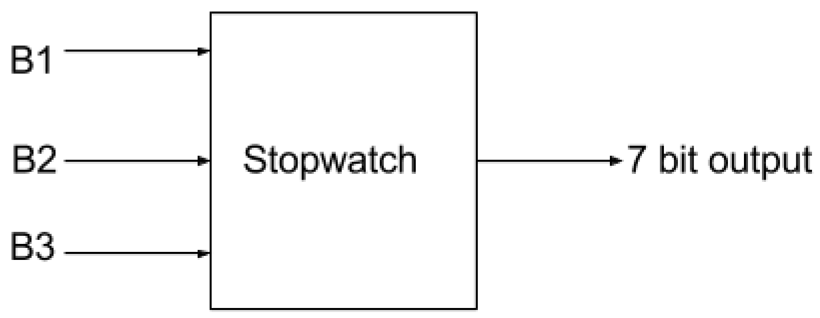


Figure #: BlackBox
for Stopwatch

The stopwatch module has five inputs: clk, RESET, rest, stop, enIN. There is an output 'y' which is the seven bit output of the seven segment display on the FPGA board. The stopwatch uses a clock divider to output two different clock frequencies. One is 100 Hz and the other on is 500 Hz. The 500 Hz clock is used for the fast display of minutes on the left two digits of the seven segment display. The two right digits are used for the hours display. A parameterized modulo counter module was created and instances of that module were made in the stopwatch module. The two right digits on the seven segments display had a modulo 10 counter that counted until 9 only and enabled the modulo 6 counter on its left to count when it reached the number 9. In the right two digits, when the modulo 6 counter reaches 5 and the modulo 10 counter reaches 9 at the same time both counters are set to zero and this enables the modulo 10 counter of the hours digit on the FPGA display to start counting.

The inputs rst and RESET, reset the module and set all the counters in it to zero. However RESET comes from the big module while rst is assigned to one of the push buttons (B3) of the FPGA board in the UCF file. There is also the input stop which is assigned to a push button (B2) on the FPGA board and when it is pressed the counters stop counting as their enable is turned to zero. When this button is pressed again, the counters get enabled again. Debouncers and edge detectors are used to handle the pressing of the buttons. Outputs of the counters are sent to the binaryToBCD whose output is sent to the seven segment module for display on the FPGA. Similar to the other modules, the ring counter and the multiplexer that activates one digit and choose its output, are used.

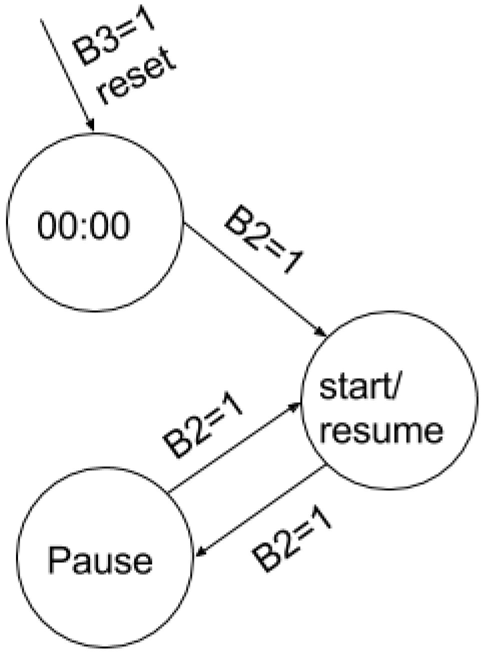
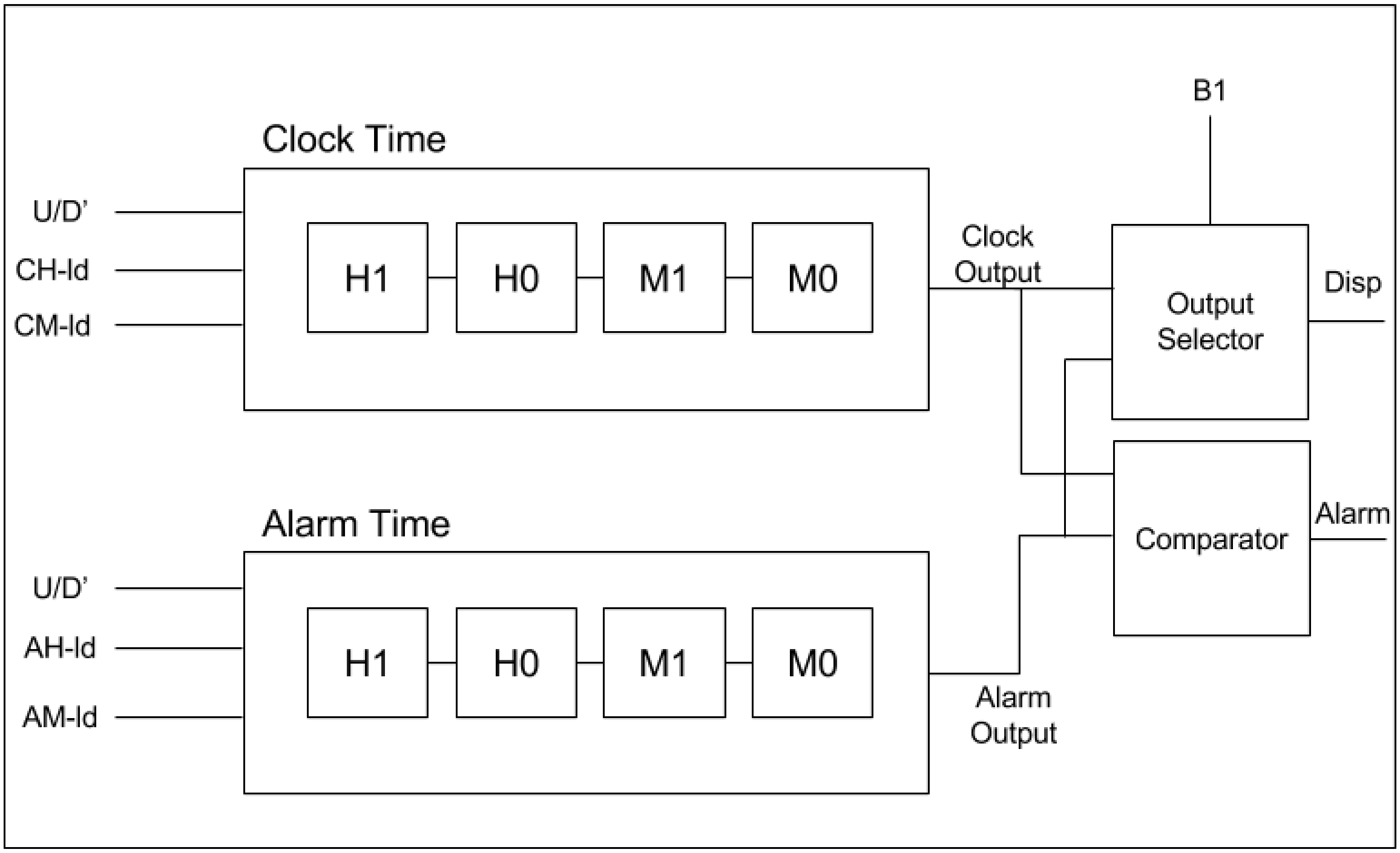


Figure #: State Diagram for Stopwatch

4) <u>Melody Module</u>

The melody module is responsible for playing the tune of the alarm sound. It gets 2 inputs (clk, en) and has 1 output (speaker). This module makes use of 2 other modules. MelodyROM module is used to store the values for the notes that are going to be played. Div_12 module is used to divide the value of the note by 12 to determine which note it is (e.g. A, B, C#...). The module has 2 down counters, one for the note frequency and one for the octave frequency. When both counters go to 0, the output signal (speaker) is inverted. I made a kind of legend that has values that correspond to notes, then I brought the notes of the song "Twinkle Twinkle Little Star" and translated them into the necessary values and I stored them into the MelodyROM module for future playback. However, the output wasn't as good as expected, but I suspect that this might be due to an error in calculating the required frequency or inputing the right frequency.

5) **Complete Digital Clock Module**



This module is the largest module that combines all other modules together. It has 7 inputs (clk, rst, hours12_24, B1, B2, B3, B4) and it has 4 outputs (DISP, rInv, ledOUT, speaker). This module calls all the other 4 modules and integrates them together. This module gets the 50 MHz clock directly from the FPGA, and there are two toggle switches that control the reset of the whole system and the hour format (12 or 24). There are also 4 inputs for the four push buttons on the FPGA which are the main controls for our Digital Clock. The module outputs DISP which is a 7 bit signal that controls what digit is displayed on the seven segment display, also rInv is a 4 bit signal that controls which digit on the seven segment display is currently enabled. The ledOUT signal controls when the LED flashes, and the speaker signal controls the sound output. Inside the module, first off, we have a register that carries the alarm time in both 12 and 24 hour modes, and the logic necessary to output its display on the FPGA (clock divider, binarytoBCD, ring counter, display mux). Also we have a procedural block that compares the Alarm Time and the Clock Time outputed from the Time module, to check if the alarm time has been reached or not, and it then controls a signal ALARM which enables the flashing of the LED and enables the Melody module. We also have the debouncers and edge detectors for the buttons used in this module, which are Button1 and Button2. We also have 2 counters that are incremented when B1 or B2 are pressed so that we can count the 2 seconds pressing time respectively. We also have a wire callen EN which carries the state the Digital Clock is in now, this is implemented in the form of a Finite State Machine, where we declared 3 Parameters: CLK, ADJ, STOP. Each parameter corresponds to a mode and the enables of the modules are toggled based on the value of the signal EN. Based on the value of EN, there is a signal called dispSEL which is a selection line for a multiplexer which selects which output to be currently displayed on the seven segment display

Contributions:

Habiba Gamal:

- StopWatch Mode
- Adjust Mode
- The second bonus (selectable option between 12 hours and 24 hours system)

Ali El-Said:

- Time Mode
- Complete Digital Clock Module
- Alarm Melody Sound (1st Bonus)

Noureldin Hashad:

- Adjust Mode
- Complete Digital Clock Module