CSA Coursework 2 Mini-Report:

Arduino Uno - Home Clock

Introduction:

For my second Computer Systems Architecture coursework, I have decided to create a home clock, using an Arduino Uno microcontroller, a LCD, and three different inputs. This is my prototype of a solution to a real-world problem. Knowing the date and time is essential to the daily life of every person, which makes tracking of time a real-world problem. The LCD on my home clock prototype shows the time(hours, minutes, and seconds) in a 12-hour format(AM/PM). It also shows the date(day, month, year) with a three-character representation of the month. Moreover, the LCD shows the room temperature, just like most of the standard home clocks do, using a TMP36 temperature sensor. The room temperature is presented in both celsius and fahrenheit. At last, the LCD also displays the edit mode, which tells the user of this prototype whether he can edit the time, date or temperature representation of the clock. I have used an input of two buttons which can manually change the time and date on the clock. I have also used two potentiometers - one for the contrast adjustment of the LCD, and one for switching the functions of the two buttons. This way, by using three unique inputs and an output, I have created a clock that does a perfect job of telling you the time, date or room temperature.

How it works:

As from the moment the clock is turned on, it starts to count seconds. The time on the LCD starts at 00:00:00 AM and it will start this way every time the clock is turned on. The start date on the LCD is 1st of January 2020 and the temperature is shown in celsius. The clock is always keeping track of time no matter if it is just standing or the user is pressing buttons to change something. There are four different types of changes that are done using the two buttons. The function of the buttons is determined by the position of the potentiometer. The type of edit that can be done at the chosen position is shown on the LCD, it says whether the user can edit time, date, year or temperature representation. In time edit, button 1 raises the minutes with +1, when 59 is reached the next value of the minutes is zero again. Every time the minutes are changed, the second counter restarts at 00, which I consider a standard practice used in most clocks. Button two changes time with +1. At the date position of the potentiometer, button 1 changes the days of the calendar and button 2 changes the months. At the year position, button 1 changes the year with +1, button two lowers it with -1. At the last position, button 1 displays the temperature in celsius while button 2 displays it in fahrenheit. Potentiometer 2 adjusts the contrast of the LCD. Besides the manual changes the user can make, the clock automatically changes hours, AM/PM, and date when it has to. It also updates the room temperature as soon as it changes. This was how this prototype works in detail.

Implementation:

There were two big challenges for me in the implementation of this prototype. One of them was the execution time of the loop function. I knew that it depends on many things and that

the execution time of every loop is not the same. This means that the use of delay method in the loop won't be accurate for measuring seconds. Finally I came out with a solution to this problem which uses millis function in order to make the execution time of the loop to always be exactly a second. This way the clock accurately measures the time, independent of the execution of the loop. The other big challenge for me was using the presentation of information on the LCD. Since I was restricted to use only two rows and a total of 36 characters and four things to show. After a bunch of different arrangements, I came out with a presentation of time and temperature on the first row and date and mode on the second. I have even needed to remove the ° character in the temperature because there was just no space for it. The only disadvantage of this implementation is that if the temperature is below -10℃ it won't have enough space. However, I think the arduino won't even work at a temperature this low. Moreover, I have implemented the button press using an interrupt function. This function has a switch statement which checks for the potentiometer value to determine what the button press should do. This implementation is very effective and allows the user to make a change at any time of the loop's execution, which is a huge advantage. The implementation of the potentiometer value check to happen at the beginning of every loop. There is con of this implementation that there is a bit of delay of the update if the potentiometer's position is changed in the middle of the loop. However, this implementation makes the execution of the interrupt method faster and the update of the potentiometer position more regular. I have implemented the temperature update the same way, at the beginning of every loop. This makes the update happen as often as it can. At the end, I have implemented the date and time change by a chain of condition statements. This way, the execution of the loop happens really fast since it does not check for hours and date change if minutes aren't changed. This way, I personally believe that I have reached the best possible implementation of this problem.

Appendix: