Project 2

Hannah Wilcox

Fall 2021 CSE 321

Table Of Contents

Introduction

This report serves to discuss work on the alarm system project in the class CSE 321, Real-Time and Embedded Operating Systems. It will cover the specifications of the project, features of the project, applications, various visualizations, a bill of materials, a testing plan, results of finished work, and how I could further improve my work.

The purpose of this project was to program a timer that would count from a maximum of 9 minutes and 59 seconds down to 0 minutes and 0 seconds. This was done using various hardware pieces such as a keypad and LCD display and allowed bare-metal software concepts in C++.

Specifications

The specifications of this project include the use of a 4x4 matrix keypad, solderless breadboard, LEDs, jumpers, an LCD display, and a NUCLEO L4R5ZI.

The keypad, breadboard, and LCD display are all hooked into the Nucleo via jumper cables. The program reads in inputs from the keypad by taking input from the column of the keypad that was pressed and knowing what row is being powered. For this to work properly each row of the keypad takes turns being powered, so that it's more obvious which button is being pressed since it narrows it down to only one option from each column.

Inputs from the columns of the keypad are read in by InterruptIn variables that activate unique ISR functions based on which column was interrupted.

The display on the LCD is controlled by multiple flags within the code, which tell whether or not to display a countdown, to pause the countdown, display a screen that prompts for user input, or show that the countdown has finished. These flags are controlled by user input on the keypad. If a user inputs A the timer is started, if the user inputs B the timer is stopped, and if the user inputs D the timer is stopped and the display prompts "Enter Time:" for user input. At this point, the user will input 3 numbers corresponding to minutes and seconds, and upon hitting A again the timer will start with this new time. The display also shows "Times Up" when the countdown reaches 0 minutes and 0 seconds.

Aside from the keypad and LCD display, multiple LEDs are wired into the Nucleo. One LED is turned on and off for every push on the keypad. The other two are to be turned

on when the countdown on the timer reaches 0. When the timer is started over again these two LEDs will turn back off.

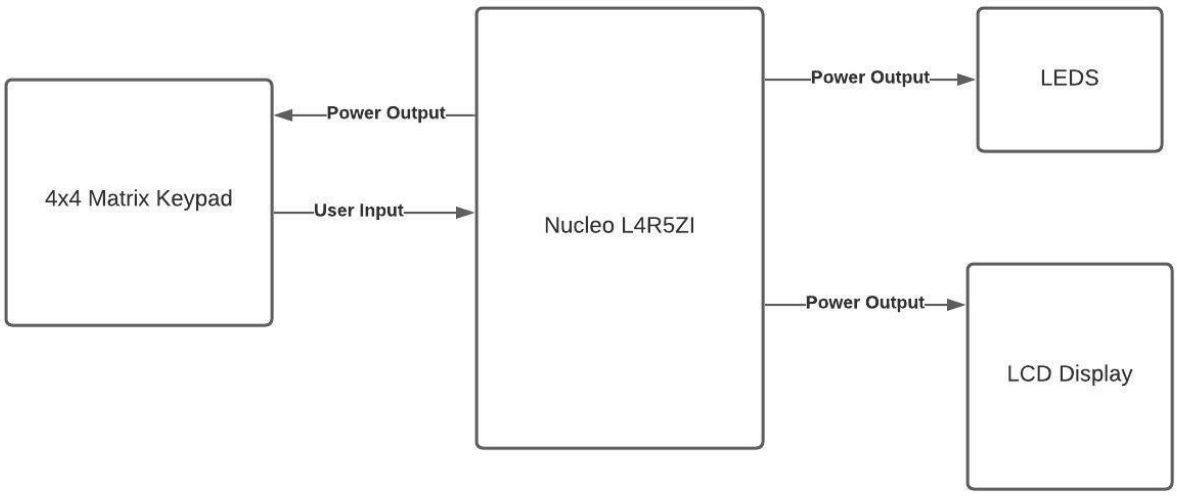
Features

- Timer that counts down from a maximum of 9 minutes 59 seconds to 0 minutes and 0 seconds
- Keypad input for user input
- Timer can be paused by user input
- Timer can be started by user input
- Timer count down time is set by user input
- LED output turned on by keypad press by user
- LED output turned on when timer is up
- LCD display output that shows remaining time

Applications

- Kitchen timer
- Interview timer - keep interviewees from going over time
- Presentation timer

Block Diagram

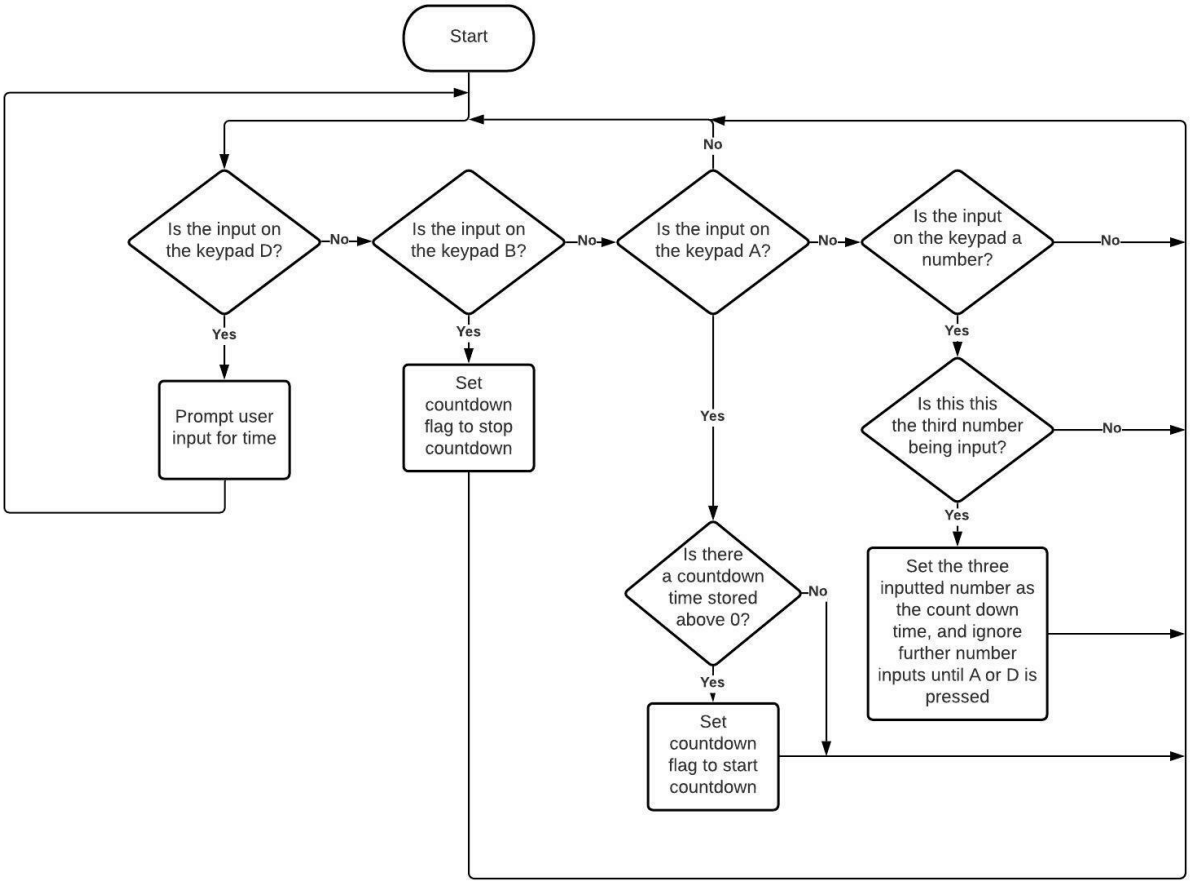


Functionality

The functionality of the countdown timer is to represent a timer counting down to 0 given a user input on a keypad.

- The user will press D to set a time for the LCD display to represent a countdown from.
    - The first 3 numbers that the user presses before pressing A to start the countdown will set the time the countdown starts at.
- The user can also press B to pause the countdown at any time, and A again to restart the countdown from where it stopped.

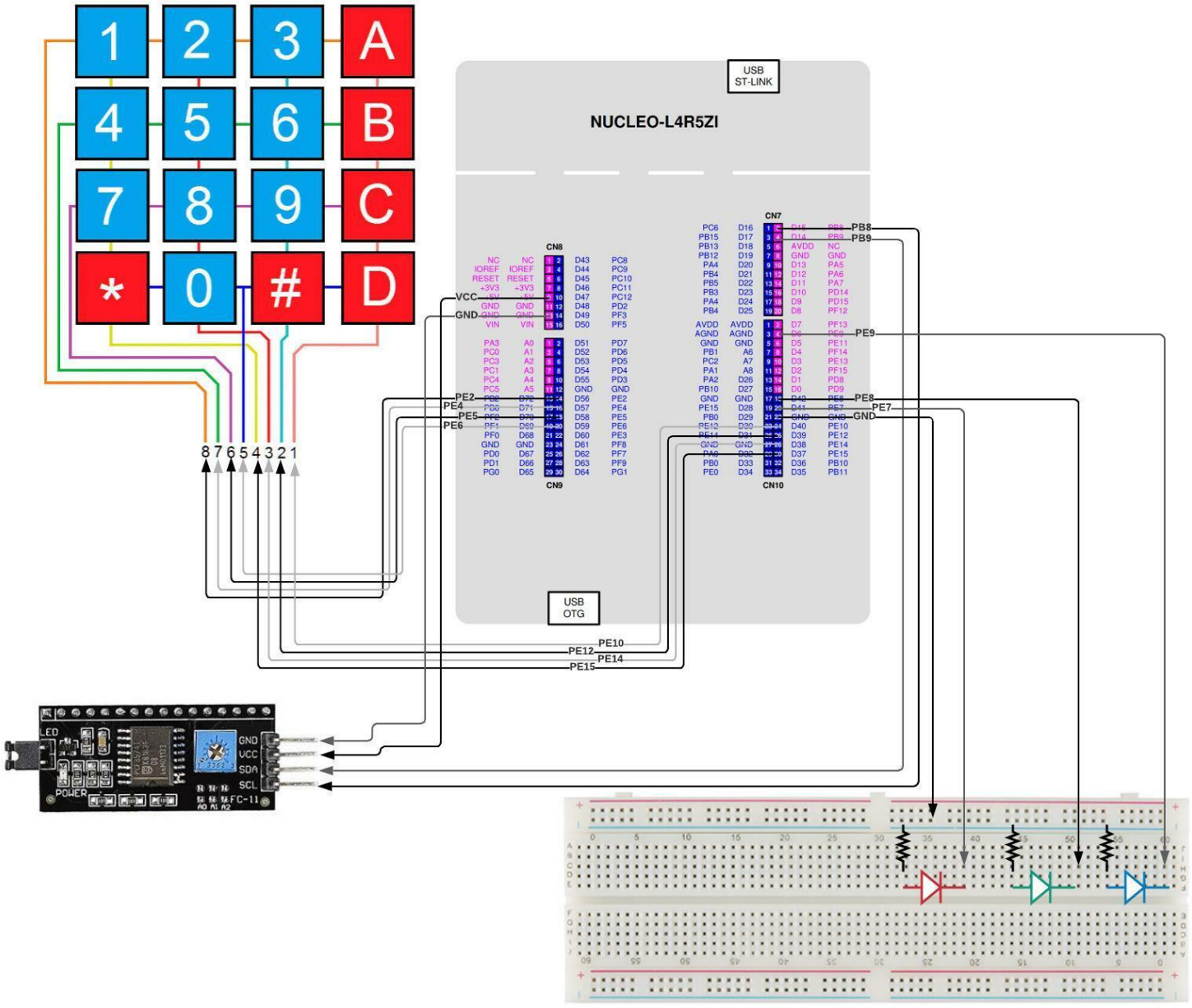
Diagram



BOM

- LCD
- Nucleo
- Keypad
- Solderless Breadboard
- Jumper wires
- LEDs

Schematic



Test Plan

For the test plan, a number of functionalities had to be tested:

- Keypad is properly detecting inputs from user
- LEDs properly light up on keypad press and countdown timer finishing
- LCD display is properly displaying the remaining time
- The time is counting down in the correct way
    - (not going into negative time and properly changing the minute and setting seconds back to 60 when the seconds go to 0)

The keypad functionality can be tested with properly implemented print statements on each press. If the correct keypad button isn't being printed, it would be a good idea to check that the wires are connected in the right ports, and then ensure that the internal powered row variable is updating correctly.

LED functionality can be tested by first seeing that the LED will turn on in the program regardless of a button press, and then moving it to the ISR functions for button press and seeing that they still light up.

The LCD display can first be tested by having some random statement being printed out to the screen and seeing that it works. This will make sure that the wires are correctly hooked into the Nucleo. Next, one can test changing the display by having it change every other run in main() to ensure that it is being properly cleared and rewritten each time. Once these two objectives work, it can be programmed with the correct functionality of the problem however this will require that the keypad has already been tested and works properly, as the display will change and update based on the keypad inputs.

Time can be tested with print statements outside of the LCD display, seeing that the seconds go down one by one and if the seconds reach 0 and the minutes are greater than 0 that the minute goes down by one and the seconds restart at 59. Tests also need to be made to ensure that the seconds and minutes don't go into the negative, meaning that they have to stop updating when both values hit 0. Once this functionality works, having a function to update time that is attached to a Ticker variable that updates every second will ensure that time is being updated every second instead of every pass of the main function.

Results

In the end, all parts of the project worked properly. The countdown timer had no issues going into negative time or improperly converting down at the end of a minute. The display properly updated with the new remaining time every second and was able to be stopped and started again by the B and A keys on the keypad. The display also showed a “Times Up” screen when the countdown time reached its end and was properly reset upon a new time being input via the user. An LED properly turned on for each keypad push, and two other LEDs properly turned on when the timer was finished. The LCD display never showed any incorrect time or screens for each situation of the project.

Recommendations for Improvement

The final code had some repetition in it that would be better optimized if it was condensed a bit more. For example, there are two conditional statements in the while loop of the main function that is meant to update the LCD display with the count downtime, but one is only accessed when the user first inputs a time. The code in both conditionals is virtually the same, so they could just be condensed into one conditional statement to reduce confusion for anyone who may want to interpret the code.

Aside from that point, I believe there is no need for further improvements as the code works well for the project and has good variable names and comments.