EECS 2021 Major Project
Demetrios Koudoumnakis
217987595

Introduction:

I created a timer and stopwatch using the Atmega328P chip on the Arduino Grove board. There are two modes, timer and stopwatch, the user can toggle between the two modes with a button and can press another button to start the mode. The timer starts from a specific time set by the user using a potentiometer, and counts down until the timer reaches zero or when the user manually stops it. The stopwatch stars at zero and counts up until the user stops it by pressing a button or when it reaches 99 seconds. Once the timer or stopwatch is stopped the user can then resume it or switch to the other mode. There is a 7 segment display to show the current mode, and a 2 digit 7 segment display to display the count.

Context:

This project uses MPLABx to program the Atmega328P chip which is the microcontroller that controls the entire project and all of the components. To program the microcontroller with MPLABx we needed a SNAP programmer. To use this I had to sever a wire on the groveboard. See Figure7. The chip is on the Arduino Grove board where it is connected to a button and a potentiometer that I use in this project. I added some external hardware to make this project whole. I added a breadboard with a single and double 7 segment display, a button, copper wires, and a NAND logic gate (7400). This project provides users a way to gather information about events. They can see how long it takes for a certain event to happen, as well as how long an event takes to happen. This project takes a deep dive into how a microcontroller operates with IO ports and registers.

Technical Requirements:

The single segment display on the breadboard will display the mode, 1 for timer and 2 for stopwatch. There is a button on the breadboard that acts as an input for B3 on the chip. This button is used to toggle through the two modes. The button embedded on the grove board attached to D6 on the chip is used to select the desired mode. When the timer is selected the user will use the potentiometer embedded on the grove board attached to port C pin 0, to select the starting time from 0 to 99 displayed on the 2 digit 7 segment display on the breadboard. Once the user is satisfied with the starting time they can press the button on the grove board to start the countdown. The same button can be used to pause and resume the timer. The timer stops either when the timer reaches 0, or when the user pauses the timer and then presses the button on the breadboard. The stopwatch starts at 0 and counts up until 99. The user is able to pause and resume the stopwatch. The stopwatch finishes once the user stops it manually by pausing it and then pressing the button on the breadboard, identical to the timer.

The software component of this project uses the IDE known as MPLABx. The program is written in C and assembler code. The program starts by declaring variables and

methods. The methods are timer, stopwatch, setTimer and delay. The variables include button readers, counters, and array indexes. The main method begins by assigning the IO ports on the Atmega328P chip. The mode is initially set to timer(1), when the button on the breadboard is pressed a “count” variable is incremented which corresponds to toggling the mode. There is a method that corresponds to each mode. Each method starts off with a while loop that waits until either button is pressed and an if statement to separate the actions of the two buttons. When the button on the breadboard is pressed we exit the method and switch mode. When the button on the grovebord is pressed the feature begins. If it is the timer, it starts off by jumping to another method where we use the potentiometer to set the timer, by pressing the button again we leave that method and start the countdown. The stopwatch works the same except it does not set the timer value, it starts at 0 and counts up. To program the 2 digit display I made an array where each index contains the corresponding pattern to display a number. I move through the array based on a “counter” variable and a mathematical calculation. The counter variable increments by 1 each second which will change the value on the display. I increment the “counter” with an inline assembler code. I did this to learn more about the interaction between “c” and “assembler”. The potentiometer is used to set the timer and display the change live on the 2 digit display. The program reads a value from 0 to 1023 depending on the position of the potentiometer. By using a mathematical expression I can express the value from the potentiometer to correspond with “counter”. See Figure1 for the flowchart of the code. Throughout the program I needed a buffer to read input ports clearly. This is why I created the delay method. In this method I use the Atmega328P very own timer, Timer0. This is a 8 bit timer that will count up from zero to a set value with the speed of the microcontroller's clock. Everytime it reaches this value the clock will reset. This process will happen as many times as needed when the method is called.

The hardware component of this project uses the grove board with its potentiometer and button. A breadboard contains a single and a double digit 7 segment display, a button and a NAND logic gate. The complexity of the hardware comes from wiring the 7 segment displays. The single digit has 7 LED’s that need to be turned on in order to form numbers from 0 to 9. We need to attach each of these LED’s to a pin on the chip in order to control it with MPLABx, see Figure2. In order to save Atmega328P pins I only used 2 pins to control the display. I did this by using a NAND logic gate, wiring and the fact that I only needed to display “1” and “2”. This is better described in Figure3 below. I attached a button on the breadboard to an input pin on the Atmega328P, this button is used to toggle between modes. The timer is displayed on the 2 digit 7 segment display, which works the same as the single digit but with common LED pins and two different grounds that would turn on each digit.See Figure5. To display a different number on each digit I had to turn on one digit and its LED’s then turn it off, and turn on the other digit and its LED’s. I programmed this in MPLABx, this process happens so fast that it is unnoticeable. I had enough pins on the Atmega328P to attach each pin on the 2 digit display. Pin connections can be seen in Figure4. See Figure6 for hardware blueprint

Figure 1

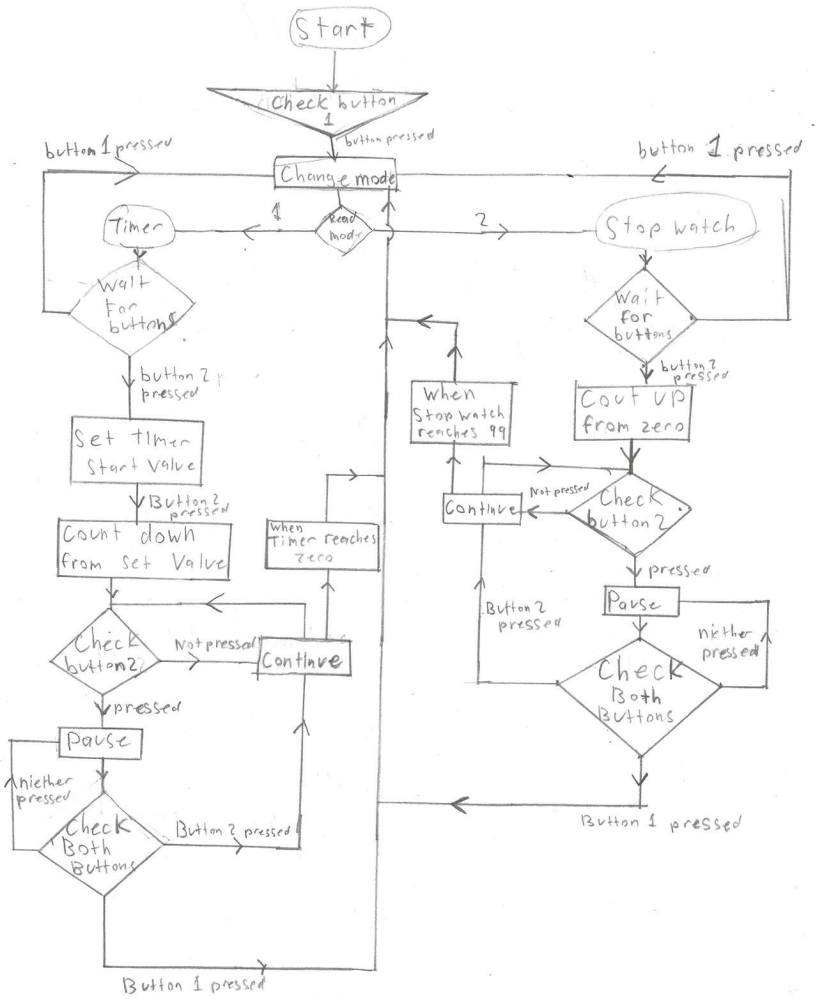


Figure 2

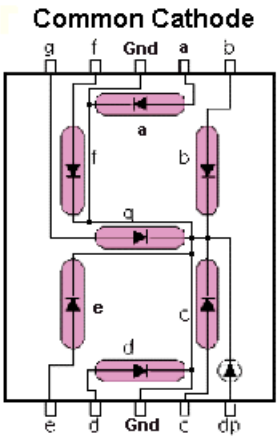
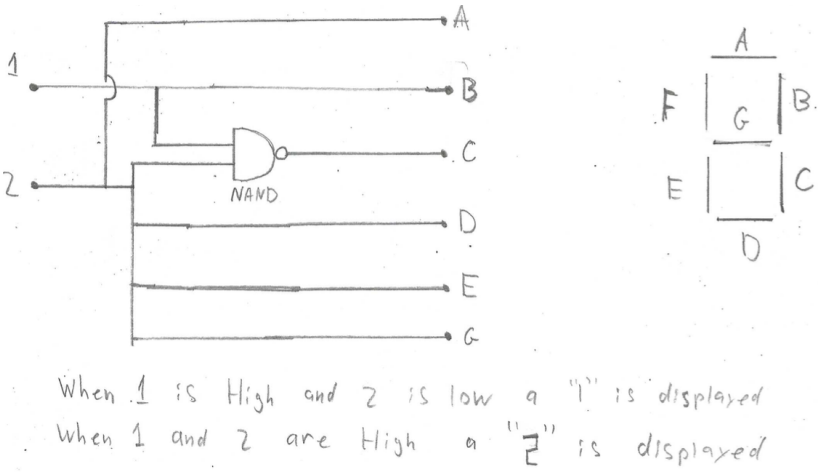


Figure 3



When 1 is High and 2 is low a "1" is displayed
When 1 and 2 are High a "2" is displayed

Figure 4

Input:
Button on PD6
Button on B3
7-segment display:
PB5 = 2
PB4 = 3
2 digit 7-segment display:
PD0 = a
PD1 = b
PD2 = c
PD3 = d
PD4 = e
PD5 = f
PD7 = g
PB0 = D1
PB1 = D2

Figure 5

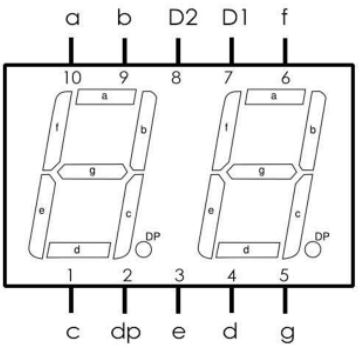


Figure 6

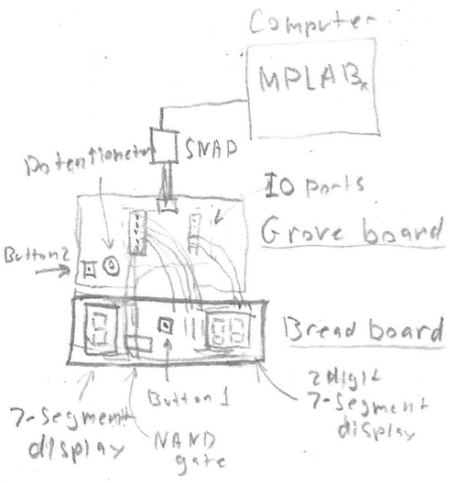
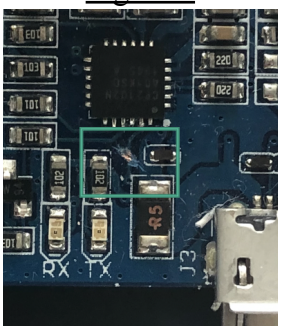


Figure 7



Component List:

- Arduino Grove Board
  - Atmega328P
  - Potentiometer A0
  - Button D6
- SNAP programmer
  - cables
- BreadBoard
  - Single digit 7 segment display
  - 2 digit 7 segment display
  - Button
  - Copper Wires
  - NAND gate (7400 IC)
- MPLABX

Procedure:

This project was challenging and required me to conduct a lot of tests and research to fully understand the components before I could combine them into one project. I had a breadboard with a button, wires, resistors and logic gates from a previous high school course. I knew I wanted to use a 7 segment display, so I ordered one from amazon and started to wire it up to the Atmega328P pins. I then started writing my program with an outline of what I

have imagined. I then figured out how to receive a button input with the button on the grove board with the help of Professor James Smith. He helped me realize that I had to debounce the button in order to receive a proper input. He also informed me about the 4 and 2 digit segment displays. After I attached the button on the breadboard to the chip I realized that if I wanted to add another 4 or 2 segment display I would run out of pins on the chip. With my previous knowledge of wiring logic gates and the fact that I only wanted to display 2 digits on the single digit display I found a way to reduce the amount of pins used from 7 to 2. I then ordered a 4 digit 7 segment display, which took a long time to arrive, and immediately burnt one of the pins. With time ticking I could only order a 2 digit segment display with a backup. anodeOnce it arrived I wired it on the breadboard and connected it to the pins on the Atmega328P. I then figured out how it works with extensive tests and integrated it with my program. After hours of testing, I finished my program. Last but not least I added the potentiometer to my project. I conducted intense research on how to get it to work and was able to add it in my code. After some last minute adjustments and tests, I have finally finished the project.

Test:

This project required a lot of experimenting as I was challenging myself to go above my knowledge and the materials of the course. For every new addition to the project I conducted tests before I integrated them so I could understand how they work. To incorporate the new additions to the project I used a trial and error method of implementation. I use the debugger function in MPLAB to step through my program to make sure everything is working the way I intend it to. I also used the disassembly view to debug my code. I conducted a great deal of tests to get this project to work to my satisfaction. I also had my father and sister test out my timer and stopwatch to test its user friendliness. I had to make many adjustments from those tests. They helped me realize errors that I did not account for. Testing was a huge part of this project, I conducted tests to understand the components and get the project working perfectly.

Contingency:

The final product was practically how I envisioned it. The only thing that I could not accomplish was to use a 4 digit display. This would have allowed my timer to not only count in seconds but also minutes. It could've been able to count from 0 to 99:59 minutes. Instead I had to use a 2 digit display that has a range of 0 to 99 seconds. After weeks and expensive shipping costs I obtained a 4 digit display. However I ordered an anode display rather than a cathode, which caused another learning curve. Then I accidentally burnt one of the segments during testing. I then decided that it was not worth trying to get another one and ordered a 2 digit display from amazon that arrived the next day. The project works the same as it was intended but with a 2 digit display instead of a 4 digit display. In the future I need to do more research in what I am ordering and order backups in case of mistakes. The timer itself increments based on a for loop rather than an actual timer. With more time and help I would be able to use an Interrupt service routine to change the time on the 2 digit display. In my future projects I will spend more time making my program efficient and effective.

Additionally:

This course has taught me so many different aspects of communicating to a microcontroller on a smaller scale. We have used both C and assembler to communicate to the Atmega328P on the groveboard. During this course we have accessed certain registers and ports on the chip, we have also spent some time on programming a CPU with logic gates. I have included all of these features into my project and so much more. In the course we only learned about using digital output ports on the chip. For my project I used both digital inputs, outputs and analog inputs. This required me to research and experiment far beyond what was covered in class. I also used an inline assembler to increment and decrement a counter. By using the inline assembler I can directly access a register and directly control what happens to the value and memory placement. Through tests I used the disassembler view to follow the commands to make sure the compiler was working the way I intended it to. I used a physical logic gate in the hardware portion of this project. I believe that this project speaks to my knowledge and understanding of this course. I advanced the material of this course and integrated my engineering skills with previous knowledge of breadboards and wiring.

Conclusion:

This project was challenging and has increased my hardware and software skills. My system has two modes that can be used by anyone for anything. To set a countdown timer and time how long it takes to do just about anything. My project uses the hardware and software material covered in class and expanded on it. Completing this project has helped me understand how microcontrollers work on the ground level. I am excited to learn and work more with microcontrollers in the near future. This project has only provided me with the foundation for so many future projects.

Sources:

- "ADC on Atmega328. part 1," *Embedds*, 19-Dec-2020. [Online]. Available: https://embedds.com/adc-on-atmega328-part-1/. [Accessed: 07-Dec-2021].

- M. Francis, B. I. L. A. L. Malik, M. I. N., Zeka, A. Degago, M. A. Khalid, A. DS, Ebeso-Tech, Aliu, Jamiu, Jhamiu, W. Kn, Linshahril, Hamid, and Eugene, "7 segment display interfacing with Pic Microcontroller," *Microcontrollers Lab*, 12-Apr-2020. [Online]. Available: https://microcontrollerslab.com/7-segment-display-interfacing-with-pic-microcontroller/. [Accessed: 07-Dec-2021].