# *Temperature monitor*

Nduvho E. Ramashia (1490804)
School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg 2050, South Africa

***Abstract*—This document outlines the decisions and steps taken to implement a temperature monitor. The monitor is based on an Atmega328P microchip. The temperature is displayed on a total of 3 Seven segment displays. They all share 7 input pins but the ground. That saves the pins and makes it possible to leave some pins on the microchip for future improvements. Timers are used to save power by doing conversions on defined intervals instead of constantly. Two timers are used to provide more control.**

## I. INTRODUCTION

This document provides a detailed approach taken in implementing a temperature monitor. The temperature is read from the environment. It is measured with an MCP9700-series temperature sensor. At the heart of the monitor is the Atmega328 microcontroller, programmed in assembly. The microcontroller is programmed to display the environment's temperature on 3 seven segment displays. The overall implementation uses only two IC's, a microcontroller, and a transducer.

The first section covers the design description of the monitor. It provides a descriptive approach taken the fully implement the monitor. The section after it shows the analysis of the designed monitor. The last section contains the overall summary of the implemented monitor.

## II. IMPLEMENTATION

### A. Design

The following components were available for the implementation of the monitor:

- Atmega328 microcontroller
- MCP9700-series temperature sensor
- 3 x seven-segment display
- 2 x 2N2222 transistors
- 21 x 470 ohms resistors
- 2 x push buttons
- 5 LEDs

The temperature is measured using the provided transducer. The provided transducer has a voltage range of 0V – 1.75V [1]. Because the ADC's internal voltage is less than the maximum voltage the transducer can create, a 5V ADC reference voltage is used. This is done to make sure that the maximum temperature can be read. The transducer has a supply voltage range of 2.3V – 5.5V. To ensure the stable functionality of the transducer, a 5V supply voltage is used.

The data output from the ADC depends on the ADC's reference voltage, the resolution of the ADC, and input voltage into the ADC [2]. It is calculated using (1) [2]. The step size of the ADC is obtained using (2) [2].

$$D_{out} = \frac{V_{in}}{step\ size} \quad (1)$$

$$Step\ size = \frac{V_{ref}}{resolution} \quad (2)$$

Where:

$V_{in}$ = Analog input into ADC

$V_{ref}$ = ADC reference voltage

Resolution = ADC resolution

The input into the ADC, the transducer's output, is obtained from (3) [1].

$$V_{out} = T_c * T_A + V_{0^oC} \quad (3)$$

Where:

$T_c$ = Ambient Temperature

$T_A$ = Temperature Coefficient

$V_{0^oC}$ = Transducer Output Voltage at 0°C

After manipulating (1) and (3), (4) was obtained. It is used to extract the temperature of the environment from the ADC output data.

$$T_A = \frac{D_{out} * StepSize - V_{0^oC}}{T_c} \quad (4)$$

Table 1 shows the values of the independent variables of (1)-(4).

TABLE I. VALUES USED TO OBTAIN THE TEMPERATURES FROM ADC DATA

| ***Variable*** | ***Value*** |
|---|---|
| $V_{ref}$ | 5V |
| Resolution | 1024 |
| Step Size | 4.88mV/step |
| $T_c$ | 10mV/°C [1] |
| $V_{0^oC}$ | 500mV [1] |

After inserting the values in table 1 into (4) and simplifying, (5) is obtained.

$$T_A = \frac{D_{out}*125}{256} - 50 \qquad (5)$$

Equation 5 is used to determine the exact temperature value from the ADC output. The determined value is then displayed on the 3 Seven segment display.

*B. Timers Design*

Two timers were used to monitor different events. Two were used to provide the flexibility of having two independent timers. This was done to make provide more control since each timer comes with its set of controls.

One of the timers controls how long apart the ADC converts the temperature. The other one controls how long each Seven segment display stays ON. The use of separate timers for converting and keeping the displays ON enables the use of more than 2 displays. It makes it easier to delay more displays without affecting the conversion delay.

The use of separate timers for conversion delay and display delay also makes it easier to change any one of them without affecting the other. That makes it easier to control how frequently the data is converted. This makes it flexible for different users/systems. Some users might want a faster conversion rate than others depending on the reason for monitoring the temperature. The separate timers also allow the control of how long each display stays ON. This makes it easier to control how long the displays are kept on depending on how the displays look when connected. This allows the user to minimize the power consumed by the displays depending on what display rate is enough. Fig. 1 shows the execution timing of the two timers.

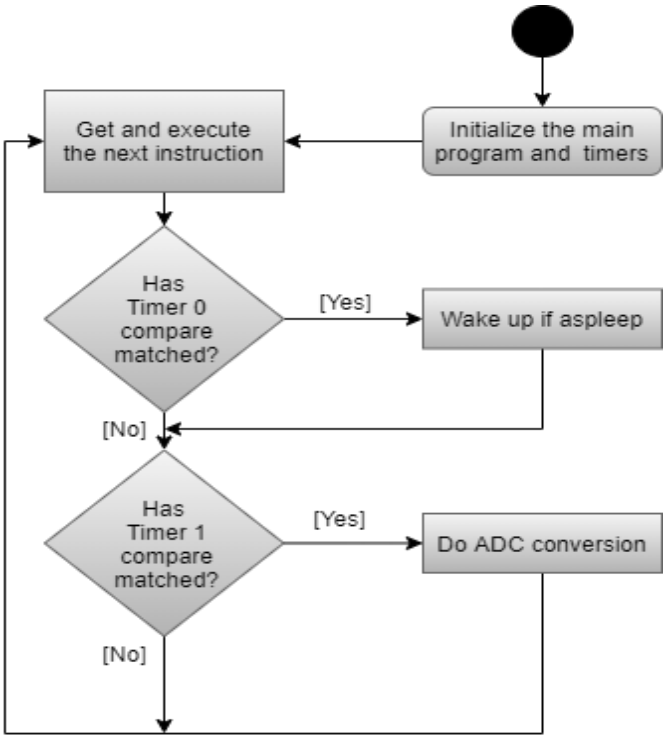


Figure 1: Timers execution timing

*C. Main Design*

The 3 seven segment displays each received the signal. The signal stays high for a period depending mainly on timer 0 compare-value. To cause a delay, the microcontroller is sent into idle mode. This method of delay is chosen because of its power-saving advantages. That makes the monitor suitable for low power systems. Fig. 2 shows the main program execution along with the timers' execution, which was shown in fig. 1.

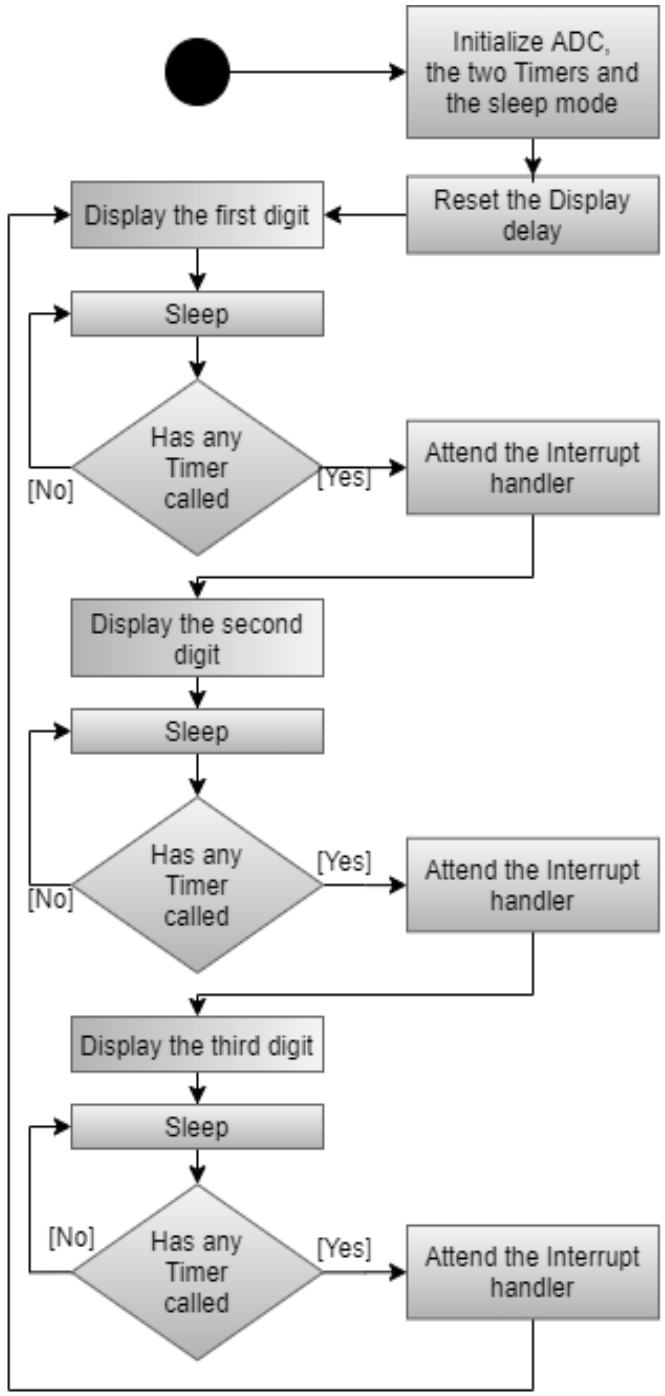


Fig. 2: Main program execution timing

### D. Circuit Design

Fig. 3 below shows the complete circuit connection. A total of 10 I/O pins were used to display the temperature on the 3 Seven segment display. Three of them are used for selecting a display depending on the digit than needs to be displayed. They are used along with 2N2222 transistors as switches. This is done so to save some pins for future use of the chip. They are also saved for future modifications. The three displays are each ON for a specific period, after which the next in line gets power. They each share 7 pins, the pins that determine what number or sign is displayed.

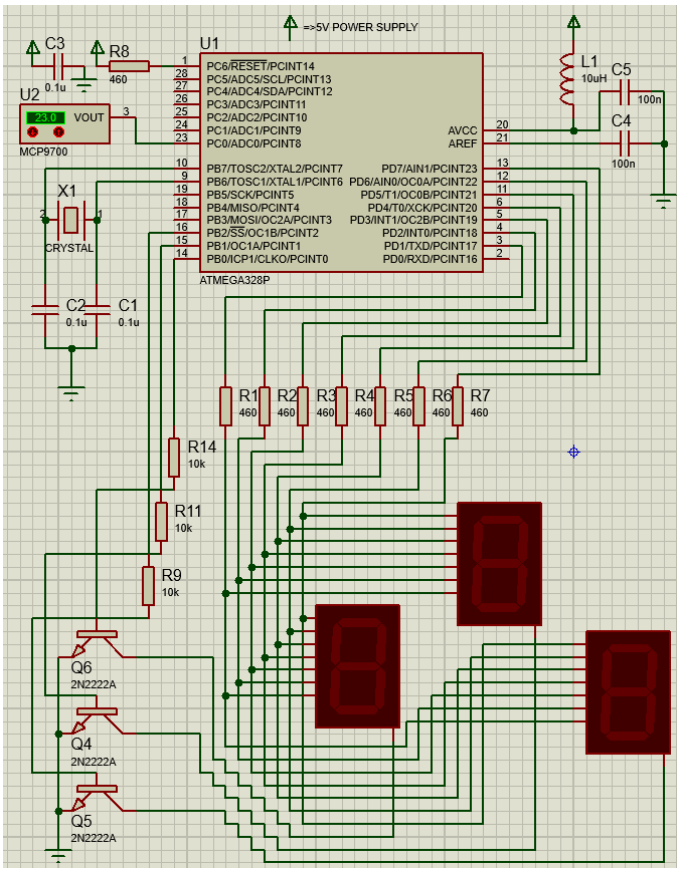


Fig. 3: Atmega328p based Temperature monitor circuit diagram.

## III. SIMULATION

Due to the unavailability of a breadboard, connecting wires, and a transistor, the program was tested only on a simulation program. The simulation program Proteus was used to simulate the connections. A HEX file is created form Atmel Studio and then loaded into the microcontroller in Proteus.

### A. Results

The code was run on a circuit connected as in fig. 3. It was also run on a circuit where the bare microcontroller was replaced with an Arduino Uno 3 board.

Both the two circuits display the same temperature. The circuit built from scratch, however, has a longer conversion delay compared to the one run on the Arduino board, which has the expected delay. This is due to the noise and inaccuracies in the circuit built from scratch.

The difference in the timing of the two circuits does not affect the displayed final output. The unintentionally reduced temperature refresh rate does however save power.

Due to the limitation of the Laptop used to run the simulation, the program used could not run the program and display the temperature at a fast enough rate to make the Seven segment displays seem all ON at the same time. As a result, the program was only tested for relatively longer delays. That was to make sure that the program can run the simulation and produce the expected temperature on the display, even at the cost of the digits not seeming like they were all sent at the same time.

### *B. Future Improvements*

For future improvements, the monitor can have its conversion rate vary depending on what a user from the environment set it. The user can control this using a potentiometer or a push button to increment the delay and reset it to a minimum reasonable value after a certain number of increments. The monitor can also be implemented to have a wider range of outputs.
The display delays can too be controlled by a user or system depending on different conditions on the environment.
The input voltage can be lowered to something less than 5V but more than 1.75V to supply on what is needed by the ADC. The monitor can have different LEDs to indicate different ranges of temperatures.

## IV. CRITICAL REFLECTION

During the time spent doing this project, the author became more familiar with the Atmega328P timers and interrupts. The experience of doing this project was to an extent challenging but was fun and educational as well. This was because of the new things the author had to learn to complete the project and improve it.

The project's basic working went well. It does however still have room for more improvements. The same goes for time management. The author can manage time better. A detailed time plan could be used to manage time better. It can also be used to divide sub-tasks of the project to make it easier to tackle instead of looking at it as one whole thing.

The change in the working environment also had a negative effect on the progress of the project. There was a time when the author's progress come to stop. That was when the author had no Laptop to work on, which made it hard to learn new things there was no laptop to prove or disprove the author's work. There was not much that could have been done about that except for the author to keep on working on multiple solutions before getting a laptop for debugging.

## V. CONCLUSION

The temperature-monitor designed works best when connected to a microcontroller via the Arduino board. That eliminates the noise and inaccuracies involved in building a building the circuit for the microcontroller to work with. It is more precise. The delays between the ADC temperature conversion and the time the Seven segment displays are ON are different and independent. This provides more control over the two.

In the future, the two delays can depend on a push-button or a potentiometer, or any other input from the environment. This gives users more control and an opportunity to utilize the microchip as they see fit. It gives a user more options when using the monitor.

## REFERENCES


[1] Microchip, "Low-Power Linear Active Thermistor ICs", Microchip Technology Incorporated, Chandler, pp.1-8, 2015

[2] M. E. Mazidi, S. Naimi, S. Naimi, "The avr microcontroller and embedded systems using assembly and C", Prentice Hall, New Jersey, pp.464-468, 2011


I Nduvho Edward Ramashia, Wits Student number 1490804 hereby declared that this, i.e. the code, the hex file, and the above report, is my work.