Lab 2: Basic I/O and ADC-based Readout of Temperature and Voltage

Kaicheng Wu
Department of Electrical and Computer Engineering
McGill University
Montreal, Canada
kaicheng.wu@mail.mcgill.ca
260892789

Tian Feng
Department of Electrical and Computer Engineering
McGill University
Montreal, Canada
tian.feng@mail.mcgill.ca
260913386

***Abstract*—In this lab, we implemented a program that allows us to alternate between reading the temperature and voltage of the STM32 board.**
***Index Terms*—GPIO, Core Temperature, Voltage, STM32**

I. Introduction

The General Purpose Input/Output (GPIO) pins are interfaces on controllers that could be programmed to perform various functions, such as digital bit input reading, digital pin output, and analog quantity reading/writing.

For this experiment, we will implement a program that monitors the core temperature and voltage of the STM32L4 processor. Moreover, we will implement a simple control mechanism to toggle between the readouts.

II. Implementation

A. Button Input

PA5 is set for the output of the LED and PC13 is set for the input of the push button with pull-up mode. Pull-up mode is more stable and robust through our tests. The push button toggles the LED–a push and release will change the on/off of the LED. We achieved this by implementing a 2-state-FSM, swapping its status between ”waiting for push” and ”waiting for release” status. The off-LED indicates the board is polling for temperature while the on-LED indicates the voltage.

B. ADC1 for Temperature

ADC1 is selected for the tempsensor channel when the LED light is off. The channel is configured to 640 cycles(about 13 us). According to the reference manual, the tempsensor needs 5-10us to perform the conversion, at 48 Mhz, which is about 500 clock cycles. A minimum redundancy is given to ensure the system is robust.

C. ADC1 for Voltage

This is implemented by simply multiplying the base voltage(3.0V) by the factor from the ADC reading and the calibration value from the memory. When the button sets the LED on, the converted voltage readings will be put in the ADCVal variable with polling. The channel setting is set to 640 cycles(about 13us) to maintain consistency with the temperature sensor.

D. Programmable Reconfiguration of ADC1

We implemented a channel select function and a uartPrintf function for debugging. Those functions ensure the readability of the project.

III. Results

We conducted an experiment where we applied heat to the processor using a hair fan for a duration of 15 seconds and used natural convection to cool it off, observing an increase and decrease in temperature as indicated by the readout. This procedure was repeated five times, and the average temperature change across the five trials was recorded.

TABLE I
Temperature measured on the core of processor after heating

	Room temperature	Core temperature pre-heating	Core temperature post-heating
Temperature (°C)	23.0	29.7	34.8

Indeed, the results we observed match our expectations.

IV. Constraints and Limits

A. Computation Speed

We find that in the debug mode the math calculation of the output temperature and voltage is slow. Maybe CMSIS could be used for the calculation to improve the performance.

B. Usage of DMA and Interrupts

We at first considered using DMA to bypass the processor and enable interrupts instead of polling. However, it was not mentioned in the lab documents so we chose the most common implementation. The polling implementation without DMA is robust and feasible under a single-task project.

V. Conclusion

To conclude, we implemented a program that lets us switch between updating the readout of the temperature and voltage each time a button is pressed. In this lab, we learned about the GPIOs and their usage, as well as the usage of the Analog to Digital Converter (ADC).

References

[1] RM0432 Reference manual STM32L4+ Series advanced Arm®-based 32-bit MCUs.(n.d.).
Retrieved from https://www.st.com/resource/en/reference_manual/rm0432-stm32l4-series-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

[2] STM32L4S5xx STM32L4S7xx STM32L4S9xx Ultra-low-power Arm Cortex-M4 32-bit MCU+FPU, 150DMIPS, up to 2MB Flash, 640KB SRAM, LCD-TFT & MIPI DSI, AES+HASH. (n.d.).
Retrieved from https://www.farnell.com/datasheets/2602792.pdf