Politecnico di Milano - Department of Electronics, Information and Bioengineering

POLITECNICO
MILANO 1863

AOS Microcontroller Project

[Light Sensor & ADC Interfacing]

Student	Christian Müller & Arno Virtanen
ID	894942 & 894986
Course	Embedded Systems
Academic Year	2017-2018
Advisor	Federico Terraneo
Professor	William Fornaciari

January 1, 2018

Contents

1 Introduction

The goal of this project work was to implement a peripheral driver for the STM32f407 microcontroller [**?**]. The implemented driver enables the use of the analog-to-digital converter (ADC). The microcontroller which contains the analog-to-digital converter is located on a development board (STM32 F4 Discovery kit) shown in figure 1. This development board contains components that can be interfaced with the microcontroller.

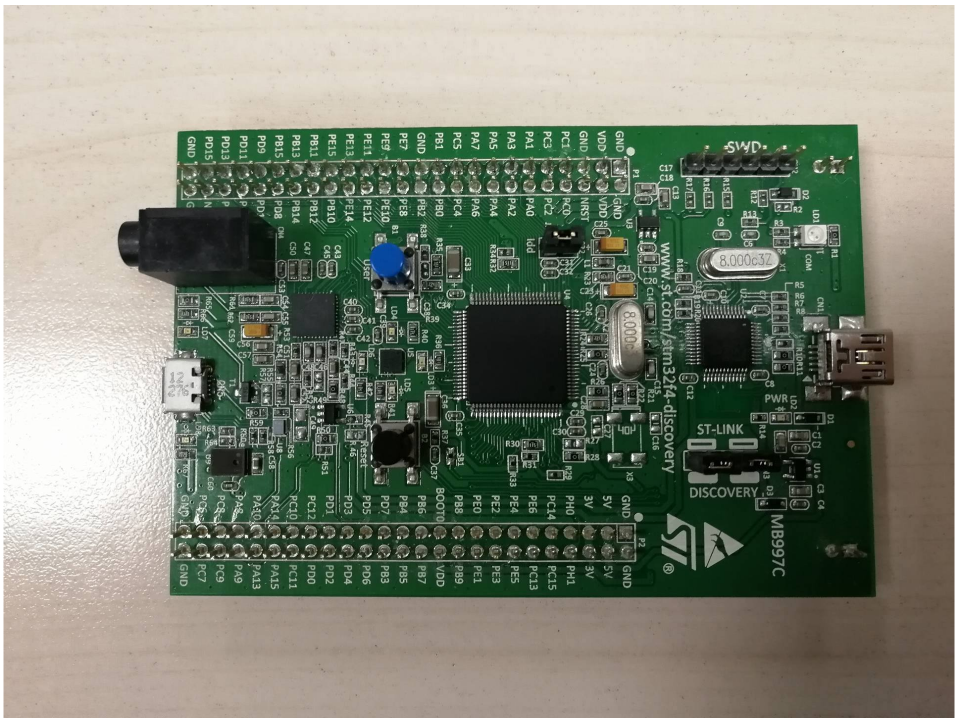


Figure 1: Discovery Board

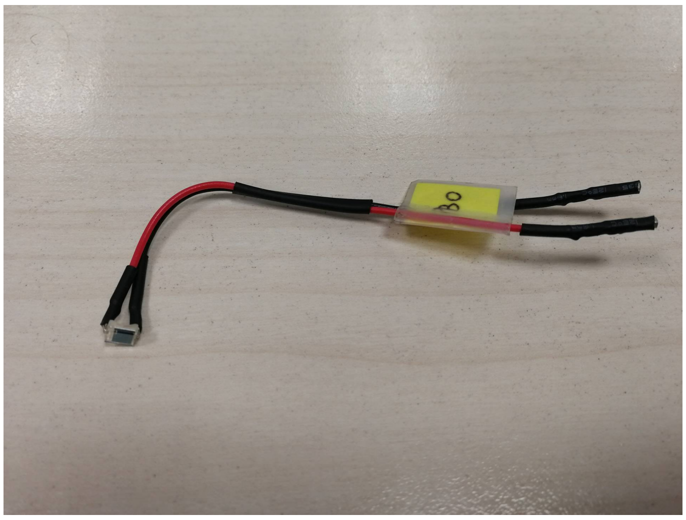


Figure 2: Photodiode

In this work the push button and user configurable LEDs of the development board were utilized as well as a external photodiode shown in figure 2. This photodiode was connected to the development boards external headers that enable connecting devices to the microcontroller. The photodiode and the components mentioned above were used to make a simple game that measures the users reaction time.

2 Design and implementation

2.1 Game logic

The idea of the implemented game was to test the reaction time of a player. This game can be played by a defined number of players. The reaction time of a player is evaluated by indicating a start signal and after this signal the player has to cover the photodiode as fast as possible to indicate a stop signal. The game logic is illustrated in figure 3.

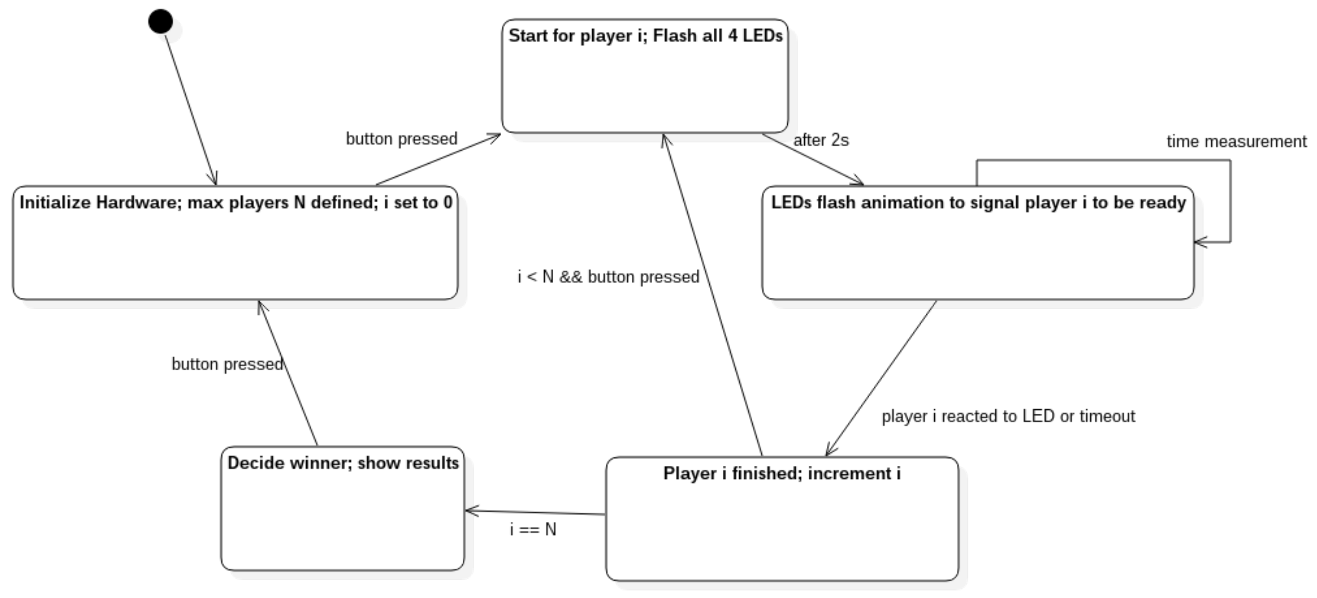

Start for player i; Flash all 4 LEDs
button pressed
after 2s
time measurement
Initialize Hardware; max players N defined; i set to 0
LEDs flash animation to signal player i to be ready
i < N && button pressed
button pressed
player i reacted to LED or timeout
Decide winner; show results
Player i finished; increment i
i == N

Figure 3: Game logic diagram

As illustrated in figure 3 the game can be started after the hardware initialization has been completed. The game is started when the board's user push button has been pressed. Information about the game status and corresponding instructions are printed on a serial monitor. These instructions are shown later on in this paper in chapter 3.

After the push button has been pressed it is the first player's turn and all of the user programmable LEDs start to flash rapidly for a short duration and then the LEDs turn off. This is an indication to the player to get ready and to be prepared to cover the photodiode soon. After this the LEDs start to light up in a clockwise orientation one by one. Once the fourth LED has lighten up, the player has to cover the photodiode as quickly as possible. Player reaction time is measured from the fourth LED lighting up and to the point when the photodiode has been covered.

Then another player starts their turn by pressing the push button and has to interact with the system in the same manner as the player before. This cycle is repeated until all of the players have had their turn. The reaction times are finally displayed and the player who has had the lowest reaction time wins the game. A new game can be started by pressing the push button.

2.2 Peripheral drivers

For simplicity a design choice for this microcontroller project was to avoid the use of multiple threads. Every game logic procedure is executed in a sequential order and only interrupts allow to break this cycle and trigger events concurrently.

In this project, interrupts are used by timers to toggle LEDs and for button detection. For the analog watchdog polling is utilized. This approach allows to sense if the user tries to cheat by covering the photodiode before the fourth LED start to flash. As an alternative to this it would be also possible to realize the watchdog with interrupts and toggle the LEDs with sleep delays.

The Miosix operating system [1] is included in the project workspace and is mainly used for a simplified accessing of the STM32F4 peripheral addresses by calling provided register names. Also the *Timer* class for measuring time and the *Random Number Generator* are exploited.

2.2.1 ADC

The use of the analog-to-digital converter builds the core of the project. A simple photodiode is connected to the board and the photodiode introduces an analog input voltage level depending on the ambient light intensity. The voltage has to be detected and read out as a digital signal in defined time intervals. Since the actual values are not important for this project, there is no data processing necessary. The values simply have to be compared to a watchdog threshold voltage and if they exceed this threshold, further action should be initiated. The analog watchdog provided by the microcontroller is used to execute this comparison.

To enable the the analog-to-digital converter peripheral and to connect the general purpose input/ouptut (GPIO) pin to the ADC peripheral a set of registers is needed to be configured. The desired implementation is to connect the light sensor to the first GPIO of port B (PB0) and connect that GPIO to the ADC.

First the GPIO port where the light sensor is connected needs to be enabled. Since the light sensor is connected to PB0 pin, the port B clock must be enabled. This is achieved by setting the GPIOBEN bit in the RCC AHB1 peripheral clock enable register. In order to connect the GPIO to the ADC it is needed to configure the GPIO port to analog mode of the four possible operation modes. This was done by writing value of 11 to number of the pin number (0) in the GPIO port mode register.

The ADC peripheral is setup after connecting the GPIO port to ADC. From the user manual of the development board [2] the used GPIO could be conncected to first two ADC registers of the three available ADC registers of the microcontroller. This GPIO is connected to the input channel 8 of the ADC. It was decided to use the ADC1 of the microcontroller thus the corresponding clock was needed to be enabled from the RCC APB2 peripheral clock enable register.

After setting up the GPIO register to be used in analog mode and connecting the register to the ADC the following functionality of the ADC was chosen:

- Resolution: 12 bits
- Prescaler: PCLK2/8
- Sampling time of input channel (8): 84 cycles

- Data alignment: right data alignment
- Conversion mode: continuous conversion mode

All functionalities stated above could be set specifically for the ADC1 register except the prescaler value that is set in the common ADC register. Also the watchdog funcitonality of the ADC is utilized to see if the converted value is below a certain threshold meaning that the photodiode has been covered. This threshold value is calibrated everytime the ADC is initialized (board is powered on or reset button has been pressed). The use of the watchdog functionality can be set specifically to be used in one of the three ADCs.

2.2.2 Push button

The user configurable push button (blue push button in figure 1) is used for starting a game after board setup, switching player turn and starting a new game after a game has finished. The push button on the development board is connected to the PA0 GPIO according to the user manual [2] of the development board. The push button uses interrupts to signal the mircocontroller that it has been pressed. The setup and use of the interrupt routine was done by using functions provided by the Miosix OS.

2.2.3 LEDs & Timer

Since the toggling of the LEDs is realized with interrupts, the LED management explicitely includes a hardware timer. The *Led* class knows three different operating modes: At the beginning of a game round all four LEDs flash rapidly, after that the number of blinking LEDs increase from 1 to 3 and finally after a random time interval also the fourth LED becomes active. For all these three modes only one timer (TIM3) needs to be utilized because the blinking procedures happen sequentially.

For every sequence the function *startLedTimer()* is called with a different time interval which sets the prescaler of the timer. Also the new LED mode is passed with the function. With the given mode every time the timer generates an overflow interrupt, the corresponding IRQ handler knows in which way the LEDs should be toggled.

2.2.4 Serial

The serial connection is utilized to display game instructions and the winner of the game in a terminal window of a machine. Communication with other devices is done by using the USART interface. The development board could be connected to a machine with a USB to TTL serial cable since the board itself doesn't a provide a way that the USART interface can communciate with USB. While using miosix the USART connections are predefined. The connection is made with pins PB10 and PB11 with PB10 being the transmitting side and PB11 being the receiving side. Also a operating voltage and ground was provided for the TTL cable. The messages sent by the microcontroller can be viewed with a terminal window application e.g. screen and using a correct transmission rate (19200) indicated in the miosix source code.

2.2.5 Connections

The connections made are described in table 1 and also shown in figure 4. Table 1 describes the functionality of the GPIO used.

Table 1: Connection table

Connector on development board	Function
PB0	Positive terminal of photodiode connected to ADC
PB10	USART Transmit
PB11	USART Receive
GND	Negative terminal of photodiode and USART ground

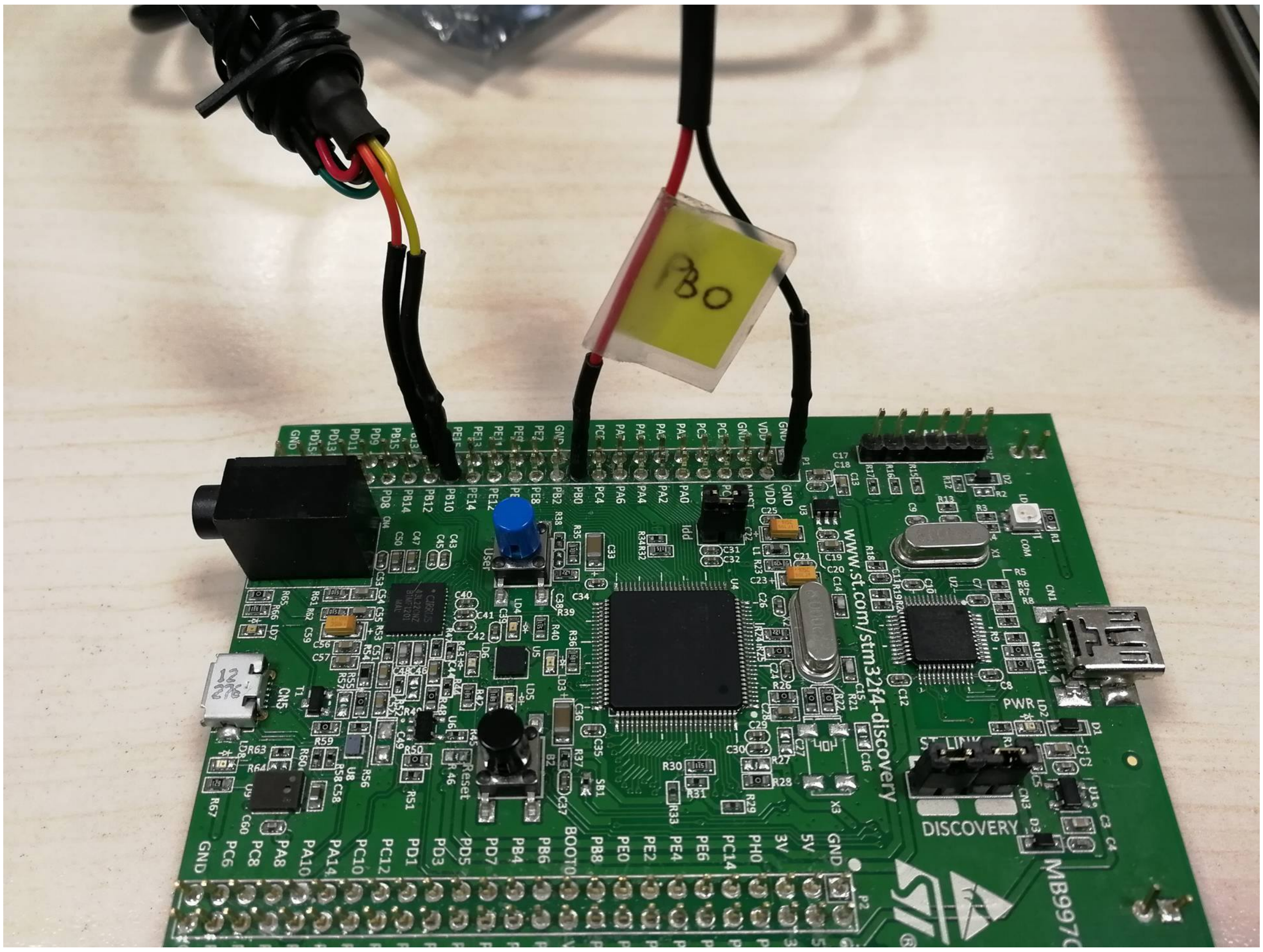
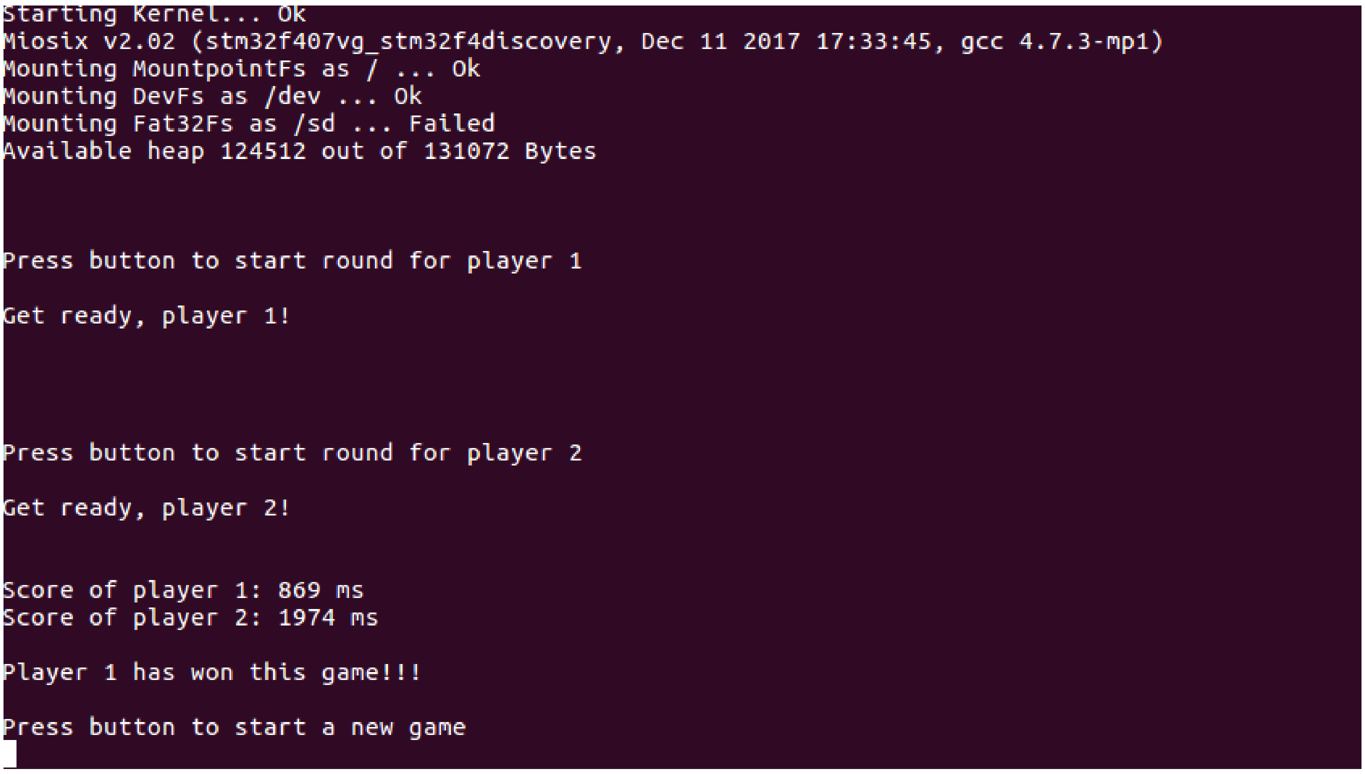


Figure 4: Connections

3 Results

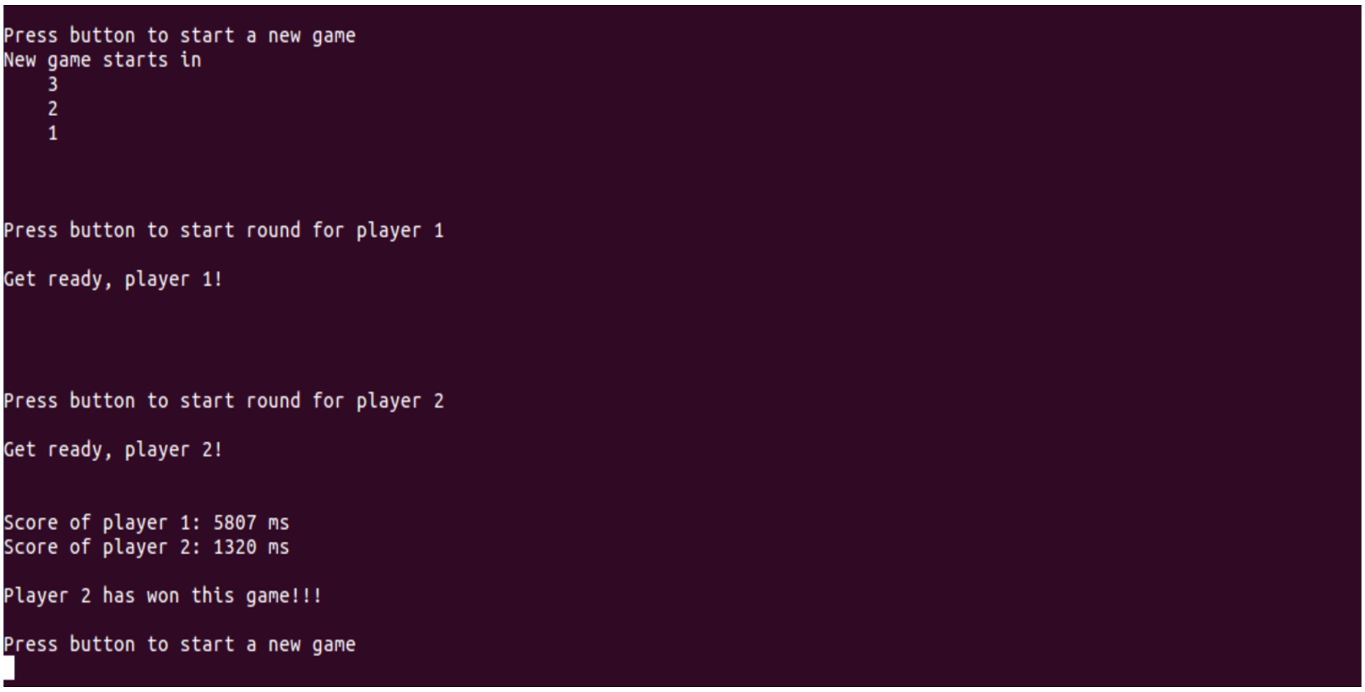
The operation of the system was tested multiple times and also in different locations in order to test the analog watchdog threshold set correctness. After testing the system many times and in different lcoations the system was operating similarly everytime and thus it was concluded that the system is working correctly. The reaction time calculation correctness was tested comparing the results shown by the system and a manual stopwatch. This method isn't the most accurate one but the values were fairly similar so it was deemed to work correctly.

In the following figures 5 and 6 is a sample run of the game logic with starting a new game with pressing the push button after first game has concluded.



```
Starting Kernel... Ok
Miosix v2.02 (stm32f407vg_stm32f4discovery, Dec 11 2017 17:33:45, gcc 4.7.3-mp1)
Mounting MountpointFs as / ... Ok
Mounting DevFs as /dev ... Ok
Mounting Fat32Fs as /sd ... Failed
Available heap 124512 out of 131072 Bytes



Press button to start round for player 1

Get ready, player 1!




Press button to start round for player 2

Get ready, player 2!


Score of player 1: 869 ms
Score of player 2: 1974 ms

Player 1 has won this game!!!

Press button to start a new game
```

Figure 5: System setup and game logic



```
Press button to start a new game
New game starts in
    3
    2
    1



Press button to start round for player 1

Get ready, player 1!




Press button to start round for player 2

Get ready, player 2!


Score of player 1: 5807 ms
Score of player 2: 1320 ms

Player 2 has won this game!!!

Press button to start a new game
```

Figure 6: New game with pressing push button

4 Conclusions

This project was about implementing a analog-to-digital converter driver for the STM32f407 microcontroller and interfacing it with a light sensor. Also the STM32 F4 Discovery kit development board was utilized to prdouce a user reaction time game with using the development boards user programmable LEDs and push button. The main emphasis of this project was to to implement the ADC driver and the game was to utilize the converted value from the ADC. The system functionality was deemed to work as planned and thus satisfying the works goals.

References

[1] Federico Terraneo. Miosix operating system. Website. https://miosix.org/index.html.

[2] STMicroelectronics. User Manual UM1472 for Discovery kit with STM32F407VG MCU. 2017.