HOCHSCHULE
RHEIN-WAAL
Rhine-Waal University
of Applied Sciences

LABORATORY

Microcontroller

5

EXPERIMENT:

Time Clock

Write your name in every sourcefile you edit and compile the code with your matriculation number (variable in the template). All files should contain all names of the persons who made changes. Do use the @author tag for this (as available in the headline of template files).

1 Time Clock

A time clock is a device that records times (arriving/leaving). These devices are used in the commercial environment to record the working time for employees. Here, we want to create a simple time clock device. A classical time clock prints the times on a paper card. New devices are using electronic memory's.

Normally this devices are able to log the times for multiple users, for example everyone has his own RFID card. In this lab we create a very simple device, for one person:

- Push button 1 for arriving
- Push button 2 for leaving
- Both push buttons for showing data

All the parts needed to build a time clock are in the MyAVR-set.

For the memory, we use the EEPROM 24C02 add-on module, while for the real time clock we use the DS1307 add-on module.

To create the device please connect the following hardware parts:

- **Push button 1** (Button Arrive): pin B0
- **Push button 2** (Button Leave): pin B1
- **Pptentiometer** : pin C0 (for optinal task)
- LCD Display **add-on**
- EEPROM **add-on**
- RTC **add-on**

1.1 Time Clock

In the normal operation the time clock should just show the current time. Also it should show if the person is present or absent. If a button for arrive or leave is pressed it should show a confirmation that this time was recorded.

1.2 Data storage

To store the data, we just have a small EEPROM with 256 bytes (24C02). So it is not a good idea to store useless data. Lets assume that the user will never work in the night. New year is never a working day. That means, for one working day we need:

- day

- month
- hour (arrive)
- minute (arrive)
- hour (leave)
- minute (leave)

So we need a fixed memory area of 6 bytes for one working day.

To prevent problems if the power is interrupted, the status (present or absent, number of saved days) should be stored in a nonvolatile memory.

Using the EEPROM would not be a good option! This status will change often and the EEPROM memory would wear out when the same flag is changed again and again.

Background: EEPROM memory can change every storage cell approx 10000 times, after that the cell is worn.

So here the free memory of the DS1307 RTC module can be used. It is battery buffered ram, where frequent writing operations are not a problem.

1.3 User Interface

To operate, the device should use only two buttons, available on the board. These buttons are for coming and leaving.

After the device is powered, it should be in standby mode. The LCD shows:

- the current time,
- present or absent
- number of saved working days

If a button for arrive or leave is pressed a confirmation should be shown. If this button is not plausible the device should ask if the user is sure. If yes, the time for coming/leaving should be overwritten.

When both buttons are pressed together, the device should show the recorded data. With every buttons press it should be possible to see the the next data set day by day. After showing all data the devise should ask if the recorded data should be deleted.

1.4 Testing Mode

To test the device, it would take to much time to wait weeks for creating a full data set. So there should be an testing mode. Please use the C preprocessor for this.

Add code that realizes an automatic time skip after a button for arrive/leave was pressed. So is is possible to fill the memory fast with test data.

Here is a hint how to use the preprocessor for this use case:

```
#define TESTMODE

#ifdef TESTMODE
...
#endif
```

To disable the test mode this should be possible just by disabling one line:

```
//#define TESTMODE
```

Also a good feature for testing is a fast way to reset the memory. Please implement a direct memory reset, when both buttons are pressed when the device is powered.

1.5 Task

Task 1:
Create the time clock with the MyAVR hardware.

Task 2 (Optional):
The access to the stored data is normally not given to the user. Use the ADC and a potentiometer to implement a simple combination lock. This lock should prevent forbidden access to the stored data.
A hard coded four digit pin number is here enough. A push button can be used to enter the next digit.