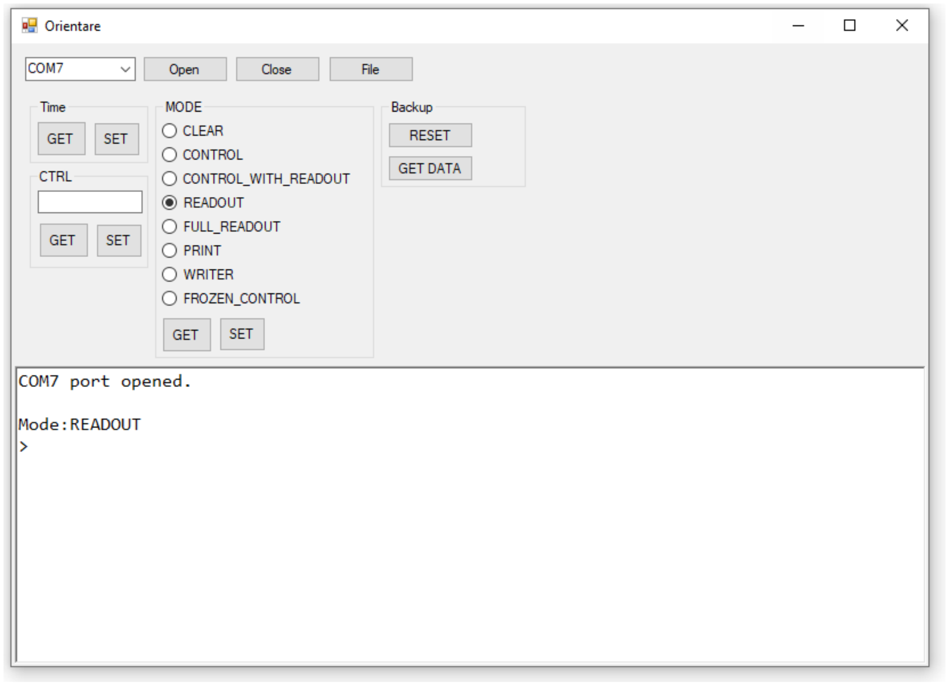
How to configure the Check Points

To configure those Check Points you will use Orientare.exe from CheckPointSoftware\Orientare\bin\Release

1. The Master Point

This station is used to download the data from Clocks (UUID CARDS).

Open Orientare.exe and configure your station with MODE > READOUT and press SET to apply.



MODE: ( ) CLEAR ( ) CONTROL ( ) CONTROL_WITH_READOUT [✓ READOUT] ( ) FULL_READOUT ( ) PRINT ( ) WRITER ( ) FROZEN_CONTROL

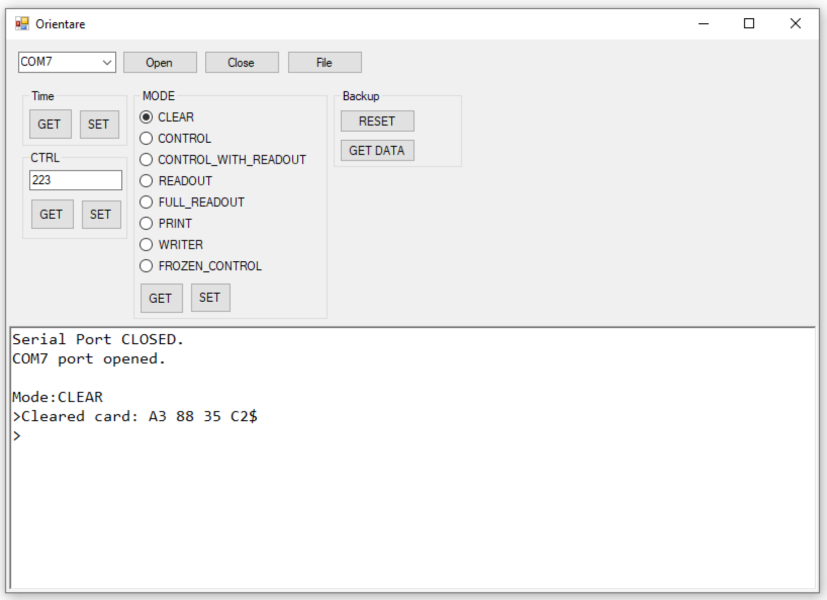
```
COM7 port opened.

Mode:READOUT
>
```

Remember to select where do you want to Save the files on your computer using "File" button or the software will stop running.

2. Clear Check Point

This station is used to clear all data from the Clock (UUID CARDS).

- Open Orientare.exe and configure your station with MODE > CLEAR and press SET to apply.

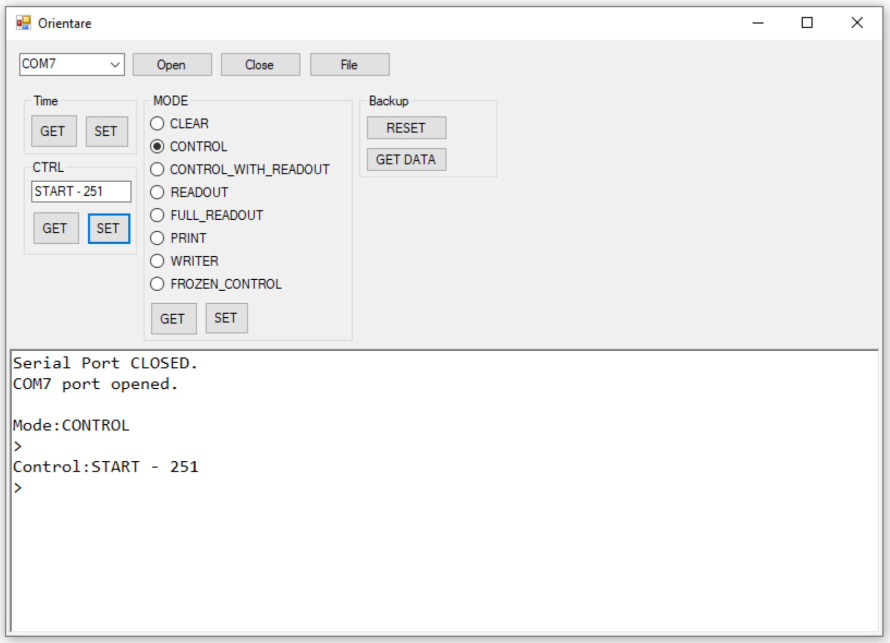


```
Serial Port CLOSED.
COM7 port opened.

Mode:CLEAR
>Cleared card: A3 88 35 C2$
>
```

3. Start Check Point

The code for Start Check Point is: 251

- Open Orientare.exe and configure your station with: MODE > CONTROL and press SET to apply and in the CTRL Field you will insert: 251 after that you will press SET.



```
Serial Port CLOSED.
COM7 port opened.

Mode:CONTROL
>
Control:START - 251
>
```

3. Finish Check Point

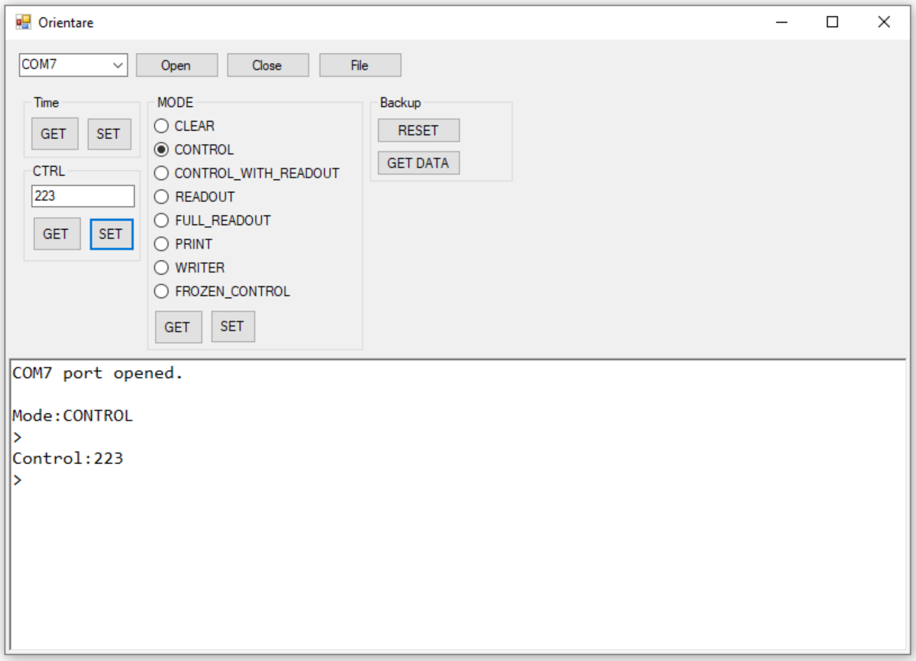
The code for Finish Check Point is: 252

- Open Orientare.exe and configure your station with: MODE > CONTROL and press SET to apply and in the CTRL Field you will insert: 252 after that you will press SET.

4. Intermediare Check Points

Those station can have the CTRL with any integer number like "223" or "10".

- Open Orientare.exe and configure your station with: MODE > CONTROL and press SET to apply and in the CTRL Field you will insert: 223 after that you will press SET.
- You need to repeat those steps for each Check Point (station).



COM7

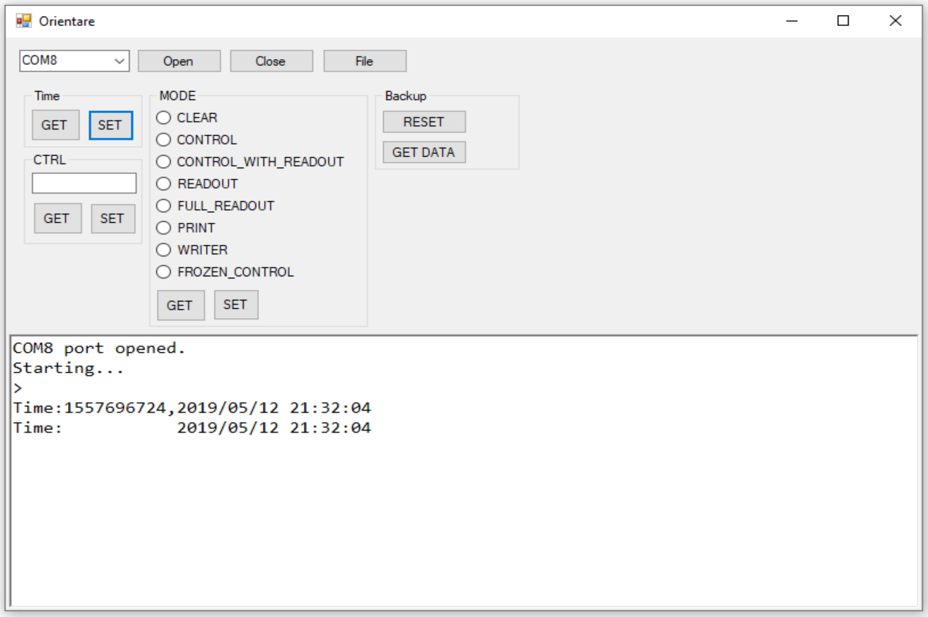
MODE: (●) CONTROL

CTRL: 223

```
COM7 port opened.

Mode:CONTROL
>
Control:223
>
```

IMPORTANT!!!

Before every competition, you need to resync the clock from every CheckPoint/Station.



COM8

```
COM8 port opened.
Starting...
>
Time:1557696724,2019/05/12 21:32:04
Time:           2019/05/12 21:32:04
```

Usage of the system

1. before start a CLEAR control (station) is used to clear out the card and prepare it
2. on start an additional station can be used (with control ID 251). On mass start a prewritten control with time can be added (WRITE CONTROL command), or a control with frozen time (FROZEN_CONTROL) can be used.
3. on the racetrack the competitors touch the RFID card by the control, where each control has its unique id (from 1 to 250)
4. on the finish the competitors touch against the finish control (ID 252).
5. after the finish the station in READOUT mode is used to read the timing on each control.

Before step 1. an administrator can reset the backup memory of each control so the memory of which cards passed can be saved.

Usage of the station

Stations support serial communication (38400 8 N 1). All configuration is done through serial terminal. Use "?" for help. *Note*: use Arduino serial monitor for communication or a system that sends *newline* at the end of the line, as most systems don't.*Note* : baud rate of 38400 is used. On some Arduino boards such speed isn't supported and it needs to be slowed - suggested speed is 9600.

Serial commands

- **SET**
  - TIME YYYYMMDDhhmmss : Set time in year[Y], month[M], day[D], hour[h], minute[m], second[s] format (eg. 20160817120024 is 2016/08/17 12:00:24)
  - CTRL number : Set control ID as number (1-250), with special codes for START and FINISH stations (special id number defined in code)
  - MODE type : Set modes (possible modes are CLEAR, CONTROL, CONTROL_WITH_READOUT, FROZEN_CONTROL, FULL_READOUT, PRINT, READOUT, WRITER)
  - RESET_BACKUP : Reset pointer to 0th location on external EEPROM backup memory. In reality not neccessary as it rolls over, but nice to have a clean start.
- **GET**
  - TIME : Get time in format YYYY/MM/DD hh:mm:ss - year[Y], month[M], day[D], hour[h], minute[m], second[s] eg. 2019/05/12 12:00:00

    - CTRL : Get control ID as number (1-250 for normal controls, special codes for START and FINISH)
    - MODE : Get mode status (CLEAR, CONTROL, CONTROL_WITH_READOUT, FROZEN_CONTROL, FULL_READOUT, PRINT, READOUT, WRITER)
    - BACKUP : Get backup of whole external EEPROM (backup) memory. It returns all locations, even those which are not used.
    - VERSION : Get current firmware version
    - VOLTAGE : Get current battery voltage if the battery is connected to analog port 2. (may not work as expected)
    - **WRITE**
        - INFO Any_text : Write any text up to 30 characters which will be used in PRINT command as a CARD owner/user
        - CTRL number YYYYMMDDhhmmss : Writes a control at the end of current controls. Useful to add a finish time in case of error, or to setup some controls (eg. START) prior to race. *Note* : WRITE commands work only in WRITER mode.
    - **PING**
        - command PING will respond with PONG. It should be used as a keep awake system.

Questions:

- What are CLEAR, CONTROL, CONTROL_WITH_READOUT, FROZEN_CONTROL, READOUT, FULL_READOUT and PRINT controls?
    - CONTROL is a normal station which saves data to RFID card, and sends (via serial) the UID of the card, timestamp in UNIX epoch format and a readable time.
    - CLEAR completely erases all the data on the RFID card.
    - READOUT reads every written control on the RFID card, from the first written to the last written on the card, and sends it via serial. It should be used with software on a computer.
    - CONTROL_WITH_READOUT does the same as control, but additionaly reads every previous control off the card. This is usable if some of the controls in the middle of the race are connected over Internet. Also it's usable to have one control for FINAL + automatic readout
    - FULL_READOUT is used when all 125 places on the card want to be read.
    - FROZEN_CONTROL is a station which has it's clock stopped. So every competitor will have the same time on that control. It's function is primarily for races with mass start (all

competitors start at the same time), so the organizers do not need to set that into the software.
    - PRINT is a station which will nicely format and calculate the results of the station on the card, and send it via serial. It also prints the name of the user/owner of the card. It doesn't need the software on a computer. *Note* : There is a WRITER mode, but in that mode a command is to be given to a station prior of putting a card next to the RFID reader.

- How many controls can one RFID card accommodate?
    - Short answer: 125
    - Long answer: The system is prepared for MiFare 1K cards, which have 64 blocks (divided in 16 sectors) of 16 bytes. Each sector has 3 usable blocks, and one trailer block which has special meaning. First sector (blocks 0 - 3)is also off the limits. Second sector is prepared for info block (block 4) while blocks 5 and 6 are left free for anything (eg. competitor names). Data starts at block 8. Out of 56 blocks, there are 14 trailer blocks which are unusable, so control data can come to 42 blocks. Data about the control is saved in 5 bytes: 1 byte is control ID (number between 0-255), and 4 bytes is UNIX timestamp of the coming to control. So there are 3 controls per block - in total 42 x 3 = 126. There is a hard limit in code to 125.
    - For an visual answer look at the Documentation/CardAllocationOTIMCON.ods

- Is data safe on RFID card?
    - Data retention of the RFID memory is 10 years, or 200000 writes. So it's OK for a normal use. Additionaly, every block on the card has an additional CRC8 check on the end, so the data is written as it should.

- For what is BACKUP used?
    - If there is support for external EEPROM, then after writing data about the control onto the RFID chip, there is an additional write to external EEPROM with UID of the RFID card and timestamp.

- How big is BACKUP?
    - Backup saves last 512 cards (one card can be more than once, for those who forget to erase).

- What is info block in the RFID card?
    - Info block is block 4, where the last location is saved, last control and time of the last control.

- It looks like the serial port isn't working?

    - Make sure you're using a serial terminal which sends \n newline (most of the terminal software doesn't). Serial monitor in Arduino IDE works nicely.
    - Also make sure you're not using LOW POWER mode - in that case use an RFID card on the station prior to serial communication (see next question).
    - My station worked fine when I turned it on, but after an hour they are dead?
        - Check that LOW_POWER option is not active, and that HIGH_POWER is active.
        - If LOW_POWER is option is active, then each station goes through three states of work which are AWAKE -> SHALLOW SLEEP -> DEEP SLEEP. The station has to be in AWAKE mode for normal function (read/write), and the system is working at full speed. If no card or no serial command is present for 5 seconds, the station goes into SHALLOW SLEEP. In that mode station checks for card or serial command once every 250ms. If a card or serial command is present, it goes into AWAKE state. If no card or serial command is present in an hour of work in SHALLOW SLEEP, the station goes into DEEP SLEEP. In DEEP SLEEP the card or serial command is checked once every 2 seconds. Due to other electronics which needs to wake up, it can be up to 10 seconds needed for a control to fully wake up into AWAKE state. Just be patient. Time of 5 seconds used in AWAKE is chosen to not slow runners which came in bursts (several runners at once). Station can be forced in AWAKE state if any of the serial command is issued - PING can be used for keep AWAKE (or use HIGH_POWER option). The time of SHALLOW SLEEP of 250ms is not noticeable to runners. It can be lowered if needed.
- How can you use a MRFC522 on 5V?
    - MRFC522 is powered on 3.3V, and it will burn if you try to power it on 5V. But it seems that it can live with 5V on it's SPI port. The schematic given is my test bed, and all the orhers are done on a 3.3V Pro mini. DS3231 can work on either 3.3V or 5V, so it doesn't matter to it, this is just for the MRFC522.
- I did a RESET_BACKUP function but old data is still present in backup?
    - RESET_BACKUP just resets the internal counter back to 0 (first position), and starts overwriting from there. Nothing to be alarmed, it's like that by design.