

Bachelor thesis



**Czech
Technical
University
in Prague**

F3

**Faculty of Electrical Engineering
Department of Measurement**

Live Transmissions of Results of Orienteering Competitions

Jan Jurica

**Supervisor: Prof. Ing. Jan Holub, Ph.D.
Field of study: Cybernetics and Robotics
May 2020**

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Jurica** Jméno: **Jan** Osobní číslo: **466302**
Fakulta/ústav: **Fakulta elektrotechnická**
Zadávací katedra/ústav: **Katedra měření**
Studijní program: **Kybernetika a robotika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Živý přenos výsledků během závodů v orientačním běhu

Název bakalářské práce anglicky:

Live Transmissions of Results of Orienteering Competitions

Pokyny pro vypracování:

Provedte rešerši dostupných zařízení pro živý přenos výsledků v průběhu závodů v orientačním běhu. Na základě této rešerše navrhnete, realizujete a otestujete zařízení, které bude tento přenos provádět vámi zvoleným způsobem. Pro návrh využijte platformu Arduino, případně STM.

Seznam doporučené literatury:

[1] DOBKIN, Daniel M., 2012. The RF in RFID: UHF RFID in Practice. 2. vyd. Oxford: Elsevier

Jméno a pracoviště vedoucí(ho) bakalářské práce:

prof. Ing. Jan Holub, Ph.D., katedra měření FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **23.10.2019**

Termín odevzdání bakalářské práce: _____

Platnost zadání bakalářské práce:

do konce letního semestru 2020/2021

prof. Ing. Jan Holub, Ph.D.
podpis vedoucí(ho) práce

podpis vedoucí(ho) ústavu/katedry

prof. Mgr. Petr Páta, Ph.D.
podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

Datum převzetí zadání

Podpis studenta

Acknowledgements

I would like to thank Prof. Ing. Jan Holub, PhD for professional guidance during the elaboration of the bachelor's thesis. I would also like to pay my special regards to Ing. Kamil Pipek for his time and willingness he devoted to me during the elaboration of the bachelor's thesis.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, May 17, 2020

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, 17. května 2020

Abstract

This thesis describes a solution for live transmission of results of orienteering competitions. First, methods for measuring orienteering races are described. Subsequently, research of current solutions for live transmission of results of orienteering competitions is made. Based on the research, a new device which enables the transmission of live results is designed, assembled and tested. The design of the device is based on available open-source software and hardware so that potential users can build it themselves at home at low costs.

Keywords: orienteering, live results, radio transmission, open-source, Arduino

Supervisor: Prof. Ing. Jan Holub, Ph.D.

Abstrakt

Tato práce popisuje řešení pro odesílání živých výsledků během závodů v orientačním běhu. Nejprve jsou popsány metody měření závodů v orientačním běhu. Následně je provedena rešerše současných řešení pro odesílání živých výsledků během závodů. Na základě rešerše je navrženo, sestaveno a otestováno nové zařízení, které umožňuje přenos živých výsledků. Design zařízení je založen na využití dostupného open-source softwaru a hardwaru, aby si ho případný uživatel mohl sám postavit doma za nízké náklady.

Klíčová slova: orientační běh, živé výsledky, radiový přenos, open-source, Arduino

Překlad názvu: Živý přenos výsledků během závodů v orientačním běhu

Contents

1 Introduction	1	5 Proposed solution	25
2 What is orienteering	3	5.1 Description of the solution	25
3 Timekeeping and punching systems	7	5.2 Control unit	28
3.1 Individual race description	7	5.2.1 Programming of the device . .	29
3.2 Punching systems	8	5.3 Punch data detection	31
3.2.1 SPORTident system	9	5.4 Data transmission	36
3.2.2 Emit system	13	5.4.1 Cellular network	36
4 Current solutions for live transmission of results	15	5.4.2 Web application for testing . .	41
4.1 Readout methods of SPORTident stations	15	5.4.3 Local WiFi network	42
4.2 SPORTident SIGSM-DN	17	5.5 Power supply	43
4.3 ROC - Radio Online Control . . .	18	6 Testing	47
4.4 jSh Radio	19	6.1 Punch records reception test . . .	47
4.5 Tinymesh project	21	6.2 Test of data upload to a remote server	48
4.6 Racom	23	6.3 Testing at a measured training race	48
		6.4 Testing at the official competition	52
		7 Discussion	53
		7.1 Future work	54

8 Conclusion	57
A Bibliography	59
B Content of enclosed CD	63

Figures

2.1 The international orienteering symbol. Source:[DU20]	3	4.4 SPORTident Web Service. Source: [SPO20]	18
2.2 An example of the orienteering map. Source:[Par20]	4	4.5 An example of Radio Online Control. Source: [BB20]	19
2.3 A competitor during a race. Photo by Vojtěch Illner	5	4.6 An example of jSh Radio device. Source: [Har20]	20
3.1 Process of punching using the SPORTident punching system. Backup pliers are on the left side of the stand. Photo by Vojtěch Illner .	9	4.7 jSh mesh radio network. Source: [Har20]	21
3.2 Comparison of SI-Card5 (left), SI-Card9 (middle) and SIAC (right).	11	4.8 The radio unit by Plengqui. Source: [L20]	22
3.3 The SPORTident BSF8 station .	11	4.9 A roll-up slim jim antenna. Source: [L20]	22
3.4 Size comparison of SI-Card9 (left) and BSF8-SRR (right).	12	4.10 Racom device with the external antenna mounted on a tree. Source: [Rac20]	24
3.5 Emit eCard (top left), control station (top right), emiTag (bottom left) and Touch-Free control station (bottom right). Source: [EU20] . . .	13	5.1 The first prototype of the device installed into a plastic suitcase . . .	26
4.1 BSF8-USB. Source: [SPO20] . . .	16	5.2 Plastic suitcase for the device . .	26
4.2 SRR dongle	16	5.3 The wiring diagram	27
4.3 SPORTident SIGSM-DM. Source: [SPO20]	17	5.4 Clone of Arduino Mega2560	28
		5.5 Interior of the SRR dongle	31
		5.6 Module of the SI-SRR receiver .	32
		5.7 SIM800L module equipped with an electrolytic 1000uF capacitor . .	37

5.8 SIM800L module with a surface-mount 1000uF capacitor, 5V voltage stabilizer and an external antenna	38
5.9 Screenshot of testing web application.....	42
5.10 ESP-01 module with ESP8266 microchip.....	42
5.11 A device with the ESP-01 module	43
5.12 Micro USB charger for Li-Ion batteries.....	44
5.13 Step-Up module	44
6.1 Training race participant punches measured control point. Suitcase with the device is placed above the control point in the top left corner of the picture.....	49
6.2 Device placement at the measured control point	50
6.3 Graph of the battery discharge process	51
6.4 Graph of signal strength during the test.....	51

Tables

6.1 Punch records reception test....	48
6.2 Results of the test of data upload to a remote server.....	48
6.3 Results of the test at the training race.....	50



Chapter 1

Introduction

The topic of this bachelor thesis focuses on modern technologies used in orienteering. I chose this topic because I have been doing foot orienteering since I was 12 years old. Since then, I have built a very positive attitude towards this sport and I would like to contribute to its further development. Therefore, I decided to combine my studies with the orienteering in this bachelor thesis.

Orienteering is becoming more popular every year, and more people are starting to do the sport. The disadvantage of competitive orienteering is that it is not very attractive to spectators. The competitors are in the forest where they cannot be seen by the audience for almost the entire duration of the race. Various solutions are used to increase the attractiveness of the competitions for spectators so that they can watch the movement of the competitors indirectly. One of these solutions is the placement of electronic devices at the measured control points. These devices then transmit live results to spectators as competitors pass through control points.

The goal of this thesis is to make research of currently available solutions for live transmissions of results of orienteering competitions. Based on the research, I will design, implement and test my own device which will transmit live results in my chosen way.

The design of the device will be based on available open-source software and hardware so that potential users can build it themselves at home at low costs. The device can serve as a cheap option for the transmission of live results of orienteering competitions.

Chapter 2

What is orienteering

Orienteering is a sport based on the ability to navigate through terrain aided only by a map and a compass. At the start of a race, the participants are given a topographical map of an unknown terrain. The map is specially prepared for each orienteering discipline. Participants are expected to find all control points which are shown on the orienteering map in a predetermined order. Control points are marked in the terrain with a control code and a high visibility flag. The flag also stands for the international orienteering symbol (Figure 2.1). Figure 2.2 shows an example of the orienteering map with marked control points. Results of an orienteering competition are determined by the successful finding of all control points of the course in the shortest time. Success in orienteering depends not only on excellent physical skills and moving speed but also on the mental capacity to choose the fastest most efficient route between control points.

Orienteering is divided into many disciplines depending on a specific method of travel. According to International Orienteering Federation [Fed20], foot orienteering (FootO), mountain bike orienteering (MTBO), ski orienteering (SkiO) and trail orienteering (TrailO) are classified as official disciplines in

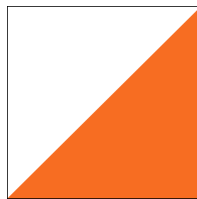


Figure 2.1: The international orienteering symbol. Source:[DU20]

the sport of orienteering. TrailO is the youngest and the most different discipline which brings the challenge of orienteering also to handicapped athletes. In TrailO, it is not important to move quickly between control points but precisely read the map and determine the exact location of the control points. Besides the official disciplines, there are many other such as radio orienteering, biathlon orienteering, rogaining and more.

Foot orienteering (or simply *orienteering*) is the oldest and the most popular orienteering sport. Participants do not need any special equipment other than sports clothes, running shoes and a compass. Orienteering competitions have a wide number of categories based on distance and difficulty and also on gender and age of competitors. Long and middle distance competitions are held in forests but shorter distances also called sprints can be set in urban areas such as city centres and parks. The cradle of orienteering is Scandinavia due to its large forests and difficult terrain. The history of the sport in the Czech Republic dates back to the 1950s. Nowadays, orienteering is becoming increasingly popular. A statistic from Český svaz orientačních sportů (hereafter CSOS) [Če20c] says that in 2018 there was over 11000 registered competitors in the Czech Republic.

Formal competitions take the form of an individual or a relay race. To maintain equal conditions for all competitors, the starting method for individual races is an interval start. The competitors of one category have the same course. The race is a time trial and the winner is the competitor with the fastest time. The relay race is usually run by teams of 3. Each team member runs a consecutive individual race. The sprint relay composed of four competitors where the first and the last must be women. However, some relay

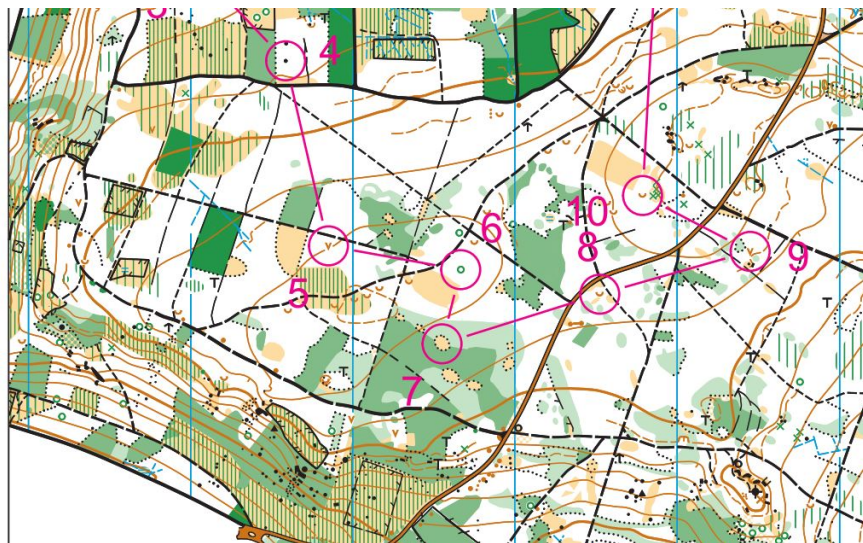


Figure 2.2: An example of the orienteering map. Source:[Par20]



Figure 2.3: A competitor during a race. Photo by Vojtěch Illner

races may be for teams which have up to ten members. The relay is a mass start event which means that up to several dozen competitors can go on the track at the same time. In this case, competitors may be running in close proximity to each other but the rules still demand independent navigation. In the Czech Republic, the orienteering season starts with the first competitions in March and ends in October. The competitions take place almost every weekend from small regional events to National Championships. World Championships are held every summer. Orienteering belongs to the Olympic sports family but despite this fact, it has never been included in the Olympic Games program.

All the following chapters are written in terms of foot orienteering (hereafter *orienteering*) and the technologies that are used in it.

Chapter 3

Timekeeping and punching systems

The evaluation of an orienteering competition is governed by the rules of the specific variant of the race. Section 3.1 describes the timekeeping of the individual race and the evaluation of the correct completion of the race by a punching system. Section 3.2 gives a closer look at the different technologies of the punching systems.

3.1 Individual race description

As it was already mentioned in Chapter 2, the individual races take the form of an interval start. Before the start, the competitor receives a control description sheet (or clue sheet) which describes the control points position in relation to objects on the track and the control point codes. The competitor is also required to carry a control card of specified punching system (Section 3.2) throughout the race. The time of the race is measured from the competitor's start time or from the moment when a start control point is punched. After the start, the competitor is given a map where all control points of the race are shown. The competitor must punch all the control points to the control card in the right order. The race timekeeping is stopped when the competitor crosses the finish line or when the finish control point is punched. The competitor is obliged to present the control card for inspection after the race. The competitor is included in the results if all control points are recorded in the control card correctly and in the right order. It is the responsibility of each competitor to have all the control points of the race punched in the control card.

the station code but also the time when the station was visited are written to the control card. Besides the possibility of the race evaluation and total time, the split times between control point can be included in the results. In addition to two electronic systems which are described in the following sections, the International Orienteering Federation (hereafter IOF) [Fed20] allows also the use of Learnjoy system or SFR system Classic. But these are not used in Europe so they are not addressed in this thesis.



Figure 3.1: Process of punching using the SPORTident punching system. Backup pliers are on the left side of the stand. Photo by Vojtěch Illner

■ 3.2.1 SPORTident system

SPORTident GmbH [SPO20] is a technological company with head office in the middle of Germany in Arnstadt. Since the year 1996, the company has been developing a timekeeping and identification systems. It provides timekeeping solutions for all orienteering sports, MTB Enduro and Trail Running. Furthermore, the SPORTident system is completely approved by the IOF with classical and contactless punching system. It is the most widely used punching system in Europe from small regional races to world-class competitions. And despite the fact that it is not the only allowed punching system, practically no other system is used in the Czech Republic.

The basic principle of the SPORTident system is the same as it is described at the beginning of Section 3.2. The system uses a combination of Radio Frequency Identification (hereafter RFID) and Short Range Radio (hereafter SRR) technology and it consists of two basic devices - cards and stations.

■ Cards

Two types of SPORTident cards are currently in use - SPORTident classic cards (hereafter SI-Cards) and SPORTident ActiveCards (hereafter SIACs). SI-Cards are small passive RFID tags that a competitor can carry on his finger attached with a rubber band. Within the information stored in SI-Card memory are SI-Card number and individual punch records. One punch record consists of a station number and split time. SI-Card10 version can store up to 128 punch records and its data exchange time with a station is 60 ms. For comparison, older SI-Card9 has storage for 50 punch records and data exchange time 115 ms. The advantage of the newer model is speed rather than memory, as the classic races do not have more than 40 control points. It is necessary to insert the tip of the SI-Card directly into the hole of a station to record the time and control point number.

The newest type of SPORTident card is the SIAC which uses SPORTident AIR+ system. It is the extension of classic SI-Card with SRR technology which enables contactless punching. SPORTident SRR (hereafter SI-SRR) has a radio frequency of 2.4 GHz which supports recording of data to SIAC as a competitor passes the station in a distance of up to 8 metres. However, the IOF allows the maximal distance to be 30 cm. The use of the SI-SRR resulted in the SIAC having its own battery which makes it more robust than the regular SI-Cards. Figure 3.2 shows the comparison of two SI-Card versions and the SIAC. The contactless punching mode of the SIAC has to be switched on before each race by punching the CHECK, SIAC-ON or START station and it is switched off by punching the FINISH or SIAC-OFF station. The SIAC can still operate in the direct mode as classic SI-Cards if the contactless mode is switched off. The storage of the SIAC is capable to store up to 128 punches. The data exchange time is 60 ms in direct mode and 50 ms in contactless mode. The main advantage of the SIAC in contactless mode is that it does not have to be inserted into the hole of a station during data exchange so the competitor can pass the station without any slow down. According to [PB20], a competitor with SIAC can be up to 1.2 seconds faster than one with classic SI-Card.



Figure 3.2: Comparison of SI-Card5 (left), SI-Card9 (middle) and SIAC (right).

■ Stations

The most recent version of SPORTident station is BSF8 (Figure 3.3). Stations are active devices with an RFID reader in the hole section. Each station also has a real-time clock system which offers time resolution of approximately 4 ms and its backup memory can store more than 20 000 punching records. BSF8 may be equipped with an extra SI-SRR transmitter (BSF8-SRR) which is explained in Section 4.1.



Figure 3.3: The SPORTident BSF8 station

The stations can operate in two modes - direct punching only and beacon mode. In the direct punching only mode, the station can be punched only by inserting a card into the hole whether the SIAC contactless punching mode is activated or not. The SIAC contactless punching is possible with the beacon mode. The station in beacon mode sends out the data and once the SIAC

passes in the preset distance it receives the punch record data. A minor disadvantage is that the station backup memory only stores the punch records created with the direct punching.

The BSF8 stations are equipped with a lithium battery. The battery of BSF8 has a capacity of 1000 mAh and is non-rechargeable. The stations are always in a low power Stand-by Mode so there is no need to switch them on. The stations are switched from Stand-by to Active Mode with the first direct punch. Then, they work in preset beacon or direct punching only mode. It is important to switch each station back to Stand-By Mode after each race using special Servis-Off card to extend its battery life. The SPORTident claims that if the BSF8 station is used in beacon mode twice a month in a race with five hundred competitors, the battery life can be 3 to 4 years.

The internal clock, the code number and the working mode of each station can be freely reprogrammed according to the needs of the race by using PC software SI-Config by SPORTident. This can be done by BSM8-USB station (Figure 4.1) which is equipped with a USB cable. Station transmission speed can be set to 4800 Baud or 38400 Baud. The main purpose of the BSM8-USB station is to work as a readout station for the cards after the race. The data from the cards are uploaded to the appropriate evaluation system.



Figure 3.4: Size comparison of SI-Card9 (left) and BSF8-SRR (right).

3.2.2 Emit system

Emit AS [AS20] is a Norwegian company which has been providing its electronic punching and timekeeping system for orienteering since the year 1995. The system was originally developed in collaboration with Norwegian and Swedish Orienteering Federation. It has also been used in international competitions. Today, the Emit system is used mostly in the Scandinavian countries and not as often as the SPORTident system.

Original Emit eCard system works on a similar principle as the SPORTident direct punching system. Each competitor carries the Emit eCard which must be precisely placed onto the control station. A paper card is placed on the underside of each eCard on which a mark is stamped during each punching as a backup in the event of an electronic punching failure. Compared to the SI-Card, the eCard is larger and the competitor must carry it in the palm of the hand in order to punch the control. Besides the eCard system, Emit developed also the Touch-Free system which is used at the competitions with contactless punching. In this case, the competitor carries emiTag attached to the wrist. During the punching process, the emiTag is placed near the control station and the successful punching is confirmed by an audible and visual signal.



Figure 3.5: Emit eCard (top left), control station (top right), emiTag (bottom left) and Touch-Free control station (bottom right). Source: [EU20]

Chapter 4

Current solutions for live transmission of results

This chapter provides research on current solutions for live transmission of results that work with the SPORTident system [SPO20], since it is the only electronic punching system used in orienteering in the Czech Republic.

4.1 Readout methods of SPORTident stations

There are two readout methods of SPORTident stations placed at the measured control points. One of them is the use of the BSF8-USB station (Figure 4.1). This station is primarily designed for reading cards after the race. With Config+ software, the BSF8-USB station can be freely reprogrammed to operate as a station place at any control point. The station can write punch records to a card and at the same time send the records via USB to any connected device. Since the BSF8-USB station does not have a battery, the connected device must not only read the data but also power the station itself. The BSF8-USB station is capable of both direct punching only and beacon mode.

The second readout method uses the BSF8-SRR station (Figure 3.4) which was developed for the contactless punching system SPORTident AIR+. In addition to the classic BSF8, this station has the SI-SRR transmitter which supports wireless transmission of punch records. The SI-SRR works in



Figure 4.1: BSF8-USB. Source: [SPO20]

the 2.4 GHz radio band which is the worldwide license-free radio band. This enables the BSF8-SRR station to transmit currently created punch record of the direct punching (i. e. when the station is punched with an SI-Card). The SIAC features a radio which can provide fully bidirectional data transfer and sends the created punch record by itself. The punch records transmit from the BSF8-SRR stations or the SIACs are received by the SI-SRR receivers. To achieve a robust data transmission with a low error rate, the SI-SRR uses two radio channels named BLUE and RED. Although the data sources (the SIAC and the BSF8-SRR) transmit simultaneously in both channels, the SI-SRR receivers always operate in only one channel. The radio channel of the SI-SRR receiver can be configured by Config+ software. The distance of the source and the receiver can be up to 6 metres. The SPORTident offers the SI-SRR receivers as fully functional dongles (Figure 4.2) or as modules which can be integrated into any hardware projects. The SI-SRR receivers are described in more details in Section 5.3.



Figure 4.2: SRR dongle

4.2 SPORTident SIGSM-DN

SPORTident SIGSM-DN is a device which provides live data transmission to a web service over a cellular network. The device is equipped with two SI-SRR receivers for both RED and BLUE channel. It has a built-in SIM card with general service which automatically selects the best signal provider. The SIGSM-DN is powered by a rechargeable battery which can be charged by a standard mini USB connector. Estimated operating time is 96 hours. The whole device is waterproof. The LCD display on the front side can show information about time, signal strength, battery level and more.

The device can be rented from SPORTident. It is delivered as a ready-to-use device with an active SIM card. Punch records are directly uploaded to SPORTident web service from where the organizers can download them to any evaluation software. The process is shown in Figure 4.4. Any additional configurations of SIGSM-DN can be done by Config+ software. The costs for the use of the device are based on the number of competitors per race and day.



Figure 4.3: SPORTident SIGSM-DN. Source: [SPO20]

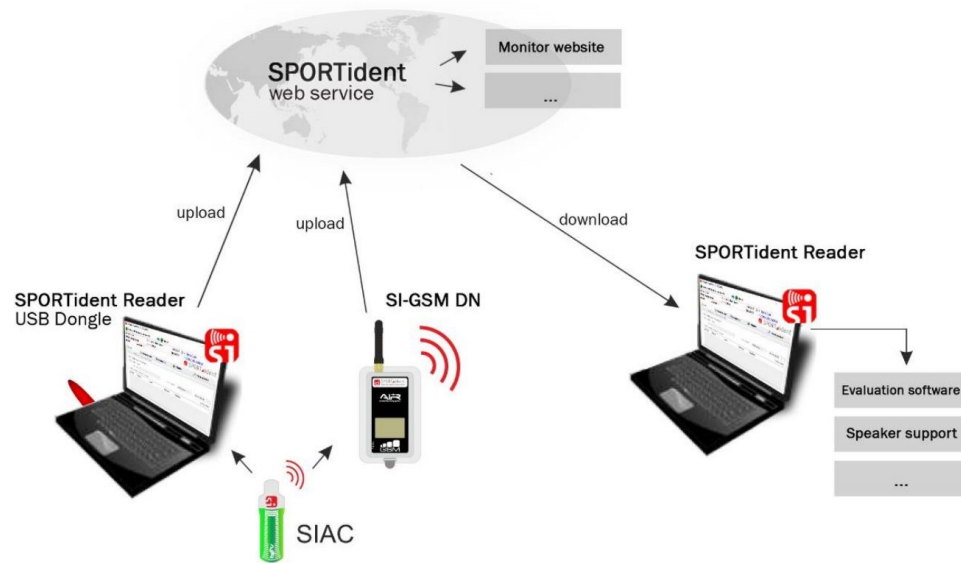


Figure 4.4: SPORTident Web Service. Source: [SPO20]

4.3 ROC - Radio Online Control

Radio Online Control (hereafter ROC) [BB20] was developed by brothers Oskar and Erik Berg from Sweden. The main goal was to make it possible for everyone to build their own device at home. The necessary hardware is listed on the project website and the user must purchase their own hardware. The build of the own device is simple because it is based on single-board computer Raspberry Pi. Bergs created a customized operating system for the Raspberry Pi which can be downloaded from the ROC website after entering some basic information about the user. After installing the ROC operating system, the device is able to receive punch data using the BSF8-USB station and SRR dongle that can be connected via USB ports. The user chooses the internet connection compatible with Raspberry Pi as needed. The suggested option is to use a 3G modem plugged via USB which can transmit data using a mobile network. The platform also allows Ethernet or WiFi connectivity. It is recommended to use at least 10 000 mAh power bank to power supply the device on remote places. After assembling the entire device, the user must register it on the ROC website. As a result, the device is able to connect to the project server where individual punch data is automatically uploaded. The data can be then downloaded using the project API to an evaluation system. There are no upload and download fees so the price of the device is based only on the price of the purchased hardware and the mobile provider's fees.



Figure 4.5: An example of Radio Online Control. Source: [BB20]

Besides the device based on Raspberry Pi, it is also possible to create the ROC device with any Android smartphone. The smartphone needs to have installed the SI-Droid ROC application by Johan Jacobsson [Jac20]. The smartphone must also support USB On-The-Go technology which allows connecting USB devices. The BSF8-USB or SRR dongle can be connected to the smartphone using an adapter. The device registration and the data extraction are similar to Raspberry Pi devices.

4.4 jSh Radio

The jSh Radio [Har20] is a project by Simon Harston from Darmstadt, Germany. It is a system for transmitting punch records from control points to the finish area through a self-configuring mesh radio network. The network consists of one gateway device and multiple router devices as shown in Figure 4.7. The gateway is located in the competition centre and is connected to a PC which uploads received punch records to an evaluation software. The routers are placed at each control point location to receive individual punch data from stations or can serve as repeaters to extend the radio signal from remote sites. The mesh radio network is created as soon as the devices are powered on. The devices periodically check the network routes which are automatically rebuilt when some devices are switched off or moved. Each router is equipped with one module of SI-SRR receiver so it is able to record punches from BSF8-SRR and SIACs.

4. Current solutions for live transmission of results

The recommended power supply for each router device is the use of three AA alkaline batteries which can provide up to 32 working hours. The gateway device is always connected to a PC and it is powered through the USB. The devices have plastic housings with locking clips and silicone seals that should provide protection from wind, rain and snow.



Figure 4.6: An example of jSh Radio device. Source: [Har20]

Data transmission between routers and gateway takes place on radio frequency 869.525 MHz. By respecting certain regulations of the European Union, this band can be used as a license-free radio band by short-range devices. The jSh Radio uses a listen-before-talk method to avoid transmission collisions and transmits with a maximum power of 27 dBm. The distance for reliable transmission between a pair of devices depends on the complexity of the terrain along the transmission path. Without any obstacles, the maximum range can reach 2km, but the forest reduces the range to about 500 meters and at high tree density even to 250 meters. So the proper placement of routers in the forest plays a big role in the efficient functioning of the network.

The punch data is transmitted over the network without any changes, so the gateway output data is the same as the SPORTident. For any other network configuration or testing, jSh controller software can be downloaded from the project website [Har20].

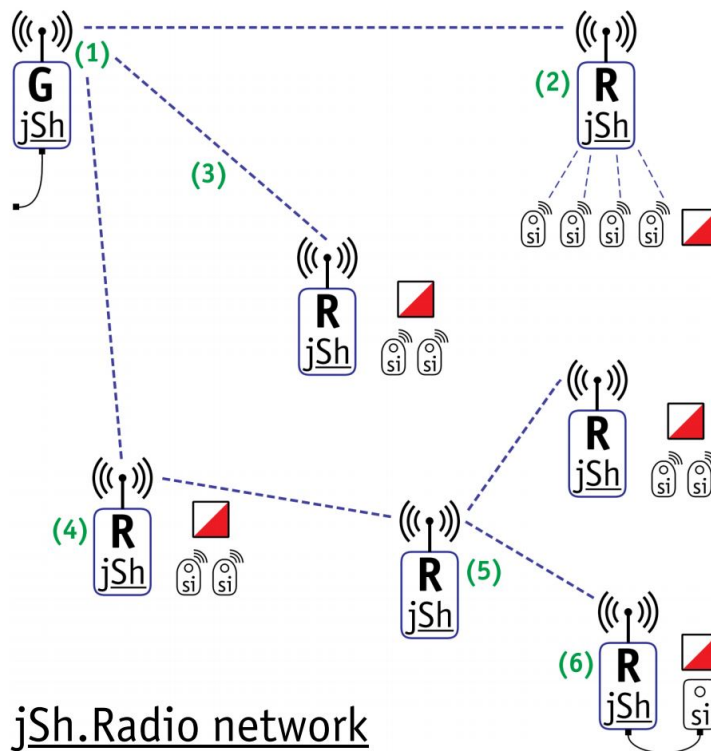


Figure 4.7: jSh mesh radio network. Source: [Har20]

4.5 Tinymesh project

The main goal of this project was to develop a lightweight radio solution that can be easily deployed in a forest. The project was created in 2016 by the GitHub.com user Plengqui and is freely available on [L20], including the technical description and codes. The project is based on a multi-hop mesh protocol with bidirectional wireless communication called Tinymesh by Radiocrafts [Rad20]. The Tinymesh is self-configuring and self-healing mesh network technology implemented in small compact modules. The structure of the network is the same as the jSh Radio network described in Section 4.4. It consists of multiple routers and one gateway and all the data from routers are automatically forwarded to the gateway. Tinymesh module used in this project is RC1701HP-TM. This module operates in 169 MHz frequency band, which is a license-free band in the European Union, with a maximum effect of 500 mW. Operating supply voltage of RC1701HP-TM is from 2.8 to 3.6 V with maximal current consumption around 400 mA.

Devices called radio units (Figure 4.8) are placed at each control point. These radio units are the routers of the Tinymesh network. The radio unit is



Figure 4.8: The radio unit by Plengqui. Source: [L20]

equipped with one RC1701HP-TM and one module of the SI-SRR receiver and controlled by a 32-bit microcontroller board Teensy-LC. Each radio unit is connected to 1.3 m long roll-up slim jim antenna (Figure 4.9) which should provide up to 1.5 km distance between a pair of units in the forest. All received punch records are buffered by the microcontroller and forwarded over the network to the gateway. The radio unit is powered by 18650 Lithium-Ion battery over a voltage regulator that creates a voltage of 3.3 V.



Figure 4.9: A roll-up slim jim antenna. Source: [L20]

The gateway device was made of the Tinymesh Demo Kit RC1701HP-TM-DK which can be easily connected to a PC via USB cable. The developer also created an application in Python 3 that enables to forward received punch records to an evaluation system. However, one of the future goals of the developer is to adapt the radio unit so that it can also serve as a gateway.

4.6 Racom

Racom [Rac20] is a company from the Czech Republic that focuses on the development and production of equipment for wireless data transmission. After its founding in 1989, the company manufactured equipment for amateur radios. Nowadays, Racom's main products are radio modems, GPRS/EDGE/UMTS routers and microwave links. Using its own technologies, Racom also provides live transmissions of results not only for orienteering but also for long-distance running, cycling, cross-country skiing and more. Racom system is a very common choice for major orienteering competitions in the Czech Republic with over twenty years of experience and high reliability. Especially at competitions with live television broadcasting where split times from control points are shown live along with the video. For example, the Racom system was used at the 2016 European Orienteering Championships in Jeseník, Czech Republic.

To ensure the transmit of punch data from measured control points during orienteering events, Racom uses a self-configuring mesh network consists of its own devices. Since 2018, these devices have been equipped with RipEX modems operating in the 400 MHz frequency band. An external antenna is connected to each device using a coaxial cable. The antenna is mounted, for example, higher on a tree and directed towards the centre of the race. To capture punch data, the SI-SRR receiver modules are connected to each device using the RS232 standard. The device is powered by a large battery that lasts the entire race. Due to the complexity of some terrains, data can be transmitted to the centre of the event using several radio hops. Each radio modem can act as a gateway station, retransmission, end station, or all of these roles at once. The data is forwarded over the network using UDP/IP packets to ensure high speed of data transmissions over the network. The gateway station located in the centre is connected to a local server which processes all incoming data. Audiences can access live results over a local WiFi network, or the live results can be uploaded to remote servers using Racom's LTE routers.



Figure 4.10: Racom device with the external antenna mounted on a tree. Source: [Rac20]

Chapter 5

Proposed solution

This chapter provides a detailed description of the design of the device that allows live transmissions of results of orienteering competitions. The solution proposed in this thesis is based on open-source software and hardware using the knowledge gained through the research from Chapter 4. The great emphasis was placed on the simplicity of the solution so that the average user of microelectronics could assemble the device. The selection of individual components was based on their low price, availability and size of their user base. An important factor was also the simplest possible deployment of the device during the orienteering event.

5.1 Description of the solution

Design of the solution of this thesis uses for live results transmissions a cellular communication network. As a result, only one stand-alone battery-powered device placed at the measured control points is required for transmissions of data. Punch records are transmitted by the device directly to a remote server where they are processed. The device consists of four main parts - a control unit (5.2), one or two SI-SRR receiver modules (5.3), module providing data transmission (5.4) and power supply (5.5). Figure 5.1 shows the first prototype of the device installed into a plastic suitcase. The suitcase provides the device with protection against rain, wind or other mechanical damage and it is easily portable thanks to the handle. The dimensions of the suitcase are 32x20x8 cm. A breadboard and a battery holder are glued to the bottom of the suitcase. All parts are wired using a breadboard which

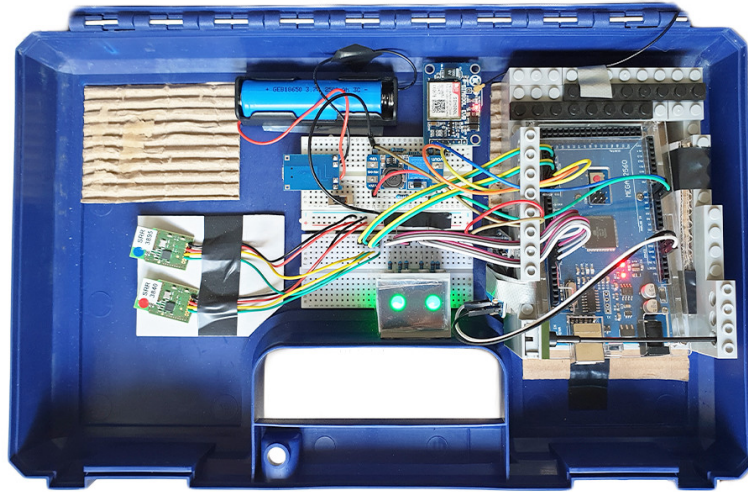


Figure 5.1: The first prototype of the device installed into a plastic suitcase

allows an easy access to individual components and their eventual replacement during development. A special holder made of lego bricks is also glued to the bottom and it allows easy replacement of the control unit. The external appearance of the suitcase is shown in Figure 5.2. The functionality of the device described in this chapter is similar to the SIGSM-DN device from Section 4.2. However, the solution of this thesis is based on open-source hardware components that allow the organizers of orienteering competitions to build the device at home and customize it to their specific needs. The wiring diagram of the device is shown in Figure 5.3. Optional device extensions are two RGB LEDs that indicate battery voltage status and cellular network strength. The logic of the program running on the device is described in Section 5.2.1.



Figure 5.2: Plastic suitcase for the device

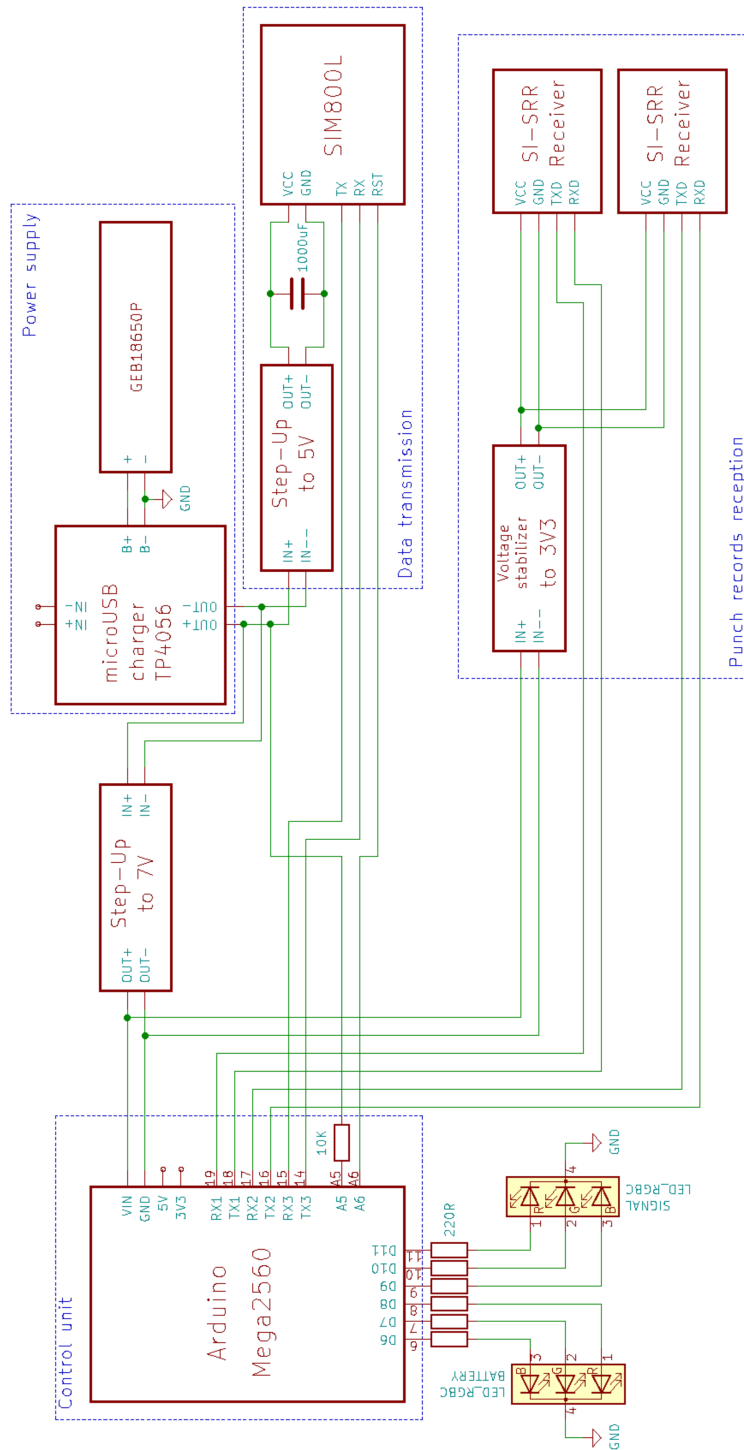


Figure 5.3: The wiring diagram

5.2 Control unit

The Arduino Mega2560 is an 8-bit single-board microcontroller based on the ATmega2560 microprocessor from Atmel. Most importantly, it is equipped with four universal asynchronous receiver-transmitters (hereafter UART) that allows asynchronous serial communication. This gives the device an ability to continuously receive data from the two SI-SRR receiver modules at the same time. Recommended input voltage for the Arduino Mega2560 is 7 to 12V. The ATmega2560 has an operating voltage of 5V, 256 KB of flash memory and 8 KB of static RAM. It is also equipped with 16 analog input pins and 54 digital input/output pins which gives plenty of space for connecting several devices without the need to use a bus. The Arduino Mega2560 also has a USB connection and a power jack that is helpful during development. More specifications can be found in [Ard20b].

The final solution described in Section 5.1 uses a clone of the Arduino Mega2560 (Figure 5.4) because of its significantly lower price. The clone has the same specifications as the original and no additional hardware definition is required.

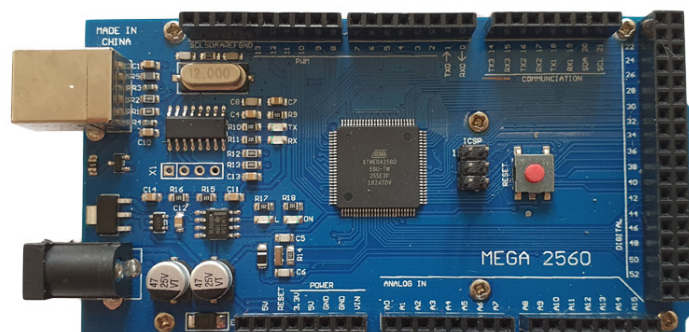


Figure 5.4: Clone of Arduino Mega2560

■ 5.2.1 Programming of the device

All codes for the device were written in C programming language using the Arduino Integrated Development Environment (hereafter IDE) [Ard20c]. The Arduino IDE is an open-source cross-platform application written in C and C++. Its source code is released under the GNU General Public Licence [Ard20a]. It supplies a software library and hardware definitions of Arduino boards so the written code can be compiled and uploaded to any Arduino board directly in the application. Codes written in the Arduino IDE are called sketches and stored in *.ino file type. Written code only requires two basic functions - setup and loop. The setup function is run once before starting the loop function which is the main program loop. These are compiled and linked into an executable cyclic executive program which is converted into a text file in a hexadecimal encoding. The hexadecimal file is then loaded by a loader program in the Arduino board firmware.

The program that was written for this thesis consists of sketches - *main.ino*, *SRRReceiversHandler.ino* (Section 5.3), *SIM800LHandler.ino* (Section 5.4.1) and optional *RGBLedHandler.ino* and *ESP8266Handler.ino* (Section 5.4.3). Codes are listed in Appendix B.

Listing 5.1 shows the *main.ino* file. The header of this file consists of definitions of the device name, serial ports, and required baudrates. The setup initialize all modules by executing their initialization functions - *SRRReceiversInit* ((Listing 5.3)) and *simInit* (Listing 5.7). After the setup, the loop function repeatedly calls function *checkSRR*. This function checks if a punch record has been received by the SI-SRR receivers. If so, the received punch record is sent to a remote server. Each received punch record is sent separately. The *checkSRR* function is described in more details in Section 5.3. Every 60 seconds, the loop function also checks the battery voltage and cellular signal strength. Functions *signalColor* and *batteryColor* indicate measured values by colouring the RGB LEDs and function *simSendStatus* sends the values to a remote server. This allows the user to monitor the device throughout the race. The functions for controlling the RGB LEDs are listed in the *RGBLedHandler.ino* file. The *simSendStatus* is described in Section 5.4.1.

Listing 5.1: Code of *main.ino* file

```

#define DEVICE_ID "DEVICE1" // Name of the device

#define BlueChannel Serial1 // SI-SRR on BLUE channel
#define RedChannel Serial2 // SI-SRR on RED channel
#define SimSerial Serial3 // Serial port for SIM800L

#define SRR_BAUDRATE 38400
#define SERIAL_BAUDRATE 115200
#define SIM_BAUDRATE 115200

// Timer for signal and battery level check
unsigned long ledTimer = 0;

void setup() {

    // Begin serial communication board and Arduino IDE
    Serial.begin(SERIAL_BAUDRATE);
    Serial.println("Initilizing Serial Monitor");
    delay(1000);

    simInit(); // Initialization of SIM800L
    SRRReceiversInit(); // Initialization of SI-SRR receivers
    batteryColor(); // Battery voltage check
    delay(2000);

    Serial.println("----- STATION IS READY -----");
}

void loop() {

    // Check of Rx buffer of SI-SRR receivers
    checkSRR();

    // Check signal and battery level every 60s
    if((millis() - ledTimer) > 60000){
        signalColor();
        batteryColor();
        simSendStatus(); // send device status to server
        ledTimer = millis();
    }
}

```


5.3 Punch data detection

SPORTident offers its SI-SRR receivers as fully functional SRR dongles or as modules for better integration into hardware projects (Figure 4.2). The SRR dongles have a USB interface for easy connection with PC or other equipment. One SRR dongle is able to manage up to 8 BSF8-SRR stations at once. The case of the SRR dongle can be easily opened. The image of the interior (Figure 5.5) shows that the SRR dongle consists of a USB controller and the module of the SI-SRR receiver. Hence, the SI-SRR receiver module is possible to obtain by dismantling the purchased SRR dongle.



Figure 5.5: Interior of the SRR dongle

Figure 5.6 shows the module of the SI-SRR receiver. It can be seen that the module is equipped with a mixed-signal microcontroller M430F2370 and low-power 2.4 GHz RF Transceiver CC2500 by Texas Instruments. The sticker indicates which channel the module was pre-programmed to. Any configuration of the module can be done by connecting it to a PC using a USB to TTL converter. Figure 5.6 also shows four wires led out of the module. The black wire is the ground and the red wire is for the 3.3 V power supply. According to the SRR dongle datasheet from [SPO20], the current consumption of the module is 30 mA. The other two wires are designed for serial communication - green is Rx and yellow is Tx. Data received by the module are automatically forwarded.

Proposed solution of this thesis uses two SI-SRR receiver modules to detect any punch record of the measured control point whether the record is sent



Figure 5.6: Module of the SI-SRR receiver

from BSF8-SRR station or from SIACs. One receiver records punches on the BLUE channel and the other on the RED channel. Both receivers must be powered by at least 3.3 V and require a DC current of 30 mA each. Therefore, the power cannot be supplied directly from the Arduino Mega2560 as its 3.3 V pin can provide only 50 mA DC. For this reason, the receivers should be powered by a battery using a voltage stabilizer that can provide them with the required current. During testing, the receivers were also powered directly from a battery with a voltage of about 3.7 V without causing any damage. This option is not recommended with any guarantee. The power supply is described in more details in Section 5.5. Both receivers are connected to the Arduino Mega2560 UARTs - blue to serial port 1 (pins Rx1 and Tx1) and red to serial port 2 (pins Rx2 and Tx2). The wiring is shown in Figure 5.3. All received punch records are automatically forwarded to the Arduino Mega2560 via one of the serial ports and stored in the serial receiver's buffer.

File *SRRReceiversHandler.ino* contains all the functions that work with SI-SRR receivers and punch records. Listing 5.2 shows header of *SRRReceiverHandler.ino* file. The global variable *punchSize* is set to 20 which is the length of one punch record in bytes. The two unsigned chars *dataBlue* and *dataRed* then allocate memory for one received punch by each SI-SRR receiver. Communication between the control unit and SI-SRR receivers is established by the *SRRReceiversInit* function (Listing 5.3). The baud rate of serial ports is set to 38400.

Listing 5.2: Header of *SRRReceiversHandler.ino* file

```
int punchSize = 20; // Size of a~punch record in bytes

// memory allocation for one punch record on each channel
unsigned char dataBlue[20];
unsigned char dataRed[20];
```

Listing 5.3: *SRRReceiversInit* function

```
void SRRReceiversInit(){  
  
    Serial.println("Initializing SI-SRR Receivers");  
    BlueChannel.begin(SRR_BAUDRATE); // Blue channel init  
    RedChannel.begin(SRR_BAUDRATE);  // Red channel init  
    delay(500);  
}
```

The individual bytes of a punch record are marked as follows from the beginning - STX, 0xD3, 0x0D, CN1, CN0, SN3, SN2, SN1, SN0, TD, TH, TL, TSS, MEM2, MEM1, MEM0, CRC1, CRC0, ETX | NAK. For the purposes of this thesis, only bytes CN0 to TL are important. CN0 is a code number of a control station. The identification number of a SPORTident card is stored in bytes SN3 to SN0. The first bit of TD byte determines whether the time of the record is A.M. or P.M. Finally, the TH and TL bytes store the time of the punch record as the number of seconds per half day. Received punch record is decoded by the *parsePunchData* function (Listing 5.4). The data is given to the function as an input variable called *binaryData*. The function then separates the bytes with the relevant information and stores their integer values in temporary variables.

Listing 5.4: *parsePunchData* function

```

void parsePunchData(unsigned char *binaryData, int &len,
                   unsigned int &StationNumber,
                   unsigned long &SICardNumber,
                   unsigned int &hours,
                   unsigned int &minutes,
                   unsigned int &seconds,
                   unsigned int &totalSeconds){

    // byte CNO
    StationNumber = (unsigned int)binaryData[5];

    // bytes SNO, SN1, SN2, SN3
    SICardNumber = (unsigned long)binaryData[9] |
                  ((unsigned long)binaryData[8] << 8) |
                  ((unsigned long)binaryData[7] << 16) |
                  ((unsigned long)binaryData[6] << 24);

    // Time of a~punch record - bytes TH, TL
    totalSeconds = binaryData[12] | (binaryData[11] << 8);
    seconds = totalSeconds % 60;
    minutes = (totalSeconds / 60) % 60;
    hours = totalSeconds / (60*60);

    // byte TD
    boolean isPM = (binaryData[10] & 1);
    if (isPM){
        hours += 12;
    }
}

```

As it was mentioned in Section 5.2.1, the main program loop repeatedly calls function *checkSRR*. The only task of the *checkSRR* is to call function *readSRRSerialPorts* (Listing 5.5). The *readSRRSerialPorts* function has three inputs - two pointers to the allocated punch record memory and the length of one punch record. The function checks if any data is available in the serial buffer for the blue channel SI-SRR receiver. If so, it tries to load 20 bytes (one punch record) of the data into the *dataBlue* variable. The number of actually loaded bytes is stored in variable *len*. Only if the value of *len* is equal to 20, one complete punch record has been stored, the *sendParsedPunchData* function is called. Inputs of the called function are the variables *dataBlue* and *len*. The same process is then performed for the red channel receiver.

Listing 5.5: *readSRRSerialPorts* function

```

void readSRRSerialPorts(unsigned char *dataBlue,
                        unsigned char *dataRed,
                        int &punchSize){

    int len = 0; // temporary length

    // Read data from BLUE channel receiver buffer
    if(BlueChannel.available())
    {
        Serial.println("BLUE data received.");
        len = BlueChannel.readBytes(dataBlue, punchSize);
        sendParsedPunchData(dataBlue, len);
    }

    // Read data from RED channel receiver buffer
    if(RedChannel.available())
    {
        Serial.println("RED data received.");
        len = RedChannel.readBytes(dataRed, punchSize);
        sendParsedPunchData(dataRed, len);
    }
}

```

The *sendParsedPunchData* function first executes the *parsePunchData* function (Listing 5.4) which stores the relevant data of the received punch record in temporary variables. Subsequently, a text string is created. The structure of the string is determined by the Application Programming Interface (hereafter API) of the remote server used by a user. The example API in *SRRReceiversHandler.ino* file corresponds to a web application (Section 5.4.2) that was created to test the device within this thesis. The last step of *sendParsedPunchData* is to execute the *simSendData* function which sends the received punch record to the remote server. A detailed description of the *simSendData* is in Section 5.4.1.

up to 2 A. The current consumption during GPRS mode is around 450 mA. The SIM800L is very sensitive to voltage drops during which it constantly shuts down. Therefore, the proper power must be provided by a battery that can provide the correct voltage during peaks (see Section 5.5). At the same time, it is also necessary to place a sufficiently large capacitor, as close as possible to the SIM800L, to cover any voltage drops.

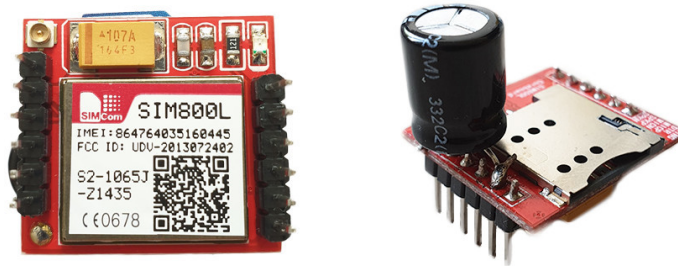


Figure 5.7: SIM800L module equipped with an electrolytic 1000uF capacitor

Figure 5.7 shows a version of a SIM800L module. The module has data pins for power supply, serial communication, ring, microphone and speaker. It is also equipped with a socket for micro SIM card and a u.fl connector for attaching an external antenna. For easier use, it is better to insert a SIM card with disabled PIN code. After powering up, the module automatically registers and connects to the cellular network. Successful registration in the cellular network is indicated by flashing LED every 3 seconds. After several tests, it was found that the surface-mount capacitor on the module is not sufficient to maintain the correct voltage and the module switches off repeatedly. For this reason, an electrolytic capacitor with 1000 uF capacity was soldered to the module between pin GND and VIN (see Figure 5.7). After this adjustment, the module was able to connect to the cellular network. One of the disadvantages of this module is that it is not equipped with any voltage stabilizer and any input voltage higher than 4.4 V can permanently damage the module. The Arduino Mega 2560 has a logic voltage of 5 V so the module must be connected to it over a voltage divider.

Another version of a SIM800L module is shown in Figure 5.8. It is equipped with a built-in voltage stabilizer. The stabilizer allows direct connection of the module to the Arduino Mega2560. The module is also powered with 5 V. After powering up, it was found that the surface capacitor on this module is able to maintain all voltage drops. The solution of this thesis works

with both mentioned modules in the same way. The module with a built-in voltage stabilizer (Figure 5.8) was chosen for the finale prototype described in Section 5.1 due to easier wiring. Furthermore, the SIM800L module is the one with a built-in voltage stabilizer. The module is connected to a small GSM SMA antenna via the u.fl to SMA adapter.

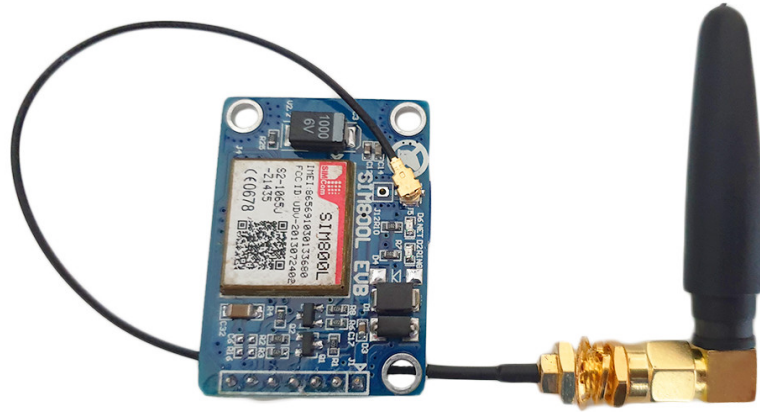


Figure 5.8: SIM800L module with a surface-mount 1000uF capacitor, 5V voltage stabilizer and an external antenna

The wiring of the SIM800L module is shown in Figure 5.3. The RST pin serves for resetting the module. Pins TXD and RXD of the SIM800L module are connected to the serial port 3 of the Arduino Mega2560 and the baud rate is set to 115200 bps. The module can be controlled by sending special AT commands over the serial link. All supported AT commands can be found in the technical specifications of the SIM800L chip [SIM20].

File *SIM800LHandler.ino* contains all the functions that work with SIM800L module. Listing 5.6 shows a header of *SIM800LHandler.ino* file. The first command includes the *SIM800L HTTP Connector* library by Olivier Staquet. This open-source library can be found on [Sta20] and it can be installed via Arduino IDE library manager. The library allows the user to avoid sending AT commands directly. The library is related only to HTTP or HTTPS connections using the GET and POST methods. After including the library, the header defines a reset pin and initialize the SIM800L module. Then there are settings of variables for GPRS connection. *APN* variable is an access point name which is always defined by the cellular network provider. *URL* variable is a network path to a remote server for receiving punch records. *URLStatus* variable is a network path to a remote server for receiving device status records. *CONTENT_TYPE* variable defines an encoding of the data.

Listing 5.6: Header of *SIM800LHandler.ino* file

```
#include "SIM800L.h"

#define SIM800_RST_PIN 6 // Reset pin for SIM800L
SIM800L* sim800l;

// information for GPRS connection - user defined
const char APN[] = "internet"; // define by network provider
const char URL[] = "www.testapp.tk/api/post_punch.php";
const char URLStatus[] = "www.testapp.tk/api/post_status.php";
const char CONTENT_TYPE[]="application/x-www-form-urlencoded";
```

The communication between the control unit and the SIM800L is established by the *simInit* function (Listing 5.7). The baud rate of the serial port is set to 115200. The *simInit* function also initializes a driver for the SIM800L module with an internal buffer of 200 bytes and a reception buffer of 512 bytes. The last step of *simInit* is the execution of the *setupSIM800L* function that is listed in *SIM800LHandler.ino* file. The *setupSIM800L* first checks if the control unit is able to communicate with the SIM800L module. If so, the function checks the signal strength and the successful registration to the cellular network. Then the function sets the access point name for the GPRS connection. The signal RGB LED flashes red during the entire SIM800L module setup process. After the setup, the signal RGB LED is coloured according to the signal strength. Red means marginal or no signal, orange is a sufficient signal and green is a very good signal.

Listing 5.7: *simInit* function

```
void simInit(){

  Serial.println("Initializing SIM800L");
  SimSerial.begin(SIM_BAUDRATE);
  delay(500);

  // Initialize SIM800L driver with
  // an~internal buffer of 200 bytes
  // and a~reception buffer of 512 bytes, debug disabled
  sim800l = new SIM800L((Stream *)&SimSerial,
                       SIM800_RST_PIN,
                       200, 512);

  // Setup SIM800L for GPRS
  setupSIM800L();

}
```

Listing 5.8: *simSendData* function

```
void simSendData(char * postData){  
  
    openConnection();  
    simPrintToClient(postData, URL);  
    closeConnection();  
}
```

The *simSendData* function (Listing 5.8) is called from the *sendParsedPunchData* function described in Section 5.3. The *simSendData* calls three functions sequentially. The *openConnection* function creates a GPRS connection. Then, the *simPrintToClient* function performs an HTTP POST with the input data to the given URL address. Successful HTTP POST is indicated by a single flash of the signal RGB LED. Finally, the *closeConnection* function closes the GPRS connection. The maximum timeout for an HTTP POST is set to 10 seconds. One HTTP POST should not be longer than 2 seconds depending on the signal strength. The *simSendStatus* function is similar to the *simSendData* function. The function measures the values of battery voltage and cellular signal strength and sends them to a remote server. As stated in Listing 5.1, device status data is sent every 60 seconds. This allows the user to monitor the device throughout the race.

Serial buffer expansion

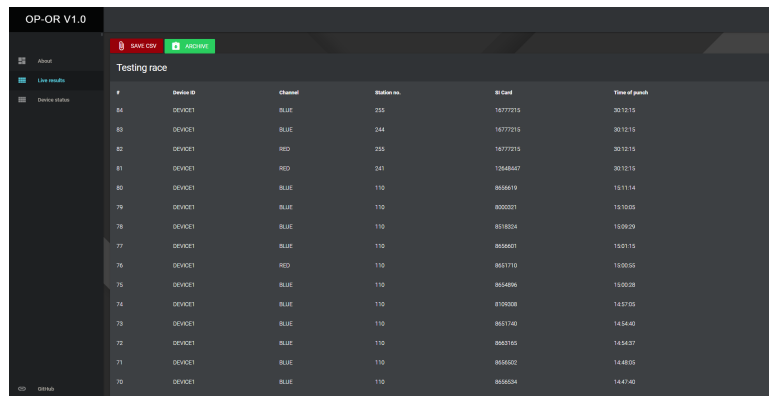
During the testing of the device, it was found that the duration of one HTTP POST is between 1 and 2 seconds. And it can be even more if the signal is weak. Punch records are sent one at a time to reduce data loss during erroneous transmission. The ATmega2560 is a single-core processor so the control unit can perform only one process at the time. For this reason, the SI-SRR receivers are connected to the UARTs. Each received punch record is stored in a serial receiver buffer until the data is read by the *readSRRSerialPorts* function (Listing 5.5). The default size of the Arduino Mega2560 serial receiver buffer is 64 bytes. The size of one punch record is 20 bytes so only 3 complete punch records can be loaded into the serial receiver buffer. However, a measured control point can be punched by more than three competitors at once during some races. In this case, the serial receiver buffer could be overflowed and punch records could be lost. Therefore, the size of the serial receiver buffer has been expanded from the default 64 to 256 bytes. This change can be made in the Arduino Mega2560 hardware definition in Arduino IDE files by changing the *SERIAL_RX_BUFFER_SIZE* value to the required 256. It is important to pay attention to the size of the RAM of the control unit. The Arduino Mega2560 has 8 kilobytes of RAM, which

gives plenty of RAM, even if the serial receiver buffers of SI-SRR receivers are full. With 256 bytes of space, the serial receiver buffer can handle up to twelve complete punch records. The serial receiver buffer can be further increased according to the needs of the user.

■ 5.4.2 Web application for testing

A web application was created for the purpose of testing the device. Web application codes are listed in Appendix B. The API of the application consists of two functions - *post_data_from_forest.php* and *post_status.php*. The *post_data_from_forest.php* receives punch records from the device and insert them to a MySQL database. Punch records must be sent using the HTTP POST method. The content type of the data is *application/x-www-form-urlencoded*. Acceptable field names are *device_id*, *channel*, *station_number*, *SI_card*, *hours*, *minutes*, *seconds*. Values of these fields are inserted into the table of the MySQL database named *test_race*. The *post_status.php* receives the device status data. Acceptable field names of the *post_status.php* are *device_id*, *signal* and *battery*. Values of these fields are inserted into the *status_table*.

The appearance of the website was created using the open-source front-end framework *Material Dashboard Lite* [IT20] from Creative IT. The website consists of four pages - *index.php*, *punches.php*, *device-status.php*, *live-results.php*. The *index.php* displays some basic information about the project. The *punches.php* displays the content of the *test_race* table. The *punches.php* offers the possibility to download the *test_race* table in CSV format and archive the current table in the database in order to start a new measurement. The *device-status.php* displays the last record of the *status_table* with the timestamp of the record. The last extension of the web application was the addition of the *live-results.php* which displays received punch records with the name of registered orienteering runners from ORIS database [Če20a].



Rank	Device ID	Channel	Station no.	BSSID	Time of finish
84	DEVICE1	BLUE	255	16777215	30.1215
83	DEVICE1	BLUE	244	16777215	30.1215
82	DEVICE1	RED	255	16777215	30.1215
81	DEVICE1	RED	241	12648627	30.1215
80	DEVICE1	BLUE	110	8668819	15.1114
79	DEVICE1	BLUE	110	8000201	15.1000
78	DEVICE1	BLUE	110	8118024	15.0929
77	DEVICE1	BLUE	110	8058801	15.0115
76	DEVICE1	RED	110	8681710	15.0000
75	DEVICE1	BLUE	110	8644096	15.0028
74	DEVICE1	BLUE	110	8100008	14.8700
73	DEVICE1	BLUE	110	8613740	14.8440
72	DEVICE1	BLUE	110	8063165	14.8437
71	DEVICE1	BLUE	110	8668802	14.8000
70	DEVICE1	BLUE	110	8668824	14.8740

Figure 5.9: Screenshot of testing web application

5.4.3 Local WiFi network

One of the alternative options for connecting the device to the Internet is to use a local WiFi network. Connection to the local WiFi network is enabled by the ESP-01 module (Figure 5.10). The ESP-01 is equipped with the ESP8266 microchip produced by Espressif Systems. The ESP8266 is a low-cost WiFi 32-bit microcontroller with a full TCP/IP stack. The ESP8266 supports AT commands that can be found in the technical specifications [Sys20].

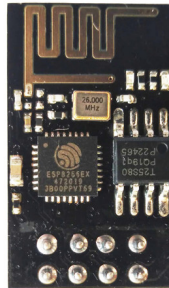


Figure 5.10: ESP-01 module with ESP8266 microchip

The ESP-01 must be powered with a voltage of 3.3 V via pins 3V3 and GND. The rest of the ESP-01 module wiring may be similar to the SIM800L. Pins TXD and RXD are connected to the serial port 3 of the Arduino Mega2560. The baudrate is set to 115200 bps. The RST pin serves for resetting the module. Unlike the SIM800L, the ESP-01 module also has an EN pin. The ESP-01 module only switches on if the EN pin has a logic value of 1.

File *ESP8266Handler.ino* contains all the functions that work with the ESP-01 module. The *WiFiEsp* library by Bruno Portaluri was used for the solution. The *WiFiEsp* is an open-source Arduino library that can be found on [Por20]. The *ESP8266Handler.ino* contains the *espSendData* function which can directly replace the *simSendData* (Listing 5.8) function and send data to a remote server using the HTTP POST method. The device equipped with the ESP-01 module can be used only in places where a local WiFi network with an Internet connection is available. The required WiFi network can be created in the centre of an orienteering event or at the measured control point using the mobile hotspot feature of a smartphone.

Figure 5.11 shows a device with the ESP-01 module. On the left side of the box is a holder with a smartphone. The smartphone creates a local WiFi network for the device.

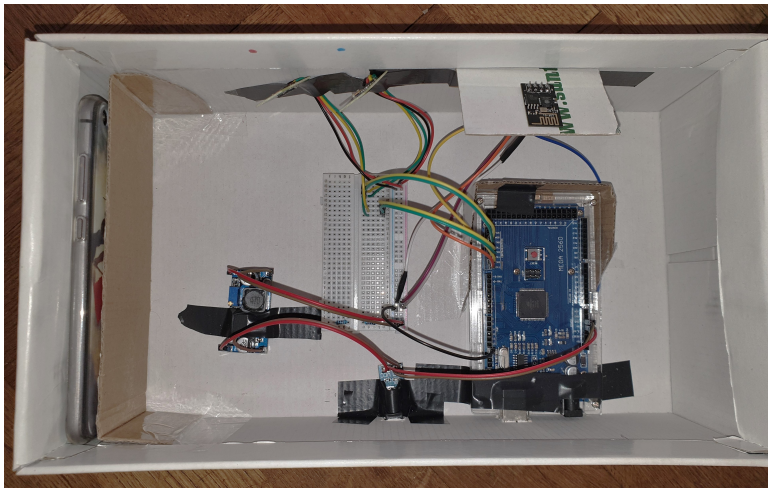


Figure 5.11: A device with the ESP-01 module

■ 5.5 Power supply

Ensuring sufficient power is one of the most important things for the proper functioning of the device. The device is powered by a lithium-ion battery (hereafter *Li-Ion battery*). The specific Li-Ion battery model used in the prototype in Figure 5.1 is the GEB18650P. The GEB18650P has a capacity of 2500 mAh and a nominal voltage of 3.7 V. The maximum surge discharge current of the GEB18650P is 12,500 mA so it can provide a sufficiently large current for all parts of the device, especially for the SIM800L module. The operating voltage is from 2.8 to 4.2 V and the charging voltage is 4.2 V. The GEB18650P has a standard 18650 size of Li-Ion battery and weighs about 45 grams. The advantage of Li-Ion batteries is that they have no memory

effect and their service life should be greater than 500 cycles. The disadvantage is that they are very sensitive to full discharge and overcharging. When the voltage of a battery falls below 2.8 V, it is almost impossible to recharge the battery. Overcharging the Li-Ion battery can damage it by overheating and can explode. For this reason, each Li-Ion battery must be equipped with electronics protection. Figure 5.12 shows a module equipped with TC4056A which is 1A linear Li-Ion battery charger. The GEB18650P is connected to the charger via pins B+ and B- and the device is connected via pins OUT+ and OUT-. The wiring diagram is in Figure 5.3. The GEB18650P can be charged via the micro USB interface on the module. The input voltage of the charger is from 4.5 to 5.5 V and the charging current is 1 A. The module ensures that the battery voltage does not drop below 2.8 V during discharging by disconnecting the powered device. The charging process is stopped at 4.2 V.

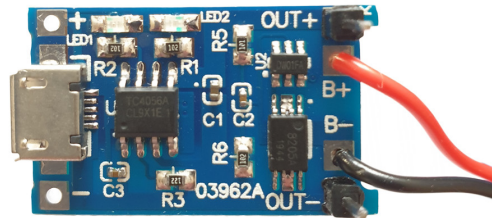


Figure 5.12: Micro USB charger for Li-Ion batteries

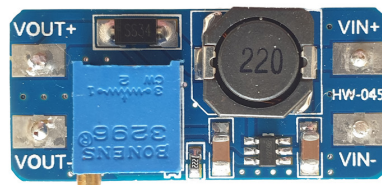


Figure 5.13: Step-Up module

Two DC to DC step-up modules from Figure 5.13 were used to increase the battery voltage to the required values. A detailed description of the step-up module can be found in [ECL20a]. The input voltage of the step-up module is from 2 V to 24 V. The maximum output voltage is 28 V and the maximum output current is 2 A. The first step-up module increases the battery voltage to 5 V which is used to power the SIM800L module. The second step-up module increases the battery voltage to 7 V. The second step-up module supplies the Arduino Mega2560 and over a voltage stabilizer

also supplies the SI-SRR receiver modules. The voltage stabilizer [ECL20c] is able to stabilize the voltage of 4.5 - 7 V to 3.3 V. The battery charge level can be measured on the Arduino Mega2560. The positive battery terminal is connected to analogue input pin A5 via a 10K resistor. The value of the measured voltage is determined from the exact value of the Arduino Mega2560 5V pin. For the accuracy of the calculation, the user must measure the voltage value of the board exactly. The *batteryColor* function colours the battery RGB LED according to the measured voltage. The *batteryColor* function is listed in the *RGBLedHandler.ino* file.

Chapter 6

Testing

This chapter is dedicated to testing the prototype device from Figure 5.1. Unless otherwise stated, the prototype device consists of the clone of the Arduino Mega2560, two SI-SRR receiver modules, the SIM800L module (Figure 5.8) with an external antenna and the GEB18650P Li-Ion battery. The results of all tests are discussed in Section 7.

6.1 Punch records reception test

This experiment tests the ability of the SI-SRR receivers to receive punch records. The SIM800L has been switched off throughout the experiment. Each received punch record was recorded only on the prototype device and was not sent to a remote server. One BSF8-SRR station and two SI-Cards were used to create the punch record transmissions. The BSF8-SRR was about 30 cm far from the SI-SRR receiver modules on the device. Both channels of SI-SRR receivers were tested separately and together. Individual tests were performed at two frequencies of control station punching - one punch per five seconds and two punches per one second. Table 6.1 shows the type of measurement, the number of created punch records (CP) and the number of received punch records on the device (DP).

Measurement	CP [-]	DP (BLUE + RED) [-]
Blue channel 1p/5s	50	50
Red channel 1p/5s	50	50
Both channels 1p/5s	50	58 (32 + 26)
Blue channel 2p/1s	50	50
Red channel 2p/1s	50	50
Both channels 2p/1s	50	58 (28 + 30)

Table 6.1: Punch records reception test.

6.2 Test of data upload to a remote server

This experiment tested the ability of the device to transmit received punch records to a remote server using the SIM800L module and the ESP8266 module. One BSF8-SRR station and two SI-Cards were used to create the punch records. Punch records were created at a frequency of one punch per five seconds. All punch records received by the device were sent to a remote server using the HTTP POST method. For the first measurement, the device was equipped with the ESP8266 module. A local WiFi network was created using a smartphone with a cellular network connection placed in a holder as it is shown in Figure 5.11. For the second measurement, the device was equipped with the SIM800L module. The cellular signal strength was -53 dBm. Table 6.2 provides a detailed overview of the experiment results. CP stands for the number of created punch records, DP stands for the number of punch received on the device and sent to a remote server and SP stands for the number of punch records received on the remote server. Numbers DP and RP include the duplicates received on both SI-SRR channels.

	CP [-]	DP [-]	SP [-]
ESP8266	100	115	115
SIM800L	100	122	122

Table 6.2: Results of the test of data upload to a remote server.

6.3 Testing at a measured training race

This experiment tests the overall functionality of the device in real conditions. The training race took place in the forest near nature reserve Toulcovy

Maštale in Pardubice Region. A SPORTident control station was placed at each control point. The BSF8-SRR station was placed at one control point. The suitcase with the device was placed approximately 1.5 metres above the BSF8-SRR station. This is shown in Figure 6.1 together with a training race participant punching the station.



Figure 6.1: Training race participant punches measured control point. Suitcase with the device is placed above the control point in the top left corner of the picture.

The participants of the training race were divided into two groups. The first group started during the morning and the second during the afternoon. There were large enough time gaps between the starts of the individual participants to avoid mutual contact between the participants. The measured control point was also punched several times by the author of this thesis. This was done in order to check the device performance and to increase the number of test punch records. These punches are hereinafter referred to as “virtual participants”. Each participant, real and virtual, punched the measured control point just once. Punch records were recorded by two SI-SRR receivers on BLUE and RED channel. One punch record received on both channels at the same time was counted as one punch record in the results. Live results could be monitored in a web application throughout the experiment. Each punch record was automatically assigned a participant name thanks to the connection to the database of registered orienteering runners from [Če20a]. The web application also allowed to monitor the internet connection of the device, the battery voltage and cellular signal strength. The device was to send this data at one-minute intervals. All data was saved for later processing.



Figure 6.2: Device placement at the measured control point

The morning group had 53 real and 10 virtual participants. However, only 31 punch records were received. This was caused by an error on the device that was having problems connecting to the Internet and eventually disconnected completely. The presumed reason was a wrong position of the device antenna. For this reason, the device was placed in a position where the antenna inside the suitcase is in a vertical position (Figure 6.2) before the start of the afternoon group. The gap in the records can be observed in the graphs from Figures 6.3 and 6.4. The afternoon group had 29 real and 10 virtual participants. The device was sending records without interruption and all 39 punch records were received. Table 6.3 provides a detailed overview of the number of received punch records. NP stands for number of participants, RP stands for number of received records on the remote server and BLUE and RED are the numbers of records received on blue and red channel including duplicates.

	NP [-]	RP [-]	BLUE [-]	RED [-]
Morning group	63	31	19	14
Afternoon group	39	39	35	8
Total	102	70	54	22

Table 6.3: Results of the test at the training race.

The graph in Figure 6.3 shows the changes in device battery voltage over time. This is called the process of discharging. The points marked on the graph are records of battery voltage values sent from the device to a remote server. The whole experiment lasted 306 minutes and 180 device status records were sent.

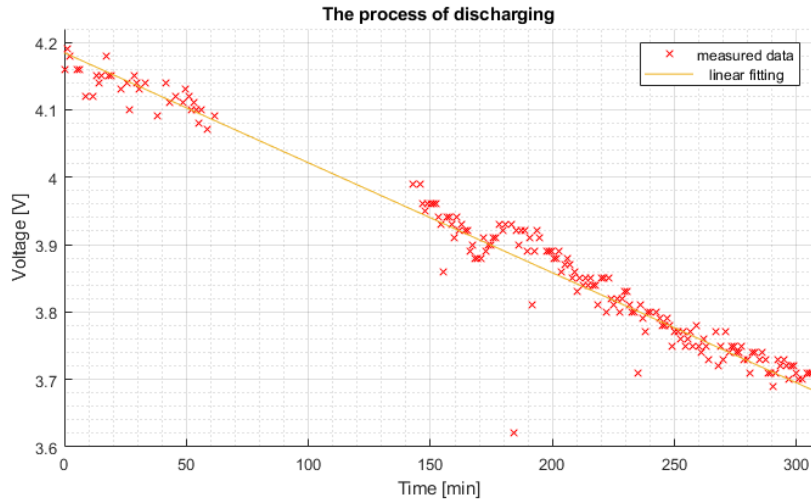


Figure 6.3: Graph of the battery discharge process

The graph in Figure 6.4 shows changes in values of cellular signal strength over time. The points marked on the graph are the cellular signal strength values measured on the device and sent to the remote server. The blue dashed line indicates the average value of the signal strength, which was calculated as the arithmetic mean of the measured values as -77 dBm.

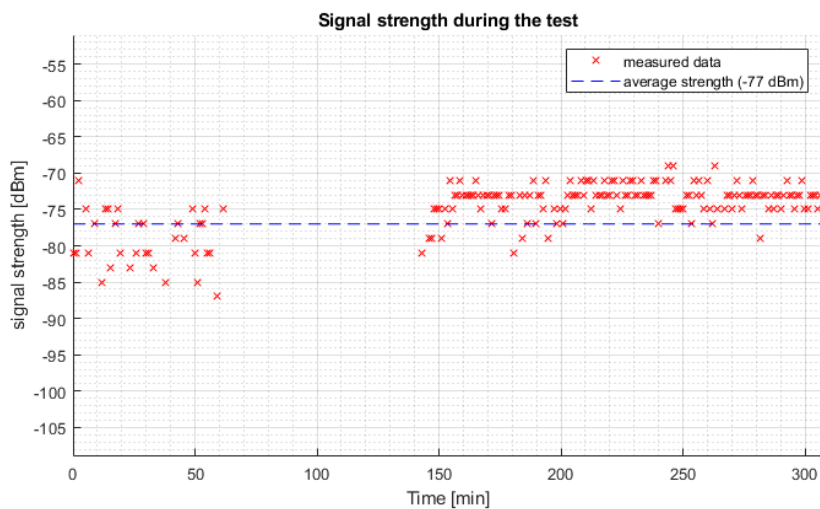


Figure 6.4: Graph of signal strength during the test

■ 6.4 Testing at the official competition

All orienteering competitions of the spring part of the 2020 season were cancelled due to the Covid-19 crisis. For this reason, it was not possible to test the device at any official competition.



Chapter 7

Discussion

The experiment from Section 6.1 tests the ability of the SI-SRR receivers to receive transmit punch records from the BSF8-SRR station. The testing of the BLUE and the RED channel separately produced the same results at both tested punching frequencies. Single SI-SRR receiver was able to receive all 50 created punch records. All 50 created punch records were also received at both testing punching frequencies if two SI-SRR receivers were receiving at the same time. The number of received punch records was even higher than the number of created punch records. This was caused by the reception of one punch record with both SI-SRR receivers. However, one SI-SRR receiver connected in pairs could not receive all the created punch records on its own, despite the 100% reception reliability of a single SI-SRR receiver. The measured results show that devices equipped with one or two SI-SRR receivers could be possible to use in individual races, where competitors punch a control point at least half a second apart. During a relay race, one control point can be punched by up to a dozen of competitors at a time. These conditions could not be simulated in the home laboratory environment. The SPORTident recommends using two connected SI-SRR receivers in this case. The reception reliability of two SI-SRR receivers in races with higher punching frequencies will have to be tested later in real conditions.

Experiment from Section 6.2 showed that using HTTP to upload data to a remote server is a reliable method. There were no data losses using the SIM800L and the ESP8266 module. On the other hand, the long transfer time from the SIM800L module using HTTP may be a disadvantage for future applications. The SIM800L is waiting for server confirmation, and one punch transmission can take up to two seconds. The solution for increasing the transmission speed could be the use of UDP but this could affect

which could be cheaper than the purchased one. The modularity of the proposed solution also allows the easy exchange of data transmission technology. A part of the future work will include testing the use of low-energy mesh radio networks which are described in Chapter 4 and possible interconnection of individual solutions.



Chapter 8

Conclusion

The device which can provide live transmission of results of orienteering competition has been created. The design has been created based on the research of the current solutions in Chapter 4. The design of the device is described in Chapter 5. All codes created for this work are listed in Appendix B. The created device has been tested in a real forest environment during a training race. The test results showed that the device is fully capable of receiving punch records of individual competitors and sending them to a remote server using a cellular network. The web application from Section 5.4.2 was fully capable of receiving and processing these punch records and displaying live results in real-time. The results of the tests are in Chapter 6 and they are discussed in Chapter 7. In addition, proposals for future work expansion are presented in Chapter 7 based on the newly acquired knowledge during the creation of this work.

Appendix A

Bibliography

- [Ard20a] Arduino, *Arduino IDE source code*, <https://www.github.com/arduino/Arduino>, 2020, [Online; accessed May 1, 2020].
- [Ard20b] ———, *Arduino Mega2560*, <https://store.arduino.cc/arduino-mega-2560-rev3>, 2020, [Online; accessed May 1, 2020].
- [Ard20c] ———, *Website homepage*, <https://www.arduino.cc/>, 2020, [Online; accessed May 1, 2020].
- [AS20] Emit AS, *Website homepage*, <https://www.emit.no/en>, 2020, [Online; accessed May 1, 2020].
- [BB20] O. Berg and E. Berg, *Radio Online Control*, http://olresultat.se/blog/?page_id=148, 2020, [Online; accessed May 17, 2020].
- [DU20] Deeday-UK, *The international orienteering symbol*, https://en.wikipedia.org/wiki/Orienteering#/media/File:Orienteering_symbol_framed.png, 2020, [Online; accessed May 17, 2020].
- [ECL20a] ECLIPSERA, *DC-DC 2A step-up module MT3608*, <https://arduino-shop.cz/arduino/1696-step-up-modul-napajeni-mt3608-2a-dc-dc.html>, 2020, [Online; accessed May 4, 2020].
- [ECL20b] ———, *Li-ion battery GEB18650P*, <https://arduino-shop.cz/arduino/5127-li-ion-akumulator-geb18650p-3-7-v-2500-mah.html>, 2020, [Online; accessed May 4, 2020].

- [Sys20] Espressif Systems, *ESP8266 Overview*, <https://www.espressif.com/en/products/socs/esp8266ex/overview>, 2020, [Online; accessed May 1, 2020].
- [Če20a] Český svaz orientačních sportů, *Informační systém ČSOS*, <https://oris.orientacnisporty.cz/>, 2020, [Online; accessed May 1, 2020].
- [Če20b] ———, *Směrnice - Ověřené systémy označování závodních průkazů*, <http://www.orientacnibeh.cz/upload/dokumenty/sekce-ob/smernice-razeni-19.pdf>, 2020, [Online; accessed April 2, 2020].
- [Če20c] ———, *Website homepage*, <http://www.orientacnisporty.cz/>, 2020, [Online; accessed May 17, 2020].

Appendix B

Content of enclosed CD

Listing B.1: Content of enclosed CD

```
CD:
|---juricja1_BP.pdf
|---arduino
|   |---ESP8266Handler.ino
|   |---main.ino
|   |---RGBLedHandler.ino
|   |---SIM800LHandler.ino
|   +---SRRReceiversHandler.ino
+---web_application
|   |---device-status.php
|   |---index.php
|   |---live-results.php
|   |---punches.php
|   |---api
|       |---post_data_from_forest.php
|       +---post_status_data.php
|   |---css
|   |---images
|   |---js
+---parts
```