CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF INFORMATION TECHNOLOGY

ASSIGNMENT OF BACHELOR’S THESIS

Title:	Time control system for archery competitions
Student:	Tomáš Beneš
Supervisor:	Ing. Jan Černý
Study Programme:	Informatics
Study Branch:	Computer engineering
Department:	Department of Digital Design
Validity:	Until the end of summer semester 2017/18

Instructions

1) Gain proper knowledge of relevant parts of the archery rules, order of shooting, time control, competition forms, and requirements for judges.
2) Select suitable electronic and mechanical parts and design circuits for archery time control system.
3) Construct a prototype and develop control firmware.
4) Test your design and firmware implementation with the use of appropriate testing methods and during an actual archery tournament.
5) Write technical documentation and user manual.

References

Will be provided by the supervisor.

doc. Ing. Hana Kubátová, CSc.
Head of Department

prof. Ing. Pavel Tvrdík, CSc.
Dean

Prague October 18, 2016

Czech Technical University in Prague
Faculty of Information Technology
Department of Computer engineering

Bachelor's thesis

Time control system for archery competitions

Tomáš Beneš

Supervisor: Ing. Jan Černý

12th May 2017

Acknowledgements

I would like to thank my supervisor Jan Černý for allowing me to participate in this project, for his support and his invested time.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended. In accordance with Article 46(6) of the Act, I hereby grant a nonexclusive authorization (license) to utilize this thesis, including any and all computer programs incorporated therein or attached thereto and all corresponding documentation (hereinafter collectively referred to as the "Work"), to any and all persons that wish to utilize the Work. Such persons are entitled to use the Work in any way (including for-profit purposes) that does not detract from its value. This authorization is not limited in terms of time, location and quantity. However, all persons that makes use of the above license shall be obliged to grant a license at least in the same scope as defined above with respect to each and every work that is created (wholly or in part) based on the Work, by modifying the Work, by combining the Work with another work, by including the Work in a collection of works or by adapting the Work (including translation), and at the same time make available the source code of such work at least in a way and scope that are comparable to the way and scope in which the source code of the Work is made available.

In Prague on 12th May 2017 . . . .

Citation of this thesis

Beneš, Tomáš. *Time control system for archery competitions.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2017.

Abstrakt

Cílem práce je seznámení se s celým procesem návrhu, sestrojení a programování lukostřelecké časomíry podle pravidel organizace World Archery. Následuje sestrojení prototypu, který je srovnán se současnými systémy. Prototyp byl úspěšně implementován a otestován na lukostřeleckých závodech a trénincích. Přínosem této práce je cenově dostupný systém pro malé lukostřelecké kluby a jednotlivce.

Klíčová slova časomíra, mikroprocesor, lukostřelba, návrh DPS, Wi-Fi

Abstract

The purpose of this thesis is to introduce process of designing, constructing and programming a time control for archery competitions according to World Archery rules. The constructed prototype is then compared with current commercial solutions available on market. System was successfully implemented and tested on archery competitions and training lessons. Contribution of this work is affordable time control for small clubs and individuals.

Keywords time control, micro-controller, archery, PCB design, Wi-Fi

Contents

List of Figures

List of Tables

Introduction

Archery is a human skill which has been in our society since prehistoric time. In today's society archery has changed it's original purpose from hunting and survival to competitive sport. Archery is an Olympic sport which is mainly about technique and concentration. That is why this thesis focuses on bringing asset to this specific, original and proud sport.

Time control system is a part and parcel of archery competitions and practice. However, commercial solutions are very expensive and not suitable for small clubs or individual archers. Small clubs are mostly trying to substitute these solutions with home-made ones which are either very old or missing some key features. Main goal of this work's focus is construction of time control using modern technologies and fulfilling all requirements of archery communities.

In the first part this thesis focuses on introduction to rules of competitive archery, which set requirements for time control. This part is followed by comparison of current system of time control and analysis of time control from other sports with similar problems.

Later on hardware problems are analyzed that are set by our requirements, such as communication between components of the system. Further on proper micro-controller and its peripherals are selected.

Eventually, implementation of the constructed circuit board and software implementation of this circuit and MCU is described in detail.

In conclusion, a prototype with designed circuit board is designed and tested with appropriate methods at archery competitions with supervision of world archery judges.

Chapter 1

Time control

A time control is a mechanism used in various sports and board games. Its main purpose is to ensure that each round or section of competition finishes in a proper time range and the competition can proceed. Construction and given format of time controls vary from sport to sport. For two man competitions there are mainly time controls in form of game clocks which consist of 2 clocks measuring time of each participant for their actions. In other sports, there is one main clock which keeps track of the match's remaining time.

1.1 Chess time control

Chess time control is the type of game clock which has two adjacent clocks with buttons on top which stops on clock and starts the other disabling the clocks from running simultaneously. That gives each player limited set of time for all of his moves in the game. When player runs out of time he loses. There are many types of time controls:

- Increment - before players make their moves, there is a specified amount of time added to their total time which can be accumulated for further moves that can take more amount of time to decide.
- Simple delay - when players are given turn they is a specific delay after which clock starts to subtract time they have left.
- Overtime penalty - these are regular clocks where players do not lose when their time reaches zero. In this case clock after reaching zero starts to count upwards to measure how much the players exceeded their time. This overtime is afterwards added to they result as penalty.
- Hour Glass - in this case player loses when he allows to difference between both clock to reach the specified amount. For example, each player starts with 30 s of time and each second he spends on his move the other player gains one.

1.2 Archery time control

Time control in archery is mainly used in competitions where archers have limited amount of time to shoot their arrows. Archers have to be able to watch their remaining time. Therefore the time control has to be clearly legible and visible for every archer. Time control in archery has to be a mixture of game clock for match plays and main clock for qualification.

1.2.1 Archery competition

Archery competition is divided into two stages. First is qualification which should make placement for match plays which are using ladder system that is at each level defined as square of 2. Each level represents match state that will eliminate exactly half of archers until semi-final matches. These matches will result into final positions (1-4).

1.2.1.1 Qualification

At competitions, archers shoot simultaneously from the same shooting line on targets which are at different distances according to each archers' category. All archers have the same time limit to shoot their arrows. Time limit differs according to type of competition.

- Indoor
  - 3 arrows
  - 120 seconds
  - 20 sets 60 arrows total
- Outdoor
  - 6 arrows
  - 240 seconds
  - 12 sets 72 arrows total

At some competitions, in order to increase number of archers, there are 2 groups of archers shooting on same targets. These groups are marked as AB and CD. During qualification, these groups are continuously switching. At the beginning of each set, group mark also needs to be visible in order to inform archers which group is about to shoot.

1.2.1.2 Match play

Here archers compete in brackets system till final matches. Two archers shoot sets where higher set score gets 2 points, same score 1 points and less score 0 points. The archer who earns 6 points wins the match. If both archers score 5 points, then there is an one arrow shoot-off, arrow closer to the center wins. In higher competition levels, the two archers do not shoot at the same time, they switch. When one archer shoots, the other has resting time which means that his or her time limit is paused.

Figure 1.1: Archery competition

Source: [1]

1.2.2 Additional usages

For every archer practice is an essential part of archery, without hours and hours of training they would never be able to achieve such accuracy and consistency. Time control can be used as helpful tool for increasing these traits.

1.2.2.1 Practice competition

Competitions are always very stressful for archers especially for beginners who never attended a competition. For every archer, it is best to try his or her first competition at practice range.

1.2.2.2 Timing

Timing is also a part of consistency. Professional archers have almost the exact timing of each set. Archery clock can be also used to practice this skill, with correct customization of acoustic signal in expected timing for archers to be sure they executed their shots correctly.

1.2.3 Control components

Time control is a system of different components each presenting different information which is essential for archer to be as clear as possible. There are two types of components, the first are visual components and others are acoustic ones. Acoustic signal consists of single tone which differs based on sequences in which it sounds.

- Digital clock

  These are usually basic 3-digit 7-segment displays which are easiest to construct. However, for international competitions there are full television systems with much more information than time remaining. But these are really only for the audience.

- Lights

  Usually in semaphore format representing simplified version of digital clocks. Each section of time is represented by different color. Usually 30 s - yellow, 10 s - red before the end of the time limit.

- Groups display

  At competitions, the group display is usually made only as manual boards with group marks on it, because most systems with digital clocks do not have groups integrated into them. It can be a separated panel instead of only showing the group mark on start of every set, then it is visible to the audience during the whole qualification.

- Pixel display

  All of the above can be simulated on single pixel display array. However, this approach requires powerful computer for televisions or very high hardware cost increment.

- Speaker or Siren

  This is mandatory part of time control because it is the main signal for archers that they can move in front of the shooting line and when they can start shooting. This component is a minimal requirement under which archery competition can be organized.

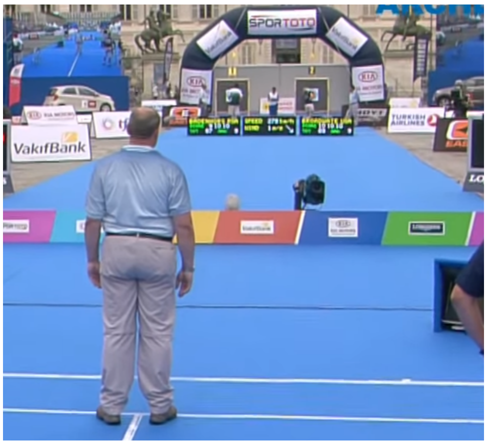
1.2.3.1 Current solutions

Most of the current solutions are divided into 2 types. First type are commercial solutions which have all of the components included, where digital clock are created with professionally manufactured large LED displays or large scale televisions. These solutions are mostly used in top competitions and are very expensive solutions for small clubs.

Figure 1.2: WA indoor solution

Source: [1]

Figure 1.3: WA outdoor solution



Source: [1]

Second type are home made solutions which are focusing on functional part and simplicity. For example archeryclock.com [2] is project using arduino controlling LED strips as digital clock, lights and acoustic equipment or television/monitor, computer is then used as a control unit. This solution is very inspirational for this work. However, it still requires a computer as the control unit which is very difficult for small clubs to have prepared for competitions with proper software.

Figure 1.4: ArcheryClock.com Solution

Source: [2]

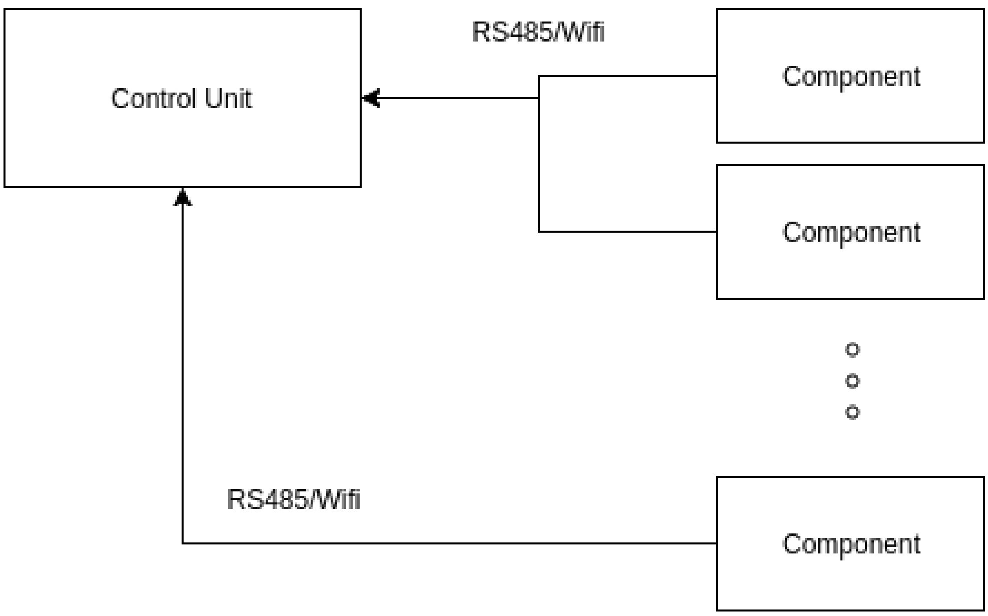
Chapter 2

Hardware analysis

2.1 System layout

System will consist of several visual components which will be connected with central control unit that will control all of the components. Acoustic components will be integrated into some of the visual components. Central control unit will be communicating with components through defined protocol. This central control unit can be computer of in case of Wifi communication a handheld device which is one of the main goals of this work.

Figure 2.1: System block diagram



2.2 Digital clock

To construct at least 20 cm - 50 cm 3-digit 7-segment display at most

$$\frac{50}{2} \cdot 7 \cdot 3 = 525cm$$

of led-strips which come usually with 60 LEDs per meter will be needed. With LED consumption about 20 mA results in

$$60 \cdot 5.25 \cdot 0.020 = 6.3A$$

at full brightness. However, we will never reach this maximum because not all LEDs will be used at the same time and construction of hardware will differ only by replacing power supply to meet maximum consumption.

2.2.1 Led strip

The first approach is to use basic RGB LED strip with control over each color and brightness with pulse width modulation. This approach is very simple, however, first problem that we encounter is number of connections that have to be made in order to have full control over each segment of the display. Each strip has RGB connection which results in

$$3 \cdot 3 \cdot 7 = 63$$

connection for full control over each LED strip.

2.2.2 I^2C register

Problem with number of connections can be reduce by uniting the color control. Whole digital clock will have always the same color and brightness which reduce our connection problem by 3 per LED strip and result into uniting pulse width modulation signals. Now we need to have connection only stating if LED strip is on. Still we have

$$3 \cdot 7 = 21$$

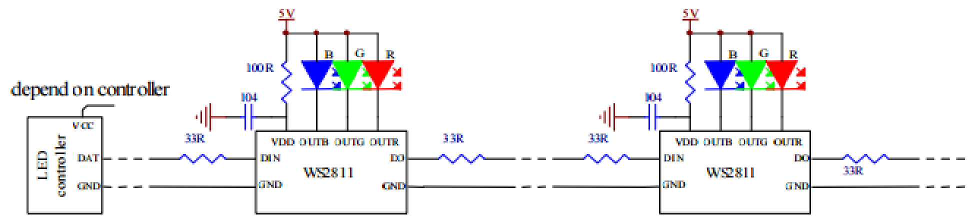
connections that has to be fully controlled with MCU. At this point we can solve this problem by using I^2C bus with registers. A board with I^2C register and transistor logic is placed behind every digit which will operate input into LED strips. I^2C register with 7 bits operate those segments of digits that are currently on and with other bits it can set color of the digit. Current state of our solution is however still very complicated for production. Large amount of hardware parts is needed .

2.2.3 Led driver

WS2811 is LED driver that allows control of LEDs on led-strip with serial interface which provides full control over leds' color and brightness over single wire. This will eliminate any additional hardware parts and grant us all required features. On the marker the are already led-strips with integrated WS2811.

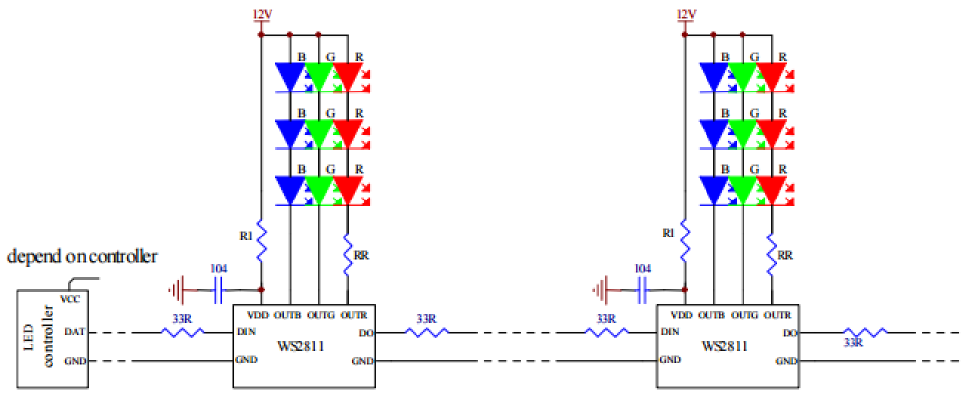
Led driver documentation suggests two main circuit usages. First one for usage with 5 V with single led. Second for usage with 12 V with three leds. In the constructed prototype we are utilizing version with 12 V.

Figure 2.2: WS2811 circuit 5 V



Source: [3]

Figure 2.3: WS2811 circuit 12 V



Source: [3]

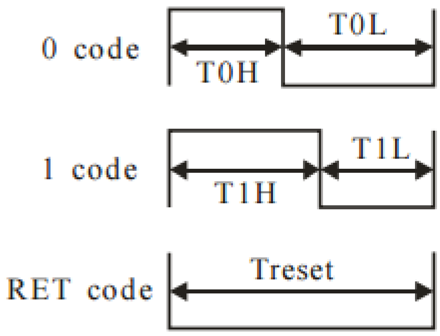
Serial interface of WS2811 consists of single wire on which pulse sequences for bit 1 and bit 0 with specific timing are defined.

Table 2.1: WS2811 Sequence timing specification

Label	Sequence	Voltage	Timing	Aprox
T0H	Bit 0	High	0.5 μs	150 ns
T0L	Bit 0	Low	1.2 μs	150 ns
T1H	Bit 1	High	2.0 μs	150 ns
T1L	Bit 1	Low	1.3 μs	150 ns
RES	Restart	Low	Above 50 μs	•

Source: [3]

Figure 2.4: Pulse sequences



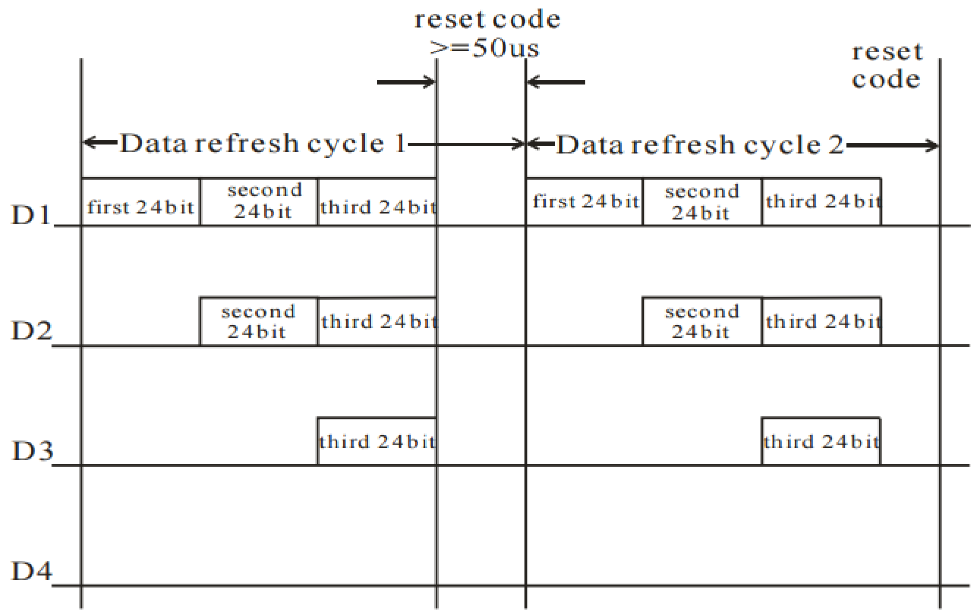
Source: [3]

Each LED will read first 24 bits which represent RGB color of LED and rest will pass-through to other LEDs in the strip. When the low state is larger than 20 ns, the LED will reset and wait again for first 24 bits. Timing of this interface is very strict. With basic calculation we can calculate worst possible time which it will take to transfer color information to ws2811 depending on number of LEDs in visual component in μs.

$$T = 3.3 * 24 * Leds$$

With this solution we will be allowed to use MCU with small amount of pins that does not require very high performance.

Figure 2.5: WS2811 Cascading



Source: [3]

2.3 Communication

For communication we need reliability over some medium distances. Parts of the system can be placed over large fields which for wired communication lead for hundreds of meters and wireless has to cover array around hundred meter diameter. We have multiple means of communication layers. For wired communication RS485, for wireless Bluethooth and Wifi.

2.3.1 RS232

RS232 is one of the most used communication interfaces in the world thanks to its simplicity and hardware support from almost any device you can find on market.

In standard, there are many connections required for two devices to communicate, however standard also supports simple 2 wire connection of devices. These devices need to operate on the same communication speed which is called baud rate or bits per second most common rates are 9600, 11400, 19200, 115200 and others.

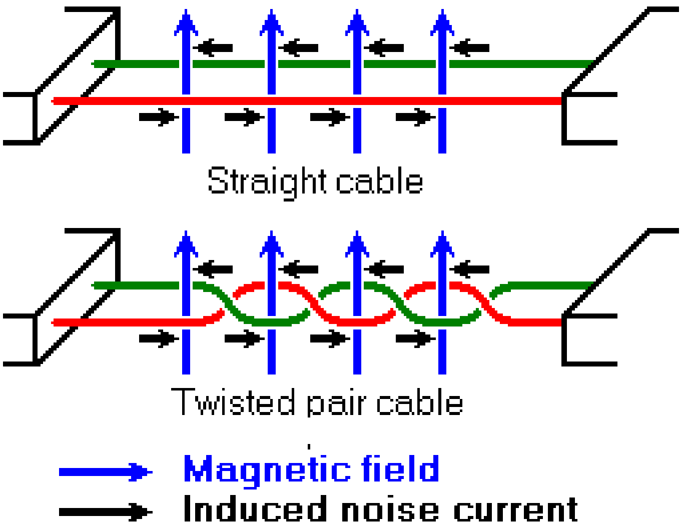
Each device has a transmit and receive slot labeled as Tx and Rx, these slots are cross connected between device for proper communication. This means only two devices can be fully connected. There are some usages where one device can communicate with multiple devices, however, it is only one way communication, because only one transmitter can be present on the receiver line.

Each transferred byte has to start with start bit and end with stop bit. In addition each byte can have multiple stop bits and can contain parity bites for error checking which then can define protocol report as communication error. In many cases this setting is set to one start bit and one stop bit without parity. Many usages where data consistency is required, then on each protocol message some error checking control for example CRC8, CRC16 and etc are placed.

2.3.2 RS485

RS485 is serial communication method for devices to communicate over distances with resistance of external noise. It is differential half duplex bus. With regular serial communication RS232, the receiver compares signal with common ground, but the trigger level 3 V can easily pick up noise which makes this method highly unreliable over greater distances. However, RS485 receiver compares levels on signal line + and signal line - and compares only the difference between them. Which means that any noise that affected one signal will affect the other one with the same effect. The best noise resistance is achieved by using twitted cables which are distributing effect induced noise current.

Figure 2.6: Twisted pair



Source: [4]

According to characteristics, RS485 has maximum distance 1200 m that will meet our requirement of distance with speed 100 Kbs which is sufficient for our synchronization signals.

Table 2.2: RS232, RS485 specifications

	RS232	RS485
Differential	No	Yes
Max number of drivers	1	32
Max number of receiver	1	32
Modes of operation	half/full duplex	half duplex
Network topology	point-to-point	multipoint
Max distance (acc. to standard)	15 m	1200 m
Max speed at 12 m	20 Kbs	35 Mbs
Max speed at 1200 m	(1 Kbs)	100 Kbs
Max slew rate	30 V/μs	n/a
Receiver input resistance	3..7 kΩ	$\geq$ 12 kΩ
Driver load impedance	3..7 kΩ	54 Ω
Receiver input sensitivity	$\pm$3 V	$\pm$ 200 mV
Receiver input range	$\pm$15 V	-7..12 V
Max driver output voltage	$\pm$25 V	-7..12 V
Min driver output voltage(with load)	+5 V	$\pm$1.5 V

Source: [4]

One of disadvantages of RS485 is that it only supports half duplex communication which means that there can be only one transmitting device on bus, otherwise all communication on bus is scrambled. This results in creating request/response communication protocol which will ensure that only one device will currently transmit its response. Requests will be only generated from one device that will be bus master.

2.3.3 Bluetooth

This type of wireless communication is one of the most supported types of communication in common devices such as computer and mobiles. This could be used in our advantage in creating user application which will control our system. However, specification of Bluetooth gives us maximum range of 60 m on newer devices. This maximum range is not guaranteed and in most cases is not working. Which makes Bluetooth unusable for our purpose.

2.3.4 Wifi

Wifi is also one of the most supported types of wireless communication which has very large operating range which outdoors is around 100 m. Indoors mobile access point for smaller instances of time control can be used. Also at most of archery ranges Wifi networks are already present which makes this option very promising.

2.3.4.1 Communication interface

Communication interface should be represented by 2x5 pin connector which would have RS232 interface connected to MCU. With additional signals such as GND, VDD and additional GPIO connected to MCU. In this pin connector communication module could be inserted, which should behave like transparent UART bridge into designated transport layer like Wi-Fi or RS485

Module for 85 would be designed specifically to fit into this interface. As Wi-Fi module we would choose one already present on market which is using ESP8266 chip. ESP8266 is low-cost Wi-Fi chip with full TCP/IP stack and programmable MCU. That can be used to create transparent UART – Wi-Fi bridge for our time control with possibility to host its own Wi-Fi network for configuration or direct control. Thanks to the possibility of creating custom software for ESP there is space for future development of this interface.

ESP8266 could be directly used to control digital clock and other parts of the system,however in case of digital clock the very strict timing protocol for led-strips which occurs every second and takes large amount of time may result in errors in tcp/ip communication of ESP8266 handled by real-time operating system present on ESP8266.

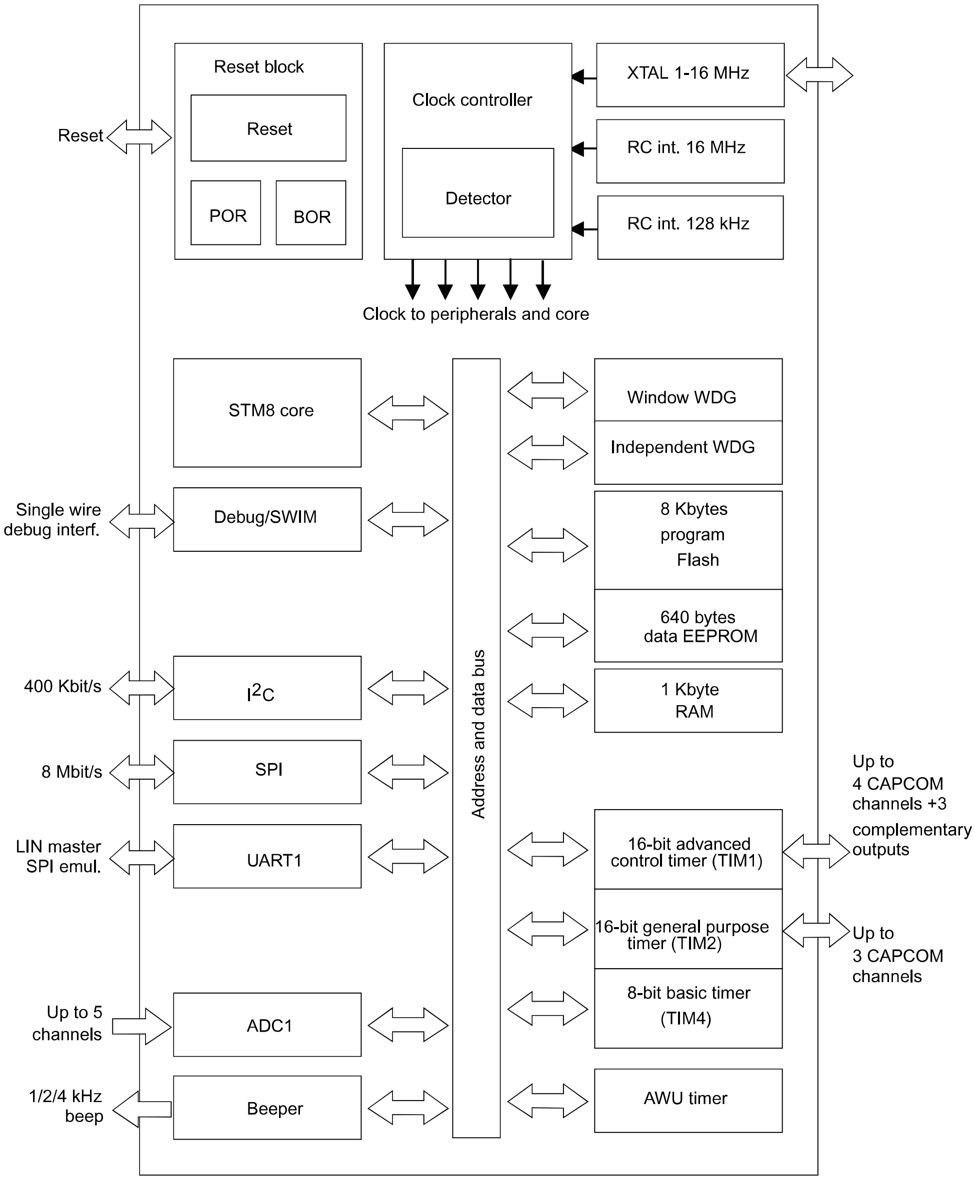
2.4 MCU

There is many MCUs meeting our small requirements for controlling LEDs and other peripherals:

- TIM1 - For precise second count down.
- UART - For communication with master which will be converted into RS485 or Wi-Fi communication.
- Beeper - For acoustic equipment which can generate range of frequencies for acoustic equipment which does not include internal oscillator.
- ADC - For measuring light environment to adjust brightness of display.

Our target was very cheap and high available MCU such as STM8S103F3 [5] which can be exchanged for STM8S105K4/6 which has higher amount flash, ram and gpio pins in case the board needs to be expanded.

Figure 2.7: STM8S103F3 Block diagram



Source: [5]

Chapter 3

Implementation

Implementation process was divided into two main section. The first one is hardware implementation during which several prototypes were constructed which served as proof of concept on universal PCBs. Each hardware interface was tested on these boards for the final prototype. Final prototype was designed a manufactured as PCB.

The second one is software implementation where several control FSMs were implemented which in cooperation control entire time control and each of its interfaces.

3.1 Hardware

Prototype board was designed which bear interfaces for LEDs and acoustic equipment. This board then would be used in each component of time control.

- Header for communication module ESP or RS485
- Led output which can be used for clock digits or information lights
- Buttons for controlling score in match play mode or changing brightness levels .
- Light sensor which can be configured for automatic adjustment of LED brightness instead of fixed levels.
- Power relay for controlling power supply for leds.

The prototype was constructed in form of digital clock with build-in speaker.

3.1.1 Circuit

Whole circuit is powered by 12 V with fuse for protection which should be chosen according to power supply and expected power consumption by LED strips. For logic part is 12 V needs to be converted into 3.3 V which has multiple solutions:

- Voltage divider - Its main advantage is simplicity of realization, however it has large heat losses.

- Zenner diode - Which is also very simple to implement and has higher resistance for voltage drops when current consumption increases. However, it shares the same disadvantage with voltage divider, it has large heat losses.

- Linear stabilizer - Linear stabilizer is the most convenient solution for this problem. It has minimal heat losses and for our consumption of few hundreds mA represents the best solution.

On board there are two communication interfaces located - RS232 and ESP8266 - which can be changed with jumpers according to the used interface connected to board. RS485 driver has RS232 interface with additional signal Write Enable which can be connected to ESP8266 interface slot.

3.1.2 Clock digits

The first problem after constructing maximal designed scale digits was light intensity of LED strips. Each LED on the strip was very bright light source which in direct look was not pleasant to the eye. This fact needed to add some sort of light diffusing component which will disperse point light into full surface of digit segment.

- Milk glass, Led channels - this is possibly the best possible solution to disperse the light. However, for construction it is very difficult to use, it requires in case of each segment presence of a container which is on top enclosed with glass. Other problem with these is also width of the channels which is commonly available in only one width slightly wider then LED strip's.

- Glue, silicon - for prototype method of casting hot-glue was used which can be purchased in any utility shop and is very cheap. With correct height of casting form and width the results are very promising. It represents a very fast way of creating segments and much easier than installing each segment on board with channel. Other advantage is also increasing water resistance.

3.1.3 MCU Interface

MCU STM32S103PF6 is connected to all interfaces implemented on board. MCU requires SWIMM connection which is used to program and debug.

Table 3.1: SWIMM connector

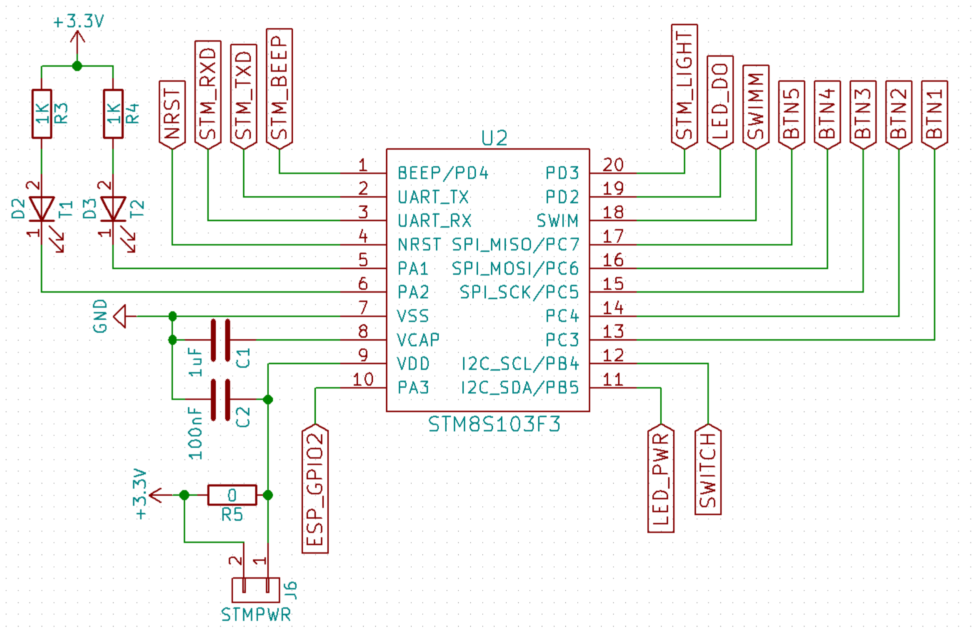
Pin	Name
1	VCC
2	SWIM
3	GND
4	NRST

Table 3.2: MCU Pinout connection

Pin	MCU Label	Interface
1	BEEP	Acoustic equipment
2	UART TX	Serial Out
3	UART RX	Serial In
4	NRST	Reset
5,6	PA1, PA2	Status LEDS
10	PA3	ESP8266 GPIO
11	PB5	Led strip power relay
12	PB4	Switch
13-17	PC3-PC7	Buttons
19	PD2	Led strip data output
20	PD3/ADC	Light voltage divider

Almost all of the connected hardware such as buttons, photo resistor and others needs to have additional hardware parts for proper connection to MCU.

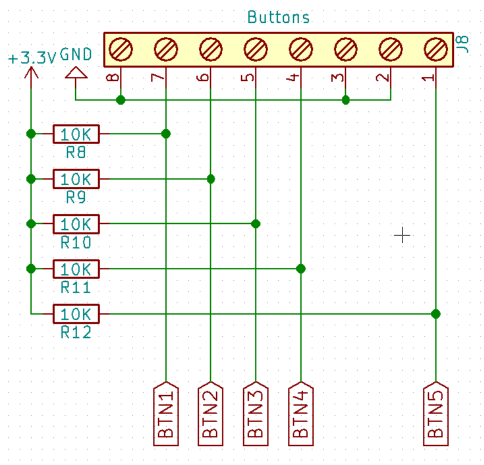
Figure 3.1: MCU interfaces



3.1.4 Buttons

Buttons are used for setting fixed brightness of display and in archery matches to set scores on display. They are connected externally with screw connector, then connected with pull-up to MCU GPIO pins as inputs. The same approach is used by internal buttons and external switch. Buttons are manufactured also in push switch mode which must be also supported by MCU because these buttons are more robust for hardware usage.

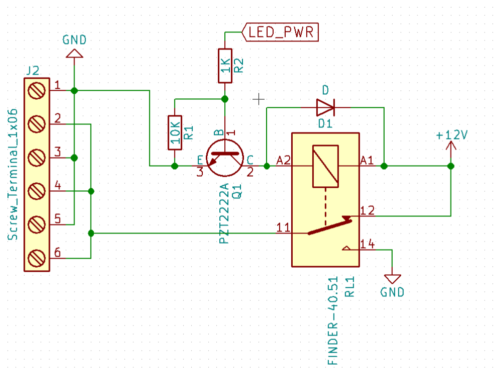
Figure 3.2: Buttons connection



3.1.5 Led Power

Each digit has its screw connector and they are all controlled with relay which cannot be directly controlled by MCU, because it requires according [6] minimum load of 300 mW to switch. At 12 V it is 25 mA which is not past MCU's maximum 100 mA however it is rated as sum of all output. For lowering load on MCU we connect it to relay through NPN transistor which requires much lower load.

Figure 3.3: Relay connection

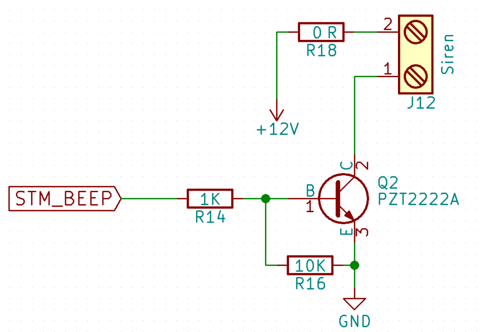


3.1.6 Acoustic equipment

Any acoustic equipment can be connected to Beep signal which will generate 1, 2 or 4 kHz signal which will then control equipment with NPN transistor. There is 0 R which can be replaced by non-0 R in case of demand by the acoustic equipment such as piezometer.

For prototype testing speaker was selected, because frequencies can be changed with the Beep signal. In contrast to piezometer or siren where frequencies are mostly fixed with internal oscillator.

Figure 3.4: Acoustic equipment connection



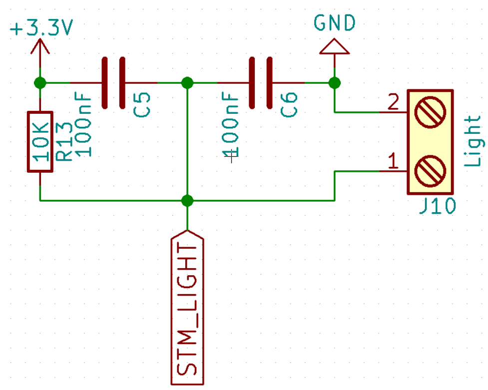
3.1.7 Light sensor

Light sensor is implemented as voltage divider which will change its voltage according to changes of resistance on photo resistor. Output voltage can be computed as

$$V_out = \frac{PR}{R3 + PR} \cdot V_in$$

where PR is resistance of photo resistor. We can then use this value to transform through linear function with maximum and minimum for our brightness. This, however, is very subjective and requires manual calibration.

Figure 3.5: Photo resistor connection



3.1.8 Printed board circuit

Designing of PCB with high currents needs to meet track width requirements otherwise there would be danger of overheating of PCB. Board should be able to withstand at least 12 V and 7 A of current to LED strips. With these information we are able to compute track width required with minimum temperature increase. In this design, this track is merged with polygon area which is 22x9 mm.

$$Area = \left(\frac{I}{k \cdot Temp\_Rise^{b}}\right)^{c^{-1}}$$

Constants k, b, c are defined from IPC-2221 standard [7] for internal layers as:

$$k = 0.024, b = 0.44, c = 0.725$$

$$Width = \frac{Area}{Thickness \cdot 1.378}$$

Thickness is in oz/ft^2 units. With this computation we can calculate that for safe 10◦C temperature rise the width should be at least 9 mm which is met by our minimal area.

Figure 3.6: Board Scheme

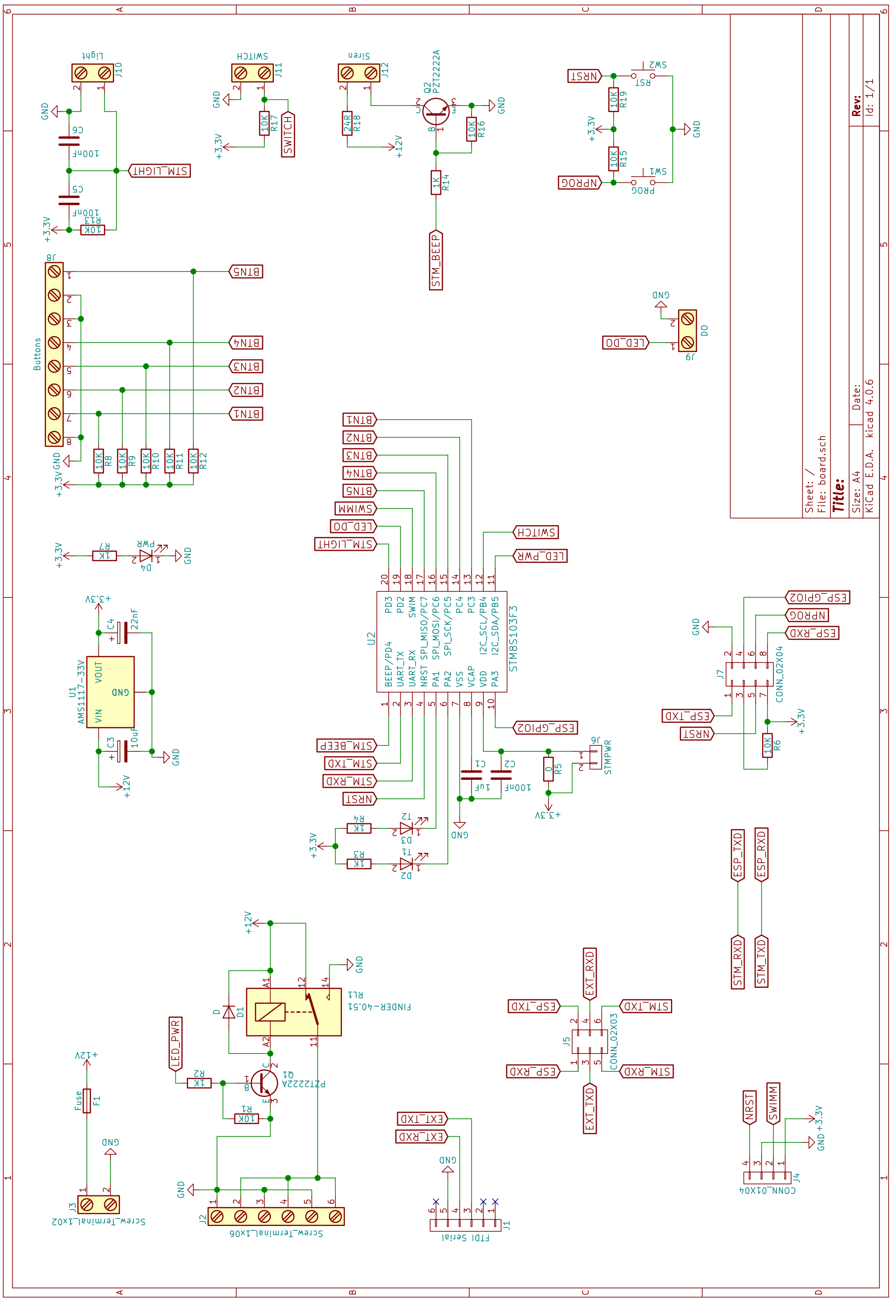
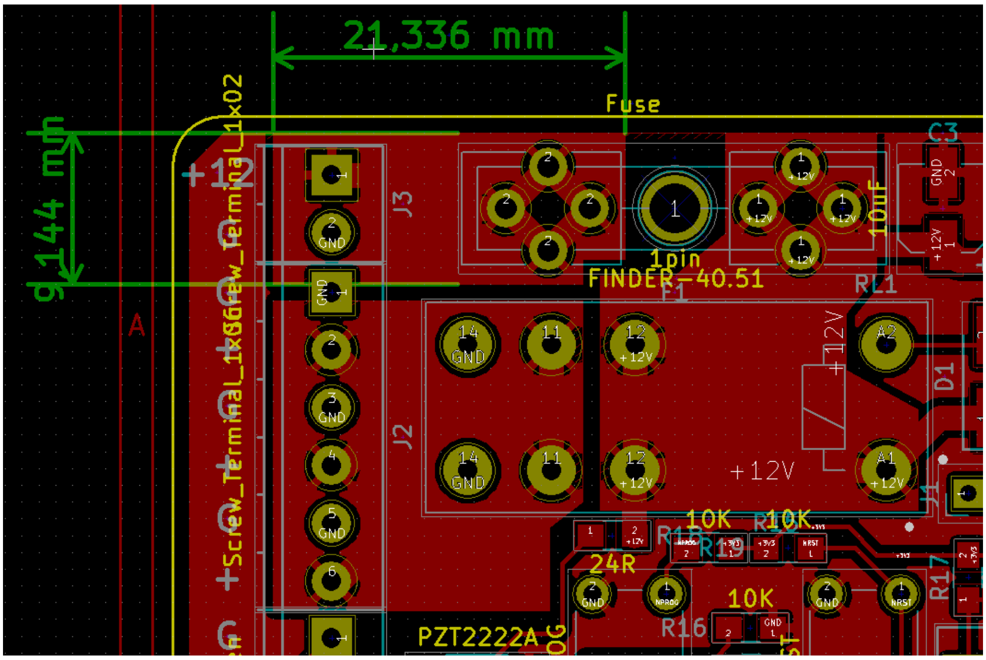


Figure 3.7: PCB Power areas



3.2 Software

Software for MCU is written in pure C with use of library STM8 standard peripheral library, which helps increase development speed. Main advantage of library is functional access to peripherals, which is much easier to use than access with registers.

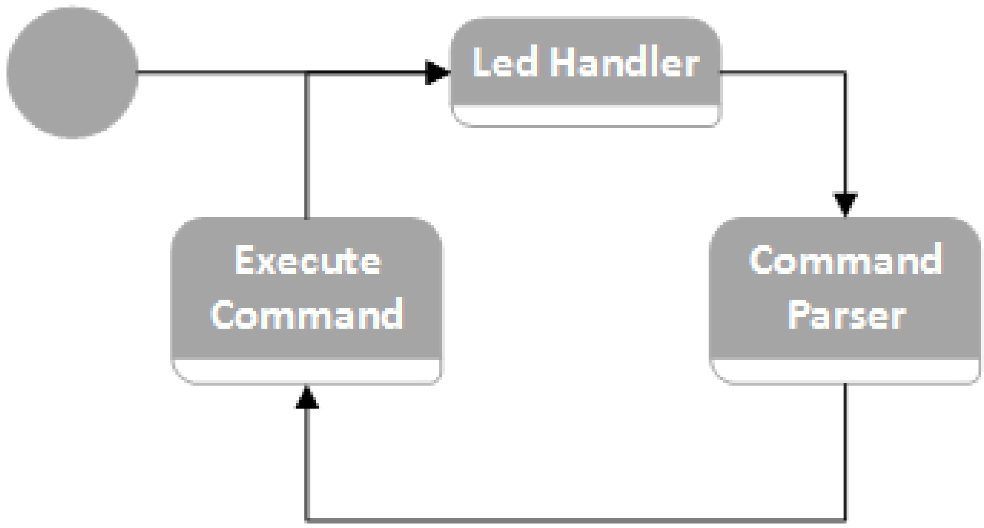
3.2.1 Main Program

Main process consists of handling several finite state machines.

The first one is responsible for handling LED display. The second one is responsible for parsing protocol defined earlier, validation of packets and sending basic ACK and NACK responses if the command was inserted in queue. Then there is a simple command executor fsm which executes correct action for each packet.

These finite state machines were tested with external unit tests on computer. Testing of these FSMs is essential for avoiding debugging in development which is a very time consuming activity on micro-controllers.

Figure 3.8: Program loop



3.2.1.1 Led Handler

FSM is responsible for time control behavior. It has multiple states depending on commands inputs.

- Display state - displaying simply given text.
- Countdown state - displaying count down value with support of color rules.
- Match state - displaying current state of the match and thus allowing control of score with buttons.

Led handler is optimized to prevent any unnecessary LED handling, because control of the whole panel is not a trivial action. Without these changes MCU was nearly useless.

Color rules are rules for color changes which can occur during countdown. When count down reaches given value it changes color according to the rule.

When LED handler initiates data output for LED strips memory buffer containing colors for each led, which is then transmitted with time critical function that has to have strictly deterministic process. This process has critical deadlines which are reasons why we cannot generate data for each LED in process of transmitting. During this process we have to disable interruptions and embed our c code with assembly instructions to ensure successful transmitting.

3.2.1.2 UART

Almost every modern MCU has hardware support of UART interface which has very small hardware fifo in which hardware interface stores received bytes. Hardware fifo has usually only few bytes (2-4). Software then has obligation to retrieve bytes from this hardware fifo and store them in memory, otherwise it will lose information.

Both RX and TX hardware fifos are handled by interruptions which should remove unnecessary busy waiting for transmitting and receiving. Each of interruptions is implemented as simple push into software fifo from which then command parser pops its input.

3.2.1.3 Command Parser

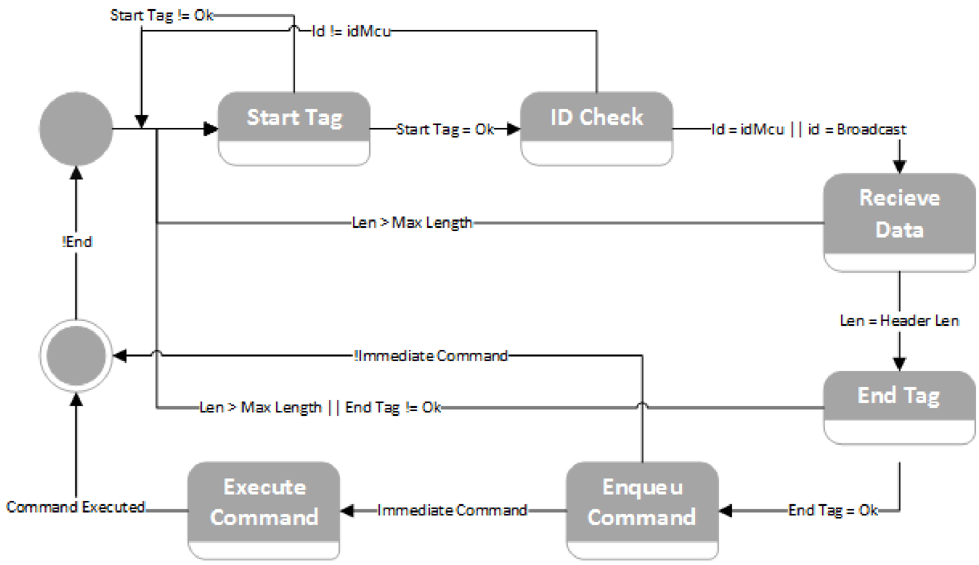
Command parser is designed to parse our transport layer of packages. Our communication is defined as request-response protocol which needs for its function identification receiver of protocol packet. This identification is stored inside MCU's EEPROM inner memory for its non-volatile memory property. It also supports broadcast of request which has no response defined because it would conflict on RS485 bus where only one device can transmit at the same time.

On RS232 and RS485 some noise bytes can occur which we have to ignore. Then starting and ending tags are implemented, which frame our packet data.

In case of parser failure FSM returns into its starting state where it awaits the starting tag and whether the parser detects message which has his identification byte. It sends NACK data packet otherwise it responses with ACK. Each request has response which is send when a command is executed.

Parser also implements support of immediate commands which suspend parsing and en-queue immediate command in the first place of command and waits until this command is executed. This command should wake dormant command executor.

Figure 3.9: Command parser FSM



3.2.1.4 Command executor

Command executor is simple FSM which pops commands from command handler and executes them. It has two special features. The first one is waiting for given time which allows us to en queue several requests at the same time and these can be then executed in order with some time between them.

Example:

1. Set countdown 120 s
2. Start countdown
3. Wait 120 s
4. Use acoustic equipment for 0.5 s

Thanks to responses from each executed command we will receive ACK response from acoustic command when is executed. This can be used for no need of external timer in host device.

This also results in very robust design with no need of continual poling or connection. However, these delayed responses can lead to bus collisions in case of RS485, if not used properly. Bus mastering and scheduling delayed responses is a responsibility of central control unit which has to use these responses only on one component on single bus.

Table 3.3: Supported commands

Command Id	Execution function
0	Returns ACK
1	Returns information about device such as device Id and firmware version
2	Sets device id
10	Sets waiting constant for command executor
11	Clears command queue
20	Sets LED handler state
21	Sets LED handler countdown constant
22	Set LED handler brightness constant
23	Add LED handler color rule
24	Set LED handler color rule
25	Clear LED handler rules
26	Set LED handler display string
40	Set state of acoustic equipment ON/OFF
41	Set state of acoustic equipment ON with time after which is set to OFF

3.3 Application tools

For each part of the designed prototype specialized software was used.

- Series of tools used for developing STM MCU and writing source code.
- Development library for future development on ESP.
- Tools used for designing schema of circuit and its implementation to PCB.
- Software for verifying generated manufacturing files of PCB.

3.4 STM MCU

The attached project is created with use of several application tools, first of which is QtCreator that is used as IDE with custom build script which is written in bash. QtCreator invokes build script which then assembles binary using SDCC [8] compiler.

SDCC is a small device C Compiler which is distributed under GPL [9]R license with other tools that we are using to assemble binary application for our MCU. This is an essential part of this project, because official tools for STM8 series only have windows implementation. And this project is developed under linux distribution which is not officially supported by STM.

In the project STM8S/A Standard peripheral library is included which has special configuration of our STM8 MCU. Build script is also trying to reduce used program memory by including only used parts of this library, without this optimization the program would exceed our memory limit given by MCU.

Additional programming of our MCU and debugging is using ST-LinkV2 which is connected to MCU with SWIM interface. Application is then flashed into MCU using stm8flash which is another application under GPL license that is the only application capable of communication with ST-Link SWIM interface under linux.

3.4.1 ESP8266

ESP8266 is a very popular MCU with Wi-Fi interface and its toolchain is fully free and available to public. There are many projects using official and unofficial SDKs which are very difficult to choose from. Official supported SDKs are RTOS SDK [10] and Non-OS SDK which is discontinued SDK and RTOS should be replaced. RTOS SDK API is almost identical to Non-OS, however, the programming approach is a little different.

Official programming guide [11] is very detailed and contains all information needed by developer. Unofficial SDKs however most of the time have outdated guides, because they are in hands of community which usually enjoys to do the fun parts of development and keeping documentation update is not one of them.

In work prepared project for RTOS SDK for future development of time control is included. It is supposed to work as transparent UART bridge with several specific functions for this solution. For now, there is a functional solution created by community which is outdated and supports only AT commands. These are not the best solution of communication between our MCU and ESP8266 and can be enhanced using additional GPIO ping connected to MCU.

As an alternative to creating control application on separate device the control application can be located on ESP with web interface which would send control commands to STM MCU. With extension on certain port it would still keep transparent UART bridge functionality. That could be used as an entry point for control ESP to control other components on network. Components would be gathered on network hosted by control ESP which could then discover all connected components by broadcast packet or it could directly send some packet to each device connected. The broadcast way would be more a reliable solution and would allow the use of other Wi-Fi host then control unit. This solution looks very promising, because of large support of web application for various devices such as tablets, personal computers and mobile phones. Another large advantage would be use of front-end oriented design which would lead to lowering of performance load on ESP.

3.4.2 KiCad

KiCad is an open-source software tool for the creation of electronic schematic diagrams and PCB artwork [12]. This software consists of several parts where each is used for given part of design process. The first part of the design process is creation of the schema in Eeschema which helps us to avoid misstates with unconnected pins and allows us to create scheme which is not dependent on part package. Later with transition to board design we assign with the help of footprint selector CvPcb package to given part. This process is essential for circuit board editor Pcbnew which needs to know package footprint for parts. Circuit editor shows you which footprints pads need to be connected to each other according to schema. There is a continual connection between each application which then propagates any changes from schema to board editor. This allows continual changes in design, which are very common, because most of the circuits in first revision have bugs.

3.4.3 Manufacture files

Manufacture files are required in order to place order by pcb manufacturer. These files represent each layer in pcb design. These files are generated from KiCad software. In order to check if every element we placed on the board including text labels is recommended to check each layer with gerber viewer such as gerbv [13] which is open-source software which supports both windows and linux platform.

CHAPTER 4

Testing

4.1 Software

The only way of testing the software is the test of FSMs except actual practical test. Testing functionality of FSM is essentially very simple in this case. Code is written in C which is portable and we can take FSM and run the desired tests on development computer. We select cases of accepted language and verify if they are accepted. Then we select special cases of words which should be designed to test overall functionality of FSM. In our case these would be for example rejection of command which has different identification, message length and filtering until start tag etc.

4.2 Hardware

In case of this board the hardware testing has only one form and that is manufacturing testing which should eliminate errors caused by manufacturing process such as lose contact pads, shorted circuits and malfunctioning parts.

These tests are designed to test each hardware interface on board. Functionality of MCU is also tested with these tests because they are based on functional MCU, without it they would fail.

These tests have a form of software which is uploaded into MCU. It contains series of actions which test functionality of each interface. There is a response in form of a LED light for-showing results of each step of testing procedure.

- Buttons - Each button is tested for its initial state and for state when pressed.
- Relay - Checking if relay is actually set.
- Light sensor - Checking state when exposed to direct sun light and when fully covered.

- Power supply is tested before powering MCU with jumper connection.
- Led output tested with actual LED strip.
- Beeper output is tested with speaker.

During final assembly and testing of final prototype board there was a mix up in pin of linear stabilizer which was in package SOT-223. There are many linear stabilizers in this package and they follow two pin layouts. One of them has GND on middle pin and the other one on the first pin. This fact was very surprising to us because we used a part from library and assigned it to PCB edit package SOT-223 and it assigned ground to the middle pin, instead to the first one. This problem was solved by purchasing the other kind of linear stabilizer.

Conclusion

The goal of this work was to design, construct and test an archery time control. Within work on the project construct proof of concept board on universal board with all parts for development of software application for MCU and large prototype board with full scale digit display was constructed.

Functioning of all circuit parts has been verified by the proof of concept boards and selected hardware parts then have been used in further development of final prototype hardware.

Afterwards PCB board was designed on which development continued with additional peripherals and safety elements. This board successfully replaced previous proof of concept boards and was mounted into prototype.

Additionally, documentation was created which describes behavior of FSM controlling time control and documents all of available commands for time control system. With command set implemented on time control there is an option of creating control application for all platforms which can use for communication Wi-Fi or RS232/RS485.

Prototype was tested with a python application which used time control's communication protocol with transparent Wi-Fi module connected to prototype board. All functional parts were tested in practice and competition environment.

Future development is planned which will replace transparent UART bridge in Wi-Fi module by more complex application. This will serve as a web server with REST API available with standard HTTP protocol or with HTTP 1.1 web-sockets.

Main advantage of this web application is its possibility of responsiveness and functionality over all platforms that can display and interact with web applications. For synchronizing multiple components the same interface as REST API using UDP protocol packets will also be implemented which would then be used for controlling other components from the main one, which is controlled by web application.

Bibliography

[1] archery, W. World archery website. https://worldarchery.org/, 2014, [Online; accessed 19-April-2017].

[2] Jegers, H. ArcheryClock.com. http://www.archeryclock.com/, 2017, [Online; accessed 19-April-2017].

[3] Adafruit. *WS2811 [online]*. [cit. 2017-02-20]. Available from: https://cdn-shop.adafruit.com/datasheets/WS2811.pdf

[4] Bies, L. RS485 serial information. https://www.lammertbies.nl/comm/info/RS-485.html, 2015, [Online; accessed 19-April-2017].

[5] STMicroelectronics. *STM8S103F3 Datasheet [online]*. [cit. 2017-02-20]. Available from: http://www.st.com/content/ccc/resource/technical/document/datasheet/ce/13/13/03/a9/a4/42/8f/CD00226640.pdf/files/CD00226640.pdf/jcr:content/translations/en.CD00226640.pdf

[6] FINDER S.p.A. *Finder relay Datasheet [online]*. [cit. 2017-02-20]. Available from: https://www.finder-relais.net/en/finder-relays-series-40.pdf

[7] IPC. *Generic Standard on Printed Board Design IPC-2221A [online]*. [cit. 2017-02-20]. Available from: http://www.ipc.org/TOC/IPC-2221A.pdf

[8] SDCC. *SDCC Compiler User Guide [online]*. [cit. 2017-02-20]. Available from: http://sdcc.sourceforge.net/doc/sdccman.pdf

[9] GNU.org. General public license. https://www.gnu.org/licenses/gpl-3.0.en.html, 2017, [Online; accessed 19-April-2017].

[10] Espressif Systems. *ESP8266 Datasheet [online]*. [cit. 2017-02-20]. Available from: http://www.espressif.com/sites/default/files/2a-esp8266-sdk_getting_started_guide_en_0.pdf

[11] Espressif Systems. *ESP8266 SDK Guide [online]*. [cit. 2017-02-20]. Available from: `http://www.espressif.com/sites/default/files/2a-esp8266-sdk_getting_started_guide_en_0.pdf`

[12] KiCad. *KiCad User Guide [online]*. [cit. 2017-02-20]. Available from: `http://docs.kicad-pcb.org/stable/en/getting_started_in_kicad.pdf`

[13] Gerbv. *Gerbv [online]*. [cit. 2017-02-20]. Available from: `http://gerbv.geda-project.org/`

Appendix A

Acronyms

A Ampere

ACK Acknowledged

API Application programming interface

FIFO First in first out container

FSM Finite state machine

GPIO General purpose input/output

GPL General public license

Hz Hertz

IDE Integrated development environment

LED Light emitting diode

MCU Micro controller unit

NACK Not acknowledged

OS Operating system

PCB Printed circuit board

RGB Red blue green

RTOS Real-time operating system

SDK Software development kit

TIM Timer

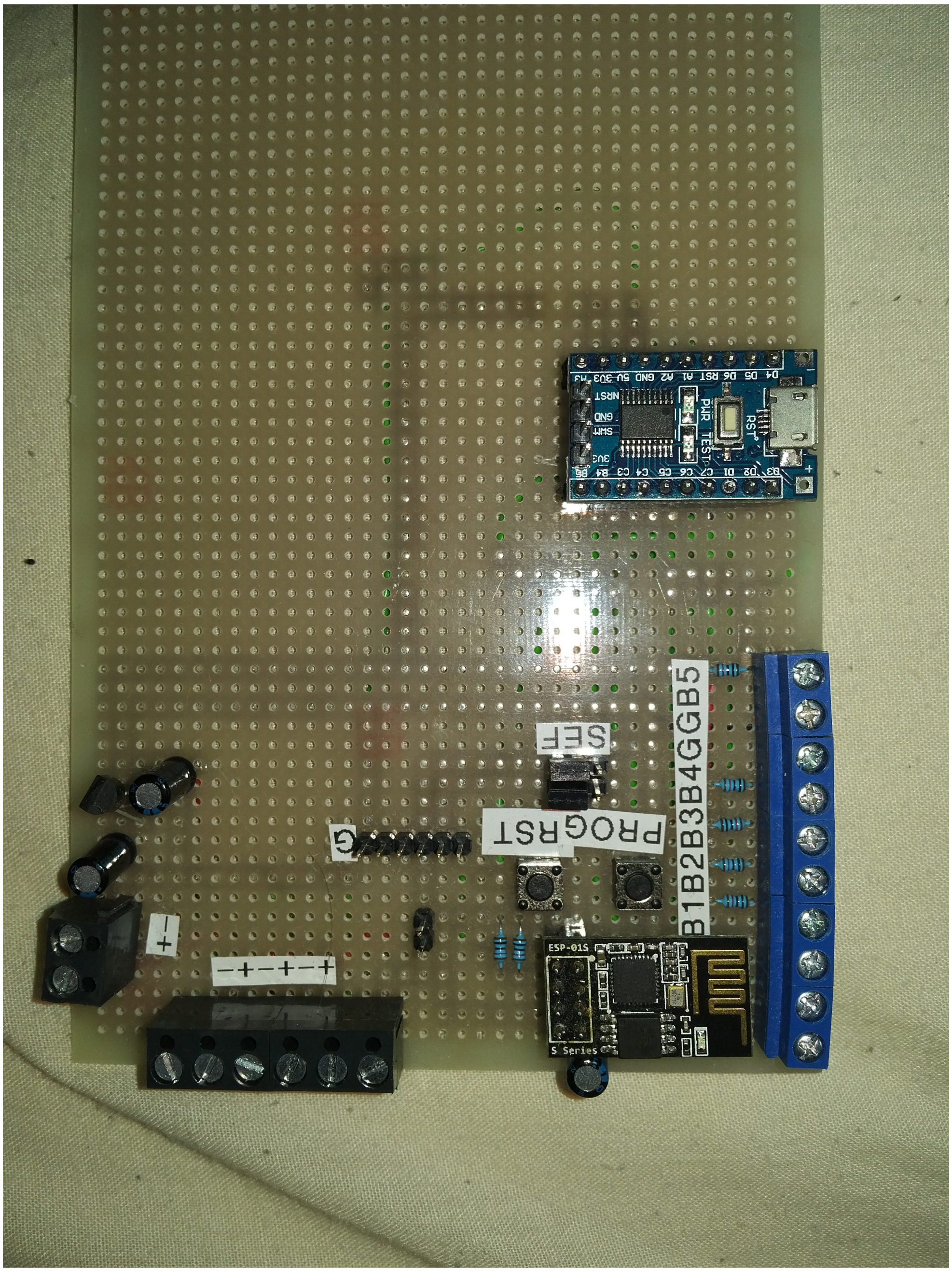
UART Universal asynchronous receiver/transmitter

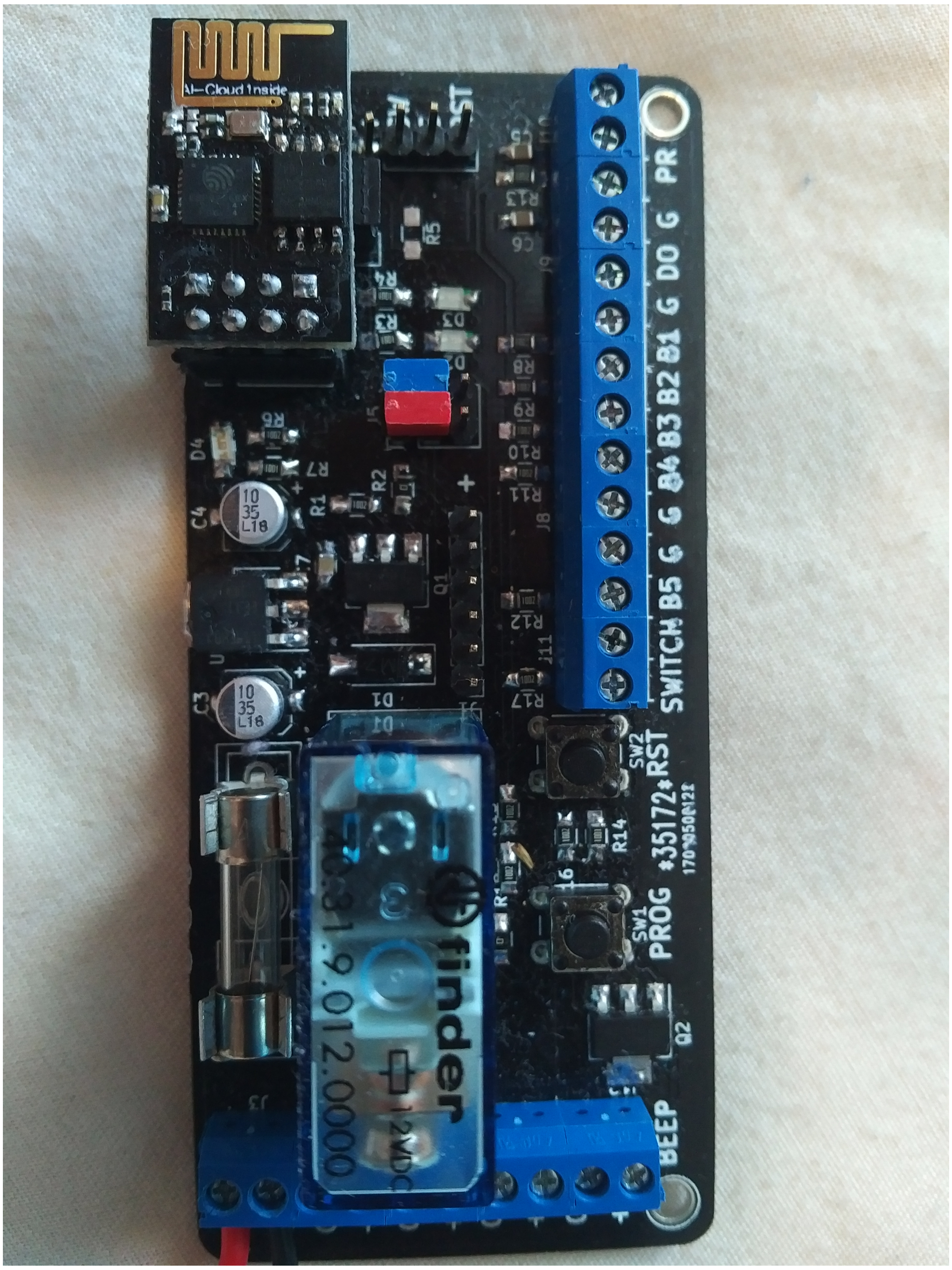
V Volt

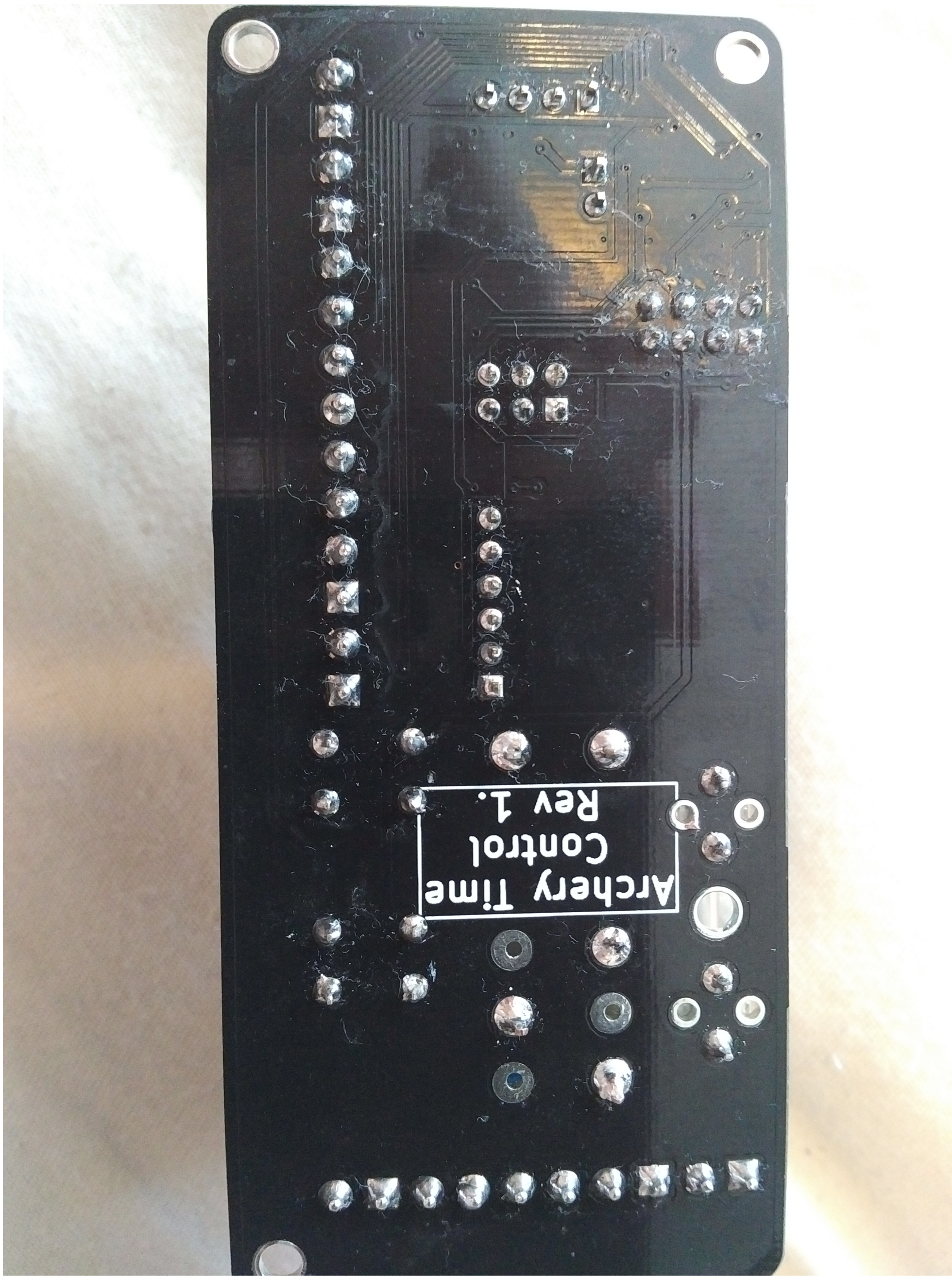
WA World archery

Appendix B

Prototype

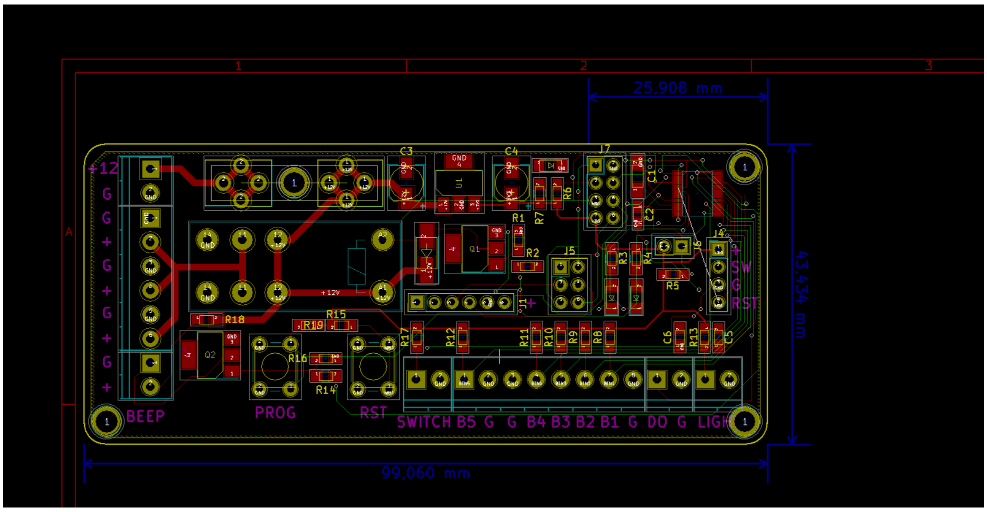

PROG
RST
B1B2B3B4GGB5
ESP-01S
S Series

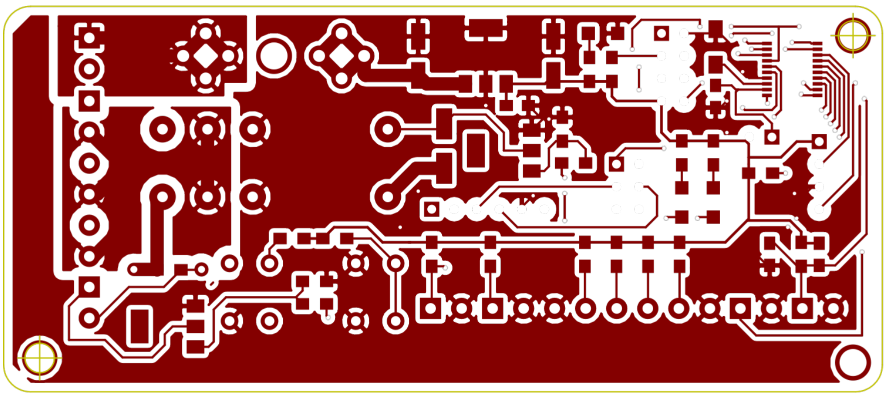

finder
40.31.9.012.0000
12VDC
SWITCH
PROG
RST
BEEP
Ai-Cloud Inside

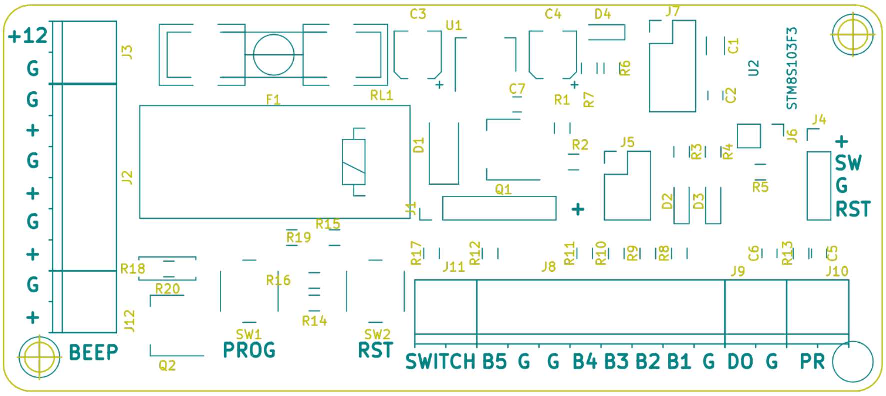

Archery Time
Control
Rev 1.

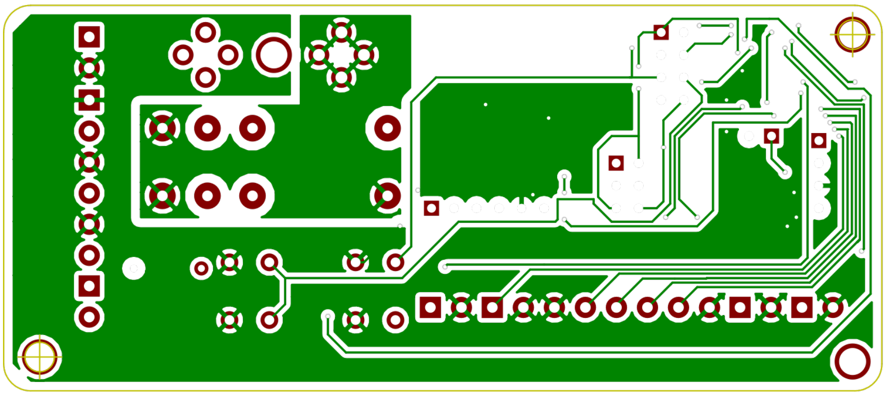
Appendix C

PCB Design






+12
G
G
+
G
+
G
+
G
+
BEEP
PROG
RST
SWITCH B5 G G B4 B3 B2 B1 G DO G PR
+
SW
G
RST
STM8S103F3



Appendix D

Contents of enclosed CD

```
readme.txt ....................... the file with CD contents description
bin ....................................... the directory with binaries
src .................................... the directory of source codes
  mcu ............................... source codes of micro-controller
  bridge ............................. source codes of Wi-Fi module
  pythonClient ................. source codes of python example client
  pcb ........................................... design files of PCB
doc .................... the directory of LaTeX source codes of the thesis
  assets ................................ the thesis assets directory
  thesis.pdf ......................... the thesis text in PDF format
  thesis.tex ......................... the thesis text in TEX format
```