VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF BIOMEDICAL ENGINEERING

MOBILNÍ SYSTÉM PRO MONITOROVÁNÍ SPORTOVNÍ AKTIVITY

MOBIL SYSTEM FOR MONITORING OF SPORTS ACTIVITY

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. FILIP MALEŇÁK

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. JIŘÍ ROZMAN, CSc.

SUPERVISOR

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav biomedicínského inženýrství

Diplomová práce

magisterský navazující studijní obor
Biomedicínské a ekologické inženýrství

Student:	Bc. Filip Maleňák	***ID:***	126756
Ročník:	2	***Akademický rok:***	2014/2015

NÁZEV TÉMATU:

Mobilní systém pro monitorování sportovní aktivity

POKYNY PRO VYPRACOVÁNÍ:

1) Prostudujte základní metody aplikované při monitorování aktivit sportovců při jejich aktivitách. 2)Seznamte se s odpovídajícími technickými řešeními. 3) Vypracujte literární rešerši z dané oblasti. 4) Vypracujte systémový návrh technického řešení zařízení umožňujícího monitorovat tepovou frekvenci a frekvenci dýchání pomocí mobilního telefonu. 5) Ověřte systémový návrh na aplikaci mobilního telefonu a systému Arduino. Výsledky řešení doložte zobrazením sledovaných biosignálů s vyhodnocením. 6) Zhodnoťte dosažené výsledky zařízení v oblasti sportovních tréninků.

DOPORUČENÁ LITERATURA:

[1] MARK, D., LAMARCHE, J. IPhone SDK - průvodce vývojem aplikací pro iPhone a iPod touch. Computer Press, Brno, 2010, ISBN 978-80-251-2820-6
[2] EVANS,B. Beginning arduino programming. Apress New York, 2011, ISBN 978-1-4302-3777-8

Termín zadání:	9.2.2015	***Termín odevzdání:***	22.5.2015

Vedoucí práce: doc. Ing. Jiří Rozman, CSc.
Konzultanti diplomové práce:

prof. Ing. Ivo Provazník, Ph.D.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Cílem této diplomové práce je rozbor metod používaných při monitorování aktivit sportovců a popis odpovídajících technických řešení se zaměřením především na sledování tepové a dechové frekvence. Je předloženo technického řešení pro monitorování vybraných biologických parametrů prostřednictvím volně dostupných HW a SW nástrojů. Navržený systém umožňuje monitorování sportovní činnosti pomocí mobilního telefonu s operačním systémem iOS.

KLÍČOVÁ SLOVA

Arduino, Dechová frekvence, EKG, iPhone, iOS, mHealth, Mobilní telefon, Polar, Tepová frekvence, Trénink, Variabilita srdeční frekvence, SportsBrain

ABSTRACT

The aim of this master's thesis is to analyse methods that are used for monitoring athlete's activities and a description of current technical solutions with a focus on heart rate and respiratory rate monitoring. The presented solution shows the possibilities of using available open source SW and HW technologies and their implementation in the design of an integrated tool for monitoring sports activities with the iOS operating system.

KEYWORDS

Arduino, Respiratory rate, ECG, iPhone, iOS, mHealth, Mobile phone, Polar, Heart rate, Training, Heart rate variability, SportsBrain

MALEŇÁK, F. *Mobilní systém pro monitorování sportovní aktivity*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2015. 103 s. Vedoucí diplomové práce doc. Ing. Jiří Rozman, CSc..

Prohlášení

Prohlašuji, že svoji diplomovou práci na téma "Mobilní systém pro monitorování sportovní aktivity" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením tohoto projektu jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne …… .....

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce doc. Ing. Jiřímu Rozmanovi, CSc. za odbornou pomoc a podnětné rady podávané vždy přátelsky a se zájmem, kdykoli to autor této práce potřeboval.

Dále děkuji Univ.-Prof. Dr. Raineru Schubertovi za podnětné konzultace a rady při zpracování části diplomové práce na univerzitě UMIT v Rakousku. V neposlední řadě děkuji svým rodičům a rodině za podporu v průběhu celého studia na vysoké škole.

V Brně dne …… .....
podpis autora

Obsah

Úvod

Monitorování biologických parametrů v průběhu sportovního tréninku představuje jeden z nejrychleji rostoucích segmentů v oblasti mHealth. Důležitou součástí těchto systémů je způsob prezentace výsledků měření pomocí mobilních aplikací s možnostmi jejich analýzy a sdílení. Nejčastější metodou monitorování sportovní aktivity je stanovení tepové frekvence v reálném čase. Měřením tepové frekvence lze stanovit hodnotu aktuální fyzické zátěže a následně odvodit individuální pásma intenzit sportovního tréninku. Podstatným parametrem, reflektujícím zátěž, je však i frekvence dechová. Tento biologický parametr je u většiny současných mobilních aplikací opomíjen. Společné měření dechové a tepové frekvence poskytuje uživateli rozšířenou informaci o sportovní aktivitě a vede tak k jejímu efektivnějšímu vyhodnocení .

Cílem této diplomové práce je návrh a realizace technického řešení, umožňující podporu uživatele v průběhu sportovního tréninku, prostřednictvím mobilní aplikace. Na rozdíl od jiných řešení, která také využívají pro monitorování sportovní činnosti hrudní pás spojený s mobilní aplikací, je systém zaměřen na současné monitorování tepové a dechové frekvence. Navržený systém dále umožňuje monitorovat čas, vzdálenost, či geografickou polohu uživatele. Nejedná se o návrh zcela nového hardwarového řešení, ale předložení možností využití dostupných technologií a jejich vzájemné propojení pro vytvoření komplexního monitorovacího systému.

1. Úvod do problematiky

Záznam a analýza biologických parametrů v průběhu fyzické zátěže představují nedílnou součást sportovní medicíny. S novými vědecko-technickými objevy rostou i možnosti záznamových zařízení. Především poznatky v elektrotechnice a informačních technologiích, spolu s možnostmi digitalizace biologických dat, vedly k rozvoji měřící a záznamové techniky. Využití informačních a komunikačních technologií v rámci zdravotní péče označujeme pojmem *eHealth* (e-zdravotnictví). V této práci se zaměříme na oblast eHealth, která naznala velkého rozvoje v posledních patnácti letech. Jedná se o oblast využití mobilních technologií a mobilních zařízení (mobilní telefony, PDA, tabletové počítače aj.) v rámci záznamu a vyhodnocení biologických dat. Tato oblast elektronického zdravotnictví je označována anglickým názvem *mHealth* (mobilní zdravotnictví).

V oblasti mobilních zařízení dnes hovoříme tzv. Post-PC éře. Tento pojem bývá používán v souvislosti s postupným nahrazováním klasických stolních a přenosných počítačů mobilními telefony či tablety. Ty dnes umožňují uživatelům vykonávat činnosti, dříve charakteristické pro standardní počítače. Důvody hledejme ve vývoji hardwarových komponent, jako jsou operační paměť či procesor, které díky nárůstu výkonu a současné miniaturizaci nabízejí v mobilních zařízeních výkon srovnatelný s levnějšími počítači. Výkonné hardwarové (dále jen HW) komponenty jsou však zbytečné, pokud jejich výkonu nedokáže využít operační systém. Vývoj mobilních operačních systémů je dalším důvodem, proč stále větší procento uživatelů přechází (v rámci některých činností) od práce na počítači k práci na mobilním telefonu či tabletu.

1.1 Mobilní operační systém

Mobilní operační systém lze volně definovat jako operační systém (OS), který představuje základní programové vybavení počítače či mobilního zařízení. Umožňuje ovládání daného zařízení a poskytuje rozhraní pro další aplikace. Na rozdíl od "desktopových" OS jsou mobilní OS přizpůsobeny pro různá mobilní zařízení. Umožňují tak pohodlné ovládání na často omezené velikosti displeje. Významným milníkem ve vývoji mobilních OS bylo uvedení mobilního telefonu iPhone společností Apple v roce 2007. Jednalo se o první mobilní telefon vybavený OS, který umožňoval jednoduché a intuitivní dotykové ovládání pouze s pomocí prstu a byl zcela přizpůsoben velikosti dotykového displeje. Jiní výrobci, jako např. společnost Microsoft, se systémem Windows Mobile nabízeli upravené standardní OS, které umožňovaly ovládání mobilních zařízení zejména pomocí speciálního dotykového pera.

Současný trh mobilních operačních systémů je reprezentován především třemi technologickými společnostmi. Tabulka 1 na následující straně udává přehled významných mobilních operačních systémů a jejich zastoupení na trhu dle [28].

Tab. 1: Zastoupení mobilních OS na trhu ve třetím čtvrtletí 2014 podle [28]

Společnost	Mobilní OS	Podíl
Google	Android	84,40 %
Apple	iOS	11,70 %
Microsoft	Windows Phone	2,90 %
Jiné	-	1 %

Z výše uvedených informací je patrné, že nejpoužívanějším mobilním OS je v současné době OS Android společnosti Google. Důvodem je především nižší cena a velký počet dostupných mobilních zařízení, vybavených tímto systémem. Při splnění požadavků společnosti Google může kterýkoli výrobce integrovat a upravit systém Android pro svá zařízení. Opakem je systém iOS od společnosti Apple, který je striktně uzavřený a pracuje pouze na zařízeních této firmy. Otázka uzavřenosti či otevřenosti OS je řešena od prvních komerčních počítačů a každý z přístupů má své klady i zápory. Kompromis v oblasti mobilních OS dnes představuje OS Windows Phone společnosti Microsoft. Pro integraci systému do svých mobilního zařízení musí výrobce splnit stanovené podmínky a integrovat systém v nezměněné podobě. Řešení předkládané v této práci je založeno na integraci systému iOS společnosti Apple. Systém iOS je i navzdory své uzavřenosti, dobrou platformou pro testování nových technologií. Vývojář má po splnění regulací společnosti Apple zajištěnu bezproblémovou práci s certifikovaným příslušenstvím. Cílem diplomové práce je tak nejen návrh a realizace systému pro monitorování sportovní aktivity, ale i demonstrace možností, které iOS poskytuje pro vývoj specifického technického řešení.

1.2 Operační systém iOS

V roce 2007 společnost Apple poprvé představila koncept mobilního telefonu iPhone. Operační systém telefonu byl zcela přizpůsoben dotykovému ovládání a nesl označení iPhone OS. Základem systému se stal operační systém MAC OS X, který je od roku 2001 základem počítačů Apple a představuje alternativu systémům Microsoft Windows či Linux OS. Dnes se pro označení iPhone OS používá celosvětově zažité zkrácené označení "iOS". Systém využívají mobilní zařízení iPhone, iPad a iPod Touch.

Podobně jako u standardních "desktopových" OS jsou schopnosti MT do značné míry definovány i možnostmi dodatečně doinstalovaného SW. Tyto telefony se pak označují jako tzv. "chytré". Jediným legálním místem, kde lze nové aplikace vyhledávat a instalovat v rámci systému iOS, je internetový obchod "AppStore", přístupný přímo z MT pomocí stejnojmenné aplikace. Společnost Apple si tímto způsobem chrání svůj systém proti škodlivému softwaru. Každá aplikace je před zařazením do volně přístupné databáze prověřena schvalovacím procesem. Pro vývojáře to představuje poněkud komplikovanější způsob distribuce aplikací. Avšak pro uživatele toto omezení znamená jistotu nezávadnosti stahovaného SW. Díky tomu systém iOS patří dlouhodobě k těm nejlépe fungujícím co do bezpečnosti a stability nabízených mobilních aplikací.

iOS SDK (Software Development Kit) jsou nástroje umožňující vývoj aplikací pro platformu iOS. Základem je vývojové prostředí počítačového programu Xcode. Tento program je dostupný pouze pro počítače Apple, vybavené operačním systémem MAC OS X a umožňuje vývoj aplikací především v programovacím jazyku Objective-C (nově i Swift). Objective-C byl navržen na počátku 80. let 20. století. Vycházel z jazyka označovaného SmallTalk-80. Jednalo se o další vrstvu jazyka C, doplněnou o možnosti tvorby objektů a manipulaci s nimi. Součástí zdrojového kódu mobilní aplikace tak může být i část psaná ve standardním jazyce C. Ve většině případů však využíváme všech dostupných vymožeností objektově orientovaného programování. V roce 1988 byl tento jazyk licencován společností NEXTSTEP a po jejím odkoupení společností Apple v roce 1996 se jazyk Objective-C a prostředí NEXTSTEP stalo základem nového operačního systému MAC OS X a později také iOS. Tato práce nemá za cíl detailní popis způsobu vývoje mobilních aplikací. Programovací jazyk Objective-C a vývojové prostředí Xcode je zde využito jako nástroj pro realizaci navrhovaného řešení mobilního systému pro monitorování sportovní aktivity. Přesto jsou pro lepší pochopení výkladu návrhu systému v kapitole 4 popsány některé základní koncepty vývoje mobilních aplikací pro systém iOS. Více informací o programování pro operační systém iOS je uvedeno v [15].

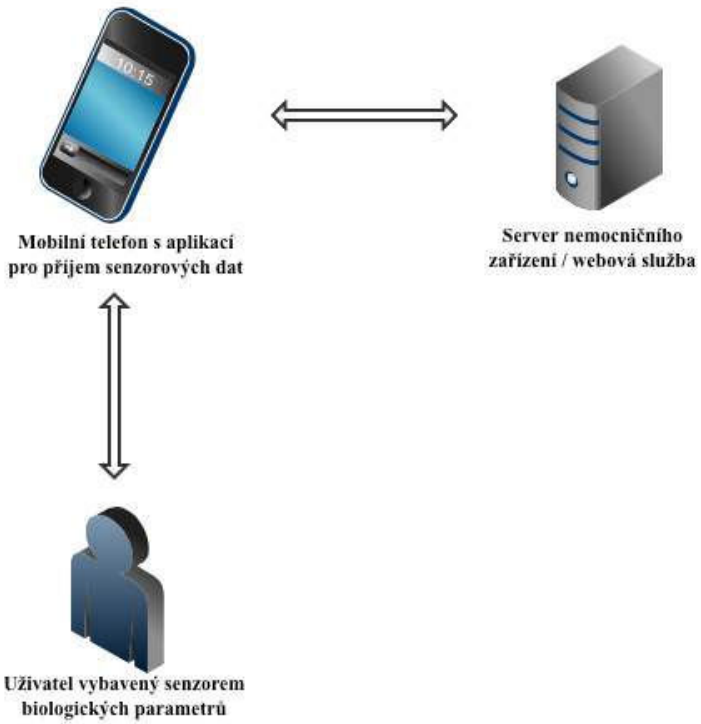
Jak bylo řečeno, je nezávisle na OS, charakteristikou mobilních operačních systémů možnost instalace aplikací, které rozšiřují funkce mobilního zařízení. V oblasti mHealth dnes existuje celá řada mobilních aplikací pro monitorování a analýzu biologických dat.

1.3 Mobilní aplikace v oblasti mHealth

Trh mobilních aplikací v oblasti mHealth zaznamenal v posledních pěti letech významný nárůst. Za poslední 2 roky se počet vydaných aplikací v oblasti mHaelth pro dvojici nejrozšířenějších mobilních OS (Android, iOS) více než zdvojnásobil na počet přesahující 100 000 (údaj platný k prvnímu čtvrtletí roku 2014). Celkový obrat odvětví dosáhl v roce 2013 jen na území Spojených států amerických 2,4 miliardy dolarů a odborníci předpokládají růst tohoto trhu na hodnotu 26 miliard dolarů do roku 2017. Zdrojem příjmů pro tvůrce aplikací již nebude jednorázové zpoplatnění stažení aplikace, jako je tomu doposud, ale její napojení na další služby. Těmito službami bude podpora významných zdravotních zařízení a poskytovatelů zdravotní péče – vzdálené monitorování pacientů, objednávkový systém, elektronický záznam pacienta atd. Současní uživatelé mobilních aplikací v oblasti mHealth jsou především chroničtí pacienti (31%) a uživatelé sportovních a fitness aplikací (28%). Současný stav využití mobilních aplikací v oblasti mHealth podle [25] je zobrazen v příloze A. V budoucnu lze předpokládat výrazný nárůst počtu mobilních aplikací určených pro lékaře či vědce [25].

1.4 mHealth systémy pro snímání biologických parametrů

Zaměříme-li se na mobilní aplikace sloužící ke sběru a analýze biologických signálů je nutné zmínit, že samotné mobilní zařízení dokáže snímat jen minimum těchto dat. Ve většině případů využívá mobilní telefon (MT) externích senzorů prostřednictvím speciální aplikace. Dnes se tyto senzory a jiná zařízení pro měření nejen biologických dat často označují pojmem "Nositelná elektronika" (z anglického Wearables). Nejrozšířenější je v současné době oblast sportu a fitness. Pozorujeme však i první "seriózní" certifikované senzory pro vzdálené monitorování pacientů. Takové řešení pak umožňuje například vzdálené monitorování EKG signálu. Lékařské zařízení v takovém případě zapůjčí uživateli pouze senzor. Ten je pak spojen s mobilním telefonem, prostřednictvím kterého odesílá snímaná data lékaři. Není tak nutné provádět nákup komplexního monitorovacího zařízení obsahujícího jak senzorovou, tak výpočetní a telemetrickou část. Lze využít výpočetních a komunikačních prostředků, které nabízí mobilní telefon. Obr. 1 znázorňuje standardní uspořádání systému v oblasti mHealth.



Obr. 1: Obecné schéma systému v oblasti mHealth

Pod senzorovou částí si lze představit senzor tepové frekvence, EKG, krevního tlaku či glykémie. Data senzoru jsou odeslána do mobilního zařízení například pomocí Bluetooth. Mobilní telefon pak slouží k jejich vyhodnocení a případnému odeslání do nemocničního zařízení nebo, zejména u sportovních a fitness senzorů, do webové služby, umožňující uživateli archivaci a sdílení naměřených sportovních výsledků.

1.5 Cíle diplomové práce

Cílem této práce je návrh a realizace mobilního systému pro monitorování sportovní aktivity. Práce bude zaměřena na nejrozšířenější oblast mHealth, představující sportovní a fitness systémy. Většina dostupných systémů a sportovních aplikací využívá jako primární a jediný biologický parametr pro stanovení fyzické zátěže tepovou frekvenci. Důležitým parametrem určujícím hodnotu aktuálního fyzického zatížení organizmu je však i hodnota frekvence dechové. Společné měření dechové a tepové frekvence poskytuje uživateli rozšířenou informaci o sportovní aktivitě, kterou tak lze lépe vyhodnotit a analyzovat.

Navržený systém (popsaný v kapitole 4) se skládá z části senzorové a části výpočetní. Jak již bylo uvedeno, důležitými parametry pro monitorování sportovní činnosti je tepová a dechová frekvence. Systém využívá senzoru elektrické aktivity srdce a senzoru snímajícího pohyby hrudníku při dýchání pro výpočet uvedených biologických parametrů. Stanovení tepové frekvence je možné přímo z měření elektrických projevů srdeční činnosti. Stanovení dechové frekvence pak podobně přímo z monitorování pohybů v oblasti hrudníku.

Naměřená senzorová data jsou přenášena pomocí otevřené mikroprocesorové platformy Adruino do mobilního telefonu s operačním systémem iOS. Integrace naměřených dat s MT probíhá prostřednictvím speciálně navržené mobilní aplikace "SportsBrain". Aplikace umožňuje nejen zobrazení aktuálních hodnot měřených biologických parametrů, ale nabízí i nástroje pro jejich vyhodnocení a sdílení. Schéma základních funkčních prvků systému SportsBrain je zobrazeno na obr. 2.

Obr. 2: Schéma funkčních prvků navrženého systému pro monitorování sportovní aktivity

V následujících kapitolách budou uvedeny poznatky, související s návrhem a následnou realizací systému. Dále pak možnosti dostupných senzorů pro měření tepové a dechové frekvence, možnosti jejich integrace v rámci mikroprocesorové platformy Arduino a operačního systému mobilního telefonu iPhone. Výsledkem diplomové práce je realizace systému popsaná v kapitole 4.

2. Biologické parametry pro monitorování sportovní aktivity

Biologické veličiny mají rozhodující význam pro řízení sportovního tréninku a charakterizují fyziologické stavy v průběhu tréninkového zatížení lépe, nežli délka či rychlost pohybu. Měření parametrů oběhového systému je základem pro hodnocení sportovní činnosti. Tyto parametry představuje především stanovení tepové a dechové frekvence. Při návrhu mobilního systému pro monitorování sportovní aktivity bude kladen důraz především na tyto dva biologické parametry [17].

Samostatnou kapitolou je funkční diagnostika, představující v mnoha lékařských oborech nepostradatelnou součást péče o pacienty. Mezi nejčastěji využívaná funkční vyšetření patří ergometrické a spirometrické vyšetření kardiovaskulárního a respiračního systému. Zejména tělovýchovné lékařství využívá řadu let funkční vyšetření při zátěži pro hodnocení sportovní výkonosti a nastavení optimálních tréninkových dávek. Dle definice „je funkční vyšetření pracovní postup, kterým se pomocí lékařské přístrojové techniky získávají biologické signály charakterizující a popisující funkční stav orgánů nebo organizmu“. Z této definice je zřejmé, že cílem předkládané diplomové práce není vývoj lékařského přístroje pro funkční vyšetření. Cílem je návrh jednoduššího monitorovacího zařízení, které bude mít pro uživatele nikoli diagnostický, ale informativní charakter [26].

Monitorování parametrů tepové a dechové frekvence je důležité především u vytrvalostních sportů. Zejména zde je žádoucí udržení konstantního výkonu po delší časový interval (obvykle nad 20 minut). Toto je dáno zejména charakterem měřených biologických veličin, které poskytují vypovídající data zvláště v aerobní části kondičního programu, či při intervalovém tréninku. Mezi cílové vytrvalostní sporty patří běh, cyklistika, veslování, běžecké lyžování, triatlon apod. [5].

2.1 Vliv zátěže na funkci organizmu

Pro pochopení vlivu fyzické zátěže na funkci organizmu zde budou uvedeny některé základní poznatky a vazby regulačních mechanizmů, které v těle působí. Jako hodnotu tzv. *Bazálního metabolismu* označujeme výdej energie uvolněné v těle v klidu (fyzickém i psychickém) a to 12 – 14 hodin po jídle, při konstantní teplotě okolí. Část energie je využívána pro udržení životně důležitých orgánů, větší část je pak přeměněna v teplo pro udržení tělesné teploty. Udávaná hodnota je 1kcal/kg/h.

Biologický systém lidského těla je chápán jako vysoce organizovaný komplex regulačních systémů. Základem regulace je několik autonomních řídících center s řadou dílčích regulačních subsystémů. Autonomní nervový systém propojený s CNS řídí základní fyziologické funkce jako krevní oběh, dýchání, regulace teploty či trávení.

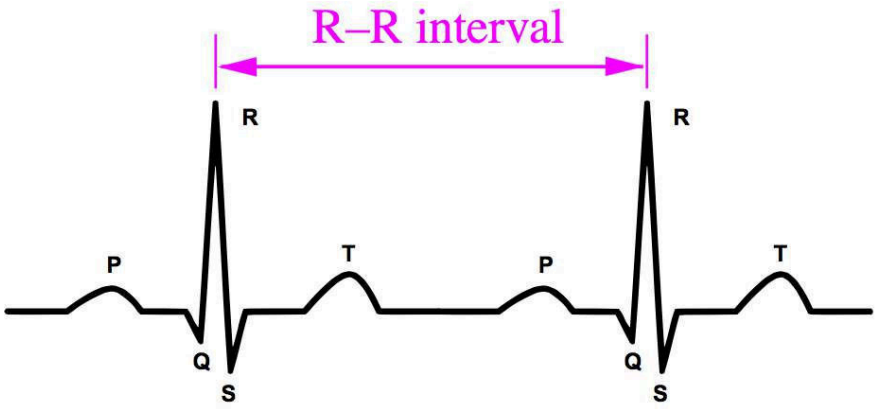
Organizmus je o stavu svých orgánů neustále informován prostřednictvím receptorů. Regulace krevního oběhu, a tedy i hodnota tepové frekvence, je v těle zajišťována třemi mechanizmy. Prvním je *velikost srdeční kontrakce*, řízena množstvím krve vracející se do srdečních komor (žilní návrat) a plněním srdce. Druhý mechanizmus představují *arteriální receptory tlaku*, které řídí srdeční aktivitu pomocí vegetativního nervového systému s pomocí CNS. Regulaci ovlivňují nervové dráhy a faktory humorální. Tyto regulace působí na sobě nezávisle a navzájem se doplňují (ovlivňují především sílu srdeční kontrakce). Třetím faktorem je *objem cirkulující krve*.

Při zátěži jsou zvýšené nároky na přívod kyslíku tkáním zajišťovány vzestupem srdečního výdeje (součin tepové frekvence a tepového objemu), redistribucí srdečního výdeje (zvýšením přívodu krve pracujícím orgánům) a vzestupem kyslíkové extrakce (zvýšením arteriovenózní diference kyslíku ve svalech). Informace o tepové frekvenci však neposkytuje úplnou informaci o reakci oběhového systému na zátěž. Její výhodou v rámci monitorování sportovní aktivity je však její relativně snadné určení. Lineárně narůstá s rostoucí zátěží až na úroveň svého maxima.

Z oblasti mozkového kmene vycházejí rytmické podněty dechovým svalům, které umožňují cyklické střídání nádechu a výdechu. Pro dýchací systém je charakteristické velké množství zpětných vazeb. Ty reagují na změny vnějšího a vnitřního prostředí tak, aby byla udržena optimální výměna plynů a stálost vnitřního prostředí. Úkolem dýchacích orgánů je především výměna plynů v krvi. Vdechové a výdechové centrum v prodloužené míše je řízeno pomocí chemoreceptorů v cévním systému. V průběhu zátěže dochází rychlejším metabolickým změnám v důsledku zvýšené spotřeby kyslíku. Snižuje se tak jeho parciální tlak. Naopak parciální tlak oxidu uhličitého v krvi roste. Tím působí na dýchací centra a vede k intenzivnější ventilační činnosti [26].

2.2 Křivka EKG

Nejčastějším neinvazivním způsobem monitorování srdeční aktivity je záznam elektrických projevů srdeční činnosti. Tato diagnostická technika, označovaná jako elektrokardiografie, využívá pro sledování srdečních kontrakcí akčních potenciálů srdce, šířících se na povrch těla. Postup elektrického vzruchu srdeční tkání a časově proměnné rozhraní mezi aktivovanou tkání a tkání v klidu vyvolává časově proměnné elektromagnetické pole v okolí srdečního svalu. Grafický záznam časové závislosti rozdílu elektrických potenciálů snímaných elektrodami, umístěnými nejčastěji na povrchu těla označujeme jako elektrokardiogram. Tímto záznamem tak dostáváme informaci o elektrických procesech probíhajících v srdečním svalu, které úzce souvisí s intenzitou fyzické zátěže. Vlastní křivka elektrokardiografického signálu (EKG) je zobrazena na obrázku (obr. 3) na následující straně [26].



Obr. 3: Ukázka křivky EKG dvou srdečních revolucí, znázornění intervalu R-R [16]

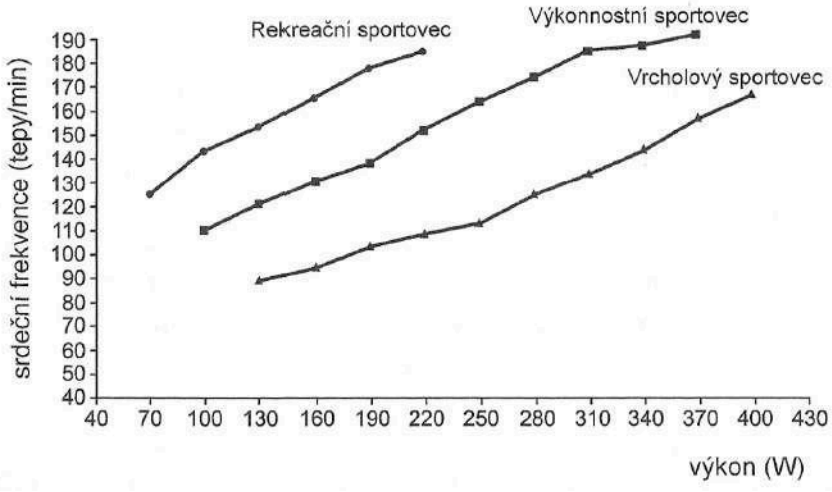
Křivka EKG přímo reflektuje významné fáze srdeční revoluce. První vlnu P lze interpretovat jako vzruchovou aktivitu sinoatriálního uzlíku, která se šíří depolarizací svalovinou síní. Mezi vlnou P a komplexem QRS je izoelektrický úsek PQ odpovídající zpomalení vedení vzruchu v atrioventrikulárním uzlu. Následuje komplex QRS, který reprezentuje postupnou depolarizaci mezikomorové přepážky, srdečního hrotu a konečně srdečních bází. Úsek ST je označován jako období stabilní aktivity srdce (fáze "plató"). Poslední vlna T odpovídá postupné repolarizaci svaloviny komor [16].

Významným parametrem, popsaným v rámci hodnocení křivky EKG signálu, je *interval R-R*. Tento interval odpovídá době mezi dvěma srdečními kontrakcemi. Je sice možné detekovat vzdálenosti i jiných významných bodů v záznamu EKG. Významný komplex QRS a především vlna R však poskytuje ideální parametry pro snadnou automatickou detekci a je i základním parametrem pro analýzu variability srdeční frekvence (viz kap. 3.2.3). Velikost R-R intervalu nese především důležitou informaci o hodnotě tepové frekvence, kterou lze stanovit podle vzorce:

$$\mathrm{TF} = \frac{60}{\mathrm{RR\ interval}}\ [\mathrm{tep/minuta}] \qquad (1)$$

2.3 Tepová frekvence

Hodnota tepové frekvence (dále jen TF) udává počet úplných srdečních kontrakcí (systola a diastola) za jednu minutu. Velmi rychle reaguje na zatížení organizmu (zejména svalstva) a je spolehlivou veličinou pro posouzení intenzity zatížení. Základním modelem testování sportovce na základě TF je stupňovitě rostoucí zátěž. Úroveň výkonnosti sportovců či pacientů lze usuzovat ze strmosti nárůstu TF. Srovnání křivky nárůstu TF u sportovců různé výkonosti je na obrázku (obr. 4). Plochý nárůst TF je charakteristický pro trénovaného jedince [17].



Obr. 4: Charakter nárůstu TF u sportovců různé výkonnostní úrovně [17].

V souvislosti s monitorováním tepové frekvence jsou podstatné dva základní parametry – *klidová* a *maximální* tepová frekvence. Maximální TF udává maximální možnou frekvenci, se kterou je srdce schopno fyziologicky pracovat. Klidovou TF chápeme jako frekvenci srdečních kontrakcí za minutu v klidu (fyzickém i psychickém). Maximální TF je u jedince neměnná. Jinak tomu je u klidové TF, ta se v průběhu života a důsledkem tréninku mění (obvykle klesá s rostoucí výkoností). V souvislosti se systémy pro záznam a analýzu sportovní aktivity je důležitá znalost maximální TF (TF_{max}). Z hodnoty TF_{max} jsou odvozeny individuální pásma intenzit, kterými se systémy řídí při vedení uživatele v průběhu sportovního tréninku [5].

S parametrem TF souvisí i pojem *spotřeba kyslíku* - VO_2. Pro sportovní trénink je důležitý především parametr VO_{2max}. Ten vyjadřuje nejvyšší množství kyslíku, které je jedinec schopen spotřebovat (obvykle při maximální tělesné zátěži). Na rozdíl od TF_{max} hodnota VO_{2max} s tréninkem roste. Jak TF tak hodnota VO_2 jsou spolehlivé údaje pro stanovení hodnoty aktuální sportovní zátěže. Odezva TF odpovídá námaze kardiovaskulárního systému na distribuci kyslíku. VO_2 zahrnuje TF, práci respiračního systému a využití kyslíku ve svalech. Vztah mezi uvedenými parametry je nelineární a zatímco stanovení TF_{max} je v praxi relativně jednoduché, stanovení VO_{2max} je možné pouze ve specializované laboratoři. V této diplomové práci se proto zaměříme na hodnocení výkonosti především pomocí měření TF [5].

2.3.1 Maximální tepová frekvence

V literatuře se často uvádí vzorec pro stanovení individuální TF_{max} výpočtem ze znalosti věku testovaného jedince. Hodnota je dle [26] a dle WHO z roku 1971 stanovena vzorcem:

$$\mathrm{TF_{max}} = 220 - \mathrm{věk} \quad (\pm 10\%) \tag{2}$$

Ovšem představa, že každý může odečíst svůj věk od hodnoty 220 a spolehlivě tak určit svou hodnotu TF_{max} je mylná. Na základě výrazu (2) získáme pouze orientační hodnotu individuální TF_{max}. Řada systémů pro monitorování sportovní aktivity s touto hodnotou však počítá a odvozuje odpovídající individuální pásma intenzit sportovní zátěže. Takový přístup je korektní, avšak sportovní trénink v tomto případě nebude veden zcela přesně. Přesnou hodnotu individuální TF_{max} získáme měřením VO_2 v průběhu zátěžového testu ve specializované laboratoři Pro většinu nevrcholových sportovců však postačí provedení jednoduchého zátěžového testu, díky kterému jsme schopni stanovit naši hodnotu TF_{max} také velice přesně. Na tomto místě je důležité zmínit, že je třeba provést individuální zátěžový test pro stanovení TF_{max} vždy samostatně pro jednotlivé sportovní aktivity. Zde uvedeme dvojici doporučených zátěžových testů podle [5] pro cyklistiku a běh, které využívá i navržený mobilní systém [5].

Stanovení TF_{max} běžeckého tréninku

Základem pro stanovení individuální maximální tepové frekvence je uvedení organizmu do stavu maximální možné zátěže v rámci vybrané sportovní aktivity. Zátěžový test pro stanovení individuální TF_{max} pro běžecký trénink podle [5] provedeme následovně:

1. Provedeme zahřívací fázi volným během alespoň 1,5 km.
2. Zahájíme zátěžový test výběhem tak, abychom na konci kola atletické dráhy či na konci mírného kopce délky přibližně 500 m vyvinuli maximální možnou rychlost.
3. Běžíme odpočinkovým tempem alespoň 2 minuty a následně opakujeme rychlé kolo na atletickém ovále či do mírného kopce.
4. Běžíme odpočinkovým tempem alespoň 2 minuty a následně naposledy opakujeme rychlé kolo na atletickém ovále či do mírného kopce. Na konci tohoto třetího úseku se dosažená TF přibližně rovná individuální TF_{max}, které jsme schopni v rámci běhu dosáhnout.

Stanovení TF_{max} cyklistického tréninku

Maximální hodnoty tepové frekvence dosahované při cyklistice se ve většině případů liší od TF_{max} při běhu. Pro správné stanovení tréninkových pásem je tedy nutné provedení samostatného zátěžového testu. Zátěžový test pro stanovení TF_{max} cyklistického tréninku podle [5] provedeme následovně:

1. Provedeme zahřívací fázi volnou jízdou alespoň 5 km.
2. Zahájíme zátěžový test rychlým výjezdem do kopce délky přibližně 1 km. Posledních 100 m vysedneme ze sedla a snažíme se vyvinout maximální možnou rychlost.

3. Jedeme odpočinkovým tempem přibližně 3 km a rychlý výjezd do stoupání zopakujeme.
4. Jedeme odpočinkovým tempem přibližně 3 km a rychlý výjezd do stoupání naposledy zopakujeme. Na konci tohoto třetího úseku se dosažená TF přibližně rovná individuální TF_{max}, které jsme schopni v rámci cyklistiky dosáhnout.

Na základě znalosti individuální TF_{max}, určené výpočtem či zátěžovým testem, je nyní možné stanovit tréninková pásma intenzit sportovní aktivity. Popisu jednotlivých tréninkových pásem v rámci běžeckého a cyklistického tréninku je věnována následující podkapitola.

2.3.2 Pásma intenzity tréninku

Tělesnou zdatnost a úroveň tréninku lze podle [5] rozdělit do pěti základních složek. Těmito složkami jsou – *pásmo zotavení, pásmo základní vytrvalosti, pásmo tempové vytrvalosti, pásmo speciální vytrvalosti a pásmo rychlosti*. Každá z uvedených složek tělesné zdatnosti se rozvíjí při specifické intenzitě tělesné zátěže, kterou lze stanovit na základě měření aktuálních hodnot TF. Ta zde představuje nejefektivnější ukazatel intenzity tělesné zátěže a její monitorování vede k udržení správného tréninkového pásma.

Zotavení (pásmo I) je úroveň tělesné zátěže, při které nepřesáhne hodnota TF 60% TF_{max}. Trénink v tomto pásmu přispívá k rychlejší a efektivnější regeneraci.

Základní vytrvalost (pásmo II) představuje úroveň tělesné zátěže, při které se hodnota TF pohybuje v rozmezí 60 - 75% TF_{max}. Typické cvičení pro rozvoj základní vytrvalosti je souvislý trénink (dlouhé pomalé úseky). Tento typ tréninku je doporučován pro každého, nezávisle na fyzické zdatnosti pro udržení tělesného zdraví.

Tempová vytrvalost (pásmo III) je úroveň tělesné zátěže při které se hodnota TF pohybuje v rozmezí 75 - 85% TF_{max}. Smyslem této kondiční přípravy je adaptace kardiovaskulárního a dýchacího systému tak, aby pracoval pod zátěží, ale bez přepětí. Typickým cvičením jsou zde například rovnoměrné úseky 40 až 45 minutového cvičení.

Speciální vytrvalost (pásmo IV) představuje schopnost pohybovat se závodní rychlostí při minimální spotřebě kyslíku a energie. Hodnota TF se zde pohybuje v rozmezí 85 - 95% TF_{max}. Příkladem je intervalový trénink.

Rychlost (pásmo V) udává schopnost organizmu pohybovat se velkou rychlostí v krátkém časovém intervalu za současné maximální tolerance zvýšené koncentrace laktátu ve svalech. Rychlostní schopnosti se rozvíjejí při TF 95 - 100% TF_{max}. Charakteristické jsou zde krátké intervalové úseky tréninku s maximální intenzitou následované dlouhým odpočinkem. Hodnoty TF pro různé úrovně zátěže jsou spolu s dalšími informacemi uvedeny v tabulce 2 [5].

Tab. 2: Tréninková pásma intenzity podle [5]

Pásma TF	Index Zatížení	Úroveň zátěže	Tempo	Energetický zdroj	Energetický proces	Složka zdatnosti
I	do 60 %	nízká	pomalé	tuky	aerobní	zotavení
II	60 -75 %	nízká	pomalé	převážně tuky	aerobní	základní vytrvalost
III	75 - 85 %	střední	střední	cukry a tuky	aerobní a anaerobní	tempová vytrvalost
IV	85 - 95 %	vysoká	rychlé	převážně cukry	anaerobní	speciální vytrvalost
V	95 - 100 %	velmi vysoká	sprint	výhradně cukry	anaerobní	rychlost

2.4 Dechová frekvence

Pod pojmem dechová frekvence (DF) rozumíme počet respiračních cyklů (nádechů a výdechů) za jednu minutu. Rytmické podněty dechovým svalům, které umožňují cyklické střídání nádechu a výdechu, vycházejí z oblasti mozkového kmene. Pro respirační systém je charakteristické velké množství zpětných vazeb, které reagují na změny vnějšího a vnitřního prostředí tak, aby byla zajištěna optimální výměna plynů a stálost vnitřního prostředí. Aktuální hodnota DF tak reflektuje potřebu organizmu na výměnu plynů, která v rámci fyzické zátěže roste v souvislosti s biochemickými ději přeměny energie [26].

Za fyziologické hodnoty DF v klidu u dospělých osob se považuje frekvence 12 – 18 dechů za minutu. U trénovaných sportovců bývá nižší. Ženy mají hodnotu DF vyšší než muži. U obou pohlaví je DF závislá i na věku. Zatímco fyziologické hodnoty DF u novorozenců jsou uváděny v rozmezí 30 – 60 dechů za minutu, u kojenců je za fyziologické považována hodnota v rozmezí 20 – 40 dechů za minutu. Pro předškolní věk fyziologické hodnoty DF dále klesají na hodnotu 20 – 30 dechů za minutu. Starší školní věk je charakteristický dalším poklesem na hodnoty 16 – 25 dechů za minutu. Důvodem postupného snižování DF společně s růstem jedince je nárůst kapacity plic. Při fyzické zátěži roste DF individuálně v závislosti na trénovanosti a pohlaví jedince. Tabulka 3 udává průměrné hodnoty DF v závislosti na stupni fyzické zátěže dospělého organizmu.

Tab. 3: Hodnoty DF v závislosti na úrovni fyzické zátěže [18]

Úroveň zátěže	Hodnota DF [dechů/min.]
nízká	15 - 25
střední	25 - 35
vysoká	35 - 45
velmi vysoká	45 - 60

Hodnota DF nemusí vždy korespondovat se zvýšením dechového objemu (celkové množství vdechnutého vzduchu v rámci jednoho vdechu) a nemusí také zcela odpovídat aktuální fyzické zátěži. Důvodem je možnost částečně ovlivnit hodnotu DF vůlí, na rozdíl od hodnoty TF. Rostoucí fyzická zátěž nás však nutí navzdory vlastní vůli hodnotu DF zvýšit a dýchat hlouběji (roste dechový objem). Jednotkou DF je jeden dechový cyklus, udávaný jako doba pro nádech a doba pro výdech. Jako hodnota *dechového intervalu* je pak označován čas mezi jednotlivými respiračními cykly. Za normálních okolností začíná nádech (inspirium) ihned po ukončení předchozího výdechu (expirium) a objem vydechnutého vzduchu se rovná objemu vzduchu vdechnutému. Rychlost nádechu i výdechu je ovlivněna nejen aktuální potřebou organizmu na výměny plynů, ale také anatomickými parametry dýchací soustavy (odpor dýchacích cest). Na základě znalosti dechového intervalu je možné vypočíst hodnotu DF podle vzorce:

$$\mathrm{DF} = \frac{60}{\mathrm{resp.interval}}\ [\mathrm{dech/minuta}] \qquad (3)$$

V rámci hodnocení tepové a dechové frekvence v souvislosti s fyzickou zátěží hovoříme o tzv. *anaerobním prahu (AP)*. Ten je definován jako dlouhodobá maximální fyzická zátěž, kterou je možné udržet. S tréninkem uvedený práh roste a je možné jej odvodit na základě nelineárního nárůstu VO_2. Jak již bylo uvedeno, skutečnou hodnotu VO_2 je možné stanovit pouze ve speciální laboratoři sportovní medicíny [18].

Podobně jako TF_{max}, tak i DF_{max} reflektuje individuální parametr sportovce. V literatuře bohužel dosud nejsou dostatečně popsány postupy stanovení tréninkových pásem na základě měření DF. Důvodem je zřejmě i možnost do značné míry DF ovlivnit vůlí. Lze však konstatovat, že pokud by byl systém monitorující sportovní aktivitu vybaven nástroji pro měření jak tepové tak dechové frekvence a uživatel by se nesnažil svoji DF vůlí ovlivnit, dokázal by systém na základě těchto dvou měřených parametrů přesněji detekovat maximální fyzickou zátěž uživatele a tak i přesněji nastavit individuální pásma intenzity tréninku. Uvedené konstatování by však vyžadovalo statistické hodnocení, které není předmětem této práce. V této diplomové práci je za základní monitorovaný parametr brána hodnota TF. Pásma intenzity zatížení jsou systémem počítána taktéž na základě TF_{max}. Hodnota aktuální a průměrné DF poskytuje pouze rozšiřující informaci o průběhu sportovního tréninku.

3. Dostupná technická řešení pro monitorování sportovní aktivity

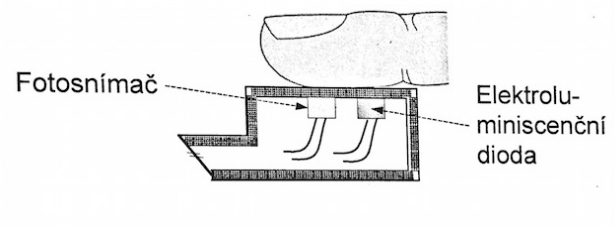
Cílem této práce je návrh komplexního technického řešení pro monitorování sportovní aktivity s využitím volně dostupných HW a SW nástrojů. Jak bylo uvedeno v předchozí kapitole, důraz je kladen především na sledování parametrů tepové a dechové frekvence. V úvodu této kapitoly budou popsána dostupná technická řešení pro monitorování uvedených parametrů. Následně budou představena technická řešení, umožňující měření v rámci otevřené mikroprocesorové platformy Arduino. Závěrem budou popsány možnosti mobilního operačního systému iOS a jeho integrace s platformou Arduino pro měření požadovaných biologických parametrů.

3.1 Monitorování tepové frekvence

Velké množství amatérských i profesionálních sportovců dnes pro zvýšení výkonnosti a přesnější vedení sportovního tréninku využívá monitorů tepové frekvence. Hodnota TF v reálném čase udává aktuální úroveň sportovní zátěže a představuje tak pro sportovce okamžitou zpětnou vazbu, reflektující průběh sportovního tréninku. Zařízení označovaná jako "sporttestery" patří k nejpoužívanějším prostředkům pro monitorování TF v průběhu sportovní činnosti. Tato zařízení jsou vyvinuta tak, aby umožňovala pohodlné a přesné snímání TF v průběhu sportovního tréninku a to s minimální úrovní šumu ve snímaném signálu.

3.1.1 Fotoelektrická pletysmografie

Na trhu je dnes zařízení stanovující hodnotu TF na základě foto optické metody. Princip metody, označované jako fotoelektrická pletysmografie, je založen na průchodu světla tkání. Zde dochází k jeho odrazu, rozptylu a absorpci. Změna tlaku krve, způsobená srdeční činností, se projeví změnou objemu kapilár. Tímto je ovlivněn i průchod světla tkání. Uspořádání systému pro měření TF na prstu je znázorněno na obrázku č. 5. Některé současné verze sporttesterů či monitorovacích zařízení ve formě hodinek mají tento monitorovací systém umístěn na vnitřní straně zápěstí. Uživatel tak v průběhu sportovního tréninku nosí pouze speciálně upravené hodinky. Takto detekuje TF například monitor aktivity společnosti Microsoft, označený jako "Microsoft Band", či "chytré" hodinky společnosti Apple, označené jako "Apple Watch" (obr. 6) [14].



Obr. 5: Možné uspořádání detektoru TF pomocí fotometrické pletysmografie [14]

Obr. 6: Apple Watch, obrazovka aktivity a senzorová část na vnitřní straně zápěstí [3]

Je využito závislosti absorpce světla na nasycení krve kyslíkem. Používají se infračervené diody (vlnové délky nad 800 nm). Na těchto vlnových délkách je pouze malý rozdíl v absorpci světla mezi okysličenou a neokysličenou krví. Amplituda vzniklého světelného toku je modulována objemem tekutiny (měnící se absorpcí světelného toku). Z detekce obálky je možné stanovit hodnotu tepové frekvence. Fotoelektrická pletysmografie je metoda stanovení TF velmi citlivá na pohyb uživatele. Sporttestery a zařízení založené na tomto principu tudíž poskytují při pohybu uživatele pouze orientační hodnoty TF [14].

3.1.2 Detekce založená na snímání elektrických projevů srdeční činnosti

O mnoho přesnější hodnoty tepové frekvence je možné stanovit na základě přímého monitorování elektrických projevů srdeční činnosti. Sposttestery založené na tomto principu umožňují velmi přesný výpočet aktuálních hodnot TF, nezávisle na pohybu uživatele. Systém se obvykle skládá z části monitorovací – *Hrudního pásu* a části výpočetní a zobrazovací. Zobrazovací část bývá většinou řešena formou *digitálních hodinek*, schopných přijímat a vyhodnocovat elektrické impulzy vysílané hrudním pásem (odpovídající elektrické aktivitě srdce).

Hrudní pás

Hrudní pás představuje senzorovou část moderních sporttesterů. Skládá se z páru elektrod umístěných po stranách filtrační a vysílací jednotky. Základem pro monitorování elektrických projevů srdeční činnosti je vodivé spojení mezi kůží a elektrodami (předpokládá se pocení uživatele). Elektrodový pár detekuje EKG křivku srdeční aktivity. Signál je přenášen do centrální jednotky hrudního pásu, kde dochází k filtraci a dalšímu zpracování (zejména komplexu QRS). Výsledný signál, obsahující informaci o srdeční aktivitě, je následně přenášen pro další vyhodnocení do výpočetní a zobrazovací části systému. Existuje mnoho různých provedení hrudních pásů pro měření TF. V této práci bude využito hrudního pásu společnosti Polar. Sporttestery této společnosti byly jedny z prvních pro komerční využití. Patří také dlouhodobě k nejpřesnějším [21].

Výpočetní a zobrazovací část

Výpočetní a zobrazovací část provádí příjem signálu z hrudního pásu. Tento signál obsahuje informaci o detekovaných kontrakcích srdce. Na základě znalosti časových intervalů mezi jednotlivými srdečními kontrakcemi lze stanovit aktuální hodnotu tepové frekvence. Nelze však s určitostí říci, jakými algoritmy jednotliví výrobci sporttesterů výpočet TF provádějí. Jedná se o duševní vlastnictví firem. Dále v textu bude uveden jednoduchý způsob, jakým lze pro účely této práce TF efektivně stanovit, máme-li k dispozici dekódované signály z hrudního pásu. Obr. 7 prezentuje špičku dnes dostupných sporttesterů - model Polar RC3 GPS.

Obr. 7: Sporttester Polar RC3 GPS – Výpočetní a zobrazovací část, hrudní pás [21]

Dnešní sporttestery obsahují kromě záznamu TF také další funkce jako je vedení nastaveného tréninku na základě aktuálních hodnot TF, záznam GPS polohy v průběhu tréninku, analýzu sportovního výkonu apod. [21].

Mobilní telefon jako součást záznamového systému

Trendem, který pozorujeme zejména v posledních letech, je integrace monitorů sportovní aktivity s mobilními telefony. Toto odvětví představuje jednu z nejrychleji rostoucích oblastí mHealth. Na trhu je velké množství hrudních pásů přenášejících data o srdeční aktivitě prostřednictvím nízkoenergetického Bluetooth LE (Low Energy) do mobilního telefonu. Mobilní telefon tak často nahrazuje výpočetní a zobrazovací část sporttesteru. Data jsou přijímána a zobrazena ve speciální aplikaci. Výhodou těchto systémů je možnost využití výpočetního výkonu, který dnes mobilní telefony nabízejí. Uživetel tak má k dispozici veškeré údaje o právě probíhající sportovní aktivitě, či následné detailní analýzy. Lze také využít mnoha vestavěných senzorů (GPS, kompas, gyroskop, akcelerometr...) pro sběr dalších informací, souvisejících se sportovní aktivitou. Nevýhodou je velikost mobilního telefonu, který je třeba mít v průběhu sportovní aktivity u sebe.

3.2 Monitorování dechové frekvence

Pro monitorování dechové frekvence lze využít několika principiálně odlišných přístupů. Jednou z možností je přímé monitorování procházejícího vzduchu ústy či nosní dírkou. Zde hovoříme o tzv. *pneumotachografii*. Pneumotachometrický snímač či hlavice je umístěna přímo do vzdušné cesty a její součástí je zpravidla i výměnný náustek. Je měřena rychlost a směr proudu vzduchu. Dostáváme tak nejen informace o dechové frekvenci, ale integrací hodnot i odpovídající respirační objemy. S touto metodou monitorování DF se setkáme především při zátěžové diagnostice a u tzv. spiroergometrického vyšetření (více informací v [19]). V rámci mobilních systémů monitorování sportovních aktivit tato metoda stanovení DF využita není [26].

Další možností monitorování dýchání je přímá detekce pohybu hrudníku. Využívají se zejména tyto metody:

3.2.1 Impedanční hrudní pás

Je založen na měření změny impedance tkáně v hrudním koši v průběhu dýchání. Elektrody jsou v hrudním pásu umístěny na protilehlých stranách v úrovni prsních bradavek. Často je toto měření kombinováno v rámci snímání EKG signálu. Je použit vysokofrekvenční proudový zdroj pro napájecí elektrody a je sledována nízkofrekvenční složka napětí korespondující s dýcháním [14].

3.2.2 Odporový hrudní pás

Základem je změna elektrického odporu protažením tenzometru. Změna elektrického odporu je následně vyhodnocena a odpovídá objemovým změnám hrudníku v průběhu dýchání. Na tomto principu pracují některé dostupné systémy pro monitorování DF v průběhu sportovního výkonu. Hrudní pás je v tomto případě podobný jako pás pro sledování TF. Na rozdíl od něj však není vybaven elektrodami ale senzory, které reagují na změnu protažení změnou elektrického odporu (obr. 8) [14].

Obr. 8: Odporový hrudní pás pro měření DF [20]

Jiný typ pásu může využívat vlastností tzv. vodivé gumy. I zde se odpor snímače mění vlivem protažení a detekují se tak přímo respirační pohyby hrudníku.

3.2.3 Respirační sinusová arytmie

Srdeční rytmus není pravidelný ale neustále se mění. Tepová frekvence je tak proměnlivá neboli variabilní. *Variabilita srdeční frekvence* – HRV (Heart Rate Variability) představuje odchylky v intervalech mezi jednotlivými kontrakcemi myokardu. Mezi základní pozorované oscilace srdeční frekvence patří oscilace v rytmu dýchání – *respirační sinusová arytmie* (RSA). Tuto závislost je možné charakterizovat tak, že v průběhu nádechu se trvání R-R intervalů zkracuje (roste srdeční frekvence) a v průběhu výdechu se trvání R-R intervalů prodlužuje (klesá srdeční frekvence). RSA vzniká podle [12] kombinací několika mechanizmů s různým podílem v závislosti na aktuální situaci. Iradiací impulzů z respiračního do kardiomotorického centra vzniká tzv. *centrální generátor RSA*, kdy inspirační neurony svojí aktivitou modifikují aktivitu vagových kardiomotorických preganglionových neuronů. V průběhu každého nádechu vzniká za účasti acetylcholinu inhibiční postsynaptický potenciál, kterým se hyperpolarizují kardioinhibiční preganglionové neurony. Tímto klesá vliv parasympatiku na srdce a srdeční frekvence se zrychluje. Existují tedy centrální mechanizmy vzniku RSA, způsobené interakcí respiračních a kardiovaskulárních center. Tento jev byl teoreticky popsán a prakticky ověřen na uměle ventilovaných pacientech. Bylo dokázáno, že fluktuace R-R intervalů kopíruje frekvenci umělé ventilace a je nejvýraznější u mladých lidí ve věku 18-24 let. Další pravděpodobné mechanizmy podílející se na vzniku RSA je možné nalézt v [12].

Pomocí spektrální analýzy je tak možné, za předpokladu dostatečného množství R-R intervalů, zpětně detekovat hodnoty DF. V ideálním případě by tak pro snímání TF a DF postačoval jediný hrudní pás, schopný detekovat srdeční kontrakce ve formě R-R intervalů. Metoda výpočtu DF na základě spektrální analýzy využitím jevu RSA však pro použití v rámci systému pro monitorování sportovní aktivity není vhodná. Cílem zařízení pro monitorování sportovní činnosti je poskytovat uživateli zpětnou vazbu o sledovaných biologických veličinách v reálném čase. Tuto podmínku odvození DF pomocí spektrální analýzy nesplňuje. Vždy je třeba nejdříve načíst dostatečný počet R-R intervalů a následně provést výpočetně náročnou spektrální analýzu, doprovázenou filtrací signálu R-R intervalů. Další nevýhodou je provázanost monitorování TF a DF. Jak hodnota TF, tak hodnota DF jsou v tomto případě stanoveny na základě stejného měření. Vyskytne-li se tak chyba v monitorování elektrické srdeční aktivity, ovlivní nejen hodnotu výsledné TF, ale i odvozenou hodnotu DF. V rámci návrhu systému je proto využito odporového detektoru. Je tak možné stanovit hodnotu DF s pomocí přímé detekce dechových pohybů uživatele a to nezávisle na monitorování TF.

3.3 Mikroprocesorová platforma pro monitorování biologických parametrů

Arduino je fyzickou open-source počítačovou platformu, skládající se z jednoduchého mikropočítače (vývojové desky) a vývojového SW prostředí pro zápis softwaru. Díky své malé konstrukci a možnostem použití standardního vyššího programovacího jazyka (vycházejícího z jazyka C) je Arduino ideální platformou pro vytváření mikroprocesorových projektů. Arduino si lze představit jako vstupní bránu pro připojení velkého množství senzorů či akčních členů (např. servomotor). Tento systém byl zvolen jako mikroprocesorová část diplomové práce z důvodu možnosti optimální integrace biologických senzorů. V této podkapitole bude uveden základní popis platformy Arduino. V podkapitole následující potom možnosti jejího využití pro monitorování srdeční činnosti a dechových pohybů.

3.3.1 Technické parametry vývojové desky Arduino UNO

Současná verze základní "vývojové desky" (z anglického development board) nese označení Arduino UNO (obr. 9). Tato deska byla využita pro základní koncepční návrh senzorové části systému pro monitorování sportovní aktivity. Z tohoto důvodu je zde uveden její podrobnější popis.

Hlavní součástí je mikroprocesor ATmega 328. Deska dále obsahuje čtrnáct digitálních vstupně/výstupních pinů. Šest z těchto pinů je možné použít pro pulsně šířkovou modulaci signálu – PWM (Pulse Width Modulation) představující diskrétní modulaci pro přenos analogového signálu. Piny pracují na napětí 5V s maximálním odběrem proudu 40mA. Pro datovou komunikaci jsou významné především piny č. 0 a 1. Tyto piny mohou být využity pro příjem (RX) a vysílání (TX) TTL sériových dat (viz dále). Parametry vývojové desky Arduino Uno jsou uvedeny v tabulce 4. Platforma Arduino poskytuje dále veškeré potřebné komponenty, jako jsou stabilizace zdroje či USB rozhraní. Jedná se tedy o ideální základ pro konstrukci mikroprocesorové části systému předloženého v této diplomové práci.

Tab. 4: Parametry vývojové desky Arduino UNO [2]

Arduino UNO			
Mikroprocesor	ATmega328	*DC proud na I/O pinech*	40 mA
Operační napětí	5 V	*DC proud na 3,3V pinu*	50 mA
Doporučené vstupní napětí	9 - 12 V	*Flash paměť*	32 KB
Limitní vstupní napětí	6 - 20 V	*SRAM*	2 KB
Digitální I/O piny	14 (6 umožňuje PWM)	*EEPROM*	1 KB
Analogové vstupní piny	6	*Taktovací frekvence*	16 MHz

Obr. 9: Vývojová deska Arduino UNO s procesorem ATmega328 [4]

3.3.2 Sériová komunikace

Mikroprocesor ATmega328 podporuje sériovou komunikaci prostřednictvím UART (Universal Asynchronous Receiver Transmitter) TTL (Transistor to Transistor Logic) na hladině 5V. Tato komunikace vychází ze standardu RS-232 a jak již bylo uvedeno, této komunikace lze využít na digitálních pinech 0 a 1 desky Arduino UNO. Komunikace je řízena čipem ATmega8U2, který umožňuje přesměrování sériové komunikace také skrze USB. Díky tomuto se Arduino jeví jako virtuální sériový port na počítači s operačním systémem Windows, OS X, nebo Linux. Prostřednictvím vývojového softwaru (Arduino IDE) je následně možné programování funkcí mikrokontroleru. Sériová komunikace na TTL se odehrává mezi limity 0 V a operačním napětím VCC (5 V nebo 3,3 V), kde VCC reprezentuje logickou "1" a 0 V logickou "0". Nejjednodušší standardní sériovou TTL komunikaci lze navázat mezi vývojovou deskou a počítačem prostřednictvím programu Arduino IDE podle [2] následovně:

```
void setup() {
 Serial.begin(9600);                          // 1
}
void loop() {
   while (Serial.available() <= 0) {          // 2
   Serial.println("Spojeni navazano");        // 3
   delay(300);
   }
Serial.println("Spojeni ukonceno");
while(1) { }
}
```

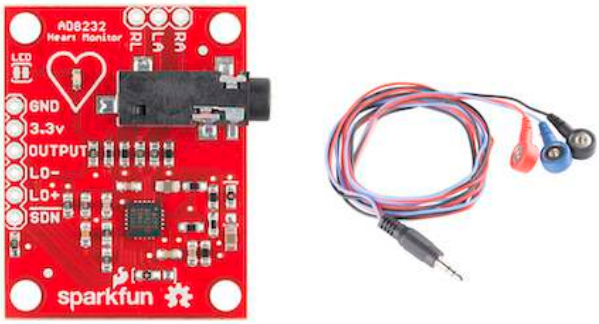
Programy platformy Arduino se skládají ze dvou základních celků – funkce *setup* a funkce *loop*. Vždy je nejdříve volána funkce *setup*, která nastavuje základní parametry, jako jsou použité piny (vstupy a výstupy). Následně je volána funkce *loop*, kde se ve smyčce zpracuje hlavní část programu. Bod **1** zdrojového kódu nastavuje rychlost přenosu dat. V bodu **2** dostáváme počet bajtů, které jsou k dispozici pro čtení. Pokud zde žádné nedostáváme, smyčka čeká na první příchozí bajt. Posledním krokem je navázání jednoduché sériové komunikace - bod **3** a odeslání textu "Spojeni navazano". Kód fakticky odesílá zprávu "Spojeni navazano" každých 300 ms, dokud nepřijme bajt (znak). V tomto momentě odesílání končí vytištěním "Spojeni ukonceno" [2].

3.4 Využití platformy Arduino pro monitorování srdeční činnosti

V předchozích kapitolách bylo uvedeno, že základním parametrem pro sledování a vyhodnocení sportovní aktivity, je tepová frekvence. Zde bude uvedena dvojice dostupných technických řešení pro monitorování srdeční činnosti pomocí zvolené mikroprocesorové platformy Arduino. Dále pak vyhodnocení využitelnosti řešení při návrhu systému pro monitorování sportovní aktivity.

3.4.1 Jednokanálový snímač EKG AD 8232

Modul AD 8232 (obr. 10) distribuovaný například společností Sparkfun Electronics slouží pro měření a předzpracování (filtraci) signálů elektrické aktivity srdce. Signál je měřen trojicí nalepovacích elektrod umístěných na těle uživatele ve specifické konfiguraci pro záznam jednoho svodu EKG. Provedení součástky umožňuje její snadné propojení s otevřenou vývojovou platformou Arduino. Zapojení modulu AD 8232 je uvedeno v tabulce 5.

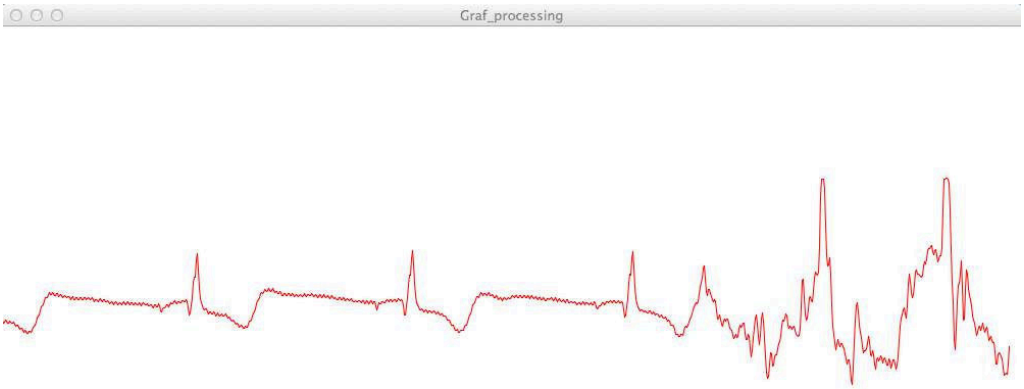


Obr. 10: Modul AD 8232 [1]

Tab. 5: Propojení modulu AD 8232 s vývojovou deskou Arduino [1]

AD 8232		
Pin	Funkce	Arduino - zapojení
GND	Zem	*GND*
3.3V	3.3V napájecí napětí	*3.3V*
OUTPUT	Výstupní signál	*A0*
LO-	Detekce kontaktu elektrod	*11*
LO+	Detekce kontaktu elektrod	*10*
SND	Vypnutí	/

Modul byl propojen a testován podle [1] s vývojovou deskou Arduino UNO. Analogový signál jedno-svodového EKG byl přiveden na analogový vstup vývojové desky Arduino. Digitální výstupy LO+ a LO- byly použity pro detekci kontaktu elektrod s kůží. Kompletní zdrojový kód pro zobrazení průběhu signálu EKG je dostupný v [1]. Hodnoty elektrické aktivity srdce byly přeneseny sériovou komunikací do počítače a zobrazeny pomocí programu Processing (obr. 11).



Obr. 11: Záznam EKG podle [1] pomocí modulu AD 8232 v klidu a při pohybu

Na základě záznamu průběhu signálu EKG je zřejmé, že modul AD 8232 není pro účely navrhovaného systému vhodný. Modul poskytuje validní data pouze je-li uživatel v naprostém klidu. Pohybem uživatele (druhá polovina záznamu) je elektrický signál znehodnocen a není možné provést rozměření intervalů R-R a následné odvození TF.

3.4.2 Čidlo RMCM-01

Hrudní pásy pro monitorování srdeční činnosti v rámci sportovní aktivity, vyvíjené firmou Polar, patří dlouhodobě k nejpřesnějším. Měření elektrické aktivity srdce je zde realizováno dvojicí elektrod tak, jak bylo popsáno v předchozí kapitole. Pro účely návrhu nových systémů, využívajících signály měřené pomocí hrudního pásu Polar, je dostupný bezdrátový přijímač RMCM-01 (obr. 12).

Obr. 12: Bezdrátový přijímač signálů hrudního pásu RMCM-01 [24]

Přijímač RMCM-01 je schopen detekovat signál kompatibilního hrudního pásu v okolí a zpracovat přijaté signály. Příkladem hrudního pásu kompatibilním s RMCM-01 je Polar T61, který je zobrazen na následující straně (obr. 13).

Obr. 13: Hrudní pás Polar T61 [22]

Připevněním hrudního pásu na tělo dojde k jeho aktivaci a detekci srdečních kontrakcí. Signál obsahující informaci o srdeční aktivitě je přenášen na kmitočtu 5,5 kHz a lze jej přijímat čidlem RMCM-01. Zde jsou na pinu HR generovány 1 ms pozitivní impulzy o velikosti 3V, kde každý impulz odpovídá jedné kontrakci srdce. Čas mezi příjmem dvou po sobě jdoucích impulzů tak odpovídá R-R intervalu signálu EKG. Technické parametry a popis pinů čidla RMCM-01 jsou uvedeny v tabulkách 6,7 [21].

Tab. 6: Popis pinů součástky RMCM-01 [21]

Pin	Popis	Pin	Popis
HR	3V 1ms impulz při každé detekci srdeční kontrakce	*WIDB_DET*	Nastavení (spojeno s VCC)
Reset	Reset	*FPLS*	Detekce všech impulzů hrudního pásu
OSC	Vstup 1 rezonančního krystalu	*LX2*	Terminál cívky antény
F32KIN	Vstup 2 rezonančního krystalu	*LX1*	Terminál cívky antény
		GND	Zem
OSC_ON	Výběr hodin (vsup/výstup)	*VCC*	Napájecí napětí

Tab. 7: Provozní parametry součástky RMCM-01 [21]

Parametr	Minimum	Typicky	Maximum	Jednotky
Napájecí napětí	2,5	3	4,4	V
Dosah	80	92	100	cm
Šířka pulzu HR		1		Ms
Šířka pulzu FPLS		6		ms
Přijímací frekvence		5,5		kHz
Frekvence vstupního krystalu	32,736	32,768	32,8	kHz
Provozní teplota	0	20	60	°C

Propojení součástky RMCM-01 s vývojovou deskou Arduino podle [24] je uvedeno v příloze B. Kompletní zdrojový kód pro výpočet a zobrazení tepové frekvence je dostupný v [24] a blíže popsán v kapitole 4. Aktuální hodnota TF je vypočtena pomocí vzorce (1) z R-R intervalu stanoveného vždy jako průměr vzdálenosti mezi osmi posledními detekovanými impulzy na pinu HR čidla RMCM-01. Výsledkem jsou hodnoty aktuální tepové frekvence přenášené sériovou komunikací do počítače (obr. 14).

Obr. 14: Hodnoty tepové frekvence určené na základě měření hrudním pásem T61 a detekcí signálu čidlem RMCM-01 podle [24]

Prezentovaná metoda záznamu a výpočtu tepové frekvence poskytuje validní data i v případě pohybu uživatele a je tak ideální pro použití v rámci návrhu systémů pro monitorování sportovní aktivity. Tato metoda tak bude použita i pro návrh a realizaci systému prezentovaného v této práci v kapitole 4.

3.5 Využití platformy Arduino pro monitorování respirace

Jak již bylo uvedeno v předchozích kapitolách, tepová frekvence není jediným parametrem reflektujícím aktuální hodnotu zátěže v průběhu sportovní aktivity. Úroveň fyzické zátěže lze stanovit také monitorováním aktuální hodnoty dechové frekvence. Zpětný výpočet hodnoty DF prostřednictvím spektrální analýzy R-R intervalů, využívající jevu respirační sinusové arytmie, není pro účely navrhovaného systému ideální. Tato metoda neposkytuje data v reálném čase a nehodí se tak pro sledování aktuální tréninkové zátěže. Zde bude proto popsána dvojice jiných dostupných řešení, které ve spojení s platformou Arduino poskytují informaci o aktuální hodnotě DF.

3.5.1 E-Health Sensor Platform V2.0

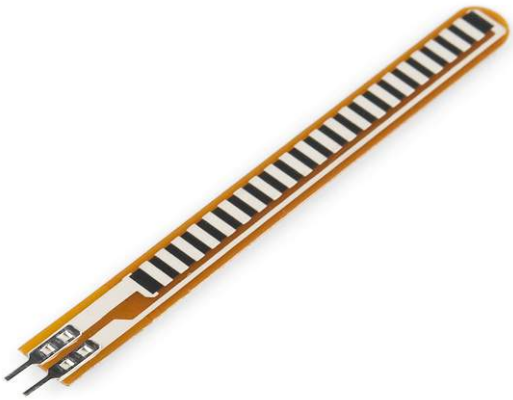
Představuje senzorovou nástavbu (Shield) pro mikroprocesorovou platformu Arduino, umožňující snímat biologická data prostřednictvím deseti senzorů. Jedním je i senzor měnící svůj elektrický odpor v souvislosti s množstvím obtékaného vzduchu. Umístěním tohoto senzoru do vzdušných cest uživatele tak lze přímo monitorovat jeho respirační aktivitu. V průběhu sportovního tréninku však není vhodné zatěžovat uživatele senzorem ve vzdušných cestách. Z tohoto důvodu není “shield“ E-Health pro návrh systému v této diplomové práci využit. Více informací lze nalézt v [7].

3.5.2 Flex senzor 2.2

V souvislosti s návrhem systému pro monitorování sportovní aktivity se jako ideální metoda pro monitorování DF jeví přímé snímání respiračních pohybů hrudníku. Takto nebude uživatel zatěžován pneumotachometrickým snímačem ve vzdušných cestách a snímání DF bude zároveň nezávislé na snímání TF. Jsme-li tedy schopni nalézt vhodný senzor, který detekci respiračních pohybů hrudníku provede s dostatečnou přesností, bude možné tyto pohyby ihned transformovat na aktuální hodnotu DF.

Vhodným senzorem je "Flex senzor 2.2" (obr. 15) od společnosti Spectra Symbol. Základním parametrem senzoru je schopnost měnit svůj vnitřní elektrický odpor v závislosti na velikosti ohybu. Aktivní část senzoru je složena z vrstvy ohebného plastu a vrstvy polymeru, obsahujícího vodivé částice. Pokud se senzor nachází v neohnutém stavu, částice polymeru vykazují elektrický odpor velikosti okolo 30 Ω. Ve stavu ohybu 90° se odpor částic mění na hodnotu okolo 50 Ω. Snímáním aktuální hodnoty el. odporu tak lze vyhodnotit stupeň ohybu senzoru. Senzor lze zapojit podle [8] jako dělič napětí s použitím rezistoru o velikosti vnitřního odporu 10 – 100 kΩ. Schéma zapojení senzoru do vývojové desky Arduino je znázorněno v příloze C. Rezistor a senzor vytvářejí dělič napětí VCC v poměru daném dvojicí jejich elektrických odporů. Detekcí velikosti napětí vstupujícího do analogového pinu vývojové desky Arduino je pak možné hodnotit velikost ohybu senzoru.

V případě, že je senzor ohybu vhodně umístěn na hrudník uživatele a je tak umožněna změna velikosti jeho ohybu v souvislosti s pohyby hrudníku v průběhu dýchání, je možné detekovanou změnu elektrického odporu senzoru považovat přímo za detekci dechových pohybů uživatele. Více o integraci senzoru Flex 2.2 v rámci systému pro monitorování sportovní aktivity je uvedeno v samostatné kapitole věnované návrhu systému (viz kap. 4).



Obr. 15: Flex Senzor 2.2 [8]

3.6 Integrace platformy Arduino s mobilním operačním systémem iOS

Cílem diplomové práce je návrh systému pro monitorování sportovní aktivity. V kapitole 1 byl uveden základní koncept navrhovaného systému. Záměrem je propojit senzory umožňující monitorování biologických parametrů prostřednictvím platformy Arduino s mobilním telefonem vybaveným systémem iOS pro zobrazení a vyhodnocení naměřených dat. V následujících podkapitolách budou uvedeny možnosti propojení mobilního telefonu vybaveného tímto operačním systémem s platformou Arduino.

3.6.1 Mobilní telefon iPhone

Pro návrh technického řešení je využit mobilní telefon iPhone ve verzi 4s. Tento model poskytuje veškeré potřebné technologie jako je Bluetooth 4.0 LE (Low Energy) či GPS s podporou A-GPS. Technická specifikace zařízení iPhone 4s je uvedena v tabulce 8. Samotné konstrukční provedení modelu 4s znázorňuje obrázek č. 16. Navržená mobilní aplikace (viz dále) je kompatibilní se všemi modely telefonu iPhone s verzí systému iOS 8.0 a novější.

Tab. 8: Technická specifikace mobilního telefonu iPhone 4s [11]

Mobilní sítě a připojení	GSM (850, 900, 1800, 1900 MHz) UMTS/HSPDA/HSUPDA (850, 900, 1800, 1900 MHz) GPRS, EDGE, 3G Wi-Fi (802.11n, 2.3 GHz) Bluetooth 4.0
Metody lokalizace	GPS, asistované GPS GLONASS Digitální kompas Wi-Fi (WPS) Podpora lokalizace v celulární síti
Displej	Rozlišení 640 x 960 px Úhlopříčka 3,5" Dotykový kapacitní (umožňuje vícenásobný dotyk)

Mobilní telefon nenabízí standardní USB propojení, ale specifický "30–pin" konektor. Tento konektor umožňuje, podobně jako standard USB, současný přenos dat a napájení mobilního telefonu. Pomocí tohoto konektoru byla navázána sériová komunikace mobilního telefonu iPhone 4s s vývojovou deskou Arduino UNO pro účely testování dílčích verzí navrženého systému pro monitorování sportovní aktivity.

Obr. 16: Konstrukční provedení mobilního telefonu iPhone 4s [11]

3.6.2 Propojení platformy Arduino s mobilním telefonem iPhone

Propojení platformy Arduino s mobilním telefonem iPhone je možné od verze 3 operačního systému iOS. I současná verze iOS 8 obsahuje možnosti propojení systému s externími HW komponentami prostřednictvím "External Accessory Framework". Avšak přísné regulace společnosti Apple ve směru k výrobcům externího HW znamenají velmi omezené množství legálních možností propojení platformy Arduino s iOS. Jednou z takových možností propojení je vývoj zcela nového HW řešení. Tento proces však vyžaduje nejen návrh vlastního zařízení, ale také nutnost schválení společností Apple v rámci MFi (Made For iPod) programu. Proces schvalování je velmi zdlouhavý s nejistým výsledkem. Proto bylo v této práci v rámci návrhu, testování a realizace využito dvojice dostupných a společností Apple již schválených řešení [2].

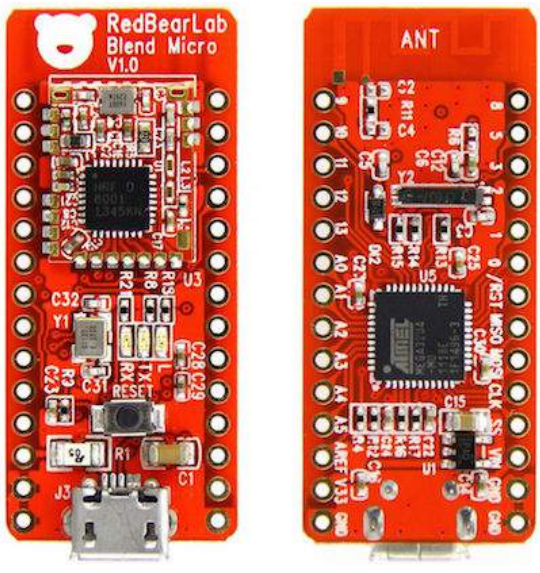
Sériový kabel Redpark

Sériový kabel od společnosti Redpark (obr. 17) je jednoduchým řešením k propojení zařízení s operačním systémem iOS s platformou Arduino. Společnost Redpark dodává spolu s tímto kabelem i knihovnu funkcí programovacího jazyka Objective-C. Ta umožňuje navázat sériovou komunikaci mezi propojenými zařízeními prostřednictvím 30-pinového konektoru mobilního telefonu iPhone protokolem TTL. Komunikace mezi mobilním telefonem a vývojovou deskou probíhá podobně jako v případě komunikace s počítačem (viz kap. 3.3.2). Zde však není Arduino propojeno USB rozhraním, ale pomocí pinu č. 0 (RX - příjem) a pinu č. 1 (TX - vysílání) prostřednictvím kabelu Redpark přímo s mobilním telefonem. Nevýhodou tohoto propojení je nutnost dodatečného napájení vývojové desky Arduino například pomocí standardní 9 V baterie. Piny 0 a 1 pro TTL komunikaci neumožňují přenášet data a zároveň poskytovat napájení tak, jako je tomu v případě USB rozhraní [27].

Obr. 17: Sériový kabel Redpark pro TTL komunikaci mezi MT iPhone a platformou Arduino [27]

Blend Micro

Součástka Blend Micro od společnosti RedBearLab (obr. 18) představuje unikátní spojení mikroprocesoru platformy Arduino a modulu Bluetooth 4.0 Low Energy, certifikovaným pro navázání komunikace s operačním systémem iOS prostřednictvím dodávané knihovny funkcí. Propojením této součástky s požadovaným senzorem je tak vytvořeno jednoduché řešení pro zpracování signálů a jejich následné odeslání do mobilního telefonu iPhone s pomocí Bluetooth. Právě tato vývojová deska je využita jako základ navrhovaného technického řešení systému pro monitorování sportovní aktivity (viz kap. 4). Propojení této vývojové desky s mobilním telefonem iPhone je možné od verze 4s. Starší verze tohoto telefonu nedisponují nízkoenergetickým rozhraním Bluetooth 4.0 [6].



Obr. 18: Integrovaná vývojová deska Blend Micro od společnosti RedBearLab [6]

V rámci vývojové desky je integrován procesor ATmega 32U4 společně s čipem Nordic nRF8001 pro komunikaci pomocí Bluetooth 4.0. Rozmístění pinů vývojové desky Blend Micro je uvedeno v příloze D, technická specifikace je uvedena v tabulce 9.

Tab. 9 Parametry vývojové desky Blend Micro [6]

Blend Micro			
Mikroprocesor	Atmel ATmega 32U4	*Možnosti propojení*	Bluetooth 4.0 LE micro USB Serial (TX/RX) I2C SPI
Bezdrátový čip	Nordic nRF8001		
Operační napětí	3.3V		
Doporučené vstupní napětí	3.3 - 12 V		
Taktovací frekvence	8 MHz		
Flash paměť	32 KB	*SRAM*	2,5 KB
Digitální I/O piny	17 (6 umožňuje PWM)	*EEPROM*	1 KB

V této (především teoretické) části práce byly uvedeny veškeré nezbytné informace založené na literární rešerši a testování dostupných technologií, které předcházejí samotné realizaci systému. Realizace systému pro sledování sportovní aktivity využívajícího mikroprocesorové platformy Arduino a mobilního telefonu iPhone je předmětem následující kapitoly č. 4.

4. Návrh a realizace systému SportsBrain

Základní myšlenkou navrženého systému pro monitorování sportovní aktivity je použití dostupných HW a SW komponent a jejich vzájemné propojení pro vytvoření funkčního technického řešení. Návrh konkrétní realizace je založen na poznatcích uvedených v předchozích kapitolách. Byla ověřena dvojice řešení pro monitorování srdeční činnosti v rámci mikroprocesorové platformy Arduino. Dále byly uvedeny možnosti pro monitorování respirace v průběhu sportovního tréninku a na závěr také dostupná řešení pro spojení platformy Arduino s mobilním operačním systémem iOS. Navržený systém nese pracovní název "SportsBrain" a jak bylo uvedeno v kapitole 1.5, je složen ze tří funkčních prvků:

Obr. 19: Schéma funkčních prvků systému SportsBrain pro monitorování sportovní aktivity

4.1 Senzorová část

Cílem navrženého systému je umožnit uživateli monitorování důležitých biologických parametrů v průběhu sportovního tréninku. V předchozích kapitolách bylo uvedeno, že vhodným parametrem reflektujícím aktuální fyzickou zátěž, je hodnota tepové frekvence. Bylo také popsáno, že důležitým parametrem je i aktuální hodnota frekvence dechové. Parametr DF je dnešními systémy, které využívají mobilních aplikací pro zpracování a zobrazení dat, opomíjen. Pro nezávislé monitorování DF a TF využívá navržený systém dvojice senzorů integrovaných do jediného hrudního pásu, vybaveného elastickým popruhem pro fixaci na hrudníku uživatele.

4.1.1 Senzor srdeční činnosti

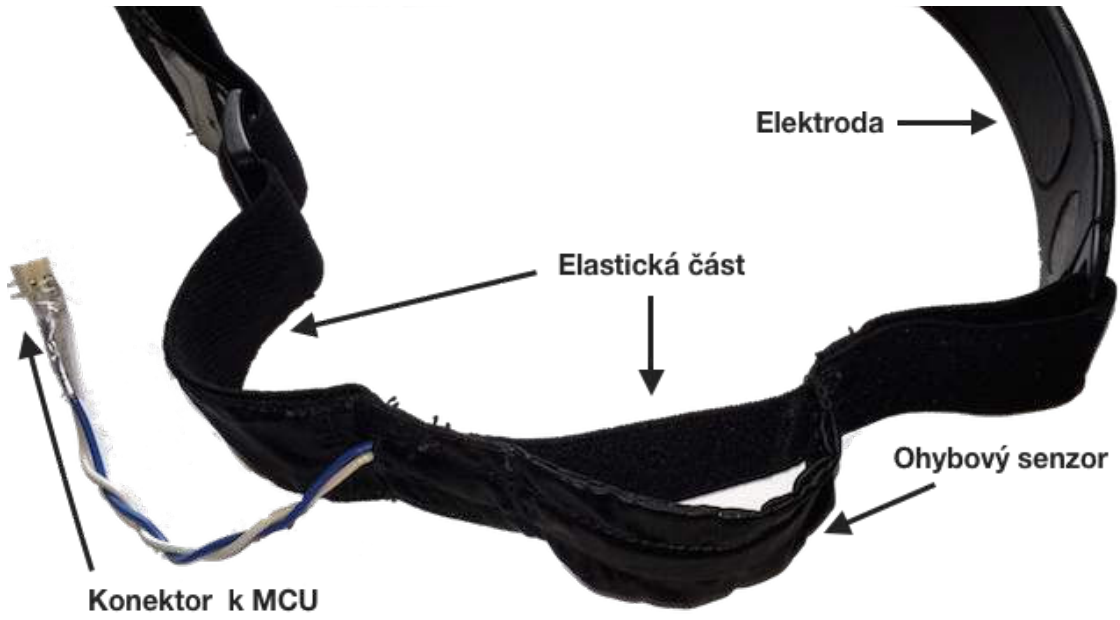
Na základě provedení testu dvojice měřících systémů, monitorujících elektrickou aktivitu srdce, je vhodnější použít hrudního pásu Polar T61. Tento hrudní pás je přímo navržen pro záznam elektrické aktivity srdce v průběhu sportovní činnosti, kde se potýkáme se značným počtem rušení způsobených především pohybem uživatele. Z tohoto důvodu je záznam jednokanálového EKG signálu pomocí součástky AD 8232 pro stanovení TF nevhodný. Signál je zde pohybem uživatele znehodnocen natolik, že není možné provést jeho rozměření pro stanovení R-R intervalů. Rušení je způsobeno především pohybem elektrod a myopotenciály.

Elektrické signály srdeční aktivity (měřené dojicí elektrod) jsou předávány do centrální jednotky hrudního pásu. Zde dochází k filtraci a zvýraznění vlny R signálu EKG. Výsledkem je informace o srdeční aktivitě přenášená prostřednictvím bezdrátového spojení na frekvenci 5,5 kHz.

4.1.2 Senzor respirace

V kapitole 3.5.2 byla uvedena teze, že vhodným umístěním senzoru Flex 2.2 na hrudník uživatele je možné snímat respirační pohyby. Základní myšlenkou je především fakt, že obvod hrudníku se v souvislosti s respiračními pohyby mění. Tyto změny jsou v průběhu sportovní činnosti natolik velké, že umístíme-li ohybový senzor na elastickou část hrudního pásu, je možné detekovat změnu elektrického odporu senzoru v průběhu dýchání.

Největší "natažení" elastické části hrudního pásu lze pozorovat v souvislosti s dechovými pohyby po stranách hrudníku. Ohybový senzor je proto integrován do elastického popruhu v místě, kde je popruh připojován na pevnou elektrodovou část. Pro ochranu a zamezení nežádoucího ohybu je senzor integrován ve voděodolné látce. Dvojice vodičů je vyvedena a zakončena konektorem pro zapojení senzoru v rámci platformy Arduino. Integrace ohybového senzoru pro detekci respiračních pohybů do elastické části hrudního pásu Polar T61 je znázorněna na obrázku (obr. 20).



Obr. 20: Integrace ohybového senzoru do elastické části hrudního pásu Polar T61

4.2 Mikroprocesorová část

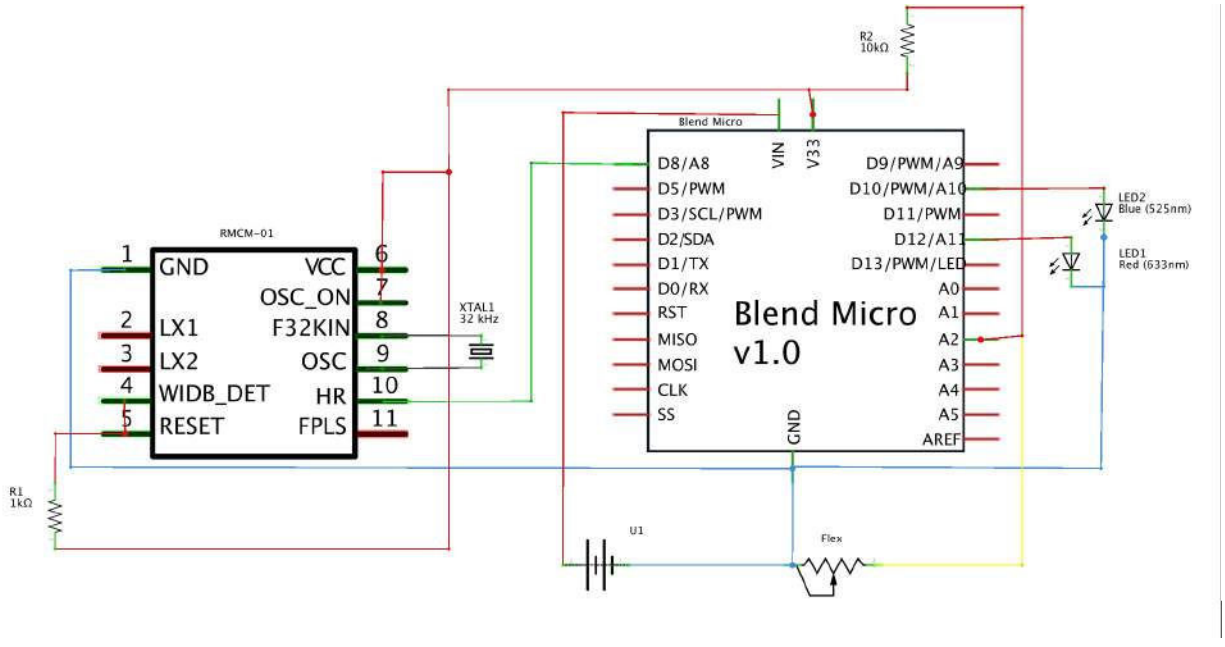
Mikroprocesorová platforma Arduino je ideálním nástrojem pro integraci jednočipového počítače (mikrokontroléru, MCU) v rámci navrženého systému. V předchozím textu byla popsána dvojice MCU platformy Arduino. Jednalo se o základní model Arduino UNO, který je možné propojit s mobilním telefonem pomocí sériového kabelu společnosti Redpark. Spojení využívá TTL komunikaci na pinech 0,1 vývojové desky a 30-pin konektor na mobilním telefonu iPhone 4s. Druhou možností je využití vývojové desky Blend Micro společnosti RedBearLab. Deska Blend Micro představuje jedinečné spojení mikroprocesoru platformy Arduino (ATmega 32U4) s modulem Bluetooth 4.0 Low Energy (Nordic nRF8001). Tato vývojová deska umožňuje bezdrátové spojení s mobilními telefony vybavenými operačními systémy iOS a Android. Je tak ideální pro integraci v rámci navrženého mobilního systému pro monitorování sportovní aktivity. V této kapitole je uveden popis zapojení senzorové části systému a to včetně obvodového schématu a rozboru zdrojového kódu programu MCU.

4.2.1 HW řešení MCU

HW řešení systému je založeno na využití vývojové desky Blend Micro Ta je propojena s čidlem RMCM-01 pro detekci elektrické srdeční aktivity podobně, jako v kapitole 3.4.2. Řešení je také založeno na propojení vývojové desky s ohybovým senzorem Flex pro detekci respiračních pohybů tak, jak je popsáno v kapitole 3.5.2.

Obvodové schéma

Obvodové schéma mikroprocesorové části systému je na obrázku č. 21. Schéma bylo vytvořena v programu "fritzing", který umožnuje pouze Americké značení odporů.



Obr. 21: Obvodové schéma mikroprocesorové části systému

Interakce s uživatelem

Interakce mikroprocesorové části s uživatelem je zajištěna dvojicí LED diod. Aktivací modré diody (LED2) je indikován přenos dat prostřednictvím Bluetooth komunikace. Červená dioda (LED1) je aktivována po krátký časový interval vždy, když je systémem detekována srdeční kontrakce. Diody jsou přímo propojeny s vývojovou deskou Blend Micro a to na digitálních pinech D10 (modrá dioda) a D12 (červená dioda). Aktivace je za výše popsaných podmínek řízena pomocí SW v MCU (viz kap. 4.2.2).

Zapojení RMCM-01

Čidlo RMCM-01 (viz kap. 3.4.2) provádí příjem signálu na frekvenci 5,5 kHz. Zde je obsažena informace z hrudního pásu o elektrické srdeční aktivitě. Čidlo obsahuje cívku a vlastní miniaturní MCU (obr. 22), které slouží a zabezpečuje zpracování přijatého signálu hrudního pásu a pro následné generování el. impulzů na pinu "HR". Integrovaný MCU však pro svou práci vyžaduje 32 kHz rezonanční krystal pro nastavení vnitřních hodin. Krystalem jsou propojeny piny "F32KIN" a "OSC". Piny "WIDB_DET" a "RESET" jsou přes odpor R1 velikosti 1 kΩ propojeny s pinem V33 vývojové desky Blend Micro. Tento pin poskytuje napětí 3,3 V, které odpovídá definovanému rozmezí vstupního napětí čidla RMCM-01 (viz tab. 17). Piny "OSC_ON" a "VCC" jsou taktéž propojeny na napětí 3,3 V, které poskytuje vývojová deska. Pin "GND" představuje zem a je spojen se stejným pinem na desce. Konečně výstupní pin "HR" je propojen s digitálním pinem D8 vývojové desky pro detekci elektrických impulzů odpovídajících srdeční aktivitě. Zpracování detekovaných impulzů a výpočet aktuální hodnoty TF je úkolem SW MCU.

Obr. 22: Vnitřní část čidla RMCM-01 [21]

Zapojení senzoru Flex 2.2

Ohybový odporový senzor Flex 2.2 byl popsán v předchozí kapitole (kap. 3.5.2). Zapojení senzoru do mikroprocesorové platformy Arduino je velmi jednoduché. V rámci obvodového schématu je senzor značen jako proměnný odporový prvek. Senzor je propojen jako dělič napětí s použitím rezistoru R2 o velikosti vnitřního odporu 10 kΩ. S tímto rezistorem je při použitém vstupním napětí 3,3 V dosaženo nejlepší citlivosti detekce ohybu. Výstup senzoru je propojen s analogovým pinem A2 vývojové desky pro detekci hodnot odpovídajících jeho ohybu. Zpracování detekovaných hodnot a odvození respiračních pohybů hrudníku uživatele je úkolem SW MCU.

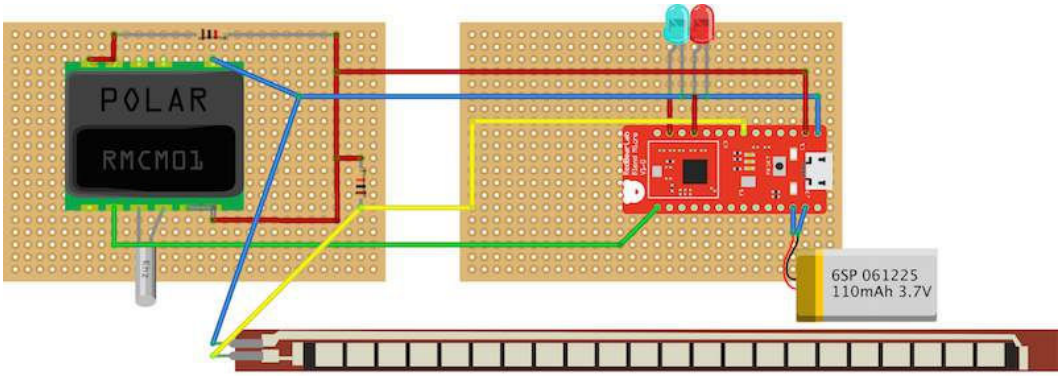
Napájení MCU

Pro napájení vývojové desky na základě specifikace (viz tab. 9) je možné zvolit lithium iontový akumulátor malých rozměrů s kapacitou 110 mAh, poskytující napětí 3,7 V. Specifikace použitého akumulátoru je uvedena v tabulce (tab. 10). Systém je schopen s touto baterií pracovat nepřetržitě po dobu alespoň jedné a půl hodiny. Výhodou použitého akumulátoru je také možnost dobíjení pomocí USB.

Tab. 10: Parametry akumulátoru [23]

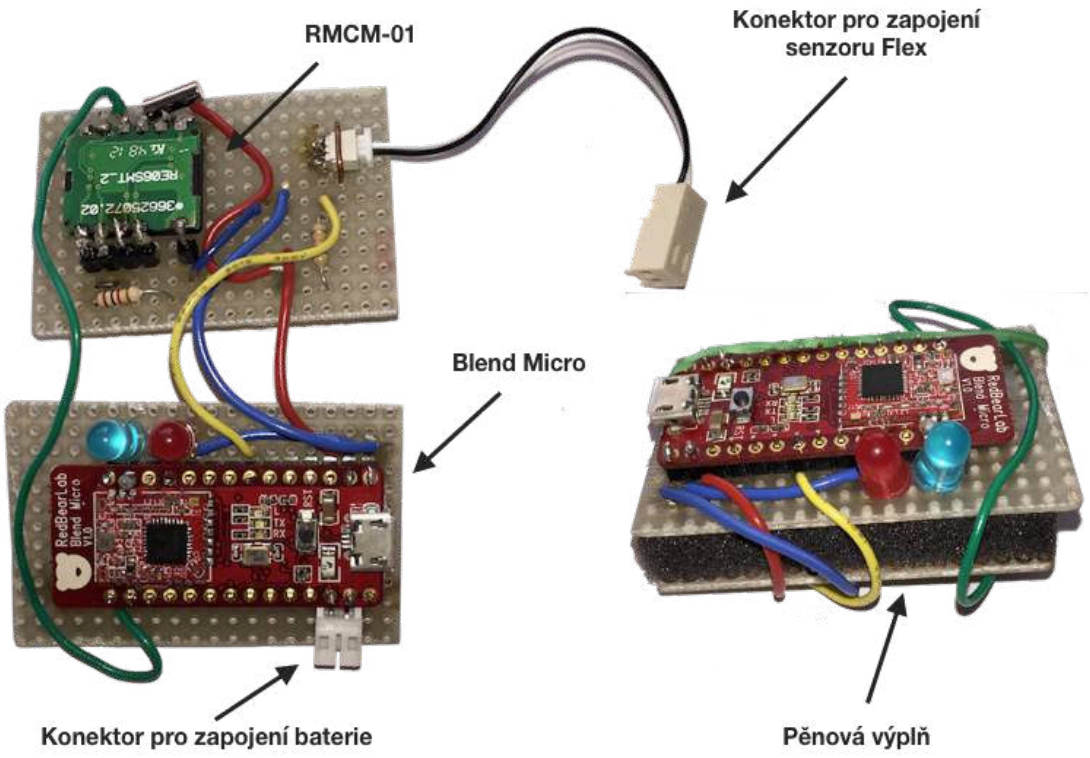
Polymer Lithium Ion Battery (PRT-00731)	
Kapacita	110 mAh
Jmenovité napětí	3,7 V
Vybíjecí proud	0,2 C_5A
Impedance	≤ 25 mΩ
Váha	2,65 g
Pracovní teplota	-20 až 60°C

Vlastní zapojení systému na dvojici pájecích desek je uvedeno na obrázku (obr. 23). Schéma bylo vytvořeno taktéž v programu "fritzing".



Obr. 23: Schéma vlastního zapojení mikroprocesorové části systému na dvojici pájecích desek

Dle návrhu byl systém sestaven na dvojici pájecích desek (obr. 23). Desky je možné přiložit na sebe, čímž dojde k celkovému zmenšení velikosti plochy mikroprocesorové části. Pro zamezení nežádoucího kontaktu vodivých částí systému jsou od sebe desky izolovány pěnovou výplní (obr. 24). Systém je díky bezdrátovému spojení hrudního pásu a čidla RMCM-01 schopen detekovat elektrickou srdeční aktivitu a vypočíst hodnotu TF bez nutnosti fixace mikroprocesorové části na hrudní pás. Pokud ovšem vyžadujeme současný záznam TF a DF, je nutné zapojit senzor Flex a provést fixaci mikroprocesorové části na hrudní pás. K tomuto účelu byla vytvořena látková kapsa (obr. 25). Tu je možné jednoduše připevnit na hrudní pás v oblasti zad. Do kapsy je následně umístěna mikroprocesorová část systému. Použitím integrovaného konektoru lze jednoduše zapojit senzor Flex a detekovat respirační pohyby hrudníku. Seznam použitých součástek a modulů je uveden v příloze E.



Obr. 24: Realizace mikroprocesorové části systému na dvojici pájecích desek

Obr. 25: Kompletní podoba realizace systému

HW realizace systému představuje unikátní spojení dostupných "open-source" technologií, které by ještě před několika lety nebylo možné. S rozvojem mobilních technologií a miniaturizací elektrických komponent je nyní možné poměrně jednoduchým způsobem konstruovat zařízení, která spadají do jedné z nejrychleji rostoucích technologických odvětví, do oblasti tzv. *"Internetu věcí"* (z anglického "Internet of Things", zkratka IoT). Přesná definice tohoto slovního spojení dosud nebyla publikována avšak jedná se většinou o vestavěná zařízení schopná bezdrátové komunikace s jinými zařízeními, včetně možnosti přímé či zprostředkované komunikace s Internetem. Navržený systém pro monitorování sportovní aktivity SportsBrain této definici jednoznačně vyhovuje.

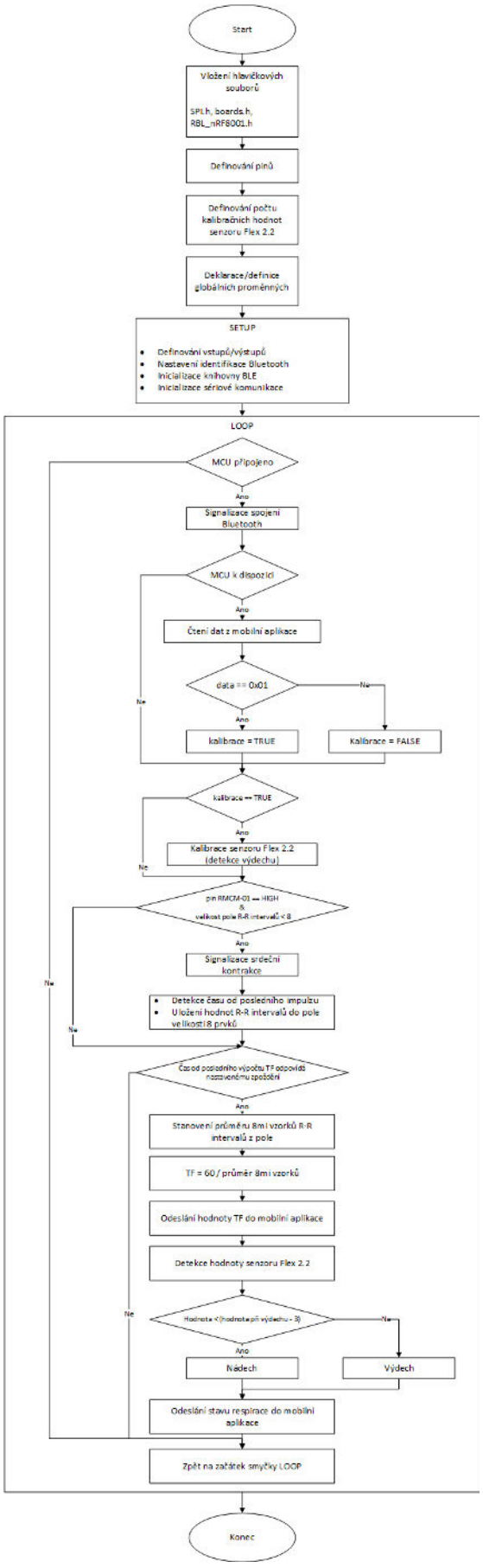
Prezentovaná HW konstrukce však není jen jednoúčelovým řešením. Vytváří platformu, na které je možné v budoucnu dále pracovat a zařadit další senzory, které systém rozšíří o monitorování jiných veličin. Tento fakt je dán především díky použití MCU otevřené vývojové platformy Arduino. V budoucnu tak lze do systému zařadit například senzor detekující počet kroků, pocení, teploty. Důležitou součástí mikroprocesoru je však také SW. SW část prezentovaného řešení je uvedena v následující podkapitole č. 4.2.2.

4.2.2 SW řešení MCU

Zdrojový kód byl vytvořen a nahrán do MCU pomocí vývojového softwaru Arduino IDE. Základními činnostmi programu jsou:

- Detekce elektrických impulzů RMCM-01 odpovídající srdeční činnosti
- Stanovení R-R intervalů a následný výpočet aktuální hodnoty TF
- Kalibrace senzoru Flex po aktivaci systému pomocí mobilní aplikace
- Detekce respiračních pohybů
- Přenos údaje o aktuální hodnotě TF do mobilního telefonu pomocí Bluetooth
- Přenos údaje o respiraci do mobilního telefonu pomocí Bluetooth

Vývojový diagram programu MCU je uveden na následující straně (obr. 26). Následně jsou popsány jednotlivé funkční bloky programu včetně uvedení zdrojového kódu, který částečně vychází z volně dostupného zdrojového kódu uvedeného v [24].



Obr. 26: Vývojový diagram programu MCU

Inicializace programu

```
#include <SPI.h>
#include <boards.h>
#include <RBL_nRF8001.h>

#define TONE 9           // Reproduktor
#define HR_PIN 8         // Vystup z RMCM-01
#define RR_PIN A2        // Vystup z Flex 2.2
#define LIGHT_PIN 10     // Detekce spojeni Bluetooth
#define HR_LIGHT 12      // Dioda TF

#define KRAT 100         // Pocet kalibracnich hodnot Flex 2.2

// Globalni promenne
long tep = 0;                   // Hodnota tepove frekvence
long vzorky[8]= {0};            // Pole R-R intervalů
long casPosledniVzorek = 0;     // Hodnota casu prichodu vzorku
byte posledniVzorek = 0;        // Index posledniho vzorku R-R
byte dalsiVzorek;               // Index nasledujiciho vzorku R-R
long casPosledniReport = 0;
int zpozdeni = 1000;            // Zpozdeni odeslani TF 1000 ms

int kal[KRAT];                  // Pole Flex 2.2 - surova data
int kal2[KRAT];                 // Pole Flex 2.2 -  po funkci map
int dech = 0;                   // globalni prom. dechu
boolean kalibrace = false;      // Inicializace kalibrace
int sensorMax = 0;              // Max. hodnota senzoru po kalibraci
int RRChange = 0;               // Ukladani dat o nadechu/vydechu
```

Předpokladem pro možnost propojení vývojové desky Blend Micro s mobilním telefonem pomocí Bluetooth 4.0 LE je vložení hlavičkových souborů SPI.h, boards.h a RBL_nRF8001.h. Jedná se o soubory dodávané výrobcem vývojové desky, které obsahují potřebné knihovny funkcí pro navázání Bluetooth komunikace. Soubory jsou volně přístupné na internetovém portálu výrobce (viz [6]).

V uvedené části zdrojového kódu jsou definována jména a čísla použitých pinů vývojové desky. Jednoznačná identifikace pinů pomocí symbolické konstanty (makra bez parametru) usnadňuje další integraci v programu. Pin je potom charakterizován pouze svým názvem a nikoli číselným údajem. Změna v definici se pak také projeví ve všech nastaveních daného pinu v programu. Následují deklarace či definice proměnných a polí, jejichž charakter vyžaduje globální definici či deklaraci. Účel jednotlivých polí a proměnných je uveden přímo v prezentované části zdrojového kódu.

Funkce “setup“

```
void setup() {
  // Definování I/O pinů
  pinMode(HR_PIN,INPUT);              // Vystup z RMCM-01
  pinMode(LIGHT_PIN, OUTPUT);         // Detekce spojeni Bluetooth
  pinMode(HR_LIGHT, OUTPUT);          // Dioda TF
  ble_set_name("SportsBrain_monitor"); // Nastaveni ID Bluetooth
  ble_begin();                // Inicializace knihovny BLE
  Serial.begin(57600);        // Inicializace seriove komunikace
}
```

Funkce “setup“ je volána automaticky ihned po startu programu. Jejím úkolem je další inicializace proměnných, charakteru pinů či použitých knihoven funkcí. V případě uvedeného zdrojového kódu je především nastaven vstupně/výstupní charakter použitých pinů. Součástka RMCM-01 generuje elektrický impulz o velikosti 3V vždy, kdy je hrudním pásem detekována srdeční kontrakce. Je tedy nutné, aby byl HR_PIN nastaven jako vstup. Další použité piny jsou nastaveny jako výstup. Jedná se o piny pro komunikaci s LED diodami.

Příkazem `ble_set_name("SportsBrain_monitor")` je nastavena identifikace pro Bluetooth komunikaci. Vývojová deska Blend Micro se následně pro zařízení v dosahu hlásí pod jménem “SportsBrain_monitor“. Příkaz `ble_begin()` složí pro inicializaci Bluetooth komunikace. Poslední příkaz `Serial.begin(57600)` nastavuje rychlost přenosu dat sériové komunikace. Zde je rychlost nastavena na hodnotu 57600 bps (Bits per second).

Funkce “loop“

Prostřednictvím funkce “setup“ je inicializováno Bluetooth spojení a nastaven směr komunikace pinů MCU. Funkce “loop“ je následně volána a představuje nejdůležitější část programu MCU. Jedná se o nekonečnou smyčku s cílem kalibrace Flex senzoru, který je následně použit pro detekci respiračních pohybů. Dále pak k detekci impulzů RMCM-01 a následný výpočet hodnoty tepové frekvence. Vypočtená hodnota TF, spolu s informací o stavu respirace, je nakonec pomocí Bluetooth přenášena do mobilního telefonu. Zde jsou popsány jednotlivé části zdrojového kódu funkce “loop“.

```
void loop (){

  if (ble_connected()) {         // Dioda signalizace Bluetooth
    digitalWrite(LIGHT_PIN, HIGH);
  } else {
      digitalWrite(LIGHT_PIN, LOW);
    }

while(ble_available())
    {
      // Cteni dat z telefonu
      byte data0 = ble_read();
      byte data1 = ble_read();
      byte data2 = ble_read();

     if (data0 == 0x0A)
      {
        if (data1 == 0x01) {     // Kalibrace
          kalibrace = true;
        }
        else                     // Ke kalibraci nedochazi
         {
           kalibrace = false;
         }
      }
    }
```

První podmínkou je nastavena signalizace spojení Bluetooth. Je-li uživatel prostřednictvím mobilní aplikace SportsBrain připojen k MCU, je toto spojení signalizováno aktivací modré diody. Přestane-li mobilní aplikace využívat senzorových dat MCU, dioda je deaktivována. Funkce `ble_available()` vrací hodnotu TRUE v případě, že je Bluetooth spojení k MCU k dispozici. V takovém případě probíhá příjem dat z mobilní aplikace.

Komunikace směrem z mobilní aplikace k MCU je zde z důvodu nastavení hodnoty "`kalibrace`". Mobilní telefon odesílá hodnotu 0x01 ihned po prvním navázání Bluetooth komunikace s MCU. V tom okamžiku předpokládáme, že má uživatel již nasazen hrudní pás a je tak možné provést kalibraci senzoru Flex. Zde je však nastavena jen hodnota "`kalibrace`". Kalibrace samotná je provedena až následně. Pokud již kalibrace proběhla a senzor Flex již snímá respirační pohyby uživatele, mobilní telefon odesílá hodnotu různou od 0x01. Hodnota "`kalibrace`" je nastavena na `"false"` a k nové kalibraci v průběhu záznamu již nedochází.

Kalibrace senzoru respirace

Při prvním navázání Bluetooth spojení mezi MCU a mobilní aplikací odesílá mobilní telefon směrem k MCU hodnotu 1 v hexadecimálním formátu. Následně je hodnota "`kalibrace`" nastavena na `"true"`. Tím je splněna podmínka pro kalibraci, která je následně provedena. Následující část zdrojového kódu programu MCU ukazuje způsob samotné kalibrace senzoru Flex pro detekci respiračních pohybů v průběhu sportovní činnosti.

```
// ------------------ Kalibrace senzoru respirace ---------------
int n;                                          // Promenna cyklu

  if (kalibrace == true){
    for (n=0; n<KRAT; n++){
      kal[n] = analogRead(A2);                  // Cteni dat
      kal2[n] = map(kal[n], 700, 750, 0, 100);  // Map 0 - 100
      Serial.println(kal2[n]);
    }
  kalibrace = false;

    for(int x = 0; x < KRAT; x++){              // Nalezeni max
      for(int y = 0; y < KRAT-1; y++)
        if(kal2[y] > kal2[y+1]) {
          int holder = kal2[y+1];
          kal2[y+1] = kal2[y];
          kal2[y] = holder;
        }
    }
  sensorMax = kal2[KRAT - 1];                   // Max senzoru
  Serial.println("Maximalni hodnota senzoru je:");
  Serial.println(sensorMax);
  }
```

Při splnění podmínky provedení kalibrace je inicializován cyklus, který uloží do pole `"kal[]"` hodnoty předávané do MCU senzorem Flex. Počet takto uložených hodnot byl nastaven v samotném úvodu programu proměnnou `"KRAT"` na hodnotu 100. Tyto

uložené bezrozměrné hodnoty se po nasazení hrudního pásu pohybují v rozmezí 700 až 750. Pro pohodlnější práci se senzorovými daty je na tomto místě vhodné převést hodnoty v uvedeném rozmezí na měřítko v rozmezí 0 až 100. K tomuto účelu slouží funkce `"map()"`. Tato funkce převádí hodnotu ze zadaného rozmezí na hodnotu nově definovaného rozmezí. Výsledkem prvního cyklu je tedy 100 hodnot senzoru v rozmezí 0 až 100 uložených do pole `"kal2[]"`.

Pro možnost detekovat respirační pohyby je nutné zvolit referenční hodnotu senzoru. Na základě této hodnoty bude následně odvozeno, zda se uživatel nachází ve fázi nádechu či výdechu. Vhodnou referenční hodnotou je zde maximální hodnota senzoru obsažená v poli `"kal2[]"`, která odpovídá jeho maximálnímu prohnutí. Ohybový senzor je v elastické části hrudního pásu integrován takovým způsobem, aby se ve fázi výdechu (stažení elastické části pásu) zvýšil jeho ohyb. Ve fázi nádechu (natažení elastické části pásu) naopak dochází k ohybu mírnějšímu. Lze tedy předpokládat, že maximální hodnota senzoru odpovídá výdechu. Následující dvojicí cyklů je proto provedeno přerozdělení hodnot uložených v poli `"kal2[]"` na základě jejich velikosti od nejmenší hodnoty po hodnotu maximální. Maximální hodnota senzoru Flex pak odpovídá poslední hodnotě v poli. Na základě provedení testování senzoru je však za maximum brána hodnota předposlední. S touto hodnotou dosahuje detekce respiračních pohybů lepších výsledků. Výsledná kalibrační hodna senzoru je tisknuta do terminálu pomocí sériové komunikace (v případě propojení vývojové desky s počítačem).

Stanovení tepové frekvence

Zde je předložena část zdrojového kódu pro samotný výpočet tepové frekvence. Pro lepší názornost je popis kódu rozdělen do více částí.

```
long cas=millis();    // Funkce vracejici aktualni hodnotu casu

if (digitalRead(HR_PIN)==HIGH){
  digitalWrite(HR_LIGHT, HIGH);            // Indikace tepu

  if (cas - casPosledniVzorek > 10 ){
    dalsiVzorek = ((posledniVzorek + 1) % 8);
    vzorky[dalsiVzorek] = cas - casPosledniVzorek;
    posledniVzorek = dalsiVzorek;
  }
        casPosledniVzorek = cas;
}
```

Základním předpokladem stanovení TF na základě vzorce (1) uvedeného v kap. 2.2 je znalost časového intervalu mezi jednotlivými srdečními kontrakcemi (R-R intervaly). Tuto hodnotu lze získat měřením času mezi jednotlivými el. impulzy, které generuje součástka RMCM-01 při každé detekované srdeční kontrakci. Aktuální hodnota času je nejdříve uložena pomocí funkce `"millis()"` do proměnné `"cas"`. Uvedená funkce na výstupu poskytuje informaci o času, který uplynul od spuštění programu v milisekundách. Následnou podmínkou je detekováno, zda je na pinu připojeném k RMCM-01 zaznamenán elektrický impulz. V kladném případě je indikována srdeční

kontrakce pomocí diody. Druhá podmínka je zde použita pro zabezpečení detekce jen těch elektrických impulzů, které skutečně odpovídají nové srdeční kontrakci. Odečtením času aktuálního vzorku (impulzu RMCM-01) a času vzorku předchozího je stanovena aktuální hodnota posledního R-R intervalu. O nový R-R interval se jedná pouze tehdy, je-li časová prodleva mezi vzorky dostatečná.

Z důvodu vysoké frekvence detekovaných impulzů v průběhu sportovní zátěže je vhodné počítat aktuální hodnotu TF jako aritmetický průměr mezi osmi po sobě jdoucími impulzy. Pole `"vzorky[]"`proto bylo inicializováno pro uložení právě osmi hodnot. Nyní dochází k plnění tohoto pole hodnotami R-R intervalů. Index daného vzorku je stanoven dělením "modulo", které zabezpečuje udržení indexu vzorku a velikosti pole (bufferu) v mezích 1 – 8.

```
digitalWrite(HR_LIGHT, LOW);      // Indikace tepu

if (cas - casPosledniReport > zpozdeni){
  if (cas - casPosledniVzorek < zpozdeni){
    long sumaVzorku = 0;   // Promenna pro sumu vsech vzorku
    int i;                 // Promenna cyklu
    for (i=1; i < 9; ++i) {
      int pom = ((posledniVzorek + i) % 8);
      sumaVzorku  = sumaVzorku + vzorky[pom];
    }
```

Zde dochází nejdříve k deaktivaci signalizace srdeční kontrakce. Signalizace pomocí LED je opět aktivována až s detekcí nového impulzu z RMCM-01. Dvojicí podmínek je nyní zajištěn výpočet a odeslání hodnoty TF každou sekundu (nastavená hodnota zpoždění systému). Následný cyklus přiřadí každému z osmi vzorků (dříve akumulovaných v bufferu) index a vypočte jejich sumu.

```
sumaVzorku /= 8;
tep = (6000000/(sumaVzorku))/100;
tep = (int) tep;
Serial.println("HR Value:");
Serial.println(tep);
// Odeslani hodnoty TF do aplikace SportsBrain
  uint16_t value = tep;
  ble_write(0x0B);
  ble_write(value >> 8);
  ble_write(value);
```

Suma vzorků je nyní podělena velikostí pole (bufferu). Tímto je získán aritmetický průměr délky R-R intervalu z osmi posledních R-R intervalů. Vypočtený čas je aplikován podle vzorce (1) a tak získána aktuální hodnota tepové frekvence, která je převedena na celočíselný formát. Výsledek je tisknut do terminálu (v případě propojení vývojové desky s počítačem). Na tomto místě však také dochází k odeslání vypočtené hodnoty TF do mobilní aplikace pomocí funkcí obsažených v knihovně `"RBL_nRF8001.h"`. Hodnota TF je přenášena jako typ `uint16_t`. Aby byla mobilní aplikace schopna rozpoznat, že přijímaná data představují právě hodnotu TF, jsou opatřena identifikátorem v hexadecimálním tvaru. Tento identifikátor je pro TF hodnota 11.

Detekce respiračních pohybů

V následující části zdrojového kódu je popsán způsob detekce respiračních pohybů, nyní již kalibrovaným senzorem Flex. K detekci dochází v průběhu podmínky pro stanovení TF. Dle nastaveného zpoždění systému tedy každou sekundu. Tato vzorkovací frekvence je dostačující. Maximální hodnota DF dosahovaná v rámci sportovní aktivity (viz tab. 3) by neměla překročit hodnotu 60 dechů za minutu.

```
  int sensorValue, degrees;            // Hodnota senzoru
  sensorValue = analogRead(RR_PIN);
  degrees = map(sensorValue, 700, 750, 0, 100);

  if (degrees < (sensorMax-3)) {
    Serial.println("Nadech");
    RRChange = 1;
  }

  else {
  Serial.println("Vydech");
  RRChange = 0;
}

// Odeslani hodnoty respirace do aplikace SportsBrain
  uint16_t dech = RRChange;
  ble_write(0x0C);
  ble_write(dech >> 8);
  ble_write(dech);
}} casPosledniReport = cas;
}
```

Do proměnné `"sensorValue"` je nejdříve uložena aktuální bezrozměrná hodnota senzoru, detekovaná na analogovém pinu A2. Proměnná `"degrees"` je hodnota senzoru převedená na měřítko 0 – 100, podobně jako při kalibraci. Následující podmínka již detekuje samotné respirační pohyby. Na základě testování systému a charakteru použitého ohybového senzoru lze za významnou změnu ohybu, představující nádech, považovat změnu o tři hodnoty proměnné `"degrees"`. V takovém případě je do proměnné `"RRChange"` uložena hodnota 1 odpovídající nádechu. Jinak je do této proměnné uložena hodnota 0 odpovídající výdechu. Aktuální stav respirace je následně ihned odesílán prostřednictvím Bluetooth do mobilní aplikace stejným způsoben jako hodnota TF. Hodnota respirace uložená v proměnné `"RRChange"` je přenášena jako proměnná `"dech"` typu `uint16_t`. Aby byla mobilní aplikace schopna rozpoznat, že přijímaná data představují právě hodnotu respirace, jsou opatřena identifikátorem v hexadecimálním tvaru. Tento identifikátor je zde hodnota 12. Nakonec je uložena aktuální hodnota času, která je využita v podmínce pro nastavení zpoždění systému.

4.3 Mobilní aplikace “SportsBrain“

I když bylo v předchozím textu uvedeno, že cílem této práce není popis způsobu vývoje mobilních aplikací, je potřebné si nejdříve vymezit některé pojmy, které se dále v práci vyskytují a jejich znalost je důležitá pro pochopení předkládaného technického řešení. Následně bude proveden popis technického řešení mobilní aplikace SportsBrain, která představuje uživatelskou část navrženého systému pro monitorování sportovní aktivity.

4.3.1 Základní poznatky z vývoje aplikací pro systém iOS

Hlavním nástrojem pro zápis, překlad a testování programů systému *iOS* je vývojové prostředí *Xcode*. Program je možné spustit na počítačích *Apple* s operačním systémem *OS X*. Jeho součástí je nástroj pro vytváření uživatelského rozhraní – *InterfaceBuilder* a také nástroj pro analýzu a ladění aplikací – *Instruments*. Součástí je taktéž simulátor mobilního telefonu, který umožňuje spustit navržené aplikace přímo v počítači bez nutnosti převodu na telefon či tablet. Základním programovacím jazykem iOS je *Objective-C* (nově i *Swift*). Vývojové prostředí *Xcode* však umožňuje zápis zdrojového kódu i v dalších jazycích jako je *C* či *C++* [13].

Objektově orientované programování

Je opakem standardnímu procedurálnímu programování jaké známe například u jazyka C. Cílem objektově orientovaného programování je propojení dat s úkoly, jež data zpracovávají. Jako *objekt* je možné si představit například tlačítko na obrazovce. Každé takové tlačítko pak musí být identifikováno tak, aby k němu bylo možné přistupovat a programovat jeho funkce. Konkrétní tlačítko je v terminologii objektově orientovaného programování *instancí třídy* obecného tlačítka. Tato vlastnost se nazývá *dědičnost* a znamená, že objekty jsou organizovány stromovým způsobem. Každý objekt je instancí nějaké třídy a každá třída může dědit své vlastnosti od třídy nadřazené. Další základní vlastností objektového programování je tzv. *zapouzdření*. Vyjadřuje fakt, že třídu lze použít bez toho, aniž bychom museli znát principy, jakými uvnitř funguje a které třídy má na základě dědičnosti v sobě zahrnuty. Tuto vlastnost lze přirovnat k použití standardních funkcí například v jazyku C. Zde se také často nestaráme o to, kde je funkce definována a deklarována, ale v kódu ji jednoduše zavoláme a zajímá nás jen její návratová hodnota. Poslední vlastností objektově orientovaného programování, která zde bude uvedena, je tzv. *polymorfizmus*. Tento pojem vyjadřuje vlastnost, kdy se objekt chová dle toho, jaké třídy je instancí. Lze tak například použít stejnou funkci pro dva různé objekty a pro každý z objektů bude funkce reagovat rozdílným způsobem [13].

Třídy

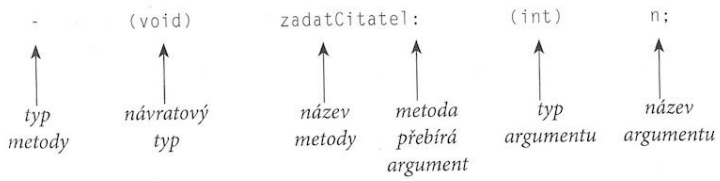
Třída je programovou reprezentací objektů se stejnými vlastnostmi. Udává předpis pro vytvoření objektu daného typu. Při definici nové třídy je nutné kompilátoru Objective-C sdělit, odkud třída pochází. Je tedy nutné určit její rodičovskou třídu. Následně zadat typ dat, který se bude do objektů dané třídy ukládat. Proměnné, které budou instance dané třídy obsahovat se nazývají *instanční proměnné*. Tyto kroky jsou prováděny ve speciálním oddílu programu označovaném jako *@interface*. Při deklaraci třídy jazyk vyžaduje dvě oddělené části kódu. V hlavičkovém souboru *.h* je nastaveno uvedené rozhraní třídy (lze však alternativně i v .m). Zdrojový kód potřebný pro samotnou funkcionalitu dané třídy se zadává v tzv. implementačním souboru s příponou *.m* a to v oddílu programu označeném jako *@implementation*. Oddíl *@interface* tedy popisuje třídu, její datové součásti a její metody. Oddíl *@implementation* pak obsahuje vlastní kód implementující příslušné metody [13].

Instance a metody

Jedinečný výskyt *třídy* je její *instancí*. Akce, které následně provádíme na této instanci, označujeme jako *metody*. Objekty představují jedinečné reprezentace třídy. Každý objekt obsahuje určitá data, která jsou obvykle soukromá. Metody používáme pro přístup k těmto datům a jejich změně. Jazyk Objective-C má syntaxi aplikování metod na třídy ve tvaru příjemce – zpráva takto:

```
[TřidaNeboInstance metoda];
```

Deklarace metod probíhá v oddílu *@interface* ve formátu zobrazeném na obrázku (obr. 27).



Obr. 27: Deklarování metody [13]

Následný oddíl *@implementation* obsahuje vlastní kód metody, tedy její definici. Definice metody je totožná z její deklarací. Avšak zatímco u deklarace je po zápisu kódu (obr. 27) zadán středník, u definice vložíme složené závorky, do kterých kód metody zadáváme.

SW architektura Model-Vzhled-Řízení

Aplikace psané v jazyku Objective-C jsou založené na konceptu Model-Vzhled-Řízení (z anglického "Model-View-Controller", zkráceně MVC). Koncept představuje logické rozdělení zdrojového kódu pro aplikaci založenou na grafickém uživatelském rozhraní (dále jen GUI). MVC rozděluje funkcionalitu programu do tří kategorií [15].

- *Mode*l: Je jádrem aplikace a obsahuje třídy.
- *Vzhled (View)*: Je tvořen vlastními ovládacími prvky jako jsou okna, tlačítka či gesta, prostřednictvím kterých uživatel s mobilní aplikací interaguje.
- *Řízení (Controller)*: Stojí na rozhraní modelu a vzhledu, které spojuje. Rozhoduje o tom, jakým způsobem aplikace reaguje na interakci od uživatele.

Outlet

Instanční proměnné definované pomocí klíčového slova `"IBOutlet"` označujeme jako *outlety*. Pro kompilátor nemá toto klíčové slovo žádný význam. Funguje však jako informace pro *Interface Builder*, kterému říká, že se jedná o instanční proměnnou, kterou chceme připojit k některému z objektů na obrazovce aplikace (např. obrázku).

Akce

Použitím zvláštního klíčového slova `"IBAction"` při deklaraci metody říkáme *Interface Builderu*, že deklarovaná metoda je akcí a lze ji spustit pomocí ovládacího prvku (např. tlačítka). Vstupem metody akce je obvykle parametr definovaný jako `id`. Tomuto parametru se obvykle přiřadí název `sender`. Ovládací prvek spouštějící danou akci použije tento parametr a předá odkaz sama na sebe. Je-li tedy volání metody vázané na stisk určitého tlačítka, parametr `sender` bude obsahovat odkaz na specifické tlačítko stisknuté uživatelem.

Delegát aplikace

Delegát obsahuje třídy, které provádějí operace pro jiné objekty. Umožňuje provádět určité operace v konkrétních předdefinovaných okamžicích. Objekt komunikuje se svým delegátem pomocí předem deklarovaných zpráv a informuje jej tak například o svých změnách. Příkladem je okamžik, kdy potřebujeme být informováni o stisku daného tlačítka či dostupnosti GPS dat, na který jsou navázány další akce v aplikaci [15].

Rámce

Rámce ("Frameworky"), poskytující podporu aplikacím Objective-C pro počítače s operačním systémem OS X, se nazývají *Cocoa*. V základu se jedná o spojení rámce *Foundation* a rozšiřujícího rámce *Application Kit*, který obsahuje třídy související s okny a tlačítky. Mobilní telefon iPhone je vlastně počítačem s omezenou verzí systému Mac OS X. Ačkoliv se základní principy programování pro OS X a iOS v zásadě neliší, má programování mobilních aplikací některá svá specifika. Rámce Cocoa jsou určeny k vývoji aplikací pro stolní počítače a notebooky s Mac OS X. K vývoji aplikací pro mobilní zařízení iPhone, iPad a iPod touch slouží rámce *Cocoa Touch*. Zde je Application Kit nahrazen rámcem *UIKit*, který nabízí podporu řady stejných typů objektů jako jsou okna, tlačítka, textová pole atd. Cocoa Touch však navíc nabízí třídy pro práci s akcelerometrem, GPS a především s rozhraním pro dotykové ovládání [13].

Při vytvoření nového projektu mobilní aplikace se programem Xcode automaticky vloží základní rámce *UIKit* a *Foundation* zabezpečující podporu všech základních typů objektů. Navržená aplikace využívá i jiné rámce, například pro možnost záznamu GPS dat. Vlastnosti jednotlivých rámců a jejich využití v navržené aplikaci SportsBrain jsou popsány na příslušném místě dále v textu.

4.3.2 Požadavky na návrh aplikace pro monitorování sportovní aktivity

Cílem mobilní aplikace vytvořené v rámci této diplomové práce je demonstrace současných technických možností propojení mobilního telefonu s open source MCU a jinými HW komponentami. Primárním cílem je vytvoření komplexní mobilní aplikace pro podporu uživatele v průběhu sportovního tréninku. Aplikace proto poskytuje funkčně obsáhlý celek spolu s intuitivním a přátelským uživatelským rozhraním. Je však především schopna jednoduchého propojení se senzorovou a MCU částí navrženého systému prostřednictvím bezdrátové technologie Bluetooth 4.0 LE. Dále nabízí uživateli kompletní přehled o právě probíhajícím sportovním tréninku včetně zobrazení aktuální hodnoty tepové a dechové frekvence, pásma zatížení či tempa. Po ukončení tréninku je aplikace schopna zobrazit podrobnou analýzu všech sledovaných parametrů s možností uložení dané aktivity a možností jejího sdílení prostřednictvím sociálních sítí. V neposlední řadě aplikace umožňuje vytvoření profilu uživatel včetně možnosti stanovit či vypočíst individuální pásma intenzity tréninku. Výše popsané funkce uvádíme v následujícím přehledu.

Dostupné funkce mobilní aplikace SportsBrain

- Monitorování běžeckého a cyklistického tréninku
- Zobrazení aktuální hodnoty TF, DF a pásma intenzity tréninku
- Zobrazení aktuální rychlosti, tempa a vzdálenosti pomocí GPS
- Analýza tréninku se zobrazením maximální a průměrné hodnoty TF a DF
- Zobrazení trasy tréninku v mapových podkladech včetně vykreslení rychlostního profilu tratě
- Osobní profil uživatele včetně automatického stanovení tréninkových pásem zatížení
- Fitness test umožňující stanovení individuální maximální TF a následný výpočet pásem zatížení, zvlášť pro běžecký a pro cyklistický trénink
- Možnost sdílení poslední aktivity pomocí SMS, e-mailu či sociálních sítí Facebook a Twitter
- Schopnost práce na pozadí

4.3.3 Ikona aplikace

Základem každé mobilní aplikace je propracovaná a především "zapamatovatelná" ikona, která jasně vyjadřuje účel aplikace. V operačním systému iOS se ikona vyskytuje na domovské obrazovce systému a slouží pro přístup do aplikace samotné. Zobrazuje se i ve výsledcích vyhledávání systému iOS, či v nastavení. Navržená ikona mobilní aplikace SportsBrain je na obrázku (obr. 28).

Obr. 28: Ikona mobilní aplikace SportsBrain

Mobilní aplikace SportsBrain je primárně vytvořena pro mobilním telefonu iPhone 4s. Pro tento MT s iOS ve verzi 8 je požadovaná velikost ikony stanovena na hodnotu 120x120 bodů pro domovskou obrazovku. Pro zobrazení ikony v nastavení musí mít velikost 58x58 bodů a pro zobrazení ve výsledcích vyhledávání 80x80 bodů.

4.3.4 Vývojový diagram

Program *Xcode* poskytuje unikátní nástroj pro návrh uživatelského rozhraní mobilních aplikací systému iOS. Tento nástroj nese označení *"Storyboard"* a je grafickým vyjádřením vztahu jednotlivých "obrazovek" aplikace. Má-li mobilní aplikace mnoho různých obrazovek, storyboard poskytuje kompletní přehled jejich vztahu. Nebude zde proto uveden kompletní vývojový diagram tak, jako u program MCU. Bylo by třeba vytvořit několik samostatných diagramů pro každou jednotlivou obrazovku aplikace, kterých je 11. Storyboard aplikace SportsBrain (viz příloha F) poskytuje dostatečný přehled o vztazích jednotlivých obrazovek, které budou v práci podrobně popsány. Bohužel doposud Xcode neumožňuje jednoduchý způsob exportu storyboardu ve formě obrázku či PDF. Obrázek pro přílohu byl proto získán jako "printscreen" obrazovky na monitoru s vysokým rozlišením 2800x1800 bodů.

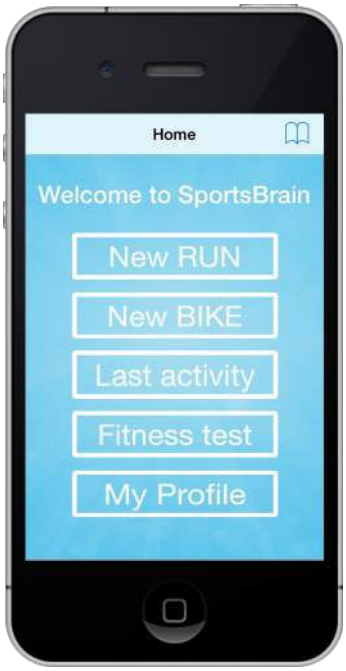
Každá obrazovka, prezentovaná v příloze F, představuje jeden samostatný celek mobilní aplikace, který může za určitých okolnosti pracovat samostatně. V navržené aplikaci SportsBrain spolu ovšem obrazovky úzce komunikují a předávají si data související se sportovním tréninkem. Každá obrazovka musí být reprezentována svým vlastním *hlavičkovým* (.h) a *implementačním* (.m) souborem, který zabezpečuje důležité

spojení dat aplikace s jejím uživatelským rozhraním. Tyto soubory nazýváme "*View Controllery*". Kdykoliv iOS zobrazí uživatelské rozhraní, zobrazený obsah je řízen pomocí *view controlleru* či skupinou spolupracujících *view controllerů*. *View controllery* tak vytváří hlavní programovou kostru, na které stavíme samotnou mobilní aplikaci. Vývojové prostředí *Xcode* poskytuje velké množství vestavěných šablon pro návrh specifické mobilní aplikace. Aplikace SportsBrain využívá šablony "*Master Detail Application*". Tato šablona je velmi praktická a poskytuje jednoduchou navigaci v aplikaci s více obrazovkami tím, že po vložení a propojení nové obrazovky automaticky vytvoří navigační lištu ("*Navigation bar*") s názvem dané obrazovky a tlačítkem pro návrat na obrazovku předchozí.

Mobilní aplikace SportsBrain obsahuje 11 obrazovek. Ne každá obrazovka však představuje samostatnou funkci. Proto je aplikace funkčně rozdělena do tzv. *modulů*. Přístup k jednotlivým modulům je pomocí domovské obrazovky (hlavní nabídky), která je uživateli zobrazena ihned po spuštění aplikace (obr. 29).

4.3.5 Hlavní nabídka

Hlavní nabídka (domovská obrazovka), představuje první kontakt uživatele s mobilní aplikací. Cílem je poskytnout uživateli přístup ke všem dostupným modulům aplikace. Na tuto obrazovku se lze v rámci aplikace z kteréhokoliv modulu také vrátit pro následný vstup do modulu jiného. Hlavní nabídka obsahuje 5 tlačítek pro přístup k jednotlivým modulům (obr. 29).



Obr. 29: Hlavní nabídka mobilní aplikace SportsBrain

Modul *"nový běh"* ("new run") obsahuje funkce aplikace spojené se sledováním, záznamem a vyhodnocením běžeckého tréninku. Modul *"nové kolo"* ("new bike") je totožný s modulem předchozím, s rozdílem zobrazení informace o aktuální rychlosti jízdy na místo informace o aktuálním tempu. Fakt, že aplikace obsahuje dva moduly pro dvojici sportovních aktivit, je velmi podstatný. Maximální dosažitelná tepová frekvence se u různých sportovních aktivit liší. Tedy i individuální tréninková pásma intenzit musejí být vypočtena zvlášť pro různé aktivity. Rozdělením aktivit běh a cyklistika do dvou modulů je aplikace schopna uživateli poskytnout informaci o pásmu intenzity tréninku, ve kterém se pro danou aktivitu na základě individuální TF_{max} nachází. Po ukončení a uložení tréninku má uživatel možnost zobrazení dosažených výsledků zpětně pomocí modulu *"poslední aktivita"* ("last activity"). Odtud je také možné výsledky sdílet prostřednictvím SMS, e-mailu či sociálních sítí. Modul *"fitness test"* nabízí uživateli možnost provedení zátěžového testu pro stanovení individuální TF_{max} v rámci běžeckého a cyklistického tréninku. Na základě těchto naměřených hodnot pak mobilní aplikace může provést výpočet pásem intenzity tréninku pro danou sportovní aktivitu. Poslední modul *"můj profil"* ("my profile") umožňuje uživateli zadat údaje o své výšce, váze či věku, zobrazit svá vypočtená pásma intenzit tréninku, či je nechat aplikací vypočíst podle výrazu (2). Podrobnější popis jednotlivých modulů včetně rozboru zdrojového kódu je uveden dále.

Jedinou možností návratu uživatele z domovské obrazovky aplikace zpět do hlavní nabídky operačního systému iOS je stisk fyzického tlačítka na mobilním telefonu – *"Home Button"*. I po stisku tohoto tlačítka však aplikace může pokračovat v záznamu sportovní aktivity.

Domovská obrazovka aplikace je propojena s "ovládacími" soubory *HomeViewController.h* a *HomeViewController.m*. Vzhledem k tomu, že i pro všechny následující obrazovky aplikace jsou soubory XXViewController.h a .m základem propojení uživatelského rozhraní s funkcionalitou aplikace a vzhledem k tomu, že domovská obrazovka slouží pouze jako "rozcestník", bude zde uveden kompletní zdrojový kód hlavičkového a implementačního souboru. Tyto dva soubory umožní nahlédnout na základní strukturu, která se následně opakuje u všech daších obrazovek. U těch již bude popis zaměřen jen na "zajímavé" části zdrojového kódu. Popisy zdrojového kódu jsou založeny na znalostech autora této diplomové práce a také na teoretických poznatcích čerpaných především z odborné literatury [15].

HomeViewController.h

Hlavičkový soubor obvykle obsahuje společné definice, makra a deklarace proměnných. Kompletní zdrojový kód souboru HomeViewController.h je uveden na následující straně č. 60.

```
#import <UIKit/UIKit.h>

@interface HomeViewController : UIViewController

// Přístup CoreData
@property (strong, nonatomic) NSManagedObjectContext
*managedObjectContext;

// Info tlačítko
- (IBAction)infoBT:(id)sender;


@end
```

Prvním příkazem je importován rámec `UIKit`. Mezi standardní "desktopovou" a dotykovou platformou jsou takové rozdíly, že je standardní rámec *AppKit* nahrazen rámcem *UIKit*. Tento má v *Cocoa Touch* stejnou roli jako *AppKit* v *Cocoa*. Obsahuje základní třídy rámce *Foundation*, jako například NSString či NSArray.

Následuje oddíl, kde je popsána třída, její datové součásti a metody. Za klíčovým slovem `@interface` je uveden název třídy a název rodičovské třídy. Zde nejsou deklaruovány členy třídy, proto se obejdeme bez složených závorek. Následuje řádek začínající klíčovým slovem `@property`. Do češtiny toto klíčové slovo překládáme jako *atribut*. Před začleněním atributů do Objective-C museli programátoři definovat páry metod, které nastavovaly a navracely hodnoty pro každou instanční proměnnou ve třídě. Takové metody nazýváme *přístupové metody* (návratová – getter, nastavovací - setter). Deklarace `@property` sdělí kompilátoru, aby návratové a nastavovací metody vytvořil automaticky při kompilaci. V kulatých závorkách za klíčovým slovem je dále nastaveno, jakým způsobem má kompilátor návratové a nastavovací metody vytvořit. Více se lze dočíst v [15]. Následuje typ a název atributu. V našem případě je na tomto místě vytvořena instance třídy `NSManagedObjectContext`, ta zabezpečuje přístup do databáze *CoreData*, kterou mobilní aplikace SportsBrain využívá.

Po deklaraci atributů je ve zdrojovém kódu řádek deklarace metody pro akci. Metoda je uvozena klíčovým slovem `IBAction` a informuje ostatní třídy a Interface Builder o tom, že námi deklarovaná třída má metodu akce s názvem `infoBT`. Celá část interface je následně zakončena klíčovým příkazem `@end`.

HomeViewController.m

V implementačním souboru je implementováno vše, co bylo deklarováno v souboru hlavičkovém.

```
#import "HomeViewController.h"

#import "NewRunViewController.h"
#import "NewBikeViewController.h"

@interface HomeViewController ()

@end
```

```
@implementation HomeViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view.
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}
```

Nejdříve je do tohoto souboru zahrnout nadřazený soubor `HomeViewController.h`. Následuje zahrnutí hlavičkových souborů `NewRunViewController.h` a `NewBikeViewController.h`, které složí pro operace s moduly "New run" a "New bike". Veškeré definice tříd byly v oddílu `@interface` již provedeny v rámci nadřazeného hlavičkového souboru, proto je zde tento oddíl prázdný. Jak bylo zmíněno, obsahuje oddíl `@implementation` vlastní kód metod (definice) deklarovaných v oddílu `@interface`.

Metoda `viewDidLoad` je podtřídou rodičovské třídy `UIViewController`, tedy našeho řídícího objektu. Ne vše, co je nutné v rámci dané obrazovky provést, lze nastavit pomocí *Interface Builderu*. Tato metoda tedy slouží pro nastavení veškerých počátečních akcí ihned po načtení dané obrazovky. Často se v této metodě načítají data, se kterými se dále v rámci obrazovky pracuje, či se nastavují počáteční parametry obrázků a tlačítek. Následující metoda `didReceiveMemoryWarning` třídy `UIViewController` slouží pro uvolnění rámce z paměti v případě, že se daný rámec zrovna nezobrazuje.

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue
sender:(id)sender
{
    UIViewController *nextController = [segue
destinationViewController];
    if ([nextController isKindOfClass:[NewRunViewController
class]]) {
        ((NewRunViewController *)
nextController).managedObjectContext = self.managedObjectContext;
    }
    else if ([nextController isKindOfClass:[NewBikeViewController
class]]) {
        ((NewBikeViewController *)
nextController).managedObjectContext = self.managedObjectContext;
    }
}
```

Metoda `prepareForSeque` slouží pro přenos informací mezi dvojicí *UIViewControllerů*. Pro možnost přístupu do databáze, kterou mobilní aplikace využívá, je i do modulů "New run" a "New bike" přenesena třída `NSManagedObjectContext`.

```
- (IBAction)infoBT:(id)sender {
    UIAlertView *message = [[UIAlertView alloc] initWithTitle:@""

message:@"Application is a part of Master's thesis\n\nMobile system
for monitoring sports activity\n\n\n\n Author: Filip Malenak\n
Supervisor: doc. Ing. Jiří Rozman, CSc.\n\n\n Brno University of
Technology\n2015"
delegate:nil
cancelButtonTitle:@"OK"
otherButtonTitles:nil];
[message show];
}
@end
```

V rámci hlavičkového souboru byla deklarována metoda `infoBT`. Na tomto místě je daná metoda definována. Metoda slouží jako zdrojový kód pro nastavení akcí, které se stanou, stiskne-li uživatel na úvodní obrazovce v pravém horním rohu tlačítko se symbolem knihy (obr. 29). Proměnná `UIAlertView` není nic jiného, než varovné hlášení, které se po stisknutí tlačítka uživateli zobrazí. Hlášení obsahuje text uvádějící autora a vedoucího diplomové práce, v rámci které byla mobilní aplikace SportsBrain vytvořena. Uživatel toto hlášení akceptuje stisknutím tlačítka *"OK"* a vrátí se zpět do hlavní nabídky. Celá část implementace je následně zakončena klíčovým příkazem `@end`.

Na tomto místě bude ukončen popis kompletního zdrojového kódu a dále se popis zaměří na jednotlivé moduly s tím, že budou rozvedeny pouze ty části zdrojového kódu, které mají zásadní význam pro jednotlivé funkce aplikace. Kompletní zdrojový kód je umístěn na přiloženém DVD.

4.3.6 Modul "New Run"

Modul *"Nový běh"* ("New run") slouží pro záznam a vyhodnocení běžeckého tréninku. Společně s modulem pro záznam cyklistiky obsahuje ty nejpodstatnější funkce mobilní aplikace SportsBrain. Právě zde dochází k propojení mobilního telefonu s mikroprocesorem platformy Arduino a následnému přenosu dat s informacemi o aktuální hodnotě tepové frekvence a aktuálním stavu respirace. Jak již bylo uvedeno v kapitole věnované mikroprocesorové části systému (viz kap. 4.2), přenos dat probíhá prostřednictvím bezdrátové technologie Bluetooth 4.0 LE. Hodnotu TF, která je vypočtena v rámci MCU, lze přímo zobrazit na mobilním telefonu. Výpočet hodnoty dechové frekvence v rámci MCU proveden není. MCU pouze každou sekundu detekuje aktuální stav respirace a ten ukládá pro následný přenos ve formě 1 – nádech, 0 – výdech. Výpočet DF je proveden až v rámci samotné mobilní aplikace.

Podle individuální TF_{max} je aplikace v modulu osobního profilu uživatele schopna stanovit pásma intenzity tréninku. Tato informace je následně využita modulem "New run", kde je uživateli na základě sledované hodnoty TF zobrazeno i aktuální tréninkové pásmo. Uživatel tak může sledováním tohoto údaje zvolnit nebo naopak zvýšit svou fyzickou zátěž pro rozvoj specifické složky tělesné zdatnosti.

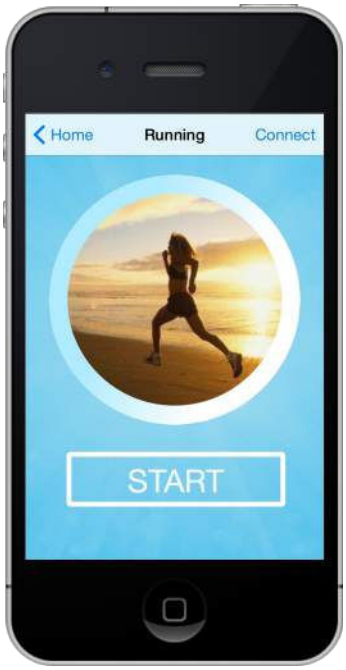
Součástí záznamu běžeckého tréninku je i sledování a ukládání dat, která mobilní telefon čerpá z vestavěného GPS senzoru. Polohová data jsou ukládána nejen pro následnou analýzu rychlostního profilu tratě v rámci vyhodnocení aktivity, ale slouží i pro zobrazení aktuální absolvované vzdálenosti a tempa.

Po ukončení běžeckého tréninku je uživateli nabídnuta možnost jeho vyhodnocení či smazání. Po výběru první možnosti je uživatel přenesen na obrazovku vyhodnocení aktivity. Zde jsou zobrazeny veškeré údaje o právě dokončené aktivitě a to včetně informace o maximální a průměrné tepové a dechové frekvenci. Zde má uživatel také možnost aktivitu uložit.

Toto jsou tedy základní vlastnosti modulu "New run". Dále je uveden podrobný popis jednotlivých uvedených funkcí.

Úvodní obrazovka aktivity

Uživatel je přenesen na úvodní obrazovku modulu "New run" po stisknutí příslušné volby v hlavní nabídce. Úvodní obrazovka obsahuje grafickou indikaci zvoleného modulu společně s tlačítkem "Connect", které je umístěno v rámci navigační lišty a které slouží pro propojení mobilní aplikace s mikroprocesorem platformy Arduino. Dále obrazovka obsahuje tlačítko "Start", po jehož stisknutí dochází k zahájení záznamu tréninku. Úvodní obrazovka modulu "New run" je znázorněna na obrázku (obr. 30).



Obr. 30: Úvodní obrazovka modulu "New run"

Spojení mobilní aplikace s MCU

Uživatel může z úvodní obrazovky modulu zahájit trénink bez nutnosti propojení na mikroprocesorovou část systému. V takovém případě však není možné snímání biologických parametrů TF a DF. Aplikace tak získává pouze data prostřednictvím vestavěného senzoru GPS. Zde je popsán postup, který by měl uživatel při použití systému dodržet pro plnohodnotný záznam sportovní aktivity. Před samotným zahájením aktivity je pomocí tlačítka *"Connect"* nutné mobilní telefon propojit s mikroprocesorovou části systému. Je-li MCU k dispozici, spojení je navázáno pomocí technologie Bluetooth. V opačné případě je uživateli zobrazeno varovné hlášení oznamující problém ve spojení.

Nyní bude v popisu funkcí modulu "New run" pokračováno společně s popisem zajímavých částí zdrojového kódu, který je obsahem implementačního souboru `"NewRunViewController.m"`. Hlavičkový soubor, obsahující deklarace metod, uveden nebude. Je plně dostupný na přiloženém DVD.

```
- (void)viewDidLoad {
    [super viewDidLoad];

    ble = [[BLE alloc] init];
    [ble controlSetup];
    ble.delegate = self;
}
```

Metoda `viewDidLoad` slouží v rámci modulu "New run" pro inicializaci spojení Bluetooth, které je tak připraveno ještě před samotným stiskem tlačítka *"Connect"*.

```
- (void)viewWillAppear:(BOOL)animated
{
    [super viewWillAppear:animated];

    self.startButton.hidden = NO;
    self.stopButton.hidden = YES;
…
```

Metoda `viewWillAppear` je použita nejen v rámci této obrazovky k nastavení viditelných a neviditelných prvků. Metoda je volána na rozdíl od metody `viewDidLoad` vždy, když je obrazovka běžeckého tréninku zobrazena a nikoliv pouze po jejím prvním načtení. Uvedená část zdrojového kódu ukazuje pouze část metody. Pro ilustraci jsou zde vypsány dva příkazy, které souží pro nastavení tlačítka zahájení a tlačítka ukončení aktivity. Tlačítko pro zahájení aktivity je na tomto místě nastaveno jako viditelné. Není však cílem, aby uživatel ještě před zahájením aktivity viděl tlačítko pro její ukončení. Proto je toto tlačítko skryto a zobrazuje se až v patřičný moment, kdy je ukončení aktivity aktuální.

Po stisknutí tlačítka *"Connect"* je volána metoda `btnScanForPeripherals`, která má za úkol skenovat dostupná zařízení v okolí a hledat odpovídající mikroprocesor platformy Arduino s modulem Bluetooth. Zdrojový kód této metody zde popsán nebude.

Jedná se o metodu, která je součástí balíku poskytovaného výrobcem vývojové desky Blend Micro. Zdrojový kód metody je dostupný na přiloženém DVD či na stránkách výrobce vývojové desky (viz [6]). Pro funkci aplikace je však nutné říci, že uvedená metoda využívá také metodu `connectionTimer`. Ta má za úkol sledovat čas, po který vyhledávání MCU probíhá. Nenastane-li spojení v nastaveném časovém intervalu, aplikace zobrazí varovné hlášení obsahující zprávu o nedostupnosti MCU.

Kalibrace senzoru respirace

Po úspěšném navázání spojení s mikroprocesorovou částí systému je volána metoda `bleDidConnect`, v rámci které probíhá důležitá kalibrace senzoru Flex na straně mobilní aplikace. Nejprve je však nutné inicializovat pole, do kterých následně ukládáme vypočtené hodnoty TF a DF. Zdrojový kód inicializace je uveden níže zejména proto, aby měl čtenář přehled o názvech jednotlivých polí, která dále aplikace využívá.

```
-(void) bleDidConnect {
  // Inicializace arraye HR hodnot
  HRhodnoty = [[NSMutableArray alloc] init];
  // Inicializace arraye RR hodnot
  RRhodnoty = [[NSMutableArray alloc] initWithCapacity:0];
  // Inicializace arraye časů pro detekci nádechu
  casoprostor = [[NSMutableArray alloc] initWithCapacity:0];
  // Inicializace arraye výsledných hodnot DF
  DFhodnoty = [[NSMutableArray alloc] initWithCapacity:0];
```

Programovací jazyk Objective-C nabízí dva druhy polí. *NSArray* a podtřídu *NSMutableArray*, která je využita i v uvedené části zdrojového kódu pro ukládání objektů do polí. Výhodou tohoto specifického pole je jeho teoreticky neomezená velikost, kterou není nutné při deklaraci zadat. Do velikosti pole lze tak průběžně zasahovat a přidávat (nebo naopak ubírat) objekty v poli uložené.

Zde je uveden zdrojový kód pro kalibraci senzoru Flex:

```
int velikostPoleHR= [HRhodnoty count];
// Odešli data na Arduino aby se zapla kalibrace senzoru
UInt8 buf[3] = {0x0A, 0x00, 0x00};

if (velikostPoleHR < 1) {
    buf[1] = 0x01;
    NSLog(@" Probiha kalibrace senzoru");
}
else                    // Jinak kalibrace probíhat nesmí
    buf[1] = 0x00;

    NSData *data = [[NSData alloc] initWithBytes:buf length:3];
    [ble write:data];
```

Do proměnné `velikostPoleHR` je uložen počet prvků pole `HRhodnoty`. Předpokladem je, že pokud je toto pole prázdné a neobsahuje žádné hodnoty přijatých tepových frekvencí, nachází se aplikace se ve stavu bezprostředně po stisknutí tlačítka *"Connect"*. V takovém případě jsou do mikroprocesoru odeslána data v hexadecimálním formátu o hodnotě 1. Tímto je MCU sděleno, aby provedl kalibraci senzoru respiračních

pohybů způsobem, který byl uveden v kapitole 4.2.2. Zde bylo uvedeno, že kalibrace senzoru je prováděna výdechem uživatele. Je tedy vhodné, aby se uživatel při stisknutí tlačítka "Connect" nacházel ve výdechu a v tomto setrval po dobu připojování MCU. Po provedené kalibraci již nic nebrání zobrazení aktuálních hodnot měřených biologických parametrů (obr. 31).

Obr. 31: Průběh připojování aplikace k MCU, zobrazení biologických veličin

Příjem a zobrazení dat MCU

Propojením mobilní aplikace s MCU je umožněn přenos dat. Jak bylo uvedeno v kapitole 4.2.2, odesílá MCU do mobilní aplikace vypočtenou aktuální hodnotu TF a to vždy jednou za sekundu prostřednictvím Bluetooth komunikace. Hodnota TF je označena identifikátorem v hexadecimálním tvaru. Tento identifikátor je pro TF hodnota 11. Současně MCU odesílá informaci o stavu respirace, opatřenou identifikátorem, taktéž v hexadecimálním tvaru. Tento identifikátor je pro respiraci hodnota 12. Data, která o stavu respirace aplikace přijímá jsou ve tvaru 1 – nádech, 0 – výdech. Na základě znalosti této informace lze vypočíst aktuální hodnotu respirace způsobem, který je popsán v další část zdrojového kódu. Zde je také popsáno, jakým způsobem dochází k výpočtu zátěžových pásem intenzity tréninku a následnému zobrazení daného pásma v průběhu aktivity. Pro větší přehlednost je popis zdrojového kódu rozdělen do více částí.

```
-(void) bleDidReceiveData:(unsigned char *)data length:(int)length{
for (int i = 0; i < length; i+=3){
// Příjem dat aktuálni hodnoty tepové frekvence
if (data[i] == 0x0B){
    UInt16 ValueHR;
    ValueHR = data[i+2] | data[i+1] << 8;
    // Zobrazeni TF na telefonu
    self.HRLabel.text = [NSString stringWithFormat:@"%d", ValueHR];
    // Ukladani hodnot HR do pole
    [HRhodnoty addObject:[NSNumber numberWithInt:ValueHR]];
```

```
    int velikostPoleHR= [HRhodnoty count];
    int sumHR = 0;      // Proměnná pro sumu vzorků pole HR
    int avgHR;          // Proměnná pro průměrnou TF
    int maxHR;          // Proměnná pro maximální TF
    if (velikostPoleHR % 10 == 0) {
      for (int k=0; k<velikostPoleHR; k++) {
        int aktualniHR = [[HRhodnoty objectAtIndex:k]
integerValue];
        sumHR = sumHR + aktualniHR;
        avgHR = sumHR/velikostPoleHR;
        self.avgHRLabel.text = [NSString stringWithFormat:@"%d
bpm", avgHR];
      }
    }
    maxHR = [[HRhodnoty valueForKeyPath:@"@max.intValue"]
integerValue];
    // Převod max HR na string
    maxHRvalue = [NSString stringWithFormat:@"%d bpm",maxHR];
```

Metoda `bleDidReceiveData` je volána automaticky, pokud mobilní telefon detekuje příjem dat dříve navázanou Bluetooth komunikací. První cyklus má za úkol čtení všech dat, která MCU odesílá. Následně jsou již podmínkou vybírána pouze ta data, která nás v daný moment zajímají. V tuto chvíli se jedná o aktuální přijatou hodnotu TF, přenášenou jako typ `"uint16_t"` pod identifikátorem `0x0B` (hodnota 11 v hexadecimální soustavě). Po dekódování přijaté hodnoty je možné její okamžité zobrazení v *"labelu"* `HRLabel`.

Pro možnost výpočtu maximální a průměrné TF je aktuální přijatá hodnotu TF uložena do pole `HRhodnoty` a to jako celočíselný typ *int*. Následně je provedeno stanovení velikosti daného pole a deklarace proměnných pro práci s polem. Cílem je zobrazovat průměrnou TF v průběhu samotné aktivity a nikoli až následně při jejím vyhodnocení. Proto je nutné průmětnou TF počítat průběžně. Průměrná TF je tedy vypočtena vždy, když je počet vzorků TF v poli dělitelný bezezbytku deseti. Jedná se o rozumný kompromis mezi výpočetní náročností a frekvencí zobrazování průměrné TF. Pozornému čtenáři neunikne, že hodnota průměrné TF bude tedy zobrazena každých 10 sekund. Její výpočet je velice jednoduchý. Je stanoven počet vzorků TF v poli a sleduje se, je-li splněna podmínka dělitelnosti bezezbytku hodnotou 10. Následně je určena suma vzorků, která je podělena velikostí pole. Vypočtená hodnota je zobrazena v *"labelu"* `AvgHRHRLabel`.

Stanovení maximální TF není výpočetně náročné a je možné je provádět průběžně s nárůstem pole. Pro stanovení maximální celočíselné hodnoty pole slouží příkaz `valueForKeyPath:@"@max.intValue"`. Výsledná hodnota je na závěr převedena na typ *string*, ve kterém je později uložena pomocí třídy *NSUserDefaults*.

```
// Výpočet zóny zatížení a zobrazení v Labelu
NSUserDefaults *defaults = [NSUserDefaults standardUserDefaults];
NSString *maxHRUserString = [defaults
stringForKey:@"maxHRuserbeh"];        // Načtení hodnoty maxTF
int maxHRUser = [maxHRUserString intValue];

// Převod string na int maxTF
int zona1a = 0.6 * maxHRUser;
int zona1b = 0.75 * maxHRUser;
int zona2b = 0.85 * maxHRUser;
int zona3b = 0.95 * maxHRUser;

 if (ValueHR < zona1a) {
  // Zóna regenerace
  int znacka = 1;
  self.zoneLabel.text = [NSString stringWithFormat:@"%d", znacka];
}
else if (ValueHR >= zona1a & ValueHR < zona1b) {
  // Zóna základní vytrvalosti
  int znacka = 2;
  self.zoneLabel.text = [NSString stringWithFormat:@"%d", znacka];
}
else if (ValueHR >= zona1b & ValueHR < zona2b) {
  // Zóna tempové vytrvalosti
  int znacka = 3;
  self.zoneLabel.text = [NSString stringWithFormat:@"%d", znacka];
}
else if (ValueHR >= zona2b & ValueHR < zona3b) {
  // Zóna speciální vytrvalosti
  int znacka = 4;
  self.zoneLabel.text = [NSString stringWithFormat:@"%d", znacka];
}
else if (ValueHR >= zona3b) {
  // Zóna rychlosti
  int znacka = 5;
  self.zoneLabel.text = [NSString stringWithFormat:@"%d", znacka];
}
```

V rámci popisu dostupných funkcí bylo uvedeno, že mobilní aplikace SportsBrain dokáže určit individuální TF_{max} a to standardním výpočtem podle vztahu (2), nebo přesněji provedením fitness testu. Bez ohledu na způsob stanovení je však výsledná hodnota TF_{max} uložena pomocí mechanismu *UserDefaults* a je dostupná kdekoliv v aplikaci. Tento mechanismus se implementuje prostřednictvím třídy *NSUserDefaults*. Uložená hodnota je jednoznačně identifikována tzv. *klíčem* a uchovává se v souborovém systému. K uloženým datům lze přistupovat voláním metody *standardUserDefauls*. Hodnota TF_{max} pro běh je uložena pod klíčovým slovem `"maxHRuserbeh"` a je využita i pro načtení do proměnné `"maxHRUser"`.

Pokud je provedeno načtení TF_{max}, je možné tuto hodnotu využít pro stanovení individuálních pásem intenzity tréninku. Pásma jsou stanovena jako procentuální součin na základě tabulky 2 (viz kap. 2.3.2). Následně je již jednoduchou podmínkou stanoveno, ve kterém z vypočtených pásem se uživatel na základě své aktuální naměřené hodnoty TF nachází. Informace je zobrazena v *"labelu"* `zoneLabel`.

```
// Příjem aktualni hodnoty respirace
else if (data[i] == 0x0C){
  UInt16 ValueRR;
  ValueRR = data[i+2] | data[i+1] << 8;
  // Ukladani hodnot RR do pole
  [RRhodnoty addObject:[NSNumber numberWithInt:ValueRR]];
  int velikostPoleRR= [RRhodnoty count];
  int aktualniRR = [[RRhodnoty objectAtIndex:velikostPoleRR-1]
integerValue];
  int predchoziRR;                // Předposlední hodnota pole
  int j = velikostPoleRR - 2;
  if (velikostPoleRR >=2) {
     predchoziRR = [[RRhodnoty objectAtIndex:j] integerValue];
     }
  // Mazání starých objektů v polích RRhodnoty
  if (velikostPoleRR >= 10) {
    [RRhodnoty removeObjectAtIndex:0];
  }
  // Indikace nádechu/výdechu pomocí šipek
  if (aktualniRR ==1) {
    indikaceNadechu.hidden = NO;
    indikaceVydechu.hidden = YES;
  }
  else if (aktualniRR == 0){
  indikaceNadechu.hidden = YES;
  indikaceVydechu.hidden = NO;
  }
  else {
  indikaceNadechu.hidden = YES;
  indikaceVydechu.hidden = YES;
  }
```

Informaci o stavu respirace ve formátu 1 – nádech, 0 – výdech je přenášena jako typ `"uint16_t"` pod identifikátorem `0x0C` (hodnota 12 v hexadecimální soustavě). Nejdříve jsou přednášená data dekódována a následně hodnota 1 či 0 uložena do pole `RRhodnoty` typu *NSMutableArray*.

Uživatele však zajímá hodnota dechové frekvence, kterou sensor respiračních pohybů přímo neposkytuje. Je tedy třeba vytvořit algoritmus, který bude pomocí hodnoty nádech/výdech, přijímané každou sekundu, počítat aktuální hodnotu DF. Nejdříve je stanovena velikost pole `RRhodnoty` a uložena poslední hodnota pole do nové proměnné `aktualniRR` a předposlední hodnota pole do proměnné `predchoziRR`. Protože stačí znát jen posledních několik hodnot pole `RRhodnoty`, lze starší hodnoty promazávat a tím šetřit místo v paměti mobilního telefonu. Uměle je tak udržována velikost pole na hodnotě 10 prvků.

Další část zdrojového kódu slouží pro signalizaci nádechu či výdechu přímo v mobilní aplikaci. Na obrázku zobrazujícím detekované biologické veličiny v mobilní aplikaci (obr. 31) je možné si povšimnout, že u zobrazovaného parametru DF je umístěna malá šipka směřující vzhůru. Tato šipka signalizuje, že se uživatel nachází ve fázi nádechu. Nachází-li se uživatel ve fázi výdechu, směřuje šipka směrem dolů.

```
// Výpočet dechové frekvence
if (aktualniRR ==1 & predchoziRR ==0) {
  double cas = CACurrentMediaTime();          // Zjsti aktuální čas
  [casoprostor addObject:[NSNumber numberWithDouble:cas]];
  int velikostCasoprostoru = [casoprostor count];
  double hodnotaCasu = [[casoprostor
objectAtIndex:velikostCasoprostoru-1] doubleValue];
  double predchoziCas;        // Proměnná pro předchozí čas nádechu
  double casProVypocet;       // Rozdíl současné a předchozí značky
  if (velikostCasoprostoru > 2) {
    int index = velikostCasoprostoru - 2;
    predchoziCas = [[casoprostor objectAtIndex:index] doubleValue];
    casProVypocet = hodnotaCasu - predchoziCas;
  }
  // Mazání starých hodnot v časoprostoru
  if (velikostCasoprostoru >= 10) {
    [casoprostor removeObjectAtIndex:0];
  }
  if (casProVypocet >= 1) {
    double DFa = 60/casProVypocet;              // Výpočet DF
    double DF = round(DFa);
    int dech = (int) DF;                        // Výsledná DF
    // Zobrazeni DF na telefonu
    self.RRLabel.text = [NSString stringWithFormat:@"%d", dech];
```

V předchozím kroku byla do proměnných `aktualniRR` a `predchoziRR` uložena informace o stavu respirace. Změna z výdechu na nádech nastane tehdy, pokud se hodnota v poli `RRhodnoty` změní z 0 na 1. Pokud se tak stane, je aktuální hodnota času uložena do proměnné `cas` pomocí metody *CACurrentMediaTime*. Zaznamenaná hodnota času je uložena do pole `casoprostor` a stanovena velikost tohoto pole. Poslední hodnota času z pole je uložena do proměnné `hodnotaCasu` a předposlední do proměnné `predchoziCas`. Rozdíl poslední a předposlední časové značky je uložen do proměnné `casProVypocet`. Nyní je tedy známo, jaký čas uplynul mezi posledními dvěma nádechy. Tento čas představuje hledaný respirační interval, který je možné použít ve vztahu (3) a přímo stanovit aktuální hodnotu DF. Výsledná hodnota DF obsažená v proměnné `DF` je na závěr převedena na typ *double*, zaokrouhlena na celá čísla a zobrazena v *"labelu"* `RRLabel`.

```
// Uložení hodnoty DF do pole
    [DFhodnoty addObject:[NSNumber numberWithInt:dech]];
    int velikostPoleDF= [DFhodnoty count];
    int sumDF = 0;  // Proměnná pro sumu vzorků pole DF
    int avgDF = 0;  // Proměnná pro průměrnou DF
    int maxDF = 0;  // Proměnná pro maximální DF
    for (int k=0; k<velikostPoleDF; k++) {
      int aktualniDF = [[DFhodnoty objectAtIndex:k] integerValue];
      sumDF = sumDF + aktualniDF;
      avgDF = sumDF/velikostPoleDF;
      maxDF = [[DFhodnoty valueForKeyPath:@"@max.intValue"]
integerValue];
  }
  // Převod maxDF a avgDF na string
  avgDFvalue = [NSString stringWithFormat:@"%d bpm",avgDF];
  maxDFvalue = [NSString stringWithFormat:@"%d bpm",maxDF];}}
```

Vypočtené hodnoty DF jsou postupně ukládány do pole `DFhodnoty`. Pro analýzu sportovní aktivity je nyní třeba určit průměrnou a maximální DF. Toto se děje stejným způsobem jako v případě stanovení průměrné a maximální TF. Je stanovena suma všech vzorků v poli DF a ta je podělena jeho celkovou velikostí. Pro stanovení maximální celočíselné hodnoty pole je opět použit příkaz `valueForKeyPath:@"@max.intValue"`. Výsledné hodnoty průměrné a maximální DF jsou na závěr převedeny na typ *string*, ve kterém jsou později uloženy pomocí třídy *NSUserDefaults*.

Veškeré výše popsané akce probíhají ihned po stisknutí tlačítka *"Connect"*. Uživatel tak vidí sledované biologické parametry tak, jak je znázorněno na obrázku 31. Vše je připraveno a nic nebrání zahájit sportovní aktivitu stisknutím tlačítka *"Start"*.

Zahájení sportovní aktivity

Zahájením sportovní aktivity dojde k několika podstatným akcím. Především je volána metoda `startPressed`, v rámci které se mimo jiné aktivuje senzor GPS. Dále se zobrazí tabulka obsahující čtveřici polí s informacemi o aktuální délce tréninku, absolvované vzdálenosti, průměrném tempu a průměrné tepové frekvenci. Obrazovka je doplněna tlačítkem pro možnost zastavit záznam aktivity. Uživatelské rozhraní aplikace v režimu zobrazení průběhu běžeckého tréninku je znázorněno na obrázku (obr. 32). V následující části textu bude uveden zdrojový kód, který je prováděn po zahájení aktivity. Na tomto místě je korektní uvést, že zatímco sledování biologických parametrů (zejména DF) výše prezentovaným způsobem je novinkou na poli mobilních aplikací a v tomto ohledu je systém SportsBrain unikátní, tak sledování sportovní aktivity s pomocí GPS je dnes naprostým standardem. Způsob sledování polohy pomocí GPS technologie je v aplikaci inspirováno volně dostupným kódem z [9].

Obr. 32: Zobrazení průběhu běžeckého tréninku

```
-(IBAction)startPressed:(id)sender
{
    // Schovej tlačítko startu
    self.startButton.hidden = YES;

    // Ukaž vše co souvisí s aktivitou
    self.stopButton.hidden = NO;
    self.timeLabel.hidden = NO;
                .
                .
                .
    self.seconds = 0;
    self.distance = 0;

    self.timer = [NSTimer scheduledTimerWithTimeInterval:(1.0)
target:self selector:@selector(eachSecond) userInfo:nil
repeats:YES];

    self.locations = [NSMutableArray array];
    [self startLocationUpdates];
}
```

Metoda `startPressed` je volána po stisknutí tlačítka *"Start"*. Dochází ke zobrazení tabulky obsahující informace o průběhu tréninku. Zároveň zmizí tlačítko *"Start"* a je zobrazeno tlačítko *"Stop"*, které slouží pro ukončení aktivity. Nastavení zobrazení prvků na obrazovce je provedeno metodou *hidden*. V prezentované části zdrojového kódu je toto nastavení uvedeno jen pro trojici prvků jako příklad. Zajímavější je následná inicializace času a vzdálenosti aktivity na hodnotu 0 a volání metody *timer*, která slouží pro nastavení intervalu volání metody `eachSecond`. Následně je v databázi aplikace vytvořeno pole pro ukládání polohy uživatele v průběhu tréninku. Nakonec se aktivuje sledování polohy GPS pomocí nástroje *Core Location* prostřednictvím metody `startLocationUpdates`.

Rámec Core Location – detekce polohy aktivity

Technologie rámce Core Location umožňuje stanovit přesnou polohu mobilního zařízení především pomocí družicového navigačního systému GPS. V prvních chvílích po stisknutí tlačítka *"Start"* však může trvat navazování kontaktu s družicemi a výpočet polohy i několik minut. Proto rámec Core Location podporuje také technologii A-GPS (Assisted GPS), která využívá asistenčního serveru pro podporu procesu lokalizace. Rámec provádí výpočet polohy uživatele prostřednictvím kombinace trojice lokalizačních technik. Již uvedeným GPS s podporou A-GPS, dále pomocí triangulace vysílačů mobilního signálu a také pomocí technologie WPS (Wi-Fi Positioning Service). Není cílem této práce popisovat principy jednotlivých lokalizačních technik, více se lze dočíst např. v [10]. Technologie, které rámec Core Location pro určení zeměpisné polohy používá, jsou pro programátora nedostupné. Rámci nelze definovat, kterou z výše popsaných technologií má zvolit. Lze jen definovat, s jakou přesností má být zeměpisná poloha určena. Rámec následně automaticky vybere z dostupných technologií tu, která nejlépe splní definované požadavky.

Definice přesnosti probíhá v rámci třídy *CLLocationManager*, neboli správce zeměpisné polohy (Location manager). V rámci metody `startPressed` je volána metoda `startLocationUpdates`, jejíž zdrojový kód je uveden.

```
- (void)startLocationUpdates
{
  if (self.locationManager == nil) {
    self.locationManager = [[CLLocationManager alloc] init];
  }
  self.locationManager.delegate = self;
  self.locationManager.desiredAccuracy = kCLLocationAccuracyBest;
  self.locationManager.activityType = CLActivityTypeFitness;

  self.locationManager.distanceFilter = 10;

  if ([self.locationManager
respondsToSelector:@selector(requestWhenInUseAuthorization)]) {
    [self.locationManager requestWhenInUseAuthorization];
  }
  [self.locationManager startUpdatingLocation];
}
```

Podmínkou se nejdříve zkontroluje, zda již správce zeměpisné polohy nebyl vytvořen. Pokud tomu tak není, je vytvořen a inicializován. Následně je nastavena přesnost, s jakou bude delegát uživatele lokalizovat. Přesnost se nastavuje prostřednictvím proměnné `CLLocationAccurancy`, což je datový typ definovaný jako *double*. Zde je použita konstanta `kCLLocationAccurancyBest`. Tímto je rámci Core Location řečeno, aby pro určení polohy použil nejpřesnější dostupnou metodu. Dále je definován typ aktivity, při které se bude zeměpisná poloha využívat. Nastavením `activityType` na `CLActivityTypeFitness` bude mobilní telefon inteligentně zapínat a vypínat senzor GPS pro maximalizaci výkonu baterie. Zastaví-li tak uživatel na křižovatce, či aby si na chvíli odpočinul, GPS senzor se vypne.

Nastavením konstanty `distanceFilter` na hodnotu 10 je zabezpečeno, že detekované polohové body budou jednotnější. Zde se nemění přesnost detekce, ale pouze se zaokrouhluje výsledný polohový bod na 10 metrů. Tohoto je využito při vyhodnocení aktivity, kde aplikace vykresluje polohové body do mapy a vytváří linii běhu. Ta je tímto jednotnější. Závěrem metody je nutné vložit upozornění, které se uživateli zobrazí po prvním spuštění aplikace. Využívá-li některá aplikace systému iOS polohu, je nutné se nejprve uživatele dotázat, zda s využitím své polohy souhlasí. Tato podmínka se objevuje ve všech verzích iOS, avšak až do verze 8 je nutné dotaz vkládat manuálně.

Nyní je aplikace připravena začít zjišťovat zeměpisnou polohu a voláním `startUpdatingLocation` je správce polohy spuštěn. Správce následně provede zjištění polohy. Musí být však nastaven také konec detekce. Jinak je detekována každá změna polohy, přesahující aktuální nastavení filtru vzdálenosti. Jednotlivé hodnoty polohy je nutné ukládat pro možnost následného vyhodnocení aktivity (rámec *Core Data)*.

Rámec Core Data – záznam polohy aktivity

Core Data je nástroj pro uchovávání dat. Zde nebude popsána celá struktura tohoto rámce (více se lze dočíst v [15]). Bude však uvedeno, jakým způsobem je tento nástroj využit v rámci mobilní aplikace SportsBrain.

Při vytváření nového projektu v prostředí Xcode je nutné povolit využití Core Data. Implementace Core Data do aplikace až v průběhu vývoje je velmi složitá. Pro šablonu "*Master Detail Application*" je však možné nástroj pro správu dat vytvořit společně se zakládáním nového projektu. Automaticky je pak vytvořen také soubor *SportsBrainBeta.xcdatamodeld*. Nástroj Core data následně umožnuje navrhovat datový model vizuálně, bez nutnosti zápisu zdrojového kódu. V editoru datového modelu jsou vytvořeny entity (entities) a z nich následně ve zdrojovém kódu spravované objekty (managed objects). Entita je tvořena atributy, které jsou čtyř druhů. Aplikace SportsBrain využívá dvojici entit s názvy *"Run"* a *"Location"*.

Entita *"Location"* obsahuje atributy: zeměpisná šířka, zeměpisná délka a časová značka. V této datové struktuře jsou uloženy body, které jsou sbírány v průběhu aktivity pomocí rámce Core Location. Entita má nastaven vztah k druhé entitě *"Run"*. Ta obsahuje atributy: vzdálenost, délka a časová značka. Zde jsou ukládány celkové informace o průběhu aktivity. Entita má taktéž nastaven vztah k entitě předchozí. Po dokončení práce na datovém modelu dojde automaticky k vytvoření čtveřice souborů: *Run.h, Run.m, Location.h a Location.m*. Tyto soubory představují datový model přenesený do zdrojového kódu a jsou dostupné na přiloženém DVD. K datovému modelu lze v aplikaci přistupovat a využívat jej k ukládání potřebných dat (aktuální poloha uživatele).

```
- (void)locationManager:(CLLocationManager *)manager
     didUpdateLocations:(NSArray *)locations
{
  for (CLLocation *newLocation in locations) {
    if (newLocation.horizontalAccuracy < 20) {
      if (self.locations.count > 0) {
        self.distance += [newLocation
distanceFromLocation:self.locations.lastObject];
      }
      [self.locations addObject:newLocation];
    }
  }
}
```

Location Manager poskytuje nová data vždy, když uživatel změní svou polohu. Před uložením nového bodu je však proveden test přesnosti. Pokud není zařízení v danou chvíli schopné lokalizovat svoji polohu přesněji než na 20 m, daný bod je považován za nepřesný a není uložen. Je-li podmínka přesnosti splněna, je v databázi k celkové vzdálenosti přičten nový úsek. Následně je polohový bod uložen do databáze.

Po stisknutí tlačítka “Start“ pro zahájení aktivity byla volána metoda `eachSecond`. Tato metoda je důležitá z hlediska aktualizace údajů založených na detekci polohy uživatele, které se zobrazují na obrazovce v průběhu aktivity.

Výpočet vzdálenosti a tempa

Metoda `eachSecond`, jak její název napovídá, je volána každou sekundu a slouží k aktualizaci dat zobrazovaných na obrazovce průběhu aktivity.

```
- (void)eachSecond {
  self.seconds++;
  // Aktualizace uplynulého času
  self.timeLabel.text = [NSString stringWithFormat:@"%@",
[MathController stringifySecondCount:self.seconds
usingLongFormat:NO]];
  // Aktualizace uplynulé vzdálenosti
  self.distLabel.text = [NSString stringWithFormat:@"%@",
[MathController stringifyDistance:self.distance]];
  // Aktualizace aktuálního tempa
  self.paceLabel.text = [NSString stringWithFormat:@"%@",
[MathController stringifyAvgPaceFromDist:self.distance
overTime:self.seconds]];
}
```

Aktualizace uvedených polí se odvolávají především na metody třídy `MathController`. Jedná se o zvláštní třídu typu NSObject, která obsahuje metody pro stanovení tempa či vzdálenosti. Třída je zahrnuta prostřednictvím souboru `mathController.m` v rámci zdrojového kódu pro *NewRunViewController* a ten tak může využívat její metody.

Třída `MathController` byla využita z volně dostupného zdrojového kódu a veškeré informace o její funkcionalitě jsou dostupné v [9]. Z tohoto důvodu příslušný zdrojový kód není uveden. Je dostupný včetně komentářů na přiloženém DVD. Průměrné tempo je vypočteno jako podíl času a vzdálenosti uplynulého tréninku. Použitá metoda `MathController` dále slouží k přidělení barevné značky každému bodu polohy v závislosti na tom, jak rychle se uživatel v daném bodě pohybuje.

Ukončení aktivity

V předchozím textu byly vyčerpávajícím způsobem popsány činnosti aplikace související s obrazovkou pro sledování průběhu běžeckého tréninku. Nyní přichází okamžik, kdy má uživatel již “doběháno“ a přeje si aktivitu ukončit a zobrazit její vyhodnocení. K ukončení aktivity slouží již dříve zmíněné tlačítko *“Stop“*. Po stisknutí je prostřednictvím metody protokolu `UIActionSheetDelegate` nabídnuta uživateli možnost pro její smazání či vyhodnocení. Zdrojový kód metody `stopPressed` je uveden níže. Obrazovka aplikace s možnostmi je na následující straně (obr. 33).

```
- (IBAction)stopPressed:(id)sender {
  UIActionSheet *actionSheet = [[UIActionSheet alloc]
initWithTitle:@"" delegate:self
cancelButtonTitle:@"Cancel" destructiveButtonTitle:nil
```

```
otherButtonTitles:@"Evaluate", @"Delete", nil];
  actionSheet.actionSheetStyle = UIActionSheetStyleDefault;
  [actionSheet showInView:self.view];}
```

Obr. 33: Ukončení aktivity

V předchozí části zdrojového kódu byly pouze vytvořeny dvě volby pro ukončení aktivity. Nyní je třeba detekovat, kterou volbu uživatel stiskl a jakým způsobem má tedy aplikace reagovat. Po stisknutí tlačítka *"Cancel"* dochází k návratu zpět na průběh aktivity. Stiskem tlačítka *"Delete"* je uživatel přenesen zpět do hlavní nabídky (obr. 29). Konečně stisknutím volby *"Evaluate"* je uživatel přenesen na obrazovku vyhodnocení aktivity, odkud má také možnost jejího uložení. Před načtením obrazovky pro vyhodnocení aktivity však musí aplikace provést následující akce:

```
- (void)actionSheet:(UIActionSheet *)actionSheet
clickedButtonAtIndex:(NSInteger)buttonIndex
{
  // save
  if (buttonIndex == 0) {
    [self saveRun]; // Po stisknutí save se uloží běh
    [self performSegueWithIdentifier:detailSegueName sender:nil];

    // Uložení avg HR a max HR do NSUserDefaults
    NSString *UlozavgHR = self.avgHRLabel.text;
    NSUserDefaults *defaults = [NSUserDefaults
standardUserDefaults];
    [defaults setObject:UlozavgHR forKey:@"avgHR"];
    [defaults synchronize];
    ((AppDelegate*)[[UIApplication sharedApplication]
delegate]).avgHRprenos = UlozavgHR;//avgHR je uložena
    [defaults setObject:maxHRvalue forKey:@"maxHR"];
    [defaults synchronize];
    ((AppDelegate*)[[UIApplication sharedApplication]
delegate]).maxHRprenos = maxHRvalue;//maxHR je uložena do proměnné
avgHRprenos
```

```
    // Uložení avg DF a max DF do NSUserDefaults
    [defaults setObject:maxDFvalue forKey:@"maxDF"];
    [defaults synchronize];
    ((AppDelegate*)[[UIApplication sharedApplication]
delegate]).maxDFprenos = maxDFvalue;//maxDF je uložena
    [defaults setObject:avgDFvalue forKey:@"avgDF"];
    [defaults synchronize];
    ((AppDelegate*)[[UIApplication sharedApplication]
delegate]).avgDFprenos = avgDFvalue;//maxDF je uložena

  // delete
  } else if (buttonIndex == 1) {
      [self.navigationController
popToRootViewControllerAnimated:YES];
  }
}
```

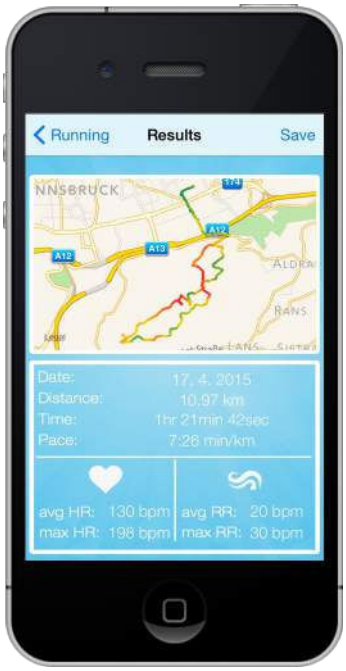
Metoda `actionSheet` slouží pro nastavení akcí po stisku jednoho z tlačítek, která byla vytvořena pomocí protokolu `UIActionSheetDelegate`. Stisknutím tlačítka *"Evaluate"* (index 0) dochází nejprve k volání metody `saveRun` (dostupná na přiloženém DVD), která ukládá výsledky založené na GPS do databáze Core Data. Následuje příkaz pro přenos všech parametrů na obrazovku vyhodnocení aktivity.

Při popisu příjmu a zobrazení dat z MCU byly uvedeny způsoby stanovení maximální a průměrné TF a DF. Tyto hodnoty byly uloženy do proměnných, dostupných kdekoliv v rámci souboru `NewRunViewController`. Cílem je však jejich uložení způsobem, aby byly přístupné v rámci celé mobilní aplikace. Proto je zde opět využito mechanismu *UserDefaults* implementovaném třídou `NSUserDefaults`. Mechanismus zde ukládá dříve vypočtené hodnoty maximální a průměrné TF a DF prostřednictvím klíčů `"avgHR"`, `"maxHR"`, `"avgDF"`, `"maxDF"` do souborového systému. Index 1 slouží pro označení tlačítka pro smazání aktivity, kdy je uživatel přenesen zpět do hlavní nabídky.

Vyhodnocení aktivity

Obrazovka vyhodnocení aktivity obsahuje kompletní shrnutí průběhu běžeckého tréninku. Horní polovinu obrazovky vyplňuje mapa zobrazující trasu absolvovaného běhu, a to včetně barevné škály odpovídající rychlostnímu profilu tratě. Pomalé úseky jsou znázorněny červenou barvou, rychlé úseky potom barvou zelenou. Zbylou část obrazovky vyplňuje tabulka s naměřenými údaji. Ta je pro lepší přehlednost rozdělena do tří částí. První část obsahuje obecná data (datum tréninku, celkový čas aktivity) spolu s daty založenými na GPS měření (vzdálenost, průměrné tempo). Druhá a třetí část tabulky obsahují informace získané s pomocí mikroprocesorové části systému. Uživateli je zobrazena informace o maximální a průměrné hodnotě dvojice sledovaných biologických parametrů. Uživatel má možnost vyhodnocenou aktivitu uložit pomocí tlačítka *"Save"*, umístěného v navigační lište. Po uložení jsou vyhodnocená data o aktivitě přenesena do modulu *"Last activity"*. Obrazovka vyhodnocení běžeckého tréninku, včetně naměřených dat je zobrazena na obrázku na následující straně (obr. 36). Zdrojový kód, řídící chování této obrazovky, je zapsán v souborech

`DetailViewController.h` a `.m`. Zde budou uvedeny jen nejdůležitější části implementačního souboru. Celý kód je dostupný na přiloženém DVD.



Obr. 34: Obrazovka vyhodnocení aktivity

```
- (void)configureView {
  self.distanceLabel.text = [NSString
stringWithFormat:@"%@",[MathController
stringifyDistance:self.run.distance.floatValue]];

  NSDateFormatter *formatter = [[NSDateFormatter alloc] init];
  [formatter setDateStyle:NSDateFormatterMediumStyle];
  self.dateLabel.text = [NSString stringWithFormat:@"%@",[formatter
stringFromDate:self.run.timestamp]];

  self.timeLabel.text = [NSString stringWithFormat:@"%@",
[MathController stringifySecondCount:self.run.duration.intValue
usingLongFormat:YES]];

  self.paceLabel.text = [NSString stringWithFormat:@"%@",
[MathController
stringifyAvgPaceFromDist:self.run.distance.floatValue
overTime:self.run.duration.intValue]];

  // Načti avg HR a max HR z NSUserDefaults
  self.avgHRLabel.text = [[NSUserDefaults standardUserDefaults]
objectForKey:@"avgHR"];
  self.maxHRLabel.text = [[NSUserDefaults standardUserDefaults]
objectForKey:@"maxHR"];

  // Načti avg DF a max DF z NSUserDefaults
  self.avgDFLabel.text = [[NSUserDefaults standardUserDefaults]
objectForKey:@"avgDF"];
  self.maxDFLabel.text = [[NSUserDefaults standardUserDefaults]
objectForKey:@"maxDF"];

  [self loadMap];
}
```

Metoda `viewDidLoad`, která je automaticky volána po načtení obrazovky, ihned volá vytvořenou metodu `configureView`. Ta načítá data na obrazovku a zobrazuje mapu. Přitom využívá metody v již dříve zmíněném souboru `MathController` a data uložená jak prostřednictvím *Core Data*, tak prostřednictvím *NSUserDefaults*.

Nejdříve jsou načtena data do labelů v první části tabulky. Jedná se o celkovou vzdálenost, datum, celkový čas aktivity a tempo. Tato data byla uložena prostřednictvím nástroje *Core Data*. Následně jsou z *NSUserDefaults* načtemy hodnoty maximální a průměrné TF a DF, a to s pomocí *klíčů*, které byly pro dané hodnoty použity v rámci *NewRunViewControlleru*. Závěrem je volána metoda `loadMap`, která má za úkol vykreslit mapu pomocí rámce *Map Kit*.

Již byl popsán rámec Core Location v souvislosti se stanovením vzdálenosti a tempa. Rámec Map Kit představuje nástavbu rámce Core Location a slouží pro zobrazení detekovaných pozičních bodů v mapových podkladech. Zobrazení trasy sportovní aktivity a vykreslení rychlostního profilu tratě je doplňkovou funkcí aplikace SportsBrain. Spíše, než pro splnění zadání práce, je této funkce využito pro demonstraci dostupných možností mobilních aplikací. Zdrojový kód pro vykreslení trasy a rychlostního profilu byl pro aplikaci převzat z [9]. Nebude zde proto proveden jeho detailní popis. Kompletně je přístupný na přiloženém DVD a veškeré informace o rámci Map Kit čtenář nalezne v odborné literatuře (např. [15]). Důležité a unikátní funkce, jako je způsob stanovení a zobrazení TF a DF, byly již vyčerpávajícím způsobem v této kapitole popsány.

Uložení aktivity

Ještě než bude popsán modul pro záznam cyklistického tréninku, bude popsána činnost, kterou aplikace provádí po stisknutí tlačítka “Save“ na obrazovce vyhodnocení aktivity. Metoda `SaveToLastActB` provádí uložení obsahu všech polí (labelů) do souborového systému aplikace pomocí mechanismu *UserDefaults*. Zde se jedná o uložení všech výsledných dat pro možnost následného zobrazení v modulu *“Last activity“*. Protože se způsob ukládání obsahu polí liší jen použitým *“klíčem“*, je zde jako příklad uvedeno uložení jediného parametru, a to parametru data.

```
- (IBAction)SaveToLastActB:(id)sender {

  // Uložení data
  NSString *Ulozdatum = self.dateLabel.text;
  NSUserDefaults *defaults = [NSUserDefaults standardUserDefaults];
  [defaults setObject:Ulozdatum forKey:@"datum"];
  [defaults synchronize];
  ((AppDelegate*)[[UIApplication sharedApplication]
delegate]).datum = Ulozdatum;
```

Uvedeným způsobem je uložen obsah všech polí na obrazovce vyhodnocení aktivity. Je tak přístupný kdekoliv v rámci aplikace.

4.3.7 Modul “New Bike“

Modul “*Nové kolo*“ (“New bike“) slouží pro záznam a vyhodnocení cyklistického tréninku. Tento modul není třeba celý popisovat s rozborem zdrojového kódu. Většina funkcí je totožná s modulem pro záznam běžeckého tréninku. Také zde dochází ke spojení s MCU částí systému prostřednictvím technologie Bluetooth. Stejným způsobem je proveden výpočet a zobrazení tepové a dechové frekvence. Aktivita je také naprosto stejným způsobem vyhodnocena. Záznam cyklistického tréninku je veden jako samostatný modul z důvodu rozdílnosti individuální TF_{max}, které je uživatel při cyklistice schopen dosáhnout. Jak již bylo uvedeno, TF_{max} je ve většině případů při cyklistice nižší, než při běhu. Je proto žádoucí samostatného výpočtu pásem intenzity tréninku. Dalším rozdílem je parametr rychlosti, který je v průběhu aktivity zobrazován namísto parametru tempa. Rychlost je pro cyklistiku často podstatnějším parametrem, než tempo. Rozdíly modulu “New bike“ a již dříve popsaného modulu “New run“ jsou tedy:

- Samostatný výpočet pásem intenzity tréninku založený na hodnotě TF_{max} pro cyklistiku (stanovenou fitness testem či výpočtem)
- Zobrazení průměrné rychlosti na obrazovce aktivity namísto tempa

Uživatel je přenesen na úvodní obrazovku modulu “New bike“ po stisknutí příslušné volby v hlavní nabídce. I se zde nachází tlačítko *“Connect“*, které slouží pro propojení mobilní aplikace s mikroprocesorem. Po stisknutí tlačítka probíhá přenos dat z mikroprocesoru, kalibrace senzoru respirace a nakonec zobrazení parametrů TF a DF stejně, jako u modulu pro běh. Také zde je záznam aktivity proveden stisknutím tlačítka “Start“. Úvodní obrazovka modulu “New Bike“ a obrazovka průběhu aktivity je na obrázku (obr. 35).

Obr. 35: Úvodní obrazovka a průběh aktivity v modulu “New bike“

Řídící soubory jsou zde `NewBikeViewController.h` a `.m` a jak již bylo uvedeno, většina zdrojového kódu se v ničem neliší od modulu pro záznam běžeckého tréninku. Rozdíl je pouze v načtení jiné hodnoty TF_{max}. Hodnota TF_{max} pro cyklistiku, stanovená fitness testem nebo výpočtem podle vztahu (2), je načtena pomocí třídy *NSUserDefaults*. Klíčové slovo je zde `"maxHRuserkolo"`. Hodnota je použita pro načtení do proměnné `"maxHRUser"`. Následně již probíhá výpočet pásem intenzity tréninku a zobrazení příslušného pásma stejným způsobem, jako v případě záznamu běžeckého tréninku.

Zobrazení průměrné rychlosti je opět založeno na volání metod v třídě `MathController`. Tentokrát však byla jedna z metod upravena tak, aby namísto informace o průměrném tempu poskytovala informaci o průměrné rychlosti. Je zřejmé, že zdrojový kód se liší opravdu jen v použití jiného klíče pro načtení TF_{max} a voláním pozměněné metody z třídy `MathController`. Není jej tedy třeba opakovaně uvádět. Je však dostupný na přiloženém DVD.

Cyklistický trénink je po stisknutí tlačítka *"Evaluate"* stejným způsobem vyhodnocen a to včetně zobrazení trasy tréninku v mapových podkladech, zobrazení vzdálenosti, času, či sledovaných biologických parametrů. Také zde má uživatel možnost aktivitu uložit již dříve uvedeným způsobem, pro následné zobrazení v rámci modulu *"Last activity"*.

4.3.8 Modul "Last activity"

Modul "*Poslední aktivita*" ("Last activity") slouží pro vyvolání dat, uložených v rámci vyhodnocení běžeckého či cyklistického tréninku. Uživateli jsou zobrazeny veškeré informace, které aplikace zobrazila při vyhodnocení aktivity. Nedochází zde ale ke zobrazení trasy tréninku v mapových podkladech.

Součástí každé moderní mobilní aplikace pro záznam sportovní aktivity musí být možnost sdílet výsledky tréninku prostřednictvím sociálních sítí či e-mailu. Tato funkce nechybí ani v aplikaci SportsBrain. Uživateli jsou po stisknutí tlačítka se specifickým symbolem v navigační liště nabídnuty všechny dostupné možnosti pro sdílení. To je umožněno nejen prostřednictvím e-mailu a sociálních sítí, ale také pomocí jiných aplikací nainstalovaných v mobilním telefonu. Uživatel má tak možnost zkopírovat údaje o své sportovní aktivitě na "zeď" sociální sítě Facebook, do příspěvku sociální sítě Twitter, do nového e-mailu, či nové SMS, ale také do aplikace OneNote. Obrazovka poslední aktivity, včetně výběru možnosti pro sdílení a předvyplněného příspěvku na Facebook, je zobrazena na obrázku na následující straně (obr. 36). Následně jsou blíže rozebrány odpovídající části zdrojového kódu implementačního souboru `LastActivityViewController.m`.

Obr. 36: Obrazovka modulu "Last activity", sdílení aktivity

Načtení uložených informací o poslední aktivitě do polí pomocí třídy *NSUserDefaults* a klíčových slov je velmi jednoduché a probíhá prostřednictvím metody `viewDidLoad` automaticky. Jako příklad je zde uvedeno načtení informace o maximální a průměrné TF poslední aktivity.

```
- (void)viewDidLoad {
    [super viewDidLoad];

    //Načtení dat do labelů
    self.MaxHRLabel.text = [[NSUserDefaults standardUserDefaults]
objectForKey:@"maximTF"];
    self.AvgHRLabel.text = [[NSUserDefaults standardUserDefaults]
objectForKey:@"prumTF"];…
```

Implementace možností pro sdílení je realizována v rámci metody `ShareBT`, která je volána po stisku tlačítka v navigační liště. Prostřednictvím třídy *UIActivityViewController* jsou uživateli automaticky nabídnuty možnosti pro sdílení. Po výběru jedné z možností se následně automaticky vyplní text uložený jako `NSString`.

```
- (IBAction)ShareBT:(id)sender {
  NSString *shareText = [NSString stringWithFormat:@"I just did a
workout with •SportsBrain• App\n\nDate: %@\nDistance: %@\nTime:
%@\nPace: %@\nMax HR: %@\nAvg HR: %@\nMaxRR: %@\nAvgRR: %@",
self.dateLabel.text, self.DistanceLabel.text,self.TimeLabel.text,
self.PaceLabel.text,
self.MaxHRLabel.text,self.AvgHRLabel.text,self.MaxRRLabel.text,self
.AvgRRLabel.text];
  NSArray *itemsToShare = @[shareText];
  UIActivityViewController *activitzVC = [[UIActivityViewController
alloc]initWithActivityItems:itemsToShare
applicationActivities:nil];
  activitzVC.excludedActivityTypes = @[];
  [self presentViewController:activitzVC animated:YES
completion:nil];}
```

4.3.9 Modul "Fitness test"

Modul "*Fitness test*" ("Fitness test") slouží uživateli ke stanovení individuální TF_{max} pro běžecký a cyklistický trénink. Jak již bylo uvedeno, právě ze znalosti TF_{max} uživatele je aplikace schopna vypočíst pásma intenzity tréninku. Následně na základě měření aktuální hodnoty TF také zobrazit dané pásmo na obrazovce průběhu aktivity. Hodnota stanovená pomocí fitness testu je považována za přesnější, než hodnota vypočtená na základě vztahu (2).

Cílem modulu je umožnit uživateli dosažení maximální TF v rámci zvolené činnosti a tuto TF_{max} změřit. V literatuře je popsáno mnoho testů pro stanovení individuální TF_{max}. Aplikace uživatele nenutí využívat jeden specifický test. Po zahájení testu pouze zobrazuje aktuální hodnotu parametru TF a průběžně stanovuje maximální naměřenou hodnotu. Uživatel tak může využít například testu uvedeného v této práci v kapitole 2.3.1. Úvodní obrazovka modulu společně s obrazovkou průběhu a vyhodnocení testu je na obrázku (obr. 37).

Obr. 37: Obrazovky modulu "Fitness test"

Uživatel má na výběr dvojici podporovaných aktivit. Postup použití senzoru TF je v tomto případě stejný jako u modulů pro záznam běžeckého a cyklistického tréninku. Nejdříve je třeba navázat spojení s mikroprocesorem pomocí tlačítka *"Connect"*, umístěného v navigační liště. Před připojením k MCU není možné test zahájit. Aplikace po navázání Bluetooth komunikace s MCU přijímá informaci o aktuální TF stejným způsobem, jaký byl popsán v kapitole 4.3.6.

Následně je možné test zahájit stiskem tlačítka *"Start Running test"*, případně *"Start Cycling test"*. Průběh testů ze strany aplikace je v obou případech totožný.

V průběhu testu je uživateli zobrazena aktuální měřená hodnota TF. Ta je ukládána do pole typu *NSMutableArray* stejně, jako v kapitole 4.3.6. Také zde je z pole průběžně určována maximální hodnota. Tato úvodní obrazovka modulu, kde probíhá propojení k MCU, zahájení testu a zobrazení aktuální TF spolu s výpočtem TF_{max}, je řízena soubory `TestViewController.h` a `m`. Test je možné ukončit tlačítkem *"Stop and evaluate"*. Naměřená hodnota TF_{max} je následně pomocí třídy *NSUserDefaults* přenesena na obrazovku vyhodnocení testu. Zde jsou z hodnoty TF_{max} vypočtena pásma intenzity tréninku (viz kap. 4.3.6). Výsledná pásma jsou uživateli zobrazena v tabulce (obr. 37). Obrazovka výsledku testu je řízena souborem `TestResultRunViewController.h` a `m`. (pro test TF_{max} běhu) a `TestResultBikeViewController.h` a `m` (pro test TF_{max} cyklistiky). Řídící zdrojový kód je totožný s kódem popsaným v kapitole 4.3.2. Proto zde uveden nebude. Je však dostupný včetně komentářů na přiloženém DVD.

Je-li uživatel spokojen s naměřenými daty, má možnost svoji TF_{max} pro danou aktivitu uložit tlačítkem *"save"* v navigační liště. Následně je hodnota TF_{max} uložena pomocí třídy *NSUserDefaults* a klíčového slova `"maxHRuserbeh"`, či `"maxHRuserkolo"`. Takto je hodnota přístupná v rámci celé aplikace a lze ji využít i v rámci modulů *"New run"* a *"New bike"* pro zobrazení aktuálního pásma intenzity tréninku (viz kap. 4.3.6)

4.3.10 Modul "My profile"

Modul "*Můj profil*" ("My profile") umožňuje uživateli zadání osobních informací. Umožňuje vložit fotografii, jméno, věk a výšku uživatele. Ze zadaných dat následně aplikace využívá věk pro stanovení TF_{max}, výpočtem podle vztahu (2). Pásma intenzity tréninku stanovená na základě této hodnoty je možné v rámci modulu také zobrazit. Obrazovka modulu "My profile" je na následujícím obrázku (obr. 38).

Obr. 38: Obrazovka modulu "My profile"

Úvodní obrazovka modulu je řízena soubory `ProfileViewController.h` a `.m`. Po zadání údajů do příslušných polí je nutné data uložit pomocí tlačítka *"Save"* v navigační liště. Data jsou uložena pomocí třídy *NSUserDefaults* použitím příslušných klíčů stejně, jako již bylo v této práci mnohokrát popsáno. V rámci tohoto modulu je podstatný především výpočet TF_{max} použitím vztahu (2). K výpočtu dochází až následně po zadání a uložení věku uživatele a stisknutí tlačítka *"Calculate"*. Aplikace nabízí zvlášť výpočet hodnoty TF_{max} pro běh a pro cyklistiku. Stiskem tlačítka je volána metoda `CalculateMaxHR`, či `CalculateMaxHRKolo`. Zde je uveden zdrojový kód jen jedné z metod jelikož jsou totožné.

```
// Tlačítko pro výpočet maxHR BĚHU podle vzorce
- (IBAction)CalculateMaxHR:(id)sender {
  int intvek = [vek.text intValue];
  int personalHRmax = 220 - intvek;
  self.maxHRlabel.text = [NSString stringWithFormat:@"%d
bpm",personalHRmax];
}
```

V metodě nejdříve dochází k uložení věku uživatele do proměnné `intvek` z pole, kam byla již hodnota věku uživatelem zadána. Následně je do proměnné `personalHRmax` uložena hodnota TF_{max}, vypočtená podle vztahu (2) a výsledek vypsán do příslušného pole na obrazovce.

Uživatel má nyní možnost vypočtenou hodnotu uložit stiskem tlačítka *"Save"* v navigační liště. Vypočtená hodnota TF_{max} je pak uložena pomocí třídy *NSUserDefaults* a klíčového slova `"maxHRuserbeh"`, či `"maxHRuserkolo"`. Takto je hodnota přístupná v rámci celé aplikace a lze ji využít i v rámci modulů *"New run"* a *"New bike"* pro zobrazení aktuálního pásma intenzity tréninku. Je zřejmé, že tímto dojde k přepsání TF_{max}, kterou si případně uživatel již dříve stanovil v modulu *"Fitness test"*. V modulu "*My profile*" je proto vždy uvedena hodnota TF_{max}, kterou aplikace bere v daném okamžiku jako aktuální a podle které se řídí stanovení pásem intenzity tréninku. Určí-li tak uživatel svoji TF_{max} pomocí modulu "Fitness test", je v modulu "My profile" zobrazena právě tato naměřená hodnota. Nechá-li uživatel aplikaci následně vypočíst hodnotu TF_{max} podle vztahu (2), zobrazená hodnota je v modulu "My profile" přepsána.

Stanovená pásma je možné zobrazit stiskem tlačítka *"Zrun"* či *"Zbike"*. Následně je zobrazena tabulka obsahující jednotlivá pásma stejně, jako po provedení fitness testu (obr. 38). Soubory, které toto zobrazení řídí, jsou `ProfileRunViewController.h` a `m.` (pro TF_{max} běhu) a `ProfileBikeViewController.h` a `m.` (pro TF_{max} cyklistiky). Řídící zdrojový kód je založen na již popsaném a v kapitole 4.3.2 uvedeném zdrojovém kódu. Proto zde uveden nebude. Je však dostupný včetně komentářů na přiloženém DVD.

5. Ověření funkce systému

Přesnost měření biologických parametrů systémem SportsBrain byla ověřena nejprve dvojicí samostatných testů. Následně byl proveden test v laboratoři sportovní medicíny. Profesionální zařízení laboratoře umožnilo současný záznam a vyhodnocení parametru TF a DF. Dosažené výsledky testů jsou uvedeny v této kapitole.

5.1 Ověření detekce tepové frekvence

Přesnost stanovení tepové frekvence systémem SportsBrain byla ověřena pomocí sporttesteru Polar S610j. Jedná se o sporttester v podobě hodinek, který umožňuje příjem signálů hrudního pásu na frekvenci 5,5 kHz. Tedy stejného, který používá systém SportsBrain. Z detekovaného signálu sporttester stanovuje aktuální, průměrnou a maximální TF v průběhu tělesné zátěže. Ověření funkčnosti systému bylo provedeno záznamem TF v rámci jednoho intervalu běžeckého tréninku a to pomocí uvedeného sporttesteru a současně pomocí systému SportsBrain. Dosažené výsledky vyhodnocení aktivity jsou uvedeny v tabulce (tab. 11) a na obrázku (obr. 39)

Tab. 11: Porovnání výsledků vyhodnocení jednoho intervalu běžeckého tréninku

Vyhodnocení intervalu běžeckého tréninku		
	Polar S610j	SportsBrain
Délka záznamu [min:sek]	02:04	02:02
Max TF [bpm]	141	143
Avg TF [bpm]	124	115

Obr. 39: Porovnání výsledků vyhodnocení intervalu běžeckého tréninku

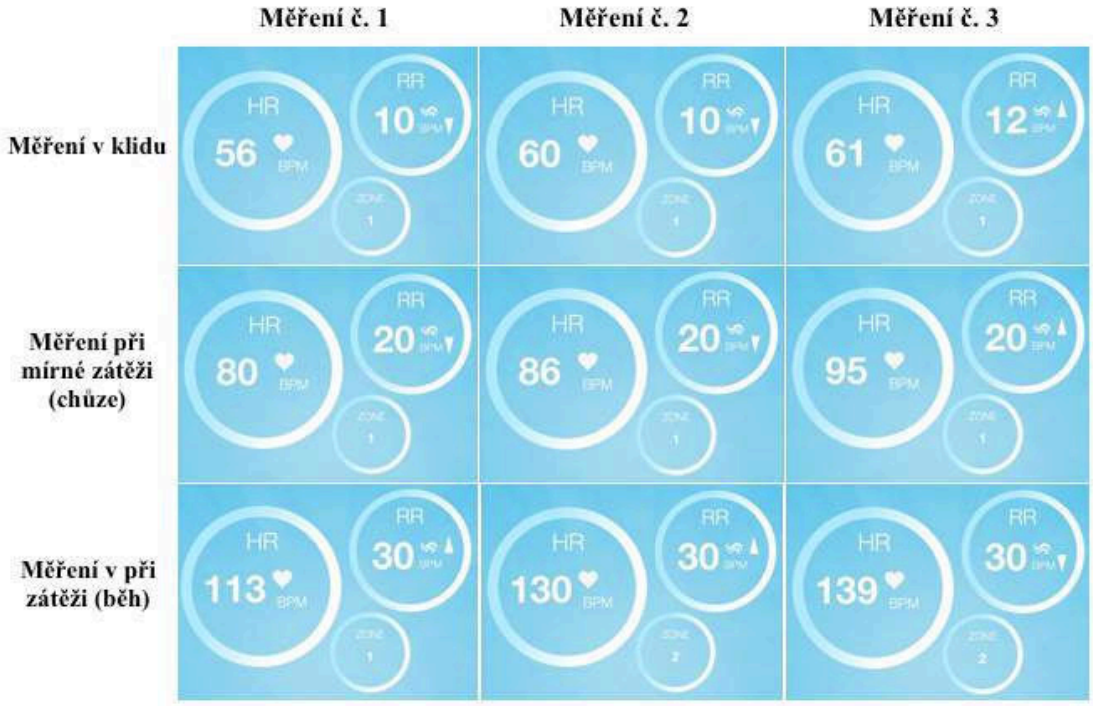
Z naměřených dat tepové frekvence lze soudit, že systém SportsBrain poskytuje v porovnání s profesionálním sporttesterem dobré výsledky. Naměřená hodnota maximální TF se liší pouze o 2 údery za minutu. Průměrná hodnota TF se pro uvedený záznam liší o 9 úderů za minutu. V průběhu záznamu se hodnoty zobrazované sporttesterem a hodnoty udávané mobilní aplikací lišily průměrně o 3 až 4 údery za minutu. U sporttesteru také nedocházelo k tak rychlým nárůstům a poklesům hodnot TF. Algoritmus sporttesteru pro výpočet a zobrazení TF pravděpodobně využívá většího vyhlazení zobrazovaných dat. Způsob snímání a přenos informací o srdeční aktivitě je u sporttesteru i systému SportsBrain stejný. Malá rozdílnost zobrazovaných údajů tak může být způsobena metodou použitou pro výpočet a vyhlazení hodnot TF.

5.2 Ověření detekce dechové frekvence

Správnost stanovení dechové frekvence pomocí systému SportsBrain byla srovnána s hodnotami, zjištěnými pomocí měření stopkami. Test zahrnoval trojici opakovaných měření pro trojici úrovní fyzické zátěže. Nejdříve byla provedena trojice měření v klidu ve stoji. Následně byla DF měřena při chůzi rychlostí 5 km/h na běžeckém pásu. Na závěr bylo měření provedeno v průběhu běžecké zátěže při rychlosti pásu 15 km/h. Celkem bylo tedy provedeno 9 měření. Pomocí stopek byl stanoven čas, za který proběhne 10 nádechů. Převrácená hodnota byla vynásobena nejdříve 60 sekundami a následně 10 dechy. Výsledky stanovené výpočtem společně s výsledky ze systému SportsBrain jsou uvedeny v tabulce (tab. 12). Průběh měření na systému Sportsbrain je uveden na obrázku (obr. 40).

Tab. 12: Výsledky stanovení dechové frekvence pomocí stopek a pomocí systému SportsBrain

DF v klidu			
Měření	Čas	DF stanovená výpočtem [dech/min]	DF stanovená systémem Sportsbrain [dech/min]
1	59,71	10	10
2	58,86	10	10
3	52,44	11	12
DF při mírné zátěž (chůze)			
Měření	Čas	DF stanovená výpočtem [dech/min]	DF stanovená systémem Sportsbrain [dech/min]
1	29,31	20	20
2	27,39	22	20
3	25,72	23	20
DF při zátěži (běh)			
Měření	Čas	DF stanovená výpočtem [dech/min]	DF stanovená systémem Sportsbrain [dech/min]
1	21,67	28	30
2	21,35	28	30
3	18,99	32	30



Obr. 40: Výsledky testu měření DF systému SportsBrain

Funkčnost stanovení DF systémem SportsBrain byla ověřena pro tři úrovně fyzické zátěže. V klidu se hodnoty stanovené systémem a hodnoty stanovené pomocí stopek téměř nelišily. Také při mírné fyzické zátěži, která byla navozena chůzí testované osoby rychlostí 5 km/h, dosahovaly výsledky téměř totožných hodnot s rozdílem maximálně 3 dechy za minutu. Stejné odchylky bylo dosaženo také při běhu rychlostí 15 km/h. Na obrázku (obr. 40) jsou znázorněny výsledky testu z mobilní aplikace. Úroveň fyzické zátěže v průběhu testu je zřejmá také z parametru TF. Rozdílnost parametru TF v rámci stanovené úrovně zátěže byla způsobena nemožností dosáhnout optimální ergostázy režimu a tím parametru TF probanda.

Na základě provedení testu lze konstatovat, že hodnoty stanovené systémem SportsBrain a hodnoty zjištěné stopkami, jsou téměř totožné. Metoda výpočtu DF pomocí stopek není dostatečně přesná a poskytuje pouze orientační stanovení DF. Chyba mohla být způsobena například špatným zastavením stopek na začátku posledního nádechu. Pro orientační kontrolu správnosti stanovení DF systémem SportsBrain je však tato metoda dostačující.

5.3 Ověření systému v laboratoři sportovní medicíny

Nejpřesnější biologická data o průběhu sportovní zátěže je možné získat ve specializované laboratoři sportovní medicíny. Systém SportsBrain byl otestován v laboratoři Fakulty sportovních studií Masarykovy univerzity. Výsledky měření fyzické zátěže získané systémem SportsBrain zde byly porovnány s výsledky stanovenými pomocí ergospirometrického systému METALYZER 3B-R2, který umožňuje současné snímání parametru tepové a dechové frekvence při definované fyzické zátěži.

5.3.1 Způsob provedení a výsledky testu

Test byl proveden pro tři úrovně tělesné zátěže. Nejdříve byl uskutečněn minutový záznam parametrů TF a DF v klidu ve stoji. Následoval minutový záznam sledovaných parametrů při chůzi na běžeckém pásu rychlostí 5 km/h (nastaveno na konzoli běžeckého pásu). Pro navození vyšší úrovně fyzické zátěže byla poté rychlost běžeckého pásu zvýšena na hodnotu 15 km/h po dobu 3 minut. Následně byl proveden minutový záznam sledovaných biologických parametrů.

Výsledky měření systémem METALYZER a systémem SportsBrain se v některých případech mohou lišit. Zatímco systém SportsBrain stanovuje parametr TF a DF každou sekundu, systém METALYZER vždy jednou za 10 sekund. Vyskytne-li se tak u sledované osoby na přechodný okamžik hodnota TF či DF výrazně vyšší či výrazně nižší než jsou ostatní hodnoty, systém METALYZER ji nemusí detekovat. Proto se mohou parametry průměrné a maximální TF a DF v porovnaných měřeních lišit.

Test v klidu

Výsledky naměřené systémem METALYZER 3B-R2 a vyhodnocené programem MetaSoft Studio společně s výsledky určenými systémem SportsBrain, jsou uvedeny v tabulce (tab. 13). Výsledek tak, jak jej poskytla mobilní aplikace, je na obrázku č. 41.

Tab. 13: Výsledky testu v laboratoři sportovní medicíny v klidu

<table>
<tr><th colspan="5">METALYZER 3B-R2</th><th colspan="2">SportsBrain</th></tr>
<tr><th rowspan="2">Čas</th><th>TF
[bpm]</th><th>DF
[dech/min]</th><th>Avg TF
[bpm]</th><th>Avg DF
[dech/min]</th><th>Avg TF
[bpm]</th><th>Avg DF
[dech/min]</th></tr>
<tr><td>0:00:10</td><td>72</td><td>14</td><td rowspan="2">76</td><td rowspan="2">14</td><td rowspan="2">67</td><td rowspan="2">12</td></tr>
<tr><td>0:00:20</td><td>75</td><td>15</td></tr>
<tr><td>0:00:30</td><td>76</td><td>13</td><td>Max TF
[bpm]</td><td>Max DF
[dech/min]</td><td>Max TF
[bpm]</td><td>Max DF
[dech/min]</td></tr>
<tr><td>0:00:40</td><td>77</td><td>14</td><td rowspan="2">78</td><td rowspan="2">15</td><td rowspan="2">77</td><td rowspan="2">15</td></tr>
<tr><td>0:00:50</td><td>78</td><td>13</td></tr>
<tr><td>0:01:00</td><td>78</td><td>15</td><td></td><td></td><td></td><td></td></tr>
</table>

Obr. 41: Výsledek měření v mobilní aplikaci

Výsledky testu v klidu ve stoji uvádějí mírné rozdíly především u měření hodnot tepové frekvence. Průměrná TF stanovená systémem METALYZER je v porovnání se stejnou hodnotou, kterou stanovil systém SportsBrain, rozdílná o 9 úderů za minutu. Maximální detekovaná hodnota TF je téměř totožná s rozdílem 1 úder za minutu. Hodnoty maximální DF jsou u obou systémů totožné. Hodnoty průměrné DF se liší o 2 dechy za minutu.

Test při chůzi

Výsledky naměřené systémem METALYZER 3B-R2 a vyhodnocené programem MetaSoft Studio, společně s výsledky určenými systémem SportsBrain, jsou uvedeny v tabulce (tab. 14). Výsledek tak, jak jej poskytla mobilní aplikace, je na obrázku č. 42.

Tab. 14: Výsledky testu v laboratoři sportovní medicíny při chůzi rychlostí 5 km/h

METALYZER 3B-R2					SportsBrain	
Čas	TF [bpm]	DF [dech/min]	Avg TF [bpm]	Avg DF [dech/min]	Avg TF [bpm]	Avg DF [dech/min]
0:00:10	66	13	69	13	70	14
0:00:20	70	16				
0:00:30	69	12	Max TF [bpm]	Max DF [dech/min]	Max TF [bpm]	Max DF [dech/min]
0:00:40	70	12				
0:00:50	71	13	71	16	88	20
0:01:00	69	12				

Obr. 42: Výsledek měření v mobilní aplikaci

Průměrná TF stanovená systémem METALYZER při chůzi je v porovnání se stejnou hodnotou, kterou stanovil systém SportsBrain, rozdílná o 1 úder za minutu. Maximální detekovaná hodnota TF se zde liší o 17 úderů za minutu. Hodnota průměrné DF je odlišná o 1 dech za minutu. Hodnota maximální DF se liší o 4 dechy za minutu.

Test při běhu

Výsledky naměřené systémem METALYZER 3B-R2 a vyhodnocené programem MetaSoft Studio, společně s výsledky určenými systémem SportsBrain, jsou uvedeny v tabulce (tab. 15). Výsledek tak, jak jej poskytla mobilní aplikace, je na obrázku č. 43.

Tab. 15: Výsledky testu v laboratoři sportovní medicíny při běhu rychlostí 15 km/h

METALYZER 3B-R2					SportsBrain	
Čas	TF [bpm]	DF [dech/min]	Avg TF [bpm]	Avg DF [dech/min]	Avg TF [bpm]	Avg DF [dech/min]
0:00:10	149	32	149	32	125	28
0:00:20	136	28				
0:00:30	149	33	Max TF [bpm]	Max DF [dech/min]	Max TF [bpm]	Max DF [dech/min]
0:00:40	152	32				
0:00:50	153	32	154	33	155	30
0:01:00	154	33				

Obr. 43: Výsledek měření v mobilní aplikaci

Průměrná TF stanovená systémem METALYZER při běhu je v porovnání se stejnou hodnotou, kterou stanovil systém SportsBrain, rozdílná o 25 úderů za minutu. Maximální detekovaná hodnota TF se zde liší o 1 úder za minutu. Hodnota průměrné DF je odlišná o 4 dechy za minutu. Hodnota maximální DF se liší o 3 dechy za minutu.

5.3.2 Vyhodnocení testu.

V porovnání s profesionálním spiroergometrem nedosahuje systém SportsBrain vždy totožných výsledků. Stanovená průměrná hodnota TF se liší nejvíce o 25 úderů za minutu a to při záznamu v průběhu běhu. Maximální TF pak nejvíce o 17 úderů za minutu v průběhu chůze. Zde se zřejmě projevuje větší nekonzistentnost snímaných dat, kterou by bylo možné u systému SportsBrain odstranit lepším vyhlazením vypočtené TF. Systém METALYZER však stanovoval údaje o TF a DF vždy jednou za 10 s, zatímco Systém SportsBrain každou sekundu. I toto může být důvod odchylky měřené maximální a průměrné TF. Provedený test pomocí systému METALYZER proto poskytuje spíše

ověření správnosti detekce dechové frekvence, kde se rozdíl ve vzorkování tolik neprojeví. Důvodem je menší počet dechových cyklů za minutu, než je počet srdečních kontrakcí. Naproti tomu měření parametru dechové frekvence bylo velmi podobné. Rozdíl v měřené DF nebyl větší, než 4 dechy za minutu. O správnosti stanovení parametru TF lépe vypovídá test provedený v kapitole 5.1. Sporttester ve formě hodinek snímá a zobrazuje TF každých 5 sekund.

Systém SportsBrain nedosahuje v přesnosti kvalit profesionálního vybavení laboratoře sportovní medicíny. I tak poskytuje uspokojivě přesná data, související s fyzickou zátěží. Přesnost a spolehlivost navrženého systému je blízká standardním a volně prodávaným sporttesterům, které umožňují záznam tepové frekvence. Navržený systém SportsBrain poskytuje navíc poměrně přesné sledování a vyhodnocení parametru dechové frekvence.

5.4 Praktické využití systému SporstBrain

Navržený systém pro monitorování sportovní aktivity byl otestován a je plně funkční. Oproti standardním systémům, které využívají hrudního pásu a mobilní aplikace pro záznam a vyhodnocení sportovní aktivity, umožňuje navržený systém navíc monitorování dechové frekvence. Monitorování DF bylo umožněno integrací senzoru ohybu do elastické části hrudního pásu. Nabízí se tak možnost výrobcům hrudních pásů (např. společnosti Polar) integrace podobných senzorů při vývoji nových zařízení. Parametr dechové frekvence je dnes neprávem opomíjen a může pro výrobce znamenat zajímavou výhodu oproti velké konkurenci, která dnes na poli fitness a sportovních systémů existuje. Mikroprocesorovou část systému SportsBrain lze integrovat například do centrální části hrudního pásu. Nový typ pásu tak může obsahovat standardní detekci elektrické srdeční aktivity, navíc doplněnou o detekci dechových pohybů. Hrudní pás lze také vybavit systémem Bluetooth 4.0 pro přenos dat přímo do mobilní aplikace.

Navržený systém lze využít v oblasti sportovního tréninku pro záznam a vyhodnocení běhu a cyklistiky. Zobrazovaný parametr TF společně se stanovením pásma intenzity tréninku umožňuje uživateli trénink v požadovaném pásmu, čímž dochází k rozvoji specifické složky tělesné zdatnosti. Navíc je uživateli zobrazen parametr dechové frekvence, který taktéž reflektuje aktuální intenzitu tělesné zátěže. Správná technika a způsob dýchání je důležitou složkou fyzického výkonu.

Celý systém SportsBrain byl navržen s použitím volně dostupných HW a SW komponent a není jen jednoúčelovým technickým řešením. Na návrhu systému lze dále pracovat a do HW části zařadit nové senzory pro možnost sledování dalších parametrů sportovního tréninku, jako například počet kroků či pocení. Integrace nových komponent a senzorů je umožněna díky využití mikroprocesoru otevřené vývojové platformy Arduino. Nové funkce lze do systému v jeho stávající podobě integrovat také úpravou řídícího SW, jak v rámci MCU, tak především programu mobilní aplikace SporsBrain.

Každý rok se na poli mobilních aplikací a operačních systémů setkáváme s novými možnostmi. Toto technologické odvětví se v současné době velmi rychle rozvíjí. Není tedy divu, že společnosti Apple, Google a Microsoft patří mezi nejhodnotnější společnosti světa a jejich příjmy se pohybují v astronomických částkách. Jedním z podstatných milníků byl vývoj a masové rozšíření počítačové techniky. Za druhý milník lze považovat nástup a rychlé rozšíření chytrých mobilních telefonů. V současné době stojíme na prahu třetího milníku, kterým je tzv. nositelná elektronika. Tato vysoce osobní zařízení budou v příštích několika letech znamenat to, co znamenal nástup počítačů, či chytrých mobilních telefonů před několika lety. Nová zařízení však budou mít zásadní význam pro sběr a zpracování biologických dat. Společnost Apple v loňském roce představila platformu "Health", společnost Google platformu "Google Fit" a společnost Microsoft platformu "Microsoft Health". Představené platformy přinášejí komplexní řešení pro záznam, vyhodnocení a sdílení biologických dat. V současné době jsou testovány řadou prestižních lékařských zařízení (např. Mayo Clinic). V příštích několika letech se stanou běžným standardem v oblasti medicínských informačních technologií. Systém SportsBrain je připraven k integraci v rámci platformy "Health" společnosti Apple. V budoucnu lze systém také transformovat na další mobilní operační systémy, jako je Android a Windows Phone.

Spojení znalostí biomedicínského inženýrství a mobilních technologií je nastupujícím trendem. Systém SportsBrain je jednoznačnou reprezentací tohoto trendu. Na Fakultě elektrotechniky a komunikačních technologií VUT bohužel prozatím není studentům toto unikátní spojení nabízeno, například v rámci specializovaného volitelného předmětu.

Závěr

Diplomová práce se věnuje dostupným metodám měření parametrů tepové a dechové frekvence v oblasti fitness a sportovních systémů. Cílem práce bylo navrhnout zařízení umožňující monitorovat tepovou a dechovou frekvenci pomocí mobilního telefonu a systému Arduino. Pro splnění cíle bylo v literární rešerši žádoucí prostudovat metody aplikované při monitorování aktivit sportovců a seznámit se s odpovídajícími technickými řešeními a možnostmi jejich propojení s mobilním telefonem. Snímání elektrických projevů srdeční činnosti v průběhu sportovní aktivity bylo ověřeno dvojicí dostupných, a s platformou Arduino kompatibilních, technických řešení. Požadavkem na stanovení dechové frekvence byla detekce dýchání v reálném čase.

Navržený systém pro monitorování sportovní aktivity SportsBrain je složen z trojice funkčních prvků. Senzorové části, mikroprocesoru platformy Arduino a mobilního telefonu se speciální mobilní aplikací. Senzorovou část představuje hrudní pás společnosti Polar, který je připevněn okolo hrudníku uživatele. Zde detekuje a filtruje EKG křivku srdeční aktivity. Signál odpovídající stahům myokardu je následně přenášen a detekován v mikroprocesorové částí systému, kde dochází k výpočtu tepové frekvence. Dechová frekvence je stanovena měřením změny elektrického odporu ohybového senzoru, který byl integrován v elastické části hrudního pásu. Parametr dechové frekvence je dnešními systémy, které využívají mobilní aplikace, opomíjen. Pro snadný přenos měřených biologických parametrů do mobilního telefonu iPhone je základem mikroprocesorové části systému vývojová deska Blend Micro. Tato vývojová deska dosahuje velmi malých rozměrů a senzorovou část systému tak lze připevnit přímo na hrudní pás. Důležitou součástí řešení je mobilní aplikace SportsBrain, která přijímá a zobrazuje měřené biologické parametry. Využitím senzoru GPS s funkcí A-GPS je aplikace schopna stanovit aktuální i průměrné tempo sportovní aktivity, dále pak celkovou vzdálenost, rychlost a čas. V rámci vyhodnocení aktivity je uživateli zobrazena trasa tréninku v mapových podkladech a to včetně rychlostního profilu tratě. Aplikace dále nabízí možnost provedení testu maximální tepové frekvence, sdílení výsledků sportovní aktivity či vytvoření individuálního profilu uživatele.

Navržený systém prezentuje možnosti volně dostupných HW a SW technologií a jejich implementaci za účelem vytvoření plnohodnotného nástroje pro monitorování sportovní aktivity. Stanovených cílů bylo dosaženo a předložená práce splňuje zadání v plném rozsahu.

Literatura

[1] AD8232 Heart Rate Monitor Hookup Guide. *Sparkfun* [online]. 2014 [cit. 2015-05-17]. Dostupné z: https://learn.sparkfun.com/tutorials/ad8232-heart-rate-monitor-hookup-guide

[2] ALLAN, Alasdair. *IOS sensor apps with Arduino*. Beijing: O'Reilly, 2011. ISBN 978-144-9308-483.

[3] Apple Watch Sport. *Apple* [online]. 2015 [cit. 2015-05-17]. Dostupné z: https://www.apple.com/watch/apple-watch-sport/

[4] Arduino Uno R3. *Australian Robotics* [online]. 2014 [cit. 2015-05-17]. Dostupné z: http://www.australianrobotics.com.au/products/arduino-uno-r3

[5] BENSON, Roy a Declan CONNOLLY. *Trénink podle srdeční frekvence: jak zvýšit kondici, vytrvalost, laktátový práh, výkon*. 1. vyd. Praha: Grada, 2012, 184 s. ISBN 978-80-247-4036-2.

[6] Blend Micro. *RedBearLab* [online]. 2014 [cit. 2015-05-17]. Dostupné z: http://redbearlab.com/blendmicro

[7] E-Health Sensor Platform V2.0 for Arduino and Raspberry Pi [Biometric / Medical Applications]. *Cooking hacks: tasty electronics* [online]. 2014 [cit. 2015-04-26]. Dostupné z: http://www.cooking-hacks.com/documentation/tutorials/ehealth-biometric-sensor-platform-arduino-raspberry-pi-medical

[8] Flex Sensor 2.2. *Sparkfun* [online]. 2014 [cit. 2015-04-26]. Dostupné z: https://www.sparkfun.com/products/10264

[9] How To Make an App Like RunKeeper: Part 1. LUEDKE, Matt. *RayWenderlich: Tutorials for developers & gamers* [online]. 2014 [cit. 2015-05-10]. Dostupné z: http://www.raywenderlich.com/73984/make-app-like-runkeeper-part-1

[10] HRDINA, Zdeněk. *Rádiové určování polohy: Družicový systém GPS*. 1. vyd. Praha: Vydavatelství ČVUT, 1999, 259 s. ISBN 80-010-1386-3.

[11] IPhone 4s Tech Specs. *Apple* [online]. 2015 [cit. 2015-05-17]. Dostupné z: https://www.apple.com/lae/iphone-4s/specs/

[12] JAVORKA, Kamil. *Variabilita frekvencie srdca: mechanizmy, hodnotenie, klinické využitie*. Martin: Osveta, 2008, 204 s. ISBN 978-80-8063-269-4.

[13] KOCHAN, Stephen G. *Objective-C 2.0: výukový kurz programování pro Mac OS X a iPhone*. Vyd. 1. Brno: Computer Press, 2010, 550 s. ISBN 978-80-251-2654-7.

[14] KOLÁŘ, Radim. FEKT VUT. *Lékařská diagnostická technika* [online]. Brno, 2006 [cit. 2015-05-17]. Dostupné z: http://www.unium.cz/materialy/vut/fekt/skripta-m37351-p1.html.

[15] MARK, Dave a Jeff LAMARCHE. *IPhone SDK: průvodce vývojem aplikací pro iPhone a iPod touch*. Vyd. 1. Brno: Computer Press, 2010, 480 s. ISBN 978-80-251-2820-6.

[16] MOUREK, Jindřich. *Fyziologie: učebnice pro studenty zdravotnických oborů*. 1. vyd. Praha: Grada, 2005, 208 s. ISBN 80-247-1190-7.

[17] NEUMANN, Georg, Arndt PFÜTZNER a Kuno HOTTENROTT. *Trénink pod kontrolou: metody, kontrola a vyhodnocení vytrvalostního tréninku*. 1. vyd. Praha: Grada, 2005, 181 s. ISBN 80-247-0947-3.

[18] PALEČEK, František. *Patofyziologie dýchání*. 1. vyd. Praha: Karolinum, 2001, 123 s. Učební texty (Univerzita Karlova). ISBN 80-246-0231-8.

[19] PLACHETA, Zdeněk, Jarmila SIEGELOVÁ a Miloš ŠTEJFA. *Zátěžová diagnostika v ambulantní a klinické praxi*. 1. vyd. Praha: Grada Publishing, 1999, 276 s. ISBN 80-716-9271-9.

[20] Pneumotrace Respiration Sensor (THOR). *Stoelting* [online]. 2014 [cit. 2015-05-17]. Dostupné z: http://access.stoeltingco.com/stoelting/2769/1466/1487/Polygraph/Pneumotrace-Respiration-Sensor-THOR

[21] POLAR ELECTRO INC-OEM DIVISION. *RMCM-01 Heart Rate Receiver Component* [online]. Lake Success, 2005 [cit. 2015-05-17]. Dostupné z: http://produceconsumerobot.com/heartfeltapparel/content/RMC M01.pdf

[22] Polar T61 Coded Transmitter and Belt Set. *Tri Now Fitness* [online]. 2014 [cit. 2015-05-17]. Dostupné z: http://www.trinowfitness.com/scripts/prodView.asp?idProduct=36

[23] Polymer Lithium Ion Battery - 110mAh. *PV Electronic: Prodej elektronických komponentů a modulů* [online]. 2015 [cit. 2015-05-04]. Dostupné z: http://www.pvelectronic.eu/napajeni-nabijeni/polymer-lithium-ion-battery-110mah.html

[24] *Pulse sensor-transmitter T31 Polar and receiver RMCM01* [online]. 2011 [cit. 2015-05-17]. Dostupné z: http://lex.iguw.tuwien.ac.at/toysrus/index.php/Pulse_sensor-transmitter_T31_Polar_and_receiver_RMCM01

[25] RESEARCH2GUIDANCE. *MHealth App Developer Economics 2014: The State of the Art of mHealth App Publishing* [online]. 2014, 43 s. [cit. 2015-05-17]. Dostupné z: http://mhealtheconomics.com

[26] ROZMAN, Jiří. *Elektronické přístroje v lékařství*. Vyd. 1. Praha: Academia, 2006, 406 s., xxiv s. barev. obr. příl. ISBN 80-200-1308-3.

[27] Serial Cable (C2-DB9V). *Redpark* [online]. 2014 [cit. 2015-05-17]. Dostupné z: http://redpark.com/serial-cable-115-2-kbps-c2-db9v/

[28] Smartphone OS Market Share, Q4 2014. *IDC* [online]. 2014 [cit. 2015-05-17]. Dostupné z: http://www.idc.com/prodserv/smartphone-os-market-share.jsp

Seznam symbolů, veličin a zkratek

A-GPS	Assisted GPS
AP	Anaerobní práh
Bps	bits per second
CNS	Centrální Nervový Systém
DF	Dechová frekvence
DF_{max}	Maximální dechová frekvence
EKG	Elekrokardiografický signál
GPS	Global Positioning System
GUI	Graphical User Interface
HRW	Heart Rate Variability
HW	Hardware
IDE	Integrated Development Environment
iOS	iPhone OS
LE	Low Energy
MCU	Jednočipový počítač, mikrokontrolér
MT	Mobilní telefon
MVC	Model-View-Controller
OS	Operační systém
PDA	Personal Digital Assistant
PWM	Pulse Width Modulation
RR	Respiratory rate
R-R	R-R interval signálu EKG
RSA	Respirační sinusová arytmie
SW	Software
TF	Tepová frekvence
TF_{max}	Maximální tepová frekvence
TTL	Transistor to Transistor Logic
UAMT	Universal Asynchronous Receiver Transmitter
VO_2	Spotřeba kyslíku
Wi-Fi	Wirelles LAN
WPS	Wi-Fi Positioning Service

Seznam příloh

A. Podíl mobilních aplikací v oblasti mHealth na trhu (převzato [25])

B. Propojení čidla RMCM-01 s vývojovou deskou Arduino (převzato [24])

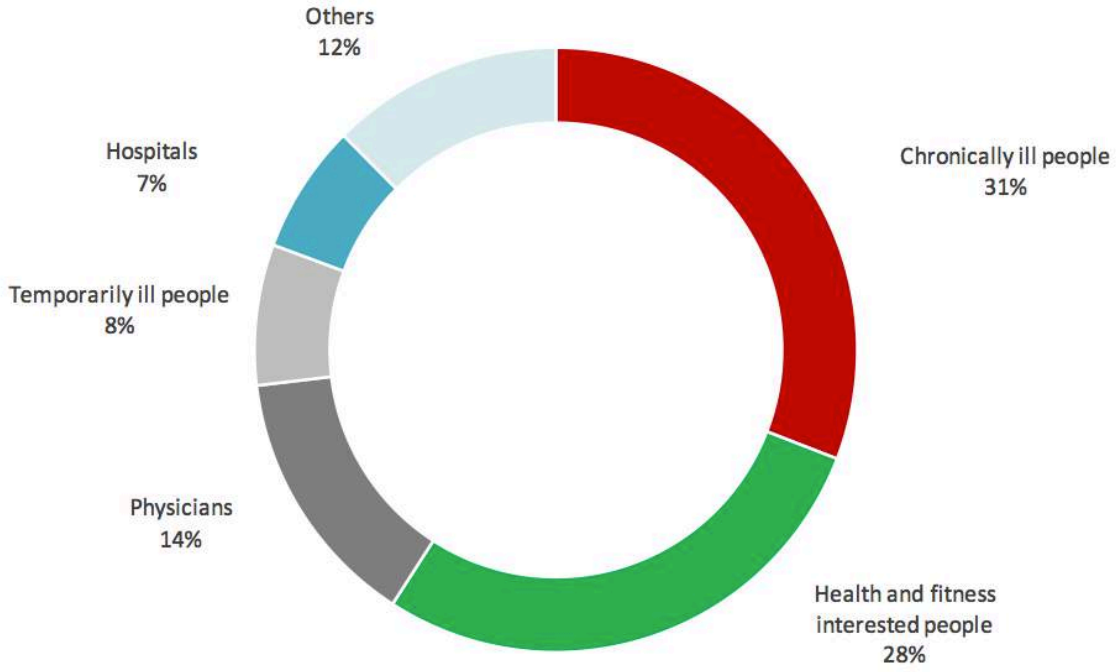
C. Propojení senzoru Flex 2.2 s vývojovou deskou Arduino (převzato [8])

D. Rozmístění pinů vývojové desky Blend Micro (převzato [6])

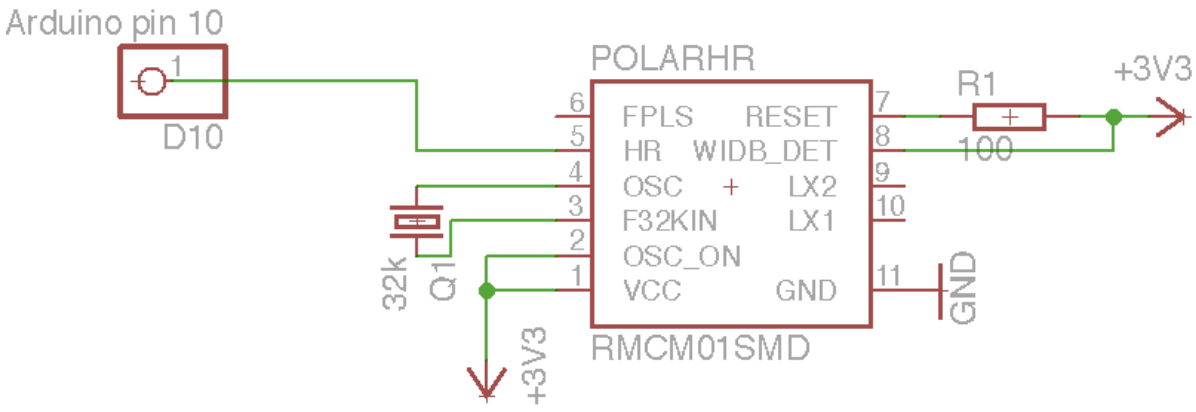
E. Seznam použitých modulů a součástek

F. Storyboard mobilní aplikace SportsBrain

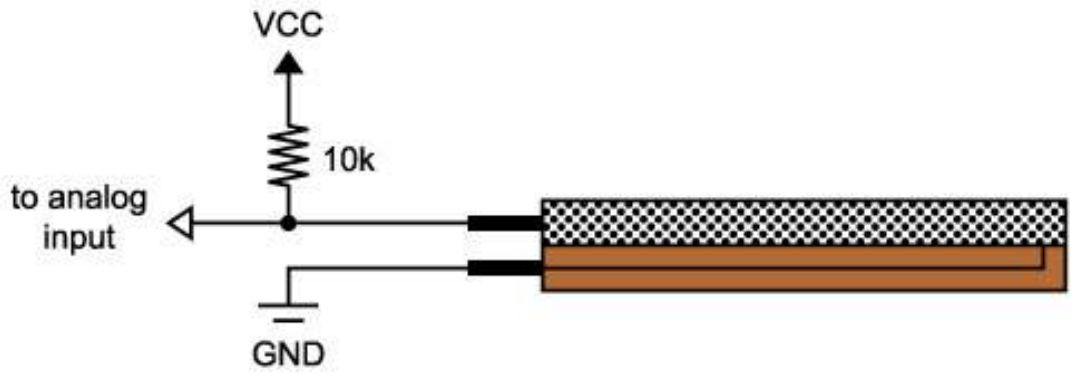
A. Podíl mobilních aplikací v oblasti mHealth



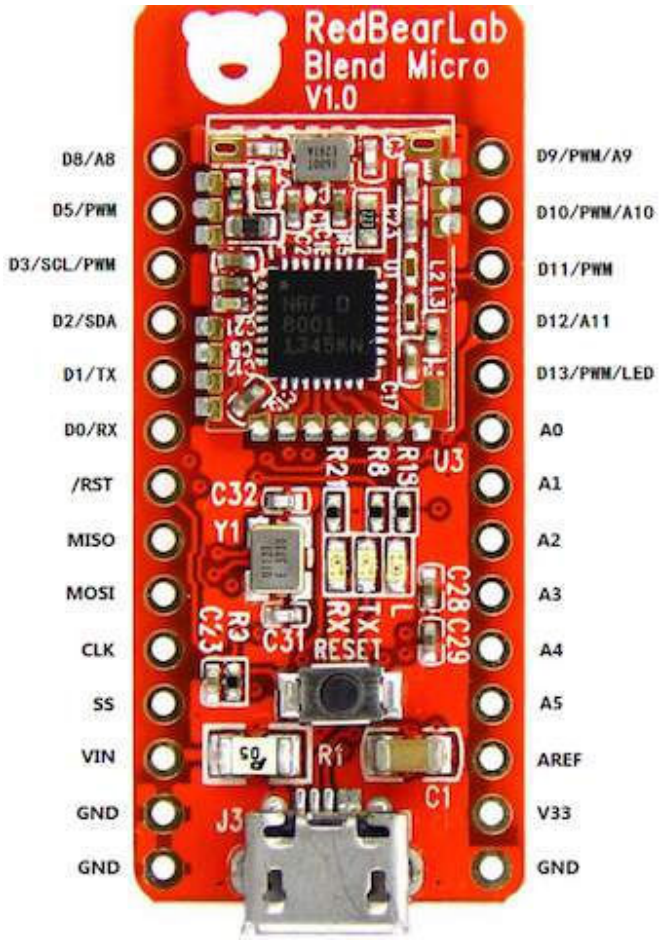
B. Propojení čidla RMCM-01 s vývojovou deskou Arduino



C. Propojení senzoru Flex 2.2 s vývojovou deskou Arduino



D. Rozmístění pinů vývojové desky Blend Micro



E. Seznam použitých modulů a součástek

Moduly		
Označení	Typ	Vlastnosti
Blend Micro	Blend Micro v1.0	Mikrokontroler
RMCM-01	Polar RMCM-01	Čidlo signálu hrudního pásu
Součástky		
Označení	Typ	Vlastnosti
Flex	Basic Flex Rezistor	Odporový detektor respiračních pohybů
LED1	Red(633 nm) LED	Dioda červené barvy 633 nm, pouzdro 0603 [SMD]
LED2	Blue(525 nm) LED	Dioda modré barvy 525 nm, pouzdro 0603 [SMD]
R1	1 kΩ rezistor	odpor 1 kΩ; odstup pinů 400 mil; bands 4; tolerance ± 5%; pouzdro THT
R2	100 kΩ rezistor	odpor 10 kΩ; odstup pinů 400 mil; bands 4; tolerance ± 5%; pouzdro THT
U1	LIPO-110 mAh	varianta 110 mAh; pouzdro lipo - 110
XTAL1	KRYSTAL - 32 kHz	krystal, kmitočet 32 kHz; odstup pinů 2.54 mm

F. Storyboard mobilní aplikace SportsBrain