VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

BEZDRÁTOVÉ OVLÁDÁNÍ ELEKTRONIKY MOBILNÍM/EMBEDDED ZAŘÍZENÍM

WIRELESS CONTROL OF ELECTRONIC DEVICES USING MOBILE/EMBEDDED DEVICE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **LUKÁŠ HEKRDLA**
AUTHOR

VEDOUCÍ PRÁCE **prof. Dr. Ing. PAVEL ZEMČÍK**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2019/2020

Zadání bakalářské práce

Student: **Hekrdla Lukáš**
Program: Informační technologie
Název: **Bezdrátové ovládání elektroniky mobilním/embedded zařízením**
Wireless Control of Electronic Devices Using Mobile/Embedded Device
Kategorie: Vestavěné systémy

Zadání:

1. Prostudujte dostupná řešení a literaturu na téma Bluetooth, případně WiFi a/nebo jiné moduly pro bezdrátové ovládání elektroniky se zaměřením na možnost využití mobilních nebo embedded zařízení.
2. Vytipujte vhodnou aplikaci vhodně demonstrující vlastnosti takových modulů, například to může být čtení hodnot čidla teploty, řízení elektrospotřebičů, nebo světel, ale i drobnosti, jako ovládání servomechanismů apod.
3. Navrhněte způsob implementace výše uvedené aplikace a popište dosažitelné vlastnosti řešení.
4. Implementujte vybranou aplikaci a její ovládání a také demonstrujte funkčnost a možnosti na vhodném příkladu.
5. Diskutujte dosažené výsledky a další možnosti pokračování práce.

Literatura:

- Dle pokynů vedoucího

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Zemčík Pavel, prof. Dr. Ing.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 28. května 2020
Datum schválení: 1. listopadu 2019

Abstrakt

Tato práce se zabývá návrhem systému, který pomocí Bluetooth technologie komunikuje s mobilním telefonem a primárně slouží k zasílání upozornění o příchozích hovorech a SMS zprávách. Systém sestává z LCD displeje, na kterém se upozornění zobrazují, bluetooth modulu, který zajišťuje komunikaci mezi displejem a mobilním telefonem a mobilní aplikace, která to vše řídí. Výsledný systém byl ověřen, že tyto funkce plní.

Abstract

This bachelor's thesis is focused on the design of a system that communicates with a mobile phone using Bluetooth technology and is primarily used to send notifications of incoming calls and SMS messages. The system consists of an LCD display on which alerts are displayed, a bluetooth module that provides communication between the display and the mobile phone and a mobile application that controls everything. The resulting system was verified to perform these functions.

Klíčová slova

Bluetooth, bluetooth modul, HC-05, LCD displej, Android, aplikace, android studio, Kotlin

Keywords

Bluetooth, bluetooth module, HC-05, LCD display, Android, application, android studio, Kotlin

Citace

HEKRDLA, Lukáš. *Bezdrátové ovládání elektroniky mobilním/ embedded zařízením.* Brno, 2020. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. Dr. Ing. Pavel Zemčík

Bezdrátové ovládání elektroniky mobilním/ embedded zařízením

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana prof. Dr. Ing. Pavla Zemčíka Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Lukáš Hekrdla
23. června 2020

Poděkování

V této části bych chtěl poděkovat vedoucímu bakalářské práce prof. Dr. Ing. Pavlovi Zemčíkovi za odbornou pomoc, cenné rady a nápady při zpracování této práce

Obsah

Seznam obrázků

Kapitola 1

Úvod

Tato práce se zabývá komunikací pomocí bezdrátových technologií. Může se jednat o bezdrátové sluchátka, hodinky komunikující s mobilem, apod. Jednou z těchto technologií je Bluetooth, které je tu s námi již přes 20 let. Původně se mělo jednat o bezdrátovou alternativu komunikace mezi počítačem a jeho periferiemi. Postupem času, zejména s rozšířením chytrých telefonů je snahou výrobců, aby naše zařízení ať už s počítačem nebo mobilem komunikovaly bezdrátově.

V minulosti veškerá komunikace mezi jednotlivými zařízeními probíhala výhradně pomocí kabelového spojení (např. počítač s tiskárnou, mobil se sluchátky, ...). Během posledních let se rozšířily bezdrátové technologie jako Wi-Fi nebo již zmíněné Bluetooth, které je dnes velmi oblíbené. Současně se stále více hovoří o nástupu sítí 5G, ale dle mého názoru bude trvat ještě několik let než se rozšíří v takovém množství jako Bluetooth nebo Wi-Fi.

Toto téma jsem si vybral, protože jeho součástí je i mobilní aplikace, kterou jsem zatím žádnou nevyvíjel. Je to tedy pro mě příležitost naučit se něco nového. A to z pohledu vývoje mobilních aplikací, tak pro mě dosud neznámého programovacího jazyka Kotlin.

Cílem bakalářské práce je navrhnout a realizovat takový systém, pomocí kterého bude možné na externí displej zobrazovat informace z mobilního telefonu. Bude se jednat především o upozornění na příchozí hovory a SMS zprávy, ale i třeba aktuální čas a datum nebo stav baterie mobilního telefonu. Výsledný systém se tedy bude skládat z LCD displeje, mobilní aplikace a Bluetooth modulu, který zajistí komunikaci mezi mobilním telefonem a displejem.

Následující kapitola se věnuje technologii Bluetooth. Třetí kapitola se zabývá technologickými moduly a ve čtvrté kapitole jsou představeny vývojové prostředky. V páté kapitole je zhodnocení současného stavu. Šestá kapitola se zabývá implementací a testováním a v sedmé kapitole je závěr.

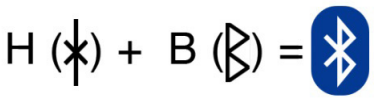
Kapitola 2

Bluetooth technologie

Tato kapitola se věnuje vzniku a vývoji technologie, principu komunikace, rozdělením do výkonových tříd a zabezpečení. Obsah kapitoly má bezprostřední vztah k práci, ale není zde uvedeno vše o technologii Bluetooth, protože by to bylo nad rámec této práce.

2.1 Vznik Bluetooth

Bluetooth je otevřený standard pro bezdrátovou komunikaci na krátkou vzdálenost, vyvinutý švédskou společností Ericsson v roce 1994. V roce 1998 vzniklo sdružení Bluetooth Special Interest Group (Bluetooth SIG), které má na starosti standardizaci a vývoj jednotlivých verzí Bluetooth technologie. Bluetooth SIG bylo založeno společnostmi Ericsson, Nokia, IBM, Toshiba a Intel[11]. Název vznikl podle dánského krále Haralda Gormssona, kterému se přezdívalo Blåtand, což v anglickém překladu znamená Bluetooth[9].



Obrázek 2.1: Vznik symbolu Bluetooth [5]

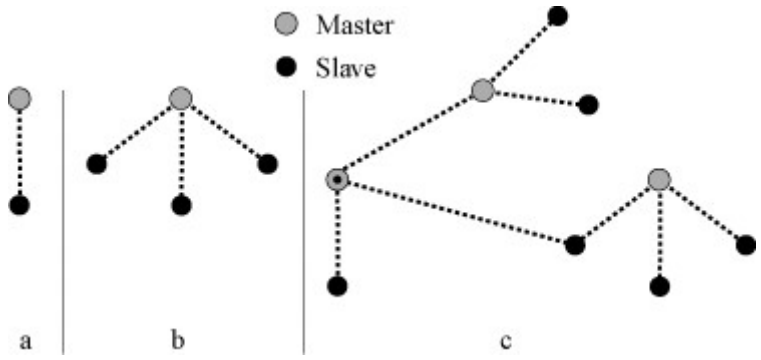
V roce 2016 dle slov ředitele Bluetooth SIG využívalo technologie Bluetooth cca 8,2 miliardy zařízení na celém světě, to jen pro představu o jak oblíbenou technologii se jedná[8].

2.2 Princip komunikace Bluetooth

Komunikace probíhá vysíláním rádiových vln v nelicencovaném ISM (Industrial Scientific Medicine) frekvenčním pásmu 2,4 GHz. Protože v tomto frekvenčním pásmu existují další bezdrátové technologie (Wi-Fi, ZigBee, ...), je toto pásmo silně zarušeno. Bluetooth proto implementuje funkci FHSS (Frequency-Hopping Spread Spectrum), pomocí které může přeskakovat mezi frekvencemi a vybrat si tak nejméně zarušenou frekvenci. V případě klasického BR/EDR má zařízení 79 kanálů s rozestupy 1 MHz a ve variantě Bluetooth Low Energy (BLE) má zařízení 40 kanálů s rozestupy 2 MHz. Za jednu sekundu lze takových skoků udělat až 1600[11].

2.3 Topologie Bluetooth

Komunikaci v rámci jedné sítě mohou navázat alespoň dvě zařízení, přičemž se jedná o spojení point-to-point. Maximální počet zařízení v jedné síti je potom osm, kdy jedno zařízení je označeno jako master a k němu se může připojit až sedm dalších zařízení, která jsou označena jako slave. Takto vzniklá PAN síť se nazývá piconet. Jednotlivé piconety lze dále sdružovat do tzv. scatternetu (obr. 2.2) a mohou tak vznikat rozsáhlejší sítě.[1]



Obrázek 2.2: Příklad piconet sítě s jedním slave zařízením (a), s více slave zařízeními (b) a scatternet sítě (c)[1]

V jedné rozprostřené síti může být maximálně 10 pikosítí[7].

2.4 Verze Bluetooth

Vzhledem k tomu, že Bluetooth technologie je tu s námi již přes 20 let, tak od jejího počátku prošla poměrně značným vývojem a dnes existuje v několika verzích (aktuálně 5.2). V každé nové verzi se objevila nějaká nová funkcionalita nebo došlo k vylepšení některé již existující.

- **Bluetooth 1.0**
  Jedná se o první výskyt této technologie. Tato verze měla spoustu počátečních problémů a tak nástup technologie nebyl tak vysoký jak se očekávalo[7]. Jednalo se o problémy s kompatibilitou, párováním, ...
- **Bluetooth 1.1**
  V této verzi byly opraveny chyby první verze a technologie byla schválena jako IEEE Standard 802.15.1. Dále byla přidána podpora pro nešifrované kanály a indikátor síly signálu[7].
- **Bluetooth 1.2**
  Tato verze přinesla rychlejší proces párování, byla zde implementována funkce FHSS pro přeskakování frekvencí a přenosová rychlost dosahovala 721 kbit/s [7].
- **Bluetooth 2.1 + EDR**
  Tato verze přináší především funkci jednoduchého a bezpečného párování, kdy je při párování povinné šifrování[9]. Dále byla zvýšena přenosová rychlost, která dosahuje 2,1 Mbit/s a až 3 Mbit/s při použití Enhanced Data Rate (EDR) a byla snížena spotřeba energie[5].

- **Bluetooth 3.0 + HS**
  V této verzi se objevila funkce High Speed (HS). To znamená, že je zde možnost spolupráce s Wi-Fi sítěmi, kdy Bluetooth slouží ke spárování zařízení a přenos dat probíhá přes Wi-Fi[9]. Rychlost tak může dosahovat až 24 Mbit/s.

- **Bluetooth 4.0**
  Tato verze nemá za úkol nahradit předchozí verzi, ale zaměřuje se na velmi nízkou spotřebu energie. Vzniká tak Bluetooth Low Energy (BLE) a technologie Bluetooth od této verze existuje ve variantách Classic, HS a BLE. Z hlediska vývoje se jedná o jednu z klíčových verzí, která nachází uplatnění v různých senzorech, čidlech, nositelné elektronice, apod, kde je třeba co nejvíce šetřit baterii[9]. Přenosová rychlost je 1 Mbit/s a dosah 50 metrů v exteriéru a 10 metrů v interiéru.

- **Bluetooth 4.2**
  Tato verze má pomoci internetu věcí (IoT), kdy implementuje protokol 6LoWPAN, který využívají třeba chytré žárovky[14].

- **Bluetooth 5.0**
  Mezi největší novinky této verze patří dvojnásobná přenosová rychlost a čtyřnásobný dosah. To ale neznamená, že zařízení budou komunikovat dvojnásobnou rychlostí na čtyřnásobnou vzdálenost. Dosah je až 240 metrů v exteriéru a až 40 metrů v interiéru, ale závisí na rychlosti. Platí, že čím nižší rychlost, tím větší dosah. Rychlost lze volit mezi 125 kbit/s, 500 kbit/s, 1 Mbit/s a 2 Mbit/s[5]. Dále dochází ke snížení spotřeby a zvýšení zabezpečení.

- **Bluetooth 5.1**
  Tato verze přináší pokrok ve velmi přesné navigaci, kdy zpřesnění se uvádí na centimetry a funguje i v uzavřených prostorech[5]. Dále dochází ke snížení energetické náročnosti.

- **Bluetooth 5.2**
  U této verze se přenosové rychlosti ani dosah nemění, ale dochází zde k dalšímu snížení spotřeby energie a soustředí se na zvýšení bezpečnosti[5].

V současné době nelze určit jakým směrem se další verze vyvíjet. Nejspíše se bude jednat o specializaci se na IoT (internet věcí) a snižování spotřeby.

2.5 Výkonové třídy Bluetooth

Bluetooth zařízení může pracovat v jedné ze tří výkonových tříd. Každá výkonová třída má definovaný maximální povolený výkon a přibližný dosah. Uvedené dosahy je nutno brát s rezervou, protože existuje mnoho faktorů, které ho mohou snížit a v praxi také snižují. Je rozdíl, pokud zařízení komunikují venku nebo ve vnitřních prostorách, kde se vyskytují různé překážky (např. zeď). Dále záleží na tom, v jak moc zarušeném prostředí se nacházíme a v neposlední řadě na kvalitě samotného zařízení.

- Třída 1
  - Maximální povolený výkon: 100 mW [1]
  - Přibližný dosah: 100 metrů [7]

  - Využití: v průmyslu, budování Bluetooth sítí, Bluetooth adaptéry, ...

- Třída 2
  - Maximální povolený výkon: 2,5 mW [1]
  - Přibližný dosah: 10 metrů [7]
  - Využití: mobilní zařízení a obecně tam, kde se předpokládá provoz na baterii (např. bezdrátová sluchátka)

- Třída 3
  - Maximální povolený výkon: 1 mW [1]
  - Přibližný dosah: 1 metr [7]

Některé zdroje uvádí i výkonovou třídu 1.5, ale ta se týká specifických zařízení. Proto není výše uvedena.

2.6 Bezpečnost Bluetooth

Bluetooth zařízení může pracovat v jednom ze tří režimů zabezpečení. První režim je bez zabezpečení a v tomto režimu lze navázat komunikaci s jakýmkoli zařízení. Druhý režim je zabezpečení na úrovni služeb, v tomto režimu se zajišťuje autorizace ke službám na zařízení. Třetím režimem je zabezpečení na úrovni spoje a v tomto režimu je třeba před navázáním spojení zajistit autentizaci a šifrování[1].

Entita	Délka
Adresa zařízení BD_ADDR	48 bitů
Privátní autentizační klíč	128 bitů
Privátní šifrovací klíč	8 - 128 bitů
Náhodné číslo	128 bitů
PIN	8 - 128 bitů

Tabulka 2.1: Entity využívané v autentizaci a šifrování[1]

Bluetooth poskytuje 3 základní bezpečnostní služby: **Autentizaci** - ověření totožnosti komunikujících zařízení, **Autorizaci** - povolení přístupu ke službám, **Důvěrnost** - přenášená data budou doručena tomu, komu mají být doručena a nebudou odposlouchávána[12].

Autentizace

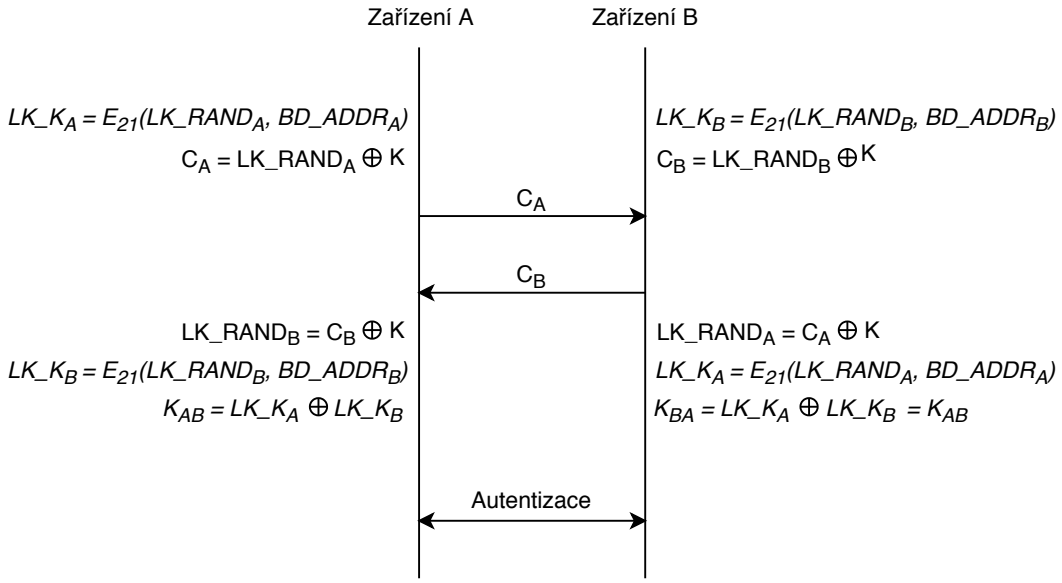
Během procesu autentizace dochází k ověření znalosti linkového klíče. Linkový klíč je 128 bitové číslo sdílené mezi zařízeními a slouží k autentizaci a současně je jedním z parametrů pro generování šifrovacího klíče. Linkovým klíčem se může stát jeden z následujících a záleží na aplikaci, který zvolí[1].

- **Klíč zařízení** (Unit key) K_A, který si zařízení vygeneruje z adresy BD_ADDR a 128 bitového náhodného čísla pomocí algoritmu E_{21}. Klíč se generuje při první instalaci, následně je uložen v paměti a neměl by být měněn. Typicky se volí u zařízení s malou pamětí, kam by se další klíč nevešel[1].

- **Inicializační klíč** (Init key) K_{init} je vytvořen při zahájení párování a v této fázi plní funkci linkového klíče, než se linkový klíč vygeneruje. Jakmile je spojový klíč vygenerován, K_{init} je zničen. K_{init} je odvozen z PINu, BD_ADDR a 128 bitového náhodného čísla IN_RAND. Pokud má jedno ze zařízení pevný PIN, použije se BD_ADDR toho druhého. Pokud mají obě zařízení volitelný PIN, použije se BD_ADDR toho zařízení, které obdrželo IN_RAND. Dvě zařízení s pevným PINem nelze spárovat[1].

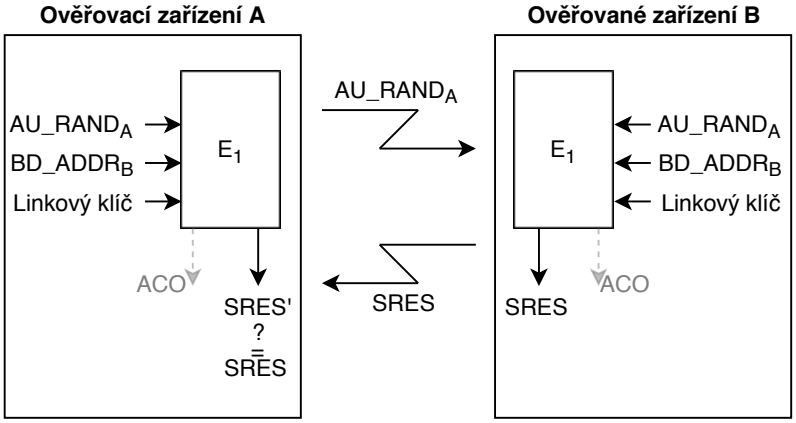
- **Dočasný klíč** (Temporary key) K_{master} je generován ze dvou náhodných čísel a je použit pokud master zařízení chce komunikovat s více zařízeními pomocí stejného šifrovacího klíče [1].

- **Kombinovaný klíč** (Combination key) K_{AB} vyžaduje informace z obou zařízení. Obě zařízení vygenerují své náhodné číslo LK_RAND_A a LK_RAND_B a společně se svými adresami BD_ADDR_A a BD_ADDR_B pomocí funkce E_{21} vygenerují čísla LK_K_A a LK_K_B. Z těchto vzniklých čísel a aktuálního linkového klíče K operací XOR vypočtou čísla C_A a C_B , které si vzájemně zašlou. Zařízení si za pomocí těchto čísel a aktuálního linkového klíče K operací XOR spočítají náhodné číslo druhého zařízení. Zařízení A z náhodného čísla LK_RAND_B a adresy BD_ADDR_B pomocí funkce E_{21} spočítá číslo LK_K_B. To stejné udělá zařízení B, ale s LK_RAND_A a BD_ADDR_A. Nyní zařízení A zná LK_K_B a zařízení B zná LK_K_A. Z těchto čísel operací XOR vznikne kombinovaný klíč K_{AB}[1].



Obrázek 2.3: Postup generování kombinovaného klíče [1]

Zařízení, které ověřuje identitu vygeneruje náhodné číslo AU_RAND_A a zašle jej ověřovanému zařízení. Obě zařízení následně zahájí proces výpočtu pomocí algoritmu $E1$, jehož vstupní parametry jsou: AU_RAND_A, adresa BD_ADDR_B a linkový klíč. Výsledkem je 24 bitová hodnota $SRES$ a 96 bitová hodnota ACO. Ověřované zařízení zašle $SRES$ druhému zařízení, které hodnotu porovná se svým výsledkem $SRES'$ (obr. 2.4).



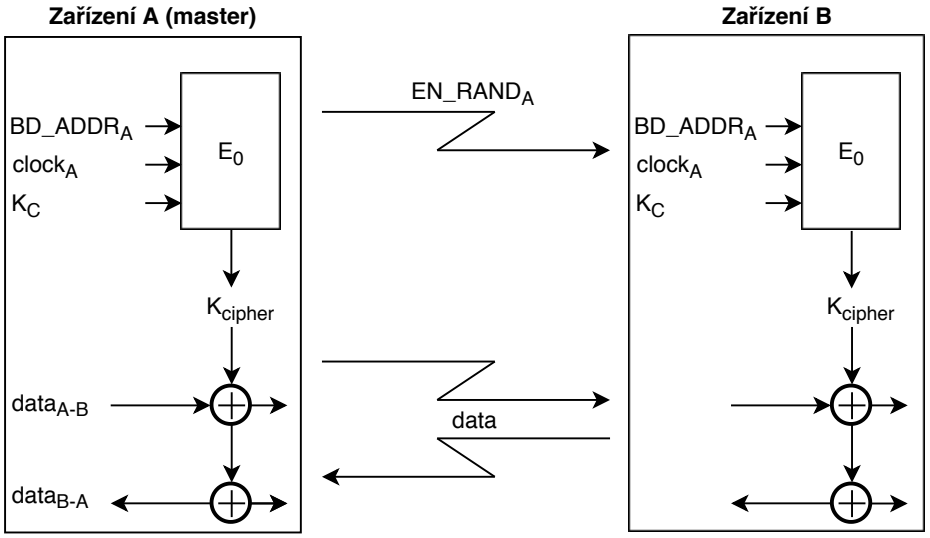
Obrázek 2.4: Proces autentizace [1]

Vypočtená hodnota ACO může být využita následně při šifrování, během generování šifrovacího klíče.

Šifrování

Pomocí algoritmu E_3 je vypočten šifrovací klíč K_C. Vstupní parametry jsou: linkový klíč, náhodné číslo a 96 bitový offset COF. Pokud je spojovým klíčem K_{master}, pak offset COF bude odvozen z adresy BD_ADDR$_{master}$, jinak se použije hodnota ACO, která byla vypočtena během procesu autentizace. [1]

Samotné šifrování paketů probíhá pomocí proudové šifry E_0, jejíž vstupní parametry jsou: šifrovací klíč K_C, adresa master zařízení BD_ADDR_A a hodinový takt master zařízení $clock_A$. Následně se přidá náhodné číslo EN_RAND_A je modifikován šifrovací klíč K_C na K'_C o délce 8 - 128 bitů. Výsledkem je šifra K_{cipher}, která je připojena k nezašifrovaným datům a tím se data zašifrují. Šifra K_{cipher} se počítá pro každý paket nová.



Obrázek 2.5: Proces šifrování paketů [1]

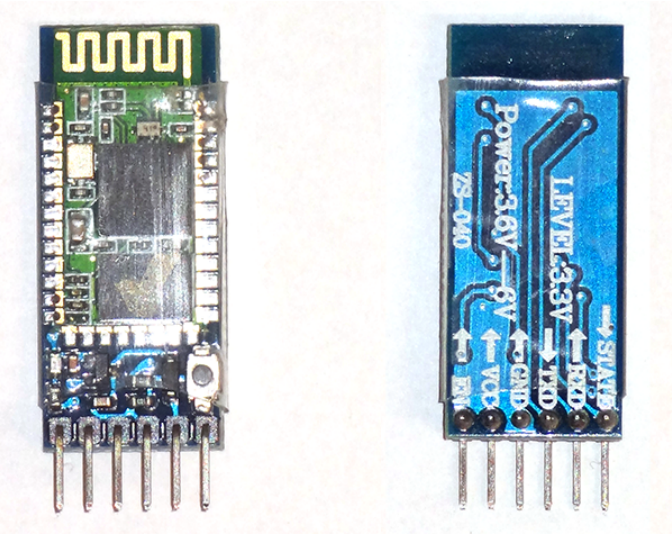
Kapitola 3

Technologické moduly a vývojové prostředky

V této kapitole jsou popsány technologické moduly a vývojové prostředky, které mají bezprostřední vztah k práci. Nejsou zde uvedeny všechny dostupné moduly a informace, protože by se nevešly do rozsahu práce.

3.1 Bluetooth modul

Bluetooth modul je hardwarové zařízení, které je schopno bezdrátově vysílat a přijímat data pomocí Bluetooth technologie. V této bakalářské práci je použit model HC-05 (obr. 3.1), kterému se bude tato podkapitola dále věnovat. Tento modul se prodává přibližně za 200 Kč.



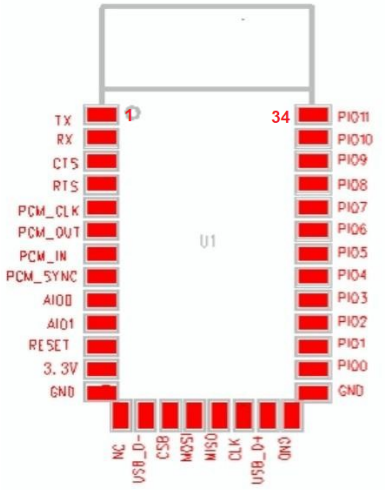
Obrázek 3.1: Bluetooth modul HC-05

Technické specifikace

HC-05 komunikuje pomocí protokolu SPP (Service Port Protocol), který je navržen pro bezdrátovou sériovou komunikaci[2]. Specifikace modulu jsou následující:

- Napájecí napětí: 3,6 - 6 V
- Maximální odběr proudu: 40 mA
- Dosah: přibližně 10 metrů
- Bluetooth verze: 2.0 + EDR
- Režim provozu: master/slave
- UART rozhraní s nastavitelným baud rate (9600, 19200, 38400, 57600, 115200, 230400, 460800)
- Programovatelné I/O piny
- Vysílací výkon až 4dBm (třída 2)

Ve výchozím stavu je modul v režimu slave a lze ho najít pod názvem "H-C-2010-06-01". PIN pro spárování je "1234"a komunikační rychlost UART rozhraní je 36400 bd/s [2].



Obrázek 3.2: Piny modulu HC-05 [2]

Název pinu	#	Popis
TX	1	UART odesílání dat
RX	2	UART přijímání dat
RESET	11	Reset
3.3V	12	Napájení (VCC)
GND	13, 21, 22	GND
PIO0	23	Programovatelný I/O pin
PIO1	24	Programovatelný I/O pin
PIO2	25	Programovatelný I/O pin

PIO3	26	Programovatelný I/O pin
PIO4	27	Programovatelný I/O pin
PIO5	28	Programovatelný I/O pin
PIO6	29	Programovatelný I/O pin
PIO7	30	Programovatelný I/O pin
PIO8	31	Programovatelný I/O pin
PIO9	32	Programovatelný I/O pin
PIO10	33	Programovatelný I/O pin
PIO11	34	Programovatelný I/O pin

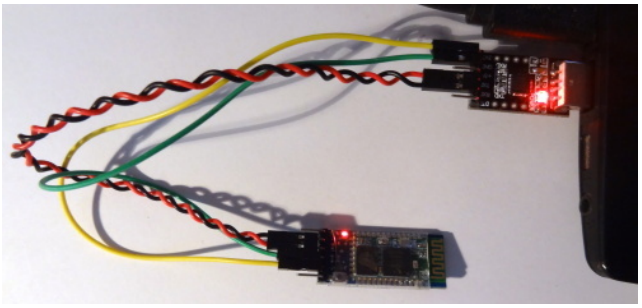
Tabulka 3.1: Piny modulu HC-05

Konfigurace modulu

Modul může běžet v režimu přenosu dat (data mode) nebo v příkazovém režimu (command mode). Ve výchozím stavu modul běží v data modu, ve kterém plní svoji primární funkci a to je komunikace přes Bluetooth. Ke konfiguraci modulu slouží příkazový režim, do kterého je potřeba se přepnout a v něm lze pomocí AT příkazů modul konfigurovat. Existují dva způsoby, jak se přepnout do příkazového režimu:

- **1. způsob**
  1. Modul je již spuštěn
  2. Přivést na PIO11 napětí (stisknout tlačítko)
- **2. způsob**
  1. Modul je vypnutý
  2. Přivést na PIO11 napětí (stisknout a držet tlačítko)
  3. Přivést na modul napětí
  4. Odpojit PIO11 od napětí (pustit tlačítko)

Při použití prvního způsobu bude komunikační rychlost stejná jako doposud. U druhého způsobu bude komunikační rychlost vždy 38400 bd/s. Tento způsob může přijít vhod, když zapomenu jaká rychlost je nastavena. Zpět do režimu přenosu dat se lze v obou případech dostat resetováním modulu pomocí AT příkazu *AT+RESET* nebo odpojením modulu od napájení a opětovném připojení.



Obrázek 3.3: Připojení HC-05 k PC

AT příkazy

AT příkaz začíná prefixem *AT* a v kombinaci s dalšími parametry lze pomocí příkazové řádky komunikovat se zařízením[3]. Ukončení příkazu (bez znaku konce řádku, \r, \n nebo \r\n) záleží na implementaci výrobcem. V případě modulu HC-05 musí být příkazy zakončeny znaky \r\n.

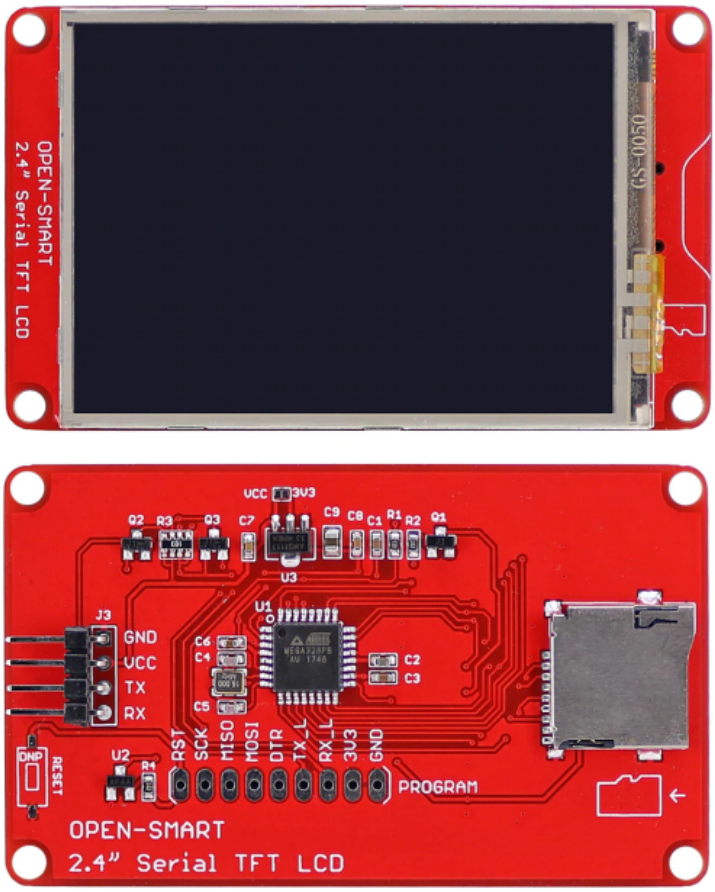
Modul HC-05 nabízí přes 30 AT příkazů, v následující tabulce jsou uvedeny pouze ty nejužitečnější.

Příkaz	**Odpověď**	**Parametry**	**Účel**
AT	OK		Test
AT+RESET	OK		Reset
AT+ORGL	OK		Výchozí nastavení
AT+ADDR?	+ADDR:<Param> OK	Param: adresa modulu	Zjištění adresy modulu
AT+NAME=<Param>	OK	Param: název modulu	Nastavení názvu modulu
AT+ROLE=<Param>	OK	Param: 0 - Slave 1 - Master 2 - Slave-Loop	Nastavení/ zjištění role
AT+ROLE?	+ROLE:<Param> OK		
AT+PSWD=<Param>	OK	Param: PIN (Výchozí: 1234)	Nastavení/ zjištění PINu
AT+PSWD?	+PSWD:<Param> OK		
AT+UART=<Param1>, <Param2>,<Param3>	OK	Param1: baud Param2: # stop bitů Param3: # parit. bitů	Nastavení/ zjištění UART
AT+UART?	+UART:<Param1>, <Param2>,<Param3> OK		
AT+PIO=<Param1>, <Param2>	OK	Param1: # PIO pinu Param2: log. hodnota	Nastavení PIO pinů

Tabulka 3.2: AT příkazy Bluetooth modulu

3.2 Displej

Pro vypisování informací byl zvolen TFT LCD displej od společnosti OPEN-SMART, zakoupený na Aliexpressu[1]. Pro tento displej jsem se rozhodl, protože komunikace probíhá přes rozhraní UART, kterým disponuje i Bluetooth modul.



Obrázek 3.4: Použitý LCD displej

Cena displeje se pohybuje okolo 12 dolarů, v přepočtu cca 280 Kč.

Specifikace displeje

Displej lze napájet napětím 5V (4,5-5,5V) nebo 3,3V (3,1-3,4V), maximální odběr proudu výrobce udává 90mA. Úhlopříčka činí 2,4"při rozlišení 320x240px. Ve výchozím stavu komunikace přes UART probíhá při rychlosti 9600bd/s, kterou lze změnit na 19200/38400/57600/115200 bd/s. Displej dále disponuje slotem na microSD kartu, tudíž lze zobrazovat i obrázky, je zde ale omezení na formát BMP. Za zmínku stojí i to, že displej je dotykový, ale tuto funkcionalitu nejspíše nevyužiji.

[1] Odkaz na displej

Ovládání displeje a přehled příkazů

Ovládání probíhá zasíláním příkazů o maximální délce 64 bajtů. Příkazy jsou ve tvaru *StartByte Délka Příkaz Data StopByte*, kde:

- *StartByte* značí začátek příkazu a je vždy 0x7E
- *Délka* značí délku příkazu, zahrnuje Příkaz, Data a StopByte
- *Příkaz* určuje o jaký příkaz je jedná
- *Data* obsahují samotná data, limit 60 bajtů
- *StopByte* značí konec příkazu a je vždy 0xEF

V následující tabulce je přehled podporovaných příkazů. Modrou barvou je zvýrazněný kód příkazu a červeně je zvýrazněn obsah dat. V případě, že nějaká hodnota v datové části je větší než 1 bajt, je rozdělena do více bajtů. Například u příkazu *SET_TEXTCOLOR*, který nastaví barvu textu, je barva *c* 16-bitové číslo (barva je ve formátu *565* - 5 bitů pro červenou barvu, 6 bitů pro modrou barvu a 5 bitů pro zelenou barvu), tudíž *cH* je 8 horních bitů a *cL* je spodních 8 bitů.

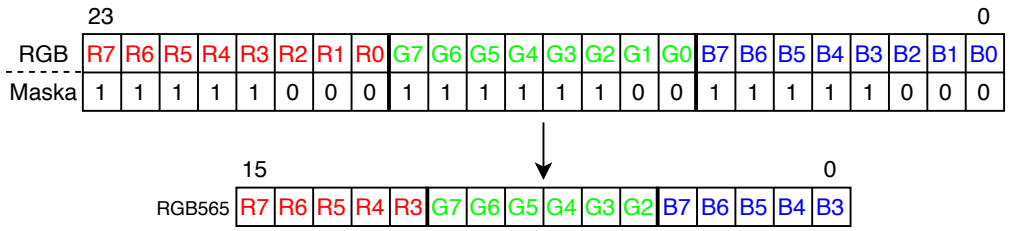
Příkaz	**Zápis (HEX)**	**Popis**
TEST	7E 02 00 EF	Testovací příkaz
READ_CURSOR	7E 02 01 EF	Načte pozici kurzoru
SET_CURSOR	7E 06 01 xH xL yH yL EF	Nastaví pozici kurzoru
SET_TEXTCOLOR	7E 04 02 cH cL EF	Nastaví barvu textu
SET_TEXTSIZE	7E 03 03 velikost EF	Nastaví velikost textu
SET_ROTATION	7E 03 04 rotace EF	Nastaví rotaci
RESET	7E 02 05 EF	Restartuje displej
SET_BACKLIGHT	7E 03 06 intenzita EF	Nastaví jas podsvícení
PRINTLN	7E 02 10 EF	Nový řádek
PRINT_CHAR_ARRAY	7E LEN 11 data EF	Vypíše text
PRINT_INT_8	7E 05 12 sign base data EF	Vypíše 8-bitové číslo
PRINT_INT_16	7E 06 13 sign base dataH dataL EF	Vypíše 16-bitové číslo
PRINT_INT_32	7E 08 14 sign base dHH dH dL dLL EF	Vypíše 32-bitové číslo
FILL_SCREEN	7E 04 20 cH cL EF	Překreslí obrazovku konkrétní barvou
DRAW_PIXEL	7E 08 21 xH xL yH yL cH cL EF	Vykreslí bod
DRAW_FASTVLINE	7E 0A 22 xH xL yH yL vH vL cH cL EF	Vykreslí vertikální čáru
DRAW_FASTHLINE	7E 0A 23 xH xL yH yL dH dL cH cL EF	Vykreslí horizontální čáru
DRAW_LINE	7E 0C 24 x0H x0L y0H y0L x1H x1L y1H y1L cH cL EF	Vykreslí čáru
DRAW_RECT	7E 0C 25 xH xL yH yL sH sL vH vL cH cL EF	Vykreslí obdélník

FILL_RECT	7E 0C 26 xH xL yH yL sH sL vH vL cH cL EF	Vyplní obdélník
DRAW_CIRCLE	7E 0A 27 x0H x0L y0H y0L rH rL cH cL EF	Vykreslí kružnici
FILL_CIRCLE	7E 0A 28 x0H x0L y0H y0L rH rL cH cL EF	Vyplní kruh
DRAW_TRIANGLE	7E 10 29 x0H x0L y0H y0L x1H x1L y1H y1L x2H x2L y2H y2L cH cL EF	Vykreslí trojúhelník
FILL_TRIANGLE	7E 10 2A x0H x0L y0H y0L x1H x1L y1H y1L x2H x2L y2H y2L cH cL EF	Vyplní trojúhelník
DRAW_ROUNDRECT	7E 0E 2B xH xL yH yL sH sL vH vL rH rL cH cL EF	Vykreslí obdélník se zaoblenými rohy
FILL_ROUNDRECT	7E 0E 2C xH xL yH yL sH sL vH vL rH rL cH cL EF	Vyplní obdélník se zaoblenými rohy
DRAW_BMP	7E LEN 30 název EF	Vykreslí obrázek z SD
READ_BAUD	7E 02 40 EF	Načte přenos. rychlost
SET_BAUD	7E 02 40 baud EF	Nastaví přenos. rychlost
READ_VERSION	7E 02 41 EF	Načte verzi firmware
READ_DRIVER_ID	7E 02 42 EF	Načte driver ID
READ_RESOLUTION	7E 02 43 EF	Načte rozlišení displeje

Tabulka 3.3: Přehled příkazů displeje [4]

Popis příkazů

V této podkapitole jsou detailně popsány jednotlivé příkazy společně s příklady použití a v případech, kdy je to vhodné, je uvedena i fotodokumentace. Jestliže se v konkrétních příkazech nastavuje barva (textu, úsečky, ...), tak nastavovaná barva je ve formátu *RGB565*, kdy 5 nejvyšších bitů je vyhrazeno pro modrou barvu, následujících 6 bitů pro zelenou barvu a nejnižších 5 bitů pro modrou barvu (převod z RGB je znázorněn na obr. 3.5).



Obrázek 3.5: Převod z RGB do RGB565

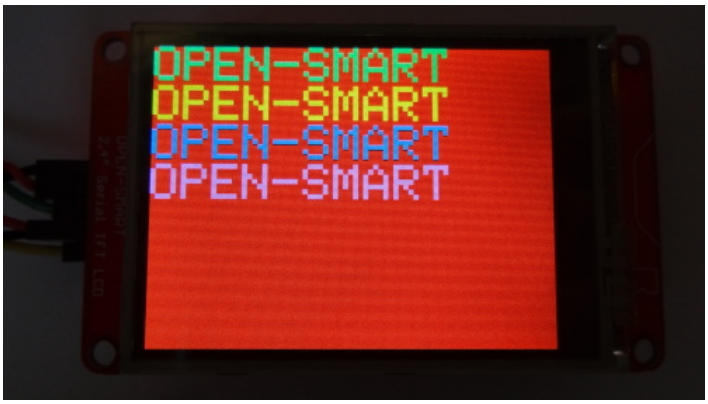
Příkazy pro vypisování textu, čísel a obrázků

Následující příkazy jsou určeny pro vypisování textu, čísel a nastavení parametrů, jako je barva písma, pozice kurzoru, apod.

- **SET_CURSOR**
  Příkaz slouží k nastavení pozice kurzoru a je ve tvaru *7E 06 01 xH xL yH yL EF*, kde:
  - xH - významnější bajt x-ové souřadnice
  - xL - méně významný bajt x-ové souřadnice
  - yH významnější bajt y-ové souřadnice
  - yL - méně významný bajt y-ové souřadnice

  Příklad: Pro nastavení kurzoru na souřadnice x=260, y=130 bude příkaz vypadat následovně: *7E 06 01 01 04 00 82 EF*

- **SET_TEXTCOLOR**
  Příkaz slouží k nastavení barvy textu vypisovaného na displej a je ve tvaru *7E 04 02 cH cL EF*, kde:
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy

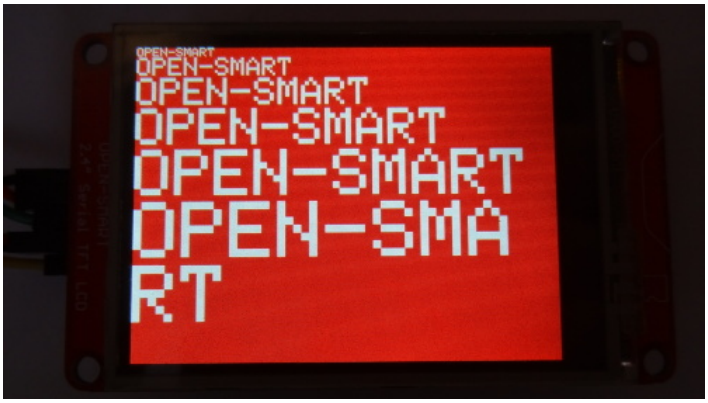


Obrázek 3.6: Příkaz SET_TEXTCOLOR

  Příklad: Pro nastavení barvy textu na žlutou barvu bude příkaz vypadat následovně: *7E 04 02 FF E0 EF*

- **PRINTLN**
  Příkaz slouží k odřádkování kurzoru a je ve tvaru *7E 02 10 EF*

- **SET_TEXTSIZE**
  Příkaz slouží k nastavení velikosti textu vypisovaného na displej. Příkaz je ve tvaru *7E 03 03 velikost EF*. Velikost každého znaku bude 5*velikost pixelů na šířku a 7*velikost pixelů na výšku.



Obrázek 3.7: Příkaz SET_TEXTSIZE

Příklad: Při zaslání příkazu *7E 03 03 03 EF* bude každý znak 15 pixelů široký a 21 pixelů vysoký

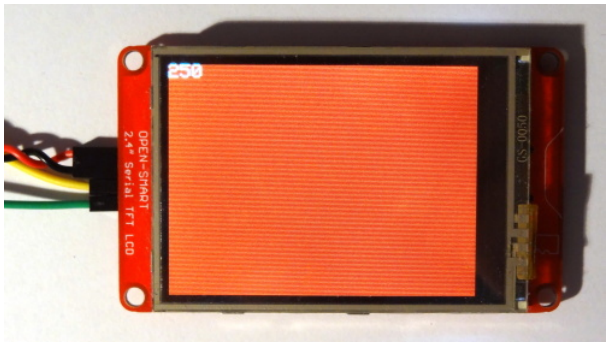
- **PRINT_CHAR_ARRAY**
  Příkaz slouží k výpisu textu na displej. Vzhledem k tomu, že příkaz může mít maximálně 64 bajtů a 4 bajty jsou již využity na StartByte, délku příkazu, typ příkazu a StopByte, na samotný text zbývá maximálně 60 znaků. Více znaků v jednom příkazu zaslat nelze, ale vzhledem k velikosti displeje si myslím, že to není nijak limitující. Vypisovat lze pouze ASCII znaky. Tvar příkazu je *7E length 11 text EF*, kde:
  - length - počet znaků + 2
  - text - jednotlivé znaky textu (v hexadecimálním zápisu)

Obrázek 3.8: Příkaz PRINT_CHAR_ARRAY

Příklad: zasláním příkazu *7E 0E 11 48 65 6C 6C 6F 2C 20 77 6F 72 6C 64 EF* se na displej vypíše "Hello, world"

- **PRINT_INT_8**
  Příkaz slouží k vypsání 8 bitového čísla. Číslo může být vypsáno ve binární, oktalové, decimální nebo hexadecimální soustavě. Zároveň může být znaménkové nebo neznaménkové (znaménkové může být pouze v decimální soustavě). Příkaz je ve tvaru *7E 05 12 sign base data EF*, kde:

- sign: pro znaménkové číslo 01, pro neznaménkové 00
- base: 02 - binární soustava, 08 - oktalová soustava, 0A - decimální soustava, 10 - hexadecimální soustava
- data - konkrétní číslo



Obrázek 3.9: Příkaz PRINT_INT_8

Příklad: zasláním příkazu 7E 05 12 00 0A FA EF se vypíše číslo 250

- **PRINT_INT_16**
  Příkaz slouží k vypsání 16 bitového čísla. Číslo může být vypsáno ve binární, oktalové, decimální nebo hexadecimální soustavě. Zároveň může být znaménkové nebo neznaménkové (znaménkové může být pouze v decimální soustavě). Příkaz je ve tvaru *7E 06 13 sign base dataH dataL EF*, kde:
  - sign: pro znaménkové číslo 01, pro neznaménkové 00
  - base: 02 - binární soustava, 08 - oktalová soustava, 0A - decimální soustava, 10 - hexadecimální soustava
  - dataH - významnější bajt čísla
  - dataL - méně významný bajt čísla

Obrázek 3.10: Příkaz PRINT_INT_16

Příklad: Zasláním příkazu 7E 06 13 00 10 AB CD EF se vypíše číslo 0xABCD

- **PRINT_INT_32**
  Příkaz slouží k vypsání 32 bitového čísla. Číslo může být vypsáno ve binární, oktalové, decimální nebo hexadecimální soustavě. Zároveň může být znaménkové nebo neznaménkové (znaménkové může být pouze v decimální soustavě). Příkaz je ve tvaru *7E 08 14 sign base dataHH dataH dataL dataLL EF*, kde:

  – sign: pro znaménkové číslo 01, pro neznaménkové 00
  – base: 02 - binární soustava, 08 - oktalová soustava, 0A - decimální soustava, 10 - hexadecimální soustava
  – dataHH - nejvýznamnější bajt čísla
  – dataH - druhý nejvýznamnější bajt čísla
  – dataL - třetí nejvýznamnější bajt čísla
  – dataLL - nejméně významný bajt čísla

Obrázek 3.11: Příkaz PRINT_INT_32

Příklad: Zasláním příkazu *7E 08 14 01 0A FF FD 3E 66 EF* se vypíše číslo -180634

- **DRAW_BMP**
  Tento příkaz na displej vykreslí na displej obrázek (z SD karty) ve formátu BMP. Příkaz je ve tvaru *7E len 30 data EF*, kde:

  – len - počet znaků v názvu souboru (bez přípony) + 2
  – data - jednotlivé znaky názvu souboru (v hexadecimálním zápisu, bez přípony)
  – Návratová hodnota:

  Návratová hodnota příkazu může být:

  – 7E 03 6F 6B EF (ok)
  – 7E 03 65 31 EF (e1 - SD kartu se nepodařilo načíst nebo není vložena)
  – 7E 03 65 32 EF (e2 - obrázek se nepodařilo otevřít nebo neexistuje)

  Příklad: Příkaz 7E 09 30 66 69 74 6C 6F 67 6F EF vykreslí obrázek "fitlogo.bmp"z SD karty

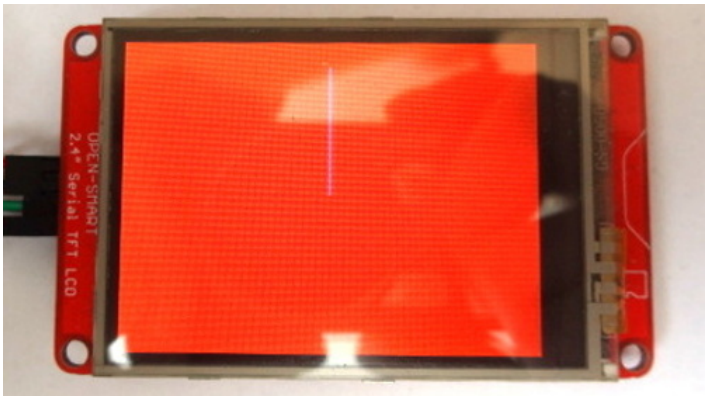
Příkazy pro vykreslování geometrických obrazců a úseček

Následující příkazy slouží k vykreslování geometrických obrazců (obdélník, kružnice, ...) a úseček.

- **DRAW_PIXEL**
  Tento příkaz slouží k vykreslení jednoho pixelu konkrétní barvou a je ve tvaru *7E 08 21 xH xL yH yL cH cL EF*, kde:
  - xH - významnější bajt x-ové souřadnice
  - xL - méně významný bajt x-ové souřadnice
  - yH významnější bajt y-ové souřadnice
  - yL - méně významný bajt y-ové souřadnice
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy

  Příklad: Zasláním příkazu 7E 08 *21 01 04 00 82 00 1F EF* se pixel na souřadnici x=260, y=130 vybarví modrou barvou

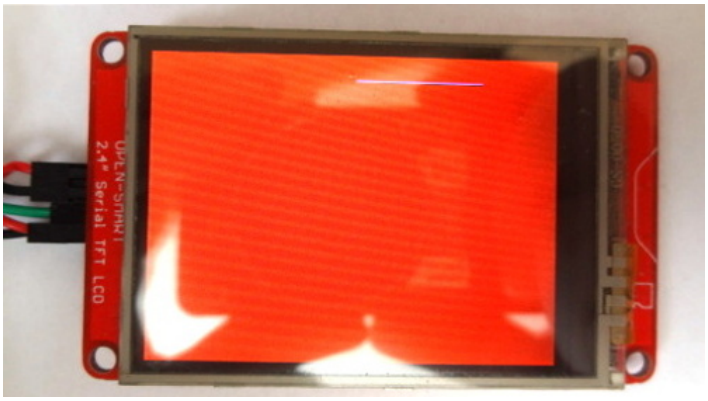
- **DRAW_FASTVLINE**
  Tento příkaz slouží k vykreslení vertikální čáry o určité výšce a barvě. Výška se udává v pixelech. Příkaz je ve tvaru *7E 0A 22 xH xL yH yL vH vL cH cL EF*, kde:
  - xH - významnější bajt x-ové souřadnice
  - xL - méně významný bajt x-ové souřadnice
  - yH - významnější bajt y-ové souřadnice
  - yL - méně významný bajt y-ové souřadnice
  - vH - významnější bajt výšky čáry
  - vL - méně významný bajt výšky čáry
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy



Obrázek 3.12: Příkaz DRAW_FASTVLINE

Příklad: Příkaz *7E 0A 22 00 A0 00 14 00 64 00 1F EF* vykreslí ze souřadnice x=160, y=20 modrou čáru o výšce 100 pixelů

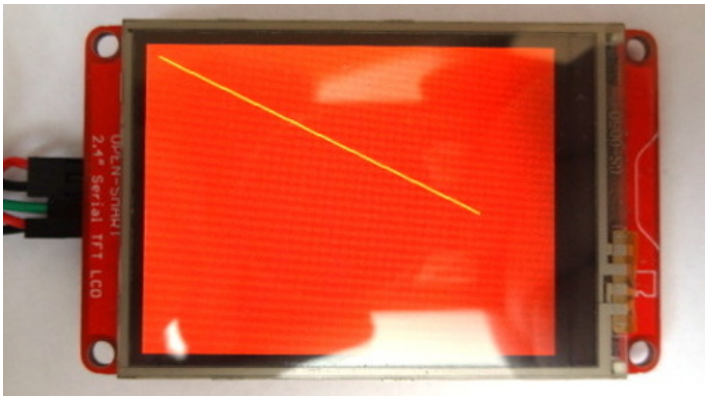
- **DRAW_FASTHLINE**
  Tento příkaz slouží k vykreslení horizontální čáry o určité délce a barvě. Délka se udává v pixelech. Příkaz je ve tvaru *7E 0A 23 xH xL yH yL dH dL cH cL EF*, kde:
  - xH - významnější bajt x-ové souřadnice
  - xL - méně významný bajt x-ové souřadnice
  - yH významnější bajt y-ové souřadnice
  - yL - méně významný bajt y-ové souřadnice
  - dH - významnější bajt délky čáry
  - dL - méně významný bajt délky čáry
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy



Obrázek 3.13: Příkaz DRAW_FASTHLINE

Příklad: Příkaz *7E 0A 23 00 A0 00 14 00 64 00 1F EF* vykreslí ze souřadnice x=160, y=20 modrou čáru o délce 100 pixelů

- **DRAW_LINE**
  Tento příkaz vykreslí čáru z bodu (x0, y0) do bodu (x1, y1) a je ve tvaru *7E 0C 24 x0H x0L y0H y0L x1H x1L y1H y1L cH cL EF*, kde:
  - x0H - významnější bajt počátku x-ové souřadnice
  - x0L - méně významný bajt počátku x-ové souřadnice
  - y0H - významnější bajt počátku y-ové souřadnice
  - y0L - méně významný bajt počátku y-ové souřadnice
  - x1H - významnější bajt konce x-ové souřadnice
  - x1L - méně významný bajt konce x-ové souřadnice
  - y1H - významnější bajt konce y-ové souřadnice
  - y1L - méně významný bajt konce y-ové souřadnice
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy



Obrázek 3.14: Příkaz DRAW_LINE

Příklad: Příkaz 7E 0C *24 00 0A 00 0A 01 04 00 82 FF E0 EF* vykreslí žlutou čáru začínající v bodě (x0=10, y0=10) a končící v bodě (x1=260, y1=130)

- **DRAW_RECT a FILL_RECT**
  Příkaz DRAW_RECT vykreslí obdélník o určité šířce a výšce, tvar příkazu je *7E 0C 25 xH xL yH yL sH sL vH vL cH cL EF*. Příkaz FILL_RECT vykreslí obdélník o určité šířce a výšce a současně ho celý vyplní barvou, tvar příkazu je *7E 0C 26 xH xL yH yL sH sL vH vL cH cL EF*. Následující parametry jsou stejné pro oba příkazy.
  - xH - významnější bajt počáteční x-ové souřadnice
  - xL - méně významný bajt počáteční x-ové souřadnice
  - yH - významnější bajt počáteční y-ové souřadnice
  - yL - méně významný bajt počáteční y-ové souřadnice
  - sH - významnější bajt šířky obdélníku
  - sL - méně významný bajt šířky obdélníku
  - vH - významnější bajt výšky obdélníku
  - vL - méně významný bajt výšky obdélníku
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy

Obrázek 3.15: Příkaz DRAW_RECT

Příklad: Příkaz *7E 0C 25 00 0A 00 0A 00 64 00 32 FF E0 EF* vykreslí žlutý obdélník z bodu (x=10, y=10), který je 100 pixelů široký a 50 pixelů vysoký.

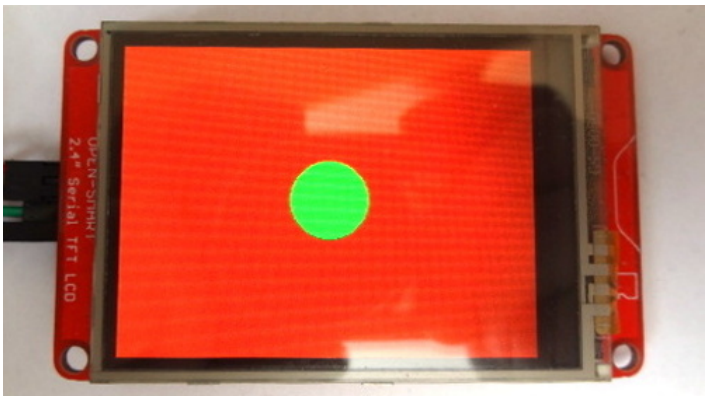
Obrázek 3.16: Příkaz FILL_RECT

Příklad: Příkaz *7E 0C 26 00 0A 00 0A 00 64 00 32 FF E0 EF* vykreslí obdélník z bodu (x=10, y=10), který je 100 pixelů široký a 50 pixelů vysoký a celý ho vyplní žlutou barvou.

- **DRAW_CIRCLE a FILL_CIRCLE**
  Příkaz DRAW_CIRCLE vykreslí kružnici o určitém poloměru se středem v bodě (x, y). Příkaz je ve tvaru *7E 0A 27 xH xL yH yL rH rL cH cL EF*. Příkaz FILL_CIRCLE vykreslí kruh o určitém poloměru se středem v bodě (x, y). Tvar příkazu je *7E 0A 28 xH xL yH yL rH rL cH cL EF*. Následující parametry jsou pro oba příkazy stejné.
  - xH - významnější bajt x-ové souřadnice středu kružnice
  - xL - méně významný bajt x-ové souřadnice středu kružnice
  - yH - významnější bajt y-ové souřadnice středu kružnice
  - yL - méně významný bajt y-ové souřadnice středu kružnice
  - rH - významnější bajt poloměru kružnice
  - rL - méně významný bajt poloměru kružnice
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy

Obrázek 3.17: Příkaz DRAW_CIRCLE

Příklad: Příkaz *7E 0A 27 00 A0 00 78 00 1E 07 E0 EF* vykreslí zelenou kružnici se středem (x=160, y=120) a poloměrem 30 pixelů

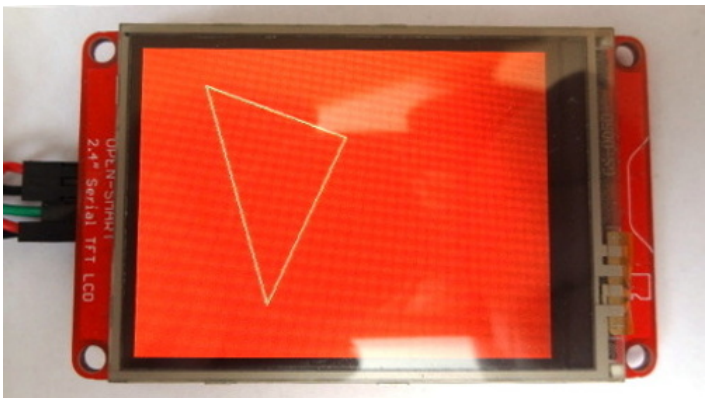


Obrázek 3.18: Příkaz FILL_CIRCLE

Příklad: Příkaz *7E 0A 28 00 A0 00 78 00 1E 07 E0 EF* vykreslí zelený kruh se středem (x=160, y=120) a poloměrem 30 pixelů

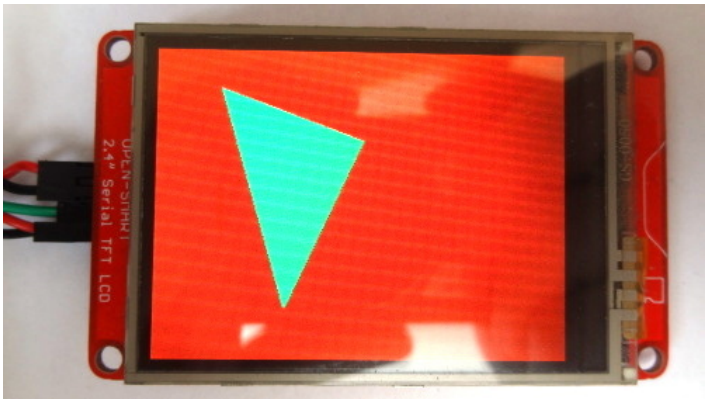
- **DRAW_TRIANGLE a FILL_TRIANGLE**
  Příkaz DRAW_TRIANGLE vykreslí trojúhelník, ale nevyplní ho barvou. Jednotlivé body trojúhelníku jsou dány souřadnicemi (x0, y0), (x1, y1) a (x2, y2). Příkaz je ve tvaru *7E 10 29 x0H x0L y0H y0L x1H x1L y1H y1L x2H x2L y2H y2L cH cL EF*. Příkaz FILL_TRIANGLE vykreslí trojúhelník a současně ho celý vyplní zvolenou barvou. Tvar příkazu je *7E 10 2A x0H x0L y0H y0L x1H x1L y1H y1L x2H x2L y2H y2L cH cL EF*. Následující parametry jsou stejné pro oba příkazy.
  - x0H - významnější bajt x-ové souřadnice bodu A
  - x0L - méně významný bajt x-ové souřadnice bodu A
  - y0H - významnější bajt y-ové souřadnice bodu A
  - y0L - méně významný bajt y-ové souřadnice bodu A
  - x1H - významnější bajt x-ové souřadnice bodu B
  - x1L - méně významný bajt x-ové souřadnice bodu B

- y1H - významnější bajt y-ové souřadnice bodu B
- y1L - méně významný bajt y-ové souřadnice bodu B
- x2H - významnější bajt x-ové souřadnice bodu C
- x2L - méně významný bajt x-ové souřadnice bodu C
- y2H - významnější bajt y-ové souřadnice bodu C
- y2L - méně významný bajt y-ové souřadnice bodu C
- cH - významnější bajt barvy
- cL - méně významný bajt barvy



Obrázek 3.19: Příkaz DRAW_TRIANGLE

Příklad: Příkaz *7E 10 29 00 32 00 1E 00 A0 00 46 00 64 00 C8 07 0E EF* vykreslí trojúhelník, který má body na souřadnicích (x0=50, y0=30), (x1=160, y1=70) a (x2=100, y2=200)



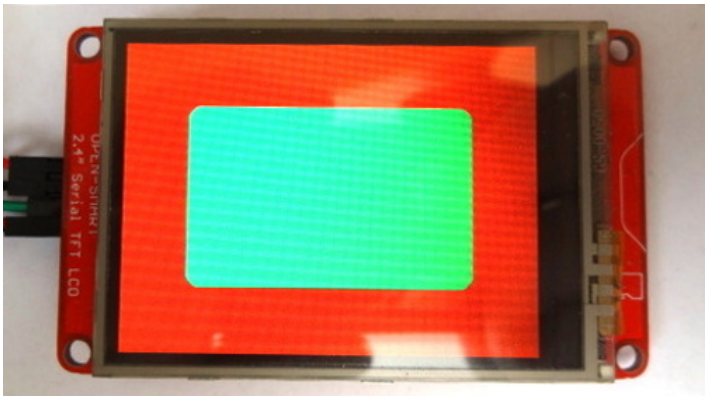
Obrázek 3.20: Příkaz FILL_TRIANGLE

Příklad: Příkaz *7E 10 2A 00 32 00 1E 00 A0 00 46 00 64 00 C8 07 0E EF* vykreslí trojúhelník, který má body na souřadnicích (x0=50, y0=30), (x1=160, y1=70) a (x2=100, y2=200) a vyplní ho zelenou barvou

- **DRAW_ROUNDRECT a FILL_ROUNDRECT**
  Příkaz DRAW_ROUNDRECT vykreslí obdélník se zaoblenými rohy a je ve tvaru *7E 0E 2B xH xL yH yL sH sL vH vL rH rL cH cL EF*. Příkaz FILL_ROUNDRECT vykreslí obdélník se zaoblenými rohy a současně ho vyplní barvou. Příkaz je ve tvaru *7E 0E 2C xH xL yH yL sH sL vH vL rH rL cH cL EF*. Následující parametry jsou pro oba příkazy stejné.
  - xH - významnější bajt počáteční x-ové souřadnice
  - xL - méně významný bajt počáteční x-ové souřadnice
  - yH - významnější bajt počáteční y-ové souřadnice
  - yL - méně významný bajt počáteční y-ové souřadnice
  - sH - významnější bajt šířky obdélníku
  - sL - méně významný bajt šířky obdélníku
  - vH - významnější bajt výšky obdélníku
  - vL - méně významný bajt výšky obdélníku
  - rH - významnější bajt rádiusu zaoblení
  - rL - méně významný bajt rádiusu zaoblení
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy

Obrázek 3.21: Příkaz DRAW_ROUNDRECT

Příklad: Příkaz 7E 0E 2B 00 32 00 32 00 DC 00 8C 00 0A 07 0E EF vykreslí obdélník se zaoblenými rohy o rádiusu 10 pixelů



Obrázek 3.22: Příkaz FILL_ROUNDRECT

Příklad: Příkaz *7E 0E 2C 00 32 00 32 00 DC 00 8C 00 0A 07 0E EF* vykreslí obdélník se zaoblenými rohy o rádiusu 10 pixelů

Informační příkazy

- **TEST**
  Příkaz slouží ke kontrole, zda je displej připraven k použití. Příkaz je ve tvaru *7E 02 00 EF*

- **READ_CURSOR**
  Příkaz slouží ke zjištění, kde se aktuálně nachází kurzor. Tvar příkazu je *7E 02 01 EF*, displej následně vrátí *7E 06 01 xH xL yH yL EF 7E 03 6F 6B EF (ok)*, kdy
  - x = xH*256+xL
  - y = yH*256+yL

- **READ_BAUD**
  Tento příkaz slouží ke zjištění nastavené komunikační rychlosti. Příkaz je ve tvaru *7E 02 40 EF*, displej následně vrátí *7E 03 40 baud EF 7E 03 6F 6B EF (ok)*, kdy baud může nabývat hodnot:
  - 00 = 9600 bd/s
  - 01 = 19200 bd/s
  - 02 = 38400 bd/s
  - 03 = 57600 bd/s
  - 04 = 115200 bs/s

- **READ_VERSION**
  Příkaz slouží ke zjištění verze firmware. Příkaz je ve tvaru *7E 02 41 EF*, displej následně vrátí *7E 03 41 verze EF 7E 03 6F 6B EF (ok)*. Když verze nabývá hodnoty *0x10*, znamená to, že verze firmware je 1.0

- **READ_DRIVER_ID**
  Příkaz slouží ke zjištění řadiče displeje a je ve tvaru *7E 02 42 EF*. Displej následně vrátí *7E 04 42 idH idL EF 7E 03 6F 6B EF (ok)*. Když je idH 0x93 a idL 0x25, znamená to, že je použitý řadič ILI9325.

- **READ_RESOLUTION**
  Příkaz slouží ke zjištění rozlišení displeje (může se lišit v závislosti na modelu nebo jinak nastavené rotace). Příkaz je ve tvaru *7E 02 43 EF*, displej následně vrátí *7E 06 43 xH xL yH yL EF 7E 03 6F 6B EF (ok)*, kdy
  - x = xH*256+xL
  - y = yH*256+yL

Ostatní příkazy

- **RESET**
  Příkaz slouží k resetování mikrokontroléru displeje do původního stavu, ve kterém se nachází po jeho spuštění. Příkaz je ve tvaru *7E 02 05 EF*.

Obrázek 3.23: Příkaz RESET

- **FILL_SCREEN**
  Tímto příkazem se smaže obsah displeje a překreslí se zvolenou barvou. Příkaz je ve tvaru *7E 04 20 cH cL EF*, kde:
  - cH - významnější bajt barvy
  - cL - méně významný bajt barvy

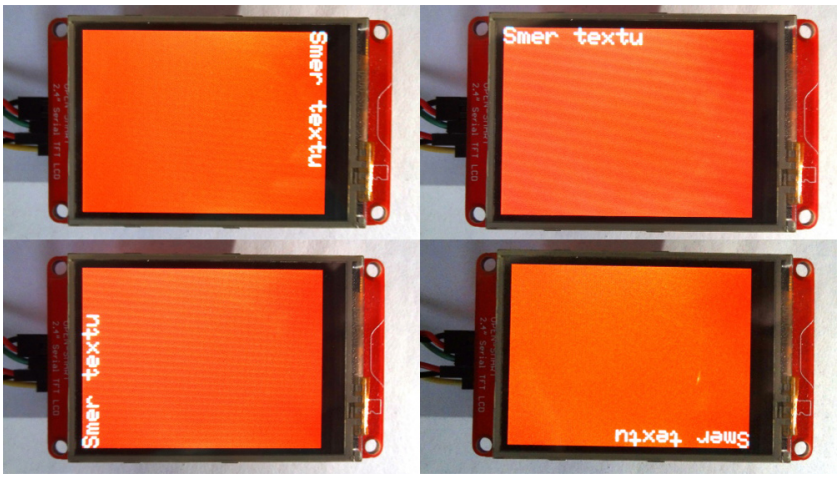
Obrázek 3.24: Příkaz FILL_SCREEN

Příklad: Zasláním příkazu *7E 04 20 F8 00 EF* se displej překreslí červenou barvou

- **SET_BACKLIGHT**
  Příkaz slouží k nastavení intenzity podsvícení. Rozsah hodnot je v rozsahu 0 - 255 (0 - 0xFF), výrobcem doporučená hodnota je 200 (0xC8). Příkaz je ve tvaru *7E 03 06 intenzita EF*

- **SET_ROTATION**
  Pomocí tohoto příkazu lze nastavit orientaci displeje. Příkaz je ve tvaru *7E 03 04 rotace EF*, kdy rotace může nabývat hodnot:

  – 00 (na obr. 3.25 vlevo nahoře)
  – 01 (výchozí, na obr. 3.25 vpravo nahoře)
  – 02 (na obr. 3.25 vlevo dole)
  – 03 (na obr. 3.25 vpravo dole)



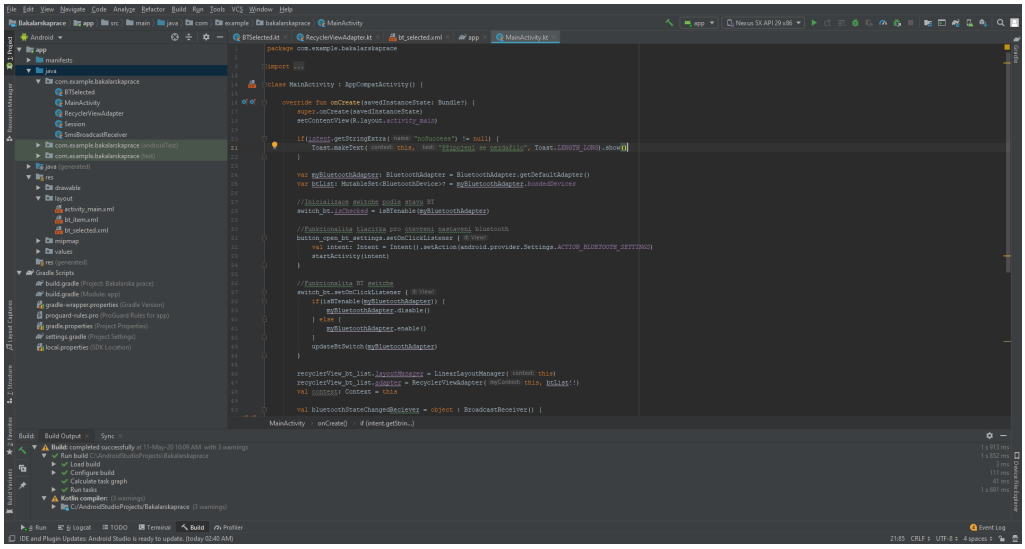
Obrázek 3.25: Příkaz SET_ROTATION

- **WRITE_BAUD**
  Tento příkaz slouží k nastavení komunikační rychlosti. Přibližně 500 ms po provedení příkazu lze zaslat *TEST* příkaz ke zjištění, zda změna proběhla v pořádku. Příkaz je ve tvaru *7E 03 40 baud EF*, kdy baud může nabývat hodnot:

  – 00 = 9600 bd/s
  – 01 = 19200 bd/s
  – 02 = 38400 bd/s
  – 03 = 57600 bd/s
  – 04 = 115200 bs/s

3.3 Vývojové prostředí Android Studio

Pro vývoj mobilní aplikace, která zajišťuje komunikaci s Bluetooth modulem jsem se rozhodl pro vývojové prostředí Android Studio (obr. 3.26) od společnosti Google, které je založeno na platformě IntelliJ IDEA a vzniklo tak ve spolupráci se společností JetBrains. Již z názvu je zřejmé, že prostředí slouží pro vývoj aplikací pro platformu Android a je dostupné zdarma na platformách Windows, Linux a macOS. Aplikace lze vyvíjet v programovacích jazycích Java nebo Kotlin.



Obrázek 3.26: Android Studio

3.4 Programovací jazyk Kotlin

Kotlin je staticky typovaný programovací jazyk vyvinutý společností JetBrains. Jazyk vznikl, protože vývojáři z JetBrains chtěli moderní programovací jazyk, který se bude vyvíjet lépe než Java, vyřeší problémy, které Java obsahuje, poběží nad JVM (Java Virtual Machine) a bude tedy plně kompatibilní s knihovnami Javy. V roce 2017 se Kotlin stal oficiálním jazykem pro Android. Jeho výhody jsou:

- **Bezpečnost:** Kotlin rozlišuje mutabilní a imutabilní proměnné. Imutabilní proměnné nelze po inicializaci změnit hodnotu, zatímco obsah mutabilní proměnné lze změnit kdykoliv. Dále nelze do proměnné uložit nulovou hodnotu. Pokud chce programátor proměnné přiřadit nulovou hodnotu, musí dát překladači explicitně najevo, že proměnná může nabývat nulové hodnoty. Potencionální chyba bude odhalena již při překladu a ne až za běhu programu[10].
- **Méně zdrojového kódu:** V porovnání s Javou, stačí k dosažení stejného výsledku mnohem méně napsaného kódu. To přináší výhodu i v čitelnosti kódu[10].
- **Interoperabilita:** Ekosystém knihoven a frameworků pro Javu je kompatibilní s Kotlinem[10]. V rámci jednoho projektu je možné využívat soubory napsané jak v Javě, tak v Kotlinu.

- **Univerzálnost:** V kotlinu lze vyvíjet mobilní aplikace, desktopové aplikace, backendové aplikace a frontendové aplikace, které poběží v prohlížeči[10].

Kapitola 4

Existující řešení

Na trhu se nachází stovky zařízení, využívající ke komunikaci technologii Bluetooth. Výhodou této technologie je především absence kabelů, nízká cena, energetická náročnost a složitost implementace.

4.1 Divoom Ditoo Pixel-Art

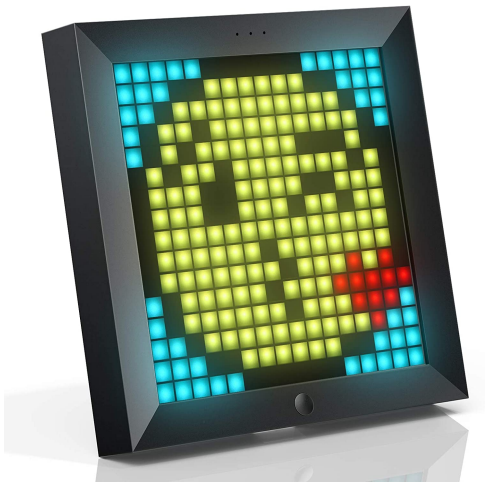
Divoom Ditoo Pixel-Art je Bluetooth reproduktor, který navíc disponuje LED maticovým displejem, slotem na microSD kartu a vestavěnou baterií. Baterie nabízí výdrž až 8 hodin a lze ji nabíjet moderním USB-C konektorem[6]. Ovládání probíhá primárně pomocí mobilního telefonu, avšak je možné využít vestavěných mechanických kláves. Tyto klávesy slouží k ovládání zvuku či displeje. Tento displej může zobrazovat různé obrazce, datum, čas nebo počasí. Herní nadšenci jistě ocení možnost zahrát si některé starší hry, např. Tetris. Reproduktor lze využívat i jako budík či časovač.

Samotné zařízení může díky svému retro stylu sloužit i jako designový prvek.

Obrázek 4.1: Divoom Ditoo Pixel-Art

4.2 Divoom Pixoo Pixel Art

Divoom Pixoo Pixel Art je rámeček s 16x16 LED displejem, který slouží k zobrazování obrázků či animací. Rámeček je pomocí Bluetooth připojen k mobilnímu telefonu, pomocí něhož lze měnit zobrazovaný obsah. Obsahem mohou být různé animace, notifikace, stopky nebo hry. Zařízení disponuje vestavěnou baterií o kapacitě 2500 mAh, kterou lze dobíjet pomocí microUSB portu[13].



Obrázek 4.2: Divoom Pixoo Pixel Art

Rámeček lze pomocí vestavěného stojánku jednoduše postavit na stůl nebo jiný nábytek.

4.3 Xiaomi Mi Band 4

Mi Band 4 je chytrý náramek, který komunikuje s mobilním telefonem pomocí Bluetooth. Náramek nabízí řadu funkcí, jako je měření kroků, spánku či srdečního tepu. V mobilním telefonu lze vést dlouhodobé statistiky z těchto funkcí. Dále náramek zobrazuje notifikace o hovorech, příchozích SMS a spoustu dalších.

Obrázek 4.3: Xiaomi Mi Band 4

Kapitola 5

Zhodnocení současného stavu

V této kapitole je zhodnocení současného stavu a na základě nastudovaných informací je navrženo následující řešení.

5.1 Zhodnocení existujících řešení

Výše uvedené existující řešení nemusí splňovat požadavky potenciálních zákazníků. Nevýhodou Divoom Ditoo je především vysoká pořizovací cena, která se pohybuje okolo 80 euro, protože obsahuje i reproduktor. Pro někoho, koho zajímá pouze displej by stačil levnější produkt bez reproduktoru. To nabízí Divoom Pixoo, ale cena se i přes absenci reproduktoru pohybuje kolem 50 euro. V případě chytrého náramku Mi Band 4 je cena příznivá (cca 25 euro), ale pokud někdo hledá stolní řešení informačního displeje, může být nevýhodou malý displej.

5.2 Zhodnocení stavu

Zadání přímo nespecifikuje konkrétní aplikaci využití bezdrátové komunikace. Rozhodl jsem se tedy pro ovládání displeje za pomoci Bluetooth modulu a mobilního telefonu. Požadavky jsou takové, že se na displeji bude zobrazovat upozornění na příchozí hovory a SMS. V klidovém stavu se bude zobrazovat aktuální datum, čas a stav baterie mobilního telefonu.

Operační systém Android při různých událostech zasílá konkrétní zprávy, ve kterých informuje o tom, co se právě stalo. Dále Android implementuje třídu Broadcast receiver, který zachytává konkrétní typ zpráv a implementuje metodu *onReceive()*, která se vykoná při zachycení zprávy. Libovolná aplikace tak může zachytávat zprávy a za základě toho vykonat akci.

Aplikace v této práci bude pomocí Broadcast receiverů zachytávat zprávy o příchozích hovorech a SMS. V případě, kdy bude zachycena zpráva o příchozím hovoru, chování metody *onReceive()* bude takové, že se na displej zašle informace o příchozím hovoru a čísle volajícího. Podobně tomu bude v případě příchozí SMS zprávy.

5.3 Návrh hardware

Po analýze dostupných technologických modulů jsem se rozhodl pro následující hardware.

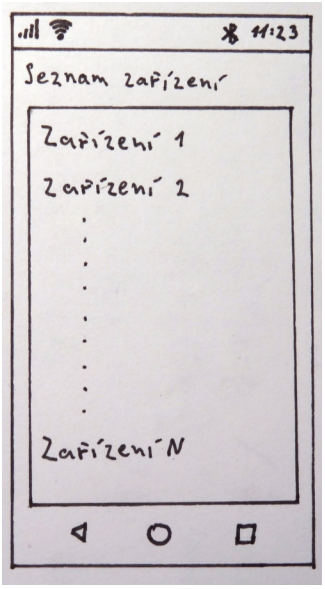
- Bluetooth modul HC-05

- LCD displej s rozhraním UART
- Převodník USB-UART

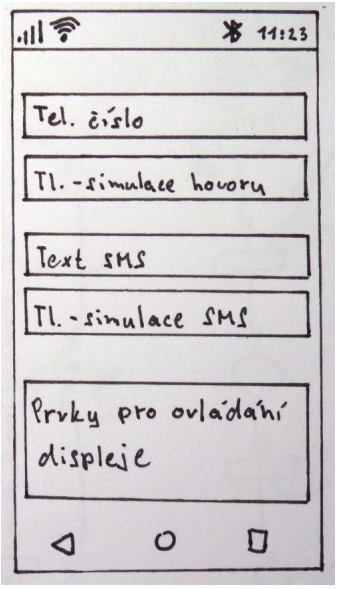
Napájení lze řešit pomocí powerbanky (externí baterie), adaptéru do sítě nebo USB portu v PC. Převodník USB-UART složil nejříve ke konfiguraci Bluetooth modulu a displeje, nyní složí k převodu napětí na 3,3 a 5 V.

5.4 Návrh uživatelského rozhraní

Návrh uživatelského rozhraní se skládá ze dvou částí. Nejdříve by mělo dojít k výběru Bluetooth zařízení, ke kterému se má aplikace připojit a druhá část bude sloužit k samotnému ovládání displeje.



Obrázek 5.1: První část návrhu UI



Obrázek 5.2: Druhá část návrhu UI

Kapitola 6

Implementace

V této kapitole je popsána implementace řešení. Při vývoji aplikace jsem postupoval od klíčových částí až po ty nejméně důležité. Základem bylo zprovoznit komunikaci s Bluetooth modulem a implementace základních funkcionalit. Doplňkové funkce nad rámec základní funkcionality byly implementovány, až když byla zprovozněna základní funkcionalita.

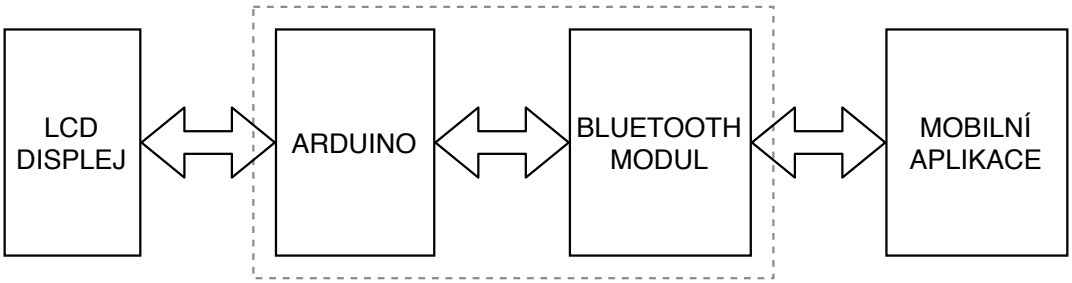
6.1 Koncepce řešení

Řešení se skládá z LCD displeje, Bluetooth modulu a mobilní aplikace. LCD displej je s Bluetooth modulem propojen pomocí UART rozhraní a mobilní aplikace s Bluetooth modulem komunikuje pomocí technologie Bluetooth. Bluetooth tedy zprostředkovává komunikační rozhraní mezi LCD displejem a mobilní aplikací.

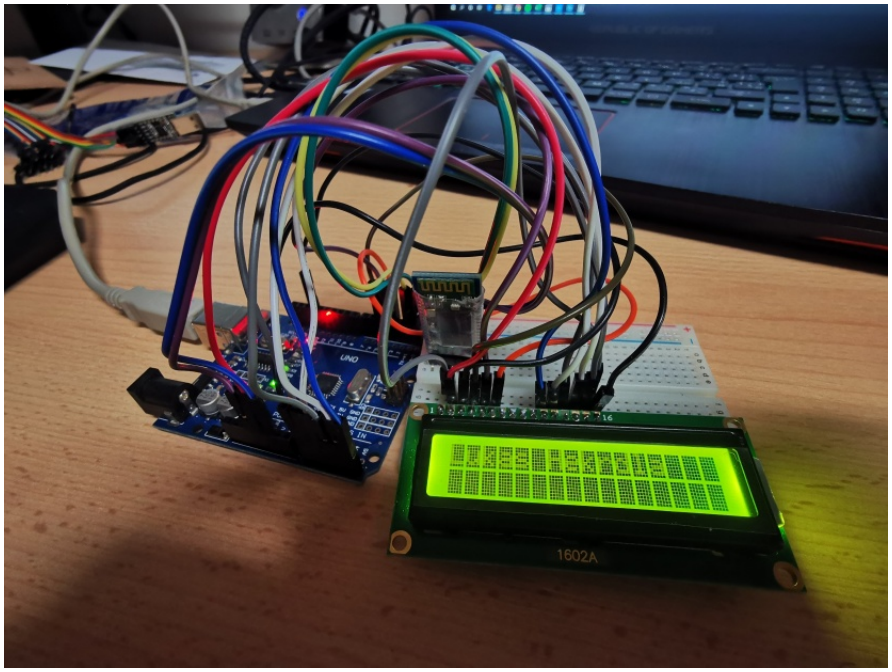
Obrázek 6.1: Blokové schéma

6.2 První řešení

První pokus o řešení byl realizován s jednoduchým dvouřádkovým displejem, kdy jeden řádek zobrazoval 16 znaků (obr. 6.3). Pro toto řešení jsem se rozhodl kvůli tomu, že na dodání displeje použitého v této bakalářské práci jsem čekal několik dní, protože byl zasílán z Číny. Tím jsem využil čas pro základní vývoj aplikace a seznámení se s vývojovým prostředím, programovacím jazykem, Bluetooth komunikací a o tom jak se mobilní aplikace vyvíjí. Tento prototyp se skládal z displeje, Bluetooth modulu HC-05 a vývojové desky Arduino Uno. Poté, co přišel z Číny výše zmíněný displej, jsem dále tento prototyp nevyvíjel.



Obrázek 6.2: Blokové schéma prvního řešení

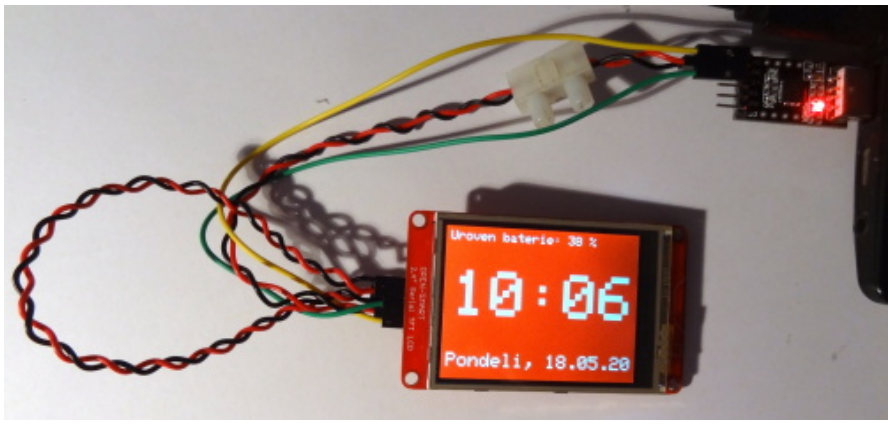


Obrázek 6.3: Prototyp řešení

6.3 Prototyp řešení

V této podkapitole je zdokumentován výsledný prototyp, který je plně funkční. Prototyp se skládá z Bluetooth modulu HC-05, LCD displeje od společnosti OPEN-SMART a převodníku USB-UART, který slouží jako napájení.

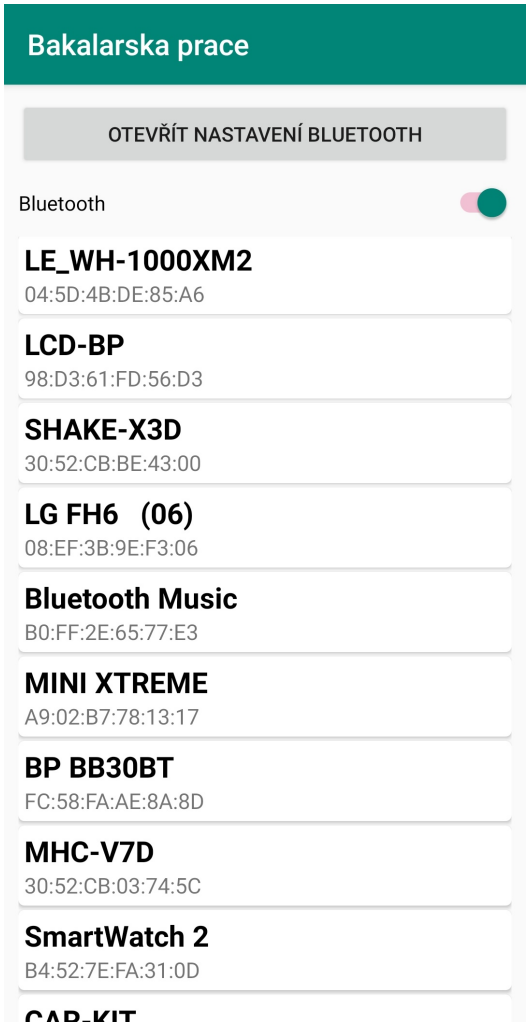
Obrázek 6.4: Blokové schéma



Obrázek 6.5: Zapojení řešení (Bluetooth modul se nachází za displejem)

6.4 Uživatelské rozhraní

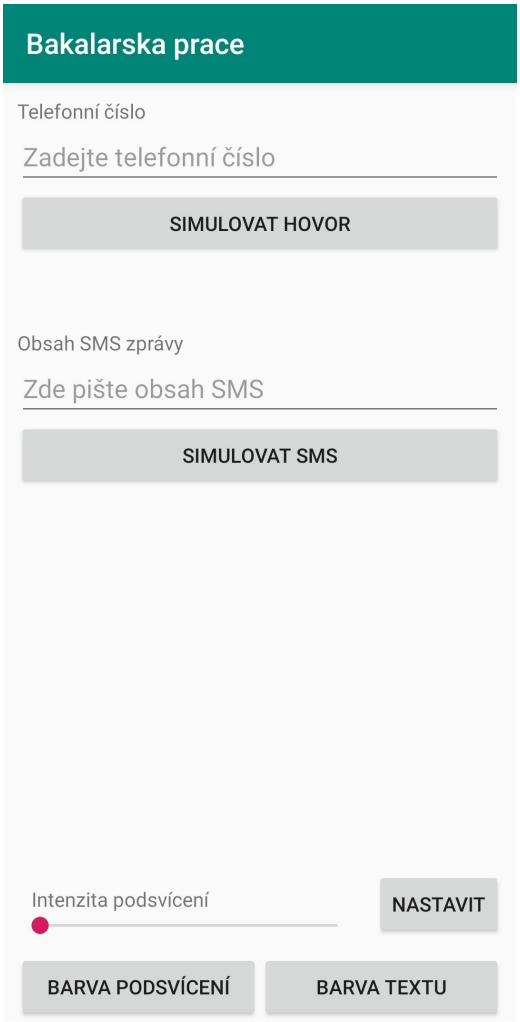
Na první obrazovce (obr. 6.6) je umístěno tlačítko pro otevření nastavení Bluetooth v operačním systému. To slouží k co nejjednoduššímu prokliku do nastavení Bluetooth v případě, že Bluetooth modul ještě nebyl k mobilnímu telefonu připojen a je třeba se s ním nejdříve spárovat. Druhým prvkem na této obrazovce je vypínač funkce Bluetooth a třetím a zároveň posledním prvkem je seznam spárovaných Bluetooth zařízení s tímto mobilním telefonem. Po kliknutí na vybrané Bluetooth zařízení začne aplikace načítat druhou obrazovku a bude se snažit navázat komunikaci se zařízením. V případě, že spojení proběhne úspěšně, načte se druhá obrazovka. Pokud ale navázání spojení nebude možné, aplikace se vrátí na první obrazovku a upozorní uživatele, že se spojení nezdařilo.



Obrázek 6.6: Úvodní obrazovka

Druhá obrazovka aplikace (obr. 6.7) je již zajímavější. Kromě toho, že zde probíhá automatické odesílání upozornění na příchozí hovory a SMS zprávy, lze hovory i zprávy simulovat. Prvním prvkem je textové pole, které slouží k vyplnění telefonního čísla, to je společné pro simulování hovorů i SMS zpráv. Pod tímto textovým polem se nachází tlačítko, které hovor odsimuluje. Dalším prvkem je textové pole pro obsah zprávy a pod ním tlačítko pro simulování SMS zprávy. V dolní části obrazovky se nachází jezdec pro nastavení intenzity podsvícení LCD displeje a hned vedle tlačítko pro potvrzení nové hodnoty. Vlevo dole je umístěno tlačítko pro nastavení barvy pozadí LCD displeje a vedle vpravo je obdobné tlačítko pro nastavení barvy textu vypisovaného na displej. V obou případech se po kliknutí na tlačítko zobrazí paleta barev[1], kde lze nastavit libovolnou barvu. Tato paleta barev je knihovou třetí strany.

[1] https://github.com/yukuku/ambilwarna



Obrázek 6.7: Obrazovka po připojení

6.5 Implementační detaily

V této podkapitole jsou popsány algoritmy, jak probíhá vypisování informací na displej.

- **Zobrazení času, data a stavu baterie**
  Celý proces vypsání aktuálního času, data a stavu baterie zajišťuje metoda, která se jmenuje *makeTime()*. Její algoritmus je následující.

  1. Ze systémových informací je získaný aktuální čas, datum a stav baterie
  2. Pomocí příkazu *SET_TEXTSIZE* je nastavena velikost písma na 10 (znak je velký 50px na šířku a 70px na výšku)
  3. Pomocí příkazu *FILL_SCREEN* je smazán obsah displeje a následně je vyplněn celý barvou
  4. Pomocí příkazu *SET_CURSOR* je nastaven kurzor na souřadnice x=15, y=70
  5. Pomocí příkazu *PRINT_CHAR_ARRAY* se je vypsán čas
  6. Pomocí příkazu *SET_CURSOR* je nastaven kurzor na souřadnice x=8, y=200

7. Pomocí příkazu *SET_TEXTSIZE* je nastavena velikost písma na 3 (znak je velký 15px na šířku a 21px na výšku)
8. Pomocí příkazu *PRINT_CHAR_ARRAY* je vypsán datum
9. Pomocí příkazu *SET_CURSOR* je nastaven kurzor na souřadnice x=8, y=8
10. Pomocí příkazu *SET_TEXTSIZE* je nastavena velikost písma na 2 (znak je velký 10px na šířku a 14px na výšku)
11. Pomocí příkazu *PRINT_CHAR_ARRAY* je vypsán stav baterie

Operační systém Android zasílá každou minutu zprávu *ACTION_TIME_TICK*. Pomocí broadcast receiveru je tato zpráva v aplikaci zachycena a chování metody *onReceive()* je implementováno tak, že se zavolá výše zmíněná metoda *makeTime()*.

Obrázek 6.8: Domovská obrazovka

- **Upozornění na příchozí hovor**
  V operačním systému Android existuje třída *PhoneStateListener*, která slouží k naslouchání o změnách stavu hovorů, signálu, apod. Jednou z metod této třídy je metoda *onCallStateChanged(state, phoneNumber)*. Moje implementace této metody je následující:

  if *state == PHONE_STATE_RINGING* **then**
  1. Pomocí příkazu *SET_TEXTSIZE* je nastavena velikost písma na 4 (znak je velký 20px na šířku a 28px na výšku)
  2. Pomocí příkazu *FILL_SCREEN* je smazán obsah displeje a následně je vyplněn celý barvou
  3. Pomocí příkazu *SET_CURSOR* je nastaven kurzor na souřadnice x=15, y=70
  4. Pomocí příkazu *PRINT_CHAR_ARRAY* je vypsán řetězec "NOVY HOVOR"
  5. Pomocí příkazu *SET_CURSOR* je nastaven kurzor na souřadnice x=0, y=150
  6. Pomocí příkazu *PRINT_CHAR_ARRAY* je vypsáno *phoneNumber*

  else then
  7. Volání metody *makeTime()*

Obrázek 6.9: Upozornění na příchozí hovor

- **Upozornění na SMS zprávu**
  V momentě, když mobilní telefon obdrží SMS zprávu, je operačním systémem zaslaná zpráva *SMS_RECEIVED_ACTION*. Pomocí broadcast receiveru je tato zpráva zachycena a chování metody *onReceive()* je implementováno následovně:

  1. Ze zaslané zprávy je získáno číslo odesílatele a obsah SMS zprávy a obojí je uloženo do textového řetězce
  2. Z textového řetězce se odstraní diakritika
  3. Pomocí příkazu *FILL_SCREEN* je smazán obsah displeje a následně je vyplněn celý barvou
  4. Pomocí příkazu *SET_TEXTSIZE* je nastavena velikost písma na 4 (znak je velký 20px na šířku a 28px na výšku)
  5. Pomocí příkazu *PRINT_CHAR_ARRAY* je vypsán textový řetězec ve formátu "číslo: obsah SMS zprávy"

Obrázek 6.10: Upozornění na SMS zprávu

6.6 Testování

Testování probíhalo za účelem ověření funkčnosti a stability výsledného produktu. Provedené testy byly následující:

- **Testování stability aplikace**
  Testování probíhalo tak, že jsem manuálně testoval stabilitu aplikace. V momentě, kdy se aplikace po nějakém kliknutí nekorektně ukončila (např. výběr nedostupného Bluetooth zařízení) snažil jsem se najít příčinu, problém odstranit a znovu otestovat.

- **Testování komunikace s displejem**
  V případě, že se na displeji nezobrazilo, to co by mělo, využíval jsem widgetu Toast[2], který v bublině zobrazí text. Takto jsem vypisoval obsah proměnných a zjišťoval zda je chyba nastala již v aplikaci nebo až při Bluetooth komunikaci.

  Optimální by bylo, pokud by jednotlivé příkazy za sebou navazovaly s co nejmenším zpožděním, ale v takovém případě bylo vypisování informací na displej nestabilní a spousta příkazů se nevykonala. Postupným zvyšováním časových intervalů mezi jednotlivými příkazy se spolehlivost komunikace zvyšovala. Po zvýšení na 230 milisekund byl systém stabilní několik hodin, proto časový interval mezi příkazy je nastaven na tuto hodnotu, přestože překreslení displeje trvá cca 0,7 - 2,3 sekundy.

- **Upozornění na příchozí hovor**
  Testování této funkcionality probíhalo tak, že z druhého mobilního telefonu jsem se

[2] https://developer.android.com/reference/kotlin/android/widget/Toast

pokoušel navázat telefonní hovor a pozoroval, zda se informace o příchozím hovoru a volajícím na displeji korektně zobrazují.

- **Upozornění na příchozí SMS**
  Z druhého mobilu jsem si zasílal SMS zprávy a pozoroval, zda se upozornění zobrazí a zda se číslo odesílatele a obsah zprávy zobrazen korektně.

- **Zobrazování času, data a stavu baterie**
  Zde jsem vypozoroval, že pokud má telefon rozsvícený displej, tyto informace se zobrazují korektně a každou minutu se aktualizují. Takto je tomu i když je displej mobilního telefonu zhasnutý, ale telefon je nabíjen. Ovšem v situaci, kdy je displej zhasnutý a telefon běží na baterii, informace se aktualizují pouze přibližně jednou za 5 minut. Osobně si myslím, že operační systém při zhasnutém displeji zasílá zprávy o změně času v delších intervalech z důvodu úspory baterie.

Pro testování jsem používal můj osobní mobilní telefon Huawei P30 s operačním systémem Android 10. Jako druhý testovací mobilní telefon sloužil dnes již starší Samsung Galaxy S3. Ideální by bylo produkt testovat na desítkách zařízení s různými verzemi operačního systému, ale tolika zařízeními nedisponuji.

Kapitola 7

Závěr

Cílem práce bylo navrhnout a demonstrovat bezdrátové ovládání elektroniky mobilním zařízením za pomoci Bluetooth technologie. Tento cíl byl splněn.

Nejdříve jsem prostudoval Bluetooth moduly a jejich možnosti využití v mobilních zařízeních. Dalším krokem bylo vytipovat vhodnou aplikaci jak modul využít, načež jsem se rozhodl, že pro demonstraci bude sloužit LCD displej, který bude upozorňovat na příchozí hovory a SMS zprávy. V tuto chvíli jsem se musel rozhodnout, jak toto implementovat. Nejdříve mě napadlo, že bych mohl využít Arduino, kdy by Bluetooth modul zprostředkovával komunikaci mezi mobilní aplikací a Arduinem a program nahraný v Arduinu by následně zajišťoval vypisování informací na displej. Poté jsem se rozhodl, že obstarám displej s rozhraním UART, který bude možné ovládat přímo. Tohle řešení přináší výhodu především ve své kompaktnosti, kdy vedle displeje je pouze Bluetooth modul a převodník USB-UART, který slouží jako napájecí část, protože poskytuje současně 5 V pro displej a 3,3 V pro Bluetooth modul.

Díky této práci jsem se seznámil s programovacím jazykem Kotlin, o kterém jsem věděl již dříve, ale až nyní jsem se jej začal učil. Dále mi tato práce dala povědomí o tom, jak se vyvíjí mobilní aplikace a přiučil jsem principům Bluetooth komunikace.

Při dalším vývoji bych chtěl lépe využít potenciálu, který displej nabízí. Například by bylo možné z geometrických obrazců (čáry, kružnice, obdélníky, ...) skládat komplexní grafické prvky, které by přidaly na uživatelské přívětivosti. Další možnost je grafické prvky navrhnout v nějakém grafickém programu, uložit na microSD kartu, vložit do displeje a následně vykreslovat (např. ikona baterie).

Literatura

[1] *IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 15.1a: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPAN): IEEE Std 802.15.1-2005 (Revision of IEEE Std 802.15.1-2002)*. New York, N.Y: [b.n.], červen 2005. 53, 437 - 459 s. ISBN 0-7381-4708-7.

[2] *HC-05 Bluetooth module*. 1. vyd. China: ITead Studio, červenec 2010 [cit. 2020-03-02]. Rev. v1.0. Dostupné z: http://www.electronicaestudio.com/docs/istd016A.pdf.

[3] *AT Commands Reference Guide* [online]. 80000ST10025a. England: Telit Communications S.p.A., září 2016 [cit. 2020-05-21]. Rev. 24 – 2016-09-07.

[4] *Serial TFT User manual* [online]. China: OPEN-SMART, květen 2018 [cit. 2020-03-19]. Ver. v1.0. Dostupné z: https://drive.google.com/drive/folders/1ReWai0mdEofMkcXbM3ROxDt2Ya159-DG.

[5] Bluetooth (INFORMACE): verze, dosah, frekvence a protokoly. *Alza.cz* [online]. Březen 2019 [cit. 2020-05-10]. Dostupné z: https://www.alza.cz/slovnik/bluetooth-art12370.htm.

[6] Chen, M. *Bluetooth Speaker Ditoo* [online]. China: Shenzhen Divoom Technology Co., LTD., únor 2019 [cit. 2020-05-28]. Dostupné z: https://fccid.io/A8I-DITOO/User-Manual/User-maunal-4433851.pdf.

[7] Kovařík, D. Bluetooth – modrozub pod drobnohledem (vědecké okénko). *Mobilizujeme.cz* [online]. Praha: [b.n.], prosinec 2011 [cit. 2020-05-07]. Dostupné z: https://mobilizujeme.cz/clanky/bluetooth-modrozub-pod-drobnohledem-vedecke-okenko.

[8] Kočí, M. Bluetooth 5 zdvojnásobí přenosovou rychlost a nabídne čtyřikrát vyšší dosah. *PCTuning.cz* [online]. Praha: EMPRESA MEDIA, a.s, červen 2016 [cit. 2020-05-10]. ISSN 1214-0201. Dostupné z: https://pctuning.tyden.cz/index.php?option=com_content&view=article&id=41493&catid=1&Itemid=57.

[9] Krompolc, T. Bluetooth je tu s námi 20 let: vše, co o této technologii potřebujete vědět. *SMARTmania s.r.o.* [online]. Únor 2019 [cit. 2020-05-09]. ISSN 1801-3066. Dostupné z: https://smartmania.cz/bluetooth-je-tu-s-nami-20-let-vse-co-o-teto-technologii-potrebujete-vedet/.

[10] Marcin Moskala, I. W. *Android Development with Kotlin: Learn Android application development with the extensive features of Kotlin*. 1. vyd. Birmingham: Packt Publishing Ltd., srpen 2017. ISBN 978-1-78712-368-7.

[11] McDermott-Wells, P. What is Bluetooth? *IEEE Potentials*. 2005, sv. 23, č. 5, s. 33–35.

[12] Padgette, J., Scarfone, K. a Chen, L. *Guide to Bluetooth Security: Recommendations of the National Institute of Standards and Technology (Special Publication 800-121 Revision 1)*. North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2012. ISBN 147816896X.

[13] Yao, Y. *Pixel Photo Fram* [online]. China: Shenzhen Divoom Technology Co., LTD., srpen 2018 [cit. 2020-05-28]. Dostupné z: https://fccid.io/A8IPIXOO/User-Manual/user-manual-4010825.pdf.

[14] Čížek, J. Bluetooth 4.2 je hotový. Lépe propojí internet věcí. *Živě.cz* [online]. CZECH NEWS CENTER a.s., prosinec 2014 [cit. 2020-05-10]. Dostupné z: https://www.zive.cz/bleskovky/bluetooth-42-je-hotovy-lepe-propoji-internet-veci/sc-4-a-176382/default.aspx.

Příloha A

Obsah paměťového média

Na paměťovém médiu se nachází:

- Bakalarskaprace - složka s projektem v Android studiu
- Zdrojove_kody - složka s mnou psanými zdrojovými kódy
- ovladani_displeje.apk - mobilní aplikace
- Displej_dokumentace.pdf - dokumentace k LCD displeji
- Bakalarska_prace.pdf - tento dokument
- Bakalarska_prace_latex - složka se zdrojovými texty pro tento dokument