VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

MONITOROVÁNÍ SPOTŘEBY ELEKTRICKÉ ENERGIE S VYUŽITÍM BLUETOOTH

ENERGY CONSUMPTION MONITORING USING BLUETOOTH

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **ONDŘEJ BARTOŇ**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. ZDENĚK VAŠÍČEK, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačových systémů (UPSY) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Bartoň Ondřej**
Program: Informační technologie
Název: **Monitorování spotřeby elektrické energie s využitím Bluetooth**
Energy Consumption Monitoring Using Bluetooth
Kategorie: Vestavěné systémy

Zadání:

1. Seznamte se s technologií Bluetooth Low Energy (BLE), zaměřte se zejména na způsob přenosu naměřených veličin a dále na řešení dostupná na trhu vhodná k realizaci měřicího bodu vybaveného technologií BLE jako je např. platforma ESP32, DA145XX, NRF528XX apod.
2. Navrhněte zařízení, které bude měřit spotřebovanou elektrickou energii na základě monitorování impulsního výstupu elektroměru. Při návrhu se snažte o maximalizaci životnosti baterie tohoto zařízení. S využitím vhodné platformy (zařízení s OS Android, RPI, ESP32 apod.) dále navrhněte přístupový bod, který bude sloužit k uchování naměřených informací, jejich vizualizaci a analýze.
3. Navržené zařízení a systém implementujte formou prototypu, např. s využitím existujících modulů a vývojových desek.
4. Vyhodnoťte a diskutujte parametry navrženého řešení a možnosti dalšího rozšíření.

Literatura:

- Dle pokynů vedoucího.

Pro udělení zápočtu za první semestr je požadováno:

- Splnění bodů 1 a 2 zadání.

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

Vedoucí práce: **Vašíček Zdeněk, doc. Ing., Ph.D.**
Vedoucí ústavu: Sekanina Lukáš, prof. Ing., Ph.D.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 30. října 2020

Abstrakt

Cílem této práce je navrhnout a implementovat prototyp systému na monitorování spotřeby elektrické energie. Měření je prováděno na impulsním výstupu elektroměru s využitím IoT mikrokontroléru Dialog Semiconductor DA14531. Naměřené hodnoty jsou pomocí technologie Bluetooth Low Energy pravidelně odesílány na počítač Raspberry Pi, kde jsou dále zpracovány a vizualizovány.

Abstract

The purpose of this thesis is to design and implement a system used for monitoring of energy consumption. The measurement is done on pulse output of the electricity meter using IoT microcontroller Dialog Semiconductor DA14531. The Measured values are send to computer Raspberry Pi using Bluetooth Low Energy. Here the data is processed and visualized.

Klíčová slova

Bluetooth Low Energy, Bluetooth LE, DA14531, Raspberry Pi

Keywords

Bluetooth Low Energy, Bluetooth LE, DA14531, Raspberry Pi

Citace

BARTOŇ, Ondřej. *Monitorování spotřeby elektrické energie s využitím Bluetooth.* Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. Ing. Zdeněk Vašíček, Ph.D.

Monitorování spotřeby elektrické energie s využitím Bluetooth

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Zdeňka Vašíčka. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Ondřej Bartoň
12. května 2021

Poděkování

Tímto bych chtěl poděkovat vedoucímu práce, panu doc. Ing. Zdeňku Vašíčkovi, Ph.D. za jeho oborné vedení práce, cenné rady a zapůjčení vývojových modulů a součástek.

Obsah

Kapitola 1

Úvod

Cílem této práce je navrhnout a implementovat zařízení sloužící k monitorování spotřeby elektrické energie. S využitím technologie Bluetooth Low Energy naměřená data posílá na počítač Raspberry Pi 4. S využitím webové aplikace jsou uložená data prezentována uživateli.

Kapitola 2

Bluetooth Low Energy (BLE)

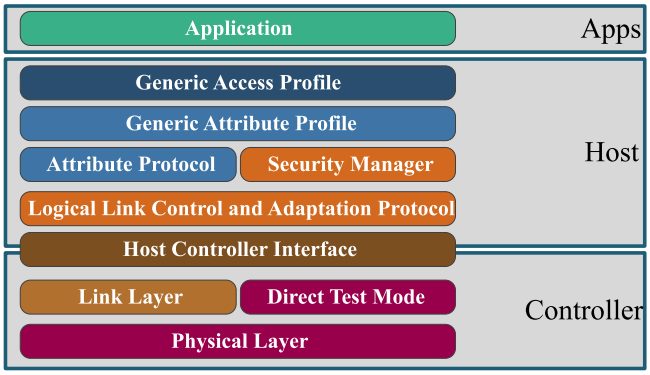
Bluetooth je bezdrátový standard pro přenos dat na krátkou vzdálenost mezi dvěma i více zařízeními. Původně bylo Bluetooth vytvořeno jako bezdrátová náhrada za sériový kabel, ale jeho využití je dnes mnohem širší – využívá se v počítačích, mobilních telefonech, bezdrátových sluchátkách, v klávesnicích, nositelné elektronice nebo například v IoT (Internet of Things). Právě v IoT je kladen velký důraz na spotřebu energie a proto je často využívána technologie Bluetooth Low Energy, která je navržena na provoz v systémech, kde není potřeba přenášet velké objemy dat, ale důraz je kladen především na nízkou spotřebu. Nízké spotřeby je dosaženo především omezením doby aktivního používání – přenos dat probíhá periodicky v předem stanovém intervalu.

Technologie Bluetooth classic a Bluetooth Low Energy spolu nejsou kompatibilní, ale většina mobilních telefonů a počítačů od verze Bluetooth 4.0 podporuje obě tyto technologie.

Informace o Bluetooth v následujících kapitolách převzaty z [2], [9], [10].

2.1 Architektura BLE

Protokol Bluetooth Low Energy (dále jen Bluetooth LE nebo BLE) se skládá ze 2 částí: *Host* a *Controller*. Obě části se dále skládají z dalších částí, jak lze vidět obrázku 2.1



Obrázek 2.1: **Bluetooth Low Energy Protocol Stack** Převzato z [6]

2.1.1 BLE Protocol Stack – Hostitel (Host)

Nižší vrstva BLE protokolu - s touto vrstvou vývojář pracuje pouze nepřímo skrze modul *Host*.

Fyzická vrstva (PHY)

Fyzická vrstva (Physical Layer) je zodpovědná za vytváření a příjem rádiového signálu. BLE využívá bezlicenční pásmo 2,4 GHz ISM – stejně jako Classic Bluetooth a technologie WiFi. Frekvenční pásmo 2.4 – 2.483 GHz je rozděleno na 40 kanálů s rozestupy 2 MHz.

Od verze 5 může být na čipu více PHY vrstev. *Coded PHY* snižuje chybovost přenosu a tím zvyšuje dosah, snižuje symbolovou rychlost, *LE 2M PHY* zvyšuje rychlost ale mírně snižuje dosah. [17]

Tabulka 2.1: Srovnání BLE PHY vrstev

	Bitová rychlost	Symbolová rychlost	BLE verze	popis
LE 1M PHY	1 Mb/s	1 Msym/s	4	
Coded PHY	1 Mb/s	500 Ksym/s nebo 125 Ksym/s	5	zvýšený dosah
LE 2M PHY	2 Mb/s	2 Msym/s	5	zvýšená rychlost

Linková vrstva (LL)

Linková vrstva (Link Layer) je část BLE, která přímo pracuje s fyzickou vrstvou (PHY). Vytváří a udržuje spojení, skenuje a vysílá advertising pakety. Již zmiňovaných 40 kanálů je rozděleno na 37 kanálů určených na přenos dat a další 3 tzv. *advertising* kanály.

Advertising kanály slouží (resp. jsou na nich vysílány advertising pakety) k objevování zařízení (device discovery), navazování spojení a vysílání dat (broadcasting) mimo navázané spojení.

Datové kanály jsou využívány pro obousměrnou komunikaci mezi zařízeními a od BLE 5 (advertising extension) k rozšíření obsahu dat vysílaných jako advertising.

Ke snížení interference, která je v pásmu 2.4 Ghz častá, používá BLE metodu AFH (Adaptive Frequency Hopping). Při vysílání touto metodou jsou data rozdělena na pakety. Kanál, na kterém je paket odesílán se mění 1600krát za sekundu a je vybrán podle deterministického algoritmu, který znají obě strany. Pokud je kanál zarušený a příjemce paket nepotvrdí, paket je poslán znovu na jiném kanálu. Zarušený kanál je dočasně označen jako nespolehlivý a není při komunikaci využíván.

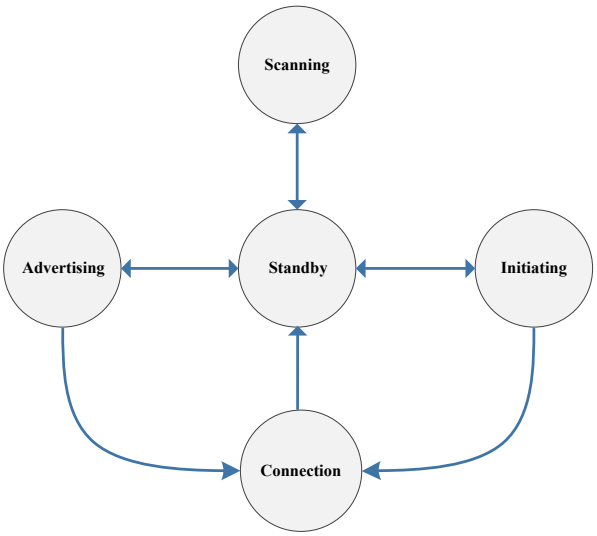
Linková vrstva je stavový automat s 5 různými stavy (viz obrázek 2.2). [15]

Host controller interface (HCI)

HCI je vrstva, která zajišťuje přenos příkazů a odpovědí mezi vrstvami hostitele (vyšší vrstvy) a kontroléru (PHY a LL). Vrstvy mohou existovat na rozdílných MCU (Micro controller unit), pak je přenos většinou realizován sériově přes UART či SPI nebo mohou existovat v jednom společném MCU.

2.1.2 BLE Protocol Stack – Kontrolér (Controller)

Horní vrstva BLE protokolu - vývojář aplikace s ní většinou pracuje pomocí knihoven poskytnutých výrobcem Bluetooth čipu.



Obrázek 2.2: **BLE Link-Layer States** Převzato z [7]

Logical link control and adaptation protocol (L2CAP)

L2CAP je vrstva, která slouží jako multiplexer – přijímá několik protokolů z vyšších vrstev a zapouzdří je do obyčejného BLE paketu, které následně předá linkovací vrstvě a pokud je paket příliš velký, rozdělí na více menších. Zajišťuje i opačný směr, tedy poskládání BLE paketů a předání dat vyšší vrstvě. Hlavní protokoly, které takto zpracovává a směruje jsou Attribute Protocol (ATT) a the Security Manager Protocol (SMP).

Security manager (SM)

SM je protokol, jehož cílem je párování zařízení a generování a distribuce klíčů. Ostatním vrstvám poskytuje funkce pro vytvoření zabezpečených spojení a pro bezpečnou výměnu dat s jiným zařízením.

Attribute Protocol (ATT)

ATT je jednoduchý protokol typu server/klient, který definuje pravidla a práva pro přístup k atributům na zařízení. Atributy jsou jednotky dat, např. poloha nebo teplota, spolu s lokálním handle, UUID a právy. V průběhu komunikace se role zařízení může měnit nebo mít i obě role. Pro získání dat zašle klient požadavek na server a ten poté pošle data jako odpověď. Server také může vyslat tzv. indikaci nebo notifikaci, při které server iniciuje přenos dat ze serveru ke klientovi. Atributy jsou na zařízení uloženy v jednoduché datové struktuře klíč:hodnota.

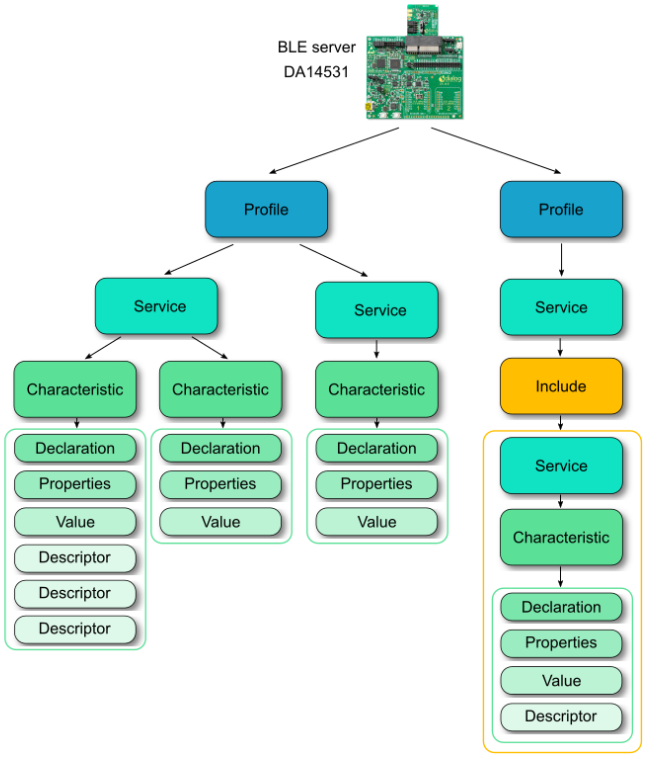
Generic Attribute Profile (GATT)

GATT dále rozšiřuje protokol ATT, přidává hierarchické uspořádání hodnot a detailně popisuje, jak budou data (atributy) přenášené. Nutností je již vytvořené spojení mezi zařízeními.

Zavádí také nové pojmy:

- Charakteristika (Characteristic) – Jedna hodnota, její vlastnosti a deskriptory
- Služba (Service) – Skupina charakteristik
- Deskriptor (Descriptor) – Metadata popisující charakteristiku
- Profil (Profile) – Předdefinované kolekce služeb (např. Heart Rate Profile)

Ke komunikaci mezi dvěma zařízeními tak při použití předdefinovaným profilů není potřeba vytvářet nové profily. Seznam standardizovaných profilů spravuje organizace Bluetooth SIC [1].



Obrázek 2.3: **Profily, služby a charakteristiky** Převzato z [8]

Generic Access Profile (GAP)

GAP je framework, který definuje, jak spolu mohou BLE zařízení komunikovat. Slouží k propagování zařízení (advertising), device discovery, otevírání a správě spojení a broadcastu. GAP definuje pro zařízení role a s nimi požadavky, které zařízení musí splňovat:

- **Broadcaster** – Pouze vysílá advertising eventy a broadcastové data
- **Observer** – Pouze přijímá advertising eventy a broadcastové data.

- **Peripheral** – Vždy slave zařízení, připojitelné a vysílá advertising. Je to jednoduché zařízení používající jedno připojení k zařízení s rolí Central.
- **Central** – Vždy master zařízení, nevysílá advertising. Je to zařízení, které má na starost vytváření a udržování spojení s zařízeními s rolí Peripheral.

Zařízení může podporovat více rolí, ale v jednu chvíli může být aktivní pouze jedna role. GAP také definuje následující režimy pro vyhledávání, spojení a párování.

- **Connectable**: Zda může navázat spojení
- **Discoverable**: Zda může být zobrazeno a vysílá advertising data
- **Bondable**: Zda se dá vytvořit dlouhodobé spojení

Převzato z [15].

Kapitola 3

Měření spotřeby elektrické energie

V následující kapitole jsou popsány způsoby měření spotřeby elektrické energie a stručně jejich výhody a nevýhody.

3.1 Měření pulzního výstupu

U většiny elektroměrů se nabízí měření el. energie pomocí pulzního výstupu. Jeden pulz odpovídá pevně danému množství spotřebované el. energie - většinou 1000 – 10000 pulzů na 1kWh. Spotřebu ve Wh vypočítáme jako počet impulzů za hodinu děleno počet Wh na jeden impulz. Nabízí se dvě varianty měření:

- **Detekce pulzů na elektroměru**

  Tuto variantu lze použít, pokud má elektroměr pulzní výstup. Při každém pulzu zaznamenáme tuto událost např. pomocí připojeného kit Arduino. Nevýhodou této varianty je možný neexistující výstup, nebo nespolupráce dodavatele energie. Také je nebezpečnější, protože výstup bývá často umístěn vedle „živých“ drátů.

- **Optické počítání pulzů**

  Tato varianta využívá světelného pulzu z LED diody. Při každém bliknutí je světelným senzorem zaznamenána změna intenzity světla a informace zpracována opět např. na připojeném Arduinu. Výhodou je, že lze použít na každém elektroměru s LED pulsním výstupem. Nevýhodou je teoreticky menší přesnost měření.

3.2 Měření chytrým elektroměrem

Novější elektroměry už někdy bývají vybavené technologií Bluetooth či jiným IoT protokolem. Zde může nastat problém, pokud výrobce použil vlastní proprietární protokol a nepublikoval žádnou aplikaci na zpracování dat. V tom případě je nutné použít některou z předchozích metod nebo si vytvořit vlastní aplikaci a případně použít reverzní inženýrství k přečtení posílaných dat.

V kapitole byly využity informace z [14].

Kapitola 4

Návrh systému

Cílem je navrhnout systém na měření a vizualizaci spotřeby elektrické energie. Systém obsahuje dvě zařízení, senzor a server. Senzor měří optickým senzorem pulzní výstup LED diody, a naměřená data odesílá přes Bluetooth Low Energy. U senzoru je kladen důraz na maximální výdrž baterie. Je tedy žádoucí, aby byl systém většinu času v energeticky nenáročném režimu spánku. Aktivní bude pouze při bliknutí LED a při odesílání dat. Server naopak bude aktivní nepřetržitě a bude přijímat data odeslaná z klienta. Napájen bude z el. sítě.

4.1 Výběr hardware

Klient

Při výběru hardware pro senzor byl kladen důraz především na energetickou náročnost SoC. Po srovnání spotřeby SoC ESP32 a spotřeby SoC DA14531 (viz tabulka 4.1) jsem se z důvodu nižší spotřeby rozhodl použít vývojový kit DA14531 SmartBond TINY™ Development Kit Pro (obrázek 4.1).

Vývojový kit má GPIO piny vyvedené do speciálního konektoru mikroBUS, do kterého lze zapojit například mikroBoard Ambient 2 click. To výrazně usnadňuje práci při vytváření prototypu. MikroBoard je osazený senzorem světla OPT30001 od firmy Texas Instruments. (viz obrázek 4.1).

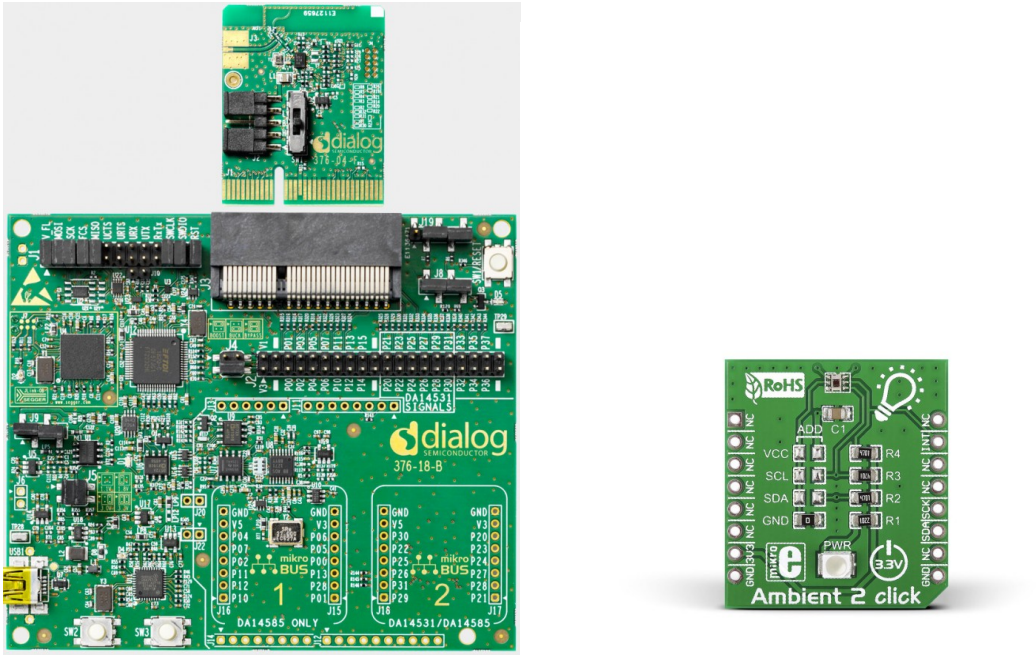
Tabulka 4.1: Srovnání spotřeby ESP32 a DA14531, Zdroj: [5], [12]

	CPU aktivní	CPU v režimu spánku
ESP32	160–260 mA	10 mA
DA14531	0.42 mA	1.8 μA

Server

Jako server jsem si zvolil počítač Raspberry Pi 4 Model B. Již v základu podporuje BLE a zároveň je i dostatečně výkony na provozování databáze a webového serveru.

Na serveru poběží webový server, databáze InfluxDB a vizualizaci bude zajišťovat aplikace Grafana. Webová stránka bude kromě vizualizace dat obsahovat také administrační část, kde půjde spravovat přidané senzory, měnit počet impulsů na 1 kWH a měnit horní a dolní hranici pro přerušení od světelného senzoru.

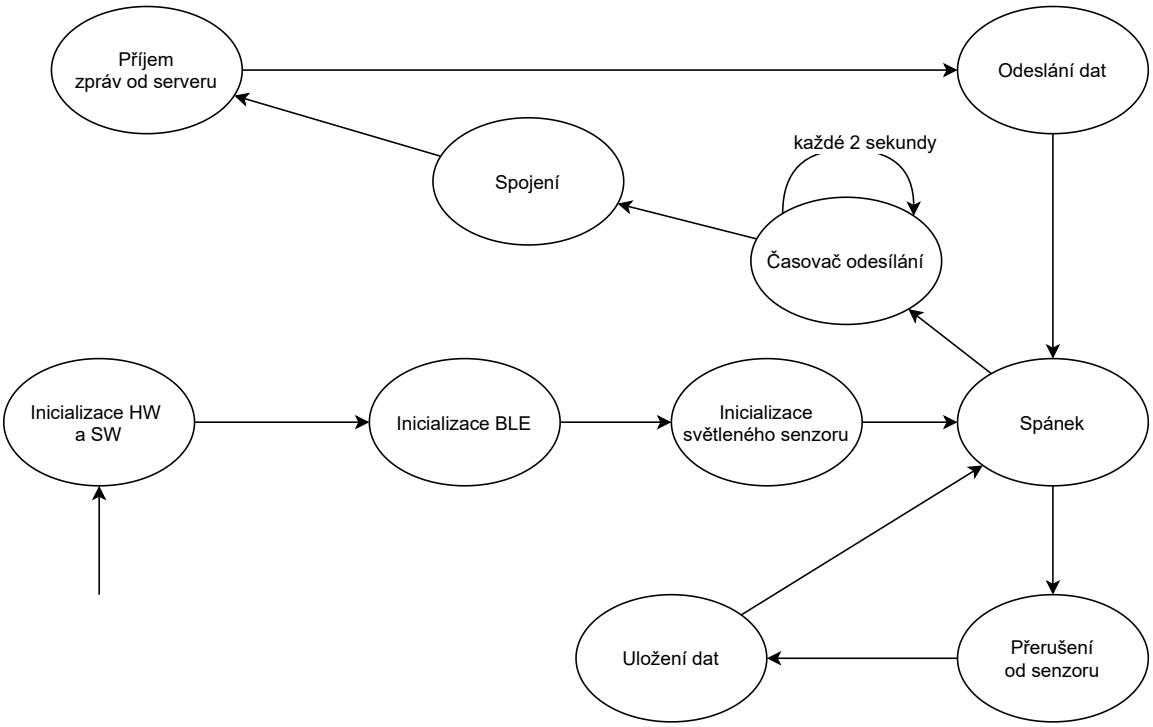


Obrázek 4.1: **DA14531 SmartBond TINY™ Development Kit Pro** a **Ambient 2 click** Převzato z [11], [13]

4.2 Chování senzoru

Na obrázku 4.2 je vidět stavový diagram chování senzoru.

- **Inicializace HW a SW** – interní procesy MCU
- **Inicializace BLE** – nastavení BLE advertising vysílání, aby se mohlo vytvořit spojení se serverem
- **Inicializace senzoru světla** – připojení přes protokol I2C, nastavení horního a dolního registru pro přerušení, zapnutí přerušení
- **Spánek** – cílem je maximalizovat dobu spánku a snížit tak spotřebu
- **Přerušení od senzoru** – zavolání obslužní funkce
- **Uložení dat** – čas přerušení je uložen do datové struktury
- **Časovač odesílání** – RTC časovač, který bude spínat přibližně každé 2 sekundy
- **Navázání spojení** – se serverem
- **Příjem zpráv od serveru**
- **Odeslání dat** – odeslání časů, kdy došlo k přerušení a resetování struktury

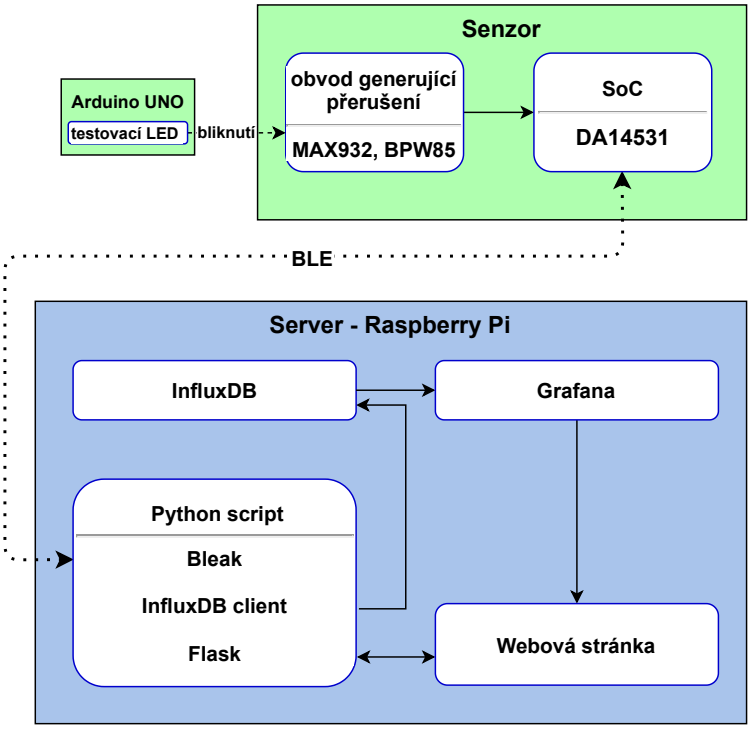


Obrázek 4.2: **Stavový diagram chování senzoru**

Kapitola 5

Použité technologie

V této kapitole jsou popsány technologie, součástky, projekty a knihovny použité při vývoji systému pro měření spotřeby elektronice energie. Při výběru těchto částí byl kladen důraz především na dostupnost, cenu a spotřebu součástek a také na vhodnost použití komponentů pro daný učel. Na obrázku 5.1 jsou zobrazeny jednotlivé součásti systému. V oddílu 5.1 je popsána architektura DA14531 a také základní informace o vývoji aplikací na tento SoC.



Obrázek 5.1: **Použité technologie**

5.1 SmartBond DA14531

SmartBond DA14531 je relativně nový SoC[1] od firmy Dialog Semiconductor určený primárně pro použití v IoT. Jeho hlavními přednostmi jsou rozměry pod 2x3 mm, technologie Bluetooth Low Energy verze 5.1 a nízká spotřeba spolu s optimalizací na provoz systému na baterie s napětím v rozpětí mezi 1.1V a 3.3V. DA14531 se obsahuje 32-bitové armové jádra *Cortex M0+*, dále 48 MB paměti RAM, 1 MB flash paměti a také OTP paměť.[2]

DA14531 dále disponuje z 9 GPIO porty, 2 UART linkami a rozhraními pro komunikaci protokoly SPI a I2C. Firmware BLE obsahuje všechny používané protokoly a standardy, jako je např. GAP, ATT, GATT, L2CAP, ale i vlastní protokol, jako je Dialog Serial Port Service (DSPS), emulující komunikaci přes sériovou linku UART. Výrobce udává, že SoC má široké spektrum uplatnění od zdravotnických pomůcek a měřiče zdraví přes majáky a detektory vzdálenosti až po využití v industriální sféře.

DA14530/1 Development Kit-Pro

Pro vývoj produktů na tomto systému nabízí výrobce vývojářské kity. Konkrétně jsou to *DA14530/1 Development Kit-Pro* (viz obrázek 4.1 na straně 10) a *DA14531 Development Kit-USB*. Oba kity umožňují připojení vývojové desky k počítači přes rozhraní USB, ale první zmíněný, *DP PRO*, toho nabízí podstatně více. Zmínit můžeme například měření spotřeby SoC, SPI flash paměť, možnost vyměnit tzv. *Daughterboard*[3] tlačítka, větší počet pinů, podpora pro desky typu mikroBUS, atd.

Zde je ale třeba zmínit, že piny na vývojové desce jsou multiplexovány, tedy na jedné „dráze“ je realizováno více služeb. Jednotlivé služby (SPI, mikroBUS, tlačítko) lze konfigurovat přemístěním vodičů. Multiplexing je zaveden z důvodu omezeného počtu volných pinů na samotné desce DA14531.

Měření spotřeby elektrické energie SoC DA14531 se tato práce věnuje v kapitole 7.

Vývoj na platformě Dialog Semiconductor

Nedílnou součástí všech mikrokontrolérů je tzv. *SDK – Software Development Kit*, neboli sada nástrojů a frameworků pro vývoj na daném zařízení. Dialog Semiconductor poskytuje SDK ke stažení na [4]. SDK navržené výrobcem zcela zásadně ovlivňuje následný vývoj softwaru, protože různé platformy i výrobci mají svá určitá specifika či omezení, se kterými je nutné při návrhu aplikace počítat. Níže si přiblížíme základní koncepty i komponenty využívané při vývoji aplikace na SoC DA14531.

Uživatel nemá přístup k žádným funkcím typu *main* apod. Modifikace SDK se silně nedoporučuje, naopak, upravovat soubory je doporučeno pouze u souborů začínajících na `user_` Předávání řízení programu mezi aplikací a SDK je realizováno pomocí tzv. *callback*. SDK definuje jaké *callbacky* jsou definovány a při jakých akcích dochází k jejich volání v souboru `user_periph_setup.h` Aplikace může přepsat již předdefinované funkce, nicméně tím také může způsobit nefunkčnost aplikace.

[1] System on Chip – Systém na čipu – integrovaný obvod obsahující většinu komponent počítače – tj. CPU, RAM, úložiště, hodiny reálného času...

[2] OTP – One Time Programable – zapsání dat do této paměti je nevratné, nelze ji přepisovat

[3] *Daugterboard* – SoC DA14531 instalovaný na samostatnou desku, připojuje se přes *PCI-I* k *Motherboard*, – ta obsahuje veškeré komponenty potřebné pro vývoj

Spánkové režimy

[3] U SoC a mikrokontroléru se často setkáváme se situací, kdy je zařízení mimo dosah elektrické sítě a musí být napájeno z baterie. Baterie mají omezenou kapacitu a proto je vhodné, a mnohdy až nutné odběr elektrické energie snížit. Jedním z nejúčinějších způsobů, jak toho dosáhnout je vypnout nebo alespoň omezit nevyužívané komponenty systému. Toto úsporné opatření se nazývá režim spánku. Obecně rozlišujeme více úrovní režimu spánku, a to podle množství vypnutých komponent na mikrokontroléru. V této sekci se zaměříme na režimy spánku dostupné na zařízení DA14531.

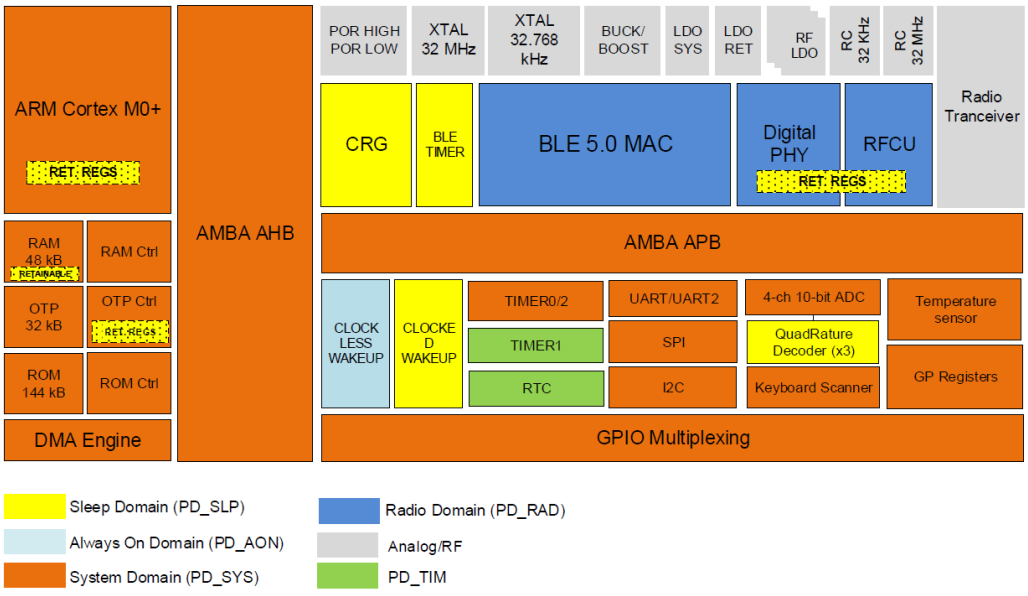
Systém při startu provádí inicializaci, ta se děje převážně bez vlivu uživatele. Uživatelova aplikace začíná svůj programu v soubor `user_periph_setup.c`. Zde se nastavují periferní zařízení, přiřazování funkcí k jednotlivým GPIO pinům (SPI, UART, I2C, ...).

DA14531 nabízí 3 režimy spánku:

- **Režim rozšířeného spánku**
  – V tomto režimu je vypnuta většina systémové domény (oranžové bloky) kromě SysRAM. Rádiová doména (BLE) je zastavena, stejně tak i doména GPIO portů (periferie) Domána AON je zapnutá a udržuje data v RAM a napájí bloky, které mohou probudit systém – tj. např. časovač vzbuzení a časovač BLE a také způsoby probuzení z hlubších spánků. – Aktivní BLE připojení jsou udržována (systém se periodicky probouzí, odešle zprávu a zase přejde do režimu spánku)

- **Režim hlubokého spánku**
  – V tomto režimu je navíc oproti předchozímu režimu vypnuta SysRAM. Zařízení probudí časovač buzení, GPIO, RTC a Timer1.

- **Režim hibernace**
  – V tomto režimu je vypnuta i zóna spánku včetně všech časovačů a hodin, jediný způsob jak zařízení probudit je přes GPIO.



Obrázek 5.2: **Blokové schéma spánkových zón na DA14531**

Při výběru vhodného spánkového režimu musíme počítat naopak s vyšší energetickou náročností na probuzení, čím „tvrději“ zařízení spí, tím také narůstající čas potřebný pro probuzení. Na krátké intervaly je tedy vhodný pouze první režim, režim rozšířeného spánku. Při hlubokém spánku a níže je vypnuta SysRAM, to při probuzení vede k nutnosti opětovně nahrát program do paměti SysRAM.

Nastavení spánkového režimu není obtížné, téměř o všechny věci se stará SDK, programátor musí pouze správně nastavit konstanty a v případě hlubokého spánku a hibernace nakonfigurovat bootovací zařízení.

Tabulka 5.1: Spotřeba DA14531 v jednotlivých spánkových zónách, zjednodušeno, Zdroj: [3]

Režim napájení	Popis	Podmínky	Typ	Jednot.
LOW 1.5V	napájení z bat.	rozšířený, 0 kB RAM, RCX	1.0	μA
LOW 1.5V	napájení z bat.	rozšířený, 48 kB RAM, RCX	1.6	μA
BUCK 3.0V	napájení z bat.	rozšířený, 96 kB RAM, RCX	4.6	μA
BOOST 3.3V	napájení z bat.	rozšířený, 96 kB RAM, RCX	4.0	μA

Z tabulky 5.1 můžeme pozorovat, že typická spotřeba DA14531 v rozšířeném režimu spánku a bez připojených periferií se pohybuje v rozmezí 1 μA – 5 μA. V naší konfiguraci se nejspíš bude systém blížit spíše k horní hranici.

RTC

Real-Time Clock (Hodiny reálného času) je komponenta která poskytuje zbytku systému přesný čas. RTC se nachází v téměř každém systému. Modul musí udržovat přesný čas bez ohledu na vnější vlivy. Z tohoto důvodu má většinou samostatné (někdy záložní, např. CR2032) napájení. Nevýhoda samostatného napájení je, že při výměně baterie dojde k vymazání všech hodnot. Přesnost je velice důležitá, protože RTC často řídí probouzení systému z režimu spánku v přesně stanoveném okamžiku. Samostatný modul nezávislý na CPU má více výhod. Tím, že je to úzce specializované zařízení, je méně energeticky náročné než CPU, které v důsledku tak může být v režimu spánku či počítat jiné věci. [16]

Většina RTC funguje na principu krystalového oscilátoru, nejčastěji o frekvenci 32.78 kHz, stejně jako v Quartz hodinkách. S touto frekvencí se snadno dělají binární operace, protože je to přesně 2^{15} cyklů za sekundu. DA14531 obsahuje dva RTC moduly. Interní RCX oscilátor a externí 32 kHZ krystal. Výběr aktivního modulu se provádí definováním makra `CFG_LP_CLK`.

Přerušení

Přerušení je další z nepostradatelných součástí mikrokontroléru. Je to signál zaslaný procesoru, který vyjadřuje, že proces nebo událost, která přerušení poslala, vyžaduje okamžitou pozornost. Protože procesor může v jednu chvíli provádět jen jednu instrukci, dokončí provádění aktuální instrukce, přeruší provádění programu a začne zpracovávat (obsluhovat) přerušení. Přerušení mohou být hardwarová i softwarová, ale v SDK6 nejsou softwarová přerušení programátorovi dostupná. Bez hardwarových přerušení by CPU musel v nekonečné smyčce kontrolovat, zda je připojené periferní zařízení připravené provést nějakou operaci. Hardwarová přerušení mohou být využita pro získání vstupních dat od uživatele,

např. stisk tlačítka s využitím GPIO, ale také pro indikaci CPU, že došlo ke nějaké události na SoC – např. přerušení pro probuzení systému ze spánku nebo změna stavu senzoru.

Spouštěcí sekvence

Při přivedení napájení na vývojový kit DK-PRO dochází ke spuštění a vykonání instrukcí na spouštěcí ROM. Zavaděč potupně zkouší v předem definované sekvenci zařízení a a z prvního aktivního začne nahrávat data do paměti. Sekvence je následující: `ext. SPI Master` –> `1 wire UART 1` –> `1 wire UART 2` –> `2 wire UART` –> `ext. SPI Slave` –> `I2C` –> `JTAG`. `JTAG` je nahrání dat z počítače, které se provede až jako poslední, přesně po 17.3ms.

Vývojové prostředí

Dialog vyvíjí vlastní integrované vývojové prostředí (IDE) SmartSnippets Studio, které slouží jako alternativa pro vývojové prostředí Keil uVision od společnosti arm. SmartSnippets je postavené na open-source základu Eclipse a překladači GCC. uVision je proprietární software, který je navíc pro profesionální použití a při překročení velikosti kódu přes 32 KB placený. Veškeré vzorové projekty od firmy Dialog jsou ale připravené na použití v uVision, takže kvůli snazšímu nastavování jsem nakonec pro vývoj používal uVision.

Kromě IDE vyvýjí Dialog také SmartSnippets Toolbox, který umí měřit aktuální spotřebu daugterboardu, nahrávat firmware do paměti, na flash, apod. Během vývoje mi sloužil jako skvělý nástroj pro sledování aktivity systému. Z grafu spotřeby lze přesně vyčíst, jestli systém zrovna spí, jak často odesílá data přes BLE, kolik času tráví v režimu spánku, v aktivním stavu, atd...

Kapitola 6

Implementace

Tato kapitola se zabývá procesem vývoje prototypu zařízení sloužícího k měření spotřeby elektrické energie. V kapitole 4 je nastíněn prvotní prototyp návrhu systému. Z tohoto návrhu jsem vycházel, ale průběžně docházelo ke změnám z důvodů nekompatibility či nevhodnosti součástek. V kapitole 5 jsou popsány využité technologie a knihovny.

Celý systém lze rozdělit na dvě hlavní části **Senzor**, jeho návrh a testování zapojení elektronického obvodu na snímání bliknutí LED, zpracovávání dat, a softwarová část senzorové části a **Server**, nastavení databáze a systému na vizualizaci dat, příjem dat, jejich validace a ukládání do databáze, tvorba webového serveru a zpracování požadavků.

6.1 Senzorová část

6.1.1 Obvod generující přerušení

Následující sekce je rozděleny do podsekcí podle způsobu měření bliknutí.

Ambient 2 Click

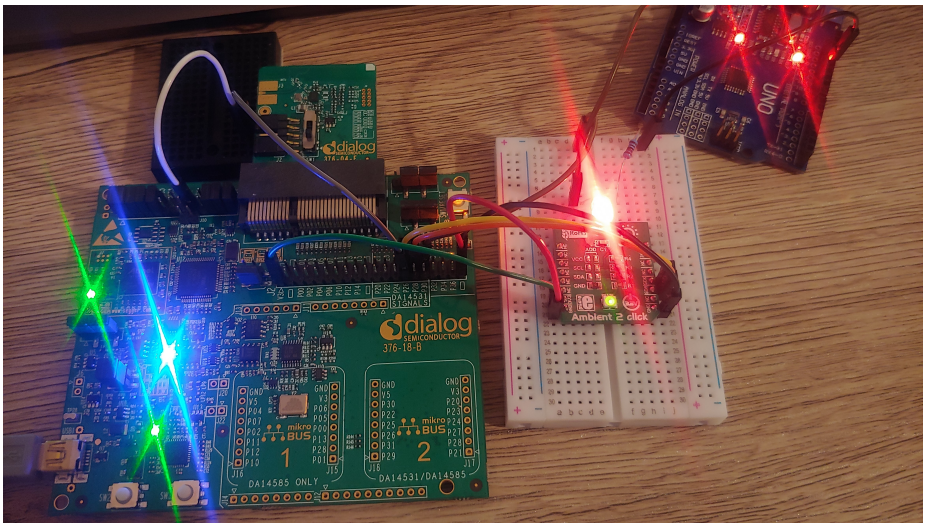
První verze návrhu počítala s využitím mikroBUS senzoru Ambient 2 Click se senzorem světla OTP3001. Tento senzor byl zvolen, protože vývojový kit DA14531 DK-PRO má vyvedení podporující desky typu mikroBUS. Tato destička je obyčejný senzor světla a pár rezistorů, ale její velkou výhodou je právě její tvar a piny.

Při pokusu o zprovoznění protokolu I2C a čtení hodnot nastal první problém. Přestože by měla destička teoreticky fungovat, ani po vyzkoušení všech možných variant, úspěšné čtení dat ze senzoru nebylo realizováno. Domnívám se, že problém je multiplexování více GPIO vývodů na jeden port, ale tuto teorii jsem neměl možnost ověřit. Po zapojení destičky do nepájivého pole a propojení s GPIO piny na DK-PRO označené jako J2 byla funkčnost senzoru i I2C komunikace ověřena. Na obrázku 6.1 lze toto nepraktické zapojení vidět. V pozadí se nachází Arduino UNO, které slouží jako náhrada za blikání elektroměru.

Senzor OTP3001 umožňuje nastavit si horní a dolní mez intenzity světla. Pokud je intenzita světla v nastaveném rozmezí, na výstupním pinu INT je logická 1. Tato změna lze využít k vyvolání přerušení procesoru. Procesor si informaci o bliknutí uloží a přerušení zamaskuje. Během testování frekvence blikání LED jsem zjistil, že tento senzor světla je pro účely měření počtu bliknutí za vteřinu nevhodný.

Elektroměr, který sloužil jako referenční udává poměr 10000 LED impulzů na 1 kWh spotřebované energie. Při orientačním měření dosahovala frekvence bliknutí až 16 Hz, tedy jedno bliknutí přibližně každých 60 ms.

Senzor OTP3001 uvádí v dokumentaci integrační čas[1] nejméně 100 ms. Z tohoto důvodu se jako nejlepší varianta jevilo vytvořit si vlastní elektrický obvod s přerušením.



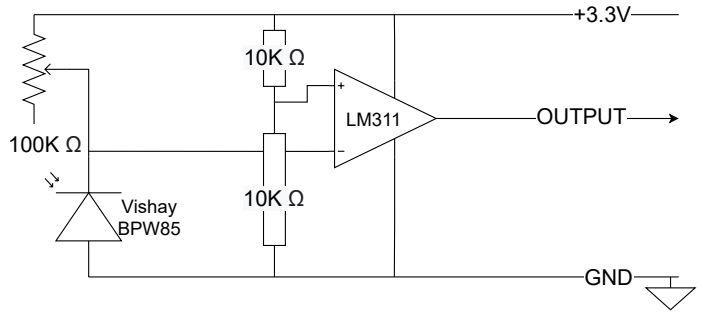
Obrázek 6.1: **Zapojení DA14531 a senzoru světla Ambient 2 Click**

Komparátory napětí – LM311 a MAX932

Měření bliknutí s využitím komparátoru a fototranzistoru funguje na principu napěťového děliče zapojeného mezi zdroj a zem. Světelné záření dopadající do kolektorového PN přechodu otevírá průchod proudu mezi bází a emitorem. Když záření na součástku nedopadá, chová se jako rezistor. Toto chování lze využít a měřit napětí zapojením fototranzistoru a potenciometru jako napěťový dělič. Komparátor porovnává napětí na přivedených vstupech a pokud je jedno větší než druhé, na výstupní pin pošle logickou jedničku. Pokud je poměr naopak, na výstupu je logická nula. Při posvícení je tedy napětí mezi zemí a místem spoje komponent rovno téměř nule. Na komparátoru se změní nerovnost a změní se tím pádem i hodnota výstupu jdoucí do mikrokontroléru. Na plusové části komparátoru je rovnoměrným napěťovým děličem dovedeno napětí přibližně 1.6 V. Obvod fungoval dobře, ale měřením spotřeby obvodu se ukázalo, že komparátor LM311 je nevhodný na použití. Jeho udávaný provozní napájecí proud je 7.5 mA! Při měření se spotřeba pohybovala okolo 3.5mA. Tyto údaje se samozřejmě v dokumentaci nachází, ale při výběru součástky jsem to i z důvodu nezkušenosti s výběrem vhodných součástek pro vytváření obvodů přehlédl.

Nově vybraný komparátor MAX932 má napájecí proud 4.5μA, tedy téměř tisíckrát méně. Komparátor MAX 932 je duální komparátor s vnitřní referencí s 1.2V na záporné straně. Z toho vyplývá, že při rozsvícené diodě napětí klesne pod 1.2V výstup komparátoru je logická nula, na rozdíl od předchozího obvodu. DA14531 lze nastavit vytvoření přerušení pouze na nástupné či sestupné hraně. Nastavením přerušení pouze na sestupnou hranu předcházíme opakovanému vyvolání přerušení v případě, že je dioda dlouho rozsvícená.

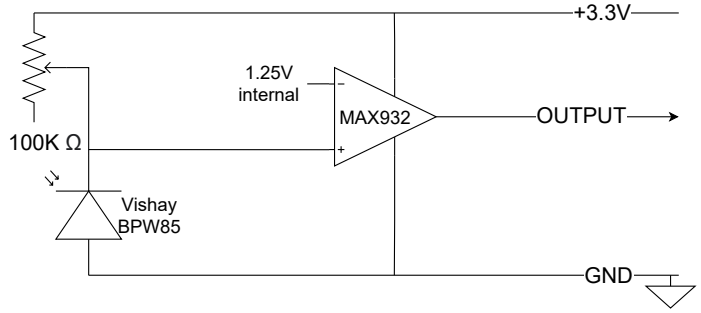
[1] Integrační čas – zjednodušeně po jakou dobu musí světlo dopadat na senzor, než bude dostupná hodnota intenzity dopadajícího světla



Obrázek 6.2: **Diagram zapojení LM311 a BPW85**

Potenciometr je nastavený přibližně 11 kΩ. Tím jsme docílili, že při absenci světla je napětí na komparátoru okolo 3 V, při svítící diodě ve vzdálenosti přibližně 1 cm od fototranzistoru je napětí okolo 0.3 V

Obvod na levé straně 6.4 obrázku je napojený na Arduino. Červené tlačítko vyvolává přerušení a zapíná/vypíná blikání LED.



Obrázek 6.3: **Diagram zapojení MAX932 a BPW85**

6.1.2 DA14531 – software

Jiné úložiště než RAM aplikace nevyužívá. Proto se, jak bylo popsáno v oddílu 5.1, při připojení přes USB kabel očekává nahrání dat přes JTAG z počítače.

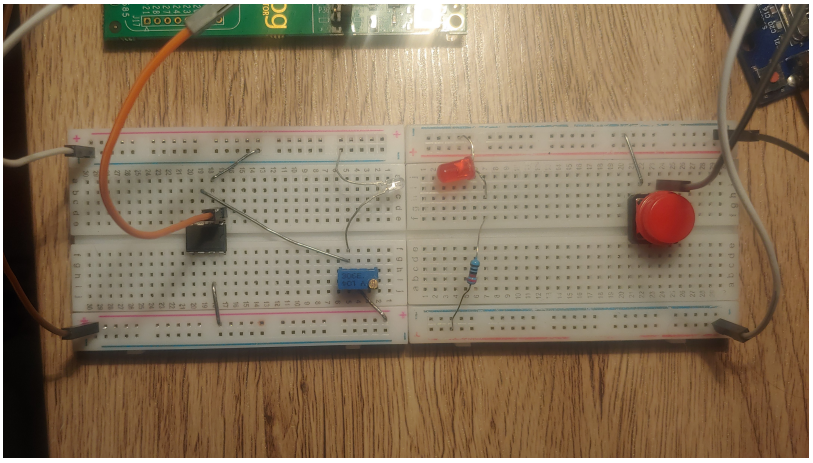
Implementace aplikace vznikla úpravou vzorového projektu ble_app_barebone, který je distribuován spolu s dalšími projekty v SDK6 od Dialog Semiconductor.

Aplikace začání v `user_periph_setup.h`, zde nastaví potřebné GPIO piny, tedy výstupní 3.3 V napájení a vstupní pin s pull-up rezistorem. Napájení senzoru[2] zůstává vypnuté.

soubor `energy_monitor.c`

Aplikace dále pokračuje do souboru `energy_monitor.c`. Vtomto souboru jsou především funkce zajišťující vysílání advertisement zpráv a správu spojení. Funkce `user_app_init()` je zavolána jako callback po spuštění systému. Zde si jde všimnout, že přestože je callback přepsán vlastní funkcí, na konci funkce je volána originální funkce. Dochází zde k nastavení

[2]Senzorem pro tuto chvíli myslíme elektronický obvod s komparátorem a fototranzistorem



Obrázek 6.4: **Finální podoba obvodu generujícího přerušení**

proměnných potřebných pro ukládání a odesílání měření. Dále dojde k přípravě dat pro vysílání advertising messages. A to v těchto funkcích:

- `mnf_data_init`
  Inicializace dat specifických pro výrobce – v tomto projektu není využíváno a proto jsou pro demonstrační účely ponechána výchozí data. Jedná se o data, která jsou odeslána s každým paketem odeslaným jako advertisement, a proto se musí vejít do 31 nebo 28 bytů, v závislosti na typu vysílaného advertisement paketu (zda se dá či nedá připojit k vysílajícímu zařízení, jedná se o 3B s flagy dat nutných k navázání spojení).

- `app_add_ad_struct`
  Obsahuje především datovou strukturu starající se o správné uložení vysílaných dat. V původním návrhu umí i odpovídat na scan reqeusts a citelně tím zvýšit propustnost dat, stále bez navázání spojení.

- `user_app_adv_start`
  Překopíruje data výrobce do advertisement struktury, především ale volá SDK a začíná vysílat.

Po úspěšném zahájení vysílání je aplikace ve stavu, kdy čeká na úspěšné navázání spojení mezi senzorem a centrálou. V této fázi je senzor stále vypnutý a přerušení budící z režimu spánku nejsou zapnuta.

Při připojování GATT klienta (centrály) aplikace kontroluje, jestli parametry spojení vyhovují oběma stranám. Dotaz klient odkládá pomocí časovače o 5 sekund, až když je jisté úspěšně navázané spojení. Tato aplikace bude s největší pravděpodobností vždy chtít po centrále, aby prodloužila interval mezi jednotlivými vysíláními. Konkrétní parametry jsou nastaveny v souboru `user_config.h`, kontkrétně v datové struktuře `connection_param_configuration`. Interval spojení je nastaven na 500 ms a latency [3] pouze 1. Doba, po které je spojení, kde druhá strana nekomunikuje, ukončeno je nastavena na 5 sekund. V tomto souboru se dále nastavují GATT role a další parametry spojení.

Při úspěšném spojení je zapnut GPIO pin napájející obvod se senzorem a povolen kontrolér řídící systémové probuzení a přerušení. Kontrolér zavolá callback funkci `app_wakeup_led_blink_cb`

[3] connection latency – kolikrát může protistrana „nedorazit“ na naplánovaná event

při každé detekované sestupné hraně na připojeném pinu. Dojde také k vypnutí advertisement zpráv. Funkce `user_app_disconnect` při ukončení spojení resetuje čítač bliknutí, vypne napájení senzoru, vypne časovač starající se o odesílání dat a obnoví vysílání advertisement zpráv.

Funkce `user_catch_rest_hndl` přijímá zprávy poslané přes GATT protokol. Přijaté zprávy třídí podle akce a typu a předává zodpovědným funkcím. Např. zpráva s id (zprávy mají přiřazené číselné konstanty) `USER_IDX_INTERVAL_VAL` je předána funkci `user_sensor_interval_mes` která se stará o její zpracování.

`app_wakeup_led_blink_cb` je callback funkce, která je zavolána při každém přerušení, které vznikne při bliknutí led. Funkce volá funkci `perpih_init`, protože nastavení GPIO po probuzení ze spánku nezůstává uloženo. Dále zvýší počítadlo bliknutí a znovu nastaví kontrolér vstávání.

GATT

V aplikaci je definována jedna služba a pod ní spadající dvě charakteristiky.

- Charakteristika sloužící k odesílání notifikací přihlášenému zařízen
  Klient přihlášený k této Charakteristice dostává periodicky 5 posledních nenulových měření.
- Charakteristika sloužící příjmu aktuálního měřícího intervalu Zapsáním do této charakteristiky dojde ke změně intervalu odesílaní dat, tedy i naměřené hodnoty budou vyšší.

V souborech `user_custs1_def.c` a `user_custs1_def.h` jsou definovány GATT služby a charakteristiky. Informace o službách a charakteristikách, jako jsou identtikátor, délka, popis a dále nastavení a práva přístupu k charakteristikám jsou uloženy v databázi v poli `custs1_att_db`. V hlavičkovém souboru jsou definovány UUID, délka zprávy a popis. Obsahuje také strukturu *enum* s ID zprávy a k nim přiřazenému rozpoznatelnému názvu.

Ukládání a odesílání naměřených hodnot

V souboru `user_custs1_impl.c` se nachází funkce zajišťující ukládání a odesílání naměřených dat. Systém pracuje na principu jednoduchého cyklického bufferu (o velikosti 15), do kterého jsou periodicky ukládány naměřené hodnoty a pseudo-unikátní číselný identifikátor. Při každém zavolání funkce `periodic_wakeup` je nahrazena nejstarší hodnota nejnovější. Každé měření dohromady zabere tři bajty – dva bajty počet bliknutí od posledního zavolání funkce a jeden bajt identifikátor. Po každém nenulovém zapsání naměřené hodnoty dojde k odeslání hodnot uložených v bufferu přes notifikaci klientovi – příjemci notifikací. Před odesláním jsou hodnoty zkopírovány do dočasného pole a seřazeny od nejnovější hodnoty po nejstarší hodnotu. Z indexu posledního zapsaného měření jsem schopen dojít postupně až k nejstarší hodnotě. Příjemce nemusí data složitě zpracovávat, protože první tři bajty jsou vždy nejnovější měření a poslední 3 bajty vždy nejstarší měření. Při přenosu dat by tedy nemělo dojít ke ztrátě měření i při ztrátě čtyř po sobě jdoucích zpráv.

Funkce `user_sensor_interval_message_handler` přijímá zprávy od klienta. Obsahem zprávy je jeden bajt, vyjadřující periodu odesílaní dat.

6.2 Serverová část

Jako zařízení pro uchovávání naměřených dat slouží počítač Raspberry Pi 4. Připojení ke klientovi, příjem dat a uložení naměřených informací do databáze obstarává skript bp.py psaný v programovacím jazyce Python 3.7.3. Po spuštění skriptu dojde načtení základních údajů z json souboru. Soubor se musí nacházet ve stejné složce jako skript. V případě, že soubor neexistuje, je vytvořen nový soubor s výchozími hodnotami: 1kWh je rovna 10000 bliknutí na impulsním výstupu a interval měření je 1 sekunda. Poté jsou vytvořena za použití knihovny Thread vytvořena a spuštěna dvě nová vlákna. První vlákno slouží obsahuje python klienta pro práci s Bluetooth Low Energy *Bleak*[4]. Ve druhém vlákně běží webový server Flask.

6.2.1 Bluetooh Client

Bleak je zkratka pro „Bluetooth Low Energy platform Agnostic Klient“. Pro naše účely slouží jako GATT klient, který vyhledá dostupné BLE zařízení podle zadaných filtrů. Při hledání se můžeme zaměřit pouze na zařízení, které nabízejí GATT službu s nám známým UUID. Takto se tedy ke všem nalezeným klientům můžeme s jistou že to jsou senzory spotřeby připojit. Při vývoji jsem původně plánoval počítal s možností připojení více senzorů, ale protože jsem nemohl vyzkoušet připojit se k více senzorům najednou, je tato možnost spíše jen teoretická.

Z nalezených zařízení s využitím knihovny Bleak vytvoříme objekty `BleakClient` s kterými jdou GATT operace provádět velmi jednoduše. Napřed je potřeba na senzoru nastavit frekvenci odesílání dat na takovou, jaká je nastavena v konfiguračním souboru. Poté se už můžeme přihlásit k přijímání notifikací od senzoru. Při přihlášení je nutné zadat funkci, která bude dané zprávy zpracovávat. Funkce `notification_handler` přijaté data zpracuje a vloží do databáze, pokud ještě vloženy nebyly. Kontrola probíhá porovnáním ID posledních 10 přijatých hodnot. Objekt `deque` při zadání maximální velikosti automaticky udržuje posledních záznamy v paměti a staré odstraňuje.

Vkládání dat do databáze InfluxDB provádí funkce `push_to_db` s využitím knihovny *influxdb*.

6.2.2 Flask

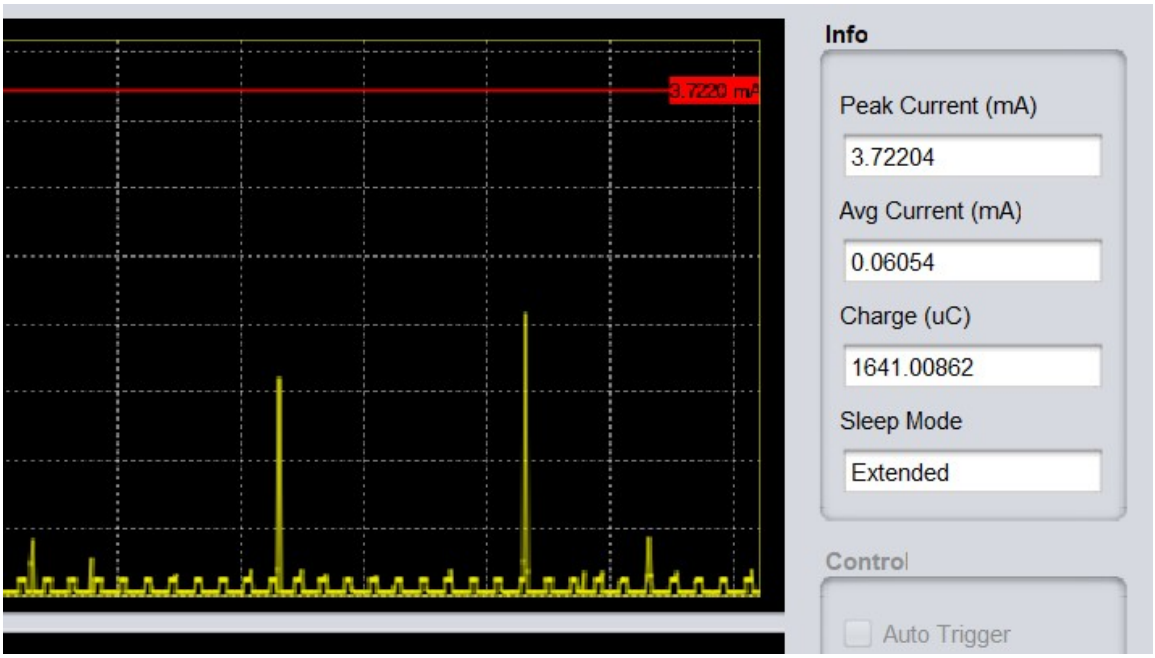
Flask je knihovna pro vytváření jednoduchých webových serverů s podporou html šablon. Webová stránka je realizována jako iframe Grafany téměř přes celou stránku s výjimkou úzkého pruhu v záhlaví vyhrazeného pro ovládací prvky senzoru.

[4] https://bleak.readthedocs.io/en/latest/

Obrázek 6.5: **Webové rozhraní Flask**

Kapitola 7

Testování



Obrázek 7.1: **SmartSnippets ToolBox**

Kapitola 8

Závěr

Cílem této práce bylo vytvořit systém na měření a vizualizaci spotřeby elektrické energie s důrazem na maximalizaci životnosti baterie. Pro přenos dat byla použita technologie Bluetooth Low Energy.

Prvním krokem bylo seznámení se s technologií Bluetooth Low Energy a protokolem GATT.

Dalé bylo nutné vybrat si vhodnou platformu, která odpovídala požadavkům zadání. Výběr Dialog Semiconductor DA14531 se ukázal jako skvělá volba z především kvůli velmi propracovanému SDK6.

Prvotní návrh systému s modulem Ambient 2 Click se ukázal jako nedostatečný z důvodu rychlosti měření. Vytvořením vlastního obvodu s komparátorem a fototranzistorem byly problémy vyřešeny.

Prototyp zařízení je funkční.

Literatura

[1] Bluetooth SIC. *GATT Specifications* [online]. [cit. 2021-01-25]. Dostupné z: https://www.bluetooth.com/specifications/gatt/.

[2] Bluetooth SIG. *Bluetooth Core Specification* [online]. [cit. 2021-01-25]. Dostupné z: https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=457080.

[3] Dialog semiconductor . *DA14531/DA14585-586 Sleep mode tutorial* [online]. [cit. 2021-05-8]. Dostupné z: http://lpccs-docs.dialog-semiconductor.com/DA14531_Sleep_Mode/introduction.html.

[4] Dialog semiconductor . *SDK6* [online]. [cit. 2021-05-8]. Dostupné z: https://www.dialog-semiconductor.com/system/files/2021-04/SDK_6.0.14.1114.zip.

[5] Dialog Semiconductor. *DA14531MOD SmartBond TINYTM Module* [online]. [cit. 2021-01-25]. Dostupné z: https://www.mouser.com/datasheet/2/713/da14531_module_datasheet_1v0-1843458.pdf.

[6] Dialog Semiconductor. *Figure 1 Bluetooth Low Energy Protocol Stack* [online]. [cit. 2021-01-25]. Dostupné z: http://lpccs-docs.dialog-semiconductor.com/DA145xx_Advertising_Tutorial/_images/Figure1.svg.

[7] Dialog Semiconductor. *Figure 3 BLE Link-Layer States* [online]. [cit. 2021-01-25]. Dostupné z: http://lpccs-docs.dialog-semiconductor.com/DA145xx_Advertising_Tutorial/_images/Figure3.svg.

[8] Dialog Semiconductor. *Figure 4 Profiles, services, and characteristics* [online]. [cit. 2021-01-25]. Dostupné z: http://lpccs-docs.dialog-semiconductor.com/tutorial-custom-profile-DA145xx/gatt.html.

[9] Embeded Centric. *Introduction to Bluetooth Low Energy / Bluetooth 5* [online]. [cit. 2021-01-25]. Dostupné z: https://embeddedcentric.com/introduction-to-bluetooth-low-energy-bluetooth-5/.

[10] Embeded Centric. *Lesson 2 – BLE profiles, services, characteristics, device roles and network topology* [online]. [cit. 2021-01-25]. Dostupné z: https://embeddedcentric.com/lesson-2-ble-profiles-services-characteristics-device-roles-and-network-topology/.

[11] Last Minute Engineers. *DA14530/1 SmartBond TINY™ Development Kit Pro* [online]. [cit. 2021-01-25]. Dostupné z: https://www.dialog-semiconductor.com/sites/default/files/styles/width_700/public/da14531-dev-kit_pro.jpg?itok=fi82wIY1.

[12] Last Minute Engineers. *Insight Into ESP32 Sleep Modes & Their Power Consumption* [online]. [cit. 2021-01-25]. Dostupné z: https://lastminuteengineers.com/esp32-sleep-modes-power-consumption/.

[13] Mikroe. *Ambient 2 click* [online]. [cit. 2021-01-25]. Dostupné z: https://www.mikroe.com/ambient-2-click.

[14] OpenEnergyMonitor. *Monitoring energy via utility meter pulse output* [online]. [cit. 2021-01-25]. Dostupné z: https://learn.openenergymonitor.org/electricity-monitoring/pulse-counting/introduction-to-pulse-counting#monitoring-energy-via-utility-meter-pulse-output.

[15] Silicon Labs. *UG103.14: Bluetooth® LE Fundamentals* [online]. [cit. 2021-01-25]. Dostupné z: https://www.silabs.com/documents/public/user-guides/ug103-14-fundamentals-ble.pdf.

[16] Techopedia. *Real-Time Clock (RTC)* [online]. [cit. 2021-05-8]. Dostupné z: https://www.techopedia.com/definition/2273/real-time-clock-rtc.

[17] Øvrebekk, T. *Bluetooth 5 Advertising Extensions* [online]. [cit. 2021-01-25]. Dostupné z: https://blog.nordicsemi.com/getconnected/bluetooth-5-advertising-extensions.