Fakulta elektrotechnická

Katedra aplikované elektroniky a telekomunikací

Diplomová práce

Aplikační SW pro zobrazovací jednotku elektrovozidla

Autor práce: Bc. Martin Beran

Vedoucí práce: Ing. Petr Weissar, Ph.D.

Plzeň 2019

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2018/2019**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Martin BERAN**
Osobní číslo:	**E16N0037P**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Elektronika a aplikovaná informatika**
Název tématu:	**Aplikační SW pro zobrazovací jednotku elektrovozidla**
Zadávající katedra:	**Katedra aplikované elektroniky a telekomunikací**

Zásady pro vypracování:

Vytvořte vhodné SW vybavení pro stávající zobrazovací jednotku elektrovozidla. Uvažujte primární nasazení v elektromotokáře FEL.

1. Navrhněte a realizujte zobrazení hodnot a stavů ve formě přehledných obrazovek.
2. Umožněte vizuální nastavování parametrů vozidla.
3. Uvažujte komunikaci pomocí CAN (vozidlo), XBee telemtrie (připojené PC) a USB pro nastavení a vyčtení historie provozu.
4. Uvažujte senzory GPS a akcelerometry, dále stav řízení (pedály apod.)

Rozsah grafických prací:	**podle doporučení vedoucího**
Rozsah kvalifikační práce:	**40 - 60 stran**
Forma zpracování diplomové práce:	**tištěná/elektronická**

Seznam odborné literatury:

Diplomová práce "Jednotka displeje pro elektromotokáru", FEL 2016, autor: Sova Martin

Vedoucí diplomové práce:	**Ing. Petr Weissar, Ph.D.** Katedra aplikované elektroniky a telekomunikací
Datum zadání diplomové práce:	**5. října 2018**
Termín odevzdání diplomové práce:	**30. května 2019**

Prof. Ing. Zdeněk Peroutka, Ph.D.
děkan

L.S.

Doc. Dr. Ing. Vjačeslav Georgiev
vedoucí katedry

V Plzni dne 5. října 2018

Abstrakt

Tato diplomová práce se zabývá návrhem softwaru pro jednotku displeje pro elektromotokáru. Jednotka obsahuje displej pro zobrazování provozních dat. Data jsou získávány ze sběrnice CAN a ze senzorů umístěných přímo na DPS jednotky. Pro nastavení zobrazení dat na displeji slouží konfigurační PC aplikace.

Klíčová slova

elektromotokára, displej, SW, CAN, STM32

Abstract

This master thesis deals with software design for display unit for electric kart. The unit includes a display for displaying operational data. Data is obtained from the CAN bus and from sensors located directly on the PCB. The configuration PC application is used to configure the displayed data.

Keywords

electric kart, display, SW, CAN, STM32

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

Dále prohlašuji, že veškerý software, použitý při řešení této diplomové práce, je legální.

V Plzni dne 29. května 2019

Bc. Martin Beran

. . . . . .

Podpis

Obsah

Seznam tabulek **viii**

Seznam symbolů a zkratek **ix**

1 Úvod **1**

2 Hardware **2**

2.1 Mikrokontrolér 3

2.2 Externí SDRAM paměť 3

2.3 Displej 3

2.4 XBee modul 4

2.5 Externí Flash paměť 4

2.6 Akcelerometr 5

2.7 Teplotní senzor 5

2.8 GPS přijímač 6

2.9 Analogové vstupy 6

2.10 Digitální vstupy a výstupy 6

3 Aplikační SW pro jednotku displeje **8**

3.1 Konfigurace mikrokontroléru 8

3.1.1 Nastavení rozvodu hodinového signálu 8
3.1.2 Nastavení LTDC periferie 9
3.1.3 Nastavení FMC periferie 11
3.1.4 Nastavení QUADSPI periferie 12
3.1.5 Nastavení UART periferií 13
3.1.5.1 UART4 periferie 13
3.1.5.2 UART5 periferie 13
3.1.6 Nastavení I^2C periferie 14
3.1.7 Nastavení CAN periferie 14
3.1.8 Nastavení USB periferie 15
3.1.9 Nastavení ADC periferie 17
3.1.10 Nastavení RTC periferie 18
3.1.11 Nastavení IWDG periferie 18
3.1.12 Nastavení TIMx periferií 19
3.1.13 Nastavení GPIO pinů 19
3.2 Knihovna pro ovládání displeje 21
3.2.1 Funkce LCD_Draw_HorizontalLine 22
3.2.2 Funkce LCD_Draw_VerticalLine 23
3.2.3 Funkce LCD_Draw_Rectangle 23
3.2.4 Funkce LCD_Draw_FilledRectangle 24
3.2.5 Funkce LCD_Draw_VerticalBar 24
3.2.6 Funkce LCD_Draw_VerticalBarZero 25
3.2.7 Funkce LCD_Draw_Char 26
3.2.8 Funkce LCD_Display_Char 26

3.2.9 Funkce LCD_Display_String 27
3.2.10 Funkce LCD_GetDigits 27
3.2.11 Funkce LCD_Display_Number 27
3.2.12 Funkce LCD_Display_Page 28
3.3 Knihovna pro ovládání akcelerometru 28
3.4 Knihovna pro ovládání teploměru 29
3.5 Knihovna pro ovládání ADC 29
3.6 Postup inicializace komponent při startu jednotky displeje 30

4 Konfigurační aplikace pro PC **32**

4.1 Komunikační protokol 32
4.1.1 Transportní protokol 32
4.1.2 Aplikační příkazy 32
4.1.2.1 Příkaz pro čtení externí Flash paměti 33
4.1.2.2 Příkaz pro čtení dat kanálů 33
4.1.2.3 Příkaz pro mazání flash paměti 34
4.1.2.4 Příkaz pro nastavení adresy a délky dat pro zápis dat do flash paměti 34
4.1.2.5 Příkaz pro přenos dat do flash paměti 35
4.1.2.6 Negativní odpovědi na příkazy 35
4.2 Nastavení komunikačních a zobrazovacích kanálů 36

5 Závěr **39**

Reference, použitá literatura **41**

Přílohy **43**

A Přílohy **44**

A.1 Jednotka displeje 44

Seznam tabulek

2.1 Tabulka časování synchronizačních signálů displeje MCT070M6W800480LML.[5] 4

4.1 Transportní protokol mezi aplikací a jednotkou displeje 32

4.2 Tabulka příkazu ReadFlashData 33

4.3 Tabulka příkazu ReadChannelsData 33

4.4 Tabulka příkazu DisplayControl–EraseChip 34

4.5 Tabulka příkazu DisplayControl–ErasSubSector 34

4.6 Tabulka příkazu RequestDownload 35

4.7 Tabulka příkazu TransferData 35

4.8 Tabulka struktury negativní odpovědi 36

4.9 Tabulka negativní kódů 36

Seznam symbolů a zkratek

DPS	Deska plošných spojů.
LTDC	LCD–TFT display controller.
FMC	Flexible memory controller.
QSPI	Quad–SPI interface.
NMEA	National Marine Electronics Association.
RTC	Real –time clock.
HAL	Hardware abstraction layer.
AHB	Advanced High–performance Bus
APB	Advanced Peripheral Bus
DMA	Direct memory access

1 Úvod

Předkládaná práce je zaměřena na realizaci aplikačního softwaru pro jednotku displeje pro elektromotokáru. Hardware této jednotky byl vytvořen Bc. Martinem Sovou v jeho diplomové práci. Jednotka displeje je určena pro zabudování do elektrické motokáry vyvíjené na Fakultě elektrotechnické Západočeské univerzity v Plzni.

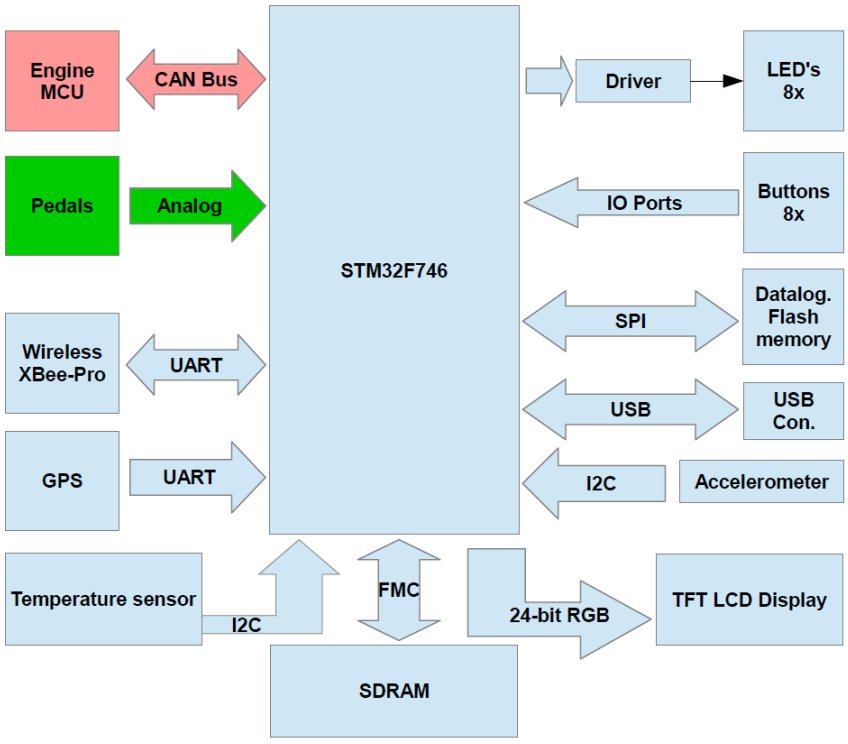
Úkolem jednotky je zobrazovat provozní veličiny na grafickém displeji a ukládat provozní data. Jednotka také umožňuje snímání polohy plynového a brzdového pedálu. Dále snímá polohu pomocí GPS přijímače, zrychlení ve třech osách pomocí akcelerometru a okolní teplotu. Jednotka displeje také umožňuje připojit XBee–Pro modul, pro bezdrátovou komunikaci a odesílání telemetrických dat při provozu elektromotokáry.

Požadavky na software byly takové, aby bylo možné nastavení obrazovek bez zásahu do kódu mikrokontroléru a aby změnu nastavení zvládl běžný uživatel. S ohledem na tento požadavek byl vybrán způsob nastavení displeje pomocí kalibračních hodnot umístěných v externí flash paměti jednotky displeje. Kalibrační hodnoty jsou vytvořeny v PC aplikaci a poté nahrány do paměti jednotky. Další alternativou by bylo zavedení bootloaderu do mikrokontroléru a vytvoření PC aplikace, která by na základě nastavení provedených uživatelem, vkládala předpřipravené části kódu do základního kódu jednotky a po kompilaci by se tento upravený kód nahrál do jednotky. Tento způsob se však jevil jako velmi komplikovaný a časově náročnější.

V První kapitole je popsán HW jednotky displeje pro potřeby inicializace jednotlivých komponent mikrokontroléru. Ve druhé kapitole je popsána inicializace komponent a funkce, které slouží pro obsluhu displeje a ostatních periferií. Ve třetí kapitole je popsána konfigurační PC aplikace, transportní protokol a příkazy pro ovládání a nastavení jednotky.

2 Hardware

Aplikační software navržený v této diplomové práci je určen pro platformu jednotky dipleje, kterou vytvořil v rámci své diplomové práce Martin Sova.[1] Tato kapitola se bude zabývat stručným popisem použitého hardwaru, s ohledem pro potřeby návrhu softwarového vybavení. Na obrázku 2.1 je vyobrazeno v blokovém schématu propojení mikrokontroléru se všemi zařízeními a sběrnicemi jednotky displeje.



Obrázek 2.1: Blokové schéma periferií připojench k mikrokontroléru.[1]

2.1 Mikrokontrolér

Základem jednotky displeje je mikrokontrolér STM32F746IGT od firmy STMicroelectronics. Jedná se o $ARM®Cortex® - M7$ 32 bitový mikrokontrolér s maximální frekvencí jádra 216 MHz. Vnitřní flash paměť mikrokontroléru je 1 MB a RAM paměť o velikosti 320 kB. K mikrokontorléru jsou připojeny dva krystalové oscilátory. Jeden s rezonančním kmitočtem 8 MHz pro jádro a periferie mikrokontroléru. Druhý oscilátor s frekvencí 32,768 kHz slouží jako zdroj hodinového signálu pro RTC periferii. Programování mikrokontroléru je možné pomocí rozhraní JTAG nebo SW [4]

2.2 Externí SDRAM paměť

Mikrokontrolér neobsahuje dostatečně velkou RAM paměť pro uchování stránky displeje, proto je k mikrokontroléru připojena externí 64 Mb SDRAM paměť IS42S16400J. Pro vykreslení displeje s rozlišením 800 x 480 px a barevné hloubce 16 bitů je potřeba 768 kB RAM paměti.[1] IS42S16400J je připoje na k FMC periferii mikrokontroléru, díky tomu je zajištěna maximální přístupová rychlost k paměti.

2.3 Displej

Pro zobrazování provozních hodnot slouží sedmipalcový displej MIDAS MCT070M6W800480LML s rozlišením 800 x 480 px. Displej je připojen na periferii LTDC mikrokontroléru, která je určena pro přímé řízení LCD panelů. V tabulce 2.1 je uvedeno časování datových signálů displeje. [5]

Item	Symbol	Min.	Typ.	Max.	Unit	Remark
Horizontal Display Area	thd		800		DCLK	
Dclk frequency	fclk	–	30	50	MHz	
One horizontal line	th	889	928	1143	DCLK	
HS pulse width	thpw	1	48	255	DCLK	
HS blanking	thb		88		DCLK	
HS front porch	thfp	1	40	255	DCLK	
Vertical Display Area	tvd		480		TH	
VS period time	tv	513	525	767	TH	
VS pulse width	tvpw	3	3	255	TH	
VS blanking	tvb		32		TH	
VS front porch	tvfp	1	13	255	TH	

Tabulka 2.1: Tabulka časování synchronizačních signálů displeje MCT070M6W800480LML.[5]

2.4 XBee modul

Jednotka je osazena XBee®/XBee–PRO® RF modulem pro bezdrátovou komunikaciv pásmu 2 4 GHz. Maximální dosah modulu ve volném prostředí je 750 m a maximální přenosová rychlost je 250 kBaud. Komunikce XBee s mikrokontrolérem je po sběrnici UART. Připojení XBee k mikrokontorléru neumožňuje harwarové řízení toku dat, ani změnu vysílacích parametrů (např. přenosové rychlosti). [9]

2.5 Externí Flash paměť

Mikrokontrolér neobsahuje EEPROM paměť,proto je pro uchování dat po vypnutí napájení připojena externí 128Mb flash paměť Micron N25Q128A typu serial NOR, která je připojena k mikrokontroléru pomocí QSPI sběrnice. Paměť lze mazat po sektorech o velikosti 64 kB, subsektorech o velikosti 4 kB, nebo celou paměť najednou.Na

obrázku 2.2 je tabulka s mapou paměti.[2]

Table 2: Sectors[255:0]

Sector	Subsector	Address Range	
		Start	End
255	4095	00FF F000h	00FF FFFFh
	⋮	⋮	⋮
	4080	00FF 0000h	00FF 0FFFh
⋮	⋮	⋮	⋮
127	2047	007F F000h	007F FFFFh
	⋮	⋮	⋮
	2032	007F 0000h	007F 0FFFh
⋮	⋮	⋮	⋮
63	1023	003F F000h	003F FFFFh
	⋮	⋮	⋮
	1008	003F 0000h	003F 0FFFh
⋮	⋮	⋮	⋮
0	15	0000 F000h	0000 FFFFh
	⋮	⋮	⋮
	0	0000 0000h	0000 0FFFh

Obrázek 2.2: Tabulka mapy externí flash paměti.[2]

2.6 Akcelerometr

Jednotka displeje je také osazena 3–osým MEMS akcelerometrem LIS331HH. Akcelerometr je připojen k mikrokontroléru sběrnicí I^2C s maximální komunikační rychlostí 400 kHz. Adresa akcelerometru na sběrnici je 0011000. Maximální rozsah měření akcelerometru je možné nastavit na $\pm$ 6 g/$\pm$12 g/$\pm$24g. Maximální rychlost měření akcelerometru je 1 kHz. [6]

2.7 Teplotní senzor

Jednotka obsahuje digitální teplotní senzor LM75BIM. Senzor komunikuje s mikrokontrolérem po sběrnici I^2C, stejně jako akcelerometr je schopen komunikovat rychlostí 400 kHz. Adresa teploměru na sběrnici je 1001101. Teplotní senzor je schopen měřit teploty od -55 °C do 100 °C s přesností ± 2 °C.[7]

2.8 GPS přijímač

Pro sledování pozice vozidla je v jednotce nainstalován GPS přijímač Maestro A2235–H. GPS přijímač automaticky po zapnutí vysílá data o poloze no sběrnici UART s přenosovou rychlostí 4800 Baud a GPS protokolem NMEA. GPS zprávy jsou odesílány s frekvencí 5 Hz. GPS přijímač nelze pomocí UART komunikace nastavovat. Na obrázku 2.3 je příklad dekódování GPGGA věty NMEA protokolu.[8]

```
$GPGGA,170139.615,4912.2526,N,01635.0378,E,1,07,1.0,357.5,M,43.5,M,0.0,0000*7D
```

#	formát	příklad	komentář
1	hhmmss.sss	170139.615	Čas (UTC), pro který platí údaje o vypočtené pozici
2	ddmm.mmmm	4912.2526	Zeměpisná šířka
3	c	N	Indikátor severní/jižní šířka (N=sever, S=jih)
4	dddmm.mmmm	01635.0378	Zeměpisná délka
5	c	E	Indikátor východní/západní délky (E=východ, W=západ)
6	d	1	Indikátor kvality: 0 — nebylo možno určit pozici 1 — pozice úspěšně určena 2 — pozice úspěšně určena (diferenční GPS)
7	dd	07	Počet viditelných satelitů 00 — 12
8	d.d	1.0	Vliv rozestavění družic na určení polohy HDOP (*Horizontal Dilution of precision*)
9	d.d	357.5	Výška antény nad geoidem
10	c	M	Jednotka pro předchozí údaj (č.9) (M=metr)
11	d.d	43.5	Geoidal separation, rozdíl mezi WGS-84 zemským elipsoidem a střední úrovní moře (geoid). Znaménko mínus znamená, že střední úroveň země je pod elipsoidem.
12	c	M	Jednotka vzdálenosti pro předchozí položku (č.11) (M=metr)
13	d.d	0.0	Stáří poslední aktualizace DGPS. Údaj je uváděn v sekundách. Jestliže údaj chybí, nepoužívá se DGPS.
14	dddd	0000	Identifikační číslo referenční stanice pro DGPS (0000 — 1023)
15	*xx	7D	Kontrolní součet

Obrázek 2.3: Příklad NMEA GPGGA věty. |Převzato z [10]|

2.9 Analogové vstupy

Na analogové vstupy jsou přivedeny signály ze senzorů polohy plynového a brzdového pedálu. Protože tyto signály jsou proudové je jednotka displeje vybavena převodníky proud–napětí AD8211. Výstupy převodníku proud–napětí jsou přivedeny na vstupy AD převodníku mikrokontroléru.

2.10 Digitální vstupy a výstupy

Jednotka displeje má 8 digitálních vstupů, které jsou přivedeny přes Schmittův klopný obvod na piny mikrokontroléru. Na 5 z 8 vstupů jsou připojena tlačítka umístěná

přímo na desce plošného spoje jednotky displeje. Zbývající tři vstupy jsou vyvedeny na konektor. Jednotka displeje je vybavena 8 digitálními výstupy, které jsou určené pro ovládání LED diod pomocí tranzistorového pole ULN2803A.

3 Aplikační SW pro jednotku displeje

Software byl vytvořen programovacím prostředí Attolic TrueSTUDIO for STM32.

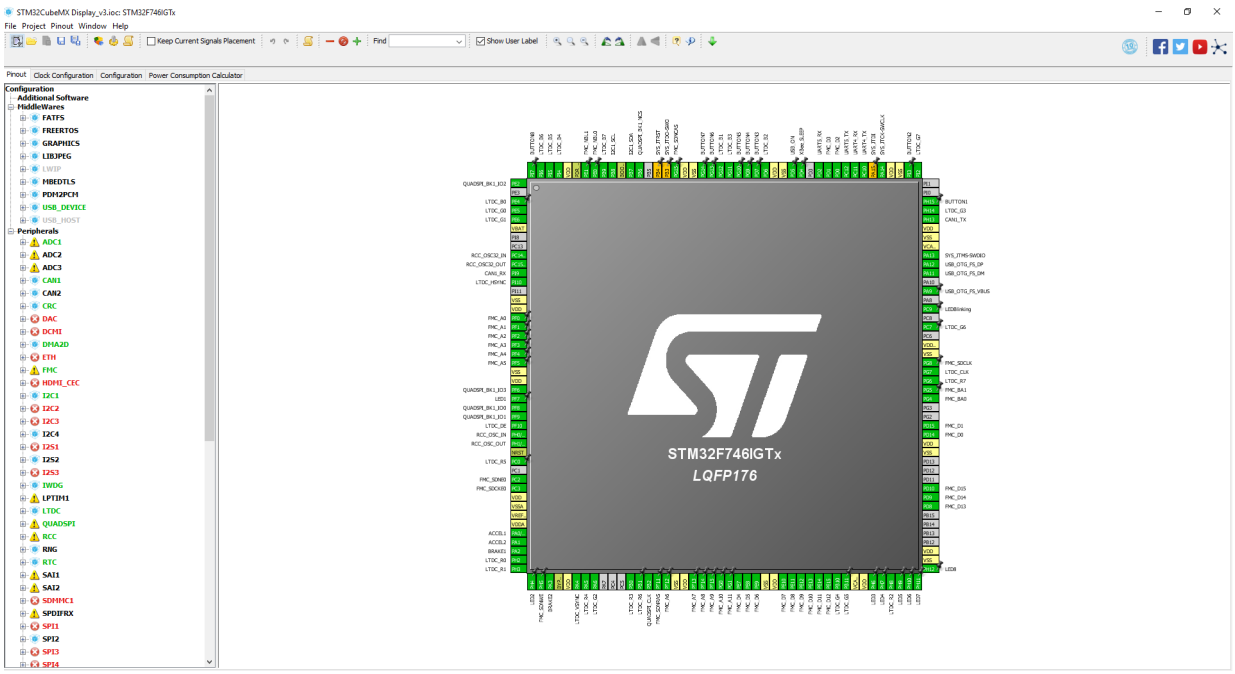
3.1 Konfigurace mikrokontroléru

Základní konfigurace mikrokontroléru byla provedena pomocí programu STM32CubeMX. STM32CubeMX slouží pro grafické nastavení mikrokontroléru a jeho periferií s využitím HAL vrstvy, která umožňuje maximální přenositelnost programů přes portfolio mikrokontrolérů STM32.[11]

Nastavení vstupně výstupních portů bylo provedeno na základě tabulky v příloze 2 v diplomové práci Martina Sovy.[1]

3.1.1 Nastavení rozvodu hodinového signálu

Na obrázku 3.2 je vidět nastavení rozvodu hodinového signálu mikrokontroléru. Zdrojem hodinového signálu je krystalový rezonátor s frekvencí 8 MHz. Tento signál je přiveden přes předděličku do fázových závěsů, kde je frekvence vynásobena na hodnotu 216 MHz. Tento kmitočet je zaveden do jádra mikrokontroléru a slouží jako hodinový signál pro AHB sběrnici. Také jsou z něj odvozeny frekvence pro sběrnice APB1 a APB2. Kdy sběrnice APB1 je taktována frekvencí 54 MHz a sběrnice APB2 běží na

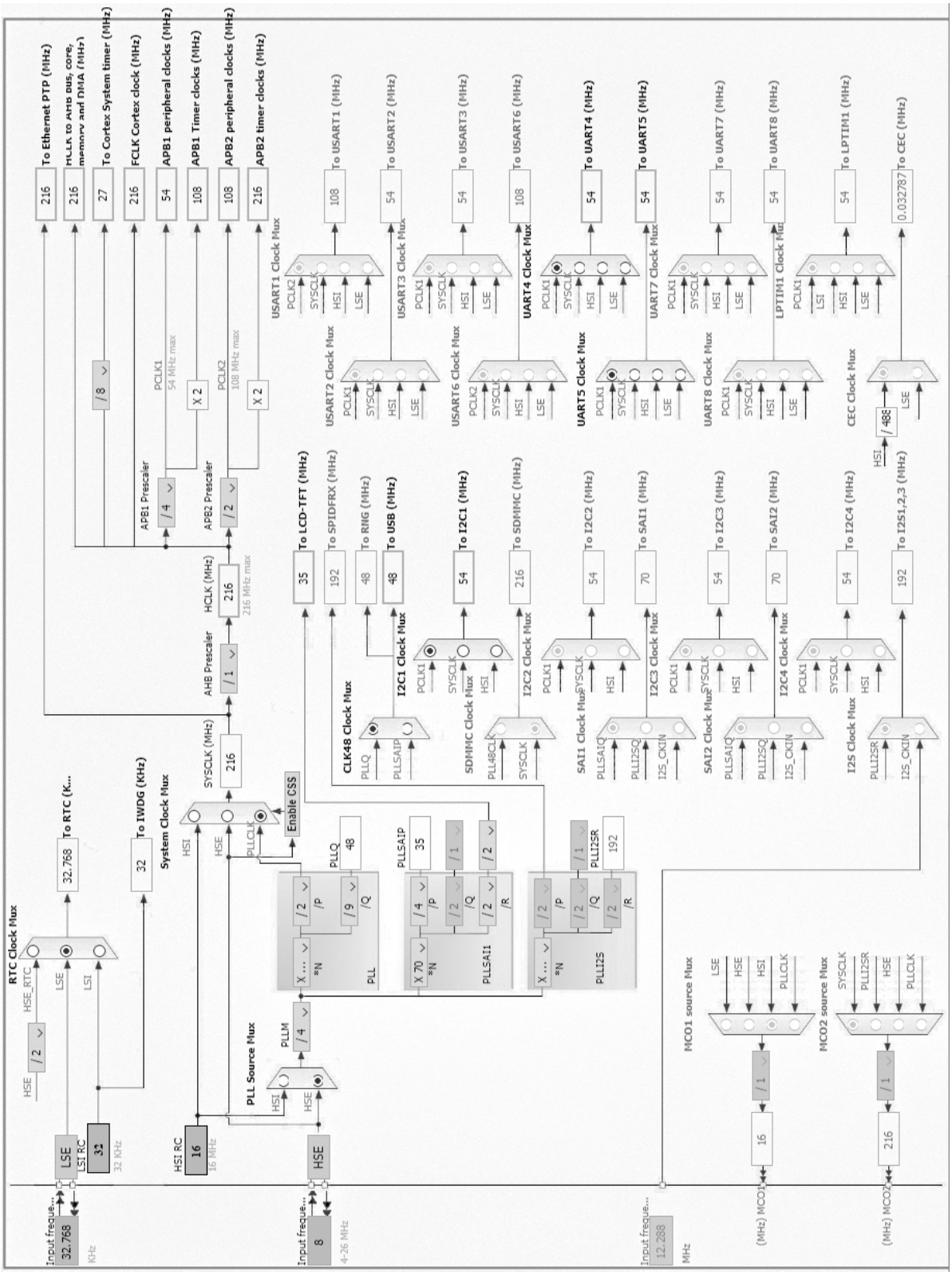


Obrázek 3.1: Základní nastavení mikrokontroléru v programu STM32CubeMX

frekvenci 108 MHz. Fázový závěs PLLSAI1 je nastaven tak, aby generoval hodinový signál s frekvencí 35 MHz pro periferii LTDC. Pro periferie UART4, UART5 a I2C1 je zdrojem hodinováho signálu zvolen hodinový signál sběrnice APB1 s hodnotou 54 MHz. Hodinový signál pro USB periferii je získán podělením kmitočtu jádra na hodnotu 48 MHz. Zdrojem hodinového signálu pro periferii RTC je krystalový rezonátor s frekvencí 32.768 kHz.

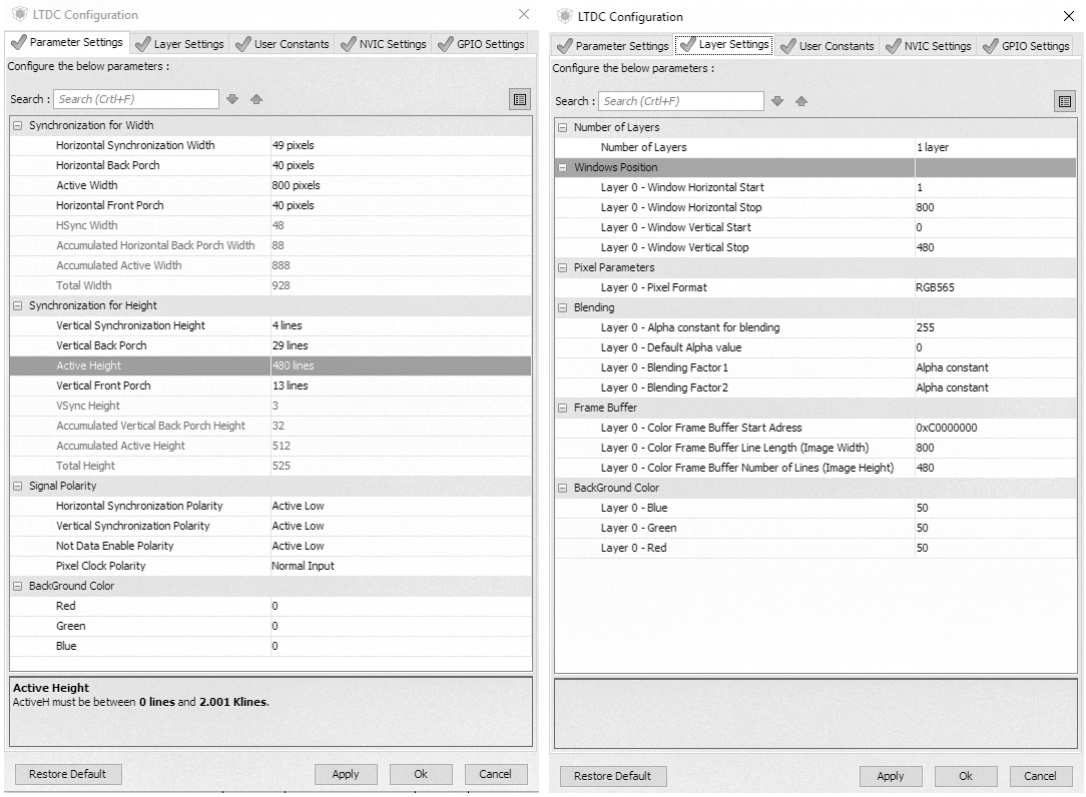
3.1.2 Nastavení LTDC periferie

Nastavení synchronizačních parametrů LTDC periferie proběhlo podle tabulky 2.1. kde horizontal synchronization width je nastavena na 48 hodinových pulzů. Horizontal front porch je nastaven na 40 hodinových pulzů. Active width je nastavena na 800 hodinových pulzů. Hodnota horizontal back porch je získána jako $Horizontalbackporch = Horizontalblanking - Horizontalpulsewidth$, což je $88 - 48 = 40$ hodinových pulzů. Vertical synchronization heigth je nastavena na 3 řádky. Parametr Vertical front porch je nastavena na 13 řádek. Active Heigth je nastavena na 480 řádek. Hodnota Vertical back porch je získána jako $Verticalbackporch = Verticalblanking - Verticalpulsewidth$,



Obrázek 3.2: Nastavení rozvodu hodinového signálu v programu STM32CubeMX

což je $32 - 3 = 29$ hodinových pulzů. Nastavení synchronizační parametrů je zobrazeno na obrázku 3.3 a). Kód pro inicializaci periferie je generován programem STM32CubeMX.



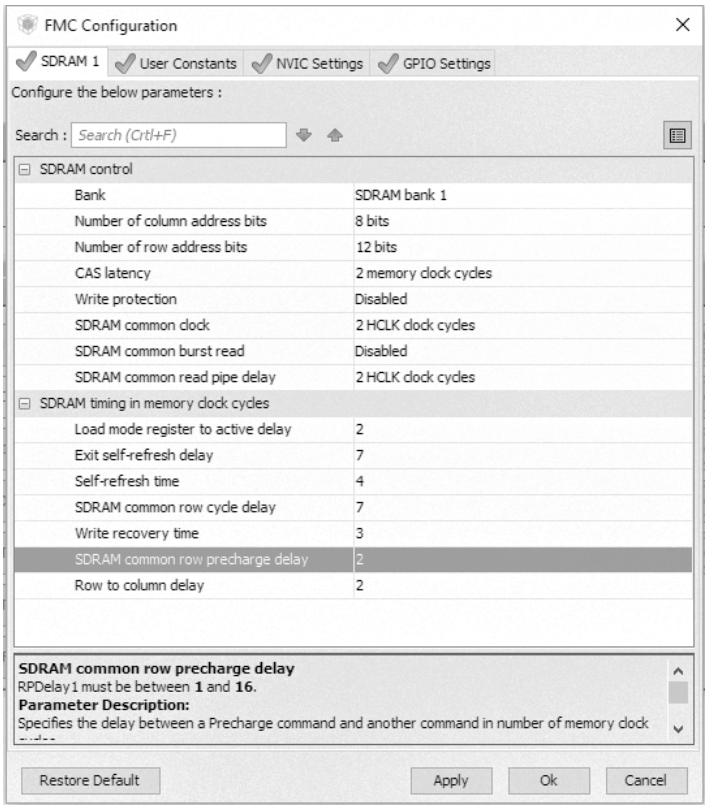
(a) Nastavení vertikální a horizontální synchronizace LTDC periferie v programu STM32CubeMX

(b) Nastavení vrstev displeje v programu STM32CubeMX

Obrázek 3.3: Nastavení LTDC periferie v programu STM32CubeMX

3.1.3 Nastavení FMC periferie

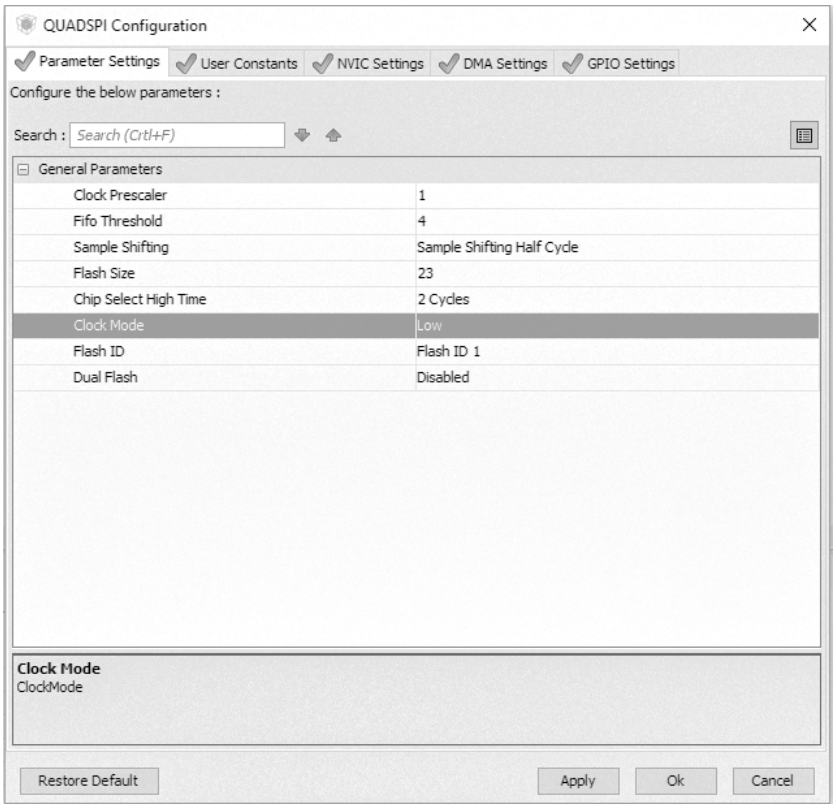
K FMC periferii je připojena externí 64 Mbit SDRAM pamět. Pro adresu sloupce je použito 8 bitů a pro adresu řádlů 12 bitů. CAS latency byla nastavena na 2 hodinové cykly. SDRAM common clock a SDRAM common read pipe delay jsou nastaveny na 2 hodinové cykly. Časový parametr Load mode register to active delay je nastaven na 2 hodinové cykly, parametr Exit self–Refresh delay na 7 hodinových cyklů, parametr Self–refresh time je nastaven na 4 hodinové cykly, parametr SDRAM common row cycle delay je nastaven na hodnotu 7 hodinových cyklů, parametr Write recovery time je nastaven na 3 hodinové cykly, parametr SDRAM common row delay má hodnotu 2 hodinových cyklů, parametr Row to column delay je nastavena na hodnotu 2 hodinových cyklů. Celé nastavení je na obrázku 3.4.[3]



Obrázek 3.4: Nastavení Quad–SPI periferie v programu STM32CubeMX

3.1.4 Nastavení QUADSPI periferie

Na periferii Quad–SPI je připojena flash paměť Micron. Flash paměť je schopná pracovat s maximální frekvencí 108 MHz, proto musí být předdělička Quad–SPI periferie nastavena na hodnotu 1. Při tomto nastavení je frekvence sběrnice AHB dělena 2. Flash size je nastavena na hodnotu 23, to odpovídá hodnotě 24 bitů. Část nastavení byla převzata z [12]. Nastavení vzorkovacího posuvu je o polovinu hodinového cyklu. Délka chip selectu je 2 hodinové cykly. Clock mode je nastaven na low. Nastavení Quad–SPI periferie je v programu STM32CubeMX je na obrázku 3.5.



Obrázek 3.5: Nastavení Quad–SPI periferie v programu STM32CubeMX

3.1.5 Nastavení UART periferií

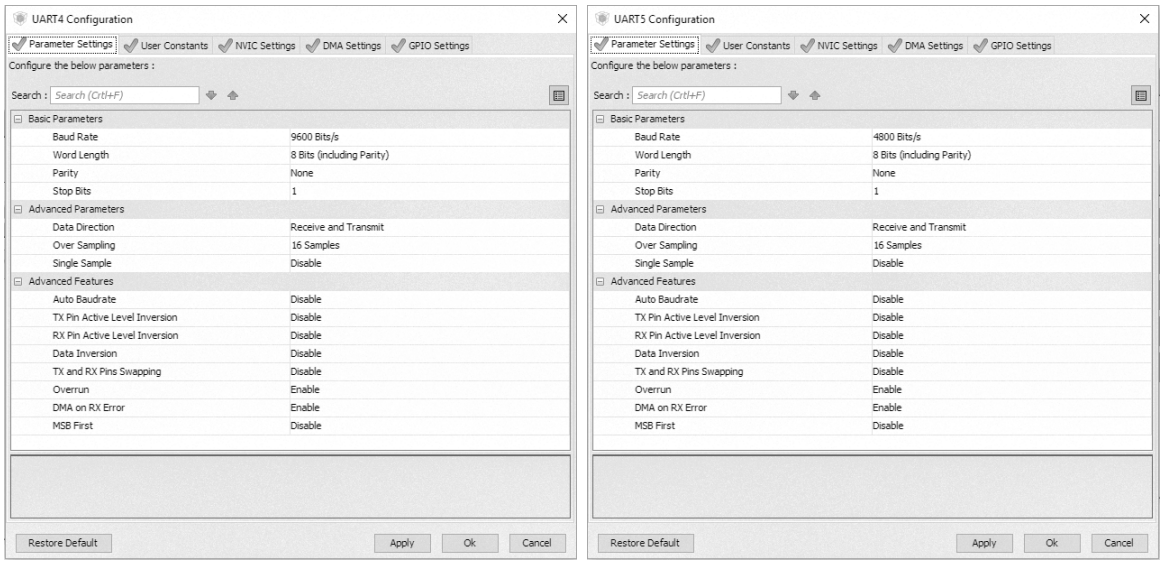
3.1.5.1 UART4 periferie

Periferie UART4 je připojena k XBeePro modulu. Přenosová rychlost je nastavena na 9600 Baud, délka dat je 8 bit, bez parity a jeden start bit. Periferie umožňuje vysílání i přijímání dat. Pro příjem je nastaveno šestnácti násobné převzorkování. Nastavení je vidět na obrázku 3.6 a).

3.1.5.2 UART5 periferie

Periferie UART5 je připojena k GPS přijímači. Přenosová rychlost je nastavena na 4800 Baud, délka dat je 8 bit, bez parity a jeden start bit. Periferie umožňuje jak vysílat tak přijímat data. Pro příjem je nastaveno šestnácti násobné převzorkování.

Nastavení je vidět na obrázku 3.6 b).



(a) Nastavení periferie UART4 v programu STM32CubeMX

(b) Nastavení periferie UART5 v programu STM32CubeMX

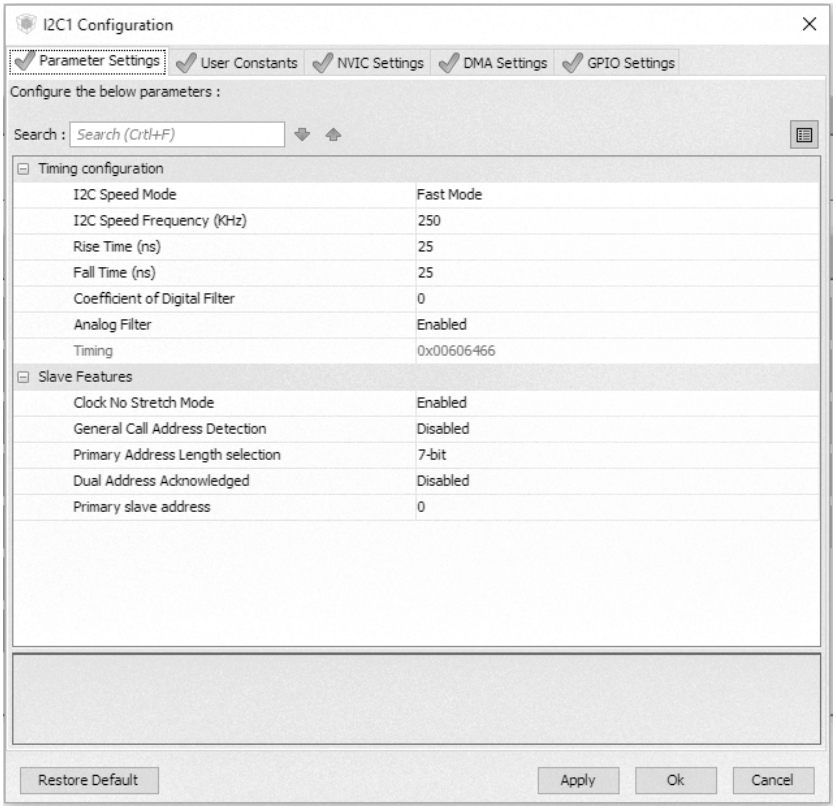
Obrázek 3.6: Nastavení UART periferií v programu STM32CubeMX

3.1.6 Nastavení I^2C periferie

Nastavení I^2C vychází z požadavků pro komunikaci s teplotním senzorem a akcelerometrem. I když jsou oba senzory schopné komunikovat rychlostí až 400 kHz, je použita komunikační rychlost jen 250 kHz. Dalšími nastavenými parametry jsou rychlost vzestupné a sestupné hrany, jenž jsou nastaveny na 25 ns. Hodnoty rychlosti vzestupné a sestupné hrany jsou nastaveny podle datasheetu akcelerometru.[6] Na obrázku 3.7 je nastavení I^2C periferie v programu STM32CubeMX.

3.1.7 Nastavení CAN periferie

CAN periferie je nastavena tak, aby rychlost CAN sběrnice byla 500 kBaud. Bit segment 1 je nastaven na 5 časových kvant, bit segment 2 je nastaven na 7 časových kvant a resynchronyzační skok na 3 časová kvanta. CAN periferie má hodinový signál odvozen od sběrnice APB1 (54 MHz). Potřebná velikost časového kvanta pro rychlost

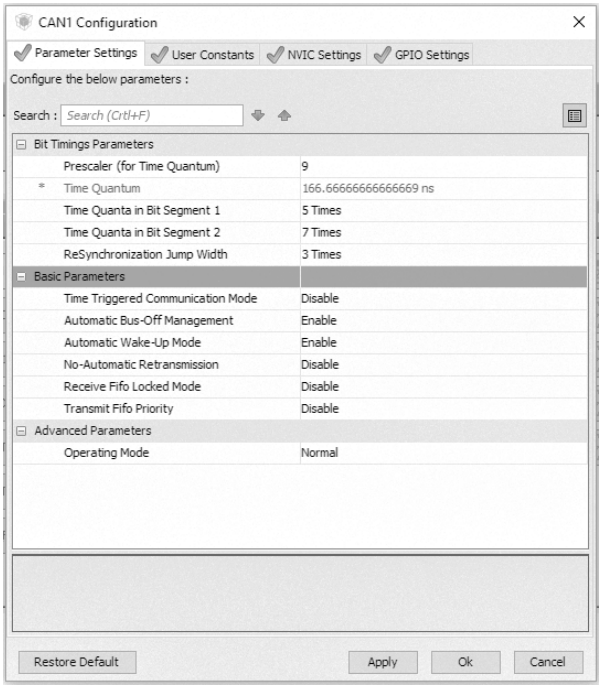


Obrázek 3.7: Nastavení I^2C periferie v programu STM32CubeMX

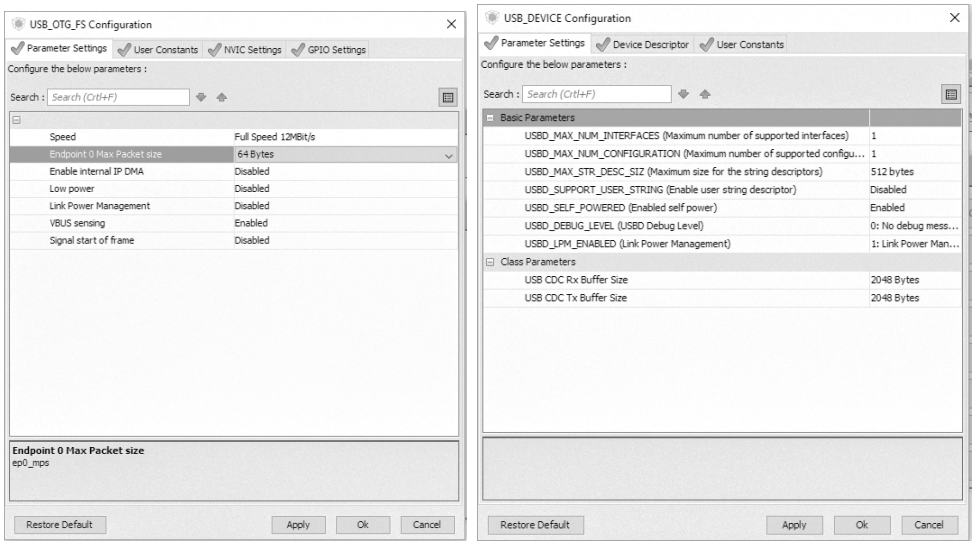
sběrnice 500 kBaud s uvedenými velikostmi bit segmentů 1 a 2 je 166.66 ns, k tomu je potřeba nastavit předděličku CAN periferie na hodnotu 9. Dále je nastaveno automatické odpojení od sběrnice v případě chyb na sběrnici. Celé nastavení v programu STM32CubeMX je na obrázku 3.8.

3.1.8 Nastavení USB periferie

USB periferie je nastavena na rychlost Full Speed 12 MBit/s a maximální velikost paketu je nastavena na 64 Bajtů. Nastavení periferie je na obrázku 3.9 a). Pro snadnější ovládání byla pro USB periferii přidána vrstva middlewaru, která je schopná přednastavit USB tak, že se tváří jako virtuální COM port, Mass storage nebo další zařízení. Pro účely jednotky displeje byl vybrán virtuální COM port. Jeho nastavení je na obrázku refobr˙USB˙cube b).



Obrázek 3.8: Nastavení CAN periferie v programu STM32CubeMX



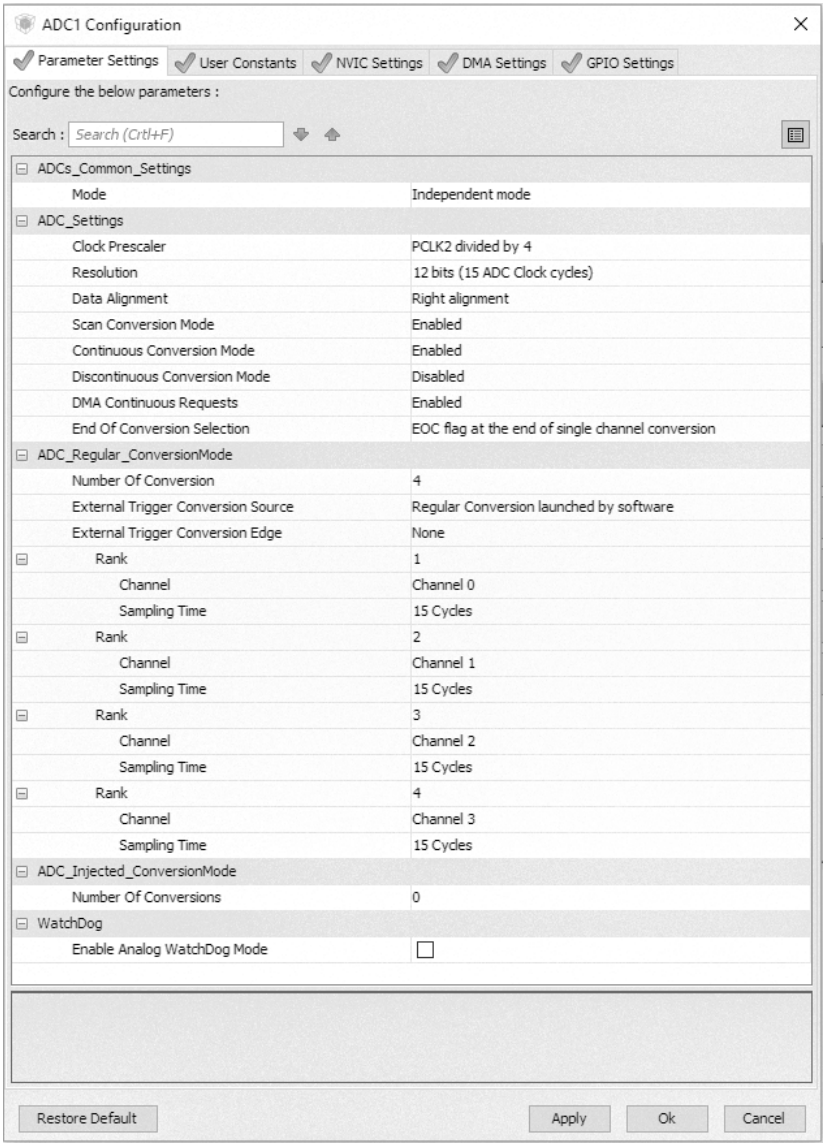
(a) Nastavení periferie USB v programu STM32CubeMX

(b) Nastavení USB middleware vrstvy v programu STM32CubeMX

Obrázek 3.9: Nastavení USB periferií v programu STM32CubeMX

3.1.9 Nastavení ADC periferie

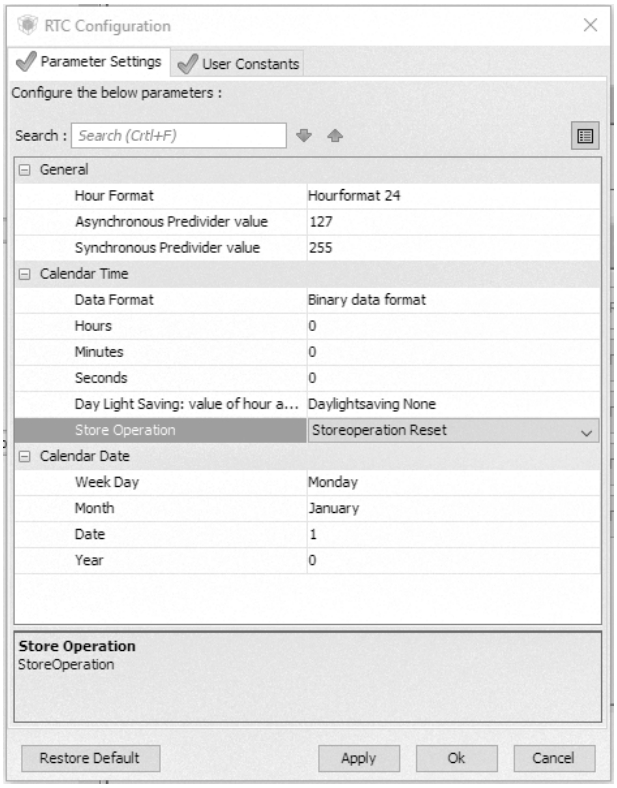
AD převodník je nastaven na maximální rozlišení 12 bit. Jsou použity 4 kanály převodníku (0 –3). Konverze kanálů je nastavena tak, aby se neustále opakovala a výsledky převodu jednotlivých kanálů jsou přenášeny pomocí DMA kanálu. Start převodu AD převodníku je pomocí softwaru. Na obrázku 3.10 je vyobrazeno nastavení AD převodníku v programu STM32CubeMX.



Obrázek 3.10: Nastavení ADC periferie v programu STM32CubeMX

3.1.10 Nastavení RTC periferie

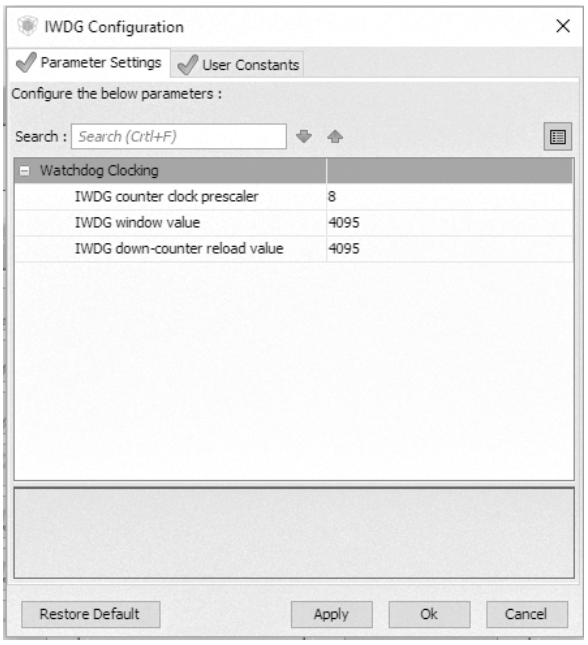
Pro periferii RTC byl nastaven 24 hodinový formát času a hodnoty asynchronní a synchronní předděličky jsou nastaveny na 127 a 255. To zajistí společně s použitím hodinového signálu z krystalového rezonátoru s frekvencí 32.768 kHz, že se sekundový registr bude inkrementovat s frekvencí 1 Hz. Kódování času bylo zvoleno binární. Hodnoty data a času bude nastavovat aplikace, proto nejsou v inicializaci nastaveny. Celé nastavení je na obrázku 3.11.



Obrázek 3.11: Nastavení RTC periferie v programu STM32CubeMX

3.1.11 Nastavení IWDG periferie

IWSDG je nastaven tak, aby resetoval mikrokontrolér v případě, že nebude hodnota jeho čítače obnovena každou sekundu.Zdrojem hodinového signálu pro IWDG je vnitřní 32 kHz rezonátor mikrokontroléru, proto je nastavena předdělička IWDG čítače na 8 a parametr IWDG douwn–counter reload value na 4095. Nastavení v programu STM32CubeMX je na obrázku **??**.



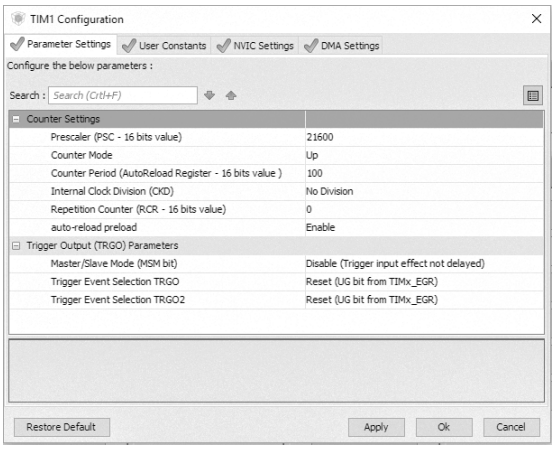
Obrázek 3.12: Nastavení IWDG periferie v programu STM32CubeMX

3.1.12 Nastavení TIMx periferií

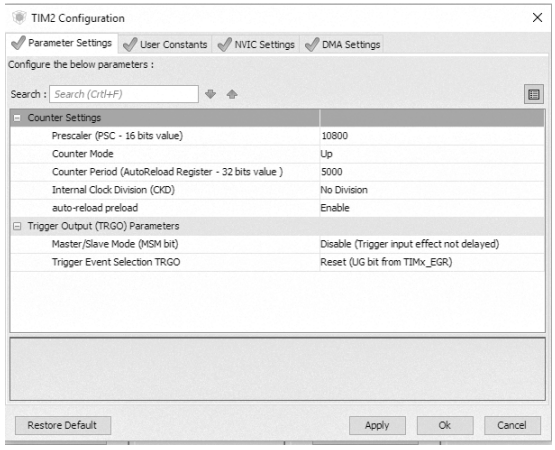
Časovače mikrokontroléru slouží jako časové základny pro vykonávání obslužných podprogramů. Pro tuto činnost byly nakonfigurovány časovače TIM1 až TIM5. Časovač TIM1 má nastavenou předděličku na 21600 a reload register na 100, díky tomu dojde k vyvolání interuptu každých 10 ms. Časovače TIM2, TIM3, TIM4 a TIM5 mají nastavenou předděličku na 10800. Hodnota reload registru pro TIM2 je 5000, což dopovídá 500 ms, pro TIM3 je 500, což odpovídá 50 ms. TIM4 má nastaven reload register na 1000, to odpovídá 100 ms a TIM5 má reload register nastaven na 10000, takže k jeho přetečení dojde každou 1 sekundu. Nastavení všech časovaču je na obrázku 3.13

3.1.13 Nastavení GPIO pinů

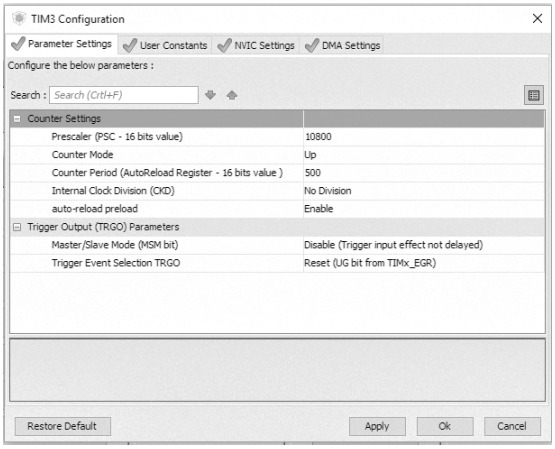
GPIO piny, které nespadají k žádné z předešlých periferií, jsou nastaveny podle obrázku 3.14 podle jejich smyslu na vstupní nebo výstupní.



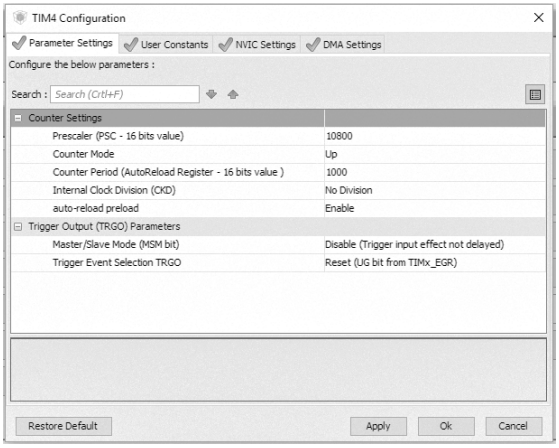
(a) Nastavení TIM1 periferie v programu STM32CubeMX



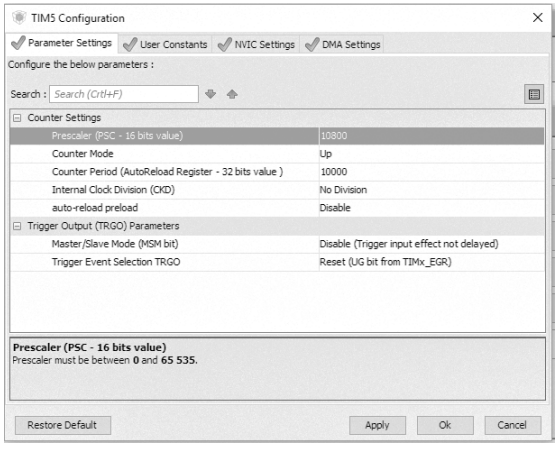
(b) Nastavení TIM2 periferie v programu STM32CubeMX



(c) Nastavení TIM3 periferie v programu STM32CubeMX

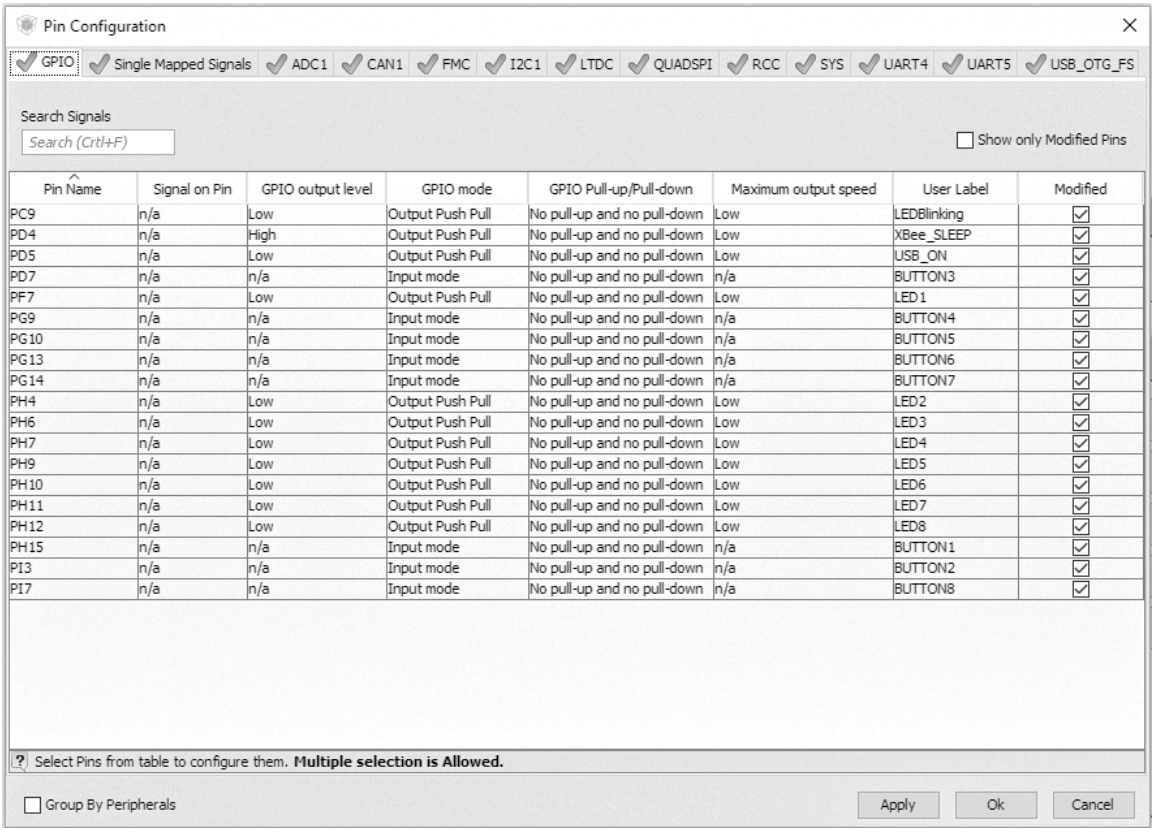


(d) Nastavení TIM4 periferie v programu STM32CubeMX



(e) Nastavení TIM5 periferie v programu STM32CubeMX

Obrázek 3.13: Nastavení TIMx periferií v programu STM32CubeMX



Pin Name	Signal on Pin	GPIO output level	GPIO mode	GPIO Pull-up/Pull-down	Maximum output speed	User Label	Modified
PC9	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LEDBlinking	☑
PD4	n/a	High	Output Push Pull	No pull-up and no pull-down	Low	XBee_SLEEP	☑
PD5	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	USB_ON	☑
PD7	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	BUTTON3	☑
PF7	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LED1	☑
PG9	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	BUTTON4	☑
PG10	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	BUTTON5	☑
PG13	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	BUTTON6	☑
PG14	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	BUTTON7	☑
PH4	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LED2	☑
PH6	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LED3	☑
PH7	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LED4	☑
PH9	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LED5	☑
PH10	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LED6	☑
PH11	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LED7	☑
PH12	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	LED8	☑
PH15	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	BUTTON1	☑
PI3	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	BUTTON2	☑
PI7	n/a	n/a	Input mode	No pull-up and no pull-down	n/a	BUTTON8	☑

Obrázek 3.14: Nastavení GPIO pinů v programu STM32CubeMX

3.2 Knihovna pro ovládání displeje

Pro vykreslování znaků na displej byla vytvořena knihovna "LCD.h"a "LCD.c". Fonty pro vykreslování textu a dalších znaků jsou uloženy v souborech "fonts.h"a "fonts.c". Fonty použité v DP byly staženy z webové stránky [13]. Jedná se o fonty: GroteskBold16x32, GroteskBold24x48, GroteskBold32x64, arial˙bold a Various˙Symbols˙32x32. Program přistupuje k fontům skrze strukturu _tFont. Struktura je uložena v souboru "fonts.h"Struktura fontu vypadá takto:

```
1 typedef struct _tFont
2 {
3   uint8_t *table;
4   uint16_t Width;
5   uint16_t Height;
6
7 } sFONT;
```

Strukrura _tFont obsahuje šířku a výšku daného fontu v pixelech a pointer na pole hodnot fontu. Data fontů jsou uloženy v souboru "fonts.c". Pro vykreslování na displej

byly vytvořeny následující funkce:

```
1  void LCD_SetTextColor(uint16_t Color);
2  void LCD_SetBackColor(uint16_t Color);
3  void LCD_SetFont(sFONT *fonts);
4  void LCD_ClearALL(void);
5  void LCD_Draw_HorizontalLine(uint16_t x_Start, uint16_t y_Start, uint16_t Length, uint16_t Width, uint16_t Color);
6  void LCD_Draw_VerticalLine(uint16_t x_Start, uint16_t y_Start, uint16_t Length, uint16_t Width, uint16_t Color);
7  void LCD_Draw_Rectangle(uint16_t x_Start, uint16_t y_Start,uint16_t Width, uint16_t Height, uint16_t WidthOfLine,
8   uint16_t Color);
9  void LCD_Draw_FilledRectangle(uint16_t x_Start, uint16_t y_Start,uint16_t Width, uint16_t Height, uint16_t Color);
10 void LCD_Draw_VerticalBar(VerticalBar *VertBar);
11 void LCD_Draw_VerticalBarZero(VerticalBarZero *VertBar);
12 void LCD_Draw_Char(uint16_t Xpos, uint16_t Ypos,  uint8_t *c);
13 void LCD_Display_Char(uint16_t Y, uint16_t X, uint8_t Ascii);
14 void LCD_Display_String(uint16_t x, uint16_t y, uint8_t *ptr, uint16_t LengthOfString);
15 void LCD_Display_Number(uint16_t x, uint16_t y, float Number, NumberOfDecimalPlace DecimalPlace);
16 void LCD_Display_ChannelLabel(ChannelLabel *LabelStruct);
17 void LCD_Display_UniversalPage(uint8_t UniPage);
18 void LCD_Display_Page(int8_t Page);
19 void LCD_GetDigits(uint32_t Number, uint8_t Digits[5] );
```

Údaje o pixelech displeje jsou uloženy s barevnou hloubkou 16 bit, kdy 5bit rozsah má červená a modrá barva má rozsah 6bit. Funkce `LCD_SetTextColor` nastaví barvu textu. Funkce `LCD_SetBackColor` nastaví barvu pozadí. Funkce `LCD_SetFont` nastaví font, kterým se bude vykreslovat znaky na displej. Funkce `LCD_ClearALL` vymaže obsah displeje. Použije přitom barvu nastavenou funkcí `LCD_SetBackColor`.

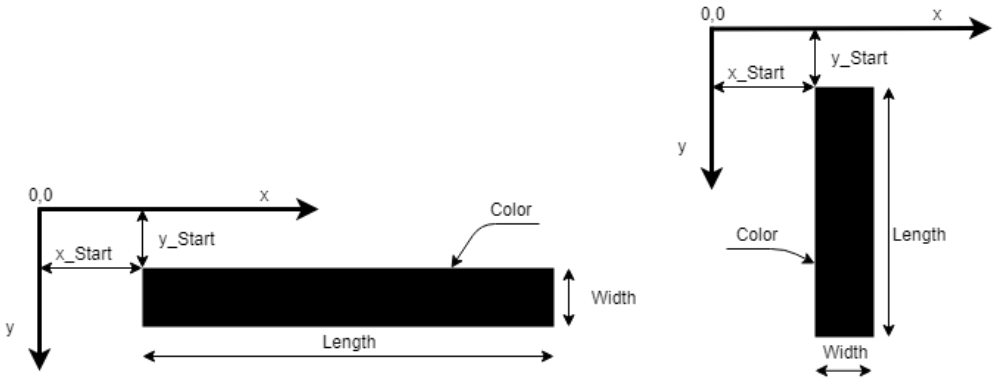
Funkce, které kreslí na displej jsou označeny předponou `LCD_Draw`. Pro vykreslení vodorovné čáry byla vytvořena funkce `LCD_Draw_HorizontalLine`, pro kreslení svislé čáry `LCD_Draw_VerticalLine`, pro kreslení obdélníku `LCD_Draw_Rectangle`, vyplněného obdélníku `LCD_Draw_FilledRectangle`, vertikálního baru `LCD_Draw_VerticalBar` a vertikálního baru se středovou hodnotou `LCD_Draw_VerticalBarZero`.

3.2.1 Funkce LCD_Draw_HorizontalLine

Funkce `LCD_Draw_HorizontalLine` má vstupní parametry `x_Start` a `y_Start`, které obsahují souřadnice levého horního rohu čáry v pixelech od levého horního rohu displeje, dalším parametrem je `Length`, ten obsahuje délku čáry v pixelech, parametr `Width` obsahuje šířku čáry v pixelech a parametr `Color` obsahuje barvu čáry. Všechny parametry jsou vysvětleny na obrázku 3.15 a).

3.2.2 Funkce LCD_Draw_VerticalLine

Funkce `LCD_Draw_VerticalLine` má vstupní parametry `x_Start` a `y_Start`, které obsahují souřadnice levého horního rohu čáry v pixelech od levého horního rohu displeje, dalším parametrem je `Length`, ten obsahuje délku čáry v pixelech, parametr `Width` obsahuje šířku čáry v pixelech a parametr `Color` obsahuje barvu čáry. Všechny parametry jsou vysvětleny na obrázku 3.15 b).



(a) Parametry funkce LCD_Draw_HorizontalLine (b) Parametry funkce LCD_Draw_VerticalLine

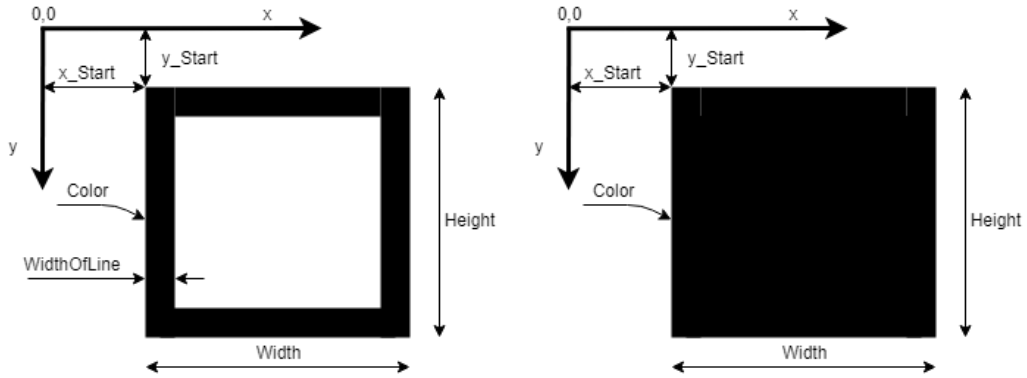
Obrázek 3.15: Parametry funkce LCD_Draw_HorizontalLine a LCD_Draw_VerticalLine

3.2.3 Funkce LCD_Draw_Rectangle

Funkce `LCD_Draw_Rectangle` má vstupní parametry `x_Start` a `y_Start`, které obsahují souřadnice levého horního rohu obdélníku v pixelech od levého horního rohu displeje, dalším parametrem je `Width`, ten obsahuje šířku obdélníku v pixelech, parametr `Height` obsahuje výšku obdélníku v pixelech, parametr `WidthOfLine` obsahuje šířku čáry obdélníku a parametr `Color` obsahuje barvu obdélníku. Všechny parametry jsou vysvětleny na obrázku 3.16 a).

3.2.4 Funkce LCD_Draw_FilledRectangle

Funkce `LCD_Draw_FilledRectangle` má vstupní parametry `x_Start` a `y_Start`, které obsahují souřadnice levého horního rohu obdélníku v pixelech od levého horního rohu displeje, dalším parametrem je `Width`, ten obsahuje šířku obdélníku v pixelech, parametr `Height` obsahuje výšku obdélníku v pixelech a parametr `Color` obsahuje barvu obdélníku. Všechny parametry jsou vysvětleny na obrázku 3.16 b).



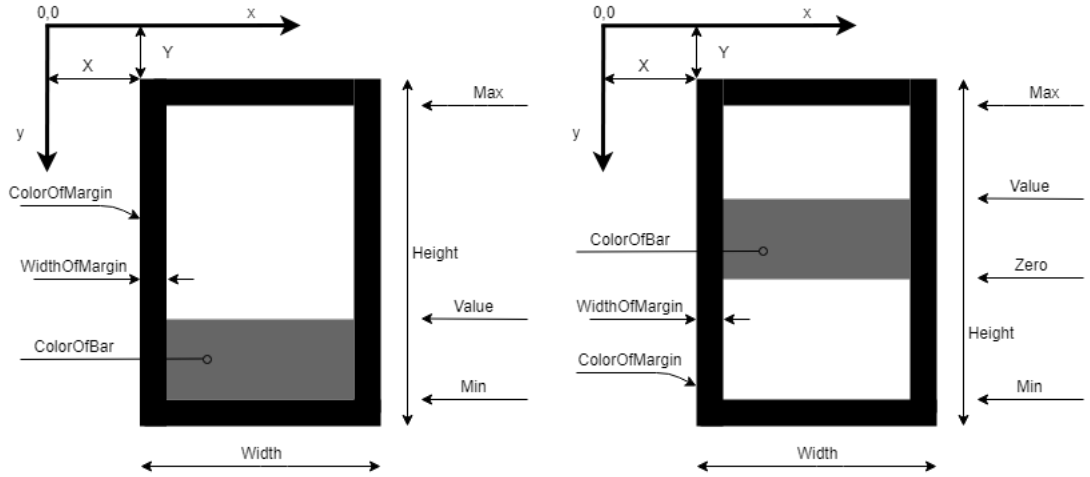
(a) Parametry funkce LCD_Draw_Rectangle (b) Parametry funkce LCD_Draw_FilledRectangle

Obrázek 3.16: Parametry funkcí LCD_Draw_Rectangle a LCD_Draw_FilledRectangle

3.2.5 Funkce LCD_Draw_VerticalBar

Funkce `LCD_Draw_VerticalBar` má jako vstupní parametr strukturu `VerticalBar`. Tato struktura obsahuje položky `X` a `Y`, které obsahují pozici levého horního rohu vertikálního baru od levého horního rohu displeje v pixelech. Dalšími položkami jsou `Widt` a `Height`. Položka `Width` obsahuje šířku baru v pixelech a položka `Height` obsahuje výšku baru v pixelech. Položka `ColorOfBar` obsahuje barvu aktivní části baru. Dalšími položkami jsou `WidthOfMargin` a `ColorOfMargin`. První z nich obsahuje šířku ohraničení baru a druhá barvu tohoto ohraničení. Položky `Min` a `Max` obsahují hodnotu minima a maxima vertikálního baru. Poslední položkou je `Value`, která obsahuje hodnotu pro vykreslení aktivní části vertikálního baru. Jednotlivé položky jsou vyobrazeny na obrázku 3.17 a).

```
1  typedef struct
2  {
3    uint16_t X;
4    uint16_t Y;
5    uint16_t Width;
6    uint16_t Height;
7    uint16_t WidthOfMargin;
8    uint16_t ColorOfMargin;
9    uint16_t ColorOfBar;
10   float Min;
11   float Max;
12   float Value;
13
14 } VerticalBar, *pVerticalBar;
```



(a) Parametry funkce LCD_Draw_VerticalBar (b) Parametry funkce LCD_Draw_VerticalBarZero

Obrázek 3.17: Parametry funkcí LCD_Draw_VerticalBar a LCD_Draw_VerticalBarZero

3.2.6 Funkce LCD_Draw_VerticalBarZero

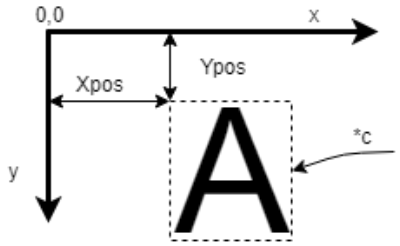
Funkce `LCD_Draw_VerticalBarZero` má jako vstupní parametr strukturu `VerticalBarZero`. Tato struktura obsahuje položky `X` a `Y`, které obsahují pozici levého horního rohu vertikálního baru od levého horního rohu displeje v pixelech. Dalšími položkami jsou `Widt` a `Height`. Položka `Width` obsahuje šířku baru v pixelech a položka `Height` obsahuje výšku baru v pixelech. Položka `ColorOfBar` obsahuje barvu aktivní části baru. Dalšími položkami jsou `WidthOfMargin` a `ColorOfMargin`. První z nich obsahuje šířku ohraničení baru a druhá barvu tohoto ohraničení. Položky `Min` a `Max` obsahují hodnotu minima a maxima vertikálního baru. Položka `Zero` obsahuje velikost střední hodnoty vertikálního baru. Poslední položkou je `Value`, která obsahuje hodnotu

pro vykreslení aktivní části vertikálního baru. Jednotlivé položky jsou vyobrazeny na obrázku 3.17 b).

```
1  typedef struct
2  {
3    uint16_t X;
4    uint16_t Y;
5    uint16_t Width;
6    uint16_t Height;
7    uint16_t WidthOfMargin;
8    uint16_t ColorOfMargin;
9    uint16_t ColorOfBar;
10   float Min;
11   float Zero;
12   float Max;
13   float Value;
14
15 } VerticalBarZero, *pVerticalBarZero;
```

3.2.7 Funkce LCD_Draw_Char

Funkce `LCD_Draw_Char` slouží pro vykreslení znaku. Vstupní parametry funkce `LCD_Draw_Char` jsou `Xpos`, `Ypos` a `*c`. Parametry `Xpos`, `Ypos` odkazují na pozici levého horního rohu znaku od levého horního okraje displeje. Parametr `*c` je ukazatel na znak, který má být vykreslen. Jednotlivé parametry jsou vyobrazeny na obrázku 3.18.



Obrázek 3.18: Parametry funkce LCD–DrawChar

3.2.8 Funkce LCD_Display_Char

Funkce `LCD_Display_Char` vykresluje pomocí funkce `LCD_Draw_Char` znak fontu, který je nastaven v globální proměnné `LCD_Currentfonts`. Vstupní parametry funkce `LCD_Display_Char` jsou `X`, `Y` a `Ascii`. Parametry `X`, `Y` odkazují na pozici levého horního rohu znaku od levého horního okraje displeje. Parametr `Ascii` je Ascii hodnota znaku, který má být vykreslen.

3.2.9 Funkce LCD_Display_String

Funkce `LCD_Display_String` vykresluje skupinu znaků pomocí funkce `LCD_Display_Char`. Její vstupní parametry jsou `X`, `Y` odkazují na pozici levého horního rohu prvního znaku od levého horního okraje displeje. Parametr `*ptr` je ukazatel na pole znaku, které má být vykresleno a parametr `LengthOfString` obsahuje počet znaků, které mají být vykresleny.

3.2.10 Funkce LCD_GetDigits

Funkce `LCD_GetDigits` slouží pro získání digitů čísla. Její vstupní parametr je `Number`, který obsahuje číslo, jehož digity chceme získat a parametr `Digits[5]`, do kterého se budou ukládat jednotlivé digity. Kdy pozice 0 v poli `Digits` odpovídá desetitisícům a pozice 5 jednotkám.

3.2.11 Funkce LCD_Display_Number

Funkce `LCD_Display_Number` vykresluje na displej číselnou hodnotu. Její vstupní parametry jsou `x`, `y`, `Number` a `DecimalPlace`. Parametry `x` a `y` obsahují pozici levého horního rohu prvního znaku zobrazovaného čísla od levého horního rohu displeje v pixelech. Parametr `Number` obsahuje hodnotu zobrazovaného čísla a parametr `DecimalPlace` obsahuje počet desetinných míst, na které má být číslo zobrazeno. Počet desetinných míst může být volen od 0 do 3, kdy 0 znamená, že číslu nebudou zobrazena desetinná místa a 3 znamená, že bude zobrazeno s rozlišením na tisíciny.

3.2.12 Funkce LCD_Display_Page

Funkce `LCD_Display_Page` slouží pro přepínání obrazovek displeje. Jejím vstupním parametrem je `Page`. Pokud má tento parametr hodnotu 1 zobrazí se na displeji následující stránka. Pokud má hodnotu -1 zobrazí se předchozí stránka. V případě, že je parametr nulový je vykreslena současná stránka znovu.

Funkce `LCD_Display_UniversalPage` slouží pro vykreslení obrazovky, která byla označena jako ,,Univerzální". Tato obrazovka umožňuje zobrazit až 9 hodnot a jejich popis pomocí funkce `LCD_Display_ChannelLabel`. V příloze A.1 je fotografie jednotky displeje vykreslující ,,Univerzální"obrazovku. V horní části obrazovky jsou vypsány stavy pro komunikaci s PC aplikací. První číslo vyjadřuje připravenost jednotky přijmout příkaz, další znaky oznamují chyby při příjmu nebo vysílání.

Funkce `LCD_Display_ChannelLabel` vykresluje na displej základní prvek pro zobrazení dat na ,,Univerzální"obrazovce displeje.

Obrázek 3.19: Vyobrazení vykreslení funkce LCD_Display_ChannelLabel

3.3 Knihovna pro ovládání akcelerometru

Pro ovládání akcelerometru byly vytvořeny funkce: `InitAccelerometer`, `SetFullScale`, `SetRegister`, `ReadRegister`, `ReadAccelerationXYZ`, `GetAccelerationXYZ`. Funkce `InitAccelerometer` aktivuje snímání akcelerace v ose x,y,z a nastaví režim akcelerometru na normal mode s frekvencí výstupních dat 100 Hz. Dále funkce nastaví maximální rozsah akcelerometru na ± 6 g. Funkce `SetFullScale` mění maximální rozsah

akcelerometru buď na ± 6 g/±12 g/±24g.Funkce `SetRegister` nastaví registr akcelerometru na požadovanou hodnotu. Funkce `ReadRegister` vyčte hodnotu registru akcelerometru. Funkce `ReadAccelerationXYZ` vyčte registry akcelerometru, které obsahují akceleraci. Funkce `GetAccelerationXYZ` vyčte pomocí funkce `ReadAccelerationXYZ` registry akcelerometru a podle nastaveného maximálního rozsahu akcelerometru, přepočte údaje registrů na reálné hodnoty akcelerace. Prototypy všech funkcí jsou uvedeny zde:

```
1 uint8_t SetFullScale(uint8_t FullScale);
2 uint8_t InitAccelerometer(void);
3 uint8_t SetRegister(uint8_t RegisterAddress, uint8_t RegValue);
4 uint8_t ReadRegister(uint8_t RegisterAddress);
5 uint8_t ReadAccelerationXYZ(int16_t *AccelerationX, int16_t *AccelerationY, int16_t *AccelerationZ);
6 uint8_t GetAccelerationXYZ(float *AccelerationX, float *AccelerationY, float *AccelerationZ);
```

3.4 Knihovna pro ovládání teploměru

Teplotní senzor nevyžaduje žádnou inicializaci, proto je možné ihned vyčíst teplotu pomocí funkce `GetRawTemperature` nebo `GetTemperature`. Funkce `GetRawTemperature` vrací teplotu vyčtenou z teplotního snímače v datovém typu int16_t nebo v datovém typu float v případě funkce `GetTemperature`. Teplotní senzor je možné pro úsporu energie přepnout do low power módu pomocí funkce `TemperatureSensor_Shutdown\verb` a funkce `TemperatureSensor_WakeUp` přepne teplotní senzor do normálního stavu. Prototypy funkcí jsou uvedeny zde:

```
1 int16_t GetRawTemperature(void);
2 uint8_t TemperatureSensor_Shutdown(void);
3 uint8_t TemperatureSensor_WakeUp(void);
4 float GetTemperature(void);
```

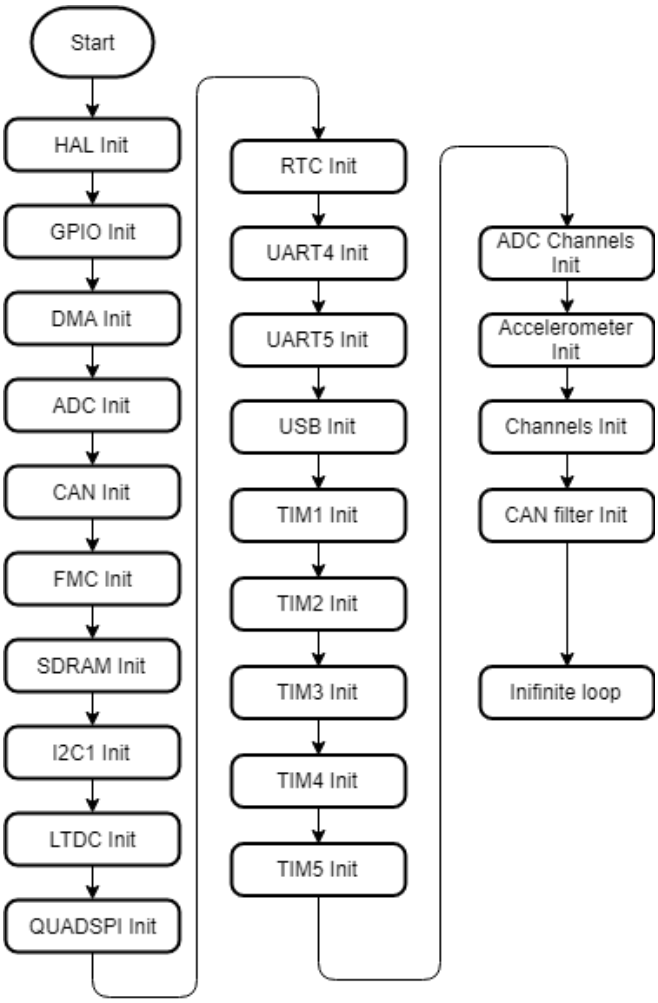
3.5 Knihovna pro ovládání ADC

Pro zapnutí A/D převodníku je potřeba zavolat funkci `Pedals_Init_and_Start`, která má jako vstupní parametry pointer na data z převodníku a počet dat. Funkce zapne A/D převodník a nastaví DMA přenos z A/D převodníku do proměnné, na kterou ukazuje pointer.

3.6 Postup inicializace komponent při startu jednotky displeje

Inicializace periferii mikrokontroléru probíhá podle vývojového diagramu na obrázku 3.20. Je nezbytné, aby před inicializací periferie LTDC, která zajišťuje ovládání displeje, byla inicializována SDRAM paměť, protože paměť obsahuje data pro vykreslení displeje.

Funkce `Channels_Init` zajišťuje vyčtení kalibračních dat z externí flash paměti a jejich zařazení do příslušných globálních proměnných, které slouží jako zdroj dat pro vykreslování na displej a nastavení filtračních registrů CAN sběrnice. To je provedeno ve funkci `CAN_filter_Init`.



Obrázek 3.20: Vývojový diagram inicializace jednotky displeje

4 Konfigurační aplikace pro PC

4.1 Komunikační protokol

4.1.1 Transportní protokol

Pro komunikaci mezi PC aplikací a jednotkou displeje je využit transportní protokol na sběrnici UART. Začátek zprávy transportního protokolu je uvozen bajtem s hodnotou 0x55. Dalším bajtem je délka dat ve zprávě, která mohou být dlouhá od 0 do 255. Po bajtu s délkou zprávy následují data a poté následuje bajt s kontrolním součtem, který se provádí přes všechna data. Transportní protokol je znázorněn v tabulce 4.1.

Start	Délka dat	Data	Kontrolní součet
byte	byte	0–255 byte	byte
0x55	xx	xx ... xx	xx

Tabulka 4.1: Transportní protokol mezi aplikací a jednotkou displeje

4.1.2 Aplikační příkazy

Pro ovládání displeje podporuje jednotka displeje několik příkazů (Servisů), přes rozhraní UART pomocí transportního protokolu. Příkazy se rozeznávají podle prvního

bajtu v datech transportního protokolu. Pokud je příkaz poslán správně, je odpověď jednotky kladná. Kladná odpověď je o 0x40 větší než hodnota poslaného příkazu.

4.1.2.1 Příkaz pro čtení externí Flash paměti

Pro vyčtení dat z externí flash paměti byl vytvořen příkaz `ReadFlashData`. Tento příkaz se skládá ze sekvence bajtů, které jsou v tabulce 4.2.

Byte	Název	Hodnota
0	Service	0x23
1	Subservice	0
2	LengthOfData	6
3 – 6	Address	0 – 0xFFFFFF
7 – 8	Length	0 – 128

Tabulka 4.2: Tabulka příkazu ReadFlashData

4.1.2.2 Příkaz pro čtení dat kanálů

Pro vyčtení hodnot kanálů byl vytvořen příkaz `ReadChannelsData`. Tento příkaz se skládá ze sekvence bajtů, které jsou v tabulce 4.3.

Byte	Název	Hodnota
0	Service	0x22
1	ChannelNumber	0–255

Tabulka 4.3: Tabulka příkazu ReadChannelsData

4.1.2.3 Příkaz pro mazání flash paměti

Pro smazání celé externí flash paměti byl vytvořen příkaz `DisplayControl-EraseChip`. Tento příkaz se skládá ze sekvence bajtů, které jsou v tabulce 4.4.

Byte	Název	Hodnota
0	Service	0x31
1	Subservice	1
2	Subsubservice	0

Tabulka 4.4: Tabulka příkazu DisplayControl–EraseChip

Pro smazání externí flash paměti po subsektorech byl vytvořen příkaz `DisplayControl-EraseSubSector`. Tento příkaz se skládá ze sekvence bajtů, které jsou v tabulce 4.5.

Byte	Název	Hodnota
0	Service	0x31
1	Subservice	2
2–3	SubSectorNumber	0–4095

Tabulka 4.5: Tabulka příkazu DisplayControl–ErasSubSector

4.1.2.4 Příkaz pro nastavení adresy a délky dat pro zápis dat do flash paměti

Pro nastavení adresy a délky dat pro programování externí flash paměti byl vytvořen příkaz `RequestDownload`. Tento příkaz se skládá ze sekvence bajtů, které jsou v tabulce 4.6.

Byte	Název	Hodnota
0	Service	0x34
1	Subservice	0
2	Number of Address and Length bytes	0x24
3–6	Address	0 –0xFFFFFF
7–8	Length	0–0x253

Tabulka 4.6: Tabulka příkazu RequestDownload

4.1.2.5 Příkaz pro přenos dat do flash paměti

Pro odeslání dat pro programování externí flash paměti byl vytvořen příkaz `TransferData`. Tento příkaz se skládá ze sekvence bajtů, které jsou v tabulce 4.7. Data jsou programována na adresu, která byla předtím odeslána pomocí příkazu `RequestDownload`.

Byte	Název	Hodnota
0	Service	0x36
1	Subservice	0
2–255	Data	0x00–0xff

Tabulka 4.7: Tabulka příkazu TransferData

4.1.2.6 Negativní odpovědi na příkazy

Negativní odpověď na příkaz jednotka odešle v případě, že byl příkaz špatně zadán nebo není podporován. Negativní odpověď má vždy první bajt s hodnotou 0x7F. Druhý bajt nese hodnotu příkazu, na který odpovídá a třetí bajt obsahuje kód, který odkazuje na chybu, která nastala.Význam jednotlivých Negativních kódů je v tabulce 4.9. Struktura negativní odpovědi je v tabulce 4.8.

Byte	Název	Hodnota
0	Negative	0x7F
1	Service	0–255
2	Negative code	0x00–0xff

Tabulka 4.8: Tabulka struktury negativní odpovědi

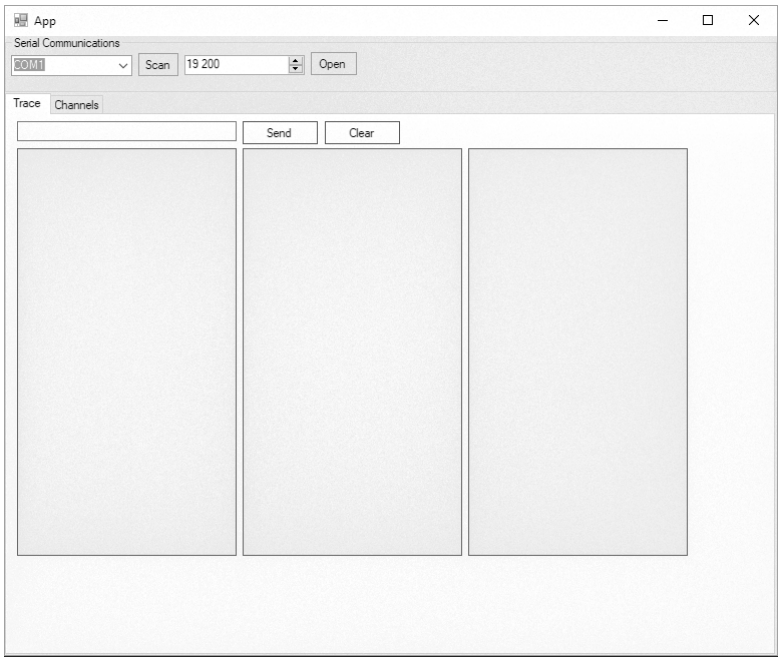
Hodnota	Význam
0x13	Špatná délka servisu
0x31	Špatný rozsah vstupních dat
0x22	Špatná sekvence příkazů
0x7E	Nepodporovaný subservis
0x7F	Nepodporovaný servis
0x80	Špatný kontrolní součet

Tabulka 4.9: Tabulka negativní kódů

4.2 Nastavení komunikačních a zobrazovacích kanálů

Pro komunikaci s jednotkou displeje a její konfiguraci slouží PC aplikace. V její horní části je panel pro nastavení COM portu a přenosové rychlosti, jak je vidět na obrázku 4.1. Na obrázku je vidět záložka Trace, která umožňuje uživateli posílat jednotce příkazy ručně a zároveň vypisuje odpovědi v hexadecimálním a dekadickém tvaru.

Druhá záložka s názvem Channels slouží pro nastavení jednotlivých kanálů displeje. Záložka je na obrázku 4.2. Pro vložení nového kanálu slouží pravá část aplikace, kde se vyplní všechny potřebné parametry. Prvním parametrem je Channel ID, který odkazuje na umístění kanálu na jednotlivých obrazovkách displeje, kdy Channel ID 0 až 8 jsou umístěny na jedné obrazovce viz obrázek 4.3. Další kanály jsou na dalších obrazovkách po devíti. Dalším parametrem je výběr zdroje dat, která bude kanál zobrazovat. Zdrojem dat může být CAN sběrnice (External) nebo jeden ze snímačů umístěných na jednotce displeje (Internal).

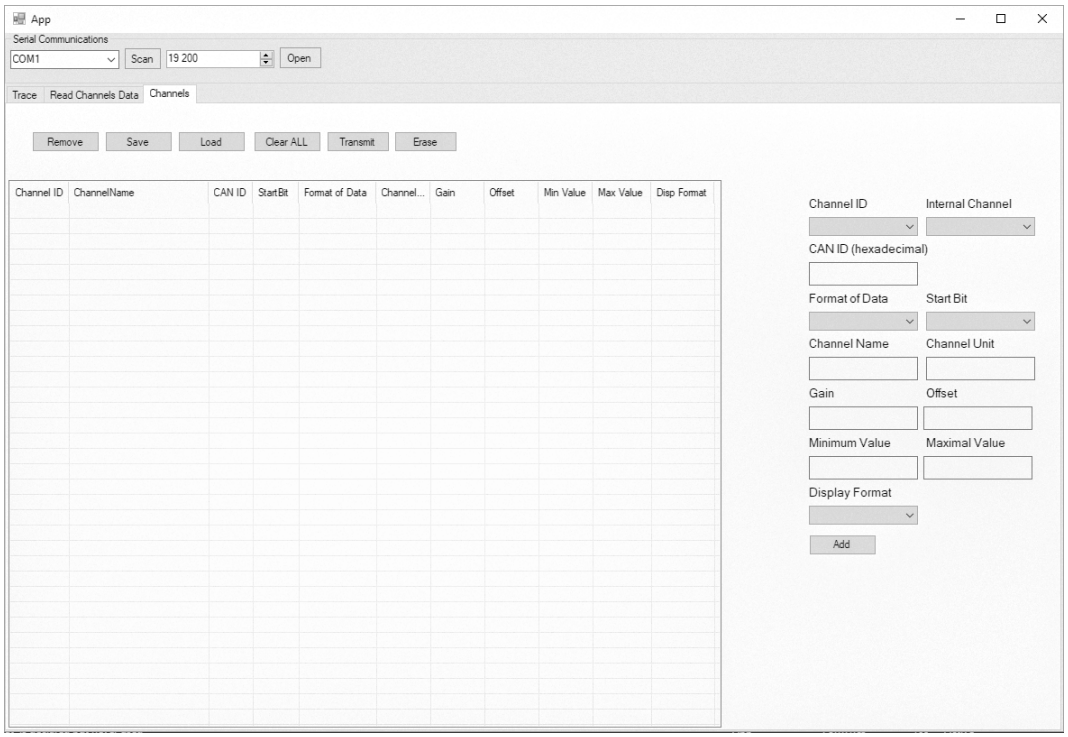


Obrázek 4.1: Aplikace pro komunikaci a nastavení displeje Trace záložka

V případě volby External, musí uživatel vyplnit CAN ID, formát, ve kterém jsou data vysílána a Start bit, který odkazuje na pozici bajtů v CAN zprávě. Dalšími parametry jsou název kanálu, jednotky (nepovinné), násobitel, ofset, minimální a maximální hodnota a formát, v jakém se bude hodnota zobrazovat.

V případě volby Internal, jsou parametry vyplněny automaticky. Uživateli není umožněno nastavit CAN ID, Start bit a formát, ve kterém jsou data vysílána. Ostatní parametry jsou předvyplněny, ale uživatel je může měnit. Těmito parametry jsou Název kanálu, jednotky (nepoviné), násobitel, ofset, minimální a maximální hodnota a formát, ve kterém se bude hodnota zobrazovat.

Přidané kanály se zobrazují v levé části aplikace. Pokud v seznamu vybereme některý z kanálů a zmáčkneme tlačítko Remove, kanál bude smazán. Pro smazání všech kanálů slouží tlačítko Clear ALL. Pro uložení nastavených kanálů slouží tlačítko Save, které otevře dialogové okno, do kterého uživatel zadá název souboru. Takto uložené kanály mohou být opětovně načteny přes tlačítko Load. Pro odeslání nastavení do jednotky displeje slouží tlačítko Transmit a pro smazání stávajícího nastavení v jednotce tlačítko Erase. Operase Transmit a Erase trvá dlouho, uživatel musí být trpělivý.



Obrázek 4.2: Aplikace pro komunikaci a nastavení displeje Channels záložka

0	1	2
3	4	5
6	7	8

Obrázek 4.3: Ukázka rozložení kanálů na Univerzálním obrazovce displeje

5 Závěr

Cílem této diplomové práce bylo vytvořit aplikační software pro jednotku displeje elektromotokáry. Software má za úkol přehledně zobrazovat hodnoty a stavy na obrazovce, umožnit nastavování parametrů vozidla, komunikovat na sběrnici CAN, pomocí XBee modulu a USB a vyčítat informace ze senzorů pozice plynového a brzdového pedálu a data z akcelerometru, teplotního senzoru a GPS přijímače. Software je učen pro hardware navržený Bc. Martinem Sovou v jeho diplomové práci [1]

Aplikační software postupně inicializuje všechny potřebné periferie mikrokontroléru STM32F746IGT. Po inicializaci periferií mikrokontroléru následuje inicializace akcelerometru a teplotního snímače. Poté se spustí převod A/D převodníku a následuje vyčtení kalibračních dat z externí flash paměti jednotky displeje a nastavení filtrů pro příjem zpráv na sběrnici CAN.

Nastavení zobrazení hodnot je řízeno pomocí kanálů. Kanály obsahují informace o zdroji dat a o dalších parametrech, potřebných k zobrazení hodnoty na displej. Zdrojem dat pro kanál může být sběrnice CAN nebo hodnota některého snímače jednotky displeje. V případě, že je zdrojem CAN sběrnice kanál obsahuje CAN ID zprávy, ve kterém je hodnota vysílána, délku dat a pozici prvního bit ve zprávě. Dalšími parametry kanálu jsou název, jednotky, ve kterých bude hodnota zobrazena, počet desetinných míst na které má být hodnota zobrazena a minimum a maximum, kterého může hodnota kanálu dosáhnout . Posledními parametry jsou parametry pro úpravu hodnoty. Jedná se o násobitel a offset. Ty slouží pro převod hodnoty přijaté z CAN sběrnice na její fyzikální veličinu. Pozice kanálů na obrazovkách jsou pevné.

Pro nastavení těchto kanálu slouží PC aplikace, která hlídá zadání všech potřebných parametrů a jejich hodnot. Dále umožňuje uložení a načtení již vytvořených konfigurací a nahrání a smazání konfigurace z jednotky displeje. PC aplikace a jednotka displeje komunikují přes XBee modul, pomocí transportního protokolu popsaného v této práci. Aplikace pro obsluhu potřebných úkonů jako je mazání a flashování paměti využívá sady příkazů.

Software bohužel nepodporuje nastavování parametrů vozidla, komunikaci přes USB a dekódování zpráv z GPS modulu..

Literatura

[1] Sova Martin *Jednotka displeje pro elektromotokáru* Plzeň, 2016. Diplomová práce. Západočeská univerzita. Fakulta elektrotechnická. Katedra aplikované elektroniky a telekomunikací. Vedoucí práce: Ing. Elis Luděk, Dostupné z: `http://hdl.handle.net/11025/23014`

[2] Micron Serial NOR Flash Memory N25Q128A: Datasheet. *Micron* [online]. [cit. 15.5.2019]. Dostupné z: `https://www.micron.com/resource\OT1\textendashdetails/4c4af943\OT1\textendash8a6b\OT1\textendash4a29\OT1\textendashb124\OT1\textendash08b345151576`

[3] ISSI SYNCHRONOUS DYNAMIC RAM 64–MBIT: Datasheet IS42S16400J *Integrated Silicon Solutions Inc.* [online]. [cit. 15.5.2019]. Dostupné z: `http://www.issi.com/WW/pdf/42\OT1\textendash45S16400J.pdf`

[4] STM32F75xxx and STM32F74xxx advanced ARM®–based 32–bit MCUs: Reference manual. *ST Microelectronics* [online]. [cit. 15.5.2019]. Dostupné z: `http://www2.st.com/resource/en/reference_manual/dm00124865.pdf`

[5] Display LCD–TFT 7": Datasheet MCT070M6W800480LML. *Midas* [online]. [cit. 15.5.2019]. Dostupné z: `http://www.midasdisplays.com/products/4\OT1\textendash10\OT1\textendash14.page`

[6] MEMS digital output motion sensor ultra low–power high full–scale 3–axes "nano" accelerometer: LIS331HH Datasheet. *ST Microelectronics* [online]. [cit.

15.5.2019]. Dostupné z: http://www2.st.com/content/ccc/resource/technical/document/datasheet/58/d2/32/92/1c/e3/43/fd/CD00250937.pdf/files/CD00250937.pdf/jcr:content/translations/en.CD00250937.pdf

[7] MLM75x Digital Temperature Sensor and Thermal Watchdog With Two–Wire Interface: Datasheet. *Texas Instruments* [online].[cit. 15.5.2019]. Dostupné z: http://www.ti.com/lit/ds/symlink/lm75b.pdf

[8] GPS Receiver A2235–H: User's Manual. *Maestro–wireless* [online]. [cit. 15.5.2019]. Dostupné z: http://www.maestro\OT1\textendashwireless.com/a2235\OT1\textendashh\OT1\textendashtechnical\OT1\textendashspecifications/

[9] XBee ® /XBee–PRO ® RF Modules: Product Manual. *Digi International Inc* [online]. [cit. 15.5.2019]. Dostupné z: https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee\OT1\textendashDatasheet.pdf

[10] MARTÍNEK, Jan. GPS a komunikační protokol NMEA – 3 (dekódování dat) In: *http://www.abclinuxu.cz* [online]. 10.10.2006 [cit. 15.5.2019]. Dostupné z: http://www.abclinuxu.cz/clanky/ruzne/gps–a–komunikacni –protokol–nmea–3–dekodovani–dat

[11] STM32CubeMX In: *https://www.st.com* [online]. [cit. 15.5.2019]. Dostupné z: https://www.st.com/en/development\OT1\textendashtools/stm32cubemx.html

[12] NOVIELLO, Carmine. stm32746g˙discovery˙qspi.c In: *https://github.com* [online]. 25.8.2016 [cit. 15.5.2019]. Dostupné z: https://github.com/cnoviello/stm32\OT1\textendashdiscof7/blob/master/stm32\OT1\textendashdiscof7\OT1\textendashlcddim/system/src/BSP/stm32746g_discovery_qspi.c

[13] UTFT Fonts In: *http://www.rinkydinkelectronics.com* [online]. [cit. 15.5.2019]. Dostupné z: `http://www.rinkydinkelectronics.com/r\OT1\textendashfonts.php`

A Přílohy

A.1 Jednotka displeje

Obrázek A.1: Jednotka displeje, Univerzální obrazovka