České vysoké učení technické v Praze
Fakulta elektrotechnická

DIPLOMOVÁ PRÁCE

Softwarově definovaný multifunkční laboratorní přístroj

Praha, 2018 **Autor: Bc. Jakub Dibelka**

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne _____ _____

podpis

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Dibelka** Jméno: **Jakub** Osobní číslo: **406262**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra měření**

Studijní program: **Kybernetika a robotika**

Studijní obor: **Senzory a přístrojová technika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Programově definovaný multifunkční laboratorní přístroj

Název diplomové práce anglicky:

Software Defined Multifuncional Laboratory Instrument

Pokyny pro vypracování:

Navrhněte a s využitím mikrořadičů řady STM32 realizujte multifunkční přístroj pro výukové laboratoře. Přístroj bude obsahovat funkce voltmetru, osciloskopu, impulsního generátoru a čítače. Veškeré funkce budou realizovány programově s využitím funkčních bloků a periferií mikrořadičů STM32. Ovládání přístroje a zobrazení výsledků bude prostřednictvím nadřazeného PC, s nímž mikrořadič bude komunikovat prostřednictvím rozhraní USB, případně bloku UART a můstku UART - USB. Vytvořte potřebné programové vybavení pro mikrořadiče i pro nadřazené PC.

Seznam doporučené literatury:

[1] RM0316 Reference Manual, STMicroelectronics, 2016; www.st.com
[2] DS10362 STM32F303 Data, STMicroelectronics, 2016
[3] Yiu, J.: Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors

Jméno a pracoviště vedoucí(ho) diplomové práce:

doc. Ing. Jan Fischer CSc., katedra měření FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **26.09.2017** Termín odevzdání diplomové práce: _____

Platnost zadání diplomové práce: **28.02.2019**

Podpis vedoucí(ho) práce | Podpis vedoucí(ho) ústavu/katedry | Podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové

7. 11. 2017
Datum převzetí zadání

Podpis studenta

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu diplomové práce, panu doc. Ing. Janu Fischerovi, CSc., za věnovaný čas, podnětné rady a připomínky při vedení mé práce. Rovněž bych chtěl poděkovat svým prarodičům a přítelkyni za jejich podporu během celého studia.

Abstrakt

Diplomová práce se zabývá návrhem a realizací multifunkčního laboratorního přístroje implementovaného na mikrořadičích řady STM32 s využitím vnitřních funkčních bloků a periferií. Součástí práce je dále návrh řídící multiplatformní aplikace pro osobní počítače představující nástroj pro řízení laboratorního přístroje a pro vizualizaci změřených dat s možností dalšího post processingu.

V práci jsou diskutovány různé přístupy realizací jednotlivých funkčních bloků, jejich výhody a nevýhody, ale také vybraná vývojová prostředí. V poslední části je proveden popis nejvhodnější metody realizující osciloskop, logický analyzátor, voltmetr, čítač a impulzní generátor.

Abstract

The aim of this diploma thesis is to design realisation of a multifunctional laboratory instrument implemented on STM32 series microcontrollers using internal functional blocks and peripherals. Part of the thesis is the software design of a multi-platform application for personal computers representing a tool for controlling a laboratory device and for visualisation of the measured data with the possibility of further post processing as well.

The thesis discusses different approaches to the implementation of individual functional blocks. Selected development environments, their pros and corn, are also described there. The end of the work contains the description of the most appropriate method of realisation of oscilloscope, logic analyser, voltmeter, counter and pulse generator.

Index terms

STM32, STM32F042, STM32F303, oscilloscope, voltmeter, logic analyser, impulse counter, generator, framework Qt, Ac6.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Vlivem neustálého rozvoje elektrotechniky se pomalu pojem "elektronika" dostává do povědomí nejen elektroinženýrů a "bastlířů", ale i laické veřejnosti.

Problémem při hromadných výukách na středních a vysokých školách je nedostatek pracovišť vybavenými měřícími zařízeními, osciloskopy a generátory signálu. Ačkoliv studenti řešící své laboratorní úlohy nevyžadují špičková zařízení, tak celková cena za všechny přístroje dosahuje vysokých částek.

Jinou alternativou by mohl být nástroj, který bychom připojili k osobnímu počítači studenta a nebylo by nezbytně nutné pořizovat drahé měřící zařízení či generátory.

Takový to nástroj by ocenil i "bastlíř", který by si při svých pokusech o oživení "chytré" krabičky mohl proměřovat jednotlivé sběrnice, rozhraní a výstupy ze senzorů.

Náplní diplomové práce je návrh multifunkčního laboratorního přístroje. Tento přístroj bude realizován na mikroprocesorech řady STM32 s využitím jejich vnitřních bloků. Konkrétně se jedná primárně o dvě řady mikrořadičů, a to STM32F042 a STM32F303RE. Řada STM32F0 byla zvolena z důvodu nízké ceny a vysoké dostupnosti, řada STM32F303 jako protějšek s vyšším výkonem a větší kapacitou RAM a Flash paměti. Nicméně, vyvíjené firmwarové vybavení procesoru je navrženo tak, aby bylo snadno rozšiřitelné nejen na procesory dané řady, ale i na řady ostatní. Realizované funkce zastávají měření a generování signálu. Pro měření slouží osciloskop, logický analyzátor, impulzní čítač a pro generování signálu impulzní generátor.

Dalším cílem je implementace ovládacího multiplatformního softwaru umístěného na osobním počítači, který by nejen ovládal laboratorní přístroj, ale i zobrazoval, zpracovával, upravoval a exportoval data.

Kapitola 2

Analýza tématu

Začátek práce je zaměřen na základní seznámení čtenáře s mikrořadiči řady STM32, jejich vnitřní strukturou a uspořádáním.

Následuje podrobný popis koncepce laboratorního přístroje a funkčních bloků, které jsou nezbytně nutné pro správnou funkci osciloskopu, logického analyzátoru, voltmetru, čítače a impulzního generátoru. Dále jsou zde zmíněna požadovaná komunikační rozhraní umožňující komunikaci mezi osobním počítačem a mikroprocesorem. Okrajově jsou uvedeny základní rozdíly mezi jednotlivými řadami a čím se primárně vyznačují.

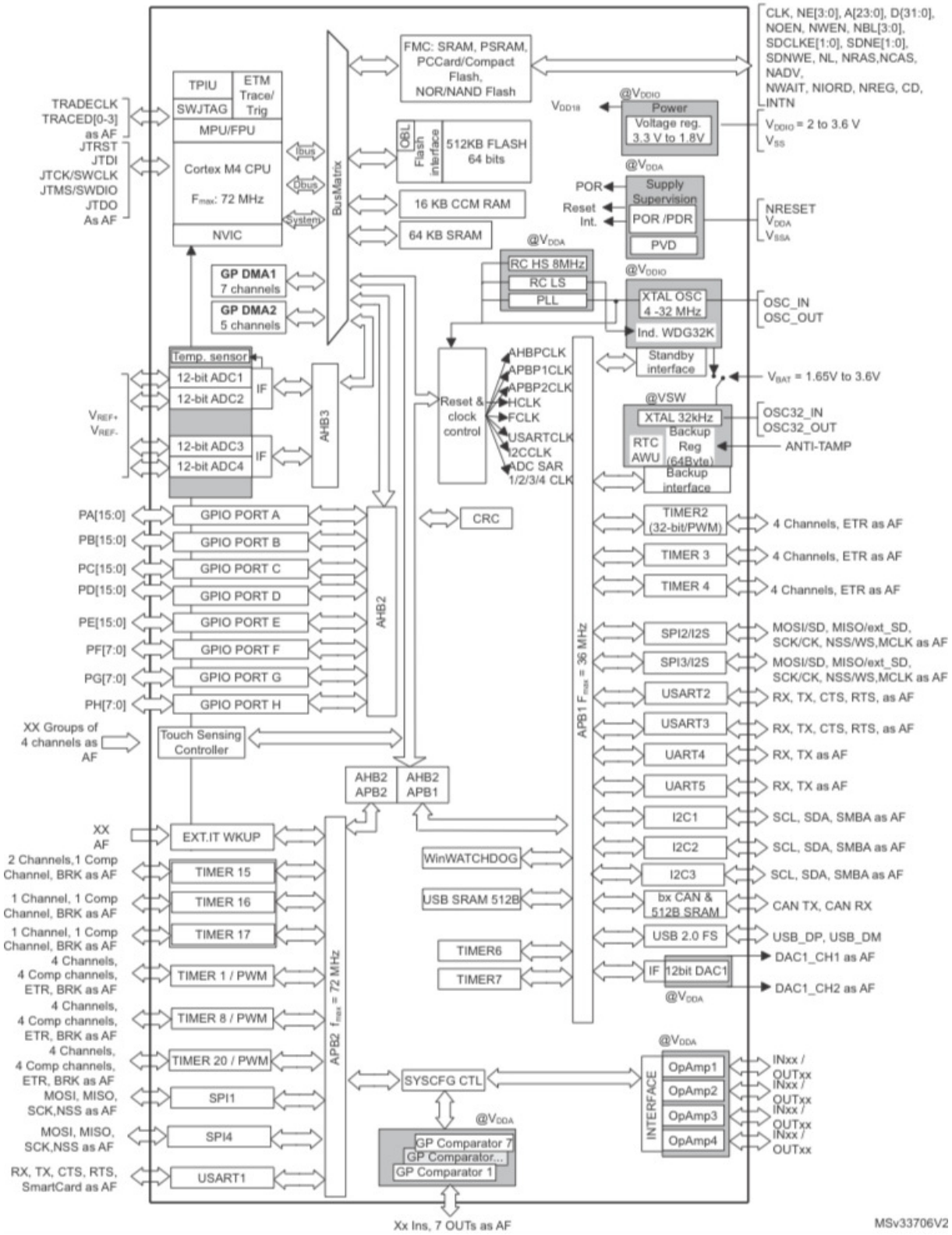
Závěr této kapitoly je věnován rozboru volby vývojových prostředí a metodice výběru knihoven pro mikrořadič, tak aby byla dosažena možnost jednoduché rozšiřitelnosti vyvíjeného firmwaru a softwaru.

2.1 Mikroprocesory řady STM32

Jedná se o 32-bitové mikroprocesory od firmy STMicroelectronics s architekturou jádra ARM Cortex M0/M0+, M3, M4 a M7.

2.1.1 Struktura mikroprocesoru

Mikrořadič se skládá z několika částí, a to sice z jádra, debugovacího systému, vnitřní sběrnice propojující periferie a paměti s jádrem. Dále také ze vstupně - výstupních obvodů, různých časovačů čítačů, periferií, bran, PLL obvodu a oscilátorů. Blokové schéma mikroprocesoru STM32F303xE [6] je uvedeno na obrázku 2.1.



Obrázek 2.1: Blokové schéma vnitřní struktury procesoru STM32F303xE, převzato z [5]

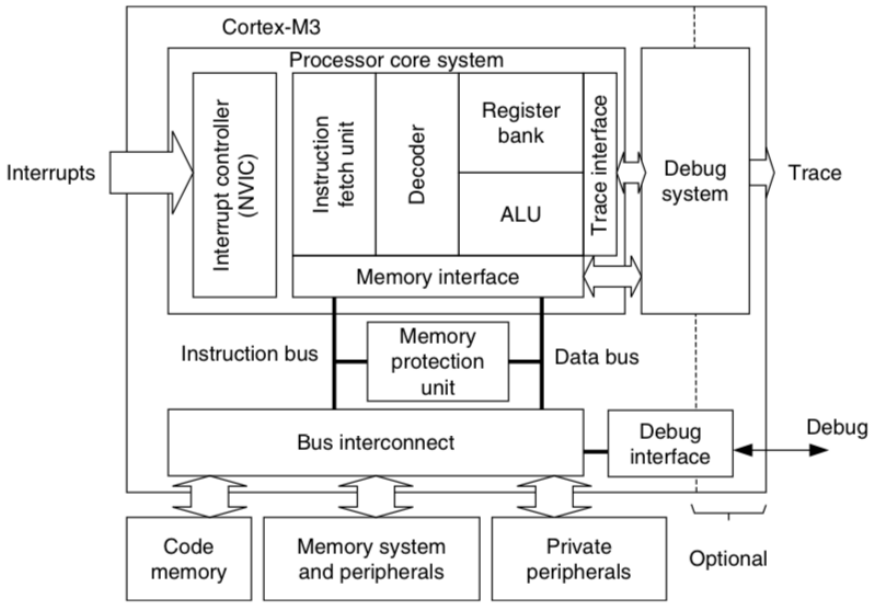
2.1.2 Jádro mikroprocesoru Cortex M3

Jádro 2.2 obsahuje vlastní výpočetní jednotku a NVIC, neboli jednotku přerušení. NVIC je schopná vyvolávat přerušení podle dané priority. To znamená, že přerušení s vyšší prioritou má přednost na obsloužení než přerušení s prioritou nižší.

Dále obsahuje debugovací obvody. Ty slouží k tomu, aby bylo možné nahrát program, popřípadě program krokovat, číst realtimově obsahy registrů a proměnných. K nahrání může být použito rozhraní SWD - serial wire debug nebo také postarší JTAG.

Instrukční sada jádra podporuje, jak ARM 32 - bitové instrukce, tak i instrukce 16 - bitové zvané Thumb a kombinaci 16 a 32 - bitu ve formě Thumb 2 sady.

Jádro je propojeno s dalšími částmi procesoru přes tři typy sběrnic. Jedná se o instrukční, datovou a systémovou sběrnici.



Obrázek 2.2: Blokové schéma jádra a jeho propojení s okolím, převzato z [10]

2.1.3 Interní periferie

Obsah periferií se v mikroprocesorech liší. Některý mikroprocesor disponuje periferií navíc, jiný nikoliv. Společné znaky, které se vyskytují ve většině typech STM32 jsou tyto periferie:

- **Bus matrix**: Sběrnice propojující jádro procesoru se zbytkem procesoru. Bus matrix deleguje datové a hodinové signály přes různé můstky do ostatních sběrnicových větví.

- **Zdroje hodinového signálu**: Zdroje signálu mohou být interní nebo externí. Interní zdroje tvoří většinou RC oscilátor. Externí zdroje, tvořené především krystalovým oscilátorem, se připojují skrze vstupy, výstupy procesoru. Takovým to příkladem může být dvojice zdrojů hodinového signálu, a to sice LSI x LSE a HSI x HSE.

  LS neboli low speed hodiny jsou používané realtimové úlohy. Písmeno I x E značí, zdali jsou interní nebo externí. HS high speed hodiny jsou určené k odvození systémového kmitočtu, na kterém poběží procesor. Důvod připojení externích zdrojů signálu spočívá v přesnosti. RC oscilátor vytvořený při výrobě procesoru není zdaleka tak přesný jako krystalový oscilátor. Příkladem může být pokus o použití HSI pro USB rozhraní, které potřebuje ke své činnosti 48 MHz. Kvůli nepřesnosti (jednotky procent) nedojde k procesu enumerace a USB rozhraní nebude fungovat.

  Problém vyřešíme připojením externího HSE krystalu. Možným řešením je u procesorů řady STM32x0 [8] použití vnitřního krystalu HSI48 spolu s dolaďovací funkcí. Tato metoda dolaďuje frekvenci v závislosti na připojeném USB rozhraní.

- **Obvody řízení frekvence hodinového signálu**: Obvody odvozují systémový kmitočet ze základního. Je možné připojit systémové hodiny přímo na různé druhy krystalů, nebo také využít obvody fázového závěsu, neboli PLL obvody. Ty slouží k vytvoření potřebného kmitočtu pomocí násobiček a před-děliček.

  Nelze ovšem nastavovat frekvenci libovolně. Existují limity omezující maximální frekvenci.

  Obecně platí, čím vyšší frekvence, tím vyšší výkon a tím i spotřeba. Pokud požadujeme nízko energetické úlohy, je nutné snížit frekvenci na minimum.

- **DMA jednotka**: Jedná se o jednotku, která umí přenášet data bez přímé obsluhy procesoru. Procesor se tak může věnovat jiným věcem. Ukázkou může být kopírování části pole do jiného paměťového prostoru. První scénář - kopírujeme-li pole pomocí for cyklu, procesor musí při každé instrukci asistovat. V danou chvíli je tímto úkolem procesor zaneprázdněn. Druhý scénář - použijeme-li řadič DMA, procesor se o kopírování nemusí starat a může se zabývat dalšími úkoly.

  Existují tři typy přenosů, a to z paměti do paměti, z paměti do periferie a z periferie do paměti.

- **Komunikační rozhraní**: Můžeme zde objevit rozhraní jako UART, SPI, I^2C, CAN. V pokročilejších typech se nacházejí Ethernet a USB typu Full speed nebo High speed.

- **Paměti**: Vyskytují se zde převážně dva typy pamětí, a to RAM paměť a paměť typu Flash. RAM paměť slouží k přímému zapisování a čtení. Její velikost se pohybuje od 8kB až po stovky kB. Po vypnutí napájení ztrácí svůj obsah.

  Paměť typu Flash slouží k uložení vlastního kódu mikroprocesoru a také k okamžitému uložení vypočítaných konstant. Oproti RAM paměti je pomalá, avšak po vypnutí napájení neztrácí svůj obsah. Velikosti se pohybují od 32kB až po jednotky MB.

- **Obvody dohledu**: Neboli také Watchdog. Tyto obvody se používají ke zjištění, zda nedošlo k chybě v kódu nebo vlivem externí události k zastavení správné činnosti procesoru.

  Princip funkce je jednoduchý. Musí se nastavit hodnota čítače obvodu dohledu. Tato hodnota je postupně odečítána. Pokud hodnota v čítači je rovna nule, dochází k resetu mikroprocesoru. To, zdali byl procesor restartován vlivem obvodu dohledu, lze vyčíst z registru. Ideální místo, ve kterém dochází k aktualizaci hodnoty čítače je v hlavní smyčce programu. Většinou toto místo je v main smyčce.

  Tato aplikace je vhodná k použití v nebezpečných aplikací, kde selhání mikroprocesoru může mít nedozírné následky.

- **ADC**: Analogově digitální převodník umožňuje převádět analogový signál na digitální. K tomu slouží převodník napětí s postupnou aproximací. ADC jednotka obsahuje několik vstupních kanálu, které jsou multiplexovány. K těmto kanálům jsou připojeny různé obvody uvnitř procesoru. Tyto obvody mohou být: měření napětí na referenčním regulátoru, popřípadě vstup od teplotního čidla. V případě přítomnosti více ADC jednotek je možné jednotky spolu spojovat a zvyšovat tak vzorkovací frekvenci.

  Jako kontrola zde může působit analog Watchdog pracující na principu hlídaní, zdali změřené napětí je v daném tolerančním pásmu či nikoliv a na to reagovat. Příklad aplikace je měření napětí na baterii.

- **DAC**: Obvod vyskytující se ve výkonnějších verzích. Digitálně analogový převodník slouží k převádění digitálního signálu na signál analogový. Používá se především v obvodech generující signály typu sinus a jim podobných.

- **GPIO brány**: Brány mají za úkol propojit mikroprocesor s okolím. Jejich konfigurací můžeme nastavit, zdali brána je nakonfigurovaná jako vstup či výstup. Také je možné propojit pin s danou periferií nebo nastavit logickou úroveň (logická 0, 1). Samozřejmostí je nastavení rychlosti daného pinu. Toto nastavení pomáhá lépe přizpůsobovat doby náběžných a sestupných hran při komunikaci.

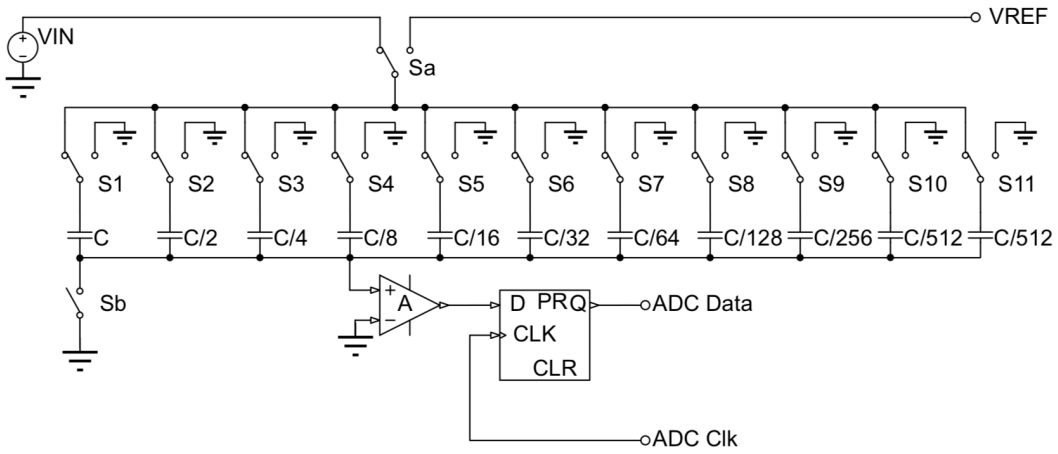
- **Interní regulátor napětí**: Napájecí napětí mikroprocesoru je ve většině případů rovno 3.3 V. Napájecí napětí jádra může být daleko menší. Tato hodnota se pohybuje od 1.2 V do 1.8 V v závislosti na řadě.

- **Časovače/čítače**: Používají se především na periodické generování událostí. Konfigurací časovačů/čítačů, dále jen "č/č", v režimu master - slave můžeme jedním synchronně spouštět další č/č,. Různými konfiguracemi můžeme generovat nebo měřit signál. Ve většině mikroprocesorech je několik 16-bitových č/č, ale vyskytuje se zde obvykle jeden 32-bitový č/č.

- **Pokročilé funkce**: Pokročilé funkce a obvody přináší podporu ovládání, např. LTDC displeje. U kterého je pomocí paralelního rozhraní dosaženo vysoké vykreslovací frekvence na rozdíl od použití sběrnice SPI. Dalším příkladem je přímá podpora přenášení dat z kamerky a podpora ovládání kapacitních tlačítek. Jedná se o akce, které by softwarově trvaly dlouho a implementačně by nebyly zcela primitivní.

2.1.4 ADC převodník

ADC jednotky umístěné v STM32 mikroprocesorech slouží k převodu analogového signálu na digitální. Obsahují převodník s postupnou aproximací. Převodník pracuje na jednoduchém principu. Nabijeme snímací kondenzátor, měříme potenciál na kondenzátoru a pomocí registru s postupnou aproximací 2.3 je vypočtena digitální hodnota signálu.

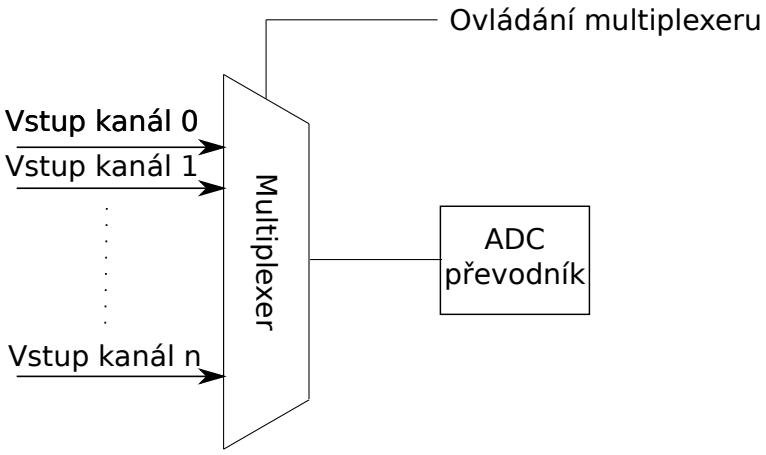


Obrázek 2.3: Struktura registru s postupnou aproximací, převzato z [4]

Klíčovým parametrem pro ADC jednotku je počet cyklů na odběr vzorku, kde se se známou frekvencí lze dopočítat hodnoty délky doby odběru vzorku. Délka doby odběru vzorku je důležitá pro děje, které nabíjí snímací kondenzátor pomalu. Dále je podstatný počet cyklů na převod analogového signálu na signál digitální. Většina STM32 mikroprocesorů má ADC jednotku vybavenou 12 - bitovým převodníkem, který dokáže převést hodnotu za 12,5 cyklu. Pokud programátor uzná, že je zbytečné převádět 12 - bitový signál, postačí jej převést pouze na 10, 8, 6 - bitový signál. Čím méně bitový je výsledný signál, tím je doba převodu kratší a lze dosáhnout větší maximální vzorkovací frekvence.

Různé řady STM32 vlastní jednu nebo více ADC jednotek. V případě systému s více jednotkami lze provést různé operace, kdy jedna ADC jednotka pomáhá druhé. Tím dojde ke zmenšení délky doby snímaní či výpočtu převedené hodnoty a tím se opět zvýší vzorkovací frekvence.

Každá ADC jednotka obsahuje až 19 kanálů, které jsou řízeny multiplexovaně 2.4. K dané jednotce jsou často připojeny i interní obvody, které měří napětí na referenčním regulátoru a vstup od teplotního čidla.



Obrázek 2.4: Multiplexování ADC převodníku

Vstupy do ADC mohou být:

- **Single - ended**: Měříme jeden vstup.
- **Diferenciální**: Měříme rozdíl mezi dvěma vstupy.

Základní operační konverze:

- **Jedna konverze**: ADC se spustí pouze jednou.
- **Kontinuální konverze**: ADC se spouští neustále po ukončení převodu.

2.1.4.1 Systémy s jedním ADC převodníkem

Pro výpočet délky doby odběru vzorku jednoho kanálu platí rovnice 2.1 . T_{sam} je délka doby odběru vzorku, n_{cycle} je počet cyklů odběru vzorku, f_{ADC} je frekvence ADC jednotky.

$$T_{sam} = \frac{n_{cycle}}{f_{ADC}} \tag{2.1}$$

Maximální vzorkovací frekvence $f_{MAXsamp}$, které lze dosáhnout na daný počet cyklu je dána jako 2.2 , kde n_{conv} je počet cyklu konverze napětí.

$$f_{MAXsamp} = \frac{f_{ADC}}{n_{cycle} + n_{conv}} \tag{2.2}$$

Maximální vzorkovací frekvence, které lze dosáhnout na daný počet cyklů n je dán rovnicí 2.3 .

$$f_{MAXsamp} = \frac{f_{ADC}}{n(n_{cycle} + n_{conv})} \tag{2.3}$$

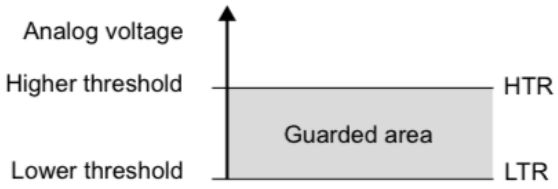
2.1.4.2 Systémy se dvěma a více ADC převodníky

Pokud je přítomno více ADC jednotek v daném řadiči, je výhodné rozdělit ADC vstupy tak, aby jednotlivá ADC jednotka měla co nejméně multiplexovaných vstupů. Tímto přístupem lze zvýšit maximální vzorkovací frekvenci.

Další možností, jak zvýšit maximální vzorkovací frekvenci, je propojení několika ADC jednotek v tzv. interleaved módu. Tento mód propojí ADC jednotky tak, že jedna jednotka zpracuje část ADC převodu a konverze. Druhá jednotka zpracuje druhou část. Tímto přístupem se můžeme dostat na dvojnásobek maximální vzorkovací frekvence.

2.1.4.3 Analog Watchdog

Analog Watchdog je obvod, který je přímo připojen na ADC jednotku. Ten se v případě potřeby stará, jestli je snímané napětí v rozsahu daném minimální (Lower threshold) a maximální hranicí (Higher threshold). V opačném případě tuto skutečnost oznámí v přerušení. Většinou počet jednotek analog Watchdogů je stejný jako počet ADC jednotek. Ačkoliv se taková funkce může zdát bezvýznamná, může být za jistých podmínek užita jako trigger k osciloskopu.



Obrázek 2.5: Znázornění hlídané oblasti analog Watchdogem, převzato z [9]

2.1.5 Časovače/čítače

V aplikačních poznámkách se označují jako $TIMx$. Mohou pracovat v režimu časovače nebo čítače. Velikost čítacího registru je obvykle 16 - bitová, popřípadě se vyskytuje "č/č" s velikostí 32 - bitů. Obvykle počet vstupů a výstupů je roven čtyřem.

Jednotlivé typy "časovačů/čítačů" se liší velikostí čítacího registru. Dále se odlišují funkcemi, kterými disponují. Jednotlivé typy můžeme rozdělit na základní, obecně - účelové a pokročilé.

2.1.5.1 Základní

Charakteristické funkce:

- **16 - bitové**
- **Inkrementace s možností znovuobnovení inicializačního počtu při přetečení**

- **Input capture**
- **Output capture**
- **Generovaní PWM signálu**
- **Podpora přerušení a DMA při různých událostech**

2.1.5.2 Obecně účelové

Charakteristické funkce:

- **16 i 32 - bitové**
- **Inkrementace, dekrementace registru s možností znovuobnovení inicializačního počtu při přetečení**
- **Input capture**
- **Output capture**
- **Generovaní PWM signálu**
- **Synchronizační nástroje s ostatními časovači**
- **One - pulse mód**
- **Podpora přerušení a DMA při různých událostech**
- **Podpora kvadraturního enkodéru**
- **A další**

2.1.5.3 Pokročilé

Charakteristické funkce:

- **16 - bitové**
- **Inkrementace, dekrementace registru s možností znovuobnovení inicializačního počtu při přetečení**
- **Input capture**
- **Output capture**
- **Generovaní PWM signálu**
- **Synchronizační nástroje s ostatními časovači**
- **One - pulse mód**
- **Komplementární výstup**

- **Podpora přerušení a DMA při různých událostech**
- **Podpora kvadraturního enkodéru**
- **A další**

2.1.6 Jednotka s přímým přístupem do paměti DMA

Jednotka s přímým přístupem do paměti umožňuje přenášet data bez činnosti mikroprocesoru.

Existují tři směry přenášení dat.

- **Z paměti do paměti**: Není možno pomocí události řídit přenos. Jedná se o pouhé překopírování.
- **Z periferie do paměti**: Je možno pomocí události řídit přenos. Například pomocí timeru událost "update". Tak můžeme řídit jistým způsobem vzorkování např. hodnoty ADC převodníku nebo hodnotu registru čítače.
- **Z paměti do periferie**: Pracuje jako přenos z periferie do paměti. Typickým užitím je modulace světla led diody při jejím plynulém rozsvícení nebo generování sinusového signálu pomocí PWM modulace a externího filtru.

DMA pracuje ve dvou režimech:

- **Kruhový režim**: Před začátkem přenosu je nastavena velikost přenášených dat (velikost Word - 4 byty, Halfword - 2 byty, Byte - jeden byte) a také jejich počet. Postupem přenosu se počet dekrementuje do nuly. Pokud se počet nepřenesených dat rovná nule, DMA jednotka automaticky nastaví počet na hodnotu, která byla nastavena před začátkem přenosu. Ta se následně spustí. Tento proces se neustále opakuje.
- **Normální režim**: Proces je podobný jako u kruhového režimu s tím rozdílem, že když je počet neodeslaných dat rovný nule, přenos skončí. Přenos obnoví pouze program, který opětovně nastaví počet přenášených dat.

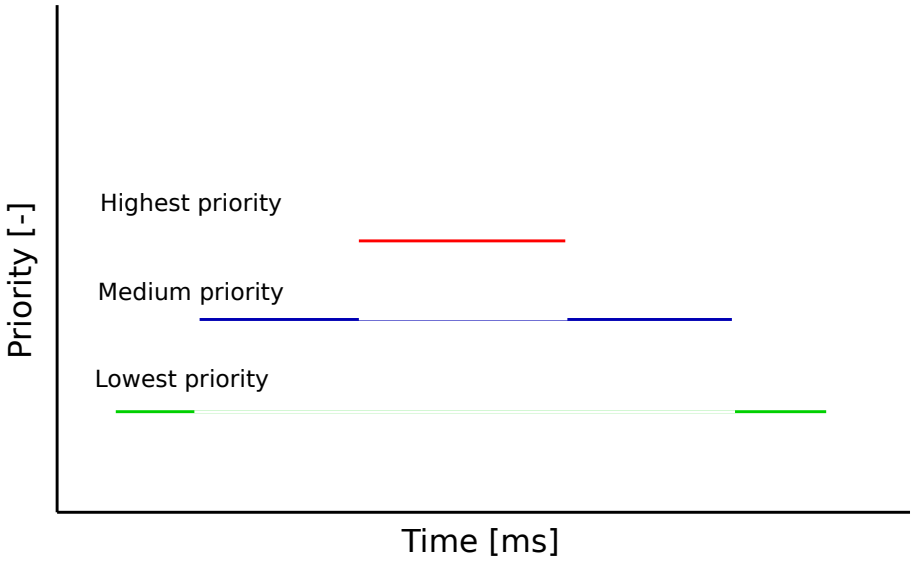
2.1.7 Jednotka přerušení NVIC

NVIC je jednotka, která má schopnost vyvolávat přerušení. Přerušení se řídí podle priorit. Existuje několik stupňů priorit. Událost s větší prioritou je obsloužena nejdříve.

Nejjednodušší příklad použití přerušení je, když budeme chtít v okamžiku stisknutí tlačítka rozsvítit led diodu. Případ bez přerušení by se řešil tak, že by se vyčítal stav registru dané brány a tím by se procesor neustále zaneprazdňoval. Pokud bychom použili přerušení EXTI na daný pin, na náběžnou či sestupnou hranu, procesor by mohl pracovat na jiných činnostech a následně při vzestupné hraně v přerušení by skočil do dané rutiny a rozsvítil led diodu.

Přerušení se nechá použít nejen na piny, ale také i na periferie. Může oznámit, že ADC převodník úspěšně překonvertoval analogovou hodnotu napětí na digitální. Dobrým příkladem je také to, že USART jednotka pracující v DMA režimu odeslala všechna data.

Následná ilustrace ukazuje, jak funguje přerušení, které má vyšší prioritu vůči přerušení s nižší prioritou 2.6.



Obrázek 2.6: Vykonávání úloh v přerušení podle priorit

2.1.8 Komunikační rozhraní USART

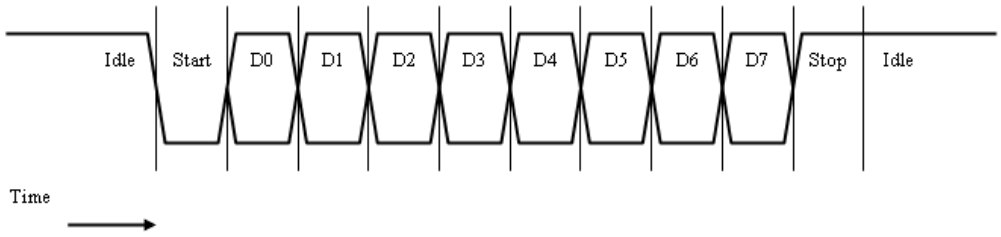
Komunikační rozhraní USART je sériové rozhraní, které se ztotožňuje se známou RS - 232. V minimální konfiguraci obsahuje vysílací a přijímací vodič.

Z hlediska STM32 mikroprocesorů je možné nastavit, jak má být použito hardwarové řízení toku dat, počet datových bitů, směry komunikace, počet stop bitů, paritní bity a rychlost.

Datový paket se skládá ze start bitu - úroveň logická 0, 7 až 9 datových bitů, které jsou nastavitelné. Datové bity jsou řazené od LSB, tedy od nejméně významného bitu po nejvyšší. Následuje stop bit na úrovni logické 1.

Obrázek níže ilustruje datový paket 2.7.

Rychlost přenosu dat je definována počtem Baudu. Baud je modulační rychlost. Baud můžeme připodobnit k jednomu vyslanému bitu za sekundu. Vezmeme-li si modulační rychlost 115200 Baudů, zjistíme, že přenosová rychlost je 115200 bit/s. Bohužel pravá datová přenosová rychlost není 115200 bit/s, ale je menší. Musíme odebrat start, parity, stop bit a počítat pouze s 8 datovými bity. Výsledná datová rychlost je 92160 bit/s. Prakticky platí, čím větší velikost Baud rychlosti, tím je potřeba přesnějšího zdroje hodin pro USART jednotku, jinak by docházelo ke ztrátě dat.



Obrázek 2.7: Znázornění komunikace po sběrnici USART, převzato z [1]

2.1.9 Komunikační rozhraní USB

Komunikační rozhraní USB je univerzální sériová sběrnice. Dnes existuje velké množství verzí od zastaralého USB 1.0. po nejnovější verze USB 3.x. Používá se nejen pro připojení myši, klávesnic, tiskáren, ale nejnovější standarty dokáží dané zařízení už i nabíjet a přenášet obrovské toky dat.

Typy USB a jejich rychlosti:

- **Low speed**: 1,5 Mbit/s
- **Full speed**: 12 Mbit/s
- **High speed**: 480 Mbit/s
- **Super speed**: 5 Gbit/s

V topologii USB stojí jeden hostitel, většinou osobní počítač, který řídí veškerou komunikaci po celé sběrnici. Připojených zařízení může být až 127. Osobní počítač obsahuje ovladače a pomocí procesu enumerace rozpozná, o jaký typ zařízení se jedná a jak s ním má nakládat.

Komunikace se zařízením probíhá přes roury. Počet rour je daný typem USB. Vždy existuje nultá roura, přes kterou se provádí proces enumerace a systém dotazovaní, nastavování a odpovědí.

Existuje několik typů přenosů:

- **Řídící přenos**: Slouží pro enumeraci USB zařízení. V případě chyby se vysílání opakuje.
- **Bulkový přenos**: Blokový přenos pro velké množství dat, ale také pro paměťová zařízení a tiskárny. V případě chyby se vysílání opakuje.
- **Isochroní přenos**: V případě chyby se vysílání neopakuje, jedná se o ztrátové vysílání. Cílem je streamovat data v reálnem čase, např. obraz, zvuk, atp.
- **Přerušovací přenos**: Jedná se hlavně o přenos, který danou frekvencí posílá maloobjemová data hostiteli. Tím může být myš, která každých 10 ms zašle příkaz k pohnutí kurzoru. V případě chyby se vysílání opakuje.

Osobní počítač obsahuje ovladače pro jisté třídy zařízení. Těmito ovladači je definováno chování zařízení a to, k čemu slouží. Takové třídy mohou být:

- **HID**: Zařízení pro uživatele s osobním počítačem, myš, klávesnice.
- **Mass Storage**: Paměťová úložiště.
- **Printer**: Tiskárny.
- **CDC**: Komunikační třída, použití pro Virtual Com Port.

V uvažovaných STM32 mikroprocesorech je použito USB typu Full speed, bulkový přenos a CDC třída.

2.1.10 Rozdíly mezi jednotlivými řadami

Jednotlivé řady mikroprocesorů se od sebe odlišují. Primární okruhy aplikací jsou dvojí. Na jedné straně stojí výkon, na straně druhé stojí spotřeba. Platí, že čím vyšší frekvence procesoru, tím provedeme více výpočtů za sekundu. Pokud se podíváme na rovnici 2.4, zjistíme, jakým způsobem je závislý odběr proudu na frekvenci a dalších parametrech. U_{CC} je napájecí napětí, C_L je přebíjená kapacita. Z toho vyplývá, že spotřeba je závislá na frekvenci.

$$I = fU_{CC}C_L \tag{2.4}$$

Firma STM nabízí tyto řady: F0, F1, F2, F3, F4, F7, H7 a řady L0, L1, L4.

Nízko energetické řady jsou L0, L1, L4. Tyto řady disponují nástroji pro snížení spotřeby. Příkladem může být ADC převodník, který má možnost se připojit na vlastní zdroj hodinového signálu z HSI14 - RC oscilátory o frekvenci 14 MHz, přičemž frekvence jádra může být daleko menší.

Pokud bychom se zaměřili na výkon, jedná se o řady F0, F1, F3, F2, F4, F7, H7, kde jednotlivé řady jsou seřazeny podle výkonu, a to od nejmenšího po největší.

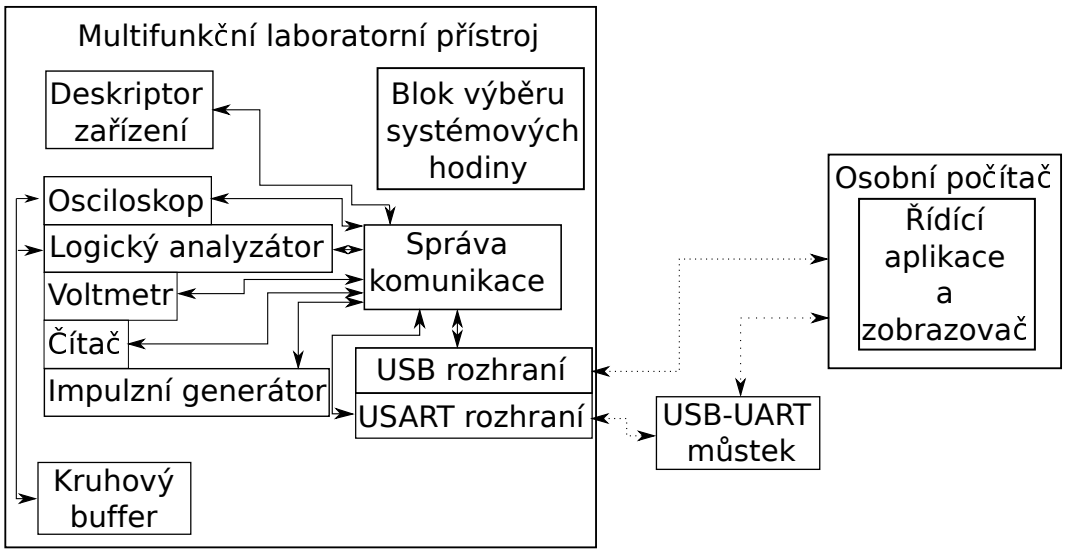
Dalším rozdílem je struktura PLL obvodu a jeho výstupů. Výběr zdrojů hodinového signálu je podstatný jednak pro jednotlivé sběrnice, ale také pro periferie či rozhraní. Velmi často se liší počtem přítomných ADC a DAC převodníků, časovačů/čítačů a jejich rozlišením, DMA jednotek, UART, SPI, I^2C, CAN, Ethernet rozhraní, počet GPIO bran a zdrojem hodinového signálu pro systémovou frekvenci.

2.2 Koncepce multifunkčního laboratorního přístroje

Multifunkční laboratorní přístroj je složen ze dvou částí. První část je implementována na mikrořadiči, druhá část na osobním počítači. Mikroprocesor se skládá z funkčních

bloků přístroje, kruhového bufferu, bloku výběru systémových hodin, správy komunikace navazující na zvolené rozhraní (USB nebo USART). Na osobním počítači je umístěna řídící aplikace.

- **Funkční bloky**: Slouží k realizaci osciloskopu, logického analyzátoru, voltmetru, čítače a impulzního generátoru.
- **Deskriptor zařízení**: Obsahuje unikátní popis zařízení.
- **Správa komunikace**: Stará se o oboustrannou komunikaci mezi mikroprocesorem a řídící aplikací přes preferované rozhraní.
- **Kruhový buffer**: Slouží k ukládání dat pro paměťově náročné funkční bloky.
- **Blok výběru systémových hodin**: Vybírá preferovaný zdroj systémového hodinového signálu.
- **Řídící aplikace a zobrazovač**: Představuje nástroj pro ovládání multifunkčního laboratorního přístroje.



Obrázek 2.8: Blokové schéma koncepce laboratorního multifunkčního přístroje

2.3 Funkční bloky přístroje

Laboratorní přístroj disponuje následujícími funkcemi: ke sledování signálu slouží osciloskop, pro sledování logických úrovní na sběrnici logický analyzátor, pro přesné měření napětí - voltmetr, k měření střídy a frekvence čítač, ke generování signálu s proměnnou střídou a frekvencí - impulzní generátor.

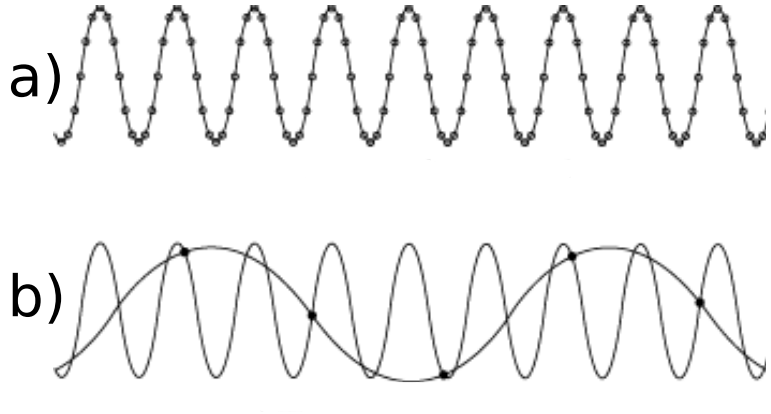
2.3.1 Osciloskop

Osciloskop je zařízení, které umožňuje převádět měřený napěťový signál na signál digitální s následnou vizualizací. Postupným vzorkováním v čase složíme výsledný signál 2.9 za a). Nejen u osciloskopu je důležitý vzorkovací teorém, který říká, jaký vztah platí mezi vzorkovací frekvencí a maximální frekvencí měřeného signálu.

Pro vzorkovací teorém platí rovnice 2.5. f_s je vzorkovací frekvence, f_{MAX} je maximální frekvence, kterou lze vzorkovat.

$$f_{MAX} < \frac{f_s}{2} \tag{2.5}$$

Pokud by byl porušen vzorkovací teorém, dojde k tomu, že rekonstruovaný změřený signál má chybnou výslednou frekvenci. Tomuto jevu se říká aliasing 2.9 za b).



Obrázek 2.9: Správně vzorkovaný signál a) a nevhodně vzorkovaný signál, kde dochází k aliasingu b), převzato z [3]

U osciloskopu rozlišujeme dvě osy, a to osu horizontální a osu vertikální. Osa horizontální označuje čas a osa vertikální uvádí hodnotu napětí. Prakticky klasické osciloskopy mají jeden nebo více vstupních kanálů. Tím je umožněno měřit více signálů najednou.

U osciloskopu rozeznáváme několik režimů spuštění osciloskopu:

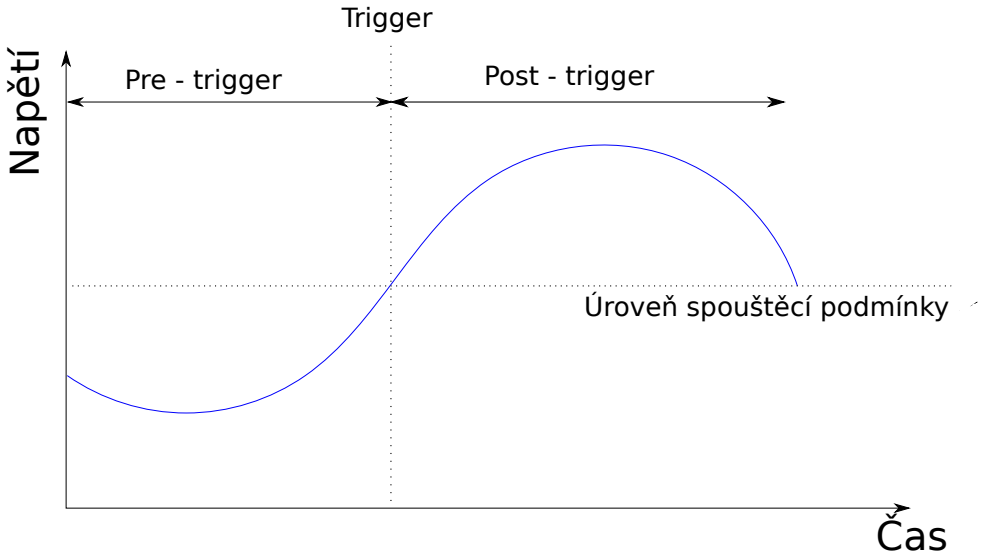
- **Automatický**: Osciloskop čeká jistou dobu a poté se spustí i bez dosažení triggrovací podmínky(úrovně spuštěcí podmínky). Je možné vidět aktuální průběhy signálu. Pokud je splněna triggrovací podmínka, zobrazí se jako v normálním režimu signál dle spouštěcí podmínky.

- **Normální**: Osciloskop zobrazuje průběh pouze tehdy, jestliže pozorovaný signál splňuje triggrovací podmínku. Pokud není splněna spouštěcí podmínka, osciloskop signál nezobrazí.

Spouštěcí podmínka neboli trigger. Obvykle trigger může reagovat na vzestupnou, sestupnou nebo vzestupnou či sestupnou hranu. Trigger rozeznává v časové části pre -

trigger a post - trigger 2.10. Pre - trigger je označení pro signál, který leží před místem triggeru. Post - trigger je oblast, která leží za místem triggeru. V napěťové ose rozeznáváme úroveň spouštěcí podmínky.

Většina osciloskopů disponuje režimem "RUN/STOP"a "SINGLE". Funkce "SINGLE"po vyhodnocení spouštěcí podmínky dokončí záznam a na obrazovce se objeví daný průběh. Funkce "RUN/STOP slouží k dokončení záznamu, okamžitému zastavení a zobrazení signálu nebo spuštění činnosti osciloskopu.



Obrázek 2.10: Znázornění triggeru, pre - triggeru, post - triggeru a úrovně spouštěcí podmínky

2.3.2 Logický analyzátor

Logický analyzátor je nástroj sloužící ke sledování logických úrovní. Na rozdíl od osciloskopu nezaznamenává napěťové úrovně, protože není potřeba převádět analogovou hodnotu na digitální, ale zaznamenává úrovně logické. Z toho vyplývá, že je schopen dosahovat větší vzorkovací frekvence.

Využívá se jako nástroj pro analýzu datového přenosu na sběrnicích. Platí zde opět vzorkovací teorém. Prakticky disponuje dlouhou pamětí, aby bylo možné analyzovat co nejdelší úsek komunikace. U logického analyzátoru se může vyskytovat možnost triggeru na vzestupnou, sestupnou, vzestupnou a sestupnou hranu.

2.3.3 Voltmetr

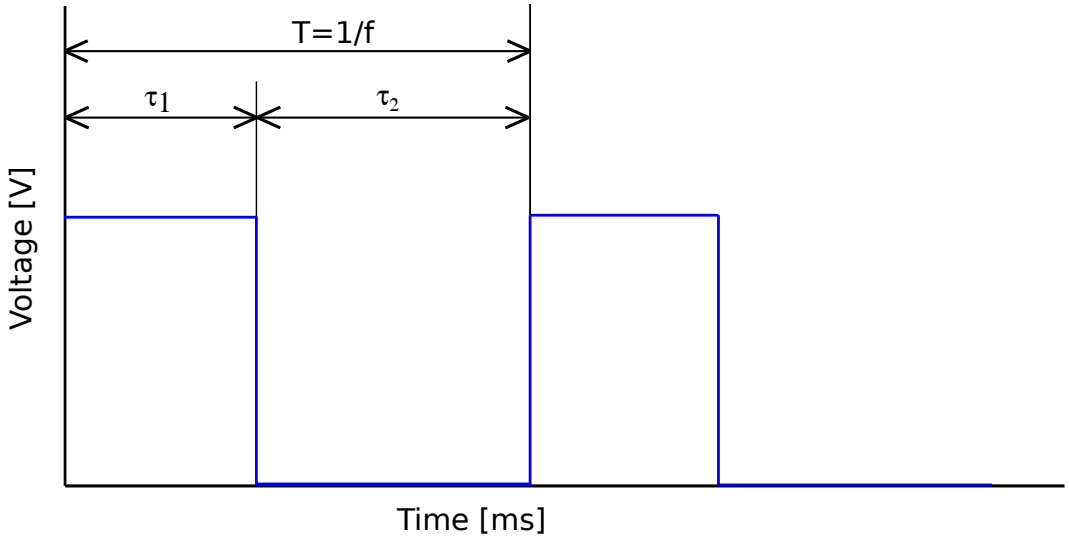
Slouží k měření elektrického napětí. Na rozdíl od osciloskopu, kde nás zajímá tvar signálu, jde v tomto případě o přesné změření napětí.

2.3.4 Impulzní generátor

Impulzní generátor generuje periodický obdélníkový signál s danou frekvencí a střídou. Střída 2.11 je definována jako poměr, kdy signál je v logické 0 a logické 1 za určenou periodu. Touto metodou můžeme generovat PWM signál s definovanou střídou.

Výpočet střídy v jednotkách procent je formulován jako rovnice 2.6.

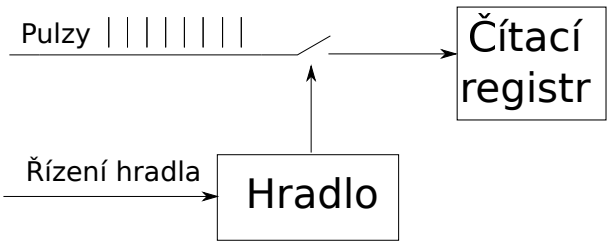
$$Duty = \frac{\tau_1}{T}100 \tag{2.6}$$



Obrázek 2.11: Vyznačení periody a střídy

2.3.5 Čítač

Čítač je zařízení, které je schopno čítat pulzy. Nejjednodušší čítač se skládá z hradla a čítacího registru 2.12. Po dobu otevření hradla jsou jednotlivé pulzy čítány čítacím registrem. Ze sečtených pulzů lze vypočítat periodu, frekvenci a střídu.



Obrázek 2.12: Princip jednoduchého čítače

2.4 Vývoj programového vybavení

Vývoj programového vybavení je rozdělen na dvě části. Pro mikroprocesor firmware a pro aplikace software. V dnešní době existuje velké množství vývojových nástrojů. Někteří výrobci ke svému portfoliu mikrořadičů přidávají i vývojové nástroje usnadňující do značné míry práci vývojářů při sestavování matice pinů, periferií, přerušení, atp. Přidávají i několik svých napsaných knihoven, které mají své výhody, ale i nevýhody a přináší různé oblasti aplikací.

2.4.1 Výběr vývojového nástroje pro mikrořadič

Jedná se o nástroje umožňující nahrát a debugovat program do různých mikroprocesorů přes různá rozhraní. Ohledně debugovaní se pyšní podobnými vlastnostmi přímé čtení proměnných, registrů, krokovaní, zastavení programu v místě označeným breakpoitem. Jejich cílem je zlepšit práci vývojáře. Liší se v zásadě cenou, licencí, počtem podporovaných mikroprocesorů a platformou, na které jdou spustit. K těmto účelům byla vybrána tři prostředí, která výše popsanými vlastnostmi disponují: IDE Keil, IAR, AC6.

2.4.1.1 IDE Keil

- **Výhody**: Podpora velkého množství procesorů.
- **Nevýhody**: Spustitelné pouze na platformě Windows, omezená licence, pro různé mikroprocesory jiné limitace (pro Arm 32kB).

2.4.1.2 IAR Embedded Workbench

- **Výhody**: Podpora velkého množství procesorů.
- **Nevýhody**: Spustitelné pouze na platformě Windows, omezená licence (30 dní nebo velikost kodu do 32kB).

2.4.1.3 AC6 (System Workbench for STM32)

- **Výhody**: Multiplatformní, neomezená velikost programu, zdarma, postavený na Eclipse IDE.
- **Nevýhody**: Podpora pouze STM32 procesorů.

Nejvhodnější vývojové prostředí je z důvodu výše zmíněných výhod System Workbench for STM32.

2.4.2 Knihovny

Firma STM poskytuje několik knihoven, které umožňují programátorovi vybrat si tu nejvhodnější.

- **Registrový přístup**: Tento přístup je nejvhodnější pro někoho, kdo chce pochopit, jakým způsobem pracuje zápis a čtení do registru. Tento přístup je také vhodný k optimalizaci kódu se zaměřením na jeho výslednou velikost nebo na jeho rychlost. Nevýhodou je nízká přenositelnost.
- **Standard Peripheral Library**: Standartní knihovna, která sloužila po mnoho let. Dnes ji již výrobce přestal podporovat. Inicializace periferií probíhala přes struktury. Přenositelnost byla do značné míry umožněna. Podporované mikroprocesorové řady jsou : L1, F0, F1, F2, F3, F4. Vzhledem k ukončení podpory není její další použití v rámci práce vhodné.
- **Hardware abstraction layer (HAL)**: Jak již název napovídá, jedná se knihovny s jistým stupněm abstrakce. Jsou velmi dobře přenositelné mezi jednotlivými řadami. Nevýhoda je velikost a nakonec i celková rychlost kódu, pro někoho se může zdát i zhoršená celková čitelnost.
- **Low Layer Api (LL)**: Nejnovější nízko úrovňová knihovna, která dovoluje psát rychlejší a menší kódy vůči HAL. Jistá nevýhoda je ta, že chybí vůči HAL předpřipravené drivery, např. pro USB periferii. Je tedy nutné napsat vlastní drivery nebo použít i část HAL knihoven. Stojí za zmínku, že drivery HAL a LL lze do značné míry kombinovat.

2.4.3 Vývojové prostředí STM32CUBEMX

STM32CUBEMX je vývojářský nástroj umožňující rychlou konfiguraci periferií a informace ohledně nabízených řad procesorů a jejich vlastností. Nástroj dokáže detekovat, jestli jednotlivé piny jsou zabrány a upozorní uživatele, že daná periferie je již použita a nelze ji pro danou operaci využít. Abychom se vyhnuli hledání alternativního pinu v datasheetu, program v momentu zvýrazní alternativní pin se stejnou funkcionalitou. Druhotným úkolem programu je generování předpřipraveného kódu pro začínající vývojáře ať už Standard Peripheral Library, HAL nebo LL knihoven.

2.4.4 Výběr vývojového nástroje pro nadřazené PC

V dnešní době se situace pro vývoj multiplatformních aplikací značně zlepšila. Neplatí již, že Java a .NET jsou pouze spustitelné na počítačích s operačním systémem Windows. Uvažované multiplatformní jazyky, které jdou přeložit na různých platformách jsou: framework Qt (C/C++), Java, .NET. Pokud bychom se zaměřili na efektivnost, rychlost kódu, podporu externích knihoven a zkušeností, zvolíme framework Qt.

2.4.4.1 Framework Qt

Qt framework byl vytvořen na přelomu století Trolltech. Tato společnosti prodala Qt společnosti Nokia. Jedná se o multiplatformní nástroj, který se zaměřuje nejen na PC platformy (Windows, MacOs, Linux), ale také na platformy mobilní (Android, iOS). Qt je

postaven na jazyku C/C++. Za jistých podmínek lze používat zcela zdarma. Momentálně Qt využívají nejen malé firmy, ale i ty s mezinárodní působností.

Qt podporuje velké množství nástrojů a přeprogramovaných modulů a knihoven. Mezi základní moduly patří komunikace s okolním světem (podpora Virtual Com Port, komunikace se serverem), podpora 2D, 3D grafiky, grafů, interakce mezi programem a kurzorem myši, nástroje pro editování grafického rozhraní a podpora různých vzhledů vzhledem k použité platformě.

Kapitola 3

Návrhy řešení funkčních bloků

Tato kapitola se zprvu věnuje popisu komunikačního protokolu. Jeho účelem je definovat formu přenášených dat mezi osobním počítačem a mikroprocesorem. Zaměříme se na jeho obecnou definici a popíšeme si příkazy pro jednotlivé funkční bloky.

Následná část se zabývá popisem deskriptoru, který je odesílán aplikací umístěnou na osobním počítači.

Závěr kapitoly se zabývá řešením pro funkční bloky, popisuje přístupy, jejich přednosti či nedostatky a u funkce osciloskop post - procesing dat, který probíhá v ovládací aplikaci.

3.1 Komunikační protokol

Komunikační protokol definuje formu přenášených dat a pakety. Cílem paketu je přenášet příkazy, nastavení a data mezi mikrořadičem a osobním počítačem. Definovaný paket musí být do značné míry promyšlený a připravený, tak aby pokryl nejen současné potřeby, ale byl i připravený na další rozšíření.

Výsledkem práce je jeden univerzální paket. Navrhovaný paket se skládá z obecné části a z části dané příslušným funkčním blokům.

3.1.1 Obecný popis paketu

Vytvořený paket se skládá z *hlavičky*, *typu paketu*, *typu funkčního bloku*, příkazu s určením, jaké části funkčního bloku patří a *délky* přenášených *dat*. Strukturu paketu ilustruje obrázek 3.1.

Hlavička	Typ	Funkční blok	Příkaz + sub-blok	Délka dat	Data
2 byty	1 byte	1 byte	1 byte	2 byty	0...65535 byte

Obrázek 3.1: Struktury paketu

3.1.2 Hlavička paketu

Hlavička paketu se skládá ze dvou bytů. Primárně slouží jako spolehlivý indikátor začátku paketu. Tabulka 3.1 zobrazuje hodnoty *hlavičky* v hexadecimálním tvaru.

	Hexadekadický zápis
1. byte	0xAA
2. byte	0xAA

Tabulka 3.1: Tabulka hodnot *hlavičky*

3.1.3 Typ paketu

Typ paketu se skládá ze dvou částí a to z části *směr komunikace* a *typu paketu*. Znak _ znamená libovolnou hodnotu. Výsledný byte je složený z obou částí.

Akce	Binární zápis
PC do mikrořadiče	01_ _ _ _ _ _
Mikrořadič do PC	10_ _ _ _ _ _

Tabulka 3.2: Tabulka hodnot typu paketu - *směr komunikace*

Akce	Binární zápis
Zakázat blok	_ _00 0000
Nastavení	_ _00 0001
Rezerováno	_ _00 0010
Data	_ _00 0100
Rezerováno	_ _00 1000
Informace	_ _01 0000
Povolit blok	_ _10 0000

Tabulka 3.3: Tabulka hodnot typu paketu - *typ paketu*

Nejprve definujeme *směry komunikace*. Tyto směry jsou za a) z mikroprocesoru do osobního počítače a za b) z osobního počítače do mikroprocesoru.

Rozlišuje se několik *typů paketu*. Jedná se především o *Povolení* a *Zakázání* funkčního bloku. Ty slouží k tomu, aby si uživatel mohl vybrat mezi funkcemi sdílejícími systémové prostředky. K nastavení hodnot mikroprocesoru slouží *Nastavení*, k vyčítání parametrů - *Informace* a pro zasílání dat - typ *Data*.

3.1.4 Funkční blok paketu

Funkční blok paketu slouží k adresování funkčního bloku mikroprocesoru. Protokol počítá i s budoucí implementací funkčního generátoru.

Funkční blok	Hexadecimální zápis
Mikroprocesor	0x00
Osciloskop	0x01
Čítač	0x02
Impulzní generátor	0x04
Funkční generátor	0x10
Voltmetr	0x20
Logický analyzátor	0x40

Tabulka 3.4: Tabulka hodnot *funkčního bloku* paketu

3.1.5 Délka dat paketu

Počet bytů dat je omezen hodnotou 65535. *Délka dat* oznamuje, kolik bytů má čtená/zapisovaná konfigurační proměnná nebo počet bytů dat.

3.1.6 Data paketu

Data jsou přenášena formou bytů. Jestli je daná hodnota více - bytová, řídicí aplikace data opětovně složí a přepočítá.

3.1.7 Příkaz a sub - blok paketu

Tato část protokolu je vytvořena na míru jednotlivým funkčním blokům. Každý blok má svoji unikátní příkazovou strukturu. Znak _ v následujících tabulkách označuje libovolnou hodnotu. Výsledný byte se skládá ze *sub - bloku* a *příkazu*.

Sub - bloky jsou tvořeny jednotlivými sub - části typů *funkčních bloků* a k nim je přidán globální typ pro nastavení celého *funkčního bloku*.

3.1.8 Příkaz funkčního bloku - mikroprocesor

Pro funkční blok *mikroprocesor* je připraven pouze jeden příkaz, a to pro získání deskriptoru zařízení.

Příkaz	Hexadecimální zápis
MCU info	0x00

Tabulka 3.5: Tabulka příkazu funkčního bloku *mikroprocesor*

3.1.9 Příkazy funkčního bloku - osciloskop

Pro funkční blok *osciloskop* jsou vymezeny příkazy v tabulce 3.6.

Příkaz	Hexadecimální zápis	Datová hodnota
Vzorkovací frekvence	0x_0	4 byty
Velikost bufferu	0x_1	4 byty
Triggrovaný kanál	0x_2	1 byte, 0, 1, 2, 3 kanál
Mód běhu	0x_3	1 byte, 0 - Auto, 1 - Normal
Hrana triggeru	0x_4	1 byte, 0 - Vzestupná, 1 - Sestupná
Napěťová úroveň triggeru	0x_5	2 byty
Procentuální vertikální trigger	0x_6	1 byte
Funkce Single	0x_7	1 byte
Funkce Start	0x_8	1 byte
Funkce Stop	0x_9	1 byte
Povolení kanálu	0x_A	1 byte, 0, 1, 2, 3 kanál
Zakázání kanálu	0x_B	1 byte, 0, 1, 2, 3 kanál
Reálná vzorkovací frekvence	0x_C	4 byty
Rozlišení osciloskopu	0x_D	1 byte, 0 - 8 - bit, 1 - 12 bit

Tabulka 3.6: Tabulka příkazů funkčního bloku *osciloskop*

Jednotlivé *sub - typy osciloskopu* 3.7.

Sub - blok	Hexadecimální zápis
Kanál 1	0x1_
Kanál 2	0x2_
Kanál 3	0x3_
Kanál 4	0x4_
Globální	0x8_

Tabulka 3.7: Tabulka *sub - bloků osciloskopu*

3.1.10 Příkazy funkčního bloku - logický analyzátor

Pro funkční blok *logický analyzátor* jsou vymezeny příkazy v tabulce 3.8.

Příkaz	Hexadecimální zápis	Datová hodnota
Vzorkovací frekvence	0x_0	4 byty
Velikost bufferu	0x_1	4 byty
Triggrovaný kanál	0x_2	1 byte, 0, 1, 2, 3 kanál
Mód běhu	0x_3	1 byte, 0 - Auto, 1 - Normal
Hrana triggeru	0x_4	1 byte, 0 - Vzestupná, 1 - Sestupná
Procentuální vertikální trigger	0x_6	1 byte
Funkce Single	0x_7	1 byte
Funkce Start	0x_8	1 byte
Funkce Stop	0x_9	1 byte
Reálná vzorkovací frekvence	0x_C	4 byty

Tabulka 3.8: Tabulka příkazů funkčního bloku *logický analyzátor*

Jednotlivé *sub - typy logického analyzátoru* 3.9.

Sub - blok	Hexadecimální zápis
Kanál 1	0x1_
Kanál 2	0x2_
Kanál 3	0x3_
Kanál 4	0x4_
Kanál 5	0x5_
Kanál 6	0x6_
Kanál 7	0x7_
Kanál 8	0x8_
Globální	0x9_

Tabulka 3.9: Tabulka *sub - bloků logického analyzátoru*

3.1.11 Příkaz funkčního bloku - voltmetr

Pro funkční blok *voltmetr* je vymezen příkaz v tabulce 3.10.

Příkaz	Hexadecimální zápis	Datová hodnota
Průměrování	0x_0	1 byte

Tabulka 3.10: Tabulka příkazu funkčního bloku *voltmetr*

Jednotlivé *sub - typy voltmetru* 3.11.

Sub - blok	Hexadecimální zápis
Kanál 1	0x1_
Kanál 2	0x2_
Kanál 3	0x3_
Kanál 4	0x4_
Globální	0x8_

Tabulka 3.11: Tabulka *sub - bloků voltmetru*

3.1.12 Příkazy funkčního bloku - impulzní generátor

Pro funkční blok *impulzní generátor* jsou vymezeny příkazy v tabulce 3.12.

Příkaz	Hexadecimální zápis	Datová hodnota
Frekvence	0x_1	4 byty
Střída	0x_2	1 byte
Reálná frekvence	0x_3	4 byty

Tabulka 3.12: Tabulka příkazu funkčního bloku *impulzní generátor*

Jednotlivé *sub - typy impulzního generátoru* 3.13.

Sub - blok	Hexadecimální zápis
Kanál 1	0x1_
Kanál 2	0x2_
Kanál 3	0x3_
Globální	0x8_

Tabulka 3.13: Tabulka *sub - bloků impulzního generátoru*

3.1.13 Příkazy funkčního bloku - čítač

Pro funkční blok *čítač* jsou vymezeny příkazy v tabulce 3.14.

Příkaz	Hexadecimální zápis	Datová hodnota
Průměrování	0x_2	1 byte
Frekvence časovače/čítače	0x_3	4 byty

Tabulka 3.14: Tabulka příkazu funkčního bloku *čítač*

Jednotlivé *sub - typy čítače* 3.15.

Sub - blok	Hexadecimální zápis
Kanál 1	0x1_
Kanál 2	0x2_
Kanál 3	0x3_
Globální	0x8_

Tabulka 3.15: Tabulka *sub - bloků čítače*

3.1.14 Příkazy funkčního bloku - funkční generátor

Pro funkční blok *funkční generátor* jsou vymezeny příkazy 3.16.

Příkaz	Hexadecimální zápis	Datová hodnota
Mod	0x_0	1 byte, 0, 1, 2, 3, napětí, čtverec, sinus, trojúhelník
Napětí	0x_3	2 byty
Frekvence	0x_2	4 byty
Offset	0x_4	2 byty
Reálná frekvence	0x_6	4 byty

Tabulka 3.16: Tabulka příkazu funkčního bloku *funkční generátor*

Jednotlivé *sub - typy funkčního generátoru* 3.17.

Sub - blok	Hexadecimální zápis
Kanál 1	0x1_
Kanál 2	0x2_
Kanál 3	0x3_
Globální	0x8_

Tabulka 3.17: Tabulka *sub - bloků funkčního generátoru*

3.2 Výběr zdroje systémového oscilátoru/krystalu

Požadavkem na nově vyvíjený systém je automatická detekce a nastavení krystalu v případě, kdy je připojen na dané piny HSE krystal nebo HSE - bypass.

HSE krystal je externí krystal připojovaný na vstupní piny mikrořadiče. Jeho použití již zde bylo diskutováno. Podobným způsobem pracuje HSE - bypass. Jedná se o hodinový signál, který je přiveden z nějakého zdroje signálu. Tímto způsobem se řeší hodinový signál u Nucleo desek.

Po startu mikrořadiče je povolen HSE vstup. Jestli bylo detekováno ustálení kmitočtu, pak dojde k povolení HSE jako zdroje systémových hodin a překonfigurují se parametry PLL obvodu. V případě neúspěchu je povolen vstup HSE - bypass a postup se opakuje.

Pokud i tento vstup selže, je zapnut RC HSI oscilátor a opět se překonfiguruje PPL. Detekce HSE krystalu je značena v deskriptoru.

Tato funkce je výhodná, pokud má daný mikroprocesor nedostatek pinů nebo se primární externí krystal poškodí.

3.3 Popis deskriptoru

Deskriptor je identifikátor daného mikroprocesoru. Jeho velikost je fixní a činí 198 bytů. Deskriptor byl navržen tak, aby respektoval maximální rozšiřitelnost grafického uživatelského rozhraní.

Maximální počet podporovaných sub - bloků je uveden níže v tabulce.

Sub - blok	Maximální počet
Osciloskop	4
Logický analyzátor	8
Voltmetr	4
Čítač	3
Impulzní generátor	3
Funkční generátor	3

Tabulka 3.18: Tabulka maximálního počtu podporovaných sub - bloků

Po připojení mikroprocesoru je zaslán deskriptor, který má několik úkolů. Nejprve zajistí vykreslení vhodného počtu sub - bloků jednotlivých funkčních bloků v grafickém rozhraní řídící aplikace. Dále informuje uživatele, jaké piny patří danému funkčnímu bloku, vykreslí typ procesoru a verzi firmwaru. Nakonec oznámí skutečnost, jaký zdroj signálu byl vybrán jako vstup do PLL obvodu (interní nebo externí) a primární komunikační rozhraní (USB nebo rozhraní USART).

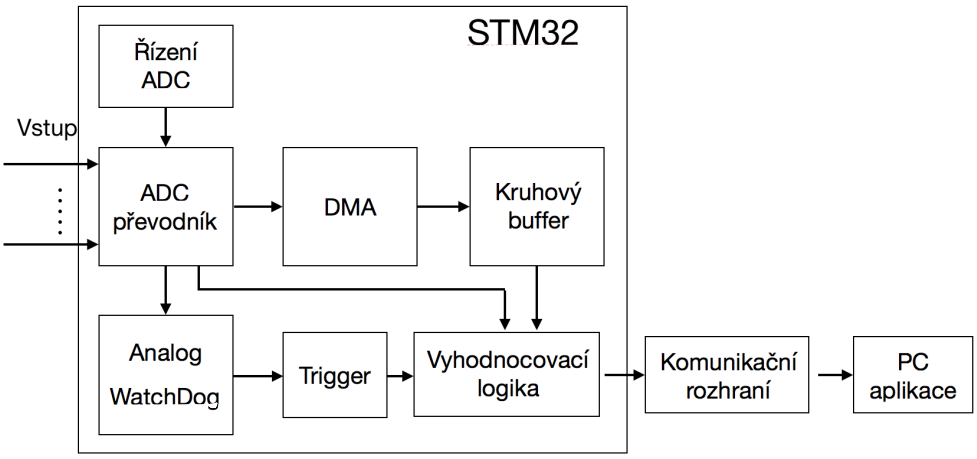
Výsledný deskriptor vypadá takto: STM32F303RE 01.00 080 E B B PA10 PA11 C PB10 PB11 A 4 PA01 PC02 PC00 PC01 G 0 0005 0006 0007 O 2 PA06 PB14 0010 I 1 PA10 0000 0000 V 4 PA01 PC02 PC00 PC01 L 4 PB00 PB01 PB02 PB03 0022 0023 0024 0025.

Název	Funkce
STM32F303RE	Typ procesoru
01.00	Verze firmwaru
080	Frekvence jádra procesoru
E	Typ vstupního oscilátoru, E-HSE, I-HSI
B	B - Primární komunikační rozhraní USB, C- USART
B PA10 PA11	Piny pro připojení USB
C PB10 PB11	Piny pro připojení USARTU
A 4 PA01 PC02 PC00 PC01	A-Označení pro osciloskop, 4 kanály, piny
G 0 0005 0006 0007	G-Označení pro funkční generátor, 0 kanálů, piny
O 2 PA06 PB14 0010	O-Označení impulzního generátoru, 2 kanály, piny
I 1 PA10 0000 0000	I-Označení čítače, 1 kanály, piny
V 4 PA01 PC02 PC00 PC01	V-Označení voltmetru, 4 kanály, piny
L 4 PB00 PB01 PB02 PB03	L-Označení logického analyzátoru, 4 piny
0022 0023 0024 0025	pokračování log. analyzátoru, piny

Tabulka 3.19: Popis jednotlivých částí deskriptoru

3.4 Osciloskop

Cílem vývoje osciloskopu je dosažení několika vlastností: ovládání vzorkovací frekvence i velikosti bufferu, triggrování v horizontální a vertikální ose, výběr sestupné nebo vzestupné hrany, podpora více kanálů, výběr kanálu pro triggrování, režim "RUN/STOP" a "Single", výběr režimu spouštění triggeru - automatický a normální mód, přepínání rozlišení 8 - bit nebo 12 - bit a post processing změřených dat v řídící aplikaci průměrováním a vyhlazením signálu.



Obrázek 3.2: Blokové schéma osciloskopu

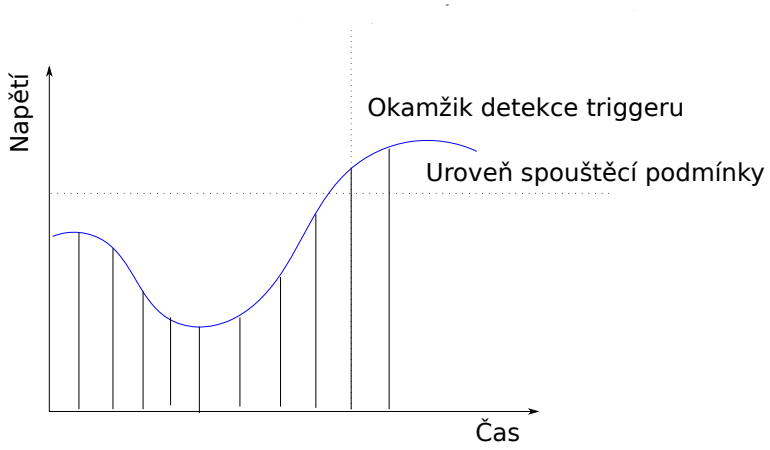
3.4.1 Přístupy realizace triggeru

Vzhledem k přítomným periferiím existuje několik přístupů, které jsou schopny realizovat trigger osciloskopu. Jestliže chceme realizovat pre - trigger resp. post - tigger, je nutné využít kruhový buffer. Jedná se o softwarový a hardwarový přístup.

3.4.1.1 Softwarový přístup realizace triggeru

Tento přístup využívá prostého periodického vyčítání hodnot z ADC jednotky. Vyčtená data se porovnají s napěťovou úrovní triggeru a uloží se do softwarově vytvořeného kruhového bufferu. Kdyby byl detekovaný okamžik triggeru 3.3, provede se ještě případný počet odběru vzorků, tak aby byla splněna vertikální pozice triggeru.

Výhodou tohoto řešení je jednoduchost realizace. Nevýhoda je velmi malá vzorkovací frekvence vůči teoretické. V případě STM32F042, kde teoretická vzorkovací frekvence je kolem 850kHz v konfiguraci jednoho kanálu, 12MHz zdroje hodin, 14 cyklů snímání a konverze byla realizovaná vzorkovací frekvence 100kHz. Tento přístup je tedy nevhodný.



Obrázek 3.3: Ilustrace detekce triggeru softwarovým přístupem

3.4.1.2 Hardwarový přístup realizace triggeru

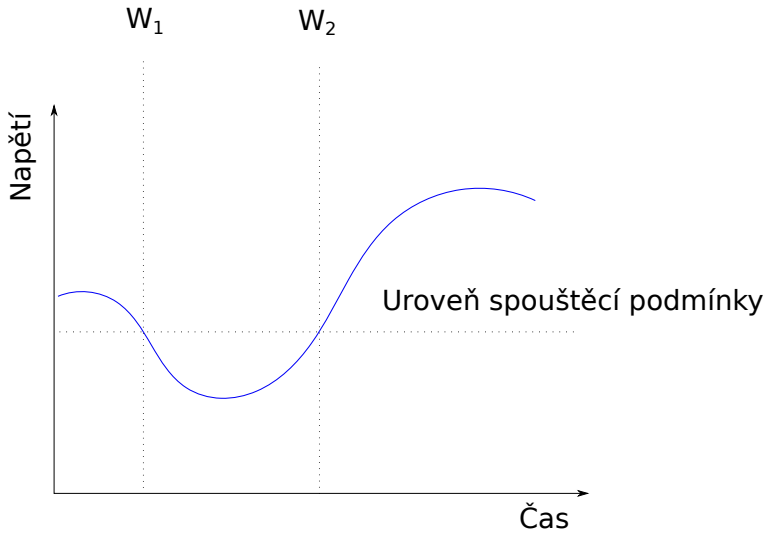
K implementaci tohoto přístupu potřebujeme DMA jednotku v režimu kruhového bufferu, přerušení od ADC jednotky oznamující ukončení převodu skupiny kanálů a analog Watchdog, který hlídá úroveň napětí.

Na začátku je spuštěn přenos z ADC jednotky do kruhového bufferu. V každém přerušení od ADC jednotky přičítáme počet vložených hodnot do kruhového bufferu. V okamžiku, kdy je dosaženo vertikální pozice bufferu, zapneme analog Watchdog jednotku spolu s přerušením a zastavíme počítání hodnot v bufferu. Tato jednotka je nakonfigurována tak, aby hlídala oblast mezi maximální hodnotou napětí a úrovní napětí na triggeru.

Obrázek 3.4 ilustruje průběh detekce triggeru. Jestliže dojde k přerušení od analog Watchdog jednotky, víme, že průběh se dostal pod úroveň triggeru, značka W_1. V tomto okamžiku překonfigurujeme analog Watchdog tak, aby hlídal hranici od nuly do úrovně

triggeru. V okamžiku W_2 nastane situace, která napoví, že se úroveň signálu dostala nad úroveň triggeru. Nyní stačí v přerušení od ADC jednotky dopočítat správný počet zbývajících konverzí a data odeslat do osobního počítače.

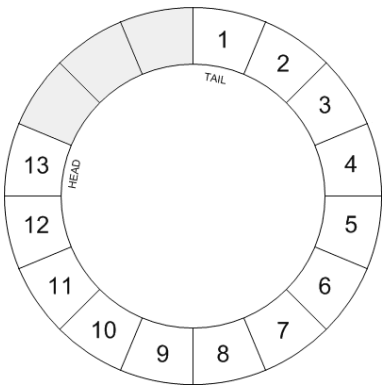
Nevýhoda je ta, že se jedná se o poměrně složitý návrh. Výhodou je rychlost reakce hardwarové jednotky. Je možné dosáhnout téměř teoretické maximální vzorkovací frekvence, což je cca 800kHz. Tato metoda byla vybrána do finální implementace.



Obrázek 3.4: Detekce triggeru pomocí analog Watchdog

3.4.2 Kruhový buffer

Kruhový buffer představuje nástroj, který ukládá data v "kruhu". Skládá se ze dvou ukazatelů. První ukazuje na novou pozici, kam se vloží nový prvek. Druhý ukazatel ukazuje na prvek, který bude přečtený. Řadič DMA umístěný v STM32 mikroprocesorech podporuje kruhový buffer.



Obrázek 3.5: Znázornění představy kruhového bufferu, převzato z [2]

3.4.3 Vzorkovací frekvence

Požadavkem osciloskopu je také konfigurace vzorkovací frekvence. ADC jednotka je propojena s výstupem z "TIMx" č/č, který při svém přetečení generuje TRGO událost. Tato událost spustí konverzi ADC jednotky.

Mikroprocesor není schopen nastavit libovolný vzorkovací kmitočet. To je způsobeno nedostatečně jemným dělícím poměrem při konfiguraci č/č hodnot "prescaler" a "autoreload". S rostoucí frekvencí klesá i přesnost nastaveného kmitočtu. Tato skutečnost by se nechala korigovat přizpůsobením frekvence samotného PLL obvodu. Takové řešení by však ovlivnilo i další části mikroprocesoru, především při snížení frekvence jádra i sníženou výkonnost. Nevýhody by převážily nad výhodami.

Výpočet frekvence je dán rovnicí 3.1. Můžeme si povšimnout přičtené jedničky. Je to z důvodu zápisu hodnoty do registru, kdy je nutné číslo o jedničku zmenšit. n_{PSC} je hodnota uložená v registru PSC, n_{ARR} je hodnota v registru ARR a f_{TIM} je zdrojová frekvence č/č.

$$f = \frac{f_{TIM}}{(n_{PSC} + 1)(n_{ARR} + 1)} \tag{3.1}$$

Uvažujeme-li f_{TIM} 48MHz a požadovaná frekvence by byla 10MHz. Pomocí výše popsaných veličin můžeme dosáhnout nejbližší frekvence 9,6MHz.

3.4.4 Módy triggeru osciloskopu

Osciloskop podporuje dva módy, a to mód automatický a mód normální. Jak již bylo zmíněno, liší se, za jakých podmínek triggrují. Realizovaný návrh se chová takto:

3.4.4.1 Automatický mód

Automatický mód čeká dobu n násobků naplnění bufferu a následně povolí analog Watchdog. Při každém přerušení od ADC je přičten počet kanálu. Jestliže finální počet je roven velikosti bufferu a nebyl detekovaný trigger, data se odešlou. Pokud je detekována triggrovací podmínka W_1 a W_2 , obrázek 3.4, dopočítá se počet zbývajících záznamů tak, aby odpovídal hodnotě vertikálního triggeru a dojde k odeslání dat.

3.4.4.2 Normální mód

Normální mód čeká do té doby než je počet vzorků v bufferu roven hodnotě vertikálního bufferu. Následně se spustí analog Watchdog a pozastaví se počítání vzorků. Analog Watchdog projde podmínkou označenou W_1 a W_2, obrázek 3.4. Při detekci podmínky W_2 je opět spuštěno počítání dat v bufferu tak, aby počet opět odpovídal hodnotě vertikálního bufferu.

3.4.5 RUN/STOP a Single

Funkce "RUN" okamžitě spustí č/č, který spouští konverzi ADC, "STOP" zastaví č/č. Single provede jeden odměr a zastaví činnost č/č.

3.4.6 Rozlišení osciloskopu

Defaultně je vybráno rozlišení 8 - bitů, a to z důvodu velikosti bufferu. Pokud bychom chtěli využít 12 - bitů, sníží se počet vzorků na polovinu.

3.4.7 Velikost bufferu

Velikost maximální velikost bufferu je dána velikostí kruhového bufferu.

3.4.8 Výběr triggrovaného kanálu

Výběr kanálu je proveden nastavením analog Watchdogu, tak aby hlídal pouze požadovaný kanál.

3.4.9 Povolení/zakázání kanálu za účelem vyšší vzorkovací frekvence nebo bufferu

Maximální vzorkovací frekvence se rozpočítává mezi povolené kanály. To samé platí i pro buffer tak, že čím je více povolených kanálů, tím je menší velikost bufferu na kanál.

3.4.10 Problém délky doby odběru vzorků

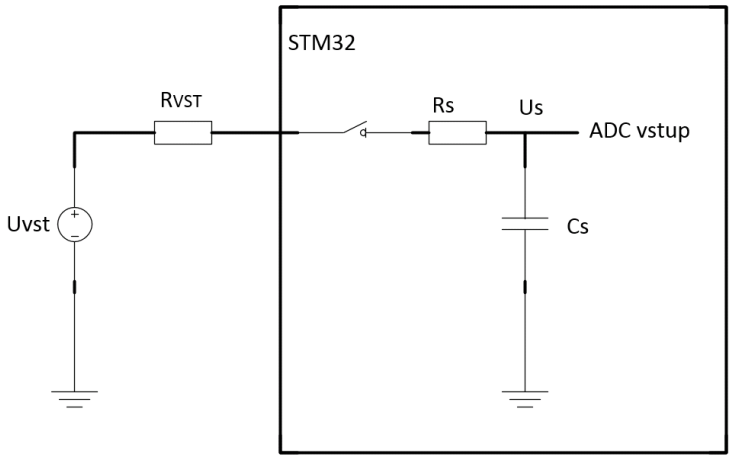
Jak již bylo dříve zmíněno, převod analogové hodnoty na digitální se skládá ze dvou částí. V první části se nabije vzorkovací kondenzátor a ve druhé části se snímané napětí převede na digitální hodnotu.

Problém nastává tehdy, pokud je délka doby odběru vzorku malá, daleko menší než $<< 5\tau$, kdy přechodový děj považujeme za ukončený. Vzorkovací kondenzátor se nestačí nabít na měřené napětí.

Tuto situaci můžeme vyřešit přidáním externího kondenzátoru nebo softwarovou dynamickou změnou délky doby odběru vzorku.

Časová konstanta τ_{RC} se vypočítá jako v rovnici 3.2, kde R_S je odpor spínače, R_{VST} je odpor zdroje a C_S je kapacita snímacího kondenzátoru.

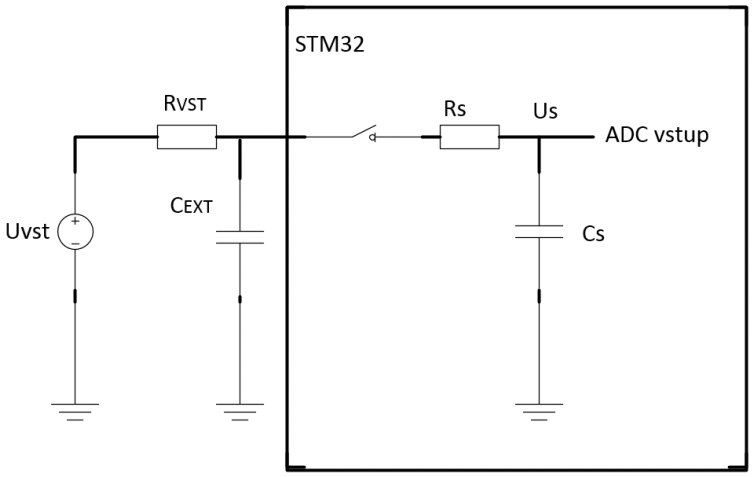
$$\tau_{RC} = (R_S + R_{VST})C_S \tag{3.2}$$



Obrázek 3.6: Zapojení ADC převodníku

3.4.10.1 Externí kondenzátor

Připojený externí kondenzátor pomůže vyrovnat proudový odběr a tím rychleji nabije snímací kondenzátor. Při volbě externího kondenzátoru je potřeba dbát na výslednou časovou konstantu, abychom si neomezili maximální frekvenci signálu.



Obrázek 3.7: Zapojení ADC převodníku s externím kondenzátorem

3.4.10.2 Dynamické nastavení délky doby odběru vzorku

ADC jednotky disponují konfigurovatelnou dobou odběru vzorku. Doba odběru vzorku ovlivňuje maximální vzorkovací frekvenci.

Ideálním řešením se ukazuje dynamické přepínání doby odběru vzorku v závislosti na počtu kanálů a vzorkovací frekvenci.

3.4.11 Post processing

Změřená data jsou dále zpracována v řídící aplikaci, kde dochází k průměrování nebo vyhlazení signálu pomocí filtru typu FIR.

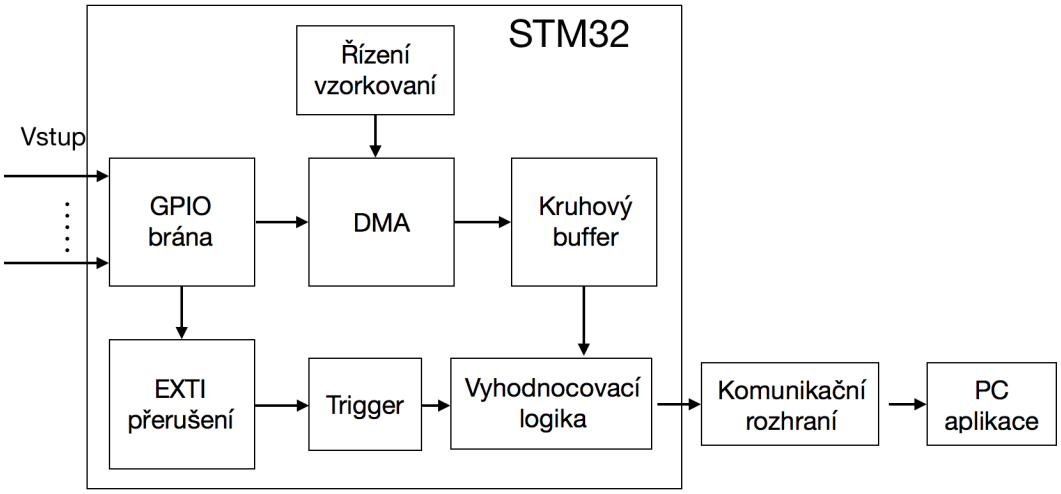
Pro funkci průměrování je nutné mít vhodně implementovaný trigger, aby nedocházelo k časovým nepřesnostem. Průměrování slouží k potlačení šumů v signálu.

Jedná se o jednoduchý číslicový filtr. Typem FIR filtru, tedy filtru s konečnou impulzní odezvou, je klouzavý průměr. Pro dosažení velké strmosti je potřeba velký řád filtru. Typem FIR filtru je tzv. klouzavý průměr.

3.5 Logický analyzátor

Výhodou logického analyzátoru je to, že nepotřebuje ADC převodník. Tím se vyhneme problému délky doby odběru vzorku. Signál je reprezentován logickými úrovněmi, tedy bity. Tím se nám zvýší počet dat, které se vejdou to bufferu.

Požadované vlastnosti logického analyzátoru: ovládání vzorkovací frekvence i velikosti bufferu, triggrování ve vertikální ose, výběr sestupné nebo vzestupné hrany, podpora více kanálů, výběr kanálu pro triggrování, režim “RUN/STOP” a “Single”, výběr režimu spouštění triggeru - automatický a normální mód.



Obrázek 3.8: Blokové schéma logického analyzátoru

3.5.1 Přístup realizace triggeru

Z vývoje osciloskopu bylo zřejmé, že je ideální použít veškeré možné hardwarové prostředky k dosažení nejlepších výsledků. Využije se tedy přímo hardwarový přístup.

3.5.1.1 Hardwarový přístup realizace triggeru

Opět budeme potřebovat DMA kruhový buffer, dále vstupní piny připojené na EXTI line přerušení, přerušení od DMA - přenos dokončen TC (transmit complete) a dva časovače.

Nastavíme první časovač, aby spouštěl DMA přenos po Update události . Dále nastavíme zdrojovou adresu na registr vstupů dané brány. Vzhledem k tomu, že tu není možnost kontrolovat, kolik bylo předáno vzorků do kruhového bufferu, tak se zprvu naplní celý buffer daty. DMA oznámí, že přenos byl dokončen. Mezi tím povolíme přerušení EXTI vstupního pinu, který chceme triggovat. Přerušení dokáže reagovat na nastavenou hranu (vzestupná, sestupná). Pokud je detekována daná hrana, spustí se druhý časovač, který si přepočítá ze vzorkovací frekvence a počtu potřebných vzorků dobu, která splní vertikální triggrovací podmínku.

Tímto lze dosáhnout vysoké vzorkovací frekvence. Nevýhoda DMA je ta, že přenáší minimálně 8 bitů. To znamená, že pokud chceme přenášet třeba jen čtyři kanály, každý kanál je jeden bit, a tak bude polovina přenášených dat k ničemu. Bohužel ruční úprava bufferu zabírá čas a zmenšili bychom si vzorkovací frekvenci.

3.5.2 Řešení spouštění logického analyzátoru

Analyzátor disponuje automatickým a normálním módem.

3.5.2.1 Automatický mód

Slouží ke zobrazování aktuálních signálů. Po startu se spustí DMA a časovač ke vzorkování. Po naplnění bufferu je zastaveno odebírání vzorků a data jsou odeslána. Po odeslání je opětovně spuštěna DMA jednotka a časovač.

3.5.2.2 Normální mód

Normální mód je popsán v části 3.5.1.1 .

3.5.3 Výběr triggrovaného kanálu

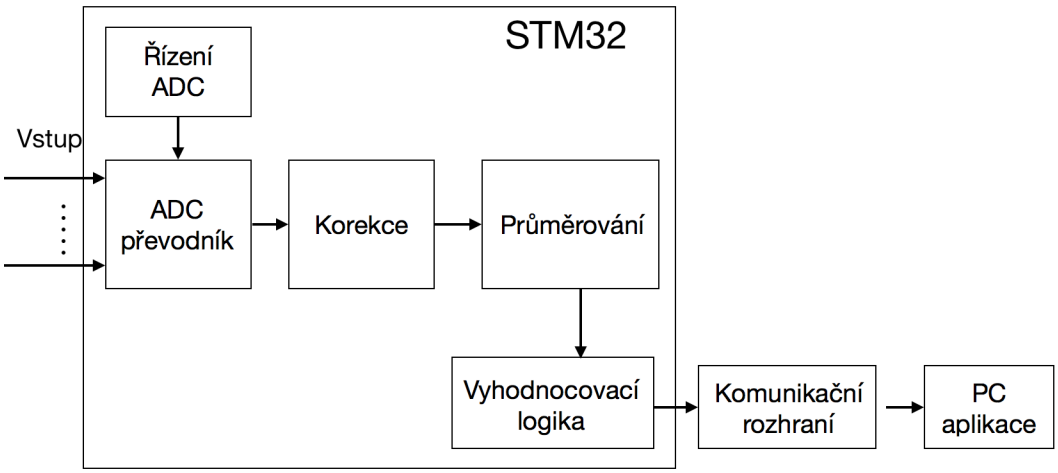
Výběr kanálu je proveden nastavením EXTI přerušením na daný vstupní pin.

3.6 Voltmetr

Cílem voltmetru je přesné měření napětí. ADC jednotka konvertuje hodnotu v rozmezí nulového a referenčního napětí ADC $U_{REF}+$. V případě, kdy má procesor málo vývodů, je spojeno přímo referenční s napájecím napětím ADC jednotky $VDDA$.

Ideální stav je ten, kdy napájecí napětí ADC jednotky je stabilní. Bohužel se v praxi stává, že toto napájecí napětí kolísá. K dosažení nejpřesnějšího změřeného napětí

lze použít dvě metody. První metoda je korekce pomocí vnitřní reference a druhá je průměrování.



Obrázek 3.9: Blokové schéma voltmetru

3.6.1 Korekce měřeného napětí pomocí reference a konstanty

STM32 obsahující ADC jednotky mají ve své paměti uloženou hodnotu $VREFINT_{CAL}$. Jedná se o hodnotu, díky které jsme schopni spolu se surovou hodnotou z interní napěťové reference dopočítat skutečnou hodnotu napájecího napětí $VDDA$ ADC jednotky a kompenzovat tak výkyvy napájecího napětí [7].

Výpočet skutečného napájecího napětí je dán 3.3. $VREFINT_{CAL}$ je hodnota uložená v paměti. $VREFINT_{VAL}$ je změřená surová hodnota vnitřní reference a 3.3 je napájecí napětí.

$$VDDA = 3.3\frac{VREFINT_{CAL}}{VREFINT_{VAL}} \tag{3.3}$$

Korigovaná změřená hodnota napětí je dána jako v rovnici 3.4. U_{Chan} je skutečná hodnota změřeného napětí, ADC_{Chan} je surová hodnota napětí a 4095 je dělicí poměr vyjádření pro 12 - bitovou hodnotu.

$$U_{Chan} = VDDA\frac{ADC_{Chan}}{4095} \tag{3.4}$$

3.6.2 Průměrování

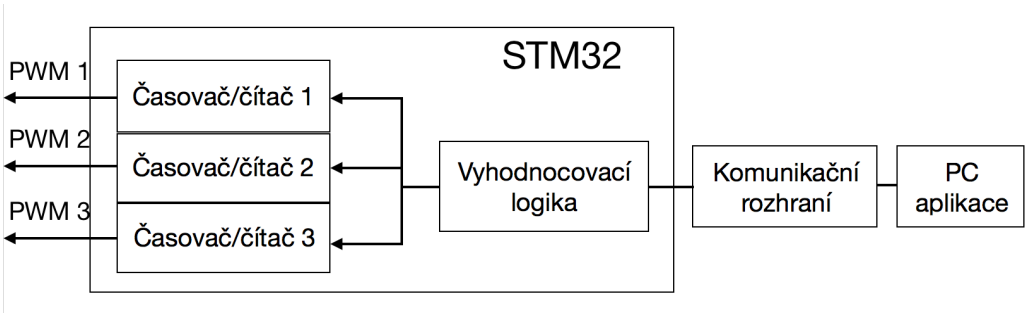
Funkce průměrování slouží k potlačení šumů, které se vykytují na měřeném signálu. Takovým to rušením může být rušení ze síťového napětí.

3.6.3 Implementace

K realizaci funkce budeme potřebovat ADC jednotku, časovač, DMA a připojení ADC jednotky na referenci napětí. Cílem je nastavení vzorkovací frekvence 100Hz. Vzhledem k nízké vzorkovací frekvenci nastavíme ADC jednotku na nejdelší možný odběr vzorku. Dále přizpůsobíme frekvenci ADC jednotky. Vzhledem k omezenosti prostředků procesoru je spouštěna konverze softwarově v Update přerušení od časovače. Ukončení konverze a překopírování dat oznámí DMA jednotka, která poskytuje surové hodnoty jednotlivých kanálů a vnitřního referenčního napětí. Ačkoliv by bylo možné počítat korekci a průměrování na osobním počítači, lze uspořit provoz dat na komunikační sběrnici tím, že si provedeme výpočet přímo v mikroprocesoru. V přerušení od DMA se uskuteční také přepočet a korekce změřeného napětí. Pokud je zvolena funkce průměrování, uloží se současné hodnoty do pomocných registrů a před zasláním dat se napětí zprůměruje. Zprůměrované hodnoty jsou odeslány do osobního počítače.

3.7 Impulzní generátor

Impulzní generátor slouží ke generování pulzů o dané frekvenci a střídě. Cílem implementace je umožnění nastavení obou parametrů pomocí časovačů/čítačů.



Obrázek 3.10: Blokové schéma impulzního generátoru

3.7.0.1 Generování PWM signálu

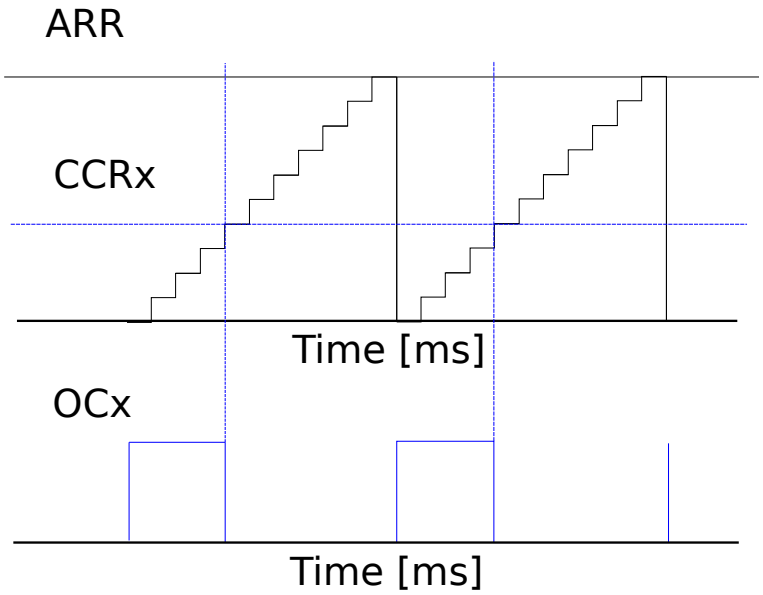
Nastavíme uvažovaný časovač/čítač do režimu PWM. Výsledná frekvence signálu je dána pomocí již známých parametrů "prescaler" a "autoreload" nebo-li registry PSC (prescaler) a ARR (auto reload register). Nakonec je určen poměrem mezi CCRx (capture/compare registr) a ARR dané Output Compare jednotky.

Nutností je vybrat správnou hodnotu registru ARR vůči PSC při volbě generovaného signálu. Pokud by byla hodnota ARR malá a PSC velká, tak by došlo k velmi hrubému nastavení střídy.

Software má implementovaný mechanismus, který inkrementuje hodnotu PSC, tak aby byl postupně splněn požadovaný dělící poměr k vytvoření dané frekvence. Dále se testuje, zda-li je splněna podmínka, že ARR je menší než 16 - bitová hodnota. Výčet

frekvence je stejný jako ve vzorci 3.1. Výpočet střídy je uveden ve vzorci 3.5. n_{ARR} značí hodnotu v registru ARR, n_{CRRx} je hodnota v capture/compare x registru.

$$Duty = \frac{(n_{CRRx} + 1)}{(n_{ARR} + 1)} 100 \tag{3.5}$$

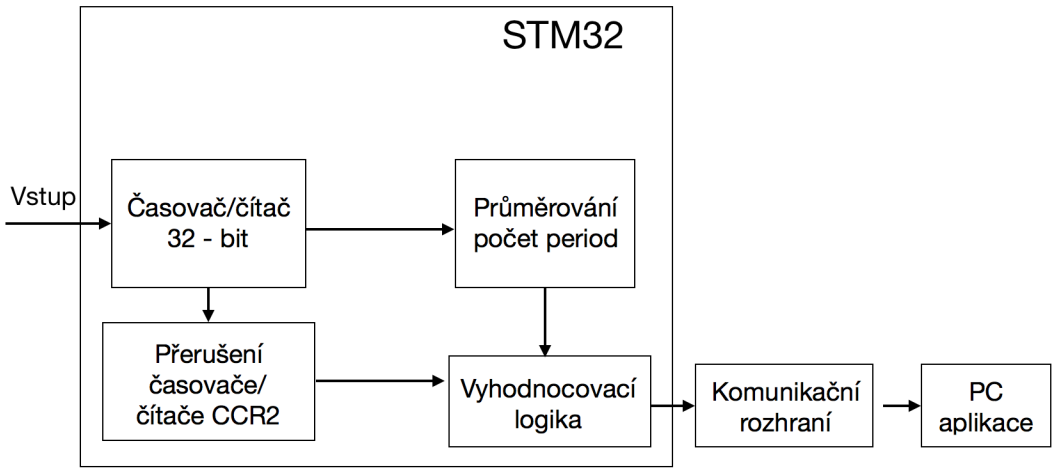


Obrázek 3.11: Princip generování PWM signálu

3.8 Čítač

Účelem čítače je měření frekvence, případně periody a střídy signálu. Nutnou podmínkou pro měření je splnění vzorkovacího teorému.

Dalším důležitým parametrem je rozlišení čítače. Toto rozlišení dává představu, kolik hodnot může čítač čítat. Pokud máme 16 - bitový čítač, lze čítat 65535 hodnot, naopak pokud bychom měli 32 - bitový čítač, výsledný počet může být cca 4,3G hodnot.



Obrázek 3.12: Blokové schéma čítače

3.8.1 Přístupy měření frekvence

Existují dva základní principy měření frekvence signálu. Tyto principy jsou tzv. přímé měření frekvence a reciproční měření periody (frekvence).

3.8.1.1 Přímé měření frekvence

Tento princip je založený na jednom časovači, který slouží jako hradlování a čítači, který počítá počet pulzů vstupního signálu. Uživatel by si vybral dobu hradlování. Pokud by doba hradlování byla jedna sekunda, maximální frekvence měřitelná 16 - bitovým čítačem by byla přibližně 65kHz a 32 - bitovým cca 4,3GHz. Pokud bychom zvolili dobu hradlování 10 sekund, maximální měřitelná frekvence by byla 10 krát menší.

Nevýhoda tohoto přístupu spočívá v potřebě dvou až tří volných časovačů/čítačů. Jedná se o záležitost, kdy 16 - bitový čítač změří malou frekvenci, tedy je nutný 32 - bitový čítač. Pokud chybí volný 32 - bitový čítač, tak lze dva 16 - bitové čítače propojit. Další nevýhodou je to, že dobu měření určuje jednak rozsah měření, ale i přesnost. Přesnost roste s dobou hradlování, naopak rozsah s dobou hradlování klesá. Tento přístup je nevyhovující.

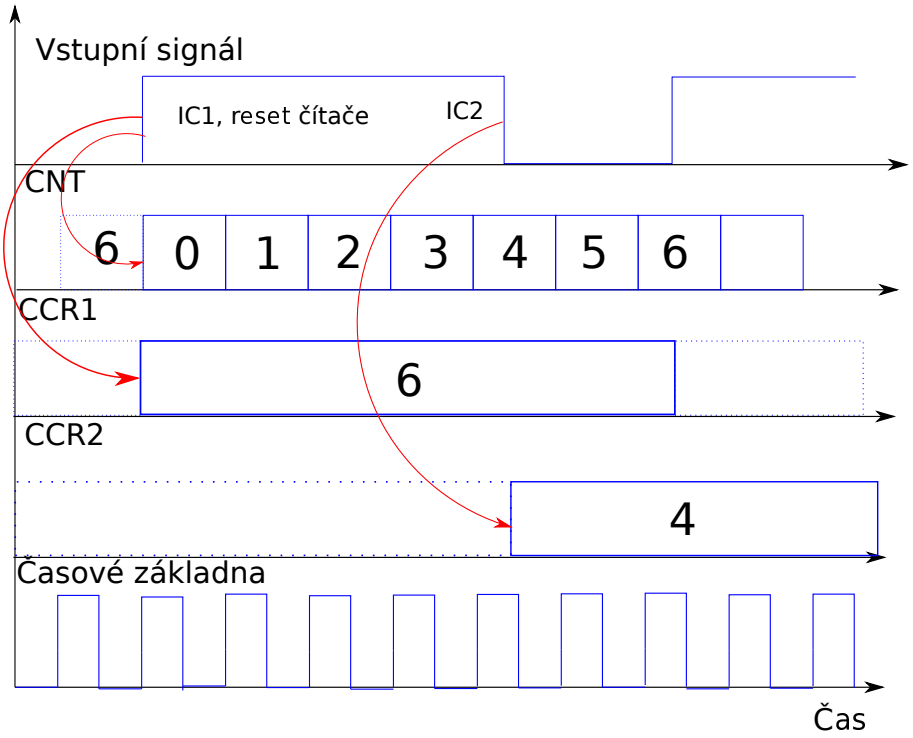
3.8.1.2 Reciproční měření frekvence

V této implementaci si vystačíme s jedním 32 - bitovým čítačem, kde na jeden kanál přivedeme signál přímo a na druhý kanál nepřímo. Využijeme tady funkci Input Capture. Kanál č. 1 je nastaven tak, aby začal počítat od vzestupné hrany. Kanál č. 2 je nastaven, aby počítal od sestupné hrany. Nastavíme Reset režim, kdy při příchodu vzestupné hrany se čítač vynuluje a začne počítat od nuly. Nastavíme si přerušení, které oznámí konec měření, registr CCR2.

Po ukončení měření máme hodnoty v registru, které když přepočítáme, zjistíme, jaká byla měřená frekvence a jaká byla střída. Obrázek níže ilustruje měření frekvence. Dalším

cílem je průměrování. To lze implementovat přidáním 64 bitové proměnné, která se při přerušení bude neustále inkrementovat a následně po dosažení daného počtu odběrů se proměnná zprůměruje. Zprůměrované hodnoty z registru CCR1 a CCR2 se odešlou řídící aplikaci.

Výhodou tohoto přístupu je snadná realizace využívající pouze jeden časovač/čítač. Nevýhodou je růst chyby měření s rostoucí frekvencí.



Obrázek 3.13: Princip měření frekvence

Výsledná frekvence se vypočítá jako v rovnici 3.6.

$$f_{MEAS} = \frac{f_{TIM}}{CCR1} \tag{3.6}$$

Výsledná střída se vypočítá jako 3.7.

$$Duty = \frac{CCR2}{CCR1}100 \tag{3.7}$$

3.8.2 Přesnost měření

Přesnost měření v případě recipročního měření frekvence je dána přesností časové základny signálu. Jak již bylo zmíněno, vyšší přesnosti lze dosáhnout připojením externího HSE krystalu namísto interního HSI RC oscilátoru. Další možností je využití HSI48 oscilátoru, který je průběžně dolaďován zdrojem ze signál z USB sběrnice.

Kapitola 4

Praktická realizace

V této kapitole si popíšeme finální realizaci grafického uživatelského rozhraní. Rozebereme si vlastnosti grafických částí pro jednotlivé *funkční bloky* následované vývojovými diagramy. Tato část je koncipovaná také jako návod pro uživatele.

4.1 Grafické uživatelské rozhraní

Grafické uživatelské rozhraní poskytuje pohodlný nástroj pro ovládání a vizualizaci změřených dat. Grafické rozhraní bylo vyvinuto ve frameworku Qt. Aplikaci lze spustit na platformách Windows a macOS. Zvolený jazyk aplikace je anglický.

Aplikace se skládá z hlavní obrazovky a obrazovek daných funkčních bloků.

4.1.1 Grafické uživatelské rozhraní hlavní obrazovky

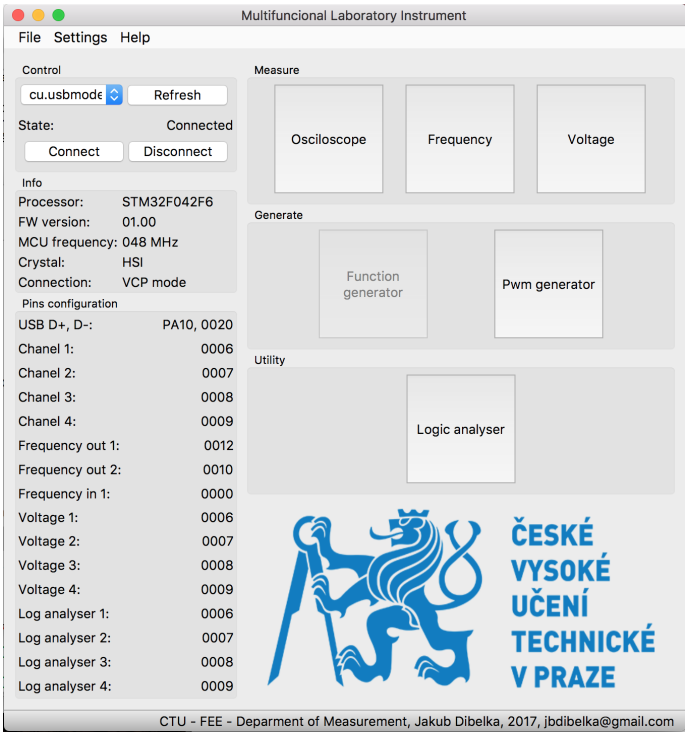
Skládá se z ovládacího menu sloužícího k výběru sériového portu, informační části a části spouštějící funkční bloky.

V levé části nalezneme *Control*, zde se vyskytují tlačítka *Refresh*, *Connect*, *Disconnect*. Tlačítko *Refresh* slouží k nalezení přítomných sériových portů. Tlačítkem *Connect* se připojíme k vybranému portu 4.2 za a). Tlačítkem *Disconnect* se od portu odpojíme.

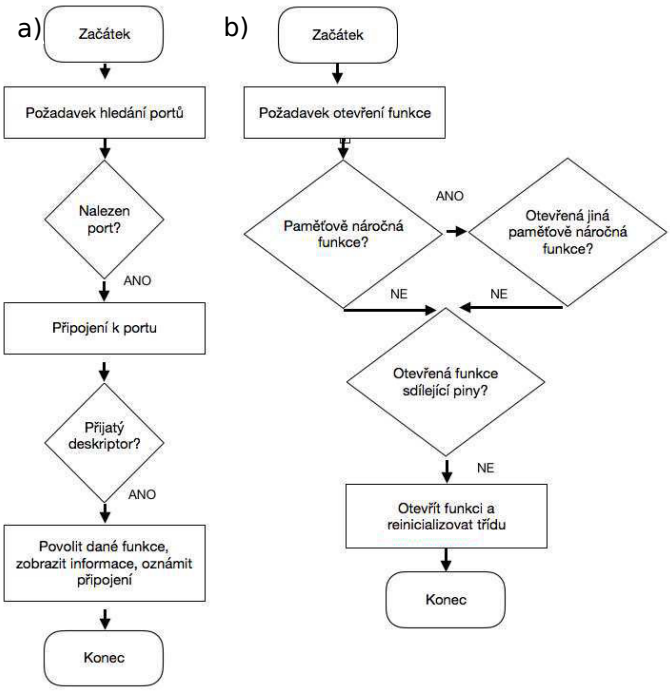
Část *Info* zobrazuje informace o typu procesoru, verzi firmwaru, frekvenci jádra procesoru, typu krystalu a primární způsob připojení. *Pins configuration* popisuje piny, které náleží jednotlivým periferiím.

V pravé části nalezneme tlačítka v sekcích *Measure*, *Generate* a *Utility*. Tlačítka v sekci *Measure* spustí osciloskop, čítač nebo voltmetr. V sekci *Generate* spustíme impulzní generátor. V poslední sekci *Utility* se nachází logický analyzátor. Příslušným kliknutím na tlačítko otevřeme danou funkci.

Vzhledem ke sdíleným pinům nebo paměťovému nároku na jednotlivé funkční bloky, obsahuje software omezení dovolující otevřít pouze volné bloky. V opačném případě nedovolí funkční blok otevřít 4.2 za b).



Obrázek 4.1: Grafické uživatelské rozhraní úvodní obrazovky



Obrázek 4.2: Vývojové diagramy, a) otevření komunikace, b) otevření funkce osciloskop

4.2 Grafické uživatelské rozhraní osciloskopu

Osciloskop se skládá ze dvou částí. První část vizualizuje změřený signál. Druhá část slouží k ovládání.

4.2.1 Popis ovládacích prvků osciloskopu

Prvním ovládacím prvkem je volba zobrazení daného kanálu. Pokud přestaneme zobrazovat daný kanál, zvýšíme, tím maximální vzorkovací frekvenci a také velikost bufferu.

Dalšími možnostmi je posun kanálu, tzv. offset. Ten se používá pro posun kanálu, tak aby se nám signály daných kanálů nepřekrývaly. U každého kanálu je možné zobrazit jednotlivé body.

Následuje volba vzorkovací frekvence a velikosti bufferu. U vzorkovací frekvence je zobrazena skutečná reálná vzorkovací frekvence.

Možností je volba triggrovacího kanálu, výběr módů (automatický a normální), výběr vzestupné či sestupné hrany a rozlišení. K posunu úrovně triggeru slouží posuvníky. Následují tlačítka pro zastavení a běh osciloskopu a tlačítko pro zobrazení jednoho průběhu a zastavení činnosti.

V další části je umístěno ovládání zobrazení vertikálního a horizontálního kurzoru.

K vyhlazení signálu slouží možnost *Smooth signal* a k průměrování *Average signal*. Dále je zde umístěno tlačítko k uložení aktuálního snímku grafu.

V poslední části leží výběr cesty k uložení dat a ovládání. Pro ukládání je potřeba zmáčknout tlačítko *Start saving* a pro ukládání do nového souboru *New file*.

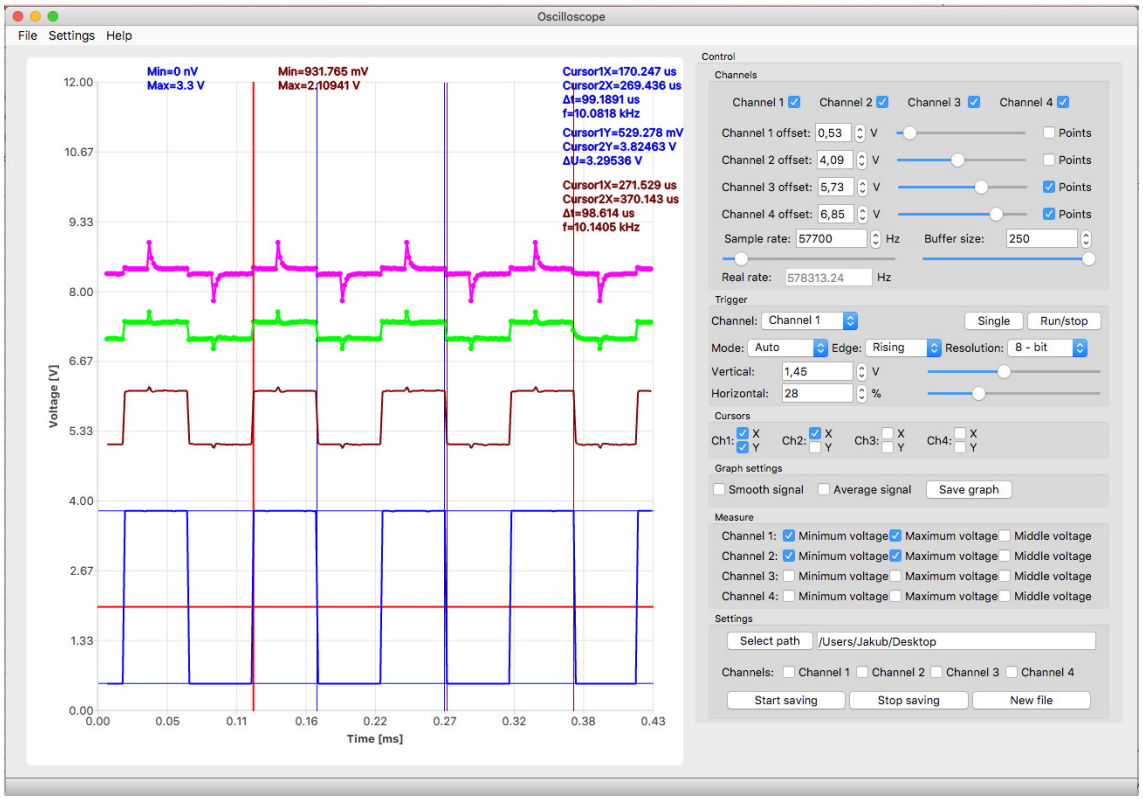
4.2.2 Popis grafu osciloskopu

Operace na grafu 4.3 jsou přibližování, oddalování, posun a posun kurzorů. Přibližování je uskutečněno kliknutím a držením levého tlačítka myši. Následným posunem vybereme oblast, která nás zajímá. Oddalování je realizováno kliknutím pravého tlačítka myši. Posun grafu je realizován otočením kolečka myši.

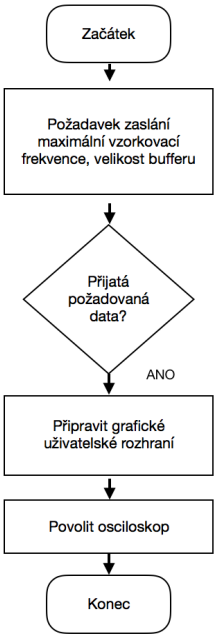
Pro posun kurzorů je nutné kliknou levým tlačítkem myši, tím se "přichytí" měřící kurzor na kurzor myši a je umožněn jeho pohyb. Po umístění kurzoru stačí opětovně zmáčknout levé tlačítko myši.

V grafu se zobrazují signály jednotlivých kanálů. Každý kanál je barevně odlišen. Modrá barva je pro kanál 1. Hnědá značí kanál 2. Zelená barva označuje kanál 3 a růžová kanál 4. Dále se v grafu zobrazují kurzory, popisky kurzorů a automaticky změřené hodnoty (minimální, průměrná a maximální hodnota napětí). Jednotlivé barvy jsou značeny stejně jako u signálů. Tučnou červenou barvou je zvýrazněna hodnota vertikálního a horizontálního triggeru.

Popisky kurzorů zobrazují hodnotu daného kurzoru a jejich rozdíl.



Obrázek 4.3: Grafické uživatelské rozhraní osciloskopu



Obrázek 4.4: Vývojový diagram inicializace osciloskopu

Zkratka	Funkce
Q	Defaultní pozice kurzorů
R	Resetuje zoom
Y	Nastaví vertikální pozici do 0 a maximální stávající hodnoty z měřítka
X	Nastaví horizontální pozici do 0 a maximální stávající hodnoty z měřítka

Tabulka 4.1: Tabulka funkčních zkratek pro graf osciloskopu

4.2.3 Výsledky realizovaného osciloskopu pro jednotlivé mikroprocesory

Pro implementované procesory bylo dosaženo vzorkovací frekvence a velikosti bufferu viz. tabulka 4.2.

Mikroprocesor	Maximální vzorkovací frekvence	Velikost bufferu
STM32F042F6	800kHz	1000
STM32F303RE	2,5MHz	20000

Tabulka 4.2: Tabulka výsledků osciloskopu

4.3 Grafické uživatelské rozhraní logického analyzátoru

Logický analyzátor se skládá podobně jako osciloskop ze dvou částí, a to z vizualizující části a částí, kde jsou umístěny ovládací prvky.

4.3.1 Popis ovládacích prvků logického analyzátoru

V první části můžeme vybrat signál, který má být zobrazen. K ovládání zobrazení jednotlivých bodů slouží volba *Points*. Následuje výběr vzorkovací frekvence a velikosti bufferu. U vzorkovací frekvence je zobrazena skutečná vzorkovací frekvence.

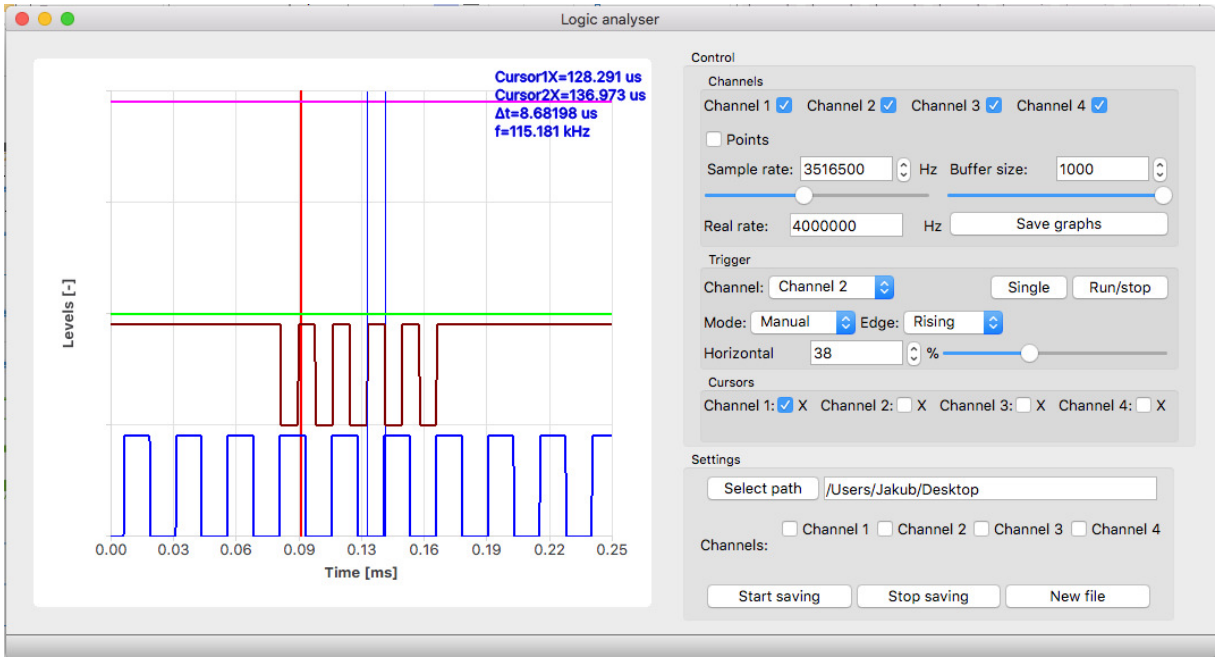
Dále je tu opět možnost uložit aktuální graf, spustit a zastavit analyzátor, vybrat triggrovací kanál, zvolit mód a hranu triggeru.

Další část dovolí ovládat kurzory a vypsat vypočtené hodnoty do grafu.

Jako v případě osciloskopu můžeme ukládat data do .cvs formátu a volit cestu, kam mají být uložena.

4.3.2 Popis grafu logického analyzátoru

Zde podobné vlastnosti jako u osciloskopu 4.2.2.



Obrázek 4.5: Grafické uživatelské rozhraní logického analyzátoru

4.3.3 Výsledky realizovaného logického analyzátoru pro jednotlivé mikroprocesory

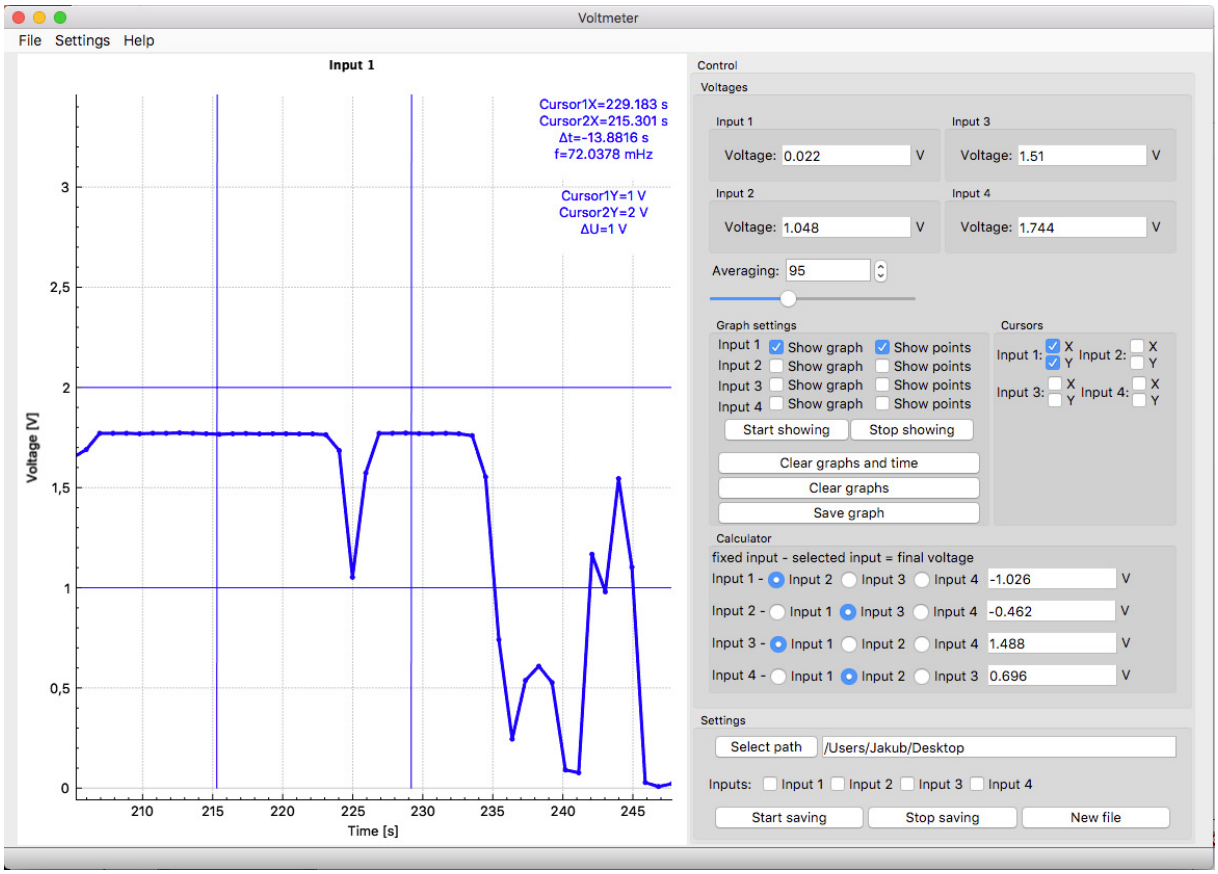
Pro implementované procesory bylo dosaženo vzorkovací frekvence a velikosti bufferu viz. tabulka 4.3.

Mikroprocesor	Maximální vzorkovací frekvence	Velikost bufferu
STM32F042F6	8MHz	1000
STM32F303RE	12MHz	20000

Tabulka 4.3: Tabulka výsledků logického analyzátoru

4.4 Grafické uživatelské rozhraní voltmetru

Voltmetr zobrazuje až 4 hodnoty měřeného napětí. Volitelně lze zobrazit i graf.



Obrázek 4.6: Grafické uživatelské rozhraní voltmetru

4.4.1 Popis ovládacích prvků a vlastností voltmetru

Defualtně jsou grafy schované. Každý graf reprezentuje jeden kanál. Ačkoliv graf není zobrazen, hodnoty se do paměti grafu postupně ukládají a při zobrazení grafu se vykreslí uložené signály.

Volitelně můžeme zobrazovat a schovávat změřené body a zobrazit kurzory. Tento graf byl realizován knihovnou *QCustomPlot* a to z důvodu, že grafy poskytované Qt nejsou schopny vysoké vykreslovací frekvence.

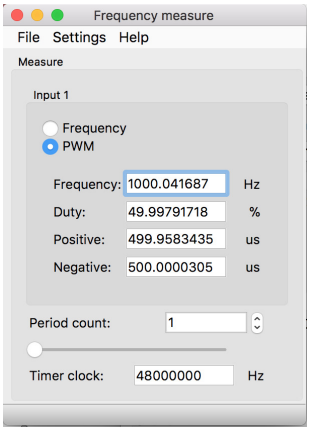
Pro nahrávání hodnot do grafu slouží tlačítko *Start showing*. Tlačítkem *Clear graphs and time* smažeme buffer grafů a nastavíme počátek časové osy na čas 0. Tlačítko *Clear graphs* smaže pouze buffer grafu. Tlačítkem *Save graph* uložíme stávající snímek grafu.

Opět je připravena možnost ukládání změřených dat do souboru.

Jako pomocný nástroj studentům tu byla implementována kalkulačka, která umožní od kanálu odečíst další kanál.

4.5 Grafické uživatelské rozhraní čítače

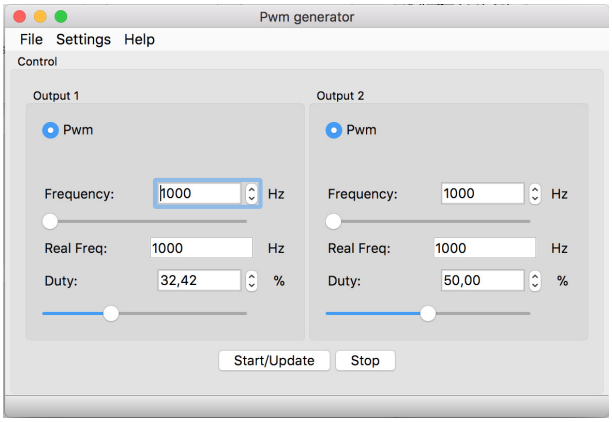
Čítač zobrazuje změřenou frekvenci, střídu a dobu, kdy byl signál v logické jedničce a kdy v logické nule. Vyskytuje se zde možnost volitelného počtu měření period.



Obrázek 4.7: Grafické uživatelské rozhraní čítače

4.6 Grafické uživatelské rozhraní impulzního generátoru

Umožňuje nastavit frekvenci generovaného signálu spolu s hodnotou střídy. Také je zde zobrazena skutečná frekvence generovaného signálu.

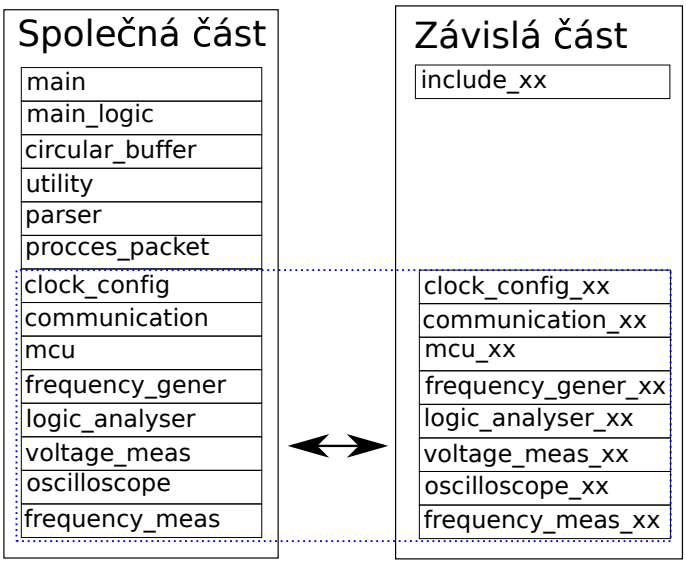


Obrázek 4.8: Grafické uživatelské rozhraní impulzního generátoru

4.7 Popis struktury firmwaru pro mikrořadič

Výsledný firmware se skládá ze dvou částí. První část obsahuje univerzální část pro všechny mikroprocesorové řady. Druhá část se skládá z mikroprocesorově závislých částí. Výsledná struktura je zobrazena na obrázku 4.9.

Pro modře vyznačenou část platí, že jednotlivé části jsou provázané. Ve společné části je definováno základní nastavení, vyčítání informací a ovládání mikroprocesorově závislé části. Písmena *xx* značí mikroprocesorovou řadu.



Obrázek 4.9: Struktura firmwaru mikrořadiče

4.8 Shrnutí vlastností výsledného multifunkčního laboratorního přístroje

Shrnutí výsledků pro mikroprocesor STM32F042F6:

- **Zdroj systémového hodinového signálu dle priority**: 1) Přítomnost HSE krystalu - HSE, 2) Přítomnost HSE bypass - HSE bypass, 3) USB komunikace - HSI48, 4) HSI oscilátor
- **Osciloskop**: 4 kanály, maximální vzorkovací frekvence 1Hz až 800kHz, velikost bufferu 1000 bytů, rozlišení 8, 12 - bitů
- **Logický analyzátor**: 4 kanály, maximální vzorkovací frekvence 1Hz až 8MHz, velikost bufferu 1000 bytů
- **Voltmetr**: 4 kanály, každý kanál vzorkovaný 100Hz, korekce vstupního napětí

- **Frekvenční generátor**: 2 nezávislé PWM generátory, generovaná frekvence 1Hz až 16MHz a střída 0 % až 100 %
- **Čítač**: 1 kanál, maximální měřitelná frekvence 24MHz.

Shrnutí výsledků pro mikroprocesor STM32F303RE:

- **Zdroj systémového hodinového signálu dle priority**: 1) Přítomnost HSE krystalu - HSE, 2) Přítomnost HSE bypass - HSE bypass, 3) HSI oscilátor
- **Osciloskop**: 4 kanály, maximální vzorkovací frekvence 1Hz až 2,5MHz, velikost bufferu 20000 bytů, rozlišení 8, 12 - bitů
- **Logický analyzátor**: 4 kanály, maximální vzorkovací frekvence 1Hz až 12MHz, velikost bufferu 20000 bytů
- **Voltmetr**: 4 kanály, každý kanál vzorkovaný 100Hz, korekce vstupního napětí
- **Frekvenční generátor**: 2 nezávislé PWM generátory, generovaná frekvence 1Hz až 16MHz a střída 0 % až 100 %
- **Čítač**: 1 kanál, maximální měřitelná frekvence 36MHz.

Kapitola 5

Závěr

Účelem práce byl návrh programově definovaného multifunkčního laboratorního přístroje, který dokáže měřit, generovat signál a pomocí můstku UART/USB nebo prostřednictvím rozhraní USB přenášet data do nadřazeného osobního počítače, kde řídící aplikace přijatá data zobrazí a vyhodnotí.

V rámci této práce vzniklo programové vybavení pro platformu STM32. Toto vybavení se skládá ze společné části, které sdílí všechny mikrořadiče a části, která je závislá na jednotlivých řadách STM32. Tímto je připraven základ pro expanzi na další mikroprocesorové řady. Každý procesor obsahuje funkci automatické detekce připojení externího krystalu. V opačném případě se vybere interní HSI oscilátor jako zdroj systémového hodinového signálu. Každý procesor obsahuje deskriptor, který zahrnuje základní informace o mikroprocesoru, využitých pinech a systémové frekvence. Data jsou přenášena přes můstek nebo rozhraní USB. Byl vytvořen unikátní formát paketu, který dovolí přenášet, nastavovat a číst z jednotlivých funkčních bloků procesoru. Dále je připraven na použití nad rámec této diplomové práce. Testovány byly mikroprocesory STM32F042F6, STM32F042K6 a STM32F303RE.

Řídící aplikace pro uživatelský počítač byla napsána ve frameworku Qt. Výhodou Qt je jeho multiplatformita. Vyvíjenou aplikaci lze nyní spustit na platformě Windows a macOS. Aplikace je tvárná a dokáže reagovat na deskriptor, který oznámí, kolika a jakými typy měřících či generujících bloků mikroprocesor disponuje. V případě měření, dokáže aplikace měřená data průběžně ukládat na pevný disk uživatele, dokáže ovšem ukládat i snímky z grafu. Tato funkce je vhodná např. pro studenty.

V rámci diplomové práce jsem splnil zadání. Realizoval jsem programově definovaný multifunkční laboratorní přístroj pro platformu STM32. Vytvořil osciloskop, logický analyzátor, voltmetr, čítač a impulzní generátor. Dále jsem vytvořil řídící aplikaci pro osobní počítač. Zařízení plně funguje a je ověřeno.

Literatura

[1] Becomingmaker.com. USI Serial UART Send on ATtiny [online], 2017. [cit. 2017-01-4], ⟨https://i0.wp.com/becomingmaker.com/wp-content/uploads/2016/07/serialtiming.png⟩.

[2] hadoopabcd.com. Circular Buffer [online], 2015. [cit. 2017-01-4], ⟨https://hadoopabcd.files.wordpress.com/2015/02/circularbuffer.png⟩.

[3] National instruments. Aliasing [online], 2008. [cit. 2017-01-4], ⟨http://zone.ni.com/images/reference/en-XX/help/370051M-01/aliasing_effects.gif⟩.

[4] STMicroelectronics. AN3116 Reference Manual [online]. [cit. 2017-01-4], ⟨http://www.st.com/content/ccc/resource/technical/document/application_note/c4/63/a9/f4/ae/f2/48/5d/CD00258017.pdf/files/CD00258017.pdf/jcr:content/translations/en.CD00258017.pdf⟩, 2010.

[5] STMicroelectronics. DS10362 [online]. [cit. 2017-01-4], ⟨http://www.st.com/content/ccc/resource/technical/document/datasheet/2c/6f/d7/64/1f/a3/4f/c9/DM00118585.pdf/files/DM00118585.pdf/jcr:content/translations/en.DM00118585.pdf⟩, 2016.

[6] STMicroelectronics. RM0316 Reference Manual [online]. [cit. 2017-01-4], ⟨http://www.st.com/content/ccc/resource/technical/document/reference_manual/4a/19/6e/18/9d/92/43/32/DM00043574.pdf/files/DM00043574.pdf/jcr:content/translations/en.DM00043574.pdf⟩, 2016.

[7] STMicroelectronics. AN2834 Application note [online]. [cit. 2017-01-4], ⟨http://www.st.com/content/ccc/resource/technical/document/application_note/group0/3f/4c/a4/82/bd/63/4e/92/CD00211314/files/CD00211314.pdf/jcr:content/translations/en.CD00211314.pdf⟩, 2017.

[8] STMicroelectronics. DS10147 [online]. [cit. 2017-01-4], ⟨http://www.st.com/content/ccc/resource/technical/document/datasheet/52/ad/d0/80/e6/be/40/ad/DM00105814.pdf/files/DM00105814.pdf/jcr:content/translations/en.DM00105814.pdf⟩, 2017.

[9] STMicroelectronics. RM0091 Reference Manual [online]. [cit. 2017-01-4], ⟨http://www.st.com/content/ccc/resource/technical/document/reference_manual/c2/f8/8a/f2/18/e6/43/96/DM00031936.pdf/files/DM00031936.pdf/jcr:content/translations/en.DM00031936.pdf⟩, 2017.

[10] Yiu, J. Definitive Guide to ARM Cortex - M3 and Cortex - M4 Processors [online]. [cit. 2017-01-4], ⟨https://www.eecs.umich.edu/courses/eecs373/labs/refs/M3%20Guide.pdf⟩, 2007.

Příloha A

Obsah přiloženého CD

- Diplomová práce ve formátu PDF