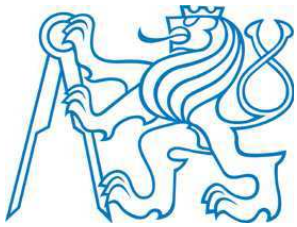
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA ELEKTROTECHNICKÁ



Diplomová práce

Realizace přístrojových funkcí periferními bloky mikrořadičů

Praha, 2013 . Autor: Bc.Pavel Novotný

Anotace

Tato diplomová práce se zabývá návrhem a realizací typických přístrojových funkcí s použitím periferií mikrokontroléru z rodiny STM32Fxx, především mikrokontrolérů s jádrem ARM Cortex-M0. Byly vytvořeny vzorová řešení a programy měřících přístrojů pro měření napětí, frekvence, periody, střídy, přístroj pro záznam průběhu signálu, DDS funkční generátor, impulsní generátor. Byly vytvořeny dvě aplikace s využitím spolupráce více mikrokontrolérů v jednom přístroji.

Annotation

This thesis deals with the design and realization of typical instument functions using peripheries of a microcontroller from STM32Fxx family, especially microcontrollers with ARM Cortex-M0 kernel. Pattern solutions, programs of measuring instruments for the measurement of electrical voltage, frequency, period and duty cycle, an instrument for recording the waveform of a signal, a DDS function generator and a pulse generator have been created. Two applications using the cooperation of more microcontrollers in one instrument have been made as well.

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická
Katedra měření

Akademický rok **2012-2013**

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	**Bc. Pavel Novotný**
Studijní program:	**Kybernetika a robotika**
Obor:	**Senzory a přístrojová technika**
Název tématu česky:	**Realizace přístrojových funkcí periferními bloky mikrořadičů**
Název tématu anglicky:	**Realization of Instrument Functions Using Microcontroller Peripherals**

Pokyny pro vypracování:

Analyzujte možnosti realizace typických přístrojových funkcí pomocí periferních bloků mikrořadičů. Navrhněte způsoby řešení jednotlivých funkcí použitím periferií mikrořadičů z rodiny STM32Fxxx, především pak STM32F05x s jádrem ARM Cortex-M0. Vytvořte vzorová řešení a příslušné programy pro využití STM32F05x v daných přístrojových činnostech (např. jako čítač pro měření frekvence, periody, střídy, počtu impulsů, voltmetr, impulsní generátor, funkční generátor, ...) a prakticky je ověřte. Posuďte výhody i omezení navrženého řešení oproti klasickému řešení se specializovanými obvody (čítači, generátory impulsů, převodníky A/D a D/A, převodníky střední a efektivní hodnoty, ...). Posuďte možnost řešení spolupráce více mikrořadičů realizujících jednotlivé funkce ve společném přístroji.

Seznam odborné literatury:

[1] Yiu, J.: The Definitive Guide to the ARM Cortex-M0, ISBN: 978-0-12-385477-3, Elsevier 2011
[2] STMicroelectronics: STM32F050x4 STM32F050x6 -Data sheet, Doc ID 023079 Rev 3, 2012
[3] STMicroelectronics: STM32F05x - Reference Manual, RM0090, doc. ID 018940 Rev 2, 2012

Vedoucí diplomové práce:	doc. Ing. Jan Fischer, CSc.
Datum zadání diplomové práce:	24. listopadu 2012
Platnost zadání do[1]:	30. června 2014

Prof. Ing. Vladimír Haasz, CSc.
vedoucí katedry

L.S.

Prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 24. 11. 2012

[1] Platnost zadání je omezena na dobu tří následujících semestrů.

Čestné prohlášení autora práce

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 autora práce

Rád bych tímto poděkoval Doc. Ing. Janu Fischerovi, CSc. za jeho trpělivost, vstřícnost, vedení a pomoc při tvorbě této diplomové práce. Dále bych chtěl poděkovat všem lidem, kteří mě vyučovali na ČVUT FEL, za to, že mi pomohli objevit mnohé krásy oboru elektroniky.

Rád bych poděkoval mé manželce, dětem, rodičům, bratrovi, tchyni za to, že vydrželi mou neustálou zaneprázdněnost.

Poděkování patří sestře a švagrovi za poskytnuté přístřeší během častých cest do Prahy.

Také bych rád poděkoval kolegům z dálkového studia, především Michalu Nečasovi za to, že mě motivoval k pokračování v magisterském studiu.

Nakonec bych rád poděkoval svému zaměstnavateli, za umožnění studia a za podporu během studia.

OBSAH

1. Úvod

V technické praxi se při vývoji, testování, opravách nebo při provozu elektrických a elektronických zařízení, neobejdeme bez potřeby měření základních charakteristik elektrických veličin. Na druhé straně, mnohdy nastává potřeba generování nejrůznějších řídících nebo testovacích signálů pro buzení elektronických obvodů.

Řešení těchto požadavků s sebou zpravidla přináší nákup specializovaných měřících přístrojů a zařízení, abychom je v případě potřeby mohli mít k dispozici. Pořízení takové techniky, vedle cenových nákladů, rovněž přináší nároky na její skladování a šetrné zacházení. To může být v případě nízkého využití přístroje značně neefektivní.
V případě pevného zabudování přístroje, hraje významnou roli cena a specifické požadavky na zařízení, které je někdy velmi obtížné, až nemožné splnit standardními přístroji, a nebo pouze za vyšší cenu zákaznického řešení.

V praxi se dnes naprosto běžně setkáváme například s poměrně levnými měřícími zařízeními typu osciloskopů, jak ve verzích modulu pro počítače, tak i klasických samostatných přístrojů, kde je možné doplněním software přístroj rozšířit o další funkce, které přístroj v okamžiku nákupu neměl. Jedná se tedy spíše o univerzální měřící přístroj, který je možné použít na více úloh v oboru měřící techniky. Na tomto příkladu je patrný vývoj techniky směrem k univerzálnímu přístroji, se schopností změny funkce pouhým přehráním programu.

Pro uspokojení výše uvedených požadavků se nabízí možnost převzetí potřebných přístrojových funkcí vývojem vlastního levného univerzálního přístroje, pracujícího s využitím vlastností programovatelného obvodu. Takovým obvodem může být například hradlové pole FPGA, které vyniká především rychlostí provádění operací programu. Tyto obvody však nejsou v praxi tak vysoce rozšířeny a populární tak jako mikrokontroléry, které nabízejí stále vyšší rychlosti zpracování instrukcí programu, vyšší hodnotu paměti, bohatší nabídku vlastností periferií při nízkém odběru energie, důležitém při bateriovém napájení přístroje.

Právě zmíněná variabilita řešení s mikrokontroléry, jejich vysoký výkon, dobrá podpora výrobců, cenová dostupnost, množství variant, to jsou vlastnosti, které dávají vývojářům do rukou nástroj pro realizaci levného zařízení s vynaložením nízkých nákladů na vývoj.

Cílem této diplomové práce je návrh a posouzení řešení měřících funkcí pomocí levného mikrokontroléru, s praktickým odzkoušením těchto zařízení :

- čítač pro měření frekvence a periody
- měření délky pulsu a střídy signálu
- voltmetr s měřením TRMS[1]
- funkční generátor, impulsní generátor
- datalogger
- odzkoušení spolupráce více zařízení v jednom přístroji

[1] True Root Mean Square. Měření skutečné efektivní hodnoty průběhu měřené veličiny.

2. Rozbor řešení úkolů

Stav současné techniky umožňuje díky zvyšování výkonu mikrokontrolérů za současného poklesu jejich ceny, nahrazovat mnohé měřící přístroje pomocí spojení výkonného procesoru a bohatých periferií vestavěných v těchto obvodech.

Jednou ze špičkových vývojových firem procesorů, je společnost ARM Limited, která je v současné době dodavatelem licencí procesorů s jádrem ARM. Dále existuje velká skupina firem, které doplní jádro procesoru o specifické periferie. Firma STMicroelectronic je jednou z této skupiny výrobců mikrokontrolérů. Tato firma nabízí pro své zákazníky velmi zdařilé a cenově dostupné vývojové kity s procesory STM32. Dalším důležitým atributem pro vývoj aplikace s mikrokontroléry, je dostupné vývojové prostředí. Takovým prostředím může být např. vývojové prostředí uVision fy. Keil, které je k dispozici v omezené eval verzi zdarma. Omezení tohoto software spočívá v maximální velikosti kódu do 32 kB a v nekomerčním použití vyvíjeného software aplikace.

2.1 Základní příklady práce s procesorem STM32F0

Vzhledem k nulové počáteční znalosti procesorů ARM, s pouze minimálními zkušenostmi s 8bitovými procesory, bylo nutné seznámení se základními periferiemi mikrokontroléru. V rámci této činnosti jsem připravil programy a dokumentaci pro budoucí studenty, kteří mohou tyto příklady použít pro první seznámení s mikrokontrolérem ARM. Programy v jazyce assembler a jazyce C jsou nahrány ve složce „Zakladni_priklady_STM32F0" na přiloženém CD. První čtyři programy (blikani, tlacitko, USART, Timer) jsem upravil z dodaných kódů vedoucím diplomové práce. Dokumentaci a ostatní programy jsem vypracoval celé samostatně.

2.2 Návrh metod a přístrojů s MCU[2]

V následujících podkapitolách budou uvedeny návrhy a uvažované metody řešení přístrojových funkcí pomocí mikrokontroléru. Metody návrhu jsou uvažovány pro obecný mikrokontrolér, pro konečnou realizaci přístrojů bude použit levný MCU STM32F051R8.

2.2.1 Metody měření napětí, voltmetr

Jedním ze základních údajů charakterizujících stav elektrického obvodu je velikost elektrického napětí.

Pro měření elektrického napětí se často používá mikrokontrolér v režimu měření napětí metodou dvojí integrace. Metoda je založena na integraci neznámého napětí, poté následuje integrace referenčního napětí opačné polarity na počáteční úroveň napětí. Z poměru časů obou integrací lze získat velikost střední elektrolytické hodnoty. Tento princip bývá použit v jednodušších měřících přístrojích, má výhodu v přesnosti a v potlačení sériového rušení.

V případě, že budeme chtít měřit časově proměnné napětí, je vhodnější metodou použít ADC převodník mikrokontroléru, signál v dostatečné délce navzorkovat a průběžně provádět výpočet z těchto dat.

[2] MCU – Microprocessor unit, mikroprocesor

Výhodou tohoto způsobu měření je jednoduchý levný měřící systém s možností využití dalších periferií obvodu, např. pro zobrazování měřené hodnoty, nebo možností komunikace s dalšími zařízeními.

Pomocí mikrokontroléru budeme schopni měřit základní hodnoty napětí, maximální, střední, efektivní hodnoty s rozlišením vstupního napětí ADC převodníku.

2.2.2 Klasické metody měření frekvence, periody

Pro měření frekvence průběhu elektrického napětí, první metodou, kterou můžeme použít je využití osciloskopu. Frekvenci signálu určíme na základě odečtení délky periody signálu. Tato metoda je málo přesná, ale může dobře posloužit pro orientační měření frekvence signálu, především proto, že dokáže podat informaci o tvaru měřeného signálu. Tato informace je důležitá pro správné vyhodnocení frekvence měřeného signálu v případech rušených signálů.

Za klasickou metodu číslicového měření frekvence signálu se považuje použití čítače v hradlovaném režimu[3]. Doba hradlování je řízena oscilátorem s vhodně určenou frekvencí. Vstupem čítače je měřený průběh signálu. Tuto metodu pro měření frekvence lze velmi dobře řešit pomocí mikrokontroléru.

2.2.3 Klasické metody měření délky periody, střídy

Měření délky periody signálu je klasickou metodou nepřímého měření frekvence. Řeší se s pomocí časovače, jehož činnost je povolena po dobu periody měřeného signálu. Z počtu načtených impulsů časovačem, lze potom určit dobu periody měřeného signálu. Hlavním úkolem metody je správné rozlišení začátku a konce periody. Pokud navíc rozlišíme kladnou část periody od záporné, dokážeme měřit i střídu signálu. Toto rozlišení bude možné provést s pomocí jednotek capture (v překladu „zachycení“) časovačů mikrokontroléru, které mají možnost spuštění a tím zachycení stavu registru načtených pulsů čítače na sestupnou nebo náběžnou hranu signálu. Pro měření signálu, který nemá obdélníkový průběh, bude nutné použít komparátor na vstupu měřícího zařízení.

2.2.4 Impulsní generátor, PWM

Jedním z přístrojů, který se používá v elektrotechnice, především pro testovací účely, je impulsní generátor. Jak již vyplývá z názvu, impulsní generátor je zařízení, které vyrábí zpravidla obdélníkové impulsy o nastavitelné šířce s nastavitelnou dobou opakování, tedy frekvencí. Jedním z parametrů impulsního generátoru je minimální šířka pulsu, kterou je schopen generátor vytvořit. Impulsní generátor lze použít také např. v oblasti zdravotnictví, v oboru magnetoterapie, pro vytváření pulsního magnetického pole buzením cívek viz článek [26].

Tento přístroj by bylo možné realizovat pomocí časovačů mikrokontroléru, v případě STM32F051, s využitím jeho 32bitového časovače TIM2. Pomocí tohoto časovače bychom mohli dosáhnout šířky pulsu 20 ns, s rozsahem periody opakování od 40 ns do 89 s.

[3] Hradlovaný režim – povolení činnosti čítače pouze po dobu trvání logického signálu na k tomu určeného vstupu čítače.

Jako nejjednodušší způsob řešení bude použití časovače v režimu PWM, s nastavitelnými parametry délky periody a délky impulsu. Pro snadnější nastavování přístroje budou k navrženému generátoru připojeny tlačítka, kterými bude možné nastavení délky periody a délky pulsu. Zobrazení nastavitelných hodnot bude provedeno pomocí dvouřádkového LCD displeje.

2.2.5 Měření polohy-enkodér

Pro měření polohy, kde dochází k převodu rotačního pohybu na přímý pohyb, jsou v průmyslu téměř výhradně používány enkodéry. Mikrokontrolér STM32F051 nabízí podporu enkodéru, navíc obsahuje 32bitový čítač a proto nebude nutné použít kaskádové zapojení standardních 16bitových časovačů, kde vznikaly problémy pokud enkodér osciloval kolem bodu přetečení čítače.

Pro kontrolu počtu pulsů se pokusím pomocí C stopy enkodéru detekovat správný počet načtených pulsů, abych měl jistotu, že enkodér pracuje správně a nedošlo ke ztrátě nebo nesprávného načtení impulsů.

2.2.6 Funkční generátor

Funkční generátor by mělo být zařízení, které bude schopné generovat základní periodické průběhy signálu jako je sinusovka, pila, obdélník.

Mikrokontrolér STM32F051, mimo jiné , obsahuje periferii DAC (digitálně anologový převodník) a dále několik jednotek časovačů. Pokud by časovač spouštěl DMA přenos v paměti uloženého průběhu na periferii DAC bylo by možné tyto signály generovat. Časovačem by bylo možné měnit frekvenci výstupního signálu. Tato varianta má bohužel omezené možnosti spočívající zejména v nízkém rozsahu přeladitelnosti frekvence generátoru.

Další možností realizace generátoru je využití klasické metody generování signálu v číslicových přístrojích založené na principu DDS (Direct digital synthesis). Touto metodou bychom měli dosáhnout lepších výsledků, zejména přeladitelnosti generátoru, nevýhodou bude vynechávání některých vzorků ve výsledném průběhu. Nicméně, tato metoda je v praxi s různými vylepšeními stále velmi používaná.

2.2.7 Přístroj pro záznam dat, datalogger

Přístroj pro záznam měřených dat označujeme jako datalogger. Je to zařízení, které má za úkol v určitých ekvidistantních okamžicích odebírat vzorky pozorované veličiny. Předmětem sběru dat může být nějaká fyzikální veličina, např. napětí, teplota, vlhkost apod.. Tyto veličiny jsou převedeny na elektrickou veličinu např. proud, napětí a dále vzorkovány dataloggerem. Vzorky jsou průběžně ukládány do paměti, z které jsou zpravidla po ukončení sledování přeneseny pro další zpracování do počítače. Perioda vzorkování dataloggeru bývá, v případě pomalu měnících se veličin jako zmiňovaná teplota a vlhkost, zpravidla v jednotkách minut. Nic ale nebrání periodu vzorkování mít i velmi krátkou, např. v jednotkách mikrosekund a použít přístroj podobným způsobem jako osciloskop. Pro zobrazení dat bude použit software na PC. Přístroj by měl být schopný data odeslat na PC např. sériovou linkou RS232.

2.2.8 Spolupráce mikroprocesorů

V některých případech technické praxe může být a je užitečná spolupráce více procesorů. Důvodem je například rozdělení úkolů na více procesorů pro zvýšení výkonu zařízení. Nebo může nastat situace, kdy jsou jednotlivé senzory umístněny na rozlehlém prostoru a je potřeba data posbírat, odeslat a zpracovat v jednom centrálním místě. Typickou aplikací jsou např. monitorovací systémy.

V současné době řeším v zaměstnání problém monitorování provozu lisovny plastů, pokusím se tedy prozkoumat možnosti řešení sběru dat do PC pomocí jednotek s mikrokontroléry STM32F051. Dále otestuji komunikaci čistě mezi procesory STM32F051 na příkladu sběru dat ze dvou enkodérů a zobrazením jejich stavu na LCD displeji. V tomto případě, bude mít každé zařízení vlastní MCU, pomocí kterého bude připojeno do sítě.

Požadavek řešení komunikace mezi zařízeními přináší několik úkolů. Mezi první patří řešení struktury a řízení sběrnice systému, zda bude síť řešena způsobem slave[4]/master[5] nebo způsobem multi[6]-slave/master, případně kombinace s multi-master.

Pro sběr dat k účelu vizualizace se výborně hodí struktura řízení sběrnice multi-slave/master. Na sběrnici bude tedy jedna jednotka master, která bude řešit komunikaci způsobem, který se nazývá pooling. Jedná se postupné oslovování připojených stanic typu slave jednotkou master s výzvou na odeslání odpovědi na dotaz.

Dalším úkolem je volba přenosového protokolu. U těchto typů zařízení bývá zpravidla jednoduchý protokol, který je firmou použit pouze pro tento účel. Vyvíjení vlastního přenosového protokolu bych se chtěl vyhnout z důvodu omezení přístroje použitím speciálního řešení. Celkem dobrou možností by mohl být protokol SCPI používaný v měřící technice. Nicméně v případě řešení monitorování strojů by mohl pro tento účel lépe vyhovovat také starší protokol MODBUS.

Použití protokolu MODBUS bude výhodné z důvodu možnosti komunikace zařízení se software od jiných dodavatelů, např. bude možné využití prostředí ControlWeb výrobce Moravské přístroje a.s. [25] určené pro sledování výrobních procesů.

Posledním problémem je vyřešení fyzické vrstvy sítě, kde vzhledem k nasazení modulů v průmyslovém prostředí a vzhledem k uvažovaným vzdálenostem cca 300 metrů, by měla být použita linka RS485. Tato linka je jednoduchá, levná a odolná proti rušení, nevyžaduje žádné speciální nároky na řízení.

2.3 Cíle DP

Cílem této diplomové práce bude návrh a realizace funkčních vzorků měřících přístrojů s využitím procesoru STM32F051. U těchto měřících přístrojů budou změřeny jejich základní parametry. Vybrané parametry přístrojů budou, pokud to bude možné, porovnány s obvody realizující měřící funkce bez využití mikrokontroléru.

[4] Slave – v tomto kontextu se jedná o podřízenou jednotku, která neinicializuje komunikaci, zpravidla odpovídá na dotazy jednotky Master.
[5] Master – inicializuje komunikaci, řídí sběrnici systému.
[6] Multi – vícenásobný.

Na konci diplomové práce bude realizován systém s více spolupracujícími mikrokontroléry a na tomto systému otestován způsob multiprocesorové komunikace.

3. Periferie mikrokontroléru v příkladech

Tato kapitola je určena pro budoucí studenty předmětů zabývající se prací s procesory ARM, zvláště s procesorem STM32F0. Materiál vznikl při počátečním seznámení s tímto procesorem. K příkladům jsou vypracovány ukázkové programy v jazyce C a assembler.

3.1 Příklad 1 „Blikani“ v assembleru

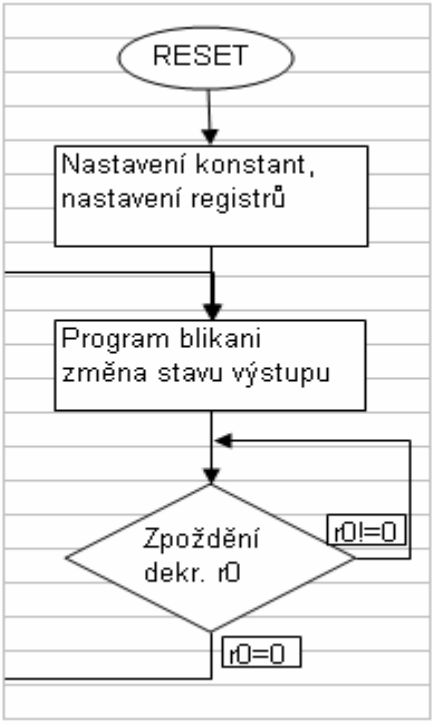
Funkce programu

Program byl vytvořen pro demonstraci nastavení periferií hodin a výstupního portu input/output mikrokontroléru s procesorem STM32F051R8. Jako hardware byl použit STM32F0-Discovery kit.

Vývojový diagram programu

Diagram programu viz obr. 3.1.1 začíná resetem mikrokontroléru, pokračuje definicí a naplněním konstant použitých pro nastavení obsahu registrů mikroprocesoru.

Po inicializaci registrů RCC a GPIO, přejde program do části blikání (označeno návěstím „blikani:“), kde při každém průchodu touto částí, změní stav výstupu pinu 9 portu C na opačnou úroveň. U modulu STM32F0-Discovery je na tomto portu připojena LED dioda LD3.



Obr.3.1.1 Vývojový diagram programu.

Definice adres, masek registrů, konstant

Jak již bylo zmíněno, na začátku programu je vhodné připravit konstanty pro naplnění požadovaných hodnot do registrů mikroprocesoru. Níže uvedený kód připravuje nastavení

hodin jádra a periferií. Ty se později zapíší do obsahu registru RCC_CR viz obr. 3.1.2, zdroj [1], str. 94 a registru RCC_CFGR obr. 3.1.3, [1], str. 96. Tyto hodnoty tedy určují, jak bude nastaven celý blok hodin procesoru a periferií [1], str. 83.

```
PPRE    EQU 0x0          ; APB divide 1
HPRE    EQU 0x0          ; AHB divide 1
SW      EQU 0x2          ; Přepínač PLL zdroj hodin pro sysclk
PLLSRC  EQU 0x0          ;Přepínač  hsi/2 zdroj hodin pro pll
PLLMUL EQU 0x280000      ; Násobička PLL (hsi/2)*12
```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	PLL RDY	PLLON	Res	Res	Res	Res	CSS ON	HSE BYP	HSE RDY	HSE ON
						r	rw					rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSICAL[7:0]								HSITRIM[4:0]					Res	HSI RDY	HSION
r	r	r	r	r	r	r	r	rw	rw	rw	rw	rw		r	rw

Obr. 3.1.2 Struktura registru RCC_CR.

Nastaveni konstant pro změnu náplně registru RCC_CFGR

```
HSION    EQU 0x1          ; maska pro HSION
HSIRDY   EQU 0x2          ; maska pro HSIRDY
PLLON    EQU 0x1000000    ; maska pro PLLON
PLLRDY   EQU 0x2000000    ; maska pro PLLRDY
```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	MCO[2:0]			Res	Res	PLLMUL[3:0]				PLL XTPRE	PLL SRC
					rw	rw	rw			rw	rw	rw	rw	rw	rw

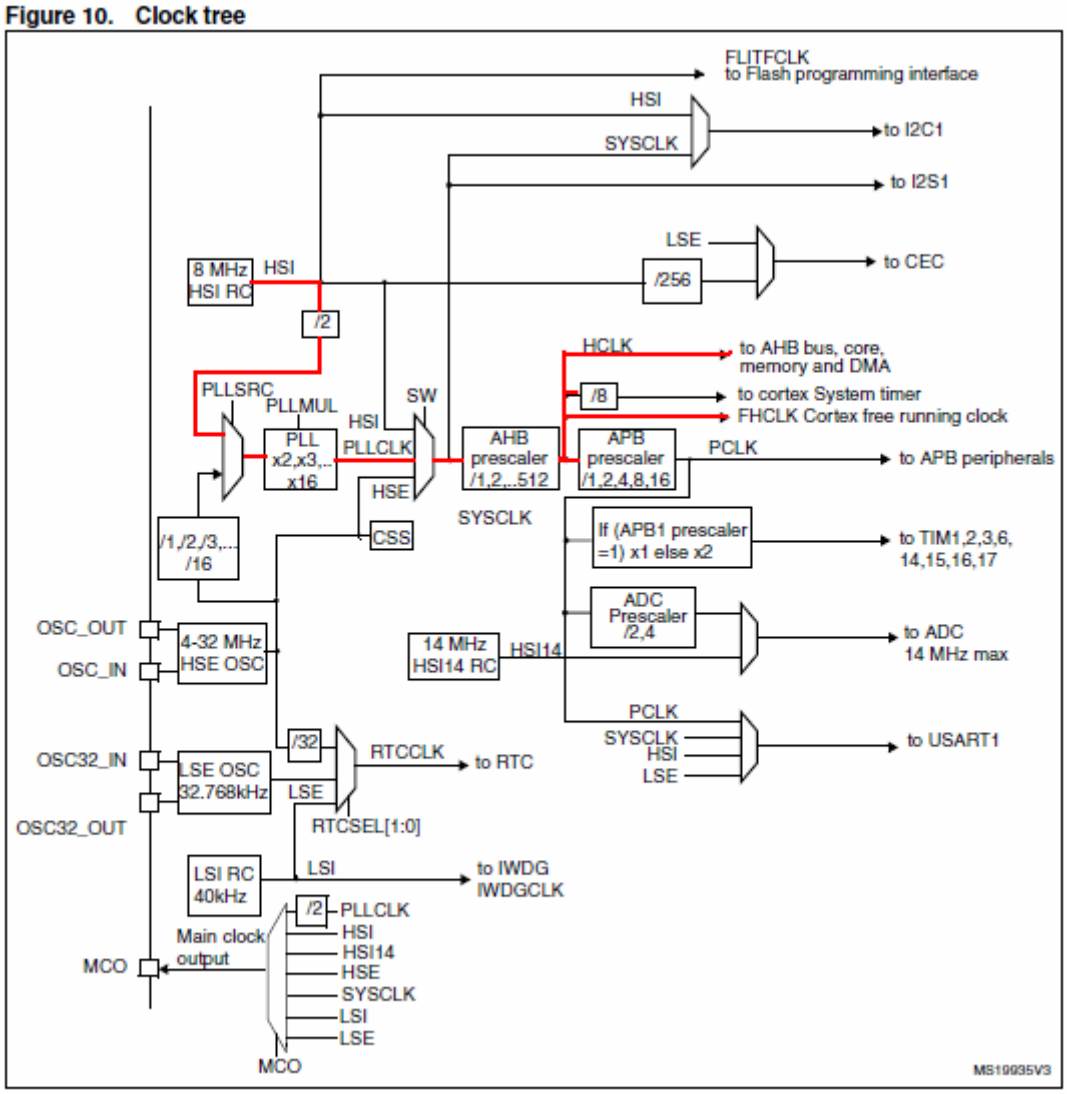
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	ADCPRE	Res	Res	Res	PPRE[2:0]			HPRE[3:0]				SWS[1:0]		SW[1:0]	
	rw				rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw

Obr. 3.1.3 Struktura registru RCC_CFGR.

Výpočet nastavení bloku hodin :

1. Použit interní generátor HSI, který dodává pomocí RC oscilátoru hodinový signál o frekvenci 8MHz.
2. Dělení pomocí pevné děličky dvěmi, blok /2.
3. Nastaven přepínač PLLSR= 0: HSI/2 selected as PLL input clock.
4. Násobička PLLMUL=0x0A : (1010) tzn. násobit vstup x12.
5. Přepínač SW= 0x02 : zdroj je PLLCLK.
6. Dělička AHB=0 : bez dělení (v registru jako HPRE: HLCK prescaler).

Výsledkem předchozích operací je frekvence HCLK pro jádro MCU **48 MHz**. Na obr. 4 je toto nastavení zachyceno pomocí červených čar, představujících nastavení hodin pro jádro a periferie procesoru.



1. For full details about the internal and external clock source characteristics, please refer to the "Electrical characteristics" section in your device datasheet.

Obr. 3.1.4 Schéma struktury hodin procesoru STM32F0xx.

Nastavení bázových adres

Konkrétní umístnění bázových adres registrů se dozvíme z mapy paměti, např. z dokumentu [1], str. 36. Tyto adresy je vhodné pro přehlednost zapsat níže uvedenou formou a poté do těchto adres, zvětšených o offset, určující adresu registru, ukládat hodnoty nastavující funkci periferií.

```
; nastaveni adres periferii v paměti
RCC_base      EQU 0x40021000
FLASH_base    EQU 0x40022000          ; adresa flash_acr registr
GPIOA_base    EQU 0x48000000
```

```
GPIOB_base    EQU 0x48000400
GPIOC_base    EQU 0x48000800
GPIOD_base    EQU 0x48000C00
TIM1_base     EQU 0x40012C00
SPI1_base     EQU 0x40013000
USART1_base   EQU 0x40013800
```

Nastavení periferie GPIO (General periphery input output) a její uvolnění.

Pro nastavení periferie GPIO jsou v programu předpřipraveny konstanty GPIOCEN a GPIOC_mode. Hodnota GPIOCEN je použita pro nastavení registru RCC_AHBENR, popis v [1], str. 104. Tento registr povoluje hodinový signál pro periferie na sběrnici AHB.

Druhá hodnota GPIOC_mode, bude použita pro nastavení registru GPIO_MODER, popis v [1], str. 129. Registr konfiguruje funkci pinů mikrokontroléru. Na obr. 3.1.5 je zobrazena struktura tohoto registru a jeho možného nastavení. V tomto příkladu je k pinu 9 portu C připojena LED dioda, proto bude hodnota MODER9 nastavena na 0x01 tj. do režimu výstupu.

```
GPIOCEN      EQU 0x80000      ; maska pro GPIOCEN
GPIOC_mode   EQU 0x40000      ; 01- general purpose output mode
```

8.4.1 GPIO port mode register (GPIOx_MODER) (x = A..D, F)

Address offset: 0x00

Reset values:

- 0x2800 0000 for port A
- 0x0000 0000 for other ports

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 2y+1:2y **MODERy[1:0]:** Port x configuration bits (y = 0..15)
These bits are written by software to configure the I/O mode.
00: Input mode (reset state)
01: General purpose output mode
10: Alternate function mode
11: Analog mode

Obr. 3.1.5 Registr GPIOx_MODER.

Program - nastavení mikrokontroléru.

V následujícím textu se pokusím vysvětlit funkci kódu použitého v tomto příkladu. Pod textem je zpravidla umístněn kód v jazyce assembler použitý pro realizaci popisované funkce.

Klíčové slovo „EXPORT“ na začátku programového bloku označuje funkci jako veřejnou a je ji tedy možné volat z ostatních modulů. Zde je funkce volána z modulu „startup_stm32f0xx.s“, který nám prostředí uVision4 připojilo při vytvoření projektu. Tento modul zajišťuje počáteční nastavení procesoru, konkrétní funkce jsou vypsány v komentáři na začátku tohoto modulu.

Následující klíčové slovo „ENTRY“, označujíce začátek programu. Na dalším řádku programu nalezneme následující text „AREA |.text|, CODE, READONLY“, kde místo části

„|.text|“ můžeme vložit vlastni název bloku, slovo CODE a READONLY určuje umístnění bloku v paměti.

```
EXPORT    __main
__main

ENTRY
```

Níže uvedeným kódem nastavíme v registru FLASH_ACR zpoždění přístupu do paměti FLASH o jeden čekací cykl, toto nastavení je nutné pokud pro frekvenci SYSCLK platí podmínka 24 MHz<SYSCLK=<48 MHz zdroj [1], str. 53.

Adresu registru FLASH_ACR uložíme instrukcí LDR do registru r1, poté hodnotu z paměťového místa danou obsahem registru r1 přesuneme do registru r0. Instrukcí ORRS k registru r0 přičteme číslo 0x1 uložené v registru r2 a výsledek operace uložíme instrukcí STR na adresu v registru r1 tj. FLASH_ACR registru.

```
; flash init - nastaveni latence na jeden čekací cyklus
LDR r1,=FLASH_base          ; nacteni hodnoty FLASH_base do R1
LDR r0, [r1]                ; nacteni hodnoty z adresy FLASH_base do R0
MOVS r2,#0x1                ; ulozeni 1 do R2
ORRS r0,r2                  ; logicky OR R0 a 1, vysledek v R0
STR r0,[r1]                 ; ulozeni R0 zpet na adresu  FLASH_base
```

V další části programu provedeme nastavení registrů RCC zajišťujících správu hodinového signálu procesoru.

Nejprve uložíme do registru r1 bázovou adresu registru RCC. Prostřednictvím registrů r0 a r2 provedeme postupně logický součet „OR“ dílčích požadovaných nastavení. Opakováním těchto součtů docílíme požadovaného nastavení obsahu registru. Výsledek nakonec uložíme instrukcí STR na adresu registru RCC_base + 4 byte(offset) tj. do registru RCC_CFGR viz obr. 3.1.3.

```
;rcc init
LDR r1,=RCC_base            ; vytvoreni obsahu pro RCC_CFGR
LDR r0,=PPRE
LDR r2,=HPRE
ORRS r0,r                   ; vysledek se ulozi do r0
LDR r2,=SW
ORRS r0,r2
LDR r2,=PLLSRC
ORRS r0,r2
LDR r2,=PLLMUL
ORRS r0,r2
STR r0,[r1,#0x4]            ; ulozime obsah do RCC_CFGR
```

V následujícím kroku provedeme zapnutí vnitřního oscilátoru HSI.

Přímo na adrese RCC_base je umístněn první registr periferie RCC a tím je Clock control registr označený RCC_CR. V tomto registru nastavením bitu 0 do stavu log. 1 viz obr. 3.1.2 zapneme vnitřní RC generátor označený HSI.

```
;vytvoreni obsahu pro RCC_CR
    LDR r0,[r1]          ;  nacteni hodnoty na adrese RCC_base do r0
    LDR r2,=HSION        ;  nacteni hodnoty pro zapnuti HSI do r2
    ORRS r0,r2           ; logicky OR R0 a R2
    STR r0,[r1]          ;  ulozeni r0 na adresu RCC_base a je zapnuto
                            HSI
```

V tomto kroku se provádí čekání do doby, než bude vnitřní generátor stabilní.

```
hsi_rdy                  ; kontrola zda je HSI stabilni
    LDR r0,[r1]          ; nacten obsah RCC_base
    LDR r2,=HSIRDY
    ANDS r0,r2           ; pronasobeni maskou HSIRDY
    BEQ hsi_rdy          ; jestliže je HSI stabilni pokracuje se jinak je
                            proveden skok na hsi_rdy
```

Nakonec v registru RCC_CR zapneme násobičku PLL, tím je nastavení hodin kompletní.

```
    LDR r2,=PLLON        ; nacteni masky PLLON
    ORRS r0,r2
    STR r0,[r1]          ; ulozeni R0 do RCC_base. zapnuti PLL
```

V následujícím kódu provedeme nastavení výstupní části programu, tedy konfiguraci periferie General purpose input output GPIO. Pro činnost periferie mikrokontroléru je vždy nutné přivést hodinový signál, ten je po inicializaci obvodu pro periferie defaultně vypnutý z důvodu úspory energie celého obvodu.

```
    LDR r0,[r1,#0x14]    ; nacteni registru RCC_AHBENR
    LDR r2,=GPIOCEN      ; nacteni povolovaci masky
    ORRS r0,r2
    STR r0,[r1,#0x14]    ; povoleni hodin pro GPIOC, v registru
                            RCC_AHBENR
```

Po povolení periferie GPIOC a hodin do této periferie definujeme funkci pinu PC9 portu C jako výstup(bude na něj připojena LED). Ostatní nastavení registru ponecháme na defaultních hodnotách, tzn. funkce pinu vstup, nepřipojené pullUp/Down rezistory, obnovovací rychlost pinu 2 MHz.

```
    LDR r1,=GPIOC_base   ; nacteni hodnoty GPIOC_base
    LDR r0,[r1]          ; nacteni hodnot z GPIOC_base
    LDR r2,=GPIOC_mode   ; nacteni masky GPIOC_mode
    ORRS r0,r2
    STR r0,[r1]          ; ulozeni noveho modu do GPIOC_base
```

Program - výkonná část

Pro výkon smyčky programu předpřipravíme do registrů r4 a r5 hodnoty 0 a 1.

```
MOVS r4,#0x0          ; ulozeni 0 do R4
MOVS r5,#0x1          ; ulozeni 1 fo R5
```

Hlavní smyčka programu je označena návěstím „blikani“. První tři instrukce zajišťují střídání stavu bitu0 v registru r4 a přesun hodnoty do registru r0. Registr r4 slouží jako paměť předchozího stavu. Instrukcí LSLS přesuneme v registru r0 požadovaný stav výstupu na pozici odpovídající pinu 9, následující instrukcí STR přesuneme hodnotu registru r0 do registru GPIOC_ODR, který přímo řídí stav výstupů portu C.

```
blikani
    ADDS r4,#0x1          ; pricteme k hodnote v r4 1
    ANDS r4,r5            ; (R4 AND R5)=R4
    MOVS r0,r4            ; hodnota y R4 presunuta do R0
    LSLS r0,#0x9          ; posun hodnotz v R0 doleva o 9bitu
    STR r0,r1,#0x14]      ; ulozeni hodnoty v R0 do GPIOC_ODR, tj.
                            rozsviceni ci zhasnuti LED na PC9
    LDR r0,=Loop          ; ulozi do r0 hodnotu Loop, ktera urcuje
                            delku  cekani
    BL countdown_loop     ; volani podprogramu cekaci smycky
    B blikani             ; skok na blikaci smycku
```

Abychom viděli blikání diody připojené na pin 9 portu C, prodlužujeme hlavní smyčku skokem do podprogramu na návěstí countdown loop. V tomto podprogramu se pomocí opakovaného vykonávání instrukcí prodlužuje okamžik změny stavu výstupu pinu 9. Při každém průchodu smyčkou podprogramu, se dekrementuje hodnota registru r0, do kterého byla před voláním podprogramu uložena hodnota proměnné Loop. Hodnota konstanty Loop je zadána na počátku programu v odstavci zadávání konstant.

```
countdown_loop        ; zpozdovaci smycka
    SUBS r0,#0x1      ; od hodnoty v r0 odecteme 1
    BNE countdown_loop ; jestlize je v r0 neni nula skaceme na
                         countdown_loop
    BX LR             ; navrat z podprogramu
```

Konec bloku programu se v assembleru označuje direktivou „END“

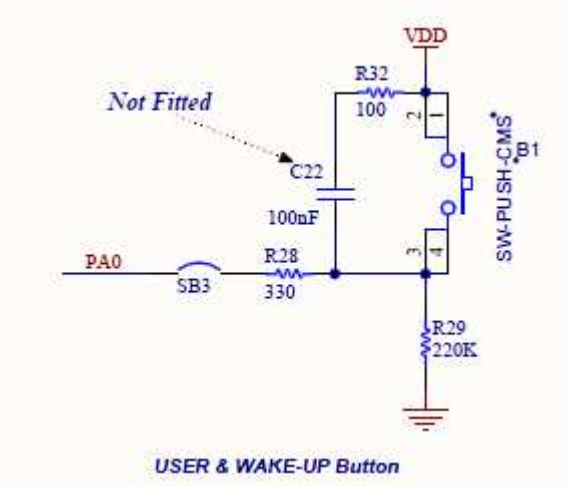
```
    END
```

Podrobný popis instrukcí assembleru je možné nalézt v [6].

3.2 Příklad 2 „Tlačítko“ v ASM

Funkce programu

Program „tlacitko“ byl vytvořen pro demonstraci snímání vstupu vstupně/výstupní periferie procesoru STM32F0xx zabudovaném v demonstračním kitu STM32F0-Discovery. Vstupem aplikace je tlačítko, na kitu označené „USER“, připojené na port A pin 0 viz obr. 3.2.1. Výstupem je LED dioda PC9, připojená k portu C pin 9. Stisk tlačítka způsobí rozsvícení LED diody po dobu stisku tlačítka dioda svítí, pokud tlačítko uvolníme dioda zhasne.



Obr. 3.2.1 Zapojení tlačítka k testovacímu kitu.

Nastavení procesoru a periferií

Nastavení hodin pro procesor, definice konstant pro nastavení adres registrů a jejich hodnot je shodná s příkladem 1 „blikání“. Body 3, 4, 5, 6 příkladu 1 platí i pro příklad 2 „Tlačitko“, nebudou zde tedy tyto nastavení znovu popisována.

Odlišná je až část programu, kde je nutné navíc povolit hodiny také pro port A, kde je připojeno tlačítko z vývojového kitu. Nastavení registru RCC provedeme podobným způsobem jako v případě spuštení hodin pro port C.

```
GPIOAEN EQU 0x20000             ;maska pro GPIOAEN

;GPIOC enable. zapnuti GPIOC a GPIOA
        LDR r0,[r1,#0x14]       ;nacteni registru RCC_AHBENR
        LDR r2,=GPIOCEN         ;nacteni povolovaci masky pro GPIOC
        ORRS r0,r2
        LDR r2,=GPIOAEN         ;nacteni povolovaci masky pro GPIOA
        ORRS r0,r2
        STR r0,[r1,#0x14]       ;povoleni hodin pro GPIOC, v registru
                                 RCC_AHBENR
```

Uložení bázových adres do portů r1, r2 pro další použití v nekonečné smyčce s návěstím „tlacitko“.

```
;GPIOA config
;GPIOA mod je v zakladu nastaven jako vstupni neni treba jiz branu A vice konfigurovat
        LDR r1,=GPIOC_base
        LDR r2,=GPIOA_base
```

Hlavní smyčka programu

```
tlacitko
        LDR r0,[r2,#0x10]     ;nacteny hodnoty GPIOA_IDR registru do regis-
                               tru r0, obsahuje udaje o stavu vstupu brany PA
        MOVS r3,#0x1
        ANDS r0,r3            ;vymaskovani hodnoty pro PA0 kde je umisteno
                               tlacitko
        LSLS r0,#0x9          ;posun hodnotz v R0 doleva o 9bitu
        STR r0,[r1,#0x14]     ;ulozeni hodnoty v R0 do GPIOC_ODR, tj.
                               rozsviceni ci zhasnuti LED na PC9

        B tlacitko            ;skok na začatek smycky
```

V části programu označené návěstím „tlacitko“ nahrajeme instrukcí LDR do registru r0 hodnotu z adresy, která je uložena v registru r2, tj. bázová adresa GPIOA registrů (0x48000000) posunutou o offset 0x10. V registru r0 se tedy bude nacházet hodnota z adresy 0x48000010, kde se nachází registr GPIOA_IDR viz.obr. 3.2.2. Tento registr v sobě udržuje aktuální stav vstupů portu A. Vývojový kit STM32F0-Discovery má na tento port připojeno testovací tlačítko PA0.

Do registru r3 instrukcí „MOVS“ přesuneme hodnotu 0x1h. V dalším kroku obsah registru r0, který obsahuje v bitu 0 informaci o stavu tlačítka vynásobíme instrukcí ANDS hodnotou log.1. V registru r3 bude tedy zachycen pouze stav bitu0 původního registru GPIOA_IDR(stav vstupů portuA), ostatní bity jsou vymaskovány a mají hodnotu 0.

Nyní je potřeba přesunout v registru r0 stav tlačítka uložený v bitu0, na pozici odpovídající připojené LED diodě k pinu 9 portu C tzn. na pozici bitu 9. K tomuto účelu je použita instrukce posunu vlevo LSLS. Takto připravenou hodnotu registru r0 uložíme instrukcí „STR“ do registru, který nastavuje stav výstupů portu A GPIOA_ODR viz. obr. 3.2.3.

Smyčka se uzavírá instrukcí nepodmíněného skoku „B tlacitko“

8.4.5 GPIO port input data register (GPIOx_IDR) (x = A..D, F)

Address offset: 0x10

Reset value: 0x0000 XXXX (where X means undefined)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **IDR[15:0]:** Port input data
These bits are read-only. They contain the input value of the corresponding I/O port.

Obr. 3.2.2 Port GPIOx_IDR, zdroj [1], str. 131.

8.4.6 GPIO port output data register (GPIOx_ODR) (x = A..D, F)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **ODR[15:0]:** Port output data
These bits can be read and written by software.

Note: For atomic bit set/reset, the ODR bits can be individually set and reset by writing to the GPIOx_BSRR register (x = A..D, F).

Obr. 3.2.3 Port GPIOx_ODR, zdroj [1], str. 131.

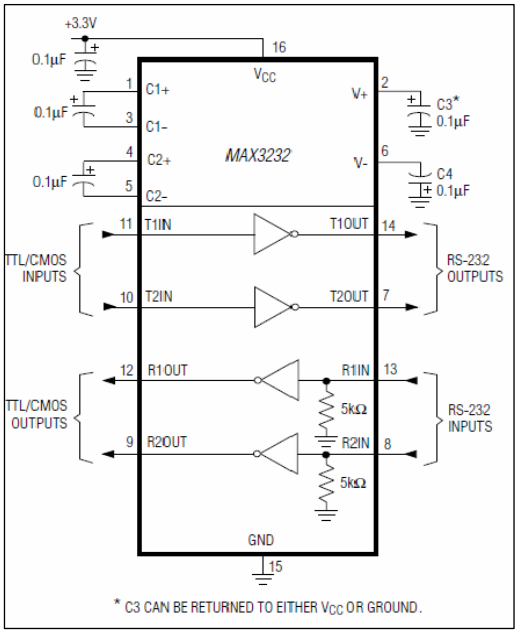
3.3 Příklad 3 „USART“ v ASM

Využití rozhranní USART

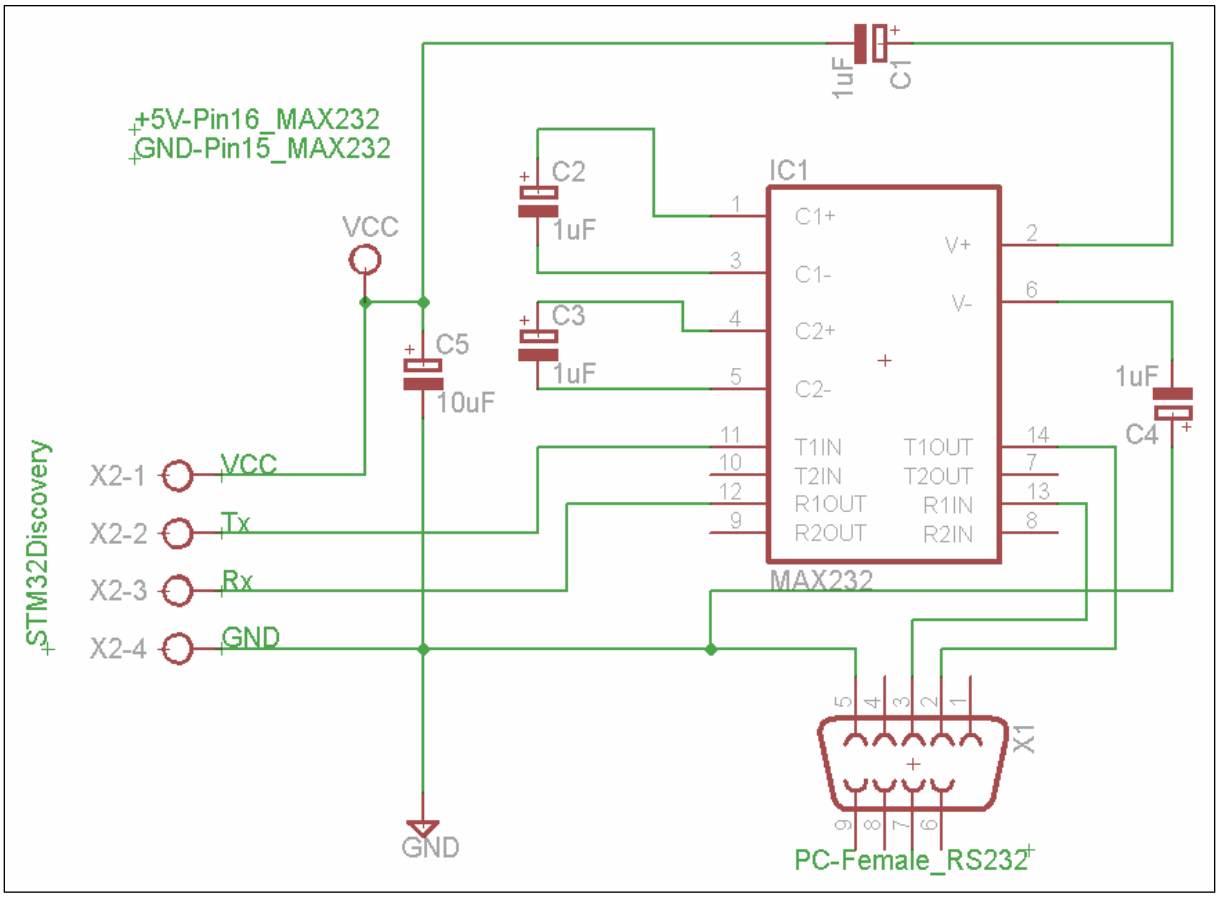
Obvod STM32F051 modulu Discovery má zabudovány dvě univerzální rozhranní s možností asynchronního a synchronního režimu, USART1 a USART2. V tomto příkladu je použito rozhranní s označením USART1. Periferie USART2 nemá všechny vlastnosti rozhranní USART1, viz [1], str. 576. Podrobný výčet vlastností obou rozhranní USART popisuje referenční manuál [1] na str. 574.

Modul Discovery neobsahuje převodník RS232, nutný pro úpravu úrovní CMOS na úrovně TTL, proto je nutné pro komunikaci převodník vyrobit, nebo jej zapojit na nepájivém kontaktním poli. Pro tento účel se používají zpravidla obvody MAX232 s napájením +5 V, nebo obvod MAX3232 s napájením 3 až 5 V. Příklad zapojení převodníku je popsán např. v datasheetu [7], nebo na obr. 3.3.1.

Pro linku RS232 se typicky používá konektor „canon 9“, případně „cannon 9“. V nejjednodušším případě zapojení jsou pro přenosovou linku použity piny 2, 3, 5 na konektoru. Pin 2 pro Rx (recieved data), pin3 pro Tx(transmited data) a pin 5 pro zem GND. Mezi komunikujícími zařízení se kabel kříží, tzn. z pinu Rx prvního zařízení se připojuje vodič na pin Tx druhého zařízení. Příklad zapojeni je uveden na obr. 3.3.2.



Obr. 3.3.1 Zapojení převodníku RS232 zdroj [7].



Obr. 3.3.2 Zapojení převodníku RS232 mezi PC a mikrokontrolérem.

Popis funkce programu

Po stisknutí tlačítka B1 bude rozsvícena LED dioda připojená k pinu PC9 a prostřednictvím periferie USART1 odeslán znak „0“. Počet odeslaných znaků záleží na době trvání stisknutého tlačítka a na nastavené přenosové rychlosti periferie USART1. Stručně řečeno pokud bude tlačítko stisknuté, budou se nepřetržitě odesílat znaky. V případě, že mikrokontrolér přijme na portu Rx nějaký znak, vyvolá přerušení a znak v obslužné rutině přerušení uloží do registru r5.

Nastavení procesoru a periferií

Příklad „USART“ používá nastavení registrů GPIO periferií a RCC bloku hodin stejná jako v předchozích úlohách „blikani“ a „tlacitko“. Navíc je doplněna část zabývající se nastavením periferie USART pro ovládání sériové linky a periferie NVIC sloužící k obsluze přerušení.

Z konfigurace RCC bloku v příkladu „blikani“ viz. obr. 3.3.4 , je nakonfigurovaná frekvence hodin za blokem AHB na 48 MHz. Následující blok APB je nastaven na dělení = 1 a hodinová frekvence označená PCLK je tedy také 48 MHz.

Pro přesnější časování přenosu je možné použít vnější krystalem řízený oscilátor HSE. V tomto příkladu byl použit pro generování hodin vnitřní RC HSI oscilátor, který je pro tuto aplikaci vyhovující.

Nastavení periferie USART

Standardně jsou z důvodu dosažení nízké spotřeby obvodu všechny periferie bez připojeného hodinového kmitočtu, proto je nutné před použitím periferie povolit hodinový kmitočet nastavením bitu v registru USART_APB2ENR. Tento krok se v programu provádí v těle funkce „conf_IO_periferie“ viz. níže.

```
LDR r0,[r1,#0x18]      ;nacteni registru RCC_APB2ENR
LDR r2,=USART1EN       ;maska pro povoleni hodin pro USART1 bit14=1
ORRS r0,r2
STR r0,[r1,#0x18]      ;ulozeni povoleni hodin do reg. RCC_APB2ENR
```

V následujícím kroku program v části „conf_USART_periferie“ provádí nastavení registrů periferie USART. Prvním krokem je přiřazení AF funkce(alternate function) pinům PA10 a PA9. Tímto přiřazením se piny odpojí od output data registru, input data registr zůstává připojen viz obr. 3.3.3 referenční manuál [1], str. 121.

```
conf_USART_periferie
        LDR r2,=GPIOA_base
        LDR r0,[r2]
        LDR r1,=0x280000       ;nastaveni alternativnich funkci na piny PA9_RX
                               a PA10_TX
        ORRS r0,r0,r1
        STR r0,[r2]            ;nastaveni modu do GPIOA->MODER registru
```

Dále je nutné namapovat z množiny alternativních funkcí na pin9 USART1_TX a pin10 USART1_RX . Tyto informace viz obr. 3.3.3 nelze bohužel nalézt v [1], nacházejí se v datasheetu obvodu STM32F051x4 [2], str. 33.

```
LDR r0,[r2,#0x24]      ;GPIOA->AFRH
LDR r1,=0x110          ;nastaveni AF2 pro pin10 Rx a pin9 Tx
STR r1,[r2,#0x24]      ;nastaveni modu do GPIOA->AFRH registru
```

Tab. 3.3.1 Tabulka alternativní funkce pinů zdroj [2], str. 33.

Table 12. Alternate functions selected through GPIOA_AFR registers for port A

Pin name	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
PA0		USART2_CTS	TIM2_CH1_ ETR	TSC_G1_IO1				COMP1_OUT
PA1	EVENTOUT	USART2_RTS	TIM2_CH2	TSC_G1_IO2				
PA2	TIM15_CH1	USART2_TX	TIM2_CH3	TSC_G1_IO3				COMP2_OUT
PA3	TIM15_CH2	USART2_RX	TIM2_CH4	TSC_G1_IO4				
PA4	SPI1_NSS/ I2S1_WS	USART2_CK		TSC_G2_IO1	TIM14_CH1			
PA5	SPI1_SCK/ I2S1_CK	CEC	TIM2_CH1_ ETR	TSC_G2_IO2				
PA6	SPI1_MISO/ I2S1_MCK	TIM3_CH1	TIM1_BKIN	TSC_G2_IO3		TIM16_CH1	EVENTOUT	COMP1_OUT
PA7	SPI1_MOSI/ I2S1_SD	TIM3_CH2	TIM1_CH1N	TSC_G2_IO4	TIM14_CH1	TIM17_CH1	EVENTOUT	COMP2_OUT
PA8	MCO	USART1_CK	TIM1_CH1	EVENTOUT				
PA9	TIM15_BKIN	USART1_TX	TIM1_CH2	TSC_G4_IO1				
PA10	TIM17_BKIN	USART1_RX	TIM1_CH3	TSC_G4_IO2				
PA11	EVENTOUT	USART1_CTS	TIM1_CH4	TSC_G4_IO3				COMP1_OUT
PA12	EVENTOUT	USART1_RTS	TIM1_ETR	TSC_G4_IO4				COMP2_OUT
PA13	SWDAT	IR_OUT						
PA14	SWCLK	USART2_TX						
PA15	SPI1_NSS/ I2S1_WS	USART2_RX	TIM2_CH1_ ETR	EVENTOUT				

Sběrnice RS232 má být v klidovém stavu na úrovni log 1, z tohoto důvodu v následující části programu prostřednictvím registru GPIOA_MODER připojí k výstupu Tx pin PA9 vnitřní zdvihací rezistor Pull UP.

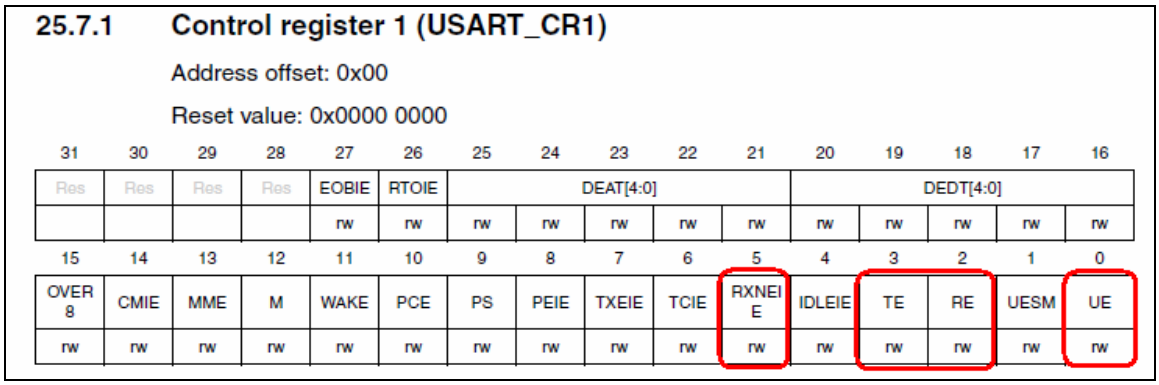
```
LDR r0,[r2,#0x0C]      ;GPIOA->MODER
LDR r1,=0x40000        ;nastaveni Pull UP pro pin9 Tx
STR r1,[r2,#0x0C]      ;nastaveni modu do GPIOA->MODER registru
```

Jak již bylo v předchozí části textu uvedeno frekvence hodinového signálu PLCK, která je připojena do periferie USART1 má hodnotu 48 MHz. Do registru USART1_BRR je nutné vložit takovou hodnotu dělitele této frekvence, aby výsledná hodnota odpovídala požadované rychlosti seriové linky. Pro žádanou hodnotu 9600 baud bude nutná hodnota registru USART_BRR=48000000/9600=5000=1388hex.

```
LDR r2,=USART1_base
LDR r1,=0x1388         ;1388=5000 , 48000000/9600=5000
STR r1,[r2,#0x0C]      ;nastaveni hodnoty baudrate do USART1->BRR
                        registru
```

Pokud budeme chtít využít přijímání znaků a nebudeme chtít použít kontrolu registru v nekonečné smyčce, můžeme povolit přerušení RXNEIE v registru USART_CR1. Přerušení je vyvoláno pokud není registr USART_RDR prázdný popis, [1] na str.632 poznámka „Bit 5 RXNE“.

```
LDR r1,=0x2D           ;povoleni RX,TX, preruseni po prichodu zpravy
                        a zapnuti usart (RXNEIE,TE,RE,UE=1)
ORRS r0,r0,r1
STR r0,[r2]            ; uloz do USART_CR1
```



25.7.1 Control register 1 (USART_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	EOBIE	RTOIE	DEAT[4:0]					DEDT[4:0]				
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVER 8	CMIE	MME	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Obr. 3.3.4 Registr USART_CR1 zdroj[1], str. 614 nastavené bity označené červeně.

Pro plnou funkci přerušení je nutné povolit funkci periferie jádra v registru NVIC_ISER. Popis a adresy registrů jsou uvedeny v programovacím manuálu zdroj [3], str.70. Jaký bit je nutné pro specifické přerušení najdete v referenčním manuálu [1], tabulka 27, str.160.

```
;nastaveni vektoru preruseni
LDR r2,=ISER_reg        ;adresa registru NVIC_ISER adr.  0xE000E100
LDR r0,=0x08000000      ;hodnota pro povoleni preruseni od
                        USART1(1<<27)
STR r0,[r2]             ;ulozeni hodnoty zpet do registru
```

Popis výkonné části programu

V hlavní části programu procházíme a kontrolujeme v hlavní smyčce stav tlačítka připojeného k portu A pinu 0. Stisk tlačítka vyhodnocujeme pomocí instrukce logického násobení ANDS. Pokud je výsledkem operace ANDS nula nastaví se příznakový bit „Z“ (zero) v registru příznaků na hodnotu 1. Adekvátně pokud bude výsledek ANDS různý od nuly bit Z bude nulový. Hodnotu příznakového bitu „Z“ vyhodnocujeme instrukcí větvení BNE (branch not equal). V případě , že bit „Z“ má hodnotu 0 (předchozí operace ovlivňující příznak Z neměla výsledek 0), provede skok na návěstí „odesli_znak“.

```
tlacitko
        LDR r1,=GPIOC_base
        LDR r2,=GPIOA_base
        LDR r0,[r2,#0x10]       ;nacteny hodnoty GPIOA_IDR registru do
                                reg. r0, obsahuje udaj o stavu vstupu brany PA
        MOVS r3,#0x1
        ANDS r0,r3              ;vymaskovani hodnoty pro PA0 kde je  tlacitko
        LSLS r0,#0x9            ;posun hodnoty v R0 doleva o 9bitu
        STR r0,[r1,#0x14]       ;ulozeni hodnoty z R0 do GPIOC_ODR, tj
                                 rozsviceni ci zhasnuti LED na PC9
        BNE odesli_znak         ;vyhodnocuje Z nastaveny posledni predchozi
                                 instrukci ktera nastavila Z tj. ANDS
        B tlacitko              ;skok na blikaci smycku
```

V části programu odesílající znak na pin 9 Tx nejdříve kontrolujeme zde je nastaven bit TXE v registru USART_ISR viz.obr. 3.3.5. Po přesunu dat do registru USART_TDR

obr. 3.3.7 hardware shodí bit TXE v registru USART_ISR. Když dojde k přesunu z registru USART_TDR do shift registru, pro postupné odeslání z dat hardware, hardware nastaví bit TXE na hodnotu 1, teprve poté je možné bez ztráty dat přesunout nová data do registru USART_TDR.

Pro pokyn počítači, aby vypsal v jeho programu přijatý znak na obrazovku používáme odeslání ukončovacího znaku, zde např. „@“, tento znak je možné podle potřeby v programu změnit (zpravidla se používá pro odřádkování „LF“ s kódem 0x0Ah.).

Přesný popis funkce bitu TXE je zobrazen na obr. 3.3.6. Také je možné využít v programu funkci bitu TC, který má význam Transfer complete, bližší podrobnosti lze nastudovat v [1] str. 632.

```
odesli_znak
        BL kontrola_TX_empthy
        LDR r1,=0x30            ;hodn.znaku ktery bude poslan ASCII "0"= 0x30h
        STR r1,[r0,#0x28]       ;ulozeni znaku 0x30 do registru USART1_TDR a tim dojde k jeho odeslani (only when TXE=1)
        BL kontrola_TX_empthy
        LDR r1,=0x40            ;ukoncovaci znak zde "@" 0x40h, obecne se pouziva LF new line=0x0Ah
        STR r1,[r0,#0x28]       ;ulozeni znaku 0x30 do registru USART1_TDR a tim dojde k jeho odeslani (only when TXE=1)

        B tlacitko
```

25.7.8 Interrupt & status register (USART_ISR)

Address offset: 0x1C

Reset value: 0x0000 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
									r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABRF	ABRE	Res	EOBF	RTOF	CTS	CTSIF	LBDF	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
r	r		r	r	r	r	r	r	r	r	r	r	r	r	r

Obr. 3.3.5 Registr USART_ISR zdroj[1], str. 629.

Bit 7 **TXE**: Transmit data register empty

This bit is set by hardware when the content of the USART_TDR register has been transferred into the shift register. It is cleared by a write to the USART_TDR register.

The TXE flag can also be cleared by writing 1 to the TXFRQ in the USART_RQR register, in order to discard the data (only in Smartcard T=0 mode, in case of transmission failure).

An interrupt is generated if the TXEIE bit =1 in the USART_CR1 register.

0: Data is not transferred to the shift register

1: Data is transferred to the shift register)

Note: This bit is used during single buffer transmission.

Obr. 3.3.6 Popis funkce bitu TXE zdroj [1], str. 631.

25.7.11 Transmit data register (USART_TDR)

Address offset: 0x28

Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	TDR[8:0]								
							rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:9 Reserved, must be kept at reset value.

Bits 8:0 **TDR[8:0]**: Transmit data value

Contains the data character to be transmitted.

The TDR register provides the parallel interface between the internal bus and the output shift register (see *Figure 229*).

When transmitting with the parity enabled (PCE bit set to 1 in the USART_CR1 register), the value written in the MSB (bit 7 or bit 8 depending on the data length) has no effect because it is replaced by the parity.

Note: This register must be written only when TXE=1.

Obr. 3.3.7 Registr USART_TDR,zdrj [1], str. 636.

Druhou hlavní funkcí programu je přijímáni znaku přicházejících na pin 10 Rx periferie USART1. Po přijetí znaku do registru USART_RDR hardware vyvolá přerušení uloží návratovou adresu do registru LR a skočí na adresu zapsanou v tabulce vektorů přerušení. Začne se vykonávat část programu uvedená pod návěstím USART1_IRQHandler.

Ke skoku do přerušovací rutiny dojde také v případě pokusu zapsat do registru USART_RDR data, když předchozí data ještě nebyla přečtena. Pokud máme povoleno více přerušení z množiny možných přerušení pro periferii USART1, všechna tato přerušení spustí stejnou funkci pro obsluhu přerušení. V této funkci by měla být tato přerušení testováním příslušného bitu registru USART_ISR rozpoznána a podle jejich druhu adekvátně obsloužena.

V tomto programu při vykonávání funkce obsluhy přerušení dojde nejprve k přečtení registru USART_RDR a jeho uložení. Dále se kontroluje zda je bit ORE registru USART_ISR nahozen, tzn. zda v registru USART_RDR došlo k přepisovaní zatím nepřečtených dat. Pokud k tomuto dojde je nutné v registru USART_ICR potvrdit vzniklou událost zapsáním jedničky do bitu ORECT, aby program mohl pokračovat v činnosti. V případě, že bit ORE není ve stavu 1, provede se skok na návěští „navrat_read“.

```
USART1_IRQHandler                ;label pro preruseni
        LDR r0,=USART1_base
        LDR r5,[r0,#0x24]        ;load USART data recieve register USART_RDR

        LDR r1,[r0,#0x1C]        ;nacteni USART_ISR registru
        LDR r2,=0x08             ;maska pro register ISR zda je bit ORE=1
        ANDS r1,r1,r2            ;
        BEQ navrat_read          ;skok kdyz preruseni neni  od overrun

clear_ORE_error
        LDR r1,[r0,#0x20]        ;nacteni USART_ICR registru
        LDR r2,=0x08             ;maska pro bit ORECF, overrun error
        ORRS r1,r1,r2
        STR r1,[r0,#0x20]        ;smazani interrupt overrun error
navrat_read
        BX LR                    ;navrat z preruseni
            END
```

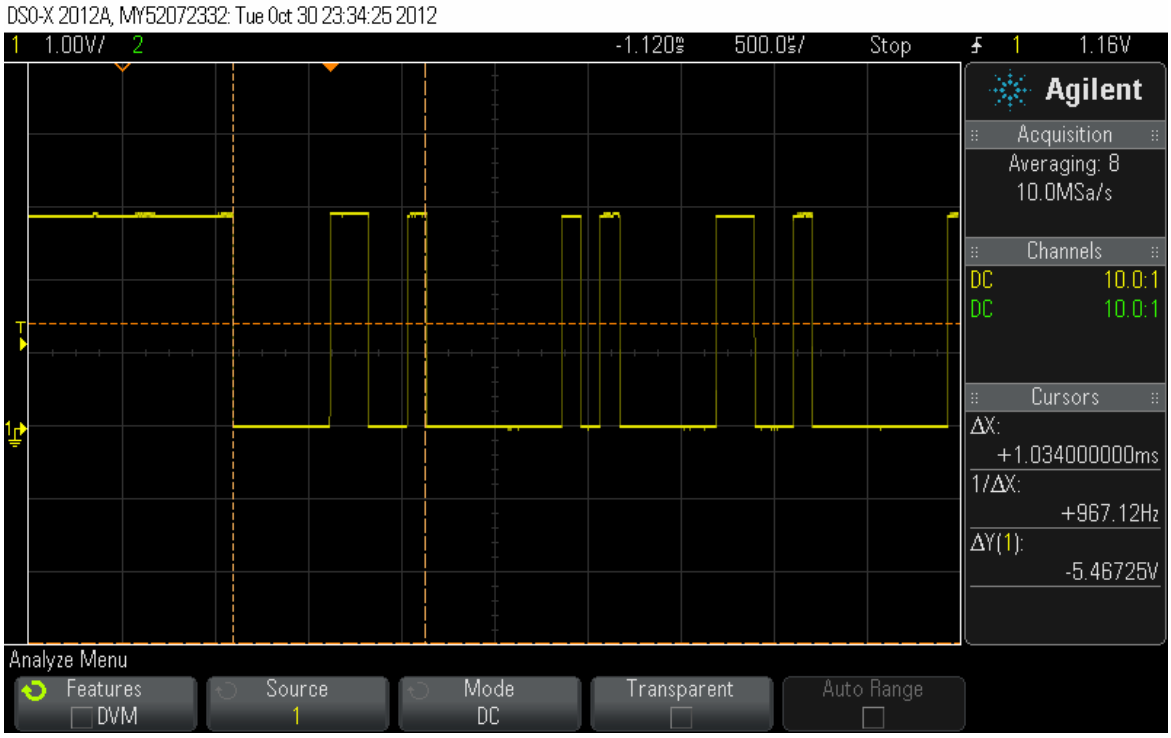
Příklady dat posílaných po RS232

Na obrázcích 3.3.8 a 3.3.9 jsou zobrazeny znaky vyslané a přijaté modulem STM32F0-DISCOVERY. Přenos jednoho znaku začíná vždy start bitem tj. z klidového stavu sběrnice log. 1 vyšle harware jeden bit s hodnotou 0, dále následuje zpravidla 8 bitů dat, nejprve se přenáší nejnižší bit. Přenos znaku hardware ukončuje vysláním stop bitu, stop bit má hodnotu 1, zde délky podle nastavení 1 bit.

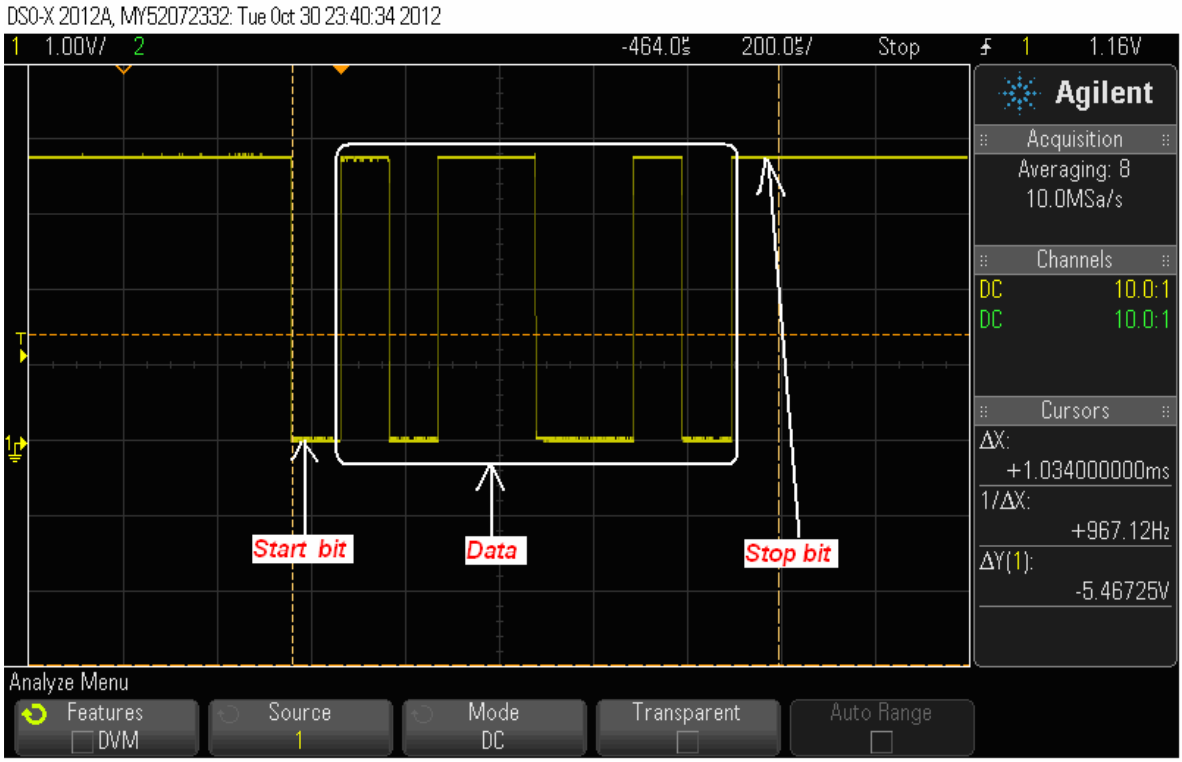
Na obrázcích 3.3.8 a 3.3.9 jsou měřeny průběhy s napětími, které poskytuje obvod STM32F051 tj. 0 až přibližně napětí 3 V, napětí linky RS232 mají definována napětí v rozsahu uvedeném v tabulce Tab. 3.3.2.

Tab. 3.3.2 Úrovně napětí RS232.

	Strana Vysílače	Strana přijímače
log.1	+5 až +15VDC	+3 až +25VDC
log.0	-5 až -15VDC	-3 až -25VDC



Obr. 3.3.8 Vysílaná posloupnost znaků 0x30h, 0x40h svorka Tx pin 9 STM32 modulu.



Obr. 3.3.9 Vyslaný znak "M", 0x4D tj.01001101b z PC, měřeno na Rx pinu 10 modulu STM32.

3.4 Příklad 4 „TIMER“ v ASM

Čítače-úvod

Tento příklad se zabývá ukázkou použití jednotky časovače mikrokontroléru STM32F051 zabudovaném ve vývojovém kitu STM32F0-Discovery.

Periferie časovače pracuje na využití vlastností čítače. Hodnota čítače je zvyšována, nebo snižována s předem určenou periodou, která je odvozena od hodinového kmitočtu periferie. Celkový počet provedených period do nějaké události, např. stav čítače rovný nule, shoda s komparačními registry, je nastavením doby chodu časovače. Tato událost časovače může vyvolat přerušení, ve kterém mohou být provedeny dílčí požadované kroky programu. Výhodou časovače oproti zpožďovací smyčce použité v příkladu 1 „blikáni“ je skutečnost, že svým chodem nezatěžuje procesor.

Obvod STM32F0xx obsahuje několik čítačů/časovačů a dále jeden speciální s názvem SYSTICK, jehož hodinová frekvence je odvozena od AHB clock dělené 8.Při hodinové frekvenci MCU 48 MHz bude základní frekvence pro SYSTICK 48/8= 6 MHz. Nastavením kalibrační konstanty např. na 6000 lze získat volání přerušení s frekvencí 1ms. Tento časovač po jeho povolení pracuje nepřetržitě a je schopen po nadefinovaném počtu period systémových hodin periodicky vyvolávat přerušení.

Základní vlastnosti rozhranní TIMx

Obvod STM32F0xx obsahuje 5 skupin časovačů. Každá skupina má vedle implementace obecných funkcionalit další specifické vlastnosti.

TIM1
- 16bit s auto reload registrem
- Směr čítání nahoru, dolů, oba směry
- 16bit dělička pro zvýšení rozsahu časovače
- 4 nezávislé vstupní kanály
- Input Capture(měření délky pulsu)
- Output compare(generování signálů)
- Generování PWM
- One-pulse mode output
- Synchronizace s externími signály a propojeni časovačů
- Break input pro uvedení časovače do známého stavu
- Přerušení s generováním DMA
- Podpora měření pomocí enkodéru, hallova senzoru
- Externí vstup hodin pro spouštění cycle-by-cycle řízení

TIM2,3
- 32bit TIM2 s auto reload registrem
- 16bit TIM3 s auto reload registrem
- Shodné vlastnosti jako TIM1

TIM6
- 16bit s auto reload registrem
- Má jen základní vlastnosti
- Nemá vstupní kanály
- Pro řízení DAC, interně propojený s DAC

TIM14 - 16bit s auto reload registrem
- Pro kalibraci hodinového kmitočtu MCU
- Jeden vstupní kanál

TIM15 -16bit s auto reload registrem
- Jeden vstupní kanál
- Break vstup

TIM16, 17 -16bit s auto reload registrem
- Jeden vstupní kanál
- Break vstup

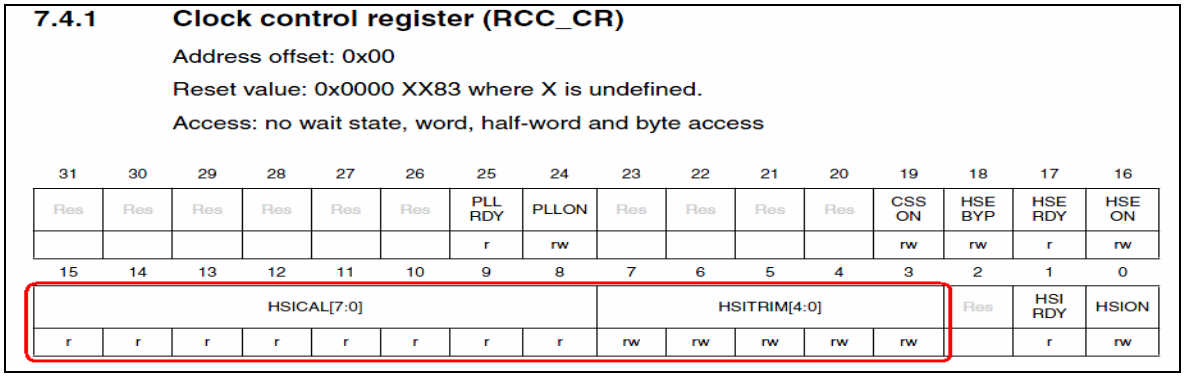
Periferie časovačů jsou u mikrokontroléru řady STM32F0 poměrně složité a je nutné v případě jejich použití, nastudovat jejich vlastnosti a nastavení z vhodné dokumentace např. [1].

Přesnost časovače

V případě, že nebudeme používat pro mikrokontrolér krystal pro generování hodinového kmitočtu, může být užitečné zvýšit přesnost RC oscilátoru HSI pomocí jemného doladění úpravou hodnoty registru RCC_CR.

Každý obvod je při výrobě kalibrován pomocí hodnot HSICAL+HSITRIM registru RCC_CR viz. obr. 3.4.1. Z důvodu teplotní závislosti frekvence tohoto RC oscilátoru a nedokonalosti nastavení korekce kmitočtu při kalibraci ve výrobě, je ponechána možnost dostavení korekce kmitočtu HSI oscilátoru uživateli pomocí zápisu hodnoty do oblasti nejméně významných 5 bitů označených HSITRIM. Defaultně je hodnota HSITRIM 0x10h, pro tuto hodnotu a teplotu obvodu 25 °C, je výrobcem garantována hodnota frekvence HSI generátoru na 8 MHz ± 1%. Při rozsahu teplot -40 °C až -105 °C může tolerance vzrůst až na ± 3%.

Pro zvýšení přesnosti oscilátoru je možné použít časovač TIM14 taktovaný hodinovou frekvencí HSI v režimu capture. Během periody externího zdroje kmitočtu LSE(Low speed external) nebo HSE/32 s přesností řádu ppm načítá čítač pulsy HSI generátoru. Z jejich poměru lze získat hodnotu pro dokalibrování HSI generátoru. Pro kalibraci je také možné použít hodnotu frekvence rozvodné síťě 50 Hz. Více informaci najdete např. v [4] nebo v [1], str. 90.



7.4.1 Clock control register (RCC_CR)

Address offset: 0x00

Reset value: 0x0000 XX83 where X is undefined.

Access: no wait state, word, half-word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	PLL RDY	PLLON	Res	Res	Res	Res	CSS ON	HSE BYP	HSE RDY	HSE ON
						r	rw					rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSICAL[7:0]								HSITRIM[4:0]					Res	HSI RDY	HSION
r	r	r	r	r	r	r	r	rw	rw	rw	rw	rw		r	rw

Obr. 3.4.1 Nastavení registru HSI (High speed external).

Popis funkce programu

Po připojení modulu k napájecímu napětí začne dioda připojená k pinu PC9 nepřetržitě blikat s frekvencí přibližně 0,5 Hz. Tato frekvence je pevně zadána v programu.

Nastavení procesoru a periferií

Příklad „TIMER“ používá nastavení registrů GPIO periferií a RCC bloku hodin podobná jako v předchozích úlohách „blikani“ a „tlacitko. Nově je použita periferie NVIC, která je nastavena na generování události přerušení „TIM1_BRK_UP_TRG_COM (TIM1 Break, update, trigger and commutation interrupt).

Nastavení RCC bloku zůstává shodné jako v příkladu 1 „blikani“, kde je podrobně vysvětleno. Frekvence hodin za blokem AHB je 48 MHz. Následující APB blok je nastaven na dělení=1, hodinová frekvence označená PCLK je 48 MHz. Dle obr. 3.4.4 v příkladu 1 „blikani“ je násobící koeficient bloku před vstupem do časovačů v případě nastavení děličky APB na hodnotu 1 je roven také jedné. Výsledná frekvence CK_INT vstupující do časovače je potom rovna 48 MHz.

Nastavení periferie TIM1

Nastavení periferie začíná obvyklým způsobem tj. povolením hodinového kmitočtu pro časovač TIM1 v registru RCC_APB2ENR.

```
;povoleni hodin RCC pro TIM1
LDR r0,[r1,#0x18]   ;nacteni registru RCC_APB2ENR
LDR r2,=TIM1EN      ;nacteni povolovaci masky hodiny bit11 TIM1EN
ORRS r0,r2
STR r0,[r1,#0x18]   ;pov.  hodin TIM1  v registru RCC_AHB2ENR
```

Změna stavu portu PC9 je prováděna v obsluze přerušení. Je tedy nutné zařídit povolení časovači TIM1 nějaké přerušení generovat. K tomu slouží následující kód, kterým se povolí volání přerušovací rutiny při události update TIM1 nastavením bitu UIE=1 (Update interrupt enable) v registru TIM1_DIER.

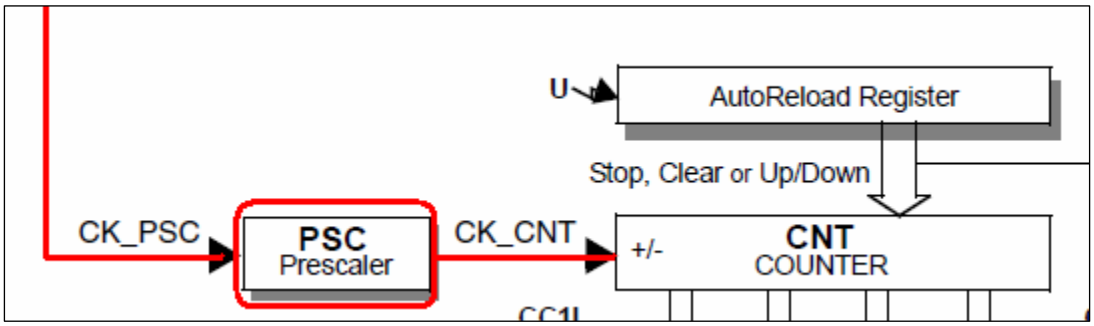
```
;TIM1_config
;interrupt enable
LDR r0,=TIM1_base
LDR r1,[r0,#0x0C]        ;nacteni hodnoty registru TIM1_DIER
LDR r2,=0x01             ;maska pro povoleni preruseni od update tim1
bit                       UIE=1
ORRS r1,r1,r2
STR r1,[r0,#0x0C]        ;ulozeni hodnoty zpet do registru TIM1_DIER
```

Při podtečení stavu čítače dojde k nahrání nové hodnoty do registru čítače z auto-reload registru TIM1_ARR. Maximální hodnota zapsatelná do tohoto registru je v rozsahu 16 bitů tj. 65536 dekadicky. Čítač je zablokován v případě nulové hodnoty tohoto registru. Hodnotu registru je možné měnit i za běhu časovače, přesné chování v závislosti na nastavení ostatních registrů popisuje manuál [1] od str. 228.

```
; ulozeni hodnoty do registru TIM1_ARR auto reload register
LDR r1,=0xFFFF          ; max velikost 16 bitu
STR r1,[r0,#0x2C]       ; ulozeni do registru TIM1_ARR
```

Další registr nastavující chování jednotky TIM1 je TIM1_PSC. Registr v sobě uchovává hodnotu, kterou bude dělen vstupní kmitočet CK_PSC viz obr. 3.4.2. V tomto příkladě CK_INT=CK_PSC.

```
; ulozeni hodnoty do registru TIM1_PSC (count prescaler)
LDR r1,=0x63            ; deleni hodin=PSC+1 max velikost 16 bitu
STR r1,[r0,#0x27]       ; ulozeni do registru TIM1_PSC
```



Obr. 3.4.2 Dělička vstupního hodinového signálu výsek z bloku časovače TIM1.

Výpočet frekvence volání přerušovací rutiny

f_{CK_INT} .....frekvence interních hodin zde 48MHz

f_{CK_CNT} .....frekvence hodin za blokem PSC

PSC .....obsah registru TIM1_PSC

ARR .....obsah registru TIM1_ARR

f_{INT} .....frekvence volání přerušení Upload interrupt event UIE

$$f_{CK\_CNT} = \frac{f_{CK\_INT}}{PSC+1} \qquad f_{INT} = \frac{f_{CK\_CNT}}{ARR} \tag{3.4.1}$$

Vzorový příklad výpočtu

$$f_{CK\_CNT} = \frac{f_{CK\_INT}}{PSC+1} = \frac{48\cdot 10^6}{999+1} = 48\cdot 10^3\, Hz \tag{3.4.2}$$

$$f_{INT} = \frac{f_{CK\_CNT}}{ARR} = \frac{48\cdot 10^3}{48000} = 1Hz \tag{3.4.3}$$

Z výpočtu vyplývá, že při nastavení PSC na 999 bude frekvence volání přerušení 1 Hz. V přerušení se vždy změní stav výstupu PC9 na opačnou úroveň, frekvence blikání LED proto bude 2x pomalejší, tedy 0,5 Hz.

Posledním nastavením časovače TIM1 je povolení jeho chodu, tzn. start časovače.

```
;counter enable
LDR r1,[r0]          ; hodnota z registru TIM1_CR1 do r1
LDR r2,=0x1          ; povoleni citace counter enabled CEN=1
ORRS r1,r1,r2
STR r1,[r0]          ; zápis do registru TIM1_CR1
```

Popis výkonné části programu

Hlavní smyčka programu je velice jednoduchá, skládá se z jedné instrukce NOP a z nepodmíněného skoku na adresu s touto instrukcí.

```
loop
        NOP
        B loop          ;nekonecna smycka
```

Tato smyčka je přerušována s frekvencí danou událostmi upload registru čítače. Vždy se čeká na dokončení zpracovávané instrukce a potom se vykonávání programu přeruší. Doba trvání instrukce NOP je jeden cykl hodinového kmitočtu procesoru, doba trvání instrukce skoku je 3 cykly. Nejistota, jitter spuštění přerušení bude tedy maximálně 3 cykly tj. 3/48 MHz 62.5 ns.

3.5 Příklad 5 „ADC“ v ASM

ADC – úvod

Pro měření vstupních úrovní napětí je obvod STM32F05x vybaven jedním ADC aproximačním převodníkem s maximálním rozlišením 12bitů. Tento převodník má 19 multiplexovaných kanálů, umožňujících připojení 16 externích a 3 interních zdrojů signálu (interní teplotní senzor, napětí baterie, interní napěťová reference). V následujícím textu se pokusím na příkladu vysvětlit některé základní vlastnosti a nastavení jednotky ADC mikrokontroléru.

Základní vlastnosti rozhranní ADC

- nastavitelné rozlišení převodníku na 12, 10, 8 nebo 6 bitů.

- auto-kalibrace převodníku ADC

- podpora DMA (přímý přístup do paměti)

-overrun interrupt, pro detekci ztráty dat z převodu

- start převodu
  - softwarem
  - timerem(tim1, 2, 3, 15)

- režim převodů
  - scan jednoho nebo více kanálu
  - jednorázový nebo kontinuální převod

- generování přerušení
  - dokončeno vzorkování
  - dokončen převod
  - dokončena sekvence převodů zvolených kanálů
  - překročení mezí analogového watchdogu
  - overrun událost

- napájení ADC převodníku V_{DDA} od 2,4 do 3,6VDC

- vstupní úrovně napětí v mezích $V_{SSA} \leq V_{IN} \leq V_{DDA}$ (3.5.1)

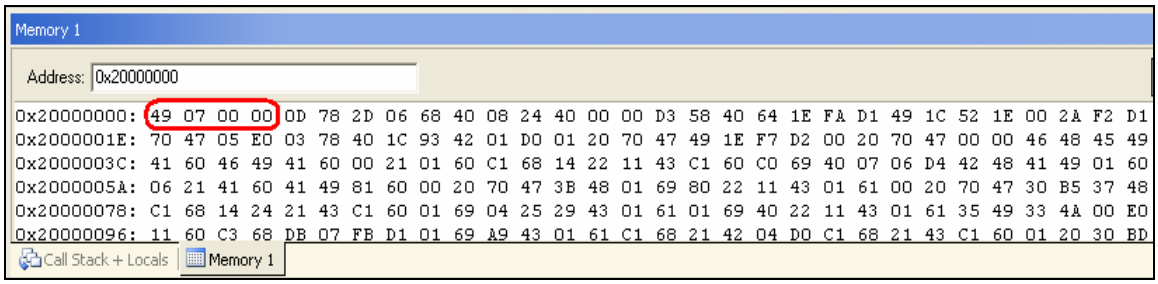
- doba převodu závislá na rozlišení, hodinové frekvenci ADC periferie

Funkce programu

Program byl napsán v prostředí uVision4 pro jednoduchou ukázku funkce rozhranní ADC převodníku. Pro testování programu byl použit vývojový kit STM32F0-Discovery.

Program sleduje v nekonečné smyčce úroveň napětí na vstupu PC4 a hodnotu normovanou dle nastaveného rozlišení zapisuje do paměti SRAM na adresu 0x20000000, kde ji můžeme za běhu programu sledovat online v okně „Memory“ ladícího prostředí (debuggeru) uVision4. Pokud není okno již zobrazené použijeme pro jeho zobrazení volbu z lišty View-

>Memory Windows->Memory1. Příklad zobrazení údaje z tohoto okna je na obrázku obr. 3.5.1. Data jsou uložena ve formátu „Little endien“, hodnota ADC tedy byla 0x749hex.



Obr. 3.5.1 Zobrazení paměti se zapsaným stavem hodnoty ADC vstupu.

Nastavení periferie ADC

Periferii ADC je možné nastavit mnoha různými způsoby, které jsou popsány v manuálu [1] od str. 170 do str. 206.V našem příkladu je použito níže popsané nastavení registrů. Pod textem je vždy zobrazen kód realizující popsané nastavení.

Na začátku nastavení periferie ADC, je opět nutné přivést do této jednotky hodinový signál z periferie RCC. Tento signál může mít maximální frekvenci až 14 MHz. V našem případě je hodnota frekvence PCLK za blokem „APB prescaler“ 48 MHz, následuje blok „ADC Prescaler“ nastavený na dělení čtyřmi (bit14 ADCPRE registru RCC_CFGR(0x04)=1). Frekvence hodinového signálu pro blok periferie ADC je tedy 12 MHz.

```
;povoleni hodin RCC pro ADC
LDR r0,[r1,#0x18]        ;nacteni registru RCC_APB2ENR
LDR r2,=ADCEN            ;nacteni povolovaci masky hodin pro ADC
ORRS r0,r2
STR r0,[r1,#0x18]        ;povoleni hodin pro ADC,  reg. RCC_AHB2ENR
```

Nyní můžeme přejít k definici vstupního portu ADC. Nahlédnutím do datasheetu [2], str. 30, tab. 13, zjistíme, že by bylo možné použít pin PC4, který je namapován ke kanálu ADC AIN_14. Přiřadíme tedy pinu PC4 v registru GPIOC_MODER alternativní funkci .

```
conf_periferie
        ;GPIOC_config
        LDR r0,=GPIOC_base
        LDR r1,=0x0200      ;AF funkce pro PC4 tj. AIN_14
        STR r1,[r0]         ;ulozeni do registru ADC_CFGR1
```

Jak již bylo zmíněno v odstavci prezentujícím základní vlastnosti periferie ADC, zařízení disponuje funkcí auto-kalibrace. V níže uvedeném kódu se nejprve spustí provedení kalibrace a následně se ve smyčce kontroluje bit ADCAL registru ADC_CR. Nulová hodnota bitu ADCAL, má význam ukončené kalibrace.

```
;ADC_config
LDR r0,=ADC_base
;kalibrace ADC
LDR r1,=0x80000000        ;maska pro spusteni kalibrace ADCAL=1
```

```
                STR r1,[r0,#0x08]       ;start kalibrace ADC

calibrate_ADC                           ;test zda je ukoncena kalibrace
                LDR r2,[r0,#0x08]       ;nahrani stavu registru ADC_CR
                ANDS r2,r2,r1           ;nasobeni s maskou bitu31 ADCAL reg.
ADC_CR
                BNE calibrate_ADC       ;pokud neni bit ADCAL=0 testuj
```

V nastavení registru ADC_CFGR1 jsme definovali mód převodu ADC na kontinuální. A dále pokud dojde k události overrun, která představuje varování MCU, že nebyla vyčtena data z ADC převodníku(registr ADC_DR) před novým zápisem, budou předchozí data přepsána novými daty.

```
                ;konfigurace ADC  registr ADC_CFGR1
                LDR r1,=0x3000          ;CONT=1(Continuous mode)
                                         OVRMOD=1(overrun management mode)
                STR r1,[r0,#0x0C]       ;ulozeni do registru ADC_CFGR1
```

Jaké kanály budou zahrnuty do skenování, metodou multiplexování, definujeme v registru ADC_CHSELR. V tomto příkladě bude skenován pouze jeden kanál 14, připojený k pinu PC4.

```
                ;kanaly pro scanovani registr ADC_CHSELR
                LDR r1,=0x4000          ;vyber scan  kanal 14
                STR r1,[r0,#0x28]       ;ulozeni do registru ADC_CHSELR
```

Nyní jsou kanály nadefinovány a zbývá povolit funkci ADC převodníku tj. bit 0 registru ADC_CR nastavit na hodnotu 1. Poté ADC periferie čeká na spouštěcí signál pro převod měřené hodnoty tzv. trigger. V našem případě použijeme softwarový trigger, musíme tedy nastavit bit 2 ADSTART na hodnotu jedna. Tímto okamžikem začne ADC převodník nepřetržitě pracovat v kontinuálním režimu s ukládáním hodnot do registru ADC_DR (0x40).

```
                ;ADC enable, start
                LDR r1,[r0,#0x08]       ;nacteni hodnoty registru ADC_CR
                LDR r2,=0x01            ;maska pro povoleni ADEN
                ORRS r1,r1,r2
                LDR r2,=0x04            ;maska pro ADSTART
                ORRS r1,r1,r2
                STR r1,[r0,#0x08]       ;ulozeni hodnoty do registru ADC_CR
```

Popis výkonné části programu

Ve výkonné části programu provádíme cyklické čtení obsahu registru ADC_DR a zápis jeho hodnoty do proměnné ADC_hodnota. Proměnná ADC_hodnota je založena ve volatile paměti SRAM(bez napájení ztratí uložený obsah). Zde je proměnná první definovaná v pořadí a má tedy přidělenu v paměti SRAM první možnou adresu tj. 0x20000000h.

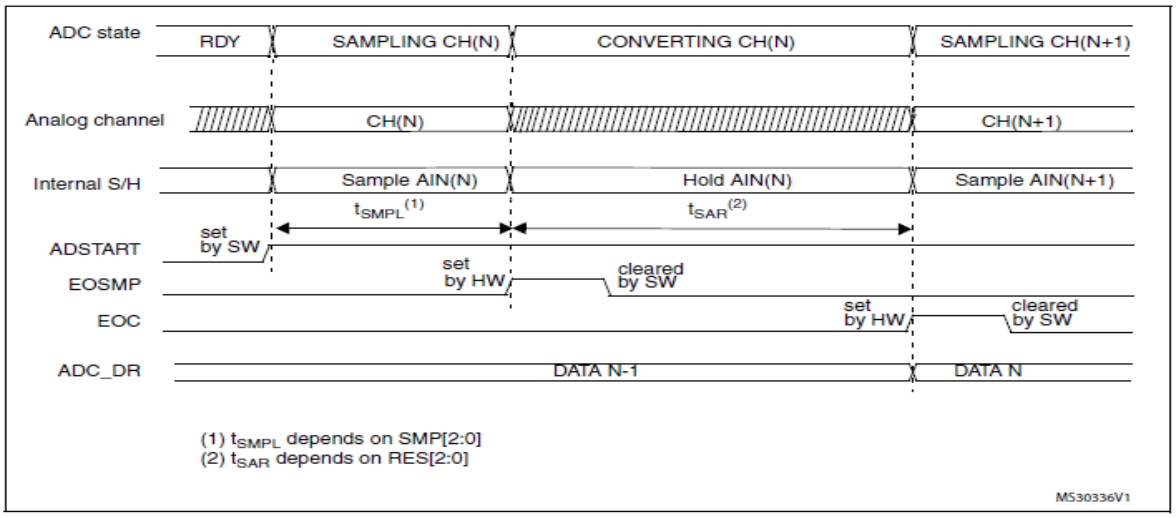
```
                LDR r0,=ADC_base
                LDR r1,=ADC_hodnota
```

```
loop
        LDR r5,[r0,#0x40]       ;zapis stav registru ADC_DR do r5
        STR r5,[r1]             ;ulozeni hodnoty ADC do pameti
        B loop

        AREA    text, DATA, READWRITE
ADC_hodnota     SPACE  4        ;alokace 4 byte(tj. 1 word) v pameti
                                 SRAM(READWRITE)
```

Časování ADC periferie

Z důvodů měření průběhů signálů ADC převodníkem je nutné zabývat se rychlostí převodu periferie. V manuálu [1] na straně 179 je uveden vzorec pro výpočet doby převodu jednoho kanálu ADC. Doba převodu je závislá na frekvenci hodinového signálu periferie ADC a na zvoleném rozlišení. Časový diagram převodu ADC je přehledně zobrazen na obr. 3.5.2. Doby cyklů v závislosti na nastavení jsou uvedeny v tab. 3.5.1.



Obr. 3.5.2 Časový diagram převodu ADC vstupu zdroj [1], str. 179.

Tab. 3.5.1 Počet cyklů pro převod při různých nastaveních, zdroj [1], str. 179.

RES[1:0] bits	t_{SAR} (ADC clock cycles)	t_{SAR} (ns) at f_{ADC} = 14 MHz	$t_{SMPL\ (min)}$ (ADC clock cycles)	t_{ADC} (ADC clock cycles) (with min. t_{SMPL})	t_{ADC} (µs) at f_{ADC} = 14 MHz
12	12.5	893 ns	1.5	14	1000 ns
10	11.5	821 ns	1.5	13	928 ns
8	9.5	678 ns	1.5	11	785 ns
6	7.5	535 ns	1.5	9	643 ns

Výpočet doby převodu ADC pro tento příklad

$t_{ADC} = t_{SMPL} + t_{SAR}$ (3.5.2)

t_{ADC} .....doba převodu jednoho kanálu

t_{SMPL} .....doba vzorkování

t_{SAR} .....doba konverze

Perioda hodinového signálu..... $t_{ADC\_CLK} = \frac{1}{12 \cdot 10^6} s = 83,\overline{3}\ ns$ (3.5.3)

Doba vzorkování..... $t_{SMPL} = 1,5 \cdot t_{ADC\_CLK} = 125\ ns$ (3.5.4)

Doba konverze..... $t_{SAR[12bit]} = 12.5 \cdot t_{ADC\_CLK} = 1042\ ns$ (3.5.5)

Celková doba převodu..... $t_{ADC} = t_{SMPL} + t_{SAR[12bit]} = (125 + 1024)\ ns = 1,149\ us$ (3.5.6)

Doba převodu je 1,149 us, to odpovídá frekvenci 870,322 kHz. Dle Nyquitsova kritéria by tedy maximální vstupní frekvence měla být cca 435 kHz.

Zapojení pro testování

Pro vyzkoušení převodníku byl připojen mezi svorky GND a 3 V potenciometr s hodnotou 50 kohm. Jeho běžec byl potom připojen na vstupní pin PC4. Změna hodnoty byla pozorována v debug módu prostředí uVision4, způsobem popsaným v kapitole „Funkce programu“ tohoto dokumentu.

3.6 Příklad 6 „DAC“ v ASM

DAC–úvod

Další periferií, kterou výrobce MCU STM32F05X implementoval je DAC převodník ve formě jednoho kanálu s názvem „Channel 1“. DAC převodník slouží ke generování výstupního napětí určeného číselnou hodnotou zapsanou do jeho registru. S DAC převodníkem je možné řídit navazující externí obvody, realizovat generátor signálu apod.

Základní vlastnosti rozhranní DAC

Níže jsou uvedeny vlastnosti DAC převodníku

- volitelné rozlišení 8, 10, 12 bitů

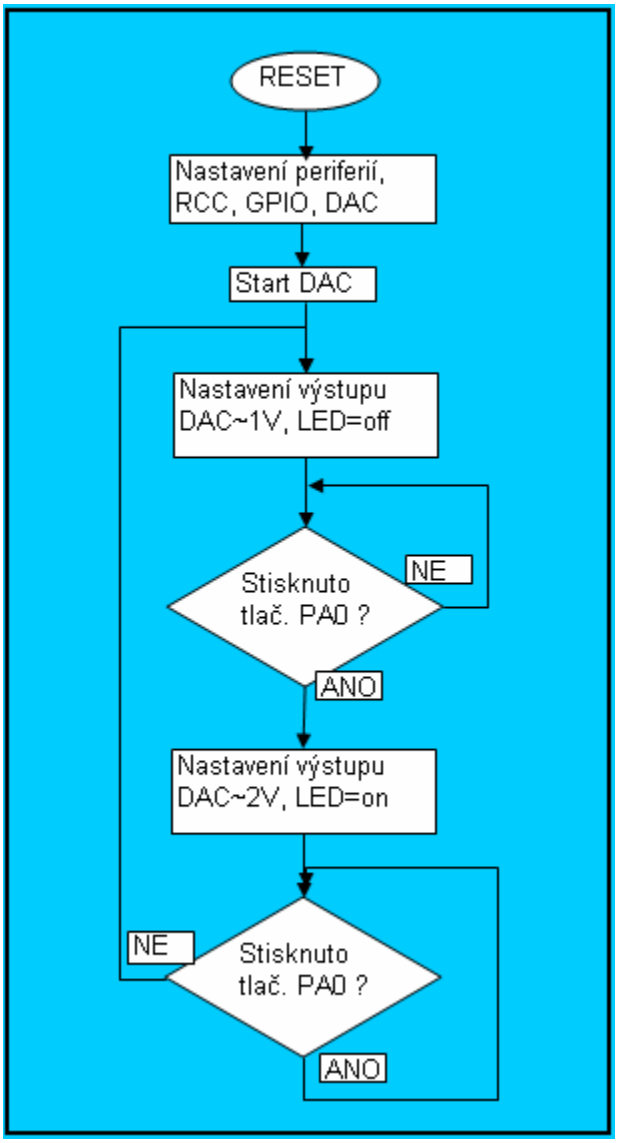
- provázání s periferií DMA

- trigger generovaný časovači TIM2, 3, 6, 15, externím pinem nebo softwarem

- vstupní napěťová reference max. rozsahu z napětí V_{DAA}

Funkce programu

Funkci programu nejlépe vystihuje vývojový diagram z obr. 3.6.1. Po nastavení periferií a odstartování DAC periferie je na výstup nastavena hodnota napětí odpovídající přibližně 1 volt. Pokud stisknete tlačítko PA0 na pinu PA4 bude po dobu stisku tlačítka hodnota napětí výstupu DAC přibližně 2 V a bude rozsvícena LED zelená dioda. Pokud dojde k uvolnění tlačítka dioda zhasne a napětí na pinu PA4 klesne zpět na úroveň jednoho voltu.



Obr. 3.6.1 Vývojový diagram programu „06_DAC“.

Nastavení periferie DAC

Nastavení začíná obvyklým připojením hodinového signálu prostřednictvím nastavením bitu 29 registru RCC_AHBENR do stavu 1.

```
,;povoleni hodin RCC pro DAC
        LDR r0,[r1,#0x1C]  ;nacteni registru RCC_APB1ENR
        LDR r2,=DACEN      ;nacteni povolovaci masky hodiny DAC
        ORRS r0,r2
        STR r0,[r1,#0x1C]  ;ulozeni do registru RCC_APB1ENR
```

Následuje povolení DAC kanálu 1 v registru DAC_CR.

```
conf_DAC
        ;DAC_config
        LDR r0,=DAC_base

        ;DAC start
        LDR r1,=DACEN1;DAC channel 1 enable
        STR r1,[r0];ulozeni do registru DAC_CR
```

5. Popis výkonné části programu

Následuje popis níže uvedených kódu hlavního programu příkladu „06_DAC“.

```
        LDR r0,=DAC_base        ;definice adres periferii
        LDR r1,=GPIOC_base
        LDR r2,=GPIOA_base
```

Program realizuje stavový automat s dvěma stavy, první stav je stav uvedený návěstím „tlac_off“, druhým stavem je stav „tlac_on, podmínkou přechodu do druhého stavu je stisk, nebo rozepnutí tlačítka připojeného k pinu PA0 MCU. Testování stavu tlačítka je prováděno ve dvou smyčkách označených návěstími „test_tlac_on“ a „test_tlac_off“. Při stavu „tlac_off“ je zhasnuta LED dioda připojená k pinu PC9 a na výstup DAC pin PA4 je nastavena hodnota napětí odpovídající přibližně napětí 1 volt. Při stavu „tlac_on“ je LED dioda rozsvícena a DAC je nastaven na hodnotu napětí odpovídající přibližně napětí 2 volt.

Zápis hodnoty pro výstup DAC převodníku, v případě 12bit rozlišení, zapíšeme do registru DAC_DHR12R1, odkud bude v případě režimu softwarového triggeru, během jednoho cyklu sběrnice APB1 přenesen do registru DAC_DOR1 a na výstupní pin DAC periferie.

```
tlac_off
        ;zhasnuti LED PC9
        LDR r3,[r1,#0x14]       ;nahrani stavu registru GPIOC
        LDR r4,=0xfffffdff      ;maska LED na PC9_OFF
        ANDS r3,r4
        STR r3,[r1,#0x14]       ;ulozeni do GPIOC_ODR
        LDR r3,=0x57F           ;dekadicky 1407~1Volt
        STR r3,[r0,#0x08]       ;nahrani do registru DAC_DHR12R1

        ;test tlacitko
test_tlac_on
        LDR r3,[r2,#0x10]       ;nacteni stavu vst. brany GPIOA_IDR do r3
        MOVS r4,#0x1            ;priprava masky
        ANDS r3,r4              ;vymaskovani hodn. pro PA0 kde je tlacitko
        BNE tlac_on             ;skok pokud bude tacitko stisknute
        B test_tlac_on

tlac_on
        ;rozsviceni LED
        LDR r3,[r1,#0x14]       ;nahrani stavu registru GPIOC
```

```
                LDR r4,=0x200          ;maska LED na PC9_ON
                ORRS r3,r4             ;soucet vysl.do r3
                STR r3,[r1,#0x14]      ;ulozeni do GPIOC_ODR
                LDR r3,=0xAFE          ;dekadicky 2814~2Volt
                STR r3,[r0,#0x08]      ;nahrani do registru DAC_DHR12R1

test_tlac_off   ;test tlacitko
                LDR r3,[r2,#0x10]      ;nact. stavu vst. brany GPIOA_IDR do r3
                MOVS r4,#0x1           ;priprava masky
                ANDS r3,r4             ;vymaskovani hodn. pro PA0 kde je tlacitko
                BEQ tlac_off           ;skok pokud nebude tacitko stisknute
                B test_tlac_off
```

Výpočet výstupní hodnoty napětí

Výpočet hodnoty výstupního napětí DAC periferie pro 12-ti bitové rozlišení

Použité vzorce:

$$DAC_{output} = V_{DAA} \cdot \frac{DOR}{4085} \qquad (3.6.1)$$

DAC_{output} .....výstupní napětí DAC převodníku

V_{DAA} .....napájecí napětí perriferie

DOR .....obsah registru DAC_DOR1(kopie DAC_DHR12R1)

$$DAC\_DHR12R1 = DAC_{output} \cdot \frac{4085}{V_{DAA}} \qquad (3.6.2)$$

Výpočet:

V_{DAA} .....cca 2,9V (3,3V stabilizátor- 0,4V úbytek na schottky diodě).
DAC_{output} 1V

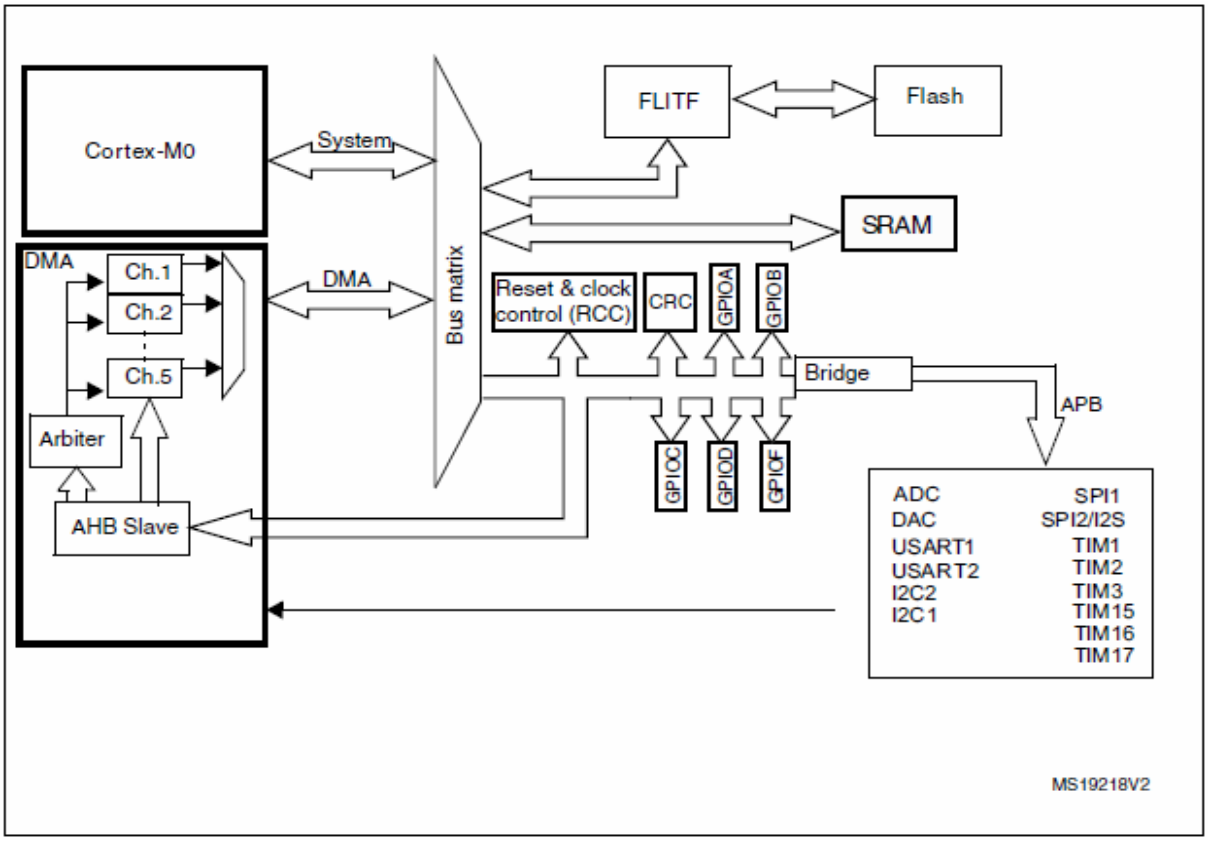
$$\boxed{DAC\_DHR12R1 = 1 \cdot \frac{4085}{2{,}9} \approx 1409} \qquad (3.6.3)$$

Pro výstupní napětí DAC převodníku by měla být do registru DAC_DHR12R1 nastavena hodnota 1409dec=0x581hex.

3.7 Příklad 7 „DMA“ v ASM

DMA – úvod

Další velice užitečnou periferií implementovanou výrobcem do obvodu STM32F051 je periferie přímého přístupu k paměti DMA. Tuto periferii je možné použít ve spojení s ADC převodníkem pro rychlý zápis dat do paměti, s DAC převodníkem v roli rychlého přístupu k řídícím datům, ve spojení s časovači TIM 1, 2, 3, 15, 16, 17, s rozhranním USART a také s dalšími periferiemi obvodu včetně pamětí MCU. Přehledné zobrazení začlenění a role DMA ve struktuře procesoru zachycuje obr. 3.7.1.



Obr. 3.7.1 DMA block diagram zdroj [1], str. 145.

Přístup DMA rozhranní k periferiím

V mikrokontroléru existují dle obr. 3.7.1 dva subjekty vystupující v roli mastera. Jsou to jádro „Cortex-M0“ a „DMA“. Oba subjekty sdílejí přístup k periferiím pomocí systémové sběrnice. DMA controler může zastavit přístup CPU k paměti nebo periferii v případě že potřebuje přistup ke stejnému zdroji. Přidělování sběrnice pak probíhá dle metody round-robin. Tento algoritmus obecně přiřadí každému procesu určité kvantum času. V STM32F0 má být CPU zajištěna nejméně polovina přenosového pásma sběrnice.

Základní vlastnosti rozhranní DMA

Níže jsou uvedeny vlastnosti DMA periferie

- 5 nezávislých kanálů s nastavitelnými čtyřmi prioritami

- spuštění přenosu softwarem nebo periferií

- nastavení velikosti paměti zdroje, cíle přenosu na 8, 16, 32 bitů

- podpora realizace kruhové paměti

- možné DMA přenosy
  - memory to memory
  - memory to periphery
  - periphery to memory
  - periphery to periphery

- programovatelný počet dat určených k přenosu až do max. 65536

- přístup k periferiím dle obr. 3.7.1

- eventy o stavu transakce Half transfer, Transfer complete, Transfer error

Funkce programu

Ukázkový program „07_DMA“ realizuje stavový automat s dvěma stavy. První stav je uvedený návěstím „stav_tlac_off“, druhý návěstím „stav_tlac_on“. Podmínkou přechodu do stavu „stav_tlac_on“ je stisknutí tlačítka připojeného k pinu PA0 MCU. Rozepnutím tlačítka program přejde do stavu„stav_tlac_off“.

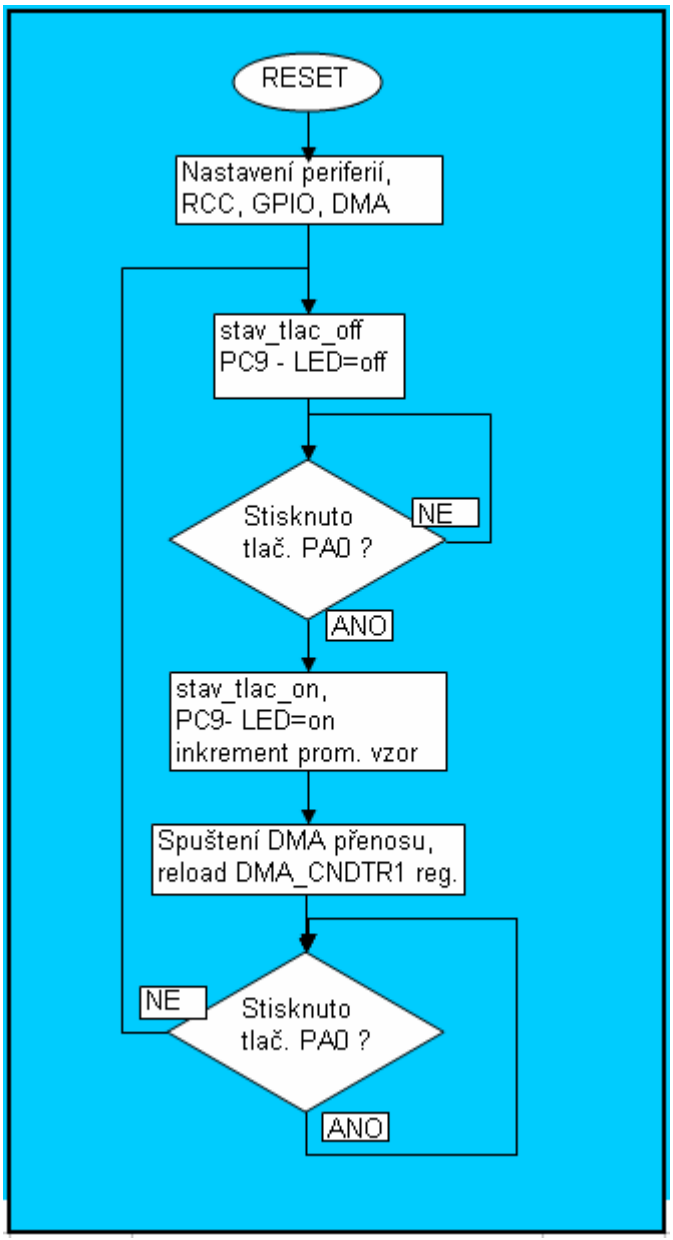
Testování tlačítka je prováděno ve dvou smyčkách označených návěstími „test_on_tlacitko“ nebo „test_off_tlacitko“. Pokud je program ve stavu „stav_tlac_off“ používá se testovací smyčka „test_on_tlacitko“. Ve stavu „stav_tlac_on“ je použita smyčka „test_off_tlacitko“.

V bloku „stav_tlac_off“ je zhasnuta LED dioda připojená k pinu PC9.

Naopak v části programu „stav_tlac_on“ je LED dioda rozsvícena a hodnota proměnné „vzor“ v paměti SRAM je inkrementována o jedničku. Dále se provede skok do podprogramu „DMA_prenos_start“, který provede zablokování periferie DMA, zápis počtu dat pro transfer a povolení periferie. Tímto krokem se spustí kopírování hodnoty proměnné „vzor“ do paměti SRAM s počáteční adresou oblasti „storage“. Transfer se opakuje dle rozsahu zadaném v registru DMA_CNDTR1, za současné inkrementace adresy pro ukládání. Výsledkem operace je naplnění paměti v délce odpovídající násobku hodnoty registru DMA_CNDTR1 a velikosti ukládané proměnné. Velikost proměnně je definována v registru DMA_CCR1 bity Msize (memory size) na 16 bitů.

Po uvolnění tlačítka se program vrátí zpět do stavu „stav_tlac_off“ a opět čeká na stisk tlačítka.

Celý tento mechanizmus je přehledně znázorněn ve vývojovém diagramu obr. 3.7.2. Přepisování paměti můžeme pozorovat prostřednictvím debuggeru programu uVision4 v okně Memory od adresy 0x20000004.



Obr. 3.7.2 Vývojový diagram zdroj programu 07_DMA.

Nastavení periferie DMA

Periferie DMA je univerzální samostatnou jednotkou mající přístup téměř ke všem periferiím mikrokontroléru. K dispozici je několik možností nastavení popsaných v manuálu [1] od str. 144 do str. 158.

Nyní si vysvětlíme nastavení periferie použité pro tento příklad.

V prvním nastavení zvolíme jako zdrojový registr DMA_CPAR1, do kterého uložíme počáteční adresu místa odkud budeme data číst. V tomto příkladu byla jako zdroj dat definována adresa předem alokovaného prostoru proměnné s názvem „vzor“.

Config_DMA

LDR r0,=DMA_base

;periphery

```
LDR r2,=vzor        ;adresa odkud budu data scanovat
STR r2,[r0,#0x10]   ;uloz r1 do  DMA_CPAR1
```

Dále je nutné definovat počáteční adresu, pro uložení dat. Pro tuto adresu použijeme registr DMA_CMAR1.

```
;memory
LDR r2,=storage     ;adresa od ktere budu ukladat
STR r2,[r0,#0x14]   ;uloz r1 do  DMA_CMAR1
```

V dalším kroku upravíme konfigurační registr DMA_CCR1. Nastavíme délku přenášených dat z periferie na 16 bitů a délku oblasti, do které budeme zapisovat také na 16 bitů. Nastavíme inkrementování adresy v cílové oblasti, to znamená, že po každém transferu bude inkrementován ukazatel adresy pro zápis dat. Kolik inkrementací adresy bude provedeno nastavíme v podprogramu „DMA_prenos_start“ do registru DMA_CNDTR1. Zdrojovou adresu ponecháme bez inkrementace.

Nastavením bitu 14 MEM2MEM definujeme směr přenosu z paměti do paměti. V tomto případě manuál [1], str.147, odstavec „Memory-to-memory mode“ zakazuje současné použití voby „Circular mode“ bit 5 CICR registru DMA_CCR1 , tj. reload registru DMA_CNDTR1 a tím neustálý transfer, kopírování nových dat do paměti. Z důvodu udržení aktuální kopie proměnné „vzor“ byl tedy přidán podprogram „DMA_prenos_start“, který je volán po změně hodnoty proměnné „vzor“. Podprogram „DMA_prenos_start“ je popsán v kapitole 5.

```
;rezim DMA
LDR r2,=0x580       ;memory and periph. size 16, memory inc. mode,
LDR r1,=0x4000      ;maska DMA memory to memory
ORRS r2,r1
STR r2,[r0,#0x08]   ;uloz r1 do  DMA_CCR1
```

Funkce podprogramu

Podprogram slouží k aktualizaci změny dat kopírovaných do paměti z proměnné „vzor“. Data jsou kopírovány do prostoru s počáteční adresou „storage“ v délce závislé na obsahu registru DMA_CNDTR1.

V podprogramu se nejprve zablokujeme periferii DMA nastavením bitu 0 registru DMA_CCR do stavu 0. Následně provedeme znovu-nahrání délky počtu dat do registru DMA_CNDTR1. Nakonec se periferie uvolní a tím dojde samostatně, nezávisle na CPU, k transferu dat podle zadaných kritérií do definované cílové oblasti. Obsah registru DMA_CNDTR1 je po každém provedeném transferu dekrementován, adresa cílové adresy je po každém přenosu inkrementována až do nulové hodnoty registru DMA_CNDTR1, tím je přenos dat ukončen.

Kód podprogramu:

```
DMA_prenos_start    ;znovu naplnení registru DMA_CCR1
        LDR r4,[r3,#0x08]   ;nahraj do r4 DMA_CCR1
        LDR r5,=0xFFFFFFFE  ;maska DMA kanal disable
        ANDS r4,r5
        STR r4,[r3,#0x08]   ;uloz r4 do  DMA_CCR1
```

```
            LDR r4,=100             ;mnozstvi dat  ktere bude preneseno do pameti
                                    delka zavisí na nastaveni DMA
            STR r4,[r3,#0x0C]       ;uloz r4 do  DMA_CNDTR1

            LDR r4,[r3,#0x08]       ;nahraj do r4 DMA_CCR1
            LDR r5,=0x1             ;maska DMA kanal disable
            ORRS r4,r5
            STR r4,[r3,#0x08]       ;uloz r4 do  DMA_CCR1

            BX LR
```

Popis výkonné části programu

Princip programu byl podrobně uveden v kapitole 3.1. Následuje výpis kódu realizující popsanou funkci hlavního programu příkladu „07_DMA“.

```
            LDR r0,=vzor
            LDR r1,=GPIOC_base
            LDR r2,=GPIOA_base
            LDR r3,=DMA_base

;**********************************part_1************************************
nuluj_vzor
            LDR r4,=0x0
            STR r4,[r0]

stav_tlac_off
            ;zhasnuti LED PC9
            LDR r4,[r1,#0x14]   ;nahrani stavu registru GPIOC
            LDR r5,=0xfffffdff  ;maska LED na PC9_OFF
            ANDS r4,r5
            STR r4,[r1,#0x14]   ;ulozeni do GPIOC_ODR, zhasnuti LED na PC9

test_on_tlacitko
            ;test tlacitko
            LDR r4,[r2,#0x10]   ;nacteni stavu vstupu brany GPIOA_IDR do r3
            MOVS r5,#0x1        ;priprava masky
            ANDS r4,r5          ;vymaskovani hodn. PA0 kde je umisteno tlacitko
            BNE stav_tlac_on    ;skok pokud bude tacitko stisknute
            B test_on_tlacitko  ;test loop


;**********************************part_2************************************
stav_tlac_on
            ;rozsviceni LED
            LDR r4,[r1,#0x14]       ;nahrani stavu registru GPIOC
            LDR r5,=0x200           ;maska LED na PC9_ON
            ORRS r4,r5
```

```
                STR r4,[r1,#0x14]       ;ulozeni do GPIOC_ODR, tj. rozsv.  LED na
                                        PC9
                ;inkrement promene vzor
                LDR r5,[r0]             ;nahrani aktualni hodnoty promene "vzor"
                ADDS r5,#0x01           ;inkrementace hodnoty
                STR  r5,[r0]            ;nahrani zvysene hodnoty do promené vzor

                BL DMA_prenos_start ;skok do podprogramu

test_off_tlacitko                       ;test tlacitko
                LDR r4,[r2,#0x10]       ;nacteni stavu vstupu brany GPIOA_IDR do r3
                MOVS r5,#0x1            ;priprava masky
                ANDS r4,r5              ;vymaskovani hodnoty pro PA0 kde je tlactko
                BEQ stav_tlac_off       ;skok pokud nebude tacitko stisknute

                B test_off_tlacitko

DMA_prenos_start                        ;znovunaplnení registru DMA_CCR1
                LDR r4,[r3,#0x08]       ;nahraj do r4 DMA_CCR1
                LDR r5,=0xFFFFFFFE      ;maska DMA kanal disable
                ANDS r4,r5
                STR r4,[r3,#0x08]       ;uloz r4 do  DMA_CCR1

                LDR r4,=0x32            ;mnozstvi dat  ktere bude preneseno DMA
                                        (word)
                STR r4,[r3,#0x0C]       ;uloz r4 do  DMA_CNDTR1

                LDR r4,[r3,#0x08]       ;nahraj do r4 DMA_CCR1
                LDR r5,=0x1             ;maska DMA kanal disable
                ORRS r4,r5
                STR r4,[r3,#0x08]       ;uloz r4 do  DMA_CCR1

                BX LR
;*****************************data alokace*********************************

                AREA uloziste,DATA,READWRITE
vzor            DCD  0                  ;alokace prostoru pro prom.  urcenou pro kopirovani
storage SPACE 100                       ;alokace prostoru pro ulozeni dat z ADC v Bytech
```

4. Voltmetr - měření napětí

Abychom mohli v elektrotechnice porovnávat a charakterizovat účinky elektrického potenciálu, byly zavedeny níže uvedené názvy pro měřené hodnoty napětí.

Program v mikrokontroléru tyto hodnoty vypočítává podle níže uvedených vztahů. Vstupními daty jsou naměřené vzorky z ADC převodníku ve 12bitovém rozlišení.

4.1 Definiční vztahy pro výpočty hodnot napětí

Maximální hodnota napětí

Je hodnota největšího prvku z omezené množiny vzorků. $U_{\max} = \max(u(t))$ (4.1)

$t \in \{t_1, ...t_n\}, n \in Z.$ n je počet vzorků pro výpočet hodnoty.

Minimální hodnota napětí

Je hodnota nejmenšího prvku z omezené množiny vzorků. $U_{\min} = \min(u(t))$ (4.2)

$t \in \{t_1, ...t_n\}, n \in Z.$ n je počet vzorků pro výpočet hodnoty.

Střední aritmetická hodnota napětí

Střední aritmetická hodnota definovaná vztahem $U_{as} = \frac{1}{T}\int_{t_1}^{t_2} |u(t)| dt$ (4.3)

Tento údaj je možné získat přístrojem který používá dvoucestné usměrnění vstupního napětí viz [8], str. 158.

Příklad výpočtu pro funkci sin:

Definice funkce vstupního napětí $u(t) = U_m \cdot \sin(\omega t)$ (4.4)

$$\frac{1}{T}\int_{t_1}^{t_2} |u(t)|\, dt = \frac{1}{T/2} \cdot \int_0^{T/2} U_m \cdot \sin(\omega t) dt = \frac{2 \cdot U_m}{T} \int_0^{T/2} \sin(\omega t) dt = \frac{2 \cdot U_m}{T} \cdot \left[-\frac{1}{2 \cdot \pi \cdot f} \cos\left(2 \cdot \pi \cdot f \cdot \frac{T}{2}\right) + \frac{T}{2 \cdot \pi} \cos(0) \right] =$$

$$= \frac{2 \cdot U_m}{T} \cdot \left[-\frac{T}{2 \cdot \pi} \cos\left(2 \cdot \pi \cdot \frac{1}{T} \cdot \frac{T}{2}\right) + \frac{T}{2 \cdot \pi} \right] = \frac{U_m}{1} \cdot \left[-\frac{1}{\pi} \cos(\pi) + \frac{1}{\pi} \right] = \frac{U_m}{1} \cdot \left[\frac{1}{\pi} + \frac{1}{\pi} \right] = \frac{2 \cdot U_m}{\pi}$$

Elektrolytická střední hodnota napětí

Definiční vztah $U_{es} = \frac{1}{T}\int_{t_1}^{t_2} u(t)\,dt$ (4.5)

, odpovídá usměrnění řízeným usměrňovačem viz zdroj [8], str. 158, 161.

$$\frac{1}{T}\cdot\int_0^T u(t)\,dt = \frac{1}{T}\cdot\int_0^T U_m\cdot\sin(\omega t)\,dt = \frac{U_m}{T}\cdot\left[-\frac{2\cdot\pi}{T}\cos\left(2\cdot\pi\cdot\frac{1}{T}\cdot t\right)+\frac{2\cdot\pi}{T}\cos\left(2\cdot\pi\cdot\frac{1}{T}\cdot t\right)\right]_0^T =$$
$$= \frac{U_m}{T}\cdot\left(-\frac{2\cdot\pi}{T}+\frac{2\cdot\pi}{T}\right) = 0$$

V případě stejné plochy od 0 do π a od π do 2π vychází střední aritmetická hodnota nulová.

Efektivní hodnota napětí

Efektivní hodnota je hodnotou, která je v praxi nejvíce používáná pro hodnocení velikosti střídavého napětí. Efektivní hodnota střídavého napětí odpovídá ekvivalentní hodnotě práce, kterou by vykonala stejná velikost stejnosměrného napětí. Jednodušší multimetry a měřící přístroje používají pro její výpočet vztahů viz tab. 4.1 mezi maximální nebo střední aritmetickou hodnotou a efektivní hodnotou funkce sinus. V laboratorních přístrojích se můžeme setkat s přístroji pracujícími s termočlánky měřícími přímo tepelné účinky odvozené z měřeného napětí (bližší popis např. v [8], str.166).

Přístroje měřící efektivní hodnotu dle definičního vztahu (4.6) bývají označovány jako RMS, nebo TRMS (True Root Mean Square).

$$U_{ef} = \sqrt{\frac{1}{T_P}\int_{t_0}^{t_0+t_P} u^2(t)\,dt} \tag{4.6}$$

Výpočet efektivní hodnoty dle vzorce (4.6) pro fci sinus

Definice funkce $u(t) = U_m\cdot\sin(\omega t)$ (4.7)

$$\int_0^T u^2(t)\,dt = \int_0^T (U_m\cdot\sin(\omega t))^2\,dt = U_m^2\int_0^T (\sin(\omega t))^2\,dt = \left|(\sin(\omega t))^2 = \frac{1-\cos(2\omega t)}{2}\right| = \frac{U_m^2}{2}\int_0^T (1-\cos(2\omega t))\,dt =$$
$$= \frac{U_m^2}{2}\left([t]_0^T - \left[\frac{1}{2\omega}\cdot\sin(2\omega t)\right]_0^T\right) = \frac{U_m^2}{2}\left(T - \frac{1}{2\omega}\cdot\sin(2\omega T) - \frac{1}{2\omega}\cdot\sin(0)\right) = \frac{U_m^2}{2}\cdot T \tag{4.8}$$

$$\sqrt{\frac{1}{T_P}\int_{t_0}^{t_0+t_P} u^2(t)\,dt} = \sqrt{\frac{1}{T}\cdot\frac{U_m^2}{2}\cdot T} = \frac{U_m}{\sqrt{2}}$$

Tab. 4.1 Vztahy mezi střední, maximální a efektivní hodnotou zdroj [8], str165.

Činitel výkyvu	$k_v = \frac{U_{ef}}{U_{str}}$
Činitel tvaru	$k_t = \frac{U_m}{U_{ef}}$
Činitel plnění	$k_p = \frac{U_m}{U_{str}}$

Funkce	Činitel tvaru	Činitel výkyvu
Sinus	$\frac{\pi}{2\sqrt{2}}$	$\sqrt{2}$
Trojúhelník	$\frac{2}{\sqrt{3}}$	$\sqrt{3}$
Obdelník	1	1

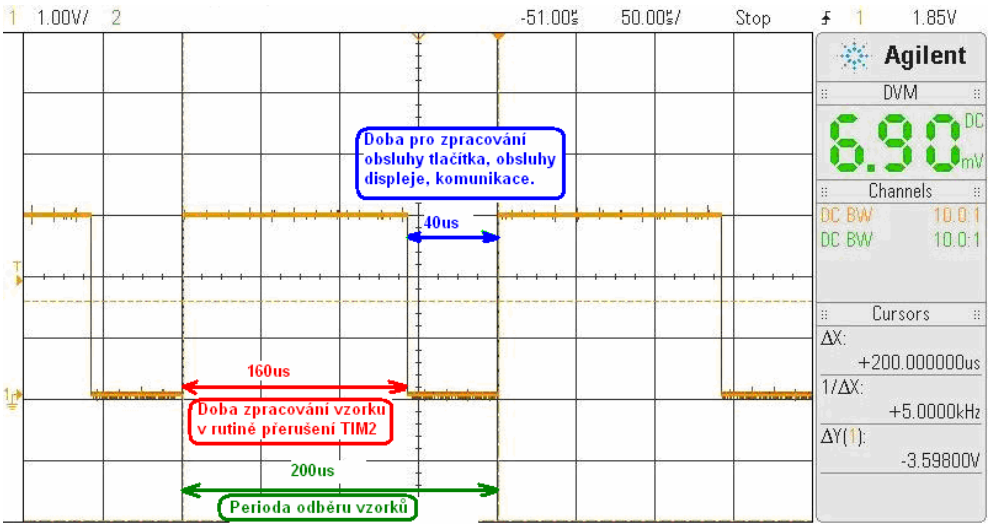
4.2 Způsob řešení přístroje voltmetr

Pro realizaci tohoto zařízení jsem použil vývojový kit STM32F0-Discovery, osazený mikrokontrolérem ARM STM32F051R8. V programu byly použity jeho vestavěné periferie převodník ADC a jednotka časovače TIM2.

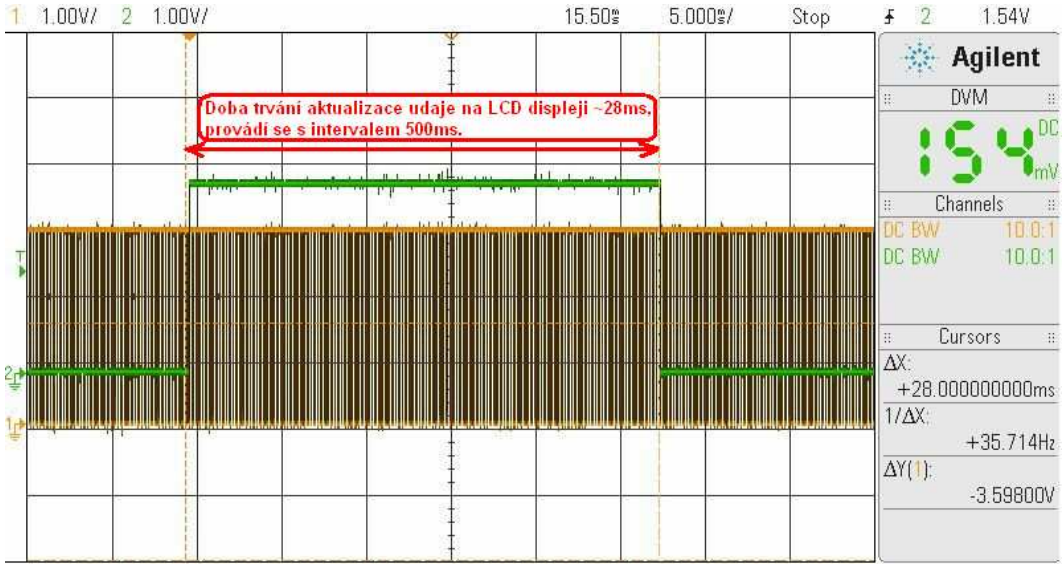
Jako první řešení přístroje mne napadlo provést 8bitové vzorkování maximální rychlostí ADC převodníku. Po načtení určitého počtu pulsů zastavit odběr vzorků, provést vyhodnocení načteného průběhu a zobrazit výsledek na LCD displeji. Tato varianta měla výhodu v možnosti použití vyššího kmitočtu vzorkování (během odběrů neprovádí se výpočet), ale také nevýhodu nespojitého odběru vzorků ze vstupního signálu.

Druhá varianta voltmetru používá průběžný odběr vzorků s vzorkovací periodou 200 us. Po každém odběru je upravena hodnota výsledku měření. Trvání zpracování naměřených hodnot, byla pomocí osciloskopu změřena v délce 140 us (viz situace obr. 4.1). Vzorkovací perioda byla tedy zvolena s dostatečnou rezervou 200 us. Během zbývajících 200 us - 40 us tj. 60 us (volný čas mezi odběry vzorků), jsou procesorem prováděny běžné časově nekritické činnosti jako obsluha tlačítek, zobrazování dat na displeji apod. Tato činnost je graficky zachycena na obrázku obr. 4.1.

Pro zobrazení naměřené hodnoty byl použit LCD display, Obnovovací perioda zápisu naměřeného údaje na display byla nastavena na 500 ms,



Obr. 4.1. Doby vzorkování s vykonáváním zpracování dat programem.



Obr. 4.2 Měření doby provádění aktualizace displeje(zelený průběh). Žlutý průběh, vzorkování vstupu voltmetru .

4.3 Parametry ADC převodníku

Doba převodu jednoho kanálu ADC převodníku mikrokontroléru STM32F0xx je závislá na použitém rozlišení převodníku a na čase nutném pro konverzi vzorkované hodnoty. Výpočet pro určení této doby je uveden níže. Tato hodnota určuje maximální vzorkovací kmitočet převodníku.

$$t_{ADC} = t_{SMPL} + t_{SAR}, \text{ kde} \tag{4.9}$$

t_{ADC} je doba získání a konverze ADC převodníku jednoho kanálu
t_{SMPL} je doba vzorkování jednoho kanálu
t_{SAR} je doba konverze vzorku jednoho kanálu

$$t_{SMPL} = 1{,}5 \cdot t_{ADC\_CLK} = 13{,}5 \cdot \frac{1}{12 \cdot 10^6} = 1{,}125\ \mu s \tag{4.10}$$

$$t_{SMPL} = 12{,}5_{12bit} \cdot t_{ADC\_CLK} = 12{,}5 \cdot \frac{1}{12 \cdot 10^6} = 1{,}042\ \mu s \tag{4.11}$$

Doba převodu ADC převodníku t_{ADC} pro rozlišení 12 bitů.

$$t_{ADC} = (1{,}125 + 1.042)\,\mu s = 2{,}166\ \mu s \tag{4.12}$$

Maximální frekvence převodu ADC převodníku pro 12bitové rozlišení

$$f_{\max} = \frac{1}{t_{ADC}} = \frac{1}{2{,}166 \cdot 10^{-6}} = 461680\ Hz \tag{4.13}$$

Kvantizační krok ADC převodníku

Kvantizační krok je rozdíl mezi dvěmi sousedními úrovněmi napětí, které dokáže ADC převodník rozlišit. Hodnota je závislá na použitém rozlišení převodníku. Pro 12bitové rozlišení odpovídá hodnotě dle výpočtu (4.14).

$$t_{ADC} = 3V / 4096 = 0{,}732\ mV \tag{4.14}$$

Typické chyby ADC převodníku zdroj datasheet [2], str.77, tab.56 pro teplotu 25°C a napájecí napětí v rozsahu 2,4 V až 3,6 V.

Celková chyba	± 3,3 (max. ±4)	LSB
Chyba offsetu	± 1,9 (max. ±2,8)	LSB
Chyba zesílení	± 2,8 (max. ±3)	LSB
Diferenciální nelinearita	± 0,7 (max. ±1,3)	LSB
Integrální nelinearita	± 1,2 (max. ±1,7)	LSB

4.4 Hlavní funkce a konstanty programu

Funkce vybraných programových funkčních modulů

stm32f0xx_LCD.c - poskytuje funkce pro výpis dat pomocí dvou-řádkového 16znakového LCD displeje. Funkce byla převzata a upravena.

obsluha_rozhrani_HMI.c - obsluha ovládaní přístroje uživatelem, menu přístroje.

stm32f0xx_voltmetr.c- blok funkcí zpracovávající naměřené vzorky do dílčích hodnot.

include "stm32f0xx_my_config.c - pbsahuje funkce volané při počáteční konfiguraci periferií.

stm32f0xx_it.c – funkce pro obsluhu přerušení především pro obsluhu události TIM2 „uppdate event“, obsluhu události „systick“ umožňující realizaci funkce delay.

Nastavitelné konstanty programu

V závorce jsou vypsány hodnoty příslušející těmto konstantám, které jsou nastavené v programu.

POCET_VZORKU - (1000) velikost pole, počet vzorku v kruhovém zásobníku pro naměřené vzorky.

ROZL_ADC - (4096) rozlišení ADC převodníku úrovní.

REFRESH_DISPLAY - (5000) počet vzorků po jejichž načtení bude obnovena hodnota dat na displeji.

REF_NAPETI - (2,97) hodnota napětí odpovídající maximálnímu hodnotě ADC převodníku.

4.5 Konfigurace periferií MCU

Nastavení časovače TIM2

Jádrem programu je časovač TIM2, který spouští odběr vzorku signálu ze vstupního pinu PC4. Nastavení časové základny tohoto čítače je provedeno v souboru „stm32f0xx_my_config.h“ na periodu 1 us. V registru „TIM2->ARR“ je nastaven počet pulsů, při kterém dojde k znovu-nastavení registru TIM2->CNT“ na počáteční hodnotu 0. Do registru „TIM2->ARR“ se zadává hodnota snížená o číslici 1. Pokud tedy chceme, aby časovač periodicky počítal 200 pulsů jeho časové základny, zadáme do registru „TIM2->ARR“ hodnotu 199.

Časovač TIM2 generuje spouštěcí signál tzv. trigger, pro ADC periferii. Nastavení tohoto režimu je provedeno zapsáním MMS bitů v registru TIM2->CR2 na hodnotu 010. Tímto se

při každé události „uppdate event“ časovače vygeneruje z časovače TIM2 signál TRGO (trigger output).

Nastavení ADC převodníku

Na druhé straně je nutné signál TRGO z periferie TIM2 připojit v periferii ADC převodníku na startovací pin pro provedení ADC převodu. V konfiguraci ADC převodníku je nutné zapsat bity EXTSEL registru ADC_CFGR1 na hodnotu 000, tzn. spouštění převodu je mapováno na TRGO. Dále je nutné ve stejném registru specifikovat událost na základě které dojde k převodu, tzn. nastavit bity EXTEN např. na hodnotu 01 (hardwarové spouštění na náběžnou hranu).

ADC periferie je nastavena na maximální rozlišení 12 bitů. Je použit kanál 14 převodníku se vzorkovacím časem 1 vzorek na 5 cyklů hodin ADC převodníku. Pro časování tohoto převodníku je použit zdroj hodin PCLK, hodnota kmitočtu hodin pro ADC převodník je dělením hodnotou 4, upravena na 12 MHz. Časovač mikrokontroléru používá stejný zdroj hodin PCLK, tím odpadá nejistota okamžiku převodu (jitter) hodnoty ADC v případě spouštění převodu časovačem TIM2. Problematika je podrobněji popsána v manuálu RM0091 viz zdroj [1], str. 175.

Nastavení SYSTICK

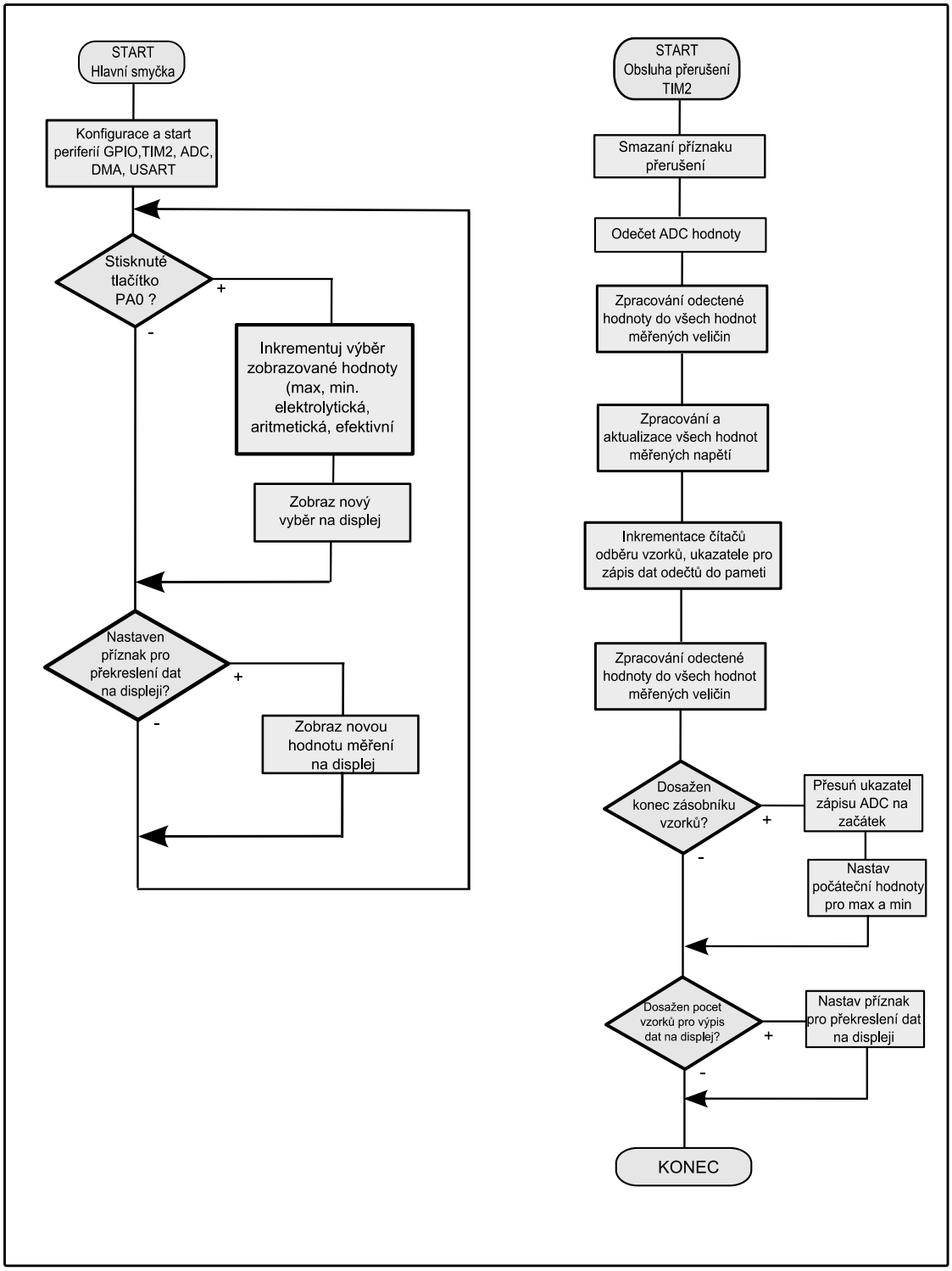
Pro potřeby realizace zpoždění v programu potřebném při obsluze displeje a realizaci filtru zákmitů při stisku tlačítka byla nadefinována periferie SYSTICK, která je schopna periodicky vyvolávat přerušení odvozené od nastavitelného počtu pulsů systémových hodin MCU. Perioda volání přerušení SYSTICK byla nastavena na 1 ms.

Periferie NVIC

Pomocí periferie NVIC je nastavena nejvyšší priorita přerušení časovači TIM2, tím je zajištěno vzorkování signálu v ekvidistantních intervalech závislých na vyvolání přerušení periferií TIM2. Periferie SYSTICK má nastavenu nižší prioritu než TIM2.

4.6 Vývojový diagram programu

Program obsahuje hlavní smyčku, z které je periodicky voláno přerušení periferií TIM2. Vývojový diagram s detaily činnosti programu popisuje obr. 4.3.



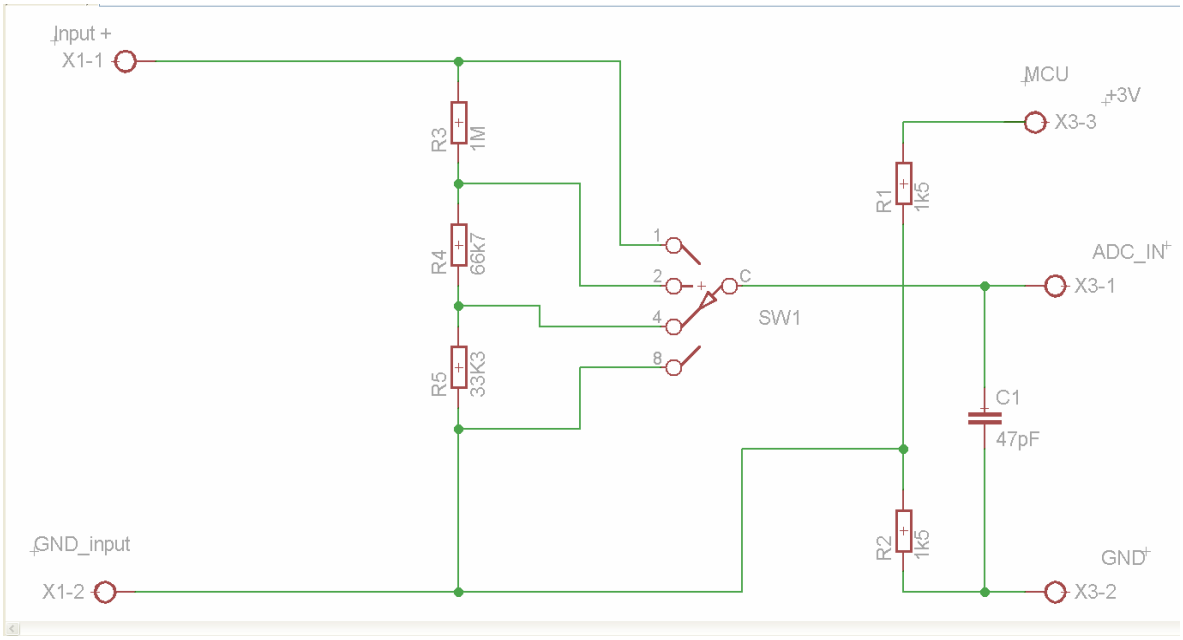
Obr. 4.3 Vývojový diagram programu přístroje Voltmetr.

4.7 Použití voltmetru pro vyšší rozsahy

Základní rozsah vstupu ADC převodníku popsaný v manuálu [1], str. 171, udává rozmezí vstupního napětí V_{IN} vztahem $V_{SSA} \leq V_{IN} \leq V_{DDA}$. V praxi tento rozsah odpovídá nejčastěji rozmezí $0\,V \leq V_{IN} \leq 3\,V$. Pro možnost měření napětí ve vyšších úrovní napětí, byl navržen dělič viz obr. 4.4 se vstupním odporem voltmetru 1 MΩ. Rezistory R1 a R2 realizují stejnosměrný posun napětí do středu rozsahu ADC převodníku. Rezistory R3,4,5 tvoří vstupní dělič s dělícími poměry 1:1, 1:10, 1:32.

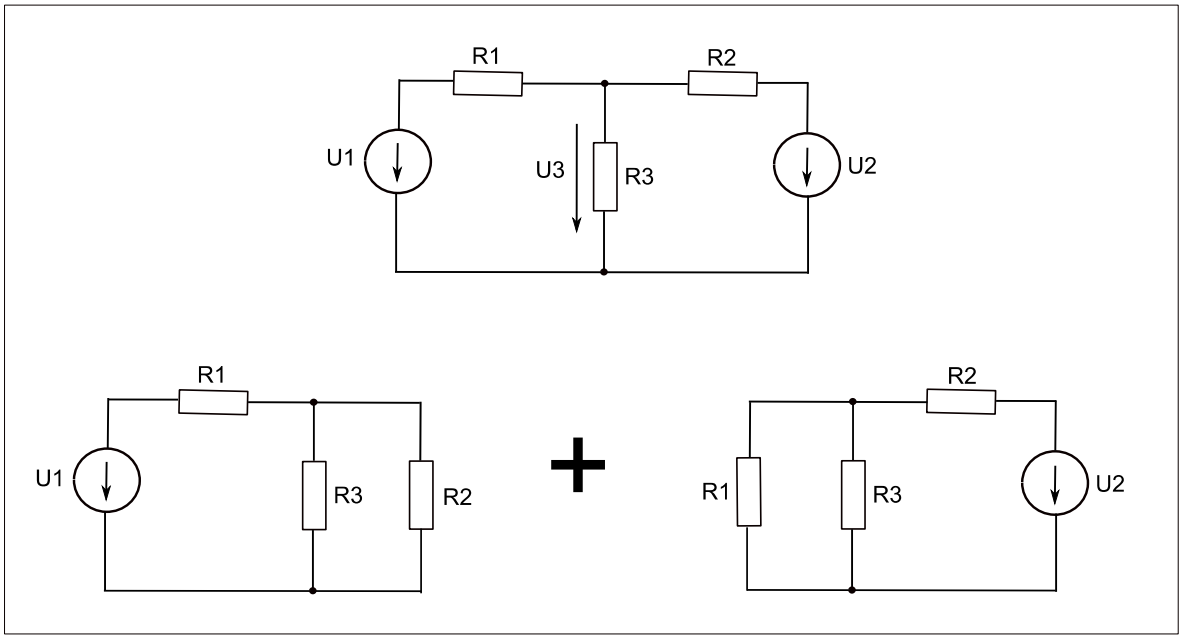
Velkou nevýhodou tohoto návrhu děliče je nesouhlasná úroveň zemních potenciálů, to může být problém, spojení potenciálů vyřadí rezistor R2, následkem toho, v případě měření střídavého napětí by mohlo dojít ke zničení ochranných diod vstupů mikrokontroléru.

Z tohoto důvodu byl navržen druhý typ děliče viz obr. 4.5 s dělícím poměrem 1:10, který má společnou zem se vstupním signálem. Nevýhodou tohoto děliče je poněkud složitější výpočet hodnot rezistorů.



Obr. 4.4 Schéma děliče napětí a posun nulové úrovně doprostřed rozsahu voltmetru.

Návrh Děliče napětí se společnou zemní svorkou



Obr. 4.4 Řešení výpočtu děliče pomocí principu superpozice .

$$I_1 = \frac{U_1}{R_1 + \frac{R_2 \cdot R_3}{R_2 + R_3}} \tag{4.15}$$

$$U_{R_{31}} = U_1 - U_{R_1} = U_1 - R_1 \cdot \frac{U_1}{R_1 + \frac{R_2 \cdot R_3}{R_2 + R_3}} = U_1 \cdot \left(1 - \frac{R_1}{R_1 + \frac{R_2 \cdot R_3}{R_2 + R_3}} \right) \tag{4.16}$$

Vliv zdroje U2,

$$I_2 = \frac{U_2}{R_2 + \frac{R_1 \cdot R_3}{R_1 + R_3}} \tag{4.17}$$

$$U_{R_{32}} = U_1 - U_{R_2} = U_2 - R_2 \cdot \frac{U_2}{R_2 + \frac{R_1 \cdot R_3}{R_1 + R_3}} = U_2 \cdot \left(1 - \frac{R_2}{R_2 + \frac{R_1 \cdot R_3}{R_1 + R_3}} \right) \tag{4.18}$$

Pro poměr dělení vstupního signálu 1:10, musí platit podmínka z rovnice (4.16)

$$\frac{R_1}{R_1 + \frac{R_2 \cdot R_3}{R_2 + R_3}} = \frac{9}{10} \tag{4.19}$$

Pro posun signálu do poloviny rozsahu, musí platit podmínka z rovnice (4.18)

$$\frac{R_2}{R_2+\frac{R_1\cdot R_3}{R_1+R_3}}=\frac{1}{2}\quad\Rightarrow\quad R_2=\frac{R_1\cdot R_3}{R_1+R_3} \tag{4.20}$$

Pro řešení jsou k dispozici dvě rovnice s podmínkami na dělící poměr a na posun napěťové úrovně signálu. Abychom dostali konkrétní řešení rovnic s ohledem na vnitřní odpor voltmetru zvolíme vnitřní odpor voltmetru a tím dostaneme třetí rovnici (4.21).

$$R_1=1\,\mathrm{M\Omega} \tag{4.21}$$

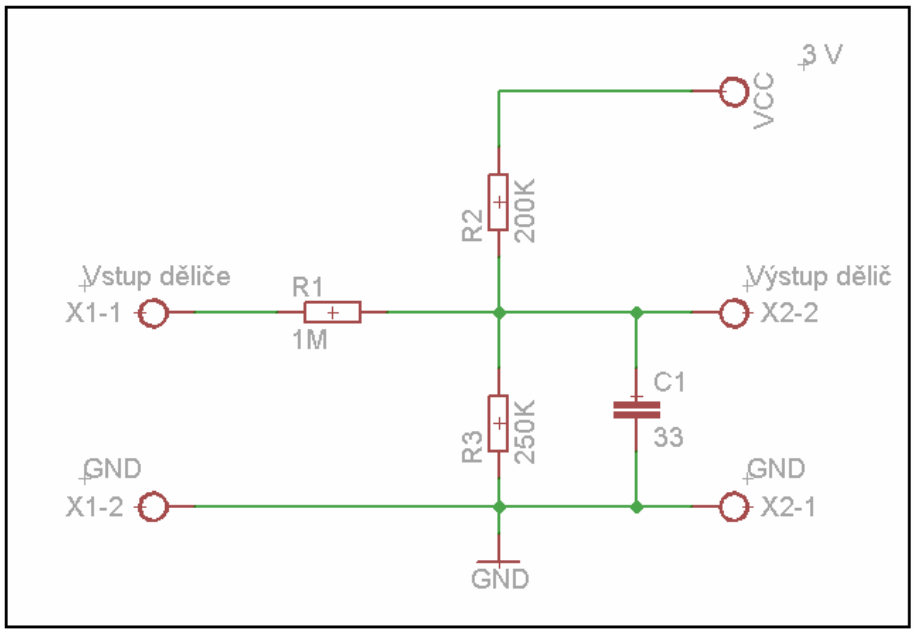
Po dosazení R3 dostaneme rovnice

$$R_3=\frac{R_2}{9\cdot R_2-1} \tag{4.22}$$

$$R_2=\frac{R_3}{1+R_3} \tag{4.23}$$

Výsledné hodnoty rezistorů děliče dle obr. 4.5 byly vypočteny jako

$R_1=1\,\mathrm{M\Omega}$, $R_2=0{,}2\,\mathrm{M\Omega}$, $R_3=0{,}25\,\mathrm{M\Omega}$



Obr. 4.5 Zapojení děliče s posunem nulové úrovně napětí doprostřed rozsahu voltmetru, zapojení se společnou zemí.

4.8 Parametry přístroje voltmetr s MCU STM32F051R8

Měřené veličiny	Maximální hodnota napětí Minimální hodnota napětí Střední elektrolytická hodnota napětí Střední aritmetická hodnota napětí Efektivní hodnota napětí TRMS
Přepínání druhu měření	tlačítko SET pin PA0
Vstupní pin pro měřené napětí	PA5
Vstupní napětí rozsahy	1/1..... ±1,5 V 1/10..... ±15 V 1/32..... ±48 V
Šířka pásma pro měření střídavých signálů	od 10 Hz do 4,5 kHz

Chyba měření hodnoty voltmetrem vzhledem ke standardu tvořeným generátorem DS-X 2012A [12] – 1,5 mV až +2,5 mV (k=1)

Voltmetr - chyby

ADC převodník voltmetru má dle datasheetu obvodu [2], v závislosti v případě 12bit rozlišení max. celkovou chybu ±3,3 LSB (LSB - nejméně významný bit), tj. přepočteno ±2,42 mV.

Měřením sinusového napětí o frekvenci 50Hz, amplitudě od 0,1Vp-p až 3Vp-p (data viz tab. 4.2 v příloze) a offsetu 1,5V generovaného osciloskopem DSO-X 2012A byla zjištěna maximální chyba údaje od hodnoty vypočtené z nastavené hodnoty Vp-p dle vztahu

$$\frac{U_m}{\sqrt{2}} \tag{4.26}$$

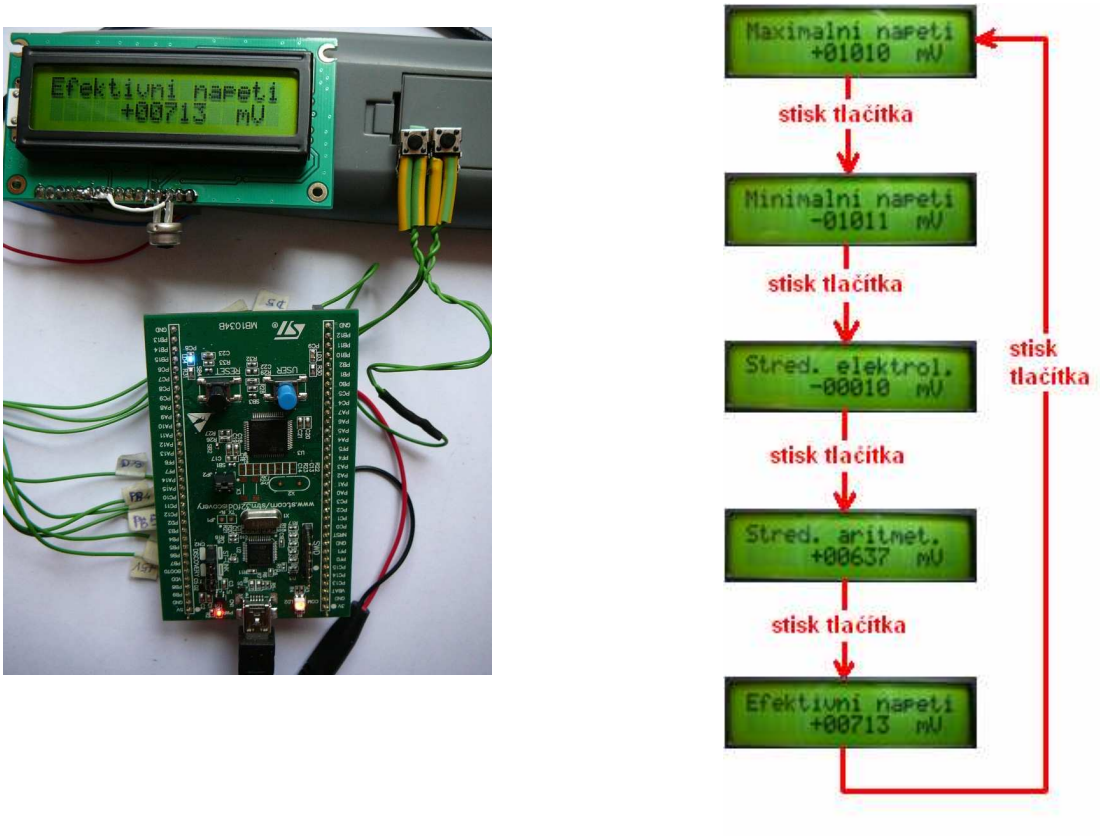
v rozsahu -1.5 mV až 2,5 mV. Tabulka naměřených hodnot byla vložena do přílohy.

Přepočet chyb na procenta

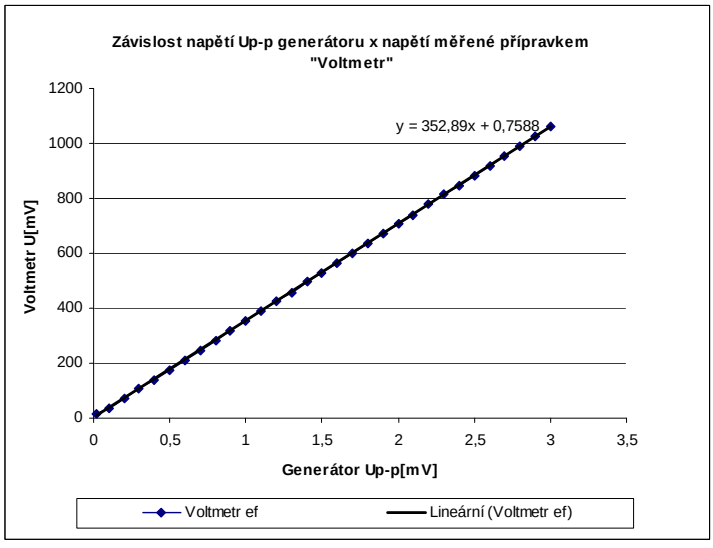
100 % odpovídá naměřené úrovni napětí 2,95 V na testovacím vzorku.

±2,5 mV odpovídá procentuelní chybě vzhledem k maximální úrovni napětí

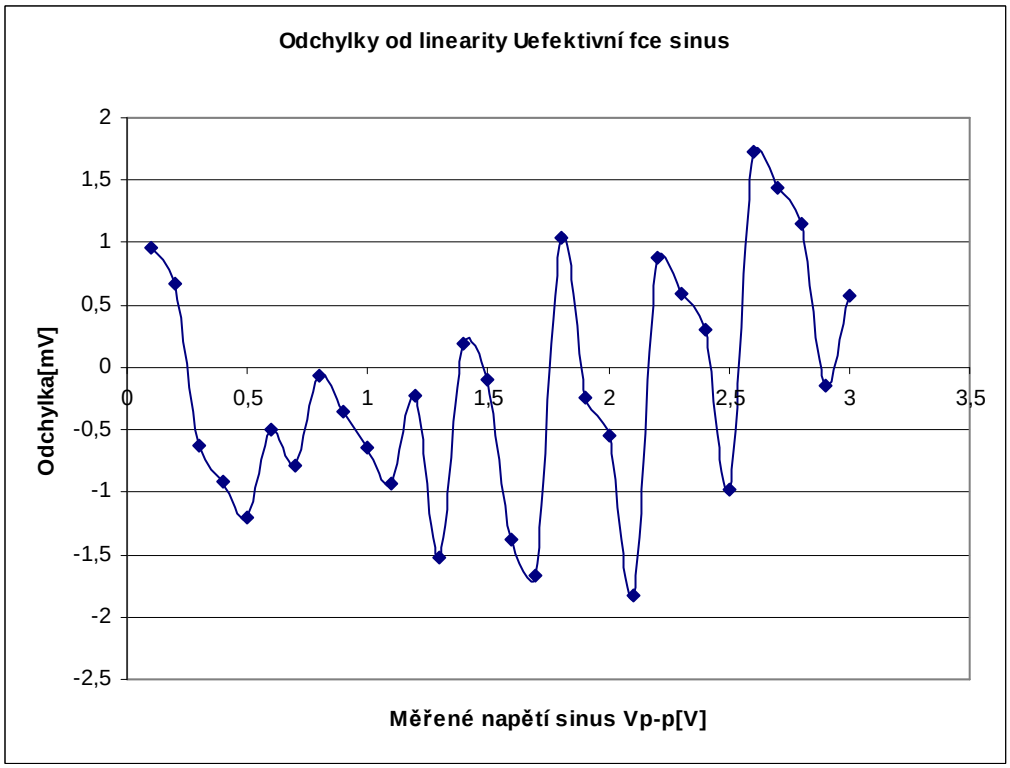
$$error = \frac{100 \cdot difference}{U_m} = \frac{100 \cdot 0{,}005}{2{,}95} = 0{,}17\% \tag{4.27}$$



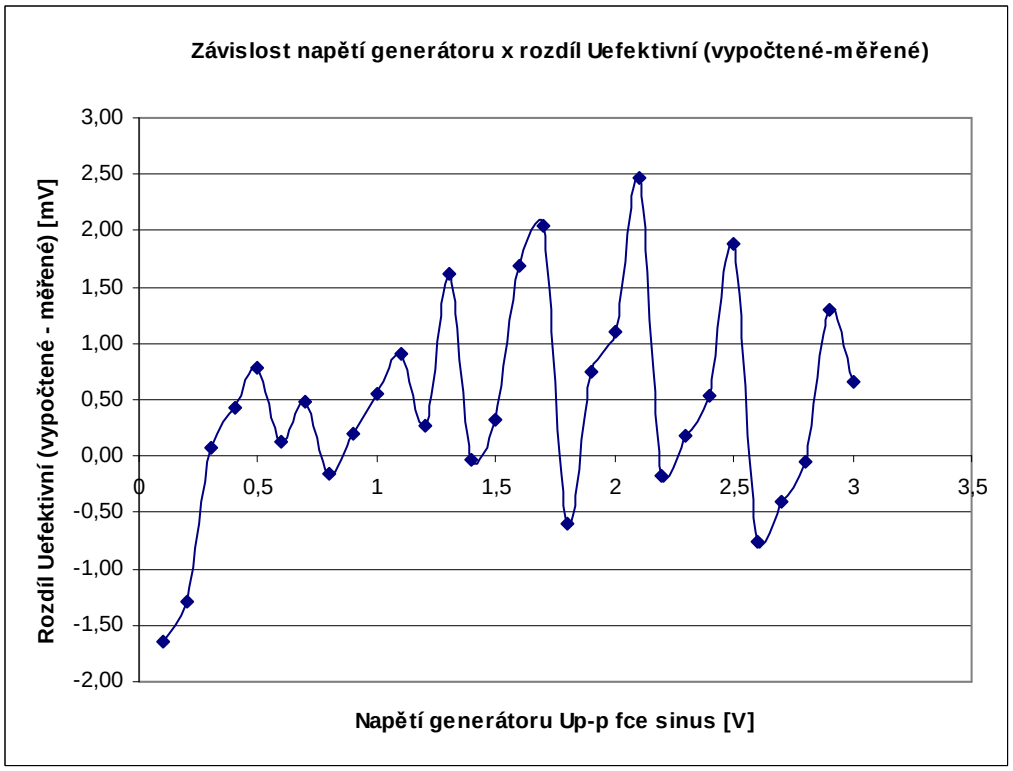
Obr. 4.5 Vlevo zapojení testování modulu voltmetr, vpravo schéma menu přístroje.



Obr. 4.6. Charakteristika závislosti napětí generátoru a měřené hodnoty voltmetrem..



Obr. 4.7 Charakteristika závislosti napětí generátoru a odchylky od linearity.



Obr. 4.8 Charakteristika závislosti napětí generátoru a rozdílu vypočtené a měřené hodnoty. Vypočtená hodnota je spočtena na základě velikosti Up-p.

4.9 Porovnání vybraných vlastností se specializovanými obvody

V měřících přístrojích nabízejících měření hodnoty skutečného efektivního napětí s označením RMS, se můžeme setkat například s obvodem AD637, výrobce Analog Devices. Tento vypočítává hodnotu efektivního napětí analogovým způsobem (logaritmický převodník efektivní hodnoty). Vysvětlení principu logaritmického převodníku je např. v [8], str.168,169. Cena obvodu se pohybuje dle specifikace od cca 12$ do 80$. Obvod nabízí dle datasheetu [10] uživateli následující vlastnosti.

Maximální nelinearita pro 0 V až 2 V rms vstup.....0,02 %
Celková chyba.....$\pm$0,1%
Aditivní chyba pro crest faktor[7] 3.....0,10 %
Šířka pásma na 2 V rms.....8 MHz
Šířka pásma na 100 mV rms.....600 kHz
Výpočet - True rms
- Square
- Mean square
- Absolute value

Výstupní pin poskytující výstupní napětí v dB v rozsahu 60 db s externí referencí pro polohu 0 dB hodnoty.

Další údaje pro porovnání obvodů je možné získat z [8], str.169. Jsou zde stručně uvedeny vlastnosti obvodů AD536, AD636, AD637, AD736. Nejvyšší uváděná aditivní chyba je max. 1%.

Z obvodů převodníků efektivní hodnoty jiného výrobce jsem pro porovnání vybral obvod LTC1966 výrobce Linear technologies. Tento obvod viz datasheet [11], pracuje na principu sigma/delta převodníku, princip vysvětlen v [8], str.219.

Výstupní napětí obvodu odpovídá vztahu (4.25) uvedeném také v [11], str.11.

$$U_{OUT} = pruměr\left(\frac{U_{in}^2}{U_{out}}\right) = RMS\left(U_{in}\right) \tag{4.25}$$

Základní parametry obvodu LTC1966
Šířka pásma pro $\pm$1 %.....6 kHz
Linearita.....0,02 %
Celková chyba od 50 Hz do 1 kHz.....0,25%
Chyba zesílení pro f_{in} od 50 Hz do 1 kHz.....0,1 %
Celková chyba pro f_{in} od 50 Hz do 1 kHz.....0,1 %

[7] crest faktor je definován jako $C = \frac{|U|_{peek}}{U_{RMS}}$, crest factor pro sinusový signál je vyjádřen jako $C = \frac{U_m}{\sqrt{2}}$, Crest factor 2 odpovídá jednocestnému usměrnění, viz zdroj [9].

Stručné vyhodnocení porovnání řešení voltmetru různými metodami

Nejvyšší chyba měření mikrokontrolérem STM32F051, byla zjištěna v maximální hodnotě 0,085 % z max. rozsahu. V porovnání s již staršími obvody AD637 a LTC1966, kapitola 4.4.8, má vyvinuté řešení nevýhodu v nižší šířce pásma a horší nelinearitě, která je dána odlišnými principy převodu měřené hodnoty.

Řešení s mikrokontrolérem má hlavní výhodu v možné variabilitě, možnosti přidávání dalších vlastností, možnosti přímého zobrazování například na display.

5. Měření frekvence a periody přímou metodou

5.1 Rozdíl mezi časovačem a čítačem

Jak již bylo zmíněno v kapitole 2. pro číslicové měření kmitočtu klasickou metodou používáme čítače.

Nejprve se pokusím vysvětlit co to vlastně je čítač a co má společného s časovačem.

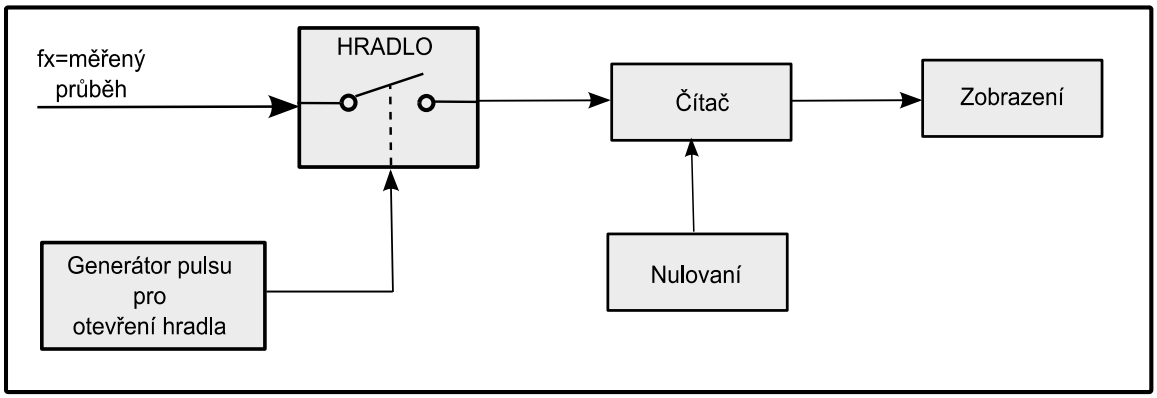
V dokumentaci obvodu [1], se setkáme s jednotkami časovačů (timerů TIMx). Tyto jednotky mohou mít dvě funkce v závislosti na zapojení zdroje jejich hodinového kmitočtu.

Mohou fungovat jako časovače (timery), kde změna, tedy přírůstek jejich hodnoty je řízen synchronně oscilátorem s pevnou periodou např. 1 s (tzv. hodinovým kmitočtem).

Druhá funkce, režim čítače (counter), je charakterizována připojením hodinového kmitočtu zpravidla zvenku obvodu. Hodnota čítače se mění v závislosti na tomto zpravidla asynchronním signálu (okamžik jeho příchodu impulsu předem neznáme, pouze na něj čekáme).

5.2 Princip měření frekvence přímou metodou

Princip měření frekvence signálu přímou metodou spočívá v počítaní pulsů měřeného signálu po určitou pevnou dobu. K zastavení činnosti čítače se používá tzv. hradlo, viz obr. 5.1. Toto hradlo otevře cestu měřenému signálu na vstup čítače pouze po určitou dobu. Po tuto dobu bude čítač počítat pulsy měřeného signálu. Následně se z hodnoty počtu načtených pulsů a doby po kterou bylo otevřené hradlo, dle vztahu (5.2), vypočítá průměrná frekvence vstupního signálu.



Obr.5.1 Princip číslicového měření frekvence pomocí hradlování čítače.

Na době otevření hradla T_0 závisí rozlišovací schopnost měření frekvence signálu. Rozlišení měření frekvence f_x lze vypočíst podle vztahu (5.1).

$$f_x = \frac{1}{\left(\frac{T_o}{N}\right)} = \frac{N}{T_o} \tag{5.1}$$

f_x - měřená frekvence

T_0 - doba otevření hradla

N – počet načtených pulsů

Pokud bude doba otevření hradla 1 s, počet pulsů načtených čítačem bude odpovídat přímo hodnotě měřené frekvence v jednotkách Hz. Vztah (5.1) degraduje na vztah (5.2).

$$f_x = N \tag{5.2}$$

5.3 Kvantovací chyba měření frekvence

Omezení tohoto elegantního způsobu měření frekvence spočívá ve vysoké kvantovací chybě měření.

Jako příklad uvedu měření frekvence 10 Hz s dobou otevření hradla po dobu 1 s. Při tomto způsobu měření bude kvantovací chyba činit 1 Hz, tj. 10 % z měřené hodnoty. Abychom dosáhli nižší kvantovací chyby např. 0,01 % bylo by nutné zvýšit dobu otevření hradla na 1000 s. To je neúměrně dlouhá doba a proto se pro měření nižších frekvencí (cca. pod 10kHz zdroj [13]) používá jiná metoda měření, která spočívá v měření délky periody signálu. Rozbor této metody bude rozepsán v další kapitole zabývající se měřením délky pulsu a střídy signálu.

V měřících přístrojích se při měření frekvence používá přepínání mezi výše popsanými metodami měření na základě hodnoty měřené frekvence.

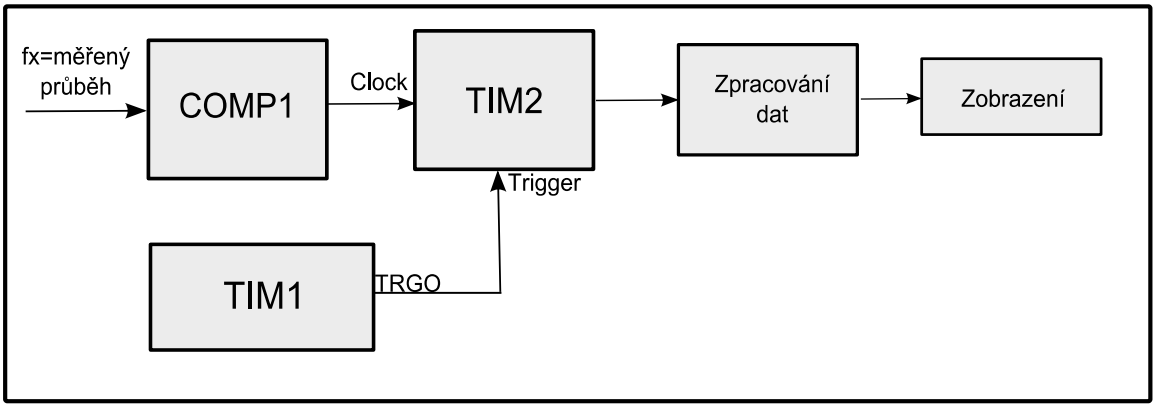
5.4 Použité periferie mikrokontroléru

Periferie časovačů

Pro řešení přístrojové funkce byl použit modul STM32F0-Discovery se zabudovaným mikrokontrolérem STM32F051R8. Tento mikrokontrolér disponuje několika 16bitovými časovači a jedním 32bitovým časovačem. Z této množiny časovačů byly použity čítače/časovače TIM1 (16bitový) pro hradlování a TIM2 (32bitový) pro počítání. Konfiguraci celého přístroje zachycuje obr.5.2.

Periferie komparátoru

Dalším problémem se kterým se potýkají systémy pro číslicové měření frekvence je nekvalitní vstupní signál. Kolísání vstupního signálu kolem rozhodovacích úrovní vstupních obvodů způsobuje načtení nesprávného počtu period měřeného signálu. Z tohoto důvodu bylo na vstup měřícího zapojení navrženo použití komparátoru s hysterezí. Mikrokontrolér disponuje dvěmi jednotkami komparátorů. Pro zlepšení odolnosti proti rušivému kolísání signálu byla použita jednotka mikrokontroléru COMP1 viz obr. 5.2.



Obr. 5.2 Propojení použitých bloků MCU pro měření frekvence.

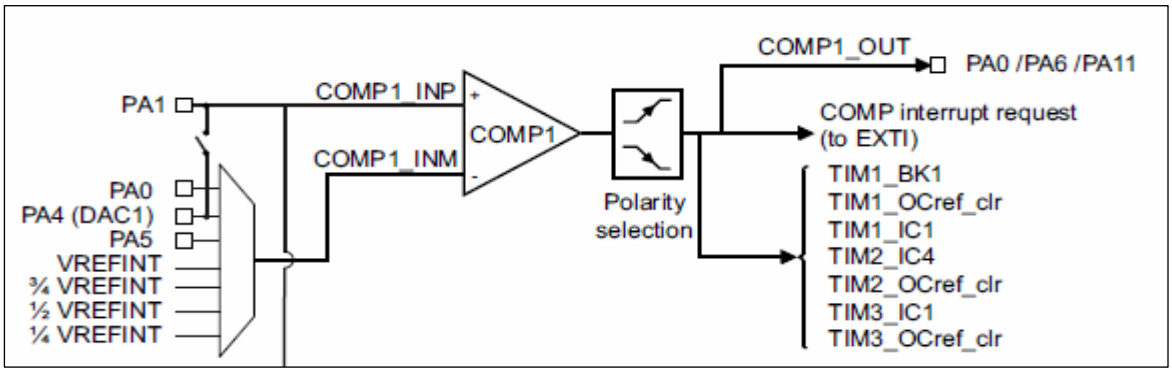
5.5 Konfigurace periferií pro měření frekvence

Periferie použitého mikrokontroléru jsou velmi propracovanými a složitými zařízeními s mnoha možnostmi vzájemného propojení. Z tohoto důvodu bude v následujícím textu popsáno a vysvětleno pouze konkrétní použité nastavení periferií mikrokontroléru v realizaci funkce měření frekvence.

5.5.1 Nastavení periferie komparátoru COMP1

Pro účel omezení rušivých vlivů na správný výsledek měření byl na vstup signálu zapojen komparátor COMP1. Tato periferie má jediný 32bitový nastavovací registr viz dokumentace [1], str. 222. Přehledové schéma komparátoru je zobrazeno na obr. 5.3.

Na neinvertující vstup komparátoru je připojen vstupní měřený signál prostřednictvím pinu PA1. Na invertující vstup je nastavením registru COMP_CSR připojena úroveň vnitřního referenčního napětí typ. 1,5 V. Dále byla nastavena pomocí bitů COMP1HYST na hodnotu 0x10 střední úroveň hystereze (dle [2] odpovídá typ. 15 mV). Vzhledem k tomu, že se nepodařilo najít variantu, jak vnitřně propojit výstup komparátoru se vstupem hodinového signálu časovače TIM2, byl výstupní signál nasměrován pomocí alternativní funkce periferie GPIOA na výstupní pin PA6. Pin PA6 je potom pomocí drátové propojky propojen na vstup hodinového signálu pin PA5 časovače TIM2.



Obr. 5.3 Schéma vnitřního zapojení periferie komparátoru mikrokontroléru, zdroj [1].

5.5.2 Nastavení periferie časovače TIM1

Časovač TIM1 v měřícím řetězci zastává funkci generátoru pulsu, po který bude povolena činnost čítače TIM2. Hodinová frekvence pro TIM1 je generována krystalem 8 MHz a po průchodu bloky periferie RCC (Reset and clock control) má hodnotu 48MHz.

Pro snadné zpracování hodnoty měřené frekvence časovačem TIM2, je vhodné povolovat dobu činnosti čítače v hodnotách 10^n , kde n je z množiny celých čísel. Pokud bude trvání pulsu zvoleno v délce 1 s, počet pulsů načtených časovačem TIM2 bude odpovídat přímo měřené frekvenci v jednotkách Hz.

Vstupní frekvence časovače PCLK je 48 MHz. Dělící blok „Prescaler“ je nastaven na hodnotu 4799 a dělí tedy tento vstupní kmitočet hodnotou 4800. Hodnota registru TIM1_ARR je nastavena na 9999, to odpovídá počtu pulsů 10000. Po načtení tohoto počtu pulsů se čítač zastaví, čas činnosti čítače odpovídá 1 s.

Časovači TIM1 je povoleno vyvolání přerušení, činnosti prováděné v obsluze přerušení jsou popsány pomocí obr. 5.7. Přerušení je podle výše uvedeného nastavení voláno po 1 s činnosti časovače. Časovač pracuje v režimu OPM (One pulse mode), je tedy nutné jej na konci přerušení znovu spustit.

Abychom mohli řídit činnost časovače TIM2 časovačem TIM1, je nutné v registru TIM1_CR2 nastavit bity MMS (Master mode selection) na hodnotu 001. Při tomto nastavení je signál enable CNT_EN časovače TIM1 použit pro řízení jeho výstupu TRGO (Trigger output).

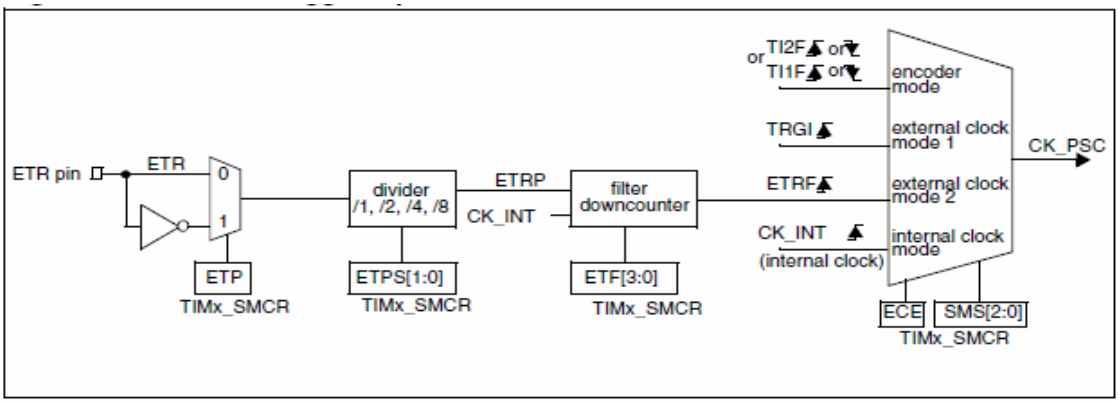
5.5.3 Nastavení periferie časovače TIM2

Časovač TIM2 slouží k počítání pulsů měřeného signálu. Dobu po kterou bude časovač počítat vstupní pulsy určuje časovač TIM2.

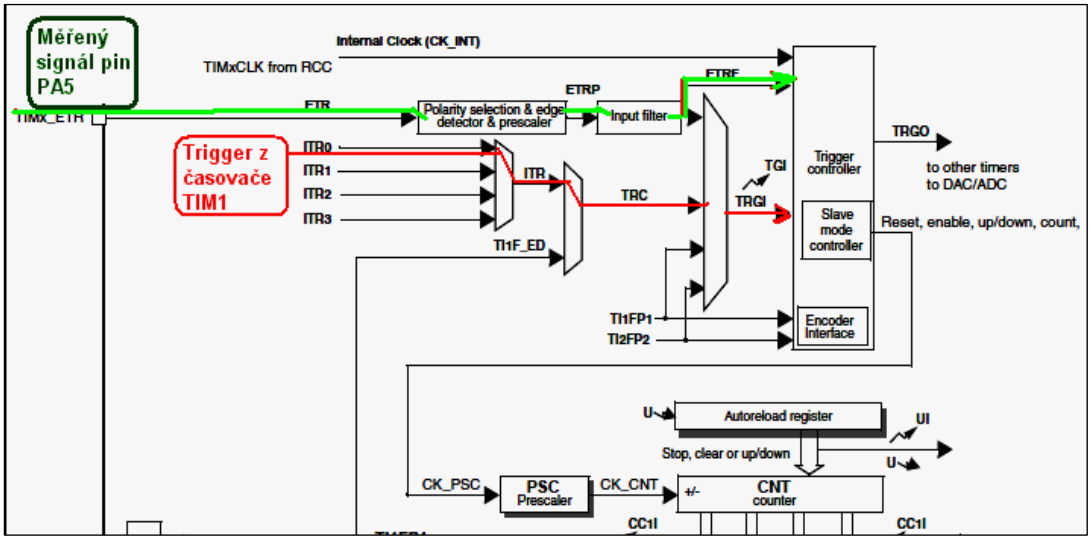
Režim hodinového kmitočtu časovače je pomocí bitu ECE = 1 registru TIM2_SMCR nastaven do režimu external clock mode 2. V tomto režimu je hodinový kmitočet (vstup čítače) řízen aktivní hranou vstupu ETRF, viz situace obr. 5.4 a obr. 5.5. Nastavením alternativní funkce pro vstup PA5, propojíme tento pin s externím vstupem časovače TIM2_ETR. Popsaná konfigurace externích hodin je vyznačena na obr. 5.5 pomocí zelené čáry.

Nastavení hradlového režimu časovače TIM2 provedeme nastavením bitů SMS (Slave mode selection) na hodnotu SMS = 101. Tím je časovač uveden do tzv. „gated mode“ (hradlovacího režimu). V tomto módu je čítači povolena činnost pouze tehdy, když je vysoká úroveň na jeho vstupu TRGI (Trigger input).

Zbývá propojit vstup TRGI časovače TIM2 s výstupem TRGO časovače TIM1. Propojení se realizuje zápisem bitu TS (Trigger selection) registru TIM2_CR2 dle tabulky tab. 5.1. Hodnota TS = 00 spojí TRGO časovače TIM1 s TRGI časovače TIM2.



Obr. 5.4 Nastavení vstup hodinového signálu časovače TIM2 pro měření frekvence, zdroj [1].



Obr. 5.5 Nastavení časovače TIM2 pro měření frekvence, upraveno z [1].

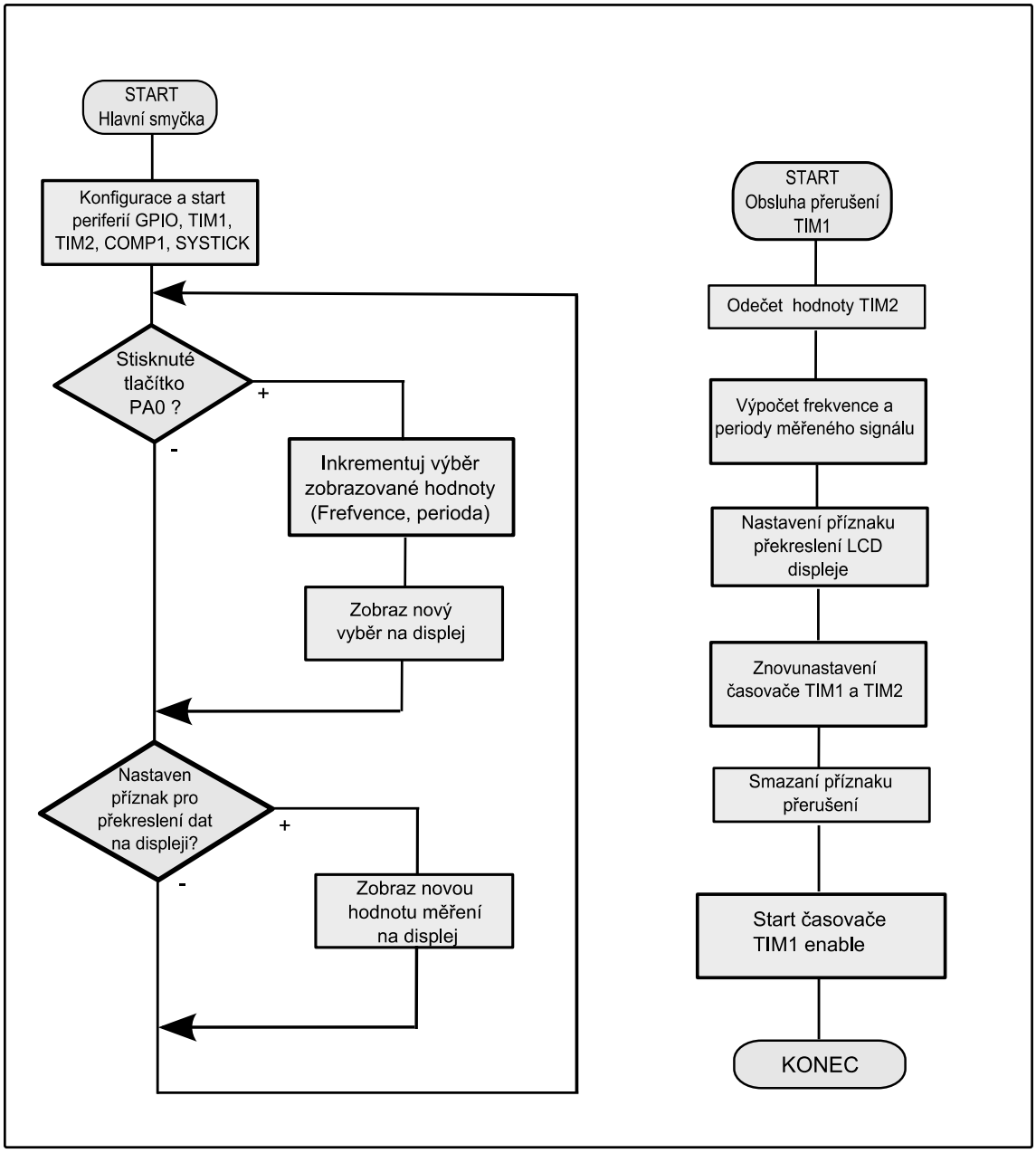
Tab. 5.1 Možnosti nastavení bitů trigger selection pro časovač TIM2, 3, zdroj [1].

Slave TIM	ITR0 (TS = 000)	ITR1 (TS = 001)	ITR2 (TS = 010)	ITR3 (TS = 011)
TIM2	TIM1	TIM15	TIM3	TIM14
TIM3	TIM1	TIM2	TIM15	TIM14

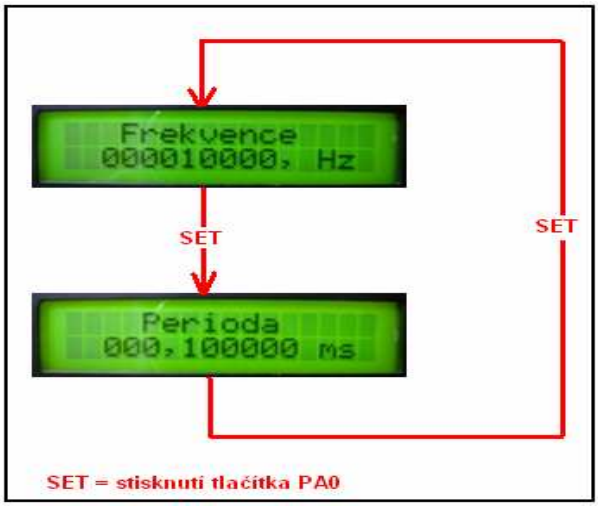
5.6 Popis programu pro měření frekvence

Program vykonává zobrazení měřené hodnoty frekvence a vypočtené hodnoty periody připojeného signálu na dvouřádkový 16znakový LCD display.

V obsluze přerušení časovače TIM1 viz obr. 5.7, ke kterému dojde po vygenerování pulsu v délce 1 s, je odečtena hodnota počtu pulsů z registru TIM2_CNT. Tato hodnota odpovídá přímo frekvenci v Hz. Z této hodnoty je také vypočtena hodnota periody signálu. Poté je nastaven příznak překreslení hodnoty LCD displeje, resetovány stavy časovačů TIM1 a TIM2. Povolením časovače TIM1 dojde k novému odměru vstupního signálu a následně k novému vyvolání přerušení, tím se celý cyklus periodicky opakuje.



Obr. 5.7 Vývojový diagram programu pro měření frekvence.



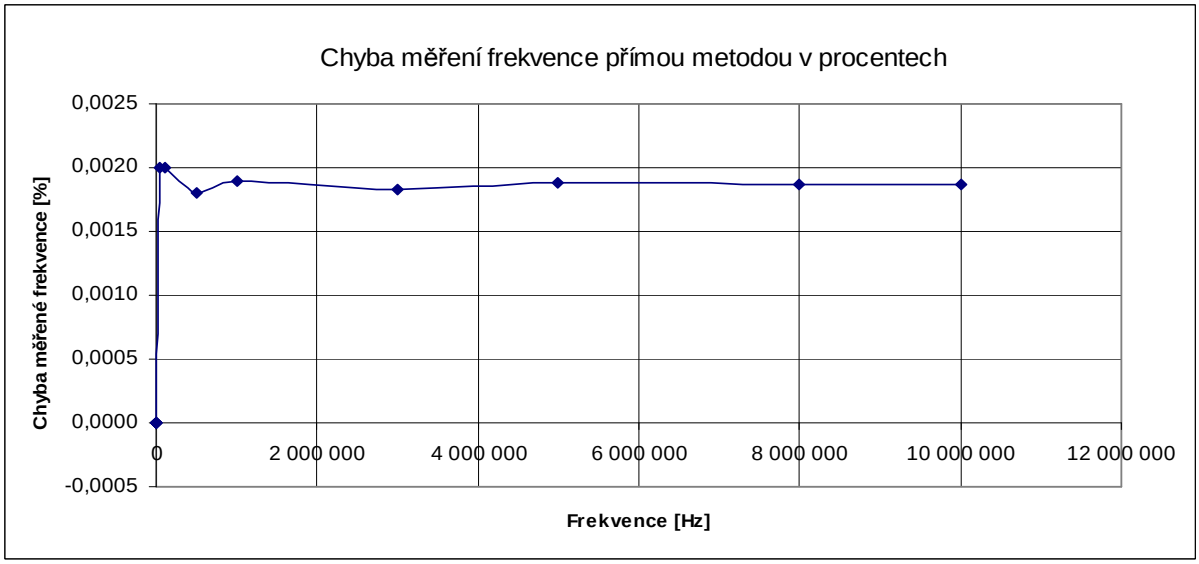
Obr. 5.6 Menu displeje přístroje pro měření frekvence signálu.

5.7 Ověření funkce zařízení pro měření frekvence

5.7.1 Měření odchylky naměřené frekvence od nastavené.

Ověření funkce měřícího přístroje bylo provedeno pomocí generování signálu tvaru obdélníku s rozsahem napětí Vp-p = 3 V. Zdrojem signálu byl vestavěný generátor osciloskopu DSO-X 2012A, jehož maximální frekvence je pro obdélníkový signál 10 MHz a pro sinusový signál 20 MHz.

Tento obdélníkový signál byl připojen na vstupní pin PA5 modulu STM32F0-Discovery. Pin PA5 je mapován na vstup čítače TIM2 pro počítání pulsů měřeného signálu. Naměřené údaje jsou zobrazeny graficky na obr. 5.8.



Obr. 5.8 Graf odchylky měřeného kmitočtu od nastaveného kmitočtu generátoru pro obdélníkový vstupní signál.

Tab. 5.1 Tabulka naměřených a vypočtených hodnot odchylek měřeného kmitočtu od nastaveného kmitočtu generátoru pro obdélníkový vstupní signál.

Frekvence nastavená [Hz]	Frekvence měřená [Hz]	Rozdíl frekvencí naměřená-skutečná [Hz]	Rozdíl frekvencí naměřená-skutečná [%]
10	10	0	0,0000
50	50	0	0,0000
100	100	0	0,0000
500	500	0	0,0000
1000	1000	0	0,0000
5000	5000	0	0,0000
10000	10000	0	0,0000
50000	50001	1	0,0020
100000	100002	2	0,0020
500000	500009	9	0,0018
1000000	1000019	19	0,0019
3000000	3000055	55	0,0018
5000000	5000094	94	0,0019
8000000	8000150	150	0,0019
10000000	10000187	187	0,0019

Z naměřených hodnot tab. 5.1 je patrné, že hodnota odchylky nepřekročila 0,002%.

5.7.2 Výpočet standardní nejistoty typu B pro přímé měření frekvence

Zdroje použité pro výpočet nejistoty [13], [14]. Uvažuje se rovnoměrné rozložení pravděpodobnosti odchylek.

Standardní nejistotu typu B se pro tento způsob sestává ze dvou nekorelovaných faktorů prvním je kvantovací chyba načítání pulsů, a druhým nestabilita krystalu použitého pro generování doby otevření hradla.

Vliv kvantovací chyby

Při době otevření hradla T_o bude rozlišovací schopnost přístroje $\Delta f_o = \frac{1}{T_o}$. (5.3)

Vliv nepřesnosti nastavení krystalu

Nepřesnost stability nastavení krystalu σf_{osc} vyjádříme v procentech vzhledem k měřené frekvenci dle vztahu (5.4) dostaneme vztah (5.5)

$$\Delta f_x = \frac{\sigma f_{osc}}{100} \cdot \frac{N}{T_o}, \tag{5.4}$$

Kde *N* je počet pulsů načtených za dobu otevření hradla T_o

Celková standardní nejistota typu B

$$u_B = \sqrt{\left(\Delta\ f_o/\sqrt{3}\right)^2 + \left(\Delta\ f_x/\sqrt{3}\right)^2} \tag{5.5}$$

Výpočet standardní nejistoty typu B pro měření frekvence kitem STM32F0-Discovery.

Nestabilitu oscilátoru, která tvoří na vyšších frekvencích dominantní složku nejistoty typu B se nepodařilo z typu oscilátoru přesně zjistit. Nestabilita nastavení oscilátoru, byla tedy odhadnuta z materiálů zdroj [15], jako $30\,ppm$.

Stanovení hodnot proměnných

$N = 10000000$, je pro počet vzorků odpovídající frekvenci měřeného kmitočtu $f_x = 10\ MHz$ při době otevření čítače TIM2 $T_o = 1\,s$.
Nepřesnost stability nastavení krystalu v procentech $\sigma\ f_{osc} = 30 \cdot 10^{-4}$ %

$$\Delta\ f_x = \frac{\sigma\ f_{osc}}{100} \cdot \frac{N}{T_o} = \frac{30 \cdot 10^{-4}}{10^2} \cdot \frac{10^7}{1} = \frac{30 \cdot 10}{1} = 300\ Hz \tag{5.6}$$

$$\Delta\ f_o = \frac{1}{T_o} = 1\ Hz$$

$$u_B = \sqrt{\left(\Delta\ f_o/\sqrt{3}\right)^2 + \left(\Delta\ f_x/\sqrt{3}\right)^2} = \sqrt{\left(1/\sqrt{3}\right)^2 + \left(300/\sqrt{3}\right)^2} = 173 Hz \tag{5.7}$$

Standardní nejistota typu B pro měřenou frekvenci 10 MHz je odhadnuta jako 173 Hz pro kr=1 (koeficient rozšíření).

5.8 Zhodnocení parametrů přístroje pro přímé měření frekvence

Zařízení s mikrokontrolérem STM32F051R8 je schopné měřit frekvenci v rozsahu od 1 Hz do 24 MHz. Maximální frekvence je omezena vzorkovacím kmitočtem externího vstupu, který může být vzorkován maximálně 48 MHz. Z Nyquistova kritéria bude maximální frekvence, kdy je možné správné měření frekvence vstupního signálu omezena na 24 MHz (viz také datasheet [2], str. 83).

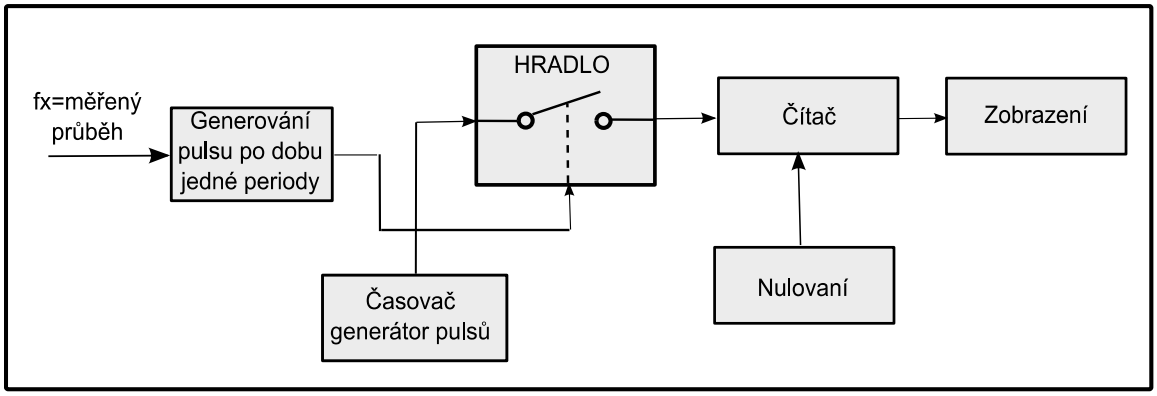
Měřením v rozsahu 1 Hz až 10 MHz při obdélníkovém průběhu a vstupnímu napětí $V_{p-p} = 3\,V$ byl zjištěn maximální rozdíl měřené a nastavené hodnoty +0,002 %. Tento rozdíl bude z větší části způsobena nepřesností krystalu použitého pro generování hodin mikrokontroléru.

6. Měření délky periody, pulsu, střídy, nepřímé měření frekvence

6.1 Metoda měření délky periody

V kapitole 2. byla nastíněna metoda pro číslicové měření délky periody. Klasický způsob řešení používá čítač, který je hradlován (činný) po dobu trvání periody měřeného signálu. Po dobu otevření hradla jsou načítány pulsy oscilátoru s pevnou známou frekvencí.

Hlavním úkolem přístroje je zachycení periody měřeného signálu a po tuto dobu otevřít hradlo viz obr.6.1.



Obr. 6.1 Princip číslicového měření délky periody.

T_{osc}

Rozlišení přístroje pro měření délky periody je závislé na délce periody generátoru pulsů časovače (na obr. 6.1 „Časovač generátor pulsů), jehož pulsy jsou po dobu otevření hradla počítány podle vztahu (6.1).

$$T_{osc} = \frac{1}{f_{osc}} \tag{6.1}$$

T_{osc} - rozlišení periody měřeného signálu

f_{osc} - frekvence generátoru (na obr. 6.1 „Časovač generátor pulsů)

Násobek počtu period s délkou periody generátoru pulsů načtených čítačem během periody měřeného průběhu odpovídá době periody měřeného signálu T_x dle vzorce (6.2).

$$T_x = T_{osc} \cdot N\,, \tag{6.2}$$

T_x - naměřená doba periody

N - počet načtených pulsů čítačem během jedné periody měřeného signálu.

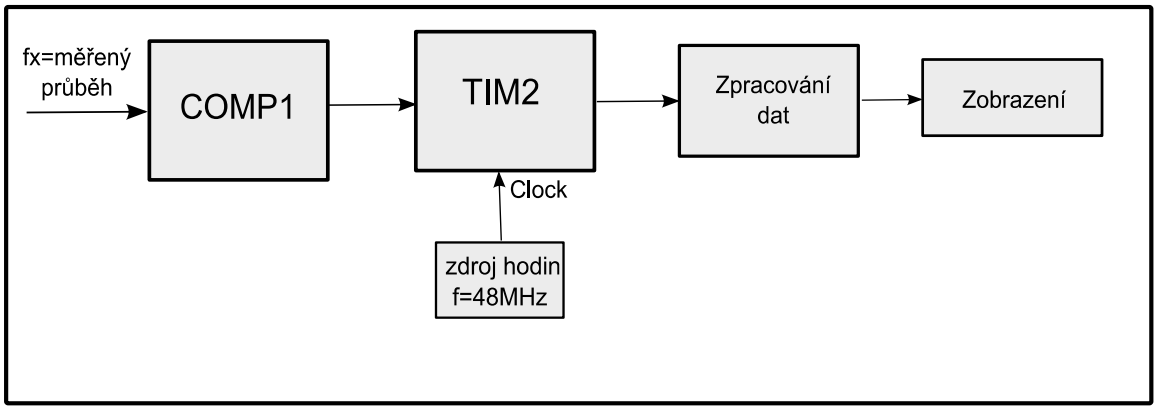
Hodnota frekvence měřeného signálu f_x je vypočtena na základě vztahu (6.3).

$$f_x = \frac{1}{T_{osc} \cdot N} = \frac{1}{T_K} \tag{6.3}$$

f_x - hodnota měřené frekvence

6.2 Měření délky periody mikrokontrolérem

Pro řešení této přístrojové funkce byl použit modul STM32F0-Discovery se zabudovaným mikrokontrolérem STM32F051R8. Mikrokontrolér obsahuje jeden 32bitový čítač/časovač a několik 16bitových, dále jsou k dispozici dva komparátory. Použité periferie mikrokontroléru, 32bitový čítač TIM2 a komparátor COMP1, byly vnitřně zapojeny do struktury dle obr.6.2.



Obr. 6.2 Základní propojení použitých bloků MCU pro délky periody.

6.3 Konfigurace periferií pro měření délky periody

6.3.1 Nastavení periferie komparátoru COMP1

Komparátor je nastaven stejným způsobem jako v případě přístroje pro měření frekvence, kapitola 5, podkapitola „5.5.1 Nastavení periferie komparátoru COMP1“. Propojení na vstup časovače PA5 je vyřešeno pomocí drátové propojky na pinech Discovery kitu. Tím je zároveň umožněno měření signálu za komparátorem a možná korekce nastavení offsetu vstupního signálu. Výstup komparátoru je na pinu PA6, vstup standardně na pinu PA1 viz obr. 5.3 kapitola 5.

6.3.2 Nastavení periferie časovače TIM2

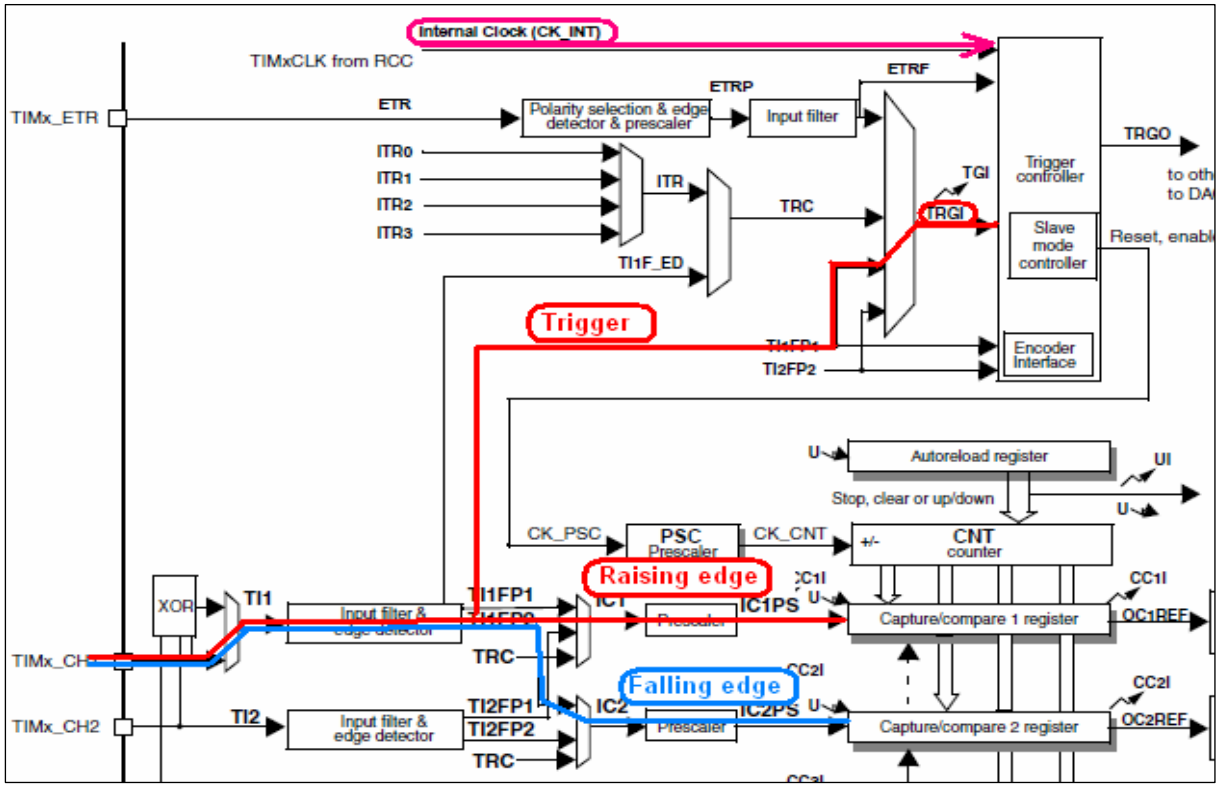
Časovač TIM2 slouží k počítání pulsů hodinové frekvence PLCK po dobu jedné periody měřeného signálu.

Pro realizaci výše uvedeného způsobu byly využity dvě jednotky capture/compare časovače CCR1 a CCR2. Tyto jednotky jsou schopné ve vstupním režimu zachytit na základě vstupů IC1 až IC4 aktuální stav počitadla TIM2_CNT. Vnitřní propojení jednotek časovače ukazuje obr. 6.3.

Jednotka CCR1 je nastavena, aby v případě sestupné hrany signálu přepsala aktuální stav čítače TIM2_CNT do svého registru TIM2_CCR1. Navíc sestupná hrana měřeného signálu způsobí vynulování stavu čítače v registru TIM2_CNT viz obr. 6.4.

Postup nastavení registrů časovače TIM2 pro měření délky periody vstupního signálu:

1. Aktivujeme hodinový signál pro TIM2 v registru RCC, jednotka APB1.
2. Jednotku CC1 nastavíme do vstupního režimu, vstup IC1 namapujeme na výstup TI1 zápisem bitů CC1S = 01 v registru TIM2_CCMR1.
3. Bity CC1NP, CC1P registru TIM2_CCER nastavíme na hodnotu 00. TI1FP1 signál bude reagovat na náběžnou hranu vstupu TI1.
4. Nastavíme bit CC1E registru TIM2_CCER na hodnotu 1. Pokud je CC1 kanál v režimu vstup, dojde k povolení capture funkce (capture enable).
5. Časovač musí být zapnutý pouze po dobu periody měřeného signálu, tzn. v rozmezí od náběžné hrany do další náběžné hrany. Nastavíme proto bit TS (trigger selection) registru TIM2_SMCR na hodnotu TS = 101, triggerem bude signál TI1FP1.
6. Zvolíme režim časovače v registru TIM2_CR2 bity SMS (Slave mode selection) do „Reset mode“. V tomto módu náběžná hrana signálu TRGI inicializuje čítač do počátečního nastavení a generuje update registrů. V našem případě zapíše do registru TIM2_CNT hodnotu 0.
7. Povolíme činnost čítače pomocí zápisu bitu CEN = 1 v registru TIM2_CR1.



Obr. 6.3 Vnitřní nastavení časovače TIM2 pro měření délky periody, upraveno z [1] .

6.3.3 Nastaveni TIM2 pro měření střídy signálu

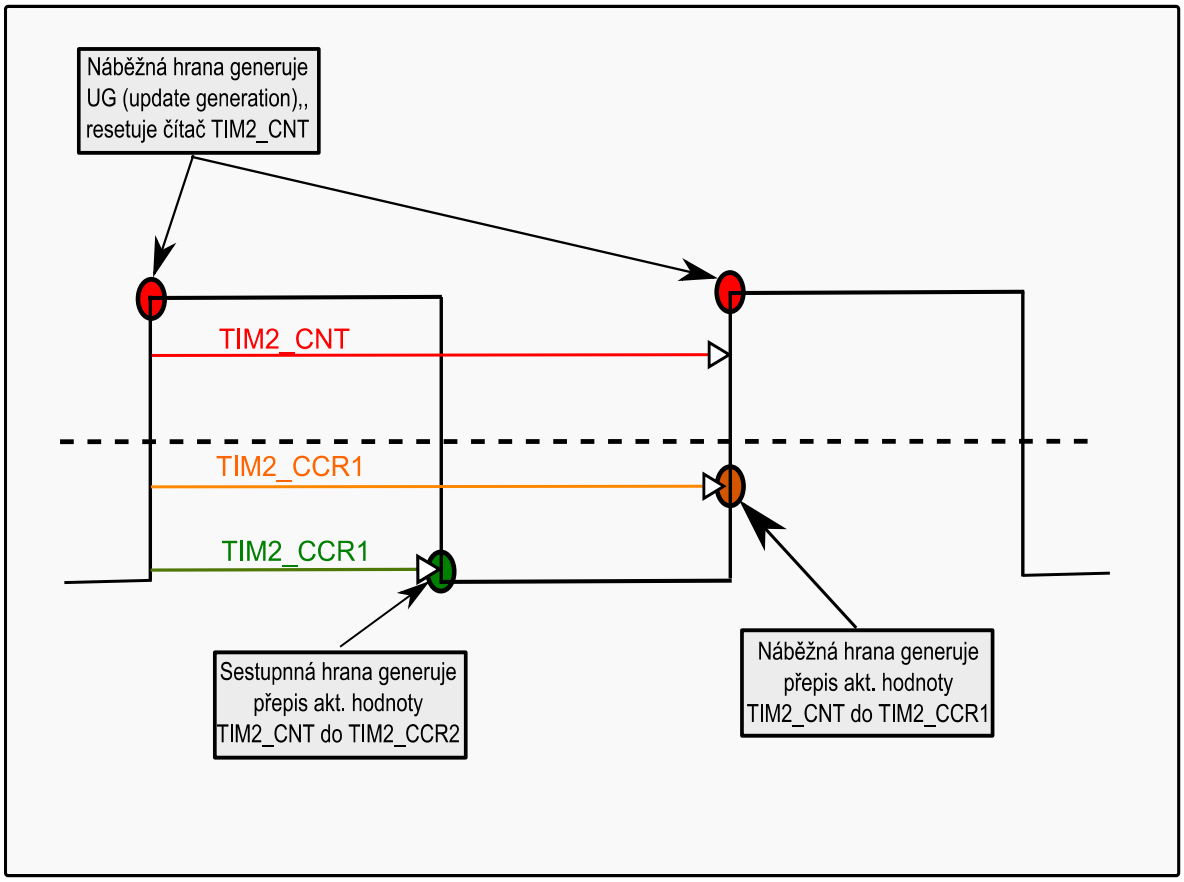
Pokud budeme chtít měřit střídu signálu, bude nutné měření doby po kterou je signál v kladné, nebo záporné části periody. K tomuto úkolu byla použita jednotka capture CCR2, která reaguje na sestupnou hranou signálu. Při sestupné hraně signálu zapíše do svého registru aktuální stav registru CCR2 viz obr. 6.4.

Postup nastavení registrů časovače TIM2 pro měření délky kladné části periody vstupního signálu:

1. Jednotku CC2 nastavíme do vstupního režimu, vstup IC2 namapujeme na výstup TI1 zápisem bitů CC1S = 01 v registru TIM2_CCMR1.

2. Bity CC2NP, CC2P registru TIM2_CCER nastavíme na hodnotu 01. TI1FP1 signál bude reagovat na sestupnou hranu vstupu TI1.

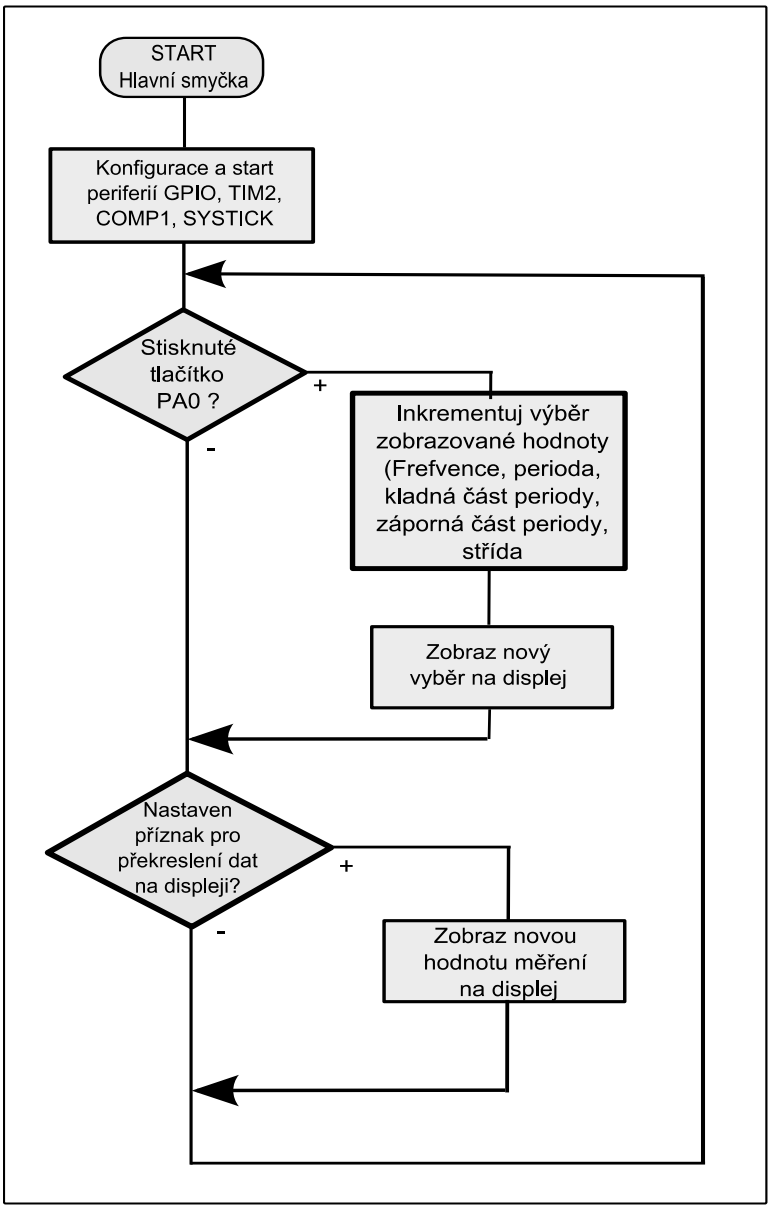
3. Nastavíme bit CC2E registru TIM2_CCER na hodnotu 1. Pokud je CC2 kanál v režimu vstup, dojde k povolení capture funkce (capture enable).



Obr. 6.4 Činnost bloků časovače TIM2 vzhledem ke vstupnímu měřenému signálu.

6.4 Vývojový diagram programu měření délky periody

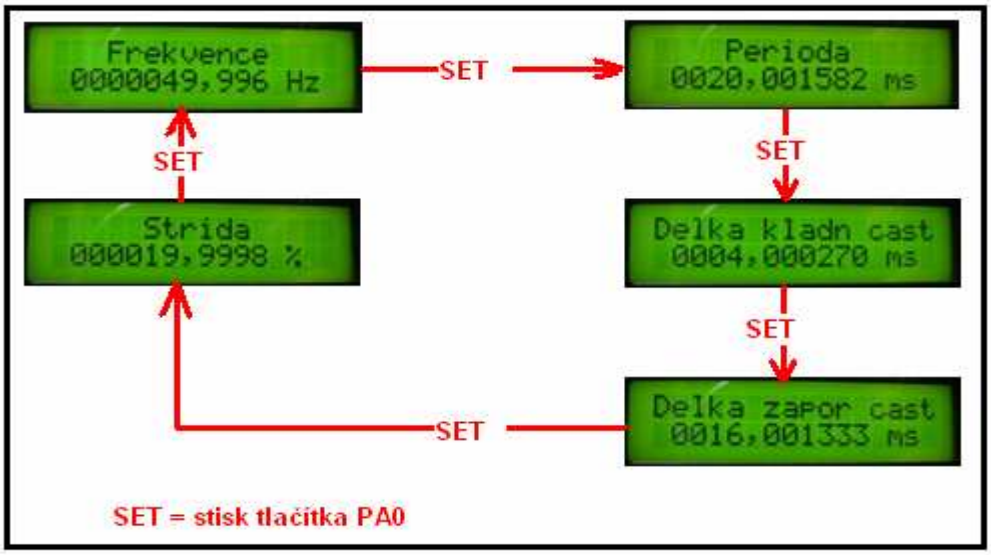
Řízení činnosti časovače TIM2 je prováděno přímo měřeným signálem a nevyžaduje žádnou součinnost programu. Program zajišťuje pouze zobrazení a výpočet hodnot získaných z naměřených hodnot. Vývojový diagram obr. 6.5 zachycuje obsluhu menu. Výpočty hodnot jsou prováděny z aktuálních hodnot s periodou 0,5 s v obsluze přerušení SYSTICK.



Obr. 6.5 Vývojový diagram programu měření délky periody.

6.5 Ovládání a menu přístroje

Ovládání přístroje je možné pomocí tlačítka přivádějící úroveň log. 0 na pin PA0. Opakovaným stiskem tlačítka je možné přecházet mezi zobrazením frekvence, periody, délky kladné části pulsu, délky záporné části pulsu a střídy. Schéma ovládání je zachyceno na obr. 6.6. V příloze je uvedeno zapojení LCD displeje k modulu STM32F0-Discovery.



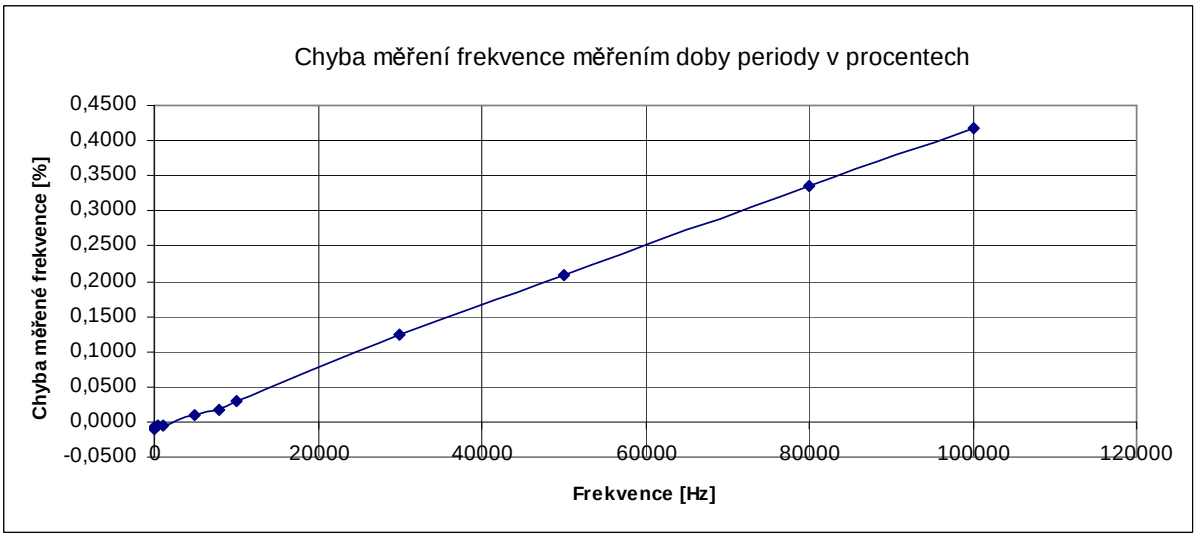
Obr. 6.6 Menu displeje přístroje pro měření délky periody signálu.

6.6 Ověření funkce zařízení pro měření délky periody

6.6.1 Měření odchylky měřené frekvence od nastavené

Ověření funkce měřícího přístroje bylo provedeno pomocí generování signálu tvaru obdélníku s rozsahem měřeného napětí Vp-p = 3 V. Zdrojem signálu byl vestavěný generátor osciloskopu DSO-X 2012A, jehož maximální frekvence je pro obdélníkový signál 10 MHz a 20 MHz pro sinusový signál .

Tento obdélníkový signál byl připojen na vstupní pin PA5 modulu STM32F0-Discovery. Pin PA5 je mapován na vstup čítače TIM2 pro počítání pulsů měřeného signálu. Naměřené údaje jsou zobrazeny graficky na obr.6.7.



Obr. 6.7 Graf odchylky měřeného kmitočtu (metodou měření délky periody) od nastaveného kmitočtu generátoru pro obdélníkový vstupní signál.

Tab. 6.1 Tabulka naměřených a vypočtených hodnot odchylek měřeného kmitočtu od nastaveného kmitočtu generátoru pro obdélníkový vstupní signál.

Frekvence nastavená [Hz]	Frekvence měřená [Hz]	Rozdíl frekvencí naměřená-skutečná [Hz]	Rozdíl frekvencí naměřená-skutečná [%]
10	9,9991	-0,0009	-0,0090
50	49,9960	-0,0040	-0,0080
100	99,9922	-0,0078	-0,0078
500	499,9687	-0,0313	-0,0063
1000	999,9583	-0,0417	-0,0042
2500	2500,1300	0,1300	0,0052
5000	5000,5210	0,5210	0,0104
8000	8001,3336	1,3336	0,0167
10000	10002,8400	2,8400	0,0284
30000	30037,5456	37,5456	0,1252
50000	50104,3840	104,3840	0,2088
80000	80267,5520	267,5520	0,3344
100000	100418,4064	418,4064	0,4184

Korekce počtu načtených pulsů

Z naměřených hodnot tab. 6.1 a graf obr. 6.7, je patrná poměrně vysoká chyba provedeného měření. Proto bylo provedeno další měření na jiném vývojovém modulu viz příloha. Hodnoty odchylek se zlepšily, nicméně chyba byla stále vysoká. Důvodem je pravděpodobně chyba hodinového kmitočtu krystalu.

Z tohoto důvodu je programem upraven počet načtených pulsů čítačem na hodnotu podle vztahu (6.4).

$$P_{upr} = P_{nact} - \left(\frac{P_{nact}}{13600} - 2 \right) \tag{6.4}$$

Po této korekci byla chyba zredukována na hodnotu 0,0002% viz tab. 6.2. Pro uvažovaný rozsah použití od 1 Hz do 2,5 kHz.

Tab. 6.2 Tabulka naměřených a vypočtených hodnot odchylek měřeného kmitočtu od nastaveného kmitočtu generátoru pro obdélníkový vstupní signál po provedené korekci hodinového kmitočtu.

Frekvence nastavená [Hz]	Frekvence měřená [Hz]	Rozdíl frekvencí naměřená-skutečná [Hz]	Rozdíl frekvencí naměřená-skutečná [%]
0,1	0,1000002	2E-07	0,0002
1	1,000002	2E-06	0,0002
10	10,00002	2E-05	0,0002
50	50,0001	0,0001	0,0002
100	100,0002	0,0002	0,0002
500	500	0	0,0000
800	800	0	0,0000
1000	1000	0	0,0000
2000	2000	0	0,0000
5000	5000	0	0,0000
8000	8001,3336	1,3336	0,0167
10000	10002,84	2,84	0,0284

6.6.2 Výpočet standardní nejistoty typu B pro měření délky periody

Standardní nejistotu typu B, vypočtená dle zdroje [12], se pro tento způsob sestává ze třech faktorů. Prvním je kvantovací chyba měření délky periody daná hodinovým kmitočtem časovače TIM2. Druhým faktorem je nestabilita nastavení kmitočtu krystalu použitého pro generování pulsů čítače TIM2. Třetím faktorem je směrodatná odchylka, způsobená různou dobou otevření hradla měřeným signálem.

Výpočet standardní nejistoty typu B doby periody

$$u_{T_x} = \sqrt{\left(\Delta' T_x / \sqrt{3}\right)^2 + \left(\Delta T_{osc} / \sqrt{3}\right)^2 + 2u_k^2} \tag{6.5}$$

Rozlišovací schopnost měření periody $\Delta' T_x = \frac{1}{f_{osc}}$ (6.6)

Vliv nastavení krystalu časovače TIM2 $\Delta T_x = \frac{\sigma f_{osc}}{100} \cdot T_x$, (6.7)

f_{osc} - frekvence oscilátoru generátoru časovače TIM2.

σf_{osc} - nepřesnost stability krystalu vyjádřená v procentech.

T_x - perioda měřeného signálu

u_k - směrodatná odchylka, způsobená různou dobou otevření hradla měřeným signálem.

Výpočet standardní nejistoty typu B pro měření délky periody kitem STM32F0-Discovery.

Kvantovací krok čítače TIM2 pro $f_{osc} = 48\ MHz$.

$$\Delta' T_x = \frac{1}{f_{osc}} = \frac{1}{48 \cdot 10^6} = 20{,}83 \cdot 10^{-9}\ s \tag{6.8}$$

Nestabilitu oscilátoru se nepodařilo z typu oscilátoru přesně zjistit, byla tedy odhadnuta z materiálů zdroj [15], jako $30\,ppm$. Nepřesnost stability nastavení krystalu v procentech $\sigma f_{osc} = 30 \cdot 10^{-4}\ \%$.

Výpočet pro frekvenci 10 Hz, dobu periody $T_x = 100\ ms$.

$$\Delta T_x = \frac{\sigma f_{osc}}{100} \cdot T_x = \frac{30 \cdot 10^{-4}}{10^2} \cdot 100 \cdot 10^{-3} = 3 \cdot 10^{-6}\ s \tag{6.9}$$

$$u_{B T_x} = \sqrt{\left(\left(\frac{1}{48 \cdot 10^6}\right) \Big/ \sqrt{3}\right)^2 + \left(\left(3 \cdot 10^{-6}\right) \Big/ \sqrt{3}\right)^2} = 1{,}73209 \cdot 10^{-6}\, s \tag{6.10}$$

Standardní nejistota typu B pro frekvenci měřeného signálu 10 Hz je odhadnuta jako $1.73\,\mu s$, tj. odpovídá změně frekvence v rozsahu 17,3 mHz pro koeficient rozšíření kr=1. Nebyla zahrnuta směrodatná odchylka u_k. Zdroje použité pro výpočet nejistoty [13], [14].

6.7 Zhodnocení přístroje měření frekvence nepřímou metodou, měření délky periody, měření délky pulsu a měření střídy signálu

Přistroj umožňuje nepřímé měření frekvence signálu, měřením délky periody, v rozsahu 1mHz až 2500Hz. U vyšších frekvencí, kvantovací chyba z důvodu malého počtu načtených pulsů, způsobuje vyšší procentuelní chyby.

Přístroj může měřit délku kladné a záporné části pulsu s minimální dobou trvání 20,83 ns. Kvantovací chyba přístroje je 20,83 ns. Maximální měřená délka pulsu je omezena zobrazením displeje na 9,99 s.

Přístroj umožňuje měření střídy signálu v rozsahu 0-100%.

7. Impulsní generátor pomocí PWM

7.1 Impulsní generátor a spojitost s PWM

Impulsní generátor má jako základní funkci, vytváření obdélníkových pulsů definované šířky, s definovanou periodou opakování. Tato definice je prakticky shodná s popisem funkce PWM (pulsně šířkové modulace). Rozdíl je pouze v možnostech nastavitelných rozsahů, kde v případě impulsních generátorů bývá nastavitelná šířka pulsu už od 5 ps s opakováním po cca 100 ns.

V případě PWM (pulsně šířková modulace) se pohybujeme s opakovací frekvencí v řádu do cca stovek kHz. Jako stupeň nastavení regulace u PWM používáme pojem střída (tj. poměr doby signálu v log.0 k době signálu v log.1) s rozsahem v procentech od 0% do maximální hodnoty 100%, někdy také poměrem hodnot.

V následujícím textu popíši vývoj impulsního generátoru principem PWM, oba tyto pojmy budou v následujícím textu pro jednodušší vysvětlení principu mít stejný význam, ačkoli obecně se jedná o odlišné oblasti.

7.2 Tvorba PWM pomocí časovače TIM2

Pro generování impulsního signálu byla použita jednotka časovače TIM2. Na obr. 7.1 je uveden výsek z úplného schématu časovače viz zdroj [1], str. 296 s naznačenými hlavními bloky.

Z důvodu velké variability nastavení periferie TIM2 pro režim PWM, bude pro lepší pochopení principu funkce zařízení, v následujícím textu popsáno pouze konkrétní nastavení použité ve vyvinutém programu pro testování funkce impulsního generátoru pomocí PWM.

7.2.1 Způsob nastavení délky periody signálu PWM

Začneme-li od začátku, hodinový signál pro časovač TIM2 pochází z externího krystalového oscilátoru s frekvenci 8 MHz. Pomocí jednotky fázového závěsu je tento hodinový kmitočet vynásoben na hodnotu 48 MHz.

S každým pulsem hodinového kmitočtu dojde v našem případě k inkrementaci hodnoty čítače v registru TIM2_CNT (na obr. 7.1 zeleně). Pokud hodnota čítače dosáhne hodnoty uložené v registru TIM2_ARR, dojde k vynulování hodnoty čítače a opětovné postupné inkrementaci hodnoty čítače s frekvencí 48MHz až do hodnoty TIM2_ARR. Cyklus se periodicky opakuje.

Z výše uvedeného je zřejmé, že hodnotou zapsanou do registru TIM2_ARR budeme měnit frekvenci generování pulsu.

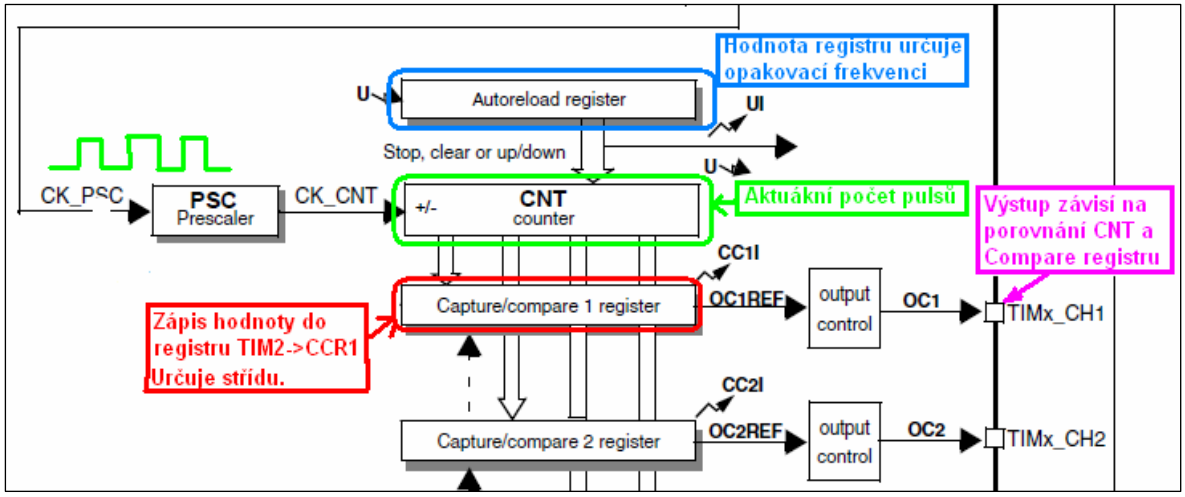
7.2.2 Způsob nastavení střídy signálu PWM

Nyní se podíváme, jak během intervalu o rozsahu 0 až TIM2_ARR řídit logickou úroveň výstupního pinu.

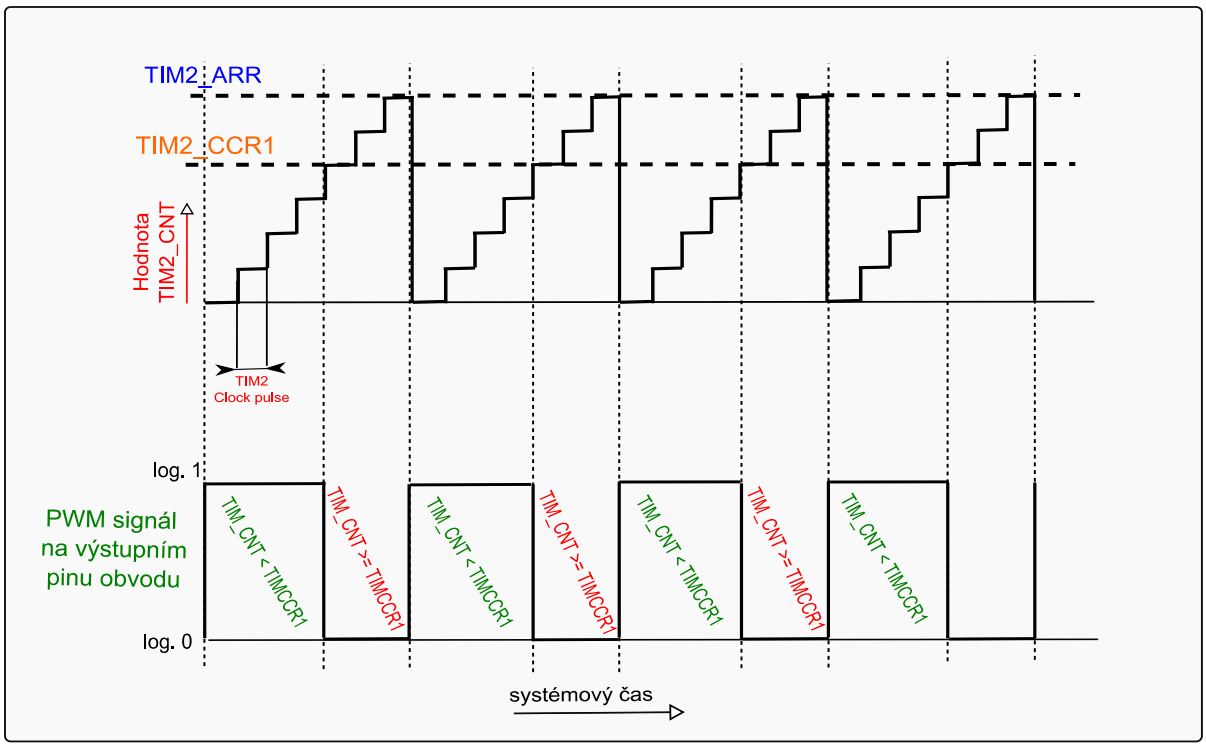
Můžeme si představit jak postupně v registru TIM2_CNT roste s každým pulsem hodin jeho hodnota. Tato hodnota je hardwarově průběžně porovnávána s hodnotou registru TIM2_CCR1. Pokud je hodnota TIM2_CNT < TIM2_CCR1, je výstup ovládaný registrem TIM2_CCR1 aktivován, tj. držen na hodnotě log. 1.

Pokud bude hodnota TIM2_CNT > TIM2_CCR1, výstup bude přepnut a držen ve stavu log. 0.

Tímto způsobem je mikrokontrolérem generován PWM signál. Pokud použijeme více kanálu časovače, můžeme v jedné periodě generovat jedním časovačem čtyři různé průběhy PWM. Popsaný způsob generování je přehledně zobrazen na obr. 7.2.



Obr. 7.1 Registry časovače TIM2 a jejich význam pro generování PWM signálu.



Obr. 7.2 Princip generování PWM signálu v souvislosti nastavením registrů TIM2 ARR, CCR1 pro nastavený režim PWM1.

7.3 Nastavení registrů časovače pro režim PWM

Možnosti nastavení generátoru PWM jsou velmi rozsáhlé a nebudu se zde zabývat jejich doslovným popisem. Zájemce odkazuji na základní materiál pro práci s obvodem STM23F05xxx [1], kapitola 16, str.295 až 355, kde jsou k nalezení podrobné informace týkající se nastavení registrů jednotky časovače TIM2.

Nicméně mne zaujaly dvě vlastnosti podporované funkce PWM, na které bych rád upozornil.

První je tzv. center aligned mode, který má využití dle zdroje [16], str. 11 při vícekanálovém způsobu řízení PWM pro snížení rušení. Snížení rušení tímto způsobem spočívá v odstranění souběhu spínání, vypínání více zařízení v jeden okamžik.

Druhou vlastností je možnost uvedení výstupu do tzv. „bezpečné úrovně“ použitím signálu OCREF_CLR_INT , podrobněji k nastudování v [1], str. 311. Tento blokovací signál může být generován například komparátorem COMP1 a při detekci překročení proudu zátěže uvést řízený systém do bezpečného stavu.

Nastavení periferie TIM2 základního režimu PWM je díky podpoře výrobce velice snadné. Následuje popis nastavení registrů TIM2 v jednotlivých bodech s uvedenými příkazy (zapsané tučně) pro provedení v bodech popsaných kroků :

1. Zapnutí hodin do periferií TIM2 a GPIOA pomocí registrů RCC.
   RCC->APB1ENR|=0x0001;
   RCC->AHBENR|=0x20000;

2. Nastavení módu pinu PA15 na alternativní funkci. Nastavení alternativní funkce AF2 pro výstupní pin PA15. Na pinu PA15 bude výstup PWM. Všechny možnosti přiřazení alternativních funkcí pinům mikrokontroléru jsou uvedeny v [2], tab.13, str. 29.
   GPIOA->MODER|=0x80000000;
   GPIOA->AFR[1]|=0x20000000;

3. Upravení délky periody například na 1 us. Frekvence hodin časovače je 48 MHz.
   TIM2->ARR=48000;

4. Upravení bodu pro vypínání výstupního signálu TIM2_CH1, podmínkou je nastavení režimu PWM módu 1. Hodnota by měla být nižší než je hodnota registru TIM2_ARR.
   TIM2->CCR1=12000;

5. Nastavení PWM módu 1, výstupní kanál je neaktivní když TIM2_CNT > TIM2_CCR.
   TIM2->CCMR1|=0x60;

6. Povolení činnosti jednotky výstupu OC1 viz obr. 7.1.
   TIM2->CCER=0x01;

7. Spuštění jednotky čítače TIM2. Tímto krokem je spuštěno generování PWM.
   TIM2->CR1|=0x01;

7.4 Ovládání přístroje pro testování generátoru PWM

Pro testovací účely byl program generátoru PWM doplněn o obsluhu LCD displeje, na kterém je možné zobrazit aktuální hodnotu stavu registru TIM2_ARR, kterým nastavujeme periodu signálu. Dále je možné zobrazit hodnotu registru TIM2_CCR1, kterým můžeme upravovat střídu generované PWM, princip viz obr. 7.2.

Nastavování hodnot zobrazených registrů je možné pomocí tlačítek připojených na piny mikrokontroléru PA0, 1, 2. Tyto tlačítka mají následující funkce :

PA0 - tlačítko SET, slouží k přesunu kurzoru směrem k vyšším řádům, jeden stisk = posun o jednu pozici. a k přepnutí do nastavovacího režimu.

PA1 - tlačítko „+“, stisk tlačítka inkrementuje hodnotu na které se nalézá blikající kurzor.

PA2 - tlačítko „-“, stisk tlačítka dekrementuje hodnotu na které se nalézá blikající kurzor.

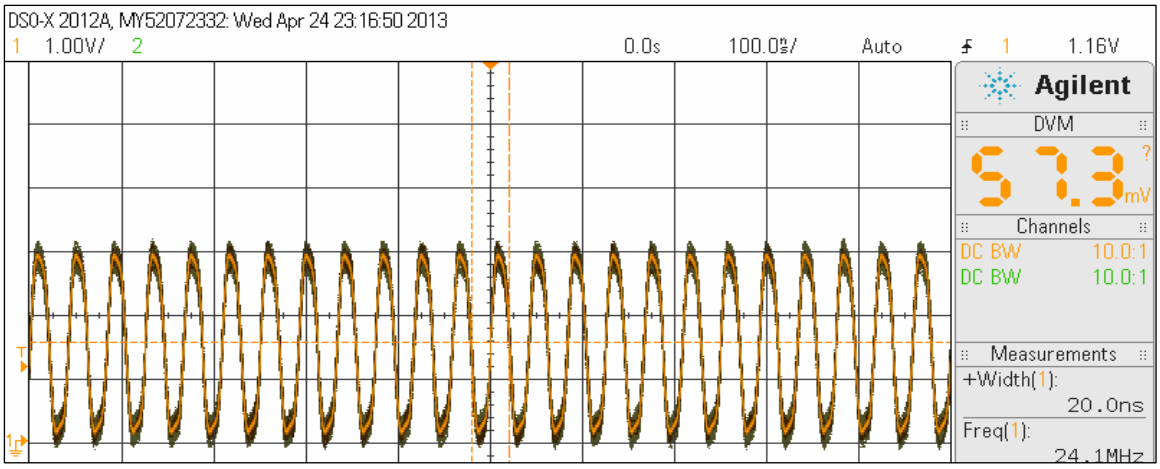
Obr. 7.3 Zobrazení nastavení registrů TIM2_ARR, CCR1na LCD displeji.

Popis připojení LCD displeje k modulu STM32F0-Discovery je uveden v příloze.

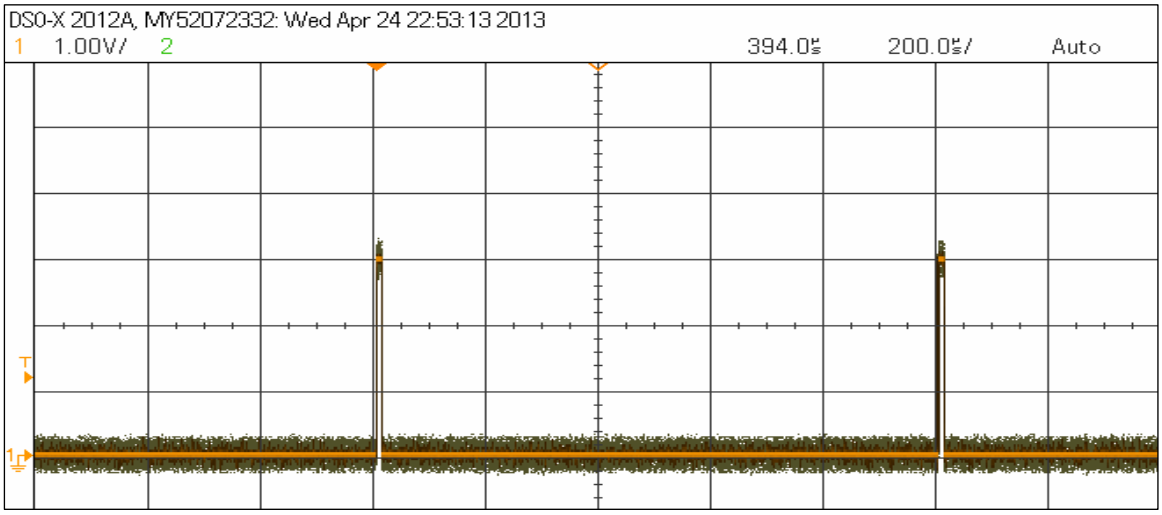
7.5 Ověření funkce impulsního generátoru

Ověření funkce generátoru bylo provedeno pomocí sledování signálu PWM vyrobeného mikrokontrolérem na displeji osciloskopu. Na obr. 7.4 je zachycen signál s maximálním možným kmitočtem 24MHz se střídou 1/1. Signál není obdélníkového tvaru z důvodu konečné rychlosti přeběhu (dle [2] typ 8 ns) a vlivu parazitních kapacit.

Obr. 7.5 zachycuje průběh generovaného signálu s opakovací frekvencí 1 kHz, tj. periodou 1 ms. Impulsy mají střídou 1/100. Generované pulsy mají tedy dobu trvání 10 us.



Obr. 7.4 Nejvyšší dosažený kmitočet 24 MHz s délkou pulsu 21 ns a střídou 1/1, generovaný mikrokontrolérem v režimu PWM1.



Obr. 7.5 Signál PWM generovaný mikrokontrolérem s šířkou pulsu 10 us a opakovací frekvencí 1 kHz.

7.6 Základní parametry PWM impulsního generátoru

Impulsní generátor realizovaný pomocí s mikrokontroléru STM32F051R8 dosahuje níže uvedených parametrů, za podmínky napájení obvodu napětím VDD v rozsahu 2.7 V až 3.6 V a velikosti zatěžovací kapacity výstupního pinu max. CL = 50 pF, zdroj [2] str. 73, tab. 52 vstupní a výstupní charakteristiky obvodu STM32F05xx.

Minimální šířka generovaného pulsu.....20,83 ns
Maximální šířka generovaného pulsu.....89,47 s
Krok změny šířky pulsu.....20,83 ns
Náběžná a sestupná hrana signálu..... max. 10 ns

Minimální opakovací frekvence pulsu.....18 mHz
Maximální opakovací frekvence pulsu.....24 MHz

7.7 Porovnání aplikace generátoru s klasickými obvody

Obvod LTC6993 viz [17], výrobce Linear technology je monostabilní pulsní generátor. Tento obvod je schopný generovat samostatné pulsy s šířkou v rozsahu 1 us až 33,6 s. Konfigurace obvodu je prováděna pomocí vnějších rezistorů. Cena obvodu cca 100 CZK.

Obvod LTC6992 viz [17], výrobce Linear technology je obvod určený pro generování PWM signálu. Tento obvod je velmi podobný obvodu LTC6993 s tím rozdílem, že obvod nemá vstup trigger pro spuštění vnitřního oscilátoru, oscilátor je u LTC6992 v provozu nepřetržitě. Vlastnosti obvodu jsou opět nastavovány pomocí vnějších rezistorů. Frekvence PWM modulace je nastavitelná v rozmezí 3,81 Hz až 1MHz se střídou 0-100%. Cena obvodu cca 130 CZK.

Výše uvedené obvody patří mezi jednodušší na trhu. Pokud je budeme porovnávat s navrhovaným řešením generátoru, musíme konstatovat, že z hlediska přesnosti a možností nastavení, a i z hlediska dosahovaných parametrů, je konstrukce impulsního a PWM generátoru pomocí obvodu STM32F51, kvalitnějším řešením. Na druhou stranu pro použití mikrokontroléru je nutné navíc vyvinout potřebný software.

8. Enkodér a zpracování jeho signálu jednotkou čítačů

8.1 Enkodér - princip, vlastnosti

Enkodér je zařízení používané v drtivé většině případů k určování polohy součástí strojů a přístrojů. Enkodéry poskytují na svém výstupu tzv. kvadraturní signál složený za dvou signálů A a B. Tyto signály jsou vzájemně posunuty o 90° viz obr 8.3. Na základě přicházejících pulsů je tedy možné rozpoznat směr otáčení zařízení, počítáním hran signálu určujeme změnu polohy. Enkodér většinou také obsahuje signál C, který během jedné otáčky poskytne jeden puls. Tento puls se používá při kalibraci polohy. Důvodem je inkrementální způsob funkce senzoru, tzn. poskytuje pouze relativní informaci o poloze narozdíl od absolutních snímačů polohy.

8.2 Typické provedení enkodéru

V aplikacích s enkodéry jsou převážně používány inkrementální senzory pracující na optickém principu, spočívající v prosvětlování skleněného disku. Tento disk má na vnějším okraji natištěnu masku složenou z tmavých proužků. Otáčení disku způsobuje zaclánění a odkrývání optických senzorů stop A a B posunutých o 90° a tím změnu jeho výstupního signálu. Na obrázku obr. 8.1 je fotografie enkodéru Omron s rozlišením 100 pulsů na jednu otáčku. Počítáním hran signálu viz obr. 8.2 získáme maximální rozlišení 400 dílů na otáčku. Prosvětlovací kotouč tohoto konkrétního enkodéru je vyroben z plechu, na vnějším okraji vpravo je patrný výsek pro detekci C stopy.

Signál enkodéru může být generován i na základě jiných fyzikálních principů, např. snímání magnetické stopy halovými senzory.

Obr. 8.1 Fotografie enkodéru Omron s rozlišením 100 pulsů na otáčku.

8.3 Způsob kalibrace polohy

Z předchozího textu je zřejmé, že pokud budeme chtít senzor použít pro měření polohy, je nutné určit výchozí bod, ke kterému budeme moci vztáhnout měřenou polohu. Na počátku měření je nutné provést tzv. kalibraci polohy (v praxi se nazývá tento postup „homing“).

Postup kalibrace může být následující:

1. Po zapnutí přístroje je pohon motoru otáčen nízkou rychlostí směrem k senzoru (např. koncový spínač, indukční senzor apod.).

2. Po detekci signálu z koncového spínače, následuje hledání hrany C signálu.

3. V okamžiku nalezení hrany tomuto bodu přiřadíme absolutní hodnotu polohy a ukončíme kalibraci polohy zařízení.

8.4 Podpora enkodéru mikrokontrolérem

Mikrokontrolér STM32F0xx lze díky podpoře výrobce snadno použít pro měření polohy pomocí enkodéru. V manuálu firmy STMicroelectronics [1], odstavec 16.3.12 na str. 320 a zde v kapitole 8.5 je podrobně popsáno nastavení časovače TIM pro realizaci této funkce.

Výhodou mikrokontroléru je implementovaný 32 bit časovač TIM2, který umožňuje hardwarové využití vyššího rozsahu polohování s jedním časovačem.

8.5 Nastavení časovače v MCU pro funkci enkodér

V následujícím textu je popsáno nastavení časovače a vstupně výstupní periferie nutné pro zpracování signálů z enkodéru.Výsledné nastavení je graficky zachyceno na obrázku 8.2 blokového diagramu časovače.

První část - připojení signálů A a B enkodéru k periferii časovače.

Nastavení periferie GPIOA.

1. V tabulce tab. 13 str. 29 datasheetu [2], vybereme vhodné piny, tak aby na ně bylo možné připojit vstupní kanály 1 a 2 časovače TIM2. Těmto pinům přiřadíme prostřednictvím registru GPIOA_MODER, příslušné alternativní funkce.

2. V registru GPIOA_AFRL nasměrujeme piny TIM2_CH1 a TIM2_CH2 na vstupní kanály rozhranní enkodéru TIMx viz obr. 8.2.

V dalších bodech provedeme nastavení registrů TIM2.

3. V registru TIM2_SMCR nastavíme „Encoder mode3“ tzn. čítač bude počítat s každou platnou hranou signálů připojených k vstupům TI1FP1 a TI2FP2 viz obr. 8.2.

4. Pomocí kombinace bitů CCxP a CC1xNP registru TIM2_CCER je možné nastavit reakci časovače (náběžná, sestupná hrana, inverze) na vstupní kanály enkodéru TI1FP1 a TI2FP2.

5. Prostřednictvím registru TIM2_CCMR propojíme výstupy IC1 a IC2 se vstupy TI1FP1 a TI2FP2.

6. V případě výskytu rušení je možné nastavit vstupní filtraci signálů s pomocí bitů IC1F a IC2F registrů CCMR1.

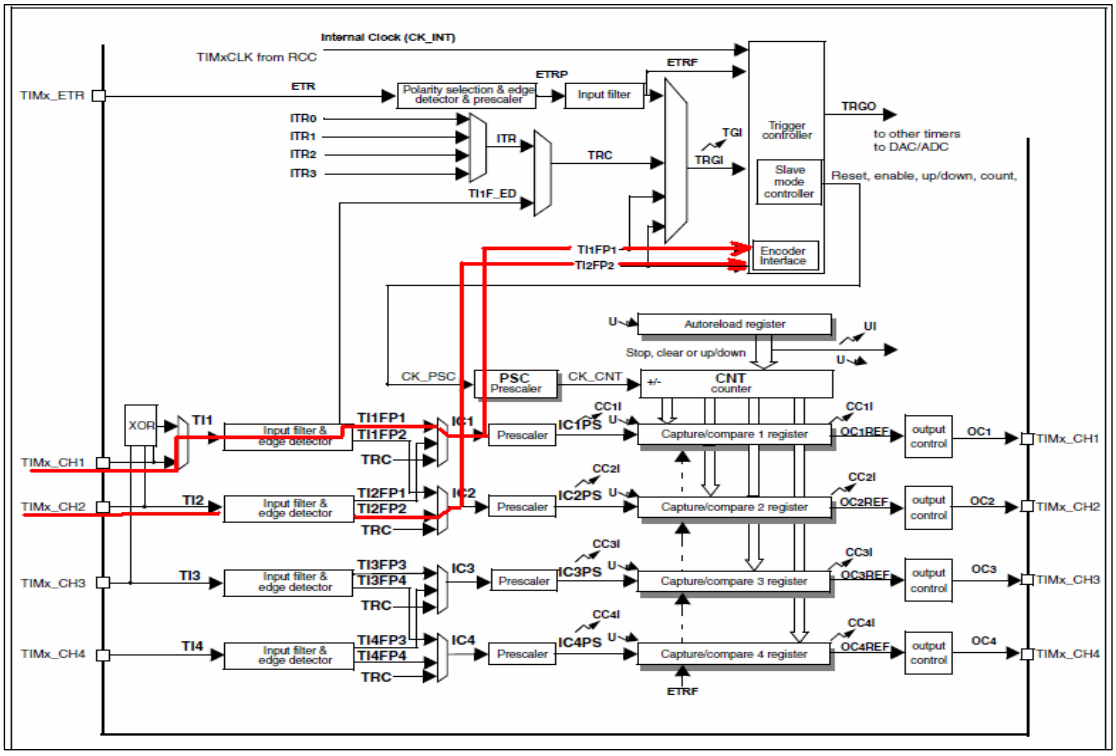
7. Posledním příkazem je povolení funkce periferie TIM2, tím dojde ke spuštění sledování vstupů s připojeným enkodérem. Aktuální počet načtených pulsů je udržován v 32 bit registru CNT.

Instrukce provádějící výše popsané nastavení.

```
GPIOA->MODER|=0x800;      //AF funkce pro vstup PA5
GPIOA->MODER|=0x08;       //AF funkce pro vstup PA1

GPIOA->AFR[0]|=0x200000; //AFRL nastaveni PA0 na AF1(TIM2_CH1_ETR)
GPIOA->AFR[0]|=0x020;     //AFRL nastaveni PA1 na AF2(TIM2_CH2)

TIM2->SMCR|=0x003;        //set encoder mode 3
TIM2->CCER=0x00;          //sensitive to rising edge
TIM2->CCMR1|=0x0001;      //CC1 chanel is configured as input, IC1 is mapped on TI1
TIM2->CCMR1|=0x0100;      //CC2 chanel is configured as input, IC1 is mapped on TI2
TIM2->CR1|=0x01;          //counter enable
```



Obr. 8.2 Schéma připojení výstupů enkodéru do bloku časovače TIM2.

Druhá část - nastavení reakce MCU na vstup C enkodéru

Pro kalibraci enkodéru a kontrolu počtu načtených pulsů mezi otáčkami je použita stopa C. Tento signál je snímán mikrokontrolérem pomocí vstupu PA0, který je připojen k periferii EXTI. Tato periferie umožňuje, po zachycení předdefinované změny na vstupu časovače TIM2_ETR (sestupná nebo náběžná hrana vstupního signálu), vyvolat funkci přerušení.

8.6 Funkce přerušení EXTI, kalibrace a kontrola enkodéru

V obsluze přerušení funkcí *EXTI0_1_IRQHandler()*, vyvolané sestupnou hranu signálu stopy C enkodéru mohou být provedeny dvě různé činnosti. První je kalibrace polohy (v případě, že nebyla dosud vykonána), druhou pak kontrola správného počtu načtených pulsů enkodérem.

Kalibrace polohy

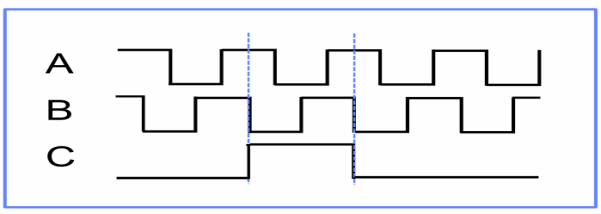
Kalibrace polohy je provedena po detekci sestupné hrany signálu na vstupu TIM2_ETR, při otáčení enkodéru směrem vlevo, podle popisu uvedeném v kapitole 8.3 a obr. 8.4 (pouze je vynechán koncový spínač uvolňující hledání C značky). Přiřazení hodnoty provedeme ve funkci obsluhy přerušení, kde přepíšeme stav registru TIM2_CNT na kladnou hodnotu celočíselně dělitelnou počtem pulsů za celou otáčku a nastavíme příznak provedené kalibrace. Tato hodnota by měla být dostatečně vysoká, tak aby nemohlo dojít k podtečení stavu čítače.

Kontrola počtu pulsů

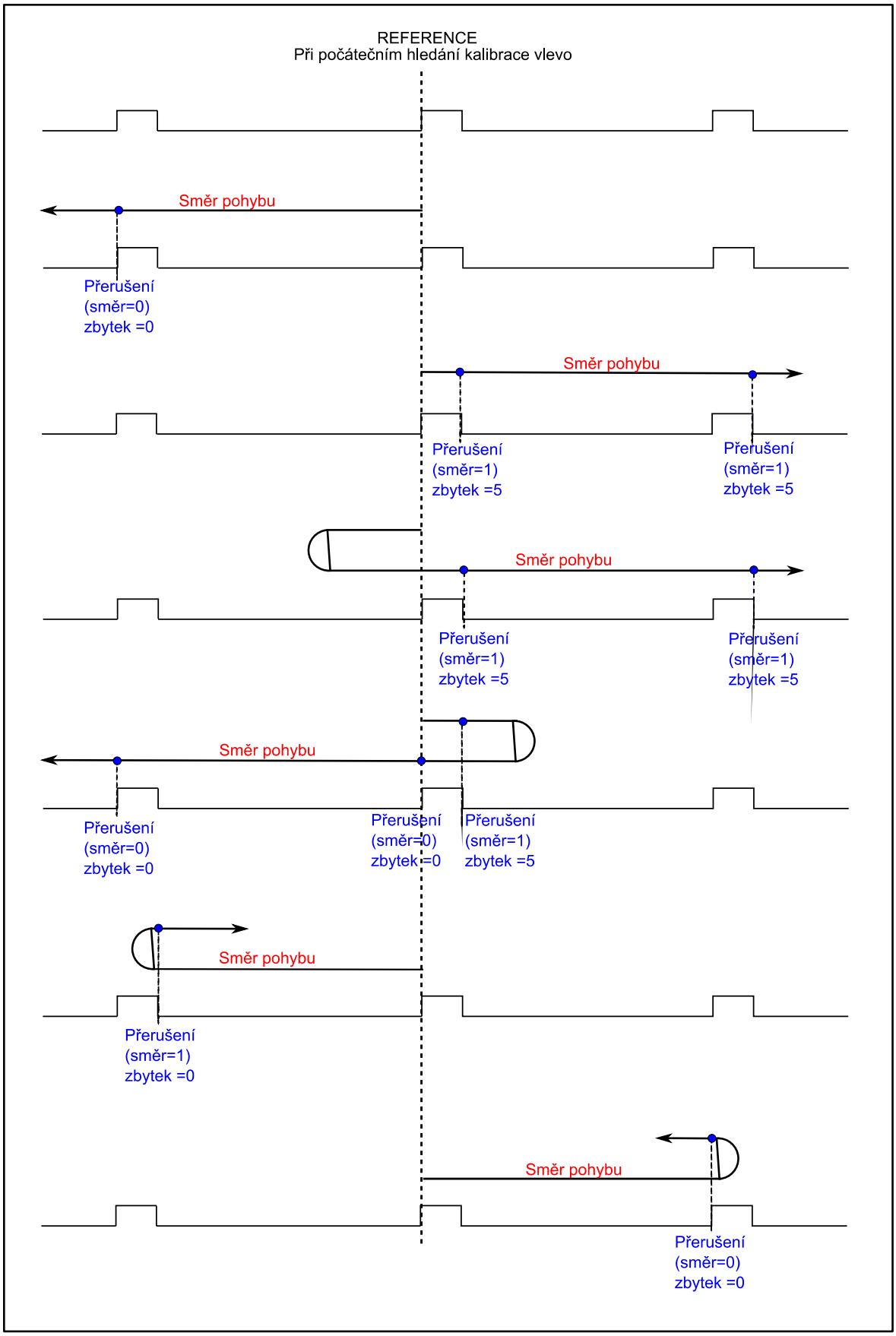
Pro testovací účely byla implementována v obsluze přerušení kontrola počtu pulsů načtených enkodérem. K dispozici byl enkodér fy. Omron E6B2-CWZ6C.

V případě vyvolání obsluhy přerušení EXTI sestupnou hranou signálu stopy C enkodéru, za podmínky již provedené kalibrace, provádíme kontrolu počtu načtených pulsů. Tato kontrola je provedena vydělením stavu čítače TIM2_CNT modulo počtem hran enkodéru na jednu otáčku, zbytek tohoto dělení je porovnáván v závislosti na směru otáčení enkodéru viz obr. 8.4. Pokud nastane odchylka od předpokládané hodnoty, dojde k rozsvícení LED diody PC8 na modulu STM32F0-Discovery.

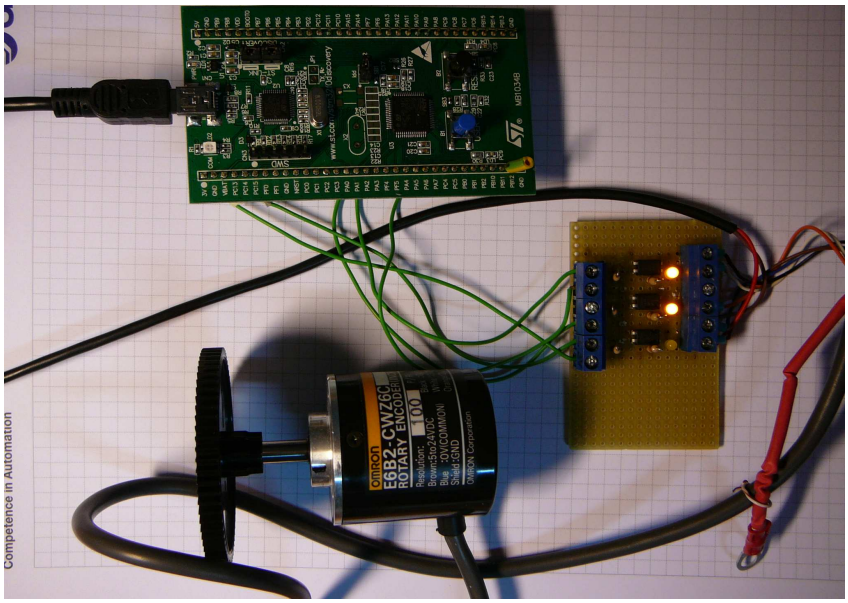
Kontrola pulsů je specifická pro použitý enkodér a v případě použití jiného typu enkodéru bude nutné změnit konstanty v programu. Obr.8.3 graficky zachycuje problémy využití C signálu enkodéru pro kontrolu počítání pulsů. Prvním problémem je nenulový počet pulsů během aktivní úrovně C signálu, druhým problémem změna úrovně B stopy signálu současně se změnou C signálu. Na obr. 8.4 jsou zobrazeny možné stavy bez poruch s vyznačenými okamžiky vyvolaného přerušení (přerušení nastaveno na sestupnou hranu, které mohou nastat při provozu enkodéru.



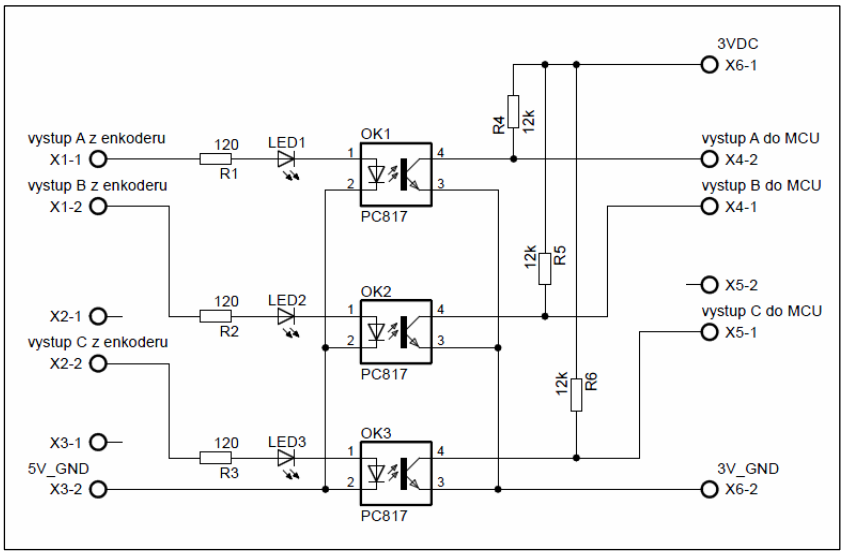
Obr. 8.3 Výstupní signály enkodéru Omron E6B2-CWZ6C.



Obr. 8.4 Možnosti vyvolání přerušení v závislosti na pohybu enkodéru a odpovídající zbytky po modulo dělení počtem pulsů enkodéru.



Obr. 8.5 Modul Discovery, enkodér a oddělovací modul enkodéru.



Obr. 8.6 Schéma oddělovacího modulu enkodéru.

8.7 Porovnání mikrokontrolér versus dekodér Agilent HCTL 2022

Obvod HCTL 2022(2032) výrobce fy. Agilent je obvodem, který se používá jako rozhraní mezi enkodérem a mikrokontrolérem. Vyrábí se ve variantách pro připojení jednoho enkodéru typ HCTL 2022, nebo dvou enkodérů HCTL 2032. Obvod obsahuje 32bitový čítač pro uložení hodnoty počtu pulsů. Pro přenos hodnoty vnitřního 32bitového čítače slouží 8bitová sběrnice, která je pomocí 2bitů multiplexovaná. Obvod podporuje kaskádové zapojení obvodu pro zvýšení rozsahu čítače pulsů. Obvod pracuje s 33 MHz hodinovým signálem. Cena obvodu HCTL 2022 se v současné době pohybuje kolem 7 eur, HCTL 2032 cca 10 eur .

Mikrokontrolér STM32F051 obsahuje řadu časovačů na které je možné připojit enkodéry, ale pouze jeden 32bitový čítač. MCU nepotřebuje realizovat přenos z externího zařízeni, hodnota pulsů je ukládána v interním registru. Hodinový kmitočet mikrokontroléru je max. 48 MHz. Cena mikrokontroléru je v současné době cca 3 eur.

Pokud bychom tyto dva obvody porovnali. v případě uvážení potřeby dalšího zařízení pro zpracování dat v případě obvodu HCTL 2022, jednoznačně je zde patrná výhoda použití mikrokontroléru. Nepatrnou výhodu má obvod HCTL 2032, který umožňuje sledovat dva enkodéry, zde bychom mohli použít mikrokontrolér s vyšším výkonem například z řady STM32F2xx, který obsahuje dva 32bitové čítače. Cena dekodéru a mikrokontroléru je na stejné úrovni. Opět je tedy výhodné použití mikrokontroléru s výhodou možnosti přímého zpracování dat z enkodéru.

9. Funkční DDS generátor

9.1 Princip DDS generátorů

DDS - direct digital synthesis je jednoduchou, snadno realizovatelnou technikou pro dosažení generování libovolného signálu s velkým rozsahem přeladitelnosti. Zároveň jsou generátory tohoto typu velmi stabilní z důvodu kmitočtu odvozeného číslicově od kmitočtu krystalu. Tuto vlastnosti kazí pouze větší jitter signálu.

Princip DDS je založen na lineární změně fáze během generování signálu. Pokud budeme uvažovat sinusový signál popsaný vzorcem.

$$f(t) = \sin(x(t)), \tag{9.1}$$

můžeme funkci $x(t)$ představující okamžitou fázi signálu, rozložit na složky

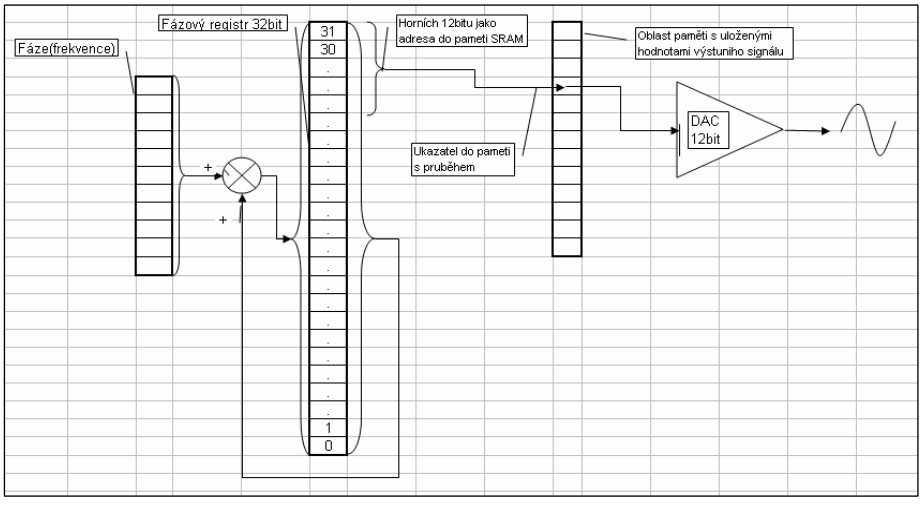
$$x(t) = a(t) + b, \tag{9.2}$$

kde $a(t)$ představuje frekvenci a b fázi signálu. Pokud bychom konstantu b lineárně měnili se změnou času t dosáhneme požadované změny frekvence.

Tuto změnu fáze je možné dosáhnout pravidelným přičítáním konstanty (fáze) vyjadřující požadovanou frekvenci k hodnotě adresy fázového registru. Výsledek se uloží do fázového registru a jeho hodnota se použije v dalším cyklu pro nový součet. Horní adresa fázového registru je ukazatelem do pole dat s uloženým tvarem signálu. Hodnota dat, na které tento ukazatel ukazuje je použita pro generování výstupu periferií DAC. Princip je graficky vysvětlen na obr. 9.1.

Nevýhodou generátoru na principu DDS je vynechání dat ze souboru vzorků signálu při generování vyšších frekvencích a naopak malý počet vzorků při generování nízkých frekvencích. Mezi výhody patří snadná realizace změny frekvence, změna v celém rozsahu frekvencí po velmi malých krocích, možnost okamžité změny frekvence, generování libovolných signálů závislých na obsahu paměti pro signál.

Generátory na principu DDS jsou v praxi používány např. pro modulaci FSK, v mobilních telefonech, v generátorech signálů, apod.



Obr. 9.1 Princip DDS generátoru.

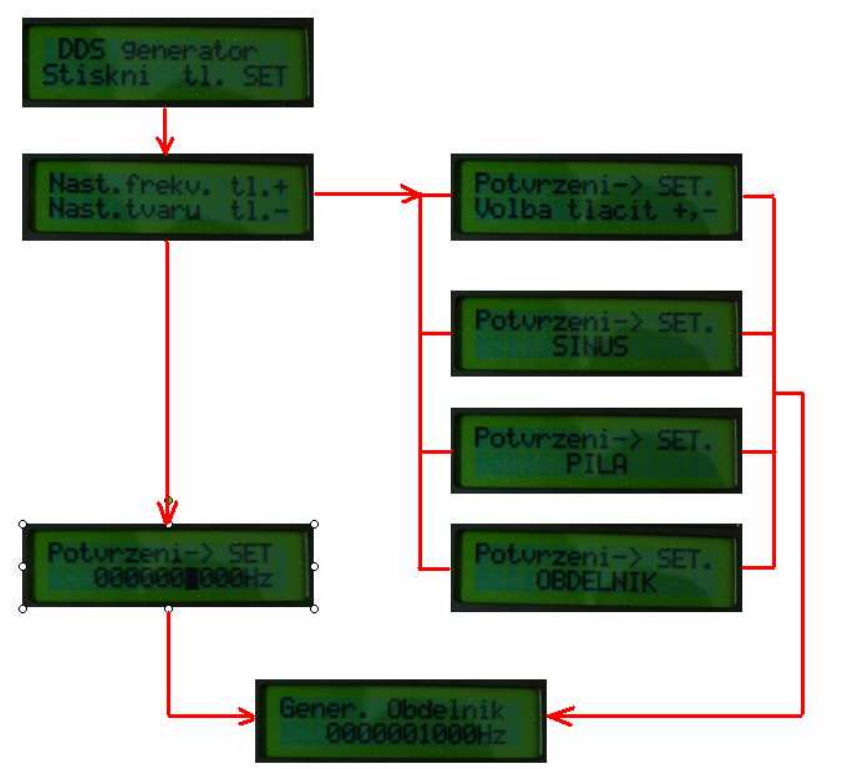
9.2 Ovládání a menu DDS generátoru

Ovládání generátoru je řešeno pomocí tří tlačítek SET, +, - a pomocí menu dvou-řádkového LCD displeje. Na obr. 9.2 je zobrazeno úplné schéma ovládání generátoru.

Po zapnutí přístroje je zobrazen na displeji název programu „DDS generator“ a pod ním výzva ke stisku tlačítka „SET“. Pokud dojde ke stisku tohoto tlačítka, je zobrazena nabídka nastavení s možností volby frekvence, nebo tvaru výstupního signálu. Pokud stiskneme tlačítko „+“ přejdeme do nastavení požadované frekvence generátoru. Dalším stiskem tlačítek „+“, nebo „-“ inkrementujeme nebo dekrementujeme hodnotu s vahou řádu na kterém se nachází kurzor (blikající symbol obdélníku). Přesunutí kurzoru na vyšší řády hodnoty frekvence se provádí jednotlivými stisky tlačítka “SET“. Pro opuštění nastavení frekvence je nutné opakovaným stiskem tlačítka projít přes všechny řády nastavení frekvence, dokud nedojde k zobrazení dalšího okna displeje s informativními údaji o právě generovaném signálu. Rozsah nastavení frekvence je omezen programem na hodnoty v mezích 0-500000Hz.

Druhou možností a nutnou podmínkou je nastavení tvaru výstupního signálu. Pro výběr jsou k dispozici tři varianty generovaných signálů sinus, pila a obdélník. Způsob volby je znázorněn v pravé části obr. 9.2. Potvrzením tlačítkem “SET“ potvrdíme nabídku tvaru generovaného signálu. Následně dojde ke spuštění generátoru a zobrazení okna displeje s údaji o generovaném signálu.

V průběhu generování signálu můžeme kdykoli stiskem tlačítka “SET“ přejít ke změně nastavení parametrů generátoru. Po dobu nastavování je pozastavena činnost generátoru.



Obr. 9.2 Schéma ovládání generátoru.

9.3 Popis bloků a funkce programu

Program generátoru byl napsán v prostření uVision4 a obsahuje níže uvedené bloky.

MAIN.c – volání funkcí z bloku stm32f0xx_my_config.c nastavující periferie.
– smyčka realizující DDS generátor.

stm32f0xx_my_config.c – obsahuje funkce pro nastavení periferií.

stm32f0xx_LCD.c – knihovna s funkcemi pro obsluhu a tisk na LCD 2x16 znaků displej.

obsluha_klaves_generator.c – obsahuje funkce pro řízení rozhranní uživatele(tlačítka, displej).

stm32f0xx_gpio.c – standardní knihovna pro práci s vstupy, výstupy mikrokontroléru.

stm32f0xx_rcc.c – standardní knihovna pro práci s hodinami pro periferie a procesor.

stm32f0xx_it.c – obsahuje funkce pro obsluhu přerušení.

stm32f0xx_exti.c – nastavení parametrů externího přerušení.

Funkce a nastavení programu

Po nastavení parametrů požadované frekvence a požadovaného tvaru výstupního signálu, který je po volbě vypočítán a nahrán do paměti SRAM mikrokontroléru je spuštěna nekonečná smyčka provádějící v každém cyklu součet fáze s aktuální hodnotou fázového registru, výsledek je uložen ve fázovém registru a horní část adresy tohoto výsledku je použita jako hodnota pro nastavení výstupu DAC převodníku. Tato smyčka je periodicky vykonávána a lze ji přerušit stiskem tlačítka PA0. Stisk tlačítka vyvolá přerušení, povolí se hodiny do periferií povolí se přerušení od periferie SYSTICK, která je použita pro generování různě dlouhých zpoždění potřebných k ovládání displeje a tlačítek. Po nastavení a opuštění obsluhy přerušení jsou povolené periferie opět zablokovány.

Nastavené konstanty

V programu je možné nastavovat pomocí změny proměnných a konstant tyto parametry:

V bloku *MAIN.c*

```
#define Ns 1024        //  počet vzorku generovaného průběhu
#define Rozl 4096      // maximální rozlišeni DAC převodníku 0-4095~12bitů
#define P2P  150       // Peek to peek - úprava rozkmitu zmenšení výkyvu průběhu z důvodu
                          chyb při výpočtu, platí jen pro sinusovku
int scale_freq=2240;  // převodní konstanta nastaveni fáze(frekvence) změna nutná při
                          změně frekvence CLK CPU
```

V bloku *obsluha_klaves_generator.c*

```
#define max_frekv 500000.00      // hodnota maximální nastavitelné frekvence
int pocet_radu=6;                // počet řádů nastaveni frekvence
```

9.4 Zapojení LCD displeje a tlačítek generátoru

Generátor se skládá ze třech celků. Prvním je vývojový kit STM32F0-Discovery výrobce STMicroelectronics. Druhým 2x16 znakový LCD displej MC1602E s řadičem HD44780 popis funkce a vývodů ve zdroji [18]. Třetím celkem jsou dvě mikrotlačítka typu NO(normally open).

Mikrotlačítka jsou připojena jedním vývodem na pin GND modulu Discovery, druhý vývod je připojen v případě tlačítka "+" na pin PA1, pro tlačítko "-" na pin PA2 modulu Discovery. Na modulu STM32F0 je využito tlačítko „USER" připojené na pin PA0 jako tlačítko „SET".

Zapojení displeje je pro všechny přístroje stejné, popis tohoto zapojení se nalézá v příloze této DP.

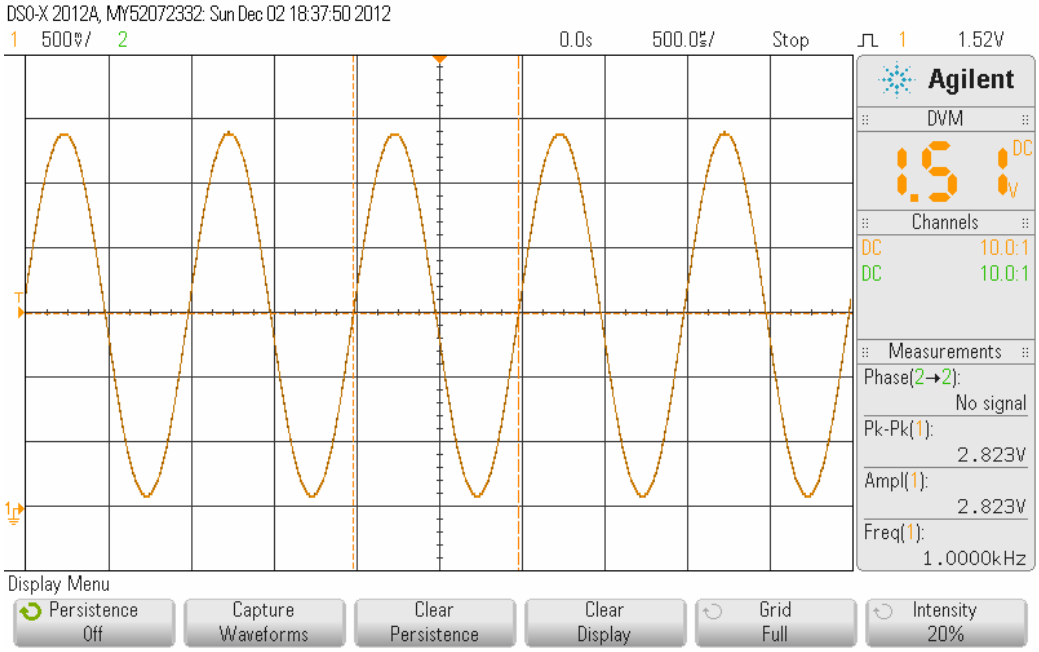
9.5 Parametry DDS generátoru

Rozsah nastavitelných frekvencí.....0-500 kHz
Nastavitelný krok frekvence.....1 Hz
Napětí peek to peek.....0-2,95 V
Nastavitelné průběhy .....Sinus, Pila, Obdélník
Počet dat na průběh.....1024
Rozlišení DAC převodníku výstupu.....4096 úrovní

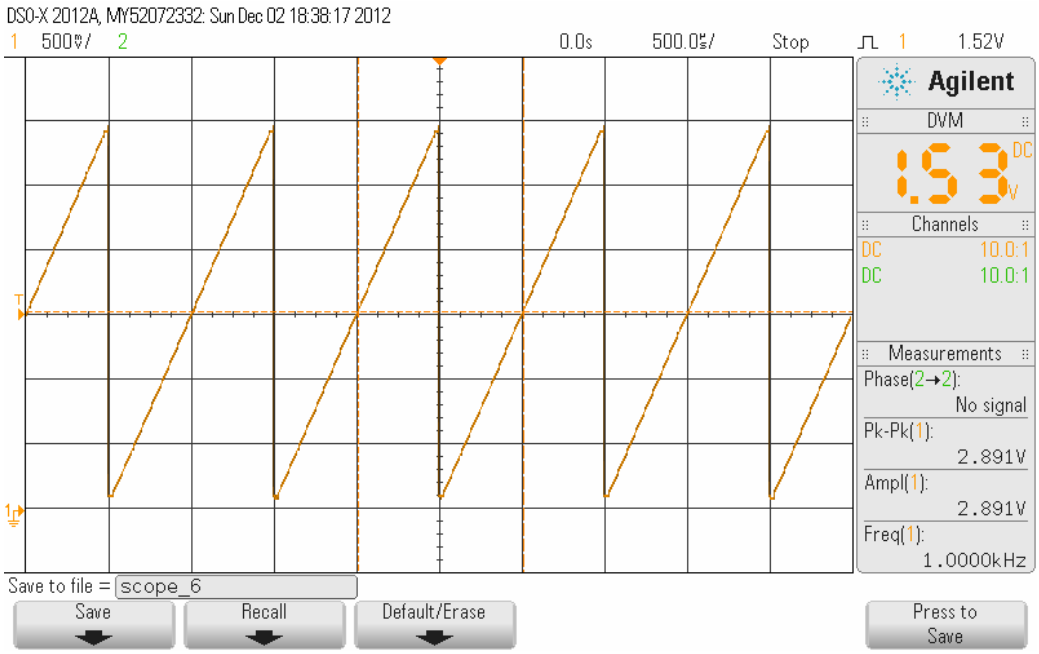
Výpočet sinusového průběhu probíhá podle Taylorova polynomu 11 stupně., viz vztah (9.3).

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \frac{x^{11}}{11!} \tag{9.3}$$

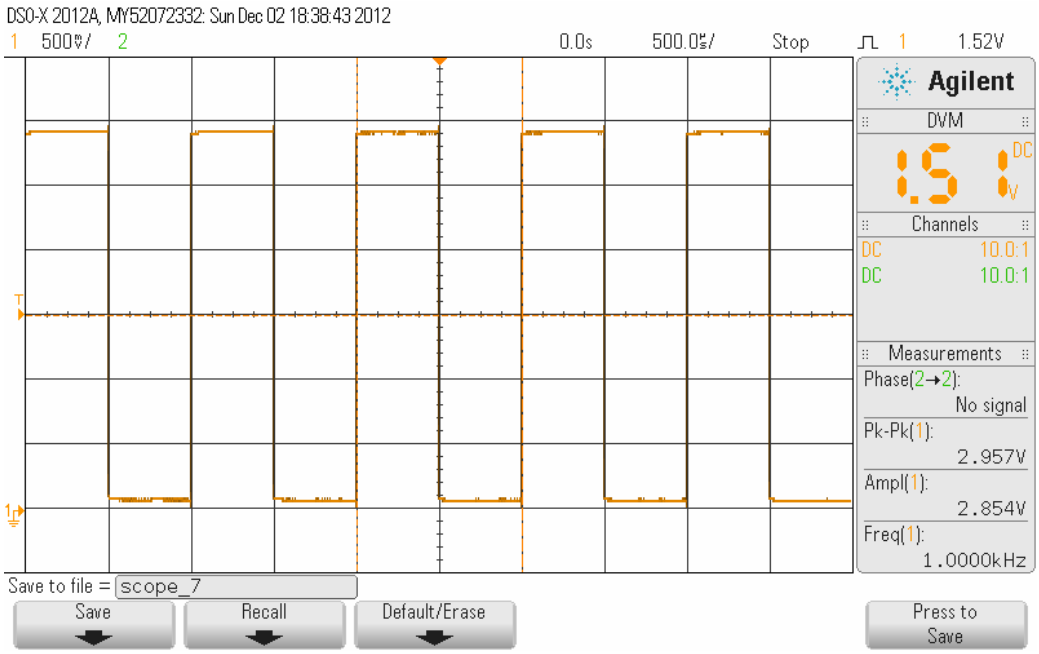
9.6 Naměřené průběhy z výstupu generátoru



Obr. 9.3 Průběh sinusového signálu z generátoru při nastavené frekvenci 1kHz.



Obr. 9.4 Průběh pilového signálu z generátoru při nastavené frekvenci 1kHz.



Obr. 9.5 Průběh obdélníkového signálu z generátoru při nastavené frekvenci 1kHz.

9.7 Frekvenční omezení generátoru

Omezení max. frekvence z obnovovací frekvence hodnoty výstupu

Obnovovací rychlost dat, refresh, neboli změna hodnoty fázového registru byla naměřena pomocí osciloskopu jako 550 ns viz obr. 9.6 Z toho vyplývá první omezení maximální frekvence podle Nyquitsova kritéria

$$f_{sample} = 2 \cdot f_{measure} \approx 1MHz\,. \tag{9.4}$$

Omezení max. frekvence na základě katalogových údajů

Druhé omezení max. frekvence pochází z datasheetu obvodu STM32F05x [2], str.79, který popisuje přeběh DAC převodníku od nejnižší po nejvyšší hodnotu výstupu DAC při 10 bitovém rozlišení a konečné nepřesnosti napětí ± 1 LSB na trvání typicky 3 us. Opět s ohledem na Nyquitsovo kritérium potřebujeme nejméně dva vzorky. Maximální frekvence generátoru by tedy byla omezena na

$$f_{\max} = \frac{1}{(3+3)\cdot 10^{-6}} = 200kHz\,. \tag{9.5}$$

Omezení max. frekvence na základě rychlosti přeběhu

Měřením byla zjištěna rychlost přeběhu nezahrnující dobu ustálení výstupu jako $SR = \frac{2{,}85V}{1\cdot 10^{-6}s}$ to by odpovídalo omezení až na (9.6)

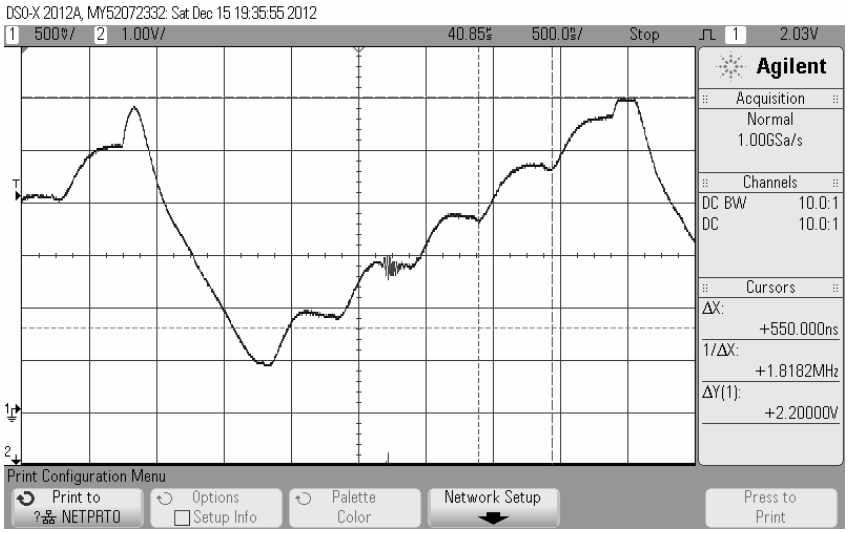
$$f_{\max} = \frac{1}{(1+1)\cdot 10^{-6}} = 500kHz \tag{9.7}$$

Omezení max. frekvence na převodní konstanty

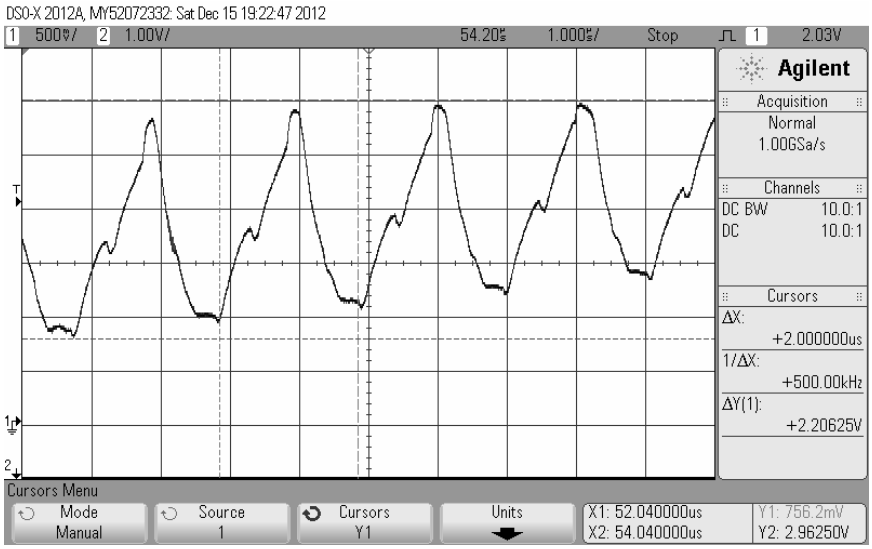
Hodnota převodní konstanty byla zjištěna pomocí měření osciloskopem. Touto konstantou se násobí zadávaná frekvence, výsledek tvoří fázi přičítanou v každém cyklu k hodnotě fázového registru.

Omezení generování hodnoty fázovým registrem vycházející z požadavku dvou vzorku na periodu

$$\frac{PocetbituFazovy\,\mathrm{Re}\,g}{pocetVzorku \cdot prevodniKonst.} = \frac{2^{32}}{2 \cdot 2240} \approx 960kHz \tag{9.8}$$



Obr. 9.6 Měření obnovení hodnoty DAC převodníku



Obr. 9.7 Generování průběhu pilového signálu 500kHz 4 vzorky na periodu.

9.8 Porovnání vlastností DDS generátoru s obvody AD5930, ADuC7128

Obvod AD5930 je programovatelný DDS generátor s výstupní frekvencí do 25 MHz. Tento obvod je schopný generovat sinusový, trojúhelníkový a obdélníkový signál. Umožňuje změnu frekvence a fáze výstupního signálu ve dvou režimech. Změna proběhne buď okamžitě po přijaté žádosti a nebo je provedena až v okamžiku, kdy je uhlová frekvence v bodě násobku 2Pi. Řízení obvodu je možné pomocí sériového rozhranní. Pro převod číslicové hodnoty na analogovou je v obvodu implementován 10bit DAC převodník. Kompletní popis obvodu je k dispozici v datasheet [19]. Cena obvodu je v současné době cca 8$.

Tento obvod má proti generátoru s DDS vyšší frekvenci výstupního signálu, možnost změny výst. signálu v celočíselných násobcích 2Pi úhlu frekvence. Navržený DDS generátor s STM32F051 používá vyšší rozlišení DAC převodníku (12bit). Další výhodou je možnost změny signálu změnou výpočtu signálu, v navrženém DDS generátoru se na začátku volby tvaru signálu nejprve signál vypočítává a teprve potom ukládá do paměti. Cena mikrokontroléru STM32F051 je k současné době velmi nízká cca 2,5$.

ADuC7128 je mikrokontrolér fy. Analog Devices s jádrem ARM7TDMI, který má vestavěnou periferii DDS generátoru. V obvodu, je podobně jako u předcházejícího generátoru, použit opět 10bit DAC převodník. Obvod je schopný generovat pouze sinusový signál s frekvencí do 21 MHz. Dále je možné nastavení fáze generovaného signálu. Kompletní popis obvodu je k dispozici v datasheetu [20]. Cena obvodu je v současné době cca 9$.

10. Datalogger

Datalogger je zařízení sloužící pro sběr dat převedených na elektrické napětí. Data jsou vzorkovány ADC převodníkem, ukládány a poté mohou být přeneseny k zobrazení do PC. V dalším textu je popsána realizace tohoto přístroje.

10.1 Popis realizace zařízení

Zařízení bylo realizováno na procesoru ST32F051R8 s využitím vývojového kitu STM32F0-Discovery výrobce STMicroelectronic.

Jádro použitého programu se skládá ze čtyř hlavních celků zajištujících funkci vzorkování vstupu dataloggeru. Jsou to časovač, ADC převodník, DMA jednotka s alokovanou pamětí SRAM pro uložení vzorků. Na obrázku obr. 10.1 jsou zachyceny tyto celky s naznačenou dále podrobněji popsanou funkcí.

První periferie, časovač TIM2 má funkci spuštění odměru. Nastavení intervalu vzorkování provádíme pomocí volby menu, nastavení „Interval vzork.“, zapsaná hodnota je uložena do registru TIM2_ARR. Při události „Update event“, tj. při dosažení nulového stavu registru čítače TIM2_CNT, dojde k automatickému znovu-naplnění (reloadu) hodnoty registru TIM2_ARR do registru TIM2_CNT. Při této události je časovačem generován puls TRG0, spouštějící jeden odměr sledovaného signálu periferií ADC.

Periferie ADC je propojena s jednotkou DMA pracující nezávisle na CPU. Tato jednotka má za úkol při zápisu periferie ADC do registru ADC_DR tuto hodnotu přečíst a uložit údaj na definovanou adresu v paměti SRAM CPU. V případě že není periferie schopná data přečíst před dalším odběrem, tedy obsluhovat vybírání z ADC je chod periferie DMA jednotkou ADC zastaven. Tím je zaručena, konzistivita načtených dat v paměti. Tuto vlastnost popisuje manuál [1] na str.186 odst. „Managing converted data using the DMA“.

Obr. 10.1 Princip funkce dataloggeru.

10.2 Použité periferie mikrokontroléru

Pro realizaci dataloggeru byly použity následující periferie:

GPIOA - obecný port vstupů výstupů.

Využity piny:
- PA0 - tlačítko “SET“
- PA1 - tlačítko “+“
- PA2 - tlačítko “-“
- PA9 - USART TX-line
- PA10 - USART RX-line

GPIOB - obecný port vstupů výstupů, je využit pro připojení LCD displeje, popis zapojení viz příloha.

GPIOC - obecný port vstupů výstupů.

Využity piny: PC4 - vstup dataloggeru
PC8 - výstup LED dioda mění stav při vzorkováni

TIM1 - jednotka časovače použita pro spouštění odběru vzorků.

ADC - jednotka analogově-digitálního převodníku pro převod vzorků.

DMA - jednotka přímého přístupu k paměti, pro rychlé ukládání do paměti nezávislá na CPU.

USART - jednotka univerzálního sériového rozhranní, pro přenos dat do PC.

EXTI - externí přerušení, použit pro tlačítko vstupu do ovládaní programu.

RCC - nastaveni hodin pro periferie, TIM1-clock je 48MHz, zdroj HSE krystal.

NVIC - spravuje vektory přerušení.

10.3 Ovládání a nastavení dataloggeru

10.3.1 Ovládání menu

Ovládání dataloggeru je řešeno pomocí tří tlačítek SET, +, - a pomocí menu dvou-řádkového LCD displeje. Na obr. 10.2 je zobrazeno úplné schéma ovládání dataloggeru. Na obrázku jsou také vyznačeny podmínky přechodů na další LCD obrazovky ve formě textu (SET, +, -) u příslušných šipek.

Po zapnutí přístroje je zobrazen na displeji v horním řádku název programu „Datalogger“ a pod ním výzva ke stisku tlačítka “SET“. Pokud dojde ke stisku tohoto tlačítka, je zobrazena nabídka nastavení parametrů dataloggeru. Po jednotlivých položkách menu se můžeme pohybovat pomocí stisknutí tlačítek “+“, nebo“-“, viz obr. 10.2.

10.3.2 Nastavení počtu vzorků

Pro zahájení měření bychom měli nejprve nastavit položku „Pocet vzorku“. Posuneme se tedy v nabídce pomocí stisků tlačítek „+“ nebo „–„ na zmíněnou položku a následným stiskem tlačítka „SET“ přejdeme k nastavení hodnoty počtu vzorků. Hodnota může být nastavena v rozmezí 0-60000.

Tlačítky +,– inkrementujeme nebo dekrementujeme hodnotu s vahou řádu na kterém se nachází kurzor (blikající symbol obdélníku). Přesunutí kurzoru na vyšší řády hodnoty provádíme jednotlivými stisky tlačítka „SET“. Po projití všech pozic nastavované hodnoty opakovaným stiskem tlačítka SET dojde k návratu na první obrazovku nastavení parametrů dataloggeru.

10.3.3 Nastavení intervalu odběru vzorků

Pro zadání intervalu odebírání vzorků nastavíme v menu položku „Interval vzork.“. Po stisknutí tlačítka SET se zobrazí nabídka časového rozlišení v mikrosekundách, milisekundách, sekundách a hodinách. Před zvolenou položkou je zobrazen symbol hvězdičky “*“, změna je možná stisknutím tlačítek +, nebo -. Dalším stiskem tlačítka SET vstoupíme k nastavení hodnoty intervalu. Hodnota nastavení je pro všechny rozsahy omezena programem na hodnoty od 0 do 99999.

10.3.4 Spuštění odměrů

Po nastavení počtu a intervalu odběru vzorků, potvrzením položky menu „SPUST ODMERY“ spustíme činnost časovače. Vstup ADC převodníku na pinu PC4 bude vzorkován podle zadaných parametrů. Vzorkování je detekováno změnou stavu pinu PC8, na který je připojena indikační LED dioda.

Na displeji se objeví nové údaje, v horním řádku vlevo s označením „V:“ zadaný počet vzorků k odměru, vpravo s označením „I:“ hodnota intervalu měření. Před písmenem I: na displeji je zobrazena aktuální jednotka intervalu dat z množiny u(us), m(ms), s , h. Zobrazený symbol „*“ před písmenem“V:“, informuje uživatele, že právě probíhá měření, po ukončení symbol zmizí.

V dolním řádku jsou data během odměru aktualizována u označení „R“, zbývající počet k vzorkování, vpravo s označením „A:“ stav registru ADC převodníku s hodnotou při sejmutou při odečtení odměru.

10.3.5 Přerušení vzorkování

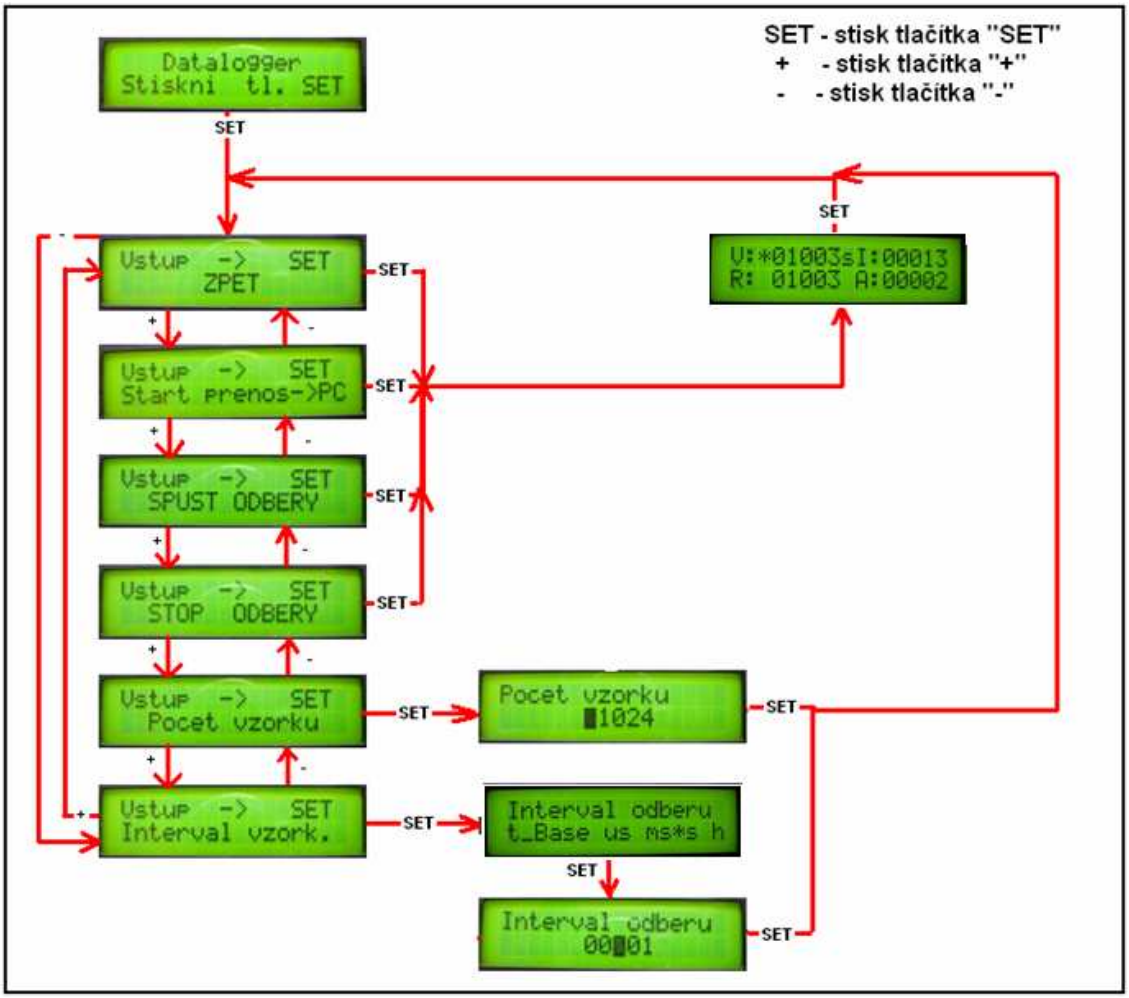
Vzorkování je možné přerušit během odebírání vzorkování v menu potvrzením volby „STOP ODMERY“, obnovení vzorkování od bodu přerušení je možné potvrzením položky „SPUST ODMERY“.

10.3.6 Přenos dat do PC

Po ukončení měření je možné data přenést pomocí rozhranní USART mikrokontroléru do počítače k dalšímu zpracování. Přenos je aktivován potvrzením volby „Start prenos->PC“. Pro testovací účely byl použit software NRM_Display odkaz [21], obr. 10.3.

Přenos probíhá následujícím způsobem: Nejprve je vyslán znak 0x30 pro zahájení přenosu. Pokud software přijme tento znak a je v režimu single očekává, že budou dále odesílány vzorky v délce zadané v okně programu „Samples“. Přístroj bez potvrzení vysílá vzorkované znaky z paměti do PC. Po načtení zadaného počtu vzorků v programu NRM display jsou data zobrazeny na obrazovce.

Přenosová linka je řešená převodníkem standardu RS232 s využitím signálů RX,TX,GND s parametry 9600Baud, 8 bit data, 1stop bit, bez parity.



Obr. 10.2 Schéma ovládání dataloggeru.

10.4 Popis bloků a funkce programu

Program byl napsán v prostření uVision4 a obsahuje níže uvedené bloky.

MAIN.c – volání funkcí z bloku stm32f0xx_my_config.c nastavující periferie.
– smyčka obnovující stav LCD displeje, zajišťující chod programu.

stm32f0xx_my_config.c – obsahuje funkce pro nastavení periferií.

stm32f0xx_LCD.c – knihovna s funkcemi pro obsluhu a tisk na LCD displej 2x16 znaků.

obsluha_rozhrani_HMI.c – obsahuje funkce pro řízení rozhraní uživatele (tlačítka, displej) a aktualizaci nastavení čítače.

stm32f0xx_gpio.c – standardní knihovna pro práci s vstupy, výstupy mikrokontroléru.

stm32f0xx_rcc.c – standardní knihovna pro práci s hodinami pro periferie a procesor.

stm32f0xx_it.c – obsahuje funkce pro obsluhu přerušení.

stm32f0xx_exti.c – nastavení parametrů externího přerušení.

stm32f0xx_usart.c – knihovna pro obsluhu rozhraní USART.

stm32f0xx_adc.c – knihovna funkcí pro řízení ADC převodníku.

Konstanty nastavení programu:

V programu je možné nastavovat pomocí změny proměnných a konstant tyto parametry:

V bloku *MAIN.c*

```
#define Ns 1024            //  počet vzorku generovaného průběhu
#define Rozl 4096          // maximální rozlišeni DAC převodníku 0-4095~12bitů
#define P2P  150           // Peek to peek-úprava rozkmitu zmenšení výkyvu průběhu
                           z důvodu chyb při výpočtu, platí jen pro sinusovku
int scale_freq=2240;       // převodní konstanta nastaveni fáze(frekvence) změna nutná
při                              změně frekvence CLK CPU
```

V bloku *obsluha_klaves_generator.c*

```
#define max_frekv 500000.00    // hodnota maximální nastavitelná frekvence
int pocet_radu=6;              // počet radu nastaveni frekvence
```

10.5 Elektrické zapojení dataloggeru

Datalogger se skládá ze třech fyzických celků. Prvním je vývojový kit STM32F0-Discovery výrobce STMicroelectronics. Druhým 2x16 znakový LCD displej MC1602E s řadičem HD44780 popis funkce a vývodů ve zdroji [18]. Třetím celkem jsou dvě mikrotlačítka typu NO(normally open).

Mikrotlačítka jsou připojena jedním vývodem na pin GND modulu Discovery, druhý vývod je připojen v případě tlačítka "+" na pin PA1, pro tlačítko "-" na pin PA2 modulu Discovery.

Na vývojovém kitu je využito tlačítko „USER", připojené na pin PA0, jako tlačítko „SET".

LCD displej pro zobrazení hodnot nastavení, je připojen prostřednictvím portu GPIOB. V příloze je uveden podrobný popis zapojení těchto pinů.

10.6 Technické parametry dataloggeru

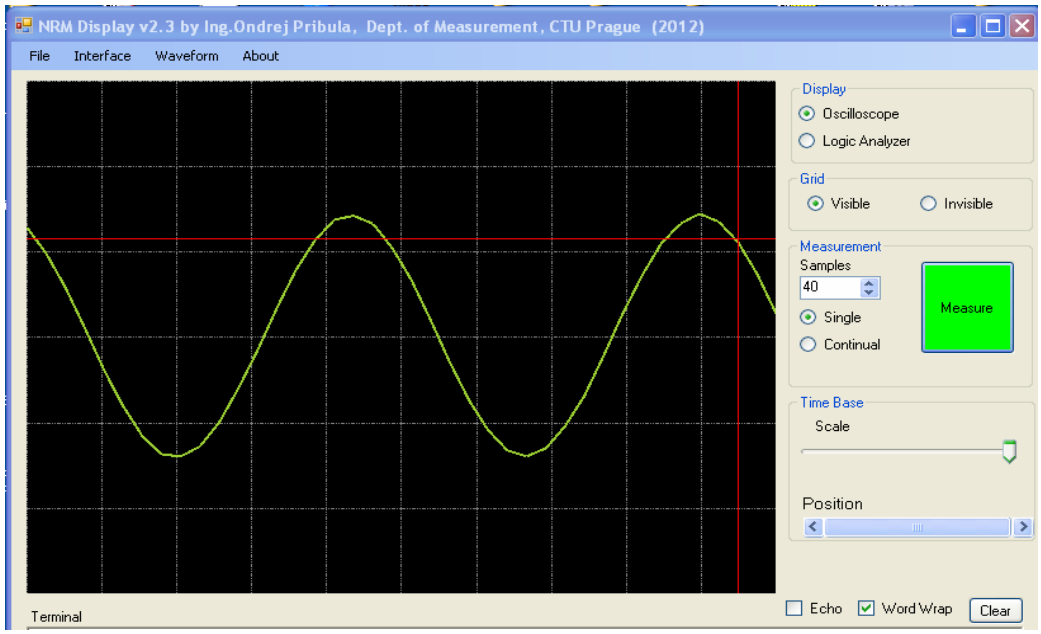
Maximální počet vzorků.....6000
Typ paměti pro uložení vzorků.....volatile SRAM
Způsob generování frekvence vnitřních hodin.....krystal 8MHz
Nastavitelné intervaly vzorkování pro rozsahy:

- us..... od 0 us do 99999 us
- ms..... od 0 ms do 99999 ms
- s..... od 0 s do 99999 s
- h..... od 0 h do 1193 h

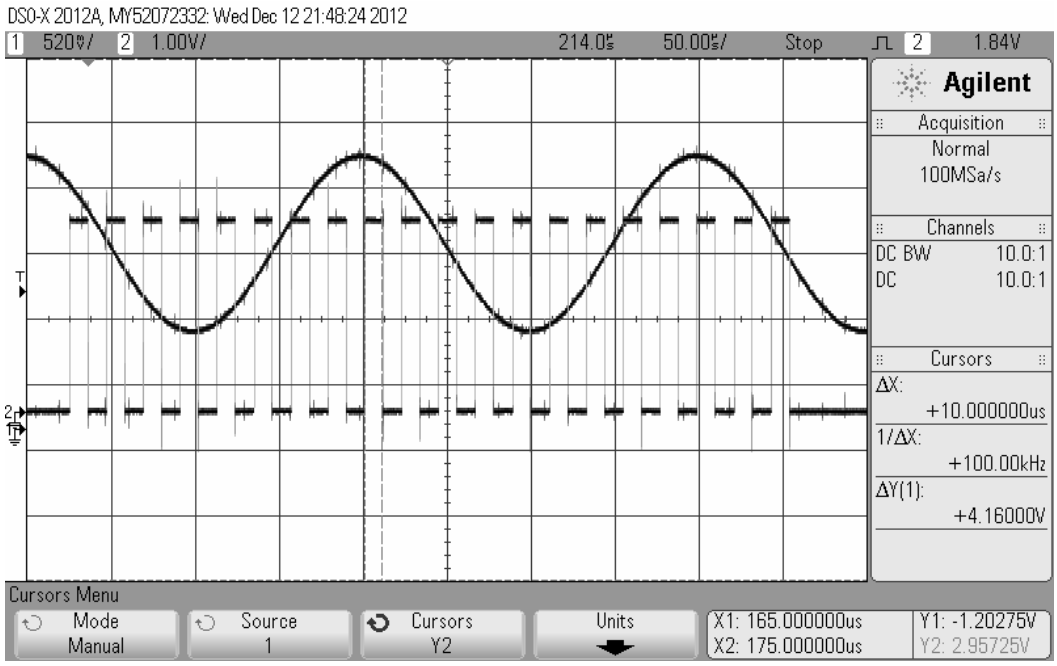
Rozsah vstupních napětí pin PC4.....0-3,0 V
Rozlišení ADC převodníku .....256 úrovní(8bitů)
Export dat do PC.....RS232
Ovládáni dataloggeru.....LCD display, tlačítka
Maximální vzorkovaná frekvence (Nyquitsovo kritérium) 250Khz

10.7 Test dataloggeru

Na obrázku 10.3 je zobrazen průběh získaný pomocí dataloggeru. Na obr. 10.4 potom zdrojový signál s naznačenými okamžiky vzorkování. Vzorkování je svázáno se změnou signálu 2. Pro zobrazení signálu v PC byl použit software [21].



Obr. 10.3 Rekonstrukce sinusového signálu 5 kHz vzorkovaného 10 us intervaly po přenosu v PC programem NRM_Display.



Obr. 10.4 Sinusový signál 5 kHz generovaný osciloskopem, obdélníkový signál generovaný dataloggerem, odpovídá vzorkovaní 10 us intervaly.

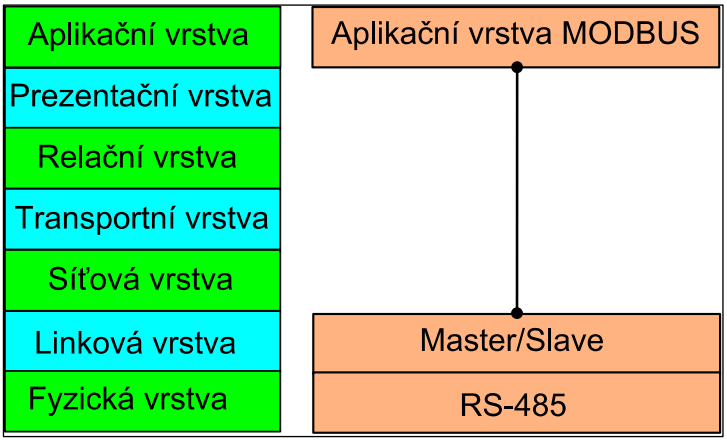
11. Spolupráce MCU ve společném přístroji

Tato kapitola se zabývá využitím spolupráce více mikrokontrolérů. Bude zde popsáno řešení funkce samostatných jednotek s mikrokontroléry, pracujících v menším systému s většími vzdálenostmi modulů. Dílčí jednotky budou schopny poskytovat řídící jednotce Master na vyžádání informace o stavu svých vybraných periferií. Získaná data jsou budou použita pro vizualizační účely .

11.1 Aplikační protokol MODBUS

V odstavci 2. „Rozbor řešení“ byl zmíněn návrh řešení multiprocesorové komunikace. Návrh popisuje části systému, které vychází ze známého sedmi-vrstvého modelu ISO/OSI, viz obr. 11.1. Pro řešení pomocí sériové linky a protokolu modbus se model zjednodušuje na tyto tři části.

1. Řešení aplikační vrstvy protokolem Modbus.
2. Způsob řízení sběrnice jeden master, více slaves.
3. Realizace fyzické vrstvy linku 485.



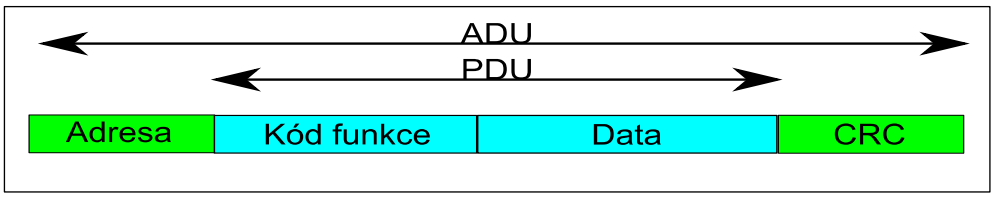
Obr. 11.1 Sedmi-vrstvý přenosový modul ISO/OSI.

Popis protokolu

Protokol Modbus je založen na principu klient/server, klientem může být například počítač s nainstalovanou vhodnou aplikací. Serverem, neboli poskytovatelem informací mohou být např. periferie typu PLC (Programable Logic Computer), různé senzory, vstupně/výstupní jednotky apod.. Další typickou vlastností protokolu je komunikace typu request/reply (požadavek/odpověď). Komunikaci zahajuje vždy Master, neboli klient zasláním požadavku serveru. Server přijme žádost a na tuto žádost sestaví odpověď, kterou pošle klientu který v určitém čase očekává odpověď na svůj dotaz.

Struktura zpráv Modbus

Zpráva se dle protokolu MODBUS obr. 11.2 skládá z části označené ADU (application data unit) a podmnožiny dat PDU (protokol data unit). Žádost i odpověď, začíná vždy adresou slave zařízení v délce 8 bitů. Tato adresa je stejná v rámci pro žádost i odpověď. Za adresou následuje 8 bitový kód funkce, který definuje obsah zprávy. Datová část označená na obr. 11.2 „DATA“ upřesňuje počet dat, nebo obsahuje požadovaná data, a může mít délku až 256 byte. Přesný obsah přenášených dat určuje kód funkce a směr přenosu (dotaz, odpověď). Bližší informace k obsahu MODBUS zpráv je nutné nastudovat z popisu protokolu [22].



Obr. 11.2 Modbus-rámec zprávy.

V navržené aplikaci jsou implementovány tři základní funkce dle specifikace protokolu Modbus.

První funkce viz obr. 11.3 má kód 0x03 a název „Read Holding Registers“. Funkci používám pro čtení informací uložených v registrech nebo v paměti mikrokontroléru.

Pro zápis do registrů a paměti MCU jsem se rozhodl použít fci 0x06 „Write Single Register“, viz obr.11.4. Tato funkce dle specifikace [2] může zapsat hodnotu jednoho registru o velikosti 16bitů.

Poslední třetí funkce s kódem 0x10 „Write Multiple Registers“, slouží k zápisu více registrů a používám ji pro zápis textu na LCD display.

Výše uvedené funkce patří mezi základní funkce protokolu Modbus a proto lze snadno z internetu získat již hotový software klienta, použitelný pro otestování funkce modulů pomocí PC.

Request

Function code	1 Byte	**0x03**
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	1 to 125 (0x7D)

Response

Function code	1 Byte	**0x03**
Byte count	1 Byte	2 x **N***
Register value	**N*** x 2 Bytes	

***N** = Quantity of Registers

Error

Error code	1 Byte	**0x83**
Exception code	1 Byte	01 or 02 or 03 or 04

Obr. 11.3 Formát MODBUS PDU rámce kódu funkce 0x03-Read Holding Registers, zdroj [1].

Request		Response	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	03	Function	03
Starting Address Hi	00	Byte Count	06
Starting Address Lo	6B	Register value Hi (108)	02
No. of Registers Hi	00	Register value Lo (108)	2B
No. of Registers Lo	03	Register value Hi (109)	00
		Register value Lo (109)	00
		Register value Hi (110)	00
		Register value Lo (110)	64

Obr. 11.4 Příklad zprávy s kódem funkce 0x03-Read Holding Register, zdroj [1].

11.2 Zabezpečení přenosu, kontrola chyb

V programu je prováděna dle specifikace protokolu Modbus kontrola dotazu. Schopnost odpovědět na dotaz se vyhodnocuje pomocí tzv. „Exception Code“, kódů vyjímek. Pro většinu Modbus funkcí existují 4 typy vyjímek s kódy 1 až 4. Význam jednotlivých kódů vyjímek není ale pro všechny funkce stejný. Společný kód pro všechny funkce má označení 1, s významem, že požadovaná funkce není podporována. Kódy 2 a 3 se týkají kontrol rozsahu požadovaných adres.

V případě detekce vyjímky slave posílá jednotce master v odpovědi kód příslušející této události. Detekce kódu 4, detekující neúspěšné zpracování požadavku nebyla implementována.

Zabezpečení celistvosti a neporušenosti zprávy je řešeno pomocí techniky doplnění zprávy CRC kódem . Pro výpočet CRC16 - Modbus kódu byla použita již existující knihovna ze zdroje [23]. V programu se provádí kontrola CRC kódu příchozí zprávy a výpočet CRC kódu pro odchozí zprávy.

11.3 Podpora komunikace mikrokontrolérem

Jak již bylo v předchozím textu zmíněno, v mikrokontroléru byl implementován program typu slave dle protokolu MODBUS [22]. Ten je schopen jednotce typu master odpovídat na dotazy s funkcemi 0x03, 0x06 a 0x10.

Výrobce mikrokontroléru vybavil pro snadné použití a zvýšení výkonu komunikace periferie obvodu řadou podpůrných vlastností. Některé z nich, níže popsané jsem použil při realizaci programu pro komunikaci mikrokontroléru.

11.3.1 Využití DMA periferie

V programu MCU byla periferie USART1 nadefinována tak, že spolupracuje s periferií DMA. Jsou vyhrazeny dva kanály DMA jeden pro příjem zprávy a druhý pro odeslání odpovědi. Pro použití DMA periferie jsem se rozhodl z důvodu, aby nebyl zatěžován procesor komunikací. Periferie DMA je periferií přímého přístupu k paměti, periferie USART potom nepotřebuje pro svou činnost procesor MCU.

11.3.2 Podpora protokolu Modbus mikrokontrolérem

Využil jsem podpory sběrnice Modbus popsané v manuálu [1], str.593 spočívající v zachycení ukončení příchozí zprávy, detekcí klidového stavu sběrnice. Doba klidového stavu je doporučena ve specifikaci protokolu Modbus [22], pro režim RTU minimálně 3,5 násobek doby trvání přenosu jednoho znaku. Po čase odpovídající tomuto klidovému stavu portu USART, je vyvoláno přerušení „recieve timeout“.

Modbus je použit v režimu RTU, který pro přenos jednoho hexadecimálního znaku použije 4 bity, je tak rychlejší oproti režimu ASCII, který pro ten samý přenos potřebuje 8 bitů.

Pro lepší pochopení uvedu příklad: Pro hexadecimální číslici „F“, režimem RTU pošleme binární kombinaci „1111“, v režimu ASCII odešleme odpovídající znak „F“ , tj. 0x46 převedeno do hexadecimálního formátu a 0100.0110 v binárním formátu osmi bity.

11.3.3 Využití podpory multiprocesorové komunikace procesorem

Mikrokontrolér nabízí dvě možnosti podpory multiprocesorové komunikace popsané v manuálu RM0091 zdroj [1], str.592 :

1. Address detection – Rozpoznání adresy probíhá s využitím bitu, který je přidáván navíc ke každému byte přenosu, nejčastěji se používá 9 bit. Pokud je tento bit ve stavu log.1, všechny stanice jsou přepnuty do režimu naslouchání, protože zbývající bity jsou adresou zařízení, pro které je zpráva určena. Pokud je bit ve stavu log.0 stanice zůstávají v režimu MUTE (klidový režim), kromě stanice, která komunikuje s jednotkou master.
   Tento režim jsem bohužel nemohl použít z důvodu absence podpory 9bitové komunikace ze strany programovacího prostředí v PC.

2. IDLE detection - Aktivací bitu MMRQ registru USART_RQR, můžeme pomocí software přepnout periferii USART do režimu MUTE. Na konci přenosu, když je linka určitou dobu v klidu, dojde hardwarem k rozpoznání tohoto stavu a k přerušení MUTE módu. Stanice je přepnuta do normálního módu a čeká na start bit viz manuál [1] str. 592.

11.4 Popis komunikace MCU

Jednotka slave, tvořená modulem s mikrokontrolérem STM32F051R8 musí komunikovat s nadřízenou jednotkou master. Pro tento úkol bylo nutné vytvořit vhodné nastavení periferií mikrokontroléru. Popis tohoto nastavení je uveden v dalším textu.

11.4.1 Nastavení registrů periferií USART a DMA

Pro komunikaci se zařízením master byla použita periferie sériového rozhranní USART1 s nastavenými parametry přenosu 9600 baud ,2 stop bity.

Povolením bitu RTOIE , bylo docíleno spouštění rutiny přerušení v případě, že po příjmu dat do registru USART->RDR uplyne nadefinovaný čas zapsaný v registru USART->RTOR a mezitím nedojde k příjmu nových dat (dle protokolu Modbus min. 3,5 znaku tj. při rychlosti přenosu 9,6 kBd cca 1,1 ms).

V periferii USART1 bylo dále povoleno použití DMA periferie přímého přístupu do paměti pro oba směry, příjem a odesílání dat. Použitím periferie DMA je možné vyloučit z účasti na přenosu procesor mikrokontroléru. Bližším popisem komunikace se zabývá následující odstavec „Postup zpracování zpráv“.

11.4.2 Postup zpracování příchozí zprávy

V následujícím textu je v bodech popsána komunikace jednotky master, kterou tvoří PC s programem napsaným v jazyce C# v programovacím prostředí Microsoft Visual studio 2010 s jednotkou slave tvořenou mikrokontrolérem STM32F051R8.

Komunikace je typu dotaz - odpověď. Dotaz produkuje PC jako master, odpověď poskytuje mikrokontrolér v roli slave.

Počáteční podmínky nastavení vybraných periferií MCU:

Povoleno přerušení od události RXNEI (tzn. registr pro přijaté zprávy USART není prázdný).

Nastaveny parametry kanálu periferie DMA pro příjem zprávy z jednotky USART, DMA kanál pro RX ponechán ve stavu disable.

Postup komunikace společná část:

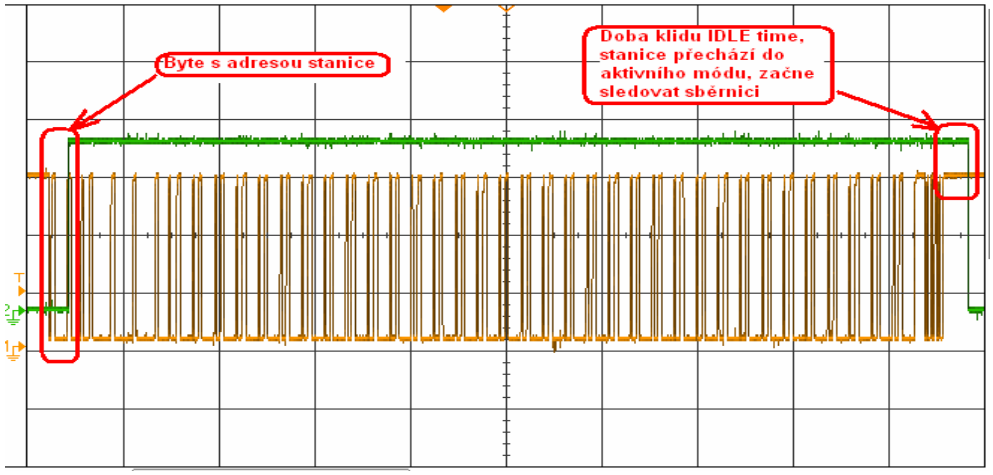
1. Master vyšle dotaz pro stanici na linku 485.
2. V STM modulu je vyvoláno přerušení RXNEI, tzn. RDR registr USART1 byl naplněn daty.
3. Porovnává se adresa přijatá v RDR registru s adresou zařízení.

Postup v případě, že adresa v dotazu odpovídá adrese stanice :

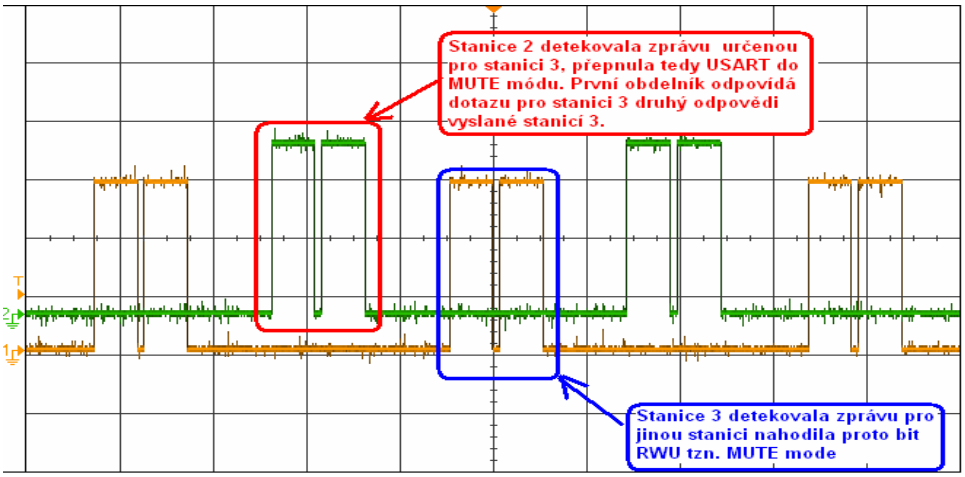
4. Nastaví se povolení přerušení „reciever timeout“ nastavením bitu RTOIE.
5. Nastaví se povolení DMA periferie bitem EN, tím dojde k příjmu zprávy a uložení zprávy do paměti.
6. Ukončení příjmu zprávy je detekováno vyvoláním přerušení „reciever timeout“. To je vyvoláno na základě klidového stavu sběrnice po přijetí dotazu viz obr.1. V těle obsluhy přerušení dojde k volání funkce, která zpracuje přijatý dotaz, vytvoří odpověď a odešle ji pomocí DMA periferie na kanál TX periferie USART.
7. Provede se znovu-nastavení kanálu DMA pro příjem nové zprávy, činnost periferie kanálu se ale zatím nepovolí, zůstává disable.

Postup v případě, že adresa v dotazu neodpovídá adrese stanice :

4. Zablokuje se volání přerušení událostí „reciever timeout“ resetem bitu RTOIE.
5. Přepnutí USART1 do MUTE módu (klidový režim).
6. Po detekci klidového stavu sběrnice tzv.Idle time dojde automaticky k přepnutí zařízení zpět do aktivního módu sledování sběrnice viz obr. 11.6, 11.7.



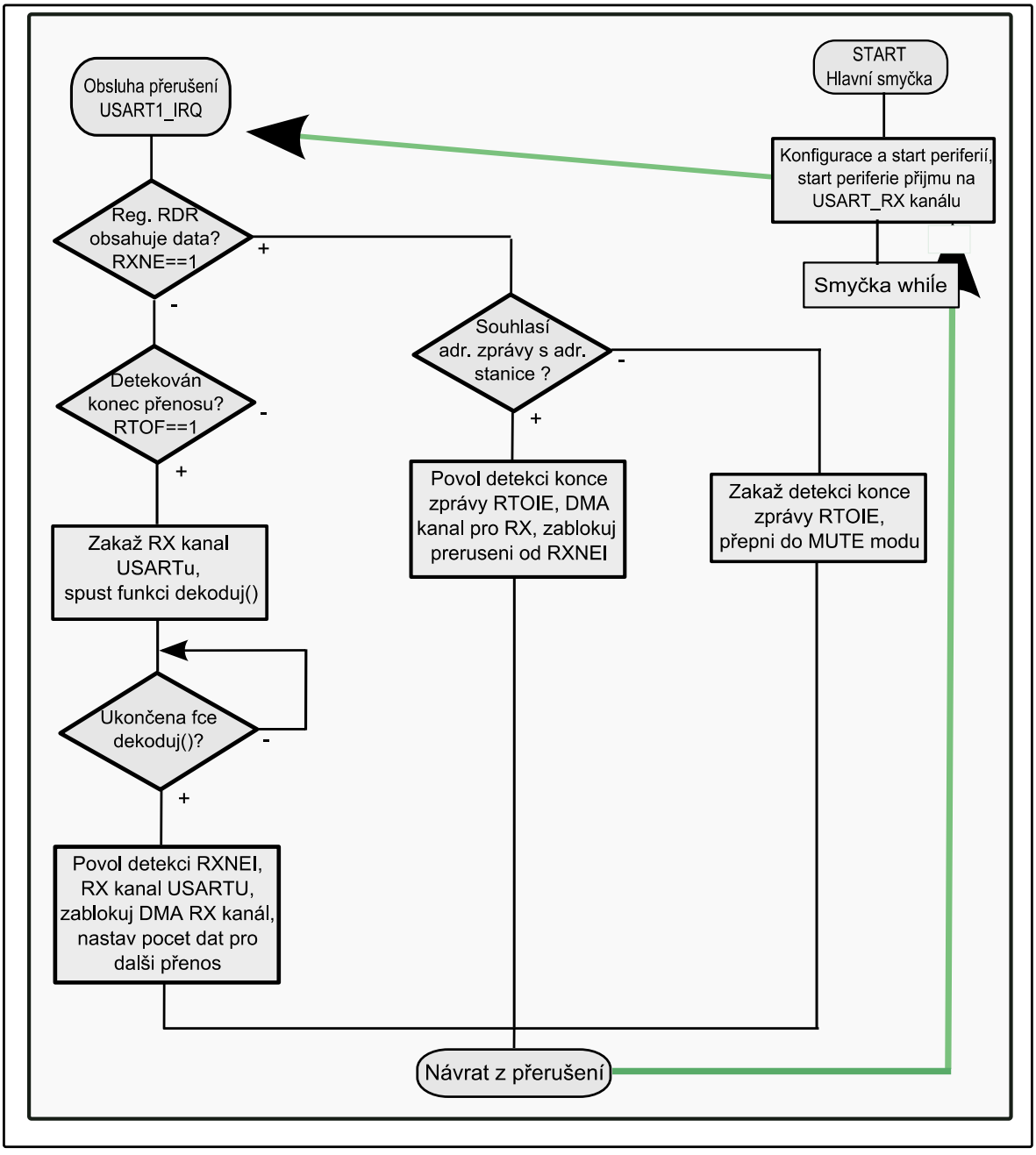
Obr. 11.6 Povel pro výpis na displej (žlutý průběh) v souvislosti se stavem bitu RWU (úroveň “1“ odpovídá MUTE mode), v případě přijetí příkazu stanicí pro kterou není určen.



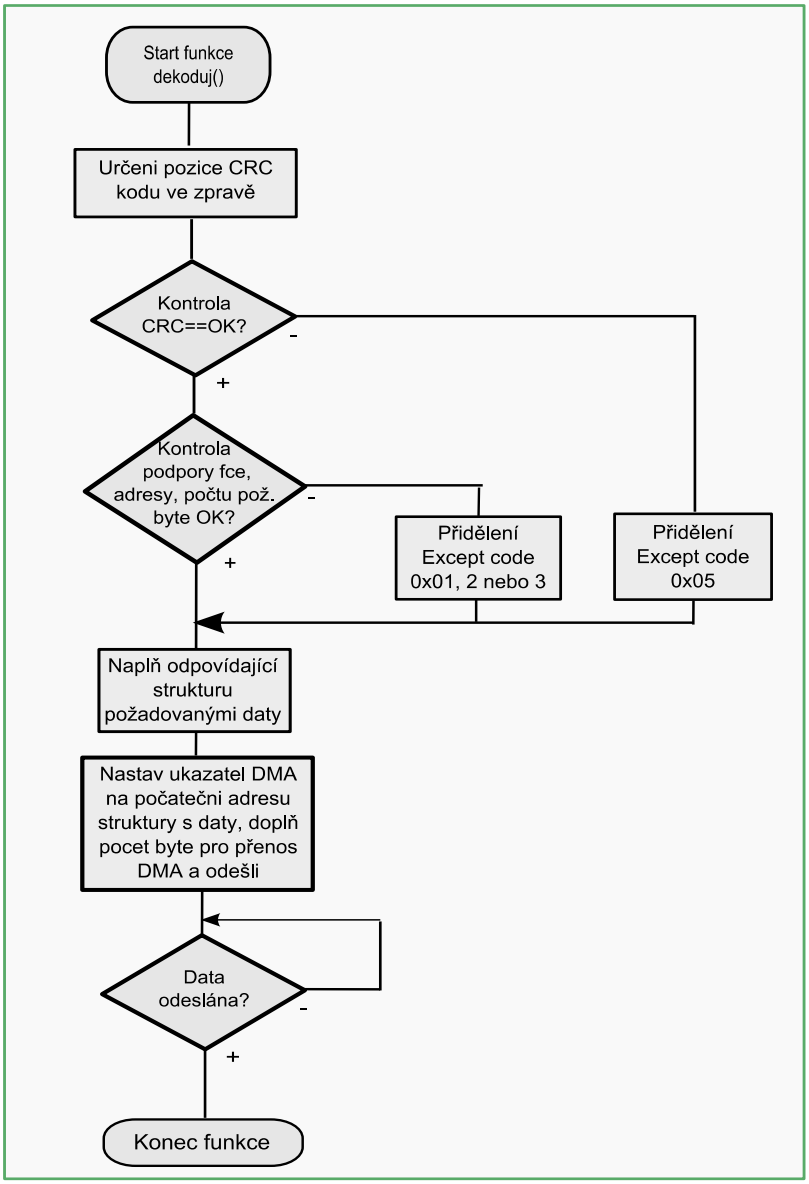
Obr. 11.7 Stavy bitů RWU (1=MUTE mód aktivní,0=stanice naslouchá). Zeleně průběh stanice 2, žlutě stanice 3).

11.4.3 Vývojový diagram jednotky slave

Program mikrokontroléru po spuštění nastaví požadované periferie včetně naslouchání kanálu RX periferie USART. V okamžiku detekce naplnění registru USART_RDR daty, začne zpracování zprávy v obsluze přerušení podle vývojového diagramu z obr. 11.8.



Obr. 11.8 Vývojový diagram zpracování requestu v jednotce slave.



Obr. 11.9 Vývojový diagram funkce pro zpracovaní dotazu a odeslání odpovědi..

Odeslání zprávy proběhne voláním funkce dekoduj(), viz obr. 11.9. Vyhodnocení přijaté zprávy probíhá dle specifikace protokolu MODBUS [22], nejprve se provádí vyhodnocení CRC kódu, pokud je vypočtený CRC kód přijaté zprávy nulový, zpráva je s vysokou pravděpodobností v pořádku. Následně je kontrolována adresa cílového zařízení, pokud i tato souhlasí, provede se kontrola obsahu zprávy, která se vyhodnocuje podle specifikace [22] požadované funkce. V případě, že nemůže slave sestavit odpověď na dotaz mastera z důvodu zjištění chyby v dotazu, odpoví kódem vyjímky viz příklad obr. 11.9. Nedojde-li ke zjištění vyjímky program začne sestavovat odpověď na dotaz. Při sestavování této odpovědi umístí MCU data do strukturní proměnné odpovídající požadované funkci, poté doplní do strukturní proměnné CRC kód zprávy. Pro odeslání zprávy je nastaven ukazatel periferie DMA na paměťové místo začátku struktury a dále je nastaven rozsah přenášených dat. Posledním krokem cyklu je uvolnění periferie DMA, tím dojde k postupnému přenosu dat do periferie USART, který již probíhá bez účasti procesoru mikrokontroléru

11.4.4 Adresování registrů v MCU

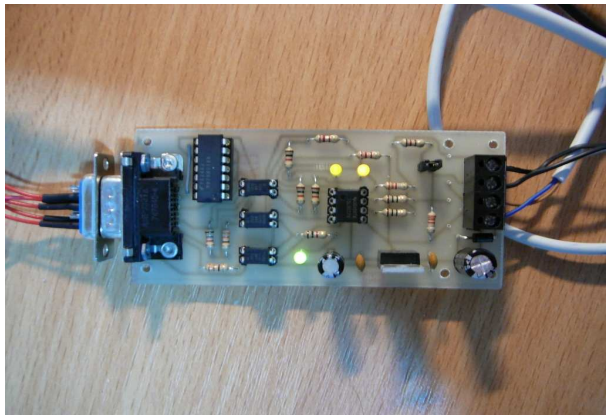
Pro použití protokolu Modbus bylo nutné vyřešit problém s velikostí adresy používané v dotazech jednotky Master. Modbus ve své specifikaci předepisuje 16bitovou velikost adresy. Naproti tomu mikrokontrolér používá 32 bitové adresy, není proto možné přímo v požadavcích uvádět skutečnou adresu v paměti mikrokontroléru. Z tohoto důvodu existuje v programu pole ukazatelů na 16bitové hodnoty registrů, které jsou mapovány, vždy na polovinu 32bitové adresy paměti MCU. Tabulku těchto ukazatelů je tedy pro oslovení správných registrů dopředu znát. Například pro sestavení požadavku na čtení stavu spodních 16bitů registru GPIOA-IDR, pomocí fce 0x03 (čtení obsahu registrů), do položky adresa zadáme adresu „0“, ukazatel dat bude nasměrován na spodních 16bitů adresy registru GPIOA-IDR.

11.5 Fyzická vrstva – linka 485

Jako fyzická vrstva pro přenos dat byla zvolena linka RS485. Propojení mezi jednotlivými stanicemi se bylo provedeno pomocí krouceného páru vodičů. Z důvodu omezení odrazů na vedení byly na konci vedení zapojeny zakončovací rezistory o hodnotě 120 ohm. Linka by měla mít pouze dva konce, na vodiče mezi těmito konci se připojují stanice s kterými je možné komunikovat. Není možné vytvářet hvězdicové, kruhové apod. topologie sítě.

Linka RS485 je řešena jako diferenciální, minimální rozdíl napětí páru vodičů je 200 mV na vstupu přijímače a 1,5 V na výstupu, zdroj [24]. Linka používá pro rozlišení páru vodičů označení A a B. V klidovém stavu je na vodiči A napětí vyšší než na vodiči B. Toto napětí je dodáno pomocí napěťového děliče viz schéma převodníku RS485, příloha 1.

Mikrokontrolér STM32F051 má pro komunikaci s ostatními zařízeními k dispozici periferie USART, I2C, SPI. Pro uvažovanou aplikaci sledování stavu strojů byla vybrána pro použití periferie USART. Napěťové úrovně z této periferie bylo nutné převést na napěťové úrovně odpovídající diferenciální lince RS485. Pro tento účel byl zkonstruován převodník napěťových úrovní viz obr. 11.5 s obvodem 75176, převodník byl doplněn o galvanické oddělení linky a mikrokontroléru.



Obr. 11.5 Převodník USART-RS485.

11.6 Příklady praktického využití multiprocesorové komunikace

Pro ověření možností multiprocesorové komunikace byly zkonstruovány dvě aplikace. První aplikace je tvořena třemi procesory STM32F051 a slouží k zobrazení dat získaných ze dvou enkodérů. Druhá aplikace je zaměřena na možnost využití monitorování stavu strojů ve výrobním provozu lisovny plastů.

11.6.1 Aplikace sběr dat s enkodérů

Aplikace může být využita pro sledování dvouosého polohovací systému. V systému pracují tři hlavní celky:

- Enkodér pro osu X s mikrokontrolérem STM32F051
- Enkodér pro osu Y s mikrokontrolérem STM32F051
- Display pro zobrazení polohy s mikrokontrolérem STM32F051

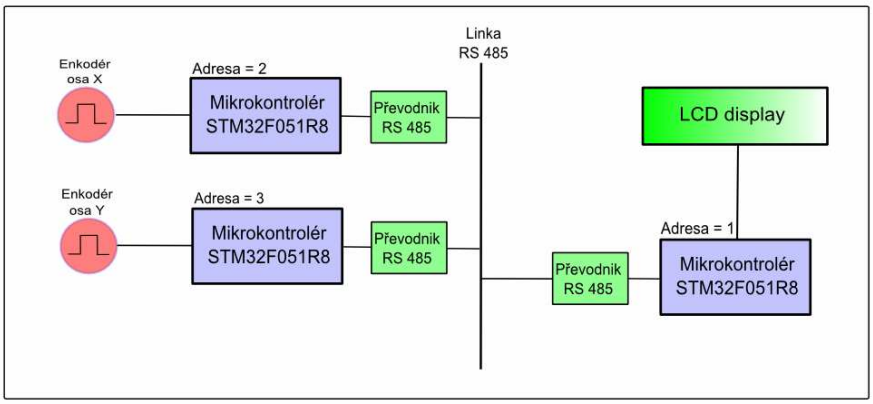
Data z enkodérů jsou zpracovávány mikrokontrolérem. Mikrokontrolér dokáže vyslat data o aktuální poloze enkodéru pokud je o to požádán jednotkou typu master. Jednotka master je tvořena displejem obr. 11.7, na kterém jsou zobrazovány data z enkodérů pro osy X a Y.

Obr. 11.7 Display jednotky master se zobrazeným stavem osy Y.

Zobrazení chyb

Kromě údaje o stavu enkodérů osy X, Y, může display zobrazit následující chybová hlášení:

- TIMEOUT ERR - od stanice nedorazila v daném čase odpověď.
- BAD CRC - zpráva má špatný CRC kód.
- BAD QUANT. - špatné požadované množství byte ke čtení.
- BAD ADRRESS - špatná adresa registru pro čtení.
- UNSUPP. FUN - funkce není podporována.



Obr. 11.8 Přehledové schéma propojení modulů.

11.6.2 Sledování stavu strojů

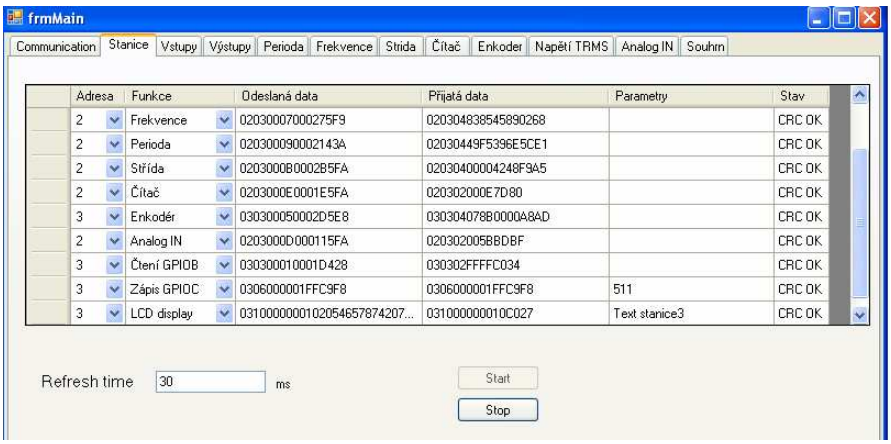
Druhá aplikace je zaměřena na možnosti sledování stavu strojů v provozu lisovny plastů. V první verzi je požadováno pouze sledováni stavu diskrétních výstupů stroje s informací o chodu stroje, informací o dostatečném stavu materiálů apod.. Informace by měla být průběžně zobrazována na obrazovce PC.

Koncept systému, prostředky pro realizaci

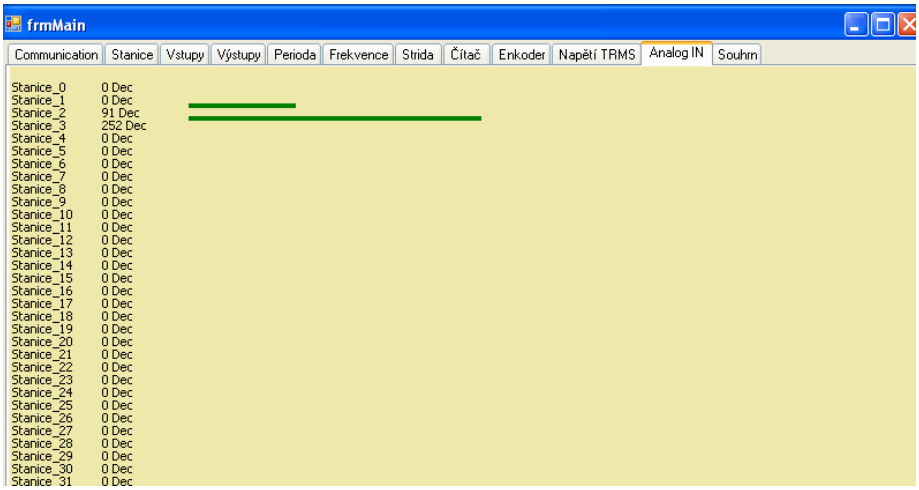
Koncepce systému počítá s umístněním modulu obsahujícího mikrokontrolér typu STM32F051 s převodníkem na linku 485 doplněným galvanickým oddělením do každého stroje. Podél strojů bude umístněno vedení linky RS485 na kterou budou moduly připojeny. Na jednom konci vedení bude umístněna jednotka Master tvořená počítačem. V počítači bude spuštěna aplikace, která bude postupně oslovovat stanice na zmíněné síti strojů, data upraví pro vhodné zobrazení o odešle na obrazovky umístněné ve výrobních prostorách.

Řešení systému

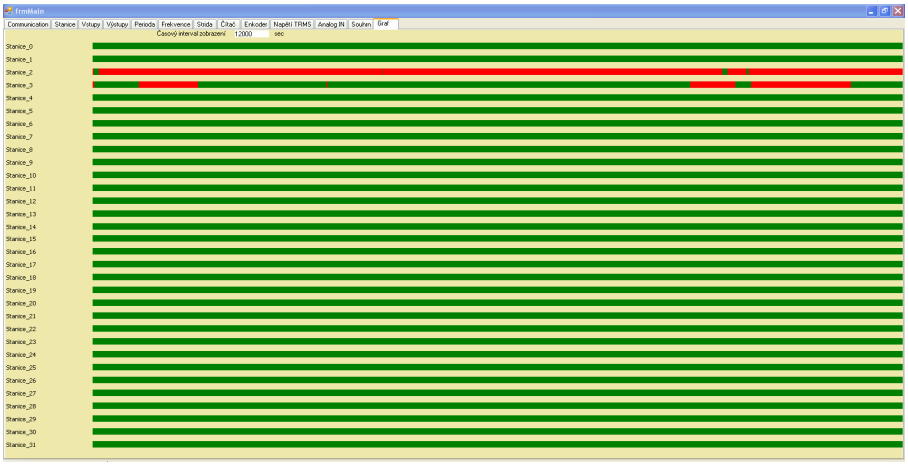
V prvním kroku byl napsán jednoduchý aplikační testovací software v programovacím prostředí Microsoft Visual Studio 2010. Tento software v PC realizuje roli mastera a umožňuje oslovovat jednotlivě stanice systému. Seznam dotazů je možné předepsat v tabulce viz obr. 11.9. Software není jednostranně zaměřen na konečnou aplikaci, umožňuje vytvářet dotazy na stav ADC převodníku obr. 11.10, stavu výstupů, vstupů, stavu hodnoty čítače apod. Pomocí software lze také řídit stav výstupů, zapisovat informace na LCD display. Pro sledovaní stavu periferií byly vytvořeny záložky s přehledovými obrazovkami.



Obr. 11.9 Seznam stanic, požadovaných dat, kompletní výpis zasílaných dat dotazu a přijaté odpovědi.



Obr. 11.10 Zobrazení aktuálních stavů analogových vstupů připojených jednotek.



Obr. 11.12 Průběžný graf vývoje stavu vstupů připojených jednotek.

12. Závěr

Hlavním úkolem této diplomové práce byla analýza, návrh a realizace přístrojových funkcí s využitím zabudovaných periferií mikrokontroléru STM32F05x, s jádrem ARM Cortex-M0. Nedílným úkolem vycházejícího ze zadáním diplomové práce, bylo vytvoření příslušných vzorových programů. Další důležitou částí a cílem práce bylo odzkoušení a změření základních parametrů realizovaných přístrojových funkcí. Závěrečným úkolem, potom porovnání těchto získaných parametrů, s parametry speciálních obvodů.

V souladu se zadáním diplomové práce bylo nutné získání základních znalostí práce s mikrokontrolérem STM32F05x. Při této příležitosti byla vytvořena na žádost vedoucího práce dokumentace a sada jednoduchých programů v programovacích jazycích assembler a C. Pro budoucí studenty jsou připraveny vzorové programy na použití vstupně/výstupních periferií, periferie ADC (analogově-digitálního převodníku), DAC (digitálně-analogového převodníku), USART (universálního sériového rozhranní), DMA (přímého přístupu k paměti mikrokontroléru). Tyto programy, zaměřené na použití základních periferií mikrokontroléru , jsou umístněny na přiloženém CD ve složce „STM32F05x_zakladni_priklady“.

Dle zadání diplomové práce byly vytvořeny s použitím mikrokontroléru STM32F051R8 tyto přístrojové funkce:

- Číslicový voltmetr
- Měřič frekvence a periody přímou metodou
- Přístroj pro měření délky periody a střídy signálu
- Impulsní generátor s využitím PWM
- Přístroj pro měření polohy pomocí enkodéru
- Funkční generátor s využitím DDS
- Přístroj pro záznam signálu datalogger
- Monitorovací systém s využitím multiprocesorové komunikace

V další části, budou v jednotlivých odstavcích stručně shrnuty hlavní dosažené výsledky řešení konkrétních realizovaných přístrojových funkcí.

Číslicový voltmetr

Přístrojová funkce voltmetru byla realizována pomocí AD převodníku ve 12bitovém režimu. Při tomto maximálním rozlišení je kvantovaci krok převodníku 0,723 mV. Vzorkovací frekvence voltmetru byla nastavena na $200\ \mu s$. Tato doba se skládá z doby nutné pro převod ADC převodníku $2{,}166\ \mu s$, doby potřebné pro výpočet a aktualizaci měřených hodnot s dobou trvání cca $160\ \mu s$ a z části doby $40\ \mu s$ potřebné pro zobrazení dat na LCD displeji. Zobrazení dat na displeji probíhá s periodou 0,5 s. Výsledná hodnota je vypočtena z množství 1000 vzorků měřeného průběhu. Voltmetrem jsou měřeny a vypočteny hodnoty max. a min. napětí, střední elektrolytická, střední aritmetická a efektivní TRMS hodnota napětí.

Šířka pásma voltmetru byla stanovena s ohledem na vzorkovací periodu jako 10 Hz až 4,5kHz.

Chyba měřené hodnoty voltmetru vzhledem ke standardu tvořeným generátorem DS-X 2012A [12] pro frekvenci sinusového signálu 50 Hz, byla naměřena v rozmezí -1,5 mV až +2,5 mV. Typické celkové chyby ADC převodníku mohou být podle [2] v tomto případě $\pm 3{,}3$ *LSB* . Nejvyšší chyba měření byla zjištěna jako 0,085 % , tato hodnota odpovídá +3,4 LSB.

Měřič frekvence a periody přímou metodou

Bylo realizováno řešení měření frekvence přímou metodou s použitím 32bitového čítače mikrokontroléru STM32F05x. Tímto přístrojem je možné měřit, dle Nyquistova kritéria, maximální frekvenci vstupního měřeného signálu 24 MHz, za podmínky vzorkovací frekvence příslušné jednotky časovače 48 MHz. Rozlišovací schopnost přístroje 1 Hz je dána dobou otevření hradla 1s, tento čas je rovněž periodou aktualizace hodnoty měřeného údaje.

Ověření funkce bylo provedeno generováním obdélníkového signálu v rozmezí 1 Hz až 10 MHz, byl zjištěn maximální rozdíl měřené a nastavené hodnoty +0,002 %.

Přístroj pro měření délky periody a střídy signálu

Pro nepřímé měření frekvence, měření délky periody, délky pulsu a střídy signálu byl realizován přístroj pracující na principu měření doby intervalů mezi okamžiky změn vstupního měřeného signálu. Časové rozlišení přístroje při hodinovém kmitočtu časovače 48 MHz, odpovídalo předpokládané hodnotě 20,83 ns.

U tohoto přístroje byly zjištěny poměrně vysoké chyby, způsobené pravděpodobně chybou kmitočtu použitého krystalu. Z tohoto důvodu byla v programu provedena korekce počtu načtených pulsů, poté chyba měřené frekvence nepřekročila hodnotu +0,0002%, při max. frekvenci signálu do 1000 Hz. Kvantovací chyba způsobí na tomto kmitočtu nejistotu měření frekvence v hodnotě 20,8 mHz.

Přístroj je určený především pro nízké frekvence, není proto omezena maximální frekvence. S vyšší frekvencí je třeba počítat s vyšší procentuelní chybou měření, z důvodu načtení menšího počtu pulsů.

Impulsní generátor s využitím PWM

Pro generování impulsního signálu, byla využita periferie časovače TIM2 s podporou tvorby PWM signálu. Byla dosažena minimální šířka pulsu 20,83 ns, odpovídající nastavenému hodinovému kmitočtu periferie časovačů 48 MHz. Šířku pulsu je možné měnit s krokem 20,83 ns. Náběžnou a sestupnou hranu výstupního signálu je možné očekávat v hodnotě 10 ns. Maximální opakovací frekvence pulsů je 24 MHz, minimální 18 mHz.

Přístroj pro měření polohy pomocí enkodéru

Periferie časovačů umožňují, díky podpoře výrobce, přímé dekódování signálu z enkodérů. Vstupy jsou vzorkovány 48MHz, maximální vstupní frekvence signálů je teoreticky 24 MHz. Mikrokontrolér obsahuje 32bitový čítač/časovač, který je díky vysokému rozsahu možných hodnot vhodný pro použití s enkodérem.

Funkční generátor s využitím DDS

Byl zkonstruován přístroj pro generování signálů principem přímé číslicové syntézy DDS. Přístroj umožňuje generování sinusových, pilových a obdélníkových signálů. Nastavení frekvence signálu je možné v rozsahu od 1 Hz do 500 kHz s krokem 1 Hz. Při nastavení maximální frekvence 500 kHz je výsledný signál skládán ze 4 vzorků. Generátor by bylo vhodné pro vyhlazení skoků signálu, způsobených při vyšších hodnotách požadované frekvence signálu malým množstvím vzorků na periodu signálu, doplnit filtrem typu dolní-propust.

Přístroj pro záznam signálu datalogger

Byl zkonstruován a odzkoušen přístroj umožňující sběr dat prostřednictvím periferie ADC převodníku. Nejmenší nastavitelná hodnota odběru vzorků přístroje je 2 uS, s krokem možné změny 1 us. Minimální doba odběru vzorku je dána součtem doby odběru vzorku ADC periferií, cca 1 us a periodou časovače 1 us. Maximální vstupní frekvence signálu pro přístroj datalogger je dle Nyquitsova kritéria 250 kHz. Použitím 32bitového časovače TIM2, bylo dosaženo maximálního periody vzorkovacího intervalu 49 dnů.

Maximální vzorkovací frekvence ADC převodníku může při nastaveném 8bitovém rozlišení a při hodinové frekvenci periferie ADC 14 MHz, dosahovat až 1,27 Mhz, to odpovídá dle Nyquitsova kritéria max. možné měřené frekvenci 635 kHz.

Rozlišení přístroje dataloggeru bylo nastaveno na 8 bitů. Důvodem je možnost uložení vyššího množství vzorků do paměti mikrokontroléru. Načtené průběhy je možné přenést do PC a pomocí programu NRM_Display [21] zobrazit na obrazovce.

Monitorovací systém s využitím multiprocesorové komunikace

Jako poslední úkol byly realizovány dvě aplikace využívající spolupráce více mikrokontrolérů.

První realizovaná aplikace využívá komunikace tří mikrokontrolérů, z nichž dva mikrokontroléry zpracovávají signál z enkodérů, třetí mikrokontrolér zobrazuje získaná data na LCD displeji. Z důvodu kompatibility s druhou aplikací byla aplikační vrstva řešena protokolem Modbus.

Druhá aplikace, skládající se z několika jednotek mikrokontrolérů a jednoho PC v roli mastera, slouží pro testování možnosti monitorování stavu strojů ve výrobním procesu. Vzhledem k rozlehlému rozmístnění strojů a možnostem obvodu STM32F051 byla fyzická vrstva řešena pomocí standardu RS485. Z tohoto důvodu byly navrženy a postaveny příslušné převodníky napěťových úrovní periferie mikrokontroléru USART na úrovně linky RS485. Aplikační vrstva přenosového protokolu je řešena protokolem Modbus. Vývoj stavu stroje za zvolený časový interval je zobrazován na obrazovce počítače. Testovací software pro zobrazení byl napsán v jazyce C#. Tento software umožňuje navíc sledování a ovládání dalších vybraných registrů mikrokontroléru.

Návrh a realizace přístrojových funkcí pomocí periferií obvodu STM32F051 ukázal možnosti a meze tohoto velmi levného a zároveň dostatečně výkonného mikrokontroléru. Největší výhodou řešení přístrojových funkcí mikrokontrolérem, je možnost okamžité změny funkce pouhým nahráním programu. Při realizaci přístrojových funkcí byl kladen důraz na

možnost praktického použití, proto byly přístroje vybaveny tlačítky a LCD displejem pro zobrazení měřených a nastavovaných hodnot.

Z důvodu provedení všech bodů dle zadání diplomové práce předpokládám, že jsem zadání této diplomové práce splnil.

13. Seznam literatury

[1] RM0091-Referenční manuál, 2012, STMicroelectronic, Doc ID 018940 Rev 2
http://www.st.com/web/en/resource/technical/document/reference_manual/DM00031936.pdf

[2] Datasheet reference-STM32F051x4 STM32F051x6 STM32F051x8, 2012, STMicroelectronic, Doc ID 022265 Rev 3
http://www.st.com/web/en/resource/technical/document/datasheet/DM00039193.pdf

[3] PM0215-Programming manual, 2012, STMicroelectronic, Doc ID 022979 Rev 1
http://www.st.com/web/en/resource/technical/document/programming_manual/DM00051352.pdf

[4] AN4067- Application note, How to calibrate STM32F0xx internal RC oscillators,2012, STMicroelectronic, Doc ID 022913 Rev 1
http://www.st.com/web/en/resource/technical/document/application_note/DM00050140.pdf

[5] Yiu J.: The define Guide to the ARM Cortex-M0, Elsevier, 2011, ISBN: 978-0-12-385477-3

[6] ARM®Compiler toolchainVersion 4.1Assembler Reference, 2010-2011, ARM, DUI 0489C (ID080411)
http://infocenter.arm.com/help/topic/com.arm.doc.dui0489c/DUI0489C_arm_assembler_reference.pdf

[7] Datasheet MAX3222/MAX3232/MAX3237/MAX3241, 2007, Maxim Integrated, 19-0273; Rev 7;
http://www.ti.com/lit/ds/symlink/max232.pdf

[8] Josef Vedral, Jan Fischer: *Elektronické obvody pro měřící techniku*, Praha 2004, ISBN 80-01-02966-2

[9] Power meter Tutorials-Crest Factor, Yokogawa Meters & Instruments Corporation, 2005-2013
http://www.yokogawa.com/ymi/tutorial/tm-tutorial_wt_08.htm

[10] Datasheet-AD637, Rev. K, 2007-2011, Analog devices, D00788-0-2/11(K)
http://www.analog.com/static/imported-files/data_sheets/AD637.pdf

[11] Datasheet-LTC1966, 2001, Linear Technology Corporation, 1966fb
http://cds.linear.com/docs/en/datasheet/1966fb.pdf

[12] InfiniiVision 2000 X-Series Oscilloscopes - Data sheet, 2012-2013, Agilent Technologies, Inc.
http://cp.literature.agilent.com/litweb/pdf/5990-6618EN.pdf

[13] Přednáška předmětu X38EMA – Elektrická měření
http://measure.feld.cvut.cz/cs/system/files/files/cs/vyuka/predmety/x38ema/EMA-Predn_5tisk.pdf

[14] Boháček, J.: Metrologie. Skripta ČVUT, Praha 2011, ISBN 978-80-01-04839-9

[15] Webové stránky společnosti KRYSTALY, a.s.
http://www.krystaly.cz/cs/Produkty/Krystaly/UM-4_UM-5

[16] Přednášky předmětu A4B38NVS – Návrh vestavných systémů
http://measure.feld.cvut.cz/system/files/files/cs/vyuka/predmety/A4B38NVS/NVS_2012_Pr10_Citace_2.pdf

[17] Datasheet LTC6992, 2010, Linear Technology Corporation, 69931234fb
http://cds.linear.com/docs/en/datasheet/69931234fb.pdf

[18] Znakové LCD displeje, 2007, DOVEDA BOYS © 1998 – 2012,
http://doveda.byl.cz/lcd/

[19] Datasheet AD5930, Rev. B, 2005-2012, Analog devices, D05333-0-7/12(B)
http://www.analog.com/static/imported-files/data_sheets/AD5930.pdf

[20] Datasheet ADuC7128/ADuC7129, Rev. 0, 2007, Analog devices, D06020-0-4/07(0)
http://www.analog.com/static/imported-files/data_sheets/ADUC7128_7129.pdf

[21] NRM_Display Version 2.3 (build v2.320121115), Ing. Ondřej Pribula, K13138, CTU Prague

[22] MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b, 2006,
Modbus Organization
http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf

[23] Lammert B.: file lib_crc.c-contains the private and public functions used for the calculation of CRC-16, CRC-CCITT and CRC-32 cyclic redundancy values., V 1.16, 1999-2008
http://www.lammertbies.nl/comm/software/index.html

[24] Kugelstadt D,: Application Report – The RS-485 Design Guide, May 2008,
Texas Instruments, SLLA272B
http://www.ti.com/lit/an/slla272b/slla272b.pdf

[25] Webové stránky společnosti Moravské přístroje a.s.
www.mii.cz/

[26] Krajča K.; Rozman J.: Generátor impulsů pro aplikaci v magnetoterapii., časopis Elektrorevue
http://www.elektrorevue.cz/clanky/04054/index.html

[27] Burghard M.: C pro mikrokontroléry, Praha 2003, ISBN 80-7300-077-6

14. Seznam příloh

Příloha 1

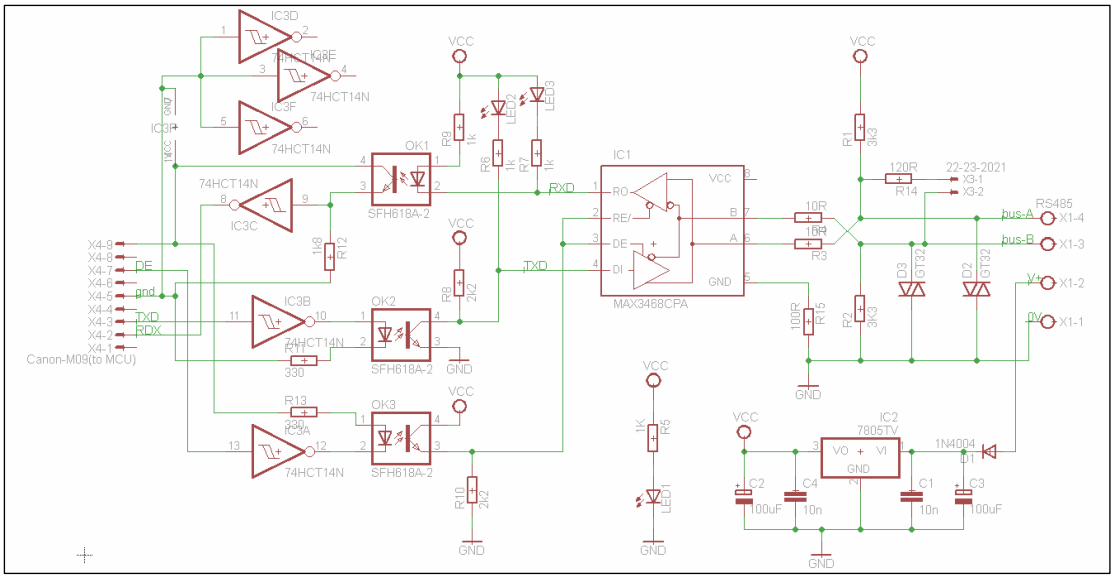
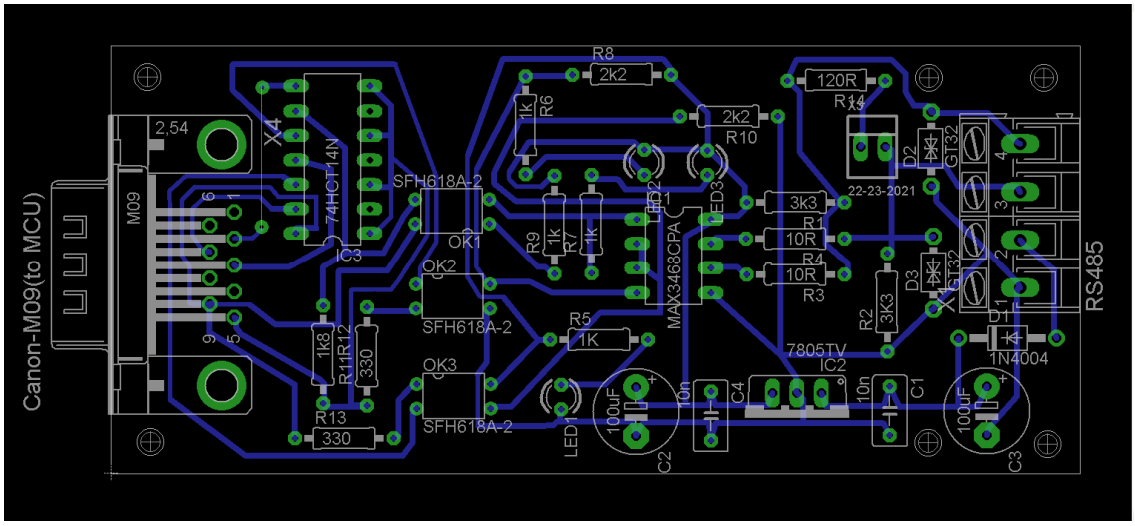


Schéma převodníku USART-RS485.



Deska s plošnými spoji převodníku USART- RS485

Příloha 2

Tabulka naměřených a vypočtených efektivních hodnot napětí, získaných měřením funkce sinus, f=50Hz z generátoru osciloskopu DSO-X 2012A. Konstanta „REF_NAPETI“ v programu voltmetr nastavena na hodnotu 2.97.

Generátor Up-p [V]	Voltmetr ef	Odchylky od linearity měřené Uef [mV]	Výpočet Uef z napětí generátoru [mV]	Rozdíl Uef (vypočtené - měřené) [mV]
3	1060	0,5712	1060,66	0,66
2,9	1024	-0,1398	1025,30	1,30
2,8	990	1,1492	989,95	-0,05
2,7	955	1,4382	954,59	-0,41
2,6	920	1,7272	919,24	-0,76
2,5	882	-0,9838	883,88	1,88
2,4	848	0,3052	848,53	0,53
2,3	813	0,5942	813,17	0,17
2,2	778	0,8832	777,82	-0,18
2,1	740	-1,8278	742,46	2,46
2	706	-0,5388	707,11	1,11
1,9	671	-0,2498	671,75	0,75
1,8	637	1,0392	636,40	-0,60
1,7	599	-1,6718	601,04	2,04
1,6	564	-1,3828	565,69	1,69
1,5	530	-0,0938	530,33	0,33
1,4	495	0,1952	494,97	-0,03
1,3	458	-1,5158	459,62	1,62
1,2	424	-0,2268	424,26	0,26
1,1	388	-0,9378	388,91	0,91
1	353	-0,6488	353,55	0,55
0,9	318	-0,3598	318,20	0,20
0,8	283	-0,0708	282,84	-0,16
0,7	247	-0,7818	247,49	0,49
0,6	212	-0,4928	212,13	0,13
0,5	176	-1,2038	176,78	0,78
0,4	141	-0,9148	141,42	0,42
0,3	106	-0,6258	106,07	0,07
0,2	72	0,6632	70,71	-1,29
0,1	37	0,9522	35,36	-1,64

Příloha 3

Nepřímé měření frekvence-tabulka naměřených dat před provedením korekce pomocí členu

y=x-(x/13400-2).

y-počet načtených hodinových pulsů po provedené korekci
x-počet načtených hodinových pulsů časovače TIM2

Frekvence nastavená [Hz]	Frekvence měřená [Hz]	Rozdíl frekvencí naměřená-skutečná [Hz]	Rozdíl frekvencí naměřená-skutečná [%]	Načtených pulsu mikrokontrolérem	Odchylka od ideálního počtu pulsů
0,1	0,09999252	-7,48E-06	-0,0075	4800035853	35853
1	0,9999257	-7,43E-05	-0,0074	48003540	3540
10	9,9991	-0,0009	-0,0090	4800355	355
50	49,996	-0,004	-0,0080	960069	69
100	99,9922	-0,0078	-0,0078	480034	34
500	499,9687	-0,0313	-0,0063	96005	5
800	799,973	-0,027	-0,0034	60003	3
1000	999,9583	-0,0417	-0,0042	48002	2
2000	2000	0	0,0000	24000	0
5000	5000,5208	0,5208	0,0104	9599	-1
8000	8001,3336	1,3336	0,0167	6118	-2
10000	10002,84	2,84	0,0284	4798	-2
30000	30037,5456	37,5456	0,1252	1598	-2
50000	50104,384	104,384	0,2088	958	-2
80000	80267,552	267,552	0,3344	598	-2
100000	100418,406	418,4064	0,4184	478	-2
150000	150943,392	943,392	0,6289	318	-2
200000	201680,672	1680,672	0,8403	238	-2
250000	252631,584	2631,584	1,0526	190	-2
300000	303797,472	3797,472	1,2658	158	-2
500000	510638,304	10638,304	2,1277	94	-2
1000000				46	-2
2000000				22	-2
4000000				10	-2
8000000				4	-2

Příloha 4

Popis zapojení LCD displeje k vývojovému modulu STM32F0-Discovery

Displej je zapojen a provozován ve 4 bitovém módu. Datové vodiče LCD displeje jsou připojeny na následující piny portu GPIOB:

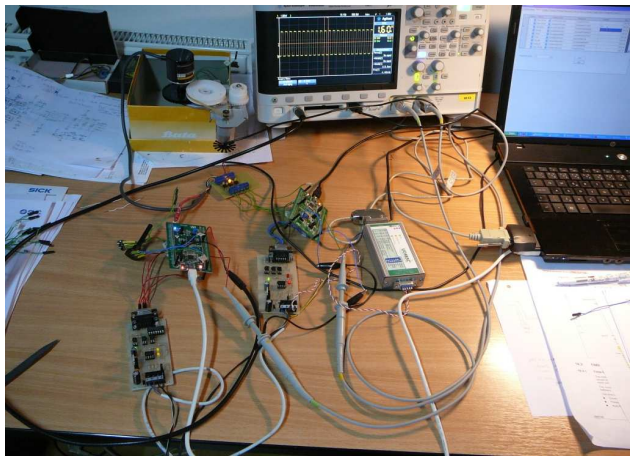
PB0 - LCD display DATA5
PB1 - LCD display DATA6
PB2 - LCD display DATA7
PB3 - LCD display DATA8

Řídící signály RS (výběr dat nebo instrukcí), RW (čtení nebo zápis), E (enable) jsou připojeny následujícím způsobem .

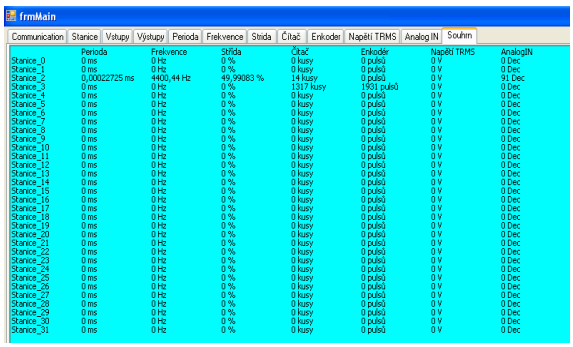
PB4 - LCD display RS (command/data)
PB5 - LCD display RW (read/write)
PB6 - LCD display E (enable)

Příloha 5

Obrázky z testování multiprocesorové aplikace



Propojení dvou modulů Discovery do sítě tvořené linkou RS485 .



Vlevo okno programu s aktuálním stavem vstupů jednotek připojených v síti, vpravo okno se zobrazením vybraných hodnot měřených mikrokontrolérem.

Příloha 6

CD přiložené k diplomové práci obsahuje:

- Diplomovou práci v pdf formátu
- Diplomovou práci v doc formátu (Microsoft Word)
- Zdrojové kódy příkladů
- Datasheety obvodů uvedených v seznamu literatury
- Zdrojové kódy přístrojových funkcí