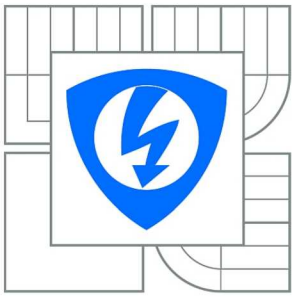
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF RADIO ELECTRONICS

DIGITÁLNÍ OSCILOSKOP S AVR

DIGITAL OSCILLOSCOPE WITH AVR

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

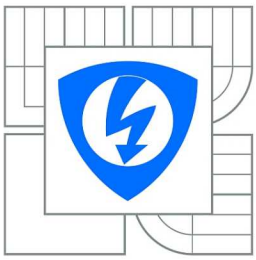
AUTOR PRÁCE — MARTIN STEJSKAL

AUTHOR

VEDOUCÍ PRÁCE — Ing. MARTIN FRIEDL

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav radioelektroniky

Bakalářská práce

bakalářský studijní obor
Elektronika a sdělovací technika

Student:	Martin Stejskal	***ID:***	125311
Ročník:	3	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Digitální osciloskop s AVR

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s mikrokontroléry firmy ATMEL. Popište principy digitálních osciloskopů a navrhněte vlastní uspořádání jednoduchého digitálního osciloskopu s mikroprocesorem AVR. Realizujte jednoduchý digitální osciloskop s mikroprocesorem AVR a proveďte na něm základní měření ověřující jeho parametry. Výsledné hodnoty srovnejte s komerčními přístroji.

DOPORUČENÁ LITERATURA:

[1] Mikroprocesorová technika [online]. 2009 [cit. 2010-05-20]. BMPT. Dostupné z WWW: <http://www.urel.feec.vutbr.cz/BMPT/index.php?strana=5&lang=CS>.

[2] MATOUŠEK, D. Práce s mikrokontroléry ATMEL AVR - ATmega16. Praha : BEN - technická literatura, 2006. 320 s. ISBN 80-7300-174-8.

Termín zadání:	6.2.2012	***Termín odevzdání:***	25.5.2012

Vedoucí práce: Ing. Martin Friedl
Konzultanti bakalářské práce:

prof. Dr. Ing. Zbyněk Raida
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Cílem práce bylo seznámit se s mikrokontroléry od firmy ATMEL a prostudovat na jakých principech fungují analogové a digitální osciloskopy. Na základě zjištěných poznatků byl blokově navržen digitální osciloskop s mikrokontrolérem ATMEL řady AVR. V dalších kapitolách byly pak řešeny vstupní obvody s využitím A/D převodníku, které převádí analogový signál na digitální a je dále zpracováván dalšími obvody. Velkou částí práce je pak samozřejmě software, který se mimo jiné zabývá datovými protokoly a zabezpečením dat

Klíčová slova

Osciloskop, AVR, mikrokontrolér, A/D převodník, LCD, zpracování dat

Abstract

The aim was got acquainted with microcontrolers from ATMEL family and study on wthat principes works analog and digital osciloscopes. On base of this informations, block design was made of digital osciloscope with microcontroler ATMEL from family AVR. The following chapters were look for a solution of input circuits with A/D convertor, witch transform analog signal to digital. Then, the digital signal is processed another circuits. Extended part of this work is software, witch must also controll data protocols and data security.

Keywords

Osciloscope, AVR, microcontroller, A/D convertor, LCD, data prosess

STEJSKAL, M. Digitální osciloskop s AVR. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 57 s. Vedoucí bakalářské práce: Ing. Martin Friedl.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Osciloskop s AVR jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené semestrální práce dále prohlašuji, že v souvislosti s vytvořením této semestrální práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ....

(podpis autora)

Poděkování

Děkuji vedoucímu práce Ing. Martinu Friedlovi za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne ....

(podpis autora)

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Osciloskop patří mezi základní vybavení každé laboratoře nebo bastlírny. V principu zobrazuje aktuální hodnotu napětí v závislosti na čase. Je především určen pro sledování periodických signálů. Je to velice užitečný přístroj například při ladění rezonančních obvodů, nastavování optimálního zesílení, měření frekvence, zobrazení AM modulace, analýzu jednotlivých uzlů v obvodu atd. Osciloskopy se dělí na analogové a digitální. Analogové byly vyvinuty jako první a používají se dodnes. Jejich výhodou je především jednoduchost, cena a dříve i větší frekvenční pásmo. Na druhou stranu, kvůli CRT obrazovce, bývají o dost rozměrnější. Digitální osciloskopy používají zobrazovací technologii LCD. Dále jsou schopné vypočítat frekvenci, střední hodnotu, efektivní hodnotu, špičkovou hodnotu zobrazeného signálu, rozpoznat a určit data na digitálních sběrnicích. Samozřejmě, že můžou mít i další funkce, nebo naopak nemusí mít ani výše jmenované. Výhodou digitálních osciloskopů jsou výše zmíněné funkce a dnes se stává standardem, že umí komunikovat s počítačem, takže i následné zpracování dat je pro uživatele jednodušší a efektivnější.

Celý dokument je rozdělen do 5 částí. V první části je teoretický rozbor a blokové schéma navrženého osciloskopu. Komunikační pravidla mezi jednotlivými bloky a zevrubné vysvětlení jednotlivých programů pro dané bloky jsou probrány v následující kapitole. Následuje softwarové řešení, kde jsou probrány základní myšlenky použitých algoritmů. Další kapitola se věnuje hardwarovému řešení samotného řízení a na závěr je shrnuta dosavadní práce na této bakalářské práci.

1 Teoretický rozbor a blokové schéma

Mikrokontroléry od firmy ATMEL patří mezi jedny z nejoblíbenějších. Je to dáno především počtem jejich periferií, výkonem, velikostí pamětí, konstrukčním řešením a v neposlední řadě i jejich cenou. K výhodám AVR oproti mikrokontrolérům (dále μC) PIC patří například: integrovaný A/D převodník (většinou 10bitový), 32 uživatelských registrů (PIC má pouze jeden akumulátor), PWM modulaci (stačí nastavit dané registry a program se nemusí PWM dále věnovat) a analogový komparátor. Dále je k dispozici zdarma simulátor a překladač do jazyka C. Vzhledem k převaze užitečných vlastností a ceně u AVR byly zvoleny právě μC řady AVR. Mikrokontroléry řady AVR se dělí na: UC3, XMEGA, mega, tiny, Baterry Management a Automotive [1].

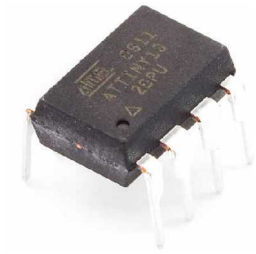
1.1 Porovnání mikrokontrolérů řady AVR

Srovnávací tabulka jednotlivých typů je v tab. 1. Z důvodu přehlednosti jsou vybrány jen některé parametry. Parametr „MIPS/MHz“ udává počet vykonaných instrukcí za jeden takt. Vzhledem k tomu, že obě jednotky mají předponu mega, lze prohlásit, že jde o účinnost vykonávání instrukcí vzhledem k taktu. Typ automotive je vlastně kombinací předchozích možností. Jejich hlavní parametr je jejich robustnost a jak již název napovídá, tak jsou určeny pro automobilový průmysl. Jejich cena je vyšší a dostupnost horší.

Tab. 1: Porovnání mikrokontrolérů řady AVR [1]

Typ	Paměť flash [kB]	Počet pinů [-]	Maximální takt [MHz]	MIPS/MHz [Hz^{-1}]
UC3	16-512	48-144	66	1,5
XMEGA	16-384	44-100	32	1,0
mega	4-256	28-100	20	1,0
tiny	0,5-8	6-32	20	1,0
Baterry Manager	8-40	28-48	8	1,0
Automotive	-	-	-	-

Mikrokontroléry typu *mega* jsou vyráběny v DIP pouzdru, což znamená, že jejich funkčnost lze velice jednoduše vyzkoušet na nepájivém poli. Naproti tomu μC *XMEGA* se vyrábí pouze v SMD pouzdře [2], takže jejich ladění a případná výměna se značně komplikuje. Na druhou stranu SMD pouzdra jsou menší a poslední dobou i levnější. Ukázky pouzder dokumentují obr. 1, obr. 2 a obr. 3. Vzhledem k parametrům, dostupnosti a ceně, byly zvoleny pro tuto práci AVR řady *mega*.



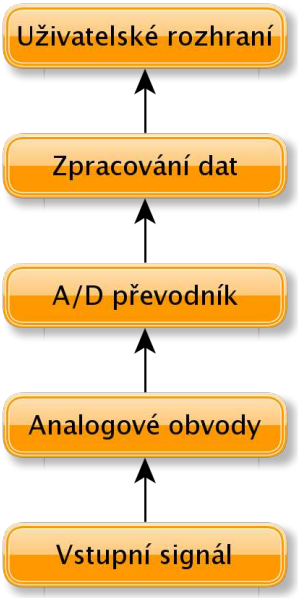
Obr. 1: Ukázka ATtiny13 v pouzdru DIL8 [3]

Obr. 2: ATmega32 v pouzdru DIL40 [4]

Obr. 3: ATmega128 v pouzdru TQFP64 [5]

1.2 Blokové složení digitálních osciloskopů

Každý digitální osciloskop musí obsahovat tyto bloky: analogové obvody (pro zesílení a pro impedanční oddělení vstupního signálu), A/D převodník, blok pro zpracování dat a uživatelské rozhraní, které bude prezentovat vstupní signál. Blokové schéma takového digitálního osciloskopu je zobrazeno na obr. 4. Šipky naznačují směr zpracovávaných dat.



Obr. 4: Blokové schéma obecného digitálního osciloskopu

Analogové obvody většinou obsahují ochranné prvky na vstupu, impedanční oddělení, děliče a předzesilovače. Tyto ochranné obvody ale zanášejí do celkového obvodu parazitní parametry. Takže jde o to, zvolit správný kompromis mezi robustností zařízení před nevhodným použitím a zároveň se musí vybrat součástky o takových parametrech, které nepříliš ovlivní kvalitu vstupních obvodů. Je vhodné, aby napěťový rozsah byl nastavován analogově, tj. nastavením vhodného zesílení, případně nastavením děliče. Pokud by byl použit A/D převodník s malým bitovým rozlišením, pak by pro malé signály vznikala velká chyba způsobená kvantovacím krokem. Při nastavování rozsahu analogovou cestou vyvstává řada problémů, zvlášť u mechanických spínacích prvků. Pokud by ale byl použit A/D převodník o velkém bitovém rozlišení, pak by doba převodu trvala delší dobu a to by u levnějších A/D převodníků mohl být problém.

Jak bylo uvedeno výše, A/D převodník je jeden ze základních prvků digitálního osciloskopu. Pokud má osciloskop fungovat spolehlivě, například na frekvenci 1 MHz, měla by být jeho vzorkovací frekvence alespoň 10x větší. Z toho plyne, že A/D převodník by měl být dostatečně rychlý. Zároveň ale požadujeme co nejvyšší rozlišení (pro co nejmenší horizontální chybu). Toto jsou ale dva protichůdné parametry, takže je nutné zvolit vhodný kompromis mezi rychlostí a rozlišením.

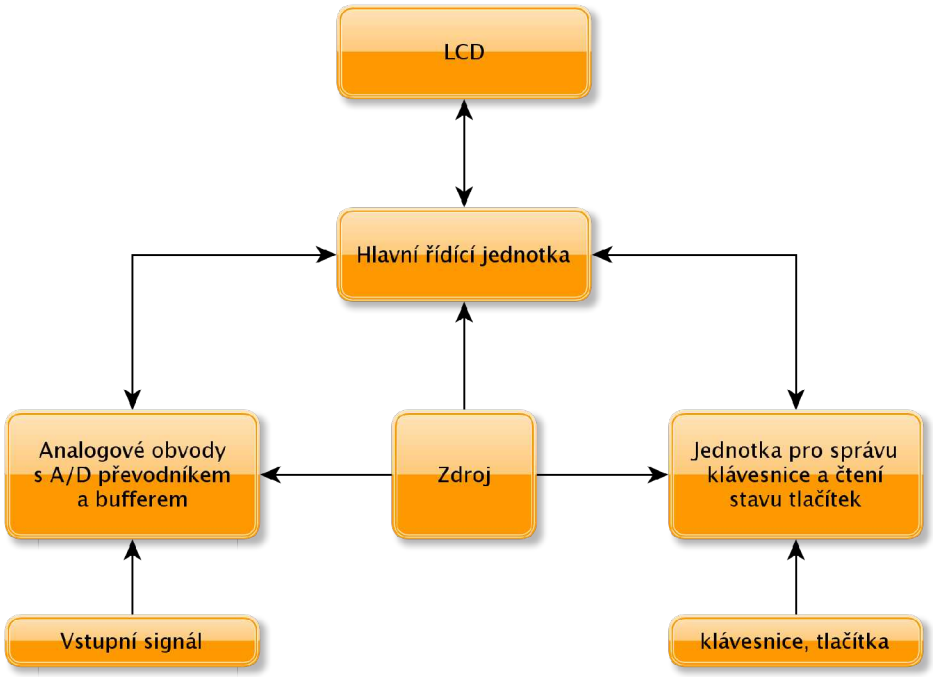
Blok pro zpracování dat zpracovává informace z A/D převodníku a má za úkol je předat uživatelskému rozhraní. Mimo jiné může vypočítávat frekvenci, dobu periody, maximální hodnotu napětí atd. Zároveň může posloužit jako softwarový trigger, který přináší oproti klasickému analogovému mnohem více možností. Nevýhoda softwarového triggeru je, že zatěžuje celý systém svými výpočty.

Bez uživatelského rozhraní by byly předchozí bloky k ničemu. Uživatelské rozhraní má za úkol komunikovat s uživatelem a poskytovat uživateli potřebné informace. Většinou je to displej s klávesnicí. V poslední době se vyskytují i tzv. USB osciloskopy, které samy o sobě nemají uživatelské rozhraní, ale využívají k tomu až počítač. S výhodou se veškeré výpočty přesouvají do

počítače, ale toto řešení je velice závislé na použitém operačním systému, takže zde není zaručena kompatibilita na každém operačním systému a nastávají problémy s různými verzemi dynamických knihoven.

1.3 Blokové schéma

Idea tohoto osciloskopu je rozdělit celý osciloskop do několik bloků (a to i hardwarově), které se dají vyměňovat nezávisle na ostatních blocích. Je to obdoba počítače. V počítači může být vyměněna paměť RAM za větší, CPU za rychlejší, grafická karta za výkonnější atd. Výhoda je zjevná: pokud bude potřeba určitou část vylepšit, pak stačí pouze vyměnit jeden blok a není potřeba konstruovat celý přístroj od začátku. Na druhou stranu se tím zvětší celková plocha potřebná pro DPS a je potřeba jednotlivé bloky propojit, což může vést k menší spolehlivosti při použití konektorů (krátké výpadky spojení → chyby). Rozdělení bloků je následující: hlavní řídící jednotka, analogové obvody s A/D převodníkem a bufferem, jednotka pro správu klávesnice a čtení stavu tlačítek a samozřejmě i zdroj. Blokové schéma je na na obr. 5.



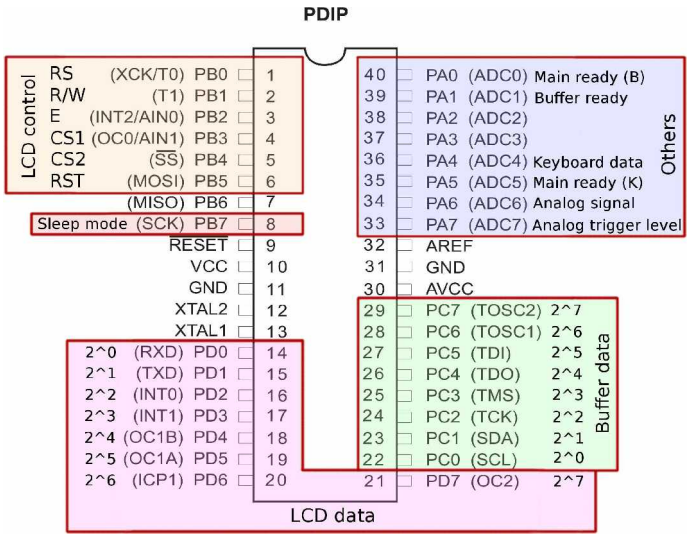
Obr. 5: Blokové schéma osciloskopu

1.4 Hlavní řídící jednotka

Tento blok je vlastně srdcem celého zařízení. Má za úkol sbírat data z ostatních bloků, odesílat potřebná data ostatním blokům (různé nastavení) a do toho obsluhovat i LCD displej. To znamená, že má na starost zpracování dat z bufferu, správnou prezentaci dat na LCD, reagovat na vstupní data z klávesnice, měnit časovou základnu, nastavovat počet vzorků a trigger, výběr vykreslení vstupních dat (body/vektor), zobrazovat jednoduchou nápovědu, provádět veškeré výpočty atd.

Pro tento účel nestačí „obyčejný“ µC. Musí být brány ohledy na počet vstupně/výstupních pinů (musí se obsluhovat LCD, buffer, µC s klávesnicí), velikost flash paměti (aby se program do paměti vešel), velikost SRAM (pro práci s proměnnými, vzorky), maximální taktovací frekvenci. Dobrým kompromisem těchto parametrů je ATmega32. Ta totiž obsahuje až 32 vstupně/výstupních pinů. Kapacitou 32 kB flash paměti dává k dispozici velký prostor pro program. Paměť SRAM o kapacitě 2 kB pro nenáročnou aplikaci jistě postačí. Maximální takt je 16 MHz a účinnost vykonávání instrukcí je 1 MIPS/MHz [6], čímž je zaručen i dobrý výpočetní výkon. Ani jeho cena není příliš vysoká. Pohybuje se okolo 140 Kč [7].

Rozvržení pinů pro ATmega32 je na obr. 6. Přenos dat pro LCD a buffer je paralelní, osmibitový. U LCD je tento přenos dat definovaný již od výrobce. Proto je výhodné využít port jako celek, který je také osmibitový. Usnadní se tím práce s daty. Při fyzickém rozložení pinů na ATmega32 je vhodné pro LCD zvolit porty B a D. Téměř celý LCD je pak připojen na „levou“ stranu µC. Zároveň na této straně máme zem a napájení +5 V, takže se o něco usnadní i návrh DPS. Přenos pro buffer byl zvolen jako paralelní z důvodu rychlosti přenosu dat a také jednoduchosti při přenosu dat. Port A je využit pro kontrolu a řízení ostatních bloků. Volné piny na portu A slouží navíc jako pomalý, ale přesný (10bitový) A/D převodník. Volné piny na portu A a B mohou být využity pro budoucí rozšíření funkcí osciloskopu.

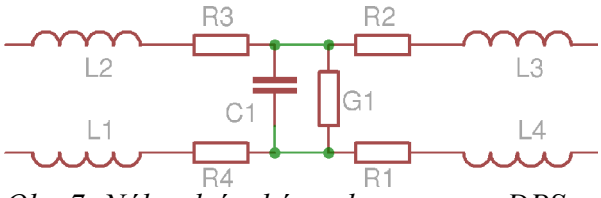


Obr. 6: Rozmístění pinů pro ATmgea32

Signál „Main ready (B)“ informuje buffer, že hlavní řídící jednotka je připravena přijímat data. Naopak „Buffer ready“ informuje hlavní řídící jednotku, že buffer je připraven odeslat data do hlavní řídící jednotky. Zpětná vazba od bufferu je nutná, protože jinak by program nemohl rozlišit, kdy jsou data na sběrnici platná a kdy ne. Jinak je tomu u komunikace s µC, který obsluhuje klávesnici a další vstupní periferie. Zde je přenos dat sériový (Keyboard data), takže stačí pouze nastavit příznak připravenosti ze strany hlavní jednotky. Počátek dat je indikován start bitem o specifické délce. Signál „Analog signal“ je určen pro integrovaný A/D převodník ATmega32. Vstupní signál je upraven pomocí NF zesilovačů tak, aby jeho rozsah byl od 0 V do +5 V. Tento signál je použit pro pomalý, ale přesný softwarový voltmetr. Pro zjištění nastavené úrovně triggeru je použit signál „Analog trigger level“. Pokud nebude zapotřebí zapnutý A/D převodník a oscilátor na desce s bufferem, lze jej dočasně vypnout pomocí „Sleep mode“. To je užitečné například pro mód, kde je zařízení využito jako voltmetr. V tomto případě je zbytečné, aby byl buffer a externí A/D převodník zapnut.

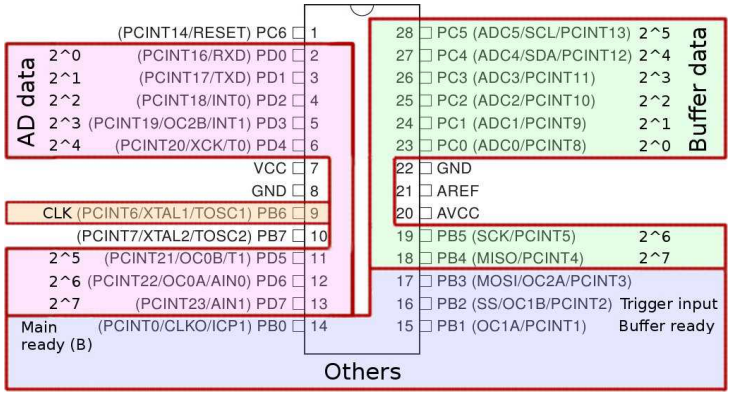
1.5 Analogové obvody s A/D převodníkem a bufferem

Na první pohled dva bloky sloučené do jednoho. Má to ale svůj důvod. Vzhledem k tomu, že je vzorkovací kmitočet vysoký, každá cesta na DPS představuje nejen sériový odpor, jak bychom čekali, ale také sériovou indukčnost, ke které je paralelně zapojena kapacita a svodová vodivost. Znázorněné je to obecně na obr. 7. Z toho plyne jasný závěr: čím delší cesta, tím je větší sériová indukčnost a větší paralelní kapacita, což při přenosu dat na vysokých frekvencích může být na obtíž. Proto je vhodné tyto spoje navrhnout co nejkratší. A proto jsou analogové obvody s A/D převodníkem a bufferem jako jeden blok.



Obr. 7: Náhradní schéma dvou cest na DPS

Jako buffer byla zvolena ATmga88. Počet vstupně/výstupních pinů je 23, což na aplikaci bufferu je dostačující. Velikost flash paměti je 8 kB, takže se jednoduchý program na čip „vejde“. Maximální taktovací frekvence je 20 Mhz a účinnost vykonávání instrukcí je 1 MIPS/MHz, což zaručuje opravdu vysokou výpočetní rychlost [8]. Zde se µC využívá na reagování na trigger a následné sbírání vzorků, které jsou po nashromáždění poslány do hlavního řídícího bloku. Rozložení pinů na ATmega88 je naznačeno na obr 8.



Obr. 8: Rozmístění pinů pro ATmega88

Jak bylo uvedeno výše, rychlost bufferu je zásadní. Proto je přenos dat jak od A/D převodníku k bufferu, tak od bufferu do hlavního bloku a zpět paralelní osmibitový. Vzhledem k tomu, že „plnohodnotný“ 8bitový port je pouze D, byl port D zvolen pro přenos dat z A/D převodníku do bufferu. Pokud by byl zvolen jakýkoli jiný port na tomto µC, pak se ukládání dat z A/D převodníku zkomplikuje a tím se sníží i maximální rychlost, kterou dokáže µC ukládat vzorky. Odesílání vzorků do hlavní jednotky již nemusí být tak rychlé a ani nemůže, protože ATmega32 může pracovat na maximálním taktu 16 Mhz [6]. Zbytek portu B je využit na komunikaci s hlavní jednotkou a pro synchronizaci. Poslední volný pin může být v budoucnu pro další rozšíření funkcí µC. Na pin PB6 je přiveden přesný taktovací signál (CLK), který je synchronní s A/D převodníkem. Tento signál zajišťuje externí oscilátor.

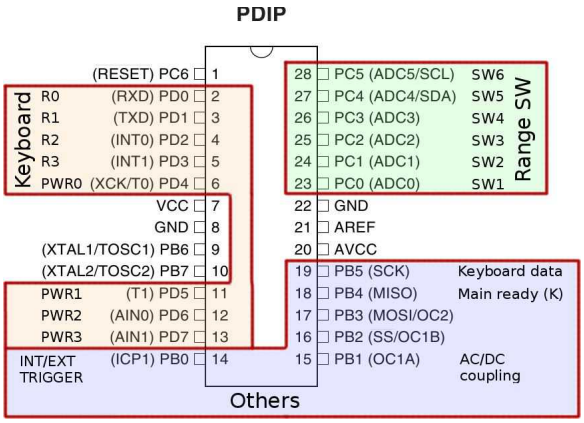
Jako A/D převodník byl použit AD9057 od Analog Devices. Jedná se o převodník, který dokáže vzorkovat rychlostí až 40 MSPS (zakoupená verze, jinak tento typ převodníku dokáže vzorkovat až 80 MSPS). Vyniká nízkou spotřebou (okolo 200 mW za provozu), nízkou cenou (okolo 10 $ [10]), obsahuje již referenční zdroj (2,5 V), power-down mode (spotřeba pod 10 mW) a paralelním 8bitovým výstupem, což je ideální pro buffer, který pracuje s 8bitovou hodnotou [9]. Tento převodník však nemá interní oscilátor, takže o generátor hodin se musí starat uživatel. Proto byl zvolen integrovaný krystalový oscilátor pracující na frekvenci 20 Mhz. Tím maximálně využijeme možnosti bufferu a zároveň nemusíme řešit problémy s útlumem na straně A/D převodníku (frekvenčně dostatečně dimenzované). Navíc bude A/D převodník s bufferem synchronizován, čímž odpadají problémy se čtením z A/D převodníku v době, kdy se na jeho výstupech mění stavy (mohlo by dojít k „nesprávnému“ načtení dat z A/D převodníku a tím by byla informace zkreslená).

1.6 Jednotka pro správu klávesnice a čtení stavu tlačítek

Hardwarově nenáročný blok. Má na starost maticovou klávesnici o 16. tlačítkách, sbírá data o zvoleném napěťovém rozsahu osciloskopu, nastavení triggeru (interní/externí) a AC/DC rozsahu.

Vzhledem k tomu, že odesílaná data jsou pouze data z klávesnice, informace o zvoleném napěťovém rozsahu, nastavení triggeru a AC/DC vazbě, lze využít sériového přenosu dat do hlavní řídící jednotky. Tato data navíc není potřeba aktualizovat tak často jako například data z bufferu. Přenos sice bude trvat delší dobu, ale vzhledem k velikosti přenášených dat to bude nepatrná doba.

Pro tento účel stačí tedy „obyčejný“ µC. Jediným rozhodujícím parametrem je v tomto případě počet vstupně/výstupních pinů a cena. Do této kategorie spadá ATmega8L (již se nevyrábí, přesto je stále snadno dostupná i na českém trhu). Maximální taktovací frekvence je pouze 8 Mhz, počet vstupně/výstupních pinů je 23 a cena se pohybuje okolo 64 Kč [11]. Rozmístění pinů je na obr. 9.



Obr. 9: Rozmístění pinů pro ATmega8L

Port D pracuje s maticovou klávesnicí 4x4 tlačítka. Piny na portu C (mimo PC6) jsou využity pro zjišťování zvoleného napěťového rozsahu od přepínače na vstupu. Port B má hned několik funkcí. Jednak se z jednoho pinu posílají sériová data, jednak se musí zajistit komunikace s hlavní jednotkou, dále má za úkol zjišťovat stav přepínače „interní/externí synchronizace“ a přepínače „AC/DC“. Zbylé piny jsou volné, opět pro budoucí rozšíření.

1.7 Zdroj

Zdroj je vybaven duálním napájením. Přístroj je tedy možné napájet přímo z USB napětím 5 V, nebo přes klasický napájecí konektor. A to od 7,5 V do přibližně 25 V DC, nebo do 18 V AC. Vyšší hodnota vstupního napětí je způsobena použitím stabilizátoru 7805 a faktem, že na vstupu je pro ochranu celého zařízení Grätzův můstek a další prvky. Na druhou stranu „díky“ tomu, že stabilizátory řady 78xx patří mezi lineární, je proudový odběr z externího zdroje téměř vždy stejný, nezávisle na velikosti vstupního napětí (ve výše definovaných mezích). Z toho plyne, že čím větší bude napájecí napětí, tím větší budou ztráty právě na těchto stabilizátorech, které se samozřejmě budou zahřívat. Stabilizátory mají sice tepelnou pojistku, ale vypnutí přístroje v průběhu měření v důsledku přehřátí jistě uživatele nepotěší. Při napájení z USB odpadá problém se zahříváním stabilizátorů, ale je nutno počítat s šumem, který se může „dostat“ z USB sběrnice. Proto je potřeba při používání myslet na tyto fakty a vybírat pokud možno co nejvhodnější napájení.

Možnost dvojího napájení značně zjednoduší problém s napájením v různých podmínkách. Napájení pro analogové a digitální obvody je odděleno. Každá část má svůj stabilizátor pro +5 V. Tím se dosáhne snížením šumu celého zařízení. Pro záporné napětí je použit integrovaný obvod 7660. Toto napětí slouží hlavně pro napájení operačních zesilovačů, které pracují se vstupním signálem.

Jako elektronické spínače jsou použity MOS-FET tranzistory s indukovaným kanálem N. Zvoleny byly tranzistory IRF-9520. Jejich parametry jsou: $I_{DSS} = 10A$, $U_{DS} = 100V$, $U_{GS} = 20V$, $P_D = 70W$, $R_{DS} = 0{,}27\Omega$ a obsahují ochrannou diodu [12]. Důvodem pro takto předimenzovaný tranzistor jsou především praktické zkušenosti při ověřování funkce a ladění celé aplikace. Tranzistor je v tomto případě odolný vůči zkratu, přepólování, přehřátí a dalším nepříznivým

jevům, které mohou při testování na kontaktním poli nastat (uvolněný drátek, nechtěný zkrat atd.). Navíc pouzdro TO220 zajistí dobrý odvod tepla z tranzistorů.

Pro kvalitní napájení analogových obvodů 5 V je vhodné použít lineární stabilizátor. Jeho účinnost je sice menší, než jak tomu bývá u moderních DC/DC měničů, avšak na výstupu DC/DC měničů se objevuje malé zvlnění o vysoké frekvenci. Tento VF signál se pak celým obvodem snadno šíří a poměr S/N se zhorší. Na druhou stranu, pokud má lineární stabilizátor fungovat, musí být na jeho vstupu alespoň 7,0 V ($U_{OUT} + U_{DROP}$), což při napájení z USB kabelu, kde je 5 V je zcela nedostačující. Proto je zde monolitický DC/DC měnič, který mění 5 V na 12 V. Vzhledem k tomu, že analogové obvody budou mít spotřebu v řádu mA (bude potřeba napájet jen několik OZ, které budou pracovat se signálem), lze použít jednoduchý AM1S-0512SZ, který je dostupný i na našem trhu. Výstupní napětí má být 12 V, což zaručuje, že lineární stabilizátor bude fungovat spolehlivě. Maximální výstupní proud je 83 mA [13], takže při využití jen několika málo mA by se měl zdroj jevit jako dostatečně tvrdý a zároveň je zde prostor pro budoucí rozšíření aplikace.

2 Funkce jednotlivých bloků a vzájemná komunikace

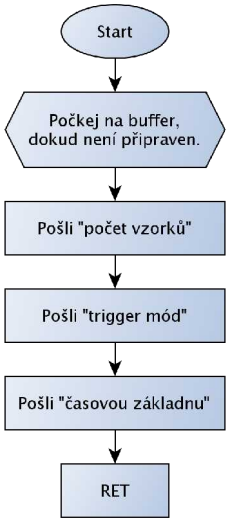
Komunikace některých bloků byla již naznačena v předchozí kapitole. Některé přenosy musí být rychlé, u jiných stačí přenést několik desítek bajtů za vteřinu. U některých periferií je už dán od výrobce způsob přenosu dat. Dále je třeba řešit „aktuálnost“ dat. Například u bufferu je potřeba řešit i validitu přijímaných dat, protože tato přijímaná data byla platná pouze k nějaké nastavené časové základně. Pokud tato časová základna byla uživatelem změněna, pak jsou tato data již neaktuální a pro uživatele nezajímavá, takže je zbytečné je zobrazovat. Dále je potřeba řešit, „kdo na koho bude čekat“. Jde o to, že pokud by měla hlavní řídící jednotka čekat na uživatele, až pustí tlačítko, nemohla by v tuto chvíli v podstatě provádět žádné operace (pokud by se nepoužilo přerušení, což by bylo nevhodné při příjmu dat z bufferu). Toto čekání na uživatele by znamenalo zbytečnou prodlevu, po kterou by byla většina periferií nevyužita. Proto se musí řešit, která periferie bude master a která slave.

2.1 Hlavní řídící jednotka

Srdce celého zařízení. Jak bylo uvedeno dříve, musí komunikovat s LCD, bufferem a μC, který má na starost obsluhovat klávesnici a zjišťovat stav přepínačů. A to není vše. Navíc si musí správně zorganizovat přijatá data z bufferu, aby je mohla korektně zobrazit, zpracovávat žádosti uživatele (z klávesnice), umožnit uživateli nastavit základní proměnné, informovat o aktuálním nastavení, zobrazit základní nápovědu atd.

Jako LCD byl zvolen grafický displej DEM128064 s řadičem kompatibilním s KS-108. Komunikace s LCD je již pevně dána výrobcem. Jak provést inicializaci, zápis a čtení dat z LCD je definováno v datasheetu [14]. Datová sběrnice je 8bitová, obousměrná. Řídící piny jsou CS1, CS2, /RST, R/W, RS, a E. Displej je v podstatě rozdělen do dvou polovin po 64x64 px. Každá polovina má svůj vlastní řadič a signály CS1, CS2 vybíráme buďto levý, pravý nebo oba řadiče (vhodné například pro mazání obsahu na LCD). Signál /RST je reset. Aktivní je v logické nule. Je zapotřebí v podstatě jen při inicializaci LCD. Signál R/W slouží k nastavení „směru dat“. Je-li nastaven na logickou jedničku, pak je nastavena funkce read a datový port na LCD je výstupní, takže datový port, který komunikuje s LCD musí být nastaven jako vstupní. Pokud je R/W v nule, pak se datový port LCD nastaví na vstupní. Takže datový port, který komunikuje s LCD, musí být nastaven jako výstupní. Signálem RS je vybíráno mezi módem „data“ a „instrukce“. Instrukcemi se myslí například zapnutí nebo vypnutí displeje, nastavení řádku (Y souřadnice), nastavení X souřadnice atd. V módu data lze buďto na LCD zapisovat informaci, nebo ji naopak vyčítat. Displej pak zapisuje, nebo čte, z nastavené Y a X souřadnice. Po zápisu jednoho bajtu se X souřadnice automaticky zvýší o 1. Pokud je dosaženo X souřadnice 63, pak následující X souřadnice bude opět 0. Poslední je E, enable. Je to v podstatě potvrzení, že se jsou všechna data již připravena na sběrnici a LCD má vykonat danou instrukci: přijmout nebo odeslat data.

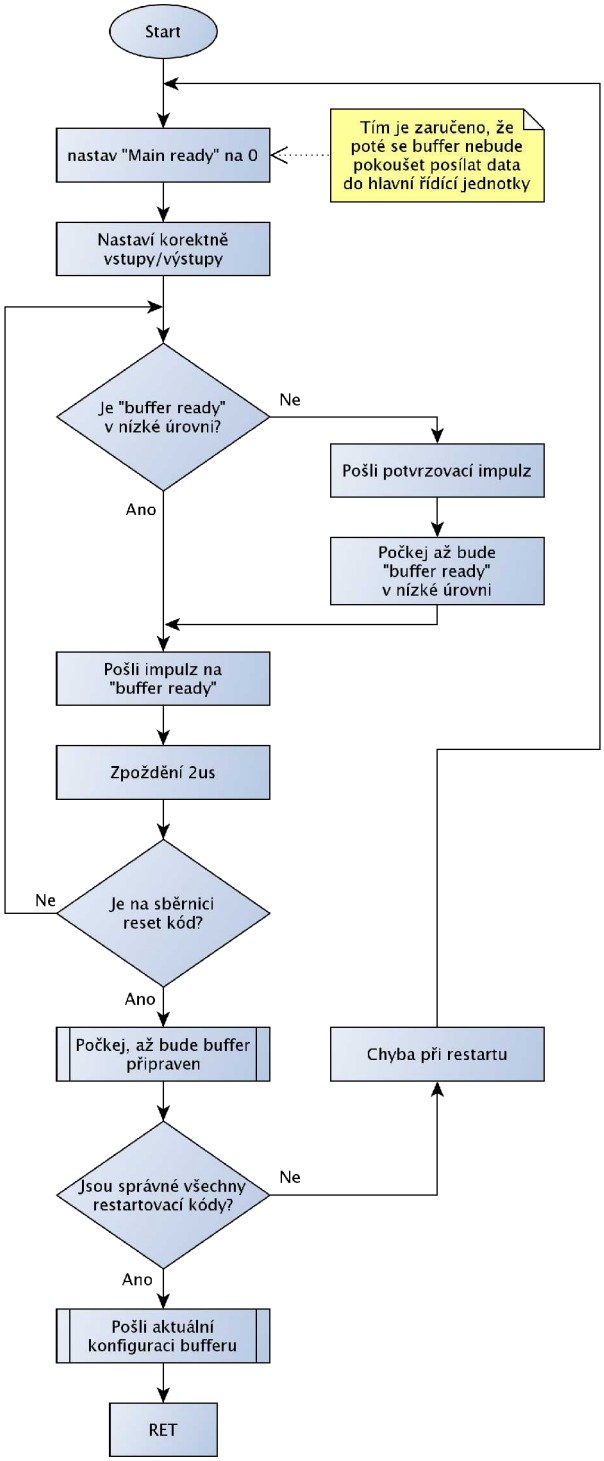
Komunikace s bufferem musí být především velice rychlá (přenáší se i stovky vzorků). Vzhledem k tomuto faktu je výhodné přenášet data paralelně. Pokud bude šířka slova 8 bitů, bude i následné ukládání do mezipaměti SRAM jednoduché a rychlé (ATmega32 je také 8bitová). Pro ušetření počtu vodičů a využití možností AVR je tento přenos dat koncipován jako obousměrný. Protokol pro přenos dat směrem od hlavní řídící jednotky k bufferu je naznačen na obr. 10. Před samotným přenosem dat se nejprve musí buffer inicializovat.



Obr. 10: Protokol přenosu dat z hlavní řídící jednotky do bufferu

Tato inicializace se musí provést pouze jednou při každém spuštění, nebo restartu. V podstatě se jedná o potvrzení tří slov ze strany hlavní jednotky. Tím je zaručeno, že buffer je v jasně definovaném stavu a vyhne se tím problémům při restartech bufferu za chodu celého zařízení. Pokud by bylo pouze jedno potvrzovací slovo, mohlo by se stát, že při restartu bufferu by se právě toto slovo mohlo objevit na sběrnici ještě před tím, než by se restart dokončil. A to by bylo v případě, že by toto slovo bylo shodné s binární hodnotou A/D převodníku. Na příkladu to bude asi jasnější: dejme tomu, že potvrzovací slovo bude 0xFF, tedy 255 v desítkové soustavě. Pokud vyvoláme restart bufferu a buffer právě odesílal nasbírané vzorky a vstupní signál byl v saturaci, pak se hodnoty tohoto vstupního signálu mohou pohybovat od 0x00 do 0xFF. A pokud by se právě odesílal vzorek, který by reprezentoval hodnotu 0xFF, pak by při restartu zůstalo na sběrnici potvrzovací slovo ještě před tím, než by skutečně došlo k restartu bufferu. A to by samozřejmě znamenalo problém. Pokud ale zvolíme vhodně více slov, můžeme se tomuto stavu s velmi vysokou pravděpodobností vyhnout. Potvrzovací slova byla zvolena takto: 0xAA (170), 0x0D (13) a 0xFC (252). Tato slova musí být stejná jak v bufferu, tak v hlavní řídící jednotce. Pokud by vstupní signál měl napodobit restart sekvenci, pak by se jednak musel spustit přesně ve chvíli, kdy hlavní jednotka čeká na dokončení restartu, a jednak by vzorky musely odpovídat uvedeným potvrzovacím slovům. Vzhledem ke koncepci rozvržení kódu, by signál nesměl vypadat ani jako sinusový, ani jako trojúhelníkový, ale ani jako obdélníkový. Z toho plyne, že by bylo i velice těžké napodobit takovýto kód. Navíc je potřeba vzít v úvahu LSB bit z A/D převodníku. Prakticky totiž LSB bit může být považován za šum, protože jeho hodnotě odpovídá velmi malé napětí (okolo 4mV), které bývá na úrovni šumu.

Samotné započetí restartu bufferu je poněkud komplexnější záležitost. Vzhledem k tomu, že v celém zařízení jsou veškeré RESET piny propojeny, aby bylo možné v případě potřeby restartovat celý přístroj, provedení jednoho bloku se stává komplikovanějším. Jako jedna z možností se nabídlo řešení pomocí přerušení na straně bufferu. Dejme tomu, že pin „buffer ready“ je pro buffer výstupním a pokud je v logické 1, pak informuje hlavní řídící jednotku, že na ni čeká (buďto na odeslání, nebo přijmutí dat). V této chvíli není možnost, jak provést restart (viz. kapitola 2.3 - Buffer – obr. 14). To znamená, že jediná možnost, která se nabízí, je provádět restart v momentech, kdy je „buffer ready“ v nízké úrovni. Proces restartu je znázorněn na obr. 11.

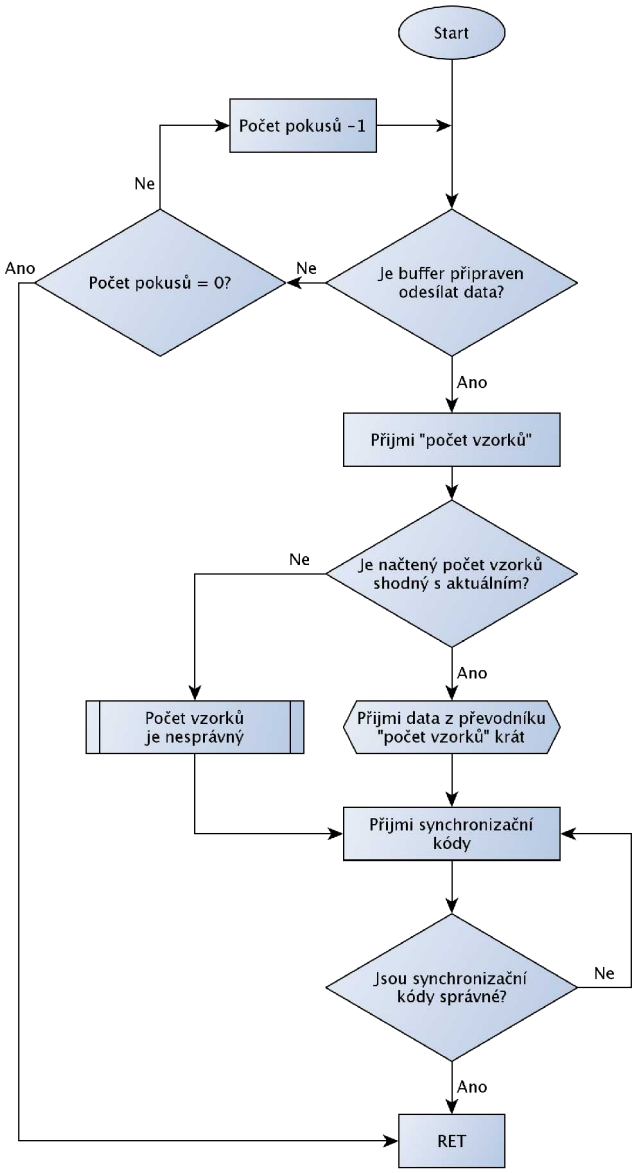


Obr. 11: Vývojový diagram - restart bufferu

Pokud je „buffer ready“ v nízké úrovni, pak na straně bufferu se přepne z výstupního pinu na pin vstupní, aby bylo možné přijmout restart impulz od hlavní řídící jednotky. Pokud je ovšem signál „buffer ready“ ve vysoké úrovni, pak musíme bufferu potvrdit přijímaná nebo vysílaná data. Jedině tak se změní stav „buffer ready“ do nízké úrovně. V tomto případě nezáleží na charakteru potvrzovaných dat, protože se stejně bude provádět restart. Jakmile se pošle restart impulz, tak se spustí krátké čekání. Během této doby by měl buffer nahrát na sběrnici předem definované slovo, které symbolizuje stav, kdy se provádí restart. Avšak tento symbol může být i hodnota z A/D převodníku, za předpokladu, že se restart neprovede. Například kvůli tomu, že hlavní jednotka si

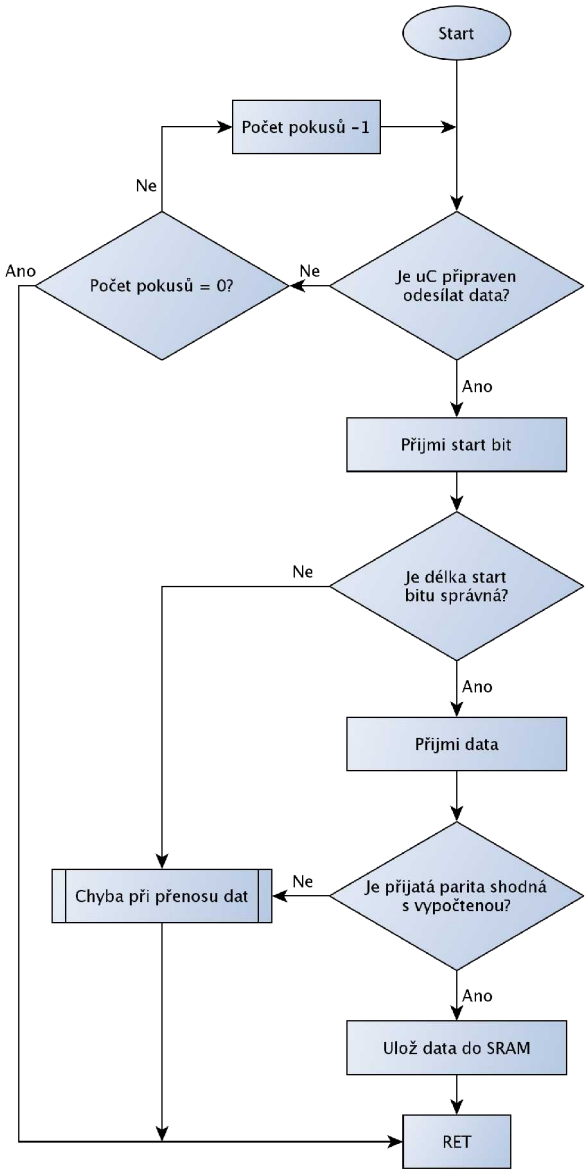
přečte status a než stav vyhodnotí, buffer zatím může změnit stav „buffer ready“ na vysokou úroveň. A proto je zde kontrola reset kódu. Pokud totiž nebude reset kód odpovídat, pak se celá procedura opakuje. Pokud by náhodou hodnota z A/D převodníku byla v daný moment shodná s reset kódem, pak by tento problém měly odstranit restartovací kódy, které jsou uvedeny výše. Na konec se pošlou bufferu konfigurační slova (informace o časové základně atd.).

Protokol pro přenos dat směrem k hlavní řídící jednotce od bufferu je pak naznačen na obr. 12. Zatímco vysílání dat je triviální, příjem dat je z mnoha důvodů složitější. Jednak se musí zkontrolovat, zda počet vzorků přijatých a nastavených je shodný. Pokud totiž není stejný, mohlo by se stát, že počet vzorků v bufferu je vlastně menší než požadovaný, a do paměti by se uložila jenom část hodnot. Vzhledem k tomu, že se cyklus opakuje „počet vzorků“ krát, hlavní jednotka by čekala na další sled dat, ale ten by nikdy nepřišel, protože aby buffer poslal data, musí mu být nejdříve poslány parametry (počet vzorků, trigger mód, časovou základnu). Proto je tu tato „pojistka“, která podchytí tento problém. Dále jsou zde synchronizační kódy. Ty zajistí, že přenos dat bude korektně ukončen (další data z bufferu v tomto momentě již nebudou poslána). Tato synchronizace je tu především pro případ, že je přijatý počet vzorků, nebo časová základna „nesprávná“, a čeká se na konec přenosu dat ze strany bufferu. Synchronizační kódy jsou v podstatě tři po sobě jsoucí 8bitová slova, která musejí být shodná jak ve vysílači (buffer), tak v přijímači (hlavní řídící jednotka). Aby nedošlo k náhodné záměně s „reálnými daty“ (která se také odesílají), byla zvolena trojice slov 0xFA,0x02 a 0xDA. Opět napodobit tuto kombinaci by bylo velice složité. Jedná se prakticky o stejný princip jako u restartu bufferu. Jak je vidět, při čtení dat z bufferu se na buffer čeká jen omezenou dobu. Je to z toho důvodu, že pokud je nastavená velká časová základna, pak by hlavní řídící jednotka musela čekat dlouhou dobu na buffer a tuto dobu by přístroj neodpovídal na žádné reakce, což není uživatelsky přívětivý způsob.



Obr. 12: Protokol přenosu dat z bufferu do hlavní řídící jednotky

Přenos dat mezi μC, který má na starost klávesnici a zjišťování stavu přepínačů, a hlavní řídící jednotkou není jednak na přenosovou rychlost náročný. Navíc aktualizace klávesnice se provádí méně často, než kolikrát se obnovuje obraz na LCD. Proto bude sériový přenos dat vyhovující. Podobně jako u bufferu při čekání na příjem dat se bude čekat jen nějakou dobu. Protokol přenosu dat je naznačen na obr. 13 Pokud během přenosu dat nastane chyba, projeví se to buďto v délce start bitu, nebo v paritním bitu (nebude souhlasit s vypočtenou hodnotou), pak se tato událost zaznamená a algoritmus se ukončí.

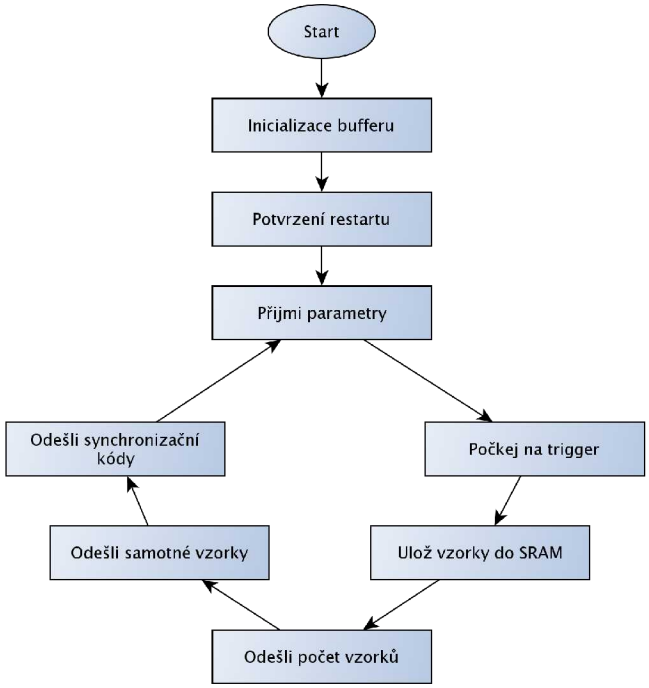


Obr. 13: Protokol přenosu dat z "klávesnice" do hlavní řídící jednotky

Komunikace s LCD displejem je definována výrobcem a její podrobný popis je obsažen v datasheetu [14]. V podstatě ihned po připojení napájení na LCD není možné LCD používat. Nejprve je potřeba provést inicializaci. Ta se provede následně: signál RST se přivede do logické nuly. Zde by měl zůstat alespoň 1 µs. Poté lze na RST nastavit logickou jedničku. Nyní se musí počkat, až se LCD zresetuje. LCD stav zaneprázdněnosti udává logickou jedničkou na 4. datovém pinu. Poté, co je LCD připraven, se může poslat kód, který zapne samotný displej. Nyní je LCD připraven, inicializaci je hotova. Dále je vhodné nastavit počáteční souřadnice displeje (x-ová souřadnice a číslo řádku), aby bylo možné pracovat s LCD za jasně definovaných podmínek.

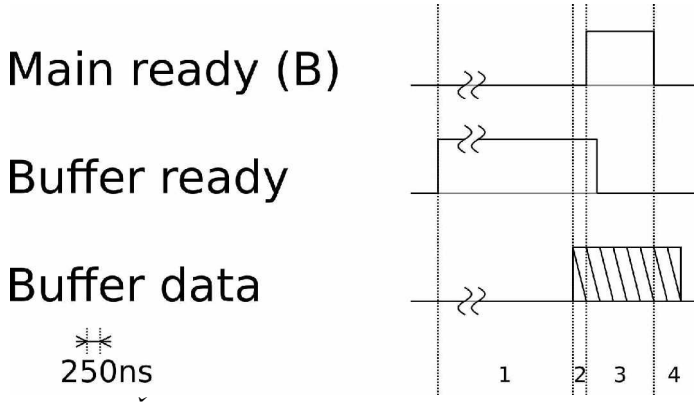
2.2 Buffer

Díky tomu, že hlavní řídící jednotka je „inteligentní“, není již zapotřebí, aby ostatní komponenty byly zbytečně složité. Zjednodušený vývojový diagram pro buffer je na obr. 14. Vývojový diagram je záměrně nakreslen ve tvaru kola, aby bylo na první pohled jasné, že se celý cyklus neustále opakuje a nejsou zde žádné výjimky (jako např. u hlavní řídící jednotky skoky z jednotlivých módů). Náročnost je v tomto případě kladena na rychlost HW a ne na jeho SW propracovanost.



Obr. 14: Zjednodušený vývojový diagram pro buffer

Z vývojového diagramu vyplývá, že se nejprve musí potvrdit restart bufferu. To je zde pro případ, že se provede restart bufferu během chodu celého zařízení, a potvrzením restartu se hlavní jednotka ujistí, že restart je kompletní. Během restartu mohou být na sběrnici různá a hlavně neplatná data, takže tímto je zajištěno, že nedojde k desynchronizaci s hlavní jednotkou. Restart se provádí pomocí přerušení, které je aktivní jen po dobu, kdy je signál „buffer ready“ v logické nule, a provede se krátkým impulzem o délce alespoň 1 cyklu (z pohledu bufferu). Rutina v přerušení vykoná potřebné nastavení a provede „hack“ ve stack poineru. Je třeba brát v úvahu, že při volání přerušení byl nějak naplněn stack, takže místo návratové adresy se zapíše adresa, která odkazuje na počátek inicializace bufferu. Výhoda tohoto „hacku“ je v tom, že oproti watchdogu se nečeká prakticky žádnou dobu. Daná prodleva by se totiž projevila jako „probliknutí“ na LCD, což nebudí dobrý dojem. Poté musí buffer dostat dané parametry, které pak zpracuje a následně je připraven zpracovaná data odeslat. Čtení slova ze sběrnice je popsáno na obr. 15.

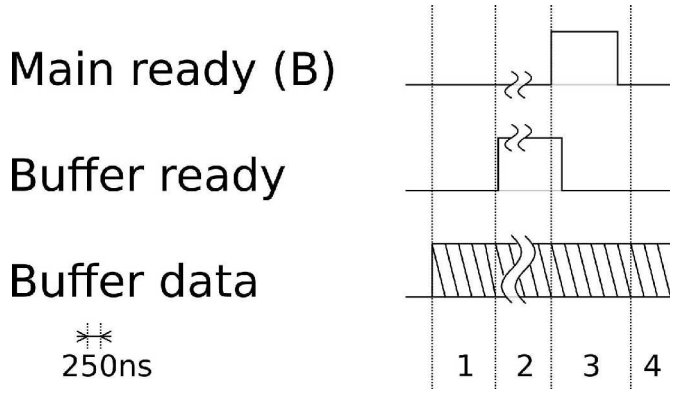


Obr. 15: Čtení slova ze sběrnice z pohledu bufferu

Samozřejmostí je, že před samotnou komunikací musí být nastaveny porty bufferu na sběrnici jako vstupní. V kroku 2 je buffer připraven k příjmu slova a čeká na hlavní řídící jednotku. Délka tohoto intervalu může být různá, v závislosti na tom, „co právě hlavní řídící jednotka dělá“. Podstatné je, že v tomto intervalu hlavní jednotka zjistí, že buffer je připraven, a na sběrnici pošle datové slovo. Tím se program dostává do fáze 2, kdy platná data jsou na sběrnici ještě dlouho před

tím, než je vůbec začne buffer číst. Ve fázi 3 buffer reaguje na připravenost hlavní jednotky a mění svůj stav na logickou nulu. Nyní musí být data na sběrnici alespoň 1,1 µs (konec intervalu 3). V této době si buffer ukládá potřebné informace do paměti. Následuje fáze 4, ve které je vlastně klidový stav sběrnice. Přesto však může, ale nemusí být datové slovo na sběrnici. Přenos jednoho slova tedy trvá přibližně 2 µs. Slovo je 8bitové.

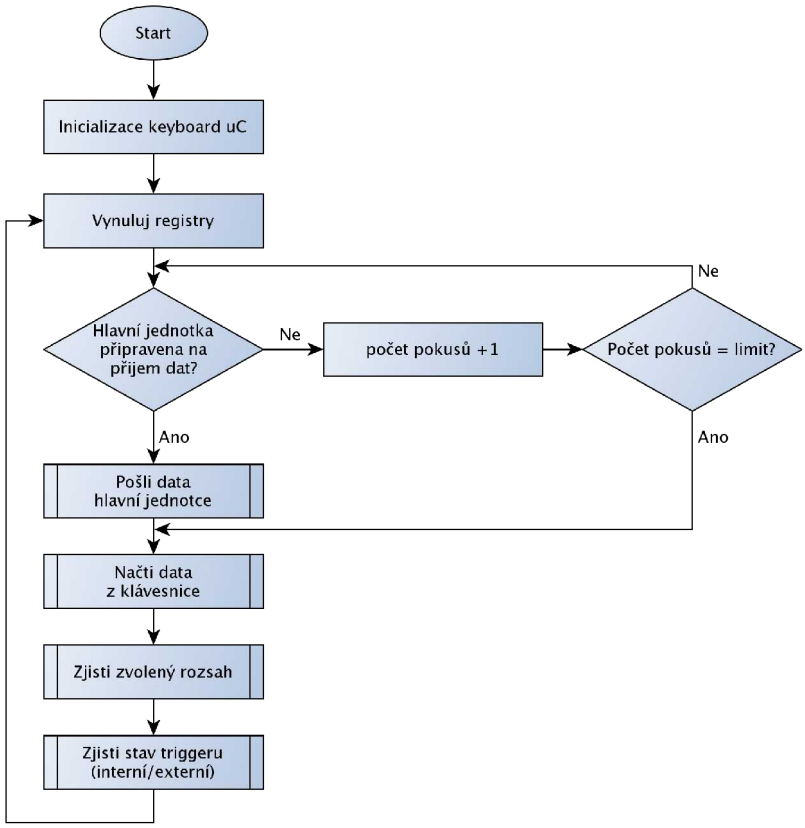
Odesílání dat z bufferu do hlavní jednotky je velice podobné. Graficky je to znázorněno na obr. 16. Ve fázi 1 buffer nahraje slovo na sběrnici. Poté, co je slovo na sběrnici nastaveno, následuje krok 2. Zde si buffer změní stav na logickou jedničku, která značí, že buffer je připraven. Nyní se jen čeká na hlavní jednotku, až bude také připravena. Tato doba je proměnlivá, opět záleží na hlavní jednotce, jaké provádí operace. V kroku 3 je již hlavní jednotka připravena. Buffer na to zareaguje a změní svůj stav. Dále čeká 1 µs. Za tuto dobu by si měla hlavní jednotka informaci uložit do své paměti. Ve fázi 4 mohou, ale nemusí být na sběrnici platná data. Toto je klidový stav. Rychlost přenosu dat je opět závislá na mnoha faktorech. V podstatě se ale dá přenést jedno slovo pod 2 µs. Opět je šířka slova 8 bitů.



Obr. 16: Zápis slova na sběrnici z pohledu bufferu

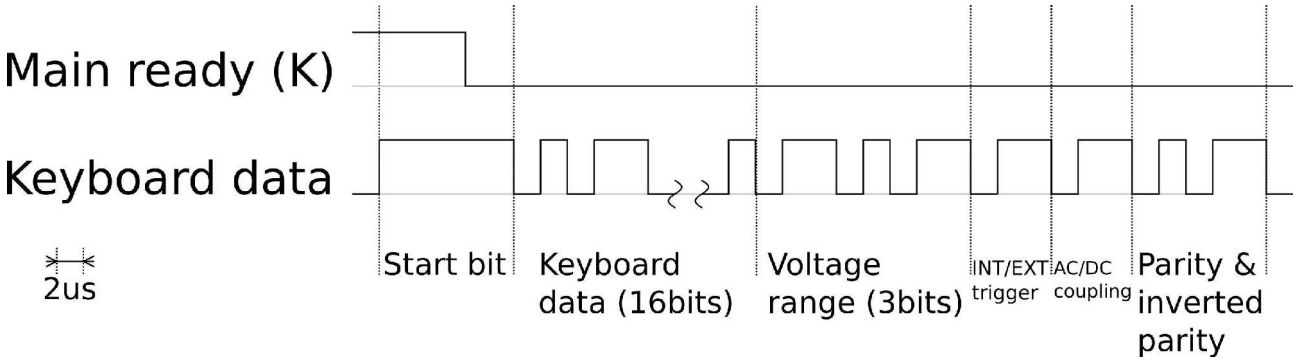
2.3 Správa klávesnice a přepínačů

Opět nikterak složitá část, ale o něco sofistikovanější než buffer. Jednak se musí hlídat stisknutá tlačítka, zákmity na tlačítkách a pak se musí dostatečně často kontrolovat, zda není hlavní jednotka připravena přijímat data. Je důležité zvolit si správný poměr čekání na hlavní jednotku a četností kontroly stavu tlačítek. Pokud by byla čekací doba na hlavní jednotku příliš velká, pak se sice data posílala často do hlavní jednotky, ale reakce klávesnice by byla velice „zpomalená“. Naopak, při druhém extrému, by se informace o stisknutých tlačítkách ukládala velice často, ale data by se „nedostala včas“ do řídící jednotky, takže by se klávesnice opět jevila jako „zpomalená“. Opět zjednodušený vývojový diagram je na obr. 17.



Obr. 17: Zjednodušený vývojový diagram pro správu klávesnice a tlačítek

Vzhledem k charakteru dat (stačí odesílat „jednou za čas malý datový tok“), je výhodné data posílat sériově. Komunikační protokol ze strany tohoto µC vypadá následovně: odešle se start bit o délce 10 µs. Následuje rámec o délce 20 bitů. Ten obsahuje informaci o stisknutých klávesách (16 bitů), zvoleném napěťovém rozsahu (3 bity), triggeru (1 bit) a vazbě (1 bit). Pak je odeslána parita (1 bit) a následuje invertovaná parita (1 bit). Pak následuje klidový stav na sběrnici. Logická nula má délku pulzu 2 µs, logická jednička pak 4 µs. Mezi jednotlivými bity jsou mezery o délce 2 µs. Je to v podstatě analogie s morseovkou. Možný datový rámec je znázorněn na obr. 18.



Obr. 18: Znázornění přenosu dat do hlavní řídící jednotky

Počet bitů v rámci musí být přijímači (hlavní jednotce) předem znám, jinak by došlo ke špatnému vyhodnocení. Prakticky je velice snadné přidat do rámce další bity, ale jak bylo uvedeno výše, je nutné, aby se změna provedla i v přijímači. Přijímač reaguje na náběžné hrany, takže v podstatě je jedno, jak dlouhé mezery mezi jednotlivými bity jsou. Měly by být ale dlouhé alespoň 4 takty cílového přijímače.

3 Softwarové řešení

Komunikace s ostatními bloky je jen nepatrná, přesto důležitá součást SW. Narozdíl od zpracování dat, které se provádí „uvnitř“ µC, je komunikace s ostatními bloky závislá i na HW. Ale samotná data jsou bez správné interpretace k ničemu. Tato kapitola se zabývá zpracováním dat „uvnitř“. Pro úplnost: 8 bitů je 1 bajt. Bit je tedy nejmenší jednotka, kdežto bajt je celé slovo, v tomto případě 8bitové, protože architektura všech µC je 8bitová.

3.1 Hlavní řídící jednotka

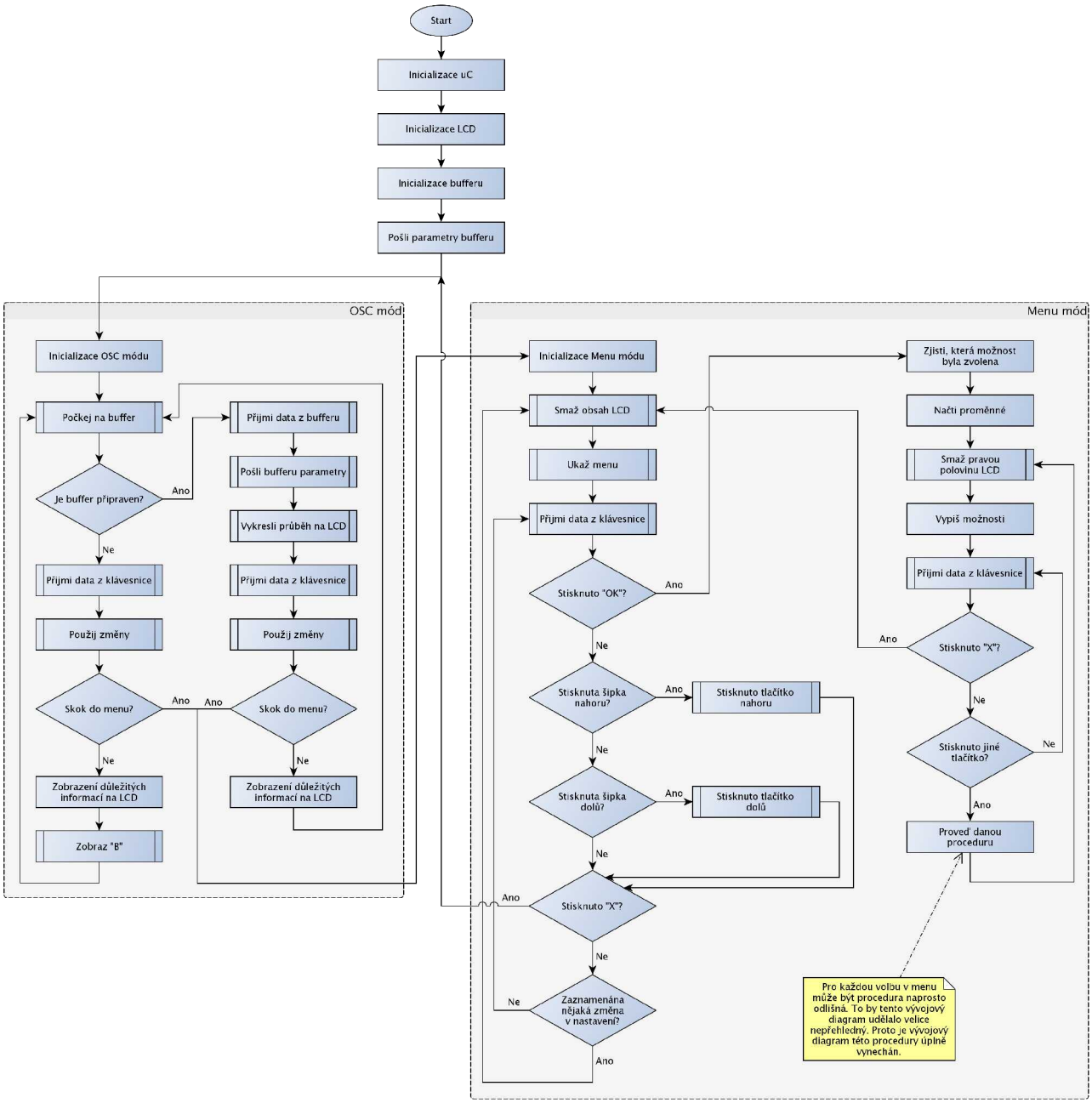
V této části je především naznačen funkční algoritmus celé aplikace. Spousta věcí je pro zjednodušení vynechána. Bližší informace o SW budou probrány v následující kapitole. Jak bylo uvedeno výše, komunikace s ostatními periferiemi není jediná věc, kterou musí tato část obsluhovat.

Jak je vidět z obr. 19, softwarově je zařízení rozděleno do dvou módů. První, který se automaticky spustí zapnutím přístroje, je mód osciloskopu (OSC mód). V tomto módu se zařízení chová jako osciloskop. V druhém módu se zařízení chová jako aplikace, jakou známe například z PC platformy. Můžeme zde nastavovat různé proměnné (například na co má trigger reagovat), spustit jednoduchou animaci „spořič“, nebo se podívat do nápovědy. Jak je vidět už z vývojového diagramu, tato část je poměrně softwarově náročná a rozlehlá (viz. poznámka v obr. 19) oproti osciloskopickému módu. Navíc zde musí být řešena různá ošetření vstupních proměnných.

Ihned po restartu/spuštění se zařízení chová jako osciloskop. Trigger je nastaven na náběžnou hranu. Nastavení triggeru (interní/externí) je dáno uživatelem, resp. polohou přepínače. V případě externího triggeru je synchronizace dána externím zdrojem. Doporučená logika je +5 V. Pokud zvolil uživatel jako trigger interní, pak záleží na HW nastavení aplikace. A to jednak na nastavené napěťové úrovni triggeru a jednak na nastavené hysterezi. Pokud se zrovna čeká na buffer (buďto ukládá vzorky, nebo ještě čeká na trigger), tak se v levém horním rohu objeví písmeno „B“, které symbolizuje zaneprázdněnost ze strany bufferu. Pokud je časová základna příliš malá na to, aby se vstupní signál vzorkoval realtime, pak se použije periodické vzorkování. To je symbolizováno opět v levém horním rohu písmenem „P“. Následuje informace o triggeru (interní/externí). Pokud je trigger nastaven na interní, je to symbolizováno znakem „I“. V opačném případě se zobrazí písmeno „E“. Následuje informace o stejnosměrné („DC“), nebo střídavé („AC“) vazbě. Na následujícím řádku je uvedena časová základna na dílek. Další řádek uživatel informuje o nastaveném rozsahu, který je opět vztažen na jeden dílek. Následuje naměřené napětí peak-to-peak.

Celý průběh je možné „zastavit“ pomocí tlačítka „RS“ (Run/Stop) a posouvat horizontálně pomocí šipek. Zde platí, že čím více vzorků je uloženo v paměti, tím více lze zobrazený průběh posouvat. Počet vzorků lze nastavit v menu. V defaultním nastavení je tato volba nastavena na „Auto“. Toto znamená, že počet vzorků se dynamicky mění s časovou základnou. A to tak, že pro krátké časové základny je zvolen větší počet vzorků než pro dlouhé časové základny. Je to z toho důvodu, že při zvolení dlouhé časové základny (např. 1s/div) je relativně velký časový interval mezi jednotlivými vzorky. Z toho plyne, že pokud má přístroj vzorkovat 200 vzorků, místo 100, pak bude doba vzorkování dvojnásobná.

Do hlavního menu se lze dostat pomocí tlačítka „M“. Pohyb v menu lze provádět několika způsoby. Šipkami nahoru a dolů se lze pohybovat vertikálně. Tlačítko „OK“ (fajfka) znamená potvrzení volby. Naopak tlačítko „X“ je určeno pro zrušení volby. Šipky vlevo a vpravo umožňují podobnou funkci jako „OK“ a „X“, ale s tím rozdílem, že jsou poněkud „hbitější“ (v podstatě něco jako dvojklik). Pořadí v menu je snadno proměnné, takže je možné přidat nové funkce na počátek menu, nebo změnit pořadí kvůli četnosti používání.



Obr. 19: Zjednodušený vývojový diagram pro hlavní řídící jednotku

Během doby, kdy je uživatel v menu, buffer stále pracuje a buďto čeká na trigger, nebo shromažďuje vzorky. Pokud je v menu změněn parametr triggeru (např. změna z náběžné na sestupnou hranu), buffer je okamžitě resetován a jsou mu poslány ihned nové parametry, takže se nemusí čekat, až se nashromáždí všechny vzorky. Toto je vhodné v případech, kdy se uživatel rozhodne změnit parametry měření. Díky tomuto restartu za běhu se nemusí čekat na vykonání předchozí úlohy a tím se šetří čas uživatele.

Přístroj má svoji vlastní nápovědu. Lze ji najít v hlavním menu pod názvem „Help“. Tato nápověda je velice stručná, ale měla by uživateli poskytnout dostatečné množství informací. Velikost nápovědy je jednak omezena velikostí paměti flash a jednak dlouhý text se na displeji o malém rozlišení špatně čte. Za zmínění stojí, že veškeré texty jsou uloženy v odděleném *asm* souboru jako posloupnost ASCII hodnot. Většina znaků odpovídá standardům ASCII. Vzhledem k tomu, že je celý systém osmibitový, je dalších 128 hodnot použito pro speciální symboly a znaky. Z výše uvedeného plyne možnost měnit jednoduše jazyk prostředí, bez zásahu do vlastního kódu. Opět podobný princip je využit u aplikací na platformě PC.

Aby se zamezilo „blikání“ displeje, tak se musí co nejvíce zkrátit interval mezi „čistým“ displejem a zápisem na displej. To znamená, že je výhodné nechat na displeji „stará data“,

zpracovat a připravit co možná nejvíce dat pro zobrazení a až poté smazat displej a potřebná data na něm co nejdříve zobrazit.

Přenos dat s ostatními periferiemi je pro větší přehlednost popsán výše u každé periferie zvlášť, protože proces komunikace s každou periferií může být různý.

Vzhledem k rozsáhlosti kódu (celý kód k hlavní jednotce má přes 16 000 řádků) byl celý kód rozdělen do několika souborů, podobně jako je tomu u knihoven v jazyce C. V tab. 2 je uveden vždy název souboru a stručný popis.

Tab. 2: Přehled souborů ve zdrojovém kódu pro hlavní řídící jednotku

Název souboru	Popis
draw_points.asm	Načítá vzorky z bufferu a ukládá je do SRAM. Dále obsahuje programy pro interpretaci nashromážděných vzorků na LCD a to jak bodového zobrazení, tak „vektorového“.
float_operation.asm	Nejen matematické operace s datovým typem float. Umožňuje načítat float pomocí ukazatelů X,Y (pro paměť SRAM) a Z (pro flash), odkládat float čísla do „proměnných“ x0 až x4 (počet lze jednoduše softwarově navýšit). Z matematického hlediska je možné provádět sčítání, odčítání, násobení, dělení, rychlé násobení a dělení 2 (díky vlastnostem binární soustavy je možné tyto operace provést mnohem efektivněji než u předchozích algoritmů) a dokonce počítat druhou odmocninu.
graphic.asm	Obsahuje podprogramy pro zobrazování dat z LCD. Umožňuje zapisovat na LCD po bajtu, nastavovat aktivní řádek, volit X-ovou souřadnici, smazat celý displej, zobrazit 8bitové číslo, zobrazit jeden digit (tzn. číslo od 0~F), vykreslovat rastr pro osciloskopický mód a pomocný rámeček, ve kterém se zobrazují informace o měření.
initLCD.asm	Rutina pro inicializaci LCD. Pro zjednodušení využívá především podprogramy z graphic.asm.
keyboard_control.asm	Dokáže přijímat data z µC, který má na starost klávesnici a tlačítka. Řeší některé složitější operace při stisku daných tlačítek v osciloskopickém módu (mění časové základny, zajišťovat vertikální posuv, atd.). Mimo jiné obsahuje i jednoduchý podprogram, který čeká na stisknutí libovolné klávesy. Ten je vhodný například při zobrazení informativních sděleních, kde je vhodné nějakým způsobem ověřit, že si uživatel přečetl danou zprávu.
lang_EN.asm	Lokalizační soubor. Jsou zde uloženy všechny texty. Pokud by bylo potřeba, aby celý přístroj používal jiný jazyk, pak stačí vyjít z tohoto souboru a změnit pouze text. Bohužel nelze text zadávat přímo, ale buďto jako čísla ASCII hodnot, nebo jako určitý kód. Pro tento případ byl napsán jednoduchý (ale omezený) program v jazyce C (soubor text2asm_db.c). Ten umí po zadání textu vygenerovat potřebný kód. Jeho omezenost spočívá u speciálních symbolů, které by bylo složité definovat.

Název souboru	Popis
letters.asm	Obsahuje kompletní binární databázi všech symbolů, takže pokud je zapotřebí změnit styl jednotlivých symbolů, lze to provést právě tady. Dále obsahuje podprogram jak pro vypsání jednoho symbolu (rychlý, ale umí vypsat pouze jeden znak), tak i podprogram pro vypisování textu na LCD a to i takového, který „se nevejde“ na LCD (jako vstupní proměnná je mimo jiné i informace, která udává, od kterého řádku v textu se má text vypisovat).
main_uC.asm	Hlavní soubor. Zde se includují ostatní soubory. V tomto souboru jsou definované téměř veškeré proměnné (proto je změna parametrů celkem jednoduchá). Je zde definováno rozvržení pracovních registrů a organizace SRAM paměti. Určuje inicializaci jednotlivých komponent. Zde jsou podprogramy využívány už jako celistvé bloky, takže lze snadno vyčíst princip zařízení jako celku. Drtivá většina kódu jsou pouze definice.
math.asm	Převážně matematické funkce. Vypočítává offset vzorků, o který se má signál „posunout“ (pokud je např. nastaven počet vzorků na 200, pak se musí zobrazit 50. vzorek až 150. vzorek. A pokud uživatel provedl posuv, pak musí vypočítat, od kterého vzorku se má signál vykreslovat). Umí převést 8bitové číslo na stovky, desítky a jednotky. Toho pak využívají některé vykreslovací podprogramy, protože se celý proces značně zjednoduší. Dále obsahuje podprogramy, které dokážou najít maximální a minimální hodnoty ve vzorcích, detekovat periodu a z té pak vypočítat frekvenci, napětí RMS a střední hodnotu v periodě. Dál obsahuje podprogram, který umí převést číslo typu float a zobrazit ho na LCD.
menu_mode.asm	Zde jsou uloženy veškeré podprogramy pro menu. Obsahuje podprogramy, které vypisují jednotlivé možnosti na LCD (v závislosti na tom, na které straně se uživatel nachází), řeší co dělat, když uživatel stiskne v konkrétním podmenu danou klávesu. Ač se to zdá jako triviální záležitost, tato část kódu tvoří podstatnou část celkového programu. Musí se řešit spousta proměnných (například pamatovat si poslední volbu v menu) a to na systému, který má sice velkou SRAM, ale málo pracovních registrů. Zápis do SRAM je totiž komplikovanější než zápis do pracovních registrů, ale paměti SRAM je více. Celkově je princip zobrazování menu složitý a bude probrán dále.
osc_info.asm	Obsahuje podprogramy, které indikují zaneprázdněnost bufferu, pozastavení celého procesu (Run/Stop funkce), periodické vzorkování, synchronizaci, vazbu (AC/DC), mód triggeru, časovou základnu, napěťový rozsah a naměřené veličiny.
voltage_mode.asm	Tento soubor obsahuje jeden celý mód. Zde je popis, jak se zařízení bude chovat ve „voltage modu“ (chová se jako voltmetr). Podprogram se volá jako celek. Prakticky obsahuje pouze dva podprogramy, které lze použít nezávisle na tomto módu. Jednak je to funkce, která čeká, až bude A/D převod hotov, a jednak je to funkce, která zobrazí 10bitové číslo na „posuvníku“. Zbytek je v podstatě jeden velký podprogram.

Název souboru	Popis
wait_4Mhz.asm	Soubor podprogramů pro zpoždění. V tomto případě je počítáno s případem, kdy hlavní řídící jednotka pracuje s taktovací frekvencí 4 Mhz. Obsahuje následující zpoždění: 2 µs, 3 µs, 1 ms, 10 ms, 50 ms, 100 ms, 500 ms a 1000 ms.
wait_8Mhz.asm	To samé jako wait_4Mhz.asm, ale s tím rozdílem, že zpoždění je zde vypočítáno pro takt 8 Mhz. Navíc obsahuje zpoždění o délce 1 µs.

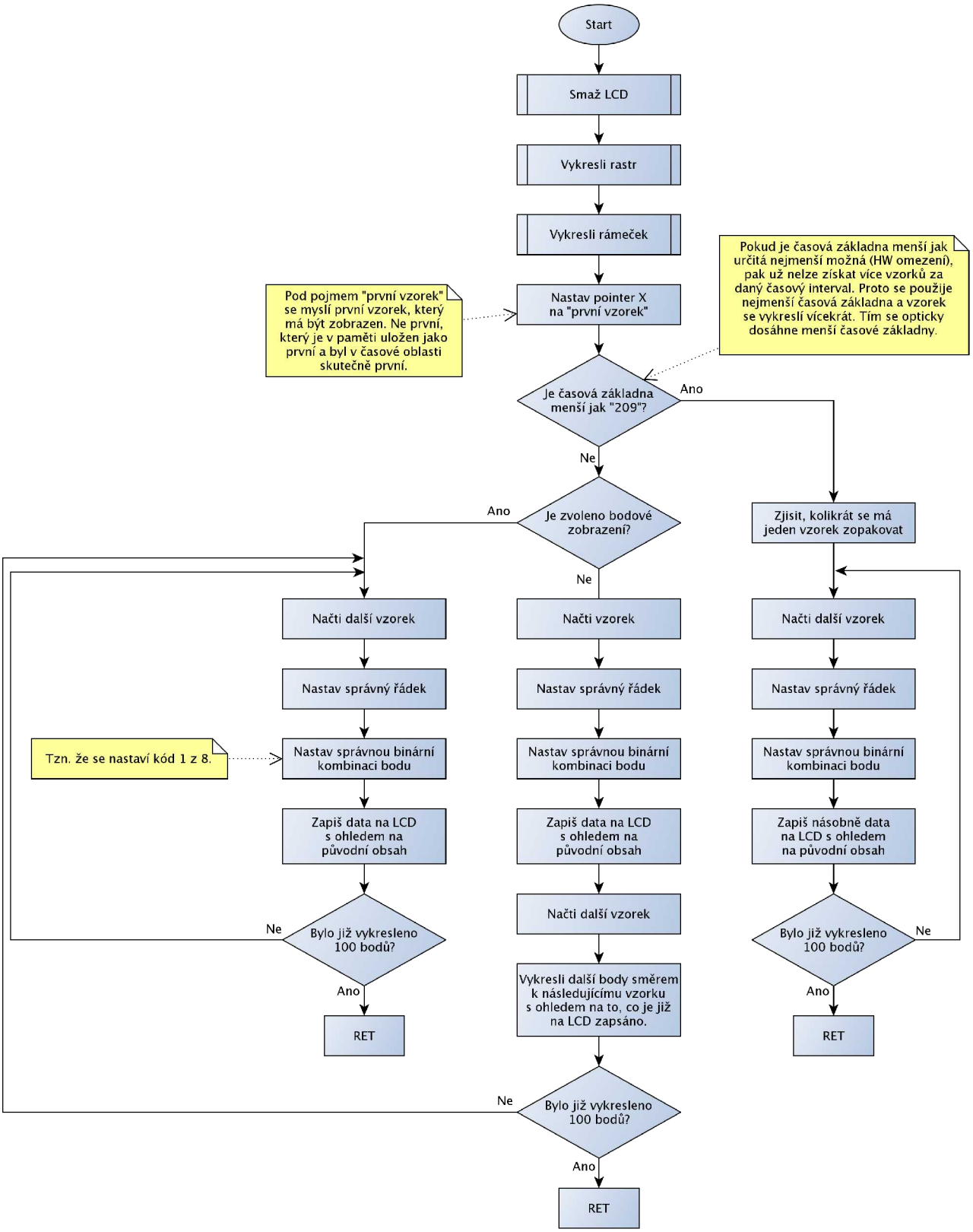
Jak bylo uvedeno výše, systém je ve většině případů nastavován proměnnými, které jsou především uloženy na počátku souboru *main_uC.asm*. To umožňuje rychle upravovat celý systém (počínaje vzhledem, přes nastavování komunikačních kódů až po korekce). Stručný popis některých definic a proměnných je v tab. 3. Z programového hlediska není těžké definici změnit na proměnnou a obráceně. Na druhou stranu definice zabírají místo pouze ve zdrojových kódech.

Tab. 3: Přehled některých definic a proměnných použitých v hlavní řídící jednotce

Definice/proměnná	Popis
sync_code_1 až sync_code_3	Synchronizační kódy pro komunikaci s bufferem. Tyto tři kódy udávají konec přenosu dat od bufferu (v případě, že odesílá uložené vzorky). Jejich hodnota musí být stejná jak v hlavní řídící jednotce, tak i v bufferu. Mohou nabývat hodnot od 0x00 až po 0xFF.
reset_code_1 až reset_code_3	Resetovací kódy. Slouží pro ujištění se hlavní jednotky, že je provedena inicializace korektně a že je buffer připravený na příjem parametrů. Opět tyto kódy musí být shodné na obou stranách. Mohou nabývat hodnot od 0x00 až po 0xFF.
reset_in_progress	Pokud hlavní řídící jednotka vyvolá restart pulz, pak je od bufferu očekáváno, že do určité doby (řády µs) nahraje na sběrnici právě tuto hodnotu. Tím je bufferem naznačeno, že restart je v průběhu a v nejbližší době bude možné potvrdit inicializaci bufferu. Může nabývat hodnot od 0x00 až po 0xFF.
noise_level	Udává hranici, kdy je vstupní signál ještě brán jako šum a kdy ho lze už vyhodnocovat jako užitečný signál. Pokud je signál vyhodnocen jako šum, pak se automaticky přeskočí výpočty periody, frekvence, RMS atd. Na obrazovce se pak zobrazí nápis „Low signal“. Může nabývat hodnot od 0x00 až po 0xFF.
Vol_mode_AD_correction	Softwarová korekce offsetu nízkošumového zesilovače. Ideálně by se na trimru R28 mělo nastavit 2,500 V. To se samozřejmě nemusí přesně podařit (nastavení probíhá je v jednotkách mV). Proto je tu tato korekce, která je přičtena k číslu z A/D převodníku. Číslo je ve dvojkovém doplňku, takže je možné zadávat i záporná čísla. V tomto případě je doporučeno zadávat proměnnou jako znaménkové číslo v desítkové soustavě kvůli větší přehlednosti v kódu.
Vol_mode_num_of_sampl	Udává velikost „okna“ pro mód voltmetru. Čím je okno větší, tím více vzorků se uloží do paměti a tím přesnější je vypočtená průměrná hodnota. Ale čím více je vzorků v paměti, tím má celé měření větší setrvačnost (naměřená maximální, minimální a průměrná hodnota) a tím déle musí uživatel čekat na ustálení. Rozsah hodnot: 2~127.

Definice/proměnná	Popis
default_number_of_samples	Počet vzorků, který bude použit ihned po zapnutí přístroje. Použitelný rozsah hodnot je od 100~255.
default_TimeBase	Časová základna, která bude použita po zapnutí přístroje. Aktuální rozsah hodnot je od 207~230, kde 207 je nejkratší časová základna a 230 je nejdelší časová základna. Jednotlivé délky časových základen lze najít ve zdrojovém kódu v souboru *osc_info.asm*.
default_trigger_mode	Mód triggeru, který bude použit po zapnutí přístroje. Aktuální možnosti jsou: 0 – HW trigger, vzestupná hrana ; 1 – HW trigger, sestupná hrana ; 2 – kontinuální ukládání vzorků (nečeká se na žádné spuštění).
default_number_of_samples_mode	Pokud je hodnota nastavena na 0, pak se počet vzorků mění adaptivně v závislosti na zvolené časové základně. Druhá možnost se hodnota 1. V tomto případě si počet vzorků definuje uživatel, pokud nedojde k výjimce (U velmi krátkých časových základen je počet vzorků nastaven „natvrdo“ kvůli určitým omezením).
default_waiting_interval_buffer	Definuje počet „pokusů“, po kterých se čeká na buffer. Čím větší tato hodnota bude, tím déle bude hlavní jednotka čekat na buffer a tím častěji mohou být přenášeny data mezi bufferem a hlavní řídící jednotkou. Na druhou stranu větší hodnota znamená větší dobu čekání. Může nabývat hodnot od 0~255.
default_waiting_interval_keyboard	To samé jako výše uvedené, ale pro μC, který obsluhuje klávesnici a zjišťuje stav přepínačů. Opět může nabývat hodnot od 0 po 255. Pokud je ale hodnota menší než 5, pak prakticky k přenosu nedochází, resp. jen velmi zřídka.

Princip vykreslování průběhu na LCD je zobrazen na vývojovém diagramu obr. 20. Celý proces je značně zjednodušený. Pokud je zvoleno „vektorové“ vykreslování, pak se musí provádět výpočty navíc a mnohonásobně se zvýší počet vyčítání dat z LCD, což trvá největší dobu. Z toho plyne, že „vektorové“ vykreslování je časově mnohem náročnější než bodové. Proto je v menu možnost změnit typ vykreslování pro zvýšení obnovovací frekvence na LCD.



Obr. 20: Princip zobrazování průběhu na displej

Podprogram „write_text_from_DB“ dokáže vypsat na LCD text, který je uložen ve flash paměti. Dokáže vypisovat i text, který je mnohem delší, než jaká je kapacita LCD a to díky tomu, že umí zobrazovat text až od definovaného řádku. Vzhledem k tomu, že celý algoritmus je značně komplexní, jsou zde pouze uvedeny vstupní proměnné a jejich význam. Vysvětlení proměnných je ve zkratce v tab. 4. Níže jsou uvedeny konkrétní příklady použití.

Tab. 4: Proměnné používané podprogramem pro vypisování textu na LCD

Proměnná	Popis
data	Nastavuje x-ovou souřadnici na LCD. Povinná proměnná. Nabývá hodnot 0~127.
T-bit	Nastavuje barvu písma. Pokud je nastaven na 0, pak je písmo černé a pozadí je bílé. Jinak je písmo bílé a pozadí je černé. Tato funkce je především využita při zvýrazňování vybrané položky.
write_text_from_DB_line_SRAM	Definuje, na kterém řádku se má text nacházet. Tato proměnná je nepovinná, ale pokud se text „nevejde“ na aktuální řádek, pak je potřeba zapisovat na další. A tehdy se použije tato proměnná. Pokud je tato proměnná použita, zůstane v ní uložena hodnota naposledy použitého řádku. Může nabývat hodnot 0~7.
wr_txt_from_DB_start_line_SRAM	Určuje, od kterého řádku se má text vypisovat. To je vhodné v případech, kdy je text příliš dlouhý, než aby se „vešel“ na jednu obrazovku. Tato proměnná je nepovinná a při návratu z tohoto podprogramu se vždy nastaví na 0 (což znamená: vypisuj text od počátku). Tato proměnná může nabývat hodnot 0~255.
wr_txt_from_DB_show_X_lines_SRAM	Udává, kolik řádků textu se má na displej zobrazit. Jedná se o nepovinnou proměnnou a při návratu z podprogramu se automaticky nastaví na hodnotu 8 (což znamená, že se má vypsat 8 řádků). Tato proměnná může nabývat hodnot 1~8! Číslování zde začíná od 1.
Auto_break_line_SRAM	Pokud je proměnná nastavena na 0, pak je vypnuté automatické zalamování řádků. To znamená, že pokud se slovo „nevejde“ na řádek, tak je rozděleno. Pokud je ale tato hodnota 1, pak se zapne automatické zalamování řádků. To znamená, že pokud podprogram zjistí, že se slovo „nevejde“ na daný řádek, pak začne dané slovo vypisovat až na další řádek. Tato proměnná je povinná a není tímto podprogramem měněna.

Příklad použití tohoto podprogramu pro výpis textu uloženého v paměti flash. Nechť text začíná na prvním řádku LCD, jeho počáteční x-ová souřadnice je 5px, není delší jak 1 řádek a barva písma je černá. Ať je to vidět černé na bílém. Zalamování textu tedy nemusí být řešeno.

```
Test_text_1:                    ; Návěští pro daný text
.DB BH,ne,nl,nl,no,sp,nw,no,nr,nl,nd,em,etx,0 ; Zakódovaný text. Může být prakticky
      ; kdekoliv v paměti. Nyní obsahuje text „Hello world!“

      LDI ZH,HIGH(2*Test_text_1)
      LDI ZL,LOW(2*Test_text_1) ; Těmito dvěma příkazy byl nastaven Z pointer na
                                ; text, který má být vypsán

      LDI data,0                ; Nastaveni proměnné „data“
      CALL select_line_both     ; Tímto se nastaví jako aktivní řádek číslo 0, tedy
                                ; první řádek na LCD. Jako argument slouží právě
                                ; registr „data“, do kterého byla uložena hodnota 0
      CLT                       ; Nastaví T-bit na 0 → černé písmo
```

```
LDI data,5                ; Tím se nastavila X-ová souřadnice
CALL write_text_from_DB   ; A text se zobrazí na LCD
```

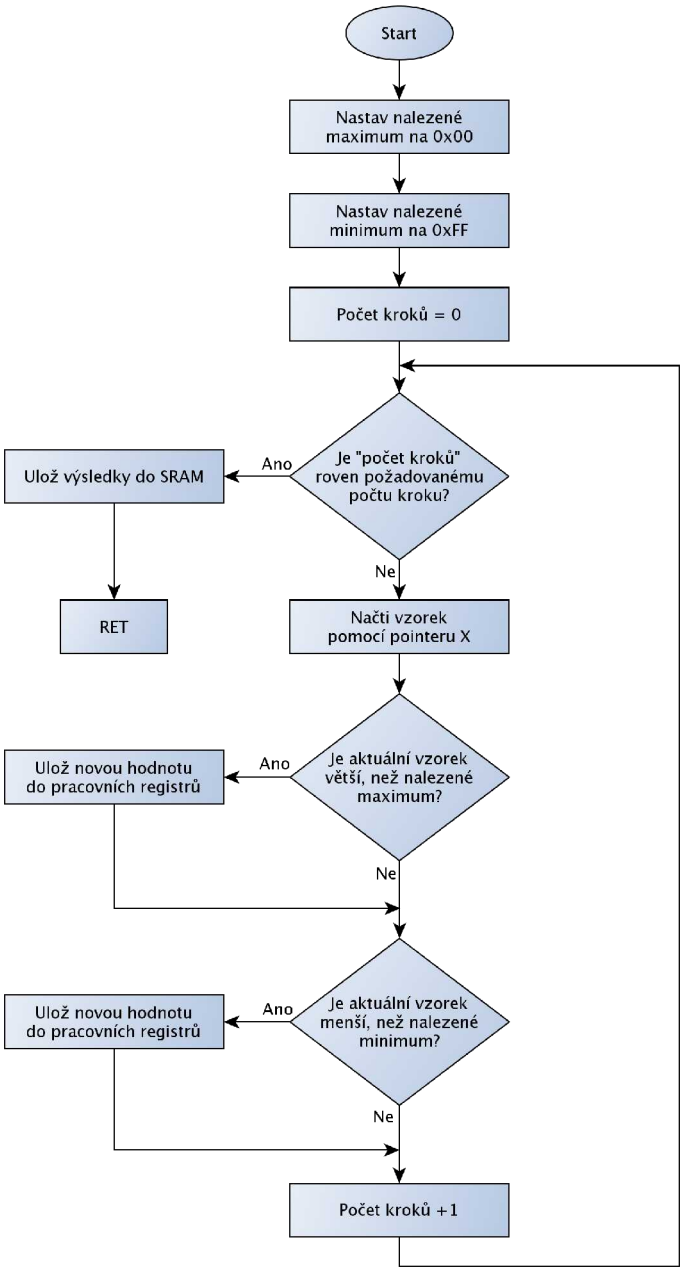
Co se týče nastavení řádku, pokud byl předtím displej smazán, pak byl automaticky nastaven jako aktivní 1. řádek na LCD, takže toto nastavení by ani nebylo potřeba. Tím by se kód omezil na 5 řádků. A pokud by nebyl předtím měněn ani T-bit, pak i instrukce *CLT* mohla být vynechána. Pokud je ale potřeba vypsat jen několik řádků z jinak obsáhlého textu, musí se definovat i nepovinné proměnné.

V následujícím příkladu budou záměrně použity všechny proměnné, aby jejich význam byl jasnější. Zakódovaný text nebude záměrně uveden. Jeho obsah je v tomto případě irelevantní. Dejme tomu, že je zapotřebí, aby se text začal vypisovat na *2. řádku LCD* od souřadnice 7. Vzhledem k tomu, že je text obsáhlý, *začne* se *vypisovat* například *od 13. řádku*. A z nějakého důvodu je potřeba, aby se *vypsaly* pouze *tři řádky* a pak už nic. A ať už to nějak vypadá, nechť je *zapnuté* automatické *zalamování* řádku. Barva písma bude nastavena na bílou (tzn. černé pozadí). Kód by pak mohl vypadat následovně:

```
LDI ZH,HIGH(2*Test_long_text_635)
LDI ZL,LOW(2*Test_long_text_635) ; Nastavení Z pointeru na daný text

LDI data,1                ; Nastavení promenné „data" na 1
STS write_text_from_DB_line_SRAM,data  ; Vzhledem k tomu, že se text bude
                                       ; vypisovat na více řádků, tak je nutné
                                       ; nastavit i tuto proměnnou
CALL select_line_both     ; Tím se nastaví 2. řádek na LCD

LDI data,12
STS wr_txt_from_DB_start_line_SRAM,data ; Tím je nadefinováno, že text bude
                                        ; vypisován až od svého 13. řádku
                                        ; (Čísluje se od nuly)

LDI data,3
STS wr_txt_from_DB_show_X_lines_SRAM,data     ; Nastavení zobrazených řádků na 3

LDI data,1
STS Auto_break_line_SRAM,data          ; Tím se nastaví automatické zalamování

SET                                    ; Nastavení bílého písma

LDI data,7   ; Nastavení X-ové souřadnice
CALL write_text_from_DB
```

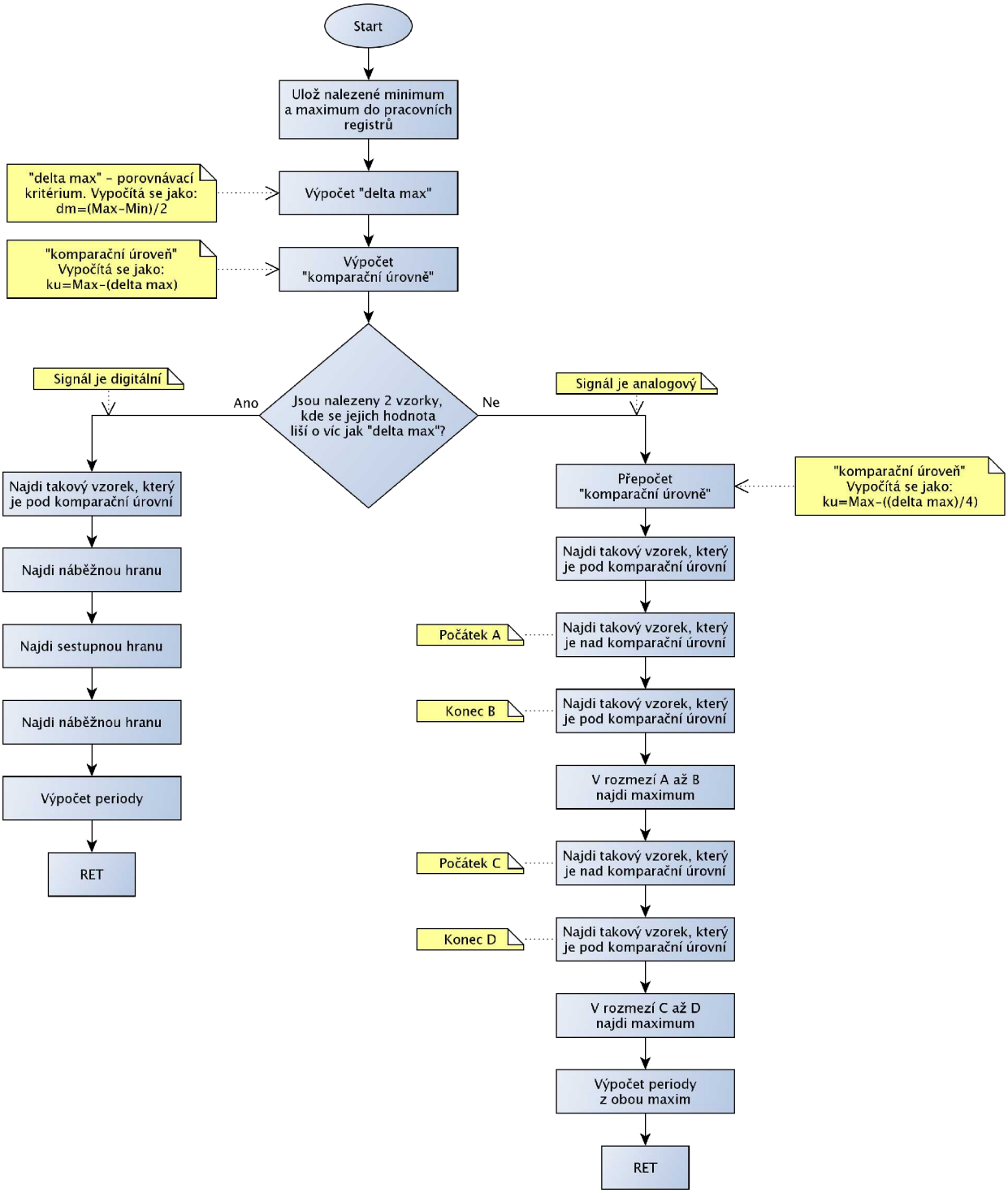
V tomto případě je kód o dost složitější, ale to je kvůli potřebám programátora. Jinak by bylo velice obtížné zobrazovat dlouhé texty s posuvem. Jak je vidět, už ovládání toho podprogramu není nejjednodušší záležitost. Navíc tento podprogram je i výpočetně náročný. Pokud je potřeba vypsat pouze jeden znak, je jeho použití neefektivní. Proto byl napsán podprogram „write_char_from_flash", který vypíše právě jeden znak na displej. Jediné vstupní parametry jsou informace o souřadnici x (data) a Z pointer, který opět odkazuje na znak, který se má vypsat a jehož barva je definovaná T-bitem. Nastavení řádku neřeší. To znamená, že nevyhodnocuje, zda se daný symbol vůbec „vejde" na daný řádek. Na druhou stranu jeho výpočetní náročnost je minimální. Použití je prakticky stejné jako v prvním případě s tím rozdílem, že by se vypsal pouze první znak.

Pro určení periody je nutné znát rozkmit vstupního signálu a podle toho vhodně nastavit parametry. To znamená, že nejprve se musí najít minimální a maximální hodnota vzorků uložených v paměti. Princip je na obr. 21. Jako vstupní proměnné jsou: X pointer (udává, od které adresy v SRAM má začít prohledávat) a pracovní registr „data" (udává, kolik vzorků se má prohledat). Výsledek se pak uloží na patřičná místa do SRAM.



Obr. 21: Princip hledání minima a maxima ve vzorcích

Takže pokud byl spuštěn podprogram, který našel minimum a maximum, pak je možné spustit další podprogram, který se pokusí najít ve vzorcích periodu. Celý algoritmus je o něco sofistikovanější, ale základní principy jsou zobrazeny na obr. 22. Opět se jedná o zjednodušení.

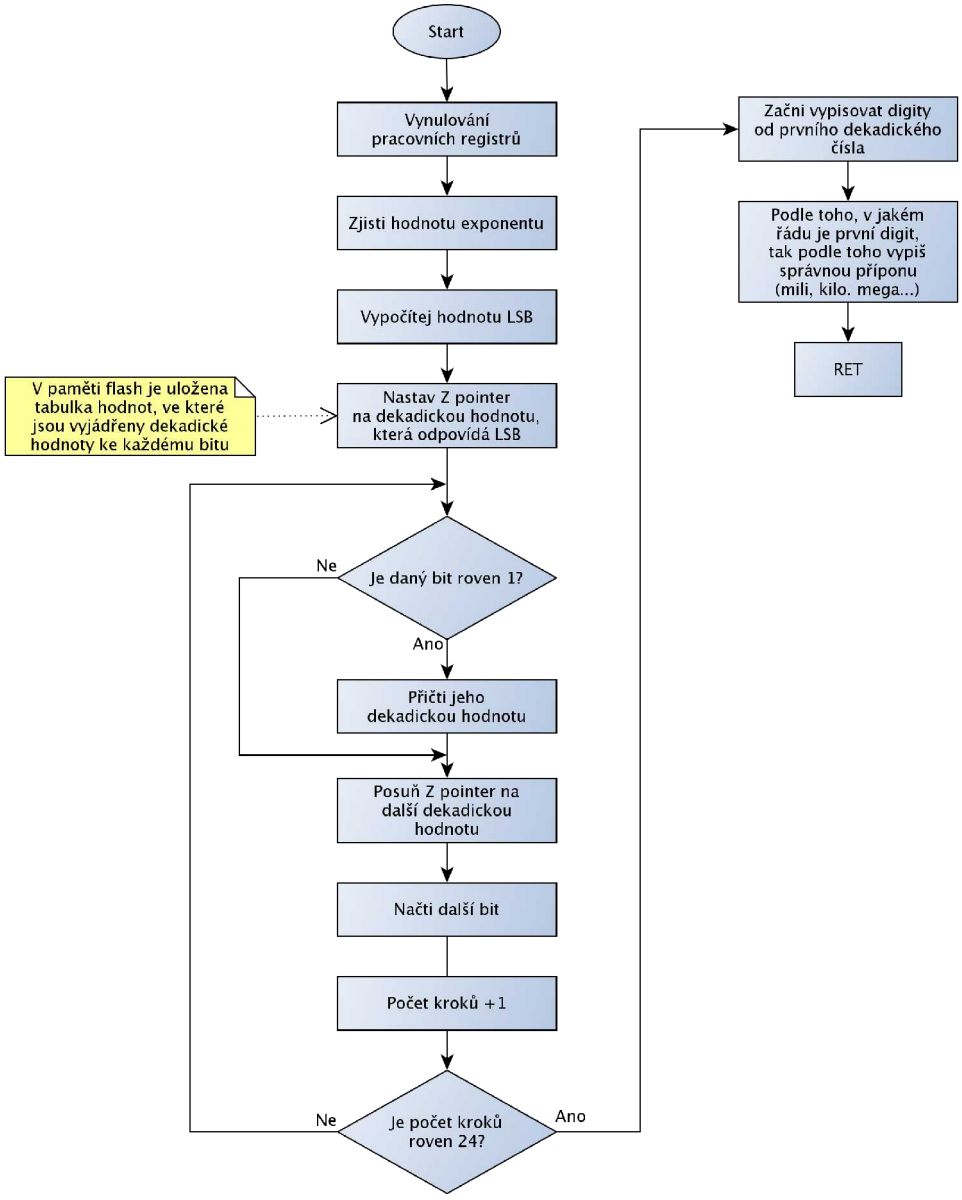


Obr. 22: Algoritmus pro hledání periody

Jediná věc není v celém algoritmu uvedena. A to je „vyskočení“ z kterékoli části podprogramu za podmínky, že počet prohledávaných vzorků překročí definovanou mez (proměnná „data“ definuje, kolik vzorků se má prověřit). V tom případě se délka periody nastaví na 0, takže výsledek je označen jako „perioda nenalezena“ → frekvenci nelze vypočítat. Samozřejmě, že existuje i více metod. Jedna z možných je najít maximum a minimum, určit střed a pak najít 3 vzorky, které budou mít tuto hodnotu [17]. Tři proto, že v principu sinusový, obdélníkový nebo pilovitý průběh prochází těmito body za jednu periodu 3x. Problém nastává, pokud nejsou tyto klíčové vzorky stejné. Pak se musí hledat například nejbližší vzorky. Avšak zde aplikovaný algoritmus je více adaptivní. Pokud je detekováno, že se jedná o digitální signál, pak se použije algoritmus, který funguje obdobně jako

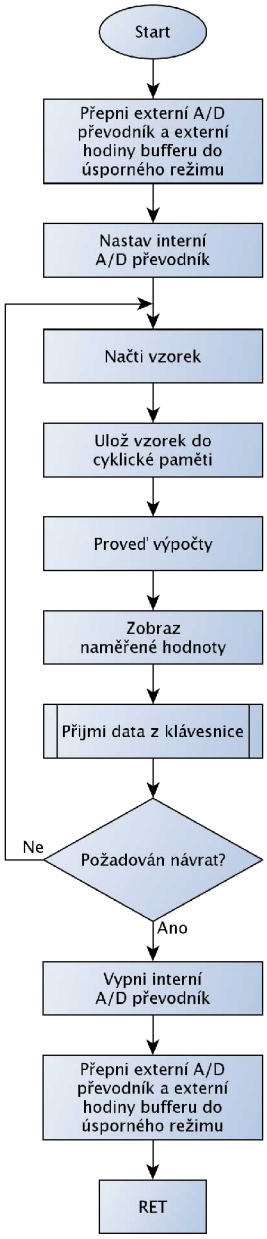
výše uvedený. Ale pokud je detekováno, že je signál analogový, pak se měří de-facto vzdálenost mezi dvěma špičkami. Tento algoritmus je poněkud složitější, ale subjektivně je robustnější.

Co se matematického vybavení týče, není důležité pouze umět pracovat s floatem, ale také provést jeho interpretaci. Myšlenka je následující: každou jedničku a nulu lze s určitou přesností vyjádřit v desítkové soustavě. To znamená, že pokud bude existovat pro každou váhu bitu předem definované číslo v desítkové soustavě, pak stačí provést součet odpovídajících čísel v desítkové soustavě. Protože rozsah datového typu float je obrovský, byla by celková tabulka hodnot také obrovská. Proto byl napsán program, který dokáže rozlišit hodnoty od 2^{-32} (platné 4 číslice nabízí teprve hodnota 2^{-23}) až po 2^{32}. Podle exponentu ve floatu se určí hodnoty jednotlivých bitů. Následuje součet v desítkové soustavě. Následně se vypíší 4 digity s vhodně vloženou desetinnou čárkou. Podle prvního čísla v desítkové soustavě se určí jednotky (giga, mega, kilo, mili atd.). Schématicky je celý proces naznačen na obr. 23.



Obr. 23: Princip převodu a zobrazení datového typu float

Zařízení je možné využít jako voltmetr. Protože jsou vzorky 10bitové (je použit interní A/D převodník v ATmega32), celý proces ze komplikuje. A to především kvůli 8bitové architektuře µC. Princip je ale vždy stejný. Opět je naznačen jako vývojový diagram na obr. 24.

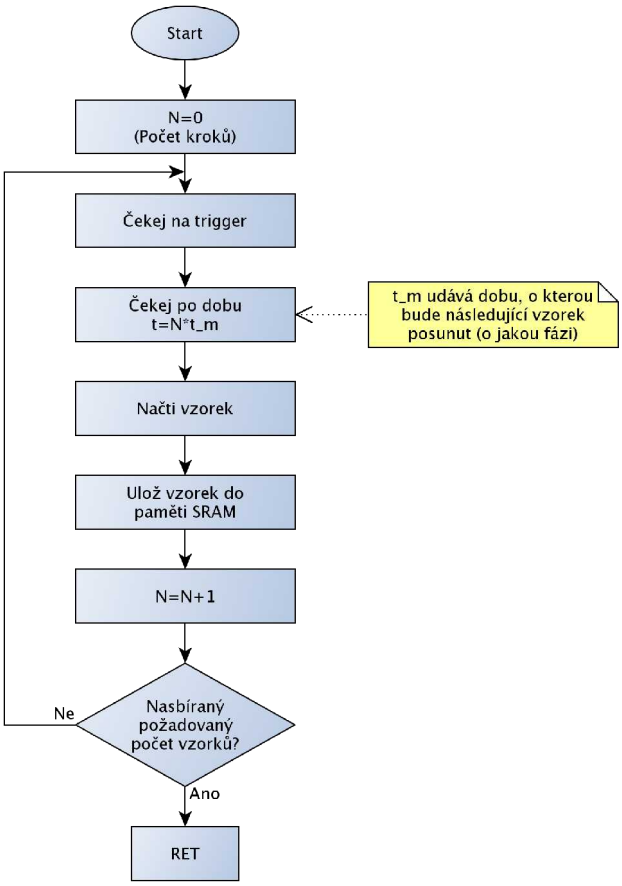


Obr. 24: Vývojový diagram pro mód voltmetru

V podstatě se jedná o velice lineární proceduru. V celém procesu je ještě skryta (záměrně) korekce. Jak bylo uvedeno výše, ta je tu pro případ, že se nepodaří přesně nastavit HW trimr. Jako cyklická paměť je použita paměť, do které se v módu osciloskopu ukládají vzorky. Na jednu stranu se tím ušetří spousta paměti, ale na druhou stranu uživatel přijde o naměřené výsledky v módu osciloskopu.

3.2 Buffer

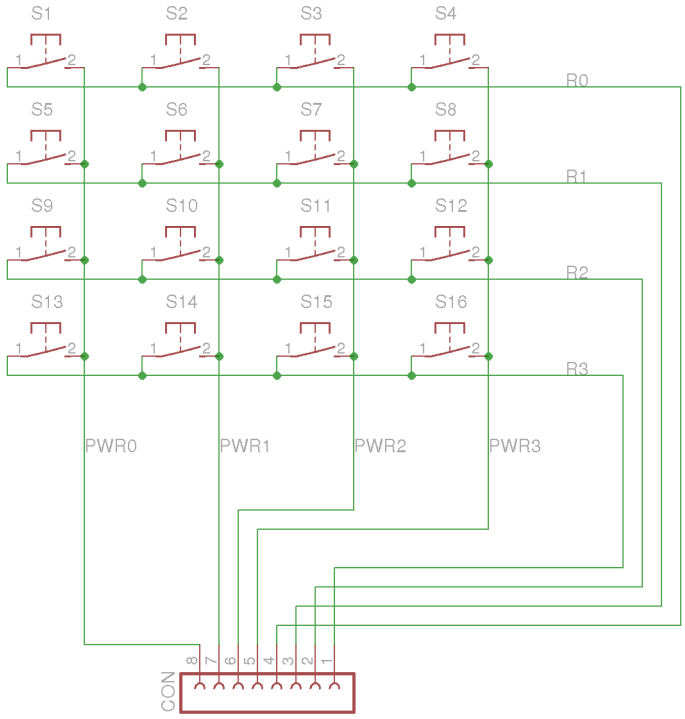
Vývojový diagram byl v podstatě probrán v minulé kapitole v kontextu komunikace s hlavní řídící jednotkou. Ze SW hlediska se nejedná o nijak sofistikovanou část. Data z A/D převodníku se ukládají bez jakékoli modifikace do paměti SRAM a odtud se opět bez jakékoli modifikace vysílají směrem k hlavní řídící jednotce. Za zmínku stojí možnost periodického vzorkování, díky kterému se dá dosáhnout vyššího vzorkovacího kmitočtu, avšak pouze pro periodické signály. Princip periodického vzorkování je uveden na obr. 25.



Obr. 25: Princip periodického vzorkování

3.3 Správa klávesnice a přepínačů

Opět po SW stránce primitivní záležitost. Vývojový diagram byl naznačen v minulé kapitole především ve spojitosti komunikace s hlavní řídící jednotkou. Při zjišťování napěťového rozsahu se v podstatě převádí kód 1 z 6 na celé číslo. To znamená, že není nutné přenášet 6 bitů, ale stačí pouze 3. Protože je klávesnice maticová, musí být i přístup k ní maticový. Díky tomu se ušetří sice spousta signálových vodičů, ale na druhou stranu je vyžadován sofistikovanější přístup. V tomto konkrétním případě obsahuje klávesnice 16 tlačítek, čemuž odpovídá 8 vodičů (4*4=16). Schématicky je to naznačeno na obr. 26.



Obr. 26: Schématické zapojení maticové klávesnice

Princip je triviální. Dejme tomu, že jako napájecí piny budou sloužit „sloupce“ a jako čtecí budou „řádky“. Uživatel stiskl tlačítko S11. Mikrokontrolér postupuje následovně (dejme tomu, že je na počátku svého podprogramu). Nastaví pin PWR0 jako výstupní a jeho úroveň nastaví na log. 1. Pak zkontroluje stav na pinu R0. Pokud by bylo stisknuto tlačítko S1, pak by se na R0 objevila log. 1. Ale to stisknuto nebylo, takže úroveň na R0 je log. 0. Program vyhodnocuje, že tlačítko S1 není stisknuto. Stejným způsobem testuje R1 (tlačítko S5), R2 (S9) a R3 (S13). V tomto případě (stisknuto pouze S11) na všech naměří log. 0. Poté nastaví pin PWR0 do log. 0 a zároveň nastaví na pinu vysokou impedanci (nebo ho nastaví jako vstupní). Pin PWR1 nastaví jako výstupní a nastaví ho na log. 1. Opět prověří R0 (S2), R1 (S6), R2 (S10) a R3 (S14). Ani na jediném nezjistí log. 1. Pin PWR1 nastaví do log. 0 a opět mu nastaví vysokou impedanci. Pin PWR2 nastaví jako výstupní a nastaví na něm log. 1. Proběhne opět kontrola R0 až R3. Na R2 je ale vysoká úroveň. Mikrokontrolér vyhodnotí, že tlačítko S11 bylo stisknuto.

Celý proces se značně komplikuje, pokud je vyžadováno, aby byl systém reagoval na stisknutí více tlačítek najednou. Naštěstí to není případ tohoto zařízení, a proto stačí relativně jednoduchý podprogram na zjištění, která klávesa je aktuálně stisknuta.

4 Hardwarové řešení

V této kapitole jsou shrnuty některé základní technické parametry klíčových součástí. Dále je zde porovnání s některými komerčními přístroji a na závěr jsou zde uvedeny poznámky k návrhu DPS.

4.1 Základní technické parametry použitých součástek

Protože každá součástka má jiné „hlavní“ parametry, jsou součástky rozděleny do dvou tabulek. V tab. 5 jsou parametry mikrokontrolérů. Jejich parametry se totiž dají mezi sebou srovnávat, takže pro větší přehlednost je jim věnována celá tabulka. V tab. 6 jsou pak parametry ostatních součástek (A/D převodník, VF OZ).

Tab. 5: Porovnání parametrů použitých µC [1]

Typ	Počet pinů	Max. Takt [MHz]	Flash [kB]	SRAM [kB]	EEPROM [B]
ATmega88	28	20	8	1	512
ATmega8L	28	8	8	1	512
ATmega32	40	16	32	2	1024

Tab. 6: Technické parametry ostatních klíčových součástek

Funkce	Typ	Parametry
A/D převodník	AD9057	8bitové rozlišení, paralelní, až 40 MSPS, napájení +5 V, interní reference 2,5 V, pracovní rozsah 2~3 V (díky tomu je zajištěna linearita v celém rozsahu), potřeba externí časování, 200 mW spotřeba při převodu, 10 mW při sleep módu
Hodinový signál	JCO 14-3-B	Frekvence 20 MHz, stabilita ±50 ppm, spotřeba při napájení 5 V 25 mA, enable/disable pin
OZ	AD8066	MOS-FET OZ, vstupní proud 2 pA, vstupní impedance 1000 GΩ || 2,1p F, rychlost přeběhu 180 V/µs, napájení 5~24 V, výstup rail-to-rail
OZ	AD8041	Rychlost přeběhu 160 V/µs, vstupní impedance 160 kΩ || 1,8p F, napájení 3~12 V, výstup rail-to-rail
LCD	DEM 128064A SY H-LY	Rozlišení 128x64 px, zelené podsvícení, řadič kompatibilní s KS0108B, paralelní přenos dat, integrovaný DC/DC měnič pro napájení krystalů, 5 V logika, spotřeba (bez podsvícení) 2,62 mA, data zůstávají uložena v paměti i po vypnutí LCD

4.2 Naměřené hodnoty a porovnání s komerčními přístroji

V tab. 7 až tab. 19 je uvedeno srovnání některých naměřených hodnot s komerčními přístroji. V případě použití přístroje jako osciloskopu, byl signál z generátoru přiveden přímo na vstup přes T-článek. Z druhé strany T-článku byl připojen komerční osciloskop. Napájení bylo provedeno z USB a to z notebooku, který byl napájen ze sítě skrze univerzální adaptér Nakami UCN-279 100W.

Tab. 7: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=0,1V ; f=1kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [mV]	U_{MAX} [mV]	U_{RMS} [mV]	U_{AVG} [mV]	f [kHz]
AVR osc	200	0,1	117,1	46,87	39,07	1,64	943,3
Agilent DSO-X 3014A	200	0,1	157	70	35,7	-3,18	1,0200

Tab. 8: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=1V ; f=1kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [mV]	U_{MAX} [mV]	U_{RMS} [mV]	U_{AVG} [mV]	f [kHz]
AVR osc	200	0,2	970,3	478,1	346,5	-3,000	1,000
Agilent DSO-X 3014A	200	0,2	1,05	519	352	7,86	1,0014

Tab. 9: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=5V ; f=1kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [V]	U_{MAX} [V]	U_{RMS} [V]	U_{AVG} [mV]	f [kHz]
AVR osc	200	1,0	4,804	2,390	1,732	-14,06	1,000
Agilent DSO-X 3014A	200	1,0	5,03	2,47	1,77	-39,3	1,0003

Tab. 10: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=0,1V ; f=10kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [mV]	U_{MAX} [mV]	U_{RMS} [mV]	U_{AVG} [mV]	f [kHz]
AVR osc	20	0,1	110,1	49,21	37,23	-1,500	10,000
Agilent DSO-X 3014A	20	0,1	137	66	35,4	-3,43	10,050

Tab. 11: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=1V ; f=10kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [mV]	U_{MAX} [mV]	U_{RMS} [mV]	U_{AVG} [mV]	f [kHz]
AVR osc	20	0,2	895,3	445,3	324,9	-2,343	10,00
Agilent DSO-X 3014A	20	0,2	1,02	503	352	-8,36	10,037

Tab. 12: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=5V ; f=10kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [V]	U_{MAX} [V]	U_{RMS} [V]	U_{AVG} [mV]	f [kHz]
AVR osc	20	1,0	5,085	2,531	1,835	-13,12	10,00
Agilent DSO-X 3014A	20	1,0	5,11	2,51	1,77	-36,8	10,001

Tab. 13: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=0,1V ; f=500kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [mV]	U_{MAX} [mV]	U_{RMS} [mV]	U_{AVG} [mV]	f [kHz]
AVR osc	2	0,1	97,74	45,87	36,63	1,874	500,0
Agilent DSO-X 3014A	2	0,1	121	58	35,3	-3,18	497,0

Tab. 14: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=1V ; f=500kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [mV]	U_{MAX} [mV]	U_{RMS} [mV]	U_{AVG} [mV]	f [kHz]
AVR osc	2	0,2	918,7	449,9	329,2	5,624	500,0
Agilent DSO-X 3014A	2	0,2	1010	495	351	-8,36	500,0

Tab. 15: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=5V ; f=500kHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [V]	U_{MAX} [V]	U_{RMS} [V]	U_{AVG} [mV]	f [kHz]
AVR osc	2	1,0	4,312	2,156	1,635	-14,06	500
Agilent DSO-X 3014A	2	1,0	5,11	2,51	1,78	-39,3	500,3

Tab. 16: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=0,1V ; f=1MHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [mV]	U_{MAX} [mV]	U_{RMS} [mV]	U_{AVG} [mV]	f [MHz]
AVR osc	2	0,1	103,1	63,28	38,21	8,906	1,000
Agilent DSO-X 3014A	2	0,1	121	66	35,5	6,37	1,0320

Tab. 17: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=1V ; f=1MHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [mV]	U_{MAX} [mV]	U_{RMS} [mV]	U_{AVG} [mV]	f [MHz]
AVR osc	1	0,2	1010	449,9	321,2	-12,65	1,000
Agilent DSO-X 3014A	1	0,2	862,4	495	351	-8,82	1,0010

Tab. 18: Porovnání s komerčním přístrojem - podmínky měření: U_{PP}=5V ; f=1MHz

Přístroj	Časová základna [µs]	Rozsah [V/div]	U_{PP} [V]	U_{MAX} [V]	U_{RMS} [V]	U_{AVG} [mV]	f [MHz]
AVR osc	1	1,0	4,453	2,51	1,756	98,43	1,000
Agilent DSO-X 3014A	1	1,0	5,11	2,25	1,78	-39,3	1,0003

Vzhledem k tomu, že přístroj má v sobě mód, ve kterém se chová jako voltmetr, byly jeho vlastnosti také porovnány. Avšak voltmetr v tomto módu neměří jen střední hodnotu. Díky tomu, že si ukládá určité množství vzorků do paměti, tak v určitém časovém „okně“ vyhodnocuje maximální, minimální a průměrnou hodnotu napětí. Čím je toto okno větší, tím se více vzorků může uložit do paměti a tím více je měření přesné (především u průměrné hodnoty). Na druhou stranu má pak celé měření značnou hysterezi. Nejlépe je to vidět na průměrné hodnotě, kde při odpojení sondy od zdroje napětí začne hodnota pomalu klesat. Naměřené hodnoty jsou v tab. 19. Multimetr UT60A měl nastavené automatické nastavení rozsahu. Jako napájecí zdroj byl použit Diametral - P230R51D.

Tab. 19: Porovnání s komerčními přístroji - mód voltmetru

Přístroj	AVR osc		UT60A
Napětí na zdroji [V]	$U_{AVR\ osc}$ (Průměr) [V]	AVR osc - rozsah [V]	$U_{MULTIMETR}$ [V]
0,0	22,67 m	0,3	23,6 m
0,1	116,5 m	0,3	118,8 m
0,3	257,9 m	0,3	260 m
1,5	1,529	3	1,542
3,3	3,305	6	3,343
5,0	4,885	6	4,94
9,0	8,855	15	8,894
12,0	11,83	15	11,96

4.3 Desky plošných spojů

Asi nejnáročnější na návrhová pravidla byla deska s analogovými obvody, A/D převodníkem a bufferem. V tomto případě se ani nemusely tolik řešit signálové a výkonové země (spotřeba OZ je malá) jako oddělení analogové části od digitální.

Na levé polovině DPS jsou analogové obvody. Uprostřed je A/D převodník a pravá polovina patří digitálním obvodům. Tímto rozložením je zajištěna největší možná vzdálenost mezi analogovou a digitální částí a tím i minimalizace rušení z digitální části. Stínění na analogové části je řešeno vylitím zemí na obou stranách desky a použitím prokovů, které jsou blízko sebe. Vzhledem k tomu, že prokovy jsou relativně blízko sebe, tak zde není velká šance, aby dielektrikum mezi zemnícími plochami fungovalo jako vysokofrekvenční rezonátor nebo jako parazitní kapacita. Většina součástek je v provedení SMD. Je to kvůli tomu, že na vysokých frekvencích znamená dlouhý vodič parazitní indukčnost. Aby byla tato parazitní indukčnost co nejmenší, jsou použita SMD pouzdra, u kterých je délka vývodů podstatně kratší než u DIP pouzder. Zároveň se celý obvod zmenší. Na druhou stranu výměna SMD součástek je podstatně náročnější, než je tomu u „klasických“ součástek. U digitálních obvodů je stínění provedeno digitální zemí, ale vzhledem k tomu, že digitální obvody jsou mnohem odolnější vůči rušení, nejsou prokovy umístěny tak blízko u sebe jako u analogové části. Výstup z oscilátoru je co nejkratší cestou přiveden jednak k A/D převodníku a jednak k samotnému bufferu.

Co se týče složitosti návrhu, následuje zdroj společně s obvodem pro správu klávesnice a tlačítek. Tyto dva obvody jsou na jedné desce, protože jsou relativně jednoduché a nečekají se nějaké zásadní změny ohledně náročnosti na parametry zdroje nebo počtu tlačítek na klávesnici.

Jednotka pro správu klávesnice a tlačítek je v podstatě jen samotný µC, ke kterému jsou připojeny výstupy od tlačítek a klávesnice. Zbytek pak tvoří zdroj. Konektory pro napájení jsou záměrně blízko u sebe, aby uživatel nemusel „dlouho“ hledat napájecí konektory. Výkonové součástky zabírají relativně velký podíl na desce, avšak jejich předimenzované parametry mají svůj důvod. Zejména je to kvůli vývoji. Nízké napětí tyto součástky nezničí a proud je omezen buďto v USB (0,5 A), nebo v externím zdroji (za předpokladu, že je použit laboratorní zdroj). U nábojové pumpy (invertor napětí) je kaskáda kondenzátorů. Jejich hodnoty jsou od největší kapacity po nejmenší (směrem k zátěži). Je to z důvodu ESR. Elektrolytické kondenzátory mají sice velkou kapacitu, ale velký ESR. To znamená, že jejich stabilizační účinky jsou omezeny právě ESR. Naproti tomu keramické kondenzátory mají malé ESR, že o něm nemá smysl uvažovat. Na druhou stranu mají malou kapacitu. Nejmenší kapacita je „vepředu“ z důvodu špičkových, ale krátkých

odběrů proudů zátěží. V tomto případě je každá cesta vlastně odpor se sériovou indukčností. Keramický kondenzátor dokáže svůj náboj předat zátěži mnohem rychleji než elektrolytický kondenzátor, protože jeho ESR je prakticky nula a „je blíž“. Tím se dosáhne toho, že výstupní napětí se změní o minimum a zátěži je dodán potřebný proud.

Poslední je DPS pro hlavní řídící jednotku. Přestože je SW nejkomplikovanější a má na starost v podstatě všechny bloky, její deska je jednoduchá, protože zpracovává pouze digitální signál. Deska obsahuje jen minimum externích součástek (většinou ochranné rezistory). Je zde i místo pro externí krystal, pokud by nestačil interní 8 MHz oscilátor a byl by zapotřebí vyšší výpočetní výkon. Operační zesilovač je zde kvůli módu, ve kterém se přístroj chová jako voltmetr. Byl vybrán nízkošumový OZ NE5532. Právě v tomto případě je zapotřebí, aby se šum z Atmega32 nedostal do analogových obvodů. A jedna z možných cest vede právě přes tento OZ. Proto byl zvolen nízkošumový OZ a bylo ověřeno, že například oproti TL082 se hladina šumu skutečně snížila.

5 Závěr

Principiálně je osciloskop již plně funkční, ale jeho vývoj nebyl jednoduchý. Například, pokud byl jako impedanční oddělovač použit OZ AD8039, tak jeho vstupní proud byl natolik velký, že nebylo možné s ním realizovat vstupní odpor 1 MΩ a větší. Proto byl místo AD8039 použit OZ AD8066, který využívá technologii MOS-FET, takže jeho vstupní proud je v řádech pA. Na druhou stranu tato technologie je citlivá na statický náboj. Dále bylo potřeba změnit ochranné diody na vstupu. Prvně byly použity schottkyho diody 1N5819. Jejich problém byl skrytý v parazitní kapacitě v závěrném směru. Ta totiž byla mnohonásobně větší, než byla vstupní kapacita OZ a tím značně degradovala celý vstupní obvod. Proto byly místo 1N5819 použity 1N5711, které mají tuto parazitní kapacitu mnohem menší. Na druhou stranu je tato dobrá vlastnost vykoupena nižším závěrným napětím, a tím se snížila i odolnost proti vysokým napětím. Ani automatické přepínaní MOS-FET tranzistorů uvnitř zdroje se neobešlo bez komplikací. Co se týče vývoje SW, tak lze prohlásit, že se jedná o velice náročný úkol. Program je psán v jazyku symbolických adres, protože tím získáme maximální výkon. Rychlost je však vykoupena menší pohodlností při programování. Programátor si musí „vše hlídat sám“. Tzn. že si musí „definovat proměnné“ a hlídat si, aby jejich obsah nebyl například omylem přepisován. V podstatě musí vymýšlet všechny rozhodovací podmínky na bitové úrovni. V jazyce C je například toto rozhodování (větší, menší, rovno) mnohem jednodušší a intuitivnější.

Naměřené hodnoty na osciloskopu řízeným AVR se lišily od naměřených přístrojem Agilent DSO-X 3014A. Je to hned z několika důvodů. Jednak přístroj Agilent DSO-X 3014A je v jiné cenové kategorii (přibližně 65000 Kč [18]) a tedy větší přesnost je očekávána. Osciloskop řízený AVR byl napájen z USB a to přes neoriginální, univerzální adaptér. V příloze je zachycena obrazovka, na které je vidět šum, který bylo možné naměřit mezi zemí a +5 V na USB portu při napájení z tohoto adaptéru. Bylo vyzkoušeno, že pokud bylo napájení z USB provedeno přes tento adaptér, pak byl šum na generovaném signálu mnohem větší, než když byl notebook napájen na baterii. S větším rozsahem rostla i absolutní chyba. To je tím, že na větší rozsah je dán stejný počet bitů. To znamená, že pokud je chyba například jeden LSB, pak na rozsahu 0,1V/div je jeho hodnota přibližně 2,35 mV. Ale pokud je zvolen rozsah 5V/div, už je chyba jednoho LSB 117,6 mV. Zároveň s rostoucí frekvencí klesala přesnost. V tomto případě se začínají projevovat parazitní vlastnosti použitých součástek a návrh layoutu desky. Při měření na rozsahu 0,1V/div, frekvenci 1 Mhz a amplitudě 50 mV periodické vzorkování selhávalo (zobrazení průběh neodpovídal skutečnosti). Proto bylo měření provedeno pro časovou základnu 2 µs, kde je vzorkování realtime.

V módu voltmetru se projevila zejména kvantizační chyba a na malých rozsazích i šum. Pro orientační měření je přístroj vyhovující. Měření bylo provedeno s napájením přes univerzální adaptér.

Pokud by byl použit kvalitnější napájecí zdroj, měl by přístroj vykazovat menší chybu měření. Záměrně bylo provedeno měření pro nejhorší možný případ.

Stále je zde spousta věcí, o které je možné přístroj vylepšit. A to především v SW části. Výhodou je, že programovací piny jsou snadno dostupné, takže přeprogramovat cílový µC je snadné a rychlé. Program v hlavní řídící jednotce zabírá přibližně ⅔ flash paměti, takže je zde stále relativně spousta prostoru pro rozšíření.

Na internetu existuje několik projektů, kde je osciloskop řízen AVR. Asi nejzdařilejší projekt je eOscope [15]. Autor uvádí, že přístroj dokáže zobrazit (bez aliasingu) signál o frekvenci 10 MHz. Rozlišení displeje je 240 x 128 px, takže výsledný signál je i dobře zřetelný. Na druhou stranu má pouze jeden pracovní rozsah a to 40 mV/dílek, dále velmi malý vstupní odpor: pouze 10 kΩ, což je pro řadu měření velký problém, protože se cílový obvod příliš zatíží. Díky PLD je sice celé zařízení

velice rychlé, ale jeho spotřeba je velká. Autor udává, že zařízení potřebuje k chodu napájení 8~10 V a 1 A. Trigger je digitální i manuální.

Další povedený projekt je Low speed AVR oscilloscope [16]. Jeho předností je úžasná jednoduchost. Celé zařízení obsluhuje pouze ATmega32. Okolních součástek je skutečně minimum. Vstupní odpor 1 MΩ je pro většinu případů vyhovující. Trigger je automatický. Napájení je 12 V. Rozlišení je menší než u předchozího projektu: 128 x 64 px. Nová verze SW již měří stejnosměrnou a střídavou složku vstupního napětí a dále měří i frekvenci. Jeho slabinou je nízký frekvenční rozsah. Autor uvádí, že je možné měřit od 10 Hz do 7,7 kHz. Je to dáno především tím, že jako A/D převodník je použit interní převodník v ATmega32.

Cílem této práce bylo sestrojit osciloskop, který by byl relativně rychlý a zároveň dokázal měřit základní veličiny. Respektive získat rychlost jako je tomu v prvním případě a zároveň si zachovat nízkou spotřebu a rozšířit program o užitečné funkce. Teoreticky maximálním zobrazená frekvence bez aliasingu při použití periodického vzorkování je 10 MHz (2 vzorky na periodu). Při realtime vzorkování pak jen 2,5 MHz. Prakticky použitelný rozsah je od 0,02 Hz do 1MHz. Vstupní odpor 1 MΩ, spotřeba při napájení z USB se pohybuje okolo 140 mA, trigger je zatím analogový.

Seznam použité literatury

[1] Atmel AVR 8- and 32-bit Microcontrollers [online], [citace 14.11.2011]. Dostupné na www: http://www.atmel.com/products/avr/default.asp?category_id=163&family_id=607&source=global_nav

[2] Atmel AVR XMEGA Technical Details [online], [citace 14.11.2011]. Dostupné na www: http://www.atmel.com/products/avr/xmega.asp?category_id=163&family_id=607&subfamily_id=196

[3] ATTINY13-20PU [online], [citace 14.11.2011]. Dostupné na www: http://www.gme.cz/mikroprocesory-atmel-avr-tiny/attiny13-20pu-p432-206/

[4] AVR 40 Pin 16MHz 32K 8A/D – ATMega32 [online], [citace 14.11.2011]. Dostupné na www: http://www.sparkfun.com/products/209

[5] ATMEGA128-16AU [online], [citace 14.11.2011]. Dostupné na www: http://www.chinaicmart.com/series-ATM/ATMEGA128-16AU.html

[6] ATmega32 [online], [citace 14.11.2011]. Dostupné na www: http://www.atmel.com/dyn/products/product_card.asp?part_id=2014

[7] Atmega32-16PU [online], [citace 14.11.2011]. Dostupné na www: http://www.gme.cz/mikroprocesory-atmel-avr-mega/atmega32-16pu-p432-180/

[8] ATmega88 [online], [citace 15.11.2011]. Dostupné na www: http://www.atmel.com/dyn/products/product_parameters.asp?category_id=163&family_id=607&subfamily_id=760&part_id=3302&ListAllAttributes=1

[9] AD9057 datasheet, REV D, [online], [citace 15.11.2011]. Dostupné na www: http://www.analog.com/static/imported-files/data_sheets/AD9057.pdf

[10] AD9057, [online], [citace 15.11.2011]. Dostupné na www: http://www.ebay.com/sch/i.html?_from=R40&_trksid=p5197.m570.l1313&_nkw=ad9057&_sacat=See-All-Categories

[11] Atmega8L-8PU [online], [citace 22.11.2011]. Dostupné na www: http://www.gme.cz/mikroprocesory-atmel-avr-mega/atmega8l-8pu-p432-193/

[12] IRF520 [online], [citace 29.11.2011]. Dostupné na www: http://www.gme.cz/unipolarni-tranzistory-n-kanal/irf520-p213-027/

[13] AM1S-0512SZ [online], [citace 29.11.2011]. Dostupné na www: http://www.gme.cz/spinane-regulatory-napeti-dc-dc-menice-pevne/am1s-0512sz-p332-222/

[14] DEM 128064B SYH-PY, Ver. 4.1.1, [online], [citace 29.11.2011]. Dostupné na www: http://www.display-elektronik.de/DEM128064BSYH-PY.pdf

[15] eOscope, Ver. 1.2, [online], [citace 12.8.2011]. Dostupné na www: http://www.eosystems.ro/eoscope/eoscope_en.htm

[16] SERASIDIS, Vassilis: Low speed AVR oscilloscope, Ver. 2.0, [online], [citace 12.8.2011]. Dostupné na www: http://www.serasidis.gr/circuits/AVR_oscilloscope/avr_oscilloscope.htm

[17] Tektronix Forum – View topic – Measuring frequency with digital oscilloscope [online], [citace 8.5.2012]. Dostupné na www: http://www1.tek.com/forum/viewtopic.php?f=5&t=37

[18] DSOX3014A Oscilloscope: 100MHz, 4 channels | Agilent [online], [citace 12.5.2012]. Dostupné na www: http://www.home.agilent.com/agilent/product.jspx?nid=-33573.970761.00&lc=ger&cc=DE

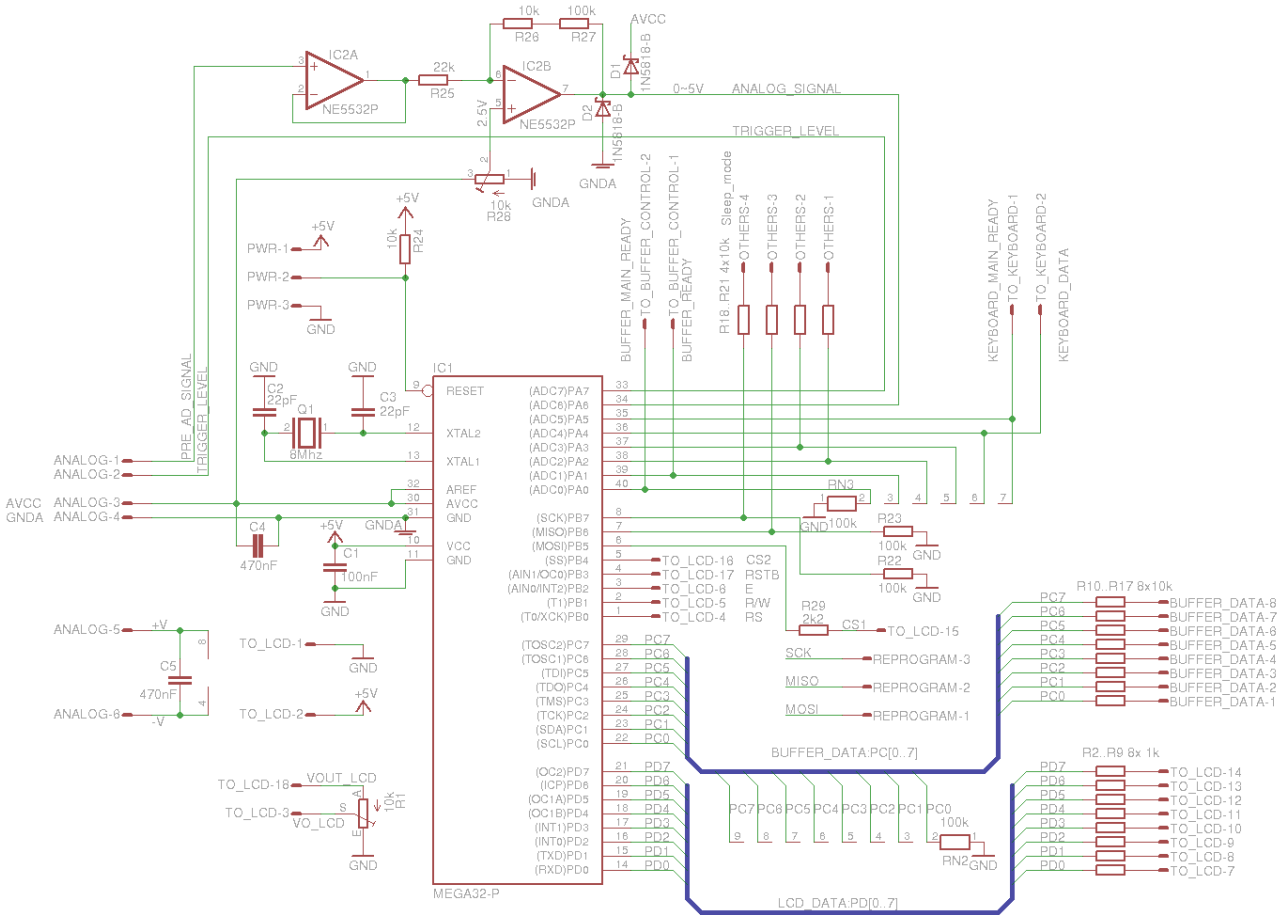
Abecední seznam zkratek

μC	Mikrokontrolér
A/D	Analog na digitál (ve smyslu převod)
AC	Střídavá veličina (napětí, proud)
DC	Stejnosměrná veličina (napětí, proud)
DC/DC	Ve smyslu měnič stejnosměrné veličiny na obvykle jinou hodnotu té samé veličiny
DPS	Deska plošných spojů
ESR	Ekvivalentní sériový odpor
LCD	Displej z tekutých krystalů
MIPS/MHz	Počet provedených instrukcí vzhledem k taktu
MSPS	Miliony vzorků za vteřinu
OZ	Operační zesilovač
S/N	Poměr signálu ku šumu

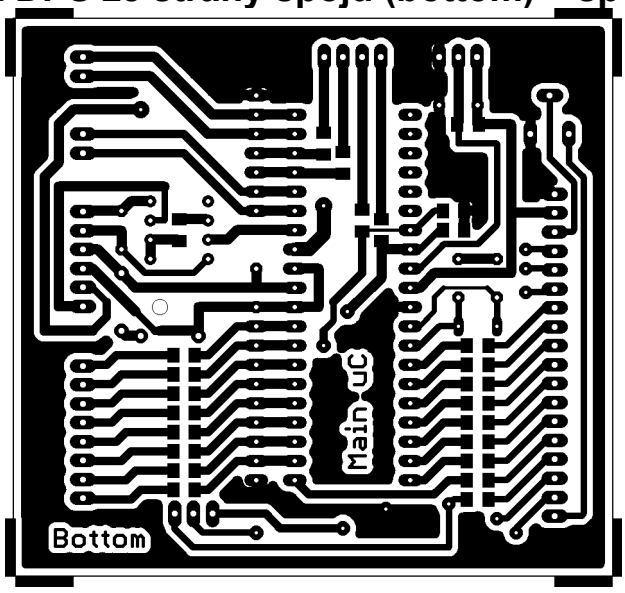
Seznam příloh

A Návrh hlavní řídící jednotky

A.1 Schéma

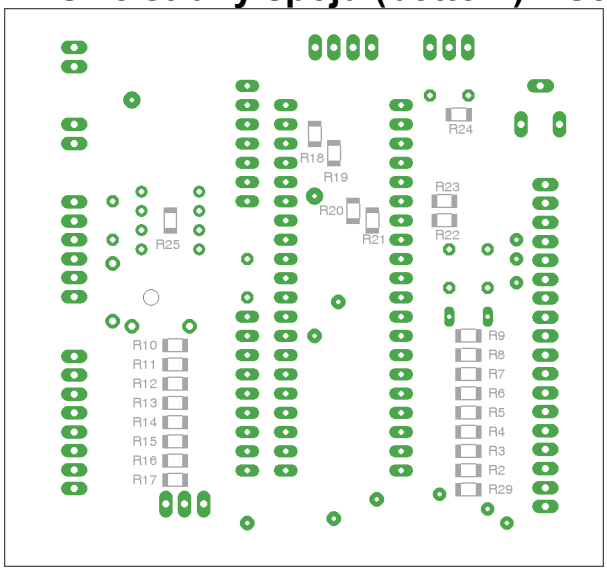


A.2 Pohled na DPS ze strany spojů (bottom) – spoje



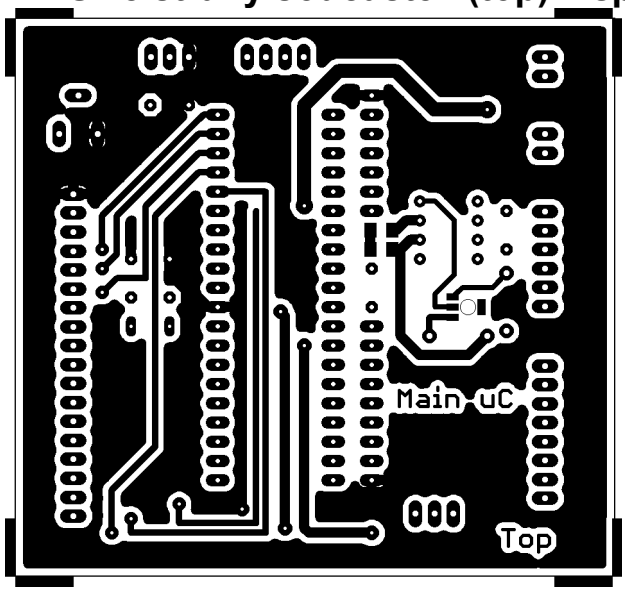
Měřítko: 1:1

A.3 Pohled na DPS ze strany spojů (bottom) – součástky



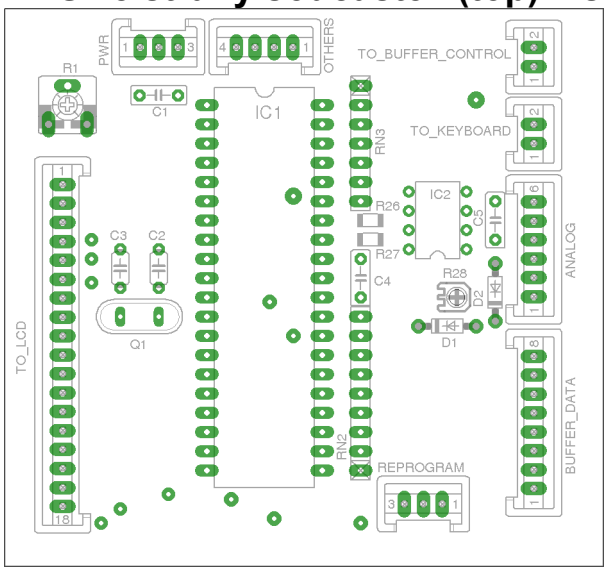
*Měřítko: 1:1

A.4 Pohled na DPS ze strany součástek (top) – spoje



Měřítko: 1:1

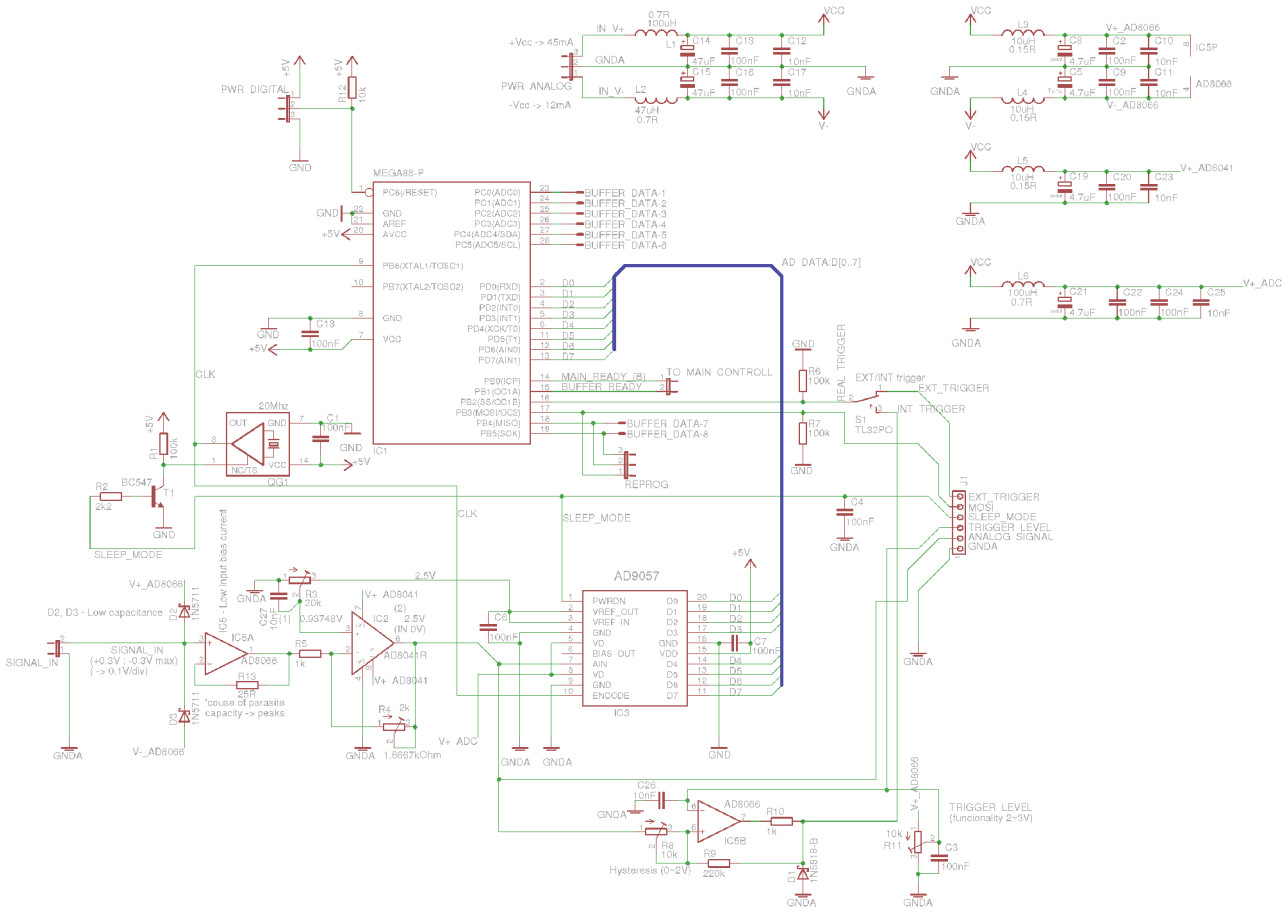
A.5 Pohled na DPS ze strany součástek (top) – součástky



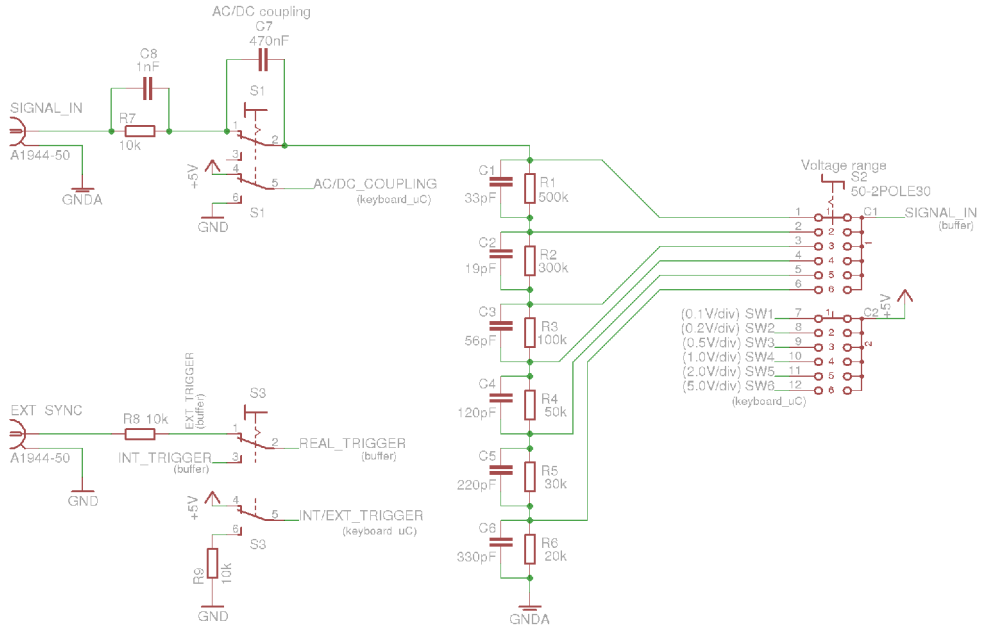
Měřítko: 1:1

B Návrh bufferu

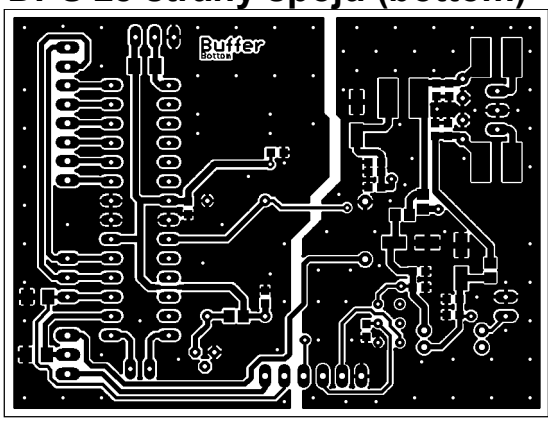
B.1 Schéma



B.2 Schéma vstupních obvodů

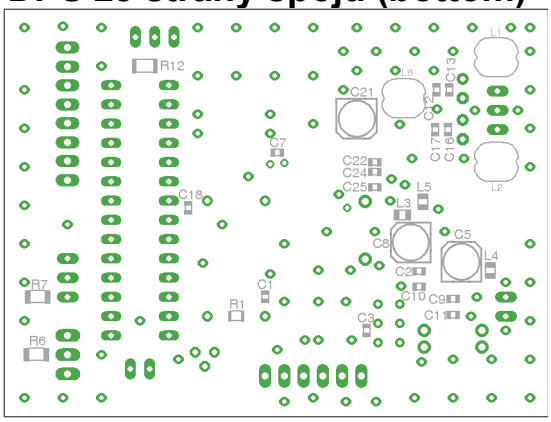


B.3 Pohled na DPS ze strany spojů (bottom) – spoje



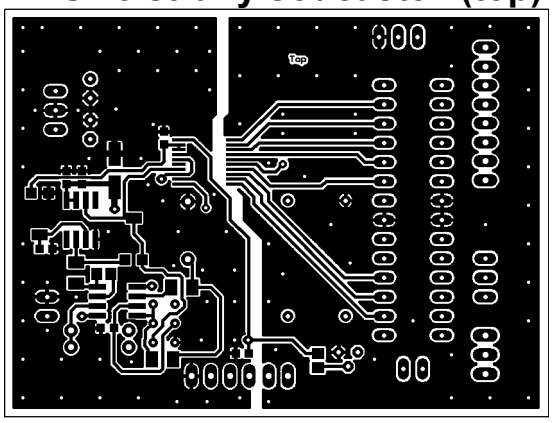
Měřítko: 1:1

B.4 Pohled na DPS ze strany spojů (bottom) – součástky



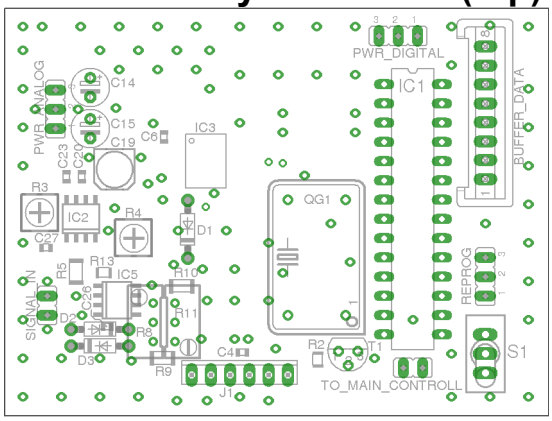
Měřítko: 1:1

B.5 Pohled na DPS ze strany součástek (top) – spoje



Měřítko: 1:1

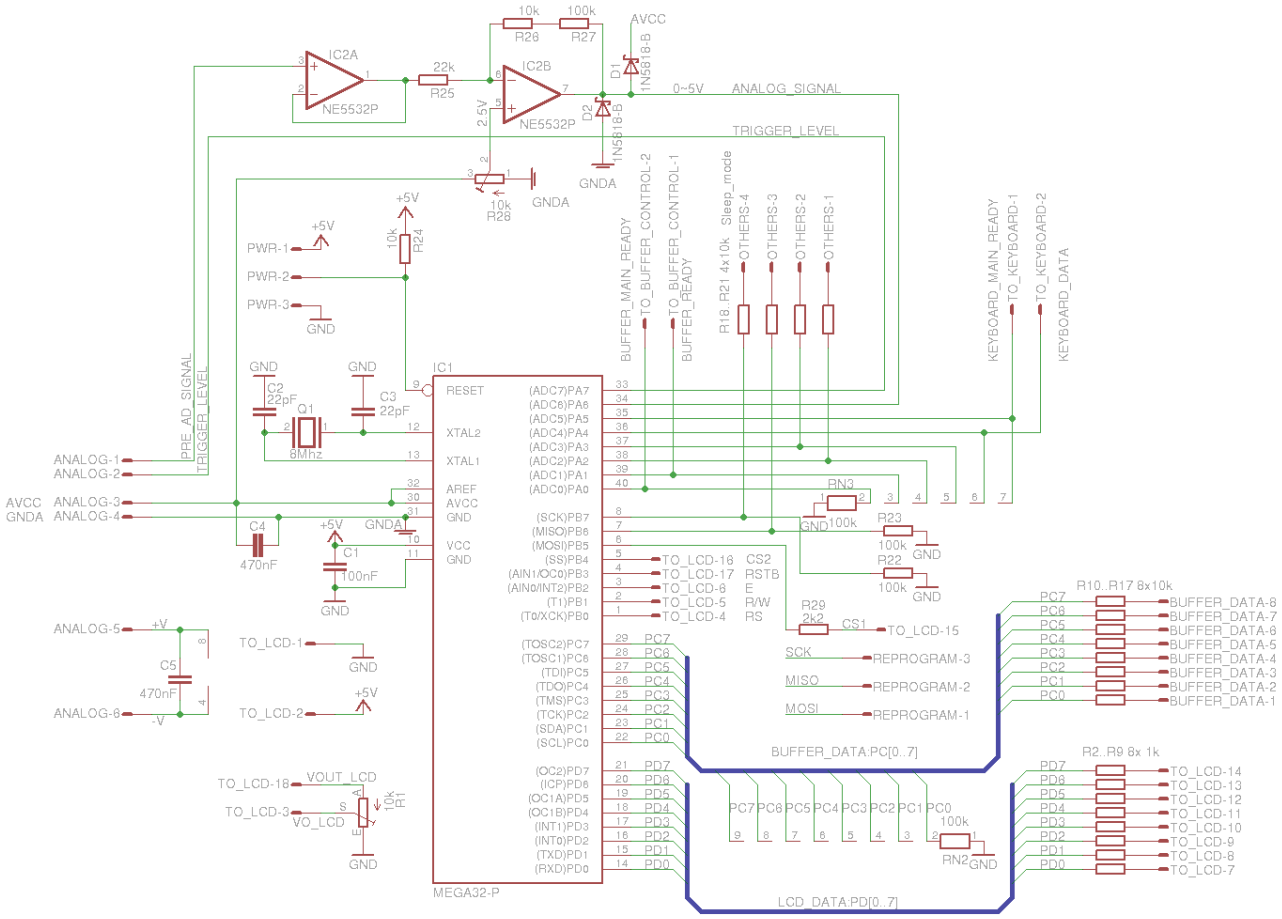
B.6 Pohled na DPS ze strany součástek (top) – součástky



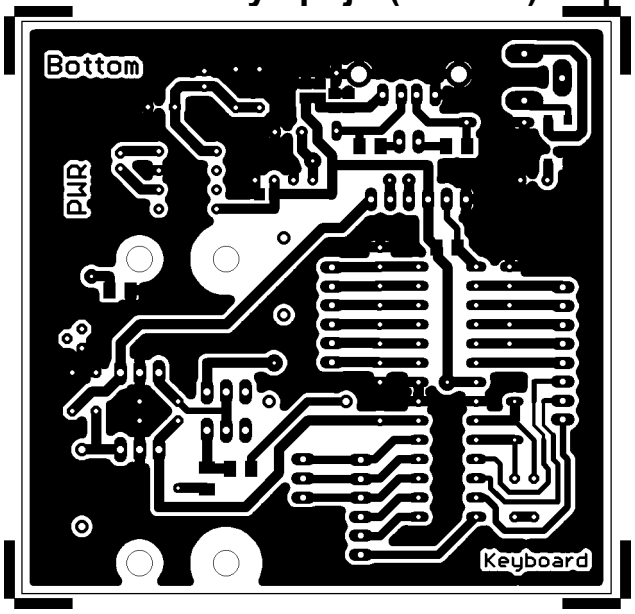
Měřítko: 1:1

C Návrh jednotky pro správu klávesnice a zdroj

C.1 Schéma

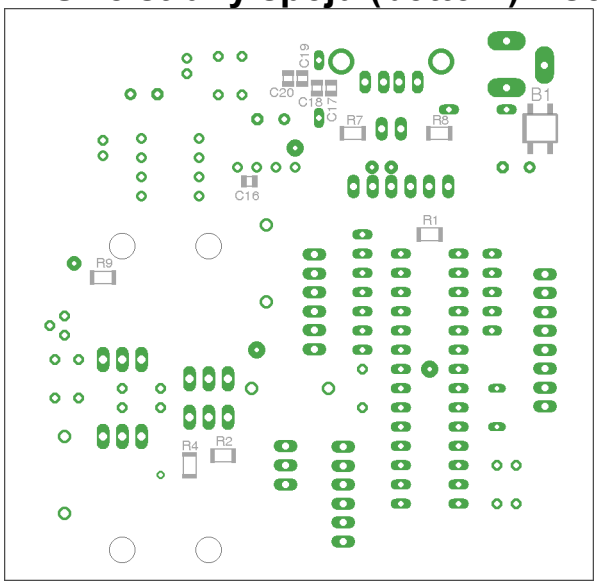


C.2 Pohled na DPS ze strany spojů (bottom) – spoje



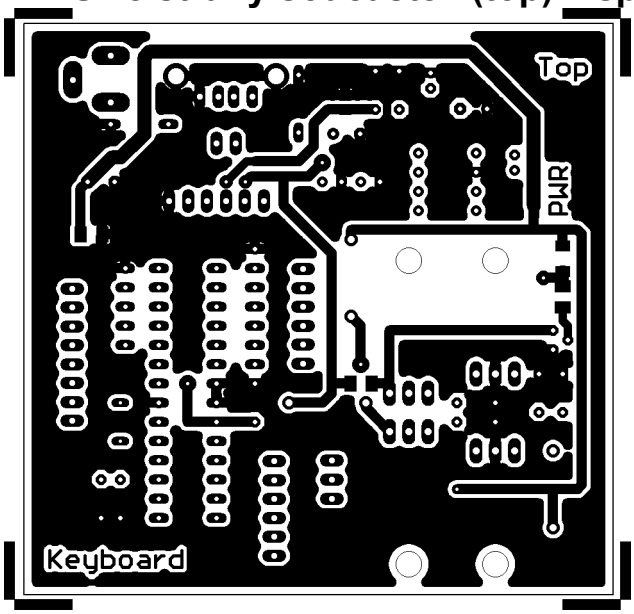
Měřítko: 1:1

C.3 Pohled na DPS ze strany spojů (bottom) – součástky



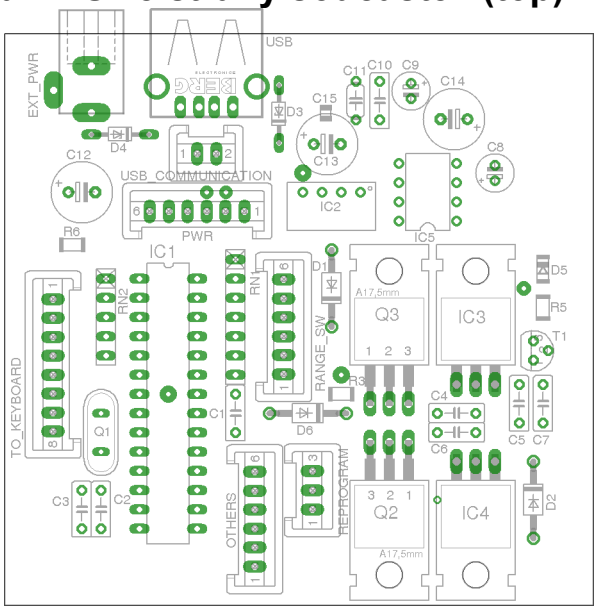
Měřítko: 1:1

C.4 Pohled na DPS ze strany součástek (top) – spoje



Měřítko: 1:1

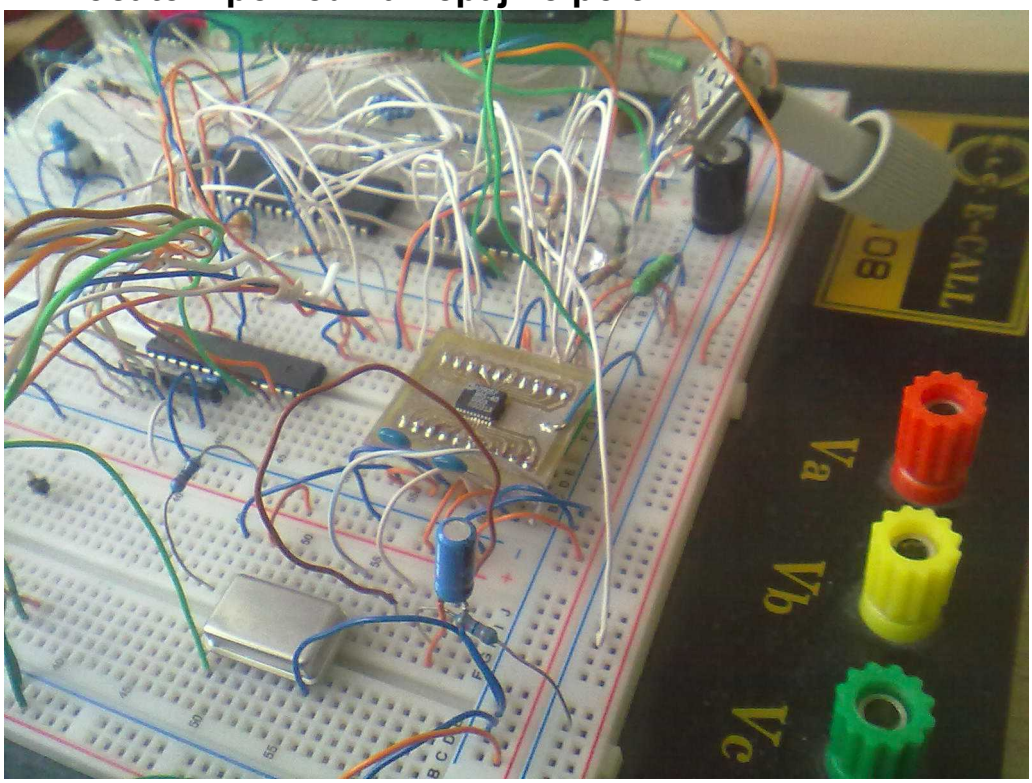
C.5 Pohled na DPS ze strany součástek (top) – součástky



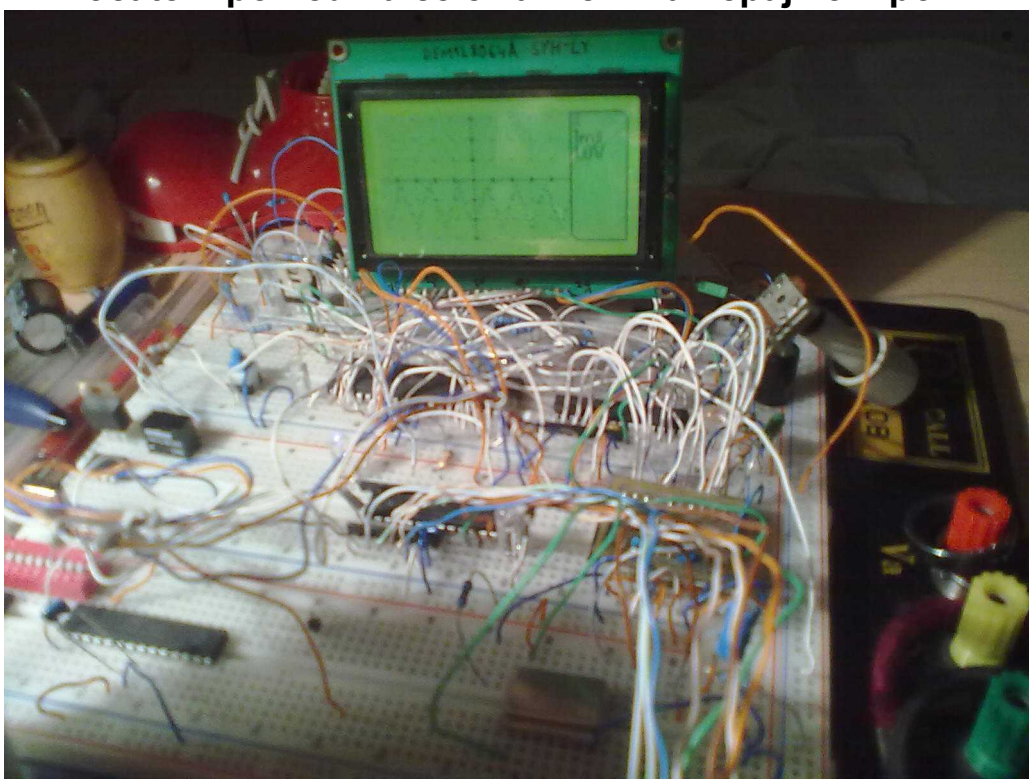
Měřítko: 1:1

D Fotografie z vývoje

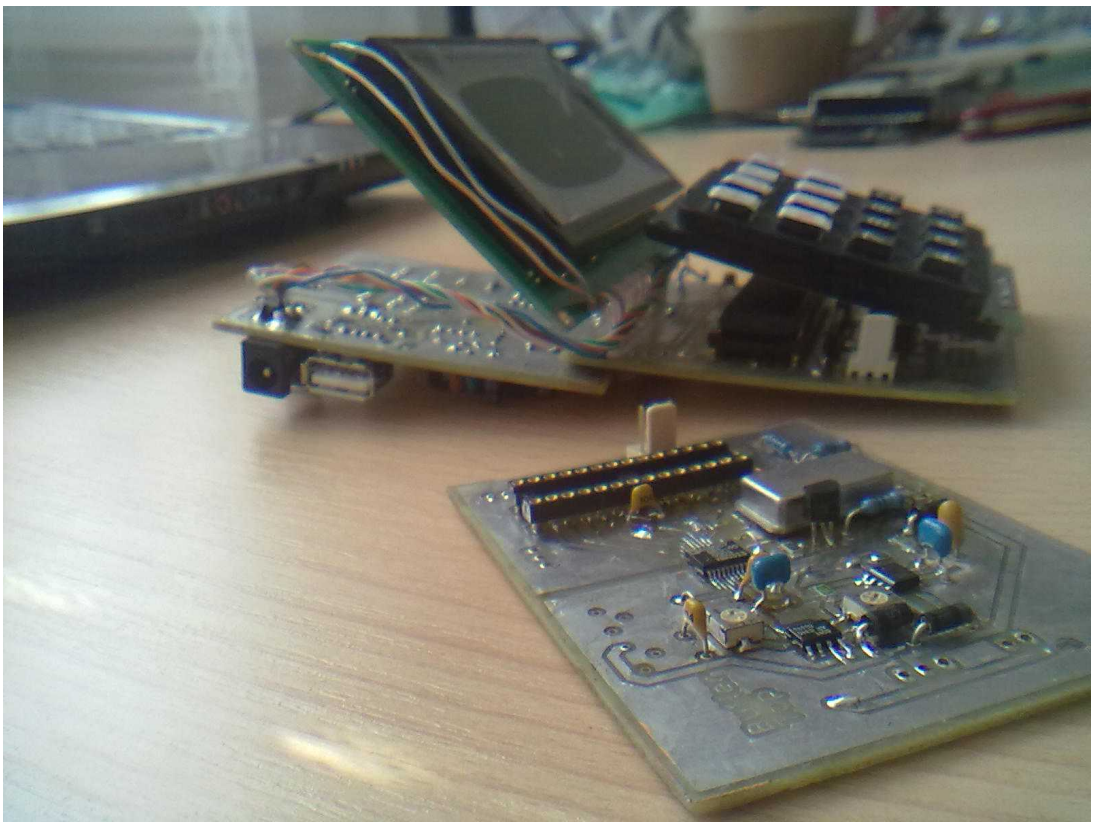
D.1 Počátek: pohled na nepájivé pole



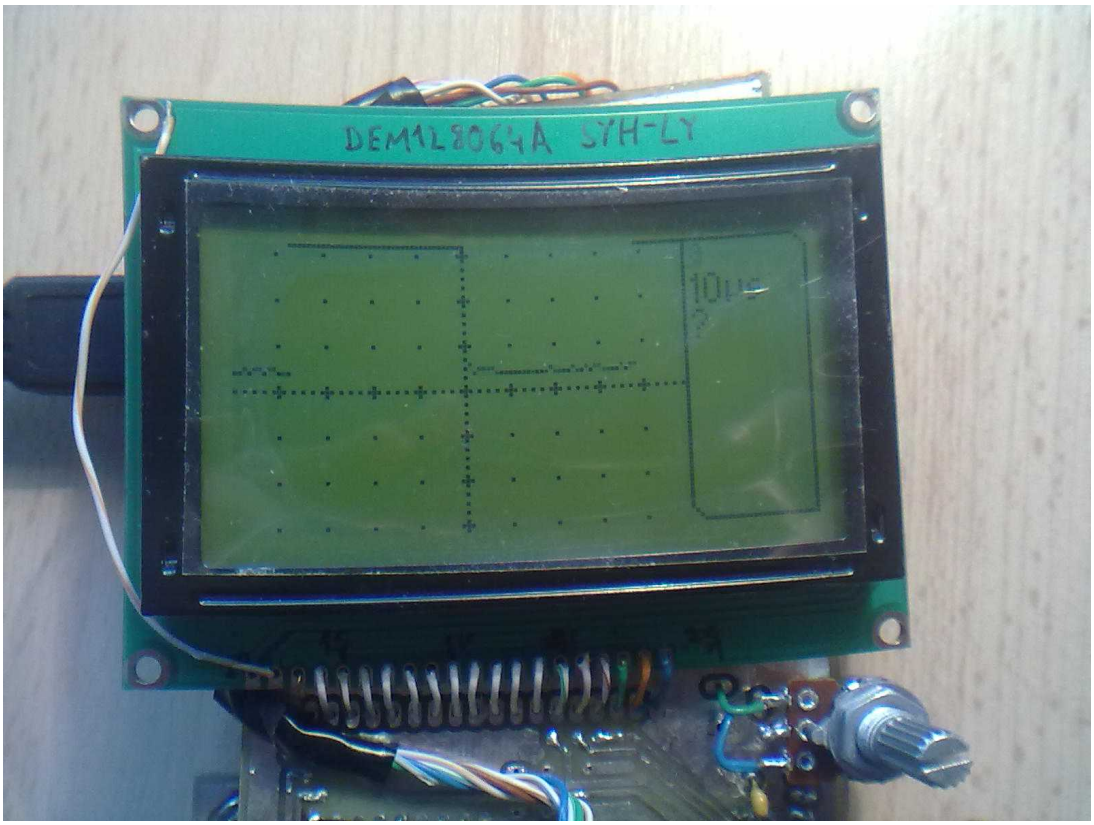
D.2 Počátek: pohled na celé zařízení na nepájivém poli



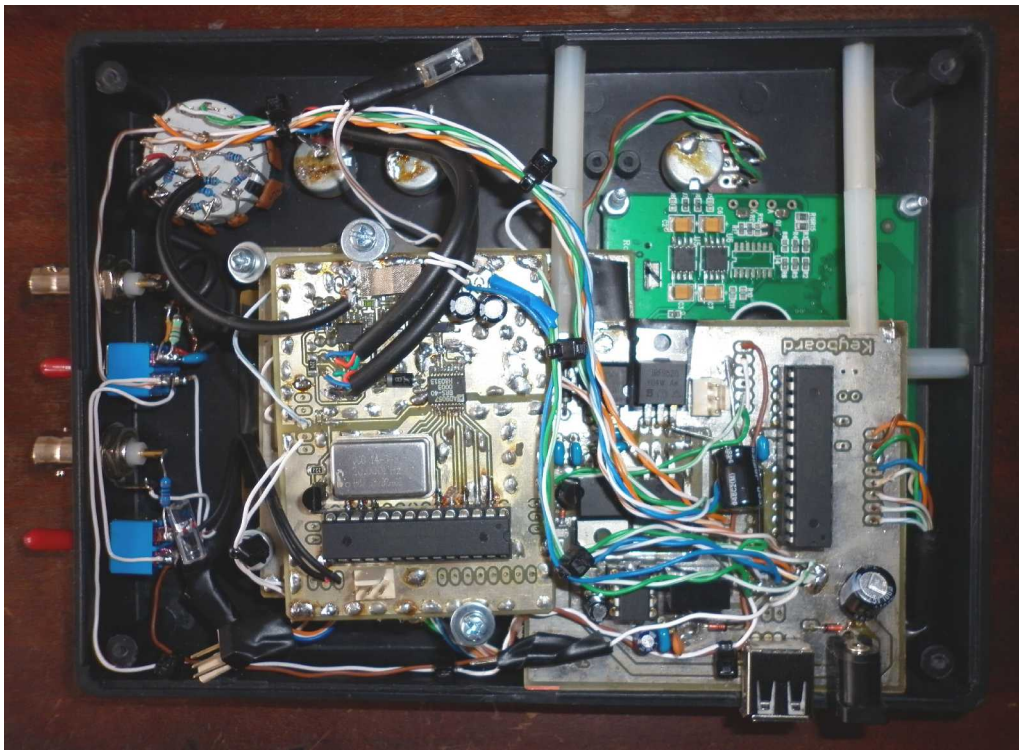
D.3 Osazené DPS při sestavování přístroje – pohled z boku



D.4 Zkouška funkčnosti



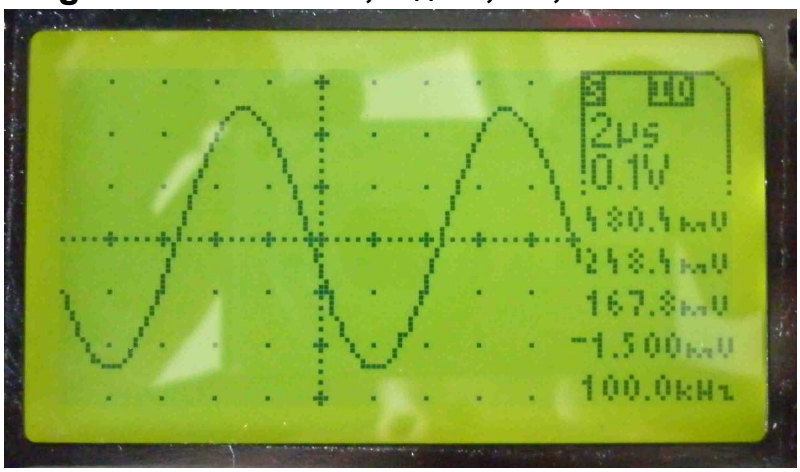
D.5 Pohled do zařízení v krabičce



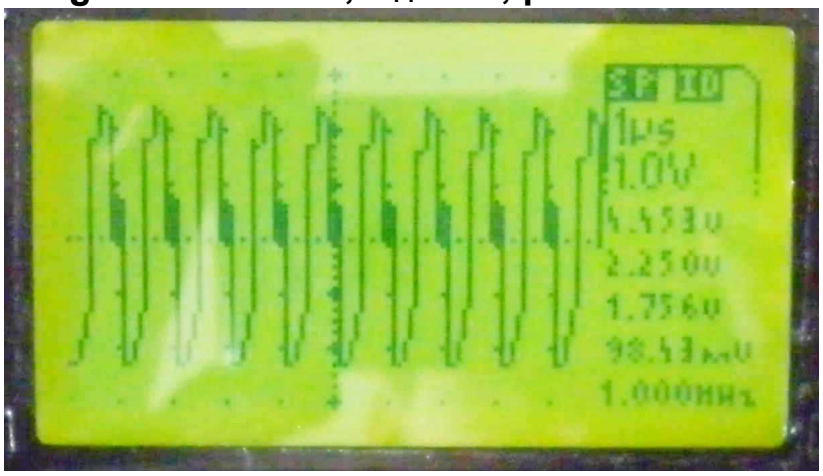
D.6 Starší verze krabičky

D.7 Nová verze krabičky

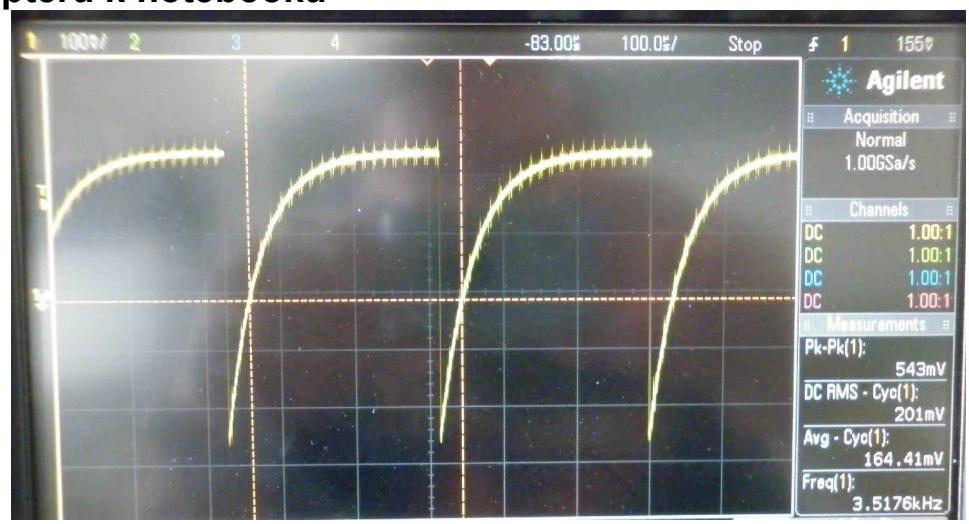
D.8 Oscilogram – f=100kHz ; U_{PP}=0,5V ; realtime vzorkování



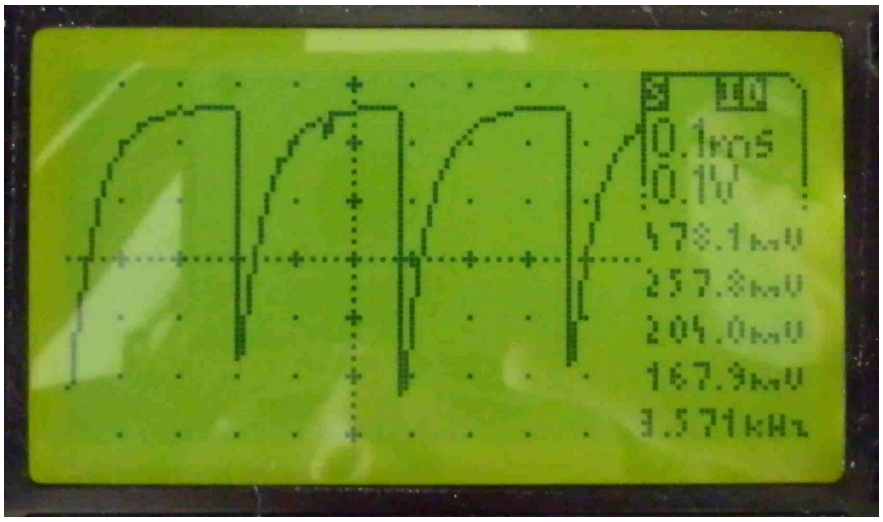
D.9 Oscilogram – f=1MHz ; U_{PP}=5V ; periodické vzorkování



D.10 Signál z generátoru na Agilent DSO-X 3014A po připojení adaptéru k notebooku



D.11 Zobrazení signálu na osciloskopu řízeným AVR



D.12 Signál z generátoru na Agilent DSO-X 3014A při bateriovém provozu notebooku

D.13 Šum na USB po připojení univerzálního adaptéru

D.14 Ukázka módu ve kterém zařízení pracuje zařízení jako voltmetr