1 Úvod

Tato práce navazuje na literaturu [1], kde jsem v rámci Semestrálních projektů 1 a 2 navrhnul a otestoval podobné zařízení za pomocí třech časovačů 555. Požadavky na realizované zařízení vychází ze zdroje [2], kde se pojednává obecně o magnetronovém naprašování a řízení tohoto procesu. Výhodou obvodového řešení s časovači 555 byla jednoduchost, generátor obsahoval pět SSI integrovaných obvodů, několik tranzistorů, diod a pasivních prvků. Cena kompletního zařízení tj. součástky a plošný spoj byla do 400 Kč. Další výhodou byla vysoká pracovní frekvence, omezená maximální frekvencí časovačů, až 1,5 MHz. Na druhé straně zde bylo několik nevýhod. Samotné zařízení se neobešlo bez dvoukanálového osciloskopu, sloužícího jako zobrazovač nastavených parametrů. To bylo daní za vysoký rozsah hodnot, kdy nebylo možné pouze ocejchovat stupnice nastavovacích prvků. S tím souvisí i samotné rozsahy, jejichž nastavení nebylo exaktní, protože výsledné hodnoty frekvence a střídy se navzájem ovlivňovaly a byly určeny polohou čtyř potenciometrů. Zjednodušeně řečeno, požadované parametry bylo nutné vždy chvíli hledat.

Cílem této diplomové práce je navrhnout a sestrojit zařízení, které splní požadované parametry a zároveň se vyhne zmiňovaným nedostatkům předchozího zařízení. Prvotní úvahou je použití mikrokontroleru doplněného o inteligentní zobrazovač. Rozbor konkrétního řešení provedu v následující kapitole. Pochopitelně návrh i výsledné zařízení bude složitější a dražší, ale půjde o autonomní celek a odpadne nutnost použití osciloskopu, jednotlivé parametry se budou nastavovat tlačítky na alfanumerickém displeji. Navíc, kromě elektrické části je nutné vytvořit i program pro mikrokontroler. To je ovšem dnešní běžná praxe na poli moderní elektroniky, kdy se hojně používají programovatelné integrované obvody. V následující kapitole rozeberu řešení s ohledem na jeho realizovatelnost a vhodnost pro magnetronové naprašování.

2 Řešení problému

Dle zadání jde o generátor se dvěma výstupy S_1, S_2. Výstupní signálem je napětí s obdélníkovým průběhem s nastavitelnou střídou a frekvencí v intervalu 10 kHz až 50 kHz. Přičemž signál bude generován v protitaktu, tj. bude rozdělen na dvě pracovní oblasti, jedna bude odpovídat výstupu S_1 a druhá výstupu S_2. Tedy, poběží-li první výstup, druhý bude blokován a naopak. V každé pracovní oblasti bude možné nastavit počet impulzů. V zásadě se jedná o stejný úkol, jako je popsán v [1], lišit se bude realizací. Maximální požadovaná frekvence 50 kHz je nižší, než u zařízení popsaného v [1], ale to je dáno fyzikální podstatou magnetronového naprašování. U tohoto technologického postupu se používá relativně nízkých frekvencí do asi 200 kHz. Od vymezených požadavků můžeme nyní přejít k řešení vlastního problému.

První úvahy směřovaly ke konceptu řešit celý úkol pouze s mikrokontrolerem. Pomocí jeho prostředků a programu realizovat ovládání, zobrazování nastavených parametrů i samotný generátor, čítač impulzů a výstupní logiku. Základní myšlenkou bylo vytvořit cyklus odpovídající časovému normálu a jeho násobením snižovat výslednou frekvenci. Pokud by časové úseky byly dostatečně krátké vzhledem k pracovní frekvenci, tak by se zároveň vyřešilo i nastavení střídy. Uvažujme, že perioda je daná například sumou deseti referenčních časů, potom při střídě 50% máme pět časových normálů v aktivní části periody a pět časových normálu v pasivní části periody. Při střídě, například, 30% ku 70% jsou to tři smyčky v aktivní části periody a sedm smyček v pasivní části periody. Takto by byla generována výsledná frekvence i plnění v celém požadovaném intervalu. Bohužel, proti této úvaze hovoří tři fakty. Prvním je vztah mezi frekvencí a periodou resp. časem, ten je dán převrácenou hodnotou. Protože se jedná o lomenou funkci, vztah mezi nimi není lineární a pro celé hodnoty času budu dostávat racionální hodnoty frekvence, viz tab. 1. S tímto faktem se můžeme elegantně vypořádat, jednoduše bychom nastavovali místo frekvence periodu, ta má v podstatě stejnou vypovídací hodnotu. Z tabulky je patrný i další problém, který není tak snadno řešitelný a je jím minimální krok. Vytvořím-li, pomocí zpožďovací smyčky normál jedna mikrosekunda obdržíme sice pro jednu smyčku značně vysokou frekvenci 1 MHz, ale ta jejich sčítáním příliš rychle klesá. Opět by bylo možné udělat určitý kompromis a spokojit se s intervalem hodnot daným krokem jedna mikrosekunda, představující méně hodnot v horní části intervalu a více v dolní. Uvažujeme-li, pouze o nastavení frekvence bylo by toto řešení přijatelné, nikoliv při požadavku na nastavení střídy, pro kterou je nutné skládat aktivní a pasivní část periody z velmi malých přírůstků. Druhá část tabulky tab. 1 poukazuje na třetí problém, kdy požadované parametry naráží na možnosti mikrokontrolerů. V tab. 1 je vidět, že pro krok 100 Hz musím čas nastavovat v řádu desítek až stovek nanosekund. Například, pro změnu frekvence ze 49,9 kHz na hodnotu 50,0 kHz je časový rozdíl pouze 40 ns a to je hodnota kritická, protože je v rozporu s pracovní frekvencí mikrokontroleru. Čas 40ns odpovídá pracovní frekvenci 25 MHz.

Tab. 1: Vztah mezi frekvencí a periodou

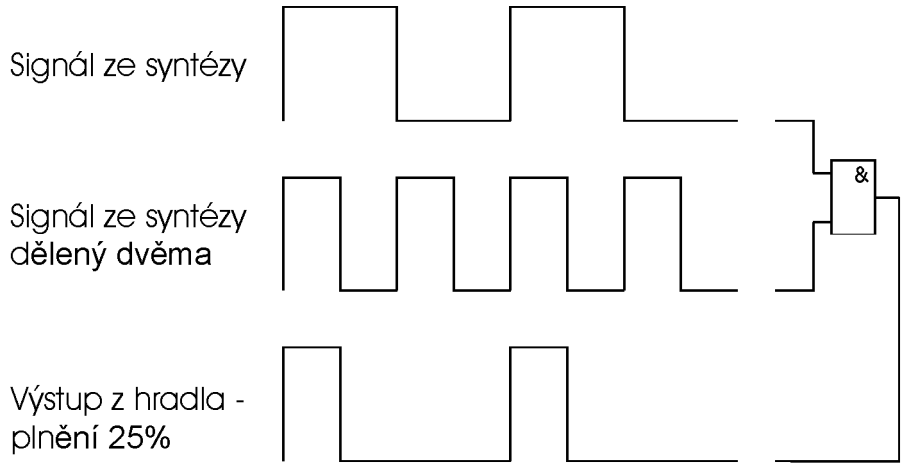
perioda [µs]	frekvence [kHz]	frekvence [kHz]	perioda [µs]
1	1000,00	50	20,000
2	500,00	49,9	20,040
3	333,33	49,8	20,080
4	250,00	49,7	20,121
5	200,00	49,6	20,161
6	166,67	49,5	20,202
7	142,86	49,4	20,243
8	125,00	49,3	20,284
9	111,11	49,2	20,325
10	100,00	49,1	20,367
11	90,91	49	20,408
12	83,33	48,9	20,450
13	76,92	48,8	20,492
14	71,43	48,7	20,534
15	66,67	48,6	20,576
16	62,50	48,5	20,619
17	58,82	48,4	20,661
18	55,56	48,3	20,704
19	52,63	48,2	20,747
20	50,00	48,1	20,790
21	47,62	48	20,833
22	45,45	47,9	20,877
23	43,48	47,8	20,921
24	41,67	47,7	20,964
25	40,00	47,6	21,008

Běžné mikrokontrolery firmy Microchip mají frekvenci oscilátoru 20 či 40 MHz, ty rychlejší až 80 MHz. Ovšem je nutné si uvědomit, že je rozdílný vztah mezi frekvencí oscilátoru a taktovací frekvencí[1], daný poměrem 4:1. Taktovací frekvence je čtyřikrát nižší. Pro takt 25 MHz potřebujeme mikrokontroler s frekvencí oscilátoru 100 MHz! Do této úvahy opět nezahrnujeme nastavení střídy, pro niž by v horní části intervalu byly třeba ještě kratší časové úseky. Dále je nutné uvážit problematiku časové náročnosti programu. Kdy musíme vždy pečlivě spočítat počet instrukcí tvořící zpožďovací smyčku, či časově kritickou část programu. Praktické řešení je použití buď krystalu se specifickou frekvencí, jehož hodnota spolu s nastavením děličky dají frekvenci požadovanou. Případně akceptujeme určitou chybu a vydělením taktovací frekvence získáme hodnotu přibližnou. U vyššího programovacího jazyka navíc dopředu nevíme, jak bude kód přeložen do assembleru tj. kolik instrukcí smyčka či podprogram vykoná. Shrnutím všech předchozích úvah je jasné, že použití mikrokontroleru samotného, coby nástroje pro generování časů je dosti problematické, obzvlášť je-li nutné generovat určitý interval s celými diskrétními hodnotami.

Výše zmíněné důvody poukazují na nutnost hledat jiné řešení. V běžném životě nás obklopuje obrovské množství různých elektrických zařízení, jejichž společnou vlastností je

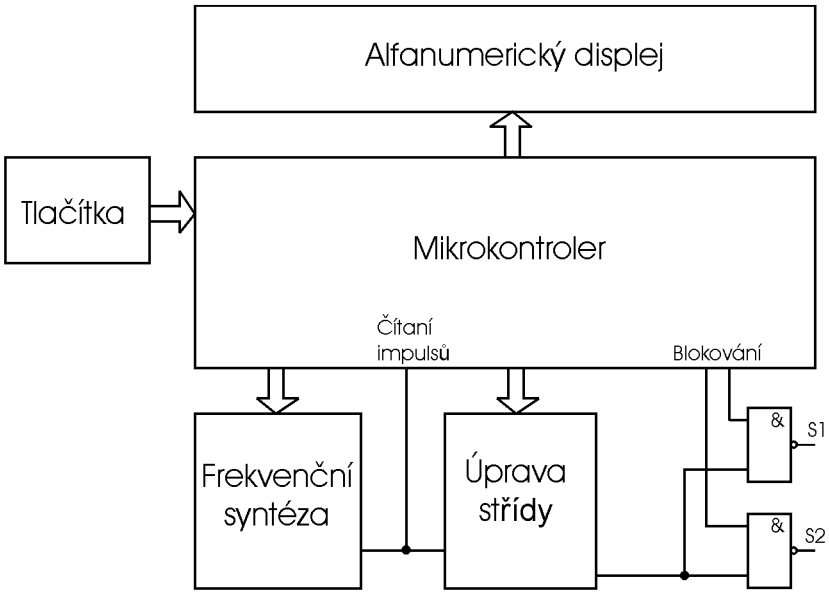
[1] Platí pouze pro mikrokontrolery firmy Microchip.

nutnost nastavení frekvence v určitém rozsahu. Mám na mysli celou škálu radiových zařízení – radiopřijímače, televizory, satelitní přijímače ap. U všech těchto zařízení je ladění, tj. volba požadované frekvence řešena stejně, na principu kmitočtové syntézy. Pochopitelně, i tyto zařízení obsahují mikrokontrolery, jejich funkce je ale jiná, zabezpečují ovládání a jsou zde implementovány ostatní funkce nutné pro chod zařízení. Domnívám se, že tento koncept je vhodný i pro vyvíjené laboratorní zařízení. Pomocí periferních obvodů by se vytvořila kmitočtová syntéza. O její řízení by se staral mikrokontroler, zároveň by zabezpečil obsluhu displeje, čítač impulzů a pomocí jednoduchého hradlového obvodu by z jednoho signálu vytvořil dva protitaktní. Kmitočtová syntéza se běžně realizuje pomocí fázového závěsu, jehož teoretický rozbor provedeme v následující kapitole.



Obr. 1: Princip úpravy střídy

Generátor je nutné doplnit o obvody pro úpravu střídy. Jednoduchým řešením je vzít signál z kmitočtové syntézy a přivést ho do děličky a následně původní signál s novým (poděleným) signálem přivést do hradla AND, viz obrázek 1. Dělící poměr bude potom přímo úměrný střídě. Detailně se na tuto část problému zaměříme v dalších kapitolách. Principiální schéma generátoru s kmitočtovou syntézou je na obrázku 2. Frekvenční syntéza se zdá být vhodným řešením, o správnosti nás přesvědčuje i její použití řadou konstruktérů u jiných úloh. Teorii i praktickému řešení kmitočtové syntézy s použitím fázového závěsu se bude věnovat následující kapitola.



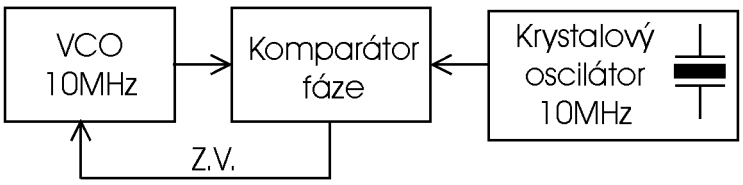
Obr. 2: Principiální zapojení

3 Frekvenční syntéza – teorie a praxe

3.1 Teorie frekvenční syntézy s použitím fázového závěsu

Systémy založené na frekvenční synteze s fázovým závěsem jsou nejpoužívanější metodou generování vysokofrekvenčních oscilací ve všech možných typech zařízení. Asi bychom nenalezli přijímač či vysílač, který by neobsahoval alespoň jeden obvod s fázovým závěsem. V anglicky psané literatuře je označován zkratkou PLL – Phase Lock Loop a má mnohostranné využití v moderní elektronice. Mezi nejčastější aplikace patří AM, FM demodulace, FSK modulace, frekvenční syntézy a frekvenční synchronizace. Oblastí našeho zájmu je frekvenční syntéza. Definice frekvenční syntézy říká, že jde o obvodové řešení založené na zpětné vazbě, kdy je ze vstupní (zdrojové) frekvence odvozena výstupní frekvence a vztah mezi nimi je přímý, či nepřímý.

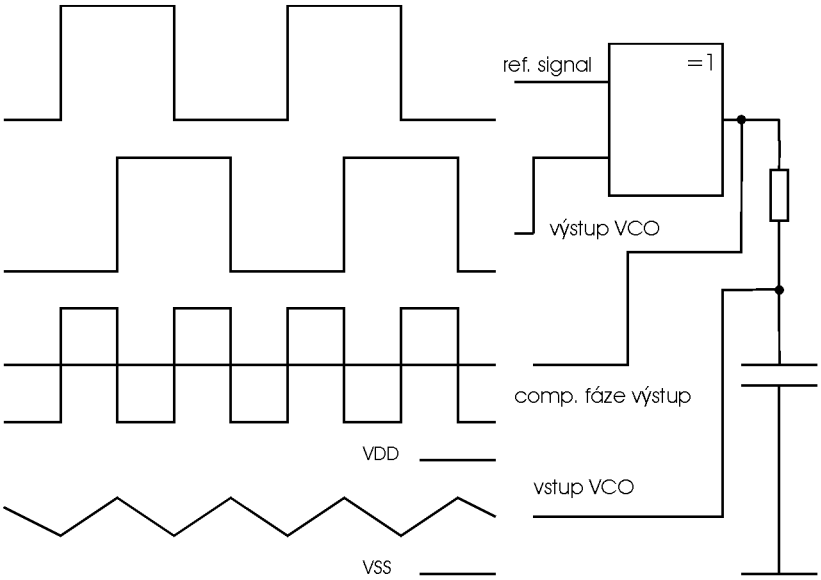
Pokud bychom chtěli generovat přesné kmity o určité frekvenci, je nejjednodušší možností použít krystalový oscilátor. Frekvence jeho kmitů je velmi stabilní, nevýhodou je pouze jediná konkrétní frekvence ekvivalentní např. jednomu kanálu. Pro více frekvencí (více kanálů) je nutné použít více krystalů, což není přístup ani praktický, ani ekonomický. Na druhé straně existuje tzv. oscilátor řízený napětím (VCO – Voltage Controlled Oscillator). Frekvence napětím řízeného oscilátoru je ovládána vstupním napětím. V rezonančním obvodu se jím ovlivňuje prvek udávající rezonanční frekvenci. V diskrétní podobě je to kapacita varikapu a v integrované proud proudového zrcadla. Nevýhodou je teplotní stabilita a závislost na napájecím napětí, negativně ovlivňující stabilitu celého obvodu a stabilitu výstupní frekvence. Máme tedy krystalový oscilátor s velmi přesnou, nepřeladitelnou frekvencí a napětím řízený oscilátor, přeladitelný a nestabilní. Snahou konstruktérů bylo hledat obvodové řešení obsahující dobré vlastnosti obou zmíněných. Výsledkem je zapojení s oběma prvky – krystalovým oscilátorem a napětím řízeným oscilátorem uvedeném na obrázku 3. Jejich frekvence jsou porovnávány v bloku označeném fázový komparátor.



Obr. 3: Základní princip PLL

Fázový komparátor je obvod se dvěma vstupy a jedním výstupem. Jsou-li vstupní frekvence i fáze shodné není na výstupu žádné napětí. V případě rozdílných frekvencí a fází je na výstupu napětí úměrné jejich rozdílu. Na pozici fázového komparátoru lze použít např.

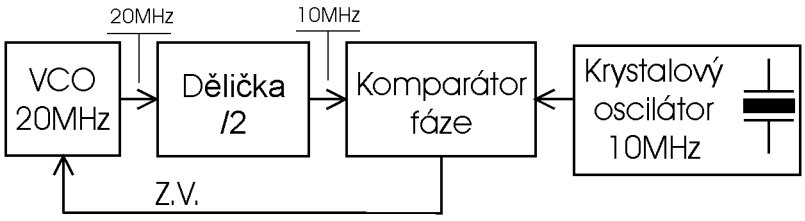
XOR hradlo, případně složitější obvod. Funkce fázového komparátoru z hlediska napěťových průběhů je uvedena na obrázku 4.



Obr. 4: Funkce fázového komparátoru a filtru [8]

Výstupní (chybové) napětí fázového komparátoru můžeme nyní přivést na vstup napětím řízeného oscilátoru. Tím dojde k tzv. zavěšení VCO na frekvenci krystalového oscilátoru. Zpětnovazební smyčkou se automaticky kompenzují rozdíly vzniklé změnou teploty a napětí. Popsané zapojení je nejjednodušší formou fázového závěsu, jež má dobrou stabilitu, ale přišlo o variabilitu tj. přeladitelnost. Pochopitelně, nejde o to dělat to samé složitěji. Úvahy od začátku počítají i s přeladitelností obvodu, jeho rozšíření nebylo doposud zmíněno kvůli snazší orientaci v problému.

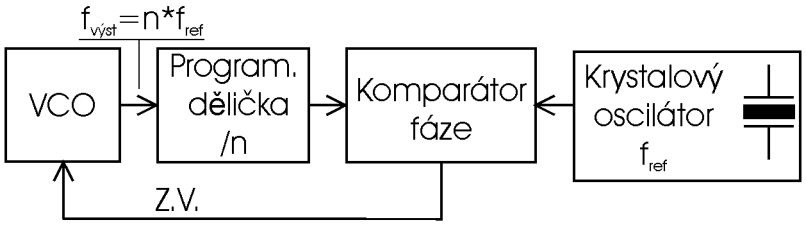
Kdybychom chtěli mít na výstupu frekvenci např. dvojnásobnou je nutné na ni nějakým způsobem přeladit VCO, což je zdánlivě nemožné. Zpětnovazební smyčka frekvenci okamžitě stáhne na referenční hodnotu (zde 10 MHz). Jinou, než referenční frekvenci lze nastavit za pomoci menšího triku.



Obr. 5: Princip generování jiné než základní frekvence

Blokem generujícím chybové napětí pro řízení VCO je komparátor fáze, na něj se musíme zaměřit a přesvědčit jej, že VCO běží na referenční frekvenci i když je tomu jinak.

Na obrázku 5 vidíme překvapivě jednoduché východisko. Frekvenci VCO nastavíme na 20 MHz a zařadíme před něj děličku dvěma, její výstup přivedeme na vstup fázového komparátoru. Pro něj je frekvence stále stejná (10 MHz). Stejným způsobem můžeme dělit libovolným číslem a generovat požadované frekvence, stabilita je přitom stále výborná, adekvátní krystalovému oscilátoru. Dělit jedním číslem je nedostatečné, minimálně zde můžeme zapojit čítač. Počet čítaných impulzů poté určuje výslednou frekvenci. Výhodnější je čítač nahradit programovatelnou děličkou jako je tomu na obrázku 6 a získat větší rozsah hodnot. Prozatím jsem se nezmínil o výstupu respektive odkud jej vyvést. Po krátkém zamyšlení je zřejmé, že je jím výstup VCO, generující kmity v rozsahu daném součinem referenční frekvence krystalového oscilátoru a čísla n tj. dělícího poměru programovatelné děličky. Výstup z VCO je vhodný pro naši aplikaci, pro jiné aplikace se využívá i výstupu fázového komparátoru. Jednotlivé bloky v obvodu existují přímo v praktické podobě, jako integrované obvody. Fázový závěs v integrované podobě obsahuje většinou VCO i fázový komparátor. Stačí jej doplnit o krystalový oscilátor, programovatelnou děličku (zcela běžně dostupná v pouzdře jednoho integrovaného obvodu) a několik pasivních prvků a máme kompletní řešení kmitočtové syntézy.



Obr. 6: Princip generování více jiných frekvencí

Pasivními prvky je myšlen, doposud nezmíněný, zpětnovazební filtr zapojený mezi komparátor fáze a VCO. Hodnoty součástek jsou příliš vysoké a nelze je integrovat na čipu, navíc závistí na účelu použití a rozsahu frekvencí. Účelem filtru je převést střídavý signál na stejnosměrný[2], kterým řídíme VCO. Nejjednodušší variantou je RC filtr – integrátor. Integrováním signálu dostaneme průměrnou hodnotu napětí, odpovídající rozdílu vstupních signálů. Průběhy signálů jsou dobře vidět na obrázku 4 kde je fázový komparátor i s RC filtrem.

Předchozí odstavce jsou malým a stručným úvodem do problematiky fázového závěsu, který je základem pro pochopení další kapitoly zabývající se konkrétním návrhem kmitočtové syntézy s fázovým závěsem.

[2] Případně i střídavý, ale pomalu se měnící.

3.2 Praktické řešení kmitočtové syntézy s použitím fázového závěsu

Po obvodové stránce je v dnešní době poměrně snadné vytvořit kmitočtovou syntézu s širokým rozsahem. K realizaci stačí dva až tři integrované obvody a několik málo pasivních prvků. Z předchozí kapitoly víme, že potřebujeme přesný oscilátor, obvod fázového závěsu a programovatelnou děličku. Můj návrh je postaven na třech integrovaných obvodech CMOS řady 4000. Jednotlivými obvody jsou, oscilátor s čtrnáctistupňovým čítačem 4060, obvod fázového závěsu 4046 integrující VCO i fázový komparátor a programovatelná dělička 4059 s rozsahem 3 – 15999. Zmiňované integrované obvody popíši v následujícím odstavci. Vyústěním kapitoly 3 bude návrh zapojením kmitočtové syntézy s uvedenými integrovanými obvody, jeho realizace, oživení a ověření funkce.

3.2.1 Oscilátor s čtrnáctistupňovým čítačem 4060

Integrovaný obvod 4060 je jednou z možností, jak sestavit oscilátor s krystalem. Místo něj by šlo použít i některé základní hradlo (např. 4011) a z něj sestavit oscilátor a tvarovač. Obvod 4060 jsem použil, protože má malé pouzdro DIL 16 a velmi nízkou cenu. Kromě samotného oscilátoru lze využít i čítač a pomocí něj generovat více různých frekvencí s použitím jednoho krystalu. Z vnitřního schématu uvedeného v katalogovém listě je patrná funkce. První částí je oscilátor vytvořený ze třech hradel, jeho výstupem je vývod 9, frekvence měřitelná zde je shodná s rezonanční frekvencí krystalu. Za ním následuje čtrnáctistupňový čítač sestavený ze stejného počtu klopných obvodů JK. Kvůli limitům pouzdra DIL 16 je vyvedeno pouze deset výstupů. První následuje za čtvrtým klopným obvodem a odpovídá mu vývod 7. Frekvence na tomto vývodu je šestnáctkrát nižší oproti frekvenci oscilátoru. Ostatní klopné obvody již mají vyvedeny výstupy, kromě jedenáctého. Analogicky frekvence za každým klopným obvodem je dvakrát nižší než u předchozího. Vnitřní schéma, rozmístění vývodů a ostatní parametry viz katalogový list.

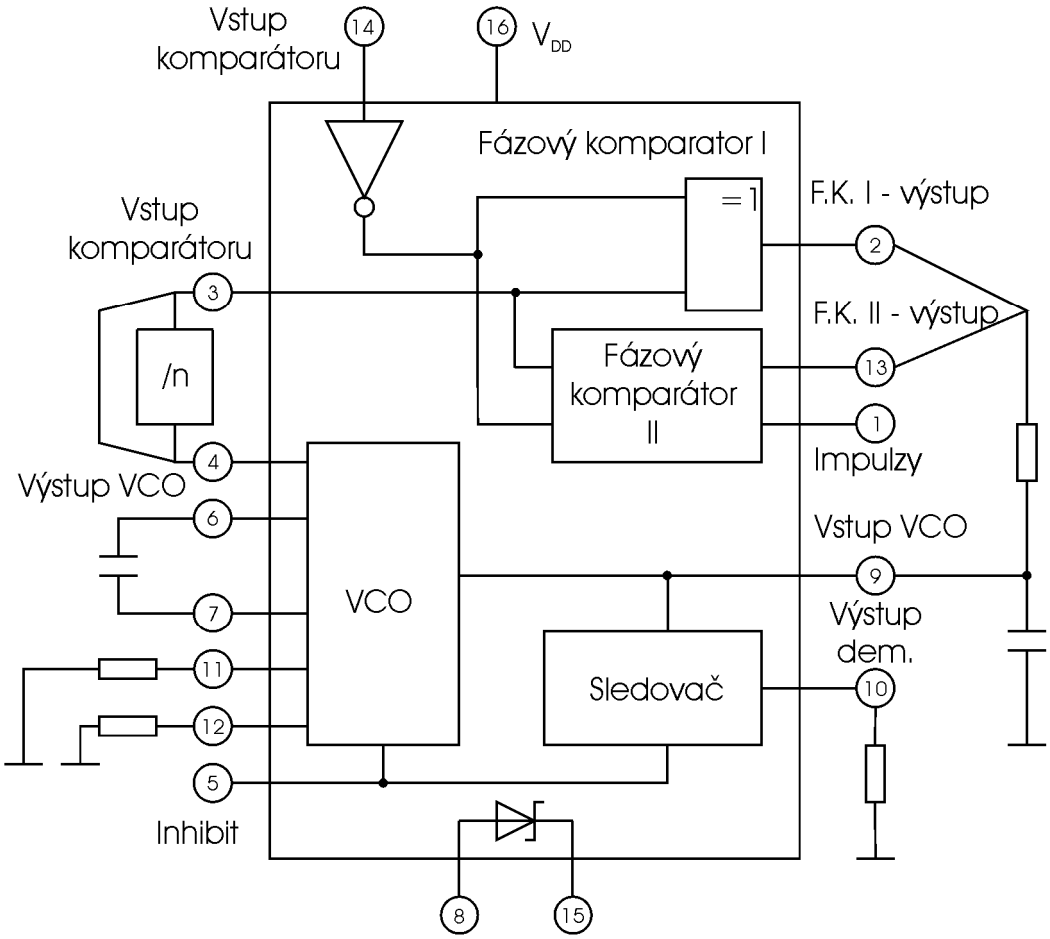
3.2.2 Obvod fázového závěsu 4046

Dalším integrovaným obvodem je fázový závěs 4046. Jde o jeden z nejznámějších běžně používaných fázových závěsů v integrované podobě. Na obrázku 7 je blokové schéma obvodu. Uvnitř je nízkopříkonový lineární VCO, dva fázové komparátory různých typů, zenerova dioda použitelná pro regulační účely a zesilovač se společným sourcem. Za výstupem fázového komparátoru je zapojen zpětnovazební filtr,[3] napětí z něj je snímáno

[3] Není součástí integrovaného obvodu.

pomocí sledovače s MOS tranzistorem. Zesilovač se společným sourcem zabraňuje zatěžování filtru, které by ovlivnilo jeho funkci. Výstup VCO může být zapojen do vstupu fázového komparátoru přímo nebo přes děličku. Vlastní integrovaný obvod fázového závěsu je nutné doplnit o několik externích pasivních prvků. Jejich hodnoty jsou závislé na použití a frekvenčním rozsahu.

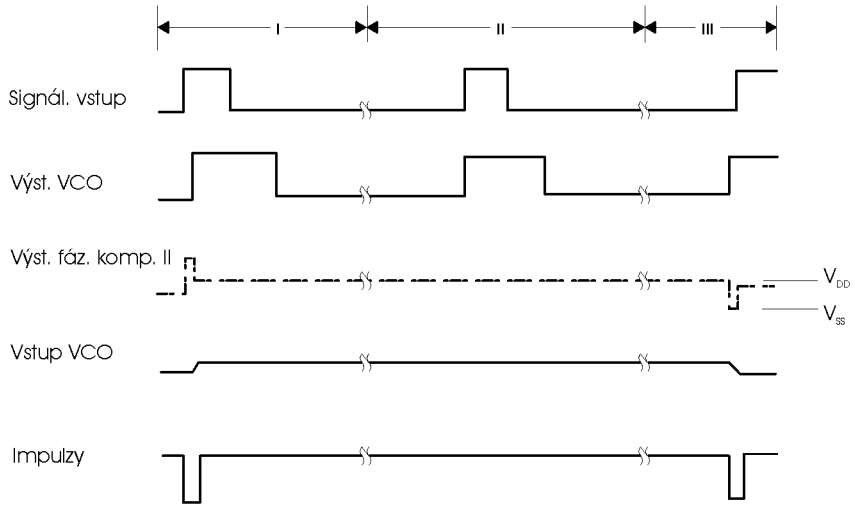
Většina obvodů fázových závěsů používá na místě fázového komparátoru symetrické směšovače sestavené z přesných zesilovačů. Zapojení 4046 je v tomto ohledu jiné, oba fázové komparátory jsou řešeny digitálně, viz vnitřní zapojení [7]. Signálový vstup je na vývodu 14 a může být vázán přímo, přičemž rozkmit signálů je v intervalu CMOS úrovní. Vstupní signály menšího rozkmitu musí být vázány kapacitně. Vstup na vývodu 14 je přiveden do zesilovače s automatickým nastavením zisku. Za ním je v sérii čtveřice invertujících zesilovačů. Jejich výstup je přiveden na oba vstupy fázových komparátorů. Fázový komparátor I je standardní XOR hradlo. Oba signály na jeho vstupech by měly mít plnění 50%, aby se zajistil maximální rozsah zavěšení.



Obr. 7: Vnitřní zapojení 4046 [8]

Pokud není na vstupu XOR hradla žádný signál je na jeho výstupu napětí rovné polovině napětí napájecího. Filtr vně obvodu toto napětí zprůměruje a zpětnovazební smyčkou přivádí

do VCO. Rozsah frekvencí, kdy je možné zavěšení fázového komparátoru I závisí na charakteristice zpětnovazebního filtru. Výhodou fázového komparátoru I je schopnost udržet zavěšení i pro vstupní signály s velkým množstvím šumu. Typické průběhy komparátoru s XOR hradlem jsou na obrázku 4 uvedeném v předchozí kapitole. Fázový komparátor II je hranou řízená digitální paměťová síť, sestavená ze čtyř klopných obvodů RS, řídícího hradla a třístavového výstupního obvodu viz vnitřní zapojení [7]. Výstupní část je sestavena ze dvou komplementárních tranzistorů. Podle toho, který je otevřený se výstup přitahuje k úrovni VDD nebo VSS. Komparátor reaguje pouze na kladné hrany signálu. Princip je následující. Pokud je frekvence vstupního signálu vyšší než frekvence signálu z VCO je p-MOS permanentně otevřený. V případě nižší frekvence je permanentně otevřený naopak n-MOS. Při stejných frekvencích a zpožďování externího signálu za signálem VCO je n-MOS otevřený po dobu odpovídající fázovému rozdílu. Analogická situace nastává pro p-MOS v případě předbíhání signálů. Potom je napětí na kondenzátoru zpětnovazebního filtru nastaveno během doby, kdy jsou oba signály shodné ve frekvenci i fázi. Oba výstupní tranzistory po tuto dobu zůstávají uzavřeny a výstupní obvod fázového komparátoru se chová jako otevřený obvod a drží napětí na kondenzátoru konstantní. Mimoto, signál na vývodu 1 (impulzy) je ve vysoké úrovni a lze ho využít k signalizaci zavěšení. Pro fázový komparátor II neexistuje fázový rozdíl mezi vstupními signály podél celého frekvenčního rozsahu VCO. Výhodou fázového komparátoru II je spotřeba, protože oba výstupní tranzistory jsou většinu času uzavřeny. Rozsah zavěšení je nezávislý na zpětnovazebním filtru. Typické průběhy fázového komparátoru II při zavěšení jsou uvedeny na obrázku 8.



Obr. 8: Průběhy fázového komparátoru II [8]

Vnitřní zapojení napěťově řízeného oscilátoru je uvedeno v katalogovém listě. Aby se zajistila nízká spotřeba celého integrovaného obvodu, je nutné tomuto kritériu podřídit i zpětnovazební filtr. U RC filtru to znamená spojení velkého odporu a malého kondenzátoru. Současně musíme dát pozor, aby se filtr nezatěžoval obvodem zařazeným za ním a

neovlivnila se jeho charakteristika. Vstup VCO zohledňuje požadavek na minimální zatížení, signál je přiveden na hradla dvojice n-MOSů. Jejich impedance je teoreticky nekonečná, což poskytuje při návrhu filtru a volbě hodnot součástek velkou svobodu. Nyní k funkci samotného VCO. Je-li vstup inhibit v nízké úrovni je tranzistor p3 plně otevřený a source tranzistorů p1 a p2 jsou připojené na napájení VDD, hradla gate 1 a gate 2 jsou aktivní. Tranzistor n1 a externí rezistor R1 tvoří sledovač signálu. Za podmínky, že odpor R1 je minimálně řádově větší než odpor n1 v sepnutém stavu (větší než 10 kΩ), proud rezistorem má lineární závislost na vstupním napětí VCO. Tento proud teče tranzistorem p1 tvořící s p2 proudové zrcadlo. Externí rezistor R2 nastavuje tranzistorem p1 dodatečný konstantní proud. Účelem proudu nastaveného rezistorem R2 je kompenzace pracovní frekvence při nulovém vstupním napětí VCO. Proud tranzistorem p2 je stejný i u p1 (proudové zrcadlo) nezávisle na napětí drainem p2. Klopný obvod RS sestavený z hradel gate 1 a 2 otvírá tranzistorové páry p4, n3 nebo p5, n2. Potom je jeden pól externího kondenzátoru C1 uzemněný, zatímco druhý je nabíjen konstantním proudem dodávaným tranzistorem p2. Jakmile napětí na C1 dosáhne hodnoty odpovídající překlopení invertorů 1 nebo 5 klopný obvod změní stav a nabitý kondenzátor se uzemní a vybije přes jeden z n-MOSů. Tehdy začíná nový pracovní cyklus. Střída generovaného signálu je dána přenosovou charakteristikou invertorů 1 a 5, ty jsou stejné a analogicky i střída je stejná resp. 50%. Čtveřice invertorů 1 až 4 a 5 až 8 má několik účelů. Prvním je tvarovat pomalu se měnící signál. Druhý úkol je zajistit malou spotřebu spolu s použitím vysoko-impedančního prvku. Poslední rolí je vytvořit zpoždění, aby klopný obvod správně reagoval na spouštěcí impulz. Pro některé aplikace je vhodné používat demodulovaný signál odebraný ze zpětnovazebního filtru. Aby se filtr nezatěžoval je odbírán z vývodu 9 přes signálový sledovač tvořený tranzistorem n4 a rezistorem R5. Výstup sledovače je na vývodu 10. Pokud je výstup používán musí se na vývod 10 zapojit zatěžovací rezistor o hodnotě minimálně 10 kΩ. Je-li výstup bez uplatnění, měl by vývod zůstat plovoucí. Aktivace a deaktivace VCO probíhá prostřednictvím vstupu inhibit. Logická nula znamená připojení a logická jednička odpojení, vhodné např. pro režim stand-by kvůli snížení spotřeby.

Předchozí odstavec uzavírá popis celého vnitřního zapojení a vysvětlení funkce. Další doplňující informace nalezneme v katalogovém listě. Zde je uvedeno i několik typických aplikací integrovaného obvodu a tabulky pro volbu některých externích součástek.

3.2.3 Programovatelná dělička s širokým rozsahem 4059

Je posledním z trojice popsaných integrovaných obvodů. V podstatě se jedná o čítač s širokým rozsahem 3 až 15999. Obvod je schopen pracovat ve třech režimech – časovač, čítač, oba s rozsahem 3 až 15999 a fixně nastavená dělička 10000. Abychom pochopili funkci obvodu, je nutné nahlédnout na vnitřní blokové schéma, viz katalogový list.

Základ integrovaného obvodu je čítací sekce složená ze třech bloků. Prvním blokem je 1st Counting Section (první čítací sekce) za ní následuje Intermediate Counting Section (prostřední čítací sekce) a posledním blokem je 5th Counting Section (pátá čítací sekce[4]). Jednotlivé sekce se nastavují prostřednictví šestnácti vstupů J_1 až J_{16}, rozdělených na čtyři skupiny po čtyřech. První čítací sekce je čítač schopný dělit maximálně deseti. Prostřední čítací sekce obsahuje tři čítače nastavitelné v rozsahu 1-10, každému odpovídá jedna ze třech čtveřic nastavovacích vstupů J_5 až J_8, J_9 až J_{12}, J_{13} až J_{16}. Poslední čítač je schopen dělit maximálně osmi. Číslo n, kterým dělíme, se zadává v kódu BCD. Funkční tabulka, viz katalogový list, ukazuje jednotlivé možnosti nastavení. K nastavení módu slouží vstupy K_a, K_b, K_c a enable vstup EL. Vstup signálu je označen CP (Clock Imput) a je přiveden přímo do první čítací sekce. Výstupním signálem je impulz o šířce hodinového cyklu a frekvenci úměrné nastavenému číslu n. Signálový výstup O (Divide-by-n Output) je kompatibilní s úrovněmi TTL.

Nastavovací vstupy J_1 až J_4 mohou být nastavovacími vstupy první čítací sekce nebo poslední (páté) čítací sekce, případně mohou být oběma sdíleny, záleží, jak je nastaven mód pomocí vstupů K_a, K_b, K_c a EL. Možnosti nastavení jsou dobře vidět ve funkční tabulce. Například jsou-li všechny vstupy K_a, K_b, K_c ve vysoké úrovni a EL v nízké úrovni integrovaný obvod pracuje jako dělička číslem n. Dělící číslo n je potom nastavitelné následovně. První čítací sekce využívá pouze vstup J_1 a může dělit maximálně dvěma. Prostřední čítací sekci mohu pomocí J_5 až J_{16} nastavit na dělení 1 až 1000. Poslední (pátá) sekce využívá vstupy J_2, J_3, J_4 což odpovídá dělení maximálně osmi. Podobně nastavím-li K_a= K_b= H, K_c=L a EL=L mám dle funkční tabulky nastaven mód dělení maximálně deseti pro první sekci. Prostřední sekce je opět kaskáda třech BCD desítkových čítačů a poslední sekce není přítomna. Ostatní režimy čítače zde nebudu popisovat, protože nejsou využitelné pro náš záměr. Jsou uvedeny spolu s ostatními parametry v katalogovém listě.

3.2.4 Obvodové řešení kmitočtové syntézy

Schéma zapojení uvedené na obrázku 9, je založeno na třech výše popsaných integrovaných obvodech. Referenční oscilátor je sestaven s obvodem 4060 a vychází ze zapojení doporučeného katalogem. Tvoří jej prvky IC1, Q1, C1, C2, R1 a R2. K obvodu fázového závěsu 4046 je oscilátor připojen pomocí pinové lišty. Tímto způsobem je možné zvolit, který z výstupů se bude využívat (jaká bude výstupní frekvence). Osadíme-li krystal do patice je možné kdykoliv snadno změnit základní frekvenci i interval hodnot volbou jiného krystalu a přepnutím výstupu oscilátoru. V tabulce 2 jsou uvedeny hodnoty krystalů a z nich odvozené základní frekvence na jednotlivých výstupech čítače. Výhodné jsou krystaly

[4] Přestože jsou zde tři bloky, tak prostřední čítací sekce je dále rozdělena na tři děličky 10. Dohromady je tedy dílčích bloků pět a poslední je právě pátá.

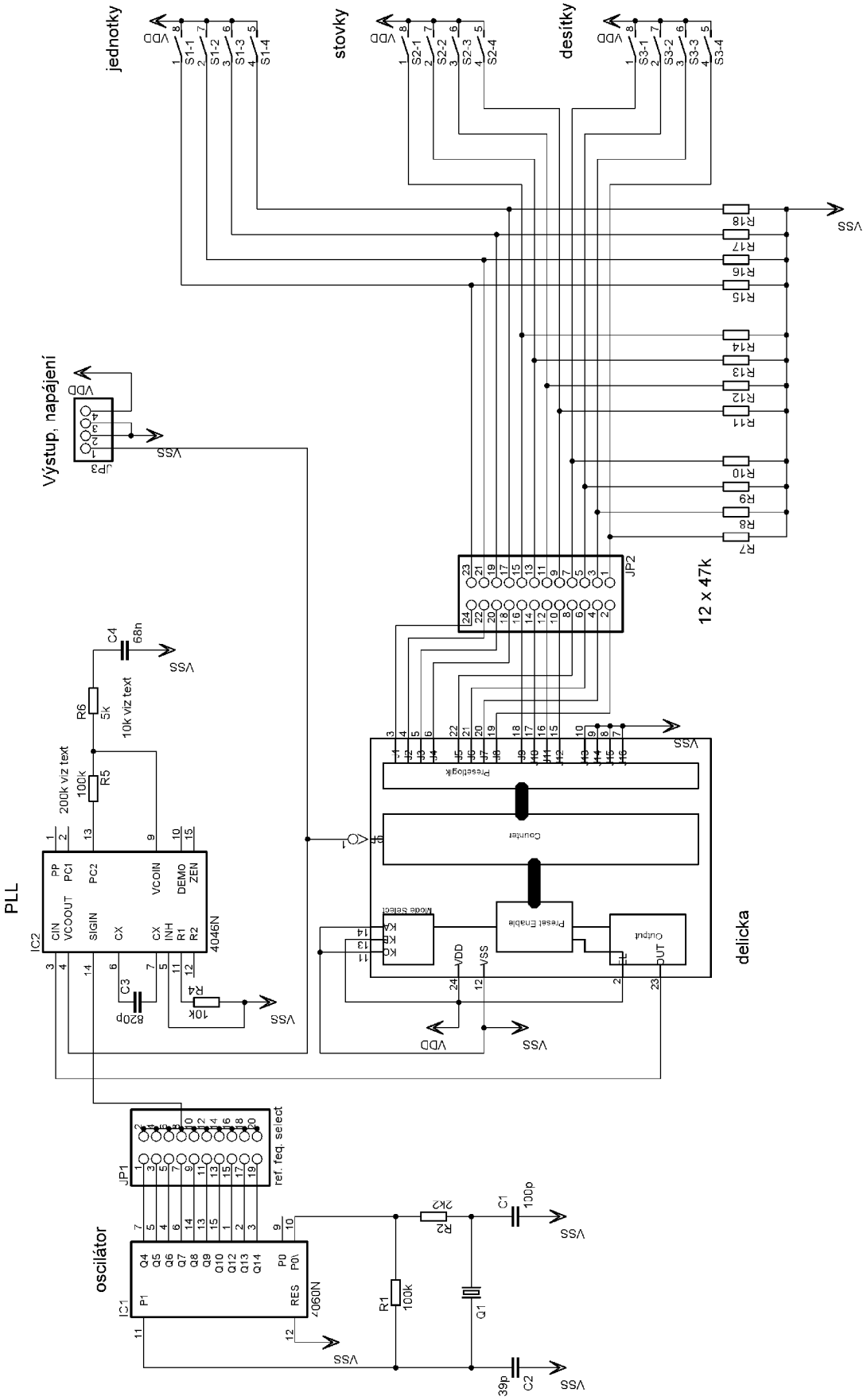
s hodnotou danou mocninou dvou. Jejich vydělením dostaneme vždy hodnotu 1 kHz odpovídající minimálnímu kroku syntézy. Krok 100 Hz a 200 Hz výhodný pro naši aplikaci se dá získat dělením frekvencí[5] 3,2768 MHz, 1638,4 kHz nebo 819,2 kHz.

Tab. 2: Základní frekvence získané volbou krystalu a výstupu oscilátoru

$f_{krystalu}$ (kHz)	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q12	Q13	Q14
16384	1024	512	256	128	64	32	16	4	2	1
8192	512,0	256	128	64	32	16	8	2	1	0,5
4096	256	128	64	32	16	8	4	1	0,5	0,25
1638,4	102,4	51,2	25,6	12,8	6,4	3,2	1,6	0,4	0,2	0,1
1024	64	32	16	8	4	2	1	0,25	0,125	0,063
819,2	51,2	25,6	12,8	6,4	3,2	1,6	0,8	0,2	0,1	0,05
512	32	16	8	4	2	1	0,5	0,125	0,063	0,031
256	16	8	4	2	1	0,5	0,25	0,063	0,031	0,016
32,768	2,048	1,024	0,512	0,256	0,128	0,064	0,032	0,008	0,004	0,002

Zapojení fázového závěsu vychází z předchozích úvah. Integrovaný obvod IC2 je doplněn o nezbytné externí prvky. Rezistor R4 a kondenzátor C3 udávají maximální rezonanční frekvenci VCO. S ohledem na určitou rezervu jsem zvolil širší rozsah 5 kHz až 100 kHz na rozdíl od zadaných 10 kHz až 50 kHz. Odpovídající středová frekvence je potom 47,5 kHz. Hodnoty pro tuto frekvenci udává graf v katalogovém listě. Zvolil jsem hodnotu R1 = 10 k, napájecí napětí bude 5 V, podle grafu tomu odpovídá hodnota kondenzátoru C1 = 700 – 800 pF. V dalším kroku je nutné určit hodnoty součástek zpětnovazebního filtru zapojeného mezi výstupem VCO a fázovým komparátorem. Filtr typu dolní propust je pasivní a tvoří ho prvky R5, R6 a C4. Poměr mezi hodnotami rezistorů R5 a R6 by měl být asi 10:1 až 20:1, přičemž R5 má ležet v intervalu (10 kΩ, 330 kΩ) a R6 (1 kΩ, 33 kΩ). Časová konstanta R5, C4 ovlivňuje čas nutný k zavěšení. Rezistor R6 udává činitel tlumení, jeho malá hodnota způsobí neustálou oscilaci VCO okolo požadované hodnoty. Celková časová konstanta udává minimální frekvenci, kdy bude zpětnovazební systém stabilní. Příliš malá hodnota způsobí, že na osciloskopu uvidíme neustálé přelaďování. Návrhy zpětnovazebních filtrů pro fázové závěsy jsou poměrně obsáhlou problematikou, které se věnují celé publikace. V této práci není prostor na složité úvahy a výpočty zpětnovazebního filtru. Proto jsem pro výpočty hodnot jeho součástek použil návrhový program PLL.zip[6] firmy Philips. Na návrh tímto způsobem se odkazují i katalogové listy. Po zadání výchozích parametrů jsem obdržel konkrétní hodnoty filtru: R5 = 100 kΩ, R6 = 5 kΩ, C4 = 68 nF. Hodnoty je vhodné později upravit na základě výsledků při ověřování funkce.

[5] Jedná se o hodnoty odvozené z tzv. hodinového krystalu 32,768 kHz.
[6] Program je na CD s diplomovou prací.



Obr. 9 – Schéma zapojení kmitočtové syntézy s fázovým závěsem

Poslední částí schématu je dělička zapojená ve zpětné vazbě, mezi výstupem VCO a vstupem fázového komparátorů. Výstup VCO je i výstupem kmitočtové syntézy. Programovatelná dělička IC3 je pomocí volby K_a= K_c= L, K_b= H a EL = H zapojena do režimu dělení číslem n s aktivní první a prostřední čítací sekcí. Poslední čítací sekce je vynechána. Maximální rozsah u tohoto režimu je 3 – 15999, přičemž jednotlivé čtveřice vstupů J_1 až J_4, J_5 až J_8, J_9 až J_{12}, J_{13} až J_{16} reprezentují jednotky až tisíce. Nastavení jednotlivých čtveřic je v kódu BCD. Uvedené schéma nevyužívá poslední čtveřici J_{13} až J_{16} tj. tisíce, tyto vstupy jsou uzemněny. Desítkově lze tedy nastavit číslo 3[7] až 999. Například při základní frekvenci 200 Hz je teoretická maximální frekvence daná děličkou 199,8[8] kHz.

Schéma na obrázku 9 je experimentální, obvod je nutné podle něj sestavit a ověřit jeho funkci. Je možné, že externí prvky fázového závěsu se budou muset přepočítat a změnit jejich hodnoty, to se týká především zpětnovazebního filtru. Vstupy děličky jsou připojeny na trojici DIL spínačů, v budoucím řešení se počítá s jejich vynecháním a napojením přímo na porty mikrokontroleru.

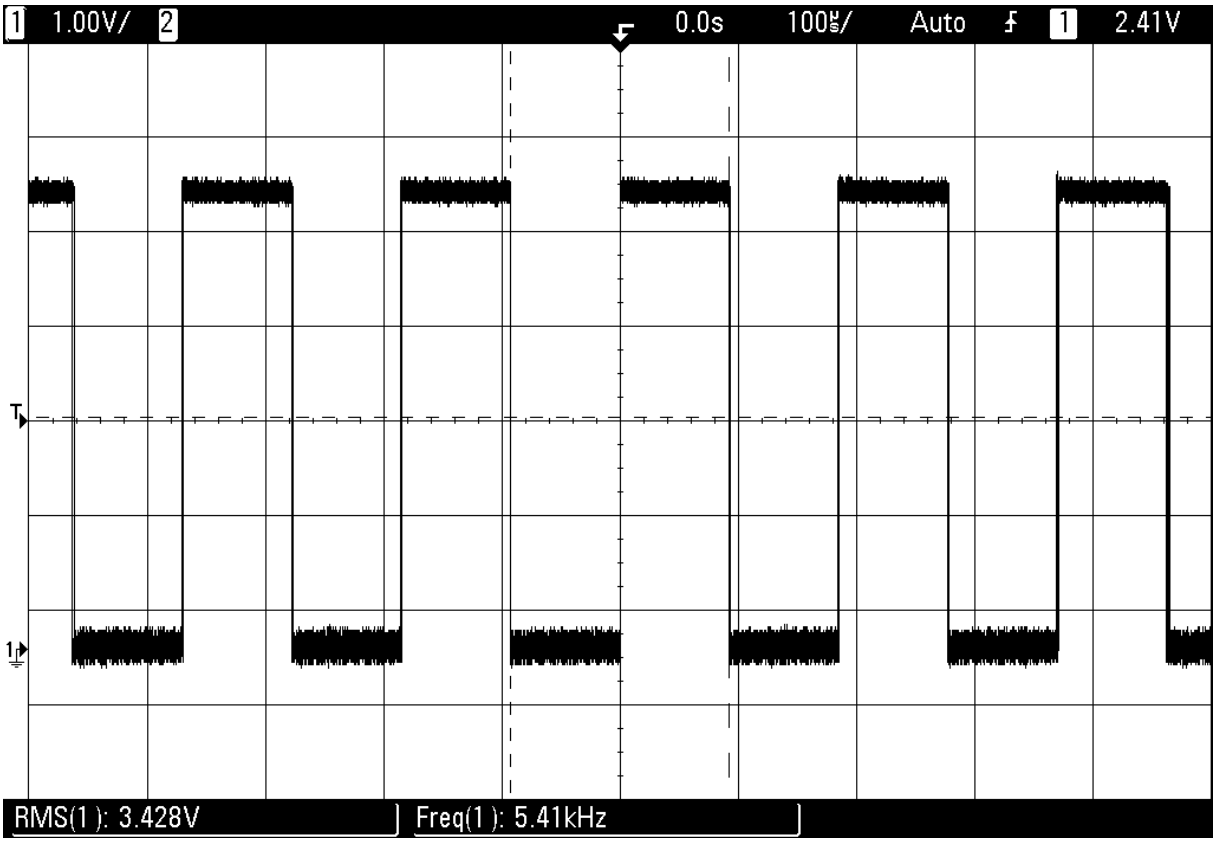
3.2.5 Ověření funkce kmitočtové syntézy

Na základě schématu na obrázku 9 jsem navrhnul jednovrstvý tištěný spoj a na něm realizoval kmitočtovou syntézu. Krystal jsem použil s frekvencí 3,2768 MHz a propojkou na pinové liště připojil poslední výstup Q14. Tato kombinace odpovídá základní frekvenci 200 Hz. Minimální hodnota děličky je potom 25 pro frekvenci 5 kHz. Maximální hodnota děličky je 499 pro frekvenci 99,8 kHz. Při oživování jsem postupoval tak, že jsem nejprve osadil všechny pasivní prvky (ty kritické do dutinek) a patice integrovaných obvodů. Dále jsem osadil IC1 a po připojení napájení jsem na osciloskopu ověřil funkci oscilátoru. Poté jsem osadil i IC2. Do neosazené patice IC3 (programovatelné děličky) jsem vložil propojku mezi vývody 1 (vstup) a 23 (výstup). A na osciloskopu ověřil, že syntéza bezproblémově běží na minimální frekvenci 5 kHz. Pro tento krok je nutné propojkou na pinové liště frekvenci upravit na hodnotu blízkou minimální hodnotě 5 kHz. Při nesprávné funkci filtru překládá jeden kmit více jiných kmitů. V tom případě je nutné upravit hodnoty součástek filtru, jehož časová konstanta je příliš malá a dochází k neustálému přelaďování. Řešením je zvýšit hodnoty rezistorů R5 a R6. Toto nastalo i v našem případě, proto byly hodnoty upraveny následovně R5 = 200 kΩ a R6 = 10 kΩ. Poslední fází oživení je osazení děličky IC3. S ní jsem nejprve ověřil nejnižší hodnotu 5 kHz. Jako druhá hodnota se volí nejvyšší, tím se ověří správné hodnoty externích prvků VCO. Pokud by byly hodnoty R4, C3 příliš nízké zastavilo by se zvyšování frekvence před požadovanou hodnotou a vyšší dělící poměry by byly bez odezvy. S námi zvolenými hodnotami je maximální horní frekvence syntézy asi 105 kHz.

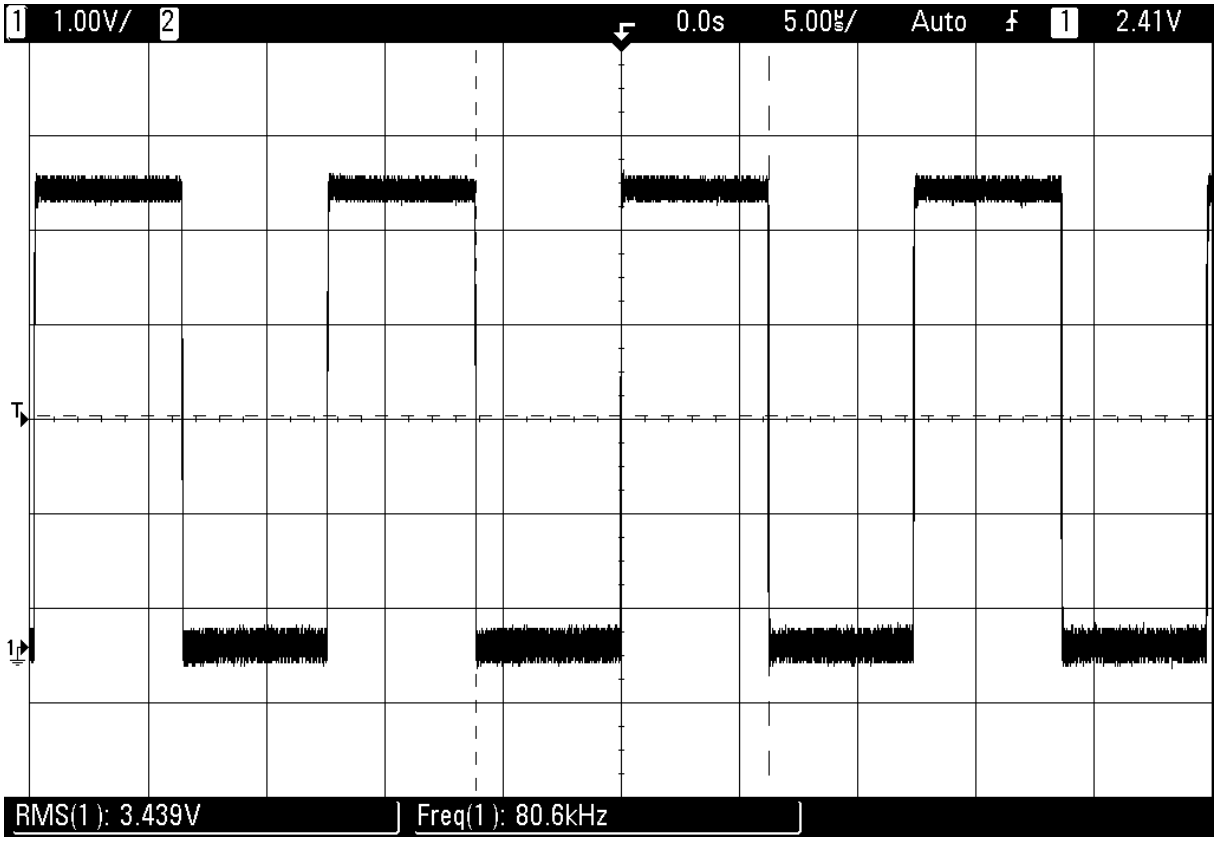
[7] Dělení třemi je minimální hodnota daná vnitřním zapojením integrovaného obvodu.
[8] Tato hodnota je pochopitelně mimo rozsah VCO.

Potvrzení správné funkce jsem ověřil navolením několika libovolných frekvencí z celého spektra a jejich kontrolou na osciloskopu, tím je oživení dokončeno. Na obrázcích 10 a 11 jsou uložené výstupní průběhy z digitálního osciloskopu.

Tento přístup, ač se zdá zbytečný, se nakonec ukázal jako velmi cenný. Díky němu se odstranilo několik chyb a upravily se hodnoty některých součástek. Výsledkem je funkční kmitočtová syntéza s rozsahem 5 kHz – 100 kHz a krokem 0,2 kHz.



Obr. 10: Výstupní průběhy kmitočtové syntézy



Obr. 11: Výstupní průběhy kmitočtové syntézy

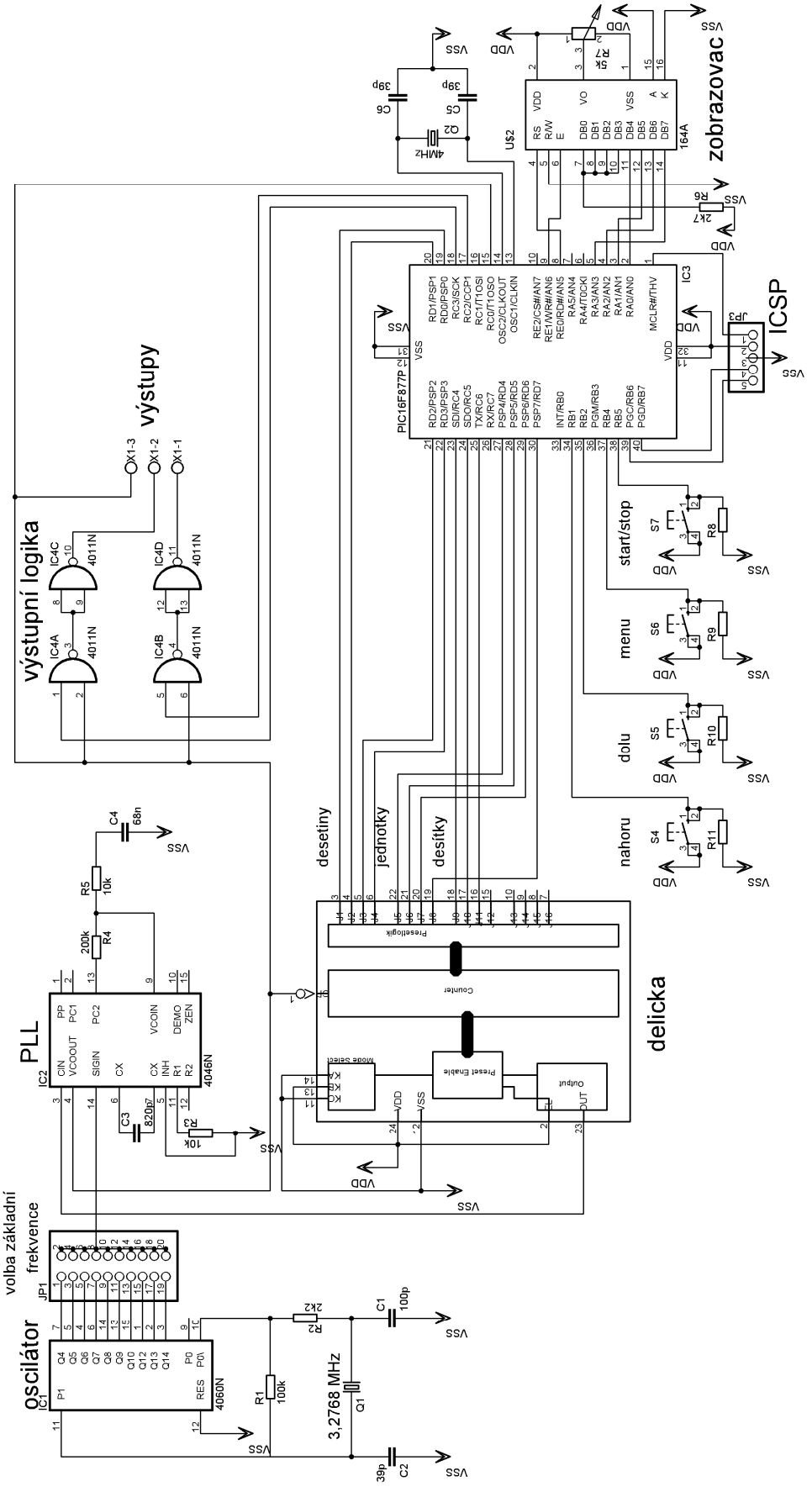
4 Protitaktní generátor

Doposud jsem se věnoval pouze frekvenční syntéze, která je jedním z bloků celého zařízení. Náplní dalších kapitol této diplomové práce bude přechod od navrhnuté frekvenční syntézy k frekvenčnímu generátoru s parametry uvedenými v kapitole 2. V první řadě se musí vyřešit obsluha generátoru tj. nastavení parametrů a jejich zobrazení. Dalším problém je rozdělení jednoho signálového výstupu na dva protitaktní, kdy každý pracovní cyklus bude obsahovat zadaný počet impulzů. Posledním úkolem bude úprava střídy výstupního signálu, jemuž bude věnována samostatná kapitola. Celkové řešení je naznačeno již v kapitole 2, kde je na obrázku 2 uvedeno i blokové schéma.

Ovládání kmitočtové syntézy, jejíž schéma je na obrázku 9, je zajištěno třemi čtyřnásobnými DIL spínači. Jednotlivé čtveřice odpovídají desetinám, jednotkám a desítkám, každá z nich je zadávána v kódu BCD. Chceme-li nastavit jednotlivé kódy prostřednictvím mikrokontroleru, stačí vynechat použité přepínače a jednotlivé vstupy děličky připojit přímo k mikrokontroleru. Dále mikrokontroler vybavit tlačítky a displejem a pomocí jeho programu realizovat nastavení děličky podle schématu: Stisknuto tlačítko „nahoru“ zvyš frekvenci o požadovanou hodnotu a zobraz ji na displeji a analogicky stisknuto tlačítko „dolů“ sniž frekvenci o požadovanou hodnotu a zobraz ji na displeji.

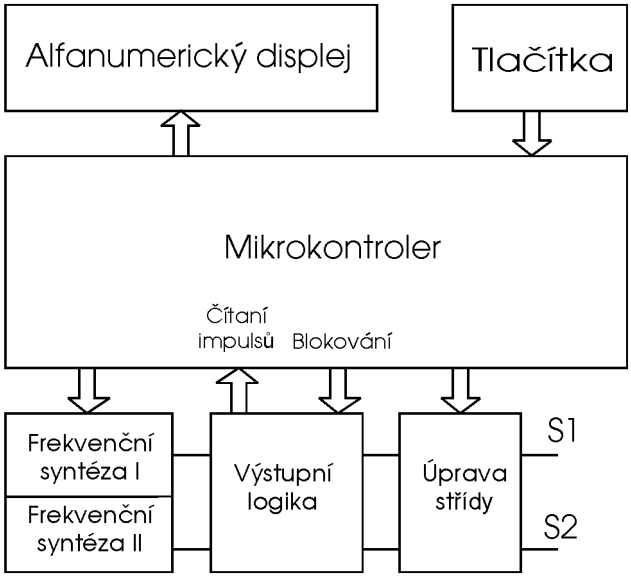
Kromě toho lze využít časovač uvnitř mikrokontroleru, nastavit jej do režimu čítání a počítat impulzy generované syntézou. Na základě údaje o impulzech mohu pomocí digitálního přepínače rozčlenit výstupní signál na dva protitaktní signály. Nejjednodušším řešením digitálního přepínače je použití dvou AND hradel, jako je tomu na obrázku 2. Funkce je následující. Hradlo odblokuji logickou jedničkou nastavenou mikrokontrolerem na základě údaje o počtu impulzů. Odblokované hradlo poté kopíruje na výstup (signálový výstup S_1 nebo S_2) vstupní signál. Aktivní výstup odpovídá aktivnímu hradlu, což je řízeno mikrokontrolerem. Například u pracovního cyklu daného deseti impulzy na výstupu S_1 a osmi na výstupu S_2 musím vytvořit program pro mikrokontroler takto. Nastav čítač, aby čítal do desíti a poté vyvolal přerušení. Odblokuj hradlo odpovídající výstupu S_1. Po vyvolání přerušení přetečením čítače zablokuj hradlo odpovídající výstupu S_1, nastav čítač na hodnotu osm. Aktivuj hradlo odpovídající výstupu S_2. Po přetečení čítače započni znovu cyklus. Kromě toho musí mikrokontroler opět zabezpečit nastavení počtu impulzů pomocí tlačítek a nastavení zobrazit na displeji.

Prvním krokem k návrhu této části je volba vhodného typu mikrokontroleru a displeje. Jako zobrazovač se jeví nejvhodnější inteligentní alfanumerický displej postavený na standardu HD44780. Co se týče mikrokontroleru, tak v zadání se přímo hovoří o mikrokontroleru PIC firmy Microchip. Protože budeme potřebovat hodně portů k ovládání periferních obvodů, zvolil jsem typ PIC16F877A.



Obr. 12: Schéma generátoru – první verze

S ohledem na koncept uvedený v kapitole 2 a předchozích několika odstavcích této kapitoly byl navržen generátor s jedinou kmitočtovou syntézou. Schéma generátoru je uvedeno na obrázku 12. Na základě schématu z obrázku 12 byl navržen plošný spoj, byla vytvořena programová obsluha a celé zařízení jsem realizoval. Při kontrolním měření byl ovšem zjištěn závažný nedostatek, způsobený opomenutím jednoho z parametrů kmitočtové syntézy při návrhu generátoru. Zmiňovanou vlastností syntézy je tzv. čas nutný k ustálení syntézy anglicky settling time. Označuje se tak doba nutná k ustálení syntézy při změně frekvence. Doba ustálení je daná násobkem časové konstanty zpětnovazebního filtru a zesílení zpětnovazební smyčky. Během doby ustálení není fázový závěs zavěšen na požadovanou frekvenci a tudíž výstupní frekvence VCO (výstupní frekvence syntézy) neodpovídá požadované frekvenci. Nyní stačí provést jednoduchou úvahu. Pro základní frekvenci 100-200 Hz používám filtr, jehož časová konstanta je zhruba 0,002. Vynásobím-li tuto hodnotu zesílením zpětnovazební smyčky, řádově tisíc, dostávám hodnotu v řádu jednotek sekund. Přitom, aby generátor správně pracoval, potřebuji mít čas ustálení výrazně lepší než je perioda nejvyšší frekvence násobená počtem impulzů, konkrétně asi 1 µs. Závěrem tohoto odstavce je, že v našem případě nelze použít jedinou kmitočtovou syntézu, která se bude střídavě přelaďovat, ke generování dvou protitaktních signálů. Jediným řešením je použití dvou bloků kmitočtových syntéz, ty naladit na požadované frekvence a poté je pouze střídavě blokovat. Blokové zapojení konceptu se dvěma kmitočtovými syntézami je uvedeno na obrázku 13. Ostatní úvahy zůstávají nezměněny.



Obr. 13: Koncept generátoru se dvěma kmitočtovými syntézami

Než uvedu návrh druhé verze generátoru, se dvěma kmitočtovými syntézami, zaměřím se na popis dvou použitých součástek. Těmi jsou mikrokontroler a inteligentní zobrazovač.

4.1 Mikrokontroler Microchip PIC16F877A

Z označení mikrokontroleru lze dekódovat, že se jedná o řadu PIC16. Označovanou společností Microchip slovem midrange, volně přeloženo jako střední třída, neboli mikrokontrolery pro běžné, všeobecné použití. Písmeno F následující za označením řady říká, že je mikrokontroler vybaven pamětí typu Flash. Mikrokontroler 16F877 je jedním z nejlépe vybavených v řadě PIC16. Jednotlivé základní vlastnosti jsou uvedeny v tabulce 3. V dalších odstavcích budu popisovat pouze ty vlastnosti a funkce mikrokontroleru, které budu dále využívat při návrhu. Veškeré ostatní detaily a funkce lze nalézt v obsáhlém katalogovém listě [10].

Tab. 3: Vlastnosti mikrokontroleru Microchip PIC16F877A

Napájecí napětí	2,0 - 5,5
Maximální pracovní frekvence (MHz)	20
ICSP	ANO
Paměť programu (kB)	14,3
Paměť programu (inst.)	8192
SRAM (bajtů)	368
EPROM (bajtů)	256
Počet I/O portů	33
Počet kanálů ADC	8
Počet PWM modulů	2
SPI sběrnice	ANO
I^2C sběrnice	ANO
USART	ANO
Časovače 8/16 bitů	2/1
Počet komparátorů	2

Jak již bylo řečeno v předchozí kapitole, výhodou mikrokontroleru je velký počet I/O portů. Třicet tři bitů I/O portů je rozděleno do pěti skupin označených písmeny A až E, tedy PORTA až PORTE. Jednotlivé porty nemají stejný počet bitů. PORTA je šestibitový a jednotlivé bity (vývody mikrokontroleru) mohou být využívány jako standardní vstupně výstupní brány. Pin RA4 je s otevřeným drainem. Po resetu mikrokontroleru jsou všechny piny portu PORTA, nastaveny jako analogové vstupy a je na ně připojen A/D převodník. Pokud je chceme využívat jako digitální I/O porty musíme nejprve nastavit příslušné bity registru ADCON1 A/D převodníku. To samé platí i pro tříbitový PORTE. Porty PORTB, PORTC a PORTD jsou osmibitové a lze je využívat jako digitální I/O rozhraní.

Kromě vstupně / výstupních portů budu využívat periferní zařízení mikrokontroleru TIMER1. TIMER1 je šestnáctibitový časovač/čítač. Pro navrhované zařízení je výhodný režim čítače. Pomocí něj lze počítat impulzy generované syntézou. Přestože má mikrokontroler ještě další dva čítače/časovače TIMER0 a TIMER2, TIMER1 je jediným, který dokáže čítat impulzy asynchronně. Funkce je patrná z obrázku FIGURE 6-2 v katalogovém listě na straně 57. Zde je také dobře vidět, které konfigurační bity je nutné

nastavit. Nastavení se provádí pomocí registru T1CON. Registr čítače je rozdělen na dva osmibitové registry TMR1H a TMR1L číselný rozsah je 0x0000 až 0xFFFF. Po přetečení registru čítače z hodnoty 0xFFFF na 0x0000 lze vyvolat přerušení. Přerušení je nutné nejprve nakonfigurovat pomocí registru INTCON a PIE1. Vyvolané přerušení je poté signalizováno tzv. Flag bitem TMR1IF. Pro náš účel je potřeba nakonfigurovat TIMER1 jako asynchronní čítač (frekvence mikrokontroleru a externího signálu jsou rozdílné) inkrementovaný externím signálem s předděličkou 1:1 a zapnutým systémem přerušení. Celé nastavení se provádí v registru T1CON. Postupovat lze bit za bitem, dle zmiňovaného obrázku v katalogovém listě. Nejprve se vypne oscilátor čítače smazáním bitu T1OSCEN, tím se zároveň nastaví vstup na PORTC.0 Dále se nastaví režim čítače nastavením bitu TMR1CS. Bity T1CKPS1 a T1CKPS0 se nastaví do 0, tím se nastaví předdělička na hodnotu 1:1. Vypne se synchronizace nastavením bitu _T1SYNC. Tím je konfigurace provedena, nyní je nutné nastavit systém přerušení a poté je možné začít čítač kdykoliv používat jeho zapnutím, pomocí nastavení bitu TMR1ON.

Systém přerušení mikrokontroleru je schopen využívat až patnáct nezávislých zdrojů. Jejich konfigurace se provádí pomocí registru INTCON a PIE1. Stavové bity přerušení jsou uloženy v registru PIR1. Pro pochopení funkce je nutné nahlédnout do katalogového listu na stranu 153, kde je obrázek FIGURE 14-10, ten zobrazuje schematické zapojení systému přerušení. Aby fungovalo libovolné přerušení, je nutné nejprve povolit tzv. globální přerušení nastavením bitu GIE. Navrhované zařízení bude využívat pouze přerušení vyvolané přetečením registrů časovače / čítače TIMER1. Pro jeho povolení je nutné nastavit bit TMR1IE. Flag bit signalizující toto přerušení je TMR1IF.

Poslední využívanou funkcí mikrokontroleru je I^2C sběrnice. Jedná se o typ komunikačního rozhraní taktéž označovaného zkratkou IIC – Inter Integrated Circuit. Pod tímto názvem se skrývá vylepšená sériová komunikace typu Multi-Master. Rozhraní I^2C v dnešní době obsahuje velké množství moderních součástek a díky němu lze propojit jednotlivé bloky s použitím pouze dvou vodičů. Osobně budu sběrnici využívat ke spojení s tzv. expanderem portů. Což je v podstatě převodník ze sériového na paralelní rozhraní. Při dalším popisu návrhu se ukáže, že je potřeba více portů, než nabízí mikrokontroler 16F877A. Nejjednodušší volbou, jak je získat je použití zmiňovaného expanderu, připojeného přes I^2C. Výstupem expanderu je osmibitový nebo šestnáctibitový obousměrný port. V mikrokontroleru 16F877A je I2C sběrnice součástí tzv. MSSP (Master Synchronous Serial Port) modulu. Sběrnici I^2C se podrobně věnuje katalogový list mikrokontroleru na straně 80 – 108, proto zde uvedu jen základní informace. Souhrn nejnutnějších informací a pravidel je obsažen v následujících dvou odstavcích.

I^2C sběrnice používá dva vodiče. Serial Clock (SCL) připojený k PORTC.3 a Serial Data (SDL) připojený k PORTC.4. Příslušné bity portů jsou s otevřeným drainem, proto je nutné zajistit správnou funkci připojením pull – up rezistorů k SCL i SDL. Zapojení s otevřeným drainem se používá, aby bylo možné úrovně na sběrnici ovlivňovat pomocí master i slave

zařízení. Připojený expandér bude na sběrnici zařízení typu slave (nebude řídit její obsluhu) a bude data pouze přijímat. Zařízení typu slave může být několik, jejich identifikátorem je unikátní adresa každého slave zařízení. Obsluhu sběrnice bude provádět mikrokontroler, ten bude zařízením typu master. Mikrokontroler musí být nakonfigurován v master modu. Chce-li master zařízení (mikrokontroler) inicializovat přenos dat, vyšle nejprve adresu zařízení s kterým chce komunikovat. Všechna slave zařízení poslouchají. Zařízení, které rozpozná adresu, vyšle signál o potvrzení (ACK – Acknowledgement). Adresa zařízení slave nese také bit říkající, kterým směrem budou data přenášena. Potřebuje-li kterékoliv zařízení více času na zpracování signálu ACK nebo přijatého bajtu drží toto zařízení vodič SCL v nízké úrovni signalizující čekání. Přenos pokračuje až po uvolnění SCL. Podmínkou je posílat data v balíku jednoho osmibitového slova, ať už jsou dlouhá jakkoliv. Slave zařízení musí master zařízení vždy potvrdit příjem pomocí ACK, v opačném případě musí master ukončit přenos. Stejně tak musí master zařízení potvrzovat přijetí každého slova signálem ACK, výjimka platí pouze pro poslední slovo, kdy absence ACK signalizuje konec dat. Po letmém seznámení nyní přejdu k popisu master modu I^2C mikrokontroleru.

Zařízení master je zdrojem hodin a podmínek pro započetí a ukončení přenosu. Ukončení přenosu lze provést pomocí podmínky „stop“ nebo opakovanou podmínkou „start“. První vyslaný bajt obsahuje sedmibitovou adresu a bit R/W. Protože chci do zařízení data pouze zapisovat, bude R/W=0. Každé slovo musí být potvrzeno ACK signálem. Typická sekvence pro vysílání by měla vypadat takto:

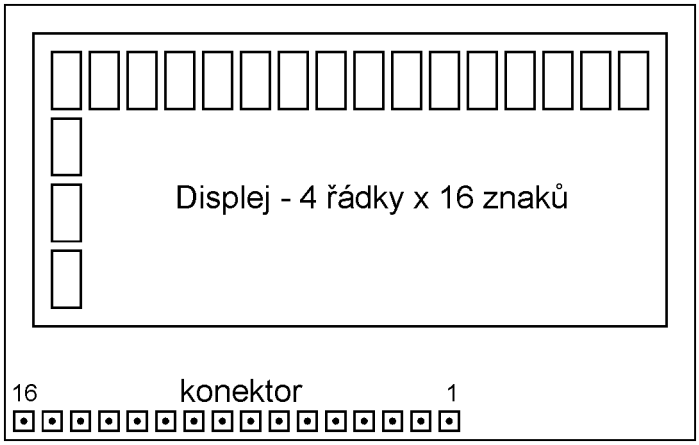
1) Program generuje podmínku „start“ nastavením bitu SEN (SSPCON2.0).
2) SSPIF (příznakový bit) je nastaven. Modul MSSP čeká požadovaný čas před započetím další operace.
3) Program zapíše do registru SSPBUF adresu slave zařízení do kterého se budou posílat data.
4) Jednotlivé bity adresy jsou postupně posílány na vodič SDA, dokud není poslán poslední bit.
5) Hodnota bitu ACK je zapsána do registru SSPCON2.6.
6) MSSP modul generuje přerušení s devátým impulzem na vodiči clock nastavením příznakového bitu SSPIF.
7) Program zapíše do registru SSPBUF osmibitové datové slovo.
8) Data jsou přenesena po vodiči SDA bit za bitem.
9) Hodnota bitu ACK je zapsána do registru SSPCON2.6.
10) MSSP modul generuje přerušení s devátým impulzem na vodiči clock nastavením příznakového bitu SSPIF.
11) Program generuje podmínku „stop“ nastavením bitu „stop enable“ – registr SSPCON2.2.
12) Jakmile je dokončena podmínka „stop“ generuje se přerušení.

Předchozí odstavec uzavírá popis I^2C i celé kapitoly věnované mikrokontroleru, více funkcí využívat nebudu a je tedy zbytečné je popisovat.

4.2 Inteligentní zobrazovač s řídícím obvodem HD44780

Inteligentní zobrazovače jsou velmi výhodné pro aplikace s mikrokontrolery. Na rozdíl od numerických displejů můžeme zobrazovat i různé textové informace. Kód pro jejich obsluhu je sice složitější, ale je nepatrnou cenou za možnost zobrazení velkého množství číselných a textových informací. Generaci znaků má na starost řídící obvod inteligentního zobrazovače. Abychom vypsali na displeji jeden znak, stačí na jeho sběrnici pouze vyslat ASCII kód požadovaného znaku, o zbytek se postará řídící obvod.

Řídící obvod HD44780 firmy Hitachi patří mezi nejpoužívanější obvody pro obsluhu alfanumerických LCD displejů. Zobrazovače s tímto obvodem se prodávají jako celek displeje a řídící elektroniky osazené na malém plošném spoji. Displeje jsou k dispozici v různých konfiguracích s jedním až čtyřmi řádky a osmi až čtyřiceti znaky na řádku. Řídící elektronika je tvořena jedním, či více integrovanými obvody HD44780, podle toho kolik má displej celkem znaků. Funkce zobrazovače je jednoduchá. V paměti řídícího obvodu jsou nadefinované body tvořící jednotlivé znaky na displeji. Adrese v paměti odpovídá standardní ASCII kód zapsaný na sběrnici zobrazovače. Neboli, na vstup displeje zapíši znak v kódu ASCII a řídící obvod se postará o jeho zobrazení na displeji. Každý znak je tvořen maticí bodů 5x7 případně u některých jednořádkových verzí je to 5x10. Obsah paměti neodpovídá plnohodnotnému ASCII kódu, jsou zde zastoupeny pouze alfanumerické znaky a několik běžné používaných symbolů, např. plus, mínus, rovnítko, procenta ap. Pro detaily je nutné nahlédnout do katalogového listu, zde bývá v tabulce uváděn kompletní obsah paměti i s adresami. Na obrázku 14 je ukázka uspořádání displeje, který budu později používat.



Obr. 14: Uspořádání zobrazovače s displejem 4x16

4.2.1 Popis rozhraní inteligentního zobrazovače s obvodem HD44780

Připojení displeje je realizováno zpravidla šestnáctivývodovým konektorem, viz obrázek 14. Zapojení jednotlivých kolíků je uvedeno v tabulce tabulce 4. První tři vývody 1, 2, 3 jsou určeny pro napájení a kontrast. Vývody 4 až 14 jsou datové, pomocí nich probíhá komunikace s okolím. Poslední dva vývody 15, 16 slouží k napájení podsvětlovacích diod.

Tab. 4: Zapojení vývodů inteligentního zobrazovače

1	V_{SS}	Napájení (zem)
2	V_{DD}	Napájení (+5V)
3	V_0	Napájení (kontrast)
4	R_S	Data (výběr registru)
5	R/W	Data (čtení/zápis)
6	E	Data (povolení zápisu)
7	DB0	Data (bit.0)
8	DB1	Data (bit.1)
9	DB2	Data (bit.2)
10	DB3	Data (bit.3)
11	DB4	Data (bit.4)
12	DB5	Data (bit.5)
13	DB6	Data (bit.6)
14	DB7	Data (bit.7)
15	V_A	Napájení (podsvětlení anoda)
16	V_K	Napájení (podsvětlení katoda)

Napájení displeje je 5 V. Pokud není požadováno samostatné zapínání podsvětlení je možné jeho vývody spojit rovnou s napájecími. Podobně můžeme připojit vývod regulace kontrastu přímo přes odporový dělič tvořený trimrem k napájení. Datových vodičů je celkem 11. Vývody 7 až 14 náleží paralelní komunikační sběrnici, na ní se posílají data jako jednotlivá osmibitová, případně čtyřbitová slova. Zbývající tři vývody jsou datové řídící. Data jsou ze sběrnice přepsána do registru zobrazovače po aktivaci řídícího vstupu E (Enable) logickou jedničkou. Registry používá zobrazovač dva, pomocí jednoho se zapisují příkazy a pomocí druhého znaky. Výběr registru probíhá pomocí řídícího vstupu RS (Register Select). Je-li na vstupu RS logická nula, zapisují se příkazy, v opačném případě se zapisují znaky. Posledním řídícím datovým vývodem je R/W (Read/Write), je-li v logické nule, můžeme na sběrnici zapisovat data. V opačném případě jsou data ze sběrnice čtena. U všech aplikací není třeba tento vodič používat, například když nepotřebujeme číst stavy ze zobrazovače. V tom případě můžeme vodič R/W připojit rovnou na zem a ušetřit jeden port mikrokontroleru. Podobně lze naložit i s nevyužívanými paralelními datovými vodiči v případě čtyřbitové

komunikace. Připojit je můžeme přes pull-up / pull-down rezistory[9] na jednu z napájecích větví. Ovšem pozor na případ spojení všech nepoužívaných vývodů rovnou na zem. Tím by mohlo snadno dojít k vnucení příkazu do zobrazovače, prostřednictvím rušivých signálů. Z toho důvodu se většinou připojuje R/W rovnou na zem a nevyužité vývody DB0 – DB3 přes pull-up rezistory na napájení.

Některé typy zobrazovačů mají pouze čtrnáctivývodový konektor, kde není zahrnuto napájení podsvětlení. Dále existují i verze kdy bývá konektor rozdělen na 2x8 (2x7) vývodů. Pro všechny tyto varianty platí zapojení jednotlivých vývodů, uvedené v tabulce 4.

Propojení mikrokontroleru s inteligentním zobrazovačem lze realizovat různými způsoby, přímo v režii zobrazovače je osmibitová a čtyřbitová komunikace. Pro ostatní varianty je nutné použít periferní obvody mezi mikrokontrolerem a zobrazovačem. Například lze převést paralelní komunikaci na sériovou a využít pouze 2 porty mikrokontroleru, ovšem těmito řešeními se v této práci nebudu zabývat. Programově je nejjednodušší osmibitová paralelní komunikace, která je i nejméně náročná na čas procesoru. U osmibitových mikrokontrolerů se využívá celý registr. Nevýhodou je spotřeba velkého množství portů mikrokontroleru. Celkem je jich potřeba minimálně deset (pokud se nevyužívá R/W). Z hlediska použití portů je výhodnější čtyřbitová komunikace využívající pouze horní čtyři bity DB4-DB7. Nevýhodou je složitější kód, kdy musíme na horní čtyři bity sběrnice displeje posílat nejprve horní půl bajt a poté dolní půl bajt. Programově se musí vyřešit rozdělení osmibitového slova na dvě čtyřbitová a případné maskování nepoužitých portů.

4.2.2 Programová obsluha zobrazovače s HD44780

Dalším krokem po zvolení typu komunikace a propojení displeje s porty mikrokontroleru je vyřešení obsluhy zobrazovače po programové stránce. Po připojení displeje k napájení je nejprve třeba po určitém čase nutném ke stabilizaci napájení provést počáteční inicializaci, aby bylo možné začít zobrazovat data. Za dostatečný čas mezi připojením napájení a inicializační sekvencí se považuje 15 ms[10].

Než ji popíši, shrnu mechanizmus zápisu dat a znaků do zobrazovače. Osmibitové slovo posílané na osmibitovou paralelní sběrnici displeje může reprezentovat buď příkazy pro konfiguraci displeje, nebo adresu znaku mu příslušející v ASCII tabulce. Z předchozí kapitoly je patrné, že pokud do zobrazovače cokoliv zapisujeme, musí být R/W v logické nule. Nyní, když zapisujeme do zobrazovače příkazy, zapíšeme na datovou sběrnici příslušné bity a odešleme je do zobrazovače aktivací vstupu E (log 1). Při zápisu znaků je postup podobný, ale

[9] Jejich hodnota se volí v intervalu 4,7 kΩ až 47 kΩ.
[10] Inicializační čas může být spojen s inicializačním časem mikrokontroleru prostřednictvím POR (Power-On Reset).

musíme nejprve zvolit registr znaků aktivací vstupu RS (log 1) a poté zapsat kód znaku a odeslat jej aktivací vstupu E.

Inicializace je sekvence konfiguračních příkazů, kdy nejprve zvolíme typ sběrnice (4 nebo 8 bitů), nastavíme režim zobrazování, zapneme displej a nastavíme adresu prvního znaku. Po ní je displej připraven pro zobrazování znaků, bez jejího provedení by vůbec nepracoval, protože jedním z příkazů je zapnutí. Výčet příkazů je uveden v tabulce 5. Mezi jednotlivými příkazy je nutné dodržet určité časy uvedené v téže tabulce, aby byly příkazy korektně vykonány. Některé z uvedených příkazů neslouží pouze k inicializaci a konfiguraci displeje. Patří sem příkazy pro nastavení pozice kurzoru a smazání displeje.

Pro inicializaci je výrobci doporučován následující postup, dále ilustrovaný i diagramem na obrázku 15. Inicializační příkazy jsou celkem čtyři a v tabulce 5 jsou označeny čísly 3, 4, 5, 6. Prvním inicializačním příkazem bývá vždy „Volba typu komunikace“ 4 / 8 bitů. Příkaz zároveň sdružuje i nastavení počtu využívaných řádků a formát matice jednoho znaku. Osobně využívám všechny řádky displeje a standardní formát znaků 5x7[11]. HD44780 se při zápisu znaku na příslušnou pozici displeje automaticky přesune na další pozici, která je o jedničku výše nebo níže. Nastavení, kterým směrem se má postupovat a zda vůbec má automaticky přecházet na další pozici se nastavuje příkazem „Posun textu“. S tím souvisí také příkaz „Posun displeje/kurzoru“, který nastavuje, zda se bude posouvat pouze kurzor nebo se bude rolovat celý text. Posledním příkazem při inicializaci je „Zapnutí displeje a nastavení kurzoru“, který není třeba dále komentovat.

Tab. 5: Příkazy HD44780 a časy k jejich vykonání

	Binárně								**Hexadecimálně**	**Min. čas**
Příkaz	**DB7**	**DB6**	**DB5**	**DB4**	**DB3**	**DB2**	**DB1**	**DB0**		
1 – Vymaž displej	0	0	0	0	0	0	0	1	0x01	1,64 ms
2 – Návrat na první pozici displeje	0	0	0	0	0	0	1	x	0x02 nebo 0x03	1,64 ms
3 – Posun textu	0	0	0	0	0	1	I/D	P	0x04 až 0x07	40 ms
4 – Zap. displeje a nastav. kurzoru	0	0	0	0	1	Z	PK	B	0x08 až 0x0F	40 ms
5 – Posun displeje / kurzoru	0	0	0	1	D/K	P/L	x	x	0x10 až 0x1F	40 ms
6 – Sběrnice, řádky, znaky	0	0	1	8/4	2/1	10/7	x	x	0x20 až 0x3F	40 ms
7 – Adresa CGRAM	0	1	1/0	1/0	1/0	1/0	1/0	1/0	0x40 až 0x7F	40 ms
8 – Adresa znaku	1	1/0	1/0	1/0	1/0	1/0	1/0	1/0	0x80 až FF	40 ms
x - nezáleží na stavu, * - hodnoty po připojení nap.										
I/D - 1=inkrementace* 0=dekrementace, P - 1=posun textu zapnut 0=posun textu vypnut										
Z - 1=disp. zapnut 0=disp. Vypnut*, PK - 1=podtržení kurzoru 0=bez podtržení*, B - 1=blikání kurzoru										
D/K - 1=posun disp. 0=posun kurzoru*, P/L - 1=posun vpravo* 0=posun vlevo										
8/4 - 1= 8-bit komunikace* 0=4-bit komunikace, 2/1 - 1=dva řádky 0=jeden řádek*, 10/7 - 1=matice jednoho znaku je 5x10 bodů 0=matice jednoho znaku je 5x7 bodů*										

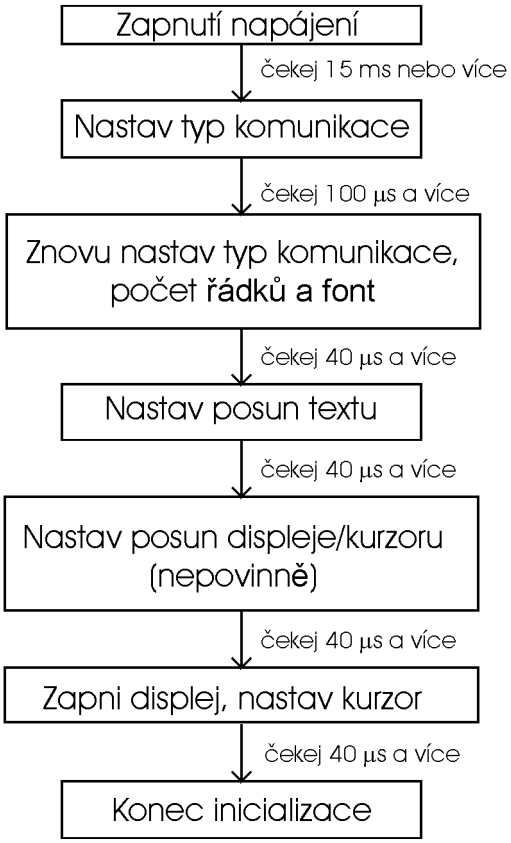
Po provedení inicializace je již možné zobrazovat znaky. Použitelné znaky jsou uloženy ve vnitřní paměti HD44780 v tzv. tabulce znaků. Prvních šestnáct pozic s adresou 0x00 až 0x0f je určeno pro uživatelem definované znaky. Můžeme tak vytvořit znaky

[11] Formát znaků 5x10 je vhodný pro některé znaky, využívající prostor pod linkou např. znaky řecké abecedy.

s diakritikou, různé symboly ap. Adresový prostor 0x10 až 0x1f je prázdný, stejně tak je tomu i u adres 0x80 až 0x9f. Za ním následuje prostor s adresami 0x20 až 0x7f, kde jsou uloženy ASCII znaky. Adresy 0xa0 až 0xdf využívají japonské znaky a 0xe0 až 0xff řecké znaky. Při zobrazování znaků je nejprve nutné nastavit vstup RS do log 1, poté nastavit na datových vodičích adresu odpovídající znaku a přenést ji do zobrazovače uvedením vstupu E do logické jedničky. Ve chvíli, kdy je E ve vysoké úrovni se na displeji objeví znak na první pozici displeje tj. vlevo nahoře.

Všechny pozice na displeji mají svou adresu, která se liší u různých zobrazovačů, podle toho, kolik mají řádků a znaků na řádku. Výchozí pozice je vždy na adrese 0x00. Adresy ostatních pozic je nutné zjistit v katalogovém listě daného zobrazovače. Přechod na požadovanou adresu se provádí pomocí příkazu „Adresa znaku“. S formátováním zobrazovaného textu souvisí také příkazy „Vymaž displej“ a „Návrat na první pozici“. První příkaz „Vymaž displej“ smaže celý displej a nastaví kurzor na výchozí pozici (adresu), tj. 0x00. Druhý příkaz „Návrat na první pozici“ pouze přesune kurzor na adresu 0x00, zobrazené informace zůstávají nezměněny.

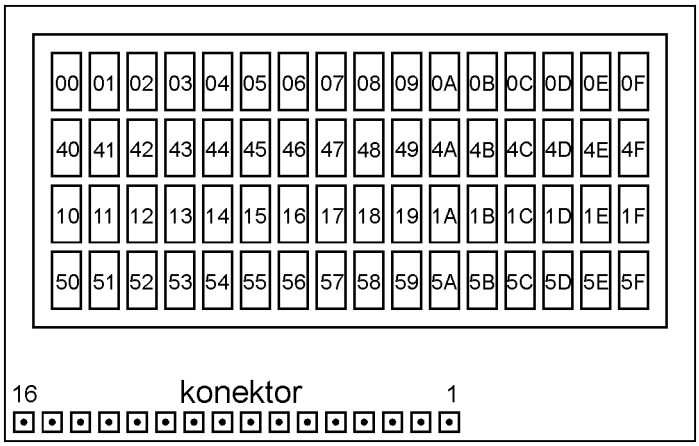
Předchozí odstavec uzavírá obecný popis standardu HD44780. V následující kapitole se budu věnovat inteligentnímu zobrazovači s čtyřřádkovým displejem, který budu dále používat při návrhu protitaktního generátoru.



Obr. 15: Inicializace zobrazovače s HD44780

4.2.3 Inteligentní zobrazovač s displejem 4x16 znaků

Již z názvu kapitoly je zřejmé, že jde o inteligentní zobrazovač postavený na standardu HD44780 s displejem schopným zobrazit až 64 znaků současně. Konfigurace 4x16 znamená čtyři řádky s šestnácti znaky na každém. Přesné označení zobrazovače je MC1604-SYL. K tomuto zobrazovači již není nutný téměř žádný popis, protože vše bylo vysvětleno v předchozích kapitolách. Fyzicky je zobrazovač zcela shodný s tím na obrázku 14. Jediné, co je nutné doplnit, jsou informace o adresách jednotlivých pozic na displeji, ty se totiž liší podle použitého displeje. Adresy jsou uvedeny na následujícím obrázku 16. Na první pohled vypadají adresy jednotlivých řádů nelogicky. Mám na mysli situaci, kdy první znak prvního řádku začíná adresou 0x00 poslední potom 0x0F decimálně 15. Ale již druhý řádek začíná hodnotou 0x40 decimálně 64. Určitě by byla vhodnější adresa 0x10 decimálně 16, která je zcela nelogicky obsazena prvním znakem třetího řádku. Důvodem tohoto rozmístění adres je vnitřní elektrické zapojení zobrazovače. Použitý zobrazovač s displejem 4x16 znaků je ve skutečnosti (s ohledem na elektrické zapojení) zobrazovač dvouřádkový, kdy první řádek je rozdělen fyzicky na dvě částí. První je současně i prvním řádkem displeje a druhý je třetím řádkem displeje. To samé platí i pro druhý a čtvrtý řádek. Navíc, ne všechny adresy displeje jsou viditelné. Na některé lze zapsat znak, ale ten se zobrazí až při zapnutí funkce rolování.



Obr. 16: Adresy jednotlivých pozic na displeji 4x16

Ostatní doplňující informace (rozměry ap.) nalezneme v katalogovém listě výrobce [14], kterým je v tomto konkrétním případě Bona Fide Technology Ltd.

5 Návrh generátoru pro magnetronové naprašování

Po kapitolách věnovaných popisu mikrokontroleru a displeje naváži na kapitolu 4. Jednotlivé úvahy zde budu řešit již konkrétně. Cílem je využít popsanou frekvenční syntézu a s pomocí mikrokontroleru a inteligentního zobrazovače navrhnout zapojení protitaktního generátoru pro magnetronové naprašování. Výsledné zařízení již bude splňovat požadované parametry, ovšem kromě nastavení střídy. Tomuto problému se budu věnovat v další samostatné kapitole, protože pro takovou úpravu signálu jsou nutné další obvody.

5.1 Obvodové řešení generátoru pro magnetronové naprašování

Návrh obvodového řešení je uveden na obrázku 17. Schéma lze rozdělit na dvě části – obvody frekvenční syntézy a obvody mikrokontroleru. Obvody frekvenční syntézy již nebudu znovu popisovat, protože se jím věnovala dostatečně kapitola 3. Pouze zde uvedu několik poznámek týkající se řešení se dvěma kmitočtovými syntézami, vycházející z kapitoly 4.

Frekvenční syntéza je řízena programovatelnou děličkou 4059. Nevýhodou integrovaného obvodu 4059 je, že jde o standardní logiku řady 4000, která nepoužívá u složitějších typů obvodů, jako je tento, žádnou z pokročilejších druhů komunikací. Například některý typ sériové komunikace, která by výrazně uspořila vstupně / výstupní vodiče mikrokontroleru. Dělička 4059 má celkem šestnáct vstupů. Původním záměrem bylo nastavovat frekvenci po 0,2 kHz. Pro maximální frekvenci 99 kHz by musel být maximální dělící poměr 495. Tomu odpovídá využití třech stupňů děličky (celá prostřední čítací část), celkem s 12 vstupy. Pro dvě děličky to je dvojnásobek. Z toho důvodu jsem snížil základní krok syntézy na 1 kHz, maximální číslo na vstupu děličky je potom 99. Navíc nastavovat frekvenci po desetinách je možná i zbytečné. Výhodou je úspora osmi vodičů. I tak se dále ukáže, že po připojení ostatních periferií mikrokontroleru jsou využity téměř všechny bity pěti portů.

Kvůli snížení základní frekvence byly upraveny některé součástky frekvenční syntézy. Krystal oscilátoru má nyní hodnotu 4,096 MHz a za výstup oscilátoru je pomocí propojky na pinové liště zvolen Q12. Touto kombinací, viz tabulka 2, dostáváme základní frekvenci 1 kHz. Dále byly upraveny hodnoty součástek zpětnovazebního filtru. Rezistory R4, R15 na hodnotu 50 kΩ a R5, R16 na 5 kΩ, zbytek zůstal nezměněn. Dělička první kmitočtové syntézy je připojena na celý osmibitový PORTD a dělička druhé kmitočtové syntézy je připojena na celý osmibitový PORTB. Dolní polovina slova odpovídá jednotkám a horní desítkám.

Řešení obvodové části s mikrokontrolerem je velmi jednoduché. Vstupně / výstupními zařízeními jsou displej a tlačítka. Displej využívá čtyřbitovou komunikaci spolu s dvěma řídícími vstupy RS a E. Celkem využívá šest bitů portů mikrokontroleru PORTA.0 až

PORTA.3 a PORTE.0 až PORTE.1. Detailní zapojení je patrné ze schématu. Alokace jednotlivých portů je uvedena také v tabulce 6. Nevyužité vstupy displeje jsou ošetřeny pull-up rezistorem a nevyužívaný výstup R/W je připojen na zem. Proměnný rezistor R7 slouží k nastavení jasu displeje. Tlačítka jsou k mikrokontroleru připojena celkem čtyři prostřednictvím PORTC a PORTE Jejich symbolika bude vysvětlena později.

Tab. 6: Alokace jednotlivých portů mikrokontroleru

Periferie:	**Specifikace**	**Porty MCU**
Displej	Datová sběrnice: DB7-DB4	PORTA.3-PORTA.0
	Řídící vstupy: E, RS	PORTE.1, PORTE.0
Tlačítka	"Nahoru"	PORTC.7
	"Dolu"	PORTC.6
	"Menu"	PORTC.5
	"Start/Stop"	PORTE.2
ICSP konektor	PGC	PORTB.6
	PGD	PORTB.7
Vstup TIMER1	Výstup syntézy	PORTC.0
Výstup Hradlo 1	6	PORTC.1
Výstup Hradlo 2	2	PORTC.2
Výstup dělička_1	J1-J4	PORTD.0-PORTD.3
	J5-J8	PORTD.4-PORTD.7
Výstup dělička_2	J1-J4	PORTB.0-PORTB.3
	J5-J8	PORTB.4-PORTB.7
I^2C sběrnice	SCK	PORTC.3
	SDL	PORTC.4
Nevyužité porty		PORTA.4, PORTA.5

Signály obou výstupů frekvenčních syntéz je nutné upravit, tak, aby byly v protitaktu a obsahovali požadovaný počet impulzů. O rozdělení signálů na dva protitaktní se stará jednoduchá logika tvořená dvěma AND hradly. Ty jsou složeny ze čtveřice NAND hradel, integrovaného obvodu IC4. Funkce je daná programem, ten nejprve odblokuje první hradlo a čítá impulzy na výstupu. Po dobu odblokování prvního hradla logickou jedničkou na PORTC.1 se signál ze syntézy připojené na vývod 5 IC4 kopíruje na výstup prvního AND hradla (vývod 10 IC4). Po celou tuto dobu je druhé hradlo zablokované a signál z druhé syntézy neprochází blokovaným hradlem na výstup. Po načítání požadovaného počtu impulzů se úloha obou hradel vzájemně prohodí.

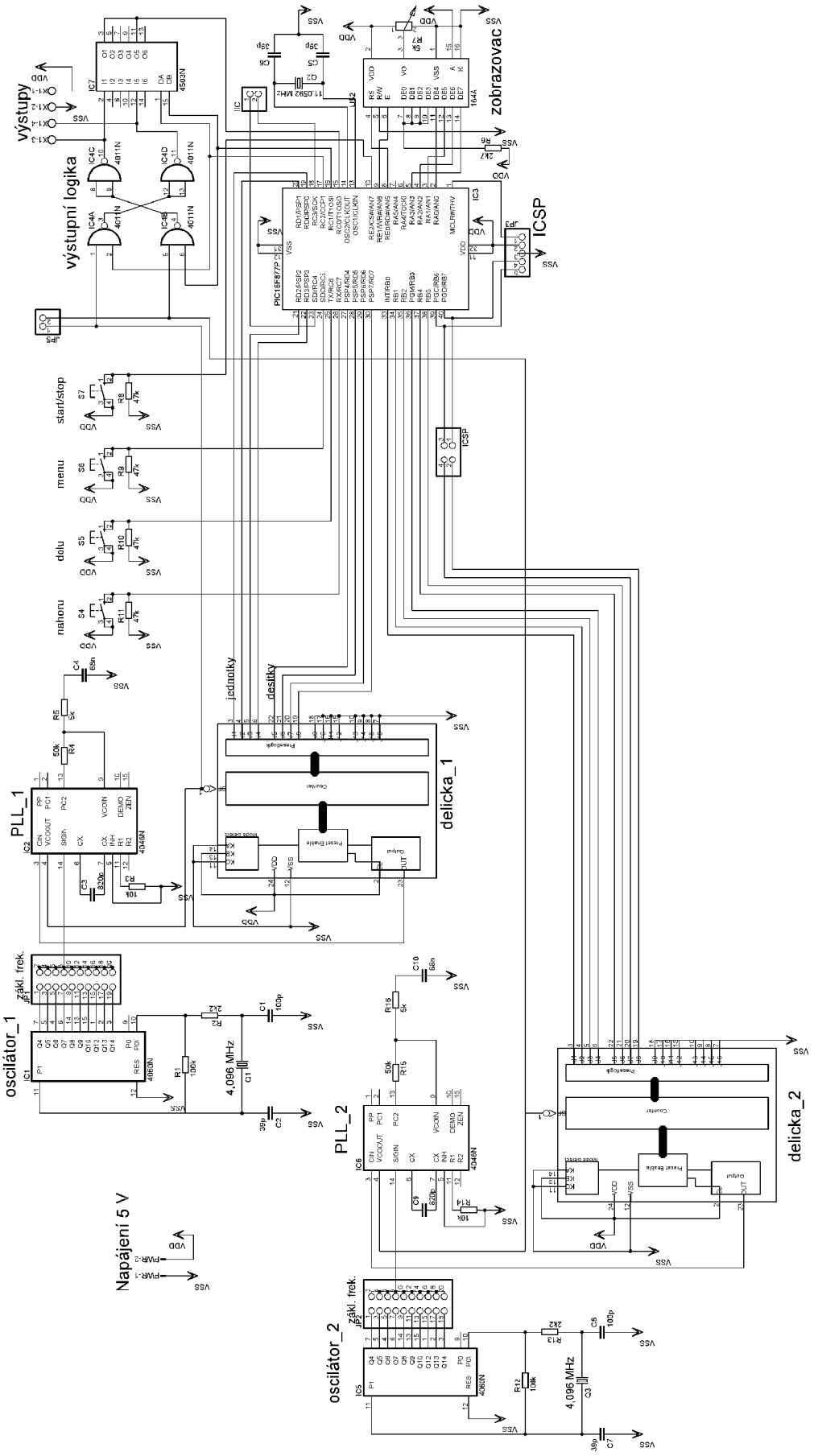
Jako čítač impulzů slouží časovač mikrokontroleru TIMER1 nastavený do režimu čítání. TIMER1 je jediným čítačem / časovačem mikrokontroleru, který dokáže pracovat asynchronně (nezávisle na taktovací frekvenci mikrokontroleru). Jeho vstup pro externí signál je připojen na PORTC.0. Aby pomocí něj bylo možné čítat impulzy ze dvou výstupů, aniž by se vzájemně ovlivňovaly, bylo nutné signály z výstupů přivést přes třístavový buffer IC7. Integrovaný obvod IC7 4503 obsahuje celkem šest bufferů s třístavovým výstupem. Ty lze uvést do stavu vysoké impedance na výstupu pomocí vstupů DA a DB. Přičemž, logická jednička na vstupu DA, uvede do vysoké impedance výstupy O1 až O4 a logická jednička na

vstupu DB, uvede do stavu vysoké impedance výstupy O5 a O6. Na schématu obrázku 17 je IC7 zapojen tak, že aktuálně nepoužívaný výstup čítání impulzů je uveden do stavu vysoké impedance, aby neovlivňoval ten aktuálně používaný.

K mikrokontroleru je také připojený konektor pro sériové programování ICSP a bity 3,4 PORTC jsou rezervovány pro připojení dalších obvodů pomocí sběrnice I^2C.

Mikrokontroler PIC16F877A není vybaven vnitřním oscilátorem a proto musí být použit externí oscilátor. Osobně využívám krystalový oscilátor s rezonanční frekvencí 11,0592 MHz. Tato hodnota udává taktovací frekvenci, která je odlišná od frekvence, s jakou jsou vykonávány jednotlivé instrukce. Jejich frekvence je čtyřikrát nižší.

Tím je popis obvodové části ukončen, další kapitola bude věnována ovládání generátoru a následně tvorbě programové části.



Obr. 17: Schéma generátoru – druhá verze

5.2 Uživatelské prostředí generátoru pro magnetronové naprašování

Obsluha celého zařízení bude prováděna pouze čtyřmi tlačítky a veškeré stavy budou zobrazeny na čtyřřádkovém displeji 4x16 znaků. Zařízení bude pracovat ve dvou režimech. Prvním je režim nastavení pracovních parametrů, tento režim se automaticky spustí po zapnutí zařízení. Druhým je pracovní režim. V režimu nastavení parametrů se pouze nastavují parametry, které se uloží do definovaných registrů. Během doby nastavení jsou obě výstupní hradla zablokována a na výstupech generátoru nejsou žádné signály. Displej v tuto chvíli zobrazuje každý parametr jako samostatnou položku displeje. Celkem jsou (v tuto chvíli, později budou rozšířeny i o nastavení střídy) implementovány čtyři volby. Nastavení frekvence f1, f2 a nastavení počtu impulzů I1, I2. Mezi jednotlivými položkami je možné přecházet stisknutím tlačítka „menu“. Nastavení parametrů se v každé položce provádí tlačítky označenými „nahoru“ a „dolů“. Krátkým stisknutím tlačítka nahoru se zvýší hodnota frekvence, či počet impulzů o jedničku. Dlouhým stiskem tlačítka se frekvence, či počet impulzů zvýší o pět. Analogická je situace u tlačítka dolů, zde platí to samé pro snižování hodnot. Funkce jednotlivých tlačítek jsou uvedeny v následující tabulce 7.

Tab. 7: Funkce jednotlivých tlačítek

Tlačítko	**Stisknuto**	**Funkce**
Nahoru	krátce	zvýší frekvenci / impulzy o 1
	dlouze	zvýší frekvenci / impulzy o 5
Dolu	krátce	sníží frekvenci / impulzy o 1
	dlouze	sníží frekvenci / impulzy o 5
Menu	Slouží k přechodu mezi jednotlivými položkami v režimu nastavení.	
Start / stop	Přepíná mezi režimem nastavení a pracovním režimem.	

Vizuální podoba jednotlivých položek v režimu nastavení je uvedena v další tabulce 8. První řádek displeje u každé položky zobrazuje, kde se nacházíme a kolik zbývá do konce. Druhý řádek říká, co se má nastavit. Třetí řádek displeje je bez využití, zobrazuje pouze dělicí čáru, která je zde kvůli grafickému sjednocení. Na čtvrtém řádku je zobrazována vždy nastavená hodnota včetně jednotek. Mezi jednotlivými položkami se přechází stisknutím tlačítka „menu“.

Tab. 8: Vizuální podoba jednotlivých položek zobrazených na displeji

Řádek	Text
1.	MENU 1/4
2.	Nastav frekvenci
3.	----------
4.	f1 = xx,x kHz

Řádek	Text
1.	MENU 2/4
2.	Nastav frekvenci
3.	----------
4.	f2 = xx,x kHz

Řádek	Text
1.	MENU 3/4
2.	Nastav impulzy
3.	----------
4.	I1 = xx cyklu

Řádek	Text
1.	MENU 4/4
2.	Nastav impulzy
3.	----------
4.	I2 = xx cyklu

Stisknutím tlačítka „start / stop“ se aktivuje druhý pracovní režim. V tomto režimu již není nutné nastavovat parametry. Generátor se takto aktivuje a na jeho výstupech je signál s nastavenými parametry. Veškeré tlačítka kromě „start / stop“ jsou neaktivní a na displeji se zobrazují nastavené parametry viz tabulka 9. Prozatím je využitý pouze první řádek, pro zobrazení obou frekvencí bez jednotek a druhý řádek pro nastavení počtu impulzů. Zastavení generátoru se provádí opětovným stiskem tlačítka start / stop. Tím se zablokují obě hradla a generátor přejde zpět do režimu nastavení.

Tab. 9: Vizuální podoba displeje v pracovním režimu

Řádek	Text
1.	f1=xx,x f2=xx,x
2.	I1=xx I2=xx

5.3 Programové řešení generátoru pro magnetronové naprašování

Celý zdrojový kód pro mikroprocesor jsem se rozhodnul napsat v jazyku C. Vyšší programovací jazyk má oproti jazyku symbolických adres několik výhod, ale také nevýhod. Za hlavní výhody považuji přehlednost kódu a snazší dosažení cíle, tj. výsledného programu. Zařízení bude určeno pro použití v laboratoři fakulty, kde bude součástí většího celku. Je

velmi pravděpodobné, že bude nutné v průběhu času dělat určité úpravy, jako je změna, či rozšíření rozsahů, doplnění o nové funkce ap. Pro konstruktéra, který bude tento problém řešit, by měla být orientace ve zdrojovém programu napsaném v jazyce C s použitím dokumentace uvedené v této práci relativně jednoduchou záležitostí. Na rozdíl od jazyka symbolických adres, kdy je po čase i pro samotného návrháře, často problematická orientace ve vlastním programu. Vyšší přehlednosti programu je dosaženo také jeho členěním do více souborů, kdy každý soubor náleží k určité fázi programu. Soubory tvořící celý projekt jsou celkem čtyři. Zdrojovým souborem je main. c, dalšími soubory jsou lcd.c, menu.c a pracuj.c. Výpis všech těchto programů je součástí příloh, přesto zde musím podotknout, že jediným způsobem, jak lze přehledně studovat kód všech souborů je otevření celého projektu v programu MPLAB IDE.

K překladu z jazyka C do jazyka symbolických adres využívám kompilátor CC5X Version 3.3A společnosti B Knudsen Data. Detaily ohledně překladače CC5X lze dohledat v jeho manuálu, viz [15]. Jeho výhodou je úsporný překlad z vyššího programovacího jazyka. Kód lze s jeho pomocí také optimalizovat a ještě více zredukovat jeho velikost. Osobně tuto funkci nevyužívám, protože zde hrozí riziko vzniku chyby vlivem nevhodné optimalizace. Optimalizace je implicitně zapnuta, vypnutí se provádí direktivou **#pragma optimize 0**. Výsledný kód může mít až 8192 instrukcí, což je celková velikost paměti programu mikrokontroleru. Celá paměť je rozdělena na čtyři stránky (Code Pages) po 2048 instrukcích. Funkce main() začíná automaticky na stránce 0. Je-li překročena její velikost, je nutné rozčlenit ručně kód do dalších stránek direktivou **#pragma codepage 1,2** nebo **3**. Direktiva se vkládá před definovanou funkci, případně před prototyp funkce. Tento proces je nutné důkladně promyslet, protože je-li funkce komplexní a ve svém těle odkazuje na další funkce, jsou tyto funkce také umístěny ve stejné bance. To může způsobit duplikace. Podmínkou je, aby byly všechny funkce v jedné bance. Nyní můžeme přejít k popisu jednotlivých programových větví, tak jak jsou uvedeny v samostatných souborech.

5.3.1 Zdrojový soubor - main.c

Běh programu začíná ve zdrojovém souboru **main.c**. V samém začátku programu main.c využívám direktivy překladače. Konfigurační pojistky mikrokontroleru jsou uvedeny v poznámce a mají pouze informativní charakter, aby každý věděl, jaké má být výchozí nastavení mikrokontroleru. Pomocí dalších direktiv se nastavuje následující:

- #pragma optimize 0 – vypnutí optimalizace kompilátoru
- #include "int16CXX.h" – zahrnutí tohoto souboru je nutné k využití systému přerušení
- #pragma origin 4 – nastavuje adresu přerušení, opět nutné k jeho chodu

Další řádky definují názvy jednotlivých bitů portu, kvůli jejich snadné identifikaci, při psaní programu. Dále následují prototypy funkcí.

Vůbec první definovanou funkcí je **interrupt serverX(void)**. Tato funkce má na starost obsluhu přerušení a musí být povinně definována jako první. Teprve až za ní, mohou být direktivy #include, říkající, které další soubory jsou součástí projektu.

Následuje definice dvou velmi často využívaných funkcí **delay()** a **super_delay()**. Jedná se o velmi jednoduché procedury, které zavoláním s celočíselným argumentem provádí dekrementaci tak dlouho, jak je velké číslo v argumentu a takto generují zpoždění. Časové zpoždění je závislé na taktu mikrokontroleru a jeho hodnoty jsou ve zdrojovém kódu uváděny v poznámkách.

Další řádky definují inicializační funkce, které se používají v celém programu pouze jednou. Důvodem, proč jsou uvedeny coby samostatné funkce a nikoliv jednotlivé příkazy je přehlednost. První funkcí je **nastav_preruseni()**, v jejím těle se provádí zápisem jednotlivých bitů aktivace přerušení vyvolané přetečením čítače TIMER1. Nastavení samotného čítače je provedeno další funkcí **init_citac()**. Poslední funkce nazvaná **inicializace()** provádí inicializaci displeje, prostřednictvím funkce **nastav_LCD()**, která bude popsaná dále, inicializaci tlačítek, volá se funkce **nastav_preruseni(), init_citac()** a do registrů n_1, n_2, i_1, i_2 se ukládají počáteční hodnoty. Dovoluji si podotknout, že doposud se všechny funkce pouze definovaly.

První volanou funkcí je **main()**, jejíž první příkaz (instrukce) je zároveň i prvním příkazem celého programu. Tato funkce je povinnou součástí zdrojového soboru. V jejím těle probíhá nejprve nastavení jednotlivých portů na vstupní a výstupní a poté se volá inicializační funkce **inicializace()**. Jejím prostřednictvím se volají všechny ostatní inicializační rutiny. Po jejich provedení program volá funkci **menu_1()**. Tím začíná běh hlavní smyčky programu, které je věnován samostatný soubor a je popsán až v kapitole 5.3.3. Nejprve musím provést rozbor souboru lcd.c náležícího obsluze inteligentního zobrazovače. Kapitola 5.3.3 je věnovaná menu a zobrazení, proto je vhodné se nejprve seznámit s funkcemi využívající zobrazovač.

5.3.2 Programová obsluha inteligentního zobrazovače – lcd.c

Celému programu pro obsluhu displeje je v projektu věnován samostatný soubor lcd.c. Jeho kompletní výpis je uveden v příloze. Program je rozčleněn do několika funkcí, které zde stručně popíši.

V kapitole věnované elektrickému návrhu je uvedeno, že u zobrazovače používáme čtyřbitovou komunikaci s mikrokontrolerem. Celkem se tedy využívají čtyři datové vodiče a dva řídící. Jednotlivé bity portu mikrokontroleru odpovídající řídícím vodičům jsou pojmenovány E a RS. Kvůli snadné orientaci je symbolika totožná s označením vodičů

zobrazovače v katalogovém listě. Protože PORTA, na který jsou připojeny datové vodiče má více bitů, než je počet datových vodičů, je nutné programově ošetřit, aby se data posílala pouze na připojené datové vodiče. Tato programátorská technika se označuje, jako maskování nevyužívaných bitů. U čtyřbitové komunikace je dále nutné osmibitové slovo rozdělit na dvě slova čtyřbitová a ty následně poslat na PORTA (datové vodiče zobrazovače).

Vůbec první programovou rutinu, kterou je nutné vytvořit je funkce, jež zapíše na datové vodiče zobrazovače příkaz. V souboru lcd.c je tato funkce označena **prikaz()**. Argumentem funkce je poté binárně, či hexadecimálně zadaný příkaz. Funkci lze rozdělit na dvě velmi podobné části. První část oddělí z příkazu, jímž je osmibitové slovo, horní čtyři bity a zamaskuje nepoužité bity PORTA. Tohle vše se provede a uloží do proměnné „**zasobnik**“. Následně se data přenesou z mikrokontroleru do zobrazovače krátkou aktivací řídícího vstupu E. Zde začíná druhá část funkce, analogická s první, s tím rozdílem, že nyní jsou do proměnné „zasobnik“ uloženy dolní čtyři bity slova.

Z popisu inteligentního zobrazovače vyplývá, že pro jeho správnou funkci je nejprve nutné provést inicializaci. Během ní se nakonfigurují možnosti zobrazení a nastaví se typ komunikace. Inicializace se provádí zápisem série příkazů do zobrazovače, proto je nutné mít nejprve k dispozici funkci pro zápis příkazů. Funkce provádějící inicializaci je nazvána **nastav_LCD**. Během ní je nastaven typ komunikace (čtyřbitová), jsou zapnuty všechny čtyři řádky a displej je zapnut s kurzorem na první pozici. Po jejím provedení je displej připraven pro korektní zobrazení znaků.

Každý znak je, stejně jako příkaz, osmibitové binární slovo, jemuž odpovídá požadovaný znak. Vztah mezi binárním slovem a zobrazeným znakem je dán ASCII tabulkou. Funkce pro zobrazení jednoho znaku je pojmenována **znak()**. Argumentem je buď binární kód znaku nebo přímo znak ohraničený apostrofy před a za ním např. 'a'. Tělo funkce znak() je totožné s funkcí prikaz(). Jediným rozdílem je aktivace řídícího vstupu RS před zápisem horní i dolní poloviny slova.

Někdy je potřeba vypsat na displeji celé slovo, případně i několik slov. Bylo by zdlouhavé každý znak zapisovat voláním jedné funkce. Z toho důvodu je součástí souboru lcd.c funkce nazvaná **zobraz()**. Jejím argumentem může být textový řetězec uzavřený do uvozovek. Například, funkce zadaná kdekoliv v programu takto: **zobraz("frekvenci");** zobrazí na displeji slovo „frekvenci“ s počátkem slova na aktuálním místě kurzoru. Podstata funkce je založena na možnostech jazyka C. Proměnná, kterou voláme funkci je typu řetězec a ve funkci ji voláme odkazem na ni, tzv. pointerem (ukazatelem). Vypsání jednotlivých znaků na displeji je provedeno pomocí smyčky for, která cyklicky volá funkci znak(). Díky automatické inkrementaci adresy znaku, zapnuté během inicializace zobrazovače, se nemusíme během volání funkce znak() starat o následující adresu kurzoru. Přechod na další pozici displeje je čistě v režii elektroniky zobrazovače. Nutné je zadat vždy pouze adresu prvního znaku zobrazovaného textového řetězce, před samotnou funkcí zobraz().

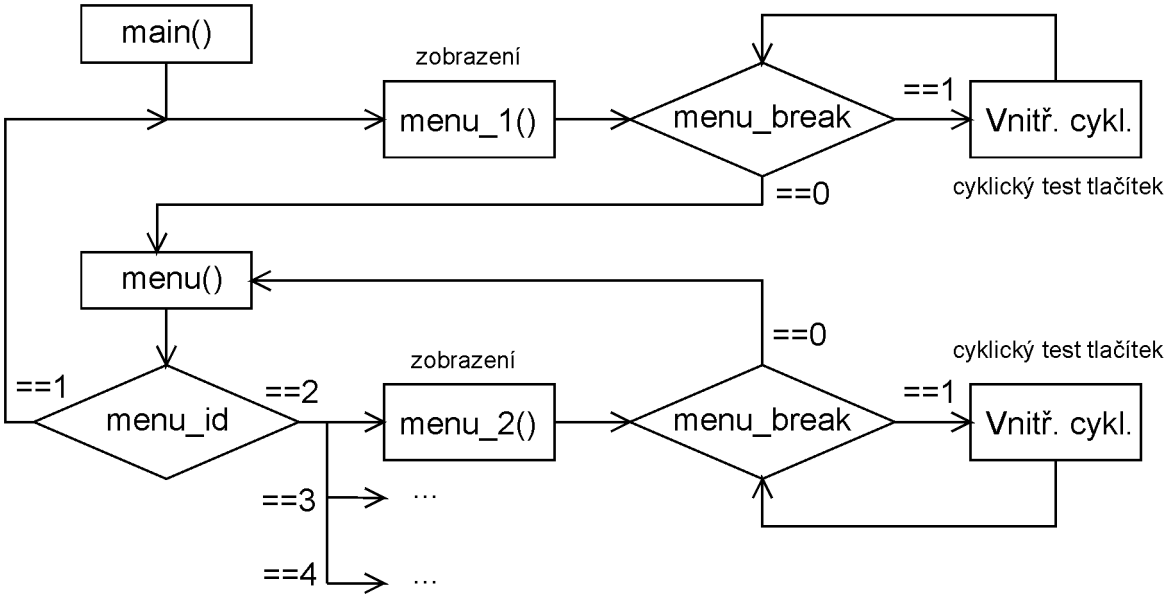
Adresa zobrazení na displeji se zadá pomocí příkazu začínajícího jedničkou, zbylých sedm bitů je potom adresa odpovídající rozložení na obrázku 14. Např. pro přechod kurzoru na druhý znak druhého řádku zavolám funkci takto: prikaz (0b11000001);.

V některých případech je vhodné využít všechny znaky displeje pro vypsání např. krátké věty. Abychom nemuseli často zadávat příkazy pro přechod na nové řádky je součástí obsluhy zobrazovače funkce **zapis_64_znaku()**. Touto funkcí je, za určitých podmínek, možné vypsat řetězec o délce až 64 znaků v kuse. Podmínkou je rozdělení slov do bloků po šestnácti znacích, aby nebyla jednotlivá slova půlená. Mám-li, například, zobrazit řetězec „Nastaveni frekvencniho generatoru“ musím po prvním slově provést doplněk do šestnácti pomocí mezer, aby se druhé slovo zobrazilo celé na dalším řádku. To samé platí i pro třetí slovo.

Posledními dvěma funkcemi, jejichž úkolem je zjednodušit tvorbu programu, jsou **vymaz()** a **domu()**. Funkce vymaz() provede smazání celého displeje tím, že na všechny adresy vloží znaky mezery a přejde na adresu 0x00. Funkce domu() provádí pouze přechod na adresu prvního znaku 0x00.

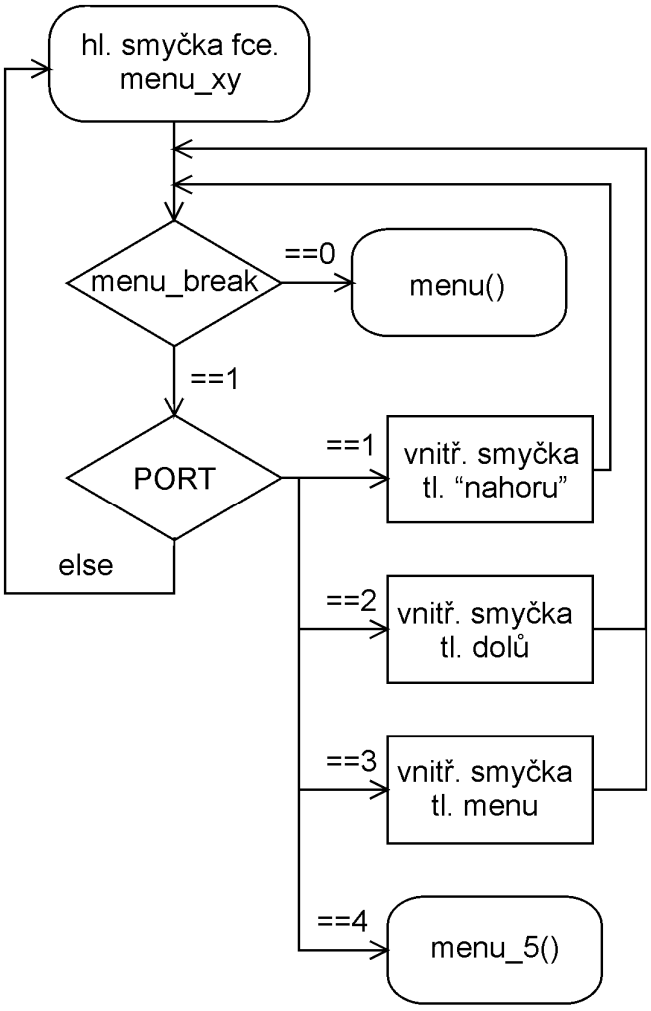
5.3.3 Program režimu nastavení generátoru - menu.c

Celý tento program má, mimo jiné, na starost kompletní uživatelské rozhraní popsané v kapitole 5.2. Konkrétně, vizuální podobu jednotlivých položek menu a funkce jednotlivých tlačítek. Stav tlačítek není snímán pomocí systému přerušení, ale je v krátkých intervalech kontrolován hlavní programovou smyčkou. Její páteř charakterizuje vývojový diagram na obrázku 18.



Obr. 18: Vývojový diagram hlavní smyčky programu souboru menu.c

Po vykonání všech příkazů funkce **main()** se zavolá funkce **menu_1()**. Tato funkce odpovídá zároveň první položce menu v režimu nastavení. Analogicky je tomu u funkcí **menu_2()**, **menu_3()** a **menu_4()** náležících k položkám menu uvedených za podtržítkem v názvu funkce. Nejprve se provede zobrazení parametrů (bude mu věnován další odstavec) na displeji odpovídající dané položce uvedené v kapitole 5.2. Poté se začne vykonávat vnitřní cyklus, ten je u těchto funkcí vždy stejný. Vnitřní cyklus představuje soubor podmínek if uzavřených do podmínky while, testující proměnou **menu_break**. Je-li její hodnota rovna jedné, probíhá cyklické testování stavu jednotlivých tlačítek. Větvení programu po stisku tlačítek je realizováno vícenásobnou podmínkou if. Na obrázku 18 není, kromě stisku tlačítka „menu“, podmíněné větvení uvedeno kvůli zjednodušení. Po stisku tlačítka „menu“ se do proměnné menu_break zapíše nula a neruší se podmínka while(menu_break==1). Program odchází z aktuální funkce menu_xy()[12] do funkce **menu()**. Této funkci se věnuje další odstavec. Větvení vnitřního cyklu je na vývojovém diagramu na obrázku 19.



Obr. 19: Vnitřní cyklus funkce menu_xy

[12] Symbol xy říká, že jde o jednu z funkcí menu_1(), menu_2(), menu_3() či menu_4(). Takto budu dále v textu označovat situaci, kdy hovořím o všech funkcích menu_1() až menu_4().

Při stisku tlačítka „nahoru“ nebo „dolů“ dochází k inkrementaci, či dekrementaci příslušného registru. Například v položce menu číslo jedna (menu_1) je to registr n_1. Po zapnutí se upravuje hodnota zadaná implicitně funkcí inicializace(). Obě tlačítka jsou dvoustavová, viz popis v kapitole 5.2. Stisknutí těchto dvou tlačítek nemá vliv na hlavní smyčku podmíněnou hodnotou proměnné menu_break. Hlavní smyčku lze přerušit pouze stisknutím tlačítka „menu“ nebo „start/stop“. Po stisku tlačítka „menu“ se změní hodnota **menu_break** na 0 a program přejde z funkce **menu_1()** do funkce **menu()**.

Funkce **menu()** testuje stisknutí tlačítka „menu“, v případě jeho stisku se provede inkrementace proměnné **menu_id**. Její hodnota udává položku menu, první položka menu má identifikátor menu_id=1, druhá menu_id=2 atd. Proto se například po stisku tlačítka „menu“ v první položce provede inkrementace na hodnotu dvě během funkce **menu()** a program přejde na funkci **menu_2()** viz obrázek 18. Funkce **menu_2()** je po programové stránce zcela totožná s předchozí funkcí menu_1 i s následujícími funkcemi menu_3() a menu_4(). Jediným rozdílem je aktualizace jiného registru, v tomto případě f_2 po stisku tlačítka „nahoru“ či „dolů“.

Každá z funkcí menu_1() až menu_4() má za úkol, mimo testování stavu tlačítek, provést zobrazení nastavených parametrů na displeji zobrazovače. Zobrazované položky lze rozdělit na dva typy. Prvním typem jsou položky statické, jejich text zůstává v dané položce menu neměnný. U všech čtyř položek menu jsou to první tři řádky displeje. Zobrazení těchto řádků probíhá vždy ihned po zavolání funkce menu_xy(), pomocí sekvence funkcí definovaných v souboru lcd.c. Druhým typem jsou položky dynamické, měnící se i v rámci dané položky menu. U všech čtyř položek menu je to vždy čtvrtý řádek zobrazující nastavenou hodnotu příslušného parametru. Přesto, že se z celého řádku mění pouze číselná hodnota, je vhodnější zobrazit vždy znovu celý řádek. Zavoláme-li jednu z funkcí menu_xy() zobrazí se nejprve tři statické textové řetězce a poté i čtvrtý, dynamický řádek. Při prvním zobrazení odpovídá jeho číselná hodnota hodnotě implicitní. K přepsání tohoto řádku dochází vždy po stisknutí tlačítka „nahoru“ nebo „dolů“.

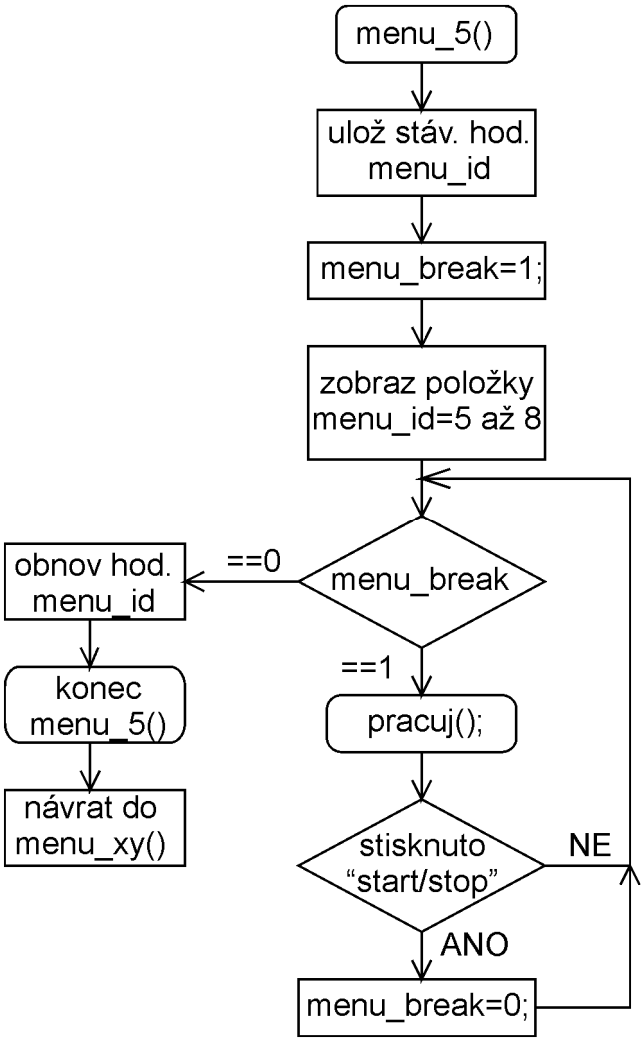
Dynamický text je zobrazován pomocí funkce **displej(),** jenž je volána vždy s argumentem odpovídající zobrazované veličině. Funkce je využívána všemi čtyřmi položkami menu v režimu nastavení, ale i funkcí **menu_5()** zobrazující parametry v pracovním režimu. Aby funkce displej() věděla, jaký text se má zobrazit kromě číselné hodnoty v argumentu, využívá opět globální identifikátor položky menu **menu_id**. Na základě její hodnoty probíhá větvení pomocí podmínky switch. Každá větev této funkce je unikátní pro danou položku menu. Problém, který bylo nutné vyřešit u zobrazení dynamických parametrů, spočíval v převodu čísla z určitého registru na jednotlivé znaky na displeji. Protože, například decimální číslo 5 v ASCII tabulce, se kterou pracuje zobrazovač, rozhodně neodpovídá znaku pětky. Musel jsem tedy vytvořit funkci pro převod číselných hodnot na ASCII kódy. Tato funkce je nazvaná **ASCI()** a pracuje tak, že se zavolá s číselným argumentem a vrátí hodnotu argumentu v ASCII kódu. Tělo funkce je velmi jednoduché a

pracuje s vícenásobnou podmínkou typu switch. Číselný rozsah této funkce je 0-9. Další problém nastává při zobrazení čísla většího, než je 9. Například, desítka má hodnotu deset, ale na displeji se zobrazuje jako jednička a nula, tedy dvě samostatná čísla se samostatnými ASCII kódy. Problém jsem vyřešil převodem čísla na kód BCD. Při BCD kódování jsou jednotky řazeny jako první čtyři bity, desítky jako bity 5-8, stovky jako bity 9-12 atd. Mám-li číslo rozdělené do skupin po čtyřech bitech, není již problém je od sebe oddělit maskováním a rotací. Za účelem převodu decimálního čísla na BCD je definována funkce **DEC2BCD**(). Tělo této funkce je opět velmi jednoduché, jde o implementaci vztahu (1), používaného pro převod z dvoumístného desítkového čísla na kód BCD.

(1) $$BCD = \frac{x\%100}{10} \cdot 6 + x$$ Kde x je číslo, které chceme převést na kód BCD.

Zobrazení dynamických textů je tedy realizováno s pomocí funkcí ASCII(), DEC2BCD() a displej(). Funkce displej využívá dále funkce pro obsluhu zobrazovače, definované v souboru lcd.c. K zobrazení statických textových informací stačí využít pouze základní funkce definované v souboru lcd.c.

Nyní přejdu k popisu dalšího větvení programu, po stisku tlačítka „start/stop“. Tlačítko „start/stop“ lze stisknout v kterékoliv položce menu odpovídající režimu nastavení. Také je možné jej stisknout hned po zapnutí generátoru, bez nastavení jediného parametru, protože po zapnutí jsou v jednotlivých registrech uloženy implicitní hodnoty. Stisknutím tlačítka „start/stop“ program přechází z hlavní smyčky aktuální funkce menu_xy na funkci **menu_5()**. Funkcí menu_5() přechází generátor z režimu nastavení do pracovního režimu, i když v průběhu této funkce jsou výstupní hradla stále ještě zablokovaná. Pracovní režim začíná vypsáním všech nastavených hodnot na displeji a teprve poté je započato generování signálu. Důvodem, proč hodnoty nejsou zobrazovány během pracovního režimu, je požadavek na minimální časové zatížení mikrokontroleru. Běh programu ve funkci menu_5() je patrný z vývojového diagramu na obrázku 20.



Obr. 20: Vývojový diagram funkce menu_5()

Nejprve se provede uložení aktuální hodnoty menu_id, protože se s touto hodnotou bude dále pracovat, kvůli zobrazení jednotlivých hodnot registrů pomocí funkce displej(). V dalším koku se nastaví hodnota menu_break na jedničku, a postupně se volá funkce displej(), argumentem jsou jednotlivé registry f_1, f_2, i_1, i_2. Před každým voláním se změní hodnota globálního parametru menu_id. Nyní jsou na displeji zobrazeny nastavené parametry a spouští se smyčka podmíněná hodnotou menu_break, která je nastavena do jedničky. Cyklus smyčky střídavě volá funkci pracuj() a testuje stisknutí tlačítka „start / stop“. Funkce **pracuj()** vykoná vždy jeden pracovní cyklus obou výstupů[13]. Její rozbor bude proveden v následující kapitole. Je-li v pracovním režimu stisknuto tlačítko „start/stop“ nastaví se hodnota menu_break na nulu, čímž se ukončí podmínka zastřešující cyklické volání funkce pracuj(). Po ukončení funkce menu_5() se program vrací zpět do té položky menu, ze které byl volán.

[13] Pracovním cyklem výstupu je míněno generování zadaného počtu impulzů u obou výstupů.

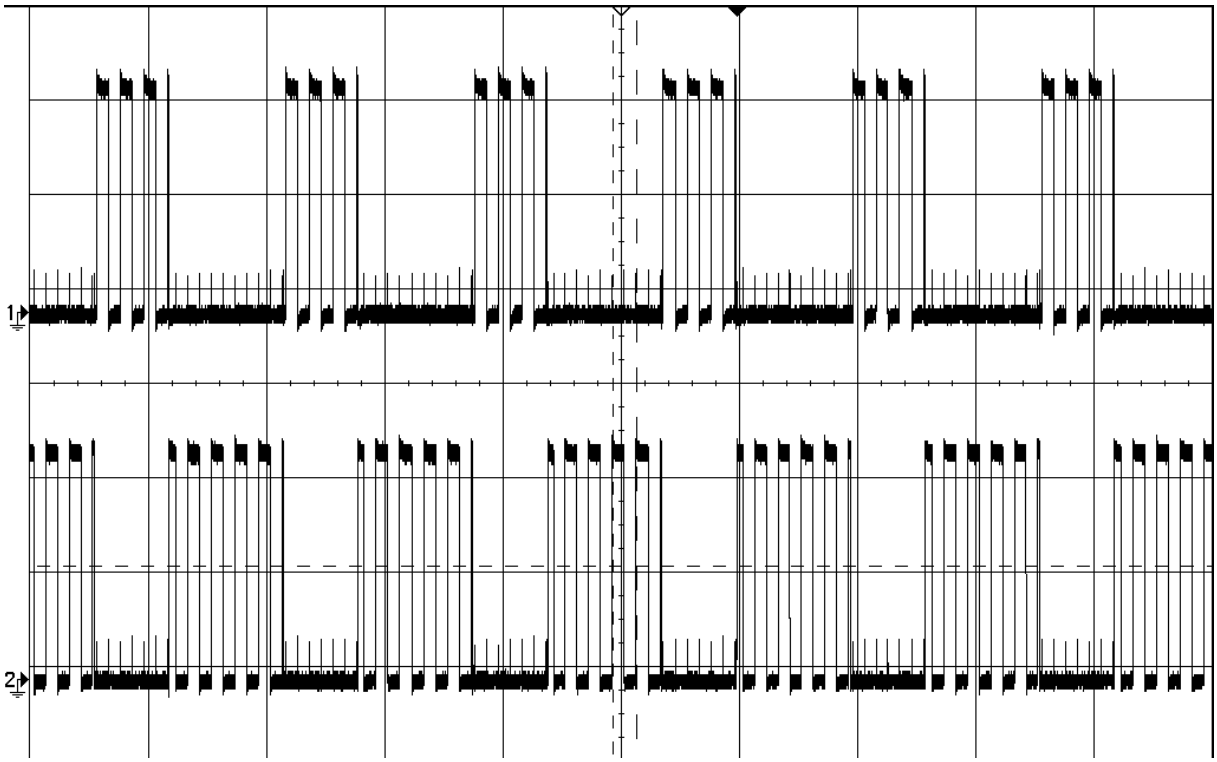
5.3.4 Program pracovního režimu generátoru – pracuj.c

Soubor s názvem pracuj.c obsahuje všechny funkce, které jsou nutné pro chod generátoru v pracovním režimu. Po stisknutí tlačítka „start / stop“ program přejde do funkce menu_5(), a během ní se cyklicky volá funkce **pracuj()** až do opětovného stisknutí tlačítka „start / stop“. Aby funkce pracuj() správně fungovala, musí se před jejím voláním nejprve zavolat funkce **nastav_delicky(n_1,n_2,i_1,i_2)**. V rámci této funkce se převedou čísla uložená v proměnné n_1, n_2 na BCD kód (protože vstupy děličky pracují s čísly v kódu BCD) a zapíší se na PORTD a PORTB. Od této chvíle jsou děličky nastaveny na frekvence odpovídající součinu základní frekvence a čísel uložených v proměnné n_1, n_2. Funkce nastav_delicky(n_1,n_2,i_1,i_2) dále uloží do proměnných impulzy_1, impulzy_2 počet impulzů v jednotlivých pracovních periodách obou výstupních signálů. Volání funkce nastav_delicky(n_1,n_2,i_1,i_2) předchází záměrně volání samotné funkce pracuj(), protože během ní (v průběhu pracovního cyklu) je nežádoucí vykonávat jakékoliv instrukce, které mohou být vykonány mimo pracovní cyklus. Podotýkám, že převod dvoumístného decimálního čísla na kód BCD je časově relativně náročný proces. Konkrétně je nutné vykonat asi 550 instrukcí odpovídajících času 0,2 ms, což není rozhodně zanedbatelné. Funkce pracuj() je napsaná s ohledem na maximální časovou úspornost. Během ní jsou čítány impulzy a na základě nich jsou řízena spouštění výstupních hradel. Celou funkci lze rozdělit na dvě pracovní oblasti odpovídající výstupu S1 a S2. První pracovní oblast začíná nastavením hodnoty odpovídající počtu impulzů na prvním výstupu S1 do registru čítače TIMER1. Následně je zapnuto čítání a odblokováno hradlo_1. Hradlo je odblokováno až do doby přetečení čítače, tím se vyvolá přerušení, hradlo_1 se zablokuje a vypne se čítání. Tím je ukončena první pracovní oblast a začíná druhá pracovní oblast, zcela totožná s první. Jediným rozdílem je, že se pracuje s hradlem_2. Funkce pracuj() je poslední popisovanou funkcí a končí poslední podkapitolu věnovanou programovému řešení. Výpis celého programu mikrokontroleru je uveden v přílohách. Další kapitola bude věnována ověření funkce sestaveného generátoru.

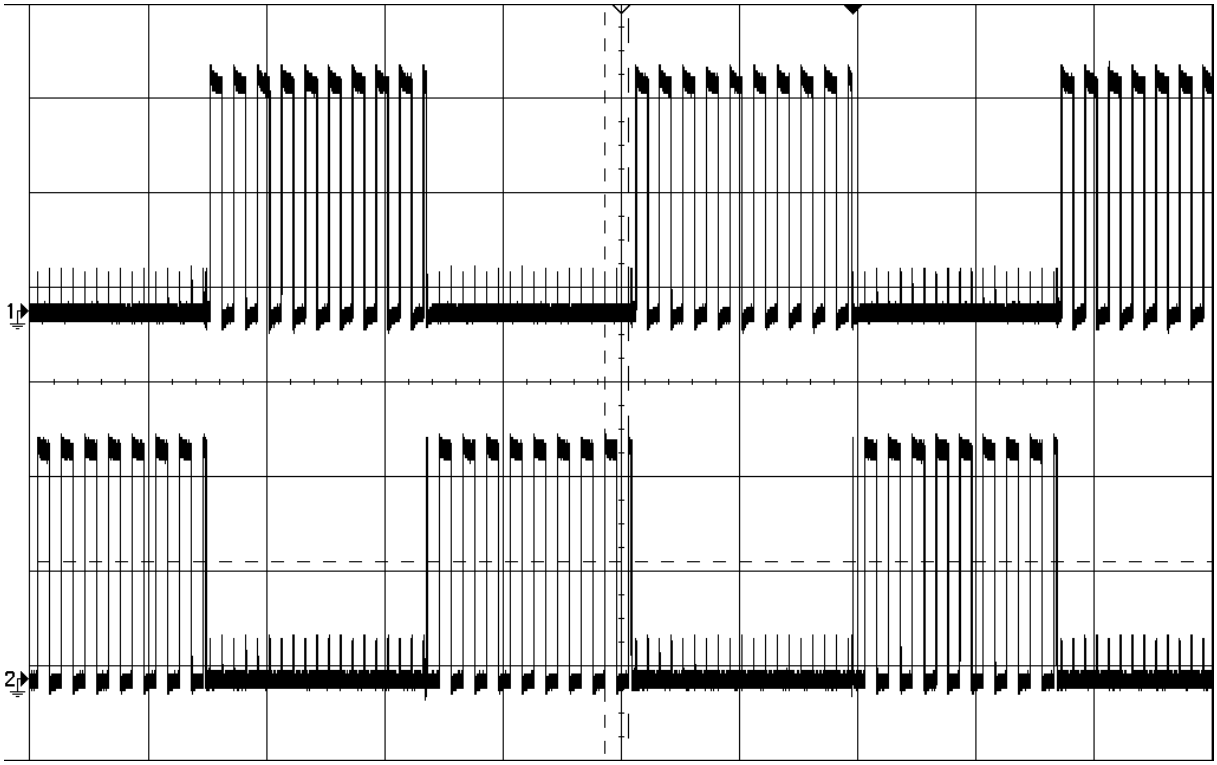
5.4 Ověření funkce generátoru pro magnetronové naprašování

Generátor byl sestaven na základě návrhu podrobně popsaného v jednotlivých kapitolách 5. Součástí byl i návrh plošného spoje, uvedený v přílohách, na kterém bylo celé zařízení realizováno. Plošný spoj je prozatím experimentální verzí, protože se předpokládají další úpravy a zdokonalení. Navíc během samotného oživení bylo opět odstraněno několik chyb, které nemá smysl popisovat, ale jsou již zahrnuty v návrhu. Experimentální verze je sestavena z klasických vývodových součástek, protože vzhledem k jejich velikosti se s nimi lépe pracuje při odstraňování problémů. Kromě samotných součástek jsou na plošném spoji napájena i všechna tlačítka a zobrazovač, opět kvůli pohodlí při vývoji. Protože se počítá s využitím laboratorního zdroje, k napájení generátoru, není na desce plošného spoje žádný stabilizátor, ani filtrační kondenzátory.

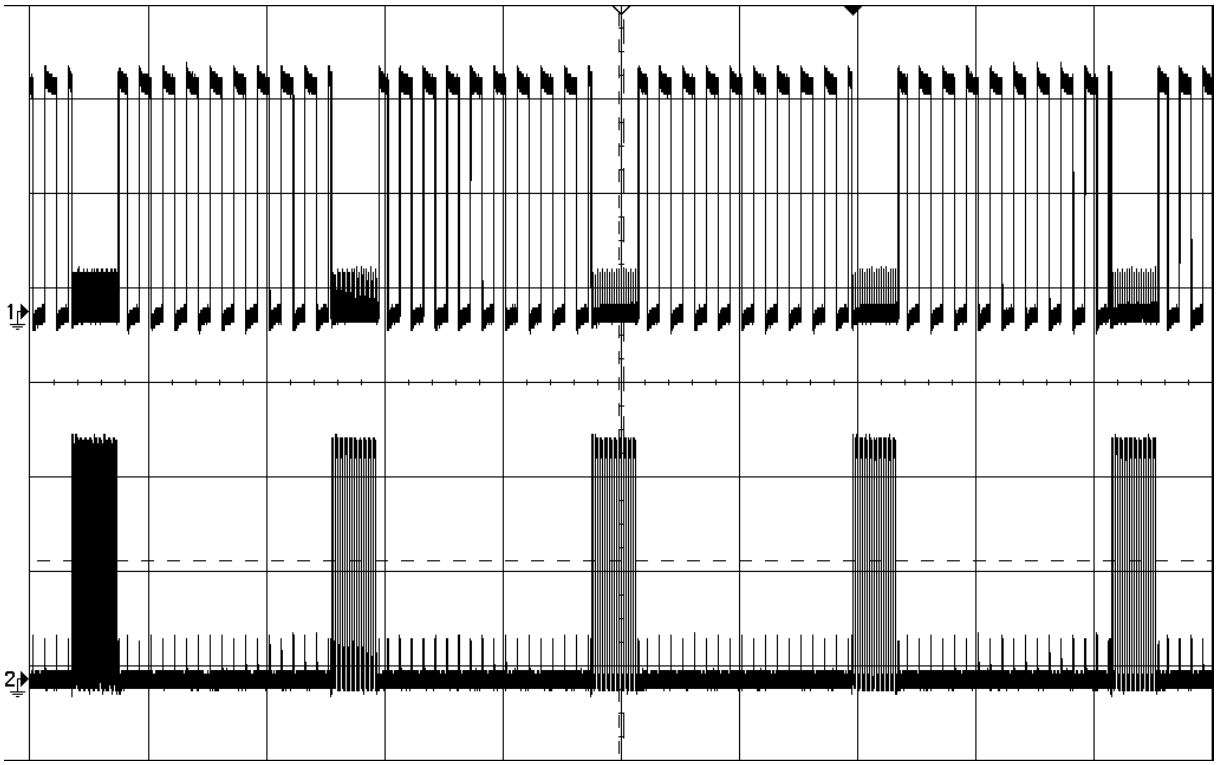
Ověření funkce spočívá v měření na osciloskopu, které by mělo prokázat, že generátor pracuje podle předpokladů. Na jeho výstupech by měly být signály o parametrech nastavených na displeji zobrazovače. Na následujících obrázcích jsou průběhy uložené přímo z digitálního osciloskopu. Na těchto obrázcích je vždy první průběh (horní) signálem S1, kterému odpovídá frekvence f1 a počet impulzů I1. Podobně druhý průběh (dolní) je signálem S2 s frekvencí f2 a počtem impulzů I2. Obrázek 21 ukazuje nastavení na nejnižší frekvenci 5 kHz s čtyřmi a šesti impulzy. Na obrázku 22 je nastavení generátoru na 10 kHz s deseti impulzy v každé periodě. Obrázek 23 ukazuje asymetrické nastavení s frekvencemi 10 kHz, 50 kHz a deseti impulzy v každé periodě, na obrázku 24 a 25 je to samé zobrazení v detailu. Na obrázku 26 je nastavení s frekvencemi 67 kHz, 90 kHz a počtu impulzů 15, 20. Obrázky 27 a 28 ukazují nastavení na vysokých frekvencích 60 kHz, 90 kHz a velkým počtem impulzů, konkrétně 20 a 60.



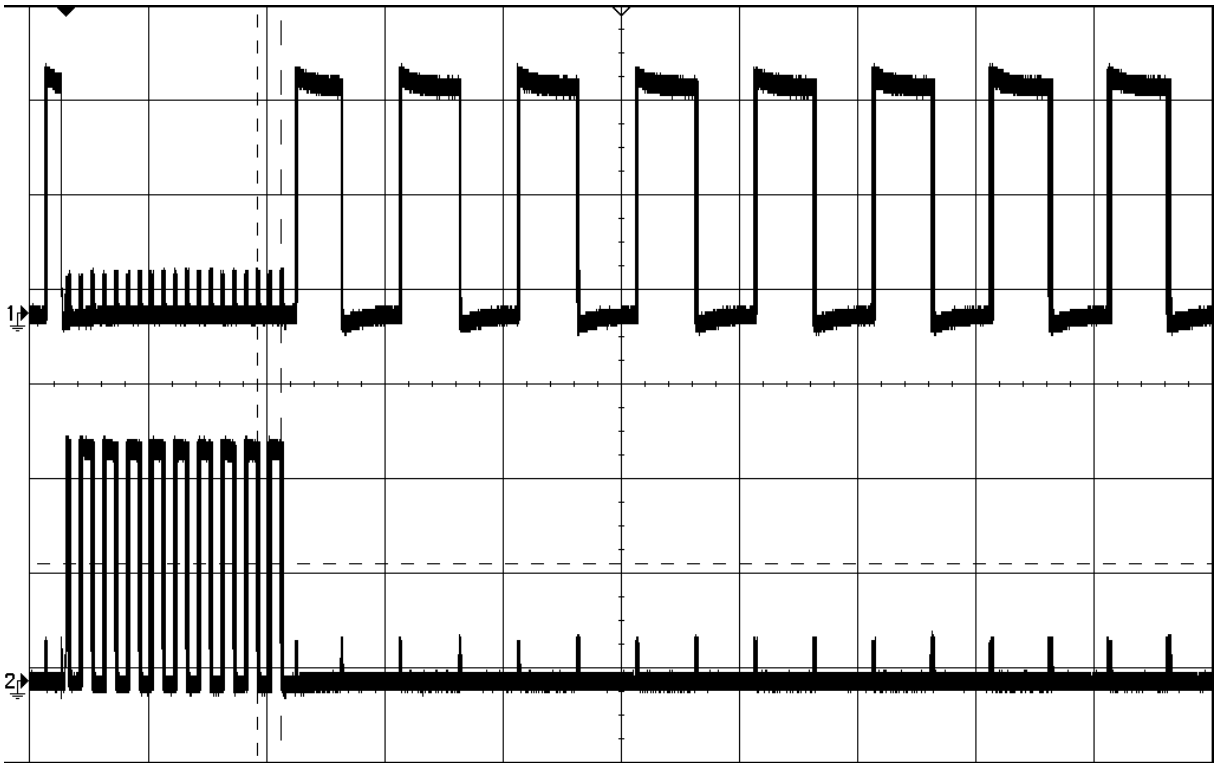
Obr. 21: Průběhy generátoru f1 = f2 = 5 kHz, 4 a 6 impulzů



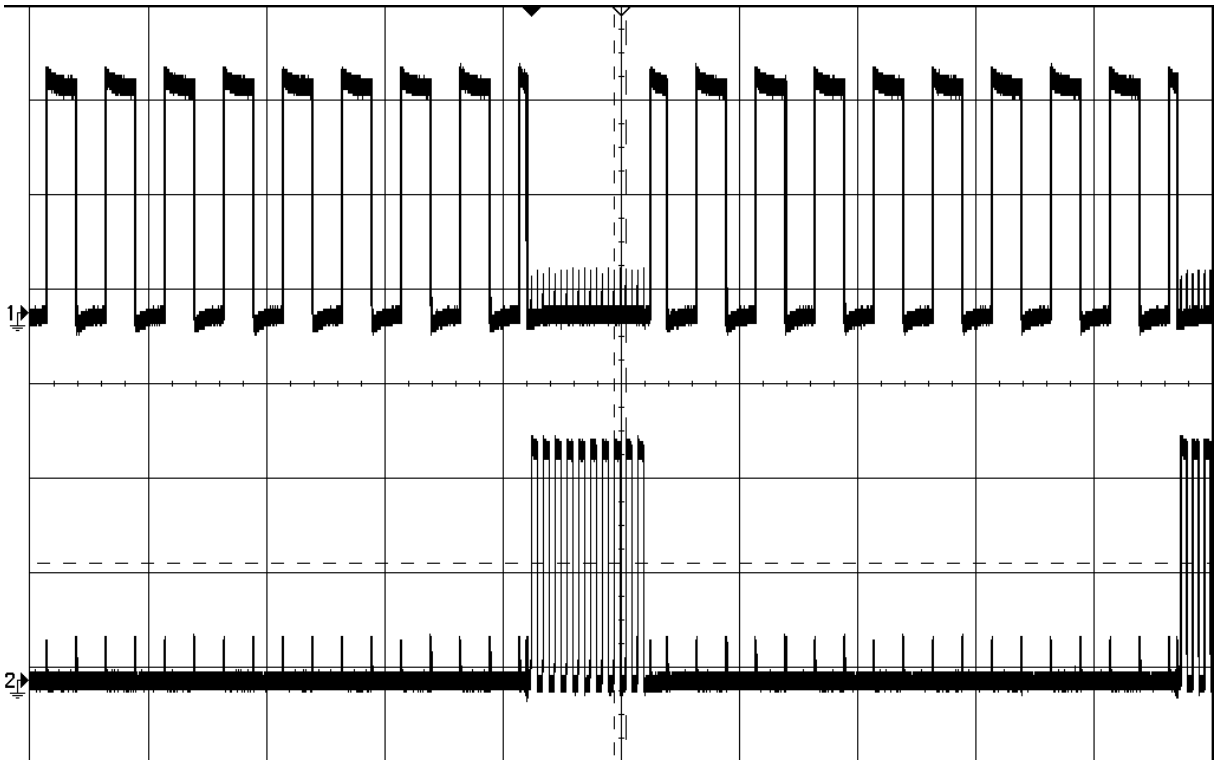
Obr. 22: Průběhy generátoru f1 = f2 = 10 kHz, 10 impulzů v každé periodě



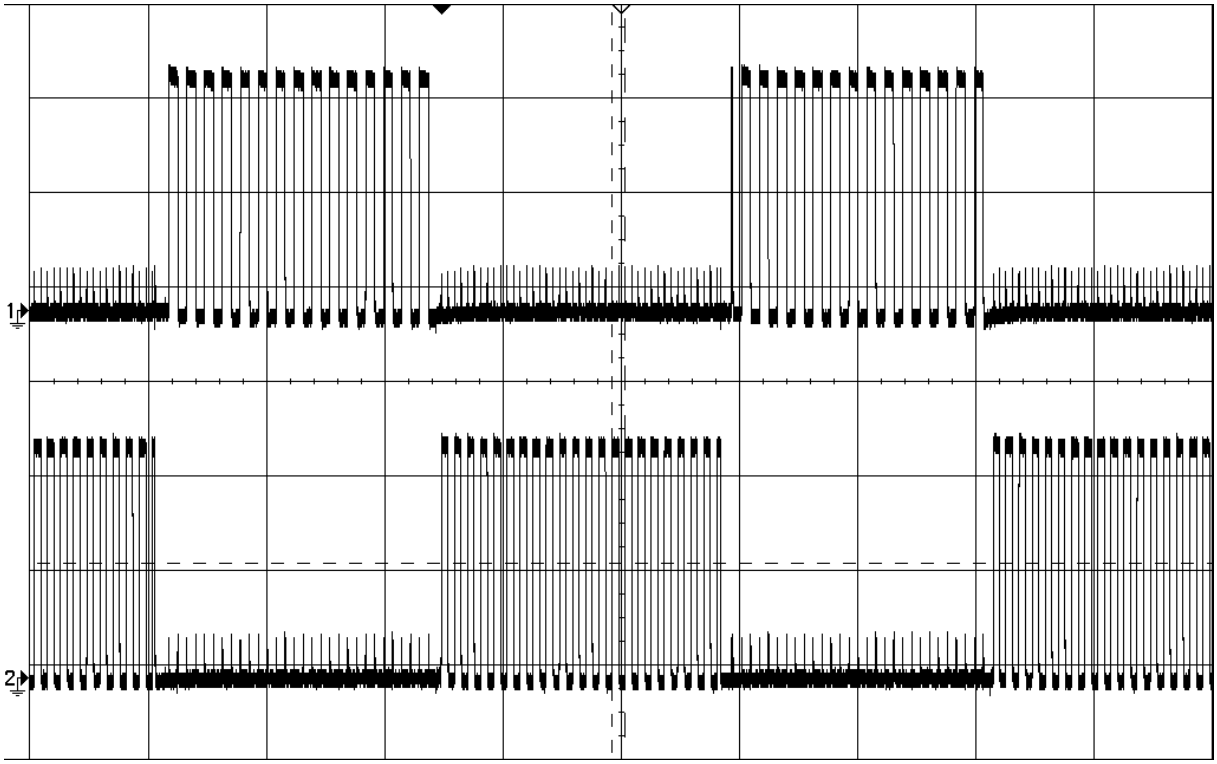
Obr. 23: Průběhy generátoru f1 = 10 kHz, f2 = 50 kHz, 10 impulzů v každé periodě



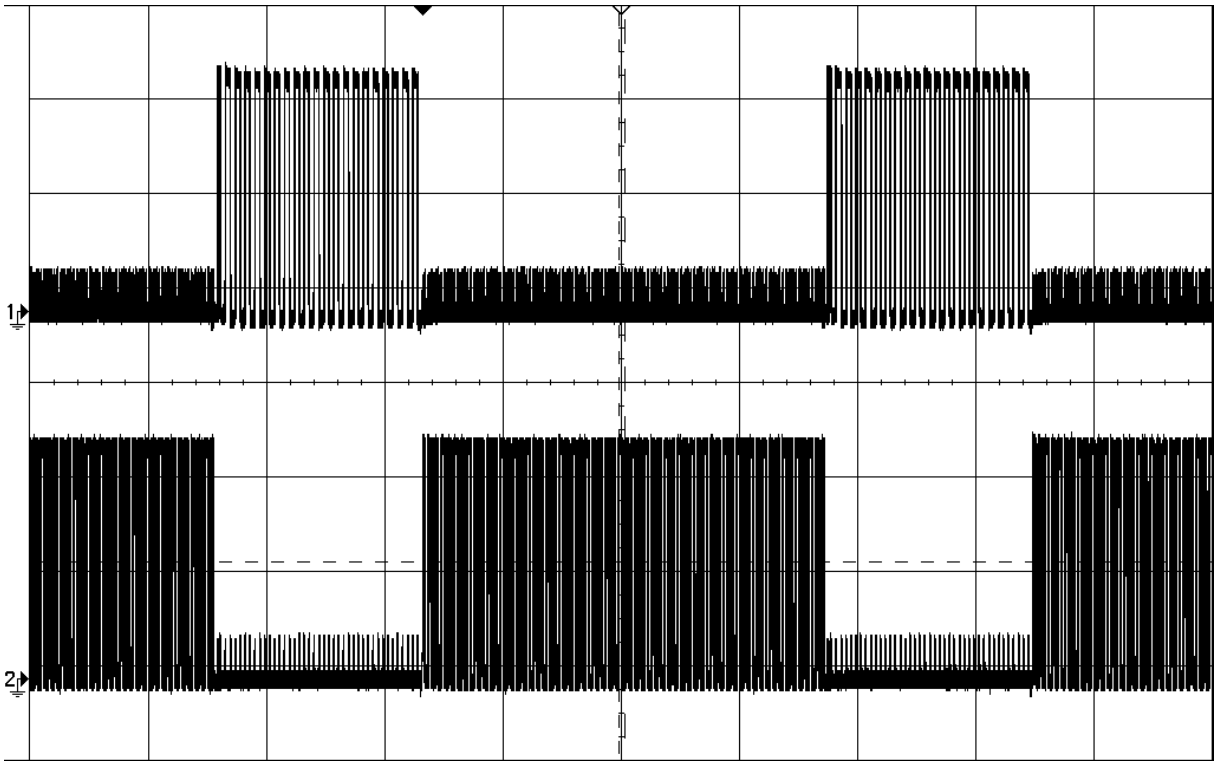
Obr. 24: Průběhy generátoru f1 = 10 kHz, f2 = 50 kHz, 10 impulzů v každé periodě, detail



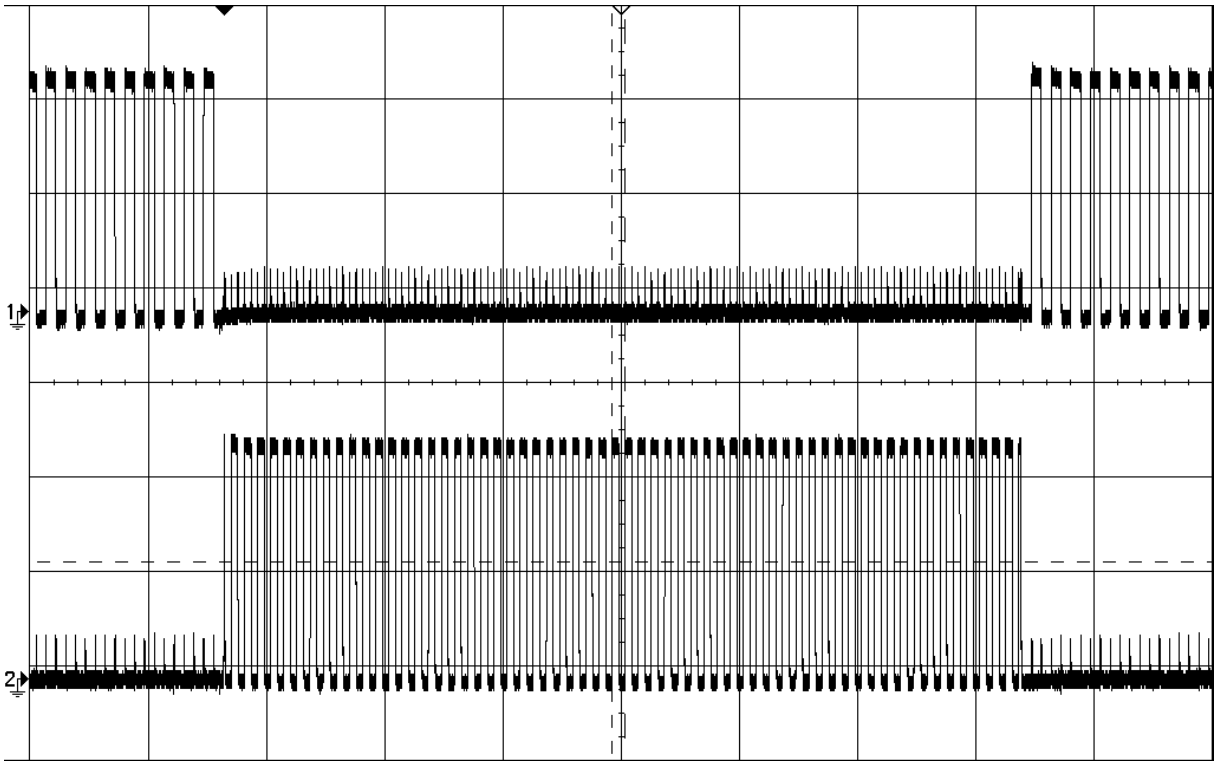
Obr. 25: Průběhy generátoru f1 = 10 kHz, f2 = 50 kHz, 10 impulzů v každé periodě, detail



Obr. 26: Průběhy generátoru f1 = 67 kHz, f2 = 90 kHz, 15 a 20 impulzů



Obr. 27: Průběhy generátoru f1 = 90 kHz, f2 = 60 kHz, 20 a 60 impulzů



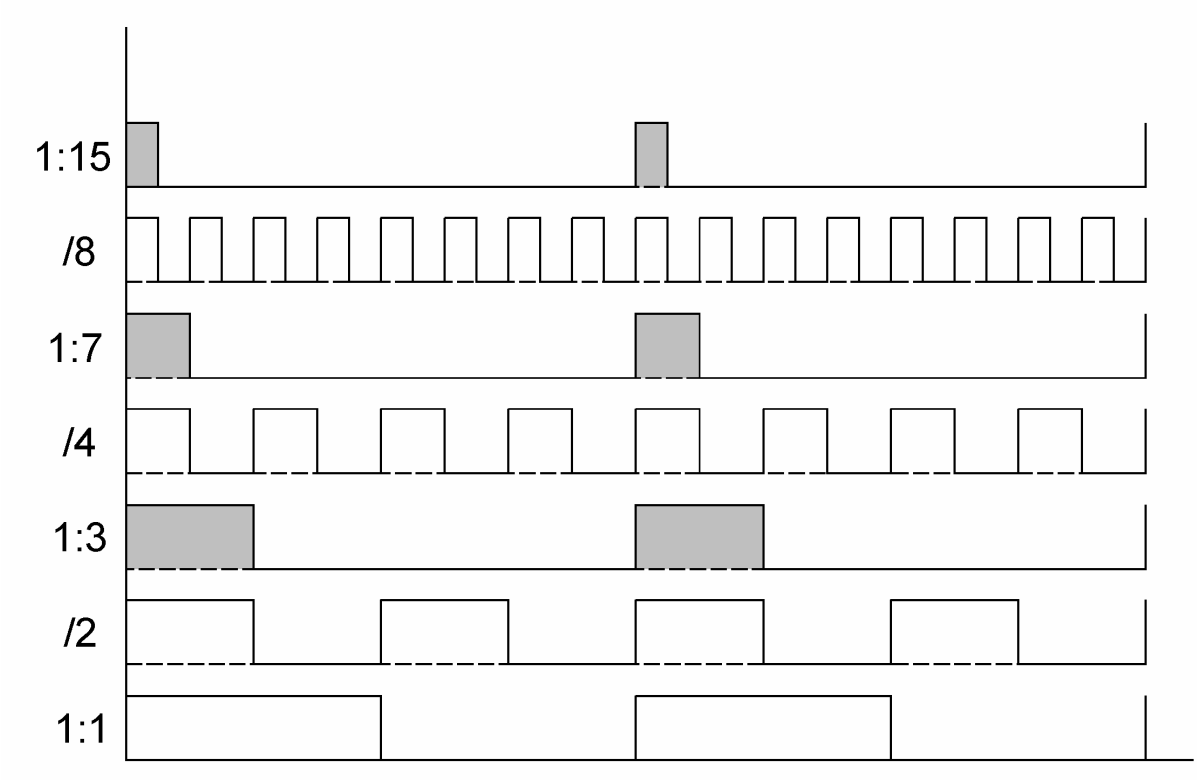
Obr. 28: Průběhy generátoru f1 = 90 kHz, f2 = 60 kHz, 20 a 60 impulzů, detail

6 Návrh modulu pro úpravu střídy výstupního signálu generátoru pro magnetronové naprašování

Modulu pro úpravu střídy je věnovaná samostatná kapitola, protože jde o jedno z dílčích řešení celého problému (generátoru). Podobně, jako je tomu u frekvenční syntézy nelze tento parametr řešit v rámci mikrokontroleru, ten bude i zde plnit pouze funkci nastavení a zobrazení parametrů.

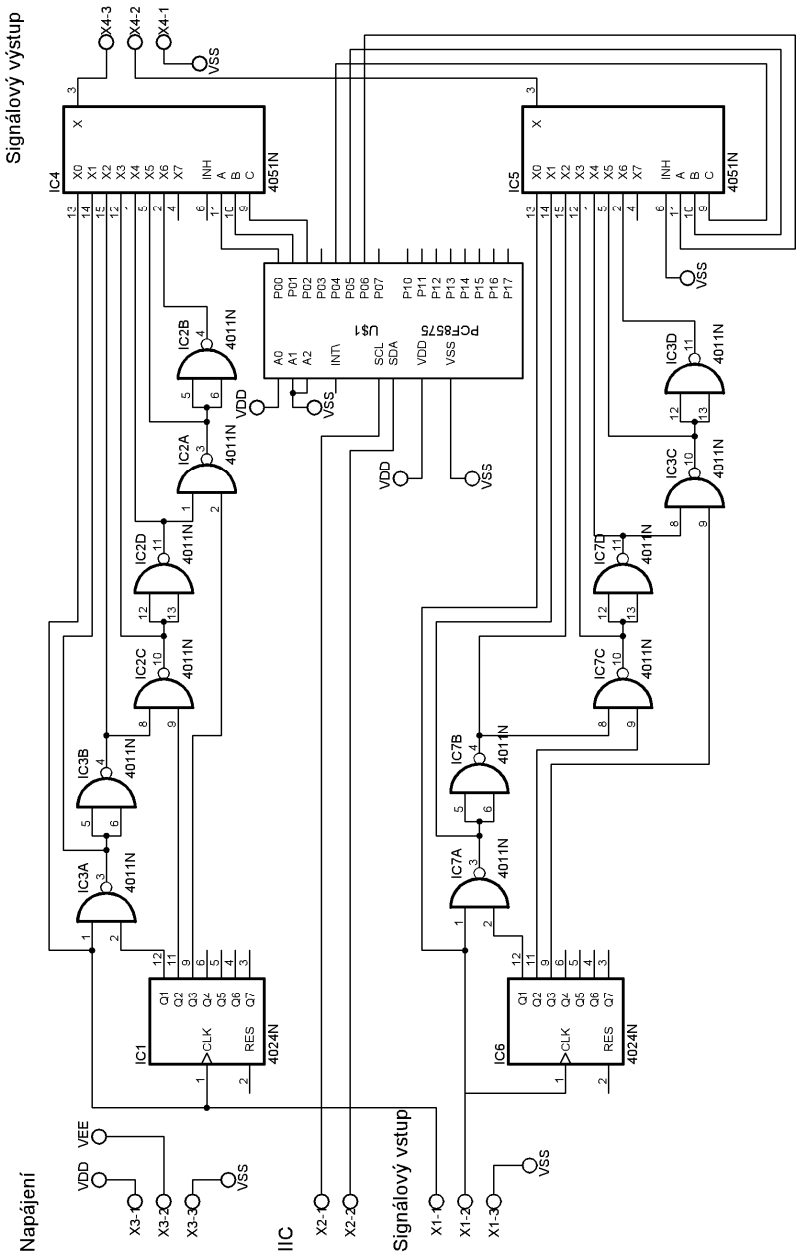
Než přejdu k obvodovému návrhu, musím vytyčit požadavky na modul pro úpravu střídy. V zadání diplomové práce sice nejsou konkrétní požadavky, ale díky znalostem problematiky je mohu lehce stanovit. Střída jednotlivých signálů při magnetronovém naprašování hraje svoji roli, ale není zásadním parametrem. Z toho důvodu není nutné její nastavení v širokých rozsazích. Osobně jsem hledal snadné obvodové řešení, které poskytne několik základních nastavení. Jde tedy o určitý kompromis na obou stranách. Úvaha mě nakonec dovedla k myšlence využít vícestupňový binární čítač a jeho signály zpracovat pomocí jednoduché logiky. Tímto způsoben lze snadno získat signály s poměry plnění 1:3, 1:7, 1:15, 1:31 atd. Tři až čtyři základní nastavení by byly postačující, přičemž spolu s jejich inverzními poměry bych získal šest resp. osm hodnot s poměrem jiným, než 1:1. Na základě těchto úvah jsem stanovil poměry modulu pro úpravu střídy na 1:15, 1:7, 1:3, 1:1, 3:1, 7:1 a 15:1.

Základní myšlenku dobře ilustruje obrázek 29. Přivedeme-li výstupní signál kmitočtové syntézy do vícestupňového binárního čítače, obdržíme na jeho výstupu z prvního stupně signál dělený dvěma. Na výstupu z druhého stupně je signál dělený čtyřmi, podobně na výstupu z třetího stupně je signál dělený osmi a tak to pokračuje až po poslední výstup. Signál za každým následujícím stupněm odpovídá signálu, děleném dvěma, na jeho vstupu. Nyní provedu-li operaci logického součinu výstupního signálu frekvenční syntézy a výstupního signálu z frekvenční syntézy děleného dvěma, získám signál o frekvenci stejné, jako má původní signál ze syntézy, ale jeho střída bude v poměru 1:3. Na obrázku 29 to ilustrují první tři průběhy od spodu. Nyní mohu vzít signál o střídě 1:3 a signál ze syntézy dělený čtyřma a provést jejich logický součin. Tím získám výstupní signál s poměrem plnění 1:7, viz třetí až pátý průběh na obrázku 21. Signál s upravenou střídou mohu zcela stejně logicky násobit s původním signálem děleným osmi a vytvořit tak signál se střídou 1:15. Takto mohu generovat tolik signálů s různou střídou, kolik mám k dispozici stupňů binárního čítače.



Obr. 29: Průběhy modulu pro úpravu střídy

Na obrázku 30 je uvedeno obvodové schéma vycházející z těchto úvah. Vstupem pro signál z kmitočtové syntézy je vstup binárního čítače 4024. Na jeho výstupech Q1, Q2, Q3 jsou signály o frekvenci dělené dvěma, čtyřmi, osmi. Tyto signály jsou přivedeny na vstupy hradel integrovaného obvodu 4011, tak jak je to popsáno v předchozím odstavci. Výstupy hradel jsou přivedeny na vstupy multiplexoru 4051.



Obr. 30: Schéma modulu pro úpravu střídy

Na výstupu multiplexoru je signál odpovídající binární hodnotě na vstupech A, B, C. V tabulce 10 je uvedena pravdivostní tabulka multiplexoru a odpovídající hodnoty na výstupu pro danou kombinaci vstupů.

Tab. 10: Nastavení multiplexoru 4051

vstupy A, B, C	výstup	poměr plnění
0, 0, 0	X0	1:1
1, 0, 0	X1	3:1
0, 1, 0	X2	1:3
1, 1, 0	X3	7:1
0, 0, 1	X4	1:7
1, 0, 1	X5	15:1
0, 1, 1	X6	1:15
1, 1, 1	X7	nevyužito

U modulu pro úpravu střídy se počítá s připojením k mikrokontroleru pomocí sběrnice I^2C. Proto je součástí schématu také expandér portů PCF8575. Obvody pro úpravu střídy postavené na čítači 4024, multiplexoru 4051 a šesti hradlech NAND jsou na schématu uvedeny dva. Každý výstup bude mít vlastní obvody pro úpravu střídy, aby se zabezpečila nezávislost nastavení obou výstupů.

Doplnění programu je velmi jednoduché. Nejprve se musí nakonfigurovat modul MSSP mikrokontroleru podle schématu uvedeného na straně 25. V další fázi se vytvoří položky menu pro nastavení střídy. Využít lze již napsané funkce menu_xy() a ty upravit pro nastavení střídy. Nyní stačí doplnit do funkce menu_5() za volání funkce nastav_delicky(n_1,n_2,i_1,i_2) příkazy kterými nastavím modul pro úpravu střídy.

Tato kapitola uzavírá text věnovaný návrhu generátoru pro magnetronové naprašování. Následující kapitola je věnovaná shrnutí a zhodnocení výsledků práce.

7 Závěr

Předchozích několik kapitol popisuje kompletní návrh generátoru pro magnetronové naprašování. V rámci této diplomové práce jsem navrhnul zapojení generátoru pro magnetronové naprašování. Generátor má dva nezávislé signálové výstupy. Signály jsou generovány v protitaktu a u každého z nich lze nastavit frekvenci v rozsahu 5 – 99 kHz s krokem 1 kHz, počet impulzů v rozsahu 1 – 99 s krokem 1, poměr plnění o hodnotách 1:15, 1:7, 1:3, 1:1, 3:1, 7:1, 15:1.

Návrh jsem rozčlenil do třech hlavních celků. Prvním je teorie a návrh kmitočtové syntézy. Kmitočtová syntéza slouží, jako generátor obdélníkových signálů s nastavitelnou frekvencí. O její řízení se stará mikrokontroler, který zabezpečuje také obsluhu dalších periferií a zobrazení nastavených parametrů pomocí displeje. Mikrokontroler a jeho program je dalším samostatným celkem. Poslední částí je krátká kapitola věnovaná periferním obvodům pro úpravu střídy. Kromě modulu pro úpravu střídy byl generátor také sestaven ve dvou vývojových verzích. U první došlo k chybě v návrhu, která negativně ovlivnila funkci. U druhé verze byla tato chyba odstraněna a bylo provedeno několik úprav, poté již generátor fungoval bez problémů.

Navržený generátor obsahuje celkem 17 integrovaných obvodů, asi 50 ostatních součástek a čtyřřádkový inteligentní zobrazovač. Cena součástek, včetně plošného spoje se pohybuje okolo 1500 Kč. Generátor by nyní bylo vhodné otestovat i s aparaturou pro magnetronové naprašování a odladit objevené chyby. Na základě toho by vznikla konečná verze generátoru. Konečnou verzi generátoru by bylo vhodné poté realizovat na oboustranném plošném spoji se součástkami SMD.

Přesto, že je splněno zadání úkolu a návrh je kompletní, existují minimálně dva náměty na další zdokonalení. Prvním je další rozšíření o časový modul. S jeho pomocí by šlo proces magnetronového naprašování dále zautomatizovat. Kromě frekvence, počtu impulzů a střídy by se nastavila i doba procesu. Po jeho spuštění by naprašování probíhalo až do uplynutí nastaveného času.

Dalším námětem na zdokonalení generátoru je jeho rozšíření o USB port a vytvoření programu pro operační systém Windows. Jehož přínos vidím ani ne tak v komfortní obsluze zařízení, jako spíše v možnosti tvorby databáze, která by mohla obsahovat přednastavené profily pro různé materiály katody i naprašovaného materiálu. Vzniklo by tak zařízení, kde by bylo možné na základě několika kliknutí myši nastavit například tloušťku vytvářené vrstvy, aniž by uživatel musel znát parametry, jako je frekvence, počty impulzů, střída a čas. Ovšem minimálně druhý námět na zlepšení by obsahově vydal na další diplomovou práci.

8 Literatura:

[1] SCHULZ, J. *Generátor impulsů pro impulsní zdroj určený k magnetronovému naprašování : Semestrální projekt 2.* Brno : VUT Brno, Fakulta elektrotechniky a komunikačních technologií, 2007.

[2] BOUŠEK, J. *Zdroj pro Pulsní Magnetronové Naprašování.* In Mikrosyn. Nové trendy v mikroelektronických systémech a nanotechnologiích. Sborník semináře. Brno 12.12.2005. Konference MIKROSYN. Nové trendy v mikroelektronických systémech a nanotechnologiích (Brno 12.12. 2005). Brno: Nakl. Novotný, 2005, Str. 105 – 110. ISBN 80-214-3116-4.

[3] KUT, J. *Phase – Locked Loops – characteristics and applications.* [online]. 2001, [cit. 2008, květen]. Dostupný z WWW: <http://cantab.jkut.com/>.

[4] GRAY, P. R.; MEYER, R. G. *Analysis and Design of Analog Integrated Circuits.* 3rd ed. Wiley, 1992. Str. 681-698. ISBN 978-0471574958.

[5] RAZAVI, B. *Design of Analog CMOS Integrated Circuits.* McGraw-Hill, 2001. Kapitola 15. ISBN 978-0072380323.

[6] Texas Instruments. *CD4060BC, 14-Stage Ripple Carry Binary Counter : Data Sheet.* [online]. Katalogové listy společnosti Texas Instruments. [cit. 2008, květen] Dostupný z WWW: <http://focus.ti.com/docs/prod/folders/print/cd4060b.html>.

[7] MORGAN, D. K. *CD4046B Phase-Locked Loop : A Versatile Building Block for Micropower Digital and Analog Applications : Application Report SCHA002A.* [online]. February 2003. [cit. 2008, květen] Dostupný z WWW: <http://focus.ti.com/general/docs/techdocsabstract.tsp?abstractName=scha002a>.

[8] Texas Instruments. *CD4046B, Phase-Locked Loop : Data Sheet.* [online]. Katalogové listy společnosti Texas Instruments. [cit. 2008, květen] Dostupný z WWW: <http://focus.ti.com/docs/prod/folders/print/cd4046b.html>.

[9] Texas Instruments. *CD4059B, Programmable Divide-by-N Counter : Data Sheet.* [online]. Katalogové listy společnosti Texas Instruments. [cit. 2008, květen] Dostupný z WWW: <http://focus.ti.com/docs/prod/folders/print/cd4059a.html>.

[10] Microchip. *PIC16F87XA, 28/40/44-Pin Enhanced Flash Microcontrollers: Data Sheet.* [online]. Katalogové listy společnosti Microchip. [cit. 2008, květen] Dostupný z WWW: <http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1335&dDocName=en010242>.

[11] ILETT, J. How to use intelligent L.C.D.s. Part One. *Everyday Practical Electronics.* February 1997, s. 84 – 89.

[12] ILETT, J. How to use intelligent L.C.D.s. Part Two. *Everyday Practical Electronics,* March 1997, s. 192 – 196.

[13] HRBÁČEK, J. *Komunikace mikrokonrtoleru s okolím.* 1. vydání, BEN – technická literatura, Praha 1999. Str. 86 – 98. ISBN 80-86056-42-2.

[14] Bona Fide Technology Ltd. *MC1604 Standard LCD module: Data Sheet.* [online]. Katalogové listy společnosti Bona Fide Technology Ltd. [cit. 2008, květen] Dostupný z WWW: <http://www.bonafide.com.hk/>.

[15] B Knudsen Data. *CC5X C Compiler for thePICmicro Devices: User`s Manual.* [online]. Manuál společnosti B Knudsen Data. [cit. 2008, květen] Dostupný z WWW: <http://www.bknd.com/>.

[16] Texas Instruments. *CD4024B, CMOS Binary Counter : Data Sheet.* [online]. Katalogové listy společnosti Texas Instruments. [cit. 2008, květen] Dostupný z WWW: <http://focus.ti.com/docs/prod/folders/print/cd4024b.html>.

[17] Texas Instruments. *CD4051B, CMOS Analog Multiplexer/Demultiplexer : Data Sheet.* [online]. Katalogové listy společnosti Texas Instruments. [cit. 2008, květen] Dostupný z WWW: <http://focus.ti.com/docs/prod/folders/print/cd4051b.html>.

[18] Philips. *PCF8575, Remote 16-bit I/O expander for I2C Bus : Data Sheet.* [online]. Katalogové listy společnosti Philips. [cit. 2008, květen] Dostupný z WWW: <http://www.nxp.com/#/homepage/cb=[t=p,p=/50807/41735/41850,f=PCF8575_3]|pp=[v=p,t=pip,i=PCF8575_3,fi=41850,ps=0]|[7]>.

9 Seznam příloh

1) Program pro mikrokontroler, soubory main.c, lcd.c, menu.c, pracuj.c.
2) Vnitřní zapojení integrovaného obvodu 4060 z katalogového listu
3) Katalogový list integrovaného obvodu 4046.
4) Aplikační poznámky k integrovanému obvodu 4046 - vnitřní zapojení VCO, FC.
5) Blokové schéma a funkční tabulka integrovaného obvodu 4059.
6) Citované strany z katalogového listu mikrokontroleru 16F877A.
7) Plošný spoj navrženého generátoru.