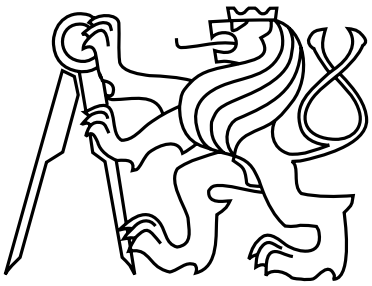
Na tomto místě bude oficiální zadání vaší práce

- Toto zadání je podepsané děkanem a vedoucím katedry,
- musíte si ho vyzvednout na studiijním oddělení Katedry počítačů na Karlově náměstí,
- v jedné odevzdané práci bude originál tohoto zadání (originál zůstává po obhajobě na katedře),
- ve druhé bude na stejném místě neověřená kopie tohoto dokumentu (tato se vám vrátí po obhajobě).

České vysoké učení technické v Praze
Fakulta elektrotechnická



Bakalářská práce

Zařízení pro dlouhodobé měření teploty, vlhkosti a tlaku

Miroslav Vokoun

Vedoucí práce: Ing. Pavel Kubalík, Ph.D.

Studijní program: Elektrotechnika a informatika strukturovaný bakalářský

Obor: Informatika a výpočetní technika

červenec 2008

Poděkování

Na tomto místě bych chtěl poděkovat hlavně vedoucímu mé práce ing. Pavlovi Kubalíkovi, Ph.D. za podporu a cenné rady při tvorbě této práce. Dále bych chtěl poděkovat celé mé rodině za všeobecnou podporu, která umožnila vznik této práce.

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.
Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne . . 2008 .....

Abstract

This thesis deals with the design and the implementation of device, which measures temperature, humidity and pressure. This device is divided into two parts - into driving part and measuring part. Measuring part consists of maximum two measuring modules, that contain sensors for measuring mentioned values and that are connected to driving part via patch cords. Driving part consists of driving circuit, which evaluates data from measuring modules per hour and stores them into portable medium.

Abstrakt

Tato práce se zabývá návrhem a realizací zařízení, které měří teplotu, vlhkost a tlak. Zařízení je rozděleno na dvě části, a to část řídící a měřící. Měřící část se skládá z maximálně dvou měřících modulů, které obsahují senzory pro měření již zmíněných veličin, a které jsou k řídící části připojeny pomocí propojovacího kabelu. Řídící část obsahuje řídící obvod, který jednou za hodinu vyhodnocuje údaje z měřících modulů a ukládá je na přenosné médium.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Cílem této bakalářské práce je navrhnout a realizovat zařízení, které bude umožňovat dlouhodobé měření teploty, vlhkosti a tlaku. Celé zařízení se bude skládat ze dvou částí. Jednou částí je část měřící, která může mít až dva měřící moduly. Měřící moduly budou obsahovat teplotní senzor, senzor vlhkosti a senzor tlaku. Ty budou odolné venkovní teplotě. Údaje z těchto senzorů se budou případně převádět na digitální signál. Tyto signály povedou kabelem do druhé části. Druhá část je řídící. Řídící část bude obsahovat mikrořadič pro vyhodnocování údajů ze senzorů. Ten je bude vyhodnocovat jednou za hodinu. Vyhodnocení údajů se skládá z rekonstrukce signálu ze senzoru teploty, vlhkosti nebo tlaku a vypočítání konečných hodnot. Tyto hodnoty se pak uloží do paměti. Zařízení bude možné napájet z baterií a bude mít nízkou spotřebu.

Tato práce je dělena do kapitol. První kapitola je tento úvod. Ve druhé kapitole je specifikace cílů práce a rešerže podobných zařízení. Ve třetí kapitole je pak probírána analýza potřebných obvodů pro zařízení a návrh realizace. Čtvrtá kapitola pojednává o provedení jednotlivých částí zařízení a popis softwaru. Pátá kapitola je zaměřena na testování. V šesté kapitole je shrnutý závěr. Sedmá kapitola obsahuje využitou literaturu.

2 Popis problému, specifikace cíle

2.1 Vymezení cílů a požadavků této práce

- Analýza a výběr vhodných senzorů pro měřící část.
- Analýza převodu naměřených hodnot na vhodný signál pro přenos mezi měřícími moduly a řídící částí.
- Analýza a výběr obvodů pro řídící část.
- Analýza ukládání dat.
- Analýza obvodů pro snížení spotřeby.
- Návrh hardwarové části.
- Analýza zpracování signálů z měřící části.
- Implementace firmwaru.
- Testování.

2.2 Rešerže zařízení na trhu

Při vyhledávání zařízení, která jsou podobná zařízení definovanému v zadání, lze narazit buď na profesionální nekomerční produkty nebo na komerční produkty. My jsme pro porovnání vybrali dvě zařízení, každé určené pro jiné odvětví. Z těch nekomerčních můžeme jmenovat tzv. datalogery, ty průběžně ukládají jednu nebo více měřených veličin do paměti, přitom podporují připojení k PC, takže si můžeme průběhy hodnot těchto veličin snadno zobrazit například do grafu. Komerční produkty jsou naproti tomu konstruovány pro co největší uživatelský komfort, mají zobrazovací displej a snadné ovládání, ale většinou nepodporují ukládání průběžných hodnot a slouží pro zjišťování okamžitých hodnot, které se pouze cyklicky zobrazují na displeji.

Jako dataloger bylo vybráno zařízení `OM-CP-PRHTEMP101` od výrobce OMEGA, které má podobné vlastnosti jako naše zařízení. Jeho parametry jsou:

- Měření teploty, vlhkosti a tlaku, ale pouze na jednom místě.
- Rozsah měření teploty od -40 do 80°C, vlhkosti od 0 do 100% relativní vlhkosti, tlaku od 0 do 30psi absolutního tlaku (tento senzor je kompenzován pro teplotu od 0 do 50°C).
- Do paměti se vejde až 13107 hodnot zvlášť pro každý senzor.
- Zařízení obsahuje hodiny reálného času, takže obvod ukládá pro každou změřenou trojici hodnot údaj o času měření.
- Napájení je řešeno z baterií.
- Paměť pro ukládání dat je realizována pamětí typu Flash, takže nedojde ke ztrátě dat při vybití baterií.
- Veškerá komunikace se zařízením probíhá přes PC, kde je dostupný port COM. Software dodávaný se zařízením přehraje naměřené hodnoty do PC, které můžeme ukládat ve formě tabulky, nebo v grafické formě (můžeme si vybrat z mnoha používaných jednotek).

- Tímto softwarem můžeme u zařízení navíc:
  - Zkalibrovat senzory.
  - Nastavit interval měření od 30 minut do jednoho dne.
  - Zvolit čas a datum, kdy má začít cyklus měření.
- Baterie má výdrž jeden rok při měřícím intervalu jedna minuta.
- Měření je signalizováno bliknutím LED diody. LCD displej není k dispozici.
- Vysoká přesnost měření.
- Malé rozměry zařízení.
- Cena na trhu celého zařízení včetně softwaru je kolem 7000,-Kč.

Jako meteostanici (z komerčních zařízení) jsme pro porovnání vybrali zařízení `WS-444 PC` od výrobce ELV AG, které je výjimkou mezi meteostanicemi, protože podporuje ukládání dat do paměti a komunikaci s PC přes sběrnici USB (to mnoho stanic neumí). Jeho parametry jsou:

- Měření teploty, vlhkosti a tlaku, rychlosti větru a množství srážek až na devíti místech (bezdrátově).
- Rozsah měření vnitřní teploty (senzor je přímo ve stanici) je od 0 do 60°C, vnější teploty od -30 do 80°C, relativní vlhkosti od 0 do 99% (jeden senzor je opět přímo ve stanici), tlaku od 300 do 1100 hPa (ve stanici), rychlosti větru od 0 do 200km/h a množství srážek od 0 až 999mm.
- Napájení je řešeno z baterií, senzory pak mají také své napájení.
- Baterie ve stanici vydrží až 2 roky.
- Po odpojení baterií ze zařízení, nebo jejich vybití dojde ke ztrátě dat v paměti.
- Do paměti se vejde až 3200 jednotlivých záznamů.
- Data z paměti lze přečíst přes rozhraní USB a vyhodnotit přiloženým softwarem. Data lze zobrazit do grafů.
- Zařízení nemá možnost tak pestrého nastavení jako u předchozího zařízení.
- Nastavit lze jen interval měření.
- Zobrazení dvou volitelných hodnot na displeji LCD, včetně indikace zaplněné paměti.
- Nižší přesnost.
- Cena na trhu samotné stanice je kolem 3500,-Kč, k ní by se musely přikoupit ještě venkovní senzory (dva, jako u našeho zařízení), tj. 2000,-Kč. Senzory rychlosti větru a množství srážek nepotřebujeme, proto je do ceny započítávat nebudeme.

Z výběru vychází tedy lépe meteostanice, dokáže totiž měřit teplotu a vlhkost na více místech, a to bezdrátově a ještě je mnohem levnější než dataloger. Tlak stanice na více místech sice neměří, to ale není potřeba, tlak by totiž měl být stejný jak v interiéru tak v exteriéru (kde jsou čidla). Dataloger má sice lepší přesnost, ale rozdíl je minimální. U teploty je rozdíl mezi chybami obou zařízení 0,3°C (chyby 0,5 a 0,8°C) a u vlhkosti 2% (chyby 3 a 5%). Chyba tlaku není u stanice stanovena.

3 Analýza a návrh řešení

3.1 Obvody v návrhu

V tabulce 3.1 je uveden přehled obvodů potřebných k realizaci celého zařízení. Jejich popis najdete v jednotlivých podkapitolách.

Činnost	**Obvod**
Měření teploty	Teplotní senzor
Měření vlhkosti	Senzor relativní vlhkosti
Měření tlaku	Senzor absolutního tlaku
Zpracování údajů ze senzorů a jejich ukládání	Mikrořadič
Úložiště dat	SD karta
Generování časového intervalu	Hodiny reálného času
Komunikace s PC	Převod USB na RS232
Napájení obvodů	Spínaný zdroj

Tabulka 3.1: Přehled obvodů

3.2 Teplotní senzor

Abychom mohli měřit teplotu pomocí mikrořadiče, je potřeba teplotu převést na nějakou vhodnější veličinu. Protože teplota je nejčastěji měřená veličina z veličin teplota, vlhkost a tlak, bylo nalezení optimálního senzoru nejsnazší. Na trhu jsou k dispozici například senzory, které mění svůj elektrický odpor podle teploty, převádí teplotu na napětí nebo střídu, nebo senzory, které jsou čistě digitální a komunikace mezi nimi a mikrořadičem probíhá přes speciální sběrnici.

Odporové senzory jsem hned na začátku z výběru vyřadil, protože se nedají připojit přímo na vstup mikrořadiče a bylo by třeba převádět el. odpor na jinou el. veličinu, například frekvenci. Proto volba padla na senzory, které se mohou připojit přímo na vstup mikrořadiče. Modul se senzory může být umístěn někde ve venkovním prostředí, proto je potřeba do kritérií výběru zahrnout i rozsah měřitelné teploty minimálně od -20 do +60 stupňů Celsia. Údaje ze senzoru by pak měly jít přepočítat na hodnotu teploty ve stupních Celsia (to je většinou splněno, u nás se většinou komerčně neprodávají senzory s jinou stupnicí). Dalším kritériem je napájení senzoru, které by mělo být 5V. V tabulce 3.2 je uveden přehled senzorů, splňujících tato kritéria.

Označení	**SMT160-30-18**	**DS18B20Z**	**LM35CZ**
Výrobce	Smartec	Dallas Semiconductor	National Semiconductor
Pouzdro	TO18	SO08	TO92
Napájení	4,75 – 7V	3 – 5,5V	4 – 30V
Rozsah teplot	-45 – 130°C	-55 – 125°C	-55 – 150°C
Výstup	pulzní šířková modulace	1-Wire rozhraní	napětí

Tabulka 3.2: Přehled vybraných teplotních senzorů

Senzor `LM35CZ` má napěťový výstup, proto by bylo potřeba ještě napětí v mikrořadiči převádět A/D převodníkem (ten je ve většině mikrořadičů standardně). Musíme ale ještě vzít v úvahu,

že modul se senzory je k řídící části spolu s mikrořadičem připojen pomocí propojovacího kabelu, ve kterém se mohou naindukovat rušivé vlivy, takže změřené napětí nemusí být správné. Proto byl tento senzor zavržen. Dalším senzorem je `DS18B20Z`, který používá pro přenos dat o teplotě sběrnici. Tato jedno-vodičová sběrnice není ovšem ve většině mikrořadičů hardwarově implementována, takže by se komunikace musela řešit softwarově. Optimální se tedy zdá být senzor `SMT160-30-18`, který převádí teplotu na střídu. Ta je mikrořadičem snadno měřitelná a signál, který je senzorem generován (pulzní šířková modulace) je poměrně odolný proti rušení. S kalibrací těchto senzorů není problém, jsou zkalibrovány již z výroby a pro přepočet teploty (například ze střídy) mají ve svých datasheetech vzorec.

3.3 Senzor vlhkosti

U senzoru vlhkosti je situace podobná jako u teplotního senzoru, tzn. je potřeba vlhkost převést na nějakou měřitelnou veličinu. V případě senzorů vlhkosti se většinou používají kapacitní senzory, které mění svou elektrickou kapacitu podle vlhkosti. Tyto senzory měří tzv. relativní vlhkost, která je udávaná v procentech a vyjadřuje poměr okamžitého množství vodních par ve vzduchu k množství vodních par, kdy je vzduch nasycen. Existují i varianty senzorů se sběrnicí, ty jsou ale cenově v jiné kategorii.

Kritériem pro výběr kapacitního senzoru je stejně jako u teplotního provozní teplota, která by měla být, jak už bylo uvedeno, minimálně od -20 do +60 stupňů Celsia. Tím se výběr zúžil na senzor od firmy Smartec s označením `SMTHS07`, který má provozní teplotu od -40 do 120 stupňů Celsia. Kapacita senzoru se pohybuje od 280pF do 380pF. Při výrobě senzorů není možné, aby byla kapacita u všech výrobků stejná, proto je potřeba senzor v aplikaci dodatečně zkalibrovat. Rozsah měření vlhkosti je u tohoto senzoru od 10 do 95 procent relativní vlhkosti.

Kapacita ale stále není měřitelná mikrořadičem. Proto je potřeba její převod na jinou veličinu nebo informaci, zpracovatelnou mikrořadičem. To je ale dost obtížné, když mají kapacitní senzory kapacitu maximálně v řádu stovek pF. Přitom parazitní kapacita přívodů k měřícímu obvodu je v řádu desítek pF a změna kapacity senzoru v závislosti na vlhkosti je v řádu jednotek až desetin pF.

3.3.1 Obvod pro měření kapacitních a jiných senzorů

Řešení nabízí opět firma Smartec, když jsem při vyhledávání datasheetů k jejím výrobkům narazil právě na obvod, který dokáže měřit tak malé rozdíly v kapacitě, a který dokáže eliminovat parazitní kapacitu přívodů senzoru k tomuto obvodu. Má označení `UTI03` (Universal Transducer Interface) a je u nás běžně dostupný i v SMD pouzdru. Podobné obvody pro měření kapacity s eliminací těchto problémů jsem nenašel. Jeho použití ale není jen pro měření kapacity, ale i pro měření odporových teploměrů a odporových můstků. Pro měření využívá metodu tří signálů, kdy měření probíhá ve třech fázích. V první fázi se měří tzv. offset, který udává velikost parazitní kapacity, v druhé fázi nějaká známá referenční hodnota a ve třetí fázi neznámá hodnota, kterou chceme měřit. Na výstupu pak je signál se třemi periodami, které jsou přímo úměrné těmto měřeným hodnotám. Odečtením periody offsetu od periody měřené a referenční hodnoty a jejich vzájemným podělením dostaneme poměr těchto hodnot. Když tento poměr vynásobíme známou referenční hodnotou, dostaneme správně změřenou neznámou hodnotu. Protože je výstupní signál obdélníkový s nějakou frekvencí (záleží na velikosti period), je poměrně odolný proti rušení a obvod tedy můžeme umístit na desku společně se senzory. Více o tomto obvodu se dozvíte v kapitole Realizace nebo v [11].

3.4 Senzor tlaku

Další problematikou, kterou zde budeme probírat, je měření tlaku. Jaký tlak vlastně máme měřit, jaký mají mít senzory rozsah, opět záleží na tom, v jakém rozsahu teplot můžeme měřit apod. Při hledání vhodných senzorů jsem našel senzory s napěťovým výstupem nebo s digitálním výstupem. Komerčně jsou k dispozici senzory pro měření absolutního tlaku a relativního tlaku. Musíme brát také v úvahu že měříme tlak plynů, nikoliv tlak mechanický (například tlak lisu).

Měření relativního tlaku je pro naší aplikaci nepoužitelné, protože tyto senzory měří rozdíl mezi okamžitým atmosférickým tlakem a tlakem, který je v jiném, od atmosférického tlaku odděleném prostředí. Měří se tedy tzv. podtlak nebo přetlak. Správný senzor je tedy pro měření absolutního tlaku. Ten využívá toho, že ve vakuu je tlak nulový. Měří se tedy rozdíl mezi okamžitým atmosférickým tlakem (nebo jiným) a nulovým tlakem. Protože my chceme měřit tlak atmosférický, což je tlak, který vytváří vzduch na zemský povrch, musíme vybírat senzor s takovým rozsahem, v jehož mezích se atmosférický tlak pohybuje.

3.4.1 Problematika atmosférického tlaku

Normální atmosférický tlak je změřený u hladiny moře (0 metrů nad mořem) a je přibližně:

$$P = 1013,25\,hPa$$

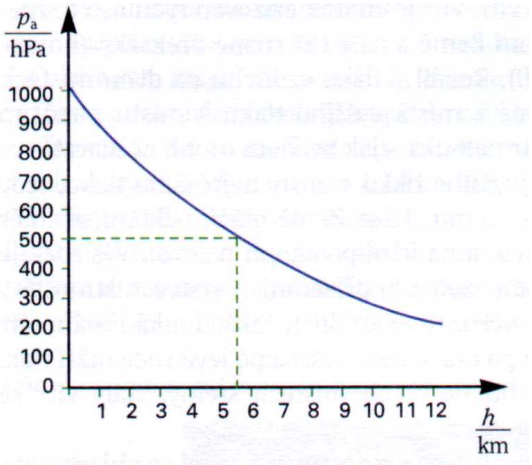
,kde jednotka hPa je nejpoužívanější jednotkou v meteorologii. Její přepočet:

$$1\,hPa = 100\,Pa$$

U senzorů se většinou setkáváme také s jednotkou psi, která je v hPa:

$$1\,psi = 68,9476\,hPa$$

Se stoupající nadmořskou výškou atmosférický tlak rychle klesá. Dobře je to vidět na obrázku 3.1.



Obrázek 3.1: Klesající tlak s výškou

Změny atmosférického tlaku ale nejsou jenom ve svislém směru, ale i ve vodorovném. To znamená, že když změříme na jednom místě nějaký tlak, tak na druhém místě, které je ve stejné výšce, můžeme naměřit tlak zcela odlišný. Nebo změříme tlak dvakrát na stejném místě, ale s

časovým odstupem, takže můžeme dostat dvě odlišné hodnoty. To je dáno například teplotou vzduchu, kdy je studený vzduch těžší než teplý, nebo i jiným tíhovým zrychlením na různých místech apod. V meteorologii je vodorovný tlak a jeho rychlé změny důležitý například pro předpověď počasí. Často v předpovědích vídáme údaje o tlaku s hodnotou kolem 1013hPa na místě, které je umístěno 500 metrů nad mořem, i když tento tlak odpovídá tlaku u hladiny moře. Je to dáno tím, že v meteorologii je podstatný právě jen vodorovný tlak, takže změřený okamžitý atmosférický tlak v místě 500m.n.m. je přepočítán na úroveň hladiny moře, aby bylo snadné hodnoty tlaku z různých míst porovnat. Protože přepočet z okamžitého tlaku na tlak u hladiny moře provádějí meteorologické stanice podle přesných tabulek, kdy je potřeba znát mimo jiné přesnou výšku měření, přesnou teplotu v místě měření, nebo tíhové zrychlení, nebudeme se v této bakalářské práci tímto přepočtem zabývat. Dalším důvodem je i to, že změny vodorovného tlaku jsou většinou velmi malé (v řádu jednotek hPa).

3.4.2 Výběr senzoru tlaku

Musíme se tedy spokojit s tím, že senzor bude měřit jak svislý tak vodorovný (okamžitý) atmosférický tlak. Z obrázku 3.1 můžeme určit, v jakém rozsahu by měl umět senzor pracovat. Dle mého názoru bude stačit, když měření bude probíhat mezi 0m.n.m a 2000m.n.m. To odpovídá tlaku od 800hPa do 1100hPa. Senzor musí mít 5V napájení. V tabulce 3.3 je uveden přehled vybraných senzorů, které měří absolutní tlak.

Označení	**MPXH6115AC6U**	**SPD100AD**
Výrobce	Freescale Semiconductor	Smartec
Pouzdro	SSOP-8	DIP-6
Napájení	4,75 – 5,25V	2,7 – 5,5V
Rozsah tlaků	150 – 1150hPa	0 – 6894hPa
Rozsah provozních teplot	-40 – 125°C	-20 – 85°C
Výstup	napětí	digitální

Tabulka 3.3: Přehled vybraných senzorů tlaku

Pro naší aplikaci by byl ideální senzor `SPD100AD`, který má digitální výstup, lze ho tedy připojit přímo na vstup mikrořadiče a nemusí nám vadit případné rušení při přenosu signálu z modulu čidel do řídícího obvodu. Tento obvod je ale docela specifický, proto není běžně dostupný a musí se objednávat přímo u výrobce. Dodací lhůty pro tento obvod nejsou nejkratší, a proto bylo od toho senzoru upuštěno. Zbývá tedy senzor `MPXH6115AC6U`, který má oproti předchozímu jednu obrovskou nevýhodu. Tou je napěťový výstup (rozsah od 0,2 do 4,8V), který je potřeba kvůli případnému rušení mezi senzorem a mikrořadičem převést na nějakou vhodnější veličinu. Senzor je již zkalibrován z výroby.

3.4.3 Převod napětí

Nejčastějším převodem z napětí je převod na frekvenci nebo střídu. Uvažoval jsem i nad řešením umístit k senzoru A/D převodník a z toho pak přes I^2C (sériová dvou-vodičová sběrnice firmy Philips) sběrnici přenést data do mikrořadiče, jenže opět z důvodu možného rušení a z důvodu ceny převodníku bylo od tohoto řešení upuštěno. Převod na střídu byl také zavržen, protože pro modulaci napětí potřebujeme nějak generovat trojúhelníkový průběh a poté tento průběh porovnávat s modulovaným napětím. V případě časovače 555 (více o tomto obvodu v [12]) by to byl jeden obvod pro generování a jeden obvod pro porovnávání. To by na modulu se senzory,

který by měl být co nejmenší, zabíralo většinu místa. Při hledání obvodů, které by měly modulaci napětí přímo integrovanou, jsem žádný obvod, který by mohl být napájen 5V a mohl by být provozován v teplotách od -20 do +60°C, neobjevil. Většinou jsou k dispozici pouze obvody pro řízení různých motorků, které jsou napájeny napětím v řádu desítek V.

Proto jsem se tedy zaměřil na hledání obvodů pro převod napětí na frekvenci. Pro hledání platila opět kritéria provozní teploty od -20 do +60°C a napájecího napětí 5V. Výstupní signál by měl být obdélníkový s amplitudou 5V. Přehled vybraných převodníků je v tabulce 3.4.

Označení	**TS555I**	**AD645JN**	**LM231**
Výrobce	STMicroelectronics	Analog Devices	National Semiconductor
Pouzdro	SO8	DIP-8	DIP-8
Napájení	2 – 16V	4,5 – 36V	4,5 – 40V
Rozsah provozních teplot	-40 – 125°C	-40 – 85°C	-25 – 85°C

Tabulka 3.4: Přehled vybraných převodníků z napětí na frekvenci

Převodník `LM231` není pro náš převod příliš vhodný. V aplikačním zapojení, které slouží pro převod z napětí na frekvenci, je totiž potřeba velké množství dodatečných obvodů, aby převod fungoval. To není pro naše použití příliš vhodné. Mnohem lépe by se dal použít převodník `AD645JN`, u kterého je aplikační zapojení opravdu jednoduché a nevyžaduje spoustu součástek. Problém nastává při bližším nastudování. Vstupní napětí může být totiž v rozsahu pouze 0 až 1V. Když vezmeme v úvahu, že výstupní napětí vybraného senzoru tlaku může být od 0,2 do 4,8V, je rozsah převodníku nedostatečný. Výstup ze senzoru by se sice dal zmenšit odporovým děličem na menší rozsah, jenže tím by jsme dle mého názoru ztratili přesnost a schopnost detekovat malé změny tlaku (v řádu jednotek hPa). Jako převodník byl tedy vybrán CMOS časovač `TS555I`, který je velmi jednoduchý a má široké možnosti použití. Pro nastavení požadované funkce většinou vyžaduje jen minimum přídavných součástek. Pokud budeme chtít obvodem generovat frekvenci, použijeme doporučené zapojení pro astabilní chování, uvedené v datasheetu [12]. Aby byla frekvence proměnná s napětím, mírně toto zapojení upravíme. Více v kapitole Realizace.

3.5 Mikrořadič

Už v zadání této práce je, že mikrořadič má být od firmy Atmel. Mikrořadič bude zjišťovat hodnoty ze senzorů jednou za hodinu, takže měření hodnot ze senzorů nemusí být tak rychlé. Spíše je potřeba se zaměřit na spotřebu tohoto obvodu. Už obvody tohoto výrobce, které nejsou zaměřeny přímo na spotřebu, ale na výkon, mají docela malou spotřebu. My však budeme vybírat z těch obvodů, které mají sice dvakrát snížený výkon, ale i dvakrát sníženou spotřebu oproti výkonnější variantě. Na výběr jich má firma Atmel totiž docela dost.

Kromě senzorů bude ještě mikrořadič zajišťovat komunikaci s reálnými hodinami přes sběrnici I^2C, komunikaci s SD kartou pro ukládání informací, komunikaci s obvodem, který umožní komunikaci mikrořadiče s PC přes sběrnici USB a alternativně komunikaci přímo s PC přes sériové nebo paralelní rozhraní. Proto bychom měli provést odhad počtu vývodů, kolik by jich měl tento obvod minimálně obsahovat. Odhad je tedy uveden v tabulce 3.5.

Protože údaje ze senzorů jsou ve tvaru obdélníkového signálu, bylo by vhodné, kdyby každý vstup, na který je připojen senzor, mohl generovat přerušení v programu mikrořadiče. Dále bude potřeba nějaký čítač, kterým bychom mohli zjišťovat délky jednotlivých period a impulzů

Funkce	**Počet vývodů**
Zpracování údajů z modulu senzorů č. 1	3
Zpracování údajů z modulu senzorů č. 2	3
Komunikace s SD kartou	6
Komunikace s hodinami reálného času	2
Hodinový alarm hodin	1
Komunikace s PC	4
Přerušení od zdroje napětí (indikace nízkého napětí)	1
Vypnutí napájení pro modul senzorů č. 1	1
Vypnutí napájení pro modul senzorů č. 2	1
Přerušení od zdroje napětí (indikace nízkého napětí)	1
Krystalový oscilátor	2
Napájení a zem	2
Celkem	27

Tabulka 3.5: Odhad počtu vývodů mikrořadiče

ze signálů jednotlivých senzorů. Shrneme tedy požadavky na mikrořadič:

- Minimálně 16kB paměti pro program.
- Minimálně jeden 16bitový čítač/časovač.
- Verze mikrořadiče se sníženou spotřebou.
- Verze v pouzdru SMD.
- SPI rozhraní.
- I^2C rozhraní.
- Co nejvíce pinů podporujících externí přerušení.

Firma Atmel nabízí sice hodně variant různých mikrořadičů, u nás se ale prodávají jen ty nejběžnější. Tímto se výběr poměrně zúžil. V tabulce 3.6 uvádím údaje dvou mikrořadičů, které se mi zdály pro použití v této práci nejvhodnější. Vybíral jsem mikrořadiče z rodiny `ATMEGA`, které jsou velmi rozšířené a mají slušnou stabilitu.

Označení	ATMEGA128L-8AI	ATMEGA168V-10AU
Výrobce	Atmel	Atmel
Registry	32x8bit	32x8bit
Rychlost	max. 8MHz	max. 10MHz
Paměť programu	128KB	16KB
Paměť dat	4KB	1KB
EEPROM paměť	4KB	512B
Rozšiřitelná paměť	ano	ne
JTAG	ano	ne
Čítače/časovače	2x8bit, 2x16bit	2x8bit, 1x16bit
PWM kanály	2x8bit, 6x(2–16bit)	6x
A/D převodník	8 kanálů, 10bit	8 kanálů, 10bit
TWI	ano	ano
USART	2x	1x
SPI	1x	1x
Pouzdro	TQFP64	TQFP32
Napájecí napětí	2,7 – 5,5V	1,8 – 5,5V
Piny s vnějším přerušením	8x	24x

Tabulka 3.6: Přehled vybraných mikrořadičů

Mikrořadič `ATMEGA168V-10AU` by byl ve výběru nejoptimálnější, jenže kvůli jeho nedostupnosti a velmi dlouhých dodacích lhůt z výběru vypadl. Mikrořadič `ATMEGA128L-8AI` je sice už počtem funkcí na trochu vyšší úrovni než předchozí, je i o něco dražší, ale pro naší aplikaci asi nejvhodnější ze všech zkoumaných (a dostupných) mikrořadičů.

3.5.1 Vybraný mikrořadič

Vybraný obvod `ATMEGA128L-8AI` je vyroben technologií CMOS, která přispívá k nízké spotřebě obvodu. Je to osmibitový mikrořadič, založený na architektuře RISC (redukovaný instrukční soubor). Díky tomu dokáže vykonat až 8 MIPS (miliónů instrukcí za sekundu) při 8MHz krystalu. Má AVR jádro s 32 pracovními registry, které jsou přímo propojeny s ALU (aritmeticko-logickou jednotkou). To umožňuje, aby instrukce, která pracuje se dvěma pracovními registry mohla k těmto registrům přistupovat v jednom taktu hodin, tzn. stejně jako instrukce, která by pracovala pouze s registrem jedním. Protože tyto operace jsou jedny z nejčastějších, rychlost vykonávání instrukcí se tím podstatně zvýší.

Významy jednotlivých periférií mikrořadiče:

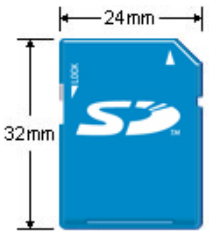
- Timer/Counter 0/1/2/3 - jsou to čítače/časovače, provádějí se s nimi různé časové operace, jako je například generování signálu na výstupu mikrořadiče nebo zjišťování, za jak dlouho se změní signál na vstupu apod.
- Rozhraní SPI - Serial Peripheral Interface slouží pro synchronní sériový plně duplexní přenos dat mezi řídícím obvodem master a vedlejším obvodem slave. Signály `MOSI` (výstup), `MISO` (vstup), `SCK` (takt hodin `SPI`).
- USART - Universal Synchronous Asynchronous Receiver Transmitter, tj. obvod pro plně duplexní synchronní nebo asynchronní přenos dat. Signály pro synchronní přenos `RXD` (přijímač), `TXD` (vysílač), `XCK` (takt přenosu). Pro asynchronní jsou to jen `RXD`, `TXD`.

- TWI - Two Wire Interface, kompatibilní s I^2C, na sběrnici může být připojeno až 128 zařízení. Rychlost až 400kHz. Použity jsou signály `SDA` (data), `SCL` (takt sběrnice).
- ADC - Analog to Digital Converter převádí analogovou hodnotu (napětí) na digitální (posloupnost bitů). 10-ti bitové rozlišení. Vlastní režim spánku pro eliminaci šumu při měření.
- JTAG - Joint Test Action Group, pro nahrávání firmware do mikrořadiče, ale i ladění aplikace za běhu. Signály `TDI` (vstupních data), `TDO` (výstupních data), `TCK` (takt přenosu), `TMS` (vybrán testovací mód).

Více viz [1].

3.6 SD karta

SD (Secure Digital) karta bude mít v našem zařízení funkci média pro ukládání dat, naměřených z jednotlivých senzorů. Je také možné místo karty SD použít kartu MMC (Multi Media Card), která je kompatibilní. SD karta používá paměťový modul typu Flash, který v sobě kombinuje výhody některých jiných typů pamětí. Nejdůležitější vlastností paměti Flash je samozřejmě schopnost uložit data trvale, tzn. při vypnutí napájení data zůstanou. Její výhodou je poměrně nízké napájecí napětí, a s tím souvisí spotřeba, která díky tomu klesá. Rozsah dovolených kapacit SD pamětí je od 8MB do 2GB, což začíná být málo, takže tvůrci karty SD vytvořili nový standard se jménem SDHC, který ale není se starým standardem SD kompatibilní. Na obrázku 3.2 můžeme vidět rozměry SD karty.



Obrázek 3.2: Rozměry karty SD

Tloušťka je 2,1mm, hmotnost 2g. Napájecí napětí je od 2,7 do 3,6 V. Karta má na svém boku ochranný přepínač, kterým můžeme zakázat zápis. Počet kontaktů je 9. Karta používá pro vyčíslení rychlosti stejný systém jako CD-ROM mechaniky. Nejrychlejší režim je 150x, tj. 22,5MB/s, což odpovídá trojnásobné rychlosti oproti CD-ROM mechanice (s rychlostí 50x). Nyní si popíšeme, jaké komunikační módy karta podporuje:

- Jedno-bitový SD mód (obousměrný přenos příkazů po 1 vodiči, obousměrný přenos dat po 1 vodiči, hodinový takt pro komunikaci, indikace přerušení)
- Čtyř-bitový SD mód (obousměrný přenos příkazů po 1 vodiči, obousměrný přenos dat po 4 vodičích, hodinový takt pro komunikaci)
- Mód SPI (vodič pro Data In - DI, vodič pro Data Out - DO, hodinový takt komunikace, indikace přerušení)

Pro komunikaci mezi mikrořadičem a SD kartou je určen právě jednoduchý režim SPI.

3.7 Hodiny reálného času

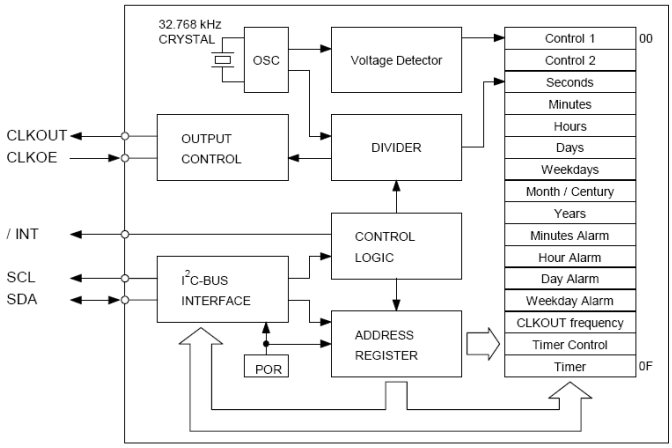
Obvod hodin reálného času obecně slouží pro aplikace, kde potřebuje znát aktuální čas. Problém je s nastavováním tohoto času po připojení obvodu k napájení. Buď hodiny připojíme k nějakému zařízení, kde nastavený čas již běží a načteme si ho, nebo ho musíme nastavit ručně. V naší práci čas přímo nepotřebujeme, ale když už budeme mít hodiny k dispozici, určitě by se časová hodnota v nějakém případě přinejmenším hodila. My budeme hodiny používat hlavně z důvodu generování stejných časových úseků, mezi kterými se bude mikrořadičem měřit z měřících modulů. Takové funkci se říká Alarm (většinou se dá generovat změna na výstupu). Při výběru obvodu bude hrát důležitou roli také spotřeba obvodu, ten bude připojen na napájení trvale. V tabulce 3.7 je přehled vybraných obvodů, které by šly v našem zařízení použít.

Označení	**RTC8564JE**	**DS1306E**
Výrobce	Epson	Dallas Semiconductor
Pouzdro	VSOJ20	TSSOP20
Krystalový oscilátor	interní	externí
Sběrnice	I^2C	SPI/3-Wire Interface
Možnost zálohy	ne	druhé napájení/vstup pro záložní baterii
Spotřeba	330nA	81μA

Tabulka 3.7: Přehled vybraných hodin reálného času

3.7.1 Vybrané hodiny

Zde je výběr obvodu poměrně jasný. Opravdu malá spotřeba, I^2C sběrnice a interní krystalový oscilátor mluví pro obvod `RTC8564JE`. Jeho blokové schéma je na obrázku 3.3.



Obrázek 3.3: Blokové schéma obvodu RTC8564JE

Tento obvod nemá žádnou paměť navíc. U většiny obvodů tato paměť je, ale nám to vadit nemusí, potřebovat jí nebudeme. Pro nastavování hodin slouží 16 registrů. Jejich adresace probíhá tak, že hodinám pošleme adresu registru, ze kterého chceme číst/zapisovat, a při dalším čtení/zápisu se tato adresa registru automaticky zvýší o 1. Tím je výrazně urychlena komunikace. Obvod může na svém výstupu generovat až dva typy přerušení, a to od Alarmu a Timeru (časovač). Alarm se používá na cyklické přerušení v intervalu například jedné hodiny nebo více.

Timer se používá spíše na kratší cyklické přerušení, nebo na přerušení, které přijde jednou a pouze indikuje vypršení námi nastaveného času. Hodiny umožňují na jednom z výstupu povolit takt z interního krystalu nebo z předděličky. Do registrů času si můžeme zapsat aktuální čas. Ten nám poběží přibližně stejně jako ten zapisovaný.

3.8 Převod USB na RS232

Mikrořadič nemá žádné USB rozhraní, pouze umí komunikovat například přes rozhraní SPI, USART apod. Nové počítače již přestávají podporovat porty COM, takže bude vhodné umožnit mikrořadiči komunikovat s PC i přes USB sběrnici. To lze s obvody od firmy FTDI. Pro převod byl vybrán obvod `FT232BM`. Velkou výhodou je, že po připojení převodníku k PC a nainstalování ovladačů VCP (Virtual COM Port, široká dostupnost - Windows/MAC/Linux), se bude obvod chovat jako klasický port COM (ale s 5V logikou, u PC se používá jiné napětí, tzn. kolem 12V, specifikace dokonce povoluje až 25V). Pokud bychom tedy na piny převodníku připojili mikrořadič přes programovací piny, mohli bychom přes rozhraní USB mikrořadič programovat. To ale dle práce v [5] nefunguje zcela v pořádku, takže musíme přistoupit k jinému řešení. Pro převodník existuje ještě jeden typ driveru, díky kterému se už nechová jako COM port, ale tak, jak si výstupy nastavíme. Proto můžeme připojit programovací piny mikrořadiče na jakýkoliv z nastavitelných výstupů převodníku.

Tento typ obvodu vyžaduje připojení externího krystalu a pokud chceme, můžeme k němu připojit ještě paměť EEPROM. Ta slouží pro jednoznačnou identifikaci obvodu v PC. Externí krystal musí mít frekvenci 6MHz, která se pak může vynásobit násobičkou 8-mi pro rychlejší komunikaci. Obvod je USB 2.0 kompatibilní.

USB sběrnice je konstruovaná pro napájení koncových zařízení. Toho my využijeme, jinak by nám převodník odebíral cenou energii z baterií. Doporučené zapojení obvodu pro napájení ze sběrnice je v [3].

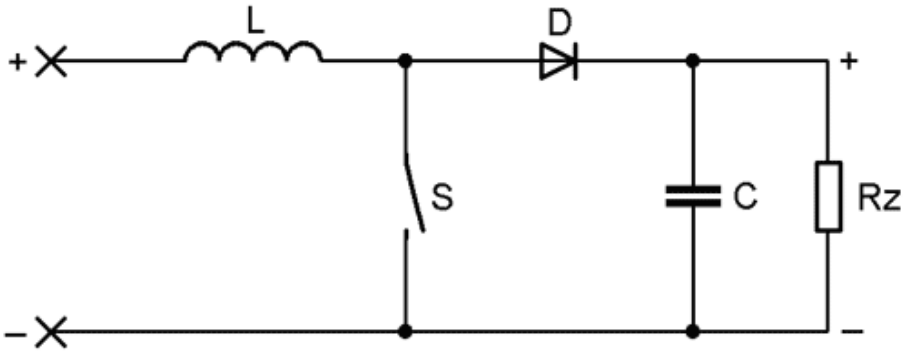
3.9 Napájecí zdroj

Pro napájení celého zařízení bude potřeba vybrat vhodný zdroj, který umožní napájení všech uvedených obvodů. Aby mělo celé zařízení co nejnižší spotřebu, bude nejlepší ho napájet co nejnižším napětím. Protože senzory byly vybírány pro napětí 5V, kdy senzory s nižším napájecím napětím nejsou vůbec na trhu, budeme se muset smířit s tím, že napájení bude pro všechny obvody 5V. Není totiž možné přidávat do obvodu nějaké další stabilizátory z důvodu co nejnižší spotřeby, jejich účinnost nebude totiž nikdy blízká 100%.

Zařízení je určeno pro měření teploty, vlhkosti a tlaku na více místech, proto bude velmi vhodné navrhnout takové napájení, které by umožňovalo zařízení napájet z baterií. Tím nastává problém, že napětí z baterií je nestálé, a při jejich zátěži klesá (vybíjení baterií). O možnost napájení zařízení regulovaným adaptérem nepřijdeme, když použijeme napětí podobné tomu u baterií. Při použití baterií velikosti AA nastává otázka, kolik jich pro zařízení použít. Pokud bychom použili tři, byla by sice výdrž celého zařízení dostatečně velká, ale pokud chceme baterie umístit přímo na zařízení, byl by problém s velikostí. Rozhodl jsem se proto pro napájení dvěma články. V ideálním případě, tzn. v nabitém stavu článků, budou baterie dávat napětí 3V. Napětí je tedy potřeba nějak zvýšit na 5V a udržet, i když napětí z baterií klesne například na 2,5V.

3.9.1 Princip step-up zdroje

To se řeší tzv step-up spínanými zdroji. Princip takového zdroje je uveden na obrázku 3.4. Jedná se o zapojení, kdy na výstupu dostáváme vyšší napětí než je napětí vstupní. Když sepne spínač S, začne se akumulovat energie do indukčnosti L. V tomto okamžiku je proud do zátěže dodáván kapacitorem C, který je již nabitý z předchozího cyklu. V okamžiku rozepnutí spínače S se napětí na indukčnosti otočí a přičte se k napájecímu napětí. O toto napětí pak bude napětí na výstupu větší. Spínání a rozepínání spínače S probíhá dostatečně rychle (v řádu kHz), takže napětí na výstupu je téměř konstantní. Nevýhodou takového zdroje je, že se nám do zařízení dostane střídavá složka napětí, kterou musíme co nejdůkladněji odfiltrovat.



Obrázek 3.4: Princip step-up spínaného zdroje

Problém nastává u SD karty, která vyžaduje napájení od 2,7 do 3,6V. Musíme tedy vymyslet co nejjednodušší řešení pro snížení napětí z 5V alespoň na 3,6V. Napadla mě myšlenka použít dvě usměrňovací diody, které by měly napěťový zlom někde kolem 0,9V každá. Ty bychom dali do série za zdroj 5V a při nějakém odběru SD karty by poté napětí kleslo na nějakou hodnotu v rozmezí od 2,7 do 3,6V.

3.9.2 Spotřeba zařízení

Abychom mohli vybrat správný zdroj, musíme zjistit spotřebu celého zařízení, resp. maximální spotřebu celého zařízení. Ta vychází ze spotřeby jednotlivých obvodů, jejichž jednotlivé maximální odběry proudu při napětí 5V si musíme zjistit. Odhad odběrů jednotlivých obvodů tedy uvádím v tabulce 3.8.

Obvod	**Typická spotřeba**	**Maximální spotřeba**
SMT160-30-18	180μA	200μA
UTI03	2,5mA	3mA
MPXH6115AC6U	6mA	10mA
TS555I	65μA	200μA
ATMEGA128L-8AI	11mA	19mA
SD karta	-	60mA
FT232BM	-	-
RTC8564JE	330nA	800μA
Celkem max. spotřeba	**-**	**93,2mA**

Tabulka 3.8: Maximální celková spotřeba

U obvodu pro komunikaci s PC `FT232BM` není uvedena spotřeba, protože napájení tohoto obvodu bude řešeno přes sběrnici USB. Spotřeba mikrořadiče `ATMEGA128L-8AI` bude ve výsledku běžně nižší, než jak bylo uvedeno, protože obvod bude většinu času v režimu spánku.

Požadavky na spínaný zdroj jsou tedy:

- Výstupní napětí 5V, při vstupním napětí od 1,8 do 3V.
- Schopnost dávat proud kolem 200mA.
- Co nejvyšší účinnost.

Práci s vyhledáváním vhodných spínacích zdrojů mi ulehčil můj vedoucí práce. K dispozici měl totiž několik obvodů `NCP1421-D`, takže mohl pro mou práci jeden poskytnout. Tento obvod je pro naší aplikaci ideální, je totiž speciálně vyvinutý pro zařízení s bateriemi. Některé jeho parametry jsou v tabulce 3.9.

Označení	**NCP1421-D**
Výrobce	ON Semiconductor
Pouzdro	Micro8
Typ zdroje	step-up
Vstupní napětí	1 – 5V
Výstupní napětí	1,5 – 5V
Max. výstupní proud	600mA
Max. účinnost	94%
Max. spínací frekvence	1,2MHz

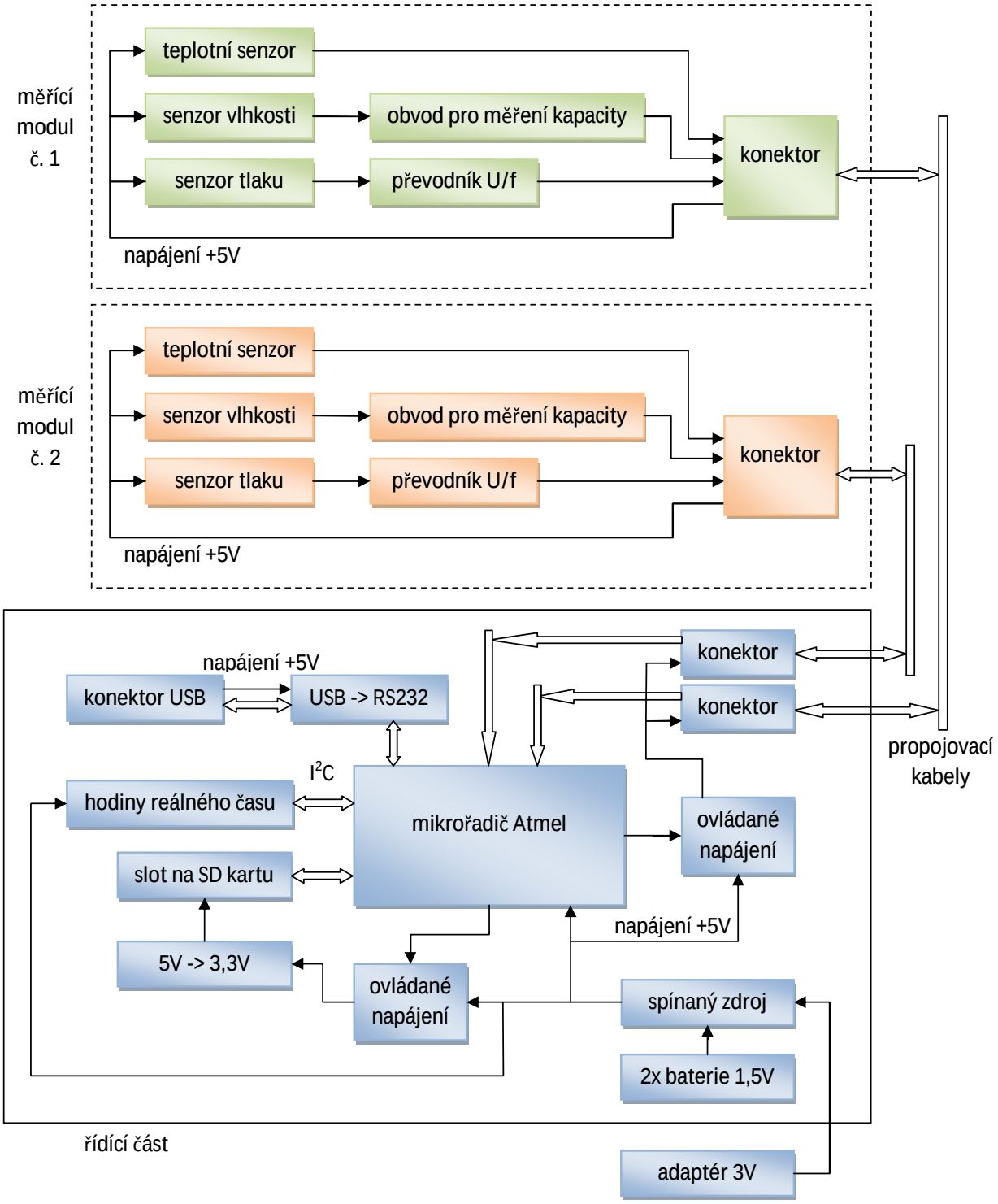
Tabulka 3.9: Parametry vybraného zdroje

3.10 Návrh řešení

V této podkapitole se budeme zabývat návrhem hardwarového řešení pro tuto práci. Části návrhu rozdělíme na:

- Řídící část zařízení, která zahrnuje mikrořadič, slot na SD kartu, hodiny reálného času, převodník z USB na RS232, zdroj pro napájení všech potřebných obvodů a obvody pro zapínání a vypínání napájení pro některé obvody.
- Měřící část zařízení, která zahrnuje senzor teploty, senzor vlhkosti spolu s obvodem pro měření kapacity, tlakový senzor spolu s převodníkem z napětí na frekvenci. Připojit k řídící části lze maximálně dva měřící moduly.

Přibližné blokové schéma, jak by mohl obvod vypadat je na obrázku 3.5. Měřící části jsou ohraničeny přerušovanou čarou, protože je lze odpojit. Dvojité čáry znamenají více vodičů (sběrnice).



Obrázek 3.5: Blokové schéma navrženého zařízení

3.10.1 Měřící část

Měřící část se skládá z měřícího modulu č. 1 a měřícího modulu č. 2. Tyto moduly jsou shodné, mohou se však nacházet každý na jiném místě. Měřící modul se stará o převedení teploty, vlhkosti nebo tlaku na jiné veličiny, které by byly měřitelné mikrořadičem.

Mikrořadič dokáže v jednom okamžiku určit buď logickou úroveň nebo přímo napětí (námi vybraný obsahuje A/D převodník). Měření napětí, jak už bylo řečeno během analýzy, pro nás není vhodné kvůli možnému výskytu rušení mezi senzorem a mikrořadičem, protože propojovací kabel může být dlouhý například i deset metrů.

Zbývá tedy převod uvedených veličin na signál dvou časově oddělených logických úrovní (obdélníkový nebo-li digitální signál), který je proti rušení mnohem více odolný. V našem případě je potřeba definovat, kdy námi vybraný mikrořadič interpretuje přečtenou napěťovou úroveň jako „logickou nulu", a kdy jako „logickou jedničku". U mikrořadičů `ATMEGA` je „log. 0" definována jako napětí v rozmezí -0,5V až 0,2V·V_{CC} a „log. 1" je definována v rozmezí 0,6V·V_{CC} až V_{CC}+0,5V. V_{CC} je napájecí napětí, tzn. 5V. Přepočet logických úrovní je uveden v tabulce 3.10.

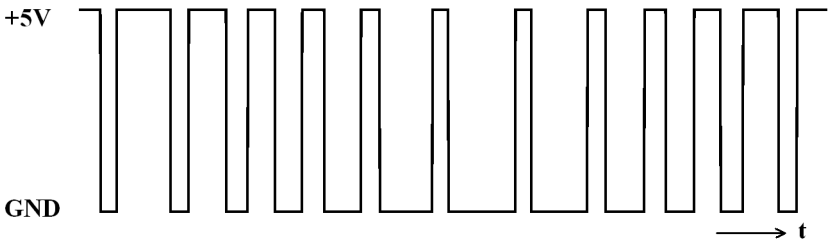
Logická úroveň	Napětí od	Napětí do
„log. 0"	-0,5V	1V
„log. 1"	3V	5,5V

Tabulka 3.10: Rozsahy napětí u logických úrovní mikrořadiče ATMEGA při 5V napájení

Logická úroveň mezi napětím od 1 do 3V není definována, tzn. pokud se na vstupu mikrořadiče objeví taková hodnota, bude interpretována jako úroveň, která byla naposledy v definovaném rozsahu. Signály, které budou vést od senzorů, se tedy musejí vyskytovat v tabulce definovaných úrovních. S tímto kritériem už bylo počítáno při výběru senzorů a obvodů pro převod, takže všechny jsou TTL (Transistor Transistor Logic) kompatibilní, tzn. jejich výstup je přibližně buď 0V nebo 5V.

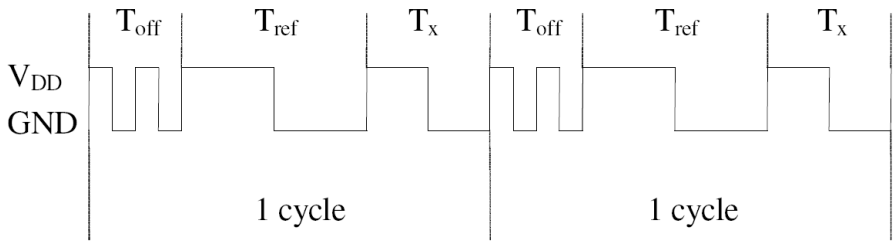
Přehled signálů na výstupu měřícího modulu:

- Signál PWM (Pulse-Width Modulation). Na výstupu je tedy signál, který při konstantní frekvenci (i periodě) mění šířku impulzu (doba, kdy je signál v „log. 1") podle teploty. Signál je přímo výstupem senzoru `SMT160-30-18`. Na obrázku 3.6 je zobrazen příklad takového signálu.



Obrázek 3.6: Možný průběh signálu PWM

- Signál s třemi proměnnými periodami (1. se skládá z dvou poloviční), které jsou generovány v daném pořadí. Frekvence opakování je konstantní. Periody signálu generuje obvod `UTI03` podle hodnoty kapacity senzoru vlhkosti `SMTHS07`. Příklad takového průběhu je na obrázku 3.7.



Obrázek 3.7: Možný průběh signálu z obvodu UTI03

- Signál s proměnnou frekvencí. Tento signál generuje obvod `TS555I` podle výstupního napětí tlakového senzoru `MPXH6115AC6U`.

Tyto signály jsou připojeny už přímo do řídící části. Z řídící části je přivedeno 5V napájení a zem.

3.10.2 Řídící část

Zpracování jednotlivých signálů bude mikrořadič provádět jejich vzorkováním. Tím získáme rekonstruovaný signál, z kterého už pak snadno dopočítáme hodnoty ze senzorů. Frekvence vzorkování zvolíme co nejvyšší, abychom zajistili co největší přesnost rekonstruovaného signálu. Na rychlosti měření zde totiž moc nezáleží. Maximální rychlost vzorkování mikrořadiče je 8MHz, frekvence měřeného signálu je max. v řádu kHz, takže přesnost měření signálu bude velmi dobrá.

Řídící část nebude obsahovat pouze mikrořadič pro zpracování signálů z měřící části, ale i další podpůrné obvody. Data ze senzorů potřebujeme někam ukládat. Na to slouží SD paměť, která má dostatečnou kapacitu i pokud budeme chtít ukládat data při dlouhodobém měření.

Z dlouhodobého hlediska není potřeba měření provádět příliš často. Naprosto postačí pokud budeme měřit hodnoty ze senzorů jednou za hodinu. Aby mikrořadič a celá měřící část neběžely hodinu zbytečně naprázdno, jsou k mikrořadiči připojeny hodiny reálného času, které budou generovat jednou za hodinu přerušení na jeho vstupu. Díky tomu může mikrořadič po měření vypnout napájení měřící části a SD karty, a sám se přepnout do režimu spánku, při kterém se ušetří cenná energie z napájecích baterií. Při přerušení od hodin se přepne zpět z režimu spánku, připojí napájení měřící části a SD karty, naměří další hodnoty, které uloží, odpojí napájení a zase se uspí.

Řídící část obsahuje také obvod pro převod dat z USB sběrnice na sériová data (RS232) a zpět, protože mikrořadič nedokáže se sběrnicí USB komunikovat přímo. Díky tomu můžeme propojit PC a mikrořadič, i když PC nemá rozhraní RS232.

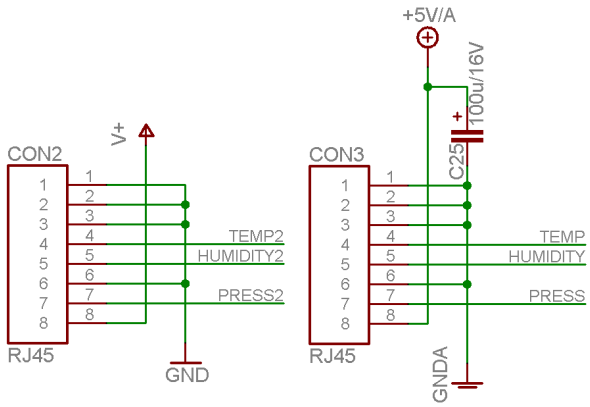
Dalším obvodem, který je umístěn v řídící části, je napájecí zdroj. Ten je určen pro napájení všech uvedených obvodů včetně měřících. Jedná se o spínaný zdroj, který konvertuje napětí z baterií (také umístěny v řídící části) nebo z externího zdroje na napájecí napětí obvodů 5V (napájení pro SD kartu je dodatečně sníženo).

4 Realizace

V této části se budeme zabývat realizací celého zařízení a vývojem obslužného softwaru pro mikrořadič. Podkapitoly jsou rozděleny opět na měřící a řídící část, ke kterým navíc přibyla podkapitola vývoj obslužného softwaru.

4.1 Hardware měřící části

Pro měřící moduly bylo potřeba vybrat vhodný typ konektoru pro připojení k řídící části. Minimální počet kontaktů konektoru je 5. Běžný řadový konektor byl zavržen z důvodu možného špatného připojení propojovacího kabelu. Proto byl vybrán konektor RJ-45, který má proti špatnému připojení zarážky. Jeho výhodou je i velká dostupnost různých TP (Twisted Pair, neboli kroucená dvojlinka) kabelů. Nevýhodou jsou jeho rozměry, i tak se ale povedlo zrealizovat dostatečně malý modul. Při zapojování konektoru bylo potřeba dávat pozor na možnost verze kříženého kabelu. Z obrázku 4.1 je vidět, že případné překřížené vodiče jsou uzemněné a signální vodiče zůstávají na stejné pozici. Vlevo je konektor z řídící části pro měřící modul č. 2 a vpravo konektor měřícího modulu.



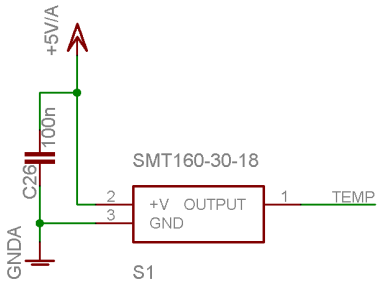
Obrázek 4.1: Obrázek zapojení konektorů na obou stranách

4.1.1 Teplotní senzor

Teplotní senzor, jak už bylo řečeno v návrhu, může být vyvedený z měřícího modulu přímo, protože jeho výstup má již námi požadovaný signál. U jeho napájení je pouze přidán tzn. blokovací kondenzátor pro eliminaci frekvenčních vlivů v napájení. Na obrázku 4.2 je jeho zapojení.

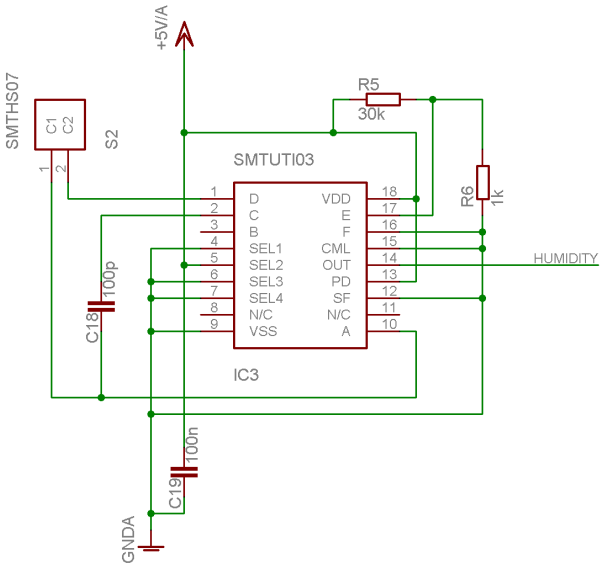
4.1.2 Senzor vlhkosti

Senzor vlhkosti je kapacitní, takže byl přidán obvod `UTI03` pro měření jeho kapacity. Nejprve bylo potřeba zvolit vhodný měřící režim tohoto obvodu. Režimů měření je celkem 16, z toho 5 jich je pro měření kapacity a zbytek pro měření odporů a odporových můstků. Pro naše měření je vhodný pouze režim č. 4, který umožňuje měření kondenzátoru s proměnným rozsahem do 300pF. Senzor `SMTHS07` toto kritérium splňuje, jeho proměnný rozsah je maximálně 100pF. Číslo režimu v binární hodnotě odpovídá nastavovaným vývodům SEL1 – SEL4. Vývod SEL2 jsme tedy připojili na napájení 5V a zbytek vývodů byl uzemněn. Vývod SF slouží k výběru



Obrázek 4.2: Obrázek zapojení teplotního senzoru

rychlosti měření. Pokud bychom ho připojili na 5V, měření by probíhalo rychlostí 10ms pro jeden cyklus. Nakonec jsme tento vývod připojili na zem (rychlost 100ms pro cyklus), tím získáme přesněji rekonstruovaný signál v mikrořadiči. Vývod PD slouží k vypnutí napájení obvodu, to my nepotřebujeme, protože o vypínání napětí celé měřící části se stará řídící část. Vývod byl tedy připojen na 5V (vždy zapnutý). Vývod CML musí být v našem měřícím režimu připojen na zem, jeho funkcí se dále zabývat nebudeme. Zapojení vývodů A – F je různé podle zvoleného režimu, pro náš případ je zapojení znázorněno na obrázku 4.3.



Obrázek 4.3: Obrázek zapojení senzoru vlhkosti a obvodu pro převod na periodu

Vývody A,B,C,D slouží pro připojení a měření tří různých kapacit, kde vývod A je společná elektroda. Vývod B nebyl připojen, protože ten slouží k měření parazitní kapacity mezi vývody B – A. Stejná kapacita je totiž i mezi vývody C – A a D – A, takže můžeme její hodnotu (délku periody přímo úměrné kapacitě) odečíst od změřených hodnot C a D (také se nejedná přímo o hodnoty kapacit, ale o délky period). Na vývod C je připojena referenční kapacita. Tu pak budeme při vývoji software měřit přesně. Vývod D je určen pro připojení senzoru. Senzor ale nebudeme zatím pájet na desku, protože je nejdříve potřeba přesně změřit referenční kapacitu. Celková kapacita na vývodech nesmí překročit 500pF, aby odchylka obvodu od linearity nebyla větší jak 10^{-3}. Podle toho jsme zvolili referenční kapacitu. Kapacita senzoru je maximálně 380pF, takže referenční byla zvolena 100pF.

Kapacita na vývodech B,C,D a A se měří střídavým napětím. Rozkmit tohoto napětí je určen

velikostí stejnosměrného napětí na vývodech E a F. Toto napětí musí vyhovovat podmínce:

$$V_{EF} < \frac{K_V}{C_{max}}$$

, kde konstanta $K_V = 60\,V{\cdot}pF$ a hodnota C_{max} je maximální hodnota kapacit C_{BA}, C_{CA} a C_{DA} vyjádřená v pF. Největší kapacitu může mít senzor, tedy $C_{max} = 380\,pF$. Podmínka je tedy:

$$V_{EF} < \frac{60}{380} < 0,158\,V$$

Vzorec pro výpočet napětí V_{EF} je:

$$V_{EF} = V_{DD}{\cdot}\frac{R_6}{R_5 + R_6}$$

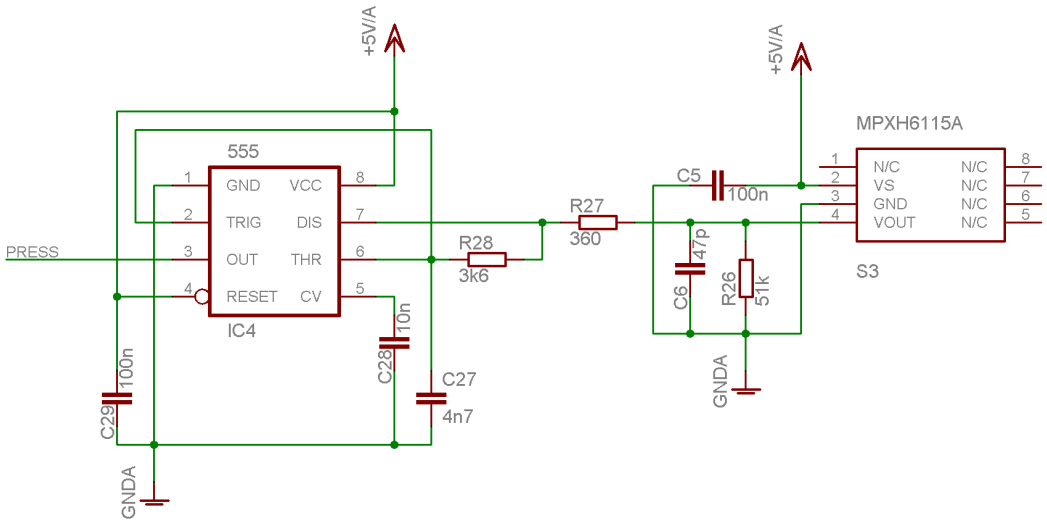
, kde V_{DD} je napájecí napětí (5V) a R_5, R_6 jsou odpory, které chceme určit. R_6 zvolíme 1kΩ a R_5 dopočítáme:

$$R_5 = \frac{V_{DD}{\cdot}R_6}{V_{EF}} - R_6 = \frac{5{\cdot}1000}{0,158} - 1000 = 30646\,\Omega$$

Odpor R_5 jsme tedy zvolili 30kΩ. Nakonec je potřeba zkontrolovat, zda-li časová konstanta obvodu nepřekročí hodnotu 500ns. Ta je rovna $C_{tot}{\cdot}(R_5//R_6)$, kde $C_{tot} = C_{BA} + C_{CA} + C_{DA}$. C_{BA} je parazitní kapacita, tu odhadneme přibližně na 30pF. $C_{tot} = 510\,pF$ a časová konstanta tedy vychází na 494ns. Těsně tedy dané kritérium 500ns splňujeme.

4.1.3 Senzor tlaku

Senzor tlaku má napěťový výstup. V návrhu už bylo uvedeno, že napětí budeme převádět na frekvenci. Pro převod byl vybrán časovač `TS555I`, který zapojíme podle doporučeného schématu pro astabilní klopný obvod v [12], ale odpor R_{27} nepřipojíme na napájení obvodu, ale na výstup senzoru tlaku. Znázorněno je to na obrázku 4.4.



Obrázek 4.4: Obrázek zapojení senzoru tlaku a obvodu pro převod na frekvenci

Pokud by bylo na výstupu senzoru napětí stejné jako napájecí napětí u časovače, platí tyto vzorce (viz [12]):

$$T = 0,693(R_{27} + 2{\cdot}R_{28})C_{27}\,[s]$$

$$f = \frac{1}{T} = \frac{1,44}{(R_{27} + 2{\cdot}R_{28})C_{27}}\,[Hz]$$

Střída signálu se potom vypočítá:

$$D = \frac{R_{28}}{R_{27} + 2{\cdot}R_{28}}$$

, kde D bude poměr mezi délkou signálu v „log. 0" a celkovou periodou. My jsme zvyklí střídu určovat poměrem délky signálu v „log. 1" k délce periody, proto si hodnotu D přepočítáme jako $D = 1 - D$.

Hodnoty kondenzátoru C_{27} a odporů R_{27} a R_{28} tedy musíme zvolit tak, aby vyhovovaly námi požadovaným podmínkám. Ty si zvolíme takto:

- Střídu požadujeme s hodnotou blízkou $D = 0,5$ (50%). Ta nesmí být přesně 0,5, jinak by odpor R_{27} vyšel s hodnotou 0Ω (nežádoucí), takže zvolíme hodnotu o něco vyšší, například 0,525.
- Frekvenci požadujeme s hodnotou blízkou $f = 40\,kHz$.

Na začátku výpočtu si zvolíme hodnotu $R_{28} = 3600\,\Omega$, a můžeme dopočítat další hodnoty. Nejdříve dopočítáme hodnotu R_{27} pomocí vzorce střídy:

$$R_{27} = \frac{R_{28}}{1 - D} - 2{\cdot}R_{28} = \frac{3600}{1 - 0,525} - 2{\cdot}3600 = 378,95\,\Omega$$

Odpor R_{27} tedy volíme (podle dostupné řady) 360Ω. Dále už jen spočítáme hodnotu C_{27} dle požadované frekvence:

$$C_{27} = \frac{1,44}{(R_{27} + 2{\cdot}R_{28})f} = \frac{1,44}{(360 + 2{\cdot}3600)40000} = 4,76\,nF$$

Hodnotu C_{27} tedy volíme 4,7nF. Kondenzátor vybereme svitkový z důvodu dobré stability při změně teploty.

Při 5V na vstupu časovače (R_{27}) tedy bude výstupní frekvence přibližně 40kHz. Pokud bude napětí nižší, kondenzátor C_{27} se bude nabíjet delší dobu, takže se nám prodlouží perioda (sníží frekvence) a změní střída (to nám nevadí, nás zajímá jen délka periody). Abychom mohli určit nějaký vzorec pro přepočet z naměřené frekvence na napětí, musíme proměřit hodnoty frekvencí při změnách vstupního napětí (od 0 do 5V). Více o tom, jak probíhalo měření, se dozvíte v kapitole Testování.

4.2 Hardware řídící části

4.2.1 Mikrořadič

Mikrořadič je hlavní obvod v celém zařízení, je propojen se všemi obvody. Při připojování jednotlivých vodičů k mikrořadiči jsme museli dbát na to, jakou funkci má daný vývod (pin).

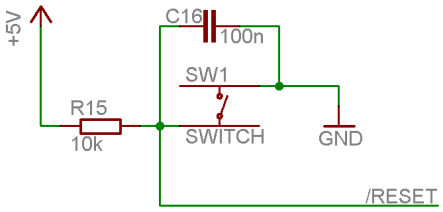
Piny mikrořadiče jsou rozděleny po osmi na porty. Výjimkou je port G, který má pinů pouze pět. Každý pin ze všech těchto portů má standardně funkci jako vstup/výstup. Tzn. u každého pinu můžeme zjistit jeho logickou úroveň (vstup) nebo ho můžeme nastavit na námi pořadovanou logickou úroveň (výstup). U každého pinu si také můžeme nastavit při vstupním režimu připojení pull-up rezistoru. Pokud tedy bude tento pin nějakým obvodem uzemněn, přečteme hodnotu jako „log. 0", a pokud nebude uzemněn přečteme „log. 1". Taková logika je negativní (obrácená), u vývodu s touto logikou se používá tranzistor s otevřeným kolektorem.

Většina portů mikrořadiče má ale tzv. alternativní funkce, takže piny nemusíme požít pouze jako klasický vstup/výstup. Některé piny můžeme například použít pro komunikaci přes SPI rozhraní, TWI rozhraní, pro externí přerušení apod. Zde tedy popíšeme na jaký pin je připojen jaký obvod:

- Signály z teplotních senzorů jsou připojeny na piny, které dokáží generovat externí přerušení (`TEMP1` na `INT4`, `TEMP2` na `INT6`). Pro měření střídy se totiž hodí, když bude generováno přerušení každou náběžnou i sestupnou hranu signálu. Uvedenou funkci mají pouze piny s alternativní funkcí `INT4` až `INT7`, přerušení generují synchronně (s taktem hodin mikrořadiče).
- Převedené signály od senzorů vlhkosti (`HUMIDITY1` a `HUMIDITY2`) jsou připojeny také na piny, generující přerušení (`INT5` a `INT7`). Už ale nepotřebujeme generovat přerušení jak při náběžné tak sestupné hraně. Pro měření periody postačí generování přerušení pouze na náběžnou hranu.
- Převedené signály od senzorů tlaku (`PRESS1` a `PRESS2`) jsou připojeny na piny s alternativní funkcí pro takt čítače/časovače 1 (`T1`) a pro takt Č/Č 2 (`T2`). Tyto piny můžeme nastavit tak, aby se inkrementovala hodnota Č/Č 1 (nebo Č/Č 2) při každé náběžné nebo sestupné hraně signálu. Pro měření frekvence je tato funkce ideální, můžeme tak například změřit čas, za jak dlouho jeden z Č/Č dosáhne nějaké hodnoty.
- Spínače pro ovládání napájení pro měřící část (`SW1`) a SD kartu (`SW2`) jsou zapojeny na piny `PB6` a `PC7`. Pro tyto piny není potřeba zapínat nějakou alternativní funkci, postačí pouze nastavování výstupu pinů do „log. 0" a „log. 1".
- Vývody SD karty jsou zapojeny na piny `PC0` až `PC5`, toto zapojení bylo převzato z jiné práce [13].
- Přerušení od hodin reálného času (`INT`) je přivedeno na pin, který dokáže generovat externí přerušení (`INT2`). Tento pin je vhodné nastavit tak, aby generoval přerušení při sestupné hraně. Logika výstupního pinu hodin je totiž negativní.
- Sběrnice I^2C je připojena na piny, které vedou z vnitřního obvodu TWI mikroradiče (`SDA`, `SCL`). Obvod TWI je se sběrnicí I^2C kompatibilní.
- Na vstupní pin `PF0` je připojen přesný hodinový takt z hodin reálného času. Tento takt není v práci použit, je připojen pouze pro případ potřeby.
- K úplně stejnému účelu jako u přerušení od hodin slouží výstup ze zdroje napájení `LBO` připojený na `INT3`. Mikrořadič pak bude generovat přerušení, když dojde k vybití napájecích baterií.
- Vývody `XTAL1` a `XTAL2` slouží k připojení krystalického oscilátoru.
- Piny `PDI/RXD`, `PDO/TXD`, `SCK` a `/RESET` slouží buď pro programování mikrořadiče přes rozhraní SPI, nebo pro komunikaci s ním přes USART (k tomu slouží už pouze piny `PDI/RXD` a `PDO/TXD`).
- Piny `VCC` a `AVCC` jsou povinně připojeny na napájení 5V, piny `GND` jsou potom povinně připojeny na zem

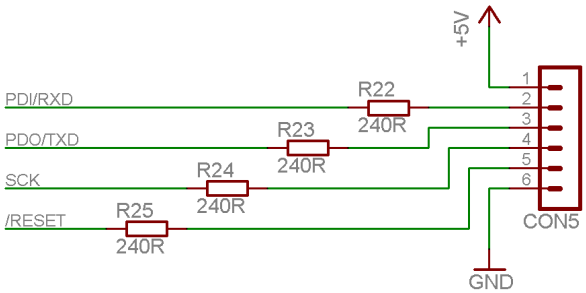
Aby bylo možné mikrořadič resetovat ručně, byl na plošný spoj přidán spínač. Reset mikrořadiče je aktivní v „log. 0", takže bylo potřeba na pin přivést napětí 5V. To bylo provedeno tak, že

mezi reset a napájení byl zařazen dostatečně velký pull-up rezistor, který je přes stisknutý spínač uzemněn. Paralelně se spínačem byl zapojen kondenzátor pro případnou eliminaci zákmitů při stisku nebo povolení spínače. Při stisku tlačítka tedy povede proud (zanedbatelný) z napájení přes pull-up na zem a z pinu resetu (přes vnitřní pull-up) také na zem. Na obrázku 4.5 je znázorněno takové zapojení.



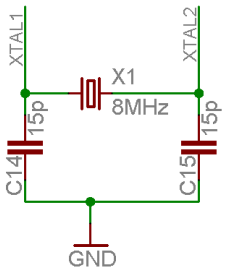
Obrázek 4.5: Obrázek zapojení resetovacího spínače

Programování mikrořadiče lze provádět paralelně, sériově nebo přes JTAG rozhraní. My se zaměříme pouze na sériové programování, které je označeno jako ISP (In System Programming). Sériové programování se provádí přes SPI rozhraní mikrořadiče a lze jím programovat jak EEPROM paměť tak Flash paměť pro firmware. Programování u tohoto mikrořadiče probíhá přes piny `PDI` (vstup sériových dat), `PDO` (výstup sériových dat), `SCK` (hodiny sériové komunikace) a `/RESET`. Zde bylo potřeba dát pozor, protože programování u jiných obvodů `ATMEGA` běžně probíhá místo pinů `PDI`, `PDO` přes piny `MISO`, `MOSI`. U tohoto mikrořadiče jsou piny `PDI`, `PDO` určeny pouze pro programování pamětí přes rozhraní SPI a piny `MOSI`, `MISO` pouze pro vstupní/výstupní komunikaci mikrořadiče s okolím (přes SPI). Pro aktivaci programovacího módu a při programování je nutné držet pin `/RESET` v „log. 0". Abychom mohli mikrořadič programovat i jinak než přes USB převodník, byly všechny potřebné programovací piny vyvedeny na konektor. Na tento konektor můžeme potom připojit jednoduchý programátor pro sériový nebo paralelní port PC. Schéma zapojení konektoru je na obrázku 4.6.



Obrázek 4.6: Obrázek zapojení konektoru pro sériové programování

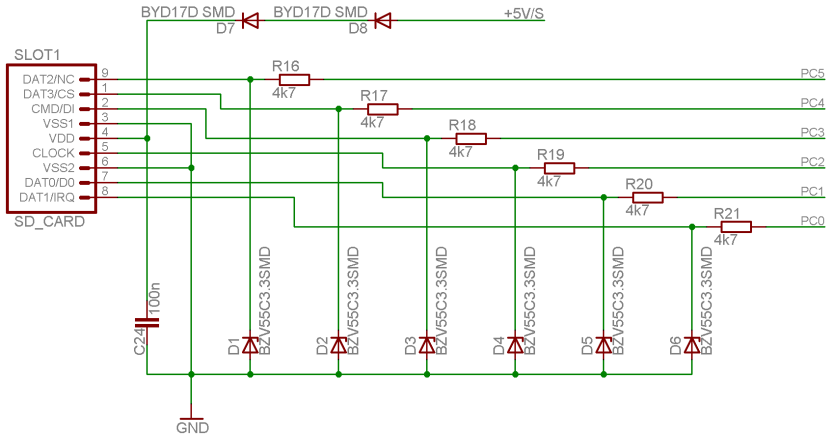
Mikrořadič má sice integrovaný oscilátor, ten ale není pro naší potřebu dost přesný, ani dostatečně rychlý. Proto využijeme možnost připojit externí krystalový oscilátor. Jeho maximální frekvence je pro náš mikrořadič 8MHz (verze s nižší spotřebou). Připojení krystalu je na obrázku 4.7, toto zapojení je podle doporučeného schématu. Hodnoty kapacit kondenzátorů jsou dány podle kmitočtu krystalu od 12 do 22pF.



Obrázek 4.7: Obrázek zapojení krystalu

4.2.2 SD karta

SD karta v zařízení slouží pro zápis naměřených dat ze senzorů. Čtení dat nebude potřeba. Karta podporuje 3 komunikační režimy se čtecím/zapisovacím zařízením. Ty dva složitější vyžadují komunikaci s kartou v režimu SD módu, jednodušší vyžaduje komunikaci přes rozhraní SPI. My jsme vybrali režim SPI. Je sice mnohem pomalejší, to nám ale vadit nebude, protože zápis na kartu bude probíhat jednou za hodinu a ještě k tomu ve velmi krátkých blocích (v řádu jednotek bajtů). Zapojení slotu pro SD kartu je na obrázku 4.8. Kromě napájení je převzato z jiné práce [13].



Obrázek 4.8: Obrázek zapojení slotu pro SD kartu

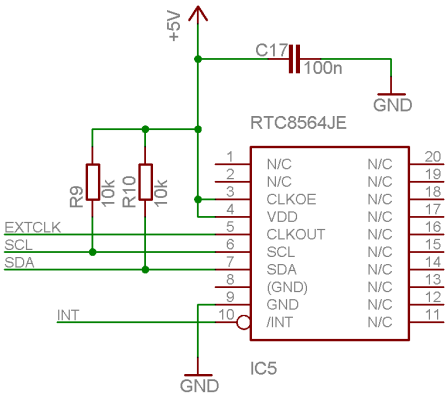
Aby bylo napájecí napětí karty v rozmezí 2,7 až 3,6V, bylo sníženo z napětí 5V dvěma usměrňovacími diodami. Na každé se sníží napětí asi o 0,9V. Přivedené napětí k SD kartě se v době nepoužívání vypíná spínačem (tranzistor). Ten ovládá mikrořadič. Vývody ze slotu pro SD kartu jsou připojeny na vstupní/výstupní piny mikrořadiče. Nejsou tedy připojeny na hardwarové rozhraní SPI. Z tohoto důvodu musí být SPI rozhraní mezi kartou a mikrořadičem řešeno softwarově. Nekompatibilita mezi 3,3V logikou SD karty a 5V logikou mikrořadiče je řešena tak, že směrem od mikrořadiče ke kartě je 5V logika stabilizována na 3,3V Zenerovými diodami (přebytek napětí je přes odpory uzemněn), a opačným směrem se s logikou neděje nic (mikrořadič interpretuje „log. 1” jako úroveň napětí od 3 do 5,5V).

4.2.3 Hodiny reálného času

K nastavení různých parametrů hodin jsou použity vnitřní registry, ke kterým lze přistupovat přes sběrnici I^2C. Sběrnice je obousměrná a používá dva vodiče `SDA` (datový vodič) a `SCL` (hodiny sběrnice). Tyto vodiče musí být externě připojeny pull-up rezistory na napájení 5V, protože porty jednotlivých obvodů, připojených na sběrnici, jsou realizovány tranzistory s otevřeným kolektorem. Tím je umožněno na sběrnici připojit více zařízení. Velikost vhodného pull-up rezistoru je dána vzorcem:

$$R = \frac{t_r}{C_{BUS}}$$

, kde t_r je maximální doba náběžné hrany jednoho ze signálů `SDA`, `SCL` a C_{BUS} je maximální parazitní kapacita těchto signálových vodičů. Hodnotu t_r zjistíme podle charakteristik (v datasheetu) jednotlivých obvodů připojených na sběrnici. My máme na sběrnici připojeny jen hodiny reálného času a mikrořadič (přes rozhraní TWI, které je s touto sběrnicí kompatibilní). Maximální doba t_r je u obou stejná, tj. 300ns. Parazitní kapacitu C_{BUS} odhadneme kolem 30pF. Výsledný pull-up rezistor tedy vychází na 10kΩ. Pokud bychom chtěli využívat maximální rychlost sběrnice 400kHz, bylo by vhodné hodnotu pull-upů o něco snížit (nepotřebujeme, o něco by se zvýšila spotřeba). Schéma zapojení je na obrázku 4.9.

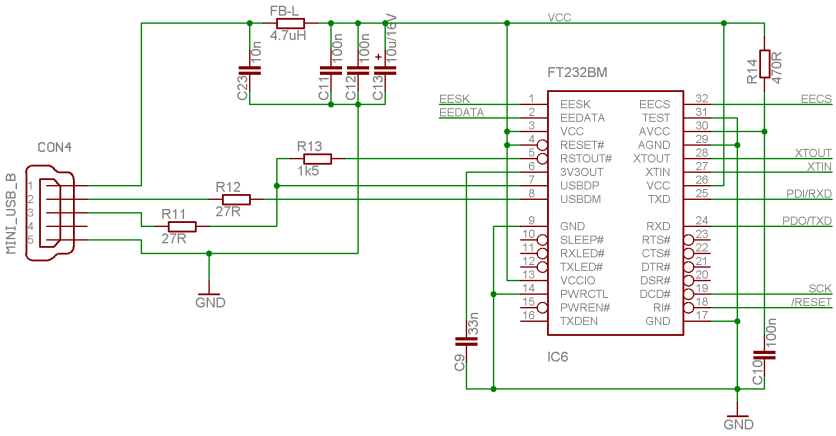


Obrázek 4.9: Obrázek zapojení hodin reálného času

Vývod `/INT` generuje v nastavený okamžik přerušení pro mikrořadič. Tento vývod používá tranzistor s otevřeným kolektorem (negativní logika). Vývod `CLKOE` je připojen do „log. 1", takže výstupní hodinový signál `CLKOUT` je povolen. Lze ho případně zakázat modifikací vnitřních registrů obvodu.

4.2.4 Převod USB na RS232

Jak už bylo řečeno v analýze, výstupy převodníku si můžeme nastavovat (při správném driveru), takže připojení programovacích pinů mikrořadiče (`PDI`, `PDO`, `SCK`) můžeme provést jako na obrázku 4.10. Protože programovací piny (pouze `PDI`, `PDO`) jsou na stejném vývodu jako piny obvodu UART (`RXD`, `TXD`), můžeme s mikrořadičem i komunikovat. Vše záleží na komunikačním softwaru.

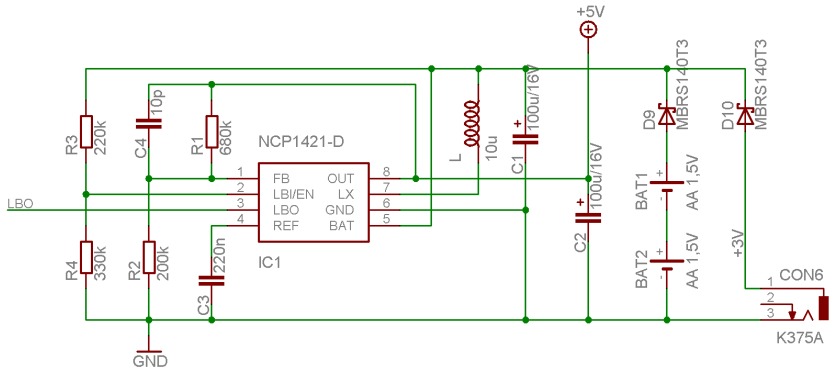


Obrázek 4.10: Obrázek zapojení převodníku USB

Zapojení je podle doporučeného v [3]. Jedná se o napájení převodníku ze sběrnice USB. Součástka FB-L (Ferrite Bead - feritové jádro) je realizována cívkou s hodnotou 4,7μH. K obvodu je připojený krystal 6MHz (piny `XTIN`, `XTOUT`), jeho zapojení je podobné jako u mikrořadiče, takže ho zde nebudeme uvádět (více ve schématu v příloze). Dále je na piny, začínající EE, připojena EEPROM pamět. Na plošném spoji jí ale pájet nebudeme, zůstane nám pájecí místo v případě potřeby.

4.2.5 Napájecí zdroj

Napájecí zdroj byl navržen tak, aby splňoval požadavky uvedené v kapitole Analýza. Při návrhu jsme vycházeli z datasheetu vybraného spínaného zdroje `NCP1421-D` [7]. Zapojení na obrázku 4.11 je podle doporučeného.



Obrázek 4.11: Obrázek zapojení napájecího zdroje

Pro návrh tohoto zdroje bylo potřeba určit tyto vstupní a výstupní parametry:

- Vstupní napětí od 1,8 do 3V, typicky $V_{IN} = 2,4\,V$.
- Výstupní napětí $V_{OUT} = 5\,V$.
- Výstupní proud $I_{OUT} = 100\,mA$ (maximální).
- Indikace vybitých baterií při $V_{LB} = 2\,V$.

- Zvlnění výstupního napětí $V_{OUT-RIPPLE} = 45\,mV_{p-p}$ (napětí špička-špička) při proudu I_{OUT}.

Výpočet zpětnovazební (FB - feedback) sítě odporů probíhal následovně:

$$R_1 = R_2(\frac{V_{OUT}}{V_{REF}} - 1)$$

, kde V_{REF} je napětí vnitřního referenčního zdroje obvodu s hodnotou 1,2V. Hodnotu odporu R_2 jsme zvolili 200kΩ a dopočítali jsme hodnotu R_1:

$$R_1 = 200\,k(\frac{5}{1,2} - 1) = 633,33\,k\Omega$$

V odporové řadě jsou jen odpory 620kΩ nebo 680kΩ. My jsme vybrali 680kΩ.

Dále provedeme výpočet odporového děliče pro indikaci vybitých baterií (LB - low battery):

$$R_3 = R_4(\frac{V_{LB}}{V_{REF}} - 1)$$

Odpor R_4 zvolíme 330kΩ a dopočítáme hodnotu R_3:

$$R_3 = 330\,k(\frac{2}{1,2} - 1) = 220\,k\Omega$$

Stanovíme průměrný proud I_{LAVG}, procházející cívkou při maximálním proudu I_{OUT}:

$$D = 1 - \frac{V_{IN}}{V_{OUT}} = 1 - \frac{2,4}{5} = 0,52$$

$$I_{LAVG} = \frac{I_{OUT}}{1 - D} = \frac{0,1}{1 - 0,52} = 208,33\,mA$$

a z této hodnoty vymezíme zvlněný špičkový proud $I_{RIPPLE-P}$ na 20% I_{LAVG}. Hodnotu cívky L pak můžeme vypočítat takto:

$$L = \frac{V_{IN} \cdot t_{ON}}{2 \cdot I_{RIPPLE-P}} = \frac{2,4 \cdot 0,75\,\mu}{2 \cdot 41,66\,m} = 21,6\,\mu H$$

, kde t_{ON} je max. čas sepnutí vývodu `LX`. Cívka se může u tohoto obvodu vybírat pouze v hodnotách od 3 do 10μH, takže my jsme s ohledem na vypočtenou hodnotu vybrali $L = 10\,\mu H$.

Hodnota výstupního kondenzátoru C_2 se vypočítá:

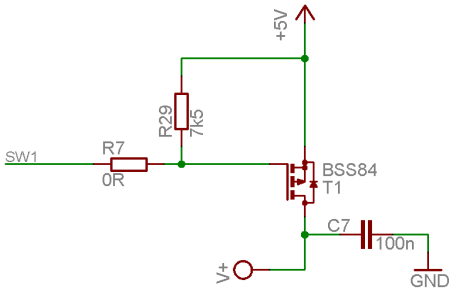
$$C_2 > \frac{I_{OUT} \cdot t_{ON}}{V_{OUT-RIPPLE} - I_{OUT} \cdot ESR_{C2}} > \frac{200\,m \cdot 0,75\,\mu}{45\,m - 200\,m \cdot 0,05} > 4,29\,\mu F$$

, kde ESR_{C2} je přibližný vnitřní sériový odpor kondenzátoru. V praxi se tento kondenzátor volí s co největší hodnotou pro co nejmenší zvlnění výstupního napětí, takže my jsme zvolili 100μF (největší hodnota tantalového kondenzátoru v SMD pouzdru).

Abychom mohli na vstup zdroje připojit nejen baterie, ale i nějaký externí adaptér, byly použity diody D_9 a D_{10}. Ty byly zvoleny tak, aby napětí za nimi nekleslo o více jak 0,3V. Dioda D_9 zabrání protékání proudu z připojeného adaptéru do baterií (pokud jsou vloženy) a analogicky dioda D_{10} zabrání protékání proudu z baterií směrem k adaptéru. Toto řešení bylo převzato od vedoucího mé práce.

Vývod `LBO` ze zdroje slouží k indikaci vybitých baterií a je vyveden k mikrořadiči. Mikrořadič podle toho může provést požadovanou operaci. Vývod je aktivní v „log. 0".

5V napájení měřící části a pro SD kartu je možné ovládat mikrořadičem (měřící část a SD kartu zvlášť). Jako spínač je použitý tranzistor DMOS s P-kanálem. Zapojení je na obrázku 4.12.



Obrázek 4.12: Obrázek zapojení tranzistoru pro spínání napájení

Vodič `SW1` je připojený k výstupu mikrořadiče. Pokud bude tento výstup ve stavu vysoké impedance (vodič `SW1` bude odpojený), uplatní se pull-up rezistor R_{29}, který na gate tranzistoru (vývod k R_7) přivede 5V. Rozdíl napětí na source (vývod k +5V) a gate bude tedy 0V a tranzistor bude rozepnutý. Stejná situace nastane, když výstup mikrořadiče nastavíme do „log. 1". Pokud tento výstup nastavíme do „log. 0", objeví se mezi gate a source rozdíl napětí 5V a tranzistor sepne (propojí source a drain). Zákmity při spínání a rozpojování tranzistoru filtruje kondenzátor C_7. Pokud bychom chtěli ochránit tranzistor proti zkratu na vývodu drain, museli bychom zvětšit odpor R_7 (nikoli libovolně).

4.3 Obslužný software mikrořadiče

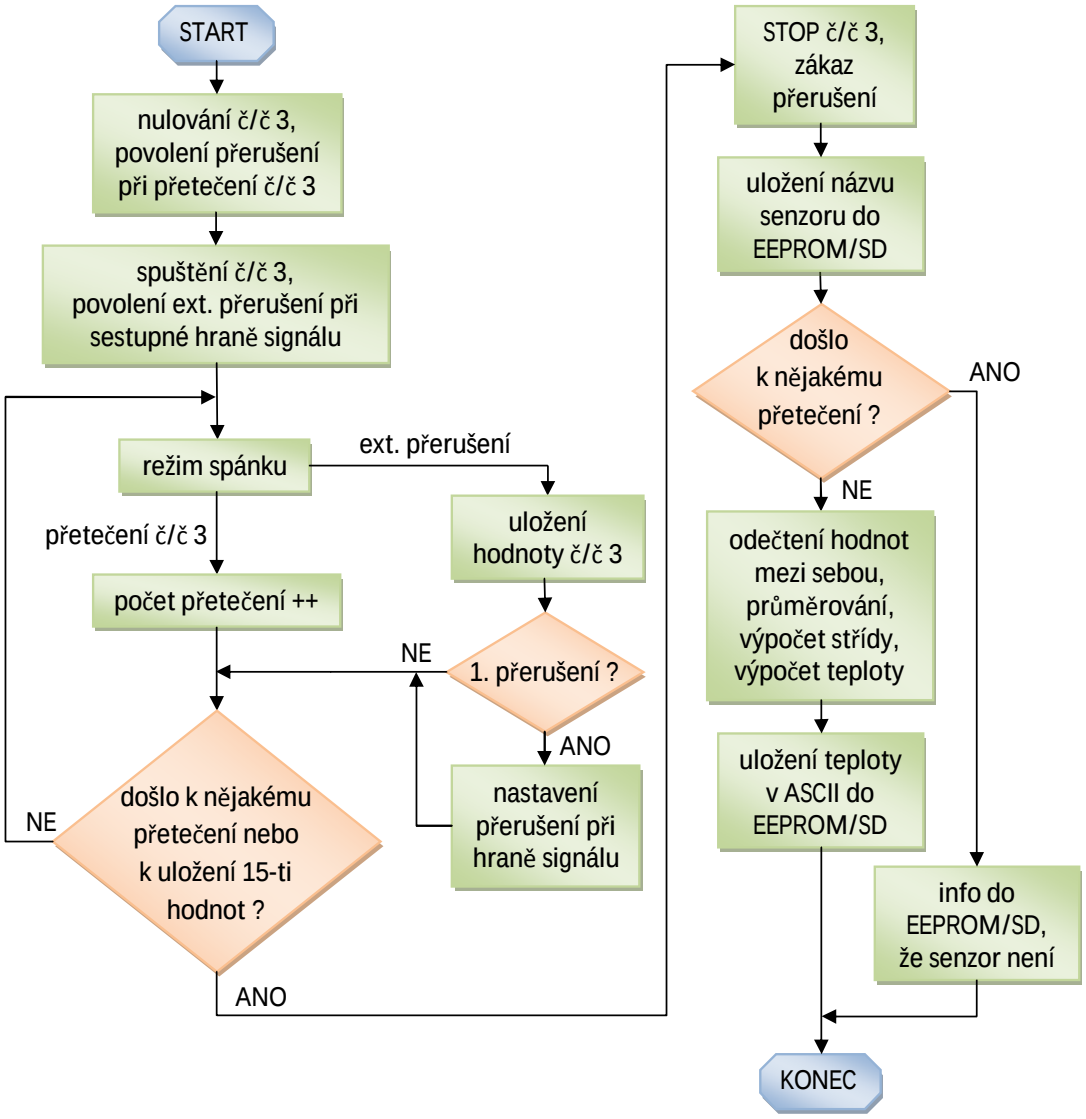
Firmware do mikrořadiče byl psán v jazyce C z důvodu přenositelnosti kódu i na jiné platformy mikrořadičů a z důvodu dobré přehlednosti kódu, kdy je snadné rozpoznat, co které příkazy provádějí. Jeho nevýhodou je pak výsledný přeložený kód, který většinou nebude tak rychlý, jako kdybychom firmware psali v assembleru (jazyk symbolických adres). V jazyce C se také mnohem hůře hledají případné chyby, šance že se vyznáme v přeloženém kódu (do assembleru) je téměř nulová.

Pro vývoj aplikace do mikrořadiče bylo použito vývojové prostředí AVR Studio s pluginem, který podporuje překladač gcc. Překladač bylo potřeba doinstalovat, já jsem vybral WinAVR. Vývoj a ladění aplikace potom probíhalo pouze v prostředí AVR Studia. Nahrávání přeložené aplikace do mikrořadiče probíhalo programem PonyProg. Ten podporuje pouze programátory připojené do paralelního nebo sériového portu PC. Žádný takový programátor jsem neměl k dispozici, proto byl podle schématu v [4] vyroben. Tento programátor se připojí na vyvedený konektor na řídící části zařízení. Problémy s programováním tímto způsobem jsou popsány v kapitole Testování. Další možný způsob naprogramování mikrořadiče je přes převodník USB, toho ale nebylo možné využít, protože v době vývoje nebyl k dispozici komunikační program s tímto převodníkem.

Při vývoji aplikace bylo potřeba zjišťovat jaké hodnoty ze senzorů mikrořadič naměřil nebo jestli hodiny reálného času generují přerušení apod. To by šlo zápisem dat mikrořadičem na SD kartu, pro kterou jsme však neměli napsaný komunikační program. Data se tedy prozatím zapisují na EEPROM paměť v mikrořadiči a z ní se čtou programem PonyProg v PC přes připojený programátor. Pro vývoj byla také na jednom z výstupů mikrořadiče připojena LED dioda (přes odpor).

4.3.1 Měření teploty

Pro změření teploty z jednoho senzoru slouží funkce `void TEMP_meas(BYTE sensor)`, kde parametr `sensor` určuje, který měřící modul (ze dvou, hodnoty 1 a 2) se má k měření vybrat. Jak je vidět, funkce nic nevrací, naměřené hodnoty se totiž zapisují do paměti přímo v této funkci. Pro pochopení, jak se provádí měření, je na obrázku 4.13 nakresleno blokové schéma jednotlivých operací.



Obrázek 4.13: Blokové schéma funkce pro měření teploty

Výhodou signálu s proměnnou střídou (tvar byl znázorněn na obrázku 3.6) je, že měříme pouze poměr impulzu k celkové periodě, takže nemusíme znát hodinový takt mikrořadiče. Pokud bychom potřebovali změřit čas nějakého úseku signálu v sekundách, takt bychom znát museli. Pro měření střídy stačí tedy měřit úsek signálu v libovolných jednotkách, které budou přímo úměrné času trvání takového úseku.

My jsme pro měření úseků signálu použili čítač/časovač 3 (16-ti bitový), který na začátku měření vynulujeme a spustíme v režimu bez předděličky (kvůli přesnosti). Pak povolíme externí

přerušení při sestupné hraně z jednoho ze vstupů `INT4` nebo `INT6` podle toho, z kterého modulu měříme, a mikrořadič uvedeme do režimu spánku. Při příchodu tohoto přerušení zapíšeme do paměti hodnotu č/č 3 a nastavíme generování přerušení jak při náběžné tak sestupné hraně. Tím jsme zajistili, že první hodnota v paměti udává začátek úseku signálu v „log. 0". Při dalších přerušeních už jen zapisujeme hodnoty č/č 3. Pokud by nedošlo k žádnému externímu přerušení, ale k přetečení č/č 3, znamenalo by to, že během 65536 taktů mikrořadiče (při 8MHz je to přibližně 8ms) se signál nezměnil (minimální frekvence signálu ze senzoru je 1kHz, tj. mění se 2x za 1ms) a senzor tedy není připojen. Povoleno je zapisovat do paměti maximálně 15 hodnot právě aby případně nedošlo k přetečení č/č.

Po naměření hodnot program vypočítává délky jednotlivých úseků signálu. Tzn. odečte 2. a 1. hodnotu v paměti, tím dostaneme délku úseku v „log. 0". Dále odečte 3. a 2. hodnotu, takže dostaneme délku úseku v „log. 1" apod. Vydělíme délku v „log. 1" délkou periody (délka v „log. 0" + délka v „log. 1") a máme střídu. Tento postup opakujeme dokud máme nějaké hodnoty s tím, že hodnoty vypočtených stříd sčítáme. Výsledek pak podělíme počtem vypočtených stříd a dostaneme průměrnou hodnotu střídy S. Tu aplikujeme na vzorec v [10]:

$$t = \frac{S - 0,32}{0,0047}\,[^{\circ}C]$$

a dostaneme teplotu, kterou zapíšeme do paměti ve znacích ASCII. Na to byla použita funkce:

```
char* dtostrf(double teplota, signed char pocet_znaku,
              unsigned char desetinnych_mist, char* buffer)
```

Více o ní se dozvíte v [14]. Funkce `sprintf` byla zavržena z důvodu nesmyslného chování.

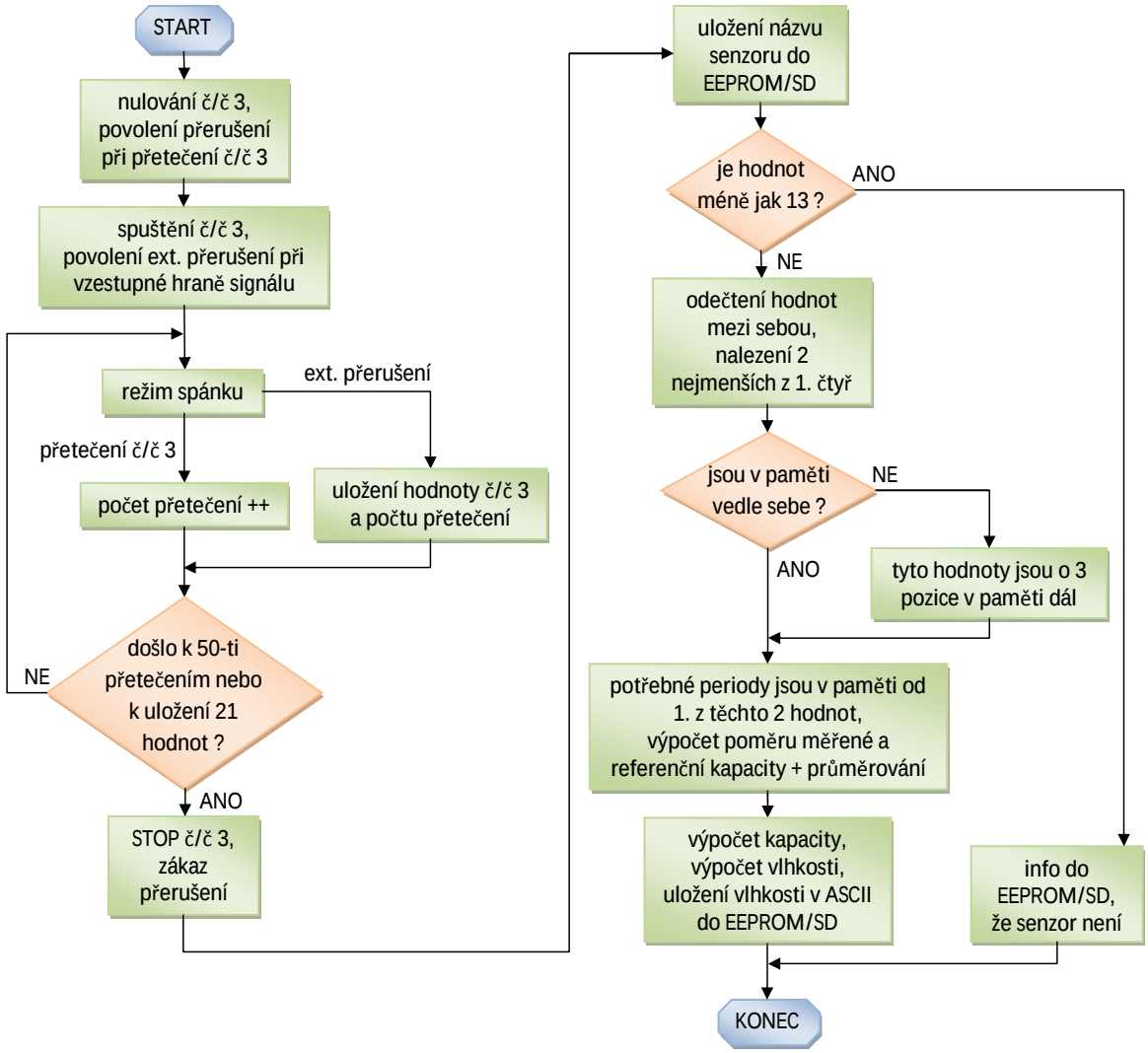
4.3.2 Měření vlhkosti

Měření vlhkosti obstarává funkce `void HUMIDITY_meas(BYTE sensor)`, parametr má stejnou funkci jako v předchozí podkapitole. Funkce opět nic nevrací, výsledky zapisuje do paměti. Princip funkce si nejdříve ukážeme na obrázku 4.14 blokového schématu a pak vysvětlíme funkci jednotlivých bloků.

Signál na vstupu (průběh byl znázorněn na obrázku 3.7) má podobnou výhodu jako u měření teploty. Délka jeho period vyjadřuje totiž poměr velikostí kapacit připojených na vstup měřícího obvodu `UTI`. Není tedy opět potřeba měřit nějakou časovou hodnotu, ale pouze libovolné hodnoty, které mezi sebou půjdou porovnat.

Pro měření je použit 16-ti bitový č/č 3, tedy stejný jako při měření teploty. Na začátku č/č vynulujeme, povolíme přerušení při přetečení a spustíme ho v režimu bez předděličky (kvůli přesnosti). Dále povolíme přerušení při náběžné hraně vstupního signálu, tím budeme měřit jednotlivé body mezi periodami. Při příchodu přetečení č/č 3 se pouze inkrementuje hodnota v paměti, která indikuje počet těchto přetečení od začátku měření. Při příchodu náběžné hrany na vstupu (`INT5` nebo `INT7`) program skočí do přerušovací rutiny, kde do paměti zapíše hodnotu č/č 3 a počet jeho přetečení. Počet přetečení se zapisuje z důvodu dlouhého cyklu vstupního signálu (až 100ms), takže než se změří celý cyklus, dojde přibližně k 12 přetečením č/č 3 (přibližně 8ms trvá jedno přetečení). Tímto řešením dosáhneme větší přesnosti než v případě použití předděličky č/č.

Po skončení jedné z rutin přerušení je testován v hlavním programu (v tomto případě v naší funkci) počet přetečení č/č 3 nebo počet uložených hodnot. Jedna uložená hodnota je v tomto případě považována jako hodnota č/č (16 bitů) a počet přetečení (8 bitů), protože při výpočtech



Obrázek 4.14: Blokové schéma funkce pro měření vlhkosti

budeme tyto hodnoty slučovat do jedné 32 bitové hodnoty pro snadnější výpočty. Pokud dosáhneme 21 hodnot (tj. úseky až pro 5 úplných cyklů signálu) nebo 50-ti přetečení, zastavíme č/č a zakážeme přerušení. Pokud jsme během 50-ti přetečení naměřili méně než 13 hodnot (méně než 3 cykly signálu, toto by ale nemělo nastat, správně by mělo být hodnot 0), program to interpretuje jako chybu a do paměti bude zapsána informace, že vlhkost není k dispozici.

Dvojice po sobě naměřených hodnot a jejich rozdíl vždy udává délku jedné z period signálu. Hodnoty těchto period si tedy uložíme někam do paměti, pro snadnější manipulaci s nimi. Jak je vidět z průběhu signálu na obrázku 3.7, cyklus je započat dvěma krátkými, přibližně stejnými periodami. Jejich součet je pak hodnotou přímo úměrnou parazitní kapacitě. Protože periody, které máme uložené v paměti, nemusí začínat touto dvojicí krátkých period, měli bychom je na začátku paměti najít. Cyklus je tvořen čtyřmi periodami, takže počátek cyklu se bude nacházet někde mezi prvními čtyřmi hodnotami.

Po nalezení počátku pak stačí počítat podle vzorce z datasheetu [11] a vypočítat tak poměr

měřeného a referenčního kondenzátoru (uveden je i výpočet poměru z prvního cyklu v paměti):

$$\frac{C_x}{C_{ref}} = \frac{T_x - T_{off}}{T_{ref} - T_{off}} = \frac{4.\,hodnota - (1. + 2.\,hodnota)}{3.\,hodnota - (1. + 2.\,hodnota)}$$

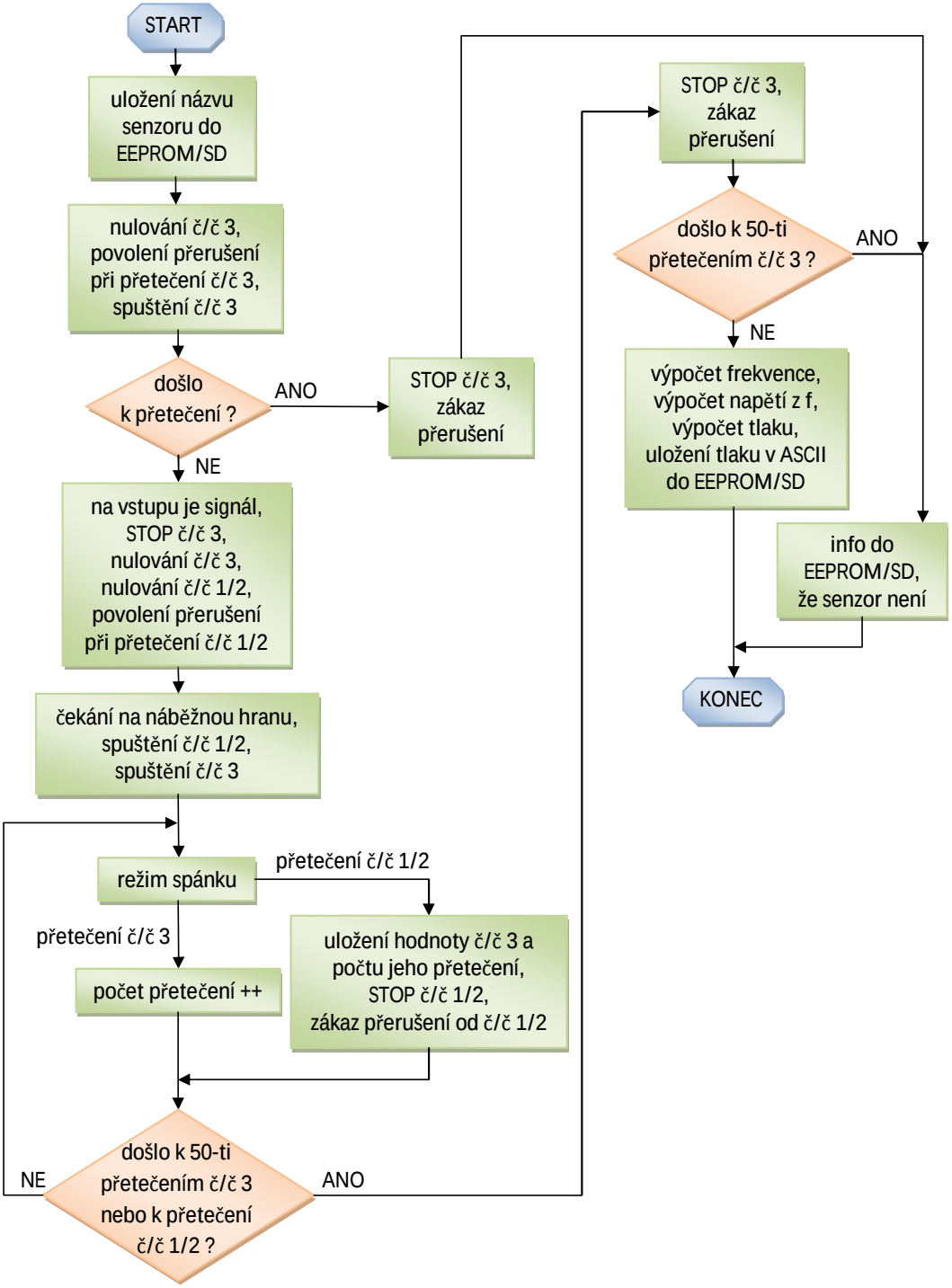
Tento výpočet se provede i pro další hodnoty v paměti a výsledné poměry se pak sečtou. Průměr tohoto poměru se pak vypočítá podělením sečtených poměrů a počtem těchto hodnot. Výsledný poměr se vynásobí referenční kapacitou a tím dostaneme hodnotu kapacity měřeného kondenzátoru, tedy senzoru vlhkosti. Podle této kapacity (C_C) se nakonec vypočítá výsledná vlhkost, dle vzorce z datasheetu [9]:

$$Vlhkost = \frac{C_C - C_S}{S} + 55\,[\%]$$

, kde C_S je kapacita senzoru při vlhkosti 55% a S je citlivost senzoru. Tyto hodnoty budou získány při kalibraci senzoru, viz kapitola Testování. Hodnota vlhkosti je potom zapsána do paměti ve znacích ASCII stejně, jako je tomu u měření teploty.

4.3.3 Měření tlaku

Frekvence signálu, která se zvyšuje nebo snižuje podle hodnoty tlaku, je měřena funkcí `void PRESS_meas(BYTE sensor)`. Výběr měřícího modulu, z kterého se má tlak měřit, se nastavuje parametrem `sensor` (hodnota 1 nebo 2). Na obrázku 4.15 je princip měření frekvence, která se nakonec převede na hodnotu tlaku.



Obrázek 4.15: Blokové schéma funkce pro měření tlaku

Nevýhodou měření frekvence je, že musíme znát takt mikrořadiče, abychom mohli změřit čas trvání periody. Pokud bude takt nepřesný (jako třeba v případě interního R/C oscilátoru), nebude tento čas také dostatečně přesný a o to více se to projeví na výsledné frekvenci, která je obrácenou hodnotou periody. My ale používáme u mikrořadiče externí krystalový oscilátor, takže přesnost bude doufejme dostačující. Navíc měříme period 256, takže výsledná hodnota bude dostatečně zprůměrována.

Na začátku měření musíme zjistit, zda-li je na vstupu (`T1` nebo `T2`) vůbec nějaký signál. Tyto vstupy nepodporují generování externího přerušení, takže zjišťování nějakých změn signálu se děje `while` cyklem. Ještě než ale začneme vstup testovat, vynulujeme č/č 3, povolíme přerušení při přetečení č/č a spustíme ho (bez předděličky). Tím zajistíme, že se program ve `while` cyklu nezacyklí, pokud by na vstupu nedošlo ke změně, protože při přetečení č/č se poruší podmínka cyklu. Pokud tedy došlo k přetečení (během 8ms se signál nezměnil), č/č je zastaven a do paměti zapsána informace, že tlak není k dispozici. Pokud ne, na vstupu je nějaký signál a můžeme pokračovat. Zastavíme č/č a vynulujeme. Čítač/časovač 1 je 16 bitový, takže abychom u něj zajistili stejné chování jako u 8 bitového č/č 2, nulujeme ho na hodnotu 0xFF00. Poté povolíme přerušení při přetečení č/č 1 nebo 2 (podle měřeného vstupu) a čekáme na náběžnou hranu signálu (už víme, že je přítomen), takže bude zajištěno, že měříme od začátku. Když se na vstupu objeví tato hrana, spustí se č/č 3 bez předděličky (opět kvůli přesnosti) a též č/č 1/2, který ovšem nečítá dle hodin mikrořadiče, ale dle hodin právě na vstupu `T1` nebo `T2` (vždy při náběžné hraně signálu). Rutina přetečení č/č 3 je pořád stejná jako u předchozích funkcí měření, tzn. při přerušení se zvýší číslo počtu těchto přetečení (tím jsme vlastně rozšířili č/č z 16 bitového na 24 bitový). Přerušení od č/č 1/2 udává, že signál na vstupu se již skládá z 256-ti svých period. Při tomto přerušení si zapíšeme hodnotu č/č 3 (včetně počtu přetečení), takže podle taktu mikrořadiče můžeme spočítat, kolik času tyto periody trvají, a zastavíme č/č 1/2 včetně zákazu jeho přerušení. Pokud by nedošlo k přetečení č/č 1/2 během 50-ti přetečení č/č 3, program to vyhodnotí jako chybný vstupní signál a do paměti tuto informaci zapíše. Na konci funkce už jen probíhá výpočet frekvence, poté výpočet napětí V_{OUT} z této frekvence a nakonec výpočet tlaku P podle vzorce z datasheetu tlakového senzoru [6]:

$$P = \frac{\frac{V_{OUT}}{V_S} + 0,095}{0,009} \cdot 10\,[hPa]$$

, kde V_S je napájecí napětí, tzn. 5V. Protože neznáme poměr mezi frekvencí a napětím (na vstupu převodníku) je potřeba tuto frekvenci při námi nastavovaném vstupním napětí naměřit, viz kapitola Testování. Výsledný tlak je zapsán do paměti stejným způsobem jako u předchozích funkcí.

4.3.4 Nastavování hodin reálného času

Pro nastavování hodin je použita sběrnice I^2C. Přes tuto sběrnici komunikuje mikrořadič (rozhraním TWI, I^2C kompatibilní) s hodinami. Tyto dvě zařízení jsou jedinými připojenými na sběrnici. V našem případě řídí veškerou komunikaci mikrořadič (master). Hodiny (slave) pouze vykonávají příkazy od mikrořadiče. Před započetím komunikace musíme nastavit rychlost taktu sběrnice. Pro náš případ postačí 100kHz (max. 400kHz) z důvodu větší spolehlivosti. Předděličku taktu necháme vypnutou (bity TWPS1 a TWPS0 jsou 0), a spočteme hodnotu pro registr TWBR (více v [1]):

$$TWBR = \frac{\frac{takt\_CPU}{takt\_sbernice} - 16}{2 \cdot 4^{TWPS01}} = \frac{\frac{8\,M}{100\,k} - 16}{2} = 32$$

Komunikace při zápisu dat do registrů hodin reálného času přes sběrnici I^2C vypadá takto:

- Nejdříve zavoláme funkci `void RTC_start_addr(BYTE reg_addr)`. Ta provádí příkazy:
  - Mikrořadič vyšle na sběrnici příkaz START.
  - Čekání, dokud nebude příkaz odeslán.
  - Kontrola, jestli byl odeslán správný příkaz. Jestli ne, zavoláme funkci `Error()`.
  - Vyslání adresy příjemce na sběrnici. Adresa má 8 bitů a dělí se na dvě části. Bity 7 až 1 udávají specifickou adresu příjemce (pro hodiny `RTC8564` rovna 0b1010001) a bit 0 udává, zda chceme ze zařízení s touto adresou číst („log. 1") nebo do něj zapisovat („log. 0"). My tedy vyšleme adresu 0b10100010.
  - Čekání na potvrzení od příjemce. Ten vyslanou adresu přijme, porovná jí se svou a pokud jsou stejné, vyšle potvrzení.
  - Kontrola, jestli byla správně odeslána adresa, a kontrola přijatého potvrzení. Musí se jednat o potvrzení od obvodu slave (hodiny). Pokud přijmeme jiný stav, voláme `Error()`.
  - Nyní se vysílají data. U našich hodin se za první blok dat (8 bitů) považuje adresa registru, do kterého chceme zapisovat. Tento registr si určíme parametrem `reg_addr`.
  - Po vyslání dat (adresy) zkontrolujeme, jestli nenastala při odesílání chyba, a jestli přišlo potvrzení od hodin. Pokud nastala chyba, voláme `Error()`.
- Potom voláme funkci `void RTC_write(BYTE data)`. Ta provádí tyto operace:
  - Odesílání dat hodinám. Ty si určujeme v parametru `data`.
  - Kontrola správnosti odeslání a kontrola potvrzení od hodin. Pokud je něco v nepořádku, tak voláme `Error()`.
  - Hodiny zvýší adresu registru o 1.
- Zde můžeme funkci `void RTC_write(BYTE data)` volat opakovaně. Adresa registru v hodinách, jak bylo řečeno v předchozím bodě, už je automaticky zvýšená o 1. Registrů je 16, takže bychom měli dávat pozor, aby nedošlo k volání této funkce po zápisu do posledního registru.
- Pokud už nechceme posílat žádná data, vyšleme na sběrnici příkaz STOP.
- Nějakou chvíli počkáme, pokud bychom totiž poslali hned za příkazem STOP příkaz START, bylo by to interpretováno jako žádost pro pokračování komunikace. Příkaz START by byl interpretován jako jiný příkaz - REP-START, ten ale nemáme na začátku v podmínce, takže by se zavolala funkce `Error()`.

Krátce popíšeme ještě komunikaci při čtení z registrů hodin (řídícím obvodem je opět mikrořadič - master), kdy si můžeme vybrat, zda chceme číst od nultého registru (neposíláme hodinám adresu registru, ta se při opakovaném čtení zvyšuje), nebo chceme číst od námi určené adresy. Popisovat budeme možnost výběru registru:

- Zavolá se funkce `void RTC_start_addr(BYTE reg_addr)` s parametrem adresy registru (v tomto případě se adresa dá poslat jenom v režimu zápisu).
- Pošleme příkaz START, ten se interpretuje jako opakovaný start (REP-START).
- Kontrola, zda-li se příkaz poslal správně, jinak `Error()`.
- Znovu vyslání adresy příjemce (hodiny), tentokrát s nultým bitem v „log. 1" (čtení), tzn. 0b10100011.

- Po odeslání kontrola, zda nedošlo k chybám a zda bylo přijato potvrzení od vysílače dat, jinak `Error()`.
- Vyšleme příkaz (pouze pro obvod TWI), že požadujeme od hodin data. Po přijetí dat se vyšle hodinám potvrzení a po jeho odeslání zkontrolujeme zda nedošlo k chybě. Přijatá data jsou uložená v registru obvodu TWI, takže si je někam uložíme (pro potřeby testování do EEPROM paměti). Tento bod opakujeme, dokud chceme číst po sobě jdoucí registry (jejich adresa se automaticky zvyšuje).
- Pro přečtení posledního registru vyšleme příkaz (opět jen pro TWI), že požadujeme od hodin poslední datový blok. Po jeho přijetí se vyšle negativní potvrzení hodinám, které je informuje, aby přestaly s vysíláním dat. Dále zkontrolujeme, zda vše proběhlo v pořádku a případně uložíme přečtenou hodnotu do paměti.
- Nakonec vyšleme příkaz STOP a stejně jako u zápisu vložíme časovou prodlevu.

Toto čtení bylo zabaleno do funkce `void RTC_read(BYTE beg_addr)`, která načte hodnoty všech registrů z hodin do paměti EEPROM na adresu `beg_addr` (pro testování, max. adresa je 240).

Pro inicializaci hodin, nebo mazání AF (Alarm Flag) bitu, kde AF je nastaven hodinami při příchodu alarmu a mazání musíme provést softwarově, byly vytvořeny tyto funkce:

- `void RTC_init()` - nastavení rychlosti a zápis do všech 16-ti registrů (od adresy 0 az 15) takto:
  - Nastavení taktu hodin sběrnice (registr TWBR a bity TWPS1 a TWPS0).
  - Volání funkce `void RTC_start_addr(BYTE reg_addr)` s parametrem 0 (nultý registr).
  - 16-krát volání funkce `void RTC_write(BYTE data)` s parametry, kterými nastavíme tyto údaje:
    - Povolení výstupu (`/INT`) při příchodu alarmu.
    - Nulování příznaku alarmu (AF).
    - Nulování všech časových hodnot.
    - Povolení alarmu při minutě 1 (tzn. v čase 00:01, 01:01, 02:01 apod., tedy každou hodinu).
    - Zákaz ostatních možností alarmu.
    - Zákaz taktu 32768Hz na výstupním pinu.
    - Zákaz vnitřního časovače.
  - Vyslání příkazu STOP.
  - Vložení prodlevy.
- `void RTC_clear_AF()` - slouží pro vynulování bitu AF:
  - Volání funkce `void RTC_start_addr(BYTE reg_addr)` s parametrem 1 (registr s bitem AF).
  - Volání funkce `void RTC_write(BYTE data)` s parametrem pro vynulování příznaku AF.
  - Vyslání příkazu STOP.
  - Vložení prodlevy.

Významy hodnot v jednotlivých registrech hodin najdete v [2]. Takto nastavené hodiny budou generovat na vstupu `INT2` mikrořadiče externí přerušení při sestupné hraně (negativní logika).

4.3.5 Zapisování do paměti EEPROM

Číst z paměti nepotřebujeme, takže zde uvedeme jen jednoduchou funkci pro zápis dat do EEPROM paměti (obsažená v mikrořadiči). Tato funkce má v programu název `void EEPROM_write(WORD Address, BYTE Data)`, kde parametr `Address` je adresa, do které chceme zapisovat (max. 4095, velikost paměti je 4kB) a parametr `Data` obsahuje osm bitů dat, které chceme zapsat. Sekvence příkazů pro zápis vypadá takto:

- Testování bitu EEWE v registru EECR. Když je nastaven, znamená to, že předchozí přístup do paměti nebyl ještě dokončen. Při nulové hodnotě můžeme začít se zápisem dat.
- Zápis parametru `Address` do 12-ti bitového registru adresy v paměti.
- Zápis parametru `Data` do osmi bitového datového registru EEPROM paměti.
- Nastavením bitu EEMWE v registru EECR povolíme zápis do paměti. Tento bit hardware automaticky nuluje po čtyřech taktech hodin mikrořadiče.
- Během těchto 4 taktů musí dojít k nastavení bitu EEWE (registr EECR), při jeho nastavení se odstartuje zápis do paměti dle adresního a datového registru. Samotný zápis pak probíhá nezávisle na programu. Pokud nedojde k nastavení tohoto bitu během 4 taktů (například příchod přerušení), k zápisu nedojde, ani se to nijak nedozvíme.

My ještě před zápisem do paměti zkontrolujeme adresu, zda-li nepřekračuje hodnotu 4095. Pokud ano změníme oddělovací znak mezi naměřenými hodnotami a adresu pro zápis vynulujeme. Díky změněnému znaku zjistíme, že jsme překročili kapacitu paměti, a že pokračujeme zápisem dat znovu od adresy 0. Kapacita paměti totiž stačí maximálně na 64 měřících cyklů (64 hodin, údaj o teplotě je zapisován ve velikosti max. dvakrát 10B, údaj o vlhkosti také a údaj o tlaku ve velikosti max. dvakrát 11B).

4.3.6 Zapisování do paměti SD

Zapisování do paměti SD bohužel nebylo implementováno z časových důvodů, i když jsem měl k dispozici hotový zdrojový kód. Ten je však ještě potřeba upravit pro potřeby této práce. Algoritmus komunikace s SD kartou je uveden v [13].

4.3.7 Hlavní program

Hlavní funkce `main` má velmi jednoduché chování, všechny složitější operace jsou již zabaleny do jiných funkcí, viz předchozí podkapitoly. Provádění této funkce je započato resetem mikrořadiče nebo připojením napájení. Inicializace mikrořadiče (vstupů/výstupů, podpůrných obvodů) a pak jeho běh probíhá takto:

- Piny spínačů napájení pro měřící část a SD kartu jsou nastaveny jako výstup do „log. 1" (vypnuté napájení).
- Nastavení pull-up rezistorů na pinech `INT2` (přerušení od hodin) a `INT3` (přerušení od zdroje při vybitých bateriích). Pro výstup hodin i zdroje je totiž použita negativní logika (aktivní v „log. 0") s tranzistory s otevřeným kolektorem.
- Povolení externího přerušení při sestupné hraně signálu `LBO` (od zdroje).

- Sepnutí spínače pro měřící část, tzn. výstup `PB6` se nastaví do „log. 0".
- Vložení prodlevy kvůli inicializaci všech senzorů (rozběh).
- Povolení globálního přerušení.
- Volání funkcí:
  - `void TEMP_meas(BYTE sensor)` s parametrem 1 (první měřící modul).
  - `void HUMIDITY_meas(BYTE sensor)` s parametrem 1.
  - `void PRESS_meas(BYTE sensor)` s parametrem 1.
- Funkce zavoláme znovu, ale s parametrem 2 (druhý měřící modul).
- Poté zavoláme `void RTC_init()` pro inicializaci hodin a povolíme od nich přerušení (`INT`) při sestupné hraně.
- Uspíme mikrořadič. Po příchodu nějakého přerušení (může přijít od signálu `LBO` nebo `INT`) se mikrořadič probudí a skočí na přerušovací rutinu tohoto přerušení. Rutina pro přerušení od zdroje napíše do paměti EEPROM informaci, že jsou vybité baterie a uspí mikrořadič (ten už poté nic vykonávat nebude). Rutina pro hodiny je prázdná a jenom způsobí probuzení mikrořadiče.
- Po příchodu přerušení od hodin (alarmu) voláme opět funkce:
  - `void TEMP_meas(BYTE sensor)` s parametrem 1 (první měřící modul).
  - `void HUMIDITY_meas(BYTE sensor)` s parametrem 1.
  - `void PRESS_meas(BYTE sensor)` s parametrem 1.
- Funkce zavoláme znovu, ale s parametrem 2 (druhý měřící modul).
- Vynulujeme AF bit v registru hodin a pokračujeme znovu bodem uspání mikrořadiče.

5 Testování

Po zapájení součástek jsem začal s testováním obvodu. Nejdříve jsem testoval napájecí zdroj, který byl nezatížený a odpojený od okolí. Na jeho výstupu bylo docela hodně zvlněné napětí, jeho hodnota byla až 0,5V špička-špička (na osciloskopu). Zkusil jsem tedy na výstup přiložit blokovací kondenzátor (pro svedení napětí o nějaké frekvenci na zem) nebo filtrovací kondenzátor s velkou kapacitou, jenže změna byla minimální. Po zatížení zdroje se však zvlněné napětí dostatečně zmenšilo, takže zdroj jsem už dále neřešil, i když i tak se zvlnění dále projevovalo, ale už ne v takové míře.

Testování pokračovalo napojením všech obvodů řídící části na zdroj přes ampérmetr, kvůli kontrole nechtěných zkratů apod., odebíraný proud byl minimální. Pak jsem proměřil jednotlivá napětí u obvodů, zde také nebyl žádný problém. Následovalo připojení modulu se senzory, opět přes ampérmetr, žádný problém se nevyskytl. Pak jsem zkusil zobrazit jednotlivé signály ze senzorů na osciloskopu. Zahřál jsem teplotní senzor a sledoval změny střídy, ty byly opravdu minimální, ale daly se pozorovat. U obvodu pro měření kapacity byly na osciloskopu krásně vidět tři různé periody (resp. 4, z toho dvě jsou krátké a shodné). Převodník napětí na frekvenci na tom byl hůře, signál nebyl totiž úplně stabilní. Nakonec jsem zjistil, že to je kvůli napájecímu zdroji, napětí na jeho výstupu je, jak už bylo řečeno, mírně zvlněné o nějaké frekvenci. Takto ne úplně dokonalé napájení ovlivňovalo výstupní signál převodníku, do budoucna bude tedy vhodné zdroj lépe vyladit. Teoreticky by nám ale tento problém vadit neměl, mikrořadič totiž reaguje na náběžnou hranu signálu a změří průměr frekvence z 256 period signálu, takže dále jsem se tímto nezabýval. Spolu s tímto problémem jsem narazil na další. Když jsem naměřil výstupní napětí senzoru tlaku, které je přivedeno na vstup převodníku, a přepočítal si napětí podle vzorce na tlak, vyšla mi podivně nízká, nereálná hodnota. To bylo způsobeno odběrem proudu převodníkem z výstupu senzoru. Pin `DIS` časovače (obrázek 4.4) totiž slouží k vybíjení kondenzátoru C_{27} a cyklicky je připojován na zem. Díky malému odporu R_{27} byl tedy výstup senzoru nadměrně zatěžován (odběr 1,5mA, max. odběr z výstupu senzoru je uveden 0,5mA). Řešení spočívalo v tom, že byly odpory R_{27} a R_{28} zaměněny za větší (3,3kΩ a 10kΩ, ve schématu v příloze zakresleny tyto hodnoty). Sice se zmenšila maximální frekvence, ale převodník již neovlivňuje výstupní hodnotu senzoru.

Posunul jsem se tedy do fáze komunikace s mikrořadičem. Na USB převodník jsem neměl komunikační program, přistoupil jsem tedy na stavbu jednoduchého programátoru, který se připojí na konektor (propojený s programovacími piny mikrořadiče) na řídící části. Přes program PonyProg jsem zkusil, zda-li přečtu Flash paměť mikrořadiče. Tu sice program načetl, ale při prohlížení obsahu byly po paměti náhodně rozházeny bajty s hodnotou jinou než 0xFF. Následovalo tedy mazání pamětí, ale problém byl pořád stejný, pokaždé s jiným výsledkem (rozházené bajty byly po každém čtení jinak). Po různém zkoušení zkrátit kabel programátoru, zapojit jiný programátor, byl výsledek stejný. Nakonec se ukázalo, že napětí na každém z programovacích pinů není úplně v pořádku (například místo 5V na vodiči `/RESET` byly jen 2V apod.). Důvodem byl převodník USB, který je na programovací piny také přímo připojen. Ten byl celou dobu odpojený od napájení (v případě potřeby je napájení ze sběrnice), takže i když je připojený na tyto piny, problém s ním jsem nepředpokládal. Došlo tedy k jeho odpájení z desky plošných spojů a programování přes vyvedený konektor začalo okamžitě fungovat.

Po sžití se s programovacím jazykem C pro mikrořadiče jsem dodatečně otestoval spínače napájení pro SD kartu a měřící část. Ty spínaly podle ovládání bez problémů, akorát při uvedení ovládacího pinu mikrořadiče ze stavu v „log. 0" (napájení připojeno) do odpojeného stavu (vysoké impedance, napájení odpojeno díky pull-up rezistoru) bylo zavírání tranzistoru poměrně dlouhé. To je dáno velikostí pull-up rezistoru, který jsem nechtěl snižovat z důvodu vyšší spotřeby. Řešení je softwarové, kdy před odpojením pinu nastavíme jeho stav do „log. 1" (napájení

odpojeno). U zapnutého napájení u SD karty bylo ještě potřeba zkontrolovat napětí za diodami (ty nám sníží napětí na potřebné). Při vložené kartě je toto napětí 3,6V, což je na hranici povolené specifikace (zkoušeno s bezcennou 16MB kartou). Při komunikaci ale karta odebírá mnohem více proudu než v klidu, takže napětí již klesne na požadovanou úroveň 3,3V (zkouška s umělou zátěží). Toto by šlo do budoucna vyřešit jinými diodami.

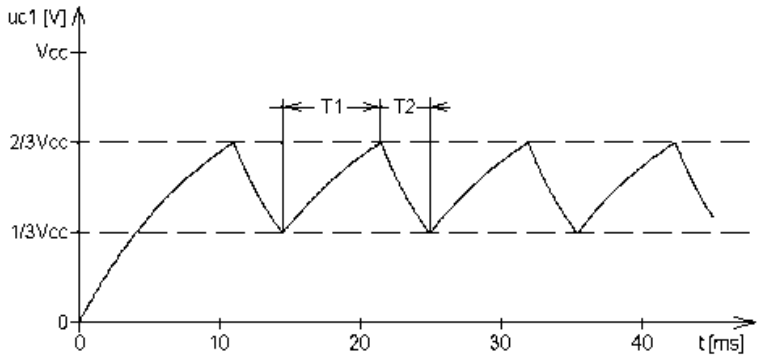
5.1 Kalibrace

Abychom měřili alespoň trochu pravdivé hodnoty ze senzorů, musíme zjistit, co vlastně znamenají hodnoty ze signálů, které jsme měřili. U obvodu pro měření kapacity [11], u teplotního senzoru [10] nebo u senzoru tlaku [6] je v datasheetu pro výpočet hodnoty ze signálu/napětí uveden vzorec, skutečná hodnota se od vypočtené liší jen s minimální chybou. U kapacitního senzoru vlhkosti [9] je pro přepočet sice také uveden vzorec, v něm ale musíme znát kapacitu senzoru při 55% relativní vlhkosti. Výrobce tedy zaručuje pro tyto senzory stejnou změnu kapacity při změně vlhkosti, ale celkové kapacity senzorů už můžou být odlišné. U převodníku z napětí na frekvenci žádný vzorec pro přepočet nemáme, jak se mění frekvence s napětím si musíme naměřit sami.

Než budeme kalibrovat senzor vlhkosti, musíme zjistit přesnější hodnotu referenční kapacity (ta už je na desce připájená), konkrétní součástka se totiž vyrábí s 5% přesností. K měření jsem použil tři stejné kondenzátory s přesností kapacity 0,1%. Jeden jsem vždy připojil místo senzoru vlhkosti (tedy k obvodu `UTI`) a mikrořadičem jsem změřil poměr mezi touto a již připojenou kapacitou. Z tohoto poměru jsem poté vypočítal, s použitím hodnoty přesné kapacity, výslednou kapacitu (tedy pro připájený referenční kondenzátor). Tyto tři výsledné hodnoty jsem pak zprůměroval. Výsledná referenční kapacita je tedy 101,54pF.

Nyní je tedy ještě potřeba zkalibrovat senzor (už můžeme celkem přesně měřit kapacitu). V [8] se můžeme dočíst, že pokud senzor vzduchotěsně uzavřeme v nádobě s vodou přesycenou solí (NaCl), kde budeme udržovat teplotu 20°C, naměříme kapacitu senzoru při 75,5% relativní vlhkosti. Podle tohoto návodu jsem tedy kalibroval, ale s rozdílem, že místo nádoby jsem použil igelitový pytlík. Z naměřené kapacity jsem poté podle grafu v [9] zjistil citlivost senzoru a kapacitu při vlhkosti 55%. Tyto hodnoty potřebujeme pro vzorec, z kterého pak můžeme určovat výslednou relativní vlhkost.

Dále potřebujeme zjistit vztah frekvence a napětí u převodníku. To jsem prováděl při odpojeném senzoru tlaku. Na vstup převodníku jsem tedy místo výstupu senzoru připojil regulovaný napájecí zdroj s napětím od 0 do 5V a zjišťoval, jaká je frekvence na výstupu převodníku při námi definovaném napětí. Frekvence byla měřena opět mikrořadičem. Při tomto měření jsem ovšem narazil na další problém, časovač měl na výstupu nulovou frekvenci při napětí od 0 do 3,33V (2/3 napětí z 5V). To ovšem není dáno špatnou funkcí obvodu, ale již od počátku špatným návrhem. Nepočítal jsem totiž s tím, že se kondenzátor C_{27} (obrázek 4.4) musí nabít na hodnotu minimálně 2/3 napájecího napětí časovače, aby byl obvod astabilní. Jak funguje nabíjení a vybíjení kondenzátoru je znázorněno na obrázku 5.1.



Obrázek 5.1: Průběh napětí na kondenzátoru při astabilním chování

Tato závažná chyba se ale neprojeví na funkci obvodu pro vstupní napětí od 3,33 do 5V. Můžeme tedy měřit výstup senzoru tlaku od 33,3 do 5V, kde 3,33V znamená tlak 845,55hPa. Takový tlak podle obrázku 3.1 odpovídá výšce 1500 metrů nad mořem, kam se běžně s tímto zařízením asi nedostaneme. Stále tedy zbývá naměřit hodnoty frekvencí při proměnném vstupním napětí (3,33 až 5V). Já jsem pro tento rozsah naměřil asi 20 hodnot a vynesl je do grafu programu Microsoft Excel. Bylo vidět, že se frekvence s napětím nemění lineárně. Nechal jsem tedy naměřené body proložit spojnicí trendu polynomického typu, která téměř přesně tyto body kopírovala, a vygenerovanou rovnici jsem použil pro výpočet napětí z frekvence v programu mikrořadiče.

6 Závěr

Výsledkem této práce je funkční měřící zařízení, které ukládá jednou za hodinu do paměti údaje o teplotě, vlhkosti a tlaku. Zařízení má díky tomu malou spotřebu. Napájení je možné z baterií. Měřící moduly jsou postaveny z obvodů, které jsou schopné měřit ve venkovních teplotách. Měření může probíhat až na dvou místech. Údaje z paměti se mohou číst přes sériové nebo paralelní rozhraní.

Skoro všechny body zadání byly splněny. Jediným nesplněným bodem je ukádání dat na přenosné médium. Ukládat data lze momentálně pouze do paměti EEPROM mikrořadiče. Tato paměť má malou kapacitu, takže zápis do ní probíhá cyklicky. Zpětně můžeme promítnout hodnoty až ze tří dnů, vždy po jedné hodině.

Pro tuto práci bylo navrženo schéma obvodu, ze schématu byly navrženy tři plošné spoje, z toho dva jsou shodné a určené pro měřící moduly. Jeden plošný spoj je určen pro řídící část. Dále byl napsán firmware pro mikrořadič v jazyce C, který jednou za hodinu zaznamenává data z měřících modulů. Kvůli tomu má za úkol obstarávat komunikaci s hodinami reálného času. Mikrořadič umí detekovat vybité baterie.

7 Literatura

[1] Atmel. datasheet - mikrořadič ATMEGA128L.

[2] Epson. datasheet - hodiny reálného času RTC8564.

[3] FTDI. datasheet - převodník USB na RS232 FT232BM.

[4] LancOS. AVR ISP (STK200/300) programátor. `http://www.lancos.com/prog.html`.

[5] R. Nesvadba. Zařízení pro ovládání objektivu U-D6REM a fluorescence BX-RFAl.

[6] F. Semiconductor. datasheet - senzor tlaku MPXH6115AC6U.

[7] O. Semiconductor. datasheet - spínaný zdroj NCP1421-D.

[8] Smartec. datasheet - kalibrace senzorů vlhkosti.

[9] Smartec. datasheet - kapacitní senzor vlhkosti SMTHS07.

[10] Smartec. datasheet - teplotní senzor SMT-160-30.

[11] Smartec. datasheet - univerzální obvod rozhraní UTI03.

[12] STMicroelectronics. datasheet - CMOS časovač TS555I.

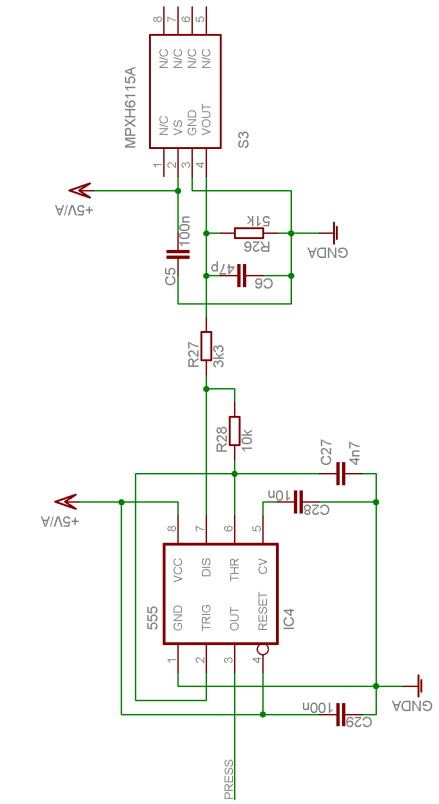
[13] B. L. Vávra. Vysoce spolehlivý systém založený na FPGA obvodech.

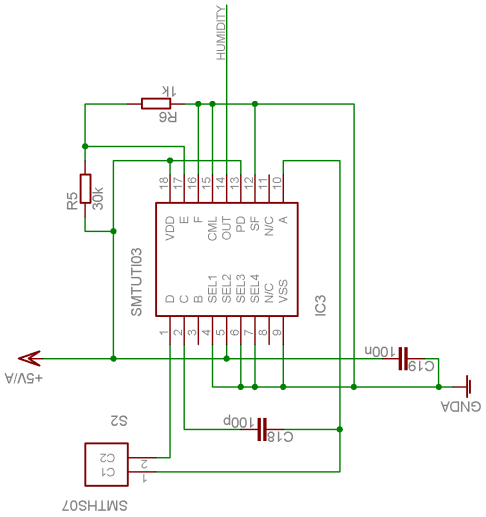
[14] WinAVR. avr-libc uživatelský manuál.

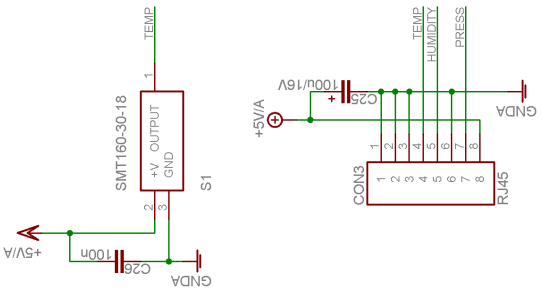
A Obsah přiloženého CD

- `\text\` - adresář se soubory *.tex a *.pdf textové části této práce
- `\firmware\` - adresář projektu pro AVR Studio se zdrojovým kódem firmwaru
- `\hardware\` - adresář projektu pro EAGLE se soubory schématu a plošného spoje
- `\datasheets\` - adresář s datasheety pro použité obvody

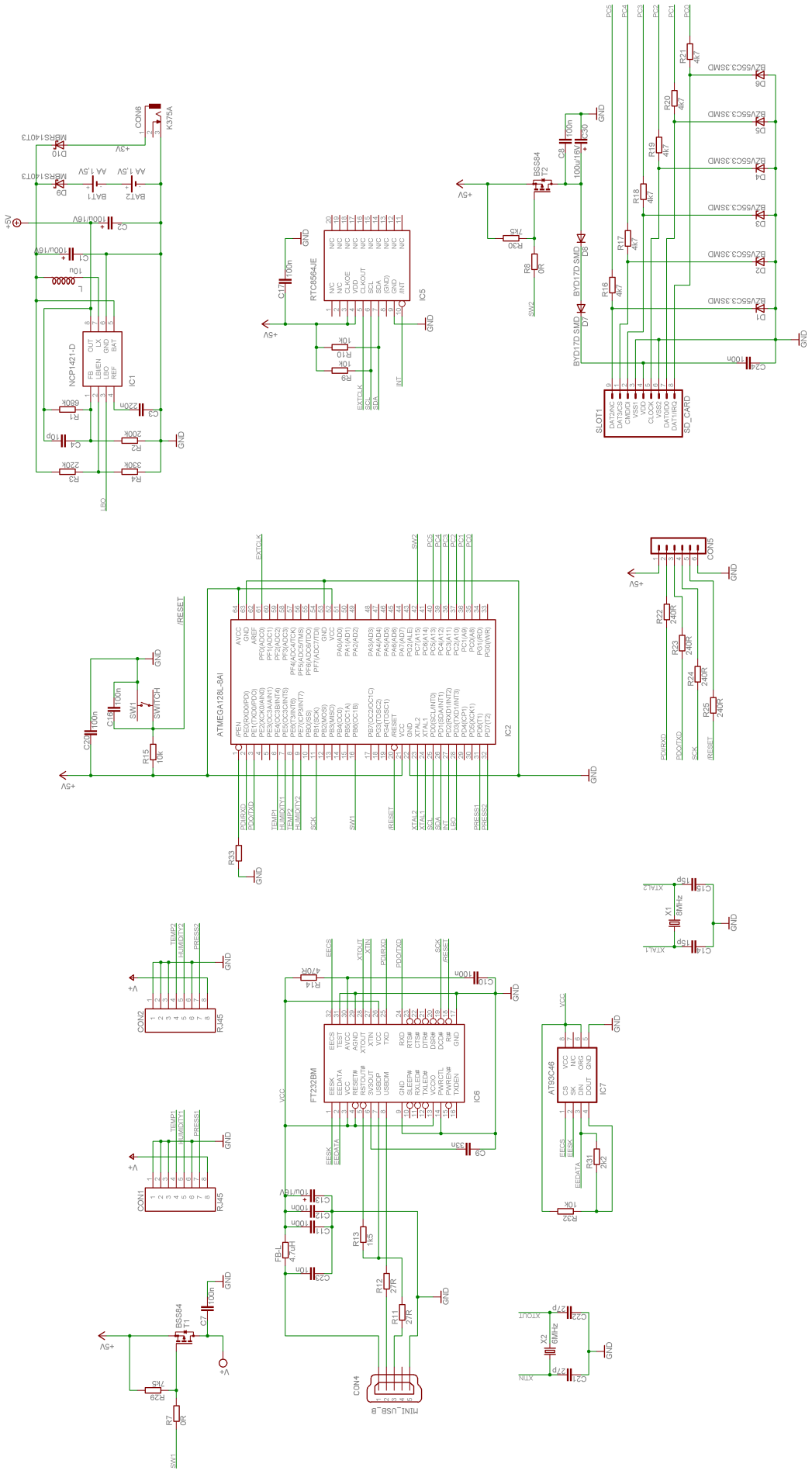
B Příloha - schéma zapojení měřícího modulu


MPXH6115A
S3
N/C
VS
GND
VOUT
+5V/A
100n
C5
51k
R26
GNDA
47p
C6
R27
3k3
R28
10k
C27
4n7
10n
C28
555
IC4
VCC
DIS
THR
CV
TRIG
OUT
RESET
100n
C29
PRESS


HUMIDITY
1k
R6
R5
30k
SMTUTI03
IC3
VDD
CWL
OUT
PD
SF
N/C
SEL1
SEL2
SEL3
SEL4
VSS
+5V/A
100n
C19
GNDA
S2
100p
C18
SMTHS07


TEMP
SMT160-30-18
+V
OUTPUT
GND
S1
+5V/A
100n
C26
GNDA
HUMIDITY
PRESS
100u/16V
C25
+5V/A
GNDA
CON3
RJ45

C Příloha - schéma zapojení řídící části


ATMEGA128L-8AI
IC2
FT232BM
IC6
RTC8564JE
IC5
NCP1421-D
IC1
AT93C46
IC7
SLOT1
SD_CARD
MINI_USB_B
CON1
CON2
CON3
CON4
CON5
CON6
SWITCH
+5V
GND
+3V
+1,5V