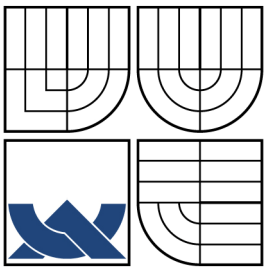
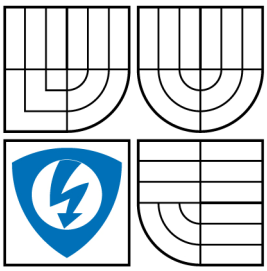


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF RADIO ELECTRONICS

NÁVRH A KONSTRUKCE ČASOVACÍHO ZAŘÍZENÍ

ALARM DEVICE

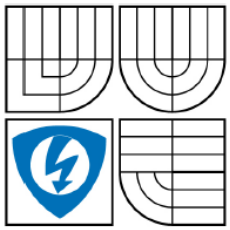
BAKALÁŘSKÁ PRÁCE

BACHELOR´S THESIS

AUTOR PRÁCE — FERENC DEMJANICS
AUTHOR

VEDOUCÍ PRÁCE — Ing. ZOLTÁN SZABÓ
SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav radioelektroniky

Bakalářská práce

bakalářský studijní obor
Elektronika a sdělovací technika

Student:	Ferenc Demjanics	***ID:***	97928
Ročník:	3	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Návrh a konstrukce časovacího zařízení

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s prací a programováním mikrokontrolérů od firmy Atmel. Navrhněte schéma zapojení časovacího zařízení, který bude napájeno ze sítě a zálohován pro případ výpadku sítě. V návrhu použijte obvod reálného času. Synchronizaci času řešte pomocí modulu pro příjem signálu DCF77. Navrhněte a realizujte časovací zařízení který bude zobrazovat datum, čas, umožňovat nastavění času a intervalu zvukové signalizace zvlášť pro každý den v týdnu.

DOPORUČENÁ LITERATURA:

[1] FRONC, V., KLUČIK, J. Mikrokontroléry ATMEL s jádrem 8051. Praha: BEN - technická literatura, 2002, 200 s. ISBN 80-7300-008-3

[2] BURKHAD, M., C pro mikrokontroléry. Praha: BEN - technická literatura, 2003, 280 s. ISBN 80-7300-077-6

[3] ANALOG DEVICES. Katalogové a aplikační listy integrovaných obvodů firmy. 2006.

Termín zadání:	9.2.2009	***Termín odevzdání:***	5.6.2009

Vedoucí práce: Ing. Zoltán Szabó

prof. Dr. Ing. Zbyněk Raida
Předseda oborové rady

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práve třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

ABSTRAKT

Časovačem se rozumí zařízení, které umožňuje přesně odměřovat čas. V každodenním životě se používá mnoho časovacích zařízení, ale jejich použití je omezeno na předem definovaný účel. Proto jsem se rozhodl zkonstruovat vlastní časovací zařízení, které umožní použití i pro individuální potřeby. Tím poskytuje širší spektrum využití v reálném životě.

Během bakalářské práce jsme pokračovali s realizací časovacího obvodu dle semestrálního projektu. Při návrhu jsme použili mikrokontrolér od firmy Atmel, přijímač a dekodér signálu DCF77. Dále obvod reálního času a DC/DC převodník od výrobců STmicroelectronics a Linear Technology. Jako zobrazovací jednotku jsme použili grafický LCD, připojený pomocí I2C, který je komerčně dostupný jako náhradní díl pro mobilní telefony.

Dle návrhu jsme sestavili a oživili obvod časovacího zařízení. Po úspěšném oživení jsme postupně sestavili program pro mikrokontrolér. Program definuje procesy pro chod zařízení. Je napsán v jazyce C, pomocí prostředí kompilátoru avr studio 4.

KLÍČOVÁ SLOVA

Časovač, Atmel, ATmega16, DCF77, DC/DC, I2C

ABSTRACT

Timer is an equipment that allows us to provide time measure. There are in all-day use a plenty of timers, but their use is ussually limited for one purpose only. Because of this I decided to compile a timer which allows to apply it for an individual needs. This granted a wider spectre of use in real life.

We used a microcontroller from Atmel, and also a DCF77 signal (atomic time signal)receiver and decoder. Further, we required a real-time circuit, DC/DC converter, and a powerswitch from manufacturers STmicroelectronics, and Linear Technology,respectively.For the imaging unit we used a graphic LCD connected via an I2C bus, which is commercially available spare part for mobile phones.

Following the design we compiled and revived a hardware part of the device. After a succesfull revive, we gradually compiled a program for microcontroller. The program defines a processes for run of the device. It was written in programming language C, with use of compiling enviroment avr studio 4

KEYWORDS

Timer, Atmel, ATmega16, DCF77, DC/DC, I2C

DEMJANICS, F. *Návrh a konstrukce časovacího zařízení.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 49 s. Vedoucí bakalářské práce Ing. Zoltán Szabó.

PROHLÁŠENÍ

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

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne ....

(podpis autora)

PODĚKOVÁNÍ

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

V Brně dne ....

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

1 ÚVOD

Cílem mé bakalářské práce je zkonstruovat a oživit časovací obvod, který byl navrhnut v semestrálním projektu. V této dokumentaci jsme proto použili i některé části semestrálního projektu pro lehčí orientaci a srozumitelnost.

V dnešní době jsou lehce dostupné jednoduché časovací obvody, třeba nějaké radiobudíky. Tyto obvody jsou však jednoúčelové a zpětně se nedá zasahovat do jejich funkce. Proto jsem se rozhodl zkonstruovat obvod, který bude možné kdykoliv přeladit dle potřeby. Poskytuje více možnosti využití, např. jako jednoduchý budík, nebo jako ovládání osvětlení v teráriu. Možné použití je i pro plánované zapnutí topení několik hodin předtím, než někdo dojde domů a existuje mnoho jiných možností využití.

My pro jednoduchost budeme používat jenom zvukové signalizace při dosažení předem nastavených podmínek (data a času). Aktuální čas bude synchronizován pomocí DCF signálu, který vysílá přesný čas z Frankfurtu. Bude zde i možnost nastavení času manuálně, pro případ, kdyby byl signál příliš slabý.

Začátkem dokumentace se seznámíme s prací a programováním mikrokontroléru ATmega16. Poté si vysvětlíme funkce obvodu reálného času. V pokračování uvedeme účel a strukturu kódování analogového signálu DCF77, kterou budeme používat pro synchronizaci aktuálního času. Ještě se zmíníme o zobrazovací jednotce a o ostatních obvodech použitých v části napájení celkového zařízení. Seznámíme se s programem, který řídí celý mikrokotrolér a s jeho nejdůležitějšími funkcemi a částmi, které jsou potřebné pro správnou funkci. Nakonec budeme mít v rukou kompletní časovací zařízení, které bude zobrazovat datum a čas, bude umožňovat nastavení času a intervalu zvukové signalizace zvlášť pro každý den v týdnu.

2 MIKROKONTROLÉRY ATMEL

2.1 Obecně

Pro řízení celého procesu jsem si zvolil obvod od firmy Atmel, která vyrábí obvody zaměřené na mikrokontroléry, paměti a radiový přenos.

Tato firma vyrábí mikrokontroléry AVR ve třech řadách:

základní	118 instrukcí
ATtiny	90 instrukcí
ATmega	130 instrukcí

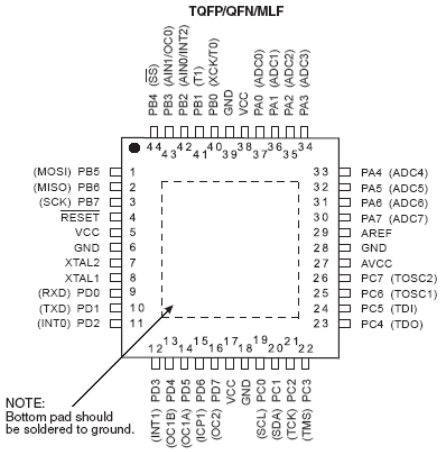
Tab. 2.1. Řady AVR.

Bližší informace v literatuře [1]nebo na internetovém stránce výrobce: [2]

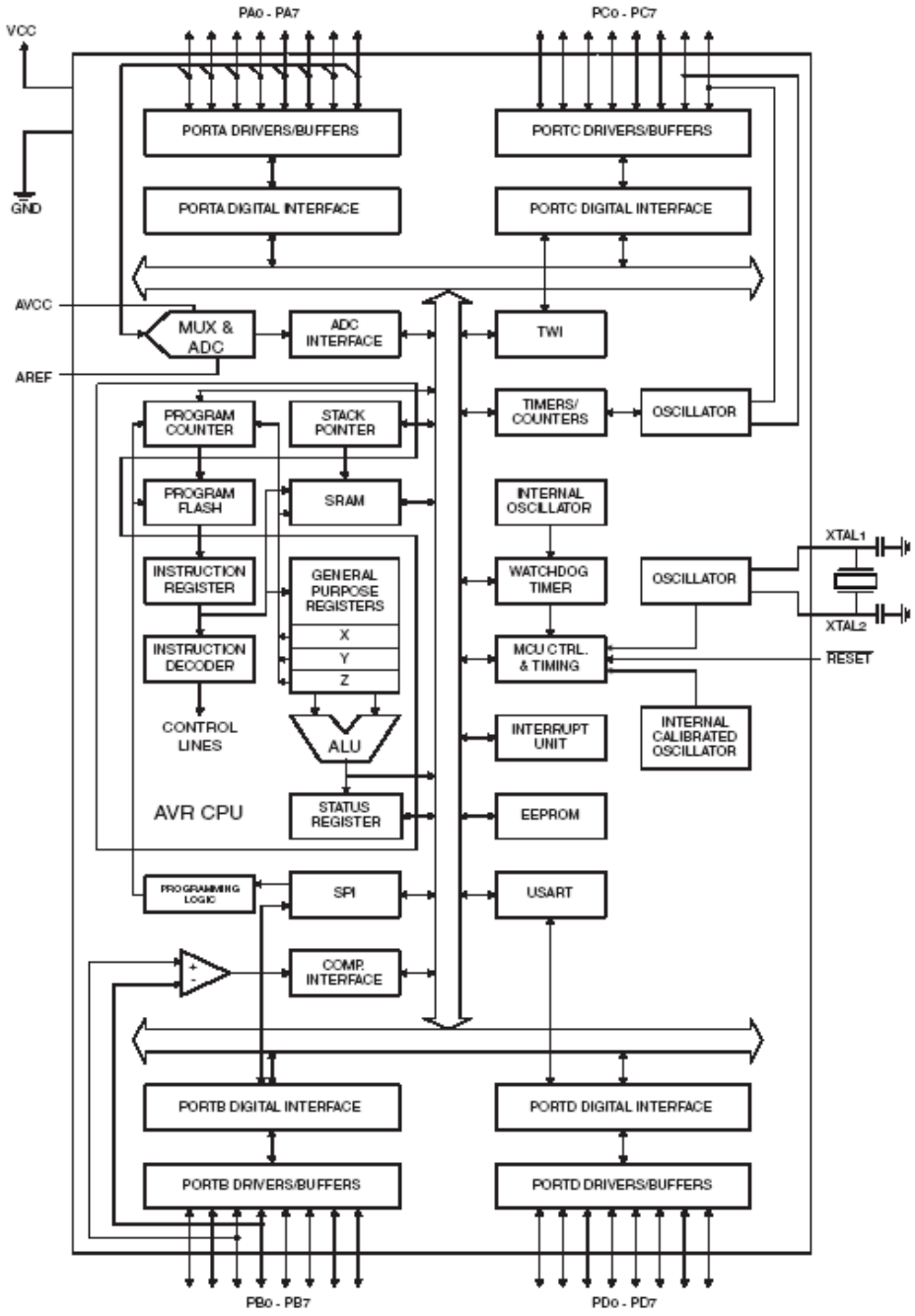
2.2 ATmega16

2.2.1 Rozložení vývodů

Konkrétně budeme používat mikrokontrolér z řady ATmega s typovým označením: ATmega16L-8AU. SMD provedení, kvůli nižším rozměrům, v pouzdru TQFP44, s rozložením vývodů dle obrázku *obr. 2.1.* Blokový diagram je znázorněn na obrázku *obr. 2.*2.



obr. 2.1. Rozložení pinů pro pouzdro TQFP44 [3].



obr. 2.2. Blokový diagram ATmega 16 [3].

Mikrokontrolér řady ATmega16 má čtyři porty označené písmeny A, B, C, D. Pro tyto porty lze programově definovat, jestli mají být vstupního nebo výstupního charakteru. A to pro každý pin jednotlivě, pomocí příslušného registru.

Tady lze vidět reálné rozložení vývodů. Téměř každý pin má dvě funkce. Základní , že je součástí některého z čtyř portů, a druhou (uvedeno v závorkách), že je součásti jiné periferie.

2.2.2 Periferie

K dispozici máme i periferní obvody:

watchdog	pro generování resetujících signálů při nečinnosti systému
TWI(I2C), SPI, USART	obvody pro sériový přenos dat
A/D	převodník: 8 kanálový, 10 bitový
ADC	analogový komparátor
JTAG	rozhraní pro ISP(in system programming)
timer/counter	2x 8bitové a jeden 16 bitový čítač/časovač s možností PWM
INT0 ÷ 2,	tři vstupy pro externí přerušení

Tab. 2.2. Periferie mikrokontroléru.

Z těchto budeme používat hlavně ISP rozhraní, pro programování bez toho, abychom museli mikrokontrolér vyjmout z obvodu a sběrnici TWI(I2C), pro komunikaci s vnějšími obvody.

2.2.3 Paměť a programování

AVR ATmega16 má paměť rozdělenou na tři části. Na programovou paměť (8k x 16 bitů), na datovou paměť SRAM (1k x 8 bitů) a na datovou paměť EEPROM (512 x 8 bitů).

V datové paměti SRAM se nacházejí datové registry. 32 obecných registrů a 64 I/O registrů. Pomocí těchto registrů se dají ovládat jednotlivé funkce obvodu. Základní je registr SREG (Stavový registr), který obsahuje jednak příznakové bity:

C - příznak přenosu, Z – příznak nuly, N – příznak záporné hodnoty, V – příznak přetečení, S - příznak znaménka, H – příznak polovičního přenosu;

a jednak řídící bity:

T – bit přenosu, I – povolení všech přerušení.

Stavový registr se používá nejvíc při běhu programu. Každý port má tři registry, pro nastavení charakteru, čtení a pro zápis údajů. Každý jeden prvek mikrokontroléru je řízen příslušným registrem. Při programování je třeba tyto registry naplnit potřebnými hodnotami. Tyto hodnoty můžeme zjistit v katalogovém listu [3], který je přidáván s každým typem výrobku.

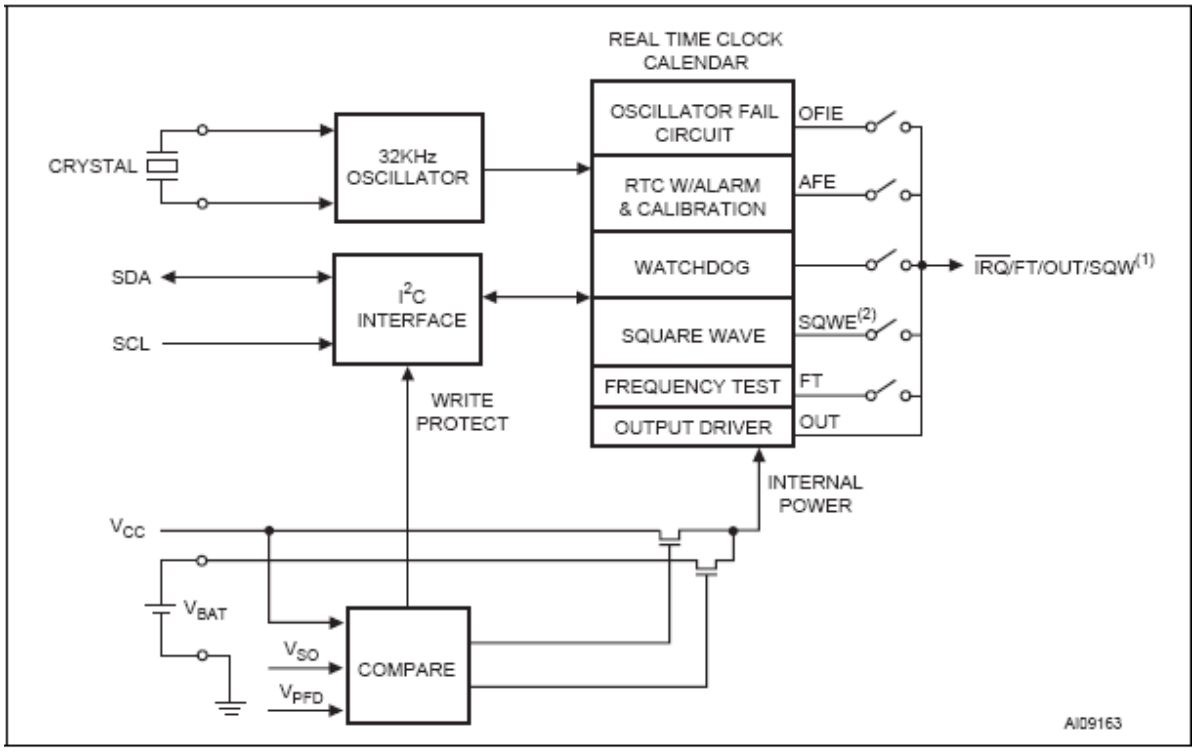
3 OBVOD REÁLNÍHO ČASU

3.1 Obecně

Obvody reálního času dokážou změřit přesný čas s relativně malou chybou. Pro funkčnost potřebují přesný oscilátor, obyčejně realizovaný pomocí krystalu.

3.2 Zvolený obvod

Pro naše zapojení jsem zvolil obvod od společnosti STMicroeletronics s typovým označením M41T81S, blokový diagram: *obr. 3.1*. SMD provedení v pouzdře SO8 bez integrovaného krystalu. Existuje provedení s integrovaným krystalem v pouzdru SOX18, ten však není dostupný na našem trhu. Avšak zapojení se dá lehce doplnit o jednu součástku. Pak je tedy potřebné přidat krystal s kmitočtem 32,768 kHz. Úplný rozbor funkce najdeme v literatuře [4]



Note: 1. Open Drain Output
2. Square Wave function has the highest priority on IRQ/FT/OUT/SQW output.

obr. 3.1. M41T81S blokový diagram [4].

Obvod má jeden čtyř-funkční vývod, který však používat nebudeme. Postačí nám pouze komunikace na rozhraní I2C.

Stejně jako výstup, nebudeme potřebovat ani některé funkce obvodu, např. watchdog, protože tyto záležitosti budeme řešit pomocí mikrokontroléru.

Budeme jen číst z obvodu aktuální čas a datum a porovnávat, jestli je splněna zadaná podmínka. Bude tady také možnost na zapsaní podmínky buzení do registru v odvodu. Pak stačí jen sledovat, jestli je aktivní příznakový bit alarmu.

Jsme schopní vyčíst údaj o čase s přesností až tisícin sekund.

3.3 Komunikace

Jak jsem se již zmínil, s obvodem komunikujeme přes datovou sběrnici I2C tak, že vyšleme na sběrnici příkaz pro čtení s příslušnou adresou 0xd0 a pak čekáme na odezvu. Odpověď na tento příkaz by měl být posloupnost dat s délkou 20 Bytů. Jednotlivé Byty znamenají v pořadí:

1	tisíciny/setiny sekund
2	sekundy
3	minuty
4	století/hodiny
5	den v týdnu
6	datum den
7	datum měsíc
8	datum rok
9	kalibrační registr
10	watchdog registr
11-15	registry pro alarm
16	registr příznaků
17-19	rezervován
20	registr pro výstup “Square Wave”

Tab. 3.1. Registry kalendářního obvodu.

3.4 Archivace

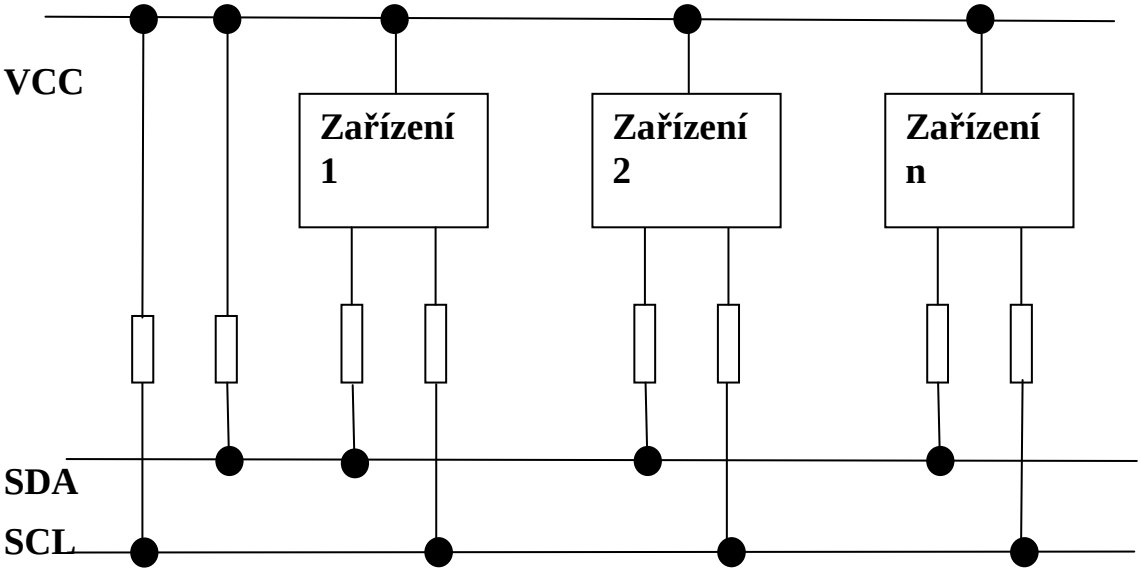
Obvod může být napájen ze dvou nezávislých zdrojů kvůli výpadkům sítě. Stačí připojit jednoduchou lithiovou baterii na přívod V_{bat}. Pří klesnutí napětí V_{ss} pod určitou hranici se obvod automaticky přepne do úsporného režimu a napájí se z baterky.

4 PROTOKOL I2C

4.1 Sběrnice

I2C sběrnice byla vyvinuta společností Philips Semiconductors [NXP]. Původně byla navržena jako rozhraní pro ovládaní baterií. Standard rozhraní definuje jak strukturu zapojení, tak i strukturu komunikačního protokolu.

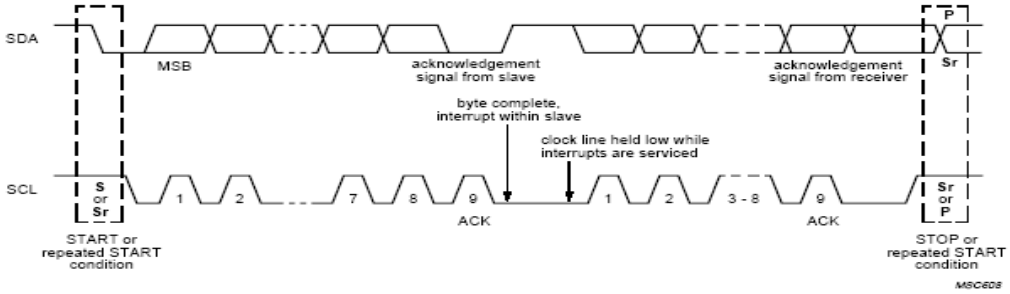
Sběrnice má dvousměrnou sériovou linku hodinového signálu (SCL) a sériovou datovou linku (SDA). Obě linky jsou nastaveny na vysokou úroveň pomocí pull-up rezistorů R_p. Ochranné rezistory R_s jsou volitelné. Žádné další linky nejsou specifikovány.



obr. 4.1. Princip sběrnice I2C.

Každé zařízení na sběrnici může fungovat jako přijímač nebo vysílač, Master nebo Slave, např.: *obr. 4.3* Současně může být připojeno na sběrnici více Master obvodů, ale pouze jeden může být ve stejném čase aktivní. Master obvody vysílají data a hodinový signál. Data jsou platná, když je linka SCL na vysoké úrovni. Slave obvody můžu data přijímat nebo odesílat pro obvody Master.

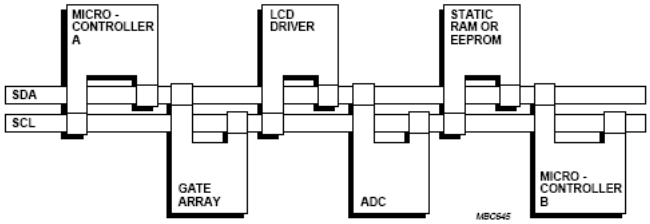
4.2 Přenos



obr. 4.2. Přenos dat na sběrnici.

Na začátku přenosu musí být vyslán znak pro start, což je snížení úrovně na nízkou na SDA, když na je na SCL úroveň vysoká. Jednotlivé bity se nastavují, když je hodinový signál na úrovni logické 0. Data se přečtou, když je hodinový signál na úrovni logické 1.

Komunikace začíná start signálem, pak se vysílá adresa potřebného obvodu, která je pevně daná výrobcem, kód příkazu a stop signál. Na tenhle příkaz odpovídá volaný obvod. Mezi každým vyslaným bajtem obvody čekají na potvrzení příjmu (ACK). Forma potvrzení je vyslání impulsu na linku hodinových signálů, pokud je na datové lince logická 0. Stop signál je, podobně jako start, změna úrovně na datové lince, pokud je SCL na vysoké úrovni.



obr. 4.3. Příklad připojení obvodů.

4.3 Rychlost

Kvůli tomu, že se jedná o dvoulinkovou sběrnici, jedná se pouze o poloviční duplex komunikaci.

Existuje několik stupňů přenosové rychlosti:

- standardní	100kbps
- režim Fast	400kbps
- režim High Speed	3,4Mbps
- režim Fast Plus	1MBps

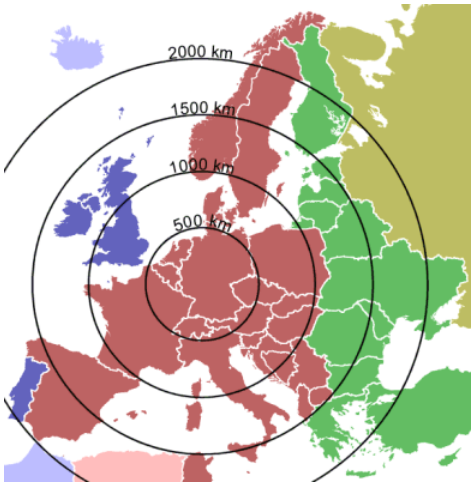
Tab. 4.1. Bitová rychlost sběrnice I2C.

Tyto informace jsem získal z literatury [5] a [6] , kde je tento standard podrobněji popsán.

5 DCF77

5.1 Princip

Existuje několik pozemních vysílačů, které vysílají radiový signál s účelem synchronizace času. Jeden z nich nese jméno DCF77 a nachází se v Německu. Vysílač je postaven vedle Frankfurtu v Mainfligenu. Posílá dlouhovlnný signál s kmitočtem 77,5 kHz. Použití nižšího kmitočtu je výhodnější kvůli většímu dosahu, odolnosti proti rušení a snadnějšímu přechodu signálu přeš jednotlivé překážky, např. stěny budov.



obr. 5.1. Dosah vysílače u Frankfurtu [8].

Příjem je možný skoro v celé Evropě. Největší dosah je 2100 km v noci a 1900 km přes den, při dobrém počasí. Reálný dosah přijmu je kolem 1500 km. Na obrázku výše jsou uvedeny vzdálenosti od vysílače a barevně časové zóny. V oblastech zabarvených modře platí čas UTC (koordinovaný světový čas). Ve střední Evropě platí čas UTC+1 (barva červená). A pak dále: zelené oblasti UTC+2 hodiny a žluté UTC+3 hodiny. V letním čase se připočte ještě jedna hodina. Pak je ve střední Evropě platný čas UTC+2.

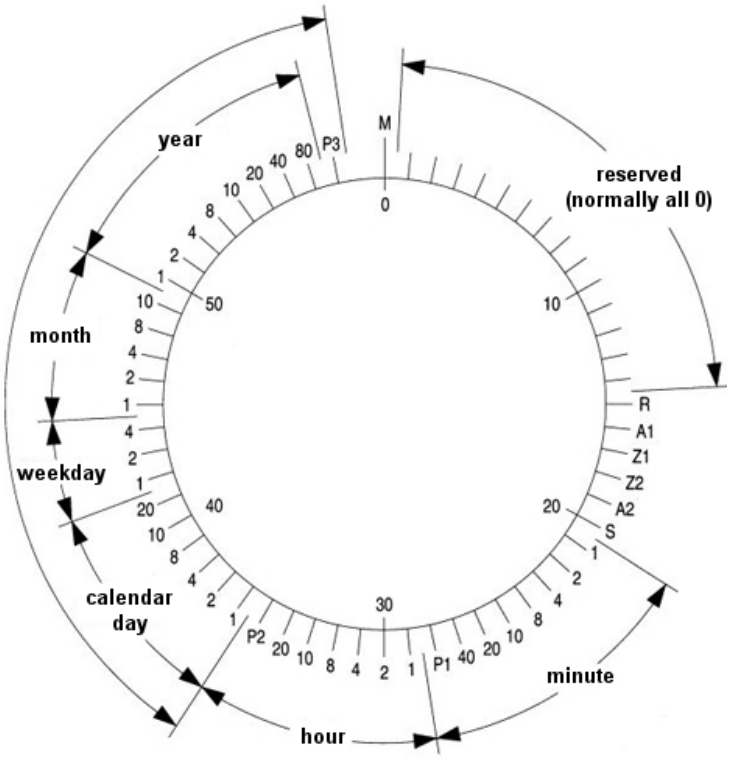
5.2 Kódování

Jeden kompletní přenos trvá jednu celou minutu. Používá se amplitudová modulace, jednou za sekundu se redukuje amplituda nosné o 6 dB. Když redukovaný signál trvá 0.1 s, znamená to logickou 0, logická 1 je znázorněna trváním signálu s redukovanou amplitudou o šířce 0.2 s. Impuls s redukovanou amplitudou v 59. vteřině je vynechán pro snadnější detekce začátku.

Jednotlivé bity jsou popsány na *obr. 5.2*. Prvních 14 bitů je pro rezervu, nebudeme je brát v úvahu. Následují bity:

R-	informace, jestli se používá náhradní anténa
A1-	oznámení o přechodu na letní/zimní čas (nastaveno hodinu před změnou)
Z1-	1 když se používá letní čas
Z2-	doplněk Z1 0 když se používá letní čas
A2-	oznámení o přestupné sekundě (nastaveno hodinu před změnou)
S-	start bit, vždy log. 1
P1÷3	paritní bity

Tab. 5.1. Speciální bity pro DCF.



obr. 5.2. Kódování DCF signálu [8].

Čas a datum je kódován ve formě BCD. V pořadí: minuty, hodiny, datum-den, den v týdnu, datum-měsíc a rok. Jak jsme si říkali, 59. bit se vynechá pro synchronizaci.

5.2.1 Přestupná sekunda

Protože je čas UTC založen na atomových hodinách, není shodnutí s časem podle rotace Země. Pro kompenzování tohoto rozdílu je zde možnost vložení nebo ubrání sekundy. Přestupné sekundy se zavádějí o půlnoci nejbližšího 30. června nebo 31. prosince.

O tom, zda je v příslušném termínu potřebné použití přestupné sekundy, rozhoduje *International Earth Rotation and Reference Systems Service* podle měření rotace Země. Maximální dovolený rozdíl mezi časem podle rotace Země a časem UTC je ±0,9 s

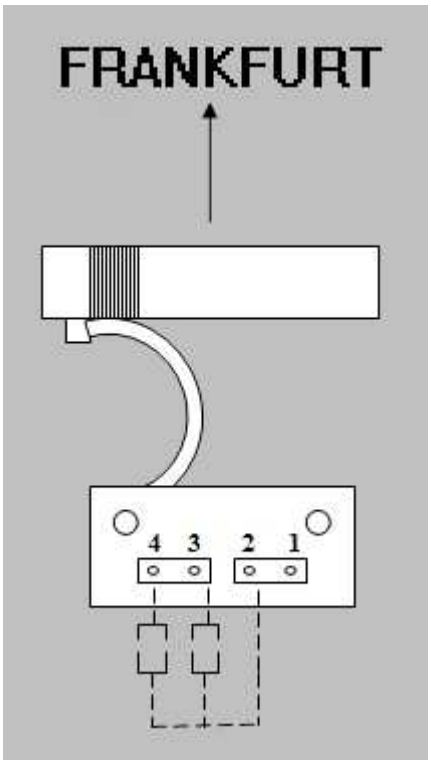
5.2.2 Letní čas

Jedná se posunutí času o jednu hodinu vpřed v letních měsících. Cílem je úspora elektrické energie používaná na večerní osvětlení, jelikož je po západu slunce většina lidí aktivnější.

V ČR (tehdy ještě ČSSR) byl zaveden v roce 1976. Dnešní forma uspořádání platí od roku 1996. Na letní čas se přechází poslední neděli v březnu. Zpět na světový čas, dle příslušné časové zóny, se přechází poslední neděli v říjnu

5.3 Přijímač

Pro přijímač a dekodér DCF77 signálu existuje mnoho zapojení, ale vzhledem k tomu, že je to vysokofrekvenční obvod, jsou náročné na konstrukci. Komerčně je dostupných několik takových modulů. Mnou zvolený přijímač vyrábí firma Conrad electronic.



obr. 5.3. Modul přijímače.

Modul se skládá z feritové antény a obvodu s dekodérem, který má čtyři svorky. První pro uzemnění, druhý je pro přívod napájecího napětí v rozmezí 1÷15 V. Svorky 3 a 4 jsou pro výstup digitalizovaného signálu, 3 neinvertovaný a 4 s invertovanou logikou.

Doporučené je použití pull-up rezistorů na datové výstupy.

Přijímač má být umístněn tak, aby nedošlo k rušení signálu, tj. ne v blízkosti elektrických strojů jako jsou motory, monitory atd.

6 OSTATNÍ OBVODY

Ke konstrukci budeme potřebovat ještě zobrazovací jednotku a několik obvodů pro řízení napájení. Jelikož bude náš časovací obvod napájen ze síťového adaptéru a nakolik z obyčejných tužkových baterií s možností nabíjení.

6.1 Zobrazovací jednotka

Pro zobrazování bude použit grafický LCD panel s rozlišením 101x33 bodů. Panel je monochromatický, bez zabudovaného podsvícení. Dostupný je jako náhradní díl pro mobilní telefony Ericsson řady T-2X, konkrétně modely T20 T28 T29 a model A1018.

Komunikace je realizována pomocí I2C protokolu. Je zde pět vývodů

Vlcd	Napájení logických částí 3 až 6,4 V;
Vlogic	Napájení LCD panelu 2,7 až 6 V;
SCL, SCA	pro komunikaci
GND	zem

Tab. 6.1. Vývody LCD panelu.

Používá se standard firmy Philips pro LCD panely menších rozměrů.

6.2 Ovládaní napájení

Napájení přístroje vyžaduje řídící obvod kvůli přepínání zdrojů. Tenhle řídící obvod bude pouze jednoúčelový, proto je snažší řízení ošetřit hardwarově něž programově.

Budeme potřebovat komparátor a přepínač mezi jednotlivými zdroji energie a DC/DC měnič.

6.2.1 DC/DC měnič

Napětí na bateriích je příliš malé pro napájení obvodu. Je potřeba zvýšit napětí pomocí DC/DC step-up měniče. Ten máme od výrobce Linear Technology, typ LTC1944.

Maximální výstupní proud je 175 mA a výstupní napětí je nastaveno na 5 V a 5,8 V. Vstupní napětí může být v rozmezí (1,2÷34) V

Pro touto činnost potřebuje obvod vnější kondenzátor indukčnosti a schottkyho diody. Další kapacitory jsou připojené kvůli filtraci rušení a zabezpečení řádné funkce obvodu.

6.3 Ostatní

Ke komunikaci s uživatelem a použitelnosti potřebujeme ještě několik součástek.

Nejdůležitější jsou tlačítka pro ovládání. Jsou čtyři a umožňují pohyb v menu a nastavení hodnot jako aktuální čas, datum a čas kdy má obvod signalizovat.

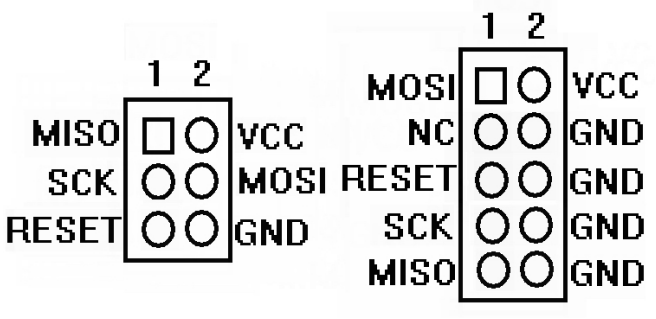
Pro zvukové signalizace budeme používat sirénu, která potřebuje jen stejnosměrné napájení. Střídavý zvukový signál je generován uvnitř sirény.

Pro použitelnost ve slabších podmínkách viditelnosti budeme používat LED podsvícení. To budeme ovládat pomocí mikrokontroléru. Tak máme možnost i pro světelné upozornění při splnění nastavených podmínek.

7 ZAPOJENÍ ZAŘÍZENÍ

7.1 Řízení přístroje

V zapojení jsme použili předem popsané součástky. Nejvýznamnější v zapojení obvodu je mikrokontrolér ATmega16, jenž má na starosti řízení celého zařízení. Příslušný program budeme psát v jazyce C nejspíše za pomoci AVR studio. Do programové paměti to nahrajeme pomocí programátoru, který je připojen přeš konektor SV_2. Zapojen je dle standardu pro ISP rozhraní. Používáme zapojení s šesti kontakty, dříve se používal konektor s 10 kontakty, více kontaktů zde bylo pro nulový potenciál GND a jeden kontakt NC (= not connected/nepřipojeno). Obě varianty jsou znázorněné na následujícím obrázku.



obr. 7.1. Zapojení ISP konektoru.

Na referenční vstupy mikrokontroléru jsou připojeny: filtr dolní propusti a zenerova dioda s referenčním napětím 3,6 V. Ty umožňují přesnější funkci A/D převodníků a analogového komparátoru.

7.2 Signalizace

Pro zvukovou signalizace je použitý reproduktor. Je připojena přeš tranzistor, protože může mít větší odběr, čímž by mohlo dojít k poškození části mikrokontroléru při přetížení.

Dioda LED může být využita pro mnoho účelů. Výsledná funkce se zvolí při psaní programu. Může třeba signalizovat chybu, nebo indikovat správnou synchronizaci se signálem přesného času. Hlavní využití je však jako pomocný prvek při oživování zařízení.

7.3 Komunikace

7.3.1 Komunikace v rámci obvodu

Na TWI (I2C) výstupy mikrokontroléru je připojena sběrnice. Ke sběrnici jsou připojeny i obvod reálného času a LCD. I když má mikrokontrolér vnitřní pull up rezistory, nemůžeme zaručit, že napětí na sběrnici bude dostatečné. Proto jsme přidali i vnější pull up odpory. Na otázku, zda jsou potřebné i vnější odpory můžeme zodpovědět po zkonstruování prototypu.

Přijímač DCF signálu je připojen na vstup, který se používá pro vnější přerušení INT0. To kvůli lehčímu detekování příjmu a dekódování přesného času.

Pro komunikaci s ostatními obvody a nebo pro připojení jiných periferijních obvodů slouží konektory SV_1 a SV_3 až SV_5. Porty *A* a *B* jsou přivedeny na konektory SV_3, SV_4. Jsou zde přidány i napájecí svorky. Jednotka USART je přivedena na konektor SV_5, k tomu je přidán pouze zemnící(GND) kontakt pro případné stínění kabelu. Na konektor SV_1 jsou přivedeny zbylé tři piny portu *C* bez jakýchkoliv svorek napájení.

7.3.2 Komunikace s uživatelem

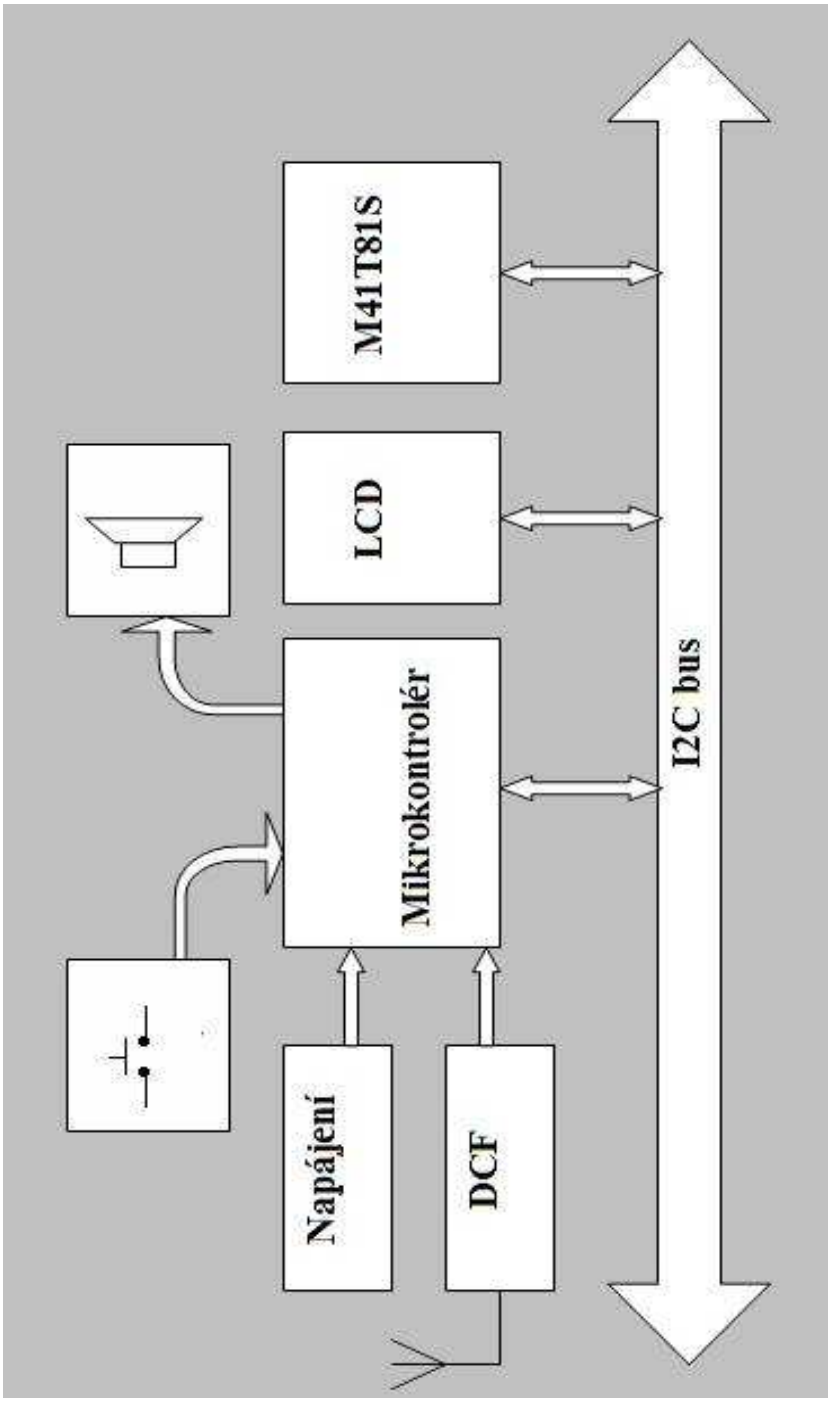
Komunikace s uživatelem bude uskutečňována pomocí LCD displeje a čtyř tlačítek. V pohotovostním režimu bude zobrazen aktuální čas a datum. Při stisknutí příslušného tlačítka bude zobrazeno menu. V něm se můžeme pohybovat,a také zadávat a zobrazovat jednotlivé údaje. Kvůli použitelnosti při horší viditelnosti jsou připojeny diody LED_3 a LED_4, které slouží pro podsvícení displeje.

7.4 Napájení

Pro výběr a připojení napájecího zdroje jsme použili obyčejný relé s přepínacím kontaktem. Běžně je jeho cívka bez napětí a na vstup DC měniče jsou připojeny baterie. Jestli je na vstup ze síťového prvku přivedeno napětí kolem hodnoty 5 V, napájení se přepne a čerpá se z toho vstupu.

Konektor J_2 slouží pro připojení stejnosměrného stabilizovaného napětí 5 V.

Obvod LT1944 je DC/DC konvertor s nastavitelným napětím na výstupu, tento typ má dvě vnější děličky nastaveny na 5 V a 5,8 V. Na vstup je přiveden výstup přes relé baterky s jmenovitým napětím 1,5 V nebo síťový prvek. Pro připojení baterií bude k dispozici obyčejná objímka. Články můžeme kdykoliv vyměnit. Baterie jsou přes odpor R_{10} připojeny na vstup mikrokontroléru s A/D převodníkem, aby se na nich mohlo měřit napětí. Jsou ale odpojitelné pomocí jumperu J_1 pro případ, kdybychom potřebovali někdy v budoucnosti celý port *A*.



obr. 7.2. Bloková schéma časovače.

8 REALIZACE

8.1 Změny

Při realizaci bylo potřebné změnit některé součástky obvodu oproti semestrálnímu projektu. Ta největší změna je přepínač s relé namísto obvodu ADM695. Změnu bylo nutné vykonat z důvodu, že ADM695 měl větší průchodný odpor při napájení z baterek. I při odběru proudu, který by měl obvod dle katalogu zvládnout, docházelo k poklesu napětí což pak nestačilo na napájení DC měniče. Při napájení ze vstupu V_{cc} nebyl žádny problém.

Další změnou je DC měnič. Napětí uvnitř obvodu bylo třeba rozšířit kvůli zobrazovací jednotce LCD. Ten pro správný kontrast potřebuje 5,8 V. Tato hodnota napětí je již příliš vysoká pro ostatní obvody.

Přidala se i ochrana proti záměně polarity napájeni. Je tvořena kombinací diody a pojistky, protože obyčejná sériová dioda by způsobila pokles vstupního napětí o napětí na PN přechodu. Nevýhoda je, že musíme zařízení rozebrat a vyměnit pojistku po každém špatním zapojení napájení.

8.2 Plošný spoj

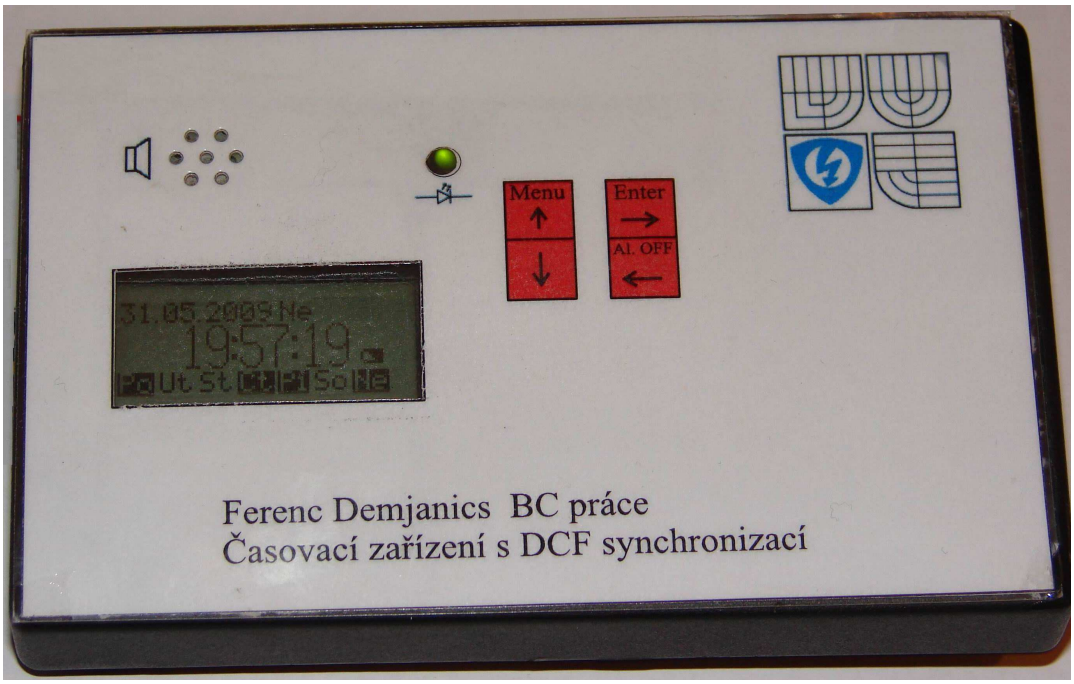
Kvůli úsporu místa jsem obvod rozdělil na dva kusy plošných spojů:

Jeden panel obsahuje pouze obvody pro komunikace s uživatelem, tj. zobrazovací jednotku tlačítka LED a reproduktor, která je integrovaná s LCD do jednoho pouzdra. Na druhém jsou umístněny všechny ostatní obvody: mikrokontrolér, kalendářní obvod, DC měnič a relé pro přepínání zdrojů.

Přijímač DCF signálu je tvořen samostatným plošným spojem, který je připojen pomocí plochého trojžilového kabelu.

8.3 Krabice

Všechny tyto věci jsou vloženy do univerzální krabice běžně dostupné v obchodě. Zvolili jsme si typ krabice, který obsahuje i otvor a místo pro objímku na baterie. Všechny tyto prvky jsou připevněné tavícím lepidlem, což je dobré kvůli výskytu nekovových prvků. Ty by mohly nepříznivě ovlivňovat příjem DCF signálu. Ten je i tak dost slabý a ve větších betonových budovách skoro nepoužitelný.



obr. 8.1. Časovací zařízení.

8.4 Osazení součástek

Při realizaci jsme použili jak SMD, tak i standardní součástky. SMD součástky jsme osadili na horní stranu desky. Standardní součástky, většinou konektory, jsme osadili na protější stranu. Je třeba dbát, aby baterka G_1 bylo připájena jako poslední, nejlépe po kompletním oživení přístroje.

Na desce s LCD zobrazovačem jsou všechny součástky i tlačítka typu SMD. Panel LCD je připevněn skrutky o průměru 2 mm. Byla zde potřeba udělat jeden přepoj. Ten jsme museli realizovat tak, aby vystupoval co nejmíň a nepřekážel tak zobrazovači. Konektor ke spojení dvou desek je umístněn na zadní straně desky.

Kvůli velké složitosti a hustotě součástek na desce jsou na osazovacím schématu vyznačeny pouze čísla součástek. Kdybychom vyznačili i hodnoty součástek, nebylo by schéma srozumitelné. Proto jsme potřebovali mít po ruce i seznam součástek.

8.5 Oživení

Nejprve je třeba alespoň vizuálně zkontrolovat desku plošného spoje, jestli nejsou nějaké chyby ve spojích. Jako první se zapojí součástky napájení. Musí se odzkoušet správná funkce této části, aby nedošlo ke zbytečnému poškození ostatních součástek. Po ověření správné funkci se mohou připájet ostatní součástky.

Pro správnou funkci mikrokontroléru je třeba vypnout funkce JTAG, protože používá stejné vývody jako interface I2C.Pak už můžeme nahrát program a vyzkoušet funkčnost přístroje. Jestli je vše v pořádku, můžeme připojit záložní baterii G_1, případně nastavit aktuální čas pomocí ISP, nebo vyčkat, než proběhne první synchronizace.

9 PROGRAM

Je to nejdůležitější část mojí práce. Program určí procesy, co má mikrokontrolér provádět. Je psán v jazyce C. A zapsán do mikrokontroléru pomocí ISP rozhraní. Tvořen je v podpůrném prostředí AVR studio, která obsahuje i simulátor a debugger.

Výpis nerealizujeme v hlavní funkci (main), ale pomocí přerušení, které nastane při přetečení čítače/časovače po cca 7/10 sekundách. Výhoda je, že když se neprovede celá operace, nemůžou zasahovat do běhu jiné faktory (přerušení DCF), nemohou tak vznikat konflikty, které se mmohou projevit formou špatného zobrazení údajů na LCD.

Pro menší objem dat v programové části paměti jsme deklarovali nové funkce. A pro přehlednost jsme je uložil do souboru *functions.h.* Následovně si objasníme činnost jednotlivých funkcí.

9.1 Funkce: main()

Samotná main funkce je dosti jednoduchá. Nastavují se zde parametry časovačů a potřebné povolení pro kalendářní obvod. Po vypsaní úvodní obrazovky se tady povolí globální přerušení. Pak končí v nekonečné smyčce while(1);.

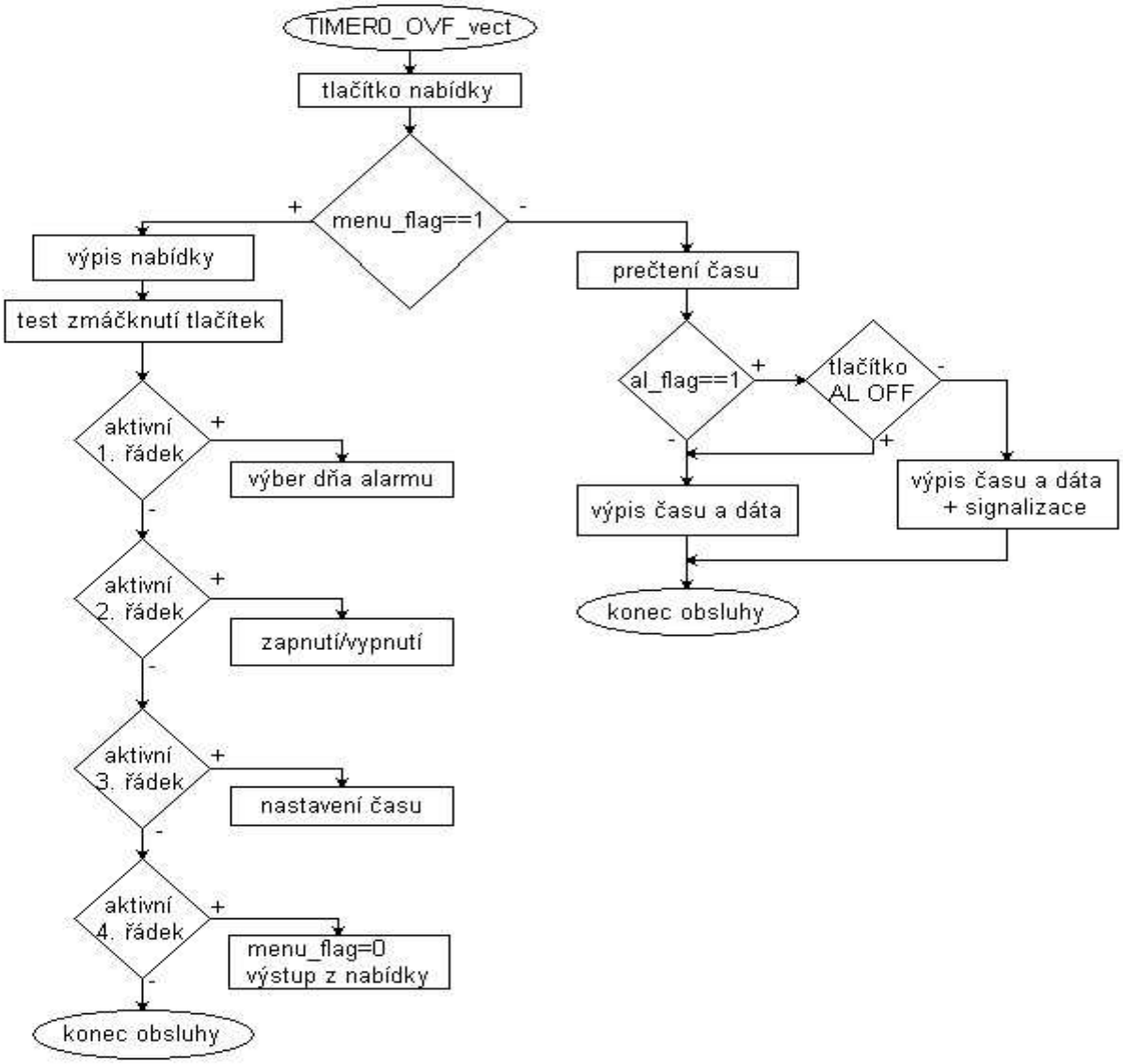
obr. 9.1. Vývojový diagram - main().

I když je tak jednoduchá, nemohl by se bez ní žádný program v jazyce C obejít. Je to něco, kde kompilátor začne svou činnost. Nekonečná smyčka ,čeká, až dojde k nějaké události, která způsobí přerušení. V našem případě budou dvě možnosti na přerušení. Jedna bude přetečení časovače, druhá bude příjem DCF signálu.

9.2 Obsluha přerušení časovače0

Na začátku obsloužení se testuje, jestli bylo zmáčknuté tlačítko "MENU" které garantuje vstup do nabídky, kde se pak mohou nastavovat jednotlivé budíky.

Tady se program větví.



obr. 9.2. Vývojový diagram - výpis času a nabídky.

9.2.1 Pohotovostní režim

Hned na začátku se volá funkce *alarm()*, která porovnává nastavené časy pro signalizace s aktuálním časem, jestliže ano, vrátí hodnotu *al_flag=1* a zapne signalizaci. Pak se hned testuje, jestli je zmáčknuté tlačítko AL OFF, které dočasně vypne zvukové signalizace způsobem, že nastaví pro al_flag hodnotu 2. Ten se vynuluje až v následujícím minutě. Tím se povolí znovu zapnutí signalizace.

Následovný krok je načtení času z kalendářního obvodu pomocí funkce *timeread()*. Ten pomocí základních funkcí přečte aktuální čas a uloží hodnoty do proměnné pole *timekeeper[]* s rozměrem 7 prvků. Z toho dekóduje čas a datum a uloží postupně do proměnných *sec1* až *year2*.

Voláním funkce *dateout()* se do prvního ze čtyř řádků vypíše zjištěné datum a dvě písmena symbolizující den v týdnu. Do druhého třetího řádku se vyšlou informace o čase s přesností jedné sekundy. Na to se používá funkce *timeout()*, která i případně indikuje zapnutý stav alarmu. Na konci druhého řádku zobrazí symbol a vykresluje znak baterky do třetího řádku. Znak baterky se nevykresluje když je zařízení napájeno ze síťového prvku.

V posledním, čtvrtém řádku, jsou uvedeny jednotlivé dny v týdnu. Základní zobrazení je ve standardních barvách, písmena černá a pozadí průsvitné. Jestliže je pro daný den zapnutý alarm, jsou zkratky toho dnu zobrazeny v invertovaných barvách, tj. pozadí načerno a písmena průsvitně

obr. 9.3. Obrazec – pohotovostní režim.

9.2.2 Režim nabídky

Jestli bylo zmáčknuté tlačítko menu, program vstoupí do režimu nabídky. Zde je možné upravovat čas a zapnutí alarmu zvlášť pro jednotlivé dny.

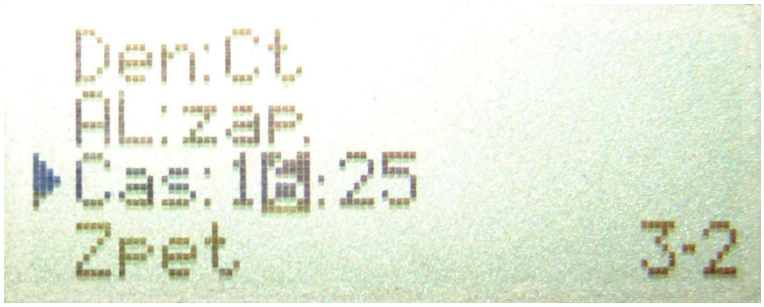
Nabídka se skládá ze čtyř řádků. První řádek je pro výběr dne, pro který chceme budík nastavit. Tlačítkem enter můžeme vstoupit do módu editace řádku. Barva písmenek zkratky dne se invertuje, tím se indikuje, že je řádek aktivní. Pak tlačítky ↑↓můžeme vybrat požadovaný den. V tomto řádku již není jiná možnost editace, takže po výběru se jen můžeme vrátit do módu pro výběr řádků tlačítkem: ←.

Mezi jednotlivými řádky je výběr zabezpečen opět tlačítky ↑↓. Pro indikaci vybraného řádku je použit znak > na levém kraji obrazu

Druhý řádek nám umožňuje povolit nebo zakázat zapnutí budíku pro vybraný den. Editace je stejná jako u prvého řádku

Třetí řádek je určen pro nastavení času s přesností jedné minuty. Nastavení se provede pro každou číslici zvlášť. Mezi čtyřmi aktivními kolonkami se můžeme pohybovat tlačítkami: →← a měnit jejich hodnotu pomocí ↑↓

Jestli se dostaneme ke čtvrtému řádku s nápisem ZPĚT, program čeká na potvrzení tlačítkem enter a pak přestoupí zpět do standardního režimu.



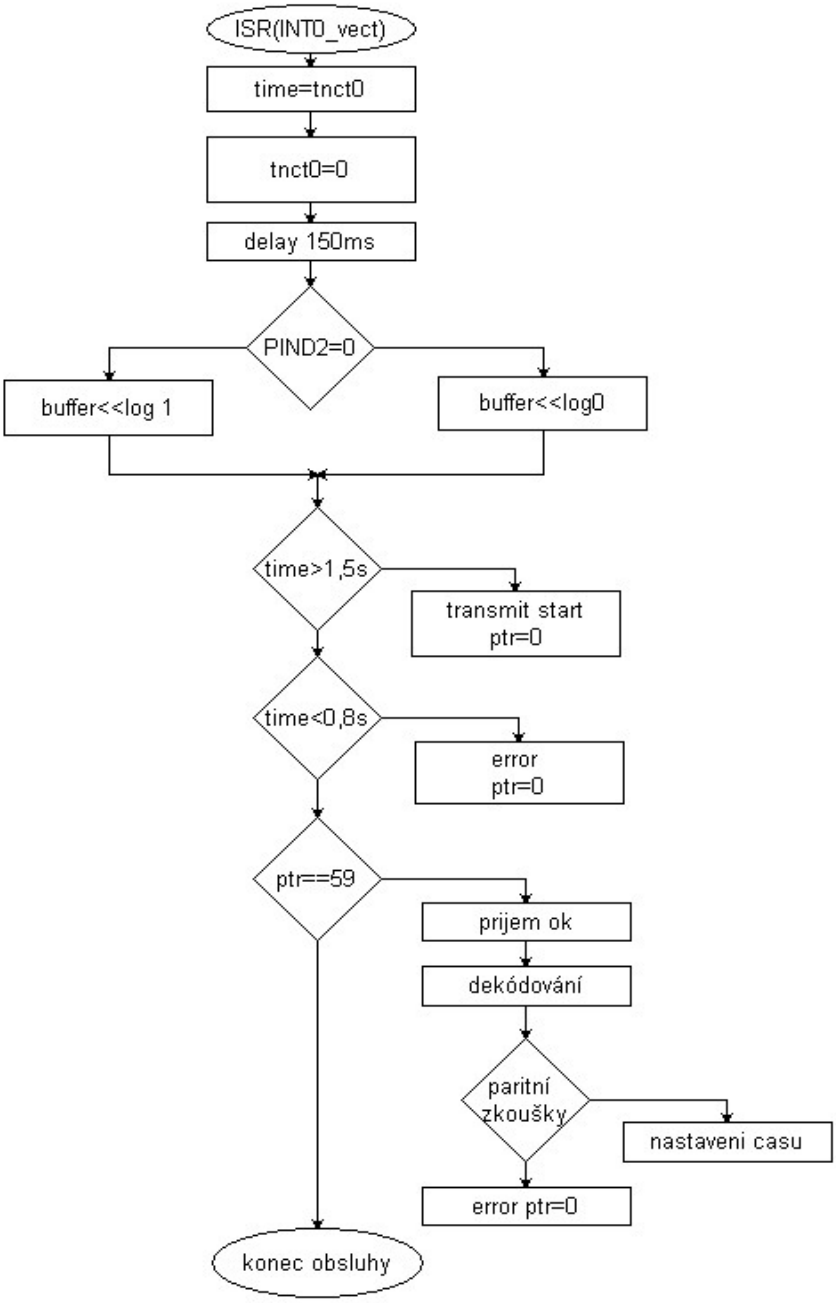
obr. 9.4. Obrazec – nabídka.

9.2.3 Signalizace

Na začátku přerušení se testuje, jestli se shoduje aktuální čas a den s nastaveným časem pro budík. Pro tento účel je určena funkce: *alarm()*. Jestli nastane rovnost parametrů a pro daný den je povoleno zapnutí budíku, aktivuje se signalizace. Ta je prováděna sirénou s kolísavým tónem a znakem‘Zz‘ na LCD v pohotovostním režimu.

Signalizaci je možné dočasně přerušit tlačítkem s nápisem ALL OFF.

9.3 Obsluha příjmu DCF77



obr. 9.5. Vývojový diagram – příjem DCF signálu.

Přijímač DCF signálu je připojen na vstup externí přerušení. Nastavena je na detekce sestupné hrany signálu. Hned po vstupu do obsluhy program čeká 80 ms. V případě, že je vstup na nízké úrovni, předpokládáme, že jde o správný signál a o nějaké rušení. Znovu se čeká 70 ms. Tak jsme se dostali do oblasti hranice mezi logické 1 a 0. Jestli je vstup stále na nízkém úrovni jedná se o logickou 0.

Jednotlivé bity synchronizace se zapisují do bufferu a v případě úplného příjmu se dekóduje během toho jisté přerušení.

Pro zabezpečení příjmu testujeme čas mezi dvěma znaky. Jestli je menší něž 0,8 s, jedná se o špatný znak, protože signál se posílá s periodou 1 s. Jestlipak je větší něž 1,5 s jedná se o synchronizační znak. V obou případech se nuluje ukazatel bufferu pro příjem. Druhé zabezpečení jsou paritní bity P_1 až P_3.

9.4 Funkce pro základní ovládání I2C:

9.4.1 Funkce: start()

```
void start(void)
        {

                TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);        //start
                while (!(TWCR & (1<<TWINT)));                  /       /test

        }
```

Tato funkce pošle znak start na sběrnice. Nejprve musíme nastavit příslušné bity registru TWCR:

TWEN - TWI Enable Bit zapnutí komunikace na i2c sběrnici

TWINT - TWI Interrupt Flag povolí přístup na sbernici

TWSTA - TWI START Condition Bit posílání znaku start

Poté program čeká než mikrokontrolér nevynuluje bit TWINT. To znamená že start kondice bylo vysláno v pořádku.

Používá se na zahájení přenosu.

9.4.2 Funkce: stop()

```
void stop(void)
        {

                TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO); //stop

        }
```

Tato funkce pošle znak stop na sběrnici. Musíme nastavit dva obdobné bity řídícího registru jako při startu a bit TWSTO (TWI STOP Condition Bit). Tady se testování nemá provádět.

Používá se na ukončení přenosu.

9.4.3 Funkce: addressw() a data()

```
void addressw(unsigned char a)
        {
        TWDR = a;                                       //slave addres
                TWCR = (1<<TWINT) | (1<<TWEN);
                while (!(TWCR & (1<<TWINT)));           //test
        }
//-------------------------------------------------
void data(unsigned char b)
        {
                TWDR = b;                                       //data
                TWCR = (1<<TWINT) | (1<<TWEN);
                while (!(TWCR & (1<<TWINT)));           //test
        }
```

Obě tyto funkce provádí jednoduché vyslání dat na sběrnice, rozdílná jména slouží pro lepší přehled o tom, kdy se v hlavní funkci vysílají obyčejná data, a kdy adresa zařízení.

Proměnné *a* a *b* se požívají na probraní jednoho bytu určení na výstup, z hlavní funkce. Registr TWDR se musí naplnit hodnotou pomocní proměnné, a až pak se povolí vysílání. S nastavením bitů TWINT a TWEN pak se opět čeká na nastavení bitu TWINT což znamená konec vysílání daného slova.

9.4.4 Funkce: read()

```
unsigned char read(void)
    {
        TWCR = (1<<TWINT) | (1<<TWEN)| (1<<TWEA);
        while (!(TWCR & (1<<TWINT)));
        return(TWDR);
    }
```

Tato funkce přebere datové slovo ze sběrnice. Vedle povolení přístupu ke sběrnici se povoluje i vyslání potvrzujícího znaku příjmu ACK. Pak se čeká na nastavení bitu přerušení. Funkce vrátí hodnotu komunikačního registru TWDR.

9.5 Funkce pro vyslání znaků

Předchozí funkce nám umožňují posílat a přijímat údaje, ale musíme je zadat numerickou formou. Pro vypsání alfanumerických znaků by bylo příliš náročné a dlouhé zakódovat znaky ručně. Kvůli tomu jsme v souboru *fonts.h* nadefinovali numerické posloupnosti, které po vyslání na zobrazovací jednotku vykreslují znaky anglické abecedy. Pro vysílání těchto znaků jsme definovali následující funkce:

9.5.1 Funkce: lcd_clear()

```
void lcd_clear(void)
    {
        start();                                    // mazani LCD
        addressw(0x70);
        data(0x9);
        data(0x0);
        for ( ik=0;ik<505;ik++)
            {
                data(0x00);
            }
        stop();
    }
```

Pomocí této funkce nastavíme všechny bity v paměti LCD řadiče na úroveň logické 0. Důsledkem toho je, že se z panelu smažou všechny předcházející znaky.

Po vyslání start znaku se nastaví adresa zobrazovače. Následující dva byty jsou pro nastavení řídících bitů a X-Y souřadnice, odkud se má začít výpis po 8 pixel (bit) vysokých sloupcích ve čtyř řádcích. Pak cyklus pošle nulové byty pro každý jeden sloupec (4x101)

Přenos se končí stop znakem.

9.5.2 Funkce charout()

```
void charout(unsigned char znak[])
    {

        jk=strlen(znak);
        data(0b00000000);
        for (ik=0 ;ik<jk;ik++)
        {
        data(znak[ik]);
        }
}
```

Písmena a speciální znaky jsou definovány jako jednorozměrné pole s proměnnou délkou. Konec polí je určen bytem s hodnotou 0. Tato funkce zjistí délku pole a postupně vyšle každý prvek. Použije přitom funkci *data().* Abychom mohli používat funkci *strlen()* musíme na začátku implementovat knihovnu *string.h.*

Řádek *data(0b00000000);* je uveden proto, aby jednotlivé znaky, poslané těsně nesplývaly. Tím se zabezpečuje lepší čitelnost.

Tato funkce má ještě dvě modifikace, *bcharout()* a *icharout().*

První posune znak o jeden pixel níž. Použije se to na spodním řádku, aby se použitím celé plochy zobrazovače zvětšila vzdálenost alespoň mezi posledním a předposledním řádkem. Rozdíl je v pouze v jednom řádku :

data(znak[ik]<<1);

Druhá modifikace, *icharout()* invertuje barvy znaku. Je to potřebné k označení vybraného části v nabídce a taktéž pro označení, pro který den je aktivován alarm v pohotovostním režimu. Liší se o tyto dva řádky:

data(0b11111111);

data(~znak[ik]);

9.5.3 Funkce: s_numout()

```
void s_numout(unsigned char num)
    {
        data(0b00000000);
        for (ik=0 ;ik<5;ik++)
            {

                data(s_numbers[num][ik]);
            }
    }
```

Pro čísla je definováno jedno dvojrozměrné pole s délkou 10x5. Jsou obsaženy znaky od 0 do 9 se šířkou 5pixelů. Jelikož mají stejnou šířku, není třeba zjišťovat délku pro cyklus. Proto je pro cyklus dán pevný počet opakování: 5. Z hlavní funkce se přebere pouze číslo řádku což odpovídá číslici určení k vypsání. Před cyklem je i tady vyslán prázdný sloupec aby znaky nesplývaly.

Tato funkce má taky dvě modifikace, obdobně jako funkce *charout().* Jsou to *b_numout()* a *is_numout().* I tady první provede posun znaku o jeden pixel níž a druhý invertuje barvy pro označení v nabídce.

9.5.4 Funkce: h_numout()

Je další varianta výpisu číslic. Při pohotovostním –normálním- režimu jsou číslice, indikující hodiny, rozloženy na dva řádky. Musí se proto nejprve vyslat horní půlka číslice a pak dolní půlka. Pro tenhle postup je definovaná proměnná *h_munbers()*. Je to dvojrozměrné pole 20x8. Pro výpis těchto znaků je potřebná funkce *h_numout().*

```
void h_numout(unsigned char num)
    {
        data(0b00000000);
        for (ik=0 ;ik<8;ik++)
            {
                data(h_numbers[num][ik]);
            }
    }
```

Tady je cyklus proveden 8 krát a vysílají se znaky z jiné proměnné jak v prvním variantu. Použití funkce je také, že při výpisu horního řádku číslic se přebere z hlavní funkce číslo řádku v poli. Při druhém řádku se má před předáním k požadovanému číslu přičíst 10 a teprve poté zahájit výpis.

10 ZÁVĚR

Během naší práce jsme se postupně seznámili se součástkami potřebnými ke konstrukci časovacího obvodu a jeho programováním. Hlavním cílem bylo navrhnout a zkonstruovat časovací zařízení, které má mnohostranné a praktické využití v reálném životě. Základní funkce zařízení jsme rozšířili o možnost připojení dalších prvků na volné vývody mikrokontroléru, pro další potřebné využití. Jako např. možnost připojení libovolných čidel (měřiče teploty, atd.), resp. použití jako řídící prvek dalších zařízení, kde je zapnutí nebo vypnutí zařízení časově závislé.

Při návrhu jsme provedli průzkum a seznamování s potencionálními obvody a součástkami pro naše zařízení. Při každém rozhodnutí o použití jsme museli brát v potaz různé parametry, výhody a nevýhody daného prvku pro naše účely. Výběr konkrétních typů základních prvků jako samotný mikrokontrolér, obvod reálného času, DCF přijímač, komunikační jednotky (LCD, tlačítka), signalizační jednotky (siréna, LED dioda) a jednotky pro řízení napájení byl zdlouhavý a časově náročný proces. Museli jsme prostudovat různé materiály a katalogové listy, abychom si mohli zvolit tu nejoptimálnější kombinaci. I ta nejmenší nekompatibilita může způsobit nepřesnosti při fungování celého zařízení.

Realizaci zařízení jsme vykonali ve dvou etapách. První byla konstrukce a oživení přípravku, k tomu jsme použili navrhnuté zapojení. Podle seznamu součástek jsme zabezpečili potřebné součástky, poté jsme zhotovili plošný spoj a vyhotovili zařízení. I když jsme si velmi pečlivě navrhli naše zařízení, při realizaci se vyskytly různé, nečekané události, kvůli kterým jsme museli změnit původní návrh a použit při řešení jiné součástky. Tyto změny jsem popsal na začátku 8. kapitoly.

Druhým krokem bylo sestavit program v jazyce C pro mikrokontrolér. V programu jsme zahrnuli řešení pro jednotlivé situace, které mohou nastat při běžném použití. Po zapnutí zařízení se na zobrazovací jednotce objeví aktuální datum se zkratkou dne, přesný čas a dni v týdnu, které jsou označené, tedy pro které je alarm zapnutý. Další funkce programu umožní nastavovat parametry budíku pomocí tlačítek, která jsou umístněna na horní straně krabice.

Po realizaci předešlých kroků je naše zařízení kompletní a vyhovuje funkčními požadavky, které byly stanoveny. Následně je možné využit výrobek jako obyčejný budík nebo pří návrhu dalších obvodů, kde je potřeba znát aktuální a přesný čas.

Použitá literatura

[1] Váňa Vladimír Mikrokontroléry Atmel AVR-popis procesoru a instrukční soubor, Praha, BEN – technická literatura 2003

[2] Internetový portál firmy Atmel http://www.atmel.com/

[3] Katalogový list mikrokontroléru ATmega16 http://www.atmel.com/dyn/resources/prod_documents/doc2466.pdf

[4] Katalogový list obvodu reálního času M41T81S http://www.st.com/stonline/products/literature/ds/10773/m41t81s.pdf

[5] Internetový portál Interfacebus http://www.interfacebus.com/I2C_Interface_Bus_Standard.html

[6] Dokumentace protokolu I2C http://rifer.narod.ru/i2c/I2C.pdf

[7] Internetový portál E-conrad http://www.e-conrad.cz/deska-prijimace-dcf+dp55380/

[8] Internetový portál Compuphase http://www.compuphase.com/mp3/h0420_timecode.htm

[9] Internetový portál Wikipedia http://www.wikipedia.cz

[10] Katalogový list obvodu LT1944 http://cds.linear.com/docs/Datasheet/1944f.pdf

[11] Internetový portál serdisplib http://serdisplib.sourceforge.net/ser/i2c_ericssont2x.html

[12] Internetový portál berty´shomepage http://sandiding.tripod.com/Bertys.html a http://sandiding.tripod.com/lcdt28.html

[13] Internetový portál rifer http://rifer.narod.ru/i2cldc.html

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

DCF – D = deutschland C = pásmo dlouhých vln F = fankfurt

TWI – Two Wire Interface

I2C – Inter Integrated Circuit

LCD – Liquid Crystal Display

DC/DC – stejnosměrný měnič napětí

SMD – Surface-Mount Device

UTC – Coordinated Universal Time

BCD – Binary Coded Decimal

AVR – Alf and Vegard´s RISC(Reduced Instruction Set Computer)

RISC – Reduced Instruction Set Computer

LED – Light Emitting Diode

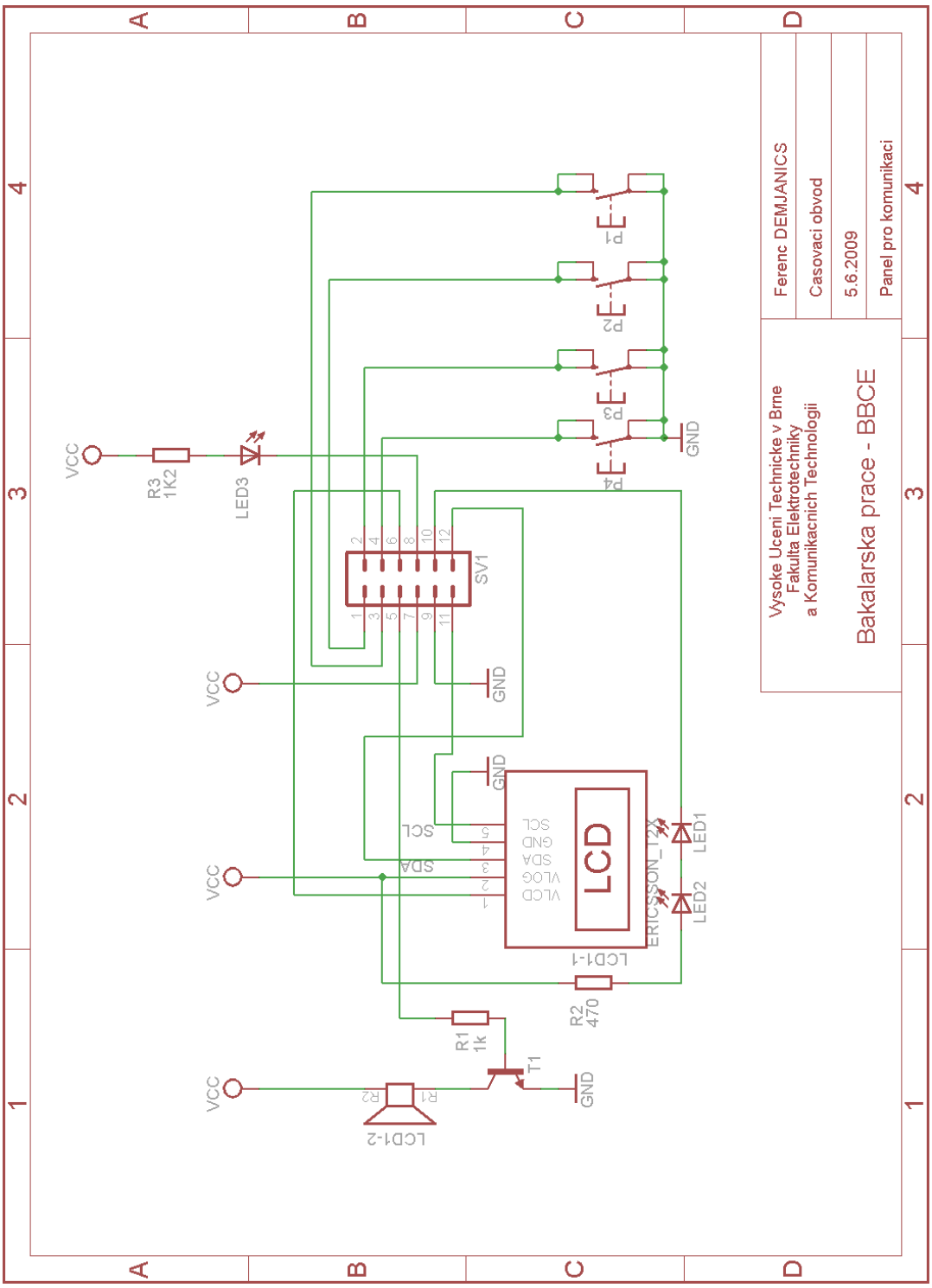
ISP – In System Programming

JTAG – Joint Test Action Group

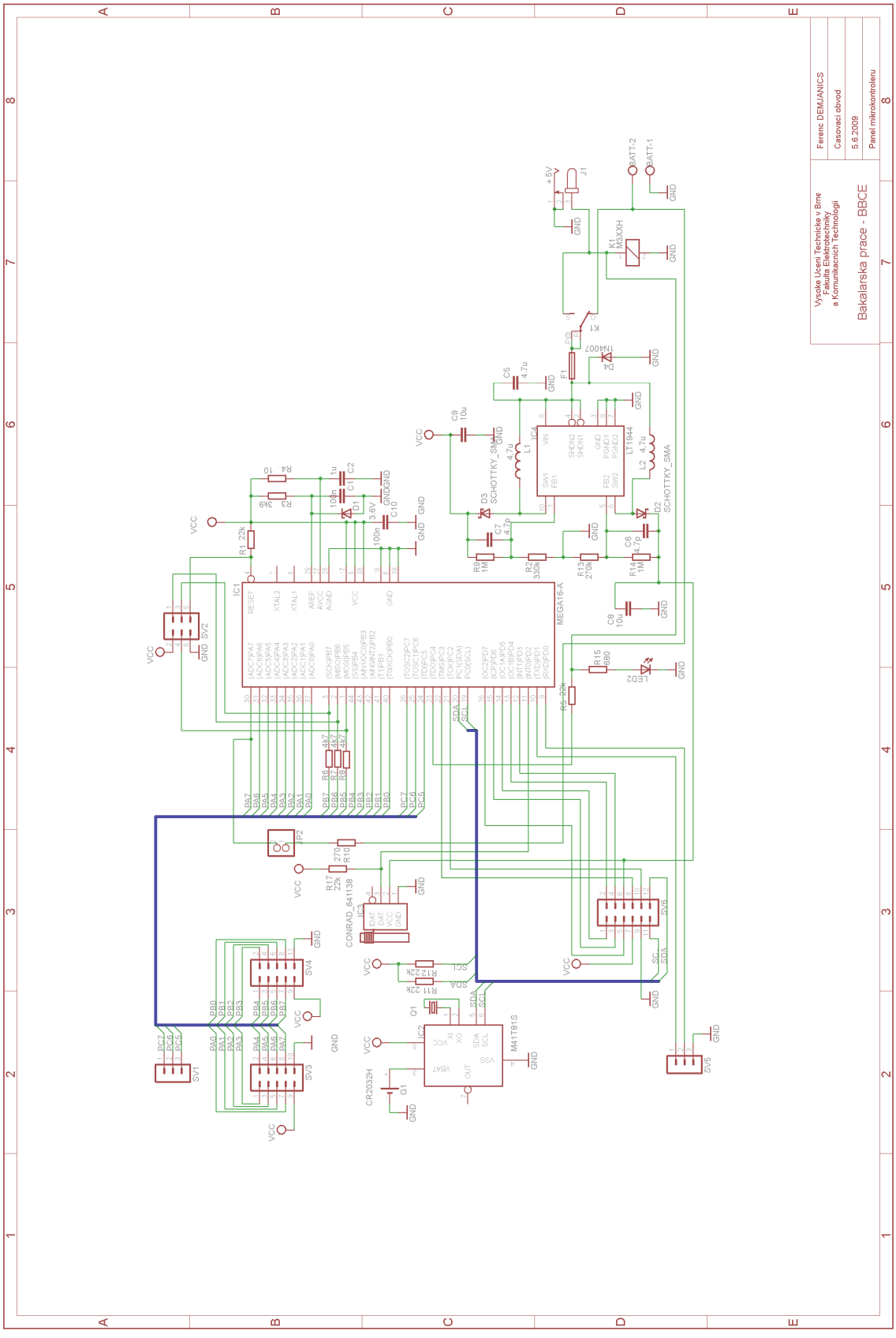
SEZNAM PŘÍLOH

A NÁVRH ZAŘÍZENÍ

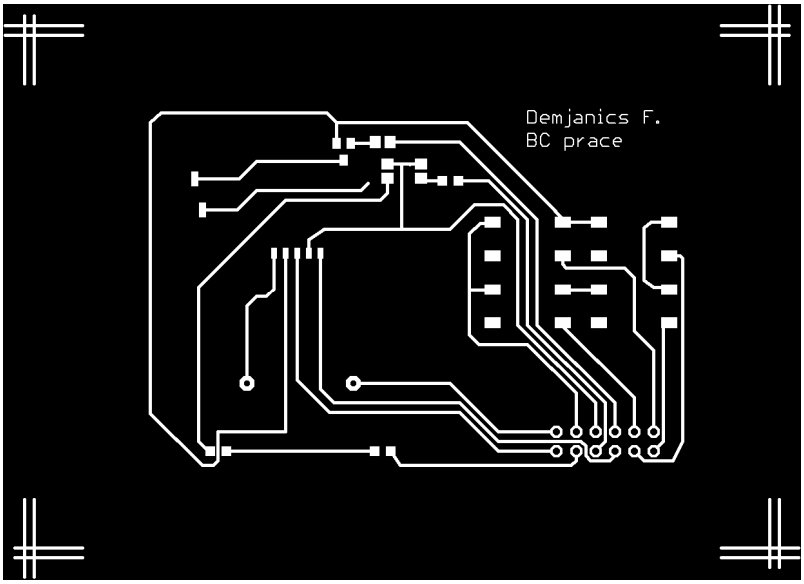
A.1 Obvodové zapojení panelu 1 pro komunikaci



A.2 Obvodové zapojení panelu 2 mikrokontroléru

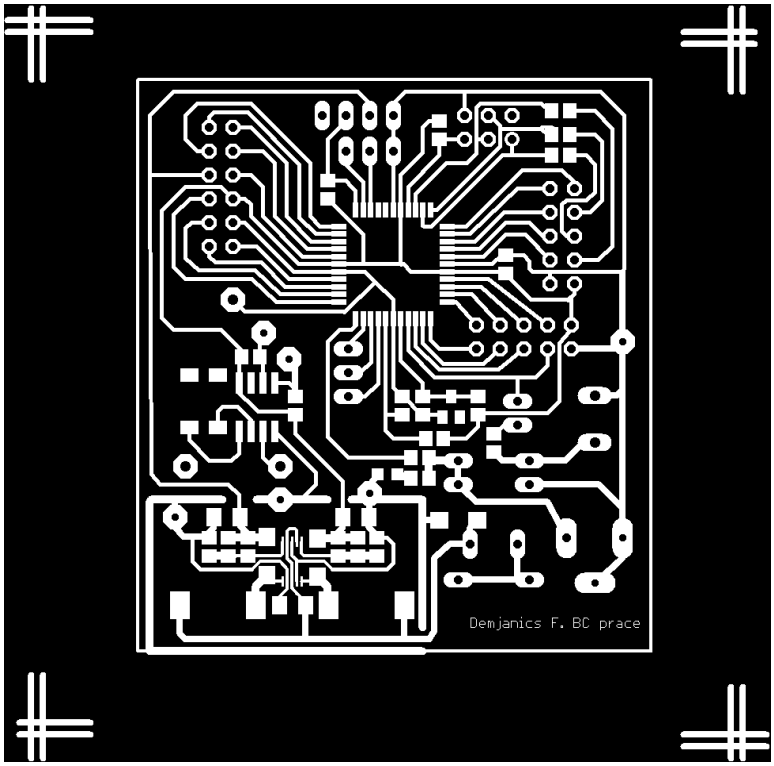


A.3 Deska plošného spoje 1 – top (strana součástek)



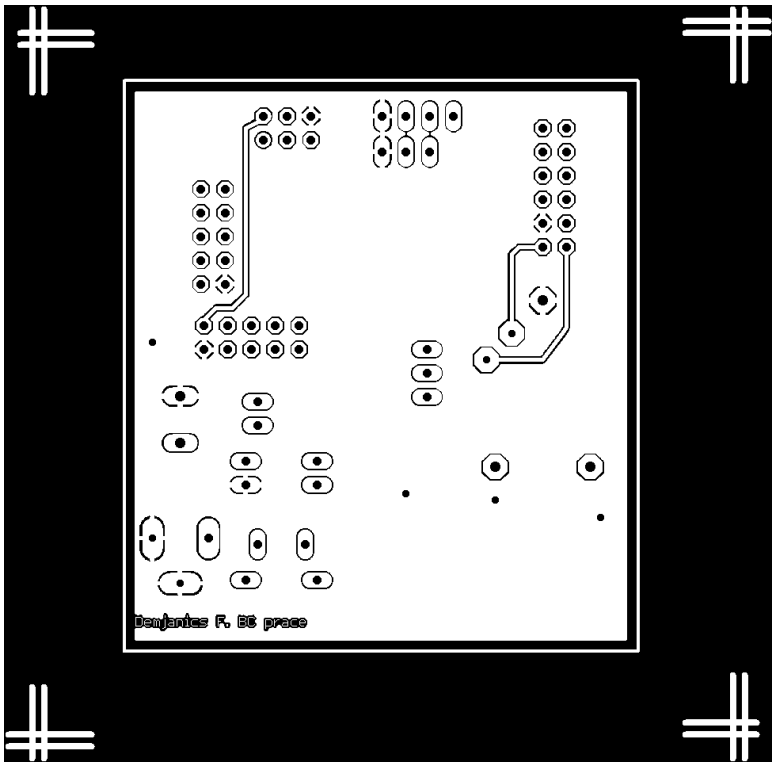
Rozměr desky 75 x 54 [mm]

A.4 Deska plošného spoje 2 – top (strana součástek)



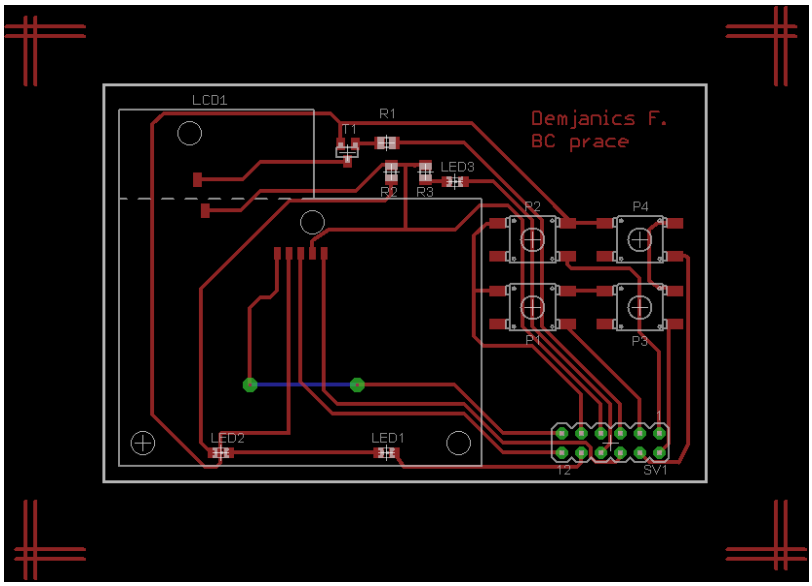
Rozměr desky 62 x 58 [mm]

A.5 Deska plošného spoje 2 – bottom (strana spojů)

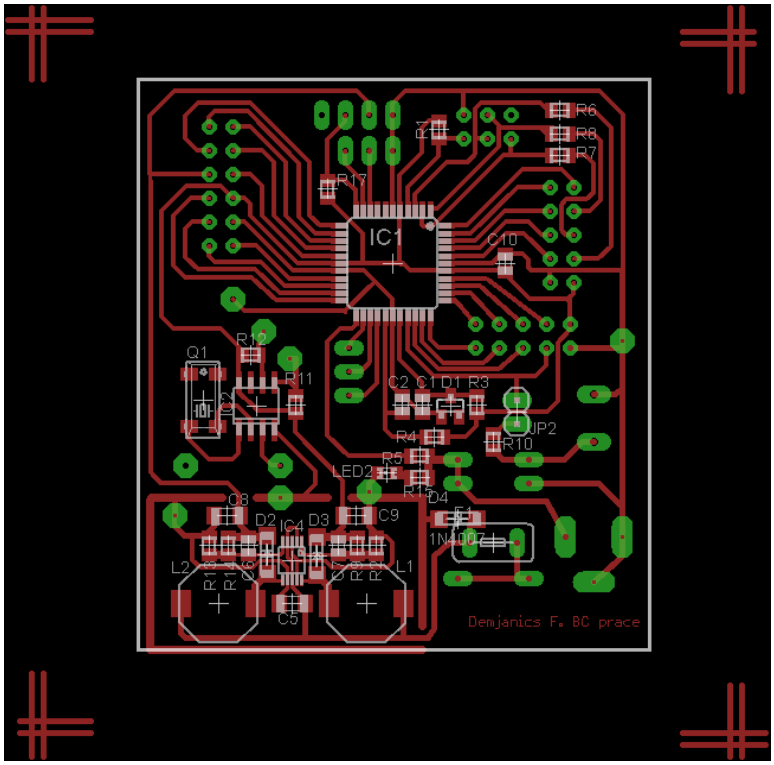


Rozměr desky 62 x 58 [mm]

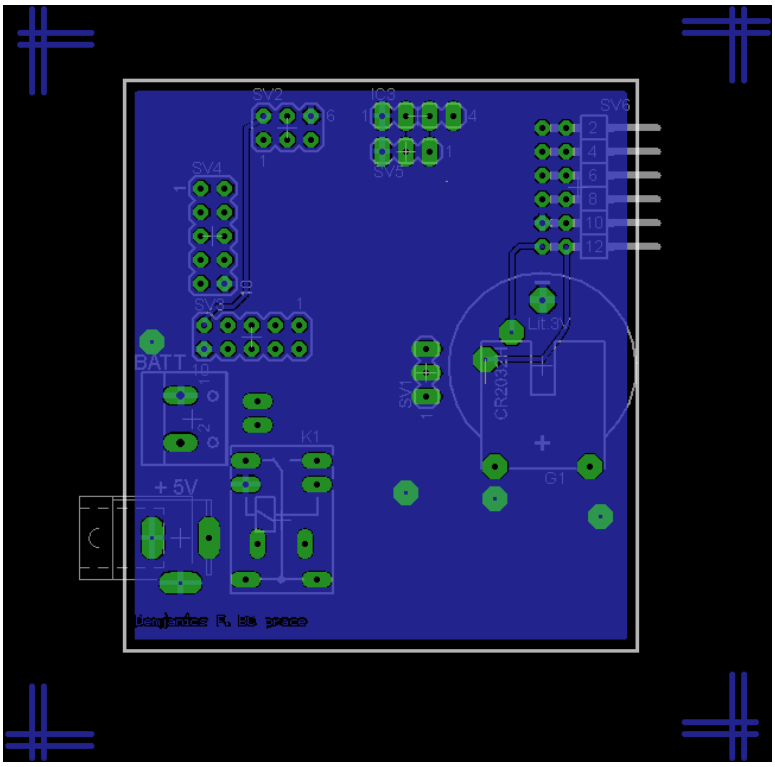
A.6 Schéma pro osazení 1 – top (strana součástek)



A.7 Schéma pro osazení 2 – top (strana součástek)



A.8 Schéma pro osazení 2 – bottom (strana spojů)



B SEZNAM SOUČÁSTEK

B.1 Pro desku plošného spoje 1

Označení	Hodnota	Pouzdro
LCD1		ERICSSON_T2X
LED1		CHIPLED_0805
LED3		CHIPLED_0805
LED4		CHIPLED_0805
P1		
P2		
P3		
P4		
R1	1K2	R0805
R2	3k9	R0805
R9	1k	R0805
R18	470	R0805
SV6	konektor 2x6	
T1	BC808	SOT23

B.2 Pro desku plošného spoje 2

Označení	Hodnota	Pouzdro
C1	100n	C0805
C2	1u	C0805
C3	100n	C0805
C5	4.7u	C1206
C6	4.7p	C0805
C7	4.7p	C0805
C8	10u	C1206
C9	10u	C1206
C10	100n	C0805
D1	3.6V ZENER	SOT23
D2	SCHOTTKY	SMA
D3	SCHOTTKY	SMA
G1	3 V lith.	CR2032H
IC1	MEGA16-A	TQFP44
IC2	M41T81S	SO08
IC3	CONRAD_641138	
IC4	LT1944	MSOP10
J1	power supply	
JP2		jumper
L1	4,7u	SCB8D43
L2	4,7u	SCB8D43
LED2		CHIPLED_0805
Q1	32,768kHz	MM20SS
R1	22k	M0805
R2	330k	M0805
R3	3k9	M0805
R4	10	M0805
R6	4k7	M0805
R7	4k7	M0805
R8	4k7	M0805

R9	1M	M0805
R10	270	M0805
R11	22k	M0805
R12	22k	M0805
R13	270k	M0805
R14	1M	M0805
R15	680	M0805
R17	22k	M0805
SV1	konektor 1x3	
SV2	konektor 2x3	
SV3	konektor 2x5	
SV4	konektor 2x5	
SV5	konektor 1x3	
SV6	konektor 2x6	
X1	batt input	