ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA APLIKOVANÉ ELEKTRONIKY A TELEKOMUNIKACÍ

DIPLOMOVÁ PRÁCE

Monitorovací systém malého čerpadla

Jan Vrána **2012/2013**

Originál (kopie) zadání BP/DP

Abstrakt

Předkládaná diplomová práce je zaměřena na řešení monitorování chodu domácích vodáren. Dlouhodobě nedetekovaná porucha v instalaci rozvodu vody může napáchat nemalé škody. Práce je v první řadě zaměřena na přiblížení domácích vodáren a chybových stavů, které mohou při jejich provozu nastat. Následně je práce zaměřena na popis metody pro detekování poruchového stavu a stavbu jednoduchého systému, který popsanou metodu využívá. V závěru je simulováno několik průběhů, jimiž byla funkce monitorovacího systému ověřena.

Klíčová slova

Domácí vodárna, integrační metoda, microcontroller, porucha, doba zapnutí / vypnutí, tlaková nádoba, rozvod vody, únik vody

Abstract

This thesis is focused on a monitoring system for domestic waterworks. Long-term undetected fault in the installation of water supply can cause serious damage. This thesis provides a description of domestic waterworks first. Then it describes methods for detection of a fault condition and a construction of a simple system which can monitor the pump. The final part consists of some graphs and waveforms for verifying the function of the monitoring system.

Key words

Domestic waterworks, integration method, microcontroller, failure, on / off time, pressure vessel, water supply, water leak.

Prohlášení

Prohlašuji, že jsem tuto diplomovou/bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

Dále prohlašuji, že veškerý software, použitý při řešení této bakalářské/diplomové práce, je legální.

.....

podpis

V Plzni dne 7.5.2013 Jan Vrána

Poděkování

Tímto bych chtěl především poděkovat panu Ing. Václavu Kouckému, CSc. za jeho odborné vedení a připomínky důležité pro realizaci mé diplomové práce. Zároveň bych chtěl poděkovat mé rodině a přátelům za podporu v průběhu celého mého studia.

Obsah

Seznam symbolů a zkratek

Vss	…	zemní potenciál 0 V
GND	…	zemní potenciál 0 V
Vdd	…	napájecí napětí
Vcc	…	napájecí napětí
Vo	…	napětí určující kontrast displeje
RS	…	signál instrukce / data
R/W	…	signál čtení / zápis
E	…	signál ENABLE
DB0-7	…	osm datových vodičů
V_{BAT}	…	pin pro externí baterii
X1 a X2	…	piny pro připojení krystalického oscilátoru
I^2C	…	druh komunikačního protokolu
SDA	…	datový pin I^2C komunikace
SDC	…	hodinový pin I^2C komunikace
SQW/OUT	…	pin pro výstupní obdélníkový signál
OUT	…	výstup
IN	…	vstup
RTC	…	Real Time Clock – obvod reálného času
LED	…	Light Emission Diode – světlo emitující dioda
LCD	…	Liquid Cristal Display – displej s tekutými krystaly
A/D	…	analogově-digitální převodník

Úvod

Předkládaná diplomová práce je zaměřena na řešení monitorování čerpadel pro domácí vodárny. Domácí vodárny vybavené malým čerpadlem mohou být velmi nápomocny při využívání vody z přilehlé studně, vrtu či vodního toku. Přes jejich výhody však může automatický systém, čerpající vodu, napáchat nemalé škody. V případě prasknutí vodovodního potrubí může včasný zásah ušetřit majiteli nemovitosti nemalé peníze. V této práci jsou přiblíženy typy domácích vodáren, havarijní stavy, které mohou při jejich provozování nastat a jsou nastíněny možnosti jak poruchové stavy detekovat. Dále je v této práci přiblížena výroba jednoduchého systému schopného detekovat a následně reagovat na havarijní stavy. V závěru práce jsou popsány průběhy, jimiž byla ověřena funkce navrženého systému.

Jako hlavní cíl práce jsem si určil vyvinout metodu pro monitorování chodu domácí vodárny. Dále navrhnout jednoduchý systém, který bude navrženou metodu implementovat a díky němuž budu schopen ověřit své výsledky.

1 Teoretická příprava

1.1 Způsoby detekce stavu systému a případných poruch

1.1.1 Senzory vlhkosti

Použití senzorů v kritických bodech domácího rozvodu vody může být velmi nápomocno při včasné detekci poruchy ve formě praskliny či netěsnosti potrubí.

Na konečnou cenu systému se senzory vlhkosti však bude mít značný vliv počet a vzdálenost kritických bodů vodovodního rozvodu. Řešení bude následně zahrnovat volbu komunikace a způsob napájení jednotlivých čidel. Při zahrnutí těchto faktorů lze předpokládat, že řešení povede na nákladný a komplikovaný systém.

1.1.2 Senzor tlaku

Tento způsob realizace předpokládá, že je tlaková vodní nádrž již opatřena tímto senzorem. Rychlost změny tlaku v nádrži by mohla následně odpovídat rychlosti vypouštění. Změna tlaku však závislá ještě na velikosti vzduchové bubliny v nádrži, což má za následek, že při malé vzduchové bublině vyvolá malá změna hladiny vody v nádrži velkou změnu tlaku. Tato velká změna tlaku může být následně chybně vyhodnocena jako velký odtok vody.

1.1.3 Senzor průtoku

Při aplikaci systému se senzorem průtoku bude pravděpodobně zapotřebí zasáhnout do rozvodu vody a za tlakovou nádrž umístit průtokový senzor. V případě aplikace by tento systém byl pravděpodobně schopný detekovat všechny chyby, které by se vyskytly za tímto senzorem. Chyby, které by se týkaly čerpadla a nádrže by však nebyly zaznamenány. Například již zmiňovaná malá vzduchová mezera by se projevila častým spínáním čerpadla, což by se za expanzní nádraží neprojevilo, protože průtok bude závislý pouze na spotřebovávaném množství vody.

1.1.4 Senzor proudu

Aplikace se senzorem proudu má zásadní výhodu v minimálním zásahu do systému. Proud může být měřen proudovým transformátorem na přívodním vodiči napájení čerpadla. Podle měření proudu může být vyhodnocováno nejen zapnutí a vypnutí čerpadla, ale v případě potřeby může být měřen i elektrický proud protékající čerpadlem. Jako chyby mohou být vyhodnoceny: časté spínání čerpadla (příliš velký odběr vody nebo malá vzduchová bublina v expanzní nádrži), dlouhá doba čerpání (příliš velký odběr vody nebo nepřítomnost vody ve studni) případně velký proudu (přibrzděný rotor čerpadla nebo čerpání z velké hloubky), který může způsobit přehřívání čerpadla.

1.2 Chyby, které mohou nastat

1.2.1 Prasklý rozvod vody - velký průtok

Velký průtok vody prasklinou bude pravděpodobně znamenat, že se bude čerpadlo zapínat častěji, než při běžném užívání vody. Doba nečinnosti se tedy zkrátí a naopak dojde k prodloužení doby sepnutí čerpadla. V krajním případě může být čerpadlo nepřetržitě v chodu. Dá se předpokládat, že zvýšené namáhání čerpadla se projeví i jeho ohřátím. Senzorem tlaku bychom teoreticky mohli zaznamenat průběh, kdy rostoucí tlak při čerpání střídá bezprostředně pokles tlaku (z důvodu nepřetržitého odtoku vody prasklinou) při nečinnosti čerpadla. Senzor průtoku by v takovémto případě zaznamenal velký a trvalý průtok.

1.2.2 Prasklý rozvod vody - malý průtok

Lze předpokládat, že malý průtok prasklinou bude nejsnáze detekovatelný čidlem průtoku. Systém, vybavený tímto čidlem, by mohl zmíněnou chybu detekovat jako stálý, nenulový průtok. Při použití systému s čidlem tlaku uvnitř expanzní nádoby by měla data, v případě vypnutého čerpadla a za předpokladu že nedochází k úmyslnému odběru vody, představovat plynulý klesající průběh. Systémem opatřeným senzorem proudu je pro tento případ relevantní pouze detekce zapnutí a vypnutí čerpadla. V případě nezanedbatelného úniku vody dojde během dobře zvoleného časového okna k několika sepnutím čerpadla

z důvodu doplnění uniklé vody. Přesáhne-li následně doba, po kterou bylo čerpadlo zapnuté, v rámci časového okna předem nastavenou hodnotu, bude tento stav vyhodnocen jako chyba.

Malý průtok vody může být zaměněn s činnostmi jako například mytí nádobí, a proto je potřeba přidat další parametr pro přesnější detekci takovéto chyby. Tímto parametrem by mohla být doba v rámci jednoho dne. Lze předpokládat, že v nočních hodinách nebude docházet k mytí nádobí zavlažování ani jiným podobným činnostem. Při zvážení tohoto předpokladu by měla být prováděna detekce malého průtoku prasklinou ve vodovodním potrubí právě v nočních hodinách.

1.2.3 Malá velikost expanzního vzduchového polštáře

Tuto chybu pravděpodobně nelze detekovat senzorem průtoku, jelikož je tento senzor umístěn až za vodní nádrží a o vyrovnávání tlaku, potřebného k vytlačení vody z nádrže, se automaticky stará čerpadlo. Jinak je tomu u senzoru tlaku, neboť se předpokládá jeho umístění v tlakové vodní nádrži. Malá velikost vzduchového polštáře by tedy mohla být detekována systémem s měřením tlaku jako velmi strmý nárůst tlaku při čerpání. Uvažujeme-li senzor měření proudu, lze malou velikost vzduchového polštáře v tlakové vodní nádrži detekovat jako častou aktivaci čerpadla s krátkou dobou čerpání. Pro tento případ lze tedy hlídat četnost spínání čerpadla a podtečení minimální doby, po kterou je čerpadlo zapnuté za standardních podmínek.

1.2.4 Nestandardní zvětšení expanzního vzduchového polštáře

Velká vzduchová bublina v tlakové vodní nádrži není chybou, jenž by mohla napáchat škody na majetku, avšak její výskyt může uživateli znesnadnit život. Sama o sobě je tato nestandardní situace projeví ucházením vzduchu pomoci vodovodního rozvodu (tzv. prskáním vody). Pomocí čidla tlaku může být oproti předchozí nestandardní situaci rozpoznán pomalý nárůst tlaku uvnitř tlakové vodní nádrže. Čidlem průtoku nebudeme schopní tuto nestandardní situaci odhalit podobně jako v předchozím případě. Pravděpodobně se prodlouží i nutná doba čerpání pro dosažení vypínacího tlaku, a proto předpokládám, že během určitého časového úseku dojde k nárůstu doby čerpání na úkor doby nečinnosti čerpadla.

1.3 Typy domácích vodáren

1.3.1 Domácí vodárna „Darling“ s tlakovou nádrží

Takováto vodárna je vhodná pro použití v podmínkách České republiky. Je vhodnou variantou pro čerpání vody ze studny, vrtu nebo blízkého vodního toku, avšak může být použita i pro čerpání dešťové vody z rezervoáru. Domácí vodárna tohoto typu se skládá z tlakové vodní nádrže, odstředivého či pístového čerpadla, sacího potrubí, tlakového spínače a ovládání. Existují dva zástupci této skupiny.

Obrázek 1.1 Ukázka domácí vodárny s čerpadlem [4]

Prvním je tlaková vodní nádoba s pryžovou membránou (vakem) jenž je umístěna uvnitř tlakové nádoby. Při zapnutí čerpadla je voda čerpána do pryžového vaku přičemž proti rozpínání vaku působí tlak vzduchu uvnitř expanzní nádoby. Tlak vzduchu uvnitř vodní nádrže musí být přibližně třikrát do roka překontrolován a případně doplněn. Po vypnutí čerpadla je voda do vodovodního rozvodu vytlačována smršťujícím se pryžovým vakem. Nevýhodou tohoto typu expanzní nádrže je skutečnost, že pryžový vak si své elastické vlastnosti neudrží věčně a proto musí po čase dojít k jeho výměně. Tento systém čerpání vody šetří čerpadlo, nebo je doba jeho chodu zkrácena na čas nezbytný k napuštění vody do tlakové vodní nádrže. V praxi se čerpadlo dimenzuje na šest zapnutí za jednu hodinu. [4] [5]

Druhým zástupcem domácích vodáren s tlakovou nádrží je systém s vodní nádrží bez membrány. Jak už napovídá název, tento systém se obejde bez pryžového vaku. Voda je zde čerpána přímo do prostoru tlakové vodní nádrže, v níž je přítomen patřičný objem vzduchu. Po vypnutí čerpadla je tlak ve vodovodním rozvodu udržován stlačeným vzduchovým polštářem. Tlak uvnitř expanzní nádrže musí být rovněž kontrolován. U některých systémů se předpokládá doplňování vzduchu do expanzní nádrže přes přívod čerpadla.

Každý z těchto dvou systémů může mít ještě dvě obdoby. Rozdíl je totiž v tom, je-li čerpadlo součástí vodárny anebo je používáno ponorné čerpadlo.

Pokud je čerpadlo součástí vodárny, nemůžeme vodu čerpat z větší hloubky než přibližně 7,6 m. Takto hloubka souvisí s výškou barometrického vodního sloupce. Existuje tedy jisté omezení pro hloubku čerpání. Po překročení konkrétní hloubky přestane čerpadlo i ve stavu zapnutí dodávat vodu do vodní nádrže. Pokud čerpadlo setrvává v tomto stavu, mělo by dojít co nejdříve k odpojení čerpadla od zdroje. Rozpoznat tento stav přitom není složité. Jak bude nastíněno později, havarijní stav projevující se dlouhou dobou sepnutí je, při použití systému s měřením proudu, nejsnadněji detekovatelným havarijním stavem. Na druhou stranu je provedení domácí vodárny společně s čerpadlem do jisté míry bezpečnější než druhý způsob. Je tomu tak proto, že čerpadlo nemůže vyčerpat ze zdroje jen takový objem vody, který se nachází nad hranicí, která určuje maximální hloubku čerpání. Čerpadlo, které je součástí domácí vodárny vytváří sáním na svém vstupu podtlak. Tohoto jevu se v některých případech využívá k doplňování vzduchu do tlakové vodní nádrže bez membrány. Před čerpadlo je do přívodního potrubí umístěn tlakový ventil, jímž je při dostatečném podtlaku nasáván vzduch z okolí ventilu do sacího potrubí čerpadla. Po vypnutí čerpadla podtlak uvnitř sacího potrubí klesne a tím se přivzdušňovací ventil uzavře. Díky tomuto podtlakovému ventilu tedy dochází k samovolnému doplňování vzduchu uvnitř tlakové vodní nádrže.

V případě že čerpadlo není součástí domácí vodárny a je použito ponorné čerpadlo, může být vyčerpán podstatně větší objem vody než v předchozím případě, za předpokladu že je čerpáno z dostatečného zdroje vody. Systém s ponorným čerpadlem se hodí do hlubinných vrtů a je schopen vytlačit vodu do výtlačné výšky, která je pro všechna čerpadla větší než sací výška. Ponornému čerpadlu hrozí jedině přehřátí, a to v případě, že vyčerpá vodu, která ho obklopuje a začne nasávat vzduch. Přehřátí lze zabránit jednoduchou úpravou a to umístění

vstupu čerpadla nad samotný motor. V takovém případě i po odčerpání vody až po sací vstup zůstane motor čerpadla ponořený ve zbytku vody.

1.3.2 Domácí vodárna s otevřeným vodojemem

Takováto domácí vodárna se umisťuje na střechu budovy. Je potřeba dbát na statiku umístění a dobré zabezpečení stability vodojemu. Při čerpání je poté voda čerpána právě do této nádrže, odkud je samospádem dopravována do domácího rozvodu vody. Při této realizaci je zapotřebí pouze jedno slabší čerpadlo, které čerpá vodu do vodojemu. Další čerpadla nejsou nutná, neboť se o tlak vody postará potenciální energie. Kromě statiky objektu je potřeba zabezpečit vodu proti zamrzání. Z těchto důvodů vyplývá, že v klimatických podmínkách České republiky je tento druh domácí vodárny téměř nepoužitelný. I přes to se s ním však můžeme setkat v podobě velkých vodárenských věží poblíž vesnic a menších měst. [5], [4]

2 Praktická část

Při vytváření systému pro monitorování malého čerpadla domácí vodárny jsem se zaměřil na nalezení jednoduchého systému schopného detekovat co možná nejvíce chyb. Při aplikaci systému na fungující domácí vodárnu by navíc nemělo být zapotřebí provádět složitý zásah do rozvodů vody.

Pro svou práci jsem si tedy vybral systém se senzorem proudu. Hlavní výhodou tohoto systému je jeho instalace. Pro jeho montáž je totiž zapotřebí pouze manipulace s elektrickým napájením čerpadla.

2.1 Metoda detekce poruchového stavu

V prvním přiblížení lze považovat dlouhou dobu čerpání za vážnou hrozbu. Tento stav může způsobit více eventualit:

- Čerpadlo je součástí vodárny a výška vodního sloupce přesáhla maximální výšku
- Čerpadlo je ponořené ve zdroji vody avšak došlo k vyčerpání zdroje a čerpadlo čerpá „na prázdno“
- Rychlost vypouštění vody je tak velká, že je srovnatelná s rychlostí dočerpávání vody do tlakové vodní nádrže

Pro detekování stavu dlouhého zapnutí je zapotřebí mít informaci o délce zapnutí čerpadla, což lze provést inkrementováním k tomu určené proměnné po každé vteřině ve stavu zapnutí čerpadla. Vyhodnocení bude potom provedeno jednoduchým porovnáním s přednastavenou konstantou, jejíž velikost odpovídá maximální přípustné délce čerpání.

Opačný případ, kdy se čerpadlo vypne po nepřirozeně krátké době od začátku čerpání je taktéž nepřípustný. Příčinou takovéhoto stavu bývá nedostatečný objem vzduchu ve vodní nádrži. Díky tomu malá změna objemu vody způsobí velkou změnu tlaku uvnitř vodní nádrže. Stačí tedy odebrat menší množství vody pro dosažení hodnoty tlaku, při které dojde k zapnutí čerpadla. Čerpadlo dokáže dočerpat menší množství vody za kratší časový úsek. Po doplnění odpovídajícího objemu vody dosáhne tlak uvnitř vodní nádrže hodnoty, při které dojde

k vypnutí čerpadla. Z toho vyplývá, že se s klesajícím objemem vzduchu ve vodní nádrži zkracuje doba zapnutí čerpadla. Detekce potom může být prováděna podobně jako v prvním případě inkrementováním k tomu určené proměnné po každé vteřině ve stavu zapnutí čerpadla. Na rozdíl od předešlého případu nelze provádět obyčejné porovnávání s konstantou určující minimální přípustnou délku čerpání. K porovnání musí dojít až po vypnutí čerpadla, ale dříve, než dojde k vynulování proměnné pro použití v dalším cyklu zapnutí čerpadla. Nejprve totiž musí být změřeno, jak dlouho bylo čerpadlo v zapnutém stavu, a teprve potom mohu rozhodnout, jestli byla doba čerpání příliš krátká.

Vážným problémem může být nekontrolovatelný únik vody. K tomuto stavu může dojít prasklinou ve vodovodním rozvodu, poruchou vodovodního kohoutku, nebo chybou uživatele, kdy zapomene uzavřít kohoutek. Při opravdu rychlém vypouštění vody může dojít k prodloužení doby čerpání, které však nebude nikterak radikální vzhledem k rychlosti, jakou je čerpadlo schopné doplnit vodu v nádrži. Pro detekci tohoto potenciálně poruchového stavu tedy bude zapotřebí komplexnější metoda pozorování. Jako první mě při realizaci napadlo počítat dobu zapnutí čerpadla a dobu, po kterou vylo čerpadlo vypnuté a v závislosti na poměru těchto dvou časů vyhodnocovat, zdali došlo k poruše. Od tohoto přístupu jsem nakonec ustoupil, protože před dalším cyklem čerpání případně nečinnosti bylo zapotřebí příslušnou proměnnou vynulovat a tím docházelo ke ztrátě informace o dlouhodobějším chování.

Pro detekci jsem použil metodu, dále popisovanou jako integrační. Princip této metody je podstatně průhlednější než výše popsaná metoda. Pro její funkci je zapotřebí pouze jedné proměnné (nazveme jí pro tento případ SUMA), která je neustále každou vteřinu inkrementována nebo dekrementována v závislosti na stavu vypnutí či zapnutí čerpadla. Důležitými parametry této metody jsou inkrementační a dekrementační konstanta a konstanta pro porovnání, zdali došlo nebo nedošlo k potenciální poruše. Velikost inkrementační a dekrementační konstanty je obecně rozdílná, přičemž je předpoklad, že konstanta pro inkrementování bude větší. Při správném nastavení se bude hodnota proměnné SUMA pohybovat stále v přípustných mezích díky tomu, že čerpadlo setrvá ve vypnutém stavu minimálně tak dlouho, dokud nedojde k vynulování SUMY. V případě že by se situace změnila a doba, ve které čerpadlo setrvává ve vypnutém stavu, by se zkracovala, nedocházelo by k vynulování SUMY před započetím dalšího čerpání. Díky tomu bude velikost SUMY na konci následujícího stavu čerpání vyšší, než tomu bylo v předchozí „periodě“. Pokud bude

takovýto stav přetrvávat, velikost SUMY se vyšplhá až na hladinu, kdy po porovnání velikosti SUMY a konstanty určující maximální přípustnou velikost SUMY dojde k vyhodnocení situace jako potencionálně poruchové a bude proveden patřičný zásah.

Pravděpodobnějším poruchovým stavem však může být menší únik vody. Takový únik vody by mohl být snadno zaměnitelný s činnostmi jako je praní, napouštění vany či mytí auta, a proto nemůže být provedena detekce této poruchy výše uvedeným způsobem. Naštěstí jsou popsané činnosti prováděny téměř výhradně v denních hodinách. V nočních hodinách potom dochází k podstatně menšímu využívání vody, a díky tomu by mohl být malý trvalý únik vody detekovatelný během nočních hodin. Pro tento případ jsou inkrementační a dekrementační konstanta duplikovány a označeny jako denní a noční konstanty. V momentě rozpoznání každého nového začátku čerpání je na základě aktuálního času z obvodu RTC rozhodnuto, zdali se budou inkrementovat a dekrementovat denní či noční konstanty. Obecně lze předpokládat, že doba čerpání je přibližně stejná v noci i ve dne zatímco doba, kdy je čerpadlo vypnuté se v noci podstatně prodlouží. Z toho vyplývá, že hlavní rozdíl bude v dekrementační konstantě, která bude pro noční hodiny výrazně menší než pro denní. Díky tomuto nastavení, bude-li docházet k častějšímu spínání čerpadla, než je v nočních hodinách přípustné, dojde stejně jako v předešlém případě k navýšení proměnné SUMA nad povolenou mez a stav bude vyhodnocen jako potenciální porucha. Čím větší je únik vody, tím dříve dojde k jeho rozpoznání.

2.2 Napájení

Napájení monitorovacího systému je řešeno připojením na síť elektrické energie. Baterií se zálohuje pouze čas v obvodu RTC. Napětí pro microcontroller nemusí být zálohováno, protože při výpadku sítě nemůže dojít k aktivaci čerpadla a tudíž se nemůže vyskytnout ani havarijní stav. Celý monitorovací systém je napájen ze zdroje s napětím 9 nebo 12 V. Takovým zdrojem může být například zakoupený adaptér. Napětí adaptéru následně snížíme lineárním stabilizátorem na potřebných 5 V. [8]

2.3 Způsob zadávání

Obsluha mnou vyvrženého systému je řešena pomocí čtyř tlačítek. Pro čtyři tlačítka jsem se rozhodl, z důvodu úspory pinů microcontrolleru a šlo o minimální počet pro zadání

správného data a času. Dvě tlačítka jsou přitom použita pro inkrementování či dekrementování číslice představující aktuální hodnotu času nebo data. Třetí tlačítko má funkci potvrzování navolené hodnoty a poslední tlačítko slouží k vrácení se o krok nazpět.

Při běhu monitorování může být jedno tlačítko použito na dočasné blokování funkce. Každým jeho stisknutím potom můžeme prodloužit čas, po který vyřadíme monitorovací systém z činnosti. Takové vyřazení z činnosti může být využito v případě, že uživatel plánuje odebírat po určitou dobu větší množství vody. Takovým případem může být například napouštění bazénu. Díky tomuto blokování funkce je možné zamezit nežádoucímu detekování chyby a následnému odpojení čerpadla od zdroje elektrické energie.

2.4 Signalizace

Signalizace je řešena pomocí LED diod, indikujících chod systému a havarijní stavy, a znakového displeje, který poskytuje ucelenější informace a je i nápomocen pro zadávání času a data.

2.5 Konfigurace LCD

Jako zobrazovací část mé diplomové práce jsem použil znakový displej, který je schopen zobrazit 24 znaků na dvou řádcích. LCD displej je opatřen řadičem HD44780A00. Informace o tomto řadiči jsou ve zdroji literatury pod odkazem [9]. Uvedený řadič komunikuje s microcontrollerem pomocí čtrnácti vodičů:

- Vss zemní potenciál 0 V
- Vdd napájecí napětí 5 V
- Vo napět určující kontrast displeje 0 ÷ 5 V
- RS přepínání mezi instrukcí (log. „0“) a daty (log. „1“)
- R/W signál čtení (log. „1“) zápis (log. „0“)
- E signál „enable“ slouží jako potvrzení dat či instrukcí
- DB0 – 7 osm datových vodičů

Řadič můžeme konfigurovat pomocí instrukcí popsaných v uživatelském manuálu. Mezi základní možnosti konfigurace patři výběr komunikace po čtyřech, nebo osmi datových vodičích. Komunikace po čtyřech datových vodičích může být užitečná v případě nedostatku pinů microcontrolleru. Naproti tomu je komunikace po osmi datových vodičích komfortnější a také poměrně rychlejší, protože namísto posílání obou polovin datového slova odděleně s nutností každou zvlášť potvrdit signálem „Enable“ je odesíláno celé datové slovo najednou.

Obsluhu řadiče LCD musíme opatřit programově. Při psaní programu pro komunikaci s řadičem LCD lze postupovat dvěma způsoby:

- Můžeme používat signál R/W a ve čtecím módu ověřujeme, zdali je řadič připraven přijmout další instrukci. Platí totiž, že po dobu vykonávání instrukce je řadičem LCD displeje nastaven tzv. „busy flag“ do logické úrovně „1“. Tento indikátor zaneprázdnění můžeme ověřovat v módu čtení, a dokud nedojde k jeho vynulování, nemůže být vyslána další instrukce. Pro využití této možnosti však potřebujeme jeden volná výstupní pin microcontrolleru, kterým budeme nastavovat signál R/W.
- Rozhodneme se ušetřit jeden výstupní pin a signál R/W připojíme na zemní potenciál. Tím trvale umožníme pouze zapisovat. Následně bereme ohled na správné časování komunikace s řadičem LCD, neboť pro každou instrukci je předepsán maximální čas potřebný k jejímu vykonání. Tyto časy můžeme nalézt v uživatelském manuálu konkrétního řadiče LCD.

2.6 Obvod RTC

Důležitou částí mé diplomové práce je udržení přesného času, k čemuž slouží obvod RTC (Real Time Controller). Těchto obvodů je velké množství. Liší se například provedením pouzdra nebo počtem pinů. Pro svou diplomovou práci jsem si vybral obvod DS1307 od firmy MAXIM [6]. Pouzdro RTC obvodu je osmi-pinové PDIP, napájecí napětí je 5 V a teplotní rozsah od 0 °C do 70 °C. Přehled jednotlivých pinů je následující:

- GND zemní potenciál 0 V
- Vcc napájecí napětí typicky 5 V
- V_{BAT} pin pro kladnou svorku záložní baterie typicky 3 V

- X1 a X2 piny pro připojení krystalického oscilátoru
- SDA vstupně výstupní datový pin pro I^2C komunikaci
- SDC vstupní hodinový pin pro I^2C komunikaci
- SQW/OUT pin pro výstupní obdélníkový signál různé frekvence

Připojením záložní baterie s napětím 3 V na pin V_{BAT} předejdeme ztrátě aktuálního času, k níž by mohlo dojít při výpadku napájení.

RTC obvod je připraven pro připojení externího hodinového oscilátoru o typické frekvenci 32,768 kHz. Bez tohoto oscilátoru není obvod schopen správné funkce.

Piny SDA, SCL a SQW/OUT potřebují externí pull-up rezistory (odpor vložený mezi napájecí napětí a konkrétní vodič) pro vytvoření recesivního stavu na sběrnici odpovídajícího logické úrovni „1“.

Informaci o data a času můžeme získat přečtením registrů RTC obvodu a stejně tak můžeme údaje aktualizovat zápisem na správnou pozici v registru. Datum a čas jsou v registrech RTC obvodu uloženy ve formě BCD kódu, z čehož vyplývá, že jsou odděleně zapsány například vteřiny a desítky vteřin. Systém uložení data a času v registrech naznačuje tabulka 4.1, z níž vidíme, že na adrese 07H se nachází konfigurační registr. Ten slouží pro povolení výstupního obdélníkového signálu o frekvenci, kterou nastavujeme pomocí bitů RS1 a RS0. Výstupní frekvence potom může být 1 Hz, 4,096 kHz, 8,1192 kHz nebo 32,768 kHz. V rámci své diplomové práce využívám výstupního signálu 1 Hz, jehož nastavení naznačují hodnoty v závorkách uvedené na konkrétních místech v tabulce 4.1. Jak je vidět tak na bitu OUT nezáleží. Jeho hodnota určuje logickou úroveň signálu v případě, že je bit SQWE vynulován.

Tabulka 2.1 Funkce registrů RTC obvodu [6]

ADRESA	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNKCE
00H	CH	10 vteřin			vteřiny				vteřiny
01H	0	10 minut			minuty				minuty
02H	0	12	10 hodin	10 hodin	hodiny				hodiny
		24	PM/AM						
03H	0	0	0	0	0	den v týdnu			den v týdnu
04H	0	0	10 dní		dny				dny v měsíci
05H	0	0	0	10 měsíců	měsíce				měsíc v roce
06H	10 let				roky				rok
07H	OUT	0	0	SQWE	0	0	RS1	RS0	nastavení
08H - 3FH									RAM 56 x 8

Jak už bylo naznačeno ve výčtu pinů, RTC obvod komunikuje pomocí I^2C komunikačního protokolu. Master (nadřízený) je v tomto případě microcontroller a RTC obvod je sleave (podřízený). Funkce vysílače a přijímače je mezi obvody předávána podle módu komunikace. Módy komunikace jsou tři:

- Zápis dat do RTC
- Čtení dat z RTC
- Čtení od určité adresy

V každém případě začíná komunikaci master tím, že na sběrnici vytvoří startovací podmínku, po níž odvysílá sedmibitovou adresu RTC (identifikátor) obvodu. Za adresou je připojen bit R/W, který informuje RTC obvod, zdali je má přejít do stavu vysílání nebo má očekávat příjem dalšího bajtu.

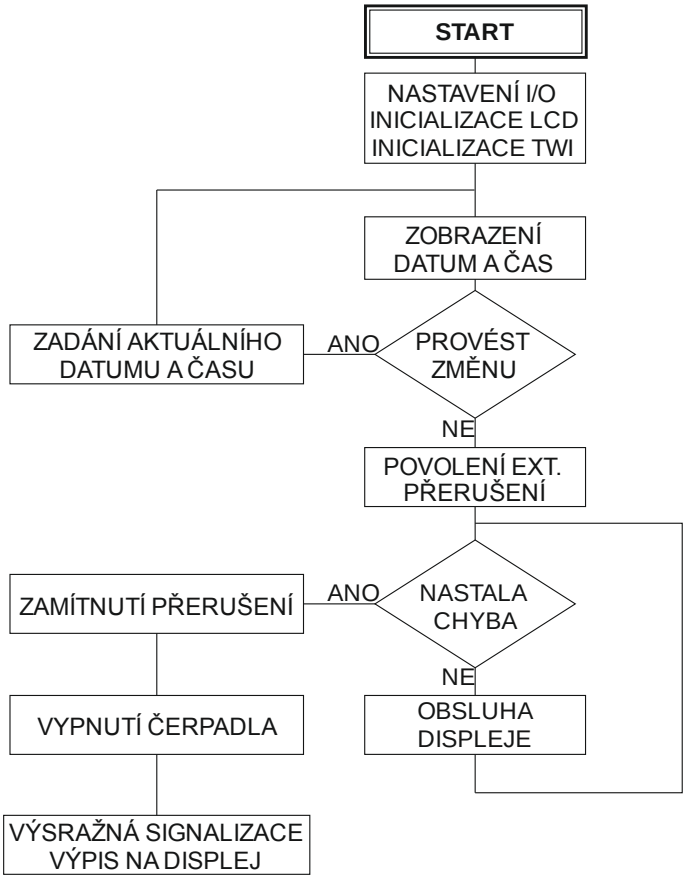
Pokud je bit R/W roven nule, RTC obvod bude očekávat příjem adresy registru, který bude přepisovat daty, přijatými od microcontrolleru v dalším osmibitovém bloku. Mezi jednotlivými bloky změní slave recesivní stav na vodiči SDA, vytvořený obvodem master, na dominantní, čímž potvrdí příjem předešlého bloku. Pro ukončení vysílání vytvoří master na SDA a SDC stop podmínku.

Pokud je R/W bit roven jedné, slave potvrdí příjem zprávy a následně začne vysílat data počínaje adresou 00H. Master potvrzuje každý osmibitový blok zvlášť stejným způsobem,

jako v předešlém případě potvrzoval slave. Pro ukončení čtení registrů RTC nejprve master nepotvrdí poslední přijatou zprávu a po uplynutí doby pro potvrzení vytvoří na vodičích SDA a SDC stop podmínku.

Čtení od určité adresy je speciálním případem čtení. V prvním datovém bloku je opět odeslána adresa slave (sedmibitový identifikátor) následovanou R/W bitem, který je roven nule. V dalším datovém bloku je adresa registru RTC, kterou má slave odvysílat jako první. Po potvrzení tohoto datového bloku vytvoří master na sběrnici stav, kterému se říká opakovaný start. Za touto podmínkou znovu odvysílá sedmibitový identifikátor následovaný R/W bitem o logické hodnotě 1. Následně převezme vysílání slave a začne vysílat obsahy registrů po sobě jdoucích počínaje registrem o předvolené adrese. [6]

2.7 Vývojový diagram funkce main



Obrázek 2.1 Vývojový diagram funkce main

Funkce main je základní funkcí každého programu v jazyce „C“. Při psaní programu pro microcontrollery obecně platí, že funkce main je nekonečná, respektive alespoň její část je prováděna cyklicky pro zaručení stálého chodu.

Ve mnou napsaném programu je cyklicky prováděna pouze část, ke které dojde po nastavení správného data a času v obvodu RTC. V první řadě však dojde k nastavení vstupů a výstupů jednotlivých bran potažmo pinů microcontrolleru. Následně dojde ke konfiguraci LCD řadiče instrukcemi pro nastavení zadávání, chování kurzoru a nakonec dojde k zapnutí a vymazání LCD.

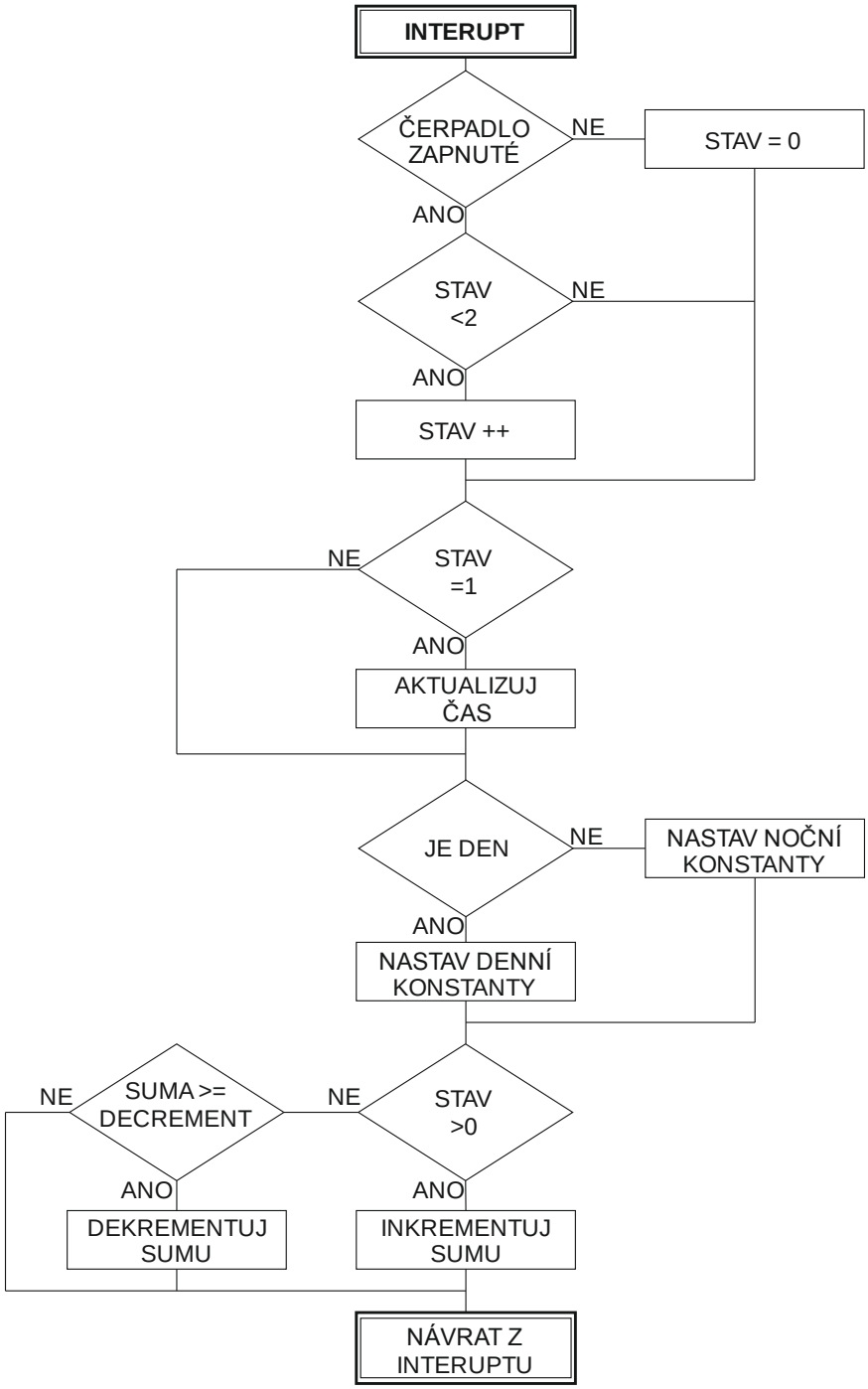
Po konfiguraci LCD dojde k přednastavení registrů TWI. TWI znamená Two Wire Interface. Tento samostatný modul microcontrolleru ATmega16 je uzpůsobený pro komunikační protokol I^2C a v mé diplomové práci je využit pro komunikaci s obvodem RTC. Nakonec dojde k zápisu do registru RTC obvodu s adresou 07H pro nastavení a povolení výstupního obdélníkového signálu o frekvenci 1 Hz.

V tuto chvíli je vše připraveno pro přečtení registrů RTC obvodu za účelem získání aktuálního data a času a jejich následnému vypsání na displej LCD. Nyní je zapotřebí zásahu uživatele, který se rozhodne zobrazený datum a čas buď akceptovat a potvrdí jeho správnost tlačítkem, nebo se jej rozhodne opravit manuálně. V tom případě stiskneme druhé tlačítko, které vyvolá zadávací mód. Na displeji se zobrazí návod, v jakém formátu má uživatel zadat požadovaný datum a čas. Po dokončení manuálního nastavení dojde k zobrazení uživatelem nastavených hodnot a uživatel se může opět rozhodnout, zdali již bude nastavení akceptovat, nebo chce svou volbu změnit.

Po potvrzení nastaveného data a času dojde k povolení přerušení od externího zdroje a program přejde do nekonečné smyčky, ve které se provádí rozhodování, zdali nedošlo k poruchovému stavu a zároveň je každou vteřinu proveden výpis na displej pro aktualizování jeho obsahu.

Pokud nastanou podmínky, které systém vyhodnotí jako poruchu, program zamítne přerušení a neprodleně vystaví signál k přerušení chodu čerpadla. Následně je oznámen poruchový stav blikáním LED diody a na displej je vypsána pravděpodobná příčina poruchy. Z tohoto stavu se lze navrátit pouze zásahem obsluhy, která tlačítkem potvrdí, že došlo k nápravě problému.

2.8 Vývojový diagram obsluhy externího přerušení



Obrázek 2.2 Vývojový diagram obsluhy externího přerušení

Přerušení (angl.. interrupt) programu je přístup, který umožní synchronně reagovat na asynchronní událost. Obecně existuje více událostí, na něž může microcontroller reagovat přerušením. Mezi ně patří dokončení převodu A/D převodníku, dokončení příjmu či vyslání znaku sériovou jednotkou nebo změna logické úrovně na vstupní bráně microcontroleru. Po výskytu události je vyvolán požadavek na přerušení IRQ (angl.. interrupt request), které jsou

uchovávány v konkrétním registru dokud nedojde k jejich zpracování. Jednotlivé požadavky na přerušení mohou být blokovány tzv. maskováním. K maskování slouží další registr, kde jednotlivé bity povolují nebo blokují svou úrovní konkrétní požadavky. Maskování je provedeno logickým součinem registru požadavků s registrem masky. Požadavky na přerušení dále zpracovává řadič přerušení, jehož úkolem je rozpoznat aktivní požadavky, rozhodnout o prioritě jejich vykonávání (v případě více požadavků) a následně musí upozornit procesor na výskyt požadavku na přerušení. Výstup řadiče přerušení určený pro informování procesoru o aktivním požadavku může být také blokován čímž je zamítnuto jakékoli maskovatelné přerušení. Kromě již popsaných maskovatelných přerušení existují i přerušení, která jsou přivedena přímo do procesoru, aniž by byla zaregistrována řadičem přerušení. Taková přerušení se nazývají nemaskovatelná, mají nejvyšší prioritu a reakce na ně bývá podstatně rychlejší, neboť nejsou zatížena zpožděním logiky řadiče. Využívají se pro havarijní případy například pro zablokování programu.

V situaci, kdy dojde k vystavení požadavku na přerušení, který není maskován, upozorní, jak už bylo popsáno, řadič přerušení procesor v případě, že je povoleno globální přerušení. Procesor nejprve dokončí aktuálně prováděnou instrukci (pokud se nejedná o dlouhou instrukci, která může být dokončena po návratu z přerušení). Následně je do zásobníku uložena návratová adresa a hodnoty důležitých registrů. Poté procesor zahájí komunikaci s řadičem za účelem získání typu přerušení, kterému je přiřazena adresa obslužného programu. Dojde k vyvolání podprogramu spolu se zamítnutím globálního přerušení (díky tomu nemůže dojít ke vnořování přerušení). Po skončení obslužného programu přerušení dojde k obnovení obsahu uložených registrů, přesunu návratové hodnoty do čítače instrukcí a povolení globálního přerušení. [7]

V případě mé diplomové práce se jedná o externí přerušení signálem o periodě jedna vteřina. To znamená, že běh programu je zastaven a microcontroller začne vykonávat obsluhu externího přerušení v závislosti na přednastaveném průběhu na externím vodiči, který je připojený ke konkrétnímu pinu microcontroleru.

Na začátku přerušení dojde k rozpoznání, zdali je čerpadlo ve vypnutém nebo zapnutém stavu. Proměnná STAV je zde pouze pro detekci přechodu čerpadla z vypnutého do zapnutého stavu. Její hodnota se pohybuje v rozmezí nula až dva. Pokud je proměnná stav rovna nule, znamená to, že čerpadlo je vypnuté. V případě že dojde k zapnutí čerpadla,

proměnná STAV je inkrementována a v tom případě je její hodnota rovna jedné. Pokud čerpadlo setrvává v zapnutém stavu, zatímco dojde k novému přerušení, proměnná STV je znovu inkrementována za předpokladu, že její hodnota ještě není rovna dvěma. Z toho vyplývá, že proměnná STAV je rovna jedné jen v přerušeních následujících bezprostředně po zapnutí čerpadla. Kromě určení momentu, kdy došlo ke změně stavu čerpadla, je také proměnná STAV využita po dobu vykonávání přerušení pro určení, zdali se čerpadlo nachází ve stavu vypnutém nebo zapnutém. Toho je docíleno jednoduchým porovnáním na nenulovost.

V momentě, kdy dojde k rozpoznání zapnutí čerpadla (STAV roven jedné) je zahájena komunikace s obvodem RTC a dojde k přečtení aktuálního času. Na základě času je potom rozhodnuto, zdali se budou inkrementovat a dekrementovat denní či noční konstanty. Čas je v microcontrolleru uložen ve formě několika proměnných, z nichž jedna se jmenuje HOURS. Tato proměnná je porovnávána na velikost což odpovídá rozhodování, zdali je v momentě zapnutí čerpadla den nebo noc. Jelikož k rozhodování zdali je den nebo noc dochází pouze v momentě rozpoznání zapnutí čerpadla, zůstanou konstanty pro inkrementování a dekrementování neměnné nejméně do následujícího stavu zapnutí čerpadla.

V poslední části obsluhy přerušení dojde k rozhodnutí, zdali je čerpadlo zapnuté nebo vypnuté. Pokud je čerpadlo v zapnutém stavu, potom dojde k navýšení aktuální hodnoty proměnné SUMA o velikost inkrementační konstanty. V případě že se čerpadlo nachází ve vypnutém stavu, dojde ke snížení hodnoty proměnné SUMA o velikost dekrementační konstanty ovšem za předpokladu, že je velikost proměnné suma větší než velikost dekrementační konstanty. Kdyby nedošlo k porovnání velikostí proměnné SUMA a dekrementační konstanty, mohlo by za předpokladu, že by byla velikost proměnné SUMA menší než velikost dekrementační konstanty dojít k tzv. „podtečení“ hodnoty proměnné SUMA. Jelikož je proměnná SUMA typu bezznaménkový integer, v případě, že by byla proměnná SUMA rovna nule a došlo by k jejímu dekrementování o hodnotu jedna, byla by po provedení této operace velikost proměnné SUMA rovna 65535.

Po provedení inkrementace nebo dekrementace dojde k ukončení přerušení a k návratu do programu na místo, ve kterém došlo k přerušení běhu programu.

2.9 Popis funkce

Navrhovaný systém má kromě zadávacích tlačítek dva vstupy určené pro monitorování čerpadla pro domácí vodárny.

Prvním vstupem je konektor k měření proudu. Na tento konektor je přivedeno napájení pro obvod snímající proud přívodním vodičem čerpadla. Výstup čidla je přiveden na pin PA7 microcontrolleru ATmega16, který může být analogově číslicovým převodníkem. Podle potřeby si tedy můžeme zvolit, zda bude mít vstupní hodnota logický, anebo analogový charakter. Ve své práci se věnuji reakci na vstupní logickou hodnotu. V případě rozšíření monitorovacího systému na monitorování analogové hodnoty, musí být vstupní analogový signál, od modulu měření proudu, v rozmezí $0 \div 5$ V

Druhým vstupem je potom konektor pro externí vypnutí čerpadla. Na jednu svorku konektoru je přiveden zemní potenciál, Na zbylé dvě svorky jsou přivedeny vstupy microcontrolleru, které jsou přes pull-up rezistory připojené na napájecí napětí. Na tyto vstupy mohu být připojeny přídavné monitorovací systémy, které v případě rozpoznání poruchy uzemní vstup microcontrolleru. Programovým opatřením následně můžeme reagovat na poruchu detekovanou přídavným monitorovacím systémem.

Výstupem navrhovaného systému je kromě displeje a led diod také konektor, na který jsou přivedeny dva piny microcontrolleru. Na těchto pinech microcontrolleru jsou vytvořeny komplementární logické úrovně. V případě poruchy dojde k záměně logických úrovní. Konektor je navíc opatřen napájecím napětím a zemním potenciálem. Vodiče tohoto konektoru mohou být podle potřeby připojeny na optočlen, jímž bude ovládáno relé. Takto upraveným výstupním signálem může být následně proveden patřičný akční zásah.

2.10 Akční zásahy

Akčním zásahem je fyzickou reakcí na detekci chybového stavu. Pokud dojde k výskytu poruchy, systém musí být schopen tuto poruchu detekovat. Systém však musí být opatřen navíc ještě výstupem, kterým může provést adekvátní zásah. Adekvátní zásah je takový zásah, který zabrání poškození čerpadla potažmo nemovitosti, nebo zmírní následky případného

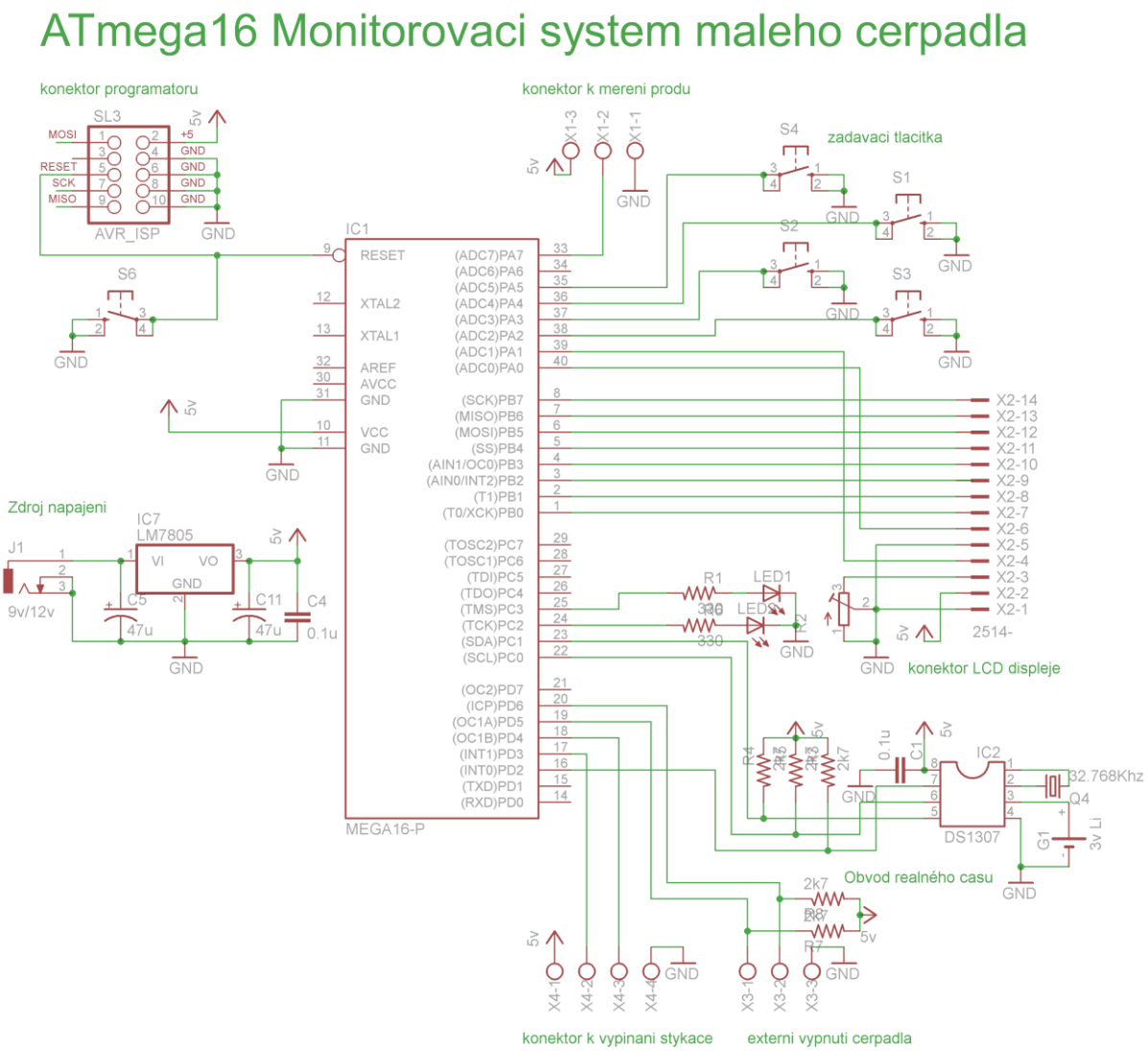
poškození. V tomto případě je adekvátním zásahem odpojení čerpadla od elektrické sítě. Existuje více možností, jak provést odpojení čerpadla od elektrické sítě.

Domácí vodárna bývá vybavena tlakovým spínačem. Tento spínač je ovládán tlakem uvnitř vodní nádrže. V některých provedeních ovládá tlakový spínač svým elektrickým výstupem stykač v přívodu čerpadla. V případě popsaného provedení může být upravenými výstupy microcontrolleru přerušován elektrický signál od tlakového spínače. Výsledkem je nepřímé odpojení čerpadla od zdroje elektrické energie.

Ve většině případů je elektrické napájení čerpadla ovládáno přímo tlakovým spínačem. V takovém případě může být do série s tlakovým spínačem zapojen přídavný stykač s cívkou, který bude ovládán elektricky upravenými vývody microcontrolleru. Domácí vodárnu potom můžeme odpojit od elektrické sítě přímo stykačem.

Poslední možností je sestrojení polovodičového spínače, kterým bychom blokovali přívod elektrické energie. Tato možnost je náhradou mechanického stykače. Výsledné řešení je elegantní, poměrně univerzální ale nákladné.

2.11 Schéma zapojení

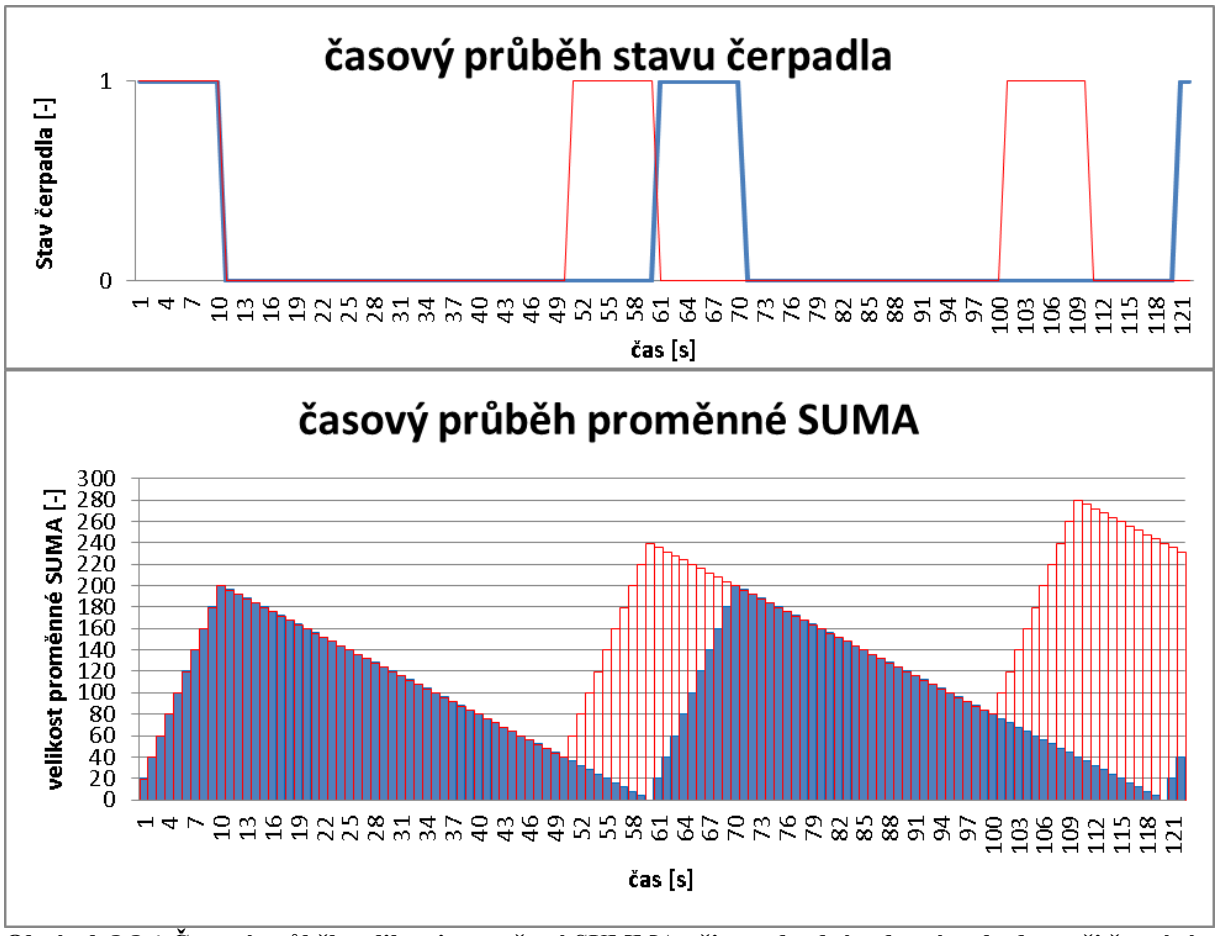


Obrázek 2.3 Schéma zapojení monitorovacího systému [3], [2], [8]

3 Ověřování funkčnosti

O funkčnosti navrženého systému implementujícího integrační metodu můžeme rozhodnout na základě testování. Ověření bylo provedeno několika teoretickými průběhy, simulujícími reálný chod čerpadla a případné poruchové stavy.

3.1 Reakce na zvětšení odběru vody



Obrázek 3.3.1 Časový průběh velikosti proměnné SUMMA při standardním denním chodu a při čerpání velkého množství vody

Na obrázku 3.1 vidíme v horní části časový průběh spínání čerpadla. Tento průběh je pouze teoretický a má sloužit k časově nenáročné kontrole funkčnosti navrženého systému. V dolní části je graf znázorňující velikost proměnné SUMA v závislosti na časovém průběhu spínání čerpadla. Graf v horní části obrázku 3.1 znázorňuje dva průběhy.

Modrý průběh pro tento případ znázorňuje standardní spínání čerpadla. Jako standardní průběh spínání čerpadla je považován průběh, kdy na deset vteřin trvající dobu čerpání

připadá padesáti vteřinová doba nečinnosti. Během doby nečinnosti je spotřebována voda z vodní nádrže a následně dojde k novému čerpání. Tento průběh se neustále opakuje, a protože v příkladu představuje standardní spínání čerpadla, neměl by být vyhodnocen jako chyba.

U červeného průběhu je doba nečinnosti zkrácena o deset vteřin. V praxi by to znamenalo, že je voda spotřebovávána rychleji, než bylo předpokládáno. Tento průběh má simulovat poruchový stav, který odpovídá prasklině ve vodovodním potrubí. Za předpokladu, že se tento stav nemění a průběh se cyklicky opakuje, by měla být chyba rozpoznána přibližně při třetím opakování.

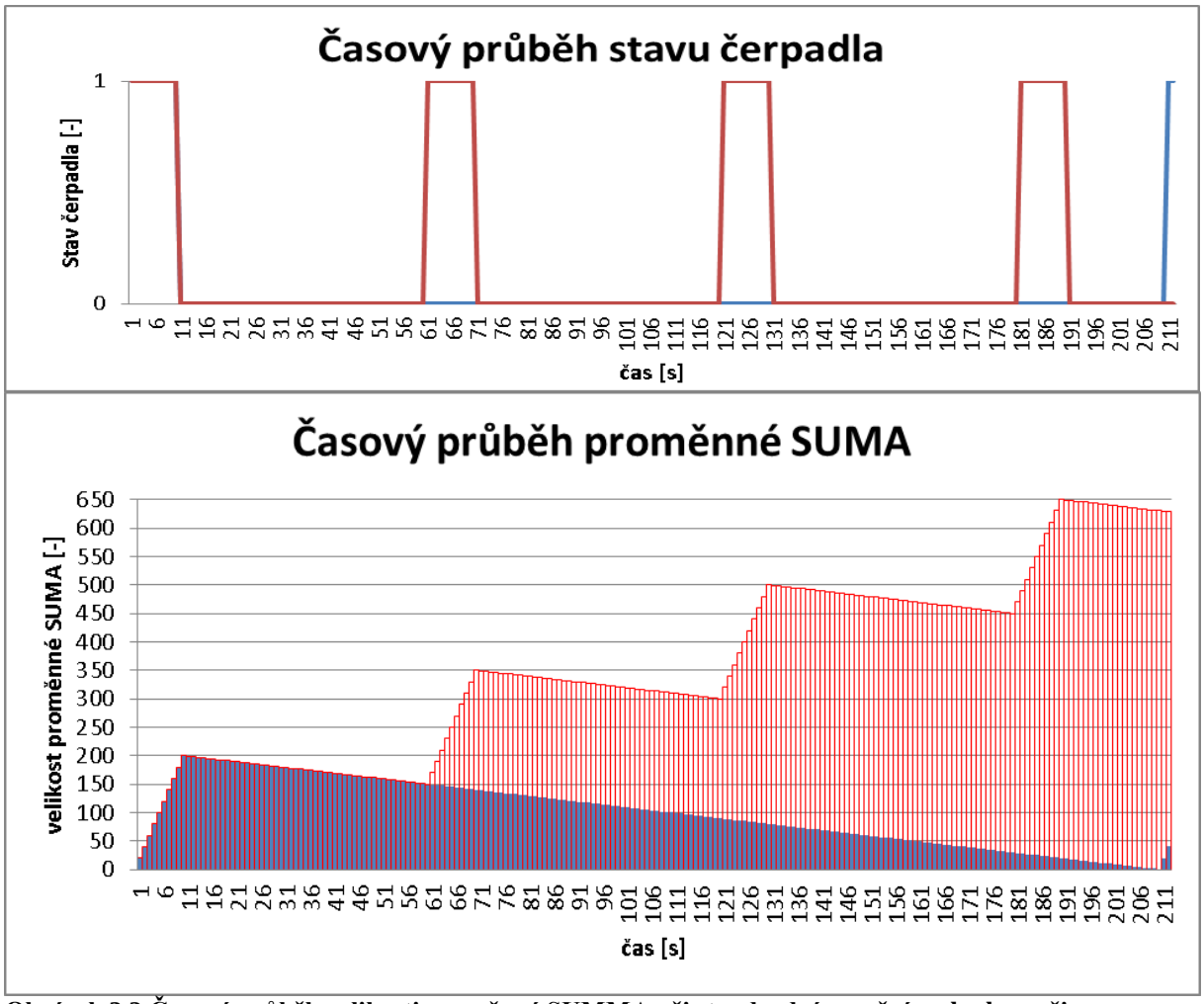
V dolní části obrázku 3.1 je druhý graf, který znázorňuje přímou implementaci integrační metody. V grafu jsou zaneseny dva průběhy odpovídající časovým průběhům spínání čerpadla z horního grafu. Oba průběhy mají společnou inkrementační a dekrementační konstantu. Inkrementační konstanta je nastavena na hodnotu 20 a dekrementace je prováděna hodnotou o velikosti 4.

Nejprve se zaměříme na modrý průběh. Jak už bylo naznačeno, doba nečinnosti je pětkrát delší, než je doba sepnutí čerpadla. Také bylo popsáno, že integrační konstanta je pětkrát větší než dekrementační konstanta. Do proměnné SUMA je během čerpání naintegrována hodnota 200 (10 vteřin krát inkrement 20). Za dobu nečinnosti však hodnota proměnné SUMA klesne na nulu a to jen proto, že je doba nečinnosti dlouhá alespoň padesát vteřin. Padesát vteřin je v tomto případě minimální doba nečinnosti pro vynulování proměnné SUMA. Tento průběh se může opakovat po nekonečně dlouhou dobu, aniž by hodnota proměnné SUMA překročila hodnotu 200. Při modrém průběhu nebude detekována chyba, pokud zachováme zmíněné velikosti integrační a dekrementační konstanty a hodnota, se kterou bude SUMA porovnávána, bude větší než 200.

Červený průběh má v tomto případě simulovat poruchu a při jeho cyklickém opakování má dojít k detekci poruchového stavu. Až do padesáté vteřiny je červený průběh identický s modrým. V padesáté první vteřině dojde ke změně. Zatímco pro modrý průběh je čerpadlo stále ve stavu nečinnosti, pro červený průběh dojde k sepnutí čerpadla. V padesáté vteřině byla velikost SUMY rovna čtyřiceti. Po deseti vteřinách čerpání se velikost SUMY zvětší o 200 a bude se tak rovnat hodnotě 240. Následující doba nečinnosti je opět po čtyřiceti

vteřinách přerušena sepnutím čerpadla. Před zapnutím čerpadla je hodnota proměnné SUMA rovna osmdesáti, protože doba nečinnosti byla opět o deset vteřin kratší než u standardního průběhu. Po dalším desetivteřinovém čerpání se zvýší hodnota proměnné SUMA na 280. V tomto bodě by mělo již dojít k detekování chyby a odpovídajícímu zásahu. Aby tedy byla v tomto bodě skutečně detekována chyba, nastavíme hodnotu konstanty, s níž budeme komparovat SUMU, na hodnotu 279. K detekci chyby tedy dojde po uplynutí sto deseti vteřin od počátku (na konci třetího cyklu čerpání), kdy velikost SUMY nabyde hodnoty větší, než je velikost komparační konstanty.

3.2 Vliv denní a noční doby



Obrázek 3.2 Časový průběh velikosti proměnné SUMMA při standardním nočním chodu a při standardním denním chodu v nočních hodinách

Na obrázku 3.2 vidíme opět dva grafy. Horní graf opět představuje dva průběhy spínání. Na těchto průbězích je simulováno spínání čerpadla v nočních hodinách. Dá se předpokládat, že v nočních hodinách dochází k výrazně menším spotřebám vody než v denních hodinách.

Intervaly mezi sepnutími čerpadla představují rychlost spotřeby vody. Pokud je spotřebováváno mnoho vody, musí čerpadlo spínat častěji. Obrázek 3.2 znázorňuje, jak se bude vyvíjet proměnná suma, když dojde v nočních hodinách ke stejnému využívání vody jako přes den. Horní graf na obrázku 3.2 přestavuje akceptovatelný noční průběh spínání a akceptovatelný denní průběh spínání (v nočních hodinách neakceptovatelný).

Modrý průběh v tomto případě představuje akceptovatelné noční spínání čerpadla. Akceptovatelné znamená, že zatímco doba čerpání je stejně dlouhá jako přes den, doba nečinnosti je výrazně delší. Pro tento příklad musí být doba nečinnosti minimálně čtyřikrát delší než přes den, aby nedošlo k vyhodnocení chyby. Na deset vteřin čerpání tedy připadá 200 vteřin nečinnosti. Stav čerpadla je v grafu prvních deset vteřin v jedničce což představuje zapnuté čerpadlo. Po deseti vteřinách přechází čerpadlo do stavu nečinnosti (reprezentovaným stavem 0). K následujícímu čerpání dojde v čase 201 vteřin. Opět se jedná o teoretický průběh.

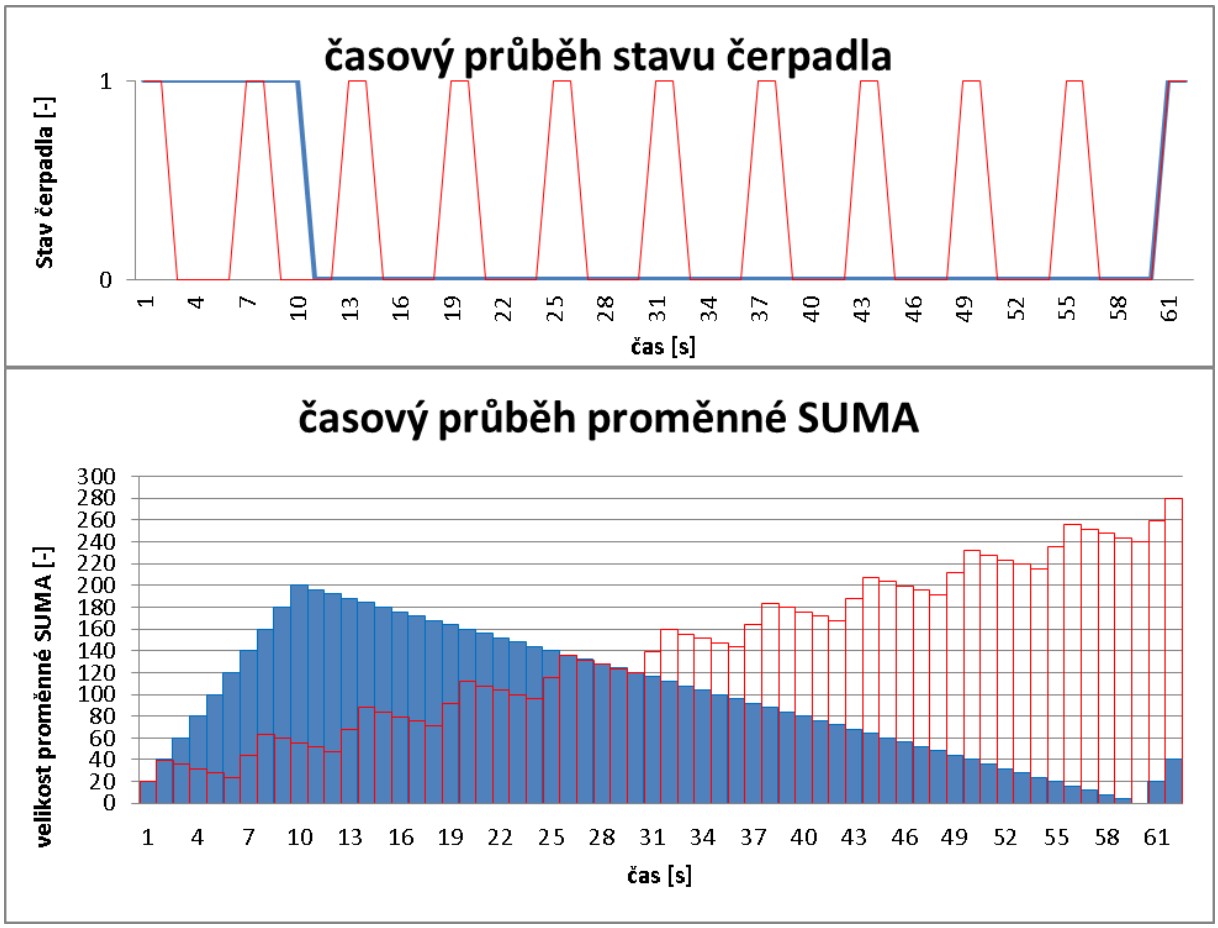
Červený průběh je akceptovatelným spínáním z předchozího případu na obrázku 3.1. Na deset vteřin trvající čerpání připadá padesát vteřin nečinnosti. Z grafu je patrné, že každých 60 vteřin přejde čerpadlo ze stavu nečinnosti do stavu čerpání. V denních hodinách tento průběh představuje neporuchový chod. V nočních hodinách je spotřeba vody reprezentována tímto průběhem neakceptovatelná. Systém při tomto průběhu detekuje chybu.

V dolním grafu jsou opět zaneseny dva průběhy velikosti proměnné SUMA v závislosti na spínání čerpadla. Proměnná SUMA je stejně jako v denních hodinách inkrementována a dekrementována příslušnými konstantami. Pro noční hodiny dochází ke změně dekrementační konstanty oproti dennímu provozu. Inkrementační konstanta představuje analogii s dobou čerpání (doba, za kterou čerpadlo doplní vodu ve vodní nádrži). Na dobu čerpání nemá vliv změna denní doby na noční. Inkrementační konstanta si tedy zachovává stejnou velikost jako pro denní čas a to 20. Pro tento experiment bylo odhadnuto, že doba nečinnosti se v noci prodlouží na čtyřnásobek denní doby nečinnosti čerpadla. Tomu odpovídá změna dekrementační konstanty. Ta se pro noční hodiny změní ze čtyř na jedna, neboli na čtvrtinu. Nárůst a pokles SUMY vlivem inkrementování a dekrementování pro oba průběhy je patrný ze spodního grafu na obrázku 3.2

Modrý průběh představuje nárůst a pokles proměnné SUMA pro akceptovatelný průběh. Během prvních deseti vteřin se každou vteřinu navýší hodnota proměnné SUMA o 10 na konečnou hodnotu 200. V jedenácté vteřině dojde k vypnutí čerpadla a SUMA je každou vteřinu dekrementována o hodnotu 1. Během doby nečinnosti, která trvá 200 vteřin, je hodnota proměnné SUMA postupně dekrementována až do nuly. Pokud by čerpadlo nadále setrvávalo se stavu nečinnosti, hodnota proměnné SUMA by byla stále nulová. V tomto teoretickém průběhu však dojde ve dvě stě jedenácté vteřině k novému čerpání. V předchozím teoretickém průběhu znázorněném na obrázku 3.1 byla určena komparační mez na hodnotu 279. Pokud by hodnota proměnné suma překročila komparační mez, došlo by k detekci chyby. Z grafu je patrné, že modrý akceptovatelný průběh nepřekročí komparační mez, a proto systém nezareaguje vypnutím čerpadla.

Červený průběh znázorňuje na obrázku 3.1 neakceptovatelný průběh. Pro tento průběh platí stejná inkrementační a dekrementační konstanta jako pro modrý průběh. Tento teoretický průběh má prověřit reakci systému v nočních hodinách. Při nočních hodinách má systém definovaným způsobem změnit inkrementační a dekrementační konstantu na přednastavené hodnoty určené pro noční chod. Systém nebude schopen detekovat chybu, představovanou červeným průběhem, pokud nedojde ke změně denních konstant na noční. Červený průběh je akceptovatelný pouze v denních hodinách. Pokud se červený průběh vyskytne v nočních hodinách, musí systém detekovat chybu. Červený průběh je z počátku identický jako modrý průběh. V prvních deseti vteřinách dojde k nárůstu hodnoty proměnné SUMA z nuly na hodnotu 200. V jedenácté vteřině přejde čerpadlo do stavu nečinnosti a SUMA je každou vteřinu dekrementována dekrementační konstantou. Za padesát vteřin, kdy je čerpadlo ve stavu nečinnosti, se hodnota proměnné SUMA zmenší z hodnoty 200 na 150. V šedesáté první vteřině dojde k novému sepnutí čerpadla a SUMA je každou vteřinu inkrementována. V šedesáté druhé vteřině překročí hodnota proměnné SUMA komparační úroveň pro detekci chyby. Hodnota SUMY se totiž za dvě vteřiny, kdy je čerpadlo ve stavu čerpání, zvětší z hodnoty 150 na 190. Komparační úroveň je stále nastavena na 179. Pokud tedy budeme simulovat spínání čerpadla podle červeného průběhu, musí dojít v šedesáté druhé vteřině k detekci chyby.

3.3 Vliv častého spínání na krátký časový úsek



Obrázek 3.3 Časový průběh velikosti proměnné SUMMA při standardním denním chodu a při častém spínání na krátký časový úsek

Teoretický průběh znázorněný na obrázku 3.3 červenou barvou má simulovat spínání čerpadla v závislosti na nízkém objemu vzduchu uvnitř tlakové vodní nádrže. Oba znázorněné průběhy jsou simulovány v denních hodinách. Systém je však schopen na tento průběh zasáhnout účinně i v nočních hodinách.

Modrý průběh je stejně jako na obrázku 3.1 akceptovatelným denním průběhem a jedná se o naprosto totožný průběh. Z průběhu je patrné, že čerpadlo je prvních deset vteřin ve stavu 1 což interpretuje zapnutý stav neboli čerpání. V jedenácté vteřině dojde k vypnutí čerpadla, které zůstává následujících padesát vteřin ve stavu nečinnosti, což je v grafu znázorněno hodnotou nula. Po uplynutí doby nečinnosti se průběh opakuje.

Červený průběh je teoretickou interpretací neakceptovatelného chodu čerpadla založeném na malém objemu vzduchu uvnitř tlakové vodní nádrže. Se snižujícím se objemem vzduchu

uvnitř nádrže dohází ke zkracování doby čerpání. Tento jev je zapříčiněn rozdílem stlačitelností vody a vzduchu. Zatímco vzduch je velmi stlačitelný, voda se chová téměř jako ideální kapalina a tudíž je téměř nestlačitelná. Pokud by ve vodní nádrži nebyl přítomen žádný vzduch, stačilo by nepatrné množství vody k vyvolání velké změny tlaku. Čím menší je tedy objem vzduchu uvnitř tlakové vodní nádrže, tím méně vody je zapotřebí k překonání tlakové hystereze. Tlaková hystereze je tvořena tlakem, při kterém dojde k zapnutí čerpadla a tlakem, při kterém dojde k vypnutí čerpadla. Protože objem vody odpovídající tlakové hysterezi je malý, tak při odebrání malého množství vody dosáhne tlak uvnitř vodní nádrže minima a dojde k opětovnému zapnutí čerpadla. Červený průběh byl vytvořen pro simulaci spínání čerpadla při nedostatečném objemu vzduchu uvnitř tlakové vodní nádrže. Doba čerpání je tedy zkrácena na dvě vteřiny a doba nečinnosti je zkrácena na čtyři vteřiny.

V grafu ve spodní části obrázku 3.3 jsou opět zaznamenány nárůsty a poklesy proměnné SUMA pro jednotlivé průběhy spínání čerpadla z horního grafu. Protože simulace byla prováděna v denních hodinách, byly uvažovány denní konstanty. Inkrementační konstanta je tedy rovna dvaceti a dekrementační konstanta je rovna čtyřem.

Modrý průběh je v rámci denních hodin akceptovatelný. Na začátku je po dobu deseti vteřin inkrementována proměnná SUMA na výslednou hodnotu 200. Následně přejde čerpadlo do stavu nečinnosti. Systém v závislosti na stavu nečinnosti dekrementuje každou vteřinu proměnnou SUMA o hodnotu 4. Během padesáti vteřin, kdy je čerpadlo vypnuté dojde k postupnému vynulování proměnné SUMA. Pokud by čerpadlo setrvávalo nadále ve stavu nečinnosti, proměnná SUMA by zůstala konstantně nulová. V šedesáté první vteřině dojde k započetí nového čerpání a tudíž je SUMA opět inkrementována.

Červený průběh proměnné SUMA je reakcí na neakceptovatelné spínání čerpadla vlivem malého objemu vzduchu uvnitř vodní nádrže. Při detekci zapnutého čerpadla tedy systém inkrementuje SUMU. Během dvouvteřinového čerpání dojde vždy k nárůstu SUMY o hodnotu 20. Mezi dvěma stavy čerpání je vždy čtyřvteřinová doba nečinnosti, při níž dochází k dekrementaci. Během každé doby nečinnosti trvající čtyři vteřiny se sníží hodnota proměnné SUMA o hodnotu 16. V průměru tedy dohází k nárůstu hodnoty proměnné SUMA. Pokud je čerpadlo delší dobu (pro tento případ déle než 62 vteřin) spínáno podle červeného průběhu, vyroste proměnná SUMA až na hodnotu 180. V předchozích případech jsme určili komparační mez, při které dojde k detekci chyby. Tato komparační mez má hodnotu 179.

Z grafu je patrné, že hodnota proměnné SUMA překročí komparační mez právě v čase 62 vteřin. Jakmile systém detekuje překročení komparační meze, vyhodnotí situaci jako chybu a provede patřičný zásah.

Systém navíc hlídá minimální délku zapnutí čerpadla. Pokud by tedy došlo k příliš krátkému sepnutí čerpadla, systém vyhodnotí okamžitě stav jako chybu.

Závěr

V první kapitole mé diplomové práce byly přiblíženy domácí vodárny a havarijní stavy, které při jejich provozu hrozí. Následně byly popsány systémy schopné monitorovat chod domácích vodáren a jejich schopnosti detekovat poruchové stavy.

V druhé kapitole mé práce je popsána integrační metoda, při které je inkrementována a dekrementována proměnná SUMA. Velikost proměnné je následně komparována s konstantou, rozhodující o poruchovosti stavu. Integrační metoda byla dále rozšířena o monitorování zapnutí čerpadla na příliš dlouho nebo příliš krátkou dobu. Současně byla nastíněna implementace popsané metody v systému s microcontrollerem ATmega16 a byl vytvořen funkční vzorek pro testování.

Implementovaná integrační metoda byla nakonec úspěšně ověřena na funkčním vzorku pomocí průběhů, simulujících zvýšený odběr vody a zmenšení objemu vzduchu uvnitř tlakové vodní nádrže. Jedním průběhem bylo ověřeno, že funkční vzorek reaguje správně v denním i nočním režimu. Zapínání a vypínání čerpadla bylo simulováno přepínačem.

Pro použití monitorovacího systému společně s konkrétní domácí vodárnou musí být pozměněny inkrementační a dekrementační konstanty a konstanta pro komparaci. Změna zmíněných konstant je prováděna opětovným nahráním softwaru, čímž je znemožněn neúmyslný zásah do funkce systému.

Seznam literatury a informačních zdrojů

[1] Atmel, ATmega16. [online]. [cit. 2013-05-06]. Dostupné z: http://www.atmel.com/Images/doc2466.pdf

[2] DharmaniTech. *Design with Microcontrollers* [online]. 2009 [cit. 2013-05-06]. Dostupné z: http://www.dharmanitech.com/2009/01/sd-card-interfacing-with-atmega8-fat32.html

[3] *Elektro bastlírna* [online]. 2005 [cit. 2013-05-06]. Dostupné z: http://www.ebastlirna.cz/

[4] Jak funguje domácí vodárna. *Domácí vodárna* [online]. [cit. 2013-05-06]. Dostupné z: http://www.domacivodarna.info/tags/jak-funguje-domaci-vodarna/

[5] Jak funguje domácí vodárna. *Domácí vodárna* [online]. neuvedeno [cit. 2013-05-06]. Dostupné z: http://vodarna-domaci.cz/jak-to-funguje/

[6] Maxim integrated, DS1307, [online]. [cit. 2013-05-06]. Dostupné z: http://datasheets.maximintegrated.com/en/ds/DS1307.pdf

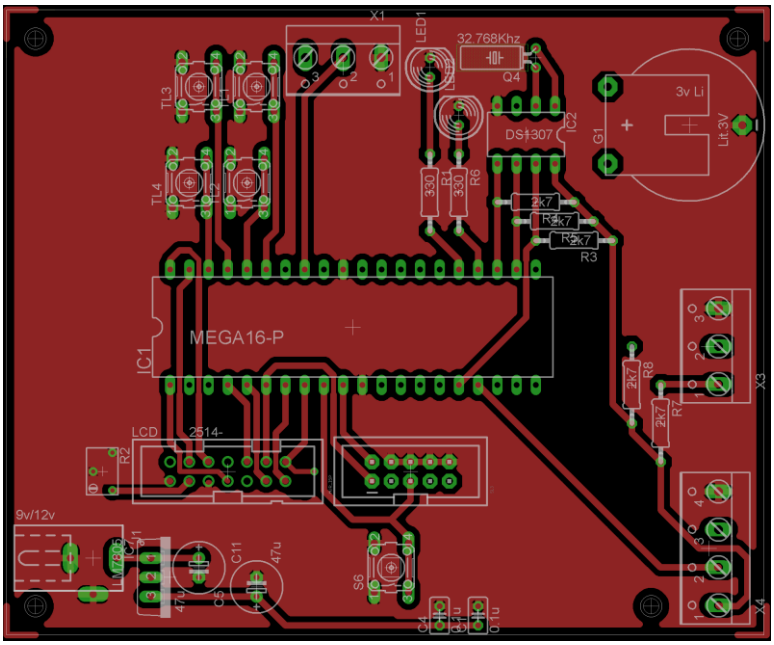
[7] PINKER, Jiří. *Mikroprocesory a mikropočítače*. 1. vyd. Praha: BEN - technická literatura, 2004, 159 s. ISBN 80-730-0110-1.

[8] PINKER, Jiří a Václav KOUCKÝ. *Analogové elektronické systémy*. 2. vyd. Plzeň: Vydavatelství Západočeské univerzity, 1999, 2 sv. ISBN 80-7082-506-52.

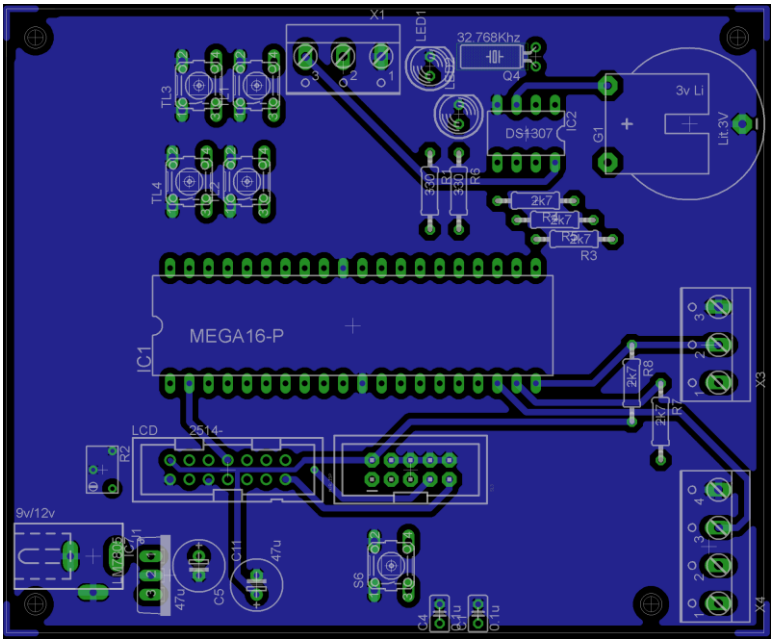
[9] Utsource, HD44780A00, [online]. [cit. 2013-05-06]. Dostupné z: http://www.utsource.net/pdf/pdf-HD44750A.html

Přílohy

Příloha A – Deska plošného spoje navrženého systému

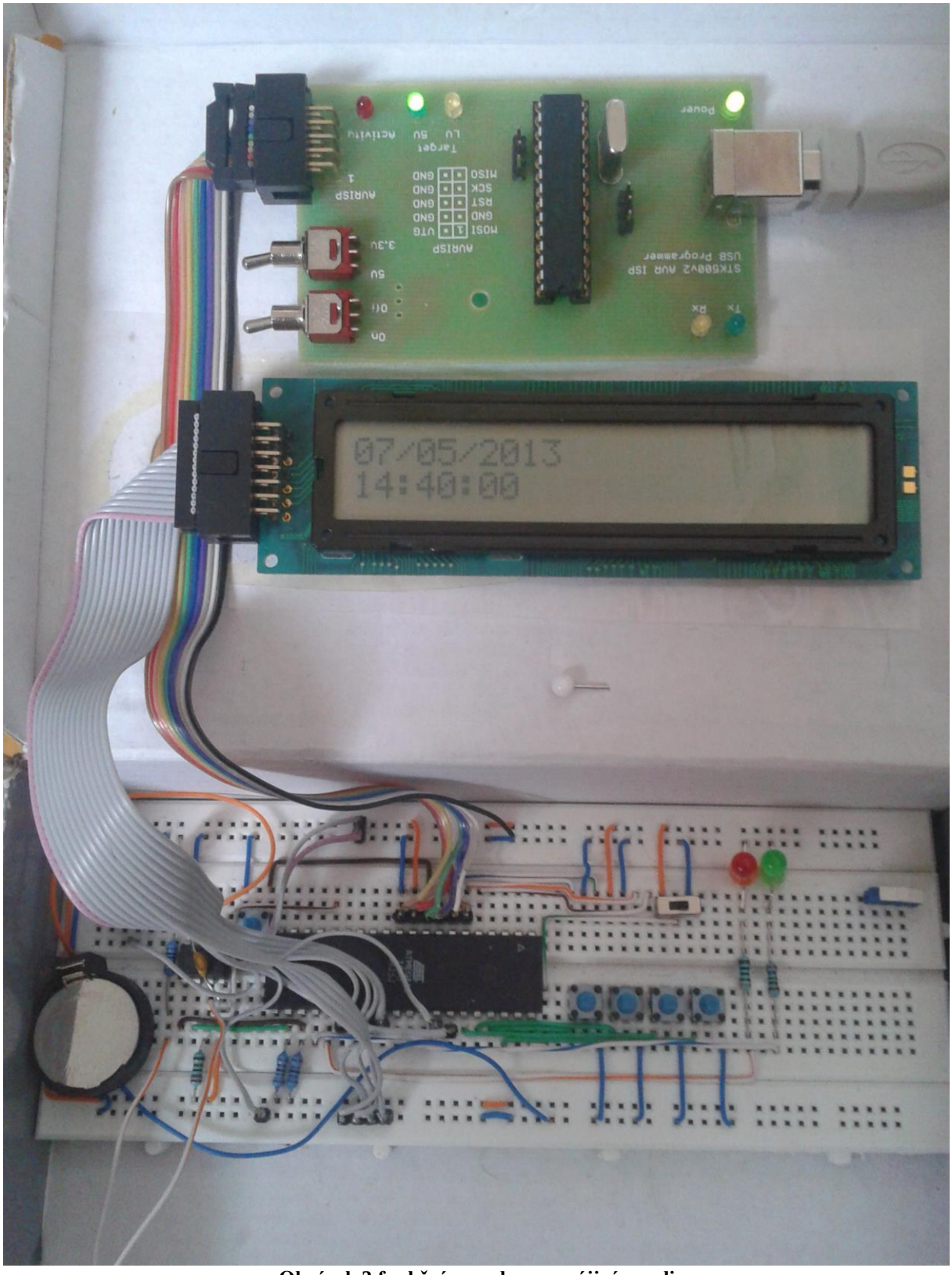


Obrázek 1 DPS strana TOP



Obrázek 2 DPS strana BOTTOM

Příloha B – Obrázek funkčního vzorku



Obrázek 3 funkční vzorek na nepájivém poli

Příloha C – Programová část

```
#define F_CPU 4000000UL

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

#include <avr/signal.h>

#include "RTC_routines.h"
#include "i2c_routines.h"

#define inc_den           20          // inkrementacni konstanta pro den
#define dec_den           4           // dekrementacni konstanta pro den
#define inc_noc           20          // inkrementacni konstanta pro noc
#define dec_noc           1           // dekrementacni konstanta pro noc
#define mez_sumy          279         // komparacni mez pro sumu
#define delka_max         12          // max delka cerpani
#define delka_min         2           // min delka cerpani

#define LCD_data          PORTB
#define LCD_instrreg      PORTA &= ~0x02
#define LCD_datareg       PORTA |= 0x02
#define LCD_disable       PORTA &= ~0x01
#define LCD_enable        PORTA |= 0x01

#define LCD_clear         0x01
#define LCD_entryset      0x06 //6 | 4
#define LCD_on                  0x0E
#define LCD_systemset     0x28
#define LCD_cursor        0x1C
#define LCD_ddram0        0x80
#define LCD_ddram1        0xC0

#define cerpadlo_on       (PINA & 0x80)

#define led_on                PORTA |= 0x02
#define led_off               PORTA &= ~0x02
#define red_led_on        PORTC |= 0x08
#define red_led_off       PORTC &= ~0x08
#define green_led_on      PORTC |= 0x04
#define green_led_off     PORTC &= ~0x04

#define KEY1_PRES         (!(PINA & 0x20))
#define KEY2_PRES         (!(PINA & 0x10))
#define KEY3_PRES         (!(PINA & 0x08))
#define KEY4_PRES         (!(PINA & 0x04))

volatile unsigned char buffer[2][24];
volatile unsigned char increment = 20;
volatile unsigned char decrement = 4;
volatile unsigned char stav = 0;
volatile unsigned char delka = 0;
volatile unsigned char error = 0;
volatile unsigned char pomocny = 0;
volatile unsigned int suma = 0;
```

```
void blink_LED(void);
void lcd_send (char pomch);
void lcd_init (void);
void buff_clr (int a);
void buff_write (int cislobuf, int druhtextu);
void lcd_get (int cislobuf, int pocetznaku);
void get_datetime (void);
unsigned char OneHzSquareWave(void);

unsigned char OneHzSquareWave(void);

int main(void)
{
	char ch,ch1;
	_delay_ms(5);

	unsigned int i;

	//PORTC = 0x03;				//pull-up pro i2c
	DDRA = 0x02;				//vystup na LED
	DDRD |= 0x60;				//vystup na konektor pro akcni zasah
	DDRC = 0x0C;				//vystup pro LED
	PORTA = 0x3c;				//pull-up pro tlacitka
	PORTD |= 0x24;				//pull-up pro external interupt

	lcd_init();
	twi_init();
	OneHzSquareWave();

nav1:
	buff_clr(0);
	RTC_displayDate();
	RTC_displayTime();
	for (i=0;i<10;i++)
	{
		if (i<8)
		{
			buffer[1][i] = time[i];
		}

		buffer[0][i] = date[i];
	}
	lcd_print(0,1);
	lcd_print(1,2);

	while(!KEY4_PRES && !KEY3_PRES);
	if (KEY4_PRES)
	{
		get_datetime();
		goto nav1;
	}

	suma = 0;
	delka = 0;

	GICR |= 0x40;
	MCUCR |= 0x03;
```

```
	sei();

	while (1)
	{
		if ((PIND & 0x40) == 1)
		{
			error = 4;
		}
		if ((PINA & 0x20) == 1)
		{
			error = 5;
		}

		if (pomocny == 1)
		{
			pomocny = 0;

			if (error < 0)
			{
				PORTD &= ~0x20;
				PORTD |= 0x40;

				GICR &= ~0x40;
				buff_write(1,3);
				lcd_print(1,2);

				blink_LED();

				buff_clr(1);
				lcd_print(1,2);
			}

			for (i=0;i<8;i++)
			{
				buffer[0][i] = time[i];
			}
			lcd_print(0,1);
		}
	}
}

void buff_write (int cislobuf, int druhtextu) // cislobuf -> cislo bufferu; b ->
druh textu
{
	switch (druhtextu)
	{
		case 0:
		{
			buffer[cislobuf][0] = 'Z';
			buffer[cislobuf][1] = 'A';
			buffer[cislobuf][2] = 'D';
			buffer[cislobuf][3] = 'E';
			buffer[cislobuf][4] = 'J';
			buffer[cislobuf][5] = ' ';
			buffer[cislobuf][6] = 'D';
			buffer[cislobuf][7] = 'E';
			buffer[cislobuf][8] = 'N';
			buffer[cislobuf][9] = ' ';
			buffer[cislobuf][10] = 'J';
```

```
		buffer[cislobuf][11] = 'A';
		buffer[cislobuf][12] = 'K';
		buffer[cislobuf][13] = 'O';
		buffer[cislobuf][14] = ' ';
		buffer[cislobuf][15] = 'C';
		buffer[cislobuf][16] = 'I';
		buffer[cislobuf][17] = 'S';
		buffer[cislobuf][18] = 'L';
		buffer[cislobuf][19] = 'O';
		buffer[cislobuf][20] = ' ';
		buffer[cislobuf][21] = ' ';
		buffer[cislobuf][22] = ' ';
		buffer[cislobuf][23] = ' ';
	} break;

	case 1:
	{
		buffer[cislobuf][0] = 'D';
		buffer[cislobuf][1] = 'D';
		buffer[cislobuf][2] = 'M';
		buffer[cislobuf][3] = 'M';
		buffer[cislobuf][4] = 'Y';
		buffer[cislobuf][5] = 'Y';
		buffer[cislobuf][6] = ' ';
		buffer[cislobuf][7] = 'Z';
		buffer[cislobuf][8] = 'A';
		buffer[cislobuf][9] = 'D';
		buffer[cislobuf][10] = 'E';
		buffer[cislobuf][11] = 'J';
		buffer[cislobuf][12] = ' ';
		buffer[cislobuf][13] = 'D';
		buffer[cislobuf][14] = 'A';
		buffer[cislobuf][15] = 'T';
		buffer[cislobuf][16] = 'U';
		buffer[cislobuf][17] = 'M';
		buffer[cislobuf][18] = ' ';
		buffer[cislobuf][19] = ' ';
		buffer[cislobuf][20] = ' ';
		buffer[cislobuf][21] = ' ';
		buffer[cislobuf][22] = ' ';
		buffer[cislobuf][23] = ' ';
	}break;

	case 2:
	{
		buffer[cislobuf][0] = 'H';
		buffer[cislobuf][1] = 'H';
		buffer[cislobuf][2] = 'M';
		buffer[cislobuf][3] = 'M';
		buffer[cislobuf][4] = 'S';
		buffer[cislobuf][5] = 'S';
		buffer[cislobuf][6] = ' ';
		buffer[cislobuf][7] = 'Z';
		buffer[cislobuf][8] = 'A';
		buffer[cislobuf][9] = 'D';
		buffer[cislobuf][10] = 'E';
		buffer[cislobuf][11] = 'J';
		buffer[cislobuf][12] = ' ';
		buffer[cislobuf][13] = 'C';
```

```
                buffer[cislobuf][14] = 'A';
                buffer[cislobuf][15] = 'S';
                buffer[cislobuf][16] = ' ';
                buffer[cislobuf][17] = ' ';
                buffer[cislobuf][18] = ' ';
                buffer[cislobuf][19] = ' ';
                buffer[cislobuf][20] = ' ';
                buffer[cislobuf][21] = ' ';
                buffer[cislobuf][22] = ' ';
                buffer[cislobuf][23] = ' ';
            } break;

            case 3:
            {
                buffer[cislobuf][0] = 'E';
                buffer[cislobuf][1] = 'R';
                buffer[cislobuf][2] = 'R';
                buffer[cislobuf][3] = 'O';
                buffer[cislobuf][4] = 'R';
                buffer[cislobuf][5] = ' ';
                buffer[cislobuf][6] = ' ' + error;
                buffer[cislobuf][7] = ' ';
                buffer[cislobuf][8] = ' ';
                buffer[cislobuf][9] = ' ';
                buffer[cislobuf][10] = ' ';
                buffer[cislobuf][11] = ' ';
                buffer[cislobuf][12] = ' ';
                buffer[cislobuf][13] = ' ';
                buffer[cislobuf][14] = ' ';
                buffer[cislobuf][15] = ' ';
                buffer[cislobuf][16] = ' ';
                buffer[cislobuf][17] = ' ';
                buffer[cislobuf][18] = ' ';
                buffer[cislobuf][19] = ' ';
                buffer[cislobuf][20] = ' ';
                buffer[cislobuf][21] = ' ';
                buffer[cislobuf][22] = ' ';
                buffer[cislobuf][23] = ' ';
            } break;
        }
}

void lcd_print (int cislobuf, int radek)//cislobuf -> cislo baufferu; b-> radka (1
nebo 2)
{
        unsigned char ch;

        LCD_instrreg;
                _delay_ms(1);

        if (radek == 1)
        {
                lcd_send(LCD_ddram0);
        }
        if (radek == 2)
        {
                lcd_send(LCD_ddram1);
        }
```

```
	LCD_datareg;
		_delay_ms(1);

	for ( ch=0 ; ch<=24 ; ch++ )
	{
		lcd_send(buffer[cislobuf][ch]);
	}
}

void lcd_get (int cislobuf, int pocetznaku) // cislobuf => cislo bufferu ve kterem
bude vysledek
{
	unsigned char adresa = 0xC0;
	unsigned int i = 0;

	LCD_datareg;
	_delay_ms(1);

	while (i<pocetznaku)
	{
		buffer[cislobuf][i] = '0';

		while (!(KEY3_PRES))
		{

			LCD_instrreg;
			_delay_ms(1);

			lcd_send(adresa);

			LCD_datareg;
			_delay_ms(1);

			lcd_send(buffer[cislobuf][i]);

			LCD_instrreg;
			_delay_ms(1);

			lcd_send(0x10);

			LCD_datareg;
			_delay_ms(1);


			if (KEY1_PRES)
			{
				_delay_ms(5);
				while(KEY1_PRES);
				if (buffer[cislobuf][i] == '9')
				{
					buffer[cislobuf][i] = '0';
				}
				else
				{
					buffer[cislobuf][i] ++;
				}
			}
```

```
                    if (KEY2_PRES)
                    {
                          _delay_ms(5);
                          while(KEY2_PRES);
                          if (buffer[cislobuf][i] == '0')
                          {
                                buffer[cislobuf][i] = '9';
                          }
                          else
                          {
                                buffer[cislobuf][i] --;
                          }
                    }
              }
              _delay_ms(5);
              while((KEY3_PRES));
              i++;
              adresa++;
       }
}

void buff_clr (int a) //a=0 nebo 1 => bufer1_clr; a=0 nebo 2 buffer2_clr
{
       unsigned int i;

       for ( i=0 ; i<24 ; i++)
       {
              if (a==0 || a==1)
              {
                    buffer[0][i] = ' ';
              }
              if (a==0 || a==2)
              {
                    buffer[1][i] = ' ';
              }
       }
}

void lcd_init (void)
{

       DDRC |= 0xF0;       //data
       DDRA |= 0xC0;       // inst/data EN
       PORTC &= ~0xF0;


       LCD_disable;
       LCD_instrreg;
       lcd_send(LCD_systemset);
       lcd_send(LCD_systemset);
       lcd_send(LCD_entryset);
       lcd_send(LCD_on);
       lcd_send(LCD_clear);
       LCD_datareg;
}

void lcd_send (char pomch)
{
       LCD_data = pomch;
```

```
		LCD_enable;
		_delay_ms(1);
		LCD_disable;
		_delay_ms(1);
}

void blink_LED(void)
{
	while (1)
	{
		red_led_on;
		_delay_ms(100);
		red_led_off;
		_delay_ms(100);
		if (KEY4_PRES)
		{
			suma = 0;
			delka = 0;
			error = 0;
			PORTD &= ~0x40;
			PORTD |= 0x20;
			GICR |= 0x40;
			return;
		}
	}
}

void get_datetime (void)
{
NAV_TIME:
	buff_clr(0);
	buff_write(0,2);

	lcd_print(0,1);
	lcd_print(1,2);
	lcd_get(1,6);

	if (buffer[1][0] < 0x30 || buffer[1][0] > 0x32)	goto NAV_TIME;
	time[0] = buffer[1][0];

	if (buffer[1][1] < 0x30 || buffer[1][1] > 0x39)	goto NAV_TIME;
	time[1] = buffer[1][1];

	if(((time[1] & 0x0f) + ((time[0] & 0x03)*10)) > 23) goto NAV_TIME;

	if (buffer[1][2] < 0x30 || buffer[1][2] > 0x35)	goto NAV_TIME;
	time[3] = buffer[1][2];

	if (buffer[1][3] < 0x30 || buffer[1][3] > 0x39)	goto NAV_TIME;
	time[4] = buffer[1][3];

	if (buffer[1][4] < 0x30 || buffer[1][4] > 0x35)	goto NAV_TIME;
	time[6] = buffer[1][4];

	if (buffer[1][5] < 0x30 || buffer[1][5] > 0x39)	goto NAV_TIME;
	time[7] = buffer[1][5];

	RTC_updateRegisters();
	RTC_writeTime();
```

```
NAV_DATE:
	buff_clr(0);
	buff_write(0,1);

	lcd_print(0,1);
	lcd_print(1,2);
	lcd_get(1,6);

	if (buffer[1][0] < 0x30 || buffer[1][0] > 0x33)     goto NAV_DATE;
	date[0] = buffer[1][0];

	if (buffer[1][1] < 0x30 || buffer[1][1] > 0x39)     goto NAV_DATE;
	date[1] = buffer[1][1];

	if(((date[1] & 0x0f) + ((date[0] & 0x03)*10)) > 31) goto NAV_DATE;

	if (buffer[1][2] < 0x30 || buffer[1][2] > 0x31)     goto NAV_DATE;
	date[3] = buffer[1][2];

	if (buffer[1][3] < 0x30 || buffer[1][3] > 0x39)     goto NAV_DATE;
	date[4] = buffer[1][3];

	if(((date[4] & 0x0f) + ((date[3] & 0x03)*10)) > 12) goto NAV_DATE;

	if (buffer[1][4] < 0x30 || buffer[1][4] > 0x39)     goto NAV_DATE;
	date[8] = buffer[1][4];

	if (buffer[1][5] < 0x30 || buffer[1][5] > 0x39)     goto NAV_DATE;
	date[9] = buffer[1][5];

	date[2] = '/';
	date[5] = '/';
	date[6] = '2';
	date[7] = '0';

	buff_clr(0);
	buff_write(0,0);

	lcd_print(0,1);
	lcd_print(1,2);
	lcd_get(1,1);

	if (buffer[1][0] < 0x30 || buffer[1][0] > 0x39)     goto NAV_DATE;
	date[10] = buffer[1][0];

	RTC_updateRegisters();
	RTC_writeDate();
}

unsigned char OneHzSquareWave(void)
{
	unsigned char errorStatus, i;

	errorStatus = i2c_start();
	if(errorStatus == 1)
	{
		i2c_stop();
```

```
	}

	errorStatus = i2c_sendAddress(DS1307_W);

	if(errorStatus == 1)
	{
		i2c_stop();
	}

	errorStatus = i2c_sendData(0x07);
	if(errorStatus == 1)
	{
		i2c_stop();
	}

	errorStatus = i2c_sendData(0x10);
	if(errorStatus == 1)
	{
		i2c_stop();
	}


	i2c_stop();
}

ISR(INT0_vect)
{
	pomocny = 1;

	if (cerpadlo_on == 1)
	{
		if (stav < 2)
		{
			stav++;
			green_led_on;
		}
		if (delka > delka_max)
			//testovani na maxim8lni delku cerpani
		{
			error = 1;
		}

	}
	else
	{
		stav = 0;
		green_led_off;

		if (delka < delka_min)
			//testovani na minimalni delku cerpani
		{
			error = 2;
		}
		else
		{
			delka = 0;
		}
	}
```

```
	if(stav == 1)
                        //provede se pro kazde zapnuti cerpadla pouze jednou
	{
		RTC_getTime();
		if (((HOURS & 0x3F) > 0x07) && ((HOURS & 0x3F) < 0x20))         //od
7:00 do 20:00 nastaveni konstant pro denni provoz  (time[0]<'2'|| (time[0]='0' &&
time[1]<'7'))
		{
			increment = inc_den;
			decrement = dec_den;
			led_on;
		}
		else
		{
			increment = inc_noc;
			decrement = dec_noc;
		}
	}

	if (stav > 0)
	{
		suma += increment;
		delka += 1;
	}
	else
	{
		if (suma >= decrement)
		{
			suma -= decrement;
		}

	}
	if (suma > (mez_sumy-1))
	{
		error = 3;
	}
}
```