Vysoká škola báňská – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra kybernetiky a biomedicínského inženýrství

Programování základních periferií mikropočítače AT MEGA – demonstrační úlohy pro praktická cvičení

Programming Basic Microcontroller Peripherals AT MEGA – Demonstration Tasks for Practical Exercises

2016 Jakub Dohnal

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra kybernetiky a biomedicínského inženýrství

Zadání bakalářské práce

Student: **Jakub Dohnal**

Studijní program: B2649 Elektrotechnika

Studijní obor: 2612R041 Řídicí a informační systémy

Téma: Programování základních periférií mikropočítače AT MEGA - demonstrační úlohy pro praktická cvičení
Programming Basic Microcontroller Peripherals AT MEGA - Demonstration Tasks for Practical Exercises

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem bakalářské práce je naprogramování určených aplikací pro laboratorní využití.

Body zadání:
1. Studium mikropočítačového systému AT MEGA a možnosti jeho programování.
2. Pro laboratorní využití naprogramujte sadu aplikací:
   - Digitalizace signálu A/D převodníkem.
   - Ukládání dat do externí velkokapacitní paměti (paměťová SD karta).
   - Realizace systémového času pomocí IO reálného času.
   - Zobrazení dat na LCD displeji.
   - Měření teploty čidlem DALLAS.
3. K realizovaným úlohám připravte krátký popis a způsob programování periferie.
4. Provedení praktického ověření funkčnosti aplikací.
5. Zhodnocení dosažených výsledků.

Seznam doporučené odborné literatury:

[1] ATMEGA644 Datasheet. Dostupné z: http://www.alldatasheet.com/datasheet-pdf/pdf/174761/ATMEL/ATMEGA644.html.
[2] ATmega644. Dostupné z: http://www.atmel.com/images/doc2593.pdf.
[3] Návod k použití vývojového kitu EvB 4.3 v4. Dostupné z: http://www.and-tech.pl.
[4] AVR tutorial. Dostupné z: http://www.solarskit.wz.cz/avrprogramming.html.
[5] MATOUŠEK, David. *Aplikace mikrokontrolérů ATmega644.* 1. vyd. Praha: BEN - technická literatura, 2013, 1 sv. (různé stránkování). ISBN 978-80-7300-492-7.
[6] BRTNÍK, Bohumil a David MATOUŠEK. *Programování mikrokontrolérů s jádrem 8051 v jazyce C: názorné příklady a funkční programy pro AT89S52.* 1. vyd. Praha: BEN - technická literatura, 2010, 151 s. ISBN 978-80-7300-264-0.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **doc. Ing. Ludvík Koval, Ph.D.**

Datum zadání: 01.09.2015

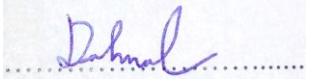
Datum odevzdání: 29.04.2016

doc. Ing. Jiří Koziorek, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení o samostatném vypracování

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.



V Ostravě dne 29. 4. 2016

Poděkování

Tímto bych chtěl poděkovat panu doc. Ing. Ludvíkovi Kovalovi, Ph.D. za cenné rady, které mi poskytl při řešení bakalářské práce. Dále bych chtěl tímto poděkovat panu Ing. Martinu Piešovi, Ph.D. za zapůjčení programátoru.

Abstrakt

Cílem této práce je vytvoření sady jednotlivých aplikací pro mikrokontrolér ATmega644P s popisem funkcí a jednotlivých operací nutných pro správnou funkčnost periferií.

První část práce se zabývá popisem vývojového kitu, na kterém jsou realizovány dané úlohy. V dalších částech jsou popisovány samotné obvody, přítomné na vývojovém kitu, jejich vlastnosti a způsoby komunikace. U každé periferie, pro kterou byl tvořen program, je uveden způsob propojení s řídicím mikrokontrolérem a nastavení vývodů v programu pro možnost změny zapojení. Dále jsou uvedeny klíčové části kódu s popisem funkce. Nakonec jsou uvedeny způsoby programování vývojového kitu za pomoci programátoru a přes USB.

Klíčová slova

Mikrokontrolér, vývojový kit, AD převodník, teplotní čidlo, obvod reálného času, LCD displej, SD karta, rozhraní, datalogger, souborový systém, programování, programátor.

Abstract

The goal of this thesis is to create a set of individual applications for ATmega644P microcontroller unit with the description of functionsand operations that are necessary for proper function of peripherals.

The first part of this thesis deals with the description of the development kit, on which tasks are implemented. In the next section thereis description of the circuits that are present on the development board, their properties and methods of communication. For each periphery there is a example of how to communicate with the kit.various ways how to program the kit via the programmer or via USB.

Key words

Microcontroller unit, development kit, AD converter, temperature sensor, real-time clock, LCD, SD card, interface, datalogger, file system, programming, programmer.

Seznam použitých zkratek

ACK	Acknowledgment (Potvrzení)
AD	Analogově/Digitální
ALU	Arithmetic Logic Unit (Aritmeticko-logická jednotka)
CISC	Complex Instruction Set Computing (Komplexní instrukční sada)
CLK	Clock (Hodinový signál)
CMOS	Complementary Metal–Oxide–Semiconductor
CS	Chip Select
EEPROM	Electrically Erasable Programmable Read-Only Memory
FAT	File Allocation Table
I^2C	Inter-Integrated Circuit
IO	Integrovaný Obvod
JTAG	Joint Test Action Group
LCD	Liquid-Crystal Display
LED	Light-Emitting Diode
LSB	Least Significant Bit
MCU	MicroController Unit
MISO	Master In Slave Out
MMC	MultiMediaCard
MOSI	Master Out Slave In
PDI	Program and Debug Interface
PWM	Pulse Width Modulation
RAM	Random Access Memory
ROM	Read-Only Memory
RTC	Real-Time Clock
SD	Secure Digital
SPI	Serial Peripheral Interface
TWI	Two Wire Interface
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

Seznam obrázků

Seznam tabulek

1 Úvod

V dnešní době je většina elektrických zařízení, která nás obklopují, řízena pomocí počítačů, nebo MCU. To, že se používají programovatelné obvody, umožňuje modifikovat dle potřeby funkčnost celého zařízení, nebo dokonce úplnou změnu celé funkčnosti obvodu. MCU představují jednočipové počítače s takovou konfigurací, aby byly schopny samostatně pracovat, a tedy k nim stačí pouze připojit vstupní a výstupní periferie a samozřejmě do nich nahrát požadovaný program. Je také možné na stejnou aplikaci použít jiný MCU, který ovšem musí odpovídat parametry a použitými prvky původnímu MCU a program se musí dodatečně upravit pro daný typ obvodu (samozřejmě nelze použít stejný hex soubor, musí se program znovu zkompilovat pro jiný typ MCU).

Tato bakalářská práce je realizovaná na vývojovém kitu EvB 4.3 v4, který již obsahuje veškeré hardwarové vybavení, a proto není nutné připojovat další obvody, či periferie.

Výsledkem práce je sada aplikací, které budou pracovat s různými periferiemi a pomocí jednoduchých programů tak bude možné demonstrovat jejich funkci a možnosti nastavení připojení. Tato práce tak snoubí různé druhy komunikace s jednotlivými periferiemi, popisuje vlastnosti jednotlivých periferií, způsoby nastavení a jsou uvedeny části kódu s popisem jednotlivých funkcí. Jedná se tedy o stručný návod k naprogramování používaných periferií. Zjednodušeně řečeno je cílem práce měření a zobrazování teploty, času, napětí a ukládání dat v podobě textu na SD kartu.

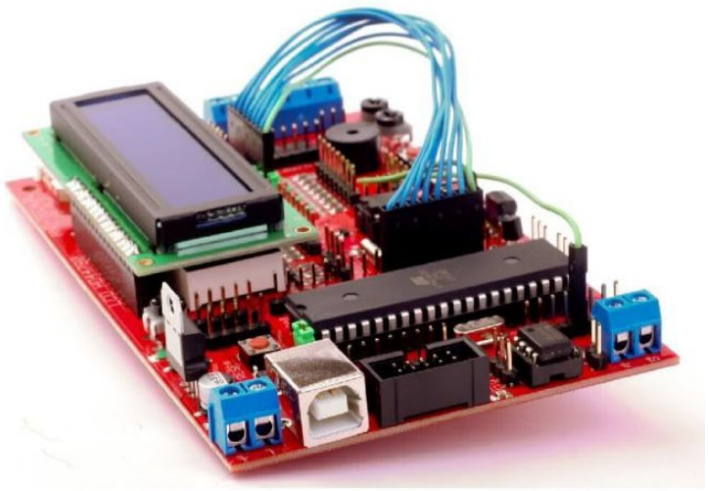
2 Vývojový kit EvB 4.3 v4

Tento vývojový kit se všeobecně používá k výuce programování MCU a různých periferií. Tyto periferie se k MCU připojují pomocí propojovacích vodičů, přičemž napájení periferií je ve většině případů již zapojeno.

Vývojová deska je vybavena následujícími prvky:

- ATMega644p v pouzdře DIP40
- Hodiny reálného času PCF8583
- Paměť EEPROM AT24C02
- Infračervený přijímač TSOP4836
- Teplotní čidlo DS18B20
- Převodník sběrnic RS485
- Patice pro karty MMC/SD
- 5 tlačítek
- 8 diod LED
- 3 tranzistorové výstupy 500 mA každý
- 2 tranzistorové výstupy 1 A každý
- 2 analogové potenciometry
- Buzzer (piezoměnič s budičem)
- 4 x 7segmentový LED zobrazovač
- USB port
- Konektor ISP
- LCD displej 2 x 16 znaků [2]

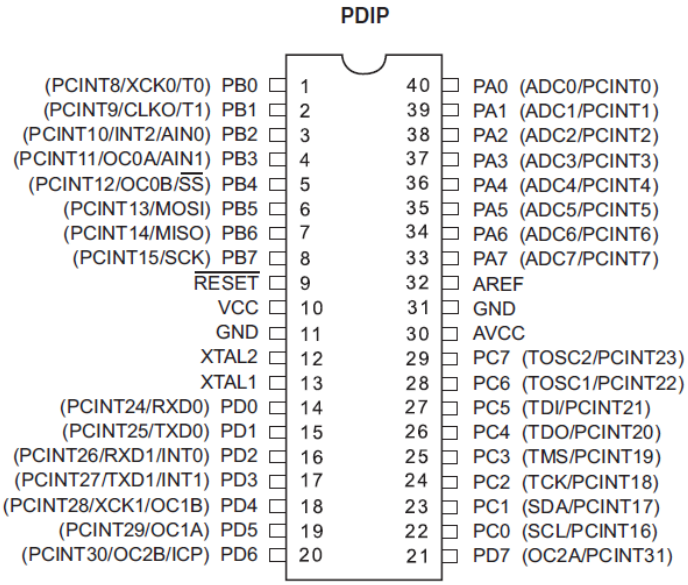
Vývojová deska může být napájená buďto přímo z USB konektoru typu B, který je chráněný pojistkou na 0,5 A, nebo lze využít napájecí vstup šroubovací svorkovnice, který má ochranu proti přepólování sériově zapojenou diodou v propustném směru a za tímto vstupem je stabilizátor na 5 V. Všechny použité prvky budou popsány v následujících kapitolách.



Obrázek 1 Pohled na vývojový kit [2]

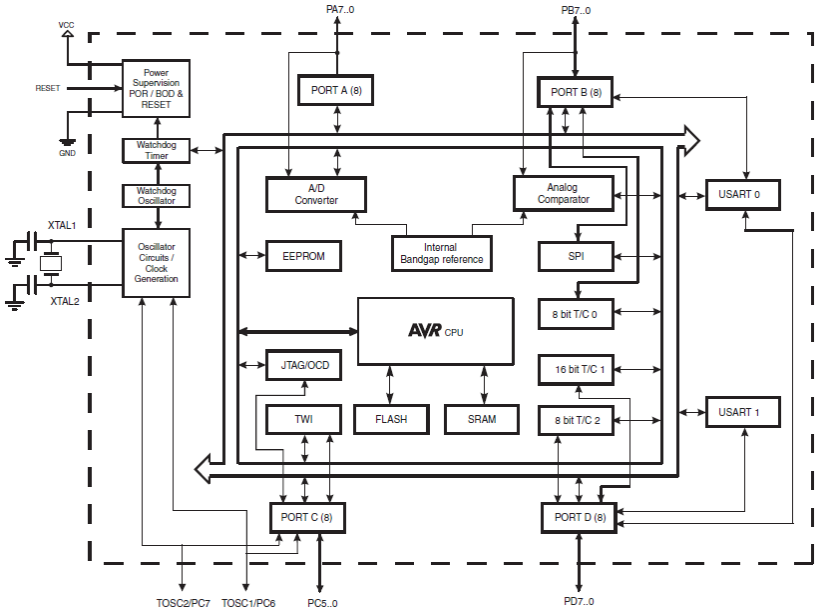
3 MCU ATmega644P

ATmega644P je osmibitový MCU od firmy Atmel. Jeho napájení je 2,7 – 5,5 V, přičemž při napájecím napětí 2,7 - 5,5 V je schopný pracovat do frekvence 10 MHz a při napájení 4,5 – 5,5 V je schopný pracovat až do frekvence 20 MHz. Tento MCU má flash paměť o velikosti 64 kB, EEPROM paměť o velikosti 2 kB a RAM paměť o velikosti 4 kB. Dále tento MCU obsahuje dva osmibitové čítače/časovače, jeden 16bitový čítač/časovač, šest PWM kanálů, osmi - kanálový desetibitový AD převodník. ATmega644P má pouzdro DIP40, kde je 32 programovatelných pinů, které odpovídají čtyřem osmibitovým portům označovaným A až D.



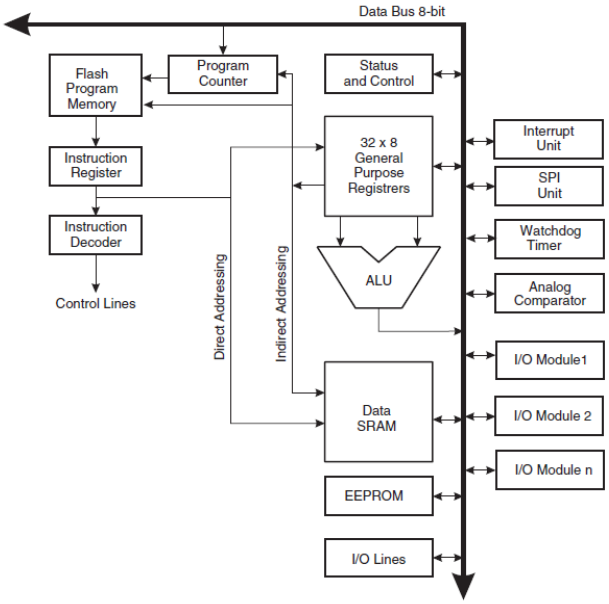
Obrázek 2 Rozložení pinů ATmega644P [1]

Piny 10 a 11 slouží k napájení MCU, pin 9 slouží k resetování MCU, kdy se po přivedení logické 0 dojde k provádění nahraného programu od začátku, nebo se přejde do programovacího módu. Piny 12 a 13 slouží k připojení externího krystalu pro určení pracovní frekvence. Na piny 30 a 31 je nutné připojit napájení pro AD převodník a na pin 32 se přivádí referenční napětí pro analogový komparátor. Jak lze vidět na Obrázek 2 a Obrázek 3, tak AD převodník je situován na portu A, rozhraní I^2C je umístěno na pinech PC0 a PC1, rozhraní JTAG je na pinech PC2 až PC5 a SPI rozhraní je na pinech PB4 až PB7.



Obrázek 3 Blokový diagram MCU [1]

AVR jádro kombinuje instrukční sadu s 32 funkčními registry. Všech 32 registrů je připojeno k aritmeticko-logické jednotce (ALU) a to umožňuje přístup k operandům, provedení operace a přiřazení výsledku do registru v jednom strojovém cyklu. Výsledná architektura dosahuje až desetinásobné propustnosti oproti konvenčním CISC MCU.



Obrázek 4 Blokový diagram AVR architektury [1]

Tento MCU využívá Harvardskou architekturu, která se vyznačuje tím, že má oddělenou paměť a sběrnici pro data a program. Instrukce v programové paměti jsou vykonávány pomocí zřetězeného zpracování (pipelinigu).

3.1 Externí přerušení

Externí přerušení může být spuštěno pomocí pinů INT0 až INT2, nebo PCINT0 až PCINT31. Pokud se povolí přerušení, tak se přerušení spustí i když je sledovaný pin nastavený jako výstup. To umožňuje vytvoření softwarového přerušení.

3.2 AD převodník

Integrovaný AD převodník má desetibitové rozlišení, ±2 LSB absolutní přesnost, čas převodu 13 až 260 µs, 8 kanálů, rozdílový mód s měnitelným zesílením 1x, 10x až 200x. Vstupní napětí je od 0 V až po napájecí napětí, AD převodník začne automaticky převod po přerušení. AD převodník je dostupný na portu A.

Pro nastavení vstupu napěťové reference AD převodníku a jeho povolení na jednotlivých pinech MCU se musí nastavit registr ADMUX.

Tabulka 1 Registr ADMUX [1]

Bit	7	6	5	4	3	2	1	0
Název bitu	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0

Pro nastavení zdroje napěťové reference slouží bity REFS1 a REFS0, jejich konfigurace se provede dle následující tabulky.

Tabulka 2 Nastavení napěťové reference pro AD převodník [1]

REFS1	REFS0	Výběr napěťové reference
0	0	Pin AREF, vnitřní reference vypnuta
0	1	Napájecí napětí
1	0	Vnitřní reference 1,1 V
1	1	Vnitřní reference 2,56 V

Bit ADLAR slouží k nastavení zarovnání výsledku převodu z AD převodníku. Pokud ho nastavíme na 1, tak dojde k zarovnání výsledku vlevo, jinak bude zarovnán vpravo.

Bity MUX4 až MUX0 slouží k nastavení vstupu pro AD převodník a také pro nastavení zesílení diferenciálních vstupů.

Tabulka 3 Nastavení vstupu pro AD převodník [1]

MUX4..0	Vstup
00000	ADC0
00001	ADC1
00010	ADC2
00011	ADC3
00100	ADC4
00101	ADC5
00110	ADC6
00111	ADC7

Tabulka 4 - Registr ADCSRA [1]

Bit	7	6	5	4	3	2	1	0
Název bitu	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0

Bit ADEN slouží k povolení AD převodníku, pokud do něj zapíšeme 0, tak bude převodník vypnut. Pro spuštění převodu slouží bit ADSC, pokud probíhá převod, tak je na tomto bitu přítomna 0, čehož se dá využít ke kontrole, zda již skončil převod. Nastavením bitu ADATE na 1 se spustí automatický trigger převodníku. AD převodník spustí převod, pokud dojde k náběžné hraně na vybraném triggerovacím signálu. Bity ADIF a ADIE slouží k nastavení přerušení.

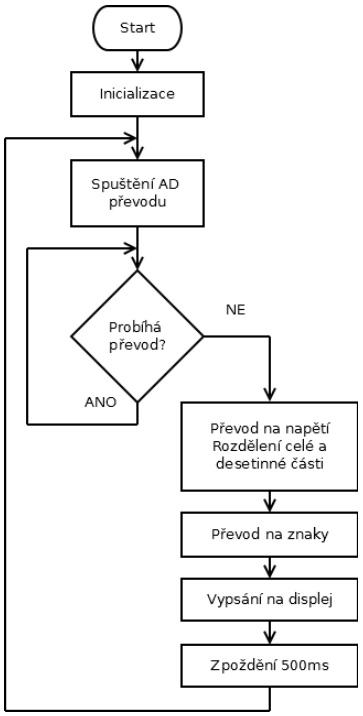
Pro nastavení děličky, která dělí frekvenci z oscilátoru na hodinový signál pro ADC, je třeba správně nastavit bity ADPS2..0.

Tabulka 5 Nastavení děličky pro AD převodník [1]

ADPS2	ADPS1	ADPS0	Dělicí poměr
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Program pro digitalizaci signálu AD převodníkem

Tento program provádí cyklické měření a zobrazování AD hodnoty a z ní odvozeného napětí, přičemž vzorkovací perioda je přibližně 0,5 s. Toto měření je vhodné pro měření pomalých dějů, lze jej využít jako jednoduchý voltmetr do 5 V (přidáním děliče by bylo možno zvětšit rozsah).



Obrázek 5 Vývojový diagram aplikace pro AD převod

3.2.1.1 Převod AD hodnoty na napětí

Pro převod hodnoty z AD převodu na napětí je třeba použít následující vzorec:

$$U = U_{ref} \cdot \frac{m}{2^n - 1}\ (V)$$

Tento vztah lze dále zjednodušit:

$$U = m \cdot k\ (V)$$

$$k = \frac{U_{ref}}{2^n - 1}\ (V)$$

kde **U** je změřené napětí; $\mathbf{U_{ref}}$ je hodnota referenčního napětí; **n** je počet bitů převodníku, **m** je číselná hodnota z AD převodníku a **k** je konstanta AD převodníku

Hodnoty použité v programu:

- $U_{ref} = 5$ V
- n = 10

$$k = \frac{5}{1024 - 1} \doteq 4{,}888\ mV$$

3.2.1.2 Nastavení AD převodníku

```
DIDR0 = 0xff;                                   // Vypnutí digitálních vstupů
ADMUX = (0 << REFS0) | (0 << REFS1);            // Napěťová reference nastavena na pin
Vref
ADMUX = ADMUX | (1 << MUX0) | (1 << MUX1) | (1 << MUX2) | (0 << MUX3) | (0 << MUX4);
                                        //nastavení ADC na vstupní pin A7
ADCSRA = (1 << ADEN);                           // Povolení ADC
ADCSRA = ADCSRA | (1 << ADPS2) | (1 << ADPS1) | (0 << ADPS0);
                                                // Dělička pro ADC nastavena na 64
```

3.2.1.3 Spuštění AD převodu

```
ADCSRA = ADCSRA | (1<<ADSC);                    // Zapne AD převod
while (ADCSRA &(1<<ADSC));                      // Čekání na dokončení převodu
AD_hodnota = ADC;                               //  Uložení  AD  hodnoty  do  proměnné,
proměnná AD_hodnota je 16bitový unsigned integer
```

3.2.1.4 Převod na napětí, vypsání na displej

Pokud se neměří jen napětí, a bude třeba přepočítat tuto hodnotu, nebo s ní nějak jinak pracovat, tak je vhodné si hodnotu napětí uložit do proměnné typu float, nebo double. Pro vypsání hodnoty napětí na displej je třeba převést tento float dále na jeden integer pro celou druhý integer pro desetinnou část. Poté je třeba převést tyto celočíselné hodnoty na znaky. Pro vyspání AD hodnoty je pouze zapotřebí tuto hodnotu také převést na znaky.

```
napeti = AD_hodnota * 0.004888;                 // Uložení hodnoty napětí

napeti_jednotky = napeti;                       // Uložení jednotek voltů

napeti_tisiciny = (napeti - napeti_jednotky)*1000;     // Uložení tisícin voltů
```

Proměnné napeti_jednotky a napeti_tisiciny jsou proměnné typu unsigned integer, takže dojde k přetypování výrazu vpravo na celočíselnou hodnotu. Pro převod integerových hodnot na pole charů slouží funkce *sprintf*(char *__s, const char *__fmt...);, kde první pole charů je výstupní proměnná, poté se zadá datový typ, z kterého se převádí a vstupní proměnná.

```
char array_AD [(sizeof(AD_hodnota))];           //  Vytvoření  pole  charů  o  velikosti
proměnné AD_hodnota
sprintf(array_AD,"%d",AD_hodnota);              //  Převod  integerové  hodnoty  na  pole
charů

// Vypsání hodnoty na LCD displej
lcd_clrscr();
lcd_puts("AD hodnota: ");
lcd_puts(array_AD);

char jednotky [(sizeof(napeti_jednotky))];      //  Vytvoření  pole  charů  o  velikosti
proměnné napeti_jednotky
sprintf(jednotky,"%d",napeti_jednotky);         // Převod jednotek napětí na pole charů
lcd_gotoxy(0,1);
lcd_puts("Napeti: ");
lcd_puts(jednotky);                             //  Vypsání  pole  charů  s  informací  o
jednotkách V
lcd_putc(',');                                  // Vložení desetinné čárky mezi celá a
desetinná čísla
```

```
char tisiciny [(sizeof(napeti_tisiciny))];      // Vytvoření pole charů o velikosti
proměnné napeti_tisiciny
sprintf(tisiciny,"%d",napeti_tisiciny);         // Převod tisícin voltů na pole charů

if (napeti_tisiciny < 100)                      // Ošetření správného zobrazení napětí
{
       lcd_putc('0');
       if (napeti_tisiciny < 10)
       {
              lcd_putc('0');
       }
}
lcd_puts(tisiciny);                             // Vypsání hodnoty tisícin voltů na
displej
lcd_puts("V");

       _delay_ms(500);                          // Zpoždění 500ms možno měnit podle
potřeby
```

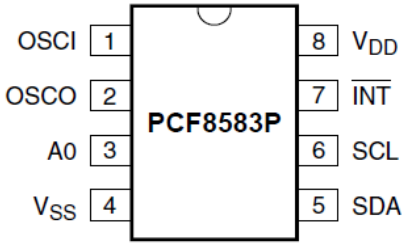
4 Externí periferie

4.1 Hodiny reálného času - PCF8583

Obvod PCF8583 jsou hodiny reálného času a kalendář, založené na 2048bitové statické CMOS RAM paměti, obsahující 256 slov po 8 bitech. Adresování a přenos dat je zprostředkován pomocí I^2C sběrnice. Vestavěný registr adres je automaticky inkrementován po každém zápisu, nebo přečtení datového bytu. Adresový pin A0 se používá pro určení hardwarové adresy, což umožňuje připojit na I^2C sběrnici více než 1 zařízení bez dodatečného hardwaru.

Interní 32,768 kHz oscilátor a prvních 8 bytů RAM paměti slouží pro obsluhu hodin, kalendáře a čítací operace. Dalších 8 bytů může být naprogramováno jako alarmy, nebo jako volná RAM paměť. Zbývajících 240 bajtů jsou volná RAM paměť.

Napájecí napětí pro I^2C sběrnici je 2,5 až 6 V. Napájecí napětí hodin je 1 až 6 V. Proud pro napájení hodin by podle datasheetu neměl přesáhnout více jak 50µA. Obvod umožňuje zvolit mezi 24 hodinovým a 12 hodinovým formátem.



Obrázek 6 Obvod PCF8583 - rozložení vývodů [3]

4.1.1 Čítací registry

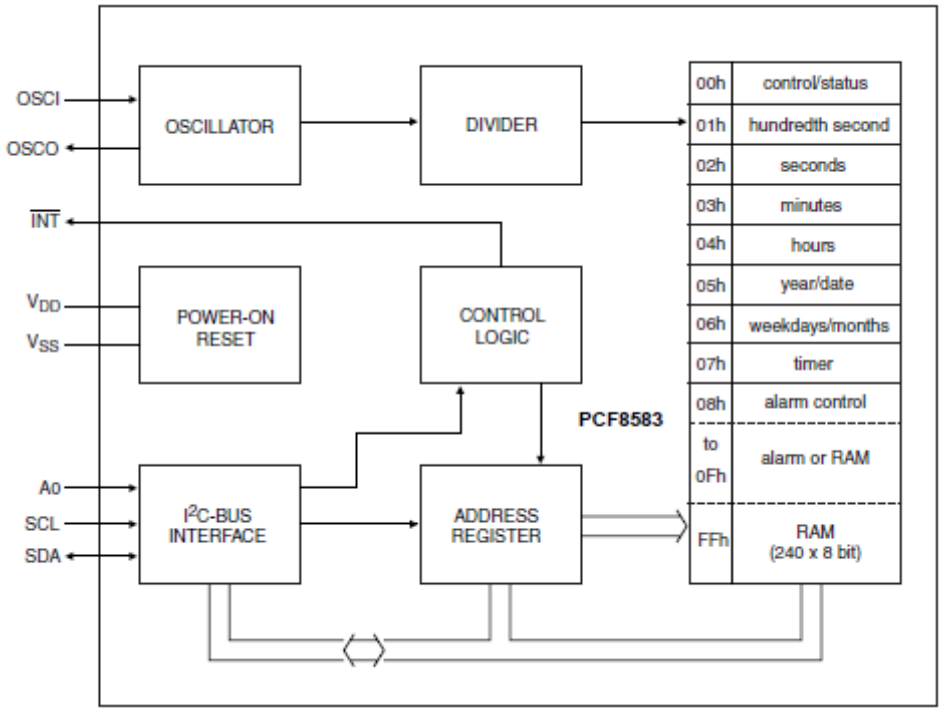
- Setiny sekund – adresa v paměti 01h. Zde jsou uloženy informace o hodnotě setin sekundy. Informace je v BCD kódu a je rozložena na desetiny, které jsou uloženy v horních 4 bitech, a setiny, které jsou uloženy ve spodních 4 bitech.
- Sekundy – adresa v paměti 02h. Zde jsou uloženy informace o hodnotě sekund. Informace je v BCD kódu a je rozložena na desítky, které jsou uloženy v horních 4 bitech, a jednotky, které jsou uloženy ve spodních 4 bitech.
- Minuty – adresa v paměti 03h. Informace jsou uloženy obdobně jako u minut.
- Hodiny – adresa v paměti 04h. Na 0. až 5. bitu jsou hodiny v BCD formátu, kde 0. – 3. bit jsou jednotky hodin a na 4. – 5. bitu jsou desítky hodin. Přepínání mezi 12 hodinovým a 24 hodinovým formátem se provádí na 7. bitu, pro 24 hodinový formát se zapíše logická 0, pro 12 hodinový formát se zapíše logická 1. Pokud zvolíme 12 hodinový formát, tak na 6. bitu je potom informace o tom, zda je odpoledne (PM – logická 1), nebo dopoledne (AM – logická 0).
- Rok/datum – adresa v paměti 05h. Na 0. až 5. bitu je datum v BCD formátu, kde 0. – 3. bit jsou jednotky dnů a na 4. – 5. bitu jsou desítky dnů. Na 6. a 7. bitu je uložený rok, který může nabývat hodnot od 0 do 3. Podle této informace lze zjistit, zda je přestupný rok, což odpovídá hodnotě 0. Toho lze docílit, když se vezme zbytek po dělení 4 a uloží se zde. Jsou ovšem určité výjimky, které je třeba ošetřit, nebo nějakým způsobem obejít.

Obecně pravidla pro zjištění, zda se jedná o přestupný rok, jsou:

1) Rok je přestupný, pokud je dělitelný číslem 4 (1996, 2004, 2008, 2012).
2) Výjimka č. 1 : Rok není přestupný, pokud je dělitelný číslem 100 (1700, 1800, 1900, 2100)
3) Výjimka č. 2 z výjimky č. 1: Rok, na který se vztahuje výjimka č. 1, je přestupný, pokud je dělitelný číslem 400 (1600, 2000, 2400). [15]

Tím se ustavil čtyřsetletý stálý cyklus shodného uspořádání dnů v jednotlivých letech.

- Dny týdne/měsíce – adresa paměti 06h. Na bitech 0 až 4 je uložena informace o aktuálním měsíci, kde bity 0 až 3 jsou jednotky měsíců a 4. bit odpovídá desítkám měsíců. Dny v týdnu jsou pak uloženy na 5. až 7. bitu a může nabývat hodnotě od 0 pro pondělí do 6 pro neděli.



Obrázek 7 Blokový diagram obvodu PCF8583 [3]

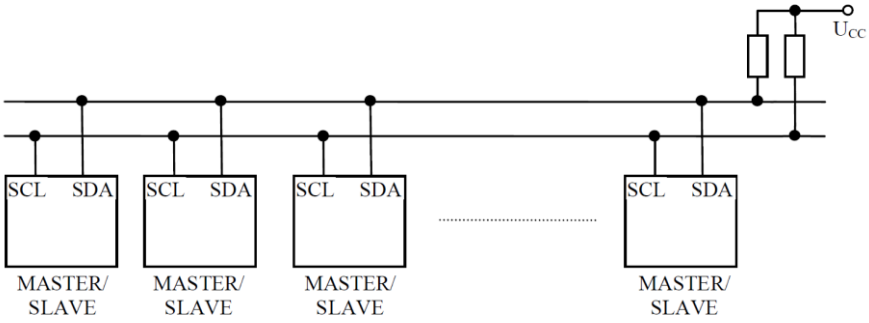
4.1.2 Rozhraní I²C

Jedná se o obousměrnou komunikaci pomocí 2 signálových vodičů, která má patentovaný název pod firmou Philips a tak tuto sběrnici označují někteří výrobci (včetně Atmelu) jako TWI, avšak jedná se o úplně stejné rozhraní. Po datovém vodiči může vždy vysílat pouze jedno zařízení a zařízení musí vždy vyčkat, zda je linka volná, neboť by jinak mohlo docházet ke kolizím.

Popis signálů:

- SDA je obousměrný datový vodič
- SCL je vodič s hodinovými pulzy určující pracovní frekvenci komunikace

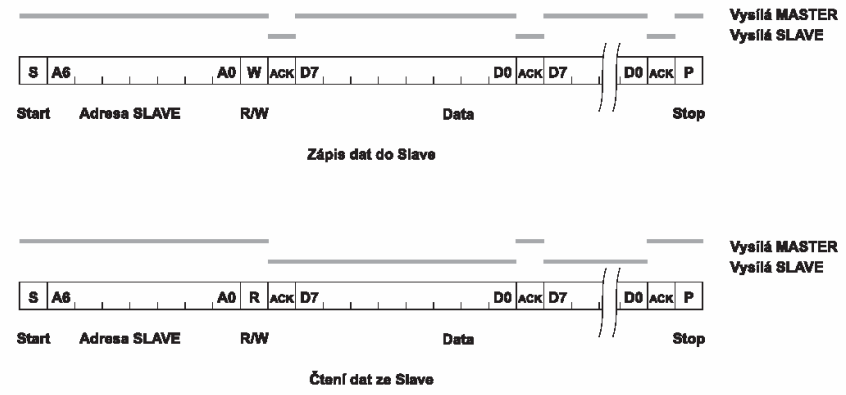
Jednotlivá zařízení zapojená přes I²C sběrnici se spojují jedním datovým vodičem SDA a jedním hodinovým vodičem SCL dohromady.



Obrázek 8 Propojení zařízení přes I²C sběrnici [9]

Oba vodiče jsou připojeny přes pull-up rezistory na napájecí napětí a z elektrického hlediska se tedy jeví jako otevřený kolektor.

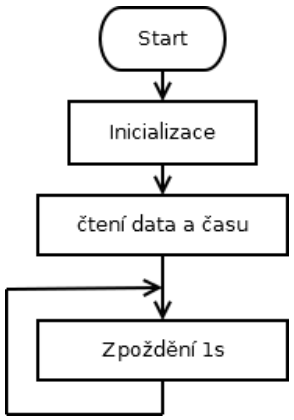
Každému přenosu předchází vyslání podmínky START. Potom je vysílána 7bitová adresa příjemce a jeden bit R/W, který indikuje požadovanou operaci (čtení/zápis). Další bit ACK je vysílán s úrovní L a je určen k potvrzení přijímací stanicí. Dále jsou přenášena data ve směru určeném předchozím bitem R/W. Každý byte je následován jedním bitem ACK. Po ukončení přenosu je vyslána podmínka STOP. [9]



Obrázek 9 Protokol komunikace I²C [9]

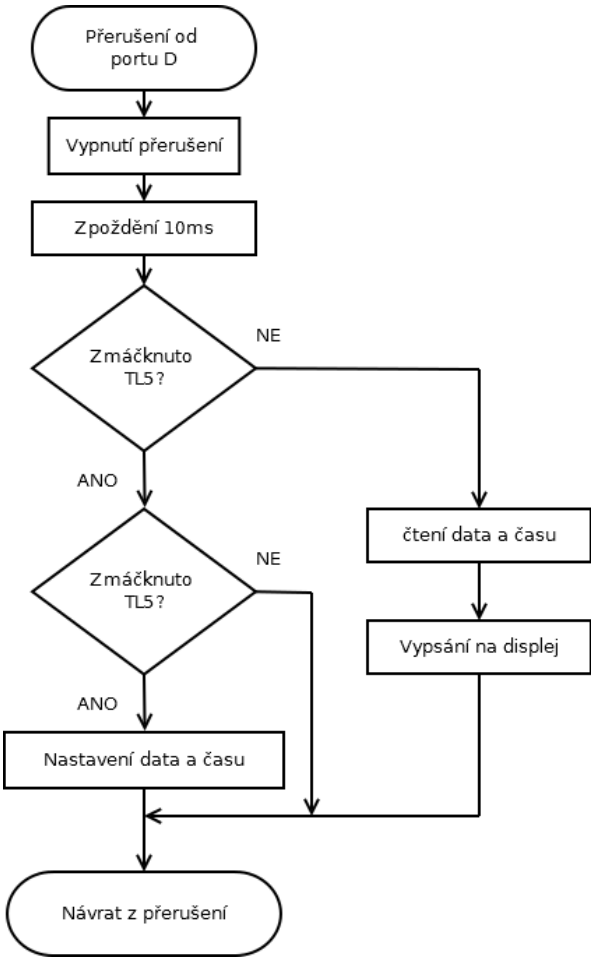
4.1.3 Realizace systémového času pomocí IO reálného času

V hlavní části tohoto programu se nachází pouze inicializace periferií a prvotní načtení data a času.



Obrázek 10 Vývojový diagram hodin - funkce main

Jak můžeme vidět na Obrázek 11 Vývojový diagram hodin - obsluha přerušení, tak veškeré operace s nastavením data a času a vypisování na displej je řešeno v přerušení.



Obrázek 11 Vývojový diagram hodin - obsluha přerušení

4.1.3.1 Propojení MCU s RTC

Obvod PCF8583 je již propojen na desce vývojového kitu pomocí cest na desce plošných spojů, není tedy třeba nic připojovat kromě přerušovacího vývodu, který je pojmenován jako INT_RTC a tlačítek.

```
#define INT_RTC (PIND & 0b01000000) >> 6          // Pin na PD6
#define TL1 (PIND & 0b00000001)                   // Pin na PD0
#define TL2 (PIND & 0b00000010) >> 1              // Pin na PD1
#define TL3 (PIND & 0b00000100) >> 2              // Pin na PD2
#define TL4 (PIND & 0b00001000) >> 3              // Pin na PD3
#define TL5 (PIND & 0b00010000) >> 4              // Pin na PD4
```

4.1.3.2 Funkce main

Pokud dojde ke změně pinu pro vstup přerušení, tak je třeba změnit přerušení, je však zapotřebí, aby přerušovací pin byl na portu D, neboť je využit jako port pro přerušení.

```
DDRD   &= ~0xff;                  // Nastavení portu D jako vstup
PORTD  |= 0xff;                   // Zapnutí pull-up rezistorů na portu D
```

```
PCMSK3 |= (1 << PCINT30);          // Zapnutí přerušení pro pin PD6 (přerušení od RTC)
PCMSK3 |= (1 << PCINT28);          // Zapnutí přerušení pro pin PD4 (TL5)
PCICR  |= (1 << PCIE3);            // Zapnutí přerušení na portu D

lcd_init();                                 // Inicializace LCD
i2c_init();                                 // Inicializace sběrnice I²C

i2c_start(PCF8583_zapis);                   // Zapsání instrukce pro zápis
i2c_write(0x0D);                            // Nastavení adresy na registr CLKOUT
i2c_write(0b10000011);                      // Aktivace přerušovacího výstupu z RTC 1Hz
i2c_stop();                                 // Konec přenosu

cteni_data();

sei();                                      // Povolení přerušení
while (1)                                   // Nekonečná smyčka
{
      _delay_ms(1000);                      // Zpoždění 1s
}
```

4.1.3.3 Obsluha přerušení

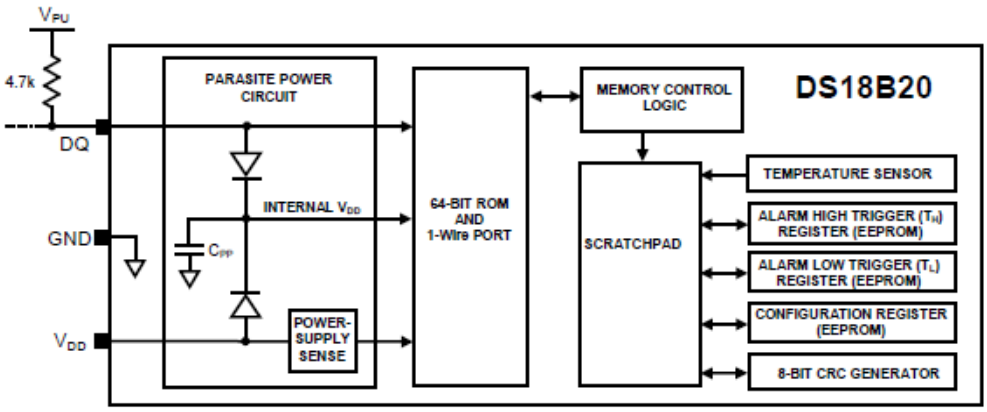
```
ISR(PCINT3_vect, ISR_NOBLOCK)               // Vektor přerušení
{
cli();                                      // Zakázení přerušení
_delay_ms(10);
if (TL5 == 0)                               // Kontroluje, jestli je zmáčknutý PIND4
{
      _delay_ms(300);
      if (TL5 == 0)                         // Kontroluje, jestli je pořád zmáčknuto TL5
      {
            nastaveni_data();
      }
      else
      {

      }
}
else
{
      cteni_data();
      vypsani_casu_na_displej();
}
      sei();                                // Povolení přerušení

}
```

4.2 Digitální teploměr - DS18B20

Tento digitální teploměr je v tří-pinovém pouzdru TO-92, kde jeden pin slouží jako společná zem pro napájení a datový signál. Druhý pin slouží jako datový a třetí pro napájení. K datovému pinu je třeba pro správnou funkčnost připojit pull-up rezistor (rezistor připojený ke kladnému pólu napájení). Napájecí napětí je od 3 do 5 V. Tento digitální teploměr měří ve stupních Celsia a lze nastavit na 9 až 12bitovou přesnost. Teploměr komunikuje s MCU pomocí jednovodičové komunikace. Operační teplota je od -55 °C až do +125 °C a jeho přesnost je ±0,5 °C při teplotě od -10 °C do +85 °C. Každý obvod DS18B20 má vlastní unikátní 64bitový sériový kód, který dovoluje, aby na společné datové lince mohlo být připojeno více těchto teploměrů.

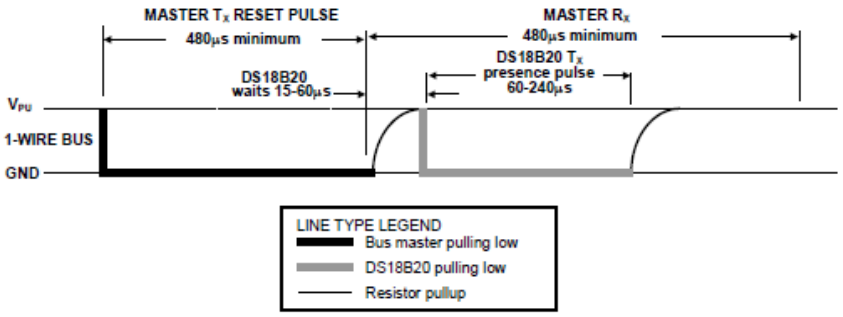


Obrázek 12 Blokový diagram digitálního teploměru DS18B20 [5]

4.2.1 Komunikace s MCU

4.2.1.1 Inicializace

Jak můžeme vidět na Obrázek 13 Zobrazení inicializační procedury v čase, tak pro správnou inicializaci je třeba nastavit na MCU pin, kterým je připojen k teploměru na výstup a poslat na něj logickou 0 na dobu alespoň 480 µs a poté přepnout na vstup a zapnout pull-up rezistor. Poté se počká 15 až 60µs a pak by měl teploměr stáhnout linku na logickou 0 na dobu 60 až 240 µs. Pokud k tomu takto dojde, tak je vše v pořádku a došlo k inicializaci senzoru. Takto lze přímo zjistit, zda je čidlo připojeno nebo není.



Obrázek 13 Zobrazení inicializační procedury v čase [5]

4.2.1.2 Zápis

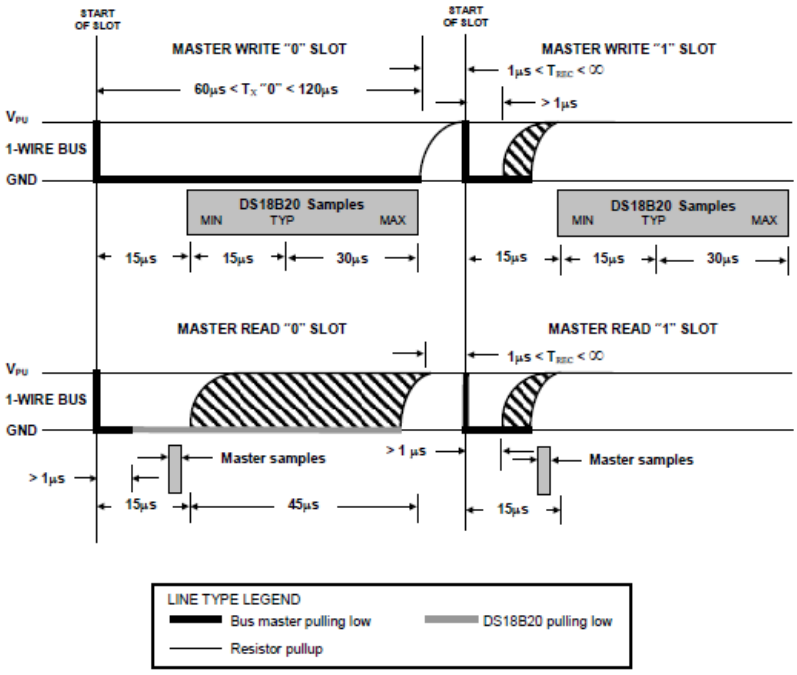
Pro zápis jsou zde dva sloty, jeden slouží k zápisu logické 0 a druhý k zápisu logické 1. Oba dva zápisy musí trvat alespoň 60 µs, s minimálním trváním zotavovací doby 1 µs. Oba dva sloty jsou inicializovány pomocí stáhnutí sběrnice na logickou 0.

Pro vygenerování zápisu logické 1 se po stáhnutí sběrnice na logickou 0 musí sběrnice do 15 µs uvolnit, čímž se přes pull-up rezistor sběrnice přepne na logickou 1.

Pro vygenerování zápisu logické 0 se po stáhnutí sběrnice na logickou 0 musí sběrnice uvolnit až po 60 µs, čímž se přes pull-up rezistor sběrnice přepne na logickou 1.

4.2.1.3 Čtení

Čas pro čtení musí trvat alespoň 60 µs a doba zotavení 1 µs. Slot pro čtení je inicializován pomocí MCU, které stáhne sběrnici na logickou 0 na minimálně 1 µs a poté uvolněním sběrnice (přepnutím pinu na čtení a zapnutí pull-up rezistoru). Senzor posílá logickou 1, pokud nechá sběrnici na logické 1 a posílá logickou 0, pokud stáhne sběrnici na logickou 0. Výstupní data ze senzoru jsou platná 15 µs po sestupné hraně, která inicializovala čtecí slot. Na Obrázek 14 Zobrazení čtení a zápisu v čase můžeme vidět čtení a zápis graficky.



Obrázek 14 Zobrazení čtení a zápisu v čase [5]

4.2.2 ROM Příkazy

Jakmile MCU detekuje přítomnost teploměru, může použít ROM příkaz. Tyto příkazy umožňují MCU zjistit kolik a jaká zařízení jsou připojena k 1vodičové sběrnici, jestli došlo k spuštění poplachu, ale i pro čtení a zápis dat.

4.2.2.1 Čti ROM – Read ROM (33h)

Tento příkaz je možné použít pouze v případě, kdy je na sběrnici připojen jeden obvod. Tento příkaz slouží k přečtení informace o rodině a sériového čísla, následuje CRC. Pokud se tento příkaz použije při více zapojených obvodech, tak dojde ke kolizi dat.

4.2.2.2 Porovnej ROM – Match ROM (55h)

V případě, že je připojeno více teploměrů dovoluje vybrat si jeden z připojených obvodů a pouze pokud přesně odpovídá 64bitovému ROM kódu, tak můžeme pokračovat předáním dalšího příkazu. Ostatní zařízení budou čekat na inicializaci.

4.2.2.3 Přeskoč ROM – Skip ROM (CCh)

MCU může pomocí tohoto příkazu přistupovat ke všem zařízením současně bez posílání ROM kódu. Například lze tímto způsobem spustit převod teploty na všech připojených teploměrech. Příkaz čti scratchpad může následovat po tomto příkazu, pouze pokud je připojeno pouze jedno zařízení.

4.2.2.4 Vyhledej ROM – Search ROM (F0h)

Jakmile se připojí napájení k MCU a jeho periferiím, musí MCU zjistit ROM kódy všech typů zařízení připojených na sběrnici. Pokud je připojen pouze jeden obvod, můžeme použít příkaz k přečtení ROM. Po ukončení identifikace jedné adresy je možné hned předat příkaz funkce paměti vyhledanému obvodu.

4.2.2.5 Vyhledej poplach – Alarm search (ECh)

Tento příkaz je podobný jako příkaz prohledej ROM s tím rozdílem, že vyhledá pouze obvody s nastavenými alarmy. Slouží pro identifikaci teploměrů, u kterých byl nastaven do aktivní úrovně klopný obvod horní, nebo dolní meze teploty.

4.2.2.6 Zapiš scratchpad – Write scratchpad (4Eh)

Umožňuje zapsání horní a dolní meze teploty a konfiguračního bytu do scratchpadu. Zápis se provádí od adresy 2. Po zapsání na adresu 3 nebo 4 je ukončen. Teploměr pak čeká na příchod inicializačního pulzu. Zápis lze ukončit kdykoliv vysláním inicializačního pulzu. Není nutné zapsat oba dva (nebo všechny tři) byty, ale pouze jeden.

4.2.2.7 Čti scratchpad – Read scratchpad (BEh)

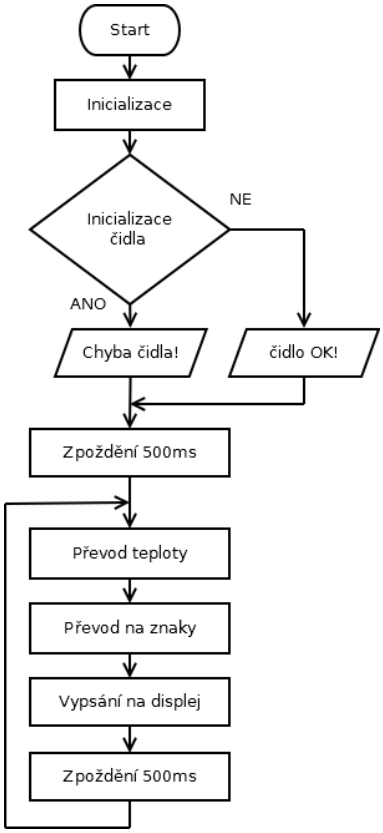
Instrukce umožní přečíst obsah scratchpaduod adresy 0 až po adresu 7. Poté teploměr vyšle CRC vyslaných dat, aby bylo možné ověřit jejich bezpečné přijetí. Přenos lze opět předčasně ukončit pomocí inicializace.

4.2.2.8 Převeď teplotu – Convert temperature (44h)

Instrukce začne převádět teplotu. Následuje převod, kde se výsledná teplota uloží do 2bytového registru. Při převodu je na sběrnici logická 0 a po ukončení převodu MCU přečte jedničku.

4.2.3 Měření teploty čidlem DALLAS

Tento program cyklicky měří teplotu, kterou dále zobrazuje na LCD displeji, úpravou programu lze měnit rozlišení teploměru od 9 do 12 bitů, kde 9 bitů odpovídá kroku 0,5 °C a s rostoucím rozlišením se krok zmenšuje (10b – 0,25 °C, 11b – 0,125°C, 12b – 0,0652°C).



Obrázek 15 Vývojový diagram programu pro měření teploty

4.2.3.1 Nastavení propojení s čidlem

V souboru ds18b20.h je potřeba nastavit následující:

```
#define DS18B20_PORT PORTC                    // Výběr portu
#define DS18B20_DDR DDC7                      // výběr registru pro změnu I/O
#define DS18B20_PIN PINC                      // Výběr vstupního portu
#define DS18B20_DQ PC7                        // Výběr vstupního pinu
```

Uvedené nastavení je pro pin PC7, lze libovolně měnit.

4.2.3.2 Funkce pro práci s čidlem

```
int16_t ds18b20_gettemp(uint8_t rozliseni);     // Tato funkce vrací 16bitovou hodnotu
teploty (nutno vynásobit konstantou 0,00625), vstupním parametrem lze změnit rozlišení
převodu a to od 9 do 12 bitů
uint8_t ds18b20_reset();                        // Funkce pro inicializaci čidla, funkce
vrací 0, když je čidlo v pořádku, 1 když se čidlo neozývá
```

4.2.3.3 Kontrola připojení/správné funkce čidla

```
if (ds18b20_reset())                            // Provede inicializace teplotního čidla
a pokud je v pořádku, tak vrátí 0, pokud není, tak 1
```

```
{
	lcd_puts("Chyba cidla!");		// Chybová hláška
}
else
{
	lcd_puts("Cidlo OK!");		// OK hláška
}

_delay_ms(500);
```

4.2.3.4 Převod teploty, vypsání na displej

Pro převod double hodnoty na pole

```
teplota = ds18b20_gettemp(9)*0.0625;		// Uložení hodnoty teploty do proměnné,
vstup funkce je počet bitů

prevod_teploty(teplota_char, teplota);		//  Do  proměnné  teplota_char  uloží
teplotu (znaky)

lcd_clrscr();					// Vymaže zobrazovaný text
lcd_puts("Teplota:");
lcd_gotoxy(0,1);
for (int i = 0; i < 5; i++)			// Vypsání teploty po znaku
{
	lcd_putc(teplota_char[i]);
}

lcd_putc(0);
lcd_putc('C');
```

4.3 LCD displej s řadičem HD44780

Tento displej je velmi oblíbený a existuje v různých variantách, které se liší barvou písma, počtem sloupců a řádků a typem podsvícení. Použitý typ má označení WC1602-STBLWNC06 a je to alfanumerický displej s dvěma řádky a šestnácti sloupci. Aktivní znaky mají bílou barvu a neaktivní modrou. Napájecí napětí pro obvody je 4,5 – 5,5 V, pro LCD ovladač je 4,7 V. Napájecí napětí prosvětlovací LED je typicky 3 V a typický proud diodou je pak 34 mA. Doba odezvy při teplotě 25 °C je maximálně 250 ms.

Obrázek 16 Displej WC1602-STBLWNC06 s programem pro zobrazení data a času

LCD displej umožňuje definici vlastních znaků, pokud potřebujeme diakritiku, nebo nějaké speciální znaky.

Tabulka 6 Popis vývodů LCD displeje [4]

Číslo pinu	Symbol	Funkce
1	VSS	zem (0V)
2	VDD	napájecí napětí
3	V0	nastavení kontrastu
4	RS	výběr registru
5	R/W	data čtení/zápis
6	E	povolovací vstup
7	DB0	datový vodič 0
8	DB1	datový vodič 1
9	DB2	datový vodič 2
10	DB3	datový vodič 3
11	DB4	datový vodič 4
12	DB5	datový vodič 5
13	DB6	datový vodič 6
14	DB7	datový vodič 7
15	A	napájení pro LED
16	K	0 V

4.3.1 Zobrazení dat na LCD displeji

Pro zobrazování dat na displeji slouží soubor hd44780.c, soubor hd44780_settings.h obsahuje nastavení vývodů pro propojení MCU a displeje a soubor hd44780.h obsahuje definice maker a funkčních prototypů funkcí. Při vypisování dat na displej je třeba mít na paměti, aby byla zobrazovaná

hodnota na displeji dostatečně dlouho dobu. Mezi výpisy je vhodné použít nějaké zpoždění, než se displej znovu přepíše dalšími daty.

4.3.1.1 Nastavení propojení s LCD displejem

Po otevření souboru hd44780_settings je třeba nastavit následující:

```
#define F_CPU                    16000000    // Zde je třeba zadat frekvenci
oscilátoru pro správné časování (16MHz)
#define LCD_BITS                 4           // 4 pro 4 bitový I/O mód, 8 pro 8 bitový
I/O mód
#define RW_LINE_IMPLEMENTED      0           // 0 když není třeba měnit mezi
zápisem/čtením (RW na LCD připojeno k zemi), 1 pro povolení RW pinu
#if (LCD_BITS==8)                            // Pokud se používá 8 bitový mód, musí se
nastavit DB0-DB7
#define LCD_DB0_PORT             PORTA
#define LCD_DB0_PIN              0
#define LCD_DB1_PORT             PORTA
#define LCD_DB1_PIN              1
#define LCD_DB2_PORT             PORTA
#define LCD_DB2_PIN              2
#define LCD_DB3_PORT             PORTA
#define LCD_DB3_PIN              3
#endif
#define LCD_DB4_PORT             PORTA       // Pokud se používá 4 bitový mód, musí se
nastavit DB4-DB7
#define LCD_DB4_PIN              0
#define LCD_DB5_PORT             PORTA
#define LCD_DB5_PIN              1
#define LCD_DB6_PORT             PORTA
#define LCD_DB6_PIN              2
#define LCD_DB7_PORT             PORTA
#define LCD_DB7_PIN              3
#define LCD_DISPLAYS             1           // Počet připojených displejů
#define LCD_DISPLAY_LINES        2           // Počet řádků displeje
#define LCD_E_PORT               PORTA       // Port pro E linku
#define LCD_E_PIN                5           // Pin pro E linku
```

4.3.1.2 Funkce pro práci s LCD displejem

```
void lcd_putc(char c);                  // Vypíše zadaný vstupní parametr typu char
void lcd_puts(const char *s);           // Vypíše zadaný řetězec (pole charů)
void lcd_init();                        // Inicializace LCD displeje, je nutno ji
provést alespoň jednou (start programu)
void lcd_command(uint8_t cmd);          // Zadávání příkazů pro LCD displej, například
pro definici vlastního znaku lcd_command (_BV(LCD_CGRAM)+0*8)

void lcd_clrscr();                      // Funkce pro vymazání displeje
void lcd_home();                        // Vypisování se přesune na počátek displeje
void lcd_goto(uint8_t pos);             // Pohyb v paměti
void lcd_gotoxy(uint8_t x, uint8_t y);  // Určení pozice v souřadnicích XY pro
vypisování
```

Pro vypisování dat na displej můžeme jednotlivé znaky vkládat do kódu za sebe, displej automaticky inkrementuje x-ovou pozici, jen je třeba nastavit správně začátek vypisování, případně ošetřit přetečení vypisování přechodem na 2. řádek.

4.4 Paměťová karta SD

Jedná se o elektronické zařízení, které slouží pro ukládání dat. Využívá paměti typu flash EEPROM a dnes se jedná o velmi používaný typ rozšiřující paměti pro mobilní zařízení. Existují 3 základní velikosti karet – SD, miniSD a microSD. Také existují 3 varianty SD karet, které poskytují různé velikosti úložiště. Karty SD mají pevně definovanou paměťových bloků na 512kB, ale je možné se setkat i s rozdílnými velikostmi bloků.

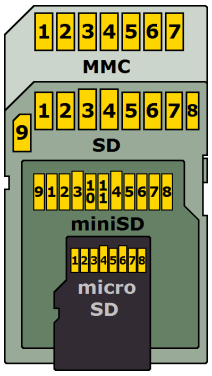
4.4.1 SD standarty

SDSC standart - umožňuje zápis dat na až 2GB velké datové úložiště

SDHC standart – umožňuje zápis dat na datové úložiště o velikosti od 2GB do 32GB

SDXC standart – umožňuje zápis dat na datové úložiště o velikosti 32GB až 2TB

Tyto standarty jsou zpětně kompatibilní, takže například na zařízení, schopném pracovat s SDHC můžeme pracovat i s SDSC, ale nemůžeme s SDXC (vyšší verze). Pro naše účely si vystačíme s kartou formátu SDSC, protože celkové velikosti souborů se bude pohybovat v řádu kB.



Obrázek 17- Rozmístění vývodů pro různé typy karet [13]

Tabulka 7 Popis vývodů SD karet [13]

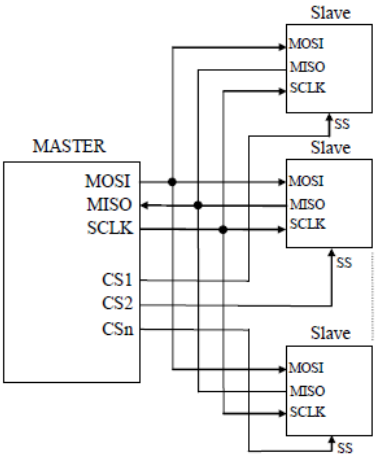
MMC Pin	SD Pin	miniSD Pin	microSD Pin	Jméno	Popis funkce (SPI)
1	1	1	2	CS	Výběr karty
2	2	2	3	DI	MOSI
3	3	3		VSS	Zem
4	4	4	4	VDD	Napájení
5	5	5	5	CLK	Hodinový signál
6	6	6	6	VSS	Zem
7	7	7	7	DO	MISO
	8	8	8	NC nIRQ	Nezapojeno Přerušení SDSC karta
	9	9	1	NC	Nezapojeno
		10		NC	Nezapojeno
		11		NC	Nezapojeno

4.4.2 Rozhraní SPI

Je to synchronní sériová komunikace na malé vzdálenosti často používaná při práci s MCU. Slouží ke komunikaci s externími periferiemi jako například digitálními potenciometry, displeji, přídavnými paměti a AD převodníky ale i ke komunikaci s dalšími MCU. Komunikace je typu Master – Slave, takže je jeden řídicí obvod Master (MCU) a může být několik podřízených zařízení Slave (periferie). Tento typ rozhraní je použit ke komunikaci s SD kartou.

Popis signálů:

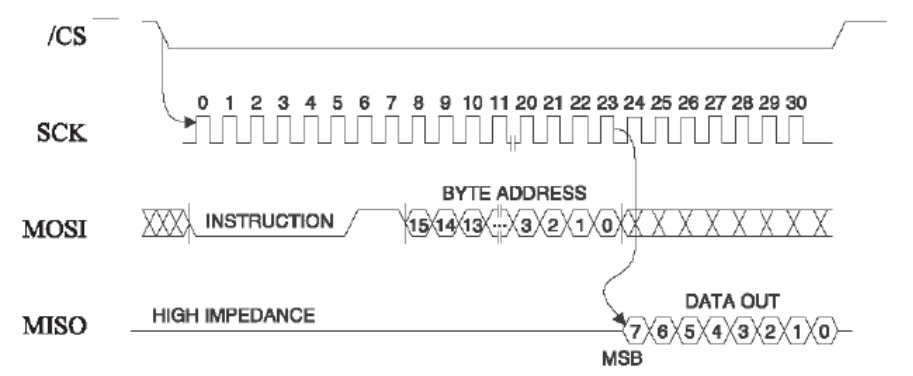
- CLK (SCK, SCLK) slouží jako hodinový signál, vysílá ho Master obvod, je připojen k CLK vstupu všech Slave obvodů.
- MOSI (SIMO) slouží jako výstup z řídicího obvodu a vstup do řízeného obvodu, je připojen k MOSI vstupu všech Slave obvodů.
- MISO (SOMI) slouží jako vstup do řídicího obvodu, výstup z řízeného obvodu, je připojen k MISO výstupu všech Slave obvodů.
- CS (SS, nCS, nSS) obsahuje ho každý Slave obvod, tímto signálem se vybírá obvod pro komunikaci. Pokud je CS neaktivní (log. 1), tak je SPI rozhraní daného obvodu neaktivní a jeho výstup MISO je ve stavu vysoké impedance. Aktivní úroveň je v log. 0. Ke každému Slave obvodu vede jeden samostatný signálový vodič z obvodu Master, tímto způsobem lze snadno vybrat Slave zařízení pro komunikaci. [9]



Obrázek 18 Zapojení zařízení pro SPI komunikaci [9]

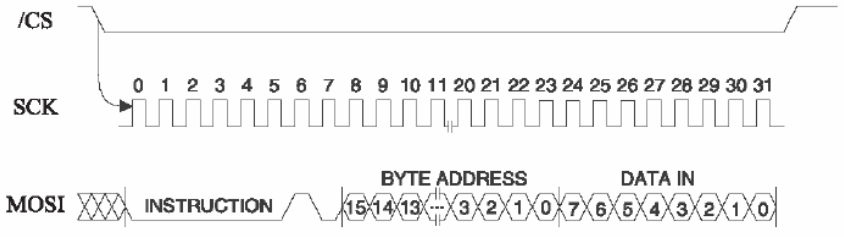
Přenos dat probíhá vždy mezi obvodem typu Master a některým obvodem Slave. Pokud tedy potřebujeme přenést data z jednoho Slave zařízení do druhého, tak musíme nejprve přečíst data z 1. Slave a uložit je do obvodu Master. Teď je můžeme zapsat z obvodu Master do 2. obvodu Slave.

Na Obrázek 19 SPI čtení je vidět příklad čtení dat po SPI sběrnici. Nejprve se musí nastavit CS signál na nulu, čímž se povolí obvod Slave a poté Master vyšle po MOSI lince instrukci pro čtení, poté pošle adresu, ze které chce číst a po jejím odeslání Slave začne posílat data po MISO lince.



Obrázek 19 SPI čtení [9]

Na Obrázek 20 SPI zápis můžeme vidět zápis dat, který se od čtení liší tím, že není třeba MISO linka, a po zaslání instrukce pro zápis a adresy na kterou se bude zapisovat, následují zapisovaná data.



Obrázek 20 SPI zápis [9]

4.4.3 Souborový systém FAT

Tento souborový systém a jeho modifikace jsou jedním z nejrozšířenějších souborových systémů. Dnes se s tímto souborovým systémem můžeme setkat především ve výměnných médiích jako například SD karty a USB flash disky a je podporován většinou přenositelných zařízení.

Disk se systémem FAT12 či FAT16 má tuto strukturu:

- Rezervované sektory (včetně boot sektoru)
- Tabulky FAT (většinou bývají dvě)
- Kořenový adresář
- Datová oblast (ostatní adresáře, soubory atd.)

Souborový systém FAT32 je rozdělen pouze na tři oblasti

- Rezervované sektory (včetně boot sektoru)
- Vlastní tabulky FAT (většinou bývají dvě)
- Datová oblast (adresáře, soubory. Včetně kořenového adresáře)

Jednotlivé oblasti nejsou zarovnány na clustery od začátku disku, jak by se mohlo zdát, ale na sektory. Tyto sektory se nacházejí hned na začátku diskového prostoru a obsahují podrobné informace o souborovém systému. Typicky se na FAT12/16 nachází jeden takový sektor – boot sektor – ve FAT32

těchto sektorů bývá třicet dva. Za rezervovanými sektory se ihned nachází tabulky FAT. Prvním rezervovaným sektorem je boot sektor. Jeho struktura je různá v závislosti na FATu, kterým je disk formátován. [11]

4.4.3.1 Tabulka FAT

Tabulka FAT je vlastně pole záznamů. Pro každý cluster z datové oblasti disku je zde jeden záznam, který udává stav tohoto clusteru. Velikost záznamů je 12 bitů pro FAT12, 16 bitů pro FAT16 a 32 bitů pro FAT32. První cluster datové oblasti se v tomto poli nachází pod indexem 2. [11]

4.4.3.2 Kořenový adresář

FAT12 a FAT16 mají vyčleněno speciální místo pro kořenový adresář. Kořenový adresář má tedy omezenou velikost. Kořenový adresář (stejně jako ostatní adresáře) se skládá ze záznamů o velikosti 32 bytů. Kořenový adresář ve FAT32 má naprosto stejnou strukturu jako jakýkoli jiný adresář. Jelikož se již nachází v datové oblasti (zaujímá hned první cluster, popř. i další, obsahuje-li hodně položek), jsou clustery, které jej obsahují, popsány v tabulce FAT. To dovoluje kořenovému adresáři obsahovat mnoho položek – jednotlivé clustery pak vytvoří spojový seznam. [11]

4.4.4 Ukládání do velkokapacitní externí paměti

Ukládání dat do externí paměti je vhodné pro zařízení jako třeba dataloggery, protože je ukládací médium (SD karta) vyměnitelné a jde tedy naměřit data, která se potom přenesou do počítače a mohou se vizualizovat.

4.4.4.1 Propojení SD karty s MCU

Propojení MCU s SD kartou je dáno vývody SPI rozhraní na MCU a toto propojení je dle Tabulka 8.

Tabulka 8 Propojení SD karty s MCU

SD karta	SPI	MCU
DAT0	MISO	PB6
DI	MOSI	PB5
CLK	CLK	PB7

4.4.4.2 Použité funkce

```
unsigned char getBootSectorData (void);                    // Funkce pro přečtení prvního sektoru karty, pokud je první sektor nepřítomen (nevložená karta, chybný souborový systém), tak vrátí 1, pokud je vše v pořádku, vrátí 0

void writeFile (unsigned char *fileName, unsigned char *text) // Funkce pro vytvoření a zápis do souboru, název souboru je omezen na formát maximálně 8.3 – 8 písmen pro název, oddělovací tečka a přípona
```

4.4.4.3 Program pro zápis textu do souboru

Za tímto jednoduchým programem se skrývá komunikace přes SPI, práce s SD kartou, práce se souborovým systémem FAT32 a vypisování postupu na displej. Úkolem tohoto programu je vytvořit soubor, do kterého se zapíše zadaný text. Výstupem tak bude soubor novy.txt, který obsahuje text „Toto je novy soubor“. Text je ukončen znaky \r a \n, což je nový řádek v systému Windows. Zápis je

nastavený tak, že dokud nepřijde znak \n, tak se zapisuje, až do hodnoty MAX_STRING_SIZE, která je nastavena na 70 a lze upravit v souboru FAT32.h.

```
const char ok[16] = {'S','o','u','b','o','r',' ','z','a','p','s','a','n','!',' ',' '};

unsigned char error;
init_devices();

error = getBootSectorData ();            // Přečte boot sektor a důležitá chybu uloží do
proměné error
if(error)
{
       lcd_clrscr();
       lcd_puts("FAT32 nenalezen!");     // Nenalezeno FAT32 kompatibilní zařízení
}
else
{
       writeFile("novy.txt", "Toto je novy soubor\r\n");     // Zápis souboru
       lcd_clrscr();
       lcd_gotoxy(0,1);
       lcd_clrscr();
       lcd_puts(ok);
}
```

4.4.4.4 Zápis na SD

V následujícím úryvku kódu je uveden zápis do jednotlivých bloků SD karty, který se nachází v souboru FAT32.c.

```
do
   {
         data = text[k++];
         buffer[i++] = data;
         fileSize++;

        if(i >= 512)
        {
          i=0;
          error = SD_writeSingleBlock (startBlock);
          j++;
          if(j == sectorPerCluster) {j = 0; break;}
          startBlock++;
          poprve = 1;
        }
       }while ((data != '\n') && (k < MAX_STRING_SIZE));     //udává   počet   zapsaných
znaků

if((data == '\n') || (k >= MAX_STRING_SIZE))
{
       i--;
       for(;i<512;i++)                          //zaplní zbytek bufferu
       buffer[i]= 0x00;
       error = SD_writeSingleBlock (startBlock);
       break;
}
```

5 Programování MCU

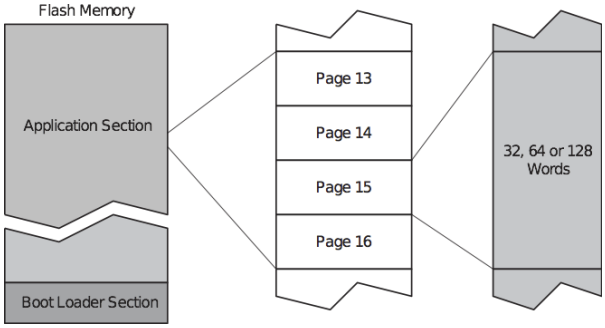
5.1 Programování přes USB

Atmel Studio bohužel nepodporuje přímé programování vývojové desky přes USB a tak je třeba nahrát bootloader a dále používat software k nahrávání programů přes USB. Tato volba však neumožňuje debuggování aplikace a tak se příliš nehodí k vývoji aplikací.

5.1.1 Bootloader

Bootloader je speciální aplikace, sloužící k nahrání aplikace nebo operačního systému do paměti a jejího následného spuštění. V počítačích je BIOS, který spustí bootloader a ten pak načte z požadovaného média jádro operačního systému do paměti a následně jej spustí. Ve světě MCU se po zapnutí napájecího napětí (a pokud jsou správně nastaveny pojistky) začne okamžitě vykonávat kód bootloaderu, který je uložen na konci paměti flash. Tento bootloader může např. přes rozhraní UART načíst aplikaci, kterou postupně uloží do paměti flash a poté skočí na její první instrukci. Není tedy potřeba žádný externí programátor.[10]

Z pohledu booloaderu se paměť MCU rozdělí na 2 části. Na část pro bootloader a část pro aplikace. Tato paměť se dále dělí na stránky s pevně danou velikostí.

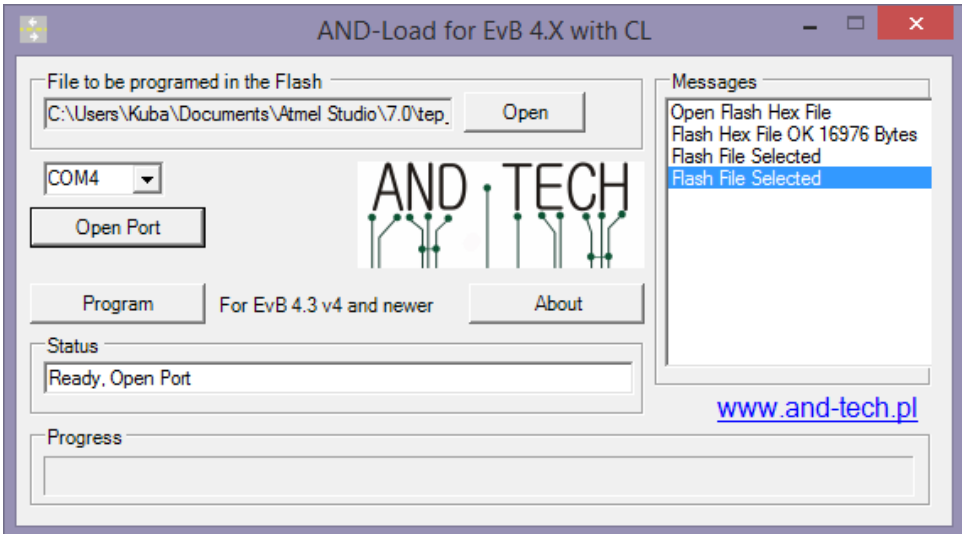


Obrázek 21 Rozdělení paměti MCU z pohledu bootloaderu [10]

Anglický manuál pro nahrání bootloaderu do MCU na vývojové desce EvB 4.3 v3 je v přiloženém CD.

5.1.2 Nahrávání programu

Pro nahrávání programu do MCU slouží program AND-Load. Pro nahrání je třeba mít zapojen vývojový kit v počítači a až poté zapnout program, jinak by program nenašel požadovaný port. Jako vstup je zde již zkompilovaný hex soubor, takže je potřeba nějaké vývojové prostředí v kterém se kód nejprve zkompiluje.



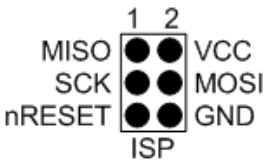
Obrázek 22 Program AND-Load

5.2 Programování pomocí programátoru JTAGICE3

Tento programátor je od firmy Atmel a je samozřejmě podporován Atmel Studiem, v kterém lze přímo programovat a debuggovat. Tento programátor umožňuje programovat přes různé typy rozhraní, mezi nejpoužívanějšími SPI, které neumožňuje debuggování, JTAG, podporující debuggování a dále PDI a aWire. Pro programování pomocí JTAG rozhraní je třeba zkontrolovat, zda je programování přes JTAG povoleno, pokud není, tak je třeba se k MCU nejdříve připojit přes SPI a nastavit správně pojistky.

5.2.1 Propojení programátoru a MCU přes rozhraní SPI

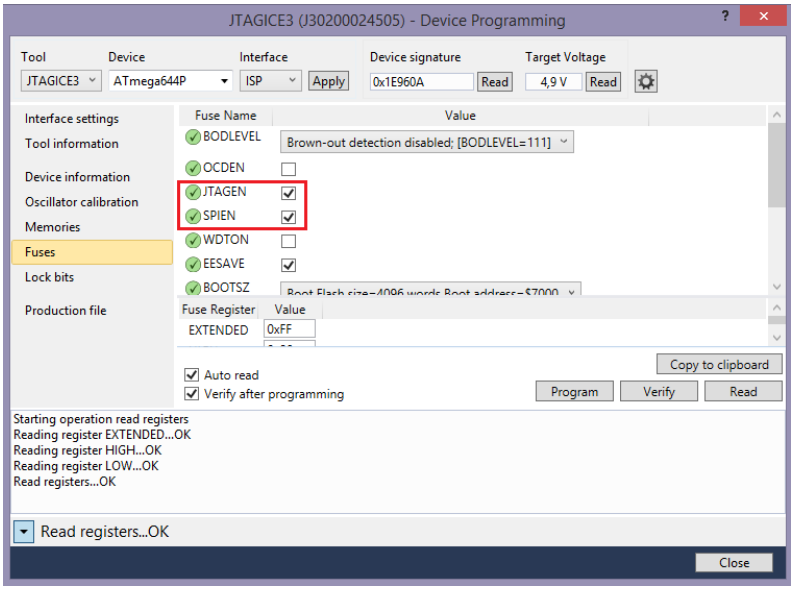
SPI rozhraní programátoru a vývojové desky nejsou kompatibilní. Je třeba propojení udělat zvlášť, například propojovacími vodiči. Liší se v umístění pinu MOSI, který je na desce umístěn o 2 pozice níže pod resetovacím pinem a při přímém propojení by byl připojen na GND. Na SPI rozhraní od programátoru je MISO vývod označen bílou tečkou.



Obrázek 23 Zobrazení vývodů SPI rozhraní programátoru JTAGICE3 [14]

Po propojení je třeba zkontrolovat v Atmel Studiu nastavení pojistek, zda je povolené programování přes JTAG a SPI nechat zaškrtnuté.

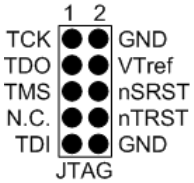
Varování: Při špatné manipulaci s pojistkami se může zablokovat obvod proti zápisu a nebude již možné tento obvod programovat.



Obrázek 24 Okno Device Programming

5.2.2 Propojení programátoru a MCU přes rozhraní JTAG

Výhodou tohoto rozhraní je možnost debuggování aplikací. Za pomoci breakpointů lze zastavit běh programu a zkontrolovat si hodnoty uložené v registrech, proměnných, sledovat stav na jednotlivých portech apod.



Obrázek 25 Zobrazení vývodů JTAG rozhraní programátoru JTAGICE3 [14]

6 Závěr

V bakalářské práci jsem se seznámil s programováním mikrokontrolérů značky Atmel a programovacím studiem Atmel Studio. Měřil jsem teplotu pomocí digitálního teploměru a provedl jsem teoretický rozbor komunikace po jednovodičové sběrnici. Dále jsem uvedl jednotlivé příkazy pro ovládání digitálního teploměru. Zjistil jsem jak zobrazovat data na LCD displeji, jak zobrazit uživatelsky definované znaky a možnost snížení počtu datových vodičů z 8 na 4. Provedl jsem digitalizaci analogového signálu za pomocí integrovaného AD převodníku, využil jsem hardwarového přerušení pro aplikaci s hodinami reálného času, kdy se aktualizovala hodnota času na displeji v přesně daných intervalech. Řešil jsem také ukládání a nastavení data a času a uložení hodnoty roku do paměti hodin reálného času, neboť při výpadku napájení celé desky, by došlo ke ztrátě nastaveného roku. Vytvořil jsem zadané programy pro práci s jednotlivými periferiemi se stručným popisem funkce a možností změnit nastavení pro použití jiných vývodů.

Na závěr jsem vytvořil aplikaci dataloggeru, která kombinuje všechny programy a tento datalogger zobrazuje datum, čas, teplotu a změřené napětí. Také je zde možnost ukládat data ve formátu csv, načíst je na počítači například v excellu a vytvořit si graf změny teploty nebo napětí během měřeného časového úseku.

Tato bakalářská práce může sloužit jako návod pro začínající programátory mikrokontrolérů, není zde však uveden základ programování v jazyce C, takže je třeba, aby měl případný zájemce alespoň základní znalosti v programování v jazyce C. Vytvořené programy ve formě projektu pro Atmel Studio jsou dostupné v přiloženém CD.

Vývojová deska obsahuje ještě několik periferií, takže je možné vytvořit, nebo modifikovat vytvořené aplikace. Například by se vybízelo měření dvou teplot přidáním externího digitálního teploměru, kde by se externí teploměr připojil k chladiči výkonového spínacího prvku a podle teploty by se pomocí obvodu pro řízení stejnosměrných motorů, obsaženého na desce, regulovaly otáčky ventilátoru. Dále by bylo možné vyměnit přítomný LCD displej za větší a sestrojit zařízení pro čtení souborů, také by se tak dalo zobrazovat více dat zároveň.

Pokud umíme změřit napětí, umíme tím pádem i měřit proud za pomoci bočníku. Velikost rozsahu měřeného napětí lze měnit velikostí referenčního napětí, nebo děličem napětí. Vynásobením napětí a proudu získáme činný výkon, takže z hlediska měření obvodových veličin, by bylo možné toto měření ještě rozšířit.

Seznam použité literatury

1. ATmega164P/V-ATmega324P/V-ATmega644P/V: Datasheet. *Atmel* [online]. [cit. 2016-04-27]. Dostupné z: http://www.atmel.com/images/atmel-8011-8-bit-avr-microcontroller-atmega164p-324p-644p_summary.pdf

2. Návod k použití vývojového kitu EvB 4.3 v4. *AND-TECH* [online]. [cit. 2016-04-27]. Dostupné z: http://www.vo.gme.cz/dokumentace/752/752-445/czn.752-445.1.pdf

3. PCF8583: Datasheet. *NXP* [online]. [cit. 2016-04-27]. Dostupné z: http://www.nxp.com/documents/data_sheet/PCF8583.pdf

4. WC1602A: Datasheet. *Wincom Tech* [online]. [cit. 2016-04-27]. Dostupné z: http://osworld.pl/wp-content/uploads/WC1602A-STBLWHTC-06.pdf

5. DS18B20: Datasheet. *Maxim Integrated* [online]. [cit. 2016-04-27]. Dostupné z: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf

6. HRBÁČEK, Jiří. *Komunikace mikrokontroléru s okolím*. 1. vyd. Praha: BEN - technická literatura, 1999, 159 s. ISBN 80-86056-42-2.

7. HRBÁČEK, Jiří. *Komunikace mikrokontroléru s okolím*. 1. vyd. Praha: BEN - technická literatura, 2000, 151 s. ISBN 80-86056-73-2.

8. MANN, Burkhard. *C pro mikrokontroléry: ANSI-C, kompilátory C, spojovací programy - linkery, práce s ATMEL AVR a MSC-51, příklady programování v jazyce C, nástroje pro programování, tipy a triky* .. 1. české vyd. Praha: BEN - technická literatura, 2003, 275 s. µC & praxe. ISBN 80-7300-077-6.

9. PALACKÝ, Petr. *Mikropočítačové řídicí systémy I*. Ostrava: Vysoká škola báňská - Technická univerzita, 2007. ISBN 978-80-248-1494-0.

10. *Bootloader v AVR* [online]. [cit. 2016-04-27]. Dostupné z: http://uart.cz/721/bootloader-v-avr/

11. Souborové systémy FAT [online]. 2008 [cit. 2016-04-27]. Dostupné z: http://www.secit.sk/content/souborove-systemy-fat

12. VÁŇA, Vladimír. *Mikrokontroléry ATMEL AVR - Programování v jazyce C: popis a práce ve vývojovém prostředí CodeVisionAVR C*. 1. vyd. Praha: BEN - technická literatura, 2003. ISBN 80-7300-102-0.

13. Secure Digital. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2016-04-27]. Dostupné z: https://en.wikipedia.org/wiki/Secure_Digital

14. Atmel-ICE. *Atmel* [online]. [cit. 2016-04-27]. Dostupné z: http://www.atmel.com/webdoc/atmelice/index.html

15. Gregoriánský kalendář. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2016-04-27]. Dostupné z: https://cs.wikipedia.org/wiki/Gregori%C3%A1nsk%C3%BD_kalend%C3%A1%C5%99

Přílohy

A. Obsah CD

- Bakalářská práce ve formátu pdf
- Anglický manuál pro nahrání bootloaderu ve formátu pdf
- Projekt s návodem pro práci s LCD displejem
- Projekt s návodem pro práci s AD převodníkem
- Projekt s návodem pro práci s teplotním čidlem DS18B20
- Projekt s návodem pro práci s obvodem reálného času PCF8583
- Projekt s návodem pro ukládání dat na SD kartu
- Projekt dataloggeru