Vysoké učení technické v Brně
Fakulta elektrotechniky a komunikačních technologií
Magisterský studijní obor Elektronika a sdělovací technika

SEMESTRÁLNÍ PROJEKT 2 (MM2E)

Modul pro monitorování teploty

...rozpracováno...

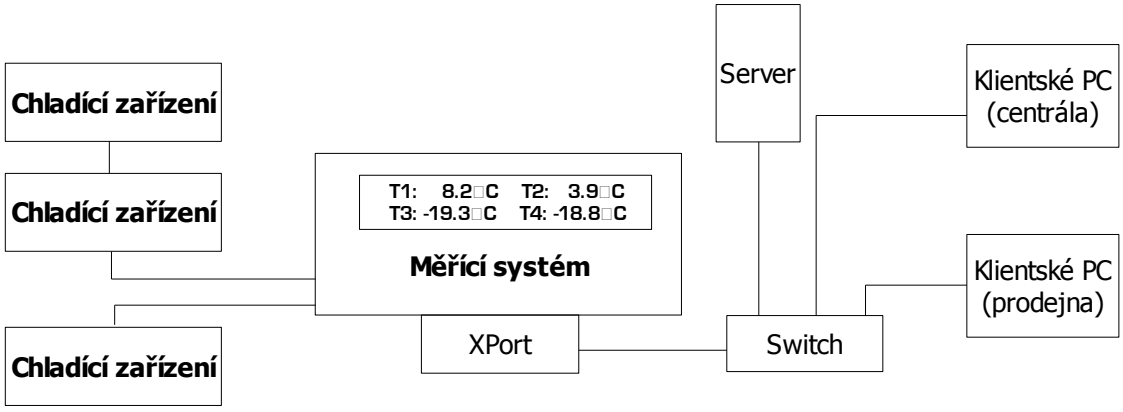
2006/2007 **Jaroslav Martínek**

Obsah

1. Úvod

Skladování potravin v obchodních a výrobních řetězcích je věcí velmi složitou a je řízeno patřičnými zákony a předpisy. V největší míře se to týká potravin chlazených a mražených, kde je potřeba dodržet skladovací parametry, v tomto případě hlavně teplotu v zákonem přesně stanovených mezích. Jelikož provozovatel takovéhoto skladu či obchodu přebírá při převzetí tohoto zboží i odpovědnost za jeho správné skladování, je velmi důležité mít aktuální přehled o stavu a teplotách v použitých mrazících a chladících zařízeních kvůli odpovědnosti za zboží ať už vůči koncovým zákazníkům či kontrolním orgánům, které tak mohou mít na vyžádání kompletní a přesné přehledy o průběhu skladování zboží. A také provozovatel se takto okamžitě dozví např. o poruše některého z provozovaných zařízení.

Celé zařízení by mělo být postaveno na tzv. „embedded-systému“, tedy autonomním mikropočítači, který zajistí samostatný provoz monitorovací části systému i v případě výpadku ostatních podpůrných částí a bude schopen bateriového provozu pro případ výpadku elektrické energie. Informace o aktuálním stavu měření a základní nastavení bude možné provádět přímo na základní jednotce pomocí alfanumerického displeje. Zařízení bude dále připojeno do místní sítě typu Ethernet, kdy bude poté možno jej ovládat a zjišťovat podrobnější informace přes WWW rozhraní. Zpracování naměřených dat je možno realizovat dvěma způsoby, buďto zasíláním na Linuxový server a ukládáním do databáze SQL typu (nejspíše MySQL), což zajistí široké možnosti zpracování, různé přehledy, vynášení do grafů atd. Nespornou výhodou této možnosti je také možnost přístupu k datům z různých míst současně, např. z místní pobočky a i z centrály skladu. Druhou možností je klientský program běžící na PC obsluhy (někde v kanceláři), kterým by se stahovala naměřená data z centrální jednotky a který by uchovávala dále zpracovával naměřené hodnoty.



Obr. 1.1 – Návrh koncepce systému

2. 1-wire sběrnice

2.1 Obecné informace

1-wire je systém sběrnice pocházející z dílen společnosti Dallas Semiconductors (nyní pod Maxim), který poskytuje nízkorychlostní přenos dat, signálů a napájení po jednom vodiči (plus druhý zemnící vodič). 1-Wire pracuje na stejném konceptu jako sběrnice I2C, ovšem s menší datovou rychlostí a daleko nižší cenou. Používána bývá převážně pro komunikaci s malými a levnými zařízeními jako např. digitální teploměry a jiné senzory.

Jak již bylo zmíněno, jednou z nejdůležitějších vlastností této sběrnice je, že zařízení k ní připojené potřebují pouze dva vodiče – datový a zem. Toho bylo docíleno tím, že zařízení obsahují 800pF kondenzátor, který zajišťuje jejich napájení z datového vodiče.

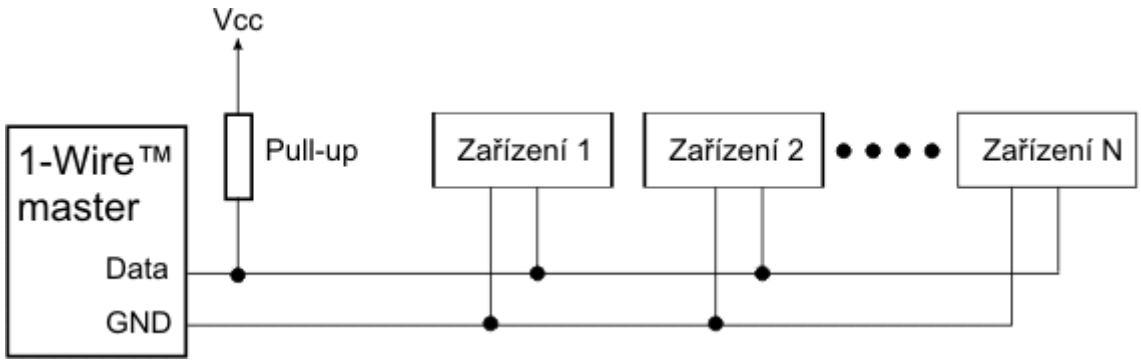
Takto může být velmi jednoduše sestaven systém senzorů, kdy každý obsahuje vlastní čidlo a logiku potřebnou pro komunikaci po sběrnici.

1-Wire zařízení bývají většinou integrované obvody umístěné v běžných pouzdrech (TO92,...), pripojení na sběrnici bývá realizované pomocí modulárních konektoru (RJ11, RJ45,...) a patřičné kabeláže.

iButton je druhou možností připojení 1-Wire zařízení. Je to vlastně jiný standard mechanického pouzdra, kdy je určitý 1-Wire komponent umístěn v kovovém „knoflíku“ (odtud iButton), vypadajícím podobně jako známe lithiové baterie např. do hodinek. Takováto zařízení se ke sběrnici připojují pomocí adaptéru.

Obr. 2.1 – Možná provedení iButton

Hardwarově je implementace protokolu řešena speciálním software v master zařízení a rezistorem. Skrze něj je na sběrnici přivedeno +5V, což zajistí napájení slave zařízení. Master zařízením může v tomto případě být téměř jakékoliv PC nebo mikropočítač. Samozřejmě jsou také k dispozici různé řadiče a převodníky, realizující protokol přímo hardwarově.



Obr. 2.2 – Zapojení zařízení na sběrnici

2.2 Komunikace po sběrnici

Master zařízení začíná sekvenci bitů „reset“ pulsem, který na 480us vyšle na sběrnici 0V. Tímto dojde k resetu všech slave zařízení v podstatě jejich odstřižením od napájení. Po tomto pulsu, tedy po opětovném zapojení napájení, o sobě dá každé přítomné slave zařízení vědět tzv. pulsem přítomnosti – přizemní sběrnici na alespoň 60us po jejím uvolněním masterem.

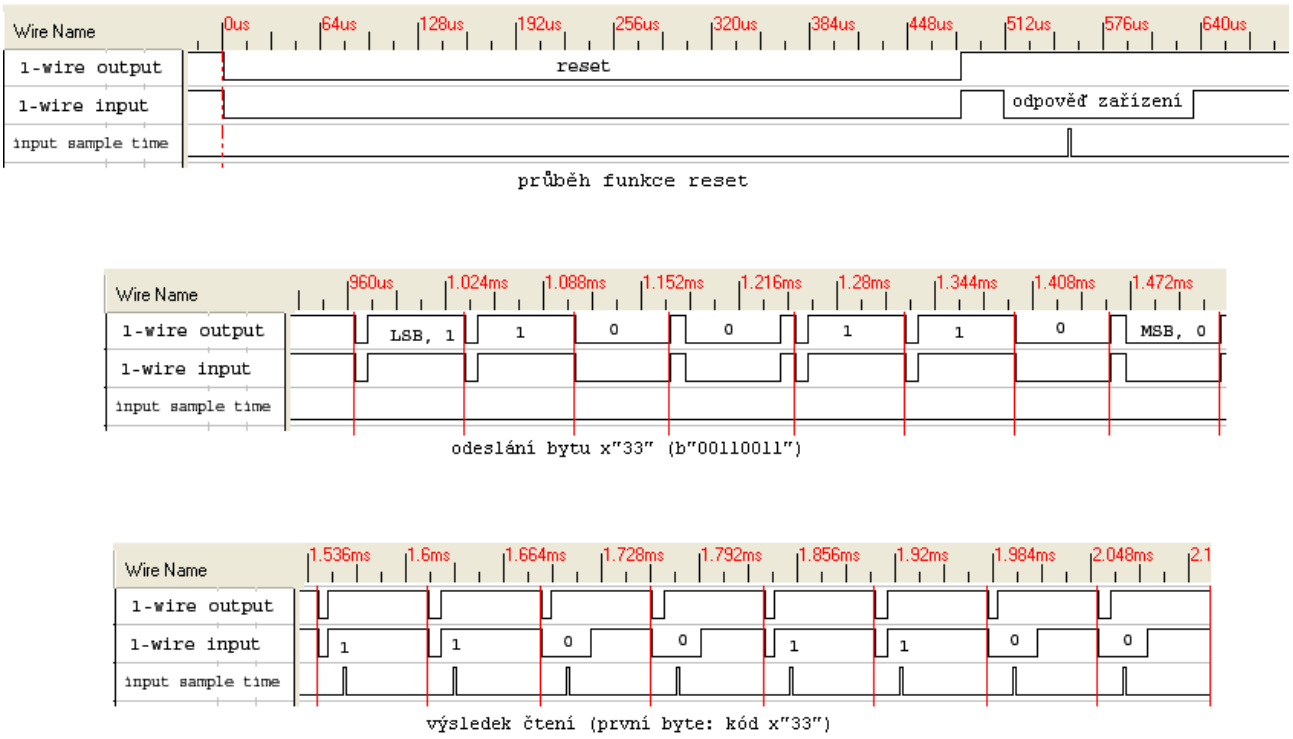
Pro vyslání logické 1 software masteru vyšle krátký 0V puls (1-15us), pro vyslání logické 0 software odešle 60us dlouhý 0V puls. Sestupná hrana těchto pulsů způsobí spuštění monostabilního klopného obvodu ve slave zařízení, který funguje jako časovač, čímž je zajištěno, aby slave četl data ze sběrnice po dobu přibližně 30us. Klopný obvod má samozřejmě své tolerance, takže časování není příliš přesné. Kvůli tomu je potřeba, aby výstupní pulsy byly dlouhé 60us pro logickou 0 a nebyly delší než 15us pro logickou 1.

Při příjmu dat vyšle master 1-15us dlouhý 0V puls na startu každého bitu. Pokud chce slave zařízení vyslat log. 1, nedělá nic a na sběrnici se tak hned oběví napájecí napětí. Pokud chce slave odeslat log. 0, přidrží sběrnici na 0V po dobu 60us.

Základní komunikační sekvence tedy vypadá tak, že je vyslán reset puls následovaný 8-bitovým příkazem. Potom jsou odesílána nebo přijímána data ve skupinách po osmi bitech.

Většina zařízení může sdílet jednu sběrnici. Pro tento účel má každé z nich jedinečné 64 bitové seriové číslo. Nejvyšší bit (MSB) tohoto seriového čísla je osmibitové číslo, které udává typ zařízení. Nejnižší bit je pak klasický osmibitový kontrolní součet (CRC).

V definici komunikace existuje také několik všeobecných vysílacích příkazů a příkazů adresovaných pouze pro určité zařízení. Master může odeslat příkaz pro výběr následovaný adresou příslušného zařízení, následující příkaz pak bude proveden přímo tímto zařízením.



Obr. 2.3 – Průběh komunikace po sběrnici

Příkazy SEARCH a ALARM SEARCH

Většina 1-wire zařízení implementuje příkaz SEARCH (kód 0xF0). Zařízení, která mohou být v nějakém "poplachovém stavu" implementují i příkaz ALARM SEARCH (kód 0xEC). Tento příkaz se od předchozího liší pouze tím, že na něj reagují zařízení, která jsou momentálně v "poplachovém stavu" (teplotní čidlo, u něhož je překročena nastavená hranice teploty apod.), což umožňuje rychleji zjistit, které zařízení má být obslouženo.

Zařízení, které přijme SEARCH příkaz, odpoví tím, že vyšle první (nejnižší) bit svého 64bitového kódu. Je-li na sběrnici víc oslovených zařízení, odpoví všechny naráz. Jak vyplývá ze specifikace 1-wire sběrnice (zařízení jsou připojena paralelně ke společnému vodiči výstupem s otevřeným kolektorem), výsledkem je logický součin (AND) všech bitů. Po vyslání tohoto bitu požádá master o vyslání dalšího bitu. Zařízení na tento požadavek odpoví negací prve vyslaného bitu. Z těchto dvou přijatých bitů lze odvodit situaci na sběrnici. Mohou nastat čtyři různé možnosti:

První bit	Druhý bit	Situace
0	0	Na sběrnici je více zařízení. Tato zařízení mají na tomto místě ve svých kódech různé hodnoty. Došlo k neshodě.
0	1	Na sběrnici je jedno či více zařízení. Tato zařízení mají na tomto místě ve svých kódech bit s hodnotou 0.
1	0	Na sběrnici je jedno či více zařízení. Tato zařízení mají na tomto místě ve svých kódech bit s hodnotou 1.
1	1	Na sběrnici není žádné zařízení, které by reagovalo na příkaz SEARCH.

Master nyní vyšle jeden potvrzovací bit. Dále se budou vyhledávání účastnit pouze zařízení, která mají na prvním místě bit s hodnotou stejnou jako vyslal master.

Pokud mají všechna zařízení na dané pozici stejnou hodnotu bitu, vyšle master tuto hodnotu. Pokud mají různé hodnoty (tzn. nastala první možnost - oba přečtené bity byly nulové), musí si master poznamenat, na které pozici došlo k neshodě, a vyšle buď 1 nebo 0. Zda master vyšle 0 nebo 1 určí na základě předchozích hledání, především pozice poslední nalezené neshody. Dále popsaný algoritmus vysílá nejprve nulu a při druhém průchodu vysílá jedničku.

Tento postup se opakuje, dokud není načteno všech 64 bitů identifikace. Pokud při načítání došlo k neshodě v nějakém bitu, znamená to, že je na sběrnici více zařízení a celý postup se opakuje s tím, že na pozici poslední neshody se nyní vysílá bit opačné hodnoty (prochází se "druhá větev").

Popis algoritmu

Procházení sběrnice sestává z opakovaných hledání jednotlivých zařízení. V každém cyklu je nalezeno jedno zařízení. V algoritmu je třeba si neustále udržovat informaci o kódu posledního nalezeného zařízení a o pozici neshody (zdali k nějaké došlo nebo ne anebo je nalezené zařízení na sběrnici samo).

Algoritmus pracuje s následujícími proměnnými:

id_bit_number - Pořadové číslo bitu, který je právě prohledáván (1 - 64).

id_bit - Bit přečtený jako první. Jeho hodnota je logický součin všech bitů na pozici id_bit_number ze všech zařízení, které se účastní vyhledávání.

cmp_id_bit - Bit přečtený jako druhý, doplněk id_bit.

LastDiscrepancy - Hodnota udávající pořadové číslo bitu, u něhož došlo při posledním hledání k neshodě.

LastDeviceFlag - Příznak posledního nalezeného zařízení.

Last_Zero - Hodnota udávající pořadové číslo bitu, u něhož došlo při aktuálním hledání k neshodě, ze které bylo pokračováno odpovědí "0".

ROM_NO - Buffer o velikosti 64 bitů (8 byte), který obsahuje aktuálně načítané číslo zařízení.

search_direction - Bitová proměnná, udávající směr dalšího hledání. Zařízení, která mají bit na pozici id_bit_number roven této hodnotě se budou dále účastnit prohledávání, ostatní zařízení přestanou komunikovat až do vyslání RESET pulsu.

Před prvním hledáním (funkce FIRST) je proměnná LastDiscrepancy vynulována. Po volání hledací rutiny je v ROM_NO číslo nalezeného zařízení. Pokud není žádné zařízení nalezeno, je hledání ukončeno s chybou.

Hledání dalšího zařízení (funkce NEXT) využívá hodnoty z prvního hledání, uložené v proměnných ROM_NO a LastDiscrepancy. "Hledací" rutina vrátí v ROM_NO číslo dalšího zařízení, nebo, bylo-li předcházející nalezené zařízení posledním, vrátí chybový kód.

Hledání jednotlivých zařízení začíná tím, že master vyšle RESET puls a čeká na potvrzení PRESENCE pulsem. Pokud přijde, hledání může pokračovat a jsou nastaveny proměnné: id_bit_number (pozice zpracovávaného bitu) na hodnotu 1 a last_zero (pozice poslední odpovědi "0" na zjištěnou neshodu) na hodnotu 0. Poté je vyslán příkaz SEARCH nebo ALARM SEARCH. Následuje smyčka, v níž je skládán obsah ROM_NO. Tato smyčka se opakuje 64krát, tedy pro každý bit identifikačního kódu.

Ve smyčce jsou nejprve přečteny dva bity (zařízení vysílá vždy hodnotu bitu a její doplněk), viz výše. Pokud se od sebe oba přijaté bity liší (což znamená, že všechna zařízení na dané pozici mají stejnou hodnotu bitu), je hodnota prvního zapsána do ROM_NO na pozici danou id_bit_number a jeho hodnota je zároveň zapsána do search_direction. Pokud jsou oba jedničkové, znamená to, že žádné zařízení na hledání neodpovídá a funkce končí. Pokud jsou oba nulové, znamená to, že je třeba se rozhodnout, jestli bude vyslána nula nebo jednička. Rozhoduje se na základě proměnných LastDiscrepancy a id_bit_number takto:

LastDiscrepancy vs. id_bit_number	**Reakce**
id_bit_number > LastDiscrepancy	Last_zero = id_bit_number search_direction = 0
id_bit_number == LastDiscrepancy	search_direction = 1
id_bit_number < LastDiscrepancy	search_direction = (ROM_NO & (1<< id_bit_number) > 0) (search_direction je nastavena na hodnotu bitu na stejné pozici v předchozím nalezeném ROM_NO)

Následně je vyslána hodnota bitu search_direction, tento bit je zapsán do ROM_NO na odpovídající pozici a je zvýšeno počítadlo id_bit_number. Pak se celá smyčka opakuje.

Po zjištění všech bitů je do proměnné LastDiscrepancy zapsána hodnota proměnné LastZero. Pokud je tato hodnota nulová, znamená to, že je nalezené zařízení poslední.

3. Teplotní čidlo DS1820

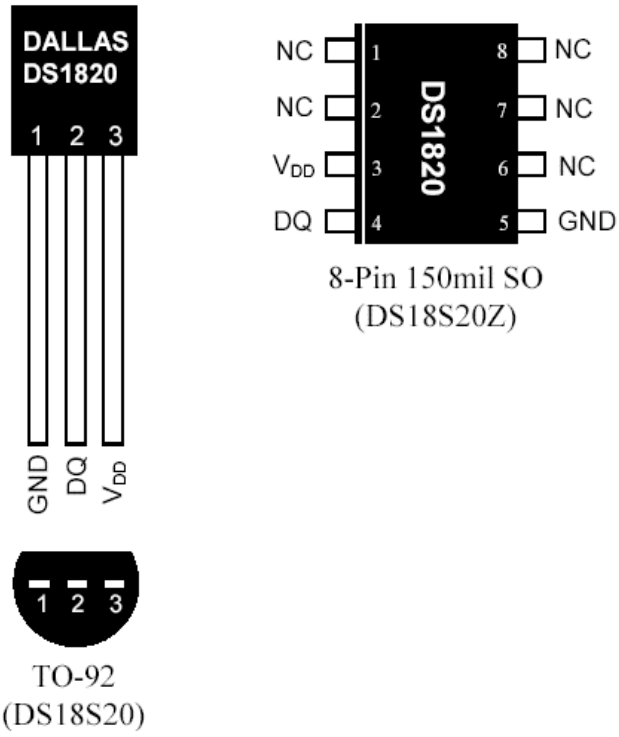
3.1 Základní vlastnosti

DS1820 je 1-wire teplotní senzor (v podstatě samostatný teploměr) taktéž od Dallas Semiconductors, který poskytuje 9-bitové měření teploty ve stupních Celsia. Obsahuje také funkci alarmu s uživatelsky programovatelným horním a dolním limitem. Jelikož výstup čtený ze senzoru (tedy výstupní teplota) je v digitální podobě, jeho spojení s mikroporcesorem je vcelku jednoduché.

Základní vlastnosti tohoto teploměru:

- Jelikož pracuje na 1-wire sběrnici, potřebuje jen jeden datový vodič, což značně zjednodužšuje připojení
- Díky schopnosti vícebodového připojení je snadnější návrh zařízení pro měření teploty
- Nepotřebuje žádné další externí součástky
- Rozsah měřených teplot je od –55°C do +125°C (–67°F až +257°F)
- V rozsahu –10°C až +85°C je udávaná přesnost ±0.5C
- 9-bitové rozlišení
- Napájecí napěti v rozsahu 3V – 5,5V
- Doba převodu teploty je 750ms

(v současnosti se typ DS1820 díky chybě některých sérií nevyrábí a je nahrazen novějším DS18S20, což ovšem nemá pro naši aplikaci žádný význam)

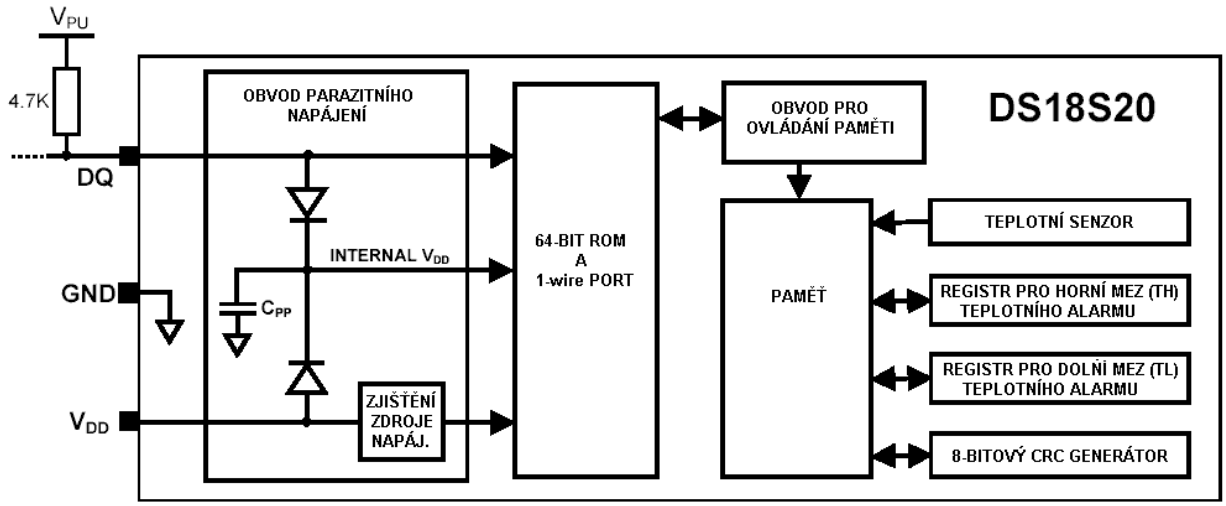


Obr. 3.1 – Možná provedení teploměru DS18S20

POUZDRO		SYMBOL	POPIS
8-PIN SOIC	TO-92		
5	1	GND	Zem
4	2	DQ	Datový vstup/výstup. Pin pro připojení k 1-wire sběrnici. Také se používá k napájení čidla v případě použití parazitního napájení.
3	3	V_{DD}	Volitelný napájecí pin. Musí být uzeměný pro použití parazitního napájení.

3.2 Popis funkce

Na obrázku 4 je nakreslen blokový diagram teplotního čidla DS18S20 a popis pinů je uveden v tabulce výše. V 64-bitové ROM paměti je uloženo unikátní sériové číslo čidla. Paměť obsahuje 2-bytový teplotní registr, ve kterém je uložen digitální výstup z teplotního senzoru. Dále poskytuje přístup k 1-bytovým registrům horního a dolního teplotního alarmu (TH a TL). Tyto registry jsou řešeny pamětí EEPROM, takže nastavené hodnoty zůstavají v platnosti i při vypnutém napájení.



Obr. 3 – Blokový diagram čidla DS18S20

Další vlastností obvodu DS18S20 je možnost funkce bez externího napájení. To je v tomto případě řešeno přes 1-wire pull-up rezistor na pinu DQ když je na sběrnici napájecí napětí. To zároveň nabije vnitřní kondenzátor (Cpp), který potom napájí zařízení když je sběrnice uzemněná.

4. Modul XPort

4.1 Základní informace

DSTni-XPort je nejkompaktnější integrovaný modul pro řešení zajištění síťové a WWW konektivity pro libovolné zařízení vybavené sériovým rozhraním.

Obr. 4.1 – Modul XPort

XPort byl navržen pro výrobce kteří potřebují nástroj pomocí kterého lze snadno a rychle zajistit připojení jejich zařízení do sítí LAN/WAN a Internetu. XPort nabízí výrobcům nevyšší dostupnou úroveň integrace v oboru terminálových serverů (převodníků RSxxx na IP). Uvnitř kompaktního krytu konektoru RJ-45 je umístěn DSTni-LX 186 procesor s řadiči, paměť, obvody rozhraní Ethernet 10/100, rychlý port RS232, diagnostické LED a 3 programovatelné I/O piny. V prostoru který je obvykle určen pro pouhý konektor poskytuje modul XPort kompletní síťové rozhraní a výpočetní výkon přibližně odpovídající PC XT. Pro připojení k lokální síti nebo internetu obsahuje XPort kompletní TCP/IP stack a operační systém nad nímž lze k standardním funkcím firmware i libovolné zákaznické aplikace. XPort také obsahuje kompletní WWW který lze využít např. pro dálkovou konfiguraci dohled, vizualizaci a ovládaní připojeného zařízení.

Kdekoliv je zapotřebí vytvořit uživatelské rozhraní a současně použít standardní, známé a široce rozšířené nástroje může XPort posloužit jako poskytovatel Java appletů obsahujících zákaznickou aplikaci pro web browser. Tímto se XPort může stát prostředníkem komunikace mezi Vámi a Vaším zařízením připojeným k síti LAN nebo internetu.

Pro zjednodušení instalace a nastavování je k dispozici konfigurační SW pro Windows™ - XPort Installer. XPort může být konfigurován lokálně po sériovém portu nebo dálkově s použitím telnetu nebo web browseru. Vestavěná paměť Flash slouží pro bezůdržbové uložení obsahu webových stránek a umožňuje bezporblémový upgrade systémových SW modulů v případě potřeby. Použitím modulů Xport jako vysoce integrované hardwarové a SW platformy lze dosáhnou významných přínosů a úspor podstatným zrychlením procesu vývoje, snížení rizik a nákladů.

4.2 Parametry

XPort - Vlastnosti

- Protokoly: TCP/IP, UDP/IP, ARP, ICMP, SNMP, TFTP, Telnet, DHCP, BOOTP, HTTP, AutoIP.
- Sériová linka: CMOS (Asynchronní, 5V Tolerant) Podpora RS422 a RS485. Rychlost 300 Bd až 921600 Bd.
- Interní WEBové stránky (384 kB).
- Odesílání e-mailů.
- Na přání k dispozici varianta (XPort SE) s šifrováním (256-bit AES Rijndael).
- Ethernet 10Base-T nebo 100Base-TX (Automatické rozpoznání).
- Provedení v konektoru RJ45.
- Výkonný processor (12 MIPS, založen na rozšřeném 16 bit DSTni-EX, 48MHz nebo 88MHz, architektura x86).
- Napájení 3,3V.
- Vývojový kit pro snadné ladění aplikací s modulem XPort.
- Certifikát RoHS a WEEE.

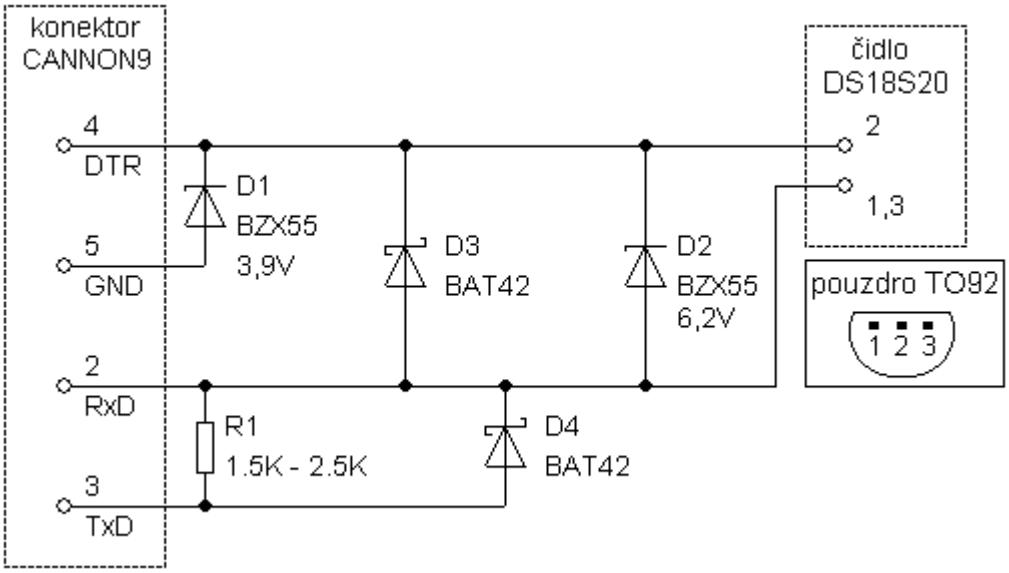
XPort - Použití

- Vývoj a výroba zařízení s Ethernetovou konektivitou.
- Snadné připojení Vašich stávajících zařízení se sériovým rozhraním do Ethernetu.
- Ovládání zařízení přes webové rozhraní.
- Průmyslová i domácí automatizace.
- Informační systémy.
- Měřící přístroje s Ethernetovým rozhraním.

5. Obsluha 1-wire teploměru v systému Linux

5.1 Hardware

Čidlo je potřeba k COM portu PC připokit přes tzv. adaptér. Lze použít aktivní adaptér DS9097U, který sice umožňuje připojení i jiných čidel fy Dallas, nicméně stojí řádově stovky Kč. Pro naše účely jednoduchého ověření měření teploty je tedy výhodnější pasivní adaptér, jehož schéma je na obrázku.



Obr 5.1 – Zapojení teploměru DS18S20 k sériovému portu PC

Jak již bylo uvedeno dříve, každé čidlo má z výroby své jedninečné čílo, což umožňuje jejich připojení paralelně k sobě. Počet čidel je teoreticky neomezený, prakticky je limit pouze v délce a kvalitě kabelů k čidlům. Samotné čidlo se k adaptéru připojí tak, že se jeho krajní nožky spojí a připojí na pin 2 sériového portu (RxD), prostřední nožička potom na pin 4 (DTR).

V tomto případě je napájení čidla zajištěno při komunikaci ze sériového portu. To má výhodu hlavně ve své jednoduchosti. Nevýhodou je omezená horní hranice čtení teploty na cca 70°C u většiny běžných PC. Také je možné narazit na nekompatibilitu s některými sériovými porty PC, v tomto případě může pomoci změna (zvětšení) odporu R1 anebo zařazení 10K trimru mezi piny TxD a GND.

5.2 Obslužný software

Obslužný program vychází z projektu DigiTemp a je uvolněn jako Open Source pod licencí GNU. Je k dispozici pod operační systémy Unix/Linux, DOS a Windows, nicméně v této chvíli je podporována pouze verze pro Linux. Program samotný je pouze konzolová aplikace, která "jen" přečte teploty z čidel, případně je uloží do logu.

Po zkompilování zdrojových kódů je aplikaci nejprve potřeba spustit příkazem:

```
digitemp_DS9097 -s/dev/ttyS0 -i
```

Ten prohledá sběrnici na zadaném portu a nalezne všechna zařízení na ní připojené. Zjištěné informace zapíše do konfiguračního souboru .digitemprc

```
DigiTemp v3.3.2 Copyright 1996-2004 by Brian C. Lane
GNU Public License v2.0 - http://www.brianlane.com
Turning off all DS2409 Couplers
..
Searching the 1-Wire LAN
1030A8BA000800E7 : DS1820/DS18S20/DS1920 Temperature Sensor
108396BA0008009F : DS1820/DS18S20/DS1920 Temperature Sensor
ROM #0 : 1030A8BA000800E7
ROM #1 : 108396BA0008009F
Wrote .digitemprc
```

Poté už stačí spustit program s parametrem -a:

```
digitemp_DS9097 -a
```

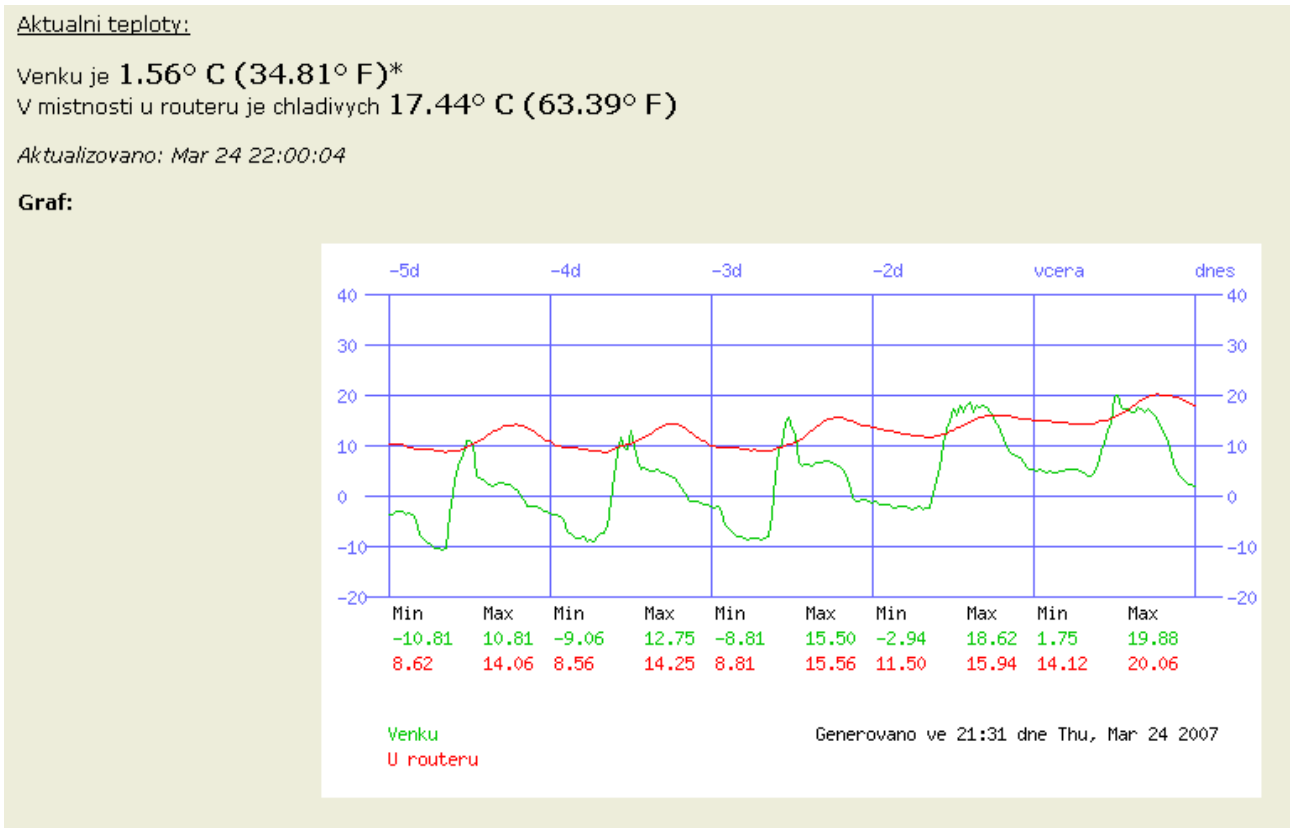
a ten již provede vlastní čtení teploty z připojených čidel:

```
DigiTemp v3.3.2 Copyright 1996-2004 by Brian C. Lane
GNU Public License v2.0 - http://www.brianlane.com
Mar 13 17:58:07 Sensor 0 C: 1.06 F: 33.91
Mar 13 17:58:09 Sensor 1 C: 12.88 F: 55.17
```

a naměřené hodnoty zároveň zapíše do log souboru v tomto tvaru:

```
Apr 30 22:01:02 Sensor 0 C: 5.12 F: 41.23
Apr 30 22:01:04 Sensor 1 C: 14.88 F: 58.77
Apr 30 22:31:02 Sensor 0 C: 5.19 F: 41.34
Apr 30 22:31:04 Sensor 1 C: 14.88 F: 58.77
Apr 30 23:01:02 Sensor 0 C: 5.06 F: 41.11
Apr 30 23:01:04 Sensor 1 C: 14.81 F: 58.66
Apr 30 23:31:02 Sensor 0 C: 5.06 F: 41.11
Apr 30 23:31:04 Sensor 1 C: 14.69 F: 58.44
```

Tyto hodnoty je pak samozřejmě možné dále zpracovávat, ukládat do databáze, tvořit grafy, přehledy atd.



Obr. 5.2 – Možný výstup naměřených dat v podobě online teploměru

Toto je velice jednoduchá aplikace měření teplot s 1-wire senzory DS1820 umožňující zároveň široké možnosti přímého zpracování dat, nicméně je zde nevýhoda nutnosti neustále běžícího měřících PC v blízkosti měřených objektů, což je nevýhodné z důvodů spotřeby, zvukového rušení a také např. napájení v případě výpadku elektrické energie. Proto se jeví jako výhodnější realizovat měření za použití mikrokontroléru a PC použít pouze na konečné zpracování dat.

6. Realizace po mikrokontrolér

6.1 Vývojová deska DEMO9S08QG8

DEMO9S08QG8 je demonstrační deska pocházející z dílen firmy Freescale Semiconductor, jenž je určena pro mikrokontroléry MC9S08QG8. Usnadňuje vývoj aplikací díky rozhraní USB-BDM, vývojovému studiu CodeWarrior a softwarovými príklady.

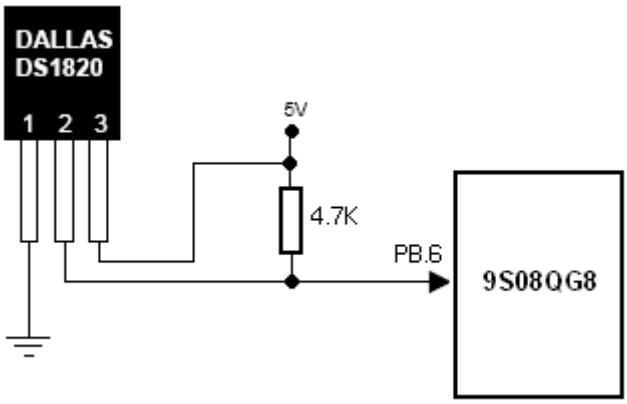
Obr. 6.1 – Vývojová deska

V desce DEMO9S08QG8 je použit mikrokontrolér MC9S08QG8, což je 8bitový mikrokontrolér založený na jádru HCS08 a disponující 8kB FLASH a 515B RAM.

Základní vlastnosti

- CPU MC9S08QG8, 16-pin DIP
  - 8kB Flash
  - 512B RAM
  - Interní 32kHz Oscilátor
  - 12x GPIO, 1x Input Only, 1x Output Only
  - Timer Interface Module
  - Komunikační porty SCI a SPI
  - IIC modul
  - 2kanálový, 16bitový Timer
  - 8kanálový, 10bitový ADC
  - Analogový komparátor
- Rozhraní USB-BDM
- Sériový port RS-232 (DB9)
- Jumper pro výběr vstupního napětí
- Konektory
- 32pin MCU I/O
- 2.0mm napájecí konektor
- Sérioový port na DB9
- Velikost desky 2,9 x 2,5

6.2 Hardwarové připojení



Obr. 6.2 – Připojení teploměru DS18S20 k mikroprocesoru

V tomto případě bylo pro jednoduchost a pro ověření funkce použito externí +5V napájení čidla a sběrnice přes pull-up rezistor

6.3 Softwarové řešení ve Freescale CodeWarrior

Jak již bylo nastíněno v popisu výše, je komunikace po 1-wire sběrnici z velké části založena na správném časování. Proto je potřeba nejprve správně navrhnout funkce pro časové zpoždění.

Interní oscilátor v mikroprocesoru MC9S08QG8 pracuje na kmitočtu 16MHz. Perioda tohoto taktu je 0,0625us, z čehož vyplývá, že pro 1us je potřeba 16 strojových cyklů. Z toho vychází následující funkce zpoždění.

```
//----------------------------------------
// Funkce zpozdeni, platne pro 16MHz takt
// jedno zavolani funkce odpovida 1us
//----------------------------------------
void delay(int micros)
{
__asm {
  LDA micros  // 2 cykly
  SUB #3 // 2 cykly
  TAX // 1 cyklus
  NOP // 1 cyklus
  NOP // 1 cyklus
    (celkem 12x)
loop:
  decx // 1 cyklus. Odecte X kazdych 16 cyklu => 1 us
  nop // 1 cyklus
  nop // 1 cyklus
    (celkem 11x)
bne loop // 4 cykly
} }
```

Funkce byla v zájmu co možná nejvyšší přesnosti realizována v assembleru. Jejím parametrem je přímo počet us, tedy doba trvání zpoždění. Funkce pro odečítání (SUBD #3) je zde z důvodu kompenzace zpoždění při volání a návratu z C funkce zpoždění, což je zhruba 2us pro kompilátor CodeWarrior.

Druhou funkcí pro zpoždění je zpoždění 1ms. V této funkci je pouze 1000x voláno předchozí zpoždění.

Nyní již přichází na řadu samotná komunikace, nejprve je potřeba provést reset sběrnice:

```
//---------------------------------------
// Reset DS1820
//---------------------------------------
Bool ResetDS1820(void)
{
       Bool presence;
       DQ = 0;             // prizemneni sbernice
       Delay(480);         // a nechat tak po cca 480us
       DQ = 1;             // navraceni zpet na napajeci napeti
       Delay(60);          // cekani na odezvu 60us
       presence = DQ;      // precteni odezvy
       DelayUs(316);       // cekani na konec timeslotu 316us
       return(presence);   // vraci stav odezvy
}      // 0 = pritomno, 1 = nic tam neni
```

Funkce nejprve vyšle na sběrnici 0 po dobu 480us a poté ji vrátí zpět na výchozí stav, tedy log. 1. Poté se čeká 60us a pokud se na sběrnici vyskytuje nějaké 1-wire zařízení, odpoví v této době 0V presence pulsem. Vrácená hodnota této funkce pak odpovídá zjištěnému stavu: 0 – čidlo nalezeno, 1 - na sběrnici se nevyskytuje 1-wire zařízení.

Pro čtení a zápis bitu na sběrnici slouží tyto funkce:

```
//---------------------------------------
// Precte 1 bit z DS1820
//---------------------------------------
Bool ReadBit(void)
{
       unsigned char i=0;
       DQ = 0;      // prizemneni sbernice pro spusteni timeslotu
       DQ = 1;
       Delay(50); // cekani 50us od startu timeslotu
       return(DQ); // vrati hodnotu DQ
}
```

Komunikace začíná vysláním 0V pulsu na sběrnici a poté se čeká 50us a vrátí se hodnota stavu sběrnice. Pokud chce senzor odeslat nulový bit, bude v této době držet sbernici na hodnotě 0, jinak sběrnice zůstává na hodnotě napájecího napětí.

```
//---------------------------------------
// Zapise 1 bit do DS1820
//---------------------------------------
void WriteBit(Bool Dbit)
{
       unsigned char i=0;
       DQ = 0;
       DQ = Dbit ? 1:0;
       Delay(60);                // zpozdeni 60us
       DQ = 1;
}
```

Princip je podobný funkci předchozí, opět se začíná 0v pulsem, který se v případě odesílání nulového bitu protáhne na 60us, jinak se sběrnice hned vrátí na hodnotu 1.

```
//---------------------------------------
// Precte 1 byte z DS1820
//---------------------------------------
unsigned char ReadByte(void)
{
       unsigned char i;
       unsigned char Din = 0;
       for (i=0;i<8;i++)   // precte byte, bit po bitu
       {
              Din|=ReadBit()? 0x01<<i:Din;
              DelayUs(6);
       }
       return(Din);
}

//---------------------------------------
// Zapise 1 byte
//---------------------------------------
void WriteByte(unsigned char Dout)
{
  Bool bit = 1;
       unsigned char i;
       for (i=0; i<8; i++) // zapise byte, bit po bitu
       {
              WriteBit((Bool)(Dout & 0x1));
              Dout = Dout >> 1;
       }
       DelayUs(5);
}
```

Tyto funkce slouží k odeslání celého datového slova, tedy postupně pomocí funkcí ReadBit/WriteBit odešlou/přijmou všech jednotlivých 8 bitů.

Tyto výše uvedené postupy používá hlavní funkce pro vlastní měření teploty:

```
//---------------------------------------
// Hlavni funkce cteni teploty
//---------------------------------------
void ReadTemp(unsigned char * buff)
{
unsigned char n;

if (ResetDS1820() == 0) {
    WriteByte(0xcc);  // skip ROM
    WriteByte(0x44);  // konverze teploty
    while (ReadByte()==0xff); // cekani na dokonceni konverze
    ResetDS1820();
    WriteByte(0xcc);  // skip ROM
    WriteByte(0xbe);  // precteni vysledku

    for (n=0; n<9; n++)     // precte 9 bytu a nasklada se do pole
    {
       buff[n]=ReadByte();  // cteni z DS1820
    }
}
}
```

Funkce provede reset sběrnice a pokud je na ní přítomen teploměr (návratová hodnota funkce ResetDS1820() je 0), realizuje vlastní měření, jehož výsledky uloží do pole bufferu.

Naměřená data jsou v tuto chvíli zobratována na sériové konzoli (v případě MS Windows např. v programu Hyperterminál) a k tomu je potřeba také funkce pro obsluhu sériového rozhraní mikroprocesoru. Inicializace se provádí následovně:

```
void InitSCI(word baud_div) {

  SCIBD = baud_div;     // nastaveni rychlosti v zavislosti na sbernici
  SCIC1 = 0x00;
}
```

A poté je již možno posílat zprávy po sériové lince pomocí funkce ShowMsg (vycházející z ukázkové aplikace vývojové desky)

```
void SendMsg(char msg[]) {
  byte ix=0;    // ukazatel na pozici ve vstupnim retezci
  byte dummy;   // docasna promenna pro cteni SCIS1
  byte nxt_char;

  SCIC2 = 0x08;    // povoleni Tx
  dummy = SCIS1;
  nxt_char = msg[ix++];
  while(nxt_char != 0x00) {
    SCID = nxt_char;
    nxt_char = msg[ix++];
    while(!SCIS1_TDRE){
//      feedCOP();
    };
  }
```

```
  while(!SCIS1_TC){
//    feedCOP();
  };
  SCIC2_TE = 0;
} /
```

Nyní již zbývá pouze hlavní funkce programu:

```
//----------------------------------------
// Hlavni funkce - main.c
//----------------------------------------

unsigned char MyTemp[9];

void main(void) {
      unsigned char tpf,tpc,i;
      DQ = 1;             // Vychozi stav sbernice
      PTBDD_PTBDD6 = 1;   // Nastavi PTB6 jako vystup
      SOPT1 = 0x01;       // Zakazani watchdogu

for(;;) {
      ReadTemp(&MyTemp[0]);
      tpf  = MyTemp[0] >> 1;
      tpc = ((MyTemp[0] >> 1)&1) ? 5:0;
      putchar(0x0C);  // smaze Hyper terminal
      ShowMsg("%bu Temperature : %2bu.%bu\r\n",i++,tpf,tpc);
      DelayMs2(1000);  // pockat 1s
      }
}
```

Po nezbytné inicializaci periferií dochází periodicky po jedné sekundě ke čtení teploty z čidla a jejímu zobrazení na sériové lince.

Tento princip obsluhy 1-wire sěrnice je velmi jednoduchý, ovšem na druhou stranu také velmi transparentní a tím také vhodný pro ověření funkce této komunikace.

7. Závěr

V tomto semestrálním projektu byla ověřena komunikace po 1-wire sběrnici.

…

Použitá literatura a zdroje