

Embedded řešení s obvodem ADuC

Embedded solutions using an ADuC IC

Bc. Robert Malík

Diplomová práce
2005



Univerzita Tomáše Bati ve Zlíně
Fakulta technologická
Institut řízení procesů a aplikované informatiky

Oficiální zadání diplomové práce

Děkuji tímto vedoucímu mé diplomové práce Doc. Ing. Františku Hruškovi, Ph.D. za odborné vedení, rady, připomínky a všechny zodpovězené dotazy týkající se dané problematiky. Zároveň chci také poděkovat všem, kteří mne při realizaci práce podporovali.

Souhlasím s tím, že s výsledky mé práce může být naloženo podle uvážení vedoucího diplomové práce, ředitele ústavu a institutu. V případě publikace budu uveden jako spoluautor.

Prohlašuji, že jsem na celé diplomové práci pracoval samostatně a použitou literaturu jsem citoval.

Ve Zlíně, 10. 06. 2005

.....

podpis diplomanta

RESUMÉ

Diplomová práce se zabývá řešením embedded aplikací s obvodem ADuC845. Tento integrovaný obvod je kombinací mikropočítače založeného na průmyslovém standardu 8051 a analogových funkcí, jako je například 24-bitový AD převodník. Proto je obvod vhodný v aplikacích, kde je třeba přesné měření, řízení a vyhodnocování naměřených dat. Hlavním cílem je navrhnout zařízení pro měření teploty pomocí kovového odporového teplotního senzoru, popsat aspekty měření a zpracování signálu a vytvořit funkční vzorek tohoto měřicího zařízení za pomoci vývojového kitu ADuC845 Quickstart+.

SUMMARY

The Master thesis deals with embedded design solutions using an ADuC845 microcontroller. This integrated circuit combines an industry-standard 8051 based microcomputer core with precision analog functions, such as 24-bit AD converter. Therefore the device is suitable for applications where high performance measurement, control and data acquisition is needed. The purpose is to design device for measuring temperature by the means of resistance temperature detector, describe basic aspects of measuring and signal conditioning, and create working sample of this measuring device based on ADuC845 Quickstart+ development kit.

OBSAH

ÚVOD	7
1 ANALÝZA ZADÁNÍ A REŠERŠE	9
1.1 EMBEDDED (VESTAVNÉ) SYSTÉMY	9
1.1.1 Charakteristika a funkce embedded systémů	9
1.1.2 Konstrukce embedded systémů	12
1.1.3 Software pro embedded systémy	13
1.2 MIKROPROCESORY ŘADY 8051	14
1.2.1 Charakteristika mikroprocesoru 8051	15
1.2.2 Nástupci mikroprocesoru 8051	17
1.2.3 Nabídka mikrokontrolerů řady 80C51	17
1.3 VÝVOJ PROGRAMŮ PRO MIKROPROCESORY 80C51	19
1.3.1 Modulární tvorba programů	20
1.4 MIKROKONVERTOR ADUC845	21
1.4.1 Analogová část	22
1.4.2 Mikroprocesorová část	26
1.4.3 Vestavěné periferie	31
1.4.4 Návrh programového vybavení pro ADuC845	33
2 ASPEKTY MĚŘENÍ A ZPRACOVÁNÍ SIGNÁLŮ	35
2.1 TECHNICKÉ PROSTŘEDKY MĚŘENÍ	35
2.1.1 Senzor	35
2.1.2 Měřicí řetězec	35
2.1.3 Vlastnosti technických prostředků měření	37
2.2 CHYBY A NEJISTOTY V MĚŘENÍ	38
2.2.1 Chyby v měření	38
2.2.2 Nejistoty v měření	39
2.3 MĚŘENÍ TEPLoty POMOCÍ RTD SENZORŮ	42
2.3.1 Princip RTD senzorů	42
2.3.2 Platinové odporové senzory	43
2.3.3 Funkční závislost senzorů Pt100	43
2.3.4 Toleranční třídy a konstrukce	44
2.4 ZPRACOVÁNÍ SIGNÁLU RTD SENZORŮ POMOCÍ OBVODU ADUC	45
2.4.1 Připojení senzoru	45
2.4.2 Linearizace	46
2.4.3 Kalibrace	52
2.4.4 Analýza nejistot	55
3 EXPERIMENTÁLNÍ SESTAVY S OBVODEM ADUC	58
3.1 OVĚŘENÍ FUNKCE AD PŘEVODNÍKU	58
3.1.1 Návrh a realizace obvodu	58
3.1.2 Programová realizace	60
3.1.3 Ověření funkčnosti	61
3.1.4 Hodnocení	63
3.2 PŘIPOJENÍ SENZORU PT100	63
3.2.1 Návrh a realizace zapojení	63
3.2.2 Výběr referenčního odporu	65
3.2.3 Výpočet hodnoty odporu z výstupu AD převodníku	66
3.2.4 Funkce DA převodníku	67
3.2.5 Programová realizace	67
3.2.6 Blokové schéma programu	69
3.2.7 Ověření funkčnosti	71
3.3 EXPERIMENTÁLNÍ MĚŘENÍ	72
3.3.1 Použité přístroje	72

3.3.2 Měření.....	73
3.4 VYHODNOCENÍ A DISKUSE VÝSLEDKŮ.....	77
4 ZÁVĚR.....	80
SEZNAM POUŽITÉ LITERATURY	82
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	83
SEZNAM OBRÁZKŮ.....	85
SEZNAM TABULEK	86
SEZNAM PŘÍLOH.....	87

ÚVOD

Tato diplomová práce se zabývá problematikou embedded (vestavných) systémů. Jedná se o malé počítačové systémy zabudované uvnitř zařízení, jejichž hlavní část tvoří program běžící zpravidla na jednom mikroprocesoru rozšířeném o další periferní prvky určené pro styk s okolím. Takové systémy dnes najdeme v mnoha přístrojích, se kterými se denně běžně setkáváme a jejich existenci v domácí a spotřební elektronice si ani neuvědomujeme. Každé moderní zařízení z měřicí, automatizační a regulační techniky si lze dnes bez těchto jednoúčelových mikroprocesorem řízených systémů jen těžko představit, a proto je znalost jejich návrhu a vývoje v současné době velmi důležitá. Přední světoví výrobci nabízejí nepřeberné množství obvodů vhodných pro realizaci embedded aplikací. Typickým zástupcem obvodů osazených mikroprocesorem řady 8051 je ADuC845 od Analog Devices, který jako jeden z mála disponuje 24-bitovým AD převodníkem, a je proto ideální pro zpracování a vyhodnocování senzorických signálů. Právě tento obvod mám k dispozici při řešení diplomové práce a pokusím se pomocí něj sestavit aplikaci typickou pro průmyslovou praxi, tedy měření teploty pomocí kovového odporového senzoru Pt100.

Tematicky lze celou práci rozdělit na tři části. První dvě části jsou převážně teoretické a odpovídají rozsahu dané problematiky, neboť pro správnou realizaci experimentální sestavy jsou nezbytné znalosti z oblasti elektroniky, mikroelektroniky, zpracování signálů, technických prostředků automatizace a informatiky. Poslední část je experimentální.

V první části jsem provedl analýzu a literární rešerši samotných obvodů na základě dostupných odborných publikací a článků. Nejprve je podrobně popsán význam pojmu embedded a následně uvedena analýza mikroprocesoru 8051 s uvedeným přehledem obvodů založených na tomto mikroprocesoru, nabízených předními světovými výrobci. Závěr kapitoly je pak zaměřen konkrétně na typ ADuC845 od Analog Devices. Druhá kapitola analyzuje aspekty měření a elektronického zpracování signálů v průmyslové automatizaci se zaměřením na měření a zpracování senzorického signálu kovových odporových senzorů teploty (RTD), včetně připojení těchto senzorů k obvodu ADuC845. Pro úplnost jsem zde také popsal základní pojmy které s touto problematikou souvisí a uvedl nové metody vyjadřování chyb a odchylek při měření.

Na základě těchto teoretických poznatků jsem pak sestavil dvě experimentální sestavy s obvodem ADuC845, popsané ve třetí kapitole. První sestava řeší samotný AD převod různých úrovní napětí, přivedených na analogové vstupy obvodu a zobrazení takto získaných hodnot v terminálu PC. Přitom sestava využívá pouze jeden ze dvou dostupných AD převodníku, a je tedy řešeno přepínání vstupních kanálů. Druhá, hlavní experimentální sestava řeší celkové vyhodnocení teploty připojeného odporového kovového snímače teploty Pt100. Přivedený senzorický signál je tedy patřičně upraven, je provedena linearizace a výsledná naměřená teplota je zobrazována v terminálu PC. Pomocí této sestavy snímače teploty jsem pak provedl měření a hodnoty porovnal s hodnotami naměřenými pomocí moderního laboratorního přístroje pro měření teploty. Vyhodnocení výsledků je uvedeno v závěru třetí kapitoly.

1 ANALÝZA ZADÁNÍ A REŠERŠE

V první kapitole chci nejprve objasnit základní pojmy spojené s řešením embedded aplikací na základě analýzy dané problematiky a rešerše pomocí dostupných zdrojů. Jedná se především o definici samotného pojmu embedded systému a charakteristiku základního stavebního prvku tohoto systému, kterým je v mém případě mikropočítač řady 8051. Kapitola rovněž uvádí základní přehled některých modelů světových výrobců těchto integrovaných obvodů rozšířených o analogové vstupy a výstupy, a v poslední části nabízí podrobný popis obvodu ADuC845 který mám k dispozici při řešení diplomové práce.

1.1 Embedded (vestavné) systémy

Embedded systémy se staly nedílnou součástí života lidí v průmyslově vyspělých zemích. Běžný občan se s nimi setkává, aniž by si to příliš uvědomoval, téměř na každém kroku. Význam pojmu je poměrně široký a v literatuře lze nalézt mnoho definic vestavných systémů [5], např.:

- Jde o systém obsahující jeden nebo několik mikroprocesorů, jehož (jejichž) úkolem je řídit činnost displejů, motorů, světel nebo jiných přístrojů. Program prováděný mikroprocesorem je „mozkem“ stroje nebo přístroje. Dostává vstupní informace od operátora nebo ze snímačů a ovládá elektronické nebo mechanické zařízení.
- Vestavný systém je malý počítačový systém, který je zabudován uvnitř zařízení (stroje, elektrického spotřebiče, elektronického systému) za účelem dosažení vyšší úrovně inteligence a tím lepší nebo efektivnější činnosti daného zařízení.
- Vestavný systém je dedikovaný počítač pracující v reálném čase jako součást nějakého uceleného systému. Slouží pro řízení celého systému a poskytuje výpočetní služby ostatním částem systému.
- Počítač, který je zabudován do systému, ale pro uživatele není jako počítač viditelný.

1.1.1 Charakteristika a funkce embedded systémů

Embedded systémy lze charakterizovat ze dvou různých úhlů pohledu. Uživatele příliš nezajímá, co je uvnitř, ale jsou pro něj důležité užité vlastnosti, spolehlivost a cena.

Naopak konstruktéři a výrobci považují za podstatné otázky metodiky vývoje, dostupnosti klíčových součástek, požadavků na přístrojové vybavení atd.

Z pohledu uživatele lze tedy embedded systémy charakterizovat takto [5]:

- jde o systémy určené pro jednotlivou konkrétní aplikaci a navržené k vykonávání pouze určité úlohy nebo skupiny úloh,
- jsou na ně kladeny různé omezující požadavky, např. na rozměry, cenu, příkon, rychlost uvedení na trh apod.,
- jde o reaktivní systémy, tedy systémy, které s využitím vhodných prostředků sledují okolní prostředí a odpovídají na podněty z něj nebo řídí v reálném čase,
- jsou u nich kladeny požadavky na bezpečnost a spolehlivou funkci, neboť selhání hardwaru nebo softwaru může být životu nebezpečné (např. u systémů jako tzv. airbag nebo ochrany před zablokováním kola automobilu).

Z pohledu vývojáře se embedded systémy jeví jako systémy:

- pro jejichž konstrukci je na trhu velké množství různých typů mikroprocesorů,
- pro něž jsou k dispozici různé operační systémy, přičemž se ve vestavných aplikacích převážně používají operační systémy reálného času,
- disponující menším množstvím systémových zdrojů, než je tomu u stolních systémů (např. menší kapacitou paměti),
- k jejichž vývoji jsou nutné vesměs dosti nákladné speciální vývojové nástroje jako emulátory, logické analyzátory, speciální překladače vyšších programovacích jazyků se zaručenou spolehlivostí generovaného kódu apod.,
- u nichž je ladění programového vybavení podstatně složitější než u programů pro PC,
- jejichž hardware i software by měly být až extrémně robustní (ve smyslu odolnosti proti vnějším poruchám, rušivým signálům, změnám teploty okolí apod.).

Mezi základní aplikace vestavných systémů patří sběr dat, přímé číslicové řízení a interakce člověka se strojem [5]. Data jsou sbírána prostřednictvím čidel, za nimiž většinou následují další analogové obvody pro impedanční přizpůsobení, galvanické oddělení a úpravu úrovně signálu. Upravený signál je v AD převodníku převeden do číslicové formy a předán řídicímu systému. V jednodušších případech přicházejí vstupní data v podobě logických signálů.

Řídicí systém obvykle realizuje určitou formu diskrétního regulačního algoritmu, jako jsou např. proporcionálně-sumačně-derivační (PSD) algoritmus, různé typy identifikačních a adaptivních algoritmů, algoritmy optimálního řízení nebo fuzzy algoritmy. Důležitou funkcí řídicího systému je vyhodnocování a zpracování chybových signálů, tzv. alarmů.

Vestavné systémy bývají součástí zařízení, jako jsou např. pračky, televizory, měřicí přístroje, tiskárny apod., kde je třeba, aby uživatel mohl zadávat ovládací příkazy nebo nastavovat požadované hodnoty. Systémy naproti tomu musí uživateli nebo i nadřazenému systému předávat provozní informace, indikovat poruchové stavy, popř. zobrazovat naměřené hodnoty. V těchto případech je velmi důležité vhodně vyřešit komunikaci člověka s přístrojem anebo strojem. Běžně se problém komunikace s uživatelem řeší pomocí malé klávesnice nebo jen několika nastavovacích tlačítek a několikařádkového zobrazovače LCD.

Použití vestavných systémů je velmi široké. Nelze zde proto vyjmenovat všechny možné aplikace. Mezi nejtypičtější oblasti použití embedded systémů patří:

- **průmyslová automatizace:** senzorika (inteligentní čidla), průmyslové regulátory, servomechanismy, roboty, atd.,
- **automobily:** automatické řízení dodávky paliva, ochrana kol před zablokováním (ABS), automatické zamykání dveří atd.,
- **domácí spotřebiče:** vestavné mikrořadiče (mikrokontrolery) řídí činnost praček, radiopřijímačů, televizorů, videorekordérů apod.,
- **telekomunikace:** mobilní telefony,
- **vojenské a kosmické aplikace.**

1.1.2 Konstrukce embedded systémů

Základním konstrukčním prvkem embedded systémů je mikroprocesor, který je doplněn dalšími přídavnými obvody, jako jsou paměti, řadič přerušení, obvody vstupů a výstupů, komunikační obvody atd. Tím však rostou rozměry i cena systému. Řešením tohoto problému je integrace mikroprocesoru a periferních obvodů na jeden čip. Vznikne tak obvod typu SOC (System-On-Chip) neboli **mikrořadič** (mikrokontroler). V aplikacích, kde je kladen důraz na AD/DA převod signálů, lze tyto obvody označit také jako **mikrokonvertory**.

Vestavené periferie jsou většinou orientovány na styk vestavného systému s okolními subsystémy (čidly, výkonovými členy), případně s člověkem, a na zajištění dalších funkcí, které jsou od vestavného systému požadovány (komunikace po průmyslových sběrnících, watchdog atd. [5]). Typické prvky mikrořadiče integrované spolu s výpočetním jádrem (mikroprocesorem) na jeden čip jsou:

- paměti RAM, ROM, EPROM, EEPROM a Flash,
- binární v/v rozhraní,
- sériové komunikační kanály,
- AD a DA převodníky,
- čítače vnějších událostí, časovače pro měření a generování časových intervalů a hodiny reálného času (Real Time Clock – RTC),
- řadiče přerušení,
- obvody pro generování pulsní šířkové modulace (Pulse Width Modulation – PWM),
- hlídací obvody (watchdog),
- řadiče sběrnic datových sítí jako např. SPI a I²C (pro spotřební elektroniku), ale také Ethernet, CAN (pro průmysl), USB, J1850 (automobilní sběrnice).

Prvním všeobecně dostupným mikrořadičem byl osmibitový mikrořadič I8048, uvedený na trh firmou Intel v roce 1976. Jeho zdokonalením vznikl v roce 1980 také osmibitový mikrořadič I8051. Ten dosáhl značné popularity a rozšíření. Největším světovým výrobcem mikrořadičů se stala firma Motorola. Spolu s ní a firmou Intel se výrobě

mikrořadičů věnují i další světoví výrobci, jako Analog Devices, Siemens, Atmel, Philips, Dallas Semiconductors, Texas Instruments, a jiní.

Mezi nejdůležitější vývojové trendy v oblasti mikrořadičů pro vestavné aplikace v současnosti patří:

- integrace pamětí Flash s kapacitou řádově stovek kB až MB na čip mikrořadiče namísto dříve používaných programovatelných pamětí (PROM, EPROM), které byly integrovány na čipu (jejich kapacita však byla řádu jednotek, maximálně desítek kB) nebo se připojovaly k mikrořadiči z vnějšku,
- přechod od osmibitových procesorových jednotek k šestnáctibitovým pro méně náročné a k 32bitovým pro náročnější aplikace,
- použití pokročilých architektur jako RISC (Reduced Instruction Set Computer – procesor s redukovanou množinou instrukcí), řetězení instrukcí (pipelining), superskalární architektury (superskalární procesory mají několik funkčních jednotek, např. ALU, což procesorové jednotce umožňuje vykonávat několik operací současně; jde o paralelismus na úrovni instrukcí),
- možnost ladit aplikace bez nutnosti vyjmout procesor ze systému a bez použití emulátoru (in circuit debugging): bloky integrované na čipech moderních mikrořadičů umožňují za chodu běžného programu načítat a modifikovat hodnoty proměnných, zastavovat, spouštět a krokovat program a číst a modifikovat obsah registrů procesoru.

1.1.3 Software pro embedded systémy

Dříve se programové vybavení pro vestavné systémy vytvářelo převážně v assembleru a někdy i přímo ve strojovém kódu [5]. Bylo to převážně dáno malou rychlostí a malou paměťovou kapacitou tehdejších systémů. Pro použití vyššího programovacího jazyka nebyla optimalizována ani architektura centrální procesorové jednotky. Tento přístup je v současné době již překonán. Všichni přední světoví výrobci moderních mikrořadičů navrhují architekturu procesorové jednotky přímo pro podporu vyšších programovacích jazyků a příliš neuvažují o tom, že by tyto procesory byly programovány v assembleru. Paměťové kapacity vzrostly z dřívějších jednotek na stovky kB a v současnosti již i na megabyty. Stejně zásadně vzrostla i rychlost provádění kódu. S rozvojem PC byly vytvořeny efektivní překladače, jimiž vygenerovaný výsledný kód je rychlejší a menšího

objemu, než jaký by člověk sám dokázal vytvořit pomocí assembleru. Použití vyššího programovacího jazyka znamená:

- rychlejší vývoj softwaru a z toho plynoucí menší náklady na vývoj celého výrobku,
- snazší údržbu softwaru,
- značné zjednodušení týmové práce na jednom softwarovém projektu,
- vzrůst bezpečnosti softwaru,
- přenositelnost programů mezi různými typy procesorů, a tudíž možnost využívat odladěné a dlouhodobě ověřené algoritmy.

Protože v něm lze snadno pracovat na nízké úrovni (blízko hardwaru), využívá se pro programování vestavných systémů převážně programovací jazyk C. V současné době se u rozsáhlejších projektů začíná používat jazyk C++ (objektové rozšíření jazyka C).

Jako operační systémy se pro vestavné aplikace používají převážně operační systémy reálného času (Real-Time Operating System – RTOS). Mezi nejčastěji užívané patří:

Windows CE (PocketPC), Palm OS, CMX, RTLinux, Symbian.

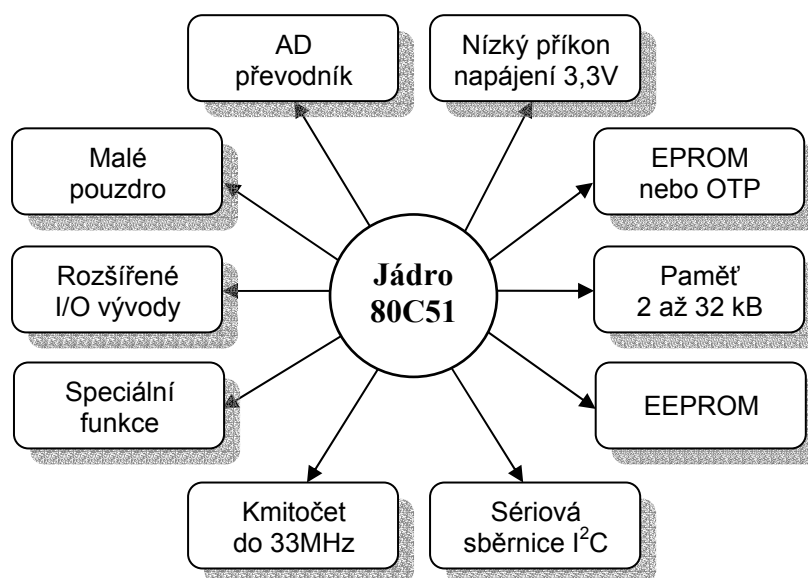
1.2 Mikroprocesory řady 8051

Tyto mikroprocesory jsou vývojově relativně staré, přesto si však díky své rozšířenosti získaly mezi vývojáři značnou popularitu, a ještě v současné době se řada výrobců orientuje na výrobu mikrořadičů s jádrem procesoru 8051 [4]. Toto jádro bývá rozšířeno o další periferie jako jsou:

- paměť programu o velikosti 2 kB až 32 kB
- paměť EEPROM pro uchování konstant, rozšířená vnitřní paměť RAM na 256B,
- až 24-bitový AD převodník s několika-kanálovým analogovým multiplexerem,
- rozšířené vstupně/výstupní brány,
- dvoudrátová přístrojová sběrnice I²C

Výrobci nabízejí procesory od základního hodinového kmitočtu 12 MHz až po 33 MHz ve standardních pouzdrech (DIL, PLCC, PQFP) [4]. Některé firmy se orientují na procesory s rozšířenými aritmetickými operacemi (Siemens 80C537) nebo s odlišným vnitřním časováním (Dallas DS80C320), který má při standardním hodinovém kmitočtu až trojnásobný výkon.

Sjednocujícím základem všech těchto rozmanitých typů procesorů je vlastní jádro procesoru 8051 (80C51) rozšířené o další periferie jak ukazuje obrázek 1.



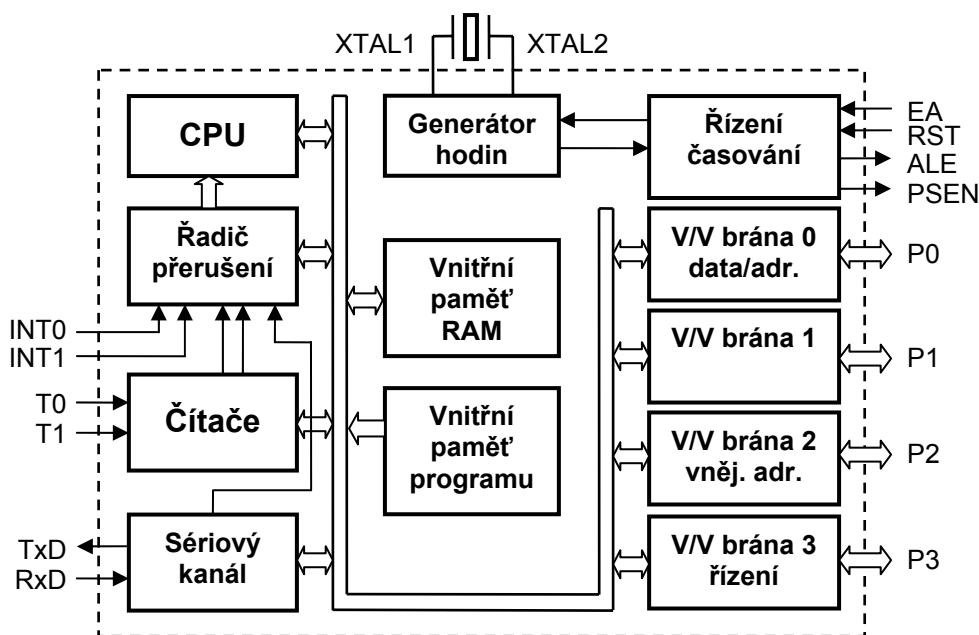
Obr. 1. Prvky rozšiřující vlastnosti základního jádra procesoru 8051

1.2.1 Charakteristika mikroprocesoru 8051

Mikroprocesor 8051 je 8-bitový jednočipový mikroprocesor s harwardskou strukturou, u které je oddělena programová a datová paměť. Procesor, jehož vnitřní struktura je blokově zobrazena na obrázku 2, je schopen samostatné činnosti po připojení vnějšího piezokeramického rezonátoru ("krystalu") na vývody XTAL1 a XTAL2 a jednoho napájecího napětí (3 až 5V).

Na čipu procesoru je umístěna vlastní procesorová jednotka CPU, která je vnitřní společnou sběrnici propojena s pamětí programu ROM nebo EPROM o kapacitě 4 kB, s pamětí RAM o kapacitě 128 bytů a se čtyřmi vstupně/výstupními branami PO až P3, které zajišťují styk procesoru s vnějšími periferiemi. Pro snadnější styk s periferiemi je procesor vybaven **řadičem přerušení**, který zpracovává 5 zdrojů přerušení. U nástupců 8051 se počet zdrojů přerušení v závislosti na počtu vnitřních periferií zvětšuje a dosahuje až

hodnoty 15. Struktura přerušení dovoluje provádět činnost zkoumaného programu po jednotlivých krocích.



Obr. 2. Vnitřní bloková struktura procesoru 8051

Čítače, které usnadňují realizaci časování, jsou 16-bitové s hodinovým signálem odvozeným z interního generátoru hodin nebo z vnějších vstupů TO nebo T1. Pro snazší sériový styk s nadřízenými počítači nebo jinými spolupracujícími procesory je vybaven duplexním (obousměrným) **sériovým kanálem**.

Procesor je dále vybaven Booleovským procesorem, který umožňuje pracovat s jednotlivými bity vnitřní paměti RAM i interních periférií, a má oddělené adresové prostory programu a dat, které jsou přístupné různými instrukcemi. Paměťový prostor můžeme dále dělit na vnitřní, umístěný na čipu, a vnější, který lze v případě potřeby vytvořit s pomocí dalších obvodů. Všechny informace, důležité pro činnost mikroprocesoru a jeho periferních obvodů integrovaných na čipu procesoru, jako jsou čítače, sériový kanál, přerušovací systém a u nástupců 8051 záchytný a komparační systém, jsou soustředěny do souboru registrů tzv. **registrů speciálních funkcí SFR**. Procesor obsahuje čtyři obousměrné datové brány P0 až P3, které obsahují výstupní vyrovnávací paměť adresovanou jako SFR registr.

1.2.2 Nástupci mikroprocesoru 8051

Nástupcem procesoru 8051 je typ 8052 a jeho další modifikace. Ten se odlišuje především tím, že má na čipu integrovanou rozšířenou vnitřní datovou paměť na kapacitu 256 bytů [4]. Tím dochází k tomu, že specializované registry leží ve stejném adresovém prostoru jako rozšířená vnitřní paměť RAM. Aby nedocházelo při adresování ke kolizím, je horních 128 bytů přístupných pouze přes instrukce nepřímého adresování a specializované registry pouze přes přímé adresování. Velkou výhodou rozšířené paměti je též fakt, že do ní může být umístěn zásobník, čímž se rozšíří část přímo adresovatelné paměti RAM.

Jednou z nových periférií, integrovaných do řady univerzálních i signálových procesorů, je **komparační systém**. Jedná se o velmi výkonnou periférii určenou převážně pro průmyslové aplikace, ve kterých je třeba: generovat pulzy s různou šířkou, realizovat měření šířky impulzů, řídit krokové motory, generovat kmitočty, D/A převody a řídit procesy. Tento systém umožňuje **pulzně šířkovou modulaci (PWM)** nebo generování řídicích signálů s minimální programovou obsluhou. Další výkonnou jednotkou moderních procesorů je **záchytný systém (capture)**, který slouží k pulzně šířkovému měření. Jedním z problémů mikroprocesorových systémů bez obsluhy s nepřetržitým provozem je jejich selhání v důsledku vnějšího rušení nebo poklesu napájecího napětí. Pro takovéto systémy byl do mikroprocesorů integrován obvod tzv. **hlídací pes (watchdog)**. Obvod se skládá z volně běžícího, obvykle 8-bitového čítače, který může být systémem po zapsání předepsané posloupnosti vynulován. Nulovací posloupnost je volena tak, aby pravděpodobnost jejího vzniku u systému nevykonávajícího požadovaný program byla minimální. Jestliže čítač nebude včas vynulován, dojde k jeho přetečení, které způsobí vznik nulování mikroprocesoru (RESET).

Výrazného pokroku bylo dosaženo v analogové části těchto integrovaných obvodů. Některé z nich disponují až 24-bitovým AD převodníkem (typ ADuC845 od Analog Devices) a řadou dalších podpůrných periférií pro analogové zpracování signálů.

1.2.3 Nabídka mikrokontrolerů řady 80C51

V současné době je u nás na trhu obrovské množství procesorů založených na jádře procesoru 8051, které vyrábí firma Intel, Analog Devices, Texas Instruments, Philips, Siemens a Atmel. Procesory můžeme rozdělit na procesory (mikrokontrolery) v pouzdech

s 24 a 28 vývody s omezeným počtem bran, a na procesory rozšířené o další periferie a brány v pouzdrech s 44 až 84 vývody. Protože je obtížné najít další společné vlastnosti těchto obvodů uvádím zde výčet jejich základních vlastností. V tabulce I je přehled některých základních typů podle výrobců.

Tabulka I. Vybrané mikrokontrolery 80C51

Typ	Hod. MHz	ROM kB	RAM B	I/O pin	AD(b) /kan.	DA(b) /kan.	Čas/ Čítač	WD	PWM (b)	Sériová linka	Pouzdro
Intel											
80C51	33	4	128	32	-	-	2	-	-	UART	DIP40, PLCC44
8xC51FB	12/16	16	256	32	-	-	3	1	5	UART	PLCC44
Atmel											
AT89C52	0-24	F8	256	32	-	-	3	-	-	UART	DIL40 PLCC44
AT89C8252	0-24	F8	256	32	-	-	3	1	-	UART	DIL40 PLCC44
Philips											
87C576	16	E8	256	4	8/10	-	3	1	-	UART, I2C	PLCC44
89LPC934	16	F8	256	26	8/4	8/1	4	1	2(8)	UART, I2C, SPI	PQFP80
Siemens											
SAB80C515 SAB80C535	12/16	8 -	256	48	8/8	-	3	1	-	UART	PLCC68
SAB80C517 SAB80C537	12/16	8 -	256	56	8/12	-	4	2	5	2xUART	PLCC84 PQFP100
Analog Devices											
ADuC812	16	F8	256	34	12/8	12/1	3	1	-	UART, SPI, I2C	LFCSP56 MQFP52
ADuC834	12	F62	2304	34	24/4	12/1	3	1	2(16)	UART, SPI, I2C	LFCSP56 MQFP52
ADuC845	12	F62	2304	34	24/10	12/1	3	1	2(16)	UART, SPI, I2C	LFCSP56 MQFP52
Silicon Laboratoires											
C8051F320	25	F16	2304	25	10/17	-	4	1	-	UART, SPI, CAN, USB	LQFP32
C8051F020	25	F64	4352	64	12/8	12/2	4	1	-	UART, SPI	TQFP100
Texas Instruments											
MSC1211Y3	25	F8	1280	32	24/8	8/1	3	1	2	UART, SPI	64TQFP
MSC1212Y5	25	F32	1280	32	24/8	16/4	3	1	2	UART, SPI	64TQFP

Vysvětlivky:

ROM - programová paměť

- xxkB programová paměť OTP pouze jednou programovatelná

- FxxkB programová paměť EEPROM nebo Flash

AD(b)/kan. - označuje rozlišení AD převodníku a počet kanálů (obdobně u DA převodníku)

WD - Watchdog Timer

PWM(b) - výstup pulsně šířkové modulace s rozlišením

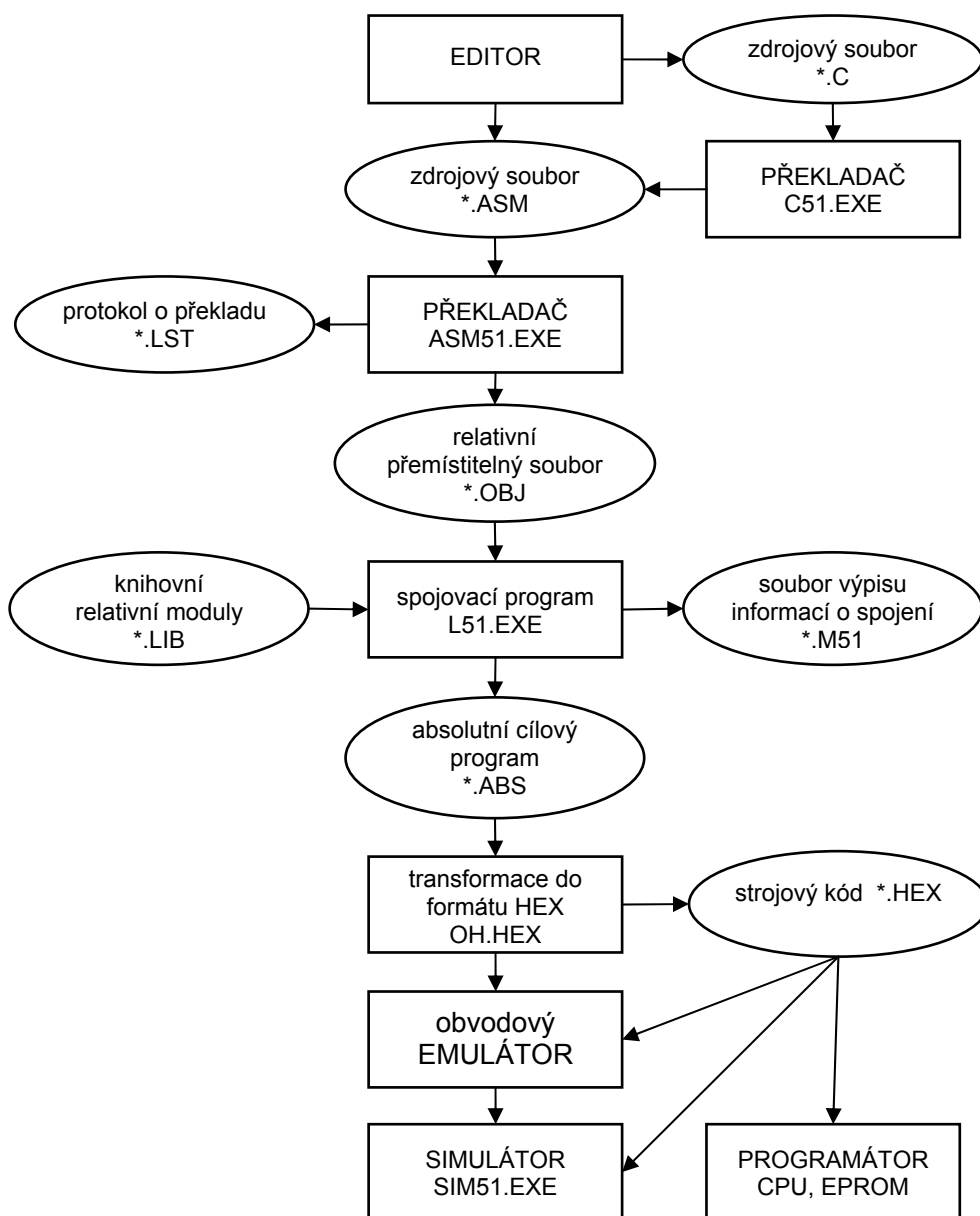
1.3 Vývoj programů pro mikroprocesory 80C51

Při realizaci programu pro daný typ procesoru je potřeba nejen znát dobře architekturu procesoru a možnosti jeho instrukčního souboru, ale také vlastnosti a možnosti programového vybavení, na kterém bude program vytvářen [4]. Kromě těchto znalostí je vhodné dodržovat určité základní postupy při tvorbě programového vybavení, které můžeme rozdělit do následujících kroků:

- Nejprve je třeba logicky správně popsat programovanou úlohu a navrhnout vhodné algoritmy pro její řešení. Pro snazší realizaci je vhodné úlohu rozdělit do menších funkčních celků, které budou později realizovány podprogramy, funkcemi nebo moduly.
- Vybrat vhodný programovací jazyk k napsání **zdrojového programu**, případně zdrojových souborů, které reprezentují jednotlivé moduly. Pro realizaci efektivních (obvykle rychlejších a paměťově méně náročnějších) programů je vhodný **jazyk symbolických adres JSA**. Tento jazyk je velmi blízký vlastnímu strojovému kódu procesoru a dnes se používá jako nejnižší programovací úroveň. Časově efektivnější je programování ve vyšším programovacím jazyce (jazyk C, Basic, atd.), které ale vede na paměťově rozsáhlejší programy. Jsou-li napsány zkušeným programátorem, potom jsou obvykle o něco pomalejší, než programy napsané v JSA. Zdrojový program v JSA nebo jazyce C je možné vytvořit v libovolném ASCII editoru.
- Připravené zdrojové texty se překládají pomocí překladačů. Je-li zdrojový text v jazyce C, potom je nejprve přeložen pomocí **kompilery** do jazyka symbolických adres (JSA). Zdrojové programy vytvořené kompilerem nebo přímo v JSA se překládají pomocí **assembleru** do relativního modulu (souboru s příponou OBJ). Tyto moduly lze spojit pomocí **linkeru** (spojovacího a přemísťovacího programu) do výsledného absolutního programu, adresovaného pro danou konkrétní konfiguraci mikroprocesorového systému. Tento program je možné nahrát do obvodového emulátoru nebo programového simulátoru a ověřit jeho činnost.
- Je-li výsledný program odladěn a vykonává požadovanou činnost, potom je nahrán do vnitřní paměti samotného procesoru nebo do paměti EPROM, EEPROM, nebo Flash. Pro programátory pamětí a procesorů je třeba výsledný absolutní modul programu transformovat do tzv. **INTEL HEX** nebo podobného formátu.

1.3.1 Modulární tvorba programů

Modulární programování přináší výhody v jednodušším a přehlednějším vytváření rozsáhlých programů. Umožňuje snazší ladění a úpravy vytvářených podprogramů, včetně možnosti jejich násobného použití a snadného začleňování do programových knihoven. Důležitou podmínkou je správné definování vstupů a výstupů podprogramů a předávání parametrů mezi moduly. Programátor pak může snadno jednotlivé moduly odladit a ověřené moduly spojit do výsledného programu nebo začlenit do knihoven. Postup při tvorbě programu je zobrazen na obrázku 3 pro programování s programovými segmenty, moduly, knihovnami a programy.



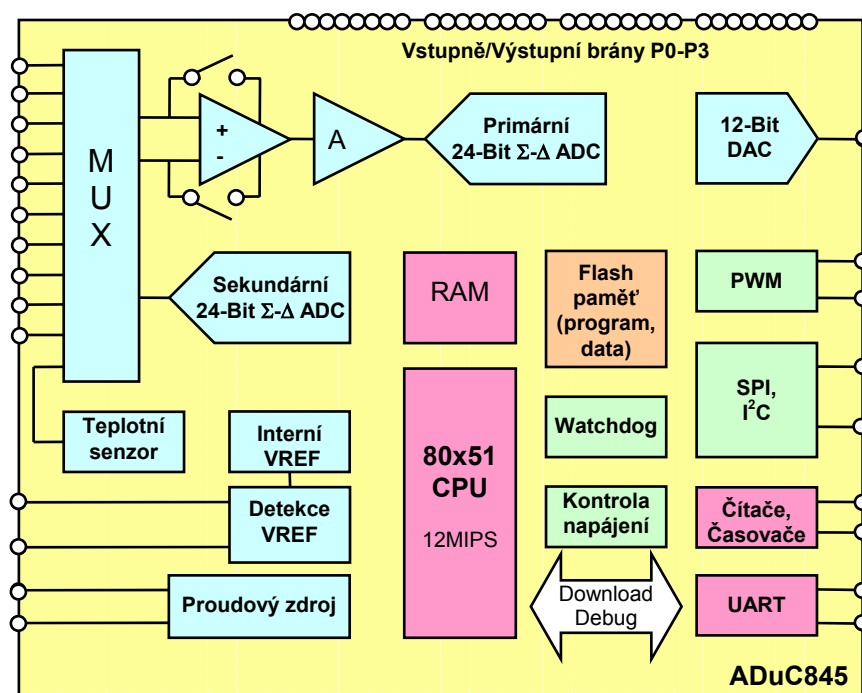
Obr. 3. Tvorba programu v jazyce symbolických adres a jazyce C

1.4 Mikrokonvertor ADuC845

Jedná se o nový obvod firmy Analog Devices představený v roce 2003, který obsahuje 24-bitový AD převodník, 8051 procesor s rozšířením, a až 62 kB programové Flash paměti. Obvod je běžně k dostání za přijatelnou cenu cca 12\$. Hlavní vlastnosti obvodu jsou:

- rozšířené 80C51 jádro
- duální 10-ti kanálový 24-bitový AD převodník
- jeden 12-ti bitový DA převodník
- teplotní senzor na čipu
- až 64 kB programové paměti
- 4kB datové paměti
- až 16 MB externí paměťový prostor
- duplexní UART komunikační obvod,
- komunikační rozhraní SPI a I²C
- obvody pro pulsně šířkovou modulaci (PWM)
- funkce Watchdog (WD)

Obvod ADuC845 je výsledkem dlouholetého vývoje a podporuje trend současné doby, tj. spojování několika bloků s velmi rozdílnou technologií výroby na jeden čip [9]. Sama firma jej nazývá mikrokonvertor namísto standardního mikroprocesor. Vnitřní blokové schéma je uvedeno na obrázku 4. Podrobněji je blokové schéma včetně označení vývodů zobrazeno v příloze 1.



Obr. 4. Vnitřní blokové schéma obvodu ADuC845

Obvod lze rozdělit na dvě části. Mikroprocesorovou a Analogovou (modrá barva). Analogová část obsahuje veškeré obvody pro zpracování signálu, jeho AD a DA převod a další podpůrné obvody, jako obvody detekce referenčního napětí, interní zdroj budícího proudu nebo teplotní senzor. Pro ovládání těchto obvodů slouží speciální funkční registry popsané v Mikroprocesorové části (viz. také příloha 1).

Z důvodů rušení uvnitř čipu a přesnosti výroby se jedná svým způsobem o unikátní řešení. Dosud nabízené AD převodníky na jednom čipu s procesorovým jádrem měly rozlišení pouze 8-12 bitů.

1.4.1 Analogová část

Analogová část mikrokonvertoru představuje hlavní konstrukční prvek obvodu. Právě z důvodu snížení rušení je tato část HW zcela oddělena od jádra 80C51 [9]. Jak napájecí, tak i zemnicí vývod je samostatný a při jejich zapojování je vhodné (pro maximální využití vlastností obvodu) jejich připojení podle výrobcem doporučeného zapojení.

AD převodníky (ADC)

V mikrokonvertoru ADuC845 jsou k dispozici dva 10-ti kanálové multiplexované 24-bitové Σ - Δ AD převodníky (primární a sekundární), umožňující maximální vzorkovací

frekvenci až 1.3 kSPS (1300 vzorků za sekundu) pro jeden vybraný kanál.

Primární AD převodník disponuje programově nastavitelným zesilovačem (PGA), vyrovnávací pamětí (BUF) a digitálním filtrem určeným pro měření signálů s velkým dynamickým rozsahem nebo nízkofrekvenčních signálů, jako jsou například senzorické signály tenzometrů, vah, tlakoměrů nebo teplotních senzorů. Vstupní kanály jsou tak určeny pro převod signálů senzorů bez nutnosti vnějšího předzpracování. Lze jej naprogramovat na jeden z osmi vstupních rozsahů od ± 20 mV až po ± 2.56 V podle vztaženého referenčního napětí ($V_{REF} \times 1.024$). **Sekundární AD převodník** nedisponuje digitálním filtrem ani programově nastavitelným zesílením. Má pevně nastavený rozsah ± 2.56 V (pro $V_{REF}=2.5V$). Oproti primárnímu ADC je tedy méně přesný.

Vlastní AD převod využívá techniku Σ - Δ , kdy modulátor převádí vzorkovaný vstupní signál na řadu impulsů, jejichž šířka udává digitální informaci (Duty Cycle). Ta je vyhodnocena pomocí programovatelného filtru s dolní propustí sinc^3 a výsledkem je pak 24-bitová hodnota uložená ve třech datových SFR registrech převodníku ADC0L, ADC0M, ADC0H (ADC1L, ADC1M, ADC1H pro sekundární AD převodník) jak ukazuje obrázek 5.

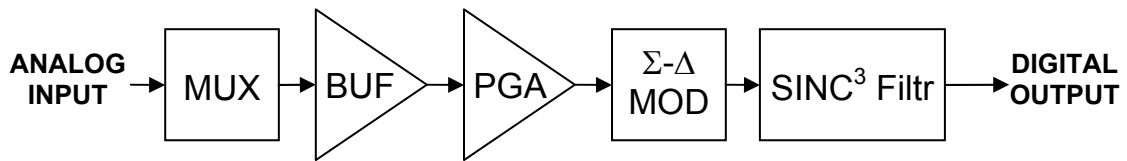


Obr. 5. Datové funkční registry AD převodníku (primární ADC)

ADC se programově konfiguruje pomocí 3 konfiguračních registrů (ADC0CON1, ADC0CON2, ADC1CON) v paměťové oblasti SFR. Takto lze například vybrat měřený kanál, a referenční signál. Pomocí SFR registru ADCMOD lze nastavit následující hlavní režimy činnosti AD převodníku:

- jednorázový převod
- trvalé kontinuální vzorkování (ADC datové registry jsou pravidelně obnovovány rychlostí nastavenou v registru SF)

Zjednodušené schéma řetězce obvodu pro zpracování a převod analogového signálu znázorňuje obrázek 6:



Obr. 6. Zjednodušené schéma měřicího řetězce AD převodníku

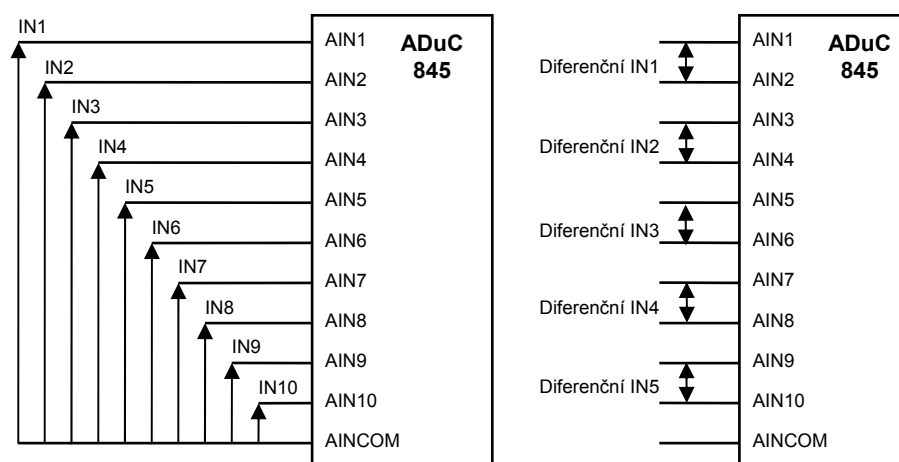
kde jednotlivé bloky jsou:

MUX	multiplexer
BUF	vyrovnávací paměť
PGA	programově nastavitelné zesílení
Σ - Δ MOD	modulátor pro AD převod
SINC ³ Filtr	filtr s dolní propustí pro potlačení kvantizačního šumu vzniklého při AD převodu

Řetězec pro zpracování signálu lze nastavit pomocí SFR registru ADCMODE do dvou režimů. tzv. „**chop enabled**“ a „**chop disabled**“. Implicitně je nastaven režim s povolením funkce „**chop**“, který poskytuje optimální výsledky zpracování signálu co se týká chyby offsetu a driftu převodníku, na úkor menší rychlosti zpracování. V tomto režimu ADC opakovaně převrací hodnotu na vstupu, tím je pozitivní i negativní offset vzniklý při AD převodu obsažen ve výstupním digitálním slovu. To je v konečné fázi sečteno s předchozí hodnotou a výsledná hodnota je pak dána průměrem těchto dvou vzorků. Výstupní rychlost v tomto režimu je v rozmezí od 5.35 Hz po 105 Hz. Při potlačení funkce „**chop**“ lze dosáhnout výstupní rychlosti od 16.06 Hz až po 1.365 kHz, je však nutná kalibrace offsetu, a také nejistota způsobená driftem AD převodníku je výrazně větší.

Konfigurace zapojení

ADuC845 lze nakonfigurovat podle pouzdra obvodu (MQFP nebo LFCSP) pro 4 až 5 diferenčních (bipolární zapojení) vstupních kanálů, nebo 8 až 10 pseudo-diferenčních (unipolární zapojení) vstupních kanálů vztažených k vývodu AINCOM, jak ukazuje obr. 7.



Obr. 7. Unpolární/bipolární připojení vstupních kanálů

Teplotní senzor

K sekundárnímu AD převodníku je interně připojen teplotní senzor s přesností ($\pm 2^\circ\text{C}$), který je vhodný nejen k indikaci teploty vlastního mikrokontroleru ale i ke kompenzaci chyb vzniklých rozdílnou teplotou okolního prostředí.

Referenční vstupy

ADuC845 má dva oddělené diferenční referenční napěťové vstupy, $\text{REFIN}\pm$ a $\text{REFIN2}\pm$. Oba lze použít pro primární ADC, pro sekundární ADC je použitelný pouze $\text{REFIN}\pm$. Obvod navíc disponuje interním referenčním zdrojem napětí o velikosti 1.25V.

Interní zdroj proudu

Mikrokonvertor obsahuje dva sdružené zdroje proudu $200\mu\text{A}$, které se dají programově konfigurovat pomocí dolních čtyř bitů SFR registru ICON. Pomocí těchto bitů lze nastavit dva proudové zdroje velikosti $200\mu\text{A}$ pro vývody obvodu (P1.6/AIN6 a P1.7/AIN7), nebo kombinací těchto dvou zdrojů vytvořit jeden o velikosti $400\mu\text{A}$ který je k dispozici na jednom z těchto vývodů. Takto vytvořené zdroje proudu jsou ideální k buzení připojeného odporového můstku nebo odporového senzoru teploty.

DA převodník

Vestavěný 12-ti bitový DA převodník s napěťovým výstupem je ovládán, obdobně jako AD převodník, přes SFR registry. Programově je možné, s použitím konfiguračního registru (DACCON), nastavit následující režimy činnosti DA převodníku:

- 8-mi nebo 12-ti bitový převod,
- rozsah výstupního napětí 0 až V_{REF} nebo 0 až AV_{DD} ,

Vlastní data pro DA převod se zapisují do datových registrů ADCL, DACH. V „L“ registrech je dolních 8 bitů a v dolní polovině „H“ registrů pak 4 horní bity pro 12-ti bitový převod.

1.4.2 Mikroprocesorová část

Výpočetní jádro mikrokonvertoru ADuC845 je plně kompatibilní s originálem 80C51. Frekvence oscilátoru je maximálně 12.58 MHz. Pro ovládání a komunikaci s analogovou částí a všemi ostatními vestavěnými periferními obvody je instrukční sada rozšířena o doplňkovou řadu instrukcí, které umožní jejich obsluhu přes SFR registry prostým zápisem nebo čtením z registru.

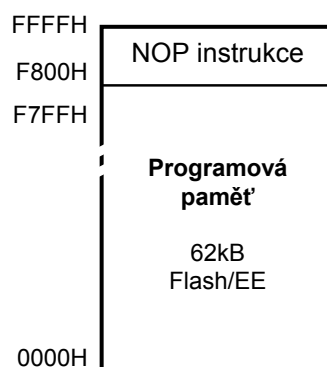
Organizace vnitřní paměti

Harwardská struktura mikrokonvertoru odděluje prostor pro data a program do dvou oddělených paměťových polí – paměť pro program a datovou paměť. Celkem se pak paměťový prostor mikrokonvertoru skládá ze čtyř paměťových bloků:

- 62kB (volitelně 32 kB nebo 8 kB) Flash/EE programová paměť
- 4kB Flash/EE datová paměť
- 256B základní RAM
- 2kB interní XRAM

Programová paměť

Velikost programové paměti je až 62kB a její mapování ukazuje obrázek 8.

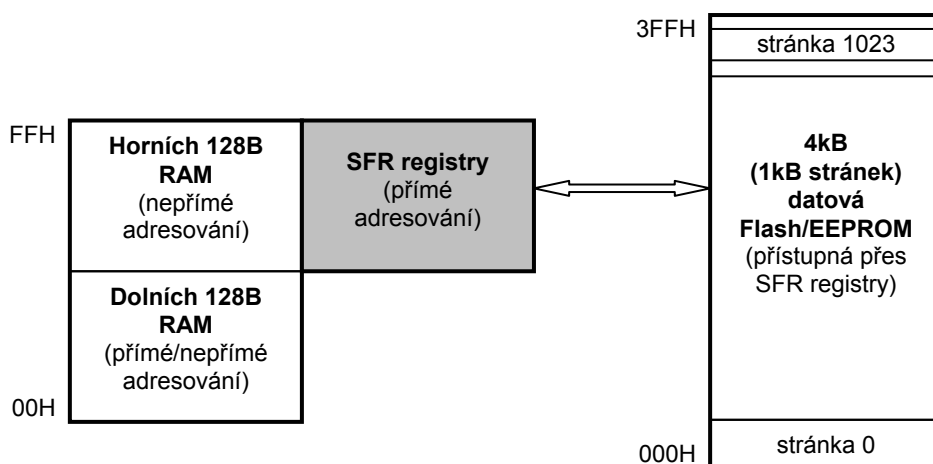


Obr. 8. Princip mapování programové paměti (pro konfiguraci 62kB)

Naprogramování této interní paměti EEPROM pro program je možné dvěma způsoby. Standardním jako u novějších klonů 80C51 nebo zcela jinak. Tzv. **In Circuit Programming** využívá sériovou linku a režim „download“ k zavedení programu do vnitřní programové paměti, bez nutnosti vyjímát mikrokonvertor z aplikace [9]. K nastavení režimu „download“ slouží pin PSEN, který se spojí přes odpor na GND a následně se provede RESET. Po spuštění programu WSD s příslušným přeloženým souborem HEX se programová paměť obvodu ADuC845 nejprve vymaže a následně naprogramuje novým obsahem. Po ukončení „downloadu“ se odstraní zkrat odporu na GND a provede se opět RESET. V tomto okamžiku začne ADuC845 vykonávat program podle nového obsahu programové paměti.

Datová paměť RAM a Flash/EEPROM

Mikrokonvertor ADuC845 má vestavěnou vnitřní paměť RAM o velikosti 256B s tím, že její horní polovina (od 80H výše) je dostupná pouze přes nepřímé adresování (@R0, @R1). Její obsluha je tedy stejná jako u standardu 80C51 (viz. kapitola 8051). Přímým adresováním se pak obsluhují **speciální funkční registry (SFR)**.



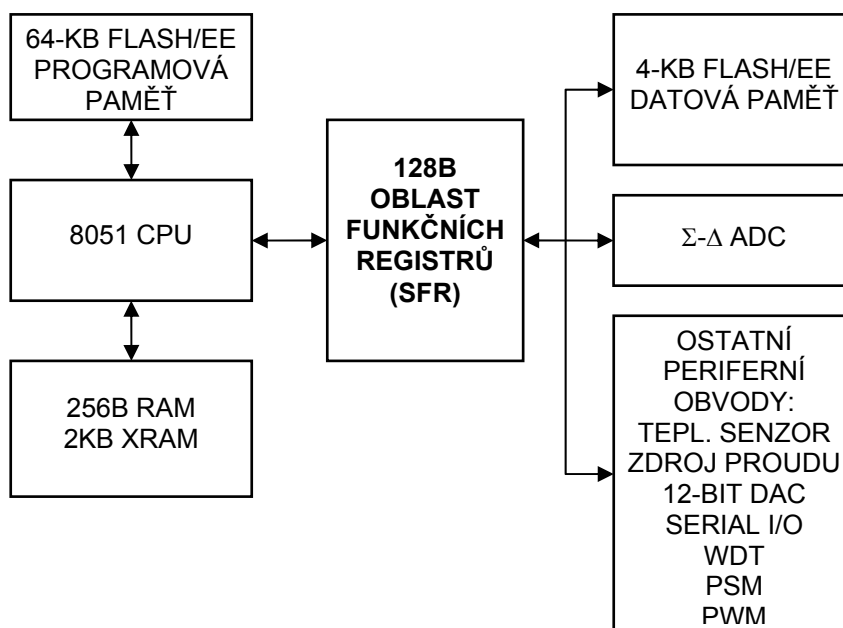
Obr. 9. Princip mapování datové paměti RAM a Flash/EEPROM

Vestavěná paměť Flash EEPROM je určena pro ukládání dat. Její velikost je 4kB. Vzhledem k požadavku přístupnosti k údajům pouze přes SFR registry, je její organizace rozdělena do 1kB stránek (00H až 3FFH) po 4 bytech. Adresa požadované stránky se zapisuje do SFR registru EADRL. Řídící příkazy pro činnost (mazání, zápis, čtení stránky, mazání celé paměti) do registru ECON a data jsou k dispozici nebo se ukládají do registru EDATA1 až EDATA4. Tato organizace přístupu do paměti Flash EEPROM je výhodná právě tím, že nespotřebuje žádné jiné příkazy, než jen zápis nebo čtení do/z registru. Obsah Flash paměti musí být však před zápisem nových dat vymazán.

ADuC845 navíc obsahuje interní datovou paměť XRAM o velikosti 2kB a možnost adresace vnější paměti pomocí příkazu MOVX, čímž lze dosáhnout až 16 MB paměťového prostoru.

Speciální funkční registry (SFR)

Oblast speciálních funkčních registrů je mapována v horních 128 bytech vnitřní paměti RAM, přístupné pouze přímým adresováním. Tato oblast obsahuje řídící, konfigurační a datové registry, které slouží jako rozhraní mezi CPU a vestavěnými periferiemi obvodu, jak ukazuje obrázek 10.



Obr. 10. Funkce SFR registrů

Mezi tyto registry patří dříve popsané základní registry standardu 80C51 jako například ACC-Střadač, PSW-Stavové slovo, ukazatele DTP a SP, apod.. ADuC845 navíc obsahuje sadu dalších pro obsluhu vestavěných periferních obvodů jako např. ADCMODE-režim AD převodníku, DACCON-řídící registr DA převodníku, ECON-řídící datový Flash/EE registr atd. Přehled těchto registrů je uveden v příloze 1.

Přerušovací systém

Mikrokonvertor ADuC845 má, oproti 80C51, celkem 11 zdrojů přerušení se dvěma prioritními úrovněmi. Veškerá konfigurace přerušovacího systému se provádí přes tři konfigurační registry, registr IE - známý z klonů 80C51, který je však rozšířen a sekundární registr IE2. Priorita přerušovacích signálů je pak nastavována v registru IP - Interrupt Priority. V tabulce II je uvedena priorita a adresy vektorů přerušení. Lze z ní vyčíst to, že všechny standardní vektory přerušení z 80C51 jsou shodné a na další paměťová místa se přidaly doplňkové vektory, specifické pro ADuC845. To samo o sobě umožňuje použití již vyvinutých programových modulů z rodiny 80C51 u tohoto procesoru bez úpravy [9].

Tabulka II. Adresy přerušení

Přerušení	Název typu přerušení	Adresa přerušení	Úroveň priority
PSMCON.5	Power Supply Monitor	0043H	1
WDS	WatchDog Timer	005BH	2
IE0	Vnější přerušení 0	0003H	3
RDY0/RDY1	Konec AD převodu	0033H	4
TF0	Čítač/časovač 0	000BH	5
IE1	Vnější přerušení 1	0013H	6
TF1	Čítač/časovač 1	001BH	7
ISPI/I2CI	SPI/I2C přerušení	003BH	8
RI/TI	Sériový kanál	0023H	9
TF2/EXF2	Časovač 2	002BH	10
TIMECON.2	Čítač časového intervalu	0053H	11

Vnitřní zapojení portů

Všechny porty mikrokonvertoru ADuC845 mají totožné funkce jako originální 80C51 a jsou slučitelné s CMOS rozhraním. S výjimkou portu P1 jsou všechny ostatní porty obousměrné. Avšak z důvodů sdílení některých pinů s vestavěnými periferními obvody, jsou vybrané piny portů upraveny a je možné je používat také jiným způsobem. Proto je port P1 pouze vstupní, neboť je současně vstupním portem osmi kanálů analogového multiplexeru pro A/D převodník. Čtení a zápis je ovládán přes odpovídající SFR registry. Jednotlivé piny portu P0, P2 a P3 mohou být nezávisle konfigurovány jako digitální výstupy nebo digitální vstupy nastavením odpovídajících bitů SFR registrů. Port 1 může být nastaven na analogové nebo digitální vstupy.

Port P0 - je standardně určen pro datovou sběrnici a dolní polovinu adresové sběrnice.

Port P1 - je pouze vstupní, pro funkci analogových vstupů není potřebné po resetování nic nastavovat, pro případ použití jako digitálního vstupu se musí zapsat do patřičného bitu portu 0. Piny P1.2 a P1.3 mohou být navíc použity jako sekundární referenční vstup (REFIN2+ REFIN2-). Piny P1.6 a P1.7 pak mohou sloužit jako zdroje budícího proudu (200μA).

Port P2 - je standardně určen pro horní polovinu adresové sběrnice. Při adresování vnější paměti RAM nad 64 kB pak i multiplexně pro adresy A16 - A24. Piny P2.5 a P2.6 mohou být použity jako PWM výstupy.

Port P3 - je standardní jako u 80C51 s některými nadstandardními funkcemi: P3.0 - RxD pro UART; P3.1 - TxD pro UART; P3.2 a P3.3 - externí HW vstup pro vnější přerušení ; P3.4 a P3.5 - vstupy časovačů T0 a T1; P3.6 a P3.7 - signály pro obsluhu vnějších pamětí RAM a ROM.

1.4.3 Vestavěné periferie

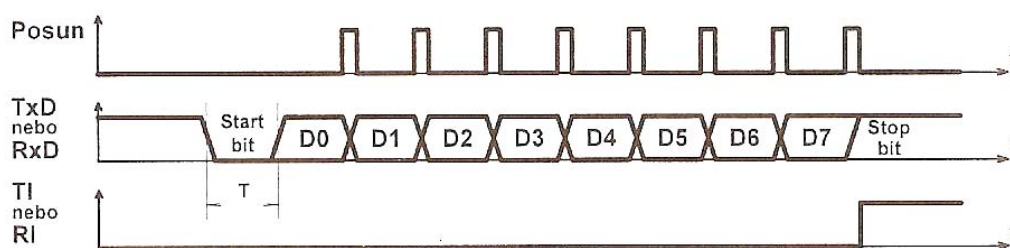
Vestavěných periferních obvodů je u mikrokonvertoru poměrně dost a jejich využití je velmi všestranné.

Sériový kanál

Pro sériovou komunikaci je mikrokontroler osazen plně duplexním UART portem, shodné konfigurace jako standard 80C51. Jeho programová obsluha je tedy shodná, pro definování přenosové rychlosti je využíván časovač TIMER0 nebo lze použít i časovač TIMER2. Pro víceprocesorové systémy je možné nastavit i 9-ti bitovou komunikaci, ale případný paritní bit je nutné spočítat programem [9]. Pro použití přenosu dat na standardních komunikačních rychlostech je nutné použít i odpovídající krystal oscilátoru, tedy největší možný je 11,0592 MHz. Pak je možné používat normalizovaných přenosových rychlosti až do 38400Bd.

Přijímací i vysílací registr je dostupný na stejné adrese se symbolickým označením **SBUF - registr sériového kanálu** v prostoru speciálních funkcí. Zápisem se naplňuje vysílací registr, čtením SBUF je přečtena hodnota z vyrovnávacího registru, do kterého byla přepsána z přijímacího registru po přijetí celého bytu. Sériový kanál může pracovat ve čtyřech módech v závislosti na naprogramovaném registru SCON a nejvyššího bitu v registru PCON.

Pro příklad uvádím mód 1 znázorněný na obr.11. Jedná se o 8-bitový UART mód. Ten je určen pro přenos 10-bitových slov, kde délka dat je 8 bitů. Slova jsou vysílána přes výstup TxD a přijímána přes vstup RxD v duplexním provozu. Slovo má 1 start bit a 1 stop bit. Byte dat se přepisuje do vstupního SFR registru SBUF. Stop bit se zapisuje do stavového bitu RB8 v SFR registru SCON.



Obr. 11. Časování sériového kanálu v módu 1

Sériové rozhraní SPI a I²C

Mikrokontroler ADuC812 je vybaven dvěma průmyslovými rozhraními, synchronním SPI a I²C. Obě tato rozhraní lze konfigurovat do režimu MASTER nebo SLAVE. Protože se piny konvertoru pro vstup a výstup dat na těchto rozhraních vzájemně sdílí, lze v jedné aplikaci použít buď SPI nebo I²C rozhraní. Blíže jsou tato rozhraní popsána v manuálu.

Pulsně šířkový modulátor (PWM)

ADuC845 nabízí programovatelný obvod Pulsně Šířkové Modulace, který lze nastavit do 6 různých režimů. Dva z nich nabízí PWM konfiguraci jako Σ - Δ DA převodník s rozlišením až 16 bitů. PWM využívá jeden kontrolní SFR registr PWMCON a čtyři datové PWM0L, PWM0H, PWM1L, PWM1H.

Watch Dog Timer (WDT) a Power Supply Monitor (PSM)

Pro zajištění bezporuchového chodu programu v mikroprocesoru se používají obvody, které monitorují chod programu tím, že tento program musí „projít“ tzv. kontrolními body. Pokud neprojde, vyvolá tento podpůrný obvod nějaké přerušení, které zajistí návrat programu do „normálního“ stavu [9]. Mikrokontroler ADuC845 je vybaven interním obvodem Watch Dog Timer, u něhož je možné programově nastavit délku odezvy pulsu RESET od 15.6 do 2000 ms [7]. Pro možnost programového zásahu při poklesu napájecího napětí je mikrokontroler vybaven tzv. monitorem napájecího napětí (PSM). Ten nastaví žádost o přerušení při poklesu napájecího napětí, podle nastavení v konfiguračním registru PSMCON. Lze vybrat pět uživatelsky nastavitelných úrovní napětí od +2,63 do +4,63V. Žádost o přerušení bude zrušena se zpožděním 256 ms po návratu napájecího napětí nad nastavenou rozhodovací úroveň. Tímto je nepřímě řečeno, že mikrokontroler bude funkční

i při napětí menším než 3V. V tomto okamžiku ale nesmí spolupracovat s žádným externím obvodem (paměť, periferní obvod atd.).

1.4.4 Návrh programového vybavení pro ADuC845

Za zcela nový přístup k ladění programového vybavení lze považovat možnosti, které nabízí firma Analog Devices. Kromě standardního překladu zdrojového textu, naprogramování a testování, nabízí firma Analog Devices k dispozici i variantu pouhého simulování v prostředí Windows na simulátoru ADSIM. Mezi další podporu využívání mikrokontroleru ADuC845 patří skutečnost, že veškeré vývojové prostředí pro Windows dává firma Analog Devices volně k dispozici na svých stránkách. To umožňuje potenciálním uživatelům si vše vyzkoušet bez výrazných nákladů. Firma také dodává vývojové kity – **QuickStart** (obr.12) a **QuickStart+** které umožňují zkoušení všech vestavěných periférií na připraveném HW zařízení. Sada **QuickStart** + navíc obsahuje veškeré nástroje nezbytné pro vývoj a testování aplikací využívajících mikrokonvertory, použitím jazyka C, a to pomocí připojeného obvodu s emulátorem a integrovaného vývojového prostředí Aspire.



Obr. 12. QuickStart vývojový kit

Vývojová sada ADuC845 QuickStart+

Rozšířená vývojová sada od Analog Devices s názvem **QuickStart+** obsahuje veškeré nezbytné nástroje pro vývoj embedded aplikací s obvody řady ADuC. Těmito nástroji jsou především:

- vývojová deska osazená patřičným mikrokonvertorem a pomocnými obvody (příloha 2)
- zdroje napětí a seriové kabely pro propojení s PC
- programové vybavení:
 - **Aspire** - Integrované Vývojové prostředí (manažer projektů, editor kódu, integrovaný assembler, simulátor, základní C kompilér, a další utility pro nahrávání programu do paměti mikrokontroleru)
 - Příklady programových kódů (v assembleru i C kódu)
- Desku Emulátoru **Accutron Spear™** – umožňující programování v jazyku C
- Dokumentace

Po sestavení těchto součástí, instalaci a připojení k PC pomocí seriového kabelu je tak k dispozici kompletní integrované prostředí pro rychlý a efektivní návrh embedded aplikací ve standardním C jazyku. Pro řešení složitějších úloh je navíc potřeba instalovat pokročilejší kompilátor, jako je například C51 firmy Keil.

2 ASPEKTY MĚŘENÍ A ZPRACOVÁNÍ SIGNÁLŮ

Následující kapitola uvádí základní aspekty měření a zpracování senzorických signálů v průmyslové praxi se zaměřením na zpracování signálů odporových teplotních senzorů. Za tímto účelem je zde uvedeno řešení připojení teplotního odporového senzoru Pt100 včetně linearizace a kalibrace, nabízené společností Analog Devices. Pro pochopení problematiky jsou zde uvedeny také základní pojmy z oblasti technických prostředků měření v průmyslové automatizaci, včetně analýzy chyb a nejistot, které měření ovlivňují.

2.1 Technické prostředky měření

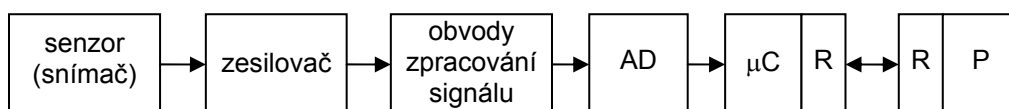
Každý systém integrované automatizace (SIA) vyžaduje mít k dispozici údaje o parametrech a o stavu řízených nebo sledovaných procesů. Tyto funkce zajišťují technické prostředky měřicích obvodů [1]. Obecně jsou nazývány jako prostředky pro získávání informací. Vytváří vazbu mezi technickým nebo technologickým zařízením a systémem automatizace. Základním prvkem těchto prostředků bývá senzor.

2.1.1 Senzor

Senzor je funkční prvek tvořící vstupní blok měřicího řetězce, který je v přímém styku s měřeným prostředím. Citlivá část senzoru se občas označuje také jako čidlo. Senzor jako primární zdroj informace snímá sledovanou fyzikální, chemickou nebo biologickou veličinu a dle určitého definovaného principu ji transformuje na měřicí veličinu - nejčastěji na veličinu elektrickou.

2.1.2 Měřicí řetězec

Základní blokové schéma měřicího řetězce obsahující typické prostředky měření je na obrázku 13.



Obr. 13. Blokové schéma měřicího řetězce se senzorem

AD – analogově číslicový převodník, včetně multiplexerů apod., μ C – mikropočítač (mikrořadič), R – komunikační rozhraní, P – nadřazený výpočetní systém

Transformovaný elektrický signál je obvykle nutno zesílit. Při zesílení signálu je nezbytné zajistit dostatečný odstup signálu od šumu senzoru a zesilovače, a od parazitních signálů (rušení elektrickým, magnetickým a elektromagnetickým polem, působením zemních smyček apod.) působících jak na senzor, tak na zesilovač. Uvedené požadavky se realizují speciálními zapojeními zesilovačů, frekvenčním omezením signálu, modulací elektrické nebo přímo neelektrické veličiny s následnou synchronní demodulací, nebo číslicovým zpracováním signálu. Další zpracování signálu je řešeno buď analogovými obvody, nebo po analogově-číslcovém převodu číslicovou technikou, vybavenou mikroprocesorem nebo mikrořadičem [3]. Právě ve zpracování elektrických signálů je podstatná výhoda v transformaci neelektrické veličiny na elektrickou, neboť se signály lze provádět libovolné matematické operace (sčítání, odečítání, násobení, dělení, derivace, integrace aj.) a signálové operace (autokalibrace, analogová nebo číslicová filtrace, frekvenční analýza rychlou Fourierovou transformací, unifikace analogového nebo číslicového signálu pro přenos dat apod.).

Analogový výstup měřicího řetězce je v nejjednodušším případě realizován přímoukazujícím přístrojem nebo zapisovačem, nebo pro přenos dat blokem unifikace signálu, modulátorem, případně optoelektronickým členem pro přenos modulovaného signálu světlovodem apod. [3]. Unifikace analogových signálů je standardizována na hodnoty 0..10V, 0..20 mA, 4 - 20 mA.

Při číslicovém zpracování signálu je výstup buď tvořen číslicovým měřicím přístrojem vybaveným rozhraním pro dálkové zpracování dat, nebo přímo rozhraním (tj. společnou hranicí s definovanými charakteristikami fyzického, signálového, funkčního a procedurálního propojení). U senzorů se nejčastěji používá sériové rozhraní, a to RS-232 a RS-485. Zatímco u rozhraní RS-232 je pro obousměrný spolehlivý provoz nutno použít pět a více vodičů a dosah přenosu dat je při rychlosti 20 kb/s 15 m, u RS-485 vystačí stíněná stočená dvojlinka při rychlosti přenosu 20 Mb/s do vzdálenosti 1200 m. Další výhodou RS-485 je symetrický vstup, který potlačuje součtové rušivé signály a potřebuje jediné napájení +5 V. Součástí každého rozhraní je komunikační protokol definující přenos naměřených dat, adres, řídicích a stavových signálů (SPI, I²C, ABUS, CAN, PROFIBUS, aj.).

2.1.3 Vlastnosti technických prostředků měření

Vlastnosti technických prostředků měření jsou charakterizovány především **statickými a dynamickými parametry** [1]. Statické parametry popisují provozně technické vlastnosti přístrojů a jejich jednotlivých prvků v ustáleném stavu. Mezi základní statické charakteristiky patří:

- **měřicí rozsah** jako rozsah hodnot měřené veličiny, kdy platí pro výstupní signál uvedená přesnost měření, statická charakteristika popisující závislost hodnot výstupního signálu na hodnotě vstupní veličiny,
- **statická chyba** jako souhrn všech chyb uplatňujících se při funkci přístroje v ustáleném stavu
- **absolutní chyba** jako difference mezi indikovanou hodnotou a konvenčně správnou hodnotou signálu
- **relativní chyba** jako poměr absolutní chyby ke vztažné hodnotě (např. maximální rozsah), často udávaná v procentech
- **nejistota měření**: předpokládané chyby všech vlivů na výsledek zkoušek
- **hystereze** znamenající rozdíl u hodnot vzestupné a sestupné charakteristiky
- **reprodukovatelnost** vyjadřující soulad charakteristiky, získané opakovaně za rozdílných podmínek (různými měřicími přístroji, měřicími metodami při působení rozdílných ovlivňujících veličin, ve velkých časových odstupech),
- **nelinearita** jako nepřímková závislost charakteristiky,
- **saturace** jako limita charakteristiky
- **citlivost** jako strmost charakteristiky
- **rozlišitelnost** jako nejmenší změna charakteristiky, která ještě může být indikována
- **mrtvé pásmo** jako necitlivost v určitém pásmu charakteristiky
- **vstupní impedance** jako hodnota zdánlivého elektrického odporu vstupu prvku
- **výstupní impedance** jako hodnota zdánlivého elektrického odporu výstupu prvku
- **správnost** jako těsnost shody mezi aritmetickým průměrem výsledků a konvenční hodnotou
- **shodnost** jako těsnost shody charakteristik, získaných jako nezávislé výsledky zkoušek za určitých podmínek ,
- **třída přesnosti** vyznačující číslo ze zvolené řady klasifikující přesnost měřidla jako největší dovolená chyba.

Při sledování časových průběhů proměnných veličin od počátku změny do ustálení odezvy je daný průběh procesu měření charakterizován **dynamickými** parametry. Příčinou přechodových dějů jsou hmotné, tepelné, elektrické a magnetické kapacity, setrvačnosti,

tlumení hmot měřících prvků, impedance, rychlosti změn fyzikálních a chemických parametrů měřených medií apod.. Dynamické parametry jsou definovány časovou konstantou nebo také přechodovou funkcí, váhovou funkcí, frekvenční charakteristikou, frekvenční funkcí, dynamickou chybou.

2.2 Chyby a nejistoty v měření

V současné době se při technických měřeních postupně přechází k novým metodám vyjadřování odchylek. Dosavadní chyby měření jsou v souladu s mezinárodními předpisy ISO, BIPM, OIM a IEC i našimi normami řady TPM 005x nahrazovány nejistotami měření z důvodu zajištění jednotného vyjadřování [6].

2.2.1 Chyby v měření

Chyby měření se vyjadřují v absolutních nebo relativních hodnotách. Podle jejich působení lze chyby rozdělit na systematické, náhodné a hrubé. Podle svého zdroje se rozdělují na chyby přístroje, metody, pozorování a vyhodnocení. Jako chyba absolutní Δx se označuje rozdíl mezi hodnotou naměřenou x_m a skutečnou x_s . Podělí-li se chyba absolutní skutečnou hodnotou dostane se poměrné vyjádření chyby, tj. chyba relativní δ_x . Platí tedy:

$$\Delta_x = x_m - x_s \quad (1)$$

$$\delta_x = \frac{\Delta_x}{x_s} = \frac{x_m - x_s}{x_s} \quad (2)$$

Systematické chyby jsou při stálých podmínkách také stálé co do velikosti i znaménka a svým působením „systematicky“ ovlivňují výsledek měření. Ke stanovení jejich velikosti zpravidla postačí vztah (1). Systematické chyby jsou sympatické tím, že je lze z velké části určit a jejich vliv je možné zmenšit např. pomocí korekce, kompenzace apod. [6].

Náhodné chyby působí zcela nahodile, jsou těžko předvídatelné a nelze je vyloučit. Při opakování měření se mění jejich velikost a znaménko. Náhodnou chybu v klasické teorii chyb nejčastěji zastupuje směrodatná odchylka výběrového souboru s nebo směrodatná odchylka aritmetického průměru s_x .

$$s = \sqrt{\frac{\sum_{i=1}^n \Delta^2 x_i}{n-1}} = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n-1}} \quad (3)$$

$$s_{\bar{x}} = \frac{s}{\sqrt{n}} = \sqrt{\frac{\sum_{i=1}^n \Delta(x_i - \bar{x})^2}{n(n-1)}} \quad (4)$$

Hrubé chyby jsou z předchozího pohledu zcela nevyzpytatelné. Měření zatížené hrubou chybou znehodnotí celý experiment, a proto naměřené hodnoty, které výrazně „vybočují z řady“, což bývá velmi často projevem tohoto druhu chyby, se vyloučí z tohoto zpracování. Omezit riziko jejich výskytu lze důsledným dodržováním příslušných měřicích postupů, podmínek měření a pozornosti obsluhy.

Výsledná chyba měření je vyjadřována jako součet systematické a náhodné složky, což lze napsat:

$$\Delta_C = e + \varepsilon \quad (5)$$

kde systematická složka $e = \bar{x} - x_s$ a náhodná složka $\varepsilon = s$

Co se týče původu chyby, chyby přístrojů jsou způsobeny nedokonalostmi použitých měřicích prostředků, které mohou vznikat ve výrobě, montáži a popř. i opotřebením.

Svou roli zde sehrává také změna charakteristik a parametrů přístroje v čase. Dalším zdrojem chyb je nevhodná instalace nebo uložení přístroje na pracovním místě. Chyby metod mají svůj původ v nedokonalosti či zjednodušení použité měřicí metody. Chyby pozorování jsou do měření vnášeny jako chyby osobní, zapříčiněné buď nedokonalostí smyslu pozorovatele nebo jeho nesoustředěností. Chyby, které mají svůj původ ve vyhodnocení jsou často jako výpočtové, vznikající v důsledku aplikování přibližných vztahů, linearizace, interpolace, extrapolace, zaokrouhlování, nedostatečným vyčíslením konstant apod.

2.2.2 Nejistoty v měření

Pojem nejistota měření je označení pro parametr související s výsledkem měření a charakterizuje rozsah hodnot, které je možno racionálně přiřadit k měřené veličině. Nejistota měření se skládá z několika dílčích složek. Ke stanovení jejich velikosti jsou principiálně k dispozici tyto dvě metody:

- 1) statistické zpracování naměřených údajů (metoda typu A)
- 2) jiné než statistické zpracování naměřených údajů (metoda typu B)

Metoda typu A

Vyhodnocování standardních nejistot vstupní veličiny metodou typu A vychází ze statistické analýzy opakované série měření. Je-li nezávislých, stejně přesných pozorování ($n > 1$), bude odhad výsledné hodnoty reprezentován hodnotou výběrového průměru (aritmetického průměru), vypočítaného stejně jako u případných chyb. Nejistota příslušná k odhadu se určí jako směrodatná odchylka této výsledné hodnoty, tedy výběrového průměru. Nejistota se zde zpravidla značí u_{Ay} a s použitím vztahu (2) je možné napsat:

$$u_{Ay} = s_{\bar{y}} = \frac{s_y}{\sqrt{n}} = \sqrt{\frac{1}{n(n-1)} \sum_{i=1}^n (y_i - \bar{y})^2} \quad (6)$$

Tato nejistota je způsobena kolísáním naměřených údajů. V případě malého počtu měření ($n < 10$) je hodnota určena pomocí vztahu (6) málo spolehlivá. Proto by bylo zapotřebí tuto nejistotu (způsobenou kolísáním naměřených hodnot) odhadnout metodou typu B na základě jiných informací, než jsou současné naměřené hodnoty.

Metoda typu B

Vyhodnocování standardních nejistot vstupní veličiny metodou typu B je založeno na jiných než statistických přístupech k analýze série pozorování. Pomocí metody typu B je možno odhadnout i vliv zcela náhodných chyb. Standardní nejistota se odhaduje pomocí racionálního úsudku na základě všech možných dostupných informací. Nejčastěji se používají:

- údaje výrobce měřicí techniky
- zkušenosti z předchozích sérií měření
- zkušenosti s vlastnostmi chování materiálů a poznatky o nich
- údaje získané při kalibraci a z certifikátů
- nejistoty referenčních údajů v příručkách

Při určování nejistoty metodou typu B se vychází z dílčích nejistot jednotlivých zdrojů u_{Bzj} . Je-li známa maximální odchylka j -tého zdroje nejistoty Z_{jMAX} , určí se nejistota u_{Bzj} podle vztahu:

$$u_{Bzj} = \frac{z_{jMAX}}{k} \quad (7)$$

kde k - součinitel vycházející ze zákona rozdělení, kterým se příslušný zdroj nejistoty řídí. V některých případech (např. z kalibračního certifikátu měřidla) může však být známa přímo hodnota standardní nejistoty u_{Bzj} . Výsledná nejistota se určí metodou B:

$$u_{By} = \sqrt{\sum A_j^2 \cdot u_{Bzj}^2} \quad (8)$$

u_{Bzj} - nejistoty jednotlivých zdrojů a A_j jejich součinitele citlivosti jednotlivých zdrojů nejistoty, které jsou známy, nebo se určí jako parciální derivace výstupní funkce podle příslušné vstupní veličiny:

$$A_j = \frac{\partial y}{\partial x_j} = \frac{\partial f(x_1, x_2, \dots, x_j, \dots, x_n)}{\partial x_j} \quad (9)$$

V praxi se jen zřídka vystačí s jedním nebo druhým typem nejistoty samostatně. Pak je zapotřebí stanovit výsledný efekt kombinovaných nejistot měření obou typů, A i B.

Výsledná kombinovaná nejistota veličiny se označuje u_{cy} a určuje se jako odmocnina ze součtu čtverců obou typů nejistot A a B podle vztahu:

$$u_{Cy} = \sqrt{u_{Ay}^2 + u_{By}^2} \quad (10)$$

Jako zdroje nejistot lze označit veškeré jevy, které nějakým způsobem mohou ovlivnit neurčitost jednoznačného stanovení výsledku měření, a tím vzdalují naměřenou hodnotu skutečné. Značnou roli zde také sehrává skutečnost, zda jde o měřicí metody přímé nebo nepřímé. Na nejistoty působí výběr měřících přístrojů analogových nebo číslicových, použití různých filtrů, vzorkovačů a dalších prostředků v celé trase přenosu a úpravy měřicího signálu.

K nejistotám výrazně přispívají rušivé vlivy prostředí v nejširším slova smyslu:

- nedokonalá či neúplná definice měřené veličiny nebo její realizace
- nevhodný výběr přístroje
- nevhodný postup při měření
- zaokrouhlování konstant

- neznámé nebo nekompenzované vlivy prostředí
- nedodržení shodných podmínek při opakovaných měřeních
- nepřesnost etalonů a referenčních materiálů

Některé z výše uvedených zdrojů se projevují výrazněji v nejistotách vyhodnocovaných metodou typu A, jiné při použití metody typu B. Mnohé zdroje ale mohou být příčinou obou skupin nejistot, a zda je největší nebezpečí v podobě zapomenutí jedné ze složek, což může mít za následek velmi zkreslený výsledek.

2.3 Měření teploty pomocí RTD senzorů

Odporové teplotní senzory teploty (RTD) patří mezi nejčastěji používané senzory teploty. U těchto senzorů vnější podnět (teplota) mění elektrický odpor hmoty senzoru [2]. Při vyhodnocování těchto změn v elektronických odvodech musíme používat pomocné napájecí napětí a proto se jedná o pasivní senzory.

2.3.1 Princip RTD senzorů

Princip odporových kovových senzorů spočívá ve využití závislosti odporu kovů na teplotě [1],[2]. Odpor senzoru se v měřicím členu transformuje na elektrickou veličinu, kterou je obvykle elektrické napětí.

Pro provozní kovové teploměry lze v rozsahu 0°C až 100°C použít vztah

$$R = R_0(1 + \alpha t) \quad (11)$$

kde α je teplotní součinitel odporu [K^{-1}]

R_0 odpor teploměru při teplotě 0°C.

t je teplota teploměru

Pro větší rozsahy teplot platí nelineární rovnice v závislosti na použitém kovu. Nejrozšířenějším materiálem čidla je platina. Kromě platiny se používají nikl včetně různých niklových slitin, měď, molybden, pro extrémně nízké teploty slitiny Rh-Fe, Pt-Co

aj.. Měřicí odpory tvořící základní konstrukční část teploměru se vyrábějí drátkovou, tenkovrstvou a tlustovrstvou technologií.

2.3.2 Platinové odporové senzory

Patří mezi jedny z nejstarších senzorů teploty. Jsou dodávána v podobě vinutých odporů z čisté platiny na keramických nebo skleněných dříčích. V současné době se dodávají též jako keramické destičky s napařenou vrstvou kovu. Vyznačují se téměř lineární převodní charakteristikou v širokém rozmezí teplot a jejich vlastnosti jsou standardizovány [2]. Typická hodnota odporu při 0 °C je 100 Ω (označení Pt100). Byly standardizovány i další dvě hodnoty, a to 500 Ω, a 1000 Ω. Jejich závislost na teplotě je tatáž jako u čidla Pt100, pouze příslušná relativní změna odporu není násobena 100 Ω, ale hodnotou 500 Ω nebo 1000 Ω. Ve světě se dále vyrábějí platinové senzory se základní hodnotou odporu 50 , 200 a 2000 Ω . Rozsah použití běžných platinových senzorů je podle provedení od –200 °C do +850 °C.

2.3.3 Funkční závislost senzorů Pt100

Závislost odporu platinového odporového senzoru teploty na teplotě je dle normy popsána dvěma rozdílnými polynomickými rovnicemi: jedna pro teploty pod 0°C a druhá pro teploty nad 0°C. Tyto rovnice jsou

$$R_{RTD}(t) = R_0 [1 + At + Bt^2 + C(t - 100^\circ\text{C})t^3] \quad (\text{pro } t \leq 0^\circ\text{C}) \quad (12)$$

$$R_{RTD}(t) = R_0 [1 + At + Bt^2] \quad (\text{pro } t \geq 0^\circ\text{C}) \quad (13)$$

kde : t = teplota senzoru [°C]

$R_{RTD}(t)$ = funkční závislost odporu na teplotě

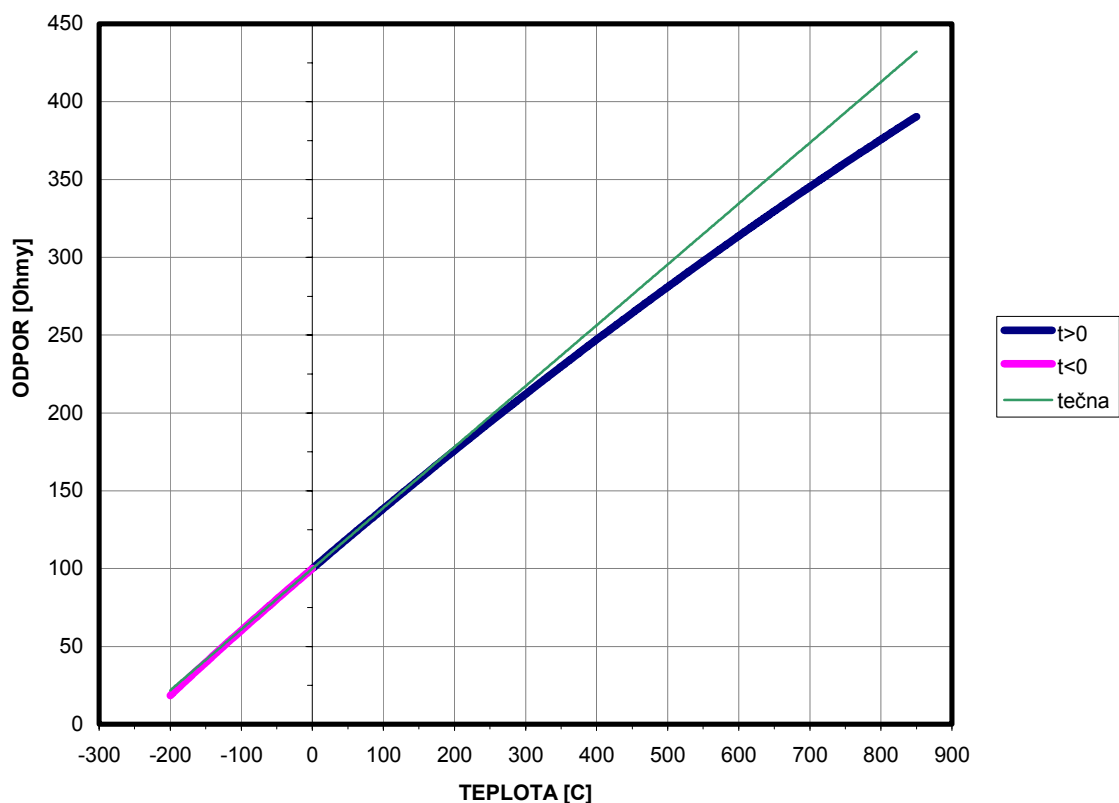
R_0 = hodnota odporu senzoru při 0°C (pro senzor Pt100 = 100Ω)

$A = 3.9083 \times 10^{-3} \text{ }^\circ\text{C}^{-1}$

$B = -5.775 \times 10^{-7} \text{ }^\circ\text{C}^{-2}$

$C = -4.183 \times 10^{-12} \text{ }^\circ\text{C}^{-4}$

Obrázek 14 znázorňuje graf závislosti odporu platinového odporového senzoru na teplotě spolu s tečnou v bodě 0°C pro porovnání.



Obr. 14. Závislost odporu senzoru na teplotě

Pro teplotu 100 °C vychází typická hodnota 138,5 Ω.

2.3.4 Toleranční třídy a konstrukce

V toleranční třídě A mají meze tolerančního pásma při 0 °C hodnotu $\pm 0,15$ °C a ve třídě B pak $\pm 0,3$ °C. Pro přesnější měření firmy obvykle třídí čidla do třídy 1/3 B, kde vychází tolerance $\pm 0,10$ °C [2]. Jednotlivá toleranční pásma jsou vymezena těmito křivkami:

- toleranční pásmo pro třídu A $|\Delta T| = 0,150 + 0,002 \cdot t$
- toleranční pásmo pro třídu B $|\Delta T| = 0,300 + 0,005 \cdot t$
- toleranční pásmo pro třídu 1/3 B (DIN 1/3 B) $|\Delta T| = 0,100 + 0,0017 \cdot t$

Standardizace a reprodukovatelnost hodnot předurčují Pt100 pro velmi přesná měření. Dlouhodobá stabilita se u provozních (tj. nikoli tzv. meteorologických) senzorů pohybuje kolem 0,05 % / 1000h při jmenovité hodnotě odporu.

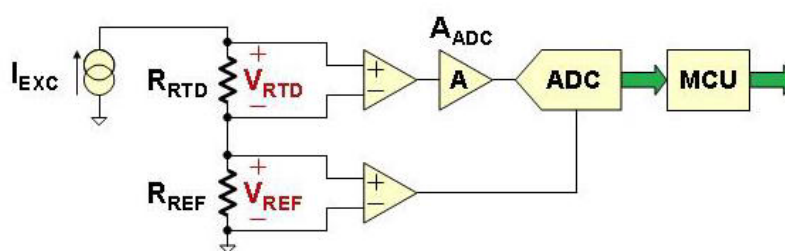
Senzory jsou konstruovány se dvěma, třemi nebo čtyřmi vývody, případně jsou dvojité. Propojení senzoru se svorkovnicí snímače přidává parazitní odpor, který zanáší chybu až několik °C. Pro měření, kde tuto chybu nelze tolerovat je nutné použít senzor se čtyřmi vývody. Nelinearita platinových senzorů se odstraňuje obvodově, nebo až při číslicovém zpracování signálu.

2.4 Zpracování signálu RTD senzorů pomocí obvodu ADuC

Dosažení nejlepších výsledků při měření vyžaduje přesné zpracování signálu, AD konverzi, linearizaci a kalibraci. Řada ADuC8XX mikrokonvertorů firmy Analog Devices obsahuje prvky jako například 24-bitový AD převodník s mikropočítačem 8052, integrované v jediném obvodu spolu s prvky pro přesné zpracování signálu, vhodné pro přímé napojení na odporové senzory teploty [8].

2.4.1 Připojení senzoru

Odporový teplotní senzor (RTD) je teplotně závislý odpor, jehož závislost se dá definovat. Funkce závislosti teploty na odporu je nelineární. Veškeré nelinearity jsou upraveny číslicově pomocí mikrokontroleru, nejprve je však nutný AD převod signálu[8]. Obvyklé schéma zapojení pro takový senzor je zobrazeno na obrázku č. 15.



Obr. 15. Schéma zapojení odporového teplotního senzoru

- kde I_{EXC} je proudový zdroj
 R_{RTD} je odporový teplotní senzor
 R_{REF} je referenční odpor
 A_{ADC} je zesílení převodníku

Proudový zdroj zde vybudí sériově zapojené odpory a tím je generováno napětí V_{RTD} přivedené na diferenční vstup AD převodníku a referenční napětí V_{REF} podle vztahu:

$$V_{RTD} = I_{exc} \times R_{RTD} \quad (14)$$

$$V_{REF} = I_{exc} \times R_{REF} \quad (15)$$

Normalizovaný výstup AD převodníku (v rozsahu 0 až 1) je pak poměr vstupního napětí a referenčního napětí vynásobený zesílením A_{ADC} :

$$ADC_{NORM} = A_{ADC} \times \frac{V_{RTD}}{V_{REF}} = \frac{I_{EXC} \times R_{RTD}}{I_{EXC} \times R_{REF}} = A_{ADC} \times \frac{R_{RTD}}{R_{REF}} \quad (16)$$

Proudový zdroj I_{EXC} se ve výše uvedené rovnici vykrátí, což znamená že i když se hodnota proudu změní nebo je nepřesná, tak výsledek AD převodu vždy přímo odpovídá poměru odporu RTD a referenčního odporu. Výběrem správného a kvalitního referenčního odporu tak lze docílit toho, že hodnota RTD bude známa s vysokou přesností, i když je použit nestabilní zdroj proudu [8].

Výpočet hodnoty odporu z výstupu AD převodníku

Úpravou předchozího vzorce dostanu vztah pro výpočet odporu senzoru

$$R_{RTD} = ADC_{NORM} \times \frac{R_{REF}}{A_{ADC}} \quad (17)$$

Výše uvedená rovnice tedy postačuje k odvození hodnoty odporu teplotního senzoru přímo z výsledku daného AD převodem. Z takto získané hodnoty odporu pak lze odvodit teplotu senzoru podle funkce závislosti odporu senzoru na teplotě.

2.4.2 Linearizace

Vztahy (12),(13) definují odpor senzoru v závislosti na teplotě $R_{RTD}(t)$. Pro vytvoření obvodu pro měření teploty je však nutné odvodit funkci závislosti teploty na odporu $T_{RTD}(r)$, která je zatížena nelinearitami přechodové charakteristiky odporového senzoru. K linearizaci takové funkce je třeba použít některou z následujících technik, které doporučuje firma Analog Devices.

Při řešení embedded aplikací se nejčastěji používají tři základní metody linearizace [8]: Přímá matematická metoda, Jednoduchá lineární aproximace, Po částech lineární aproximace. Všechny tyto metody jsou velmi vhodné pro aplikace využívající

Mikrokontrolery. Tabulka III popisuje výhody a nevýhody těchto metod, a uvádí vhodné případy jejich použití.

Tabulka III. Metody linearizace

Metoda	Výhody	Nevýhody	Hodnocení
Přímá matematická metoda	• Velmi přesná • Nevyžaduje vyhledávací tabulku	• Vyžaduje matematickou knihovnu (obvykle >1kB) • Relativně pomalá (10 až 50 ms*)	Vhodná pokud matematická knihovna je již vyžadována jinou funkcí aplikace
Jednoduchá lineární aproximace	• Velmi rychlá (< 1ms*) • Zabírá málo paměti • Poměrně přesná v úzkém měřicím rozsahu • Nevyžaduje vyhledávací tabulku • Nevyžaduje matematickou knihovnu	• Malá přesnost v celém měřicím rozsahu	Užitečná pokud je nedostatek paměti a teplotní rozsah měření je úzký
Po částech lineární aproximace	• Rychlá (<5ms*) • Může být velmi přesná • Nevyžaduje matematickou knihovnu	• Větší nároky na paměť než metoda Jednoduché lineární aproximace • Větší nároky na paměť než Matematická metoda pokud je matematická knihovna vyžadována jinou funkcí	Nejpoužívanější metoda (>90% embedded aplikací)

*Udávaná rychlost reprezentuje neměřené hodnoty při taktovací rychlosti 12.58 MHz

Přímá matematická metoda

V části 2.3.3. je vyjádřen explicitně vztah popisující závislost odporu odporového senzoru teploty na teplotě $R_{RTD}(t)$. Přímá matematická metoda využívá vztah závislosti teploty na odporu přímým odvozením z této rovnice.

Pro teploty nad 0°C lze jednoduše odvodit kvadratickou rovnici, která má dvojí řešení. Výsledným hodnotám které jsou kladné odpovídá jedno:

$$T_{RTD}(r) = \frac{-A + \sqrt{A^2 - 4B\left(1 - \frac{r}{R_0}\right)}}{2B} \quad (18)$$

kde výrazy A, B a R_0 jsou definovány dříve a r je hodnota odporu senzoru.

Pro řešení této funkce v reálném čase je vhodné vztah upravit do tvaru konstant aby se omezil počet matematických operací:

$$T_{RTD}(r) = \frac{Z_1 + \sqrt{Z_2 + Z_3 \times r}}{Z_4} \quad (19)$$

kde:

$$Z_1 = -A = -3.9083 \times 10^{-3}$$

$$Z_2 = A^2 - 4B = 17.58480889 \times 10^{-6}$$

$$Z_3 = \frac{4 \times B}{R_0} = -23.10 \times 10^{-9}$$

$$Z_4 = 2 \times B = -1.155 \times 10^{-6}$$

Tato rovnice je tedy platná pouze pro teploty 0°C a vyšší, a jelikož jde o matematické řešení, je 100% přesné v celém kladném rozsahu. Nejistoty způsobené zaokrouhlováním při použití 32-bitové plovoucí čárky v 8051 C kódu vykazují zhruba +0.0001°C/-0.0005°C [8]. Použitím rutiny *RTDmath.c* běžící na mikrokontroleru ADuC845 s taktovací rychlostí 12.58 MHz, je tento výpočet vykonán za méně než 4.2 ms [8].

Vztah závislosti odporu pro teploty nižší než 0°C je polynomiální rovnice čtvrtého řádu a je tedy nepraktické řešit ji jako přímý výraz popisující teplotu v závislosti na odporu [8]. Použitím počítačových matematických programů jako je například Mathematica[®] však můžeme najít blízkou aproximaci k této inverzní přechodové funkci. Řešením jsou pak následující polynomiální výrazy druhého až pátého řádu pro teploty 0°C a méně:

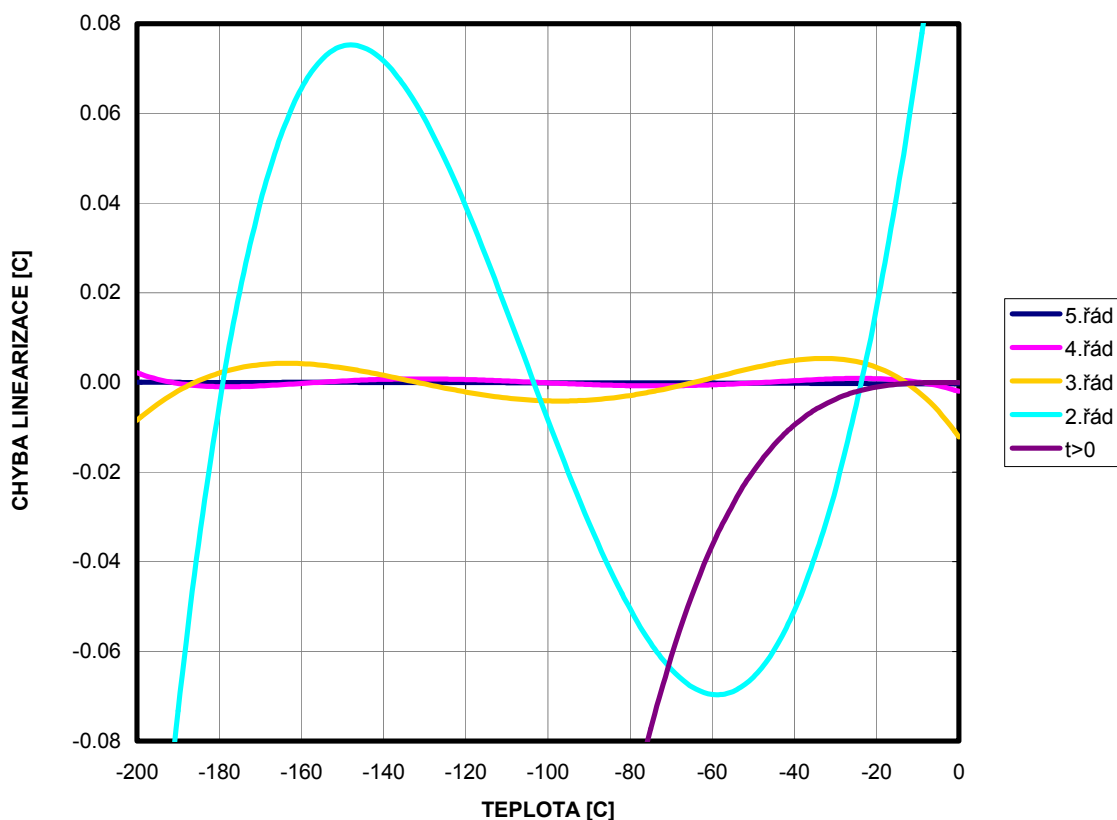
$$T_{RTD}(r) = -2420199 + 2.222812 \times r + 2.585885 \times 10^{-3} \times r^2 - 4.826040 \times 10^{-6} \times r^3 - 2.81834 \times 10^{-8} \times r^4 + 1.524259 \times 10^{-10} \times r^5$$

$$T_{RTD}(r) = -2419610 + 2.216253 \times r + 2.854064 \times 10^{-3} \times r^2 - 9.912120 \times 10^{-6} \times r^3 + 1.705183 \times 10^{-8} \times r^4$$

$$T_{RTD}(r) = -2420906 + 2.227625 \times r + 2.517790 \times 10^{-3} \times r^2 - 5.861951 \times 10^{-6} \times r^3$$

$$T_{RTD}(r) = -2429703 + 2.283841 \times r + 1.472734 \times 10^{-3} \times r^2$$

Obrázek 16 pak znázorňuje chyby v závislosti na teplotě pro jednotlivé polynomy spolu s polynomem použitým pro kladné teploty (pro srovnání).



Obr. 16. Graf chyb polynomiálních funkcí

V grafu je patrné, že chyba lineárního vztahu při použití polynomu pro kladné teploty je v oblasti blízké nule mnohem menší než u polynomů 2., 3. a 4. řádu. První rovnice (5. řádu) je nejpreciznější, ale je časově nejnáročnější co se výpočtů týká. Naopak poslední (2. řádu) je nejméně přesná, pro výpočet je však nejrychlejší. Tabulka IV popisuje některé tyto charakteristiky, naměřené při běhu rutiny *RTDmath.c* na mikrokonvertoru ADuC845, při taktovací rychlosti 12.58 MHz [8].

Tabulka IV. Charakteristiky polynomiálních funkcí

Polynom	Maximální čas výpočtu	Přesnost	Práh
5.řád	41 ms	+0.0001°C/-0.00005°C	0°C/100 Ω
4.řád	31 ms	+0.0022°C/-0.001°C	-8.75°C/96.6 Ω
3.řád	21 ms	+0.0053°C/-0.0085°C	-12.5°C/95.1 Ω
2.řád	11 ms	+0.075°C/-0.17°C	-70.5°C/72.1 Ω

Hranici při které je chyba při použití vztahu pro kladné hodnoty menší než chyba způsobená aproximací charakterizuje sloupec *Práh* v tabulce. Pod touto hranicí je již

použití polynomu určeného pro záporné teploty vhodnější. Nevýhodou této matematické metody je nutnost práce s plovoucí desetinnou čárkou a použití matematických funkcí.

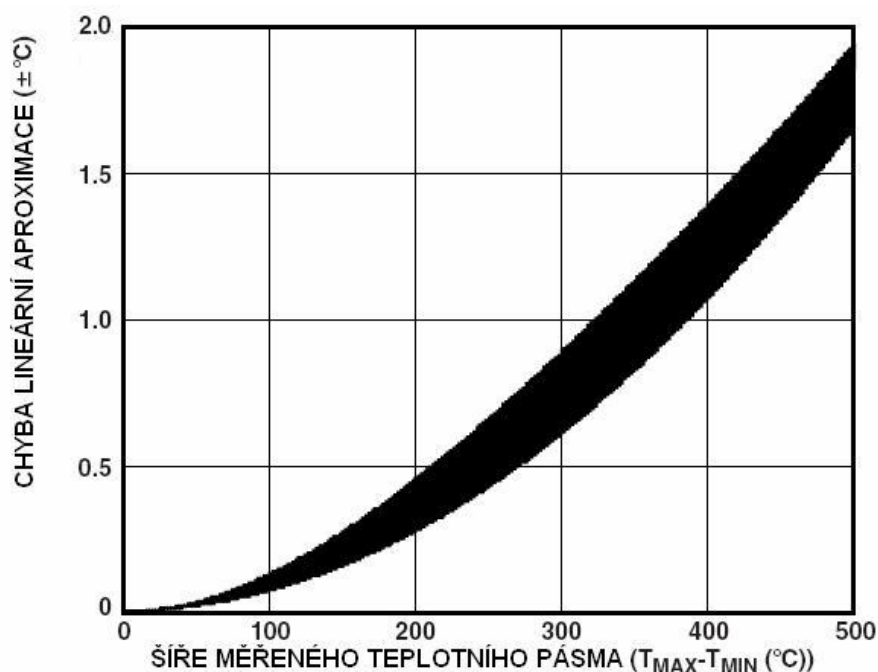
Jednoduchá lineární aproximace

Jak ukazuje obrázek číslo 14, tak závislost odporu senzoru na teplotě se při malých teplotních rozsazích blíží lineární charakteristice, a je tedy vhodné ji lineárně aproximovat. V případech kdy je rozsah měření malý, tak může tato linearizace poskytnout uspokojivé výsledky. Například při obvyklém rozsahu měření v průmyslových podmínkách -40°C až $+85^{\circ}\text{C}$ je přesnost dosažená pomocí jednoduché lineární aproximace rovna $\pm 0.3^{\circ}\text{C}$.

Závislost teploty na odporu senzoru (r) je tedy obecně dána lineární funkcí

$$T_{LIN}(r) = A \times r + B \quad (20)$$

Konstanty A a B se určí matematicky. Obrázek 17 zobrazuje celkovou aproximační chybu při vymezené velikosti intervalu z celkového měřitelného rozsahu do 500°C [8]. Velikost chyby nezávisí nejen na daném rozsahu měření, ale také na poloze tohoto rozsahu v celkovém teplotním pásmu použitelném pro měření teploty odporovými teplotními senzory.



Obr. 17. Chyba lineární aproximace

Po částech lineární aproximace

Tato linearizační metoda je vlastně lineární aproximace aplikovaná na zvoleném počtu intervalů [8]. Tím je generována řada lineárních segmentů tak, že koncový bod každého segmentu navazuje na počáteční bod segmentu následujícího. Tím vznikne řada bodů pospojovaných přímkami. Tyto body (koeficienty) mohou být po výpočtu uloženy v paměti ROM nebo Flash. Pomocí takto vzniklé tabulky koeficientů pak mikrokontroler provede jednoduchou lineární interpolaci k určení teploty podle odporu senzoru. Každý koeficient obsažený v tabulce je tedy bod přechodové charakteristiky senzoru, daný dvojicí hodnot – odporu a teploty. Tabulka má tedy následující tvar:

$$\{r_0, t_0; r_1, t_1; r_2, t_2; \dots; r_N, t_N\}$$

Po zadání této množiny má pak mikrokontroler za úkol nejdříve vybrat vhodnou dvojici koeficientů ($\{r_m, t_m\}$ a $\{r_n, t_n\}$) a poté provést lineární interpolaci mezi těmito dvěma body k určení teploty. Vlastní lineární interpolační rovnice platná v intervalu $\langle r_m, r_n \rangle$ (pro vybraný segment) má tedy tvar

$$T_{SEG}(r) = t_m + (r - r_m) \times \frac{t_n - t_m}{r_n - r_m} \quad (21)$$

Pro N lineárních segmentů ($N+1$ koeficientů) tak musí být v paměti uloženo celkem $2N+2$ hodnot. Každý z těchto segmentů má však stejnou šíři, což umožňuje zmenšení tabulky na N hodnot, pokud v ní uložíme pouze body odpovídající teplotám $\{t_0; t_1; t_2; \dots; t_N\}$

a bod odpovídající odporu vypočteme podle vztahu

$$r_n = r_0 + n \times r_{SEG} \quad (22)$$

kde r_0 odpor pro nulý koeficient $\{r_0, t_0\}$

r_{SEG} velikost intervalu segmentu

a tyto pevně dané hodnoty jsou uloženy v ROM paměti spolu s tabulkou koeficientů. Lineární interpolační rovnice pro daný segment pak je

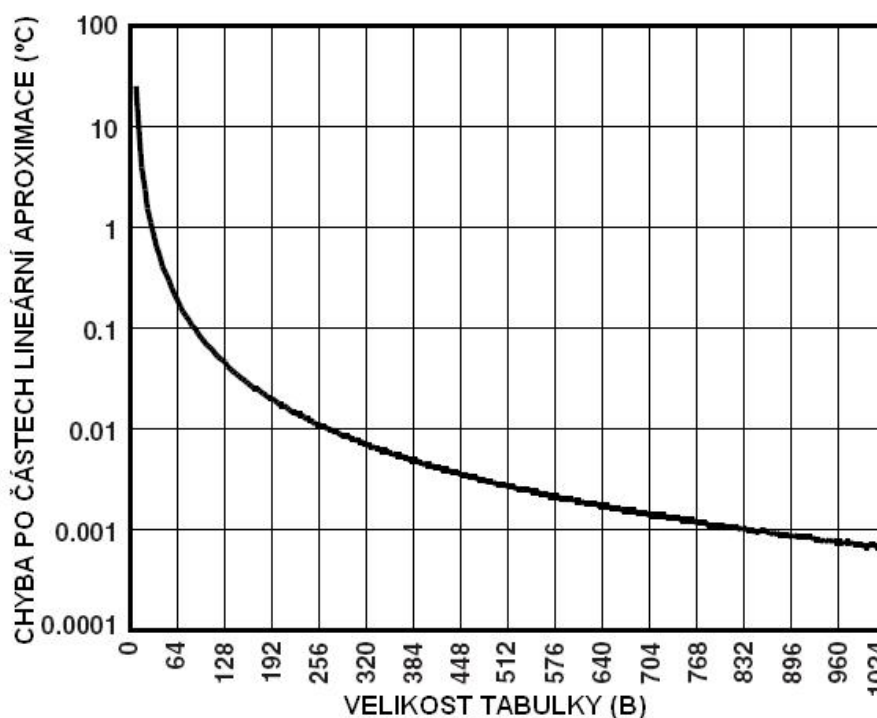
$$T_{SEG}(r) = t_i + [r - (r_0 + i \times r_{SEG})] \times \frac{t_{i+1} - t_i}{r_{SEG}} \quad (23)$$

kde i určuje vybraný segment a vypočte se pomocí hodnoty r s použitím funkce pro zaokrouhlení podle vztahu:

$$i = \text{trunc}\left(\frac{r - r_0}{r_{SEG}}\right) \quad (24)$$

Výše odvozený vztah pro výpočet teploty ($T_{SEG}(r)$) je tedy opět lineární aproximací mezi dvěma koeficienty t_i a t_{i+1} . V praxi tedy musí mikrokontroler nejprve určit i , zjistit tak které dva koeficienty t_i a t_{i+1} jsou dva nejbližší pro vstupní hodnotu odporu r , a poté pomocí i jednoduše vypočítat teplotu senzoru odvozenou z jeho odporu.

Celková chyba generovaná touto Po částech lineární aproximací závisí na počtu segmentů (počtu koeficientů a tedy na velikosti vyhledávací tabulky) a na celkovém rozsahu měření teploty. Obrázek 18 ukazuje závislost chyby na velikosti vyhledávací tabulky pro měřicí teplotní rozsah -200 až +850 °C [8].



Obr. 18. Chyba po částech lineární aproximace (rozsah -200°C až +850°C)

2.4.3 Kalibrace

Mikrokonvertor ADuC845 má vestavěné funkce pro kalibraci chyb offsetu a zesílení některých vnitřních prvků obvodu. Pokud je však brán v úvahu celý měřicí řetězec včetně

odporového senzoru, lze dosáhnout mnohem menší celkové chyby, a použití těchto interních kalibračních funkcí není nutné [8].

Veškeré senzory jsou zatížené svými chybami, které bývají většinou specifikovány jejich výrobcí. Naštěstí je většina těchto nejistot snadno softwarově odstranitelná. Kalibrační funkce popsané dále jsou buď jednobodové nebo dvoubodové a dají se použít ve spojitosti se všemi třemi dříve uvedenými linearizačními metodami.

Jednobodová kalibrace

Jak již bylo popsáno dříve, tak definice přechodové charakteristiky senzoru výrazně závisí na hodnotě R_0 což je odpor při teplotě 0°C , který u velké části odporových senzorů teploty odpovídá nominální hodnotě 100Ω . Právě tato konstanta R_0 však bývá největším zdrojem nejistot těchto senzorů, protože její hodnota se výrazně liší kus od kusu podle vybraného modelu senzoru [8]. Tato hodnota je násobena ve výrazu pro výpočet odporu $R_{RTD}(t)$, chyba je tedy multiplikativní. Může být jednoduše odstraněna použitím násobku *scale* v tomto výrazu popsaném dříve, jakožto vztahu popisujícím výsledek AD převodu:

$$R_{RTD} = ADC_{NORM} \times scale \quad (25)$$

Pokud je senzor uveden do stavu, ve kterém můžeme přesně stanovit jeho teplotu a je provedena AD konverze, pak hodnota *scale* může být vypočtena podle vztahu:

$$scale = \frac{R_{CAL}}{ADC_{CAL}} \quad (26)$$

kde ADC_{CAL} je aktuální normalizovaný výsledek konverze

R_{CAL} je ideální (předpokládaná) hodnota odporu

R_{CAL} může být vypočítána pomocí vztahu $R_{RTD}(t)$ uvedeného v kapitole 2.3.3.

Tímto způsobem lze nastavit nominální hodnotu R_0 , odporového teplotního senzoru, a zároveň také referenčního odporu.

Dvoubodová kalibrace

Tato metoda navíc kompenzuje chyby offsetu, které se mohou vyskytovat [8]. Pro názornost předpokládejme že byla provedena AD konverze jednoho vzorku signálu senzoru o přesně známé teplotě, následně kalibrována jednobodovou metodou. Poté je

senzor zahřát na jinou velmi přesnou známou teplotu a provede se opět AD převod. Vztah vyjadřující hodnotu $scale$ pro tyto dva vzorky pak můžeme zapsat:

$$scale = \frac{R_{CAL2} - R_{CAL1}}{ADC_{CAL2} - ADC_{CAL1}} \quad (27)$$

kde R_{CAL1} a ADC_{CAL1} je odpor a výsledek AD převodu pro první vzorek (1.kalibrační bod) a R_{CAL2} a ADC_{CAL2} je odpor a výsledek AD převodu pro druhý vzorek (2.kalibrační bod). Toto je způsob odvození strmosti funkční závislosti ADC_{NORM} na R_{RTD} použitím dvou bodů této funkce. Nyní známe hodnotu $scale$ a hodnotu offsetu můžeme odvodit z výše uvedeného R_{RTD} vztahu po nahrazení R_{RTD} a ADC_{NORM} výrazy R_{CAL1} a ADC_{CAL1} :

$$offset = R_{CAL1} - ADC_{CAL1} \times scale \quad (28)$$

Pokud jsou R_{CAL1} a ADC_{CAL1} nulové hodnoty (což znamená absenci předchozího bodu) pak výraz pro hodnotu $scale$ je stejný jako pro jednobodovou aproximaci.

Kalibrační metody mohou být velkým přínosem, často však bývá jejich aplikace v provozních podmínkách nemožná. Pokud kalibrace nemůže být provedena výše uvedenými metodami, lze využít alespoň vnitřní funkce kalibrace AD převodníku, které jsou popsány v manuálu mikrokonvertoru ADuC8045. K tomu stačí odpojit odporový senzor, propojit vstupy (0Ω) a spustit systémovou funkci nulové kalibrace. Poté se zapojí namísto senzoru přesný odpor o hodnotě 719.36Ω a spustí se funkce kalibrace pro maximální hodnotu. Tímto způsobem se dají kompenzovat vnitřní nejistoty AD převodníku, na nejistoty vlastního odporového senzoru se však nevztahuje.

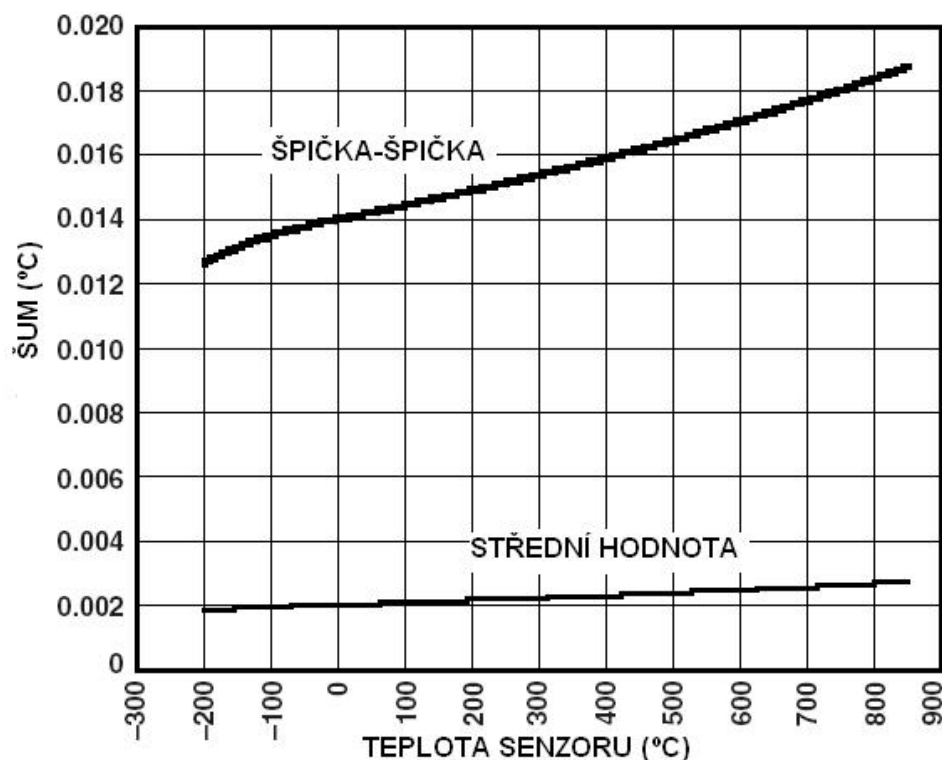
K přednostem mikrokontroleru ADuC845 patří datová Flash paměť která může sloužit právě pro trvalé uložení kalibrovaných hodnot ($scale$ a $offset$). Hodnoty tak mohou být obnoveny automaticky po každém zapnutí.

2.4.4 Analýza nejistot

V aplikacích které se zabývají zpracováním signálu existuje řada zdrojů nejistot (vstupní šum zesilovače, linearita AD převodu, Johnsonův šum, teplotní drift zesilovače, teplotní drift odporu, atd.) [8]. Určení těch nejzávažnějších zdrojů chyb bývá důležitým úkolem při řešení řady aplikací. Mikrokonvertory řady ADuC8XX integrují všechny aktivní fáze zpracování signálu do jediného monolitického celku, plně specifikovaného výrobcem, což činí analýzu nejistot mnohem jednodušší. Nejdůležitější je tedy správné porozumění nejistot, které se vztahují na vlastní základní prvek v měřícím obvodu, tedy nelinearit senzoru.

Pokud systém není kalibrován pro specifický senzor, (například pomocí jednobodové nebo dvoubodové metody), pak právě tento senzor má zcela určitě největší vliv na celkovou absolutní nejistotu. Nejistota vlastního senzoru by měla být podrobně specifikována právě jeho výrobcem, a liší se v závislosti na vybraném specifickém senzoru.

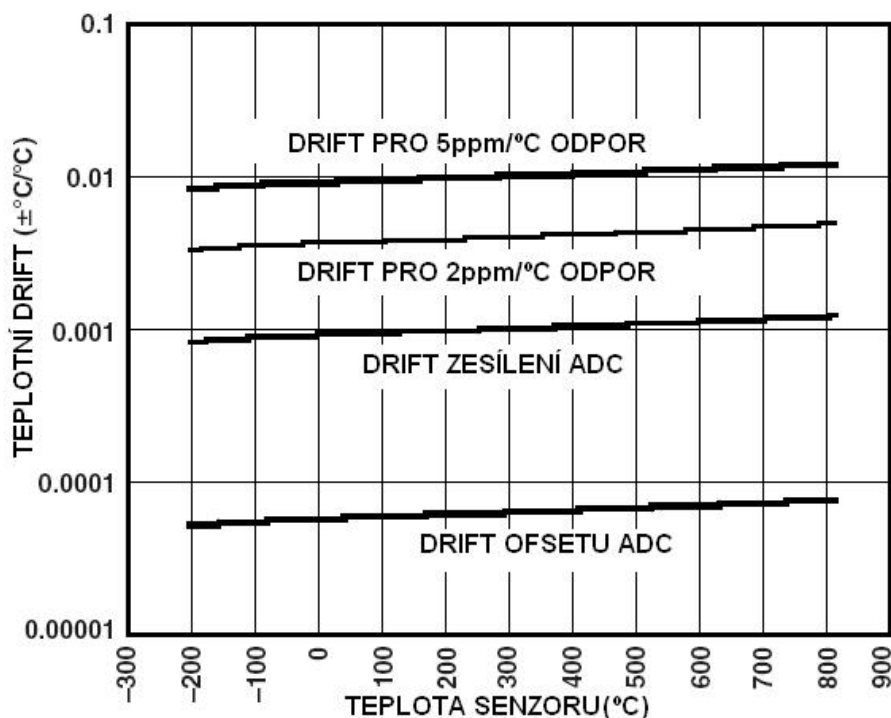
Dalším typem nejistoty je šum. Při návrhu aplikací je třeba brát v úvahu tři hlavní zdroje šumu: Johnsonův šum, napětový šum na vstupu zesilovače a AD převodníku a proudový šum na vstupu zesilovače a AD převodníku. Celkově se tyto chyby vyhodnocují mimojiné jako suma jejich čtverců, a tedy menší zdroje nejistot jsou zanedbatelné pokud je jeden z nich výrazný. V tomto případě je hlavním zdrojem šumu napětový šum na vstupu zesilovače a AD převodníku. Při nastaveném zesílení 7.8125 je střední hodnota tohoto šumu $0.37 \mu\text{V}$, špičkově pak $2.44 \mu\text{V}$. Vliv tohoto šumu na výslednou teplotu ukazuje obrázek 19.



Obr. 19. Vliv šumu AD převodníku na výslednou teplotu

Jak vidíme v tomto grafu tak špičková hodnota šumu při maximální měřené teplotě nepřesahuje 0.019°C a je mnohem menší při nižších teplotách.

Dalším zdrojem nejistoty který stojí za zmínku je teplotní drift, lépe řečeno offset AD převodníku, teplotní drift zesilovače a teplotní drift referenčního odporu [8]. Tyto nejistoty jsou způsobeny změnou teploty obvodů pro zpracování signálu, a také změnou teploty referenčního odporu. Vztahují se tedy spíše na teplotu okolního prostředí vlastního čipu, než na odporový senzor. Hodnota teplotního driftu (tedy citlivost na změny teploty okolí) se navíc mění stejně jako funkce teplotního odporového senzoru kvůli nelineární charakteristice $T_{RTD}(r)$. Konečný výsledek je znázorněn na obrázku číslo 20. Osa x je teplota odporového senzoru. Osa y je teplotní drift vyjádřený ve °C jakožto změna naměřené hodnoty na změnu teploty prostředí ve °C. Například, pokud je teplota odporového senzoru fixována na 100°C, pak velikost tohoto driftu (při použití odporu s 5 ppm/°C) je $\pm 0.01^\circ\text{C}/^\circ\text{C}$. Takže pokud se teplota okolního prostředí změní o řekněme 50°C, pak odečtená hodnota naměřené teploty se může změnit až o $\pm 0.5^\circ\text{C}$ (při zanedbání ostatních faktorů ovlivňujících teplotní drift).



Obr. 20. Teplotní drift vs. teplota senzoru

Není těžké si představit, že v průmyslových podmínkách, kdy teplota okolního prostředí často kolísá v rozsahu od -40°C do $+85^{\circ}\text{C}$ a víc, může být tento teplotní drift poměrně výrazným zdrojem nejistoty. V těchto případech může posloužit teplotní senzor integrovaný přímo v čipu ADuC845, pro vyhodnocení jeho teploty, která kopíruje teplotu okolí, a pak už jen zbývá programově vyhodnotit tuto teplotu a v závislosti na ní kompenzovat výše popsané nejistoty.

Další nejistota je způsobena budícím proudem, který při průchodu odporovým senzorem tento senzor zahřívá. Závisí tedy na velikosti proudu, který by měl být co nejmenší. V mém případě je odporový senzor buzen proudem o velikosti $400\mu\text{A}$, při kterém hodnota tohoto disipovaného tepla nepřesáhne $16\mu\text{W}$ pro teplotní senzor PT100 [8]. Ohřev způsobený tímto teplem se liší podle modelu senzoru, obvykle však bývá minimální a tedy zanedbatelný. Pokud tomu tak není, lze budící proud programově snížit na $200\mu\text{A}$, čímž se disipované teplo na senzoru zmenší čtyřikrát. Další zdroje nejistot lze považovat za zanedbatelné. Například hodnota Johnsonova bílého šumu bývá výrazně menší než napěťový šum na vstupu AD převodníku. Za zmínku ještě stojí integrální nelinearita AD převodníku, tato hodnota však ve většině případů bývá pod hranicí šumu [8].

3 EXPERIMENTÁLNÍ SESTAVY S OBVODEM ADUC

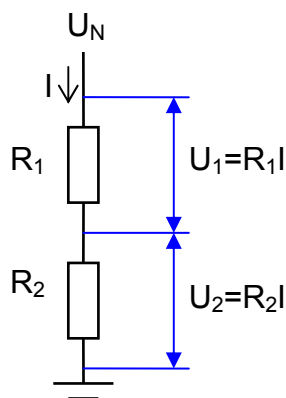
Hlavním cílem mé diplomové práce je návrh a ověření funkce experimentální sestavy s obvodem ADuC845. Tento obvod nabízí nepřeberné množství funkcí, a proto bych rád realizoval sestavu toho nejběžnějšího případu zpracování senzorického signálu, kterým je měření teploty pomocí odporového teplotního čidla Pt100. V experimentální části je nejprve řešena úloha převodu přivedeného napětí se zaměřením na vlastní AD převod a zobrazení výsledků za účelem pochopení funkce samotného AD převodníku. Druhá sestava se pak zaměřuje na komplexní vyhodnocení teploty připojeného čidla Pt100.

3.1 Ověření funkce AD převodníku

První experimentální sestavou bych se rád seznámil se základními funkcemi mikrokonvertoru a pochopil principy obsluhy AD převodníku. Základním úkolem v této sestavě je přivést na analogové vstupy napětí, které bude následně vyhodnoceno a zobrazeno na obrazovce PC pomocí Hyperterminálu systému Windows určeného ke komunikaci s periferiemi pomocí sériové linky COM.

3.1.1 Návrh a realizace obvodu

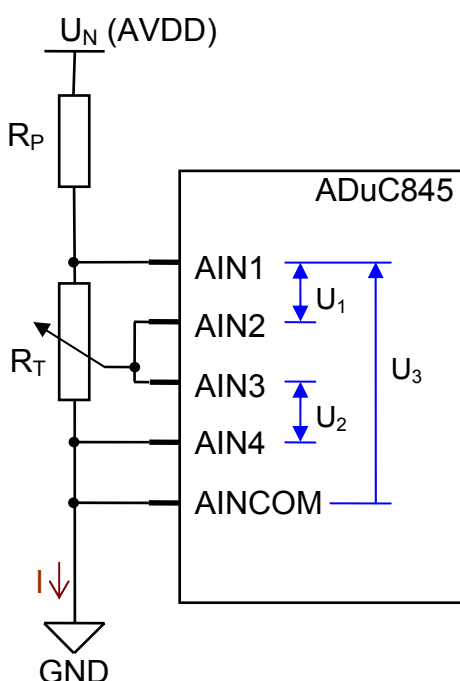
Podle obrázku 7 je obvod ADuC schopen zpracovat až 10 vstupních analogových kanálů zapojených unipolárně, nebo až 5 kanálů zapojených bipolárně (diferenčně). Pro ověření funkce chci využít jednoduchého obvodu odporového děliče (obr. 21) a přivést na vstupy mikrokonvertoru několik různých úrovní napětí.



Obr. 21. Odporový dělič

K tomu poslouží vývod AVDD (U_N) jakožto zdroj napětí pro analogovou část vývojového kitu. Hodnotu tohoto napětí jsem určil pomocí multimetru který naměřil 4.88V. Při použití interního referenčního napětí o velikosti 1.25V je pak rozsah napětí které lze přivést na primární AD převodník 0-1.28V ($V_{REF} \times 1.024$). Z toho důvodu je nejprve nutné snížit velikost napětí na přípustnou hodnotu. K tomu poslouží předřadný odpor R_P o velikost 4,42 k Ω . Pro dělení napětí pak trimr (R_T) reprezentující dva proměnlivé odpory o velikosti 0 až 1.168 k Ω .

Schéma zapojení odporového děliče k obvodu ADuC845 ukazuje obrázek 22.



Obr. 22. Připojení děliče k mikrokonvertoru

Proud procházející obvodem je podle Ohmova zákona:

$$I = \frac{U_N}{R_P + R_T} = \frac{4.88V}{4.42k\Omega + 1.17k\Omega} = 0.87mA \quad (29)$$

a na předřadném odporu tedy dochází k úbytku napětí o hodnotu

$$U_P = R_P \times I = 4.42k\Omega \times 0.87mA = 3.86V \quad (30)$$

Zbýlých 1.02 V se pak dělí podle polohy trimru na napětí U_1 a U_2 , a tyto hodnoty jsou přivedeny na vstupy AIN1-AIN2 a AIN3-AIN4 jako dva bipolární kanály, které vyhodnotí primární převodník. Pro ověření lze pak přenastavením vstupních kanálů nastavit unipolární zapojení AIN1-AINCOM pro napětí U_3 ($U_3 = U_1 + U_2$, nebo také $U_3 = U_N - U_P$).

K realizaci tohoto zapojení jsem vytvořil jednoduchý obvod pomocí univerzálního plošného spoje, který je pomocí konektoru připojen na svorkovnici J2 vývojového kitu (viz. příloha 2), která obsahuje analogové vstupy/výstupy mikrokonvertoru. Pro přehlednost jej pojmenuji třeba ADDAJ2 (viz. příloha 3). Celkové zapojení dále obsahuje některé pasivní prvky pro R/L filtraci signálů a ochranu vstupů proti přepětí, které jsou osazeny přímo na desce vývojového kitu.

3.1.2 Programová realizace

Programová realizace spočívá především ve správné obsluze SFR funkčních registrů, a to především nastavení vstupního kanálu, rozsahu měřeného signálu a správnému čtení SFR registrů ADC0L, ADC0M, ADC0H (obr. 5), ve kterých je uložen výsledek AD převodu primárního ADC. Normalizovaná hodnota (v rozsahu 0 až 1) se získá spojením těchto tří bytů a podělením hodnotou rozlišení převodníku (2^{24}):

$$ADC_{NORM} = (ADC0H * 0x10000 + ADC0M * 0x100 + ADC0L) / 16777216.0$$

Tento normalizovaný výstup AD převodníku je poměr vstupního napětí V_{IN} a hodnoty rozsahu měřeného napětí *span*:

$$ADC_{NORM} = \frac{V_{IN}}{span} \quad (31)$$

Při použití interního referenčního napětí $V_{REF} = 1.25V$ je maximální možný rozsah měřeného napětí 0 až 1.28V (*span*). Tento rozsah vyhovuje i mým podmínkám, kdy hodnota vstupního signálu nepřesáhne 1.03 V, a odpovídá mu nastavení SFR registru ADC0CON na hodnotu 0x27.

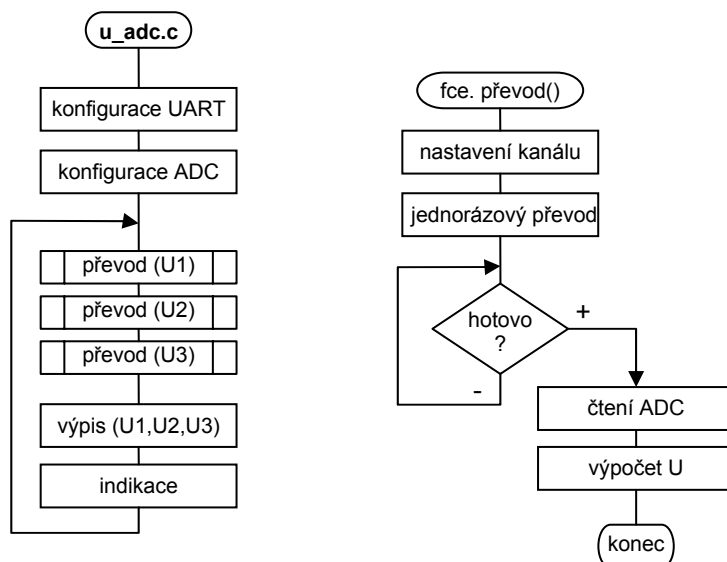
Výsledná hodnota napětí se pak bude rovnat:

$$V_{IN} = ADC_{NORM} \times span = ADC_{NORM} \times 1.28 \quad (32)$$

Dále je třeba konfigurovat komunikační obvod UART tak, aby probíhala komunikace s Terminálem systému Windows a byl možný výstup naměřených hodnot. Například pomocí časovače 3 na rychlost 9600 baudů v módu 1 (ob. 11). K tomu slouží SFR registry T3CON, T3F, a SCON.

Blokové schéma programu

Základní blokové schéma programu a použité funkce ukazuje obr. 23:

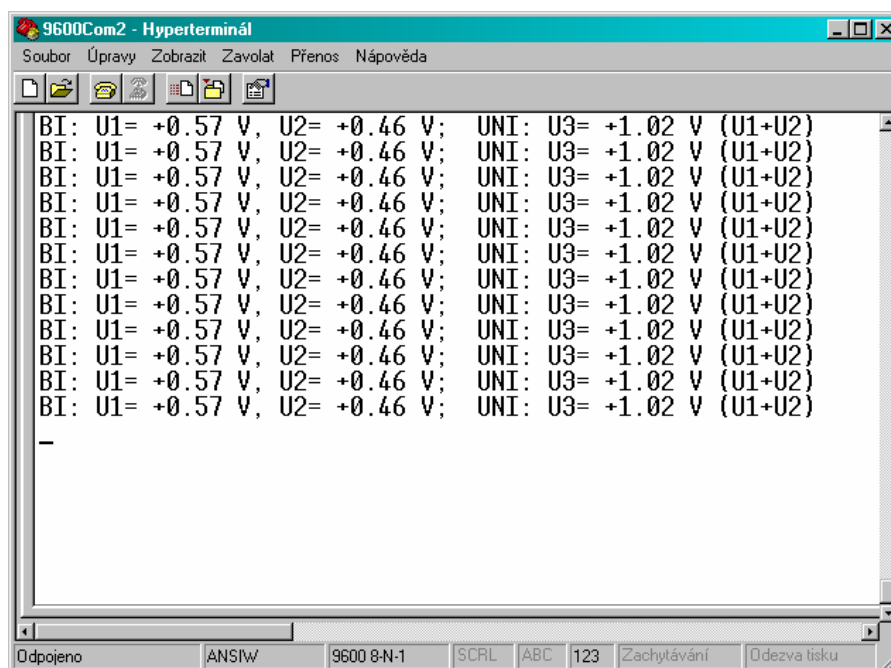


Obr. 23. Blokové schéma těla programu a hlavní funkce

Zdrojový kód programu v C jazyku je v příloze 5. Jak vyplývá z obrázku, tak program nejdříve provede konfiguraci komunikačního obvodu UART a základní konfiguraci primárního AD převodníku (rozsah, kódování, ...). Poté vstoupí do smyčky ve které postupně volá funkci převodu na třech vstupních kanálech. Argumentem funkce je hodnota pro SFR registru ADC0CON2 pro nastavení vstupních kanálů. Funkce obsahuje čekací smyčku pro testování nastavení bitu RDY0 který indikuje dokončení převodu, po kterém následuje čtení výsledku AD převodu a výpočet napětí podle výše uvedených vzorců. Smyčka těla programu pak dále obsahuje bloky pro výpis naměřených hodnot na Terminálu PC a indikaci pomocí LED na desce vývojového kitu.

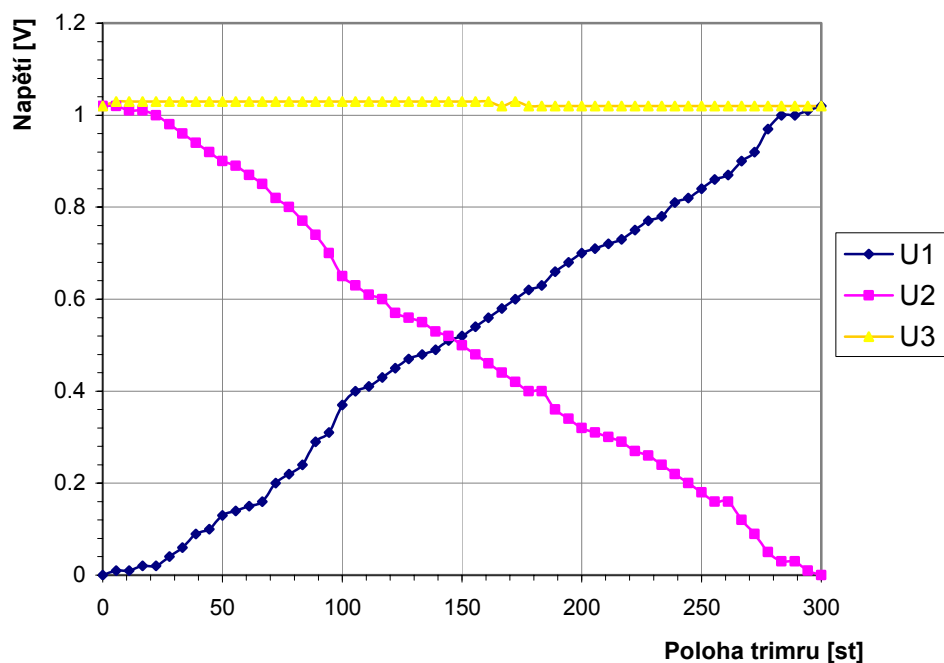
3.1.3 Ověření funkčnosti

Výstup dat na obrazovce PC pomocí Terminálu systému Windows ukazuje obr. 24.



Obr. 24. Ukázka výstupu dat na PC

Pro ověření funkčnosti této aplikace provedu měření přivedeného napětí U_1 , U_2 , U_3 při současném otáčení trimrem. Výstupní data jsou ukládána do textového souboru. Po jejich zpracování dostávám následující graf:



Obr. 25. Graf závislosti napětí U_1, U_2, U_3 na poloze trimru

S rostoucím napětím U_1 klesá U_2 a naopak. Horní mez těchto hodnot naznačuje konstantní napětí U_3 které odpovídá hodnotě 1.02V. Graf potvrzuje správnost předchozích výpočtů, a znázorňuje jednotlivá napětí přivedená na analogové vstupy mikrokonvertoru ADuC845.

3.1.4 Hodnocení

Obdobným způsobem by bylo možné obsadit všech deset analogových vstupů a obsluhovat je jediným AD převodníkem. S narůstajícím počtem kanálů však roste doba potřebná k převodu a zpracování. V uvedené aplikaci byla rychlost výstupu dat necelé 2 Hz při nastavení registru SF na hodnotu 0x45, která odpovídá při kontinuálním převodu jednoho kanálu rychlosti vzorkování 19.79 Hz. Toto téměř desetinásobné zpomalení je zřejmě způsobeno přenastavováním kanálů (multiplexeru) a čekacími smyčkami, které jsou v programu použity. Zrychlení procesu by bylo možné dalším snížením hodnoty SF registru nebo také vypnutím režimu „CHOP ON“, což by však mělo negativní dopad na nejistoty způsobené AD převodem. Lepším řešením by bylo využití sekundárního AD převodníku.

3.2 Připojení senzoru Pt100

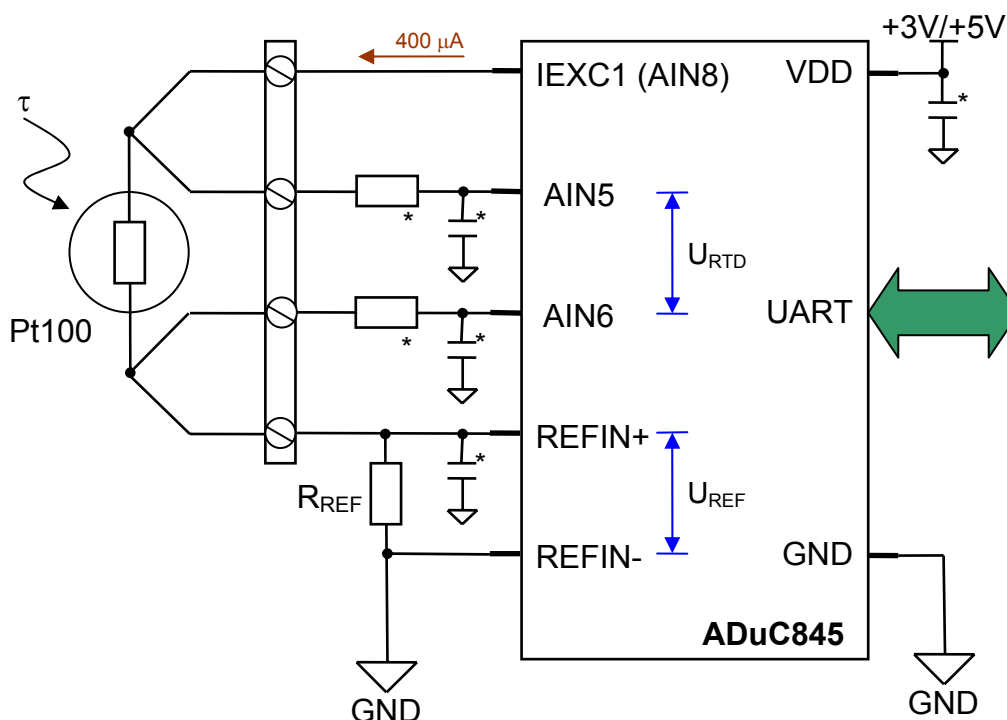
V druhé sestavě se pokusím připojit k obvodu ADuC845 odporový teplotní senzor (RTD), a vyhodnotit jeho teplotu. Konkrétně jsem vybral jedno z nejčastěji používaných teplotních čidel, a to čidlo Pt100. Úkolem je tedy správné zapojení celé sestavy, vyhodnocení senzorického signálu včetně linearizace a kalibrace, a výstup naměřené hodnoty teploty na obrazovce PC pomocí Hyperterminálu systému Windows. Při řešení této úlohy budu vycházet především z materiálů dostupných na webových stránkách společnosti Analog Devices, včetně linearizačních rutin popsanych v odstavci 2.4.2.

3.2.1 Návrh a realizace zapojení

Při řešení zapojení využiji, stejně jako v první sestavě, konektor J2 desky vývojového kitu, který obsahuje veškeré analogové vstupy a výstupy mikrokonvertoru spolu s napájením a zemnicím vývodem (příloha 2). K dispozici mám snímač teploty v pouzdře se závitem s kabelovým vývodem vyráběný firmou REGMET. Jedná se o typ RA3p/D-2 (viz. příloha

4), tedy v dvou vodičovém provedení se stíněním, třídou přesnosti B a rozsahem teplot -30 až $+180^{\circ}\text{C}$.

Při návrhu obvodu zapojení tohoto senzoru k mikrokonvertoru ADuC845 vycházím především z obrázku č. 15. Veškeré prvky obvodu chci osadit na již použitou desku univerzálního plošného spoje ADDAJ2. Z toho důvodu nelze použít analogové vstupy AIN1 až AIN4, které jsou obsazené. Navržené schéma zapojení senzoru ukazuje obr. 26.



Obr. 26. Připojení čidla Pt100 k mikrokonvertoru

(* označuje prvky ochrany vstupů a prvky pro R/L filtraci signálů)

Toto je stejné zapojení jako na obrázku 15, veškeré aktivní komponenty - zdroj budícího proudu, diferenční vstupy pro V_{RTD} a V_{REF} , zesílení A_{ADC} , AD převodník a mikrokontroler) jsou obsaženy v jediném integrovaném obvodu, spolu s ostatními prostředky jako například seriovými komunikačními porty. Zapojení dále obsahuje svorkovnici a některé pasivní prvky pro R/L filtraci signálů a ochranu vstupů proti přepětí. Toto je kompletní zapojení senzoru, vyžadující už jen zdroj napětí a některý z periferních obvodů pro číslicové rozhraní (RS-232 nebo RS-485). Všechny tyto prvky jsou implementovány na

desce vývojového kitu. Svorkovnice je určena pro 4-vodičové připojení senzoru. Při použití 2-vodičového je nutné propojení krajních svorek.

3.2.2 Výběr referenčního odporu

Správná volba referenčního odporu hraje klíčovou roli při realizaci zapojení, neboť hodnota odporu spolu s nastavením zesílení převodníku a odporem senzoru je přímo obsažena ve vztahu pro výpočet normalizovaného výstupu AD převodníku (16).

Zesílení A_{ADC} vypočítám ze vztahu :

$$A_{ADC} = \frac{V_R}{span} \quad (33)$$

kde V_R je nastavený rozsah vstupu referenčního kanálu (2.5V nebo 1.25V)

$span$ rozsah vstupu AD převodníku

Hodnotu $span$ lze nastavit pomocí ADC0CON SFR registru na 20, 40, 80, 160, 320, 640, 1.28 a 2.56 mV.

Po dosazení zesílení do vztahu (16) pak dostávám výraz pro výpočet normalizované hodnoty výstupu AD převodníku:

$$ADC_{NORM} = A_{ADC} \times \frac{R_{RTD}}{R_{REF}} = \frac{U_{REF}}{span} \times \frac{R_{RTD}}{R_{REF}} \quad (34)$$

Pro maximální teplotu kterou mohu s daným snímačem měřit ($T_{max}=180^{\circ}C$) pak podle vztahu (16) určím odpor senzoru R_{RTD} . Hodnoty $span$ a R_{REF} volím a po dosazení dostávám hodnoty normalizovaného výstupu ADC_{NORM} . Pro lepší orientaci uvádím výsledné hodnoty v tabulce V. Jako odpory jsem zde uvedl běžně dostupné odpory řádu k Ω unifikované řady E12.

Tabulka V. Výběr odporu a koeficientu span

ADC_{NORM}		span - nastavené zesílení převodníku (pro V_{ref}=2.5) [mV]								
		20	40	60	80	160	320	640	1.28	2.56
Unifikovaná řada odporů E12 [kΩ]	1	21.06	10.53	7.02	5.26	2.63	1.32	0.66	329.1	164.5
	1.2	17.55	8.77	5.85	4.39	2.19	1.10	0.55	274.2	137.1
	1.5	14.04	7.02	4.68	3.51	1.75	0.88	0.44	219.4	109.7
	1.8	11.70	5.85	3.90	2.92	1.46	0.73	0.37	182.8	91.41
	2.2	9.57	4.79	3.19	2.39	1.20	0.60	0.30	149.6	74.79
	2.7	7.80	3.90	2.60	1.95	0.97	0.49	0.24	121.9	60.94
	3.3	6.38	3.19	2.13	1.60	0.80	0.40	0.20	99.71	49.86
	3.9	5.40	2.70	1.80	1.35	0.67	0.34	0.17	84.37	42.19
	4.7	4.48	2.24	1.49	1.12	0.56	0.28	0.14	70.01	35.01
	5.6	3.76	1.88	1.25	0.94	0.47	0.24	0.12	58.76	29.38
	6.8	3.10	1.55	1.03	0.77	0.39	0.19	0.10	48.39	24.20
	8.2	2.57	1.28	0.86	0.64	0.32	0.16	0.08	40.13	20.06

Normovaná hodnota výstupu AD převodníku ADC_{NORM} musí být v rozsahu 0..1. Pro plné pokrytí specifikovaného rozsahu měření (-30 až +180°C) volím při nastaveném rozsahu $span = 80mV$ odpor 5.6 kΩ. Hodnota odporu naměřená multimetrem je 5.58 kΩ.

3.2.3 Výpočet hodnoty odporu z výstupu AD převodníku

V následující části jsou popsány především vztahy pro odvození teploty na základě hodnoty získané na výstupu AD převodníku.

Další úpravou předchozího vztahu (34) dostanu vzorec pro výpočet odporu senzoru

$$R_{RTD} = ADC_{NORM} \times \frac{R_{REF}}{A_{ADC}} \quad (35)$$

pro zjednodušení nahradím zlomek koeficientem *scale*

$$scale = \frac{R_{REF}}{A_{ADC}} \quad (36)$$

Zesílení převodníku A_{ADC} je rovno

$$A_{ADC} = \frac{V_R}{span} = \frac{2.5V}{80mV} = 31.25 \quad (37)$$

Koeficient *scale* pro dané hodnoty zesílení a odporu je pak podle (36)

$$scale = \frac{5580}{31.25} = 178.56$$

Koeficient *scale* je tedy daný faktor pro určení odporu, použitý v kódu programu. Pro větší přesnost je do rovnice ještě přidán koeficient *offset*. Tento koeficient reprezentuje daný offset senzoru a může být použit pro případnou kompenzaci chyb, jak je popsáno v sekci Kalibrace (2.4.3).

$$R_{RTD} = ADC_{NORM} \times scale + offset \quad (38)$$

Toto je tedy rovnice vyjadřující funkci odporu senzoru v závislosti na výstupu AD převodníku, obsahující pouze dvě konstanty – *scale* a *offset*.

3.2.4 Funkce DA převodníku

Abych využil další periferie obvodu, pokusím se nastavit hodnotu DA převodníku v závislosti podle naměřené teploty. K tomu jsem vytvořil funkci DACoutput, která má tři vstupní parametry (*T*- naměřená teplota, *T_{MAX}* - maximální teplota, a *T_{MIN}*- minimální teplota). Výstupem funkce je pak 12-bitová hodnota DAC zapsaná v registrech DA převodníku podle vztahu:

$$DAC = (T - T_{MIN}) \times \frac{2^{12}}{T_{MAX} - T_{MIN}} \quad (39)$$

Při rozsahu -30 až +180°C tak naměřené hodnotě -30°C odpovídá 0 V a hodnotě +180°C odpovídá napájecí napětí (5V). Tímto jednoduchým způsobem tak celou sestavu rozšířím o funkci převodníku teploty na napětí.

3.2.5 Programová realizace

Programová realizace sestává z vytvoření funkčního programu aplikace v jazyku C pomocí vývojového prostředí Aspire dodávaného s vývojovou sadou QuickStart+. Jde především o správné čtení/zápis dat SFR registrů. Výsledky získané AD převodem jsou zpracovány a výsledná data budou odesílána na obrazovku PC. Abych využil další periferie mikrokonvertoru, pokusím se nastavit hodnotu DA převodníku v závislosti na naměřené teplotě a rozsahu měření.

Nastavení AD převodníku

Pro zpracování signálu použijí primární převodník. Zápisem hodnoty 22h do SFR registru ADC0CON1 se nastaví rozsah vstupu na požadovaných 80mV. Zápisem hodnoty 4Ch do SFR registru ADC0CON2 se pak aktivuje vstup referenčního napětí a na primární ADC je přiveden vstup AIN5-AIN6, na který je připojen senzor. Čtení normalizovaného výsledku převodu ADC_{NORM} pak probíhá obdobně jako v předchozí aplikaci.

Výpočet teploty podle odporu (Linearizace)

K výpočtu teploty slouží funkce obsažené v souborech RTDmath.c, RTDlin.c a RTDpwl.c, které jsou volně dostupné na webových stránkách Analog Devices. Jedná se o linearizační funkce popsané v kapitole 2.4.2. (Přímá matematická metoda, Jednoduchá linearizace a Po částech lineární linearizace). Součástí každého zdrojového C-kódu jsou především funkce pro výpočet teploty podle zadaného odporu a výpočet odporu podle zadané teploty.

Interní senzor teploty mikrokontroleru

Jednoduchým nastavením sekundárního převodníku lze získat teplotu naměřenou senzorem integrovaným v mikrokontroleru. K tomu slouží registr ADC1CON, který při nastavení na hodnotu 0Eh přivede signál ze senzoru přímo na vstup sekundárního převodníku. Další zpracování signálu je obdobné jako při zpracování signálu externího senzoru Pt100.

Komunikace

Program komunikuje obdobně jako aplikace první experimentální sestavy tak, že naměřená data jsou vysílána na COM port PC. Zobrazení je pak umožněné pomocí Hyperterminálu systému Windows. K tomu je třeba konfigurovat obvod komunikační UART. Například pomocí časovače 3 na rychlost 9600 baudů v módu 1 (obr. 11). K tomu slouží SFR registry T3CON, T3F, a SCON.

Pomocí terminálu lze také vysílat znaky opačným směrem a ovládat tak program. K tomu slouží přerušení vyvolané obvodem UART, které ukládá přijaté byty do SFR registru SBUF. Aplikace tak navíc umožňuje jednoduchou obsluhu pomocí menu.

Kalibrace

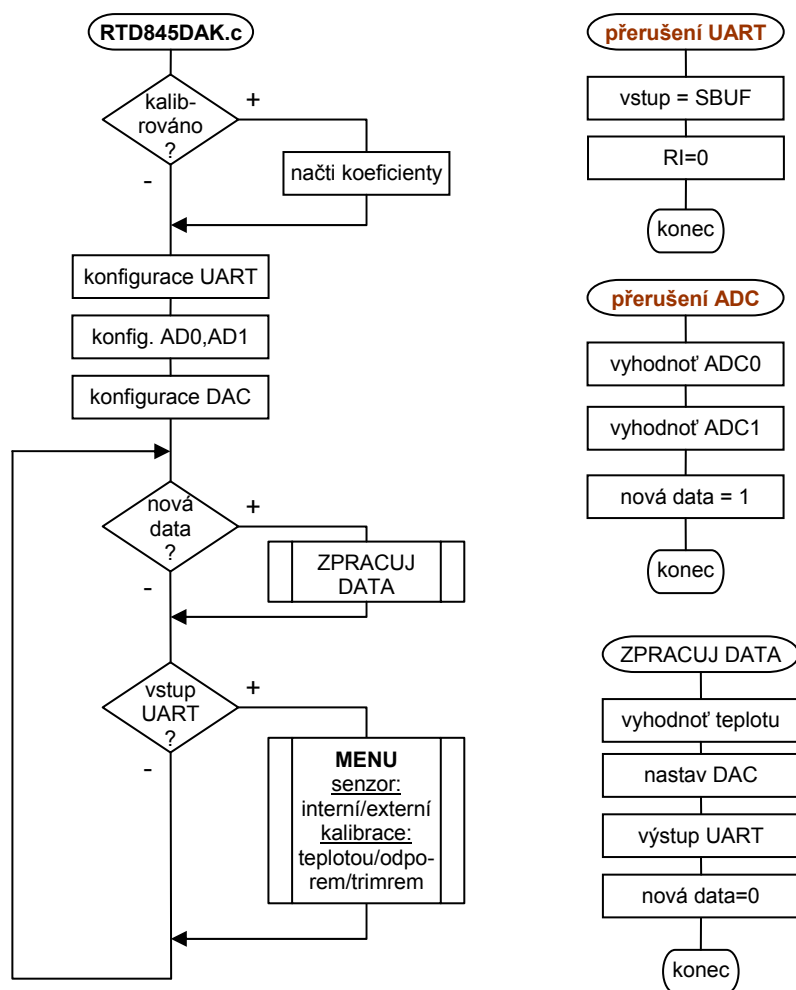
Pomocí tohoto menu je možné provedení uživatelské 2-bodové kalibrace. A to buď nastavením aktuální známé teploty senzoru, nebo aktuálním známým odporem senzoru (viz sekce Kalibrace). Třetí možnost nastavení kalibrace využívá trimru předchozí experimentální sestavy osazeného na desce ADDAJ2, kdy otáčením lze změnit hodnotu indikované teploty o $\pm 2^{\circ}\text{C}$. Stiskem klávesy je kalibrace ukončena a je indikován výsledek operace spolu s nastavenými koeficienty *offset* a *scale*. Pokud je kalibrace úspěšná, tak jsou přenastaveny hodnoty koeficientů *scale* a *offset*, které jsou navíc uloženy do Flash paměti.

Nastavení DA převodníku

Zápisem hodnoty DACCON = 0x07h do SFR registru DACCON nastavím DA převodník na rozlišení 12-bitů v rozsahu 0-5V (napájecí napětí).

3.2.6 Blokové schéma programu

Zjednodušené základní blokové schéma programu a použité funkce ukazuje obr. 27, spolu s některými funkcemi včetně použitých přerušení.

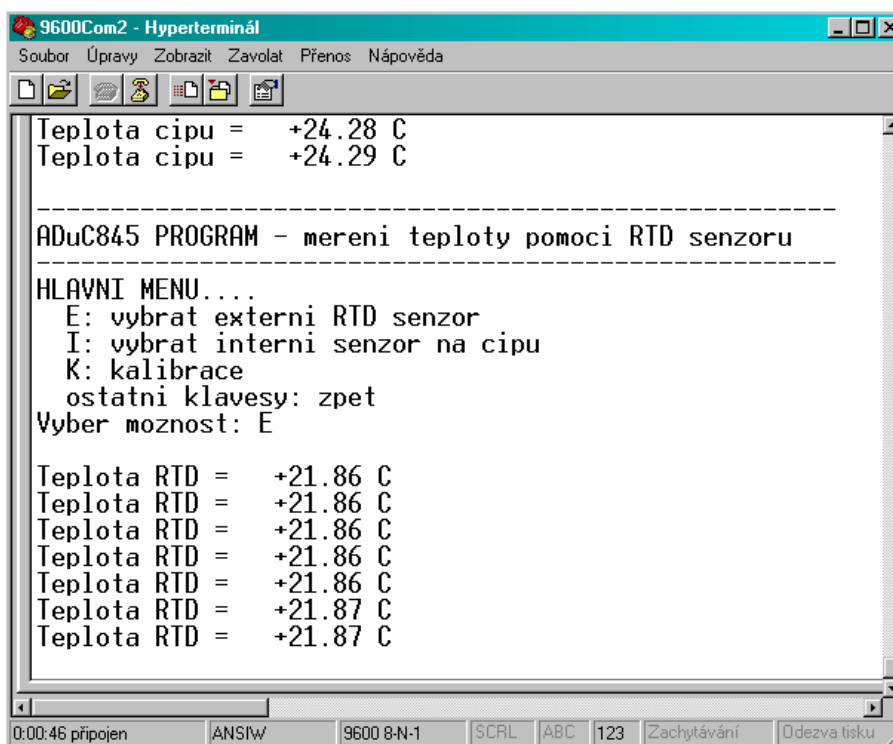


Obr. 27. Blokové schéma programu

Program nejdříve zjistí zda jsou v paměti EEPROM uloženy koeficienty získané při předchozí kalibraci, pokud tomu tak je, aktualizuje současné koeficienty. Poté postupně konfiguruje obvod pro komunikaci UART, AD převodníky, a DA převodník. Režim AD převodníků je nastaven SFR registrem ADCMODE na kontinuální vzorkování primárního i sekundárního převodníku. Program dále vstupuje do smyčky ve které kontroluje přítomnost nových dat získaných AD převodníkem. Pakliže jsou nová data přítomna, provede jejich zpracování (přepočet teploty, výstup na DAC a výstup v okně Terminálu pomocí UART). Ve smyčce je dále kontrolován vstup na komunikačním obvodu UART, který indikuje stisknutí klávesnice na PC. Tím se vyvolá obslužné menu. Aplikace tedy využívá dva zdroje přerušení. Přerušení vyvolané dokončením AD převodu, a přerušení vyvolané bitem registru RI, který indikuje přítomnost bytu přijatého komunikačním obvodem UART. Zdrojový kód programu v C jazyku je v příloze 6.

3.2.7 Ověření funkčnosti

Po uložení přeloženého kódu do programové paměti se aplikace automaticky spustí po restartu (nebo přivedení napájecího napětí). V Hyperterminálu systému Windows je pak indikována naměřená teplota výpisem řádku v intervalu zhruba 5Hz. Stisknutím klávesnice lze vyvolat obslužné menu. V tomto menu je možný výběr připojeného externího senzoru Pt100, nebo interního senzoru integrovaného v obvodu ADuC845. Výstup dat na obrazovce PC pomocí Terminálu systému Windows ukazuje obr. 28.



```
9600Com2 - Hyperterminál
Soubor Úpravy Zobrazit Zavolat Přenos Nápověda

Teplota cipu = +24.28 C
Teplota cipu = +24.29 C

-----
ADuC845 PROGRAM - mereni teploty pomoci RTD senzoru
-----
HLAVNI MENU...
E: vybrat externi RTD senzor
I: vybrat interni senzor na cipu
K: kalibrace
ostatni klavesy: zpet
Vyber moznost: E

Teplota RTD = +21.86 C
Teplota RTD = +21.86 C
Teplota RTD = +21.86 C
Teplota RTD = +21.86 C
Teplota RTD = +21.86 C
Teplota RTD = +21.87 C
Teplota RTD = +21.87 C

0:00:46 pripojen  ANSIW  9600 8-N-1  SCRL  ABC  123  Zachytavani  Odezva tisku
```

Obr. 28. Výstup programu v terminálu PC

Lze také provést kalibraci, a to zadáním buď teploty nebo odporu senzoru, nebo pomocí trimru nastavit aktuální měřenou teplotu. Při použití této funkce a při použití Matematické nebo Po částech lineární aproximace však velikost kódu překračuje povolený limit kompilery. Kalibrací se přednastaví koeficienty *scale* a *offset*. Tyto koeficienty lze také přednastavit zpět na původní hodnoty.

3.3 Experimentální měření

Hlavním úkolem experimentálního měření je ověření funkčnosti experimentální sestavy a především diskuse správnosti naměřených hodnot teploty. Nejprve provedu měření teploty podle koeficientů odvozených v kapitole 3.2.3. Poté provedu měření po provedení 1-bodové a 2-bodové kalibrace. Naměřené hodnoty porovnám spolu s hodnotami které odečtu na referenčním teploměru podle kterého bude provedena i kalibrace. Pro ověření správnosti funkce DA převodníku odečítám také hodnoty napětí naměřeného na výstupu DA převodníku.

3.3.1 Použité přístroje

Pro měření teploty používám kromě experimentální sestavy s obvodem ADuC ještě přístroj pro měření teploty a přístroj pro měření napětí.

Měření teploty

Pro měření teploty je to digitální teploměr GREISINGER typ GMH 3250 s termočlávkovou sondou. Základní parametry přístroje udávané výrobcem jsou:

- rozlišení: 0.1°
- pracovní teplota: -25 až +50 °C

Parametry připojené sondy udávané výrobcem jsou:

- termočlávková určená pro měření teploty kapaliny, typ K (NiCr-Ni)
- měřicí rozsah: -65°C až +550°C
- přesnost podle DIN EN 60584: $\pm 0.05\%$ (pro $T > -60^\circ\text{C}$)
- teplotní drift: 0.01%/K

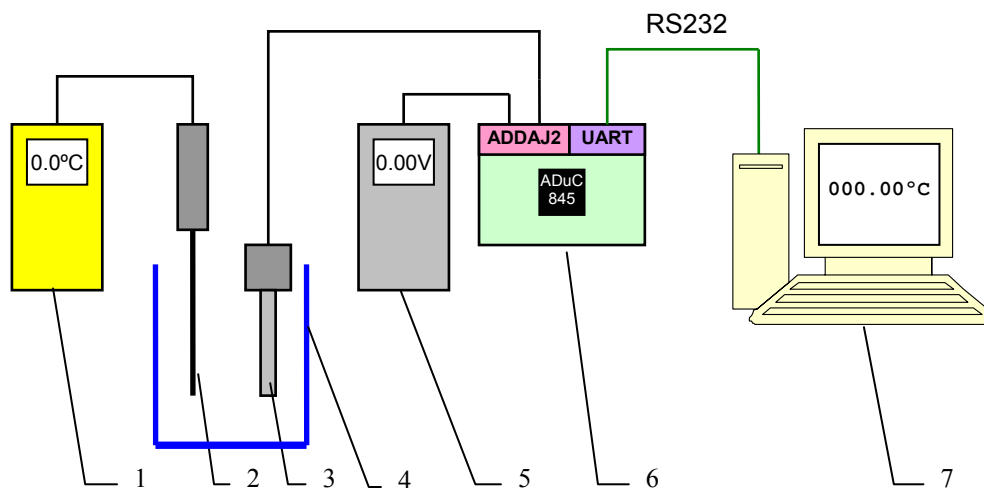
Měření napětí na výstupu DA převodníku

Jedná se o běžný digitální multimetr, konkrétně typ DT-830B

- nastavený rozsah: 0 až 2000 mV nebo 0 až 20V
- rozlišení: 1 mV
- přesnost: 0.3 %

3.3.2 Měření

Zjednodušené schéma sestavy měření je na obrázku č. 29.



1-digitální teploměr; 2-termočláňková sonda; 3-snímač Pt100;
4-nádoba s vodou; 5-multimetr; 6-sestava ADuC s deskou ADDAJ2; 7-PC

Obr. 29. Schéma sestavy měření

Měření budou prováděna ve vodě pro tři ustálené teplotní stavy

1. Směs ledu a vody reprezentující bod tuhnutí vody (0°C)
2. Vlažná voda ustálená při pokojové teplotě
3. Vroucí voda reprezentující bod varu vody (100°C)

Hodnoty tedy vždy odečítám po ustálení teploty v 10-ti sekundových intervalech postupně z terminálu PC, digitálního teploměru a multimetru. Celkem 20x pro každý bod měření. Snímač vkládám do vody tak, aby byl zcela ponořen až po závit. Měření jsou prováděna při okolní teplotě 22.6°C (naměřeno digitálním teploměrem) a tlaku 1010.7 hPa (přepočteno na hladinu moře).

Bez kalibrace

Bez kalibrace jsou hodnoty koeficientu scale a offset odvozené v kapitole 3.2.3 :

scale = 178.24 offset = -0.16

Tabulka VI. Naměřené hodnoty (bez kalibrace)

	led+voda			vlažná voda			vroucí voda		
	ADuC	GMH	ADuC	ADuC	GMH	ADuC	ADuC	GMH	ADuC
číslo měření	teplota [C]	teplota [C]	Vout [mV]	teplota [C]	teplota [C]	Vout [mV]	teplota [C]	teplota [C]	Vout [V]
1	1.52	0.0	725	21.83	20.1	1200	100.15	98.1	3.01
2	1.53	-0.1	726	21.83	20.1	1199	100.22	98.1	3.01
3	1.53	0.0	725	21.83	20.1	1199	100.16	98.2	3.01
4	1.53	0.0	726	21.83	20.1	1200	100.20	98.1	3.01
5	1.53	0.0	725	21.83	20.1	1200	100.24	98.2	3.01
6	1.53	-0.1	727	21.83	20.1	1199	100.17	98.1	3.01
7	1.53	-0.1	725	21.83	20.1	1199	100.20	98.1	3.01
8	1.53	-0.1	726	21.83	20.1	1199	100.16	98.2	3.01
9	1.53	-0.1	726	21.82	20.1	1199	100.17	98.3	3.01
10	1.53	0.0	727	21.83	20.1	1199	100.20	98.2	3.01
11	1.53	-0.1	725	21.83	20.1	1199	100.20	98.2	3.01
12	1.53	-0.1	725	21.83	20.1	1199	100.19	98.2	3.01
13	1.53	-0.1	726	21.82	20.1	1199	100.12	98.2	3.01
14	1.53	-0.1	726	21.82	20.1	1199	100.14	98.2	3.01
15	1.53	-0.1	726	21.82	20.1	1199	100.18	98.2	3.01
16	1.53	-0.1	725	21.82	20.1	1199	100.19	98.2	3.01
17	1.53	-0.1	726	21.82	20.1	1199	100.17	98.3	3.01
18	1.53	-0.1	726	21.82	20.1	1199	100.31	98.2	3.01
19	1.53	0.0	726	21.82	20.1	1199	100.31	98.3	3.01
20	1.53	0.0	726	21.82	20.1	1199	100.43	98.3	3.02
průměr	1.530	-0.065	725.8	21.826	20.10	1199.2	100.206	98.20	3.011
sm.odch	0.002	0.048	0.623	0.005	0.000	0.3571	0.070	0.067	0.002

1-bodová kalibrace

1-bodová kalibrace provedena po ustálení teplot v lázni voda-led. Nové koeficienty jsou:

scale = 177.1101 offset = -0.159

Tabulka VII. Naměřené hodnoty (1-bodová kalibrace)

	led+voda			vlažná voda			vroucí voda		
	ADuC	GMH	ADuC	ADuC	GMH	ADuC	ADuC	GMH	ADuC
číslo měření	teplota [C]	teplota [C]	Vout [mV]	teplota [C]	teplota [C]	Vout [mV]	teplota [C]	teplota [C]	Vout [V]
1	-0.10	-0.1	688	19.90	20.0	1155	98.21	98.4	2.96
2	-0.10	-0.1	688	19.90	20.0	1155	98.26	98.4	2.96
3	-0.10	-0.1	688	19.91	20.0	1155	98.26	98.4	2.96
4	-0.10	-0.1	688	19.91	20.0	1155	98.27	98.5	2.96
5	-0.10	-0.1	688	19.91	20.1	1155	98.26	98.4	2.96
6	-0.11	-0.1	688	19.91	20.1	1155	98.26	98.4	2.96
7	-0.10	-0.1	688	19.91	20.1	1155	98.22	98.4	2.96
8	-0.10	-0.1	688	19.91	20.1	1155	98.26	98.4	2.96
9	-0.10	-0.1	688	19.91	20.1	1155	98.24	98.4	2.96
10	-0.10	-0.1	688	19.91	20.1	1156	98.24	98.4	2.96
11	-0.10	-0.1	688	19.92	20.1	1155	98.25	98.5	2.96

12	-0.10	-0.1	688	19.92	20.1	1155	98.19	98.4	2.96
13	-0.10	-0.1	688	19.91	20.0	1155	98.16	98.4	2.96
14	-0.11	-0.1	688	19.92	20.0	1155	98.21	98.4	2.96
15	-0.10	-0.1	688	19.92	20.1	1155	98.20	98.4	2.96
16	-0.10	-0.1	688	19.91	20.0	1155	98.23	98.4	2.96
17	-0.10	-0.1	688	19.91	20.0	1155	98.23	98.4	2.96
18	-0.11	-0.1	688	19.91	20.0	1155	98.18	98.4	2.96
19	-0.10	-0.1	688	19.91	20.0	1155	98.16	98.4	2.96
20	-0.11	-0.1	688	19.91	20.0	1155	98.14	98.4	2.96
průměr	-0.102	-0.1	688	19.911	20.05	1155.1	98.222	98.41	2.96
sm.odch	0.004	0.000	0.000	0.005	0.050	0.218	0.038	0.030	0.000

2-bodová kalibrace

Druhý bod kalibrace proveden po ustálení teplot ve vlažné vodě. Nové koeficienty jsou:

scale = 178.1348 offset = -0.7382

Tabulka VIII. Naměřené hodnoty (2-bodová kalibrace)

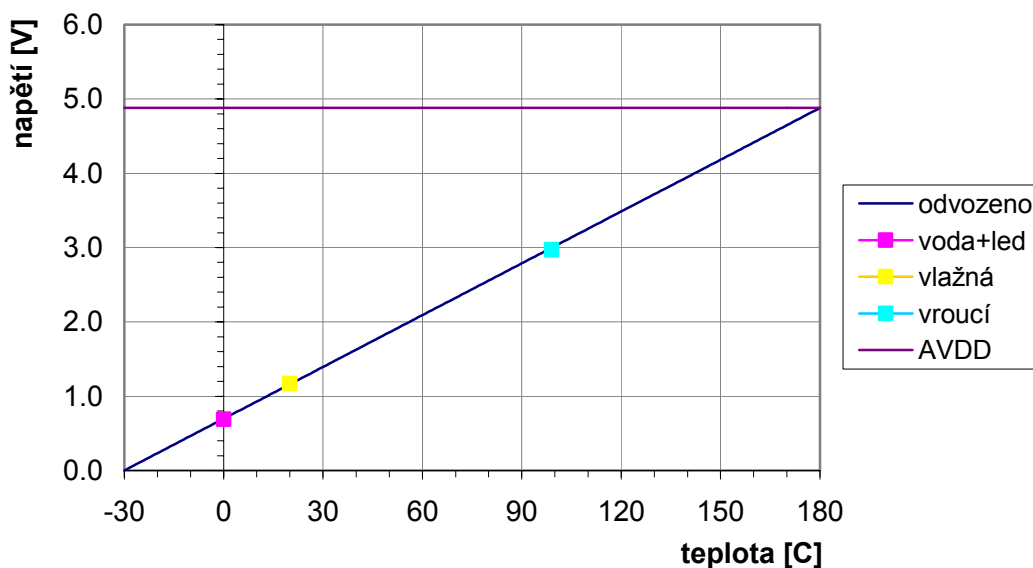
	led+voda			vlažná voda			vroucí voda		
	ADuC	GMH	ADuC	ADuC	GMH	ADuC	ADuC	GMH	ADuC
číslo měření	teplota [C]	teplota [C]	Vout [mV]	teplota [C]	teplota [C]	Vout [mV]	teplota [C]	teplota [C]	Vout [V]
1	-0.09	0.0	688	20.31	20.3	1165	98.59	98.4	2.97
2	-0.09	0.0	688	20.31	20.3	1164	98.58	98.4	2.97
3	-0.09	0.0	688	20.31	20.3	1164	98.55	98.4	2.97
4	-0.08	-0.1	688	20.32	20.3	1165	98.56	98.5	2.97
5	-0.09	0.0	688	20.32	20.3	1165	98.55	98.4	2.97
6	-0.08	0.0	688	20.32	20.3	1165	98.50	98.4	2.97
7	-0.09	0.0	688	20.32	20.3	1165	98.47	98.4	2.97
8	-0.09	0.0	688	20.31	20.3	1164	98.43	98.3	2.97
9	-0.08	0.0	688	20.31	20.3	1165	98.43	98.4	2.97
10	-0.08	0.0	689	20.33	20.3	1164	98.54	98.4	2.97
11	-0.08	0.0	688	20.33	20.3	1164	98.60	98.4	2.97
12	-0.09	0.0	688	20.33	20.3	1165	98.56	98.4	2.97
13	-0.08	0.0	688	20.33	20.3	1164	98.54	98.3	2.97
14	-0.09	0.0	688	20.34	20.3	1164	98.46	98.4	2.97
15	-0.09	-0.1	688	20.34	20.3	1165	98.48	98.3	2.97
16	-0.08	0.0	688	20.34	20.3	1164	98.50	98.4	2.97
17	-0.08	0.0	688	20.34	20.3	1165	98.43	98.4	2.97
18	-0.09	0.0	688	20.34	20.3	1165	98.46	98.3	2.96
19	-0.09	0.0	688	20.34	20.3	1165	98.40	98.3	2.96
20	-0.09	0.0	688	20.34	20.3	1165	98.41	98.3	2.97
průměr	-0.086	-0.01	688.1	20.327	20.3	1164.6	98.502	98.38	2.969
sm.odch	0.005	0.030	0.218	0.012	0.000	0.490	0.062	0.054	0.003

Pro přehlednost výsledků uvádím výsledné průměrné naměřené hodnoty teploměrů spolu s jejich rozdílem v tabulce IX. V tabulce jsou barevně rozlišeny kalibrační body. Zelené barvě odpovídá 1. kalibrační bod (voda+led). Žluté barvě odpovídá 2. kalibrační bod (vlažná voda).

Tabulka IX. Výsledné hodnoty

Bez kalibrace			
	1.voda+led	2.vlažná voda	3.vroucí voda
ADuC	1.53	21.83	100.21
Greisinger	-0.07	20.10	98.20
Rozdíl	-1.595	-1.725	-2.011
1-bodová kalibrace			
	1.voda+led	2.vlažná voda	3.vroucí voda
ADuC	-0.10	19.91	98.22
Greisinger	-0.10	20.05	98.41
Rozdíl	0.002	0.134	0.189
2-bodová kalibrace			
	1.voda+led	2.vlažná voda	3.vroucí voda
ADuC	-0.09	20.33	98.50
Greisinger	-0.01	20.30	98.38
Rozdíl	0.076	-0.026	-0.127

Hodnoty napětí na výstupu DA převodníku odpovídají předpokládaným hodnotám pro daný nastavený rozsah snímače (-30 až +180°C), což dokazuje graf na obrázku č. 30:

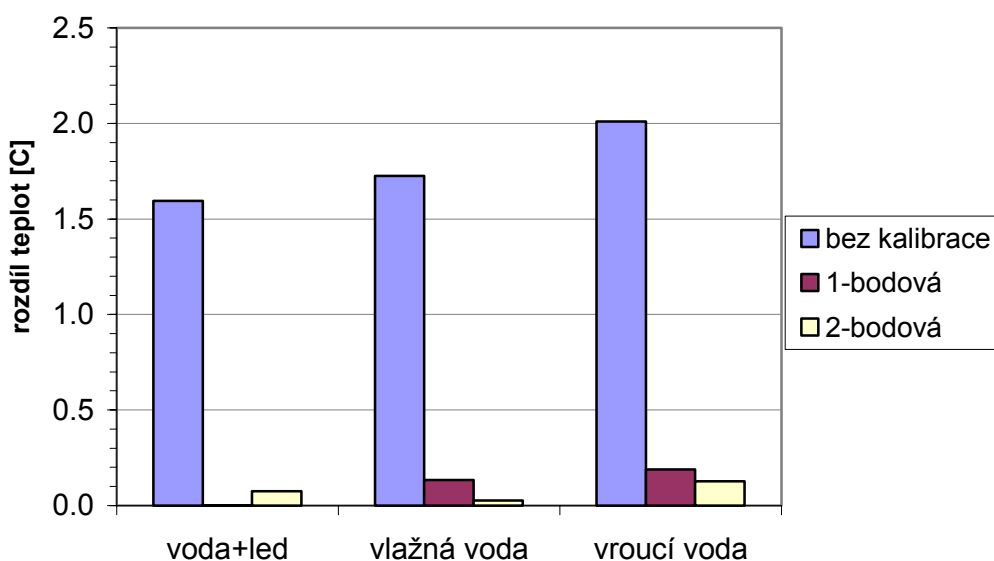


Obr. 30. Graf závislosti napětí výstupu DAC na teplotě

Kde modrá přímka znázorňuje funkční závislost DAC na teplotě podle vztahu (39) a zobrazené body odpovídají naměřeným hodnotám. Pro úplnost je v grafu zobrazeno také napájecí napětí mikrokonvertoru AVDD, které při dané konfiguraci DAC udává rozsah výstupního napětí.

3.4 Vyhodnocení a diskuse výsledků

Během měření jsem prakticky ověřil funkčnost navržené experimentální sestavy, a to jak po stránce realizovaného obvodu zapojení tak po stránce programové realizace. Ze získaných dat jsem pak vytvořil tabulky na základě kterých mohu provést vyhodnocení naměřených výsledků. Vyjádřením rozdílu hodnot v závislosti na kalibrační metodě pomocí tabulky IX dostávám následující graf:



Obr. 31. Graf rozdílu naměřených teplot

Z hodnot zobrazených v grafu je zřejmé, že největší rozdíl v naměřených teplotách byl při měření s nekalibrovaným snímačem. Hodnoty naměřené experimentální sestavou s obvodem ADuC845 jsou zhruba o 1.5 až 2.0 °C větší než hodnoty naměřené laboratorním digitálním teploměrem Greisinger. Otázkou samozřejmě zůstává, do jaké míry jsou hodnoty indikované digitálním teploměrem pravdivé? Bereme-li v úvahu teplotu naměřenou v lázni voda-led, pak průměrná hodnota -0.1°C je jistě přesnější než hodnota +1.52°C indikovaná pomocí obvodu ADuC a senzoru Pt100. Hodnotu naměřenou

laboratorním digitálním teploměrem tedy považuji za správnou, lépe řečeno je nejpřesnější, kterou za daných podmínek mám k dispozici, a je tedy vhodná k tomu, abych podle ní provedl kalibraci sestavy mnou vytvořeného snímače.

Prvním kalibračním bodem jsem tedy zvolil ustálenou teplotu lázně voda-led. Pro tento stav jsou hodnoty naměřené pomocí teploměrů téměř shodné, další dva ustálené stavy kapaliny se pak liší se vzrůstající tendencí (viz. fialové sloupce v grafu na obrázku 31). To vysvětluje korekce koeficientů provedená při kalibraci, kdy koeficient offsetu zůstává téměř stejný, a koeficient scale se zvětšil, čímž se změnila strmost funkční závislosti podle vztahu (38).

Druhým kalibračním bodem jsem pak zvolil ustálenou teplotu vlažné vody. V tomto ustáleném stavu je tedy rozdíl naměřený pomocí teploměrů minimální, mírně se zvětšil pro bod 1 (lázeň voda-led), zato se však zmenšil při měření teploty varu (žluté sloupce v grafu na obrázku 31). To potvrzují i korekce koeficientů. Pomocí této metody jsem získal jejich optimální hodnoty:

$$\text{scale} = 178.1348 \text{ (původně } 178.24)$$

$$\text{offset} = -0.7382 \text{ (původně } -0.16.24)$$

Pomocí 2-bodové kalibrace lze tedy skutečně dosáhnout lepších výsledků jak je popsáno v kapitole 2.4.3. Jednobodová kalibrace nabízí také uspokojivé výsledky, je však podle mého názoru vhodnější v případech, kdy je rozsah měřicího pásma teploty menší, neboť touto metodou se přenastavuje především koeficient scale, který má vliv na strmost závislosti odporu na výstupu AD převodníku (38) což by při chybném nastavení vedlo k výrazným rozdílům odchylek v celém pásmu měření. Naproti tomu nepřesnost způsobená chybným offsetem je v celém pásmu stejná, může však mít výrazný vliv na výsledek měření.

Korekcí provedenou 2-bodovou kalibrací se výrazně změnil jen koeficient offsetu senzoru. To potvrzuje správnost určení koeficientu scale, vypočítaného z hodnot dosazených do vzorce (36) (hodnota referenčního odporu a nastavené zesílení AD převodníku). Offset

senzoru se tedy výrazně zvětšil, což vysvětluje rozdíl naměřených hodnot před kalibrací o téměř 2°C.

Otázkou zůstává, co je zdrojem tak velké chyby. Pokud beru v úvahu hodnoty udávané firmou Analog Devices, kdy nejistoty způsobené šumem při AD převodu (obrázek 19) nepřevyšují 0.2°C (špička-špička), dále fakt, že při zpracování jsem použil matematickou metodu bez aproximací, tak hlavní zdroj nejistot musí být na straně samotného senzoru a jeho připojení k mikrokonvertoru. To lze jednoduše zjistit přímo měřením odporu senzoru včetně připojovacích kabelů při ustálené teplotě, a porovnáním s odporem vypočteným podle vztahu (17) z teploty indikované digitálním teploměrem, která je $t = 21.4\text{ }^{\circ}\text{C}$ a odpovídá jí odpor senzoru Pt100 $R_{21.4} = 108.34\text{ }\Omega$.

Hodnota odporu naměřená multimetrem na použitém snímači pak je:

$$R_{Pt100} = 109.7\text{ }\Omega \quad (T_{Pt100} = 24.911\text{ }^{\circ}\text{C})$$

Rozdíl odporů těchto dvou hodnot odporů pak dává offset:

$$\text{offset} = R_{21.4} - R_{Pt100} = -1.36$$

A to je mnohem více než offset nastavený pomocí kalibrace. Toto jsou pouze hrubé výpočty provedené na základě hodnoty odporu naměřené obyčejným multimetrem. Pro jejich detailnější prozkoumání by bylo zapotřebí kvalitnějších přístrojů. Podle nich, a na základě výsledků získaných dříve, lze však usuzovat, že největším zdrojem nejistot je právě použitý dvou vodičově připojený snímač Pt100. Výrobcem udávaný odpor vodičů snímače $0.16\text{ }\Omega$ ($0.08\text{ }\Omega/\text{m}$) tuto chybu zmenšuje jen minimálně. Výrazně lepším řešením by bylo 4-vodičové připojení kvalitnějšího snímače třídy A (viz 2.3.4 a P3).

Za zmínku stojí také fakt, že přesnost provedené kalibrace (tedy do 0.2° C) je pro běžnou praxi maximálně dostačující, neboť i v tomto řádu má nepatrná změna ponoru daného snímače vliv na naměřenou teplotu, a je také třeba dodržet neměnné podmínky okolního prostředí (např. zabránit průvanu, styku s jinými tělesy, atp.). Výsledky obdržené při měření na experimentální sestavě s obvodem ADuC845 a senzorem Pt100 tedy považuji za uspokojivé.

4 ZÁVĚR

Po podrobné analýze problematiky embedded systémů a prostudování způsobu měření a zpracování signálů v průmyslové automatizaci jsem úspěšně navrhl a realizoval dvě experimentální sestavy s vývojovým kitem ADuC845. První sestava slouží k vyhodnocení napětí přivedeného na analogový vstup obvodu a druhá sestava je určena k měření teploty pomocí odporového kovového snímače Pt100 určeného pro měření teplot v rozsahu -30°C až 180°C . Tato sestava řeší kromě správného připojení senzoru k obvodu ADuC845 především programovou realizaci, a tedy správný AD převod signálu a jeho následné zpracování včetně linearizace, a také výstup naměřených hodnot na obrazovku PC pomocí komunikačního rozhraní RS-232. Uživatelské menu aplikace také umožňuje provedení uživatelské kalibrace, po které jsou upravené hodnoty patřičných koeficientů uloženy v energeticky nezávislé paměti a jsou tedy k dispozici po každém zapojení sestavy. Navíc sestava pracuje jako převodník teploty na napětí, a to díky DA převodníku který nastavuje hodnotu výstupního napětí v závislosti na naměřené teplotě a daného rozsahu měření.

Pomocí druhé realizované sestavy jsem provedl měření teploty vody pro několik ustálených teplotních stavů a získané hodnoty porovnal s hodnotami naměřenými pomocí laboratorního elektronického teploměru. Výsledky měření ukázaly, že naměřené hodnoty se liší řádově o jednotky $^{\circ}\text{C}$ (až 2°C) a bylo tedy vhodné experimentální sestavu pro měření teploty kalibrovat. Kalibrace byla provedena podle laboratorního teploměru a následná měření ukázala, že rozdíl hodnot indikovaných experimentální sestavou a laboratorním teploměrem byl řádově pouze v desetinách $^{\circ}\text{C}$ (maximálně 0.2°C) v rozsahu 0 až 100°C , což při daných podmínkách považuji za velmi dobrý výsledek. Původní rozdíl několika stupňů, naměřený při teoreticky odvozených koeficientech, byl podle mého názoru způsoben především nejistotami použitého snímače Pt100 a jeho připojovacích vodičů.

Obvod ADuC845 se ukázal jako ideální prostředek pro použití v průmyslové automatizaci, a to především pro zpracování senzorických signálů odporových (odporové snímače teploty, tenzometry,...) a také termočlánekových. Obsahuje množství podpůrných obvodů, jako například referenční vstupy a zdroje budícího proudu, díky kterým je připojení těchto

senzorů možné při použití minimálního počtu jakýchkoli dalších elektronických prvků. Za pomoci vestavěného mikroprocesoru 8051 je pak možná realizace programového vybavení obvodu pomocí standardních vývojových prostředků. Takto je možné programové zpracování senzorických signálů včetně linearizace. Energeticky nezávislá paměť je vhodná pro permanentní uložení dat. Obvod obsahuje řadu dalších periferních obvodů, a nabízí tak nepřehledné množství dalších možných aplikací a řešení, jako například připojení LCD displeje a klávesnice. Především pomocí programového vybavení je pak možná realizace některého typu regulátoru (včetně moderních adaptivních). Toto mohou být vhodná témata dalších Diplomových prací.

Obecně lze říci, že embedded systémy nalézají uplatnění ve všech odvětvích průmyslu. Řešení s obvodem ADuC845 mohou být vhodná především v aplikacích které si žádají přesné měření a zpracování nízkofrekvenčních signálů, jako například:

- inteligentní (smart) snímače a převodníky teploty nebo tlaku
- regulační smyčky (4 – 20 mA)
- zařízení pro monitorování stavu pacientů
- přenosné měřicí a testovací přístroje

Současná praxe vyžaduje prostředky pro měření obecných (neelektrických) veličin, které mohou být převedeny na napětí nebo proud, digitalizovány a dále předávány v digitální podobě. Embedded systémy jsou vhodné pro řešení moderně konfigurovaných víceúrovňových měřicích a řídicích systémů. Lze očekávat, že kvalita a komfort našeho života budou v budoucnu stále více ovlivňovány „inteligencí“ těchto systémů.

SEZNAM POUŽITÉ LITERATURY

- [1] Hruška, F.: Technické prostředky automatizace IV. Snímače, převodníky a regulátory, průmyslová výpočetní technika a ovládací jednotky. UTB ve Zlíně, FT, září 2001, s.107. ISBN 08-7318-026-X.
- [2] Hruška, F.: Technické prostředky automatizace III. Senzory, jejich principy a funkce. UTB ve Zlíně, FT, únor 2002, s.118. ISBN 80-7318-053-7.
- [3] ĎAĎO, S.,KREIDL,M.: Senzory a měřicí obvody. Praha: ČVUT, 1996.
- [4] SKALICKÝ, P.: Mikroprocesory řady 8051. Praha: BEN, 1998. 159 s. ISBN 80-86056-39-2.
- [5] MACHO, T.: Vestavné systémy. Časopis Automa (2002), č. 12.
- [6] PALENČÁR, R.,VDOLEČEK, F.:Nejistoty v měření. Časopis Automa (2001), č.7-8.
- [7] ANALOG DEVICES: ADuC845 Datasheet, Analog Devices, Inc., 2004. Dostupné z: <<http://www.analog.com/>>.
- [8] KING,G.,FUKUSHIMA, T.: RTD Interfacing and Linearization Using an ADuC8XX MicroConverter, Analog Devices, Inc., 2004. Dostupné z: <<http://www.analog.com/>>
- [9] JALOVECKÝ, R.: Mikrokonvertor ADuC812, HW Server, 2001. Dostupné z: <<http://www.hw.cz/>>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

A_{ADC}	zesílení AD převodníku
AD	analogově-číslicový
ADC	analogově-číslicový převodník
ADC_{NORM}	normalizovaná hodnota AD převodu v rozsahu 0 až 1
ADuC	analogově-číslicový mikrořadič (mikrokonvertor)
AIN	analogový vstup
ALU	aritmeticko-logická jednotka
AVDD	napájení analogové části obvodu
BUF	vyrovnávací paměť
CPU	centrální procesorová jednotka
D.C.	střída impulsního signálu (Duty Cycle)
DA	číslicově-analogový
DAC	číslicově-analogový převodník
DVDD	napájení číslicové části obvodu
EEPROM	elektricky mazatelná programovatelná paměť
EPROM	elektricky programovaná paměť
Flash	programovatelná stálá paměť
IC	integrovaný obvod (Integrated Circuit)
I_{EXC}	budící proud
JSA	jazyk symbolických adres
LSB	bit s nejnižší váhou
μC , MC	mikropočítač
MSB	bit s nejvyšší váhou
MUX	multiplexer

NTC	odporové polovodičové snímače - negistory
offset	koeficient offsetu senzoru
PGA	programově nastavitelný zesilovač
PSM	obvod pro hlídání napětí (Power Supply Monitor)
PTC	odporové polovodičové snímače - pozistory
PWM	pulsně šířková modulace (Pulse Width Modulation)
R_0	odpor teploměru při teplotě 0°C
R_{CAL}	kalibrovaná hodnota odporu
R_{REF}	referenční odpor
$R_{RTD}(t)$	funkční závislost odporového teplotního čidla na teplotě
RAM	paměť s přímým přístupem (Random Access Memory)
ROM	paměť určená pouze ke čtení (Read-Only Memory)
RTC	hodiny reálného času
RTD	odporové kovové snímače teploty (Resistance Temperature Detector)
scale	koeficient pro výpočet odporu z výstupní hodnoty ADC
SFR	registr speciálních funkcí
SIA	systém integrované automatizace
span	rozsah vstupního napětí AD převodníku
T_{MAX}	maximální teplota
T_{MIN}	minimální teplota
$T_{RTD}(r)$	funkční závislost teploty odporového teplotního čidla na odporu
UART	komunikační obvod
V_{IN}	napětí na vstupu AD převodníku
WD	hlídací pes (watchdog)

SEZNAM OBRÁZKŮ

Obr. 1. Prvky rozšiřující vlastnosti základního jádra procesoru 8051	15
Obr. 2. Vnitřní bloková struktura procesoru 8051	16
Obr. 3. Tvorba programu v jazyce symbolických adres a jazyce C	20
Obr. 4. Vnitřní blokové schéma obvodu ADuC845	22
Obr. 5. Datové funkční registry AD převodníku (primární ADC)	23
Obr. 6. Zjednodušené schéma měřicího řetězce AD převodníku	24
Obr. 7. Unpolární/bipolární připojení vstupních kanálů	25
Obr. 8. Princip mapování programové paměti (pro konfiguraci 62kB)	27
Obr. 9. Princip mapování datové paměti RAM a Flash/EEPROM	28
Obr. 10. Funkce SFR registrů	29
Obr. 11. Časování sériového kanálu v módu 1	32
Obr. 12. QuickStart vývojový kit	33
Obr. 13. Blokové schéma měřicího řetězce se senzorem	35
Obr. 14. Závislost odporu senzoru na teplotě	44
Obr. 15. Schéma zapojení odporového teplotního senzoru	45
Obr. 16. Graf chyb polynomiálních funkcí	49
Obr. 17. Chyba lineární aproximace	50
Obr. 18. Chyba po částech lineární aproximace (rozsah -200°C až +850°C)	52
Obr. 19. Vliv šumu AD převodníku na výslednou teplotu	56
Obr. 20. Teplotní drift vs. teplota senzoru	57
Obr. 21. Odporový dělič	58
Obr. 22. Připojení děliče k mikrokonvertoru	59
Obr. 23. Blokové schéma těla programu a hlavní funkce	61
Obr. 24. Ukázka výstupu dat na PC	62
Obr. 25. Graf závislosti napětí U ₁ , U ₂ , U ₃ na poloze trimru	62
Obr. 26. Připojení čidla Pt100 k mikrokonvertoru	64
Obr. 27. Blokové schéma programu	70
Obr. 28. Výstup programu v terminálu PC	71
Obr. 29. Schéma sestavy měření	73
Obr. 30. Graf závislosti napětí výstupu DAC na teplotě	76
Obr. 31. Graf rozdílu naměřených teplot	77

SEZNAM TABULEK

Tabulka I. Vybrané mikrokontrolery 80C51	18
Tabulka II. Adresy přerušení	30
Tabulka III. Metody linearizace	47
Tabulka IV. Charakteristiky polynomiálních funkcí	49
Tabulka V. Výběr odporu a koeficientu span	66
Tabulka VI. Naměřené hodnoty (bez kalibrace).....	74
Tabulka VII. Naměřené hodnoty (1-bodová kalibrace).....	74
Tabulka VIII. Naměřené hodnoty (2-bodová kalibrace)	75
Tabulka IX. Výsledné hodnoty	76

SEZNAM PŘÍLOH

- P 1 : Referenční příručka obvodu ADuC845
- P 2 : Schéma desky vývojového kitu QuickStart
- P 3 : Schéma desky připojení ADDAJ2
- P 4 : Snímače teploty REGMET
- P 5 : Programový kód experimentální sestavy č.1
- P 6 : Programový kód experimentální sestavy č.2

Data Transfer Operations

Arithmetic Operations		Logical Operations
ADD A, Rn	1	1
ADD A, @Ri	1	2
ADD A, direct	2	2
ADD A, #data	2	2
ADDC A, Rn	1	1
ADDC A, @Ri	1	1
ADDC A, direct	2	2
ADDC A, #data	2	2
SUBB A, Rn	1	1
SUBB A, @Ri	1	1
SUBB A, direct	2	2
SUBB A, #data	2	2
INC A	1	1
INC Rn	1	1
INC @Ri	1	2
INC direct	2	2
INC DPTR *	1	3
DEC A	1	1
DEC Rn	1	1
DEC @Ri	1	2
DEC direct	2	2
MUL AB	1	9
DIV AB	1	9
DA A	1	2

* INC DPTR increments the 24bit value DPH/DPL.

ANL	A, Rn
-----	-------

ANL A,R1	1	1	
ANL A,@R1	1	2	
ANL A,direct	2	2	
ANL A,#data	2	2	
ANL direct,A	2	2	
ANL direct,data	3	3	
ORL A,R1	1	1	
ORL A,@R1	1	2	
ORL A,direct	2	2	
ORL A,data	2	2	
ORL direct,A	2	2	
ORL direct,data	3	3	
XRL A,R1	1	1	
XRL A,@R1	1	2	
XRL A,direct	2	2	

Program Branching

XRL	direct	A		
XRL	direct	#data		
CLR	A		clear A to zero	1
CPL	A		complement A	1
RLC	A		rotate A left	1
RLC	A		...through C	1
RR	A		rotate A right	1
RR	A		...through C	1
SWAP	A		swap nibbles	1

addressing

@Ri	indirect addressing using RO or R1
direct	8-bit internal address (00H-FFH)
#data	8-bit constant included in instruction
#data16	16-bit constant included in instruction
bit	8-bit direct address of bit
rel	signed 8-bit offset
addr11	11-bit address in current 2K page
addr16	16-bit address
x	any of Rn, @Ri, #data

ADD A,x C = carry out of bit 7

RRC A	C = ACC.7
RLC A	C = ACC.0
SETB C	C = 1
CLR C	C = 0
ANL C,bit	C = C and bit
ANL C,bit	C = C and NOTbit
ORL C,bit	C = C or bit
ORL C,bit	C = C or NOTbit
MOV C,bit	C = bit
CINE x,y,rel	C = (x+y)

MOFF
CSP

CSP	27	29	SDATA (RC)
	28	30	P2.0 / MOSQ (SPH)
	29	31	P2.1 / MOSI (SPI)
	30	32	P2.2 / MISO (SPI)
	31	33	P2.3 / SS / T2
	32	34	XTAL1 (in)
	33	35	XTAL2 (out)
	34	36	DVcc
	35	37	DVss
	-	38	DGN2
CSP	36	39	P2.4 / T2EX
	37	40	P2.5 / PWM0
	38	41	P2.6 / PWM1
	39	42	P2.7 / PWMCLK
	40	43	EA
	41	44	PSEN
CSP	42	45	ALE
	43	46	P0.0 / AD0
	44	47	P0.1 / AD1
	45	48	P0.2 / AD2
	46	49	P0.3 / AD3
	47	50	DGN0
CSP	48	51	DVcc
	49	52	P0.4 / AD4
	50	53	P0.5 / AD5
	51	54	P0.6 / AD6
	52	55	P0.7 / AD7
	53	56	P1.0 / P11

ANL	A, Rn	1
-----	-------	---

34	35	DATA
35	37	DATA
-	38	DATA
36	39	P2.4 / T2EX
37	40	P2.5 / PWM0
38	41	P2.6 / PWM1
39	42	P2.7 / PWMCLK

111

The figure illustrates the memory map of the 8051 microcontroller, showing three distinct memory segments. Each segment is represented by a vertical bar with a width indicating its size and a height indicating its address range. The segments are labeled with their respective sizes, internal code space, and the location of NOP instructions.

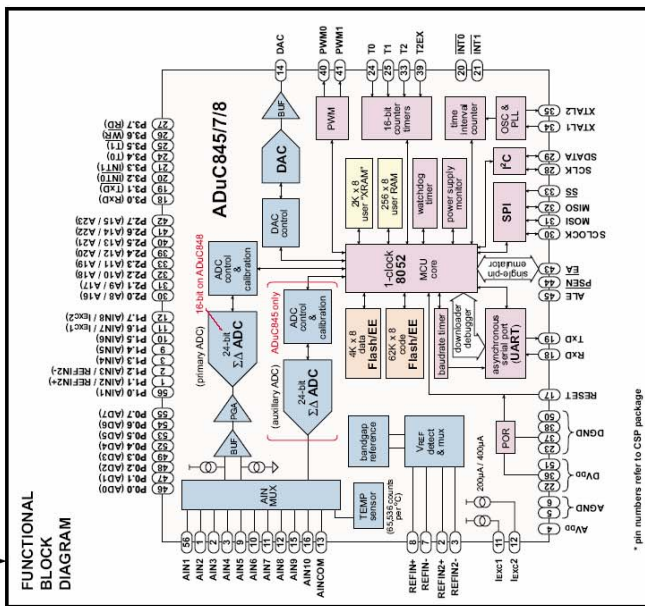
- Segment 1 (Left):** Labeled "8" at the top. It shows an internal code space (Flash/EE) from 0000h to 0000h and a 2000h NOP instructions block.
- Segment 2 (Middle):** Labeled "32" at the top. It shows an internal code space (Flash/EE) from 0000h to 8000h and a 2000h NOP instructions block.
- Segment 3 (Right):** Labeled "62" at the top. It shows an internal code space (Flash/EE) from 0000h to F800h and a 2000h NOP instructions block.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523
--	---	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Interrupt Bit	Interrupt Name	Vector Address	Relative Priority
PSMCON.5	Power Supply Monitor Interrupt	43h	1
WDS	WatchDog Timer Interrupt	5Bh	2
IE0	External Interrupt 0	03h	3
RDY0/RDY1	End of ADC Conversion Interrupt	33h	4
TF0	Timer0 Overflow Interrupt	0Bh	5
IE1	External Interrupt 1	13h	6
TF1	Timer1 Overflow Interrupt	1Bh	7
ISPI//IC1	SPI//I ² C Interrupt	3Bh	8
RI/TI	UART Interrupt	23h	9
TF2/EXF2	Timer2 Interrupt	2Bh	10
TIMECON.2	Time Interval Counter Interrupt	53h	11

© 2004 Analog Devices, Inc. All rights reserved. Trademarks and registered trademarks are the property of their respective owners. Analog Devices, the Analog Devices logo, and AD are trademarks of Analog Devices, Inc. in the U.S. and other countries. ADI is a registered trademark of Analog Devices, Inc. in the U.S. and other countries. Companies convey a license for the purchaser to use the software and hardware components in an ITC system, provided that the system conforms to the ITC standardization of the ITC system. For more information, visit www.analog.com/microconverter.
G4952Z-04/04(01)
Analog Devices, Inc.
100 Technology Park
Norfolk, MA 01956, U.S.A.

ADuC845/ADuC847/ADuC848
MicroConverter® Quick Reference



ADC:

24-bit* primary ADC, differential w/ programmable gain
24-bit auxiliary ADC, single-ended w/ fixed gain (ADuC845 only)
10-channel input mux (*ADC is 16bit on ADuC848)

12-bit, 15 μ s, voltage output

Flash/EEPROM: up to 64KB Flash/EEPROM

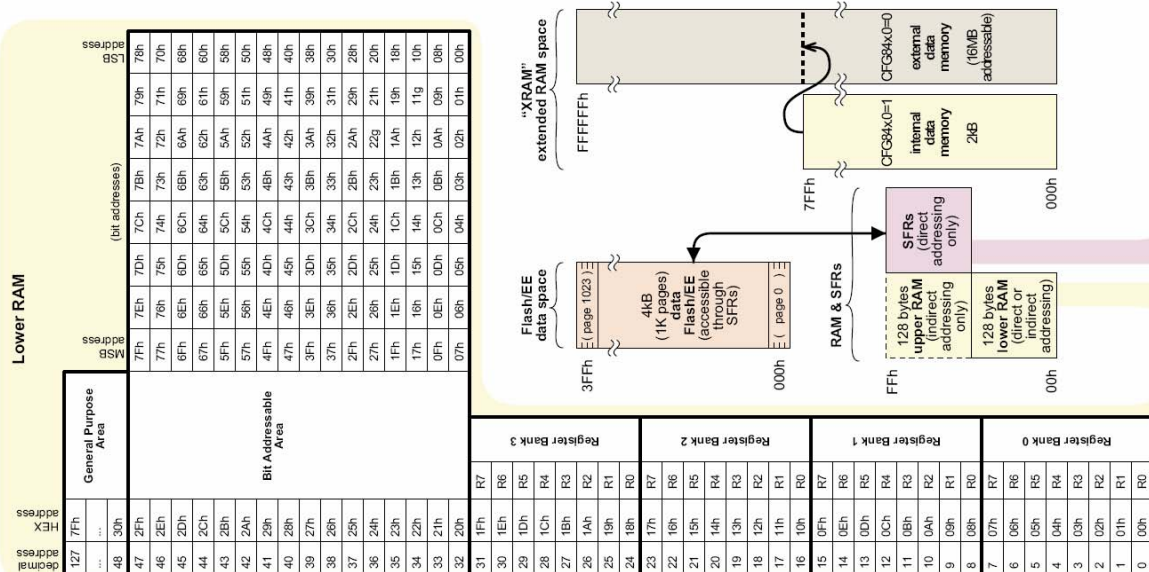
Microcontroller:

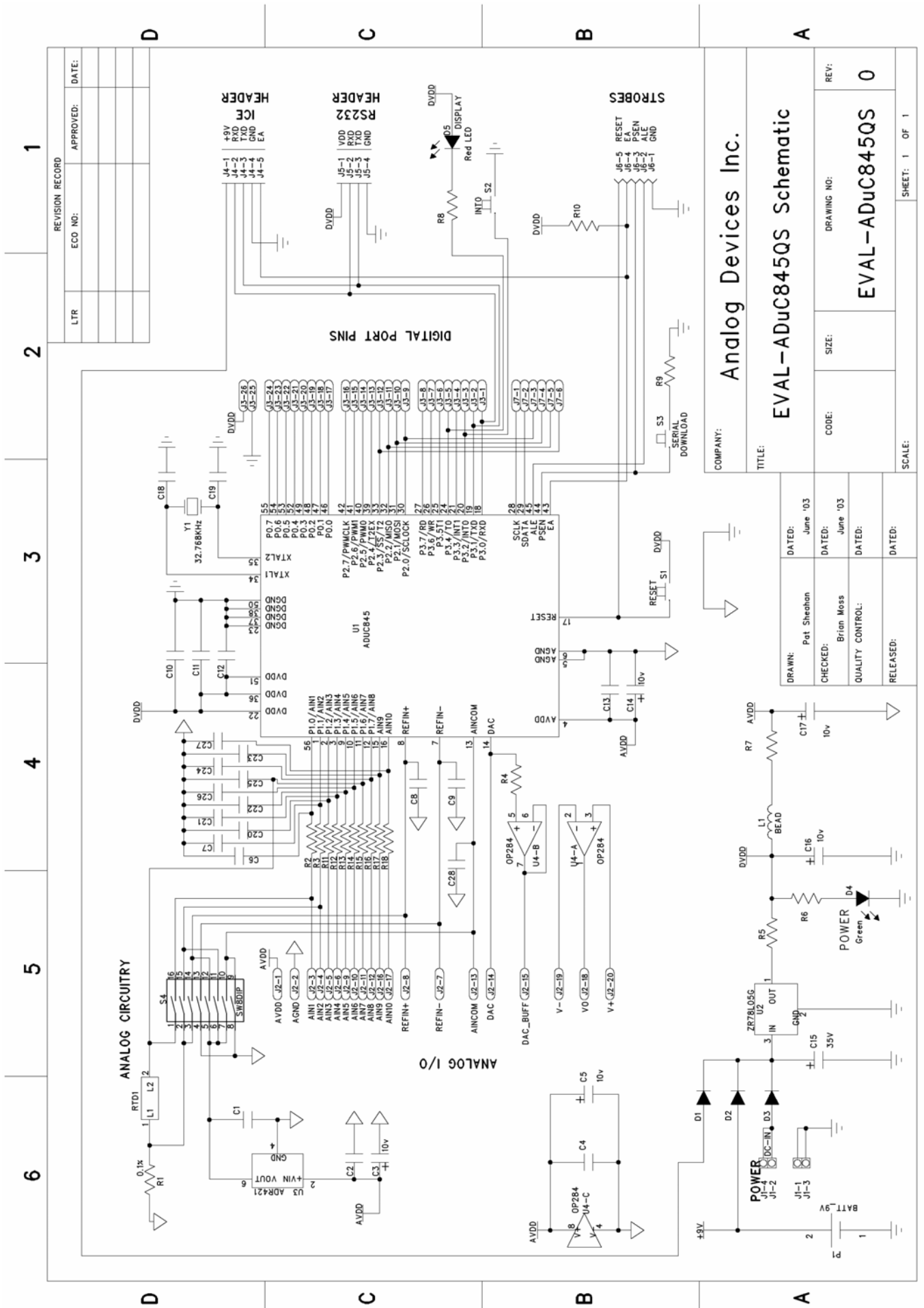
32 I/O lines, programmable PLL clock (98.3kHz to 12.6MHz from 32kHz crystals)

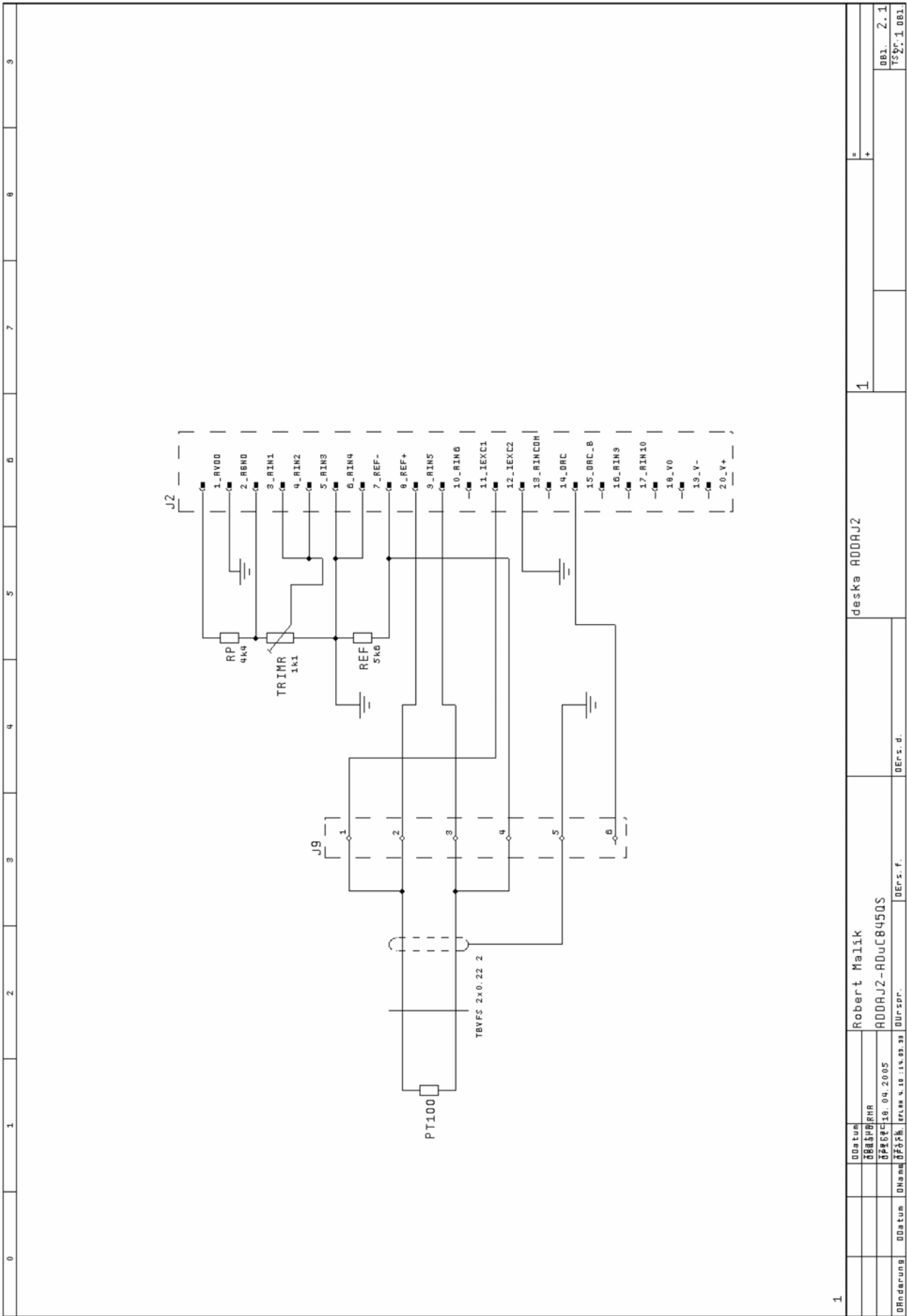
on-chip download/debug & single-pin emul

temperature sensor, power

Data Memory: RAM, SFRs, user Flash/EE (all read/write)







1

		Robert Malik		deska ADOAJ2		1							
		00atum		08.05.2005		ADOAJ2-ADuC845QS							
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									
		00atum		18.04.2005									

Snímače teploty s kabelovým vývodem



Tyto snímače teploty lze použít v prostředí, kde není možno z prostorových či jiných důvodů umístit klasické typy snímačů s hlavicí. Uvedené verze pouzder jsou využitelné pro všechny běžně používané typy teplotních čidel.

Typ čidla / Typ pouzdra	SK1	SK2	SK3	SK4	SK5	SK7
Ni1000/5000 ppm	SK1L-X	SK2L-X	SK3L-X	SK4L-X	SK5L-X	SK7L-X
Ni1000/6180 ppm	SK1S-X	SK2S-X	SK3S-X	SK4S-X	SK5S-X	SK7S-X
Ni891/6371 ppm	SK1J-X	SK2J-X	SK3J-X	SK4J-X	SK5J-X	SK7J-X
Pt100/3850 ppm	SK1P-X	SK2P-X	SK3P-X	SK4P-X	SK5P-X	SK7P-X
Pt500/3850 ppm	SK1PB-X	SK2PB-X	SK3PB-X	SK4PB-X	SK5PB-X	SK7PB-X
Pt1000/3850 ppm	SK1PA-X	SK2PA-X	SK3PA-X	SK4PA-X	SK5PA-X	SK7PA-X
NTC 20 kΩ	SK1H-X	SK2H-X	SK3H-X	SK4H-X	SK5H-X	SK7H-X

X - určuje délku kabelu danou v m

Základní technické parametry

Teplotní rozsah	-30 až 180°C *
Třída přesnosti	B
Krytí	IP 67
Měřicí proud Ni1000, NTC20 kΩ, Pt1000	max. 0,5 mA
Pt500	max. 1 mA
Pt100	max. 2 mA
Standardní délka vodiče	1 m
Vnitřní odpor vodiče R_v na 1 m kabel 2 x 0,35 mm ²	0,05 Ω
Vnitřní odpor vodiče R_v na 1 m kabel 2 x 0,22 mm ²	0,08 Ω
Vnitřní odpor vodiče R_v na 1 m kabel 2 x 0,14 mm ²	0,13 Ω
Vnitřní odpor vodiče R_v na 1 m kabel 4 x 0,56 mm ²	0,03 Ω

* Speciální sondy s kabely LTG a GGCV lze použít až do teploty 350°C.

Typy pouzder

SK1x - mosazné pouzdro:

SK1x-1TB... stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)

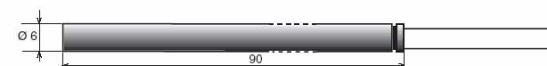
SK1x-1LY... stíněný kabel LYICY 2 x 0,14 mm²



SK2x - nerezové pouzdro:

SK2x-1TB... stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)

SK2x-1LT... stíněný kabel LTG 2 x 0,35 mm² (do 350°C)



SK3x - nerezové pouzdro:

kabel CSSS 2x0,35 mm²

nebo 3x0,35 mm²



SK4x - hliníkové pouzdro:

SK4x-1TB... stíněný kabel TBVFS 3x0,56 mm², 4 x 0,22 mm²

SK4x-1TV... nestíněný kabel TBVS 3 x 0,22 mm²

SK4x-1CS... nestíněný kabel CSSS 3 x 0,35 mm²



SK5x - nerezové pouzdro:

SK5x-1TB... stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)

SK5x-1GG... stíněný kabel GGCV 4 x 0,14 mm² (do 350°C)



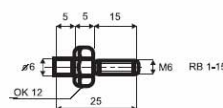
SK7x - nerezové pouzdro:

SK7x-1TB... stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)

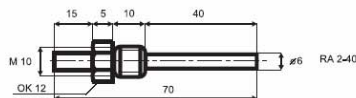


Snímače teploty v pouzdrech se závitem

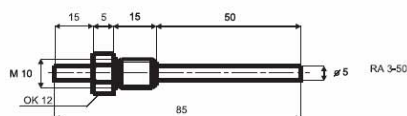
RB1x - nerezové pouzdro pro univerzální použití;
stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)



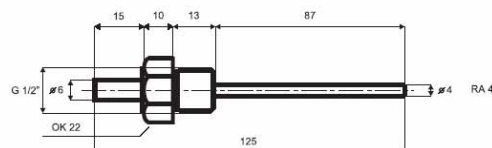
RA2x - nerezové pouzdro pro univerzální použití;
stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)



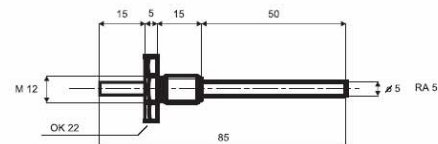
RA3x - nerezové pouzdro pro univerzální použití;
stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)



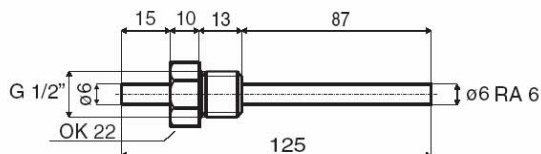
RA4x - nerezové pouzdro pro klimatizační kanály nebo potrubí;
stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)



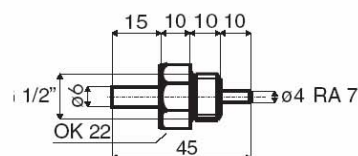
RA5x - nerezové pouzdro pro univerzální použití;
stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)



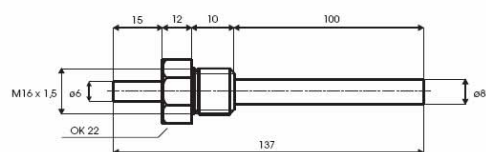
RA6x - nerezové pouzdro pro univerzální použití;
stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)



RA7x - nerezové pouzdro pro univerzální použití;
stíněný kabel TBVFS 2x0,22 mm² (nebo třívodičově)



RA 10X - nerezové pouzdro pro kompresory



Způsob objednání

V objednávce se uvádí typ a počet snímačů, např.: **5 ks snímače SK3P/T- 3,**
tj. snímač Pt100 v pouzdře SK3, třívodičové zapojení,
délka kabelu - 3m

Poznámka: 1) U snímačů Pt100 a Pt500 je nutné uvést verzi zapojení, tj. u dvou vodičového zapojení např. SK3P/D
a u třívodičového zapojení např. SK3P/T.
2) U některých typů snímačů lze kabelový vývod opatřit kovovou chráničkou.

```

//*****
// Soubor      : u_adc.c - experimentalni sestava c.1
// Autor       : Robert Malik
// Datum       : 18 dubna 2005
// Kompiler    : Keil C51
// Hardware     : ADuC845 vyvojovy kit, napetovy delic na desce ADDAJ2
// Popis        : Proveďte AD převod signálu přivedených na vstupech
//                AIN1-AIN2, AIN3-AIN4, a AIN1-AINCOM a jejich výpis
//                v okně Hyperterminalu
//*****

// použijte knihovny
#include <ADuC845.h>    // SFR registry pro ADuC845
#include <stdio.h>      // standardní I/O funkce

// definice konstant
#define span 1.28      // rozsah vstupního signálu

// deklarace globálních proměnných
float U1, U2, U3;

// lokální funkce
void Delay(int length); // čekací smyčka
void BlinkTwice();      // indikace pomocí LED (2x blikne)
float GetU(int gate);    // provede AD převod na daném kanálu

//-----
// Hlavní program
void main (void){
    // Prednastavení hodnot
    U1=0;
    U2=0;
    U3=0;

    // Konfigurace UART
    T3CON = 0x83;    // nastavení časovace 3...
    T3FD = 0x12;     // ...9600 Baudu
    SCON = 0x52;     // konfigurace UART s použitím časovace 3

    // Konfigurace primárního převodníku ADC0
    SF = 0x45;       // rychlost 19.79Hz
    ADC0CON1 = 0x27; // unipolární kodování, 0-1.28V.

    for(;;){
        U1=GetU(0x0A); // bipolární AIN1-AIN2
        U2=GetU(0x0B); // bipolární AIN3-AIN4
        U3=GetU(0x00); // unipolární AIN1-AINCOM
        printf("BI: U1= %1.2f V, U2= %1.2f V; UNI: U3= %1.2f V (U1+U2)\n", U1, U2, U3);
    // Výstup hodnot přes UART
        BlinkTwice(); // indikace pomocí LED
    }
}
//-----

float GetU(int gate) // Funkce pro AD převod na daném kanálu
// vstup: nastavení ADC0CON2 (hexadecimálně)
// 0x0A=AIN1-AIN2
// 0x0B=AIN2-AIN3 viz. manual
// Výstup: napětí
{ float ADC0=0;
  float Uin=0;
  ADC0CON2 = gate; // Int. ref. 1.25V
  ADCMODE = 0x22; // ADC0, "Chop Enabled", jednorázový převod primárního ADC
  while (!RDY0) {} // čekací smyčka (dokud není hotov převod)
  ADC0 = (ADC0H*0x1000 + ADC0M*0x100 + ADC0L)/16777216.0; // načtení hodnot ze SFR
  registru
  Uin = ADC0*span; // převod hodnoty
  RDY0=0; // nulování RDY0 bitu
  return(Uin); // výstup výsledné hodnoty napětí
}

void BlinkTwice() // Indikace pomocí LED (2x blikne)
{ T0 ^=1;
  Delay(5000);
  T0 ^=1;
  Delay(5000);
  T0 ^=1;
}

```

```
    Delay(5000);  
    T0 ^=1;  
}  
  
void Delay(int length) // Casova prodleva  
{ while (length >=0)  
    length--;  
}
```

```

//*****
// Soubor      : RTD845DA.c - experimentální sestava c. 2
// Autor       : Robert Malik, AD
// Datum       : 18 dubna 2005
// Kompilér    : Keil C51
// Hardware     : ADuC845 na desce vyvojového kitu QuickStart+
//              : Deska pro obsluhu analogových vstupu/vystupu
//              : ADDAJ2 s připojeným snímacem Pt100
// Popis        : AD převod signálu odporového senzoru PT100
//              : (RTD) s linearizací. Výsledná teplota odesílána
//              : jako ASCII text přes UART rychlostí 9600 baud.
//              : Vyhodnocuje také teplotu interního senzoru
//              : obvodu ADuC845(chip).
//              : Nastavení DAC podle teploty v rozahu 0-5V.
//              : Vstupně/výstupní nastavení pomocí uživ. menu.
// Poznámka     : Funkce použití kalibrace trimrem v poznámkách (aby velikost
//              : kodu nepřesáhla limit kompilátoru 4kB)
//*****

// použité knihovny...
#include<ADuC845.h> // SFR registry pro ADuC845
#include<stdio.h>   // standardní I/O funkce

#include<intrins.h> // pro práci s plovoucí radkou

// definice konstant...
#define DEFAULTOFFSET0 (-0.7382) // přednastavený koeficient offsetu (RTD)
#define DEFAULTSCALE0 178.1348 // přednastavený koeficient scale (RTD)
#define DEFAULTOFFSET1 0 // přednastavený koeficient offsetu (chip)
#define DEFAULTSCALE1 128.0 // přednastavený koeficient scale (chip)
// #define TRIMSPAN 2 // rozsah nastaveného offsetu pomocí trimru

// deklarace globálních proměnných...
float ADC0norm; // normalizovaný výsledek převodu ADC0
float ADC1norm; // normalizovaný výsledek převodu ADC1
float offset0; // koeficient offsetu (RTD)
float scale0; // koeficient scale (RTD)
float offset1; // koeficient offsetu (chip)
float scale1; // koeficient scale (chip)
float Rprevcal; // předchozí kalibrovaná hodnota odporu
float ADC0cal; // normalizovaný výsledek prim. převodníku ADC0
// float Rrtd, Trtd; // odpor/teplota senzoru
float ADC0temp, ADC1temp; // dočasné proměnné pro normalizovaný výsledek AD převodu
bit onchip; // 0 for RTD, 1 for on-chip sensor
bit newADCdata; // 1 pro nová data AD převodníku
unsigned char serial_in; // data vstupu na sériové lince

// lokální funkce obsažené v tomto souboru...
void ADC_isr (); // rutina pro obsluhu prerušení ADC
void UART_isr (); // rutina pro obsluhu prerušení UART
void Temperature (); // zpracování výsledné teploty
void SerialCommand (); // zpracování příkazu z kom. linky
void Calibrate (void); // uživatelská kalibrace
signed char Cal (float Rcal); // funkce pro hrubou kalibraci
// float TrimTemperature(); // rutina pro kalibraci pomocí trimru
void PutFlashFloat (unsigned int addr, float *floatptr); // EEPROM vložení
void GetFlashFloat (unsigned int addr, float *floatptr); // EEPROM načtení
void DACoutput (float temp, float tmin, float tmax); // obsluha DAC

// funkce pro linearizaci obsažené v dalším souboru (RTDmath.c) ....
float T_rtd (float r); // přechodová funkce senzoru RTD
float R_rtd (float t); // převrácená přechodová funkce senzoru RTD
float Tmin_rtd (); // minimální specifikovaná teplota
float Tmax_rtd (); // maximální specifikovaná teplota
float Rmin_rtd (); // minimální specifikovaný odpor
float Rmax_rtd (); // maximální specifikovaný odpor

//
// Hlavní program =====
void main (void) {

    PLLCON = 1; // nastavení hodin jádra procesoru (6.291456MHz)
    CFG845 = 0x01; // povolení interní paměti XRAM

    // přednastavení hodnot
    offset1 = DEFAULTOFFSET1;

```

```

scale1 = DEFAULTSCALE1;
onchip = 0;
serial_in = 0;
newADCdata = 0;
ADC0norm = 0;
ADC1norm = 0;
Rprevcal = 0;           // poc. hodnota kalibrovaného odporu(predchozi hodnota)

// nacteni calibracnich koeficientu z datove flash pameti
GetFlashFloat (0,&offset0); // nacteni koeficientu offset (adresa 0)
GetFlashFloat (1,&scale0);  // nacteni koeficientu scale (adresa 1)
if ( _chkfloat_(offset0)==4 || _chkfloat_(scale0)==4) {
    offset0 = DEFAULTOFFSET0; // pokud neexistuji...
    scale0 = DEFAULTSCALE0;   // ..pouzij prednastavene
}

// nastaveni budiciho proudoveho zdroje
ICON = 0x07;           // 400uA na vyvodu IEXC2 pin (P1.7) [pro buzeni ext. RTD]

// konfigurace UART
T3CON = 0x85;          // nastaveni casovace 3...
T3FD = 0x12;           // ... na 9600 baudu
SCON = 0x52;           // konfigurace UART (s pouzitim casovace 3)

// konfigurace obou AD prevodniku a start prubeznych prevodu
//ADC0CON1 = 0x24;      // unipolarne, s vyr. pameti, zesileni=7.8125 (span=320mV)
ADC0CON1 = 0x22;        // unipolarne, s vyr. pameti, zesileni=31.25 (span=80mV)
ADC0CON2 = 0x4C;        // externi referencni signal, senzor na AIN5&6
ADC1CON = 0x0E;         // interni referencni signal, "on-chip" teplotni senzor
SF = 0xFF;              // rychlost AD prevodu = 5.35Hz
ADCMODE = 0x33;         // prubezny prevod obou AD prevodniku, rezim "chop on"

// konfigurace DA prevodniku
DACCON = 0x07;          // rozsah 5V(AVDD), 12bit, pin 14

// povoleni preruseni, & hlavni programova smycka
IE = 0xD0;              // povoleni preruseni ADC a UART
while(1) {              // hlavni smycka
    if (newADCdata) Temperature(); // pri novem vysledku AD prevodu zobraz teplotu
    if (serial_in) SerialCommand(); // pri vstupu na seriove lince prejdi do menu
}
}

//
// -----
// Rutina pro obsluhu preruseni ADC
// Spoustena pri kazdem dokonceni AD prevodu

void ADC_isr () interrupt 6 {
    // pokud promenne ADCnorm pripraveny pro nova data..
    if (newADCdata==0) {
        // ..pak dej vysledek ADC do promennych ADCtemp
        ADC0temp = (ADC0H*0x10000+ADC0M*0x100+ADC0L)/16777216.0;
        ADC1temp = (ADC1H*0x10000+ADC1M*0x100+ADC1L)/8388608.0-1.0;
        // ..a urci prumernou hodnotu pomoci ADC dat ulozenych v promennych ADCnorm
        ADC0norm = (ADC0norm + ADC0temp)/2; //ADC0norm = 0.8*ADC0norm + 0.2*ADC0temp;
        ADC1norm = (ADC1norm + ADC1temp)/2; //ADC1norm = 0.8*ADC1norm + 0.2*ADC1temp;
    }

    // vynuluj bity pro povoleni dalsiho AD prevodu
    RDY0 = 0;           // primarni ADC
    RDY1 = 0;           // sekundarni ADC

    // indikace pritomnosti novych dat AD prevodniku
    newADCdata = 1;
}

//
// -----
// Rutina pro obsluhu preruseni UART

void UART_isr () interrupt 4 {
    if (RI){             // pokud prijmut byte..
        RI=0;            // ..pak vynuluj indikator (RI flag)
        serial_in = SBUF; // ..a vloz prijaty byte do promenne serial_in
    }
}

```

```

//
// Funkce pro zpracovani vysledne teploty
// Spoustena pri kazdem dokonceni AD prevodu. Urci hodnotu odporu,
// zavola linearizacni funkce a nakonec vysle vyslednou hodnotu
// teploty na vystup UART

void Temperature () {
    float Rrtd, Trtd;    // odpor a teplota senzoru RTD
    float Tchip;        // teplota "on-chip" senzoru

    // vyhodnoceni teploty RTD podle vysledku ADC (a kalib. koeficientu)
    Rrtd = ADC0norm*scale0+offset0;    // odpor RTD [ohm]
    Trtd = T_rtd(Rrtd);                // teplota RTD [C]

    // vyhodnoceni teploty interniho "on-chip" senzoru
    Tchip = ADC1norm*scale1+offset1;    // teplota "on-chip" senzoru [C]

    // indikace nacteni ADC dat z ADCnorm promennych, a povoleni zapisu novych dat
    newADCdata = 0;

    // vystup na DA prvodniku
    DACOutput(Trtd,Tmin_rtd(),Tmax_rtd());

    // vystup vysledku pres UART
    if (onchip) {
        printf ("Teplota cipu = ");
        printf ("%+8.2f C",Tchip);
    }
    else {
        printf ("Teplota RTD = ");
        printf ("%+8.2f C",Trtd);
        // a pripadna indikace chyb
        if (Trtd<Tmin_rtd()) printf (" *ERROR: Teplota mensi nez povolena dolni mez!");
        if (Trtd>Tmax_rtd()) printf (" *ERROR: Teplota vetsi nez povolena horni mez!");
    }
    printf ("\r\n");

    P3 ^= 0x10;    // indikace pomoci LED (P3.4 na vyvojovem kitu)
}

//
// Funkce pro zpracovani prikazu komunikacni linky

void SerialCommand () {

    // zobrazeni hlavniho menu
    printf("\r\n");
    printf("-----\r\n");
    printf("ADuC845 PROGRAM - mereni teploty pomoci RTD senzoru \r\n");
    printf("-----\r\n");
    printf("HLAVNI MENU...\r\n");
    printf(" E: vybrat externi RTD senzor\r\n");
    printf(" I: vybrat interni senzor na cipu\r\n");
    printf(" K: kalibrace\r\n");
    printf(" ostatni klavesy: zpet\r\n");
    printf("Vyber moznost: ");

    // pri stisku klavesy provedeni patricne operace
    IE &= 0xEF;    // nejdriv zakaz serioveho preruseni
    switch (_getkey()) {
        case 'i' :
            case 'I' : printf ("I");
                        onchip = 1;
                        break;

            case 'e' :
            case 'E' : printf ("E");
                        onchip = 0;
                        break;

            case 'k' :
            case 'K' : printf ("K");
                        Calibrate();
                        break;

            default : break;
    }
    IE |= 0x10;    // povoleni serioveho preruseni
    printf("\r\n\r\n");
}

```

```

    // vynulovani serial_in pro povoleni dalsiho preruseni
    serial_in = 0;
}

// _____
// Uzivatelska kalibrace

void Calibrate () {
    signed char result; // indikator vysledku kalibrace
    float Rcal, Tcal;    // kalibracni odpor a teplota
    float Rrtd, Trtd;    // aktualni odpor a teplota RTD

    // urceni teploty RTD podle vysledku ADC a aktualnich kalibracnich koeficientu
    ADC0cal = ADC0norm; // aktualni hodnota ADC0
    Rrtd = ADC0cal*scale0+offset0; // aktualni odpor RTD [ohms]
    Trtd = T_rtd(Rrtd); // aktualni teplota RTD [C]

    // zobrazeni kalibracniho menu
    printf("\r\n\r\n");
    printf("Aktualni hodnota teploty a odporu: %+8.2fC, %+8.2fohm\r\n", Trtd, Rrtd);
    if (Rprevcal==0) printf(">1.Kalibracni bod<\r\n"); // indikace kalibracnich bodu
    else printf(">2.Kalibracni bod<");
    printf("\r\n\r\n");
    printf("KALIBRACNI MENU....\r\n");
    printf(" T: zadat aktualni znamou hodnotu teploty RTD\r\n");
    printf(" O: zadat aktualni znamou hodnotu odporu RTD\r\n");
    //printf(" S: kalibrace pomoci trimru\r\n");
    printf(" P: obnovit prednastavene hodnoty kalibracnich koeficientu\r\n");
    printf(" jina klavesa: preruseni a navrat\r\n");
    printf("Vyber moznost: ");

    // pri stisku klavesnice provede patricnou operaci
    IE &= 0xEF; // // nejdriv zakaz serioveho preruseni
    switch (getkey()) {
        case 't' :
            case 'T' : printf ("T\r\n");
                        printf ("zadej teplotu senzoru[C]: ");
                        scanf ("%f",&Tcal);
                        Rcal=R_rtd(Tcal);
                        result=Cal(Rcal);
                        break;

            case 'o' :
            case 'O' : printf ("O\r\n");
                        printf ("zadej odpor senzoru [ohm]: ");
                        scanf ("%f",&Rcal);
                        Tcal=T_rtd(Rcal);
                        result=Cal(Rcal);
                        break;

/*      case 's' :
            case 'S' : printf ("S\r\n");
                        printf ("Nastav aktualni teplotu pomoci trimru:\r\n");
                        Tcal=TrimTemperature(); //Funkce pro kalibraci pomoci trimru
                        Rcal=R_rtd(Tcal);
                        result=Cal(Rcal);
                        break;*/

            case 'p' :
            case 'P' : printf ("P\r\n");
                        scale0 = DEFAULTSCALE0;
                        offset0 = DEFAULTOFFSET0;
                        Rprevcal = 0;
                        result = -1;
                        break;

            default : result = -2;
                        break;
    }
    IE |= 0x10; // // povoleni serioveho preruseni

    // pri novych kalibracnich koeficientech ulozi nove do flash pameti
    if (result==0||result==-1) {
        PutFlashFloat (0,&offset0); // koeficient offsetu do adresy 0
        PutFlashFloat (1,&scale0); // koeficient scale do adresy 1
    }

    // indikace vysledku
    switch (result) {
        case -2 :

```

```

        break;
    case -1 :
        printf("\r\n");
        //printf("VYSLEDEK: Puvodni hodnoty kalibracnich hodnot prenavstaveny.\n");
        printf("PRENAVSTAVENO: scale=%+4.4f ofset=%+4.4f ",scale0,offset0);
        break;
    case 0 :
        printf("\r\n");
        printf("VYSLEDEK: Kalibrovano pro %+8.4f C\n",Tcal);
        printf("          scale=%+4.4f ofset=%+4.4f ",scale0,offset0);
        break;
    case 1 :
        printf("\r\n");
        printf("VYSLEDEK: Mimo povoleny rozsah, kalibrace neprovedena!");
        break;
    case 2 :
        printf("\r\n");
        printf("VYSLEDEK: Konflikt s predchozimi kalibracnimi hodnotami, kalibrace
neprovedena!");
        break;
    default :
        printf("\r\n");
        printf("VYSLEDEK: Neznamy chyba kalibrace!");
        break;
}
printf("\r\n");
printf("..pokracovani stiskem libovolne klavesnice.. ");
IE &= 0xEF; // zakaz serioveho preruseni
_getkey(); // cekani na stisk klavesnice
IE |= 0xD0; // obnoveni serioveho preruseni
printf("\n");
}

//
// Funkce pro hrubou kalibraci
// vstup: Rcal = ocekavany odpor pri aktualni teplote
// vystup: Cal() = indikace chyby (viz nize)
// globalni promenne: scale0, offset0, Rprevcal, ADC0cal
// Vykona kalibraci pro aktualni data pri zachovani predchozich
// ("prevcal" globalni promenne).

#pragma disable // zakaz preruseni pri teto funkci

signed char Cal (float Rcal) {
    signed char returnerr; // indikace chyb (0 = bez chyb)
    float ADC0prevcal; // ADC vysledek pri predchozich bodech kalibrace

    returnerr=3; // indikace chyby programu

    // urceni hodnoty ADC pri predchozich kalibracnich bodech(z Rprevcal)
    ADC0prevcal = (Rprevcal-offset0)/scale0;

    // nejdrive kontrola zda je vstup mimo rozsah
    if (Rcal<Rmin_rtd() || Rcal>Rmax_rtd())
        returnerr=1; // vstup Rcal mimo rozsah
    else if (Rcal>Rprevcal && ADC0cal<=ADC0prevcal)
        returnerr=2; // konflikt s predchozi hodnotou
    else if (Rcal<=Rprevcal && ADC0cal>=ADC0prevcal)
        returnerr=2; // konflikt s predchozi hodnotou

    // pokud vse vporadku proved kalibraci
    else {
        // scale0 je vpodstate deltaR/deltaADC
        scale0=(Rcal-Rprevcal)/(ADC0cal-ADC0prevcal);
        // vyhodnoceni offsetu pomoci nove hodnoty scale0 a jednoho znamého bodu
        offset0=Rcal-ADC0cal*scale0;
        Rprevcal=Rcal; // uchovani Rprevcal pro dalsi bod
        returnerr=0; // indikace uspesne kalibrace
    }
    return (returnerr);
}

//
// Rutina pro kalibraci pomoci trimru na desce RIOB1
// Nastavovanim trimru se meni teplota v rozsahu +-TRIMSPAN,
// ukonceni kalibrace stiskem libovolne klavesy
// Vystup: aktualni nastavena teplota RTD

```

```

/*
float TrimTemperature() {
    //float Rrtd, Trtd;
    float Offset;
    IE = 0x00;          // zakaz preruseni
    ADC1CON = 0x2A; // sekundarni prevodnik: interni referencni signal 1.25V, AIN1-AIN2, uni
    RI=0;                // nulovani bitu RI
    while(!RI){          // dokud neni stisknuta klavesa
        while(!RDY0&!RDY1) {} // cekani na ukonceni prevodu
        ADC0norm = (ADC0H*0x10000+ADC0M*0x100+ADC0L)/16777216.0; //vypocet normalizovanych
hodnot
        ADC1norm = (ADC1H*0x10000+ADC1M*0x100+ADC1L)/16777216.0;
        Offset = (ADC1norm*1.25)*TRIMSPAN-(TRIMSPAN/2); //nastaveni mezi ofsetu podle
polohy trimru
        // vyhodnoceni teploty RTD podle vysledku ADC (a kalib. koeficientu)
        Rrtd = ADC0norm*scale0+Offset; // odpor RTD [ohm]
        Trtd = T_rtd(Rrtd); // vypocet teploty RTD [C]
        printf("Aktualni teplota RTD= %8.2f C",Trtd); //vystup na terminal
        printf("\r\n");
        RDY0 = 0; // nulovani bitu primarniho ADC
        RDY1 = 0; // nulovani bitu sekundarniho ADC
    }
    ADC0cal = ADC0norm;
    ADC1CON = 0x0E; // Obnoveni konfigurace sek. prevodniku:interni referencni signal
1.25V,on-chip tepl. senzo
    IE = 0xD0; // Povoleni preruseni UART a ADC
    return(Trtd);
}
*/

//
// -----
// Ulozeni hodnoty s pohyblivou radovou carkou do flash pameti
// vystup: addr = adresa stranky flash pameti (0 az 0x3FF)
// vstup: *floatptr = ukazatele na hodnotu s pohyblivou radovou carkou
// Ulozi hodnotu do 4-bytove stranky interni flash data pameti
void PutFlashFloat (unsigned int addr, float *floatptr) {
    // vloz 4 byty do EDATA SFR bytu
    EDATA1 = *((unsigned char *)floatptr);
    EDATA2 = *((unsigned char *)floatptr+1);
    EDATA3 = *((unsigned char *)floatptr+2);
    EDATA4 = *((unsigned char *)floatptr+3);
    // uloz EDATA SFR byty do 4-bytove flash data stranky na adrese addr
    EADRH = *((unsigned char *)&addr); // horni byte
    EADRL = *((unsigned char *)&addr+1); // dolni byte
    ECON = 5; // vymazani stranky
    ECON = 2; // programovani stranky
}

//
// -----
// Nacteni hodnoty s pohyblivou radovou carkou z flash pameti
// vstup: addr = adresa stranky flash pameti (0 to 0x3FF)
// vystup: *floatptr = ukazatele na hodnotu
void GetFlashFloat (unsigned int addr, float *floatptr) {
    // cteni EDATA SFR bytu ze 4-bytove flash datove stranky na adrese addr
    EADRH = *((unsigned char *)&addr); // horni byte
    EADRL = *((unsigned char *)&addr+1); // dolni byte
    ECON = 1; // cteni stranky
    // vlozeni EDATA SFR bytu do 4 bytu of the promenne float
    *((unsigned char *)floatptr) = EDATA1;
    *((unsigned char *)floatptr+1) = EDATA2;
    *((unsigned char *)floatptr+2) = EDATA3;
    *((unsigned char *)floatptr+3) = EDATA4;
}

//
// -----
// Vypocet vystupniho napeti 12-bit DA prevodniku podle teploty
// Vstup: teplota v rozsahu (tmin-tmax)
// Vystup: napeti na DAC 0..5V(AVDD)

void DACoutput (float temp, float tmin,float tmax) {
    float DAC; // hodnota DAC 0-4096 (FFF)

    DAC=(temp-tmin)*(4096/(tmax-tmin)); // vypocet hodnoty DAC
    DACH=DAC/0x100; //horni pulbyte DAC
    DACL=DAC/0x01; //dolni byte DAC
}

```

```

//*****
// Soubor      : RTDmath.c
// Autor       : AD,preklad a uprava Robert Malik
// Datum       : 18 dubna 2005
// Kompiler    : Keil C51
// Popis       : Funkce pro linearizaci signalu odporovych
//               senzoru teploty (RTD) pouzivajici primou
//               matematickou metodu.
//*****

// použijte knihovny....
#include<math.h> // matematicke funkce

// definice konstant....
#define AA (3.9083E-3) // A konstanta prechodove funkce
#define BB (-5.775E-7) // B konstanta prechodove funkce
#define CC (-4.183E-12) // C konstanta prechodove funkce
#define Z1 (-3.9083E-3) // Z1 koeficient inverzni prech. funkce pro kladne teploty
#define Z2 (17.58480889E-6) // Z2 koeficient inverzni prech. funkce pro kladne teploty
#define Z3 (-23.10E-9) // Z3 koeficient inverzni prech. funkce pro kladne teploty
#define Z4 (-1.155E-6) // Z4 koeficient inverzni prech. funkce pro kladne teploty
#define TMIN (-30) // dolni mez teploty (minimalni) [C]
#define TMAX (180) // horni mez teploty (maximalni) [C]
#define RMIN 88.22166 // dolni mez odporu (minimalni) [ohm]
#define RMAX 168.4783 // horni mez odporu (maximalni) [ohm]

// -----
// Teplota podle funkce RTD
// vstup: r = odpor RTD
// vystup: T_rtd() = odpovidajici teplota RTD
// Vyhodnoti teplotu podle odporu pouzitim prime matematicke metody.

float T_rtd (float r) {
    float t;

    // overeni zda je vstupni hodnota odporu ve specifikovanych mezich
    if (r<RMIN) // pokud r < dolni mez..
        t = TMIN; // ..nastavi t na dolni mez
    else if (r>RMAX) // pokud r > horni mez
        t = TMAX; // ..nastavi t na horni mez

    // pokud vstup (r) v definovanem rozsahu, pak resi vystup.
    else {

        // pokud r < práh, pak pouzij prechodovou funkci pro zapornou teplotu (aproximace 5.-
        2. radu)
        if (r<100) t=-242.0199+2.222812*r+2.585885E-3*pow(r,2)-4.826040E-6*pow(r,3)-
        2.818340E-8*pow(r,4)+1.524259E-10*pow(r,5);
        // if (r<96.6) t=-241.9610+2.216253*r+2.854064E-3*pow(r,2)-9.912120E-
        6*pow(r,3)+1.705183E-8*pow(r,4);
        // if (r<95.1) t=-242.0906+2.227625*r+2.517790E-3*pow(r,2)-5.861951E-6*pow(r,3);
        // if (r<72.1) t=-242.9703+2.283841*r+1.472734E-3*pow(r,2);

        // pokud r >= práh, pouzij prechodovou funkci pro kladnou teplotu
        else t=(Z1+sqrt(Z2+Z3*r))/Z4;

    }
    return (t);
}

// -----
// Odpor podle teploty
// vstup: t = teplota RTD
// vystup: R_rtd() = odpovidajici odpor RTD
// Vypocita odpor RTD podle teploty primou matematickou metodou

float R_rtd (float t) {
    float r;

    if (t>=0) r = 100 * (1 + AA*t + BB*pow(t,2));
    else r = 100 * (1 + AA*t + BB*pow(t,2) + CC*(t-100)*pow(t,3));

    return (r);
}

```

```

// _____
// Funkce minimalni teploty
// Vraci minimalni teplotu specifikovanou ve vyhledavaci tabulce.
float Tmin_rtd () {
    return (TMIN);
}

// _____
// Funkce maximalni teploty
// Vraci maximalni teplotu specifikovanou ve vyhledavaci tabulce.
float Tmax_rtd () {
    return (TMAX);
}

// _____
// Funkce mainimalni hodnoty odporu
// Vraci minimalni hodnoru odporu specifikovanou ve vyhledavaci tabulce.
float Rmin_rtd () {
    return (RMIN);
}

// _____
// Funkce maximalni hodnoty odporu
// Vraci maximalni hodnoru odporu specifikovanou ve vyhledavaci tabulce.
float Rmax_rtd () {
    return (RMAX);
}

```