VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

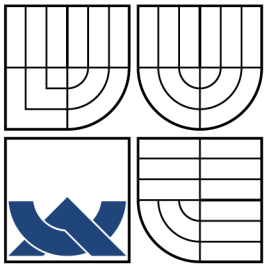
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

VÝUKOVÝ VÝVOJOVÝ KIT S PROCESOREM FREESCALE S08

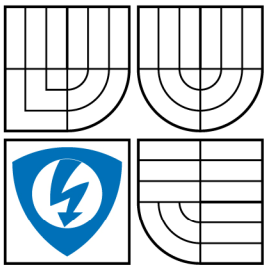
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE JAN HAVLÍK
AUTHOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

VÝUKOVÝ VÝVOJOVÝ KIT S PROCESOREM FREESCALE S08

EDUCATIONAL DEVELOPMENT KIT BASED ON FREESCALE S08 PROCESSOR

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — JAN HAVLÍK
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ ŠEBESTA, Ph.D.
SUPERVISOR

BRNO 2009

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení:	Jan Havlík
Bytem:	Vančurova 18, Brno, 615 00
Narozen/a (datum a místo):	29. prosince 1985 v Brně

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 53, Brno, 602 00
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Dr. Ing. Zbyněk Raida, předseda rady oboru Elektronika a sdělovací technika
(dále jen „nabyvatel“)

Čl. 1

Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):

   - ☐ disertační práce
   - ☐ diplomová práce
   - ☒ bakalářská práce
   - ☐ jiná práce, jejíž druh je specifikován jako .....
     (dále jen VŠKP nebo dílo)

Název VŠKP:	Výukový vývojový kit s procesorem Freescale S08
Vedoucí/ školitel VŠKP:	Ing. Jiří Šebesta, Ph.D.
Ústav:	Ústav radioelektroniky
Datum obhajoby VŠKP:	_____

   VŠKP odevzdal autor nabyvateli*:

   - ☒ v tištěné formě – počet exemplářů: 2
   - ☒ v elektronické formě – počet exemplářů: 2

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.

3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.

4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.

2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.

3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti

   - ☒ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy (z důvodu utajení v něm obsažených informací)

4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.

2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou, se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.

3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.

4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: 6. června 2008

…….. ……

Anotace

Práce je věnována architektuře a využití procesorů Freescale řady S08, dále je věnována návrhu vývojového výukového kitu s procesorem MC9S08JM60 se zaměřením na aplikace v elektronice a sdělovací technice. Je sestaven soubor několika ukázkových aplikací pro ověření funkce ve vývojovém prostředí CodeWarrior.

Annotation

The work is dedicated to architecture and usage processors Freescale series S08, further proposal evolutionary tutorial kitu with processor MC9S08JM60 with a view to application in electronics and communication engineering. There is compiled set of several speciment application for check function in evolutionary environment CodeWarrior.

Klíčová slova

MCU, S08, bdm, Flexis, ColdFire, run mode, standby mode, sleep mode, Freescale, CodeWarrior, Processor Expert,

Keywords

MCU, S08, bdm, Flexis, ColdFire, standby mode, run mode, Freescale, CodeWarrior, Processor Expert,

HAVLÍK, J. *Výukový vývojový kit s procesorem Freescale S08: bakalářská práce*. Brno: FEKT VUT v Brně, 2009. 26s., 2 příl.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Výukový vývojový kit s procesorem Freescale S08 jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

V Brně dne 6. června 2008 .....
podpis autora

Poděkování

Děkuji veoucímu bakalářské práce Ing. Jiřímu Šebestovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 5 června 2009 .....
podpis autora

Obsah

1 Úvod

Mikrokontrolér, jinak nazývaný jednočipový monolit, je speciální druh mikroprocesoru využívaný pro specifické aplikace. V současné době se vyskytuje téměř ve všech elektrických zařízeních v oblasti průmyslu, lékařství, spotřební elektroniky i jinde. Mikrokontroléry v elektrických zařízeních mají řadu výhod oproti starým technologiím. Mezi základní výhody použití mikrokontroléru patří snížení počtu elektrických prvků a tím snížení nákladů na výrobu plošných spojů. Další výhodou je snížení napěťové náročnosti. Některé mikrokontroléry pracují od napětí 1,8V. Funkce navrženého systému lze změnit pouhým přeprogramováním bez nutnosti zásahu do hardwaru.

Nyní existuje velké množství výrobců mikrokontrolerů různých řad. Mezi nejznámější patří Intel, Atmel, Freescale. V mé práci budou rozebrány a popsány mikrokontroléry od výrobce Freescale řady S08. Každý výrobce mikrokontrolérů má vlastní vývojové prostředí pro tvorbu aplikací. Při návrhu aplikace s mikrokonroléry Atmel se využívá vývojové prostředí Avr, při práci s mikrokontroléry Freescale zase CodeWarrior, nebo FreeMaster.

Mikrokontroléry všech výrobců se vyrábí v „rodinách“ (v angličtině Family), jejichž členy mají shodné či podobné základní vlastnosti (princip funkce, napájecí napětí, periferie a pod.). V této práci se zaměřuji na MCU řady S08 rodiny JM od výrobce Freescale. Je navržen vývojový kit s tímto MCU, který má velké množství využití v elektronice a sdělovací technice díky svým periferiím.

V závěru práce je vytvořen soubor ukázkových aplikací vytvořených ve vývojovém prostředí CodeWarrior .

2 Mikrokontrolér

Mikrokontrolér (MCU- Microcomputer Unit), je jinak též označovaný jako jednočipový mikropočítač. Tato programovatelná součástka je určena pro nejrůznější kontrolní a řídící úlohy v průmyslových oborech, pro realizaci meřících a řídicích systémů atd. Díky své univerzálnosti, malé velikosti, nízké ceně a spotřebě nachází své uplatnění ve velkém množství aplikací. Lze ji nalézt v mnoha současných elektronických zařízeních.

Charakteristika

Procesorové jádro
Paměť RAM
Paměť s programem - ve formě ROM, EPROM (dnes už zastaralé) a nebo FLASH
Časovače
Vstupní - výstupní zařízení (I/O kontroléry):

- kontrolér sériové komunikace (např. UART (RS232), I2C, SPI)
- A/D a D/A převodníky
- Vstupně/výstupní (I/O) porty (piny) pro všeobecné použití (obvykle kompatibilní s úrovní TTL)

MCU k základním funkcím potřebuje skutečně málo, obvykle jen napájení a zdroj hodinového signálu, který je realizován vnitřním oscilátorem s připojeným vnějším krystalem. Pro správnou funkci mikrokontroléru je vhodné připojit i resetovací obvod, i když mnohé moderní mikrokontroléry ho mají už integrovaný.

Mikrokontroléry se vyznačují nízkými náklady, nízkou spotřebou a dostatečnou hardwarovou výbavou pro řízení aplikací. Základní dělení mikrokontrolérů je podle šířky registrů a sběrnice (8 bitové, 16 bitové, 32 bitové). Vyrábí se v širokém sortimentu výkonů a velikostí. Nejmenší typy mají jen 8 vývodů včetně napájení a rozměry pouzdra přibližně 3x3 mm, nejvýkonnější typy mají pouzdra se 100 – 200 vývody.

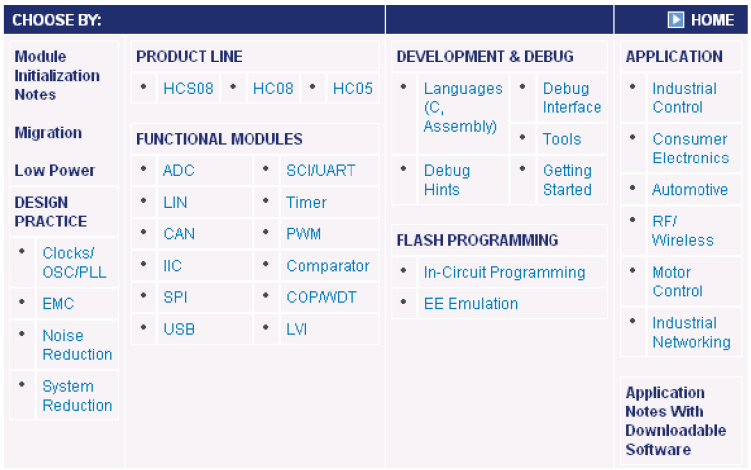
2.1 Typy architektur v mikroprocesorové technice

a) Z hlediska organizace paměťových adresních prostorů lze mikrokontroléry rozdělit na dvě skupiny. Mikrokontroléry s lineárním uspořádáním adresního prostoru (Von Neumannova architektura). Pro tuto architekturu je charakteristické použití jediného adresního prostoru. V něm je mapovaná paměť programu, paměť dat i registry pro řízení IO obvodů. Tuto architekturu používají mikrokontroléry Freescale. Intel 80196, Hitachi a řada dalších.

b) Mikrokontroléry s odděleným adresním prostorem pro paměť programu a paměť dat. Tato koncepce je označována jako Harwardská koncepce. Harwardskou architekturu používají Mikrokontroléry Intel 8051, 8052 a typy z nich odvozené, vyráběné celou řadou výrobců, různé řady mikrokontrolérů PIC firmy Microchip, mikrokontroléry AVR firmy Atmel a další.

3 Popis mikrokontrolérů Freescale řady S08

Do skupiny 8 bitových procesorů firmy Freescale s jádrem S08 patří S08AC, S08D, S08EL, S08GB, S08GT, S08JM, S08JS, S08LC, S08LG, S08LL, S08QA, S08QB, S08QD, SO8QE, S08QG, S08R, S08SE, S08SH, S08SL. Každý typ mikrokontroléru obsahuje určité periférie. Např. mikrokontrolér S08LC obsahuje podporu LCD displejů, S08JM zase USB. Všechny typy se vyznačují vysokým výkonem a nízkým napájecím napětím od 1,8V. Velkou snahou výrobce je snižování spotřeby energie mikrokontrolérů.



Obr. 1

Často vyskytované periférie u mikrokontrolérů Freescale

ADC	A/D převodník
CAN	Datová sběrnice místní sítě (Controller Area Network)
IIC	Sériová sběrnice
SPI	Sériové komunikační rozhraní
USB	Standardní sériové rozhraní
SCI	Asynchronní komunikační rozhraní
UART	Univerzální asynchronní přijímač/vysílač
Timer	Časovač
PWM	Pulsní šířková modulace
Comparator	Analogový komparátor
WDT	Wireless

3.1 Jádro (Core)

HC08 Core

Jednočipové mikrokontroléry Freescale řady 68HC08 jsou typickými představiteli své kategorie, navazujícími na řadu HC05, se kterou jsou kompatibilní na úrovni zdrojového kódu (směrem vzhůru). Obě tyto řady jsou vyráběny technologií HCMOS, starší typy 6805 byly vyráběny technologií HMOS. Struktura jádra vychází z historického mikroprocesoru 6800 a staršího jednočipového mikropočítače 6801. Jádro řadiče 68HC08 (a 68HC05) bylo proti mikropočítači 6801 poněkud zjednodušeno (chybí druhý střadač), na druhé straně bylo doplněno o instrukce pro práci s bity v paměti RAM a v registrech periferií. Procesor je doplňován periferiemi tak, aby co nejlépe vyhovoval požadavkům konkrétní aplikace. Vybavenost periferiemi a velikost hlavní programové paměti se projeví i ve značení příslušného mikrokontroléru [3].

Význam symbolů v označení mikrokontrolerů řady HC08

MC 68HC 908 GP 32 CFB

MC	Označení výrobce (FREESCALE);
68HC	Označení základní technologie;
908	Typ paměti 908 = FLASH, 708 = EPROM, nebo OTP, 08 jen ROM;
GP	„rodina“ proces7orů (GP = General Purpose, pro obecné použití), právě tato písmena označují vybavení periferiemi. Příklady dalších dvojic písmen: AZ, BD, EY, GP, GR, GT, GZ, JB, JL, KH, KX, LJ, MR, RK, SR.
32	Přibližná velikost hlavní paměti v Kb
CFB	Typ pouzdra

HCS08 Core

Čipy MC9S08QG jsou vyráběny moderním výrobním procesem na bázi technologie 0,25 um. Výsledkem procesu je, že tyto mikrořadiče mohou být taktovány frekvencí až 20 MHz (10MHz sběrnice) při napájení >2,1 V a 16 MHz (8MHz sběrnice) při napájení <2,1 V, což zajišťuje i při nízkém napájecím napětí dostatečně vysoký procesní výkon. Jádro HCS08 používá obdobnou instrukční sadu jako dřívější jádro HC08, s přidanou podporou pro vstup do ladicího režimu (instrukce BGND). Nová instrukce BGND umožní uživateli plynulý přechod z módu běhu aplikace do módů ladění s využitím modulů Background Debug Controller (DBG) a In-Circuit Emulation (ICE) integrovaných na čipu mikrořadiče.

Modul Background Debug Controller (BDG) obsahuje sadu instrukcí, které podporují čtení a zápis paměťových oblastí bez přerušení CPU. Modul In-Circuit Emulation (ICE) má výkonné krokovací a trasovací jednotky, které umožňují komfortní ladění a monitorování chodu programu. Ladící cyklus se zde uskutečňuje prostřednictvím komunikačního rozhraní, které využívá jediný pin procesoru BKGD. Po ukončení ladícího cyklu může být tento pin využit uživatelskými aplikacemi jako další I/O. Výše uvedené moduly, spolu s jednoduchým ladícím rozhraním, dovolují profesionální ladění včetně ukládání a následné analýzy všech změn běhu programu bez nutnosti použití externích emulátorů, či jiných monitorovacích programů, a při zachování maximálního počtu pinů I/O.

V oblasti bateriemi napájených aplikací je klíčovým požadavkem minimalizace spotřeby energie mikrořadiče při současném zachování výpočetního výkonu potřebného pro aplikaci. Aby bylo možno splnit uvedené požadavky na mikrořadičích MC9S08QG, byly tyto čipy vybaveny úspornými režimy – jedním „Wait“ a třemi „Stop“. Přechod do těchto úsporných režimů je umožněn instrukční sadou mikrořadiče (instrukce WAIT a STOP).

Vstupem do režimu „Wait“ dojde k povolení všech přerušení a k následnému zablokování hodin procesorového jádra. Všechny periferie jsou v tomto režimu aktivní, a mohou tedy generovat přerušení, která uvedou procesorové jádro zpět do módu běhu. Zadáním jakéhokoli režimu „Stop“ dojde k zablokování hodin centrální sběrnice, což způsobí zastavení nejen procesorového jádra, ale i všech periferií, které jsou z ní taktovány. Z režimu „Stop“ je možné přejít zpět do módu běhu pouze přerušením, které je aktivováno periferiemi s vlastními hodinami (ADC, RTI) nebo externími signály (KBI/IRQ). Režimy nízké energetické náročnosti „Wait“ a „Stop“ budou dále souhrnně označovány jako režimy „úspory energie" [3].

Význam symbolů v označení mikrokontrolerů řady HCS08

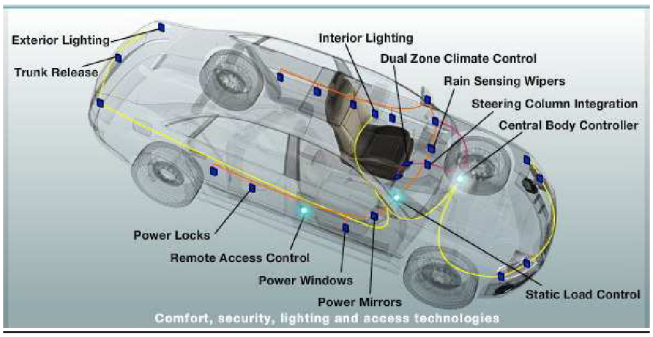
MC 9 S08 GB 60 C LDE

MC	Označení výrobce (FREESCALE);
9	Typ paměti 9 = FLASH, 8 = ROM
S08	Typ jádra 08 – HC08, S08 – HCS08, RS08 – RS08
GB	„rodina“ procesorů (GP = General Purpose, pro obecné použití)
60	Přibližná velikost hlavní paměti v Kb
C	Teplotní rozsah (C = –40°C to 85°C)
LDE	Typ pouzdra

4 Využití procesorů Freescale S08

V současné době se mikrokontroléry vyskytují skoro v každém elektronickém zařízení. Využitelnost závisí na jejich nízkých nákladech a velké hardwarové výbavě. Použití procesorů v obvodech zmenší počet součástek, a tím i náklady na výrobu plošných spojů. Dále zmenší napěťové nároky na použité obvody. Velkou výhodou mikrokontrolérů je snadnost vývoje a testování aplikací. Změnit funkci lze pouhým přeprogramováním ve vývojovém prostředí (CodeWarrior). Freescale Semiconductor má více než 50ti letou historii v mikroelektronice. Produkci tvoří polovodiče pro automobilový průmysl, dále polovodiče pro spotřební průmysl, počítačové sítě a bezdrátové komunikace [2].

Oblasti použití:



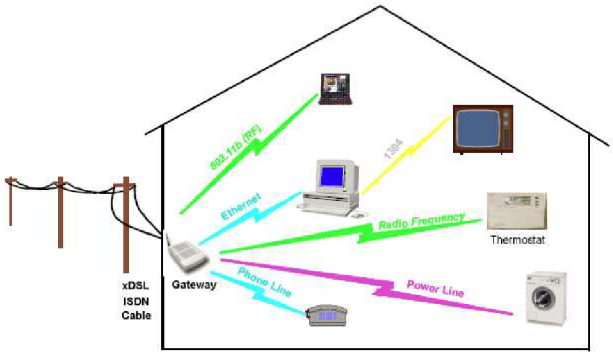
Obr. 2

<u>Automobilový průmysl</u>
Podvozek/bezpečí
Řídící informační systémy/zábava
Vnitřní připojení do sítě
Řídící jednotka
Mobil GT™

<u>Uživatelské použití</u>
Domácí spotřebiče
Domácí síťové systémy

<u>Průmysl</u>
Stavební řízení
Osvětlení
Výroba
Lékařství
Řízení motoru
Pokladní systém
Bezpečnost
Síťové aplikace

<u>Počítačové sítě</u>
Domov & SOHO připojení do sítě
Firemní síť
Bezdrátová infrastruktura
Kabelová infrastruktura
Pokročilá architektura počítačových telekomunikací

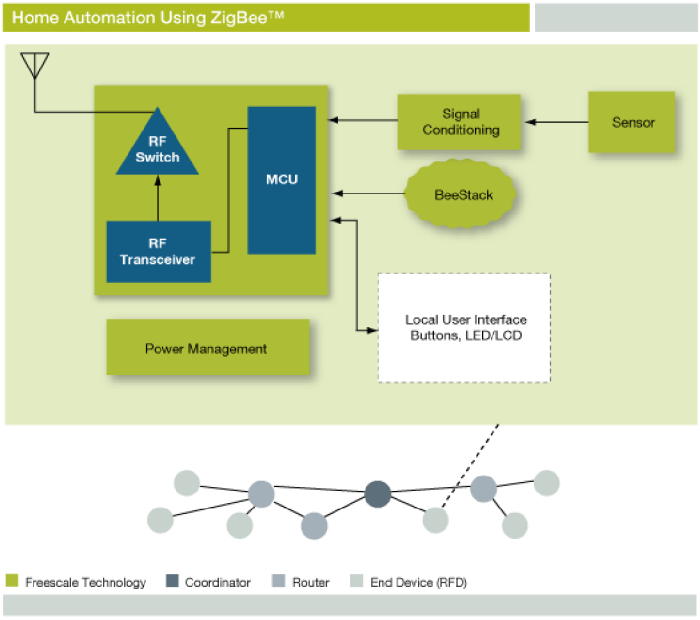


Obr. 3

<u>Bezdrátová a mobilní komunikace</u>
Mobilní telefony
Přenosní multimediální přehrávače
Přenosná navigační zařízení
Domácí zábava
Bezdrátové aplikační infrastruktury

4.1 Automatizace domácnosti

Radiová komunikace je velmi cenné řešení pro domácí automatizaci prostřednictvím bezdrátové sítě. Automatizace domácnosti bude znamenat neustálé přidávání většího množství zařízení, která budou snímat senzory (senzory pokojového obsazení, snímač vodní hladiny, teploměry, dveřní senzory). Radiová komunikace je nejvhodnější pro detekování senzorů rozmístěných po celém, domě. Tyto a další úkoly mohou být dosaženy s tímto chytrým mikrokontrolérem MC9C08QD4 včetně dobře navrženého softwaru. Mcu je v 8pinovém pouzdře s implementovaným vysílačem (WDT), pracujícím na kmitočtu 2,4GHz



Obr. 4

Senzor	např. teploměr
Signal Conditioning	Úprava signálu z analogového na digitální (A/D převodník)
BeeStack	Software, který poskytuje možnost spolehlivé komunikace s bezdrátovou sítí .
Local User Interface Buttons, LED/LCD	Lokální uživatelské rozhraní (tlačítka, diody, LCD, potenciometry)
Power Management	Řízení výkonu
MCU	Mikrokontrolér
RF Transceiver	Vysílač s příjímačem (MC13191FC) *f = 2,4 – 2,5GHz*

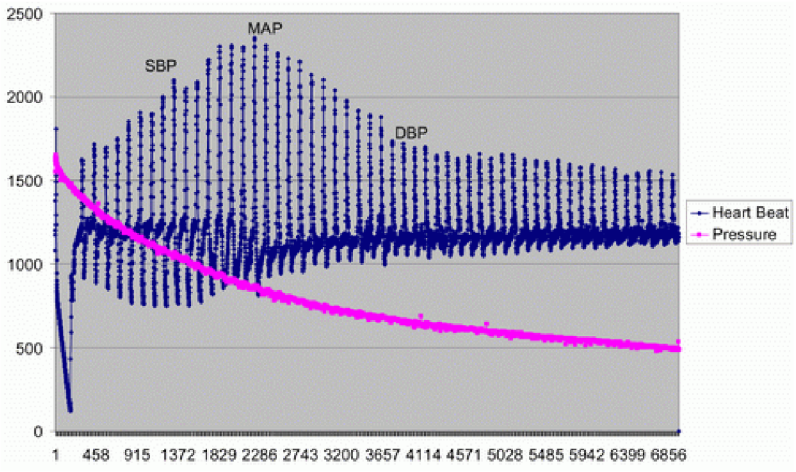
4.2 Monitor krevního tlaku

V minulosti bylo možné změření tlaku jen při návštěvě obvodního lékaře, nebo nemocnice. Vývoj techniky počítá s automatizovanými domácími monitory krevního tlaku, umožňujícími pacientovi měření tlaku dle jeho uvážení. Běžní uživatelé používají měření jednou denně. Pacienti s vyššími riziky zvýšeného tlaku používají měření několikrát denně.

Jak se elektronicky měří krevní tlak?

I když na první pohled vypadá pro technika, nezasvěceného do problému, návrh a realizace elektronického měřiče krevního tlaku možná jako těžký úkol, tak při alespoň základním pochopení principu měření krevního tlaku to už jako nepřekonatelný problém nevypadá. Prakticky nejde o nic jiného než elektronicky zrealizovat postup, který dělá každá zdravotní sestra, která pacientovi měří tlak. Zatímco ona využívá pro zjištění závislosti změny tepu srdce na postupně klesající úrovni tlaku v manžetě (měřícím rukávu) svoje oči a sluch, elektronika si musí vystačit pouze s měřením změny tlaku, který však dokáže, na rozdíl od sestry, měřit s velkou přesností a postřehnout změny, které by zdravotní sestra, sledující pouze stupnici tlakoměru, vůbec nezaznamenala. Jaký měřící proces se tedy skrývá pod označením "měření krevního tlaku" ?

Během vypouštění manžety (měřícího rukávu), upevněné na ruce měřené osoby (pacienta), je možné sledovat malé kolísání celkového tlaku v manžetě. Toto kolísání je dáno periodickými změnami tlaku uvnitř tepen cirkulací krve (pumpování srdce). Pokud se tato kolísavá složka (signál) zesílí a odfiltruje se od nízkofrekvenční a stejnosměrné složky (tzv. trend) průchodem filtru typu horní propust s mezní frekvencí 1 Hz, získá se nejen hledaný průběh změny/kolísání tlaku, ale zároveň jeho analýzou také záznam průběhu funkce srdce, tedy z periody signálu kolísání tlaku zjištěnou změnu frekvence tlukotu srdce (tepu) v závislosti na čase. Docela jednoduchou filtrací přesně změřeného průběhu tlaku v manžetě je tedy možné zjistit to, k čemu zdravotní sestra zároveň potřebuje tlakoměr a stetoskop. Na následujícím obrázku *Obr.5* je pak typická ukázka průběhu celkového měřeného tlaku v měřící manžetě v čase (růžová závislost "Pressure") a z něho získaná časová závislost průběhu tlukotu srdce (modrý průběh "Heart Beat") [3].



Obr. 5. Průběh krevního tlaku (růžový průběh "Pressure") a tepu srdce (modrý průběh "Heart Beat") v závislosti na čase a změny tlaku v měřící manžetě

Na modrém průběhu tepu srdce si lze všimnout třech vyznačených zkratek:

SBP (Systolic Blood Pressure) - systolický krevní tlak = měřená horní hranice tlaku
DBP (Diastolic Blood Pressure) - diastolický krevní tlak = měřená spodní hranice tlaku
MAP (Mean Arterial Pressure) - střední tlak artérie = zlomový bod, po jehož překročení se změní trend změny tepu.

Měření systolického a diastolického tlaku pomocí MCU HCS08 a ColdFire V1

V praxi se pro vyhodnocení signálu tlukotu srdce a zjištění systolického (SBP) a diastolického (DBP) krevního tlaku využívá oscilometrické metody. Tu lze realizovat dvěma způsoby v závislosti na složitosti měřiče a požadovaném komfortu pro pacienta:

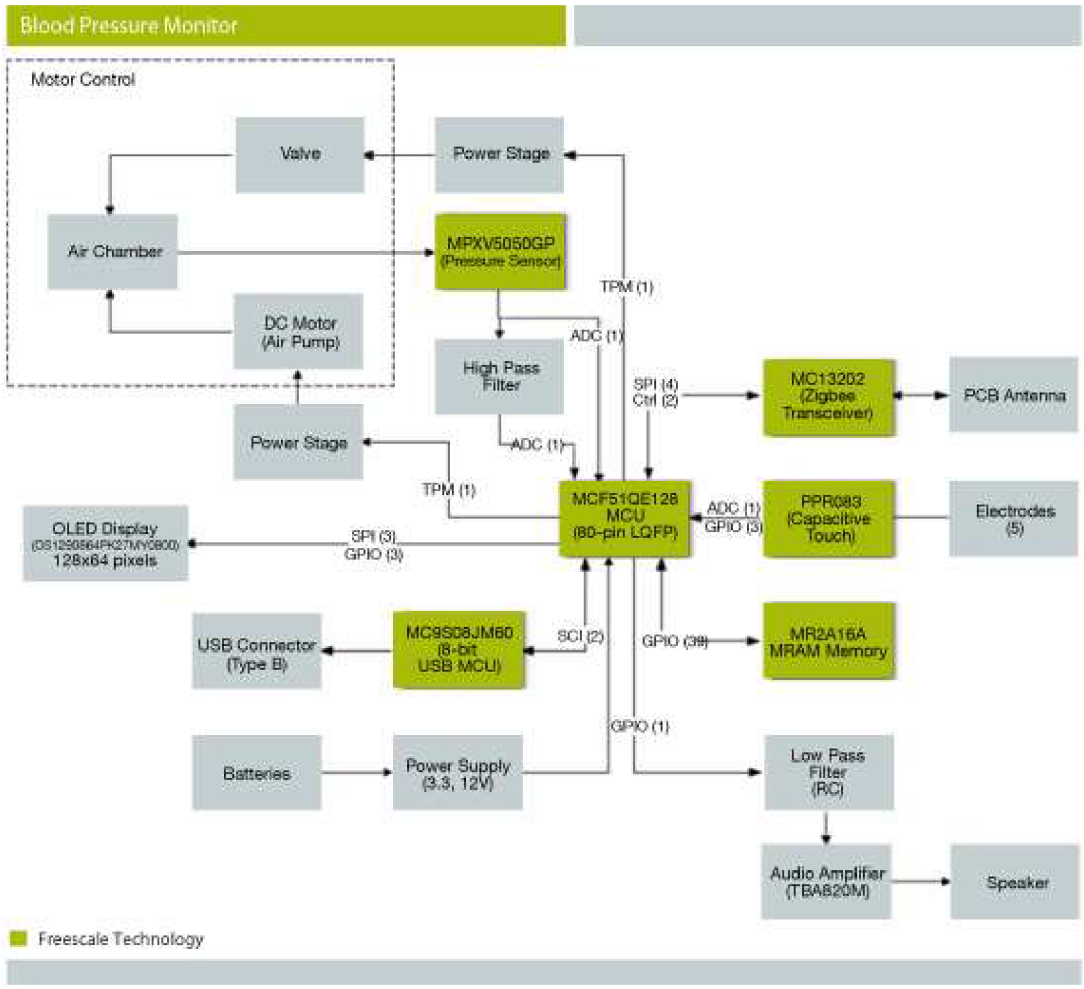
- metoda sestupné změny-jednodušší metoda měření, ale méně příjemná pro pacienta
- metoda náběžné změny-složitější metoda měření, která je však příjemnější pro pacienta

Realizace první jednodušší varianty je založena na poznatku, že amplituda signálu tlukotu srdce se mění s tím, jak se manžeta (rukáv) tlakuje přes bod SBP. Zatímco se rukáv vypouští, amplituda signálu roste až do doby překročení systolického tlaku pacienta. Jak se tlak v manžetě dále snižuje, amplituda tlukotu srdce se zvyšuje, až dosáhne maximálního pulsu známého jako MAP. Poté rychle klesá až do dosažení diastolického tlaku. Nevýhoda tohoto principu spočívá v tom, že manžeta na ruce pacienta musí být natlakována na úroveň vyšší, než je dopředu neznámý systolický tlak pacienta. Problém je tedy v tom, že dopředu nevíme, jak moc je nutné manžetu natlakovat, a je tedy nutné manžetu přefouknout, aby bylo zajištěno, že opravdu došlo k jeho překročení bodu SBP. Často je to až zbytečně moc, což je nepohodlné pro pacienta. Regulace upouštění tlaku pomocí vzduchového ventilu z manžety, i měření sestupného průběhu, je však pro elektronickou realizaci jednodušší a lze v tomto případě použít jen jednodušší MCU HCS08.

Při použití 32bitového MCU ColdFire V1 jako řídícího a výpočetního jádra systému je však možné nasadit druhou metodu, obrácenou verzi metody první, realizovanou pomocí reverzní oscilometrické metody. Problém je zde hlavně v tom, že se začíná měřit na velmi malém signálu, a navíc je nutné odfiltrovat šum přístroje, vznikajícího skrze motorky, provádějícími tlakování manžety.

Základní popis zapojení měřiče tlaku

Jeden z příkladů možné konstrukce elektrického zapojení měřiče krevního tlaku je uveden na blokovém schématu na *obr. 6*. Za základní prvky celého zapojení lze logicky považovat senzor tlaku, zde realizovaný univerzálním integrovaným senzorem tlaku Freescale MPXV5050GP a řídícím a vyhodnocovacím procesorem (mikrokontrolérem). Jak již bylo uvedeno výše, v případě jednoduššího principu měření (metoda sestupné změny) stačí použít 8bitové MCU, zde HCS08, zatímco u výpočetně náročnějšího reverzního principu (metoda náběžné změny), kdy se měří průběh při rostoucím tlaku v manžetě, je nutné použít již 32bitové MCU, například základní MCU ColdFire s jádrem V1. Oba obvody jsou z výrobního programu Freescalu. Protože se předpokládá napájení z baterie či akumulátoru, byla zvolena nízkopříkonová verze MCU Flexis QE128 (MCU MC9S08QE128), která se vyznačuje nejen sníženou spotřebou v plném běhu, ale i 3 stupni sníženého příkonu s různým omezením napájení a provozu různých částí chipu dle aktuální potřeby běhu aplikace/programu. Stupně jsou realizované módy, spuštěné softwarově. Mimo tyto dvě součástky je pak dále nepostradatelný automaticky regulovaný mechanismus vypouštění, případně i napouštění (natlakování) měřící manžety, a filtr typu horní propust, který by však v případě přesného mnohabitového A/D převodníku a výkonného DSP procesoru či MCU bylo patrně možné realizovat i softwarově (číslicový filtr). Ostatní obvody již více méně tvoří rozšiřující periferní části zařízení.



Obr. 6

5 Základní vlivy na spotřebu elektrických zařízení

Většina současných elektronických zařízení je složena z řídících integrovaných obvodů (různých programovatelných součástek) s běžícím softwarem, pamětí (RAM, EEPROM, Flash), motorů tvořících převodníky elektrické energie na mechanický pohyb, různých periferních obvodů (např. senzory, fyzická komunikační drátová i bezdrátová rozhraní, světelné a zvukové měniče) a samozřejmě zdrojů/měničů. Dalo by se tedy říct, že právě tyto komponenty určují elektrickou spotřebu celého zařízení. To je sice pravda, ale často je potíž vyhledat vhodnou aplikaci pro danou volbu a jejich vzájemném propojení konstruktérem/uživatelem. Totiž i jinak úsporné součástky a prvky mohou tuto svojí vlastnost rychle ztratit, pokud se jejich "vrozených" dispozic v konstrukci zařízení nevyužívá, jsou nevhodně propojeny nebo se prostě jen nehodí pro danou činnost (jsou přetíženy nebo nevytíženy) [3].

Co se tedy hlavně podílí na vysoké spotřebě el. energie moderních elektronických zařízení ?

- špatně zvolené a provozované mikroprocesory či hradlová pole
- špatně navržený a naprogramovaný řídící software
- nevhodně zvolené typy motorů a hlavně jejich řízení
- málo účinný nebo špatně sladěný standby režim

Standby režim zařízení

Aniž to musí být na první pohled patrné, celoevropsky i celosvětově dělá dost velkou část spotřeby el. energie napájení zařízení v tzv. standby módu, tedy v režimu, kdy zařízení sice není plně aktivní, ale je připraveno na rychlé zahájení své funkce (probuzení). V dnešní době již má většina integrovaných obvodů obsažený nějaký standby nebo sleep mód, ve kterém jsou sice součástky stále napájeny, ale mají potlačeny všechny funkce, nesouvisející se schopností se na určitý povel "probudit".

U neprogramovatelných hardwarových součástek to bývá téměř vždy pomocí přivedení daného logického stavu na k tomu určený vstup, u programovatelných součástek to buď bývá kombinace hardwarové a softwarové, kdy se obvod přepne do nízkopříkonového módu softwarově, ale probudí změnou stavu na některém vstupu nebo ve vnitřní periferii nebo plně softwarové. V tomto směru je vždy nutné se snažit vybírat součástky s co nejnižší spotřebou v tomto režimu, aby celková suma příkonu všech obvodů na plošných spojích v zařízení nebyla zbytečně velká, a nevhodně nekombinovat součástky s a bez standby či sleep módu, aby nedošlo k situaci, že například řídící procesor bude uspán, ale některé periferie, které stejně bez jím dodávaných dat nic nezmůžou, budou vesele běžet na plný výkon. Také je nutné opět vhodně napsat tu část softwaru, která přechod do standby režimu a opětovné probuzení všech obvodů řídí.

5.1 Metody snížení spotřeby zařízení

- Používání hardwarových modulů či softwarového přerušení na hlídání změn na vstupech integrovaných obvodů (když se nic neděje, může být obvod v nízkopříkonovém módu).

- Pro hlídání a zachytávání hran na vstupech, např. MCU, je nejlepší využívat časovače a bloky input capture, které obvykle podporují nějaký nízkopříkonový režim.

- Pro analogové signály je vhodné tam, kde je to možné, zvolit raději méně na energii náročné analogové komparátory než A/D převodník, nebo jeho pomocí funkci A/D převodníku aktivovat.

- Nepoužité vstupy a výstupy by měli být vždy někam připojeny, pro zamezení vzniku plovoucích vstupů, které mohou nejen způsobovat rušení a generovat náhodné stavy, ale také mohou být zdrojem různým proudů. U MCU a jiných programovatelných obvodů je vhodné u všech nevyužitých vstupů aktivovat vnitřní pull-up rezistor nebo u přepínatelných I/O pinů je vždy nastavit jako výstupní.

- Volit co nejvíce integrované součástky, tzn. obvody, které mají co nevíce v zařízení požadovaných prvků společně na jednom chipu.

- Jednotlivé součástky na desce plošných spojů dávat co neblíže k sobě

5.2 Principy využívané v integrovaných obvodech Freescale

Pro potřebu maximálního snížení spotřeby integrovaných obvodů firma Freescale využívá různé triky, mezi které patří i ty následující:

- Maximální snížení napájecího napětí a vícenapěťové napájení - protože je spotřebovávaný výkon mimo jiné úměrný velikosti napájecího napětí, je výhodné napájet různé části chipu jen tak malým napětím, které je opravdu nutné. Z těchto důvodů jsou MCU, DSP a další procesory vybavovány několika napájecími vstupy pro rozdílná napětí.

- Technologie s tranzistory s různými prahovými napětími (Multi-VT Process) — každý tranzistor v digitální integrované součástce má určité prahové napětí VT, které definuje sepnutí tranzistoru a budící proud tranzistoru. Nízkoprahový tranzistor sice poskytuje vysoký výkon díky zvýšení budícího proudu, ale odvrácenou stránkou je pak zvýšení svodového proudu a tím i spotřeby celého obvodu. Proto Freescale na jednom chipu kombinuje a využívá jak tranzistory s nízkým prahovým napětím, tam kde je to z pohledu vysokého výkonu nutné, tak i tranzistory s vyšším prahovým napětím a nižšími svodovými a parazitními proudy pro méně náročné části. To ve výsledku znamená nižší proud v standby módu.

- Technologie Active Well Bias - pomáhá řídit svodový proud hradla a tak umožňuje v reálném čase řídit spotřebu/výkon tranzistoru. Jak již bylo poznamenáno výše, pro maximální výkon se využívá spínacích tranzistorů s nízkým prahovým napětím, které však v klidovém standby režimu mají relativně velký ztrátový výkon definovaný svodovým proudem (Leakage Current). Technologie Active Well Bias ovlivňuje velikost prahového napětí tím, že zvýší napětí substrátu nad úroveň napájecího napětí VDD u PMOS tranzistorů a pod napětí země (GND) u NMOS tranzistorů. Tranzistor je tak v tzv. back bias režimu a minimalizuje se svodový proud.

- Technologie SMARTMOS - hybridní technologie Freescalu poskytuje nástroj k vytváření velmi energeticky efektivních kombinovaných obvodů s vysokou integrací. Jde o možnost současné integrace signálových analogových a digitálních částí společně s výkonovými MOSFET bloky. To například umožňuje řešení obvodů řízení napájení PMIC (Power Management IC) zahrnující ochranu zátěží, efektivní řízení a vícenásobný výstup.

- Použití spínaných zdrojů, které se vyznačují výrazně vyšší účinností než lineární LDO stabilizátory. Jejich nevýhodou je však složitější konstrukce a tedy vyšší cena, a generování většího rušení, které je zvláště nevhodné pro radiové RF aplikace. Navíc spínané měniče, pracující na vysokých frekvencích a využívající ke spínání pulzně-šířkovou modulaci (PWM), kterou nelze s uspokojivou účinností provozovat pro malé zátěže. Pro ně však lze použít spínání na základě pulzně-frekvenční modulace (PFM), které mají velmi dobrou účinnost a tím prodlužují životnost baterie. Na druhou stranu se však díky delší periodě spínání pomaleji "probouzejí" z nízkopříkonových módů.

- Dynamic process temperature compensation (DPTC) - je mechanismus, který měří frekvenci referenčního obvodu v produktu. Ten zaznamenává závislost rychlosti produktu na použité technologii a pracovní teplotě. DPTC pak sníží napájecí napětí na minimum, které stačí pro provoz na požadované pracovní frekvenci.

- Dynamic voltage and frequency scaling (DVFS) - umožňuje za běhu nastavovat hodinovou frekvenci podle aktuálního výkonového zatížení. Se snížením frekvence je také možné za běhu snížit napájecí napětí a tím i spotřebu celého obvodu. U Freescalu jsou běžné dva typy implementace této technologie: s podporou hardwaru a softwarová. Hardwarový mechanismus automaticky sleduje zatížení procesoru a řídí napájecí napětí a frekvenci s minimálním vlivem softwaru a operačního systému. U obvodů neobsahující DVFS hardware je však možné tuto funkci realizovat softwarově.

- State-retention power gating (SRPG) - je technologie, která umožňuje snížit napájecí napětí až na nulovou hodnotu pro většinu bloků logických hradel, které momentálně nepracují, zatímco zůstává připojeno jen napětí potřebné pro udržení logických stavů. SRPG může snížit spotřebu obvodu, když je aplikace v nízkopříkonovém módu, ale zároveň umožňuje rychlé probuzení (fast wake-up times). Snížení napájení až k nule umožňuje odstranit dynamický i statický výkon (spotřebu).

- Hradlování hodinového signálu (Clock gating) - je efektivní strategie, která se široce využívá jako pomoc ke snížení spotřeby, zatímco udržuje stejnou úroveň výkonnosti a funkčnosti. Obvody spotřebovávají více výkonu, když jsou taktovány, než když jsou hodiny hradlovány nebo vypnuty. Hodiny mohou totiž spotřebovat až 40 % celého aktivního výkonu součástky. Zastavením hodinového signálu v nevyužívaných částech chipu tak dochází k výrazným úsporám energie.

- Umístění zdroje hodin (Clock locating) - fyzické mapování generátoru hodin na chipu může mít významný efekt na účinnost využití energie prostým zkrácením signálových cest zvláště u vysokofrekvenčních hodin. Například globální distribuce hodin musí vhodně spravovat zpoždění na konci cest. Špatně navržená distribuce hodinového signálu vyžaduje větší bufferování k úpravě náběžných hran, čímž se zvyšuje ztrátový výkon v síti přenosu hodin. signálu.

- Výkonové módy (Power Modes) - pro komplexní přímou regulaci spotřeby jsou integrované obvody vybavovány alespoň jedním nebo více výkonovými režimy. Ty umožňují přizpůsobit napájení různých částí obvodu, například MCU, podle aktuálního požadovaného výpočetního výkonu, rychlosti nebo množství využívaných periférií. Někdy je jako základní mód označován plně aktivní obvod s maximálním příkonem (run mode) a další módy jsou pak odstupňované nízkopříkonové (wait, stop, standby, sleep). Správným softwarovým přepínáním těchto režimů lze výrazně zredukovat průměrnou spotřebu zařízení a tedy prodloužit například životnost baterií.

6 Volba vhodného procesoru

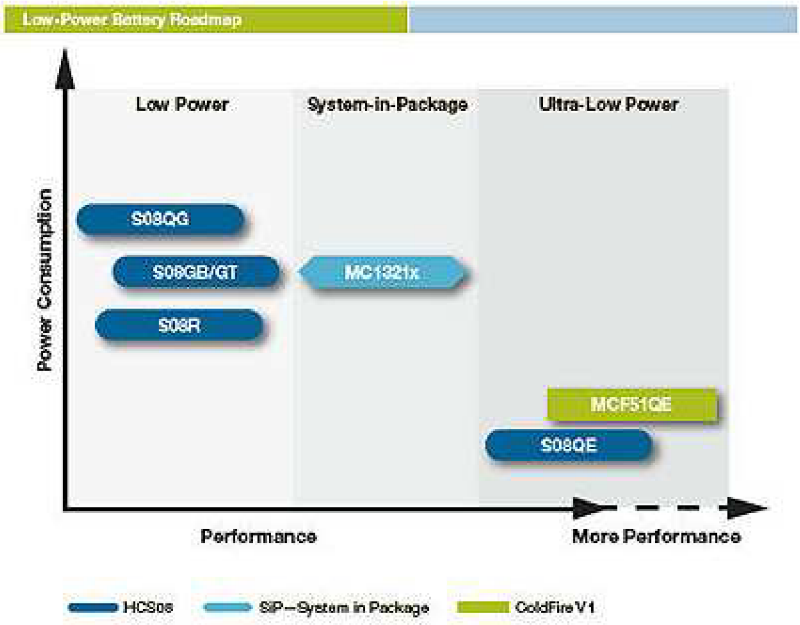
Toto téma lze snadno demonstrovat na dvou rozdílných typech nízkonapěťových a nízkopříkonových procesorů rodiny Flexis QE firmy Freescale. Proti sobě například můžeme postavit základní 8bitové MCU řady S08 QE (MC8S08QE128) a výkonné 32bitové MCU ColdFire V1 QE (MCF51QE128). Oba se spokojí s velmi nízkým napájecím napětím 1.8V až 3.6V, ale vyznačují se velmi rozdílným výpočetním výkonem, resp. rychlostí jádra. Výkonné rychle taktované jádro však také vyžaduje pořádně "nakrmit" a tak, jak lze vidět z tabulky, spotřeba MCU S08 je typicky poloviční v plném run módu a v nízkopříkonovém run módu, ale už jen lehce rozdílná v stop3 módu a úplně rovna v napájecím stop2 módu, kdy jsou už jen minimálně napájeny některé základní bloky (RAM apod.).[3]

	MCF51QE128	MC9S08QE128
Run mode @ 2 MHz CPU / 1 MHz bus	2 mA	1 mA
Run mode @ 50 MHz CPU / 25 MHz bus	27 mA	11 mA
Low-power run Mode @ 32 kHz CPU/16 kHz bus	50 uA(Flash) 19 uA (RAM)	22 uA (Flash) 9 uA (RAM)
Stop 2 – Lowest power mode; partial power down of circuits	370 nA	370 nA
Stop 3 - Int. circuits loosely regulated; clocks at low frequency	520nA	450 nA
Stop 3 - Wake Up Time	6 µs	6 µs

Obr. 7

Málo znalý konstruktér či uživatel by pak asi měl nutkání z pohledu maximálního ušetření energie sáhnout právě po MCU S08. Jenže je nutné vzít v úvahu, jak budou oba porovnávaní zástupci nuceni dlouho pracovat v nejžíznivějším plném run módu. 32bitové MCF51QE má totiž zhruba 4x větší výkon než 8bitová S08 a navíc vykonává instrukce v rytmu frekvence CPU, na rozdíl od S08, kde se toto řídí 2x pomalejší frekvencí sběrnice. A tak zatímco při náročných výpočetních operacích bude S08 téměř neustále pracovat v run módu, aby to "stíhala", ColdFire to zvládne mnohokrát rychleji a zbytek času může strávit například v stop3 módu. Papírový náskok S08 ve spotřebě energie z dlouhodobého pohledu tak mizí, a naopak ve výsledku na tom může být i hůř než ColdFire.

To byl ale náročný výpočetní úkol. Co se však stane, když bude aplikace vyžadovat hlavně práci s periferiemi na chipu MCU (např. A/D převodníkem, sériovou komunikací apod.)? Rychlost těchto periferií je u obou zástupců zhruba stejná a je dána například vzorkovací frekvencí či přenosovou rychlostí, které jsou zvláště v porovnání s rychlostí jádra ColdFiru velmi pomalé. Ve výsledku tak ColdFire hodně času stráví čekáním na potřebná data. Žel, musí čekat v nejlepším případě v nízkopříkonovém run módu, protože další šetrnější módy již nepodporují provoz těchto periferií. Zde tak MCU S08 jasně vítězí.



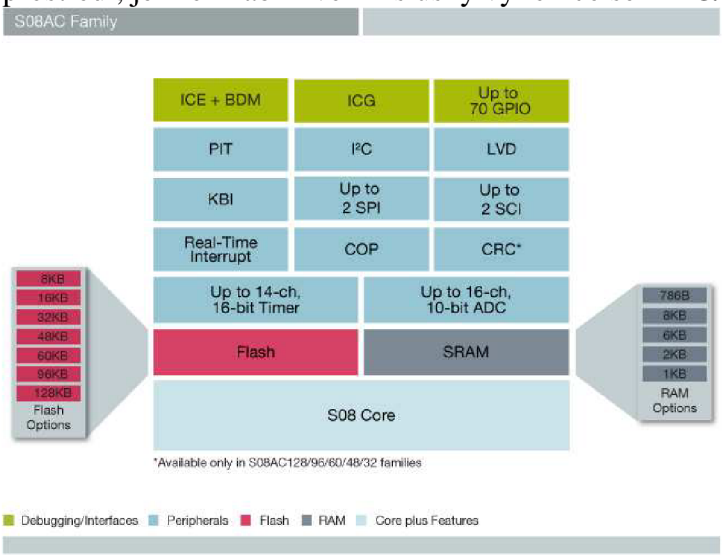
Obr. 8 Závislost výkonu vybraných procesorů na spotřebě

7 Charakteristika vybraných MCU

Mikrokontroléry rodiny AC

Společnost Freescale Semiconductors představuje novou rodinu mikrokontrolérů Fexis AC s excelentními EMC/EMI vlastnostmi do „rušivých“ prostředí.

Jelikož s rostoucím výkonem i rostoucími nároky na množství komponent v návrzích narůstají i problémy s případným rušením, je třeba se náležitě věnovat i prostředkům, kterými lze takovýmto problémům zabránit. Společnost Freescale představila novou rodinu mikrokontrolérů Flexis AC, obsahující 8bitová zařízení MC9S08AC128/96/60/48/32 a 32bitové V1 ColdFire mikrokontroléry MCF51AC256/128, přímo navrženou do rušivých prostředí, jelikož nabízí velmi slušný výkon co se EMC/EMI vlastností týče.



Obr. 9

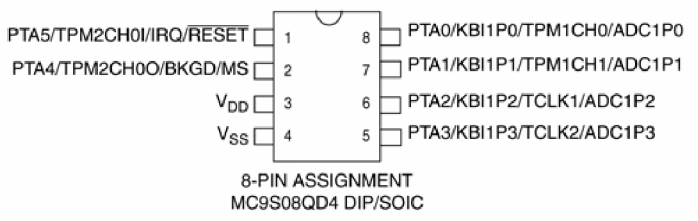
Tyto mikrokontroléry dále pak jistě potěší řadou on-chip periferií, redukujících počet komponent systému, jeho složitost a jistě i náklady. MCU Flexis AC nabízí možnost přechodu od 8bitových produktů MC9S08AW společnosti Freescale.

Rodina Flexis AC je vhodná především pro jednočipové aplikace pro řízení motorů a HMI řešení (Human-Machine Interface), zahrnujících bílé zboží, jako jsou myčky, sušičky, ledničky, pračky a pod. [2].

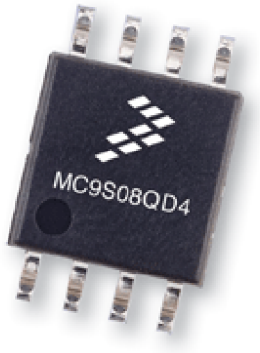
Vlastnosti mikrokontroléru MC9S08AC
40MHz jádro S08 s 20MHz sběrnicí
až 8kB SRAM / až 128kB Flash
kompatibilita vývodů s mikrokontroléry AW60
ICG (Internal Clock Generator)
2x SPI, 2x SCI a I2C
16bitové časovače: 1x 2kanálový, 2x 6kanálový
16kanálový, 10bitový ADC
Real Time Interrupt
až 70 GPIO
pouzdra 44 LQFP, 64 QFP a 80 LQFP

Mikrokontrolér rodiny QD

Po osvědčené sérii mikrořadičů HC(S)08 s vyšším počtem pinů (MC9S08AW60) nyní společnost Freescale přichází s novou řadou součástek HC(S)08 v 8pinovém pouzdře, pojmenovanou MC9S08QD4 [3].



Obr. 10



Obr 11

Mezi hlavni přednosti tohoto mikrořadiče jednoznačně patří integrace tří programovatelných 16bitových Timer-PWM kanálů a čtyř vstupů 10bitového analogově-číslicového převodníku (ADC). Tyto periferie spolu s nízkopříkonovým 16MHz procesorovým jádrem umožní řízení celé řady standardních aplikací a soustav, pracujících v reálném čase. Řada mikrořadičů MC9S08QD4 je určena především pro aplikace řízení malých přístrojů, ventilátorů, ručního nářadí, inteligentních relé a jednoduchých průmyslových pohonů. Tyto mikrořadiče lze dále použít v oblastech realizace přístrojové techniky, jednoduché měřicí techniky, poplašných zařízení, alarmů a mnoha dalších.

Vlastnosti:

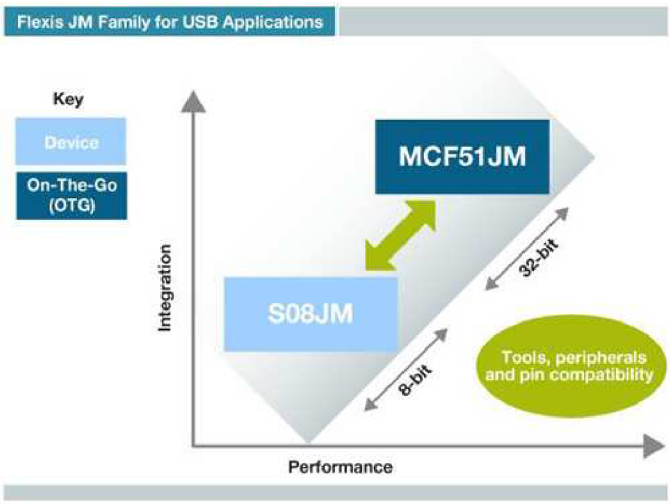
MC9S08QD4 nabízí vysoce integrovaný a výkonný prostředek, který je s vlastnostmi na prvek pro tak nízký počet pinů skutečně optimální. Kromě hlavních předností tohoto mikrořadiče, tedy programovatelných Timer-PWM kanálů a analogově-číslicových převodníků, je nutno vyzvednout integrovanou paměť Flash, programovatelnou přímo v aplikaci, a interní modul zdroje hodin poskytující rozlišení 0,2% s odchylkou pouze 2% v rámci celého rozsahu pracovních teplot a napětí. Integrovaná pamět Flash a interní modul zdroje hodin redukují počet externích součástek, jako jsou krystaly nebo rezonátory a sériové paměti EEPROM na plošném spoji [3].

Mikrokontroléry rodiny JM

Společnost Freescale rozšířila nabídku obvodů Flexis o první obvody s podporou USB rozhraní – rodinu Flexis JM. Tyto obvody kombinují 8, 16 a 32bitový výkon spolu s možnostmi konektivity USB pro nejjednodušší a velmi rychlý vývoj širokého spektra aplikací, určených pro průmysl i spotřební elektroniku.

8bitový S08JM a 16/32bitový MCF51JM jsou softwarově a pinově kompatibilní mikrokontroléry Flexis od společnosti Freescale. To, co je na nich unikátní, je, že nyní podporují USB OTG konektivitu a umožňují tak návrhářům vyvíjet jednoduché 8bitové či vysoce výkonné 32bitové aplikace s podporou USB a co je hlavní, s naprosto stejnými vývojovými prostředky. Představte si uvedení celé řady USB produktů s různými stupni výkonu bez nutnosti použití různých vývojových nástrojů.

Do jednotlivých řad patří dva konkrétní obvody, lišící se dostupnou pamětí a dalšími parametry; základ (jádro) však zůstává stejný. Jejich obrovskou výhodou je pinová, periferní a softwarová kompatibilita, která umožňuje návrhářům plynulý přechod mezi jednoduchostí 8bitu a výkonem 32bitového systému. Rodiny obvodů S08JM a MCF51JM mají nově k dispozici rozhraní USB, která umožní ještě větší kontrolu a komunikační možnosti nových i stávajících zařízení.

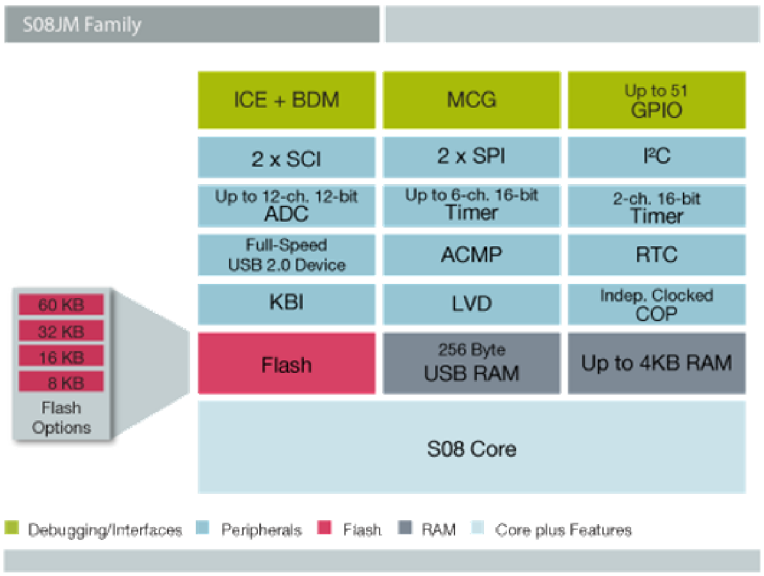


Obr. 12

Osmi bitová řada Flexis S08JM

Řada obvodů S08JM rozšiřuje základní nabídku 8bitových MCU Freescale Flexis o konektivitu USB. Společně s až 60KB paměti Flash, full speed USB 2.0 a 12kanálovým AD převodníkem s 12bitovým rozlišením, tvoří slušný základ pro mnoho jednoduchých aplikací, kde by softwarová implementace USB rozhraní či externí obvody zbytečně navyšovaly cenu zařízení. Rodina S08JM mimo jiné obsahuje několik ochranných systémů, jako je detektor nízkého napájecího napětí a COP modul (Computer Operating Properly). Všechny mikrokontroléry v této řadě používají vylepšené jádro HCS08 [3].

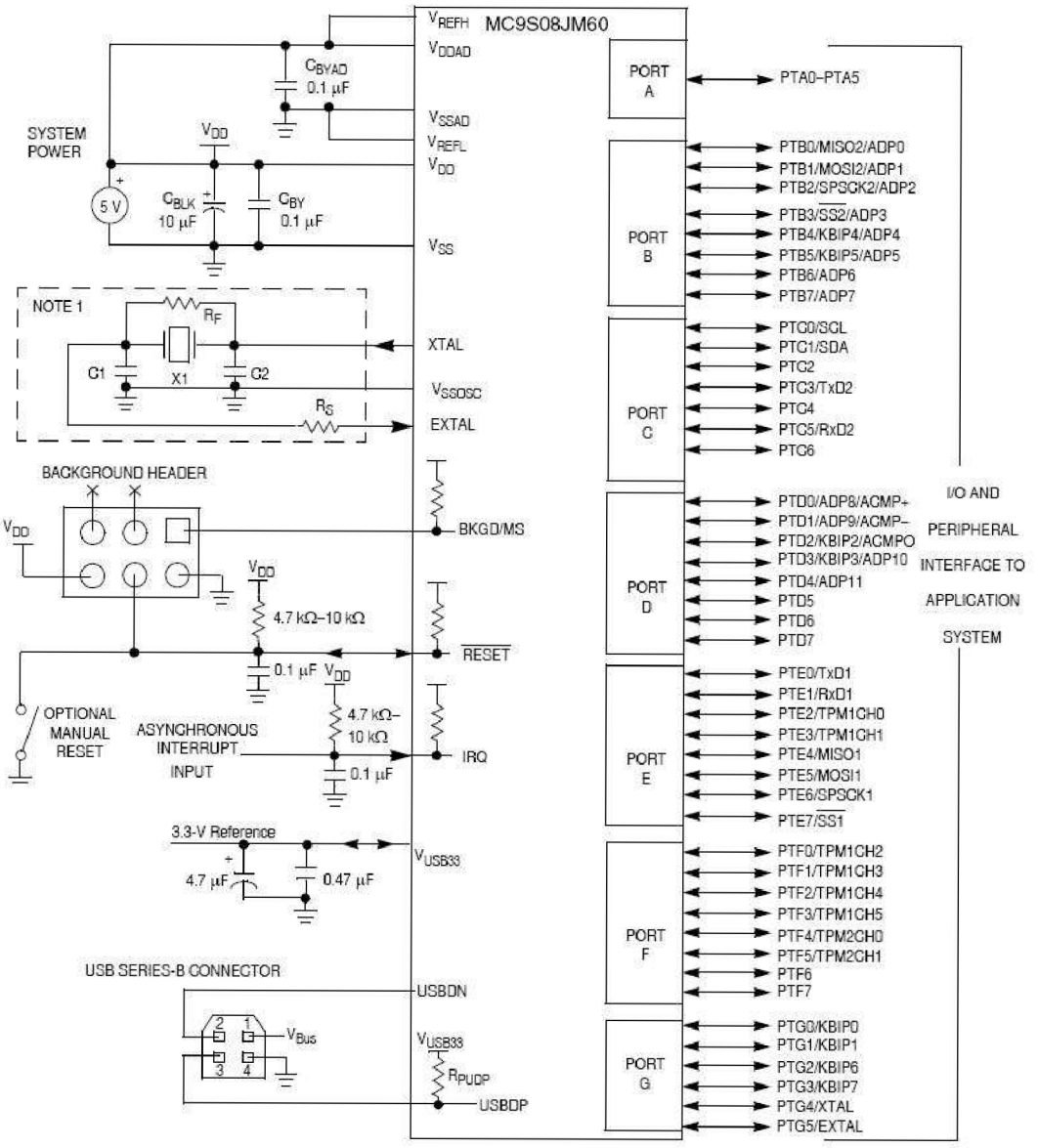
Obvody S08JM se perfektně hodí pro aplikace řízení v průmyslu, v domácí automatizaci i spotřební elektronice. Takovými aplikacemi mohou být např.: záložní zdroje UPS, průmyslové tiskárny, dotykové panely a mnoho dalších. Freescale poskytuje dva bezplatné softwarové USB stacky pro ulehčení integrace USB rozhraní do budoucích i stávajících aplikací, USB-LITE Stack od společnosti CMX a USB-MINI Stack společnosti Freescale.



Obr. 13

Vlastnosti mikrokontroléru MC9S08JM60

jádro taktované na 48MHz
24MHz interní zběrnice
debugovací systém
paměť Flash: až 60KB
paměť RAM: až 4KB
USB RAM: 256B
USB 2.0 full-speed (12Mbps) device
12 kanálový AD převodník, 12bitové rozlišení
analogový komparátor
2x SCI
2x SPI 8bitové, nebo 16bitové
I2C sběrnice
časovač (timer) 1x 2kanálový, 1x 6kanálový
8x KBI (Key board interrupt)
input/output porty



Obr. 14 Piny a připojení MC9S08JM60

8 Vývojové prostředí CodeWarrior

Původní CodeWarrior byl vyvinut společností Metrowerks. První verze byla určena pro počítače Macintosh s procesorem Motorola 68000, přičemž hlavní část vývoje byla prováděna s původní skupinou THINK C standardu. Stejně jako THINK C, který byl znám pro svou rychlost kompilace, byl i CodeWarrior rychlejší než Macintosh Programmer's Workshop (MPW) od společnosti Apple. V srpnu 2005 byl stále ještě rychlejší než originální Applovské GCC založené na vývojových nástrojích Xcode [3].

CodeWarrior byl klíčovým faktorem úspěchu společnosti Apple při přechodu ze strojové architektury procesoru 68K na procesor PowerPC, protože poskytl kompletní a solidní překladač. Konkurenční MPW nástroje a Symantec C++ mu většinou nemohly konkurovat. Pro Metrowerks bylo snadné vytvářet vícekódové architektury ("fat binary"), které mohly být použity na procesorech 68K i PowerPC.

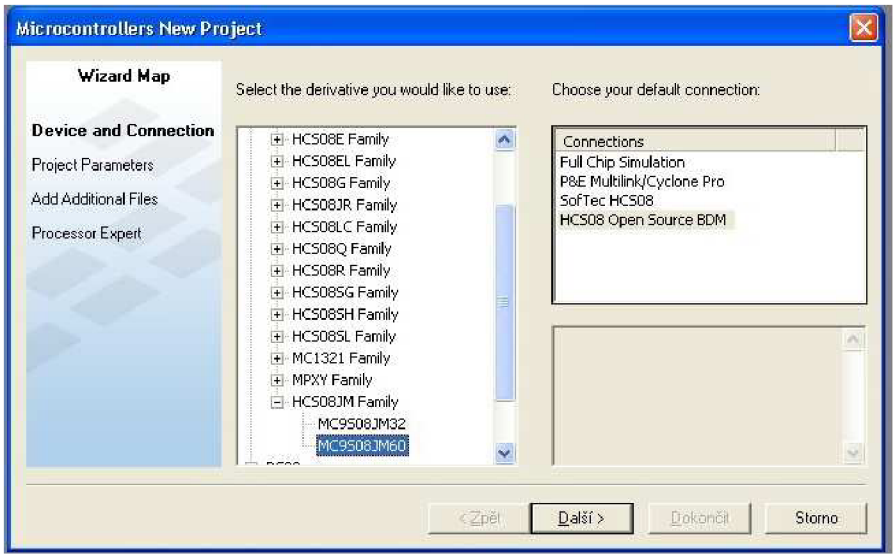
Poté, co byla v roce 1999 společnost Metrowerks koupena Motorolou, začala věnovat méně pozornosti překladačům pro stolní počítače a soustředila se na vestavěné systémy. Dne 29 července 2005 společnost oznámila, že s příchodem nové verze CodeWarrior Pro 10, dojde k přerušení dalšího vývoje CodeWarrioru pro systémy Mac. Přestože Metrowerks nikdy neudal přesné důvody, bylo souzeno, že hlavním důvodem byl pokles poptávky po CodeWarrioru poté, co Apple začal zdarma distribuovat vývojové nástroje s operačním systémem Mac OS X. Kromě toho Apple přešel na procesory Intel, pro které neměl Metrowerks odpovídající produkt. V roce 2005 Metrowerks prodal svoji technologii kompilátorů pro procesor Intel firmě Nokia.

Během svého rozkvětu byl CodeWarrior oblíben díky častému vydávání nových verzí, několika aktualizacím během roku a znám díky masivní reklamní kampani. Jejich tzv. „geekware“ košile byly dokonce na módních stránkách novin New York Times.

Inicializační nástroj zařízení vývojového studia umožňuje rychle a snadno konfigurovat on-chip registry a generovat inicializační kód. Samotný kód pak může být přidán přímo do projektu nebo uložit jako samostatný textový soubor.

Další součástí vývojového studia je software Processor Expert, který umožňuje rychle vytvářet návrhy aplikací a pomocí grafického rozhraní definovat potřebnou funkčnost pro svou aplikaci, přičemž Processor Expert se postará o generaci otestovaného a optimalizovaného kódu pro aplikaci zvolené Flexis nebo ColdFire zařízení.

CodeWarrior Development Studio disponuje i příjemným průvodcem tvorby projektů – tzv. Project Wizard, který vývojářům napomůže rychleji vytvořit fungující projekt, a to už i jen několika kliknutími myši.



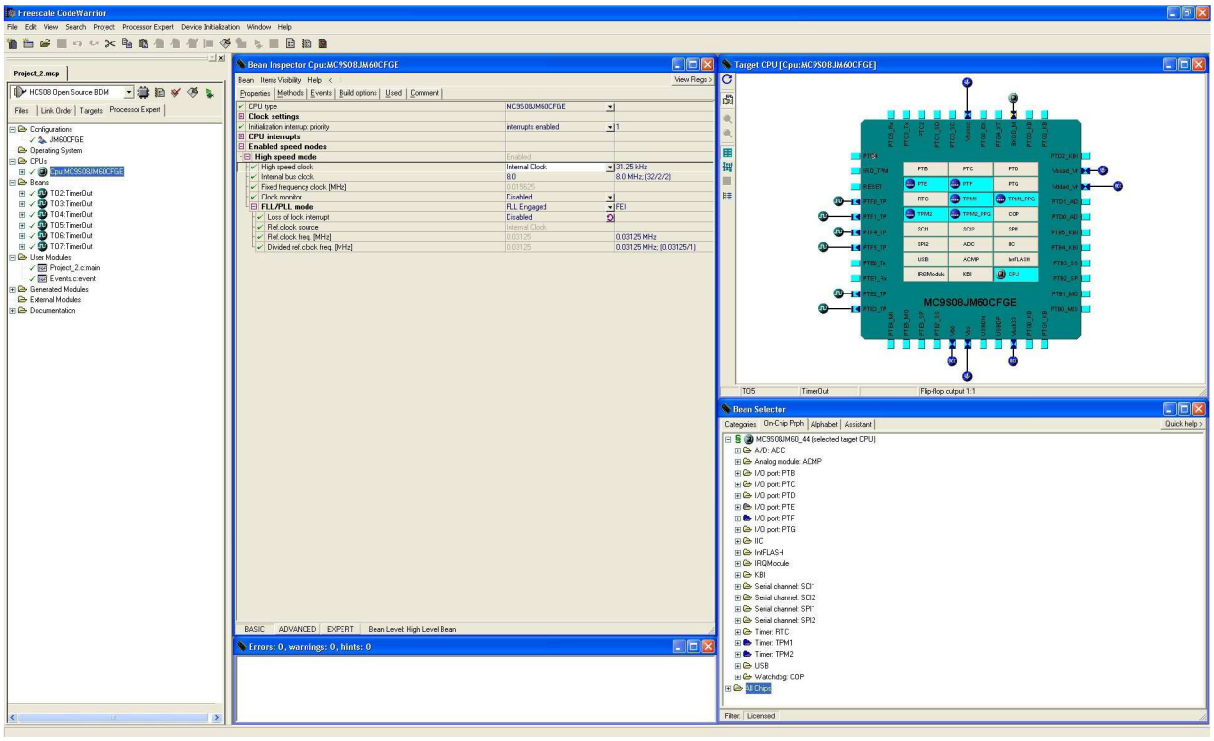
Obr. 15 Vytváření nového projektu pomocí průvodce Project Wizard

8.1 Processor Expert

Úvodní inicializace mikrokontroléru, jeho jádra a periferií, obvykle vyžaduje dobrou znalost architektury daného integrovaného obvodu a hlavně jeho registrů. To však často znamená, pokud začínáte s novým MCU, nastudovat často i několika set stránkový pdf manuál. Takový postup však v dnešní době rychlého vývoje produktů a častého přechodu z jednoho typu mikrokontroléru na jiný není úplně to pravé. Trendem je vzít určité MCU s požadovanými vlastnostmi a hned realizovat řídící algoritmus [3]. K tomuto postupu tvorby aplikace slouží integrovaná utilita Procesor Expert. Zde se vyberou potřebné vlastnosti a parametry myší a utilita již sama vše přeloží do zdrojového C kódu, kam již stačí jen doplnit potřebný algoritmus.

8.2 Ukázka využití Processor Expertu

Jak již bylo zmíněno, Procesor Expert je integrovanou utilitou vývojového prostředí CodeWarrior. Dovoluje snadno nastavit systémové registry jádra a jednotlivých periferií, vstupů i výstupů pomocí volby vlastností z nabídek a menu, a následně z toho vygenerovat kód v jazyku C a spolu s tím i vygenerovat konkrétní funkce v C



Obr. 16 Nastavovací okno Procesor Expertu

V levém okně je navigační strom (Beans) ukazující vložené položky, které se později budou překládat. Prostřední okno je pro nastavování hlavních parametrů MCU jádra (Bean Inspector Cpu), např. nastavení externího taktování, pokud je použito. V pravém spodním okně (Bean selektor) se volí požadované periferie, které se použijí v programu. V pravém horním okně je zvolené pouzdro MCU.

V navigačním stromě (Beans) je šestkrát vložena položka časovač (timer), na kterém mohou být ve vývojovém kitu připojeny LED diody. U jednotlivých časovačů se nastaví šířka pulzu (pulse width). Nyní máme vše požadované navolené a nastavené (kdykoliv lze však se do Processor Expertu vrátit a cokoliv doplnit nebo změnit). Pro vygenerování a nahrání kódu do MCU stiskneme debug.

Aniž bychom museli složitě vytvářet zdrojový kód, pomocí Procesor Expertu jsme pouhým klikáním myši naprogramovali jednoduchou aplikaci blikajících diod.

9 Závěr

Bylo popsáno několik typů mikrokontrolerů od výrobce Freescale řady S08. Jsou to 8bitové mikrokontrolery, které se rozdělujují do několika skupin. Tyto skupiny jsou nazývány rodinami. Jednotlivé rodiny MCU mají určité charakteristické vlastnosti. Při popisu MCU řady S08 byl pro bližší popis vybrán MCU z rodiny JM, který podle mého uvážení má obrovské využití v elektronice a sdělovací technice. MCU z rodiny JM obsahuje podporu USB rozhraní (viz kapitola 7). S tímto mikrokontrolérem byl navržen vývojový kit, nazvaný podle MCU jako DEMO9S08JM60 (viz. příloha 1). Návrh desky plošných spojů byl realizován v programu Eagle jako dvoustranná deska. Při výrobě desky byla použita fotomaska. Na desce jsou obsaženy všechny základní periferie, jejichž funkce je ověřena souborem ukázkových aplikací (viz příloha 2).

10 Literatura

[1] HRBÁČEK, J. Komunikace mikrokontrolérů s okolím 1 a 2. Praha. BEN – technická literatura, 2002.

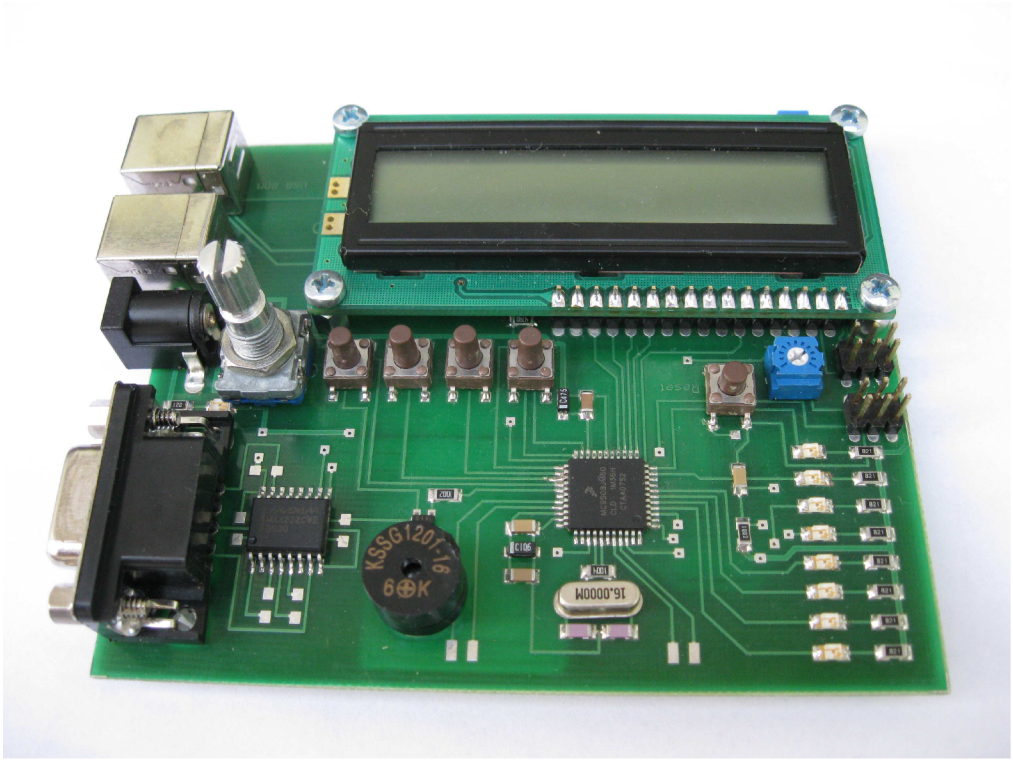
[2] Internetové stránky firmy Freescale. Dostupné na: http://www.freescale.com

[3] Internetové stránky http://hw.cz/

[4] Datasheet obvodu MC9S08JM60

Příloha 1 - Vývojový kit DEMO9S08JM60

Uživatelská příručka



DEMO9S08JM60

DEMO9S08JM60 je bohatě vybavená vývojová a výuková deska, určená pro pohodlnou práci s mikrokontrolérem Freescale. S její pomocí uživatel snadno a rychle pronikne do problematiky programování mikrokontroléru včetně obsluhy periferií, a to jak interních, tak i externích (všechny běžně používané externí periferie jsou na desce obsaženy).

Periférie

- osm LED diod
- znakový LCD displej
- čtyři tlačítka
- resetovací tlačítko
- obvod krystalového oscilátoru 16MHz
- UART, budič RS 232
- rotační kodér
- piezoměnič
- periferní USB rozhraní
- napájecí obvod
- debugovací obvod (Opec Source S08 BDM)

Napájení desky

K napájení je možné použít stejnosměrné napětí 5V. Odběr ze zdroje je podle množství využívaných periférií různý, případně podle využití vlastních obvodů, připojených a napájených z desky. Pro DEMO9S08JM60 je vhodný jakýkoliv zdroj s napětím 5V se zatížitelností 500mA a výše. Druhou možností napájení je přes USB rozhraní, obsažené v mikrokontroléru. Pokud napájíme kit přes USB debugovacího obvodu (USB BDM), musíme mít zapojený „káblík“, který propojuje BDM obvod se samotným kitem. Při odpojeném káblíku jsou tyto dva okruhy galvanicky odděleny.

Resetovací obvod

Pin RESET je připojen přes pull-up na +5V, tedy naprogramovaný mikrokontrolér se při připojení napájecího napětí ihned rozběhne (pokud má správně nastavený a připojený oscilátor). K resetování je možné použít tlačítko RESET.

Oscilátor

Na desce je osazen krystal 16MHz

Debugovací obvod

K nahrávání vytvořených aplikací ve vývojovém prostředí CodeWarrior do mikrokontrolerů řady S08 slouží debugovací obvod BDM open source. Debugovací obvod je galvanicky oddělen od kitu. Před debugováním aplikace je třeba zapojit propojovací káblík mezi BDM a kitem. BDM obvod jde využít k debugování externích zařízení. Aby se efektivně využilo místa na desce, je debugovací obvod umístěn pod znakovým LCD.

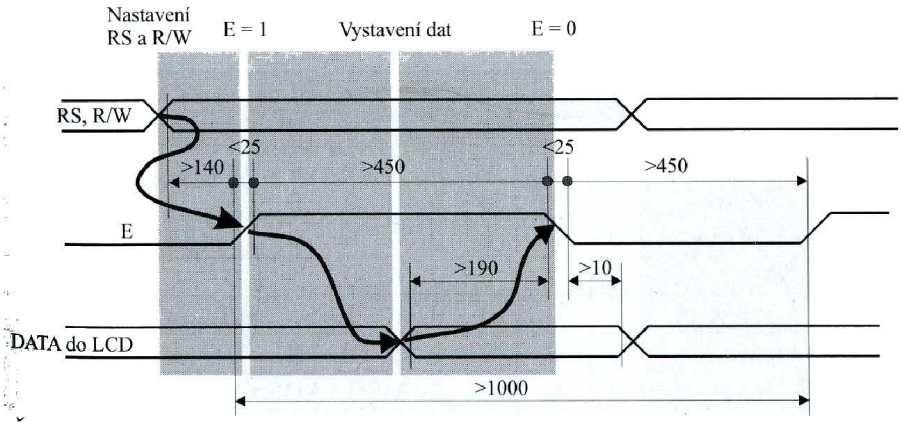
Znakový LCD displej

Jde o dvouřádkový zobrazovač. Každý řádek má 16 znaků. Znaky jsou tvořeny maticí 5 x 8 bodů. Je opatřen LED prosvětlením pro podsvícení displeje.

Dvouřádkový zobrazovač

Zobrazovač může pracovat po čtyřech nebo osmi datových linkách. Zde zobrazovač pracuje po čtyřech datových linkách D4 – D7. Data se do něj posílají dvěma zápisy. Nejdříve horní, a pak dolní půlbyte [1].

Základní komunikace s mikrokontrolérem probíhá následovně: Pomocí RS, R/W a datových vodičů jsou do LCD zobrazovače přenášeny příkazy a data určená k zobrazení. Řídící pin R/W je trvale uzemněn, tzn., že se pouze posílají data k zobrazení, ale žádná data se z LCD nečtou. Časové průběhy přenosu dat mezi mikrokontrolérem a zobrazovačem jsou na níže uvedeném obrázku (uvedené časy jsou v ns).



Časové průběhy zápisu dat

Tabulka osazení vývodů dvouřádkového zobrazovače

Číslo pinu	Signál	Popis
1	V_{SS}	zem (0V)
2	V_{DD}	napájecí napětí (4,75–5,25V)
3	V0	nastavení kontrastu (min.0,65 V)
4	Rs	identifikace registrů nebo dat
5	R/W	zápis (čtení)
6	E	povolovací signál
7	DB0	zem (0V)
8	DB1	zem (0V)
9	DB2	zem (0V)
10	DB3	zem (0V)
11	DB4	data/kód
12	DB5	data/kód
13	DB6	data/kód
14	DB7	data/kód
15	V_{LED}	napájení prosvětlovací LED (4,75–5,25 V)
16	V_{LSS}	0V

Protože mikrokontrolér je poměrně pomalé zařízení vzhledem k časovým relacím ovládání zobrazovače, není těžké dodržet časové posloupnosti bez vkládání čekacích smyček. Stačí čtyři kroky pro provedení zápisu nebo čtení do/ze zobrazovače.

Tabulka využití portů na desce DEMO9S08JM

Port B	Pin	Funkce
PTB0	23	S2
PTB1	24	S3
PTB2	25	S4
PTB3	26	S5
PTB4	27	buzzer
PTB5	28	-

Port C	Pin	Funkce
PTC0	40	SCL
PTC1	41	SDA
PTC2	42	led 8
PTC3	43	UART – TxD2
PTC4	1	led 7
PTC5	44	UART – RxD2

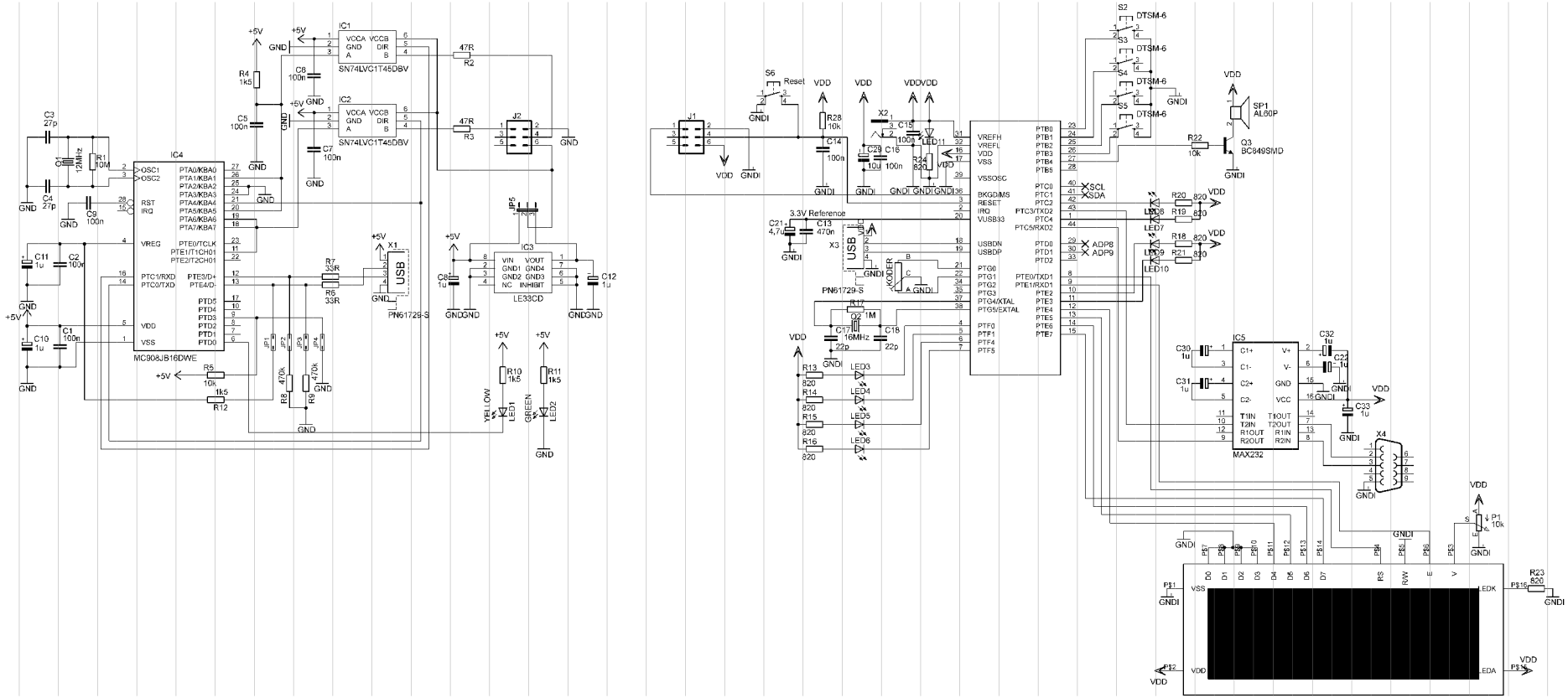
Port D	Pin	Funkce
PTD0	29	ADP8
PTD1	30	ADP9
PTD2	33	-

Port E	Pin	Funkce
PTE0	8	RS (LCD)
PTE1	9	E (LCD)
PTE2	10	led 9
PTE3	11	led 10
PTE4	12	D4 (LCD)
PTE5	13	D5 (LCD)
PTE6	14	D6 (LCD)
PTE7	15	D7 (LCD)

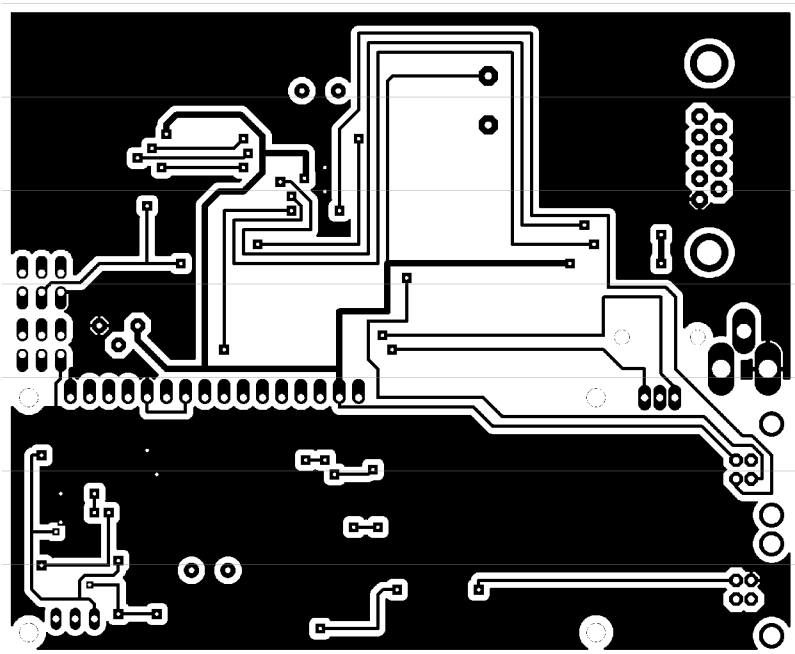
Port F	Pin	Funkce
PTF0	4	led 3
PTF1	5	led 4
PTF4	6	led 5
PTF5	7	led 6

Port G	Pin	Funkce
PTG0	21	rotační kodér
PTG1	22	rotační kodér
PTG2	34	-
PTG3	35	-
PTG4	37	XTAL
PTG5	38	EXTAL

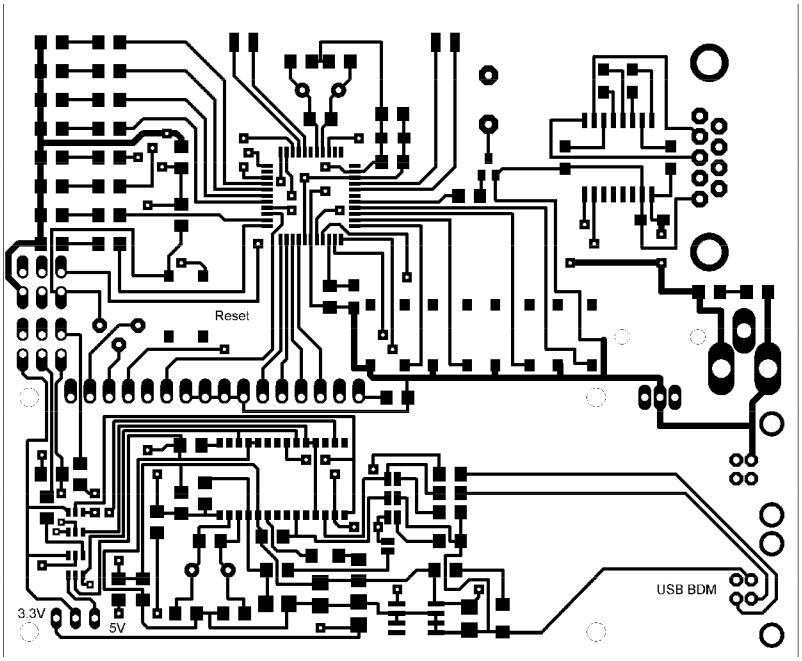
Schéma zapojení



Deska plošných spojů

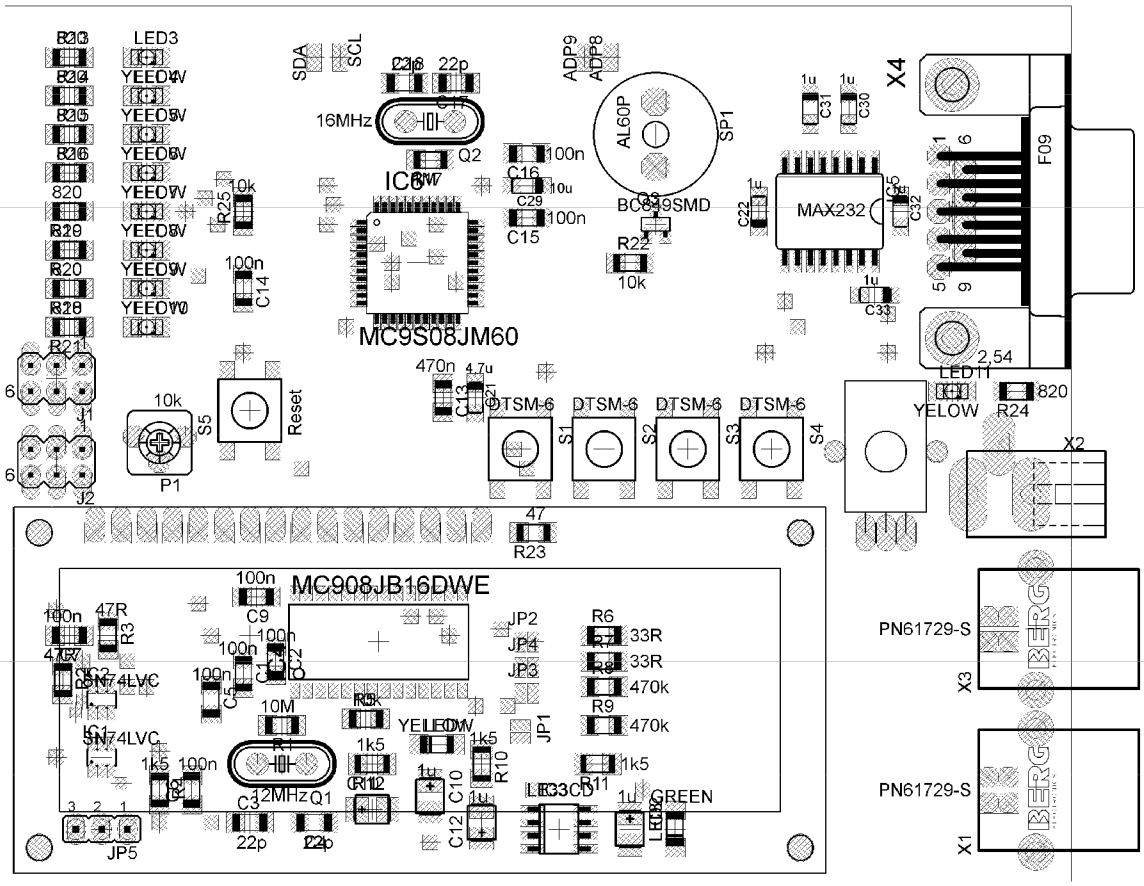


Bottom



Top

Rozmístění součástek



Seznam součástek

Zařízení	*Hodnota*	*Pouzdro*
R1	10M	R1206
R2, R3, R23	47R	R1206
R5, R22, R25	10k	R1206
R6, R7	33R	R1206
R8, R9	470k	R1206
R4, R10, R11, R12	1k5	M1206
R17	1M	R1206
R13, R14, R15, R16, R18, R19, R20, R21, R24	820	R1206
P1	10k	PK50
C1, C2, C5, C6, C7, C9, C14, C15, C16	100nF	C1206
C3, C4, C17, C18	22pF	C1206
C8, C10, C11, C12, C22, C30, C31, C32, C33	1uF	SMC_A
C13	470uF	SMC_A
C21	4,7uF	SMC_A
C29	10uF	SMC_A
LED1 - LED11	yellow	SMT1206
Q1	12MHz	HC49/S
Q2	16MHz	HC49/S
Q3	BC849SMD	SOT23
IC1	SN74LVC	SOT-23-6L
IC2	SN74LVC1	SOT-23-6L
IC3	LE33CD	SO-08
IC4	MC908JB16DWE	SOIC28
IC5	MAX232	SO16L
IC6	MC9S08JM60	LQFP44
J1, J2	MA03-2	jumper
J3	MA03	jumper
S1, S2, S3, S4, S5, S6	DTSM-6	
SP1	AL60P	
LCD	MC1602E-TRV	
KODER	RE20	
X1, X3	PN61729-S	
X2	733980-62	
X4	F09HP	

Příloha 2 – Soubor ukázkových aplikací

Beep

Při stisku tlačítka S1 je generován signál s buzzeru a současně se rožne LED 3

```
/* Including needed modules to compile this module/procedure */
#include "Cpu.h"
#include "Events.h"
#include "TPM3.h"
/* Include shared modules, which are used for whole project */
#include "PE_Types.h"
#include "PE_Error.h"
#include "PE_Const.h"
#include "IO_Map.h"

// ---------------------- definice LED diod --------------------

#define SET_LED3_ON         PTFD_PTFD0 = 0
#define SET_LED3_OFF        PTFD_PTFD0 = 1
#define SET_LED3            PTFD_PTFD0
#define GET_LED3_STATE      PTFD_PTFD0

// ---------------------- definice buzzeru ---------------------

#define SET_BUZZER_ON         PTBD_PTBD4 = 0
#define SET_BUZZER_OFF        PTBD_PTBD4 = 1
#define SET_BUZZER            PTBD_PTBD4
#define GET_BUZZER_STATE      PTBD_PTBD4


#define GET_SWITCH2_STATE      PTBD_PTBD0

void main(void)
{

  /* Write your local variable definition here */

  /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ***/
  PE_low_level_init();
  /*** End of Processor Expert internal initialization.                    ***/


    /* initialize the SWITCHs as an input with pull-up */
  PTBDD_PTBDD0 = 0;     /* input from switch 2 */
  PTBPE_PTBPE0 = 1;     /* pull-up enable */

  PTBDD_PTBDD4 = 1;     /* output to buzzer */
  PTFDD_PTFDD0 = 1;     /* output to LED3 */

  TPM1SC_TOIE = 1;      /* enable overflow interrupt */

  /*** Don't write any code pass this line, or it will be deleted during code generation. ***/
  /*** Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! ***/
  for(;;){}
} /*** End of main routine. DO NOT MODIFY THIS TEXT!!! ***/

/* END beep */
```

```
// ---------------------- přerušeni od časovače ----------------------

ISR(timer)
    {
    TPM1SC_TOF = 0;

    SET_LED3 = GET_SWITCH2_STATE;              //rožnuti ledky při stisku tlačítka
    if  (GET_SWITCH2_STATE == 0)
       SET_BUZZER = ~GET_BUZZER_STATE ;        //toggle buzzer
    }
```

Binární čítač se zobrazením na LCD a diodách

Aplikace je ovládána čtyřmi tlačítky. Tlačítko S1 čítá směrem nahoru, tlačítko S2 čítá směrem dolů, tlačítko S3 nuluje čítač a tlačítko S4 skočí na maximální hodnotu

```
/* Including needed modules to compile this module/procedure */
#include "Cpu.h"
#include "Events.h"
#include "Byte1.h"
/* Include shared modules, which are used for whole project */
#include "PE_Types.h"
#include "PE_Error.h"
#include "PE_Const.h"
#include "IO_Map.h"

#include "lcd_h.h"
#include <string.h>
#include <stdio.h>

//---------------------- switchs defines----------------------

#define GET_SWITCH2_STATE      PTBD_PTBD0
#define GET_SWITCH3_STATE      PTBD_PTBD1
#define GET_SWITCH4_STATE      PTBD_PTBD2
#define GET_SWITCH5_STATE      PTBD_PTBD3

//---------------------- LEDs defines--------------------------
#define SET_LED3_ON        PTFD_PTFD0 = 0
#define SET_LED3_OFF       PTFD_PTFD0 = 1
#define SET_LED3           PTFD_PTFD0

#define SET_LED4_ON        PTFD_PTFD1 = 0
#define SET_LED4_OFF       PTFD_PTFD1 = 1
#define SET_LED4           PTFD_PTFD1

#define SET_LED5_ON        PTFD_PTFD4 = 0
#define SET_LED5_OFF       PTFD_PTFD4 = 1
#define SET_LED5           PTFD_PTFD4

#define SET_LED6_ON        PTFD_PTFD5 = 0
#define SET_LED6_OFF       PTFD_PTFD5 = 1
#define SET_LED6           PTFD_PTFD5

#define SET_LED7_ON        PTCD_PTCD4 = 0
#define SET_LED7_OFF       PTCD_PTCD4 = 1
#define SET_LED7           PTCD_PTCD4

#define SET_LED8_ON        PTCD_PTCD2 = 0
#define SET_LED8_OFF       PTCD_PTCD2 = 1
#define SET_LED8           PTCD_PTCD2

#define SET_LED9_ON        PTED_PTED2 = 0
#define SET_LED9_OFF       PTED_PTED2 = 1
#define SET_LED9           PTED_PTED2

#define SET_LED10_ON       PTED_PTED3 = 0
#define SET_LED10_OFF      PTED_PTED3 = 1
#define SET_LED10          PTED_PTED3
```

```
// ---------------------- funkční prototypy ---------------------

void LED_write (unsigned short int LED_stat);


//---------------------- global variables ----------------------

unsigned int pom = 0;                   //pomocná proměna pro zpožďovací smyčku
unsigned int pom2 = 0;                  //pomocná proměna pro zpožďovací smyčku
char i = 0;                             //aktuální stav LED diod, globální proměna
char    text[16];


void main(void)
{

unsigned short int tl_change = 0;   //informace o změně stavu na tlačítku (detekce hrany)
 unsigned short int tl_status = 0;   //informace o současné logické úrovni tlačítka
 unsigned short int odkl = 0;        //pomocná proměna pro tlačítka

 /* Write your local variable definition here */

 /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ***/
 PE_low_level_init();
 /*** End of Processor Expert internal initialization.                    ***/

/* initialize the SWITCHs as an input with pull-up */
 PTBDD_PTBDD0 = 0; /* input from switch 2 */
 PTBPE_PTBPE0 = 1; /* pull-up enable */
 PTBDD_PTBDD1 = 0; /* input from switch 3 */
 PTBPE_PTBPE1 = 1; /* pull-up enable */
 PTBDD_PTBDD2 = 0; /* input from switch 4 */
 PTBPE_PTBPE2 = 1; /* pull-up enable */
 PTBDD_PTBDD3 = 0; /* input from switch 5 */
 PTBPE_PTBPE3 = 1; /* pull-up enable */

 lcd_init( );

 /* initialize the LEDs as outputs */
 PTFDD_PTFDD0 = 1; /* output to LED3 */
 PTFDD_PTFDD1 = 1; /* output to LED4 */
 PTFDD_PTFDD4 = 1; /* output to LED5 */
 PTFDD_PTFDD5 = 1; /* output to LED6 */
 PTCDD_PTCDD4 = 1; /* output to LED7 */
 PTCDD_PTCDD2 = 1; /* output to LED8 */
 PTEDD_PTEDD2 = 1; /* output to LED9 */
 PTEDD_PTEDD3 = 1; /* output to LED10 */
 /* off all LEDs*/
 SET_LED3_OFF;
 SET_LED4_OFF;
 SET_LED5_OFF;
 SET_LED6_OFF;
 SET_LED7_OFF;
 SET_LED8_OFF;
 SET_LED9_OFF;
 SET_LED10_OFF;


 lcd_puts( "UREL" );                // zobrazení slova na lcd displeji
 Cpu_Delay100US(10000);
```

```
  /* Write your code here */
  /* For example: for(;;) { } */

  /*** Don't write any code pass this line, or it will be deleted during code generation. ***/
  /*** Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! ***/
  for(;;){

  // časové zpoždění -> redukce zákmitu tlačítka
  for (pom=0;pom<10000;pom++)
       asm ("nop");

  odkl = 0;


// přiřazení kódu tlačítkům
  if (GET_SWITCH2_STATE == 0)
     odkl |= 0x01;

  if (GET_SWITCH3_STATE == 0)
     odkl |= 0x02;

  if (GET_SWITCH4_STATE == 0)
     odkl |= 0x04;

  if (GET_SWITCH5_STATE == 0)
     odkl |= 0x08;
// vyhodnocení změny na tlačítku
          tl_change = tl_status ^ odkl;
          tl_status = odkl;

// reakce na spuštění tlačítek
  if ((tl_change & ~tl_status)&0x01)
    i++;
  if ((tl_change & ~tl_status)&0x02)
    i--;
  if ((tl_change & ~tl_status)&0x04)
    i=0x00;
  if ((tl_change & ~tl_status)&0x08)
    i=0xff;

  sprintf (text, "DEC: %3u",i);       //převod proměnné na text
  lcd_gotoxy(0,0);                    //pozice x 0 y 0
  lcd_puts( text ) ;                  // zobrazení slova na lcd displeji

  sprintf                                                (text,                                                "BIN:
%u%u%u%u%u%u%u%u",(i>>7)&0x01,(i>>6)&0x01,(i>>5)&0x01,(i>>4)&0x01,(i>>3)&0x01,(i>>2)&0x01,(
i>>1)&0x01,i&0x01);
  lcd_gotoxy(0,1);              //pozice x 0 y 1
  lcd_puts( text ) ;            // zobrazení slova na lcd displeji


  LED_write (i);                // aktualizace dat na led diody

  }
```

```
  /*** Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! ***/
} /*** End of main routine. DO NOT MODIFY THIS TEXT!!! ***/

/* END main_display */
/*
** ###################################################################
**
**     This file was created by UNIS Processor Expert 3.03 [04.07]
**     for the Freescale HCS08 series of microcontrollers.
**
** ###################################################################
*/


// ----------------- rozsvítí nebo zhasnou ledky podle parametru funkce
void LED_write (unsigned short int LED_stat){


SET_LED3 = (LED_stat)&0b00000001 ;  //vybrání 1.pozice a nastavení 1. ledky
LED_stat >>= 1;                     //posun stavu 2. ledky na 1. pozici
SET_LED4 = (LED_stat)&0b00000001 ;
LED_stat >>= 1;
SET_LED5 = (LED_stat)&0b00000001 ;
LED_stat >>= 1;
SET_LED6 = (LED_stat)&0b00000001 ;
LED_stat >>= 1;
SET_LED7 = (LED_stat)&0b00000001 ;
LED_stat >>= 1;
SET_LED8 = (LED_stat)&0b00000001 ;
LED_stat >>= 1;
SET_LED9 = (LED_stat)&0b00000001 ;
LED_stat >>= 1;
SET_LED10 = (LED_stat)&0b00000001 ;
}
```