VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

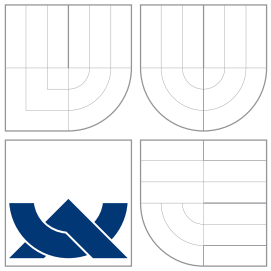
OBSLUHA PAMĚŤOVÉHO SYSTÉMU PRO PLATFORMU COLIBRI XSCALE PXA270

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

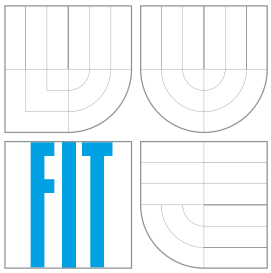
AUTOR PRÁCE
AUTHOR

LUKÁŠ MAČIŠÁK

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

OBSLUHA PAMĚŤOVÉHO SYSTÉMU PRO PLATFORMU COLIBRI XSCALE PXA270

MEMORY SYSTEM MANAGEMENT FOR COLIBRI XSCALE PXA270 PLATFORM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — LUKÁŠ MAČIŠÁK
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÁCLAV ŠIMEK
SUPERVISOR

BRNO 2009

Abstrakt

Tato bakalářská práce popisuje platformu Colibri XScale PXA270, analyzuje možnosti využití DMA přenosů na této platformě a popisuje principy práce s paměťmi typu flash a SDRAM. Práce obsahuje návrh a implementaci firmware pro obsluhu paměťového systému této platformy. Součástí práce je i popis a implementace testů polovodičových pamětí s přímým přístupem. Pro ukázku funkčnosti a možností použití firmware je implementována demonstrační aplikace.

Abstract

This Bachelor's Thesis describes platform Colibri XScale PXA270, analyses potential usage of DMA transfers on this platform and describes principles of work with flash and SDRAM memories. Thesis includes design and implementation of firmware for servicing memory systems on this platform. One part of this thesis is also the description and implementation of tests for semiconductor memories with random access. The demonstrative application is implemented to show usage and functionality of this firmware.

Klíčová slova

Paměť, flash, SDRAM, DMA, firmware, PXA 270, dynamická paměť, testovaní paměti, MARCH.

Keywords

Memory, flash, SDRAM, DMA, firmware, PXA 270, dynamic memory, memory test, MARCH.

Citace

Lukáš Mačišák: Obsluha paměťového systému pro platformu Colibri XScale PXA270, bakalářská práce, Brno, FIT VUT v Brně, 2009

Obsluha paměťového systému pro platformu Colibri XScale PXA270

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Václava Šimka. Další informace mi poskytnul Ing. Petr Axman. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Lukáš Mačišák
18. května 2009

Poděkování

Chcel by som poďakovať vedúcemu práce Ing. Václavovi Šimkovi a konzultantovi Ing. Petrovi Axmanovi za ich užitočné rady a pripomienky. Chcem poďakovať firme UNIS a.s. za poskytnutie hardvéru a softvéru potrebného na tvorbu tejto práce.

Obsah

1 Úvod **2**

2 Princípy práce s pamäťovými obvodmi **3**
2.1 SDRAM pamäte 4
2.2 Flash pamäte 8

3 Popis Colibri XScale PXA270 **11**
3.1 Popis vývojovej dosky 11
3.2 Procesor Intel XScale PXA270 12
3.3 Popis SDRAM pamäte 14
3.4 Popis FLASH pamäte 15

4 Návrh firmware pamäťovej zbernice a k nej pripojeným pamätiam **16**
4.1 Zoznam súborov tvoriacich firmware 16
4.2 Bloková schéma 17
4.3 Radič pamäte 17
4.4 Obslužné funkcie flash pamäte 18
4.5 Manažér dynamickej pamäte 19
4.6 Radič DMA 19

5 Demonštračná aplikácia **21**
5.1 Popis demonštračnej aplikácie 21
5.2 Zoznam a popis súborov 22

6 Testy SDRAM a flash pamätí **23**
6.1 Popis chýb polovodičových pamätí 23
6.2 Testovacie vzory 24

7 Záver **27**

A Obsah DVD **30**

B Manuál **31**

Kapitola 1

Úvod

Úlohou práce bolo pochopenie princípov práce s pamäťami flash a SDRAM, naštudovanie architekrtúry procesoru Intel XScale PXA270 a praktická realizácia firmware pre obsluhu pamäťového systému. Táto práca vznikla za pomoci spoločnosti UNIS a.s., ktorá poskytla potrebný hardvér a softvér. Práca je rozdelená do viacerých kapitol ktoré sa venujú rôznym problematikám.

Druhá kapitola sa venuje pamätiam flash a SDRAM. V tejto kapitole sú popísané ich vlastnosti a základné princípy fungovania a práce s nimi. U pamätí SDRAM je rozobraná aj problematika časovania signálov. U pamätí flash je aj porovnanie vlastností NAND vs. NOR technológie.

V tretej kapitole je popísaný použitý hardvér, tj. architektúra procesora, popis modulu Colibri XScale PXA270 a vývojovej dosky. Bližšie je popísaný radič pamäte a radič DMA, použitá flash a SDRAM pamäť. Možnosti využitia DMA prenosov sú tiež popísané v tejto kapitole.

Štvrtá a piata kapitola sa venuje praktickej realizácii. Vo Štvrtej kapitole je návrh firmware, obsahuje blokovú schému a popisuje implementované moduly. Piata kapitola sa venuje demonštračnej aplikácii, ktorej úlohou je ukázať funkčnosť a taktiež praktické použitie implementovaného firmware.

Šiesta kapitola popisuje možnosti testovania polovodičových pamätí s priamym prístupom. V tejto kapitole sú popísane chyby, ktoré môžu v týchto pamätiach vznikať a rôzne testovacie vzory, ktoré slúžia na odhalenie týchto chýb. Taktiež je tu popísané praktické použitie niektorých z testovacích vzorov. V závere kapitoly sú uvedené dosiahnuté výsledky testov na použitých pamätiach.

V poslednej siedmej kapitole je záver, ktorý rozoberá dosiahnuté výsledky, splnenie zadania a možnosti použitia firmware.

Kapitola 2

Princípy práce s pamäťovými obvodmi

Pamäťové obvody, ďalej už len pamäť, je možné definovať ako sklad digitálnych dát určených pre spracovanie počítačom. Pamäte môžme podľa rýchlosti rozdeliť do týchto základných skupín:

- **registre procesora** – najrýchlejší typ pamäte.
- **rýchla vyrovnávacia pamäť** – obsahuje dáta, ktoré procesor často spracováva alebo pravdepodobne bude spracovávať.
- **operačná pamäť** – je pomalšia ako rýchla vyrovnávacia pamäť a rýchlejšia ako externá pamäť. Tvorí medzivrstvu medzi týmito pamäťami. Medzi operačné pamäte patria napr. pamäte typu SDRAM, DDR-SDRAM, DDR2-SDRAM.
- **externá pamäť** – z vyššie určených typov najpomalšia a má väčšinou najväčšiu kapacitu. Využíva sa ako sklad dlhodobo potrebných dát. Medzi externé pamäte patria napr. pevné disky a flash pamäte.

Podľa závislosti na napätí môžme pamäte rozdeliť na:

- **napäťovo závislé** – aby sa dáta zachovali, musia byť pripojené na napájacie napätie. Do tejto skupiny patria operačné pamäte, rýchle vyrovnávacie pamäte a registre.
- **napäťovo nezávislé** – pri strate napájaceho napätia pamäte sa dáta zachovajú. Do tejto skupiny patria externé pamäte, pamäte typu FLASH, EPROM, EEPROM, atď.

Podľa závislosti na obnovovaní dát môžme pamäte rozdeliť na:

- **statické** – nepotrebujú obnovovanie dát, aby sa dáta zachovali. Sem patrí napr. SRAM, EEPROM.
- **dynamické** – potrebujú obnovovanie dát v pravidelných intervaloch, aby sa dáta zachovali. Sem patrí napr. SDRAM, DDR-SDRAM.

Podľa prístupu k dátam ich môžme rozdeliť na pamäte:

- **s priamym prístupom** – prístupová doba ku každej pamäťovej bunke je rovnaká.
- **so sekvenčným prístupom** – prístupová doba ku pamäťovej bunke nie je rovnaká.

Táto práca sa konkrétne venuje v súčasnej dobe veľmi používaným pamätiam typu SDRAM a flash.

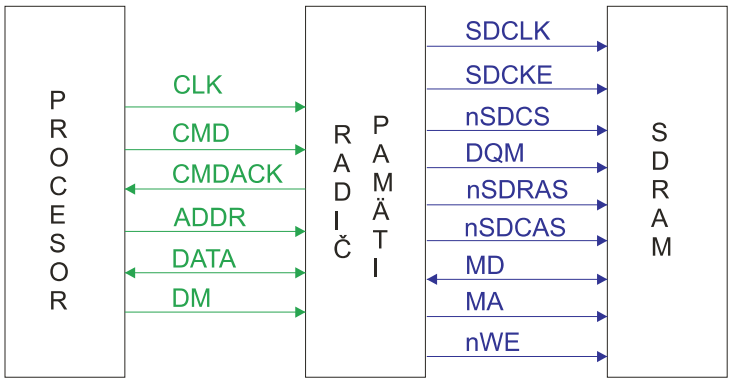
2.1 SDRAM pamäte

SDRAM – Synchronous Dynamic Random Access Memory, je to pamäť s priamym prístupom. Je tvorená polovodičovými prvkami. Táto pamäť umožňuje čítanie aj zápis a to pomocou príkazov. Informácie sa uchovávajú pomocou pamäťového prvku, ktorý tvorí kondenzátor a tranzistor. Kondenzátor je energeticky závislý, preto je nutné túto pamäť pravidelne obnovovať. Každý takýto prvok obsahuje informáciu o veľkosti jeden bit. Niekoľko pamäťových prvkov tvorí jednu pamäťovú bunku. Veľkosť pamäťovej bunky môže byť napríklad 1Byte = 8Bitov a teda 8 pamäťových prvkov. Pamäťové bunky sú usporiadané do viacerých pamäťových polí - bánk. Každá pamäťová bunka je identifikovaná adresou banky, adresou stĺpca a riadku v danej banke.

Komunikácia medzi radičom pamäte a pamäťou a samotný prenos dát je synchronizovaný pomocou hodinového signálu. Hodinový signál generuje radič pamäte. Narozdiel od DRAM sa prenos dát vykonáva bez opakovaného zápisu adresy, čo umožňuje prenos dát po blokoch („burst“). Prenosom dát po blokoch sa rýchlosť SDRAM oproti DRAM výrazne zvyšuje. Veľkosť prenášaného bloku dát je definovaná v registri módu.

SDRAM pamäte sa vo vstavaných systémoch používajú ako operačné pamäte, z ktorých je vykonávaný program, alebo ako zásobník pre dynamické dáta. Tieto a podrobnejšie informácie o signáloch, načasovaní signálov a fungovaní pamäte SDRAM je možné nájsť v [5] a [15].

Popis signálov:



Obrázek 2.1: Signály pamäte SDRAM

Na obrázku 2.1 sú zobrazené základné signály potrebné na obsluhu pamäte SDRAM. Počet a úloha jednotlivých signálov sa môže podľa architektúry radiča pamäte a samotnej pamäte líšiť. Napríklad niektoré pamäte môžu mať osobitný vstupný signál na adresáciu banky, v našom prípade sa adresa banky bude prenášať spolu s adresou riadku a adresou stĺpca. Signály môžeme rozdeliť do dvoch základných skupín: signály medzi procesorom a radičom pamäte a signály medzi radičom pamäte a pamäťou.

Signály medzi procesorom a radičom pamäte:

- **CLK** – hodinový signál generovaný procesorom pre radič pamäte,
- **CMD** – príkaz od procesoru,

- **CMDACK** – potvrdenie prijatia príkazu radičom pamäte,
- **ADDR** – adresa pamäťovej bunky,
- **DATA** – prenášané dáta do alebo z procesoru,
- **DM** – maska odosielaných dát.

Procesor generuje pre radič pamäte synchronizačný hodinový signál (CLK). Procesor posiela radiču pamäte príkazy (CMD), ktoré radič pamäte spracuje, pošle procesoru potvrdenie prijatia (CMDACK) a vykoná ich. Pri zápise do pamäte procesor posiela radiču adresu (ADDR) kam sa budú dáta zapisovať, dáta (DATA) ktoré sa majú zapisovať a nastavuje masku dát (DM). Pri čítaní z pamäte procesor posiela radiču adresu, odkiaľ sa majú dáta načítať.

Signály medzi radičom pamäte a pamäťou SDRAM:

- **SDCLK** – hodinový signál generovaný radičom pamäte,
- **SDCKE** – aktivácia hodinového signálu SDCLK,
- **nSDCS** – výber pamäťového čipu („chip select“),
- **nSDRAS, nSDCAS a nWE** – spoločne definujú kód príkazu, ktorý je definovaný podľa tabuľky 2.1,
- **DQM** – maska odosielaných dát,
- **MD** – prenášané dáta,
- **MA** – adresa pamäte, tj. číslo riadku, stĺpca a banky.

Radič pamäte generuje pre pamäť hodinový signál (SDCLK) a povoľovací signál (SDCKE). Pri aktívnom SDCKE je SDCLK povolený, pri neaktívnom je zakázaný. Zákázanie hodinového signálu sa využíva pri vypnutí pamäte a pri seba-obnoveniu pamäte („self-refresh“). Konkrétna pamäťová bunka je presne určená pomocou adresy riadku, stĺpca a banky. V prípade že je pamätí viacero, tak na výber konkrétnej pamäte slúži výber čipu (nSDCD). Adresa riadku, stĺpca a banky sa posiela pomocou adresových signálov (MA). Základné príkazy, ktoré posielá radič pamäte, sú popísané v tabuľke 2.1. Dáta sa prenášajú pomocou dátových signálov (MD). Pre zápis do pamäte je potrebné ešte nastaviť signál povolenia zápisu (nWE) a masku dát (DQM). Register módu sa využíva pri prenose dát po blokoch, kedy je v tomto registri definovaná veľkosť prenášaných dát. V registri módu je definovaná aj CAS čakacia doba, tj. doba, za ktorú budú dáta k dispozícii na dátových vodičoch.

Vysvetlenie skratiek z tabuľky 2.1

- NOP – žiadna operácia,
- CBR – obnovenie dát („refresh“),
- MRS – nastavenie mód registru,
- ACT – aktivácia banky,

- RD – čítanie,
- WR – zápis,
- PRE – zablokovanie banky,
- RB – adresa riadku banky,
- CB – adresa stĺpca a banky,
- B – adresa banky,
- maska – maska dát,
- 0 – signál má hodnotu log. 0,
- 1 – signál má hodnotu log. 1,
- x – ľubovoľná hodnota signálu.

Príkaz	**Hodnoty signálov**				
	nSDRAS	**nSDCAS**	**nWE**	**MA**	**DQM**
NOP	1	1	1	x	x
CBR	0	0	1	x	x
MRS	0	0	0	x	x
ACT	0	1	1	x	RB
RD	1	0	1	0	CB
WR	1	0	0	maska	CB
PRE	0	1	0	x	B

Tabulka 2.1: Základné príkazy generované radičom pamäte pre SDRAM pamäť

Princíp činnosti

Zobrazenie priebehu signálov pri zápise, čítaní a zablokovaní banky je na obrázku 2.2. Príkazy sa dekódujú podľa dekódovacej tabuľky 2.1. Vysvetlenie významu jednotlivých časov a synchronizácie signálov je v kapitole 2.1. Pri príkaze zápisu do pamäte alebo zablokovaní banky pamäte je potrebné deaktivovať signál nWE. Pred samotným čítaním alebo zápisom je inicializovaný register módu, kde je definovaná veľkosť dát a čas tCL (kapitola 2.1).

Čítanie z pamäte SDRAM:
Čítanie z pamäte môžme rozdeliť do troch fáz:

1. fáza – radič pamäte príkazom ACT aktivuje banku. Po adresnej zbernici sa posiela okrem adresy banky, ktorá sa má aktivovať, aj adresu riadku z ktorého sa bude čítať.
2. fáza – radič pamäte príkazom RD zaháji operáciu čítania dát z pamäte. Po adresnej zbernici sa posiela adresa stĺpca, od ktorej sa majú dáta čítať.

3. fáza – dáta budú prístupné na dátovej zbernici za čas tCL. Maska dát má pri čítaní hodnotu 0. V tejto fáze je možné poslať príkaz uzamknutia banky, pričom zamknutie banky nastane po skončení prenosu dát.

Zápis do pamäte SDRAM:
Zápis do pamäte môžme rovnako rozdeliť do troch fáz:

1. fáza – radič pamäte príkazom ACT aktivuje banku. Po adresnej zbernici sa posiela okrem adresy banky, ktorá sa má aktivovať, aj adresa riadku, na ktorý ideme zapisovať.

2. fáza – radič pamäte príkazom WR spustí zápis dát do pamäte. Po adresnej zbernici sa posiela adresa stĺpca, od ktorého sa má zapisovať. Radič pamäte taktiež určuje aj masku dát.

3. fáza – zotavovacia fáza. Začína ukončením zápisu a končí zablokovaním banku.

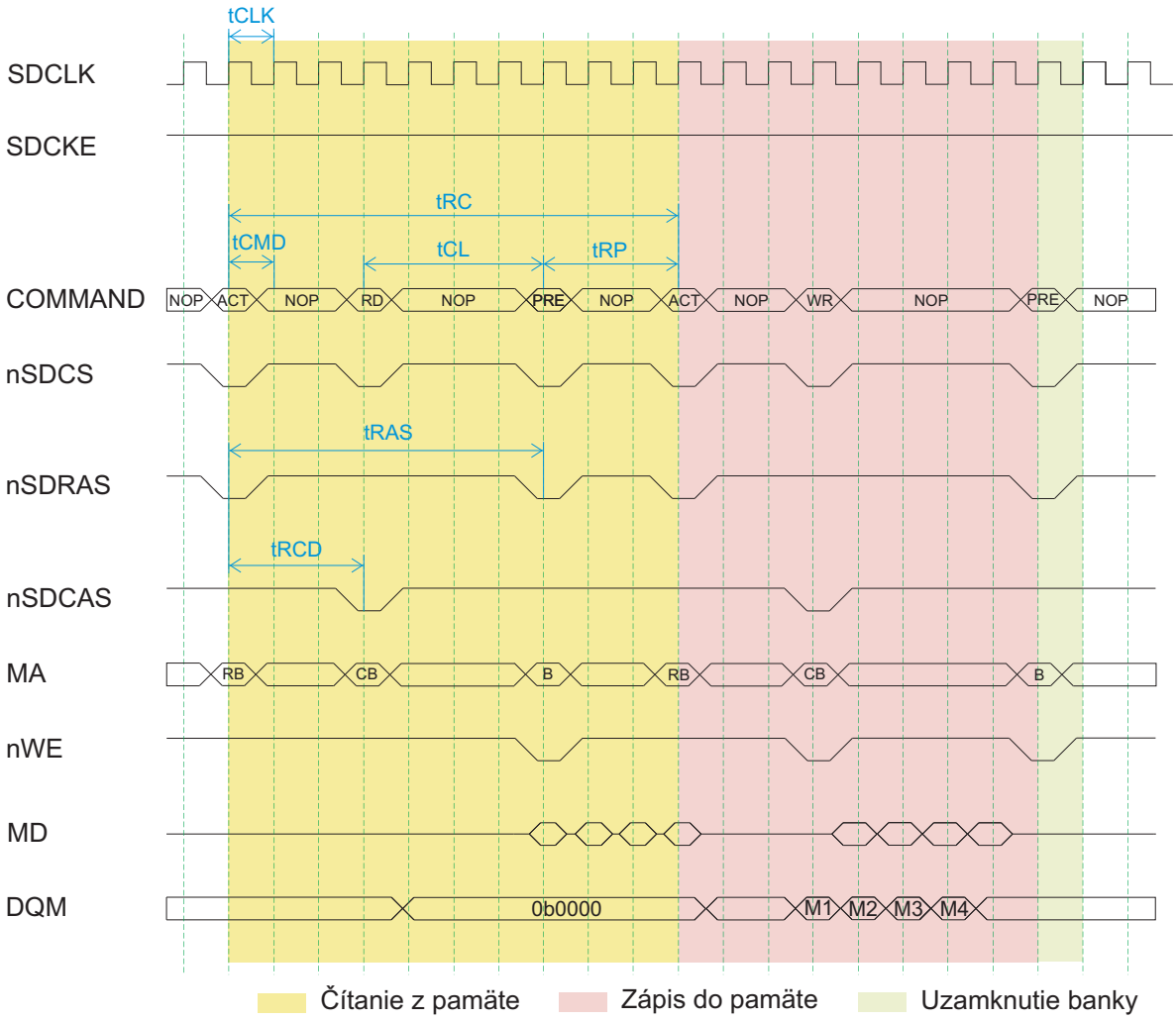
Zablokovanie banky pamäte:
Zablokovanie banky pamäte je príkazom PRE. Po adresnej zbernici sa posiela adresa banky, ktorá sa má zablokovať.

Časovanie signálov

Grafické zobrazenie signálov je na obrázku 2.2. Na tomto obrázku sú definované časy, ktoré sú potrebné na správne zosynchronizovanie signálov, pričom význam jednotlivých časov je nasledovný:

- **tCLK** – trvanie jedného hodinového impulzu,
- **tRC** – čas, za ktorý je banka opäť aktivovaná,
- **tCMD** – trvanie príkazu,
- **tCL** – CAS čakacia doba – doba, za ktorú budú prístupné dáta na dátových vodičoch,
- **tRP** – doba, po ktorej je možné banku opäť aktivovať,
- **tRAS** – doba, po ktorú je banka aktivovaná,
- **tRCD** – oneskorenie signálu nSDCAS za signálom nSDRAS.

Pri určovaní dĺžky týchto časov je dobré sa riadiť manuálom od výrobcu pamäte. Ak nastavíme tieto časy kratšie, ako je minimum definované výrobcom, tak pamäť nemusí pracovať správne. Nastavením týchto časov na doby oveľa dlhšie, ako sú definované výrobcom, sa zníži rýchlosť práce pamäte. Okrem spomenutých signálov je potrebné ešte definovať aj dobu, po ktorej má nastať obnova dát. Túto dobu je taktiež dobré nastaviť podľa manuálu. Pri častom obnovovaní dát znižujeme rýchlosť pamäte. Pri menej častom obnovovaní je možné, že dáta budú stratené. Je dôležité určiť správne optimum načasovania týchto signálov pre rýchlu a spoľahlivú činnosť pamäte.



Obrázek 2.2: Časovanie signálov SDRAM pamäte

2.2 Flash pamäte

Vysvetlenie skratky názvu pamäte: ROM – Read-Only Memory je podobne ako RAM, pamäť s prístupom na ľubovoľnú pamäťovú bunku. ROM narozdiel od RAM je energeticky nezávislá a umožňuje iba čítanie. EEPROM – Electrically Erasable Programmable Read-Only Memory je elektrický mazateľná a elektricky zapisovateľná ROM. Flash pamäť, alebo celým názvom Flash-EEPROM, je narozdiel od EEPROM organizovaná po blokoch, ktoré je možné samostatne programovať. Výhodou flash pamätí oproti pevným diskom je ich vyššia rýchlosť čítania a väčšia mechanická odolnosť. Životnosť flash pamätí je 10 000 až 1 000 000 mazaní. Popísané princípy, rozdelenie a bližšie informácie je možné nájsť na [1] a [3].

Podľa typu pamäťového prvku môžme rozdeliť tieto pamäte na:

- **NAND** flash pamäte,
- **NOR** flash pamäte.

Podľa toho, či pamäť využíva hodinový signál, ich môžme rozdeliť na:

- **asynchrónne** flash pamäte,
- **synchrónne** flash pamäte.

NAND versus NOR flash pamäte

- **NAND flash** – pamäťovú bunku tvoria hradla NAND. Je rýchlejšia pri mazaní a zápise ako NOR flash. Oproti NOR flash, pamäťová bunka zaberá menšiu plochu a má 10 násobne vyššiu životnosť. Ďalšou výhodou sú menšie výrobné náklady. Nevýhodou je pomalý prístup do pamäte a pamäť je možné čítať len po blokoch. Tieto typy flash pamätí sa využívajú napr. ako náhrada pevných diskov.
- **NOR flash** – pamäťovú bunku tvoria hradla NOR. Je pomalšia pri zápise a mazaní ale rýchlejšia pri čítaní ako NAND flash. Po vymazaní ma každý bit hodnotu 1. Pri programovaní sa nastavujú potrebné bity na hodnotu 0. Tieto pamäte sa typicky používajú ako sklady pre kód programu a pre programy, ktoré sa vykonávajú priamo z tejto pamäte, napr. v mobilných telefónoch, PDA, vo vstavaných systémoch.

Princíp práce s FLASH pamäťou

Flash pamäť je ovládaná pomocou príkazov. Skladá sa z blokov. Blok je skupina pamäťových buniek, ktoré sa mažú spoločne. Má vstupné a výstupné vyrovnávacie pamäte, jednotku na prijímanie a vykonávanie príkazov. Má stavový register a konfiguračný register a jednotku riadenia zápisu. Taktiež každý blok má svoj stavový register. Všetky operácie s pamäťou sa robia na základe príkazov, pričom softvér musí zabezpečiť, aby bol najprv do flash pamäte odoslaný príkaz a až následne dáta.

Čítanie z pamäte môže byť po pamäťových bunkách, po najmenšej adresovateľnej jednotke, kedy je čítanie najpomalšie. V tomto prípade sa zakaždým posiela adresa pamäťovej bunky a dáta sú prístupné s oneskorením prvého prístupu do pamäte. Dáta môžu byť načítavané aj po stránkach, kedy sa dáta načítajú do stránkovacej vyrovnávacej pamäte. Veľkosť dát, ktoré sa načítajú, je rovná veľkosti stránkovacej vyrovnávacej pamäte. V tomto prípade je hodnota prvej adresovanej pamäťovej bunky prístupná s oneskorením prvého prístupu do pamäte a hodnota každej ďalšej bunky s oneskorením ďalšieho prístupu do pamäte. Oneskorenie ďalšieho prístupu je menšie ako pri prvom prístupe, keďže sa neprenáša adresa pamäťovej bunky. Tento spôsob čítania je u väčšiny pamätí prednastavený pri reštarte alebo po zapnutí pamäte. Tieto dva spôsoby sú asynchrónne, nevyužívajú hodinový signál a sú dostatočne výkonné pre väčšinu aplikácií. Ak je flash pamäť synchrónna, tak je možné čítanie po blokoch. Princíp je podobný ako u SDRAM. Veľkosť bloku je definovaná v konfiguračnom registri. Tento spôsob čítania je najrýchlejší. Bližšie vysvetlenie spôsobov čítania flash pamätí je možné nájsť v [2].

Zápis do pamäte sa u asynchrónnej aj synchrónnej flash vykonáva iba po jednotkách o veľkosti jednej pamäťovej bunky. Zápis väčšieho množstva dát je možný pomocou zapisovacej vyrovnávacej pamäte, kedy sa dáta zapíšu do zapisovacej vyrovnávacej pamäte, a následne sa zaháji ich zápis do flash pamäte. Softvér musí pri zapisovaní zabezpečiť skontrolovanie flash pamäte, či nie je zaneprázdnená jednotka na riadenie zápisu. Po zápise je vhodné skontrolovať stavový register, či zápis prebehol v poriadku.

Flash pamäť okrem čítania a zápisu umožňuje **mazanie pamäte po blokoch, zamknutie bloku a odomknutie bloku**. Ak je blok zamknutý, tak nie je možné doňho

zapisovať alebo ho vymazať. Okrem vymenovaných základných vlastností môže mať flash pamäť aj ďalšie, ktoré závisia na výrobcovi.

Po vykonaní operácie zápisu, mazania alebo zamykania, nie je možné z pamäte čítať, pokiaľ príkazom softvér neuvedie pamäť do režimu, ktorý to umožní.

Kapitola 3

Popis Colibri XScale PXA270

Obrázek 3.1: Modul Colibri XScale PXA270 [17]

Colibri XScale PXA270, ktorý je na obrázku 3.1, je počítačový modul so sbernicou typu SO-DIMM a procesorom Intel XScale PXA270. Podrobný popis modulu je v [17]. Základná charakteristika:

- Procesor PXA270 312 / 520 MHz.
- Pamäť tvorí 64MB SDRAM a 32MB flash.
- Rozhrania: 32bitová zbernica, PCMCIA, LCD, Touch screen, Audio I/O, I2C, CMOS/CCD, SPI, SDCard, Memory Stick, 85GPIOs, USB host / device, 100 MBit Ethernet.
- Podporované operačné systémy: WinCE, Enbedded Linux Kernel 2.4. a 2.6., QNX.

3.1 Popis vývojovej dosky

Modul Colibri XScale PXA270 bol pri tvorbe firmweru umiestnený vo vývojovej doske, ktorá je na obrázku 3.2. Podrobný popis dosky je v [16]. Vývojová doska ponúka nasledujúce rozhrania:

- zbernicu s vyvedenými GPIO konektormi.
- 32bit CPU zbernicu pre PXA 270.
- užívateľské rozhrania: 1 x TFT, 1 x VGA, 2 x PS/2, Audio I/O, 10 prepínačov, 8 led diód.
- rozhrania pre pamäťové karty: 1 x SD-Card, 1 x CF.
- komunikačné rozhrania: 1 x USB Host, 1 x USB Host/Device, 10/100 MBit Ethernet, 2 x RS232, IrDA, CAN.

Obrázek 3.2: Vývojová doska pre modul Colibri XScale PXA270 [16]

3.2 Procesor Intel XScale PXA270

Podrobný popis jadra procesoru je v [8].

Architektúra

- sedem až osem úrovňová „superpipline“ RISC technológia, ktorá umožňuje vysokú rýchlosť procesoru pri nízkej spotrebe.
- možnosť dynamicky meniť napätia a frekvenciu za chodu modulu, čo umožní nastaviť optimálný výkon a spotrebu.
- manažment výkonu umožnuje možnosti šetrenia spotreby prepnutím sa do pohotovostného („standby“) režimu, režimu nečinnosti („idle“) alebo spánku („sleep“).
- 32 kByte vyrovnávacia pamäť pre inštrukcie.
- 32 kByte vyrovnávacia pamäť pre dáta.
- 32položkový manažment pamäte inštrukcií.
- 32položkový manažment pamäte dát.
- jednotka monitorovania výkonu.
- debugovacia jednotka s 256 položkovým zásobníkom histórie trasy umožňuje ladenie programu.
- 32bitové rozhranie medzi jadrom procesoru a coprocesormi.
- 8položková zapisovacia vyrovnávacia pamäť umožňuje jadru procesora pokračovať vo vykonávaní program, pokiaľ sú dáta zapisované do pamäte.

Radič pamäte

Radič pamäte je založený na UMA architektúre, kde všteky pamäťové zariadenia zdieľajú spoločný adresný priestor a dátovú zbernicu. Radič pamäte tvoria štyri základné bloky: dynamická pamäť, statická pamäť, rozhranie pre pamäťové karty a rozhranie pre zariadenie

so schopnosťou riadiť externú zbernicu. Každý blok má vyhradené vlastné piny.

Vlastnosti:

- rozhranie pre 4 partície pamäte SDRAM do veľkosti 1 GByte.
- rozhranie pre 6 partícií flash pamätí do veľkosti 384 MByte, pričom 4 partície môžu byť synchrónne.
- podpora 1.8V JEDEC LP-SDRAM operácií na 104 MHz.
- umožňuje uviesť SDRAM pamäť do režimu seba-obnovovania. Tento režim sa využíva v pohotovostnom alebo spánkovom režime alebo pri zmene frekvencie hodinového signálu.
- poskytuje signály a ovládanie pre DMA prenosy.
- poskytuje tri nezávislé výstupne hodinové signály, ktoré môžu mať rovnakú, polovičnú, alebo štvrtinovú hodnotu zo vstupného hodinového signálu.

Radič DMA

DMA – Direct Memory Access slúži na prenos dát medzi zariadeniami pri minimálnej účasti procesora. Procesor iba sprostredkuje prenos dát a počas prenosu môže pokračovať vo svojej predchádzajúcej činnosti. Po skončení prenosu môže byť procesor pomocou prerušenia informovaný o výsledku operácie.

Zariadenia, medzi ktorými je možné realizovať DMA prenos, môžme rozdeliť do týchto skupín:

- interná pamäť – sem patrí interná SRAM.
- externá pamäť – napríklad SDRAM, flash.
- interné periferné zariadenie IPZ – napríklad AC 97, UART, RTC.
- externé periférne zariadenie EPZ.

DMA prenos môže byť v dvoch režimoch:

1. flow-through režim je režim, kedy prenos dát prebieha cez vyrovnávaciu pamäť radiča DMA.
2. fly-by režim je režim, kedy prenos dát prebieha mimo vyrovnávaciu pamäť radiča DMA.

Možnosti využitia DMA prenosov:

- umožňuje prenos dát z pamäte do pamäte, z IPZ do pamäte a naopak. Pamäť môže byť externá alebo interná. Prenos dát je možný iba vo flow-thourgh režime.
- umožňuje prenosy z maximálne troch EPZ do IPZ, EPZ, externej alebo internej pamäte a naopak. Prenos dát môže byť v režime flow-trought. Ak má EPZ schopnosť riadiť externú zbernicu, prenos môže byť v režime fly-by.

Základné vlastnosti :

- veľkosť vyrovnávacej pamäte DMA radiča je 32 Bytov.
- má 32 DMA kanálov rozdelených do štyroch skupín podľa priority.
- umožňuje obsluhu 63 požiadavkov od IPZ a obsluhu troch požadavkov od EPZ.
- každý z 32 kanálov môže pracovať v režime s popisovačmi alebo bez popisovačov alebo s popisovačmi v špeciálnom móde. Popisovač definuje spôsob práce jedného DMA kanálu. V popisovači je uložená zdrojová adresa a cieľová adresa, veľkosť prenášaných dát, nastavenie prerušení a požiadavkov, prípadná inkrementácia zdrojovej alebo cieľovej adresy.
- Maximálna veľkosť prenášaných dát pomocou jedného popisovača je 8 kByte-1. Je možné použiť zreťazovanie popisovačov v prípade potreby prenášať väčšie množstvo dát. Popisovač v špeciálnom móde umožňuje automatickú kontrolu prenesených dát a umožňuje výber jedného z dvoch možných ďalších popisovačov.
- programovateľná veľkosť prenášaných blokov a programovateľná veľkosť šírky prenášaných dát podľa pripojeného periférneho zariadenia.

Ďalšie vlastnosti

- kompatibilný s ARM V5, ale neposkytuje hardvérovú podporu inštrukcií s plávajúcou čiarkou.
- 256 kByte interná pamäť typu SRAM.

3.3 Popis SDRAM pamäte

- výrobné označenie osadených SDRAM je K4M561633G a podrobný popis je v [15].
- jeden pamäťový čip tvoria 54 FBGA
- Colibri XScale PXA270 obsahuje dve takéto SDRAM pamäte, celková veľkosť pamäte je 2 x 32 MByte = 64 MByte.
- pamäť je rozdelená do 4 bánk, každá banka má veľkosť 4 194 304 Wordov, pričom 1 Word = 16 bitov. Z toho vyplýva veľkosť SDRAM: 4M * 16bits * 4banky = 32 MBytov. Najmenšia adresovateľná jednotka je 1 Word.
- presné označenie pamäte je LP-SDRAM. LP-SDRAM znamená že sú to nízko príkonové pamäte. Pamäť je napájana 3,3V napätím.
- keďže radič pamäte má 32 dátových signálov a SDRAM pamäť má 16bitový dátový vstup, tak tieto dve SDRAM pamäte majú riadiace a adresné vodiče zapojené paralelne. Dátové vodiče sú rozdelené. Pričom prvých 16 bitov z 32bitových zapisovaných dát sa zapíše do jedného čipu a zvyšných 16 bitov do druhého čipu.
- umožňuje maskovanie dát a vlastnú seba-obnovu dát.
- dĺžka cyklu obnovy dát jednej banky pamäte je 64 ms.

- pamäť má programovateľnú veľkosť prenášaného bloku dát a programovateľné čakacie doby.

3.4 Popis FLASH pamäte

- výrobné označenie osadenej flash pamäte 28F128 a podrobný popis je v [12].
- Colibri XScale PXA270 obsahuje dve takéto flash pamäte, celková veľkosť pamäte je 2 x 16 MByte = 32 MByte.
- keďže radič pamäte má 32 dátových signálov a flash pamäť má 16 bitový dátový vstup, tak tieto dve flash pamäte majú riadiace a adresné vodiče zapojené paralelne. Dátové vodiče sú rozdelené. Pričom prvých 16 bitov z 32bitových zapisovaných dát sa zapíše do jedného čipu a zvyšných 16 bitov do druhého čipu, podobne ako u SDRAM pamätí.
- je rozdelená do 128KBytových blokov.
- je asynchrónna.
- má 32Bytovú zapisovaciu vyrovnávaciu pamäť.
- je mazateľná po blokoch, umožňuje zamykanie pamäte po blokoch.
- umožňuje prerušenie mazania alebo programovania.
- umožňuje zápis 1 Wordu = 16 bitov alebo celej zapisovacej vyrovnávacej pamäte.

Kapitola 4

Návrh firmware pamäťovej zbernice a k nej pripojeným pamätiam

Táto kapitola sa venuje praktickej realizácií obslužného firmware. Spôsob, akým vhodne implementovať firmware a aplikácie, pre vstavané systémy je popísaný v [13]. Konkrétne je v tejto kapitole popísaná inicializácia radiča pamäte, nastavenia SDRAM a flash pamäte, obslužné funkcie pamäte flash, inicializácia a spustenie DMA prenosu, vytvorenie a práca s dynamickou pamäťou.

Firmware je implementovaný v jazyku ISO C, ktorý je popísaný v [10], a jazyku Assembler, ktorý je popísaný v [9]. Pre linker je implementovaný súbor príkazov linker.cmd, vysvetlenie príkazov je popísané v [7].

Pri implementácii bolo nutné použiť aj manuál k jadru procesoru [8], manuál k modulu Colibri XScale PXA270 [11], manuál k vývojovej doske [16], manuál k flash [12] a k SDRAM pamäti [15].

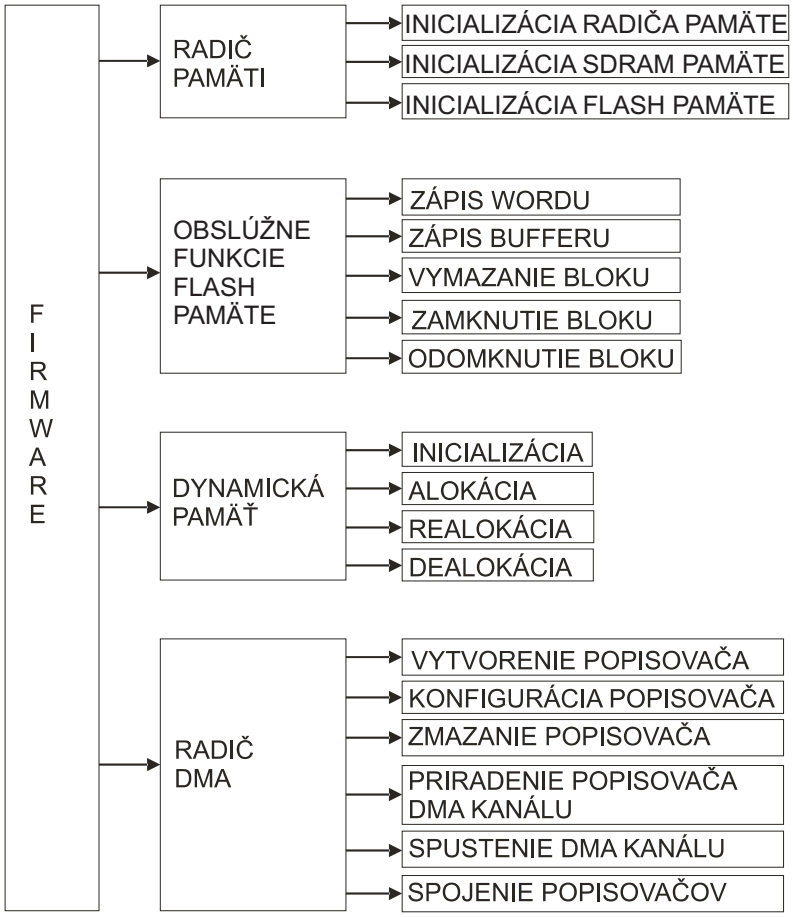
Firmware aj demonštračná aplikácia (viď kapitola 5) boli implementované pod vývojovým prostredím CodeWarior. Súčasťou práce je aj mem_sys_PXA270.mcp súbor, ktorý definuje projekt pre CodeWarior. Na preloženie zdrojového kódu je potrebný Intel XScale prekladač a jeho knižnice.

4.1 Zoznam súborov tvoriacich firmware

- **start_asm.s, start.h** – inicializácia radiča pamäte, vytvorenie zásobníkov, kopírovanie a spustenie zvyšného kódu programu z flash do SRAM pamäte.
- **flash.c, flash.h** – obslužné funkcie flash pamäte.
- **mem_mng.c, mem_mng.h** – vytvorenie a správa dynamickej pamäte.
- **dma.c, dma.h** – funkcie potrebné na vykonávanie DMA prenosov.
- **types.h** – definícia dátových typov.
- **iomap.h, iomap_asm.h** – I/O mapa.
- **linker.cmd** – command file pre linker, definuje adresy, kam sa má aplikácia uložiť.

- **common.h** – zoznam návratových hodnôt a konštánt používaných vo viacerých moduloch.

4.2 Bloková schéma



Obrázek 4.1: Bloková schéma firmware

4.3 Radič pamäte

Tento blok inicializuje registre radiča pamäte pre správnu funkčnosť SDRAM pamäte, flash pamäte a ich komunikácie s radičom pamäte. Implementácia je napísaná v jazyku Assembler z dôvodu jednoduchej a rýchlej inicializácie registrov.

Nastavenie hodinových signálov: Hodinový signál generovaný procesorom pre radič pamäte má frekvenciu 204 MHz. Radič pamäte umožňuje z tohto signálu generovať tri nezávislé hodinové signály, dva sú určené pre pamäte SDRAM a jeden pre pamäte flash. Maximálna možná frekvencia hodinového signálu osadenej SDRAM je 111 MHz, preto nastavený hodinový signál generovaný radičom pamäte pre pamäť SDRAM je polovičný zo vstupného hodinového signálu, teda 104 MHz. Pri nastavovaní hodinového signálu sú všetky partície SDRAM deaktivované, po zmene hodinových signálov nasleduje čakacia

slučka kvôli ustáleniu hodinových signálov a následne aktivácia partícii. Pamäť flash je asynchrónna a pre jej činnosť nieje potrebný hodinový signál.

Nastavenie hierarchie pamäte SDRAM: Colibri XScale PXA270 umožňuje pripojenie 4 x 64 MByte SDRAM v 256 MByte režime alebo 4 x 256 MByte SDRAM v 1024 MByte režime. V našom prípade je pripojená jedna 64 MBytová pamäť, preto je nastavený režim 256 MByte. Pripojená pamäť má 4 banky a veľkosť jednej banky je 4 194 304 Words (1 Word = 16 bitov). Z tohto dôvodu je nastavený počet bánk na 4, 13 bitová adresa riadku a 9 bitová adresa stĺpca. Jedna pamäťová bunka je presne adresovaná adresou banky, riadku a stĺpca. Ak máme 13 bitovú adresu riadku a 9 bitovú adresu stĺpca tak to je 2^{13} x 2^{9} = 4194304 kombinácií, čo presne pokryje potrebný adresný priestor pamäte.

Nastavenie časovania pamäte SDRAM: Radič pamäte umožňuje nastaviť tieto časy: tRP, tRCD, tRASMIN, tRC. Grafické zobrazenie týchto časov je vyznačené na obrázku 2.2. V manuále SDRAM [15] sú definované tieto hodnoty týchto časov: tRP = 18 ns, tRCD = 18 ns, tRASMIN = 50 ns, tRC = 68 ns a CL („Cas Latency“) môže mať hodnotu 3, 2 alebo 1. Pri hodinovom signále 104 MHz môžme tieto časy prerátať na počet hodinových impulzov – clk, pričom 1 clk = 1/(104∗106) = 9,615 ns: tRP = 2 clk, tRCD = 2 clk, tRASMIN = 6 clk, tRC = 8 clk. Na základe týchto výpočtov a možnosti nastavenia radiča pamäte je nastavená táto možnosť časovania: tRP = 2 clk, CL = 2, tRCD = 2 clk, tRASMIN = 5 clk, tRC = 8 clk. Nastavenie doby, po ktorej nastáva obnovenie dát pamäte SDRAM – DRI: Táto doba definuje dĺžku cyklu v clk kedy nastáva obnova dát v celej SDRAM. Výpočet sa vykonáva podľa nasledujúceho vzorca:

$$DRI = \left(\frac{refresh\_period}{rows} * MEM\_CLK - 31 \right) / 32$$

Vysvetlenie premenných:

- refresh_period / rows je doba za ktorú sa obnovia dáta na danom počte riadkov: $64 * 10^{-3}$ s / 4096 riadkov.
- MEM_CLK je frekvencia hodinového signálu generovaného radičom pamäte pre pamäť: 104 MHz.

Výpočtom získame že DRI = 50 clk.

4.4 Obslužné funkcie flash pamäte

Jednu z častí obslužného firmware tvoria funkcie, ktoré umožňujú prácu s flash pamäťou. Funkcie sú implementované na základe vývojových diagramov ktoré sú popísané v manuále k pamäti [12].

Pomocou týchto funkcií je možné zapísať do flash pamäte dáta o veľkosti 2 x 1 Word = 2 x 16 bitov alebo dáta o veľkosti 2 x veľkosť zapisovacej vyrovnávacej pamäte, tj. 2 x 32 Bytov. Tieto veľkosti sú dvojnásobné, pretože pamäťové čipy sú dva a 32bitové dáta z radiča pamäte sa rozdeľujú po 16bitov na tieto dva čipy.

Ďalšia z funkcií umožňuje mazanie jedného alebo viacerých blokov pamäte. Funkcii sa pomocou parametrov predá počiatočná a cieľová adresa. Následne funkcia vymaže bloky v rozsahu od bloku, ktorý obsahuje počiatočnú adresu, po blok, ktorý obsahuje cieľovú adresu.

V tomto module sú implementované aj funkcie, ktoré umožňujú zamknutie a odomknutie jedného alebo viacerých blokov pamäte. Funkciám sa pomocou parametrov predá počiatočná a cieľová adresa a bloky sa zamykajú / odomykajú v rozsahu adries, podobne ako pri mazaní.

V každej z týchto funkcií sú použité aj testy, ktoré overujú, či je pamäť napájaná dostatočne veľkým napätím, či nie je zaneprázdnený zapisovací mechanizmus, či nenastala chyba pri vykonávaní príkazu a správnosť adries. Pri zápise a mazaní je daný blok testovaný, či nie je zamknutý. Pred zahájením práce s flash pamäťou sú všetky prerušenia deaktivované a po skončení naspäť aktivované. Po ukončení každej z týchto operácií je pamäť nastavená naspäť do stavu, ktorý umožňuje jej čítanie.

4.5 Manažér dynamickej pamäte

Tento modul slúži na vytvorenie, správu a prideľovanie dynamickej pamäte. Princíp správy a prideľovania dynamickej pamäte je následovný: každý blok dynamickej pamäte, či už alokovaný alebo nealokovaný začína stavovým slovom o veľkosti 16bitov. Stavové slovo definuje či je daný blok alokovaný alebo nie a tiež veľkosť bloku.

Pri počiatočnej inicializácii tvorí dynamickú pamäť len jeden blok, pričom jeho stavové slovo nesie informáciu, že tento blok nieje alokovaný a tiež informáciu o veľkosti bloku, ktorá je totožná s veľkosťou celej dynamickej pamäte. Pri inicializácii je potrebné určiť počiatočnú adresu dynamickej pamäte a jej veľkosť.

Pri alokácii pamäte sa prechádza sekvenčne blok po bloku a hľadá sa nealokovaný blok s dostatočnou veľkosťou. Ak sa daný blok nájde tak sa inicializuje jeho stavové slovo. V prípade vzniku nového nealokovaného bloku sa inicializuje aj jeho stavové slovo. Funkcia vracia ukazateľ na alokované miesto. Ukazateľ ukazuje až za stavové slovo. Pri dealokácií sa zmení bit v stavovom slove dealokovaného bloku, ktorý určuje, či je pamäť alokovaná/dealokovaná. V prípade že vzniknú dva susedné nealokované bloky, tak sa spoja do jedného bloku.

Tento modul obsahuje okrem funkcií pre inicializáciu, alokáciu a dealokáciu aj funkciu pre realokáciu pamäte. Taktiež funkciu pre alokáciu a realokáciu pamäte s tým, že vrátený ukazateľ bude ukazovať na adresu zarovnanú na 16Bytov. Táto funkcia je implementovaná kvôli DMA popisovačom, ktoré musia byť umiestnené na adrese zarovnanej na 16Bytov.

4.6 Radič DMA

V tomto module sú implementované funkcie potrebné pre DMA prenosy.

Implementovaná je funkcia pre vytvorenie popisovača. Táto funkcia zabezpečí alokovanie popisovača a nastavenie zdrojovej adresy, odkiaľ sa budú dáta prenášať, nastavenie cieľovej adresy, kam sa budú dáta prenášať a veľkosť prenášaných dát. Taktiež je implementovaná funkcia pre zmazanie popisovača.

Implementovaná je funkcia pre nastavenie správania sa popisovača. Tá definuje činnosť popisovača, či má inkrementovať zdrojovú alebo cieľovú adresu, aké prerušenia sa počas alebo po skončení prenosu majú generovať a či má čakať na požiadavok pred samotným prenosom dát.

Implementovaná je aj funkcia pre zreťazovanie popisovačov. Tú je možné použiť v prípade, ak chceme okamžite po ukončení činnosti jedného popisovaču zahájiť činnosť ďalšieho, alebo ak maximálna možná veľkosť prenášaných dát pre jeden popisovač nestačí.

Ďalej je implementovaná funkcia pre pridelenie DMA popisovača DMA kanálu a funkcia pre spustenie prenosu na danom DMA kanále. V prípade, ak na danom kanále už beží DMA prenos, tak sa čaká na skončenie prenosu a až následne sa pridelí popisovač DMA kanálu.

Kapitola 5

Demonštračná aplikácia

5.1 Popis demonštračnej aplikácie

Súčasťou práce je aj demonštračná aplikácia. Táto aplikácia obsahuje užívateľské funkcie, ktorých prvoradou úlohou bolo overenie fukčnosti firmware a druhoradou úlohou je praktická ukážka práce s implementovaným firmware.

Aplikácia beží v prázdnej nekonečnej slučke, pričom riadená je pomocou prerušení. Aplikácia je uložená v pamäti flash, pričom jej počiatočná adresa je definovaná v súbore linker.cmd. Pri spustení aplikácie sa inicializuje radič pamäte a vytvoria sa zásobníky. Zvyšný kód aplikácie sa kopíruje do pamäte SRAM, odkiaľ sa aj vykonáva. Adresa, kam sa zvyšný kód kopíruje, je rovnako definovaná v linker.cmd.

Spustenie aplikácie je možné napríklad pomocou pripojenia cez JTAG a vhodného softvéru, ktorý umožní nastavenie programového čítača na potrebnú adresu, tj. napríklad bootloader. Ak by sa aplikácia rozšírila o počiatočnú inicializáciu procesora a nahrala sa od adresy 0, na túto adresu je nastavený programový čítač automaticky po reštarte, tak by sa aplikácia spúšťala automaticky pri zapnutí alebo po reštarte hardvéru.

Ovládanie aplikácie je pomocou terminálu. Príkazy z terminálu sa cez SCI prenášajú do procesoru PXA270 a naopak. Modul pre spracovanie dát z terminálu je implementovaný s súbore sci_rec.c a modul pre odosielanie dát po SCI do terminálu je implementovaný v súbore sci_send.c. Manuál k ovládaniu aplikácie je možne zobraziť zadaním príkazu `help`. Manuál k aplikácii je popísaný aj v dodatku 1. V prípade potreby podrobnejšieho vysvetlenia niektorého z príkazov je možné získať podrobnejší popis zadaním príkazu `help [príkaz]`, napr. `help memcpy`.

Pomocou týchto príkazov je možné kopírovať bloky pamäte z flash do SDRAM, z SDRAM do flash, z SDRAM do SDRAM, pričom pri zápise dát do flash pamäte sa využíva modul flash.c. Pri zápise dát do SDRAM pamäte sa dáta prenášajú pomocou DMA kanálov a využíva sa modul dma.c. Je možné zadať viacero cieľových adries, kam sa má blok pamäte kopírovať, čím sa otestuje viacero naraz spustených DMA prenosov. Pri prenášaní väčšieho množstva dát, ako je maximálna veľkosť pre jeden popisovač, sa prenos vykonáva prostredníctvom viacerých zreťazených popisovačov. Všetky popisovače sú alokované a po ukončení prenosu dealokované v dynamickej pamäti, čím je testovaný modul mem_mng.c.

Pre otestovanie správneho kopírovania dát je možné použiť príkaz `memcmp`, ktorý porovná bloky pamäte, alebo pre vizuálne overenie je možné pomocou príkazu `print` vypísať obsah bloku pamäte. Pre otestovanie správnej funkčnosti dynamickej pamäte je možné si zobraziť jej výpis pred a po ukončení prenosu a overiť si správnosť alokácie a dealokácie, tj. správne nastavovanie stavových slov. Týmito funkciami sa taktiež s časti overuje správne nastavenie

radiča pamäte.

Funkčnosť SDRAM a flash pamäte a správne nastavenie radiča pamäte je možné overiť príkazmi `test sdram` alebo `test flash`. Podrobnejšie vysvetlenie princípu testov a popis čo všetko sa testuje, je vysvetlený v kapitole 6.

Demonštračná aplikácia taktiež obsahuje modul pre prácu s LCD displejom lcd.c. V tomto module je implementovaná inicialiazácia displeja, inicializácia DMA popisovačov a funkcie pre vykresľovanie analógových hodín na displej. Tento modul demonštruje ďalšiu praktickú ukážku využita DMA prenosov. Radič LCD displeja má vyhradených sedem špeciálnych DMA kanálov, päť pre tri zobrazované vrstvy, jeden pre hardverový kurzor a jeden pre príkazy LCD. V aplikácii sa posielajú v nekonečnej slučke pomocou jedného DMA kanálu dáta, ktoré sa zobrazujú na základnej vrstve displaya, tj dáta pozadia. Pomocou druhého DMA kanálu sa posielajú dáta ktoré sa zobrazujú v prvej prekrývajúcej vrstve. V tejto vrstve sa zobrazujú hodiny, tj. pozadie hodín, hodinová a minútová ručička. Princíp zobrazovania hodín je na základe dvoch obrázkov, pričom jeden sa aktuálne zobrazuje a druhý sa prekresľuje. Pri prerušení od časovania, ktoré detekuje zmenu času o jednu minútu sa zmení adresa popisovača pre prvú prekrývajúcu vrstvu a vykreslia sa aktualizované hodiny. Následne sa cyklus opakuje. Hodiny je možné aktivovať príkazom `clock enable` a deaktivovať príkazom `clock disable`.

5.2 Zoznam a popis súborov

- **main.c** – main, v ktorom sú volané počiatočné inicializácie a nekonečná slučka.
- **sci_rec.c, sci_rec.h** – spracovanie príkazov z terminálu, volanie funkcií na vykonanie príkazu, vyhodnotenie výsledku operácie.
- **sci_send.c, sci_send.h** – odosielanie dát na terminál.
- **int_asm.s** – tabuľka vektorov prerušení a funkcia ktorá volá obsluhu prerušenia.
- **int.c, int.h** – obsluhy prerušení, implementovaná je obsluha DMA prerušení, obsluha prerušenia pri prijatí dát z terminálu, obsluha časovačov ktoré sa používajú pri flash pamäte a práci s displejom.
- **user.c, user.h** – funkcie ktoré vykonávajú niektoré príkazy z terminálu: výpis nápovedy, spustenie testov, výpis obsahu pamäte, porovnanie obsahu pamäte, kopírovanie bloku pamäte.
- **lcd.c, lcd.h** – inicializácia, aktivácia a deaktivácia LCD displeja. Vytvorenie a inicializácia DMA popisovačov. Vytvorenie základného obrázku a prekrývajúceho obrázku. Prepínanie a prekreslovanie prekrývajúceho obrázku.
- **test_FLASH.c, test_FLASH.h** – implementácia testov pre flash pamäť.
- **test_SDRAM.c, test_SDRAM.h** – implementácia testov pre SDRAM pamäť.

Kapitola 6

Testy SDRAM a flash pamätí

Súčasťou bakalárskej práce je aj testovanie funkčnosti pamätí SDRAM a flash. Obe tieto pamäte sú polovodičové a sú to pamäte s ľubovoľným prístupom a práve na tieto typy pamätí budú nasledujúce testy zamerané. Chyby na pamätiach môžme rozdeliť do dvoch skupín: na trvalé a prechodné. Medzi trvalé chyby patrí napríklad defekt pamäťovej bunky, porucha v spojoch pamäťovej matici, v čítacích alebo zapisovacích zosilňovačoch, v dekodéroch adresy alebo v inej podpornej logike. Prechodné chyby, narozdiel od trvalých, môžu vznikať aj na úplne funkčnom pamäťovom čipe – napríklad nárazom alfa častice, nedodržaním rozsahu prevádzkovej teploty, chybným časovaním signálov. Chyby pamätí je možné testovať bez fyzického zásahu pomocou testovacích vzorov. Testovací vzor je postupnosť čítania a zápisu z pamäte z presne daných adries. Ak by mal testovací vzor odhaliť všetky chyby, tak by jeho zložitosť bola minimálne kvadratická, a časovo veľmi náročná. Preto sa využíva kombinácia viacerých testovacích vzorov s jednoduchšou zložitosťou tj. lineárnou, logaritmickou, poprípade maximálne kvadratickou. Najkvalitnejšie testy, ktoré majú kvadratickú alebo zložitejšiu časovú závislosť, môžu trvať aj niekoľko dní alebo mesiacov, záleží aj na veľkosti testovanej pamäte. Bližšie vysvetlenie a popis možných chýb pamätí je v [6][4][14]. Podrobné vysvetlenie testovacích vzorov je v [6]. Porovnanie úspešností a časovej náročnosti testovacích vzorov je v [4].

6.1 Popis chýb polovodičových pamätí

Chyby na polovodičových pamätiach, ktorými sa budeme v tejto kapitole zaoberať, môžme rozdeliť do troch základných skupín: chyby na pamäťových bunkách, chyby na dekóderi adries a dynamické chyby.

Príklady chýb na pamäťových bunkách:

- trvalá porucha – pamäťová bunka má vždy hodnotu 0 alebo 1.
- prechodová porucha – je to chyba, kedy je pamäťová bunka schopná prejsť z jedného stavu do druhého, ale nie je schopná sa vrátiť do predchádzajúceho stavu. Napr. ak je bunka v stave s hodnotu log. 0 a zmeníme jej stav na stav s hodnotou log. 1, tak táto operácia sa vykoná bez chyby. Ale ak chceme vrátiť bunku do stavu s hodnotu log. 0, tak to už nieje možné.
- väzbová porucha – táto porucha pri zmene hodnoty pamäťovej bunky spôsobí chybu na susednej pamäťovej bunke. Tento typ poruchy má viacero podtypov. Napríklad

inverzná väzbová porucha je taká, že pri zmene hodnoty bunky, sa hodnota susednej bunky invertuje. Idempotentná väzbová porucha je taká, že pri zmene hodnoty jednej bunky, sa hodnota susednej bunky stane fixnou, tj. nedá sa zmeniť. Stavová väzbová porucha je taká, že pri určitej hodnoty je hodnota susednej pamäťovej bunky fixná a pri opačnej hodnote funguje susedná pamäťová bunka správne.

- porucha závislá na poli susedných buniek – podobná ako väzbová porucha, s tým rozdielom, že hodnota pamäťovej bunky a jej porucha je závislá na viacerých okolitých pamäťových bunkách.

Chyby na dekóderi adries:

- žiadna pamäťová bunka nie je sprístupnená pri určitej adrese.
- viacero pamäťových buniek je sprístupnených pri určitej adrese.
- určitá pamäťová bunka nie je sprístupnená pri akejkoľvek adrese.
- určitá pamäťová bunka je sprístupnená pri viacerých adresách.

Príklady dynamických chýb:

- zotavovacie chyby – vznikajú, keď sa pamäťová bunka nestihne dostatočne rýchlo zotaviť a prejsť do nového stavu pred čítaním hodnoty tejto bunky.
- strata dát nespôsobená čítaním alebo zápisom. Napríklad u dynamických pamätí môže nastať strata dát pri zlyhaní signálu pre obnovu dát alebo chybným načasovaním tohto signálu.

6.2 Testovacie vzory

Testovacích vzorov existuje veľmi veľa. Medzi najjednoduchší testovací vzor patrí Ones/Zeros. V tomto teste je celá pamäť zapísaná nulami, následne sa otestuje či su nuly zapísane všade. Potom sa to isté spraví s jednotkami. Zložitosť testu je 4n, pričom n je veľkosť pamäte. Tento test nedokáže odhaliť chyby na dekóderi adries a mnohé ďalšie. Ďalším možným testom je Checkerboard. Je podobný ako Ones/Zeros, len sa najprv sa zapisuje periodický vzor 0101... a v druhom kroku 1010... Tento test ma rovnakú lineárnu zložitosť 4n a dokáže odhaliť už aj niektoré chyby na dekóderoch adries.

Ďalšími testovacími vzormi sú testovacie vzory typu Marching. Pri týchto testoch sa najprv celá pamäť vynuluje, potom sa postupne načítavajú pamäťové bunky, overí sa hodnota aktuálnej bunky, zapíše sa inverzná hodnota. V druhom cykle sa to zopakuje ešte raz, s tým rozdielom, že sa bunky načítavjú od konca pamäte a všetky majú hodnotu jedna a prepisujú sa na hodnotu nula. Týchto testov je viacero typov s menšími rozdielmi v princípe a s rôznou zložitosťou. Medzi tieto vzory patria napríklad: March B, March C, March C-, March Y. Zložitosť je však stále lineárna. Tieto testy dokážu odhaliť chyby na dekóderoch adries, väčšinu chýb na pamäťových bunkách a dynamické chyby. Nedokážu však chybnú pamäťovú bunku lokalizovať.

Medzi najkvalitnejšie testovacie vzory s kvadratickou zložitosťou patria Walking ones/zeros a Galpat. Pri týchto testoch sa vždy po zmene hodnoty jednej pamäťovej bunky testuje celá pamäť. Tieto testy vedia odhaliť chyby na pamäťových bunkách a lokalizovať chybnú pamäťovú bunku, taktiež chyby na dekóderoch adries a dynamické chyby.

V našom prípade sú na testovanie pamäte použité dva testovacie vzory a to March C- a March Y. Týmito vzormi sa dajú testovať pamäťové bunky, spoje v pamäťovej matici, dekódery adries, nastavenie časovani a obnovovacieho času SDRAM pamäte.

March C- Model

Tento test je na implementáciu veľmi jednoduchý. Jeho časová náročnosť je 10n. Dôvod jeho implementácie a použitia sú výsledky úspešnosti odhalenia chýb, ktoré sú uvedené v [4].

Popis testovacieho vzoru March C-:

1. zapíš do každej pamäťovej bunky hodnotu nula.
2. od adresy nula až po koniec pamäte načítavaj hodnotu pamäťovej bunky. Otestuj hodnotu pamäťovej bunky na hodnotu nula a zapíš do nej hodnotu jedna.
3. od adresy nula až po koniec pamäte načítavaj hodnotu pamäťovej bunky. Otestuj hodnotu pamäťovej bunky na hodnotu jedna a zapíš do nej hodnotu nula.
4. od poslednej adresy pamäte až po adresu nula načítavaj hodnotu pamäťovej bunky. Otestuj hodnotu pamäťovej bunky na hodnotu nula a zapíš do nej hodnotu jedna.
5. od poslednej adresy pamäte až po adresu nula načítavaj hodnotu pamäťovej bunky. Otestuj hodnotu pamäťovej bunky na hodnotu jedna a zapíš do nej hodnotu nula.
6. otestuj každú pamäťovú bunku na hodnotu nula.

March Y Model

Tento test je na implementáciu rovnako ako March C- veľmi jednoduchý. Jeho časová náročnosť je 8n. Dôvod jeho implementácie a použitia je, že podľa výsledkov detekcie chýb v [4] má testovací vzor March C- úspešnosť v detekcii prechodovej poruchy na pamäťovej bunke len 0,2% a March Y má túto úspešnosť 100%. Naopak March Y má úspešnosť detekcie idempotentnej väzbová poruchy 50% a March C- 100%, z tohto dôvodu sú implementované obidva testy.

Popis testovacieho vzoru March Y:

1. zapíš do každej pamäťovej bunky hodnotu nula.
2. od adresy nula až po koniec pamäte načítavaj hodnotu pamäťovej bunky. Otestuj hodnotu pamäťovej bunky na hodnotu nula a zapíš do nej hodnotu jedna. Otestuj hodnotu pamäťovej bunky na hodnotu jedna.
3. od poslednej adresy pamäte až po adresu nula načítaj hodnotu pamäťovej bunky. Otestuj hodnotu pamäťovej bunky na hodnotu jedna a zapíš do nej hodnotu nula. Otestuj hodnotu pamäťovej bunky na hodnotu nula.
4. otestuj každú pamäťovú bunku na hodnotu nula.

Praktické použitie testovacích vzorov

V bakalárskej práci je na testovanie SDRAM pamäte použitý testovací vzor March C- a March Y. Najmenšia adresovateľná bunka má veľkosť 8bitov a pri testovaní nadobúda hodnotu 0 alebo 255. Test je možné spustiť príkazom z terminálu `test sdram [počiatočná adresa] [koncová adresa]`, kedy sa testuje iba pamäť v rozsahu adries alebo príkazom `test sdram all` kedy sa testuje celá SDRAM pamäť. Pri testovaní sa testovaná pamäť zmaže, je však možné použiť príkaz na kopírovanie pamäte a dáta si zálohovať.

Na testovanie flash pamäte je použitý upravený March Y testovací vzor, pretože životnosť flash pamätí sa pri mazaní a zapisovaní znižuje. Testovací vzor je upravený z toho dôvodu, že flash pamäť sa dá mazať iba po blokoch a mazaním sa nastavuje pamäťová bunka na hodnotu $2^{32} - 1$. Veľkosť jednej pamäťovej bunky u flash pamäte je 32bitov. Podobne ako u SDRAM sa testovaná pamäť vymaže. Test sa spúšťa rovnako ako pre SDRAM pamäť s tým rozdielom, že druhý parameter je flash.

Popis upraveného testovacieho vzoru March Y pre flash pamäť:

1. vymaž každý blok pamäte.
2. od adresy nula až po koniec pamäte načítavaj hodnotu pamäťovej bunky. Otestuj hodnotu pamäťovej bunky na hodnotu $2^{32} - 1$ a zapíš do nej hodnotu nula. Otestuj hodnotu pamäťovej bunky na hodnotu nula.
3. od poslednej adresy pamäte až po adresu nula načítaj hodnotu pamäťovej bunky. Otestuj hodnotu pamäťovej bunky na hodnotu nula. Po otestovaní celého bloku, daný blok vymaž. Otestuj hodnotu každej pamäťovej bunky v bloku na hodnotu $2^{32} - 1$.
4. otestuj každú pamäťovú bunku na hodnotu $2^{32} - 1$.

Vďaka týmto testom bolo možné následne otestovať správne nastavenie časovania SDRAM pamäte a správne nastavenie doby, po ktorej má nastať obnova dát v SDRAM pamäti. Pri nastavenom časovaní podľa špecifikácii výrobcu, tj: tRP = 2 clk, tRCD = 2 clk, tRASMIN = 6 clk, tRC = 8 clk a pri nastavenej dobe obnovy dát 50 clk fungovala SDRAM bezchybne. Test celej pamäte trval približne 25 minút. Pri zmene časovania signálov smerom nahor, tj. že počet clk bol vyšší, pamäť taktiež fungovala bezchybne, s tým rozdielom že doba trvania testu bola dlhšia o pár sekúnd a teda SDRAM pamäť pomalšia. Pri zmene časovania signálu smerom nadol, tj. na kratšie časy ako sú minimálne stanovené výrobcom, testovanie pamäte vykazovalo chyby u väčšiny pamäťových buniek.

Pri funkčnom nastavení časovania signálov pre SDRAM pamäť a pri zmenšovaní intervalu, kedy nastáva obnova dát pamäte, bolo testovanie pamäte úspešné. Zmenšovanie tohto intervalu spôsobuje častejšie obnovovanie dát a teda na rýchlosť pamäte, jej funkčnosť však bola stále 100% aj pri intervale 1clk. Pri predlžovaní intervalu, kedy nastáva obnova dát pamäte, sa chyby v testoch prejavili až pri intervaloch väčších ako 260 clk.

Kapitola 7

Záver

Úlohou bakalárskej práce bolo podrobne sa zoznámiť s architektúrou Colibri XScale PXA270, analyzovať možnosti využitia DMA prenosov, popísať princípy práce s pamäťami typu flash a SDRAM, navrhnúť a implementovať firmware pre obsluhu pamäťového systému platformy Colibri XScale PXA270, implementovať sady testov pre overenie pamätí SDRAM a flash a vhodným spôsobom demonštrovať funkčnosť implementovaného riešenia.

Všetky body zadania bakalárskej práce boli splnené. Práca obsahuje štúdiu venujúcu sa práci s pamätiami flash a SDRAM. Popisuje architektúru Colibri XScale PXA270 a možnosti použitia DMA prenosov. Súčasťou práce je firmware a demonštračná aplikácia ktorá s vytvoreným firmware pracuje. Demonštračná aplikácia okrem testovania firmware demonštruje praktické použitie DMA prenosov a tiež vykonáva testovanie pamätí.

Firmware obsahuje navyše oproti plánovaným možnostiam aj modul pre vytvorenie a prácu s dynamickou pamäťou. Táto dynamická pamäť sa využíva u popisovačov DMA prenosov. Ďalším rozšírením bakalárskej práce je použitie LCD TFT displeja. Displej bol použitý z dôvodu demonštrácie ďalšej možnosti použitia DMA prenosov a práce s firmware.

Platforma Colibri XScale PXA270 má svojími malými rozmermi a ponúkanými možnosťami široký rozsah použiteľnosti v priemysle na riadenie zariadení a spracovanie informácii. V rozsiahlejších aplikáciach je práca s rozšírenou pamäťou, akými sú flash a SDRAM, potrebná a dôležitá. Implementovaný firmware môže slúžiť ako základ práce s pamäťou pri implementovaní v praxi použiteľných aplikácii.

Literatura

[1] Flash Memory. [cit. 2009-5-3], [online].
URL http://en.wikipedia.org/wiki/Flash_memory

[2] Intel a paměti Flash. november 2003 [cit. 2009-5-7], [online].
URL http://noel.feld.cvut.cz/vyu/scs/prezentace2003/Flash-Intel/

[3] NAND vs. NOR flash technology. rev. february 2002 [cit. 2009-5-3], [online].
URL http://www2.electronicproducts.com/NAND_vs_NOR_flash_technology-article-FEBMSY1-FEB2002.aspx

[4] A., R.: Walking, marching and galloping patterns for memory tests. [cit. 2009-5-4], [online].
URL http://www.eng.auburn.edu/users/agrawvd/COURSE/E7250_05/REPORTS_TERM/Raghuraman_Mem.doc

[5] Bruce, J.; Spencer, N. W.; David, W. T.: *Memory Systems – Cache, DRAM, Disk*. Morgan Kaufman Publisher, 2008, iSBN 978-0-12-379751-3.

[6] Dean, A. R.: *High Performance Memory Testing – Design Principes, Fault Modeling and Self-Test*. Kluwer Academic Publisher, 2003, iSBN 1-4020-7255-4.

[7] Intel Corporation: *Intel Linker – User's Manual*. september 2004, [DVD-ROM].

[8] Intel Corporation: *Intel XScale Core – Developer's Manual*. january 2004, [DVD-ROM].

[9] Intel Corporation: *Intel Assembler For Intel XScale Microarchitecture – Reference Manual*. april 2005, [DVD-ROM].

[10] Intel Corporation: *Intel C++ Compiler – User's Manual*. march 2005, [DVD-ROM].

[11] Intel Corporation: *Intel PXA27x Processor Family – Developer's Manual*. january 2006, [DVD-ROM].

[12] Intel Corporation: *3 Volt Intel StrataFlash Memory*. [cit. 2009-5-2], [online].
URL
http://www.alldatasheet.net/datasheet-pdf/pdf/66047/INTEL/28F128.html

[13] Labrosse; Ganssle; Noergaad; aj.: *Embedded Software – know it all*. Morgan Kaufman Publisher, 2008, iSBN 978-0-7506-8583-5.

[14] P., T.: Trvalé a přechodné chyby pamětí DRAM. rev. 26.6.2008 [cit. 2009-5-2], [online].
URL http://www.root.cz/clanky/trvale-a-prechodne-chyby-pameti-dram/

[15] Samsung Electronics: *K4M561633G - R(B)N/G/L/F Mobile SDRAM*. january 2006 [cit. 2009-5-2], [online].
URL http://www.samsung.com/global/system/business/semiconductor/product/2007/6/11/MobileSDRAM/MobileSDRSDRAM/256Mbit/K4M561633G/ds_k4m561633g.pdf

[16] Toradex: *Colibri Evaluation Board Datasheet Rev. 2.1*. rev. 07-03-07 [cit. 2009-5-2], [online].
URL http://www.toradex.com/@api/deki/files/14/=Colibri_EvalBoard_Datasheet_Rev_2_1_2007-03-07.pdf

[17] Toradex: *Colibri XScale PXA270 Datasheet*. rev. 24-dec-06 [cit. 2009-5-2], [online].
URL http://www.toradex.com/@api/deki/files/29/=Colibri_PXA270_Datasheet_Rev_1.4.pdf

Dodatek A

Obsah DVD

`./Technicka_sprava/` – technická správa (tento dokument) vo formáte .pdf

`./Technicka_sprava/src` – zdrojové súbory pre LaTeX

`./Citovane_manualy/` – citované manuály

`./Prakticka_cast/firmware/` – zdrojové súbory firmware

`./Prakticka_cast/user/` – zdrojové súbory demonštračnej aplikácie

`./Prakticka_cast/mem_sys_PXA270.exe` – spustiteľná verzia aplikácie

`./Prakticka_cast/linker.cmd` – súbor pre linker

`./Prakticka_cast/mem_sys_PXA270.mcp` – súbor pre CodeWarior, možnosť otvorenia aplikácie ako projekt v CodeWariore

Dodatek B

Manuál

Nápoveda: zadaním príkazu `help` sa na termináli zobrazí táto nápoveda:

```
help -- zobrazi sa tato napoveda
help [cmd] -- zobrazi sa napoveda konretneho prikazu
sdram|flash [start addr] [end addr] -- testovanie bloku pamati
test sdram|flash all -- testovanie celej pamati
erase [start addr] [end addr] -- zmazanie blokov flash pamati
lock [start addr] [end addr] -- zamknutie blokov flash pamati
unlock [start addr] [end addr] -- odomknutie blokov flash pamati
memcpy [src addr] [size] [des addr] -- kopirovanie obsahu pamati
memcmp [src addr] [size] [des addr] -- porovnanie obsahu pamati
print [start addr] [size] -- vypis dat z pamati
write [start addr] -- zapise testovacie data do pamati
clock enable -- spusta hodiny na LCD displayi
clock disable -- zastavuje hodiny na LCD displayi
restart -- restart programu
```

Testovanie pamäti:

- POZOR! Táto funkcia testovanú pamäť vymaže!
- Testuje sa funkčnosť SDRAM alebo flash pamäti.

príkaz: `test sdram|flash [start addr] [end addr]`

- Testuje sa pamäť v rozsahu adries start addr až end addr.
- Pri testovaní flash pamäti sa testuje aj celý blok ktorý obsahuje počiatocnú alebo cieľovú adresu.

príkaz: `test sdram|flash [start addr] [end addr]`

- Testuje sa celá SDRAM alebo flash pamäť.

Zmazanie blokov flash pamäti:

príkaz: `erase [start addr] [end addr]`

- Zmažú sa bloky flash pamäti v rozsahu zdrojovej a cieľovej adresy.

Zamknutie blokov flash pamäti:

príkaz: `lock [start addr] [end addr]`

- Zamknú sa bloky flash pamäti v rozsahu zdrojovej a cieľovej adresy.

Odomknutie blokov flash pamäti:

príkaz: `unlock [start addr] [end addr]`

- Odomknú sa bloky flash pamäti v rozsahu zdrojovej a cieľovej adresy.

Kopírovanie obsahu pamäti:

príkaz: `memcpy [src addr] [size] [des addr]`

- Skopíruje sa obsah pamäti o veľkosti size z adresy src addr na adresu des addr.
- Počet cieľových adries môže byť 1 až 5 ale iba v rámci jednej pamäti, tj. buď z flash alebo SDRAM.
- V prípade kopírovanie do flash pamäti nesmie byť zdrojová adresa z flash pamäti!
- V prípade kopírovania do flash pamäti bude koncová adresa zarovnaná na 32bitov.
- Pri kopírovaní do flash pamäti je size vo Wordoch(32bit).
- Pri kopírovaní do SDRAM pamäti je size v Bytoch(8bit).

Porovnanie obsahu pamätí:

príkaz: `memcmp [src addr] [size] [des addr]`

- Porovná sa obsah pamäti o veľkosti size a počiatočnou adresou src addr s pamäťou o veľkosti size a počiatočnou adresou des addr.

Výpis dát z pamäte:

príkaz: `print [start addr] [size]`

- Výpis obsahu pamäť o veľkosti size a s počiatočnou adresou start addr.

Zápis testovacích dát do pamäti:

príkaz: `write [start addr]`

- Zapíšu sa vygenerované dáta o veľkosti 120Bytov na adresu start addr.
- Táto funkcia slúži iba na pomoc pri demonštrácií iných finkcií.

Spustenie a zastavenie hodín na LCD displeji:

príkaz: `clock enable|disable`

- Enable spúšťa a disable zastavuje hodiny na LCD displeji.

Reštart programu:

príkaz: `restart`

- Príkazom sa program reštartuje.