VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

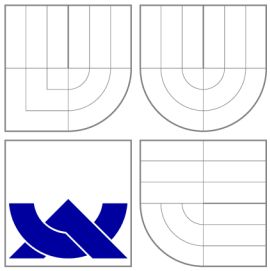
PODPORA DMA PRO RODINU MIKROKONTROLERŮ HCS08

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

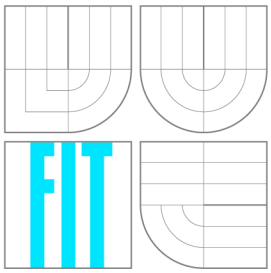
AUTOR PRÁCE
AUTHOR

Bc. ADRIÁN NOVOSÁD

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

PODPORA DMA PRO RODINU MIKROKONTROLERŮ HCS08

DMA SUPPORT FOR HCS08 MICROCONTROLLERS FAMILY

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. ADRIÁN NOVOSÁD
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÁCLAV ŠIMEK
SUPERVISOR

BRNO 2013

Abstrakt

Vestavěné systémy jsou jednoúčelové systémy určené k vykonávaní specifických úloh, díky čemuž je možné je ve fázi návrhu optimalizovat pro co nejmenší cenu a plochu na čipu. Tímto lze zvýšit spolehlivost a výkon výsledného hardwareu. Optimalizace však často vedou k situaci, kdy některé architektury postrádají technologie, které jsou běžné v počítačích pro všeobecné využití. Stejná situace nastala i v rodině mikrokontrolérů HCS08, do které nebyla implementována technologie přímého přístupu do paměti DMA. Tato práce se zabýva návrhem a implementací řadiče DMA a jeho začleněním do architektury mikrokontrolérů HCS08.

Abstract

Embedded systems are dedicated to perform specific tasks, so design engineers can optimize them to reduce the size and cost of the product and increase the reliability and performance. However, result of these optimizations is that some architectures may lack commonly used technologies such as direct memory access (DMA). We may encounter with this situation in family of microcontrollers HCS08. The main theme of this work is to describe a design of DMA controller that can be added into the family of microcontrollers HCS08.

Klíčová slova

mikronotrolér, HCS08, 68hc08, M68HC08, přímý přístup do paměti, DMA, Intel 8237, FPGA, FITkit

Keywords

microcontroller, HCS08, 68hc08, M68HC08, direct memory access, DMA, Intel 8237, FPGA, FITkit

Citace

Adrián Novosád: Podpora DMA pro rodinu mikrokontrolerů HCS08, diplomová práce, Brno, FIT VUT v Brně, 2013

Podpora DMA pro rodinu mikrokontrolerů HCS08

Prohlášení

Prohlašuji, že jsem tuto semestrálni práci vypracoval samostatně pod vedením pana Ing. Václava Šimka.

.....
Adrián Novosád
22. května 2013

Obsah

Kapitola 1

Úvod

Zdokonaľovanie technológií častokrát vedie k tvorbe nových konceptov, ktoré náhrádzajú staré, pomalé, drahé a neefektívne technológie. Počítače už od počiatku svojej existencie boli navrhnované tak, aby umožňovali komunikáciu so vstupno/výstupnými zariadeniami akými sú napríklad úložisko dát alebo obrazovka. Zdokonaľovanie architektúr procesorov viedlo k situácii, kedy výkon procesorov ďaleko prevyšoval rýchlosť akou boli schopné vstupno/výstupné zariadenia komunikovať. Zatiaľ čo niektoré koncepty architektúry počítačov nútili procesor čakať na dokončenie vstupno/výstupnej operácie, iné umožňovali namiesto čakania prevádzať efektívny výpočet.

Dôležitým poznatkom v oblasti návrhu architektúry počítačov bol fakt, že komunikácia so vstupno/výstupnými zariadeniami je častokrát založená práve na prenose dát medzi operačnou pamäťou a vstupno/výstupným zariadením. Tento prenos dát bol vždy realizovaný za plnej účasti procesoru, čo ale viedlo k jeho zbytočnému vyťažovaniu, pretože za predpokladu malých zmien v architektúre počítača by bolo možné tento prenos dát uskutnočniť aj bez účasti procesora.

To viedlo k vzniku technológie priameho prístupu do pamäte DMA (angl. Direct Memory Access), ktorá umožňuje rýchlejší prenos dát medzi vstupno/výstupnými zariadeniami a operačnou pamäťou RAM. Procesor je vďaka tejto technológii v maximálnej možnej miere oslobodený od vybavovania vstupno/výstupných operácii, čím sa docieli zvýšenia celkového výkonu počítača.

V oblasti vnorených systémov hrá najdôležitejšiu úlohu najmä cena a príkon, vďaka čomu vnorené systémy disponujú iba obmedzenými prostriedkami. Inak tomu nebolo ani v rodine 8-bitových mikrokontrolérov HCS08. I keď je táto rodina mikrokontrolérov vybavená nadštandardne vzhľadom na svoju kategóriu, neboli v nej implementované niektoré technológie, ktoré sa bežne používajú v moderných počítačoch alebo v niektorých iných rodinách mikrokontrolérov. Jednou z týchto technológii je práve technológia priameho prístupu do pamäte DMA.

1.1 Cieľ práce

Cieľom tejto práce je navrhnúť a implementovať radič DMA, ktorý v architektúre mikrokontrolérov HCS08 umožní realizovať DMA prenos dát medzi periférnymi zariadeniami a operačnou pamäťou. Podmienkou implementácie je zachovanie kompatibility s inštrukčnou sadou architektúry HCS08.

Kapitola 2

Úvod do problematiky

V poradí druhá kapitola objasňuje základné pojmy, ktoré sú nevyhnutné pre správne pochopenie problematiky tejto práce. Prvá časť kapitoly sa zaoberá stručným popisom architektúry mikrokontrolérov rodiny HCS08 a bližšie rozoberá činnosť periférnych zariadení, pre ktoré bude možné uplatniť DMA prenos dát. Druhá časť kapitoly vysvetľuje princípy priameho prístupu do pamäte a rozoberá konkrétne implementácie používané v osobných počítačoch v zberniciach ISA a PCI. Záver kapitoly sa venuje štúdiu problematiky technológie DMA, pričom sa zameriava najmä na využitie v oblasti vnorených systémov.

2.1 Periférne zariadenie

Periférne zariadenie[3] alebo periféria je v počítačovej technike zariadene, ktoré je pripojené k počítaču a rozširuje možnosti jeho využitia a ovládania. Príkladom týchto zariadení sú napríklad klávesnica, myš, pevný disk, grafická karta, sieťová karta a mnoho ďalších. Periférne zariadenia rozdeľujeme na vstupné (klávesnica, myš), výstupné (monitor, tlačiareň) a vstupno/výstupné (pevný disk).

V osobných počítačoch sa pod periférnym zariadením rozumieme hardware, ktorý je k počítaču pripojený ale nie je jeho integrálnou súčasťou.

2.2 Mikrokontrolér

Mikrokontrolér[3] (jednočipový mikropočítač, niekedy označovaný ako μC, uC alebo MCU) je špeciálny druh integrovaného obvodu obsahujúci jadro procesora, pamäť a programovateľné periférne zariadenia, ktoré slúžia na komunikáciu mikrokontroléra so svojím okolím.

Mikrokontroléry sa používajú najmä v zariadeniach pre automatické riadenie činnosti, v ktorých nezáleží príliš na výpočtovom výkone ale podstatná je cena. Integrácia pamäte a periférnych zariadení spolu s mikroprocesorovou jednotkou na jeden čip ich robí cenovo výhodnými pre použitie vo vnorených systémoch. Okrem nízkej ceny sa vyznačujú aj nízkym príkonom a malou plochou na čipe.

Mikrokontroléry disponujú širokým spektrom rôznych periférnych zariadení, ktoré sa obvykle líšia od bežne používaných v osobných počítačoch. Periférne zariadenia mikrokontrolérov sú integrované na jeden čip spolu s mikroprocesorom a pamäťou, čím sa líšia od definície periférnych zariadení pre osobné počítače. Bežnými perifériami v mikrokontroléroch sú zariadenia pre komunikáciu a generovanie rozličných druhov signálov, ktoré majú

uplatnenie v oblasti vnorených systémov. Často disponujú taktiež programovateľnými časovačmi a komponentami pre vyhodnocovanie a prácu s analógovými veličinami.

2.3 Rodina mikrokontrolérov HCS08

Rodina mikrokontrolérov HCS08[9] (tiež označovaná ako 68HCS08) je široká rodina 8-bitových CISC mikrokontrolérov od firmy Freescale Semiconductor. Predstavuje vysokovýkonný mikrokontrolér s nízkou spotrebou vhodný pre implementácie vo vnorených systémoch. Tento mikrokontrolér vychádza z rady 6800 firmy Motorola Semiconductor a stojí na von Neumannovej architektúre, čiže využíva spoločnú pamäť pre dáta i inštrukcie. Významným rysom architektúry HCS08 sú pamäťovo mapované registre všetkých periférnych zariadení, čo znamená, že všetka komunikácia medzi periférnymi zariadeniami a procesorom prebieha prostredníctvom zápisu alebo čítania z pamäťového systému.

Operačné kódy inštrukčnej sady majú variabilnú dĺžku, čo má za následok to, že inštrukcie bez operandov vyžadujú menej miesta na uloženie v pamäti a rovnako aj potrebujú menej hodinových cyklov k vykonaniu ako inštrukcie s operandmi.

Mikrokontroléry rodiny HCS08 sú dostupné v rozličných modeloch, ktoré sa líšia vo veľkosti dostupnej pamäte, typom a počtom dostupných periférnych zariadení a rovnako aj použitým púzdrom. Jednotlivé modely sú rozdelené do kategórií podľa očakávaného využitia, čím sa líši aj ich taktovacia frekvencia, napájacie napätie a s tým spojený aj príkon a cena.

2.3.1 Programovací model HCS08

Architektúra HCS08 pozostáva z jednej dátovej zbernice zdieľanej medzi mikroprocesorom, pamäťou a periférnymi zariadeniami a rovnako aj jednou adresovou zbernicou, ktorá slúži pre adresáciu miesta v pamäti alebo registrov periférnych zariadení. Dátova zbernica je rozdelená na dve časti: jedna pre zápisové operácie a druhá pre čítacie operácie.

Jadro procesora predstavuje akumulátorovú architektúru, ktorá poskytuje široké možnosti adresácie a indexácie dát v pamäti. Adresová zbernica je široká 16 bitov, čo umožňuje adresovať 2^{16} bytov resp. 65536 bytov. Šírka dátovej zbernice je 8 bitov, čo znamená, že dáta a inštrukcie môžu byť z procesora alebo do procesora prenášané v 8-bitových blokoch.

Mikrokontroléry HCS08 nie sú vybavené žiadnou cache pamäťou. Dáta a inštrukcie sú tak priamo čítané a zapisované z/do pamäte. Žiaden z modelov taktiež nedisponuje ani jednotkou na výpočet čísiel v pohyblivej rádovej čiarke a rovnako ani žiadnymi akcelerátormi výpočtov.

Registrová sada mikroprocesora pozostáva z piatich registrov:

- 8-bitový akumulátor (A)
- 16-bitový indexovací register (H:X)
- 16-bitový programový čítač (PC)
- 16-bitový ukazovateľ zásobníka (SP)
- 8-bitový stavový register CCR.

Register A slúži ako jediný ako pracovný register, nad ktorým je možné vykonávať všetky aritmetické a logické operácie, zatiaľ čo statový register poskytuje informácie o výsledkoch jednotlivých operácií. Veľkosť operandov všetkých operácií je obmedzená veľkosťou akumulátora tj. 8 bitov, prípadne 16-bitovým registrom H:X. Detailnejší popis registrovej a inštrukčnej sady je možné nájsť v programátorskej príručke mikrokontroléra HCS08 [9] alebo v manuáli konkrétneho modelu mikrokontroléra.

2.3.2 Časovanie vykonávania inštrukcií

V rodine mikrokontrolérov HCS08 sa dĺžka vykonávania inštrukcií meria v počte periód frekvencie systémov zbernice BUSCLK. Touto frekvenciou sú taktované zbernice, pamäťový systém a rovnako aj periférne zariadenia. Mikroprocesor je taktovaný na dvojnásobok frekvencie BUSCLK. Rôzne modely mikrokontrolérov HCS08 sú dimenzované na rozličné maximálne frekvencie BUSCLK. Štandardne sú to hodnoty 8 MHz, 10 MHz, 20 MHz a 25 MHz.

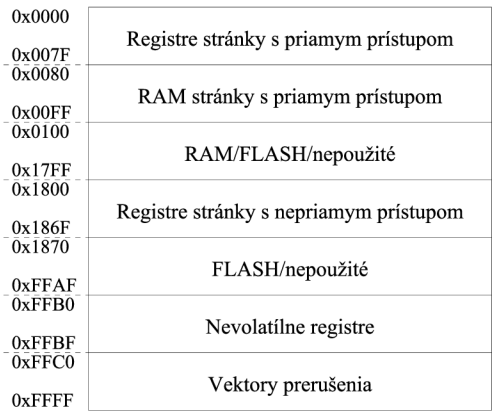
Dĺžka vykonávania inštrukcií je premenlivá, pričom najkratšia inštrukcia trvá jednu periódu frekvencie BUSCLK, zatiaľ čo najdlhšia trvá jedenásť periód. Priemerná doba vykonávania inštrukcií je približne 4 periódy frekvencie BUSCLK na inštrukciu. Z toho vyplýva, že pri maximálnej frekvencii (BUSCLK = 25 MHz) je priemerná doba vykonávania jednej inštrukcie 160 ns, čiže za sekundu sa vykoná priemerne 6.25 miliónov inštrukcií (6.25 MIPS).

2.3.3 Mapovanie adresového priestoru

Registre periférnych zariadení sú adresované rovnakou zbernicou ako RAM a FLASH pamäť. Niektoré adresy pamäťového priestoru tak slúžia k prístupu k týmto registrom, iné zas k prístupu k RAM alebo FLASH pamäti. Niektoré adresy pamäťového systému nemusia byť implementované – závisí od konkrétneho modelu mikrokontroléru.

Jednotný adresový priestor zjednodušuje inštrukčnú sadu, pretože akákoľvek inštrukcia, ktorá môže pristupovať do pamäte, môže rovnako pristupovať aj k registrom periférnych zariadení.

Obrázok 2.1 zobrazuje všeobecné rozdelenie adresového priestoru rodiny mikrotrontrolérov HCS08.



Obrázok 2.1: Mapovanie pamäťového priestoru

Na adresách v rozmedzí 0x0000 - 0x00FF sa nachádza oblasť s priamym prístupom (angl. direct page area). Tento názov pochádza z rozsahu adries, ktoré je možné adresovať v priamom adresovacom móde. Prístup do tejto oblasti je rýchlejší než prístup do ostatných častí adresového priesoru, pretože k adresovaniu stačí jediný byte, zatiaľ čo zvyšok pamäte musí byť adresovaný dvoma bytami. Inštrukcie pristupujúce do priamo–adresovateľnej oblasti potrebujú k vykonaniu menej hodinových cyklov frekvencie BUSCLK než inštrukcie určené pre nepriamo adresovateľnú oblasť.

Priamo adresovateľný priestor je rozdelený do dvoch častí:

1. Oblasť s priamo adresovateľnými registrami, začínajúca na adrese 0x0000 a končiaca na adrese 0x007F. Do tejto oblasti je namapovaná väčšina registrov periférnych zariadení.
2. Oblasť s priamo adresovateľnou RAM pamäťou, začínajúca na adrese 0x0080 a končiaca na adrese 0x00FF. Táto oblasť je vhodná na ukladanie často používaných premenných, kvôli rýchlejšiemu prístupu.

V rodine mikrokontrolérov HCS08 je možné sa stretnúť ešte s jednou oblasťou mapujúcou registre periférnych zariadení, a to na adrese 0x1800. K týmto registrom sa obvykle pristupuje oveľa menej a preto sú mapované do oblasti s nepriamym prístupom.

Každý model mikrokontroléru HCS08 disponuje 16-timi nevolatílnymi registrami, ktoré si dokážu uchovať svoju hodnotu aj po odpojení napájania. Programujú sa súčasne s FLASH pamäťou a nachádzajú sa na adrese 0xFFB0.

Vektory prerušení sa nachádzajú na úplnom konci adresového priestoru od adresy 0xFFC0. Táto oblasť slúži pre ukladanie adries, na ktorých sa nachádzajú obslužné rutiny prerušení, z ktorých väčšina pochádza práve od periférnych zariadení.

2.3.4 Periférne zariadenia

Mikrokontroléry rodiny HCS08 disponujú širokým spektrom periférnych zariadení, medzi ktoré patrí napríklad programovateľný interný generátor hodinového signálu, programovateľné čítače a časovače, ADC prevodník, analógový komparátor, PWM modul a moduly pre sériovú komunikáciu SPI, SCI a IIC.

Z hľadiska implementácie priameho prístupu do pamäte sú zaujímavé najmä komunikačné rozhrania SCI, SPI a IIC, preto sa v tejto práci budem zaoberať práve nimi. Rozhrania pre generovanie hodinového signálu, čítače ani časovače obvykle nepotrebujú prenášať dáta medzi operačnou pamäťou a teda nie sú ani vhodné pre realizáciu DMA prenosu. Vhodným kandidátom pre DMA prenos sa javí byť ADC prevodník. Problémom však je nemožnosť implementovať ADC prevodník v FPGA, v ktorom bude implementovaný tento projekt a preto sa týmto prípadom nebudem zaoberať. Alternatívnu cestu k zaznamenávaniu hodnôt z ADC prevodníka poskytujú spomenuté komunikačné rozhrania, ku ktorým je možné ADC prevodník pripojiť.

2.4 Seriové komunikačné rozhrania rodiny mikrokontrolérov HCS08

Ako bolo spomenuté v kapitole 2.3.4, pre implementáciu priameho prístupu do pamäte sú najvhodnejšie tie rozhrania, u ktorých sa predpokladá väčší prenos dát medzi operačnou pamäťou. Takými rozhraniami sú seriové komunikačné rozhrania SCI, SPI a IIC.

2.4.1 Asynchrónny sériový komunikačný modul SCI (Serial Communication Interface)

SCI je plne duplexné asynchrónne sériové rozhranie, tj. súčasne je možné dáta prijímať i vysielať. Systém SCI pozostáva z generátora hodinového signálu, modulu vysielača, modulu prijímača a niekoľkých pomocných obvodov pre prenos dát. Pre každý smer prenosu je potrebný jeden vodič.

Asynchrónne zariadenia sa vyznačujú zložitejšou štruktúrou, pretože musia byť schopné prenášať dáta s rozličnou frekvenciou a fázou. Každé zariadenie má svoj vlastný generátor hodinového signálu, ktorým sú vzorkované prenášané dáta. Než začnú dve zariadenia spolu komunikovať musí najskôr medzi nimi prebehnúť synchronizácia ich generátorov hodín.

Obrázok 2.2 zobrazuje dátový rámec komunikačného rozhrania SCI. Komunikácia prebieha na základe známej prenosovej rýchlosti a je zahájená štartovacím bitom. Po ňom nasleduje niekoľko dátových bitov (typicky 8), voliteľný paritný bit a jeden alebo dva stop bity.

Obrázok 2.2: Dátový rámec SCI

Zostupná hrana štartovného bitu synchronizuje hodiny vysielača a prijímača a za predpokladu, že je ich prenosová rýchlosť rovnaká (alebo takmer rovnaká), prijímač bude schopný úspešne obdržať vyslané dáta.

2.4.2 Synchrónny sériový komunikačný modul SPI (Serial Peripheral Interface)

Jedná sa o rozhranie určené pôvodne k pripojovaniu a komunikáciu s periférnymi zariadeniami, môže však byť jednoducho použité i ku komunikácii medzi mikrokontrolérmi.

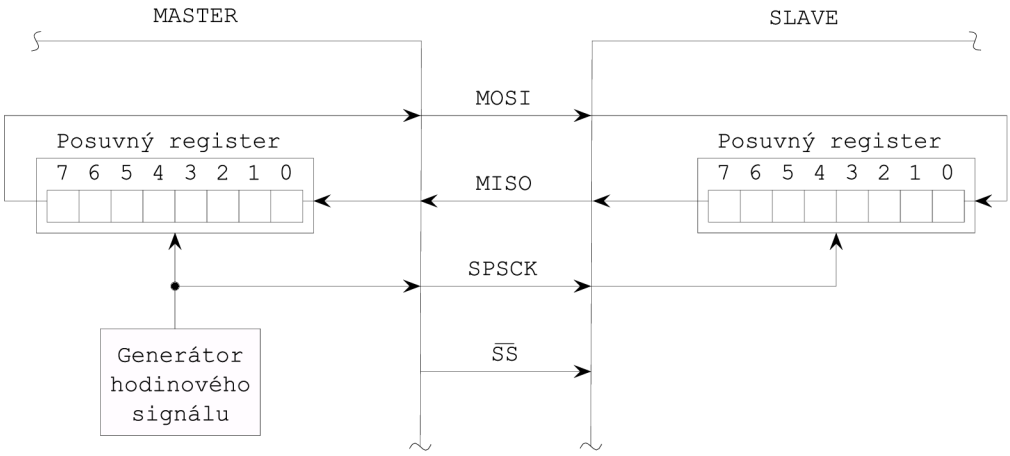
Rozhranie SPI je plne duplexné, čo znamená, že v každom okamžiku vždy prebieha prenos oboma smermi. Toto rozhranie umožňuje okrem spojenia point-to-point, teda spojenia dvoch zariadení, taktiež vytvoriť SPI zbernicu pre pripojenie viacero komponent, z ktorých vždy môže súčasne komunikovať iba jedna dvojica. Zariadenie pripojené na rozhranie SPI môže byť v jednom z dvoch režimov: master alebo slave.

Obrázok 2.3 znázorňuje blokovú schému SPI rozhrania. Master je zariadenie, ktoré generuje synchronizačný hodinový signál (na obrázku označený ako SPSCK), iniciuje a riadi komunikáciu na zbernici SPI. V zbernici SPI smie byť iba jedno zariadenie v režime master.

Zariadenia typu slave sú obvykle periférie, s ktorými master komunikuje. Práve master určuje, s ktorým zariadením typu slave bude v danom čase komunikovať. Výber slave zariadenia prebieha prostredníctvom signálu $\overline{\text{SS}}$ (z angl. Slave Select). Master spravidla býva mikrokontrolér.

Prenos dát je typicky realizovaný za pomoci posuvných registrov, ktorých obsah si zariadenia počas prenosu vzájomne vymenia. K tomu slúžia signály MISO (z angl. Master In Slave Out) a MOSI (z angl. Master Out Slave In). Cez signál MISO putujú dáta smerom od zariadenia typu slave k zariadeniu typu master. Signál MOSI zabezpečuje prenos opačným

smerom. Veľkost prenášaného slova je obvykle stanovená na 8 alebo 16 bitov, avšak je možné sa stretnúť i s inými šírkami slova pri niektorých špecifických aplikáciach.



Obrázok 2.3: Bloková schéma SPI

2.4.3 Synchrónny sériový komunikačný modul IIC (Inter-Integrated Circuit)

IIC (častokrát tiež označované ako I^2C) je synchrónne sériové rozhranie vyvinuté firmou Philips Semiconductors[6] (v súčasnosti NXP Semiconductors), ktorého hlavným cieľom je jednoduchosť a lacná implementácia. Komunikačný protokol tohoto rozhrania je typu master-slave, pričom komunikácia prebieha spôsobom half-duplex, čiže komunikácie vždy prebieha iba jedným smerom.

Rozhranie IIC pozostáva z jedného dátového vodiča SDA (Serial Data Line) a jedného vodiča prenášajúceho synchronizačný signál SCL (Serial Clock Line). Každé zariadenie pripojené k zbernici môže byť softwarovo adresované pomocou unikátnej adresy, pričom počet zariadení pripojených k zbernici je obmedzený iba kapacitnou reaktanciou zbernice. Štandardné implementácie umožňujú 7-bitové a 10-bitové adresovanie. Počet zariadení na zbernici je tak limitovaný na 127 resp. 1023.

IIC je multi-master zbernica zahŕňajúca vo svojom komunikačnom protokole detekciou kolízií a proces arbitrácie, aby sa zabránilo poškodeniu dát v prípade, kedy dve zariadenia typu master súčasne započnú prenos dát. Zariadenie typu master môže operovať ako vysielač aj prijímač.

Dáta sú prenášané sériovo po jednotlivých bitoch. Dátový rámec je vyobrazený na obrázku 2.4. Prenesenie jedného bytu dát pozostáva z nasledujúcich krokov:

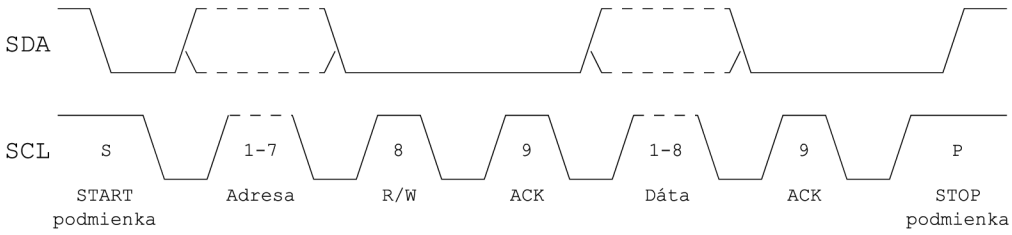
1. START podmienka – štartovacia hrana, ktorá udáva začiatok prenosu. Je definovaná ako prechod signálu SDA z log. 1 do log. 0 zatiaľ čo je hodnota signálu SCL v log. 1.
2. Prenos adresy – 7-bitová adresácia zariadenia typu slave
3. Smer prenosu R/W – master týmto bitom informuje zariadenie slave o požadovanom smere prenosu dát v nasledujúcom bloku. Pre log. 0 bude master v režime vysielača a naopak pre log. 1 bude master v režime prijímača.

4. Potvrdenie adresy ACK (z angl. acknowledge) – nasleduje bezprostredne po prenesení adresy. Zariadenie typu slave týmto bitom dáva najavo, že rozpoznalo svoju adresu a uvedie signál SDA do log. 0. V prípade, že je signál SDA počas tohoto hodinového cyklu v log. 1, znamena to, že žiadne zariadenie s požadovanou adresou momentnálne nie je pripojené k zbernici, resp. nie je k dispozícii.

5. Prenos ôsmich bitov dát smerom od vysielača k prijímaču.

6. Potvrdenie dátového bloku – prijímač informuje o úspečnom resp. neúspešnom prijatí dát a to zaslaním buď hodnoty ACK (z angl. acknowledge) alebo NACK (z angl. not acknowledge). Hodnota ACK je definovaná ako log. 0 a znamená úspešné prijatie dát. Hodnota NACK je definovaní ako log. 1 a znamená buď neúspešné prijatie dát alebo žiadosť o ukončenie prenosu zo strany prijámača. Vysielač musí po obdržaní hodnoty NACK ukončiť prenos.

7. STOP podmienka – ukončovacia hrana, ktorá udáva koniec prenosu. Je definovaná ako prechod signálu SDA z log. 0 do log. 1 zatiaľ čo je hodnota signálu SCL v log. 1.



Obrázok 2.4: Dátový rámec protokolu IIC

Pôvodná maximálna rýchlosť prenosu bola stanovená na 100 kbit/s, neskôr boli štandardom pridané ďalšie rýchlostné módy, ktoré umožňujú prenosy do rýchlosti 400 kbit/s, 1 Mbit/s, 3.4 Mbit/s a 5 MBit/s v závislosti na móde.

2.5 Režimy prenosu dát medzi operačnou pamäťou a periférnymi zariadeniami

V počítačoch sa stretávame s troma základnými konceptami prenosu dát medzi operačnou pamäťou a periférnymi zariadeniami[7]. Každý z týchto konceptov vyžaduje k svojej činnosti určité hardwarové vybavenie. Týmito konceptami sú režim PIO, prenos riadený prerušením a nakoniec režim priameho prístupu do pamäte DMA.

2.5.1 Režim PIO (Programmed Input/Output)

Programmed input/output (PIO) je pôvodná metóda prenosu dát cez zbernicu IDE/ATA. Prenos medzi zariadeniami a operačnou pamäťou je uskutočňovaný za účasti procesoru, kedy procesor najskôr iniciuje prenos a následne cyklicky kontroluje dokončenie prenosu. Inicializácia prenosu spočíva v prenesení požadovaných dát z registrov procesora do registrov radiča periférneho zariadenia. O prenos dát medzi radičom periférneho zariadenia a periférnym zariadením sa stará samotný radič. Kontrola dokončenia prenosu spočíva v

neustálom čítaní a vyhodnocovaní stavových registrov radiča periférnych zariadení. Tento proces je taktiež známy pod anglickým názvom *polling*. Stavové registre zároveň slúžia pre informovanie procesora o výskyte chyby počas prenosu dát.

Akonáhle je prenos dát dokončený, môže začať prenos ďalšieho bloku dát. Dôležitým znakom režimu PIO je skutočnosť, že periférne zariadenia ani ich radiče priamo neinformujú procesor o dokončení operácie prenosu a nevyvolávajú ani prerušenie.

Problém takéhoto prenosu spočíva v nízkej prenosovej rýchlosti vstupno/výstupných zariadení. Po zahájení prenosu musí procesor čakať na jeho dokončenie po dobu desiatok, stoviek, pripadne i tisícok hodinových cyklov. Z hľadiska programovania vedie takýto princíp ku komplikovaným čakacím slučkám, ktoré sa ešte viac komplikujú pri súčasnom obsluhovaní viacero periférnych zariadení. Z hľadiska efektivity možno túto metódu prenosu dát považovať za veľmi neefektívu, pretože procesor je po dobu prenosu dát blokovaný a nemožno ho využiť k iným výpočtom.

2.5.2 Prenos riadený prerušením

Ďalším spôsobom prenosu dát medzi operačnou pamäťou a periférnym zariadením je prenos riadený prerušením. Vylepšenie oproti režimu PIO spočíva v tom, že procesor nemusí čakať na dokončenie prenosu. V princípe sú si však tieto režimy značne podobné.

Prenos dát je opäť iniciovaný procesorom, kedy sú požadované dáta prenesené z registrov procesora do registrov radiča periférneho zariadenia. Rozdiel oproti režimu PIO však spočíva v tom, že procesor cyklicky nekontroluje dokončenie prenosu ale o jeho dokončení bude priamo informovaný od radiča periférneho zariadenia pomocou prerušenia. Znamená to, že procesor je počas prenosu uvoľnený pre vykonávanie inej činnosti.

Po tom ako je procesor informovaný o dokončení prenosu, musí pozastaviť svoju aktuálnu činnosť a obslúžiť prerušenie od radiča periférneho zariadenia. Týmto sa rozumie obvykle prenos stavového registra radiča periférneho zariadenia do registrov procesora a jeho následným vyhodnotením za účelom overenia, či prenos dát prebehol v poriadku. V prípade čítania dát z periférneho zariadenia ďalej nasleduje prenos z dátových registrov radiča periférneho zariadenia do registrov procesora a ich následné spracovanie, napríklad uloženie do operačnej pamäte. Po tomto úkone môže procesor zahájiť ďalší cyklus prenosu. Z tohoto vyplýva, že dáta medzi operačnou pamäťou a periférným zariadením sú opäť prenášané skrz procesor.

Zásadný rozdiel oproti režimu PIO teda spočíva v tom, že stavové registre radiča periférneho zariadenia sú do procesora prenášané iba raz, po tom ako je prenos dát dokončený. Nevýhoda prenosu riadeného prerušením sa prejaví pri prenose väčšieho bloku dát, pretože zariadenia obvykle umožňujú prenos iba malého objemu dát (napr. jedno slovo), a tak je nutné proces prenosu iniciovať po každom prenesenom slove.

Prenos dát riadený prerušením vyžaduje hardwarové vybavie v podobe radiča prerušení, ktorý umožňuje vybavovať prichádzajúce prerušenia od viacero zariadení podľa ich priority. Prerušenie s najvyššou prioritou je ďalej poslané do procesora, ktorý dané prerušenie obslúži. K obsluhe prerušení slúžia špeciálne obslužné rutiny, ktoré bývajú súčasťou operačného systému alebo si ich programátor musí vytvoriť.

Obvyklý postup procesora po obdržaní žiadosti na prerušenie je nasledovný:

1. Procesor dokončí aktuálne rozpracovanú inštrukciu.

2. Procesor zistí od ktorého zariadenia pochádza žiadosť prerušenia. Táto informácia slúži ako index do tabuľky vektorov prerušení, na základe ktorej procesor zistí adresu v pamäti, na ktorej sa nachádza obslužná rutina.

3. Pretože sa v obslužných rutinách využívajú registre procesora, musí sa uchovať ich pôvodný obsah. To sa deje uložením obsahov príslušných registrov na zásobník. Medzi nimi sú aj stavové registre a register programového čítača PC. Tento úkon môže trvať niekoľko hodinových cyklov.

4. Vykoná sa telo obslužnej rutiny, ktoré pozostáva z kontroly statových registrov a prípadného prenosu dát z/do periférného zariadenia. Tento prenos prebieha skrz registre procesora.

5. Po ukončení obslužnej rutiny musí procesor obnoviť pôvodné hodnoty registrov a to ich načítaním zo zásobníka. Tento úkon opäť zvykne trvať niekoľko hodinových cyklov. Následne môže procesor pokračovať v pôvodnej činnosti.

Ako vyplýva z uvedeného postupu, procesor musí k obsluhe prerušenia vykonať niekoľko operácii naviac v podobe uchovania a obnovenia obsahov svojich registrov. Prenos jedného bloku dát z/do radiča periférneho zariadenia je tak o niečo dlhší než v režime PIO, kde nie je potrebné uchovávať hodnoty registrov. Výhodou však je už spomínané uvoľnenie procesora po dobu vykonávania činnosti periférneho zariadenia.

V počítačovej technike je možné sa bežne stretnúť s prerušovacím systémom, ktorý zahŕňa niekoľko typov prerušení. Typickými predstaviteľmi sú napríklad tzv. maskovateľné prerušenia, ktoré môžu byť procesorom ignorované, alebo softwarové prerušenia, ktoré sú vyvolané programom resp. inštrukciou procesora. Softwarové prerušenia majú široké uplatnenie pri implementácii jadra operačného systému.

2.5.3 Priamy prístup do pamäte (DMA)

DMA (angl. Direct Memory Access) je spôsob priameho prenosu dát medzi operačnou pamäťou a perifémymi zariadeniami. Dáta nie sú prenášané skrz procesor, vďaka čomu je možné dosiahnuť vyššieho výkonu.

DMA sa používa pre prenos väčšieho objemu dát a je využívaná viacerými komponentami počítača ako napr. radič pevného disku, grafická karta, sieťová karta, zvuková karta a podobne. DMA sa s výhodou používa v systémoch pracujúcich v reálnom čase, čiže pre aplikácie ktorých činnosť sa nesmie prerušiť (resp. prerušenie je pre ne kritické). Pretože sa jedná o obsiahlu kapitolu, ktorá je predmetom tejto práce, rozhodol som sa venovať detialnému popisu činnosti v samostatnej kapitole s číslom 2.6.

2.6 Princíp činnosti priameho prístupu do pamäte (DMA)

Ako už bolo spomenuté, DMA je spôsob piameho prenosu dát medzi operačnou pamäťou a perifémymi zariadeniami bez účasti procesoru.

2.6.1 Popis činnosti

Základným stavebným prvkom je radič DMA[8], ktorý riadi prenos dát medzi operačnou pamäťou a perifémymi zariadeniami. Zahájenie prenosu je iniciované procesorom, ktorý na-

konfiguruje radič DMA rovnakým spôsobom, akým sú konfigurované akékoľvek iné radiče periférných zariadení, tj. zápisom do ich registrov. O prenos dát medzi radičom periférneho zariadenia a operačnou pamäťou sa potom stará radič DMA, zatiaľ čo je procesor uvoľnený pre vykonávanie inej činnosti. Dokončenie prenosu ohlási radič DMA prostredníctvom systému prerušenia, prípadne iným spôsobom, akým je napríklad uvoľnenie systémových zberníc.

Štandardnými registrami každého DMA radiča sú register adresy, register čítača a niekoľko riadiacich registrov. Register adresy udáva miesto v operačnej pamäti z ktorého sa budú dáta čítať alebo do ktorého sa budú dáta zapisovať. Register čítača slúži k nastaveniu požadovaného objemu dát v bytoch (alebo iných jednotkách), ktoré sa majú preniesť. Riadiace registre slúžia na konfiguráciu módu prenosu, prípadne iných špecifických nastavení.

Po nastavení bázovej adresy a objemu dát vyšle procesor signál radiču periférneho zariadenia pre začatie prenosu. Z teoretického hľadiska je od tohoto okamžiku procesor uvoľnený pre inú činnosť až do chvíle, kým sa neprenesú všetky dáta. Radič DMA vystavuje adresy na adresovej zbernici a taktiež nastavuje signály riadiacej zbernice pre čítanie a zápis. Akonáhle sú dáta pripravené na prenos medzi periférnym zariadením a pamäťou, radič DMA inkrementuje svoj interný adresový register, zatiaľ čo sa pripravené dáta prenášajú. Tento proces sa opakuje až kým sa neprenesie celý požadovaný blok dát.

2.6.2 Módy prenosu dát

V osobných počítačoch sa pri technológii DMA často operuje s rôznymi módmi prenosu dát[8], ktoré sa líšia v prekrývaní činnosti radiča DMA a procesoru. Tieto módy sú závislé na architektúre vyhotovenia a prideľovania zberníc, ako aj pracovných frekvenciách procesora, zberníc a operačnej pamäte.

V dôsledku zdieľania systémových zberníc viacero zariadeniami (vrátane procesora) dochádza k situáciam, kedy nie je možné vykonávať niektoré operácie súčasne. Napríklad v architektúre s jednocestou operačnou pamäťou resp. jednou adresovou zbernicou nie je možné aby dve zariadenia súčasne pristupovali do pamäte. Znamená to, že zatiaľ čo sú dáta prenášané prostredníctvom priameho prístupu do pamäte, nesmie procesor do tejto pamäte pristupovať a naopak, počas prístupu procesora do operačnej pamäte musí byť DMA prenos pozastavený. Výlučný prístup k systémovým zberniciam zabezpečuje systém prideľovania zberníc v podobe jednoduchých signálov fungujúcich na princípe žiadosť – potvrdenie alebo špeciálneho radiča pre prideľovanie zbernice. V praxi sa používajú 3 základné módy – burst mód, cycle stealing mód a transparentný mód.

Burst mód

V burst móde (taktiež nazývaný blokový mód) je celý blok dát prenesený v jednej sekvencii. Radič DMA po získaní prístupu k systémovej zbernici začne prenos celého požadovaného bloku dát ako sekvenciu slov a systémová zbernica je vrátená späť procesoru až po prenesení celého bloku. Tento mód je užitočný napríklad pri nahrávaní programu alebo dát do pamäte, avšak jeho nevýhoda tkvie v tom, že počas celého prenosu nemá procesor prístup k systémovým zberniciam čo môže viesť k jeho úplnému pozastaveniu. Napriek tejto nevýhode je prenos dát značne rýchlejší než prenos dát skrz procesor.

Cycle stealing mód

Cycle stealing mód sa používa v systémoch, v ktorých by procesor nemal byť pozastavený počas celého prenosu v burst móde. V cycle stealing móde radič DMA získa prístup k systémovej zbernici rovnakým spôsobom ako v burst móde. Rozdiel je však v tom, že v cycle stealing móde sa radič DMA vzdá systémovej zbernice každým preneseným slovom. Pre prenos každého slova musí radič vyslať žiadosť o pridelenie systémovej zbernice, ktorá mu je pridelená akonáhle procesor dokončí rozpracovanú inštrukciu. Pomocou tohoto módu je možné jemne striedať výpočet procesora a prenos dát medzi operačnou pamäťou a periférnymi zariadeniami.

Prenos bloku dát v tomto móde trvá podstatne dlhšie než v burst móde, na druhej strane však nepozastavuje procesor na dobu potrebnú k preneseniu celého bloku dát. Cycle stealing mód je užitočný v systémoch, ktoré monitorujú a spracovávajú dáta v reálnom čase.

Transparentný mód

Transparentný mód vyžaduje najviac času k prenosu bloku dát, avšak zároveň je to najefektívnejší spôsob prenosu z hľadiska celkového výkonu systému. Radič DMA prenáša dáta iba v prípade, ak procesor vykonáva operácie, ktoré nepožadujú prístup k systémovej zbernici. Výhodou takéhoto prenosu je fakt, že procesor nie je nikdy pozastavený v dôsledku obsadenia zbernice DMA radičom. Nevýhodou transparentného módu je potreba špeciálneho hardwaru na detekciu situácie kedy proceser nepotrebuje systémovú zbernicu, ktorý môže byť veľmi zložitý a drahý.

2.6.3 Problém koherencie dát

V systémoch, ktoré používaju cache pamäte môže DMA prenosť viesť k problémom s koherenciou cache pamätí. K tejto situácii dochádza napríklad v prípade, kedy DMA radič prenáša dáta priamo z operačnej pamäte do periférneho zariadenia, avšak aktuálne dáta sa nachádzajú v pamäti cache a neboli ešte prepísané do operačnej pamäte. V tomto prípade sa z operačnej pamäte čítajú staré, neauktuálne dáta. Druhým prípadom je situácia, kedy sa dáta čítajú z periférneho zariadenia a zapisujú do operačnej pamäte. Ak boli tieto dáta pôvodne nacachované, musia byť v cache pamäti zneplatnené. V opačnom prípade by procesor pracoval s neaktuálnymi dátami.

Tento problém môže byť ošetrený buď špeciálnym hardwarom, ktorý bude monitorovať DMA prenosy a podľa situácie korektne pracovať s cache pamäťami, alebo softwarovo, kedy operačný systém zabezpečí správnosť dát.

Nakoľko mikrokontrolér HCS08 nedisponuje žiadnymi pamäťami cache, nebudem sa týmto problémom ďalej zaoberať.

2.6.4 DMA v osobných počítačoch a vo vnorených systémoch

V osobných počítačoch sa stretávame s rozličnými implementáciami DMA prenosu, ktoré sa líšia každou zbernicou. V tejto kapitole bližšie popíšem implementáciu priameho prístupu do pamäte v zberniciach ISA, PCI a AHB.

Zbernica ISA

ISA (skratka z anglického Industry Standard Architecture) je štandard pre zbernice IBM kompatibilných počítačov. Zbernica ISA je rozšírením natívnej zbernice procesora Intel 8086 o signály radiča prerušenia a radiča DMA. Je určená na pripájanie periférnych kariet k základnej doske.

Základ implementácie priameho prístupu do pamäte DMA tvorí radič DMA od firmy Intel s označením 8237. V zbernici ISA sa nachádzali dva takéto radiče, jeden pre 8-bitový prenos a druhý pre 16-bitový prenos dát. Každý z týchto radičov umožňoval prenos dát až v 4 kanáloch, tj. dokázal obsluhovať prenos dát pre štyri periférne zariadenia súčasne.

Každý DMA kanál pozostával z 16-bitového adresového registra a 16-bitového registra pre požadovaný objem dát. Začiatok prenosu iniciuje ovládač zariadenia (čiže procesor) nastavením registrov DMA kanálu spolu so smerom prenášania dát. Následne procesor vydá pokyn pre začiatok prenosu. Ukončenie prenosu je signalizované procesoru prostredníctom systému prerušenia.

Prideľovanie zbernice zabezpečuje dvojica signálov HRQ (Hold Request) a HLDA (Hold Acknowledge). Signálom HRQ radič DMA signalizuje procesoru, že niektoré zo zariadení vyžaduje DMA prenos. Signálom HLDA dáva procesor vedieť radiču DMA, že systémová zbernica je uvoľnená pre DMA prenos. Procesor je v tejto fáze pozastavený. Po dokončení DMA prenosu sa preruší žiadosť HRQ a procesor prestane vysielať aktívny signál HLDA. Tým je systémová zbernica vrátená späť procesoru. Podrobnejšiemu popisu činnosti DMA radiča Intel 8237 sa venuje kapitola 2.7.

Zbernica PCI

PCI (z anglickej skratky Peripheral Component Interconnect) je štandard pre zbernicu počítača k pripojeniu periférnych zariadení k základnej doske. Zbernica PCI nahradila zastaralú ISA architektúru.

Zbernica PCI nemá žiaden centrálny radič DMA ako je tomu v prípade zbernice ISA. Namiesto toho, každé zariadenie pripojené na zbernicu môže individuálne žiadať o pridelenie zbernice a skrz ňu pristupovať k operačnej pamäti. Tieto žiadosti obsluhuje centrálny radič PCI zbernice (zvyčajne southbridge v moderných PC architektúrach). Zariadenie, ktorému bola pridelená PCI zbernica sa stáva masterom a v tomto stave môže iniciovať prenos dát na zbernici. Takéto zariadenie môže byť na zbernici v jednom momente iba jedno. Zbernica PCI umožňuje nielen prenos dát medzi periférnym zariadením a operačnou pamäťou ale aj vzájomne medzi periférnymi zariadeniami.

Zbernica AHB

V obvodoch System on Chip (SoC) ako aj vo vnorených systémoch sa častokrát používa komplexná zbernica AMBA High-performance Bus. AMBA definuje dva typy komponentov – master a slave. Rozhranie slave je podobné režimu PIO, kedy program bežiaci na procesore pristupuje k registrom periférneho zariadenia. Rozhranie master môže byť použité zariadením k vykonávaniu DMA prenosu dát.

Zbernica AHB nepodporuje trojstavovú logiku, preto nie je možné aby sa zariadenia pripojené k zberenici správali ako odpojené, a rovnako neumožňuje ani meniť smer prenosu dát na vodičoch. Kvôli tejto vlastnosti musia byť zariadenia, ktoré potrebujú prenášať veľké množstvo dát z/do operačnej pamäte pripojené cez obe rozhrania - master aj slave.

Rovnako ako zbernica PCI, ani zbernica AHB nepotrebuje centrálny radič DMA. Namiesto toho potrebuje arbiter, ktorý je potrebný pre prípad výskytu viacero zariadení pripojených cez rozhranie master.

2.7 Programovateľný DMA radič Intel 8237

Pre návrh radiča DMA pre rodinu mikrokontrolérov HCS08 som sa rozhodol vychádzať zo známeho DMA radiča Intel 8237 používaného v zberniciach ISA. Preto v tejto časti bližšie popíšem jeho činnosť. Text tejto sekcie je prevzatý z článku od Dušana Kollára[5] a doplnený informáciami z manuálu k Intel 8237 [4].

2.7.1 Vlastnosti

Vlastnosti radiča DMAC (DMA Controller) typu 8237 možno zhrnúť do nasledujúcich bodov:

- DMAC obsahuje 4 prenosové kanály a každý z nich môže adresovať až 64 KB pamäte.
- Na základe požiadavky DREQ (DMA Request) od periférneho zariadenia preberá DMAC funkciu riadenia zbernice. V prípade, že žiadajúcich zariadení je viacej, uplatní sa vstavaný prioritný systém (voľba pevnej alebo rotujúcej priority).
- Hoci obvod DMAC možno naprogramovať aj do režimu súvislého prenosu dát, v PC sa činnosť DMAC pozastavuje len na vykonanie prenosu jedného slova, aby sa nenarušili podmienky obnovovania obsahu dynamických pamätí.

2.7.2 Bloková schéma zapojenia

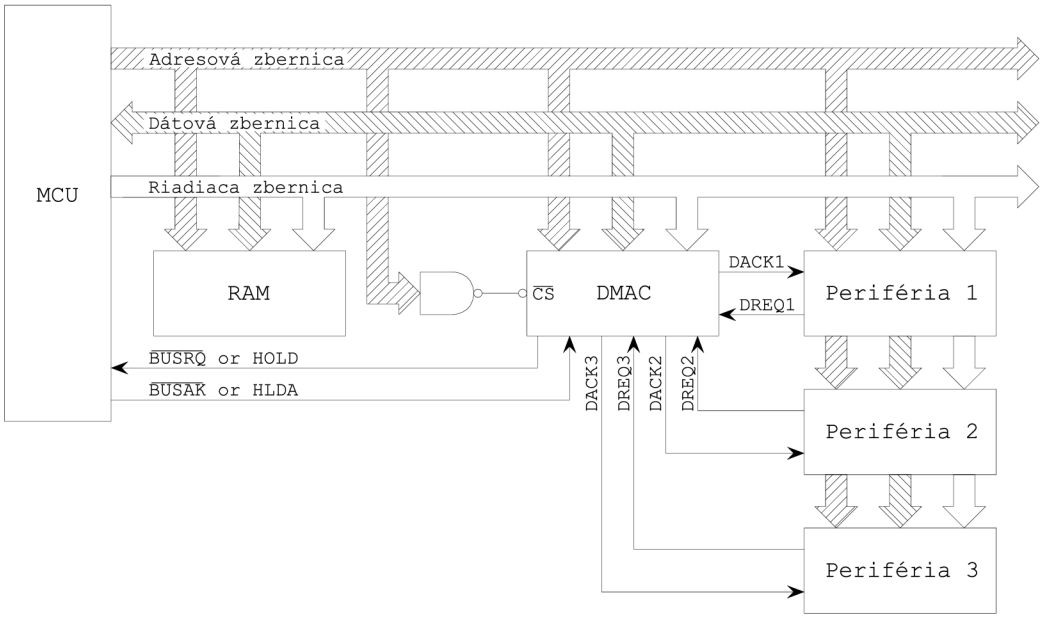
Obrázok 2.5 zobrazuje blokovú schému pripojenia radiča DMA Intel 8237 v počítači s 16-bitovou adresovou zbernicou. Na obrázku sú zobrazené 3 periférne zariadenia, pripojené k adresovej, dátovej a riadiacej zbernici. Podobným spôsobom je zapojený aj radič DMA.

Najdôležitejšiu úlohu v zapojení hrajú dvojice vodičov DREQ-DACK a HOLD-HLDA. Prostredníctvom dvojice DREQ-DACK sú prepojené periférne zariadenia s radičom DMA a fungujú systémom žiadosť-potvrdenie. Periférne zariadenie vyšle žiadosť do radiča DMA skrz signál DREQ (z angl. DMA Request) a ten pošle potvrdenie žiadosti signálom DACK (z angl. DMA Acknowledge).

Druhá dvojica vodičov funguje úplne rovnakým spôsobom ako prvá dvojica, avšak implementuje rozhranie medzi procesorom a radičom DMA. Signálom HOLD žiada radič DMA procesor k pozastaveniu činnosti, zatiaľ čo signálom HLDA (z angl. Hold Acknowledge) posiela procesor potvrdenie.

2.7.3 Inicializácia prenosu DMA

Inicializácia radiča DMAC sa uskutočňuje zápisom niekoľkých riadiacich slov do vnútorných registrov DMAC, ktoré sú adresované pomocu adries na zbernicových vodičoch A0 - A3. Pretože DMAC 8237 má len 8 bitovú dátovú zbernicu, tak niektoré adresy predstavujúce 16-bitové bunky sa nahrávajú na dva krát po 8-bitových slovách. Pred vykonaním konkrétneho prenosu cez niektorý z kanálov DMA musí byť nastavený:



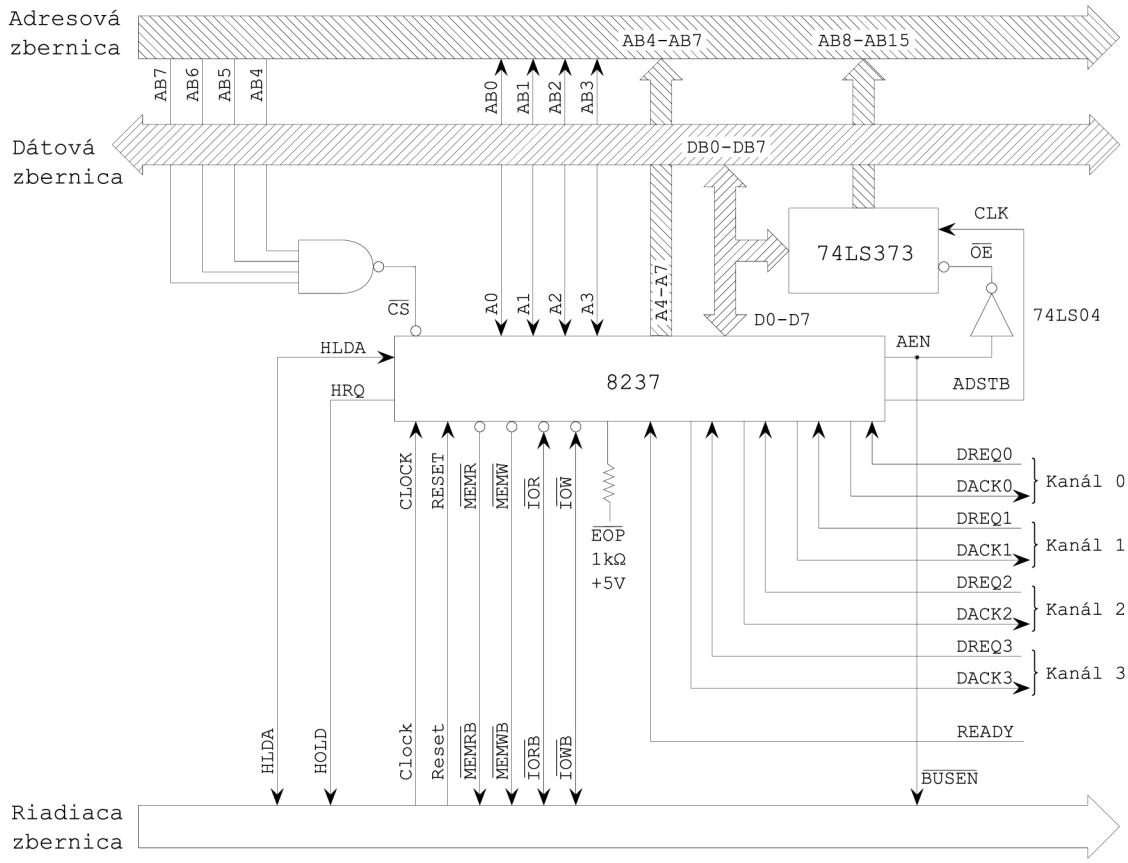
Obrázok 2.5: Pripojenie radiča DMA typu 8237 v počítači s 16-bitovou adresovou zbernicou.

- Typ a smer prenosu (8 alebo 16 bitové dáta; z pamäte alebo do pamäte).
- Register začiatočnej adresy v pamäti. Počnúc od tejto adresy sa budú dáta z pamäte čítať, resp. do pamäte zapisovať v závislosti na smere prenosu. Pretože v osobných počítačoch so zbernicou ISA je adresa najmenej 20-bitová, používa sa na doplnenie adresy z 8 bitov na potrebných 20 bitov, resp. 32 bitov ešte ďalší tzv. register stránky (na obrázku 2.6 označený ako 74LS373). Adresa sa ukladá po častiach pomocou špeciálneho synchronizačného signálu ADSTB (Address strobe).
- Register čítača (počet bytov), ktorý určuje množstvo dát, ktoré sa má preniesť.

2.7.4 Priebeh prenosu DMA

Prenos DMA sa vykonáva na žiadosť DREQ niektorého z vonkajších adaptérov.

1. DMAC rozhodne o priorite žiadosti konkrétneho signálu DREQ s ohľadom na prioritu ostatných signálov DREQ a vyšle na zbernicu signál HRQ. Tento signál sa zosynchronizuje s hodinami mikroprocesora na zbernicový signál HOLD.
2. Mikroprocesor v reakcii na signál HOLD dokončí vykonávanie rozpracovanej inštrukcie, uvedie svoje výstupy na zbernici do nevodivého stavu a vyšle naspäť k DMAC signál odpovede HLDA (Hold Acknowledge), čím signalizuje že zbernica bude počnúc nasledúcim cyklom voľná pre začatie DMA prenosu. Podobný signál ($\overline{\text{BUSEN}}$ - Bus enable) vyšle mikroprocesor aj smerom k budičom zberníc, ktoré odpoja mikroprocesor od systémovej zbernice.
3. DMAC na príchod signálu HLDA zareaguje vyslaním signálu DACK k príslušnému adaptéru. (Pre žiadajúci adaptér je tento signál výberovým, umožňujúcim mu prístup na zbernicu).



Obrázok 2.6: DMAC 8237 v systéme zberníc PC pre zabezpečenie DMA prenosu pre 4 zariadenia. DMAC 8237 môže priamo zadať len 8 najmenej významných bitov adresy. Zvyšné adresné bity sa ukladajú po častiach pomocou synchronizačného signálu ADSTB do registra stránky 74LS373, takže nakoniec sa vytvorí potrebná, napríklad 20-bitová adresa pamäte.

4. DMAC sa teraz môže ujať riadenia prenosu. V zapojení na obrázku 2.6 najprv vyšle hornú časť adresy (cez vodiče dátovej zbernice D0 - D7, D8 - D12) v sprievode riadiacich signálov AEN a ADSTB, ktorá reprezentuje časť adresy zdrojovej alebo cieľovej adresy v pamäti. Po uložení hornej časti adresy pomocou signálu ADSTB sa vodiče D0 - D7 stanú znova súčasťou dátovej zbernice DB a vodiče A0 - A7 znovu dolnou časťou adresovej zbernice AB. Signál AEN súčastne inicializuje výstup signálu $\overline{\text{BUSEN}}$.

5. DMAC odteraz už ovláda ridenie prenosu dát cez zbernicu a počas niekoľkých hodinových cyklov uskutoční prenos jednej položky údajov. Na rozdiel od mikroprocesora, radič DMA pomocou signálov $\overline{\text{MEMR}}$ a $\overline{\text{IOW}}$ ovláda súčastne prenos dát z pamäte do periférneho zariadenia, resp. prenos opačným smerom pomocou signálov $\overline{\text{MEMW}}$ a $\overline{\text{IOR}}$. Situáciu znázorňuje obrázok 2.7.

   Na takýto typ prenosu stačí jeden kanál DMA. Na prenos typu pamäť-pamäť sú však potrené 2 kanály DMA, pretože signály $\overline{\text{MEMR}}$ a $\overline{\text{MEMW}}$ sa nemožu generovať súčasne. Napríklad kanál DMA0 môže špecifikovať zdroj dát a kanál DMA1 zase cieľ

prenosu dát. V tomto prípade sa nevyžaduje dialóg DREQ-DACK.

6. Všeobecne v prípade dávkového alebo blokového prenosu by DMA prenos dát riadený radičom DMAC 8237 pokračoval až do zakončenia prenosu celého bloku dát (angl. Terminal Count - keď počítadlo bytov dopočíta do 0) alebo do okamihu pozastavenia DMA prenosu vonkajším signálom $\overline{\text{EOP}}$ (End of process). Z dôvodov potreby obnovy obsahu dynamických pamätí sa však v PC používa len jednoslovný DMA prenos. Preto po skončení jedného prenosu DMA sa preruší žiadosť HRQ a mikroprocesor prestane vysielať aktívny signál HLDA. Budiče systémovej zbernice sa pripoja k mikroprocesoru a začne prebiehať normálny procesorový zbernicový cyklus.

Zbernicové cykly mikroprocesora sa teda prekladajú podľa požiadavky (realizovanej pomocou prerušenia INT) zbernicovými cyklami DMA. Najvyšší dosiahnuteľný výkon DMA systému by nastal v prípade pravidelneho striedania zbernicových cyklov DMA a zbernicových cykov mikroprocesora. V skutočnosti z dôvodov zabezpečenia synchronizácie mikroprocesor prekladá viacej svojich cyklov s cyklom DMA a preto je výsledná rýchlosť DMA prenosu nižšia.

2.7.5 Riadenie smeru prenosu DMA

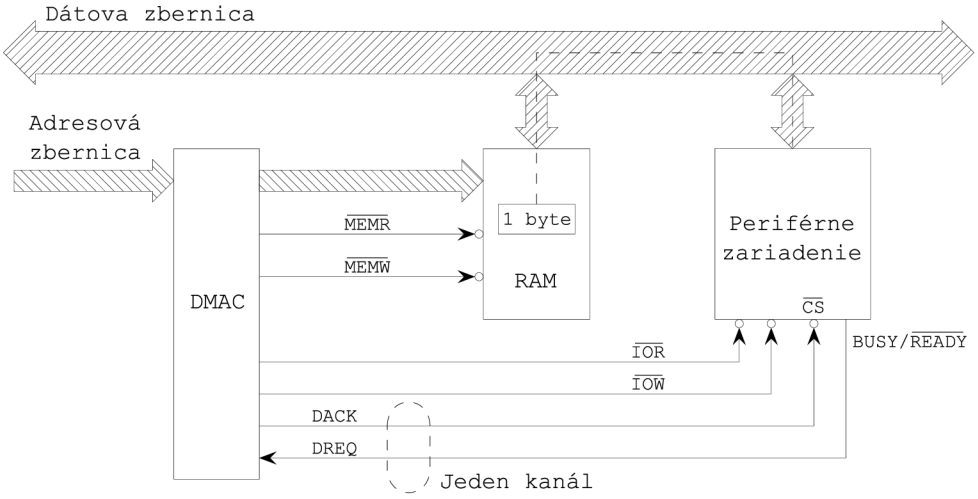
Obrázok 2.7 znázorňuje 3 možné smery prenosu dát, ktoré je možné za pomoci radiča Intel 8237 dosiahnuť. K ovládaniu smeru dát na dátovej zbernici slúžia dvojice vodičov $\overline{\text{MEMR}}$/$\overline{\text{MEMW}}$ a $\overline{\text{IOR}}$/$\overline{\text{IOW}}$. Prvá dvojica ovláda čítacie a zápisové operácie operačnej pamäte, druhá dvojica ovláda čítacie a zápisové operácie periférnych zariadení. Názvy vodičov sú skratky z angl. Memory Read/Memory Write, resp. IO Read/IO Write. Pod označením IO (Input/Output) sa v počítačovej technike rozumie periférne zariadenie.

Vodič DREQ sa dá chápať ako vodič, ktorý informuje radič DMA o tom, či je periférne zariadenie zaneprázdnené alebo už dokončilo svoju činnosť a je pripravené na DMA prenos dát. Signál DACK okrem potvrdenia žiadosti zároveň plní funkciu výberového vodiča označeného ako $\overline{\text{CS}}$ (z angl. Chip Select).

Možné smery prenosu

	Zdroj		Cieľ	
1.	$\overline{\text{MEMR}}$	$\longrightarrow$	$\overline{\text{IOW}}$	Potrebný 1 kanál $\longrightarrow$ Čítanie z pamäte a zápis do periférneho zariadenia
2.	$\overline{\text{IOR}}$	$\longrightarrow$	$\overline{\text{MEMW}}$	Potrebný 1 kanál $\longrightarrow$ Čítanie z periférneho zariadenia a zápis do operačnej pamäte
3.	$\overline{\text{MEMR}}$	$\longrightarrow$	$\overline{\text{MEMW}}$	Potrebné 2 kanály $\longrightarrow$ Čítanie z operačnej pamäte a zápis do operačnej pamäte; vyžaduje dočasné úložisko v radiči DMA.

Z dvojice vodičov $\overline{\text{MEMR}}$ a $\overline{\text{MEMW}}$ smie byť v jeden moment nastavený iba jeden z nich. Rovnaká podmienka platí aj pre dvojicu vodíčov $\overline{\text{IOR}}$ a $\overline{\text{IOW}}$.



Obrázok 2.7: DMAC používa signály $\overline{\text{MEMR}}$, $\overline{\text{MEMW}}$, $\overline{\text{IOR}}$ a $\overline{\text{IOW}}$, čo umožňuje 3 typy prenosov dát (Prenos DMA z pamäte do pamäte vyžaduje 2 prenosové kanály DMA).

2.8 Využitie DMA v oblasti vnorených systémov

V oblasti vnorených systémov zohráva priamy prístup do pamäte rovnako dôležitú ako v počítačoch pre všeobecné využitie. Technológia DMA má uplatnenie najmä v oblasti real-time spracovania a vyhodnocovania informácií, čo je práve oblasť, v ktorej sa vnorené systémy využívajú najviac.

Vnorené systémy môžu za pomoci technológie DMA efektívnejším spôsobom preposielať dáta medzi komunikačnými perifémymi zariadeniami, kedy jedno periférne zariadenie k monitorovaniu a zberu dát, a druhé periférne zariadenie na preposielanie vyhodnotených dát. Takáto jednoduchá obsluha by bola príliš ekonomicky nákladná za použitia mikroprocesora pre všeobecné využitie. Preto je technológia priameho prístupu do pamäte veľmi žiadúca aj v oblasti vnorených systémov.

Kapitola 3

Návrh

Kapitola s číslom 3 sa venuje návrhu konceptu radiča priameho prístupu do pamäte a jeho začlenením do architektúry rodiny mikrokontrolérov HCS08. V úvode bližšie popíšem okolnosti, z ktorých som vychádzal pri návrhu. Ďalej sa budem venovať samotnému návrhu DMA radiča, úpravam v architektúre rodiny mikrokontrlérov HCS08, ktoré umožnia pripojiť navrhnutý radič DMA, a v závere prevediem rozbor zachovania kompatibility s pôvodnou architektúrou rodiny HCS08.

V nasledujúcej časti práce budem pod pojmom *periférne zariadenie* myslieť niektoré zo sériových komunikačných rozhraní z rodiny mikrokontrolérov HCS08.

3.1 Projekt 68hc08 na webovej stránke OpenCores

Webová stránka OpenCores umožňuje užívateľom zdieľať ich vlastné softwarové implementácie hardwarových zariadení. Jej primárnym cieľom je vytvárať a zverejnovať implementácie jadier procesorov, koprocesorov, komunikačných zariadení a iných hardwarových riešení. Typickým implementačným jazykom týchto projektov sú VHDL a Verilog. Projekty sú väčšinou zverejnené pod licenciou slobodného softwaru GNU LGPL.

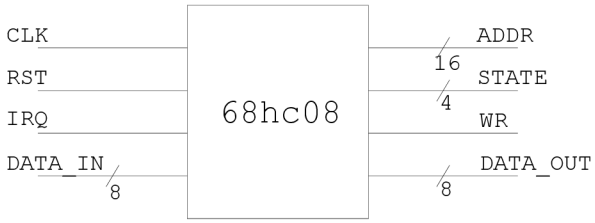
K mojej práci som sa rozhodol použiť projekt s názvom *68hc08* z webovej stránky OpenCores, autorom ktorého je Ulrich Riedel. Projekt implementuje v jazyku VHDL jadro mikroprocesora z rodiny mikrokontrolérov M68HC08. Táto rodina je predchodcom rodiny mikrokontrolérov HCS08, avšak zásadné rozdiely medzi týmito rodinami sa týkajú najmä periférnych zariadení. Rozdiel v jadrách mikroprocesorov oboch architektúr spočíva iba v malých rozdieloch doby vykonávania niektorých inštrukcií a v pridaní niekoľkých nových operačných kódov. Mikroprocesor projektu *68hc08* umožňuje dosiahnuť dvojnásobný výkon oproti originálnemu mikrokontroléru M68HC08 a operácie násobenia a delenia sú urýchlené o niekoľko taktov. Všetky ostatné vlastnosti sú zachované oproti originálu.

Obrázok 3.1 zobrazuje konektory mikroprocesora projektu *68hc08* zverejnenom na webe OpenCores. Vstupnými obvodmi sú vodič hodinového signálu CLK, resetovací vodič RST, vodič prerušenia IRQ a vstupný 8-bitový vektor pre dáta s názvom DATA_IN. Keďže tento projekt implementuje iba jadro mikroprocesora, implementácia radiča prerušení chýba, rovnako ako aj periférne zariadenia samotné. Ďalšou dôležitou vecou, ktorá nie je v tomto projekte implementovaná je pamäťový systém.

Výstup obvodu tvorí 16-bitový adresovací vektor ADDR, 4-bitový stavový vektor STATE, 8-bitový dátovy vektor DATA_OUT a vodič WR riadiaci zápis a čítanie pre pamäťový sys-

tém.

Oba dátové vektory slúžia pre čítanie alebo zápis dát z/do pamäťového systému. Adresovým vektorom sa vystavuje adresa miesta v pamäťovom systéme.



Obrázok 3.1: Jadro mikrokontroléru M68HC08 projektu *68hc08* zverejnenom na webe OpenCores.

3.2 Koncept radiča DMA pre mikrokontrolér HCS08

K implementácii radiča DMA som sa rozhodol vychádzať z DMA radiča Intel 8237, používaný v zberniciach ISA. Pre potreby mikrokontroléru z rodiny HCS08 však musím vykonať niekoľko zmien oproti kontroléru Intel 8237, ktoré sa týkajú najmä spôsobu adresácie operačnej pamäte a módov DMA prenosu. Ďalšie zmeny je nutné vykonať v architektúre rodiny HCS08 a to konkrétne v periférnych zariadeniach, aby boli schopné prevádzať DMA prenos. Poslednou dôležitou zmenou v architektúre HCS08 je úprava systémových zberníc.

3.2.1 Zbernice architektúry

Za účelom jednoduchého pripojenia radiča DMA do architektúry HCS08, zavediem systém zberníc, ktorý bude vychádzať z konceptu projektu *68hc08* a konceptu podobnom zberniciam ISA, a zároveň bude funkčne ekvivalentný k originálnej architektúre HCS08.
Procesor, pamäťový systém a všetky periférne zariadenia vrátane DMA radiča budú pripojené na spoločnú adresovú, dátovú a riadiacu zbernicu, podobne ako tomu je v zbernici ISA.

Adresová zbernica bude slúžiť pre adresáciu celého pamäťového priestoru tj. vrátane operačnej pamäte, pamäte FLASH, registrov radiča DMA a periférnych zariadení a ďalších miest v adresovom priestore, ktoré sú popísané v kapitole 2.3.3. Šírka tejto zbernice bude rovnaká ako v originálnej architektúre, čiže 16 bitov. Adresy na adresovú zbernicu bude môcť vystavovať buď mikroprocesor alebo radič DMA.

Dátová zbernica, ako z názvu vyplýva, bude využívaná na prenos dát. Na túto zbernicu budú pripojené všetky zariadenia vrátane periférnych, DMA radiča, mikroprocesora a pamäťového systému. Každé zo zariadení môže zo zbernice čítať alebo na ňu zapisovať. Jej šírka opäť zostáva rovnaká ako v pôvodnej architektúre, tj. 8 bitov.

Za pomoci riadiacej zbernice bude ovládaný prístup k adresovej a dátovej zbernici, aby nedochádzalo ku kolíziam zápisu. Takisto bude slúžiť pre riadenie toku dát na dátovej zbernici.

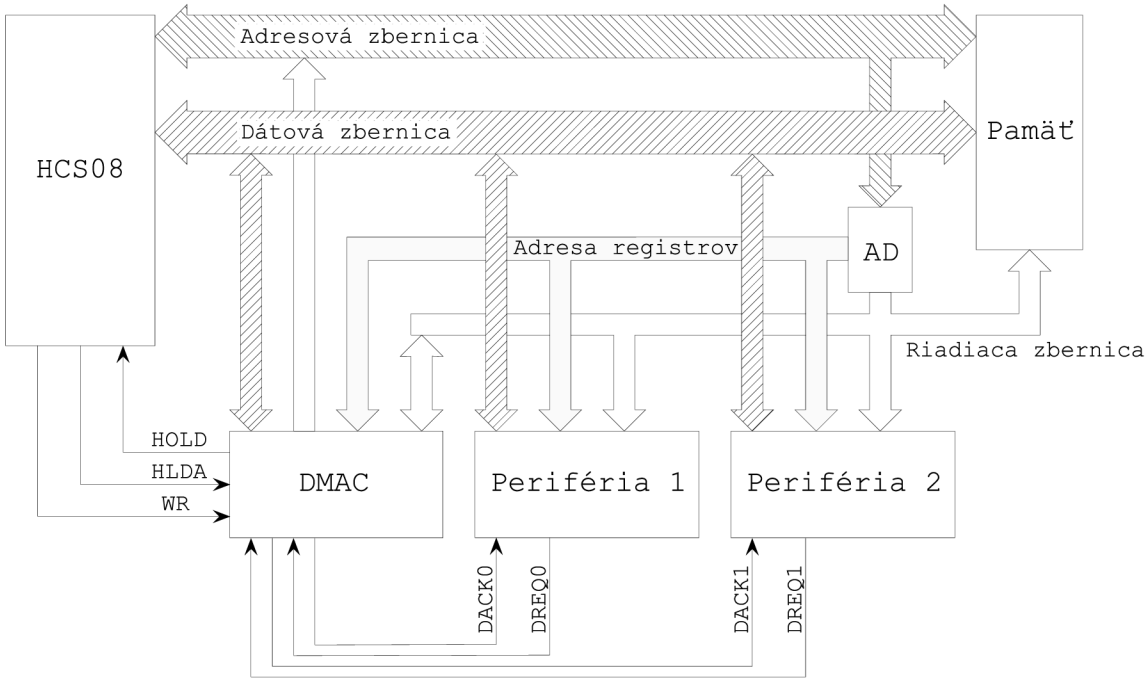
3.2.2 Vlastnosti

Ako už bolo spomenuté, základná štruktúra DMA radiča bude vychádzať z radiča Intel 8237. Zásadný rozdiel oproti tomuto kontroléru bude v adresácii registrov radiča DMA a taktiež v spôsobe vystavania adries na adresovej zbernici počas DMA prenosu. Radič bude stavaný presne na šírku dátovej a adresovej zbernice v architektúre, vďaka čomu nebude potrebovať register stránky 74LS373 pre postupné vystavovanie adresy. Tým sa značne zjednoduší proces vystavovania adresy na adresovú zbernicu.

DMA radič bude 4-kanálový, čiže bude k nemu možné pripojiť až štyri periférne zariadenia. Keďže je tento projekt implementovaný v jazyku VHDL, je možné počet kanálov ľubovoľne upravovať podľa potrieb. Každý kanál bude pozostávať z niekoľkých registrov pre nastavovanie a riadenie DMA prenosu.

3.2.3 Bloková schéma zapojenia radiča DMA

Obrázok 3.2 zobrazuje blokovú schému zapojenia DMA radiča v architektúre mikrokontrolérov rodiny HCS08. Radič DMA (na obrázku označený ako DMAC) a rovnako aj všetky periférne zariadenia sú pripojené k spoločnej dátovej a riadiacej zbernici. Na tieto zbernice je pripojený aj mikroprocesor HCS08 a pamäťový systém, s tým že mikroprocesor je na časť riadiacej zbernice pripojený nepriamo, prostredníctvom adresového dekodéra AD.



Obrázok 3.2: Bloková schéma zapojenia DMA radiča v systéme.

Na adresovú zbernicu smie zapisovať iba mikroprocesor a radič DMA. Výlučný prístup k zbernici zaisťujú vyhradené signály HOLD a HLDA. Keď radič DMA obdrží žiadosť na prenos dát od niektorého z periférnych zariadení prostredníctvom signálu DREQ, musí sa procesor najskôr vzdať všetkých systémových zberníc a prejsť do stavu vysokej impedancie.

Nastavením signálu HOLD dáva radič DMA vedieť mikroprocesoru, že niektoré z periférných zariadení chce vykonať DMA prenos. Procesor dokončí rozpracovanú inštrukciu, prejde do stavu vysokej impedancie a nastavi signál HLDA (HOLD Acknowledge), čím dáva radiču DMA vedieť, že mikroprocesor sa vzdal systémových zberníc. Následne môže prenos DMA začať.

3.2.4 Riadiaca zbernica

Riadiaca zbernica zohráva dôležitú úlohu v smerovaní toku dát po dátovej zbernici a taktisto v zaisťovaní výlučného prístupu k zberniciam. Signály riadiacej zbernice zobrazuje obrácok 3.3. Oproti architektúre ISA som použil odlišný systém povolovacích signálov. Namiesto dvojích signálov $\overline{\text{MEMR}}/\overline{\text{MEMW}}$ a $\overline{\text{IOR}}/\overline{\text{IOW}}$ budú základným stavebným prvkom signály EN vychádzajúce z adresového dekodéra AD a dva signály WR, kde jeden nastavuje procesor a druhý radič DMA.

Signál EN povoluje konkrétne zariadenie pre operáciu zápisu alebo čítania. O nastavovanie tohoto signálu sa stará adresový dekodér AD. Konkrétne obrázok 3.3 zobrazuje štyri povolovacie signály EN pomenované podľa toho, do ktorého zariadenia príslušný signál smeruje. Sú to signály DMA_EN, IO1_EN, IO2_EN a MEM_EN.

Zatiaľ čo signál EN povoluje operácie čítania a zápisu, signál WR určuje smer prenosu dát. Nastavenie tohoto signálu do log. 0 znamená, že sa z daného zariadenia bude čítať, nastavenie do log. 1 naopak povolí zápis do tohto zariadenia. V riadiacej zbernici sa nachádzajú dva signály WR a to konkrétne signály CPU_WR a DMA_WR. Signály sú pomenované podľa zariadenia, ktoré tento signál nastavuje.

Riadenie smeru prenosu na dátovej zbernici

Nastavenie niektorého zo signálov DACK znamená, že momentálne prebieha DMA prenos, preto tieto signály slúžia na multiplexovanie riadiacich signálov EN a WR.

Ak momentálne DMA prenos neprebieha (všetky signály DACK sú v log. 0), sú všetky periférne zariadenia a rovnako aj pamäťový systém riadené EN signálmi z adresového dekodéra a signálom CPU_WR nastavovaného procesorom. Riadenie komunikácie na zberniciach je tak v plnej réžii procesora.

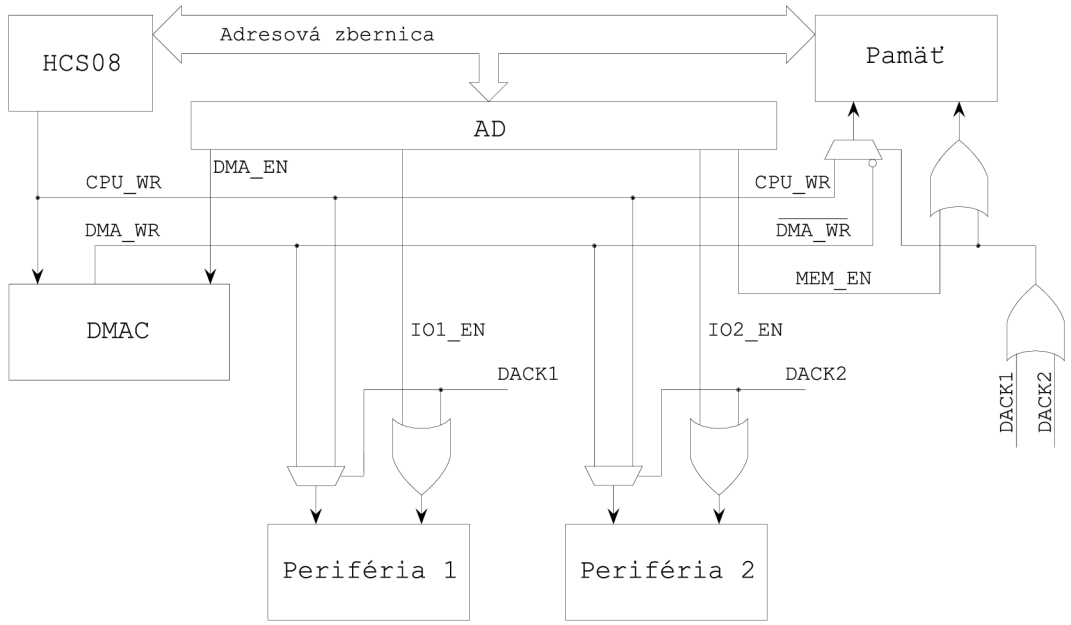
Zariadenie, ktoré obdržalo signál DACK žiadalo o DMA prenos a preto nad ním určite bude realizovaná operácia čítania alebo zápisu. Preto súčasne s nastavením signálu DACK do log. 1, sa musí log. 1 objaviť aj na povolovacom signále EN vedúceho do príslušného periférneho zariadenia. V opačnom prípade musí byť signál EN nastavovaný procesorom prostredníctvom povolovacieho signálu IOx_EN vedúcého z adresového dekodéra. O túto činnosť sa stará hradlo OR, ktoré má na vstupe práve signály IO_EN a DACK a jeho výstup smeruje do odpovedajúceho periférneho zariadenia ako povolovací signál EN.

Podobne je tomu v prípade signálu WR slúžiaceho na riadenie operácií čítania a zápisu. Pre jednoduchosť som sa rozhodol použiť dva oddelené signály, prvý s názvom CPU_WR, ktorý nastavuje procesor a druhý s názvom DMA_WR, ktorý nastavuje radič DMA. Má to praktický učel, pretože DMA prenos sa líši od prenosu dát riadeného procesorom. Zatiaľ čo počas prenosu riadeného procesorom sú dáta vždy prenášané iba medzi procesom a pamäťovým systémom alebo medzi procesom a perifémym zariadením, počas DMA prenosu sa dáta prenášajú medzi perifémym zariadením a pamäťovým systémom. Znamená to, že počas DMA prenosu musí byť WR signál vystavený pre dve jednotky súčasne – pre periférne

zariadenie a pre pamäťový systém s tým, že hodnoty signálov pre obe jednotky musia byť navzájom opačné.

Počas DMA prenosu budú výstupy procesora v stave vysokej impedancie, takže signál CPU_WR sa bude správať ako úplne odpojený vodič. Pre jednoduchosť aj počas simulácie nebudem však používať hradlo OR ako tomu je v prípade signálu EN ale budem hodnoty signálov WR multiplexovať do jednotlivých zariadení. Na vstup multiplexorov privádzam oba riadiace signály CPU_WR i DMA_WR a multiplexor riadim signálom DACK. Počas DMA prenosu bude na výstupe multiplexora hodnota DMA_WR, v opačnom prípade to bude hodnota signálu CPU_WR. Do pamäťového systému je hodnota signálu DMA_WR privádzaná skrz invertor, aby sa zabezpečil správny smer prenosu medzi pamäťovým systémom a periférnym zariadením.

Multiplexor pamäťového systému je riadený všetkými signálmi DACK súčasne, pretože pamäť vždy figuruje v DMA prenose. Túto funkciu zabezpečuje hradlo OR, do ktorého sú privedené všetky signály DACK a jeho výstup ovláda spomínaný multiplexor.



Obrázok 3.3: Zapojenie signálov riadiacej zbernice.

Adresový dekodér AD

Vo všeobecnej architektúre mikrokontrolérov rodiny HCS08 sú registre periférnych zariadení mapované do prvých 128 bytov operačnej pamäte[1]. Znamená to, že dolných 7 bitov adresovej zbernice je možné použiť pre adresáciu registrov periférnych zariadení. Horných 9 bitov adresovej zbernice potom udáva, kam sa majú dáta zapísať. Ak je všetkých 9 horných bitov nulových, jedná sa o prístup k registrom periférnych zariadení. Akákoľvek iná hodnota znamená prístup do operačnej pamäte.

[1]Počet použitých bytov pre mapovanie registrov sa líši každým modelom mikrokontroléra HCS08. Niektoré z nich dokonca využívajú viac ako 128 bytov, nikdy však nie viac ako 256 aby sa ponechal priestor aj pre operačnú pamäť s priamym prístupom.

Adresový dekodér tvorí stromové zapojenie niekoľkých multiplexorov, ktoré zabezpečujú správny výber zariadenia s ktorým chceme pracovať. Výber funguje iba na základe adresy vystavenej na adresovej zbernici. Adresový dekodér nastavuje povolovacie signály EN, pričom vždy môže byť aktívny (v log. 1) iba jeden z nich. Každé z periférnych zariadení, pamäťový systém a rovnako aj radič DMA má vyhradený jeden povolovací signál EN.

3.2.5 DMA kanály

Kanálom DMA sa rozumie samostatné rozhranie pre priamy prístup do pamäte. V koncepte mojho radiča budú 4 kanály a každý z nich bude umožňovať prenos dát oboma smermi tj. z operačnej pamäte do periférneho zariadenia i naopak. Každé periférne zariadenie bude napojené na jeden kanál, čo umožní pripojiť až štyri periférne zariadenia resp. paralelne prevádzať prenos dát na štyroch periférnych zariadeniach súčasne s možnosťou priameho prístupu do pamäte.

Ak DMA radič obdrží žiadost o priamy prístup do pamäte na viacero kanáloch súčasne, musí ich odbaviť postupne podľa určitého poradia. K tomuto účelu budú kanály očíslované od 0 do 3 a jednotlivé žiadosti o priamy prístup do pamäte budú odbavované podľa pevnej priority, ktorú určuje číslo kanála (kanál s nižším číslom bude mať vyššiu prioritu).

Každý DMA kanál bude pozostávať z dvoch 16-bitových adresových registrov, pričom jeden bude slúžiť k obsluhe žiadostí, ktoré chcú z operačnej pamäte čítať, a druhý k obsluhe žiadostí, ktoré chcú do operačnej pamäte zapisovať. Tento koncept umožní obsluhovať poloičný i plný duplex. Alternatívnym riešením by bolo vyhradenie dvoch kanálov pre jedno periférne zariadenie, jeden pre čítacie a druhý pre zápisové operácie.

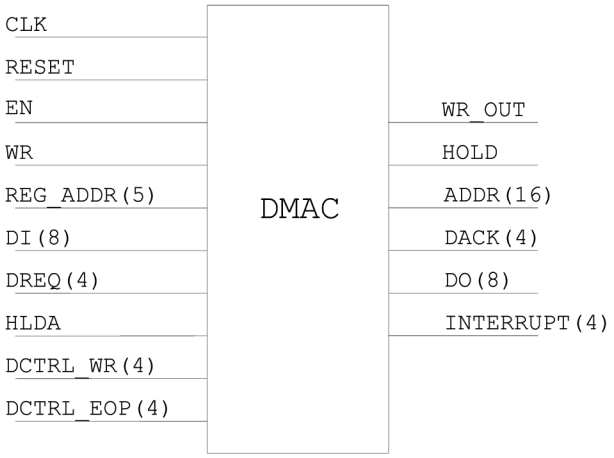
Po obdržaní žiadosti o priamy prístup do pamäte musí DMA radič vedieť, ktorým smerom chce periférne zariadenie realizovať prenos dát, aby vedel ktorý z dvoch adresových registrov má použiť k vystaveniu adresy na adresovú zbernicu. K tomuto účelu bude súčasťou každého DMA kanála špeciálny vodič, ktorý bude tvoriť doplnkovú informáciu k žiadosti o priamy prístup do pamäte.

3.2.6 Rozhranie radiča DMA

Oproti radiču DMA Intel 8237 som sa rozhodol za účelom zjednodušenia urobiť niekoľko zmien v samotnom riadení DMA prenosu. Táto zmena sa týka signalizovania ukončovania prenosu. V radiči Intel 8237 sa k signalizácii ukončenia prenosu používa obojsmerný signál $\overline{\text{EOP}}$.

Môj radič DMA bude fungovať iba ako zariadenie, ktorého účelom bude vystavovať správne adresy na adresovú zbernicu a prípadne nastavovať signály prerušenia v prípade dokončenia DMA prenosu. Jeho činnosť nebude možné zakázať ako je tomu v prípade Intel 8237, pretože v jednoduchých mikrokontroléroch, ktoré sa obvykle používajú bez operačného systému sa javí táto možnosť bezvýznamná. DMA prenos sa bude zapínať/vypínať v každom perifémom zariadení individuálne počas jeho programovania k prenosu. Užívateľ si tak bude môcť počas nastavovania periférneho zariadenia zvoliť, či chce daný prenos realizovať štandardným spôsobom (tzv. pollingom) alebo prostredníctvom priameho prístupu do pamäte.

Obrázok 3.4 zobrazuje rozhranie mnou navrhnutého radiča DMA. Jeho rozhranie možno rozdeliť na tri časti – signály riadiacej zbernice, adresovacie a dátové signály registrov a signály pre obsluhu žiadostí priameho prístupu do pamäte.



Obrázok 3.4: Vstupy a výstupy radiča DMA

Signály riadiacej zbernice

Základnými vstupnými signálmi riadiacej zbernice sú CLK, RESET, EN a CPU_WR. Signál CLK a RESET tvorí štandardnú dvojicu globálneho hodinového a resetovacieho signálu. Signály EN a CPU_WR slúžia pre riadenie zápisu do registrov radiča DMA alebo ich čítania.

Medzi výstupné signály patrí DMA_WR, ktorým radič DMA riadi smer prenosu dát na dátovej zbernici. Činnosť riadiacich signálov som bližšie popísal v kapitole 3.2.4.

Adresovacie a dátové signály registrov

Prístup k registrom radiča zabezpečuje trojica signálov REG_ADDR, DI a DO. Vstupný signál REG_ADDR predstavuje 5-bitový vektor, ktorý je priamo pripojený na dolných 5 bitov adresovej zbernice. Tento vektor slúži na adresáciu registrov radiča DMA a umožňuje adresovať $2^5 = 32$ registrov. Signály DI a DO tvoria 8-bitové vektory, ktoré sú priamo pripojené na dátovú zbernicu a prostredníctvom nich je možné nahrať dáta do požadovaných registrov resp. čítať dáta z požadovaných registrov.

Signály pre obsluhu žiadostí priameho prístupu do pamäte

Signály pre obsluhu žiadostí k priamemu prístupu do pamäte možno rozdeliť na dve časti. Prvú časť predstavujú signály DMA kanála, ktoré tvoria prepojenie medzi radičom DMA a periférnym zariadením. Druhú časť tvoria signály prepájajúce radič DMA a procesor.

DMA kanál tvorí nasledujúca štvorica signálov:

1. DREQ – žiadosť o priamy prístup do pamäte od periférneho zariadenia. Plní rovnakú funkciu ako v radiči Intel 8237 avšak jeho funkcia je doplnená o signál DCTRL_WR, ktorý určuje smer DMA prenosu. Radič DMA tak vie s ktorým adresovým a čítacím registrom má pracovať.

2. DACK – potvrdenie žiadosti pre priamy prístup do pamäte, smerujúce do periférneho zariadenia. Taktiež plní rovnakú funkciu ako v radiči Intel 8237.

3. DCTRL_EOP – špeciálny signál, ktorý je modifikáciou pôvodného signálu EOP v radiči Intel 8237. Z každého periférneho zariadenia smeruje jeden tento signál do radiča DMA a jeho účelom je informovať radič DMA o dokončení DMA prenosu. Významným rozdielom oproti radiču Intel 8237 je, že môj radič nebude vedieť koľko dát sa bude počas DMA prenosu prenášať. Túto informáciu bude vedieť jedine radič periférneho zariadenia, ktorú nastaví programátor pred zahájenim prenosu na príslušnom perifémom zariadení.

   Po prenesení všetkých dát pošle radič periférneho zariadenia prostredníctvom tohoto signálu informáciu, že dáta boli prenesené. DMA radič na základe tejto informácie nastaví vo svojich registroch príznak o dokončení prenosu a v prípade povoleného prerušenia pre príslušný DMA kanál taktiež vystaví prerušenie signálom INTERRUPT.

   Súčasne so signálom DCTRL_EOP sa nastavuje aj signál DCTRL_WR, ktorý určuje, či bola dokončená čítacia alebo zapisová operácia.

   Špeciálnou operáciou je súčasné nastavenie signálu DREQ a DCTRL_EOP. Táto situácia znamená, že radič perifémych zariadení úplne dokončil svoju operáciu a prenos DMA príslušného periférneho zariadenia sa vypína.

4. DCTRL_WR – funguje ako doplnková informácia k signálom DREQ a DCTRL_EOP a svojou hodnotou určuje smer prenosu dát medzi perifémym zariadením a operačnou pamäťou.

Týmto konceptom sa zabezpeči možnosť realizácie rozdielneho počtu čítacích a zápisových operácií, čo môže byť výhodné najmä pre plne duplexné periférne zariadenia kedy užívateľ požaduje prijať iné množstvo dát ako odoslať. Rovnako je tento koncept výhodný v prípade, ak bude periférne zariadenie operovať v režime slave, kedy množstvo a smer dát určuje externé zariadenie, ale programátor mikrokontroléra HCS08 chce mať túto sitáciu pod kontrolou.

Signály prepájajúce radič DMA a procesor sú pomenované rovnako ako v radiči Intel 8237 tj. HOLD a HLDA, taktiež ich funkcia a význam je úplne totožná.

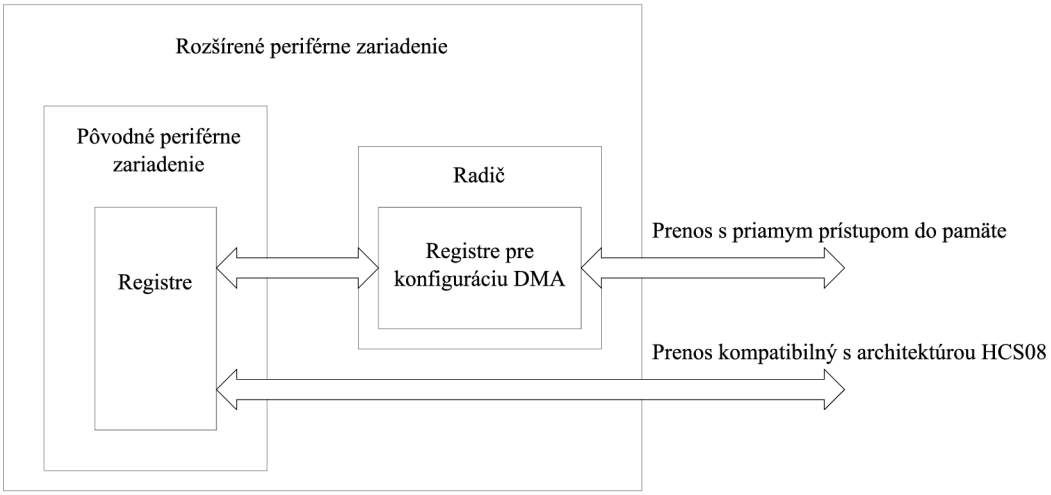
Posledným dôležitým signálom je 16-bitový vektor ADDR, ktorý je priamo pripojený na adresovú zbernicu. Prostredníctvom tohoto vektora vystavuje radič DMA hodnoty adresových registrov na adresovú zbernicu počas DMA prenosu. Adresa na tomto vektore je vystavená iba po dobu, kedy je niektorý zo signálov DACKx v log. 1. V opačnom prípade, je tento vektor prepnutý do stavu vysokej impedancie aby nedochádzalo ku kolíziam na adresovej zbernici.

3.3 Rozšírenie perifémych zariadení

Periférne zariadenia je nutné rozšíriť o koncept, ktorý umožní naprogramovať dané zariadenie pre prenos dát prostredníctvom priameho prístupu do pamäte, ale zároveň zachová pôvodný koncept programovania zariadenia a prenosu dát, a teda bude spätne kompatibilný s pôvodnou architektúrou. Týmto rozšírením je radič periférneho zariadenia, ktorý figuruje ako nadstavba nad perifémym zariadením.

Obrázok 3.5 zachytáva koncept rozšíreného periférneho zariadenia, ktoré v sebe zapúzdruje štruktúru pôvodného zariadenia a radič. Spätne kompatibilný prenos dát nevyužíva k svojej

činnosti radič, ale je realizovaný priamo pôvodným zariadením. Riadiť a konfigurovať kompatibilný prenos je možné pomocou čítania a zápisov do registrov pôvodného periférneho zariadenia.



Obrázok 3.5: Schéma rozšíreného periférneho zariadenia

Radič periférneho zariadenia obsahuje sadu registrov, prostredníctom ktorých je možné nakonfigurovať zariadenie pre prenos dát s využitím technológie priameho prístupu do pamäte. Tento radič následne riadi prenos dát, pričom využíva registre pôvodného periférneho zariadenia, z ktorých sám číta alebo do nich zapisuje. Po nakonfigurovaní a spustení zariadenia pre prenos dát technológiou priameho prístupu do pamäte sú registre pôvodného rozhrania nedostupné až do doby, kým sa neprenesú všetky požadované dáta. Dokončenie prenosu je ohlásené radiču DMA a ten môže následne vyvolať prerušenie.

Kapitola 4

Implementácia

V tejto kapitole popíšem implementáciu navrhnutého riešenia. Prácu som implementoval v jazyku VHDL a jednotlivé komponenty architektúry som rozdelil do niekoľkých modulov.

Pri implementácii som vychádzal z modelu MC9S08SG16[2] (skrátene SG16) rodiny mikrokontrolérov HCS08. Tento model je určený na všeobecné využitie a je uložený v púzdre s minimálnym počtom pinov. SG16 pozostáva z 1 KB operačnej pamäte RAM a 16 KB FLASH pamäte určenej pre program. Ďalej obsahuje periférne zariadenia IIC, SPI, SCI, AD prevodník, časovače RTC a MTIM, generátor pulzno-šírkovej modulácie TPM a analógový komparátor ACMP. Z uvedených periférií som implementoval iba komunikačné rozhrania IIC a SPI.

4.1 Modul `cpu.vhd`

Tento modul obsahuje jadro mikroprocesora 68HC08 popísaného v kapitole 3.1. Hlavná entita je pomenovaná X68UR08 a zapúzdruje v sebe samotné jadro. K rozhraniu entity som pridal 8-bitový vektor prerušení, vstupný port DMA_HOLD a výstupný port DMA_HLDA. Implementáciu som taktiež doplnil o nové operačné kódy, pridané do architektúry HCS08.

Po obdržaní požiadavky signálom DMA_HOLD mikroprocesor dokončí rozpracovanú inštrukciu, pozastaví svoju činnosť a nastaví signál DMA_HLDA. Signál DMA_HLDA je nastavený vždy po dobu dvoch hodinvých cyklov, počas ktorých sa prenesie jeden byte z operačnej pamäte do periférneho zariadenia alebo naopak.

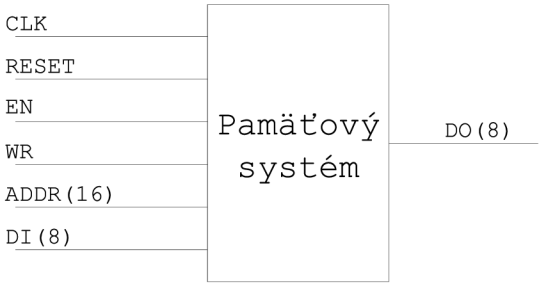
V architektúre HCS08 je mikroprocesor taktovaný na dvojnásobok frekvencie BUSCLK, avšak mikroprocesor X68UR08 je postavený tak, aby navonok zachovával identickú činnosť pri polovičnej frekvencii.

4.2 Modul `memory.vhd`

Tento modul tvorí pamäťový systém architektúry. Celý tento systém je vystavaný z pamäťových komponent RAMB16_S9 (18 Kb pamäťové moduly BlockRAM), ktoré v tomto zapojení umožňujú adresovať 2048 8-bitových položiek. Z pamäťových modulov nevyužívam paritné bity. Obrázok 4.1 zobrazuje rozhranie hlavnej entity pamäťového systému, ktorá nesie názov MEMORY.

Implementácia FLASH pamäte pozostáva z ôsmich modulov RAMB16_S9, ktoré dohromady tvoria 16 KB blok pamäte. Moduly tvoriace FLASH pamäte majú napevno nastavený

vstupný signál WE (Write Enable) do log. 0, čo znemožní zápis do týchto pamätí.



Obrázok 4.1: Rozhranie entity MEMORY

Jeden modul RAMB16_S9 je vyhradený pre mapovanie registrov a pamäte RAM. Od adresy 0x000 sú namapované registre stránky s priamym prístupom, nasledované 1024-bytovým blokom pamäte RAM. Na adrese 0x480 začína blok registrov s nepriamym prístupom.

Pamäťový systém je doplnený o multiplexor, ktorý na základe horných 5 bitov adresy vystavenej na adresovej zbernici zvolí, s ktorou inštanciou BlockRAM modulu sa bude pracovať. Jednotlivé inštancie sú pomenované ako `MEMXX`, kde `XX` je hexadecimálna hodnota udávajúca horných 8 bitov adresy, od ktorej príslušná inštancia pokrýva pamäťový priestor.

Inicializácia pamäťových modulov je vykonaná za pomoci generických premenných INIT_00 – INIT_3F, pomocou ktorých je možné nahrávať program do FLASH pamäte.

4.3 Modul `ad.vhd`

Modul `ad.vhd` implementuje adresový dekodér, ktorého účelom je na základe adresy vystavenej na adresovej zbernici správne vybrať zariadenie, s ktorým sa má pracovať, čo umožňuje mapovanie registrov do pamäťového priestoru. Hlavná entita je pomenovaná AD a pozostáva z jedného vstupného 16-bitového portu a jedného 4-bitového výstupného portu. Adresový dekodér tvori stromové zapojenie niekoľkých multiplexorov. Jednotlivé signály výstupného portu slúžia ako výberové signály jednotlivých zariadení – IIC, SPI, pamäťový systém a radič DMA.

4.4 Modul `dma.vhd`

V tomto module sa nachádza implementácia navrhnutého radiča DMA zapúzdrená v entite s názvom DMAC. Rozhranie tejto entity popisuje kapitola 3.2.6 a jednotlivé porty vyobrazuje schéma s číslom 3.4.

DMA kanál č. 0 je vyhradený pre rozhranie SPI a kanál č. 1 pre rozhranie IIC. Druhý a tretí kanál je nevyužitý a možno ich použit k pridávaniu ďalších periférnych rozhraní.

4.5 Modul `i2c.vhd`

Modul `i2c.vhd` implementuje celé rozhranie IIC vrátane radiča prenosu s priamym prístupom do pamäte. Tento modul pozostáva z dvoch entít pomenovaných I2C a I2C_CORE. Hlavnou

entitou je I2C a jej základné rozhranie je totožné s rozhraním pamäťového systému znázornenom na obrázku 4.1. Toto rozhranie je naviac doplnené o výstupný prerušovací signál INTERRUPT a taktiež o signály pre riadenie DMA prenosov, ktoré pozostávajú z výstupného signálu DREQ a výstupných signálov DACK, DCTRL_EOP a DCTRL_WR. Ďalšími doplňujúcimi signálmi sú obojsmerné signály I2C_SCL a I2C_SDA, ktoré spolu reprezentujú sériové komunikačné rozhranie IIC.

Entita I2C implementuje registre rozhrania IIC a rovnako aj registre radiča periférneho zariadenia. Prostredníctom týchto registrov je potom ovládaný modul I2C_CORE, ktorý je zapúzdrený v entite I2C. Modul I2C_CORE implementuje samotné jadro rozhrania IIC a pozostáva z dvoch konečných automatov, jeden pre režim master a druhý pre režim slave.

Implementované rozhranie IIC nepodporuje multi–master mód a ani s ním spojený mechanizmus synchronizácie hodinových signálov.

4.6 Modul `spi.vhd`

Tento modul implementuje komunikačné rozhranie SPI vrátane radiča periférneho zariadenia, určeného k riadeniu prenosu s priamym prístupom do pamäte. Modul pozostáva z dvoch entít – SPI a SHIFT_MODUL. Hlavnou entitou je SPI a jej základné rozhranie je totožné s rozhraním pamäťového systému znázornenom na obrázku 4.1. Toto rozhranie je naviac doplnené o výstupný prerušovací signál INTERRUPT a taktiež o signály pre riadenie DMA prenosov, ktoré pozostávajú z výstupného signálu DREQ a výstupných signálov DACK, DCTRL_EOP a DCTRL_WR. Ďalšími doplňujúcimi signálmi rozhrania sú obojsmerné signály SPI_SS, SPI_SPSCK, SPI_MISO a SPI_MOSI, ktoré spolu reprezentujú sériové komunikačné rozhranie SPI.

Entita SPI implementuje registre rozhrania SPI a taktiež aj registre radiča periférneho zariadenia. Jej úlohou je predspracovanie signálov SPI rozhrania, ktorými následne riadi posuvný modul implementovaný v entite SHIFT_MODUL. Tento posuvný modul stojí na konečnom automate a je zapúzdrený do entity SPI. Obsahuje implementáciu posuvného registra a jednoslovných bufferov pre vysielanie a príjem dát.

Implementované rozhranie SPI nepodporuje špeciálne nastavenia pinov ani možnosť pokračovania prenosu v prípade prechodu mikrokontroléra do tzv. *wait* módu.

4.7 Modul `top.vhd`

Modul `top.vhd` predstavuje top entitu architektúry. Zapúzdruje v sebe mikroprocesor, pamäťový systém, adresový dekodér a rozhrania SPI a IIC. Názov tejto entity je TOP a jej rozhranie pozostáva z nasledujúcich signálov:

- Vstupné hodinový signál CLK pre namapovanie hodinového signálu z FPGA. Frekvencia tohto hodinového signálu je úmerná frekvencii BUSCLK.
- Vstupný resetovací signál RESET.
- Vstupno/výstupné signály periférneho rozhrania SPI – SPI_SS, SPI_SPSCK, SPI_MISO a SPI_MOSI.
- Vstupno/výstupné signály periférneho rozhrania IIC – I2C_SCL a I2C_SDA.

Kapitola 5

Registre a konfigurácia DMA prenosu

Prvá časť tejto kapitoly sa zaoberá popisom registrov radiča DMA a taktiež popisom registrov pridaných do rozhrania IIC a SPI, ktoré umožňujú nakonfigurovať príslušné rozhranie k prenosu dát prostredníctvom priameho prístupu do pamäte. Druhá časť kapitoly definuje postup konfigurácie periférnych zariadení pre uskutočnenie DMA prenosov.

5.1 Registre radiča DMA

Táto sekcia popisuje registre mnou navrhnutého radiča DMA. Registre som pomenoval anglickými názvami pre jednotnosť s architektúrou HCS08. Táto sada registrov je odlišná od registrov radiča Intel 8237.

5.1.1 DMA Interrupt Enable Register

Tento register slúži pre zapínanie alebo vypínanie prerušení od jednotlivých DMA kanálov. Nastavenie niektorého z bitov DIEx do log. 1 zapína prerušenie na danom kanáli. Prerušenie na príslušnom kanáli je vyvolané po obdržaní DCTRL_EOP signálu a následnom splnení podmienok nastavených v registri DMAIC.

	7	6	5	4	3	2	1	0
R	0	0	0	0	DIE3	DIE2	DIE1	DIE0
W	-	-	-	-				
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.1: DMA Interrupt Enable Register (DMAIE)

5.1.2 DMA Interrupt Control Register

Týmto registrom sa nastavujú podmienky, za ktorých má byť vyvolané prerušenie na jednotlivých kanáloch po obdržaní DCTRL_EOP signálu od príslušného periférneho zariadenia. Podmienku jedného kanálu tvorí dvojica bitov WIx (Write Interrupt) a RIx (Read Interrupt), a podľa kombinácie môže nadobúdať niektorý z nasledujúcich štyroch významov:

1. WIx = 0, RIx = 0 → Prerušenie sa vyvolá vždy po obdržaní DCTRL_EOPx signálu. Na hodnote DCTRL_WRx nezáleží. Po vyvolaní tohoto prerušenia môže periférne zariadenie ešte stále prenášať dáta a preto nie je možné zahájiť nový DMA prenos.

2. WIx = 0, RIx = 1 → Prerušenie sa vyvolá po obdržaní DCTRL_EOPx signálu iba vtedy, ak je hodnota DCTRL_WRx v log. 0. Znamená to, že všetky požadované dáta z operačnej pamäte boli prostredníctvom priameho prístupu do pamäte prenesené do periférneho zariadenia. Dáta sa z operačnej pamäte čítajú v predstihu a po obdržaní tohoto prerušenia je možné, že ešte neboli všetky načítané dáta prenesené cez rozhranie periférneho zariadenia. Preto nie je možné okamžite zahájiť nový DMA prenos.

3. WIx = 1, RIx = 0 → Prerušenie sa vyvolá po obdržaní DCTRL_EOPx signálu iba vtedy, ak je hodnota DCTRL_WRx v log. 1. Znamená to, že periférne zariadenie obdržalo požadované množstvo dát a všetky tieto dáta boli prostredníctom priameho prístupu do pamäte prenesené do operačnej pamäte. Ani v tomto prípade nemusel byť prenos dát cez rozhranie periférneho zariadenia dokončený a preto po obdržaní tohoto prerušenia nemožno okamžite zahájiť nový DMA prenos.

4. WIx = 1, RIx = 1 → Prerušenie sa vyvolá iba vtedy, ak periférne zariadenie dokončilo svoju činnosť a je pripravené k zahájeniu ďalšieho prenosu. Táto situácia nastáva po obdržaní signálu DCTRL_EOPx spolu s nastaveným signálom DACKx. Obdržanie tejto špeciálnej dvojice signálov znamená, že zariadenie buď prenieslo všetky požadované dáta alebo sa počas prenosu vyskytla chyba a prenos bol ukončený. Danú situáciu je nutné skontrolovať za pomoci stavových registrov periférneho zariadenia a radiča DMA. Toto nastavenie sa odporúča používať aj v prípade jednosmerného prenosu, kedy sa po jeho dokončení požaduje zahájenie prenosu ďalšieho bloku dát.

	7	6	5	4	3	2	1	0
R W	WI3	RI3	WI2	RI2	WI1	RI1	WI0	RI0
Reset	1	1	1	1	1	1	1	1

Tabuľka 5.2: DMA Interrupt Control Register (DMAIC)

Použitie iného ako defaultného nastavenia podmienok má význam v prípade, ak periférne zariadenie bude z operačnej pamäte čítať rozdielne množstvo dát než do nej zapisovať. V tomto prípade si môže programátor zvoliť, kedy chce byť informovaný o dokončení prenosu, avšak musí brať na zreteľ fakt, že DMA prenos v momente príchodu prerušenia ešte nemusel byť úplne dokončený a nie je možné zahájiť nový prenos. Čakanie na úplné dokončenie prenosu je možné následne realizovať zmenou prerušovacej podmienky v obslužnej rutine prerušenia na hodnotu WIx = 1 a RIx = 1.

5.1.3 DMA Direction Register

DMA Direction Register slúži k nastaveniu smeru, ktorým sa budú posúvať bázové adresy po každom prenesenom slove. Smer je možné nastaviť pre každý adresový register zvlášť. Bity WD3 – WD0 nastavujú smer posunu adries pre jednotlivé kanály, z ktorých sa počas priameho prístupu do pamäte číta. Podobne bity RD3 – RD0 nastavujú smer posunu adries pre jednotlivé kanály, do ktorých sa bude zapisovať.

Defaultná hodnota (log. 0) je nastavená pre inkrementáciu adries. Pre dekrementáciu je nutné príslušný bit nastaviť do log. 1.

	7	6	5	4	3	2	1	0
R W	WD3	WD2	WD1	WD0	RD3	RD2	RD1	RD0
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.3: DMA Direction Register (DMADIR)

5.1.4 DMA Status Register 1

Bity tohoto registra slúžia pre signalizáciu dokončenia DMA prenosu na príslušnom kanáli. Jednotlivé bity WTCF3 – WTCF0 (Write Transfer Completed Flag) sa automaticky nastavujú do log. 1 po prenesení všetkých dát z periférneho zariadenia do operačnej pamäte. Podobne bity RTCF3 – RTCF0 (Read Transfer Completed Flag) sa nastavujú po prenesení všetkých dát z operačnej pamäte do periférneho zariadenia.

Nastavenie týchto bitov zabezpečujú signály DCTRL_EOP3 – DCTRL_EOP0 doplnené o hodnotu vystavenú na signáloch DCTRL_WR3 – DCTRL_WR0.
Zápis log. 1 do niektorého z bitov tohoto registra vynuluje príslušný bit.

	7	6	5	4	3	2	1	0
R W	WTCF3	WTCF2	WTCF1	WTCF0	RTCF3	RTCF2	RTCF1	RTCF0
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.4: DMA Status Register (DMAS1)

5.1.5 DMA Status Register 2

Bity DIF3 – DIF0 (DMA Interrupt Flag) sa nastavujú individuálne pre každý kanál automaticky pri vyvolaní prerušenia. Hodnota log. 1 na niektorom z bitov znamená, že príslušný kanál žiada o prerušenie. Bit prislúchajúceho kanála musí byť softwarovo vynulovaný v obslužnej rutine zápisom log. 1. Vyvolanie prerušenia je spôsobené dokončením prenosu dát a následným splnením podmienky nastavenej v registri DMAIC za predpokladu, že je prerušenie povolené v registri DMAIE.

	7	6	5	4	3	2	1	0
R	0	0	0	0	DIF3	DIF2	DIF1	DIF0
W	-	-	-	-				
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.5: DMA Status Register (DMAS2)

5.1.6 DMA Read Address High/Low Register

Nasledujúca dvojica registrov uchováva hornú a dolnú časť adresy, z ktorej sa budú počas priameho prístupu do pamäte čítať dáta. Počas DMA prenosu vystaví radič DMA hodnotu

týchto dvoch registrov na adresovej zbernici ako 16-bitovú hodnotu. Po prenesení jedného slova (tj. po tom ako periférne zariadenie zhodí signál DREQ do log. 0) je adresa v týchto registroch posunutá na nasledujúcu položku v pamäti. Smer posunu určujú bity RD3 – RD0 registra DMADIR.

V radiči DMA sa celkovo nachádzajú 4 dvojice týchto registrov, každá pre jeden kanál.

	7	6	5	4	3	2	1	0
R W	Bit 15	14	13	12	11	10	9	Bit 8
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.6: DMA Read Address High Register (DMARAH3 – DMARAH0)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.7: DMA Read Address Low Register (DMARAL3 – DMARAL0)

5.1.7 DMA Write Address High/Low Register

Nasledujúca dvojica registrov uchováva hornú a dolnú časť adresy, do ktorej sa budú počas priameho prístupu do pamäte dáta zapisovať. Počas DMA prenosu vystaví radič DMA hodnotu týchto dvoch registrov na adresovej zbernici ako 16-bitovú hodnotu. Po prenesení jedného slova je adresa v týchto registroch posunutá na nasledujúcu položku v pamäti. Smer posunu určujú bity WD3 – WD0 registra DMADIR.

V radiči DMA sa celkovo nachádzajú 4 dvojice týchto registrov, každá pre jeden kanál.

	7	6	5	4	3	2	1	0
R W	Bit 15	14	13	12	11	10	9	Bit 8
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.8: DMA Write Address High Register (DMAWAH3 – DMAWAH0)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.9: DMA Write Address Low Register (DMAWAL3 – DMAWAL0)

5.2 Registre sériového komunikačného rozhrania IIC

Táto sekcia sa zaoberá popisom registrov sériového komunikačného rozhrania IIC. Pre lepšie porozumenie programovacieho modelu tohoto rozhrania uvádzam stručný prehľad jeho re-

gistrov z modelu SG16 rodiny mikrokontrolérov HCS08. Detailný popis uvedených registrov je možné nájsť v referenčnom manuáli modelu MC9S08SG32[9]:

1. **IIC Adress Register (IICA)** – Obsahuje položky AD7 – AD1, ktoré spolu tvoria 7-bitovú adresu zariadenia. Touto adresou je možné toto zariadenie adresovať na zbernici.

2. **IIC Frequency Divider Register (IICF)** – Slúži pre nastavenie prenosovej frekvencie dát na zbernici.

3. **IIC Control Register 1 (IICC1)** – Tento primárny riadiaci register obsahuje položky pre základné nastavenie periférie. Významnými bitmi sú IICEN, IICIE, MST, TX a TXAK.

   Položka IICEN povoluje alebo zakazuje rozhranie IIC. Zápisom log. 1 do tohto bitu sa spustí prenos dát na zbernici.

   Za pomoci položky IICIE je možné zapnúť prerušenie od rozhrania IIC. Prerušenie je generované po prenesení jedneho bytu dát alebo po rozpoznaní svojej adresy na zbernici. Prerušenie taktiež môže vyvolať strata arbitrácie zbernice.

   Zápis log. 1 do položky MST uvádza zariadenie do režimu master. Pre hodnotu log. 0 je zariadenie v režime slave.

   Položka TX určuje smer prenosu dát. Pre hodnotu log. 1 funguje rozhranie v režime vysielača, v opačnom prípade v režime prijímača.

   Posledným dôležitým bitom je položka TXAK, ktorá pre režim prijímača nastavuje potvrdzovací bit prijatého bloku dát. Po nastavení tohoto bitu do log. 1 bude nesledujúci prijatý blok dát nepotvrdený. Tento mechanizmus umožňuje ukončiť prenos zo strany prijímača. Pre hodnotu log. 0 bude zariadenie prijaté dáta potvrdzovať, čím sa zároveň signalizuje vysielaču, že môže poslať ďalšie dáta.

4. **IIC Status Register (IICS)** – Stavový register, ktorý umožňuje sledovať priebeh posielania dát po zbernici IIC.

5. **IIC Data I/O Register (IICD)** – Dátový register, ktorým je zároveň možné riadiť prijímacie a vysielacie operácie na zbernici IIC. Ak je zariadenie v režime vysielača, zápis do tohoto registra zaháji vysielanie dát. Analogicky čítanie tohoto registra v režime prijímača zaháji prijímanie dát.

6. **IIC Control Register 2(IICC2)** – Predstavuje sekundárny riadiaci register obsahujúci položky GCAEN, ADEXT a 3-bitovú položku AD10 – AD8.

   Bit GCAEN v hodnote log. 1 zapína možnosť adresácie zariadenia prostredníctvom tzv. *general call* adresy, ktorá slúži na hromadné adresovanie viacerých zariadení. *General call* adresa je protokolom definovaná ako 7-bitová postupnosť 0b0000000.

   Log. 1 v položke ADEXT zapína rozžírenie adresy zariadenia na 10 bitov, kedy je 7-bitová adresa z registra IICA zhora doplnená o 3-bitovú položku AD10 – AD8.

Nasledujúce registre sú doplnkom vyššie uvedených registrov, za pomoci ktorých je možné naprogramovať rozhranie pre prenos dát prostredníctvom priameho prístupu do pamäte. Ich umiestnenie je v radiči periférneho zariadenia IIC.

5.2.1 IIC DMA Control Register

Primárnym účelom tohoto registra je zapínanie alebo vypínanie prenosu s priamym prístupom do pamäte. Spusteniu prenosu DMA musí predchádzať nastavenie radiča DMA a taktiež nastavenie pôvodného rozhrania IIC. Postup konfigurácie popisuje kapitola 5.4. Zápisom log. 1 do DMAEN sa spustí prenos dát, čím sa zároveň znemožní prístup k registrom pôvodného rozhrania po dobu celého prenosu, o ktorý sa stará radič periférneho zariadenia. Po prenesení všetkých požadovaných dát alebo pri výskyte chyby počas prenosu sa hodnota DMAEN automaticky vynuluje, čím sa sprístupnia registre pôvodného rozhrania.

Log. 1 atribútu ADEXT zapína 10-bitové adresovanie zariadení na zbernici IIC. Počas jedného prenosu s priamym prístupom do pamäte je možné na zbernici IIC adresovať iba jedno zariadenie.

Bity DAD10 – DAD8 predstavujú horné 3 bity 10-bitovej adresy zariadenia, s ktorým bude rozhranie IIC počas DMA prenosu komunikovať. Tieto bity sa použijú k adresácii zariadenia iba v prípade, kedy je hodnota atribútu ADEXT v log. 1.

	7	6	5	4	3	2	1	0
R	DMAEN	ADEXT	0	0	0	DAD10	DAD9	DAD8
W			-	-	-			
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.10: IIC DMA Control Register(IICDMAC)

5.2.2 IIC DMA Address Register

Na základe komunikačného protokolu rozhrania IIC musí každému prenosu dát predchádzať adresácia slave zariadenia, s ktorým chce master zariadenie komunikovať. Tento register slúži pre nastavenie adresy zariadenia, s ktorým bude prebiehať komunikácia, tj. adresy, ktorú tento modul vyšle na zbernicu IIC po zahájení DMA prenosu. V prípade 7-bitového adresovacieho módu tvoria bity DAD7 – DAD1 celú 7-bitovú adresu. Pri zapnutom 10-bitovom adresovacom móde predstavujú tieto bity dolných 7 bitov adresy, ktoré sú doplnené na 10 bitov za pomoci hodnôt DAD10 – DAD8 z registra IICDMAC.

Nastavenie tejto adresy má význam iba v prípade, ak rozhranie IIC operuje v režime master. V tomto prípade, sa po spustení DMA prenosu automaticky adresuje zariadenie na zbernici vyslaním adresy DAD7 – DAD1, resp. DAD10 – DAD1 v prípade 10-bitového adresovacieho módu. Pre režim slave je nutné nastaviť adresu zariadenia v registroch pôvodného rozhrania IIC.

	7	6	5	4	3	2	1	0
R	DAD7	DAD6	DAD5	DAD4	DAD3	DAD2	DAD1	0
W								-
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.11: IIC DMA Address Register (IICDMAA)

5.2.3 IIC DMA Read Count Register

Tento register udáva množstvo dát v bytoch, ktoré sa majú preniesť počas DMA prenosu z operačnej pamäte do periférneho zariadenia IIC. Maximálna dĺžka dát je 255 bytov. Dáta sa z pamäte čítajú postupne po jednom slove a ukladajú do interného registra radiča periférneho zariadenia, ktorý slúži ako cache pamät o veľkosti jedného slova. Dátové slovo je následne odoslané cez zbernicu IIC do externého zariadenia, s ktorým prebieha komunikácia.

Prednačítanie dát sa realizuje v dostatočnom predstihu, vždy približne jednu periódu T_d pred odoslaním príslušného slova, kde T_d je čas potrebný na odoslanie jedného slova cez IIC zbernicu.

Každým načítaným slovom z operačnej pamäte sa automaticky dekrementuje hodnota registra IICDMARC. Po načítaní všetkých dát z pamäte, tj. po tom ako hodnota registra IICDMARC dosiahne nulu, odošle IIC modul do radiča DMA signál DCTRL_EOP s hodnotou DCTRL_RW nastavenou do log. 0. Tým sa signalizuje dokončenie čítacej operácie a radič DMA v prípade zapnutého prerušenia a splnenia prerušovacej podmienky vyšle prerušovací signál do procesora. Ak je rozhranie v režime master a operuje ako vysielač alebo ak operuje v režime slave a je nulová aj hodnota registra IICDMAWC, odošle sa následne ešte ukončovací signál DCTRL_EOP s nastavenou hodnotou DREQ, čo znamená úplné ukončenie DMA prenosu.

Pri programovaní DMA prenosov je nutné brať na zreteľ, že po tom ako bolo signalizované dokončenie čítacej operácie, je modul IIC ešte určitý čas nedostupný, pretože dáta, ktoré sa načítali z operačnej pamäte sa ešte nestihli preniesť cez zbernicu IIC. Úplné dokončenie prenosu je možné detekovať bitom DMAEN z registra IICDMAC, ktorý sa po dokončení prenosu zhodí do log. 0. Kým je DMAEN v log. 1 nie je možné zahájiť ďalší prenos s priamym prístupom do pamäte. Druhou možnosťou detekcie úplného dokončenia prenosu je povolenie prerušenia na príslušnom kanáli DMA a nastavenie prerušovacej podmienky do defaultnej hodnoty, čo spôsobí vyvolanie prerušenia až po úplnom dokončení prenosu.

	7	6	5	4	3	2	1	0
R W	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0 -
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.12: IIC DMA Read Count Register (IICDMARC)

5.2.4 IIC DMA Write Count Register

Tento register udáva množstvo dát v bytoch, ktoré sa majú preniesť počas DMA prenosu z periférneho zariadenia IIC do operačnej pamäte. Maximálna dĺžka dát je 255 bytov. Dáta sa do pamäte zapisujú postupne, tak ako sú čítané zo zbernice rozhrania IIC. Načítané slovo zo zbernice je uložené v špeciálnom registri nachádzajúcom sa v radiči periférneho zariadenia. Tento register nie je zvonka prístupný a jeho hodnota sa automaticky vystaví na výstupnú dátovú zbernicu v momente, kedy sa hodnota signálu DACK prepne do log. 1.

Po každom zapísanom slove do operačnej pamäte je hodnota registra IICDMAWC automaticky dekrementovaná. Po zapísaní všetkých požadovaných dát do pamäte, tj. po tom ako obsah registra IICDMAWC nadobudne nulovej hodnoty, odošle IIC modul do radiča

DMA signál DCTRL_EOP s hodnotou DCTRL_RW nastavenou do log. 1. Tým sa signalizuje dokončenie zápisovej operácie a radič DMA v prípade zapnutého prerušenia a splnenia prerušovacej podmienky vyšle prerušovací signál do procesora. Ak je rozhranie v režime master a operuje ako prijímač alebo ak operuje v režime slave a je nulová aj hodnota registra IICDMARC, odošle sa následne ešte ukončovací signál DCTRL_EOP s nastavenou hodnotou DREQ, čo znamená úplné ukončenie DMA prenosu.

Rovnako ako v prípade operácie čítania z operačnej pamäte je nutné pamätať na to, že po zapísaní všetkých požadovaných dát do operačnej pamäte môže ešte ďalej prebiehať prenos na zbernici IIC, a preto nie je možné okamžite zahájiť ďalší DMA prenos. Pre signalizáciu úplného dokončenia prenosu platí rovnaký postup ako v prípade registra IICDMARC.

	7	6	5	4	3	2	1	0
R W	WC7	WC6	WC5	WC4	WC3	WC2	WC1	WC0 -
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.13: IIC DMA Write Count Register (IICDMAWC)

5.3 Registre sériového komunikačného rozhrania SPI

Táto sekcia sa zaoberá popisom registrov sériového komunikačného rozhrania SPI. Pre lepšie porozumenie programovacieho modelu tohoto rozhrania uvádzam stručný prehľad jeho registrov z modelu SG16 rodiny mikrokontrolérov HCS08. Detailný popis uvedených registrov je možné nájsť v referenčnom manuáli modelu MC9S08SG32[9]:

1. **SPI Control Register 1 (SPIC1)** – Primárny riadiaci register pre nastavenie základných parametrov. Medzi významné položky patrí SPIE, SPE, SPTIE, MSTR, CPOL, CPHA a LSBFE.

   Bit SPIE zapína prerušenie od rozhrania po prijatí jedného slova dát. Žiadosť prerušenia je viazaná na príznak SPRF z registra SPIS.

   Zápis log. 1 do položky SPE zapína zariadenie SPI. Pre log. 0 je zariadenie vypnuté.

   Položka SPTIE zapína prerušenie od rozhrania v prípade prázdneho vysielacieho bufferu. Žiadosť prerušenia je viazaná na príznak SPTEF z registra SPIS.

   Prepínanie medzi režimom master a slave umožňuje položka MSTR. Pre hodnotu log. 1 je zariadenie v režime master, v opačnom prípade operuje v režime slave.

   Dvojica bitov CPOL a CPHA nastavuje polaritu a fázu synchronizačného hodinového signálu prenosu cez rozhranie SPI.

   Pri položke LSBFE nastavenej do log. 0 začína prenos dátového slova po sériovej linke od najvýznamnejšieho bitu. Prenos od najmenej významného bitu je možné nastaviť zápisom log. 1 do LSBFE.

2. **SPI Control Register 2 (SPIC2)** – Sekundárny riadiaci register pre špeciálne nastavenie využitia vodičov periférneho rozhrania SPI. V tejto práci som tento register neimplementoval.

3. **SPI Baud Rate Register (SPIBR)** – Register pre nastavenie prenosovej rýchlosti dát rozhrania SPI.

4. **SPI Status Register (SPIS)** – Stavový register pre sledovanie priebehu posielania dát. Významné položky predstavuje SPRF a SPTEF.

   Položka SPRF sa automaticky nastaví do log. 1 po prijatí jedného slova dát. Dáta sú uchované v špeciálnom prijímacom bufferi, ktorého hodnotu je možné získať čítaním z dátového registra SPID.

   Položka SPTEF v log. 1 signalizuje, že vysielací buffer je prázdny. Tento buffer slúži ako prednačítavací a umožňuje tak kontinuálny prenos dát.

5. **SPI Data Register (SPID)** – Dátový register pre zápis dát k odoslaniu alebo čítaniu prijatých dát. Zápis do tohoto registra automaticky zapíše dáta do vysielacieho bufferu, čím sa zhodí bit SPTEF do log. 0. Podobne čítanie z tohoto registra načíta hodnotu z prijímacieho bufferu, čo spôsobí zhodenie bitu SPRF do log. 0.

Nasledujúce registre sú doplnkom vyššie uvedených registrov, za pomoci ktorých je možné naprogramovať rozhranie pre prenos dát prostredníctvom priameho prístupu do pamäte. Tieto registre sú súčasťou radiča periférneho zariadenia SPI.

5.3.1 SPI DMA Control Register

Jediným účelom tohoto registra je zapínanie alebo vypínanie prenosu s priamym prístupom do pamäte. Spusteniu prenosu DMA musí predchádzať nastavenie radiča DMA a taktiež nastavenie registrov pôvodného rozhrania SPI. Postup konfigurácie popisuje kapitola 5.4. Zápisom log. 1 do DMAEN sa spustí prenos dát, čím sa zároveň znemožní prístup k registrom pôvodného rozhrania po dobu celého prenosu, o ktorý sa stará radič periférneho zariadenia. Po prenesení všetkých požadovaných dát alebo pri výskyte chyby počas prenosu sa hodnota DMAEN automaticky vynuluje, čím sa sprístupnia registre pôvodného rozhrania.

	7	6	5	4	3	2	1	0
R	DMAEN	0	0	0	0	0	0	0
W		-	-	-	-	-	-	-
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.14: SPI DMA Control Register(SPIDMAC)

5.3.2 SPI DMA Read Count Register

Tento register udáva množstvo dát v bytoch, ktoré sa majú preniesť počas DMA prenosu z operačnej pamäte do periférneho zariadenia SPI. Plní rovnakú funkciu ako register IICDMARC popísaný v kapitole 5.2.3. Jediným rozdielom je prednačítavanie dát, ktoré je v prípade rozhrania SPI približne dve periódy T_d pred odoslaním príslušného slova, kde T_d je čas potrebný na odoslanie jedného slova cez SPI rozhranie. Dôvodom tohoto rozdielu je prítomnosť ďalšieho registra určeného k prednačítavaniu dát, ktorý je priamo z definície rozhrania SPI pre rodinu mikrokontrolérov HCS08 súčasťou tohoto rozhrania. Dáta tak môžu byť prednačítané do dvoch registrov.

	7	6	5	4	3	2	1	0
R W	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0 -
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.15: SPI DMA Read Count Register (SPIDMARC)

5.3.3 SPI DMA Write Count Register

Tento register udáva množstvo dát v bytoch, ktoré sa majú preniesť počas DMA prenosu z periférneho zariadenia IIC do operačnej pamäte. V rozhraní SPI plní tento register rovnakú funkciu ako register IICDMAWC rozhrania IIC popísaný v kapitole 5.2.4.

	7	6	5	4	3	2	1	0
R W	WC7	WC6	WC5	WC4	WC3	WC2	WC1	WC0 -
Reset	0	0	0	0	0	0	0	0

Tabuľka 5.16: SPI DMA Write Count Register (SPIDMAWC)

5.4 Konfigurácia periférnych zariadení

Pre prenos dát prostredníctvom rozhrania IIC alebo SPI je dôležité dodržať určitý postup. Mnou navrhnutá architektúra oboch rozhraní je kompatibilná s architektúrou modelu MC9S08SG16, preto postup ich programovania k prenosu dát za pomoci *pollingu*, rovnako ako aj postup k prenosu riadeného prerušením je totožný s postupom popísanom v manuáli[2] alebo v programátorskej príručke mikrokontrolérov HCS08[9].

5.4.1 Rozhranie IIC

K naprogramovaniu rozhrania IIC pre prenos dát prostedníctvom priameho prístupu do pamäte je nutné dodržať nasledujúce kroky:

1. Pred konfiguráciou rozhrania musí byť prípadný predošlý DMA prenos ukončený, tj. hodnota bitu IICDMAC:DMAEN musí byť nulová. Zápis log. 0 do tohoto bitu spôsobí okamžité ukončenie aktuálneho DMA prenosu.

2. Bity IICC:IICEN a IICC:IICIE musia byť nastavené do log. 0.

3. Je nutné nastaviť rýchlosť komunikácie na zbernici IIC pomocou registra IICF.

4. Pre beh zariadenia v režime master musí byť nastavený bit IICC1:MSTR do log. 1. V režime master je taktiež dôležité zvoliť smer prenosu dát prostredníctvom IICC1:TX.

5. Pre beh zariadenia v režime slave musí byť nastavený bit IICC1:MSTR do log. 0. V režime slave je zariadenie adresované iným zariadením a preto je nutné nastaviť jeho adresu v registri IICA. Pre 10-bitovú adresáciu musí byť taktiež nastavený bit IICC2:ADEXT do log. 1 a spolu s ním položky AD10 – AD8 z registrea IICC2. Možnosť adresácie zariadenia pomocou *general call* adresy je možné zapnúť položkou IICC2:GCAEN.

Modul môže byť v režime slave uvedený do režimu vysielača nastavením bitu IICC1:TXAK do log. 1. Tým sa prvý prijatý byte dát potvrdí hodnotou NACK, čo ukončí prenos na zbernici.

6. Ak bude modul figurovať v režime prijímača, je nutné v radiči DMA nastaviť adresu operačnej pamäte, kam sa budú ukladať prijaté dáta. Táto adresa sa nastavuje v dvojici registrov DMAWAH a DMAWAL príslušného DMA kanála. Analogicky pre režim prijímača je nutné nastaviť adresu operačnej pamäte, z ktorej sa budú dáta čítať. Pre nastavenie tejto adresy slúži dvojica registrov DMARAH a DMARAL príslušného DMA kanála.

   V režime slave môže IIC modul operovať v režime prijímača aj vysielača, v tomto prípade je nutné nastaviť obe adresy.

   Spolu s adresou je dôležité nastaviť smer posúvania adresy bitmi DMADIR:RD DMADIR:WD príslušného kanála.

7. Do položiek DMAS1:WTCF a DMAS1:RTCF príslušného kanála musí byť zapísaná log. 1, pre prípadné vynulovanie príznakov dokončenia z predchádzajúceho prenosu.

8. Pre signalizáciu dokončenia DMA prenosu za pomoci prerušenia je nutné pre príslušný DMA kanál zapnúť prerušenie bitom DMAIE:DIE a následne nastaviť prerušovaciu podmienku dvojicou bitov DMAIE:WI a DMAIE:RI.

9. Pre režim master musí byť v registri IICDMAA nastavená adresa zariadenia, s ktorým bude modul komunikovať. Pre 10-bitovú adresáciu je nutné taktiež nastaviť bit IICDMAC:ADEXT do log. 1 a spolu s ním aj horné 3 bity adresy IICDMAC:DAD10 – IICDMAC:DAD8.

10. V neposlednom rade je dôležité nastaviť množstvo dát, ktoré má byť prenesené počas DMA prenosu. Pre režim prijímača sa toto množstvo nastavuje registrom IICDMAWC a pre režim vysielača registrom IICDMARC.

    Pre režim slave je možné nastaviť oba tieto registre, zariadenie tak odošle a príjme požadované množstvo dát. V tomto prípade môže byť každý z registrov nastavený na inú hodnotu. Ak počas prenosu hodnota registra IICDMARC dosiahne nulovú hodnotu, tzn. všetky dáta z operačnej pamäte sa preniesli do IIC modulu, bude modul vysielať nulové hodnoty, pričom sa z pamäte nebudú čítať žiadne ďalšie dáta.

    Hodnota registra IICDMAWC určuje počet bytov, ktoré má rozhranie prijať a následne zapísať do operačnej pamäte. Posledný prijatý byte je potvrdený hodnotou NACK, čo spôsobí ukončenie prenosu. Ak je hodnota registra IICDMAWC od začiatku prenosu nulová, bude prvý prijatý byte potvrdený hodnotou NACK a tento prijatý byte sa následne zahodí.

11. Zápisom log. 1 do IICDMAC:DMAEN sa spustí DMA prenos.

12. Dokončenie DMA prenosu je signalizované zhodením bitu IICDMAC:DMAEN do log. 0 a v prípade zapnutého prerušenia od radiča DMA taktiež vyvolaním prerušenia na príslušnom DMA kanáli. Po dokončení prenosu je nutné skontrolovať hodnoty DMAS1:WTCF a DMAS1:RTCF príslušného DMA kanála. Nulové hodnoty znamenajú, že sa nepreniesli všetky požadované dáta (hodnota registra udáva počet neprenesených bytov) a počas prenosu pravdepodobne došlo k chybe. Zdrojom chýb môže

byť buď strata arbitráže alebo prijatie hodnoty NACK. Tieto dve situácie možno skontrolovať položkami IICS:ARBL a IICS:RXAK.

5.4.2 Rozhranie SPI

K naprogramovaniu rozhrania SPI pre prenos dát prostedníctvom priameho prístupu do pamäte je nutné dodržať nasledujúce kroky:

1. Pred konfiguráciou rozhrania musí byť prípadný predošlý DMA prenos ukončený, tj. hodnota bitu SPIDMAC:DMAEN musí byť nulová. Zápis log. 0 do tohoto bitu spôsobí okamžité ukončenie aktuálneho DMA prenosu.

2. Bity SPIC1:SPIE, SPIC1:SPE a SPIC1:SPTIE musia byť nastavené do log. 0. Týmto nastavením sa zakážu prerušenia a vypne periférne zariadenie.

3. V registri SPIC1 je ďalej nutné príslušne nastaviť bity MSTR, CPOL, CPHA a LSBFE.

4. Pomocou registra SPIBR musí byť nastavená prenosová rýchlosť.

5. V radiči DMA je potrebné nastaviť adresu operačnej pamäte, kam sa budú ukladať prijaté dáta (registre DMAWAH a DMAWAL) a taktiež adresu, z ktorej sa budú dáta čítať (DMARAH a DMARAL). Spolu s adresou musí byť príslušne nastavený smer posunu ukazovateľov adresy za pomoci bitov DMADIR:RD a DMADIR:WD.

6. Do položiek DMAS1:WTCF a DMAS1:RTCF príslušného kanála musí byť zapísaná log. 1, pre prípadné vynulovanie príznakov dokončenia z predchádzajúceho prenosu.

7. Pre signalizáciu dokončenia DMA prenosu za pomoci prerušenia je nutné pre príslušný DMA kanál zapnúť prerušenie bitom DMAIE:DIE a následne nastaviť prerušovaciu podmienku dvojicou bitov DMAIC:WI a DMAIC:RI.

8. Nakoniec je nutné nastaviť množstvo dát, ktoré ma byť prenesené počas DMA prenosu. Požadovaný počet prijatých slov za nastavuje registrom SPIDMAWC a požadovaný počet odoslaných slov registrom SPIDMARC.

   Rozhranie SPI je plne duplexné a prenos je realizovaný na základe vzájomnej výmeny dátových slov. Týmto princípom sa vždy príjme rovnaké množstvo dát ako odošle. Napriek tomuto je možné nastaviť rozhranie aby prijímalo rozdielny počet dátových slov ako vysielalo, prípadne aby operovalo výhradne ako vysielač alebo prijímač. Pre operovanie v režime vysielača musí byť hodnota registra SPIDMAWC nulová. Analogicky pre operovanie v režime prijímača musí byť nulová hodnota registra SPIDMARC.

   Nulová hodnota registra SPIDMAWC spôsobí zahadzovanie prijatých dát. Pri nulovej hodnote registra SPIDMARC bude rozhranie vysielať samé nuly.

9. Zápisom log. 1 do SPIDMAC:DMAEN sa spustí DMA prenos.

10. Dokončenie DMA prenosu je signalizované zhodením bitu SPIDMAC:DMAEN do log. 0 a v prípade zapnutého prerušenia od radiča DMA taktiež vyvolaním prerušenia na príslušnom DMA kanáli. Po dokončení prenosu je nutné skontrolovať hodnoty DMAS1:WTCF a DMAS1:RTCF príslušného DMA kanála. Nulové hodnoty znamenajú, že sa nepreniesli všetky požadované dáta (hodnota registra udáva počet neprenesených bytov) a počas prenosu pravdepodobne došlo k chybe.

Kapitola 6

Analýza a testovanie

V prvej časti tejto kapitoly popíšem analýzu efektivity prenosu dát, ktorú je možné teoreticky dosiahnuť za pomoci implementovaného priameho prístupu do pamäte. Druhá časť kapitoly sa venuje popisu testovania navhrnutého mikrokontroléra.

6.1 Analýza efektivity prenosu dát

Navrhnutý a implementovaný mikrokontrolér umožňuje realizovať prenos dát medzi periférnymi zariadeniami a operačnou pamäťou celkovo troma spôsobmi – pollingom, prerušením alebo priamym prístupom do pamäte. Polling a prenos riadený prerušením je z hľadiska implementácie totožný s pôvodnou architektúrou rodiny HCS08.

- **Polling** – Registre periférneho zariadenia sú neustále testované na nastavenie príznaku dokončenia. Následne sú dáta za účasti procesora prenesené medzi operačnou pamäťou a periférnym zariadením. Tento prenos štandardne využíva procesor na 100%, čo je kritické najmä pre aplikácie pracujúce v reálnom čase.
- **Prenos riadený prerušením** – Po nakonfigurovaní prenosu je procesor uvoľnený až do doby, kým sa neskončí prenos jedného slova dát. Následne je vyvolané prerušenie a dáta musia byť medzi operačnou pamäťou a periférnym zariadením prenesené za účasti procesora pomocou prerušovacej rutiny. Takáto rutina musí pozostávať minimálne z nasledujúcich inštrukcií:

  1. LDA/LDX – Načítanie slova z pamäte alebo z dátového registra periférneho zariadenia do registra A/X.
  2. STA/STX – Zápis slova z registra A/X do dátového registra periférneho zariadenia alebo do operačnej pamäte.
  3. RTI – Ukončenie prerušovacej rutiny.

  Vyvolanie prerušenia vždy spôsobí uloženie obsahu systémových registrov na zásobník. Mikrokontroléru HCS08 tento úkon trvá 11 hodinových cyklov (vzhľadom na frekvenciu BUSCLK). Ukončenie prerušovacej rutiny trvá 9 hodinových cyklov a prenos dát minimálne 6 hodinových cyklov. Obsluha prerušenia tak celkovo trvá minimálne 26 hodinových cyklov.

  Obsluha prerušenia býva často sprevádzaná kontrolou príznakov registrov a nasledovaná minimálne jednou skokovou inštrukciou. Tento úkon trvá najmenej 5 hodinových

cyklov (za použita inštrukcií BRSET alebo BRCLR), čo celkovo dáva 31 hodinových cyklov pre obsluhu prerušenia. V praxi však táto hodnota môže byť ešto o niekoľko desiatok cyklov vyššia.

- **Prenos s priamym prístupom do pamäte** – Mnou navrhnutý prenos riadený prerušením vždy pozastaví procesor po dobu dvoch hodinových cyklov na prenos jedného dátového slova, čo v prípade prenusu riadeného prerušením znamená úsporu minimálne 24 hodinových cyklov na každé prenesené dátove slovo.

Pretože metóda polling vyťažuje mikroprocesor počas prenosu dát na 100%, nebudem sa týmto prípadom ďalej zaoberať. Množstvo času t, ktoré mikroprocesor strávi riadením prenosu je možné percentuálne vyjadriť podľa nasledujúceho vzorca:

$$t = \frac{\text{baudrate} * t_{\text{transfer}}}{\text{BUSCLK}} * 100\% \tag{6.1}$$

kde baudrate je prenosová rýchlosť komunikačného rozhrania v bytoch za sekundu a t_{transfer} je doba prenosu jedného bytu medzi periférnym zariadením a operačnou pamäťou vyjadrená v počte períód frekvencie BUSCLK.

6.1.1 Čas potrebný k obsluhe rozhrania IIC

Maximálna prenosová rýchlosť rozhrania IIC je pre mikrokontroléry HCS08 stanovená na 100kb/s. Na základe vzorca 6.1 pri štandardnej hodnote BUSCLK = 8MHz strávi procesor 4,06% času obsluhou prenosu riadeného prerušením (pre minimálnu obslužnú rutinu), zatiaľ čo prenos dát s priamym prístupom do pamäte zaberie procesoru iba 0,31% času.

6.1.2 Čas potrebný k obsluhe rozhrania SPI

Maximálna rýchlosť vzorkovania dát rozhrania SPI je pre režim master stanovená na polovičnú hodnotu BUSCLK, avšak maximálne do rýchlosti 5–6MHz. Pre frekvenciu systémovej zbernice BUSCLK = 8MHz je možné dosiahnuť teoretickej prenosovej rýchlosti až 500kB/s. Podľa vzorca 6.1 tak procesor strávi až 162% času obsluhou prenosu riadeného prerušením, čo znamená, že procesor nie je schopný prenos dát pri takejto prenosovej rýchlosti obsluhovať resp. nie je schopný zabezpečiť kontinuálny prenos dát.

V rovnakej situácii vyjde prenos riadený prerušením oveľa lepšie, pretože procesor strávi prenosom dát 12,5% času, a možno tak zabezpečiť kontinuálny prenos dát a zároveň umožniť procesoru realizovať efektívny výpočet. Pre obojsmerný prenos dát je čas potrebný na obsluhu prenosu dvojnásobný.

6.2 Testovacia aplikácia

K testovaniu práce som použil niekoľko softwarových i hardwarových nástrojov. Prvotné testovanie prebiehalo v simulácii za pomoci aplikácie ModelSim od firmy Mentor Graphics. K implementácii programov pre mikrokontrolér som použil vývojové prostredie CodeWarrior od firmy Freescale. Na záver som prácu otestoval v obvode FPGA na platforme FITkit.

6.2.1 Freescale CodeWarrior

Freescale CodeWarrior je vývojové prostredie určené k vývoju programov pre rodiny mikrokontrolérov firmy Freescale Semiconductor. Programy je možné vytvárať v jazyku C alebo v assembleri.

Pre jednoduchší vývoj programov k mojej práci som vytvoril hlavičkový súbor `MC9S08SG16_EXT.h` a k nemu prislúchajúci súbor `MC9S08SG16_EXT.c`, ktorý obsahuje makrá pre prístup k registrom radiča DMA a k registrom radičov periférnych zariadení. Tento hlavičkový súbor je možné použiť ako doplnok k hlavičkovému súboru `MC9S08SG16.h` určeného pre programovanie mikrokontrolérov MC9S08SG16.

Výstupom po preložení programu v prostredí CodeWarrior je súbor vo formáte `S19`, ktorý obsahuje mapovanie binárnych dát do pamäťového priestoru. Pre jednoduché nahrávanie programu do pamätí BlockRAM som vytvoril aplikáciu s názvom `s19tovhdl`, ktorá prevedie súbor vo formáte S19 do súbora vo formáte VHDL. Výstupný súbor obsahuje inštancie pamätí BlockRAM inicializovaných cez generické premenné INIT_00 – INIT_3F. Za pomoci skriptu s názvom `vpp` je potom možné tieto inštancie vložiť priamo do súbora `memory.vhd` na patričné miesto.

Za pomoci prostredia CodeWarrior chcem overiť a dokázať, že implementovaný mikrokontrolér je kompatibilný s mikrokontrolérom MC9S08SG16.

6.2.2 ModelSim

Aplikácia ModelSim umožňuje testovanie obvodov napísaných v jazyku VHDL za pomoci simulácie. Výstupom simulácie je priebeh jednotlivých signálov v čase, vďaka čomu je možné optickým spôsobom overiť správnosť aplikácie.

K overeniu činnosti som vytvoril dve inštancie môjho mikrokontroléra a vzájomne posielal dáta medzi nimi prostredníctom oboch rozhraní IIC aj SPI. Prvý mikrokontrolér som naprogramoval tak, aby realizoval prenos dát prostredníctom DMA a druhý kontrolér s využitím prerušenia. Vytvorená aplikácia a zapojenie je do značnej miery podobné aplikácii popísanej v nasledujúcej kapitole s číslom 6.2.3. Prenos dát sa mi podarilo úspešne uskutočniť.

6.2.3 FITkit

K testovaniu práce som sa rozhodol použiť platformu FITkit 2.0[10], ktorá je dostupná na Fakulte informačných technológií Vysokého učenia technického v Brne.

FITkit je samostatný hardware, ktorý obsahuje výkonný mikrokontrolér MSP430 od firmy Texas Instruments, hradlové pole FPGA (angl. Field Programmable Gate Array) a radu periférnych zariadení. FITkit verzie 2.0 obsahuje FPGA Spartan 3 od firmy Xilinx, konkrétne model XC3S400-4PQ208C. Časť pinov FPGA je možné priamo mapovať na vstupno-výstupný port kitu, čo som použil k mapovaniu vodičov periférnych zariadení IIC a SPI.

K programovaniu FITkitu som použil aplikáciu QDevKit, ktorá je dostupná na webových stránkach FITkitu. Do tejto aplikácie som vytvoril projekt, ktorý umožňuje vypisovať obsah operačnej pamäte z môjho mikrokontroléra HCS08 vysyntetizovanom v FPGA. Takýmto spôsobom je možné skontrolovať, či prenos dát prebehol úspešne.

K FITkitu s vysyntetizovaným projektom som pripojil mikrokontrolér MC9S08JM60[1] osadený v 64-pinovom púzdre, ktorý takiež obsahuje periférne rozhrania IIC a SPI. K demonštrácii činnosti som vytvoril dva programy v prostredí CodeWarrior:

1. Pre môj mikrokontrolér som vytvoril program, ktorý prenáša 16B bloky dát cez obe rozhrania, SPI i IIC, súčasne. Dáta, ktoré sa majú odoslať cez rozhranie SPI sú uložené na adrese 0x0100 v operačnej pamäti. Podobne dáta, ktoré sa majú odoslať cez rozhranie IIC sú uložené na adrese 0x0110. Všetky prenosy dát sú realizované prostredníctvom priameho prístupu do pamäte.

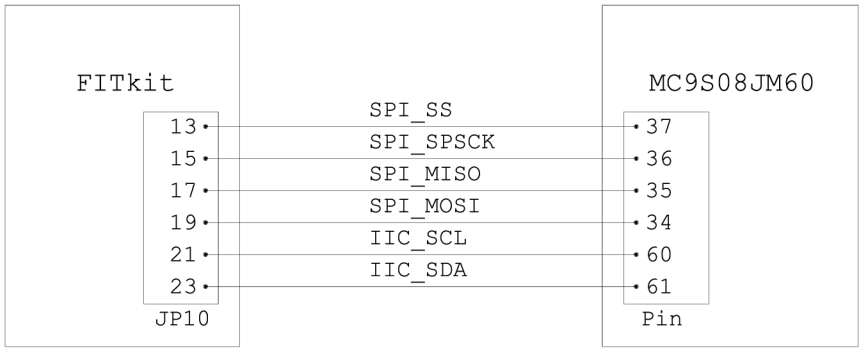
   Rozhranie SPI je plne duplexné, zatiaľ čo rozhranie IIC je polovične duplexné. Tomu som prispôsobil aj demonštračnú aplikácia a jednotlivé rozhrania som naprogramoval nasledovne:

   (a) Rozhranie SPI najskôr operuje v režime master. Súčasne s vysielaním dát rozhranie SPI dáta prijíma a ukladá na adresu 0x0200. Po odoslaní 16B bloku dát sa rozhranie prepne do režimu slave a príjme 16B blok dát. Hodnoty týchto prijatých dát musia korešpondovať s hodnotami odoslanými v predchádzajúcom bloku, ktoré sú inkrementované o 1. Tento prijatý blok dát sa uloží na adresu 0x0210.

   (b) Rozhranie IIC najskôr operuje v režime master ako vysielač. Po odoslaní celého bloku dát sa prepne do režimu prijímača a prijaté dáta uloží na adresu 0x0230. Hodnoty týchto prijatých dát musia korešpondovať s hodnotami odoslanými v predchádzajúcom bloku, ktoré sú inkrementované o 1.

2. Mikrokontrolér MC9S08JM60 tvorí druhú stranu komunikačného rozhrania. Vždy keď mikrokontrolér vysyntetizovaný vo FITkite vysiela dáta, mikrokontrolér MC9S08JM60 figuruje ako prijímač. Mikrokontrolér MC9S08JM60 prijaté dáta inkrementuje o 1 a následne ukladá do pamäte. Po prenesení bloku dát o veľkosti 16 bytov sa smer prenosu obráti a MC9S08JM60 funguje ako vysielač. Naspäť odosiela inkrementované dáta obdržané v predošlom dátovom bloku. Všetky prenosy v mikrokontroléri MC9S08JM60 sú realizované prostredníctvom prerušení.

Prepojenie oboch mikrokontrolérov znázorňuje obrázok6.1. Porty môjho mikrokontroléra som namapoval na konektor FITkitu s označením JP10. Pre signály SCL a SDL rozhrania IIC som na strane FITkitu povolil interné pull–up rezistory aby mohla komunikácia prebiehať správne.



Obrázok 6.1: Prepojenie môjho mikrokontroléra vysyntetizovaného v FPGA FITkitu a mikrokontroléra MC9S08JM60.

Po spustení oboch mikrokontrolérov je možné na strane FITkitu z prostredia QDevKit vypísať obsah pamäte RAM mikrokontroléra príkazom `BRAM READ`. K tomuto účelu som pre pamäť RAM použil dvojportovú pamäť BlockRAM, kde port A slúži pre operácie mikrokontroléra a port B pre výpis obsahu tejto pamäte do terminálu prostredia QDevKit. K výpisu som použil dostupné prostriedky pre programovanie FITkitu, konkrétne modul SPI_ctrl a SPI_adc pre komunikáciu medzi mikrokontrolérom MSP430 osadenom na FITkite a FPGA čipom.

Na základe výpisu z operačnej pamäte RAM môjho mikrokontroléra som overil, že moja realizácia rozšíreného mikrokontroléra o priamy prístup do pamäte funguje správne.

Výsledky syntézy

Maximálna frekvencia obvodu je bohužiaľ nižšia než pracovná frekvencia mikrokontrolérov HCS08. Tejto skutočnosti bolo treba nutné prispôsobiť aj prenosové rýchlosti komunikačných rozhraní pri testovaní. Maximálna možná frekvencia, na ktorej môže obvod bežať je 6,155MHz. Mikrokontrolér som pripojil na hodinový signál SMCLK = 7,3728 MHz, ktorého frekvenciu som vydelil dvomi. Obvod mikrokontroléru tak počas testovania operoval na frekvencii 3,6864 MHz.

Výpis syntezátora, informujúci o využití plochy na čipe FPGA:

```
Number of Slices:                     3579  out of   3584    99%
Number of Slice Flip Flops:            932  out of   7168    13%
Number of 4 input LUTs:               6844  out of   7168    95%
Number of IOs:                         124
Number of bonded IOBs:                 116  out of    141    82%
Number of BRAMs:                         9  out of     16    56%
Number of GCLKs:                         1  out of      8    12%
Number of DCMs:                          1  out of      4    25%
```

Ako možno vidieť, moja aplikácia má veľmi vysoké využitie konfigurovateľných logických blokov CLB (slice–ov) a look–up tabuliek (LUT). Ich využitie blížiace sa k 100% znamená, že plocha na čipe je takmer celá využitá a žiaden väčší obvod by sa do FPGA už nevošiel.

Kapitola 7

Záver a zhodnotenie

V rámci tejto práce som sa oboznámil s architektúrou rodiny mikrokontrolérov HCS08 a s princípmi technológie priameho prístupu do pamäte (DMA). Na základe nadobudnutých poznatkov som navrhol a v jazyku VHDL implementoval radič DMA, ktorý rozšíril architektúru HCS08. Navrhnutý radič je inšpirovaný radičom Intel 8237, používaným v osobných počítačoch v zberniciach ISA. Súčasťou práce bola aj implementácia sériových komunikačných rozhraní SPI a IIC, ktoré bolo nutné rozšíriť o podporu realizácie autonómnych DMA prenosov.

Všetky rozšírenia som navhrhol a implementoval s ohľadom na zachovanie kompatibility s architektúrou HCS08, konkrétne modelu s označením MC9S08SG16. Konfigurácia radiča DMA prebieha rovnakým spôsobom ako konfigurácia akéhokoľvek iného periférneho zariadenia v architektúre. DMA prenos je naviac možné úplne vypnúť a prenos dát realizovať spôsobom kompatibilným s modelom MC9S08SG16.

V rámci mojej práce sa mi podarilo úspešne otestovať implementovaný mikrokontrolér v obvode FPGA vývojového kitu FITkit. Vysyntetizovaný mikrokontrolér dokázal úspešne komunikovať s okolím. Problémom však je nízka pracovná frekvencia a pomerne veľká plocha, ktorú implementácia zaberá na čipe FPGA. Táto veľkosť aplikácie je jedným z dôvodov nízkej pracovnej frekvencie.

7.1 Možné rozšírenia a úpravy

Prácu je možné v budúcnosti rozšíriť o asynchrónne komunikačné rozhranie SCI. Ďalšími možnými rozšíreniami sú podpora multi–master módu pre rozhranie IIC, ktorý som v práci neimplementoval a taktiež podpora programovateľných signálov rozhrania SPI.

Medzi možné úpravy možno zaradiť optimalizáciu celkového návrhu mikrokontroléra a jeho periférii za účelom zvýšenia pracovnej frekvencie a zníženia potrebnej plochy na čipe FPGA.

Literatúra

[1] Freescale Semiconductor, Inc.: MC9S08JM60 Data Sheet. Technická zpráva, Freescale Semiconductor, Inc., 2009.
URL <http://www.freescale.com/files/microcontrollers/doc/data_sheet/MC9S08JM60.pdf>

[2] Freescale Semiconductor, Inc.: MC9S08SG32 Data Sheet. Technická zpráva, Freescale Semiconductor, Inc., 2012.
URL <http://www.freescale.com/files/microcontrollers/doc/data_sheet/MC9S08SG32.pdf>

[3] Heath, S.: *Embedded Systems Design*. Newnes, druhé vydání, 2002, ISBN 07-50655-46-1.

[4] Intel Corporation: 8237/8237-2 High performance programmable DMA controller. Technická zpráva, Intel Corporation.
URL <http://zet.aluzina.org/images/8/8c/Intel-8237-dma.pdf>

[5] Kollár, D.: Základy technického a programového vybavenia [online]. http://www.dnp.fmph.uniba.sk/~kollar/pc_hw_sw/pc8.htm, 10. 9. 2008 [cit. 2013-01-03].

[6] NXP Semiconductors: *I2C-bus specification and user manual*. 2012.
URL <http://www.nxp.com/documents/user_manual/UM10204.pdf>

[7] Organization, C.: Programmed I/O, Interrupt & Direct Memory Access (DMA). http://www.louiewong.com/archives/137, 2. 10. 2009.

[8] Osborne, A.: *An Introduction to Microcomputers: Volume 1: Basic Concepts*. Osborne McGraw Hill., 1980, ISBN 09-31988-34-9.

[9] Pereira, F.: *HCS08 Unleashed: Designer's Guide To the HCS08 Microcontrollers*. BookSurge Publishing, 2008, ISBN 14-19685-92-9.

[10] Vašíček, Z.: FITkit. http://merlin.fit.vutbr.cz/FITkit/, 2012.

Príloha A

Obsah CD

Priložené CD obsahuje na najvyššej úrovni 3 zložky.

`report` – Adresár obsahuje zdrojové texty tejto technickej správy, ktoré je možné preložiť príkazom `make`. Ďalej obsahuje výslednú verziu dokumentácie vo formáte `pdf` nazvanú `projekt.pdf`. Nachádzajú sa tu i projektové súbory obrázkov vo formáte `svg`, ktoré boli vytvorené v programe `Inkscape`.

`ModelSim` – Táto zložka obsahuje zdrojové súbory implementovaného mikrokontroléra spolu s testovacími súbormi pre beh v prostredí ModelSim.

`FITkit` – Táto zložka obsahuje zdrojové súbory implementovaného mikrokontroléra spolu s projektom vytvoreného pre aplikáciu QDevKit. Podzložku `HCS08-DMA` je možné pridať priamo do adresára projektov aplikácie.

Príloha B

Tabuľka vektorov prerušenia

Nasledujúca tabuľka zobrazuje zoznam implementovaných prerušovacích vektorov a k nim príslušnú adresu. Neimplementované vektory sú označené znakom pomlčky (-).

Adresa	Vektor	Názov vektora
0xFFC0:0xFFC1	-	-
0xFFC2:0xFFC3	-	-
0xFFC4:0xFFC5	-	-
0xFFC6:0xFFC7	-	-
0xFFC8:0xFFC9	-	-
0xFFCA:0xFFCB	-	-
0xFFCC:0xFFCD	-	-
0xFFCE:0xFFCF	IIC	Viic
0xFFD0:0xFFD1	-	-
0xFFD2:0xFFD3	-	-
0xFFD4:0xFFD5	-	-
0xFFD6:0xFFD7	-	-
0xFFD8:0xFFD9	-	-
0xFFDA:0xFFDB	-	-
0xFFDC:0xFFDD	-	-
0xFFDE:0xFFDF	-	-
0xFFE0:0xFFE1	SPI	Vspi
0xFFE2:0xFFE3	-	-
0xFFE4:0xFFE5	-	-
0xFFE6:0xFFE7	-	-
0xFFE8:0xFFE9	-	-
0xFFEA:0xFFEB	DMA kanál 3	Vdma3
0xFFEC:0xFFED	DMA kanál 2	Vdma2
0xFFEE:0xFFEF	DMA kanál 1	Vdma1
0xFFF0:0xFFF1	DMA kanál 0	Vdma0
0xFFF2:0xFFF3	-	-
0xFFF4:0xFFF5	-	-

Tabuľka B.1: Implementované vektory prerušenia (Strana 1/2)

Adresa	Vektor	Názov vektora
0xFFF6:0xFFF7	-	-
0xFFF8:0xFFF9	-	-
0xFFFA:0xFFFB	-	-
0xFFFC:0xFFFD	SWI	Vswi
0xFFFE:0xFFFF	Reset	Vreset

Tabuľka B.2: Implementované vektory prerušenia (Strana 2/2)

Príloha C

Zoznam registrov

Nasledujúce tabuľky vyobrazujú zoznam registrov v mojej architektúre. Neimplenentované registre sú označené znakom pomlčky (-). Prístup k týmto registrom je presmerovaný do operačnej pamäte. Niektoré položky z registrov sú defaultne nevyužité alebo nie sú podporované. Takéte položky sú označené znakom 0.

Adresa	**Register**	**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**Bit 0**
0x0000–0x002F	-	-	-	-	-	-	-	-	-
0x0030	SPIDMAC	DMAEN	0	0	0	0	0	0	0
0x0031	SPIDMARC	Bit 7	6	5	4	3	2	1	Bit 0
0x0032	SPIDMAWC	Bit 7	6	5	4	3	2	1	Bit 0
0x0033	-	-	-	-	-	-	-	-	-
0x0034	IICDMAC	DMAEN	ADEXT	0	0	0	DAD10	DAD9	DAD8
0x0035	IICDMAA	DAD7	DAD6	DAD5	DAD4	DAD3	DAD2	DAD1	DAD0
0x0036	IICDMARC	Bit 7	6	5	4	3	2	1	Bit 0
0x0037	IICDMAWC	Bit 7	6	5	4	3	2	1	Bit 0
0x0038–0x003F	-	-	-	-	-	-	-	-	-
0x0040	DMAIE	0	0	0	0	DIE3	DIE2	DIE1	DIE0
0x0041	DMAIC	WI3	RI3	WI2	RI2	WI1	RI1	WI0	RI0
0x0042	DMADIR	WD3	WD2	WD1	WD0	RD3	RD2	RD1	RD0
0x0043	DMAS1	WTCF3	WTCF2	WTCF1	WTCF0	RTCF3	RTCF2	RTCF1	RTCF0
0x0044	DMAS2	0	0	0	0	DIF3	DIF2	DIF1	DIF0
0x0045–0x004F	-	-	-	-	-	-	-	-	-
0x0050	SPIC1	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	-	LSBFE
0x0051	SPIC2	0	0	0	0	0	0	0	0
0x0052	SPIBR	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
0x0053	SPIS	SPRF	0	SPTEF	0	0	0	0	0
0x0054	-	-	-	-	-	-	-	-	-
0x0055	SPID	Bit 7	6	5	4	3	2	1	Bit 0

Tabuľka C.1: Zoznam registrov stránky s priamym prístupom (Strana 1/2)

Adresa	Register	Bit 7	6	5	4	3	2	1	Bit 0
0x0056– 0x0057	-	–	–	–	–	–	–	–	–
0x0058	IICA	AD7	AD6	AD5	AD4	AD3	AD2	AD1	0
0x0059	IICF	MULT		ICR					
0x005A	IICC1	IICEN	IICIE	MST	TX	TXAK	RSTA	0	0
0x005B	IICS	TCF	IAAS	BUSY	ARBL	0	SRW	IICIF	RXAK
0x005C	IICD	Bit 7	6	5	4	3	2	1	Bit 0
0x005D	IICC2	GCAEN	ADEXT	0	0	0	AD10	AD9	AD8
0x005E– 0x006F	-	–	–	–	–	–	–	–	–
0x0070	DMARAH0	Bit 15	14	13	12	11	10	9	Bit 8
0x0071	DMARAL0	Bit 7	6	5	4	3	2	1	Bit 0
0x0072	DMAWAH0	Bit 15	14	13	12	11	10	9	Bit 8
0x0073	DMAWAL0	Bit 7	6	5	4	3	2	1	Bit 0
0x0074	DMARAH1	Bit 15	14	13	12	11	10	9	Bit 8
0x0075	DMARAL1	Bit 7	6	5	4	3	2	1	Bit 0
0x0076	DMAWAH1	Bit 15	14	13	12	11	10	9	Bit 8
0x0077	DMAWAL1	Bit 7	6	5	4	3	2	1	Bit 0
0x0078	DMARAH2	Bit 15	14	13	12	11	10	9	Bit 8
0x0079	DMARAL2	Bit 7	6	5	4	3	2	1	Bit 0
0x007A	DMAWAH2	Bit 15	14	13	12	11	10	9	Bit 8
0x007B	DMAWAL2	Bit 7	6	5	4	3	2	1	Bit 0
0x007C	DMARAH3	Bit 15	14	13	12	11	10	9	Bit 8
0x007D	DMARAL3	Bit 7	6	5	4	3	2	1	Bit 0
0x007E	DMAWAH3	Bit 15	14	13	12	11	10	9	Bit 8
0x007F	DMAWAL3	Bit 7	6	5	4	3	2	1	Bit 0

Tabuľka C.2: Zoznam registrov stránky s priamym prístupom (Strana 2/2)

Adresa	Register	Bit 7	6	5	4	3	2	1	Bit 0
0x1800– 0x185F	-	–	–	–	–	–	–	–	–

Tabuľka C.3: Zoznam registrov stránky s nepriamym prístupom