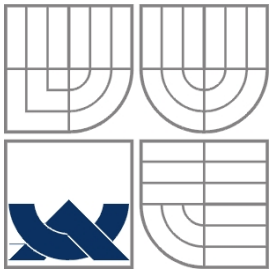
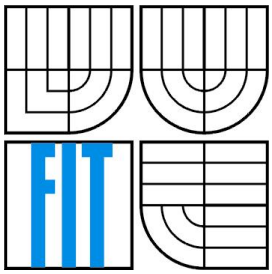


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

MODUL ROZHRANÍ ETHERNET PRO PLATFORMU FITKIT

ETHERNET INTERFACE MODULE ON FITKIT PLATFORM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

BC. JIŘÍ ŠOUSTAR

VEDOUCÍ PRÁCE
SUPERVISOR

ING. VÁCLAV ŠIMEK

BRNO 2008

Abstrakt

Práce se zabývá problematikou vestavěných systému, síťové komunikace a podpory síťové komunikace ve vestavěných systémech. Dále představuje výukovou platformu FITkit a nabízí koncepci rozšíření této platformy o síťové rozhraní založené na síťovém standardu Ethernet. Na základě této koncepce práce podrobně popisuje vhodné řešení realizující síťové rozhraní pro platformu FITkit formou rozšiřujícího modulu, který má za úkol zajistit dostatečnou podporu při vývoji síťových aplikací na této platformě.

Klíčová slova

Vestavěný systém, počítačová síť, síťová komunikace, síťový protokol, síťové rozhraní, referenční model ISO/OSI, model TCP/IP, Ethernet, mikrokontrolér, platforma FITkit.

Abstract

This graduation thesis is aimed at design and implementation of embedded systems, network communication and support of the network communication for embedded systems. Furthermore it introduces school platform FITkit and offers a conceptional extension of the network interface for this platform based on Ethernet network standard. Based on that conception I'm trying to find and describe suitable solution for FITkit platform which realizes a network interface as a form of extension module capable of development supporting.

Keywords

Embedded system, computer network, network communication, network protocol, network interface, reference model ISO/OSI, model TCP/IP, Ethernet, microcontroller, platform FITkit.

Citace

Šoustar Jiří: Modul rozhraní Ethernet pro platformu FITkit, diplomová práce, Brno, FIT VUT v Brně, 2008.

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením ing. Václava Šimka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jiří Šoustar
10.5.2008

Poděkování

Rád bych touto cestou poděkoval především svému vedoucímu ing. Václavu Šimkovi za ochotu a odborné rady, které mi poskytoval. Dále děkuji firmě Duhasys s.r.o. za pomoc při realizaci tohoto projektu a zvláště pak za možnost spolupráce a tak získání cenných zkušeností v oblasti vestavěných systémů.

Obsah

1 Úvod

Cílem této práce je navrhnout a realizovat koncepci síťového rozhraní pro platformu FITkit. Tato koncepce by měla být realizována formou rozšiřujícího modulu, který zajistí přístup do sítě předem specifikovaného typu a umožní tak implementovat na platformě FITkit síťové aplikace. Nastává zásadní otázka: „Jakou síťovou technologii zajišťující síťovou komunikaci zvolit?“ Jelikož je v oblasti lokálních počítačových sítí značně rozšířena technologie Ethernet, bude jistě ideálním řešením navrhnout a zkonstruovat síťové rozhraní pro platformu FITkit založenou právě na této technologii. Rozšiřující modul se tedy bude nazývat modul rozhraní Ethernet či zkráceně EIM (Ethernet Interface Module). Dále uvádím stručný obsah jednotlivých kapitol.

Druhá kapitola se snaží přiblížit čtenáři pojem „Vestavěný systém“. Z počátku se zaměřuje na definici vestavěného systému. Po té přichází jemný úvod zabývající se možnostmi a samotným cyklem návrhu vestavěného systému. Na návrh navazuje neméně důležité téma architektury vestavěného systému, kde čtenář získá informace o tom co vlastně architektura vestavěného systému je, jak je důležitá její přítomnost při vývoji a jaké pozitiva přináší.

Třetí kapitola se věnuje tématu počítačových sítí. Dále popisuje výhody počítačových sítí v oblasti vestavěných systému, důležité vlastnosti jako jsou: vzdálenost mezi připojenými zařízeními, typ přenosového média a architektura sítě. Nakonec se snaží objasnit myšlenku referenčního síťového modelu OSI, na jehož principech je postavena většina dnešních počítačových sítí.

Čtvrtá kapitola popisuje podporu sítě a síťové komunikace ve vestavěném systému. Nejprve představuje síťový model TCP/IP, na němž jsou často sítě ve vestavěných systémech stavěny. Pak postupně charakterizuje jednotlivé vrstvy tohoto modelu a uvádí síťové protokoly, které se ve vestavěném systému častokrát implementují.

Pátá kapitola se snaží objasnit pojem platforma FITkit. Charakterizuje důležité vlastnosti této platformy jako je její účel, koncepce, struktura a přínos. Dále uvádí způsob návrhu a implementace aplikací na platformě FITkit, přičemž se pozastavuje u podpory tvorby síťových aplikací.

Šestá kapitola se zaměřuje na koncepci, operační módy a přístupové schéma modulu rozhraní Ethernet pro platformu FITkit. V prvé řadě popisuje koncepci rozšiřujícího modulu platformy FITkit, ve které se snaží nastínit hrubé řešení dané problematiky. Dále popisuje operační módy, ve kterých modul pracuje a jejich specifické vlastnosti související s funkcí modulu. Nakonec detailně charakterizuje přístupové schéma modulu, ve kterém analyzuje možné varianty přístupu k modulu z FITkitu a snaží se o výběr optimálního řešení.

Sedmá kapitola se zaměřuje na hardwarové řešení modulu rozhraní Ethernet. Celý problém hardwarového řešení rozděluje na menší podproblémy, které představují funkční bloky provádějící svoji specifickou úlohu. Mezi tyto bloky patří procesorová jednotka, fyzická vrstva Ethernetu, datové úložiště, definice vstupů/výstupů a napájení. Po rozboru a popisu těchto bloků následuje

charakteristika struktury zapojení, která dává souhrnný přehled o propojení jednotlivých funkčních bloků. Kapitola je zakončena pohledem na samotnou realizaci modulu se zaměřením na tvorbu schéma zapojení a tvorbu desky plošného spoje.

Osmá kapitola se zaměřuje na popis softwarového řešení modulu rozhraní Ethernet. Nejprve podává obecnou představu dané problematiky a rozděluje ji na menší podproblémy realizované formou modulů založených na vrstveném modelu. Dále definuje obecná rozhraní pro komunikační a síťová zařízení, na kterých je postavena podstatná část softwarového řešení. Po této definici popisuje konkrétní moduly komunikačních zařízení, síťových zařízení a síťových rozhraní. Předposlední část této kapitoly se zaměřuje na funkční moduly, které zastupují zásadní postavení v rámci řešení dané problematiky. V poslední části kapitoly je popsán způsob implementace softwarového řešení.

Poslední kapitola popisuje tvorbu demonstrační aplikace na platformě FITkit využívající ke své funkci modul rozhraní Ethernet. Kapitola začíná výběrem vhodné aplikační domény, která by výstižně demonstrovala funkčnost modulu rozhraní Ethernet. Pak následuje část zabývající se analýzou a návrhem řešení demonstrační aplikace, ve které je provedena detailní analýza a návrh nových síťových modulů spadající do vybrané aplikační domény. Po té je popsána samotná realizace a implementace demonstrační aplikace na platformě FITkit. Nakonec je věnována pozornost systému konfigurace, který je reprezentován sadou definovaných konfiguračních příkazů.

Na závěr této úvodní kapitoly je nutné uvést, že tato práce v následujících čtyřech kapitolách čerpá ze semestrální práce spadající pod předmět semestrální projekt, který probíhal v zimním semestru.

2 Vestavěný systém

Tato kapitola se snaží přiblížit čtenáři široce zastoupený pojem „Vestavěný systém“. Z počátku se zaměřuje na definici vestavěného systému. Po té přichází jemný úvod zabývající se možnostmi a samotným cyklem návrhu vestavěného systému. Na návrh navazuje neméně důležité téma architektury vestavěného systému, kde čtenář získá informace o tom co vlastně architektura vestavěného systému je, jak je důležitá její přítomnost při vývoji a jaké pozitiva přináší. A nakonec je představen vrstvený model vestavěného systému.

2.1 Definice vestavěného systému

Vestavěný systém je aplikovatelný výpočetní systém, který se odlišuje od klasických výpočetních systémů takových jako jsou osobní počítače nebo superpočítače [1].

Přesná definice pro pojem vestavěný systém se hledá stěží, ale lze o vestavěných systémech tvrdit, že se stále rozvíjí s pokroky v technologiích a cena takového systému dramaticky klesá. Specifické vlastnosti charakterizující vestavěný systém jsou následující:

- *Vestavěné systémy jsou více limitovány v technickém či programovém vybavení nežli jiné výpočetní systémy.* Limity v technickém vybavení mohou být: výpočetní výkonnost, spotřeba elektrické energie, velikost paměti, podpora dedikovaných rozhraní a další. V programovém vybavení mohou být limity oproti jiným výpočetním systémům tyto: celkově méně aplikací, žádný nebo limitovaný operační systém a tudíž nedostatečná abstrakce přímo na úrovni aplikačního rozhraní.
- *Vestavěný systém je ve většině případů navrhnut k vykonávání dedikované funkce.* Mnoho vestavěných zařízení je primárně navrhnuto pro jednu specifickou funkci. Toto tvrzení musíme brát s rezervou. Například PDA (Personal Data Assistant) je vestavěné zařízení a přesto je schopné vykonávat širokou škálu primárních funkcí.
- *Vestavěný systém je výpočetní systém s vyššími požadavky na kvalitu a spolehlivost než jiné typy výpočetních systémů.* Některé rodiny vestavěných zařízení mají dokonce velmi vysoké požadavky na kvalitu a spolehlivost. Takové zařízení jsou také někdy nazývány jako ”Hard RT embedded systems“. Jako příklad si můžeme uvést řídící jednotku ABS v automobilu.

Elektronická zařízení jsou prakticky v každém obchodním segmentu klasifikovány jako vestavěné systémy, ale existují i názory, které se navzájem vylučují. Například PDA (Personal Data Assistant) je pro jednu skupinu vestavěným systémem, zatímco druhá skupina jej vnímá jako

výpočetní systém obdobný přenositelnému osobnímu počítači. Z toho vyplývá, že typ výpočetního systému lze určit subjektivně až v případě konkrétního systému. Tento výrok jen potvrzuje tvrzení, že se velmi špatně hledá jediná přesná definice toho co je vestavěný systém.

Tabulka 1 : Příklady vestavěných zařízení

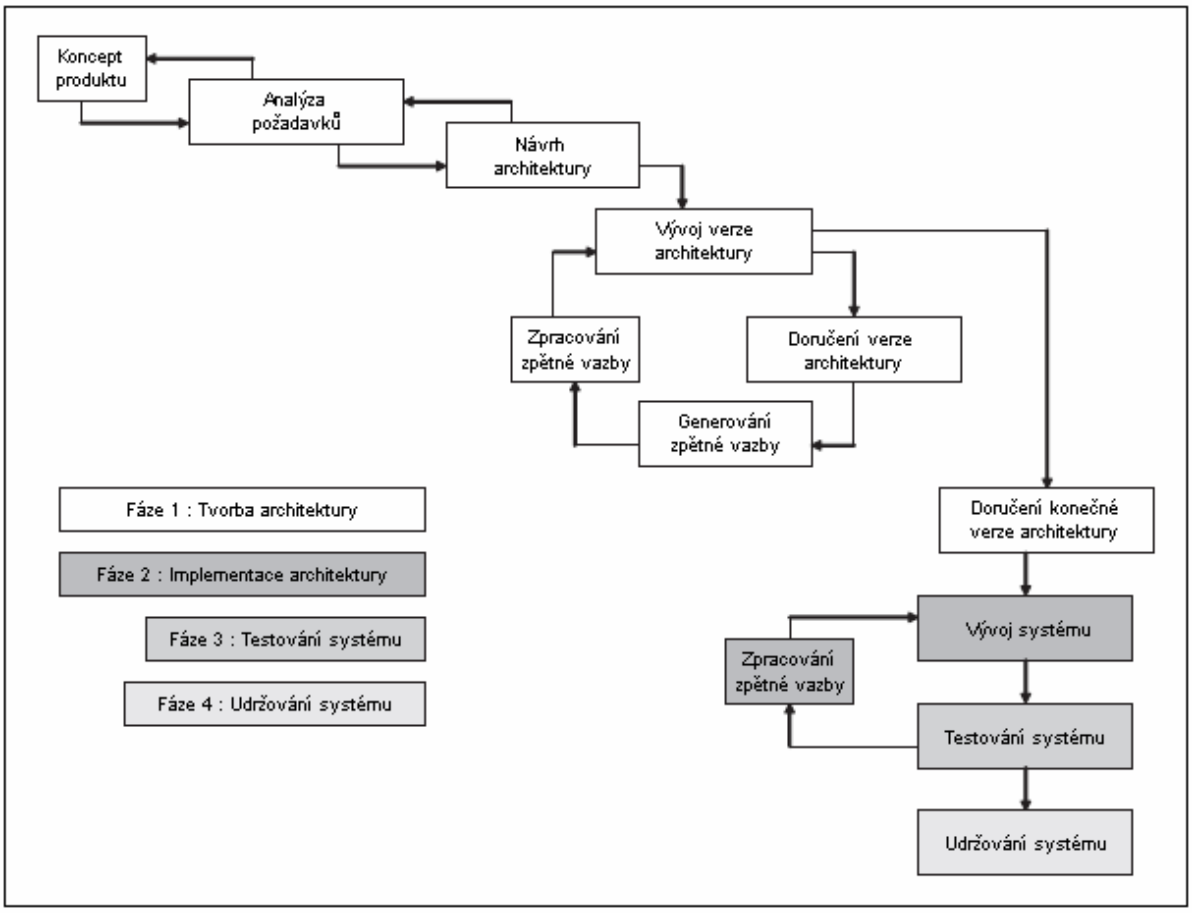
Oblast	**Vestavěné zařízení**
Automobilový průmysl	Systém zapalování Systém ovládaní motoru Brzdný systém
Spotřební elektronika	Digitální, Analogové televize Set-Top Box PDA Kuchyňské spotřebiče Hračky, Hry Telefony Kamery GPS
Průmyslové ovládaní	Robotické a kontrolní systémy
Zdravotnictví	Infuzní pumpa Srdeční monitoring Dialyzační zařízení
Počítačové sítě	Směrovače Přepínače Rozbočovače
Kancelář	Fax Foto kopírky Tiskárny Monitory Skenery

2.2 Návrh vestavěného systému

Pokud si přiblížíme návrh vestavěných systému z pohledu návrháře, tak existuje několik modelů, které mohou být aplikovány na popis cyklu návrhu vestavěného systému. Většina těchto modelů je založena na jednom nebo kombinaci následujících modelů vývoje vestavěných systémů [1]:

- *Big-bang* model, ve kterém v podstatě neexistuje žádný sofistikovaný proces rozdělení vývoje na menší podproblémy, jak je tomu zvykem ve většině ostatních modelů.
- *Code-and-fix* model, ve kterém jsou pevně definovány požadavky na cílový produkt, ale nejsou nijak formálně analyzovány před počátkem vývoje systému.
- *Waterfall* model, ve kterém je proces vývoje systému rozdělen do určitých kroků, kde výsledek jednoho kroku je vstupem dalšího kroku.
- *Spiral* model je velice podobny předešlému modelu, ale navíc umožňuje v průběhu vývoje systému zpětnou vazbu, díky které se může vývoj systému vrátit o několik kroků zpět.

Většina publikací zabývající se vestavěnými systémy a jejich návrhem uvádějí, že je v praxi nejrozšířenějším modelem vývoje vestavěných systémů “Development Lifecycle Model“ dále uváděn zkratkou DLM. Tento model je založen na kombinaci populárních modelů vývoje “Waterfall“ a “Spiral“. Navíc průzkum zabývající se postupem při vývoji vestavěných systémů a především efektivností takového postupu došel k závěru, že naprostá většina projektů striktně se držící modelu DLM končila s úspěchem [1]. Následující obrázek znázorňuje podrobně schéma modelu DLM.



Obrázek 1 : Ukázka modelu DLM

Z obrázku je patrné, že model DLM rozděluje cyklus vývoje systému do čtyřech fází:

- *Tvorba architektury* je proces plánování návrhu vestavěného systému.
- *Implementace architektury* je proces implementace vestavěného systému.
- *Testování systému* je proces testování vestavěného systému za účelem nalezení nedostatků či chyb a jejich odstranění.
- *Udržování systému* je proces uvedení vestavěného systému do provozu a poskytnutí veškeré technické podpory uživateli po celou dobu životnosti takového systému.

2.3 Architektura vestavěného systému

Architektura vestavěného systému je abstrakcí vestavěného zařízení. To znamená, že jde o generalizaci systému, ve které není detailně zachycena implementační informace taková jako zdrojový kód nebo elektrické schéma zařízení. Na úrovni architektury jsou ve vestavěném systému konkrétní komponenty reprezentovány kompozicí vzájemně se ovlivňujících prvků. Tedy detailní implementace prvku je zabstraktněna informací o jejím chování a vztahu vůči okolním prvkům. Prvky architektury mohou být integrovány do vestavěného zařízení nebo mohou existovat mimo vestavěné zařízení a komunikovat s integrovanými prvky.

Informace podle úrovně architektury je fyzicky reprezentována formou struktury. Struktura je jedna z možností reprezentace architektury, obsahující množinu prvků, vlastností a vztahů mezi prvky. Struktura by se dala označit jako stručná a rychlá informace o celém systému dávající představu o prostředí a prvcích, které v tomto prostředí vzájemně komunikují [1]. Většinou je velice obtížné zachytit všechny složitosti systému do jediné struktury, takže je architektura zařízení typicky tvořena více než jednou strukturou. Nejpoužívanějšími typy struktur architektury vestavěného systému je následující čtveřice:

- Modul
- Komponenta
- Konektor
- Alokátor

Používání architektury velice ulehčuje návrh vestavěných systémů, jelikož se jedná o jeden z nejsilnějších nástrojů k řešení problému, které jsou na nás kladeny při návrhu nového systému. Mezi obecné problémy, se kterými se často při návrhu setkáváme jsou:

- Definování a vyjádření návrhu celého systému.
- Konečná cena systému.
- Určení systémové integrity, takové jako spolehlivost a bezpečnost.
- Omezení daná vlastnostmi některých prvku systému (např. velikost paměti, životnost baterie, maximální spotřeba, atd.).
- Prodejnost, profit.

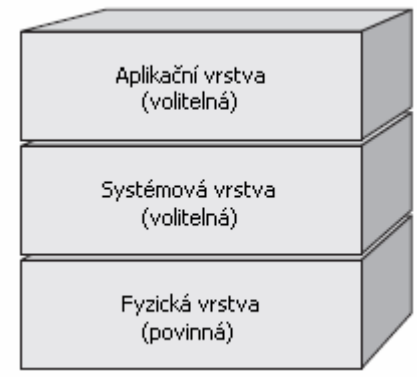
Architektura vestavěného systému může vyřešit tyto problémy již v ranném stádiu vývoje. Bez definice nebo obeznámení s detaily vnitřní implementace problému může být architektura první nástroj, který bude použit k definování infrastruktury návrhu, možných návrhových voleb nebo návrhových omezení. Co dělá architekturní návrh tak silný je schopnost informovat a urychlit

komunikaci s širokou skupinou lidí i bez znalostí technického pozadí problému. Výstupem architektury jsou jasné požadavky na systém a není tedy problémem velice brzy analyzovat a testovat kvalitu zařízení a jeho výkonnost za různých okolností. Kromě toho, pokud je architektura správně vytvořena lze odhadnout přesněji cenu v době návrhu a tu poté pomocí nahrazení či úpravy jistých prvků, které se zdají být neperspektivní redukovat. A nakonec, většina struktur dané architektury může být využita v jiném projektu s podobnou charakteristikou, čímž se nám zkrátí čas na vývoj a tím sníží celkové náklady.

2.4 Model vestavěného systému

Tento referenční model je v podstatě vrstvená (modulární) reprezentace modulárních struktur architektury vestavěného systému [1]. Model vestavěného systému se skládá z jedné povinné vrstvy a dvou volitelných vrstev.

- Fyzická vrstva (Hardware Layer) je povinná vrstva, která popisuje všechny fyzické komponenty umístěné na desce vestavěného zřízení.
- Systémová vrstva (System Software Layer) je volitelná vrstva, která popisuje logické komponenty, které programově zajišťují rozhraní mezi fyzickou a aplikační vrstvou.
- Aplikační vrstva (Application Software Layer) je volitelná vrstva, která popisuje logické komponenty, které programově zajišťují aplikační požadavky kladené na systém.



Obrázek 2 : Model vestavěného systému

3 Síťová komunikace

Tato kapitola se věnuje tématu počítačových sítí. Dále popisuje výhody počítačových sítí v oblasti vestavěných systému, důležité vlastnosti jako jsou: vzdálenost mezi připojenými zařízeními, typ přenosového média a architektura sítě. Nakonec se snaží objasnit myšlenku referenčního síťového modelu OSI, na jehož principech je postavena většina dnešních počítačových sítí.

3.1 Preference sítí

Se sítovým připojením může výpočetní systém přes lokální síťové rozhraní posílat i přijímat informace různého druhu na krátké i dlouhé vzdálenosti pomocí drátového i bezdrátového spojení. Obrovská výhoda při použití sítí je možnost komunikace různých výpočetních systémů prostřednictvím standardizovaných síťových protokolů. V sítích vestavěných systémů tak může každý systém komunikovat s jiným, sdílet s ním důležité informace a odpovídat na jeho požadavky. Osobní počítače pak mohou jednoduše vestavěné systémy monitorovat a ovládat.

3.2 Komponenty sítě

Pokud budeme vyvíjet vestavěné zařízení s podporou síťové komunikace musíme určit veškeré komponenty na úrovni logické i fyzické, ze kterých bude síť složena. K určení těchto komponent je v prvé řadě nutné obeznámit se s typem a vlastnostmi sítě, do které bude zařízení připojeno. Použitím těchto znalostí společně se síťovým modelem např. OSI můžeme síťové komponenty určit.

3.3 Důležité vlastnosti sítě

Seznámení se s sítí, do které bude zařízení připojeno je velice důležité, protože klíčové vlastnosti této sítě budou diktovat standardy potřebné při implementaci vestavěného systému [1]. Takové tři základní vlastnosti sítě jsou:

- Vzdálenost mezi připojenými zařízeními.
- Typ přenosového média.
- Architektura sítě.

3.3.1 Vzdálenost mezi připojenými zařízeními

Sítě jsou definovány buď jako LAN (Local Area Networks) nebo WAN (Wide Area Networks). LAN je síť, ve které všechna zařízení tvoří jakousi vnitřní logickou kontinuitu, jsou si navzájem blízko. Příkladem může být síť v jednom domě či pokoji. Naproti tomu WAN je síť, která spojuje zařízení nebo LAN sítě skrz širší geografickou oblast. Příkladem mohou být mezikontinentální, mezinárodní či meziměstské sítě. Existují ještě další podtypy WAN sítí: MAN (Metropolitan Area Networks) pro vnitro městské sítě nebo CAN (Campus Area Networks) pro školní sítě.

3.3.2 Typ přenosového media

V sítích jsou zařízení spojena pomocí přenosového média, které jsou buď omezena nebo neomezena. Omezená přenosová média jsou kabely a dráty. Ty jsou také označovány jako vodivá spojení, protože se informace přenáší přímo po mediu formou elektrického proudu nebo světla. Neomezená přenosová media jsou bezdrátová spojení. Ty se analogicky označují jako nevodivá spojení, protože informace se nepřenášejí vodivě daným mediem, ale skrz okolí tvořené vzduchem, vakuem nebo vodou. Přenosy po omezeném přenosovém médiu respektive neomezením přenosovém mediu nazýváme drátové respektive bezdrátové přenosy.

Klíčové vlastnosti rozlišující jednotlivá přenosová media jsou:

- Jaký typ dat může médium přenášet (analogovy, digitální).
- Kolik dat je schopno médium přenést (kapacita).
- Jak rychle může médium přenášet data od zdroje k cíli (rychlost).
- Do jaké vzdálenosti jsou data na médiu přenášena bez poškození.
- Jak moc je médium odolné vůči okolním rušivým vlivům.

Příklady drátových přenosových médií :

- UTP (Unshielded Twisted Pair).
- Koaxiální kabel.
- Optické vlákno.

Příklady bezdrátových přenosových médií :

- Pozemní vlny.
- Satelitní vlny.
- Radiové vlny.
- Infračervené vlny.

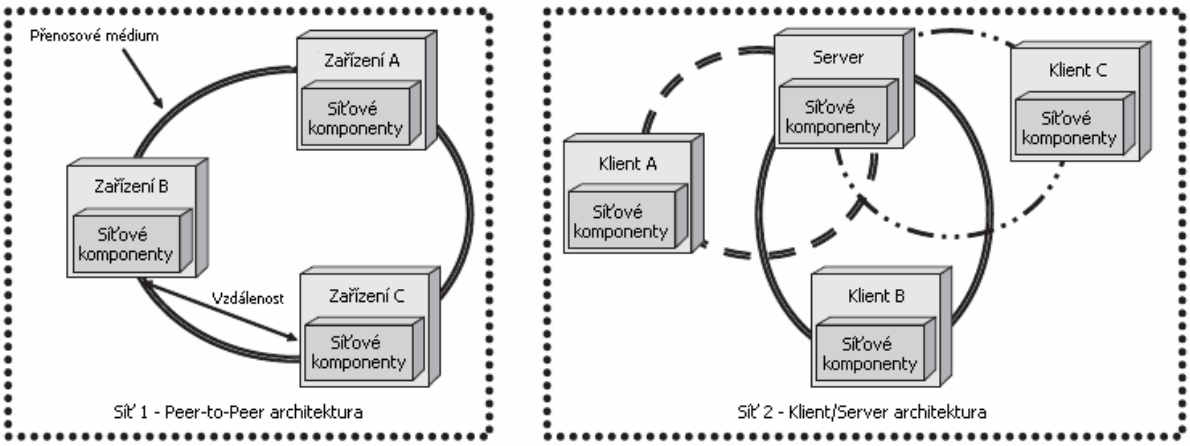
3.3.3 Architektura sítě

Vztah mezi připojenými zařízeními v síti vyjadřuje architektura sítě. Rozeznáváme tři typy síťových architektur: *Peer-to-peer*, *Klient/Server* a *hybridní* [1].

Peer-to-peer architektury jsou implementace sítí bez centrálního řídícího elementu. Každé zařízení v síti musí spravovat vlastní zdroje a požadavky. Všechna zařízení komunikují rovnocenně a vzájemně využívají své zdroje. Peer-to-peer sítě jsou obvykle implementovány jako LAN.

Klient/Server architektury jsou implementace sítí, ve kterých je centrální zařízení nazývané server. Server je zodpovědný za vyřízení většiny sítových požadavků a správu zdrojů. Další zařízení v síti se nazývají klienti. Klienti mají k dispozici menší počet zdrojů a proto musí využívat zdrojů serveru. Klient/Server architektura je více komplexní než Peer-to-peer architektura, ale má jeden kritický atribut, kterým je možnost poruchy serveru. V Klient/Server architekturách má pouze server odpovědnost za poskytnutí dalších potřebných zdrojů v případě poruchy. Nicméně je více bezpečný než Peer-to-peer model, protože pouze server má tu schopnost vidět na další zařízení. Klient/Server architektury by se také mohly označit za výkonnější, protože server je většinou dostatečně výkonné zařízení, aby mohlo poskytnout potřebné zdroje klientům. Tato architektura je implementována buď jako LAN nebo WAN.

Hybridní architektura je kombinací Peer-to-peer a Klient/Server architektury. Tato architektura je také implementována buď jako LAN nebo WAN.

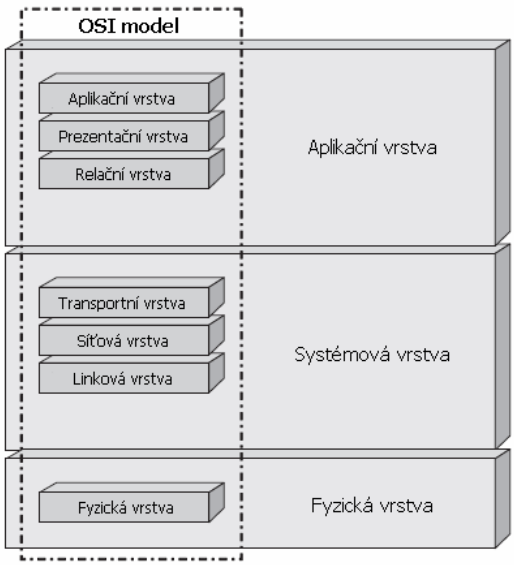


Obrázek 3 : Diagram síťových architektur

3.4 Síťový model OSI

Referenční síťový model OSI (Open System Interconnection) byl uveden začátkem roku 1980 mezinárodní organizací pro standardizaci ISO (International Organization for Standardization). OSI model člení složitou problematiku počítačových sítí na menší podproblémy reprezentované požadovanými fyzickými či programovými komponentami uvnitř síťového zařízení formou sedmi vrstev: fyzické, linkové, síťové, transportní, relační, prezentační a aplikační [1].

Vrstvy síťového modelu OSI lze samozřejmě mapovat do modelu vestavěného systému. Toto mapovaní se fyzické vrstvy nedotkne, ale linková, síťová a transportní vrstva je mapována do systémové vrstvy modelu vestavěného systému. Zbývající vrstvy se mapují do aplikační vrstvy vestavěného systému.



Obrázek 4 : Síťový model OSI integrovaný do modelu vestavěného systému

3.4.1 Fyzická vrstva

Fyzická vrstva definuje, spravuje a zpracovává datové signály přicházející skrz přenosové médium. Tato vrstva je také zodpovědná za příjem dat z vyšších vrstev vestavěného systému, které pak odesílá skrz přenosové médium. Stejně tak to platí obráceně, kdy tato vrstva přijme data skrz médium a předá je vyšším vrstvám vestavěného systému ke zpracování.

Fyzická vrstva reprezentuje veškeré síťové vybavení fyzicky přítomné ve vestavěném zařízení. Na úrovni fyzické vrstvy jsou síťové komponenty připojeny přímo na přenosové médium. Vzdálenost mezi připojenými zařízeními a stejně tak síťová architektura jsou důležité vlastnosti, protože na této vrstvě mohou klasifikovat zda půjde o síť typu LAN či WAN respektive LAN či WAN protokoly, které mohou být ještě rozděleny dle přenosového média připojující zařízení k síti.

3.4.2 Linková vrstva

Linková vrstva je zodpovědná za příjem datových bitů z fyzické vrstvy a jejich naformátování do skupin nazývaných rámce. Různé standardy linkových protokolů mají odlišné formáty a definice, ale obecně tato vrstva čte bitová pole, aby zjistila zda dorazil celý rámec, jestli není poškozený, zda je určen pro nás nebo od koko rámec dorazil. Pokud je rámec určen pro zařízení, potom jsou všechny hlavičky (pokud je jich více) odstraněny a zbylé data nazývána datagram jsou předány vyšší vrstvě.

Pokud dorazí datagram z vyšší vrstvy, musí k němu být opět připojeny hlavičky, čímž vznikne plnohodnotný rámec, který je předán fyzické vrstvě, která jej odešle.

Stejně jak tomu bylo u fyzické vrstvy tak i linková vrstva klasifikuje své protokoly na LAN, WAN nebo na protokoly, které mohou být použity v obou případech. Protokoly linkové vrstvy mohou být limitovány přenosovým médiem, ale existují i takové protokoly (např. PPP nad RS-232, PPP nad Bluetooth), které mohou být portovány na velmi odlišné přenosové média. Linková vrstva také přináší možnost vzájemně propojovat sítě s různými protokoly fyzické vrstvy. Například Ethernet LAN a Wifi LAN.

3.4.3 Síťová vrstva

V OSI síťové vrstvě mohou být sítě rozděleny do menších podsítí nazývaných segmenty. Zařízení uvnitř segmentu komunikují pomocí jejich fyzických adres. Zařízení v různých segmentech komunikují pomocí speciální adresy nazývané síťová adresa. Zatímco konverze mezi fyzickou adresou a síťovou adresou může probíhat již na linkové vrstvě pomocí protokolů linkové vrstvy (např. ARP, RARP). Protokoly síťové vrstvy tak mohou provádět konverzi fyzické a síťové adresy stejně tak dokázat přiřadit síťovou adresu.

Síťová vrstva je z pohledu adresného schéma sítě zodpovědná za správu přenosu datagramů a směrování datagramů od zdrojového zařízení k cílovému. Pokud je datagram určen pro zařízení, potom jsou všechny hlavičky (pokud je jich více) odstraněny a zbylé data nazývána segment jsou předány vyšší vrstvě. Pokud dorazí paket z vyšší vrstvy jsou k němu opět připojeny hlavičky, čímž vznikne plnohodnotný datagram, který je předán linkové vrstvě.

3.4.4 Transportní vrstva

Transportní vrstva je zodpovědná za navázání a ukončení spojení mezi dvěma zařízeními. Tento typ komunikace je označen jako point-to-point komunikace. Protokoly této vrstvy tudíž umožňují vyšším aplikačním vrstvám připojit zařízení point-to-point s jiným. Některé protokoly této vrstvy mohou také zajistit spolehlivý point-to-point přenos, protože pakety jsou rozumnou rychlostí přijímány a posílány v odpovídajícím pořadí. Navíc tyto protokoly mohou poskytnout patřičnou odezvu po přijetí neporušených paketů. V případě chybějící odezvy nebo detekce chyby jsou pakety znovu přeposlány.

Když protokoly transportní vrstvy zpracují segment přijatý z nižší vrstvy, potom jsou všechny hlavičky (pokud je jich více) odstraněny a zbylá data nazývána zpráva jsou předány vyšší vrstvě. Pokud dorazí zpráva z vyšších vrstev jsou k ní opět připojeny hlavičky, čímž vznikne plnohodnotný segment, který je předán síťové vrstvě.

3.4.5 Relační vrstva

Spojení mezi dvěma síťovými aplikacemi na dvou různých zařízení se nazývána relace. Relační vrstva spravuje point-to-point spojení pro více aplikací mezi zařízeními. Správa relace je udržována relační vrstvou. Relacím jsou přirazeny porty, takže protokoly relační vrstvy musí pro každý port: oddělit a řídit data, udržovat datový tok, řešit výskyty chyb a zajistit bezpečnost.

Když relační vrstva zpracuje zprávu přijatou z nižší vrstvy, potom jsou všechny hlavičky (pokud je jich více) odstraněny a zbylá data jsou předány vyšší vrstvě. Pokud dorazí zpráva z vyšších vrstev jsou k ní opět připojeny hlavičky, čímž vznikne plnohodnotná zpráva na úrovni relační vrstvy, která je předána transportní vrstvě.

3.4.6 Prezentační vrstva

Protokoly v prezentační vrstvě jsou zodpovědné za transformaci dat do formátu vyšší aplikační vrstvy, tak aby mohla data aplikace zpracovávat. Je také zodpovědná za transformaci dat v závislosti na datovému formátu přenosu (little endian, big endian). Někdy je také v této vrstvě implementována komprese, dekomprese, šifrování dat, dešifrování dat a konverze znakových sad.

Prezentační vrstva zpracuje zprávu přijatou z nižší vrstvy, potom jsou všechny hlavičky (pokud je jich více) odstraněny a zbylá data jsou předány vyšší vrstvě. Pokud dorazí zpráva z vyšších vrstev jsou k ní opět připojeny hlavičky, čímž vznikne plnohodnotná zpráva na úrovni prezentační vrstvy, která je předána relační vrstvě.

3.4.7 Aplikační vrstva

Zařízení zahájí síťovou komunikaci s jiným zařízením právě na aplikační vrstvě. Jinými slovy, protokoly aplikační vrstvy jsou buď přímo síťové aplikace nebo protokoly implementované v síťových aplikacích. Na úrovni aplikační vrstvy se aplikace, jednoduše řečeno: „virtuálně připojují k aplikacím ostatních zařízeních“ za pomoci nižších vrstev, které již byly popsány.

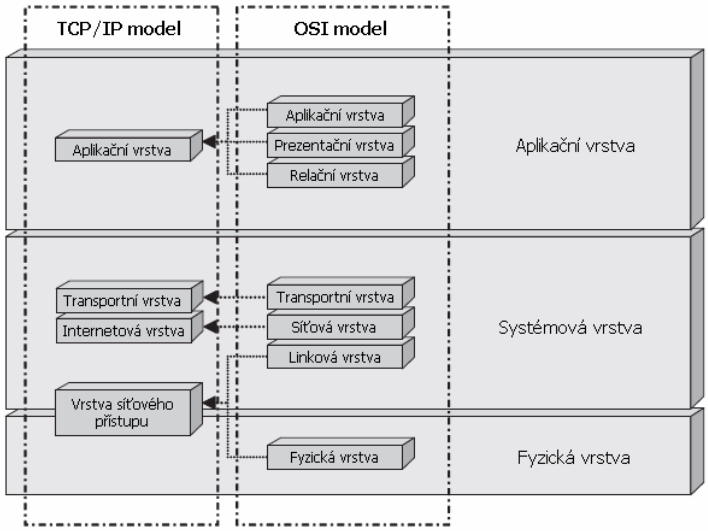
4 Podpora sítě ve vestavěném systému

Tato kapitola popisuje podporu sítě a síťové komunikace ve vestavěném systému. Nejprve představuje síťový model TCP/IP, na němž jsou často sítě ve vestavěných systémech stavěny. Pak postupně charakterizuje jednotlivé vrstvy tohoto modelu a uvádí síťové protokoly, které se mnohdy ve vestavěném systému implementují. Výjimkou je aplikační vrstva, kde je pouze stručně shrnuta funkce této vrstvy. Protokoly aplikační vrstvy totiž neplní čistou funkci přenosu dat po síti, ale jsou aplikačně závislé, což nemá s podporou sítí ve vestavěném systému nic společného.

4.1 Síťový model ve vestavěných systémech

Síťový model OSI je jednoduchý referenční nástroj užívaný především k demonstraci a pochopení obecného principu, na kterém jsou dnešní sítě postaveny [1]. V praxi není pravidlem, že je model složen ze všech sedmi vrstev a v každé vrstvě je implementován právě jeden protokol. Funkčnost jedné vrstvy modelu OSI může být implementována buď v jednom protokolu nebo může být implementována ve více protokolech. Stejně tak jeden protokol může implementovat funkčnost více vrstev OSI modelu.

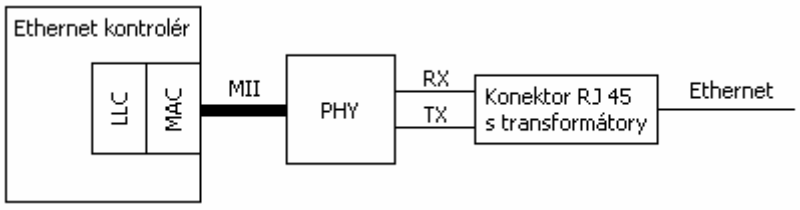
V reálném světě sítí je jedním z nejvíce využívaných modelů síťový model TCP/IP, který je složen ze čtyř vrstev: vrstvy síťového rozhraní, internetové vrstvy, transportní vrstvy a aplikační vrstvy. Když budeme sledovat vztah mezi OSI a TCP/IP modelem na obrázku 5, tak aplikační vrstva OSI modelu zahrnuje funkcionalitu tří vrchních vrstev (aplikační, prezentační a relační). Vrstva síťového rozhraní OSI modelu zahrnuje dvě spodní vrstvy (fyzickou a linkovou). Internetová vrstva koresponduje síťové vrstvě OSI modelu. Zbývající transportní vrstva je identická s transportní vrstvou OSI modelu.



Obrázek 5 : model TCP/IP integrovaný do modelu vestavěného systému

4.2 Vrstva síťového přístupu

Tato vrstva tvoří rozhraní mezi přenosovým médiem a internetovou vrstvou modelu TCP/IP. K detailnějšímu popisu funkce této vrstvy potřebujeme určit typ sítě a standard, na kterém je síť založena. Pokud se budeme bavit o sítích typu LAN, tak v nich silně převažují sítě postaveny na standardu Ethernet. V takových sítích tato vrstva plní několik funkcí [2]. Seznam těchto funkcí je následující: 1) Transformuje signály šířené přenosovým médiem do binární podoby přenášených dat, k čemuž většinou slouží speciální elektronický obvod (PHY), 2) Realizuje podvrstvu MAC, 3) Realizuje podvrstvu LLC. K přesnému pochopení těchto pojmů přispěje následující podkapitola věnující se standardu Ethernet.



Obrázek 6 : Schéma vrstvy síťového přístupu

4.2.1 IEEE 802.3 Ethernet standard

První zmínky o Ethernetu se objevují kolem roku 1970, kdy společnost Xerox přišla s článkem popisující tehdy novou technologii Ethernet [4]. V roce 1980 společnosti DEC, Intel a Xerox důkladně formalizovali Ethernet ve společném dokumentu. Teprve v roce 1985 byla stanovena IEEE konečná koncepce standardu Ethernet pod označením 802.3.

V Ethernet standardu jsou k dispozici dva módy provozu: half-duplex a full-duplex. V half-duplex módu jsou data posílány za použití populárního protokolu CSMA/CD (Carrier-Sense Multiple Access/Collision Detection) na sdíleném médiu. Hlavní nevýhodou half-duplexu je nízká efektivita, protože se o linku dělí přijímač i vysílač. Dalším nevýhodou half-duplexu je délka linky, přičemž se s větší délkou musí snižovat velikost přenášeného rámce. Toto omezení snižuje drasticky efektivitu především pro vysoké rychlosti přenosu. Ve full-duplex módu jsou data současně vysílána i přijímána pomocí dvou linek. Podpora přenosových rychlostí na médiu v síti postavené na technologii Ethernet je postačující [4]. Přesné hodnoty podporovaných rychlostí jsou postupně definovány od nejnižší po nejvyšší v následujícím seznamu:

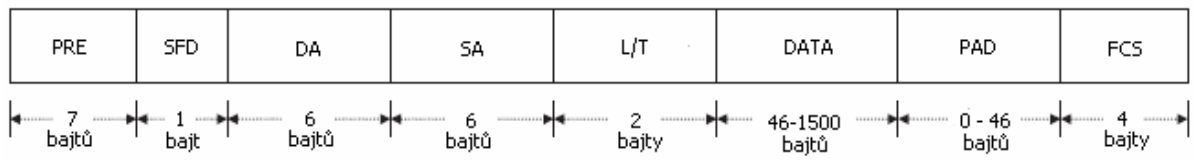
- 10 Mbps – 10TBase Ethernet (802.3).
- 100 Mbps – Fast Ethernet (802.3u).
- 1000 Mbps – Gigabit Ethernet (802.3z).
- 10-Gigabit Ethernet – IEEE 802.3ae.

Ethernet systém se skládá ze tří základních prvků: 1) z fyzického média užívaného k přenosu Ethernet signálů mezi zařízeními, 2) z množiny MAC (Medium Access Control) pravidel vestavěných v každém Ethernet rozhraní, které dovoluje více zařízením spravedlivě přistupovat ke sdílenému Ethernet kanálu a 3) Ethernet rámce, který se skládá ze standardizované množiny bitů, užívané k přenosu dat přes Ethernet systém.

Datová linková vrstva je rozdělena do dvou podvrstev, MAC (Media Access Control) a MAC-client. MAC podvrstva má za úkol zabalení dat, složení těchto dat před odesláním, kontrolu chyb během a po přijetí rámce, zahájení vysílání rámce a zotavení se z nepodařeného odvysílání. MAC-client podvrstva může představovat LLC (Logical Link Control) entitu, která poskytuje rozhraní mezi Ethernet MAC a vyššími vrstvami modelu OSI nebo Bridge entitu, která poskytuje LAN-to-LAN rozhraní mezi sítěmi, které používají stejný protokol (např. Ethernet-Ethernet) a také mezi sítěmi, které nepoužívají stejný protokol (např. Ethernet-Token Ring). LLC entita je definována v IEEE 802.2 standardu. Bridge entita je definována v IEEE 802.1 standardu.

Každé Ethernet zařízení pracuje nezávisle na všech stanicích v síti: není zde žádný centrální arbitr. Všechny stanice připojené k Ethernetu využívají sdílené médium. To pro každou stanici tedy znamená, že v případě odesílání dat musí nejdříve na kanálu naslouchat a když je kanál volný potom může teprve data odeslat. Po každém odeslaném rámci musí všechny stanice na síti rovnocenně zápasit o případnou příležitost odeslání dalšího rámce. Přístup ke sdílenému kanálu je určen MAC mechanismem. Každá rámec je odeslán do sdíleného kanálu, proto musí všechny stanice ověřit cílovou adresu. Pokud adresa souhlasí s adresou zařízení je tento rámec zařízením přečten celý a předán vyšším síťovým vrstvám. Pokud adresa nesouhlasí s adresou zařízení je rámec zahozen.

Více Ethernet segmentů může být spojeno do větší Ethernet LAN použitím opakovačů, které opětovně zesílí a synchronizují signál nebo použitím přepínačů, které signál také zesílí a synchronizují, ale ještě mohou snížit kolizní doménu. Také existují logické topologie Ethernet sítí. Ty nejznámější jsou topologie sběrnice (LAN Bus Topology) a topologie hvězda (LAN Star Topology). Struktura Ethernet rámce je znázorněna na následujícím obrázku:



Obrázek 7: Struktura Ethernet rámce

- PRE – představuje 7 bytů střídajících se log. 0 a log. 1, informuje o tom, že přichází nový rámec, také poskytuje možnost synchronizace.
- SFD – speciální vzorek bitů indikující počátek rámce.
- DA – identifikuje cílovou stanici, která by měla rámec přijmout.

- SA – identifikuje zdrojovou stanici, která rámec odeslala.
- L/T – indikuje délku rámce v bajtech nebo typ rámce (záleží na velikosti hodnoty).
- DATA – přenášená data v délce 46 až 1500 bajtů.
- PAD – doplnění dat nulami v případě menšího rámce než 64 bajtů (minimální velikost rámce)
- FCS – 32 bitový CRC kód používaný přijímací stranou k ověření, zda není rámec poškozen.

4.2.2 Protokol ARP

Protokol ARP (Address Resolution Protocol) slouží k mapování síťových IP (Internet protocol) adres na fyzické síťové adresy zařízení přítomných v Ethernet síti, nazývané MAC adresy [5]. K samotnému mapování slouží tabulka obvykle nazývána ARP cache, která udržuje korelace mezi každou MAC adresou a jí korespondující IP adresou. ARP poskytuje patřičná pravidla pro vytváření těchto korelací a poskytování konverze adres v obou směrech. Délka adresy IPv4 je 32 bitů, délka adresy MAC je 48 bitů. ARP protokol je plně definován v dokumentu RFC 826.

4.3 Internetová vrstva

Internetová vrstva zapouzdřuje síťovou vrstvu modelu OSI. To znamená, že slouží k určení jednoznačné identifikace zařízení v rámci více sítí. Jednoznačnou identifikaci zařízení zaručí síťová adresa zařízení, což je v případě internetové vrstvy IP adresa. Tato adresa identifikuje i datagram, který se přenáší po internetu a tudíž může být pomocí směrovačů úspěšně doručen cílovému zařízení.

4.3.1 IPv4 protokol

IP protokol obsahuje adresní a kontrolní informace k umožnění směrování datagramů v síti [5]. Jde také o jeden ze základních protokolů spadající do TCP/IP modelu. Společně s protokolem TCP je srdcem internetových protokolů. Internet protokol je stejný pro komunikace na sítích typu LAN i WAN. IP protokol má dva primární účely: poskytnutí doručení datagramů na nespojově orientované síti a poskytnutí fragmentace a znovu složení datagramů na linkách s odlišnou hodnotou MTU (Maximum Transmission Unit).

IP adresy mohou být rozděleny a použity k vytvoření adres pro podsítě. Každému zařízení na TCP/IP síti je přidělena unikátní 32-bitová logická adresa, která je rozdělena do dvou částí: na část síťové adresy definující adresu sítě a část adresy zařízení v rámci této sítě. Část síťové adresy v celé 32-bitové adrese určuje síťová maska.

Když jsou nějaké data posílána přes internet jsou rozdělena do menších kousků nazývaných paket. Každý paket obsahuje IP adresu cíle i zdroje. Jelikož byla data rozděleny do paketů může každý z nich cestovat po síti odlišnou cestou a tudíž mohou tyto pakety dorazit k cílové stanici v jiném pořadí. Může se i stát, že některé pakety nedorazí vůbec. Z toho vyplývá, že musí existovat

další vyšší vrstva, která zaručí spolehlivé doručení paketů ve správném pořadí. IPv4 protokol je plně definován v dokumentu RFC 791.

4.3.2 Protokol ICMP

Protokol ICMP (Internet Control Message Protocol) je postaven na IP protokolu. ICMP zprávy doručeny v IP paketech se používají především za účelem poskytování informací o událostech spojených se sítí [5]. ICMP pakety jsou doručovány nespolehlivě, takže zařízení nemohou počítat s přijmutím zprávy v případě síťového problému. Klíčové funkce ICMP jsou: oznamovat síťové chyby, oznamovat síťové zahlcení, oznamovat časové vypršení a asistence při odstraňování problému na síti. ICMP protokol je plně definován v dokumentu RFC 792.

4.4 Transportní vrstva

Transportní vrstva modelu TCP/IP je identická transportní vrstvě modelu OSI. Tudíž pro ni platí stejné zásady. Krátce řečeno, jejím hlavním úkolem je zabezpečit spolehlivý transport dat mezi dvěma zařízeními a umožnit více aplikacím na vyšší vrstvě sdílet síťové rozhraní pomocí síťových portů.

4.4.1 Protokol TCP

TCP protokol poskytuje spolehlivé doručení dat pomocí sekvenčního potvrzování s případným přeposláním dat, když je potřeba [5]. Na jednom zařízení může být spuštěno více síťových aplikací současně. Nastává problém tehdy, když aplikace na zdrojovém zařízení zasílá data aplikaci na cílovém zařízení, protože aplikace nejsou nijak identifikovány. Tento problém řeší tzv. TCP porty. Každé síťové aplikaci je přirazen podle určitých pravidel číslo portu, pak je každá aplikace identifikována číslem portu. Z toho vyplývá, že je aplikace v zařízení identifikována v rámci sítě dvojicí a to IP adresou a číslem portu. Někdy se tato dvojice označuje jako “socket“, neboli zásuvka k aplikaci.

Při počátku komunikace pomocí TCP protokolu musíme mezi dvěmi zařízeními sestavit spojení (virtuální okruh). V průběhu komunikace se data přenášejí v tomto virtuálním okruhu. Po skončení komunikace se spojení může přerušit.

TCP protokol má celkem čtyři přednosti: 1) je spolehlivý, lehce se vypořádá se ztrátou, zpoždění, duplicitou či jiném pořadí přijatých paketů, 2) má efektivní kontrolu datového toku, 3) podporuje plně duplexní přenos, 4) podporuje multiplexování. TCP protokol je plně definován v dokumentu RFC 793.

4.4.2 Protokol UDP

Tento protokol používá stejným způsobem porty jako protokol TCP, ale narozdíl od protokolu TCP je tento protokol nespolehlivý, nepodporuje kontrolu datového toku a nemá žádné metody k zotavení z poruchy [5]. To znamená, že může docházet u tohoto protokolu ke ztrátám paketů, ale tento nedostatek je vykompenzován menším počtem protokolových informací posílaných v paketu a nízkou réžijí spojenou s činností protokolu. To znamená, že je používán v oblastech, kde není vyžadována vysoká spolehlivost, ale vysoká propustnost dat. Jako příklad si můžeme uvést internetové vysílání rádia nebo videa. UDP protokol je plně definován v dokumentu RFC 768.

4.5 Aplikační vrstva

Aplikační vrstva představuje cílové síťové aplikace či síťové protokoly, které v sobe zapouzdřují metody spojené s relací spojení, prezentací dat i samotnou aplikací. Mezi nejznámější protokoly této vrstvy patří HTTP, FTP, SMTP, IMAP4, POP3, SNTP nebo DNS.

5 Platforma FITKIT

Tato kapitola se snaží objasnit pojem platforma FITkit. Charakterizuje důležité vlastnosti této platformy jako je její účel, koncepce, struktura a přínos. Dále uvádí způsob návrhu a implementace aplikací na platformě FITkit, přičemž se pozastavuje u podpory tvorby síťových aplikací.

5.1 Vlastnosti platformy FITKIT

5.1.1 Účel

Cílem nasazení platformy FITkit do výuky je umožnit studentům, aby mohli navrhovat a prakticky realizovat nejen softwarové, ale také hardwarové projekty či dokonce celé aplikace. Účelem této platformy také je, aby každý student měl FITkit k dispozici a aby s ním mohl pracovat ve škole, doma i na kolejích. Důležité je, že FITkit bude využíván ve výuce řady kurzů a studenta bude provázet po celou dobu studia napříč bakalářským i magisterským studijním programem. Znalosti, získané ve výuce, pak bude moci využít pro tvorbu ročníkových, semestrálních a diplomových prací. Výsledkem by měla být výrazná podpora výuky technologií hardware a software výpočetních systémů s důrazem na praktické aplikační výstupy s tím, že výsledky budou zpřístupněny a využitelné pro co nejširší okruh zájemců nejen z řad studentů informatiky [6].

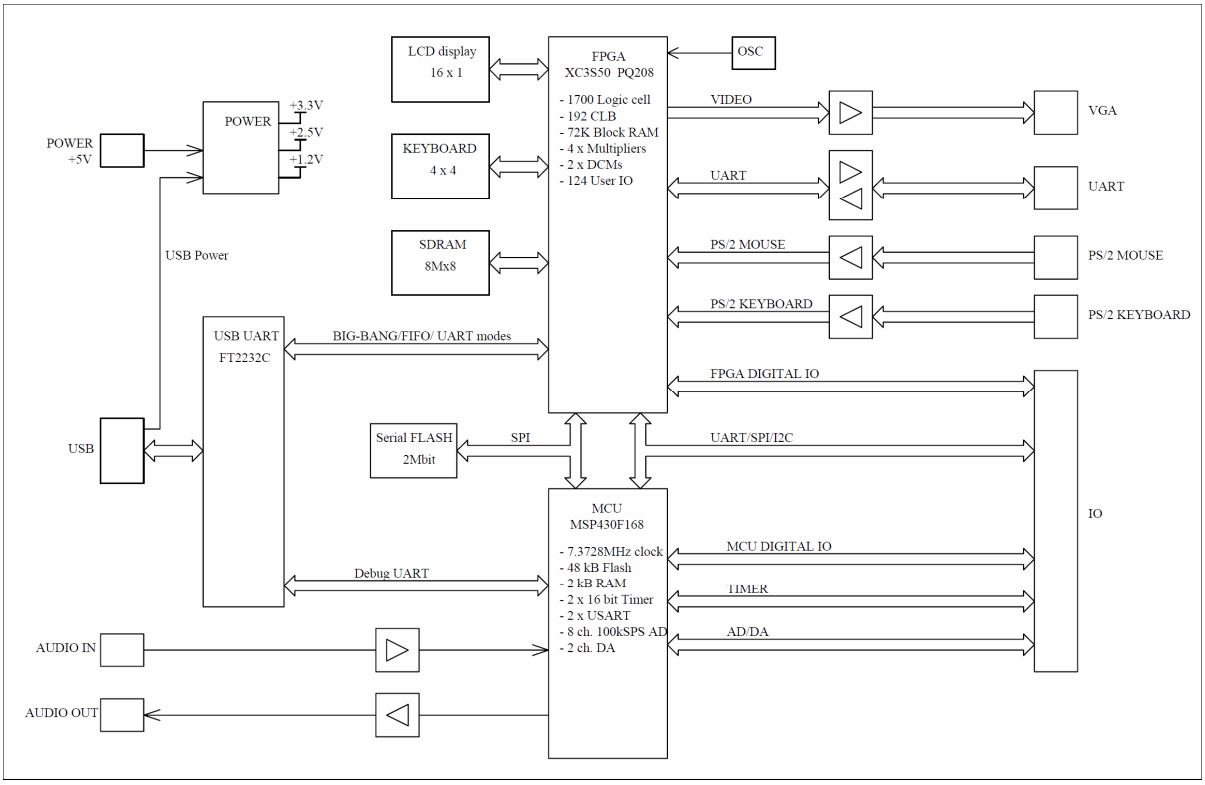
5.1.2 Koncepce

Platforma je koncipována jako open-source (pro software) a open-core (pro hardware), což znamená, že veškeré výsledky práce studentů s platformou FITkit jsou přístupné na internetu ve zdrojové formě pro kohokoli [6]. Cílem tohoto konceptu je nastartovat efekt, kdy každý, kdo využije výsledky práce vytvořené uživateli platformu FITkit, bude je dále poskytovat ve zdrojové formě a umožní tím jejich použití pro všechny zájemce jak z řad studentů FIT, tak studentů z jiných škol či odborné veřejnosti.

5.1.3 Struktura

FITkit obsahuje dva podstatné prvky definující celkovou funkčnost a výkonnost této platformy. Prvním prvkem je mikrokontrolér s nízkým příkonem od společnosti Texas Instrument sloužící k seznámení se s obecnou problematikou programování mikrokontrolérů a jejich vnitřních periferií. Druhým neméně důležitým prvkem této platformy je programovatelné hradlové pole (FPGA - Filed Programmable Gate Array) od společnosti Xilinx, pomocí kterého je možné porozumět problematice reprogramování hardwaru na bázi hradlových polí FPGA jenž lze, podobně jako software na počítači, neomezeně modifikovat pro různé účely dle potřeby. Z toho tedy vyplývá, že uživatel nemusí

vytvářet nový hardware pro každou aplikaci znovu. Na desce je navíc přítomna paměť typu SDRAM o kapacitě 1 MB a paměť typu FLASH o kapacitě 256 KB. Tyto paměti mohou být využity v programech samotného FPGA nebo mikrokontroléru. FITkit nabízí základní vstupní či výstupní zařízení. Vstupní zařízením je maticová klávesnice 4x4. Výstupním zařízení je řádkový LCD displej podporující zobrazení s rozlišením 1x16 znaků. V neposlední řadě FITkit oplývá řadou vstupních, výstupních nebo vstupně/výstupních datových rozhraní. Vstupní datové rozhraní představují dva porty PS2 a zvukový vstup. Výstupní rozhraní zahrnuje VGA rozhraní a zvukový výstup. A nakonec vstupně/výstupní rozhraní tvoří RS232 kanál, USB-RS232 kanál, dvě SPI rozhraní, AD či DA převodník a obecné GPIO porty připojené k mikrokontroléru nebo FPGA. Celková struktura platformy FITkit je znázorněna následujícím blokovým diagramem.



Obrázek 8 : Blokový diagram platformy FITKIT

5.1.4 Přínos

Hlavním přínosem platformy FITkit je fakt, že umožňuje obsáhnout značnou část spektra znalostí a dovedností, které poté absolvent může uplatnit v praxi a to především v oblasti vestavěných systémů, které se v dnešní době dominantně uplatňují v běžném životě a jejichž význam ještě výrazně poroste [6].

5.2 Návrh aplikací na platformě FITKIT

Při návrhu aplikací se využívá skutečnosti, že vlastnosti hardwaru se v dnešní době převážně popisují vhodným programovacím jazykem (např. VHDL), díky čemuž se návrh softwaru a hardwaru provádí do značné míry obdobně. Veškerý potřebný návrhový software je k dispozici zdarma [6].

Software pro FGPA je tvořen popisem v jazyce VHDL. Z tohoto popisu lze pomocí profesionálních návrhových systémů zcela automaticky vygenerovat výstupní konfiguraci pro FPGA. O samotné načítání konfigurace do FPGA se stará mikrokontrolér. Ten může konfiguraci pomocí terminálového protokolu Xmodem nahrát přímo do FPGA nebo ji zapsat do paměti typu FLASH, ze které bude konfigurace do FPGA načítána při startu FITkitu [7].

Software pro mikrokontrolér je vyvíjen v jazyce C a do spustitelné formy se překládá pomocí volně dostupného GNU překladače. Vygenerovaný kód se nahraje do mikrokontroléru pomocí speciálního programu komunikující přes sériový port [7].

5.2.1 Podpora síťových aplikací

Jak již bylo řečeno v podkapitole zabývající se strukturou platformy FITkit, je tato platforma vybavena skupinou vstupně/výstupních rozhraní. Do této skupiny patří: RS232 kanál, USB-RS232 kanál, dvě SPI rozhraní, AD či DA převodník a obecné GPIO porty připojené k mikrokontroléru nebo FPGA. V této skupině očividně není žádné přímé síťové rozhraní, z čehož plyne, že vyvíjet síťové protokoly či implementovat samotné síťové aplikace na platformě FITkit nelze. Tento problém lze vyřešit pouze rozšiřujícím modulem síťového rozhraní.

6 Specifikace modulu rozhraní Ethernet

Tato kapitola se zaměřuje na koncepci, operační módy a přístupové schéma modulu rozhraní Ethernet pro platformu FITkit. V prvé řadě popisuje koncepci rozšiřujícího modulu platformy FITkit, ve které se snaží nastínit hrubé řešení dané problematiky. Dále popisuje operační módy, ve kterých modul pracuje a jejich specifické vlastnosti související s funkcí modulu. Nakonec detailně charakterizuje přístupové schéma modulu, ve kterém analyzuje možné varianty přístupu k modulu z FITkitu a snaží se o výběr optimálního řešení.

6.1 Koncepce modulu

K nejrozšířenějším síťovým standardům v lokálních počítačových sítích nepopíratelně patří technologie Ethernet. Z toho lze usoudit, že při výběru vhodné síťové technologie pro rozšiřující síťový modul platformy FITkit vezmeme v potaz právě tento fakt a zvolíme jako nejvhodnější síťovou technologii právě Ethernet. Tímto krokem do budoucna zajisté zaručíme maximální přenositelnost a universálnost síťových řešení založených na FITkitu v prostředí lokálních sítí. Modul, který bude toto síťové rozhraní poskytovat budeme dále označovat a nazývat modul rozhraní Ethernet či zkráceně EIM (Ethernet Interface Module).

Modul rozhraní Ethernet je integrován do platformy FITkit formou externího modulu, který komunikuje s FITkitem prostřednictvím sériového rozhraní UART nebo rychlejšího sériového rozhraní SPI. Modul je vybaven funkcí pro příjem Ethernet rámců, které dále předává k zpracování FITkitu. Toto samozřejmě platí i obráceně. To znamená, že je modul také vybaven funkcí pro vysílání Ethernet rámců, které přebírá od FITkitu. Těmito funkcemi je zřejmě zajištěn přístup k linkové vrstvě síťového modelu a dává tedy možnost zpracovávat další vyšší síťové vrstvy až po samotnou aplikační vrstvu. Z čehož vyplývá, že programátor může implementovat síťové protokoly na těchto vrstvách a síťové aplikace na nich postavených. Modul lze samozřejmě eventuálně nakonfigurovat nebo jeho konfiguraci získat. Děje se tak přechodem do stavu konfigurace, ve kterém lze získat či nastavit specifické parametry ať už síťové konfigurace, komunikační konfigurace nebo funkcionální konfigurace. Funkce modulu rozhraní Ethernet je tedy závislá na stavu, ve kterém se nachází. Tyto stavy budou dále uváděny jako operační módy modulu a jejich význam je podrobně popsán v následující samostatné podkapitole.

Z předešlého odstavce tedy vyplývá, že modul rozhraní Ethernet ve se své podstatě realizuje transparenci fyzické síťové vrstvy a přináší FITkitu přímé rozhraní k linkové vrstvě sítě, přičemž disponuje možností specifické konfigurace patřičných parametrů spojených se správnou funkcí síťové komunikace.

6.2 Operační módy modulu

Modul rozhraní Ethernet pracuje vždy v jednom z daných operačních módu. Operační módy existují dva a to konkrétně mód konfigurační a mód datový.

V konfiguračním módu je možné modul konfigurovat nebo z něj konfiguraci získat. Konfigurace členíme na tři typy: funkční, komunikační a síťovou. *Funkční konfigurace* ovlivňuje parametry spojené s funkcionalitou modulu. Do této sféry patří konfigurace ovlivňující typ datového módu. Tato konfigurace je rezervovaná pro budoucí použití, protože prozatím není k dispozici více typů datových módů, ale pouze jeden. Musíme, ale počítat s možností rozšíření modulu o další typy datového módu, což vyžaduje právě tuto konfiguraci. *Komunikační konfigurace* ovlivňuje parametry spojené s datovou komunikací mezi FITkitem a modulem. Spadá sem konfigurace ovlivňující typ komunikačního zařízení nebo rychlosti komunikace. Poslední *Síťová konfigurace* je využita ke konfiguraci parametrů spojených s rozhraním lokální sítě. Mezi tyto parametry patří hardwarová adresa zařízení, internetová adresa, maska sítě a adresa brány.

Konfigurace probíhá zasláním konfiguračního příkazů modulu, který tento příkaz identifikuje, provede samotnou konfiguraci a vrátí status zpracování příkazu nabývající hodnoty korektně provedeného příkazu nebo hodnoty určující příčinu nepodařeného provedení příkazu. V případě korektně provedeného příkazu jsou jakékoli změny spojené s významem příkazu okamžitě platné. Typ konfiguračního příkazu je buď nastavovací s prefixem “SET“ nebo získávací s prefixem “GET“. Nastavovací příkazy nesou konfigurační data před jejich zpracováním, kdežto získávací příkazy nesou konfigurační data až po jejich zpracování. Přesný popis konfiguračních příkazů bude následovat v kapitole zabývající se implementací konfigurátoru. Následuje tabulka s přehledem všech podporovaných konfiguračních příkazů:

Tabulka 2 : Přehled konfiguračních příkazů

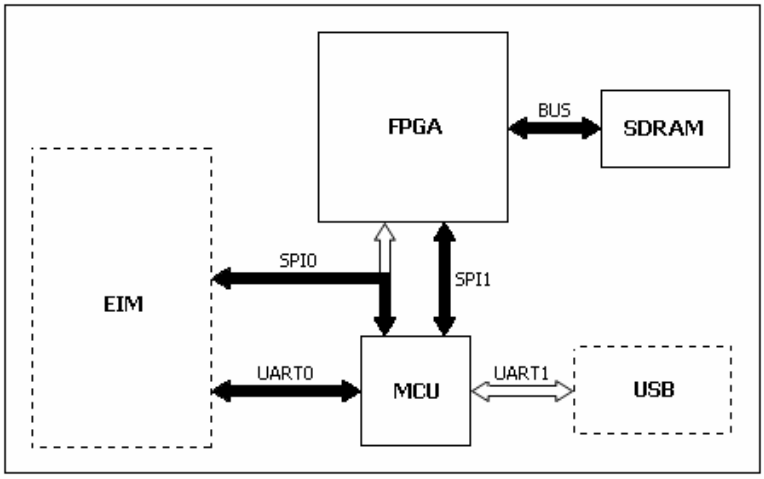
Typ konfigurace	Název příkazu	Význam
Funkční	SETDTAMOD	Nastaví typ datového módu
	GETDTAMOD	Získá typ datového módu
Komunikační	SETCOMDEV	Nastaví typ komunikačního zařízení
	GETCOMDEV	Získá typ komunikačního zařízení
	SETCOMSPD	Nastaví rychlost komunikačního zařízení
	GETCOMSPD	Získá rychlost komunikačního zařízení
Síťová	SETIP	Nastaví internetovou adresu
	GETIP	Získá internetovou adresu
	SETMASK	Nastaví masku sítě
	GETMASK	Získá masku sítě
	SETGW	Nastaví adresu brány
	GETGW	Získá adresu brány
	SETMAC	Nastaví HW adresu modulu
	GETMAC	Získá HW adresu modulu

Druhým operačním módem je mód datový. V datovém módu modul zpracovává data ze sítě, které pak zasílá FITkitu nebo FITkit zasílá data modulu a ten je odesílá do sítě. Modul vlastně vykonává funkci fyzické vrstvy síťového modelu, kdy FITkitu zasílá přijatý rámec linkové vrstvy. FITkit zase zasílá rámce linkové vrstvy modulu, který je vzápětí odesílá do sítě. V tomto módu nelze provádět konfiguraci modulu. Kdyby se tak stalo, pokus o konfiguraci bude ignorován a ohlášen patřičným chybovým návratovým kódem.

6.3 Přístupové schéma modulu

Pokud se zaměříme přímo na aktivní prvek FITkitu, který bude realizovat přístup k modulu rozhraní a tím poskytne vhodné zázemí pro funkce komunikující s modulem dojdeme k závěru, že lze využít jak mikrokontrolér tak i FPGA.

První varianta s mikrokontrolérem využije ke komunikaci rozhraní UART nebo SPI. Konkrétní výběr komunikačního rozhraní je proveden pomocí konfiguračního příkazu „SETCOMDEV“, který má jako parametr identifikátor určující typ komunikačního rozhraní potažmo zařízení. Výhoda tohoto řešení je implementace funkcí ve strukturálním jazyce, což značně snižuje složitost návrhu i vývoje takových funkcí. Další výhodou je přenositelnost takového kódu, takže návrh a implementace funkcí související přímo s komunikačním rozhraním modulu Ethernet se provede pouze jednou, což vede k značné úspoře času i práce. Nevýhodou tohoto řešení je malá kapacita datové paměti, která činí pouhé dva kilobajty. S touto kapacitou je prakticky nemožné síťové aplikace realizovat. Východiskem z této svízelné situace je využití externí datové paměti na FITkitu za pomoci řadiče implementovaného v FPGA. Následující blokové schéma znázorňuje tuto variantu s vyznačenou datovou cestou.



Obrázek 9 : Blokové schéma rozhraní modulu

Druhá varianta přístupu k modulu spočívá v návrhu a implementaci jednotky správy komunikace s modulem přímo do FPGA čipu. Mikrokontrolér by v tomto případě pouze zapisoval nebo četl speciální registry této jednotky. Toto řešení je jistě elegantnější, ale rozhodně mnohonásobně složitější a časově náročnější.

Existuje i třetí varianta přístupu, která je sice trošku úsměvná, ale přesto zajímavá. V této variantě je mikrokontrolér na FITkitu využit pouze jako most mezi modulem rozhraní Ethernet a USB rozhraním. Funkce jsou implementovány formou knihoven až na úrovni osobního počítače, čímž by se vlastně z FITkitu stala externí síťová karta připojitelná pomocí USB portu.

Po tomto rozboru je zřejmé, že optimální řešení přináší první varianta, kdy hlavní úlohu spojenou ovládání síťového rozhraní přebírá mikrokontrolér a FPGA čip slouží pouze ke komunikaci s datovou pamětí.

7 Hardwarové řešení

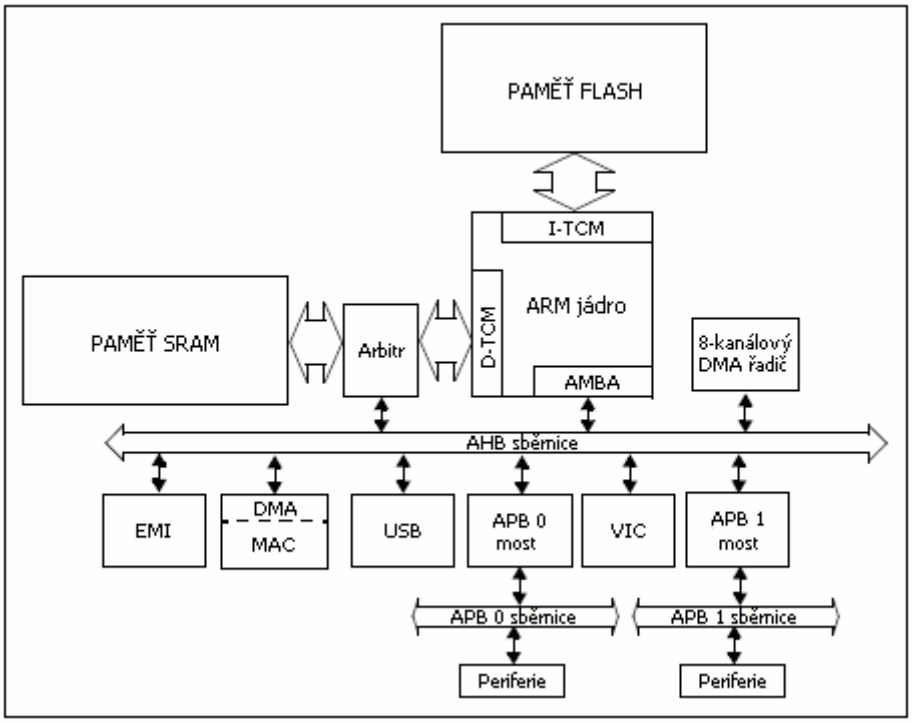
Tato kapitola se zaměřuje na hardwarové řešení modulu rozhraní Ethernet. Celý problém hardwarového řešení rozděluje na menší podproblémy, které představují funkční bloky provádějící svoji specifickou úlohu. Mezi tyto bloky patří procesorová jednotka, fyzická vrstva Ethernetu, datové úložiště, definice vstupů/výstupů a napájení. Po rozboru a popisu těchto bloků následuje charakteristika struktury zapojení, která dává souhrnný přehled o propojení jednotlivých funkčních bloků. Kapitola je zakončena pohledem na samotnou realizaci modulu se zaměřením na tvorbu schéma zapojení a tvorbu desky plošného spoje.

7.1 Procesorová jednotka

Nejprve je nutné zvolit vhodnou procesorovou jednotku, která by disponovala dostatečným výkonem a požadovanými periferiemi. Za dostatečně výkonnou procesorovou jednotku můžeme označit zařízení vybavené RISC instrukční sadou a pracující alespoň na frekvenci 25Mhz, protože jen tehdy dokáže zpracovávat data rychlostí 100 Mbit/s pomocí obvodu fyzické vrstvy ethernetu. Aby bylo řešení co nejvíce kompaktní je nutná přítomnost co nejvíce elektronických prvků přímo v procesorové jednotce. Na procesorové jednotce by měly být přítomné minimálně následující periferie: komunikační asynchronní rozhraní UART, komunikační synchronní rozhraní SPI a řadič Ethernetu s vrstvou MAC 802.3. Na trhu je mnoho vyhovujících procesorových jednotek od různých výrobců, ale samotný výběr těžký nebude, jelikož jsem měl možnost pracovat s procesorem řady STR912F vyrobený společností ST Microelectronics [8]. Tento procesor plně vyhovuje našim požadavkům a navíc je jeho dostupnost a cena také přívětivá. Takže jej mohu bez okolků zvolit jako hlavní procesorovou jednotku modulu rozhraní Ethernet.

Tento procesor spadá do série 32 bitových mikrokontrolérů vybavených výkonným ARM jádrem s označením ARM966E-S RISC pracujícím na frekvenci v rozmezí 25-96 Mhz. Toto jádro je postaveno na Harvardské architektuře, ve které je oddělena kódová paměť od datové paměti. Kapacita kódové paměti je 512 KB a datové 96 KB, což je pro naše řešení naprosto dostačující. Samotná centrální procesorová jednotka je skalárního typu s řetězenou linkou o hloubce pěti instrukcí. Jádro je tedy právem rozšířeno o předpřipravovací frontu instrukcí a skokovou cache tabulku o velikosti patnácti záznamů. Tímto rozšířením se zajistí kratší doba čekání na další požadovanou instrukci. Pokud se zaměříme na vnitřní periferie procesorové jednotky, zjistíme, že jedná o velmi rozmanitý celek, který je tvořen blokem pro správu napájení, obvodem reálného času, řadičem přerušení, A/D převodníky, rozhraním pro připojení externí paměti, čtyřmi časovači, řadičem třífázového motoru a následujícími komunikačními rozhraními: 10/100 Ethernet MAC s podporou DMA, USB FullSpeed (12 Mbps), CAN 2.0B Active, třemi bloky UART s podporou IrDA protokolu, dvěmi IIC bloky

a dvěmi bloky pro SPI. Procesor dále zahrnuje standardizované JTAG rozhraní, díky kterému lze v mnoha vývojových nástrojích pro vestavěné systémy ladit kód přímo za běhu a tak zefektivnit fázi implementace. Následující obrázek znázorňuje blokové schéma, čímž dává souhrnný přehled o stavbě procesoru.



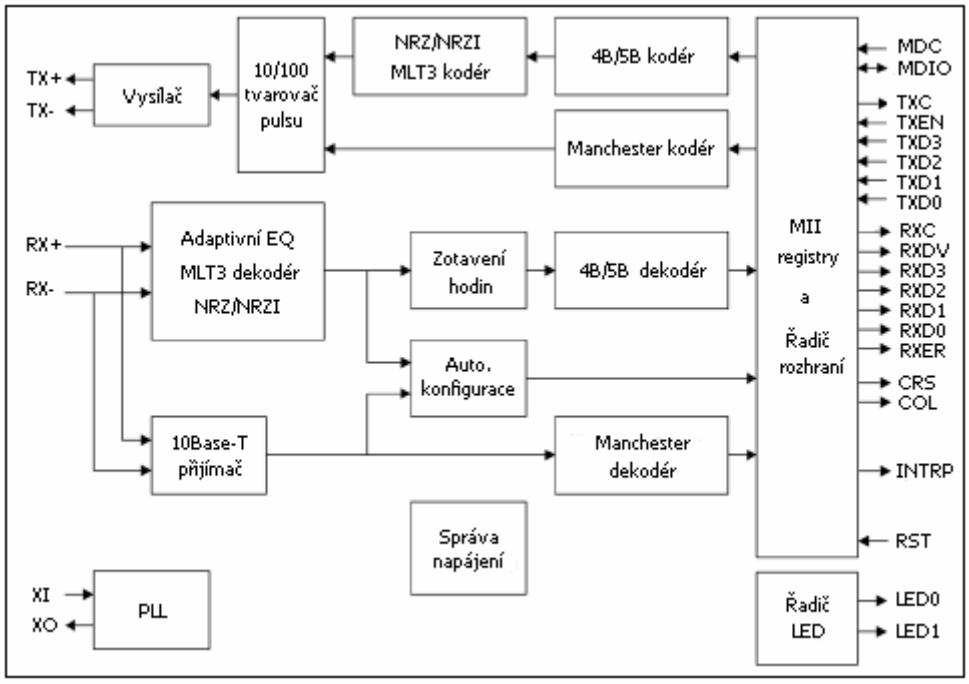
Obrázek 10 : Blokové schéma procesorové jednotky

7.2 Fyzická vrstva Ethernetu

Fyzickou vrstvu Ethernetu tvoří obvod s názvem PHY. Tento obvod je prostředníkem mezi Ethernet MAC v procesorové jednotce a daty na fyzickém médiu. Jeho hlavním úkolem je upravit fyzický signál odpovídající standardu Ethernet na data v bitové formě a naopak transformovat data z bitové formy na fyzický signál. Komunikace mezi Ethernet MAC a PHY probíhá pomocí standardizovaného rozhraní MII (Media Independent Interface) na frekvenci 25 Mhz, kdy se v každém taktu zpracují 4 bity, čímž je možné dosáhnout rychlosti až 100 Mbit/s. Existuje ještě redukovaná verze rozhraní nazývaná RMII, která pracuje na 50 MHz s podstatně menším počtem signálů, ale tuto verzi bohužel nepodporuje Ethernet MAC v naší procesorové jednotce. Z čehož vyplývá, že můžeme využít pouze rozhraní MII, i když obvod PHY podporuje obě rozhraní. Pokud se zaměříme na rozšířené funkce těchto obvodů, tak většina z nich podporuje automatickou konfiguraci spojení (Auto-Negotiation) a detekci špatně zvoleného typu kabelu. Obvod PHY s aktivovanou funkcí automatické konfigurace spojení dokáže po připojení kabelu dohodnout s protější stranou parametry spojení jako je přenosová

rychlost a přenosový mód. Funkce pro detekci špatně zvoleného typu kabelu (např. záměna přímého s kříženým) je schopna odhalit tuto vadu a automaticky ji napravit přepnutím příslušných signálů.

Pro naše účely je ideálním kandidátem obvod PHY KSZ8041NL jehož výrobcem je společnost Micrel [9]. Hlavní výhodou toho obvodu je přítomnost všech standardních funkcí, slušná dostupnost, nízká cena a především malý rozměr. Tento obvod podporuje rychlosti 10/100 Mbit/s s možností přenosových módů Half duplex i Full duplex. Z rozšířených funkcí obvod samozřejmě podporuje automatickou konfiguraci spojení a detekci špatně zvoleného typu kabelu s automatickou nápravou označovanou výrobcem jako Auto MDI/MDI-X. Následující obrázek podává podrobný přehled základných funkčních bloků, ze kterých se tento obvod PHY skládá.



Obrázek 11 : Blokové schéma obvodu PHY KSZ8041NL

Samotný obvod PHY by nepostačoval k plnohodnotné funkci fyzické vrstvy Ethernetu, protože obvod musí být galvanicky oddělený od sítě Ethernet. Toho se docílí pomocí oddělovacích transformátorů, které mimo oddělování slouží k filtraci a snížení šumu v signálu. V dnešní době existují konektory RJ-45, které mají zabudovány oddělovací transformátory přímo v sobě, čímž se ušetří místo na desce plošných spojů a sníží délka signálových cest. Jedním z dobře dostupných konektorů, který tyto vlastnosti splňuje je konektor od výrobce BELFuse s označením MAGJACK SI 52003-F [10]. Tento konektor je navíc vybaven dvěmi LED diody indikující přenosový mód a průběh datové komunikace.

7.3 Datové úložiště

V řešení modulu rozhraní Ethernet by rozhodně nemělo chybět datové úložiště pro zajištění perzistence důležitých dat. Důležitými daty může být konfigurace modulu, ale i jakákoli jiná data, která je třeba uchovat. Konkrétní obvod realizující tuto funkci je paměť typu FLASH s označením M45PE80 od výrobce Numonyx [11]. Kapacita této paměti je 1MB. Přístup k paměti je zajištěn pomocí SPI rozhraní, přičemž rychlost při optimální frekvenci dosahuje přibližně 300KB/s při operaci čtení a 180KB/s při operaci zápisu.

7.4 Vstupy a výstupy

Modul obsahuje čtyři typy rozhraní vstupů a výstupů: obecné rozhraní, rozhraní FITkitu, JTAG rozhraní a Ethernet rozhraní:

Obecné rozhraní umožňuje využít co nejvíce komunikačních rozhraní mikrokontroléru. Mezi tyto komunikační rozhraní patří: USB, IIC, SPI, UART, AD převodník a obecné porty. Takovou menší poznámkou hodící se právě na toto místo je skutečnost, že při návrhu modulu bylo počítáno i s variantou, kdy modul nebude využit jako síťové rozhraní FITkitu, ale jako obecně funkční kit využitelný v mnoha aplikacích. To je víceméně dáno podporou síťového rozhraní, přítomností datového úložiště o kapacitě 1MB, dostupností prakticky všech komunikačních rozhraní mikrokontroléru, přítomností standardizovaného ladícího JTAG rozhraní a přítomností vlastní správy napájení.

FITkit je připojen k mikrokontroléru pomocí rozhraní FITkitu. Toto rozhraní je tvořeno dvěmi datovými komunikačními rozhraními UART a SPI, což dává programátoru podle daných potřeb možnost výběru výhodnější datové komunikace.

Standardizované JTAG rozhraní je rozhraní určené pro JTAG ladiče. Díky tomuto rozhraní je zpřístupněno použití mnoha ladících zařízení od různých výrobců, kteří s nimi poskytují komfortní ladící nástroje společně s vývojovým prostředí pro vestavěné systémy.

Ethernet rozhraní je tvořeno obvodem PHY, ke kterému je připojen konektor RJ-45 vybavený oddělovacími transformátory.

7.5 Napájení

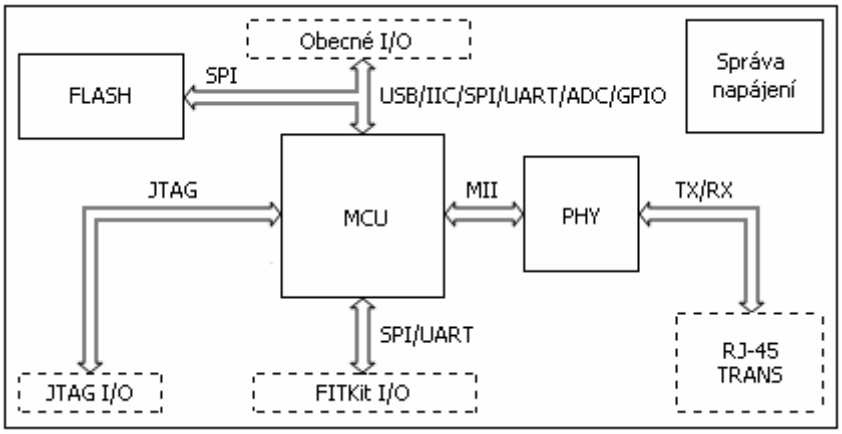
Modul se napájí napětím o velikosti 5V. Zdroj tohoto napětí je volen pomocí propojky na desce. Propojka umožňuje vybrat napájení buď z FITkitu nebo z JTAG ladiče. Napájení z FITkitu by mělo být zvoleno v případě, když k němu bude modul připojen. Pokud je třeba ladit modul bez FITkitu je možné zvolit zdroj napětí z JTAG ladiče a tak ladit modul samostatně.

Vstupní napětí o velikosti 5V je samozřejmě nutné stabilizovat na hodnoty vyžadované elektronickými prvky na modulu. Hodnoty stabilizovaných napětí jsou 3.3V pro veškerou logiku na desce a 1.8V pro jádro procesorové jednotky. Vyhledat vhodný obvod, který by splňoval tyto podmínky dalo značnou práci, ale nakonec se podařilo nalézt lineární stabilizátor TPS70151PWP od výrobce Texas Instrument [12]. Tento obvod dokáže duálně stabilizovat vstupní napětí na dvě výstupní napětí o velikosti 3.3V respektive 1.8V s možností dodání proudu o velikosti 500mA respektive 250mA. Dostupnost a cena tohoto obvodu je přijatelná. Velkou předností tohoto obvodu je především jeho podpora duální stabilizace, nízké rozměry a možnost chlazení s využitím plošného spoje. Tyto přednosti z něj vytváří velice elegantní a kompaktní řešení napájení.

7.6 Struktura zapojení

Struktura zapojení je patrná z obrázku 12. Hlavním prvkem této struktury je mikrokontrolér, který plní funkci hlavní řídící jednotky modulu rozhraní Ethernet a zpřístupňuje co nejvíce svých systémových zdrojů přesněji řečeno komunikačních rozhraní, tak aby mohl být využit maximální počet jeho schopností. K mikrokontroléru jsou připojeny následující bloky:

- Fyzická vrstva Ethernetu pomocí rozhraní MII.
- Datové úložiště tvořené FLASH pamětí, které je připojeno pomocí sběrnice SPI. Toto rozhraní je také svedeno do obecně vstupně/výstupního rozhraní.
- Vstupy a výstupy modulu rozhraní Ethernet.
- Správa napájení starající se o stabilizaci a přívod požadovaného napájení ke všem elektronickým prvkům modulu.



Obrázek 12 - Blokový diagram modulu rozhraní Ethernet

7.7 Praktická realizace

Při praktické realizaci jsem využil široce zastoupeného nástroje pro tvorbu elektronických schémat a desek plošných spojů EAGLE verze 4.16r2. Nejprve bylo nutné navrhnout a nakreslit schéma zapojení. Toto schéma je rozděleno do třech logických částí. První část znázorňuje zapojení fyzické vrstvy Ethernetu, druhá část znázorňuje jádro modulu a poslední část se zaměřuje na napájení modulu. Přesná podoba schémat je přiložena do přílohy na konci této práce. Dalším krokem bylo transformovat schéma zapojení na desku plošných spojů. Tato transformace zahrnovala tvorbu osmi knihoven obsahující součástky přesných rozměru a popisů. Seznam knihoven je následující:

- CAPACITORS – knihovna kondenzátorů.
- CHIPS – knihovna čipů.
- CONNECTORS – knihovna konektorů.
- CRYSTALS – knihovna krystalů.
- DIODES – knihovna diod.
- INDUCTORS – knihovna indukčností.
- JUMPERS – knihovna propojek.
- RESISTORS – knihovna rezistorů.

V okamžiku dokončení knihoven součástek nastala fáze tvorby obrazce plošných spojů podle schématu zapojení a přesných rozměrů součástek. Výstupem tohoto postupu byla dvouvrstvá deska plošného spoje definována několika vrstvami:

- TOP – vrchní vrstva spojů.
- BOTTOM – spodní vrstva spojů.
- PADS – vrstva vrtaných pájecích plošek.
- VIAS – vrstva pokovených děr.
- DRILLS – vrstva pozic a rozměrů vrtáků.
- HOLES – vrstva pozic a rozměrů děr.
- DIMENSION – vrstva rozměru desky.
- PLACE – vrstva rozmístění a názvů součástek.
- DOCUMENT – vrstva dokumentace součástek.
- PRINT – vrstva potisku desky.

Po návrhu bylo třeba provést DRC kontrolu desky plošných spojů obsahující seznam testů validity desky. Mezi tyto testy patří například test minimálního rozměru izolační mezery, minimální rozteče vrtání či minimálního rozměru spoje. Po úspěšně provedené DRC kontrole finální verze desky

plošných spojů následovala její výroba. Výrobu desky jsem svěřil společnosti Gatema s.r.o. Po výrobě se deska plošných spojů osadila součástkami. Osazení provedl ing. Karel Radkovský, za což mu touto cestou velice děkuji. Vzhled primárních vrstev desky plošného spoje je uveden v příloze na konci této práce.

8 Softwarové řešení

Tato kapitola se zaměřuje na popis softwarového řešení modulu rozhraní Ethernet. Nejprve podává obecnou představu dané problematiky a rozděluje ji na menší podproblémy realizované formou modulů založených na vrstveném modelu. Dále definuje obecná rozhraní pro komunikační a síťová zařízení, na kterých je postavena podstatná část softwarového řešení. Po této definici popisuje konkrétní moduly komunikačních zařízení, síťových zařízení a síťových rozhraní. Předposlední část této kapitoly se zaměřuje na funkční moduly, které zastupují zásadní postavení v rámci řešení dané problematiky. V poslední části kapitoly je popsán způsob implementace softwarového řešení.

8.1 Koncepce

Celý problém softwarového řešení rozdělíme na menší podproblémy a zaměříme se na řešení každého podproblému zvlášť. Pod pojmem podproblém si můžeme představit modul, tudíž softwarové řešení je založeno na modulární výstavbě programu.

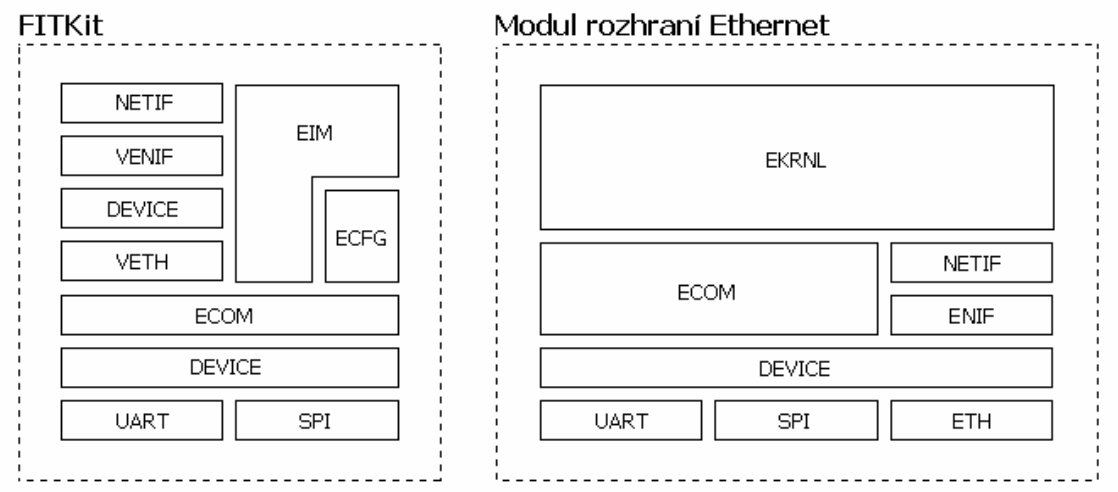
Nyní je otázkou, z jakých typů modulů bude program složen. Odpověď je dána úlohou programu, která je závislá na lokaci. Pokud se zaměříme na program umístěný v modulu rozhraní Ethernet, tak jeho úlohou bude:

- Příjem a vysílání linkových rámců typu Ethernet.
- Transformace linkových rámců.
- Řízení datové komunikace s FITkitem.
- Zpracování požadavků konfigurace.
- Spravování svého stavového prostoru.

Ze souhrnu těchto úloh vyplývá, že programové schéma v modulu rozhraní Ethernet bude tvořeno následujícími moduly: řadičem Ethernet komunikace, řadiči datové komunikace mezi modulem rozhraní Ethernet a FITkitem, komunikátorem mezi modulem rozhraní Ethernet a FITkitem, řídícím jádrem, rozhraním zařízení (řadičů) a síťovým rozhraním. Budeme-li se bavit o programu umístěném na FITkitu, tak jeho úlohou bude:

- Příjem a vysílání linkových rámců typu EIM.
- Spravování datové komunikace s modulem rozhraní Ethernet.
- Generování požadavků konfigurace.
- Poskytnutí speciálních funkcí ovládání modulu rozhraní Ethernet.

Z výčtu těchto úloh vyplývá, že toto programové schéma bude tvořeno následujícími moduly: virtuálním řadičem Ethernet komunikace, řadiči datové komunikace mezi modulem rozhraní Ethernet a FITkitem, komunikátorem mezi modulem rozhraní Ethernet a FITkitem, konfigurátorem modulu rozhraní Ethernet, speciálním modulem pro ovládání modulu rozhraní Ethernet, rozhraním zařízení (řadičů) a síťovým rozhraním. Následující obrázek zobrazuje programové schéma rozdělené dle lokace modulů.



Obrázek 13 - Programové schéma softwarového řešení

Lokace modulu rozhraní Ethernet je tvořena spodní vrstvou reprezentující moduly komunikačních řadičů UART, SPI a ETH. Tato vrstva má funkci procesu linkové komunikace. Bezprostředně nad touto vrstvou se nachází modul rozhraní zařízení DEVICE, který zajišťuje jednotné rozhraní k řadičům. Dalšími moduly jsou NETIF a ENIF plnící funkci síťového rozhraní. Modul EKRNL tvoří jádro modulu rozhraní Ethernet. Jádro řeší stavový prostor a řídí běh modulu rozhraní Ethernet skládající se z procesu vyřizování síťových požadavků a požadavků konfigurace. Posledním modulem je komunikátor ECOM, který zajišťuje pomocí specifického protokolu komunikaci mezi modulem rozhraní Ethernet a FITkitem.

Lokace FITkitu je též tvořena spodní vrstvou reprezentující moduly komunikačních řadičů UART a SPI zajišťující linkovou komunikaci. Přístup k těmto řadičům je opět postaven na rozhraní zařízení DEVICE. Přítomnost síťového rozhraní vyžaduje moduly NETIF a VENIF, ale tyto moduly jsou postaveny na řadiči síťového zařízení typu Ethernet, který není na FITkitu k dispozici. Tento problém řeší virtuální síťový řadič VETH, který je přístupný pomocí rozhraní zařízení DEVICE. Tento virtuální síťový řadič je vlastně obálkou nad komunikátorem ECOM, který zajišťuje komunikaci mezi FITkitem a modulem rozhraní Ethernet. Modul ECFG představuje konfigurátor modulu rozhraní Ethernet. Posledním modulem je softwarový modul EIM zapouzdřující speciální funkce pro ovládání a konfigurování modulu rozhraní Ethernet. Schéma a funkce jednotlivých modulů budou detailněji popsány v následujících podkapitolách.

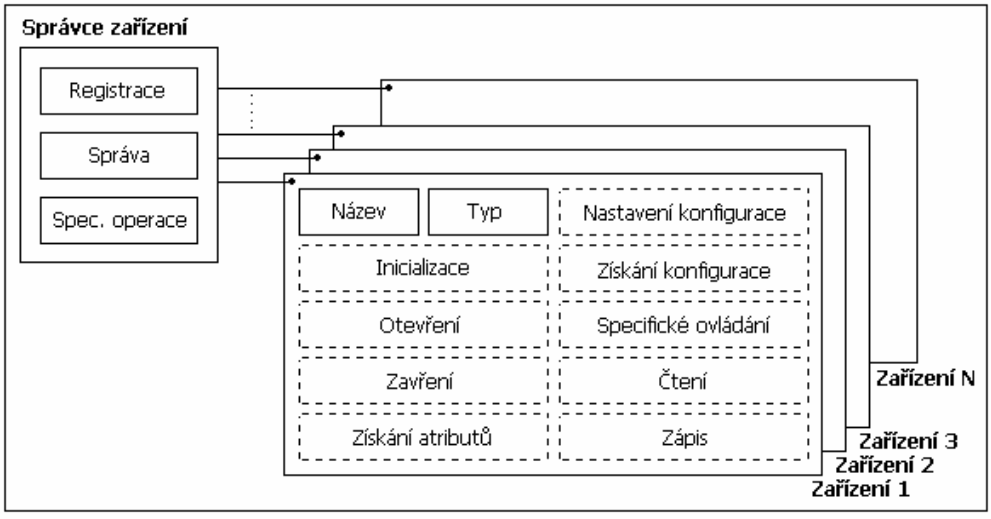
8.2 Definice rozhraní

Úkolem obecného rozhraní je vytvořit obálku nad specifickým typem entity a tak zakrýt její implementační detaily. Potom není problémem implementovat více entit stejného typu zapouzdřující obecné rozhraní. K těmto entitám pak lze přistupovat pomocí speciálního identifikátoru, který se předá správci obecného rozhraní, čímž vzniká značná universálnost modulu implementující obecné rozhraní. Velkou výhodou tohoto řešení je také zajištění stále stejného přístupu k dané entitě i za předpokladu, že se změní její implementační detaily, což zvyšuje celkovou přenositelnost obecného modulu implementující obecné rozhraní. V našem řešení je využito celkem dvou obecných rozhraní. Prvním je rozhraní zařízení, druhým síťové rozhraní.

8.2.1 Rozhraní zařízení

Zařízení je struktura, která nabízí atributy a škálu metod související s funkcí zařízení. Mezi atributy patří název zařízení a jeho typ. Typ zařízení záleží na typu přístupu zařízení k datům, který může být proudový, blokový nebo paketový. Zařízení potom označujeme jako sériové, blokové nebo síťové. Metody zařízení zapouzdřené ve struktuře slouží k inicializaci zařízení, otevření zařízení, získání atributů zařízení, nastavení konfigurace zařízení, získání konfigurace zařízení, specifickému ovládání zařízení, čtení ze zařízení a zápisu do zařízení.

Prvkem, který realizuje rozhraní zařízení je modul nazývající se správce zařízení (DEVICE), který mimo zprostředkování metod zařízení disponuje registrací zařízení, správou seznamu registrovaných zařízení a speciálními operacemi jako je například nalezení zařízení dle jména. Následující obrázek představuje logické schéma správce zařízení.

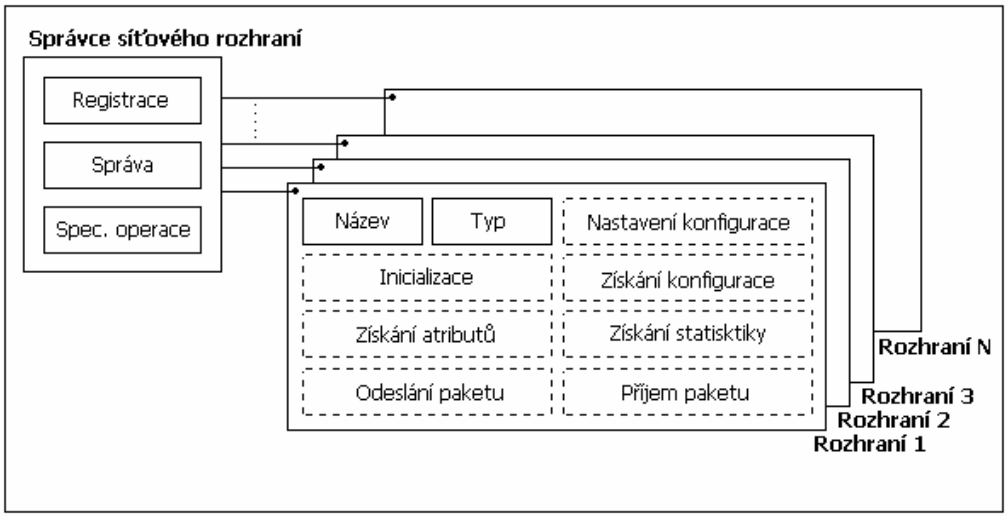


Obrázek 14 : Schéma správce zařízení

8.2.2 Síťové rozhraní

Síťové rozhraní je struktura, která nabízí atributy a množinu metod související se síťovým rozhraní. Mezi atributy patří název síťového rozhraní a jeho typ. Typ síťového rozhraní je závislý na typu sítě, kterým může být Ethernet, PPP, LOOP a další. Metody síťového rozhraní zapouzdřené ve struktuře slouží k inicializaci síťového rozhraní zařízení, získání atributů síťového rozhraní, nastavení konfigurace síťového rozhraní, získání konfigurace síťového rozhraní, získání statistiky síťového rozhraní, příjmu síťového paketu a odeslání síťového paketu. Podoba konfigurace síťového rozhraní je na rozdíl od konfigurace zařízení pevně stanovena již na úrovni rozhraní. Tato konfigurace je složena z internetové adresy, masky sítě, internetové adresy brány a hodnoty maximální přenosové jednotky sítě uváděné v bajtech. Statistika síťového rozhraní zaznamenává specifické hodnoty průtoku dat přes síťové rozhraní. Konkrétně jde o počty odeslaných nebo přijatých síťových paketů s podporou zaznamenávání chyb.

Prvkem, který realizuje síťové rozhraní je modul nazývající se správce síťového rozhraní (NETIF), který mimo zprostředkování metod síťového rozhraní disponuje registrací síťového rozhraní, správou seznamu registrovaných síťových rozhraní a speciálními operacemi jako je například nalezení síťového rozhraní dle jména či internetové adresy. Následující obrázek představuje logické schéma síťového rozhraní.



Obrázek 15 : Schéma správce síťového rozhraní

8.3 Implementovaná zařízení

V minulé kapitole jsme si popsali obecná rozhraní, mezi které patřilo také rozhraní zařízení. Nyní následuje část, kde si uvedeme seznam konkrétních zařízení, které je třeba naimplementovat pro správnou funkci modulu rozhraní Ethernet. Seznam konkrétních zařízení je tedy závislý na funkcích,

které by měla zařízení realizovat. První takovou funkcí je zajištění fyzické komunikace mezi FITkitem a modulem rozhraní Ethernet. Tuto funkci nám zprostředkují sériová zařízení UART a SPI. Další funkcí je zabezpečení síťové komunikace mezi FITkitem a modulem rozhraní Ethernet, o což se postará zařízení VETH. Poslední funkcí, kterou realizuje zařízení ETH je přístup k síti typu Ethernet. Podrobnější popis jednotlivých zařízení je uveden v následujících podkapitolách.

8.3.1 Zařízení UART a SPI

Sériové zařízení UART i SPI slouží k sériové datové komunikaci. Jelikož se jedná o zařízení přistupuje se k nim pomocí rozhraní zařízení, díky čemuž lze využívat implicitních metod tohoto rozhraní. Zařízení je pak možné pomocí těchto metod:

- Inicializovat.
- Připojit a spustit nebo odpojit a zastavit.
- Nakonfigurovat nebo získat konfiguraci těchto parametrů.
  - UART: počet datových bitů, typ parity, počet stopbitů, rychlost datové komunikace
  - SPI: fázi hodin, polaritu hodin, rychlost datové komunikace
- Zapsat nebo přečíst data.

Sériové zařízení UART respektive SPI je ve své podstatě řadič bloku standardní sériové komunikace respektive řadič bloku SPI sériové komunikace, které jsou přítomny na mikrokontroléru. Vnitřní mechanismus těchto řadičů je postaven na přerušení, které se generuje ve dvou případech. Prvním případem je příjem datového bytu, který se vzápětí uloží do ukládací paměti příjmu, odkud jej lze pomocí metody čtení vyčíst. Díky této vlastnosti se řadič vypořádá bez problému i s delší časovou prodlevou než nastane vyčítání přijatých dat. Druhým případem generování přerušení je vyprázdnění vysílacího registru při odesílání dat z ukládací paměti vysílání. Tento způsob naopak značně urychluje odesílání, protože jsou data při odesílání pouze zapsána do ukládací paměti, aniž by se čekalo na jejich celkové odeslání.

8.3.2 Zařízení ETH

Síťové zařízení ETH slouží k odesílání i přijímání linkových rámců ze sítě postavené na technologii Ethernet. Opět se jedná o zařízení a my tedy můžeme využít implicitních metod rozhraní zařízení, které jsou víceméně shodné s uvedenými metodami výše popisovaných zařízení. Jediný rozdíl se nachází v metodě pro nastavení nebo získání konfigurace, která je nyní tvořena pouze jedním parametrem a to hodnotou hardwarové adresy známé pod zkratkou MAC adresa.

Ve své podstatě jde o řadič bloku Ethernet síťové komunikace, který je umístěn na mikrokontroléru. Po inicializaci tohoto bloku, datových popisovačů a obvodu PHY je veškerý datový

tok spojený s odesíláním nebo příjímáním rámců indikován pomocí speciálních vlajek registrů. Data příjmu nebo odesílání jsou pak uložena na speciálním místě v paměti, které je určeno nastavením datových popisovačů. Pomocí rozhraní zařízení je potom možné z této paměti číst nebo do ní zapisovat data, která chceme přijmout nebo odeslat. Na závěr stojí za zmínku, že veškerý přesun dat mezi blokem Ethernet síťové komunikace a pamětí má na starosti DMA řadič. Takže jsou data přenášeny transparentně bez zásahu procesoru, který by se zbytečně touto operací zatěžoval.

8.3.3 Zařízení VETH

Síťové zařízení VETH je ve své podstatě virtuálním Ethernet síťovým zařízením. Za virtuální jej označujeme proto, protože není ve skutečnosti ve vrstvě řadičů a neuskutečňuje tak fyzickou síťovou komunikaci. Namísto toho využívá k zasílání i příjmu síťových dat komunikátoru modulu rozhraní Ethernet, který bude popsán v dalších podkapitolách. Pokud to budeme brát pohledu zařízení jde samozřejmě o klasické zařízení, tudíž pro něj existuje rozhraní, které poskytuje implicitní metody s významem identickým zařízení ETH. Přítomnost virtuálního síťového zařízení v systému je dána požadavkem síťového rozhraní na FITkitu.

8.4 Implementovaná síťová rozhraní

Síťová rozhraní jsou rozšířením síťových zařízení o konfigurovatelné lokální síťové informace, síťovou statistiku a implicitní metody. Podoba lokálních síťových informací, síťové statistiky i implicitních metod již byla popsána v kapitole charakterizující síťové rozhraní. Pokud jsou síťové zařízení rozšířena o síťové rozhraní znamená to tedy, že je počet síťových rozhraní v našem systému roven počtu síťových zařízení. Z čehož vyplývá, že jsou celkem dvě a to konkrétně síťové rozhraní s názvem ENIF a VENIF.

8.4.1 Síťové rozhraní ENIF

Síťové rozhraní ENIF je typu Ethernet. Hlavní funkcí tohoto rozhraní je poskytnutí metod pro zasílání a příjímání síťových paketů ze sítě pomocí síťového zařízení ETH. Rozhraní navíc udržuje lokální síťové informace jako je internetová adresa, maska sítě, adresa brány a hodnota maximální přenosové jednotky v bajtech. Aplikační vrstvě je po registraci tohoto rozhraní vrácen identifikátor, se kterým lze bez problému přistupovat a komunikovat po síti kdekoli v aplikaci.

8.4.2 Síťové rozhraní VENIF

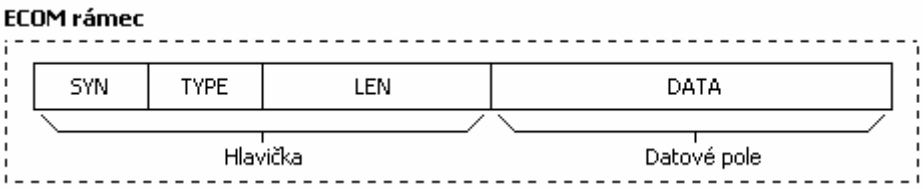
Síťové rozhraní VENIF má prakticky totožné vlastnosti se síťovým rozhraním ENIF, ale na rozdíl od něj je virtuální, protože využívá virtuálního síťového zařízení VETH. Další odlišností je skutečnost, že je jeho hlavním úkolem poskytnutí obecného síťového rozhraní platformě FITkit.

8.5 Implementované funkční moduly

Hlavní úlohou funkčních modulů je za pomoci nižších vrstev, které jsou tvořeny rozhraním zařízení nebo síťovým rozhraním zajistit správnou činnost modulu rozhraní Ethernet. Mezi tyto úlohy patří zajištění protokolové komunikace mezi FITkitem a modulem rozhraní Ethernet, umožnění konfigurace modulu rozhraní Ethernet, podpora jádra spravující stavový prostor a korektní funkci modulu rozhraní Ethernet a konečně zabezpečení přístupu k modulu rozhraní Ethernet aplikační vrstvě. Následující kapitoly popisují tyto moduly a dávají obecný přehled o jejich struktuře a funkci.

8.5.1 Modul ECOM

Modul ECOM představuje komunikátor, pomocí kterého mohou FITkit a modul rozhraní Ethernet vzájemně komunikovat. Samotná komunikace probíhá pomocí zasílání linkového rámce. Formát tohoto rámce je tvořen hlavičkou a datovou částí. Hlavička nese synchronizační a sémantické informace rámce. Datová část nese data, která mají význam daný typem ve hlavičce. Přesný formát linkového rámce je znázorněn v následujícím obrázku.

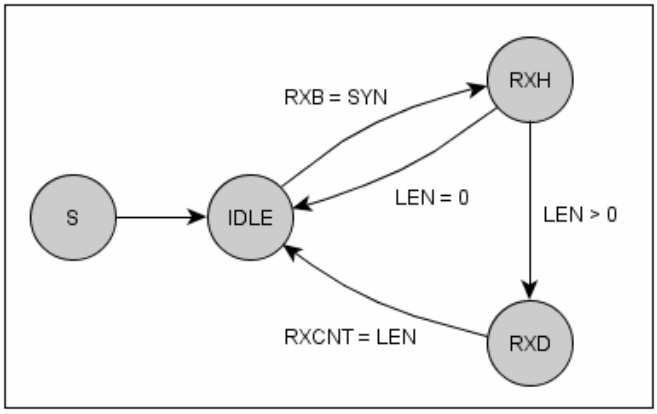


Obrázek 16 : Formát ECOM rámce

Jednotlivá pole výše uvedeného linkového rámce mají samozřejmě svůj specifický význam, který hraje významnou roli v průběhu komunikace mezi FITkitem a modulem rozhraní Ethernet. Tento význam je uveden v následujícím seznamu polí tvořící rámec.

- SYN: synchronizační hodnota 55 hexadecimálně (velikost 1B).
- TYPE: typ linkového rámce (velikost 1B).
  - DTA: datový rámec.
  - CFG: konfigurační rámec.
  - ACK: potvrzovací rámec.
  - OPEN: rámec otevírající komunikaci.
  - CLOSE: rámec zavírající komunikaci.
  - ENCM: rámec aktivující konfigurační mód.
  - RECM: rámec deaktivující konfigurační mód.
- LEN: délka datového pole v bajtech (velikost 2B).
- DATA: datové pole (velikost záleží na hodnotě délky datového pole).

V modulu jsou přítomny dvě metody, které slouží pro odesílání nebo příjímání linkových rámců. Metodě pro odesílání rámce se předávají vstupní parametry značící typ rámce, délku dat a ukazatel na tyto data. Její úloha je tedy jednoduchá, musí složit z těchto parametrů rámec a odeslat ho pomocí přiřazeného rozhraní zařízení. Metoda pro příjem rámce provádí obrácený postup. To znamená, že musí pomocí rozhraní zařízení složit příchozí rámec a informovat o přijatém rámci vyšší vrstvu. Příjem rámce je založen na stavovém automatu tvořeného čtyřmi stavy. První stav je startovní, ze kterého se po startu sytému přejde do stavu čekání na synchronizační bajt. Po příjmu synchronizačního bytu přecházíme do stavu sestavovaní hlavičky rámce. Po jejím složení mohou nastat dva případy závislé na délce datového pole. Pokud je délka datového pole nulová jde o služební rámec a můžeme tedy informovat vyšší vrstvu o přijatém rámci, přičemž opět přecházíme do stavu čekání na synchronizační bajt nového rámce. V případě nenulové délky datového pole musíme přejít do stavu příjímání datového pole. Teprve po příjmu celého datového pole můžeme informovat vyšší vrstvu o přijatém rámci a pak přejít do stavu čekání na synchronizační bajt nového rámce. Následující obrázek znázorňuje podobu stavového automatu formou diagramu.



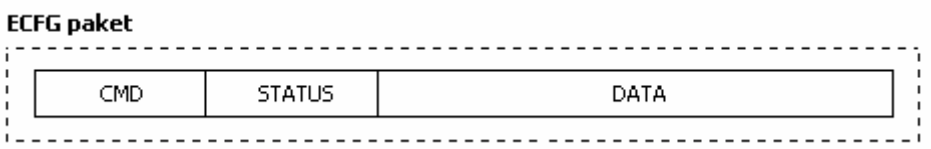
Obrázek 17 : Stavový automat příjmu rámce

8.5.2 Modul ECFG

Modul ECFG implementuje veškeré prostředky pro konfiguraci modulu rozhraní Ethernet. Tento modul se tedy stává konfigurátorem modulu rozhraní Ethernet. Provádění konfigurace je realizováno zasílání konfiguračních rámců pomocí komunikátoru modulu rozhraní Ethernet. Tyto konfigurační rámce zapouzdřují data, které představují konfigurační paket. Konfigurační paket je opět tvořen několika poli. Prvním polem je typ konfiguračního příkazu, druhým polem je status provedení konfiguračního příkazu a posledním polem je datové pole. Typ konfiguračního příkazu určuje konkrétní konfigurační příkaz a je tedy využit ke správnému výběru konfigurace. Status provedení konfiguračního příkazu obsahuje status proběhlé konfigurace, což je víceméně spojeno s informací korektně provedené konfigurace s odpovídajícími konfiguračními daty. Z toho tedy vyplývá

skutečnost, že je toto pole platné pouze ve směru z modulu rozhraní Ethernet. Posledním polem je datové pole pevné velikosti, které je využito u příkazů pracující s konfiguračními daty.

Konfigurace tedy probíhá následujícím způsobem: FITkit zašle konfigurační rámec se zapouzdřeným konfiguračním paketem modulu rozhraní Ethernet a dále čeká na odpověď. Modul rozhraní Ethernet tento paket analyzuje a provede konfiguraci. Po jejím provedení v zápětí odpoví identickým konfiguračním paketem s nastaveným statusem a případně i konfiguračními daty, pokud je konfigurační příkaz vyžadoval. FITkit tento paket přijme a po analýze statusu provedení konfigurace převezme případná konfigurační data. Následující obrázek přibližuje podobu formátu konfiguračního paketu.



Obrázek 18 : Formát konfiguračního paketu

Konfigurátor dále musí brát na zřetel okolnost, která souvisí s operačními módy modulu rozhraní Ethernet. Jejich přesný popis bude uveden v následující kapitole, prozatím je pro nás důležitý fakt, že lze konfiguraci provádět pouze za předpokladu aktivovaného konfiguračního módu v modulu rozhraní Ethernet. Poslední funkcí konfigurátoru je tedy přítomnost metody pro aktivaci nebo deaktivaci konfiguračního módu. Tato metoda zasílá modulu rozhraní Ethernet patřičný speciální služební rámec a čeká na jeho potvrzení. Až po potvrzení aktivace konfiguračního módu je zaručeno, že je modul rozhraní Ethernet připraven zpracovávat konfigurační příkazy. Obdobně až po potvrzení deaktivace konfiguračního módu je zaručeno, že je modul rozhraní Ethernet schopen zpracovávat síťová data.

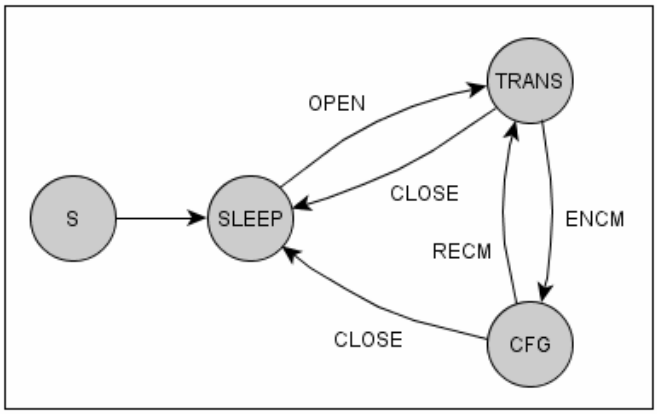
8.5.3 Modul EKRNL

Modul EKRNL tvoří jádro spravující běh modulu rozhraní Ethernet. Toto jádro je řízeno pomocí stavového automatu, který je znázorněn na obrázku 19. Po inicializaci je jádro ve stavu spánku. V tomto stavu jádro očekává služební rámec, který otevře komunikační spojení mezi FITkitem a modulem rozhraní Ethernet. Do té doby je jádro nečinné a jakýkoliv pokus o konfiguraci nebo datový přenos ignoruje. V okamžiku přijetí služebního rámce otevírající komunikační spojení jádro načte výchozí konfiguraci modulu a přejde do stavu transparentního přenosu dat neboli datového módu. Ve stavu transparentního přenosu dat je hlavní úlohou jádra přebírání linkových rámců typu Ethernet od síťového rozhraní a jejich odesílání pomocí komunikátoru platformě FITkit formou datových rámců. To samozřejmě platí i obráceně. Platforma FITkit může pomocí svého síťového rozhraní posílat modulu rozhraní Ethernet linkové rámce typu Ethernet. Vrstvy pod síťovým

rozhraním platformy FITkit se postarají o řádné zabalení linkového rámce typu Ethernet do formy datového rámce, který je poslán pomocí komunikátoru jádru v modulu rozhraní Ethernet. Jádro datový rámec rozloží a odešle čistý rámec typu Ethernet pomocí svého síťového rozhraní do lokální sítě typu Ethernet.

Ze stavu transparentního přenosu dat je možné přejít pomocí speciálního služebního rámce do stavu konfigurace, čímž se modul rozhraní Ethernet uvede do konfiguračního módu. V tomto módu lze pomocí konfigurátoru provádět jeho konfiguraci. Jelikož je tento mód výhradně určen pro zpracování konfigurace, nelze v něm provádět datové přenosy až do doby jeho opuštění, což se opět uskuteční zasláním speciálního služebního rámce.

Otevření komunikačního spojení mezi modulem rozhraní Ethernet a platformou FITkit je akceptováno pouze v případě, když je jádro ve stavu spánku. Uzavření komunikačního spojení mezi modulem rozhraní Ethernet a platformou FITkit je akceptováno pouze v případě, když je jádro ve stavu transparentního přenosu nebo ve stavu zpracování konfigurace.



Obrázek 19 : Stavový automat jádra

8.5.4 Modul EIM

Posledním z řady funkčních modulu je modul EIM. Tento modul poskytuje plnohodnotnou přístupovou vrstvu aplikacím, které chtějí na platformě FITkit využívat funkcí modulu rozhraní Ethernet. Modul k provádění konkrétních funkcí využívá metod komunikátoru a konfigurátoru, čímž zakrývá zbytečně složitý vnitřní mechanismus spojený s jejich ovládáním. Uživateli je tak poskytnut přístup k modulu rozhraní Ethernet pomocí vhodně zvolené množiny metod. Tato množina obsahuje metodu pro inicializaci přístupové vrstvy, ve které se zaregistruje síťové rozhraní VENIF, síťové zařízení VETH a výchozí komunikační zařízení UART. Dalšími metodami této množiny jsou metody pro otevření a zavření komunikačního spojení o což se postará komunikátor pomocí odeslání speciálního služebního rámce. Posledními metodami spadající do této množiny jsou metody vykonávající funkční, komunikační i síťovou konfiguraci. Po úvaze nad přístupem k modulu rozhraní Ethernet by se mohlo někomu zdát, že v této množině metod chybějí metody pro odesílání a přijímání

síťových dat. Skutečně metody pro odesílání a přijímání síťových dat v tomto modulu nejsou přítomny, ale to není žádnou chybou, nýbrž k tomuto účelu je v systému k dispozici síťové rozhraní VENIF, které se o tuto úlohu bez problému postará.

8.6 Popis implementace

Softwarové řešení bylo implementováno pomocí programovacího jazyka C. Pouze ve zvláštních případech, které vyžadovaly zápis přímo v instrukcích bylo využito instrukční sady daného procesoru. Samotná vývojová prostředí, která byla využita při implementaci modulů se lišila jejich lokací.

Moduly na platformě FITkit byly překládány volně šiřitelným GNU překladačem “mspgcc“ pro typ procesoru MPS společnosti Texas Instrument. Cílový kód byl po překladu nahrán do procesoru přes sériový port pomocí speciálního nástroje, který poskytuje opět společnost Texas Instrument. Z toho tedy vyplývá, že nebylo možné tímto postupem ladit program pomocí ladiče, ale pouze formou ladicích výpisu přes sériový port.

Moduly implementované na modulu rozhraní Ethernet byly vyvíjeny v prostření Embedded Workbench pro procesory s ARM jádry od švédské společnosti IAR. Toto vývojové prostředí poskytuje komfortní ladící nástroje, se kterými není problémem vyvíjet vestavěné systémy rychle, bezchybně a efektivně. Ladění v tomto systému je postaveno na ladiči JLINK podporující standardizovaný protokol JTAG. Ladič JLINK tak podporuje značné množství různých typů procesoru s jádry ARM od mnoha výrobců. Takže byl ideální volbou pro vývoj softwaru na mikrokontroléru STR912 s jádrem ARM od společnosti ST Microelectronics.

Samotná implementace a ladění modulů probíhalo od nejnižších vrstev týkající se komunikace a rozhraní až po nejvyšší vrstvy představující funkci modulu rozhraní Ethernet. Nakonec byla napsána nejvyšší vrstva aplikace, která demonstruje funkčnost celého řešení pomocí zpracování ICMP paketů. Vývoj této demonstrační aplikace je popsán v následující závěrečné kapitole.

9 Demonstrační aplikace

Tato kapitola popisuje tvorbu demonstrační aplikace na platformě FITkit využívající ke své funkci modul rozhraní Ethernet. Kapitola začíná výběrem vhodné aplikační domény, která by výstižně demonstrovala funkčnost modulu rozhraní Ethernet. Pak následuje část zabývající se analýzou a návrhem řešení demonstrační aplikace, ve které je provedena detailní analýza a návrh nových síťových modulů spadající do vybrané aplikační domény. Po té je popsána samotná realizace a implementace demonstrační aplikace na platformě FITkit. Nakonec je věnována pozornost systému konfigurace, který je reprezentován sadou definovaných konfiguračních příkazů.

9.1 Výběr aplikační domény

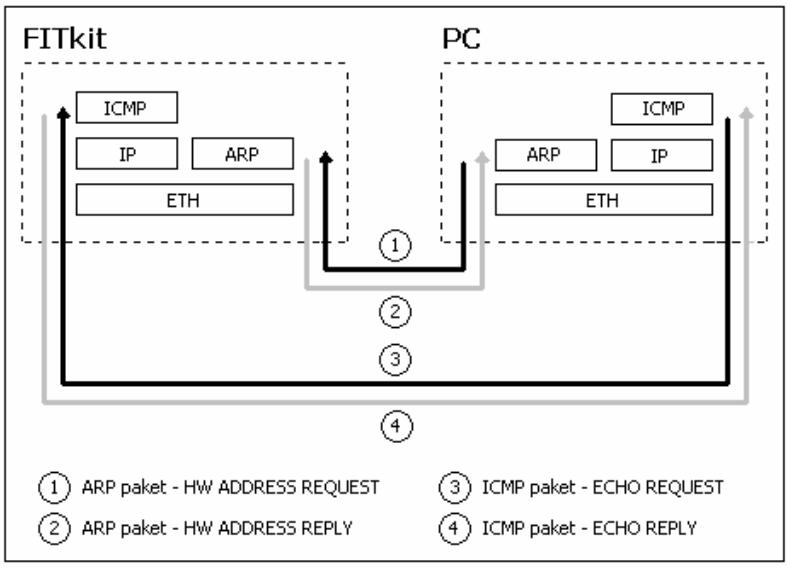
Pro vhodnou demonstraci je nutné zvolit vhodnou aplikační doménu, která by zřetelně a výstižně předvedla funkčnost modulu rozhraní Ethernet pro platformu FITkit. Při výběru této domény se zaměříme na její jednoduchost a univerzalitu. Ideálním kandidátem je aplikační doména zaměřující se na zpracování ICMP paketů. Pro účel demonstrace je tato úloha zcela postačující, není nijak obzvlášť složitá a její hlavní výhodou je právě vysoká univerzalita vyplývající ze zpracování ICMP paketů, které lze generovat prakticky na jakémkoli počítači vybaveném náležitým operačním systémem. Funkcí demonstrační aplikace je tedy odpovídání na ICMP pakety generované standardními síťovými programy většiny operačních systémů.

9.2 Analýza a návrh řešení

Před samotnou realizací aplikace je třeba provést analýzu týkající se síťových modulů, které jsou nezbytné pro správnou funkci demonstrační aplikace. Tyto síťové moduly budou tvořit opět vrstvový model, jehož úlohou bude přijímání a odesílání ICMP paketů v síti typu Ethernet.

Jelikož komunikujeme právě po síti typu Ethernet, musí náš první síťový modul realizovat ARP protokol. Ve skutečnosti není třeba implementovat celý ARP protokol, ale pouze jeho část, která je zodpovědná za vyřizování požadavků o hardwarovou adresu zařízení. Další síťový modul představuje internetovou vrstvu, pomocí které lze pakety adresovat v internetu. ICMP paket je do této vrstvy zapouzdřen a může být tímto způsobem adresován specifickému zařízení ve stejné nebo vzdálené síti. Poslední síťový modul zpracovává samotné ICMP pakety. Pod pojmem zpracování ICMP paketu je přesněji myšleno přijmutí ICMP paketu nesoucí zprávu ECHO REQUEST, na kterou se vzápětí odpoví ICMP paketem nesoucí zprávu ECHO REPLY.

Všechny tyto síťové moduly jsou samozřejmě implementovány na platformě FITkit. Síťové moduly tedy využívají ke komunikaci modul rozhraní Ethernet a tak jej plnohodnotně testují.

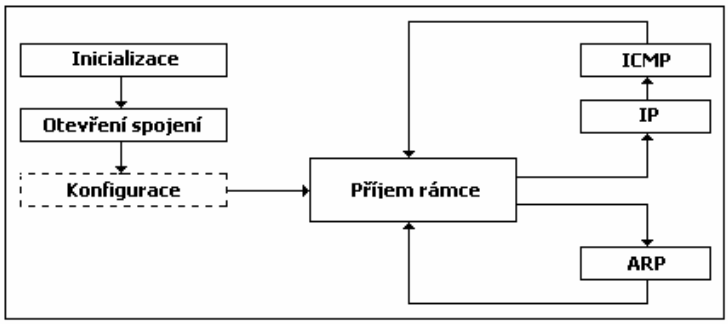


Obrázek 20 : Průběh komunikace při zpracování prvního ICMP paketu

Výše uvedený obrázek znázorňuje průběh komunikace při zpracování prvního ICMP paketu. Jak vidíme nejdříve je odeslán všesměrový požadavek o hardwarovou adresu, na kterou FITkit okamžitě reaguje odpovědí nesoucí jeho hardwarovou adresou. Poté teprve následuje odeslání ICMP paketu se zprávou ECHO REQUEST. FITkit na tuto zprávu odpovídá ICMP paketem se zprávou ECHO REPLY, čímž končí průběh zpracování prvního ICMP paketu.

9.3 Realizace na platformě FITkit

Realizace demonstrační aplikace na platformě FITkit prochází několika fázemi. V první fázi se zavolá metoda inicializace modulu EIM, ve které proběhne zaregistrování komunikačních zařízení, síťového zařízení a síťového rozhraní. Ve druhé fázi se otevře komunikační spojení. Ve třetí fázi se provede síťová konfigurace, která může být v případě použití výchozích hodnot vynechána. V poslední fázi se čeká na příchozí rámec ze síťového rozhraní. Při události přijetí rámce je ověřen jeho typ a rámec je dále předán buď modulu ARP nebo modulu IP potažmo modulu ICMP, který jej zpracuje.



Obrázek 21 : Schéma realizace aplikace

9.4 Podpora konfigurace

Na tomto místě bych rád uvedl, že demonstrační aplikace navíc umožňuje otestování funkčnosti systému konfigurace pomocí sériové konzole (např. PComm Terminal). K tomu slouží předem definované konfigurační příkazy, jejichž seznam je následující:

- Nastavení / Získání typu datového módu.
- Nastavení / Získání typu komunikačního zařízení.
- Nastavení / Získání rychlosti komunikačního zařízení.
- Nastavení / Získání síťové adresy.
- Nastavení / Získání masky sítě.
- Nastavení / Získání adresy brány.
- Nastavení / Získání hardwarové adresy.

Detailní popis významu a syntaxe těchto příkazů je uveden na přiloženém datovém médiu v sekci, která popisuje způsob postup realizace demonstrační aplikace na platformě FITkit.

Závěr

Cílem této práce bylo navrhnout a realizovat síťové rozhraní k platformě FITkit formou rozšiřujícího modulu. K dosažení tohoto cíle bylo nutné analyzovat problematiku spojenou se síťovým rozhraním a navrhnout patřičné hardwarové a softwarové řešení rozšiřujícího modulu.

Hardwarové řešení skýtalo tvorbu elektrického schéma, tvorbu plošného spoje, výrobu plošného spoje, osazení součástkami a samotné oživení. Všechny etapy hardwarového řešení byly postupně uskutečněny a výsledkem tohoto snažení se stal funkční modul rozhraní Ethernet pro platformu FITkit. Dále následovala fáze softwarové řešení, ve které bylo zapotřebí na základě navrženého programového schéma provést implementaci veškerých modulů určených pro správnou funkci modulu rozhraní Ethernet. Po dokončení této implementace a otestování dokončených modulů následoval vývoj demonstrační aplikace, která měla za úkol pomocí našeho síťového rozhraní prověřit zpracování linkových rámců na platformě FITkit. Demonstrační aplikace byla po návrhu a následné implementaci plně funkční, čímž se jen potvrdilo, že cíl této práce, jejíchž záměrem bylo realizovat síťové rozhraní pro platformu FITkit, byl splněn.

V původní myšlence návrhu síťového rozhraní pro platformu FITkit byly navrženy dva datové módy. Stávající transparentní datový mód měl být rozšířen o další datový mód, ve kterém by se aktivovala vrstva přítomna na modulu rozhraní Ethernet. Tato vrstva by zajišťovala podporu transportních protokolů TCP a UDP. Toto rozšíření by umožňovalo pohybovat se přímo na nejvyšší aplikační vrstvě síťového modelu a pomocí tzv. zásuvek (sockets) vytvářet velmi jednoduchým a efektivním způsobem síťové aplikace, aniž by se programátor musel starat o nižší síťové vrstvy, které nemají se samotným významem aplikace nic společného. Bohužel tento datový mód nebyl z časových důvodů dokončen a je tedy vhodným kandidátem na budoucí rozšíření stávajícího řešení síťového rozhraní pro platformu FITkit.

Na samotný závěr mohu uvést, že nyní existuje patřičné síťové rozhraní pro platformu FITkit, které je tvořeno speciálním rozšiřujícím modulem. Tento modul po připojení poskytuje dostatečné funkce pro zpracování linkových rámců na síti typu Ethernet a tak umožňuje studentům implementovat síťové aplikace přímo na platformě FITkit.

Literatura

[1] Noergaard, T. *Embedded Systems Architecture*, Elsevier, 2005, 657 s.

[2] Axelson, J. *Embedded Ethernet and Internet Complete*, Lakeview Research, 2003, 497 s.

[3] Catsoulis, J. *Designing Embedded Hardware*, O'Reilly, 2005, 318 s.

[4] Spurgeon, J. *Ethernet: The Definitive Guide*, O'Reilly, 2003, 512s.

[5] *Network Protocols Handbook*, Javvin Technologies, 2005, 340s.

[6] *Stránky o platformě FITkit*: *Úvod* [online]. Fakulta informačních technologií, 2008.
URL: <http://merlin.fit.vutbr.cz/FITkit/>.

[7] *Stránky o platformě FITkit*: *FITkit tutorial* [online]. Fakulta informačních technologií, 2008.
URL: <http://merlin.fit.vutbr.cz/FITkit/docs/navody/20060210a.html >.

[8] *STR91xF product catalog*: *Features* [online]. ST Home – Microcontrollers, 2008.
URL: <http://www.st.com/mcu/devicedocs-STR912FAW44-101.html>.

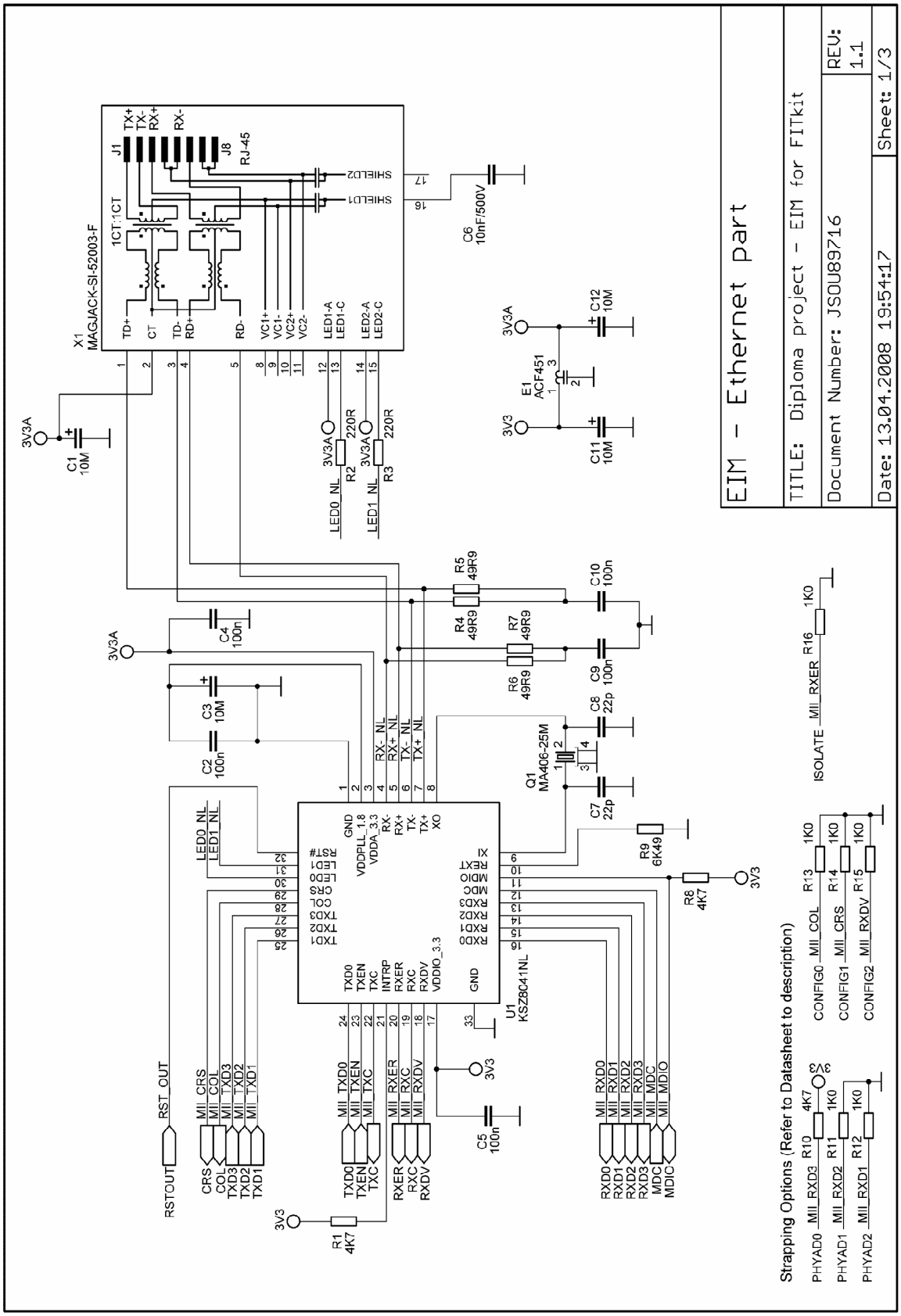
[9] *KSZ8041NL product brief*: *General description* [online]. Micrel Home – PHYs, 2008.
URL: <http://www.micrel.com/_PDF/Ethernet/ksz8041nl_pb.pdf >.

[10] *MagJack Connector Modules*: *Datasheet* [online]. Belfuse Home – Connectors, 2008.
URL: <http://www.belfuse.com/ProdPage-Magjack.asp?pPart=SI-52003>.

[11] *M45PE80 datasheet*: *Features* [online]. Numonyx Home – NOR Memories, 2008.
URL: <http://www.numonyx.com/Documents/Datasheets/M45PE80.pdf>.

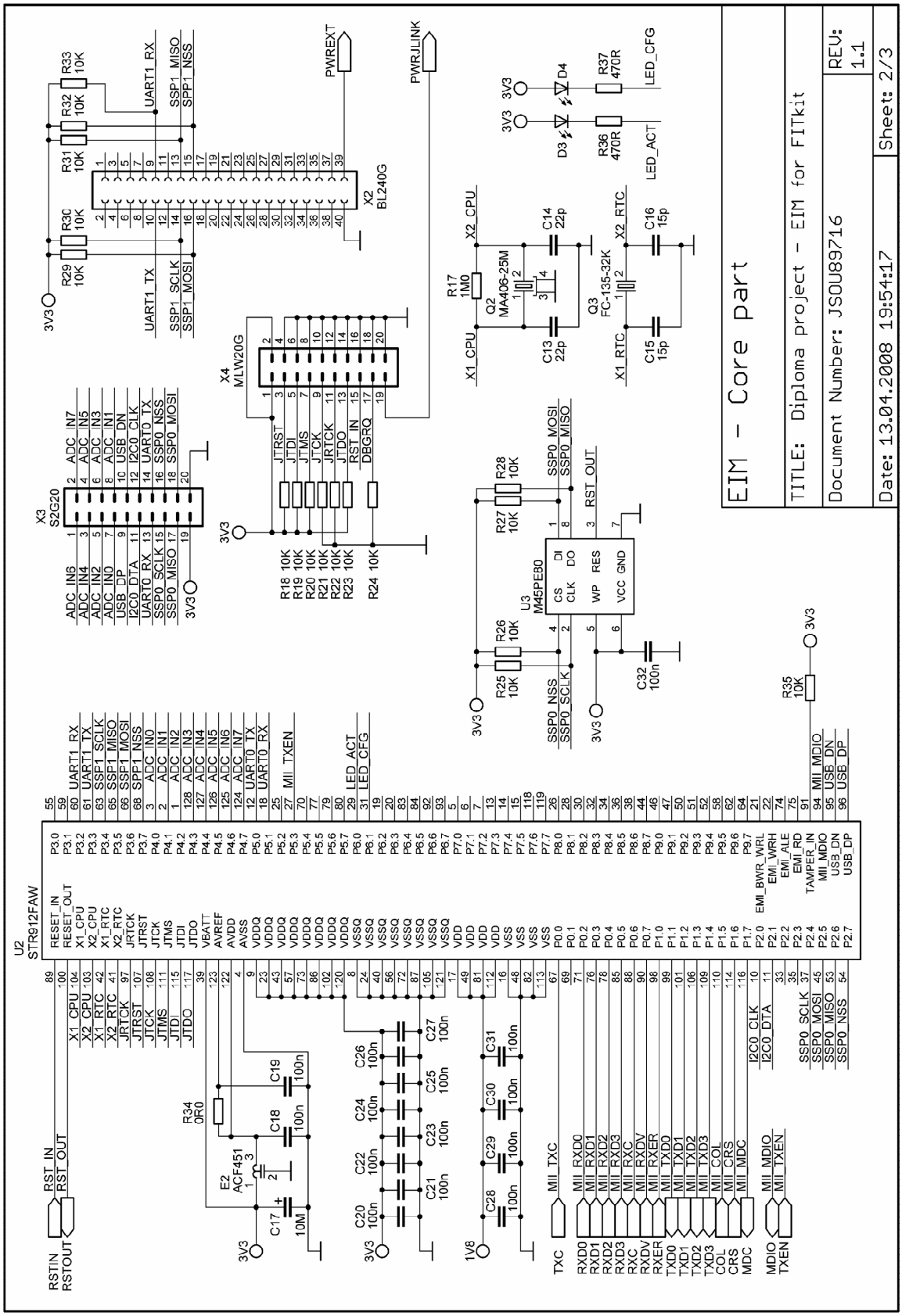
[12] *TPS70151 product description*: *Features* [online]. TI Home – Power Management, 2008.
URL: <http://focus.ti.com/docs/prod/folders/print/tps70151.html>.

Příloha A



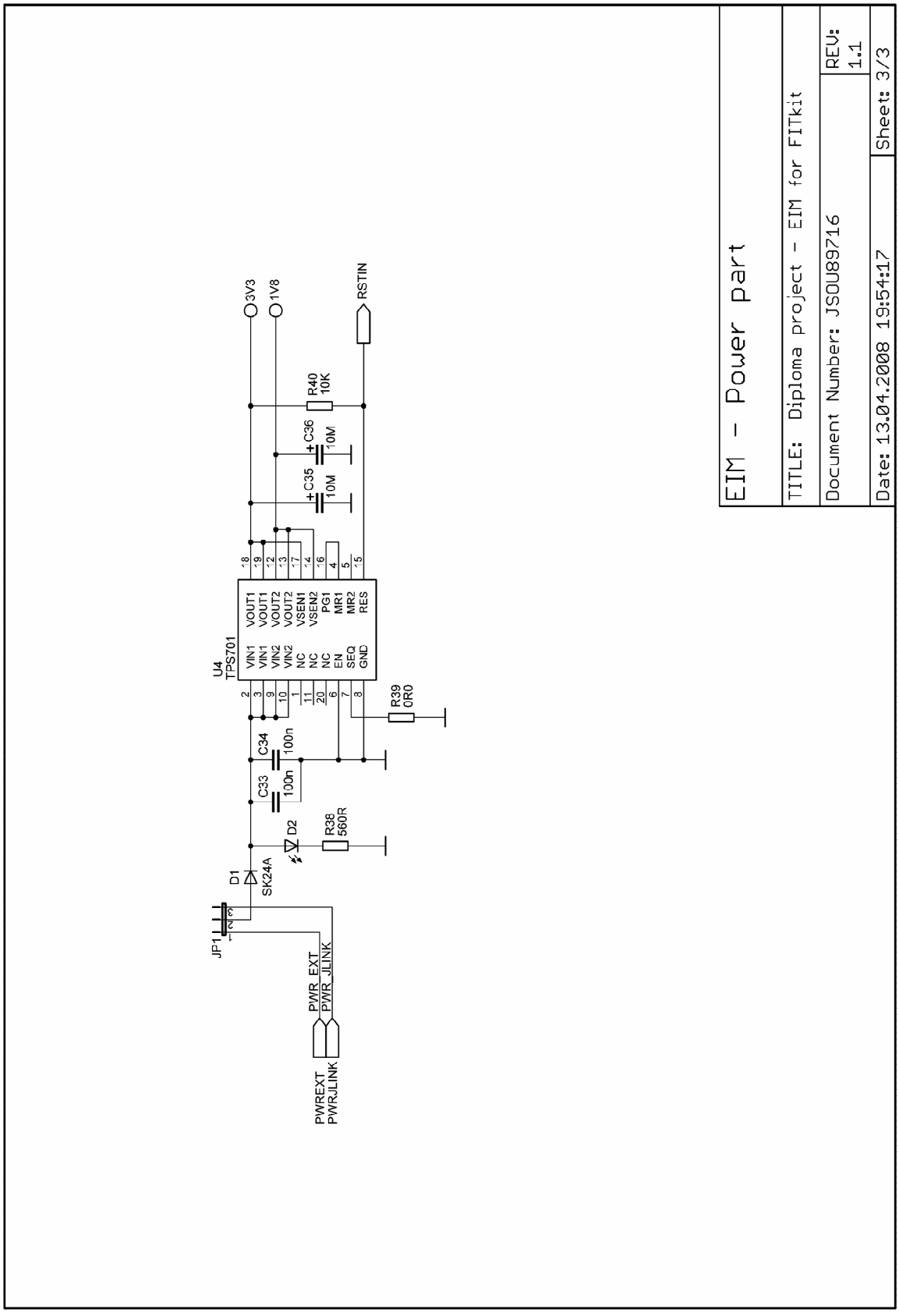
Obrázek 22 : Schéma bloku fyzické vrstvy Ethernetu

Příloha B



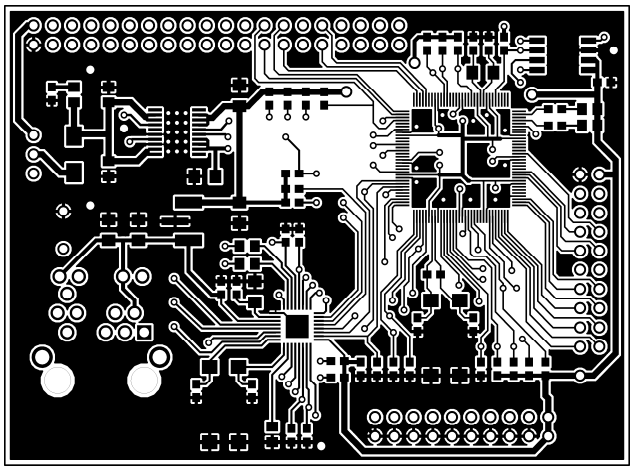
Obrázek 23 : Schéma centrálního bloku

Příloha C

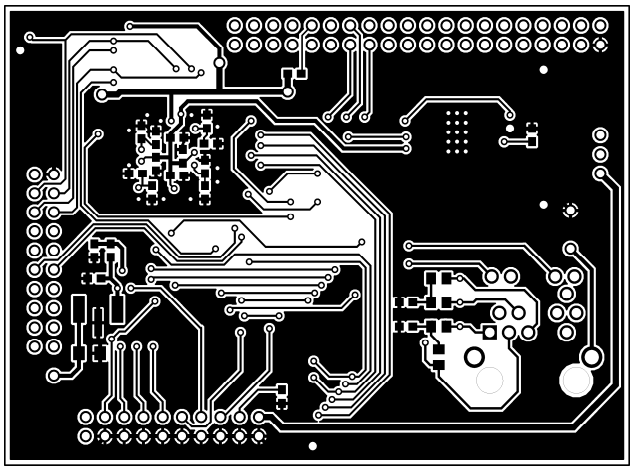


Obrázek 24 : Schéma napájecího bloku

Příloha D

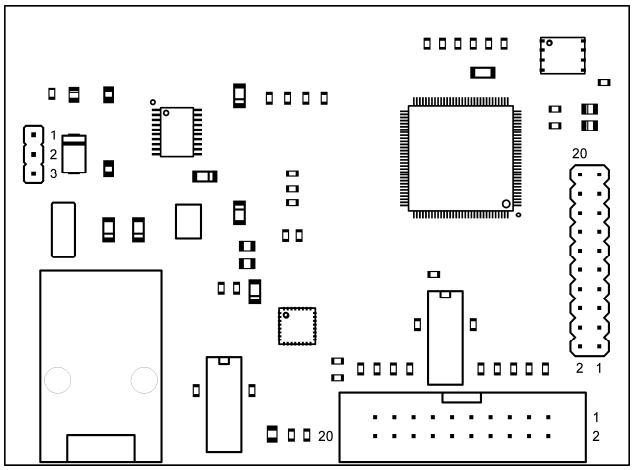


Obrázek 25 : Vrchní vrstva plošného spoje

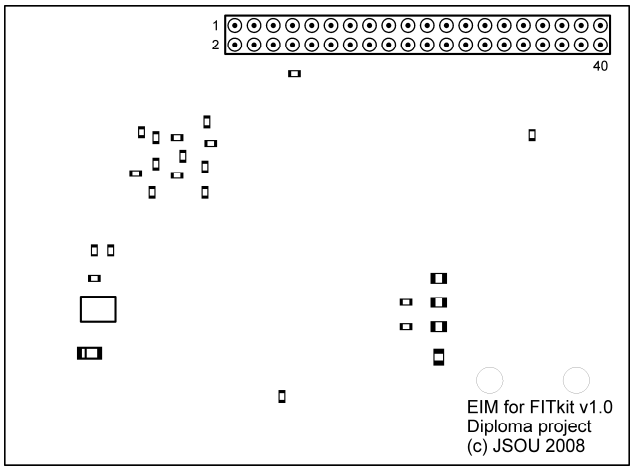


Obrázek 26 : Spodní vrstva plošného spoje

Příloha E

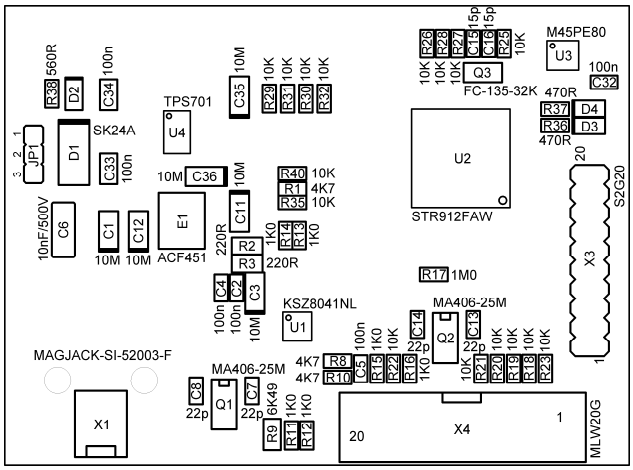


Obrázek 27 : Vrchní vrstva rozmístění součástek a potisku desky

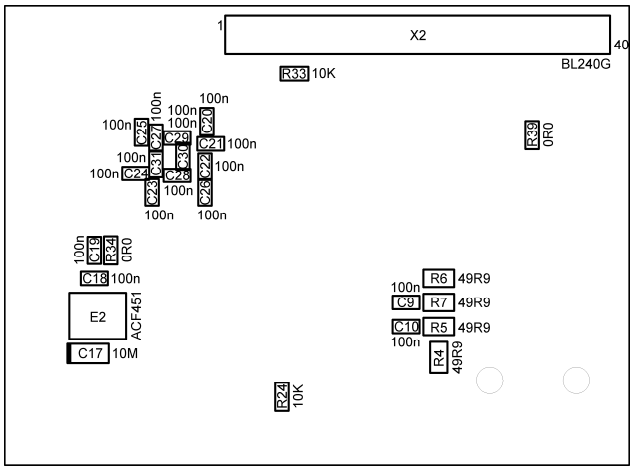


Obrázek 28 : Spodní vrstva rozmístění součástek a potisku desky

Příloha F



Obrázek 29 : Vrchní vrstva popisu součástek



Obrázek 30 : Spodní vrstva popisu součástek