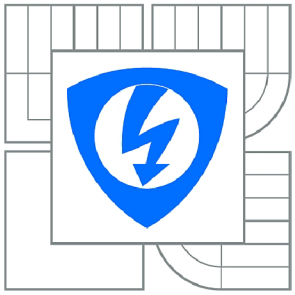
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION

IMPLEMENTACE OVLADAČE I2S AUDIO V SYSTÉMU FREESCALE MQX RTOS

IMPLEMENTATION OF THE I2S DRIVER INTO THE FREESCALE MQX RTOS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

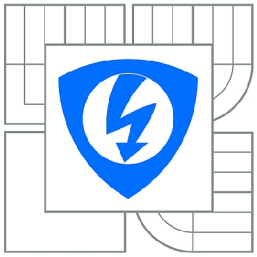
AUTOR PRÁCE — Bc. KAREL MOŽNÝ

AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL KUČERA, Ph.D.

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Karel Možný	***ID:***	109700
Ročník:	2	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Implementace ovladače I2S Audio v systému Freescale MQX RTOS

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte I/O subsystémy a ovladače různých typů sériových rozhraní v operačním systému MQX. Seznamte se Freescale Tower konceptem vývojových desek. Navrhněte audio desky pro Tower Concept, propojující I2S signály s vhodným I2S audio DAC a výstupem na sluchátkový konektor nebo reproduktory.
Navrhněte softwarový ovladač I2S sběrnice v systému Freescale MQX: Definujte programové rozhraní ovladače a jeho implementaci pro zvolenou procesorovou platformu.
Demonstrujte funkčnost ovladače vytvořením aplikace pro přehrávání zvukových souborů typu WAV, MP3 nebo podobných.
Pokud to časové možnosti dovolí, získejte důležité kvalitativní a kvantitativní parametry navrženého systému.

DOPORUČENÁ LITERATURA:

[1] Freescale MQX Real-Time Operating System User's Guide [online]. Rev. 6. Freescale Semiconductor, Inc., 4/2011. Dostupné z WWW: http://www.freescale.com/webapp/sps/site/overview.jsp?code=MQXSWDW&tid=m32MQX.
[2] Barr, M., Massa, A.: Programming Embedded Systems: With C and GNU Development Tools. O'Reilly Media; 2 edition (October 1, 2006). ISBN-13: 978-0596009830.

Termín zadání:	6.2.2012	***Termín odevzdání:***	21.5.2012

Vedoucí práce: Ing. Pavel Kučera, Ph.D.
Konzultanti diplomové práce:

doc. Ing. Václav Jirsík, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Abstrakt

Tato práce se zabývá návrhem a vývojem ovladače sběrnice I^2S pro operační systém reálného času MQX běžícího na procesoru řady ColdFire V4. Dále je v práci také uvedeno vytvořené zapojení a desky plošných spojů sloužící k otestování funkčnosti a vlastností vytvořeného ovladače. Tyto desky používají jako zdroj signálu rozhraní S/PDIF, výstupem je analogový signál určený k poslechu na integrovaném sluchátkovém zesilovači. V závěru práce je popsána ukázková aplikace demonstrující funkci ovladače na vyvinutém hardware.

Klíčová slova

Digitální zvuk, I^2S, S/PDIF, D/A převodník, Sluchátkový Zesilovač, MQX, RTOS, Freescale Tower, ColdFire V4

Abstract

This thesis deals with design and development of a I^2S module driver for real-time operating system MQX running on a ColdFire V4 architecture based processor. Further there are presented circuit diagrams and PCBs created for testing of functions and properties of the driver. Signal input is a digital audio in form of S/PDIF interface, output is an analogue signal, that can be listened by the means of a headphone amplifier. The conclusion describes a sample application demonstrating function of the driver on developed hardware.

Keywords

Digital Audio, I^2S, S/PDIF, DAC, Headphone Amplifier, MQX, RTOS, Freescale Tower, ColdFire V4

Bibliografická citace:

MOŽNÝ, K. *Implementace ovladače I2S Audio v systému Freescale MQX RTOS.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 76 s. Vedoucí diplomové práce Ing. Pavel Kučera, Ph.D..

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Implementace ovladače I2S Audio v systému Freescale MQX RTOS jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.
Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: 21. května 2012 ……

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Pavlu Kučerovi, Ph.D za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce. Dále také děkuji odborným konzultantům práce Michalu Hanákovi, Lukáši Petrovi, Juraji Vančovi a celé firmě Freescale Polovodiče Česká Republika s.r.o za jejich odborné rady a poskytnutí vývojových nástrojů, díky kterým bylo možno tuto práci realizovat.

V Brně dne: 21. května 2012 ……

podpis autora

OBSAH

SEZNAM POUŽITÝCH ZKRATEK

AES3 – profesionální ekvivalent spotřebitelského rozhraní S/PDIF.

ADC – z angl. Analog-to-Digital Converter. Jedná se o obvod realizující převod analogového signálu na digitální.

API – z angl. Application Programming Interface. Jde o rozhraní pro programování aplikací (obvykle funkce, případně makra).

BDM – z angl. Background Debugging Mode. Rozhraní poskytující možnost debuggování v embedded systémech, které se vyskytuje zejména v produktech společnosti Freescale.

BNC – Bajonet Neill-Concelman. Typ konektoru pro koaxiální kabel vyráběn pro zakončení kabelů o impedanci 50 nebo 75 Ω.

CAN – z angl. Controller Area Network. Je sběrnice používaná nejčastěji pro vnitřní komunikační síť senzorů a funkčních jednotek v automobilech.

CAU – z angl. Cryptology Acceleration Unit. Jednotka mikroprocesoru sloužící pro urychlení kryptografických výpočtů.

DAC – z angl. Digital-to-Analog Converter. Digitálně-analogový převodník. Obvod sloužící pro převod digitálního signálu na analogový.

DDR – z angl. double-data-rate. Typ dynamických pamětí. Vyžaduje periodickou obnovu dat.

DMA – z angl. Direct Memory Access. Způsob přímého přenosu dat mezi operační pamětí a vstupně výstupními zařízeními.

DPS – Deska plošných spojů.

DSP – z angl. Digital Signal Processor. Specializovaný mikroprocesor s architekturou optimalizovanou pro algoritmy používané při zpracování digitálně reprezentovaných signálů.

DTS – z angl. Digital Theatre System. Digitální vícekanálový formát prostorového ozvučení se ztrátovou kompresí.

EMAC – z angl. Enhanced Multiply and Accumulate. Jednotka pro zrychlení výpočtu operace násobení a součtu v jednom kroku.

FIFO – z angl. First In First Out. Buffer typu fronta.

FIR – z angl. Finite Impulse Response. Filtr jehož impulzní odezva je časově omezena.

ISR – z angl. Interrupt Service Routine. Funkce sloužící k obsluze přerušení.

LSB – z angl. Least Significant Bit. Nejméně významný bit v datovém slovu.

MFB – z angl. Multi Feedback Topology. Topologie vícenásobné zpětné vazby používaná u filtrů, která využívá přidání dvou pólů do přenosové funkce.

MMU – z angl. Memory Managment Unit. Součást CPU zajišťující překlad virtuální adresy na fyzickou, ochranu paměti, kontrolu cache, arbitraci sběrnice atd.

MSB – z angl. Most Significant Bit. Nejvýznamnější bit v datovém slovu, tzn. ten který má nejvyšší váhu při rozkladu pomocí z-adického rozvoje.

PCM – z angl. Puls-code Modulation. Pulzně kódová modulace. Modulační metoda použitá pro převod analogového signálu na digitální.

PWM – z angl. Pulse Width Modulation. Pulzně šířková modulace. Diskrétní modulace pro přenos analogového signálu pomocí dvouhodnotového signálu.

RCA – Typ konektoru, který také někdy bývá nazýván CINCH používaný k připojení audio nebo video signálů.

RPC – z angl. Remote Procedural Calls. Vzdálené volání procedur, služba ke spouštění úloh na vzdálených procesorech.

RTOS – z angl. Real-Time Operating System. Operační systém reálného času.

S/PDIF – z angl. Sony/Philips digital Interconnect Format . Typ rozhraní používaného pro přenos digitálního zvuku.

SCK – z angl. Serial Clock. Vodič určený na sběrnici I^2S pro přenos hodinového kmitočtu.

SD – z angl. Serial Data. Označení vodiče sběrnice I^2S po kterém jsou vysílána nebo přijímána data.

SDHC – z angl. Secure Digital high Capacity. Typ paměťových karet používaný v přenosných zařízeních jako jsou fotoaparáty a mobilní telefony

SNR – z angl. Signal-to-Noise Ratio. Číslo obvykle v dB vyjadřující odstup šumu od užitečného signálu.

SPI – z angl. Serial Peripheral Interface. Sériové periferní rozhraní. Používá se pro komunikaci mezi řídícími mikroprocesory a ostatními integrovanými obvody.

SRAM – z anglického Static Random Access Memory. Statická polovodičová paměť, která k uchování svých dat nepotřebuje jejich periodickou obnovu.

SSI – z angl. Serial Synchronous Interface. Sériové synchronní rozhraní, které slouží pro připojení libovolné periferie a je možné jej použít jako rozhraní I^2S.

UART – z angl. Universal Asynchronous Receiver and Transmitter. Zařízení pro sériovou synchronní komunikaci. Jeho pomocí lze komunikovat přes rozhraní RS-232 a podobná.

ULPI – z angl. UTLI+ Low Pin Interface. Nízkonákladové rozhraní s minimálním počtem pinů sloužící k připojení fyzické vrstvy sběrnice USB.

WS – z angl. Word Select. Označení vodiče sběrnice I^2S využitého pro synchronizaci dat.

XLR – Audio konektor pro profesionální využití s třemi a více kontakty.

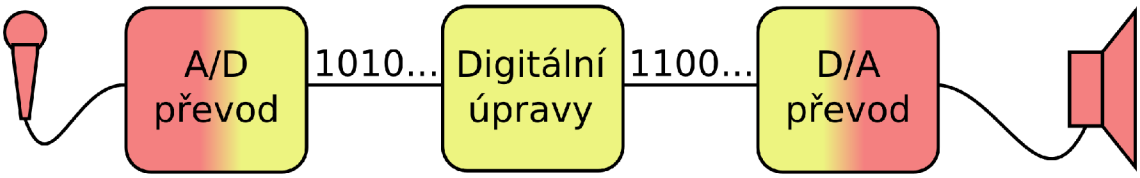
SEZNAM PŘÍLOH

1 ÚVOD

V této práci je uvedeno řešení problému, který se vyskytl při vývoji systému MQX ve firmě Freescale Polovodiče Česká Republika s.r.o. Tímto problémem byl v první řadě neexistující ovladač sběrnice I^2S pro tento operační systém a dále potřeba zapojení a k němu vytvořené DPS na které by tento ovladač bylo možno testovat za pomocí vývojové platformy Freescale Tower.

2 DIGITÁLNÍ ZVUK

Digitální zvuk vzniká převedením spojité reprezentace zvukového signálu do diskrétní podoby. Tento převod začíná převodem spojitého mechanického kmitání na elektrický signál pomocí mikrofonu. Spojitá elektrická reprezentace zvuku je poté přivedena na vstup analogově-digitálního převodníku, který pomocí procesu vzorkování a kvantizace převede analogový signál do jeho digitální podoby. Takový signál je možno dále zpracovávat a upravovat aniž by došlo ke ztrátě jeho kvality. Celý řetězec zpracování zvuku lze vidět na následujícím obrázku:



Obrázek 1: Zjednodušené schéma zpracování zvuku

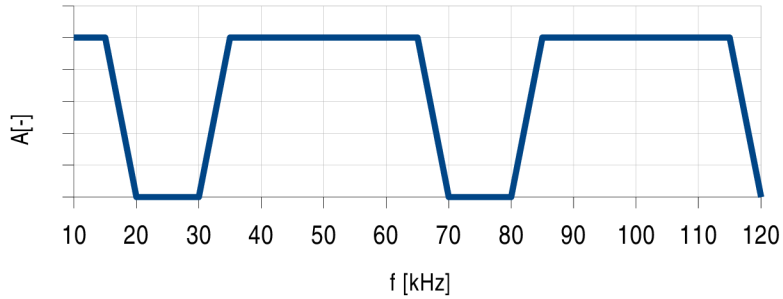
2.1 Vzorkování

Vzorkování je proces, pomocí kterého je spojitý čas t převeden na čas diskrétní tak, že hodnoty signálu jsou ukládány pouze v určitých časových intervalech s přírůstkem $\Delta T=T$ a nikoliv spojitě jako v případě analogového signálu. Tento časový interval je určen vzorkovací frekvencí f_s ($f_s=T^{-1}$), jejíž velikost je určena Nyquistovým vzorkovacím teorémem jako:

$$f_s > 2 \cdot f_{max} \quad (1)$$

kde: f_{max} = maximální frekvence obsažena ve vzorkovaném signálu.

Spektrum navzorkovaného signálu je podobné signálu původnímu s tím rozdílem, že je v navzorkovaném signálu nakopírováno tak, že se jeho kopie (nebo také aliasy) se objevují se středy vycentrovanými na celistvých násobcích vzorkovací frekvence. Například signál se šířkou pásma $B=20\,kHz$, který je vzorkovaný kmitočtem 50 kHz bude mít spektrum podobné následujícímu obrázku:



Obrázek 2: Kopírování spektra ve vzorkovaném signálu

Pokud by šířka pásma vzorkovaného signálu nebyla omezena ($f_{max} > f_s/2$), došlo by k překrytí jednotlivých kopií a tím vzniku aliasingu. Tento jev způsobí zobrazení frekvencí vyšších než $f_s/2$ zpět před polovinu vzorkovací frekvence podle vztahu:

$$f_A = f - \frac{f_s}{2} \quad (2)$$

kde:
f = libovolná frekvence pro kterou platí $f > f_s$.
f_A = frekvence do které proběhne zobrazení.

2.2 Převzorkování

Vzorkování i rekonstrukce navzorkovaného signálu vyžaduje, aby měl vstupní signál striktně omezenou šířku pásma, jinak dojde k aliasingu a tím ztrátě informací. V praxi by díky tomuto požadavku bylo třeba navrhnout pro dosažení dostatečné kvality zvuku analogový filtr osmého až desátého řádu k omezení pásma na 20 kHz a potlačením frekvencí vyšších než 24 kHz o přibližně 80 dB. Filtr takto vysokého řádu je možné navrhnout, avšak k jeho konstrukci by bylo třeba součástek s velmi nízkými tolerancemi. Dalším problémem je nelinearita fáze na vysokých frekvencích a zpoždění signálu, které vedou ke slyšitelnému zkreslení.

Možným řešením tohoto problému je vzorkování vstupu na vyšší frekvenci a tím tedy snížení nároků na vstupní filtry. Nízkofrekvenční filtrování a redukce vzorkovací frekvence pak mohou být provedeny v diskrétní podobě, kde je aplikace těchto operací jednodušší.

Převzorkování probíhá ve dvou krocích: zvyšování vzorkování a digitální filtrace. V průběhu převzorkování je signál $x(n)$ se vzorkovací frekvencí f_s změněn na signál $y(n)$ se zvýšením vzorkováním na frekvenci $k.f_s$ vložením k-1 vzorků s nulovou hodnotou. Tím dojde k rozložení energie každého z původních vstupních vzorků na k nových vzorků a tím i zeslabení každého z nich k-krát. Je tedy třeba tento jev kompenzovat vynásobením hodnot signálu $y(n)$ číslem k. Po zvýšení vzorkovací frekvence je provedena digitální filtrace pomocí interpolace vzorků $y(n)$, čímž vznikne signál $z(n)$. Spektrum tohoto signálu je pak stejné jako spektrum spojitého signálu, avšak opakuje se na frekvencích $k.f_s$.

Pro filtraci signálu je využit filtr typu dolní propust. Impulzní odezvou ideálního filtru typu tohoto typu je funkce $sinc(x)$, která je v čase rozložena na intervalu $(-\infty, \infty)$. Avšak z hlediska implementace je třeba tento interval omezit, čímž vznikne filtr s omezeným počtem prvků (FIR). Použití konečného počtu prvků dojde k oříznutí

impulzu ve filtru a tím ke zvlnění jeho odezvy. Lepším řešením než pouze oříznout impulzní charakteristiku je vytvoření hladkého přechodu tím, že jednotlivé koeficienty ve filtru vynásobíme okénkovou funkcí, jejíž vrchol leží uprostřed impulzu.

2.3 Jitter

Jitter je definován jako časová nestabilita v průběhu vzorkování nebo rekonstrukce analogových signálů a tedy vzniká jak při převodu A/D tak i D/A. Čtení jednotlivých vzorků signálu je prováděno v intervalech, které určují pulzy z oscilátoru. Pokud časová vzdálenost jednotlivých pulzů hodinového kmitočtu není ekvidistantní dojde ke změně vzorkovací/rekonstrukční frekvence u jednotlivých vzorků. Tato chyba v časování vede k uložení nesprávné hodnoty v případě A/D převodu nebo převodu správné hodnoty avšak ve špatný čas v průběhu D/A převodu a tím vnesení šumu a zkreslení do signálu.

2.4 Kvantizace a kvantizační šum

Pro uložení digitálního signálu je třeba, aby každý vzorek byl reprezentován jako posloupnost bitů; hodnotě napětí z nekonečného rozsahu je třeba přiřadit konečnou diskrétní hodnotu. Tato operace je nazývána kvantizace. Například v případě 16bitového formátu můžeme reprezentovat proměnné vstupní napětí pomocí 2^{16} diskrétních úrovní. Je tedy zřejmé, že kvantizace omezuje celkovou kvalitu digitálního zvuku počtem úrovní, do kterých je možné vstupní analogovou veličinu rozdělit. Rozdělení signálu do požadovaných úrovní spočívá v jednoduchém zaokrouhlení vstupní hodnoty na hodnotu nejbližší diskrétní úrovně.

Mějme nyní jednoduchý kvantizátor Q jehož vstupem je signál $x(n)$ a výstupem kvantizovaný signál $y(n)$. Pokud $x(n)$ je náhodná nekorelovaná veličina, pak jeho libovolný vzorek bude ležet se stejnou pravděpodobností v intervalu $(p - q/2)$ jako v intervalu $(p + q/2)$. Kvantizační chyba bude rovna $e(n) = y(n) - x(n)$ a její velikost bude náhodná s rovnoměrným rozložením a maximální hodnotou $q/2$. Tuto chybu je možno změřit a její velikost je vyjádřena pomocí poměru signál/šum (SNR nebo také S/N) a dána vztahem:

$$SNR_{dB} = 20\log_{10}(2^n) \approx 6{,}02 \cdot n \quad (3)$$

kde: n = počet bitů do kterých je veličina kvantizována.

Pro 16bitový rozsah je pak vypočtená teoretická maximální hodnota SNR = 96 dB.

Pokud se však v signálu $x(n)$ objevuje korelace (takovým signálem je například záznam hudby) pak nelze tuto jednoduchou analýzu uplatnit. Chyba se pak stane také korelovanou se signálem a projeví se jako zkreslení namísto širokopásmového šumu.

Uvažujme nyní 16bitový A/D převodník se vzorkovací frekvencí 44,1 kHz. Jeho kvantizační šum bude přibližně 96 dB pod úrovní maximální hodnoty užitného signálu a bude rovnoměrně rozprostřen na frekvencích od stejnosměrného signálu do 22,05 kHz. Pokud však použijeme vyšší vzorkovací kmitočet, výkon šumu zůstane stejný, ale rozprostře se do širšího frekvenčního pásma. Zvýšením vzorkování dvakrát se výskyt šumu rozšíří na frekvenční rozsah 0 – 44,1 kHz. Jestliže jako filtr využijeme diskrétní dolní propust s mezní frekvencí 22,05 kHz zajistíme tak snížení výkonu kvantizačního šumu v požadovaném pásmu na polovinu a zvýšení SNR o 3 dB aniž by došlo k ovlivnění zvukového signálu. Tento efekt lze dále rozšířit a každým zdvojením vzorkovacího kmitočtu snížit šum o další 3 dB.

Použitím stejného principu lze například dosáhnout toho, že použijeme-li 15bitový A/D převodník, jehož vzorkovací frekvence bude 176,4 kHz dosáhneme na frekvenčním pásmu zvuku stejného výkonu jako s použitím převodníku 16bitového se vzorkováním 44,1 kHz. To je umožněno tím, že nárůst kvantizačního šumu je vyrovnán pomocí zlepšení poměru signál/šum. Obdobně lze zlepšit parametry v případě D/A převodu.

Chyba kvantizace se projevuje jako šum při vysokých úrovních signálu, ale pokud se úroveň signálu přiblíží úrovni nejméně významného bitu (LSB), stane se kvantizační chyba signálem a je slyšitelná na výstupu. Tento efekt je v praxi odstraňován použitím ditheru.

2.5 Dither

Dither je proces spočívající v přidání nízkoúrovňového analogového šumu *d(n)* ke vstupnímu signálu kvantizátoru *x(n)* tak aby jeho výstup byl při nízkých úrovních náhodný. Tím dojde k odstranění korelace mezi signálem a kvantizační chybou proto, že výstup nyní nebude závislý pouze na vstupu, ale také na náhodné veličině.

Kritickým parametrem použitého šumu *d(n)* je jeho rozložení pravděpodobnosti. To musí být zvoleno tak, aby došlo k efektivnímu narušení korelace mezi kvantizační chybou a vstupem, ale přitom měl signál co nejnižší výkon z důvodu ovlivnění výstupu. Pokud zvolíme rovnoměrné rozložení s nulovou střední hodnotou a amplitudou *q/2* dojde k porušení korelace uvažovaných signálů. Na výstupu se pak objeví šum s výkonem přibližně $q^2/12$ a po přičtení šumu vytvořeného kvantizátorem se celkový výkon šumu bude pohybovat kolem $q^2/6$.

Toto řešení však není úplné, protože vygenerovaný výkon je stále korelován se vstupním signálem. Přidáním dalšího šumového signálu na vstup dojde k odstranění této

závislosti, zároveň se ale také zvětší výkon šumu na hodnotu $q^2/4$. Kvalita zvuku se však zlepší, jelikož chyba ani její výkon nyní nejsou závislé na vstupním signálu.

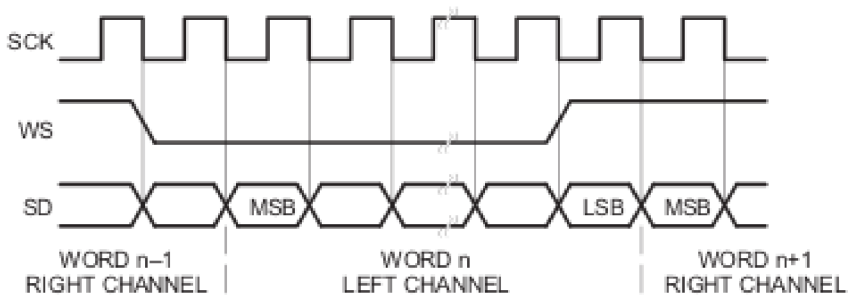
Postup přidávání dalších signálů pro zmenšení korelace je možno opakovat, dochází tím ale k dalšími zarušování signálu a prakticky je ověřeno, že použití dvou šumových veličin dostačuje pro práci se zvukem.

3 SBĚRNICE I²S

S nástupem digitální techniky v oblasti spotřební audio elektroniky, která obsahuje kodeky, DSP, digitálně analogové a analogově digitální převodníky vznikl v této oblasti požadavek na vytvoření standardizované sběrnice, která by umožňovala flexibilní přenášení digitalizovaného zvukového signálu mezi těmito zařízeními. Na základě tohoto požadavku vytvořila firma Philips rozhraní I²S známé také pod názvem Integrated Interchip Sound.

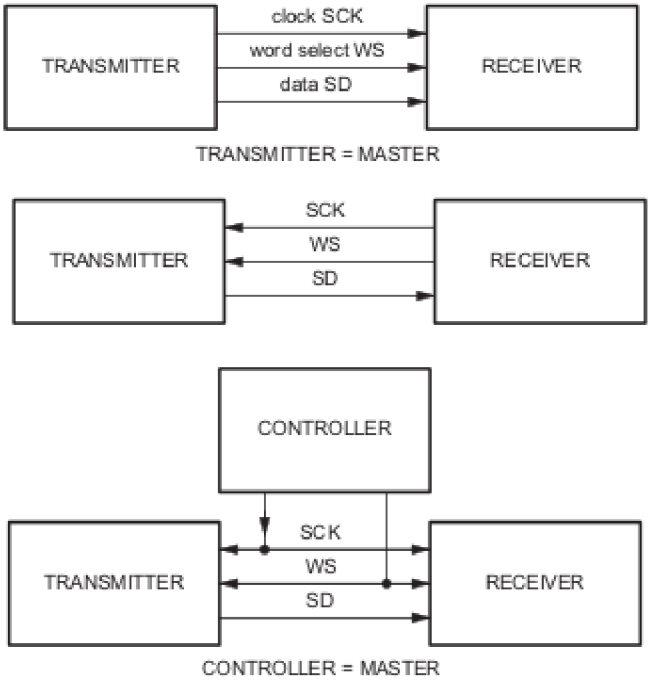
3.1 Základní struktura

Sběrnice přenáší pouze zvuková data. Další signály (např. ovládací a kódovací) jsou vedeny samostatně. Kvůli minimalizaci potřebného počtu vodičů a zjednodušení zapojení je použito pouze třívodičové sériové sběrnice. První z vodičů je využit k přenosu dvou časově multiplexovaných kanálů (signál SD – serial data), druhý slouží k přenosu informace o kanálu (signál WS – word select) a třetí k přenosu hodinového signálu (signál SCK – serial clock).



Obrázek 3: Signály sběrnice I²S[17]

Vzhledem k tomu, že vysílač i přijímač využívají stejného hodinového signálu k přenosu dat, musí vysílač v režimu master generovat všechny tři signály. Ve složitějších systémech s více vysílači a přijímači je však obtížné určit kdo má tyto signály generovat. V takových systémech je možno určit jeden z prvků sběrnice jako tzv. „system master“, který pak generuje hodinový signál a určuje časový sled dat ostatních prvků. Je tedy možné mít zařízení, které se chová jako vysílač – slave i přijímač – master (viz následující obrázek).



Obrázek 4: Příklady uspořádání sběrnice I2S [17]

3.2 Sériová data

Data jsou po sběrnici přesouvána ve dvojkovém doplňku, nejvýznamnější bit (MSB) je přenášen jako první. To umožňuje, aby vysílač i přijímač měli různou délku slova. Vysílač tedy nemusí vědět, kolik bitů je schopen přijímač pojmout a přijímač nemusí mít informaci o bitové délce přenášených dat. Pokud je datové slovo větší než slovo zpracovávané vysílačem, jsou data ořezána na potřebnou velikost pro odeslání. V případě že přijímač přijme více dat než je jeho slovo jsou nadbytečné bity ignorovány. Pokud se situace obrátí a přijímač přijme méně dat než je šířka jeho slova, jsou chybějící bity doplněný nulami. Pozice MSB je tedy pevná, zatímco poloha nejméně významného bitu (LSB) je určena šířkou přenášeného slova. Vysílač vždy posílá MSB následujícího slova jednu periodu hodinového signálu po změně signálu WS. Sériová data z vysílače mohou být synchronizována buď na nástupnou nebo sestupnou hranu hodinového signálu. Avšak v přijímači musí být data hradlovány vždy na nástupné hraně.

3.3 Funkce signálu word select

Signál word select slouží k indikaci přenášeného kanálu. Pokud je WS = 0, pak je

přenášen kanál č.1 – levý, pokud je WS = 1, pak je přenášen kanál č.2 – pravý. Ke změně úrovně signálu může docházet na nástupné i sestupné hraně hodinového signálu, nemusí však být symetrický. V zařízení v režimu slave je tento signál hradlován na nástupné hraně hodinového signálu. Úroveň signálu se mění vždy jednu periodu hodinového signálu před přenosem MSB datového slova. To umožňuje vysílači v režimu slave odvodit synchronní časování sériových dat pro odeslání. Navíc je také umožněno přijímači uložit předchozí datové slovo do paměti a uvolnit místo pro následující data.

4 ROZHRANÍ S/PDIF

Zkratkou S/PDIF (z angl. Sony/Philips digital Interconnect Format) se rozumí digitální rozhraní pro přenos zvukového signálu v zařízeních spotřební elektroniky na krátké vzdálenosti a je založen na profesionálním standartu AES3. Obě tyto rozhraní jsou standardizována normou IEC 60958. Signál je přenášen v metalickém nebo optickém médiu pomocí koaxiálního kabelu s konektory RCA, případně optickým kabelem s konektory TOSLINK.

Využívá se zejména k propojení komponentů systémů domácích kin a jiných zařízeních zpracovávajících signál vysoké věrnosti (Hi-Fi). Pomocí tohoto rozhraní lze přenášet buďto dva kanály nekomprimovaného zvuku ve formátu PCM nebo několik komprimovaných kanálů ve formátech jako je např. Dolby Digital nebo DTS.

4.1 Rozdíly mezi AES3 a S/PDIF

Rozhraní S/PDIF bylo vyvíjeno současně s AES3 a vzniklo z důvodu sjednocení profesionálního a spotřebního rozhraní. S/PDIF je na protokolové vrstvě s AES3 téměř shodné, je však rozšířeno o mechanismus pro zabránění kopírovaní obsahu. Na fyzické vrstvě se obě rozhraní liší především použitými konektory (AES3 využívá konektory XLR a BNC) a kabelem, kde došlo k změně z kroucené dvoulinky o impedanci 110 Ω na koaxiální kabel o impedanci 75 Ω, případně optický kabel. Srovnání obou rozhraní lze vidět v následující tabulce:

	AES3 symetrický	AES3 nesymetrický	S/PDIF
Kabel	110 Ω STP	75 Ω koaxiální	75 Ω koaxiální
Konektor	XLR	BNC	RCA/TOSLINK
Úrovně	2-7 V_{pp}	1-1,2 V_{pp}	0,5-0,6 V_{pp}
Min. vstupní úroveň	0,2 V	0,32 V	0,2 V
Max. vzdálenost	100 m	1000 m	10 m
Modulace	dvoufázová	dvoufázová	dvoufázová
Kódování informací	ASCII	ASCII	SCMS
Bitová hloubka max.	24 bitů	24 bitů	20 bitů (24 volitené)

Tabulka 1: Hlavní rozdíly mezi rozhraním AES3 a S/PDIF

5 SÉRIOVÉ SYNCHRONNÍ ROZHRANÍ (SSI)

SSI je full-duplexní sériové rozhraní, které umožňuje procesoru komunikovat s množstvím sériových zařízení jako jsou: kodeky, DSP, procesory, periferie a audio kodeky, které využívají pro připojení sběrnici I^2S nebo rozhraní standartu Intel AC'97.

5.1 Implementace rozhraní v použitém procesoru

Procesor MCF54418 obsahuje dva moduly SSI, kde každý z nich má odděleny sekce pro příjem a vysílání dat. Tyto oddělené sekce však sdílí vnitřní popřípadě vnější zdroj hodinového kmitočtu a signál pro synchronizaci rámců. Obě tyto sekce mohou pracovat jek v režimu master tak v režimu slave. Pro každý z modulů jsou implementovány dva hardwarové buffery typu FIFO o velikosti 15 krát 32 bitů, což umožňuje existenci dvou nezávislých kanálů pro oba směry přenosu dat. Dále je pomocí programovatelného datového rozhraní umožněno přenášet data v několika formátech (zarovnání dat MSB, LSB a I^2S) a různou bitovou délkou datového slova (8, 10, 12, 16, 18, 20, 22 nebo 24 bitů). Obsažena je také kompletní podpora standartu AC'97 s oddělenou volbou hodinového a synchronizačního kmitočtu.

5.2 Signály rozhraní SSI

Přehled signálů sběrnice a jejich funkce je uveden v následující tabulce:

Název	Funkce	Směr	Pull-up
SSI_CLKIN	Vstup signálu master clock	Vstup	pasivní
SSI_BCLK	Hodinový kmitočet sběrnice	Vstup/Výstup	pasivní
SSI_MCLK	Výstup signálu master clock	Výstup	pasivní
SSI_FS	Synchronizace dat	Vstup/Výstup	pasivní
SSI_RXD	Příjem dat	Vstup	-
SSI_TXD	Vysílání dat	Výstup	pasivní

Tabulka 2: Přehled signálů rozhraní SSI

Signál SSI_CLKIN

Časování modulu SSI může probíhat z vnitřní frekvence procesoru, nebo za pomocí hodinového signálu přivedeného na pin určeného pro vstup signálu SSI_CLKIN. Tento signál je sdílen mezi oběma moduly SSI v procesoru a je multiplexován se signálem SSI_MCLK (nachází se na stejném pinu). Externí zdroj signálu je možno využít jak v

případě vysílání, tak přijímání v módech master i slave. V módu master je využit pro generování signálů SSI_FS a SSI_BCLK pro připojené zařízení. Ve slave módu funguje jako vstup signálu master clock.

Signál SSI_BCLK

Tento vstupně výstupní signál je využívám vysílačem i přijímačem a určuje platnost dat na lince SSI_RXD nebo SSI_TXD. V případě vysílání je tento signál generován vnitřně a odvozen od frekvence signálu SSI_MCLK případně SSI_CLKIN pomocí děličky. V případě, že je procesor v módu slave slouží jako vstup pro signál z externího zdroje.

Signál SSI_MCLK

Jedná se o celistvý n-násobek ($n \in \{128, 256, 384, 512, 768, 1024\}$) signálu pro synchronizaci dat (SSI_FS) a je některými zařízeními jako jsou audio kodeky nebo D/A převodníky využíván jako zdroj řídící frekvence pro vykonávání instrukcí, proto je nazýván jako „master clock“. Někdy bývá v literatuře (např. [4]) označován také jako převzorkovací frekvence.

Signál SSI_FS

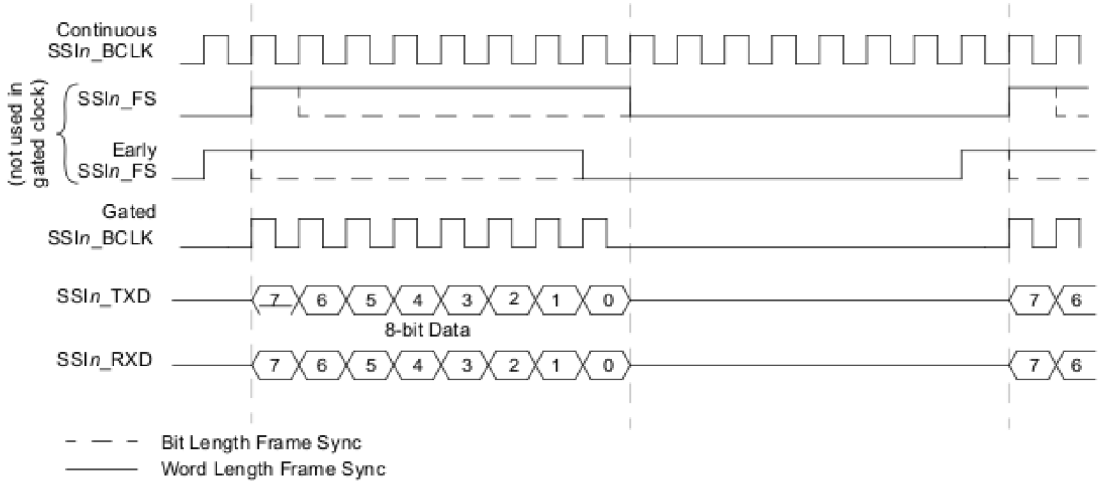
Vstupně výstupní signál SSI_FS slouží vysílači a přijímači k synchronizaci přenášených dat. Signál může mít délku jednoho slova nebo bitu a jeho změna může být vykonávána jeden bit před vysíláním nebo přímo na začátku vysílání dat.

Signál SSI_RXD

Pomocí tohoto signálu jsou procesorem přijímána data do posuvného registru, odkud jsou dále překlápěny do vstupního FIFO bufferu.

Signál SSI_TXD

Za pomocí tohoto signálu vysílač odesílá data protistraně. Tyto data jsou na linku odesílány z výstupního posuvného registru, kam jsou kopírovány z výstupního FIFO bufferu.



Obrázek 5: Signály sběrnice SSI [4]

6 OPERAČNÍ SYSTÉMY REÁLNÉHO ČASU

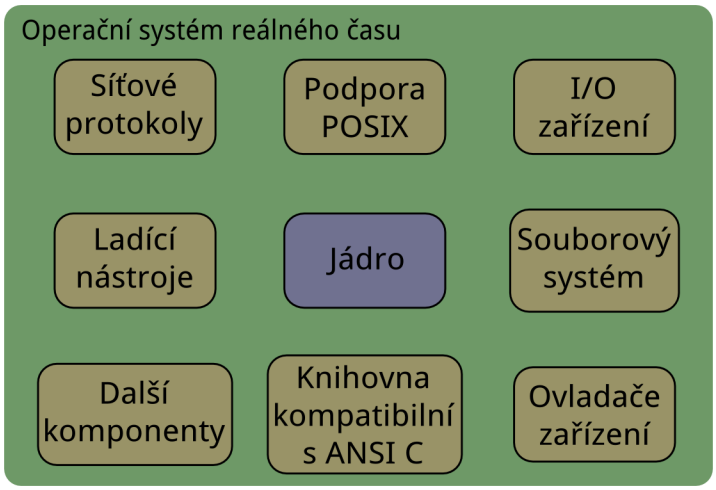
Operační systém je základním programem, který poskytuje uživateli rozhraní mezi hardwarem a ostatním softwarem a jeho základní rolí je správa hardwarových zdrojů podle potřeb aplikací.

Pokud se pak jedná o operační systém reálného času (RTOS) je takový systém vytvořen tak, aby splňoval požadavek na logicky správný výsledek operací v časovém úseku, jehož délka je omezena. Jinými slovy lze také říci, že takový systém je časově deterministický. RTOS lze rozdělit podle náročnosti na plnění časových limitů do tří skupin následovně:

1. *Hard real-time*: tolerance nedodržení časových limitů je u těchto systémů extrémně malá, případně nulová. Jejich nedodržení může způsobit katastrofální selhání systému.
2. *Firm real-time:* nedodržení limitů může způsobit neakceptovatelné snížení kvality.
3. *Soft real-time:* limity mohou být nedodrženy a z jejich porušení se lze zotavit. Snížení kvality je v těchto systémech přijatelné.

6.1 Architektura a klíčové pojmy v RTOS

Architektura RTOS je závislá na jeho použití. Obvykle bývají RTOS snadno škálovatelné, aby splnily požadavky pro co největší rozsah použití pro nejrůznější aplikace. V jednoduchých případech tak může RTOS sestávat pouze z jádra, zatímco ve složitějších mohou být použity různé komponenty, jako jsou síťové protokoly, souborové systémy apod. tak, jak je vidět na následujícím obrázku:



Obrázek 6: Obecná architektura RTOS

Jádro (kernel)

Jádro je nejmenší a centrální komponentou operačního systému. Do jeho úkolů patří správa paměti a zařízení a také ostatním aplikacím poskytuje systémové zdroje. Dalšími službami, které jádro poskytuje může být například zajišťování ochrany pro programy a multitasking v závislosti na tom o jakou se jedná architekturu systému. Třemi nejpoužívanějšími architekturami jádra jsou:

1. *Monolitický kernel* – Všechny základní systémové služby jsou implementovány v prostoru kernelu. Typickými zástupci této architektury jsou GNU/Linux a MS Windows.
2. *Mikrokernel* – V jádře běží pouze základní komunikace mezi procesy a řízení vstupů a výstupů. Všechny ostatní služby pak běží v uživatelském prostoru. Systémy s touto architekturou jsou napři. GNU Mach a QNX.
3. *Exokernel* – v jádře běží pouze služby pro ochranu zdrojů (např. sledování vlastnictví, hlídání využití). To je prováděno poskytováním nízkoúrovňového rozhraní pro operační systémy složené z knihoven.

Úloha

K dosažení paralelismu v operačních systémech reálného času je aplikace rozložena na malé sekvence programu, které se nazývají úlohy. V kontextu reálného času je úloha základní spustitelnou jednotkou kódu a je řízena třemi klíčovými časové kritickými vlastnostmi: časem spuštění (release time), časem ve kterém musí být úloha splněna (deadline) a časem, který je potřeba ke spuštění úlohy (execution time). Každá úloha je definována pomocí následujících komponent:

- *Řídící blok úlohy* – datová struktura uložená v paměti RAM a přístupná pouze systému.
- *Zásobník úlohy* – data definována programem v paměti RAM. Je přístupný pomocí programového čítače.
- *Rutina úlohy* – kód programu uložený v ROM.

Úloha může existovat ve čtyřech stavech:

- *Aktivní* – úloha běží, procesor vykonává její instrukce.
- *Připravená* – úloha je připravena k běhu, čeká na přidělení procesorového času.
- *Zablokovaná* – úloha čeká na zdroje (přístup k periferii atp.)
- *Nečinná* – úloha nepotřebuje procesorový čas ani jiné zdroje.

Za běhu aplikace pak úlohy přechází mezi jednotlivými stavy. V aktivním stavu je však v případě jednojádrového procesoru za běhu aplikace v jejím libovolném bodě pouze jedna úloha. V průběhu výměny úloh je kontext úlohy (registry, process control

block atd.), která bude nečinná uložen, zatímco kontext aktivované úlohy je načten. Tento proces se nazývá přepnutí kontextu.

Správa úloh

Správa úloh umožňuje vytvářet software jako několik oddělených bloků kódu, z nichž každý má odlišný cíl a jiné přidělené limity. Zahrnuje také mechanismy jako je plánovač a dispatcher, které se starají o vytváření a údržbu úloh.

Plánovač

Plánovač udržuje záznamy o stavech jednotlivých úloh a volí z těch, které jsou připraveny tu, která bude aktivní; přiřazuje jí tedy procesorový čas. Pomáhá také maximalizovat využití CPU jednotlivými úlohami ve víceúlohovém prostředí a minimalizuje tak čekací časy. Obecně lze plánovače rozdělit do dvou skupin nepreemptivní a preemptivní založené na prioritách.

Nepreemptivní plánování, nebo také kooperativní multitasking vyžaduje od jednotlivých úloh, aby spolupracovaly a explicitně si předávaly kontrolu na CPU. Pokud některá z úloh uvolní procesor, je spuštěna další připravená úloha s nejvyšší prioritou.

Synchronizace úloh

Pomocí synchronizace úloh je zajištěn přístup ke sdíleným prostředkům (periferie, paměť atd.) a umožňuje také spouštění více úloh zároveň. Synchronizace je obvykle dosaženo pomocí dvou mechanizmů: událostí a semaforů.

Události jsou obvykle využívány k synchronizaci v případech, že není třeba, aby dvě úlohy sdílely prostředky. Umožňují jedné, popřípadě více úlohám čekat na určitou událost. Mohou existovat ve dvou různých stavech: v sepnutém stavu nebo rozepnutém stavu. V případě, že je událost v sepnutém stavu, může se čekající úloha přepnout do aktivního stavu a pokračovat v provádění další činnosti. Naproti tomu pokud je událost v rozepnutém stavu, musí čekající úloha zůstat uspaná.

Semafory pro svojí funkčnost využívají počítadlo a čekací frontu. Počítadlo slouží k indikaci počtu dostupných zdrojů, zatímco čekací fronta spravuje přístup úloh k prostředkům. Semafor se do určité míry chová obdobně jako klíč, který definuje, zda úloha může nebo nemůže ke zdroji přistupovat. Přístup ke zdrojům získá úloha pouze v případě, že uzamkne semafor. Číslo uložené v počítadle pak určuje kolikrát může být semafor uzamčen. Pokud dojde k uzamčení semaforu úlohou je počítadlo dekrementováno o jedničku. Obdobně pokud je počítadlo inkrementováno v případě, že je semafor uvolněn. Semafory lze rozdělit na tři typy:

1. *Binární semafory* – semafor s hodnotou 0 a 1 k indikaci dostupnosti nebo nedostupnosti zdroje.

2. *Čítající semafory* – nabývají hodnot 0 nebo vyšší a tedy mohou být uzamčeny nebo odemčeny vícenásobně.
3. *Semafory s vzájemným vyloučením* – nabývají hodnoty 0 a 1, ale počet zamčení může být libovolné kladné celé číslo pro realizaci rekurzivního uzamykání.

Meziúlohová komunikace

Vzájemná komunikace mezi úlohami zahrnuje zejména sdílení dat pomocí sdílené paměti a jejich přenos. K tomu je využíváno několik mechanizmů do nichž náleží:

- Fronty zpráv
- Roury
- Vzdálené volání procedur (RPC)

Fronta zpráv je komunikační objekt pomocí kterého jsou zprávy ukládány nebo vyzvedány ze sdílené paměti. Úlohy a obsluha přerušení (ISR) mohou se zprávami ve frontě operovat pomocí služeb jádra. V případě že se některá z úloh pokusí o čtení zprávy z prázdné fronty dojde k jejímu zablokování. Ukládání a vyzvedávání zpráv se může řídit jedním z následujících schémat: zásobník (LIFO), fronta (FIFO) nebo sekvence s prioritami. Ve většině případů fronta úloh sestává z několika parametrů jako jsou: řídící blok, jméno, unikátní ID, paměťové buffery, délka, maximální délka zprávy a jeden nebo více seznamů s čekajícími úlohami.

Roura je objektem, který poskytuje jednoduchý komunikační kanál pro výměnu nestrukturovaných dat mezi úlohami. Nejčastěji je roura jednosměrná a má dva deskriptory pro každý z jejích konců pro čtení a zápis. Data jsou do roury zapisována jako nestrukturovaný proud bytů za pomocí jednoho deskriptoru a vyčítána principem FIFO z druhého deskriptoru. Oproti frontě zpráv tedy není v rouře ukládáno několik bloků dat, ale jejich proud, navíc datový tok v rouře nelze priorizovat.

Vzdálené volání procedur umožňuje použití distribuovaných výpočtů tím, že vyvolává spouštění úloh na vzdáleném počítači tak jako by se jednalo o počítač lokální.

Správa paměti

RTOS usilují o minimalizaci nároků na paměť tím, že používají pouze funkcionalitu potřebnou pro uživatelské aplikace. Správa paměti sestává ze dvou částí, jsou to správa zásobníku a správa haldy.

Ve víceúlohových RTOS je třeba každou z úloh vytvořit s takovým objemem paměti, aby do něj bylo možno uložit celé jejich kontexty z důvodu přepínání kontextu. Alokace této paměti je vykonávána s využitím řídícího bloku úlohy. Tomuto bloku paměti se říká zásobník jádra (kernel stack) a proces jeho správy se nazývá Stack Managment.

Po dokončení inicializace programu je část paměti procesoru zaplněna kódem programu, daty programu a zásobníkem systému. Zbytku paměti se pak říká halda. Halda je pamětí používanou jádrem v případě dynamické alokace paměti. Paměť je rozdělena do bloků s pevnou velikostí, které pak mohou být úlohami využity. Pokud úloha ukončí využívání této bloku, musí jej navrátit do paměťového fondu. Tento proces správy paměti je také znám jako Heap Managment.

Správa časovačů

V embedded systémech jsou systémové a uživatelské úlohy často plánovány tak, aby byly vykonávány ve specifickém čase. K tomu, aby bylo takové plánování úloh možné je třeba vyvolávat periodicky přerušení, pomocí kterých jsou kontrolovány zpoždění a časových limitů. Většina dnešních RTOS poskytuje uživateli jak tzv. relativní časovače pracující s tiky a absolutní časovače jejichž pracovní jednotkou je kalendářní datum a čas. Pro oba typy časovačů RTOS poskytují služby zpoždění a upozornění úlohy založené na signalizačních mechanismech (např. pomocí událostí). Další poskytovanou službou časovačů je v kooperaci s plánovačem úloh zjišťování toho, zda úlohy dodržují nebo nedodržují plnění jejich časových lhůt.

Přerušení a obsluha událostí

Přerušení je hardwarový mechanismus používaný pro informaci procesoru o tom, že nastala asynchronní událost. Implementace podpory přerušení je velkou výzvou při návrhu RTOS vzhledem k tomu, že musí být umožněn asynchronní přístup do jeho vnitřních datových struktur. Přerušení a obsluha událostí v RTOS má následující funkce:

- Definuje obsluhy přerušení.
- Vytváří a maže ISR.
- Referencuje stavy ISR.
- Povoluje a zakazuje přerušení.
- Mění a referencuje masky přerušení.

a pomáhá zajistit:

- Integritu dat tím, že zamezuje vzniku přerušení během modifikace datových struktur.
- Minimální latence přerušení tím, že zakazuje přerušení, pokud systém provádí kritické operace.
- Nejrychlejší možnou odezvu na přerušení, které by mohly narušit preemptivní výkon systému.

- Nejkratší možný čas potřebný k dokončený obsluhy přerušení s minimálním využitím systémových prostředků.

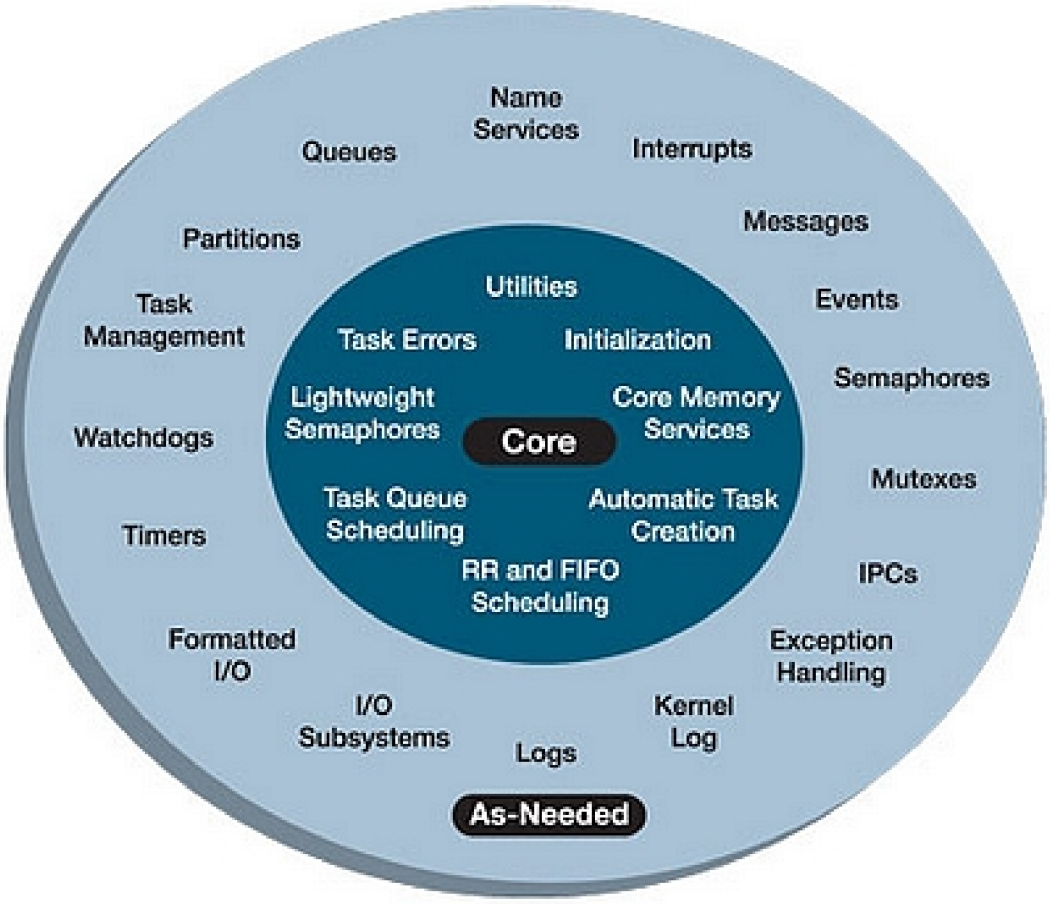
Správa vstupů a výstupů

Jádro RTOS je často vybaveno správou vstupních a výstupních periferií. K tomuto účelu poskytuje jednotné rozhraní (API) a dozorovací nástroje pro embedded systémy k organizaci a přístupu k velkému množství nejrůznějších ovladačů hardwarových zařízení. Většina API ovladačů je však standardních pouze uvnitř specifického RTOS.

7 OPERAČNÍ SYSTÉM MQX

Operační systém reálného času MQX od společnosti MQX Embedded byl navržen pro jednoprocesorové, víceprocesorové a distribuované aplikace.

MQX sestává z jádra a volitelných komponentů. Ve výsledné aplikaci jsou v případě jádra přítomny pouze ty funkce které jsou volány buďto samotným MQX nebo uživatelskou aplikací. Aby bylo dosaženo požadované funkčnosti uživatelská aplikace konfiguruje a rozšiřuje jádro přidáváním volitelných komponentů. Na následujícím obrázku je zobrazena struktura MQX, kde uprostřed je jádro a kolem něj jeho možná rozšíření.



Obrázek 7: Vnitřní struktura MQX [6]

Komponenta	**Funkce**	**Typ**
Inicializace	Inicializace a vytváření úloh (tasků)	jádro
Task Managment	Dynamický management úloh	jádro
Plánování procesů - Scheduling	Round Robin a FIFO	jádro
	Explicitní využívání úlohových front	volitelné
Synchronizace a komunikace mezi úlohami	Odlehčené semafory	jádro
	Semafory	volitelné
	Odlehčené eventy	volitelné
	Eventy	volitelné
	Mutexy	volitelné
	Zprávy	volitelné
	Fronty úloh	volitelné
Prostředky pro vnitřní komunikaci procesoru		volitelné
Časování	Komponenta času	volitelné
	Odlehčené časovače	volitelné
	Časovače	volitelné
	Watchdog	volitelné
Správa paměti	Paměť s proměnnou velikostí bloků	jádro
	Paměť s pevnou velikostí bloků (paměťové oddíly)	volitelné
	MMU, vyrovnávací a virtuální paměť	volitelné
	Odlehčená paměť	volitelné
Obsluha přerušení		volitelné
V/V ovladače	V/V subsystém	volitelné
	Formátované vstupy/výstupy	volitelné
Nástroje	Zásobník	jádro
	Záznamy jádra	volitelné
	Záznamy	volitelné
	Odlehčené záznamy	volitelné
Obsluha chybových stavů	Chybové kódy úloh, obsluha výjimek, testování za běhu	jádro
Manipulace s frontami		jádro
Jmenná komponenta		volitelné

Komponenta	Funkce	Typ
Embedded debugging	EDS Server	volitelné

Tabulka 3: Přehled součástí MQX [6]

7.1 Inicializace

Inicializace patří do jádra MQX. Aplikace je spuštěna voláním funkce `_mqx()` (hlavní funkce operačního systému). Tato funkce inicializuje hardware a spouští samotné MQX. Jakmile je MQX spuštěno, vytvoří a spustí úlohy které jsou nadefinovány v aplikaci jako úlohy s automatickým spouštěním.

7.2 Task Managment

Task managment se stará o automatické vytváření úloh po spuštění MQX. Díky němu může aplikace také vytvářet, spravovat a ukončovat úlohy za běhu nebo dynamicky měnit jejich vlastnosti:

- Spouštět několik instancí té samé úlohy.
- Vytvářet neomezený počet úloh v aplikaci.
- Přiřadit úlohám výstupní funkci, která je volána při jejich ukončení.
- Vytvářet obsluhy výjimek.

7.3 Plánování procesů

Plánování úloh vyhovuje standartu POSIX.4 (doplněn o rozšíření reálného času) a podporuje následující typy schedulingu:

- FIFO (také nazývaný priority-based preemptive) scheduling je obsažen v jádře – aktivní úloha je ta, která má nejvyšší prioritu a čekala nejdelší časový úsek.
- Round-Robin (nebo také time slice) scheduling. Součást jádra systému – aktivní úloha je ta, která má nejvyšší prioritu a je připravena nejdelší dobu aniž by spotřebovala své přidělené kvantum času.
- Explicitní scheduling (využívání úlohových front) je volitelná součást MQX. Lze jej využít k explicitnímu plánování úloh nebo vytvoření komplexnějších synchronizačních mechanizmů. Vzhledem k tomu, že poskytuje pouze základní funkcionalitu, je rychlý. Aplikace může při vytváření fronty úloh specifikovat použití Round-Robin nebo FIFO plánování.

7.4 Správa paměti

Správa paměti s proměnnou velikostí bloků

K alokaci volných paměťových bloků proměnné velikosti je v MQX k dispozici několik služeb obsažených v jádře, které jsou podobné rutinám `malloc()` a `free()`, obsažených v knihovnách jazyka C. V MQX je možné alokovat paměťové bloky uvnitř i vně výchozího paměťového fondu a přiřazovat je jak úlohám, tak systému. Paměť alokovaná pro úlohu se stává jejím zdrojem a je uvolněna automaticky po ukončení úlohy, která paměť alokovala.

Správa paměti s pevnou velikostí bloků (paměťové oddíly)

Paměťové oddíly jsou volitelnou součástí systému MQX a slouží k alokaci paměťových bloků s pevnou velikostí. Umožňuje rychlou a deterministickou alokaci paměti, snižuje fragmentaci a šetří paměť. Paměťové oddíly mohou být vytvořeny uvnitř (pak se jedná o dynamické oddíly) i vně výchozího paměťového fondu (statické oddíly). Stejně jako v případě paměťových bloků s proměnnou velikostí, mohou být bloky s pevnou velikostí přiřazeny úloze nebo systému a jsou automaticky uvolňovány po ukončení úlohy, která je alokovala.

Vyrovnávací paměť

Funkce MQX dovolují ovládat vyrovnávací paměť instrukcí a dat, která je obsažena v některých typech procesorů.

Řízení MMU

Pro některé typy procesorů je třeba před povolením vyrovnávací paměti potřeba inicializovat jednotku správy paměti – MMU (z angl. Memory Managment Unit). V systému MQX je možno MMU inicializovat, zapínat, vypínat a přiřazovat jí paměťové oblasti. Ovládání jednotky probíhá pomocí stránkovacích tabulek.

Odlehčená správa paměti

Pokud je aplikace omezená požadavky na velikost dat a nebo programu je možno použít modul odlehčené správy paměti. Její rozhraní má méně funkcí a velikost jejího kódu a dat je také menší. Výsledkem požití modulu je ovšem také zmenšená robustnost (dojde k odstranění kontrolních součtů hlaviček) a zpomalení aplikace (prodlouží se doba potřebná k ukončení úlohy).

7.5 Synchronizace úloh

K synchronizaci úloh slouží několik komponent, z nich některé lze využít ve dvou variantách: plnohodnotné nebo odlehčené. Systém MQX využívá standardní synchronizační objekty jako jsou mutexy, semafory a události.

Odlehčené události

Odlehčené události jsou volitelnou součástí systému MQX. Jsou způsobem jak zajistit synchronizaci úloh s minimálním využitím prostředků procesoru využívající změn stavů bitů.

Události

Události podporují dynamickou správu objektů, které jsou formátovány jako bitová pole. Úlohy a rutiny přerušení mohou využít událostí k synchronizaci a předávání jednoduchých informací ve formě změn bitových stavů. Je také možné je sdružovat do skupin. Tyto skupiny mohou využívat bitů s automatickým nulováním, které MQX nuluje okamžitě poté co jsou nastaveny. Aplikace může nastavovat bity událostí i na vzdáleném procesoru.

Odlehčené semafory

Odlehčené semafory jsou součástí jádra. Umožňují úlohám synchronizovat přístup ke sdíleným prostředkům s minimálním využitím procesorových prostředků – vyžadují minimální množství paměti a běží vysokou rychlostí. Ve svém principu jsou to čítající FIFO semafory bez dědění priorit.

Semafory

Semafory lze využít několika způsoby: k synchronizaci úloh, ochránění přístupu ke sdíleným prostředkům nebo k implementaci signalizačního mechanizmu typu producent/konzument. Semafory dále umožňují řazení typu FIFO nebo podle priorit a také dědění priorit. Lze využít několik typů semaforů: pojmenované, rychlé, striktní nebo volné. Jako v případě odlehčených semaforů se jedná o semafory čítající.

Mutexy

Mutexy (z angl. mutual exception) jsou volitelnou komponentou, který vyhovuje normě POSIX.4a (rozšíření standardu o vlákna) a umožňují úlohám využívat principu vzájemného vyloučení při přístupu ke sdíleným prostředkům. Poskytují několik možných typů zjišťování jejich stavu: dotazování, fronty FIFO, priorizované fronty, spin only a limited-spin fronty a také dědění a ochranu priorit. Mutexy jsou striktní, tzn. že úloha nemůže uvolnit mutex, aniž by předtím nabyla jeho vlastnictví (neuzamkla ho).

Zprávy

Úlohy mohou mezi sebou mohou komunikovat pomocí zpráv, které jsou přístupné pro všechny úlohy. Každá úloh si pro sebe otevírá vlastní vstupní frontu zpráv. Každá fronta zpráv je jednoznačně identifikovatelná pomocí ID, které jí přiřazuje MQX při jejím vzniku. Pouze úloha, která otevře frontu zpráv z ní může zprávy přijímat. Platí také, že libovolná úloha může posílat zprávy do libovolné otevřené fronty, pokud zná její ID.

Úlohy alokují zprávy z fondů zpráv. MQX obsahuje dva typy těchto fondů – soukromý a systémový. Pro jejich využití platí následující: libovolná úloha může alokovat zprávu (systémovou) ze systémového fondu a libovolná úloha, může alokovat zprávu ze soukromého fondu, pokud zná jeho ID. Zprávy jsou volitelnou součástí MQX.

Fronty úloh

Slouží jako rozšíření k poskytovanému mechanizmu plánování procesů poskytující jednoduchý a efektivní způsob synchronizace úloh. Úlohy ve frontě mohou být pozastaveny a nebo z fronty vyjmuty.

Prostředky pro vnitřní komunikaci procesoru

Aplikace mohou běžet na několika procesorech najednou s jedním spustitelným obrazem systému MQX na každém z těchto procesorů. Tyto obrazy spolu spolupracují a komunikují s využitím zpráv, které jsou předávány skrze paměť nebo komunikační linky vnitřní komunikace procesoru. Úlohy aplikace v každém z obrazů mohou být a často bývají rozdílné.

7.6 Časování

Funkce času jsou volitelnou součástí systému, která umožňuje sledování jak absolutního, tak relativního času, kde parametry absolutního času lze měnit. Rozlišení času je závislé na rozlišení definovaném aplikací nastaveném pro danou hardwarovou platformu při startu MQX.

Odlehčené časovače

Jsou volitelnou součástí a poskytují způsob pro volání funkcí v aplikace v periodických intervalech s minimálním využitím systémových prostředků. Odlehčené časovače jsou instalovány vytvořením fronty, která je vyvolávána periodicky a poté přidáním časovače tak, aby jeho čas vypršel s offsetem od počátku periody této fronty.

Při přidání odlehčeného časovače do fronty je uživatelem specifikována funkce kterou systém MQX zavolá pomocí obsluhy přerušení při vypršení časovače. Vzhledem

k tomu, že je funkce časovače obsluhována pomocí přerušení, ne všechny funkce MQX mohou být funkcemi časovače.

Časovače

Umožňují periodické spouštění funkce v aplikaci. Systém MQX podporuje dva druhy časovačů: jednočinné (k vyvolání přerušení dojde pouze jednou a časovač se zastaví) a periodické (přerušení přichází periodicky s nastaveným intervalem). Časovače mohou být nastaveny tak, aby byly spuštěny v určitý čas nebo po uplynutí nastaveného času.

Při nastavení časovače uživatel specifikuje funkci oznámení, která je volána úlohou časovače při jeho vypršení. Tato funkce může být využita k synchronizaci úloh pomocí poslání zprávy, vznikem události nebo použitím některého z dalších synchronizačních mechanizmů. Časovače jsou volitelnou součástí systému MQX.

Watchdog

Watchdogy jsou volitelnou součástí systému, která uživateli umožňuje detekovat deadlocky a starvation úloh.

7.7 Obsluha přerušení

Přerušení a obsluha výjimek jsou volitelnými součástmi na úrovni balíčků pro zvolený procesor (PSP z angl. Processor Specific Package viz vysvětlující příloha č.6). Systém MQX obsluhuje hardwarová přerušení v rozsahu definovaném pomocí balíčku pro danou hardwarovou platformu (BSP z angl. Board Specific Package) a ukládá minimální rozsah kontextu pro každou aktivní úlohu, podporuje také plně vnořená přerušení v případě že je podporuje procesor. Uvnitř rutin přerušení může aplikace povolit libovolnou úroveň přerušení. Ke zlepšení odezvy na přerušení systém MQX odloží naplánování úlohy dokud nedojde k obsluze všech požadavků na přerušení. Navíc k naplánování úlohy dojde pouze v případě, že byla nová úloha převedena do stavu „připraven“ některou z rutin obsluhy přerušení.

Obsluha přerušení není úlohou. Jedná se o malou velmi rychlou rutinu která rychle reaguje na hardwarové přerušení, je obvykle napsána v jazyce C a jejími povinnostmi je přenastavení zařízení, získání dat a upozornění odpovídající úlohy. Může být využita k signalizaci úlohy obsahující libovolnou neblokující funkci MQX.

7.8 Nástroje

V systému MQX je dispozici je několik programátorských a systémových nástrojů usnadňující ladění aplikací a sledování stavu systému. Jsou to: záznamy, odlehčené záznamy, záznamy z jádra a nástroj pro monitorování využití zásobníku.

Záznamy

Záznamy jsou volitelnou součástí, která umožňuje uchovávat a získávat informace o aplikaci. Každý záznam má časovou známku a číslo sekvence. Získané informace lze využít k testování, ladění, ověřování a analýze výkonu aplikace.

Odlehčené záznamy

Jsou podobné obyčejným záznamům, avšak využívají informace, které mají pevnou velikost. Díky tomu jsou rychlejší než standardní záznamy a jsou využívány při zaznamenávání činnosti jádra.

Záznamy jádra

Jsou volitelnou součástí která umožňuje zaznamenávat aktivitu MQX. Je možné je vytvořit na specifickém místě v paměti, nebo o jejich umístění nechat rozhodnout systém. Mohou být zkonfigurovány tak aby ukládaly informace o přepnutí kontextu, volání funkcí a obsluze přerušení.

Využití zásobníku

MQX obsahuje několik funkcí, které umožňují monitorovat vytížení zásobníku přerušení a zásobníku úloh všemi úlohami a tak efektivně zjistit zda mají alokován dostatečný prostor v paměti procesoru.

7.9 Ošetření chyb

Každá úloha má chybový kód, který odpovídá jeho kontextu. Specifické funkce MQX pak tento kód čtou a aktualizují.

Obsluha výjimek

V MXQ je možno specifikovat výchozí rutinu přerušení, která se stará o neošetřené výjimky a také obslužnou funkci v případě že dojde v výjimce v průběhu obsluhy přerušení.

Operace s frontami

K manipulaci s frontami je určena komponenta, která zajišťuje vytvoření obousměrného seznamu všech prvků fronty, s frontami lze provádět standardní úkony: inicializaci, přidání prvků, odebrání prvků a jejich čtení.

Jmenná komponenta

Tato komponenta poskytuje databázi jmen, která mapuje textové řetězce na číselné popisovače vytvářené systémem, jako je např. ID fronty.

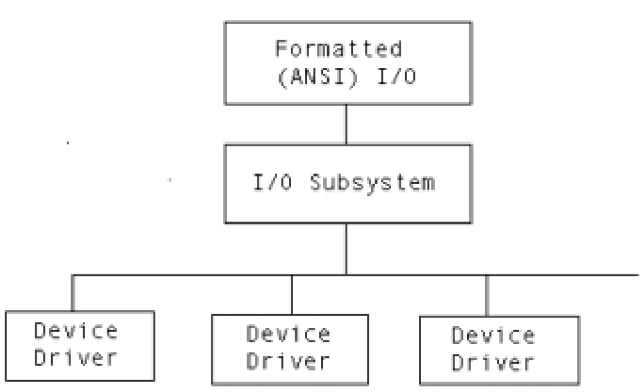
7.10 Inicializace a spuštění MQX

Spuštění systému MQX probíhá zavoláním funkce `_mqx()`, jejíž parametrem je inicializační struktura. V závislosti na hodnotách proměnných této struktury je provedeno následující:

- Nastavení a inicializace vnitřních dat MQX včetně výchozího fondu paměti, front připravených úloh, zásobníku přerušení a zásobníku úloh.
- Inicializace hardwaru (například aktivace chip select).
- Povolení časovačů.
- Nastavení výchozí hodnoty časového kvanta.
- Vytvoření „nečinné“ úlohy, která je aktivní v případě že žádná z úloh není ve stavu „připravena“.
- Vytvoření úloh, které jsou v seznamu úloh definovány jako úlohy s automatickým spuštěním.
- Spuštění plánovače úloh.

7.11 Vstupně výstupní ovladače

Vstupně výstupní ovladače jsou volitelnou součástí na úrovni BSP, sestávají z formátovaných vstupů a výstupů a V/V subsystému. Vrstvy V/V modelu lze vidět na následujícím obrázku:

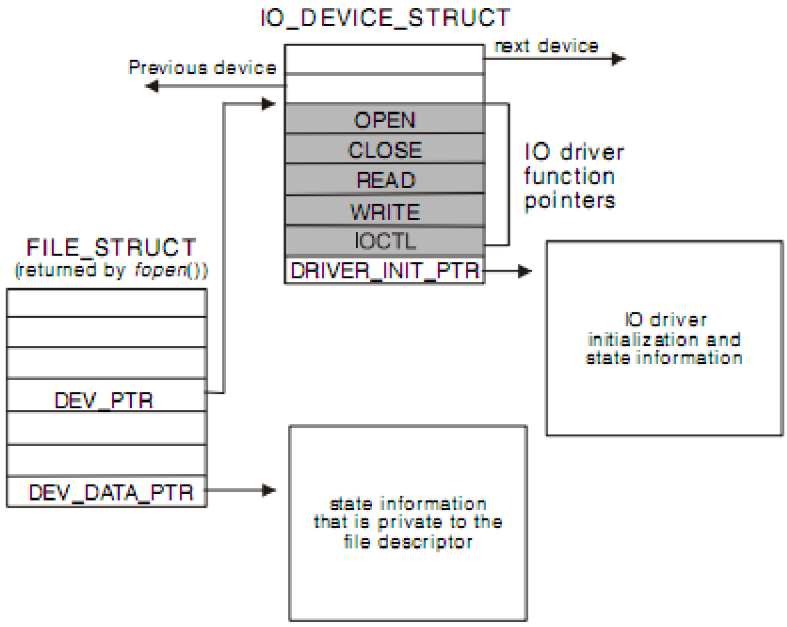


Obrázek 8: Vrstvy V/V ovladače systému MQX [16]

Díky tomuto vrstvenému přístupu je možné aby ovladače zařízení přistupovaly k ovladačům jiných zařízení.

Struktura V/V zařízení

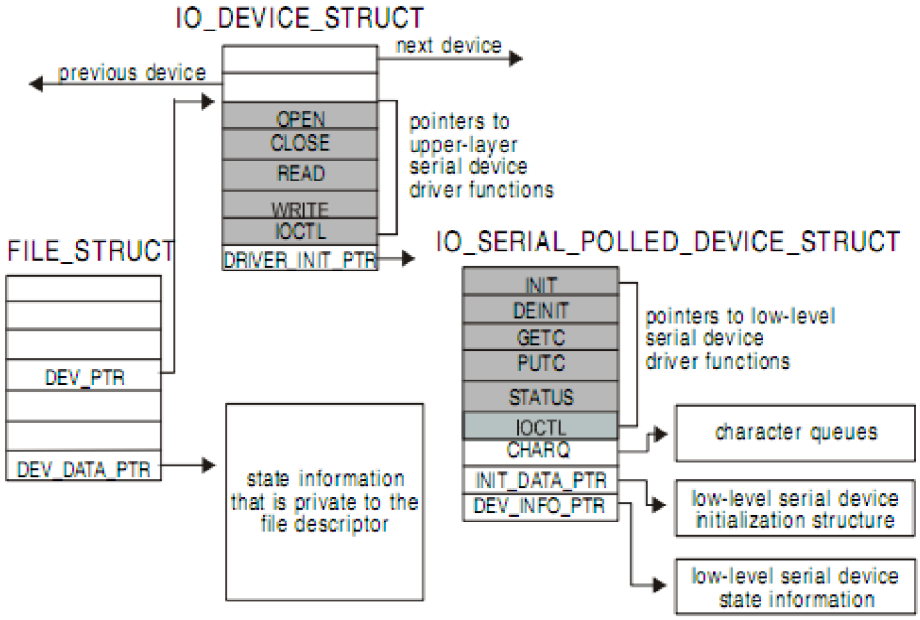
Na obrázku č.9 je vidět vztah mezi handlem souboru (FILE_STRUCT), který je navracen funkcí `fopen()`, strukturou V/V zařízení alokovanou při instalaci zařízení a funkcemi V/V ovladače pro všechny zařízení.



Obrázek 9: Struktura V/V ovladače[16]

Struktura V/V ovladače pro sériová zařízení

Ovladače sériových zařízení jsou komplexnější v tom, že mají oproti obecné vrstvě ovladače navíc nízkoúrovňovou vrstvu pro jednoduchý přístup k sériovému hardwaru. Na následujícím obrázku je vidět toto rozšíření:



Obrázek 10: Struktura V/V ovladače sériového zařízení[16]

Knihovna formátovaných vstupů a výstupů

Knihovna formátovaných vstupů a výstupů je podmnožinou implementace standardní knihovny ANSI C. Tato knihovna obstarává volání V/V subsystému. V systému MQX se tyto funkce liší přidáním prefixu „`_io_`“, např. ekvivalent `fclose()` je tedy `_io_flose()`. Podrobný soupis podporovaných funkcí lze nalézt v [16] strana 3 a 4.

Subsystém vstupů a výstupů

Implementace V/V subsystému je mírně odlišná od standardních vstupů a výstupů standartu POSIX. Je schodná s UNIX modelem využívajícím funkce `open()`, `close()`, `read()`, `write()` a `ioctl()`. V/V subsystém pak volá funkce ovladače V/V zařízení. Vstupy a výstupy v MQX využívají ukazatele na soubor, který je navracen funkcí `fopen()` místo souborových popisovačů.

Názvy zařízení

Řetězec, který je použit k pojmenování zařízení musí končit dvojtečkou („:“). Další znaky za dvojtečkou jsou chápány jako další informace pro zařízení, které jsou mu předány při volání funkce `fopen()`. Například zavoláním:

```
fopen("mfs1:foobar.txt");
```

Dojde k otevření souboru *foobar.txt* na zařízení `mfs1`.

Instalace ovladačů zařízení

K instalaci ovladače zařízení je třeba použít jeden z následujících postupů:

- Zavolat funkci `_io_device_install()`, kde slovo `device` je nahrazeno názvem rodiny ovladačů z uživatelské aplikace. Tato funkce obvykle uvnitř volá `_io_dev_install()` k zaregistrování zařízení do systému MQX. Dále také vykonává inicializaci zařízení jako je alokace paměti a dalších objektů potřebných pro jeho operace (např. semafory).
- Zavolat funkci `_io_dev_install()` z BSP nebo uživatelské aplikace.

8 NAVRŽENÝ OVLADAČ SBĚRNICE I²S V SYSTÉMU MQX

Ovladač je navržen podle konceptu na obr. č.9. Tedy jeho programátorské rozhraní je zapouzdřeno ve struktuře `IO_DEVICE_STRUCT` ve které jsou uloženy ukazatele na vlastní funkce ovladače a strukturu `MCF54XX_I2S_INIT_STRUCT` obsahující informace sloužící k inicializaci při instalaci ovladače. Celý ovladač se skládá z několika souborů uložených v cestě *%instalační adresář MQX%\mqx\source\io\i2s*, jejichž přehled a stručný popis je uveden v následující tabulce:

Název souboru	Obsah
i2s_mcf54xx.h	Inicializační struktura pro ovladač a funkční prototypy specifické pro zvolenou vývojovou desku.
i2s_mcf54xx_prv.h	Definice konstant a datových struktur pro vnitřní rozhraní v ovladači.
\int\i2s_int_prv.h	Definice struktury zařízení a prototypy I/O funkcí.
i2s.h	Definice konstant a struktur pro vnějšího rozhraní ovladače.
i2s_audio.h	Definice konstant a struktur pro zařízení pracující se zvukem.
i2s_mcf5441.c	Kód hardwarové inicializace modulu SSI.
\int\i2s_int.c	Kód zajišťující mezivrstvu pro připojení ovladače do MQX.
\int\i2s_int_mcf54xx.c	Kód obstarávající nízkoúrovňové funkce ovladače.

Tabulka 4: Přehled souborů ovladače

Ovladač využívá ke své funkci přerušení sběrnice SSI a její hardwarové zásobníky typu FIFO. Tyto zásobníky slouží jako vyrovnávací paměť v průběhu formátování dat a dalších podpůrných operací.

8.1 Instalace ovladače

Ovladač je možno nainstalovat dvěma způsoby: buďto povolit jeho instalaci v MQX při kompilaci pomocí makra `BSPCFG_ENABLE_II2S0` v souboru *user_config.h* (soubor pro globální nastavení parametrů kompilace pro MQX) nebo v samotné aplikaci pomocí volání funkce `_mcf54xx_i2s_install()` s požadovanými parametry. Tyto parametry jsou následující:

1. ukazatel na textový řetězec, který bude sloužit jako identifikátor zařízení pro funkci `fopen()`. V případě instalace systémem je dostupné zařízení s názvem `ii2s0:`.
2. Ukazatel na inicializační strukturu typu `MCF54XX_I2S_INIT_STRUCT`.

Inicializační struktura pak obsahuje následující položky:

Název	Datový typ	Funkce
CHANNEL	uint_8	Výběr kanálu SSI.
MODE	uint_8	Nastavení módu ovladače (master – slave).
DATA_BITS	uint_8	Počet platných datových bitů ve vzorku.
CLOCK_SOURCE	uint_8	Nastavení zdroje signálu master clock.
TX_DUMMY	boolean	Vysílání bez vstupních dat (ano – ne).
LEVEL	_int_level	Úroveň přerušení využívaná ovladačem.
BUFFER_SIZE	uint_32	Velikost vnitřního zásobníku ovladače.
AUDIO_DATA_FORMAT	const * IO_FORMAT	Ukazatel na strukturu definující formát vstupních a výstupních dat.

Tabulka 5: Funkce jednotlivých proměnných inicializační struktury

Kontrola správného rozsahu hodnot jednotlivých položek struktury je prováděna uvnitř interní inicializační funkce ovladače `_mcf54xx_i2s_int_init()`. V případě překročení povoleného rozsahu je funkcí navrácen jeden z následujících chybových kódů:

Nazev chyby	Podmínka
I2S_ERROR_INVALID_PARAMETER	Ukazatel na inicializační strukturu = NULL.
I2S_ERROR_CHANNEL_INVALID	Zadaný kanál není dostupný (> 1).
I2S_ERROR_MODE_INVALID	Mód neodpovídá makrům I2S_MODE_SLAVE ani I2S_MODE_MASTER.
I2S_ERROR_WORD_LENGTH_UNSUPPORTED	Zadaná délka datového slova je neplatná .
I2S_ERROR_CLK_INVALID	Zadaný zdroj signálu master clock je neplatný
I2S_ERROR_BUFFER_SMALL	Zadaná velikost zásobníku je příliš malá (< 2).
AUDIO_ERROR_INVALID_IO_FORMAT	Neplatný požadovaný formát dat.

MQX_OUT_OF_MEMORY	Nedostatek paměti pro vytvoření bufferů nebo informační struktury.

Tabulka 6: Chybové kódy inicializační funkce ovladače

V případě správné inicializace je navrácena hodnota makra `I2S_OK`. Po instalaci ovladač převezme nad periferií SSI plnou kontrolu a přepne ji do režimu kompatibilního se sběrnicí I^2S.

8.2 Uživatelské rozhraní ovladače

Uživatelské rozhraní sestává z funkcí uvedených v následující tabulce:

Funkce volaná v uživatelském programu	Funkce volaná uvnitř MQX.	Nízkoúrovňová funkce ovladače (závislá na platformě).
_io_fopen	_io_i2s_open	_mcf54xx_i2s_int_init
_io_fclose	_io_i2s_close	_mcf54xx_i2s_int_deinit
_io_read	_io_i2s_read	_mcf54xx_i2s_int_rx
_io_write	_io_i2s_write	_mcf54xx_i2s_int_tx
_io_ioctl	_io_i2s_ioctl	_mcf54xx_i2s_int_ioctl

Tabulka 7: Přehled funkcí ovladače

Ovladač tedy sestává ze tří úrovní:

- Uživatelská – tato je stejná pro všechny ovladače zařízení v MQX, pouze se mění její parametry a podle nich se volá funkce vnitřní vrstvy.
- Vnitřní – ta je volána systémem a je specifická pro dané zařízení (sběrnice, souborový systém atd.), ale stejná pro všechny hardwarové platformy.
- Nízkoúrovňová – funkce této úrovně jsou volány na základě toho na jakém procesoru je systém MQX spuštěn.

Nastavení a funkci ovladače je možno změnit, nebo přečíst pomocí funkce `ioctl`. Požadovaný příkaz pro ovladač je předáván jako první parametr funkce, druhým parametrem jsou potřebné údaje pro provedení příkazu. Příkazy implementované v ovladači jsou definovány v souborech *i2s.h* případně *i2s_audio.h* a uvedeny v následující tabulce:

Název příkazu	Datový typ parametru příkazu	Funkce
IO_IOCTL_I2S_SET_MODE_MASTER	–	Nastavení sběrnice do módu master
IO_IOCTL_I2S_SET_MODE_SLAVE	–	Nastavení ovladače do módu slave
IO_IOCTL_I2S_SET_CLOCK_SOURCE_INT	–	Přepnutí na vnitřní zdroj signálu master clock
IO_IOCTL_I2S_SET_CLOCK_SOURCE_EXT	–	Přepnutí na vnější zdroj signálu master clock
IO_IOCTL_I2S_SET_DATA_BITS	uint_8_ptr	Nastavení délky slova vysílaných dat.
IO_IOCTL_I2S_DISABLE_DEVICE	–	Zakázání příjmu i vysílání
IO_IOCTL_I2S_ENABLE_DEVICE	–	Povolení příjmu a vysílání
IO_IOCTL_I2S_SET_MCLK_FREQ	uint_32_ptr	Nastavení frekvence signálu master clock
IO_IOCTL_I2S_SET_FS_FREQ	uint_32_ptr	Nastavení synchronizační frekvence sběrnice.
IO_IOCTL_I2S_TX_DUMMY_ON	–	Povolení vysílání bez vstupních dat.
IO_IOCTL_I2S_TX_DUMMY_OFF	–	Zakázání vysílání bez vstupních dat.
IO_IOCTL_I2S_GET_MODE	uint_8_ptr	Získání aktuálního módu.
IO_IOCTL_I2S_GET_CLOCK_SOURCE	uint_8_ptr	Získání nastaveného zdroje master clock,
IO_IOCTL_I2S_GET_DATA_BITS	uint_8_ptr	Získání nastavené délky slova.
IO_IOCTL_I2S_GET_MCLK_FREQ	uint_32_ptr	Získání aktuální frekvence signálu master clock.
IO_IOCTL_I2S_GET_BCLK_FREQ	uint_32_ptr	Získání aktuální frekvence hodinového signálu.

IO_IOCTL_I2S_GET_TX_DUMMY	boolean*	Získání stavu vysílání bez dat.
IO_IOCTL_I2S_GET_FS_FREQ	uint_32_ptr	Získání aktuální frekvence sinchronizačního signálu.
IO_IOCTL_I2S_GET_STATISTICS	I2S_STATISTICS_STRUCT_PTR	Získání statistik o V/V operacích ovladače.
IO_IOCTL_I2S_SET_TXFIFO_WATERMARK	uint_8_ptr	Nastavení hodnoty watermark ve vysílači.
IO_IOCTL_I2S_SET_RXFIFO_WATERMARK	uint_8_ptr	Nastavení hodnoty watermark v přijímači.
IO_IOCTL_I2S_GET_TXFIFO_WATERMARK	uint_8_ptr	Získání hodnoty watermark z vysílače.
IO_IOCTL_I2S_GET_RXFIFO_WATERMARK	uint_8_ptr	Získání hodnoty watermark z přijímače.
IO_IOCTL_I2S_SET_CLK_ALWAYS_ENABLED_ON	–	Zapnutí trvalého vysílání na nedatových vodičích.
IO_IOCTL_I2S_SET_CLK_ALWAYS_ENABLED_OFF	–	Vypnutí trvalého vysílání na nedatových vodičích.
IO_IOCTL_I2S_GET_CLK_ALWAYS_ENABLED	boolean*	Získání nastavení trvalého vysílání na nedatových vodičích.
IO_IOCTL_I2S_CLEAR_STATISTICS	–	Vymazání statistik
IO_IOCTL_AUDIO_SET_IO_DATA_FORMAT	AUDIO_DATA_FORMAT_PTR	Nastavení formátu vstupních a výstupních dat.
IO_IOCTL_AUDIO_GET_IO_DATA_FORMAT	AUDIO_DATA_FORMAT_PTR	Získání nastaveného formátu vstupních a výstupních dat.

Tabulka 8: Přehled příkazů poskytovaných ovladačem

Uvedené příkazy lze rozdělit do tří skupin:

1. Příkazy pro přepínání dvouhodnotových parametrů. Tyto obvykle obsahují v názvu výrazy ON a OFF (z angl. zapnout – vypnout). Například `IO_IOCTL_I2S_TX_DUMMY_ON` a `IO_IOCTL_I2S_TX_DUMMY_OFF`.
2. Příkazy pro nastavování hodnot. Ty obsahují v názvu slovo SET (z angl. nastavit) Například `IO_IOCTL_I2S_SET_DATA_BITS`.

3. Příkazy pro zjištění hodnot. Ty mají v názvu slovo GET (z angl. získat). Například `IO_IOCTL_I2S_GET_FS_FREQ`.

Všechny uvedené příkazy mají zabudovánu kontrolu vstupních parametrů. V případě že příkaz vyžaduje parametr, ale místo něj dostane prázdný ukazatel s hodnotou `NULL` vrátí se funkce `ioctl` s návratovým kódem `I2S_ERROR_INVALID_PARAMETER`. Pokud je předaný parametr mimo jeho povolený rozsah, pak se funkce vrátí s návratovým kódem `I2S_ERROR_PARAM_OUT_OF_RANGE`.

IO_IOCTL_I2S_SET_MODE_MASTER

Pomocí tohoto příkazu je sběrnice přepnuta do módu master. Toto přepnutí je možno provést pouze v případě, že ovladač nevykonává V/V funkci. Přepnutím dojde k nastavení odpovídajícího pinu procesoru do stavu ve kterém přijímá resp. vysílá signál master clock na sběrnici podle toho jestli je zvolen interní nebo externí zdroj tohoto signálu.

IO_IOCTL_I2S_SET_MODE_SLAVE

Tímto příkazem dojde k nastavení procesoru do módu slave. V tomto módu jsou nastaveny potřebné registry tak, že veškeré signály sběrnice jsou přijímány z externího zdroje.

IO_IOCTL_I2S_SET_CLOCK_SOURCE_INT

Při přepnutí na vnitřní zdroj signálu master clock tímto příkazem dojde k přivedení hodinového signálu z vnitřního fázového závěsu procesoru přes nastavitelnou děličku na signál master clock. Poté jsou také podle nastavení dílčích děliček přepočteny všechny hodnoty odvozených signálů.

IO_IOCTL_I2S_SET_CLOCK_SOURCE_EXT

Přepnutím na externí zdroj master clock dojde k odpojení fázového závěsu procesoru od periferie a veškeré signály sběrnice je třeba do procesoru přivádět externě.

IO_IOCTL_I2S_SET_DATA_BITS

Tímto příkazem je možné omezit šířku datového slova na sběrnici. Parametrem příkazu je bitová hloubka, které musí být 8, 10, 12, 16, 18, 20, 22 nebo 24 bitů. Ořezávání probíhá hardwarově a tedy výkon ovladače nijak nezmenšuje.

IO_IOCTL_I2S_DISABLE_DEVICE a **IO_IOCTL_I2S_ENABLE_DEVICE**

První příkaz slouží k vypnutí periferie, druhý k zapnutí periferie. To probíhá tak, že jsou zakázány resp. povoleny přerušení od vysílače i přijímače a poté je přijímač i vysílač vypnut nebo zapnut.

IO_IOCTL_I2S_SET_MCLK_FREQ

Tento příkaz slouží k nastavení frekvence signálu master clock. V případě že je zvolen interní zdroj signálu jsou podle požadované frekvence v hertzích předané v parametru vypočteny hodnoty děliček a zpětně je vypočtena frekvence těmito děličkami nastavená. Tím dojde k nastavení nejbližší možné frekvence (oproti požadované), která je poté uložena do informační struktury, odkud je možno si její hodnotu přečíst pomocí odpovídajícího příkazu. Obdobně jsou vypočítány ostatní frekvence, které jsou na signálu master clock závislé. Pokud je zdroj signálu vnější, je do informační struktury uložen rovnou předaný parametr.

IO_IOCTL_I2S_SET_FS_FREQ

Pomocí tohoto příkazu lze nastavit frekvenci synchronizačního signálu, který odvozen ze signálu master clock. Frekvence je předána pomocí parametru z něhož jsou vypočteny potřebné hodnoty děliček.

IO_IOCTL_I2S_TX_DUMMY_ON a **IO_IOCTL_I2S_TX_DUMMY_OFF**

Tyto dva příkazy slouží k zapnutí/vypnutí režimu vysílání bez vstupních dat, který funguje tak, že po sběrnici jsou vysílána data vygenerovaná ovladačem místo dat uživatelských. Generovaná data sestávají z harmonického signálu frekvenci 440 Hz, jsou vzorkována na frekvencí synchronizačního signálu a uložena ve vnitřní paměti ovladače.

IO_IOCTL_I2S_GET_MODE

Příkaz pro získání nastaveného módu sběrnice. Vrací do parametru hodnotu makra `I2S_MODE_MASTER` nebo `I2S_MODE_SLAVE`.

IO_IOCTL_I2S_GET_CLOCK_SOURCE

Tento příkaz vrací přes ukazatel předaný v parametru hodnotu makra `I2S_CLK_INT` pro vnitřní zdroj nebo `I2S_CLK_EXT` podle zvoleného zdroje signálu master clock.

IO_IOCTL_I2S_GET_DATA_BITS

Do paměti na kterou ukazuje ukazatel předaný jako parametr tento příkaz předává zvolenou nastavenou šířku datového slova.

IO_IOCTL_I2S_GET_MCLK_FREQ

Pro získání aktuální frekvence signálu master clock je použit tento příkaz. Přes ukazatel předaný jako parametr je tato frekvence uložena v hertzích jako 32 bitů dlouhé celé číslo.

IO_IOCTL_I2S_GET_BCLK_FREQ

Pomocí tohoto příkazu lze získat frekvenci hodinového kmitočtu sběrnice. Tato frekvence je odvozena od frekvence synchronizačního signálu a délky datového slova. Vypočítaná hodnota v hertzích je navrácena pomocí ukazatele předaného jako parametr.

IO_IOCTL_I2S_GET_TX_DUMMY

Je příkazem, který slouží ke zjištění stavu režimu vysílání bez dat. Hodnota proměnné do které je uložen výsledek dotazu nabývá hodnot log.0 nebo log.1 podle toho jestli ovladač zpracovává uživatelská data (log.0) nebo vysílá testovací harmonický signál (log.1). K předání hodnoty je použit parametr, který je ukazatelem na booleovskou proměnnou.

IO_IOCTL_I2S_GET_FS_FREQ

Tímto příkazem je do proměnné na kterou ukazuje parametr předána hodnota frekvence synchronizačního signálu SSI_FS v hertzích.

IO_IOCTL_I2S_GET_STATISTICS

Tento příkaz vrací přes ukazatel v parametru statistiku ovladače. Ta je uložena ve formě struktury datového typu `I2S_STATISTICS_STRUCT`, která obsahuje proměnné jejichž popis je uveden v následující tabulce:

Název	Datový typ	Funkce
INTERRUPTS	uint_32	Počet přerušení vygenerovaných ovladačem.
UNDERRUNS_L	uint_32	Počet podtečení levého bufferu při vysílání.
UNDERRUNS_R	uint_32	Počet podtečení pravého bufferu při vysílání.
OVERRUNS_L	uint_32	Počet přetečení levého bufferu při přijímání.
OVERRUNS_R	uint_32	Počet přetečení pravého bufferu při přijímání.
RX_PACKETS	uint_32	Celkový počet přijatých vzorků.
TX_PACKETS	uint_32	Celkový počet odeslaných vzorků.
PACKETS_PROCESSED_L	uint_32	Počet odeslaných vzorků na levém kanálu.
PACKETS_QUEUED_L	uint_32	Počet bufferovaných vzorků na levém kanálu.
PACKETS_REQUESTED_L	uint_32	Požadovaný počet odeslaných vzorků pro levý kanál.
PACKETS_PROCESSED_R	uint_32	Počet odeslaných vzorků na pravém kanálu.
PACKETS_QUEUED_R	uint_32	Počet bufferovaných vzorků na pravém kanálu.
PACKETS_REQUESTED_R	uint_32	Požadovaný počet odeslaných vzorků pro pravý kanál.

Tabulka 9: Obsah struktury I2S_STATISTICS_STRUCT

Obsah této struktury je vždy automaticky smazán po zavření ovladače funkcí `_io_fclose()`. V případě že uživatel požaduje smazání dříve, je možné to udělat pomocí příkazu `IO_IOCTL_I2S_CLEAR_STATISTICS`.

IO_IOCTL_I2S_SET_TXFIFO_WATERMARK

Tímto příkazem lze změnit hodnotu watermark pro oba vysílací FIFO buffery. Při poklesu počtu vzorků v zásobníku pod watermark dojde k vygenerování přerušení. Změnou této hodnoty je tedy možné ovlivnit frekvenci přerušení generovaných ovladačem. Je však třeba učinit kompromis tak, aby nedocházelo k podtečení zásobníku vlivem nedostatečné rychlosti zpracování dat ovladačem. Ve výchozím stavu je tato hodnota nastavena na hodnotu watermark = 5, povolený rozsah hodnot je 0-15 vzorků. Nastavenou hodnotu lze získat příkazem

IO_IOCTL_I2S_SET_RXFIFO_WATERMARK

Hodnotu watermark pro přijímač lze změnit pomocí tohoto příkazu. Funguje obdobně jako příkaz pro vysílač s tím rozdílem, že přerušení je generováno v případě že počet vzorků v bufferu přesáhne nastavenou hodnotu. Ve výchozím stavu je hodnota watermark = 8. Povolený rozsah hodnot je stejný jako v případě vysílače.

IO_IOCTL_I2S_GET_TXFIFO_WATERMARK a
IO_IOCTL_I2S_GET_RXFIFO_WATERMARK

Příkazy pro získání nastavené hodnoty watermark pro vysílač a přijímač. Hodnota je předána pomocí ukazatele v parametru jako celé číslo.

IO_IOCTL_I2S_SET_CLK_ALWAYS_ENABLED_ON a
IO_IOCTL_I2S_SET_CLK_ALWAYS_ENABLED_OFF

Tyto příkazy slouží k zapnutí a vypnutí vysílání hodinového, synchronizačního a master clock signálu v případě, že nejsou vysílána data. Zmíněné signály totiž mohou být zařízeními připojenými na sběrnici vyžadovány i když nepřijímají ani nevysílají data zejména proto, že je využívají jako svoje hodinové kmitočty.

IO_IOCTL_I2S_GET_CLK_ALWAYS_ENABLED

Tímto příkazem je možné zjistit aktuální nastavení pro vysílání na nedatových vodičích. V případě že je vysílání zapnuto je pomocí ukazatele v parametru předána hodnota log.1 v opačném případě je v parametru log.0.

IO_IOCTL_AUDIO_SET_IO_DATA_FORMAT

Tento příkaz slouží k nastavení formátu vstupních a výstupních dat. Celé nastavení je příkazu předáno pomocí ukazatele na strukturu typu `AUDIO_DATA_FORMAT` jejíž proměnné a jejich funkce jsou uvedeny v následující tabulce:

Název	Datový typ	Funkce
ENDIAN	uint_8	Endian dat.
ALIGNMENT	uint_8	Zarovnání dat ve vzorku (vlevo – vpravo).
BITS	uint_8	Bitová hloubka dat.
SIZE	uint_8	Velikost dat v bajtech.
CHANNELS	uint_8	Počet kanálů

Tabulka 10: Proměnné struktury určující formát dat

Pro některé proměnné ENDIAN a ALIGNMENT jsou definovány makra, jejichž hodnoty mohou nabývat:

- ENDIAN: buďto AUDIO_BIG_ENDIAN nebo AUDIO_LITTLE_ENDIAN.
- ALIGNMENT: buď AUDIO_ALIGNMENT_RIGHT nebo AUDIO_ALIGNMENT_LEFT.

IO_IOCTL_AUDIO_GET_IO_DATA_FORMAT

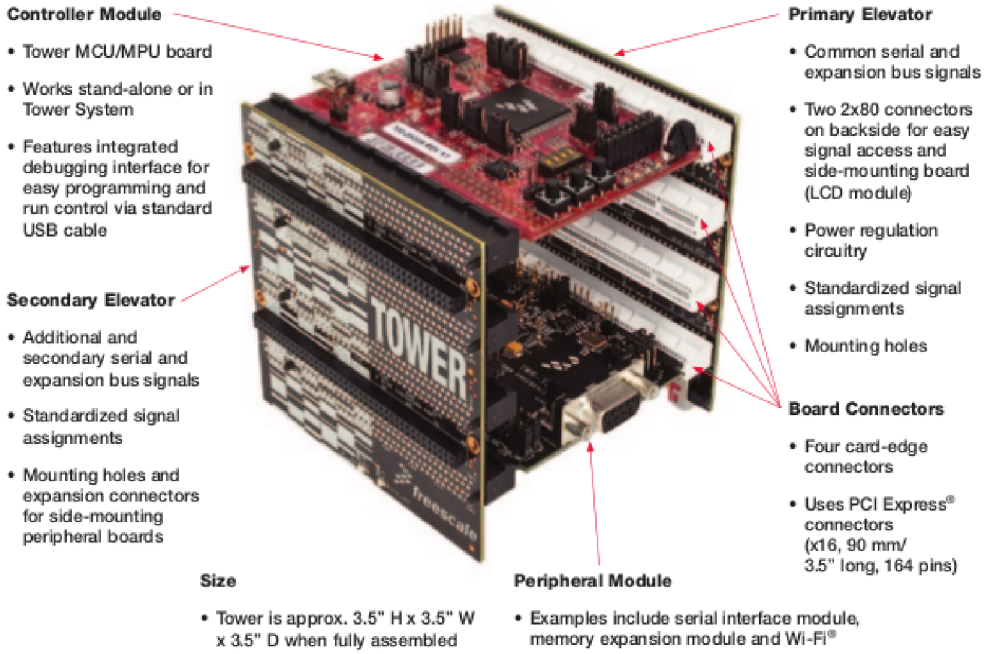
Je příkaz pro získání nastaveného formátu dat. K uložení formátu je použit ukazatel na strukturu typu `AUDIO_DATA_FORMAT`, který je příkazu předán jako parametr.

8.3 Vnitřní funkce ovladače

O veškeré úkony ovladače se starají funkce uvedené v tabulce č.7. Ovladač ke své funkci využívá dva softwarové kruhové buffery – jeden pro přijímací část ovladače a druhý pro část vysílací. Paměť bufferů je alokována při otevření zařízení a odalokována je při jeho zavření. V případě, že ovladač nevykonává žádnou činnost tak zbytečně nevytěžuje systémové prostředky. Data vstupující nebo vystupující z bufferů jsou upravena do požadovaného formátu funkcemi `FormatInData()` a `FormatOutData()`. Obě funkce provádí převod mezi endianitami, popřípadě zarovnávají bity v datovém slově dle potřeby. Detailní popis průběhu vykonávání jednotlivých nízkoúrovňových funkcí poskytují vývojové diagramy uvedené v přílohách práce.

9 FREESCALE TOWER SYSTÉM

Freescale Tower systém je modulární vývojová platforma pro 8mi, 16ti a 32bitové mikrokontroléry a mikroprocesory, která umožňuje pokročilý vývoj skrze rychlý návrh a vytvoření řešení. Obsahuje několik vývojových desek a modulů, díky kterým lze vytvářet uživatelské aplikace od těch nejjednodušších až po pokročilé.



Obrázek 11: Freescale Tower systém[15]

9.1 Použitý procesor

Pro vývoj hardware a software byl firmou Freescale zapůjčen systém obsahující procesor MCF54418. Jeho základní parametry jsou:

- Jádro ColdFire obsahující MMU a EMAC, frekvence jádra 250 MHz.
- 16 kB mezipaměti instrukcí a 16 kB mezipaměti dat.
- 64 kB interní SRAM paměti.
- Podpora bootování z SPI kompatibilní paměti flash.
- Podpora bootování z NAND paměti flash.

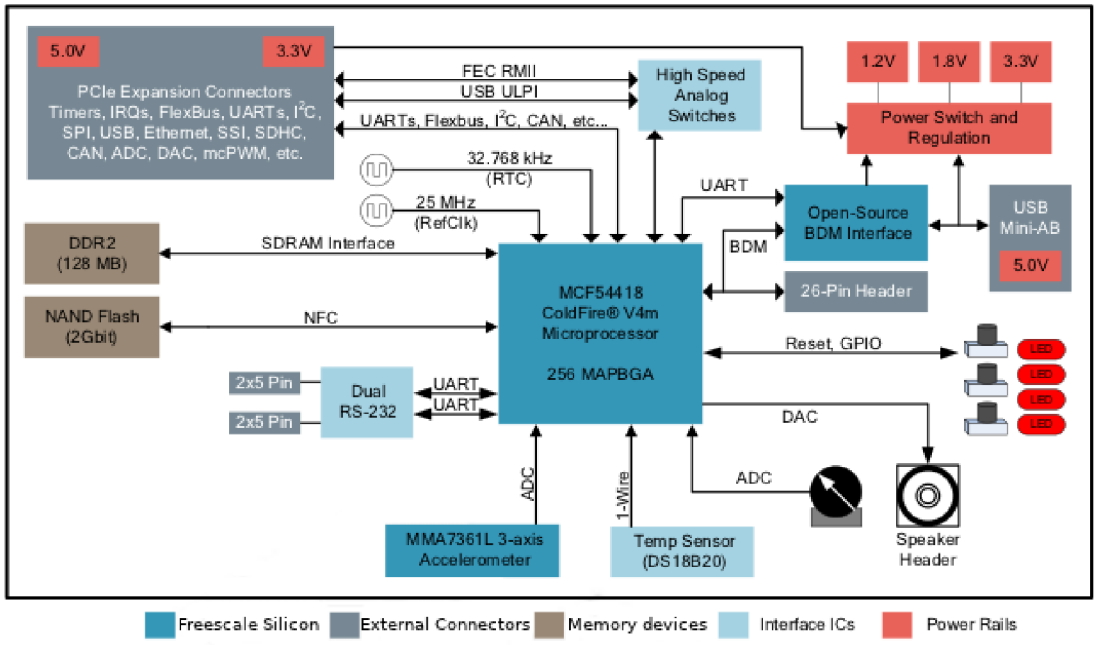
- Technologie crossbar switching (XBS) pro souběžný přístup k periferiím nebo paměti RAM z několika bus masterů.
- 64 kanálový DMA řadič.
- Řadič DDR1/DDR2.
- Řadič USB 2.0 On-the-Go s podporou ULPI.
- Dva porty pro čipové karty.
- Dva ethernetové řadiče řadiče pro rychlosti 10/100 Mbit/s.
- IEEE 1588-2002.
- Řadič typu host pro karty SDHC.
- Dva moduly CAN.
- Jednotka pro akceleraci kryptografie (CAU).
- Generátor náhodných čísel.
- Synchronní sériová linka (SSI).
- Čtyři 32bitové časovače s podporou DMA.
- Čtyři rozhraní pro sériové periferie s podporou DMA (DSPI).
- Deset portů typu UART.
- Šest rozhraní sběrnice I^2C.
- 12ti bitový analogově digitální převodník.
- Vícekanálové PWM.
- Dva digitálně analogové převodníky.
  Detailní popis architektury a funkcí tohoto procesoru lze nalézt v [4].

9.2 Modul TWR-MCF5441X

Tento modul obsahuje procesor MCF54418 a jeho periferie z nichž nejdůležitější jsou:

- DDR2 SDRAM 128 MB.
- Paměť flash typu NAND – 256 MB.
- Dvě sériové linky typu RS-232.
- Standardní 26ti pinový konektor BDM.
- Open source BDM postavené na obvodu MC9S08JM60.
- Standardní 6ti pinový konektor BKGD/MS.
- Tříosý akcelerometr tvořený obvodem MMA7361L.
- Digitální snímač teploty.
- Čtyři LED diody.

- Spínače a tlačítka pro uživatelský vstup.
- Potenciometr.
- Konektor pro připojení reproduktoru (výstup obvodu LM4889).



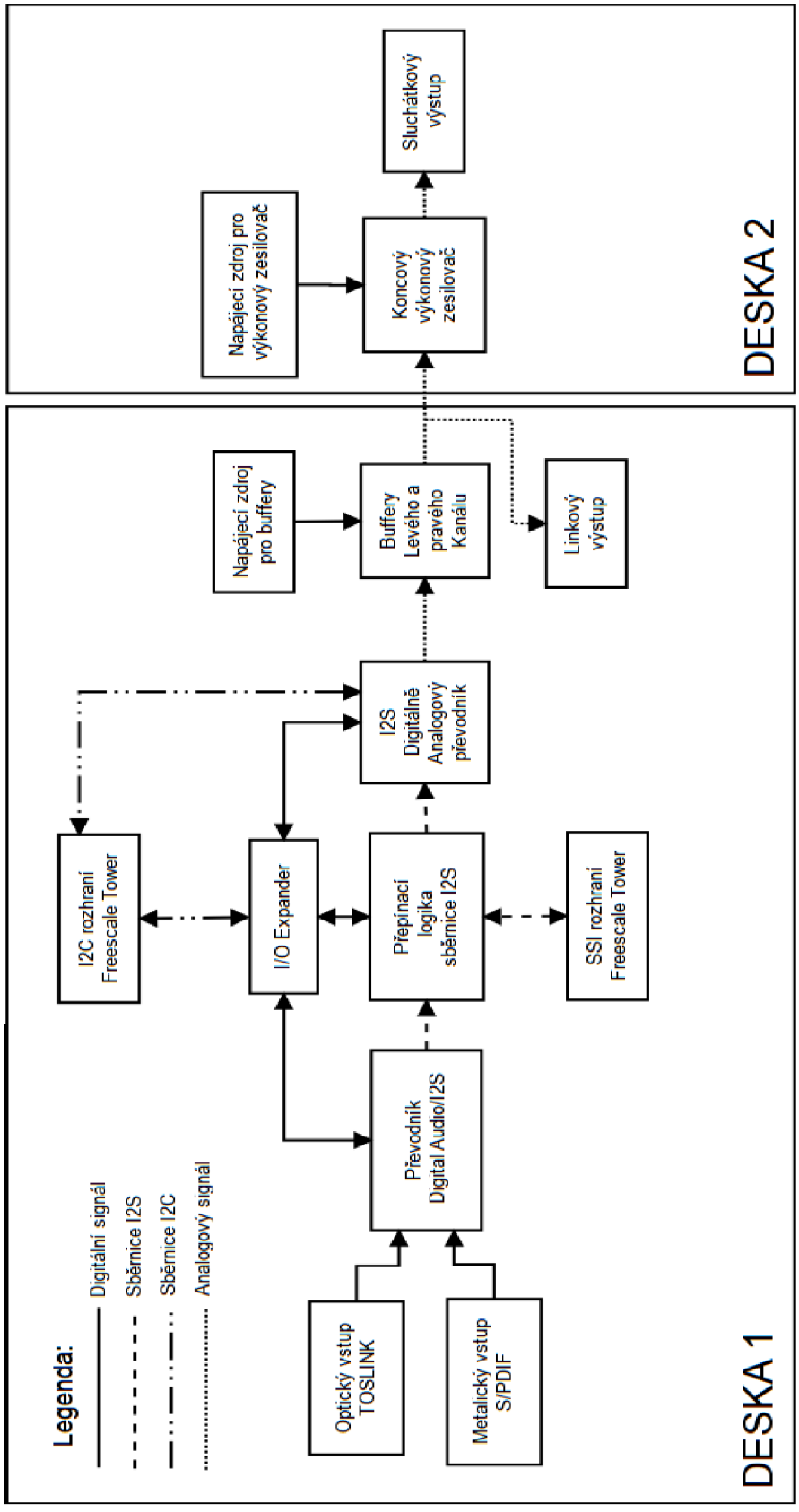
Obrázek 12: Blokové schéma modulu TWR-MCF5441X [15]

9.3 Modul TWR-SER2

Tento modul je periferní deskou a poskytuje systému Freescale Tower přídavná komunikační rozhraní. Tato deska obsahuje následující:

- Dva konektory RJ-45 s integrovanými oddělovacími transformátory a LED diodami poskytující fyzickou vrstvu pro rozhraní Ethernet.
- Vysokorychlostní fyzické rozhraní pro USB.
- Full speed/Low speed rozhraní Host USB využívající rozhraní integrovaného USB řadiče procesoru.
- Čtyři sériové linky:
  - Převodník sériová linka – USB s obvodem MC9S08JS1 na konektoru USB-mini-B.
  - Transceivery RS-232/RS-485 na sdíleném konektoru DB9.
  - Dva RS-232 transceivery s možností plného řízení datového toku na jumperech o velikosti 2x5 pinů.

10 POPIS ZAPOJENÍ



Obrázek 13: Blokové schéma navrženého zapojení

Celé zapojení je navrženo tak, aby bylo možno maximálně otestovat funkčnost navrhovaného ovladače. Proto je na desce zdroj signálu pro testování nahrávání na sběrnici I^2S, D/A převodník pro testování přehrávání po sběrnici I^2S, zdroj externího master clock signálu pro otestování funkce s vnitřním i vnějším zdrojem tohoto signálu a tedy funkčnosti ovladače jak v módu master tak i slave. Zapojení již prošlo první opravnou revizí, jejíž schéma lze nalézt v příloze č.1c.

10.1 Blok optického vstupu

Optický signál vstupuje do desky plošných spojů konektorem OX1 typu TOSLINK, jehož napájení je filtrováno pomocí LC filtru složeného ze součástek L1, C7 a C8. Poté převedený signál pokračuje na konektor JP1, který slouží k výběru zdroje signálu.

10.2 Blok metalického vstupu

Signál je na DPS přiveden pomocí konektoru X1 v provedení CINCH. Za konektorem je připojen rezistor R3 sloužící k impedančnímu přizpůsobení linky k němuž je paralelně připojen oddělovací transformátor TR1. Výstupní signál z transformátoru je poté převeden z diferenciálních úrovní na nesymetrický výstup pomocí obvodu IC3. Výstup tohoto obvodu je připojen na pin č.3 konektoru JP1.

10.3 Blok převodu digitálního signálu na signály sběrnice I^2S

O převod signálu na sběrnici I^2S se stará obvod DIR9001 fy Texas Instruments. Jedná se o 24 bitový převodník ze standartu S/PDIF pro vzorkovací frekvence do 96 kHz jehož detailní popis vnitřní funkce a parametry lze nalézt v [7]. Tento obvod vyžaduje ke svojí správné funkčnosti několik externích součástek. První z nich je obvod generování externího resetu IC1 připojený k pinu $\overline{\text{RST}}$ a dále filtr ze součástek C9, C5 a R3 pro vnitřní napětím řízený oscilátor připojen k pinu FILT. Funkce převodníku se nastavují pomocí pinů PSCK0, PSCK1, FMT0, FMT1. Tyto piny je možno nastavovat pomocí V/V expandéru IC11 a lze je tedy nastavit na úroveň log.0 nebo log.1 pomocí sběrnice I^2C. K expandéru jsou také připojeny pin ERROR a $\overline{\text{AUDIO}}$ pomocí nichž lze detekovat neplatný formát přijímaných dat.

Nastavení PSCK[1:0]		Výstupní frekvence fázového závěsu		
PSCK1	PSCK0	SCKO	BCKO	LRCKO
0	0	$128f_s$	$64f_s$	f_s
1	1	$256f_s$	$64f_s$	f_s
1	0	$384f_s$	$64f_s$	f_s
1	1	$512f_s$	$64f_s$	f_s

Tabulka 11: Nastavení frekvencí výstupních signálů [7]

LRCKO	BCKO	SCKO (záleží na nastavení PSCK[1:0])			
f_s	$64f_s$	$128f_s$	$256f_s$	$384f_s$	$512f_s$
32 kHz	2.048 MHz	4.096 MHz	8.192 MHz	12.288 MHz	16.384 MHz
44.1 kHz	2.8224 MHz	5.6448 MHz	11.2896 MHz	16.9344 MHz	22.5792 MHz
48 kHz	3.072 MHz	6.144 MHz	12.288 MHz	18.432 MHz	24.576 MHz
88.2 kHz	5.6448 MHz	11.2896 MHz	22.5792 MHz	33.8688 MHz	45.1584 MHz
96 kHz	6.144 MHz	12.288 MHz	24.576 MHz	36.864 MHz	49.152 MHz

Tabulka 12: Frekvence výstupních signálů pro různé vzorkovací frekvence f_s [7]

Nastavení FMT[1:0]		Formát výstupních sériových dat na pinu DOUT
FMT1	FMT0	
0	0	16 bitů, MSB první, zarovnáno vpravo
0	1	24 bitů, MSB první, zarovnáno vpravo
1	0	24 bitů, MSB první, zarovnáno vlevo
1	1	24 bitů, MSB první, formát I2S

Tabulka 13: Nastavení formátu výstupních dat v závislosti na pinech FMT[1:0][7]

Rozšíření V/V pinů základní desky systému Freescale Tower

Vzhledem k omezenému počtu V/V pinů procesoru MCF54418 umístěném na základní desce systému Freescale Tower je nutno jejich počet rozšířit pomocí tzv. V/V expandéru. Jako vhodný obvod pro tento úkol byl vybrán obvod PCA9555 od Texas Instruments. Jedná se o 16bitový paralelní expandér pro sběrnici I^2C. Detailní charakteristiky a popis lze nalézt v katalogovém listu [8].

10.4 Propojení sběrnic I²S a SSI

Procesor komunikuje s zařízeními v zapojení po sběrnici I²S pomocí rozhraní SSI. Tyto dvě rozhraní však nejsou stejná. Je tedy třeba definovat spojení jednotlivých vodičů těchto sběrnic a jejich funkci. To je provedeno následovně:

1. Signál SSI_MCLK je připojen k signálu master clock (ve schématu SCK) rozšiřující sběrnici I²S a externímu zdroji tohoto signálu (viz kapitola 10.12). To umožňuje otestování navrhovaného ovladače jak s interním, tak s externím zdrojem hodinového signálu.
2. Signál SSI_FS je připojen k signálu word select sběrnice I²S (ve schématu LRCK), je tedy využit k přepínání přenášeného audio kanálu a jeho frekvence odpovídá vzorkovací frekvenci přenášeného audio signálu.
3. Signál SSI_BCLK je připojen k signálu SCK sběrnice I²S a slouží k potvrzování dat na datovém vodiči DATA.
4. Signál SSI_RXD je připojen k výstupu převodníku S/PDIF, jelikož slouží pouze pro příjem signálu a převodník je v zapojení jeho jediným zdrojem.
5. Signál SSI_TXD je připojen k datovému vodiči sběrnice I²S (ve schématu DATA).

10.5 Blok přepínací logiky sběrnice I2S

Signály sběrnice I²S vystupující z převodníku DIR9001 je třeba přivést buď do DAC převodníku, na sběrnici SSI systému Freescale Tower nebo opačným směrem ze sběrnice SSI do D/A převodníku. K realizaci této funkčnosti slouží skupina integrovaných obvodů IC2, IC3 a IC8. Obvody IC2 a IC3 jsou jednocestné sběrnicové posilovače (buffery) s třístavovými výstupy typu 74LVHT125 (viz [13]), obvod IC2 je posilovač obousměrný typ CD74HCT243 (viz [9]) což procesoru umožňuje data ze sběrnice jak číst, tak zapisovat. Směrování signálů na sběrnici I2S je prováděno pomocí pinů I/O1[0:2] expandéru (ve schématu zapojení se jedná o signály $\overline{\rightarrow DAC}$, $\overline{SPDIF\rightarrow}$ a $\overline{DATA\_EN}$). K ovládání připojení procesoru (a tedy sběrnice SSI) je využito pinů I/O0[3:4] expandéru (ve schématu signály $\overline{TXC}$ a RXC).

I/O1[0:2]			I/O0[3:4]		Propojení signálů
IO1.0	IO1.1	IO1.2	$\overline{\text{TXC}}$	RXC	
0	0	0	1	0	S/PDIF spojeno s DAC, procesor odpojen
0	0	0	0	1	S/PDIF spojeno s DAC, procesor odpojen
0	1	1	0	0	Procesor spojen s DAC (procesor vysílá data)
1	0	1	1	1	S/PDIF spojen s procesorem (procesor přijímá data)
Ostatní					Neplatná kombinace

Tabulka 14: Možnosti nastavení propojení komponent na sběrnici I2S

10.6 Blok digitálně analogového převodníku

Převod signálu ze sběrnice I^2S zajišťuje obvod PCM1789 firmy Texas Instruments. Tento obvod je 24 bitový stereo D/A převodník pro audio techniku s diferenciálními výstupy využívající převodu metodou sigma-delta s podporou vzorkovacích frekvencí 8 kHz-192 kHz. Detailní popis a parametry obvodu lze nalézt v [12].

Nastavení a řízení převodníku je možno provádět třemi způsoby pomocí obvodu připojeného na pin MODE, k výběru sběrnice I^2C je třeba pin uzemnit. K obvodu jsou také připojeny řídící piny AMUTEI a AMUTEO. Pin AMUTEI slouží k aktivaci funkce MUTE (nastavení vstupního analogového signálu na nulovou úroveň), pin AMUTEO lze použít k jednoduché diagnostice vnitřního stavu obvodu (log.1 indikuje ztlumení obvodu). D/A převodník je napájen ze dvou zdrojů napájení: 3,3 V pro digitální část a 5 V pro část analogovou. Obvod ke své správné funkci nevyžaduje žádné další externí součástky kromě filtrů napájecího napětí.

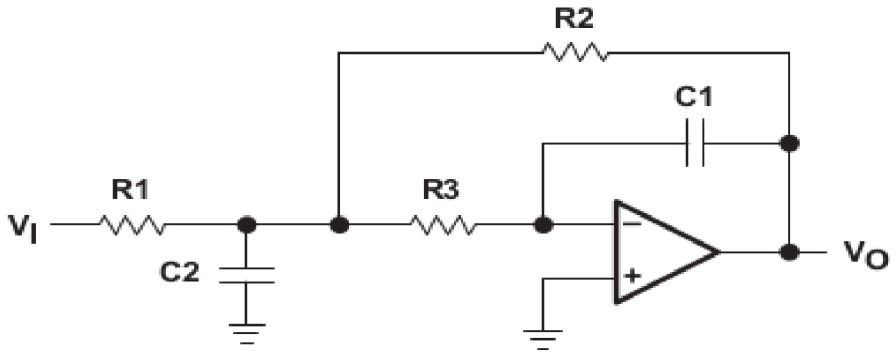
10.7 Bloky levého a pravého filtru/bufferu

D/A převodníky na principu sigma-delta využívají techniky k potlačení šumu na pracovních frekvencích. K tomu potlačení ovšem dochází za cenu generování šumu na vyšších frekvencích (nad Nyquistovou frekvencí nebo také $f_s/2$). Tento šum musí být odfiltrován pomocí dolnopropustních filtrů k zajištění optimálních vlastností převodu. Filtrování se odehrává z části v samotném D/A převodníku a z části je nutno šum filtrovat pomocí externích obvodů. K tomuto účelu a zároveň k převodu diferenciálních signálů vystupujících z převodníku slouží integrovaný obvod IC1. Jedná se o dvojitý operační zesilovač OPA2134 Burr-Brown s velmi nízkým zkreslením a šumem určený pro audio aplikace vyrobený technologií FET. Podrobné vlastnosti charakteristiky a parametry lze nalézt v [10].

Zapojení filtrů je koncipováno jako Butterworthovo druhého řádu s vícenásobnou zpětnou vazbou (topologie MFB) pro potlačení citlivosti k změnám hodnot součástek vlivem frekvence a teploty. Tento obvod je na obou kanálech zapojený jako diferenciální zesilovač se zesílením A=1 a zároveň filtr typu dolní propust s dělícím kmitočtem f = 53 kHz. Tuto frekvenci lze vypočítat podle [21] jako:

$$f = \frac{1}{2\pi\sqrt{R2R3C1C2}} \quad (4)$$

kde hodnoty odpovídají následujícímu zapojení:



Obrázek 14: Schéma pro výpočet dělící frekvence filtru[21]

po dosazení hodnot ze schématu zapojení do vztahu (4):

$$f = \frac{1}{2\pi\sqrt{11\cdot 10^{3}\cdot 820\cdot 330\cdot 10^{-12}\cdot 3\cdot 10^{-9}}} = 53259\ Hz$$

Výstupní signály jsou přivedeny na konektory JP2 a SV1. Pomocí těchto konektorů je možno signál rovnou odebírat pro další zpracování (JP2) nebo ho přivést vstup výkonového modulu (SV1). Obvod je napájen napětím ±10 V z kapacitního násobiče a invertoru složeného z obvodů IC5 a IC6.

10.8 Blok napájecího zdroje filtrů/bufferů

Z důvodu zajištění co nejnižšího rušení způsobeného napájecím zdrojem je pro napájení obvodů aktivních filtrů využit oddělený napájecí zdroj tvořený nábojovou pumpou složenou z obvodů IC5 a IC6 (MAX865). Oba tyto obvody jsou DC-DC měniče na principu nábojové pumpy s dvojitým kladným i záporným výstupem. Jejich napájecí napětí se může pohybovat v rozsahu 1,5-6 V, spínací frekvence obvodu se pohybuje v rozmezí 20-38 kHz a zkratový výstupní proud je 100 mA. Detailní popis funkce a charakteristiky lze nalézt v [20]. Obvody jsou zapojeny paralelně pro zvýšení

dodávaného proudu dle zapojení výrobce. Zvlnění výstupního napětí lze vypočítat dle vztahů uvedených v literatuře [20] jako:

$$V_{RIPPLE} = \frac{I_{LOAD}}{2 f_{PUMP} C_{RESERVOIR}} \quad (5)$$

kde:

V_{RIPPLE} = Zvlnění výstupního napětí na kladném (V+) nebo záporném (V-) výstupním terminálu.

I_{LOAD} = Proud dodávaný do zátěže. Pro kladnou napájecí větev jde o součet proudů zátěže (I_{V+}) a proudu do nábojové pumpy (I_{V-}).

f_{PUMP} = Frekvence spínání nábojové pumpy (typicky 30 kHz).

$C_{RESERVOIR}$ = Kapacita kondenzátoru sloužícího k uskladnění energie (100 μF).

Výstupní proud lze zjistit z výstupní impedance bufferu a tedy operačního zesilovače, vstupní impedance připojeného obvodu (výkonového zesilovače) a napájecího napětí (výstupní napětí nábojové pumpy). V případě připojení výkonového zesilovače tedy:

$$I_{LOAD} = I_{OUT} = \frac{V_{SUP}}{R_{OUT} + R_{IN(POWER)}} \quad (6)$$

kde:

I_q = Napájecí proud OZ tvořícího buffer.

I_{OUT} = Výstupní proud bufferu.

V_{SUP} = Napájecí napětí bufferu/Výstupní napětí měniče.

R_{OUT} = Výstupní odpor bufferu, je dán výstupní impedancí OZ.

$R_{IN(POWER)}$ = Vstupní odpor výkonového zesilovače.

Jedinou neznámou je zde vstupní odpor výkonového zesilovače, který je třeba vypočítat dle hodnot ze schématu zapojení a katalogové hodnoty vstupního odporu jako:

$$R_{IN(POWER)} = R5 + R8 \| R_{IN} \quad (7)$$

Po dosazení:

$$R_{IN(POWER)} = 1 \cdot 10^3 + \frac{3{,}3 \cdot 10^3 \cdot 300 \cdot 10^3}{3{,}3 \cdot 10^3 + 300 \cdot 10^3} = 4{,}26\, k\Omega$$

Po dosazení do vztahu (6):

$$I_{LOAD} = \frac{20}{0{,}01 + 4{,}26 \cdot 10^3} = 4{,}7\, mA$$

Po dosazení do vztahu (5) tedy:

$$V_{RIPPLE} = \frac{2 \cdot 4{,}7 \cdot 10^{-3}}{2 \cdot 30 \cdot 10^3 \cdot 100 \cdot 10^{-6}} = 1{,}6\, mV$$

10.9 Blok výkonového zesilovače

Pro účely poslechu výstupního signálu D/A převodníku je navrhované zapojení vybaveno koncovým výkonovým zesilovačem pro sluchátka TPA6120A2. Jedná se o Hi-Fi stereo audio zesilovač využívající proudové zpětné vazby s dynamickým rozsahem 120 dB. Podrobné parametry a charakteristiky lze nalézt v [14].

Obvod je zapojen jako neinvertující zesilovač se zesílením *A=3,33* díky kterému je pak maximální výstupní napětí $U_{out} \approx 26{,}4$ V. Toto výstupní napětí je poté přiváděno na konektor X1 typu JACK o velikosti 3,5 mm ze kterého může být odebírán výkon do zátěže maximálně 2,5 W při impedanci minimálně 8 Ω. Z důvodu vyššího výkonu a tím i vyšších nároků na chlazení i napájecí napětí a proud je tento obvod včetně jeho napájecího zdroje umístěn na samostatné desce plošných spojů a chlazen pomocí plochy mědi po obou stranách DPS. Ztrátový výkon obvodu lze vypočítat s využitím účinnosti dle následujících vztahů:

$$\eta = \frac{P_L}{P_{SUP}} \quad (8)$$

kde

$$P_L = \frac{V_{LRMS}^2}{R_L} \text{ , a } V_{LRMS} = \frac{V_P}{\sqrt{2}} \text{ a tedy } P_L = \frac{V_P^2}{2R_L} \text{ pro jeden kanál (9)}$$

$$P_{SUP} = V_{CC}\,\overline{I_{CC}} + V_{CC}\,I_{CC(q)} \quad (10)$$

$$\overline{I_{CC}} = \frac{1}{\pi}\int_0^{2\pi} \frac{V_P}{R_L}\sin(t)\,dt = \frac{-V_P}{\pi R_L}\left[\cos(t)\right]_0^{\frac{\pi}{2}} = \frac{V_P}{\pi R_L} \quad (11)$$

kde

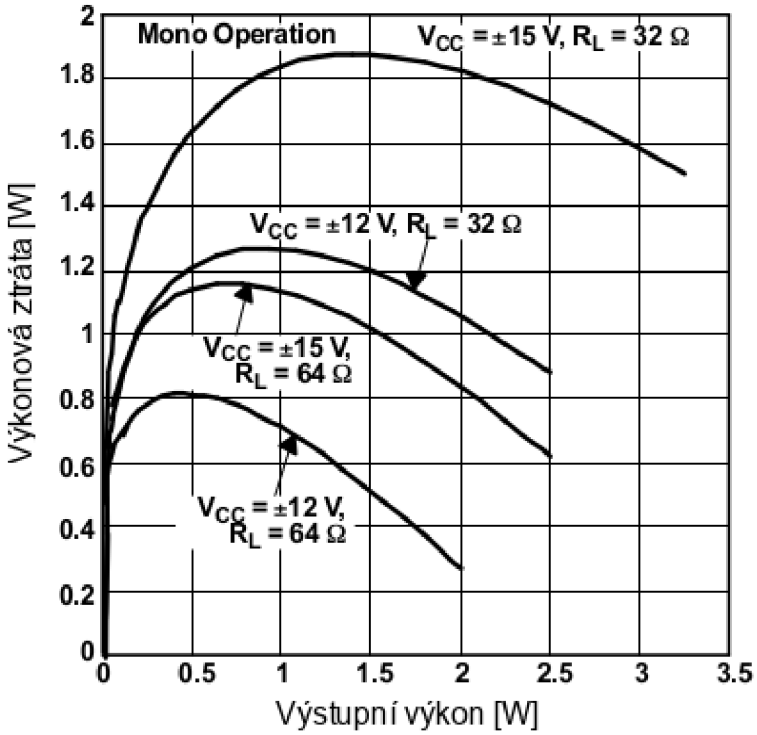
$$V_P = \sqrt{2P_L R_L} \text{ a tedy } P_{SUP} = \frac{V_{CC}V_P}{\pi R_L} + V_{CC}\,I_{CC(q)} \quad (12,13)$$

kde:

P_L = Výkon dodávaný do zátěže (pro jeden kanál).
P_{SUP} = Výkon odebíraný ze zdroje napájení.
V_{LRMS} = Efektivní napětí na zátěži.
R_L = Odpor zátěže.
V_P = Špičkové napětí na zátěži.
$\overline{I_{CC}}$ = Průměrná hodnota proudu odebíraného ze zdroje.
$I_{CC(q)}$ = Klidový proud obvodu (pro jeden kanál).
V_{CC} = Rozsah napájecího napětí zdroje (30 V v případě napájení ±15 V).
η = Efektivita zesilovače.

Efektivita se pro případ funkce ve stereo režimu nezmění, protože dojde k zdvojnásobení P_L a P_{SUP}, zdvojnásobí se však výkonová ztráta zesilovače. Ztrátový výkon lze vypočíst dle následujícího vztahu:

$$P_{DISS} = (1-\eta) P_{SUP} \quad (14)$$



Obrázek 15: Výkonová ztráta zesilovače v závislosti na výstupním výkonu pro zvolené typické zátěže a napájecí napětí[14]

Vzhledem ke skutečnosti že obvod pracuje jako proudový operační zesilovač, je třeba věnovat velkou pozornost volbě zpětnovazebních rezistorů. Se snižováním hodnoty odporu rezistoru dochází ke snižování šumu zanášeného do obvodu, avšak nižší hodnota také posunuje dominantní pól k vyšším frekvencím a tím dochází ke snižování stability zapojení. Naopak vyšší hodnota odporu zanáší do obvodu větší šum, ale stahuje dominantní pól směrem k nižším frekvencím a tím zvyšuje stabilitu. Obdobným problémem se může stát připojená zátěž s vysokou impedancí, která snižuje stabilitu. Tu lze zvýšit zvětšením hodnoty zpětnovazebního odporu, není však podle literatury [14] doporučeno přesáhnout 10 kΩ. Výstupní stereo signál je možno odebírat z konektoru X1 typu JACK.

10.10 Blok napájecího zdroje výkonového zesilovače

Tento blok zajišťuje výkonovému zesilovači potřebné napájecí napětí ±15 V. Jádrem napájecího zdroje je DC-DC měnič IR0515S jehož maximální výstupní proud je 100 mA pro každé z výstupních napětí a napájecí napětí 5 V. Tento obvod byl zvolen na základě několika požadavků:

- Celkový dodávaný výkon do zátěže alespoň 2 W (požadavek zadání práce na vybuzení sluchátek, případně malých reproduktorů).
- Napájecí napětí 5 V (je již v zapojení přítomno).
- Výstupní napájení 15 V (jeví se jako ideální pro napájení obvodu výkonového zesilovače).

Napájecí napětí je k měniči přivedeno přes LC filtr složený ze součástek C9 a L2, který je určen k potlačení zpětného šíření rušení vznikajícího spínáním měniče a může dosahovat velmi vysokých hodnot (až 50 mV špička-špička při šířce pásma 20 MHz [19]). Dělící frekvenci je vhodné zvolit alespoň o řád nižší než je spodní hranice spínací frekvence měniče (~60 kHz) a lze ji vypočítat jako [18]:

$$f = \frac{1}{2\pi\sqrt{LC}} \quad (15)$$

a po dosazení:

$$\frac{1}{2\pi\sqrt{1.10^{-6}680.10^{-6}}} = 6103\,Hz$$

Za filtrem jsou umístěny kondenzátory C10 a C11 sloužící jako lokální zásobník energie. Tyto kondenzátory jsou elektrolytické tantalové a řazeny paralelně pro zajištění co nejnižšího ESR (ekvivalentní sériový odpor).

Na výstupech jsou opět připojen filtry pro každou napájecí větev ze součástek L1 a C6 resp. L2 a C13 za kterými jsou zásobníky energie pro zesilovač složené z dvojic kondenzátorů C7, C8 a C14, C15. Hodnoty těchto součástek jsou zvoleny s ohledem na maximální kapacitní zátěž měniče, která činí 47 μF [19].

10.11 Blok napájecího napětí a ochran

Tento blok je na desce zařazen z důvodu nedostatečného výkonu napájecího napětí v systému Freescale Tower. Ten je totiž napájen pomocí rozhraní USB, které poskytuje proud pouze nejvýše 500 mA při napájecím napětí 5 V.

Z vnějšího zdroje je napájecí napětí o velikosti 5 V je do obvodu přiváděno přes konektor J1. Za tímto konektorem se nachází tavná přístrojová pojistka F1 pro proudy

do 0,5 A. Paralelně k pojistce a napájecímu konektoru je připojen transil D2 s průrazným napětím 6,4 V, který slouží jako ochrana proti přepětí a přepólování napájecího napětí. Za transilem je pak LED dioda s předřadným odporem sloužící k indikaci připojení napájecího napětí a lineární stabilizátor IC12 pro převod vstupního napětí na 3,3 V.

10.12 Blok zdroje hodinového kmitočtu pro sběrnici I^2S

Z důvodu neexistence přesného zdroje hodinového kmitočtu v procesoru MCF54418, který by byl vhodný jako master clock na sběrnici I^2S je třeba takový zdroj vytvořit a přidat jej do zapojení.

Pro tento účel byl vybrán obvod PLL1705 (ve schématu IC9) výrobce Texas Instruments. Jedná se o integrovaný obvod fázového závěsu s vícenásobným výstupem a paralelní konfigurací. Tento obvod je schopen generovat na svých výstupech násobky vzorkovací frekvence ze vstupu o frekvenci 27 MHz nebo připojeného krystalu. Jeho detailní charakteristiky a parametry lze nalézt v literatuře [22].

Obvod ke své správné funkci potřebuje minimum vnějších součástek. Jde zejména o filtry napájecího napětí (C22, C24, C25, C27, C29 a C30) a dále pak krystal Q1. Výstupní hodinový signál je odebírán z pinu SCKO2, který poskytuje frekvenci $256f_S$. Volba příslušné vzorkovací frekvence probíhá pomocí pinů FS1, FS2 a SR, které jsou připojeny k I/O expandéru IC11. Volba požadované frekvence tedy probíhá pomocí sběrnice I^2C dle následující tabulky:

FS1	FS2	SR	Vzorkovací frekvence [kHz]	Násobení 2krát	SCKO2 [MHz]
0	0	0	48	Ne	12,288
0	1	0	44,1	Ne	11,2896
1	0	0	32	Ne	8,192
1	1	X	rezervováno	rezervováno	rezervováno
0	0	1	96	Ano	24,576
0	1	1	88,2	Ano	22,5792
1	0	1	64	Ano	16,384

Tabulka 15: Nastavení výstupní frekvence fázového závěsu

Výstupní signál je dále přiveden přes hradlo IC7 (jednolinkový buffer s vysokoimpedančním výstupem) na pin SSI0_MCLK/CLKIN systému Freescale Tower. Hradlo je ovládáno pomocí signálu $\overline{\text{EXT\_MCLK\_EN}}$, který je připojen k I/O expandéru IC11.

11 POPIS DESEK PLOŠNÝCH SPOJŮ

Celé zapojení je rozděleno na dvě desky plošných spojů z důvodu potlačení rušení vytvářeného DC-DC měničem a lepší možnosti chlazení pro výkonový zesilovač. Tato koncepce také umožňuje výrazné snížení spotřeby celého zapojení v případě že není potřeba využívat výkonový výstup. V práci je uvedena druhá verze návrhu po úpravách provedených po konzultaci s oddělením pro návrh hardware ve společnosti Freescale. Desky jsou pojmenovány AUDIO a POWER. Na desce AUDIO jsou umístěny následující bloky:

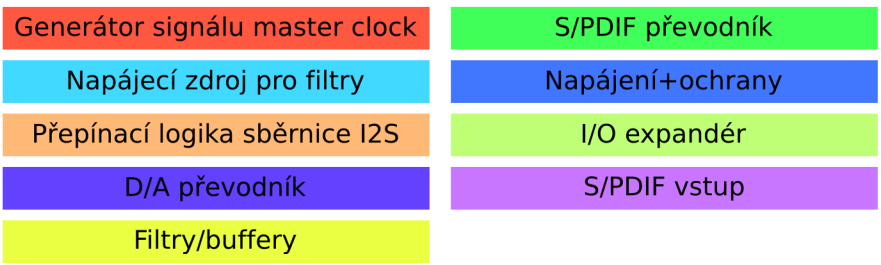
- Vstupní konektory pro signál a napájecí napětí.
- Zdroj 3,3 V pro digitální obvody.
- Zdroj ±10 V pro filtry/buffery.
- Převodník S/PDIF na I^2S.
- Přepínací logika sběrnice I^2S.
- D/A převodník.
- Generátor signálu master clock pro sběrnici I^2S.

Na desce POWER jsou bloky:

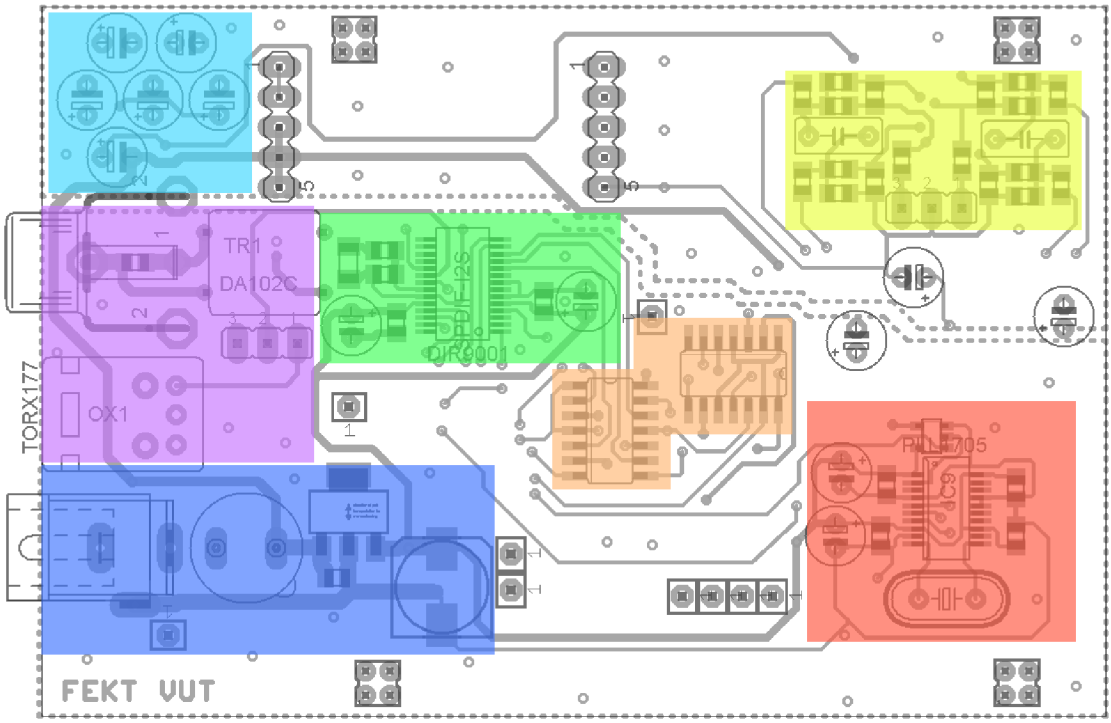
- Výkonový zesilovač.
- DC-DC měnič s filtry pro napájení výkonového zesilovače.

11.1 Deska AUDIO

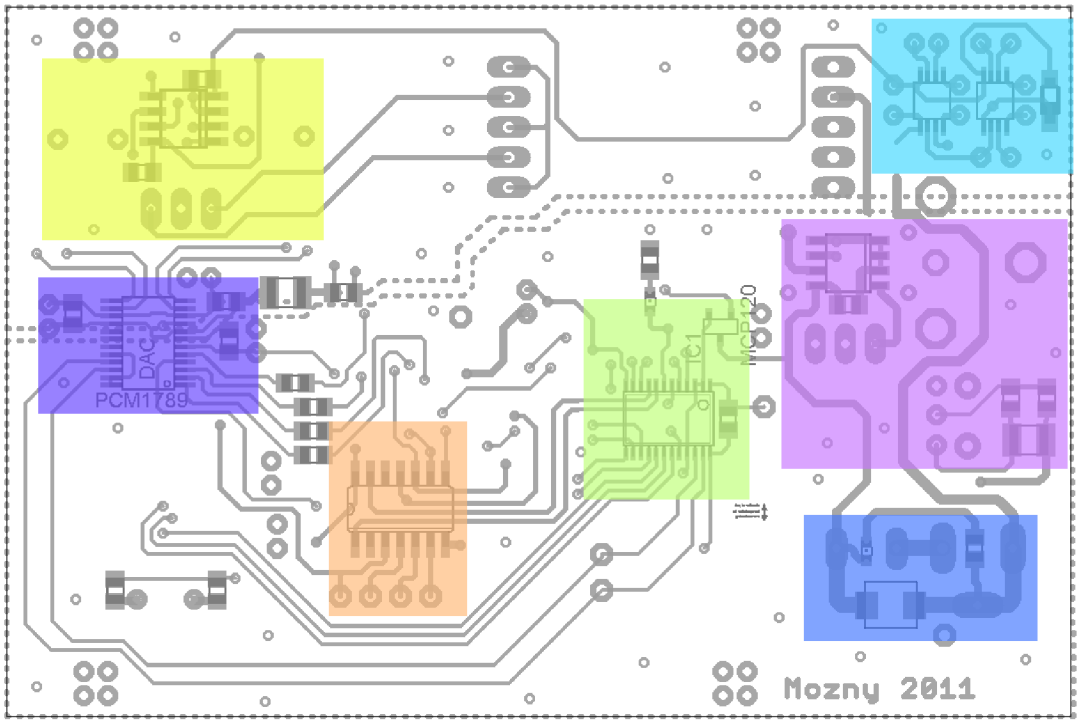
Jedná se o oboustrannou prokovenou desku plošných spojů. Je rozdělena na dvě části pomocí odděleného zemnění analogové a digitální části desky (na obrázcích je rozdělení naznačeno čárkovanou čarou). Toto rozdělení bylo zvoleno pro zamezení pronikání rušení, které vytváří sběrnice I^2S a digitální obvody do citlivých vstupních obvodů D/A převodníku a filtrů. V analogové části (horní část desky) se nachází při pohledu shora ve směru zleva doprava: zdroj pro filtry/buffery, konektory pro připojení výkonového modulu, filtry/buffery a pod nimi D/A převodník (ze spodní strany viz obr. 17). V digitální části desky se nachází ve stejném směru konektory CINCH a TOSLINK pro připojení signálu S/PDIF, převodník S/PDIF na sběrnici I^2S, přepínací logika sběrnice



I²S, generátor signálu master clock a vlevo dole pak vstup a zdroj napájecího napětí. Barevné značení na obrázcích odpovídá následující legendě:



Obrázek 16: Rozmístění bloků na desce AUDIO - pohled z horní strany

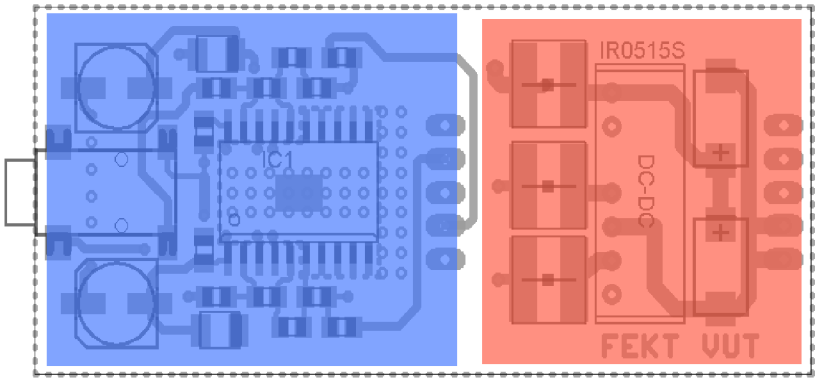


Obrázek 17: Rozmístění bloků na desce AUDIO - pohled ze spodní strany

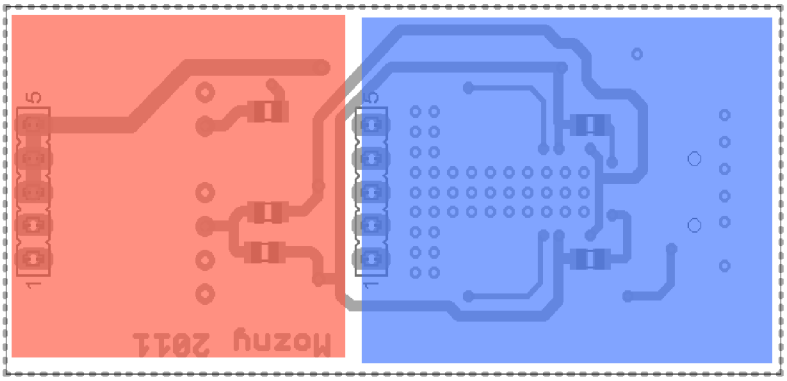
11.2 Deska POWER

Stejně jako v případě desky AUDIO se jedná o oboustrannou DPS. Tato deska je vytvořena jako výkonový modul, který je možno v případě potřeby připojit k desce AUDIO. Je napájena napětím 5 V z konektoru SV2, signál je přiváděn z filtrů na desce AUDIO pomocí konektoru SV1. Je rozdělena na dvě hlavní části: blok DC-DC měniče a blok výkonového zesilovače.

Veškeré součástky připojené k výkonovému zesilovači (IC1 ve schématu 2) jsou podle doporučení výrobce připojeny co nejkratšími cestami pro potlačení vzniku oscilací. Pro zajištění lepšího odvodu tepla je pouzdro zesilovače připájeno chladící ploškou k co největší ploše mědi s velkým množstvím prokovených otvorů (viz [14]) tyto otvory slouží k lepšímu prostupu odváděného tepla na druhou stranu DPS. Deska má také maximální možnou plochu mědi pro co největší snížení indukčnosti zemnění. Pokud by však byl zemnící vodič i pod součástkami vstupů a zpětné vazby znamenalo by to zvýšení kapacity na vstupních pinech zesilovače a tím i rizika vzniku oscilací. Proto je v okolí vstupních pinů a bezprostředně pod součástkami na tyto piny připojenými zemnění odstraněno. Rozložení bloků na DPS je vidět na následujících obrázcích, kde modrá barva značí výkonový zesilovač a červená napájecí zdroj.



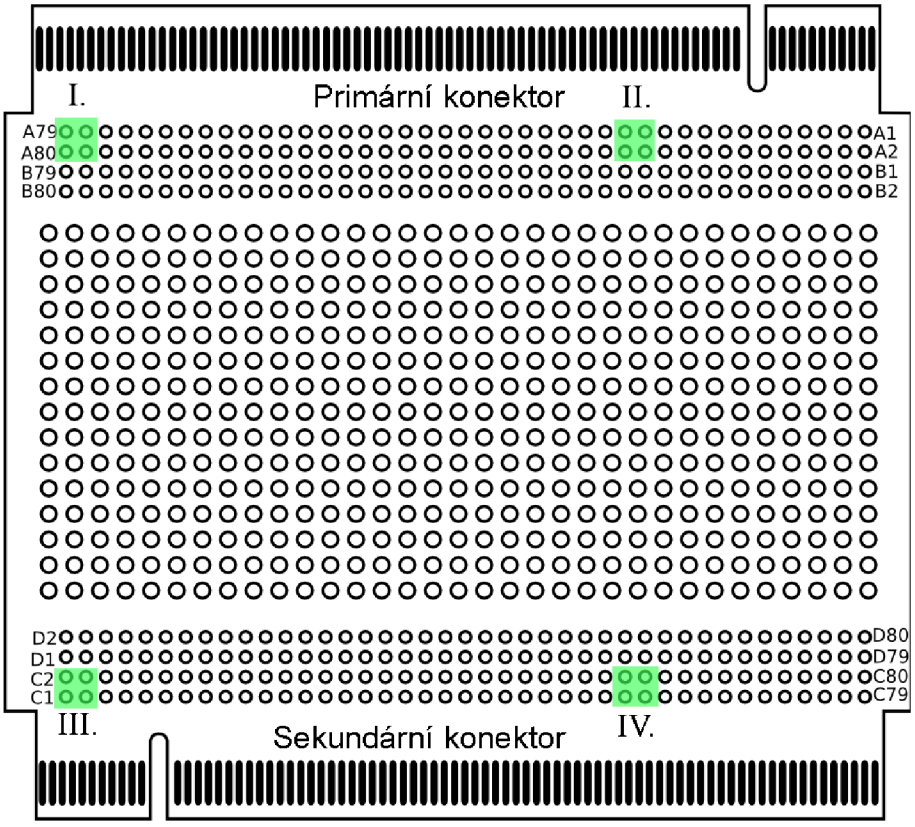
Obrázek 18: Rozmístění bloků na desce POWER - pohled z horní strany



Obrázek 19: Rozmístění bloků na desce POWER - pohled ze spodní strany

11.3 Připojení k systému Freescale Tower

K připojení do systému Freescale Tower je využita prototypová deska poskytovaná pod názvem TWR-PROTO. Modul umožňuje pomocí primárního a sekundárního konektoru přístup ke všem signálům v systému včetně napájecích napětí 5 V a 3,3 V. K návrhu zařízení je možno využít uživatelskou plochu. Ta je použita pro připojení navržených DPS tak, že deska AUDIO se zasune pomocí čtyř konektorů umístěných po jejím obvodu (na obr. č.21 vyznačeno červeně) do odpovídajících zdířek (na obr. č.20 vyznačeno zeleně) na desce TWR-PROTO.



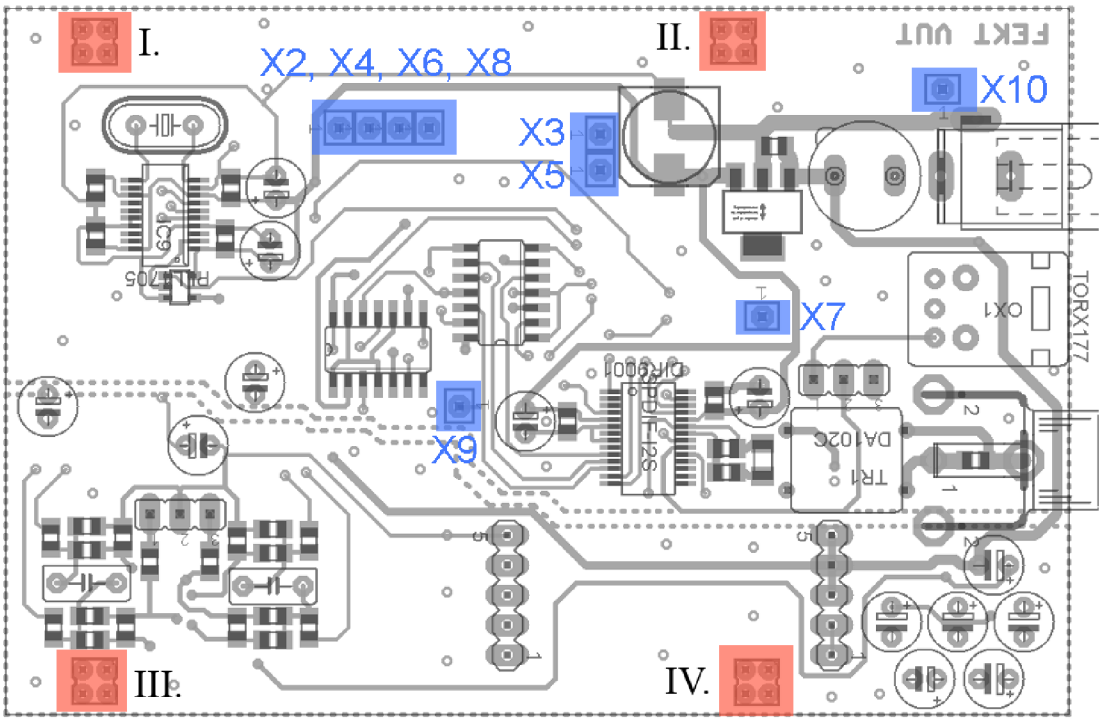
Obrázek 20: Schématické znázornění desky TWR-PROTO

Dále je třeba připojit signálové vodiče sběrnice SSI a I^2C dle následují tabulky (na obr. č. 21 vyznačeno modře):

Název signálu	Pin na TWR-PROTO	Pin na desce AUDIO
SCL	A7	X5
SDA	A8	X3
SSI_MCLK/CLKIN	A21	X2
SSI_FS	A23	X4

Název signálu	Pin na TWR-PROTO	Pin na desce AUDIO
SSI_BCLK	A22	X6
SSI_TXD	A25	X8
SSI_RXD	A24	X9
$\overline{\text{INT}}$	A35	X7
GND	A2	X10

Tabulka 16: Propojení signálů desky AUDIO a TWR-PROTO



Obrázek 21: Deska AUDIO se zvýrazněnými konektory pro připojení k Freescale Tower

12 TESTOVACÍ APLIKACE

Pro účely otestování funkčnosti ovladače byl vytvořen ukázkový projekt, který testuje základní operace s ovladačem jako jsou nahrávání, přehrávání a nastavování jeho jednotlivých parametrů. Celý projekt se skládá z několika zdrojových a hlavičkových souborů, které je možno rozdělit na několik podskupin: soubory shellu, soubory pro práci s rozhraním I^2C a soubory vláken. Jejich přehled je uveden v následující tabulce:

Název souboru	Obsah souboru
i2c_common.h	Makra a prototypy funkcí pro obecné I^2C zařízení.
i2c_common.c	Funkce pro práci s obecnými zařízeními na sběrnici I^2C.
log_messages.h	Prototypy funkcí a makra pro ladící zprávy.
log_messages.c	Funkce pro tisk ladících zpráv s časovými značkami.
main.h	Prototypy vláken.
main.c	Zdrojový kód jednotlivých vláken.
pca9555.h	Makra pro práci s obvodem PCA9555.
pcm1789.h	Makra pro práci s obvodem PCM1789.
sh_audio.h	Makra, struktury a prototypy funkcí pro rozšíření shellu.
sh_audio.c	Zdrojový kód pro funkce rozšiřující shell.
wav.h	Makra, struktury a prototypy funkcí pro práci se soubory WAV.
wav.c	Zdrojový kód funkcí pro práci se soubory WAV.

Tabulka 17: Soubory testovací aplikace

12.1 Funkce a struktura aplikace

Testovací aplikace využívá pro interakci s uživatelem příkazový řádek (shell) systému MQX, jehož funkčnost rozšiřuje o příkazy `play` a `record` pro přehrávání a nahrávání audio souborů ve formátu WAV jehož struktura a popis jsou uvedeny v [23]. Požadovaný soubor pro přehrávání resp. nahrávání je čten nebo ukládán na paměťovou kartu typu SD se souborovým systémem FAT ve verzi FAT12, FAT16 nebo FAT32.

Aplikace sestává ze čtyř úloh: `Init_task`, `Sdcard_task`, `Shell_task` a `Sdcard_write_task`. Pro přepínání kontextu není využita žádná systémová forma plánování úloh. Místo toho přepínání probíhá v případech, že některá z úloh čeká na některý ze synchronizačních objektů nebo je aktivní úloha přepnuta do zablokovaného stavu (např. pomocí funkce `_task_block()`). Jednotlivé úlohy mají různou prioritu, která je jim přiřazena při startu systému.

12.2 Úloha Init_task

Tato úloha je v aplikaci jako jediná vytvořena s parametrem MQX_AUTO_START_TASK a je tedy spuštěna automaticky po spuštění jádra MQX. Jejím úkolem je nastavit systémový čas, inicializovat potřebné periferie (paměťová karta a testovací deska) a vytvoření ostatních úloh. O průběhu jednotlivých operací je uživatel informován pomocí hlášení v konzole tak, jak je vidět na následujícím příkladu:

```
Setting up time.....................[OK]
Initializing audio board.............[FAIL]
  Error 0x82
Opening ESDHC channel................[OK]
Installing SD card...................[OK]
Creating SD card task................[OK]
  SD card installed to "a:"
Creating shell task..................[OK]

I2S driver test application

Shell (build: Mar 28 2012)
Copyright (c) 2008 Freescale Semiconductor;
shell>
shell>
```

Jak je v příkladu vidět, došlo k chybě při inicializaci audio desky a byl vypsán příslušný chybový kód, který v tomto případě odpovídá nepřítomnosti desky v zapojení. Dále je také vidět, že došlo k úspěšné instalaci paměťové karty a její souborový systém je připojen v umístění „a:“. Obecně může dojít k několika chybám, jejichž přehled včetně chybových kódů je uveden v následující tabulce (tato tabulka neobsahuje chybové kódy z MQX ty lze najít v příslušné literatuře [6] a [16]):

Chybový kód (hexadecimálně)	Příčina
0x0000DEAD	Nelze otevřít ovladač sběrnice I^2S. Pravděpodobně není nainstalovaný.
0xFFFFFFFF	Chybný zápis nebo čtení na sběrnici I^2C. Počet odeslaných resp. přijatých bajtů neodpovídá požadavku aplikace.
0x00000081	Nelze otevřít ovladač sběrnice I^2C. Pravděpodobně není nainstalovaný.

Chybový kód (hexadecimálně)	Příčina
`0x00000082`	Zařízení s požadovanou adresou není na sběrnici přítomno. Může se jednat o špatně zadanou adresu zařízení, případně nepřipojenou sběrnici I^2C.

Tabulka 18: Chybové kódy úlohy Init_task a jejich příčiny

Vývojový diagram úlohy popisující její funkci lze nalézt v příloze č.4a.

12.3 Úloha Sdcard_task

Úkolem této úlohy je zajistit instalaci a odinstalaci paměťové karty, správce oddílů a souborového systému. Tyto operace jsou prováděny v nekonečné smyčce s čekáním, což umožňuje provádět vložení a vyjmutí paměťového média za běhu a přepínání kontextu pro další úlohy. V této smyčce je postupně provedeno následující:

1. Zjištění stavu karty (zda je vložena a jestli je nebo není pouze ke čtení).
2. Instalace správce oddílů a jeho otevření.
3. Inicializace souborového systému na prvním oddílu paměťové karty.
4. Otevření souborového systému.
5. Ověření typu souborového systému.

Pokud je karta za běhu aplikace vyjmuta, jsou v opačném pořadí provedeny opačné operace (uzavření a odinstalace). O vložení a vyjmutí karty nebo případných chybách při instalaci nebo odinstalaci jednotlivých komponent je uživatel informován výpisem zpráv v příkazovém řádku včetně příslušného chybového kódu navráceného systémem (tyto kódy lze nalézt v literatuře [24]). Vývojový diagram úlohy je v příloze č.4C.

12.4 Úloha Shell_task

Tato úloha obstarává spuštění příkazového řádku na rozhraní definovaném pomocí makra `BSP_DEFAULT_IO_CHANNEL` v konfiguračním souboru systému *user_config.h*. Z důvodu snadného zobrazení a komunikace s aplikací je zvoleno rozhraní „ttyd:“, které odpovídá sériovému rozhraní RS-232 vyvedenému na komunikační desku TWR-SER2. Parametry spojení jsou následující:

- rychlost komunikace: 115200 baud
- parita: žádná
- datové bity: 8

- handshaking: Xon/Xoff

Samotný shell je spuštěn pomocí funkce `Shell()`, které je jako parametr předána konstantní struktura obsahující požadované příkazy a jim odpovídající volané funkce. Přehled použitých příkazů v shellu aplikace je uveden v tabulce:

Příkaz	Volaná funkce	Popis
cd	Shell_cd	Příkaz pro změnu adresáře.
del	Shell_del	Příkaz pro smazání souboru/složky.
dir	Shell_dir	Příkaz pro výpis obsahu adresáře.
exit	Shell_exit	Příkaz pro ukončení shellu.
help	Shell_help	Příkaz pro výpis nápovědy.
mkdir	Shell_mkdir	Příkaz pro vytvoření adresáře.
ren	Shell_ren	Příkaz pro přejmenování soubor/adresáře.
rmdir	Shell_rmdir	Příkaz pro smazání souboru/adresáře.
play	Shell_play	Příkaz pro přehrání souboru WAV.
record	Shell_record	Příkaz pro záznam souboru WAV.
?	Shell_command_list	Příkaz pro výpis všech ostatních příkazů včetně krátké nápovědy.

Tabulka 19: Použité příkazy shellu v testovací aplikaci

Vývojový diagram této úlohy lze nalézt v příloze č.4b.

12.5 Úloha Sdcard_write_task

Tato úloha je spouštěna při nahrávání zvuku a jejím úkolem je vyzvedávat zprávy obsahující nahraná data z fronty a ukládat je na paměťovou kartu. Tato funkčnost je implementována odděleně od úlohy shellu ve které běží samotné přehrávání a nahrávání proto, že zápis na kartu je velmi pomalý a docházelo by k přetečení zásobníků ovladače sběrnice I^2S s tím k narušení souvislosti záznamu.

Po spuštění úlohy dojde k otevření fronty zpráv a úloha vstoupí do smyčky, ve které kontroluje zda jsou ve frontě zprávy a jestli nedošlo k dokončení záznamu. V případě přítomnosti zpráv se tyto pokusí zapsat na paměťovou kartu. Po dokončení zápisu je aktivována událost oznamující dokončení zápisu na kartu a úloha je odstraněna. Vzhledem k tomu, že tato úloha má nejnižší prioritu, je záznam prováděn pouze v mezičase, tak aby nenarušil chod aplikace. Vývojový diagram lze nalézt v příloze č.4d.

12.6 Funkce Shell_play a příkaz play

Pomocí příkazu play zadaného s patřičnými parametry do shellu na sériovém portu je možno přehrát libovolný zvukový soubor na SD kartě připojené k Freescale Tower. Nápovědu k použití lze vypsat pomocí příkazu `help play`, jehož výstup vypadá následovně:

```
Usage: play <device> <filename>
  device        = playback device (i.e. "i2s:")
  filename      = wav file to play
```

Příkaz má tedy dva parametry z nichž první je názvem zařízení na němž se bude soubor přehrávat a druhý je cesta k samotnému souboru. Příklad výpisu v terminálu při přehrání konkrétního souboru pak může vypadat takto:

```
shell>play i2s: A:\test.wav
  Playing...done
  Interrupts: 22495
  Underruns left: 0
  Underruns right: 0
shell>
```

Na příkladu si můžeme všimnout, že v čase přehrávání je vytištěno hlášení „`Playing...`“, o jeho bezchybném dokončení je uživatel informován hlášením „`done`“. V případě chyby v kterékoliv části programu je příslušná hlaška vypsána v průběhu přehrávání (viz vývojový diagram v příloze č.4e). Po dokončení přehrávání anebo výpisu chyb je zobrazena krátká informační statistika (počet přerušení, podtečení levého a pravého kanálu).

Vnitřně funkce sestává z několika kroků: načtení souboru a kontrola jeho hlavičky, nastavení zařízení ovladače I^2S podle zjištěných parametrů, nastavení parametrů hardwarové platformy a čtení souboru a kopírování dat do ovladače po blocích 512 bajtů. Tato hodnota je zvolena shodně s velikostí bloku SD karty, čímž je dosaženo optimální rychlosti čtení a nedochází ke zpomalování případných dalších procesů.

12.7 Funkce Shell_record a příkaz record

Příkaz `record` je použit pro nahrávání zvuku ze zařízení na SD kartu. Nápovědu lze obdobně jako v případě funkce pro přehrávání zobrazit pomocí `help record`. Výstup nápovědy vypadá následovně:

```
Usage: record <device> <filename> <format>
  device          = recording device (i.e. “i2s:”)
  filename        = filename for recording
```

```
  format          = length_in_seconds:sampling_frequency:bit_depth:
channels
```

Tato funkce tedy vyžaduje oproti nahrávání o jeden parametr navíc, kterým je formát nahrávaného souboru. Formát se zkládá ze čtyř polí oddělených dvojtečkou, kterými jsou: délka nahrávky v sekundách, vzorkovací frekvence, bitová hloubka nahrávky a počet kanálů. Pro konkrétní parametry pak výpis v shellu může vypadat například takto:

```
shell> record i2s: A:\test_record.wav 10:44100:16:2
  Recording...done
  Received bytes: 176400
  Interrupts: 11085
  Overruns left: 0
  Overruns right: 0
```

V průběhu nahrávání a ukládání nahraných dat na paměťovou kartu je tedy zobrazeno hlášení „`Recording...`“ a po jeho dokončení přibude hlášení „`done`“. V případě narušení chodu aplikace jsou chybová hlášení vytištěna do shellu (viz vývojový diagram v příloze č.4f). Po dokončení nahrávání je uživateli zobrazen krátký přehled informací o proběhlé operaci. V tomto přehledu je informace o: počtu nahraných a uložených bajtů na paměťovou kartu (bez hlavičky), počtu přerušení na periferii a přetečení pravého a levého zásobníku.

Samotná funkce pak funguje tak, že:

1. Vytvoří soubor na paměťové kartě a podle zadaných parametrů vyplní hlavičku.
2. Nastaví parametry ovladače.
3. Nastaví parametry hardwarové platformy.
4. Vypočítá počet požadovaných datových bloků.
5. Vytvoří úlohu `Sdwrite_task`.
6. Vytvoří frontu zpráv pro předávání získaných dat úloze `Sdwrite_task`.
7. Ve smyčce získává data ze zařízení a předává je k ukládání do fronty zpráv.
8. Počká na dokončení ukládání dat a vytiskne statistiku.

13 ZÁVĚR

V průběhu práce se mi podařilo úspěšně navrhnout a naprogramovat ovladač sběrnice I^2S. Uživatelské rozhraní je ovladačem poskytováno formou jednoduchého API. Jednotlivé operační vrstvy ovladače vyhovují vrstvovému modelu v systému MQX a v současné době je ovladač zařazen v jeho alfa verzi v poslední aktualizaci operačního systému MQX 3.8.

Vytvořil jsem také schémata zapojení a desky plošných spojů k ověření funkčnosti ovladače. Tyto desky jsou navrženy ve velikosti vhodné pro umístění do systému Freescale Tower. Jejich koncepce nabízí uživateli použít kombinaci základní desky a výkonového modulu, případně využít vlastní řešení koncové části řetězce zpracování zvuku. Ke své funkci desky využívají v digitální části kvalitní D/A převodník a v části analogové výkonný sluchátkový zesilovač. Svými parametry navržené řešení zcela vyhovuje zamýšlenému účelu. Základní funkce jako jsou vysílání a odesílání dat po sběrnici byly úspěšně otestovány také pomocí modulu s audio kodekem SGTL5000 od firmy Freescale.

Pro účely testování a dalšího vývoje jsou k dispozici testovací aplikace jak pro desku firmy Freescale, tak pro mnou vytvořené desky. Tyto aplikace prezentují funkčnost ovladače pomocí přehrávání a nahrávání souborů ve formátu WAV z respektive na paměťové médium ve formě SD karty. Vstup a výstup těchto aplikací je poskytován pomocí shellu na sériové lince.

Dalšími možnými kroky ve vývoji popsaného zařízení je zrevidování celého zapojení a desek plošných spojů a jejich plná integrace do systému Freescale Tower.

SEZNAM POUŽITÉ LITERATURY

[1] RIZVI, Syied; KHAN, Imran. *In The Mix : Tips for Mixed Signals Design . Printed Circuit Design and Manufacture* [online]. 2005, [cit. 2011-04-10]. Dostupný z WWW: www.nexlogic.com/Portals/0/0502rizvi.pdf.

[2] OTT, Henry W. *Partitioning and Layout of mixed signal Mixed-Signals PCB*. In Printed Circuit Design [online]. , June 2001 [cit. 2011-04-10]. Dostupné z WWW: www.hottconsultants.com/pdf_files/june2001pcd_mixedsignal.pdf.

[3] WU, John. *Precision Analog Design Demand Good PCB Layouts*. [online]. 2005 [cit. 2011-04-10]. Dostupný z WWW: http://www.x2y.com/filters/TechDay09kr_hpa_Track2_1_Precision_Analog_Designs_Demand_GoodPCBLayouts%20_JohnWu.pdf.

[4] FREESCALE SEMICONDUCTOR, Inc. *MCF5441x Reference Manual* [online]. Rev. 3. Post Office Box 655303, Dallas, Texas 75265 : Freescale Semiconductor, Inc. Technical Information Center, EL516 2100 East Elliot Road Tempe, Arizona 85284 , 9/2010 [cit. 2011-04-10]. Dostupné z WWW: http://cache.freescale.com/files/32bit/doc/ref_manual/MCF54418RM.pdf.

[5] FREESCALE SEMICONDUCTOR, Inc. *Freescale MQX RTOS Reference Manual* [online]. Rev. 6. Post Office Box 655303, Dallas, Texas 75265 : Freescale Semiconductor, Inc. Technical Information Center, EL516 2100 East Elliot Road Tempe, Arizona 85284, 4/2011 [cit. 2011-04-10]. Dostupné z WWW: http://cache.freescale.com/files/32bit/doc/ref_manual/MQXRM.pdf?fpsp=1&WT_TYPE=Reference%20Manuals&WT_VENDOR=FREESCALE&WT_FILE_FORMAT=pdf&WT_ASSET=Documentation

[6] FREESCALE SEMICONDUCTOR, Inc. *Freescale MQX RTOS Users Guide* [online]. Rev. 3. Post Office Box 655303, Dallas, Texas 75265 : Freescale Semiconductor, Inc. Technical Information Center, EL516 2100 East Elliot Road Tempe, Arizona 85284, 4/2011 [cit. 2011-04-10]. Dostupné z WWW: http://cache.freescale.com/files/32bit/doc/user_guide/MQXUG.pdf?

fpsp=1&WT_TYPE=Users%20Guides&WT_VENDOR=FREESCALE&WT_FILE_FORMAT=pdf&WT_ASSET=Documentation

[7]*Katalogový list obvodu DIR9001* [online]. Post Office Box 655303, Dallas, Texas 75265 : Texas Instruments, 2009 [cit. 2011-04-10]. Dostupné z WWW: http://www.ti.com/lit/gpn/dir9001.

[8]*Katalogový list obvodu CAT9554* [online]. Revize F. P.O. Box 5163, Denver, Colorado 80217 USA : Literature Distribution Center for ON Semiconductor, 2009 [cit. 2011-04-10]. Dostupné z WWW: http://www.onsemi.com/pub/Collateral/CAT9554-D.PDF

[9]*Katalogový list obvodu CD74HCT243* [online]. Post Office Box 655303, Dallas, Texas 75265 : Texas Instruments, 1997, 2003 [cit. 2011-04-10]. Dostupné z WWW: http://focus.ti.com/docs/prod/folders/print/cd74hct243.html#technicaldocuments

[10]*Katalogový list obvodu OPA2134* [online]. Post Office Box 655303, Dallas, Texas 75265 : Texas Instruments, 11-AUG-2009 [cit. 2011-04-10]. Dostupné z WWW: <focus.ti.com/lit/ds/symlink/opa2134.pdf>

[11]FREESCALE SEMICONDUCTOR, Inc. *MCF5441x ColdFire® Microprocessor Data Sheet* [online]. Rev.6. Post Office Box 655303, Dallas, Texas 75265 : Freescale Semiconductor, Inc. Technical Information Center, EL516 2100 East Elliot Road Tempe, Arizona 85284 , 09/2010 [cit. 2011-04-10]. Dostupné z WWW: http://cache.freescale.com/files/32bit/doc/data_sheet/MCF54418.pdf

[12]*Katalogový List obvodu PCM1789* [online]. Post Office Box 655303, Dallas, Texas 75265 : Texas Instruments, OCTOBER 2008, JANUARY 2009 [cit. 2011-04-10]. Dostupné z WWW: http://www.ti.com/lit/gpn/pcm1789

[13]*Katalogový List obvodu SN74LVTH125* [online]. Post Office Box 655303, Dallas, Texas 75265 : Texas Instruments, AUGUST 1997, OCTOBER 2003 [cit. 2011-04-10]. Dostupné z WWW: http://www.ti.com/lit/gpn/sn74lvth125

[14]*Katalogový List obvodu TPA6120A2* [online]. Post Office Box 655303, Dallas, Texas 75265 : Texas Instruments, MARCH 2004 [cit. 2011-04-10]. Dostupné z WWW: http://www.ti.com/lit/gpn/tpa6120a2

[15]FREESCALE SEMICONDUCTOR, Inc.*MCF5441X Tower Module : User Manual* [online]. Rev. 1.1. Post Office Box 655303, Dallas, Texas 75265 : Freescale Semiconductor, Inc. Technical Information Center, EL516 2100 East Elliot Road Tempe, Arizona 85284, 2010 [cit. 2011-04-10]. Dostupné z WWW: http://cache.freescale.com/files/netcomm/doc/user_guide/TWRMCF5441XUM.pdf?fpsp=1

[16]FREESCALE SEMICONDUCTOR, Inc.*Freescale MQX I/O Drivers : Users Guide* [online]. Rev. 9. Technical Information Center, CH370 1300 N. Alma School Road Chandler, Arizona 85224 : Freescale Semiconductor , 04/2011 [cit. 2011-04-10]. Dostupné z WWW: http://cache.freescale.com/files/32bit/doc/user_guide/MQXIOUG.pdf.

[17]*I²S bus specification.* Philips Semiconductor, 1986. 7 s. Dostupné z WWW: http://www.classic.nxp.com/acrobat_download2/various/I2SBUS.pdf.

[18]BURR-BROWN CORPORATION. *Application Bulletin*: *DC-TO-DC Converter Noise Reduction* [online]. PO Box 11400 Tucson, AZ 85734 Street Address: 6730 S. Tucson Blvd.: Burr-Brown Corporation, 1987, 4 s.[cit. 2011-12-23]. Dostupné z: http://www.ti.com/general/docs/lit/getliterature.tsp?literatureNumber=sbva012

[19]XP POWER LIMITED. *3 Watts IR Series*. 401 Commonwealth Drive, Haw Par Technocentre,#02-02, Singapore 149598 : XP Power, 2011, 1 s.[cit. 2011-12-23]. Dostupné z: http://xppower.com/pdfs/SF_IR.pdf

[20]MAXIM INTEGRATED PRODUCTS. *MAX865*: *Compact, Dual-Output Charge Pump*. Rev. 1, 7/97. 120 San Gabriel Drive, Sunnyvale, CA 94086 408-737-7600: Maxim Integrated Products, 1997, 8 s. [cit. 2011-12-23]. Dostupné z: http://datasheets.maxim-ic.com/en/ds/MAX865.pdf

[21]KARKI, Jim. TEXAS INSTRUMENTS INCORPORATED. *Application Report*: *Active Low-Pass Filter Design*. SLOA049A. 12500 TI Boulevard, Dallas, Texas 75243, USA: Texas Instruments Incorporated, October 2000, 24 s. [cit. 2011-12-23]. Dostupné z: http://www.ti.com/lit/an/sloa064/sloa064.pdf

[22]TEXAS INSTRUMENTS INCORPORATED. *3.3V DUAL PLL MULTICLOCK GENERATOR* [online]. Rev. A. 12500 TI Boulevard, Dallas, Texas 75243, USA: Texas

Instruments Incorporated, August 2002, 20 s.,September 2002 [cit. 23.12.2011]. SLES046A. Dostupné z: http://www.ti.com/lit/ds/symlink/pll1706.pdf

[23]JAN, Scott Wilson. Microsoft WAVE soundfile format: WAVE PCM soundfile format. STANFORD UNIVERSITY. *Center for Computer Research in Music and Acoustics* [online]. 20. ledna 2003 [cit. 2012-04-01]. Dostupné z: https://ccrma.stanford.edu/courses/422/projects/WaveFormat/

[24]FREESCALE SEMICONDUCTOR, Inc. *Freescale MQX™MFS™ User's Guide* [online]. Rev.2.2. Freescale SemiconductorTechnical Information Center, CH3701300 N. Alma School RoadChandler, Arizona 85224, 2010, 08/2010 [cit. 2012-04-04]. Dostupné z: http://www.freescale.com/files/32bit/doc/user_guide/MQXMFSUG.pdf

[25]HEIJLIGERS, Marc. Introduction to digital audio. DAC GROUP. *Marc's Homepage* [online]. 2001 [cit. 2012-05-03]. Dostupné z: http://members.chello.nl/~m.heijligers/DAChtml/Digital%20Theory/Digital%20theory.html

[26]RENESAS ELECTRONICS CORPORATION. *General RTOS Concepts* [online]. Renesas Electronics America Inc. 2880 Scott Boulevard, Santa Clara 95050 Phone: 408-588-6000 Fax: 408-588-6130, 2010, 18 s. [cit. 3.5.2012]. RES05B0008-0100. Dostupné z: http://documentation.renesas.com/doc/products/tool/apn/res05b0008_r8cap.pdf