ZÁPADOČESKÁ UNIVERZITA V PLZNI

FAKULTA ELEKTROTECHNICKÁ

KATEDRA APLIKOVANÉ ELEKTROTECHNIKY A TELEKOMUNIKACÍ

Diplomová práce

Návrh inteligentního systému budovy

Autor práce: **Brych Milan**

Vedoucí práce: **Ing. Matouš Bartl**

Plzeň 2013

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2012/2013**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Milan BRYCH**
Osobní číslo:	**E11N0172P**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Elektronika a aplikovaná informatika**
Název tématu:	**Návrh inteligentního systému budovy**
Zadávající katedra:	**Katedra aplikované elektroniky a telekomunikací**

Zásady pro vypracování:

Práce je zadána na základě požadavku společnosti ST Microelectronics. Výsledkem má být ukázka systému inteligentní budovy použitelná na stánku při prezentacích v rámci odborných výstav. Hardware jako jsou inteligentní senzory, sběrný embedded systém a akční členy bude poskytnut zadávající společností. Hlavní těžiště práce spočívá ve tvorbě firmware a software. Výstupem jsou rovněž aplikační poznámky pro použití vyvinutých prvků.

1. Prostudujte dostupnou teorii systémů inteligentních budov.
2. Seznamte se s portfoliem společnosti ST Microelectronics a vyberte vhodné konstrukční prvky pro které navrhnete řešení ukázového systému.
3. Zpracujte požadovaný software.
4. Popište dosažené výsledky a sepište požadované aplikační poznámky.

Rozsah grafických prací:	**podle doporučení vedoucího**
Rozsah pracovní zprávy:	**30 - 40 stran**
Forma zpracování diplomové práce:	**tištěná/elektronická**

Seznam odborné literatury:

Student si vhodnou literaturu vyhledá v dostupných pramenech podle doporučení vedoucího práce.

Vedoucí diplomové práce:	**Ing. Matouš Bartl** Katedra aplikované elektroniky a telekomunikací
Konzultant diplomové práce:	**Ing. Matouš Bartl** Katedra aplikované elektroniky a telekomunikací

Datum zadání diplomové práce:	**15. října 2012**
Termín odevzdání diplomové práce:	**9. května 2013**

L.S.

Doc. Ing. Jiří Hammerbauer, Ph.D.
děkan

Doc. Dr. Ing. Vjačeslav Georgiev
vedoucí katedry

V Plzni dne 15. října 2012

Anotace:

Diplomová práce s názvem *Návrh inteligentního systému budovy* popisuje hardwarové a softwarové nástroje použité při vývoji programového vybavení pro procesory od společnosti STM. Obsahuje teoretický rozbor komunikačních sběrnic systému a nakonec i funkci jednotlivých funkcí bloků. Krom hlavního zadání je v práci popsáno několik samostatných úkolů, které v průběhu práce vyplynuly, a jejich zkoumání bylo dostatečně spjaté se samotnou prací.

Klíčová slova:

ST Microelectronics , mikroprocesor, sběrnice CAN, komunikační protokoly, automatizace budovy, DMA – přímý přístup do paměti, DCMI – rozhranní pro digitální kameru, ARM, CMSIS, SPI, I2C

Annotation:

Master thesis with title Intelligent building system design describes hardware and software tools used for developing software for microcontrollers from company ST Microelectronics. It contains theoretical analysis of communication buses of system and it the last chapter also functions of individual blocks. Beside main target is several separated tasks in thesis which has arisen in the course of work and their investigation was sufficiently connected with thesis.

Key words:

ST Microelectronics, microcontroller, CAN bus, Ethernet, communication protocols, building automation, Eclipse, DMA – direct memory access, DCMI – digital camera interface, ARM, SPI, I2C, CMSIS

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce. Dále prohlašuji, že veškerý software, použitý při řešení této diplomové práce, je legální.

V Plzni dne 11.2.2013 .....

Brych Milan

Poděkování

Tímto bych rád poděkoval vedoucímu diplomové práce Ing. Matouši Bartlovi za cenné profesionální rady, připomínky a metodické vedení práce. Zároveň mé poděkování patří o společnosti ST Microelectronics za možnost uskutečnit vývoj na zapůjčených vývojových prostředcích a softwaru. Rovněž děkuji za směřování práce vedoucí na seznámení se s moderními technologiemi vývoje.

Obsah

Úvod

Tato diplomová práce byla zadána firmou ST Microelectronics za účelem prozkoumání možností tvoření senzorických a informačních sítí na drátových a bezdrátových platformách z portfolia společnosti.

Výsledky práce jsou použitelné v praktické realizaci a pro možné prezentace aplikovaných platforem společnosti STM.

Struktura sítí, datové jednotky a použitá přenosová media byla volena vždy po předchozí konzultaci a s ohledem na maximální pružnost a funkcionalitu zvoleného řešení.

1. Představení myšlenky a topologie sítě

1.1. Cíl řešení

V rámci této práce mělo vzniknout zejména programové vybavení pro hardware od společnosti ST Microelectronics, které by realizovalo síť použitelnou pro ovládání a automatizaci budov bytového, kancelářského nebo průmyslového charakteru. Samotná myšlenka nebyla svázána topologickým návrhem nebo jakoukoli jinou podmínkou.

Základní myšlenkou „inteligentní“ senzorické sítě byla variabilita, budoucí modularita, jednoduchost komunikačních protokolů a snadná integrace do případných vyšších vrstev řešení. V souladu s tímto požadavkem se ubíralo i možné předávání dat do PC platformy případně do lokálních nebo síťových databází.

Pro splnění tohoto cíle bylo zvažováno několik možností řešení datového propojení jednotlivých buněk sítě a různé obvodové verze jednotlivých buněk a uzlů. Rozložení senzorických a akčních členů nebylo nijak omezeno. Byla požadována kompaktnost jednotlivých prvků z důvodu lepší integrace. Součástí práce nebyl návrh hardware částí, ale řešení mělo s proveditelností kompaktních prvků počítat. Další výchozí předpoklad byl pokud možno, co nejsnazší implementace a případně nejnižší servisní režie, což jsou vlastnosti, které by ze systému dělali snadno využitelný a obsluhově nenáročný.

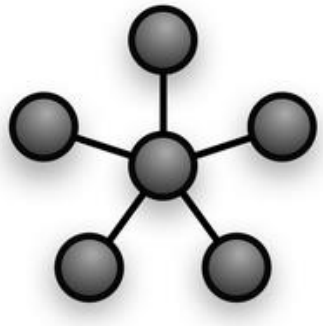
1.2. Možnosti topologie

Kvůli několika hlediskům cílové realizace proběhla úvaha nad topologií použité sítě. Měla být brána v potaz spolehlivost, náročnost realizace, ekonomická náročnost a variabilita v průběhu používání. V závislosti na různých typech prostor se liší i konkrétní požadavky na topologii (obytná a průmyslová nebo kancelářská budova má výrazně rozdílné rozměry a také požadavky na datové rozvody případně možnosti uložení prvků sítě).

Hvězda

Výhody:

- Vyhrazená linka pro každý připojený bod
- V případě poruchy ovlivněn jen jeden konkrétní bod
- Velice rychlá a snadná lokalizace závady na připojeném bodě případně na vedení k němu
- Výborné možnosti rozšiřitelnosti



Obrázek 1 – Hvězda

Zdroj: www.blog.teachbook.com.au

Nevýhody:

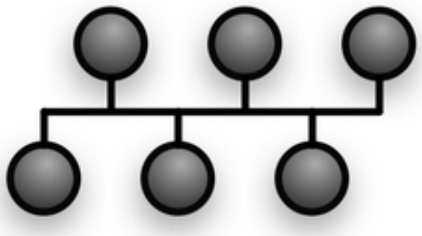
- V tomto případě až neúnosná spotřeba vodičů – mnohdy možná nerealizovatelné a opravdu dlouhé spojení od aktivního prvku k jednotlivým buňkám

V případě zpracovávaného projektu vyžaduje tato topologie neúnosné množství kabeláže a délku spojů. Její nasazení by vedlo k celkové nepřehlednosti řešení. Na druhou stranu je v síti nejmenší potřebný počet aktivních prvků, což minimalizuje nejen cenu, ale i možnou poruchovost. Ovládání a nahrávání nového firmware by rovněž bylo velkou výhodou. Tato topologie je pro systém nevyhovující.

Sběrnice

Výhody:

- Nejnižší nároky na kabeláž
- Velice snadné zřízení a rozšiřování od koncových bodů



Obrázek 2 – Sběrnice

Zdroj: www.blog.teachbook.com.au

Nevýhody:

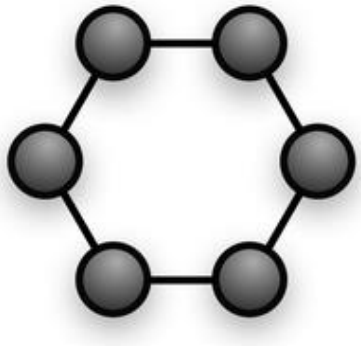
- V případě přerušení sběrnice přestává fungovat velká část sítě

- Při větším počtu stanic velice časté kolize v komunikaci a z toho vyplývající silné snížení přenosových kapacit

Pro případ tohoto systému je tato sběrnice značně nevyhovující – vyžadovala by vedení sběrnice, tak aby procházela všemi možnými místy, kam lze při realizaci či v budoucnu umisťovat stanice. Tuto topologii nelze z principu věci použít.

Kruhová sběrnice

Kruhová sběrnice je strukturou velice podobná předchozímu řešení. Je o něco méně náchylná na jednonásobné přerušení sběrnice, které je mnohdy špatně identifikovatelné. Ale rovněž ji nelze použít z principu modularity sítě (nelze dopředu jasně stanovit rozmístění stanic).

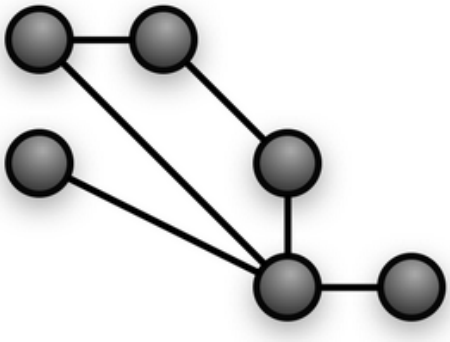


Obrázek 3 – Kruh

Zdroj: www.blog.teachbook.com.au

Topologie typu síť (Mesh)

Velice pružná a adaptivní topologie, která skýtá v podstatě jen výhody. Není náchylná k jednotlivým poruchám až do odříznutí páteřních cest nebo jednotlivých buněk. Nabízí i rozumný kompromis v použité kabeláži. Jedinou nevýhodou zůstává velice složitá režie komunikace v případech sítí jako v popisovaném systému.



Obrázek 4 –Mesh

Zdroj: www.blog.teachbook.com.au

Stromová topologie s koncentrátory dat

Výhody:

- Výrazná úspora potřebné kabeláže

Nevýhody:

- Porucha na prvkách bližších ke kmenu způsobí výpadek větších částí – větví - systému

U této topologie výrazně klesá množství potřebné kabeláže. Je finančně ještě stále málo náročná. Zároveň vzájemná komunikace mezi jednotlivými koncentrátory dat by vyžadovala složitější řízení sítě a netriviální zpracování dat. V zásadě se jedná o jednotlivé hvězdy pospojované k jedné kmenové sběrnici pomocí hub zařízení. I když toto rozložení je již velice blízké potřebnému, bylo potřeba vymyslet kombinaci s dalšími, aby výsledek vyhovoval užití v praxi.

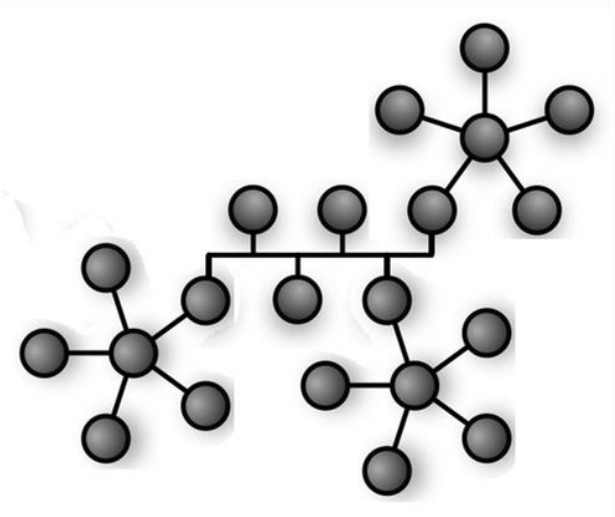
Samostatné moduly připojené ke sběrnici - hybridní topologie

Tato varianta je určitá kombinace předchozích řešení. Předpokládá se využití drobných řídících a zároveň senzorických modulů. Podobného principu využívají sítě typu LON. Vzhledem ke své vysoké variabilitě a pružnosti je tato struktura ideální variantou pro budování sítí tohoto charakteru. Nevýhodou zůstává její cenová náročnost oproti předchozím variantám, jelikož každý prvek je v této síti aktivní a vyžaduje jak procesor pro místní zpracování dat, tak i komunikační rozhranní (případně další rozhranní jako JTAG pro ladění aplikací a snadný update firmaware, rozhraní pro bootloader – např. UART nebo CAN – výhoda a možnost nahrávat firmware v rámci celé sítě na konkrétní jednotky bez nutnosti separace jednotek nebo jejich přímé demontáže…).

Značnou výhodou je obrovská modularita, kde celou síť lze libovolně funkčně přestavovat, sdružovat jednotlivé prvky do skupin. Síť tak může vykonávat jakkoli složité úkoly, které závisí pouze na externím řídícím programu. Pomocí adres v této síti dochází ke komunikaci s jednotlivými body, které nezávisle na svém umístění odevzdávají data a

vykonávají příkazy. Veškeré řídící struktury jsou tedy obsažené v nadřazeném hardware a lze je měnit na jediném místě.

Pokud se síť v takovéto topologii optimálně navrhne, lze v budoucnu jednoduše měnit její funkcionalitu a přizpůsobovat pozdějším potřebám. Případné přidávání dalších jednotek nebo celých funkčních bloků je poměrně jednoduchá záležitost nevyžadující složitější nastavování systému nebo změnu jeho struktury. Na rozdíl od pevných topologií tak umožňuje přizpůsobit systém bez přemisťování funkčních prvků a ušetřit náklady a nároky na její využití. Z pohledu komerčního využití je velkým přínosem, že nasazení nového funkčního bloku ke stávajícím je jednoduché a umožní například i vzájemnou synergii jednotlivých bloků. (např. stávající systém obsahuje řízení teplot a spotřeby energie spolu s řízením osvětlení a zásuvkových obvodů. Přidání nového bloku zabezpečovacích senzorů a bezpečnostních prvků – ten může při vyvolání poplachu ovládat osvětlení a podobně).

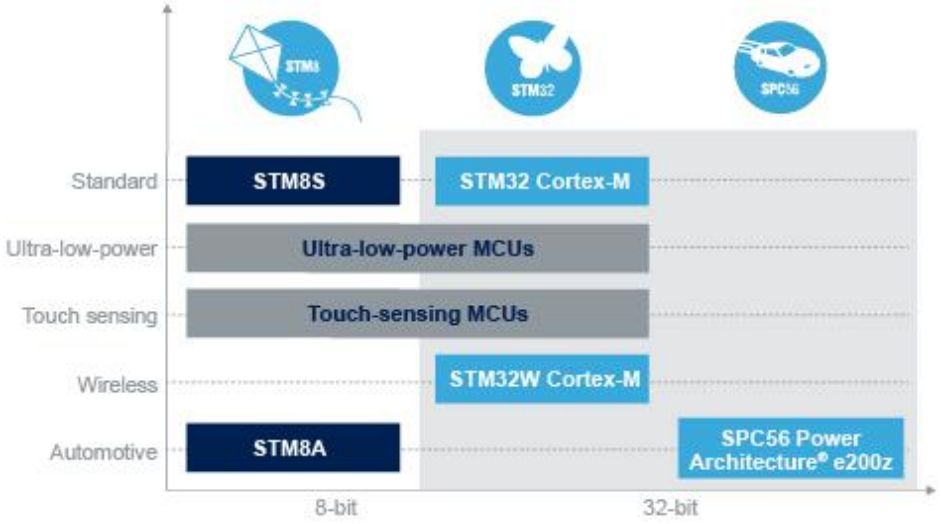


Obrázek 5 – Hybridní topologie

Zdroj: vlastní tvorba 2013

2. Použité HW prostředky

2.1. Procesory ST

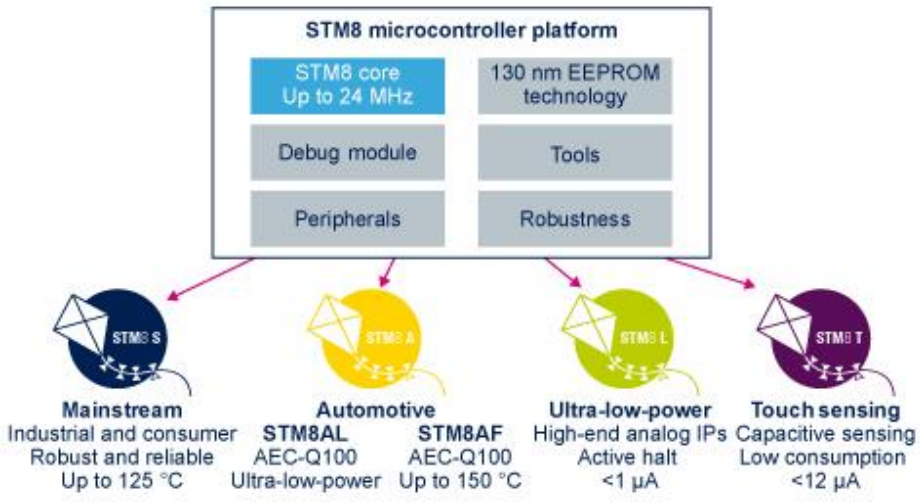


Obrázek 6 – Portfolio STM Zdroj: www.st.com

STM8

- Procesory postavené na proprietární architektuře společnosti STM
- Napájecí napětí 2,95 – 5,5 V
- Až 24 MHz a 20 MIPS

Pro nejjednodušší jednotky obsahující jen základní senzory a akční členy. STM8S je vybavena dokonce komunikačními periferiemi UART, SPI, I2C a CAN. Tento procesor lze s výhodou použít pro ovládání zásuvek, vypínačů a podobných základních prvků sítě. Z 8-bitové rodiny je to jediný využitý model, protože další zajímavá varianta 8L (low power series) nedisponuje CAN periferií.



Obrázek 7 – rozdělení řady STM 8 Zdroj: www.st.com

STMF0xx

- Archytektura ARM 32-bit Cortex M0
- Napájecí napětí 2 – 3,6 V
- Až 48 MHz
- Některé 5V tolerant pins

Základní 32-bitová řada procesorů STM. Tyto procesory umožňují připojit kapacitní senzory k periferii obsluhující touchsensing – výborné využití pro dotykové ovládání osvětlení – umožňuje i rotační pohyb. Lze ovládat hlasitost hudby. Svou výbavou je předurčen pro jednodušší ovládací prvky, případně řízení akčních členů pro konkrétní spotřebiče, případně obsluhu dílčích funkcí místností. Rovněž již obsahuje DMA řadič s 5 kanály. Má dostatek čítačů a ADC kanálů a standardní výbavu komunikačních periferií, není ovšem vybaven CAN řadičem a nelze tedy použít jako samostatný bod.

STM32F1xx

- Jádro ARM 32-bit Cortex M3
- Napájecí napětí 2,0 – 3,6 V
- Až 72MHz
- Všechny piny jsou 5V tolerant

Výkonnější a spotřebou úspornější řada procesorů než F0. Díky opravdu bohaté vybavenosti periferiemi lze tuto řadu použít na ovládací prvky jednotlivých místností a pomocí CAN řadiče připojovat do sítě. V kombinaci s procesory F0 se dá realizovat velice vybavená ovládací jednotka disponující dotykovým ovládáním, možností snímat mnoho digitálních i analogových hodnot a naopak řídit všechny připojené akční členy.

Od řady F1 se odvíjí i low power rodina STM32F1L, která v projektu použitá není, ale určitě by v budoucnu stála za implementaci do míst, kde je třeba úspory energie – přineslo by to mnoho výhod například při výpadku energie – systém by tak mohl být na záložní zdroj déle funkční.

Další zatím nevyužitou alternativou je STM32W, která nabízí integrované 2,4 GHz transievery. Nevýhodou je, že bezdrátové periferie jsou vyráběny jiným výrobcem a tak programová implementace není triviální.

V brzké době by měla přijít na trh nová rodina s bezdrátovou periferií a dokonce podpořená novým proprietárním stackem ZigbeeIP, který slibuje velký výkon, energetickou úsporu a snadnou programovatelnost na základě CMSIS struktur. Následná implementace této rodiny do systému by mohla přinést velké výhody a ve spojení s energy harvestingem velice zajímavé energeticky nezávislé moduly, které by nebylo potřeba připojovat klasickým způsobem.

STM32F2

- Jádro ARM 32-bit Cortex M3
- Napětí 1,8 – 3,6 V
- Až 120MHz
- USB 2.0
- Ethernet MAC 10/100
- SDIO for SD cards
- DCMI rozhranní pro stream dat z kamery

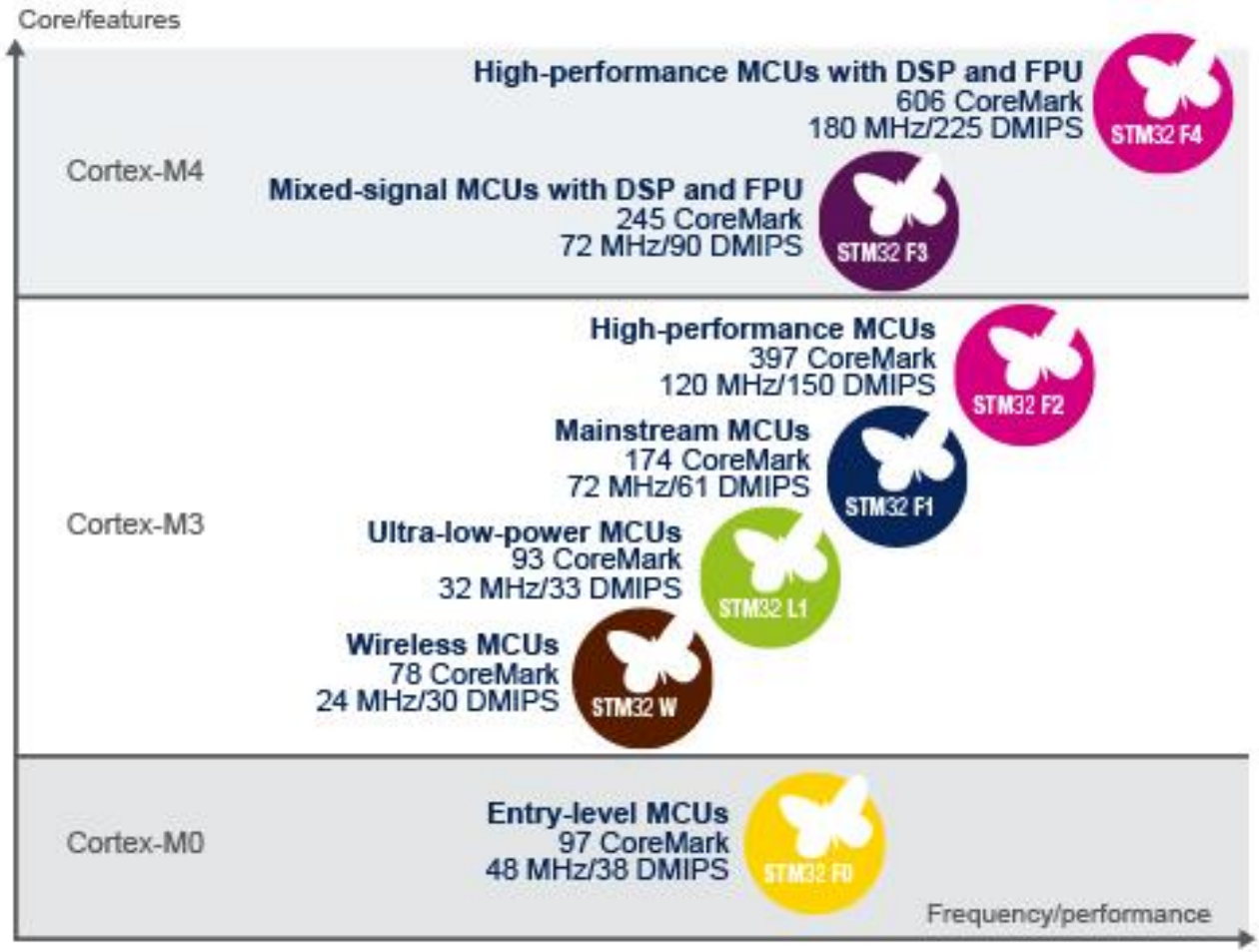
Velice výkonný procesor, umožňující vykonávat veškeré obslužné úkony pro koncové stanice připojené na sběrnici. K procesoru se dají připojovat senzory, komunikující po SPI, I2C, one-wire sběrnici a mnoho dalších. Lze zpracovávat velké množství digitálních i analogových dat. Tyto procesory lze požít navíc jako komunikační body pro propojení přes Ethernet – umožňují výměnu větších objemů dat (např. multimediální obsah jako je video zachycené přes DCMI rozhranní). Pomocí LwIP stacku lze procesory poměrně pohodlně začlenit do větší sítě řízené klasickými konvenčními routery (mohou získávat – případně přidělovat – adresy z DHCP serveru atd.).

K těmto procesorům lze pohodlně připojit i LCD displeje s dotykovým sklem a použít je tedy jako dotykové vstupní rozhranní, kterými lze měnit některé aktuální nastavení systému (například teploty v místnosti atd.). V době nečinnosti lze zobrazovat fotografie. Dále lze přenášet na povel např. data ze vstupní kamery.

Tato řada procesorů je dostatečně výkonná pro veškeré výše uvedené úkony. Jediné omezení nastává při zpracování většího obrazu. Veškeré výše uvedené funkce jsou v jednotce s tímto procesorem implementovány a vzhledem k míře použití periferií a přenosů přes DMA řadiče, je vytížení jádra značně minimalizováno.

Připojení do sítě je realizováno, stejně jako v předchozích případech, pomocí CAN sběrnice.

V případě nutnosti navýšení výkonu a rozšíření možností lze použít ještě výkonnější řadu STM32F4 založenou na jádrech ARM Cortex M4 – v případě veškeré dosavadní implementace však nebylo nutné tento potenciál využít. Řada STMF4 nabízí také jednotku FPU a podporu pro číslicové zpracování signálů.



Obrázek 8 – rozdělení řady STM 32 Zdroj: www.st.com

2.2. Evaluační vývojové desky (Evaluation boards)

Veškerý vývoj aplikací probíhal na evaluačních deskách od STM.

Evaluační desky jsou koncipovány, tak aby nabídly, co nejširší využitelnost periferií, které jsou v procesoru obsažené. Mají z hardwarového hlediska přizpůsobený návrh, tak aby z obvodového hlediska nebylo nutné upravovat jakékoli hodnoty součástek. Z tohoto důvodu je možné aplikace vyvíjet rychle a bez ohledu na HW stránku. Mnohdy tato výhoda přináší i úskalí, kdy je třeba pracovat s vývody, které jsou napevno použité k některým okolním čipům nebo zařízením.

Pro každou řadu procesorů je vyráběna verze desky, která přesně odpovídá použitým periferiím. Na deskách jsou dále přítomné čipy ST-LINK, které tak umožňují připojení desky k počítači jen přes USB. Je na výběr více zdrojů napájení (např. USB, externí zdroj…). Komunikační rozhranní jako CAN, UART, USB, I2C, SPI, SDIO a až 1Gbyte Ethernet jsou k dispozici dle možností procesoru. U STM32 F1xx Eval je k dispozici ještě obyčejný LCD. U STM32 F2 je LCD s vyšším rozlišením a dotykovým rozhranním.

Datové listy všech použitých desek pro vývoj aplikace jsou v přílohách, kde lze dohledat další konkrétní specifika jednotlivých verzí.

3. SW vývojové prostředky:

3.1. Nástroje na programování procesorů (IDE)

STM8xx

Vývoj aplikací pro 8-bitové procesory probíhá v jiných IDE než pro 32-bitové, jelikož 8-bitové procesory jsou postaveny na proprietární architektuře společnosti ST.

Pro tuto platformu byl využíván IDE od ST pod názvem STVD (ST visual developtment) s kompilátorem Cosmic – toto prostředí není úplně přátelské a vyžaduje trpělivější přístup. Je neustále vyvíjené, tak snad budou některé z nedostatků v budoucnu odstraněny (problém je zejména délka přestupu do debug módu a chování nástrojových lišt).

Dalším software pro vývoj je 8-bit verze IAR. Zde je práce o poznání jednodušší a přístup je téměř srovnatelný s klasickými IDE pro 32-bit procesory. Funkcionalita nijak neomezuje práci.

Obě tyto prostředí jsou volně dostupné (ale s omezením velikosti kompilovaného kódu).

STM32

Vývoj pro 32-bit aplikace probíhal v 32-bit verzi software IAR IDE a Keil uVision (pod záštitou ARM). Opět jsou obě IDE stažitelné ve volných verzích pro určitou velikost kódu.

Obě vývojová prostředí a jejich použitelnost je velice podobná a přechod mezi nimi je velice pohodlný. Za dobu používání jsem nenarazil na jedinou nefunkčnost. Kail uVision dokonce v nové verzi implementoval automatické doplňování textu a našeptávání symbolických jmen – tím se práce velice usnadnila.

Většina kódů je psána v projektech, které jsou hybridně použitelné v obou vývojových prostředích. Je u nich použita rozdílná konfigurace projektů, ale využívají stejné knihovny, stacky a zdrojové kódy.

Přes veškerou podobnost mezi Kail uVision a IAR jsou stále mezi prostředími značné (i když na první pohled skryté) rozdíly, které mají nemalou váhu na výslednou aplikaci a její

běh. V několika případech se ukázalo, že lepší optimalizace kódu provádí Keil uVision. Tím, že se jedná o IDE vyvíjené pod společností ARM, jsou optimalizace mnoha funkcí a algoritmů lépe přizpůsobené jádrům ARM, používaných v procesorech od STM.

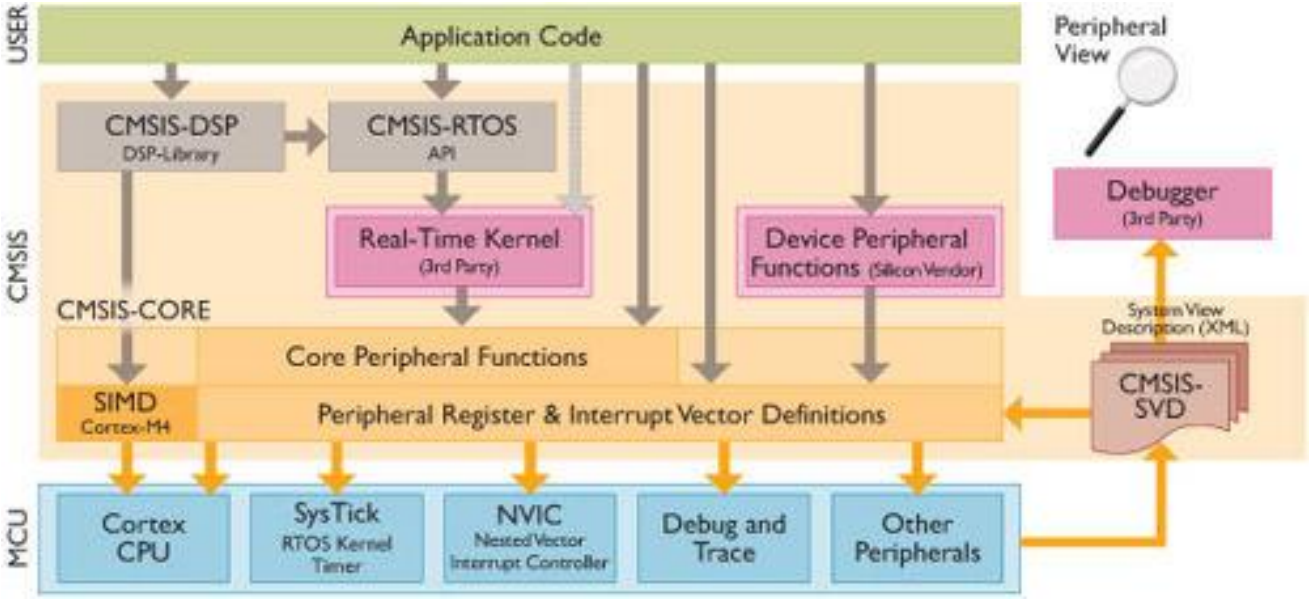
V rámci práce proběhlo i několik benchmark testů na výkonnost obou compilerů. Konkrétním příkladem může být benchmark na výkonnost výsledných kódů po kompilaci pro provádění funkcí, které nejsou hardwarově implementované. Například jádra ARM32 Cortex M0 nemají implementovanou hardwarovou děličku a dělení čísel se tedy řeší softwarovou smyčkou, která dělení vykoná. Předmětem testu bylo měření času, za který pak proběhne určitý počet výpočtů s dělením. Pro obě prostředí byl vybrán stejný vzorek náhodných čísel a byl měřen čas průchodu přes 100 výpočtů.

Výsledky ukázali, že Keil uVision dokáže optimalizovat takovýto kód až do takové míry, kdy dojde k 4x rychlejšímu výpočtu než u prostředí IAR.

Metodika měření, výsledky a kód použitý pro tento benchmark je v příloze.

3.2. CMSIS

Pro práci s veškerými periferiemi u všech rodin procesorů jsou k dispozici knihovny. Struktura knihoven je založena na standardu CMSIS (Cortex Microcontroller Interface Standard), což je abstraktní vrstva nad hardware pro procesory s jádry Cortex-M. Tento standard značně zvyšuje přehlednost kódu, usnadňuje konfiguraci procesorů a jejich periferií a umožňuje přenositelnost kódu s minimalizací nároků na úpravy kódu při tomto přenosu.



Obrázek 9 – CMSIS bloková stavba Zdroj: www.arm.com

CMSIS je využívaný nejen pro ARM jádra, ale i pro periferie dodané výrobcem. Postihuje tedy celou hardwarovou vybavenost procesorů. Z toho plyne, že CMSIS se u různých výrobců vždy o něco liší. Standard obsahuje rovněž různé softwarové vrstvy pro snadnější implementaci některých složitějších struktur:

- CMSIS-CORE – podpora jader Cortex-Mx
- CMSIS-DSP – knihovna obsahující funkce pro DPS jednotky v procesorech – zahrnuje instrukční sadu pro digitální zpracování signálu. Využívá FPU jednotku. Je určena zejména pro jádro Cortex-M4
- CMSIS-RTOS API – podpora real time operačních systémů
- CMSIS-SVD – převodní rozhranní pro debug HW výrobců

3.3. Prostředí Eclipse a vývoj pro STM32:

Samostatnou částí vývoje bylo uvedení do provozu free vývojového prostředí bez omezení velikosti kódu a s možností debug tak, aby byla platforma použitím podobná komerčním nástrojům. Varianta neměla využívat jakékoli komerční součásti, které by časem mohli podléhat licenčním omezením za strany vyvíjejících stran.

V dílčích podobách se již podobná řešení objevovala na internetu, nicméně se jednalo vždy o částečné řešení, které zahrnovalo například jen kompilaci a linkování kódu. Nebo naopak pouze nahrávání kódu do procesoru. Nejdále ze všech takovýchto řešení byl projekt ODeV. Tento projekt v sobě ale zahrnoval jen výše zmíněnou kompilaci a linkování. K nahrávání kódu a debug používala nástroje z Atolic Studia, které jsou sice dostupné zdarma, ale pod licencí Atilic Studia.

Část s komunikací mezi prostředím a procesorem byla potřeba zaměnit za licenčně volný prostředek. Stejně tak následný debug aplikace. Z tohoto důvodu vznikl nový balík již existujících aplikací, sestavený za účelem vytvoření kompletního vývojového prostředí pro procesory STM s jádry ARM.

Jedná se o komplet těchto součástí:

Java runtime envirovment

Prostředí potřebné pro běh veškerých aplikací postavených na platformě Java - stejně tak i vývojového programu Eclipse. Je poskytováno od společnosti Oracle zdarma jako nutné prostředí pro běh Java aplikací.

Sourcery codebench Lite

Balík nástrojů od společnosti Mentor, jehož kompilátor gcc je jako součást zdarma. Stejně tak gdb server (původně samostatný projekt vyvíjený volně – pod Mentorem si stále zachovává GNU licenci, ale jeho vývoj je společností Mentor sponzorován a následně využíván v komerčním prostředí). Stažitelný je pod podmínkou registrace z oficiálních stránek společnosti Mentor. Jeho použití je však volné i v komerční sféře.

Eclipse for C/C++ development

Je verze prostředí Eclipse, která umožňuje vývoj C aplikací a je středobodem celé myšlenky. Slouží jako editor, který následně využívá externí kompilace v gcc a přes Open OCD a GDB server provádí komunikaci a debug HW prostředku. Eclipse dále využívá některé další balíky, které umožňují vytváření projektů s přednastavením pro ARM procesory a integrují do projektů důležité knihovny. Prostředí Eclipse je dostupné rovněž zdarma.

Open OCD

Prostředek vytvářející virtuální server, který zprostředkovává a udržuje komunikační linku mezi prostředím a externím debug HW (konkrétně ST-link V2.0) na různých protokolech (zde konkrétně Ethernet port 3333). Přes tento server zasílá prostředí Eclipse veškeré příkazy. Open OCD se zároveň stará o konfiguraci komunikačních rychlostí a o zasílání debug příkazů podle přicházejících povelů z Eclipse.

Pro Open OCD bylo nutné napsat veškeré komunikační sekvence jednak pro spojení s ST-linkem a jednak pro základní konfigurací procesoru (tyto sekvence jsou tedy přímo pro konkrétní procesory STM) – Open OCD provede totiž zastavení jádra a základní debug výpis procesoru (počty breakpointů a watchpointů…)

GDB server

Vytváří klientskou část aplikace. Vysílá na portu 3333 a uskutečňuje samotnou debug command line. Stará se o vysílání a přijímání debug příkazů skrze linku vytvořenou pomocí Open OCD. GDB server je jedna ze součástí balíku Sourcery Codebench.

Tento aplikační balík je zároveň multiplatformní (jelikož v podstatě všechny použité součásti jsou původně psané pro systémy Linux a až následně kompilovány pro OS Windows), což mu dává velkou sílu přenositelnosti řešení, jak mezi verzemi OS Windows, tak na platformy Linux. V současné době je toto řešení již přineseno i na KAE ZČU a bude používáno pro výuku vývoje aplikací pro procesory od STM s ARM jádry.

Zároveň bude pokračovat snaha o zdokonalení tohoto systému a jeho přenesení do různých distribucí Linux systémů.

4. Komunikace

Komunikace použité v projektu

Nejlépe z výběru vyšli Ethernet a CAN, jelikož jde o protokoly, které jsou volné a licenčně nevázané. Nevzniká tak jakékoli riziko omezení možností, jako se tomu stalo například u sítí typu LON.

Struktura sítě – Jak již bylo naznačeno výše, byla jako nejvýhodnější struktura vybrána kombinace mezi topologiemi hvězda a strom – tudíž rozvětvená.

Problém volby komunikačního protokolu – Poměrně dlouho dobu bylo uvažováno nad protokolem Ethernet, kvůli jeho vysoké přenosové rychlosti (umožňoval datové toky multimediálního obsahu jako třeba videí z kamer apod.) a snadné propojitelnosti k případným koncentrátorům dat nebo počítači, který by získaná data za sítě ukládal rovnou do databáze, kde by data byla dále zpracovávána. Značná komplikace je ovšem v implementaci tak rozsáhlého protokolu – sice existují i poměrně kompaktní stacky přímo pro procesory STM, které jsou běžně na těchto procesorech používány, ale jejich velikost a složitost do značné míry znemožňuje aplikaci na méně výkonných jádrech a úplně vylučuje aplikaci na 8-bitových jádrech. Navíc Ethernet periferie jsou integrované pouze do výkonných procesorů a to by znamenalo jejich nasazení u všech aktivních prvků v síti. Nehledě na zbytečnou ekonomickou náročnost řešení by to znamenalo na každé desce vystavět Ethernet budiče.

V síti je potřeba i mnoho jednoúčelových a jednoduchých prvků, které mohou zastat i 8-bitové procesory. Přináší to ekonomickou i energetickou úsporu. Pozornost se tedy upřela na sběrnici CAN.

CAN periferie jsou integrování téměř do každé řady procesorů STM od 8 po 32 bitové. Implementace budičů je mnohem snazší a programová obsluha rovněž. Není třeba v procesorech držet velké řídící stacky, které by po většinu času stejně nevyužívaly své rozmanité možnosti. CAN naopak přináší větší spolehlivost a možnosti integrace. Jedinou daní za použití CAN je nutnost v rozsáhlejších sítích používat více mezi sebou komunikujících koncentrátorů a výrazně nižší přenosové rychlosti. Není tak zde možné přenášet multimediální obsah, ale pouze provozní informace celého systému.

Z pohledu provozních informací se tedy CAN jeví rozhodně výhodněji. V případě potřeby přenosu multimediálního obsahu nebo větších datových kvant lze použít pospojování pouze některých jednotek a Ethernet komunikaci umožnit nezávisle na hlavní provozní sběrnici. Vzniká tak ideální kombinace, kterou se vydávají moderní trendy. Např. v automobilovém průmyslu je dnes již naprosto standardní, že provozní a datové sběrnice využívají rozdílných protokolů. Pro každý typ přenosu je tak využíván právě takovým protokol, který nejvíce odpovídá charakteru přenášených dat.

4.1. Komunikace SPI (Seriál Pheripheral Intarface)

Jedna ze sériových a velice jednoduchých komunikací, jíž je možné propojit dvě a více jednotek. Většinou v uspořádání, které je použito i v tomto případě, kde je jeden master a jeden případně více slave bodů. Komunikace má čtyři signály:

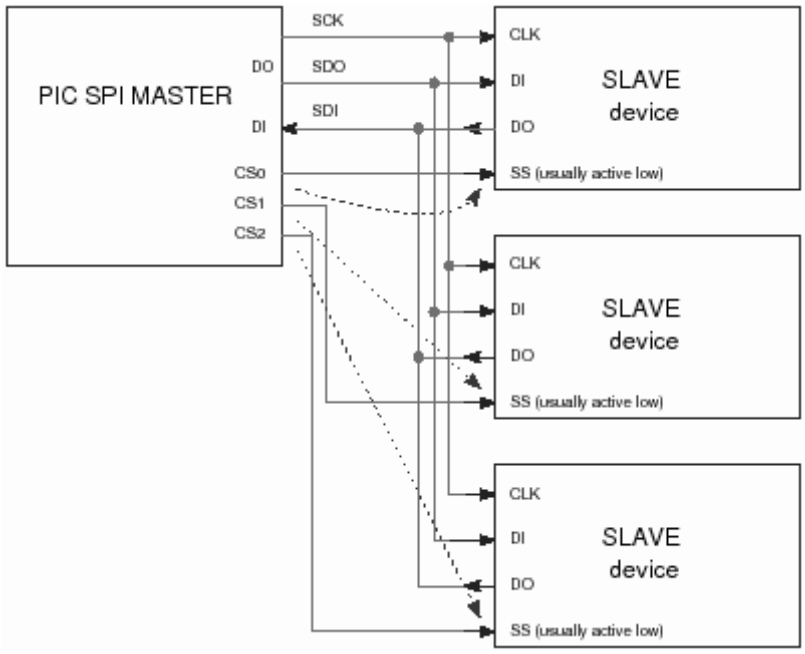
MISO – master in slave out

MOSI – master out slave in - oba slouží pro přenos mezi dvěma uzly ve směru od i k master jednotce – jedná se o plně duplexní výměnu dat. Občas se značí DI a DO.

SCK – hodinový signál udávaní master jednotkou – přenos je tedy synchronní

SS – slave select - je signál generovaný rovněž master uzlem a slouží pro výběr slave jednotky – někdy je také značený CS – chip select a je obvykle aktivní v nízké úrovni.

Komunikace SPI je velice jednoduchá, ale lze ji použít jen pro komunikaci na krátké vzdálenosti a má nevýhodu, že se jedná o neověřovaný přenos – tj. nemá ACK bity a nelze zpětně ověřit přijetí informace. Nicméně pro komunikaci s čidly po jednom integrovaném modulu je SPI naprosto dostačující.

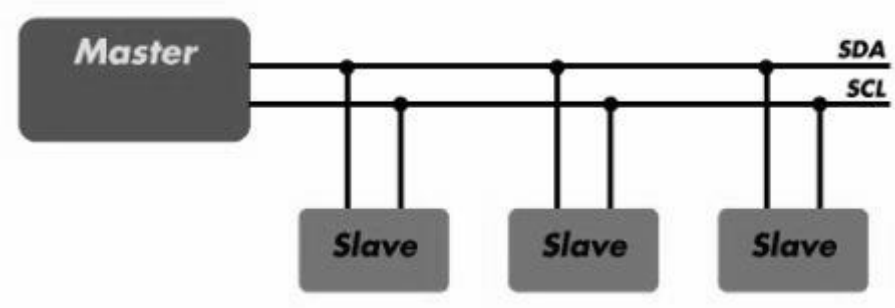


Obrázek 10 – př. Komunikace SPI

Zdroj: www.root.cz

4.2. Komunikace I2C (Inter-Integrated Circuit)

I2C je složitější komunikace na obsluhu i obvodovou náročnost zařízení. Je zde opět hodinový signál vybavovaný master o frekvenci 100 nebo 400 kHz. Datový vodič je jen jeden (SDA) a jedná se proto o poloduplexní přenos. Neexistuje HW výběrový signál, ale všechny jednotky jsou oslovovány podle definované adresy.



Obrázek 11 – blokové schéma komunikace I2C

Zdroj: www.hw.cz

I2C je v projektu ze stejného důvodu jako SPI – pro komunikaci s některými čidly. Tyto dva způsoby komunikace totiž zajišťují dostatečně široké a škálovatelné portfolio digitálních senzorů připojitelných ke koncovým buňkám. V případě nutnosti připojení paměti EEPROM ke koncové buňce lze I2C dokonce využít pro spojení s touto pamětí.

V případě potřeby lze I2C ještě kombinovat s vyšší vrstvou v podobě API knihovny a to konkrétně s I2C CPAL (Communication Peripheral Application Library). Tato knihovna vytváří vyšší řídící vrstvu nad rozhranním I2C a stará se o kontrolu komunikace. Vystavuje „busy flags“, přepíná módy mezi Master a Slave, obsluhuje přerušení a stará se o error management. Tuto knihovnu se vyplatí použít až při složitějších komunikačních procesech. Zde je velikost a složitost knihovny vyvážena přínosem na usnadněné režii komunikace. CPAL je proprietární knihovna od ST nikterak nelicencovaná a lze ji využívat volně.

4.3. Komunikace CAN (Controller Area Network)

Sběrnice původně vyvinutá pro automotive oblast firmou Robert Bosh GmbH. Sloužila hlavně pro spojení jednotlivých elektronických zařízení ve vozidlech, ale rychle expandovala do mnohých dalších průmyslových odvětví např. i do průmyslové automatizace. Její vysoká míra zabezpečení přenosu informace ji předurčuje pro použití s vysokými nároky na bezpečnost. Pro svou nižší přenosovou rychlost je vhodná právě pro přenos různých provozních informací a příkazů či sběr dat.

CAN byl do projektu zvolen, protože téměř všechny řady procesorů od ST disponují CAN periferií a po přidání CAN budičů lze velice snadno a rychle realizovat funkční spojení. V případě vývoje aplikací na Eval-Board jsou navíc budiče již zapojeny a jednotlivé desky lze spojovat tedy okamžitě. Zároveň je CAN velice dobře slučitelný s vybranou topologií sítě.

Charakteristika CAN

- Sdílená broudcastová sběrnice s prioritizací dle identifikátoru
- Multimaster síť
- Pevně definovaná odezva systému
- Detekce vadných zpráv a režie nápravy – znovu vyslání – je implementována na HW úrovni.
- Detekce a správa defektních jednotek rovněž řízena na HW úrovni
- Pro CAN není jasně definované přenosové medium

CAN je založen na „broudcastovém“ principu komunikace – to znamená, že vyslané pakety se šíří celou sítí a až na základě adres a vstupních filtrů jsou jednotlivými jednotkami zpracovávány – nicméně k přijetí zprávy dojde (to má značnou výhodu právě pro spolehlivost komunikace, protože správnost zprávy – zejména syntaktickou – ověřují veškeré jednotky v síti). Tento princip zároveň umožňuje přidávání nebo změnu konfigurace jednotek s maximální jednoduchostí, bez jakýchkoli hardwarových a softwarových úprav sítě. Pomocí identifikátorů lze zároveň velice snadno určovat prioritu jednotlivých buňek nebo celých skupin.

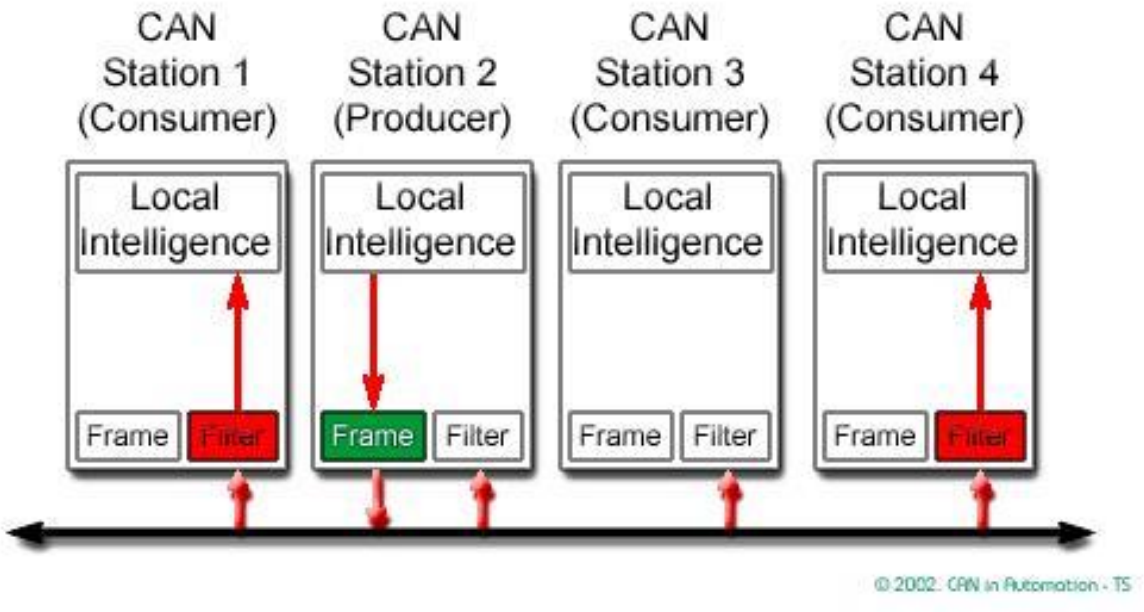
Vzdálenost a rychlost komunikace je definována následující tabulkou:

Tabulka 1 – rychlosti komunikace CAN v závislosti na délce sběrnice

Rychlost přenosu	Maximální délka sběrnice
5 kBit/s	10 km
10 kBit/s	6,7 km
20 kBit/s	3,3km
50 kBit/s	1,3 km
100 kBit/s	620 m
125 kBit/s	530 m
250 kBit/s	270 m
500 kBit/s	130 m
1 MBit/s	40 m

Schéma vyslání a přijetí zprávy (Linková vrstva):

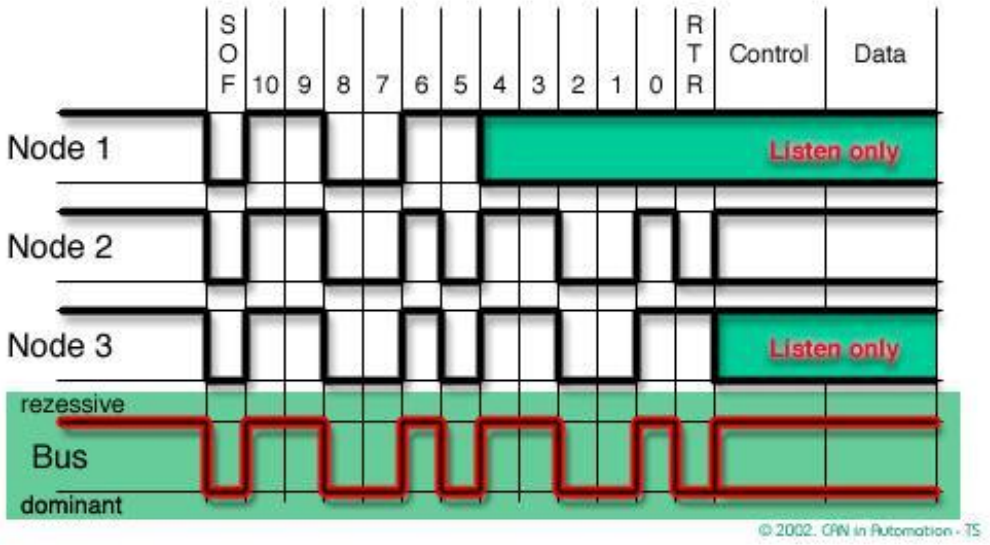
Vyslaná zpráva je přijata všemi stanicemi (V příkladu je odesilatelem stanice 2). Stanice zprávu zkontrolují, ale ke zpracování dojde jen u stanic, které mají nastavený filtr na typ příchozí zprávy. Ostatní stanice se k SW zpracování zprávy nedostanou.



Obrázek 12 – př. Sledu komunikace CAN

Zdroj:www.can-cia.org

Na následujícím obrázku je pak vidět, jak stanice zavěšené na sběrnici naslouchají příchozí zprávě a na základě identifikátoru opouštějí stanici, po zjištění, že zpráva neprojde jejich filtrem. Následně jen naslouchají a kontrolují správnost zprávy. K tomuto dochází ihned po té, co jednotka zjistí, že vyslala recesivní stav, ale na sběrnici je dominantní – znamená to, že vysílá někdo s vyšší prioritou. V případě zjištění nesrovnalostí zasahují stažením sběrnice do dominantního stavu a vynutí opakování zprávy.

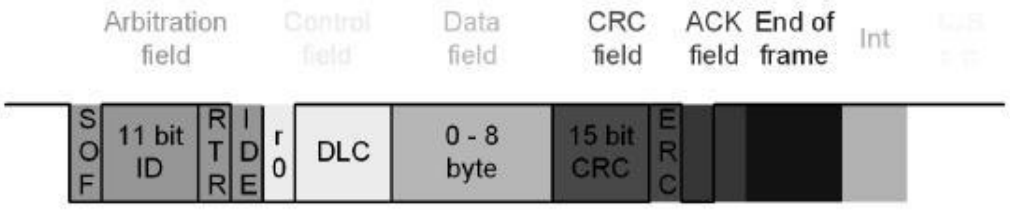


Obrázek 13 – Konflikty na sběrnici CAN Zdroj: www.can-cia.org

To, že může každá jednotka na sběrnici vnutit dominantní stav (úroveň 0) je díky připojení na sběrnici na principu otevřeného kolektoru.

CAN 2.0A

Následující dva standardy CAN se liší délkou identifikátoru. CAN2.0 A má 11-bitový identifikátor.

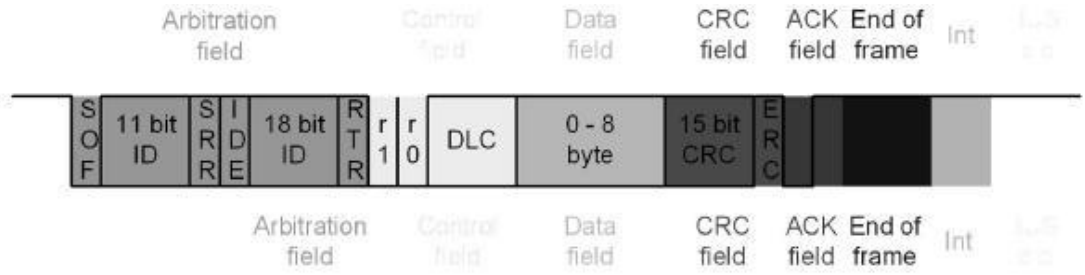


Obrázek 14 – komunikační rámec CAN 2.0A Zdroj: skripta řídící a informační sběrnice Ing. Petr Krist Phd.

- SOF – start of frame
- ID – identifikátor
- RTR – Remote Transmition Request – definice zda zpráva je nositelem dat nebo je to požadavek na data
- IDE – identifikace běžného a rozšířeného formátu
- r0 – nevyužitý bit
- DLC – informace o počtu datových bitů
- Data field – přenášená data
- CRC – kontrolní kód přenášeného rámce
- ERC – ukončení kontrolního kódu
- ACK – potvrzovací bity – přijímače v celé síti stahují sběrnici do dominantního stavu a potvrzují tak správnost přijetí zprávy. Vysílač, tak může poznat, zda alespoň jedna stanice data přijala.
- EOF – End of frame
- Int – prodleva 3 bity po vyslání rámce

CAN 2.0B

Identifikátor CAN 2.0 B má 29 bitů.



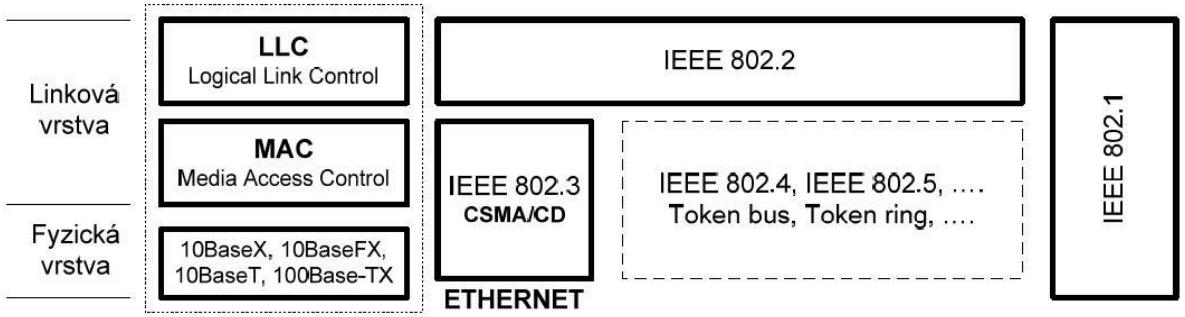
Obrázek 15 – komunikační rámec CAN 2.0B Zdroj: skripta řídící a informační sběrnice Ing. Petr Krist Phd.

- SRR – nastavuje prioritu – standardní rámec má přednost před rozšířeným
- r1 – nepoužitý bit

4.4. Komunikace Ethernet

Ethernet má poměrně bohatší historii než CAN a i jeho šíře využití je v dnešní době masivnější. Jedná se o nejrozšířenější protokol v komunikačních počítačových sítích, ale uplatnění nachází i v průmyslu ve verzi Ethernet PowerLinka mnoho dalších. Na fyzické vrstvě nejčastěji využívá metalických nebo optických vedení (v historii koaxiální kabely zakončované terminujícími odpory a dnes zejména kabely o osmi žilách v různém provedení a míry stínění, která je závislá na normě použité rychlosti). Využívá přenosových rychlostí 10Mb/s – 10Gb/s na metalických spojeních a 100mb/s – 1Tb/s na optických vedeních. V podstatě veškerou standardizaci protokolu a fyzických rozhranní má na starosti nadnárodní instituce IEEE, která nenormalizovala původně vyvinutý protokol od společností DEC, Intel a Xerox.

Struktura protokolu:



Obrázek 16 – Struktura protokolu CAN

Zdroj: skripta řídící a informační sběrnice Ing. Petr Krist Phd.

Na fyzické vrstvě, jsou definice norem neustále doplňovány s postupujícím vývojem a posunem možností.

Linková vrstva může mít u Ethernetu mnoho podob. Pro všechny přístupové metody MAC je pak společná podvrstva řízení logických spojů LLC (logical link control).

Transportní vrstva je realizována rovněž mnoha protokoly (např. IPv4, IPv6, ARP a mnoho dalších).

Nad transportní vrstvou jsou už jen aplikační protokoly užívané v konkrétních aplikacích (z nejznámějších např. POP3, SSH, FTP, DHCP, DNS…).

LLC

Součást linkové vrstvy, která se stará o linkové spojení – tzn. vytváří a ukončuje spojení, stará se o adresaci spojení, kontroluje chyby a poskytuje přenosové služby. V případě Ethernetu realizuje tyto funkce mezi MAC a vyššími vrstvami.

MAC

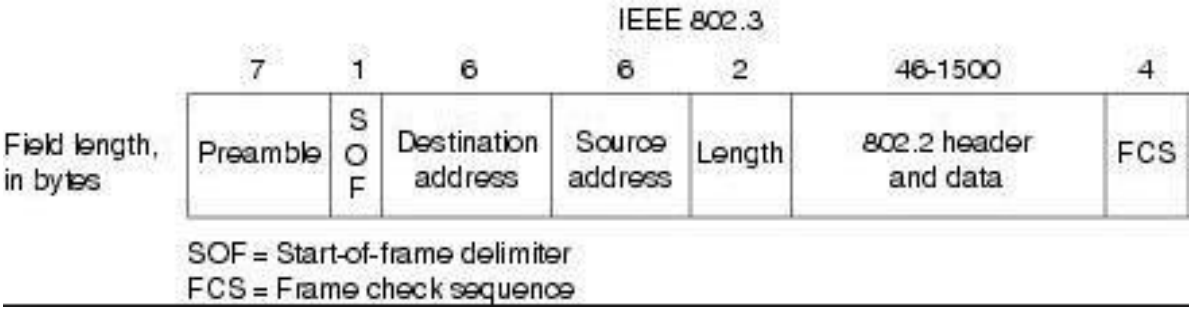
Vrstva MAC zajišťuje přístup koncového zařízení (komunikační stanice) ke sdílené sběrnici. V Ethernet protokolu se používá metoda náhodného přístupu CSMA/CD (Carrar sense multiple access / collision detection).

Na sběrnici může vysílat pouze jedna stanice v jeden okamžik. Na základě výše uvedeného principu pak při kolizních stavech stanice odstupují od sběrnice a vysílají kolizní sekvenci (JAM signál – alternující 0 a 1). Ke sběrnici přistupují až po náhodném čase (ten vychází z tzv. slot time). Zároveň je hlídáno, zda k těmto stavům nedochází neúměrně často, což by silně snižovalo kapacitu sběrnice a ukazovalo by to na problém s fyzickou vrstvou – nejpravděpodobněji špatný návrh kabeláže a tím vysoké doby přenášení rámců mezi stanicemi na sběrnici.

Kolizní doména se rozděluje pomocí routerů a switchů, tak aby se snížil počet stanic na sběrnici a nedocházelo k tak častým kolizím.

Podrobnější popis těchto vrstev není předmětem práce a je možné jej dohledat v konkrétních normách instituce IEEE.

Formát Ethernet rámce



Obrázek 17 – komunikační rámec Ethernet

Zdroj:www.cisco.com

Čísla nad jednotlivými částmi jsou počty oktetů (oktet je skupina 8 bitů – bylo stanoveno pro zajištění kompatibility mezi systémy používajícími různou základní dálku bytů).

- Preambule – alternující bity pro zachycení hodin příjemce
- SOF – start of frame – značí začátek rámce
- Destination addres – MAC adresa příjemce
- Source address – MAC adresa odesilatele
- Lenght – určuje délku pole, které nese data
- Data – datový obsah o délce 46 – 1500 oktetů
- FCS – frame check semence – jedná se o kontrolní CRC součet. Do součtu je počítán celý rámec bez Preamble a na jeho základě jsou rámce rozdělovány na zdravé a vadné

Minimální mezera mezi dvěma rámci je 12 oktetů.

lwIP

Jedná se o nejrozšířenější opensource IP stack původně vyvíjený Adamem Dunkelsem. Dnes vývoj přešel pod skupinu lidí kolem Kierana Mansleye. Mírně se změnila i licence nicméně projekt zůstal otevřený. Tento stack je využíván mnoha společnostmi pro implementaci TCP funkcionalit do embedded systémů. Pro svou velikost je poměrně snadné ho nahrávat i do zařízení s omezenou pamětí (po kompilaci stack zabírá přibližně 40kB paměti). I přes tuto „malou“ velikost se stále jedná o jeden s největších a nejsložitějších stacků pro implementaci běžných protokolů do mikroprocesorů.

lwIP má implementované tyto protokoly:

-IP, ICMP, UDP, TCP, IGMP, ARP, PPPoS, PPPoE

A tyto funkce:

-DHCP client, DNS client, AutoIP/APIPA (Zeroconf), SNMP agent (private MIB support)

- API: specialized APIs for enhanced performance, optional Berkeley-alike socket API

- Rozšířené funkcionality: IP forwarding over multiple network interfaces, TCP congestion control, RTT estimation and fast recovery/fast retransmit

- Přídavné aplikace: HTTP server, SNTP client, SMTP client, ping, NetBIOS nameserver

Do dne sepsání práce se podařilo implementovat funkčnost DHCP client, posílání paketů a http serveru. Na dalších funkcích se bude pracovat v následující době.

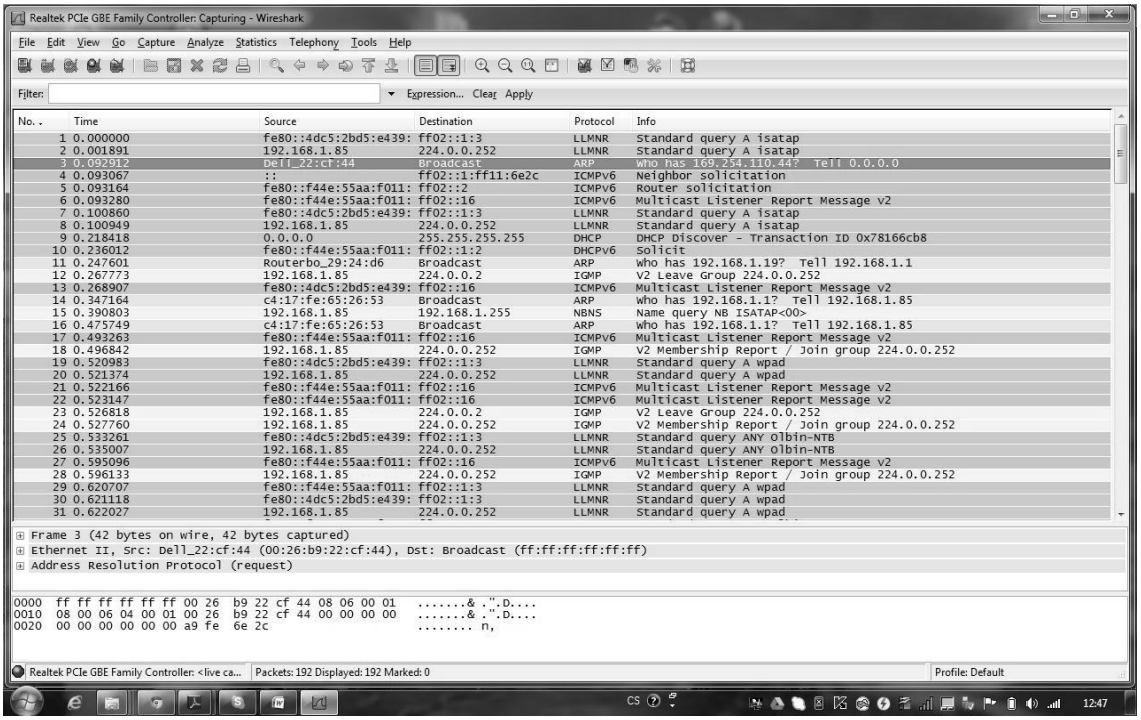
Procesory od ST mají integrované MAC jednotky jako jednu z periferií. Na Evaluačních deskách od ST jsou pak pro pro komunikaci mezi MAC a Ethernetem používané Ethernet Physical Layer Trensievery DP83848 s oddělovacím transformátorem a krystalem.

4.5. Nástroje pro práci s CAN a Ethernet

Ethernet:

Pro zachytávání Ethernet rámců je používán program Wireshark. Program je pod GNU licencí, ale svou funkcionalitou předčí i výborně vybavené profesionální programy.

Jedná se o software, který zachytává a zobrazuje Ethernet datagramy, umí je filtrovat podle obsahu nebo podle protokolů a umožňuje základní práci s rozborem rámců (zobrazovat obsah jednotlivých částí rámce). Veškerý provoz v síti lze ukládat a provádět rozbor později. Výhodou je, že program zachytí a označí i nevalidní datagramy. S jeho pomocí se dá poměrně snadno odlaďovat jakákoli aplikace komunikující po Ethernetu. Jsou zároveň vidět reakce ostatních zařízení, jako jsou např. směřovače.



Obrázek 18 – screenshot zachytávání rámců pomocí programu Wireshark

Zdroj: Vlastní tvorba 2013

CAN

Pro zachytávání komunikace CAN byl použit starší nástroj ST Combox. Ve většině případů, ale docházelo k zachytávání přímo dalšími procesory, jelikož komunikace přes CAN není až tak náročná a nevyžaduje zvláštní péči při odlaďování rámců.

5. Zajímavé periferie Využívané v projektu

5.1. DMA

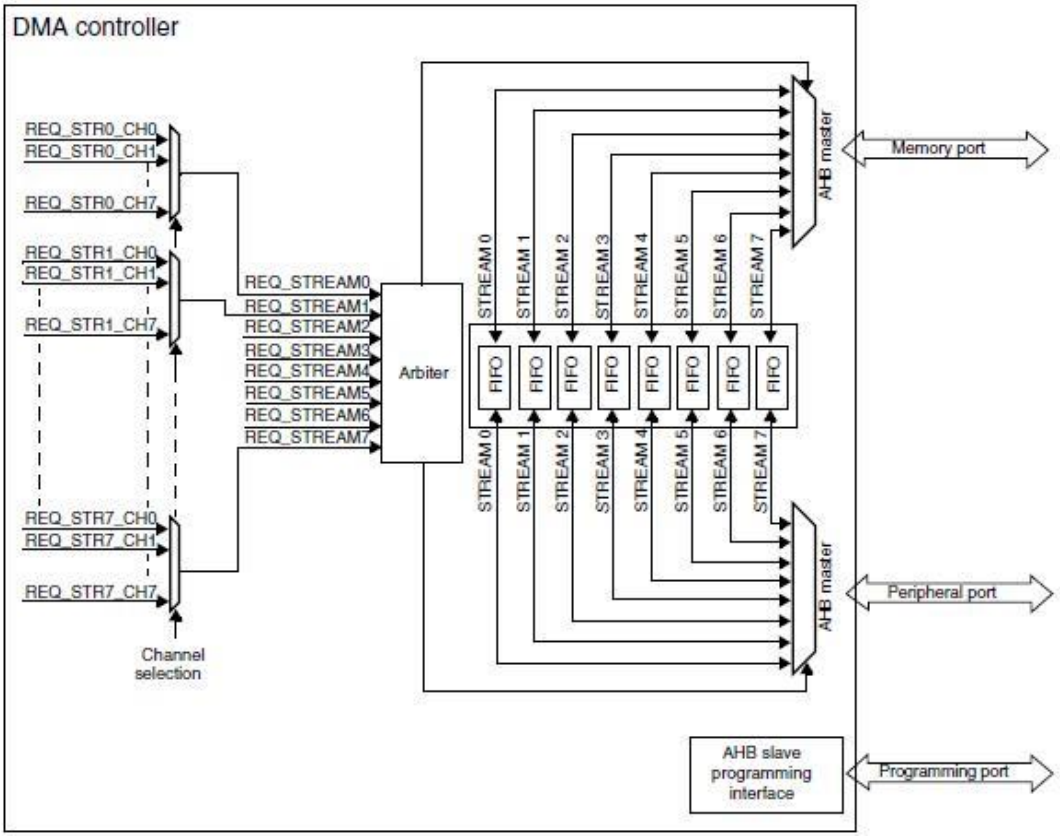
Popis DMA

(Direct Memory Access – přímý přístup do paměti)

Používá se pro zprostředkování vysokorychlostních přenosů mezi oblastmi paměti nebo mezi pamětí a periferiemi. Data jsou velice rychle přesouvána bez jakékoli účasti procesoru nezávisle činnosti jádra. To umožňuje nechávat veškeré prostředky jádra volné pro ostatní operace.

Dva DMA kanály mají dohromady 16 streamů, z nichž každý je namapovaný na nějaké místo v paměti a periferiích a každý stream může mít 8 různých zdrojů centů. Každý má i arbitráž priorit jednotlivých DMA požadavků. Všechny požadavky jsou vyvedené do celkové propojovací matice.

Zde je vidět vnitřní struktura DMA řadiče a propojovací matice jednotlivých stramů. Interní triggery pro DMA mohou pocházet v podstatě z jakéhokoli zdroje (tedy periferie, která má výstupní trigger TRGO).



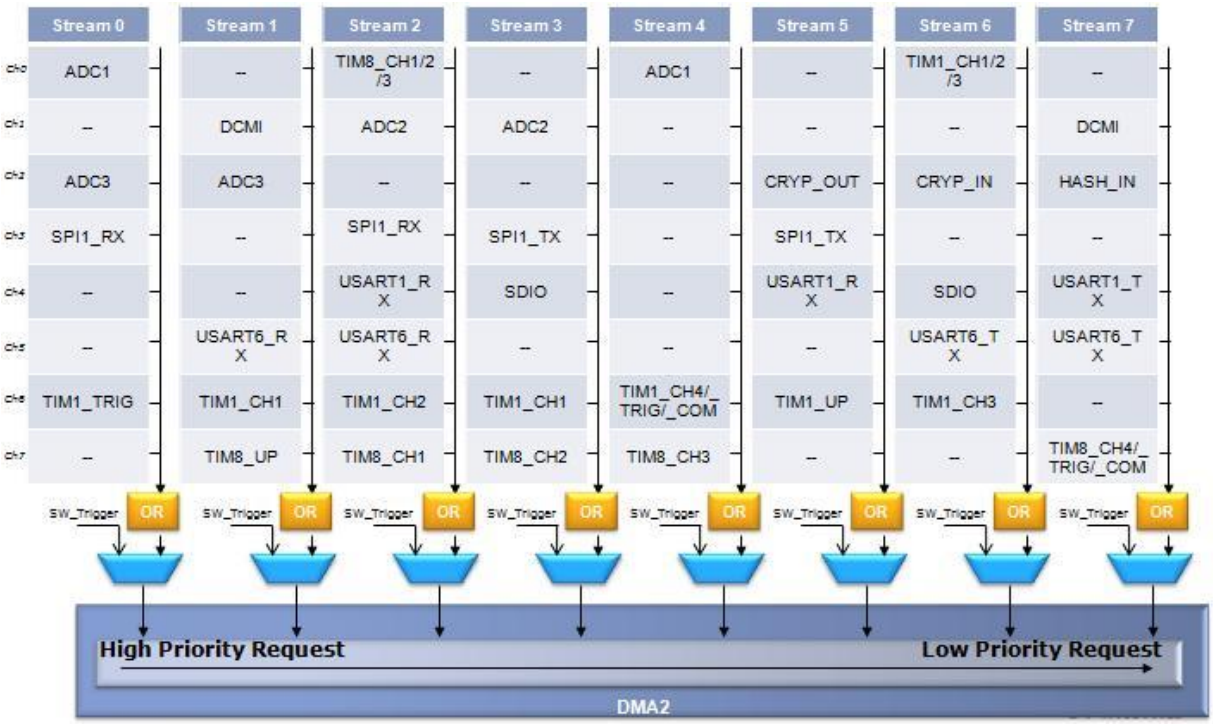
Obrázek 19 – Propojovací matice DMA řadiče

Zdroj: datasheet DS STM32F4xx

Základní vlastnosti DMA v STMF4xx

8 stream pro každý DMA řadič a 8 kanálů pro každý stream. U F4 jsou přítomny 2 DMA řadiče

- 4x32 bitová FIFO paměť pro každý stream
- Cíl i zdroj přenosu může mít rozdílnou šířku paměti
- Pro přenos typu memory-to-memory je podporován softwarový trigger
- V průběhu přenosu lze inkrementovat cílovou nebo zdrojovou paměť, případně obě
- Umožňuje dávkové přenosy
- Pro každý stream je k dispozici 5 event flagů
- 4 úrovně priorit mezi použitými stream



Obrázek 20 – namapování periferií na kanály jednotlivých stremů DMA

Zdroj: Školící materiály společnosti STM pro řadu STM4xx

Na tomto obrázku je vidět vnitřní napojení jednotlivých periferií na jednotlivé streamy druhého kanálu DMA. První kanál DMA je velice podobný jen chybí softwarové triggery. U jednotlivých periferií se musí povolit interní linka k DMA řadiči a po aktivování daného streamu a kanálu dochází při eventech na periferii k aktivaci DMA přenosu. Jednotlivé fáze

přenosu si lze hlídat testování flagů stavu DMA a určovat tak dokončené přenosy případně vyvolávat přerušení (flagy DMA: DMA half transfer, DMA transfer komplete, DMA transfer error, DMA FIFO error, direkt mode error).

Nastavení DMA v CMSIS

Následující kód nastavuje DMA přenos z interního bufferu do externí RAM paměti a po dokončení přenosu vyvolá přerušení. Z příkladu je patrné jak jednoduché nastavení DMA přenosu je. Struktura nastavování a uvedení DMA do provozu je totožná pro všechny typy přenosu.

```
DMA_DeInit(DMA1_Channel6);
DMA_InitStructure.DMA_PeripheralBaseAddr = (uint32_t)SRC_Const_Buffer;
DMA_InitStructure.DMA_MemoryBaseAddr = (uint32_t)DST_Buffer;
//Výběr zdrojové a cílové adresy přenosu
DMA_InitStructure.DMA_DIR = DMA_DIR_PeripheralSRC;
DMA_InitStructure.DMA_BufferSize = BufferSize;
//Volba směru přenosu a velikos použitéhu bufferu
DMA_InitStructure.DMA_PeripheralInc = DMA_PeripheralInc_Enable;
DMA_InitStructure.DMA_MemoryInc = DMA_MemoryInc_Enable;
//Volba inkrementace adres při přenosu – zde se obě inkrementují tzn. přenáší se obsah paměti
//na jiné místo o stejné paměťové šířce
DMA_InitStructure.DMA_PeripheralDataSize = DMA_PeripheralDataSize_Word;
DMA_InitStructure.DMA_MemoryDataSize = DMA_MemoryDataSize_Word;
//šířka přenášených bloků je na obou stranách jedno slovo
DMA_InitStructure.DMA_Mode = DMA_Mode_Normal;
DMA_InitStructure.DMA_Priority = DMA_Priority_High;
DMA_InitStructure.DMA_M2M = DMA_M2M_Enable;
//nastavení módu a priority přenosu
DMA_Init(DMA1_Channel6, &DMA_InitStructure);
//inicializace celé struktury – v tomto bodě dochází k nahrání zvolených nastavení do registrů
periferie
/* Enable DMA1 Channel6 Transfer Complete interrupt */
DMA_ITConfig(DMA1_Channel6, DMA_IT_TC, ENABLE);
//Zapnutí přerušení vyvolaného DMA při stavu Tranfer Complete (Dokončení přenosu)
/* Enable DMA1 Channel6 transfer */
DMA_Cmd(DMA1_Channel6, ENABLE);
//Zapnutí DMA – předpokládá předešlou aktivaci hodin pro tuto periferii
```

Obsluha DMA přenosů u procesorů STM je velice snadná a lze jí realizovat i mnoho paralelních přenosů, které ušetří mnoho prostředků jádra. Například v přiloženém případu přenosu dat z DCMI pomocí FSMC by byl přenos takového objemu dat jen těžko realizovatelný jiným způsobem. Vzhledem k jednoduchosti využití a elegantnosti výsledného řešení jsou DMA přenosy v projektů používány velice hojně a ve spojení s mnoha periferiemi.

5.2. DCMI

(Digital camera interface)

Periferie přímo uzpůsobená pro kombinaci s digitálními kamerami a pro odesílání obrazu z kamer do paměti. Jedná se o paralelní synchronní rozhranní schopné přenášet vysokorychlostní toky dat z externích 8, 10, 12 nebo 14 bitových CMOS kamer. Podporuje přenos surových nekomprimovaných dat nebo přenos komprimovaných obrazových dat ve formátu JPEG (některé kamerové moduly umí konvertovat obrazová data přímo do JPEG na svém čipu).

Rychlost přenosu je až 30 snímků za vteřinu pro rozlišení VGA tj. 640x480 pixelů. V kombinaci s funkcionalitou FSMC a DMA lze přenášet celý obrazový buffer na jiné místo v paměti procesoru, případně do externí paměti, jako je tomu v přiloženém příkladu.

5.3. FSMC

(Flexible Static Memory Controller)

Popis

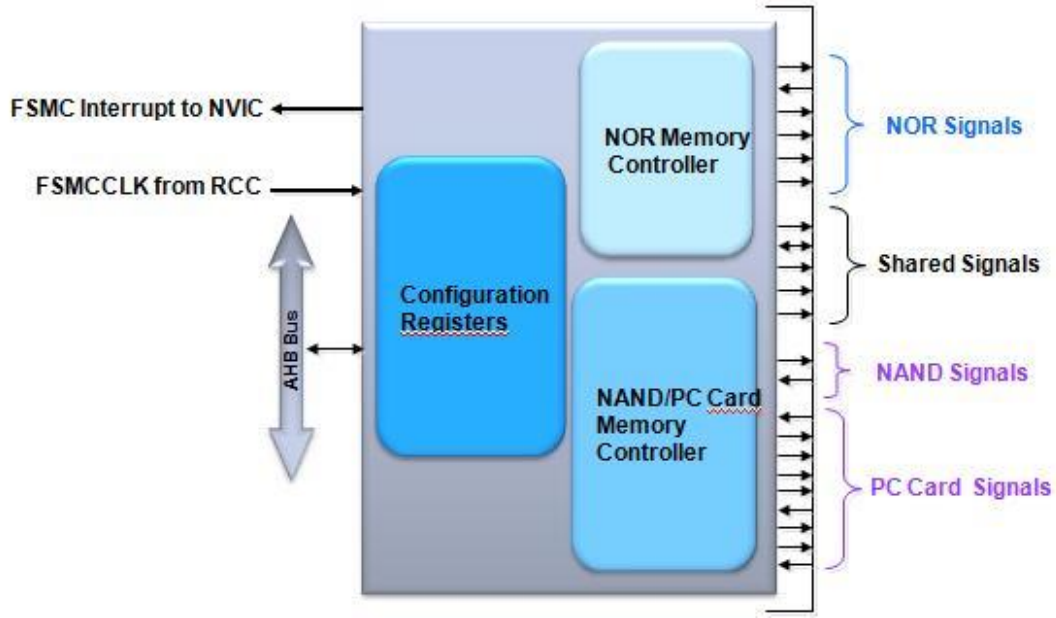
Blok, který je schopen komunikovat se synchronními i asynchronními paměťmi a 16-ti bitovými PC paměťovými kartami.

Jedná se o blok zprostředkující převod komunikace mezi interním paměťovým rozhranním na AHB a externím paměťovým prostorem. Nejjednodušeji lze princip vysvětlit jako namapování externích pamětí do vnitřního paměťového prostoru. Veškerý provoz do externí paměti pak probíhá stejně, jako by paměť byla interní – tedy rozšířená. Rychlost této komunikace může být až 60 MHz.

Funkcionalita

- K dispozici jsou 4 Banky podporující různé druhy pamětí
- Nezávislý chip select pro každou banku
- Nezávislá konfigurace pro každou banku
- Rozhranní pro statické paměti zahrnuje podporu SRAM, ROM, NOR, PSRAM
- Podpora paralelních LCD řadičů (Intel 8080 a Motorola 6800) – pomocí této funkce lze velice snadno, v kombinaci s DMA řadičem, zapisovat na LCD displej celou zobrazovací oblast předpřipravenou předem v paměti

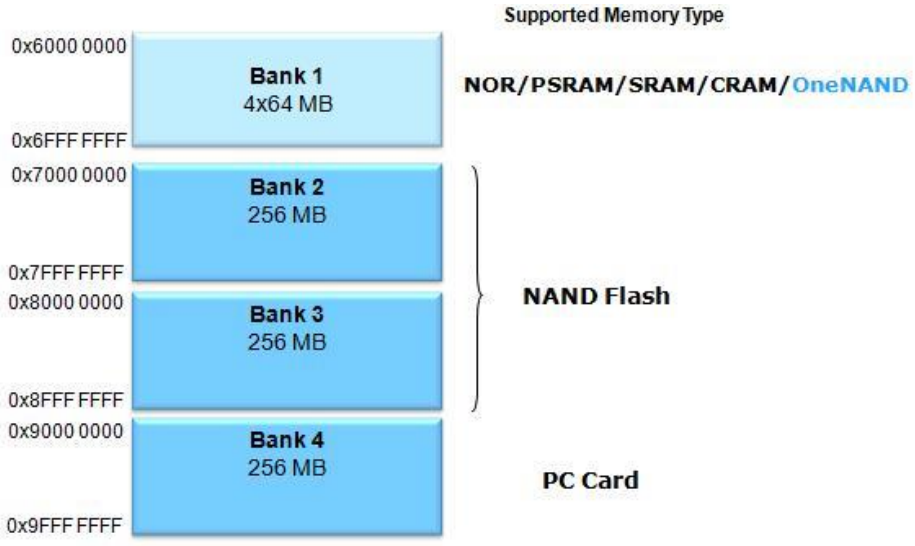
Blokové schéma FSMC bloku



Obrázek 21 – blokové schéma FSMC

Zdroj: Školící materiály společnosti STM pro řadu STM4xx

- AHB Interface
- NOR Flash/PSRAM řadič
- NAND Flash/PC Card řadič
- Řadič externích zařízení



Obrázek 22 – rozdělení paměťové mapy prostoru FSMC

Zdroj: Školící materiály společnosti STM pro řadu STM4xx

Externí adresní prostor je rozdělen na 4 stejně velké části (každá má velikost 256MB – 256 pro PC Card, 2x256MB pro NAND Flash a zbylých 4x64MB pro zbylé typy pamětí.)

5.4. Další zajímavé vlastnosti:

Spread spektrum clock generátor PLL

Speciální funkce fázového závěsu hodinového generátoru. Umožňuje mírně oscilovat se zavěšeným kmitočtem, tak aby výsledný hodinový signál pro jádro procesoru a byl posouván okolo hlavní frekvence.

S touto frekvencí samozřejmě ujíždí i veškeré časování periferií. Na standardní běh procesoru nemá toto ujíždění frekvence víceméně žádný vliv. Problém by mohl nastat u komunikačních rozhranní závislých na okolních zařízeních.

Cílem celého přelaďování kmitočtu je snížení rušení procesorem na hlavní hodinové frekvenci. V případě vykonaného testu byly zkoušeny různé nastavení SSCG a sledováno rušení do okolí. Následně byly testovány vlivy na komunikační periferie.

Přelaďování lze velice citlivě nastavovat jednak v pohledu šíře pásma, ve kterém lze kmitočet přelaďovat ale i to v jaký časových intervalech bude průběh jedné periody přeladění – to umožňuje přesněji nastavovat přelaďování vhledem k době trvání jednoho komunikačního rámce.

Testovací kód s projektem je součástí příloh na CD stejně tak, jako výsledné grafy ze spektrálního analyzátoru.

6. Současný stav a další rozvoj práce

Veškeré moduly mezi sebou komunikují přes CAN. Zachytávání rámců probíhá zatím pouze do konzole v PC přes Combox. Protokol CAN použitý v projektu je zatím využíván jak pro přenos systémových a datových zpráv. Moduly s Ethernet rozhranním jsou schopné vyměňovat diagramy s informacemi (zatím neposílají jakýkoli multimediální obsah), ale jsou schopné fungovat v kombinaci s komerční síťovou technikou (reagují na ARP protokol a jsou schopné vyzískat dynamické adresy z DHCP, případně komunikovat na pevných IP adresach).

V současné době jsou naprogramovány moduly:

- Pro snímání teplot (venkovních i vnitřních)
- Snímače stmívání (venkovní osvětlení)
- Vypínání a zapínání zásuvek s přednastavením na časový plán a vyblokování některých zásuvek pro trvalou aktivaci
- Logické vstupy pro detekci otevřených dveří a oken
- Ovládání elektroventilů řídících přívod vody do topných těles
- Regulace teploty (výstupy na ventily i infrazářiče – případně další elektrické varianty vytápění. Nastavení tepelné hystereze)
- Modul pro řízení stahování venkovních rolet (logické výstupy pro externí systém, vč. zpětné vazby stavu)
- Řízení nastavení stínící techniky (Podle intenzity osvětlení – možnost ručního zásahu a nastavení časové hystereze, chování upraveno podle rychlosti větru)
- Ovládání elektroventilů pro zahradní systém na základě údajů z čidel vlhkosti a senzorů deště
- Venkovní vstupní brána řízená radiovým signálem 868Mhz (s laserovou závorou)

Další moduly ve vývoji:

- Ethernet komunikace schopná přenášet signál z DCMI, případně již komprimované JPEG obrázky
- Detektory pohybu pro nezabezpečovací účely (osvětlení interiérů i exteriérů při detekci pohybu)

- Bezdrátové moduly založené na ZigBeeIP protokolu a procesorech s bezdrátovou periferií – ve stádiu validace produktu v STM
- Moduly zabezpečovací – detektory pohybu, otřesů, laserové závory, případně aktivace bezpečnostních kamer

Celá komunikace by měla být přepracována na některý ze složitějších aplikačních protokolů nad CAN komunikací. Například na CANopen nebo nějakou jeho modifikaci tak, aby bylo možné programovat moduly dynamičtějším způsobem. Vize takovéhoto přístupu spočívá v tom, že moduly by po připojení do sítě měly automaticky ohlásit svou přítomnost a svou funkcionalitu. Zbytek sítě by se měl modulu přizpůsobit a začít s ním komunikovat a nově připojený by měl dostat i adresu dle segmentu, kam byl připojen.

Knihovna by neměla řešit jen připojování nových stanic, ale i vhodným způsobem informovat o problémech na sběrnici – posíláním diagnostických zpráv a testováním funkcionalit jednotlivých bloků.

Dalším cílem by byl podsystém, který by mohl vykonávat zabezpečovací a alarmové funkce, případně spojení s kamerovým systémem nebo pultem nějaké bezpečnostní agentury. Takovéto systémy již vyžadují mnohem vetší spolehlivost funkčnosti a zároveň šifrovanou komunikaci. Ze strany zadavatele práce byl navržen koncept založený na dynamickém generování bezpečnostního šifrovacího kódu, takže implementace tohoto zabezpečení bude vyžadovat vhodnou volbu principu dynamické generace a zároveň programovou realizaci takového algoritmu.

Zatím posledním cílem, který je do budoucna stanoven je sestavení alespoň základní databáze zachytávající příchozí data. Toto zachytávání by mělo probíhat nejlépe skrze USB port. Z toho plyne nutnost koncentrátor dat postavit programově tak, aby CAN data jen neshromažďoval a nepředával dál po CAN rozhranní, ale využít USB periferii a příchozí data z CAN překládat tok na USB periferii. Koncentrátor by měl být schopen rovněž data z PC průběžně přijímat a předávat podřízeným jednotkám.

Závěr

Činnost na tomto projektu byla nadmíru obohacující. Práce pro společnost STM přináší velké rozšíření obzorů v možnostech mikroprocesorové techniky a elektroniky vůbec. Na projektu je použito hodně pokročilých technologických možností procesorů z portfolia STM jako téměř všudypřítomné přenosy DMA, mapování externích pamětí nebo práce s multimediálním obsahem. Systém, který v rámci projektu vznikl, disponuje mnoho funkcionalitami a moderními prvky. Již v současné fázi je schopen samostatné činnosti. V průběhu prací se zároveň naskytla příležitost objevit mnoho dalších, bočních, projektů, na kterých byla práce neméně zajímavá a přínosná (například programovací IDE založené na Eclips a podobně – princip po zájmu katedry aplikované elektroniky a telekomunikací).

Na zadané téma v proběhu prací vzniklo mnoho dalších nových požadavků, překračující rozsah původního zadání. V diplomové práci jsou proto zahrnuty principy použité pouze pro vývoj aplikace ohraničené zadáním. Nadstavbové a další rozpracované funkce jsou zmíněny a vysvětleny pouze v poslední kapitole. Vzhledem ke zdárným krokům v rámci projektu bude projekt dále pokračovat a práce na něm podporovány ze strany STM i v budoucnu.

Seznam použité literatury a zdrojů

Datasheet DS STM8S Doc ID 14733 Rev 12

Datasheet DS STM32F0 Doc ID 022265 Rev 3

Datasheet DS STM32F1 Doc ID 15274 Rev 6

Datasheet DS STM32F2 Doc ID 17050 Rev 8

Datasheet DS STM32F4 DocID024244 Rev 1

Reference manual RM STM8SA Doc ID 14587 Rev 8

Reference manual RM STM32F0 Doc ID 018940 Rev 2

Reference manual RM STM32F1 Doc ID 13902 Rev 14

Reference manual RM STM32F2 Doc ID 15403 Rev 5

Reference manual RM STM32F4 Doc ID 018909 Rev 4

Veškeré výše uvedené dokumenty jsou dohádatelné pod Doc ID na stránkách www.st.com a zároveň jsou přiložené jako přílohy na CD v uvedené verzi.

Skripta pro Řídící a informační sběrnice, Ing. Petr Krist Phd.

Seznam obrázků, tabulek (zdroje uvedeny v textu)

Seznam příloh

Datasheet DS STM8S Doc ID 14733 Rev 12

Datasheet DS STM32F0 Doc ID 022265 Rev 3

Datasheet DS STM32F1 Doc ID 15274 Rev 6

Datasheet DS STM32F2 Doc ID 17050 Rev 8

Datasheet DS STM32F4 DocID024244 Rev 1

Reference manual RM STM8SA Doc ID 14587 Rev 8

Reference manual RM STM32F0 Doc ID 018940 Rev 2

Reference manual RM STM32F1 Doc ID 13902 Rev 14

Reference manual RM STM32F2 Doc ID 15403 Rev 5

Reference manual RM STM32F4 Doc ID 018909 Rev 4

CD00291090 – cpal application

Veškeré použité obrázky jsou na CD ve složce pics

Projekty přiložené na CD:

Na CD ve složce *projects* Jsou uložení veškeré projekty členění ve složkách podle verze jádra procesoru. Všechny projekty jsou vyčištěné od výsledků kompilace a připravené v projektech pro více vývojových prostředí. Projekty strukturou a uspořádáním knihoven vychází ze šablony, kterou používá STM ve svých projektech.