ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

fakulta elektrotechnická - katedra řídící techniky

Diplomová práce

Řízení pohonu řezacího stroje pro tvarové dělení materiálu

David Klement

květen 2005

ČVUT Praha

Katedra řídicí techniky ***Školní rok:*** 2003/2004

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: David Klement

Obor: Technická kybernetika

Název tématu: Řízení pohonu řezacího stroje pro tvarové dělení materiálu

Zásady pro vypracování:

1. Nastudujte dokumentaci k řídicímu systému.
2. Navrhněte řízení pohonu stroje.
3. Implementujte a odzkoušejte navržené řízení.

Seznam odborné literatury: Dodá vedoucí práce.

Vedoucí diplomové práce: Ing. Přemysl Šůcha

Datum zadání diplomové práce: listopad 2003

Termín odevzdání diplomové práce: květen 2005

doc. Ing. Michael Šebek, DrSc.
vedoucí katedry

L.S.

v zast.
prof. Ing. Vladimír Kučera, DrSc.
děkan

V Praze dne 25.03.2004

Anotace

Diplomová práce popisuje návrh řídícího hardware pro řízení servopohonu. Řídící jednotka je založena na procesoru SH7727 od firmy Renesas(Hitachi), na níž je použit operační systém SH-Linux. Hlavním vstupně/výstupním rozhraním pro komunikaci se servopohonem je sběrnice CAN (Controller Area Network) realizovaná pomocí řadiče SJA1000 (Philips).

Návrh se skládá z několika částí. Výběr vhodných procesorů a jejich porovnání, návrh použitého řešení, hardwareová realizace s ohledem na elektromagnetickou kompatibilitu, portování a zprovoznění ovladače pro CAN rozhraní.Pro lepší ilustraci je práce doplněna přílohami a ukázkou provedených měření osciloskopem.

Annotation

The diploma thesis describes the hardware design of a control unit for traction control. The control unit is based on the processor SH7727 (Renesas) working on operating system Linux. The main input/output interface is CAN Bus, realized by a stand-alone CAN controller SJA1000 (Philips Semiconductors).

This work describes processor selection, hardware design considering electromagnetic compatibility and finally CAN-driver putting into service. Work is supplemented with appendixes and a sample of measurement with oscilloscope extends the description.

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č.121/2000 Sb. , o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne ……. …….

Podpis

Poděkování

Velmi rád bych poděkoval a vyslovil uznání všem, kteří mi pomáhali při vzniku této práce. Především Ing. Přemyslu Šůchovi, vedoucímu mé diplomové práce za trpělivé vedení a množství praktických rad. Dále Ing. Pavlu Píšovi, který mi vždy ochotně pomohl cennými informacemi.

Zvláštní poděkování patří Ing. Miloslavu Žižkovi, jedinečnému člověku a výbornému odborníkovi, který mi poskytl zázemí potřebné pro vznik celé práce a pomohl svými znalostmi i prostředky.

Nakonec bych chtěl poděkovat rodičům za poskytnuté zázemí a své milované ženě za její trpělivost a lásku.

Obsah

1. Úvod 2
2. Výběr platformy 3
   2.1 Cirrus Logic EP9312 3
      2.1.1 Charakteristika 3
      2.1.2 Funkční blokový diagram 4
      2.1.3 Výhody 4
   2.2 Renesas SH7727 5
      2.2.1 Charakteristika 5
      2.2.2 Funkční blokový diagram 6
      2.2.3 Výhody 6
   2.3 Zhodnocení výběru 7
3. Komunikační rozhraní pro řízení 8
   3.1 CAN - Controller Area Network 8
      3.1.1 Úvod 8
      3.1.2 Vlastnosti 8
   3.2 Fyzické přenosové médium 9
   3.3 Typy zpráv 10
4. Návrh hardware 12
   4.1 SJA1000 12
      4.1.1 Funkční blokový diagram a popis signálů 12
      4.1.2 Charakteristika 13
   4.2 Obvodové řešení 14
   4.3 Návrh plošného spoje 20
   4.4 Osazení DPS 20
5. Software a oživení 21
   5.1 Příprava software 21
      5.1.1 Instalace baličků pro SH3 crosskompilaci 21
      5.1.2 Příprava debuggeru Insight 22
   5.2 Kompilace LinCAN driveru pro SH3 platformu 23
      5.2.1 Vlastní kompilace 24
   5.3 Oživení 26
      5.3.1 Ověření základní funkcionality 26
      5.3.2 Připojení kontroléru SH7727 k PC 26
      5.3.3 Přístup k registrům SJA1000 27
   5.4 Zavedení ovladače LinCAN a jeho funkce 29
      5.4.1 Zavedení ovladače 29
      5.4.2 Otestování funkčnosti 29
6. Závěr 31
7. Literatura 32
8. Přílohy:
   Příloha č.1: Deska plošných spojů interface SH7727IF – strana A
   Příloha č.2: Deska plošných spojů interface SH7727IF – strana B
   Příloha č.3: Osazovací plán deska interface SH7727IF – strana A
   Příloha č.4: Osazovací plán deska interface SH7727IF – strana B
   Příloha č.5: Osazená deska interface SH7727IF

1. Úvod

Hlavní úlohou bylo navržení vhodného řešení kontroléru pro průmyslové řízení servopohonu řezacího stroje pro tvarové dělení materiálu.

Výchozím byl kontrolér navržený panem ing. Miloslavem Žižkou založený na procesoru SH7727 fy. Renesas [1]. Tento kontrolér je již delší dobu využíván pro řízení řezacího stroje pro tvarové dělení materiálu. Princip spočívá v polohování plazmového hořáku, který vyřezává součástky z ocelového plátu. Protože se jedná o poměrně agresivní a ionizované prostředí a materiál je poměrně drahý, jsou kladeny na kontrolér poměrně vysoké nároky co se týče spolehlivosti a funkčnosti.

Úkolem bylo rozšířit daný kontrolér o průmyslové rozhraní (CAN) nebo nalézt ekvivalentní řešení s jiným procesorem. Důležitými faktory byla co nejnižší finanční náročnost a z toho plynoucí komerční využitelnost a dále dostupnost použitých komponent. Nezanedbatelným parametrem je také příkon daného zařízení. Pokud totiž vystačíme bez nucené ventilace tak se snižuje možnost zavlečení drobných částeček a prachu, který pak může snížit spolehlivost a funkčnost.

Klíčovým cílem bylo, aby mohl být využit potenciál LinCAN driveru [8] a jeho portování na SH platformu.

Následující kapitola popisuje kriteria výběru vhodného procesoru pro řídící kontrolér a důvody konečného výběru procesoru SH7727. Ve třetí kapitole následuje popis vybraného komunikačního rozhraní CAN [4] a popis jeho předpokladů.

Další kapitola ukazuje cestu k návrhu hardwareové realizace kontroléru resp. jeho rozšíření o komponentu CAN rozhraní.

V další části je popis přípravy softwareového prostředí potřebného pro kompilaci software pro platformu procesoru Renesas SH7727, popis kompilace debuggeru pro ladění programů a portování ovladače LinCAN [8] na SH7727. A další části kapitoly je popis oživení kontroléru a potvrzení funkce driveru snímkem z osciloskopu.

Práce je zakončena závěrem s popisem dosažených výsledků.

2. Výběr platformy

Tato kapitola se zaobírá možnostmi výběru vhodného procesoru pro řízení technologického procesu v náročnějších podmínkách. Vyběr byl dále postoupen požadavkům jako je dostupnost, cena a výsledné malé rozměry hotového kontroléru a tím i zvýšená odolnost vůči elektromagnetickému rušení.

Nezanedbatelnou úlohu hrála také spotřeba zařízení a s tím související tepelné emise. Proto vybírané procesory měly nízkou spotřebu elektrické energie, aby nebylo nutno výsledné zařízení chladit nucenou cirkulací vzduchu, která zvyšuje možnost znečištění prachem a jinými mechanickými nečistotami.

2.1 Cirrus Logic EP9312

Nejvhodnějším obvodem pro variantu počítající s náhradou stávajícího SH7727[1] byl procesor EP9312[3] (CIRRUS LOGIC). Podstatným kritériem u tohoto procesoru byla jeho dostupnost na českém trhu a jeho zajímavé možnosti v oblasti vstupů a výstupů.

2.1.1 Charakteristika

Cirrus Logic je EP9312 založen na ARM920T architektuře a je to tzv. *system-on-a-chip design* (celý systém na jednom chipu) s velkým množstvím cílových zařízení a použitelný pro aplikace jako:

- přenosné počítače
- internetová rádio
- osobní digitální asistenti (PDA)
- průmyslové počítače
- průmyslová ruční zařízení (různé uživatelské terminály)
- pokladní systémové terminály
- testovací a měřící zařízení

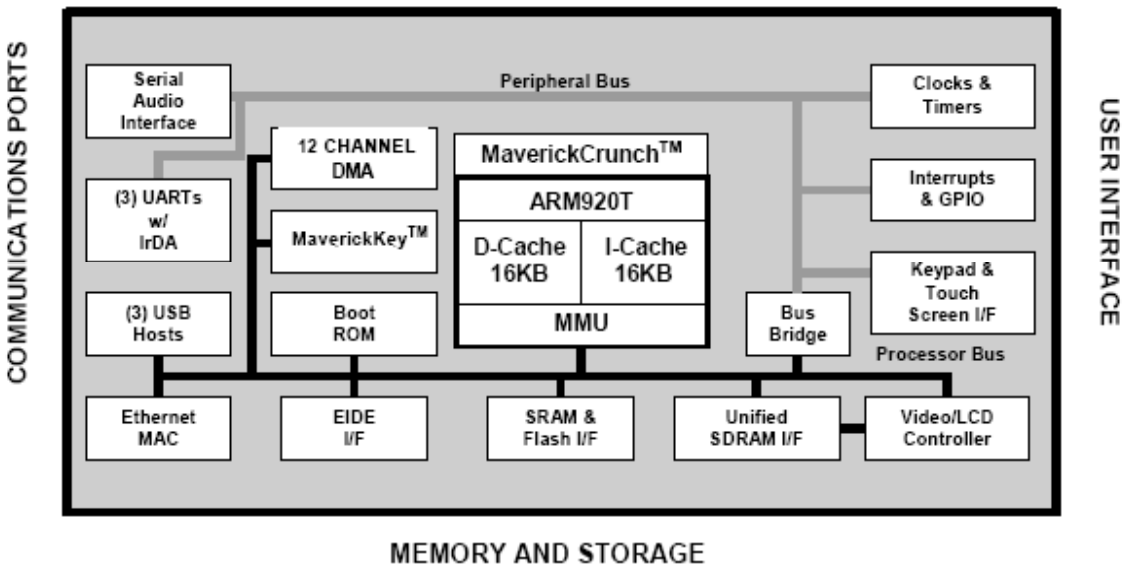
EP9312 patří do první série procesorů založených na této ARM920T architektuře. Jádro ARM920T mikroprocesoru má oddělenou 2x 16Kbyte, 64-cestnou situovaná-asociativní instrukční a datová cache. Je rozšířen MaverickCrunch koprocesorem umožňujícím real - time kompresi.

MaverickKey je hardwareově programovatelné ID (identifikátor) a je řešením pro rostoucí požadavky na bezpečný webový obsah, obchod (digitálních média jako knihy nebo hudba...). Protože softwarové metody rychle zastarávají, MaverickKey umožňuje využívat specifický hardwarový identifikátor jako například přiřazení identifikátoru pro SDMI (Secure Digital Music Initiative) nebo jiný autentifikační mechanismus.

Dále obsahuje výkonný 1/10/100 Mbps ethernetový kontroler (EMAC) spolu s externími rozhraními pro SPI (serial peripheral interface), sběrnici I2S (Integrated Information System, obdoba I2C sběrnice) použitelnou pro až 6-ti kanálový zvuk, rastr/LCD (přímá podpora pro dotykové displaye), IDE storage zařízení (jako jsou pevné disky, CompactFlash karty a další), klávesnici, tři porty USB 2.0 full speed host (OHCI) a také tři UARTs (sériové porty), hodiny reálného času se softwareovým nastavením a kompenzací.

EP9312 je low-power RISC jednočipový počítač s maximem operačního taktovací kmitočtu 200MHz (184 MHz pro průmyslové podmínky). ARM jádro funguje při napětí 1.8V, zatímco I/O operuje při 3.3V s příkonem mezi 100mW až 750mW (v závislosti na rychlosti).

2.1.2 Funkční blokový diagram



Obr.1: Funkční blokový diagram

2.1.3 Výhody

- vysoká odolnost vůči elektromagnetickému rušení, daná malými rozměry zařízení
- zabudovaná podpora pro ethernet rozhraní
- přímá podpora EIDE zařízení
- hodiny reálného času použitelné pro časové plánování řízeného procesu
- velmi nízká spotřeba a s ní spojené nízké tepelné emise

2.2 Renesas SH7727

Dalším řešení počítalo v rozšíření funkcionality stávajícího řešení s SH7727 [1] a rozšíření funkcionality o dnes hojně průmyslově využívané rozhraní CAN [4]. Rozhraní CAN sběrnice bylo zvoleno pro zaručenou bezpečnost přenosu dat a kvůli odolnosti vůči rušení.

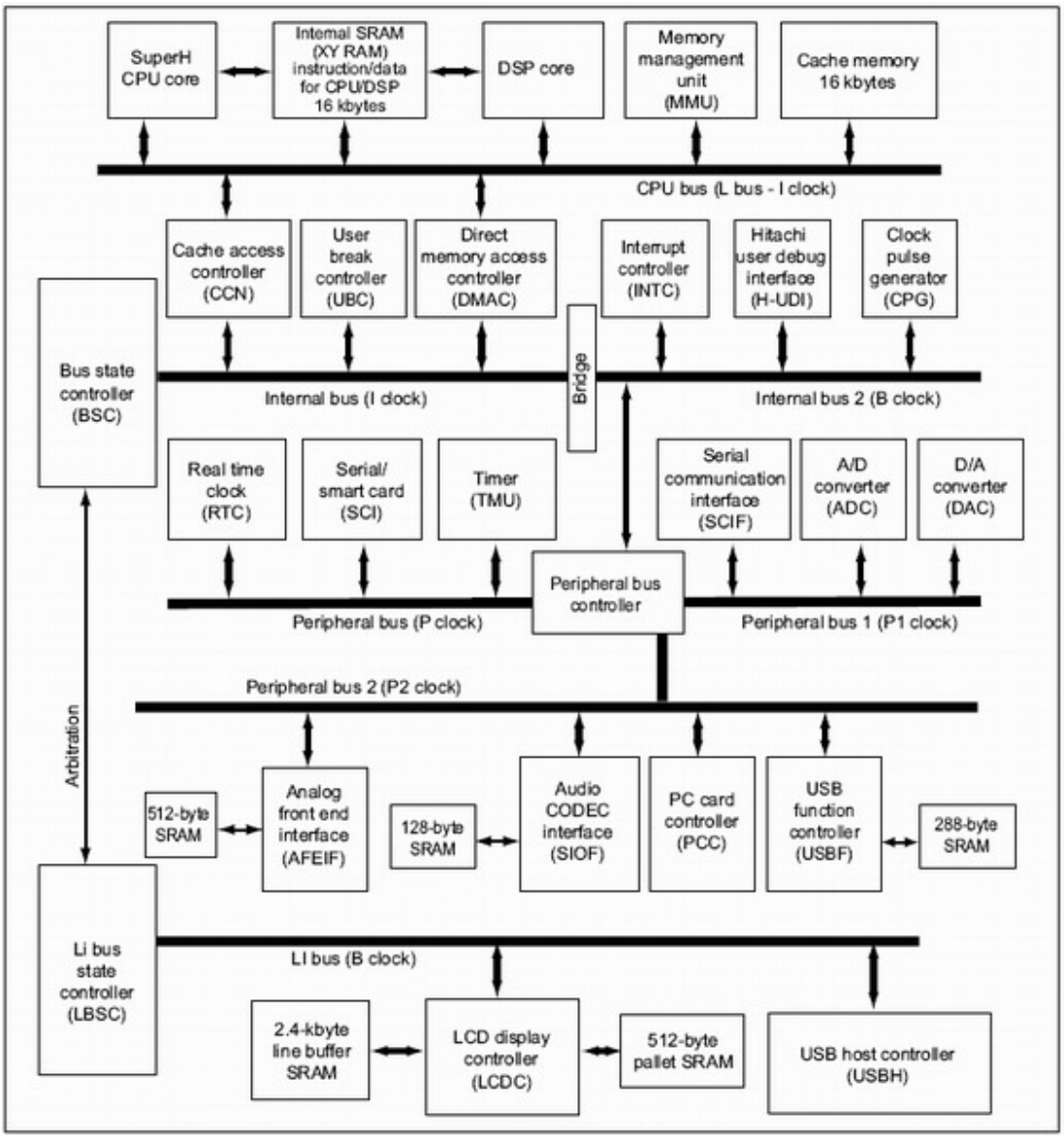
2.2.2 Charakteristika

SH7727 je procesor založený na Hitachi SuperH^{TM} RISC engine třetí generace – SH3. Jeho architektura spojuje mnoho různých periferií s mnoha funkcemi , jako jsou:

- vyrovnávací paměť (cache memory)
- memory management unit (MMU)
- řadič přerušení (interrupt controller)
- tři sériové komunikační porty (SCI,SCIF, SIOF)
- hodiny reálného času (real-time clock - RTC)
- možnost uživatelského přerušení (user break controller - UBC)
- signálový processor (DSP)
- analogově-digitální a digitálně-analogové převodníky (ADC, DAC)
- podporu pro USB porty

Operační frekvence může být až 160MHz což ve spojení s RISC technologií dává dostatečný výpočetní výkon pro širokou škálu aplikací. Maximální příkon při pracovní frekvenci 160MHz je 650mA (typ.330mA) což při napájení 3,3V činí max. 2,15W (typ.1,1W). Na pracovní frekvenci 100MHz je to jen max. 1,45W (typ.0,83W).

2.2.2 Funkční blokový diagram



Obr.2: Funkční blokový diagram

2.2.3 Výhody

- vysoká odolnost vůči elektromagnetickému rušení, daná malými rozměry zařízení
- zabudované A/D a D/A převodníky
- user debug interface
- RTC hodiny reálného času
- 3 seriové linky
- **podpora komunity SH-Linux** [5]

2.3 Zhodnocení výběru

Dalším uvažovaným řešením, byl také procesor SH7760 s Hitachi SuperH™ RISC engine čtvrté generace –SH4. Který byl i původně zvolen jako řešení pro danou úlohu, mimo jiné i díky implementaci dvou CAN sběrnic.

Pro tento procesor bylo vypracováno schéma zapojení a následný částečný návrh desky plošných spojů, ale nastal problém s možností dostupnosti výroby této desky a montáže tohoto procesoru. Jeho pouzdro bylo totiž v provedení BGA a s tím byla i spojena potřeba strojové montáže na desku.

Následoval tedy návrat k výběru mezi Cirrus Logic EP9312 a Renesas SH7727. Z počátku se zdál výhodnější EP9312, ale pak došlo na otázku operačního systému tohoto procesoru. Port Linuxu pro platformu ARM procesoru prozatím počítal pouze s předchůdcem (EP7312) a nebylo jisté kdy bude zahrnuta podpora i pro EP9312.

Naopak Renesas SH7727 měl podporu operačního systému SH-Linux [5] a výchozí kontrolér založený na tomto procesoru již běžel, bylo tedy rozhodnuto o rozšíření daného kontroléru o sběrnici CAN [4].

3. Komunikační rozhraní pro řízení

V této kapitole je stručný popis vybraného průmyslového komunikačního rozhraní pro řešení dané úlohy.

3.1 CAN - Controller Area Network

Průmyslový komunikační protokol firmy Bosh. Následující odstavec popisuje jeho původní nasazení a jeho postupné rozšiřování v aplikacích v průmyslu.

3.1.1 Úvod

Controller Area Network (CAN) [4] je sériový komunikační protokol, vyvinutý v 80. letech dvacátého století firmou Bosch pro nasazení v automobilech. Když přední výrobci integrovaných obvodů implementovali podporu protokolu CAN do svých produktů, došlo k využití tohoto protokolu i v různých průmyslových aplikacích. Důvodem je především nízká cena, snadné nasazení, spolehlivost, vysoká přenosová rychlost, snadná rozšiřitelnost a dostupnost potřebné součástkové základny.

V současné době má protokol CAN své pevné místo mezi ostatními fieldbusy a je definován normou ISO 11898. Ta popisuje fyzickou vrstvu protokolu a specifikaci CAN 2.0A. Později byla ještě vytvořena specifikace CAN 2.0B, která zavádí tzv. standardní a rozšířený formát zprávy (lišící se v délce identifikátoru zprávy). Tyto dokumenty definují pouze fyzickou a linkovou vrstvu protokolu podle referenčního modelu ISO/OSI.

3.1.2 Vlastnosti

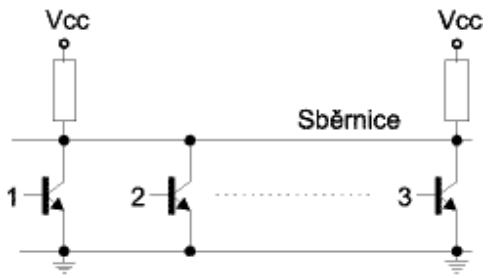
CAN byl navržen tak, aby umožnil provádět distribuované řízení systémů v reálném čase s přenosovou rychlostí do 1Mbit/s a vysokým stupněm zabezpečení přenosu proti chybám. Jedná se o protokol typu *multi-master*, kde každý uzel sběrnice může být *master* a řídit tak chování jiných uzlů. Není tedy nutné řídit celou síť z jednoho "nadřazeného" uzlu, což přináší zjednodušení řízení a zvyšuje spolehlivost (při poruše jednoho uzlu může zbytek sítě pracovat dál). Pro řízení přístupu k médiu je použita sběrnice s náhodným přístupem, která řeší kolize na základě prioritního rozhodování. Po sběrnici probíhá komunikace mezi dvěma uzly pomocí zpráv (datová zpráva a žádost o data), a management sítě (signalizace chyb, pozastavení komunikace) je zajištěn pomocí dvou speciálních zpráv (chybové zprávy a zprávy o přetížení).

Zprávy vysílané po sběrnici protokolem CAN neobsahují žádnou informaci o cílovém uzlu, kterému jsou určeny, a jsou přijímány všemi ostatními uzly připojenými ke sběrnici. Každá zpráva je uvozena identifikátorem, který udává význam přenášené zprávy a její prioritu, nejvyšší prioritu má zpráva s identifikátorem 0. Protokol CAN

zajišťuje, aby zpráva s vyšší prioritou byla v případě kolize dvou zpráv doručena přednostně a dále je možné na základě identifikátoru zajistit, aby uzel přijímal pouze ty zprávy, které se ho týkají (*Acceptance Filtering*).

3.2 Fyzické přenosové médium

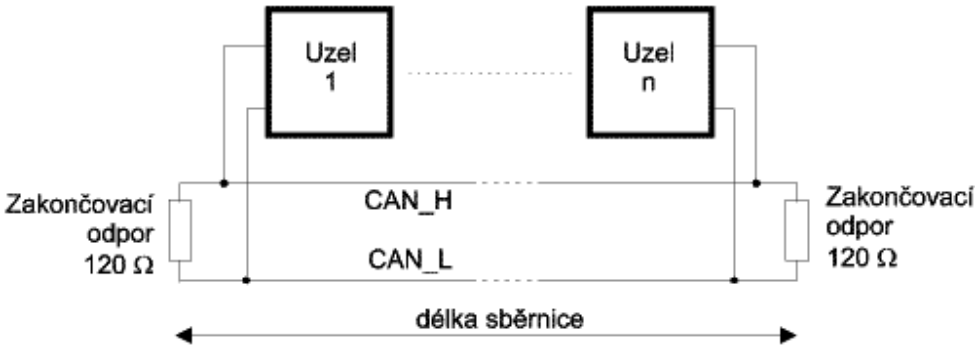
Základním požadavkem na fyzické přenosové médium protokolu CAN je, aby realizovalo funkci logického součinu. Standard protokolu CAN definuje dvě vzájemně komplementární hodnoty bitů na sběrnici - *dominant* a *recessive*. Jedná se v podstatě o jakýsi zobecnělý ekvivalent logických úrovní, jejichž hodnoty nejsou určeny a skutečná reprezentace záleží na konkrétní realizaci fyzické vrstvy. Pravidla pro stav na sběrnici jsou jednoduchá a jednoznačná. Vysílají-li všechny uzly sběrnice *recessive* bit, pak na sběrnici je úroveň *recessive*. Vysílá-li alespoň jeden uzel *dominant* bit, je na sběrnici úroveň *dominant*. Příkladem může být optické vlákno, kde stavu *dominant* bude odpovídat stav "svítí" a *recessive* stav "nesvítí". Dalším příkladem může být sběrnice buzená hradly s otevřeným kolektorem (obr. 3), kde stavu *dominant* bude odpovídat logická nula na sběrnici a stavu *recessive* logická jednička. Pak, je-li jeden tranzistor sepnut, je na sběrnici úroveň logické nuly (*dominant*) a nezáleží již na tom, zda je či není sepnutý i nějaký jiný tranzistor. Pokud není sepnut žádný tranzistor, je na sběrnici úroveň logické jedničky (*recessive*).



Obr. 3: Příklad realizace fyzické vrstvy protokolu CAN

Pro realizaci fyzického přenosového média se nejčastěji používá diferenciální sběrnice definovaná podle normy ISO 11898. Tato norma definuje jednak elektrické vlastnosti vysílacího budiče a přijímače a zároveň principy časování, synchronizace a kódování jednotlivých bitů. Sběrnici tvoří dva vodiče (označované CAN_H a CAN_L), kde *dominant* či *recessive* úroveň na sběrnici je definována rozdílovým napětím těchto dvou vodičů. Dle nominálních úrovní uvedených v normě je pro úroveň *recessive* velikost rozdílového napětí V_{diff} = 0V a pro úroveň *dominant* V_{diff} =2V. Pro eliminaci odrazů na vedení je sběrnice na obou koncích přizpůsobena zakončovacími odpory o

velikosti 120 Ohmů. Jednotlivá zařízení jsou na sběrnici připojena pomocí konektorů, nejčastěji jsou používány konektory D-SUB.



Obr. 4: Principiální struktura sítě CAN podle ISO 11898

Přesné elektrické charakteristiky sběrnice (obr. 4) jsou popsány v normě ISO 11898.

Čistě teoreticky může být ke sběrnici připojeno neomezené množství uzlů, ale s ohledem na zatížení sběrnice a zajištění statických a dynamických parametrů norma omezuje maximum na 30 uzlů. Maximální délka sběrnice je pro rychlost 1Mbit/s udávaná 40m. Pro nižší rychlosti není délka sběrnice specifikována. Při rychlosti 10kbit/s lze provozovat sběrnici i delší než 5km, v závislosti na kvalitě použitého kabelu.

3.3 Typy zpráv

Specifikace protokolu CAN definuje čtyři typy zpráv.

První dvě se týkají datové komunikace po sběrnici. Je to jednak datová zpráva, která představuje základní prvek komunikace uzlů po sběrnici, a dále pak zpráva na vyžádání dat, kdy uzel žádá ostatní účastníky na sběrnici o zaslání požadovaných dat. Datová zpráva umožňuje vyslat na sběrnici 0 až 8 datových bajtů. Pro jednoduché příkazy uzlům (např. příkazy typu vypni/zapni) není nutné přenášet žádné datové bajty (význam příkazu je dán identifikátorem zprávy), což zkracuje dobu potřebnou k přenosu zprávy a zároveň zvětšuje propustnost sběrnice, zvláště pak při silném zatížení. Zpráva na vyžádání dat je vyslána uzlem, který požaduje zaslání určitých dat. Odpovědí na tento požadavek je odeslání požadovaných dat uzlem, který tato data má k dispozici.

Příklad datové zprávy

Obr. 5: Datová zpráva podle specifikace CAN 2.0A

Význam jednotlivých částí datové zprávy (obr. 5) podle specifikace 2.0A je následující:

Začátek zprávy (SOF = Start Of Frame)- začátek zprávy, 1 bit dominant.
Řízení přístupu na sběrnici (Arbitration Field) - určení priority zprávy

- Identifikátor zprávy - 11 bitů, udává význam přenášené zprávy
- RTR bit (Remote Request) - 1 bit, příznak udává, zda se jedná o datovou zprávu nebo o žádost o vyslání dat. V datové zprávě musí být tento bit dominant, v žádosti o data recessive.

Řídící informace (Control Field)

- R0, R1 - rezervované bity
- Délka dat - 4 bity, počet přenášených datových bajtů ve zprávě. Povolené hodnoty jsou 0 až 8.

Datová oblast (Data Field) - datové bajty zprávy. Maximálně 8 bajtů je vysláno od MSB
CRC (CRC Field) - 16 bitů, zabezpečovací CRC kód

- CRC kód - 15 bitů
- ERC (CRC oddělovač) - 1 bit recessive

Potvrzení (ACK Field) - 2 bity
ACK (bit potvrzení) - 1 bit
ACD (oddělovač potvrzení) - 1 bit recessive
Konec zprávy (End Of Frame) - 7 bitů recessive
Mezera mezi zprávami (Interframe Space) - 3 bity recessive, odděluje dvě zprávy.

Podrobný popis zpráv a doplňující informace lze nalézt v [4].

4. Návrh hardware

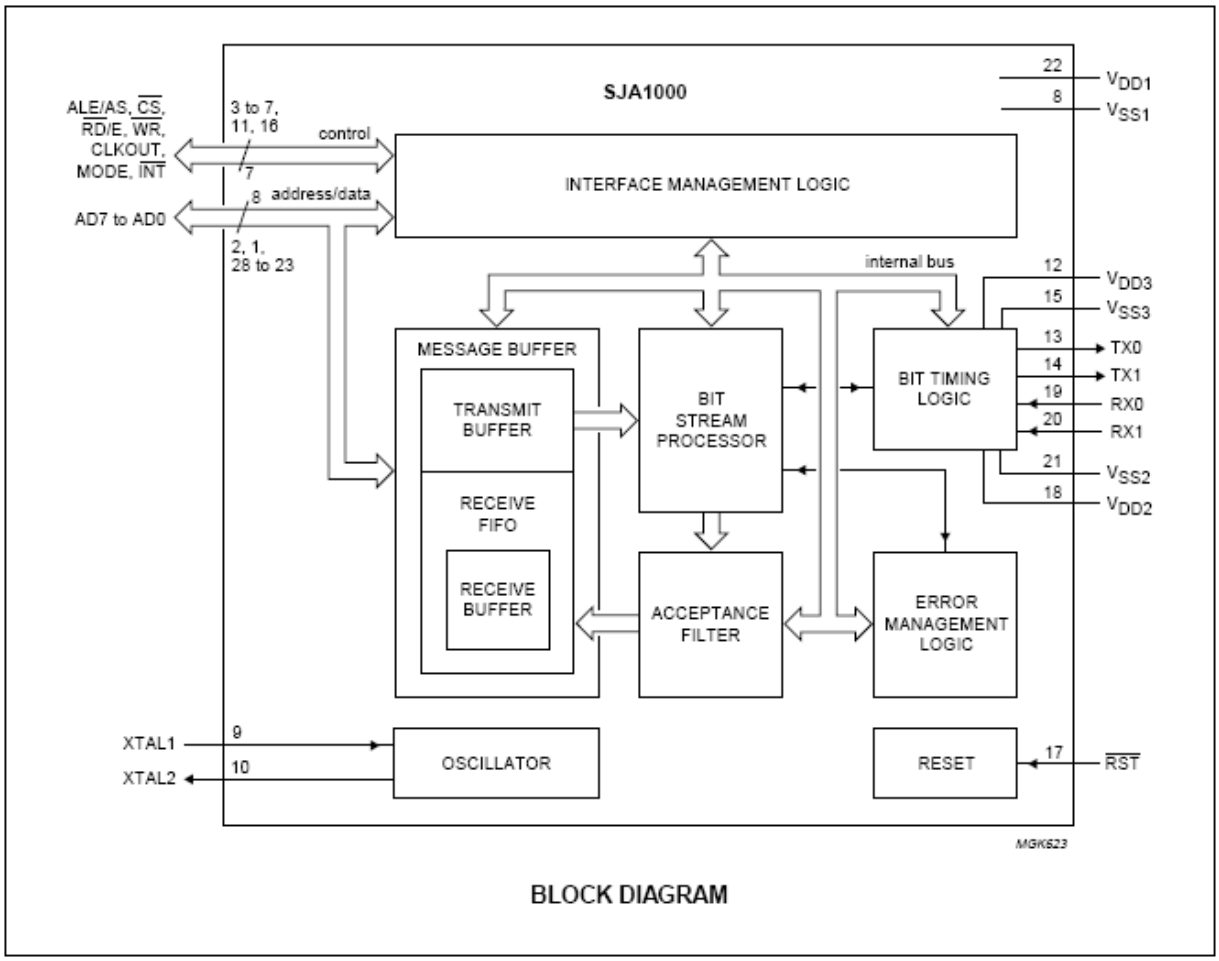
Tato kapitola popisuje hardwareové řešení rozšíření SH7727 o řadič sběrnice CAN a popis použitého CAN řadiče.

Při hledání způsobu implementace CAN [4] rozhraní do stávajícího kontroléru s SH7727 jsme se rozhodli pro CAN kontrolér firmy PHILIPS – SJA1000 [2] vzhledem k jeho nízké ceně, dostupnosti a také podpoře driveru LinCAN [8].

4.1 SJA1000

Jedná se o autonomní CAN kontroler schopný zajistit komunikaci CAN protokolem specifikace CAN2.0A i CAN2.0B (viz. kapitola 3.). Dále následuje popis tohoto kontroléru a jeho základní charakteristika.

4.1.1 Funkční blokový diagram a popis signálů



Obr. 6: Funkční blokový diagram sja1000

Na obr. č. 6 je zjednodušené blokové schéma obvodu SJA1000:

- Interface Management Logic – zajištuje komunikaci mezi bloky a řídí interní sběrnici
- Message Buffer – paměť pro uložení zpráv od/pro nadřazené zařízení (SH7727)
- BIT Stream Processor (BSP) – třídí data mezi vysílacím Message Bufferem a vstupem/výstupem na CAN sběrnici
- BIT Timing Logic – zajišťuje připojení CAN rozhraní, buď přímo nebo jako v našem řešení přez budiče CAN sběrnice 82C250.
- Error Management Logic – umožňuje detekci chybových stavů v komunikaci
- Acceptance Filter – filtr pro příchozí zprávy
- Oscillator – umožňuje připojení krystalu do 24MHz, jiná řešení mohou využívat i externí oscilátor například od mikroprocesoru.

Popis základních signálů:

- ALE – signál pro zápis platné adresy A0 až A7
- /CS – chip select input (zajišťuje výběr daného kontroléru)
- /RD – read enable
- /WR – write enable
- /INT – výstup generující přerušení pro mikrokontroler
- CLKOUT – výstup vnitřního oscilátoru
- MODE – přepíná mezi Intel/Motorola mode
- /RST – reset
- AD7 až AD0 – multiplexovaná sběrnice adresy-data

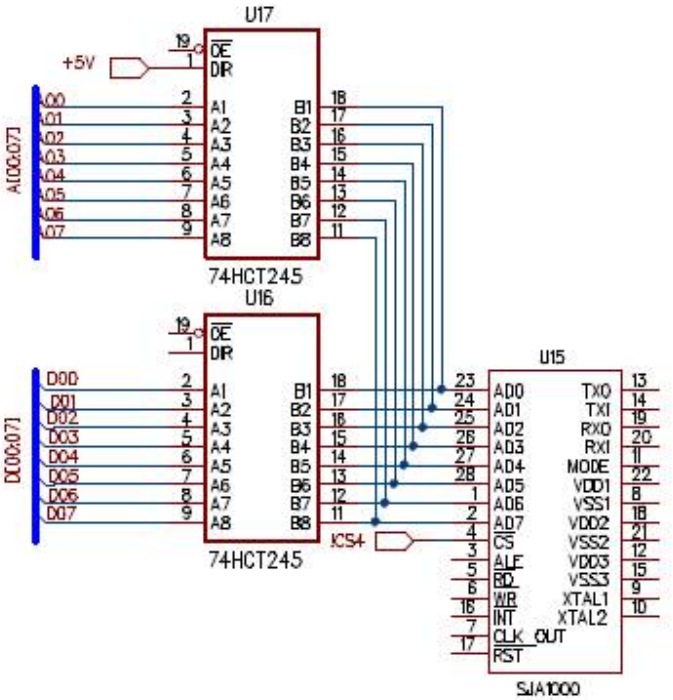
4.1.2 Charakteristika

- pinová a elektrická kompatibilita s PCA82C200 (autonomní CAN kontrolér)
- PCA82C200 mód (BasicCAN mód)
- rozšířený buffer (64 byte FIFO)
- CAN 2.0B protokolová kompatibilita (podporuje 11 i 29- bitový identifikátor)
- přenosové rychlosti až 1 Mbits/s
- PeliCAN režimová rozšíření
- 24 MHz frekvence hodin
- operační rozsah teplot -40 °C až 125 °C

4.2 Obvodové řešení

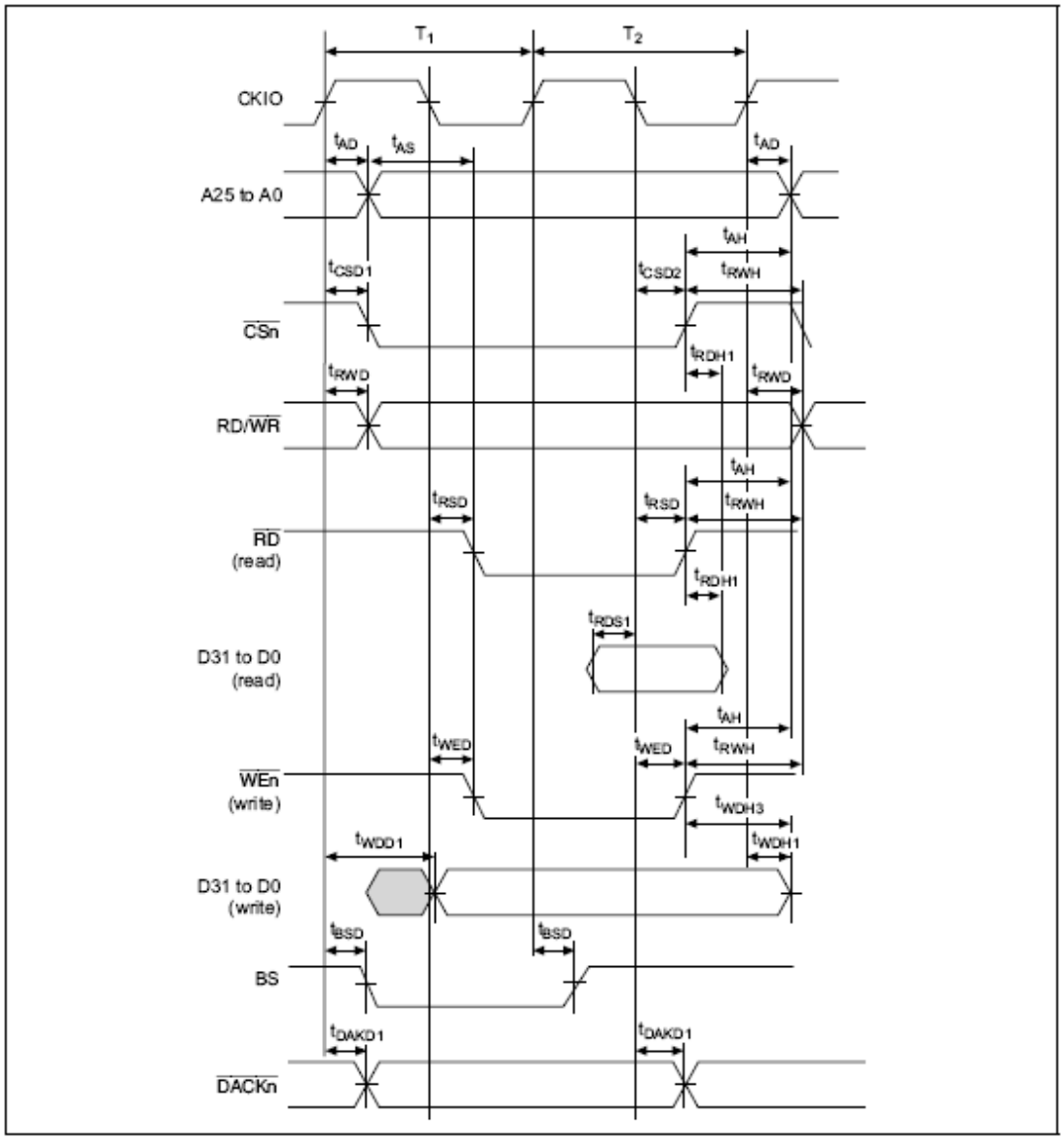
SJA 1000 používá časově multiplexované (kombinované) sběrnice pro adresy a data a tak muselo být vyřešeno pripojení k SH7727 [1], který používá oddělenou adresní a datovou sběrnici. Při návrhu zapojení velmi pomohla inspirace zapojením ISA CAN karty „PIKRONISA“ [7]

Základní řešení spočívalo ve vytvoření sběrnice vhodné pro SJA1000. Jeho realizace byla provedena pomocí dvou 8-mi bitových obousměrných oddělovačů sběrnice (74HCT245). Řešení ukazuje obr.7.

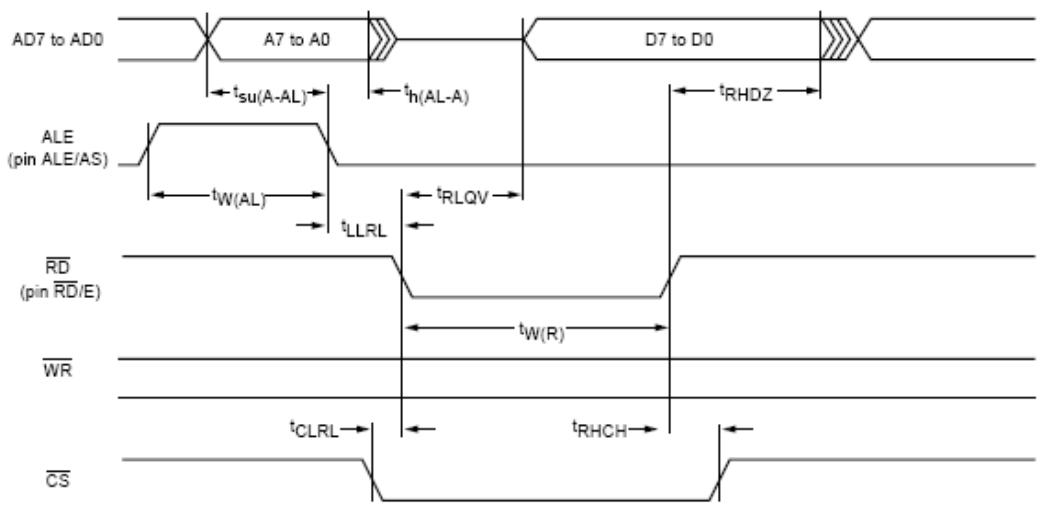


Obr. 7: Vytvoření multiplexované sběrnice

Řídící signály pro směr dat, přepínání adresy-data a řídící signály pro SJA1000 byly odvozeny od signálů procesoru porovnáním časových průběhů komunikace procesoru SH7727 [1] (obr. 8) a SJA1000 [2] (obr. 9 a 10). SJA1000 umožňuje dva druhy komunikace s nadřazeným kontrolérem – Intel nebo Motorola mode. Tyto režimy se volí vstupem MODE (Intel – log. 1 / Motorola – log.0). V našem případě používáme módu Intel a proto jsou uvedeny průběhy pouze pro tento režim. Průběhy pro Motorola mód lze nalézt v [2].

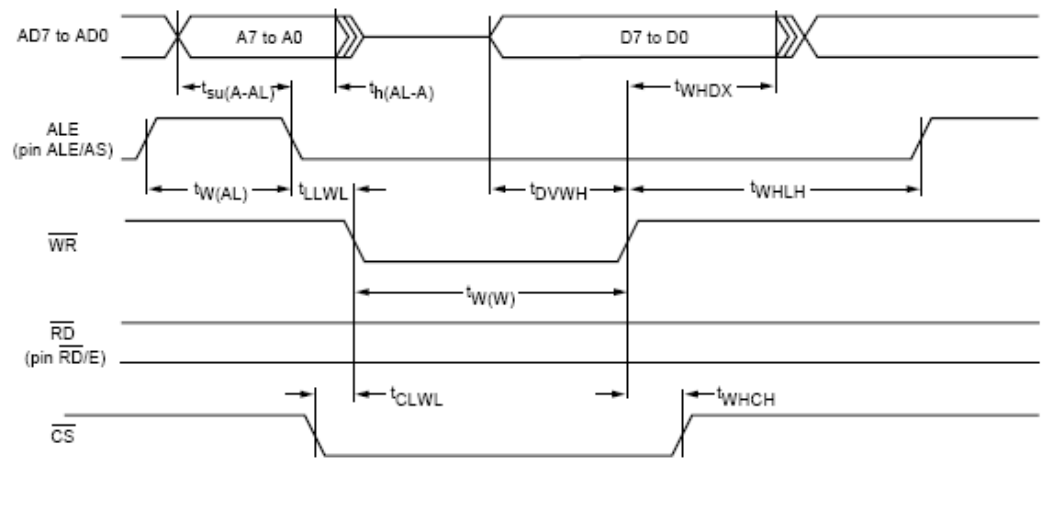


Obr. 8: Průběhy signálů SH7727



Read cycle timing diagram; Intel mode.

Obr. 9: Průběhy signálů SJA1000 pro čtení dat



Write cycle timing diagram; Intel mode.

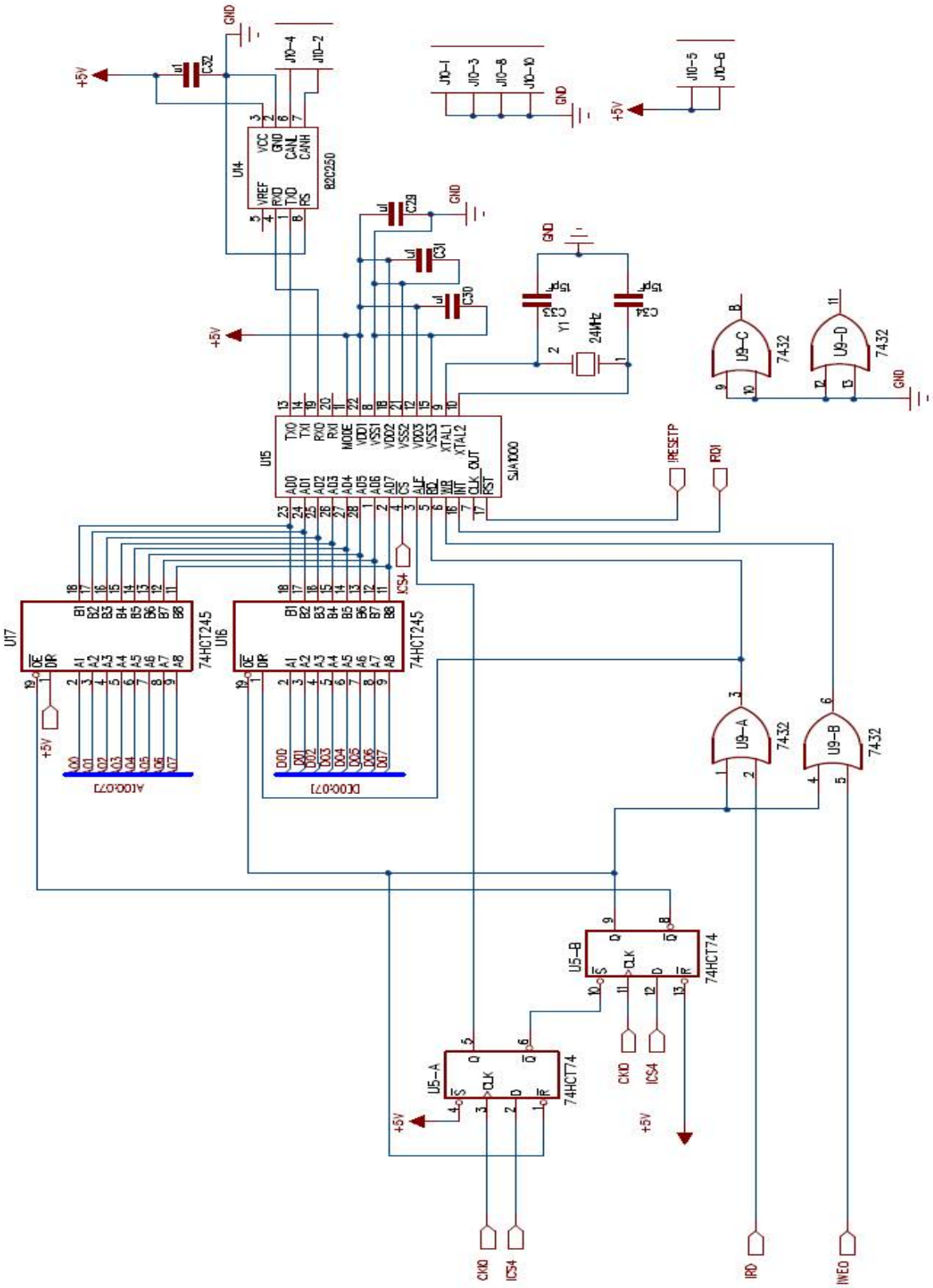
Obr. 10: Průběhy signálů SJA1000 pro zápis dat

Jak ukazuje výše umístěný obr. 8, SH7727 nedisponuje signálem ALE, který při sestupné hraně zapisuje do CAN kontroléru platnou adresu (viz obr 9 a 10).

Pro správnou komunikaci musí být zajištěno, aby byl signál ALE (obr. 9,10) v log.1. Sestupnou hranou signálu /CS dojde k výběru daného chipu (SJA1000) a v tomto okamžiku je SJA1000 připraven přijmout adresu, která je po ustálení zapsána sestupnou hranou ALE. Po časové prodlevě může být generován signál /RD (obr. 9) pro čtení nebo signál /WR (obr. 10) pro zápis dat. Čtení resp. zápis ustálených dat je proveden náběžnou hranou signálu /RD resp. /WR. S touto náběžnou hranou (nebo s prodlevou) můžeme uvést signál /CS a ALE do log.1, aby byly připraveny pro další cyklus.

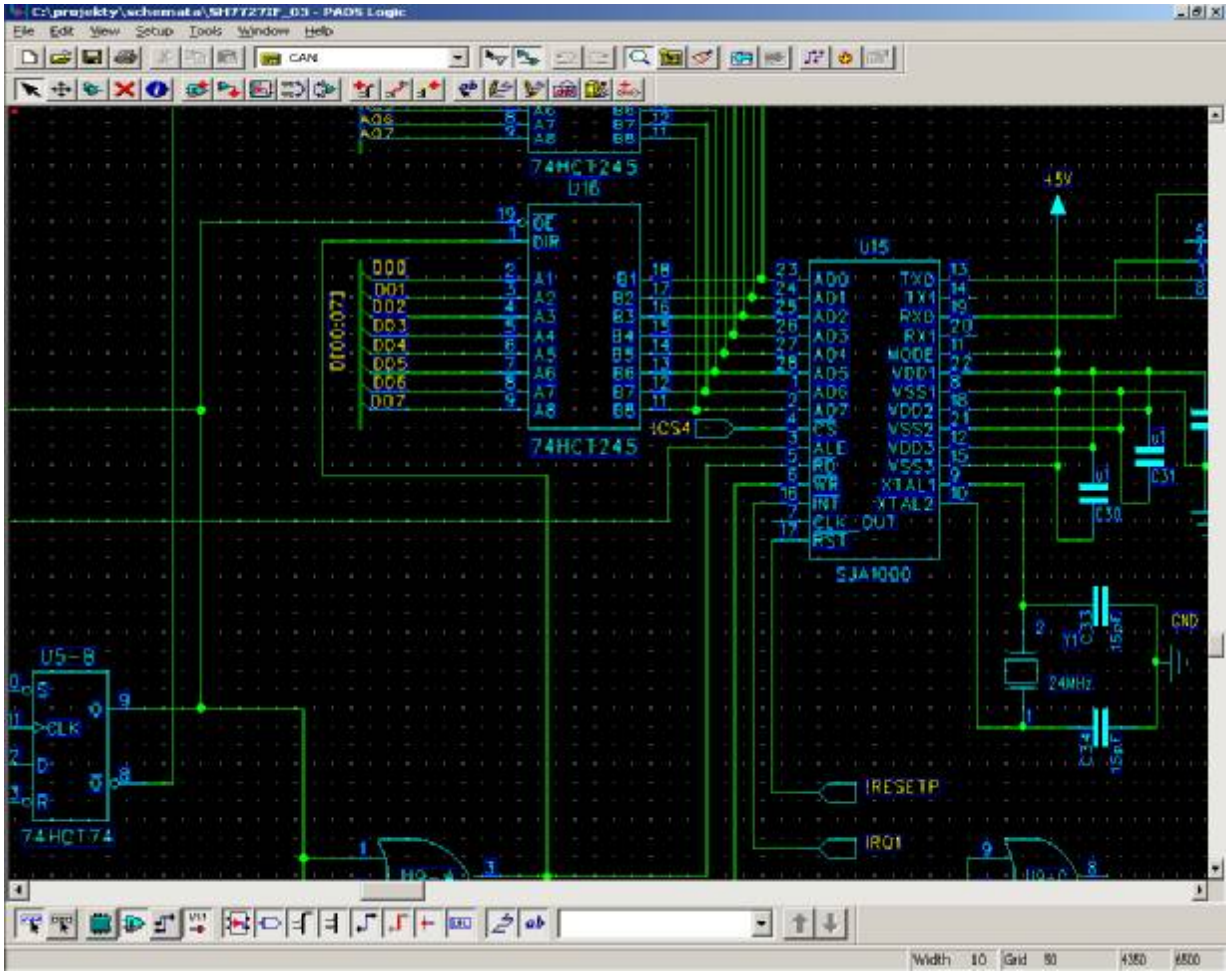
Ke generování signálu ALE byla použity signály CKIO (hodiny procesoru SH7727)(obr. 8) a signál /CS4 (/CSn chip select)(obr. 8). Od signálu CKIO se generuje pomocí klopných obvodů typu D (obr. 11 U5:74HCT74) tím, že je od sestupné hrany signálu /CS vytvořen pomocí prvního D-klopný obvodu pulz ALE. Sestupnou hranu ALE obstarává druhý D-klopný obvod po uplynutí jedné periody CKIO od začátku pulzu ALE. Signály /RD a /WR jsou odvozeny od signálů procesoru SH7727 /RD a /WE0. Hradla typu OR (obr. 11 U9:74HCT32) zajišťují, aby se signály /RD a /WR dostaly k SJA1000 až po skončení sestupné hrany signálu ALE. Druhý D-klopný obvod zároveň zajišťuje výběr adresy / data (obr. 11 U17 / U16) z adresové A[00:07] a datové D[00:07] sběrnice. Směr přenosu dat je řízen pomocí signálu /RD a je zajistěno, že nemůže dojít k zároveň ke generování signálu /RD i /WR.

Konečné řešení ukazuje obr.11: Zapojení realizace CAN na další stránce.



Obr. 11: Zapojení realizace CAN

Schéma zapojení bylo kresleno v návrhovém software Power Logic 5.0.1 firmy Mentor Graphics [11] (obr. 12).

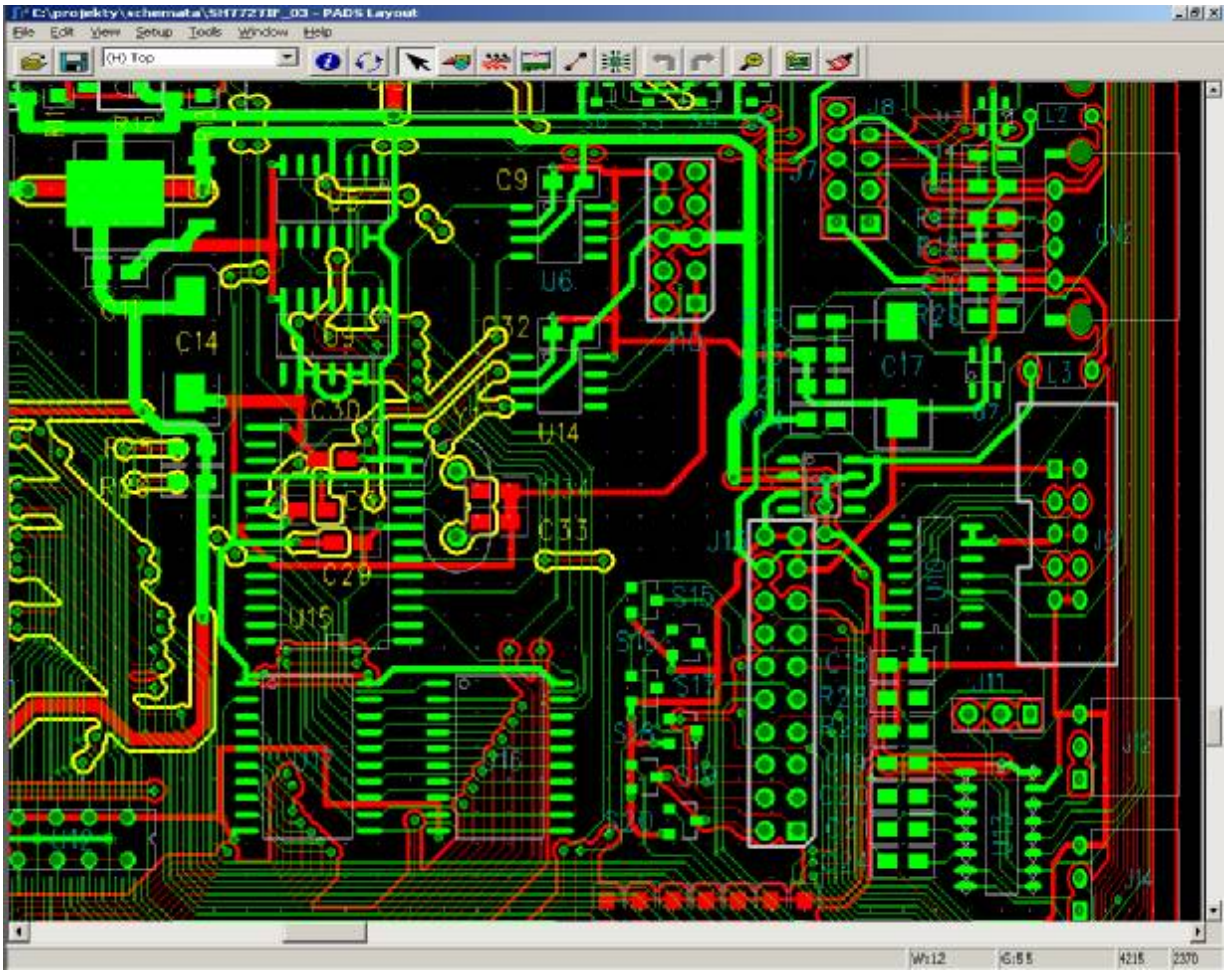


Obr. 12: Power Logic

4.3 Návrh plošného spoje

Při návrhu plošného spoje byl kladen důraz na elektromagnetickou kompatibilitu, aby byla zemnící plocha co nejvíce spojitá a napájecí spoje dostatečně široké.

Spoje byly navrhovány pomocí software Power PCB 5.0.1 firmy Mentor Graphics [11] (obr. 13). Vzhledem k tomu, že se jednalo o dvouvrstvou desku nešlo efektivně použít autorouteru.



Obr. 13: Power PCB

4.4 Osazení DPS

Osazení proběhlo podle osazovacích plánů (viz. přílohy č.3 a č.4). Mimo uvedené schéma deska obsahuje zdroj, CompactFlash, seriové porty a USB porty (ale to není náplní této diplomové práce).

5. Software a oživení

Kapitola obsahuje popis přípravy softwareového prostředí potřebného pro kompilaci software pro platformu procesoru Renesas SH7727, popis kompilace debuggeru pro ladění programů a portování ovladače LinCAN [8] na SH7727. Ve druhé části kapitoly je popis oživení kontroléru a potvrzení funkce driveru snímkem z osciloskopu.

5.1 Příprava software

Pro přípravu software potřebného pro platformu SH3 (SH7727) [1] byl použit operační system Linux nainstalovaný na stolním počítači třídy PC, konkrétně distribuce Debian testing/unstable [12] kvůli dobré softwareové základně.

5.1.1 Instalace balíčků pro SH3 crosskompilaci

Aby mohly byt kompilovány programy pro platformu SH3 bylo nutno doinstalovat balíčky nutné pro crosskompilaci (kompilace programů pro jednu platformu na druhé platformě).

Nejprve bylo nutno zajistit, aby byly v systému nainstalovány knihovny:

- *tk8.3*
- *tk8.3-dev*
- *tcl8.3*
- *tcl8.3-dev*

A to buď pomocí správce balíčků synaptic (GUI pro apt) nebo přímo pomocí

 # apt-get update
 # apt-get install tk8.3 tk8.3-dev tcl8.3 tcl8.3-dev

Pak bylo nutno stáhnout potřebné balíčky ze stránky projektu SH-Linux [5] z adresy

http://www.sh-linux.org/rpm-2003/RPMS/noarch/host/ :

- *binutils-sh-linux-2.13.90.0.18-1.i386.rpm*
- *gcc-sh-linux-3.2.3-5.i386.rpm*
- *glibc-sh3-linux-2.2.5-8.noarch.rpm*

- *glibc-sh-linux-2.2.5-8.noarch.rpm*
- *zlib-sh3-linux-1.1.3-5.noarch.rpm*
- *zlib-sh-linux-1.1.3-5.noarch.rpm*

Protože GNU/Linux Debian nepoužívá balíčkovací systém RPM byly balíčky nainstalovány pomocí nástroje *alien*:

 # alien –i binutils-sh-linux-2.13.90.0.18-1.i386.rpm
 # alien –i gcc-sh-linux-3.2.3-5.i386.rpm
 # alien –i glibc-sh-linux-2.2.5-8.noarch.rpm
 # alien –i glibc-sh3-linux-2.2.5-8.noarch.rpm
 # alien –i zlib-sh-linux-1.1.3-5.noarch.rpm
 # alien –i zlib-sh3-linux-1.1.3-5.noarch.rpm

Pro následnou kompilaci LinCANu [8] je dále potřebné stáhnout zdrojový kód linuxového jádra použitého na SH7727. K nalezení je opět na stránkách projektu SH-Linux [5].

5.1.2 Příprava debuggeru Insight

Insight [13] je grafickou nadstavbou k linuxovému debuggeru GDB. Ze stránek ftp://sources.redhat.com/pub/gdb/old-releases/ byl stažen balíček:

- *insight-6.0.tar.gz*

i když existují i novější verze, tak byla zvolena tato, protože vyšší verze se vyznačují mírnou nestabilitou.

Postup kompilace:

Vytvořit adresář (např. insight) a do něj rozbalit balíček *insight-6.0.tar.gz*

-pro platformu PC a cíl (target) PC:

vytvořit v adresáři insight podadresář pc a v něm spustit:

$ ../configure
$ make
$ su
 # make install

vytvoří se spustitelný soubor *insight* , který spouští debugger pro PC platformu.

-pro platformu PC a cíl SH3:

vytvořit v adresáři insight podadresář sh a v něm spustit:

$ ../configure --program-prefix=sh --target=sh
$ make
$ su
 # make install

vytvoří se spustitelný soubor *shinsight* , který spouští debugger pro SH platformu, kterou je možno připojit k PC například pomocí sériového rozhraní RS-232.

5.2 Kompilace LinCAN driveru pro SH3 platformu

Všechny podstatné informace pro získání zdrojového kódu LinCAN driveru jsou dostupné na stránkách Ing. Pavla Píši (http://cmp.felk.cvut.cz/~pisa/#can). Poslední verze zdrojového kódu byla stažena pomocí CVS.

Připravíme si adresář např. *lincan* a v něm spustíme:

$ cvs -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/ocera login
$ cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/ocera co
ocera/components/comm/can

(\ znamená pokračování řádku)

do adresáře *lincan* se stahne datový strom *ocera/components/comm/can*

Dále je potřeba ke kompilaci LinCANu novější verze kompilátoru *make*. Tu lze nalézt zde: http://cmp.felk.cvut.cz/~pisa/can/make-3.81beta1.tar.gz

Stažený a rozbalený balíček byl nainstalován pomocí:

$ ./configure
$ make
$ su
 # make install

Nový *make* se standartně nainstaluje do adresáře */usr/local/bin* a pro použití tohoto nového *make* je potřeba zajistit, aby systém při jeho volání hledal přednostně v tomto adresáři. Druhou méně elegantní možností je přejmenování starého *make* na např. *make_old* (nachází se v */usr/bin*) a přikopírování nového *make.*

5.2.1 Vlastní kompilace

První co bylo nutno udělat byla úprava konfiguračního souboru *config.omk* pro crosskompilaci nacházejícího se v *ocera/components/comm/can*

Config.omk :

```
LINUX_DIR= cesta ke zdrojovému kódu sh-kernelu

CFLAGS+=-ggdb
CXXFLAGS+=-ggdb

CFLAGS+=-I $(LINUX_DIR)/include
CXXFLAGS+=-I $(LINUX_DIR)/include

CC=sh3-linux-gcc
CXX=sh3-linux-g++
AR=sh3-linux-ar
LD=sh3-linux-ld

# uLan driver compiled into userspace library
#CONFIG_OC_UL_DRV_USLIB=y
#CPPFLAGS+=-DFOR_USER_SPACE -DUL_DRV_IN_LIB

CONFIG_OC_CANMOND=n

# Config for can/lincan/src
CONFIG_OC_LINCAN=y
CONFIG_OC_LINCANRTL=n
CONFIG_OC_LINCANVME=n
CONFIG_OC_LINCAN_PORTIO_ONLY=n
CONFIG_OC_LINCAN_MEMIO_ONLY=n
CONFIG_OC_LINCAN_CARD_pip=n
CONFIG_OC_LINCAN_CARD_pccan=n
CONFIG_OC_LINCAN_CARD_smartcan=n
CONFIG_OC_LINCAN_CARD_nsi=n
CONFIG_OC_LINCAN_CARD_cc_can104=n
CONFIG_OC_LINCAN_CARD_ems_cpcpci=n
```

CONFIG_OC_LINCAN_CARD_pc_i03=n
CONFIG_OC_LINCAN_CARD_pcm3680=n
CONFIG_OC_LINCAN_CARD_aim104=n
CONFIG_OC_LINCAN_CARD_m437=n
CONFIG_OC_LINCAN_CARD_pcccan=n
CONFIG_OC_LINCAN_CARD_ssv=n
CONFIG_OC_LINCAN_CARD_bfadcan=n
CONFIG_OC_LINCAN_CARD_pikronisa=y
CONFIG_OC_LINCAN_CARD_eb8245=n
CONFIG_OC_LINCAN_CARD_kv_pcican=n
CONFIG_OC_LINCAN_CARD_msmcan=n
CONFIG_OC_LINCAN_CARD_oscar=n
CONFIG_OC_LINCAN_CARD_adlink7841=n
CONFIG_OC_LINCAN_CARD_unican=n
CONFIG_OC_LINCAN_CARD_virtual=y
CONFIG_OC_LINCAN_CARD_template=y

Pak se samotná kompilace provedla v adresáři *lincan* pomocí níže uvedených příkazů:

$ cd ocera/components/comm/can
$ ./switch2standalone
$ make

Vlastní driver *lincan.o* se vykompiluje do adresáře:

ocera/components/comm/can/_compiled/modules

A utility použitelné k testování LinCANu do adresáře:

ocera/components/comm/can/_compiled/bin-utils

5.3 Oživení

5.3.1 Ověření základní funkcionality

Jako první bylo nutné zkontrolovat a nastavit regulátor napětí 3.3V a 5V. Dalším krokem bylo ověření funkce oscilátoru u CAN kontroléru SJA1000 na osciloskopu.

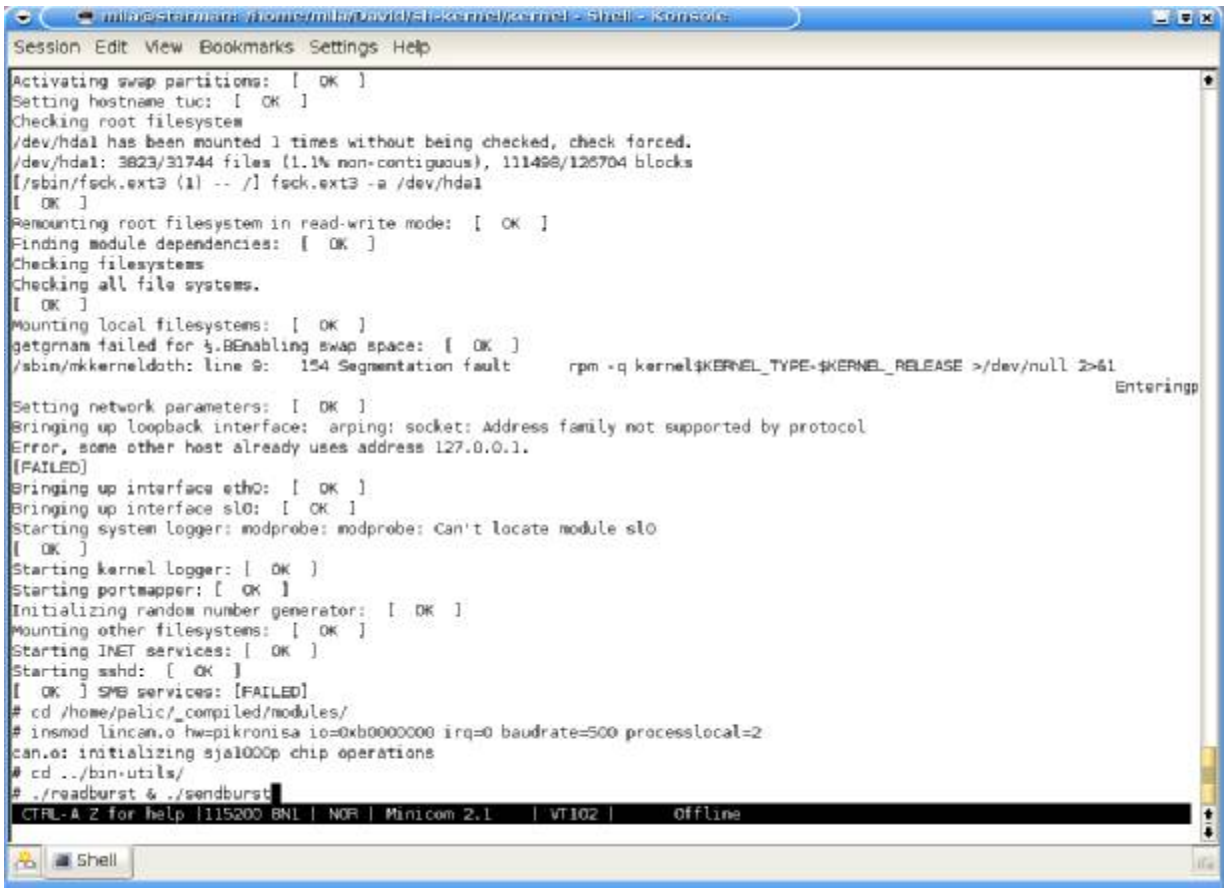
Teprve po tomto ověření bylo možno propojit desku procesoru s vyrobenou deskou vstupně/výstupních portů.

5.3.2 Připojení kontroléru SH7727 k PC

Připojení bylo realizováno pomocí sériové linky RS-232. Použity byly pouze signály RX(receive), TX(transmit) a GND(signálová zem).

Na straně počítače třídy PC byl pro práci v seriové konzoli použit program *minicom.*

Parametry přenosu *115200 Bd, 8N1 (8bit, žádná parita, 1 stop bit)*. Příklad je na obr.14



Obr. 14: Příklad práce v seriové konzoli na připojeném SH7727

5.3.3 Přístup k registrům SJA1000

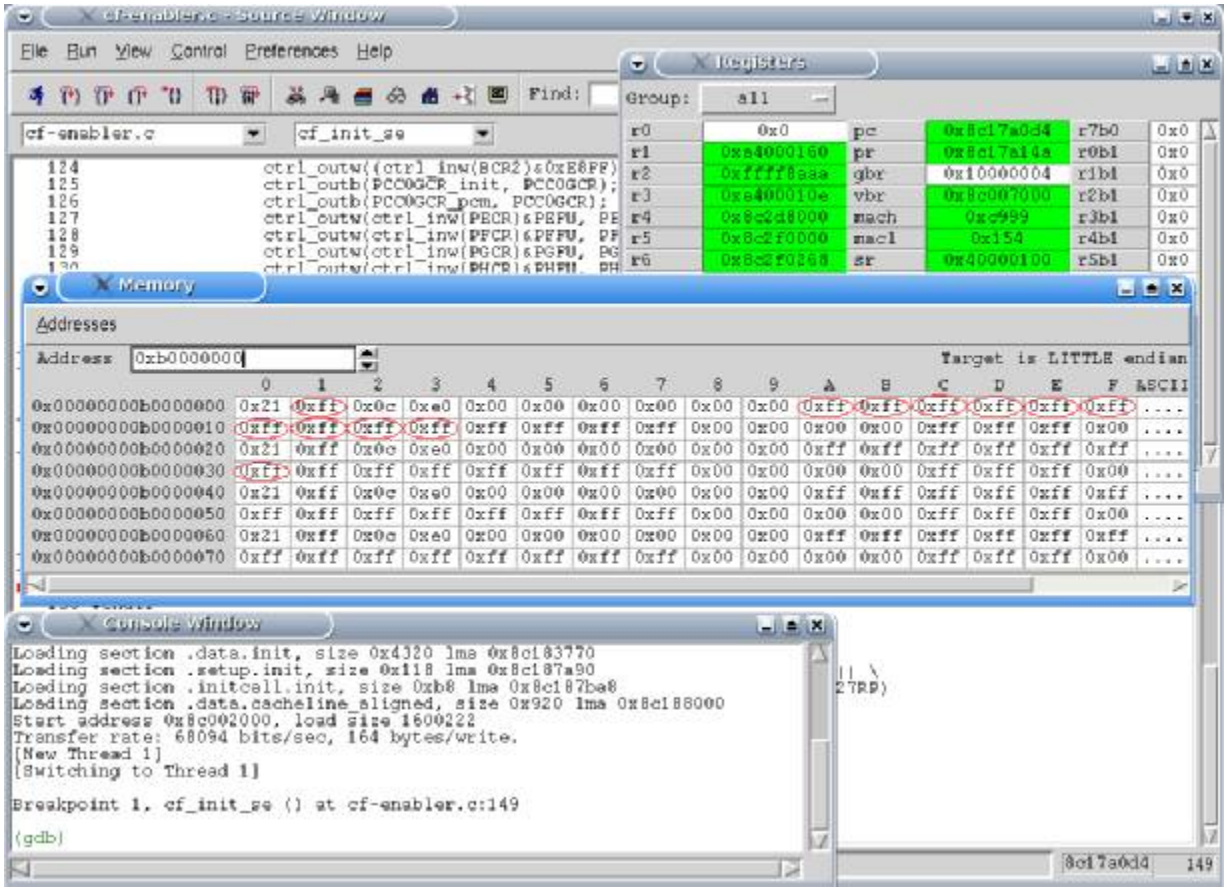
Dalším krokem bylo ověření možnosti přistupovat do registrů CAN kontroléru SJA1000.

Pro přístup k adresně-datovému prostoru kontroléru s SH77727 použijeme debuggeru Insight zkompilovaném v 5.1.2

Podle [1] a použitého zapojení (příloha č.1) výběr pomocí signálu !CS4, !WE0 a použití D0-D7 a A0-A7 odpovídá paměťovému prostoru řídících registrů SJA1000 namapovaný prostor od adresy 0xB0000000.

CAN ADDRESS	SEGMENT	OPERATING MODE		RESET MODE	
		READ	WRITE	READ	WRITE
0	control	control	control	control	control
1		(FFH)	command	(FFH)	command
2		status	–	status	–
3		interrupt	–	interrupt	–
4		(FFH)	–	acceptance code	acceptance code
5		(FFH)	–	acceptance mask	acceptance mask
6		(FFH)	–	bus timing 0	bus timing 0
7		(FFH)	–	bus timing 1	bus timing 1
8		(FFH)	–	output control	output control
9		test	test; note 2	test	test; note 2
10	transmit buffer	identifier (10 to 3)	identifier (10 to 3)	(FFH)	–
11		identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	(FFH)	–
12		data byte 1	data byte 1	(FFH)	–
13		data byte 2	data byte 2	(FFH)	–
14		data byte 3	data byte 3	(FFH)	–
15		data byte 4	data byte 4	(FFH)	–
16		data byte 5	data byte 5	(FFH)	–
17		data byte 6	data byte 6	(FFH)	–
18		data byte 7	data byte 7	(FFH)	–
19		data byte 8	data byte 8	(FFH)	–
20	receive buffer	identifier (10 to 3)	identifier (10 to 3)	identifier (10 to 3)	identifier (10 to 3)
21		identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC
22		data byte 1	data byte 1	data byte 1	data byte 1
23		data byte 2	data byte 2	data byte 2	data byte 2
24		data byte 3	data byte 3	data byte 3	data byte 3
25		data byte 4	data byte 4	data byte 4	data byte 4
26		data byte 5	data byte 5	data byte 5	data byte 5
27		data byte 6	data byte 6	data byte 6	data byte 6
28		data byte 7	data byte 7	data byte 7	data byte 7
29		data byte 8	data byte 8	data byte 8	data byte 8
30		(FFH)	–	(FFH)	–
31		clock divider	clock divider; note 3	clock divider	clock divider

Obr. 15: Registry CAN kontroléru SJA1000 [2]



Obr. 16: Registry CAN kontroléru SJA1000 – čtení pomoci Insight debuggeru

Z předchozích obrázků (15,16) je vidět, že procesor SH7727 [1] s CAN kontrolérem SJA1000 [2] komunikuje a je možno přistoupit k zavedení ovladače zkompilovaného v 5.2.1.

5.4 Zavedení ovladače LinCAN a jeho funkce

Následuje popis přenesení a zavedení driveru LinCAN z 5.2.1 do kontroléru s SH7727 a potvrzení jeho funkčnosti na osciloskopu.

5.4.1 Zavedení ovladače

Souborový systém SH-Linuxu [5] je uložen na paměťové kartě CompactFlash, proto nejjednodušším a nejrychlejším způsobem je nakopírovat soubor na CompactFlash pomocí čtečky paměťových karet.

Po nabootování kontroléru zadáme pomocí minicomu (5.3.2) v adresáři s driverem příkaz:

 # insmod lincan.o hw=pikronisa io=0xb0000000 irq=0 baudrate=500

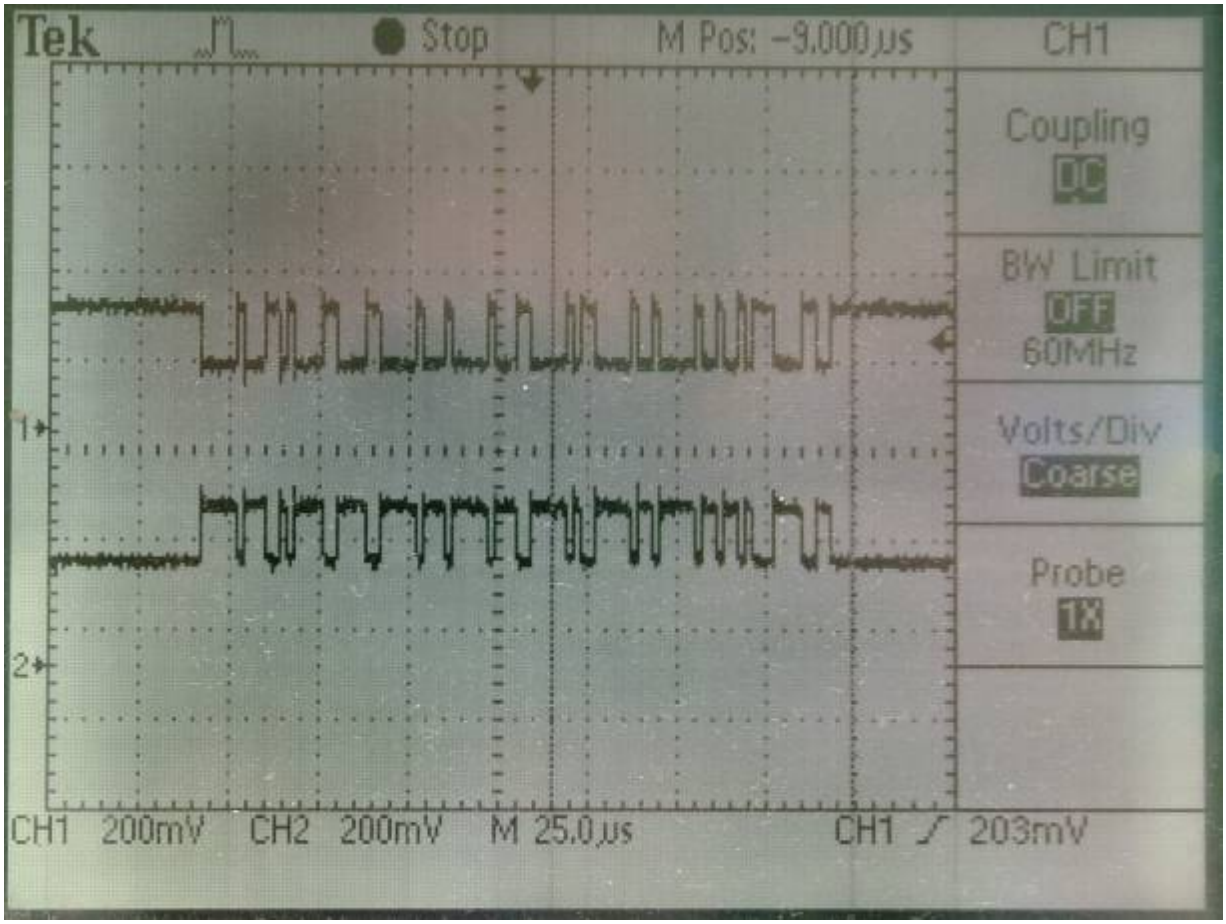
kde insmod je příkaz pro zavedení modulu, hw označuje typ LinCANem podporovaného hardware, io bázovou adresu (viz 5.3.3), irq použité přerušení(0 znamená žádné) a baudrate je rychlost v kBd

při úspěšném zavedení ovladač ohlásí:

can.o initializing sja1000p chip operations

5.4.2 Otestování funčnosti

Nyní je možno pomocí utilit LinCANu
(adresář *ocera/components/comm/can/_compiled/bin-utils*)
posílat a přijímat CAN zprávy. Příklad generované zprávy na výstupu SJA1000 je na obr. 17, který dokládá funkčnost zařízení.



Obr. 17: Pokusná CAN zpráva generovaná SJA1000

6. Závěr

Daná práce se zabývala vývojem řídícího kontroléru určeného pro pálící stroj pro tvarové dělení materiálu.

Cílem bylo inovovat stávající kontrolér založený na procesoru SH7727 [1] (Renesas) dvěma různými způsoby. Buď navrhnout kontrolér s novým (lepším) procesorem nebo zvýšit funkcionalitu stávajícího kontroléru. Zpočátku byla snaha o návrh s novějším procesorem (konkrétně SH7760), ale po navržení schématu a rozpracování podkladů pro desku plošných spojů nastal problém s možností výroby těchto desek. Důvodem bylo pouzdro procesoru typu BGA, které je nutno montovat strojově a bylo to spojeno s extrémně dlouhými výrobními termíny.

Řešení popisované v této práci bylo vlastně náhradním řešením. Zmiňovaný procesor SH7760 obsahuje totiž dva CAN porty, kvůli kterým byl pokus o jeho použití. Z důvodů výše popsaných bylo implementováno do stávajícího procesoru SH7727 CAN rozhraní realizované na osvědčeném a levném autonomním CAN kontroléru SJA1000 [2] (Philips Semiconductors).

Realizace připojení SJA1000 je hodně podobná zapojení ISA CAN karty „PIKRONISA“ pro starší počítače typu PC z důvodu co nejvíce vytěžit z podpory LinCAN driveru pro tuto kartu.

Tato volba se ukázala jako úspěšná a driver LinCAN se podařilo funkční přenést na SH7727. Dokladem toho je zobrazení zkušební CAN zprávy na osciloskopu, jak ukazuje obr. 17

I když je tento procesor už starší, má stále co nabídnout a jeho výpočetní výkon stačí na širokou škálu řízených technologických procesů.

7. Literatura

[1] SH7727 Hardware Manual, Hitachi, Ltd.
Leden 2003
http://www.renesas.com
http://documentation.renesas.com/eng/products/mpumcu/e602209_sh7727.pdf

[2] SJA1000 Data Sheet, Philips Semiconductors
Leden 2000
http://www.semiconductors.philips.com/acrobat_download/datasheets/SJA1000_3.pdf

[3] Základní informace o procesoru Cirrus Logic EP9312
http://www.cirrus.com/en/products/pro/detail/P131.html

[4] Ing. František Spurný, Informace o CAN - Controller Area Network
http://fieldbus.feld.cvut.cz/can/

[5] SH-Linux, port operačního systému Linux pro SH platformu
http://www.sh-linux.org

[6] SHLinux, stránky podpory embedded Linuxu běžícím na zařízeních s procesory SH
http://www.shlinux.com

[7] Ing. Pavel Píša, Schéma zapojeni ISA CAN karty PIKRONISA
http://cmp.felk.cvut.cz/~pisa

[8] LinCAN - Linux/RT-Linux CAN driver and CANopen framework
http://cmp.felk.cvut.cz/~pisa/#can

[9] Linux, kompletní příručka administrátora, E. Nemeth, G. Snyder, T. R. Hein
Computer Press Brno 2004

[10] Administrace systému Linux, S. Shah
Grada 2002

[11] Mentor Graphics - Návrhový systém plošných spojů Power Logic/Power PCB 5.0.1
http://www.mentor.com

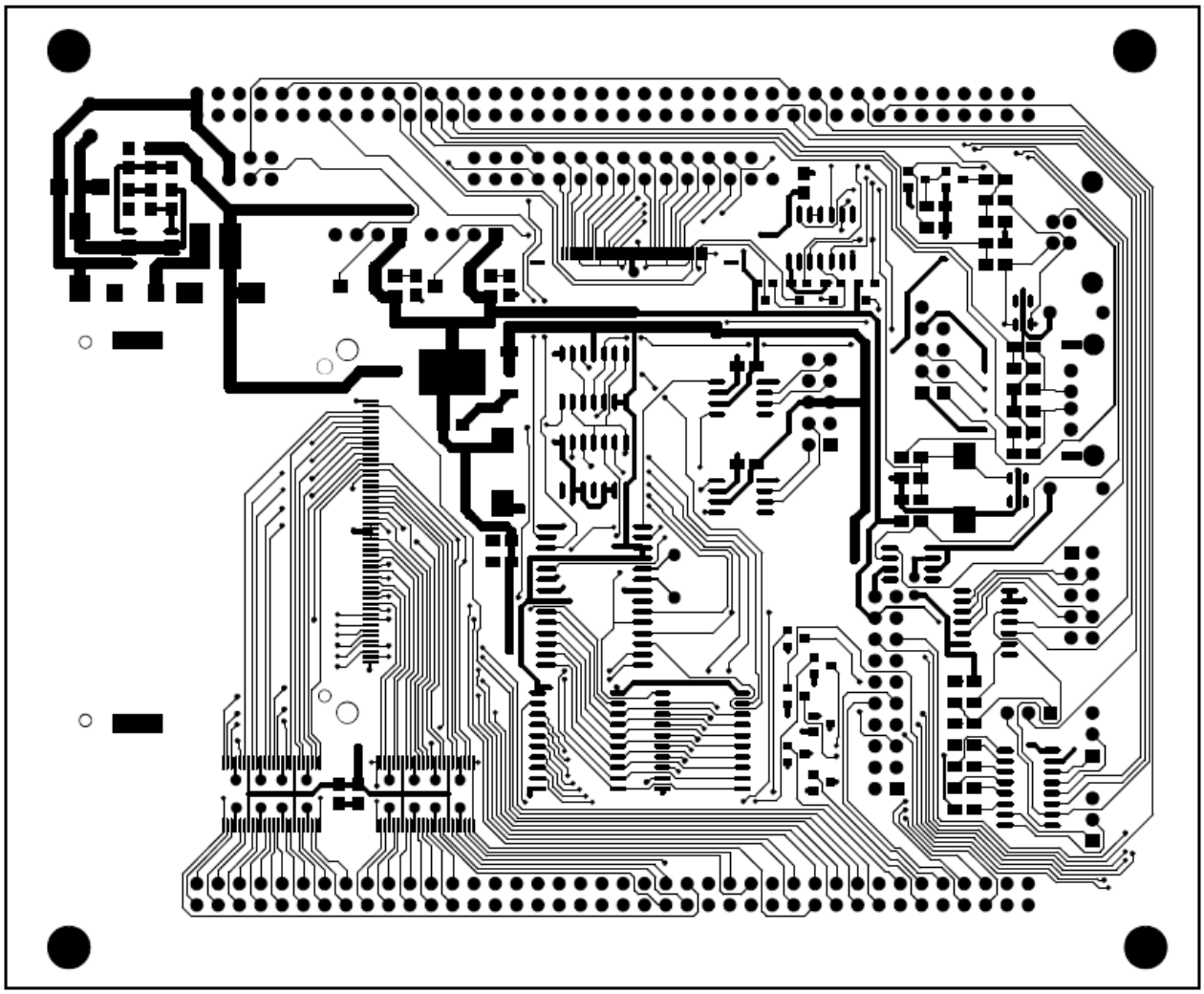
[12] GNU/Linux Debian, operační systém
http://www.debian.org

[13] Debugger Insight - The GDB GUI, grafická nástavba gdb debuggeru
http://sources.redhat.com/insight
ftp://sources.redhat.com/pub/gdb/old-releases/

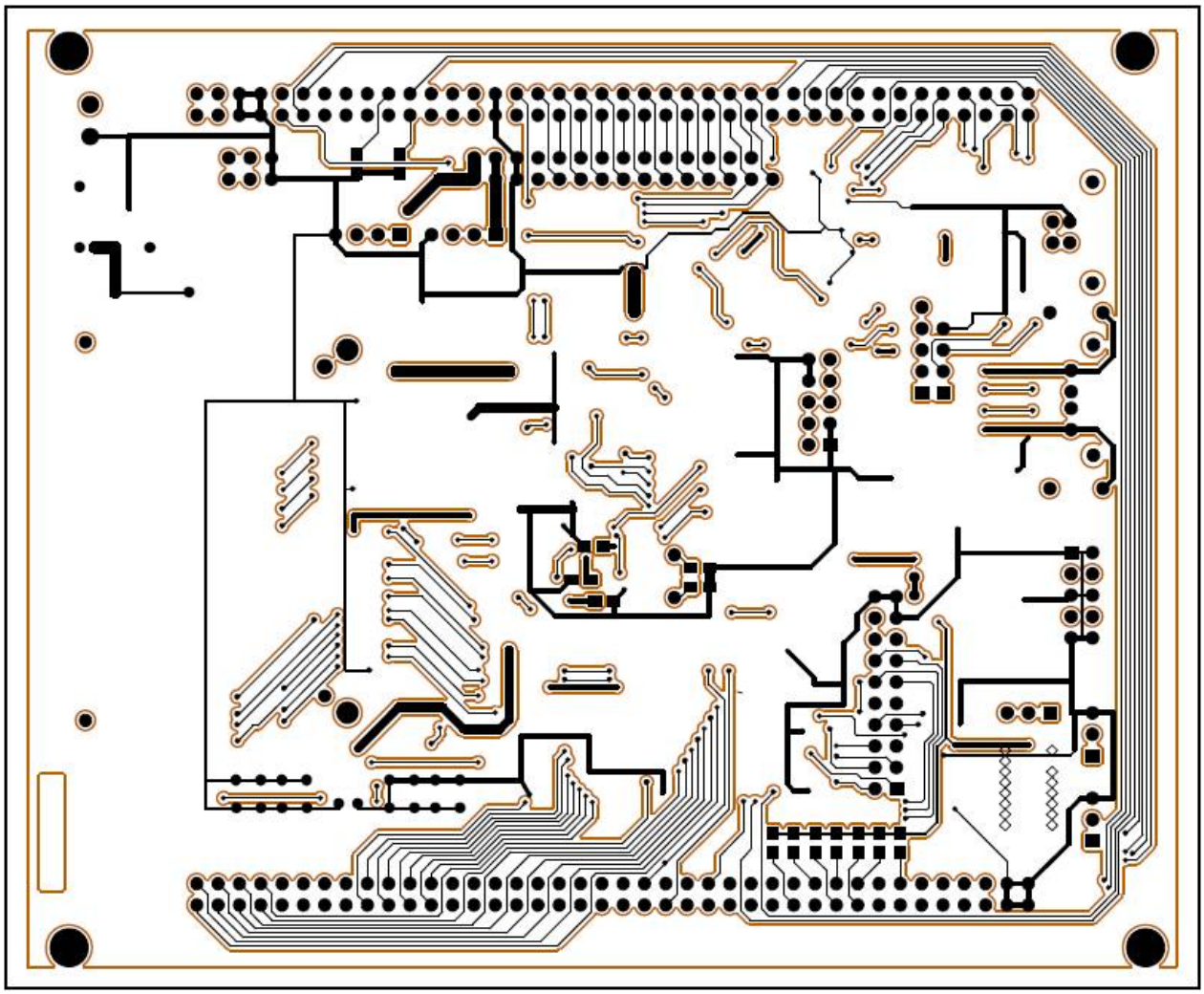
8. Přílohy

Příloha č.1: Deska plošných spojů interface SH7727IF – strana A
Příloha č.2: Deska plošných spojů interface SH7727IF – strana B
Příloha č.3: Osazovací plán deska interface SH7727IF – strana A
Příloha č.4: Osazovací plán deska interface SH7727IF – strana B
Příloha č.5: Osazená deska interface SH7727IF

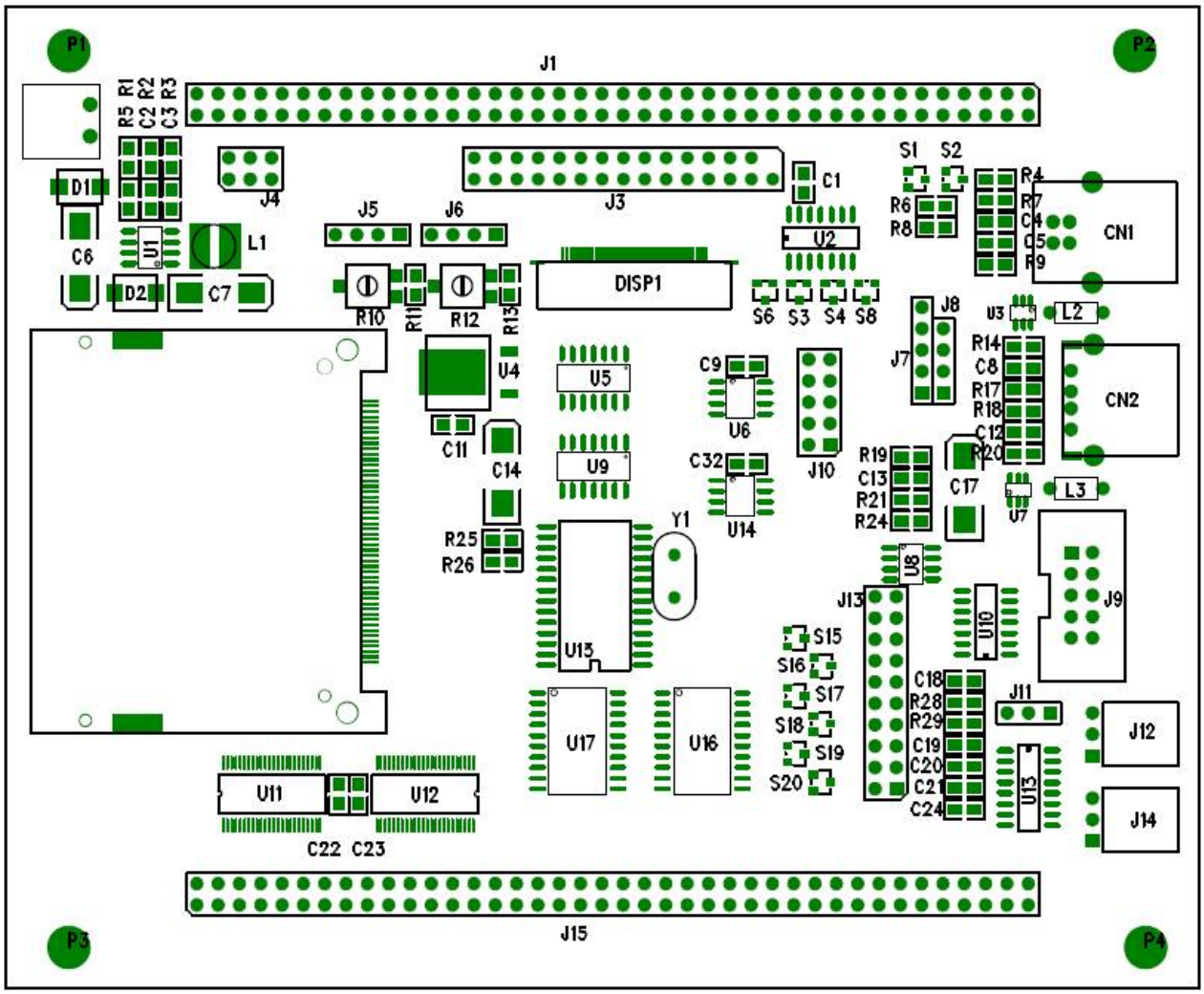
Příloha č.1: Deska plošných spojů interface SH7727IF – strana A



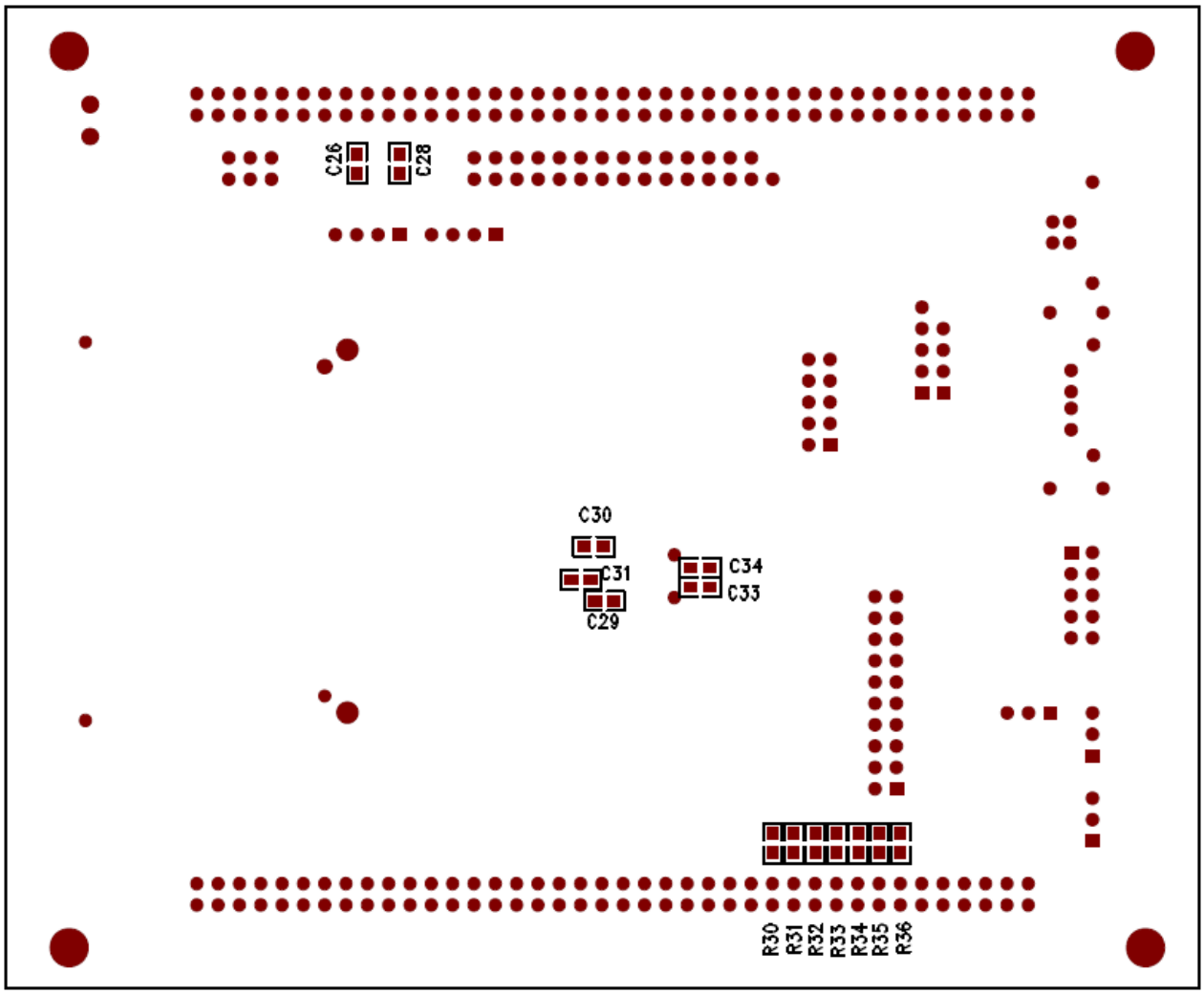
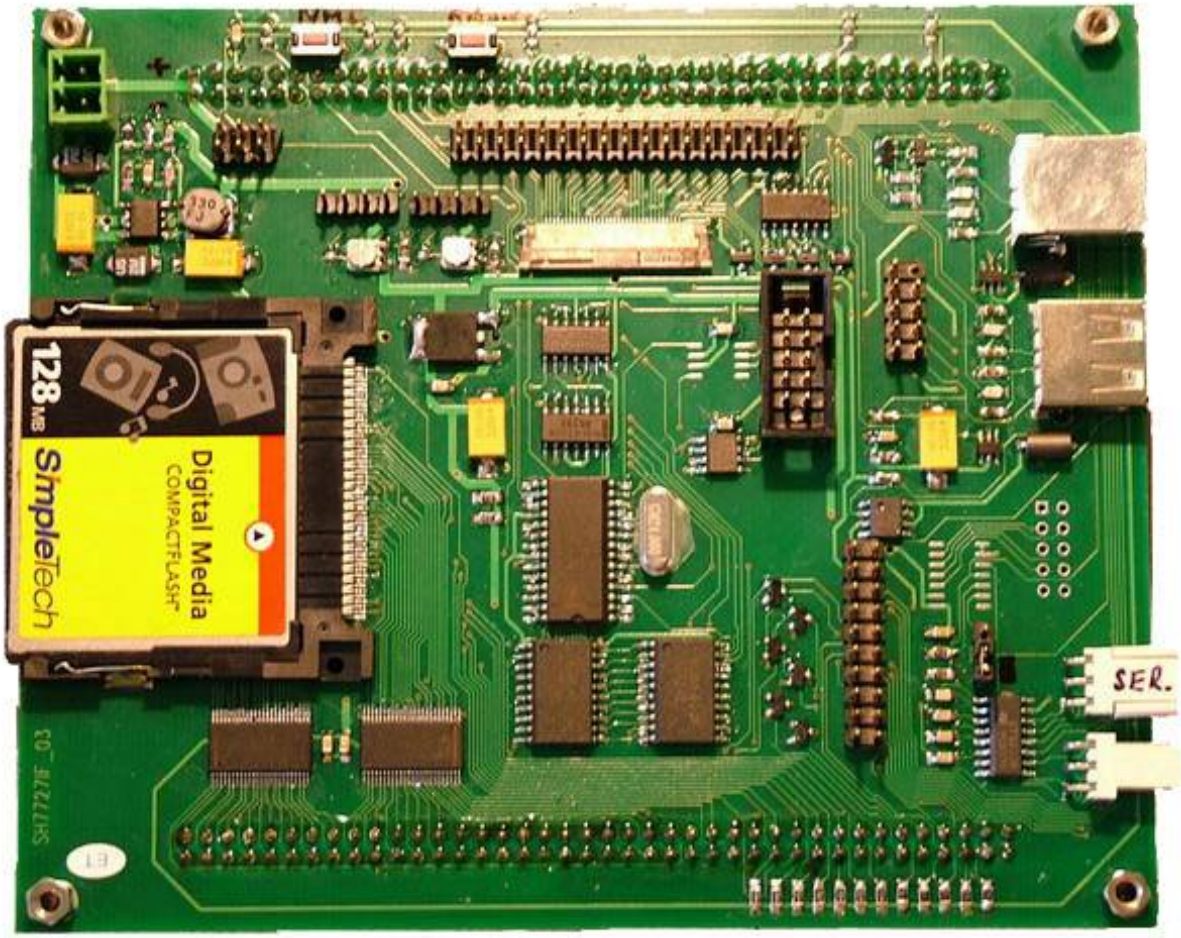
Příloha č.2: Deska plošných spojů interface SH7727IF – strana B



Příloha č.3: Osazovací plán deska interface SH7727IF – strana A



Příloha č.4: Osazovací plán deska interface SH7727IF – strana B



Příloha č.5: Osazená deska interface SH7727IF