České vysoké učení technické v Praze
Fakulta elektrotechnická

BAKALÁŘSKÁ PRÁCE

Univerzální analyzátor letadlových sběrnic

Vypracoval: Marek Maška
Vedoucí práce: Ing. Pavel Pačes, Ph.D.
Rok: 2013

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Marek Maška**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Univerzální analyzátor letadlových sběrnic**

Pokyny pro vypracování:

1. Navrhněte a otestujte hardware převodníku paketů sběrnice ARINC429 na USB.
2. Při návrhu použijte procesor LPC1759FBD80551, pro rozhraní ARINC429 použijte produkty firmy Holt, napájení převodníku bude z letadlové sítě 27 VDC a pro připojení k počítači typu PC pomocí USB rozhraní využije převodníku firmy FTDI.
3. Navrhněte zařízení včetně programového vybavení, které umožní odeslat a přijmout jeden paket a bude vybavené nejméně jedním vysílacím a přijímacím kanálem s možností nezávislého nastavení přenosové rychlosti.

Seznam odborné literatury:

[1] Levora, Tomáš: Převodník Ethernet - ARINC429. Bakalářská práce, Katedra měření FEL ČVUT, Praha 2009.
[2] Levora, Tomáš: Vývoj systému pohyblivé mapy a modulu přijímače signálu GPS s rozhraním ARINC743. Diplomová práce, Katedra měření FEL ČVUT, Praha 2011.
[3] Holt Inc.: ARINC 429, online: www.holtic.com/category/352-arinc-429.aspx

Vedoucí: Ing. Pavel Pačes, Ph.D.

Platnost zadání: do konce zimního semestru 2013/2014

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 11. 2. 2013

Poděkování

Rád bych poděkoval všem, kteří mi pomáhali při vypracování bakalářské práce. Především panu Ing. Pavlu Pačesovi, Ph.D. za vedení práce a jeho cenné rady a připomínky.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, dne ……. ……...

Marek Maška

Anotace

Práce se zabývá návrhem univerzálního analyzátoru letadlové sběrnice ARINC 429. V úvodu je popsána základní charakteristika sběrnice. Dále se práce zabývá návrhem hardwarové i softwarové části analyzátoru. Součástí hardwaru je 32bitový mikrokontrolér s jádrem ARM Cortex-M3, řadič sběrnice ARINC 429 od firmy HOLT a převodník UART/USB od firmy FTDI. Softwarová výbava mikrokontroléru je naprogramována v jazyce C, program s uživatelským rozhraním pro analýzu dat v osobním počítači s je naprogramován v jazyce Java.

Abstract

The thesis proposes a universal analyzer aircraft ARINC 429 bus. The introduction describes the main characteristics of thebus. The thesis deals with the hardware and software of the analyzer. The hardware is 32-bit microcontroller with core ARM Cortex-M3, ARINC 429 bus driver from the HOLT company and UART/USB driver from FTDI company. Software equipment of microcontroller is programmed in C. Program with a user interface for analyzing data in a personal computer is programmed in Java.

Obsah

1 Protokol ARINC 429 1
1.1 Úvod a historie 1
1.2 Fyzická vrstva 1
1.3 Struktura zpráv 4
1.4 Interpretace dat 5
2 Hardwarová realizace analyzátoru 7
2.1 Procesorová jednotka 7
2.2 Budič fyzické vrstvy ARINC 429 8
2.2.1 Sběrnice SPI 9
2.2.2 Komunikace s obvodem HI-3593 10
2.2.3 Princip fungování obvodu HI-3593 10
2.3 Připojení k počítači 11
2.4 Napájecí zdroj 12
3 Softwarová realizace analyzátoru 13
3.1 Programové vybavení mikrokontroléru 13
3.2 PC program 14
3.2.1 Funkce programu 14
3.2.2 Ovládání programu 15
3.2.3 Zpracování dat 15
3.3 Popis komunikace 16
4 Závěr 19

1 Protokol ARINC 429

V první kapitole mé práce se budu věnovat popisu letadlové sběrnice ARINC 429. Po krátkém úvodu popíši fyzickou vrstvu a poté vyšší vrstvy protokolu, které zahrnují definice struktury zpráv a rozlišují datové typy.

1.1 Úvod a historie

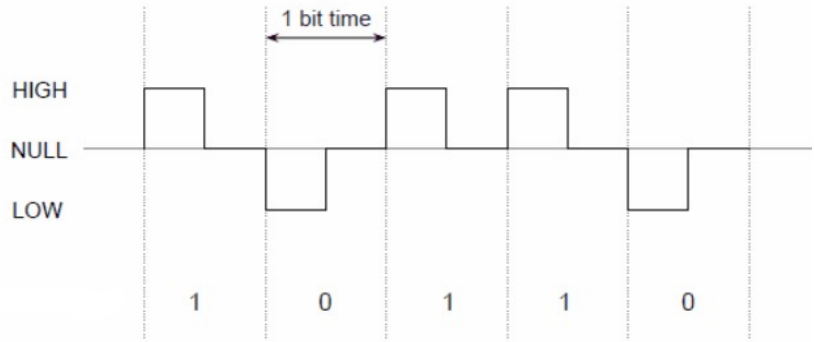
Protokol sběrnice ARINC 429 je produktem společnosti Aeronautical Radio, Inc. Jedná se o letadlovou sběrnici v komerčním letectví. Specifikace ARINC 429 je také známá pod názvem Mark 33 Digital Information Transfer System (DITS) .

Historie ARINC 429 sahá do roku 1978, kdy byla z původní specifikace ARINC 419 vyvinuta první verze ARINC 429-1. Dnešní platná verze specifikace, kterou se budu zabývat ve své práci, je ARINC 429-15 a pochází z roku 1995 [2].

Protokol funguje na principu neustálého posílání zpráv, kdy je na sběrnici jeden vysílač a až dvacet přijímačů. Každá data jednoho druhu se opakují s danou periodou a to i v tom případě, že se jejich hodnota nemění.

1.2 Fyzická vrstva

Fyzická vrstva sběrnice ARINC 429 používá jako přenosové medium stíněnou kroucenou dvoulinku s impedancí 78 Ω a se symetrickým signálem. Pro přenesení bitů je použita bipolární modulace s návratem do nuly (bipolar return-to-zero). Bipolární kódování znamená, že napětí nabývá hodnot symetrických okolo nuly. Tato modulace je naznačena na obrázku 1. Návrat napětí do nuly v polovině intervalu přenášeného bitu zajišťuje synchronizaci komunikujících jednotek, a proto není nutné vést zvlášť hodinový signál nebo se spoléhat na přesné časování, jako je tomu v případě asynchronního přenosu [2].



Obrázek 1: Bipolární modulace s návratem do nuly [2]

Definovány jsou tři hodnoty: logická jednička (HIGH), logická nula (LOW) a stav NULL. Definované velikosti napětí se na straně vysílače a přijímače liší. Vysílač musí splňovat přísnější kritéria pro velikost signálu, aby mohl přijímač správně vyhodnotit i zarušený signál. Konkrétní hodnoty vyjadřuje tabulka 1.

Stav	**Vysílač**	**Přijímač**
HIGH	10 V ± 1 V	+6,5 V až +13 V
LOW	-10 V± 1 V	-6,5 V až -13 V
NULL	0 V ± 0,5 V	-2,5 V až +2,5 V

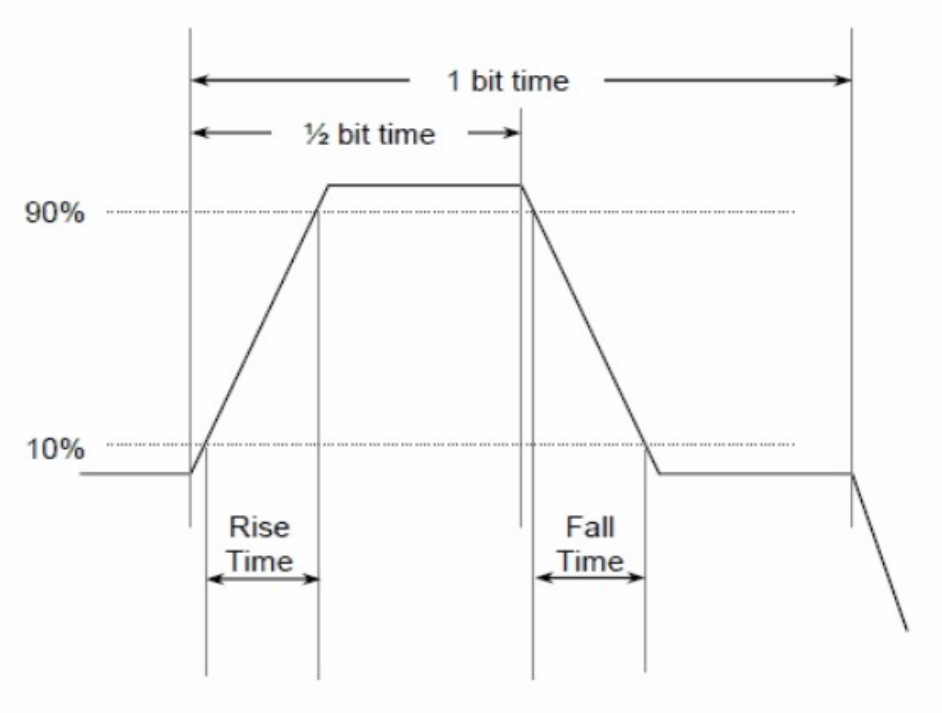
Tabulka 1: Úrovně napětí přenášených hodnot [2]

Specifikace ARINC 429 definuje dvě přenosové rychlosti. Pomalejší Low speed, která může být v rozmezí od 12 kHz do 14,5 kHz a rychlejší High speed, která má definovanou rychlost 100 kHz s možnou odchylkou 1 %. Dvě různé přenosové rychlosti nelze provozovat současně na jedné sběrnici.

Signál na vodičích kroucené dvoulinky je symetrický – pokud je na jednom vodiči napětí kladné, na druhém je záporné a naopak. V případě vysílání hodnoty HIGH je tedy na prvním vodiči +5 V a na druhém -5 V (plus tolerance). Přijímač tedy neporovnává jedno napětí oproti nule, ale napětí dvou vodičů proti sobě. Tento způsob kódování logických hodnot je v kombinaci s krouceným

párem vodičů velmi odolný vůči vnějšímu rušení.

Aby komunikace po sběrnici ARINC 429 nerušila své okolí, jsou v definici zahrnuty parametry náběžných a sestupných hran. Ve chvíli skokové změny napětí totiž vodičem teče proud, kterým se nabíjí parazitní kapacity a vstupní kapacita přijímače. Protože okamžitá hodnota proudu vychází ze vzorce $i(t)=\frac{du(t)}{dt}C$, platí, že čím rychleji se změní napětí, tím větší bude proudová špička a tím i větší rušení do okolí [6]. Konkrétní hodnoty pro High speed i Low speed jsou zachyceny na obrázku 2 v tabulce 2.



Obrázek 2: Časový průběh signálu [2]

	High speed	Low speed
Bit rate (přenosová rychlost)	100 kb/s ± 1 %	12 až 14,5 ± 1 %
1 bit time (doba 1 bitu)	10 µs ± 2,5 %	(1/Bit rate) µs ± 2,5 %
½ bit time (doba ½ bitu)	5 µs ± 25 %	(1/Bit rate) / 2 µs ± 5 %
Rise time (doba vzestupné hrany)	1,5 µs ± 0,5 µs	10 µs ± 5 µs
Fall time (doba sestupné hrany)	1,5 µs ± 0,5 µs	10 µs ± 5 µs

Tabulka 2: Časy průběhu signálů [2]

1.3 Struktura zpráv

Sběrnice ARINC 429 je typu bod-bod vždy s jedním vysílačem a až dvaceti přijímači. Přijímač neustále vysílá buď platné slovo, anebo hodnotu NULL. Každé slovo má stejnou délku 32 bitů a obsahuje vždy tyto položky: Label, SDI, Data, SSM a paritní bit. Struktura slova je znázorněna v tabulce 3.

31	30	29	28 10	9	8	7 0
P	SSM		Data	SDI		Label

Tabulka 3: Struktura slova protokolu ARINC 429 [1]

Label je osmibitový, v rámci jedné sběrnice, unikátní identifikátor slova. Pomocí něho lze najít v normovaných tabulkách význam přenášených bitů. Label se interpretuje jako trojciferné číslo složené z osmičkových číslic. Celkově je možné na jedné sběrnici použít až 255 různých labelů od 1 do 377 osmičkově [2]. Jak jsou jednotlivé číslice labelu zapsány můžeme vyčíst z tabulky 4. Label je specifický tím, že se na sběrnici vysílá v opačném pořadí, což je vidět v tabulce 5.

	LSB						MSB	
Číslo bitu	7	6	5	4	3	2	1	0
Osmičková hodnota bitu	4	2	1	4	2	1	2	1

Tabulka 4: Způsob zápisu labelu ve slově [1]

7 0	8	9	10 28	29	30	31
Label	SDI		Data	SSM		P

Tabulka 5: Pořadí odesílání bitů na sběrnici ARINC 429 [1]

Další položkou je SDI (Source/Destination Identifier). Tato položka není povinná a je s ní možné v některých případech rozšiřovat datovou oblast. Je určena k identifikaci cíle dat a použitelná je v případě, že je na sběrnici více

zařízení, která přijímají stejný druh dat. Pomocí těchto dvou bitů je možné adresovat až čtyři zařízení.

Nejvíce místa zabírá oblast pro přenos samotných dat. V datové oblasti je 19 bitů, které mohou být interpretovány různými způsoby. Konkrétní způsob interpretace určuje label společně s normovanou tabulkou. Mezi základní typy dat patří číslo zapsané binárně neboli BNR (každý bit má dvakrát větší hodnotu než bit o jedna nižší), číslo zapsané BCD kódem (jako pět jednotlivých desítkových číslic), diskrétní data (může být kombinace předchozího a hodnot typu pravda/nepravda), potvrzovací data při obousměrné komunikaci a data protokolu pro přenos souborů [1].

Dvojice bitů SSM mění význam podle druhu přenášených dat a i dle jejich konkrétního významu. Pro BCD kódování mají bity význam dle tabulky 6.

SSM	Význam při BCD kódování dat
00	plus, sever, východ, vpravo, do, nad
01	není schopný vrátit platnou hodnotu
10	test funkce periferie
11	minus, jih, západ, vlevo, od, pod

Tabulka 6: Význam bitů SSM při BCD kódování dat [1]

Při použití binárního kódování je již informace o znaménku obsažena v datech, proto se význam SSM mění. Stav 00 znamená chybové hlášení (jiné než 01) a stav 11 znamená, že je vše v pořádku a data jsou platná. Kombinace 01 a 10 má význam stále stejný. Diskrétní data mají v SSM oproti binárnímu kódování zaměněné stavy 00 a 11.

Posledním bitem v přenášeném slově je parita. Jedná se o primitivní způsob odhalení případné chyby přenosu. ARINC 429 používá lichou paritu. Poslední bit je tedy dopočítán tak, aby byl celkový počet jedniček ve slově lichý.

1.4 Interpretace dat

Ke správné interpretaci dat je potřeba znát tabulky, které popisují, jaké informace nesou jednotlivé labely. Kromě způsobu interpretace dat zde najdeme i informace o názvu veličiny, názvu její jednotky, rozsahu dat či časovém intervalu, v jakém je slovo s daným labelem vysíláno na sběrnici a podobně.

Tyto tabulky existují přepsané do XML souborů, ze kterých je možné všechny informace vyčíst. Mým úkolem bylo umožnit tyto soubory programově načítat, a tak umožnit automatickou interpretaci dat. V samotném analyzátoru je tedy pevně naprogramováno pouze rozpoznání labelu, který je vždy na stejném místě a stejně kódován. Obsah zbytku slova lze libovolně měnit pomocí změny obsahu XML souborů.

2 Hardwarová realizace analyzátoru

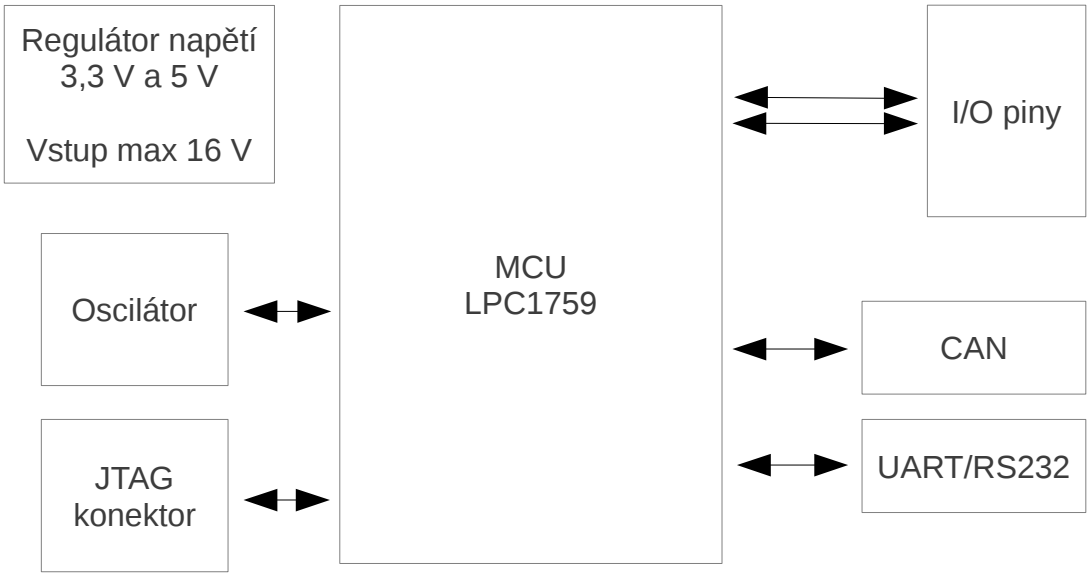
Po hardwarové stránce bylo požadováno modulární uspořádání s možností využití již hotových modulů. Modulární uspořádání je výhodné jak z důvodu, že jednou navržený modul se dá použít ve více různých zařízeních, tak z důvodu, že v jednom zařízení se dají jednotlivé moduly vyměnit za jiné.

Analyzátor se skládá ze čtyř modulů. Jsou jimi procesorová jednotka, budič fyzické vrstvy ARINC 429, modul pro připojení k počítači a napájecí zdroj.

2.1 Procesorová jednotka

Srdcem zařízení je univerzální jednotka osazená mikrokontrolérem LPC1759 od firmy NXP. Tuto jednotku jsem navrhoval v rámci semestrálního individuálního projektu. Jedná se o náhradu původní jednotky s méně výkonným mikrokontrolérem Freescale HC12. Mikrokontrolér LPC1759 je výkonný procesor s 32bitovým jádrem ARM Cortex-M3 a s taktovací frekvencí až 120 MHz. Vyrábí se v 80pinovém pouzdře LQFP80. Na desce plošných spojů jsem všech jeho 52 vstupně výstupních pinů vyvedl na header konektory. Mikrokontrolér pracuje s napájením 3,3 V a všechny jeho piny jsou 5 V tolerantní [3].

Pro napájení jednotky je možné osadit lineární stabilizátory napětí s výstupním napětím 3,3 V a 5 V nebo přivést správné napětí přímo z vnějšího zdroje. Při použití stabilizátorů LF33CDT (3,3 V) a LF50CDT (5 V) je maximální vstupní napětí 16 V [3, 12].



Obrázek 3: Blokové schéma procesorové jednotky [3]

Jednotku je možné osadit budiči sběrnic CAN a RS232. Této možnosti ale při realizaci analyzátoru ARINC 429 nevyužívám. Blokové schéma je znázorněno na obrázku číslo 3.

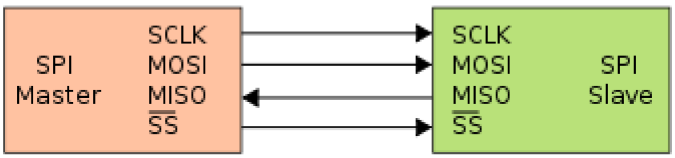
2.2 Budič fyzické vrstvy ARINC 429

Protože fyzická vrstva protokolu ARINC 429 využívá kódování s návratem do nuly, napětí ±10 V, a má konkrétní požadavky na délku náběžných a sestupných hran, je vhodné použít speciální budič této sběrnice. Pro tento účel se jako ideální jeví obvod HI-3593 od firmy HOLT, který v jednom integrovaném obvodě obsahuje dva přijímače a dva vysílače sběrnice ARINC 429. Pracuje s napětím 3,3 V a disponuje integrovaným měničem napětí na ±10 V, pro jehož správnou funkci je potřeba připojit jen dva kondenzátory [5].

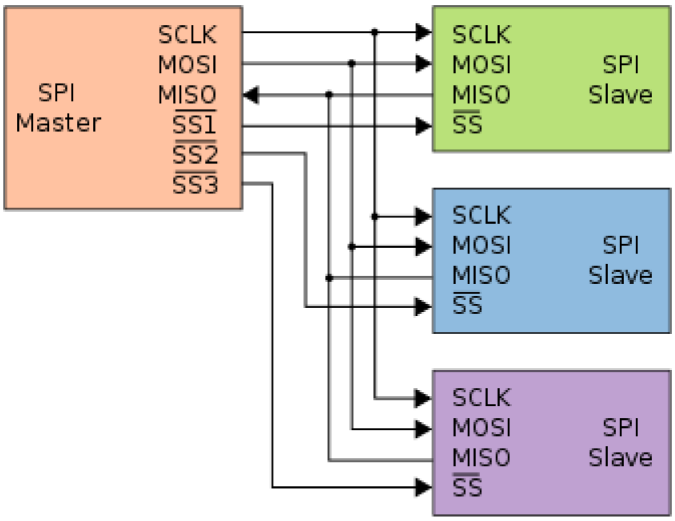
Obvod ale není pouze převodníkem napěťových úrovní. Jedná se o řadič, který se stará o celé vysílání a příjem jednotlivých slov. Obsahuje vstupní a výstupní paměť typu FIFO, řeší správnou rychlost přenosu, může automaticky počítat a kontrolovat paritu vysílaných i přijímaných slov. Komunikace s procesorem probíhá pomocí sběrnice SPI rychlostí až 10 MHz [5].

2.2.1 Sběrnice SPI

Sběrnice SPI používaná převodníkem HI-3593 slouží k sériovému přenosu dat. Sběrnice je typu master/slave, přičemž master je vždy jeden (zpravidla procesor) a zařízení typu slave může být více. Master mezi nimi přepíná pomocí pinů $\overline{CS}$ ($\overline{SS}$). Standard SPI definuje signály SCLK, MOSI a MISO. SCLK je hodinový signál, který vysílá master a jeho vzestupná respektive sestupná hrana určuje platnost dat. Zkratka MOSI znamená Master Out, Slave In – tedy z pohledu master se jedná o vysílací kanál, z pohledu slave o kanál přijímací. Signál MISO pak slouží pro komunikaci opačným směrem [9]. Způsob připojení SPI zařízení na sběrnici je znázorněno na obrázcích 4 a 5.



Obrázek 4: SPI – jedeno zařízení typu slave [10]



Obrázek 5: SPI – tři zařízení typu slave [11]

2.2.2 Komunikace s obvodem HI-3593

Vzhledem k tomu, že obvod není jen prostým převodníkem napěťových úrovní, je komunikace s ním řešena pomocí instrukcí. Přenos začíná master stažením pinu $\overline{CS}$ do nuly (aktivní stav). Poté následuje 8bitová instrukce. S každou vzestupnou hranou na SCLK se odesílá jeden bit. První bit v instrukci je příznakem, zda se jedná o čtení (logická 1), nebo zápis (logická 0). V případě zápisu master po odvysílání instrukce dále pokračuje ve vysílání dat. Délka dat se liší podle použité instrukce. Může to být 0 až 32 bajtů. V případě požadavku na čtení po odvysílání instrukce slave odpoví. Délka dat je také proměnlivá v rozmezí 1 až 32 bajtů.

2.2.3 Princip fungování obvodu HI-3593

Pro správnou funkci obvodu je potřeba přivést vnější hodinový signál na pin ACLK a nastavit děličku, aby výsledná frekvence byla 1 MHz. Vnější signál může mít frekvenci 1 MHz a poté sudé hodnoty od 2 MHz do 30MHz. Nastavení děličky se provádí odesláním instrukce 0x38 následované 1 bajtem – dělitelem. Já jsem použil signál s frekvencí přímo 1 MHz, proto jsem nastavil děličku na 0x00 (dělička je vypnutá).

Před zahájením přijímání nebo odesílání znaků na sběrnici ARINC 429 je potřeba nastavit ovládací registry přijímačů a vysílače (i když výchozí stav by v některých případech mohl také vyhovovat). Registry se nastavují instrukcemi 0x10 a 0x24 (přijímací registry) a instrukcí 0x08 (vysílací registr). Dá se v nich nastavit například způsob zacházení s paritními bity, otočení bitů v labelu do „přirozeného“ pořadí a další. Já jsem ve své práci zejména využil přepínání přenosové rychlosti mezi High speed a Low speed, která musí být dle zadání volitelná.

Pro odeslaní znaků je nejdříve potřeba zapsat znaky do výstupní FIFO paměti o velikosti 32 slov. Dle nastavení vysílacího registru (bit 5 – TMODE) se znaky z FIFO vysílají ihned, nebo je přenos potřeba odstartovat zapsáním instrukce 0x40. Pro kontrolu obsazenosti výstupního bufferu je možné

kontrolovat vysílací stavový registr, kde jednotlivé bity určují, zda je fronta prázdná, z poloviny obsazená nebo plná.

Přijímaná slova se ukládají do vstupní FIFO paměti také o velikosti 32 slov. Každý přijímací kanál má svojí vlastní paměť. Obsazenost vstupní paměti je možné kontrolovat v přijímacích stavových registrech. Pokud zjistíme nějaké přijaté slovo, je potřeba o jeho zaslání do procesoru požádat instrukcí 0xA0 pro první přijímací kanál respektive 0xC0 pro druhý. Řadič na tuto instrukcí odpoví 32bitovou zprávou obsahující ARINC 429 slovo.

2.3 Připojení k počítači

Pro komunikaci s počítačem lze použít různé moduly. V této práci jsem se zabýval pouze možností připojení k počítači přes USB. Dalšími možnostmi je například využití Ethernetového nebo WiFi modulu.

Využil již hotový převodní UART/USB, který je osazen čipem FT232 od firmy FTDI. Po připojení k počítači se převodník tváří jako obyčejný sériový port, což velmi usnadňuje komunikaci. O distribuci ovladačů se stará výrobce čipu a ovladače jsou dostupné pro téměř všechny současné operační systémy od Windows 98, Linux 2.4, Mac OS 8/9 po všechny novější verze těchto operačních systémů [13].

Převodník dosahuje vysoké přenosové rychlosti až 3 Mbaud. Zvolená rychlost nemůže být zcela libovolná. Při výběru přenosové rychlosti je potřeba postupovat podle vzorce uvedeného v manuálu [13]. Vzhledem k tomu, že ani v mikrokontroléru nelze přenosovou rychlost volit libovolně a je nutné respektovat omezené možnosti děličky, zvolil jsem rychlost 1 Mbaud. Jelikož data po sběrnici ARINC 429 přicházejí rychlostí maximálně 100 kb/s, je zvolená rychlost dostatečně vysoká na to, aby bylo možné přenášet dva ARINC 429 kanály do PC i s časovými značkami a synchronizací.

2.4 Napájecí zdroj

V letadlech se používá napájecí napětí o velikosti 27 V. Součástí zadání byl tedy požadavek, aby analyzátor mohl být tímto napětím napájen. V mém případě je možné použít 5V napájení z USB komunikačního modulu. Protože zařízení má umožňovat nahrazení komunikačního modulu za jiný, bylo potřeba vyřešit napájení i z 27 V. Vzhledem k tomu, že běžné lineární stabilizátory nejsou schopny s takto vysokým vstupním napětím pracovat, bylo potřeba zajistit napájení se spínaným zdrojem (konkrétně stabilizátory osazené na procesorové desce pracují se vstupním napětím jen do 16 V [3, 12]).

Napájecí moduly se spínanými zdroji máme na katedře měření k dispozici, nicméně pro účely tohoto analyzátoru jsou zbytečně výkonné, tedy i velké, a jejich zapojení obsahuje hodně součástek. Mým úkolem bylo navrhnout napájecí zdroj co nejmenší a nejjednodušší s rozsahem vstupního napětí zahrnujícím 27 V, s výstupním napětím 5 V a jehož proudová zatížitelnost může být méně než 1 A.

Dle těchto parametrů jsem vybral spínaný zdroj LT1933 od firmy Linear Technology. Jedná se o step-down spínaný zdroj s nastavitelným výstupním napětím. Vstupní rozsah napětí je 3,6 V – 36 V. Při výstupním napětí 5 V je potřeba dodat na vstupu minimálně 6,3 V a výstup je zatížitelný maximálním proudem 500 mA. Při vstupním napětí nad 16 V, je maximální výstupní proud 600 mA [14].

Hlavní předností tohoto obvodu je jeho velikost. Vyrábí se v pouzdře SOT-23 se šesti piny. Také nízké hodnoty externích komponent (kondenzátorů a cívky) umožňují vybrat součástky v malém pouzdře. Vstupní i výstupní kondenzátory jsem vybral keramické v SMD pouzdrech 0805 a 1206. Díky tomu bylo možné dosáhnout rozměrů desky plošných spojů 32x13 mm.

3 Softwarová realizace analyzátoru

Oživení analyzátoru pomocí softwaru znamenalo vytvořit dva programy. Jeden pro mikrokontrolér LPC1759, který zajišťuje komunikaci s řadičem sběrnice ARINC 429 po SPI a s počítačem prostřednictvím rozhraní UART, které je déle převedeno na USB.

Druhý program pro počítač musel umět komunikovat s mikrokontrolérem prostřednictvím virtuálního sériového portu a dále zpracovávat a generovat data.

3.1 Programové vybavení mikrokontroléru

Program v mikrokontroléru má poměrně jednoduchou úlohu. Je pouze mezičlánkem mezi počítačem a řadičem sběrnice ARINC 429. Program jsem napsal v jazyce C s použitím vývojového prostředí LPCXpresso. Při vývoji jsem použil knihovnu CMSIS LPC17xx (v 2.00), která například definuje názvy registrů a obsahuje funkce pro jednoduché povolení přerušení. Neobsahuje funkce pro práci s periferiemi. Knihovnu pro práci s periferiemi jsem sice našel na stránkách výrobce, ale nevyužil jsem ji. S periferiemi pracuji pomocí přímého zápisu hodnot do registrů.

Po zapnutí napájení nejdříve mikrokontrolér vypočítá a nastaví PLL tak, aby dosáhl požadované pracovní frekvence, která je v tomto případě 112 MHz. Na nastavení frekvence jsem vytvořil funkci, která jako první parametr přebírá cílový kmitočet a jako druhý parametr kmitočet krystalu. Obojí v MHz. Parametry mají typ double, takže jde použít i desetinná místa pro jemnější rozlišení, než je 1 MHz. Funkce poté vypočte hodnoty nastavení PLL tak, aby odchylka požadované a skutečné frekvence byla co nejmenší a zároveň PLL nastaví.

Déle následuje zapnutí přerušení a nastavení periferií. Zejména UARTu pro komunikaci s počítačem, SPI pro komunikaci s řadičem sběrnice ARINC 429 a časovačů. Ty jsou použity pro generování signálu o frekvenci 1 MHz, kterou

řadič používá jako svůj hodinový signál, generování přerušení s periodou 1 ms, které slouží jako hodinový signál pro počítadlo času (hodnota pro časové značky).

Hlavní činností mikrokontroléru je poté přijímání instrukcí z počítače, které vyhodnocuje a zachová se podle jejich obsahu. Pokud se jedná o instrukci vyzývající k zahájení příjmu slov ze sběrnice ARINC 429, mikrokontrolér nejprve vynuluje počítadlo času (hodnota pro časové značky) a začne kontrolovat příslušný příchozí stavový registr řadiče HI-3593. Pokud zjistí, že byla přijata data, pošle řadiči instrukci s příkazem k vyčtení dat, po jejich přijetí k nim přidá časovou značku, identifikační bajt a synchronizační bajty a celou zprávu odešle do počítače.

V případě příjmu datové zprávy z počítače si mikrokontrolér zprávu nejdříve uloží na zásobník a ve chvíli, kdy se časová značka zprávy shoduje s vnitřním počítadlem času, ji odešle do řadiče HI-3593.

3.2 PC program

Ovládací program analyzátoru jsem programoval v jazyce Java. Díky tomu je program multiplatformní. Testován byl v operačním systému Windows a Linux.

Pro práci s virtuálním sériovým portem jsem použil hotovou knihovnu jSSC (java-simple-serial-connector) [7], která s ním umožňuje bezproblémově a jednoduše pracovat. Dále jsem použil knihovnu JfreeChart [8], pomocí které jsou vykreslovány grafy interpretovaných přijímaných dat.

3.2.1 Funkce programu

Základní funkcí programu je příjem a odesílání slov protokolu ARINC 429. V případě příjmu je možné aktuálně přijímaná data zobrazit rovnou v grafu. Kromě toho je možné všechna data ukládat do souborů na pevný disk, přičemž každý label má svůj vlastní soubor. Ukládá se vždy originální slovo v hexadecimálním tvaru, časová značka a také interpretovaná data. Všechna data jsou oddělená středníkem a lze je podobně jako CSV soubory importovat do

aplikace Excel nebo jiného tabulkového procesoru.

Pro interpretaci dat se využívá XML souborů, které obsahují popisy jednotlivých labelů. Při zahájení přenosu dat se všechny XML soubory z vybrané složky parsují, a data z nich se ukládají v instancích třídy WordSpec.

Součástí programu je i generátor dat umožňující generovat konstantu, sinusový signál a jiné a odesílat je na sběrnici ARINC 429.

3.2.2 Ovládání programu

Program je ovládán pomocí stromu v levé části okna. V jedné chvíli je možné připojit teoreticky neomezené množství analyzátorů. Připojený hardware se automaticky rozpozná a zobrazí. Po rozevření větve se zařízením jsou vidět dostupné vysílací a přijímací kanály. V jednotlivých kanálech je poté seznam přijímaných respektive vysílaných labelů. Je třeba dbát na to, že se zde zobrazují pouze labely, k nimž je dostupný XML soubor s popisem jejich významu, ostatní labaly jsou ignorovány.

Obsah pravé horní části okna se mění podle aktuálně vybrané položky. Nastavují se zde parametry analyzátoru, zadávají hodnoty pro generátor dat, nebo se zde zobrazuje graf přijímaných dat.

Dolní část pravé části okna je vyhrazena pro trvalé zobrazení grafu, do kterého jdou přidat libovolné položky přijímaných slov. Přidání nebo odebrání položky se provádí dvojklikem myši na příslušnou položku ve stromě. Levé tlačítko přidá datovou řadu k levé ose y, pravé tlačítko přidá datovou řadu k pravé ose y.

3.2.3 Zpracování dat

Program obsahuje abstraktní třídu Connection, kterou dědí třída PortController, a ta se stará o práci se sériovým portem. V případě nahrazení modulu USB/UART za jiný je potřeba napsat příslušnou třídu jako náhradu za PortController, která bude dědit Connection. Současně bude zřejmě potřeba provést další malé úpravy programu.

Data přijatá sériovým portem se ukládají nejprve v přijímacím bufferu (ten jsem implementoval jako pole typu byte). Pokud je v bufferu dostatek bajtů, aby to vydalo na jednu celou zprávu včetně synchronizace a časové značky, jsou tato data přečtena, je zkontrolována synchronizace a následně jsou data uložena do instance třídy ArincWord, která obsahuje jak ARINC 429 slovo, tak časovou značku. V tomto tvaru jsou data skladována ve dvou různých bufferech. Jeden slouží jako úložiště před zápisem na disk, a druhý jako úložiště pro zobrazení v programu.

Před zápisem na disk jsou data bufferována z toho důvodu, že jednotlivé labely přicházejí velmi často (s periodou v řádu jednotek až stovek milisekund). Při zápisu na disk nejdéle trvá právě zahájení zápisu (kvůli době stabilizace čtecí hlavy). Proto je výhodnější zapisovat data po větších celcích. Pokud konkrétní label překročí v bufferu nastavenou velikost, tak se všechny slova s tímto labelem předají třídě pro zápis do souborů. Tím se buffer od daného labelu vyčistí.

Oproti tomu buffer pro okamžité zobrazování dat je cyklický. Pokud se zaplní, tak se odstraní pouze nejstarší slovo. Tímto je zajištěno, že uživatel může sledovat na monitoru vždy aktuální data, zároveň jejich množství při dlouhém měření zůstává stejné, a díky tomu nezaplňují zbytečně paměť.

3.3 Popis komunikace

Komunikace mezi počítačem a mikrokontrolérem je realizována pomocí virtuálního sériového portu na straně počítače a UARTu na straně MCU. Jedná se o asynchronní kanál s předem danou rychlostí přenosu. Struktura přenášených dat je pevně dána. Každé vysílání začíná start bitem (logická 0), poté následuje osm významových bitů, paritní bit (používám lichou paritu) a nakonec stop bit (logická 1). Pro odeslání jednoho ARINC 429 slova, které má 32 bitů, jsou tedy potřeba čtyři slova (bajty) sériového přenosu.

Kromě samotných dat ze sběrnice ARINC 429 je nutné mezi počítačem a mikrokontrolérem přenášet i informace o cílovém/zdrojovém kanálu. Tuto

informaci je potřeba přenášet společně s každým odesílaným slovem. Z počítače je navíc nutné umožnit zaslání pokynu k nastavení přenosové rychlosti jednotlivých kanálů a pokyn k zahájení nebo ukončení příjmu ARINC 429 slov. Komunikačním protokol tedy musí rozlišovat mezi datovými zprávami a instrukčními zprávami.

Protože na USB sběrnici dochází ke zpožděním v řádu milisekund, posílají se současně s daty také časové značky. Tyto značky jsou 32bitová čísla určující okamžik příjmu nebo odeslání slova řadičem v milisekundách od začátku zahájení přenosu. Vyskytují se v komunikaci oběma směry. Ve směru do počítače přidává značky mikrokontrolér a počítač je pak spolu s daty ukládá a využívá je při vykreslování dat do grafů. Ve směru do mikrokontroléru slouží k taktování okamžiku odeslání dat do řadiče HI-3593, a tedy na sběrnici ARINC 429.

Struktura zprávy je následující. Nejdříve se odesílá identifikační bajt (viz tabulka 7), jehož horní čtyři bity rozlišují mezi datovou zprávou (0x3) a instrukční zprávou (0xC). Dolní čtyři bity určují číslo přijímacího nebo vysílacího kanálu, kterého se zpráva týká. Oba typy kanálů jsou číslovány 0-7. Nejvyšší bit tohoto čtyřbitového slova rozhoduje o tom, jestli se jedná o kanál vysílací nebo přijímací. V případě, že se jedná o instrukční zprávu, následuje instrukce s délkou jednoho bajtu. Pokud se jedná o datovou zprávu, je identifikační bajt následován čtyřbajtovou časovou značkou, a ihned poté čtyřbajtovým slovem sběrnice ARINC 429. Stejným způsobem probíhá komunikace oběma směry.

Data (0x3) / instrukce (0xC)				$Tx/\overline{Rx}$	Číslo kanálu		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Tabulka 7: Struktura identifikačního bajtu

V případě ztráty jednoho bajtu dojde nejen ke ztrátě významu aktuálně přenášené zprávy, ale i k posunutí významu všech dalších přijímaných zpráv.

Existují dva způsoby synchronizace, kterými se dá tomuto problému zabránit. Prvním je sledování času přijatých bajtů. Za jedno slovo lze pak považovat pouze bajty, které byly přijaty ihned po sobě (je nutné zvolit rozumně krátký timeout). Druhý způsob se provádí přidáním synchronizačních bajtů. V této práci jsem vybral druhý způsob, a to zejména proto, že převodník UART/USB nepracuje v reálném čase. Je to dáno charakteristikou sběrnice USB, která využívá komunikaci v 1 ms dlouhých rámcích. Proto nelze použít čas přijatých bajtů jako orientační bod pro synchronizaci.

Synchronizační bajty se na vysílací straně přidají k datům pravidelně na začátku každé zprávy. Přijímací strana tyto bajty očekává, a pokud je nezaznamená, tak zprávu vyhodnotí jako neplatnou. V takovém případě přijímací strana ignoruje všechna další data až do příjmu následujících synchronizačních bajtů. Pokud se ve zprávě náhodou objeví data, která vypadají jako synchronizace, přijímač tuto skutečnost ignoruje, protože délka zprávy je pevně určena tím, jestli se jedná o instrukci nebo o data. Tím je zabezpečeno, že pokud bude komunikace probíhat bezchybně, nedojde nikdy k falešné synchronizaci. Uvedený proces ovšem není schopen zaručit správnost přijímaných dat. Stará se pouze o to, aby se jedna chyba nešířila po celou dobu trvání přenosu.

Jako synchronizační bajt jsem zvolil číslo 0xAA. Toto číslo je zajímavé tím, že se v něm střídají jedničky a nuly (10101010). Pro snížení pravděpodobnosti výskytu dat, která by měla stejnou podobu jako bajt synchronizační, jsem se rozhodl vysílat dva synchronizační bajty po sobě.

V případě potřeby je možné jejich počet snadno zvýšit nebo použít komplikovanější řešení. To spočívá v použití jednoho synchronizačního bajtu, kdy ale přijímač počká na přijetí více zpráv (například čtyř) a až poté zkontroluje, jestli jsou všechny synchronizační bajty na svých místech. Platnost takto přijatých dat by se nepotvrzovala jako celek, ale klouzavě vždy jen pro první data ve frontě. Programováním tohoto řešení jsem se ale nezabýval.

4 Závěr

Analyzátor letadlové sběrnice ARINC 429 v tuto chvíli splňuje hlavní požadavky, jimiž bylo vytvoření hardware, spojení s počítačem a přijímání a odesílání jednotlivých slov. Každému přenosovému kanálu lze nastavit samostatně přenosovou rychlost na High speed nebo Low speed. Počítačový program umí sám rozpoznávat nově připojená zařízení a zobrazovat jimi přijímaná slova.

Funkce převodníku byla testována ve smyčce, kdy vysílací kanál byl přiveden přímo na jeden přijímací kanál obvodu HI-3593. Pro test byla použita dříve zaznamenaná data profesionálním produktem. K dispozici jsem měl jednotlivá 32bitová slova ARINC 429 a zároveň jejich interpretaci v číselných hodnotách. Porovnáním těchto hodnot s hodnotami z mého analyzátoru jsem ověřil, že analyzátor interpretuje data stejným způsobem – dá se tedy předpokládat, že správně.

Další test se týkal generátoru dat sinusového signálu. Vygenerovaný signál jsem jedním kanálem odesílal, druhým přijímal zpět a zobrazoval v grafu. Přijatý signál měl sinusový průběh odpovídající zadaným parametrům.

Přepínání přenosové rychlosti vysílacího kanálu jsem úspěšně ověřil pomocí osciloskopu. Přepínání rychlosti přijímacích kanálů jsem ověřil pouze nepřímo. Pokud ve zpětné smyčce nastavené rychlosti přijímacího a vysílacího kanálu lišily, nebyly přijímány žádná slova. V případě, že měl přijímací kanál stejné nastavení rychlosti jako kanál vysílací, slova byla přijímána při High speed i Low speed. Z toho lze usuzovat, že přepínání rychlostí funguje i u obou přijímacích kanálů.

Uživatelské rozhraní umožňuje zobrazovat zejména grafy přijímaných signálů. Ostatní funkce jako je nastavení generátoru signálu nebo načtení uložených dat jsem nestihl implementovat. Pro využití všech možností analyzátoru by bylo vhodné tyto funkce doplnit.

Literatura

[1] Levora, Tomáš: Převodník Ethernet – ARINC 429. Bakalářská práce, Katedra měření FEL ČVUT, Praha 2009.

[2] ARINC 429 Protocol Tutorial, Avionics Interface Technologies, dostupné na: http://aviftech.com/files/2213/6387/8354/ARINC429_Tutorial.pdf

[3] Maška, Marek: Procesorová jednotka s LPC1759. Prezentace k individuálnímu projektu, FEL ČVUT, Praha 2013.

[4] Katalogový list rodiny procesorů LPC175x, volně dostupný na: http://www.nxp.com/products/microcontrollers/cortex_m3/LPC1759FBD80.html

[5] Katalogový list obvodu HI-3593, volně dostupný na: http://www.holtic.com/products/3026-hi-3593.aspx

[6] Záhlava, Vít. Návrh a konstrukce desek plošných spojů. Vyd. 1. Praha: Česká technika - nakladatelství ČVUT, 2005.

[7] Java knihovna funkcí pro komunikaci prostřednictvím sériového portu. https://code.google.com/p/java-simple-serial-connector/

[8] Java knihovna funkcí pro vykreslování grafů. http://www.jfree.org/jfreechart/

[9] Serial Peripheral Interface Bus http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus

[10] Blokové schéma SPI, http://en.wikipedia.org/wiki/File:SPI_single_slave.svg

[11] Blokové schéma SPI, http://en.wikipedia.org/wiki/File:SPI_three_slaves.svg

[12] Katalogový list obvodu LFxxCDT, volně dostpné na: http://www.st.com/web/catalog/sense_power/FM142/CL1015/SC312/PF63606

[13] Katalogový list obvodu FT232RL, volně dostpné na: http://www.ftdichip.com/Products/ICs/FT232R.htm

[14] Katalogový list obvodu LT1933, volně dostpné na: http://www.linear.com/product/LT1933