

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV MIKROELEKTRONIKY

DEPARTMENT OF MICROELECTRONICS

IMPLEMENTACE VZDÁLENÉHO TERMINÁLU NA SBĚRNICI MILBUS

IMPLEMENTATION OF REMOTE TERMINAL ON MILBUS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

Petr Čechura

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. Vojtěch Dvořák, Ph.D.

BRNO 2023



Bakalářská práce

bakalářský studijní program **Mikroelektronika a technologie**

Ústav mikroelektroniky

Student: Petr Čechura

ID: 230234

Ročník: 3

Akademický rok: 2022/23

NÁZEV TÉMATU:

Implementace vzdáleného terminálu na sběrnici Milbus

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s požadavky kladenými na vzdálený terminál (RT) na sběrnici MIL-STD-1553B. Prostudujte standardy MIL-STD-1553B a ECSS-E-ST-50-13C a definujte požadavky a architekturu terminálu na sběrnici Milbus. Následně popište rozhraní v jazyce VHDL a provedte verifikaci návrhu vzhledem ke stanoveným požadavkům a vzorovou implementaci do zvoleného obvodu FPGA.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce.

Termín zadání: 6.2.2023

Termín odevzdání: 1.6.2023

Vedoucí práce: Ing. Vojtěch Dvořák, Ph.D.

doc. Ing. Pavel Šteffan, Ph.D.

předseda rady studijního programu

UPOZORNĚNÍ:

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

ABSTRAKT

Tato bakalářská práce se zabývá komunikační sběrnicí Milbus a návrhem vzdáleného terminálu do obvodu FPGA na hradlové úrovni pomocí jazyka VHDL.

První kapitola se věnuje standardu MIL-STD-1553B – rozebírá funkcionality, které sběrnice obsahuje, odkrývá pravidla komunikace a možná omezení.

Druhá kapitola seznamuje čtenáře se standardem ECSS-E-ST-50-13C, popisující použití sběrnice ve vesmírných technologiích, nebo obecně robustních, komplexních systémech, kde je vyžadováno efektivní řízení komunikace.

V poslední (praktické) části je na základě poznatků z předchozích kapitol vytvořena architektura vzdáleného terminálu s cílem poskytnout základní funkce (příjem a odesílání dat). Na základě stanovených požadavků je architektura verifikována a v závěru je provedena syntéza do obvodu FPGA Spartan3, model XC3S50.

KLÍČOVÁ SLOVA

MIL-STD-1553B, ECSS-E-ST-50-13C, Milbus, FPGA, VHDL, sběrnice, přenos dat, vojenské technologie

ABSTRACT

This bachelor thesis deals with the Milbus communication bus and the design of a remote terminal into a gate-level FPGA circuit using VHDL.

The first chapter is dedicated to the MIL-STD-1553B standard - it discusses the functionalities that the bus contains, revealing the communication rules and limitations.

The second chapter introduces the reader to the ECSS-E-ST-50-13C standard, describing the use of the bus in space technology, or in general, robust, complex systems where effective communication control is required.

In the last (practical) part, the architecture of the remote terminal is developed based on the knowledge gained in the previous chapters in order to provide the basic functions (receiving and sending data). Based on the specified requirements, the architecture is verified and finally synthesized into a Spartan3 FPGA, model XC3S50.

KEYWORDS

MIL-STD-1553B, ECSS-E-ST-50-13C, Milbus, FPGA, VHDL, bus, data transfer, military technology

ČECHURA, Petr. *Implementace vzdáleného terminálu na sběrnici Milbus*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav mikroelektroniky, 2022, 72 s. Bakalářská práce. Vedoucí práce: Ing. Vojtěch Dvořák

Prohlášení autora o původnosti díla

Jméno a příjmení autora: Petr Čechura

VUT ID autora: 230234

Typ práce: Bakalářská práce

Akademický rok: 2022/23

Téma závěrečné práce: Implementace vzdáleného terminálu na sběrnici Milbus

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno
.....
podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce doktoru Vojtěchu Dvořákovi za ochotu při konzultacích, cenné rady a skutečný zájem o problematiku.

Obsah

Úvod	12
Teoretická část studentské práce	13
1 Sběrnice MIL-STD-1553B	13
1.1 Základní popis	13
1.2 Stručná historie	13
1.2.1 Použití sběrnice	13
1.2.2 Specifikace standardů MIL-STD-1553A a MIL-STD-1553B . .	14
1.3 Zařízení ve sběrnici	14
1.3.1 Řadič sběrnice	15
1.3.2 Terminál	15
1.3.3 Sledovač sběrnice	15
1.4 Komunikace mezi periferiemi	15
1.5 Fyzická vrstva	17
1.6 Formáty slov	19
1.6.1 Příkazové slovo	19
1.6.2 Datové slovo	21
1.6.3 Stavové slovo	22
1.7 Řízení přenosu po sběrnici	23
1.7.1 Přenos dat z BC do RT	23
1.7.2 Přenos dat z RT do BC	24
1.7.3 Přenos dat z RT do RT	24
1.7.4 Mode code bez dat	24
1.7.5 Mode code následovaný daty	25
1.7.6 Broadcast	25
2 Standard ECSS-E-ST-50-13C	26
2.1 Základní popis	26
2.2 Služby	26
2.2.1 Služba synchronizace komunikace ¹	26
2.2.2 Služba času ²	29
2.2.3 Služba přenosu dat ³	30

¹Anglický termín: *Communication synchronize service*

²Anglický termín: *time service*

³Zahrnuje dvě služby, označované ve standardu ECSS-E-ST-50-13C jako *Set Data* a *Get Data service*)

2.2.4 Služba toku datových bloků ⁴	32
Praktická část studentské práce	33
3 Návrh architektury terminálu	34
3.1 Základní požadavky	34
3.1.1 Chyby na lince	34
3.1.2 Chyby v terminálu	34
3.2 Fyzická vrstva	36
3.3 Vnitřní struktura terminálu	37
3.3.1 Manchester dekodér	38
3.3.2 Manchester enkodér	41
3.3.3 Hlavní stavový automat	43
3.3.4 Verifikační buffer	45
3.3.5 Sloha terminálu	46
3.3.6 Komunikace s proprietární pamětí	46
3.4 Verifikační prostředí	47
3.4.1 Verifikační testy	47
3.4.2 BFM – Model sběrnice	48
3.4.3 Verifikační sloha	48
3.4.4 Proprietární paměť	49
3.5 Syntéza a implementace	50
3.5.1 Optimalizace s ohledem na plochu	50
3.5.2 Optimalizace s ohledem na rychlosť	50
Závěr	51
Literatura	52
Seznam symbolů a zkratek	53
Seznam příloh	54
A Architektury terminálu v literatuře	55
B Komunikace dle standardu ECSS-E-ST-50-13C	57
B.1 Odesílání dat	57
B.2 Příjem dat	59
B.3 Deskripce datového bloku (DTD)	61
B.4 Potvrzení přenosu datového bloku (DTC)	61

⁴Anglický termín: *Data block transfer service*

B.5	Žádost o příjem (ATR)	62
B.6	Potvrzení příjmu (ATC)	62
C	Obvod HI-15960	63
D	Popis stavů v Hlavním stavovém diagramu	64
E	Verifikační testy	67
E.1	Test č. 1 – Odeslání datových slov	67
E.2	Test č. 2 – Chyba na lince (v příkazovém slově)	68
E.3	Test č. 3 – Chyba na lince (v datovém slově)	68
E.4	Test č. 4 – Chyba na lince (méně datových slov)	69
E.5	Test č. 5 – Mode code požadavek	69
E.6	Test č. 6 – Příjem dat z řadiče v režimu <i>Broadcast</i>	70
E.7	Test č. 7 – Odeslání dat v režimu <i>Broadcast</i>	70
E.8	Test č. 8 – Příjem dat z jiného terminálu v režimu <i>Broadcast</i>	71
E.9	Test č. 9 – <i>Mode code</i> požadavek v režimu <i>Broadcast</i>	71
E.10	Test č. 10 – Chyba v terminálu	72

Seznam obrázků

1.1	Topologie zařízení ve sběrnici	14
1.2	Kódování Manchester – převzato z [1]	16
1.3	Ukázka převodu diferenciálního přenosu na nesymetrický	17
1.4	Možné způsoby připojení terminálu ke sběrnici (upraveno z [1])	18
1.5	Význam jednotlivých bitů v příkazovém slově	19
1.6	Význam jednotlivých bitů v datovém slově	21
1.7	Význam jednotlivých bitů ve stavovém slově	22
1.8	Struktura zprávy při přenosu dat z řadiče sběrnice do terminálu	23
1.9	Struktura zprávy při přenosu dat z terminálu do řadiče sběrnice	24
1.10	Struktura zprávy při přenosu dat z terminálu 2 do terminálu 1	24
1.11	Struktura zprávy při použití režimu <i>Mode code</i> bez přenosu dat	24
1.12	Struktura zprávy při použití režimu Mode code s přenosem dat	25
2.1	Struktura komunikačního rámce při použití služby času	27
2.2	Obsazování volných slotů v komunikačních rámcích	27
2.3	Ukázka možné komunikace mezi subsystémy a periferiemi	28
2.4	Průběh časové synchronizace za využití služby času na úrovni sub-systémů	29
2.5	Průběh komunikace při odesílání dat pomocí služby přenosu dat	31
2.6	Průběh komunikace při příjmu dat pomocí služby přenosu dat	31
3.1	Průběh signálu na obvodu HI-15690 při přijímání dat a odesílání dat	36
3.2	Propojení mezi FPGA a obvodem HI-15690	36
3.3	Návrh architektury terminálu	37
3.4	Architektura Manchester dekodéru	38
3.5	Posloupnost stavů v dekodéru při přijímání příkazového slova	39
3.6	Stavový diagram Manchester dekodéru	40
3.7	Architektura Manchester enkodéru	41
3.8	Stavový diagram Manchester enkodéru	42
3.9	Hlavní stavový diagram	44
3.10	Princip verifikačního bufferu s ukazateli <i>head</i> a <i>tail</i>	45
3.11	Časování u proprietární paměti	46
3.12	Struktura verifikačního prostředí	47
3.13	Struktura proprietární paměti	49
A.1	Architektura terminálu podle publikace [5]	55
A.2	Architektura terminálu podle publikace [11]	56
B.1	Odesílání dat z řadiče do terminálu podle služby datových toků	57
B.2	Průběh odesílání dat z řadiče do terminálu metodou Best Effort v rámci služby datových toků	58

B.3	Průběh odesílání dat z řadiče do terminálu metodou Verified Length v rámci služby datových toků	58
B.4	Příjem dat z terminálu do řadiče podle služby datových toků	59
B.5	Průběh příjmu dat z terminálu do řadiče metodou Best Effort v rámci služby datových toků	60
B.6	Průběh příjmu dat z terminálu do řadiče metodou Verified Length v rámci služby datových toků	60
B.7	Struktura zprávy Deskripce datového bloku	61
B.8	Struktura zprávy Potvrzení přenosu datového bloku	61
B.9	Struktura zprávy Žádost o příjem	62
B.10	Struktura zprávy Potvrzení příjmu	62
C.1	Vnitřní struktura obvodu HI-15960 (převzato z [4])	63

Úvod

Tato bakalářská práce se zabývá studiem sběrnice MIL-STD-1553B a následným návrhem a implementací vzdáleného terminálu, který je schopen podle zmíněného standardu komunikovat. Dále je rozebrán standard ECSS-E-ST-50-13C, který celou sběrnici uvádí do kontextu vesmírných technologií a popisuje, jak mohou být její funkcionality a možnosti využity při správě robustních systémů. Protože sběrnice není příliš rozšířená v České republice, drtivá většina dokumentace je v anglickém jazyce. V rámci zpracování tématu jsou proto mnohé termíny překládány, vždy s odkazem na původní anglický výraz, aby bylo možné dohledat dodatečné informace v originální literatuře.

Výsledný terminál je popsán v jazyce VHDL a podroben testům, které ověřují splnění požadavků, vyplývajících ze standardu MIL-STD-1553B. Implementace je provedena do obvodu FPGA Spartan3 od firmy Xilinx, model XC3S50.

1 Sběrnice MIL-STD-1553B

1.1 Základní popis

Z hlediska rozdelení komunikačních sběrnic je MIL-STD-1553B **sériovou** sběrnicí, používající *master-slave* model pro řízení datového toku. Byl vyvinut Americkou armádou, a i přesto, že existují modernější a bezpečnější sběrnice (Firewire, ATM, FDDI - viz [13]), je v současnosti stále využíván z důvodu obtížné implementace nové technologie do zaběhlých systémů. [12]

Protože standard vznikl v 80. letech 20. století a hlavním požadavkem byla především spolehlivost, nelze jej považovat za příliš bezpečný proti kybernetickým útokům. O této problematice rozsáhle pojednávají publikace [2] a [12].

1.2 Stručná historie

Myšlenka pro vytvoření sběrnice MIL-STD-1553 vznikla po druhé světové válce s rostoucí složitostí a komplexností subsystémů v leteckých technologiích. Stávalo se téměř nemožným, aby každé dva komunikující subsystémy vyžadovaly speciální komunikační kanál. Proto byla v roce 1968 na žádost americké vlády zformována skupina odborníků, kteří dostali za úkol vytvořit komunikační sběrniči, která by umožňovala bezpečnou a multiplexovanou komunikaci mezi jednotlivými periferiemi. [7]

Reakcí bylo v roce 1973 uvedení standardu sběrnice **MIL-STD-1553**. Standard byl v následujících letech dvakrát aktualizován, z čehož vznikly dvě modifikace: MIL-STD-1553A (starší verze – 1975) a MIL-STD-1553B (novější verze – 1978). [7] [5] Verze B se v nezměněné podobě používá dodnes a byla upravována jen v podobě vyhlášek (angl. *notices*), které pouze měnily některé zaběhlé termíny a odstranily omezení na letecké systémy. Poslední vyhláška (číslo 7) vyšla v roce 2008. [5]

V roce 2018 byla uvedena revize MIL-STD-1553C, která zařazuje požadavky na elektromagnetickou kompatibilitu do samostatného dokumentu a pozměňuje některé operace s terminálem. [3]

1.2.1 Použití sběrnice

Historicky první implementace sběrnice (verze A) byly ve stíhacím letounu F-16 a vojenské helikoptéře AH-64A Apache. Ověřování v praxi rychle podnítilo myšlenku standard zrevidovat, protože dostupné funkce nestačily pro požadavky systému. [5] Roku 1981 přijalo standard NATO a vydalo vlastní verzi pod názvem STANAG 3838,

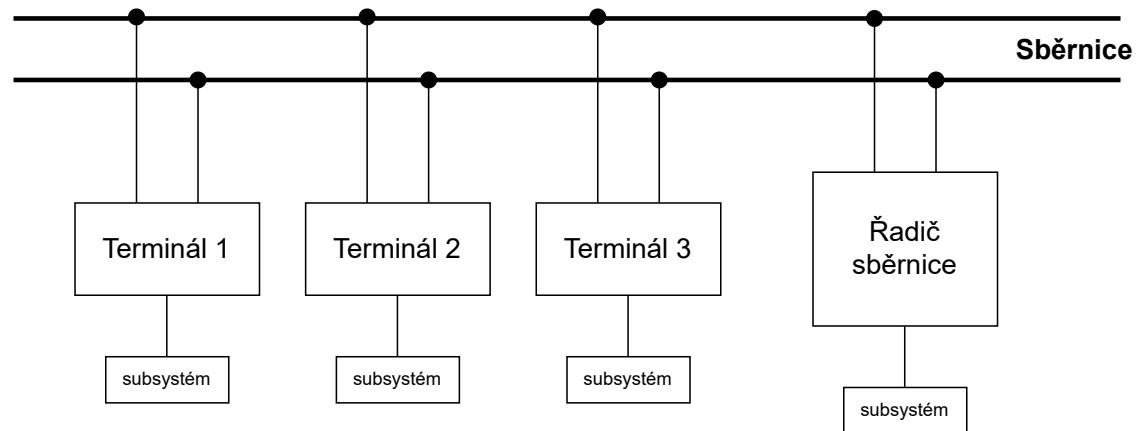
která byla použila v mnoha vojenských systémech (např. Hawk 1T trainer, Lynx helicopter, SAK 57 automatic cannon...).

1.2.2 Specifikace standardů MIL-STD-1553A a MIL-STD-1553B

Novější standard MIL-STD-1553B je skoro ve všech ohledech složitější a rozsáhlejší; umožňuje například použití utilit jako *Mode code* (viz 1.6.1) a *Broadcast mode* (viz 1.7.6), které rozšiřují možnosti komunikace mezi subsystémy a nabízí lepší přizpůsobení architektury podle požadavků konkrétní aplikace za cenu větší komplexnosti. Rovněž definuje některé další požadavky (například elektromagnetickou kompatibilitu, parametry transformátoru ve fyzické vrstvě), které starší standard nechával na návrháři, jiné naopak redefinuje (specifikace vazebního členu (*coupler*), doba odezvy RT) a některé dokonce vynechává (použité konektory ve sběrnici). Konkrétních změn je poměrně velký počet a protože se tato práce zabývá právě novějším standardem, nebudu se na ně dále odkazovat. [1]

1.3 Zařízení ve sběrnici

Protože celá sběrnice je postavena na topologii *master-slave*, mají jednotlivé subsystémy pevně danou hierarchii a rozdělují se na **řadič sběrnice** (angl. *bus controller*, zkr. **BC**), **terminál** (angl. *remote terminal*, zkr. **RT**) a **sledovač sběrnice** (angl. *bus monitor*, zkr. **BM**).



Obr. 1.1: Topologie zařízení ve sběrnici

1.3.1 Řadič sběrnice

Řídící jednotka celé sběrnice (tedy *master*), která jediná může zahájit přenos mezi jednotlivými periferiemi. Kromě řízení přenosu také analyzuje a vede historii chybých zpráv. Ve sběrnici se může v aktivním stavu nacházet pouze jeden *řadič sběrnice*, ale v některých aplikacích lze nalézt tzv. záložní řadič sběrnice (angl. *backup bus controller*), který může sloužit jako náhrada.

1.3.2 Terminál

Představuje periferii pro všechna zařízení, která mezi sebou mají komunikovat po sběrnici (tedy *slaves*). Bez dovolení řadiče sběrnice nemůže sama zahájit přenos. Standard MIL-STD-1553B dovoluje až 31 terminálů, každý má ve sběrnici svou unikátní adresu. Právě touto periferií se tato práce zabývá a její návrh je v části 3.

1.3.3 Sledovač sběrnice

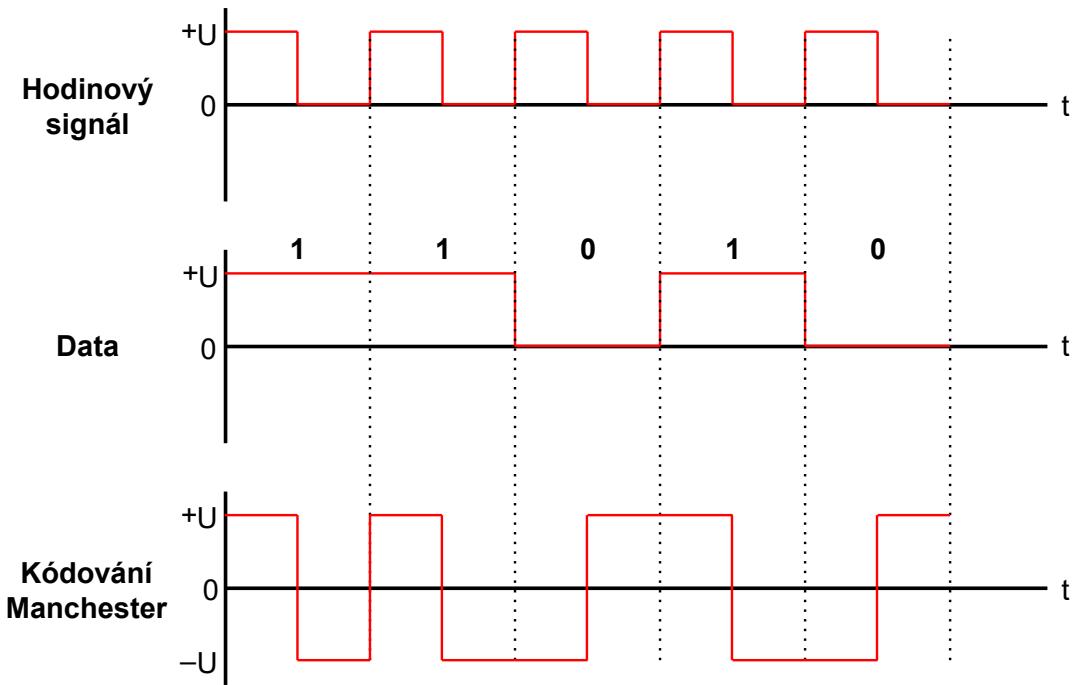
Ve sběrnici se může nacházet ještě tzv. sledovač sběrnice, který pouze monitoruje celý přenos a ukládá data do off-line paměti. Nijak nezasahuje do komunikace, nemůže posílat data do jiných periferií. Používá se pro zpětnou revizi komunikace například v případě poruchy.

1.4 Komunikace mezi periferiemi

Na sběrnici může dojít k několika požadavkům na komunikaci, kde každá má jasné daný průběh. Průběh přenosu dat ve všech případech řídí BC, jednotlivá slova se pak rozdělují na **příkazové slovo** (nese informaci o nadcházející akci, inicializuje přenos), **datové slovo** (obsahuje přenášená data) a **stavové slovo** (nese informaci o stavu zařízení). Jejich podrobný popis je v oddílu 1.6.

Komunikace se neomezuje pouze na přenos dat, řadič sběrnice může například požádat vybraný terminál o synchronizaci časovače, zakázat další komunikaci nebo jej restartovat. Konkrétní příkazy jsou součástí **příkazového slova** v režimu *Mode code* a jsou rozebrány v kapitole 1.6.1.

Všechny byty na sběrnici jsou přenášeny za pomoci tzv. **kódování Manchester**,



Obr. 1.2: Kódování Manchester – převzato z [1]

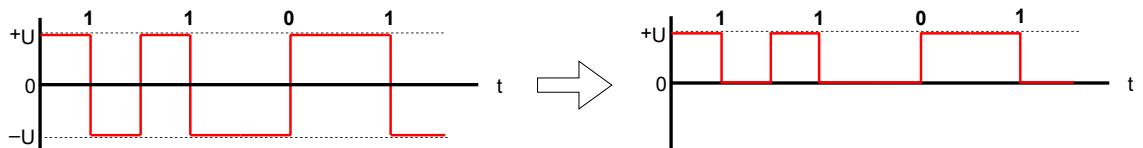
což je speciální druh kódování, který rovněž používá např. *Ethernet* [10] a který umožňuje synchronní přenos bez nutnosti zavádět do sběrnice hodinový signál. Výsledný signál je totiž exkluzivním součtem datového a hodinového signálu, takže signál neustále mění svou napěťovou úroveň i při konstantní logické hodnotě.

Kromě synchronního přenosu kódování umožňuje přenášet signál přes transformátory, čehož pak využívá fyzická vrstva. Nevýhodou je obecně nutnost daný signál dekódovat a také omezení použitelné šířky pásma dané přenosové linky na polovinu (bez kódování by bylo možné přenášet informaci dvojnásobnou rychlostí).

1.5 Fyzická vrstva

Nejnižší vrstva komunikace má ve standardu 1553B striktně definované požadavky. Vychází především ze skutečnosti, že signál je po lince přenášen v tzv. **diferenciální podobě**, což znamená, že log. 1 je reprezentována kladným napětím a log. 0 záporným napětím (oproti zemi). Výhodou je především větší odolnost vůči elektromagnetickému rušení (což je u leteckých systémů nezbytné). [6]

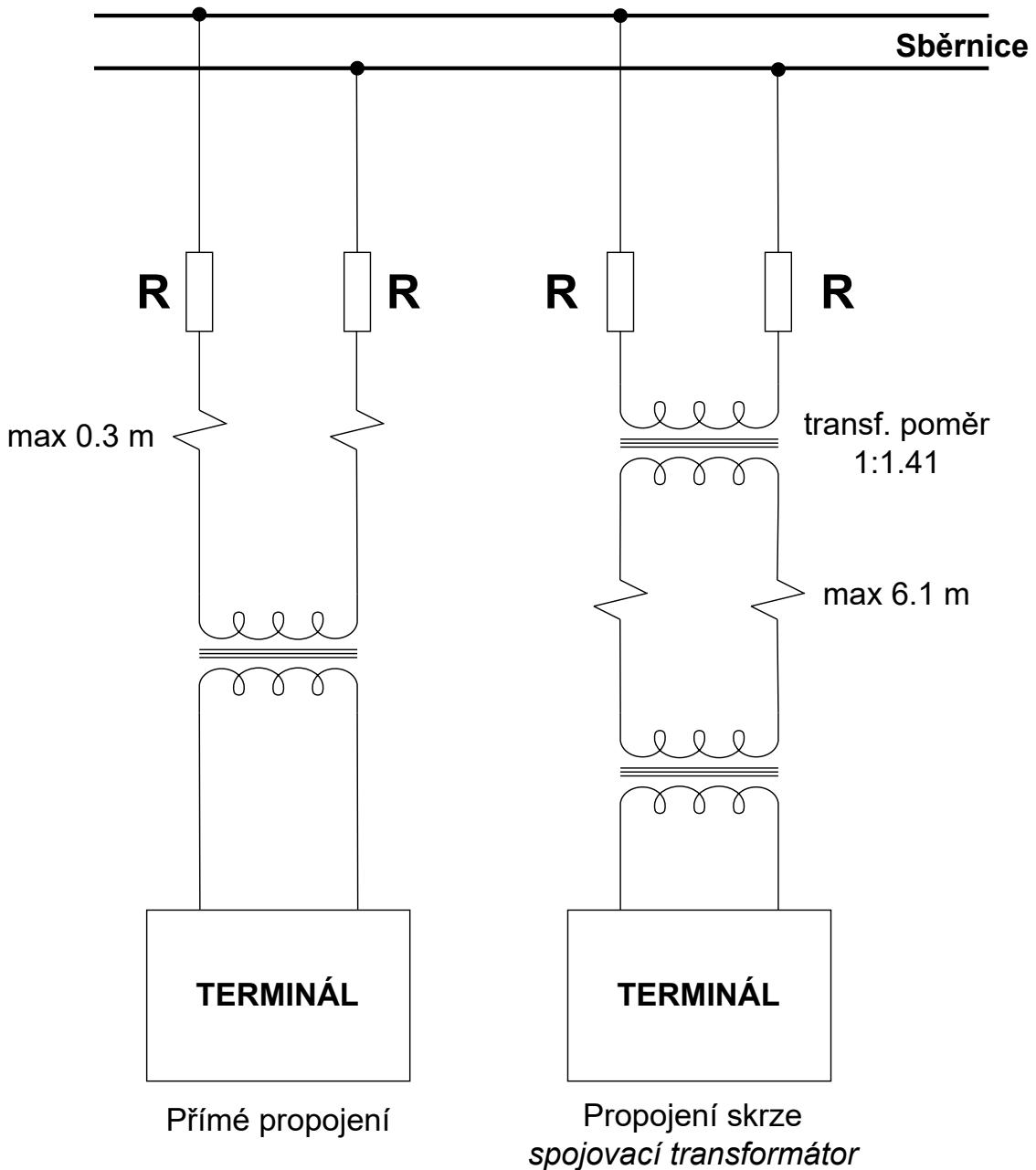
Nicméně periferie, které sběrnice MIL-STD-1553B používá, jsou uzpůsobeny pro signály se společnou a zemí, kde log. 0 je na potenciálu země. Pro převod mezi diferenciálním přenosem a přenosem se společnou zemí se dají použít speciální integrované obvody, které buď mohou být přímo součástí periferie, nebo na externím čipu. Jako příklad externího čipu lze uvést obvody HI-15690 a HI-1565 od firmy Holt Integrated Circuits [4].



Obr. 1.3: Ukázka převodu diferenciálního přenosu na nesymetrický

Kably, které přenášejí signál po lince, by měly být realizovány pomocí **kroucené dvojlinky** a odstíněny proti rušení. Standard přímo definuje, že jejich kapacita nesmí přesahovat 30 pF na stopu (tedy přibližně 90 pF na metr) a jmenovitá impedance musí mít hodnotu od 70 do 85 Ω při frekvenci 1 MHz. Jmenovitý útlum pak nesmí při stejně frekvenci být větší než 5 dB na 100 m. Vodiče, které splňují dané požadavky a jsou přímo vyráběné pro implementaci sběrnice, nabízí firmy jako Digikey nebo Connectivity Power Control.

Mezi přijímačem diferenciálního signálu a samotnou linkou jsou potřeba ještě **oddělovací transformátory**, které slouží nejen pro galvanické oddělení, ale hlavně pro impedanční přizpůsobení. Na lince je požadavek držet koncovou impedanci na hodnotě, kterou mají vodiče, periferie ale mají vysokou vstupní impedanci pro nízký příkon a lepší citlivost. Pokud by byla sběrnice připojena k periferii přímo, docházelo by k odrazu signálu a vzniku stojatých vln na vedení, což by způsobilo rušení a riziko poškození periferií. Standard uvádí možnosti **přímého** (*direct*) spojení skrze jeden transformátor a dva rezistory, případně pak spojení skrze další **spojovací** (*coupling*) transformátor, který snižuje rušení a je požadován v aplikacích, kde je velká vzdálenost mezi sběrnicí a periferií.



Obr. 1.4: Možné způsoby připojení terminálu ke sběrnici (upraveno z [1])

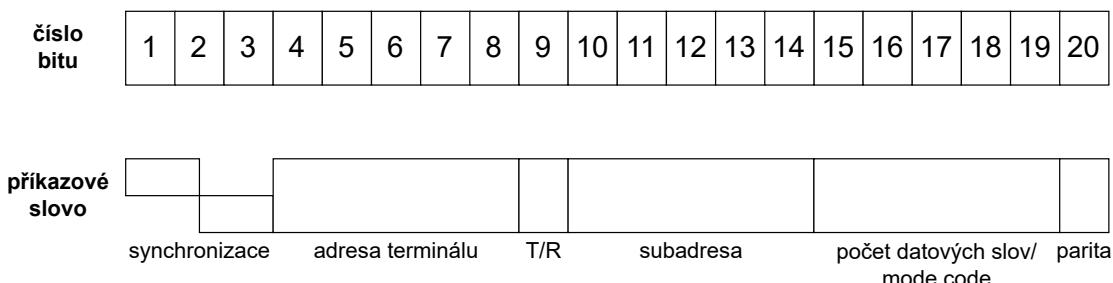
1.6 Formáty slov

Všechny informace jsou po sběrnici přenášeny ve formě zpráv, kde každá zpráva obsahuje slova s délkou 20 bitů. První tři byty vždy zabírá **synchronizační vlna**, která jako jediná není modulována kódováním Manchester. U příkazového a stavového slova začíná synchronizace v log. 1, u datového rámce v log. 0, což je záměrně pro odlišení přenosu dat.

Poslední bit je pak paritní (doporučená je lichá parita). Zbylých 16 bitů každého slova využívá jiným způsobem a některé jsou dokonce ignorovány, protože nemají využití.

1.6.1 Příkazové slovo

Každá komunikace je zahájena **příkazovým slovem**, které terminál informuje o následující akci. Současně může obsahovat hodnotu **Mode code**.



Obr. 1.5: Význam jednotlivých bitů v příkazovém slově

1. **Adresa terminálu** – Zabírá 5 bitů, může tedy obsahovat číslo od 0 do 31. Adresu 11111 standard doporučuje nepoužívat jako unikátní adresu pro jeden terminál, ale pro tzv. *Broadcast message*, která je doručena všem terminálům na sběrnici.. To znamená, že sběrnice může mít až 31 terminálů.
2. **T/R** – Bit určující, zda bude terminál přijímat zprávu (log. 0) nebo vysílat.
3. **Subadresa** – Určuje adresu subsystému daného terminálu. Komplexnější terminály mohou subsystémů obsahovat více, standard jich umožňuje mít až 30. Adresy 00000 a 11111 se pak používají jako signalizace terminálu, že následujících 5 bitů neurčuje délku zprávy, ale jeden z příkazů *Mode code*. První bit se dá optimálně použít pro rozlišení mezi **příkazovým** a **stavovým slovem** (viz 1.6.3), čímž dojde k redukci adres subsystémů na polovinu.
4. **Počet datových slov/mode code** – Při *Mode code* specifikuje daný příkaz. Jinak určuje délku přenášených dat (tedy počet **datových slov**), která může dosahovat hodnoty 31.

Mode code

Funkce **Mode code**¹ přidává řadiči sběrnice širokou paletu možností, jak řídit přenos po sběrnici, od synchronizace časovačů v terminálech až po jejich reset. Všechny možné funkce jsou v tabulce 1.1. Jsou rozděleny jednak podle toho, zda při jejich realizaci dochází k přenosu dat (tj. v dané zprávě je přítomno datové slovo), a také zda je možné je použít pro režim *Broadcast*. Většina jich nenajde u jednoduších aplikací velké využití, proto jsou zde popsány jen ty nejdůležitější z nich. Podrobný popis všech funkcí lze nalézt v publikaci [1].

Tab. 1.1: Výpis funkcí režimu *Mode code*

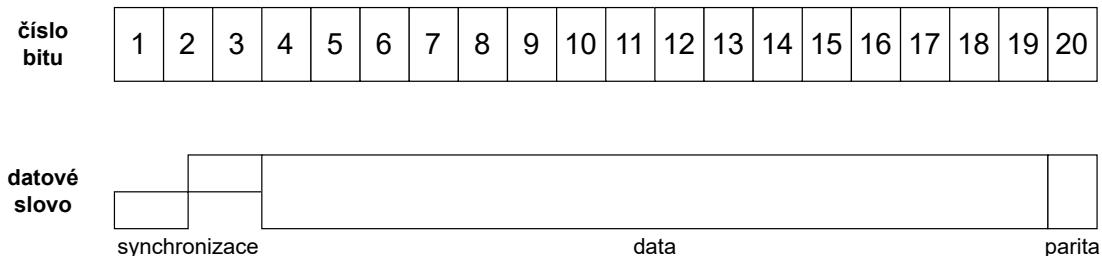
T/R	Mode code bit	Funkce	Přítomnost datového slova	Možnost použití Broadcast
1	00000	Dynamické řízení sběrnice	Ne	Ne
1	00001	Synchronizace	Ne	Ano
1	00010	Odeslání stavového slova	Ne	Ne
1	00011	<i>"Initiate self-test"</i>	Ne	Ano
1	00100	<i>"Transmitter shutdown"</i>	Ne	Ano
1	00101	<i>"Override transmitter shutdown"</i>	Ne	Ano
1	00110	<i>"Inhibit terminal flag bit"</i>	Ne	Ano
1	00111	<i>"Override inhibit terminal flag bit"</i>	Ne	Ano
1	01000	Reset terminálu	Ne	Ano
1	01001	<i>"Reserved"</i>	Ne	-
1	01111	<i>"Reserved"</i>	Ne	-
1	10000	Přenos vektorového slova	Ano	Ne
0	10001	Synchronizace	Ano	Ano
1	10010	<i>"Transmit last command"</i>	Ano	Ne
1	10011	<i>"Transmit bit word"</i>	Ano	Ne
0	10100	<i>"Selected transmitter shutdown"</i>	Ano	Ano
0	10101	<i>"Override sel. transmitter shutdown"</i>	Ano	Ano
1/0	10110	<i>"Reserved"</i>	Ano	-
1/0	11111	<i>"Reserved"</i>	Ano	-

¹Termín je bez překladu

- **Dynamické řízení sběrnice** (angl. *Dynamic bus control*, zkratka *DBC*) – Příkaz, kterým řadič sběrnice žádá terminál o převzetí řízení sběrnice. Terminál (a celá aplikace) k tomu musí být pochopitelně přizpůsoben. Odpověď terminálu je v následovaném stavovém slově (viz 1.6.3).
- **Synchronizace** – Požadavek na synchronizaci terminálu, tj. resetování vnitřního časovače. Požadavek může být následován datovým slovem (při 10001 je následován, při 00001 nikoliv), který může specifikovat další konfiguraci terminálu. O jakou konfiguraci se jedná a jak je bitově reprezentována, zůstává na návrháři. V obou případech je odpověď terminálu stavové slovo.
- **Odeslání stavového slova** (angl. *Transmit status word*) – Příkaz pro zaslání stavového slova, který se týká posledního příkazového slova, jenž terminál obdržel. Hodnota stavového slova je po odeslání v systému ponechána.
- **Reset terminálu** (angl. *Reset remote terminal*) – Okamžitý reset terminálu. Terminál před resetem odešle stavové slovo o momentálním stavu.
- **Odeslání vektorového slova** (angl. *Transmit vector word*) – Způsobí zaslání stavového slova daného terminálu, následovaného datovým slovem, kde je specifikována porucha, kterou terminál má. Jedná se o reakci na obdržení předchozího stavového slova, ve kterém byl Service request bit v log. 1.

1.6.2 Datové slovo

Na lince by se datové slovo bez předchozího **příkazového slova** nemělo nikdy objevit. Terminál bezpečně datové slovo rozezná podle odlišné *synchronizační vlny*.



Obr. 1.6: Význam jednotlivých bitů v datovém slově

1. **Data** – Jedno datové slovo může mít maximálně 16 bitů, které reprezentují přenášenou informaci mezi periferiemi. Právě tyto bity terminál dekóduje a předá systému, který jej obsluhuje.

1.6.3 Stavové slovo

Terminál po skončení komunikace ve většině případů odesílá **stavové slovo**, kde může informovat řadič o vzniklých problémech. **Stavové slovo** není nikdy odesláno, pokud hrozí kolize dat (režim *Broadcast* nebo neočekávaná chyba).



Obr. 1.7: Význam jednotlivých bitů ve stavovém slově

1. **Adresa terminálu** – Adresa koncového terminálu (viz 1.6.1).
2. **Chyba zprávy** – Při log. 1 signalizuje chybný přenos jednoho nebo více datových rámců. Přenos je považován za chybný, jestliže:
 - (a) Nebyla splněna kontrola parity
 - (b) Přenos nezačal synchronizačním impulzem
 - (c) Data nebyla přenášena kódováním Manchester (případně forma byla chybná)
3. **Instrumentace (volitelné)** – Umožňuje odlišení příkazového a stavového slova tím, že u stavového je vždy v log. 0 (u příkazového je pak v log. 1). Je na návrháři, zda této možnosti využije. Pokud ne, je tento bit jednoduše ignorován.
4. **Žádost o obsluhu (volitelné)** – Bit, kterým může terminál signalizovat řadiči sběrnice, že něco není v pořádku. Ten by na to měl reagovat a pomocí *Mode code* (Transmit vector word) zjistit, o co se konkrétně jedná. Použití tohoto bitu je volitelné a není-li využit, měl by být v log. 0.
5. **Rezervováno** – 3 bity, které v současné verzi standardu nemají využití a měly by zůstat v log. 0.
6. **Příjem v režimu Broadcast** – Nabývá log. 1, pokud bylo předchozí slovo odesláno všem terminálům (byl využit režim *Broadcast*)

7. **Zaneprázdnění** (*volitelné*) – Nabývá log. 1, jestliže terminál momentálně není schopen doručit data z/do subsystému. Řadič sběrnice by na to měl adekvátně reagovat a zvážit například další akce. Není-li tato funkce implementována, bit je vždy v log. 0.
8. **Příznak chyby subsystému** (*volitelné*) – Signalizuje (log. 1) možnou chybu na straně subsystému, čímž dává řadiči sběrnice informaci, že data mohou být chybná. Při nevyužití je vždy v log. 0.
9. **Potvrzení DBC** (*volitelné*) – Odpověď terminálu na požadavek řadiče sběrnice o dynamické řízení sběrnice. Je-li v log. 1, terminál požadavek přijímá a stává se novým řadičem (viz 1.6.1).
10. **Příznak chyby terminálu** (*volitelné*) – Signalizuje (log. 1) možnou chybu na straně terminálu. při nevyužití je vždy v log. 0.

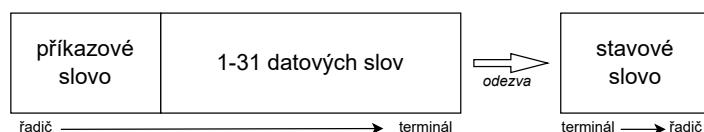
1.7 Řízení přenosu po sběrnici

Komunikace na sběrnici probíhá formou zpráv o přesně určeném formátu a pořadí jednotlivých slov. Ve všech případech ji iniciuje řadič sběrnice, nemůže se tedy stát, že by terminál zahájil odesílání dat sám od sebe. Mezi jednotlivými zprávami je nutné dodržovat **časový interval** (angl. *intermessage gap*), který standard uvádí na $4 \mu s$.

Terminály mají rovněž určenou dobu **dobu odezvy** (angl. *response time*), která je mezi 4 a $12 \mu s$. Řadič sběrnice ji měří od poloviny posledního odeslaného bitu do poloviny prvního přijatého bitu.

1.7.1 Přenos dat z BC do RT

Řadič sběrnice zahájí komunikaci příkazovým slovem, kde specifikuje vybraný terminál a nastaví jej pro příjem dat. Následují datová slova. Po obdržení všech slov terminál pošle stavové slovo s informací o možných chybách.



Obr. 1.8: Struktura zprávy při přenosu dat z řadiče sběrnice do terminálu

1.7.2 Přenos dat z RT do BC

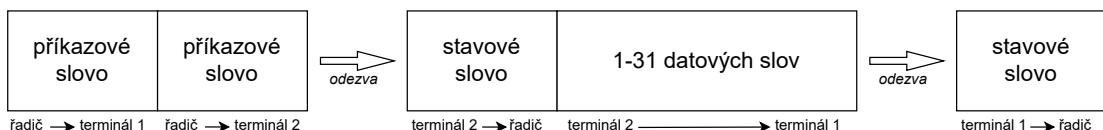
Řadič sběrnice zahájí komunikaci příkazovým slovem, kde specifikuje vybraný terminál a nastaví jej pro odesílání dat. Terminál odpoví stavovým slovem a odpovídajícím počtem datových slov.



Obr. 1.9: Struktura zprávy při přenosu dat z terminálu do řadiče sběrnice

1.7.3 Přenos dat z RT do RT

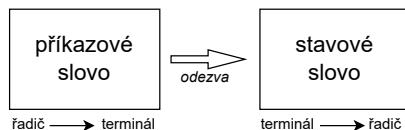
Řadič sběrnice nejprve nastaví příkazovými slovy terminál 1 pro příjem dat a terminál 2 pro odeslání dat. Terminál 2 zašle do řadiče stavové slovo a datovým slovem pak přenese data do terminálu 1. Ten po přijetí všech dat pošle řadiči stavové slovo.



Obr. 1.10: Struktura zprávy při přenosu dat z terminálu 2 do terminálu 1

1.7.4 Mode code bez dat

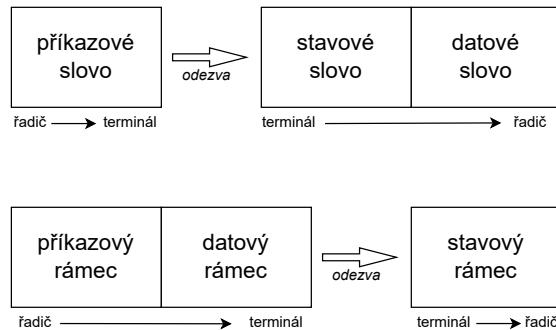
Pokud řadič při použití režimu *Mode code* neoperuje s daty, zasílá do terminálu pouze příkazové slovo, na které terminál odpovídá stavovým slovem.



Obr. 1.11: Struktura zprávy při použití režimu *Mode code* bez přenosu dat

1.7.5 Mode code následovaný daty

Je-li v režimu *Mode code* přítomno datové slovo, může být buď přenášen z řadiče do terminálu, nebo být součástí odpovědi terminálu na daný příkaz.



Obr. 1.12: Struktura zprávy při použití režimu Mode code s přenosem dat

1.7.6 Broadcast

Režim *Broadcast*² lze použít zaslání zprávy všem terminálům na sběrnici. Řadič sběrnice jej aktivuje tak, že jako adresu terminálu v příkazovém slově použije bitovou hodnotu 11111. Komunikace má pak stejný průběh jako v předešlých případech. Při přenosu dat z terminálu 2 do terminálu 1 je příkazové slovo s adresou 11111 zasláno terminálu 1, terminál 2 pak obdrží instrukci, že má poslat data (která obdrží všechny ostatní terminály).

Mode code má *Broadcast* povolen jen v určitých případech (viz tabulka 1.1). Při odesílání dat v režimu *Broadcast* řadič neočekává přijetí **stavového slova** v odpovědi, protože by došlo ke kolizi dat.

²Termín není překládán, protože čeština neobsahuje vhodný ekvivalent

2 Standard ECSS-E-ST-50-13C

2.1 Základní popis

Protože sběrnice našla své využití i ve vesmírných technologiích, byl v roce 2008 vydán standard ECSS-E-ST-50-13C, který do velké míry shrnuje originální standard (a často se na něj odkazuje), nabízí ale rovněž praktické tipy pro návrh periferií, konkretizaci fyzické vrstvy s ohledem na moderní technologie a především pak přichází s tzv. **službami** (angl. *services*), kterými nabízí modernější řízení sběrnice. [8] Zároveň se zmíněný standard daleko více věnuje i subsystémům, které obsluhují periferie sběrnice, např. pro ně definuje názvy příkazů, které periferiím posílají, a obecně popisuje komunikaci na vyšší komunikační vrstvě.

2.2 Služby

Implementace a použití služeb nevyžaduje žádné změny v hardwarovém návrhu, z pohledu komunikace se jedná spíš o doporučené použití utilit, které sběrnice již obsahuje. Je nutno dodat, že služby nejsou nezávislé na sobě a **služba synchronizace komunikace** je potřebná pro všechny ostatní.

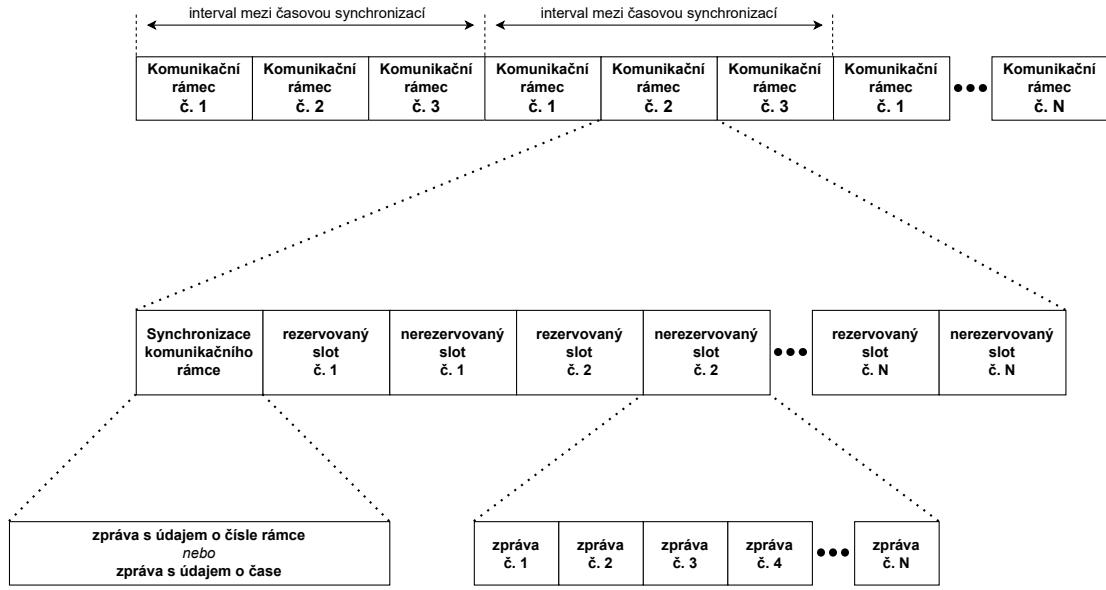
2.2.1 Služba synchronizace komunikace¹

Rozděluje komunikaci na tzv. **komunikační rámce**, kde u každého rámce je předem určeno, jakou ponese informaci. Na začátku každého rámce je použito příkazové slovo v režimu *Mode code* (viz 1.6.1) s datovým slovem, který specifikuje číslo daného komunikačního rámce. Následuje série **rezervovaných** (angl. *populated*) a **nerezervovaných** (angl. *unpopulated*) slotů (angl. *content*), které mohou obsahovat zprávy. Rezervované sloty jsou vyhrazeny pro závažné okolnosti, které musí být ošetřeny co nejdříve. Nerezervované pak může řadič sběrnice použít pro zahájení komunikace (která není "závažná"). Pokud chce tedy odeslat více zpráv za sebou, musí pokaždé počkat, až bude volný nerezervovaný slot.

Na obrázku 2.1 je odkryta struktura komunikačních rámců. Pokud je zároveň použita **služba času**, každý první komunikační rámec neobsahuje číslo rámce, ale údaje k synchronizaci času.

Počet rámců, slotů a zpráv v příslušných blocích komunikace není pevně stanoven a je určen konkrétní aplikací.

¹Anglický termín: *Communication synchronize service*

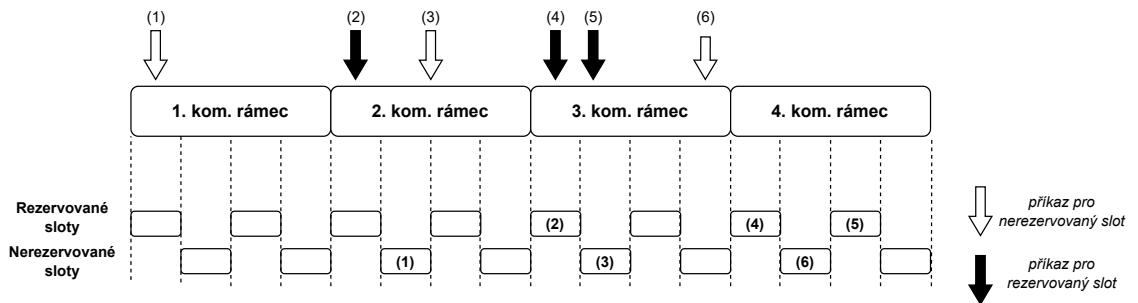


Obr. 2.1: Struktura komunikačního rámce při použití služby času

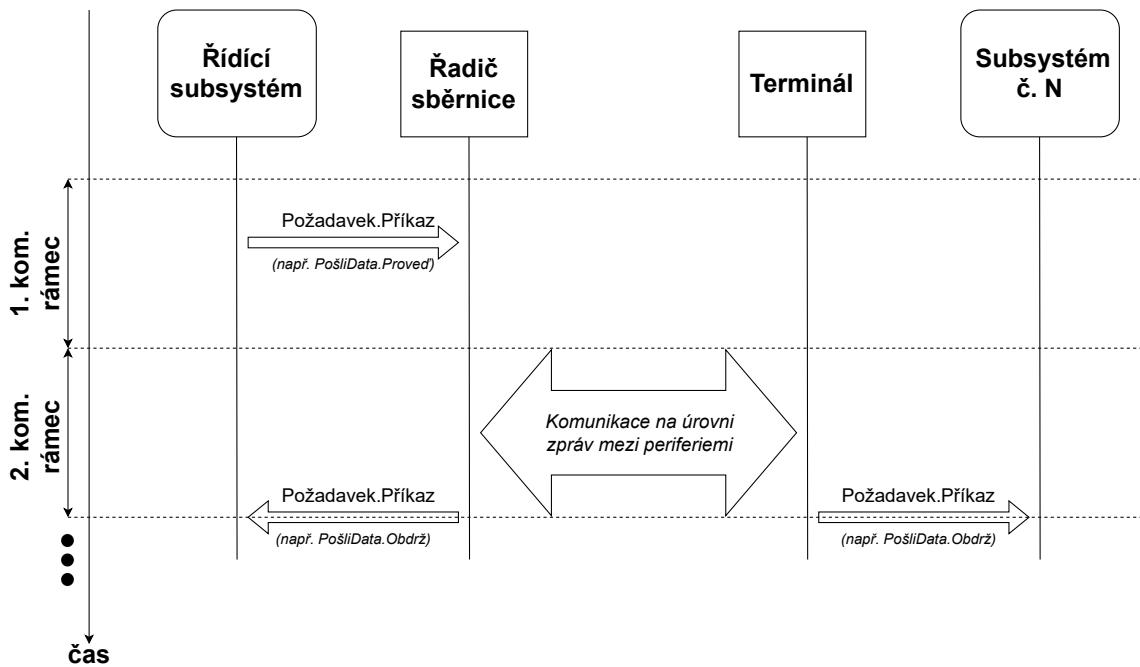
Pravidla obsazování slotů zprávami

Obsazování slotů ovládá řídící substitém (řadič sběrnice). Jejich provedení je vždy až v následujícím komunikačním rámcu, ve kterém jsou zprávy odeslány v příslušných slotech.

Komunikace mezi substitémem a periferií probíhá za pomocí **příkazů** "Proved" (angl. *Submit*) a "Obdrž" (angl. *Deliver*) a **požadavkem**, který je unikátní pro každou službu.



Obr. 2.2: Obsazování volných slotů v komunikačních rámcích



Obr. 2.3: Ukázka možné komunikace mezi subsystémy a periferiemi

"Proved" signalizuje subsystému, aby v dalším komunikačním rámci odeslal zprávu v příslušných slotech. Konkrétní podoba zprávy (tj. jaké sloty obsadí a jak bude dlouhá) je určena konkrétním požadavkem.

Na obrázku 2.3 je vzorová struktura komunikace na úrovni subsystémů. Subsystém vznese **Požadavek** a konkrétní **příkaz**, řadič jej zaznamená a čeká, až bude volný komunikační rámec. Zatímco komunikace mezi periferiemi je řízena komunikačními rámcii, subsystém může periferii kontaktovat kdykoliv (a naopak).

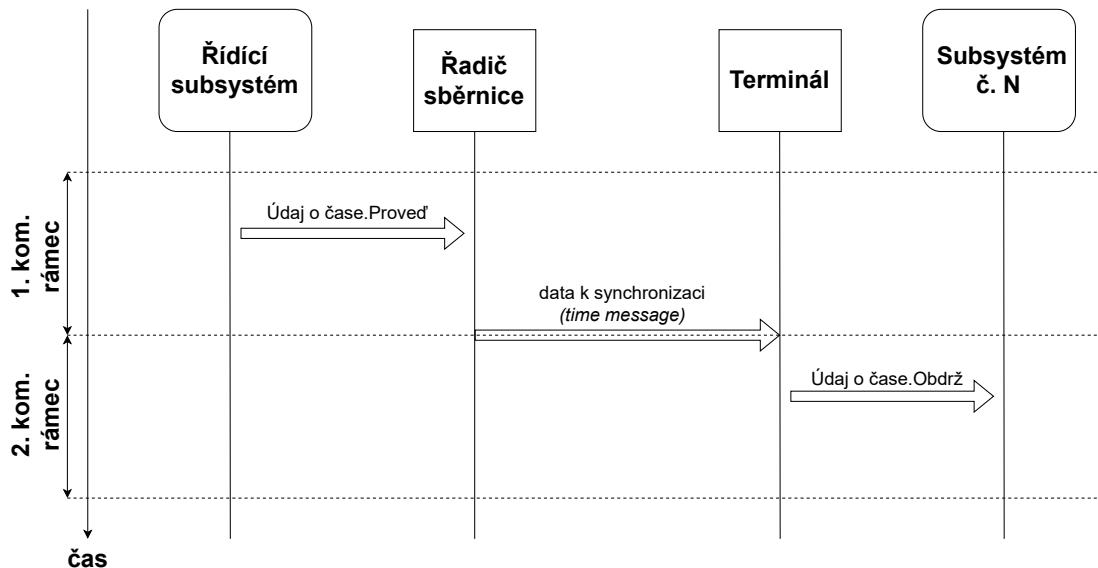
2.2.2 Služba času²

Je-li použita, řadič sběrnice periodicky každých několik komunikačních rámciů (počet závisí na návrhu) posílá na začátku rámce v režimu *Broadcast* požadavek k synchronizaci vnitřního časovače (viz 1.6.1 – Synchronizace).

Samotná synchronizace může probíhat dvěma způsoby:

1. Řadič vyšle zprávu, která obsahuje data k synchronizaci. (tzv. *Time message*) Terminál zprávu přijímá a rovnou s daty operuje (tj. považuje daný čas za validní). Výhodou je jednoduchost, nevýhodou pak nepřesnost, která roste s délkou údaje o čase.
2. Řadič vyšle zprávu, která obsahuje data k synchronizaci. (*Time message*) Terminál zprávu přijímá, ale nepracuje s daty, dokud nedostane další povel (tzv. *Time synchronization message*). Ten je obvykle vysílán po oddělené lince. Výhodou je přesnější synchronizace (a odstranění nepřesnosti u delších údajů o čase), nevýhodou komplexnost a vyšší nároky na linku.

Na úrovni subsystémů je pak komunikace s periferiemi řízena **požadavky údaj o čase** (angl. *TimeData*) a **synchronizace času** (angl. *TimeSynchronise*).



Obr. 2.4: Průběh časové synchronizace za využití **služby času** na úrovni subsystémů

²Anglický termín: *time service*

2.2.3 Služba přenosu dat ³

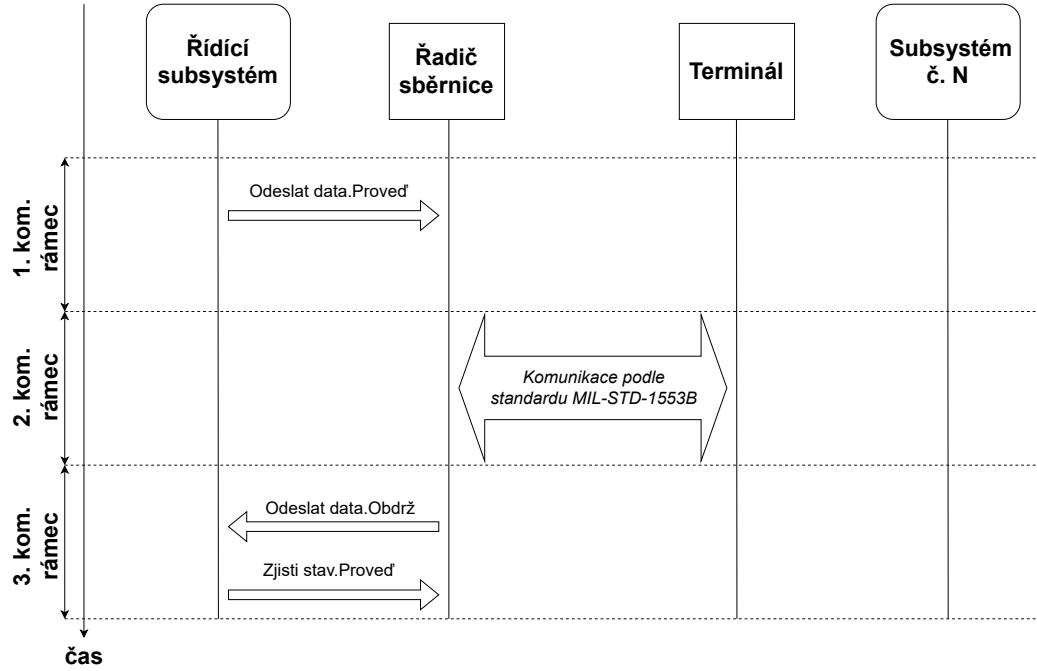
Představuje jednodušší variantu řízení komunikace na sběrnici.

Pro zahájení přenosu dat používá subsystém *požadavky odeslat data* (angl. *Send-Data*) a **přijmi data** (angl. *ReceiveData*).

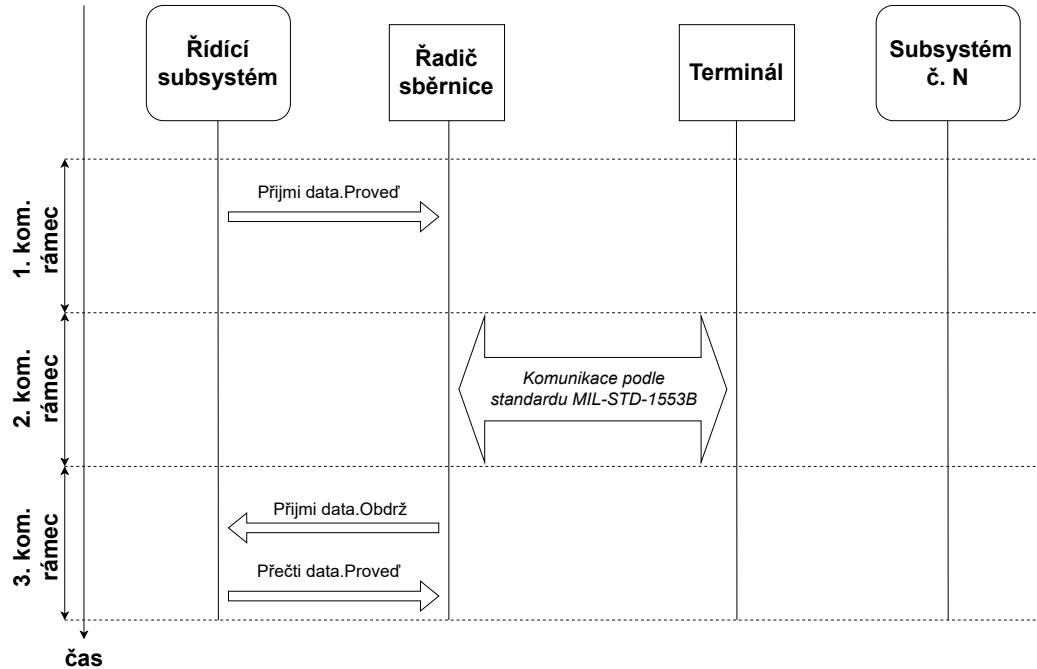
Po přenosu pak může v případě odesílání dat za pomoci *požadavku Zjisti stav* vyžádat od terminálu informace o přenosu (a zjistit například možné chyby). V případě přijímání dat si řídící subsystém vyžádá data od řadiče *požadavkem Přečti data* (angl. *ReadData*).

Průběh provedení služby je na obrázcích 2.5 a 2.6. Komunikace podle standardu MIL-STD-1553B probíhá v komunikačních rámcích. Podle toho, zda přenos využívá rezervované nebo nerezervované sloty rozděluje standard službu na **typ 1** a **typ 2**. Výhodou této služby je jednoduchost a nízké požadavky na terminál a řadič sběrnice, nevýhodou pak skutečnost, že délka zprávy je pevně daná při návrhu hardwaru a při potřebě přenášet větší množství dat je potřeba **požadavek** opakovat.

³Zahrnuje dvě služby, označované ve standardu ECSS-E-ST-50-13C jako *Set Data* a *Get Data service*)



Obr. 2.5: Průběh komunikace při odesílání dat pomocí služby přenosu dat



Obr. 2.6: Průběh komunikace při příjmu dat pomocí služby přenosu dat

2.2.4 Služba toku datových bloků⁴

U komplexnějších systémů, kde je vyžadována vysoká spolehlivost, organizace a možnost přenášet velké množství dat, najde uplatnění *služba toku datových bloků*. Ta podporuje rozdílnou délku dat, *handshake* (potvrzení zprávy o přijetí) a více možností, jak komunikaci přizpůsobit konkrétní potřebě.

Požadavky pro řízení komunikace se stejně jako u **služby přenosu dat** nazývají **Odeslat data** a **Přijmi Data**. Standard požaduje, aby před každým zahájením komunikace došlo k resetování periferie, k čemuž slouží požadavek **Reset**.

Velikost jednoho datového bloku závisí na způsobu využití subadres, které jsou součástí **Deskripce datového bloku**.

1. **adresování typu flat** – 2 byty z prostoru pro subadresu jsou použity pro rozlišení různých částí datového bloku. Délka zprávy pak nemůže být delší než 1024 bajtů.
2. **adresování typu deep** – Celý datový blok je přenášen jednou subadresou, takže zpráva může dosahovat délky 4096 bajtů.

Služba umožňuje použít dva způsoby přenosu, tzv. *Quality of service* (zkr. *QoS*)⁵

1. **Best effort** – Datové toky jsou přenášeny za sebou (v souladu s komunikačními rámci), aniž by periferie potvrzovala přijetí dat.
2. **Verified Length** – Za každým datovým blokem následuje **Potvrzení přenosu datového bloku**.

Odesílání dat

Přenos dat z řadiče do terminálu probíhá pomocí následujících zpráv:

- **Přenos datového bloku** (zkr. DDB) – (angl. *Distribution Data Block*) Obsahuje pouze přenášená data, která mohou mít velikost od 1 do 1024 (nebo 4096 v případě adresování **deep**).
- **Deskripce datového bloku** (zkr. DTD) – (angl. *Distribution Transfer Descriptor*) Nese informaci o následujícím přenášeném bloku a zároveň jej lze použít pro reset (při zahájení komunikace). Je více rozebrán v příloze B.3.
- **Potvrzení přenosu datového bloku** (zkr. DTC) – (angl. *Distribution Transfer Confirmation*) Odeslán terminálem po každém přijetí datového bloku (pokud je jako *QoS* použit *Verified Length*) (viz B.4).

⁴Anglický termín: *Data block transfer service*

⁵Protože *Quality of Service* je v telekomunikaci zaběhlý pojem, je termín ponechán bez překladu

Komunikace mezi subsystémy je iniciována požadavkem Reset, který v dalším komunikačním rámci následuje **Odeslat data**. Reset způsobí odeslání DTD s příkazem k resetování terminálu, požadavek Odeslat data invokuje přenos DTD a DDB. Na obdržení a zpracování dat má terminál jeden celý komunikační rámec. Pokud je *Quality of service* typu *Verified Length*, další rámec odešle terminál řadiči DTC a až v následujícím je připraven přijmout další datový blok; v případě *Best effort* může místo odesílání DTC rovnou přijmout datový blok.

Přijímání dat

Přenos dat z terminálu do řadiče probíhá pomocí následujících zpráv:

- **Příjem datového bloku** (zkr. ADB) – (angl. *Acquisition Data Block*) Obsahuje pouze přenášená data, která mohou mít velikost od 1 do 1024 (nebo 4096 v případě adresování **deep**).
- **Žádost o příjem** (zkr. ATR) – (angl. *Acquisition Transfer Request*) Zpráva, kterou terminál zahajuje přenos signalizuje řadiči, že má data k odeslání (řadič pravidelně kontroluje každý terminál, zda nechce zahájit přenos). Obsahuje velikost dat a další informace o tom, jak má komunikace probíhat. (viz B.5)
- **Potvrzení příjmu** (zkr. ATC) – (angl. *Acquisition Transfer confirmation*) Odeslána řadičem poté, co obdrží všechna data z terminálu. Terminál nemůže poslat další zprávu, dokud neobdrží ATC – forma *handshake*. (viz B.6)

Protože terminál nemůže sám od sebe zahájit přenos dat, řadič v každém komunikačním rámci kontroluje, zda má určený terminál data k odeslání. Pokud ano, odešle terminál řadiči zprávu **Žádost o příjem** a ve druhém následujícím komunikačním rámci je přenos realizován (zpráva **Příjem datového bloku**) a zakončen zprávou **Potvrzení příjmu**. Při **Best effort** je zpráva s daty přijata každý 3. komunikační rámec, u **Verified length** každý pátý.

Přijímání a odesílání dat je ilustracemi popsáno v příloze B.1.

3 Návrh architektury terminálu

3.1 Základní požadavky

Návrh architektury terminálu podle standardu MIL-STD-1553B je závislý na požadavcích, které jsou kladený na aplikaci. V rámci této práce je požadavkem především:

- Správné dekódování signálu
- Zpracování přijatých dat a jejich předání subsystému
- Detekce a určení chyb, které nastaly při přenosu
- Odeslání dat do sběrnice podle pravidel komunikace
- Možnost použití *Mode code* režimu – vystavení hodnoty na výstup a provedení pro **Synchronizaci a Odeslání stavového slova** (viz 1.6.1)

Kompletní seznam požadavků, ke kterým je vztažena verifikace a následné testy, jsou v tabulce 3.1. Implementace architektury bude provedena do FPGA obvodu Spartan3 od firmy Xilinx. Jako **pracovní frekvenci volím 32 MHz**.

Na lince může dojít hned k několika chybám, které terminál musí umět vyhodnotit a zpracovat podle standardu MIL-STD-1553B. Chyby mohou být rozděleny na **chyby na lince** a **chyby v terminálu**.

3.1.1 Chyby na lince

Jedná se o chyby, které terminál vyhodnotí již při přijetí dat. Může jít o:

- Chybnou paritu v datovém či příkazovém slově
- Neznámé kódování, resp. nerozpoznatelnou logickou hodnotu.
- Neočekávateľný průběh komunikace (např. předčasné ukončení)

Společným znakem je, že na každou z těchto chyb reaguje terminál přiřazením log. 1 do bitu **Chyba zprávy ve Stavovém slově** (viz 1.6.3).

Stavové slovo je odesláno v případě, že komunikace jinak proběhla v pořádku a pouze obsahovala chyby v datech (např. vadná parita). Pokud není jasné, zda komunikace probíhá tak, jak terminál předpokládá, stavové slovo odesláno není (aby nedošlo ke kolizi dat).

3.1.2 Chyby v terminálu

Pokud byla zpráva přijata bez komplikací a k chybě došlo až při jejím zpracování, jde o **chybu v terminálu** a ve **Stavovém slově** je pro tuto signalizaci určen bit **Příznak chyby terminálu**.

V našem případě může jít jen o neplatný přístup do **verifikačního bufferu** nebo do subsystému. Protože po zpracování řadič sběrnice očekává stavové slovo, je při výskytu této chyby odesláno vždy.

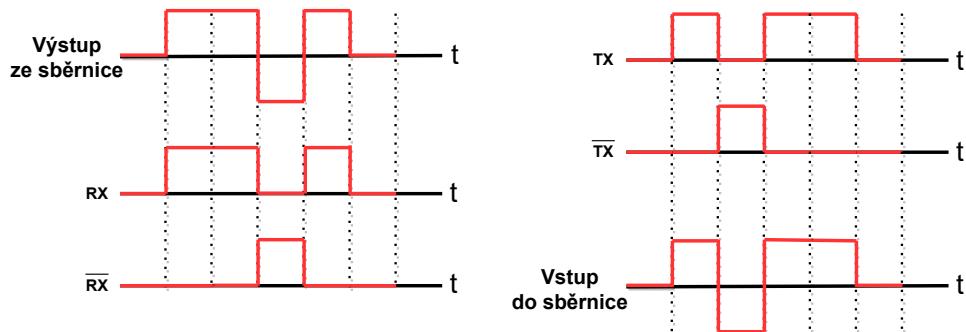
Tab. 3.1: Verifikační matici terminálu

Číslo pož.	Požadavek	Způsob ověření
1	Frekvence FPGA: 32 MHz	viz 3.1
2	Frekvence linky: 1 MHz	viz 3.3
3	Parita zpráv: lichá	simulace
4	Cílová technologie FPGA: Spartan3	viz 3.5
5	Odolnost proti metastabilitě	viz 3.3
6	Rozpoznání synchronizační vlny a správné dekódování signálu Manchester	simulace
7	Odesílání dat podle standardu MIL-STD-1553B (synchronizační vlna + manchester kódování)	simulace
8	Příjem dat od řadiče a uložení do subsstémů dle standardu MIL-STD-1553B	simulace
9	Příjem požadavku na odeslání dat ze subssytému a provedení dle standardu MIL-STD-1553B	simulace
10	Rozpoznání chyby ve zprávě a adekvátní reakce (stavové slovo)	simulace
11	Rozpoznání chybné zprávy (požadavku) a adekvátní reakce (stavové slovo)	simulace
12	Příjem zprávy v režimu <i>Broadcast</i>	simulace
13	Příjem požadavku v režimu <i>Mode code</i> (provedení a vystavení na výstup)	simulace
14	Příjem požadavku pro odeslání dat v režimu <i>Broadcast</i>	simulace
15	Příjem požadavku <i>Mode code</i> v režimu <i>Broadcast</i>	simulace
16	Reakce na chybu v terminálu	simulace

3.2 Fyzická vrstva

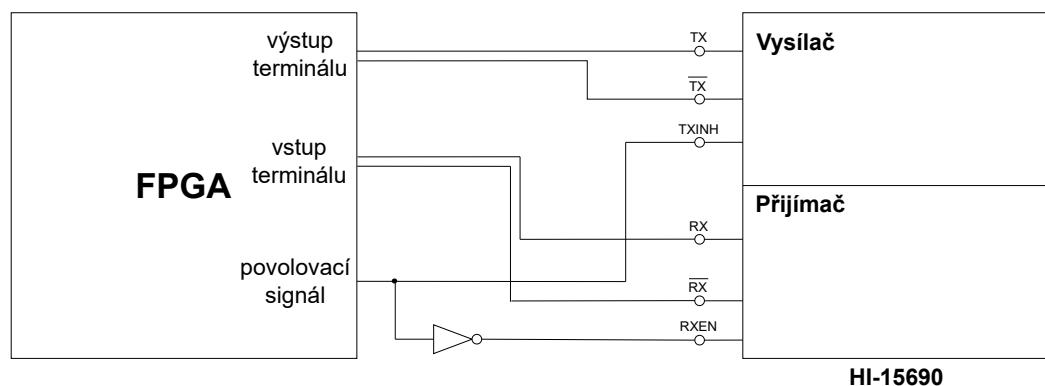
Mezi sběrnicí a obvodem FPGA je přítomen pomocný obvod, který zpracovává symetrický signál a převádí jej na nesymetrický. Pro svou aplikaci jsem zvolil obvod HI-15690 od firmy Holt Integrated circuits. Vnitřní struktura obvodu je v příloze C. Když je signál ze sběrnice v kladné polaritě, vývod RX je v log. 1; při záporné polaritě je naopak \overline{RX} v log. 1. Je-li sběrnice na potenciálu země (neprobíhá přenos), oba vývody jsou v log. 0. Při příjmu dat je RXEN v log. 1.

Pro vysílání do sběrnice používá terminál svorky TX (pro kladnou polaritu) a \overline{TX} (pro zápornou polaritu). Při vysílání je TXINH v log. 1.



Obr. 3.1: Průběh signálu na obvodu HI-15690 při přijímání dat a odesílání dat

Na obrázku 3.2 je naznačeno, jak může vypadat propojení mezi obvodem FPGA a pomocným obvodem HI-15690. Výstup terminálu je připojen přímo na vývod TX a s bitovou negací na vývod \overline{TX} , díky čemuž je zapotřebí jediný výstup. Vstup do terminálu už tak jednoduše ošetřit nejde, obvod FPGA musí zpracovávat oba vývody RX a \overline{RX} . Povolovací signál je výstupem stavového automatu, který určuje, zda terminál vysílá (log. 1) nebo přijímá data (log. 0).



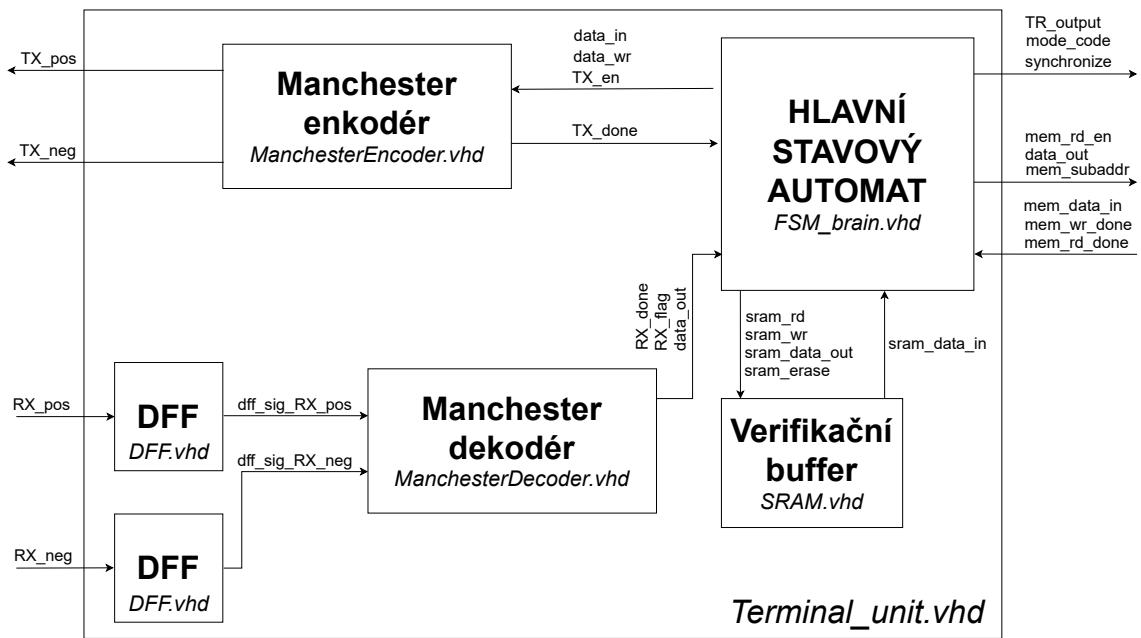
Obr. 3.2: Propojení mezi FPGA a obvodem HI-15690

3.3 Vnitřní struktura terminálu

Standard MIL-STD-1553B nezmiňuje žádnou doporučenou architekturu terminálu (ani jiných periferií), provedení je tedy čistě na návrháři. Některé publikace, které se sběrnicí zabývají, obsahují hrubý náčrt architektury; některé z nich jsou uvedeny v příloze A.

V rámci bakalářské práce jsem pak navrhl vlastní architekturu (viz obrázek 3.3.1). Architektura se skládá ze čtyř hlavních komponent, kde každá má svoji specifickou funkci a dají se použít nezávisle na sobě. Data ze sběrnice přijímá a vyhodnocuje **Manchester dekodér**, které následně posílá do **Hlavního stavového automatu**. Ten podle přijatých bitů rozhoduje o řízení celého terminálu – podle potřeby data ukládá do **verifikačního bufferu** a je-li k tomu vyzván **příkazovým slovem**, odesílá zprávy po sběrnici za pomoci **Manchester enkodéru**. Terminál je schopen komunikovat se subsystémem skrze jednoduchou proprietární sběrnici.

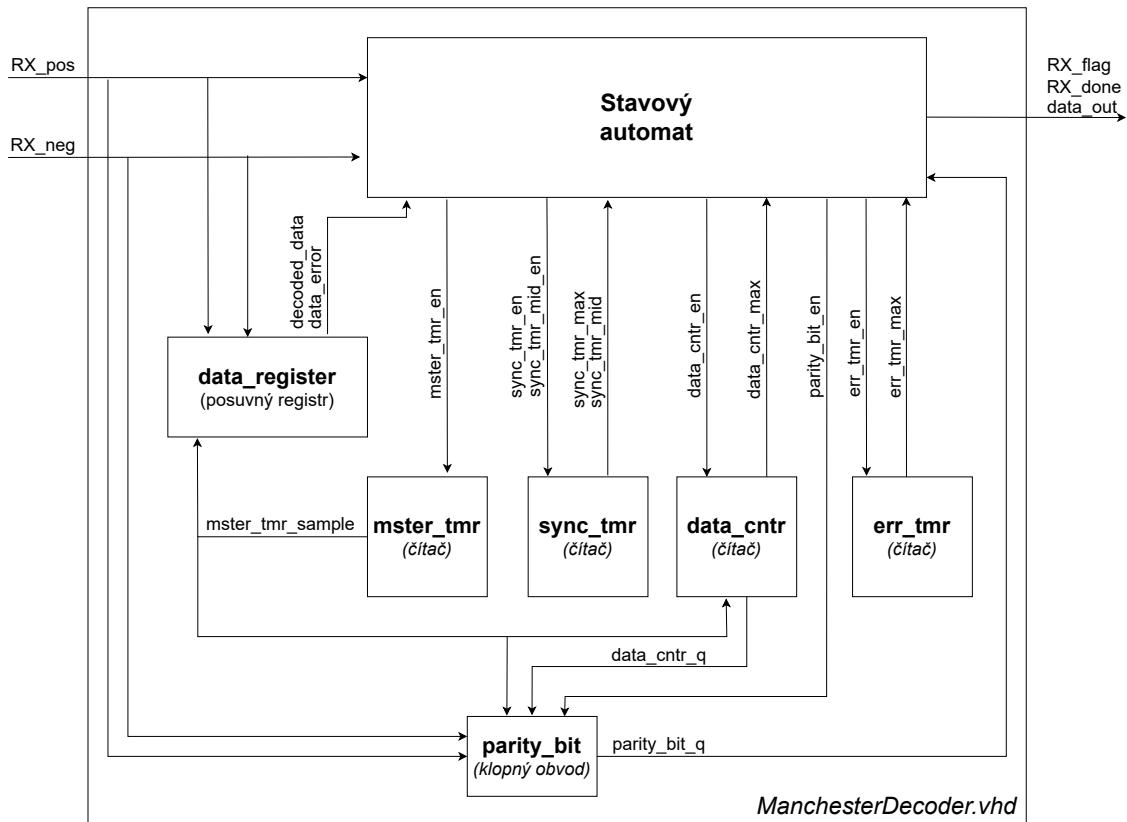
Vstupní signály jsou připojeny na dva klopné obvody typu D (**DFF**), které zabraňují metastabilitě.



Obr. 3.3: Návrh architektury terminálu

3.3.1 Manchester dekodér

Na dekodér je přímo připojen vstupní signál ze sběrnice. Jeho hlavním úkolem je data přijmout, dekódovat, vyhodnotit případné chyby a informace poslat do Hlavního stavového automatu. Samotný dekodér rovněž obsahuje stavový automat, který má 8 stavů a slouží hlavně pro rozpoznání *synchronizační vlny*, kterou musí podle standardu MIL-STD-1553B začínat každé slovo.



Obr. 3.4: Architektura Manchester dekodéru

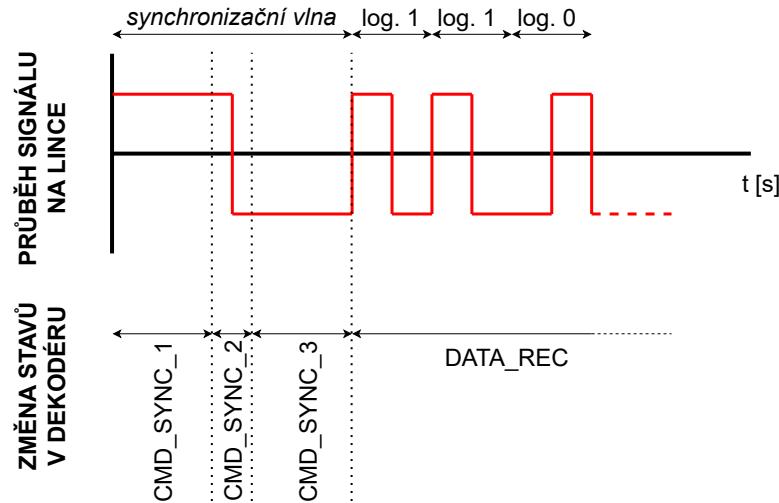
Při vzorkování signálu jsou použity tři čítače a jeden datový registr.

K samotnému dekódování logické hodnoty slouží *mster_tmr*, který signalizuje přetečení vždy ve čtvrtině periody linky, což je místo, kde se v manchester kódování nachází dekódovaná logická hodnota. Do datového registru (*decoded_data*) je při každém takovém přetečení uložena logická hodnota – pokud ji nelze přesně určit, je *data_error* v log. 1, což zastaví vzorkování a vyšle do Hlavního stavového automatu zprávu o chybě.

Při každém uložení hodnoty do registru je inkrementován čítač *data_cntr*, který přeteče při dosažení hodnoty 17, což je délka zprávy. Tím byla přijata celá zpráva

a podle parity je vyhodnocena její správnost. Výpočet parity je sekvenční za pomocí klopného obvodu (*parity_bit*), který průběžně provádí exkluzivní součin právě přijaté hodnoty s posledně přijatou hodnotou.

Poslední čítač je (*err_tmr*), který nezávisle na ostatních kontroluje, zda vstupní signál jedné polarity nepřekračuje 1.5násobek periody, což by znamenalo chybu na lince. Pokud za úspěšně přijatým slovem ihned následuje další slovo, dekodér rovnou měří synchronizaci, v opačném případě přechází do výchozího stavu.

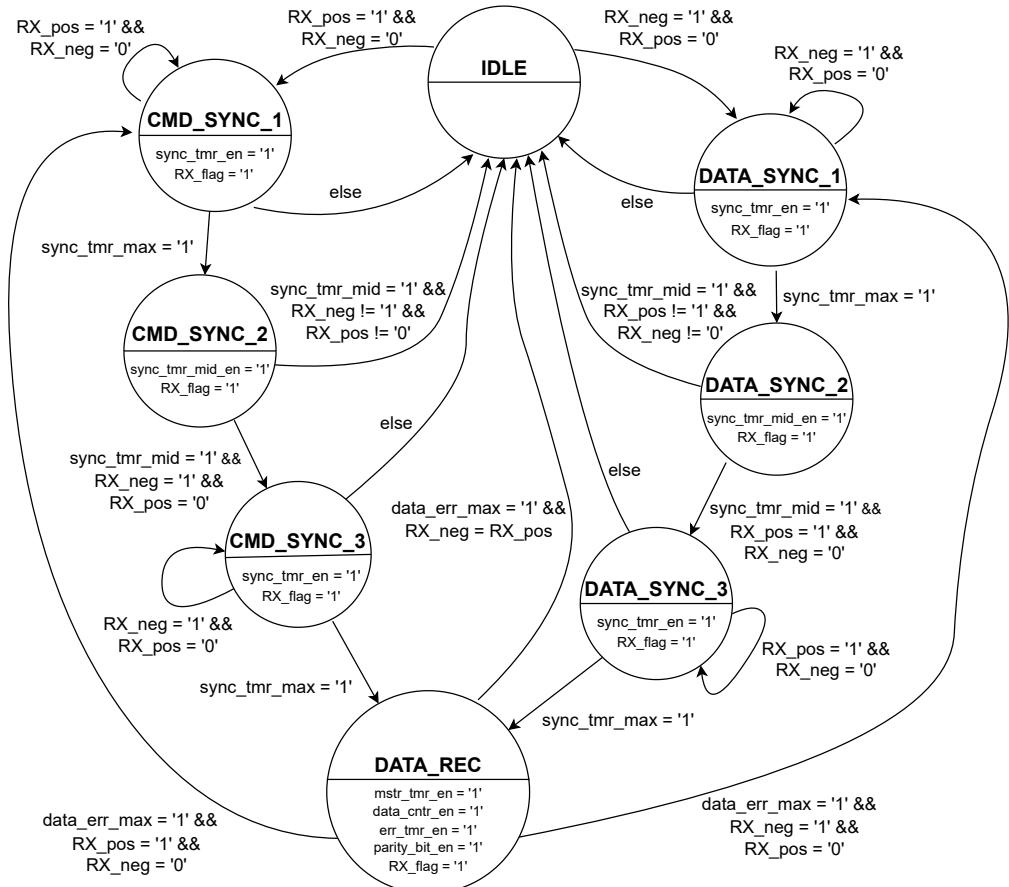


Obr. 3.5: Posloupnost stavů v dekodéru při přijímání příkazového slova

Stavový diagram je uveden na 3.3.1. Z výchozího stavu (**IDLE**) se automat může dostat do dvou možných stavů (**CMD_SYNC_1** nebo **DATA_SYNC_1**), podle polarity vstupního signálu. Protože *synchronizační vlna* musí zůstat v jedné polaritě po dobu 1.5krát perioda linky (tedy $1.5 \mu\text{s}$), je v těchto dvou stavech spuštěn čítač *sync_tmr*, který přeteče v hodnotě $1.375 \mu\text{s}$ – záměrně před koncem očekávatelné délky první půlvlny synchroničního pulzu, kvůli případnému nepřesnému časování. Pokud čítač stihne přetéct, stavový automat přechází do druhého stavu (**CMD_SYNC_2** nebo **DATA_SYNC_2**), v opačném případě (při ztrátě signálu či předčasně změně polarity) přechází do výchozího stavu.

Ve druhém stavu setrvává po dobu čtvrt periody ($0.25 \mu\text{s}$), poté zkонтroluje, zda došlo na lince ke změně polarity. Pokud ano, přechází do třetího stavu (**CMD_SYNC_3** nebo **DATA_SYNC_3**), jinak přechází do výchozího stavu. V posledním stavu setrvá znova $1.375 \mu\text{s}$ (zbytek *synchronizační vlny*), pak postupuje do posledního stavu (**DATA_REC**), ve kterém jsou vzorkována data.

Do hlavního stavového automatu dekodér posílá dekódované paralelní slovo (bez



Obr. 3.6: Stavový diagram Manchester dekodéru

parity), informaci o validitě slova (ve formě RX_done – odesláno při každém přechodu ze stavu **DATA_REC** do stavu **IDLE**) a pokud přijímá data (tj. nachází se v jiném stavu než výchozím), je tato skutečnost signalizována signálem RX_flag .

Tab. 3.2: Interpretace signálu RX_done

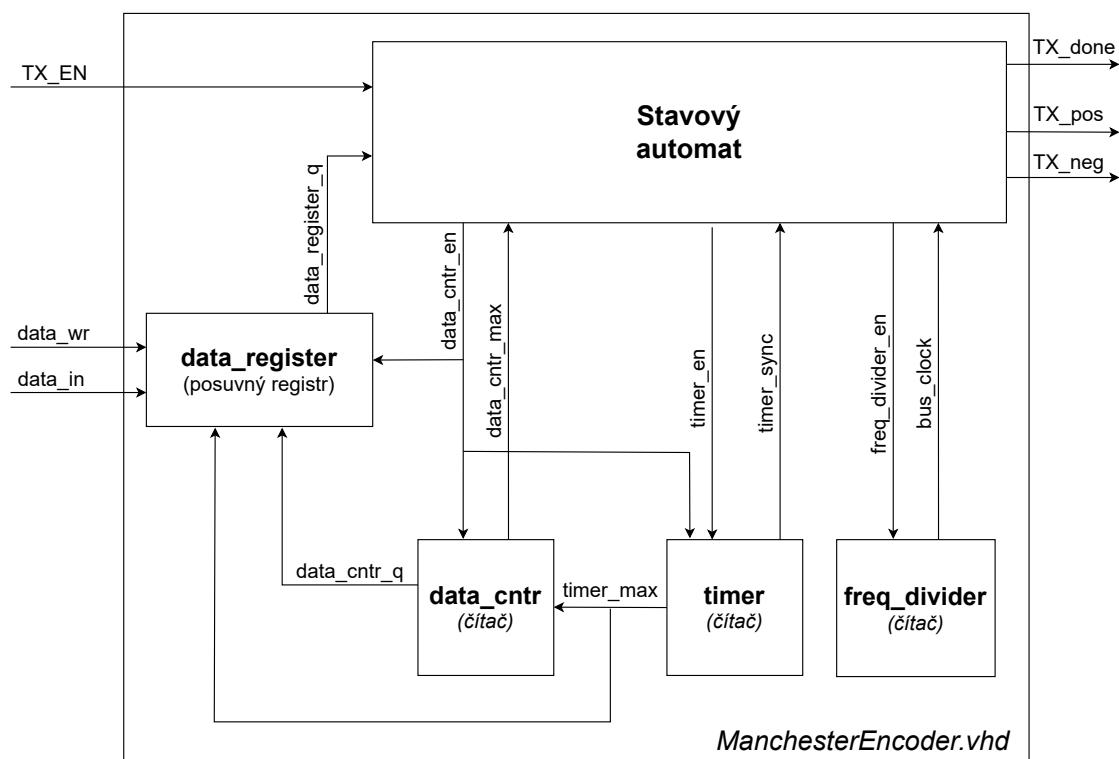
Hodnota	Význam
"00"	Výchozí stav; žádná data nebyla přijata
"01"	Přijato příkazové slovo
"10"	Přijato datové slovo
"11"	Došlo k chybě při vzorkování nebo je nesprávná parita

3.3.2 Manchester enkodér

Enkodér obsahuje tři čítače, jeden posuvný registr a sekvenční generátor parity, který je součástí posuvného registru.

Čítač (*timer*) slouží pro měření délky synchronizační vlny a dosahuje maxima po $1.5 \mu\text{s}$. Zároveň (ve stavu **ENCODE**) signalizuje uplynutí doby jedné periody, tedy $1 \mu\text{s}$, po čemž je inkrementován druhý čítač (*data_counter*), který počítá odeslané bity a dosahuje maxima při hodnotě 17 (šířka každého odeslaného slova + parita). Poslední čítač (*freq_divider*) dělí hodinový signál FPGA na frekvenci 1 MHz, která je použita pro kódování dat podle manchester kódování (viz obrázek 1.2).

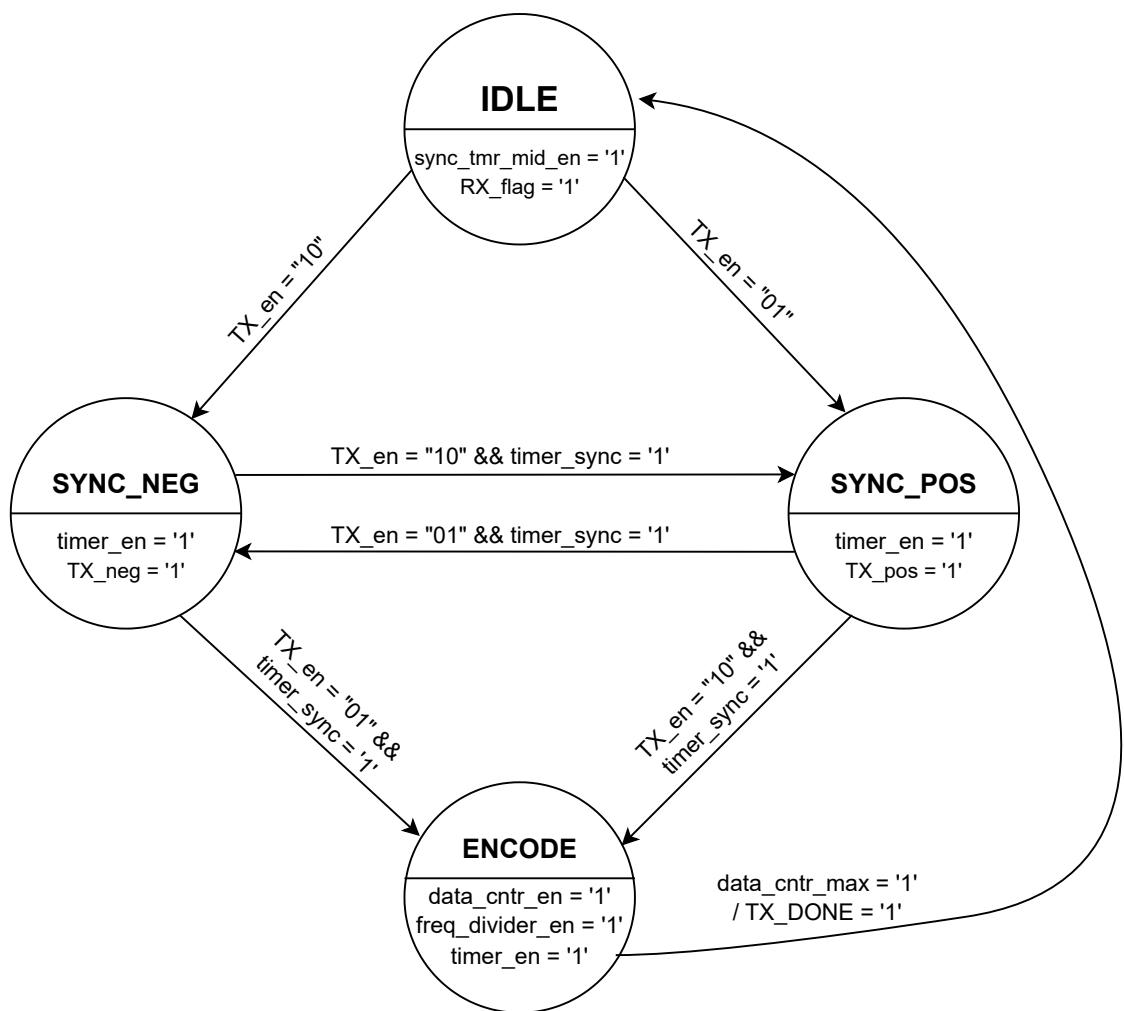
Posuvný registr obsahuje v okamžiku zahájení vysílání data z Hlavního stavového automatu a je používán pro serializaci dat. Po odeslání všech 16 bitů je do registru vsazena parita, která je při odesílání počítána sekvenčním generátorem parity.



Obr. 3.7: Architektura Manchester enkodéru

Enkodér přijímá data z Hlavního stavového automatu a na vyžádání vysílá do sběrnice, včetně *synchronizační vlny* a paritního bitu.

Obsahuje jednoduchý stavový automat o čtyřech stavech (viz obrázek 3.8). Při startu vysílání přechází automat z výchozího stavu (**IDLE**) do stavu **SYNC_POS** (pro příkazové slovo) nebo **SYNC_NEG** (pro datové slovo), ve kterém vysílá první část synchronizační vlny. Poté přechází do opačného stavu (po **SYNC_POS** do **SYNC_NEG** a naopak) a vysílá druhou část synchronizační vlny. Nakonec následuje stav **ENCODE**, ve kterém vysílá data z Hlavního stavového automatu.



Obr. 3.8: Stavový diagram Manchester enkodéru

3.3.3 Hlavní stavový automat

Řídící komponentou celého terminálu je stavový automat o 11 stavech. Pro řízení má k dispozici dva interní čítače – *cntr* (5-bit) a *err_tmr* (11-bit). První jmenovaný slouží pro počítání přijatých nebo odeslaných datových slov. Druhý je nastaven tak, že jedno maximum má při $50 \mu s$, což je maximální doba, kterou v některých stavech terminál čeká na příchozí slovo (po jejím překročení je vyhodnoceno, že došlo k **chybě na lince**); druhé maximum je v $3 \mu s$, což maximální prodleva subsystému k odpovědi na uložení/načtení dat (překročení je vyhodnoceno jako **chyba v terminálu**).

Dále stavový automat používá několik registrů:

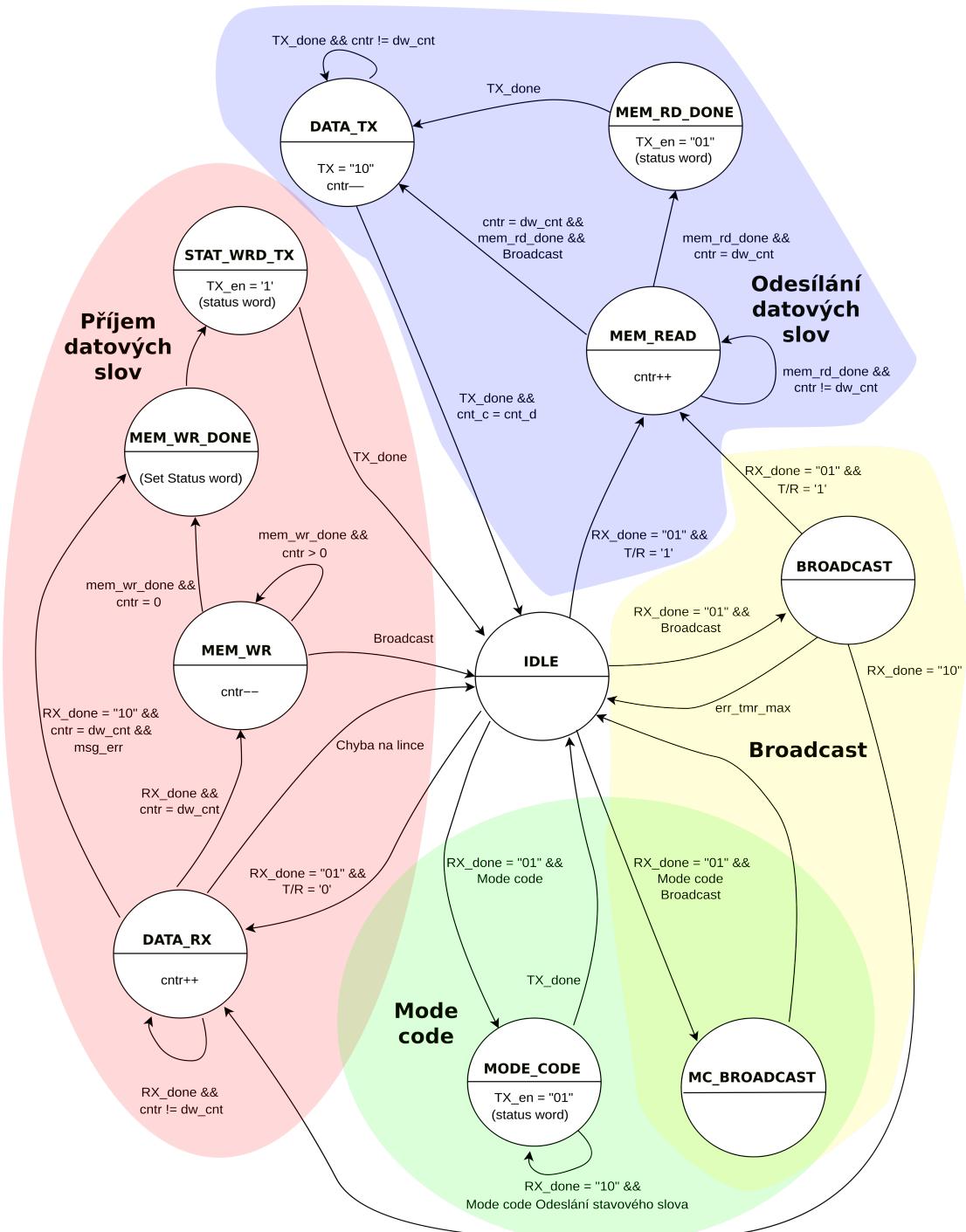
- *sw_msg_err*, *sw_brdcast* a *sw_term_err* ukládají hodnoty pro stavové slovo, které představuje signál *stat_w*
- *saddr* a *dw_cnt* jsou 5-bitové registry; obsahují hodnoty z příkazového slova a jsou dále použity při vyhodnocování podmínek.
- *mode_c* a *sync* nesou informaci z požadavku *Mode code* a jejich hodnota je vystavena jako výstupní signál z terminálu.
- Klopny obvod *data_wr* je použit pro generování krátkého impulzu při zápisu do Manchester enkodéru. Poslední klopny obvod *msg_err* drží informaci o výskytu chyby ve zprávě při příjmu dat a na základě jeho obsahu terminál rozhoduje o další akci.

Signál *TR_output* nabývá log. 1, pokud terminál odesílá data na linku. Implementován je pro koordinaci s obvodem HI-15690, viz obrázek 3.2.

Stavový diagram lze rozdělit na 4 hlavní části:

1. **Příjem datových slov** – Bylo přijato příkazové slovo s požadavkem o přijetí dat. Data jsou přijata (**DATA_RX**), uložena (**MEM_WR**), stavové slovo je nastaveno (**MEM_WR_DONE**) a odesláno (**STAT_WRD_TX**).
2. **Odesílání datových slov** – Bylo přijato příkazové slovo s požadavkem o odeslání dat. Data jsou načtena z paměti (**MEM_READ**), je odesláno stavové slovo (**MEM_RD_DONE**) a následně data (**DATA_TX**).
3. **Broadcast** – Bylo přijato příkazové slovo s *Broadcast adresou* (stav **BROADCAST**). Na základě dalšího přijatého slova následuje **odesílání** nebo **příjem datových slov**.
4. **Mode code** – Bylo přijato příkazové slovo s *Mode code* požadavkem. Po jeho provedení (**MODE_CODE**) je odesláno stavové slovo (**STAT_WRD_TX**). Pokud požadavek je zároveň v režimu *Broadcast* (**MC_BROADCAST**), stavové slovo odesláno není.

Jednotlivé stavy jsou detailně popsány v **příloze D**.



Obr. 3.9: Hlavní stavový diagram

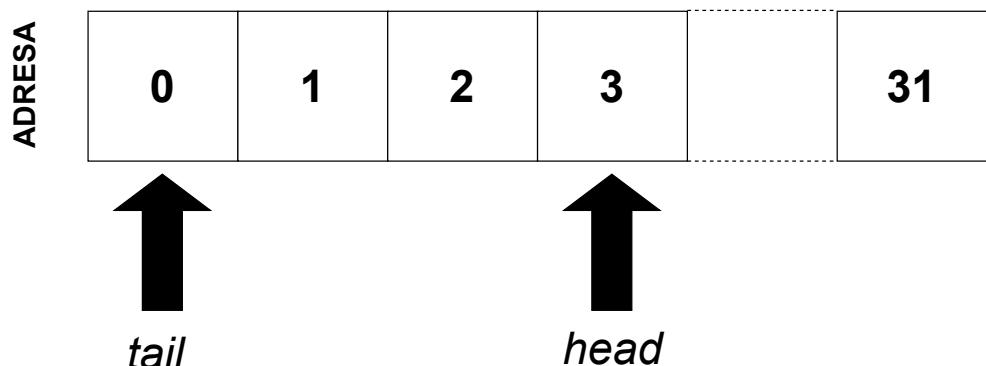
3.3.4 Verifikační buffer

Dočasné úložiště dat, do kterého Hlavní stavový automat ukládá právě přijatá (nebo z paměti načtená) data, která následně ukládá do proprietární paměti (nebo odesílá do enkodéru).

Jedná se o jednoduchou paměť SRAM typu FIFO (*first in, first out*). Velikost bufferu je 512 bitů, tedy tolik, aby byla schopna pojmit 32 16-bitových slov, což je maximum, které může obsahovat jedna zpráva. Pro zápis a čtení z bufferu jsou součástí komponenty čítače *head* a *tail*, které fungují jako ukazatele na adresu v bufferu. Při každém zápisu do bufferu jsou na adresu ukazatele *head* zapsána příslušná data a čítač je inkrementován. Při čtení z bufferu jsou data odebrána z místa, kam ukazuje *tail*, a čítač je rovněž inkrementován.

Pokud *head* i *tail* ukazují na stejnou adresu, buffer je prázdný. Pokud má *head* o jedna menší hodnotu než *tail*, buffer je plný.

Princip bufferu je na obrázku 3.10.



Obr. 3.10: Princip verifikačního bufferu s ukazateli *head* a *tail*

3.3.5 Sloha terminálu

Sloha *Terminal_package* obsahuje konstanty, které jsou použity při VHDL popisu komponent. Je zde možné měnit například adresu terminálu. Ostatní konstanty (například parita nebo *Mode code* hodnoty) jsou určené standardem MIL-STD-1553B a měly by zůstat beze změny.

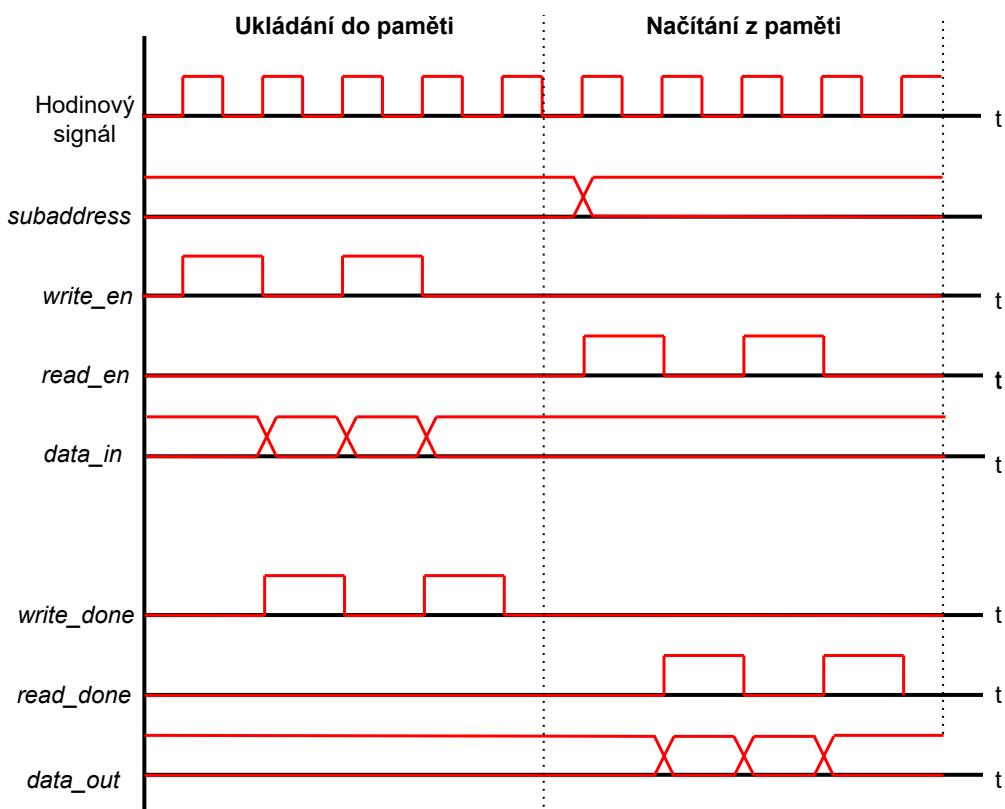
3.3.6 Komunikace s proprietární pamětí

Proprietární paměť (viz 3.4.4) je komponenta, simulující subsystém pro účely verifikace.

Zápis a čtení jsou terminálem inicializovány pomocí krátkých řídících signálů *write_en* a *read_en*, paměť po provedení dané operace posílá odpověď signály *write_done* a *read_done*.

Data jsou přenášena z/do terminálu formou stavových signálů *data_in* a *data_out*. Terminál společně s požadavkem na čtení/zápis dodává (kromě případných dat) i subadresu v podobě 5-bitového slova (*subaddress*), který určuje paměťový blok pro manipulaci s daty.

Časování jednotlivých signálů je znázorněno na obrázku 3.11.

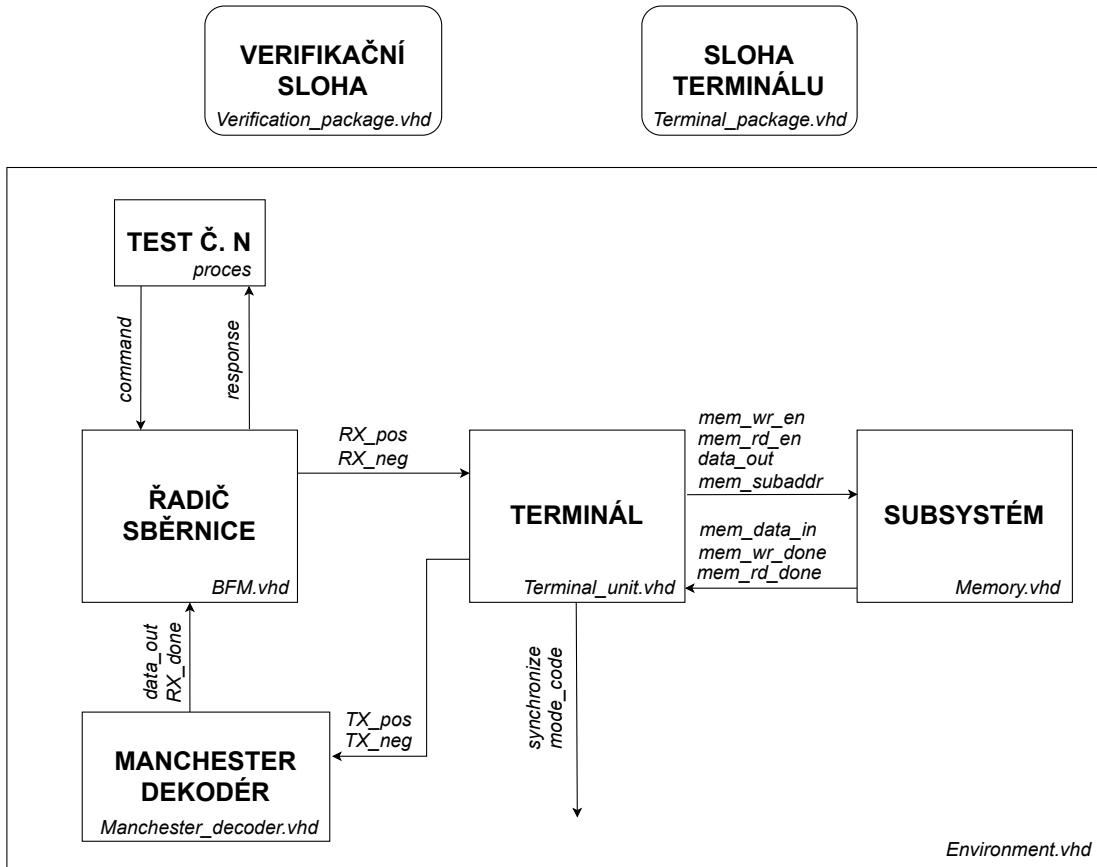


Obr. 3.11: Časování u proprietární paměti

3.4 Verifikační prostředí

Pro ověření funkčnosti a splnění všech zadaných požadavků bylo vytvořeno verifikační prostředí, které umožňuje simulovat průběhy chybné i bezchybné komunikace a přijímat odpovědi z terminálu. Obsahuje komponentu **BFM** (*bus functional model*) a **Verifikační slohu**.

Výběr testu je možný přiřazením do proměnné TEST_NUMBER. Kromě procesu, ve kterém jsou popsány všechny testovací scénaře, obsahuje topový modul prostředí (*Environment.vhd*) také proces pro generování hodinového signálu FPGA (32 MHz).



Obr. 3.12: Struktura verifikačního prostředí

3.4.1 Verifikační testy

Testy, kterým je terminál v rámci verifikace podroben, přímo vyplývají z požadavků, které jsou na něj kladený. v tabulce 3.3 jsou testy shrnutý, včetně názvu a čísla požadavku, které pokrývá. Testy jsou rozepsány v příloze E.

Tab. 3.3: Seznam testů pro verifikaci

Číslo testu	Název testu	Pokryté požadavky	Výsledek
1	Odeslání datových slov	3, 6, 7, 8, 9	✓
2	Chyba na lince (v příkazovém slově)	10 (1/3)	✓
3	Chyba na lince (v datovém slově)	10 (2/3)	✓
4	Chyba na lince (méně datových slov)	10 (3/3)	✓
5	<i>Mode code</i> požadavek	13	✓
6	Příjem dat z řadiče v režimu <i>Broadcast</i>	12 (1/2)	✓
7	Odeslání dat v režimu <i>Broadcast</i>	14	✓
8	Příjem dat z terminálu v režimu <i>Broadcast</i>	12 (2/2)	✓
9	<i>Mode code</i> požadavek v režimu <i>Broadcast</i>	15	✓
10	Chyba v terminálu	16	✓

3.4.2 BFM – Model sběrnice

Simulační model emulující průběhy signálů na lince MILBUS. Pro příjem dat je použita již vytvořená komponenta **Manchester dekodér**, BFM tedy reaguje na signály *RX_done* a *data_out*.

Ze souboru *Environment.vhd* přijímá BFM příkaz k provedení komunikace ve formě signálu *command*, kde je číslem (*command_number*) specifikován druh komunikace (odeslání stavového slova, odeslání příkazového slova, odeslání chybného slova, přijetí slova, atd.).

3.4.3 Verifikační sloha

Kromě definice nových datových typů, použitých pro propojení komponent (kvůli přehlednosti), zahrnuje sloha i všechny procedury, které jsou použity pro generování slov vysílaných do terminálu.

- **BFM procedury**

1. **Make_sync** – Do terminálu je vyslána synchronizační vlna (podoba je specifikována v argumentech procedury).
2. **Make_manchester** – Do terminálu je vyslán bit v manchester kódování.

- **Environment procedure**

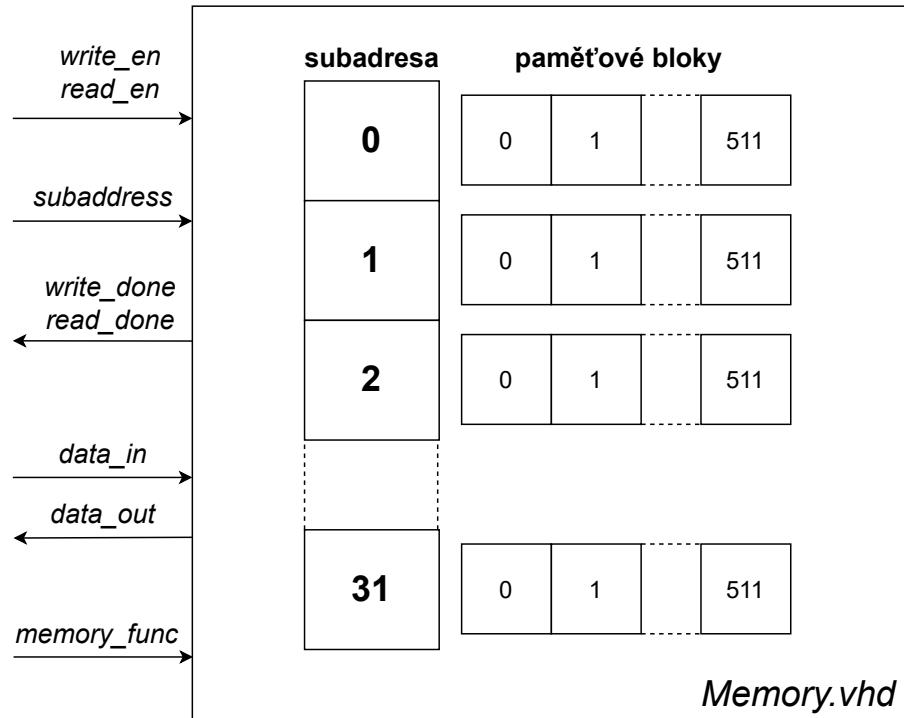
1. **Send_command_word** – Do terminálu je vysláno příkazové slovo, včetně vygenerované parity.
2. **Send_data_word** – Do terminálu je vysláno datové slovo, včetně vygenerované parity.
3. **Send_invalid_command_word** – Do terminálu je vysláno příkazové slovo. Počet odeslaných bitů, správnost parity a přítomnost *synchronizační vlny* lze specifikovat v argumentech procedury.
4. **Send_invalid_data_word** – Do terminálu je vysláno datové slovo. Počet odeslaných bitů, správnost parity a přítomnost *synchronizační vlny* lze specifikovat v argumentech procedury.
5. **Receive_word** – Čekání na příjem slova z terminálu.

3.4.4 Proprietární paměť

Komponenta (*memory.vhd*), která v rámci verifikace simuluje subsystém. Jedná se o paměť se 30 subadresami, každá má kapacitu 512 bitů.

Paměť obsahuje vstupní signál *memory_func*, který ji při log. 0 vyřadí z provozu (viz 3.3). Časování terminálu a proprietární paměti je uvedeno v sekci 3.3.6.

Struktura paměti je na obrázku 3.13.



Obr. 3.13: Struktura proprietární paměti

3.5 Syntéza a implementace

Pro vytvořenou jednotku byla provedena syntéza a implementace do obvodu FPGA Spartan3 pro model XC3S50, a to za pomoci syntetizéru XST od firmy Xilinx. Cíl optimalizace byl pro porovnání zvolen s ohledem **na plochu** i na **rychlost**.

IOB klopné obvody jsou registry, které syntetizér záměrně používá pro vstupně-výstupní signály, protože tak lze zvýšit rychlosť a spolehlivost dané aplikace. Použitá paměť RAM má dohromady kapacitu 18432 bitů, využito je pouze 512 bitů.

3.5.1 Optimalizace s ohledem na plochu

Syntetizér se snaží vytvořit s minimálním počtem použitých zdrojů – ty jsou k nahlédnutí v tabulce 3.4.

Statická časová analýza ukázala, že maximální použitelný kmitočet je 73.4 MHz. Hodnota *Slack* (tj. rezerva k periodě kmitočtu 32 MHz) je 17.376 ns.

Tab. 3.4: Celkový počet použitých zdrojů při optimalizaci s ohledem na plochu

Zdroj	využito	dostupné v FPGA	obsazení [%]
LUT (4 vstupy)	433	1536	28
IOB klopné obvody	69	124	55
Klopné obvody	154	1536	10
paměť RAM	1	4	25

3.5.2 Optimalizace s ohledem na rychlosť

Rychlosť aplikace byla navýšena na úkor použitých zdrojů, viz tabulka 3.5. Maximální použitelný kmitočet je 88.747 MHz. Hodnota *Slack* je 19.732 ns.

Tab. 3.5: Celkový počet použitých zdrojů při optimalizaci s ohledem na rychlosť

Zdroj	využito	dostupné v FPGA	obsazení [%]
LUT (4 vstupy)	627	1536	40
IOB klopné obvody	69	124	55
Klopné obvody	176	1536	11
paměť RAM	1	4	25

Závěr

Sběrnice MIL-STD-1553B je i přes své stáří stále používanou technologií, zejména pak ve vojenských a (v poslední době) také vesmírných systémech. Může za to především skutečnost, že se jedná o zaběhlou a léty osvědčenou technologií a mnohé systémy jsou pro ni přizpůsobeny. Díky své lehko uchopitelné nátuře a dostatečnému množství dokumentace se navíc stalo běžnou praxí implementovat jednotlivé periferie do obvodů FPGA, což ještě více posiluje užitečnost a variabilitu sběrnice. Protože sběrnice je popsána výhradně v zahraniční literatuře, součástí této práce byl překlad a shrnutí do srozumitelné podoby, aby se čtenář seznámil se základními (i pokročilými) funkcemi, jimiž sběrnice oplývá. Rovněž je rozebrán standard ECSS-E-ST-50-13C, který se věnuje použití sběrnice ve vesmírných (nebo obecně robustních) systémech.

V praktické části pak byla navržena architektura vzdáleného terminálu a popsána v jazyce VHDL. Terminál obsahuje celkem 4 hlavní komponenty, navrženy tak, aby mohly být použity nezávisle na sobě. Stavový automat lze navíc snadno rozšířit, např. o další hodnoty *Mode code*.

Protože požadavky na terminál ve velké míře závisí na konkrétní aplikaci, byly implementovány především základní funkcionality (odesílání/příjem dat, základní formy *Mode code* zpráv, *Broadcast*), které jsou následně verifikovány v závěrečné části práce.

Verifikace byla ve všech případech úspěšná, terminál splňuje všechny požadavky, popsané v tabulce 3.1.

Literatura

- [1] Review and rationale of mil-std-1553 a and b, 1978. URL: <https://www.milstd1553.com/wp-content/uploads/2012/12/MIL-STD-1553B.pdf>.
- [2] Open systems avionics network to replace mil-std-1553. In *19th DASC. 19th Digital Avionics Systems Conference*, page 4E5/1–4E5/6. IEEE, Philadelphia, PA, USA, 1 edition, 2000.
- [3] Mil-std-1553b vs mil-std-1553c. *Electra IC: Design & Verification*, 2018.
- [4] Hi-15690, 2021. URL: <http://www.holtic.com/products/3157-hi-15690.aspx>.
- [5] Chris Delong. Mil-std-1553b digital time division command/response multiplex data bus. In *Industrial Communication Technology Handbook, Second Edition*, pages 1–43. CRC Press, San Francisco, California, USA, 2nd edition edition, 2017.
- [6] Jakub Drbal. Implementace rychlých sériových sběrnic v obvodech fpga. Diplomová práce, Vysoké učení technické v Brně, Brno, 2014.
- [7] Leroy Earhart. Mil-std-1553. *TEST SYSTEMS, Inc.*
- [8] ECSS. Space engineering. 11 2008. URL: <https://www.milstd1553.com/wp-content/uploads/2012/12/MIL-STD-1553B.pdf>.
- [9] Robert Keim. How to decode manchester-encoded data using hardware. URL: <https://www.allaboutcircuits.com/technical-articles/how-to-decode-manchester-encoded-data-using-hardware/>.
- [10] Miloš Kutílek. Přenos informací po síti ethernet. Diplomová práce, Vysoké učení technické v Brně, Brno, 2013.
- [11] Enumala Srikrishna, L Madan Mohan, and A Mallikarjuna Prasad. Development of mil-std-1553b synthesizable ip core for avionic applications. *International Journal of Computer Science Issues (IJCSI)*, 8(5):486, 2011. URL: <https://www.ijcsi.org/papers/IJCSI-8-5-3-488-491.pdf>.
- [12] Orly Stan, Adi Cohen, Yuval Elovici, and Asaf Shabtai. On the security of mil-std-1553 communication bus. *Security and Safety Interplay of Intelligent Software Systems*, pages 153–171.
- [13] Duncan Young and John Wemekamp. Mil-std-1553 alternatives look to knock off the king. *Electronic Design*, 45(20):162–163, 1997.

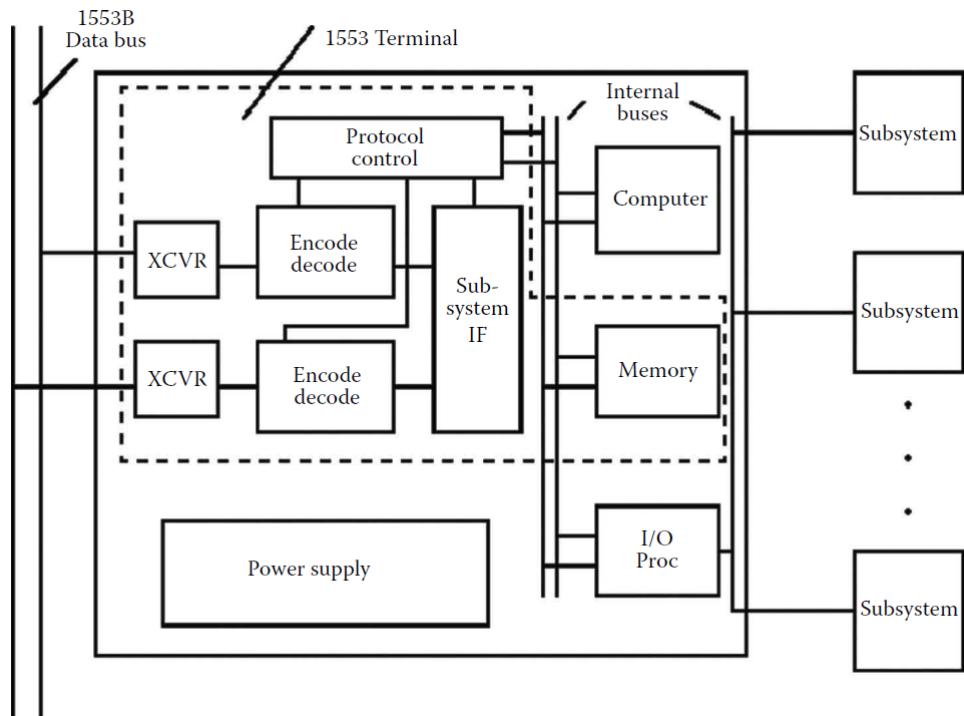
Seznam symbolů a zkratek

ADB	Příjem datového bloku – Acquisition Data Block
ATC	Potvrzení příjmu – Acquisition Transfer confirmation
ATR	Žádost o příjem – Acquisition Transfer Request
BC	Řadič sběrnice – Bus controller
BFM	Model sběrnice – Bus Functional Model
DBC	Dynamické řízení sběrnice – Dynamic Bus Control
DDB	Přenos datového bloku – Distribution Data Block
DFF	Klopný obvod typu D – D Flip Flop
DTC	Potvrzení přenosu datového bloku – Distribution Transfer Confirmation
DTD	Deskripce datového bloku – Distribution Transfer Descriptor
FIFO	Fronta (první dovnitř, první ven) – First In, First Out
FPGA	Programovatelné hradlové pole – Field Programable Gate Array
IOB	Vstupně-výstupní registr – Input/Output Buffer
LUT	Náhledová tabulka – Lookup Table
QoS	Quality of Service
RT	Terminál – Remote terminal
T/R	Vysílání/Příjem – Transmit/Receieve
SRAM	Statická paměť – Static Random Access Memory

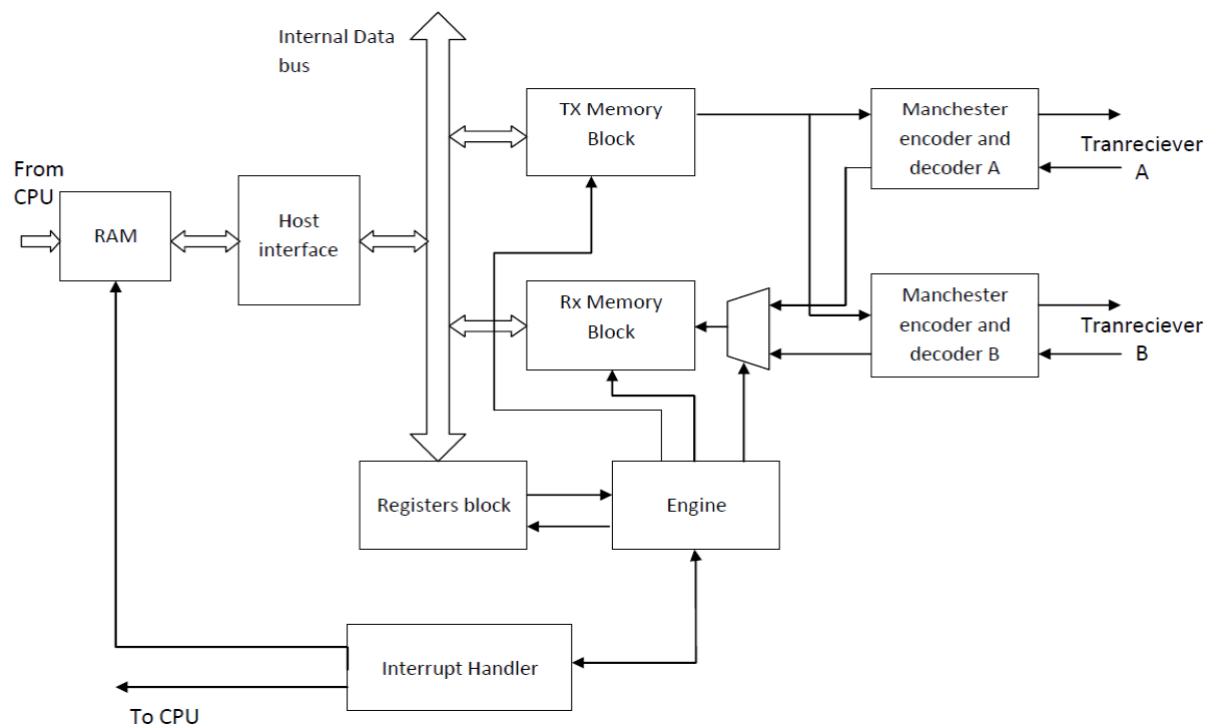
Seznam příloh

A Architektury terminálu v literatuře	55
B Komunikace dle standardu ECSS-E-ST-50-13C	57
B.1 Odesílání dat	57
B.2 Příjem dat	59
B.3 Deskripce datového bloku (DTD)	61
B.4 Potvrzení přenosu datového bloku (DTC)	61
B.5 Žádost o příjem (ATR)	62
B.6 Potvrzení příjmu (ATC)	62
C Obvod HI-15960	63
D Popis stavů v Hlavním stavovém diagramu	64
E Verifikační testy	67
E.1 Test č. 1 – Odeslání datových slov	67
E.2 Test č. 2 – Chyba na lince (v příkazovém slově)	68
E.3 Test č. 3 – Chyba na lince (v datovém slově)	68
E.4 Test č. 4 – Chyba na lince (méně datových slov)	69
E.5 Test č. 5 – Mode code požadavek	69
E.6 Test č. 6 – Příjem dat z řadiče v režimu <i>Broadcast</i>	70
E.7 Test č. 7 – Odeslání dat v režimu <i>Broadcast</i>	70
E.8 Test č. 8 – Příjem dat z jiného terminálu v režimu <i>Broadcast</i>	71
E.9 Test č. 9 – <i>Mode code</i> požadavek v režimu <i>Broadcast</i>	71
E.10 Test č. 10 – Chyba v terminálu	72

A Architektury terminálu v literatuře



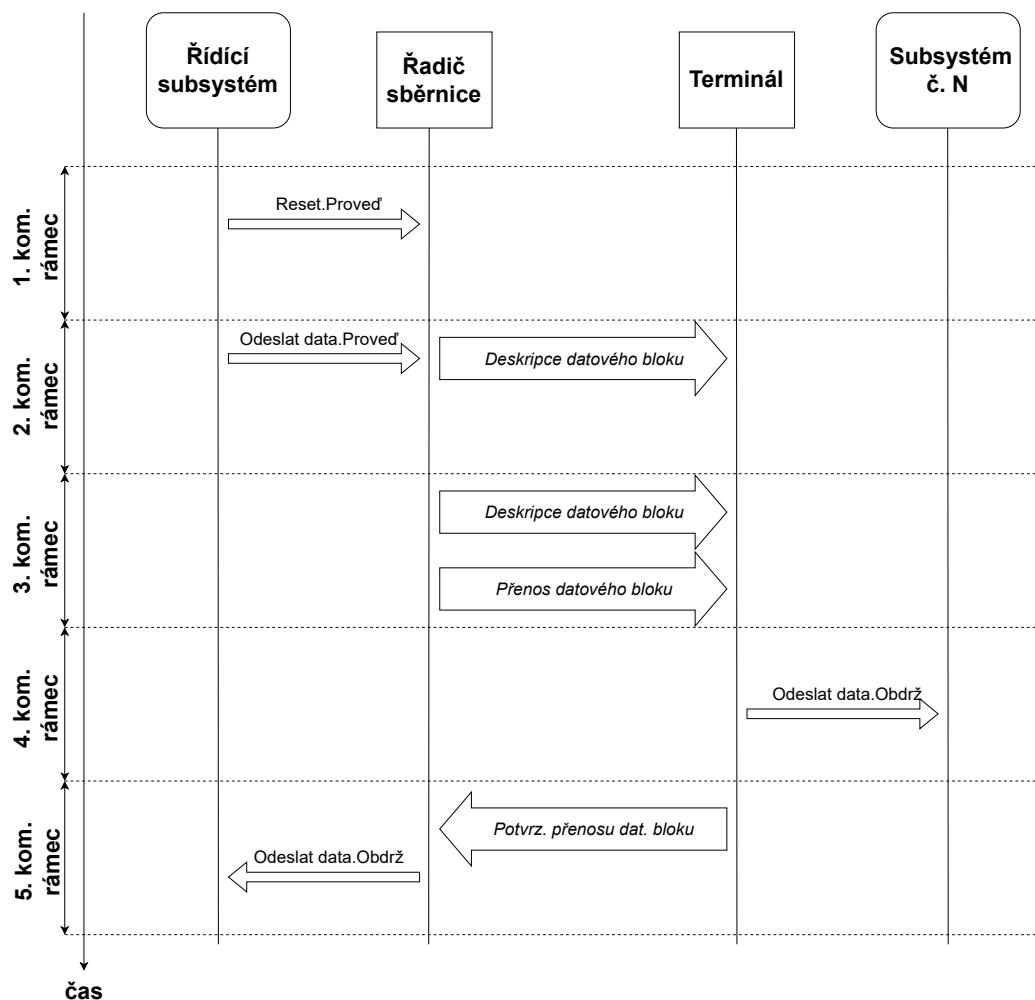
Obr. A.1: Architektura terminálu podle publikace [5]



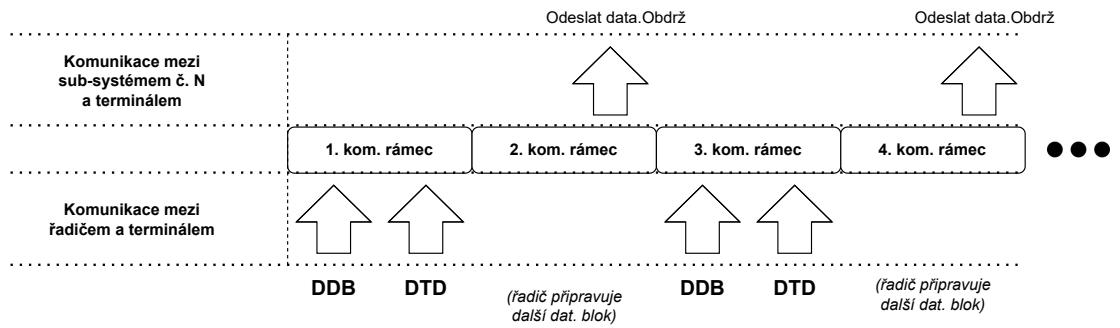
Obr. A.2: Architektura terminálu podle publikace [11]

B Komunikace dle standardu ECSS-E-ST-50-13C

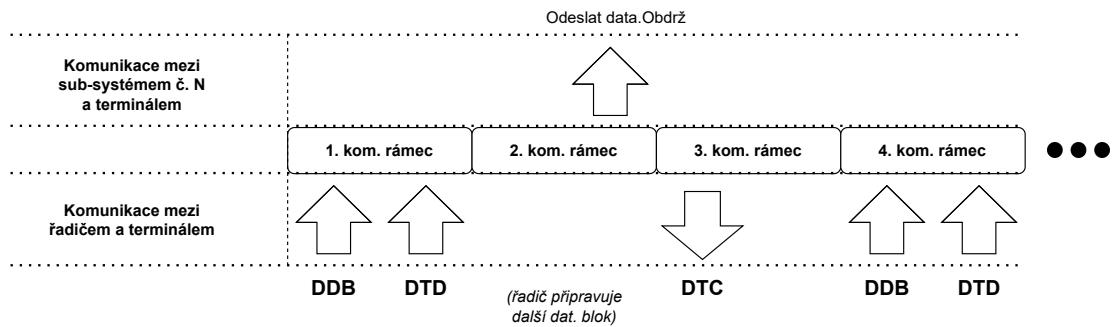
B.1 Odesílání dat



Obr. B.1: Odesílání dat z řadiče do terminálu podle služby datových toků

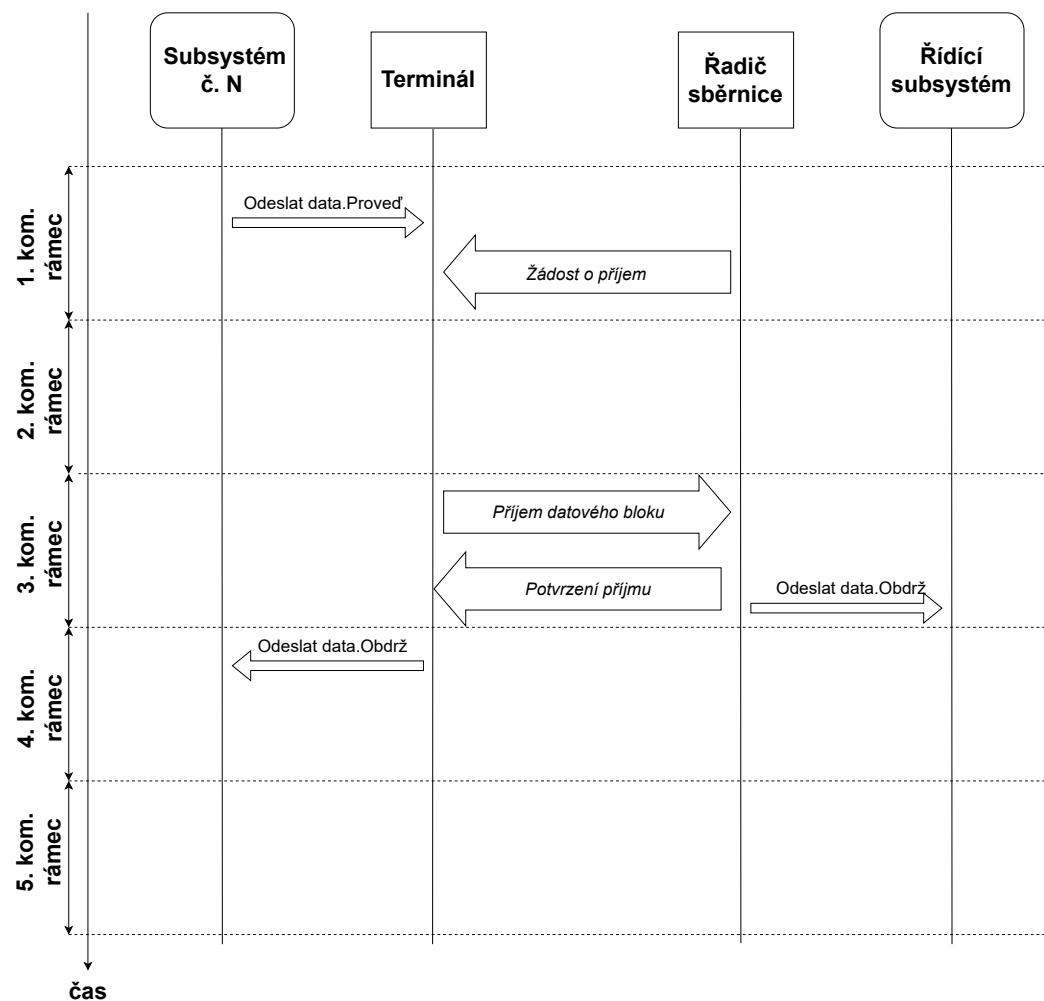


Obr. B.2: Průběh odesílání dat z řadiče do terminálu metodou **Best Effort** v rámci služby datových toků

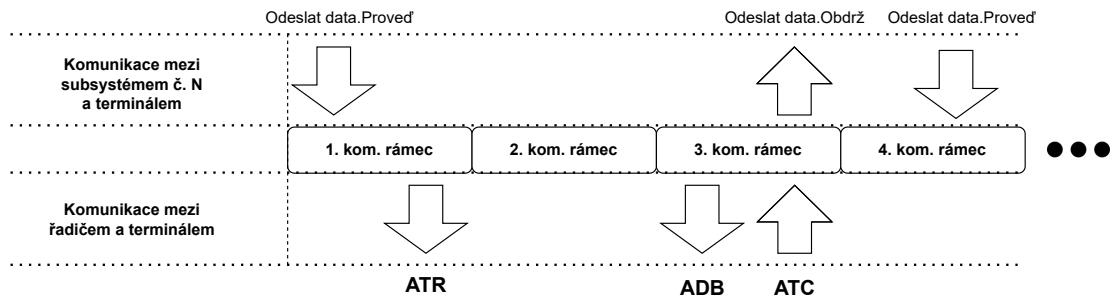


Obr. B.3: Průběh odesílání dat z řadiče do terminálu metodou **Verified Length** v rámci služby datových toků

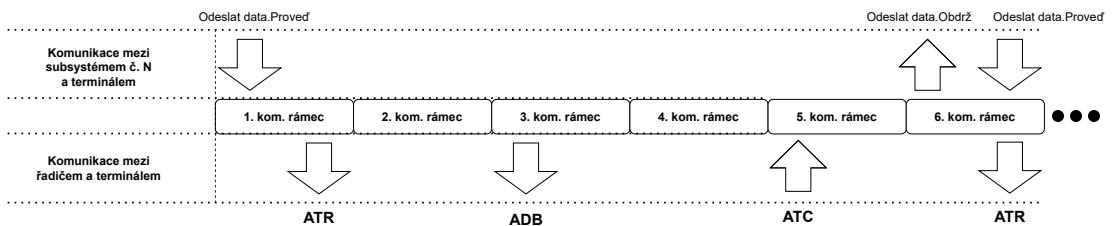
B.2 Příjem dat



Obr. B.4: Příjem dat z terminálu do řadiče podle služby datových toků



Obr. B.5: Průběh příjmu dat z terminálu do řadiče metodou **Best Effort** v rámci služby datových toků



Obr. B.6: Průběh příjmu dat z terminálu do řadiče metodou **Verified Length** v rámci služby datových toků

B.3 Deskripce datového bloku (DTD)

1. datové slovo		2. datové slovo					
"0000"	Velikost dat. bloku	Quality of Service	Reset	Adresování	Subadresa	číslo odeslaného dat. bloku	
4 bity	12 bitů	1 bit	1 bit	1 bit	5 bitů	8 bitů	

Obr. B.7: Struktura zprávy Deskripce datového bloku

- **Velikost datového bloku** – Určuje počet bajtů v DDB
- **Quality of Service** – Log. 0 znamená **Best effort**, log. 1 **Verified Length**
- **Reset** – Signalizace terminálu, že má provést reset
- **Adresování** – Log. 0 znamená **Flat**, log. 1 **Deep**
- **Subadresa** – V případě **Deep** adresování obsahuje subadresu subsystému, při **Flat** má být trvale v hodnotě "01011"
- **Číslo odeslaného dat. bloku** – S každým dalším datovým blokem je inkrementován o 1

B.4 Potvrzení přenosu datového bloku (DTC)

1. datové slovo		2. datové slovo					
"0000"	Velikost dat. bloku	Chyba	Reset	Adresování	Subadresa	číslo odeslaného dat. bloku	
4 bity	12 bitů	1 bit	1 bit	1 bit	5 bitů	8 bitů	

Obr. B.8: Struktura zprávy Potvrzení přenosu datového bloku

- **Velikost datového bloku** – Určuje počet bajtů v DDB
- **Chyba** – Nabývá log. 1, pokud při přenosu došlo k chybě
- **Reset** – Signalizuje řadiči, že byl proveden reset terminálu
- **Adresování** – Log. 0 znamená **Flat**, log. 1 **Deep** (*měl by mít stejnou hodnotu jako v DTD*)
- **Subadresa** – V případě **Deep** adresování obsahuje subadresu subsystému, při **Flat** má být trvale v hodnotě "01011" (*měl by mít stejnou hodnotu jako v DTD*)
- **Číslo odeslaného dat. bloku** – S každým dalším datovým blokem je inkrementován o 1 (*měl by mít stejnou hodnotu jako v DTD*)

B.5 Žádost o příjem (ATR)

1. datové slovo		2. datové slovo					
"0000"	Velikost dat. bloku	Quality of service	Reset	Adresování	Subadresa	číslo přijatého dat. bloku	
4 bity	12 bitů	1 bit	1 bit	1 bit	5 bitů	8 bitů	

Obr. B.9: Struktura zprávy Žádost o příjem

- **Velikost datového bloku** – Určuje počet bajtů v ADB
- **Quality of Service** – Log. 0 znamená **Best effort**, log. 1 **Verified Length**
- **Reset** – Signalizuje řadiči, že má být terminál resetován
- **Adresování** – Log. 0 znamená **Flat**, log. 1 **Deep**
- **Subadresa** – V případě **Deep** adresování obsahuje subadresu subsystému, při **Flat** má být trvale v hodnotě "01011"
- **Číslo přijatého dat. bloku** – S každým dalším datovým blokem je inkrementován o 1

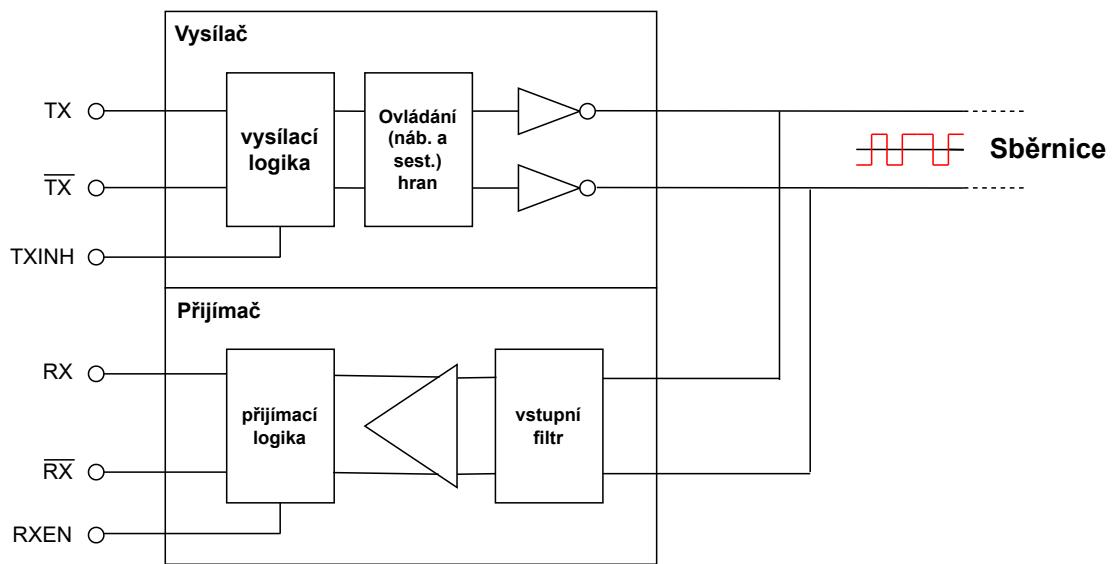
B.6 Potvrzení příjmu (ATC)

1. datové slovo		2. datové slovo					
"0000"	Velikost dat. bloku	Chyba	Reset	Adresování	Subadresa	číslo přijatého dat. bloku	
4 bity	12 bitů	1 bit	1 bit	1 bit	5 bitů	8 bitů	

Obr. B.10: Struktura zprávy Potvrzení příjmu

- **Velikost datového bloku** – Určuje počet bajtů v DDB
- **Chyba** – Nabývá log. 1, pokud při přenosu došlo k chybě
- **Reset** – Je-li v log. 1, terminál žádá řadič, aby u něj provedl reset
- **Adresování** – Log. 0 znamená **Flat**, log. 1 **Deep** (*měl by mít stejnou hodnotu jako v ATR*)
- **Subadresa** – V případě **Deep** adresování obsahuje subadresu subsystému, při **Flat** má být trvale v hodnotě "01011" (*měl by mít stejnou hodnotu jako v ATR*)
- **Číslo přijatého dat. bloku** – S každým dalším datovým blokem je inkrementován o 1 (*měl by mít stejnou hodnotu jako v DTD*)

C Obvod HI-15960



Obr. C.1: Vnitřní struktura obvodu HI-15960 (převzato z [4])

D Popis stavů v Hlavním stavovém diagramu

- **IDLE** – Výchozí stav terminálu, ve kterém čeká na příchozí příkazové slovo (datové slovo je ignorováno). V případě signalizace chyby ($RX_done = "11"$) je do stavového registru uložen příznak o chybě.

Možné další stavы:

- **DATA_RX** – Bylo přijato příkazové slovo s požadavkem na přijetí a uložení dat do paměti.
- **MEM_READ** – Bylo přijato příkazové slovo s požadavkem na odeslání dat z paměti.
- **MODE_CODE** – Bylo přijato příkazové slovo ve formě *mode_code* požadavku.
- **BROADCAST** – Bylo přijato příkazové slovo s *broadcast* adresou.
- **MC_BROADCAST** – Bylo přijato příkazové slovo s *broadcast* adresou ve formě *mode_code* požadavku

- **DATA_RX** – Terminál očekává datová slova a po každém přijetí (a uložení do paměti SRAM) inkrementuje svůj interní čítač.

Možné další stavы:

- **MEM_WR** – Všechna datová slova byla přijata (počet očekávaných slov je určen v předchozím příkazovém slově) a všechna jsou bez chyby.
- **MEM_WR_DONE** – Všechna datová slova byla přijata (počet očekávaných slov je určen v předchozím příkazovém slově), ale minimálně jedno z nich je chybné.
- **IDLE** – Počet přijatých datových slov neodpovídá očekávání, případně bylo přijato příkazové slovo. Do stavového registru je uložen příznak o chybě.

- **MEM_WR** – Terminál se opakovaně pokouší přenést data z vnitřní paměti SRAM do vnější proprietární paměti. Po každém úspěšném uložení je dekrementován interní čítač.

Možné další stavы:

- **IDLE** – Všechna data byla úspěšně uložena (pokud ne, je o tom veden příznak ve stavovém slově) a přenos probíhal v režimu *Broadcast*.
- **MEM_WR_DONE** – Všechna data byla úspěšně uložena (pokud ne, je o tom veden příznak ve stavovém slově).

- **MEM_WR_DONE** – Mezistav pro nastavení stavového slova.

Možné další stavy:

- **STAT_WRD_TX**

- **STAT_WRD_TX** – Terminál odesílá stavové slovo.

Možné další stavy:

- **IDLE** – Po odeslání stavového slova přechod do výchozího stavu. Pokud přenos stavového slova nebyl vynucem požadavkem *Mode code*, stavové slovo je resetováno.

- **MEM_READ** – Načítá data z vnější paměti do vnitřní (SRAM) a po každém úspěšném načtení je inkrementován interní čítač.

Možné další stavy:

- **DATA_TX** – Všechna data byla načtena (pokud ne, je o tom uložen příznak ve stavovém registru) a přenos probíhá v režimu *Broadcast*.
- **MEM_RD_DONE** – Všechna data byla načtena (pokud ne, je o tom uložen příznak ve stavovém registru).

- **MEM_RD_DONE** – Terminál odesílá stavové slovo.

Možné další stavy:

- **IDLE** – Stavové slovo bylo odesláno, ale během načítání dat z paměti (ve stavu **MEM_READ**) došlo k chybě.
- **DATA_TX** – Stavové slovo bylo odesláno.

- **DATA_TX** – Terminál odesílá požadovaná data. Po každém úspěšném odeslání je dekrementován interní čítač.

Možné další stavy:

- **IDLE** – Data byla odeslána.

- **MODE_CODE** – Mezistav pro požadavek *Mode code*; *Mode code* hodnota je vystavena jako výstupní signál z terminálu a pokud se rovná "10001" (Synchronizace), očekává se datové slovo.

Možné další stavy:

- **IDLE** – Očekávané datové slovo (v případě Synchronizace) nepřišlo; do stavového registru je uložen příznak o chybě.
- **STAT_WRD_TX** – Po každém *Mode code* požadavku (včetně Synchronizace) následuje odeslání stavového slova.

- **BROADCAST** – Terminál obdržel požadavek pro příjem nebo odeslání v režimu *broadcast* a nyní čeká na další slovo.

Možné další stavы:

- **IDLE** – Očekávané příkazové nebo datové slovo nepřišlo do $50 \mu\text{s}$. Do stavového registru je uložen příznak o chybě.
- **MEM_READ** – Terminál přijal příkazové slovo s požadavkem pro odeslání dat do ostatních terminálů.
- **DATA_RX** – Terminál přijal datové slovo, které okamžitě ukládá do paměti SRAM a očekává další datová slova.

- **MC_BROADCAST** – Terminál obdržel požadavek *Mode code* v režimu *broadcast*. Pokud se jedná o Synchronizaci, terminál očekává datové slovo.

Možné další stavы:

- **IDLE** – Po obdržení datového slova (nebo po příliš dlouhé prodlevě → uložení příznaku o chybě) automat přechází do výchozího stavu. Pokud se nejednalo o Synchronizaci, automat přechází do výchozího stavu okamžitě.

E Verifikační testy

Níže jsou rozepsány všechny testy, kterým byl terminál při verifikaci podroben.

Legenda:

- DC = *data word count* = počet datových slov
- Addr = adresa terminálu (není-li uvedena → adresa vytvořeného terminálu)
- SA = subadresa (adresa subsystému)
- T/R = *Transmit/Receive bit* ('1' = vysílání)

E.1 Test č. 1 – Odeslání datových slov

Do terminálu jsou odeslány dvě zprávy, které obsahují data k uložení do dvou různých subsystémů. Následně terminál obdrží požadavek k odeslání části dat z jednoho subsystému.

Průběh:

1. Reset terminálu
2. Odeslání příkazového slova (T/R = '0', DC = 7, SA = "11100")
3. Odeslání sedmi datových slov (s inkrementující hodnotou)
4. Přijetí stavového slova
5. Odeslání příkazového slova (T/R = '0', DC = 4, SA = "00101")
6. Odeslání čtyř datových slov (s inkrementující hodnotou)
7. Přijetí stavového slova
8. Odeslání příkazového slova (T/R = '1', DC = 3, SA = "11100")
9. Přijetí stavového slova
10. Přijetí 3 datových slov

Výsledek:

Data z obou zpráv terminál ukládá do proprietární paměti, ze které je při třetí zprávě načítá a vysílá zpět. Stavová slova neobsahují žádné chyby, přijatá data korelují s odeslanými daty.

E.2 Test č. 2 – Chyba na lince (v příkazovém slově)

Do terminálu jsou odeslány dvě zprávy, které začínají chybným příkazovým slovem (chybná parita a krátká zpráva). Po každé zprávě je požadavkem *Mode code* zkонтrolováno, že terminál chybu zaznamenal a uložil daný příznak do stavového slova.

Průběh:

1. Reset terminálu
2. Odeslání příkazového slova ($T/R = '0'$, DC = 3, SA = "00010", chybná parita)
3. Odeslání tří datových slov (s inkrementující hodnotou)
4. Odeslání Mode code (Transmit status word) požadavku
5. Přijetí stavového slova
6. Odeslání příkazového slova ($T/R = '0'$, DC = 4, SA = "00001", krátká zpráva)
7. Odeslání čtyř datových slov (s inkrementující hodnotou)
8. Odeslání Mode code (Odeslání stavového slova) požadavku
9. Přijetí stavového slova

Výsledek:

Terminál na první zprávu reaguje pouze uložením příznaku chyby, datová slova ignoruje. Na vyžádání poté posílá stavové slovo. Při druhé zprávě se celá situace opakuje.

E.3 Test č. 3 – Chyba na lince (v datovém slově)

Do terminálu jsou odeslány dvě zprávy s požadavkem o přijetí dat, která jsou chybná (chybná parita a přenos bez synchronizace). Po první zprávě je obdrženo stavové slovo s příznakem chyby, po druhé zprávě je pro stavové slovo odeslán požadavek *Mode code*.

Průběh:

1. Reset terminálu
2. Odeslání příkazového slova ($T/R = '0'$, DC = 3, SA = "00001")
3. Odeslání tří datových slov (chybná parita)
4. Přijetí stavového slova
5. Odeslání příkazového slova ($T/R = '0'$, DC = 3, SA = "01100")
6. Odeslání tří datových slov (bez synchronizace)
7. Odeslání Mode code (Transmit status word) požadavku
8. Přijetí stavového slova

Výsledek:

Terminál při první zprávě přijímá datová slova, už při zpracování je kvůli chybné paritě vyhodnotí jako chybná. Po přijetí všech data neukládá a posílá stavové slovo o chybě. Při druhé zprávě nejsou datová slova přijata vůbec (chybí synchronizace), terminál uloží příznak o chybě a na vyžádání jej odesílá ve stavovém slově.

E.4 Test č. 4 – Chyba na lince (méně datových slov)

Do terminálu je odeslána zpráva s počtem datových slov, který je menší než bylo specifikováno v příkazovém slově. Následně je pro stavové slovo odeslán požadavek *Mode code*.

Průběh:

1. Reset terminálu
2. Odeslání příkazového slova ($T/R = '0'$, $DC = 5$, $SA = "01100"$)
3. Odeslání tří datových slov (s inkrementující hodnotou)
4. Počkat $35 \mu s$
5. Odeslání *Mode code* (Odeslání stavového slova) požadavku
6. Přijetí stavového slova

Výsledek:

Terminál přijímá očekávaná data, ale protože jich není správný počet, data neukládá; na vyžádání *Mode code* je odesláno stavové slovo s příznakem o chybě.

E.5 Test č. 5 – Mode code požadavek

Do terminálu je odesláno chybné příkazové slovo pro invokaci chyby ve stavovém slově. Poté jsou postupně odesílány *Mode code* požadavky – nejprve Odeslání stavového slova, pak Synchronizace a nakonec neimplementovaná *Mode code* hodnota.

Průběh:

1. Reset terminálu
2. Odeslání chybného příkazového slova (chybná parita)
3. Odeslání požadavku *Mode Code* (Odeslání stavového slova)
4. Přijetí stavového slova
5. Odeslání požadavku *Mode Code* (Synchronizace)
6. Odeslání datového slova
7. Přijetí stavového slova
8. Odeslání požadavku *Mode Code* (-)
9. Přijetí stavového slova

Výsledek:

Terminál ukládá při první zprávě příznak o chybě, v reakci na požadavek *Mode code* pak odesílá stavové slovo (\rightarrow hodnota je ponechána, viz 1.6.1). Po *Mode code* (Synchronizace) očekává datové slovo, které vystavuje jako výstupní signál *synchronize*, poté odesílá stavové slovo (\rightarrow hodnota je resetována). Po třetím *Mode code* požadavku znovu dochází k odeslání stavového slova (\rightarrow hodnota je resetována).

Po všech požadavcích *Mode code* je bitová hodnota *Mode code* vystavena jako výstupní signál.

E.6 Test č. 6 – Příjem dat z řadiče v režimu ***Broadcast***

Do terminálu je odeslána zpráva s *Broadcast* adresou se čtyřmi datovými slovy k uložení. Následuje požadavek pro odeslání uložených dat.

Průběh:

1. Reset terminálu
2. Odeslání příkazového slova (T/R = '0', DC = 4, Addr = "00000", SA = "10001")
3. Odeslání čtyř datových slov. (s inkrementující hodnotou)
4. Odeslání příkazového slova (T/R = '1', DC = 4, SA = "10001")
5. Přijetí stavového slova.
6. Přijetí čtyř datových slov.

Výsledek:

Terminál přijímá data v režimu *Broadcast* a ukládá je do subsystému. Stavové slovo neodesílá. V další zprávě přijatá data z paměti načítá a posílá zpět na linku (včetně stavového slova).

E.7 Test č. 7 – Odeslání dat v režimu ***Broadcast***

Na linku je odeslán požadavek pro přijetí dat s *Broadcast* adresou. Následuje příkazové slovo s adresou testovaného terminálu a požadavkem o vysílání 4 datových slov. Data (určena ostatním terminálům) jsou následně obdržena.

Průběh:

1. Reset terminálu
2. Odeslání příkazového slova (T/R = '0', DC = 4, Addr = "00000", SA = "10010")
3. Odeslání příkazového slova (T/R = '1', DC = 4, SA = "10101")
4. Přijetí stavového slova
5. Přijetí čtyř datových slov

Výsledek:

Terminál po prvním příkazovém slově přechází do ***Broadcast*** stavu, ve kterém setrvává, dokud neobdrží požadavek pro vysílání dat – poté odesílá data z vybraného subsystému na linku (včetně stavového slova).

E.8 Test č. 8 – Příjem dat z jiného terminálu v režimu *Broadcast*

Na linku je odeslán požadavek o přijetí dat v režimu *Broadcast*. Následuje požadavek pro jiný terminál (jiná adresa než testovaný terminál) k vysílání 14 datových slov. Datová slova jsou poté odeslána. Požadavkem o odeslání dat z testovaného terminálu je ověřeno, zda terminál data přijal.

Průběh:

1. Reset terminálu
2. Odeslání příkazového slova (T/R = '0', DC = 14, Addr = "00000", SA = "10010")
3. Odeslání příkazového slova (T/R = '1', DC = 14, Addr = "10001", SA = "10101")
4. Odeslání čtrnácti datových slov (s inkrementující hodnotou)
5. Odeslání příkazového slova (T/R = '1', DC = 5, SA = "10010")
6. Přijetí stavového slova
7. Přijetí pěti datových slov

Výsledek:

Terminál po prvním příkazovém slově přechází do ***Broadcast*** stavu, ve kterém setrvává, dokud neobdrží první datové slovo (příkazové slovo předtím ignoruje, protože má jinou adresu). Po přijetí všech datových slov terminál data ukládá (bez odeslání stavového slova). Následně v další zprávě část dat odesílá na linku.

E.9 Test č. 9 – *Mode code* požadavek v režimu *Broadcast*

Do terminálu je odeslán *Mode code* v režimu *Broadcast* (Synchronizace). Poté následuje znova *Mode code* v *Broadcast* režimu, tentokrát s neimplementovaným *Mode code* požadavkem.

Průběh:

1. Reset terminálu
2. Odeslání požadavku *Mode code* v režimu *Broadcast* (synchronizace)
3. Odeslání datového slova
4. Odeslání požadavku *Mode code* v režimu *Broadcast* (-)

Výsledek:

Terminál přijímá *Mode code* požadavky, vystavuje je jako výstupní signál a podle potřeby provádí (Synchronizace). Stavové slovo po provedení odesláno není.

E.10 Test č. 10 – Chyba v terminálu

Paměť (subsvstém) je vyřazena z provozu. Do terminálu je odeslána zpráva s požadavkem o přijetí datových slov.

Průběh:

1. Nastavení signálu MEMORY_FUNC na false
2. Reset terminálu
3. Odeslání příkazového slova (T/R = '0', DC = 7, SA = "11100")
4. Odeslání sedmi datových slov (s inkrementující hodnotou)
5. Přijetí stavového slova

Výsledek:

Terminál přijímá data a pokouší se je uložit do paměti; po $3 \mu\text{s}$ ovšem vyhodnocuje, že paměť neodpovídá, proto odesílá stavové slovo s příznakem o chybě.