Předmluva

Vrtulník, jakožto velmi komplikovaný systém z pohledů automatického řízení, je stále velkou výzvou pro všechny, kdo se zabývají teorií regulačních systémů a mají vztah k létání. Na katedře Kybernetiky FEL ČVUT doposud nebyl v provozu model helikoptéry, odpovídající svým chováním a složitostí provozu reálného stroje.

Tato diplomová práce společně s prací Milana Martináka [1] navazuje na diplomovou práci dvojice Ing. Jaroslava Fojtíka [6] a Ing. Pavla Krska [5]. Obojí zadání jsou natolik provázaná, že je nelze oddělit. Většinu činností jsem prováděl ve spolupráci s kolegou Martinákem a to i z důvodu bezpečnosti, jelikož práce na modelu a jeho testování vyžaduje pro zachování pravidel bezpečného provozu dva pracovníky.

Naši předchůdci nemohli vzhledem k velkému objemu práce, její složitosti a možnostem výpočetní techniky v době realizace, model dokončit. Proto hlavním cílem mé práce je upravit stávající model a především vytvořit takový software pro jeho ovládání, který umožní vývoj a ověřování složitých regulačních algoritmů pro řízení helikoptéry v reálném čase.

Řešení projektu vyžadovalo znovuzrekonstruování některých částí hardware a vyřešení celé řady dílčích problémů. Po konzultaci s vedoucím práce jsem hlavní pozornost věnoval vývoji řídicí aplikace v prostředí Matlab a s ní úzce související problematice komunikace modelu s počítačem a interface. K ověření výsledů práce byl vytvořen demonstrační letový program, jenž ukazuje možnosti a chování modelu.

Obecnými vlastnostmi modelu a fyzikálním popisem včetně základních letových režimů se zabývá první kapitola nazvaná Model helikoptéry, principy řízení a letu. Většina realizované problematiky se týká přenosu dat, jejich kódování a zabezpečení. Teoretický přehled je uveden v kapitole třetí s názvem Přenos dat. Upoutání modelu, jeho provoz v laboratoři a problematiku napájení řeší kapitola Konstrukce trenažéru a napájecího zdroje. Hardwarovou realizaci s důrazem na problematiku komunikace popisuji v kapitole Hardwarová část projektu. Podrobnějším popisem veškerých částí hardwaru se zabývá ve své práci kolega Milan Martinák. Řídicí aplikaci, včetně popisu důležitých funkcí a GUI rozhraní, popisuje kapitola Softwarová část projektu. Poslední část je věnována demonstračnímu programu a jeho uvedení do provozu.

Čtenáři, který není obeznámen s problematikou modelu, doporučuji k nastudování podkapitoly týkající se celkové koncepce hardwaru i softwaru, jež uvádím pro lepší orientaci v celém systému.

Na konci práce je v příloze uveden kompletní výpis programů, které byly v rámci projektu vytvořeny. V textu jsou obsaženy odkazy na tyto programy a taktéž vysvětlení a popis nejdůležitějších částí.

Rád bych poděkoval panu Ing. Františku Vaňkovi za celkovou podporu při realizaci projektu,
Ing. Jaroslavu Pekařovi a Ing. Roubalovi za cenné informace při tvorbě vizualizace a řídicích algoritmů, dále Ing. Michalu Kutilovi a Tomáši Krajníkovi za hodnotné připomínky a náměty. V neposlední řadě chci poděkovat také rodičům za umožnění studia.

1.Obsah

Předmluva 1

1. Obsah 2

2. Model helikoptéry, principy řízení a letu 5
2.1. Konstrukce a vlastnosti modelu 5
2.1.1 Funkční části vrtulníku 5
2.1.2 Hlavní a vyrovnávací rotor 6
2.1.3 Drak modelu a pohonná jednotka 6
2.1.4 Systémy pro řízení a přistávání 7
2.2. Princip řízení vrtulníku 7
2.2.1 Bell-Hillerův stabilizátor 8
2.2.2 Síly působící na vrtulník 8
2.2.3 Režimy letu 9
2.3. Souřadný systém 10
2.2.3 Vztahy mezi souřadnými systémy 11

3. Přenos dat 13
3.1. Teorie informace 13
3.2. Kódování 13
3.2.1 Kódová reprezentace zpráv 14
3.2.2 Elektrická reprezentace zpráv 14
3.2.3 Kódy s paritou 15
3.3. Modulace signálu 15
3.3.1 Přenos v základním a přeloženém pásmu 15
3.3.2 Modulace pro přenos analogových signálů 15
3.3.2.1 Amplitudová analogová modulace 16
3.3.2.2 Šířková impulsová modulace 17
3.3.3 Modulace pro přenos datových signálů 17
3.3.3.1 Amplitudová modulace (ASK) 18
3.4. Přenosové cesty a přenosové kanály 18
3.4.1 Přenosové cesty 18
3.4.1.1 Elektromagnetická přenosová cesta 18
3.4.1.2 Metalická přenosová cesta 19
3.4.2 Přenosové kanály 19
3.4.2.1 Frekvenční rozdělení kanálů 20
3.5. Způsoby přenosu dat 21
3.5.1 Simplexní, duplexní a halfduplexní přenos 21
3.5.2 Sériový, paralelní a sérioparalelní přenos 21
3.5.3 Synchronní, asynchronní a arytmický přenos 22

4. Konstrukce trenažéru a napájecího zdroje 23
4.1. Konstrukce a umístění heliportu v laboratoři 23
4.1.1 Ukotvení heliportu v laboratoři 25
4.1.2 Antigravitační systém 26
4.1.3 Praktické zkušenosti s heliportem 26
4.2. Snímání polohy 26
4.2.1 Charakteristiky inkrementálních snímačů 27
4.3. Snímač otáček hlavního rotoru 28
4.4. Napájecí zdroj 29

4.4.1 Konstrukce zdroje ... 29
4.4.2 Přívod napájení k vrtulníku ... 31
4.4.3 Praktické zkušenosti ... 31

5. Hardwarová část projektu ... 32
5.1. Celková koncepce ... 32
5.2. Řízení pomocí RC soupravy ... 34
5.3. Komunikace PC-interface ... 35
5.3.1 Integrované obvody Xilinx ... 35
5.3.2 Architektura interface ... 35
5.3.3 Čtení a zápis na sběrnici 8080 ... 36
5.3.3.1 Propojení sběrnic ... 37
5.3.3.2 Propojení kabelů ... 37
5.3.4 Adresování na sběrnici 8080 ... 37
5.4. Deska PCB č.1 ... 40
5.4.1 Stav původní desky ... 40
5.4.2 Deska plošných spojů a další úpravy ... 40

6. Softwarová část projektu ... 41
6.1. Analýza původního stavu software ... 41
6.1.1 Propojení software a hardware z hlediska PC ... 41
6.1.2 Program pro ovládání v prostředí Matlab ... 42
6.1.3 Program pro testování modelu v prostředí MS DOS ... 42
6.1.4 Závěry vyplívající z analýzy ... 42
6.2. Řídicí software a komunikace s PCI kartou ... 43
6.2.1 Celková koncepce software a propojení s modelem ... 43
6.2.2 PCI karta ... 45
6.2.2.1 Praktické zkušenosti s kartou ... 46
6.2.3 Programové prostředí a použité toolboxy ... 46
6.2.4 Vzorkovací frekvence a propustnost jádra ... 59
6.2.5 Funkce pro čtení a zápis ... 59
6.2.6 Funkce pro úpravu informací o poloze ... 52
6.2.7 Procedury pro vizualizaci a virtuální model ... 52
6.2.7 Funkce pro diagnostiku a kontrolu hardware ... 54
6.3. GUI aplikace ... 54
6.3.1 Informace o poloze vrtulníku ... 55
6.3.2 Ovládání servomechanismů ... 56
6.3.3 Diagnostika ... 57
6.3.4 Vizializace ... 57
6.3.5 Demonstrační mód ... 58
6.3.6 Praktické zkušenosti s aplikací ... 59

7. Regulace a ukázka letu ... 60
7.1. Provozní podmínky ... 60
7.2. Parametry modelu a způsob regulace ... 60
7.3. Rozvrh akcí a časování ... 61
7.4. Bezpečnost při provozu ... 62
7.5. Přechodové charakteristiky ... 62

8. Závěr ... 65

9. Reference ... 66

Příloha A Výpis funkcí pro řízení hardware 67
A.1 INITrt 67

A.2 WR1 68
A.3 READcidlo 68
A.4 check 69
A.5 INITcitace 70

Příloha B Výpis funkcí pro diagnostiku 71
B.1 test_kanal 71
B.2 test_vnej_reg 71

Příloha C Výpis procedur pro vizualizaci 72
C.1 initvisual 72
C.2 makevisual 72

Příloha D Obsah přiloženého CD 73

2. Model helikoptéry, principy řízení a letu

Tato kapitola se zabývá základními vlastnostmi použitého modelu helikoptéry, objasňuje pojmy důležité pro další úpravy na vrtulníku a samotné řízení.

2.1. Konstrukce a vlastnosti modelu

Náš vrtulník je postaven na sériově vyráběném modelu EP CONCEPT od firmy KYOSCHO. Původní RC souprava od firmy Robbe, která helikoptéru řídila, byla čtyřkanálová, pracující v modelářském pásmu 40Mhz. Vrtulník vychází z klasické koncepce, tedy má jeden nosný rotor s Bell-Hillerovým ovládáním a jeden vyrovnávací rotor umístěný na konci ocasu.

Pohonnou jednotkou je stejnosměrný elektromotor LE MANS AP36. Spalovací motory mají větší užitečnou hmotnost, ale hlučnější provoz a produkují spaliny. Použití elektromotoru je vynuceno umístěním celého zařízení v laboratoři.

Z pohledu regulace představuje vrtulník nelineární mnoharozměrový (MIMO) systém vyššího řádu. Simulace jednodušších modelů je možná a dává výsledky odpovídající reálnému systému, jehož simulaci provádíme. Avšak u tak složitého systému, jakým je vrtulník, zůstává možnost regulovat a měřit na reálné aplikaci neocenitelnou možností při ověřování moderních metod řízení složitých dynamických systémů.

Naším úkolem je upravit stávající model helikoptéry tak, aby jej bylo možno za použití moderních technologií řídit v reálném čase z počítače, data získaná při jeho pohybu analyzovat a současně vizualizovat či uložit pro pozdější zpracování.

Helikoptéra je těžko řiditelný systém, a proto je potřeba zajistit bezpečnost modelu jeho vhodným uchycením na trenažéru (více v kapitole 4), ale také během samotného řízení, implementací bezpečnostních pojistek do ovládacího software. Model je spojen s trenažérem výsuvnou tyčí, na jejíž obou koncích jsou Cardanovy klouby, což umožňuje modelu prakticky volný pohyb v prostoru tvaru kuželu, vztyčeném nad heliportem. Za bezpečnost modelu platíme daň v podobě omezeného pracovního prostoru a též malými změnami jeho dynamických vlastností.

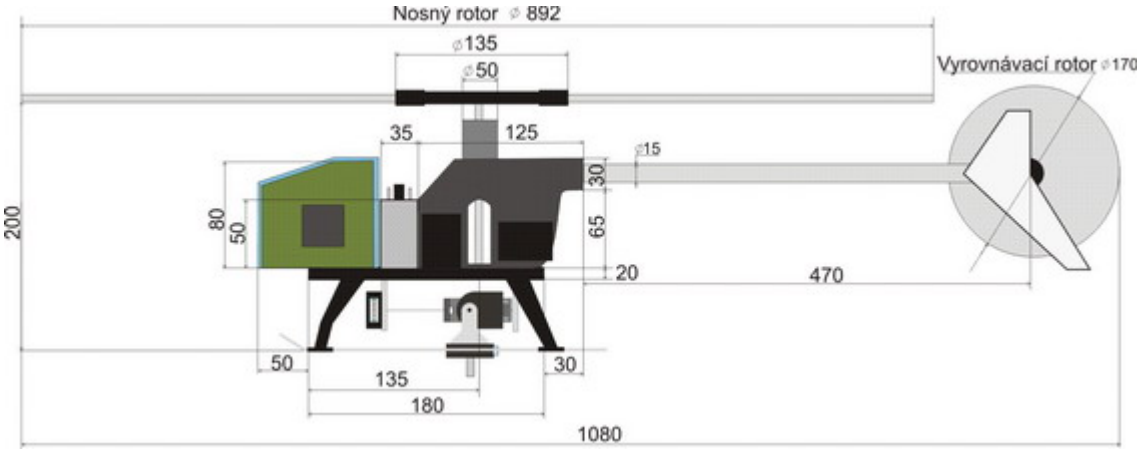
2.1.1 Funkční části vrtulníku

Nejdůležitější části vrtulníku, jak zde budou probrány, jsou na obrázku 2.1. Jedná se o typ vrtulníku s jedním nosným rotorem. Jeho otáčením vzniká na listech nosného rotoru síla, která je nosnou silou celého vrtulníku. Jako důsledek působení odporu prostředí na listy se vytváří reakční moment, jehož působení by vedlo k rotaci trupu helikoptéry, a proto je nutno jej kompenzovat pomocí vyrovnávacího rotoru.

Obr.2.1 Hlavní části vrtulníku

2.1.2 Hlavní a vyrovnávací rotor

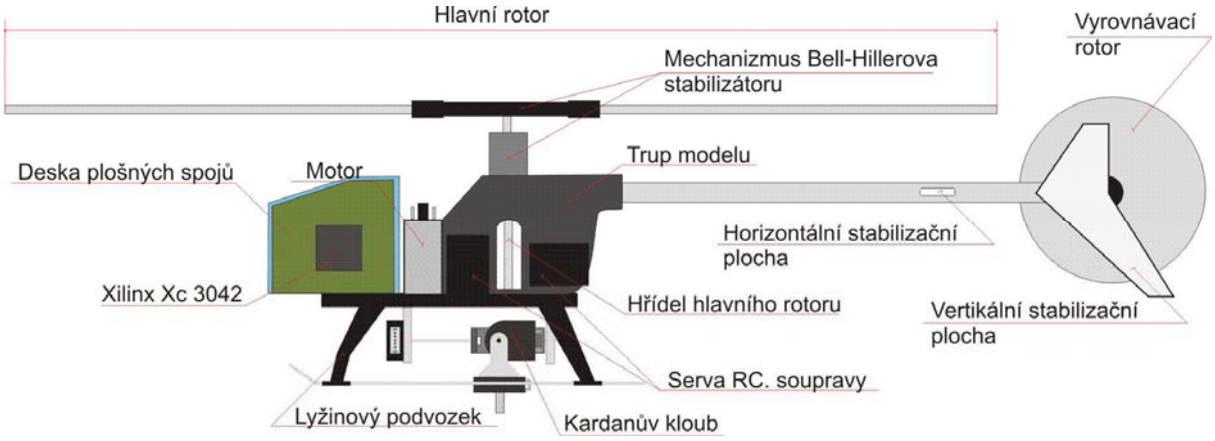
Nosný rotor je dvoulistý a má průměr 892mm. Je opatřen tuhými závěsy rotorových listů a pomocným řídicím rotorem Hillerova stabilizátoru, který tvoří dva krátké listy na koncích opatřené řídicími ploškami. Přes pákové mechanismy lze pomocí tří serv nastavovat cyklické i kolektivní ovládání listů rotoru. Na konci ocasu, který je tvořen duralovou trubkou o délce 450mm a průměru 15mm, je umístěn vyrovnávací rotor. Jeho průměr je 172mm a je poháněn gumovým pásem procházejícím středem ocasní trubky. Listy jsou ovládány lankem spojeným s příslušným servem. Na konci ocasu jsou umístěny vertikální a horizontální stabilizační plochy. Základní rozměry ukazuje obrázek 2.2.



Obr. 2.2 Základní rozměry modelu

2.1.2 Drak modelu a pohonná jednotka

Drak modelu je tvořen základnou, která spojuje nosnou část modelu s ocasní částí a slouží k ukotvení hřídele hlavního rotoru. Velikost trupu je 200mm na délku a 60mm na šířku. V těle základny se nachází převodový systém pro pohon obou rotorů. Ten je přímo spojený s hřídelí hlavního rotoru přes jednosměrnou spojku (tzv. cvrčka), který zaručuje otáčení rotoru i v případě, že dojde k výpadku hnací jednotky. Toto uspořádání umožňuje takzvané přistání autorotací. Elektromotor je umístěn vertikálně v ose trupu. Schematické uspořádání jednotlivých prvků ukazuje obrázek 2.3.



Obr. 2.3 Základní uspořádání

2.1.2 Systémy pro řízení a přistávání

Natáčení listů vrtule je realizováno pomocí akčních členů, tvořených čtyřmi modelářskými servy, umístěnými po dvou na obou stranách vrtulníku. Řízení je zajištěno přes původně čtyřkanálovou RC soupravu ATTACK-4 firmy Robbe, pracující na frekvenci 40,695Mhz, která je doplněna pátým kanálem pro pulsně-šířkové řízení výkonu motoru. Helikoptéra nese přijímač o rozměrech 50x60mm, umístěný před motorem na pravé straně.

Interface pro komunikaci s počítačem je realizován dvěma obvody 8253, které spoluvytvářejí pulsně-šířkovou modulaci pro serva. Tento signál je zaveden do původní RC soupravy před AM modulátor. Pro softwarové řízení je použit program Matlab doplněn o Realtime Toolbox. Informace o poloze vrtulníku jsou s pomocí IRC snímačů, umístěných na trenažéru a pod vrtulníkem, předávány ke zpracování do obvodů XLINX a poté posílány přes komunikační interface do počítače. Podrobnější informace jsou v kapitole 6. Jako přistávací zařízení slouží dvě ližiny vyrobené z duralových trubek o průměru 4mm, spojené s trupem nosníky z pružného plastu.

2.2 Princip řízení vrtulníku

Helikoptéru lze ovládat změnou směru a velikosti vektoru tahu u hlavního a vyrovnávacího rotoru. Změnou velikosti tahu vyrovnávacího rotoru korigujeme momentovou nerovnováhu, způsobující rotaci vrtulníku kolem hlavní osy. Naopak změnou tahu nosného rotoru docilujeme stoupání a klesání stroje. Při korekcích tahu se nastavují všechny listy společně, proto se toto řízení nazývá kolektivní. Výkon motoru lze také ovládat, ale jeho změna se nepoužívá k přímému ovlivňování letu, a proto je obvykle udržován v oblasti s největší účinností.

Náklon vrtulníku v určené ose lze ovlivnit změnou náklonu listů v průběhu jedné otáčky, což vede ke změně směru vektoru tahu. K této změně úhlu náběhu dochází cyklicky v průběhu otáčení rotoru,

nazývá se proto cyklické řízení. Pohled na kolektivní a cyklické řízení je na obrázku 2.4a, respektive 2.4b

2.2.1 Bell-Hillerův stabilizátor

Z důvodu přirozeně špatné stability helikoptéry během většiny letových režimů jsou vrtulníky vybaveny stabilizátory, jejichž úkolem je tlumit naklánění vektoru tahu nosného rotoru. Jsou umístěny mezi pilota a cyklické řízení. Z mechanických stabilizačních systémů je nejznámější Bellův –Hillerův. Podstata spočívá v použití mechanického setrvačníku a pomocných řídicích plošek, kterými je vybaven hlavní rotor. Bell-Hillerův stabilizátor v modelu je na obrázku 2.4.

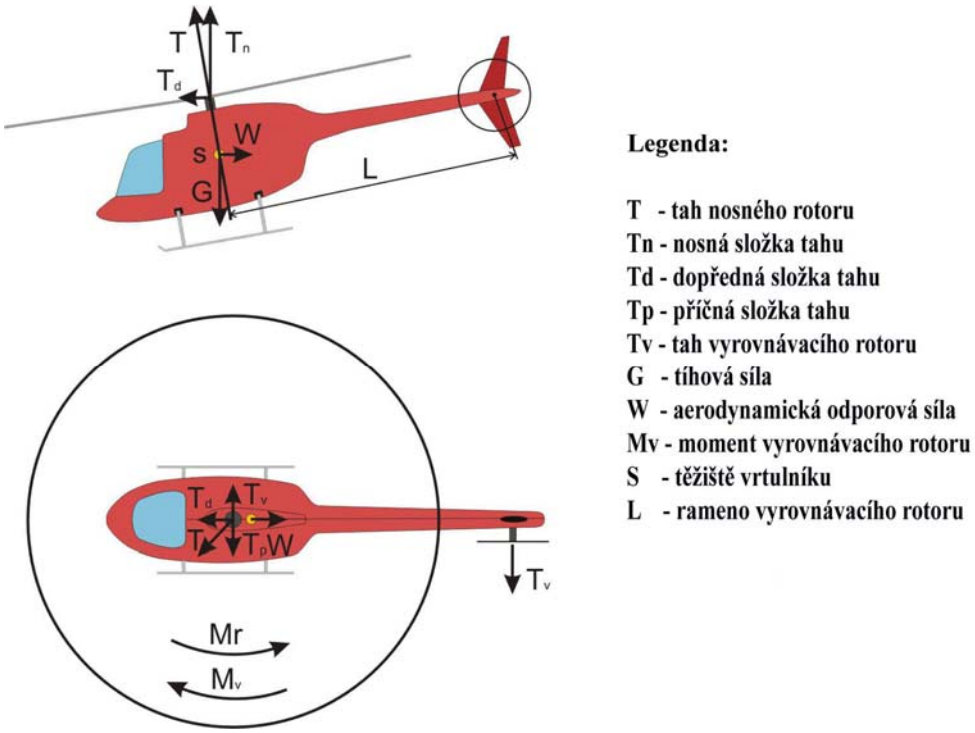
Obr. 2.4a Cykliky a stabilizátor-boční pohled

Obr. 2.4b Cykliky-horní pohled

2.2.2 Síly působící na vrtulník

Na vrtulník působí řada sil a momentů, které musí být akčními členy vrtulníku vhodně kompenzovány, aby byl schopen letu. Hlavní sily působící na vrtulník při ustáleném letu jsou schematicky zobrazeny na obr. 2.5.Tah nosného rotoru se rozkládá na nosnou a posuvnou složku.

Moment vyvolaný otáčením hlavního rotoru způsobuje rotaci trupu v opačném směru a musí být kompenzován tahem vyrovnávacího rotoru Tv. Ten však zároveň způsobuje naklánění vrtulníku na stranu a musí být vyrovnáván vhodným směrováním vektoru tahu hlavní vrtule. Řízení vrtulníku je obecně v porovnání s řízením letadla velmi náročný úkol, kde se jednotlivé řídicí prvky značně ovlivňují a řízení jsou si podobná pouze při rychlém pohybu helikoptéry vpřed.



Obr. 2.5 Síly působící na helikoptéru v ustálením letu

2.2.3 Režimy letu

Použitý model využívá všech výše popsaných prvků řízení a umožňuje v závislosti na použitém trenažéru následující letové režimy:

Start

Letový režim nutný pro vyřešení všech následujících úloh. Při plném výkonu motoru zvyšujeme natočení listů hlavního rotoru a tím i jeho tah směrem vzhůru vzrůstá, čímž vrtulník vzlétá. Postupným zvyšováním výkonu motoru a následnou úpravou kolektivního řízení se model dostává do vzduchu. Vliv přízemního jevu, který je způsoben odlišným prouděním vzduchu v blízkosti země, lze v případě malé, nad zemí umístěné plošiny, prakticky zanedbat.

Visení

Vrtulník setrvává na jednom místě a udržuje si konstantní výšku. Tento způsob letu spolu s režimem otáčení ve vodorovné poloze by vzhledem k možnostem modelu a trenažéru měl představovat nejčastější úlohu regulace.

Přímý let a let do stran

Pomocí změny cykliky přejde vrtulník do pohybu ve směru náklonu. Rychlost letu je určena mírou náklonu a délka dráhy letu je omezena rozměry vodící tyče. Proto je tento letový režim možný pouze na krátké dráze.

Otáčení ve vodorovné poloze

Jedná se o režim, při kterém je vrtulník ve vodorovné poloze a řízením tahu vyrovnávacího rotoru rotuje kolem osy hlavního rotoru. V počátečních fázích návrhu regulátorů lze usnadnit řízení v tomto režimu částečnou fixací vodicí tyče, čímž se výrazně zlepší stabilita modelu.

Plynulá zatáčka během letu

Provádí se z režimu přímého letu při současné změně cykliky a tahu zadní vrtule. Konstrukce trenažéru umožňuje v tomto režimu především pohyb po kruhové trajektorii.

Přistání

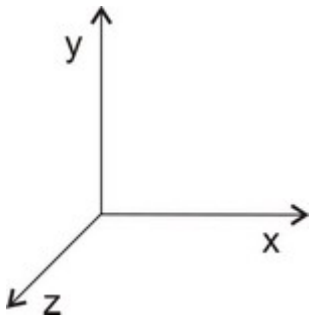
Operace opačná ke startu modelu. Také zde by se mohl uplatňovat přízemní jev v případě větší přistávací plošiny.

Přistání autorotací

Tato možnost je obdobou nouzového přistání u letadel a používá se v případě vysazení motoru. Vzhledem k tomu, že pro tuto operaci je potřeba určitá výška, nelze ji s ohledem na délku vodicí tyče trenažéru vyzkoušet.

2.3 Souřadný systém

Pro pohyb letadla se používá různých soustav souřadnic.V našem případě se nejedná o navigaci, spíše o studium dynamiky pohybu, proto nebudeme při výpočtech zohledňovat zaoblení Země. Tyto systémy jsou navzájem provázané, popisují polohu letadla v prostoru a vůči proudu vzduchu. Používají se pravoúhlé souřadné soustavy podle normy GOST nebo ISO. V dalších odvozeních budeme uvažovat normu GHOST, jejíž orientace je na obrázku 2.6.



Obr. 2.6 Orientace-norma GHOST

Pro popis letu vrtulníku používáme tři navzájem provázané souřadné systémy:

Zemský souřadný systém

Zde je rovina x_g, z_g rovnoběžná s vodorovným povrchem Země a je umístěna na přistávací ploše heliportu tak, že počátek souřadné soustavy je ve středu kruhové desky a osa z_g je rovnoběžná se spodní částí nosné konstrukce heliportu. Osa y_g pak odpovídá výšce nad plochou heliportu.

Letová soustava souřadnic

Počátek souřadné soustavy se nachází v těžišti vrtulníku, osa x_l je orientována od zádě k přídi vrtulníku, osa y_l směřuje kolmo vzhůru a osa z_l vytváří točivou souřadnou soustavu.

Aerometrická souřadná soustava

Je závislá na vektoru vzduchu obtékajícího vrtulník. Osa x_a je orientována vodorovně s vektorem vzduchu její orientace je však opačná, osy y_a, z_a jsou orientovány k ose x_a stejně jako u letadlové souřadné soustavě.

2.3.1 Vztahy mezi souřadnými soustavami

Díky transformačním vztahům mezi jednotlivými soustavami lze přepočítat již známou polohu objektu v soustavě jedné do soustavy druhé. Postup vytváření transformačních matic vychází z Heidelbergovy notace a lze jej použít mimo jiné i pro výpočet polohy vrtulníku vůči trenažéru nebo výpočtech pro vizualizaci.

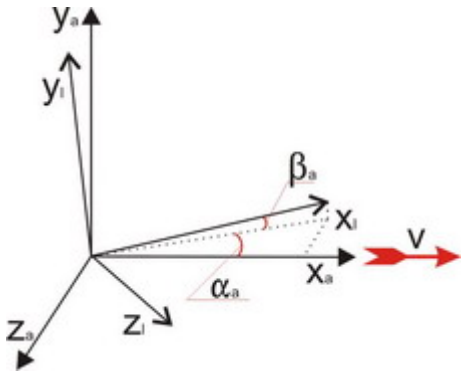
Transformace souřadnic mezi letadlovým a aerometrickým souřadným systémem

Vztah mezi těmito soustavami je zobrazen na obr. 2.7 a vyjadřuje aerodynamické podmínky za letu vzhledem k letounu. Aerodynamická soustava se popisuje pouze třemi parametry, z tohoto důvodu není transformace úplná, jak je tomu v předešlém případě. Parametry aerometrické soustavy jsou:

β_l ***úhel náběhu:*** úhel svíraný vektorem vzdušné rychlosti s osou letadla x_l v rovině symetrie x_l, y_l.

α_a ***úhel vybočení:*** úhel dán vektorem vzdušné rychlosti s osou x_l v rovině křídel x_l, z_l

v_l ***vzdušná rychlost:*** absolutní hodnota rychlosti letounu vůči okolnímu vzduchu



2.7 Transformace souřadnic, letový aerometrický systém

Rovnice popisující vztah letadlové a aerometrické souřadné soustavy v homogenních souřadnicích:

$$x_a = \begin{bmatrix} \cos\alpha_a & 0 & -\sin\alpha_a & 0 \\ 0 & 0 & 0 & 0 \\ \sin\alpha_a & 0 & \cos\alpha_a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta_a & -\sin\beta_a & 0 & 0 \\ \sin\beta_a & \cos\beta_a & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_l$$

Transformace souřadnic mezi zemským a letadlovým souřadným systémem

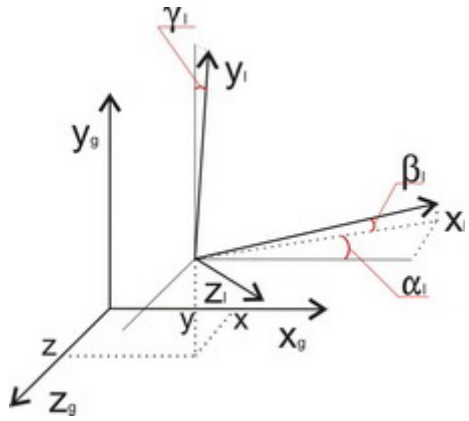
Tato transformace udává obecnou polohu vrtulníku v prostoru, vztaženou k zemské soustavě souřadnic. Poloha letadla je obecně udávána šesti parametry:

x_l, y_l, z_l ***poloha vzhledem k těžišti vrtulníku***

α ***kurz vrtulníku*** (osa rotace y_l)

β ***podélný náklon*** (osa rotace z_l)

γ ***příčný náklon*** (osa rotace x_l)



Obr. 2.8 Transformace souřadnic, zemský letadlový souř. syst.

Transformace z letadlového souřadného systému v homogenních souřadnicích do zemského souřadného systému v homogenních souřadnicích je dána vztahem:

$$x_g = \begin{bmatrix} \cos\alpha_l & 0 & -\sin\alpha_l & 0 \\ 0 & 0 & 0 & 0 \\ \sin\alpha_l & 0 & \cos\alpha_l & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta_l & -\sin\beta_l & 0 & 0 \\ \sin\beta_l & \cos\beta_l & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \cos\gamma_l & -\sin\gamma_l & 0 \\ 0 & \sin\gamma_l & \cos\gamma_l & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_l \quad (2.1)$$

kde $x_l = [x_l \quad y_l \quad z_l \quad 1]$ a $x_g = [x_g \quad y_g \quad z_g \quad 1]$

3. Přenos dat

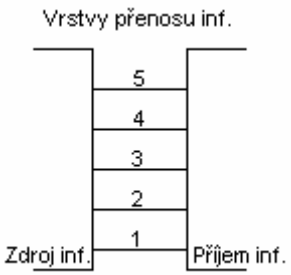
Ke komunikaci s modelem helikoptéry se používá bezdrátové a metalické přenosové cesty. Považuji proto za důležité uvést základní informace o těchto metodách přenosu informací, jež jsou nedílnou součástí měřících a řídicích systémů. Bezdrátová přenosová cesta je použita u povelového řízení, elektromagnetická cesta realizuje přenos dat z modelu helikoptéry do interface, respektive do PC.

3.1 Teorie informace

Základy teorie informace definoval C:E:Shannon.Jedná se o analýzu problematiky přenosu dat včetně poruch, parametrů kanálů i přenášeného signálu, definuje kvantitativní míru informace. Základní myšlenkou teorie informace je způsob sběru, přenosu a zpracování informace. Přenos informace lze rozdělit do pěti vrstev (obr.3.1) :

1.vrstva
Statistika: Shannonova teorie používá statistické zpracování. Dovoluje popisování vlastností informačních znaků a jejich kvalifikaci.

2.vrstva
Syntaxe : řeší se následující problémy:
- jaké soubory symbolů jsou nadefinovány
- jaké soubory jsou definovány pro jednotlivé vrstvy
- jak jsou slova řazena do vět



Obr. 3.1 Vrstvy přenosu informace

3.vrstva
Sémantická: Informace se může přenášet různými médii, podle nich se volí typy kódů.

4.vrstva
Pragmatická: řeší způsob řízení přenosu informace mezi vysílačem a přijímačem.

5.vrstva
Apoetická: Vrstva je nejvyšší, zajišťuje cíleně orientované programové vybavení.

3.2 Kódování

Kódováním můžeme nazvat přiřazení stavu prvků jedné množiny (zprávy) stavu prvků druhé množiny (signálu). Z hlediska přenosu zpráv nám kódování definuje jak elektrickou reprezentaci, tak přizpůsobení signálu přenosovému kanálu, zlepšení synchronizačních vlastností, ale na vyšší úrovni i zabezpečení proti chybám a zneužití dat.

Zprávu kódujeme v kodéru, který je umístěn na straně vysílače. Na straně přijímače je dekodér, jenž sadě znaků kódované zprávy přiřadí sadu znaků původní zprávy. Pokud je zpráva zabezpečena korekčním kódem, může být dekodérem při chybně přijaté zprávě provedena oprava v rámci zabezpečení.Ke kódování se používá velká řada kódů, které mají různé vlastnosti, proto je obvyklé používat vícenásobné kódování, kdy je zpráva například nejdříve zabezpečena proti chybám a poté upravena pro přenos.

3.2.1 Kódová reprezentace zpráv

Při kódování vycházíme z množiny prvků zdroje zpráv, kterým kodér přiřazuje prvky množiny přenosového systému, představující kódová slova. Pro zajištění jednoznačnosti musí být množina prvků zdroje zpráv menší nebo rovna množině prvků kodéru.
Celkový počet kódových slov L vypočteme podle vztahu (3.1), kde n je délka kódového slova a m je počet prvků v příslušné abecedě.

$$L = m^n \tag{3.1}$$

Rozeznáváme dvoustavové (m=2), třístavové (m=3) a vícestavové kódy (m >3), v závislosti na objemu abecedy kódu **m**. V dnešní době se používá dvoustavový nebo vícestavový kód, potřebný počet slov se dosahuje změnou délky kódového slova.

Dělení kódů:
1) Podle počtu prvků v kódové kombinaci

Jednoprvkové kódy
Kódové slovo obsahuje pouze jeden prvek. Používají se v řízení a snímání s málo stavy.

Více prvkové kódy
Kódové slovo má více prvků, nejvíce se používají kódy se dvěma stavy (binární). Binární kódy přímo reprezentují data v počítači.

2) Podle délky kódových skupin

Nerovnoměrné kódy
Jsou charakterizovány různou délkou kódových skupin. Jsou navrženy tak, aby přenos byl maximálně efektivní. Nejpoužívanější znaky mají nejkratší délky. Jejich nevýhodou je složitější detekování.

Rovnoměrné kód
Vyznačují se rovnoměrnou délkou kódových skupin. Na úrovni přenosového kanálu se vždy využívá tohoto kódu a komprese se nechává na vyšší úrovni kódování zpráv.

3.2.2 Elektrická reprezentace

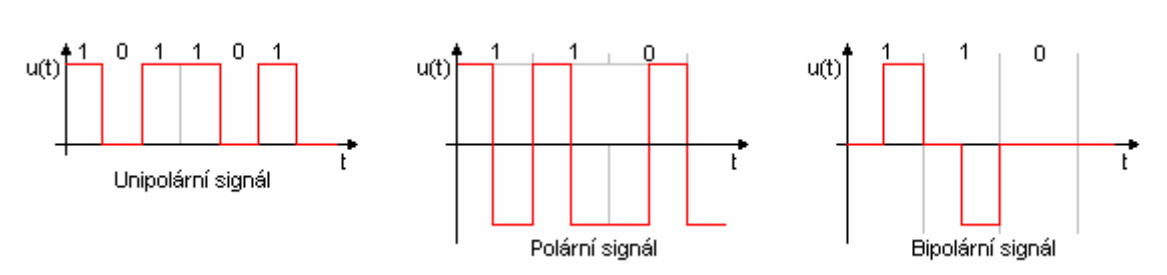
Každý signál přenášející informace má svou elektronickou reprezentaci. Tato je důležitá zvláště u binárních signálů. Můžeme ji rozdělit následujícím způsobem:

Unipolární signál
Jde o signál jedné polarity, označované také jako signály RZ (return to zero). Používá se při přenosech na krátké vzdálenosti. Nevýhodou těchto signálů je velikost stejnosměrné složky

Polární signál
Obsahuje pouze kladnou nebo zápornou polaritu, proto se nazývají NRZ (Non Return to Zero). Střídáním kladné a záporné polarity se výrazně potlačí stejnosměrná složka. Nevýhodou je potřeba většího přenosového pásma a nutnost vložení synchronizační informace.

Bipolární signály

Kombinují výhody předchozích signálů. Používají tři napěťové úrovně (kladná, nulová a záporná amplituda)



Obr. 3.2 Typy elektrických signálů

3.2.3 Kódy s paritou

Představují detekční kódy systematické. Vyznačují se přesně definovanou informační částí přenášeného bloku (rámce) a částí zabezpečovací. Ochrana paritou je velmi jednoduchý a často využívaný způsob zabezpečení. Vlastní zabezpečení kódového slova vzniká tak, že každý k-prvkový soubor se doplní jedním prvkem , aby počet prvků log1 byl sudý nebo lichý. Hovoříme tak o sudé nebo liché paritě. Pro sudou paritu platí, že součet bitů mod 2 všech prvků včetně paritního je 0, pro lichou je mod2 = 1. Paritou lze detekovat pouze lichý počet chyb. Oprava probíhá opětovným posláním chybné zprávy. Výhodou zabezpečení s paritou je jednoduchost řešení kodéru a dekodéru a možnost zabezpečit libovolně dlouhou skupinu prvků.

3.3 Modulace signálu

Modulace představuje nástroj umožňující přenášet, upravovat a pracovat se signálem způsobem, který jeho původní vlastnosti nedovolují. Tato metoda je často používaná v mnoha technických aplikacích.

3.3.1 Přenos v základním a přeloženém pásmu

Bezdrátové přenosové cesty většinou nedovolují přenášet signál v základním pásmu, které generuje zdroj signálu. Důvodem je požadavek na využití více přenosových kanálů, případně technické problémy s přenosem v některých frekvenčních oblastech. Modulaci používáme právě v těch případech, kdy není možné přenést signál v základním pásmu. Signál ze zdroje (modulační signál) vstupuje společně s nosným signálem do modulátoru. Modulačním signálem jsou v modulátoru řízené některé parametry (frekvence, amplituda …) nosného signálu. Výsledkem modulace je transformace frekvenčního pásma signálu, kde po potlačení základního pásma vznikne signál v přeloženém pásmu (modulovaný signál). Inverzní úloha se řeší v demodulátoru, který z přijatého modulovaného signálu zrekonstruuje signál modulační.

Přenosu v přeloženém pásu se hojně využívá i v metalických přenosových cestách, kde umožňuje vytvořit více přenosových kanálů.Rozeznáváme modulaci pro přenos analogových a digitálních signálů. Ke každému typu uvedu pro názornost jednu metodu.

3.3.2 Modulace pro přenos analogových signálů

Analogovým signálem rozumíme signál spojitý v čase i amplitudě. Tímto signálem je modulován nosný signál, který může být harmonický nebo impulsní. Impulsní modulace se hojně využívají v řídicí a měřící technice. Modulační signál mění u nosného signálu některý z jeho parametrů (amplituda, frekvence, fáze, šířka pulzu), mluvíme tak o amplitudové, frekvenční, fázové a

šířkové modulaci. První tři způsoby se využívají u modulace analogové i impulsní. O šířkové modulaci mluvíme v souvislosti s impulsní modulací.

Samostatnou kapitolu tvoří modulace zajišťující číslicové kódování analogových signálů a jeho digitální přenos. Mezi tyto druhy modulace patří například impulsová kódová modulace nebo delta modulace.

3.3.2.1 Amplitudová analogová modulace

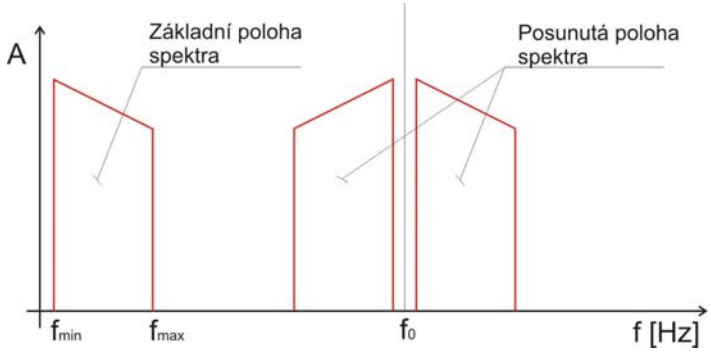
Nosným signálem je harmonický signál, u kterého modulačním signálem měníme amplitudu.V modulátoru se provádí prostý součin modulačního a nosného signálu. Násobením signálů dostaneme ve frekvenční oblasti posunutí spektra modulačního kolem frekvence nosné (obr.3.3). Tomuto případu se říká lineární modulace.
Princip amplitudové modulace lze vyjádřit vztahem:

$$a_A(t) = \left[A + \Delta A \frac{x(t)}{Xm} \right] \cos(\omega_0) = A[1 + m_A g(t)] \cos(\omega_0) \tag{3.2}$$

kde: $a_A(t)$ je amplitudově modulovaný signál

$a(t)$	nosný signál, pro AM platí $a(t) = A\cos(\omega_0 t)$
$x(t)$	modulační signál
A	amplituda nosného signálu
ΔA	maximální změna amplitudy nosného signálu
Xm	maximální hodnota modulačního signálu
m_A	činitel amplitudové modulace ($m_A = \Delta A / A$)
$g(t)$	normovaný modulační signál $x(t)$ $g(t) = x(t) / Xm$

Popsaný amplitudově modulovaný harmonický signál $a_A(t)$ zaujímá frekvenční pásmo dané dvojnásobkem maximálního kmitočtu frekvenčního spektra modulačního signálu.Výkon signálu je rozdělen tak, že rozhodující část připadá na složku s kmitočtem nosného signálu a menší část zbývá na obě postranní pásma. Vzhledem k tomu, že nosný signál nenese žádnou informaci, je toto rozdělení značně nevýhodné.



Obr. 3.3 Frekvenční pásmo amplitudové modulace

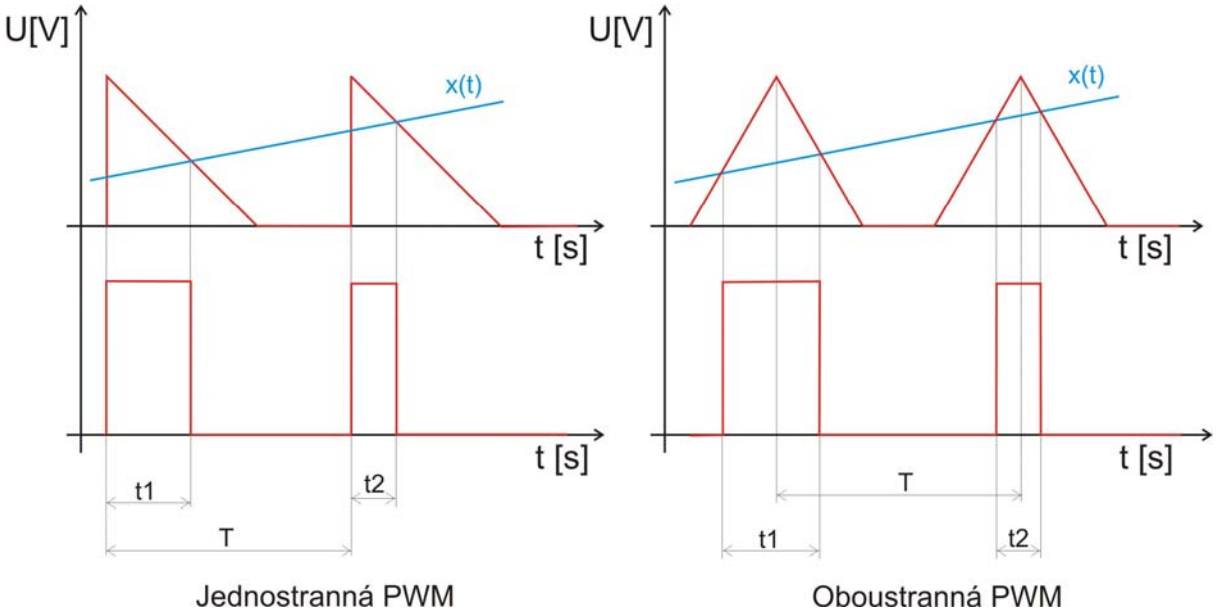
Snahy o zlepšení výkonové bilance a zúžení potřebného frekvenčního pásma vedly k různým variantám amplitudové modulace. Mezi důležité zástupce patří modulace DSB (double side band), při ní jsou plně vysílaná postranní pásma při potlačené nosné. Za cenu složitější modulace a následně

demodulace lze potlačit i druhé postranní pásmo, které nese stejnou informaci jako pásmo první. Této modulaci se říká modulace SSB (single side band). Je jedno, které postranní pásmo je potlačeno, při demodulaci je to ale třeba zohlednit. V obou případech je potlačená nosná, nutno však podotknout, že 100% potlačení nosné nikdy nenastane.

3.3.2.2 Šířková impulsová modulace (PWM)

Jedná se o analogovou impulsovou modulaci. Impulsová modulace převádí spojitý analogový signál na signál diskrétní ve tvaru sledu pulzů, jejichž charakteristické parametry se mění příslušným typem modulace. Impulsové modulace jsou odolnější proti vlivu rušení, což je vykoupeno nutností přidělení větší šířky přenášeného frekvenčního pásma přenosového kanálu.

U PWM modulací měníme pouze šířku impulzů, jejich amplituda , frekvence a fáze zůstává zachována. Pokud se šířka pulzu mění jednostranně vzhledem k určujícímu bodu, jedná se o jednostrannou PWM. Při oboustranné šířkové modulaci se šířka pulzu mění symetricky oproti určujícímu bodu (obr.3.4). PWM modulace se s výhodou používá k přenosu stejnosměrných signálů a signálů s nízkým kmitočtem, a to často ve spojení s některou ze základních modulací. Takové modulaci, kdy je modulovaný signál použit jako modulační signál v jiné modulaci, se říká vícenásobná modulace.



Obr. 3.4 PWM modulace

3.3.3 Modulace pro přenos datových signálů

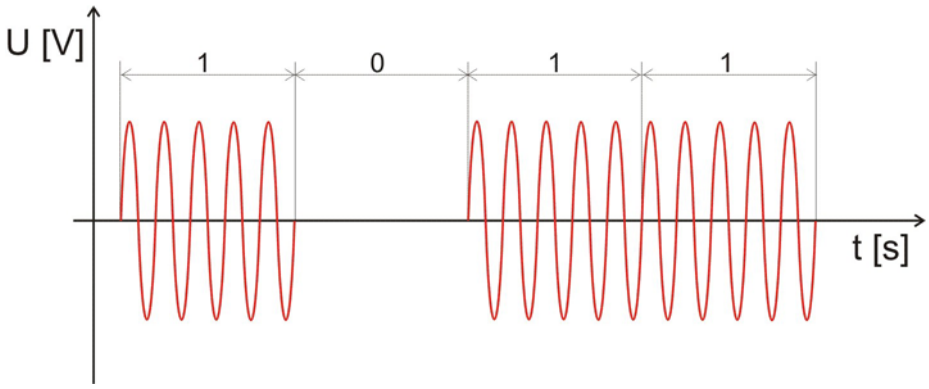
Modulace pro přenos číslicových dat jsou odvozeny od analogových modulací. Daty v tomto kontextu rozumíme především binární soubory, proto výše uvedená modulace je vhodná pro přenos sériových dat. V praxi je možný přenos binárních dat buď přímo tzv. intenzivní metodou nebo pomocí modulace. V bezdrátové přenosové cestě je intenzivní způsob přenosu prakticky nemožný, proto se využívá některá z modulací.

3.3.3.1 Amplitudová modulace (ASK)

Amplitudová modulace patří mezi nejjednodušší druhy modulací pro přenos číslicového signálu. Harmonický nosný signál je amplitudově modulován datovým signálem, který nabývá diskrétních hodnot (pro binární signál obvykle log0 a log1). V praxi je rozšířené tzv. dvoustavové

kódování – signál je , signál není. Podle typu logiky tyto stavy odpovídají log1 a log0. Jde o ASK s hloubkou modulace 100%.

Její největší výhodou je jednoduchost modulace i detekce (demodulace) na přijímací straně. Náchylnost k poruchám je její největší slabinou. Zlepšení odolnosti se dociluje použitím menší hloubky modulace, při které nedochází k přerušení signálu. Z pohledu detekce je však tato metoda méně výhodná. I přes malou odolnost proti rušení je ASK pro svou jednoduchost často využívaná.



Obr. 3.5 Amplitudová modulace ASK

Frekvenční spektrum ASK odpovídá frekvenčnímu spektru radioimpulzu. Pro přenos je potřeba přibližně $\Delta F = 2/\tau$, kde τ je délka nejkratšího přenášeného impulzu.

3.4 Přenosové cesty a přenosové kanály

Pod pojmem přenosová cesta definujeme fyzikální prostředí, ve kterém se uskutečňuje přenos údajů nebo zpráv pomocí měronosné veličiny. Měronosná veličina je materiálním nosičem přenášené zprávy v prostoru a čase, tedy libovolná fyzikální veličina, jejíž informační parametr se v čase mění souhlasně s přenášenou zprávou.

Pro efektivnější využití přenosové cesty členíme časovou i frekvenční oblast do několika částí. Každou takovou částí můžeme přenášet jednu měronosnou veličinu. Takový frekvenční nebo časový segment nazýváme přenosovým kanálem.Z fyzikálního hlediska přenosová cesta a přenosový kanál spolu úzce souvisí. Z hlediska uživatele je přenosová cesta nezajímavá, uživatele zajímají pouze parametry přenosových kanálů jako je šířka přenosového pásma, možnosti rušení atd.

3.4.1 Přenosové cesty

Fyzikální prostředí přenosových cest dělíme podle použitého média, frekvenčního pásma a charakteru prostředí. Z pohledu této práce je nejzajímavější metalická přenosová cesta a elektromagnetická přenosová cesta.

3.4.1.1 Elektromagnetická přenosová cesta

Elektromagnetické přenosové cesty zajišťují přenos informace bez přímého propojení metalickým nebo optickým kabelem. K přenosu jsou využívány elektromagnetické vlny. Přenosová cesta je vytvářena v prostoru mezi vysílací a přijímací anténou. Vyzářená energie je vázána na elektromagnetické pole vytvořené v prostoru vysílací antény. Šíření elektromagnetických vln je závislé na délce vlny a prostředí, kterým vlna prochází. Parametry prostředí jsou nepravidelné a také

náhodné. Elektromagnetické cesty pro bezdrátový přenos se dělí podle použitého frekvenčního spektra(obr. 3.6, obr. 3.7).

V automatizačních systémech se užívá telemetrických systémů pro sběr dat v pásmu VKV (30-200 MHz). Zvláštní kapitolu zaujímají družicové spoje, které se s rozvojem techniky stále více prosazují. Nejvíce se používá přenosové pásmo KU, pro které je vyhrazeno pásmo 14 GHz pro přenos k satelitu a pásmo 12 Ghz pro přenos zpět.

3.4.1.2 Metalická přenosová cesta

Metalické přenosové cesty jsou nejčastějším přenosovým médiem pro přenos měronosných signálů. Pro měřící a řídicí systémy využíváme celé spektrum metalických přenosových cest, ať jsou to koaxiální kabely nebo kroucené dvoulinky pro přenos na dlouhé vzdálenosti nebo rovné vodiče hodící se pro střední a krátké vzdálenosti.

Každá metalická přenosová cesta je charakterizována tzv. primárními parametry vztaženými na jednotku délky:

Odporem $R_0 \left[\Omega / km\right]$

Indukčností $L_0 \left[H / km\right]$

Kapacitou $C_0 \left[F / km\right]$

Vodivostí $G_0 \left[S / km\right]$

Z primárních parametrů lze vypočítat vlnovou impedanci Z_o (3.3) a měrnou míru přenosu γ (3.4). Hodnota α u měrné míry přenosu udává poměrný útlum charakterizující zmenšování amplitudy signálu v závislosti na vzdálenosti od počátku vedení. Imaginární část β určuje poměrný fázový úhel. Pomocí poměrného fázového úhlu a úhlové frekvence ω určíme fázovou rychlost $v_f = \omega / \beta$.

$$Z_o = \sqrt{\frac{R_0 + j\omega L_0}{G_0 + j\omega C_0}} \tag{3.3}$$

$$\gamma = \alpha + j\beta = \sqrt{\left(R_0 + j\omega L_0\right) + \left(G_0 + j\omega C_0\right)} \tag{3.4}$$

Útlum na celé délce vedení vypočítáme vynásobením poměrného útlumu α délkou vedení. Stejným způsobem určíme i fázový rozdíl. Abychom dosáhli efektivního přenosu, musíme výstupní impedanci zdroje signálu přizpůsobit vstupní impedanci vedení. Totéž platí na straně přijímače.

3.4.2 Přenosové kanály

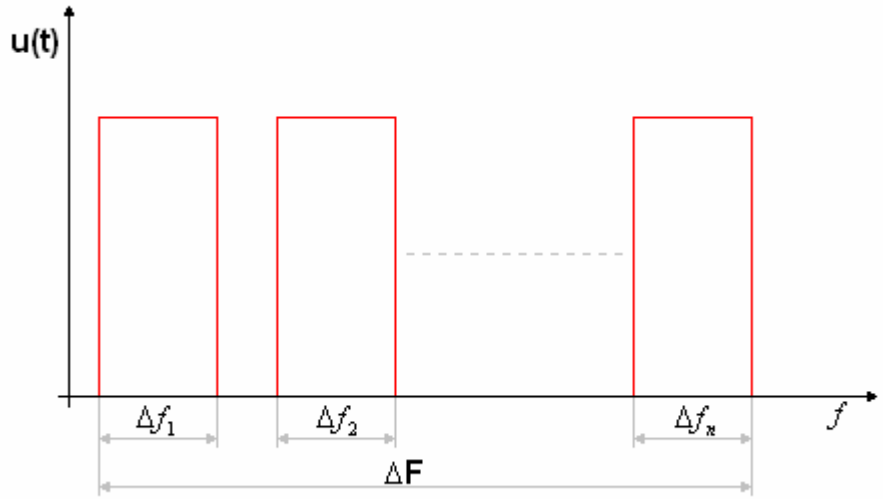
V mnoha reálných aplikacích je ekonomicky nevýhodné používat pro každá spojení mezi vysílačem a přijímačem jednu přenosovou cestu. Využíváme proto jednu kvalitní přenosovou cestu se současným vícenásobným využitím - to vede k vytvoření přenosových kanálů.

Princip umožňující při přenosu signálů používat přenosové kanály se nazývá sdružování. Technické zařízení, které sdružuje signály do jednoho, se nazývá multiplextor. Demultiplextor rozděluje přijatý signal z multiplextoru na jednotlivé složky.

Při používání bezdrátové cesty lze pohlížet na okolní prostředí jako na jednu přenosovou cestu, která je frekvenčně rozdělená na jednotlivé přenosové kanály.Rozdělení přenosové cesty na jednotlivé kanály lze realizovat mnoha způsoby. Mezi nejpoužívanější patří časové a frekvenční dělení , které má, jak jsem již zmínil, uplatnění u bezdrátového přenosu.

3.4.2.1 Frekvenční rozdělení kanálů (FDMA)

Při frekvenčním rozdělení kanálů jsou jednotlivým signálům navzájem přiřazena nepřekrývající se frekvenční pásma $\Delta f_1, \Delta f_2$ až Δf_n. Spektra přenášených signálů $u_k(t)$ příslušných kanálů nesmí překročit stanovené meze Δf_k. Počet kanálů n se stanoví podle šířky spektra přenosové cesty ΔF a požadované šířky jednotlivých kanálů Δf s příslušnou frekvenční rezervou mezi dvěma sousedícími kanály, která potlačuje přeslechy mezi sousedními kanály (obr.3.6). V praxi se však překrývání pásem běžně vyskytuje.



Obr. 3.6 Přiřazení frekvenčních spekter jednotlivým kanálům

Signál, který chceme přenášet v k kanálu, upravujeme frekvenční nebo amplitudovou modulací na nosnou frekvenci, která je v polovině frekvenčního rozsahu k kanálu.

Obr. 3.7 Frekvenční kanály na časové ose

Velkou výhodou frekvenčního dělení kanálů je možnost současného vysílání signálů patřících různým kanálům. Jednotlivé vysílače a přijímače nemusí být při tom soustředěny na jednom místě, ale mohou být připojeny na jakémkoliv místě přenosové cesty. Nedostatkem je vzájemný vliv kanálů, vznikající překrýváním spekter signálů, které zasahují do sousedního pásma.

3.5 Způsoby přenosu dat

Na tomto místě chci uvést základní přehled terminologie používané v přenosu dat u povelového řízení modelu helikoptéry a sběru dat.

3.5.1 Simplexní, duplexní a halfduplexní přenos

Simplexní přenos
Přenos dat a řídicích znaků je realizován pouze jedním směrem, tím odpadá možnost zpětného potvrzení přijatých dat. Datový spoj je jednosměrný, nevyžaduje dva přenosové kanály. Simplexní přenos je vhodný pro sběr dat.

Halfduplexní přenos
Přenos dat a řídicích signálů ve směru od vysílače k přijímači je shodný se simplexním přenosem, v opačném směru se po samostatném kanálu přenášejí řídicí signály.

Duplexní přenos
Je charakterizován tím, že přenos probíhá oběma směry současně. Přenosové kanály jsou využívány pro přenos dat a řídicích signálů současně, tím dosáhneme velký přenosový výkon.

3.5.2 Sériový, paralelní a sérioparalelní přenos

Podle způsobu přenosu jednotlivých bitů v kódových skupinách rozlišujeme sériové přenosy na:

Sériový přenos
Jednotlivé bity jsou v časové oblasti přenášeny postupně. Díky této vlastnosti stačí pro uskutečnění přenosu jeden přenosový kanál.

Paralelní přenos
Všechny bity kódového slova jsou přenášeny současně. Pro každý bit je potřeba samostatný přenosový kanál.

Sérioparalelní přenos
Kombinace předchozích způsobů.

3.5.3 Synchronní, asynchronní a arytmický přenos

Na přenášená data působí v komunikačních kanálech vlivy způsobené nedokonalostí vedení (např. útlum, zpoždění) a rušením od okolí. Přenášený signál je na přijímací straně těmito vlivy zkreslen, proto je pro přijímač při přijímání posloupnosti bitů problematické rozpoznat začátek a konec této sekvence a správně detekovat jednotlivé bity v optimálních intervalech.

Z hlediska synchronizace rozlišujeme přenos na:

Synchronní přenos
Přijímač a vysílač pracuje u tohoto druhu přenosu synchronně. Pojem synchronizace v tomto případě představuje nastavení dvou signálů do žádané vzájemné polohy. Toho je dosahováno tzv. synchronizačním signálem, který je generován na straně vysílače. K přijímači musí být přenášen zvláštním přenosovým kanálem nebo je tento synchronizační signál odvozen od jiného časového rastru, který má k dispozici jak přijímač, tak vysílač.

Asynchronní přenos
Sled vysílaných a přijímaných dat není časově vázán, rozpoznání posloupnosti sledu několika bitů shodné úrovně je obtížný, proto se tento přenos prakticky nepoužívá.

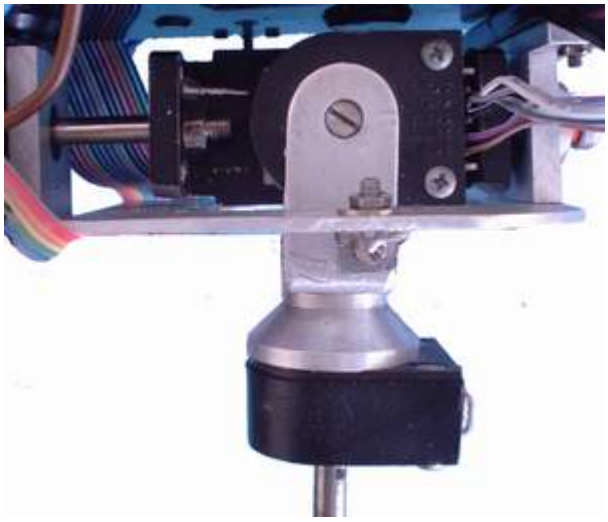
Arytmický přenos
Je kombinací předchozích dvou způsobů. Jednotlivé bloky zpráv jsou přenášeny asynchronně. Po detekci začátku bloku na přijímací straně dojde díky synchronizační informaci, kterou musí každý blok obsahovat, k synchronizaci mezi vysílačem a přijímačem, trvající celý časový úsek bloku.

4. Konstrukce trenažéru a napájecího zdroje

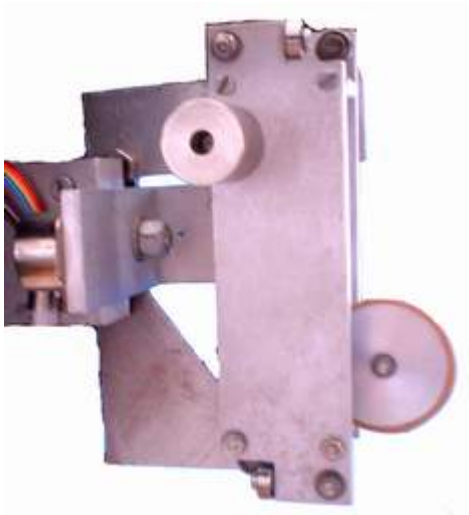
Trenažér představuje pevné propojení vrtulníku s heliportem. Jeho konstrukci a funkce rozebereme v této kapitole. Předem je nutno říci, že byl navržen primárně jako ochrana modelu proti mechanickému poškození. Model je určen pro práci v laboratoři, a tudíž není možno nechat helikoptéru volně létat, zvláště v případě tak nelineárního a nestabilního systému, jaký vrtulník představuje. Druhé kritérium, použité při konstrukci trenažéru, vychází z nutnosti ponechat možnost simulace základních letových režimů za předpokladu omezeného operačního prostoru.

4.1 Konstrukce a umístění heliportu v laboratoři

Výše zmíněná kritéria, tedy co nejmenší ovlivnění pohybu vrtulníku v jeho operačním prostoru a zároveň ochranná funkce při kolizi, splňuje konstrukce, založená na dvojici výkyvných kloubů a spojovací tyči. Kloub pod helikoptérou, zobrazen na Obr. 4.1a, je spojen s kloubem uchyceným ve spodní části trenažéru (Obr 4.1b) výsuvnou tyčí. Toto uspořádání má 6 stupňů volnosti a umožňuje tak vrtulníku pohyb v rámci základních letových režimů.

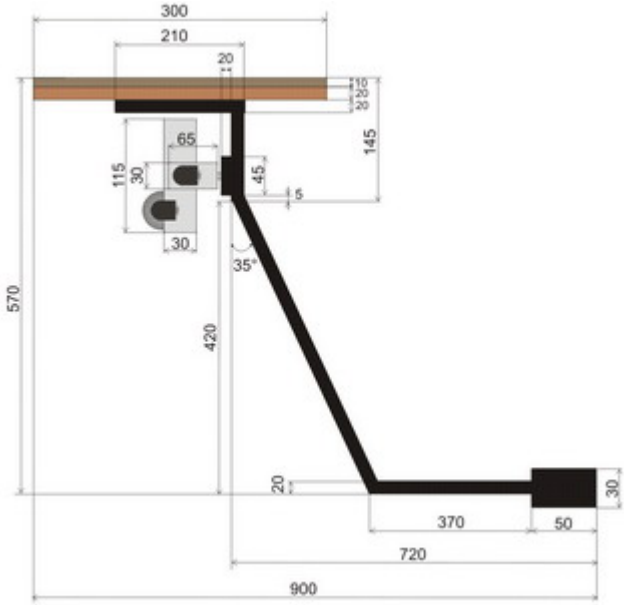


Obr.4.1a Kloub pod helikoptérou



Obr. 4.1b Kloub ve spodní části trenažéru

První dvojice výkyvných kloubů umožňuje náklon ve dvou navzájem kolmých vertikálních rovinách a je pevně uchycena pod dřevěnou kruhovou deskou o průměru 300 mm, tvořící přistávací plochu pro vrtulník (heliport). Přistávací plocha heliportu je polepena vrstvou z měkčené pryže, která má za úkol tlumit následky způsobené tvrdým přistáním modelu. Výkyvné klouby jsou spojené výsuvnou duralovou tyčí, která prochází kruhovým otvorem o průměru 130 mm. Rozměry heliportu jsou na obrázku 4.2.



Obr 4.2 Rozměry heliportu

Konstrukční řešení trenažéru omezuje pohyb kloubů, které se mohou otáčet v rozmezí +43º až -43° . Vzhledem k tomuto omezení a ke konečné délce tyče (400 mm) tvoří prostor, ve kterém se vrtulník v bodě uchycení může pohybovat , kužel rozšiřující se směrem vzhůru (obr. 4.3). Při bližším pohledu vidíme, že se jedná o sférickou souřadnou soustavu. Na konci výsuvné tyče je pevně připevněna druhá dvojice kloubů, doplněna o rotační kloub. Dvojice kloubů ovlivňuje příčný a podélný náklon vrtulníku v rozmezí +60° až -60°. Změnou souřadnice rotačního kloubu v rozmezí 0 do 360° umožňuje libovolný kurz modelu.

Obr. 4.3 Operační prostor konce výsuvné tyče

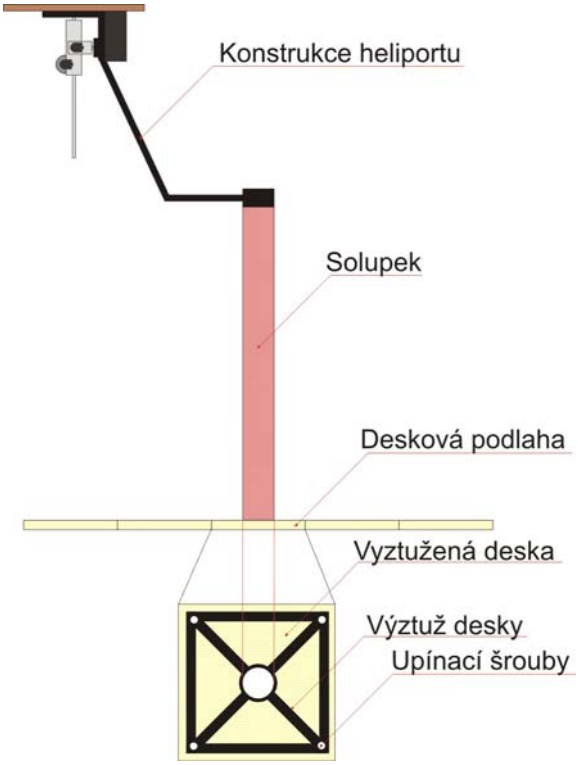
4.1.1 Ukotvení heliportu v laboratoři

Na vrtulník v letu působí soustava sil a naopak on působí silami na své okolí. V ideálním případě by celý jeho provoz měl být realizován na volném prostranství, kde nemůže docházet k turbulencím a vzduchovým rázům, způsobeným odrazem vzduchové hmoty od okolních předmětů. Avšak vzhledem k jeho umístění v laboratoři bylo nutno najít vhodné místo, které by zaručovalo minimální interferenci proudícího vzduchu s překážkami, aby nedocházelo k dalšímu zhoršování nelinearity modelu a přitom neohrožovalo při činnosti své okolí.

Samotná konstrukce popsaná v předchozí kapitole končí 1 m stranou a 0,5 m pod úrovní kruhové desky. Vycházeje z této skutečnosti bylo nutno navrhnout vhodný stojan, který by splňoval požadované vlastnosti. Naši předchůdci volili mezi dvěma možnostmi:

1) Konstrukce uchycená ke zdi
2) Nosná tyč připevněná k podlaze

Naši předchůdci zvolili druhou variantu, kterou blíže popíšu. Podlaha v laboratoří je složena z dřevěných desek, které jsou přišroubovány ke kovovému rámu v podlaze. Ve středu jedné z desek je vyvrtán otvor, kterým prochází vertikální ocelový sloup o průměru 60 mm. Dřevěná deska je zpevněna ocelovým křížem, do něhož je sloupek ukotven. Sloupek je vysoký 1.5 m, na konci je opatřen závitem, ke kterému je šroubem M6 připevněn heliport (obr. 4.3).



Obr. 4.4 Ukotvení heliportu v laboratoři

4.1.2 Antigravitační systém

První praktické pokusy o zálétání vrtulníku ukázaly, že model není schopný kvůli své velké hmotnosti samostatného vzletu. Základní model, zvolený našimi předchůdci k přestavbě, nebyl dostatečně výkonný a to způsobilo, že instalací přídavných funkčních částí na drak modelu (konstrukce nesoucí IRC čidla, IRC snímače, deska Xilinx 1, PWM) se zvýšila jeho hmotnost nad hranici, kdy je schopen se vznést. Bylo potřeba nalézt řešení tohoto problému, který zabraňoval dalšímu testování. Přídavné části jsou vyrobené z lehkého hliníku, tudíž snížení jejich hmotnosti by bylo velmi obtížné. Hmotnost IRC snímačů instalovaných na modelu je při porovnání s konstrukcí, která čidla nese, zanedbatelná, proto jejich případná výměna za novější se jevila jako zbytečná. Na základě daných možností jsem se problém snažil vyřešit nikoli zmenšením hmotnosti vrtulníku, nýbrž zvýšením jeho výkonu. Proto byl nainstalován silnější motor s větším počtem otáček. Zvýšením otáček však došlo k přílišnému namáháni mechaniky rotoru, která by časem v tomto pracovním režimu selhala.

V průběhu testování se zjistilo, že původně dodávané listy rotoru, vyrobené z umělé hmoty, nedokáží udržet vztlak a podléhají torzi, což značně ubírá helikoptéře na výkonu. Vzhledem k roku výroby modelu se již nevyrábějí dřevěné ani duralové listy rotoru, které by tento problém mohly vyřešit. Jako nejschůdnější řešení se ukázala instalace podpůrného antigravitačního systému, skládajícího se z "letecké" gumy, která nadlehčuje vrtulník přes vodící tyč. Toto řešení není ideální, neboť do systému zavádí další nelinearitu způsobenou napínáním gumy, nicméně je postačující do doby, než budou k dispozici tvrdší listy rotoru.

4.1.3 Praktické zkušenosti s heliportem

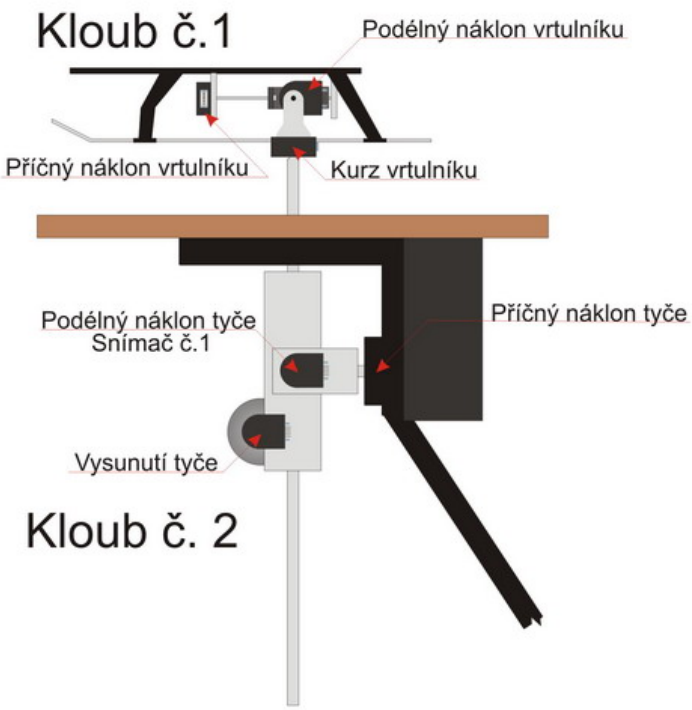
Celá konstrukce heliportu se při provozu modelu helikoptéry jeví jako dostatečně robustní a navíc pomáhá pohlcovat vibrace vytvářené modelem. Jedinou vadou, kterou jsme našli, je kolize vyrovnávacího rotoru s první částí upevňovací konstrukce, která nastává při poloze vrtulníku v záklonu s přídí kolmo vzhůru v rovině upevňovací konstrukce.

Vzhledem k tomu, že odstranění této kolize není jednoduché (vyžadovalo by změnu první částí upevňovací konstrukce, případně mechaniky trenažéru) a pravděpodobnost popsané kolize, ke které může dojít pouze při pádu vrtulníku, je malá, neprováděli jsme žádné změny ve stávající konstrukci. Při testování regulačních algoritmů, při nichž očekáváme nestabilitu, je na tuto kolizi nutno brát zřetel. Pokud by vyvstal požadavek na odstranění bezprostřední kolize listů vyrovnávacího rotoru s konstrukcí, doporučuji opatřit vyrovnávací rotor ochranným krytem. Při kolizi by došlo pouze k odražení modelu od konstrukce.

4.2 Snímání polohy

Znalost přesné polohy je podmínkou pro úspěšné řízení modelu helikoptéry, proto byl trenažér opatřen rotačními inkrementálními snímači polohy. Vzhledem k tomu, že se nepředpokládá autonomní pohyb modelu, je nejefektivnějším řešením určovat polohu z postavení mechanických částí.

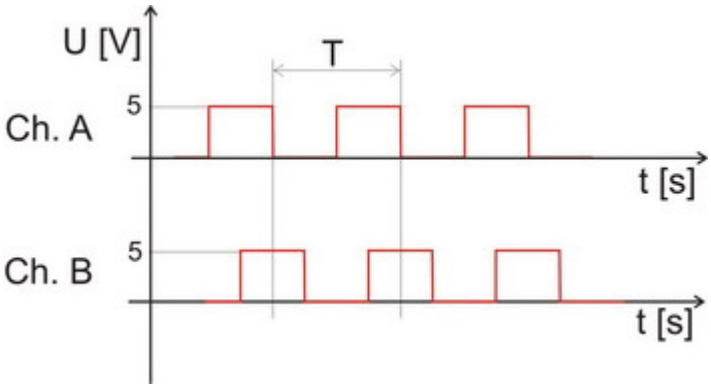
Snímače polohy osazené na trenažéru jsou dvojího typu, a to IRC 450.1 typ B respektive Bz. Použité snímače se liší počtem period výstupního signálu na otáčku a použitím indexu, který se u typu Bz generuje jednou za celou otáčku inkrementálního snímače. U modelu není tato vlastnost využita. Rozlišení snímače typu Bz je 540 dílků/ot , typ B má rozlišení 512 dílků/ot. Vysunutí tyče a kurz vrtulníku je snímán snímači typu B, ostatní klouby jsou osazeny snímači v provedení Bz. Aby bylo možné snímat vysunutí tyče inkrementálním snímačem polohy, je vysunutí tyče převedeno pogumovanou kladkou na otáčení, čímž se zvýšilo i rozlišení pohybu. Rozmístění snímačů ukazuje Obr. 4.5.



Obr. 4.5 Rozmístění IRC snímačů polohy

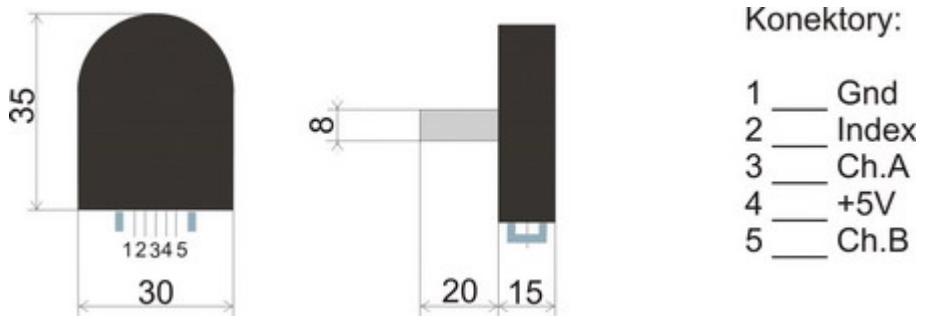
4.2.1 Charakteristiky inkrementálních snímačů

Oba typy inkrementálních snímačů vyžadují napájecí napětí 5V, jejich proudový odběr je v řádech desítek mA. Fázově posunutý výstup v TTL logice má označení Ch.A, Ch.B. Výstupy jsou vůči sobě fázově posunuty o ±¼ periody vztaženo k signálu v Ch.A, znaménko udává směr pohybu.Typický průběh viz. Obr 4.6.



Obr. 4.6 Výstupní signál z IRC snímačů pro pravotočivou rotaci

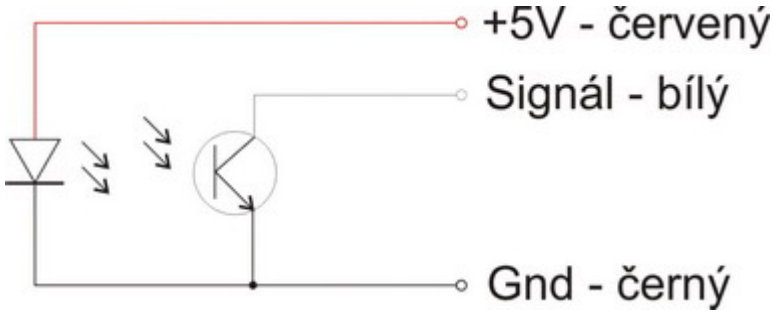
Zapojení snímačů a jejich rozměry jsou přehledně vyobrazeny na Obr. 4.7.



Obr. 4.7 Rozměry a zapojení IRC snímačů

4.3 Snímač otáček hlavního rotoru

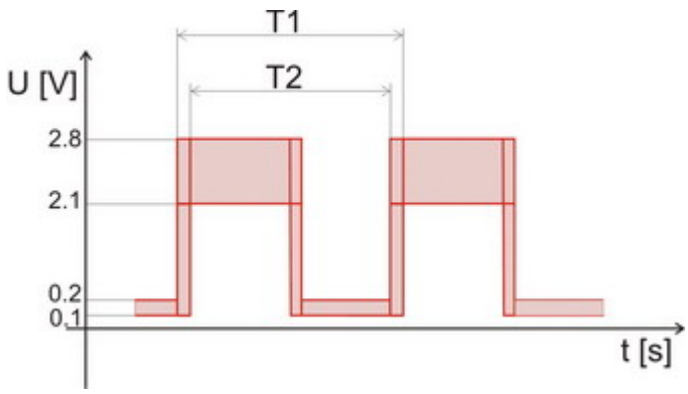
Snímač je realizován světelnou závorou, jejíž clonka byla z prostorových důvodu umístěna na hřídeli pohánějící vyrovnávací rotor (obr 4.8). Pokud je motor v záběru, jsou otáčky vyrovnávacího rotoru úměrné otáčkám nosného rotoru. V době volnoběhu tato úměrnost neplatí - z tohoto důvodu čidlo dává chybnou informaci. S touto nepřesností při měření musíme počítat při návrhu řízení.



Obr. 4.8 Schéma snímače otáček (s barevným označením vedení)

Snímač otáček osadil v rámci své diplomové práci Pavel Beneš, výstupní průběh signálu je na obr. 4.9. Jednotlivé pulzy se navzájem liší jak amplitudou, tak šířkou (šrafovaná oblast). Pro přímé zpracování v obvodu Xilinx je tento signál značně nevhodný. Z tohoto důvodu se výstupní signál upravuje pomocí TTL tranzistorového zesilovače.

Při měření výstupního signálu snímače otáček je potřeba mezi zdroj napětí a červený, respektive bílý konektor vložit sériový odpor (v mém případě 470 Ω resp. 20kΩ), jinak dojde k poškození diody, respektive tranzistoru.



Obr. 4.9 Průběh výstupního signálu

Tranzistorovým zesilovačem odstraníme pouze rozdílnou amplitudu výstupního signálu. Rozdílná šířka pulzů nám zavádí nejednoznačnost v počtu pulzů na jednu otáčku hlavního rotoru. Díky této chybě připadá na jednu otáčku hlavního rotoru devět až deset pulzů.

4.4 Napájecí zdroj

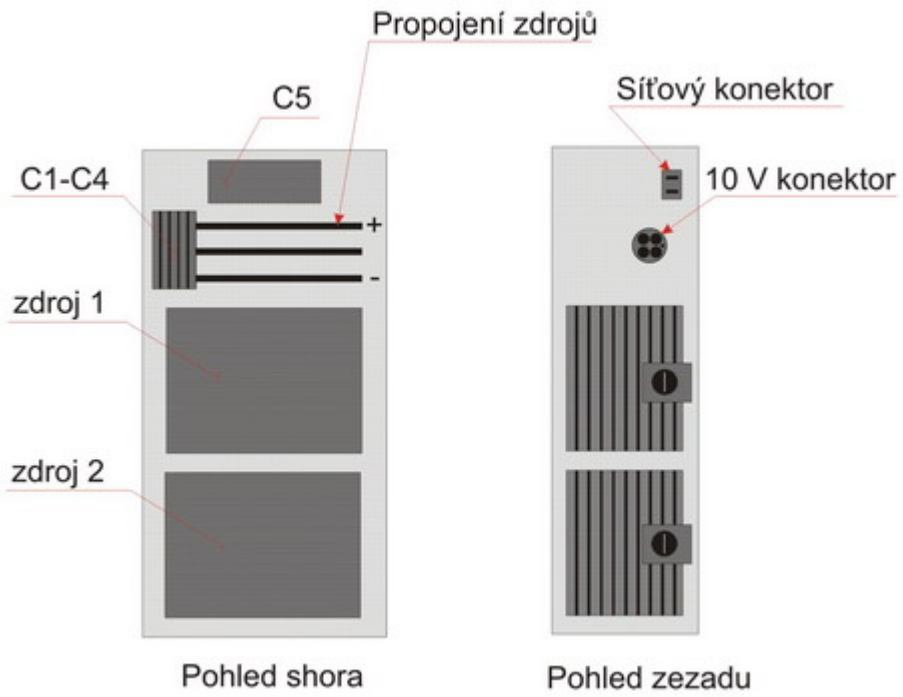
Model vrtulníku je konstruován na napájecí napětí v rozsahu *9 – 12V* . Při maximálním výkonu a napětí 10V odebírá ze zdroje 24A. Protože se ve vybavení laboratoře nenacházel a v současné době nenachází průmyslově vyráběný zdroj, který má požadované vlastnosti, bylo nutno vyrobit zdroj vlastní.

4.4.1 Konstrukce zdroje

Zdroj je postaven na dvou sériově zapojených impulsních měničích 5V/50A typu DC C 205 vyrobených v ZPA Děčín, které jsou uloženy do standardizované skříně.

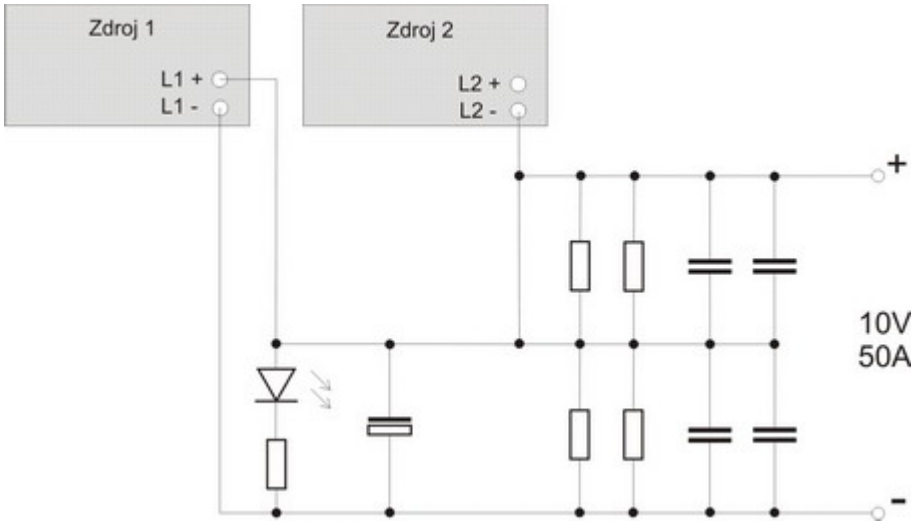
Zdroje jsou připojeny ke konektorům umístěným na zadním panelu (obr. 4.10). Z důvodu zabránění možné záměny je každý konektor jiného typu. Síťový konektor je typu “střední žehlička“, pro vývod 10V napětí je použit čtyřdutinkový kruhový konektor. Zapojení ukazuje obr 4.11.

Obr. 4.10 Výstupní 10V konektor



Obr. 5.11 Rozmístění součástek zdroje

Schéma vnitřního zapojení zdroje je na obr 4.12. C1-C4 jsou filtrační kondenzátory napětí. Proudová špička, která vzniká při změně výkonu motoru z nulové hodnoty na sto procent, dosahuje hodnot blízkých nominální hodnoty zdroje 50A, proto doporučuji plynulou regulaci zdroje. Proudová špička je tlumena odporem přívodních vodičů, který není při daných proudech zanedbatelný. Zde bych chtěl upozornit, že při změně napájecích vodičů je třeba brát zřetel na jejich odpor a s tím související úbytky napětí na vedení.
Použitý typ impulzních měničů nemůže pracovat naprázdno (vznikají velmi vysoká napětí ve spínacích částech zdroje), proto je ke každému zdroji připojen zátěžový 5Ω odpor složený z paralelní kombinace dvou odporů 10Ω/6W.Při provozu protéká každým odporem proud 0.5A. Celková výkonová ztráta představuje 5W pro jeden zdroj. Zdroje jsou takto naprázdno zatíženy 2% nominálního proudu.



Obr. 4.12 Schéma zdroje

4.4.2 Přívod napájení k vrtulníku

Všechny pokusy našich předchůdců o použití uhlíkových kartáčů, které by umožňovaly volný pohyb v kloubu č.1, selhaly z důvodů rychlé oxidace a s tím souvisejícím zvýšením přechodového odporu (to má za následek již zmíněnou velkou napěťovou ztrátu).

Tuto cestu jsme proto zavrhli a přívod energie je vyřešen tzv.dvojlinkou, kterou vedeme z důvodu co nejmenší změny dynamiky modelu kolmo vzhůru vnitřkem spojovací tyče ke kloubu č.1. Zde dvojlinka tvoří volný závit, umožňující pohyblivost modelu (závit je nastaven tak, že umožňuje modelu tři otáčky kolem středové osy). Při následném zkoušení regulace na modelu je na toto řešení nutno brát zřetel - je zde nebezpečí ukroucení dvoulinky a následná ztráta napájení.

Uvažovali jsme o instalaci spojky, která by omezovala pohyb vrtulníku o tři otáčky do každého směru. Toto řešení by si však vyžádalo další navýšení hmotnosti už tak těžkého modelu, a proto jsme jej zamítli.Toto omezení je však implementováno softwarovou cestou, viz kapitola 6.

4.4.3 Praktické zkušenosti a provozní zkouška

Po celou dobo provozu pracoval napájecí zdroj spolehlivě. Po vypnutí napájení drží zdroj asi 10 sekund napětí, je proto nutno vyčkat na zhasnutí kontrolní led diody.

5. Hardwarová část projektu

Kapitola obsahuje popis celkové koncepce modelu z pohledu systému povelového řízení, použitých přenosových cest, dále provedené změny na původním hardware a podrobněji pojednává o systému sběrnic a jeho napojení na interface. Důraz je kladen na rozbor vnitřní sběrnice interface, kde je podrobná znalost průběhu čtecích a zapisovacích cyklů důležitá pro implementaci řídicích funkcí v části software (kapitola 6), jakožto i adresování kanálů a inicializace výstupních registrů Pro rozsáhlejší popis prvků hardware včetně všech zapojení a schémat odkazuji případného zájemce na diplomovou práci [1] Milana Martináka.

5.1. Celková koncepce

Pro snadnou orientaci v propojení jednotlivých částí modelu a jejich funkcí uvedu schéma celkové koncepce modelu (obrázek 5.1) včetně stručného popisu komunikačních kanálů a aktivních prvků.

Kanál 1: PC -> vrtulník

Povelové řízení představuje jeden komunikační kanál, zajišťuje ovládání serv helikoptéry a výkon motoru. Jedná se tedy o komunikační kanál z PC do vrtulníku. Na jeho začátku je k vysílání řídicích dat, která se objeví na výstupu PCI karty a jsou zpracovávána v interface (podrobněji viz kapitola 5.3), použita původně 4 kanálová RC souprava. Vzhledem k požadavku regulace otáček motoru byla rozšířena o další kanál. Využívá se pulzně-šířkové modulace, jejíž signál vytvářejí časovače 8253 řízené přes kartu z PC.

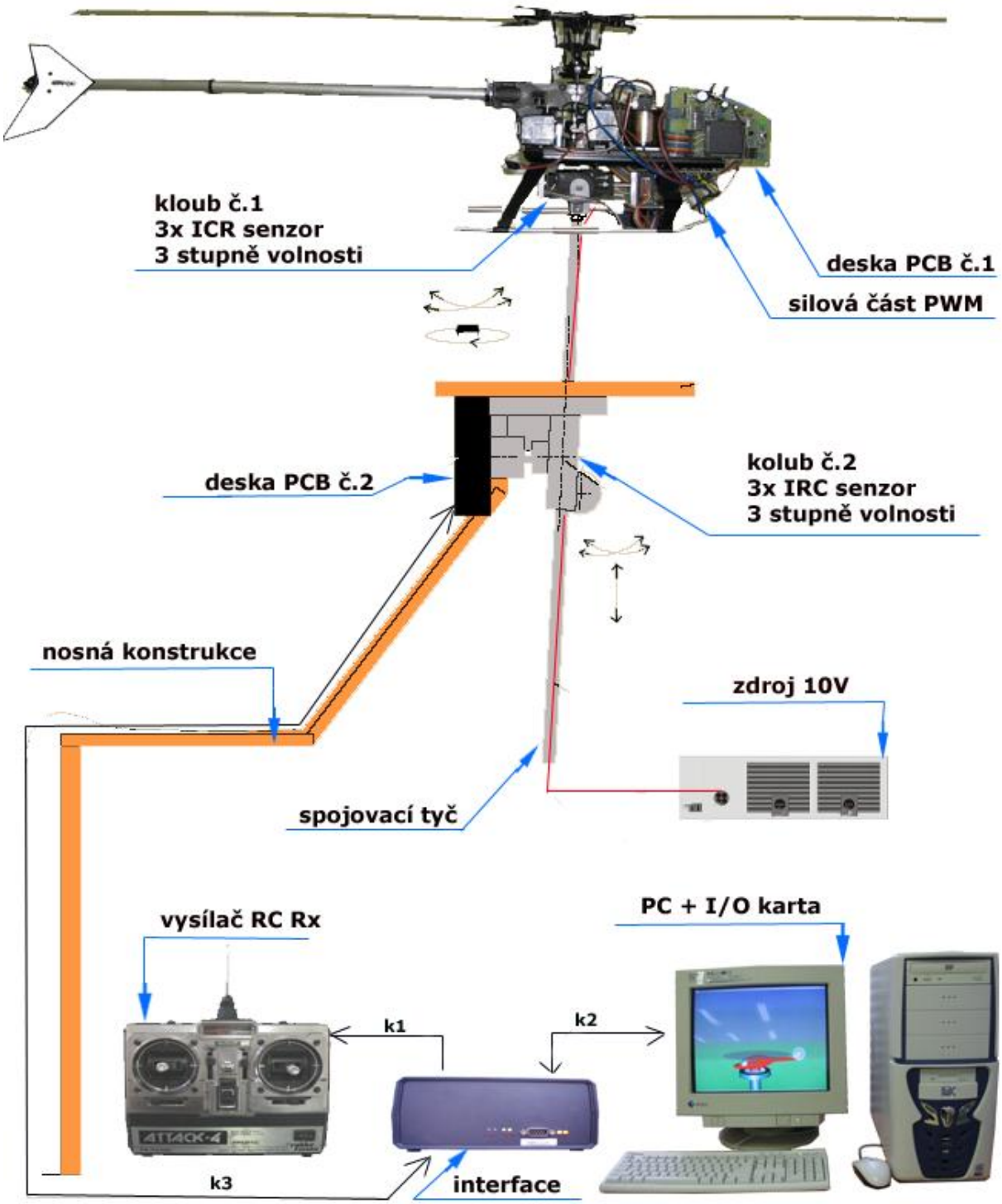
Signál z vysílače přijímá původní, k RC soupravě dodávaný, přijímač. První čtyři pulsy jsou použity k řízení serv a pátý, pro otáčky motoru, je přiváděn do Desky č.1. V této desce je signál zpracován programovatelným polem Xilinx (kapitola 5.3.1), který vytváří odpovídající PWM signál pro řízení otáček motoru.

Kanál2: vrtulník -> PC

Druhý komunikační kanál slouží ke sběru dat. Vrtulník i trenažér je osazen trojicí IRC snímačů, jež mají za úkol poskytovat řídicím algoritmům v počítači informace o poloze helikoptéry v prostoru. Pro informaci o výkonu motoru je na vrtulníku ještě snímač otáček. Informace z čidel umístěných na vrtulníku jsou zpracovány přímo deskou č.1. Deska č.2 zpracovává signály zbylých snímačů situovaných na heliportu.

O zpracování se opět starají pole Xilinx ve zmiňovaných deskách, kde upravují signál na absolutní hodnotu a posléze sériové přenášejí údaje ze všech senzorů do interface , kde probíhá jejich kódování a uložení do registrů simulované sběrnice 8080. Po této sběrnici je schopen počítač komunikovat z interface a získává tak data o poloze modelu.

Součástí celkové koncepce je i heliport s trenažérem a napájecí zdroj (viz kapitola 4). Zdroj napájí pouze samotnou helikoptéru pomocí přívodního dvoužilového kabelu. Interface, vysílač a deska č.2 jsou napájeny z PC přes řídicí kartu. Pokud je tedy zdroj vypnut nebo selže, způsobí to vyřazení *kanálu 1* a části *kanálu 2*, která přenáší data ze snímačů na helikoptéře. Snímače pod heliportem budou i nadále ukazovat polohu správně.



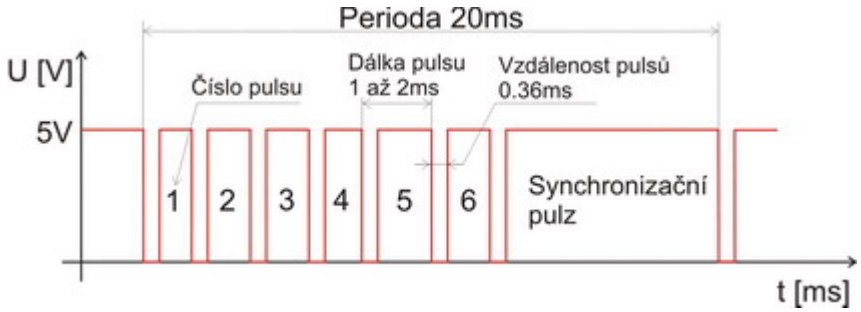
Obr 5.1 Celková koncepce – hardware

5.2 Řízení pomocí RC soupravy

K modelu standardně dodávaná RC souprava má 4 kanály s pulsně-šířkovým dělením kanálu. Stejnosměrné napětí, generované výchylkou potenciometru spojeného s pákovými ovladači, určuje šířku daného pulsu. Takto vzniklý signál je amplitudově modulován a přenášen na modelářské frekvenci 40,695 MHz. Za modulátorem následuje vysokofrekvenční zesilovač a anténa.

Jak bylo zmíněno v kapitole 5.1, potřeba řídit výkon motoru vedla k rozšíření soupravy o další dva kanály. Informaci pro řízení motoru nese pouze pátý kanál, šestý není využit, ale musí být vždy přítomen, aby nedocházelo k rozpadu komunikace. Délka jednotlivých pulsů nese informaci o požadovaném natočení každého servomotoru.

Celý vysílací rámec obsahuje také důležitý synchronizační puls. Vzdálenost mezi jednotlivými pulsy se smí pohybovat v rozmezí 1 až 2 ms. Pokud při vytváření vysílacího rámce není respektován tento interval u některého z kanálů, může dojít ke ztrátě synchronizačního pulsu a rozpadu komunikace. Datový rámec a význam jednotlivých pulsů ukazuje obrázek 5.2 (obrázek byl přejat z diplomové práce [1]), respektive tabulka 5.1.



Obr. 5.2 Výsílací rámec RC vysílače

Číslo pulsu	Popis
1	Podélná cyklika
2	Příčná cyklika
3	Kolektivní řízení
4	Nastavení listů zadního rotoru
5	Výkon motoru
6	Nevyužito

Tab. 5.1 Popis pulzů vysílacího rámce

Pro řízení vrtulníku pomocí PC se ukázalo jako nejjednodušší řešení generovat šířkově modulovaný signál popsaný v této kapitole přímo pomocí řídicí karty v počítači a ten následně přivést do modulátoru. Původní kodér je odpojen a nevyužívá se.

5.3 Komunikace PC-interface

Znalost vnitřní architektury interface, adresování jednotlivých registrů a cykly pro čtení a zápis jsou podkladem pro vytvoření řídicí aplikace v PC tak, jak je popsána v kapitole 6.

5.3.1 Integrované obvody Xilinx

Obvod Xilinx je programovatelné logické pole s velkou mírou integrace. Nebudu zde popisovat architekturu ani způsob programování těchto polí, omezím se pouze na základní výčet vlastností a stručnou charakteristiku. Více informací lze nalézt v diplomové práci [5] a [6].

Z pohledu uživatele se Xilinx jeví jako programovatelný LSI obvod, podobně jako hradlové pole. Základní stavební jednotku tvoří konfigurační logický blok. V modelu používáme obvody řady XC3000. Obvody jsou schopny pracovat s logikou TTL a CMOS. Každý obvod obsahuje globální signál RESET (na deskách č.1 a č.2 vyveden jako spínač). Program pro obvod Xilinx je uchováván v paměti typu CMOS-SRAM. Do této paměti je nahrán konfigurační program ihned poté, co obvod začne být napájen. Jako zdroj programu může sloužit běžná paměť typu PROM nebo sériová linka. V modelu využíváme pamětí XC1736.

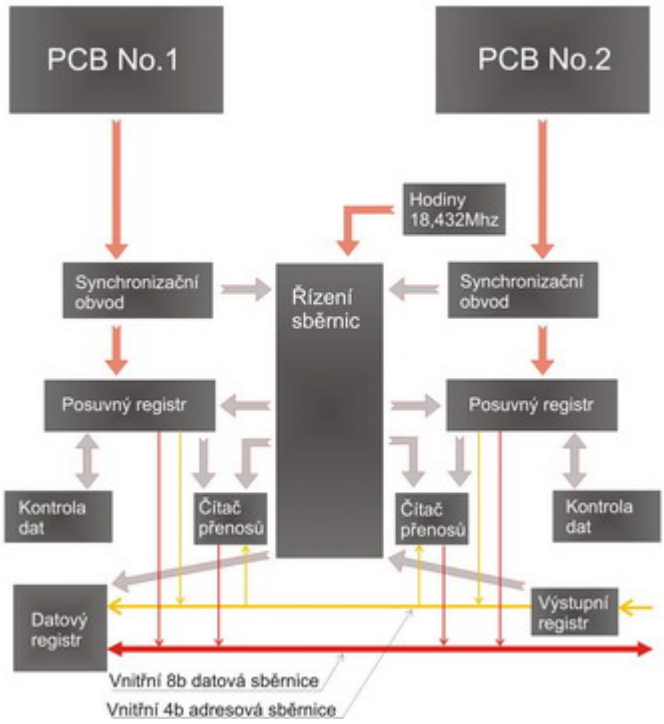
5.3.2 Architektura interface

Jak bylo naznačeno výše, data z čidel představují dvě sériové linky, po kterých jsou informace přenášeny do interface. Zde jsou rozdělena podle adres do příslušných registrů pomocí programovatelného pole Xilinx. Přenos dat v rámci interface ukazuje blokové schéma na obrázku 5.2 (obrázek byl přejat z diplomové práce [1]) .

Celá procedura probíhá tak, že synchronizační obvod pozná začátek bloku dat podle synchronizační slabiky a následně dojde k synchronizaci s přijímacím časovačem. Dále jsou data ze synchronizačního obvodu postoupena posuvnému registru o délce 32b, což odpovídá velikosti celého vysílacího slova. Po naplnění posuvného registru se provede kontrola přijatých dat. Nedojde-li k detekci chyby, uloží Xilinx data podle adres do správných registrů. Tam jsou uchovávána až do příští aktualizace.

Výstupní registr plní funkci oddělovače adresního prostoru PC od adres vnitřní sběrnice. Vnitřní sběrnice tedy představuje sdílené médium a je třeba stanovit pravidla pro její správu a přístup. Požadavky na přidělení sběrnice zaznamenává obvod oddělovače. Fyzicky to zajišťuje třístavový budič 74HCT373, řízený obvodem Xilinx.

Systém přidělování sběrnice je založen na principu fronty. To znamená, že požadavky jsou zpracovávány v pořadí, v jakém přicházejí. Nastane-li situace, že přijde více požadavků ve stejný okamžik, je rozhodnuto o pořadí přidělování podle pevně stanovených priorit. Nejvyšší prioritu má počítač. Odtud vyplývá omezení na rychlost čtení dat počítačem, které se nesmí opakovat s periodou kratší než 1,5 µs. Pokud by se tak stalo, pak by počítač četl neustále stejná data, protože by žádný z kanálů vnitřní sběrnice nedostal povolení k zápisu do registrů.



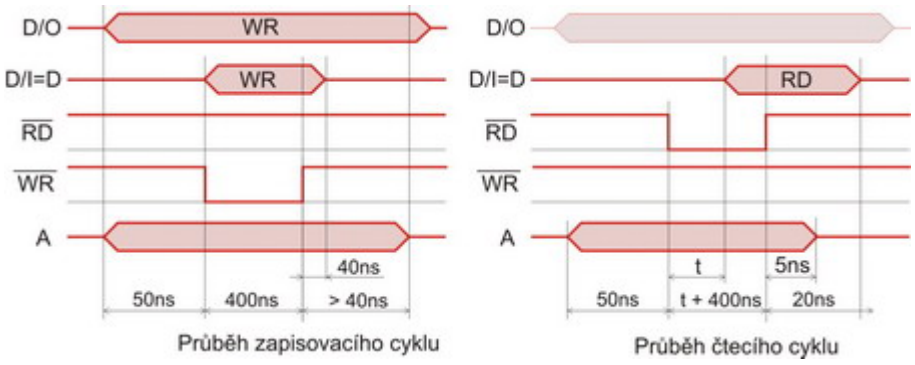
Obr 5.2 Přenos dat v rámci interface

5.3.3 Čtení a zápis na sběrnici 8080

Simulovaná sběrnice mikroprocesorového systému 8080 představuje komunikační spojnici mezi řídicí kartou v PC a registry - jak systému sběru dat, tak registry vysílacích časovačů. Sběrnice je obousměrná s šířkou 8 bitů. Původně umožňuje 16 bitovou adresaci, ale pro dané použití byla adresa omezena na 5 bitů.

Popíšeme si časový průběh zápisového a čtecího cyklu (viz obrázek 5.3). Signály pro povolení čtení a zápisu RD, WD jsou aktivní v logické nule. Adresa se musí objevit na sběrnici s předstihem před aktivním signálem RD či WD, kde setrvává po celou dobu cyklu s dostatečným přesahem. Čtení i zápis se provádí s nástupní hranou signálu RD, respektive RW. Je tedy nutné, aby se data D na sběrnici vyskytovala s potřebným předstihem před touto hranou a setrvala tam i po ní. Časový rámec uváděný na obrázku 5.3 představuje minimální doporučené hodnoty u obou cyklů. Software počítače, který provádí čtení a zápis prostřednictvím určených funkcí (viz kapitola 6.2.5), používá pomalejšího časování, avšak je nezbytně nutné dodržet logickou posloupnost signálů včetně jednotlivých předstihů a přesahů vůči signálům RD a WD.

Dodržet časování podle obrázku 5.3 lze pouze v případě nízkoúrovňového přístupu k prostředkům řídicí karty, což vzhledem k použitému vývojovému prostředí (viz kapitola 6 Software) není možné. Funkce pro zápis a čtení popsané dále v kapitole Software upravují časový průběh čtecího i zápisového cyklu na základě dlouhodobého experimentování.



Obr. 5.3 Čtecí a zápisový cyklus – časový průběh

5.3.3.1 Propojení sběrnic

Sběrnice 8080 je obousměrná, zatímco vstupy a výstupy V/V karty v PC jsou jednosměrné. Aby nedošlo ke kolizi, vyhodnotí obvody pro řízení sběrnice na základě signálů RD a WD, která operace se bude provádět. Vstupy na kartě (D/I0 – D/I18) jsou přímo propojeny k příslušným bitům datové sběrnice D.To je důvod, proč se data na D/I objevují v obr. 5.3 jak ve čtecím, tak v zapisovacím cyklu. Správnost čtení je zajištěna programovým vybavením.

Adresová sběrnice A je k výstupním signálům D/O8 - D/O12 připojena obdobně. Přímé spojení je v případě adresy umožněno tím, že počítač je jejím jediným zdrojem. Naopak výstupní datové signály D/O0 - D/O7 musí být odděleny od datové sběrnice z důvodů stálé přítomnosti dat na výstupu. Oddělovač je reprezentovaný obvodem 74HCT373 (viz kapitola 5.1.1), který je řízen signálem WR. To způsobuje, že se na obr. 5.3 v zapisujícím cyklu objeví data na sběrnici D (spojená s D/I) až po aktivaci WR, kdežto u čtecího cyklu jsou data na D/O od sběrnice D oddělena.

5.3.3.2 Propojení kabelů

Řídicí karta v PC je propojena s Interface dvojicí plochých kabelů (k1 a k2), zakončených na obou stranách samopřeznými konektory s dvaceti vývody. Oba kabely (jeden vstupní D/I a jeden výstupní D/O) přivádějí do interface napětí 5V a 12V. Nižší napětí se využívá k napájení Interface, vysílačky a desky PCB č.2. Oba kabely jsou označeny stejně jako vstupy, do kterých mají být zastrčeny. V případě záměny nedojde k poškození karty ani Interface, způsobí to však nefunkčnost řízení a sběru dat z vrtulníku. Tuto chybu je schopna odhalit diagnostika implementovaná v řídicí aplikaci (kapitola 6).

5.3.4 Adresování na sběrnici 8080

Adresní prostor na sběrnici 8080 je rozdělen do tří částí, pro vysílací čítače je vyhrazeno osm adres, datové registry zabírají sedm a kontrolní registry dalších sedm adres.

Vysílací čítače

Jsou tvořeny dvojicí universálních programovatelných obvodů 8253. Každý takový obvod dokáže vytvořit tři čítače/časovače po 16b, které mohou pracovat jako MKO. Časovače pracují na kmitočtu 256kHz. Potřebujeme-li délku pulsu 1-2ms při daném kmitočtu, musíme nastavovat předvolby čítačů v rozmezí 256 až 512.

Každý čítač obsahuje čtyři registry, které jsou připojeny k vnější datové sběrnici komunikující s PC. Tři z těchto registrů jsou registry předvolby čítačů, čtvrtý registr je registr řídicí. Adresování registrů je v tabulce 5.2.

Registr	Adresa
Obvod č. 1	
Předvolba čítač 1	00000B
Předvolba čítač 2	00001B
Předvolba čítač 3	00010B
Řídicí registr	00011B
Obvod č. 2	
Předvolba čítač 4	00100B
Předvolba čítač 5	00101B
Předvolba čítač 6	00110B
Řídicí registr	00111B

Tab. 5.2 Adresování vysílacích čítačů

Při každém zapnutí je potřeba obvody 8253 pro správnou funkci znovu inicializovat. Pokud bychom to neudělali, riskujeme zápis hodnot do nesprávných registrů a případné poškození modelu. Inicializaci provedeme nahráním 8 bitových konfiguračních dat na adresu řídicího registru a postupně takto nastavíme všechny čítače. Tabulka 5.3 ukazuje nastavení všech osmi bitů pro každý z čítačů.

	D7	D6	D5	D4	D3	D2	D1	D0
Význam bitu	SC1	SC0	WR1	WR2	M2	M1	M0	BCD
Čítač 1	0	0	1	1	0	0	1	0
Čítač 2	1	0	1	1	0	0	1	0
Čítač 3	0	1	1	1	0	0	1	0

Tab.5.3 Konfigurační data pro inicializaci vysílacích čítačů

Způsob zápisu dat do vysílacích registrů i jejich inicializace z pohledu PC je popsána podrobně v kapitole 6 Software.

Datové registry

Do těchto registrů ukládá Xilinx údaje ze senzorů polohy. Maximální doba potřebná k dodání dat PLP Xilinxem je 780 ns, což je způsobeno přístupem na sdílené médium. Maximální

dosažitelný počet přenosů za jednu sekundu je 1125, proto provádět pokus o čtení dat častěji než 1ms nemá smysl.

Význam registru	Adresa
Zdroj dat: PCB No.1	
Příčný náklon vrtulníku	10000B
Podélný náklon vrtulníku	10001B
Kurz modelu	10010B
Otáčky nosného rotoru	10011B
Zdroj dat: PCB No.2	
Příčný náklon tyče	10100B
Podélný náklon tyče	10101B
Vysunutí tyče	10110B

Tab.5.4 Adresování datových registrů

Kontrolní registry

Kontrolní registry slouží pro účely diagnostiky. Jejich popisem a vyhodnocováním možných chyb se zabývá kapitola 6.2.8 Funkce pro diagnostiku a kontrolu hardware. Adresování je v tabulce 5.5.

Význam registru	Adresa
Čítač přenosů z PCB No.1	11000B
Čítač přenosů z PCB No.2	11001B
Kontrolní slovo P	11011B
Kontrolní slovo K	11100B
Kontrolní slovo &	11101B
Kontrolní slovo J	11110B
Kontrolní slovo F	11111B

Tab. 5.5 Adresování kontrolních registrů

5.4 Deska PCB č.1

Deska PCB č.1 musela být vytvořena zcela znova z důvodů špatného návrhu původního hardware, proto zde uvedu základní charakteristiky desky nové. Detailní popis a schémata zapojení obsahuje diplomová práce [1].

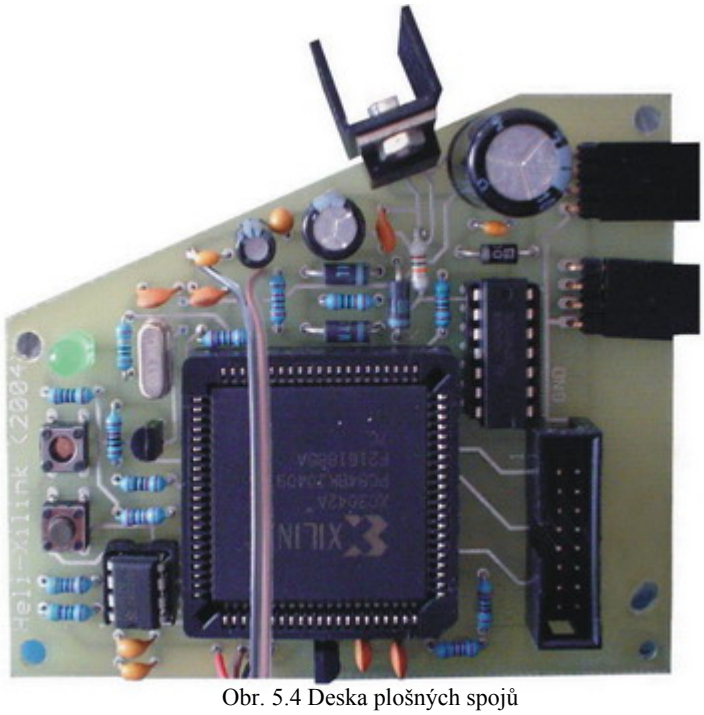
Jedná se o řídicí desku umístěnou přímo na modelu vrtulníku. Plní funkci zpracování dat z čidel polohy a senzoru otáček motoru. Dále vytváří signál PWM pro řízení otáček motoru.

5.4.1 Stav původní desky

Stav původní desky neumožňoval provoz modelu. V důsledku špatně vyrobeného plošného spoje, který obsahoval množství dodatečně realizovaných propojek pomocí drátů, docházelo k výpadkům Xilinxu a tedy i výkonu motoru. Při konstrukci desky se nepočítalo s mechanickým namáháním spojů vlivem častých otřesů vrtulníku. Schéma zapojení desky PCB č.1 uvedené v pracích [5] a [6] neodpovídalo skutečnému stavu desky. Bylo tedy nutné vytvořit schéma vlastní na základě podrobné analýzy.

5.4.2 Deska plošných spojů a další úpravy

Nová deska PCB č.1 (obrázek 5.4) se v provozu osvědčila a vyřešila problémy původního hardware. Spolu touto deskou byl nově vytvořen plošný spoj pro napájení a silovou část PWM (více [1]), rekonstruována sběrnice pro přenos údajů z čidel do desek č.1 a č.2 a vyměněna celá ocasní vodící tyč a listy rotoru.



Obr. 5.4 Deska plošných spojů

6. Softwarová část projektu

Kapitola podává ucelený přehled o řídicím software, programovém prostředí a použitých toolboxech. Dále obsahuje popis jednotlivých funkcí a grafické nástavby programu.

6.1 Analýza původního stavu software

Pro účely přestavby vrtulníku a vytvoření softwaru pro jeho ovládání, diagnostiku a komunikaci bylo zapotřebí nejprve důsledně analyzovat stávající stav modelu a softwaru. Jednotlivé charakteristiky původního softwaru rozebereme v následujících třech kapitolách. Vzhledem ke skutečnosti, že vývoj nového softwaru se neopíral o původní aplikace ani na ně nenavazoval, nebudu se podrobně zabývat rozborem jednotlivých programů. Zájemce o detailnější popis odkazuji na diplomovou práci [1].

6.1.1 Propojení softwaru a hardwaru z hlediska PC

Počítač s modelem komunikuje prostřednictvím simulované řídicí sběrnice 8080. Využívalo se 15 portů rozšiřující karty PCL-812, rozdělených na kanály řídicích signálů, vstupní a výstupní datovou sběrnici. Karta takto může přes sběrnici zapisovat a číst z paměťových registrů, které obsahují informace o poloze vrtulníku a veškeré řídicí signály. Popis funkcí jednotlivých registrů obsahuje kapitola 5.

Při čtení registrů obsahujících informace o poloze vrtulníku se přenáší pouze nejnižších 8bitů registru polohy, a proto je nutno softwarově rekonstruovat zbylých 8 vyšších. Specifika zápisu spočívají ve skutečnosti, že je třeba dát si pozor na skutečnost, že vnitřní registry jsou 16 bitové. Zápis do obou částí se provádí pomocí dvou po sobě následujících sběrnicových cyklů. Vzhledem ke zvolenému režimu činnosti musí být vyšší polovina registru vždy naplněna hodnotou 1 a nižší polovina určuje pohyb akčního členu (podrobně v kapitole 6.2.5). Délka výstupního pulsu se pohybuje od 1ms do 2ms a řídicí hodinový kmitočet je nastaven na 256kHz.

Přijatá data představují pouze informaci o změně oproti poslední poloze, což je dáno inkrementálním charakterem použitých čidel. Je tedy nutno tyto údaje předzpracovávat a normovat přímo v programu.

V modelu trenažéru jsou použity dva různé typy IRC snímačů, proto je potřeba každý sejmutý údaj transformovat odpovídající konstantou do správného tvaru. V tabulce 6.1 jsou uvedeny jednotlivé konstanty včetně způsobu jejich výpočtu. Konstanta K_1 platí pro snímač typu BZ (540 dílků/otáčka), zatímco konstanta K_2 platí pro smímač B (512 dílků/otáčka).

Vzdálenost vrtulníku od heliportu, tedy v podstatě délka vysunuté tyče, je měřena také IRC snímačem, opatřeným snímacím kolečkem o průměru 35mm. Pro přepočet délky tyče platí konstanta K_3. U snímače délky je přesnost 2x snížena, protože nemá smysl měřit s přesností větší než 0.1mm, kterou není schopna udržet ani mechanická část trenažéru. Označení Dc v tabulce 6.1 představuje naměřený počet dílků na čidle. Tyto přepočty používám v řídicí aplikaci k vizualizaci číselných informací o poloze i pro vykreslení 3D modelu vrtulníku v prostoru.

Označení	Popis	Výpočet
α_1 [rad]	Úhel pro snímač typu BZ	$2*\pi/(4*540)*Dc= 0.002901=K_1*Dc$
α_2 [rad]	Úhel pro snímač typu B	$2*\pi/(4*512)*Dc= 0.003067=K_2*Dc$
l [mm]	Délka vysunutí tyče	$d*\pi/(2*512)*Dc= 0.1073865=K_3*Dc$

Tabulka 6.1 Převodní konstanty pro výpočet úhlů a vysunutí tyče

Tato specifika spolu s požadavkem na rychlé vzorkování na kartě kladla v době realizace značné nároky na hardware, a proto jediný funkční program umožňující odděleně číst data o poloze nebo ovládat jednotlivá serva byl realizován v prostředí MS DOS a naprogramován v jazyce Pascal a částečně Assembler.

6.1.2 Program pro ovládání v prostředí Matlab

Pro komunikaci s vrtulníkem byl vytvořen jednoduchý program pro Matlab 386, který umožňoval inicializaci řídicí karty PCL-812. Neobsahoval však žádné funkce pro řízení modelu či diagnostiku.

6.1.3 Program pro testování modelu v prostředí MS DOS

Program umožňuje test karty PCL-812, inicializaci I8253, test vnějších registrů a obou komunikačních kanálů. Dokáže číst data z jednotlivých čidel, avšak neodstraňuje chyby způsobené rušením v klidovém stavu helikoptéry. Lze ovládat servomechanismy po jednom a roztočit motor.

Chybí možnost zobrazovat polohu vrtulníku a zároveň plně ovládat jednotlivá serva. Vizualizace letu nebyla vzhledem k výpočetním kapacitám realizována, stejně tak regulace nebo demonstrace některého z letových režimů. Nejsou implementovány ani programové ochrany hardwaru v případě několikanásobného otočení vrtulníku nebo možnost okamžitě znovu inicializovat řídicí registry a vypnout motor při případném rozpadu komunikace.

6.1.4 Závěry vyplývající z analýzy

Přístup k hardwarovým prostředkům karty, možnosti při výpočtech a zpracování přijatých dat jsou natolik odlišné od systému a možností, které nabízí Matlab 6.5, ve kterém jsem se rozhodl realizovat nový řídicí software, že nebylo možno původního programu nijak využít. Nový software byl realizován pouze na základě hardwarových specifikací systému model-PC karta.

6.2 Řídicí software a komunikace s PCI kartou

Řídicí software jsem navrhoval s ohledem na snadné ovládání základních primitiv modelu - jako je přímé řízení serv a motoru nebo snadné čtení a zpracování informací o poloze vrtulníku. Modularita aplikace je zaručena rozdělením primárních ovládacích prvků do jednotlivých souborů, funkcí a procedur. Aplikace je snadno rozšiřitelná i pro komplexní regulační úlohy, které mohou následovat po úplné identifikaci modelu.

.

6.2.1 Celková koncepce softwaru a propojení s modelem

Pro snadnější orientaci v softwaru popisovaném podrobně v dalších částech této kapitoly a pro přehled o vzájemných vazbách, které se v něm vyskytují, uvádím blokový diagram celkové koncepce (Obr.6.1) a stručný popis.

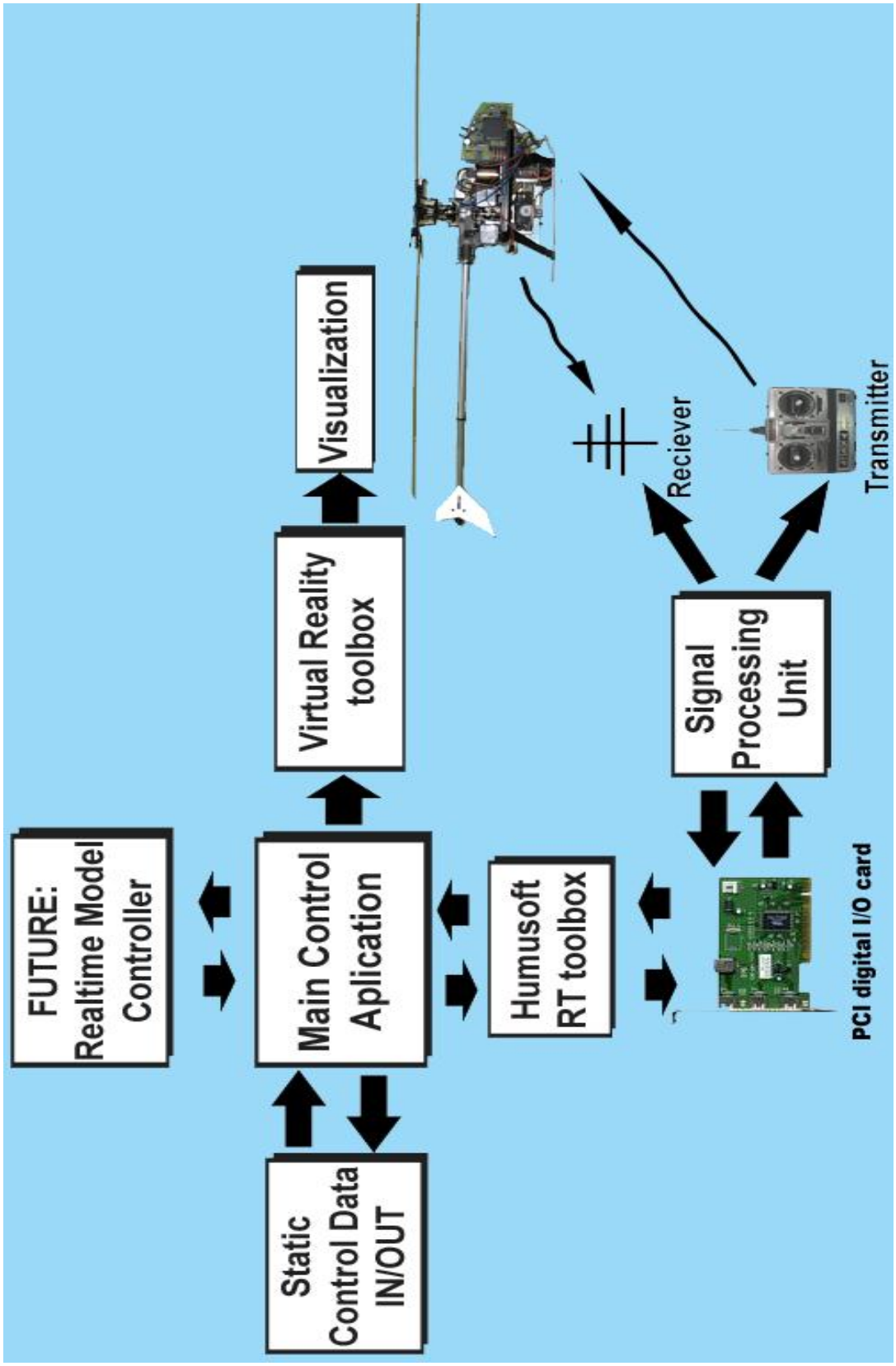
Regulačním členem, který fyzicky přijímá data o poloze vrtulníku a zároveň vysílá informace potřebné k řízení modelu, je digitální I/O karta sběrnice PCI. V programovém prostředí Matlab verze 6.5 běží hlavní řídicí aplikace, která prostřednictvím přídavných toolboxů umožňuje komunikaci s modelem a zpracování dat. Komunikace hlavní aplikace s PCI kartou využívá funkcí Real-Time toolboxu, zatímco vizualizace je vytvářena přeposíláním dat o poloze a jejich zpracováním pomocí Virtual Reality toolboxu.

Data přijímaná z průběhu letu mohou být ukládána na disk k pozdější analýze, zpracování nebo vizualizaci. Naopak aplikace umožňuje pomocí předdefinovaného souboru hodnot vrtulník řídit. Tímto způsobem je vytvořen například demonstrační letový program. Součástí hlavní aplikace je jednoduchý PID regulátor (podrobněji v kapitole 7) pro zadní vyrovnávací rotor helikoptéry, který umožňuje stabilizaci modelu při rotaci kolem hlavní svislé osy. GUI nádstavba dovoluje ovládat veškeré prvky aplikace pomocí myši a klávesnice.

Fyzicky je program realizován v souborech typu m-file a spouští se souborem *mor.m.* Dalšími typy souborů jsou *fig* (informace o rozmístění, typech vlastnostech prvků GUI rozhraní) a *wrl* (soubor virtuálního světa s 3D modelem vrtulníku). M-fily funkcí jsou umístěny v samostatném adresáři nazvaném *funkce.*

Jak již bylo zmíněno, model dosud nebyl plně identifikován. V aplikaci se do budoucna předpokládá použití Controll toolboxu a Simulinku, jež obsahují funkce a nástroje umožňující aplikovat regulátor na identifikovaný model.

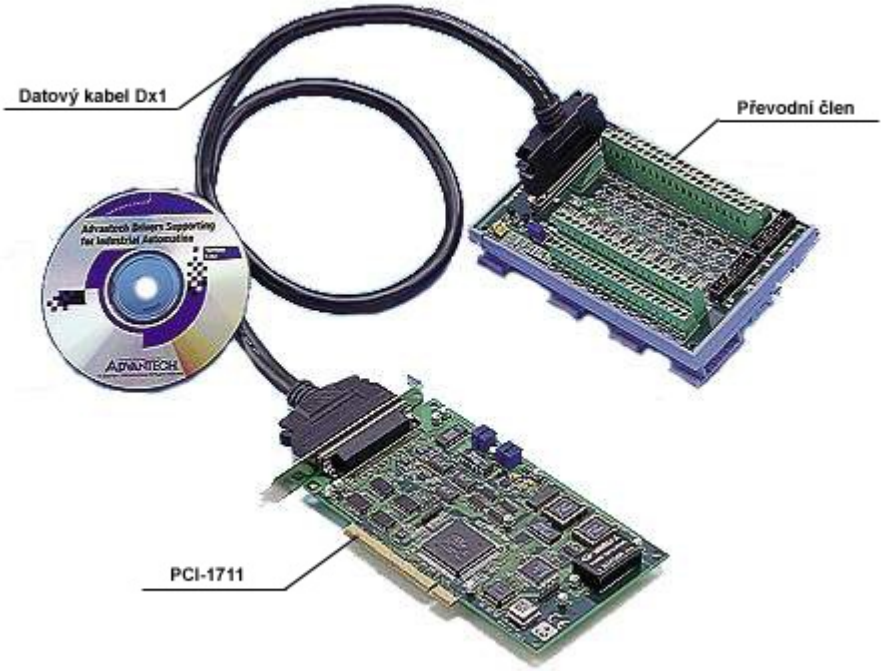
Aplikace je díky modularitě snadno upravitelná a lze předpokládat, že bude pracovat na nových verzích Matlabu a toolboxů bez nutnosti větších úprav. Námi zvolená PCI karta, popsaná v následující kapitole, je dostatečně rychlá pro zpracování přenášených dat, avšak ani její výměna v budoucnu (např. z důvodu kompatibility sběrnice nabízené základní deskou počítače) by nevyvolala potřebu přeprogramovat celou aplikaci, ale pouze upravit funkci pro zápis, čtení a inicializaci (viz kapitola 6.5).



Obr.6.1 Celková koncepce softwaru – blokové schéma

6.2.2 PCI karta

Původní řídicí karta byla určena pro sběrnici ISA, která se ale již u moderních základních desek nepoužívá a kterou vytlačila rychlejší sběrnice PCI. Proto bylo pro náš počítač P4, 3GHz nutno vybrat kartu novou. Požadavkem byla rychlá vzorkovací frekvence (více než 80 kHz), dále 16 digitálních výstupů, alespoň 8 digitálních vstupů a v neposlední řadě podpora karty ze strany RT toolboxu.



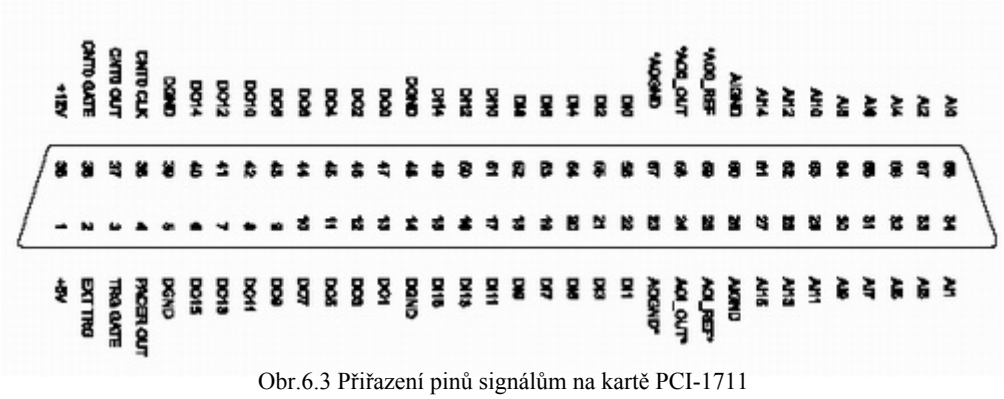
Obr 6.2 Řídicí karta PCI-1711 a převodní člen

Na základě těchto požadavků jsme vybrali kartu PCI-1711 od firmy Advantech (Obr 6.2). Základní parametry karty obsahuje tabulka 6.2.

Parametry	Upřesnění
16 analogových vstupů	Rozsah: ± 10V, doba převodu 10 µs
16 analogových výstupů	Rozsah: 0~ +10V
16 digitálních vstupů	Úrovně: Low~ 0.8Vmax, High~ 2.0Vmax
16 digitálních výstupů	Úrovně: Low~ 0.8Vmax, High~ 2.0Vmax
12bit A/D převodník	Maximální vzorkovací frekvence 100 kHz
Integrovaný programovatelný čítač	Base clock: 10 MHz
Integrovaný FIFO buffer na vzorky	Velikost: 1K

Tabulka 6.2 Parametry karty PCI-1711

Převodní člen (Obr 6.2) dodávaný ke kartě umožňuje připojení dvacetižilového samopřezného konektoru pro vstup i výstup do interface. Připojení převodního členu k PCI kartě je realizováno datovým kabelem, dodávaným výrobcem karty. Přiřazení pinů karty jednotlivým signálům je na obrázku 6.3



Obr.6.3 Přiřazení pinů signálům na kartě PCI-1711

Abychom v řídicí aplikaci dokázali přistupovat při čtení a zápisu dat k odpovídajícím prostředkům na kartě, potřebujeme znát převodní tabulku vstupů a výstupů kartě se vstupy a výstupy v RT Toolboxu (viz tabulka 6.3).

Adventech PCI-1711	Real Time Toolbox
A/D 0 .. A/D 15	Inputs 1 .. 16
D/A 1 .. D/A 2	Impuls 1 .. 2
D/I 0 .. D/I 15	Inputs 17 .. 32 in bit mode Inputs 17 .. 20 in nibble mode Inputs 17 .. 18 in byte mode Inputs 17 .. 4 in word mode
D/O 0 .. D/O 15	Inputs 3 .. 18 in bit mode Inputs 3 .. 6 in nibble mode Inputs 3 .. 4 in byte mode Inputs 3 in word mode

Tab. 6.3 Převodní tabulka kanálů :PCI karta-TR Toolbox

6.2.2.1 Praktické zkušenosti s kartou

Karta je stabilní v prostředí Matlab, snadno inicializovatelná a přístupná pomocí funkcí RT toolboxu. Pokud však dojde po inicializaci karty k odpojení datového kabelu Dx1 (viz Obr. 6.2) od převodní desky, může nastat kolizní stav, který vyústí v netečnost systému na veškeré vstupní podněty (nutnost restartovat počítač). Příčina souvisí s neošetřenou správou prostředků karty v OS Windows a použitými ovladači. Kabel však není nutno za provozu odpojovat, proto tento stav může nastat pouze neodbornou manipulací.

6.2.3 Programové prostředí a použité toolboxy

Aplikace je navržena a testována v prostředí Matlab 6.5. Využívá základních matematických funkcí, GUI rozhraní a přídavných toolboxů. Pro orientaci v popisu řídicích a vizualizačních funkcí uvedu základní vlastnosti a stručný popis možností těchto dvou toolboxů, avšak pouze v souvislostech, které se přímo týkají řídicí aplikace pro model helikoptéry. Problematika obou toolboxů je mnohem obsáhlejší a případné zájemce o komplexní popis odkazuji na použitou literaturu [10], [11] a referenční manuály.

Realtme Toolbox

Software vytvořený firmou Humusoft, použitá verze 3.11. Zprostředkovává komunikaci v reálném čase mezi I/O zařízeními a jádrem Matlabu. Mezi základní vlastnosti patří synchronizace přenášených dat, jednotlivý i hromadný zápis a čtení dat, práce s historií a správa prostředků I/O zařízení prostřednictvím ovladače. RT toolbox komunikuje prostřednictvím časovačů, které je možno nastavit pomocí funkce rtdef a následně skrze daný časovač číst nebo zapisovat data v intervalech vzorkování, určených pro každý časovač zvlášť.

V aplikaci se využívá vlastnosti toolboxu zvané historie dat, která umožňuje nadefinovat velikost zásobníku pro čtení i zápis, a tedy pracovat s daty v blocích, což je výhodné z hlediska zpracování i úspory výpočetního výkonu. V aplikaci se využívá těchto funkcí:

rtload('hwdrvr',addr,opt)
Nahraje ovladač hardware *hwdrvr* s adresou I/O definovanou v *addr* a vektorem parametrů *opt.*

rtunload
Vymaže všechny ovladače hardware z paměti. Časovače a proměnné historie zůstávají.

rtdef(t,'type',per,ch)
Definuje časovač *t* s vzorkovací frekvencí *per* a přiřadí ho kanálu *ch*. Typ musí být *In* nebo *Out*.

rtdef('hvar',len,width,mode)
Definuje proměnnou historie *hvar* o délce vzorku *len*, šířce kanálu *width* a chováním určeným podle *mode*.

rtclear
Vymaže všechny definované časovače a proměnné historie z paměti.

rtlink(t,'hvar')
Přiřadí proměnnou historie *hvar* k časovači číslo *t*.

rtunlink('hvar')
Zruší přiřazení proměnné historie *hvar* k jejímu zdrojovému časovači.

rtwr('hvar',y)
Zapíše proměnnou *y* do proměnné historie *hvar.*

rtrd(t, 'property')
Přečte proměnnou *property* z časovače číslo *t*.

rtrd('hvar', 'property')
Přečte proměnnou *property* z proměnné historie *hvar.*

restart all
Spustí všechny definované časovače.

rtstop all
Zastaví všechny definované časovače.

rtflush
Vyprázdní všechny definované proměnné historie a nastaví ukazatel na první vzorek každé proměnné.

Příklady použití výše uvedených funkcí jsou popsány v kapitolách popisujících vlastní funkce řídicí aplikace.

Virtual Reality Toolbox

Představuje vizualizační nástroj pro práci s vektorovou 3D grafikou. Pracuje s pojmem virtuální svět, který představuje fyzicky soubor s příponou wrl, jehož obsahem je seznam objektů, prvků a jiných seskupení, včetně pravidel jejich chování, vzhledu a mnoha dalších atributů. Pracoval jsem s verzí toolboxu 3.1.

Nádstavbou toolboxu je VRL Editor**,** který umožňuje editaci a vytváření třírozměrných modelů. Tento nástroj jsem použil při vytvoření kostry modelu, jejím pokrytí polygony a texturami, dále k nastavení atributů modelu i samotného virtuálního světa. Pro komunikaci s VR toolboxem byly použity tyto funce:

myworld= vrworld('filename')
Vytvoří objekt *vrworld* asociovaný s virtuálním světem načteným ze souboru *filename.*

vrsetpref('preference_ name','preference_value')
Nastaví preferenci *preference_name* na hodnotu *preference_value.*

mynode=vrnode(vrworld_object,'node_name')
Vytvoří ukazatel na nod v definovaném stětě.

myfigure=vrfigure(vrworld)
Vytvoří ukazatel na figure zobrazující svět *vrworld.*

vrdrawnow
Obnoví informace ve vrfigure na aktuální.

vrview('filename', - internal)
Vytvoří virtuální svět asociovaný se souborem wrl, otevře ho a spustí v interním prohlížeči.

Příklady použití výše uvedených funkcí jsou popsány v kapitole 6.2.7.

6.2.4 Vzorkovací frekvence a propustnost jádra

Jak již bylo naznačeno, RT toolbox používá časovače k opakovanému přístupu k I/O zařízení. Časovače jsou spouštěny s pevně definovanými periodami. Je důležité si uvědomit, jakým způsobem dochází k synchronizaci a dělení času CPU mezi jednotlivé časovače a zbylé procesy. Následující vysvětlení má napomoci pochopit logiku definování časovačů ve funkci INITrt a INITcitace (kapitola 6.1.5).

Jádro RT toolboxu využívá k synchronizaci pouze jeden HW časovač, standardně vestavěný do PC. Tento hlavní HW časovač určuje periodu, ve které se bude volat hlavní řídicí procedura RT toolboxu. Časování ostatních, uživatelem definovaných časovačů, je určováno programově podle hlavního časovače. V každém intervalu HW časovače jsou kontrolovány ostatní timery a v případě potřeby je provedena příslušná akce. Je-li potřeba provést současně dvě akce, provede se ta s vyšší prioritou a druhá je zpožděna do dalšího cyklu. Prioritu časovače určuje programátor při jeho definování přidělením číselného identifikátoru. Časovač s nejmenším identifikátorem má nejvyšší prioritu. Požadavek od časovače s nižší prioritou může být dočasně pozastaven, vznikle-li ve stejném okamžiku požadavek časovače s prioritou vyšší.

RT toolbox konzumuje část času CPU pro provedení svých interních operací, což však v praxi zabírá jen malou část doby běhu celého procesu. Implicitní nastavení vzorkovací periody jádra je v RT toolboxu nastaveno na 1ms pro všechny definované časovače, což je vzhledem k námi požadované

rychlosti čtení dat a řízení nedostačující. Je možno si nadefinovat periodu vzorkování pro každý časovač zvlášť a základní perioda se odvodí od nejrychlejšího z nich.

Obecně platí, že čím rychlejší je perioda vzorkování, tím rychlejší je možno navrhovat řídicí smyčku a získávat přesnější informace čtením dat. Vzhledem ke způsobu čtení a zápisu nebylo potřeba zavádět periody různé pro každý ze tří použitých časovačů (podrobnosti viz následující kapitola), nicméně jsou v programu definovány jako konstanty. Jejich změnou snadno změníme periodu vzorkování zvoleného časovače. Námi zvolená základní perioda vzorkování je 0.02 ms u všech použitých časovačů. Zaváděním kratších period vzorkování se snižuje výkon CPU. Při takto zvolené periodě to představuje zátěž procesoru asi 15%.

Při volbě základní periody 1µs dochází ke zhroucení systému. To je způsobeno skutečností, že doba běhu procedur se blíží zvolené hlavní periodě a tato situace není v RT toolboxu korektně ošetřena.

6.2.5 Funkce pro čtení a zápis

Funkce pro čtení a zápis představují v programu základ pro komunikaci s vrtulníkem na nejnižší úrovni dané aplikační vrstvy. Jejich prostřednictvím se provádí zápis a čtení dat na řídicí kartě. Zápisové a čtecí cykly se řídí časovými rámci a pravidly pro přístup k sběrnici popsanými v kapitole hardware. Úplný výpis kódu funkcí je součástí přílohy. Uvedu stručnou konstrukci a popis činnosti následujících funkcí:

INITrt

void initrt (void)
Vyčistí paměť od všech nadefinovaných časovačů a zruší zavedené ovladače v TR toolboxu. Provede inicializaci karty PCI-1711 a zavede pro ni ovladač do paměti příkazem :
rtload('pci1711',65535,[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1]).
Nadefinuje čtyři časovače a jim odpovídající historie. První časovač zapisuje na kartu data pro řízení, druhý zapisuje informace o povolení čtení, respektive zápisu. Třetí posílá na výstup informace o adrese a čtvrtý čte z karty informace o poloze vrtulníku.
Všechny časovače mají nastavenu periodu vzorkování 0.02 ms. Uvedu příklad definice časovače pro zápis řídicích dat. Zbylé časovače jsou zavedeny stejným způsobem.

Ts_data= 0.00002;	- perioda vzorkování
Delka_historie_data=6	- délka historie pro data
rtdef(1,'out',Ts_data,[3 4 5 6 7 8 9 10]);	- časovač s prioritou1, provádí zápis dat s danou periodou vzorkování na výstupy 3-10.
rtdef('all',delka_historie_data,8,0);	- historie dané délky s aliasem 'all'
rtlink(1,'all');	- svázání časovače s historií

Po zavedení této funkce je možno používat funkce pro čtení *READcidlo* a pro zápis *WR1*.

INITcitace

void initcitace (void)
Funkce slouží k inicializaci výstupních registrů a je nezbytné ji spustit ještě před funkcí *initrt*. Registry pracují jako programovatelný generátor monostabilního impulsu. Prvních 5 registrů

slouží k řízení serv a motoru. Je však potřeba inicializovat všech 6 registrů, aby nedocházelo k rozpadu komunikace.
Po jedné inicializaci pracují registry korektně až do výpadku napájení. Inicializace probíhá zápisem iniciačního slova postupně pro první tři registry na adresu A1 a poté pro zbylé tři registry na adresu A2, jak ukazuje tabulka 6.3. Funkce ihned po inicializaci nastaví registry na počáteční hodnotu 0. Kritický je pátý registr, určující výkon motoru, který musí být nastaven na nulu, aby nedošlo k poškození modelu nebo k úrazu.

Adresa	Registr číslo	Iniciační slovo
A1 [1 1 0 0 0]	Registr 1	[0 1 0 0 1 1 0 0]
	Registr 2	[0 1 0 0 1 1 0 1]
	Registr 3	[0 1 0 0 1 1 1 0]
A2 [1 1 1 0 0]	Registr 4	[0 1 0 0 1 1 0 0]
	Registr 5	[0 1 0 0 1 1 0 1]
	Registr 6	[0 1 0 0 1 1 1 0]

Tabulka 6.3 Inicializace výstupních registrů

Funkce si pro účely inicializace definuje vlastní časovače způsobem popsaným výše a po svém skončení je ruší.K nastavení hodnot registrů využívá funkci *wr*.

WR1

void wr1(byte kanál, int hodnota)
Pomocí této funkce lze ovládat serva a motor tím, že se zapíše požadovaná hodnota na vybraný kanál. Podle čísla kanálu se vybere příslušná adresa (viz tabulka 6.4). Dekadická hodnota se převede na binární. Vytvoří se matice historie pro data, adresu a řízení čtení/zápis. Jsou spuštěny časovače a zapíše se historie z matic na výstup karty. Po zapsání posledního záznamu historie jsou časovače zastaveny a vynulovány.

Povolený rozsah hodnot je 0-256, v případě zadání většího čísla než 256 nebo menšího než 0 je hodnota oříznuta na krajní číslo intervalu. Pokud je zadán neplatný kanál, není zapsána žádná hodnota.

Kanál	Akční člen	Adresa
1	Servo1	[0 0 0 0 0]
2	Servo2	[1 0 0 0 0]
3	Servo3	[0 1 0 0 0]
4	Servo4	[0 0 1 0 0]
5	Motor	[1 0 1 0 0]
6	Nepřiřazeno	[0 1 1 0 0]

Tabulka 6.4 Adresy kanálů pro zápis

WR

void wr (byte kanál, bin hodnota)
Plní stejnou úlohu jako wr1. Zapisovaná hodnota se zadává jako binární číslo, a tudíž nemusí být převáděna uvnitř funkce. Nepoužívá se historie dat, ale pro zápis každého bitu je definován samostatný časovač. Slouží pouze k nastavení prvotních hodnot po inicializaci výstupních čítačů ve funkci *initcitace*.

READcidlo

int readcidlo (byte kanál)
Provede čtení dat ze zvoleného kanálu, reprezentujícího některý ze sedmi snímačů polohy nebo vnějších a kontrolních registrů. Funkce tedy slouží k získávání informací o poloze vrtulníku, případně k získání dat potřebných pro diagnostiku čtením kontrolních registrů. Podle čísla kanálu je vybrána příslušná adresa, viz tabulka 6.5, a přečtena binární hodnota, která je převedena na dekadickou. Funkce readcidlo vrací tuto přečtenou hodnotu.

Je nutno si uvědomit, že pokud čteme informaci pocházející z inkrementálního čidla, je výstupem funkce číslo reprezentující velikost změny oproti poslednímu čtení. Dostáváme tudíž pouze relativní informaci o poloze. K získání absolutní polohy je třeba na začátku znát (zvolit) polohu modelu a pomocí přírůstků vypočítávat její aktuální hodnotu.

Kanál	Typ informace	Adresa
1	Vysunutí spojovací tyče	[0 1 1 0 1]
2	Podélný náklon – čidlo dole	[1 0 1 0 1]
3	Příčný náklon – čidlo dole	[0 0 1 0 1]
4	Otáčky motoru	[1 1 0 0 1]
5	Kurs – čidlo na modelu	[0 1 0 0 1]
6	Podélný náklon – čidlo na modelu	[1 0 0 0 1]
7	Příčný náklon – čidlo na modelu	[0 0 0 0 1]
8	Vnější registr P	[1 1 0 1 1]
9	Vnější registr K	[0 0 1 1 1]
10	Vnější registr &	[1 0 1 1 1]
11	Vnější registr J	[0 1 1 1 1]
12	Vnější registr F	[1 1 1 1 1]
13	Kontrolní registr 1. kanálu	[0 0 0 1 1]
14	Kontrolní registr 2. kanálu	[1 0 0 1 1]

Tabulka 6.5 Adresy kanálů pro čtení

6.2.6 Funkce pro úpravu informací o poloze

Číslo získané pomocí funkce *wr1* musí být dále zpracováno, abychom získali odpovídající informaci o poloze. To zajišťuje funkce *check.* Vzhledem k přítomnosti rušení v komunikačním kanálu dochází často i v případě, že se vrtulník nehýbe, k přenosu malých změn polohy. K odstranění tohoto jevu používám funkci *filtr.*

check
int check (int hodnota)

Jak již bylo zmíněno dříve, při čtení údajů o poloze vrtulníku z čidel získáváme pouze posledních 8 bitů jeho polohy. Je jasné, že rozsah možných hodnot je podstatně větší. Musíme tedy provést rekonstrukci zbylých bitů. Funkce check provádí rekonstrukci po vyčíslení rozdílu mezi aktuální a minulou hodnotou. Je-li rozdíl vetší než polovina rozsahu registru (v našem případě 128), pak je nutno provést korekci vyššího řádu. Jediným omezením pro rekonstruktor je určitá maximální doba, za kterou je potřeba provést opětovné čtení registru, ale to zajišťuje dostatečná vzorkovací frekvence časovače pro čtení.

filtr

real filtr (real hodnota1, real hodota2 ,real hodnota3)

Jednoduchý klouzavý filtr, umožňuje odstranit nežádoucí rušení v případě, že se vzorek představující chybu způsobenou rušením nebude opakovat vícekrát za sebou.

6.2.7 Procedury pro vizualizaci a virtuální model

K vizualizaci se využívá Virtual Reality toolbox. Model a virtuální svět je uložen v souboru s příponou wrl. Vrtulník i jeho jednotlivé části jsou představovány objektovými proměnnými s množstvím nastavitelných parametrů.

Pro pochopení práce s modelem ve VR světě uvedu příklad vizualizace změny polohy. Model vrtulníku je objekt nazvaný ‘vrtulník‘, který má kromě dalších také vlastnosti translace a rotace, například podle osy Z. Pomocí funkce *vrnode* získáme ukazatel na objekt ‘vrtulník’ a můžeme nastavit jeho parametry rotaci a translaci pomocí funkce *setfield*. Reprezentace obecné rotace objektu v RT toolboxu je definována jako směrový vektor (osa rotace) a otočení podle tohoto vektoru. Z čidel však získáváme údaje o rotaci vrtulníku podle jednotlivých os, což představuje rotaci definovanou v kartézské soustavě souřadnic pomocí tří úhlů. Mezi těmito dvěma soustavami existuje jednoznačná transformace, pomocí níž lze úhly natočení získané z čidel transformovat do soustavy používané v VR Toolboxu a vrtulník vizualizovat.

Výpočet transformace však při každé změně polohy vyžaduje sestavení transformačních matic, výpočet a zaokrouhlení výsledku. To má vliv na rychlost vlastní vizualizace (zpomalení až o 20%). Proto jsem hledal možnost, jak polohu modelu zobrazovat bez nutnosti složité transformace souřadnic. Lze to provést pomocí hierarchie objektů, které model představují. Objekt ‘vrtulník’ je otcem objektu nazvaného ‘podvrtulník1‘ a ten je otcem objektu ‘podvrtulník2’, který obsahuje samotný model. V takovéto struktuře lze samostatně a bez nutnosti transformace nastavit rotaci podle osy X u obj. ‘podvrtulník2’, podle osy Y u obj. ‘podvrtulník1’ a konečně rotaci podle osy Z u obj. ‘vrtulník’. Tohoto postupu se využívá v proceduře *makevisual*.

initvisual

initvisual (void)

Procedura je volána při spuštění aplikace a načítá soubor virtuálního světa ‘heli.wrl‘. Nastaví prohlížeč pro vizualizaci v jednoduchém okně funkcí *vrsetpref* a následně definuje nody nutné pro předávání informací o poloze. Načtení souboru wrl a definice nodu popisuje tento příklad:

myworld = vrworld('heli.wrl');	- otevře soubor wrl
vrsetpref('DefaultPanelMode','off');	- nastaví okno prohlížeče
open(myworld);	- nahraje svět do paměti
mynode = vrnode(myworld, 'vrtulnik');	- definuje ukazatel (nod) na objekt ‘vrtulnik’
f = vrfigure(myworld);	- ukazatel na figure představující obecné parametry virtuálního světa
set(f,'Transparency','on');	- povolení průhledných textur a polygonů na objektech světa daného ukazatelem f.

makevisual

makevisual (byte cislo_pohledu)
Pracuje s nody zavedenými pomocí procedury *initvisual.* Otevře okno s vizualizací virtuálního světa, pokud již není otevřeno, předá parametry o poloze modelu a nakonec jej vykreslí. Provádí se při každé změně polohy vrtulníku.

Předávání informací o poloze modelu se provádí například následovně:
setfield(mynode, 'rotation',a);
Je nastaven parametr 'rotation' u nodu 'mynode' definovaného způsobem popsaným v proceduře *initvisual,* na hodnotu proměnné a . Tato proměnná je tedy vektor o čtyřech souřadnicích, kde první tři definují osu otáčení a čtvrtá úhel otočení podle této osy. Požaduji-li tedy otočit model podle osy Z o 60°, bude vector *a* mít tvar: a=[0 0 1 60].

Obě popsané procedury se nacházejí přímo v těle hlavního programu (soubor mor.m).

6.2.8 Funkce pro diagnostiku a kontrolu hardwaru

Koncepce modelu ovládaného dálkově pomocí počítače je rozsáhlý systém, ve kterém mohou vznikat velmi snadno chyby způsobené nejčastěji přerušením některého z komunikačních kanálů nebo výpadkem řídicích registrů či některého z programovatelných polí Xilinx. Hledání příčin jednotlivých selhání je usnadněno základní diagnostikou, která omezí oblasti vzniku chyby na tři základní možnosti. Jedná se o selhání komunikačního kanálu jedna nebo dva (ověřuje funkce *Test_kanal*) a dále o testování výstupních registrů funkcí *Test_vnej_reg.*

Test_kanal

bool test_kanal (byte cislo_kanalu)
Testuje funkčnost kanálu jedna nebo dva. Funkce vrací hodnotu odpovídající stavu kanálu, OK nebo FAILED. Kontrola kanálu probíhá následovně:

Pomocí funkce *readcidlo* čteme údaje z kontrolního registru prvního, respektive druhého kanálu. Tyto hodnoty představují počty přenosů od posledního čtení. Přečte se 500 hodnot a uloží se do jednorozměrného pole. Funkce najde minimum z těchto hodnot a spočte jeho četnost. Pokud je četnost nejmenšího prvku v poli pěti set vzorků větší než sto, znamená to selhání komunikačního kanálu. V opačném případě je kanál funkční.

Možné příčiny selhání:

Nejčastěji se jedná o nepřipojení propojovacích vodičů mezi vysílačem a přijímačem. Druhou možností je nefunkční vysílač, jelikož část prvního kanálu, který spojuje model vrtulníku s přijímacími registry, je vedena vzduchem.

Test_vnej_reg

bool test_vnej_reg (void)
Registry obvodu Xilinx přístupné čtením kanálu 8 – 12 obsahují identifikační řetězec 'PK&JF'. Funkce přečte informace z těchto kanálů a získané hodnoty porovná s požadovanými. Pokud se neshodují, vrací funkce stav FAILED.

Možné příčiny selhání:

K chybě dochází v případě vypnutého napájení nebo vadného či neinicializovaného obvodu Xilinx.

6.3 GUI aplikace

Grafické uživatelské prostředí bylo vytvořeno pomocí editoru prostředí Matlab. Editor je koncipován podobně jako například tvorba GUI ve Visual C++. Informace o rozmístění ovládacích prvků a nastavení některých parametrů těchto objektů se ukládá do souboru s příponou *fig*. Soubor s GUI aplikací má název *senzor_read.fig.* Tomuto zdrojovému souboru odpovídá stejnojmenný m-file, který obsahuje funkce volání grafických prvků v případě, že nastane nějaká akce, zmáčknutí tlačítka a podobně.

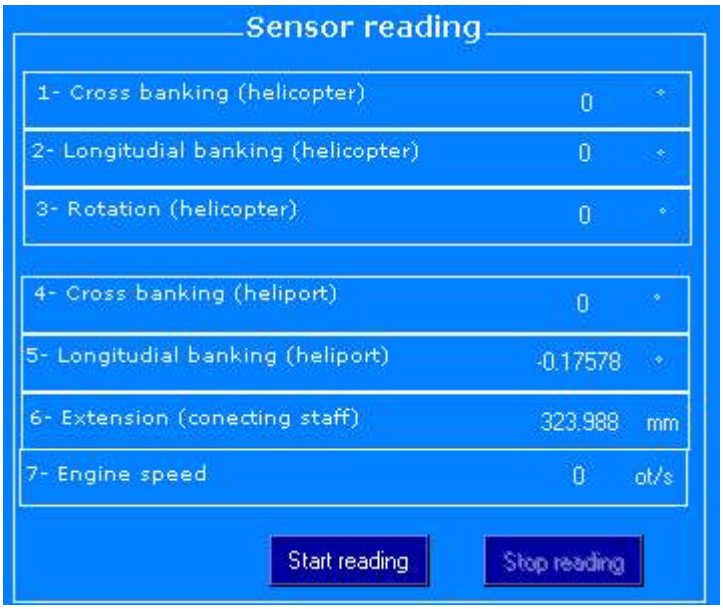
Ovládací prvky představují v programu objektové proměnné, je možné s nimi pracovat pomocí funkcí *set* a *get.* Pomocí funkce *set* lze nastavovat jednotlivé parametry vybraného objektu. Funkce *get* slouží k získávání informací o objektu. Každý prvek obsahuje speciální proměnnou *UserData*, pomocí které je možno předávat vzájemně data mezi aplikacemi a prvky. Celkový pohled na spuštěnou aplikaci je na obrázku 6.4.

Obr. 6.4 GUI aplikace – celkový náhled

6.3.1 Informace o poloze vrtulníku

Čtením senzorů na vrtulníku a heliportu získáváme data o poloze, která jsou po zpracování zobrazována v oddíle *Sensor reading* (obrázek 6.5). Při zapnutí čtení dochází k nastavení výchozí polohy a všech čidel na nulu. Je tedy potřeba model na heliportu nastavit do výchozí polohy a poté spustit čtení. Výchozí poloha bude považována za počátek souřadného systému. Horní skupina ukazuje údaje z čidel na helikoptéře, zatímco dolní z čidel na heliportu, s výjimkou senzoru pro otáčky motoru.

Úhly jsou vypisovány ve stupních, délka posuvu v milimetrech a rychlost motoru v otáčkách za sekundu. Přesnost zobrazení je na tři desetinná místa. Při výpadku kanálu 1 jsou čtena pouze data z kanálu 2.

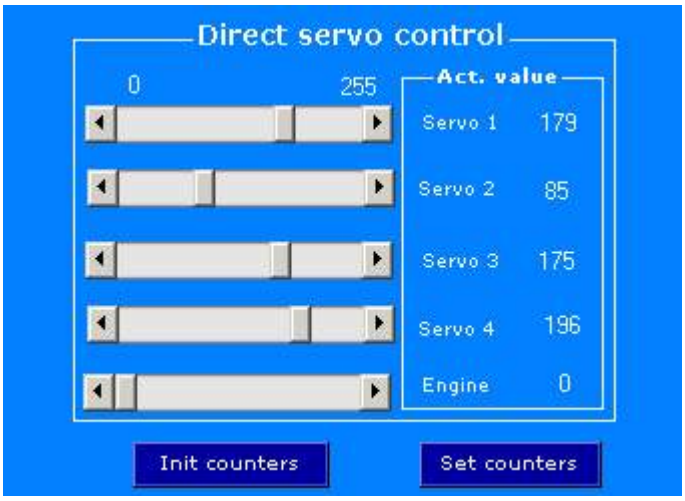


Obr.6.5 GUI aplikace- čtení senzorů

6.3.1 Ovládání servomechanismů

Serva na vrtulníku lze přímo ovládat z aplikace a vyzkoušet tak jejich funkčnost. Aktuální hodnota serva nastavená pomocí posuvníku se zobrazuje napravo. Diskrétní krok pro posouvání je nastaven na hodnotu dva dílky. Je přednastavena počáteční hodnota pro všechna serva 128, pouze pro motor je na počátku nulová hodnota, aby nedošlo k poškození modelu. Tuto konfiguraci lze v případě potřeby obnovit tlačítkem *Set couters*.

Při spuštění aplikace dochází automaticky k inicializaci výstupních registrů. Pokud by v průběhu práce s modelem došlo k rozpadu komunikace, lze inicializaci provést tlačítkem *Init counters*. Při zapnutém demo režimu je servo 4 ovládáno regulátorem a lze sledovat prováděné akční zásahy. Oddíl pro přímé řízení servomechanismů je na obrázku 6.6.



Obr 6.6 GUI aplikace – řízení servomechanismů

6.3.1 Diagnostika

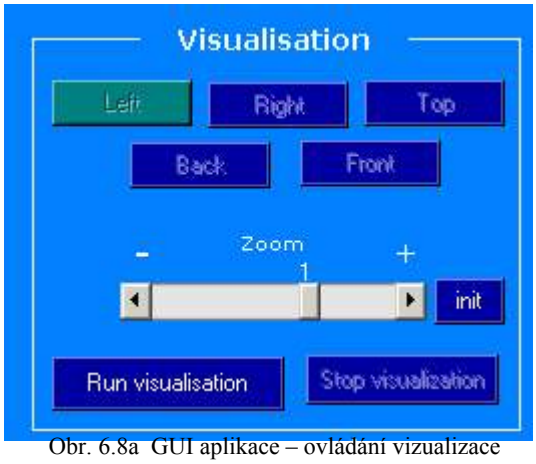
Diagnostika (obr.6.7) je také spuštěna bezprostředně po startu aplikace a indikuje provozní stav jednotlivých kanálů a výstupních registrů. Pokud dochází při provozu k výpadkům údajů z některých čidel nebo k selhání řízení pomocí serv, je možno hledat příčinu podle diagnostiky. Získat aktuální diagnostické informace lze pomocí tlačítka *Refresh* a během této operace nelze provádět čtení ani zápis dat do modelu. To je ošetřeno zrušením funkčnosti kritických prvků po dobu testování.

Obr. 6.7 GUI aplikace - diagnostika

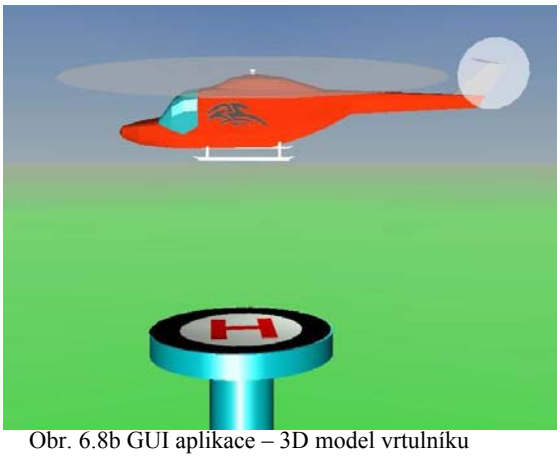
6.3.1 Vizualizace

Oddíl vizualizace (obr 6.8a) umožňuje pozorovat chování modelu pomocí jeho 3D reprezentace. Je nutno vizualizovat na základě informací o poloze, proto se spuštěním vizualizace automaticky spouští i čtení polohy ze senzorů. Otevře se nové okno (obr 6.8.b) podle parametrů (perspektiva pohledu) nastavených před jeho spuštěním. V průběhu vizualizace lze měnit přiblížení modelu nebo nastavit na původní hodnotu tlačítkem *init*. K vykreslování polohy vrtulníku v reálném čase dochází tehdy, je-li okno s vizualizací aktivní.

Pokud je vizualizace zastavena, okno s 3D modelem se zavře a ukončí se také čtení polohy.



Obr. 6.8a GUI aplikace – ovládání vizualizace



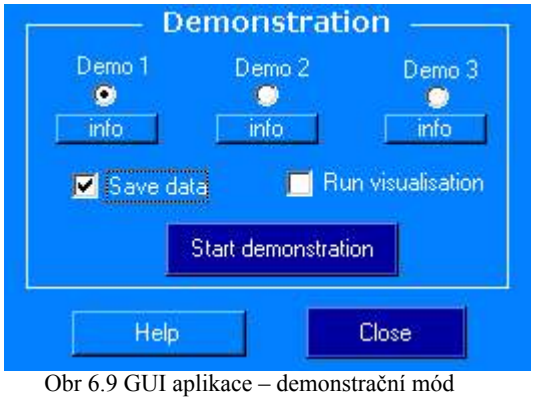
Obr. 6.8b GUI aplikace – 3D model vrtulníku

6.3.1 Demonstrační mód

Má umožnit předvedení základních letových módů modelu po jeho plné identifikaci. Nyní je aktivní pouze ukázka *Demo1*. Popis ukázky se zobrazí po kliknutí na tlačítko info. Před spuštěním demonstrace lze přednastavit, zda se mají ukládat data z průběhu letu nebo zda má být spuštěna vizualizace. Zapnutá vizualizace nemá podstatný vliv na snížení výkonu systému a téměř neovlivňuje ani rychlost řídicí smyčky.

V průběhu je vyřazena většina ovládacích prvků aplikace, aby jejich nevhodným aktivováním nedošlo k havárii helikoptéry. Výjimkou je tlačítko pro okamžité zastavení demonstrace, které vypne motor a řízení serv.

Data o poloze jsou ukládána do matice *demodata1,* jejíž řádky představují údaje sejmuté jednotlivými senzory. Pohyb po sloupcích odpovídá pohybu po časové ose. Akční zásahy jednotlivých servomotorů jsou ukládány obdobným způsobem do matice *demodata2.* Průběh letu lze z těchto údajů snadno rekonstruovat a vizualizovat. Ovládací panel je na obrázku 6.9.



Obr 6.9 GUI aplikace – demonstrační mód

6.3.1 Praktické zkušenosti s aplikací

Aplikaci je potřeba ukončit tlačítkem *Close*, aby došlo k uvolnění použitých proměnných z paměti a k deaktivaci toolboxů. Data uložená v průběhu letu zůstávají ve workspace i po ukončení aplikace. Při spuštěném demo režimu, kdy pobíhá vizualizace, ukládání dat a samotném řízení je vhodné nepouštět na pozadí žádné další aplikace, tento režim je výpočetně náročný na zdroje i CPU počítače.

7.Regulace a ukázka letu

Pro ověření funkčnosti a demonstrací možností, které přináší řídicí aplikace a nově upravený model, jsme vytvořili jednoduchou letovou ukázku. V následujícím textu popíšeme provozní podmínky nutné pro úspěšné předvedení ukázky, dále popíšeme systém řízení a časování akcí modelu v průběhu letu.

7.1. Provozní podmínky

Před spuštěním letové ukázky je potřeba zkontrolovat stav vrtulníku a jeho umístění na heliportu. Listy rotoru nesmějí být plně utaženy a je potřeba je vzájemně narovnat, jinak mohou při několika prvních otáčkách při startu modelu kolidovat se spojovací tyčí. Na heliportu je vyznačena poloha umístění vrtulníku pomocí dvou červených pruhů, na něž mají dosednout ližiny helikoptéry. Bílá šipka označuje orientaci přídě.

Heliport je nutno umístit co nejdále od všech větších pevných ploch, aby nedocházelo ke zpětným vlnovým rázům, které negativně ovlivňují průběh letu. Implementovaná letová ukázka předpokládá fixaci spodních kloubů trenažéru! Bez této fixace je letová ukázka *demo1* neproveditelná, jelikož k udržení stability modelu je třeba implementovat regulátor pro komplexně identifikovaný systém.

Systém je náchylný na vnější zdroje rušení, proto je třeba v blízkosti modelu vypnout jeho možné zdroje. Ze zkušeností z provozu vyplývá, že se jedná o mobilní telefony, některé zdroje napájení stanic PC, jiné vysílače v modelářském pásmu a silnější vysílače obecně. Přestože jsme nenašli v okolí žádné zdroje rušení pracující na frekvencích blízkých frekvencím použitým při komunikaci mezi modelem a PC, někdy se rušení vyskytuje. Rušení způsobuje výpadky v řízení, což se projevuje kolísavým výkonem motoru, chybách v přenosu údajů z čidel a následné špatné interpretace polohy vrtulníku v řídicím softwaru.

7.2. Parametry modelu a způsob regulace

Regulace v rámci ukázkového letu *demo1* spočívá v řízení náklonu listů zadního vyrovnávacího rotoru vrtulníku, aby nedocházelo k jeho otáčení kolem osy rotace Z. Parametry modelu se mění podle toho, zda je vrtulník na zemi a působí na něj tření, nebo je ve vzduchu a tření způsobené pouze uchycením v trenažéru je minimální. Abychom si při regulaci vystačili pouze s jedním regulátorem, je potřeba provést start (rozumí se odlepení vrtulníku od země) co nejrychleji, aby regulátor pracoval co nejdříve ve správné oblasti. Dalšími parametry ovlivňujícími regulaci jsou rychlost otáček motoru a vztlak.

K řízení zadního rotoru jsme použili diskrétní verzi PID regulátoru. Obecný diskrétní PID regulátor má tvar:

$$R(z) = r_1 \frac{z-1}{z} + r_0 + r_{-1} \frac{z}{z-1}$$

Velikost akčního zásahu je omezena rozsahem natočení listů vrtule a díky přítomnosti integrační složky vzniká windup.
Implementace regulátoru v programu je následující:

Ekminula = Ek;	- ukládání předešlé hodnoty prop. složky
Ek=z-nn(5);	- určení nové proporcionální složky
kd = (Ek - Ekminula);	- určení derivační složky
ki = ki + Ek;	- určení integrační složky
zesileni=0.5;	- konstantní zesílení
windup = 250/zesileni;	- velikost omezení windup
ki=min(max(ki,-windup/konstInt), windup/konstInt);	- výpočet integrační složky
drift=150;	- posunutí zásahu do oblasti pracovního bodu
zasah = zesileni(Ek + konstInt*ki + konstDr*kd)+drift;*	- výpočet zásahu PID regulátoru v okolí

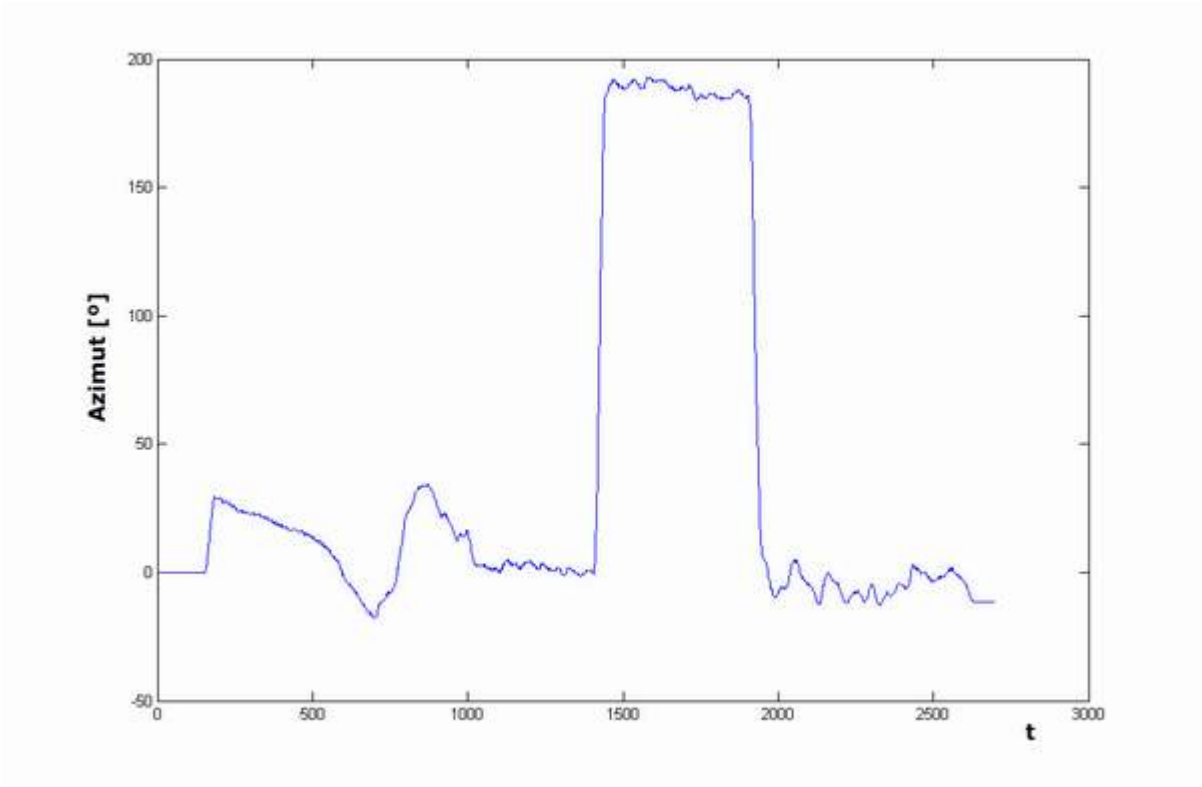
7.3 Rozvrh akcí a časování

Letová ukázka se skládá ze sedmi po sobě jdoucích časových intervalů. V průběhu každého z nich dochází k jedné z akcí typu změna výkonu motoru, náklonu listů nebo žádané hodnoty. Změny jsou prováděny postupně s proměnným krokem *x* dílků na časovou jednotku. Výjimkou je změna žádané hodnoty - ta probíhá skokem. Po celou dobu demonstrace je regulován zadní rotor helikoptéry. Časovou jednotkou je jeden programový cyklus, což odpovídá 0.031 sec.

Na obrázku 8.1 kde je vyobrazen skutečný průběh azimutu letové ukázky, kde jsou dobře patrné změny výše popisovaných řídicích parametrů. Časový rozvrh akcí popisuje tabulka 8.1.
Rozvrh a časování (tabulka 8.1) je sledující:

Čas	Akce
0 - 200	Nastaví výkon motoru na hodnotu 50 a čeká 200 cyklů., než dojde k odeznění vibrací na heliportu
400 – 700	Zvedne výkon motoru na hodnotu 100
700 - 1000	Zvýšíme výkon motoru na 250 a zároveň provedeme natočení listů hlavního rotoru až na maximum (255)
1400 – 1600	Režim visení trvající 400 cyklů ukončíme v čase 1400 změnou žádané hodnoty z 0 na 180 stupňů.
1900 – 2100	Po režimu visení a poté opět změna žádané hodnoty zpět na 0 stupňů.
2200 - 2700	Snižování výkonu motoru a náklonu listů hlavního rotoru na hodnotu nula.
> 2700	Dojde k vypnutí motoru a nastavení serv na defaultní hodnoty

Tab. 8.1 Časování akcí v ukázce demo1



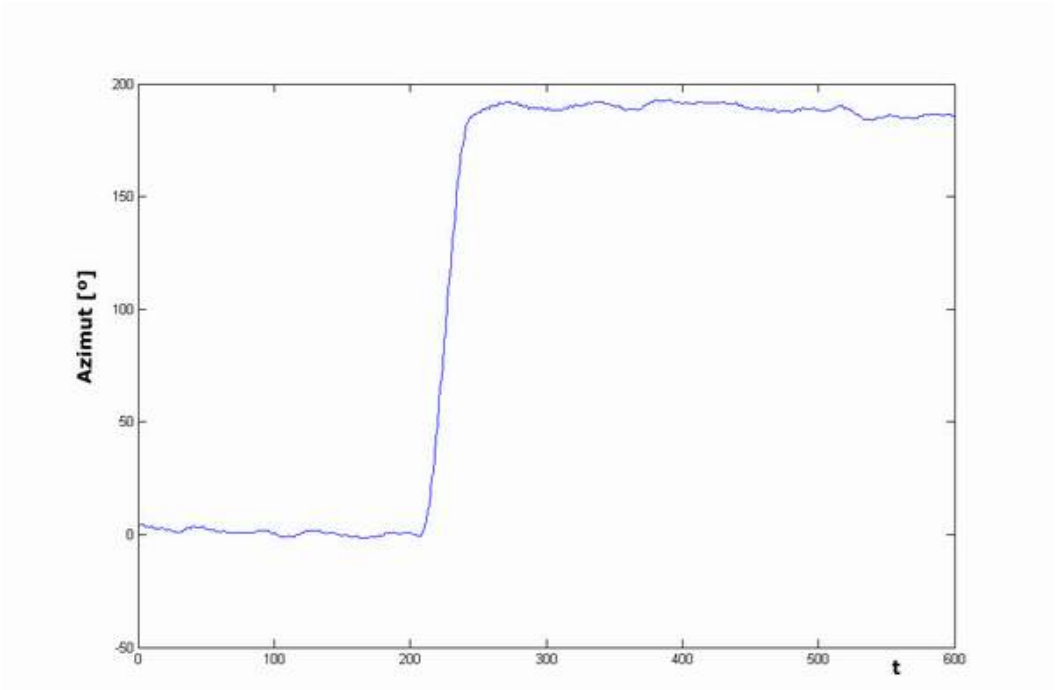
Obr. 8.1 Průběh azimutu

7.4 Bezpečnost při provozu

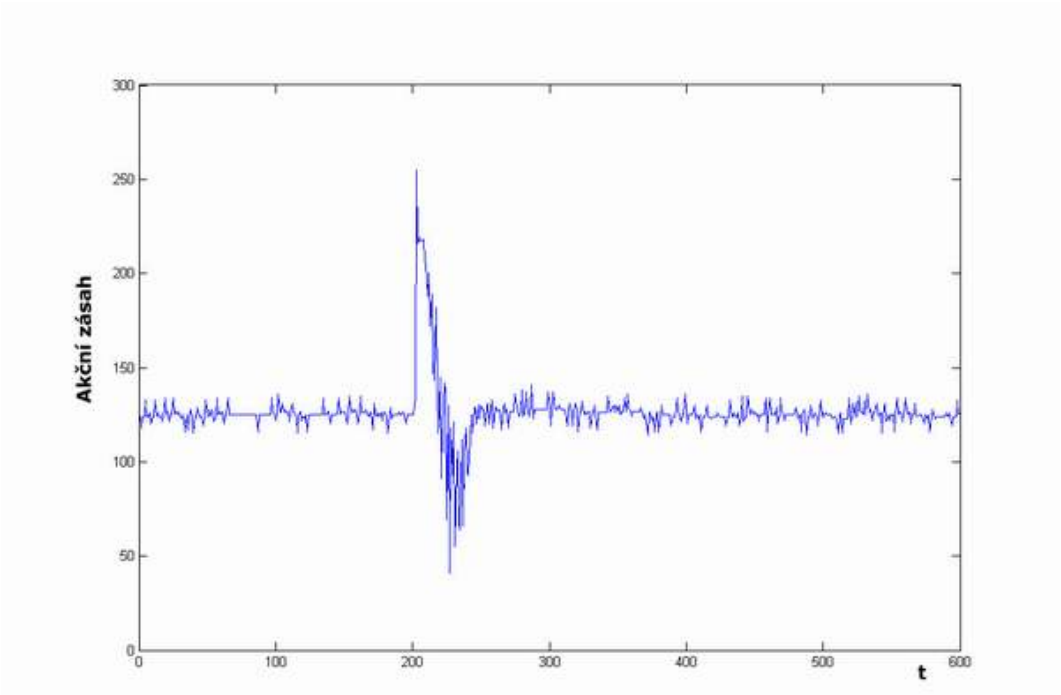
V průběhu letové ukázky je třeba dbát základních bezpečnostních pravidel. Je třeba, aby jedna osoba kontrolovala průběh demonstrace a byla připravena v aplikaci zastavit letovou ukázku. Tím se vypne motor a vrtulník může přistát autorotací. Je vhodné, aby byl dostupný vypínač hlavního zdroje napájení vrtulníku, pokud by nebylo možno vypnout motor pomocí aplikace (s touto situací jsme se dosud nesetkali).

7.5 Přechodové charakteristiky

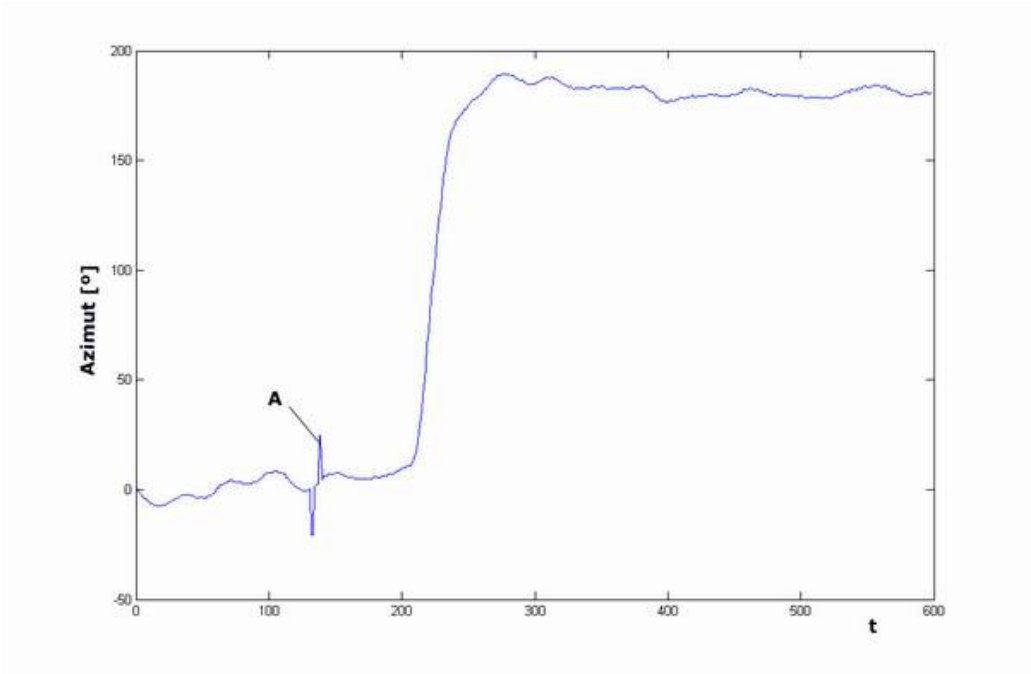
Obrázky 8.2 a 8.4 zachycují přechodovou charakteristiku systému jako odezvu na změnu žádané hodnoty z 0º na 180º v čase t = 200 a vypočtené regulační zásahy PID regulátoru (obr. 8.3 a 8.5). První tři uvedené průběhy nejsou ovlivněny rušením. Poslední tři obrázky zachycují typické chování systému při rušení (chybná hodnota vlivem rušení je označena písmenem A).



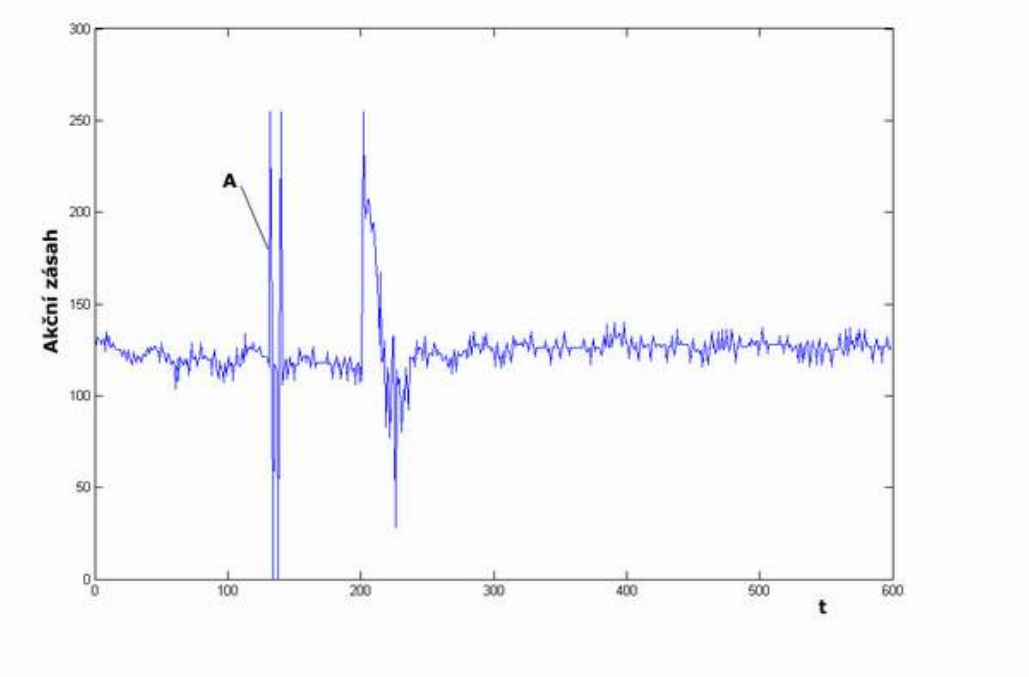
Obr. 8.2 Přechodová charakteristika bez vlivu rušení



Obr. 8.3 Regulační zásah bez vlivu rušení



Obr. 8.4 Přechodová charakteristika při rušení



Obr. 8.5 Regulační zásah ovlivněný rušením

8. Závěr

Hlavním cílem mé práce bylo vytvořit řídicí aplikaci, která by splňovala nároky pro řízení v reálném čase, současnou vizualizaci, diagnostiku celého systému a také předvést krátký letový program.

Realizovaná aplikace dovoluje ovládat veškeré prvky pro řízení modelu, sleduje jeho polohu v prostoru a je schopná třírozměrné vizualizace. Implementované diagnostické nástroje dovedou odhalit nefunkčnost jednotlivých komunikačních kanálů a některé další možné chyby. Modularita celého systému je základem pro snadnou implementaci pokročilých řídicích algoritmů, což bude možné po úplné identifikaci systému.

Laboratoře ČVUT FEL využívají již delší dobu pro některé regulační úlohy ustálených řešení v podobě programových prostředí, jejich modulů a postupů. Při tvorbě tohoto projektu jsem se snažil držet těchto zavedených pravidel. Programové prostředí Matlab a použité toolboxy umožňují komunikaci jádra programu s řídicí kartou v PC (v případě RT Toolboxu) a přesnou trojrozměrnou vizualizaci za použití Virtual Reality Tollboxu. Většina parametrů aplikace je snadno konfigurovatelná a bylo by možno ji bez větších problémů upravit i na jiný model.

Pro uvedení modelu do provozu bylo zapotřebí vykonat mnohem větší rozsah prací. Podílel jsem se zčásti na úpravách hardware týkajících se konstrukce trenažéru, restaurování samotného modelu i vytváření řídicí desky PCB č.1. Výroba hardwarových částí je časově náročná a musela být dokončena, aby bylo možno začít s vývojem software.

Ze zkušeností našich předchůdců jsme se poučili a námi navržené úpravy jsme se snažili realizovat s maximální robustností, aby v budoucnu nebylo nutno stále se opakovaně vracet k HW části modelu a korigovat neustále nově vznikající nedostatky.

Vyvrcholení mé práce představuje demonstrační letový program, který ukazuje a nastiňuje další možnosti práce s modelem. Je pouze jednoduchou ukázkou letových schopností vrtulníku, ale postačující k demonstračním účelům. Věřím, že po identifikaci modelu bude možné zvládnou složitější letové manévry.

Svoji část projektu jsem se snažil udělat co nejlépe. Do jaké míry se mi to podařilo, nechť posoudí ti, kteří budou v práci s modelem dále pokračovat.

Svým nástupcům přeji hodně zdaru při návrhu a testování složitých řídicích algoritmů, které - jak doufám - budou moci důkladně prověřit funkčnost modelu a trenažéru.

9. Reference

[1] Bury H.: Úpravy modelu helikoptéry. Diplomová práce ČVUT FEL, Praha 2005.

[2] Kocourek P. Přenos informace. Skriptum ČVUT FEL, Praha 2001.

[3] Horáček P. Systémy a modely. Skriptum ČVUT FEL, Praha 2000.

[4] John J. Systémy a řízení. Skriptum ČVUT FEL, Praha 1999.

[5] Krsek P. Řídicí systém modelu helikoptéry. Diplomová práce ČVUT FEL, Praha 1995.

[6] Fojtík J. Programový systém rychlé komunikace pro model helikoptéry. Diplomová práce ČVUT FEL, Praha 1995.

[7] Beneš P. Řízení helikoptéry. Diplomová práce ČVUT FEL, Praha 1993.

[8] GM Eletronic, s. r. o.: Součástky pro elektroniku. Katalog, Praha 2004.

[9] Advantech Co. Ltd [online].
<http://www.advantech.com>.

[10] Humusoft: Realtime Toolbox for use with Matlab User's manual. Version 3.0, 1999.

[11] The MathWorks [online].
<http://www.mathworks.com>.

[12] GES Electronics [online].
<http://www.ges.cz>.

[13] Asix [online].
<http://www.asix.cz>.

[14] Weiss, O. Katalog tesla [online].
<http://web.telecom.cz/oweisz/katalog.htm>.

[15] Popis programovatelných čítačů.
<http://dt.fme.vutbr.cz/~measure/C8254.htm>.

Příloha A Výpis funkcí pro řízení hardwaru

Zdrojové kódy jsou napsány jako m-files pro systém Matlab. Jsou zde uvedeny jednotlivé výpisy důležitých řídicích struktur včetně komentářů.

A.1 INITrt

```
function INITrt()

rtclear all;
rtclear;
rtunload;

%inicializace karty
rtload('pci1711',65535,[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1]);

Ts_rdrw=  0.00002;
Ts_data=  0.00002;          %timing a inicializace zápisu + adres

Ts_adresa=0.00002;

delka_historie_rdrw=6;
delka_historie_data=6;
delka_historie_adresa=6;

rtdef(1,'out',Ts_data,[3 4 5 6 7 8 9 10]);
rtdef('all',delka_historie_data,8,0);
rtlink(1,'all');

rtdef(2,'out',Ts_rdrw,[17 18]);
rtdef('rdrw',delka_historie_rdrw,2,0);
rtlink(2,'rdrw');

rtdef(3,'out',Ts_adresa,[11 12 13 14 15]);
rtdef('adr',delka_historie_adresa,5,0);
rtlink(3,'adr');

Ts_read=0.00004;            %timing a inicializace čtení
delka_historie_read=1;

rtdef(4,'in',Ts_read,[17 18 19 20 21 22 23 24]);
rtdef('read',delka_historie_read,8,1);
rtlink('read',4);
```

A.2 WR1

```
function wr1(CHno,value)

value=DECtoBIN(value);

switch CHno
   case 1,          addr=[0 0 0 0 0];
   case 2,          addr=[1 0 0 0 0];
   case 3,          addr=[0 1 0 0 0];          %adresování serv a motoru
   case 4,          addr=[0 0 1 0 0];
   case 5,          addr=[1 0 1 0 0];
   case 6,          addr=[0 1 1 0 0];
end

Adresa=[addr;  addr;   addr;   addr;   addr;      addr];

 if CHno < 4

   RdRw=[ 1 0;   1 0;   1 1;  1 0;   1 1;    1 1];

Else

   RdRw=[ 1 1;  1 0;   1 1;    1 0;    1 1;   1 1];
end;

Data=[ value;    value;    value;   1 0 0 0 0 0 0 0;   1 0 0 0 0 0 0 0;    1 0 0 0 0 0 0 0];


rtwr('adr',Adresa);
rtwr('rdrw',RdRw);
rtwr('all',Data);
%rtrd('adr');

rtstart all;
while (rtrd('adr','t'))<5 end;
rtstop all;
rtflush;
```

A.3 READcidlo

```
function [out]=READcidlo(CHno)

switch CHno
   case 1,                  %vysunutí tyče
     addr=[0 1 1 0 1];
   case 2,                  %podélný náklon tyče
     addr=[1 0 1 0 1];
   case 3,                  %příčný náklon tyče
     addr=[0 0 1 0 1];
   case 4,                  %otáčky motoru
```

```
      addr=[1 1 0 0 1];
   case 5,                    %kurs modelu
      addr=[0 1 0 0 1];
   case 6,                    %podélný náklon modelu
      addr=[1 0 0 0 1];
   case 7,                    %příčný náklon modelu
      addr=[0 0 0 0 1];
   case 8,                    %vnější reg. P
      addr=[1 1 0 1 1];
   case 9,                    % vnější reg. K
      addr=[0 0 1 1 1];
   case 10,                   % vnější reg. &
      addr=[1 0 1 1 1];
   case 11,                   % vnější reg. J
      addr=[0 1 1 1 1];
   case 12,                   % vnější reg. F
      addr=[1 1 1 1 1];
   case 13,                   %kontrolní reg. 1.kanálu
      addr=[0 0 0 1 1];
   case 14,                   %kontrolní reg. 2.kanálu
      addr=[1 0 0 1 1];
end

Adresa=[addr;  addr;  addr;   addr;   addr;  addr;];

RdRw=[ 0 1;   0 1;   1 1;    0 1;    1 1;   0 1];

rtwr('rdrw',RdRw);
rtwr('adr',Adresa);

rtstart all;
while (rtrd('adr','t'))<1 end;
a=rtrd('read');
out=bintodec(a);
rtstop all;
rtflush;
```

A.4 check

```
function [out]=check(in)
delta=in;
                                        %korekce podtečení
 if delta > 128
    delta=delta-256;
 end
                                         %korekce přetečení
  if delta < -128
     delta=delta+256;
  end
   out=delta;
```

A.5 INITcitace

```
function INITcitace()
rtclear all;
rtclear; rtunload;

%inicializace
rtload('pci1711',65535,[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1]);
Ts=0.00002;
rtdef(1,'out',Ts,17);                    %rd enable
rtdef(15,'out',Ts,18);                   %wr disable

%adresa
rtdef(2,'out',Ts,11); rtdef(3,'out',Ts,12); rtdef(4,'out',Ts,13); rtdef(5,'out',Ts,14); rtdef(6,'out',Ts,15);

%zapis
rtdef(16,'out',Ts,3); rtdef(17,'out',Ts,4); rtdef(18,'out',Ts,5); rtdef(19,'out',Ts,6); rtdef(20,'out',Ts,7);
rtdef(21,'out',Ts,8); rtdef(22,'out',Ts,9); rtdef(23,'out',Ts,10);

%inicializace čítačů
 k=1; rtstart all; i=1;
rtwr(15,'y',1);rtwr(1,'y',1);

%adresa řídicí reg-1
rtwr(2,'y',1); rtwr(3,'y',1); rtwr(4,'y',0); rtwr(5,'y',0); rtwr(6,'y',0);
i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',0);

%citace rtwr(16,'y',0); rtwr(17,'y',1); rtwr(18,'y',0); rtwr(19,'y',0); rtwr(20,'y',1); rtwr(21,'y',1);
rtwr(22,'y',0); rtwr(23,'y',0);

i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',1);
i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',0);
%citac1
rtwr(16,'y',0); rtwr(17,'y',1); rtwr(18,'y',0); rtwr(19,'y',0); rtwr(20,'y',1); rtwr(21,'y',1); rtwr(22,'y',0);
rtwr(23,'y',1);

i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',1);
i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',0);

%čítač2
rtwr(16,'y',0); rtwr(17,'y',1); rtwr(18,'y',0); rtwr(19,'y',0); rtwr(20,'y',1); rtwr(21,'y',1); rtwr(22,'y',1);
rtwr(23,'y',0);

i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',1);
```

```
%adresa řídicí reg-1
rtwr(2,'y',1); rtwr(3,'y',1); rtwr(4,'y',1); rtwr(5,'y',0); rtwr(6,'y',0);
i=i+k;while (rtrd(1,'t'))<i end;
rtwr(15,'y',0);

%citac0
rtwr(16,'y',0); rtwr(17,'y',1); rtwr(18,'y',0); rtwr(19,'y',0); rtwr(20,'y',1); rtwr(21,'y',1); rtwr(22,'y',0);
rtwr(23,'y',0);

i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',1);
i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',0);

%čítač1
rtwr(16,'y',0); rtwr(17,'y',1); rtwr(18,'y',0); rtwr(19,'y',0); rtwr(20,'y',1); rtwr(21,'y',1); rtwr(22,'y',0);
rtwr(23,'y',1);
i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',1);
i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',0);

%čítač0
rtwr(16,'y',0); rtwr(17,'y',1); rtwr(18,'y',0); rtwr(19,'y',0); rtwr(20,'y',1); rtwr(21,'y',1); rtwr(22,'y',1);
rtwr(23,'y',0);
i=i+k; while (rtrd(1,'t'))<i end;
rtwr(15,'y',1);
rtstop all;
h=[0 0 0 0 0 0 0 0];
wr(1,h); wr(2,h); wr(3,h); wr(4,h); wr(5,h); h=[0 0 0 0 0 0 0 1];
wr(6,h);

rtclear all;
rtclear;                    %konec inicializace
rtunload;
```

Příloha B Výpis funkcí pro diagnostiku

Funkce určené k detekci závad na modelu a při komunikaci.

B.1 test_kanal

```
function [out]=test_kanal(in)
count=0;
n=zeros(1,500);
for i=1:500,
    n(1,i)=readcidlo(in);                    %počítá přenosy
end;
x=min(n);
for i=1:500,
   if x==n(1,i);
      count=count+1;
  end
end;

if count>100,    out='FAILED';               %selhaní přenosu, málo přenesených dat
  else    out='OK';
end;
```

B.2 test_vnej_reg

```
function [out]=test_vnej_reg
p=readcidlo(8);
k=readcidlo(9);
a=readcidlo(10);                             %kontrola určených znaků v registrech
j=readcidlo(11);
f=readcidlo(12);
if (p==80)&(k==75)&(a==38)&(j==74)&(f==70),
   out='OK';
else
   out='FAILED'                              % pokud není přesná shoda, test selhal
end
```

Příloha C Výpis procedur pro vizualizaci

Uvedeny jsou základní procedury umožňující vytvoření virtuálního světa a ovládání vizualizovaných objektů.

C.1 initvisual

```
myworld = vrworld('heli.wrl');                    %načtu svět a nastavím parametry
vrsetpref('DefaultPanelMode','off');
open(myworld);

mynode = vrnode(myworld, 'roty');                 %definuji nody pro rotace
mynode1 = vrnode(myworld, 'rotx');
mynode2 = vrnode(myworld, 'vrtulnik');

f = vrfigure(myworld);                            %určí parametry prohlížeče
set(f,'Transparency','on');
set(f,'CameraBound','off');

vrdrawnow;
```

C.2 makevisual

```
a=[0 r 0];                        %posun
a4=[0 0 1 nn(3)/60];              %naklon pricny dole
a1=[0  1 0  nn(5)/59];            %rotace kolem osy y
a2=[1 0 0 nn(6)/59];              %naklon podelny nahore
a3=[0 0 1 nn(7)/59];              %naklon pricny nahore

setfield(mynode, 'translation',a);  %nastaví rotace pro určené nody
setfield(mynode, 'rotation',a1);
setfield(mynode1, 'rotation',a2);
setfield(mynode2, 'rotation',a3);
vrdrawnow;

zoo=get(zoom,'Value');            %ovládá pohyb kamery ve světě
if (zoo~=zoo1)&(mm==1) ,
zoo3=10^(zoo/100-3);
set(f,'ZoomFactor',zoo3);
zoo1=zoo;
```

Příloha D Obsah přiloženého CD

CD obsahuje následující dokumentaci k projektu:

Eagle ..... adresář obsahuje použitá obvodová schémata, včetně programu Eagle

Fotodokumentace ..... obrázky vrtulníku a dalších periferií

Program ..... programové soubory řídicí aplikace pro prostředí Matlab

Text ..... textová podoba této diplomové práce a související práce [1]

Video ..... záznam z demonstrační ukázky letu