České vysoké učení technické v Praze - Fakulta elektrotechnická

Katedra řídicí techniky

Bakalářská práce

Jan Rusz

Praha 2006

Zadání bakalářské práce

Student: Jan Rusz

Obor: Kybernetika a měření

Název tématu: Deska vstupů/výstupů pro vývojový modul s HC12

Zásady pro vypracování:

1. Seznamte se s vývojovým modulem s mikrokontrolérem Motorola HC12D60 a vývojovými nástroji pro něj.
2. Navrhněte a realizujte desku s opticky oddělenými digitálními vstupy a opticky oddělenými silovými výstupy pro tento modul.
3. Vyzkoušejte navrženou desku na jednom z modelů v laboratoři K909 (strojovna) a naprogramujte jednoduchý ukázkový program, který bude řídit vybraný model.

Seznam odborné literatury: Dodá vedoucí práce.

Vedoucí bakalářské práce: Ing. Ondřej Špinka

Datum zadání bakalářské práce: srpen 2005

Termín odevzdání bakalářské práce: červen 2006

Prohlášení

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

V Praze dne ……. …….

podpis

Poděkování

Na tomto místě bych rád poděkoval vedoucímu práce panu Ing. Ondřeji Špinkovi za pomoc a vedení v průběhu tvorby této bakalářské práce.

V neposlední řadě patří můj dík i mé rodině, která mě po celou dobu studia vytrvale podporovala.

Abstrakt

Tato práce se zabývá návrhem řídící jednotky pro model autodráhy. Řídící jednotka je postavena na bázi Motorola HC12D60. Skládá se ze dvou částí. V první je popsán návrh, konstrukce a realizace desky s opticky oddělenými digitálními vstupy a opticky oddělenými výstupy. Ve druhé části je popis softwarového řešení řízení modulu autodráhy.

Abstract

This thesis deals with design of control unit for freeway model. Instruction control unit is based on Motorola HC12D60. It consists of two separate parts. First part focus on design, construction and testing of control for digital separated optical inputs and optical separated outputs circuit board. In the second one, a concept of software solution for controling freeway module is described.

Obsah

Seznam zkratek

CAN	(Controller Area Network) Sériová sběrnice
CD	(Compact Disc) Kompaktní disk - médium pro záznam dat.
DPS	Deska plošných spojů
EEPROM	(Electrically Erasable Programmable Read Only Memory) Typ elektronické paměti nevyžadující pro udržení obsahu dat napájecí napětí. Paměť umožňuje elektrické vymazání libovolné paměťové buňky paměti a poté zapsání dat do této buňky paměti.
FLASH	Paměť typu FLASH - typ elektronické paměti nevyžadující pro udržení obsahu dat napájecí napětí. Paměť umožňuje elektrické vymazání, přičemž vymazání probíhá vždy v rámci celého bloku dat (sektoru) najednou.
HC12	Označení jedné konkrétní rodiny 16-ti bitových mikroprocesorů Motorola.
HW	Hardware.
ISO/ANSI-C	Standard programovacího jazyka C dle norem ISO/ANSI.
OS	(Operating System) Operační systém.
OZ	(Operational Amplifier) Operační zesilovač.
PWM	Pulsně šířková modulace (Pulse Widht Modulation)
RAM	(Random Access Memory) Typ elektronické paměti používané jako pracovní paměť v mikroprocesorových systémech. Paměť umožňuje číst data z libovolné buňky paměti a zapsat data do libovolné buňky paměti. Paměť vyžaduje pro udržení obsahu dat napájecí napětí.

ROM	(Read Only Memory) Typ elektronické paměti, ve které jsou data uložena pevně, tedy data nelze měnit. Data lze z paměti pouze číst. Paměť se používá jako paměť programu v mikroprocesorových systémech.
RS-232C	Asynchronní sériové komunikační rozhraní, které je součástí skupiny rozhraní osobního počítače.
SCI	(Serial Communication Interface) Sériové asynchronní rozhraní typu point-to-point.
SPI	(Serial Peripheral Interface) Sériové synchronní rozhraní s architekturou masterslave. Rozhraní je určené k propojování více elektronických prvků v rámci jednoho elektronického zařízení (v rámci jednoho plošného spoje).
SW	Software

Úvod

Cílem této práce je vytvoření řídící jednotky pro model autodráhy a seznámení s vývojovým modulem s mikrokontrolérem Motorola HC12D60 a jeho vývojovými nástroji. Řídící jednotka je oddělena pomocí opticky oddělených digitálních vstupů a opticky oddělených silových výstupů. Pomocí této řídící jednotky naprogramovat jednoduchý výukový program pro model autodráhy v laboratoři K909.

První část textu se zabývá základním popisem vývojového modulu a jeho využitím. Popisuje základní prvky využité pro realizaci a hardwarovou strukturu.

Druhá část se zabývá stručným popisem vývojových nástrojů použitých při tvorbě této práce. Zabývá se popisem programu Vloader, využitého k nahrávání dat do mikroprocesoru, dále produktem firmy Metrowerks a jejich vývojovým prostředím CodeWarrior, které je v práci využito pro kompilaci programů v jazyce C pro procesory značky Motorola. Součástí je také popis programu Orcad, pomocí něhož byl vytvořen hardware.

Třetí část textu se pak zabývá popisem interface navržené řídící jednotky. Je zde popsán návrh, konstrukce a realizace modulu, základní funkce využitých součástek.

Poslední část textu popisuje implementaci algoritmu pro tento modul. Popisuje využitý software při tvorbě programu.

Popis použitého hardware

1.1Vývojový modul s mikrokontrolérem Motorola HC12D60

Pro tento řídící modul byl zvolen Miroslavem Musilem viz. [1] mikroprocesor Motorola HC12D60. Jedná se o procesor s 16-ti bitovou architekturou. Procesor obsahuje 20-ti bitovou aritmeticko-logickou jednotku, 60kb programové paměti FLASH, 2 kB paměti RAM, 1 kB paměti EEPROM, řadič sběrnice CAN (verze 2.0A a 2.0B), několik 8-mi bitových vstupních/výstupních portů, dva 8-mi kanálové analogově-digitální 10-ti bitové převodníky, čtyři kanály generátoru signálu pulsně-šířkové modulace (PWM), dvě asynchronní sériová rozhraní (SCI), synchronní sériové rozhraní (SPI), a další.

1.1.1 Hlavní realizační prvky vývojového modulu

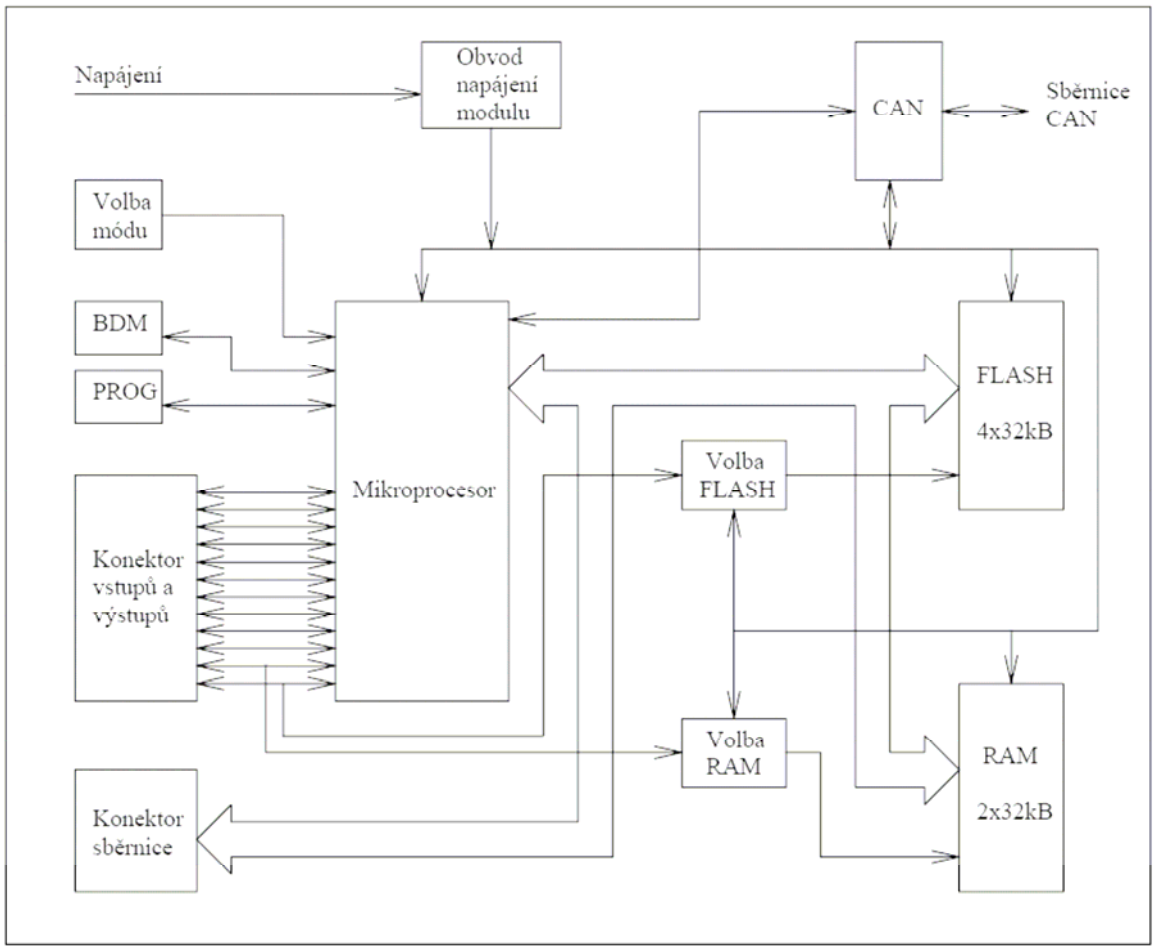
U realizace bylo třeba vyřešit problém s vícenásobným zapisováním dat do paměti, protože mikrokontrolér slouží také jako výukový celek. Paměti FLASH mají obecně omezený počet cyklů pro výmaz/zápis, čímž se rozumí vymazání paměti a následné zapsání dat do paměti. Po překročení výrobcem udaného počtu dochází k opotřebení fyzické struktury paměti. Výrobce zaručuje minimálně 100 cyklů výmaz/zápis interní programové paměti FLASH, což je nedostatečné pro použití k výukovým účelům.

Proto bylo využito externí paměti programu. Byla využita 2 Mbitová FLASH paměť AM29F200BB firmy AMD. Paměť potřebuje pouze 5V napájecí napětí a výrobce zaručuje minimálně 1 000 000 cyklů výmaz/zápis bez ztráty funkčnosti, což je dostatečné množství pro využití k výukovým účelům.

Pro propojení osobního počítače s řídícím modulem bylo zvoleno sériové rozhraní RS-232C (sériové komunikační porty osobního počítače). Vlastní umisťování programů aplikací do externí FLASH paměti programu mikroprocesoru mohou přes toto rozhraní zajišťovat dva mezi sebou komunikující programy, jeden program v řídicím modulu a jeden program v osobním počítači.

1.1.2 Hardwarová struktura řídicího modulu

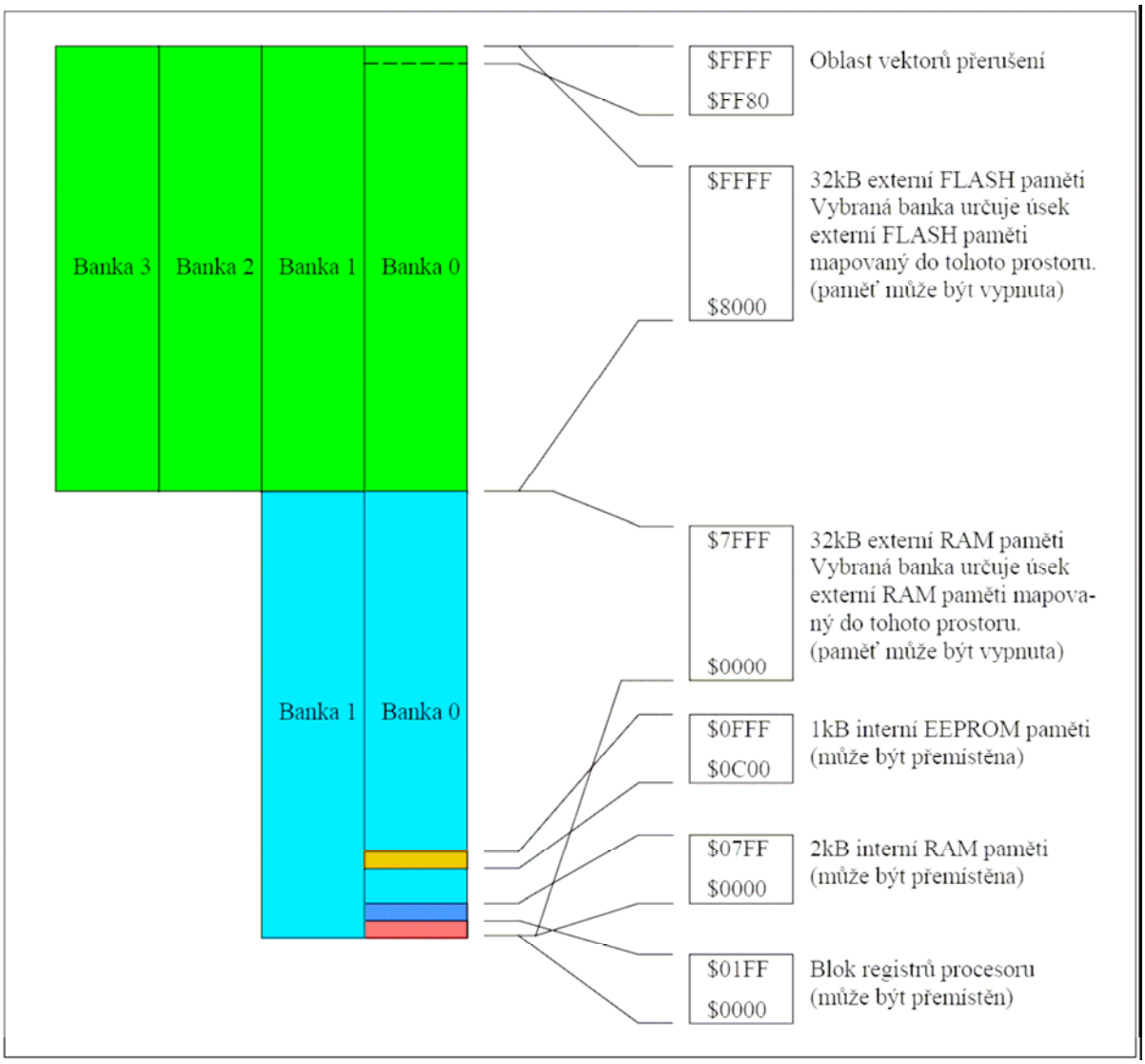
Hardwarovou strukturu řídicího modulu znázorňuje obrázek Obr. 1, převzat z diplomové práce Miroslav Musil[1]. Hlavním prvkem řídícího module je mikroprocesor. Jeho vstupy a výstupy znázorňuje blok "Konektor vstupů a výstupů". Sběrnice je taktéž vyvedena na konektor, blok "Konektor sběrnice". Externí FLASH paměť, blok "FLASH", a externí RAM paměť, blok "RAM" jsou připojeny na externí sběrnici. Blok "Volba FLASH" provádí přepínání bank externí FLASH paměti. Blok "Volba RAM" umožňuje mapování úseků externí RAM paměti do adresového prostoru mikroprocesoru. Sběrnici CAN řídícího modulu symbolizuje blok "CAN". "Volba módu" je obvod, jehož nastavení určuje mód činnosti mikroprocesoru. Blok "BDM" symbolizuje obvod programovacího/debugovacího BDM rozhraní mikroprocesoru. Blok "PROG" symbolizuje přípojné místo pro připojení programovacího adaptéru, pomocí něhož probíhá umísťování programů do paměti. Blok "Obvod napájení modulu" zajišťuje napájení řídícího modulu.



Obr. 1 – Hardwarová struktura řídícího modulu

1.1.3 Adresový prostor mikroprocesoru

Adresový prostor mikroprocesoru je zachycen na obrázku Obr. 2, převzat z diplomové práce Miroslav Musil[1]. Adresový prostor má dvě části. První část adresového prostoru má rozsah adres 0000H až 7FFFH a nalézá se v ní externí RAM paměť. Do tohoto adresového prostoru se mapuje jedna ze dvou bank (Banka 0 nebo Banka 1 externí RAM paměti). Druhá část adresového prostoru mikroprocesoru má rozsahu adres 8000H až FFFFH a nalézá se v ní externí FLASH paměť. Do tohoto adresového prostoru se mapuje jedna ze čtyř bank (Banka 0, Banka 1, Banka 2 nebo Banka 3 externí FLASH paměti). Blok interní RAM paměti mikroprocesoru, blok interní EEPROM paměti mikroprocesoru a blok registrů mikroprocesoru se nachází v adresovém prostoru.



Obr. 2 – Adresový prostor mikroprocesoru řídícího modulu

1.2 Popis modelu autodráhy

V rámci testování navrženého hardwaru a softwaru je využit model autodráhy, který se nachází v učebně strojovny K909. Model se skládá s autodráhy ve tvaru osmičky, na kterém je implementováno 6 senzorů, které snímají průjezd autíčka a se kterými můžeme měřit rychlost průjezdu, (viz. další podkapitoly). Senzory jsou implementovány pouze na jedné kolejnici.

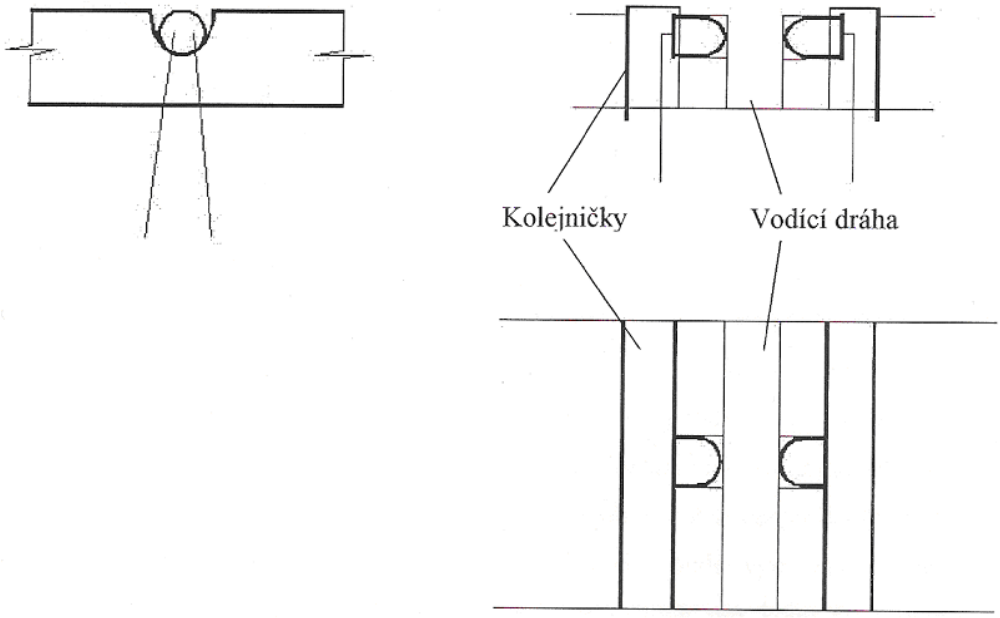
1.2.1 Připojovací modul

Tento modul byl navržen Jiřím Pejšou viz. [2], a slouží také pro komunikaci s PLC. Slouží k napájení počítačem řízené dráhy. Napájení celého připojovacího modulu je +24V a –12V. Další napěťové úrovně modelu jsou +18V, nebo –5V. Tyto úrovně jsou závislé na řídících výstupech připojeného PLC, které ovládají výstupní výkonové tranzistory. Při návrhu bylo nutné zajistit ochranu proti jejich současnému otevření, aby nemohlo dojít k jejich zničení, případně poškození autíčka. Toto řeší ochranné tranzistory, které zkratují bázi s emitorem řídícího tranzistoru druhé úrovně a na výstupu je potom stav odpojení od napájecího napětí. Tím jsou chráněny nejen tyto výstupní tranzistory, ale také napájecí zdroj. Další ochranou napájecího zdroje jsou použity stabilizátory napětí, které mají integrovanou proudovou pojistku. Pro tuto práci je modul připojen přes desku s opticky oddělenými digitálními vstupy a opticky oddělenými silovými výstupy na modul s Motorolou HC12D60. Za optickým oddělením jsou navrženy komparátory, které zajišťují potřebné napěťové úrovně pro mikroprocesor (viz kapitola 3). Rychlost autíčka je řízena přes tento modul PWM signálem, který je generován Motorolou HC12D60.

Obr. 3 – Obvody napájení

1.2.2 Snímače

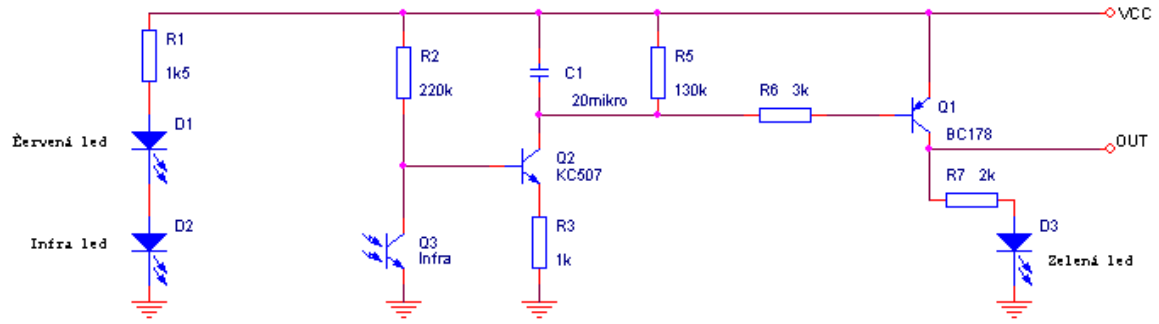
Pro tuto část řídícího systému je důležité, jaký fyzikální princip snímání je použit, vzhledem k odolnosti proti rušení. U tohoto modelu je využito optické brány. Z důvody odolnosti proti dennímu světlu je použita kombinace fototranzistoru a fotodiody pracující v oblasti infračerveného záření. Pro návrh snímačů bylo použito vložení fotoprvků přímo do vodící dráhy (viz obrázek Obr. 4, převzat z bakalářské práce Jiří Pejša[2]).



Obr. 4 – Uložení snímačů

1.2.3 Hardwarová realizace snímače

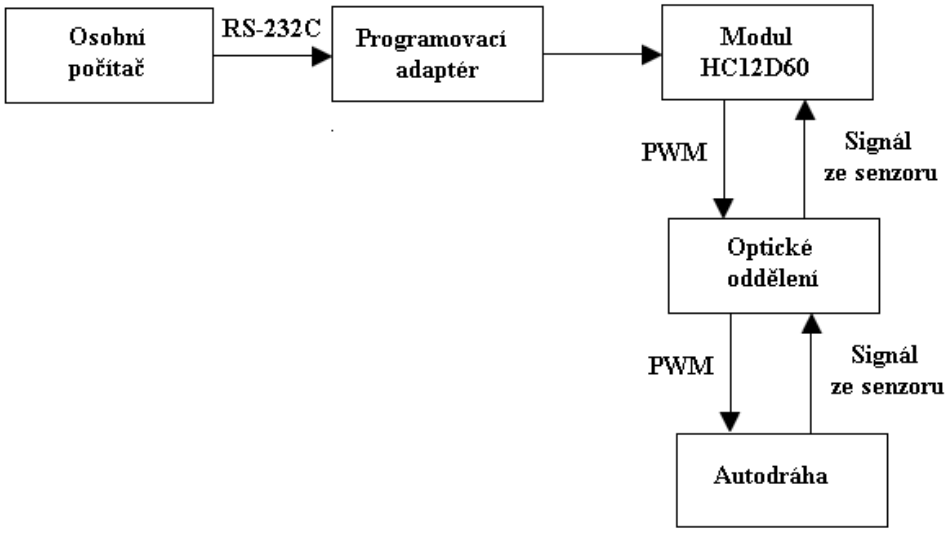
Jestliže je senzor nezastíněn, je infra tranzistor Q3 otevřen a tím pádem je tranzistor Q2 uzavřen. Kondenzátor C1 se vybijí nebo je vybit diky odporu R5 a tranzistor Q1 je uzavřen. Na výstupu je logická 0. Pokud dojde k zastínění senzoru, tak tranzistor Q3 je uzavřen a Q2 otevřen, čímž se kondenzátor C1 nabije. Tranzistor Q1 je otevřen a na výstupu je log. 1. Kondenzátor C1 zde slouží k prodloužení impulsu pro otevření tranzistoru Q1.



Obr. 5 – Vyhodnocovací obvod snímače

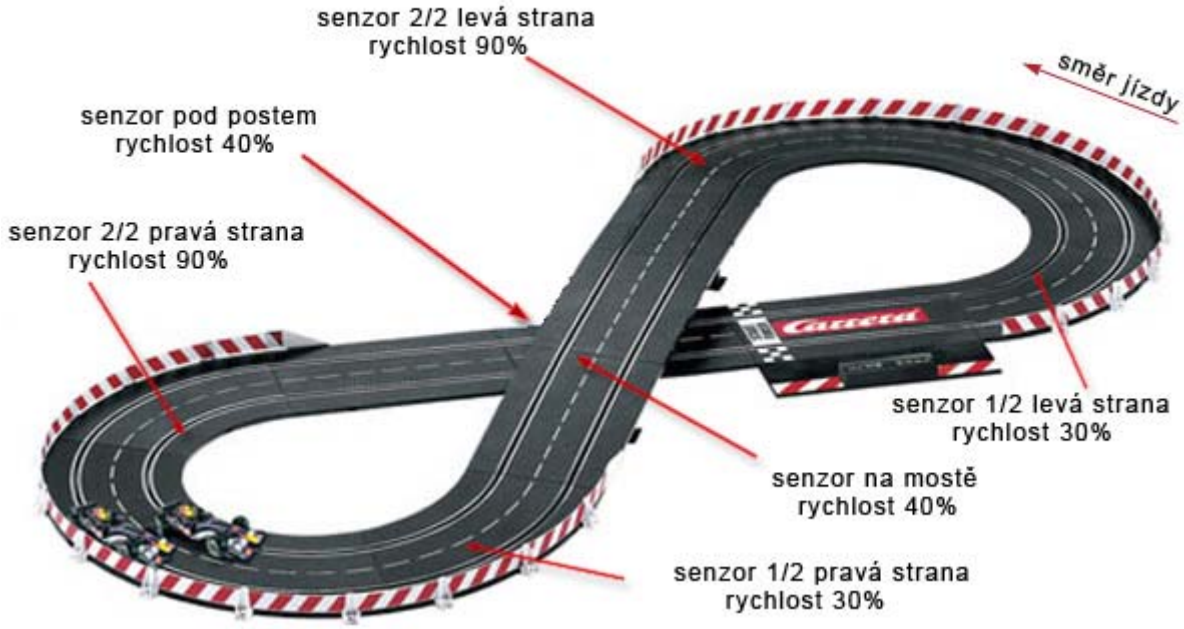
1.2.4 Strategie řízení

Mikrokontrolér s procesorem Motorola HC12D60 je připojen k počítači pomocí programového adaptéru přes sériový port. K nahrání programu do mikroprocesoru slouží program Vloader, (viz. kapitola 2.1). Pro spuštění programu je třeba přepnout odpovídající paměť FLASH, k čemuž slouží příslušný jumper obsažený na modulu. Pro komunikaci mezi modulem autodráhy a mikrokontrolérem slouží navržená deska s opticky oddělenými digitálními vstupy a opticky oddělenými silovými výstupy. Propojení je realizováno 1 výstupem a 6 vstupy. Jako výstup přichází z mikrokontroléru signál PWM, který určuje rychlost autíčka. Jako 6 vstupů slouží přerušení od 6 senzoru obsažených na modelu autodráhy. Ty jsou napojeny na port H mikroprocesoru.



Obr. 5 – Blokové schéma zapojení

Rychlost autíčka je řízena pomocí pulsu PWM. Na základě polohy určené snímači mikroprocesor určuje střídu PWM. Za 1. snímačem před nájezdem do zatáčky autíčko zpomaluje. Po projetí 2. snímačem autíčko začíná přibrždťovat aby nevykolejilo z dráhy. Po projetí 3. snímačem autíčko nabírá plnou rychlost. Pro druhou polovinu autodráhy je situace stejná jako pro část první. Míra PWM určená ve všech úsecích byla určena experimentálně. Dalším problémem bylo vypořádat se z možností špatně fungujícího či špatně reagujícího senzoru. Tato chyba může nastat při vyšších rychlostech, špatném zatížení autíčka a také díky některým nerovnostem na dráze. Tento problém je vyřešen využitím čítače. Po projetí senzorem se automaticky spouští čítač. Pokud nedojde v experimentálně nastavené době k impulsu z následujícího senzoru, čítač sám nastaví odpovídající střídu PWM.



Obr. 6– Model autodráhy

Použité vývojové nástroje

2.1 Vloader

Pro komunikaci mezi programem v osobním počítači a řídícím modulem slouží program Vloader, (viz Obr.9, převzat z diplomové práce Miroslav Musil[1]). Vloader pracuje na osobních počítačích a je určen pro práci pod operačním systémem Windows 95/98/2000/NT.

Mezi funkce programu patří možnost uložení načtených dat z Banky 0 externí FLASH paměti mikroprocesorové části řídicího modulu do souboru, zapsání dat programového kódu načtených ze souboru do Banky 0 paměti řídicího modulu, verifikace obsahu Banky 0 FLASH paměti modulu dle dat načtených ze souboru a funkce spuštění aplikace umístěné v Bance 0 FLASH paměti řídicího modulu. Program Vloader komunikuje s programem základního softwarového vybavení řídicího modulu přes sériové asynchronní rozhraní RS-232C.

Program obsahuje příkazy příkazy "Test spojení", "Programování paměti", "Načtení obsahu paměti", "Vymazat pracovní paměť" a "Spustit aplikaci".

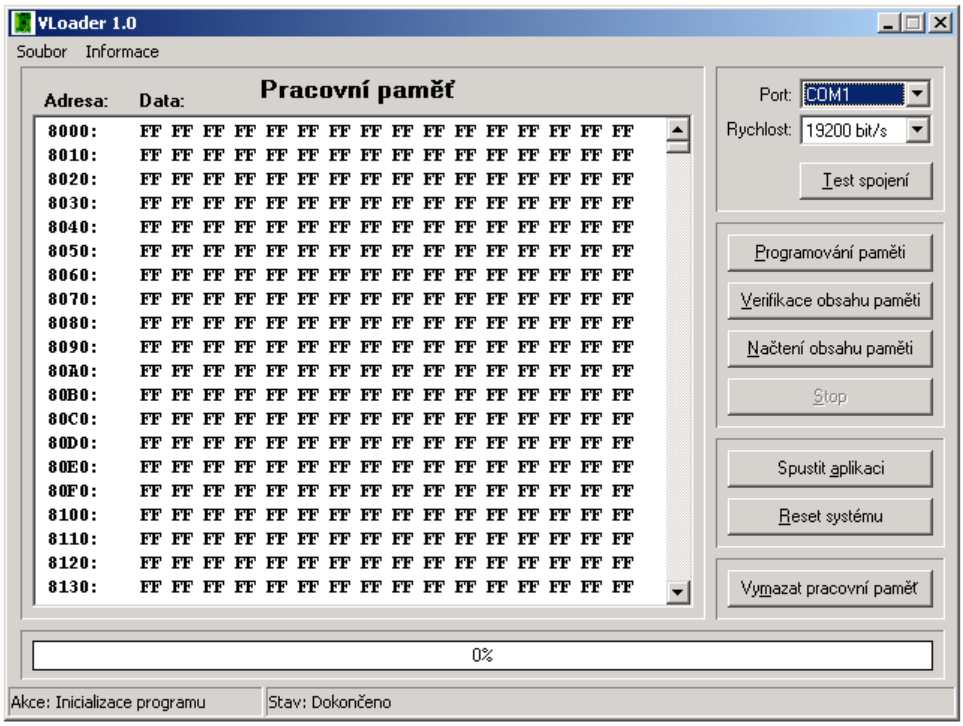
Příkaz "Test spojení" a odpověď na něj se používá pro testování spojení mezi osobním počítačem a řídicím modulem a současně lze pomocí něj zjistit verzi softwaru základního programového vybavení řídicího modulu. Doporučená rychlost pro práci přes sériové rozhraní je 19 200 bit/s.

Příkaz "Programování paměti" a odpověď na něj se používá pro naprogramování bloku dat do Banky 0 externí FLASH paměti řídicího modulu.

Příkaz " Načtení obsahu paměti " a odpověď na něj se používá pro čtení bloku dat z Banky 0 externí FLASH paměti řídicího modulu.

Příkaz " Vymazat pracovní paměť " a odpověď na něj se používá pro vymazání sektoru externí FLASH paměti řídicího modulu v rámci Banky 0 paměti.

Příkaz "Spustit aplikaci" a odpověď na něj se používá pro spuštění programu aplikace umístěného v Bance 0 externí FLASH paměti řídicího modulu. Ukončit spuštěnou aplikaci lze pouze resetováním mikroprocesorového systému řídicího modulu přes resetovací signál generovaný programovacím adaptérem.



Obr. 9– Uživatelské rozhraní programu Vloader

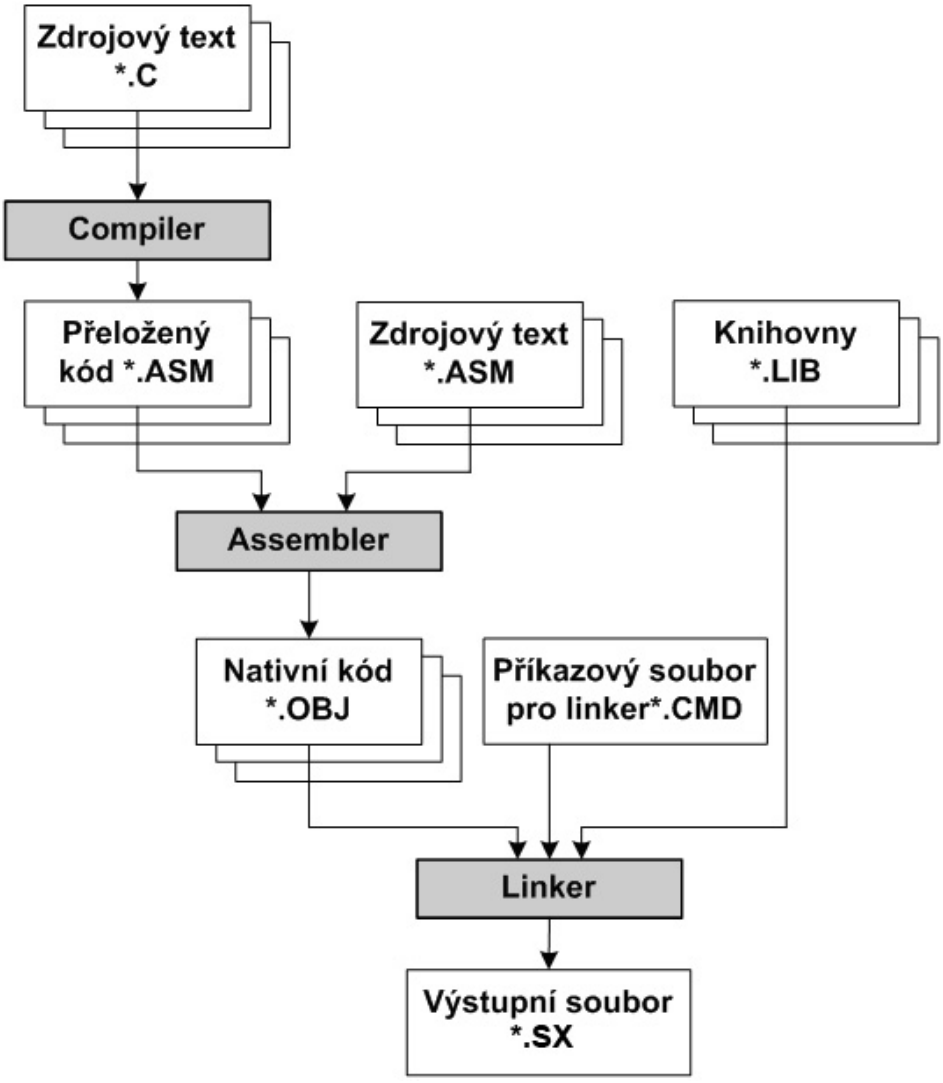
2.2 Vývojové prostředí Metrowerks CodeWarrior

Program CodeWarrior ADS 2.0 je komerční verze od firmy Metrowerks je určena k vývoji aplikací pro 8 a 16 bitové mikroprocesory Motorola řady HC08 a HC12. Pro psaní zdrojového kódu aplikačního softwaru lze použít jak asembler příslušného procesoru, tak programovacího jazyka C/C++ (norma ISO ANSI C). Prostředí CodeWarrior obsahuje nástroje pro psaní a editaci zdrojových kódů, nástroj pro nahrávání spustitelného programu do cílového systému a dále také simulátor, umožňující jednoduché ladění aplikačního software bez nutnosti nahrání do cílového systému.

Před samostatnou kompilací je třeba vytvořit nový projekt. Přesný postup pro vytvoření nového projektu je uveden v [3] v sekci Creating New Aplication. Program CodeWarrior umožňuje připojení cílového systému na bázi procesoru 68HC12 pouze prostřednictvím tzv. BDM portu (viz.[4]), což je jednovodičové připojení 16-bitových mikropočítačů řady HC12, které je určeno pro komunikaci s cílovým procesorem tj. nahrávání aplikačního kódu do FLASH paměti, odlaďování a spouštění aplikace. Nahrávání a spouštění aplikačních kódu probíhá prostřednictvím sériové linky tj., přes sériový port počítače. Po vygenerování nového projektu, je nutné přidělit adresový prostor do souboru *.prm, kde jsou

uvedeny rozsahy adres a mapování jednotlivých částí do paměti. V případě oznámení o nedostatku paměti při kompilaci, je třeba změnit velikost zásobníku v příslušné části *.prm.

Další důležitým faktorem je kompatibilita s programem Vloader. Ten při nahrávání binárních souborů do FLASH paměti pracuje pouze se soubory s koncovkou *.s19. Linker programu CodeWarrior defaultně kompiluje soubory do hexadecimálního souboru s koncovkou *.hex. Je tedy nutné změnit nastavení. To lze provést přes menu Edit/Generic Settings/Target Settings Panel/Linker for HC12/Options/Smart Linker Options Settings//Output, zaškrtnout položku Generate S-record file a potvrdit. Výstupní binární soubor bude po kompilaci uložen do složky bin umístěné v kořenovém adresáři projektu a bude mít koncovku *.sx. S touto položkou již program Vloader pracuje úspěšně.

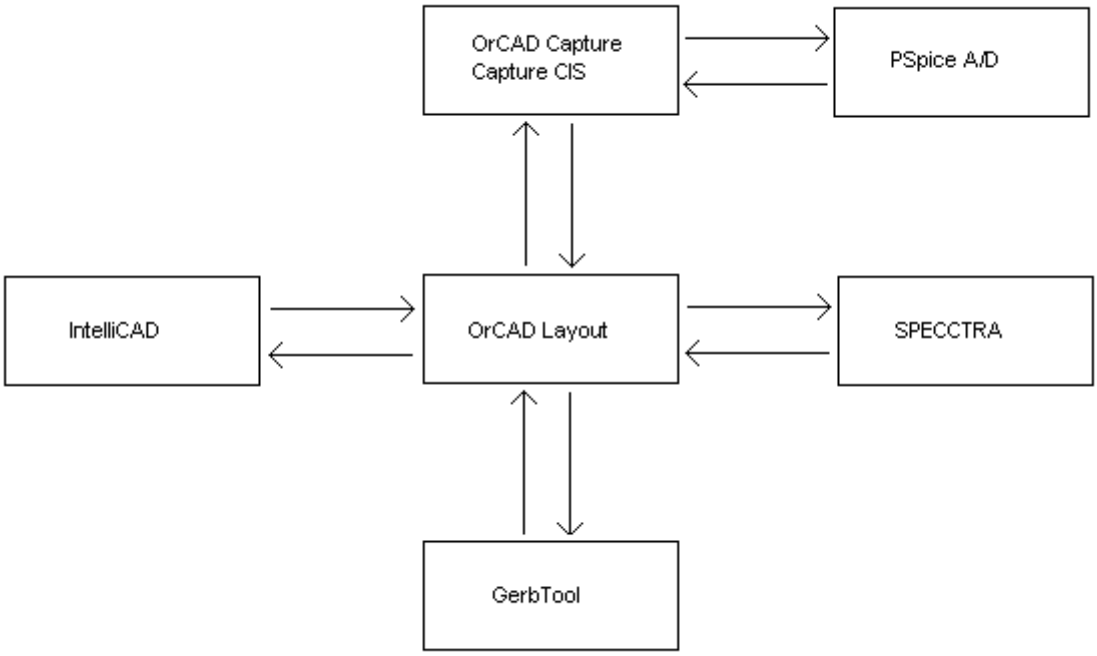


Obr. 10– Kompilace

2.3 Vývojové prostředí pro návrh a výrobu plošných spojů

Pro tuto práci byl zvolen rozšířený softwarový produkt OrCAD verze 9.2 pro systém Windows 95/98/NT/2000/XP. Orcad je určen pro nakreslení schématu a následný návrh desky, tak i pro analogově číslicovou simulaci, zpracování postprocesů, propojení k návrhovým systémům hradlových polí a programovatelných součástek, podpora k propojení na podnikové databázi atd.

OrCAD má filozofii návrhu schématu, desek plošných spojů a obvodových simulací založených na samostatných produktech. Pro návrh schématu se používá program Capture resp. Capture CIS, umožňující práci s knihovnami schematických značek, resp. databází součástek, návrhem schématu a s výstupy pro další zpracování, jako například návrh DPS, analogové nebo číslicové simulace, návrh programovatelných obvodů atd. Program poskytuje mnoho nástrojů pro další zpracování návrhu jako netlist, materiálová rozpiska, apod. Obvodové simulace se provádějí v programu PSpice, který umožňuje simulaci jak analogových tak číslicových obvodů. Přístup do Simulátoru je možný přímo z programu Capture. Pro návrh DPS je k dispozici produkt OrCAD Layout, který umožňuje vlastní návrh DPS, práci s knihovnami pouzder součástek a výstupy pro výrobu a osazování DPS. V rámci programovatelného balíku OrCAD Layout je k dispozici přímý vstup do autorouteru SPECCTRA a dále mechanický 2D editor IntelliCAD a převaděče pro import a export návrhů DPS mezi jinými návrhovými systémy. K dispozici je také množství knihoven pouzder součástek. Program plně podporuje SMD a BGA technologii. Další součástí Orcad Layout je program GerbTool pro předvýrobní zpracování dat. Vstupními soubory do programu GerbTool jsou gerberovská data pro jednotlivé vrstvy a dále data pro vrtání ve formátech přímo generovaných Layouem. Výstupními soubory jsou gerberovská data a soubory souřadnicového vrtání. Tyto data jsou nezbytná pro výrobce DPS. Komunikace mezi jednotlivými produkty probíhá v reálném čase. To například znamená, že při kliknutí na součástku nebo na spoj v Layoutu se zvýrazní součástka nebo spoj i ve schématu. Návrhový systém OrCAD nabízí na rozdíl od produktů jiných firem velmi komfortní ovládání.



Obr. 11– Princip elektronického návrhu systémem OrCAD

Popis interface modulu

Modul slouží jako univerzální optické oddělení mezi mikrokontrolérem Motorola HC12D60 a přídavným systémem (pro tuto práci s modulem autodráhy). Jeho součástí je 16 optických vstupů a 16 optických výstupů, vhodných k použití na různých modulech. Deska je složena z rezistorů, diod, LED diod, optočlenů a operačních zesilovačů.

3.1 Obecný popis funkce modulu

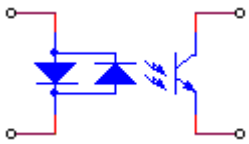
Modul je napájen napětím 5V. Připojení na napětí je ošetřeno diodou N4001 proti přepólování. Je zde taktéž použita LED dioda pro signalizaci, že je modul připojen k síti. Modul obsahuje dva 34 pinové porty, vstupní a výstupní port. Výstupní port č.1 slouží jako opticky oddělené digitální výstupy. Piny 1-16 výstupního portu je využito pro tyto výstupy, piny 17-18 slouží jako zem a piny 19-34 slouží jako výstupy portu č.2. Výstupy přicházejí přes pomocný rezistor na optočlen PC844. Za optočlenem následuje komparátor napěťových úrovní, který se stará o vyrovnání správného napětí, které vstoupí na port mikroprocesoru

(max. 5V). Za komparátorem je dále využita sada 16 diod, která slouží k signalizaci logické úrovně na portu mikroprocesoru. Vstupní port č.2 slouží jako opticky oddělené digitální vstupy. Piny 1-16 vstupního portu je využito pro tyto vstupy, piny 17-18 slouží jako zem a piny 19-34 slouží jako výstupy portu č.1. Vstupy přicházejí přes pull-up rezistor na optočlen PC844. Přes toto optické oddělení je posléze přiveden signál na výstup. Pro ochranu maximálního vstupního napětí na optočlen je využito rezistoru 5k1. Na desce je využito možnosti odpojení signalizačních diod, či spojení galvanicky oddělených zemí pro případ kontroly.

3.2 Použité části modulu

3.2.1 Optoelektronické členy

Pro optické oddělení modulu je využito obvodu PC844 osazeného čtyřmi optočleny. Optočlen je tvořený zdrojem (obvykle GaAs LED diodou) a detektorem záření (obvykle Si fotodioda nebo Si fototranzistor) vzájemně spojených optickou vazbou v jednom pouzdře. Vstup a výstup jsou proto elektricky odizolovány a podle typu pouzdra snesou izolační napětí 1,5 kV až 5 kV (= rozdíl efektivních hodnot napětí mezi libovolnou vstupní a výstupní svorkou, při kterém dochází k průrazu mezi vstupem a výstupem). Přenosovou účinnost charakterizuje proudový přenosový poměr CRT (Current Transfer Ratio), který udává poměr výstupního ku vstupnímu proudu optočlenu v procentech při daném pracovním napětí, zátěži a teplotě. S fotodiodou na výstupu je CRT = 0,2 – 0,3 %, s fotodiodou a tranzistorem je CRT = 10 – 20 % a s fototranzistorem je CRT = 10 – 100 %. Hodnoty převzaty z [10]. Rychlost optočlenu je většinou limitována detektorem na výstupu. Pro rychlý přenos číslicového signálu se proto vyrábí optočleny s fotodiodou integrovanou na jednom čipu s rychlým zesilovačem, jehož výstup je slučitelný s číslicovými obvody.



Obr. 12 – Použitý optočlen

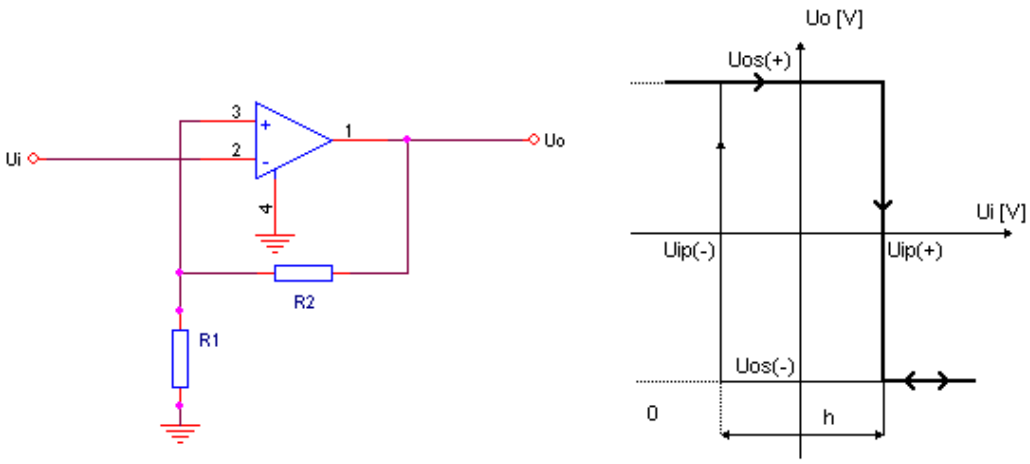
Přenos signálu mezi zdrojem a detektorem na velkou vzdálenost se provádí prostředím s malým útlumem a vysokou šumovou imunitou, kterým je optické vlákno. Jedno nebo několik takových vláken s povrchovou a mechanickou ochranou (z kevlaru a polyuretanu) tvoří optický kabel. Vlákno se skládá z vnitřního skleněného jádra s indexem lomu n_1 a je pokryto skleněným pláštěm s indexem lomu n_2. Jelikož $n_1 > n_2$, dochází pro určité rozmezí úhlu dopadu k odrazu záření na rozhraní jádro-plášť a energie záření se pak šíří převážně jádrem vlákna. Z důvodu nízkého útlumu vlákna se přenos uskutečňuje v přenosových oknech 850 nm, 1300 nm a 1550nm, přičemž s rostoucí hodnotou vlnové délky útlum vlákna klesá, ale cena potřebných optoprvků roste.

Fototranzistor

Fototranzistor má pouzdro a přechod báze kolektor upraven tak, aby za něj dopadalo záření jako na fotodiodu. Výsledný fotoproud si lze představit jako proud báze bipolárního tranzistoru, který je zesílen na přechosu báze-emitor v normálním aktivním režimu. Na rozdíl od fotodiody pak fototranzistor vykazuje zesílení v řádu desítek až několika set. Ve výstupních charakteristikách pak nalezneme jako parametr intenzitu ozáření místo proudu báze. Jednou z nevýhod fototranzistoru je závislost zesílení na intenzitě dopadajícího záření. Z důvodu velké plochy přechodu báze-kolektor je velká i jeho kapacita. Za zesílení proto zaplatíme nižší mezní frekvencí (cca. 200 kHz), která navíc klesá s klesající intenzitou záření. Proto v rychlých aplikacích nalezneme místo fototranzistoru fotodiodu zapojenou v bázi rychlého tranzistoru.

3.2.2 Komparátory

Komparátory s operačním zesilovačem se často využívají u číslicových zařízení. Pro tuto práci je použito komparátoru s kladnou zpětnou vazbou. Použitím komparátoru s kladnou zpětnou vazbou získáme obvod s výraznou nelinearitou, který je základem konstrukce velmi často používané skupiny impulsních systému s unikátními vlastnostmi.



Obr. 13 – Invertující komparátor s kladnou zpětnou vazbou

Zpětnovazební rezistory R_1 a R_2 jsou zapojeny do neinvertujícího vstupu – jedná se o kladnou zpětnou vazbu. Na výstupu je proto možný výskyt pouze kladného nebo záporného saturačního napětí. Stav obvodu je určen polaritou, amplitudou vstupního signálu a předchozí historií obvodu. Předpokládejme, že je napětí u_i záporné a platí $|u_i| >> |U_{os}|$. Protože je na neinvertujícím vstupu napětí (které je odvozeno od napětí na výstupu přes dělič R1, R2) kladnější než napětí na vstupu invertujícím, bude na výstupu kladné saturační napětí $U_{os(+)}$. Jestliže zvyšujeme vstupní napětí u_i, stav obvodu se nemění pokud vstupní napětí nepřekročí napětí prahové $U_{ip(+)}$. Toto napětí odpovídá napětí na neinvertujícím vstupu podle rovnice:

r.:1
$$U_{ip(+)} = U_{os(+)} \frac{R_1}{R_1 + R_2} + U_{os(+)}$$

Výstupní napětí nyní odpovídá zápornému saturačnímu napětí a do kladné hodnoty saturačního napětí se překlopí, pokud vstupní napětí u_i poklesne pod hodnotu určenou záporným prahovým napětím $U_{ip(-)}$:

r.:2
$$U_{ip(-)} = U_{os(-)} \frac{R_1}{R_1 + R_2} + U_{os(-)}$$

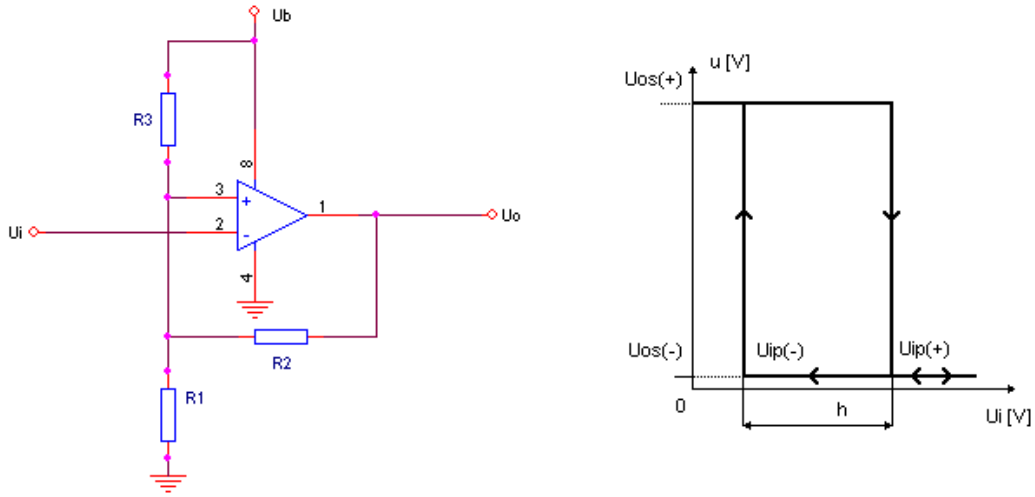
Statická charakteristika viz obr. 13 je dvojznačná. Při nulovém napětí na vstupu u_i může být na výstupu buď kladné saturační napětí nebo záporné saturační napětí. Jedná se proto o bistabilní klopný obvod, který je základem velké třídy aplikací v oboru impulzní techniky. Při popisu se často používá termín hystereze h, která je určena vztahem:

r.:3 $$h = U_{ip(+)} - U_{ip(-)}$$

Hystereze je určena děličem R_1, R_2 a velikostí obou saturačních napětí U_{os}.

V reálném modulu nám jako vstupní napětí u_i přichází výstup z optočlenu. Pro správné úrovně napětí je zde přidáno referenční napětí 5V. Zavedením referenčního napětí se nemění velikost hystereze komparátoru, ale pouze se posunuje celá statická charakteristika ve směru osy u_i ve smyslu referenčního napětí.

Komparátory v modulu jsou napájeny nesymetrickým napětím 5V (dále se používá 3,3V, případně 12V). Operační zesilovač je typu rail-to-rail. Aby obvod správně fungoval, je doplněn rezistorem R_3. Pro modul je využit obvod TLC272 obsahující dvojici OZ. Zesilovač má minimální rozdíly mezi napájecím napětím a kladným saturačním napětím. Naopak záporné saturační napětí se blíží k nule.



Obr. 14 – Komparátor použitý v modulu

Pokud dělič R_1, R_3 nahradíme zdrojem napětí v sérii s vnitřním odporem této kombinace, můžeme zapojení nahradit klasickým zapojením se symetrickým napětím. S uvažováním této úpravy dostaneme prahové napětí u_{ip}:

r.:4 $$U_{ip(+-)} = U_{os(+)} \frac{\frac{R_1 R_3}{R_1 + R_3}}{\frac{R_1 R_3}{R_1 + R_3} + R_2} + U_b \frac{R_1}{R_1 + R_3} \frac{R_2}{\frac{R_1 R_3}{R_1 + R_3} + R_2}$$

Hodnoty rezistorů $R_1 – R_3$ byly nastaveny experimentálně na R_1 = 1k Ω, R_2 = 330k Ω, R_3 = 5,6k Ω.

3.2.3 Pull-up rezistory

Další důležitou součástí modulu je použití tzv. pull-up rezistorů. Pull-up rezistor je používán v návrhu elektronických logických obvodů. Pull-up rezistory můžeme najít na vstupech logických systémů, aby se udržela definovaná úroveň napětí, v případě odpojení vnějšího zařízení. Pull-up rezistory mohou být taky využity jako rozhraní mezi dvěma typy logických zařízení pracujících na různých napětích.

V logických obvodech využívajících bipolární tranzistory pracujících na 5Vdc, je typická hodnota pull-up rezistorů 1000-5000 Ω. Hodnoty převzaty z [12]. Velikost odporu závísí na požadovaném teplotním a napěťovém rozsahu. Pro CMOS a MOS logické obvody mohou být použity mnohem vyšší hodnoty rezistorů, několik tisíc až miliónů ohmů, jelikož požadovaný proud na logických vstupech je malý.

Pull-up rezistory mohou být použity na výstupy, jako TTL logika s otevřeným kolektorem. Takové výstupy jsou použity pro řízení externích zařízení, např. pro kabely nebo pro jednoduchý způsob řízení sběrnice s více připojenými zařízeními.

Pull-up rezistory mohou být diskrétní zařízení připojené na stejné desce jako logické zařízení. Mnoho mikrokontrolerů má vnitřní programovatelné pull-up rezistory proto, aby bylo potřeba minimum externích komponent.

Některé nevýhody pull-up rezistorů jsou zvýšená spotřeba při změně vstupu a snížená rychlost v porovnání s aktivním zdrojem proudu. Některé druhy logických obvodů jsou citlivé na přechodové jevy spojené s použitím pull-up rezistorů. Tato citlivost má za následek použití filtrů pro tyto rezistory.

Popis software

Softwarová část této práce se zabývá řízením modelu autodráhy. Software je realizován v jazyce C. Ke kompilaci na mikroprocesor HC12D60 je využito programu MetroWerks CodeWarrior, (viz. kapitola 2.2). Software slouží jako jednoduchý výukový program pro seznámení práce s mikroprocesorem Motorola HC12D60.

4.1 Obecný popis funkce programu

Model autodráhy obsahuje šestici optických senzorů, které jsou umístěny tak, abychom mohli zajistit maximální rychlost průjezdu dráhou, aniž by došlo k vykolejení autíčka z dráhy. Program je navržen experimentální metodou ověřenou na modelu. Rychlost autíčka určuje střída PWM. Při spuštění programu je nastavena prvotní velikost střídy PWM na malou hodnotu, při které se může autíčko rozjet ze všech míst dráhy. Samostatný program je v cyklu a zajišťuje průjezd autíčka dráhou. Ten se spustí po úspěšném projetí jednoho ze šesti senzorů. Trasa průjezdu je optimalizována tak, aby po projetí senzorem před zatáčkou autíčko začalo brzdit a rychlost byla asi 40% maximální rychlosti. V zatáčce autíčko sníží rychlost na 30% aby nedošlo k vykolejení. Po úspěšném projetí zatáčkou autíčko zvyšuje rychlost na maximum. Další tři senzory na druhé polovině autodráhy zastávají tutéž funkci. Během jízdy může dojít ke špatnému zachycení signálu ze senzorů. Tento problém je vyřešen taktéž softwarově a to využitím timer counteru (čítače), který hlídá čas, za který by mělo dojít k přepnutí mezi dvěma sousedními senzory a tudíž ke změnění velikosti střídy PWM a rychlosti autíčka. Časové konstanty jsou opatřeny pomocnými konstantami, které zajišťují vynulování po vypršení daného časového intervalu. V případě správného zachycení signálu ze senzorů je čítač vynulován.

4.2 Popis jednotlivých částí programu

Program ke své činnosti využívá řadu knihoven. Mezi tyto knihovny patří základní knihovna jazyka C *stdlib.h* a dále knihovna *hidef.h*. Dále jsou zde 2 knihovny určeny přímo pro práci s mikroprocesorem. Jako první je zde knihovna *68HC12.h*, která definuje základní

části mikroprocesoru ke kterým patří watchdogy, vstupně výstupní porty (A, B, G, H, P, T), clock, A/D převodník, časovače, SCI, EEPROM. Druhá knihovna *pwm.h* je přímo určena pro práci s PWM. Jsou zde definovány funkce pro obsluhu práce s PWM; funkci *Initpwm(void)* pro inicializaci a funkci *pwm(unsigned int value),* pomocí které dochází v programu k vysílaní velikosti střídy PWM. V těchto knihovnách jsou nadefinovány názvy jednotlivých částí a těm jsou přiřazeny reálné adresy. Tyto definice jsou vhodné pro přehlednost programů a další práci s nima.

V hlavičce programu je nadefinována počáteční velikost střídy PWM, dále pak velikosti střídy PWM příslušné jednotlivým senzorům. Jsou zde nadefinovány také všechny proměnné a pomocné konstanty.

Mezi důležité části hlavičky patří tyto inicializace:

void initPorts() … inicializace portu H a jeho nastavení jako port vstupní

void InitPWM() … inicializace PWM

void initIO() … inicializace vstupně/výstupních portů

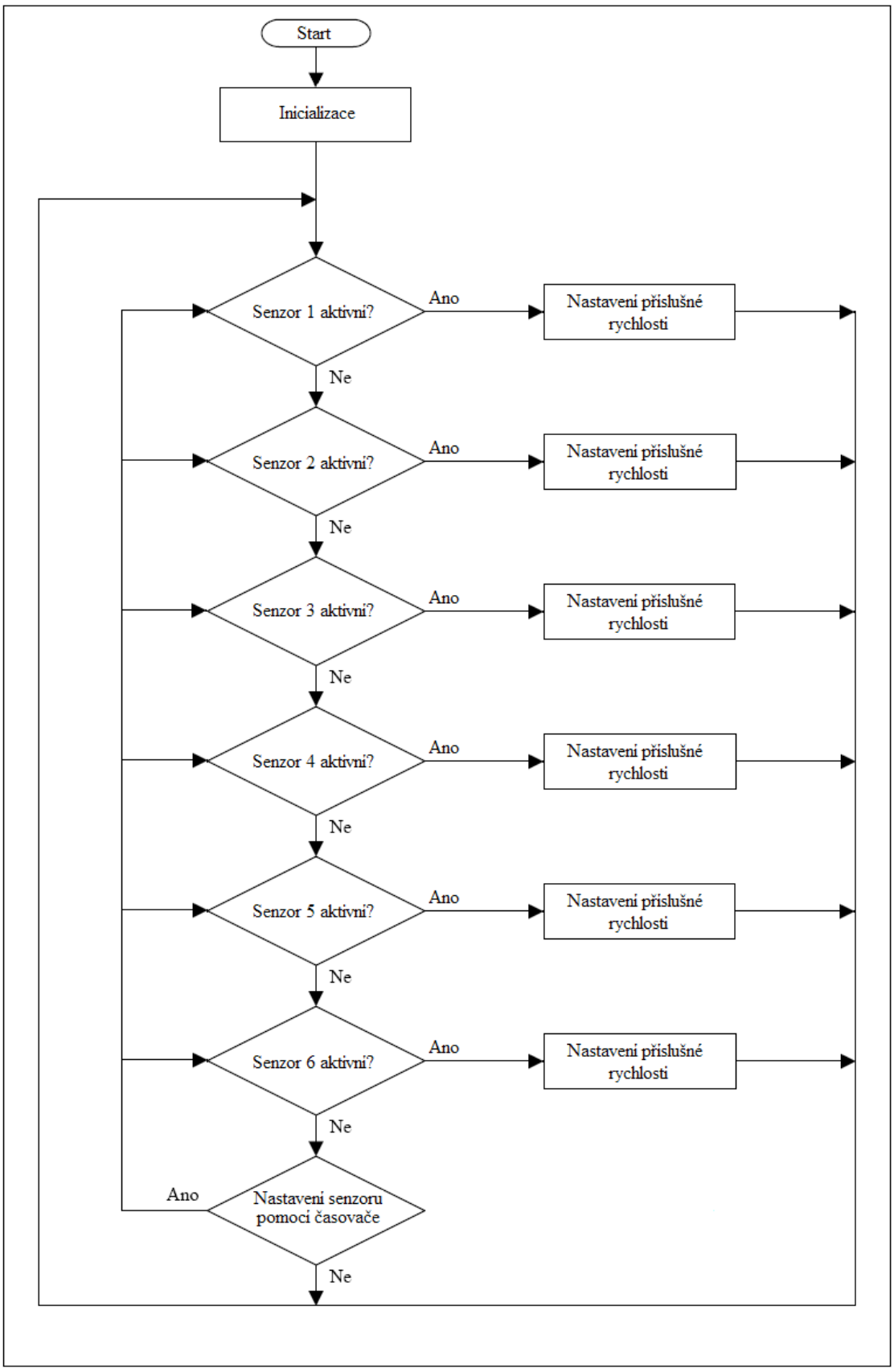
void initTimer() … inicializace čítače

Je zde také obsažena funkce pro přerušení příchozí od časovače:

void interrupt void timerHan()

Jako poslední je zde povolení přerušení *EnableInterrupts*

Hlavní část programu nám tvoří nekonečný cyklus tvořený pomocí *while*. V hlavním cyklu je soubor několika podmínek. Program běží v nekonečné smyčce a očekává změnu na jednotlivých bitech portu H. Data na portu jsou při každém cyklu zaznamenána do nezávislé proměnné a maskována s logickýma 1. Změna je zaznamenávána pomocí bitového posunu. Při změně bitu z logické 0 na logickou 1 je zaktivován příslušný bit portu a podle něho nastavena patřičná velikost střídy PWM. Je zde také využito několik pomocných proměnných, které zabraňují vzniku kolize v programu. Mezi poslední část hlavního cyklu patří využití podpůrného časovače, který nám hlídá možné chyby, které mohou vzniknout špatnou hardwarovou funkcí (nefunkčnosti senzoru, špatného zachycení). Pokud nedojde v experimentálně nastavené době ke změně signálu, časovač očekává změnu a zajistí automatické posunutí na následující senzor. Při tom opět využívá pomocných konstant, aby nedošlo ke kolizi programu. Všechny tyto funkce jsou zaznamenány ve vývojovém diagramu (viz. obrázek Obr. 15).



Obr. 15– Vývojový diagram programu pro řízení autodráhy

Závěr

Cílem této práce bylo seznámit se s mikroprocesorem Motorola HC12D60 a vývojovými nástroji pro něj, navrhnutí ukázkového softwaru pro řízení modulu autodráhy pomocí tohoto mikroprocesoru, a dále navrhnutí a realizace desky s opticky oddělenými digitálními vstupy a opticky oddělenými silovými výstupy pro tento modul, určenou pro výuku v laboratoři Katedry řídící techniky ČVUT FEL.

V rámci praktické části BP jsem navrhl a v programu CodeWarrior implementoval algoritmus pro optimalizaci řízení modulu, který pomocí zabudovaných senzorů zajišťuje co nejrychlejší průjezd autíčka po dráze. Naprogramování proběhlo pomocí sériového portu RS232.

Jako vylepšení modulu autodráhy se nabízí pevnější struktura modelu, aby nevznikaly hrboly na trati a dobré vyvážení autíčka. Dále by se na modelu mohly zabudovat senzory i na druhé části kolejnic a doprostřed dát skříženou dráhu, čímž by došlo ke složitější a zajímavější struktuře algoritmu a optimalizaci obou autíček. Pokud by se na dráhu umístil větší počet senzorů, nemusel by se využívat pomocný čítač pro kontrolu průjezdů, a řízení by bylo zcela nezávislé a přesné.

Reference

[1] Diplomová práce - Miroslav Musil, ČVUT Praha, 2003.

[2] Bakalářská práce – Jiří Pejša, ČVUT Praha, 1998.

[3] OSEK builder v. 2.2.1, user's manual, Motorola, 2001.

[4] MC68HC(9)12D60, Advanced Information, Rev. 3, Motorola, 2002.

[5] CodeWarrior user's manual,Metrowerks,2001.

[6] Záhlava, V.: OrCad 10 . Vydavatelství Grada Publishing, a.s., Praha 2004.

[7] Herout P.: Učebnice jazyka C. Nakladatelství KOPP, České Budějovice 2005.

[8] PC844, Electronic Component Specification, SHARP, 1998.

[9] TLC272, Electronic Component Specification, Texas Instruments, 1987.

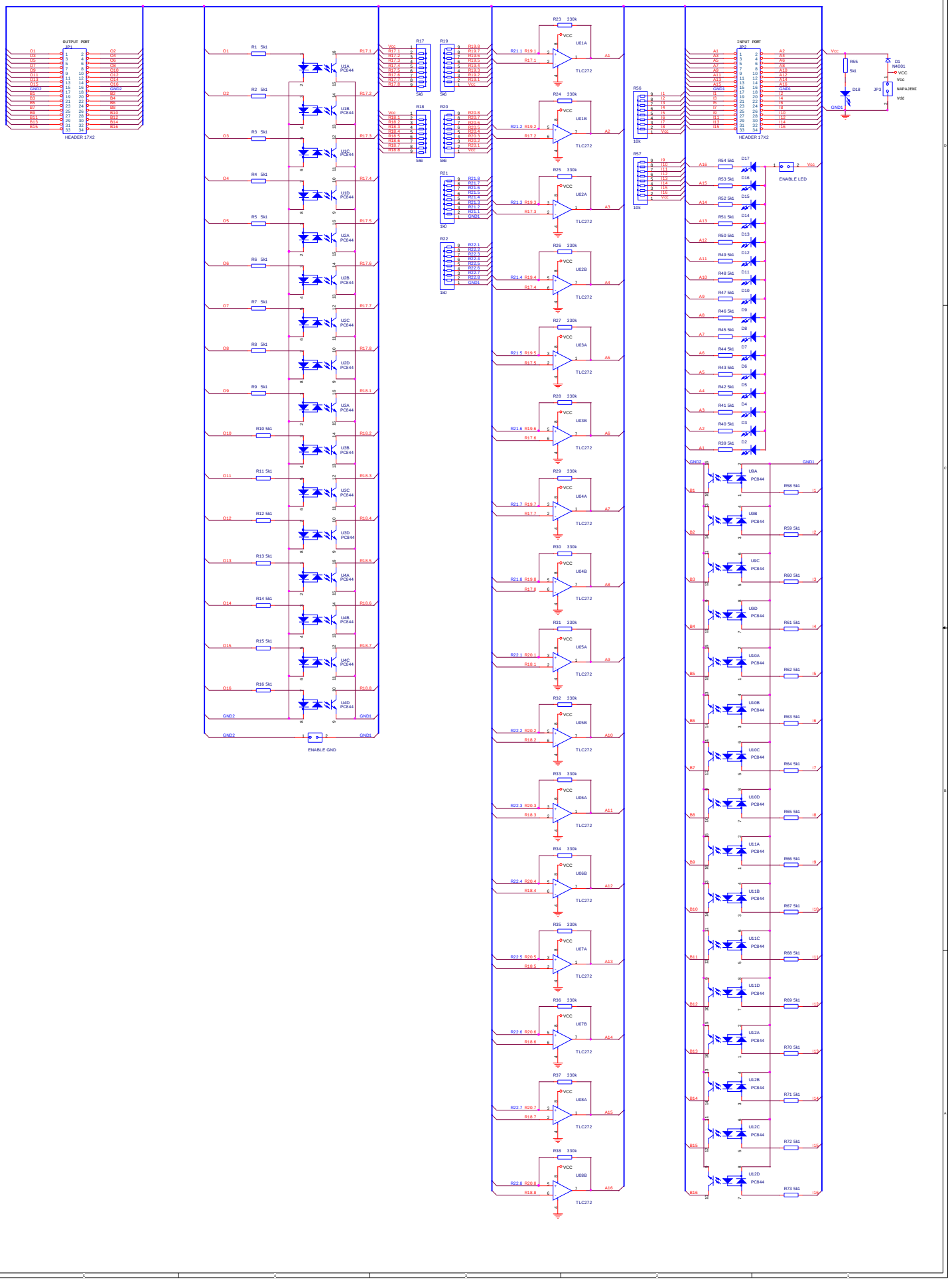
[10] Vobecký, J., Záhlava, V., Elektronika. Vydavatelství Grada Publishing, a.s., Praha 2001.

[11] Vysoký, O., Elektronické systémy II. Vydavatelství ČVUT, Praha 2003.

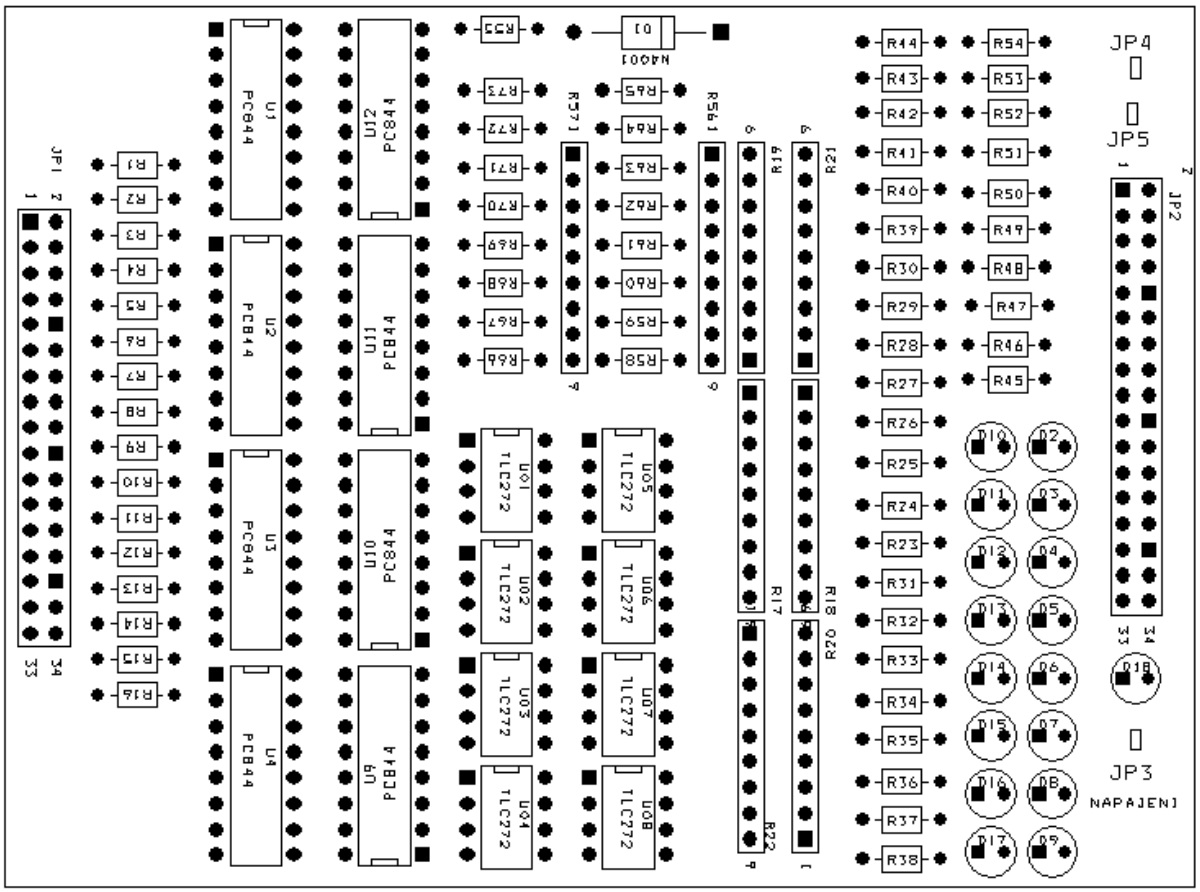
[12] http://www.wikipedia.com

Příloha A - Hardware modulu

1. Schéma zapojení řídícího modulu
2. Osazovací výkres řídicího modulu
3. Seznam elektronických součástek modulu
4. Popis konektorů autodráhy


OUTPUT PORT
JP1
HEADER 17X2
INPUT PORT
JP2
HEADER 17X2
ENABLE GND
ENABLE LED
TLC272
PC844
NAPAJENIE
Vdd
Vcc
GND1
GND2

Osazovací výkres řídicího modulu

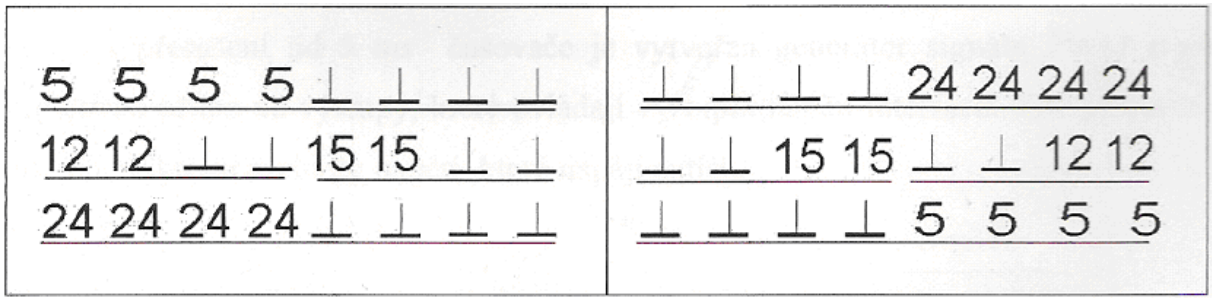


Seznam elektronických součástek modulu

Item	Quantity	Part	Reference
1	50	5k1	R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,R12, R13,R14, R15,R16,R55,R58,R59, R60,R61, R62,R63,R64,R65,R66,R67, R68,R69,R70,R71, R72,R73
2	16	330k	R23,R24,R25,R26,R27,R28,R29,R30,R31,R32, R33,R34,R35,R37,R38
3	4	8x5k6	R17,R18,R19,R20
4	2	8x1k0	R21,R22
5	2	8x10k	R56,R57
6	1	N4001	D1
7	17	LED	D2,D3,D4,D5,D6,D7,D8,D9,D10,D11, D12, D13,D14,D15,D16,D17
8	16	PC844	U1,U2,U3,U4,U9,U10,U11,U12
9	16	TLC272	U01,U02,U03,U04,U05,U06,U07,U08
10	2	HEADER_17x2	JP1,JP2
11	1	HEADER _2x1	JP3
12	2	JUMPER_2x1	JP4,JP5

Popis konektorů autodráhy

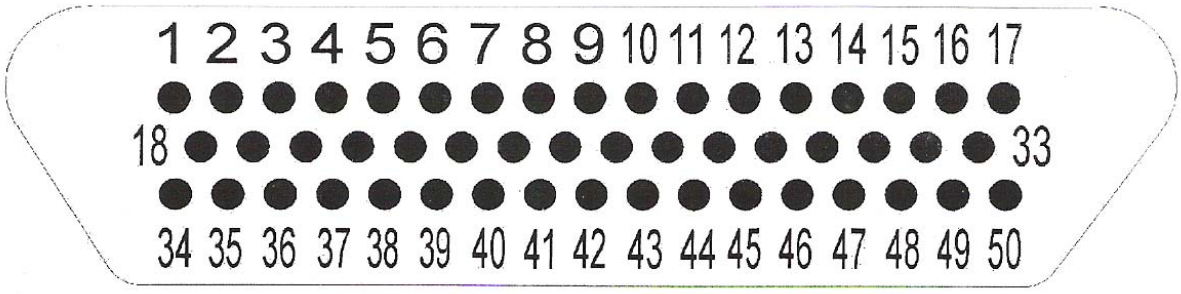
Byly použity konektory CANON 50, 50ti žilový kabel a vytvořená kabeláž připojení modulu s Motorolou HC12D60 přes sériovou linku.



Konektor napájení

Na obrázku je zapojení napájecího konektoru. Číslice označují úroveň napětí vůči příslušné zemi. Protože jde o zdroje s plovoucími zeměmi (příslušné země jsou označeny podtržením), bylo nutné jednotlivá místa s nulovým potenciálem propojit. K interface jsou přivedeny tyto úrovně dráty barvy:

Zelená: 0V
Hnědá: +24V
Žlutá: -5V



CANON 50

Příloha B - Software modulu

1. Software modulu autodráhy
2. Knihovna pro práci s PWM

Software modulu autodráhy

```
#include <hidef.h>
#include <math.h>
#include "../shared/68HC12.h"
#include "../shared/pwm.h"

#define true 1
#define false 0

#define PWM_START 4000
#define PWM_SENS_1 5000
#define PWM_SENS_2 9000
#define PWM_SENS_3 9000
#define PWM_SENS_4 6000
#define PWM_SENS_5 5000
#define PWM_SENS_6 6000

unsigned char time = 0;
int hport, time_const;
int time_up, port1, port2, port3, port4, port5, port6;


/* timer interrupt handler */
interrupt void timerHan( void ) {

   time++; // increment speed timer counter
   COPRST = 0x55;
   COPRST = 0xAA; //null watchdog
   TFLG2 = 0x80; // clear the overflow flag
}


/* timer initialization */
void initTimer( void ) {
   TSCR = 0x80; // timer enable
   TMSK2 = 0xA0; // enable timer overflow interrupt and pull-up, prescaler set to 1

}


/* IO initialization */
void initIO( void ) {
   DDRH = 0x00; // set portH as output
   PORTH = 0x00; // set portH to 0
   //COPCTL = 0x06; // watchdog enable, timeout 524ms
}
```

```
/* main function */
void main( void ) {

   InitPWM();
   initTimer();
   initIO();

   EnableInterrupts;
   pwm(PWM_START);  //set start size of PWM
   time_const = 100000; //infinite time constant(before car reached the first senzor)
   time_up = false;     //timer help constant

   while ( TRUE ) {     //infinite cycle

      if(!time_up) {      //when time isn't active, map the port with logical 1
         hport = PORTH;
         hport &= 0x03F;
      }
      time_up = false; //set off timer help constant

      /*SENZOR 1 - left 2/2 of track*/
      if ((hport >> 1) & 1) {  // 2. bit
         time = 0; // reset speed timer counter
         pwm(PWM_SENS_1);   //set on new value of PWM
         time_const = 15;   //set on time constant
         port1 = true;      //set on a bit watching constant
      }

      /*SENZOR 2 - right 2/2 of track*//*doesn't work*/
      if ((hport >> 2) & 1) {  // 3. bit
         time = 0;   // reset speed timer counter
         pwm(PWM_SENS_2);     //set on new value of PWM
         time_const = 30;     //set on time constant
         port2 = true;        //set on a bit watching constant
      }

      /*SENZOR 3 - right 1/2 of track*/
      if ((hport >> 3) & 1) { // 4. bit
         time = 0;   // reset speed timer counter
         pwm(PWM_SENS_3);     //set on new value of PWM
         time_const = 40;     //set on time constant
         port3 = true;        //set on a bit watching constant
      }

      /*SENZOR 4 - on the bridge*//*doesn't work*/
      if (hport & 1) { // 1. bit
         time = 0;  // reset speed timer counter
         pwm(PWM_SENS_4);    //set on new value of PWM
         time_const = 10;    //set on time constant
         port4 = true;       //set on a bit watching constant
```

```
}

/*SENZOR 5 - left 1/2 of track*/
if ((hport >> 4) & 1) { // 5. bit
   time = 0;   // reset speed timer counter
   pwm(PWM_SENS_5);     //set on new value of PWM
   time_const = 12;     //set on time constant
   port5 = true;        //set on a bit watching constant
}

/*SENZOR 6 - under the bridge*/
if ((hport >> 5) & 1) { // 6. bit
   time = 0;   // reset speed timer counter
   pwm(PWM_SENS_6);     //set on new value of PWM
   time_const = 30;     //set on time constant
   port6 = true;        //set on a bit watching constant
}

/*WATCHING TIMER CYCLE*/
if ( time > time_const ) { // when it is time
   time = 0; // reset speed timer counter

   if (port1) {  //if senzor 1 is active
      hport = 8; //go to 3. point
      port1 = false; //set off a bit watching constant
      time_up = true;      //set on timer help constant
   }

   if (port2) { //if senzor 2 is active
      hport = 32;   // go to 6. point
      port2 = false;   //set off a bit watching constant
      time_up = true;  //set on timer help constant
   }

   if (port3) {  //if senzor 3 is active
      hport = 1;                        //go to 1. point
      port3= false;    //set off a bit watching constant
      time_up = true;       //set on timer help constant
   }

   if (port4) {  //if senzor 4 is active
      hport = 2;     // go to 4. point
      port4= false;  //set off a bit watching constant
      time_up = true;      //set on timer help constant
   }

   if (port5) {  //if senzor 5 is active
      hport = 4;   //go to 2. point
      port5= false;     //set off a bit watching constant
      time_up = true;   //set on timer help constant
```

```
            }

            if (port6) {  //if senzor 6 is active
               hport = 16;  //go to 5.point
               port6= false;     //set off a bit watching constant
               time_up = true;   //set on timer help constant
            }
         }

      }
   }
```

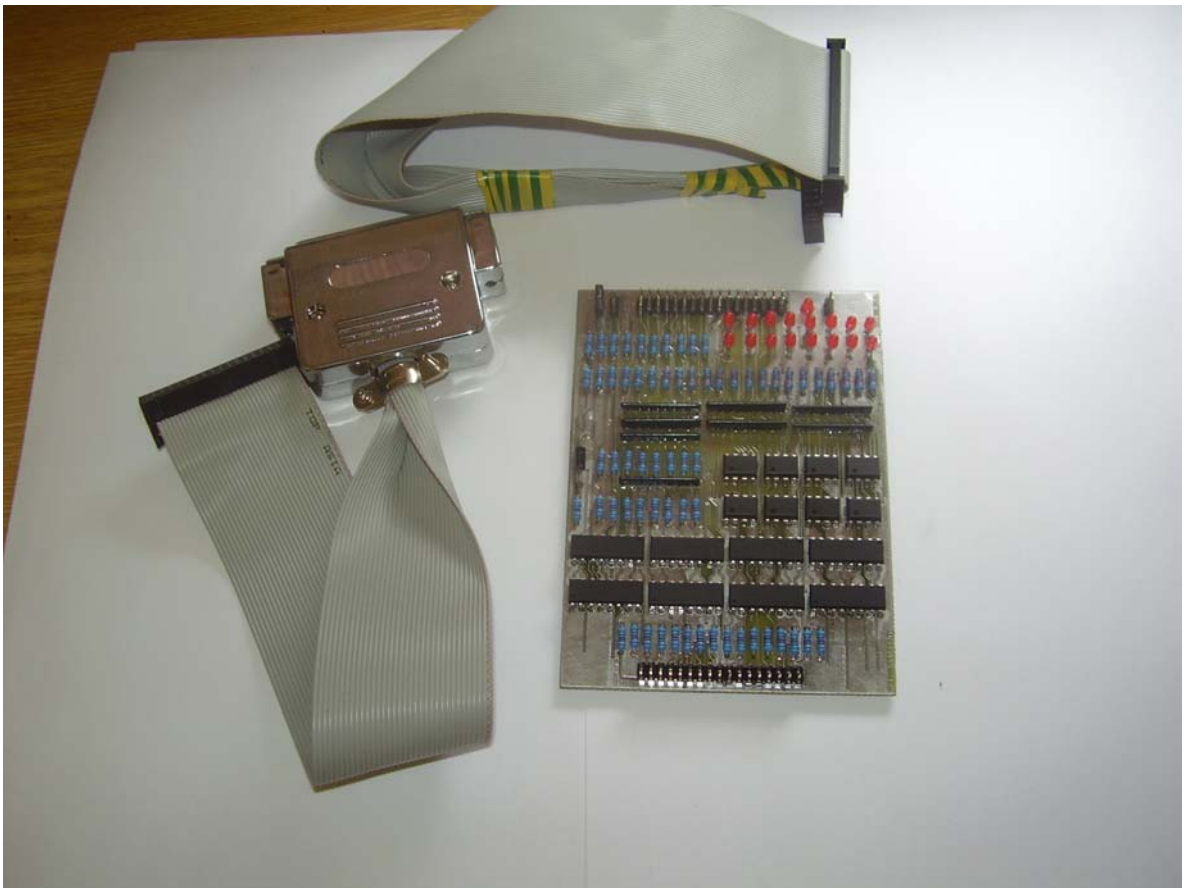
Knihovna pro práci s PWM

```
#ifndef _pwmh_
#define _pwmh_

/* PWM registers */
#define PWCLK (*((unsigned char *) 0x40))
#define PWPOL (*((unsigned char *) 0x41))
#define PWEN (*((unsigned char *) 0x42))
#define PWRES (*((unsigned char *) 0x43))
#define PWSCAL0 (*((unsigned char *) 0x44))
#define PWSCNT0 (*((unsigned char *) 0x45))
#define PWSCAL1 (*((unsigned char *) 0x46))
#define PWSCNT1 (*((unsigned char *) 0x47))
#define PWCNT0 (*((unsigned char *) 0x48))
#define PWPER0 (*((unsigned char *) 0x4C))
#define PWDTY0 (*((unsigned char *) 0x50))
#define PWDTY1 (*((unsigned char *) 0x51))
#define PWDTY (*((unsigned int *) 0x50))
#define PWCTL (*((unsigned char *) 0x54))
#define PWTST (*((unsigned char *) 0x55))
//#define PORTP (*((unsigned char *) 0x56))
//#define DDRP (*((unsigned char *) 0x57))

/* Init values */
#define PWCLK_VALUE 64 // volba periody (frekvence) signalu
#define PWPOL_VALUE 0x33
#define PWEN_VALUE 0x03
#define PWSCAL0_VALUE 0x00
#define PWPER_VALUE 155
#define PWCTL_VALUE 0x02

/* function prototypes */
int InitPWM(void);
void pwm(unsigned int value);
#endif
```

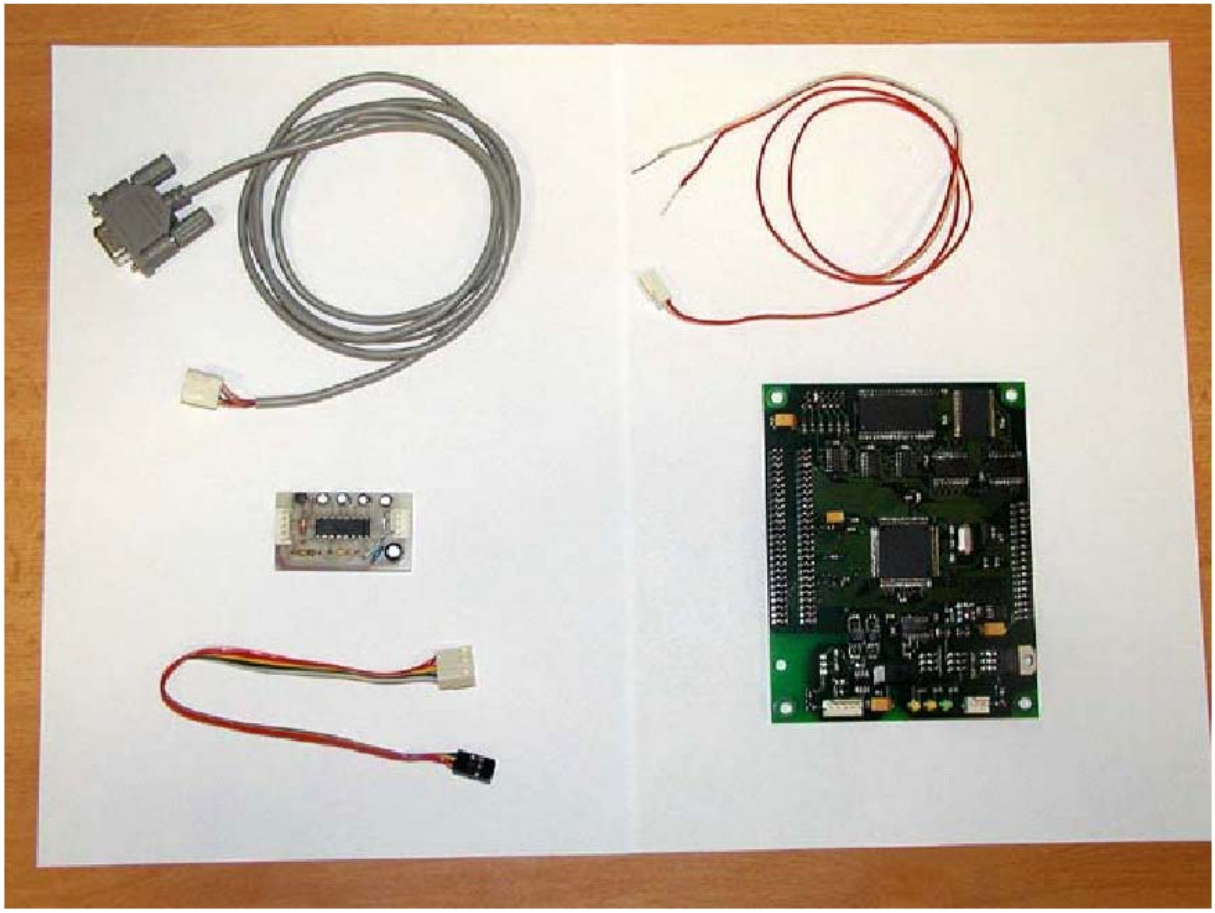
Příloha C - Obrazová část

1. Modul s kabeláží
2. Mikrokontrolér, programovací adaptér a kabeláž

Modul s kabeláží



Mikrokontrolér, programovací adaptér a kabeláž



Příloha D – Obsah přiloženého CD

Adresář	Obsah adresáře
\Modul	Obsahuje návrh plošného spoje řídicího modulu. Návrh je proveden za pomoci softwarového nástroje Orcad.
\Software	Obsahuje software pro řízení modelu autodráhy.
\PDF	Obsahuje dokumenty v PDF formátu.
\Photos	Obsahuje obrazovou dokumentaci.