VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta informačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2020 Jaroslav Jablonický

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ZAŘÍZENÍ S MODELÁŘSKÝMI SERVY
SYSTEM WITH MODEL SERVOS

BAKALÁRSKA PRÁCA
BACHELOR'S THESIS

AUTOR PRÁCE — **Jaroslav Jablonický**
AUTHOR

VEDÚCI PRÁCE — **Prof. Dr. Ing. Pavel Zemčík,**
ADVISOR

BRNO 2020

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2019/2020

Zadání bakalářské práce

Student: **Jablonický Jaroslav**
Program: Informační technologie
Název: **Zařízení s modelářskými servy**
System with Model Servos
Kategorie: Uživatelská rozhraní

Zadání:

1. Prostudujte způsob řízení modelářských serv a možnosti jejich ovládání počítačem, nastudujte relevantní literaturu.
2. Navrhněte jednoduchý strojek sestrojený z modelářských serv a způsob jeho ovládání, zaměřte se zejména na to, aby výsledek byl jednoduchý a reprodukovatelný.
3. Popište a diskutujte možnosti ovládání výše navrženého strojku modelářskými servy a diskutujte též vlastnosti vybraného řešení.
4. Navržený systém implementujte a demonstrujte jeho funkčnost na vhodném příkladě.
5. Diskutujte dosažené výsledky a možnosti pokračování práce.

Literatura:

- Dle pokynů vedoucího

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Zemčík Pavel, prof. Dr. Ing.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 28. května 2020
Datum schválení: 5. listopadu 2019

ABSTRAKT

Táto práca je zameraná na skúmanie spôsobu ovládania modelárskych serv počítačom. Tiež bude slúžiť ako prehľad alebo príručka pre začiatočníkov. Práca obsahuje návrh jednoduchého robota, presnejšie sa jedná o jednoduchú robotickú ruku, ale najmä obsahuje návrhy ovládania z rôznych platforiem za použitia rôznych prostriedkov. V závere sú odporúčania pre jednotlivé platformy.

KĽÚČOVÉ SLOVÁ

servo, Raspberry Pi, Arduino, FITKIT, PWM, robotické rameno, micro maestro

ABSTRACT

This work is focused on exploring how to control modeling servos by computers. It will also serve as an overview or a guide for beginners. The work includes the design of a simple robot, more precisely it is a simple robotic hand, but in particular it contains control designs from different platforms using different means. Finally, there are recommendations for each platform.

KEYWORDS

servo, Raspberry Pi, Arduino, FITKIT, PWM, robotic arm, micro maestro

JABLONICKÝ, Jaroslav. *Zařízení s modelářskými servy*. Brno, 2020, 59 s. Bakalárska práca. Vysoké učení technické v Brně, Fakulta informačních technologií, . Vedúci práce: Prof. Dr. Ing. Pavel Zemčík,

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Zařízení s modelářskými servy“ vypracoval samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád by som poďakoval vedúcemu práce, Prof. Dr. Ing. Pavlovi Zemčíkovi, za ochotu, tlač demo modelu a rady, ktoré my poskytoval počas vypracovávania práce. Ďalej by som chcel poďakovať pánovi Miroslavovi Martákovi za požičanie a pomoc s testami na osciloskope. Ďakujem.

Brno ....

podpis autora

Obsah

Zoznam obrázkov

Zoznam tabuliek

1 Úvod

Každú chvíľu sa nájde nejaký nadšenec, ktorý si povie, že by chcel vytvoriť autonómneho robota, vylepšiť svoj RC model, alebo si zmodernizovať život izbou, ktorá ho pozdraví, odhrnie záves, zmení intenzitu svetla, či zamkne dvere. A práve všetky tieto veci majú spoločnú potrebu vyvinúť fyzickú akciu na základe impulzu, či už z ovládača alebo senzoru. Týmto fyzickým činiteľom najčastejšie býva nejaký motorček, alebo priamo modelárske servo zvládajúce viac než sa len točiť dokola, ale aj nastaviť presnú polohu svojho vychýlenia. Nasledovne človek začne hľadať spôsob a možnosti ich ovládania. Existuje veľa roztrúsených článkov dokonca aj video príručiek pre ovládanie, ale vždy predpokladajú určité porozumenie látke, alebo sa jedná o platformu, ktorá mu nie je po chuti alebo ju nevlastní. A práve táto práca sa pokúša tieto články zjednotiť a poskytnúť záujemcovi základy pre tvorbu.

Čiže v práci sa budem zaoberať skúmaním rôznych riešení z viacerých platforiem, ich testovaním a vyhodnotením, ich splnenie schopnosti ovládania modelárskych serv. Medzi skúmané platformy zahrniem hlavne platformu o ktorej nie je veľa článkov a to je personálny počítač. Nasledovne zahŕňam aj klasické populárne platformy Raspberry Pi a Arduino, aj keď platformu Arduino len okrajovo bez testov. Ako poslednú platformu zahrniem FITKIT, túto platformu zahŕňam hlavne ako študijnú pomôcku na VUT FIT a jej možné použitie v cvičeniach. Obsahom cvičení by mohla byť jednoduchá generácia signálu, doplnením kódu buď z MCU alebo z FPGA. Všetky tieto skúmané riešenia by mali byť schopné generovať signál na ovládanie modelárskych serv i keď nemusia byť použiteľné pre všetky aplikácie.

Ďalej pre niektoré tieto skúmané a testované riešenia vyrábam demo kód a pripojím ich na demo aplikáciu. Táto aplikácia bude robotické rameno z modelárskych serv.

V kapitole 2 je krátke zhrnutie teórie k servám, ich parametre a ovládanie ale i pojmov ako pulzná modulácia signálu, nasledovanou popisom jednotlivých platforiem. Každý popis platformy bude obsahovať všeobecné informácie o platforme jej vstupno-výstupné rozhranie a tiež popis existujúcich riešení. Kapitola 4 obsahuje podobné rozdelenie ako v kapitole 3 a to platforma mnou vybrané riešenia, výsledky ich testov. Ďalej bude obsahovať popis demo aplikácie jej vývin a konštrukciu. Na záver bude práca ešte obsahovať rýchly prehľad riešení a základných vlastností.

2 Serva a ich riadenie počítačovými systémami

Táto kapitola zahŕňa popis súčasného stavu techniky serv a ich riadenie pomocou počítačov. Práca nie je encyklopédiou, ale poskytuje len základný popis a informácie pre pochopenie obsahu.

2.1 Modelárske Serva

Známe tiež pod názvom servo, sú motory u ktorých sa dá nastaviť presná poloha natočenia osi. Pre moje účeli používame modelárske servá pre rádiom ovládané modely alebo robotiku. Klasický príklad použitia modelárskych serv je ich pripojenie k rádiovému prímaču, od ktorého dostáva riadiaci signál pre určenie natočenia hriadele.[29]

Typicky sa servá skladajú z elektromotoru na jednosmerný prúd, ozubených kolies reprezentujúcich prevodovku a z riadiacej elektroniky, ktorej základom je mikrokontrolér. Prevodovka plní rovnakú funkciu ako aj v aute, to je nastavenia sily a rýchlosti. Jej ďalšou funkciou je točenie spetnoväzobného potenciometru na základe uhlu nastaveného motorom. Riadiaca elektronika spracováva vstupný riadiaci signál, z ktorého vyhodnotí požadované natočenie s odporom spetnoväzobného potenciometru a potočí motor požadovaným smerom. Riadiaca elektronika sa dá detailnejšie rozložiť na monostabilný preklápací obvod, potenciometer, sčítačku a mostíkový spínač. Princíp spracovania vstupného signálu je nasledovný: impulz zopne preklápací obvod, ktorý na základe súčasného stavu indikovaného potenciometrom generuje impulz s obrátenou polaritou oproti vstupnému. Oba tieto impulzy idú do sčítačky, z ktorej ide výsledný impulz do mostíkového spínača pre zosilnenie, aby bol schopní roztočiť motorček a tým cez prevodovku pohnúť páčkou serva a súčasne potenciometrom pre spätnú väzbu predstavujúcu nový aktuálny stav. Tento proces neskončí lebo potenciometer znovu zopne preklápací obvod. Výsledkom je znovu pulz z obrátenou polaritou k vstupnému ale s dĺžkou bližšou výstupnému signálu. Ukončovacou podmienkou na zastavenie motorku je vyrovnanie dĺžok porovnávaných impulzov.[25][31]

Digitálne servá síce používajú digitálnu riadiacu elektroniku, ale vstupný riadiaci signál majú rovnaký ako analógové, čím je zabezpečená kompatibilita medzi nimi. Všeobecne sú tieto servá schopné spracovať aj vyššiu frekvenciu signálu, sú presnejšie a silnejšie s čím je ale často spojená aj vyššia spotreba oproti analógovým. Rozdielne

[1] source:`https://upload.wikimedia.org/wikipedia/commons/thumb/e/ec/Exploded_Servo.jpg/1280px-Exploded_Servo.jpg`

Obr. 2.1: Rozobrané servo[1]

správanie prejavujú v čase keď už nedostávajú žiadni nový signál, digitálne pokračujú bez zmeny a analógové sa "vypnú".

Delenie serv

Delenie serv aj keď je, tak nie je štandardizované takže sa od neho niektorí výrobcovia jemne odkláňajú.[19][23]

1. **Podľa typu riadiacej elektroniky**
   - **Analógové**
   - **Digitálne**
2. **Podľa veľkosti**– Servá sa štandarde delia do šiestich kategórií popísaných v tabuľke 2.1 nižšie. Delenie okrem veľkosti zahŕňa aj váhu serva. Kategórie sa môžu líšiť v závislosti od výrobcu, napríklad zjednotením viacerých kategórií do jednej.
3. **Podľa spôsobu pohybu hriadele**
   - **pozičné** – Jedná sa o klasické najpredávanejšie servo, kde natočenie zodpovedá určitému stupňu z rozsahu, stupeň určuje riadiaci signál.
   - **kontinuálne** – Riadiaci signál určuje rýchlosť točenia, či už jedným alebo druhým smerom, typickým využitím je pohon modelu.
   - **lineárne** – Obsahuje zmenenú prevodovku, aby pohyb osi tvoril priamku a nie časť kružnice, najčastejším použitím je vysúvanie podvozkov mode-

lov lietadiel.

4. **Podľa materiálu prevodovky**
   - **Plastové** – Všetky ozubené kolieska sú plastové, výhodou je najnižšia váha a cena.
   - **Kovové** – Prevody sú kovové, tieto servá nesú označenie MG (Metal Gears), výhodou je ich odolnosť a pevnosť, sú drahšie a obvykle mávajú vyšší výkon, lebo plastové sa pri vyššom výkone opotrebujú rýchlejšie.
   - **Hybridné** – Kombinácia predošlích dvoch kategórií, označujú sa HG (hybrid Gears).

Velkosť	**Váha** [g]	**Šýrka** [mm]	**Výška** [mm]	**Použitie**
Nano	<8	7,5	18,5	Mikro lietadlá, Mikro helikoptéry
Sub-Micro	8 – 16	11.5	24	Lietadlá s rozpätím 1400mm, 200-450 helikoptéry
Micro	17 – 26	13	29	až do 2000mm rozpätie krídiel, 500 helikoptéry
Mini	27 – 39	17	32.5	600 helikoptéry
Standard	40 – 79	20	38	>2000mm rozpätie aj turbínové lietadlá, 700-800 helikoptéry
Large	>=80	>20mm	>38mm	Lietadlá gigantickej veľkosti a tryskové lietadlá

Tab. 2.1: Rozdelenie serv podľa veľkosti[20]

Odklonenia sú predovšetkým spojené s veľkosťou serva, určitý výrobcovia spájajú niektoré kategórie do seba.

2.2 Zapojenie a ovládanie RC serv

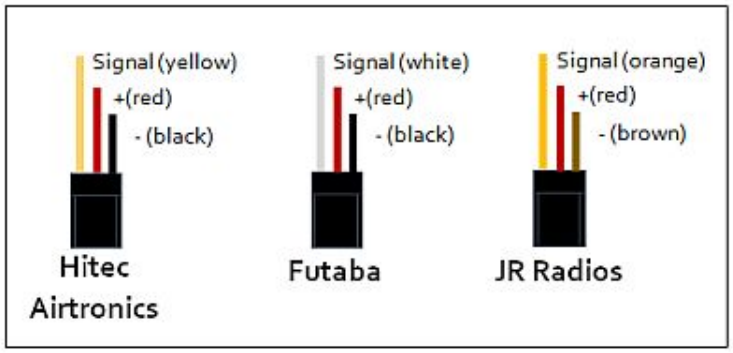
Pri zapojení sa môžeme stretnúť s tromi farebnými schémami.[28]

Posun je určený na základe šírky vstupného obdĺžnikového signálu o frekvencii 50Hz, t.j. nástupná hrana každých 20ms. Riadiaci signál je popísaný pod sekciou PWM nižšie. Typický rozsah pohybu modelárskych serv je 0-180° niektoré servá aj keď majú rozsah pohybu 180°, tak za použitia len bezpečného signálu dosahujú pohybu 120°, dalo by sa tiež udávať pohyb od kľudovej/neutrálnej polohy serva,

[2]source:`https://www.swanrobotics.com/wp-content/uploads/2015/03/servoconnect.jpg`

Farebná schéma		
Zem	Napájanie	Signál
Hnedá	Červená	Oranžová
Čierna	Červená	Žltá
Čierna	Červená	Biela

Tab. 2.2: Tabuľka farebných schém káblov serv



Obr. 2.2: Zapojenie serva[2]

ktorej zodpovedá 1,5ms dlhý pulz, pre mnou použité servá to robí -60° až 60°, pre 1 až 2 ms dlhé pulzy.[27][33]

Odporúča sa používať dĺžky pulzov z rozsahu 1-2ms z dôvodu fyzického dorazu na servách. Pri dosiahnutí tohto dorazu sa môže stať jedna z nasledujúcich vecí. Za prvé sa zvýši odoberaní prúd a hrozí zhorenie riadiacej elektroniky. Za druhé ak sú použité lacnejšie servá s plastovými prevodmi, tak hrozí zlámanie zubov a s tým spojenou stratou presnosti natočenia.[25][6]

Parametre serv

Medzi najčastejšie uvádzané parametre patria rozmery serva, jeho hmotnosť, uhol a typ prevodovky. Ostatné parametre väčšina predajcou neuvádza.[23]

- **Rozmery** – Fyzické rozmery serva, bližší popis v kapitole s delením serv vyššie.
- **Hmotnosť** – Váha serva, súvisí s veľkosťou, materiálom prevodov, či typom motoru
- **Ťah** – Udáva váhu ktorú je servo schopné potiahnuť pri určitej dĺžke ramena. Najčastejšie sa udáva kilogram na centimeter. To značí počet kilogramov ktoré posunie keď má servo rameno o dĺžke 1cm.
- **Rýchlosť** – Udáva rýchlosť vychýlenia ramena. Udávané je to v sekundách

na stupne. Referenčné stupne sú 45 a 60 stupňov. Napríklad mnou použité servo GO-13MG má rýchlosť 0,14s/60°, čiže za 0,14s prejde 60°.

- **Typ prevodovky** – reprezentuje očakávanú odolnosť voči poškodeniu, popis v kapitole s delením serv vyššie.
- **Napájanie**
- **Uhol** – Operačný rozsah o ktorý je možné pohnúť hriadeľov, udávaný v stupňoch. Bežne sa jedná o 90 a 180° uhol.
- **Impulz pre strednú polohu** – Určuje dĺžku pulzu, ktorým uvedieme servo do neutrálnej polohy. Najčastejšie je to pulz o dĺžke 1,5ms, niekedy sa ešte udávajú aj pulzy pre krajné polohy t.j. 1-2ms alebo 1,25-1,75ms podľa operačného rozsahu.
- **Rozlíšenie** – Definuje presnosť umiestnenia hriadele po prijatí riadiaceho signálu. Typicky udávané v stupňoch. Bežné servá majú rozlíšenie v rozsahu 1 až 10 stupňov.

Ako zaujímavosť by som zmienil, že kovové prevodovky serv majú podtyp určený pre modely lietadiel najmä kvôli odolnosti a malej hmotnosti, týmto typom sú titánové prevodovky so skratkou TG.

2.3 Pulz s moduláciou PWM

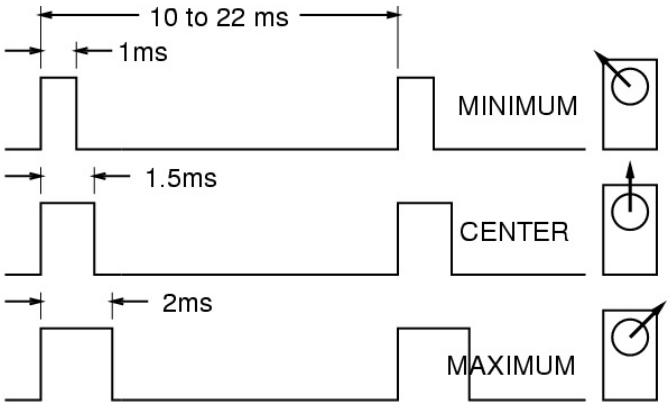
Moduláciou sa všeobecne rozumie proces ovplyvňovania nosného signálu. Nosný signál je ovplyvnený modulačným signálom. Typicky sa moduluje jedna z trojice charakteristík signálu. Tieto charakteristiky sú fáza, amplitúda a frekvencia.

PWM alebo Pulse width modulation označenie metódy úpravy periodického signálu zmenením šírky impulzu v závislosti od vstupných veličín na vstupe za účelom prenosu informácie, či regulácie elektrického výkonu s vysokou efektivitou. Táto efektivita je daná tým, že regulátor je v ideálnych podmienkach vždy buď úplne otvorený alebo úplne uzavretý. Z tohto dôvodu v ňom nevznikajú tepelné straty na odporovom prvku spôsobené úbytkom napätia. Pre pochopenie základného princípu je treba si definovať pár pojmov. Minimálnym a maximálnym napätím rozumieme hodnotu napätia v lokálnom extréme. Amplitúda reprezentuje rozdiel maximálneho a minimálneho napätia. Cyklus je interval vlny v ktorom nájdeme jednu opakujúcu sa časť signálu. Perióda je čas za ktorý prebehne jeden opakujúci sa cyklus. Frekvencia je obrátená hodnota periódy. Pracovná doba reprezentuje dobu po ktorú je signál aktívni alebo ľudovo hore. Keď si nastavíme maximálne napätie na 5V s minimom v 0V s pracovnou dobou na 50% tak po pripojení voltmetra získame 2,5 V. Asi najjednoduchšie časté využitie tejto vlastnosti je s pripojením LED, či už malého počtu alebo celého pásu a ovládania intenzity, či ľudovo ako veľmi svietia.[5]

PWM na riadenie serv

Pre väčšinu ľudí sú pojmy ako PWM frekvencia a šírka impulzu niečo čomu nemusia plne rozumieť, aby mohli upravovať RC modely ako hoby, lebo tieto pojmi sú zabalené v rôznych predpripravených prípravkoch.

Pulzne šírkovo modulovaný signál je prenášaný cez signálový vodič (oranžový, žltý alebo biely záleží od výrobcu), tento pulz zodpovedá za nastavenie správnej polohy serva. Použití riadiaci PWM signál má frekvenciu 50Hz, čo zodpovedá 20ms perióde, pulzy sa môžu líšiť dĺžkou/šírkou pulzu, tieto šírky sú štandardizované čiže všetky modely pracujúce s frekvenciou 50Hz by mali vychýliť svoje rameno do rovnakého uhlu. Veľmi často sa v špecifikáciách objavuje číslo 1,5ms (1500-1520µs) špecifikujúce šírku impulzu na vycentrovanie či nastavenie serva do neutrálnej polohy, ktorá sa nachádza v presnom strede medzi krajnými bodmi. Tento údaj sa špecifikuje z dôvodu, že niektoré serva môžu využívať kratší impulz na centrovanie. Skracovaním a predlžovaním impulzu je možné pohybovať ramenom do oboch strán v rámci operačného rozsahu. Pre dosiahnutie hraničných polôh sa pulz môže zmeniť až o 1ms. Ako som už spomenul skôr, tak nútením serva za tieto hraničné polohy môže spôsobiť jeho vyhorenie, či zničenie jeho prevodov, s dôvodov mechanickej zarážky alebo prekročenia limitov elektroniky. [25][6]



Obr. 2.3: Ovládací pulz serv[3]

Toto platí pre klasické servá a s určitou abstrakciou aj na lineárne servá, kde to nie je pohyb po kružnici ale pohyb po priamke. Výnimkou sú kontinuálne serva u ktorých sa význam riadiacich impulzov trochu líši. Pretože nenastavujú presnú polohu, ale sú schopné sa neustále točiť dokola, tak neutrálna poloha zodpovedá zastavenému motoru. Pri menení pulzu môžeme servo roztočiť na jednu, či na druhú stranu podľa toho ku ktorému hraničnému bodu sa blížime a zároveň čím sme tomuto bodu bližšie tým má servo vyššiu rýchlosť otáčania daným smerom.

[3]source:`http://wearcam.org/ece385/lecturelab6/servo_pwm_pulses.gif`

3 Prehľad existujúcich počítačových riešení riadenia serv

Táto kapitola predstavuje zjednodušení popis platforiem a existujúcich riešení na daných platformách.

3.1 Platforma FITKIT

[11] Jedná sa o učebnú pomôcku založenú na Spartan 3,používanú pre HW predmety na FIT VUT Brno. FITkit je samostatný hardvér, ktorý obsahuje výkonný mikrokontrolér s nízkym príkonom, hradlové pole FPGA (field programmable gate array), a mnoho periférií. Generovanie programovacích reťazcov pre FPGA v jazyku VHLD prebieha úplne automaticky s pomocou profesionálnych návrhových systémov. Softvér pre mikrokontrolér je vytváraní v jazyku C a do spustiteľnej formy sa prekladá pomocou GNU (GCC), ktorý je možné používať zadarmo. Všetky potrebné softvéry pre návrh sú tiež zdarma.

Obr. 3.1: Platforma FITKIT[1]

Základné informácie o kite z oficiálnych stránok:

- MCU rodiny MSP430 (Texas Instruments)
- FPGA Spartan 3 XC3S50-4PQ208C nebo XC3S400-4PQ208C (Xilinx)
- USB prevodník FT2232C
- audio rozhranie
- konektory PS2
- rozhranie VGA
- konektor RS232

[1]source:`https://merlin.fit.vutbr.cz/FITkit/imgs/uvod/001.png`

- DRAM 8x8Mbit
- Klávesnica
- Riadkový LCD displej
- Rozširujúce konektory

Podnetom vzniku tejto platformy bola nutnosť zabezpečiť hardvérovú platformu, ktorá je ľahko dostupná a samostatná, aby študenti nemali nutnosť vlastniť iné hardvérové prípravky, napríklad rôzne programátori.

FITKIT I/O

Fitkit bol navrhnutý so základnými perifériami na rok jeho vydania to je 2008 pre verziu 2. FITKIT obsahuje porty PS2, USB, VGA, USB B, čo bolo na daný rok dostatočné. USB je použité na napájanie FITKITU a komunikáciu s počítačom s priepustnosťou 1MS/s, ktorú potvrdzuje LED 8. Ďalej tam spadá konfigurovateľní dvojkanálový prevodník UART/FIFO, RS232 alebo RS422/RS485 a podporu virtuálneho COM.

LCD displej: Podporuje 16 znakov na riadok, verzia 2 má dvojriadkový displej. Displej obsahuje pamäte ROM na preddefinované ASCI symboly, RAM pre uchovanie 128 Bajtov s podporov definície vlastných znakov. Klávesnica: 4x4 maticová klávesnica VGA:Klasické analógové rozhranie pre pripojenie obrazovky. R,G,B kanály obsahujú 3 bitový DAC pre každý kanál. Podporované rozlíšenie je 604x480 bodov s vykreslovacou frekvenciou 25MHz a snímkovou frekvenciou 60Hz.

FPGA

„Programovateľné logické obvody je skupina viacerých druhov digitálnych integrovaných obvodov, ktorých funkcia je určená užívateľom prostredníctvom predpisu (programu) definujúcemu prepojenie jednotlivých blokov vo vnútri obvodu. Nemyslia sa tým však obvody postavené na mikroprocesoroch(mikrokontroléry).“[7] Jedná sa o programovateľné hardvérové pole, čo je hardvérová súčiastka ktorej štruktúru je možné definovať pomocou konfiguračného reťazca. Spadá do rodiny logických integrovaných obvodov. Štruktúru reprezentuje matica programovateľných blokov(CLB). Vďaka týmto blokom je možné tento hardvér meniť vďaka jazykom na popis hardvéru a netreba fyzicky prepájať, či prepájkovávať súčiastky. Tiež je možné za ich pomoci vytvoriť skoro akýkoľvek číslicový obvod, či už rôzne procesory alebo riadiace obvody. Táto vlastnosť je odlišnosťou od zákaznických integrovaných obvodov, ktorých funkcia je definovaná už pri ich konštrukcii. FPGA obvody majú široký rozsah uplatnení a to vo vývoji zariadení, málo kusových sériách, prototypoch. [3][4]

Ako už bolo spomenuté vyššie pre syntézu sa využívajú takzvané HDL po slovensky, jazyky popisujúce hardvér. K známym HDL patria AHDL, ktorý už je starší

a zameranejší na analógové obvody a potom modernejšie VHDL. VHDL podporuje návrh i simulovanie digitálnych integrovaných obvodov. Ďalšou vlastnosťou je, že podporuje sériové aj paralelné popisy hardvéru. A nakoniec Verilog. Podobný jazyku C poskytujúci návrhy analógových, digitálnych aj zmiešaných obvodov s rôznou úrovňou

MCU

Mikrokontrolér je jednočipový microprocesor prispôsobený pre špecifické koncové aplikácie. Bežne čip obsahuje procesorové jadro, pamäti, časovače a programovatelné vstupno-výstupné periférie.

V dnešnej dobe je už veľmi podobný, ale stále menej zložitý ako SoC. SoC sú jednočipové systémy ktoré nájdeme napríklad v telefónoch alebo v Raspberry Pi. Jednoduchým rozdielom medzi nimi je Soc väčšinou obsahuje, môže ale nemusí, mikrokontrolér. Mikrokontroléry môžeme nájsť všade okolo nás v takzvaných automaticky kontrolovaných produktoch a zariadeniach, ako sú riadiace systémy v automobiloch, v lekárskych implantátoch, kávovaroch, diaľkových ovládaniach, hračkách a v ostatných vstavaných systémoch. Mikrokontroléry pre zmiešané signáli sú veľmi rozšírené, lebo sú schopné pracovať s analógovými aj digitálnymi prvkami v ich okolí. A vďaka rozširovaniu podpory internetu vecí sa ich použitie stále zvyšuje. Okrem nízkej ceny je ich silná stránka aj malá spotreba a schopnosť spať, čiže bežia len vnútorné hodiny, počas ktorej ich spotreba klesá na rády miliwatov až mikrowatov. Vstavaný systém je najrozšírenejšou formou rozšírenia mikrokontrolérov. I keď vstavané systémy môžu byť veľmi sofistikované, tak väčšina má minimálne požiadavky na pamäť, bez operačného systému a malou zložitosťou softvéru. Typicky mávajú na vstupy a výstupy pripojené relé, led diódy, malý LCD displej, rádio súčiastky, rôzne senzory na teplotu, vlhkosť, akceleráciu. Ďalej v týchto vstavaných systémoch zodpovedá za odozvu na podnety. Na základe udalosti môže vzniknúť prerušenie pre ktoré je nutné začať obslužnú rutinu a pozastaviť aktuálnu akciu, po skončení môžu vrátiť obsluhu pôvodne obsluhovanému objektu. Prerušenia môžu signalizovať veci od potvrdení úspešnosti operácie cez rôzne chybové hlásenia až po zobúdzanie zo spánku a spätnému ukladaniu a čakaniu na ďalšiu udalosť na periférii.[8][9]

Mikrokontrolér na fitkite je z rodiny MSP430 Texas Instruments.[10] Jedná sa o 16-bitový nízkonapeťový mikroprocesor s 92kB FLASH pamäti a 8kB RAM pamäti.

3.2 Platforma Raspberry Pi

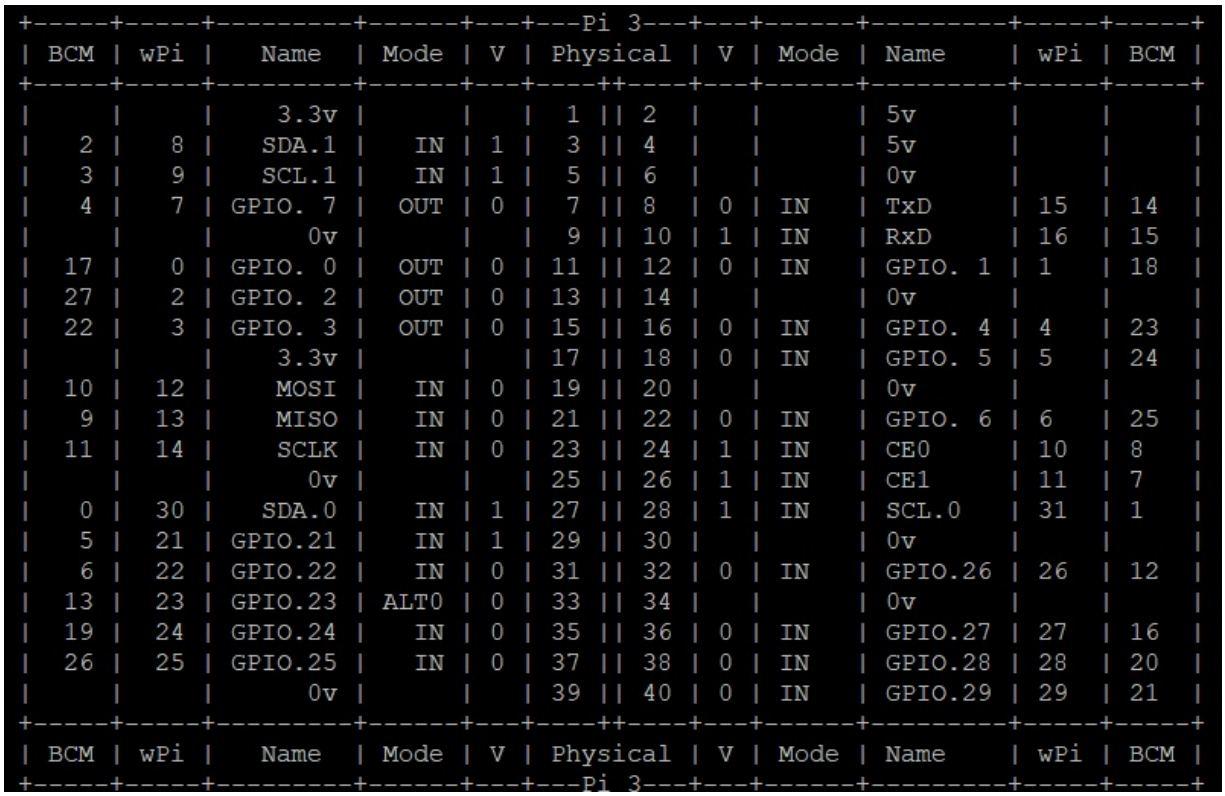
Je veľmi populárny minipočítač. Ide o jednodoskový počítač o približnej veľkosti porovnateľnej ku kreditnej karte. Rozmery udávané výrobcom: 85.60mm x 56mm x 21mm. Za tento malý pc vďačíme: Raspberry Pi Foundantion.

Obr. 3.2: Platforma Raspberry Pi 3 model B[2]

Táto malá krabička v sebe skrýva GPU s podporov OpenGL ES 2,0 hardvérovej akcelerácie a 1080p30 H.264 enkodér a dekodér. GPU tiež zvláda 1Gpixel/s 1,5Gpixel/s alebo 24 GFLOPy pre všeobecné výpočty a viacero textúrových filtrov a DMA infraštruktúr. Toto je zhruba odpovedajúce výkonu pôvodného XBOXsu. Čo sa týka procesoru tak sa Raspberry Pi 3 model b má procesor so štyrmi jadrami o frekvencii 1,2GHz.[17]

Okrem jeho použiteľnosti ako domáceho servera, či média centra má ďalšiu silnú stránku a tou je jeho rozšírené vstupno-výstupné rozhranie, presnejšie nás bude zaujímať 24 konektorov pre všeobecné vstupno-výstupné použitie ďalej len GPIO. GPIO podporuje 3,3V a 5V napájanie s maximálnym prúdom podľa použitého zdroja, s možnosťou nastavovania logických úrovní a programovateľného chovania. Robia spolu s platformou Arduino dobrú voľbu pre výuku, či tvorbu nízkonapäťových obvodov.

[2]source:`https://cf3.s3.souqcdn.com/item/2016/04/07/10/52/20/94/item_XL_10522094_13680816.jpg`



BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM
		3.3v			1 || 2			5v		
2	8	SDA.1	IN	1	3 || 4			5v		
3	9	SCL.1	IN	1	5 || 6			0v		
4	7	GPIO. 7	OUT	0	7 || 8	0	IN	TxD	15	14
		0v			9 || 10	1	IN	RxD	16	15
17	0	GPIO. 0	OUT	0	11 || 12	0	IN	GPIO. 1	1	18
27	2	GPIO. 2	OUT	0	13 || 14			0v		
22	3	GPIO. 3	OUT	0	15 || 16	0	IN	GPIO. 4	4	23
		3.3v			17 || 18	0	IN	GPIO. 5	5	24
10	12	MOSI	IN	0	19 || 20			0v		
9	13	MISO	IN	0	21 || 22	0	IN	GPIO. 6	6	25
11	14	SCLK	IN	0	23 || 24	1	IN	CE0	10	8
		0v			25 || 26	1	IN	CE1	11	7
0	30	SDA.0	IN	1	27 || 28	1	IN	SCL.0	31	1
5	21	GPIO.21	IN	1	29 || 30			0v		
6	22	GPIO.22	IN	0	31 || 32	0	IN	GPIO.26	26	12
13	23	GPIO.23	ALT0	0	33 || 34			0v		
19	24	GPIO.24	IN	0	35 || 36	0	IN	GPIO.27	27	16
26	25	GPIO.25	IN	0	37 || 38	0	IN	GPIO.28	28	20
		0v			39 || 40	0	IN	GPIO.29	29	21

BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM

Pi 3

Obr. 3.3: GPIO pini Raspberry Pi[13]

Knižnica RPi.GPIO

V samotnom popise knižnice ktorý hovorí:„This package provides a class to control the GPIO on a Raspberry Pi .Note that this module is unsuitable for real-time or timing critical applications.. . . “[14] , sa môžeme dozvedieť, že knižnica poskytuje triedu na ovládanie GPIO a že táto metóda je nevhodná pre časovanie kritických aplikácií. Ďalej popisuje nedostatky ako sú nepodporované SPI,I2C, hardvérové PWM, sériová funkčnosť. Sám autor knižnice doporučuje pre real-time časovače platformu Arduino. Hlavným dôvodom pre nevhodnosť riešenia na časovanie autor uvádza „garbage collector“ v jazyku Python a jeho nepredpovedateľné spúšťanie.[18]

Knižnica RPIO.GPIO

Knižnica RPIO je pokročilý modul pre Raspberry Pi. Poskytuje PWM s pomocou DMA s použiteľným minimom zmeny signálu o 1µs. Ďalej predstavuje náhradu za RPI.GPIO pre ovládanie vstupov a výstupov. Tiež podporuje GPIO prerušenia a TCP sokety. Knižnica sa skladá z dvoch komponentov, a to modulu do Python-u 2 a 3 a nástroja do terminálu pre manipuláciu s GPIO. Knižnica obsahuje triedu so základnými metódami prednastavenými pre ovládanie serv pracujúcich s 20 ms

periódou. Okrem tejto triedy modul obsahuje aj iné triedy pomáhajúce s generáciov pulzov a poskytuje aj veľa príkladov s rozsiahlou dokumentáciou.

Wiring Pi utilita

Tiež spojená s menom gpio utilita. Táto utilita vytvára rozhranie pre komunikáciu GPIO pinov a tým pomáha lepšie a hlavne ľahšie komunikovať s GPIO Raspberry Pi. Táto utilita je pre shell-scripting v Bash-y. Príklad: namiesto „echo öut» /sys/class/gpio/gpio18/direction„ sa použije „gpio mode 18 out“. Manuál je samozrejme dostupný pod „man gpio“. Tiež poskytuje podporu k rozširujúcim doskám pripájaním cez GPIO pini. Okrem PWM program podporuje aj kernel moduly ako sú spi a i2c. S týmto prístupom máme garantované hardvérovo časované pulzy, lebo využívame priamo jeden z PWM kanálov. Tomuto odpovedá aj počet pinov pre PWM a to 2, pin 23 a 26.

Projekt ServoBlaster

Softvérový modul na ktorom sú založené dve skúmané riešenia, konkrétne pigpio a pi–blaster. Modul poskytuje rozhranie pre ovládanie viacerých serv z GPIO pinov Raspberry Pi. Ovládanie je založené na posielaní príkazov ovládaču, ktorý na základe požiadavky vyrobí požadovanú dĺžku pulzu a udržuje ju. Modul je prednastavený a testovaný pre osem serv. Aj keď autor uvádza možnú prekonfiguráciu na až 21 serv. Ovládač si vytvára vlastný súbor zariadenia do ktorého sa zapíšu príkazy. Formát príkazov je <servo-number>=<servo-position> alebo P<header>-<pin>=<servo-position>. Prednastavené hodnoty a označenia sa nachádzajú v tabuľke nižšie.

Servo number	Pin number	Pin in P1 header
0	4	P1-7
1	17	P1-11
2	18	P1-12
3	21/27	P1-13
4	22	P1-15
5	23	P1-16
6	24	P1-18
7	25	P1-22

Tab. 3.1: ServoBlaster: Prednastavené mapovanie pinov

Mapovanie pinov na číslo serva je popísané v súbore servoblaster—cfg ktorý je tiež vytvorený v priečinku zariadení. Toto mapovanie je možné zmeniť zadaním správneho príkazu. Všetky pini ktoré bude možné používať a niesú prednastavené

musia byť zadané pred spustením samotného ovládača, lebo ovládač si na základe počtu serv vytvorí ich rozloženie štartovných časov v perióde. Modul je schopný vytvárať PWM signál s pomocou DMA[26][24] a pulzy pre DMA časuje buď z PWM kanálu alebo PCM periférie.

Knižnica piGPIO

Knižnica pigpio je phyton a C knižnica zahrnutá v systéme Raspbian a iné distribúcie ju tiež môžu jednoducho stiahnuť. Táto knižnica podporuje hardvérové PWM, predpokladom je, že využíva ten istý čip ako predošlé riešenie. Rozdielom oproti predošlému riešeniu, ktoré dovoľuje PWM len na dvoch pinoch je PWM na všetkých GPIO pinoch, čiže je možné ovládať 17 serv. Dosahuje toho využitím démona pigpiod, napísaného v jazyku C, bez ktorého zobrazí chybu a žiada o jeho zapnutie. Tento démon/služba zabezpečuje obsadenie hardvéru a komunikáciu s ním. Táto knižnica má viac metód generácie samotného signálu, pre účely ovládania serv a jednoduchosť sa dá využiť funkcie „set_servo_pulsewidth“, ktorá má výstižné meno a nastavuje len dĺžku pulzu. Obmedzenia tejto funkcie sú , že posiela pulzy s rozsahom šírky 0,5 až 2,5 ms čo je rozsah pre servá s pohybom 180°. Medzi hlavné funkcie knižnice patria: hardvérové vzorkovanie a časové značenie od 5µs nahor, hardvérové PWM, hardvérové pulzy pre servá, oznámenia cez komunikačnú rúru callbacky, zvukový zápis ako jedinú operáciu, generáciu vĺn a správu sériovej linky. Ďalej s pripojením nadstavby piscope je možné vytvoriť logický analyzátor vĺn v reálnom čase zobrazujúci priebehy na GPIO pinoch.[14]

Projekt piblaster

Medzi jedno z prvých riešení používajúce hardvérové časovanie, ktoré som našiel patrí tento komunitný projekt, za ktorý vďačíme pánom: Thomas Sarlandie, Richard Hirst a ďalším. Popis od tvorcov je nasledovný: „This project enables PWM on the GPIO pins you request of a Raspberry Pi . The technique used is extremely efficient: does not use the CPU and gives very stable pulses. Pi-blaster project is based on the excellent work of Richard Hirst for ServoBlaster. Pi-blaster has also been forked and made to work with MQTT instead of a local device file. That project is available here.[3]“[15], z čoho vyplýva, že tento projekt dovoľuje PWM na GPIO pine, ktorý si vyžiadame od Raspberry Pi a použitá technika je efektívna, lebo nepoužíva procesor pre generáciu. Potom nasledujú kredity a odkaz na projekt z rovnakého základu ale využívajúci protokol MQTT z čoho usudzujem, že sa dá použiť tiež lokálne ale je predpripravený na preposielanie správ iným zariadeniam na správu.

[3] `https://github.com/gherlein/pi-blaster-mqtt`

Tento projekt využíva vlastného démona/službu, ktorého nastavenia sú popísané v sekcii s riešením. Na časovanie pulzov používa techniku priameho prístupu do pamäte. Pred spustením je treba nastaviť všetky pini, ktoré chceme používať ako výstupy. Po spustení sa vytvorí rúra s FIFO radou pre zapisovanie požiadavok. Formát s požiadavkou je „echo "BCM=hodnota» /dev/pi-blaster“, kde hodnota je medzi 0 až 1. Táto hodnota predstavuje percentuálny podiel doby signálu v logickej 1.

Raspberry Pi I/O

Raspberry Pi 3 model B, ktorý som použil má 4 USB A porty, fastethernet, 802.11ac/n wifi, HDMI v1.3a, mikro USB pre napájanie, 2.5 audio jack, fast ethernet. Iné modely síce uvádzajú že podporujú Gigabit ethernet, ale je to prepojené s USB takže rýchlosť bude končiť okolo 400Mb/s. Samozrejme nesmiem zabudnúť na 40 pinov. Tieto pini sa nachádzajú na okraji dosky. GPIO sa dá softvérovo prednastaviť ako vstup či výstup so širokou možnosťou uplatnenia. Z týchto pinov je 17 pre všeobecné použitie, 6x GND, 2*5V napájacie, 2*3,3V napájanie a ostatné pini sa dajú použiť ako GPIO pini, ale sú schopné aj alternatívnych funkcií alebo sú navrhnuté na pripojenie periférií, či prednastavené pre použitie určitých protokolov. Vstupy aj výstupy sú nastavené na spracovanie logických signálov. Výstupy sú navrhnuté pre logickú 1 na 3V3 a logickú 0 na 0V. Vstupy sú na tom podobne a tiež čítajú v rozmedzí od 0V do 3V3. Ďalej majú podporu nastavenia pull-up a pull-dowm registru. Až na výnimku dvoch gpio pinov je možné tento register nastaviť softvérovo. Ako bolo spomenuté vyššie tak niektoré pini poskytujú alternatívne nastavenia ako sú PWM, SPI, I2C, Serial.

3.3 Platforma personálny počítač

[12] Osobný počítač tiež nazývaný komputer, computer anglicky presonal computer, so skratkou PC je počítač ktorého veľkosť, možnosti a cena dovoľujú jeho osobné použitie, či použitie jednotlivcom. Osobné počítače sú určené na prevádzku priamo koncovým užívateľom, a nie technikom alebo počítačovým expertom. Na rozdiel od mainframov, minipočítačov určených pre obsluhu mnohých požiadaviek a zdieľanie zdrojov medzi mnohými používateľmi sú osobné počítače určené práve pre jedného používateľa. Práve z tohto vychádza názov osobný počítač, lebo v čase jeho vzniku boli počítače veľmi rozmerné a nie len pre jedinú osobu. Osobný počítač vznikol kombináciou rôznych štandartou, z ktorých najväčšiu kontribúciu štandartou majú IBM a Apple. V dnešnej dobe je oveľa väčšie percento osobných počítačov a preto keď dnes niekto použije pojem počítač, tak vlastne používa synonymum

pre osobný počítač, alebo aj počítačovú zostavu s jej vstupnými aj výstupnými zariadeniami. Ďalej sa tento termín rozšíril na konci sedemdesiatych a začiatkom osemdesiatych rokov minulého storočia, práve keď sa menil pomer počítačov v laboratóriách a domácnostiach. Spolu s prechodom počítačov do domácnosti prišla aj zmena významu z vedeckého ponímania na zábavné zariadenie. Dnešné osobné počítače sú ponímané a využívané ako zábavné elektrospotrebiče pripojené na internet a so schopnosťou prehrávania, modifikácie, ukladania rôzneho multimediálneho obsahu, tiež môže slúžiť ako prepojenie medzi ostatnými zariadeniami. Osobné počítače v dnešnej dobe sú postavené najmä na báze procesorov architektúry x86 objavujú sa aj procesory ARM ale len v menšom počte. Ďalej obsahujú výkonnú grafickú kartu, ukladacie médium s úložiskom v stovkách GB až jednotkách TB na starších mechanických diskoch alebo novších netočiacich sa diskoch SSD, operačnú pamäť v jednotkách až desiatkach GB, pripojenie do internetu a lokálnej počítačovej siete. Počítače tiež obsahujú porty na pripojenie periférií ako sú LCD monitory, tlačiarne, vstupné a výstupné zariadenia a iné zariadenia schopné komunikovať s počítačom ako mobilné telefóny či hudobné prehrávače. V minulosti si užívatelia museli písať vlastné programy a aplikácie pre využitie hardvéru v počítači no dnes je programové vybavenie počítačov uspôsobené tak aby ich mohol ovládať i neznalý užívateľ. Toto zabezpečuje operačný systém počítača. Najrozšírenejším je systém spoločnosti Microsoft pod menom Windows, medzi najpopulárnejšie verzie patria 7 a 10. Ako alternatívne možnosti pre operační systém je macOS od spoločnosti Apple alebo jeden z Open Source systémov s Linux-ovým jadrom.

I/O personálneho počítača

Väčšina personálnych počítačov obsahuje rozhranie USB typu A cez ktoré je možné naviazať sériovú komunikáciu za pomoci virtuálneho COM portu. Veľmi staré modely mávali fyzický sériový port. Bohužiaľ dnešné personálne počítače nemajú podporu pre hardvérovo časované pulzy, je síce možné vyrobiť softvérovo časovaný pulz ale tieto pulzy nebývajú tak presné a sú ľahko ovplyvniteľné prerušeniami či zaťaženým procesoru.[4]. Medzi ďalšie I/O ktoré sa dá nájsť je výstup pre zvuk, či už 3,5mm kombo(štvor pražcový) alebo klasický(dva s tromi pražcami) jack, fast/gigabit ethernet, výstup je obraz analógový(VGA, ...) alebo digitálny(hdmi, display port, ich mini varianty, DVI len na starých modeloch, ...), pre moderné prenosné počítače sa pre úsporu miesta používa USB typu C, ktoré zvláda niesť viacero typov signálov zároveň(napájanie, obraz, usb komunikáciu, ...). I/O personálnych počítačov nie je uspôsobené pre pripájanie nízkonapäťových obvodov a je treba využiť

[4]softvérové pulzy predvedené v sekcii Raspberry

rôznych programátorov, mikrokontrolérov alebo iných špecializovaných prípravkov prenášajúcich požadovanú komunikáciu cez a na USB zbernicu. [12]

3.4 Platforma Arduino

Arduino je open-source hardvér a softvér spoločnosť, projekt a užívateľská komunita, ktorá navrhuje a vyrába jednodoskové mikrokontroléry a prípravky s mikrokontrolérmy pre rôzne skladby, či interaktívne objekty schopné vnímať a kontrolovať analógovo aj digitálne. Tieto produkty sú licencované pod GNU Lasser General Public License alebo pod GNU General Public License, ktoré dovoľujú ako výrobu Arduino dosiek tak softvérovú distribúciu kýmkoľvek. Arduino je na trhu dostupné ako predpripravený zložený výrobok alebo ako skladačka pre nadšencov.

Obr. 3.4: Platforma Arduino[5]

Návrhy Arduino dosiek používa rôzne mikroprocesory a kontroléry. Dosky sú vybavené súbormi digitálnych a analógových vstupno-výstupných pinov, cez ktoré je možné pripájať rôzne rozširujúce dosky, nepájavé polia a iné obvody. Dosky sú schopné sériových komunikácií, vrátane USB konektoru na určitých doskách. Tieto zbernice sú používané na nahrávanie programov z osobných počítačov. Mikrokontroléry sú typicky programované za využitia dialektu z programovacích jazykov C a C++. Navyše oproti využitiu tradičných kompilačných nástrojov projekty Arduina poskytujú integrované vývojárske prostredie na základe použitého jazyka projektu.

Projekt Arduino začal v roku 2003 ako program pre študentov na Interaction Design Institute Ivrea v Ivree, Taliansko. Cieľom bolo poskytnutie nízko-nákladnej a jednoduchej cesty pre začiatočníkov aj profesionálov k vytváraniu zariadení schopných interakcie s prostredím s využitím rôznych senzorov a motorov.

[4]source:`https://store-cdn.arduino.cc/uni/catalog/product/cache/1/image/520x330/604a3538c15e081937dbfbd20aa60aad/a/0/a000066_featured_1_.jpg`

Arduino I/O

Väčšina dosiek Arduino má jednoduché I/O, ktoré bežne obsahuje sadu analógových a sadu digitálnych pinov vždy na opačných okrajoch dosky. Konektor na sériovú komunikáciu, napájací konektor a napájacie pini. Dnešné modely ponúkajú 14 pinov pre všeobecné vstupno-výstupné použitie a z toho 6 pinov s podporov PWM v skupinách po dvoch. Vždy dva pini na jeden časovač. Ďalej má doska 6 analógových pinov, ktoré môžu zastať úlohy digitálnych pinov.

4 Analýza súčasného stavu a návrh riešenia

Dnes existuje veľa platforiem aj viacero prístupov k ovládaniu modelárskych serv, či všeobecnému generovaniu PWM signálov, z danej platformy. Pre príklad zoberieme najznámejšie platformy, menovite Arduino, Raspberry Pi, tak máme k dispozícii najmenej sedem rôznych spôsobov generácie PWM signálu, k tomu sa môžu pripočítať menej známe knižnice, či obrovské množstvo mikrokontrolérov, dosiek plošných spojov, ktoré sú tiež schopné generovať PWM signály. V rámci zadania a práce je teda obmedzený počet platforiem. Teda v rámci práce sa zameriam na platformy Raspberry Pi, FITKIT, ktorý zastupuje Mikrokontroléry a logické obvody, komerčnú dosku Micro maestro. Na týchto platformách budem skúmať a zhodnocovať ich parametre ako sú presnosť, stabilita, efektivita a iné. Ďalej je treba napísať riadiaci program a zostrojiť jednoduchý strojček pre fyzickú reprezentáciu funkčnosti.
Základné požiadavky na platformu zahŕňajú:

- PWM signál musí generovať platforma sama
- Cena by mala byť porovnateľná s komerčným riešením
- Ako napájanie musí stačiť batéria, buď modelárska 6V alebo 5V z USB powerbanky
- Musí obsluhovať viacero serv, minimálne 4
- Stabilita signálu na 0,1%

4.1 Platforma Raspberry Pi

Pretože táto platforma poskytuje veľké množstvo riešení a prístupov je ich treba kategorizovať a bližšie skúmať zástupcov z daných kategórií. Kategória je určená na základe spôsobu generácie riadiaceho signálu.

Kategória SW časovačov

Zástupca zo softvérových riešení je Rpi.GPIO. Tento spôsob reprezentuje riešenia ktoré na vytvorenie riadiaceho signálu používajú len procesor s aktívnym alebo pasívnym čakaním. Od tohto riešenia sa neočakáva veľká pestrosť ani stabilita signálu, no treba ho zahrnúť, kôli veľkému množstvu doporučený a tutoriálov, ktoré ho využívajú.

Kategória HW časovačov

Ďalším skúmaným zástupcom bude Wiring Pi tiež uvádzané ako GPIO utilita pre terminál. Spadá do kategórie hardvérových riešení, čiže riešenie využívajúce hardvérové hodiny nezávislé na zbytku systému. Toto konkrétne riešenie využíva kryštálu

priamo na doske. Riešenie bolo zvolené z dôvodu jeho predpokladanej presnosti, nízkej náročnosti a zahrnutiu v oficiálnom operačnom systéme. Známa nevýhoda je nedostatočný počet kanálov. Konkrétne iba dva, čo znamená že nespĺňa základné požiadavky na platformu, treba preskúmať možnosti lacného a jednoduchého rozšírenia pre ovládanie väčšieho počtu serv.

Kategória s využitím DMA

Pre kategóriu využívajúcu priameho prístupu do pamäte pre generáciu riadiaceho signálu sa odkloním od jedného zástupcu, lebo z dokumentácie a rýchleho náhľadu do kódu nie je jasné, či dosahujú rovnakú stabilitu, presnosť alebo ich nárok na systémové zdroje. Obe tieto riešenia majú v sebe zaintegrované rovnaký projekt riešiaci DMA na tejto platforme a sú teda jeho nadstavbou. Z tohto dôvodu zdieľajú niektoré vlastnosti. Konkrétne minimálny krok ktorý sú schopné generovať. Možnosť výberu vnútorného časovača pre DMA, a to buď periférium PWM alebo PCM. Ďalšia spoločná vlastnosť je posielať predvolený riadiaci signál na všetky GPIO pripojenia. Tiež zdieľajú vlastnosť rezervácie periférií, čo spôsobuje nemožnosť menenia frekvencie signálu po spustení obslužnej služby, pre moje potreby to nemá vplyv.

4.2 Platforma FITKIT

Pre FITKIT sa ponúkajú dva prístupy. FPGA predstavujúce špecializované platformy a MCU zastupujúce obrovský počet rôznych čipov mikrokontrolérov. Takže namiesto testovania rôznych dosiek a MCU čipov sa zameriam na čipy z FITKIT dosky, ktoré pokryjú základnú techniku generácie riadiaceho signálu.

Riešenie s využitím FPGA

Základná frekvencia čipu na FITKITe je 7.3728MHz. Riešenie pomocou FPGA má veľmi jednoduchý základ pre generáciu požadovaného signálu. Princíp spočíva z počítadla, ktoré beží voľne a inkrementuje sa, s každou nábežnou hranou hodinového signálu a nuluje sa pri dosiahnutí požadovanej periódy. Následne register, v ktorom je uložená šírka nášho požadovaného signálu. Poslednou súčasťou je komparátor, ktorý má na vstupoch čítač s registrom a jeho výstupom je môj signál s požadovanou frekvenciu a šírkou. Toto riešenie je schopné generácie signálu na všetkých fyzických výstupoch.[30]

Riešenie s využitím MCU

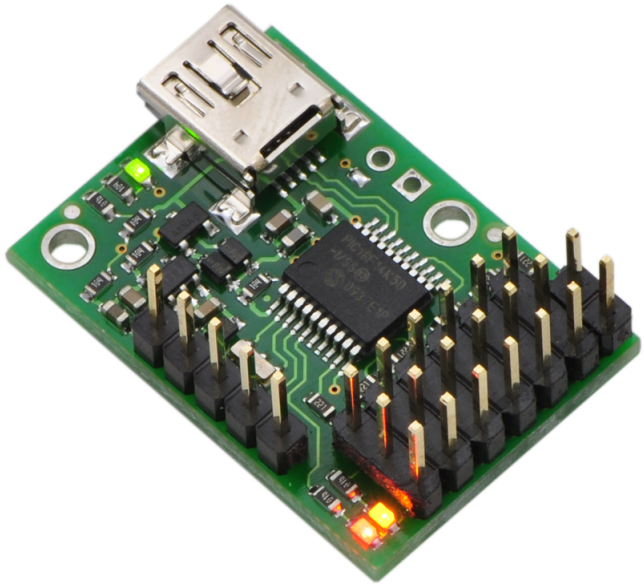
Riešenie bude využívať prerušenia k časovaniu riadiaceho signálu. Časovanie bude založené na príklade blikania LED dostupnom z oficiálnych stránok FITKITu. Z tohto príkladu je možné vyčítať základnú frekvenciu čipu na doske, nominálne je to frekvencia 32768Hz, čo zodpovedá 0x8000 v hexadecimálnom zápise. Toto riešenie môže byť náchylné na inú záťaž na periférиách, najmä na komunikáciu cez USB zbernicu.

4.3 Platforma Personálny počítač

Personálne počítače nemajú GPIO zbernice ani neboli navrhované s úmyslom generácie signálov potrebných pre účel riadenia serv. Pre ovládanie serv z počítača je teda potrebná externá doska s mikrokontrolérom. Napríklad Micro Maestro 6-Channel USB Servo Controller, ktorý poskytuje potrebné prepojenie aj časovače na ovládanie.

Pololu mikro maestro

Dosky s tejto rodiny boli navrhnuté pre ovládanie serv. Preto sa dá táto možnosť pokladať ako komerčné riešenie ovládania. Táto rodina má viac členov s rôznim počtom pripojiteľných serv.



Obr. 4.1: Pololu micro maestro[1]

[1]sourco:https://a.pololu-files.com/picture/0J1951.1200.jpg?7e54ef728c9e5b6707a45ae5868bfb66

Mikro maestro je možné pripojiť k počítaču pomocou mini USB. Možnosti ovládania sú celkom rozšírené konkrétne:

- grafické rozhranie
- skript
- ovládanie jazykmi s rozšírenou komunikáciou USB rozhrania za pomoci balíčku pre vývojárov.

Grafické rozhranie je priamočiare a je prednastavené na rozsah predstavujúci bezpečný rozsah pohybu serva, čiže 50Hz o šírke 1ms až 2ms. Tento rozsah sa dá rozšíriť pre jednotlivé kanály kvôli kalibrácii serv. Pre skriptovanie je potrebné, aby nebol program v rozpoznávaní komunikácie a bola stanovená rýchlosť a ostatné parametre komunikácie. Použitý je špeciálny jazyk pre meastro dosky. Jedná sa o „maestro script language“, jazyk je štruktúrovo podobný assembleru.

Za použitia development kitu je možné písať vlastné aplikácie napríklad v C#, Visual Basic, .NET a Visual C++.

Parametre

Od dosky Pololu micro maestro 6 môžeme očakávať tieto parametre:

- **Samostatné kanáli**: 6
- **Podporované protokoly**: USB, UART
- **Kompatibilné napájanie**: 5–16V
- **Podporované frekvencie**: 33 – 100Hz
- **Minimálny krok**: 0,25µs

Vlastnosti jednotlivých riešení

Tabuľka 4.1 zobrazuje prehľad riešení, ktoré budú podrobené testovaniu. Tiež zobrazuje základné vlastnosti týchto riešení.

4.4 Návrh testovania a modelu

Pre demonštračné účely bude treba vytvoriť jednoduchý strojček. Za pomoci tohto modelu bude vytvorený záznam presunu predmetu s využitým testovaných riešení. Zároveň bude použití pri meraniach a testoch riešení. Model bude obdobou robotického ramena zkonštruovaného z troch pohybových uzlov.
Po tomto ramene budem potrebovať:

- Pohyb musí byť spôsobený modelárskymi servy.
- Ľahké vymieňanie riadiacej platformy.

[2] in integer space, less in float space

Platforma	**Raspberry Pi**				**FTKIT**		**pc**
Riešenie	**rpi.gpio**	**wiring pi**	**pigpio**	**pi-blaster**	**FPGA**	**MCU**	
resource	CPU	HW timer	PWM channel or PCM for DMA timing	PWM channel for DMA timing	FPGA	CPU interupts	HW timer mikro maestro
kanály	17+	2	17+	17+	5+	4	6
base Langluage	py	bash	c	c	vhdl	c	its own
Lang. support	py	any that can use system() call	py or c	any that can write to file	hdl	C or py needs programer	c based, java, .NET ...
step	0.5ms[2]	<0.25 µs	2µs	2µs	<18µs	3µs	0.25µs

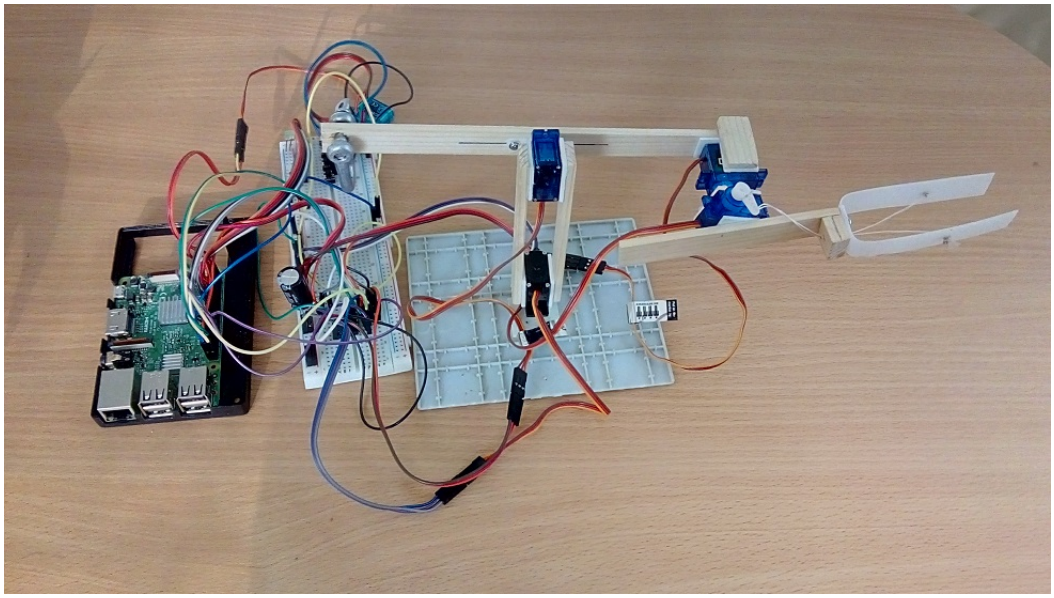
Tab. 4.1: Vlastnosti jednotlivých riešení

- Model musí byť schopný zdvihnúť nejaký objekt.
- Model musí byť schopný udržať zdvihnutý objekt pri jeho premiesťňovaní.
- Finančná náročnosť

- Prenositeľnosť

Návrh Modelu z prototypu

Ako prototyp modelu slúži drevené rameno na obrázku 4.2, ktoré slúži ako základná predstava o ramene. Z prototypu je možné návrh pre 3D tlač rozdeliť do niekoľkých častí. Hlavnou časťou bude základňa, v ktorej musí byť koncentrovaná váha, aby mohla poskytnúť stabilitu a podporu celej konštrukcii. Stožiar a rameno bude treba skrátiť a vystužiť vzhľadom na použité servá. Úchop musí byť schopný udržať objekt bez zbytočnej námahy na servo. Pre väčšie množstvo riadiacich platforiem a rôznych riešení na nich je treba zabezpečiť ľahké prepojenie, alebo prepnutie na inú riadiacu platformu. Čiže integrácia riadiacej platformy priamo do modelu nie je žiaduca. Ako bolo spomenuté vyššie hlavným materiálom pre tento model bude plast vrstvený 3D tlačiarňou.

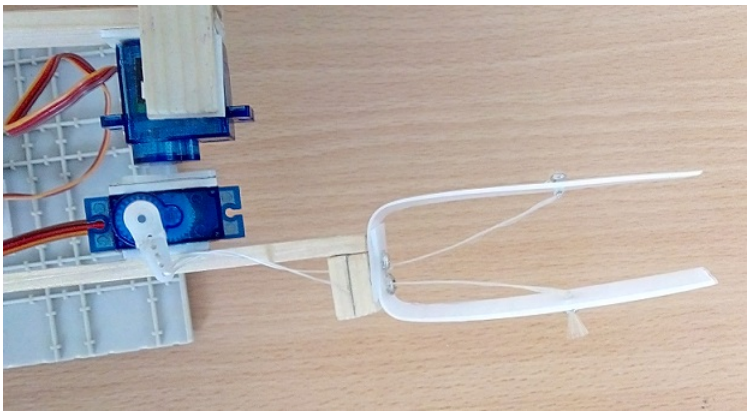


Obr. 4.2: Drevená demo aplikácia

Návrh Testov

Na vyhodnotenie či je platforma s konkrétnym riešením spôsobilá je treba určiť vlastnosti na ktoré sa treba zamerať. Pre moje potreby to bude najmä:

- stabilita generovaného signálu a to najmä jeho šírky, nominálne budú testy uspôsobené na 200ns
- počet výstupných kanálov, minimálne 4
- krok s ktorým sa dá posúvať



Obr. 4.3: Drevená demo aplikácia pohľad zhora

- stabilita periódy, ktorú je možné skoro zanedbať pri použití kvalitnejších serv.

Všeobecné testy a pokusy robené na každej platforme budú zahŕňať kontrolu základných vlastností daného riešenia.

Všeobecný test vlastností

Tento test je zameraný na kontrolu vlastností riešenia. Takže nastavenie serva do neutrálnej polohy a do krajných polôh. Kontrola nastaveného kroku. Pripojenie štyroch, piatich alebo šiestich serv podľa platformy a ich samostatnosť.

Špecifický test vlastností platformy Raspberry pi

Tento test je zameraný priamo na platformu Raspberry pi s odhliadnutím od frekvencie pre riadenie serv. Z tohto testu získam informácie potrebné pre dizajn testov mierených na túto platformu a zároveň môžem odlíšiť pôvod nestability z riešenia od platformy.

Všeobecný vizuálny test

Rýchli vizuálny test nadväzujúci na predošlí test. Spočíva v pripojení záťaže na servá a ich následné nastavenie do náhodných polôh. Pre tento test je dôležité sledovanie serv na chvenie a iné prejavy nestability. Test môže odhaliť aj nestabilitu, ktorá je spôsobená šumom v napájacej vetve.

Všeobecný test s modelom

Z tohto testu bude vznikať výslední záznam a demonštrácia funkčnosti riešenia. Obsahom testu bude presunutie ramena k predmetu, jeho následné uchopenie, zdvi-

hnutie a presun. Počas testu sa bude pozorovať chvenie stabilita ako samostatných serv tak aj modelu.

Všeobecný test s osciloskopom

Bude prebiehať v podobnom duchu ako test vlastností, ale s dôrazom na presnosť generovaného signálu. Tento test sa bude držať predpísaných testov pre kalibráciu serv. Predvolené hodnoty teda budú 20ms pre periódu a 2ms šírka. Z tohto testu budú získané údaje ako je smerodajná odchýlka na šírke signálu, maximálna chyba od stabilnej hodnoty, odchýlka od ideálneho signálu. Pre periódu je dostačujúci údaj maximálni záchvev.

Špecifický test s osciloskopom pre Raspberry pi

Jedná sa o rozšírenie všeobecného testu na základe predošlého špecifického testu na platforme. Je treba započítať vplyv samotného operačného systému lebo je použitý plnohodnotný operačný systém. Ide o sériu testov mierených na potencionálne nedostatky vybraných riešení.
Základná séria pre všetky riešenia zahŕňa:

- Testovanie pri nezaťaženom procesore
- Testovanie pri čiastočnom zaťažení – zahŕňal len pozorovanie bez uchovania údajov
- Test pod vysokou záťažou systému: zahŕňa zaťaženie procesora na hranu napájacích problémov, vyťaženie USB zbernice

Cielená séria testov

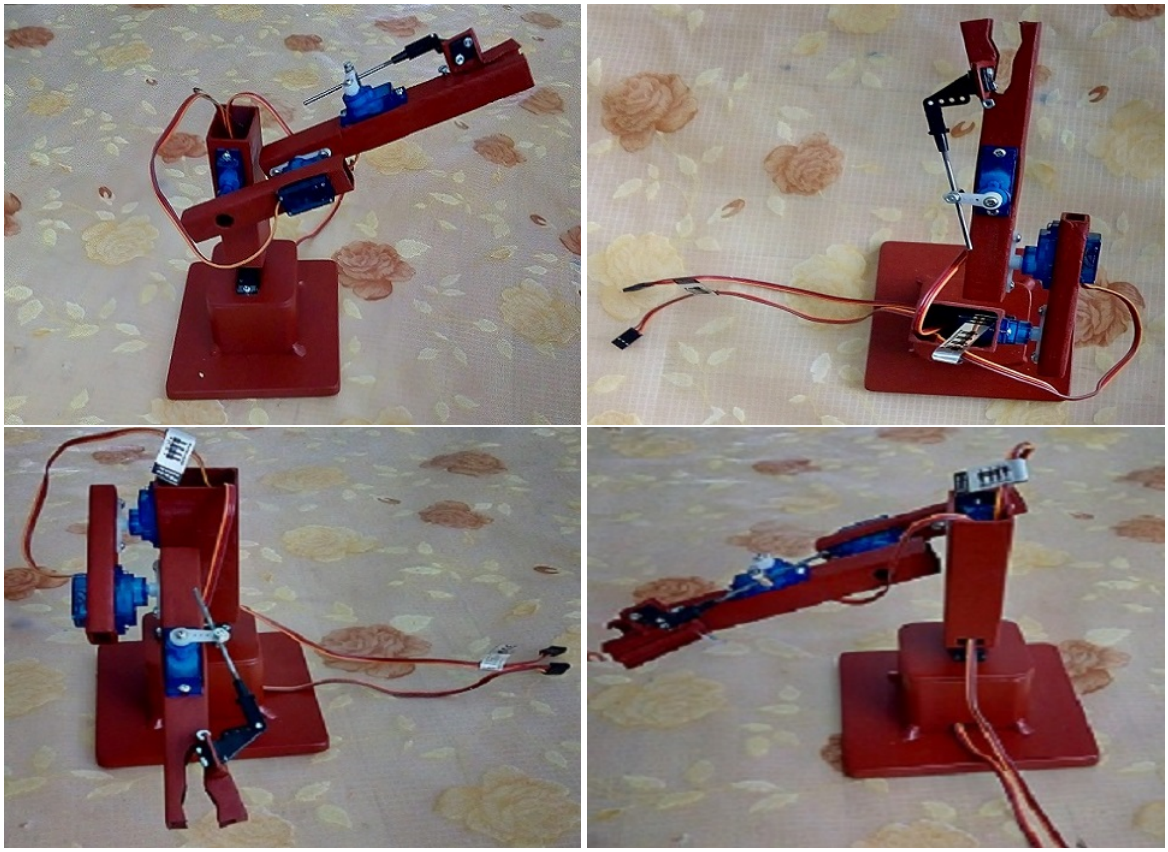
Pre kategóriu DMA tu pribúda sledovanie vyťaženia systémových zdrojov. Všetky riešenia v tejto kategórii majú určitú formu démona/služby, ktorý je zodpovedný za rezerváciu zdrojov a dekódovanie požiadavku na zmenu stavu registru.

Zaťaženie PWM kanálu pre zvuk. Test mierený na potencionálny nedostatok na platforme Raspberry pi, Teória za týmto testom je spojená s využitím prvého PWM kanálu, ktorý je bežne využívaný systémom pre spracovanie analógového zvuku. Tu nie je jasná interakcia na základe dokumentácie ani inšpekcie kódu. Očakávaná interakcia je nefunkční analógový zvuk z dôvodu predošlej rezervácie jedným z riešení.

5 Zhodnotenie riadenia serv z rôznych platforiem

V tejto kapitole sa zaoberám zvolenými riešeniami na platformách: Raspberry Pi, FITKIT, Personálnom počítači. Platformu Arduinu som netestoval z dôvodu, že na ňu existuje veľa návodov a je dobre preskúmaná. Všeobecné testy vlastností boli robené s pomocou osciloskopu poskytnutého školou. Ale jeho technické prevedenie obmedzilo presnosť meraní, presnosť v desatinách ms. Preto bol použitý iba na prvotné všeobecné testy vlastností a jednoduché vizuálne testy. Pre jeho nedostatočnú presnosť bol nahradený iným prenosným osciloskopom s 16kB pamäťou na vzorky, čo dovoľuje pri potrebnej perióde 20ms a šírke pulzu 2ms presnosť na 0,2 µs.

5.1 Demo aplikácia



Obr. 5.1: Výsledný model demo aplikácie

Súbory k tlači i návrh nájdete na odovzdanom médiu. V súbore print sú stl súbory mnou predpripravené k tlači, doporučujem si ich skontrolovať. Medzi ostatnými

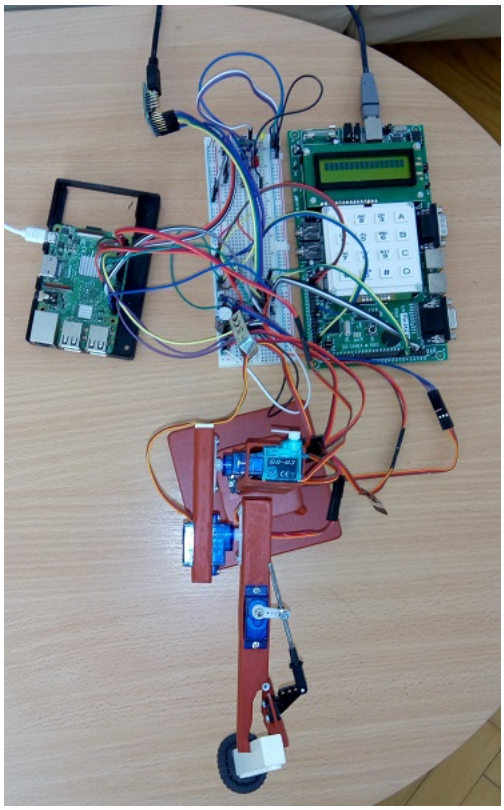
súbormi je an 3D model vo formáte pdf aj originálne súčiastky z Solid Edgeu a s pôvodne exportovanými stl súbormi. Ak si chcete model vytlačiť je treba brať v úvahu otvory predpripravené pre mnou použité serva: GO-13 a 3x SUMO 1199BU.

Pripojenie dema k platformám

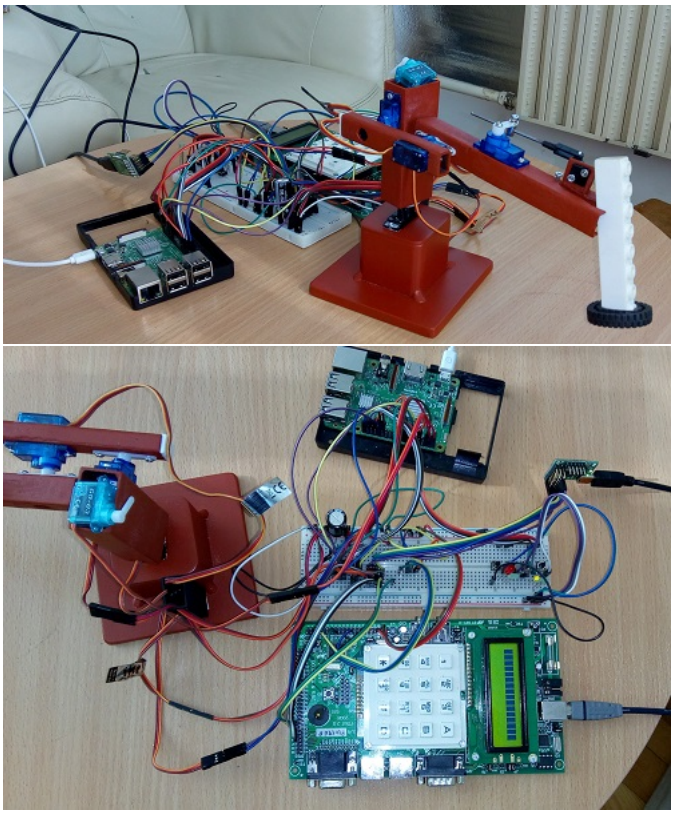
Pre ukážky jednotlivých riešení som prepojil platformy cez nepájavé pole. Napájanie je z pinou zbernice JP9 na FITKITe. Všetky platformy zdieľajú zem. Demo nie je pripojené k všetkým riešeniam zároveň. Riešenia sú rozdelené do troch skupín:

- RPI riešenia: RPi.GPIO, GPIO, pigpiod
- FITKIT FPGA, Pololu micro maestro
- FITKIT MCU prerušenia

V zapojení na obrázku 5.2 je demo pripojené k FITKITu FPGA a Pololu micro maestru. Pre tieto platformy a riešenia som sa rozhodol pre ich dobré výsledky počas testov, s výnimkou RPi.GPIO(sw PWM) ktoré je pripojené na samostatné servo a MCU s prerušeniami, ktoré síce nie je vhodné ako reálne riešenia ale dá sa použiť k demonštrácii pohybu.



Obr. 5.2: Pripojenie demoaplikácie k platformám, zapojené pololu maestro(stand by) a FITKIT fpga(signál) 1/2



Obr. 5.3: Pripojenie demoaplikácie k platformám, zapojené pololu maestro(stand by) a FITKIT fpga(signál) 2/2

Záznam k Demu

Výsledný záznam k demu som zachytil za pomoci kamery VEGA 5 fun na statickom stojane. Pomocou riešení z predchádzajúcich sekcií menovite ako sú zoradené v zázname: PC s mikro maestrom, FITKIT na MCU, FITKIT na FPGA, Raspberry Pis knižnicou pigpio a Raspberry Pis wiring Pi utilitou. Záznam bol upravený a zostrihaný v programe Blender, ktorý bol použitý i pri návrhu k tlači. Hudba pochádza od Royalty Free Music from Bensound[1].

Výsledný záznam bol nahraný na platformu YOUTUBE a je dostupný na adrese: `https://youtu.be/zP1TQnDVWzc`

[1] `www.bensound.com`

5.2 Riešenia platformy Raspberry Pi

Bola použitá verzia: Raspberry Pi 3 Model B s operačným systémom Raspbian, zdroj 5V,2.1A alebo power banka(16000mAh, 2A). O časovanie a generáciu signálov sa primárne využíva čip BCM2837.[32] Tento čip je napojený na kryštál s frekvenciou 19,2Mhz, tiež je zodpovedný za správu iných periférií, ďalšou funkciou tohto čipu je aj správa prerušení. Schému PWM kanálov čipu zobrazuje obrázok 5.4. Tento čip je schopný generovať PWM signál podľa dvoch algoritmov. Pre účely práce je vhodné použiť druhý algoritmus. Mód Mark-space, do ktorého sa dá prepnúť príkazom „pwm-ms“. Tento algoritmus posiela dáta s pomerom M/S, kde M sú dáta na poslanie a S je rozsah. Toto potom na výstupe spôsobí, že M vzorkou je hore z celkových S možných. Pre lepšiu predstavu majme M=2 a S=4, čo je presne 50% pracovná doba pulzu, t.j. polovicu periódy bude pulz v logickej jednotke a druhú v nule. Pre porovnanie tento algoritmus by zmenou M zmenil len pracovnú dobu a perióda by ostala konštantná, ale prvý algoritmus by zmenil aj samotnú periódu.

Raspberry Pi pre generáciu a určenie frekvencie používa nasledovný vzorec:

$$Frequency = \frac{basefrequency}{pwmclock * pwmrange}$$

Kde base frequency zodpovedá 19,2Mhz a pwmclock s pwmrange sú deličky s ktorými môžeme manipulovať na dosiahnutie požadovanej frekvencie.

$$50Hz = \frac{19200000}{pwmc * pwmr}$$

Nastavenie druhej deličky priamo nadväzuje na minimálny krok, ktorý je možné spraviť. Vzorec na výpočet kroku teda vyzerá:

$$\frac{perióda(20ms)}{pwmr(8000)} = 2,5\mu s \quad krok$$

Rozsah deličiek je: pwmc [2...4095] a pwmr [2...∞].

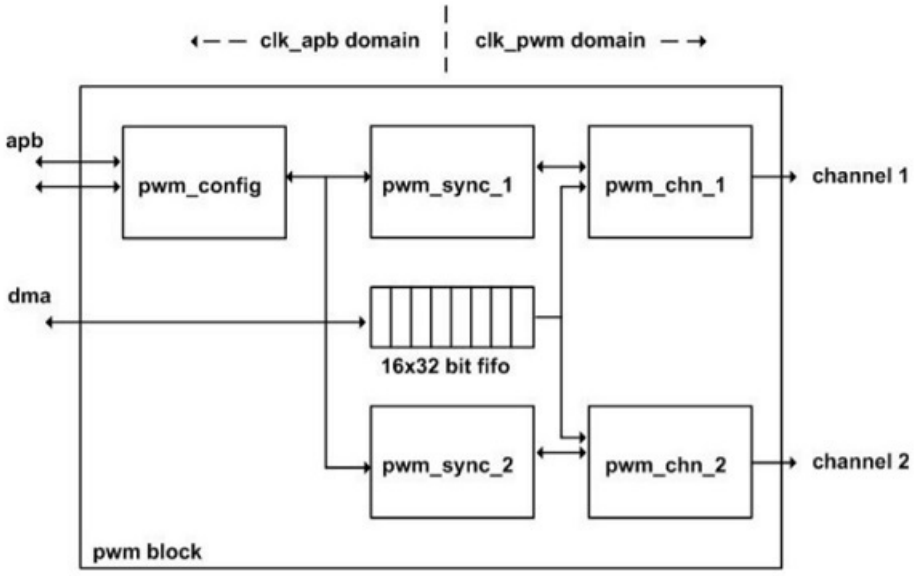
I keď je teda základná frekvencia 19,2 MHz, tak minimálne nastavenia deličiek dovoľujú použiť len štvrtinu, čo zodpovedá 4,8Mhz pri nastavení oboch deličiek na 2 a pracovným cyklom na 1, čiže 50%. Tieto nastavenia deličiek pre frekvenciu sú zdieľané pre oba PWM kanály z čoho vyplýva, že zdieľajú frekvenciu a môžeme meniť len dĺžku pulzu samostatne.

Špecifický test vlastností platformy Raspberry Pi

Výsledky testov k samotným riešeniam nájdete v sekcii daného riešenia.

Ako základný test platformy som použil syntetický stress test na procesor. Pri teste bol výstup PWN nastavený na najvyššiu manipulovateľnú hodnotu, nominálne

4,8MHz, stress testom na procesor boli zistené dve vlastnosti. Za prvé sa potvrdil kryštál ovládaný čipom BCM2837, ktorý pri zahriatí spôsoboval chvenie v desatinách mHz a nižšie. Druhou je napájací obvod cez mikro-USB. Nielenže cestičky na doske od USB k procesoru sú výraznejšie tenšie ale aj idú dlhšou cestou oproti cestičkami medzi procesorom a GPIO a týmto pádom cez USB sú nielen väčšie straty ale aj nižšie možné napájanie oproti napájaniu z GPIO. Z tohto dôvodu doporučujem napájanie cez GPIO ak chcete používať Raspberry Pi dlhodobejšie s vyšším zaťažovaním procesoru.

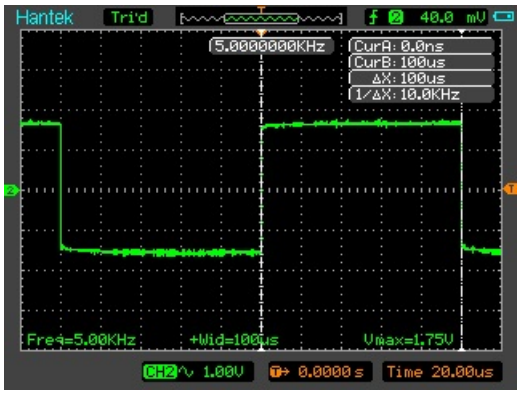


Obr. 5.4: Raspberry Pi schéma PWM kanálov[2]

Druhotné testy platformy na osciloskope pod zaťaženým boli navrhnuté z dôvodu prehriatia a vypnutia systému za použitia syntetických testov. Optimalizovaný test pod záťažou mal za cieľ zaťažiť procesor niekde v rozsahu 80-90% na každom jadre. Dosiahol som toho za pomoci VNC serveru pre vzdialenú plochu a prehrávaným HD(720p) alebo fullHD(1080p) videí obe 60fps bez použitia hardvérovej akcelerácie. Pri HD to robilo 90-98%. Zároveň som zaťažoval USB zbernicu prehrávaním týchto súborov z prenosného média.

Druhotný test som tiež pripojil na generovanú frekvenciu 5kHz, kde som sledoval stabilitu signálu pod touto záťažou5.5. Výsledkom bola nemerateľná zmena lebo ani frekvencia ani šírka sa nezmenila. Najmenšia zmena ktorú som bol schopný zmerať

[2]source:`https://cs140e.sergio.bz/docs/BCM2837-ARM-Peripherals.pdf`

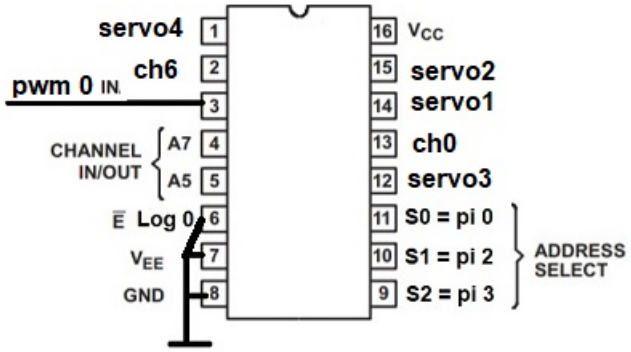


Obr. 5.5: Raspberry Pi PWM kanál pod záťažou(druhá séria testov)

na kHz bola 0,1mHz, no ani daný záchvev som nezachytil. I keď sa signál generuje pomocou kryštálu tak som žiadne zmeny na signály nenameral, čo predstavuje dostatočnú základňu pre ďalšie testy na modelárskych servách.

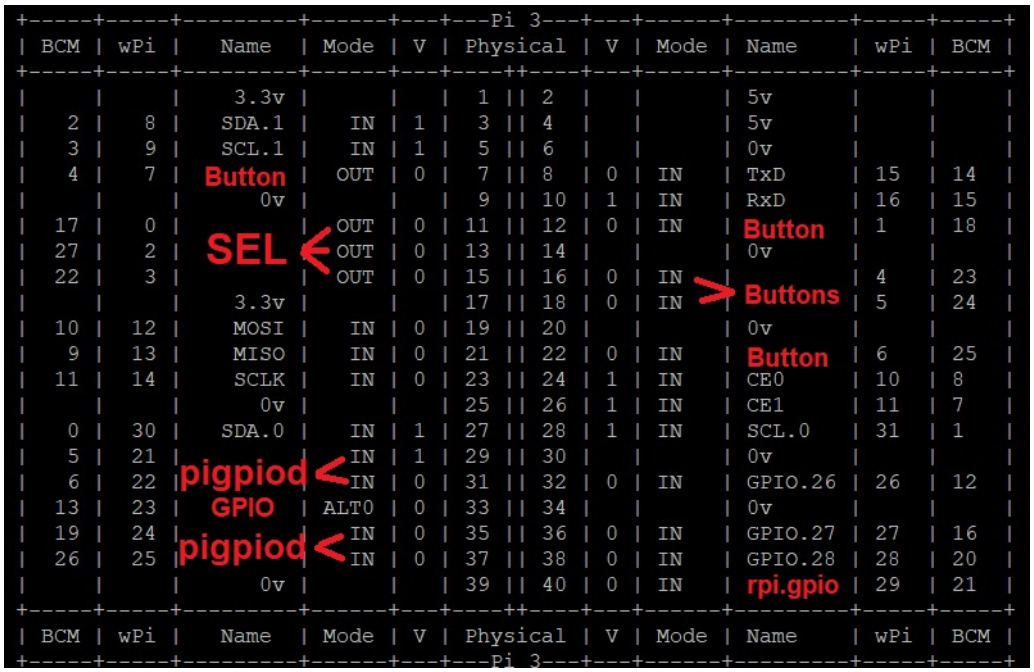
Demo s Raspberry Pi

Demo na Raspberry Pi som spravil skriptom, z ktorého spúšťam jednotlivé riešenia. Výber riešení som zariadil stlačením jedného z externých tlačídiel. Možnosti sú nasledovné. Softvérové PWM na ktorom je len jedno servo pre ukážku chvenia. Pod druhým je Wiring Pi utilita. Tretie skrýva démona pigpiod. Posledné tlačidlo nie je pripojené lebo riešenie pi-blaster pri testoch ukázalo nedostatok v návrhu a zároveň blokuje prepínanie medzi riešeniami a bol by nutný reštart. Rozdielom v zapojení



Obr. 5.6: Pripojenie demoaplikácie na demultiplexor

demultiplexora oproti zapojeniu u testov je prepojenie aktivačnej nohy na zem, lebo i tak sa zapína logickou nulou. Ďalej na nohách pomenovaných servoX boli pripojené výstupy ovládané riešením pigpio, ktoré vždy zaručilo odpojenie PWM z GPIO riešenia.



Obr. 5.7: Pripojenie demoaplikácie na Raspberry

Knižnica RPi.GPIO

Túto knižnicu používam ako základ prečo nechceme používať softvérovo časované pulzy. A zároveň ako protiargument pre jej mnohé odporučenia.

konfigurácia a implementácia k testom:

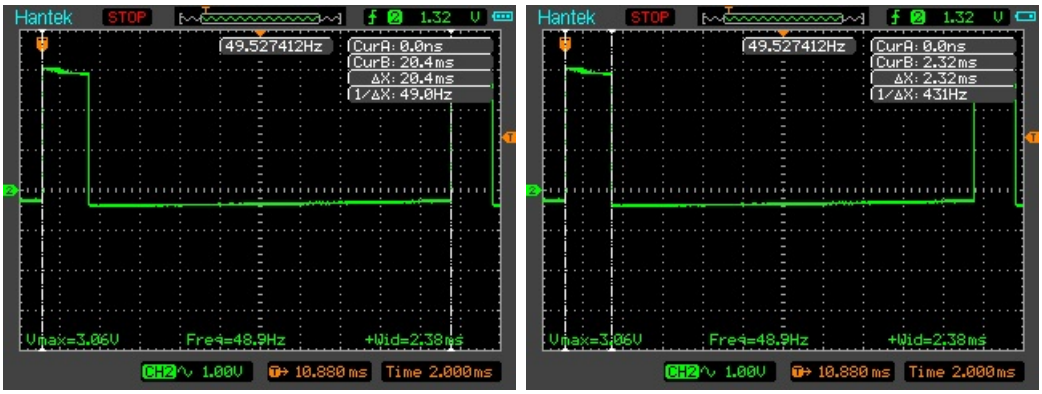
- **nastavenie frekvencie**: GPIO.PWM(servo_pin, 50)
- **zmena dĺžky**: p.ChangeDutyCycle(10), povolený rozsah pre serva 5 až 10
- **počet kanálov**: 1, jedná sa len o referenčné hodnoty
- **dĺžka pulzu**: 2ms
- **Pripojených serv**: 2
- **použité testy**
    1. Všeobecný test vlastností
    2. Všeobecný vizuálny test
    3. Všeobecný test s osciloskopom
    4. Špecifický test s osciloskopom pre Raspberry pi

Od knižnice neočakávam prívetivé výsledky lebo je časovaná softvérovo a môžu ju ovplyvniť systémové prerušenia či aj garbage collector integrovaný v Pythone.

Výsledky testov

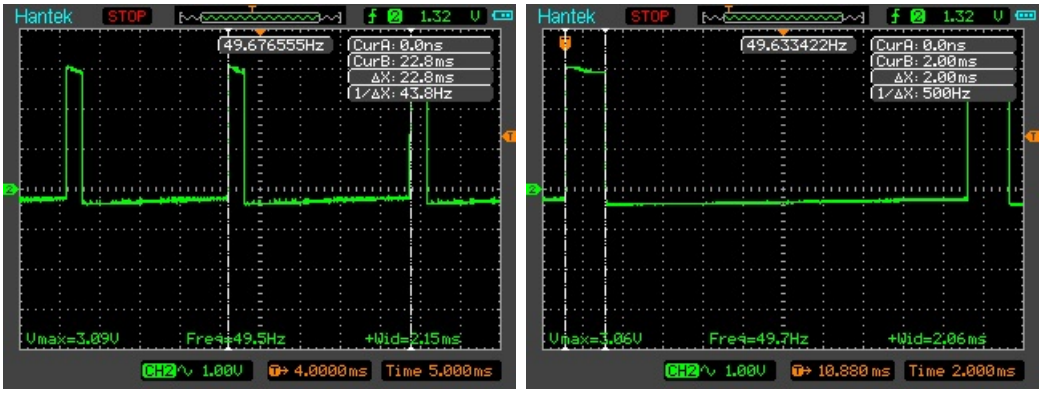
Výsledok zodpovedá predpokladu, s tým že pri jedinom serve a s Raspberry Pi v úplne kľudovom stave, to je bežiaci len operačný systém sa dané servo natáčalo

správne bez príliš očividného chvenia. Ale v okamžiku keď som sa pripojil pomocou ssh na Raspberry a dal si vypísať bežiace procesy čím vyskočilo zaťaženie iba jedného jadra na 20% sa servo začalo chvieť a meniť polohu i keď som nemenil nastavenia jeho polohy. Pri testoch na osciloskope bolo dokonca zjavné, že aj keď som s ničím nehýbal bola generovaná frekvencia nestabilná a stále sa menila. Najvyššia zaznamenaná odchýlka na perióde bola až 2.8ms a najmenšia pri prvotných testoch na menej kvalitnom osciloskope bola 0.16ms, ktorý nezaznamenal jej zmenu počas testov. Ďalším nedostatkom objaveným na kvalitnejšom osciloskope bola aj občasná zmena amplitúdy o 0.2V. Pri pozorovaní sa jedno najväčšie servo samé od seba začalo hýbať vždy od 0° po asi 90° kde malo nastavenú polohu.



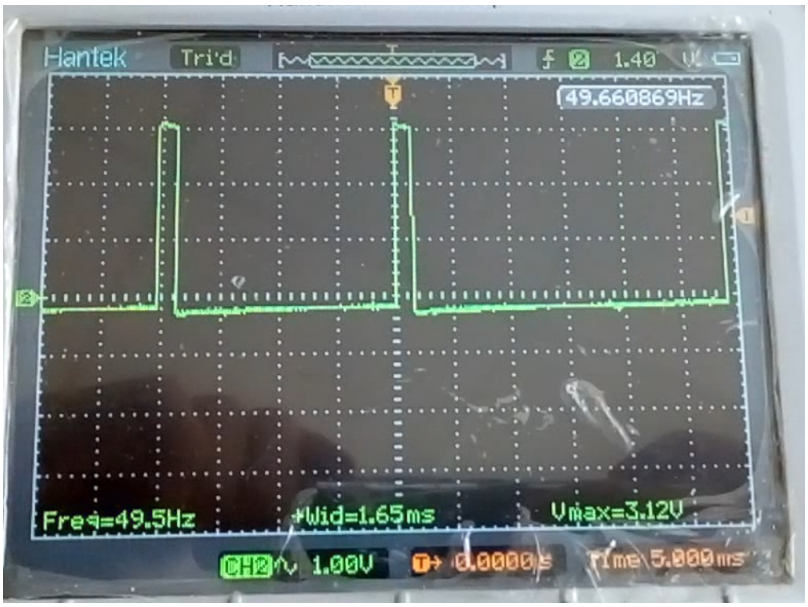
Obr. 5.8: RPi.GPIO bez záťaže na procesore

Špecifický test som na toto riešenie aplikoval aj keď jeho použiteľnosť vyvrátili už prvotné testy. Výsledkom bol veľmi nestabilný signál, ktorého priebeh som zachytil pomocou mobilného telefónu na video s ktorého som vystrihol priebeh s veľkým odskokom periódy.



Obr. 5.9: RPi.GPIO pod záťažou

Chyba periódy je vypočítaná zo zachytenej frekvencie funkciou counter. Táto hodnota je zobrazená v strede obrázkov.



Obr. 5.10: RPi.GPIO pod záťažou, vystrihnuté z videa

Rpi.GPIO	**Nastavená[ms]**	**Ustálená[ms]**	$\delta[\mu s]$	**Maximálna chyba[µs]**
Šírka	2	1.970	515.1966	1212.2

Tab. 5.1: Výsledky merania šírky: RPi.GPIO

Rpi.GPIO	**Nastavená[ms]**	**Ustálená[ms]**	**Maximálna chyba[µs]**
Perióda	20	22.8	10000

Tab. 5.2: Výsledky merania periódy: RPi.GPIO

Záver nad riešením

Riešenie **nespĺňa** základné predpoklady v dostatočnom rozsahu, aby bolo použité na ovládanie modelárskych serv.
Hlavné body prečo toto riešenie nevyhovuje:

1. Riešenie neprešlo vizuálnym testom už pri pripojení druhého serva.
2. príliš veľká smerodajná odchýlka
3. Perióda kolísa o 50%

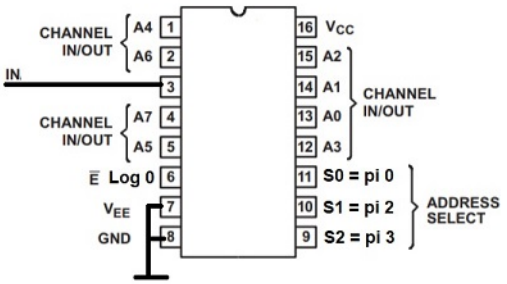
Knižnica/utilita Wiring Pi

Tento program využíva priamo natívneho PWM čipu, takže očakávam rovnakú stabilitu ako pri testoch samotnej platformy.

konfigurácia a implementácia k testom:

- **nastavenie frekvencie**: $50Hz = 19200000/pwmc/pwmr$

  - gpio mode pin PWM – prepnutie pinu 23 alebo 26 do funkcie PWM
  - gpio pwmc=48 – rozsah 2 až 4095
  - gpio pwmr=8000 – rozsah 2 až ∞[3]
  - gpio pwm-ms – použitie Mark-space algoritmu
- **zmena dĺžky**: gpio pin PWM 800, rozsah pre serva je 400 až 800
- **počet kanálov**: 2, len dva PWM kanály
- **Externý hardvér: demultiplexor(16 nôh)**[1]
  - Jeden PWM kanál 8 serv
  - ovládanie – 3 pini nutné pre selekt signál
  - prepínanie medzi servami nasledovne: gpio pin PWM 0 následne nový selekt
- **krok**: 2,5µs , iná konfigurácia dovoľuje aj menší krok
- **Pripojených serv**: 4
- **dĺžka pulzu**: 1,5 a 2ms
- **použité testy**
  1. Všeobecný test vlastností
  2. Všeobecný vizuálny test
  3. Všeobecný test s osciloskopom
  4. Špecifický test s osciloskopom pre Raspberry pi
  5. Cielené testy so zvukom



INPUT STATES				ON CHANNEL
ENABLE	S_2	S_1	S_0	
L	L	L	L	A0
L	L	L	H	A1
L	L	H	L	A2
L	L	H	H	A3
L	H	L	L	A4
L	H	L	H	A5
L	H	H	L	A6
L	H	H	H	A7
H	X	X	X	None

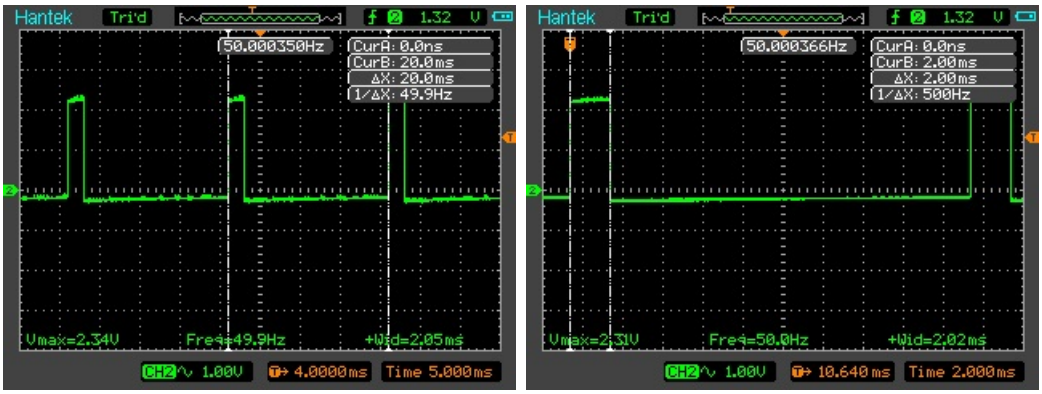
Obr. 5.11: Pripojenie demultiplexora k Raspberry Pi

Z dôvodu PWM len na dvoch pinoch používam externý hardvér. Konkrétne demultiplexor cd74hc4051 a skúmam, či spôsobí skreslenie.

[3]Horná hranica nie je špecifikovaná testoval som 10^6

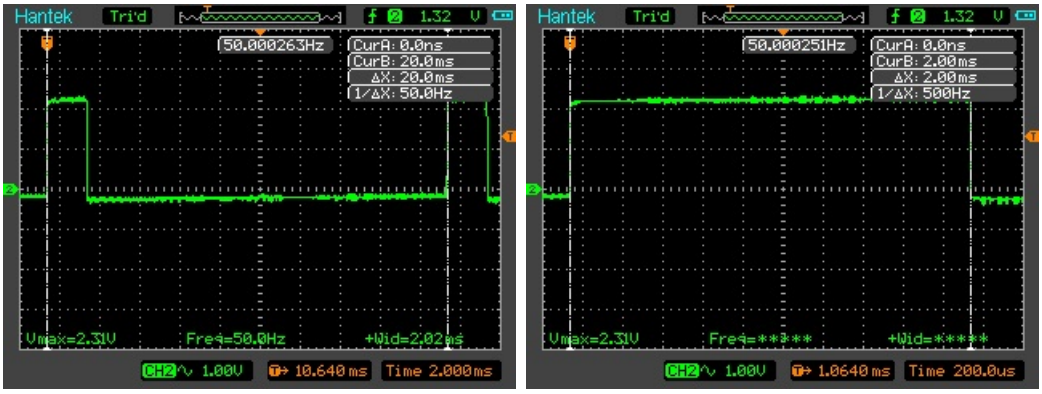
Výsledky testov

Čip demultiplexora priebeh neskresľuje oproti priamemu pripojeniu na pin Raspberry Pi. Pri vypnutí demultiplexora cez enable signál čip prepúšťa signál a skresľuje ho, preto je treba mať čip stále zapnutý. Pretože sú servá pripojené cez demultiplexor tak je signál posielaný vždy len na jediné servo, tým pádom ostatné servá majú väčšiu vôľu k pohybu pri záťaži. Výsledky zobrazené na osciloskope zodpovedajú predpokladu hardvérového PWM.



Obr. 5.12: Wiring Pi utilita bez záťaže na procesore

Hromadné testy odhalili správanie utility pri viacnásobnom prepínaní módov. Pri prepnutý pinu na PWM keď sa už móde PWM nachádza spôsobí zmenu na deličkách a prepne PWM na výpočet pomocou prvého algoritmu.



Obr. 5.13: Wiring Pi utilita so záťažou na procesore

Chyba periódy je vypočítaná zo zachytenej frekvencie funkciou counter. Táto hodnota je zobrazená v strede obrázkov. Chyba v dĺžke signálu je získaná z osciloskopu za pomoci trigeru a analýzy csv súborov z meraní. Veľkým nedostatkom je, že ak je na serve väčšia váha a nedostáva signál tak závažie môže zmeniť polohu serva,

Wiring Pi	Nastavená[ms]	Ustálená[ms]	$\delta[\mu s]$	Maximálna chyba[μs]
Šírka	2	2	0.074927	0,2

Tab. 5.3: Výsledky merania šírky: Wiring Pi

Wiring Pi	Nastavená[ms]	Ustálená[ms]	Maximálna chyba[µs]
Perióda	20	20	0.015

Tab. 5.4: Výsledky merania periódy: Wiring Pi

čo sa môže stať pri použitý demultiplexora a preto by som doporučil toto riešenie na zariadenia kde váha nie je priamo na serve alebo predimenzovať dané servo.

Záver nad riešením

Riešenie **čiastočne spĺňa** základné predpoklady v dostatočnom rozsahu, aby bolo použité na ovládanie modelárskych serv.
Hlavné body prečo toto riešenie nevyhovuje:

1. Nedostatočný počet kanálov na pripojenie serv, dá sa obísť za pomoci demultiplexora(28 czk)
2. Strata ťahu pri využití externého čipu

Body prečo toto riešenie vyhovuje:

1. Negatívny bod o ťahu sa dá ignorovať pri použití digitálnych serv
2. Priblíženie k ideálnemu signálu bez dodatočnej kalibrácie
3. Vysoká stabilita signálu šírky aj periódy

Knižnica piGPIO

Riešenie pomocou tejto knižnice a jazyku Python spočíva v zapnutí démona a zavolania funkcie na nastavenie šírky pulzu. V mojom prípade ešte v programe robím umelé spomalenie pre pohyb z dôvodu ilúzie plynulého pohybu. A to menením šírky po malých krokoch, po každom kroku nasleduje krátke aktívne čakanie. Krok a čakanie opakujem kým nedosiahnem požadovanej šírky.

konfigurácia a implementácia k testom:

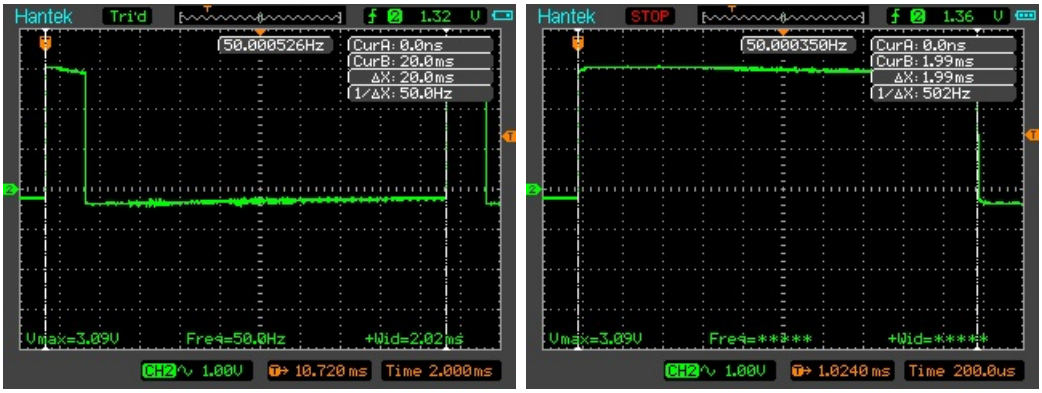
- **nastavenie frekvencie**: set_servo_pulsewidth má prednastavenú periódu 20ms
- **zmena dĺžky**: set_servo_pulsewidth(pin,1500), rozsah 1000 až 2000
- **počet kanálov**: 26, všetky GPIO pini
- **Pripojených serv**: 4
- **krok**: 2µs
- **dĺžka pulzu**: 2ms

- **použité testy**
    1. Všeobecný test vlastností
    2. Všeobecný vizuálny test
    3. Všeobecný test s osciloskopom
    4. Špecifický test s osciloskopom pre Raspberry pi
    5. Cielené testy so zvukom
    6. Sledovanie systémových zdrojov počas predošlých testov

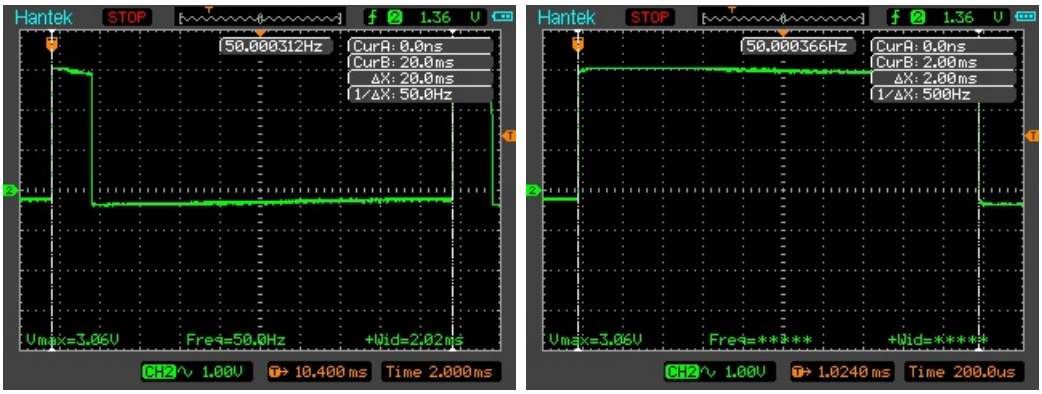
Tento program využíva priamo DMA pre generáciu takže očakávam generáciu stabilných pulzov.

Výsledky testov

Testy bez záťaže boli robené iba ako referencia k porovnaniu k záťažovým testom.

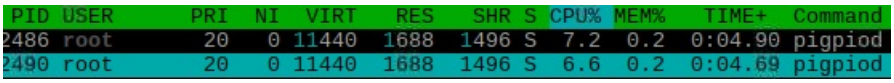


Obr. 5.14: piGPIO bez záťaže na procesore



Obr. 5.15: piGPIO so záťažou na procesore

Počas testov som zaznamenával zaťaženie systémových zdrojov, ktorých výsledok je zachytený obrázkom 5.16. Pri stress testoch sa táto záťaž tiež nemenila a ostávala v okruhu 7% pre jednu inštanciu a celkovo to robilo 14% zaťaženie na CPU. Špe-



Obr. 5.16: piGPIO systémové zdroje

cifický test spôsobil len dlhšiu dobu prvotného zapnutia a vypnutia. Táto doba je pochopiteľná lebo je treba spustiť démona/službu. Záťaž na procesore nespôsobila žiadnu nestabilitu detekovatelnú servom. Najväčšia odchýlka na perióde bola 0,21µs. Chyba periódy je vypočítaná zo zachytenej frekvencie funkciou counter. Táto hodnota je zobrazená v strede obrázkov. Chyba v dĺžke signálu je získaná z osciloskopu za pomoci trigeru a analýzy csv súborov z meraní.

piGPIO	**Nastavená[ms]**	**Ustálená[ms]**	δ[**µ**s]	**Maximálna chyba[µ**s]
Šírka	2	1.9996	0.102597835	0.4

Tab. 5.5: Výsledky merania šírky: piGPIO

piGPIO	**Nastavená[ms]**	**Ustálená[ms]**	**Maximálna chyba[µs]**
Perióda	20	20	0.21

Tab. 5.6: Výsledky merania periódy: piGPIO

Záver nad riešením

Riešenie **spĺňa** základné predpoklady v dostatočnom rozsahu, aby bolo použité na ovládanie modelárskych serv.
Body prečo toto riešenie vyhovuje:

1. Podpora generovania výstupu na všetky GPIO kontakty platformy
2. Sebestačné
3. Dostatočná stabilita signálu šírky aj periódy

Negatíva riešenia:

1. Záťaž na procesor z obslužného démona/služby

Projekt piblaster

Pre tento projekt som nevytváral žiadne zdrojové súbory. Dôvodom je nepríjemná integrácia, nutný reštart systému po jeho použití a základ jadra zdielaný s pigpio z pôvodného servoblaster. Všetky testy som robil za pomoci terminálu. Pre použitie pre modelárske serva je treba zmeniť základnú konfiguráciu. Prvé je treba si prepísať známe pini, pretože ich číslovanie sa zmenilo medzi jednotlivými verziami Raspberry.Zmenu je treba vykonať v „pi-blaste.c“ prepísaním BCM čísla contaktu do štruktúry „known_pins[]“. Nasledovne treba zmeniť „CYCLE_TIME_US“ na 20000 a „SAMPLE_US“ na 20.

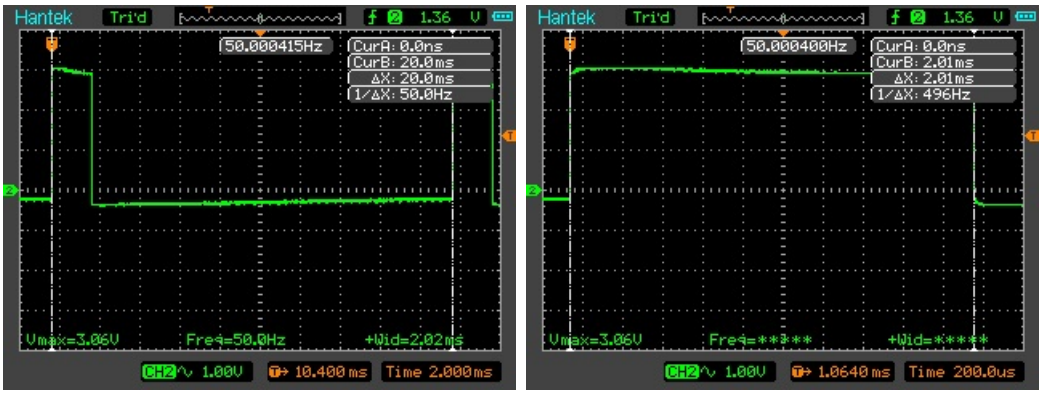
konfigurácia a implementácia k testom:

- **nastavenie frekvencie**: Zmenením parametrov pi-blaster.c
- **zmena dĺžky**: echo BCM=duty > /dev/pi-blaster, rozsah 0.050 až 0.1
- **počet kanálov**: 17, všetky GPIO pini
- **Pripojených serv**: 4
- **krok**: 2µs, minimum doporučené autorom projektu servoblaster z ktorého piblaster vychádza
- **dĺžka pulzu**: 2ms
- **použité testy**
    1. Všeobecný test vlastností
    2. Všeobecný vizuálny test
    3. Všeobecný test s osciloskopom
    4. Špecifický test s osciloskopom pre Raspberry pi
    5. Cielené testy so zvukom
    6. Sledovanie systémových zdrojov počas predošlých testov

Podobne ako pigpio je použité DMA, tak predpokladám stabilné pulzy. Do testov pridávam test opakovaný test chovania pri použití periférie pre zvuk, z dôvodu kontroly kolízie obsadenia rovnakého PWM kanálu.

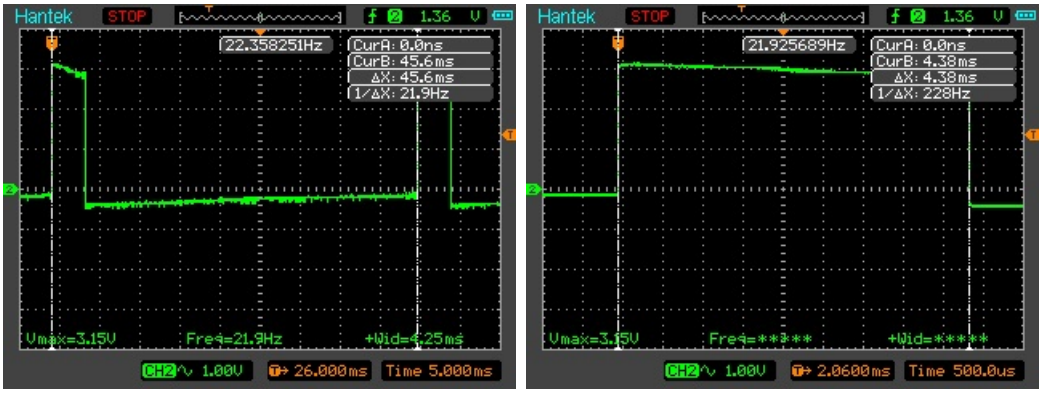
Výsledky testov

Testy na osciloskope dopadli rovnako ako aj pre iné hardvérové riešenia a to ostré hrany so správnou frekvenciou a šírkou. Testy bez záťaže boli robené iba ako referencia k porovnaniu k záťažovým testom. Špecifický test pod záťažou potvrdil ďalší nedostatok spojený a prebraný od projektu servoblaster. Tento nedostatok je neošetrenie prístupu k prvému PWM kanálu pre generáciu analógového zvuku. Tento kanál je použitý projektom na časovanie DMA. Pri teste sa toto prejavilo ako skok frekvencie k 20Hz. Táto chyba sa objaví vždy ak sa pripojí akýkoľvek jack i keď sa nič neprehráva alebo ak sa dá prehrať súbor zo zvukom. Tomuto chovaniu sa dá vyhnúť zmenou konfigurácie systému, aby predvolene používal digitálny výstup pre

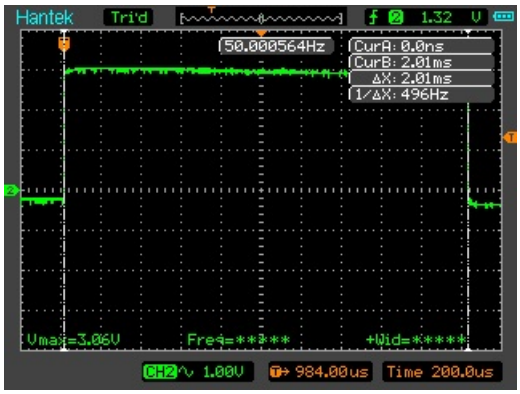


Obr. 5.17: pi-blaster bez záťaže na procesore

HDMI namiesto analógu(predvolený), čo potvdilo opakovanie druhého testu, ktorý obsahoval zvuk[4].



Obr. 5.18: pi-blaster so záťažou na procesore

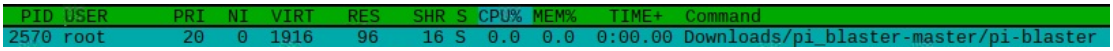


Obr. 5.19: pi-blaster so záťažou na procesore s HDMI výstupom predvoleným[5]

[4]žiaden zvuk nebol prehrávaný

[5]Graf je zhodný s testom bez audia

Počas testov som zaznamenával zaťaženie systémových zdrojov, ktorých výsledok je zachytený obrázkom 5.20. Pri stress testoch sa táto záťaž tiež nemenila. Zaujímavosťou je, že tento démon/služba nedostal žiadny čas procesoru.



Obr. 5.20: pi-blaster systémové zdroje

piblaster	**Nastavená[ms]**	**Ustálená[ms]**	δ[**µ**s]	**Maximálna chyba[µ**s]
Šírka	2	1.980	0.588982669	2.2

Tab. 5.7: Výsledky merania šírky: piblaster

piblaster	**Nastavená[ms]**	**Ustálená[ms]**	**Maximálna chyba[µs]**
Perióda	20	20	0.21

Tab. 5.8: Výsledky merania periódy: piblaster

Chyba periódy je vypočítaná zo zachytenej frekvencie funkciou counter. Táto hodnota je zobrazená v strede obrázkov. Chyba v dĺžke signálu je získaná z osciloskopu za pomoci trigeru a analýzy csv súborov z meraní.

Záver nad riešením

Riešenie **spĺňa** základné predpoklady v dostatočnom rozsahu, aby bolo použité na ovládanie modelárskych serv.
Body prečo toto riešenie vyhovuje:

1. Podpora generovania výstupu na všetky GPIO kontakty platformy
2. Sebestačné
3. Dostatočná stabilita signálu šírky aj periódy
4. Záťaž na procesor z obslužného démona/služby

Negatíva riešenia:

1. Náročná integrácia
2. Blokácia zvukového kanálu a možný konflikt
3. Rezervácia zdrojov, väčšinou nutní reštart na uvoľnenie

5.3 Riešenia platformy FITKIT

Bola použitá jeho verzia 2.0 s kombináciou Windows 7 predpripravenou FIT VUT. Pre túto platformu som sa rozhodol testovať po jednom riešení pre FGPA a MCU bez ich kombinovania. Takto som sa rozhodol lebo je možné použiť podobný čip, ktorý nie je súčasťou FITKITu samostatne.

Demo s FITKITom

Zapojenie káblov je popísané v tabuľke nižšie 5.9. Pre ovládanie som zvolil vstavanú klávesnicu na fitkite. V MCU používam klávesy A, B, C, D na zvolenie serva, ktorému chcem poslať signál. Klávesy 1, 2 a 3 sú krajné polohy a stred. Samotné ovládanie posunu je namapované na klávesach 4 a 6. FPGA mapuje klávesy 7, 9 pre servo 4, 4 a 6 pre servo 2 a tak má každé servo samostatné ovládanie bez nutnosti medzi nimi prepínať.

Demo fitkit						
	Zem	Napájanie	Signál			
FPGA	JP9 40	JP9 39	JP10 39	JP10 41	JP10 43	JP10 45
MCU	JP9 40	JP9 39	JP9 7	JP915	JP9 23	JP9 37

Tab. 5.9: FITKIT: Aktívne pini konektorov pre demo

FPGA

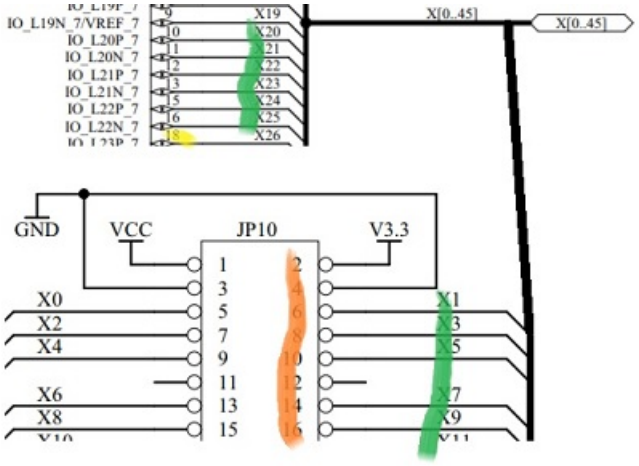
FPGA riešenie by malo byť schopné generácie pulzov bez záchvevov pomocou vytvorených logických obvodov v jeho vnútri. Takže by nemalo byť ovplivnené ani inou komunikáciou ako MCU s prerušeniami.

konfigurácia a implementácia k testom:

- **nastavenie frekvencie**: pomocou 19bit čítača na zníženie základnej 7.3728 Mhz frekvencie
- **zmena dĺžky**: Porovnanie registra s čítačom, rozsah x"1ce7äž x"39ce", t.j. 1-2ms
- **počet kanálov**:4, možné všetky pini JP10[6]
- **Pripojených serv**: 4
- **kanály s posunom**
- **krok**: 18µs, zodpovedá môjmu kroku 0x85, krok ide znížiť pod 1µs
- **dĺžka pulzu**: 1-2ms

[6]teoreticky, jedna z možných nadstavieb na túto prácu

- **použité testy**
  1. Všeobecný test vlastností
  2. Všeobecný vizuálny test
  3. Všeobecný test s osciloskopom

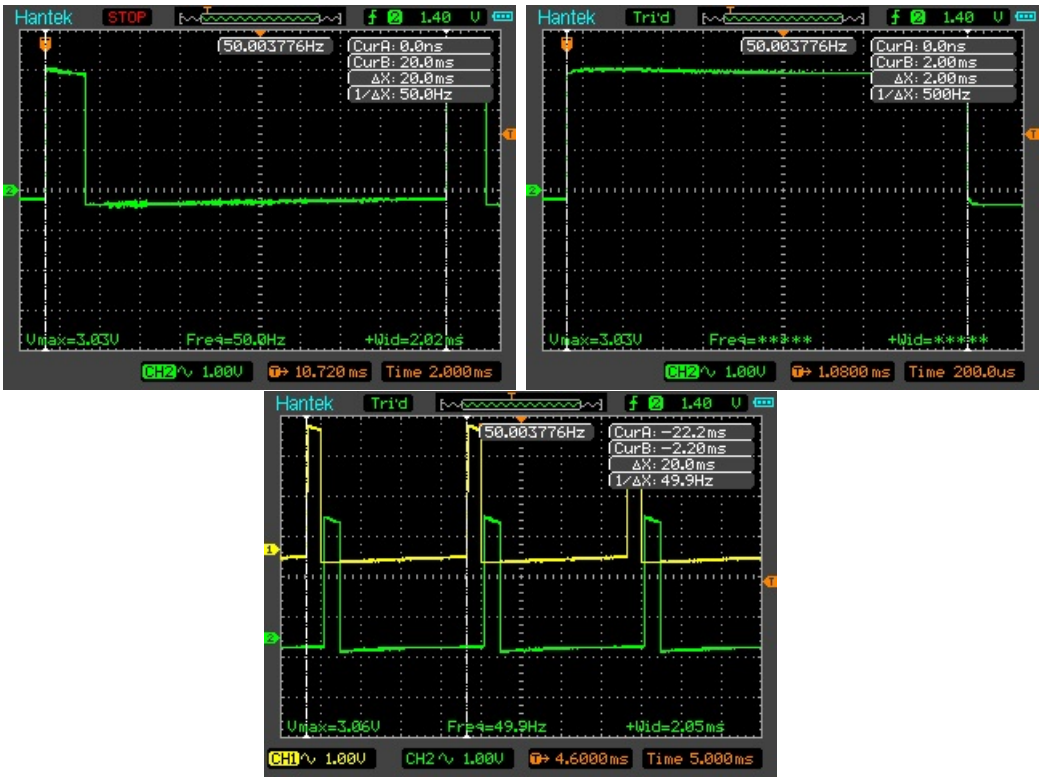


Obr. 5.21: Vizualizácia mapovania FPGA pinov na konektor JP10

Vizualizácia predstavuje mapovanie zbernice JP10, jej pinov(oranžová) na prepoje(zelená) na PXX čísla(žltá) použité v ucf súbore. Od riešenia očakávam ostré pulzy, lebo kódom sa vytvára zoskupenie logických obvodov, rovnako ako keby sa pospájali externé čipy medzi sebou.

Výsledky testov

Testy na osciloskope potvrdili stabilitu signálov. Najväčšia odchýlka na perióde bola 0,15µs. Chyba v dĺžke signálu je získaná z osciloskopu za pomoci trigeru a analýzy csv súborov z meraní.



Obr. 5.22: FITKIT FPGA prvý a druhý kanál osciloskopu

FPGA	**Nastavená[ms]**	**Ustálená[ms]**	$\delta[\mu s]$	**Maximálna chyba[µs]**
Šírka	2	1.9932	0.095513387	0.2

Tab. 5.10: Výsledky merania šírky: FPGA

FPGA	**Nastavená[ms]**	**Ustálená[ms]**	**Maximálna chyba[µs]**
Perióda	20	20	0.15

Tab. 5.11: Výsledky merania periódy: FPGA

Testy ktoré by prekazovali použiteľnosť tohto FPGA riešenia, by mali skúsiť zaťaženie systémových zdrojov. Iný typ testu by mohol skúšať maximálny počet pripojiteľných serv, bez nutnosti externej batérie alebo DC adaptéra a či má FITKIT dosť pamäte pre všetky signály serv.

Záver nad riešením

Riešenie **spĺňa** základné predpoklady v dostatočnom rozsahu, aby bolo použité na ovládanie modelárskych serv.
Body prečo toto riešenie vyhovuje:

1. Podpora generovania výstupu na všetky kontakty platformy
2. Sebestačné
3. Dostatočná stabilita signálu šírky aj periódy

Negatíva riešenia:

1. Náročnejšia kalibrácia – Frekvencia FITKITu z kompilačného logu, po prepočte na požadovanú spôsobovala odchýlku v rádoch jednotiek µs

MCU

Toto riešenie trpí nielen pre jeho ovplyvniteľnosť inou záťažou, ale aj ako pri rozširovaní pomocou demultiplexora pre riešenie Wiring Pi z platformy Raspberry je možné mať aktívny len jeden kanál.

konfigurácia a implementácia k testom:

- **nastavenie frekvencie**: ticks = 32768/50/SAMPL
- **zmena dĺžky**: CCR0 += ticks*duty_cycle, manipulujem len s duty
- **počet kanálov**: 4 pini na JP9
- **Pripojených serv**: 4
- **krok**: 0,003ms, čo je možné zmeniť premenou SAMPL
- **dĺžka pulzu**: 2ms
- **použité testy**
    1. Všeobecný test vlastností
    2. Všeobecný vizuálny test
    3. Všeobecný test s osciloskopom

Od MCU očakávam možné záchvevy na signále, z dôvodu použitia prerušení na časovanie a možného príchodu prerušenia s vyššou prioritou.

Výsledky testov

Osciloskop potvrdil predpoklad nestability riešenia, za pomoci prerušení. Dôvodom tohto tvrdenia je chyba periódy s maximom 0,163ms a zároveň jej neustály pohyb približne o 0,08ms. Chyba v dĺžke signálu je získaná z osciloskopu za pomoci trigeru a analýzy csv súborov z meraní.

MCU	**Nastavená[ms]**	**Ustálená[ms]**	δ[**µ**s]	**Maximálna chyba**[**µ**s]
Šírka	2	1.9981	0.194063955	0.6

Tab. 5.12: Výsledky merania šírky: MCU

Medzi ďalšie testy k MCU by bolo možné zaradiť: test simulujúci komunikáciu a test simulujúci komunikáciu s inou výpočetnou záťažou na procesore.

MCU	Nastavená[ms]	Ustálená[ms]	Maximálna chyba[µs]
Perióda	20	19.83	163

Tab. 5.13: Výsledky merania periódy: MCU

Záver nad riešením

Riešenie **nespĺňa** základné predpoklady v dostatočnom rozsahu, aby bolo použité na ovládanie modelárskych serv.
Body prečo toto riešenie nevyhovuje:

1. Zložité rozšírenie nad 4 výstupy
2. Stabilita signálu šírky na hrane – Pri prenose údajov cez USB sa servo viditeľne chvelo
3. Strata ťahu na servách

Pozitíva riešenia:

1. Ťah vyrieši použitie digitálnych serv
2. Riešenie sa dá použiť pre účely podobné môjmu demu

5.4 Riešenia platformy Personálny počítač

Použitá bola doska Pololu mikro maestro 6. Miesto externej batérie som použil na napájanie z Raspberry alebo fitkitu. Raspberry bolo použité pri prvotných testoch.

konfigurácia a implementácia k testom:

- **nastavenie frekvencie**:
  - za pomoci grafického rozhrania
  - skript – frekvencia je prednastavená
- **zmena dĺžky**:
  - za pomoci grafického rozhrania
  - skript – 8000 [servo_num] servo, rozsah 4000 až 8000
- **počet kanálov**: 6
- **Pripojených serv**: 5
- **krok**: 0,25µs
- **dĺžka pulzu**: 1-2ms
- **nastavenie rýchlosti**: 30 [servo] speed
- **použité testy**
  1. Všeobecný test vlastností
  2. Všeobecný vizuálny test
  3. Všeobecný test s osciloskopom

4. Test GUI vs Script
5. test napájacieho obvodu

Demo s PC

Pololu micro maestro bolo pripojené iba prinamo na signál k servam a samotné napájanie zabezpečoval FITKIT. Pre demo som použil grafické rozhranie od výrobcu.

Od dosky očakávam dobré výsledky, lebo bola na tento účel navrhnutá, má vlastný časovač aj mikrokontrolér.

Výsledky testov

Po pripojený k osciloskopu bol signál s ostrými hranami aj požadovaných parametrov. To znamená perióda 20ms aj keď s odchýlkou 21µs a šírkou signálu 2ms.

Počas testov bola amplitúda 4,8V, čím sa ukázalo že doska berie napájanie bez toho aby ho štandardizovala na 3,3V, modulovaný signál totiž vzniká z 5V napájania z raspberri pi.

Ďalej sa prejavilo že čip používa posunu o 3.34ms medzi jednotlivými kanálmi na ušetrenie energie a procesorového času.

Obr. 5.23: Micro maestro prvý a druhý kanál osciloskopu

PC	Nastavená[ms]	Ustálená[ms]	δ[µs]	Maximálna chyba[µs]
Šírka	2	1.9994	0.063060354	0.2

Tab. 5.14: Výsledky merania šírky: PC

PC	Nastavená[ms]	Ustálená[ms]	Maximálna chyba[µs]
Perióda	20	19.97	21.5

Tab. 5.15: Výsledky merania periódy: PC

Chyba periódy je vypočítaná zo zachytenej frekvencie funkciou counter. Táto hodnota je zobrazená v strede obrazovky. Chyba v dĺžke signálu je získaná z osciloskopu za pomoci trigeru a analýzy csv súborov z meraní.

Záver nad riešením

Riešenie **spĺňa** základné predpoklady v dostatočnom rozsahu, aby bolo použité na ovládanie modelárskych serv.
Body prečo toto riešenie vyhovuje:

1. Dostatočná stabilita signálu šírky aj periódy
2. Komerčné riešenie cenovo porovnateľné s Raspberry Pi
3. Sebestačné
4. Dokumentácia
5. Dobrá integrácia

Negatíva:

1. Rozšíriteľnosť výstupov

Zhrunutie testov

Pre platformu FITKIT som skúmal a popísal po jednom riešení pre FPGA a MCU. Záverom je, že MCU s prerušeniami nie je vhodné riešenie ak je nutné komunikovať s okolím, alebo ak je na procesore iná záťaž tak daný čip nemá dostatočný výkon. Preto je určite lepšie riešenie s využitím FPGA nielen, že je stabilnejšie a poskytuje viac možných samostatných serv, ale je aj efektívnejšie z pohľadu napájania. Posledná zmienka k FITKITu je, že na MCU je možné použiť DMA kanály pre generáciu pulzov, ale toto riešenie som bližšie neskúmal.

Platforma Arduino má najviac fór a článkov na ovládanie serv za jeho pomoci. Túto platformu som, netestoval lebo je už dobre preskúmaná a zdokumentovaná. Samotné výstupy PWM kanálov Arduina je šesť na troch rôznych časovačoch, ktoré

neposkytujú frekvenciu používanú modelárskymi servami. Z pomedzi riešení použiteľných sa javili riešenia pomocou konverzie čítaného napätia idúceho cez potenciometer a jeho prepisu na požadovaný rozsah dobrou voľbou. Ďalšie riešenia boli robené za pomoci knižníc servo.h a PWM.h. Samotné výstupy PWM kanálov arduina aj to šesť kanálov na troch rôznych časovačoch tak neposkytujú frekvenciu používanú modelárskymi servami.

Na platforme personálneho počítača je vždy potrebná externá doska, ktorá poskytuje časovanie. Veľmi dobrou možnosťou je doska Micro maestro od Pololu, kde najmenej poskytuje šesť kanálov. Pre komunikáciu s personálnym počítačom je nutné si nainštalovať ovládač. Odporúčam najskôr nainštalovať ovládač a až nasledovne pripojiť dosku, vyhnete sa možnému problému so všeobecným USB ovládačom a strate informácie o COM kanále. Plus je treba spraviť prvotnú konfiguráciu pre správnu funkciu skriptov.

	Channels	**Step**[µs]	δ **width**[µs]
Mikro maestro	6	0.25	0.063
Wiring Pi	2	<2.5	0.075
piGPIO	17+	2	0.103
pi-blaster	17+	2	0.589
Rpi.GPIO	17+	-	515.197
FPGA	6+	<18	0.096
MCU wtih interupts	4	3	0.194
Arduino	12	5.5	0.157

Tab. 5.16: Súhrn z testov platforiem

Platforma Raspberry Pi bola podrobená najviac testom a skúmal som na nej aj najväčšie množstvo riešení. Okrem riešení som testoval aj samotnú stabilitu PWM signálov pri frekvencii 5kHz kde som predpokladal že sa záchvevy zobrazia skôr než na testovaných riešeniach. Najlepšie z testovaných dopadlo riešenie pigpio, ktoré udržovalo stabilný signál pod záťažou a poskytuje PWM na všetkých GPIO pinoch za pomoci DMA. Nasledujú riešenia: pi-blaster ak je zabezpečené, že sa neprehráva zvuk a je odpojený jack, tak poskytuje veľmi dobré výsledky. Pi-blaster poskytuje PWM na všetky GPIO pini za pomoci DMA. Obe riešenia pi-blaster aj pigpiod vychádzajú z riešenia servoblaster z ktorého preberajú základ časovania DMA. U utility Wiring Pi je vhodné spomenúť že má len dva PWM kanály čiže len dva pini, čo sa týka signálu tak je to PWM priamo z čipu ktorého stabilitu som testoval aj na vyšších frekvenciách a vďaka tomu dosahovalo najlepšiu stabilitu periódy. Posledné hardvérové riešenie, ktoré nebolo kompatibilné s mojou verziou je RPIO.GPIO a preto

bolo vypustené z testov. Testy som robil aj na softvérovom PWM Rpi.GPIO pre referenciu chovania.

Pri testoch platformy sa prejavila chyba v napájacom okruhu cez USB.

Testy som robil väčšinou v troch fázach. Prvé len na školou poskytnutým osciloskopom, kde som testoval len základnú frekvenciu. V druhej fáze boli testy na kvalitnejšom osciloskope na prázdno a pod syntetickou záťažou. Pod záťažou sa Raspberry prehrialo a vyplo, čo dovolilo odhaliť spomínanú chybu napájania. Posledné testy boli pod mnou vyrobenou záťažou a na osciloskope. Posledná séria testov odhalila nové nedostatky niektorých riešení.

Pre budúce použitie by som odporučil kombináciu Raspberry a polulo maestro spojených uartom alebo Raspberry s externým DMA/MCU a využívať Raspberry na riadiaci program a externý čip pre generáciu.

6 Záver

Cieľom tejto bakalárske práce bolo skúmanie spôsobu ovládania modelárskych serv z rôznych počítačových platforiem. Tento cieľ bol splnený, zároveň boli splnené čiastkové podciele.

Prvý podcieľ bolo preštudovanie spôsobu riadenia modelárskych serv a možností ich ovládania počítačovým systémom, zhrnutie tohto preštudovania tématu sú v kapitolách 1 a 2. Návrh robotického ramena som popísal v kapitole 4 sekcii 4, Tento návrh dreveného ramena bol určený ako základ k testom na jednotlivých platformách. Ďalej som popísal možnosti ovládania modelárskych serv z platforiem: FITKIT, Raspberry Pi a z personálneho počítača. Tento popis sa nachádza v kapitole 3. Druhá iterácia návrhu robotického ramena bola založená na poznatkoch z dreveného návrhu a z testov jednotlivých platforiem.

Medzi zajujímavé výsledky patrí presnosť dosiahnutá riešeniami na platforme Raspberry Pi menovite: piGPIO, wiring Pi utility a pi-blaster[1], dosiahnutá Stabilita na šírke impulzov bola porovnateľná s komerčným riešením micro maestro a to o jeden záchvev viac, numericky 0.06306 a 0.074927 µs. Odchýlku µs 0.095513387 sa podarilo dosiahnuť aj na FITKITe s použitím FPGA. K ideálnej perióde 20ms bola najbližšie utilita Wiring Pi [2] a to o 0,01µs.

Tiež by som zmienil projekty, ktoré sme s majiteľom osciloskopu Miroslavom Martákom diskutovali počas meraní. Prepojenie Raspberry Pi s GPS modulom z vyzdvihnutých meteorologických balónov, z tohto modulu sa dajú získavať nielen GPS súradnice, ale napríklad aj čas z atómových hodím. Nadstavba na RC podvozok z Raspberry Pi, pololu micro maestra a kamery VEGA(alternatíva GOpro).

Nadväzujúce práce by mohli moju prácu rozšíriť, či prehĺbiť o testovanie MCU a FPGA FITKITu pod záťažou buď z výpočtu alebo komunikáciu po zbernici. Ďalej by bolo možné skúmať prepojenie serv a personálneho počítača inou doskou ako je Pololu micro maestro.Tiež by mohlo byť zaujímavé skúmanie maximálneho počtu pripojiteľných serv k týmto platformám, lebo riešenia v tejto práci mali pripojené maximálne 5 serv súčasne a udávam teoretický počet maximálneho počtu.

[1] len v istých situáciách

[2] Tiež pod menom gpio utilita

Literatúra

[1] *CDx4HC405x, CDx4HCT405x High-Speed CMOS Logic Analog Multiplexers and Demultiplexers datasheet (Rev. M)*. Texas Instruments Incorporated, Texas, 2017 [Online; navštívené 13.12.2018].
http://www.ti.com/lit/ds/symlink/cd74hc4051.pdf

[2] *Direct memory access*. Wikimedia Foundation, Inc, CA: San Francisco, /2002 [Online; navštívené 13.4.2019].
https://en.wikipedia.org/wiki/Direct_memory_access

[3] *FPGA*. Wikimedia Foundation, Inc, CA: San Francisco, 10.4.2006, [Online; navštívené 9.3.2019].
https://sk.wikipedia.org/wiki/Programovate%C4%BEn%C3%BD_logick%C3%BD_obvod

[4] *FPGA*. Wikimedia Foundation, Inc, CA: San Francisco, 10.8.2001, [Online; navštívené 24.3.2019].https://en.wikipedia.org/wiki/Field-programmable_gate_array

[5] *Impulzová šírková modulácia*. Wikimedia Foundation, Inc, CA: San Francisco, 19.5.2014, [Online; navštívené 6.2.2020].https://sk.wikipedia.org/wiki/Impulzov%C3%A1_%C5%A1%C3%ADrkov%C3%A1_modul%C3%A1cia

[6] *Jak fungují modelářská serva*. webz.cz 07.07.2015, [Online; navštívené 23.11.2018].
http://vlastikd.webz.cz/bastl/serva.htm

[7] *Logické obvody*. Wikimedia Foundation, Inc, CA: San Francisco, 10.4.2006, [Online; navštívené 20.3.2019].
https://sk.wikipedia.org/wiki/Programovate%C4%BEn%C3%BD_logick%C3%BD_obvod

[8] *MCU*. Wikimedia Foundation, Inc, CA: San francisco 25.2.2002, [Online; navštívené 6.4.2019].
https://en.wikipedia.org/wiki/Microcontroller

[9] *MCU*. Wikimedia Foundation, Inc, CA: San Francisco, 10.4..2006, [Online; navštívené 13.4.2019].
https://sk.wikipedia.org/wiki/Mikrokontrol%C3%A9r

[10] *MSP430x2xx family guide* Texas Instruments Incorporated, Texas, 2013, [Online; navštívené 9.1.2019].
https://www.ti.com/lit/ug/slau144j/slau144j.pdf

[11] *Návody – FITKIT*. Fakulta informačních technologií, VUT Brno 2006, [Online, navštívené; 27.11.2018].
http://merlin.fit.vutbr.cz/FITkit/navody.html

[12] *Osobný počítač*. Wikimedia Foundation, Inc, CA: San Francisco, 3.3.2005, [Online; navštívené 9.4.2018].
https://sk.wikipedia.org/wiki/Osobn%C3%BD_po%C4%8D%C3%ADta%C4%8D

[13] *Physical computing with Raspberry Pi*. Jisc Services Limited, Bristol, 2016, [Online; navštívené 14.3.2019].
https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/robot/cheat_sheet/

[14] *pigpio library*. abyz.me.uk , 2014, [Online; navštívené 11.1.2020].
http://abyz.me.uk/rpi/pigpio/download.html

[15] *pi-blaster*. GitHub, Inc, San Francisco, 2012, [Online; navštívené 9.1.2019].
https://github.com/sarfata/pi-blaster

[16] *Pulse-width modulation*. Wikimedia Foundation, Inc, CA: San Francisco, 6.9.2002, [Online; navštívené 28.11.2018].
https://en.wikipedia.org/wiki/Pulse-width_modulation

[17] *RPi Low-level peripherals*. eNom, Inc., WA: Bellevue, 2013, [Online; navštívené 9.10.2019].
https://elinux.org/RPi_Low-level_peripherals#C

[18] *RPi.GPIO 0.6.5*. Python Software Foundation, 24.7.2015, [Online; navštívené 19.11.2018].
https://pypi.org/project/RPi.GPIO/

[19] *SERVO MOTORS*. robotiksistem.com, 11.5.2009, [Online; navštívené 5.2.2018].
http://www.robotiksistem.com/servo_motor_types_properties.html

[20] *SERVO SIZE EXPLAINED*. Motion RC, 18.2.2018, [Online; navštívené 19.11.2018].
https://www.motionrc.com/blogs/motion-rc-blog/servo-size

[21] *Servo (radio control)*. Wikimedia Foundation Inc, CA: San Francisco , 9.11.2011, [Online; navštívené 28.11.2018].
https://en.wikipedia.org/wiki/Servo_(radio_control)

[22] *Using DMA for pulse counting on Kinetis*. Arm Limited, 2015, [Online; navštívené 3.2.2019].

https://os.mbed.com/media/uploads/GregC/an5083-using_dma_for_pulse_counting.pdf

[23] BARNA, Andrej: *Zařízení s modelářskými RC servy*. Bakalářská práce, Vysoké učení technické v Brně, Fakulta informačních technologií, Brno 2017. Vedoucí práce Zemčík Pavel.
https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=159114

[24] Chichak, Andrei: *DMA - A LITTLE HELP FROM MY FRIENDS*. EPAG Domainservices GmbH, 2017, [Online; navštívené 29.12.2018].
https://www.embedded.fm/blog/2017/2/20/an-introduction-to-dma

[25] Eglowstein, Howard: *Introduction to Servo Motors*. Science Buddies, 2013, [Online; navštívené 13.12.2018].
https://www.sciencebuddies.org/science-fair-projects/references/introduction-to-servo-motors

[26] Long, Jason: *Embedded in Embedded*. Elektor International Media BV, United Kingdom, 2018, ISBN: 978-1-907920-73-8,
http://www.ganssle.com/articles/adma.htm

[27] Malášek, Jan: *Servo control interface in detail*. Pololu Corporation, NV: Las Vegas, 2019, [Online; navštívené 17.12.2018].
URL https://www.pololu.com/blog/17/servo-control-interface-in-detail

[28] Paláček, Josef: *Umíte si dobře vybrat servo: e-book*. modelorlicko.com,15.8.2015, [Online; navštívené 11.1.2019].
http://www.modelorlicko.com/phocadownloadpap/03-rady-navody/prislusenstvi/Umite-vybrat-servo.pdf

[29] PELIKAN, DANIEL: *Serva*. pelikandaniel.com, 11.2.2011, [Online; navštívené 19.12.2018].
http://www.pelikandaniel.com/?sec=page&id=22

[30] Proaño, Juan F.: *FPGA VHDL PWM pulse width modulation waveshare development board implementation xilinx Spartan 3*. quitoart.blogspot.com, Dublin, 2017 , [Online; navštívené 29.11.2018].
http://quitoart.blogspot.com/2017/11/fpga-vhdl-pwm-pulse-width-modulation.html

[31] Salt, John: *Your Guide To RC Servos*. RC Helicopter Fun.com, 4.2018, [Online; navštívené 7.12.2018].
https://www.rchelicopterfun.com/rc-servos.html

[32] Sergio, B.: *BCM2837 ARM Peripherals*. Tucows Domains Inc, St. Miami, 2012, [Online; navštívené 3.4.2019].
https://cs140e.sergio.bz/docs/BCM2837-ARM-Peripherals.pdf

[33] Sorani, Dana: *Electrical Engineering forum*. Stack Exchange Inc, NY: New York, 2014, [Online; navštívené 19.12.2018].
https://electronics.stackexchange.com/questions/129961/how-to-get-the-pwm-frequency-and-duration-of-each-pulse

Zoznam symbolov, veličín a skratiek

DMA	Direct memory access, Priamy prístup do pamäte
USB	Universal Serial Bus
COM	Comunication por, označenie sériového portu, virtuálneho či fyzického
PWM	Pulse with modulation
GPIO	General purlose input/output, Vstupy a výstupy pre všeobecné použitie
RC	Radio control
RAM	Random access memory
CPU	central processing unit
Soc	System on a chip
PCM	Pulse-code Modulation
BCM	Broadcom pin number

A Obsah pamäťového média

Na disku sa nachádza tehto obsah:

- Text práce, spolu s LaTeXtextami
- Zdrojové súbori programov k ovládaniu dema
- vyhotovený záznam, jeho 1min verziu
- súbory modelu,
  - jednotlivé súčiastky
  - súbory pre 3D tlač
  - vizualizáciu modelu