UNIVERZITA PALACKÉHO V OLOMOUCI

PEDAGOGICKÁ FAKULTA

Katedra technické a informační výchovy

DIPLOMOVÁ PRÁCE

Bc. Tomáš Vodehnal

Konstrukce edukačního robota s enkodérem na bázi Arduino Nano

Olomouc 2022

vedoucí práce: Mgr. Radim Děrda

PROHLÁŠENÍ STUDENTA:

Prohlašuji, že jsem závěrečnou práci vypracoval(a) samostatně. Veškerou literaturu a další zdroje, z nichž jsem při zpracování čerpal(a), uvádím v seznamu použité literatury a zdrojů.

V Hejnicích dne 14.4.2022

vlastnoruční podpis studenta

……

Bc. Tomáš Vodehnal

PODĚKOVÁNÍ:

Děkuji Mgr. Radimu Děrdovi za vedení diplomové práce, poskytování rad a materiálů pro tvorbu. Dále děkuji svému spolužákovi Jiřímu Rudolfovi za pomoc s 3D tiskem a v neposlední řadě také své manželce, která měla pochopení pro spousty hodin mého studia.

ANOTACE:

Práce se zabývá tématem nové informatiky a její praktické realizaci ve výuce na druhém stupni základní školy. Praktická realizace spočívá v pracovních listech pro žáky a metodických listech pro učitele, které provedou vyučujícího i žáka konstrukcí robota od návrhu podvozku až po jeho naprogramování a zprovoznění.

ANNOTATION:

The thesis deals with the topic of new informatics and its practical implementation in teaching at the second stage of primary school. The practical implementation consists of worksheets for students and methodological sheets for teachers, which guide the teacher and the student through the construction of the robot from the design of the chassis to its programming and commissioning.

KLÍČOVÁ SLOVA:

robotika, nová informatika, 3D tisk, Arduino, algoritmizace, programování, TinkerCad

KEYWORDS:

robotics, new informatics, 3D print, Arduino, algorithmization, programming, TinkerCad

Obsah

Úvod

Zvláště v posledních letech robotiky značně přibývá, a to nejen v průmyslovém odvětví. Roboti se dostávají již do každodenního života běžných občanů. Jako příklad můžeme uvést robotické vysavače, drony, automatizované domácnosti a podobně. Na tento rozmach reaguje i ministerstvo školství a v roce 2021 vydalo revidované rámcové vzdělávací programy, které přidávají značné množství prostoru informatice, v rámci které je i samotná robotika.

Cílem této práce bylo vytvoření robota pro edukační účely spolu s enkodérem. Práce se týká nejenom samotné konstrukce, ale také vytvoření pracovních listů pro žáky a metodických listů pro učitele. Práce tak reaguje na aktuální potřebu učitelů a ředitelů škol, kteří se s novým pojetím informatiky setkávají. Na trhu jistě existuje mnoho různých řešení, pomocí kterých mohou školy nové rámcové vzdělávací programy naplnit, avšak ne vždy se jedná o cenově dostupné produkty. Rovněž také ovládání samotného robota programem je jistě pro žáky poučné, avšak chybí zde rozvíjení praktické zručnosti. Konstrukce takového robota od prvního šroubku až po poslední program eliminuje obě uvedené nevýhody. Cena se počítá pouze za skutečně použité součástky a o rozvíjení praktické zručnosti a konstruktivního myšlení při práci určitě není nouze.

1 Teoretická část

1.1 Robotika jako vědní disciplína

Robotika jako obor, jak už z názvu vyplývá, se zabývá konstrukcí a studiem robotů nebo jiných podobných zařízení. Pojem robot zavedli bratři Čapkové a je znám od roku 1920 z divadelní hry R.U.R. Robot chápeme jako stroj, který získává informace o prostředí a nějakým způsobem na ně reaguje, čímž toto prostředí mění. [1] [19]

Robotika jako taková zahrnuje mnoho dalších disciplín. Na prvním místě mechaniku a elektroniku, dále také teorie řízení, měřící techniky a umělé inteligence. Velmi výrazně je robotika propojena s automatizací, která se na trhu objevuje již od 60. let 20. století. [1]

Robotické automaty dnes nalezneme téměř v každém větším nebo i menším podniku, který se zabývá výrobou prakticky čehokoli. Jedná se o statické stroje, které automaticky provádějí nějaký rutinní úkol. Do popředí výzkumu se v posledních letech dostávají mobilní roboti a mnoho firem, ale i domácností tyto roboty využívá. Jako jedním z prvních byl Amazon v okamžiku, kdy automatizoval své skladové prostory pomocí robotických vozíků. Ale i v českých podmínkách máme místa, která se mohou pyšnit automatizovanými robotickými jednotkami. Jedním takovým místem je překvapivě pražská nemocnice v Motole. Za poslední roky se do domácností prolomily robotické vysavače. [1]

Obrázek 1 - želvy ve FN Motol [18]

1.2 Didaktické zasazení robotiky

V knize Já robot od Isaaca Asimova se v první kapitole objevuje konflikt maminky a jejího manžela. Maminka má strach o svou dceru, která si hraje celé dny s robotem. Má strach z toho, že by se robot mohl porouchat a její dceři se něco stane. Z příběhu je jasně patrné, že pokud my jako lidé něčemu nerozumíme, pak se toho přirozeně bojíme. Vzhledem k velkému rozvoji robotiky v posledních letech je tedy jasné, že nová generace musí získat o robotech alespoň letmé povědomí. Snížíme tím strach z nových technologií v jejich budoucím životě. [5]

Z výše uvedených odstavců je zřejmé, že výuka robotiky na školy rozhodně patří a bude dobré, aby budoucí generace těmto strojům alespoň trochu porozuměla. Také proto proběhla velká revize rámcových vzdělávacích programů, která upravuje vzdělávací oblast Informatika a přidává do ní algoritmizaci, programování a robotiku. [6]

1.2.1 Digitální kompetence

Jednou z velkých změn RVP je úprava klíčových kompetencí. Do programu byla přidána zcela nová klíčová kompetence s názvem digitální kompetence. Tato kompetence se zaměřuje na správné používání digitálních technologií v praktickém životě. [6]

První bod této kompetence se dotýká především digitálních zařízení, služeb a aplikací. Žák by měl být na konci ZV schopen správného používání moderních zařízení. Měl by být schopen používat nové technologie ve svém životě a měl by se umět rozhodnout co pro jaký problém použít. [6]

Další část se týká dat a obsahu. Díky novým technologiím máme k dispozici nepřeberné množství dat a je potřeba se s tímto faktem vyrovnat. Žák by měl být schopen data rozpoznávat, pracovat s nimi. Měl by samostatně rozhodovat, jak data sdílet a uchovávat. [6]

V dalším bodě je rozvíjena kreativita při používání moderních technologií. Cílem je, aby žáci nebyli pouze konzumenti digitálního obsahu, ale aby také byli schopní obsah tvořit. [6]

Je zřejmé, že informatika a moderní technologie usnadňují práci. K tomuto používání moderních technologií jsou vedeni i žáci na základních školách. V životě se setkáváme s mnoha rutinními činnostmi, které je možné pomocí moderních technologií zefektivnit nebo zautomatizovat. [6]

Kromě užitečných moderních technologií se nezřídka na trhu objevují zařízení, která spíše, než aby člověku pomáhala, tak jej zahlcují. Žák by tedy měl být schopen se s novými zařízeními seznámit a kriticky zhodnotit jejich přínos pro svůj život. [6]

Posledním bodem digitálních kompetencí je bezpečnost. A to bezpečnost práce při používání zařízení, ale také stále více diskutovaná bezpečnost dat. [6]

1.2.2 Nová informatika

Původní vzdělávací oblast informační a komunikační technologie byla změněna na oblast s názvem Informatika. Často se o ní také mluví jako o nové informatice. [6]

Nová informatika pro druhý stupeň ZŠ zahrnuje:

- Data, informace a modelování
- Algoritmizace a programování
- Informační systémy
- Digitální technologie

Z těchto kapitol se naše práce zaměřuje především na algoritmizaci a programování. [6]

Algoritmy obecně řeší nějaké úlohy, které jsou často komplexní, avšak každá komplexní úloha může být rozdělena na více jednoduchých úloh, které je již možné řešit. Jedním z očekávaných výstupů podle RVP ZV je: „Žák rozdělí problém na jednotlivě řešitelné části a navrhne a popíše kroky k jejich řešení. Zde vidíme základní slovíčko „problém“, což nám uvozuje zaměření na problémovou výuku. I naše práce je stavěna z velké části problémovými metodami, kdy žák je postaven před nějaký problém, který má vyřešit. Ku pomoci jsou mu různé návodné otázky a nápovědy, které žáka směrují ke zdárnému cíli. [6]

Další očekávaný výstup se týká práce v blokově orientovaném programovacím jazyce. Proto i v naší práci jsou žáci vedeni k práci v takovém prostředí. Existuje mnoho možností, jak je možné pracovat v blokově orientovaném programovacím jazyce. My jsme vybrali prostředí TinkerCad z důvodu jeho jednoduchosti a hlavně komplexnosti. Tento program umožňuje jak programování, tak modelování, tak simulaci elektronických obvodů.

Posledním očekávaným výstupem v rámci algoritmizace je: „Žák ověří správnost postupu, najde a opraví v něm případnou chybu.“ V tomto případě se jedná v programátorské řeči o tzv. ladění programu. Je téměř nemyslitelné, že se program podaří naprogramovat na „první dobrou“. Často práce vypadá tak, že se vytvoří základní kostra programu, a ta se odzkouší. Při každém spuštění programu se zjistí chyby, které program vykazuje, následně se opraví a program se spustí znovu. I s tímto se žáci v naší práci seznámí. Důležité je na tuto část práce připravit učitele, kteří potřebují dostatečné znalosti k vyřešení těchto problémů, aby mohli žáky navést správným směrem. [6]

1.2.3 Informatické myšlení

Informatickým myšlením rozumíme posun od klasické informatiky, která zahrnovala především instrumentální práci s různými programy a nástroji jako například MS Word. Nově se informatika posouvá k rozvoji tvořivých kompetencí s využitím právě informatiky. [9]

Informatické myšlení bylo v dřívějších dobách požadováno pouze po programátorech a vývojářích, kteří tvořili nejrůznější programy pro uživatele. Doba se však posunula a dnes tuto kompetenci potřebuje téměř každý. Pro velkou většinu lidí je denním chlebem práce s digitálními technologiemi. Již děti na základních školách vlastní chytré telefony, a proto je potřeba informatické myšlení rozvíjet. [10]

Podle instituce ISTE je informatické myšlení definováno takto:

> *Informatické myšlení je proces postavený na snaze řešit problémy, který musí vykazovat minimálně tyto znaky:*
>
> - *Formulace problému tak, aby k řešení bylo možné s výhodou použít technologie.*
> - *Organizace dat do logické struktury.*
> - *Reprezentace dat v abstraktní formě prostřednictvím modelů a simulací.*
> - *Řešení realizované formou algoritmu (řada naplánovaných kroků).*
> - *Hledání, analyzování a implementace možných řešení s cílem dospět k co možná nejúčinnějšímu a nejefektivnějšímu výsledku.*
> - *Zevšeobecnění a přenesení způsobu řešení na širší škálu podobných problémů. [10]*

1.2.3.1 Výukový robot jako nástroj

Ve výuce algoritmizace a robotiky na základních školách není cílem vychovat z žáků programátory a techniky, kteří po ukončení vzdělávání začnou pracovat jako vývojáři softwaru nebo hardwaru. Na prvním místě jde o rozvoj informatického myšlení, z čehož také vyplývá, že výukový robot také není cílem, avšak je pouze nástrojem. Žák si během práce na robotu, aniž by si to uvědomoval, osvojuje mnoho konceptů informatického myšlení. [9]

Použití robota je však namístě. Robot jako takový má obrovský motivační charakter. Pro žáky je velmi atraktivní. Výsledky své práce tak žáci ihned vidí na reálném hmatatelném objektu. [9]

Robot přináší do výuky prvky tvořivého myšlení. Žák se při práci setkává s celou řadou problémů, které musí vyřešit za pomocí svých znalostí a dovedností nebo technických možností robota. Žák se tak přirozeně učí rozdělit problém na dílčí části a ty pak samostatně vyřešit. Toto myšlení je základem pro algoritmizaci a programování. [9]

1.2.4 Programování

Výuka programování je velkým strašákem pro učitele, kteří ještě před dvěma lety učili informatiku podle starých RVP. Avšak doba se změnila a RVP s ní. Nyní je na učitele ICT

kladen požadavek na výuku programování. Programování samotné zde však není pro pouhou šikanu učitelů. Je zde hlavně proto, že rozvíjí u žáků logické myšlení, týmovou spolupráci, schopnost se vyrovnat s krizovými situacemi a rozvíjí tolik omílané myšlení informatické. [13]

> *Všichni v této zemi byste se měli učit programování, protože vás to naučí myslet.*
> *(Steve Jobs) [13]*

K programování je zapotřebí znát programovací jazyk. Slovo jazyk zde je z dobrého důvodu. Skutečně se totiž jedná o nový jazyk, podobně jako čeština má svá pravidla, svůj slovosled a žáci se ji učí od samého začátku, tak i programovací jazyk má svá pravidla. Těmto pravidlům se říká syntaxe a potřeba se je správně naučit jinak nám počítač nebude rozumět. [13]

Psaní samotného kódu může být pro někoho kdo se programovat teprve učí velkou obtíží. Naštěstí dnes existuje mnoho různých vývojových prostředí, ve kterých nepotřebujete k vytvoření programu napsat ani čárku textu. Žáci tak jednoduše pouze přesouvají bloky stylem drag and drop. Učí se tak logice celého programu a fungování počítače. [13]

1.3 Metody výuky

Tato kapitola se nezabývá obecným popisem výukových metod, ale popisuje vybrané metody, které jsou aplikovány žáky během práce na robotu.

1.3.1 Názornost

Pravidlo názornosti známe již od jednoho z největších pedagogů v dějinách Jana Amose Komenského. Ten ve svém díle velká didaktika napsal:

> *Proto budiž učitelům zlatým pravidlem, aby všechno bylo předváděno všem smyslům, kolika možno. Totiž věci viditelné zraku, slyšitelné sluchu, vonné čichu, chutnatelné chuti a hmatatelné hmatu; a může-li něco být vnímáno najednou více smysly, budiž to předváděno více smyslům.*
>
> *(Jan Amos Komenský – Velká didaktika) [8]*

Slovní metody mají velkou výhodu hlavně v rychlosti a obsahu předání. Avšak jejich slabinou je, že reálnou skutečnost schovávají za slova. Žák, který se učí pouze slovními metodami by tak měl celý skutečný svět schovaný za pojmy a knihami. Avšak i v práci, která je spíše praktická tyto metody najdou své místo. Každá praxe sebou vždy nese nějakou teorii, ze které vychází. Tuto teorii je nutno předat žákům a k tomu se nejvíce hodí právě slovní metody. [7]

Druhou velkou skupinou a v práci více využívaných metody jsou metody názorně demonstrační. Ta se již opírá o Komenského zlaté pravidlo názornosti. Podle Maňáka a Švece se názornost dělí do několika stupňů:

- Předvádění reálných předmětů a jevů
- Realistické zobrazení skutečných předmětů a jevů
- Jejich záměrné pozměněné zobrazování
- Postihování reality prostřednictvím schémat, grafů, znaků, symbolů, abstraktních modelů, atd. [7]

1.3.2 Vysvětlování

Ačkoli se práce snaží o maximální možnou aktivizaci žáků. Existují ve výuce i momenty, kde se bez jednoduchého vysvětlení určitých zákonitostí neposuneme. Metoda vysvětlování spočívá v působení slovy na kognitivní myšlení žáka. Tato metoda je velmi jednoduchá a univerzálně použitelná v mnoha výukových situacích. Vysvětlování spočívá v logickém a systematickém slovním popisu určité zákonitosti. Důležité je, aby byla respektována dosavadní znalostní úroveň žáků. [7]

Osvědčený způsob postupuje od jednoduchého ke složitějšímu, od konkrétního k abstraktnímu. Vhodné je také užívat názorných pomůcek nejdříve věrných kopií a později také nákresů a náčrtů. [7]

Vysvětlení složitějšího jevu musí jít po malých krocích. Proto i naše práce obsahuje rychlé a stručné pracovní listy. Během postupného vysvětlování musí vyučující neustále kontrolovat, zda žák rozumí předchozímu kroku, jinak se nemůže posunout dále. [7]

1.3.3 Instruktáž

Instruktáž se řadí mezi názorně demonstrační metody. Vzhledem k charakteru práce je právě tato metoda velmi žádaná. Instruktáž nám umožňuje žákům zprostředkovat podněty k jejich praktické činnosti. [7]

Instruktáž může být založena pouze na slovním podání. Problémem v tomto případě je nedostatečná názornost pro samotné technické postupy. Navíc klade na žáky nárok porozumět poměrně dlouhému textu. [7]

Názornějším způsobem je instruktáž založená na hmatových nebo pohybových instrukcích. Tento typ instruktáže spočívá v předvedení samotné činnosti spojené se slovním výkladem. Vyučující taktéž může vést ruce žáka. [7]

1.3.4 Projektová výuka

Podle Maňáka a Švece má projektová výuka interdisciplinární charakter a měla by spojovat práci hlavy a rukou. Během výuky je poukazováno na komplexnost celého problému a cílem je osvojení znalostí a dovedností. Toto je velmi důležité, protože cílem z pohledu učitele není sestavení robota, ale osvojení si znalostí a dovedností žáky. Z pohledu žáka je samozřejmě cíl sestavení robota, tento cíl ve výuce funguje jako silný motivační prvek. [7]

Postup řešení projektového úkolu je členěn do těchto fází:

1. Stanovení cíle
2. Vytvoření plánu řešení
3. Realizace plánu
4. Vyhodnocení

Stanovení cíle je důležitou první částí. Již v předchozím odstavci bylo uvedeno, že cíl vyučujícího a žáka je různý a následná činnost vede ke splnění cílů obou. Důležité pro žáky je přijmout úlohu za svou, důležitá je tak motivace. [7]

Druhým bodem je vytvoření plánu. Žáci musí být seznámeni s celkovým plánem. Je vhodné tento plán vystavit, aby bylo každému zřejmé, kde se právě v plánu nachází. Vhodně zpracovaný plán může rovněž pomoci s motivací žáků. [7]

Při realizaci plánu se jedná o práci na samotném projektu. Tuto fázi by měl vyučující určitým způsobem moderovat. Zjišťovat, ve které části práce žáci právě jsou, jaké jsou jejich problémy, které řeší a podněcovat je k dalšímu bádání. [7]

Na závěr práce musí proběhnout vyhodnocení celkového projektu. V případě sestavení robota je vyhodnocení poměrně nasnadě. Žáci sami zjistí, zda se jim jejich výrobek zdařil či nikoli. Je možné uspořádat například závody jednotlivých vozidel. [7]

Časově se jedná o mimořádně dlouhý projekt, který probíhá paralelně s ostatní výukou na škole. Žáci mají pro práci na svém projektu vyhrazený čas během hodin informatiky. [7]

Při projektové výuce může být užito různých organizačních forem, a to i v jejich kombinaci. Nejčastěji se jedná o individuální výuce prokládané skupinovými aktivitami.

Projektová výuka se řadí mezi komplexní výukové metody, proto se uvnitř projektu objevuje mnoho dílčích jednoduchých metod. [7]

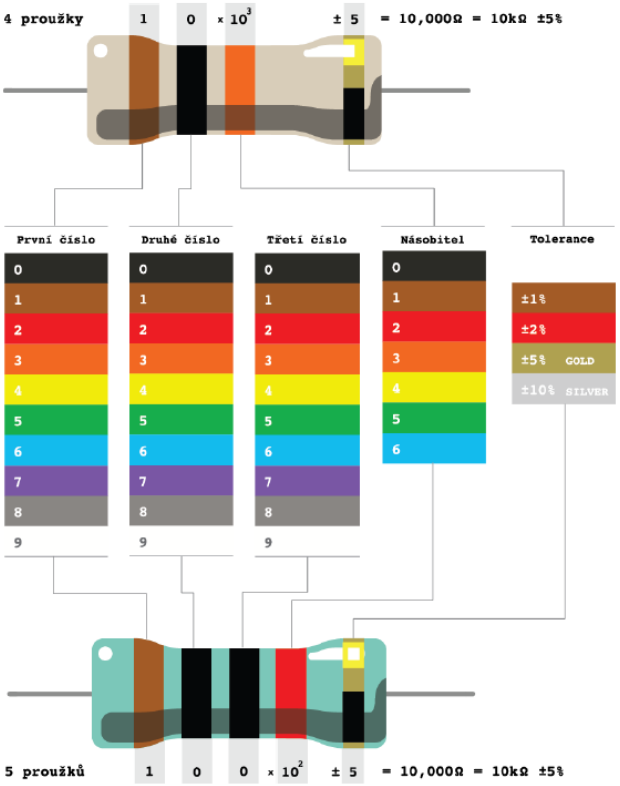
1.4 Popis používaných součástek

1.4.1 Rezistor

Rezistor je jednoduchá diskrétní součástka. Diskrétní znamená, že dělají jednu elementární funkci. Rezistor konkrétně klade odpor elektrickému proudu. Jde v podstatě o jednoduchý převodník elektrické energie na energii tepelnou. V důsledku toho se rezistor při své činnosti zahřívá. Proto kromě samotné velikosti odporu je potřeba počítat ještě s výkonem, který se na rezistoru bude ztrácet. Nejčastěji se setkáme s rezistory se jmenovitým výkonem 0,25W. [2]

1.4.1.1 Barevné značení rezistorů

Jak co nejrychleji zjistit hodnotu rezistorů? Při prvním pohledu na rezistor si můžeme všimnout, že jsou na něm natištěny barevné pruhy. Podle těchto barev se dá z rezistoru vyčíst jeho hodnota a tolerance. [17]



Obrázek 2 - barevné značení rezistorů [17]

1.4.2 Motory

Motor pro naše účely představuje součástku, která tvoří silový výstup našeho elektronického systému. Jinými slovy počítač nám něco vypočítá a na základě svého výpočtu potřebuje ovlivnit nějakou veličinu v reálném světě. [2]

1.4.2.1 Magnetické pole

Princip fungování motoru spočívá na magnetickém poli. Je všeobecně známo, že protékající proud vodičem vytváří ve svém okolí magnetické pole. V okamžiku, kdy tedy navineme vodič například okolo hřebíku a připojíme k němu napětí, začne jím protékat elektrický proud. Tudíž se v okolí objeví magnetické pole, které má vliv na své okolí. [2]

1.4.2.2 Části motoru

Nyní jsme zjistili, jaké je propojení elektrické veličiny a magnetického pole, můžeme tedy pomocí elektřiny ovládat pohyb. Pokud chceme něčím pohnout součástkou první volby

bude jistojistě motor. Motorů máme mnoho druhů a velikostí od nejmenších modelářských motorů až po velké elektromotory v lokomotivách. [2]

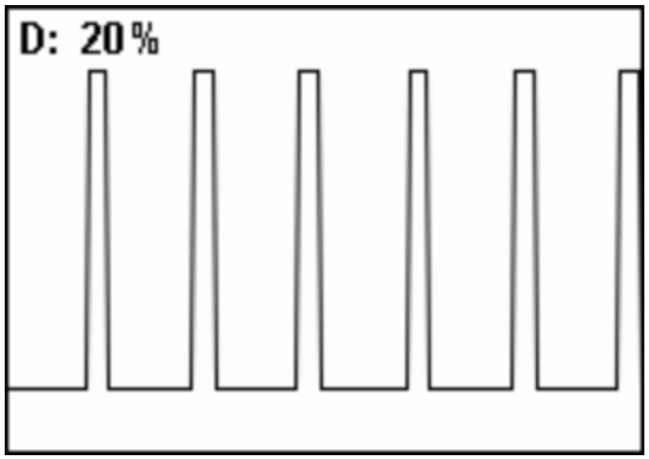
Základní dvě části, ze kterých se motor skládá je *stator* a *rotor*. Jak nám již název napovídá, stator je statická část, která se nehýbe a rotor je dynamická část, která se otáčí. Existují různá uspořádání těchto částí, nejčastěji se setkáme s motory, které mají stator tvořeny permanentním magnetem a rotor je tvořen cívkou, nebo lépe několika cívkami. [2]

Proud, který cívkou prochází vytváří okolo sebe magnetické pole, které reaguje s magnetickým polem permanentních magnetů. Tato pole se vzájemně odpuzují nebo přitahují v závislosti na směru proudu protékajícího cívkou a vyvolávají tak pohyb rotoru. Směr protékajícího proudu se u střídavých motorů mění samovolně spolu se střídáním napětí. U stejnosměrných motorů je zapotřebí vložit součástku zvanou *komutátor*, který zajišťuje změnu směru protékajícího proudu. [2]

Je možné konstruovat stejnosměrný motor i bez komutátoru, potom však je potřeba, aby řídící elektronika řídila i směr proudu protékajícího motorem. Toto je i náš případ pro konstrukci robota. Více v kapitole 1.4.2.3 řízení otáček motoru PWM. [2]

1.4.2.3 PWM

PWM neboli pulse width modulation, česky ji překládáme jako pulzně šířkovou modulaci. Náš číslicový systém (Arduino) umí skvěle pracovat s jedničkami a nulami, ale s analogovými hodnotami je to horší. PWM nám dovede alespoň trochu simulovat analogovou hodnotu. [2]



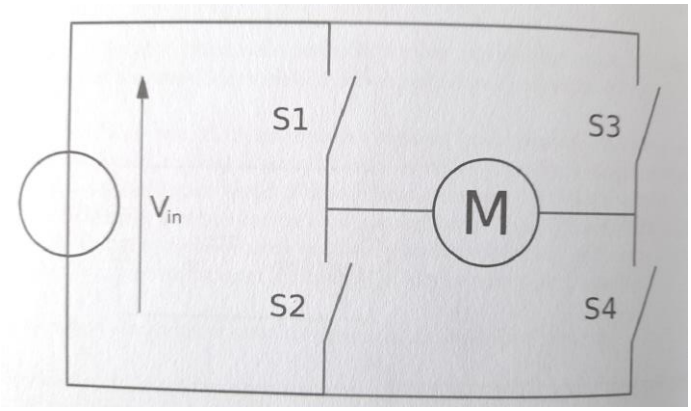
Obrázek 3 - PWM s plněním 20% [3]

Pokud bychom do motoru, respektive do H-můstku pustili signál, který bude trvale v hodnotě logické 1. Potom se motor bude točit plným výkonem. Kdybychom pustili logickou nulu, motor se zastaví. Jak to ale udělat, aby se motor otáčel třeba na 50% svého výkonu? Do motoru nebudeme pouštět celkově jedničku ani celkově nulu, ale signál, který můžeme měnit. Takový signál si rozdělíme na jednotlivé periody a v každé periodě necháme chvíli hodnotu 1 a chvíli hodnotu 0. Tyto „chvíle“ mohou být různě široké (odtud pulzně šířková modulace). Pro lepší vysvětlení uvedeme příklad. [2]

Představme si, že jednu periodu zvolíme 1s. Potom pokud bychom chtěli 50% výkon motoru, tak budeme 0,5s mít výstup v logické 1 a druhou 0,5s v logické 0. Pro výkon 25% necháme zapnutou logickou jedničku ¼ periody, tedy 0,25s a zbytek času bude logická 0. [2]

1.4.3 H-můstek

Je důležité si uvědomit jednu fyzikální podstatu magnetického pole. Magnetické pole si můžeme představit také jako místo, ve kterém je uloženo nějaké množství energie. V okamžiku, kdy přestane protékat proud cívkou, okolo které magnetické pole vzniklo, tak toto pole zaniká. V důsledku toho se nahromaděná energie magnetického pole musí někam přeměnit. To nám říká zákon zachování energie. Tato energie se samozřejmě přemění na elektrický proud, který má ale opačný směr než ten, kterým jsme magnetické pole vybudili. V praxi to pro nás znamená, že kdybychom motor připojili přímo k řídícímu čipu, potom by ho tento zpětný proud jistě zničil. [2]



Obrázek 4 - princip fungování H-můstku

Druhým důvodem, proč nemůžeme připojit motory přímo na čip je ten, že řídící elektronika není stavěna na spínání velkých proudů. Proto musíme mezi procesor a samotný motor vložit nějaké zapojení, které

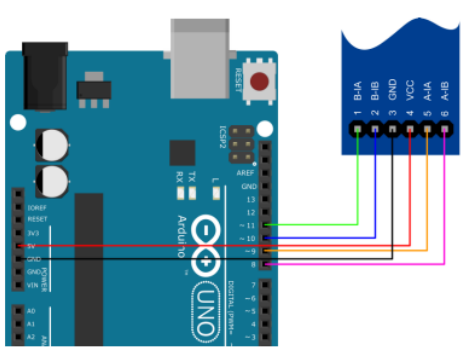
1. bude schopné spínat velké proudy do motorů
2. poradí si se zpětným proudem

Jednoduchou součástkou splňující obě podmínky je tzv. H-můstek. Toto zapojení navíc umožňuje řídit směr otáčení motoru. [2]

Jak ukazuje Obrázek 4 pochopení principu fungování H-můstku je jednoduché. Pokud jsou sepnuty spínače 1 a 4 pak proud prochází motorem z leva do prava a motor se tak otáčí jedním směrem. V druhém případě jsou sepnuty spínače 2 a 3 a proud tak prochází směrem opačným a motor se tedy otáčí směrem opačným. [2]

1.4.4 L9110S

Součástkou první volby, která je i prodávaná v didaktických sadách pro robota by mohla být L293D ta má však jednu velkou nevýhodu a tou je, že její pracovní napájecí napětí se pohybuje mezi 4,5V a 25V. Celý robot by měl být napájen powerbankou, která má výstupní napětí 5V. V okamžiku, kdy dojde k nějakému, byť i krátkému zatížení, a tudíž poklesu napětí, celý obvod se nám tak resetuje. Toto je velmi nechtěný stav, a proto jsme zvolili raději součástku L9110S. [11]



Obrázek 5 - zapojení L9110S

Pracovní napětí tohoto modulu začíná na 2,5V, čímž nám odpadá problém s resetováním modulu. Další velkou výhodou tohoto modulu je jednoduchost jeho zapojení. Popis pinů a jejich zapojení zobrazuje Obrázek 5. [11]

1.4.5 Senzor vzdálenosti

Vzdálenost od určité překážky bychom v reálném světě pravděpodobně změřili metrem. Avšak náš robot nemůže jen tak jednoduše roztáhnout metr a přečíst si, jak daleko je od překážky. K tomu použijeme senzor vzdálenosti (v práci jej označujeme také jako ultrazvukový senzor). [2]

Fyzikální podstatou tohoto senzoru je fakt, že rychlost zvuku je konečná. Možná by se mohlo zdát, že zvuk se pohybuje ohromně velkou rychlostí, avšak v časech, ve kterých počítá náš procesor, což je přibližně 16 operací za mikrosekundu, již rychlost zvuku tak obrovská není. Rychlost šíření zvuku záleží na mnoha faktorech (vlhkost vzduchu atd.) V praxi se používá hodnota 340m/s. [2]

Princip fungování ultrazvukových senzorů je následující. Reproduktor vyšle signál o mnohem vyšší frekvenci, než je slyšitelný rozsah a čeká za jak dlouho se tento signál odrazí zpátky do mikrofonu. Tuto dobu potom můžeme dosadit do vzorce pro výpočet vzdálenosti. [2]

Vycházíme z dobře známého vzorce pro výpočet rychlosti.

$$v = \frac{s}{t}$$

Rovnice 1 - základní tvar pro výpočet rychlosti

Upravíme si jej pro výpočet vzdálenosti:

$$s = v.t$$

Rovnice 2 - upravený tvar pro výpočet dráhy

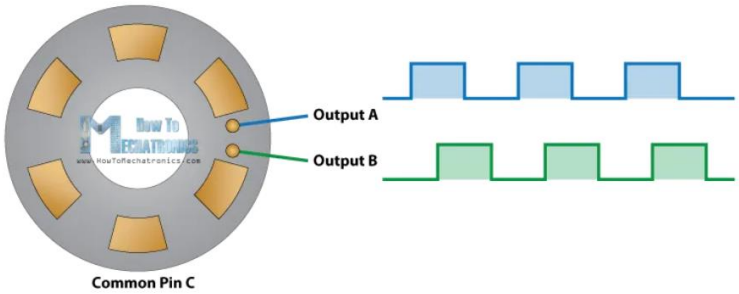
Nesmíme však zapomenout na fakt, že změřený čas je za cestu zvuku směrem k překážce a zpátky. Proto výslednou vzdálenost překážky musíme vydělit 2.

$$s = \frac{v.t}{2}$$

Rovnice 3 - upravená rovnice pro výpočet dráhy

1.4.6 Senzor čáry a enkodér

Enkodér i senzor čáry fungují na stejném principu. Dokonce jsou k němu využity i stejné součástky. Fyzikálním principem je různá odrazivost barevných ploch. Jak je známo obecně bílé a světlé plochy odráží světlo více než plochy tmavé a černé, ty naopak světlo pohlcují. Jako senzor odrazu využijeme fototranzistor, který se tím více otevře, čím více světla dopadá do báze tohoto tranzistoru. Vedle něj umístíme infračervenou diodu, která trvale svítí v neviditelném spektru. Pokud k senzoru přiblížíme světlý předmět, pak se světlo odrazí zpátky a dopadne do báze fototranzistoru. Jakmile však přiblížíme černý předmět (například vodící čáru), pak se většina světla pohltí a do báze fototranzistoru dopadne mnohem méně světla. [2]



Obrázek 6 - princip fungování rotačního enkodéru [15]

V předchozím odstavci jsme si vysvětlili princip fungování samotného optického senzoru. Nyní uvedeme, jakým způsobem pracují rotační enkodéry. Enkodér je součástka, která má za úkol zjistit kterým směrem a případně o jaký úhel se kolo otáčí. [15]

Základem je aby rotační část byla rozdělena různě odrazivými povrchy v přesně definovaných výsečích. Jak ukazuje Obrázek 6 zde každá výseč zaujímá 30°. Tyto výseče tak definují i maximální rozlišení enkodéru. Další důležitou podmínkou je, aby čidla byla umístěna tak, aby byla schopna ve stejný okamžik snímat stejnou výseč. Zároveň však musí být ode sebe časově posunuté. Tento časový posun je na témže obrázku znázorněn dvěma výstupními signály. Je zřejmé, že oba mají nástupnou hranu vůči sobě posunutou. [15]

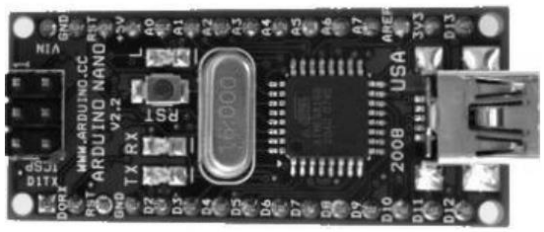
Tyto dva výstupní signály je třeba následně programově zpracovat. K tomu poslouží Arduino. Směr otáčení se určí jednoduše. Pokud první nástupná hrana je registrována na výstupu A a následně projde nástupní hranou výstup B, potom je zřejmé, že se enkodér točí po směru hodinových ručiček. V opačném případě, tedy že nejprve nastoupí výstup B a potom výstup A, tak to znamená, že se enkodér otáčí proti směru hodinových ručiček. [15]

Směr je tedy určen nyní je potřeba určit úhel natočení. Řekněme, že při každé nástupné hraně signálu přičteme do nějakého počítadla 1. následně víme, že počet vzestupných hran díky enkodéru je 6. Tedy jedna vzestupná hrana odpovídá úhlu 60°. Stačí nám tedy vynásobit hodnotu čítače 60 a získáváme tak informaci o úhlu natočení kola. Čítač je potřeba při každé otočce resetovat do 0. [15]

1.5 Arduino

1.5.1 Arduino Nano

Arduino Nano je vývojová deska z rodiny Arduino. Deska je záměrně designovaná tak, aby bylo možné ji pohodlně připojit do kontaktního pole. Na první pohled se může zdát, že kvůli své velikosti nemůže poskytnout tolik funkcí jako například jeho větší bratr Arduino Uno. Opak je ale pravdou. Vývojová deska je osazena stejným procesorem jako Arduino Uno. A dokonce dodržuje přesně stejné značení a číslování pinů jako Arduino Uno. [16]



Obrázek 7 - Arduino Nano [3]

1.5.2 Arduino IDE

Samotná zkratka IDE vychází z anglického integrated development environment. Česky bychom toto sousloví mohli přeložit jako integrované vývojové prostředí. Často se používá pouze termín vývojové prostředí. [3]

Vývojové prostředí Arduino IDE je počítačový program, který nabízí běžné funkce, které bychom od programu pro psaní kódu očekávali. Mezi ty základní patří:

- Verifikace kódu
- Nahrání kódu do desky
- Uložení a otevření souboru
- Podpora knihoven
- Sériový monitor

Program nabízí funkcí daleko více avšak pro účely této práce si vystačíme s výše jmenovanými. [3]

1.5.3 Programovací jazyk Wiring

Veškeré Arduino desky je možné programovat v jazyce C nebo C++. Wiring je knihovnou jazyka C++, která rozšiřuje tento jazyk o mnoho knihoven a funkcí speciálně vyvinutých pro Arduino. Tato knihovna je již natolik komplexní, že se často označuje za samostatný programovací jazyk. [3]

Základní struktura programu obsahuje dvě hlavní funkce.

```
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run
repeatedly:
}
```

Funkce `void setup(){}` slouží pro psaní kódu, který se spustí jenom jednou na začátku programu, to znamená při nahrání programu do čipu, stisknutí tlačítka reset nebo připojení napájení. Typicky se tento blok používá pro nastavení vstupně výstupních pinů iniciaci proměnných a podobně. [3]

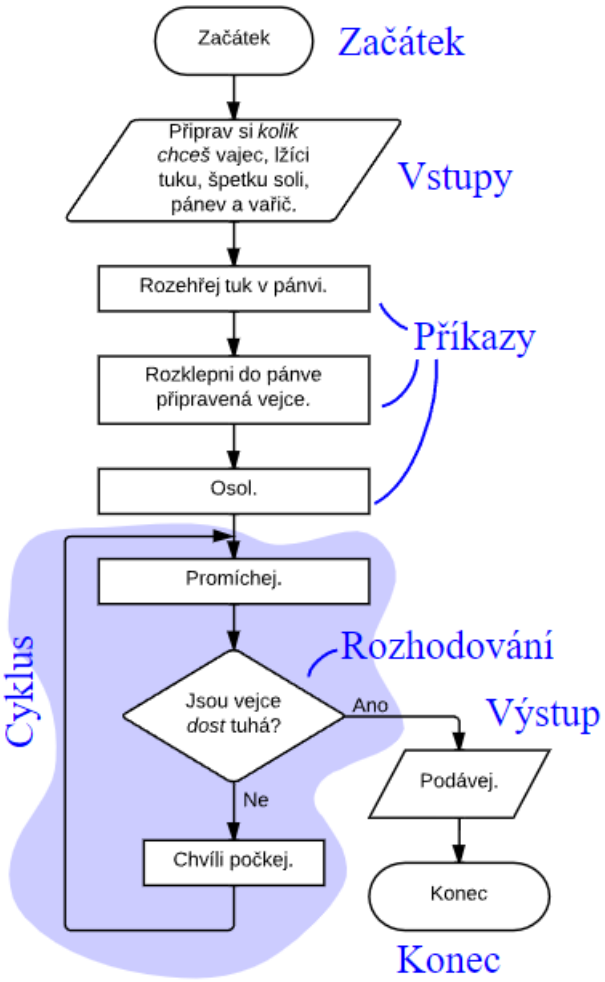
Funkce `void loop(){}` obsahuje kód, který se opakuje stále dokola až do doby, kdy je napájení odpojeno. Mikročipy nemohou jen tak nečinně ležet. Jinými slovy musí vždy a neustále pracovat a něco počítat. [3]

Tyto dvě základní funkce musí program obsahovat vždy. Je možné si dopsat další vlastní funkce a pro přehlednost programu je to i vhodné, dokonce didakticky vhodné, ale nikdy se nesmí stát, že budou tyto bloky v programu chybět. I v případě, že by byly prázdné. [3]

1.5.4 Vývojové diagramy

Běžně užívaný český jazyk není vždy vhodný pro sdělování informací o algoritmech a o tom, jak by měly algoritmy fungovat. Proto se pro tyto účely vyvinul nový druh jazyka v podobě vývojových diagramů. [20]

Vývojové diagramy pracuj s pevně definovanými tvary. **Začátek a konec** programu se vyznačuje obdélníkem se zakulacenými rohy nebo kruhem s popiskem start (s) a konec (k). Dalším blokem je kosodélník, ten se využívá k zápisů **vstupních nebo výstupních dat** do nebo z programu. Dále na obrázku vidíme několik obdélníkových bloků. Ty slouží pro **zápis příkazů**, kdy má program sám něco udělat například něco spočítat, přesunout, vyřešit a podobně. Posledním blokem je důležitý **rozhodovací blok** v programu je nejčastěji reprezentován podmínkou if. Tento rozhodovací blok má dva výstupy, kde jeden představuje kladnou odpověď a druhý odpověď zápornou. **Cykly** se zapisují pomocí šipek, které se vracejí do určité části programu. [20]



Obrázek 8 - vývojový diagram [20]

1.6 Připojení periferních součástek k Arduinu

1.6.1 Digitální a analogové piny

V předchozích částech textu jsme několikrát zmínily slovo pin. Často ve spojení vstupní a výstupní nebo analogové a digitální. Pin je alfa a omega celého programování prakticky všech čipů. Slovo pin používáme pro označení jedné nožičky integrovaného obvodu. Každý pin má jinou funkci a u mikroprocesorů má obvykle funkcí několik. [3]

Piny slouží pro připojení externích součástek a modulů. Například chceme-li pomocí čipu ovládat LED diodu, musíme ji připojit k nějakému pinu. Dále je důležité rozlišit, zda s pinem pracujeme jako se vstupem nebo výstupem. Pokud chceme Arduinem ovládat nějaké zařízení, například motor nebo LED diodu, je potřeba nastavit pin jako výstupní. Naopak pokud potřebujeme nějakým zařízením ovládat Arduino, například tlačítko nebo senzor, potom musíme nastavit pin jako vstupní. [3]

K této operaci má jazyk wiring přímo definovanou funkci. Její syntaxe je následující:

```
pinMode(číslo_pinu, OUTPUT|INPUT);
```

Uvnitř závorky jsou dva tzv. argumenty funkce. Pomocí těchto argumentů určujeme, jak se bude funkce chovat. Neboli nastavíme pin, který je označený konkrétním číslem jako vstupní nebo výstupní. Právě funkce pinMode se obvykle nastavuje v části setup. Viz kapitola 1.5.3 [3]

Čísla pinů zjistíme z dokumentu, který se obecně označuje jako pinout. Na internetu jich je k dostání obrovské množství, samozřejmě jeden také nalezneme přímo na webových stránkách Arduina nebo v příloze tohoto dokumentu. [16]

1.6.2 Digitální vstup a výstup

Již máme určeno, zda je pin vstupní nebo výstupní. Nyní je na čase s tímto pinem začít pracovat.

Jednodušší varianta pro práci je pin výstupní. V podstatě pouze říkáme, zda má procesor na výstupní pin zapsat logickou jedničku nebo nulu. Logická jednička odpovídá napětí +5V a logická nula potom napětí 0V. Je tedy zřejmé, že pokud bychom na pinu měli připojenou LED diodu v propustném směru, pak logická jednička diodu rozsvítí a logická nula zhasne. Funkce vypadá následovně:

```
digitalWrite(číslo_pinu, LOW|HIGH);
```

Funkce má opět dva argumenty. Jedním je již známé číslo pinu a druhým je hodnota logické nuly (LOW) nebo logické jedničky (HIGH) [3]

Ovládání vstupního pinu je také jednoduché. V principu přikážeme čipu, aby zjistil, jestli je na konkrétním pinu logická jednička (+5V) nebo logická 0 (0V) a tuto hodnotu zapsal do nějaké proměnné. Syntaxe pro čtení vstupu je následující:

```
Promenna = digitalRead(číslo_pinu);
```

Vidíme, že na začátku řádku nám přibyla proměnná, to je část paměti, do které zapíšeme hodnotu pinu. [3]

Pozor důležité je si uvědomit, že zde se jedná pouze o digitální čtení a zápis dále je možné pracovat i s analogovými hodnotami, ale o tom až v kapitole 1.6.4.

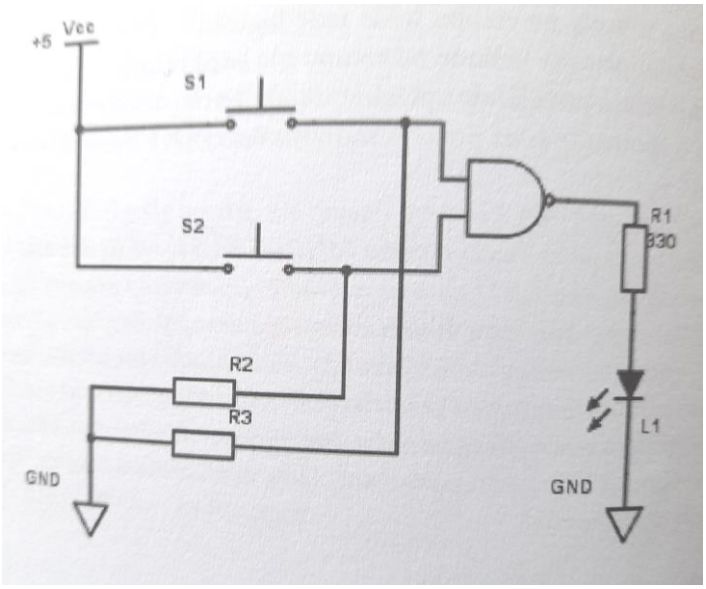
1.6.3 Pull – Up a Pull – Down zapojení

Představme si situaci, že k Arduinu chceme připojit tlačítko. Nejjednodušší věc, která asi napadne každého začínající ho elektronika je, že spínač vložíme mezi pin procesoru a napájecí napětí +5V. Potom logicky po stisknutí tlačítka se na vstup procesoru propojí logická 1. Co se ale stane, pokud tlačítko uvolníme? Mezi tlačítkem a procesorem není ani logická

jednička (tu jsme odpojili) a ani uzemnění (to není připojeno). Jedná se o neurčitý stav a každá řídicí elektronika jej může vyhodnotit jinak. [2]

Jednoduše tedy připojíme trvale k pinu procesoru zem a tím nám neurčitý stav zmizí. To je sice pravda, ale zamysleme se nad tím, co se stane v tomto případě po stisknutí tlačítka. Propojí se nám na přímo napájecí napětí přes tlačítko přímo do země. Tomuto propojení se říká zkrat. [2]

Proto použijeme nějaký velký rezistor obvykle s hodnotou okolo 10kΩ a připojíme zem k procesoru přes tento rezistor. Potom při vypnutém tlačítku máme přes rezistor připojenou logickou nulu a mizí nám tak neurčitý stav a při stisknutí tlačítka je odpor dostatečně velký, aby zamezil zkratu. [2]

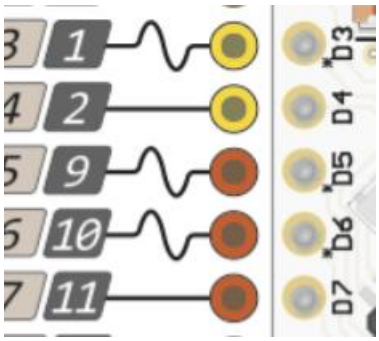


Obrázek 9 - příklad užití pull-down rezistorů [2]

Tomuto zapojení se odborně říká pull-down zapojení. Slovo down značí, že rezistor je připojení přímo na zem. Analogicky k tomuto případu se používá ještě pull-up zapojení, které funguje na opačném principu a rezistor je zde zapojen na k napájení zdroje. [2]

1.6.4 Analogový vstup a výstup

Již jsme se seznámili s digitálním vstupem a výstupem, co když ale chceme pracovat s analogovou částí. Například u našeho robota se hodí řídit rychlost motorů. K tomu má jazyk wiring dvě důležité funkce. Než se však pustíme do jejich popisu, je potřeba si uvědomit, že ne všechny piny Arduina umožňují práci s analogovými informacemi. Pokud se podíváme do popisu pinů, můžeme si všimnout, že některé piny jsou označeny vlnovkou. Právě tyto piny je pak možné použít jako analogový výstup. [2]



Obrázek 10 - vlnovkou označené piny pwm [16]

Podobně jako pro digitální práci s piny jazyk disponuje funkcemi digitalWrite a digitalRead, tak pro práci s analogovými hodnotami máme k dispozici funkce analogWrite a analogRead.

Funkce analogWrite má následující syntaxi:

```
analogWrite(číslo_pinu, hodnota);
```

Jako hodnotu můžeme zapsat číslo od 0 do 255. Je to dáno tím, že pracujeme s osmibitovými procesory a ty jsou pomocí osmi bitů schopné zakódovat číslo maximálně 255. Položme si nyní řečnickou otázku: „Jak je možné, že ryze digitální zařízení dovede na svůj výstup vložit analogovou hodnotu?“ Odpověď nalezneme v kapitole 1.4.2.3. [2]

Podobně jako má Arduino označené analogové výstupy, máme určené i analogové vstupy. Ty se však neoznačují vlnovkou, ale písmenem A. Tyto vstupy disponují 10 bitovým AD převodníkem, který nám měřitelnou hodnotu 0 – 5V rozdělí na 2^{10} = 1024 hodnot. [2]

Funkce analogRead funguje podobně jako její digitální verze. Syntaxe je následující:

```
Promenna = analogRead(číslo_pinu);
```

Do proměnné se nám uloží hodnota napětí na pinu jako číslo od 0 do 1024. [2]

1.7 3D modelování a 3D tisk

1.7.1 Historie 3D tisku

Může se zdát, že 3D tisk je nová disciplína posledních přibližně 20 let. Opak je však pravdou. První experimenty s 3D tiskárnami nalezneme v roce 1984. V tomto roce si Charles W. Hull nechal patentovat technologii zvanou stereolitografie. [4]

Dalším velkým milníkem pro 3D tisk byl rok 2005. Doktor Adrian Bowyer v tuto dobu založil projekt RepRap. Tento projekt byl od počátku open source, to znamená, že kdokoli si mohl prohlédnout zdrojové kódy a zdarma s nimi libovolně nakládat. Možná právě z tohoto důvodu se v této době 3D tisk rozvíjí neuvěřitelnou rychlostí. O 3D tisk se tak začala zajímat odborná i laická veřejnost celého světa. [4]

1.7.2 Technologie 3D tisku

3D tisk je v principu založen na nanášení jednotlivých tenkých vrstev. Představme si, že jakýkoli model rozkrájíme na tenké plátky, podobně jako se krájí salám a následně je skládáme vrstvu po vrstvě na sebe. Technologie 3D tisku lze rozdělit do 3 kategorií (v závorkách jsou uvedeny příklady těchto technologií):

- Materiál v podobě tiskové struny je extrudovaný (FDM, FFF)
- Tekutý materiál je vytvrzován na definovaných oblastech (SLA)
- Materiál v podobě prášku je spékán laserem (SLS) [4]

1.7.2.1 SLA technologie

Jak jsme uvedli v kapitole 1.7.1, nejstarší technologií 3D tisku je stereolitografie. Označujeme ji zkratkou SLA. Základním materiálem pro tisk je fotocitlivý polymer, který je na požadovaných místech vytvrzován pomocí světla. Existují tři hlavní kategorie procesu, které dělíme podle zdroje světla.

1. **SLA - Laser**: Zdrojem tohoto světla je laser, jehož paprsek je usměrňován zrcadly na požadovanou část vrstvy, kterou je potřeba vytvrdit. Doba tisku tak závisí na ploše jednotlivých vrstev.
2. **DLP**: Digital light processing je v podstatě klasický projektor, který cíleně osvítí všechny požadované body najednou. Výhodou je rychlost tisku, která je nezávislá na velikosti plochy jako to bylo v předchozím případě.
3. **MSLA:** Mask SLA v principu funguje podobně jako DLP avšak osvit jednotlivých pixelů zajišťuje maska tvořená LCD displejem. Stejně jako v předchozím případě je čas pro tisk jedné vrstvy nezávislý na ploše vrstvy. [4]

1.7.2.2 SLS technologie

Tato technologie zatím nepronikla mezi širokou veřejnost. Uplatňuje se v průmyslovém odvětví. Princip fungování by se dal připodobnit k fungování laserové tiskárny na papír. Na vrstvu se nanese materiál v podobě prášku. Tuto tenkou vrstvu prášku poté osvětluje laser, který jednotlivé části vrstvy speče a vytvoří tak tuhou strukturu. Z toho vyplývá, že po dokončení tisku je celý model zasypaný nespečeným práškem. [4]

1.7.2.3 FFF/FDM technologie

Jako poslední uvádíme technologii FFF, která je pravděpodobně jednou z nejpoužívanějších technologií v hobby úrovni. Základním materiálem je roztavený plast dodávaný v podobě plastové struny tzv. filamentu. Tato struna prochází rozehřátou tryskou a je vstřikována na plochu tisknutého modelu. Způsob pohybu tiskové hlavy v prostoru umožňuje tiskárny rozdělit na tyto podkategorie:

1. Kartézská: Pohyb je definovaný pomocí 3 základních os x, y a z. Každou osu ovládá krokový motor. Osy x a z zajišťuje pohyb samotné tiskové hlavy. Osu y potom pohyb podložky, na které se samotný model tiskne.
2. Delta: Extruder je zavěšen na třech ramenech. Výhodou je vysoký prostor pro tisk, avšak nevýhodou je matematicky složitější pohyb tiskové hlavy.

3. Polar: Jak již název napovídá jedná se o pohyb hlavy v polárních souřadnicích. Tisková hlava se pohybuje po dvou osách a samotná podložka s modelem se otáčí. Výhodou je konstrukčně jednoduchá tiskárna. Nevýhodou složitost při přípravě modelu pro tisk. [4]

1.7.2.4 Komponenty FFF tiskárny

V minulé kapitole jsme několikrát použili slova jako například extruder, podložka a podobně. Nyní je čas si vysvětlit z čeho se 3D tiskárna skládá.

Základem každé tiskárny je *extruder*, neboli tisková hlava. Do tiskové hlavy směřuje filament, který je v ní zahříván a roztavován. Dále je již tekutý plast vtlačován do trysky, která vytváří tenkou plastovou vrstvu na povrchu tisknutého modelu. [4]

Extruder vytlačuje materiál na další důležitou část tiskárny a tou je *vyhřívaná podložka*. Vyhřívání je zde z důvodu teplotní roztažnosti látek a zabraňuje tak kroucení výsledného modelu. [4]]

Krokové motory zajišťují pohyb všech částí. Výhodou je přesné řízení otočení těchto motorů. V celé tiskárně najdeme 4 motory. Jeden pro každou osu pohybu a poslední pro vytlačování filamentu v tiskové hlavě. [4]

Důležitou a nedílnou součástí je *řídící jednotka*, která má za úkol zpracovat tisková data a převést je do pohybů jednotlivých částí. [4]

Veškeré výše uvedené vybavení 3D tiskárny musí být umístěno v konstrukčním *rámu*. [4]

1.7.3 Postup 3D tisku

Celý proces 3D tisku se skládá z několika kroků, které na sebe navzájem navazují. [12]

1.7.3.1 Získání modelu

Prvním a logickým krokem je získání modelu. Ten je možné buď stáhnout, což z hlediska výuky nedává příliš smysl, nebo vymodelovat v k tomu určených programech. Programy pro práci s 3D modely jsou k dispozici jak profesionální, kde se cena licence pohybuje v tisících korunách, tak hobby programy, které jsou k dispozici zdarma nebo jsou dokonce online. [12]

Zde uvádíme krátký přehled vybraných programů:

Tabulka 1 - Přehled programů pro 3D modelování [12]

Profesionální	Hobby	Hobby online
Autodesk Maya	Sketch Up	**TinkerCAD**
3ds Max studio	Blender	Shapesmith
AutoCAD		

V tabulce jsme záměrně zvýraznili program TinkerCAD. Je to z jednoduchého důvodu. Tento program je velmi jednoduchým programem pro 3D modelování a vhodný pro začátečníky. Princip práce je jednoduchý a intuitivní. V pravé části obrazovky se nachází knihovna modelů, které můžeme jednoduše přetáhnout do pracovní plochy. V pracovní ploše je pak možné měnit objektům parametry a aplikovat na ně funkce průnik nebo sjednocení. [4]

1.7.3.2 Slicování

Výsledný soubor z TinkerCADu je obvykle ve formátu stl. Tento formát není možné jednoduše nahrát do 3D tiskárny. Model je potřeba převést do formátu, kterému 3D tiskárna rozumí. K tomuto úkonu slouží program Slicer. Slicer rozdělí celý model na jednotlivé vrstvy, ve kterých tiskárna nanáší materiál a spočítá přesné vektory pro pohyb tiskové hlavy. Program pro slicování modelu doporučujeme PrusaSlicer, který je zdarma a v případě tisku na 3D tiskárně od Josefa Průši, program disponuje přednastavenými profily jednotlivých tiskáren. [4]

Použití 3D tisku ve školním prostředí je rychlost celého procesu. 3D tisk je velmi pomalý proces a u větších modelů, což podvozek robota bezpochyby je, trvá časově velmi dlouhou dobu, než se model vytiskne. Často tak není možné model vytisknout v rámci jedné vyučovací hodiny. [12]

1.7.3.3 Příprava podložky

Důležitým krokem před samotným tiskem je odstranění veškerých nečistot a mastnoty z tiskové podložky. Tento krok je velmi důležitý, protože jinak se na podložku neuchytí nanášený materiál a dojde k poškození modelu. [4]

1.7.3.4 Spuštění tisku

Spuštění tisku znamená nahrání vygenerovaných dat z programu slicer do tiskárny. Důležité je, aby tok těchto dat byl během tisku nepřerušený, jinak dojde k nenávratnému poškození tisku. Nejvhodnějším způsobem je využití SD karty nebo jiného podobného zařízení. Celý proces tisku pak není závislý na žádném jiném zařízení. [4]

1.7.3.5 Postprocessing

Vytištěný model je možné ihned použít jako funkční díl. Avšak ze zkušeností víme, že vymodelovat model tzv. na první dobrou se nám nepodaří, a tím méně žákům. Pravděpodobně bude potřeba model na různých místech různě upravit. K tomu slouží různé techniky od jemného broušení, až po řezání. [4]

1.8 Tvorba vzdělávacího materiálu

Jak již bylo zmíněno výše, v posledních letech prošla informatika na školách obrovskou změnou a nemyslíme pouze RVP. Máme na mysli také to, že informatika jako vědní obor je téměř jeden z nejrychleji se rozvíjejících oborů vůbec. To ovšem v praxi učitele znamená, že veškerý materiál v podobě učebnic, pracovních sešitů a podobně velmi rychle zastará. Proto je potřeba, aby učitelé, kteří informatiku učí, své materiály neustále aktualizovali, čímž se oni sami stávají tvůrci učebnic a pracovních sešitů. [14]

1.8.1 Vrstvy poznání

Koncepce vrstev poznání nám říká, že jsou určité úrovně poznání, které žáci mohou poznat. Vytvořený materiál by měl poskytovat dostatek prostoru pro nadané a zvídavé žáky a zároveň ti méně zdatní žáci, by pomocí tohoto materiálu měly získat alespoň základní přehled o probíraném tématu. [14]

Tvorba vzdělávacího obsahu probíhá ve 3 vrstvách:

1. vrstva: Učitel, který tuto vrstvu vytváří musí dobře znát obor, který má didakticky transformovat. Musí vybrat nejjednodušší, avšak vědecky správné informace, které by měl znát po absolvování výuky úplně každý. Důležitá je motivace žáků. Té je možné dosáhnout častými didaktickými hrami, odkazy do historie i popkultury. Ve výsledku žák musí mít pocit, že se ho probírané téma nějakým způsobem týká. [14]

2. vrstva: Jedná se o samotnou práci žáka. Motivaci získává z vrstvy první. Zde jde především o zadané úlohy a jejich řešení. [14]

3. vrstva: Nejvyšší vrstva je směřována k nadaným a zvídavým žákům. Tato vrstva počítá s vnitřní motivací a samotné studenty již dále nemotivuje. Žákům by se mělo dostat prohloubení znalostí nad úroveň kurikulárního učiva. [14]

2 Praktická část

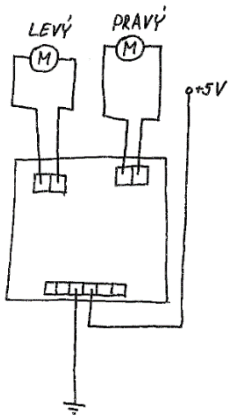
Celá práce na praktické části spočívala ve dvou hlavních částech. Nejprve bylo nutné zkonstruovat robota, který se skládá z konstrukce podvozku na 3D tiskárně, elektrického zapojení součástí a finálního naprogramování samotného programu. Druhá část práce spočívala v tvorbě pracovních listů pro žáky a metodických listů pro učitele. Cílem bylo vytvořit vzdělávací materiál pro pedagogy, kteří se relativně nově potýkají s revidovaným RVP ZV, které zahrnuje robotiku.

2.1 Používaná zjednodušení

Jak jsme zmínili v teoretické části, není možné žákům předávat znalosti z oboru informatiky, resp. Robotiky bez značného zjednodušení. Proto i my v práci používáme některá zjednodušení, která žákům usnadní vstup do jinak složitého oboru elektroniky.

2.1.1 Zjednodušené nákresy elektrických obvodů

Zápis elektrických zapojení se řídí přesně stanovenými normami. Tyto normy stanovují, jakým způsobem se jaké součástky zakreslují. Jedná se o technickou, často abstraktní dokumentaci. My v práci používáme takové nákresy součástek, které se snaží zobrazit součástku tak, jak skutečně vypadá. Jde tedy o takový hybrid mezi plně technickým výkresem a skutečným zobrazením součástek.



Obrázek 11 - zjednodušený nákres obvodu L9110S

2.1.2 Zjednodušené nákresy konstrukčních částí

Podobně jako v nákresech elektrických obvodů jsme zjednodušili i nákresy konstrukčních částí. V části, kde žáci modelují svůj vlastní podvozek pro robota využíváme zjednodušené kresby a neúplné kótování. Opět tyto výkresy jsou upravovány normami, které z didaktického důvodu nedodržujeme.

2.2 Pracovní a metodické listy

Celkovou konstrukci robota jsme rozdělili do větších logických celků, které na sebe navazují. Zároveň v diagramu (Obrázek 12) je barevně odlišena obtížnost jednotlivých kroků. Pro úspěšnou konstrukci robota, který bude schopen sledovat čáru a zastavit se před překážkou, postačuje splnění pouze modrých nejjednodušších částí práce.

Pracovní listy se snaží vést žáka pomocí návodných otázek a kreseb. Kombinují v sobě několik způsobů výuky. Celkově se určitě jedná o projekt, a tedy projektovou výuku. Ta v sobě zahrnuje konstruktivistické hledisko, kdy žák vychází z něčeho, co zná a na tyto znalosti jsou stavěny další informace. Často se v pracovních listech objevuje práce s problémem, kdy záměrně je žák zaveden do problémové situace, kterou mu následně pomůžeme vyřešit pomocí návodných otázek. Tyto problémy jsou stavěny tak, aby si žák uvědomil důvody různých zapojení nebo postupů práce.

Každý pracovní list se skládá ze dvou stran, kdy na jedné straně je žák postaven před zadání úkolů, které má vyřešit. Po otočení stránky si žák může ověřit svá řešení.

Robot Rudolf je postavičkou robota, který má za cíl motivovat žáky a provádět je celým procesem konstrukce robota. Občas přináší žákům odlehčení v podobě vtipu, ale objevují se u něj také zajímavé a rozšiřující myšlenky.

Jak bylo uvedeno v teoretické části, praktická část práce se neobejde bez alespoň minimálního množství teorie. Zodpovědnost za předání teoretických znalostí žákům je přesunuta na vyučujícího, který má k dispozici metodické listy. Tento dokument svou strukturou přesně kopíruje pracovní listy žáků. Učitel zde má řadu poznámek a dalších možností aktivit pro výuku. Zároveň jsou zde upozornění pro vyučující na nutnou teorii, kterou je potřeba žákům předat.

Celá práce je rozdělena do několika větších logických celků. Obrázek 12 zobrazuje rozdělení těchto logických celků. Modré části jsou nutné minimum pro sestavení robota, který bude schopen sledovat černou čáru. Poslední červené celky jsou jakousi nadstavbou, která z technického hlediska není nezbytně nutná, ale umožňuje rozvinutí a procvičení dalších informatických dovedností.

Obrázek 12 - Rozdělení listů

Metodické listy přesně kopírují listy pracovní. V první řadě obsahují rozšiřující informace k jednotlivým pracovním listům. Tyto informace slouží vyučujícímu k hlubšímu porozumění problému. Dále obsahují upozornění na obtížné části, nákresy, které je možné použít ve výuce a doporučení k aktivitám pro žáky. Metodické listy člení celou práci do jednotlivých vyučovacích hodin nebo dvouhodin.

2.2.1 Modelování podvozku a 3D tisk

Veškerá práce modelování podvozku byla prováděna v programu TinkerCad. S tímto programem se v práci žáci seznamují prvně, proto je několik prvních pracovních listů věnováno samotným základům práce a ovládání TinkerCadu. Učitelé jsou v rámci metodických listů na tuto skutečnost upozorněni a zároveň zde najdou krátký návod, jak s TinkerCadem pracovat.

2.2.1.1 Základní ovládání

Žák se zde seznamuje se samotným ovládáním prostředí. Učí se ovládat kameru, vkládat nové tvary a měnit jejich rozměr. Učitel by měl žákům ukázat více možností, jak mohou měnit velikost objektů včetně přesného zadávání čísla.

2.2.1.2 Deska podvozku

V rámci tohoto pracovního listu se žák poprvé setkává s technickým nákresem a kótováním. Učitel má za úkol vysvětlit žákům, jak toto kótování funguje a s jakými jednotkami se v nákresech pracuje. Záměrně jsou výkresy didakticky transformovány a nejde tedy 100% normované výkresy. Žák se tedy naučí podle výkresu vytvořit dva základní modely. Tuto dovednost pak bude moci ještě mnohokrát projevit v pozdější části práce.

2.2.1.3 Pravítko

Nástroj pravítko je klíčovým nástrojem TinkerCadu pro umisťování součástek v rámci prostoru. Žák se tedy učí pravítko umístit přesně na hranu již existujícího tvaru a následně pomocí pravítka umisťuje ostatní objekty. Zde je didakticky transformované kótování. V technických výkresech se díry většinou kótují ke svému středu, ale v programu TinkerCad by toto kótování bylo pro žáky matoucí, protože program neumí umisťovat středy objektů k jednotlivým kótám. Proto v nákresu jsou díry kótovány na svou hranu, nikoli na svůj střed.

2.2.1.4 Sloupky pro nosník I a II.

Nejdříve je žák seznámen s další klíčovou funkcí programu TinkerCad a tou je slučování objektů. Žák by měl zjistit, že je rozdíl, pokud sloučí dva objekty, anebo objekt a díru. Po této zkušenosti se přesouváme k modelování další části podvozku a tou je malý sloupeček, který bude sloužit pro třetí opěrný bod robota. Tyto sloupky jsou pro tento druh podvozku klíčové. Žák se zde učí, že složitější útvary jsou složeny z několika jednodušších. Na závěr pracovního listu se sloupky opět umístí pomocí pravítka na hlavní desku podvozku.

2.2.1.5 Třetí bod

Tento list předkládá žákovi první větší výzvu. Žák by se měl pokusit sám vymodelovat nosník, na který bude zavěšen třetí opěrný bod robota. Díra uprostřed nosníku není kótovaná, protože žák má zadáno, že její umístění je přesně v prostředku nosníku. Žák se proto musí seznámit s další funkcí TinkerCadu a tou je zarovnávání objektů. Na závěr se má žák zamyslet nad důvodem proč jsou díry v nosníku o 0,4mm širší než sloupky.

2.2.1.6 Sloupky pro desky

Základní ovládání TinkerCadu je úspěšně za námi. Další úkoly jsou již zadány po větších celcích. Dalším úkolem jsou opěrné sloupky pro desky, které ponesou elektronické součástky a powerbanku. Zde je velmi důležité pečlivě zkontrolovat umístění sloupků jinak by na ně následně modelované desky neseděly.

2.2.1.7 Spodní a horní deska

Tato dvojice pracovních listů je téměř identická. Liší se pouze v nákresu. Žák se nejdříve má zamyslet nad chybějícími kótami, tím se zajistí, že žáci nebudou pouze slepě následovat návod, ale také se pokusí přemýšlet jak velké a proč jsou kde umístěné otvory. Po zdárném doplnění hodnot již žák pouze vymodeluje tuto desku.

2.2.1.8 Pouzdro pro čidla

Jedna z nejsložitějších částí na vymodelování. Žáka nejprve vedeme k vymodelování tří kvádrů, z nichž jeden bude plný a další dva slouží k vytvoření díry. Tyto kvádry jsou následně naskládány na sebe a zasunuty v definovaném směru a hloubce do pevného kvádru. Pokud žák správně dodrží celý postup vznikne mu pouzdro s průzorem, do kterého bude možné vložit senzor sledování čáry.

2.2.1.9 Sledovač čáry

Sledovač čáry se skládá z právě vymodelovaných bloků a jejich roztečí. Opět žáka necháme, aby zkusil doplnit chybějící kóty a následně převedl nákres do 3D programu.

2.2.1.10 Drážka pro ultrazvuk

Poslední část k vymodelování je místo, kam bude umístěn ultrazvukový senzor. Na závěr je vše složeno dohromady.

2.2.1.11 Kola

Modelování kol je v TinkerCadu nesmírně složitý proces, a proto v rámci didaktické transformace je tento krok vynechán. Kola žák dostane již hotové od vyučujícího. Pro bystřejší

žáky je však možno poskytnout jim přiložený svg soubor, který je možné nahrát do TinkerCadu a vymodelovat tak loukoťová kola.

2.2.1.12 3D tisk

Pracovní listy již 3D tisk neobsahují. Ten je obsažen pouze v metodických listech. Je to z toho důvodu, že každá škola disponuje jiným druhem 3D tiskárny a práce se tak může u každého druhu lišit. V metodických listech je proto uveden krátký návod jak s tiskárnou pracovat. 3D tisk modelů zabere mnoho času, ale vzhledem k charakteru dalších pracovních listů je možné v práci pokračovat, protože pro sekci Elektronika v Tinkercadu není potřeba nic víc než počítač.

2.2.2 Zapojení elektroniky virtuálně

Předpokládáme, že žáci, kteří jsou vedeni k sestavení robota, se ještě nikdy předtím nesetkali s kontaktním polem a jeho zapojováním. Proto elektrická část práce je rozdělena na dvě části:

- Zjednodušený návrh v simulačním programu TinkerCad
- Samotné zapojení na kontaktním poli

Za pomoci pracovních listů se žák postupně seznamuje s absolutními základy elektroniky. Práce a první seznámení s elektronikou je pomocí programu TinkerCad, který nabízí jednak velmi jednoduchý způsob zapojování, zároveň zde objevujeme velkou názornost a posledním hlavním důvodem je ekonomičnost. Předpokládáme, že první zkušenosti žáků s elektronikou se neobejdou bez, lidově řečeno, odpálení součástek.

2.2.2.1 Kontaktní pole

První pracovní list se věnuje práci s kontaktním polem. Žák pochopí princip jeho vnitřního zapojení a způsob jakým se do kontaktního pole zapojují součástky.

2.2.2.2 Rozsvítíme diodu

Další list s názvem: „Rozsvítíme diodu," se věnuje první základní součástce a tou je LED dioda. Na tomto pracovním listu využíváme konstruktivistického způsobu výuky. Vycházíme z předpokladu, že žák již součástku zná, proto je na začátek zahrnuta aktivita, kdy v rámci celé třídy mají po skupinkách vymyslet co nejvíce zařízení, kde se mohou s LED diodou setkat. Tuto aktivitu vede vyučující a je zahrnuta v jeho metodickém listu. Žákův pracovní list tuto část neobsahuje.

Pracovní list se věnuje rozeznání anody a katody součástky a porozumění, že každý vývod se zapojuje k jinému pólu zdroje. Dále je žák veden k zapojení napájecího zdroje do obvodu a jeho přímého propojení s diodou. Závěr pracovního staví žáka před problém, který je potřeba vyřešit. Když žák vše zapojí, jak má, tak dioda shoří. Vyučující žákům připomene, proč si veškerá zapojení zkoušíme nejdříve v počítači a plynule naváže dalším pracovním listem.

2.2.2.3 Předřadný odpor

Tento list může být pro žáky poněkud obtížnější. Student se seznamuje s další elektrotechnickou součástkou a tou je rezistor. Součástka je mu představena jako věc, která mění elektrickou energii na energii tepelnou. Vyučující může žákům připomenout zákon zachování energie a provázat tak svůj předmět s předmětem fyzika.

Nejobtížnější částí tohoto listu však zůstává samotný výpočet předřadného odporu za pomoci Ohmova zákona. Na druhou stranu se žák setká se silným motivačním prvkem a tím je mu odměna v podobě svítící LED diody a jeho prvního funkčního elektrického obvodu.

2.2.2.4 Arduino

Sice by se i nadále mohl žák seznamovat s dalšími elektrotechnickými součástkami, ale pro účely skladby robota musíme postoupit dál, a tedy k samotnému Arduinu. V úvodu je třeba žáky seznámit s novými pojmy, se kterými se bude nově setkávat. Těmito pojmy jsou:

- Čip
- Procesor, mikroprocesor
- Pin
- Digitální
- Analogový

První žákova práce spočívá v prohlédnutí všech pinů Arduina a vyhledání napájecích pinů, které propojí do kontaktního pole místo baterií. Dále žáka vedeme k tomu, aby se zamyslel nad rozdílem mezi analogovými a digitálními piny. Jeho úkolem je tyto piny spočítat.

2.2.2.5 Hello world

Hello world je označení pro úplně první program, který programátor ve svém životě naprogramuje. Standardně jde o jednoduchý prográmek, který na obrazovku vypíše slova: „Hello world," v českém překladu znamenající: „Ahoj světe." Tímto programem často programátor vstupuje do nového světa, světa programování. I Arduino má svůj program Hello world. Avšak místo vypsání textu na obrazovku tento program začne blikat LED diodou.

Na úvod je před žáka předloženo schéma zapojení, které pro něj nemusí být na první pohled pochopitelné, protože ve schématu je Arduino znázorněno pouze jako obdélník se dvěma vyznačenými piny. Pro žáky může být zavádějící, že skutečné Arduino má pinů mnohem více. Učitel tak musí žákům objasnit, že jsou zakresleny pouze ty piny, které jsou pro zapojení důležité a jedná se pouze o zjednodušení nákresu.

Výhodou TinkerCadu je, že program Hello world je již defaultně nahrán ve virtuálním čipu, a tak stačí, když žák připojí svou LED diodu k pinu 13 a po spuštění simulace je mu motivační odměnou blikající dioda. Žákům je představen kód, který je v Arduinu nahrán a jsou vedeni k jeho jednoduché editaci.

2.2.2.6 Tlačítko

Další dva pracovní listy se věnují problematice digitálního vstupu Arduina. Cílem je, aby žák připojil k Arduinu tlačítko, kterým bude diodu ovládat. Hned v úvodu vedeme žáky k zamyšlení, jaký stav bude na vstupu procesoru, pokud bude tlačítko rozpojené. Žák se tak seznámí s novým pojmem a tím je: „nedefinovaný stav."

2.2.2.7 Znič nedefinovaný stav

Myšlenku nedefinovaného stavu neopouštíme ani na dalším listu. Zřejmě by každého napadlo, že nedefinovaný stav jednoduše zlikvidujeme připojením pinu k definovanému stavu, tedy k nule. Žák se tak setká s dalším pojmem a tím je: „zkrat." Žáka vedeme k tomu, aby si připomněl, co se stalo s LED diodou, pokud byla připojena přímo k napájení. Pracovní listy ho tak provádí postupně přes jednotlivé problémy a návodně mu radí, jak tyto problémy vyřešit.

Na závěr listu se tak dostáváme k úplnému zapojení tzv. „pull-down rezistoru**"**

2.2.2.8 Je tlačítko zmáčknuté nebo ne?

Elektrické zapojení tlačítka máme úspěšně za sebou. Nyní přichází pro žáky zatím asi nejsložitější část. Je potřeba nastavit pomocí algoritmu logiku, která bude diodu ovládat v závislosti na tlačítku.

Žáka nejdříve vedeme k tomu, aby zjistil, jaké bloky v kategorii vstup může využívat. Avšak tyto bloky není možné jednoduše připevnit k již existujícímu programu. Proto žáka vedeme dál do kategorie proměnné a vysvětlujeme mu, že i program si může různé informace zapamatovat a k tomu právě využívá proměnné. Nyní již je možné připojit zapamatovanou proměnnou ke zbytku programu.

Dál musíme žáky seznámit s podmínkami. Ty jsou základním stavebním kamenem pro jakékoli programování. Jednoduchou podmínkou je podmínka: „pokud." Žák si velmi rychle

osvojí její význam, protože znamená přesně to stejné, co v mluvené češtině a je tedy zřejmé, že celý algoritmus přepsaný do slov bude znamenat: „pokud zmáčknu tlačítko, tak rozsviť diodu.“

Každá podmínka používá pro své rozhodování operátory. Operátory jsou žákům již známé, ačkoli na první pohled to vypadá, že se s nimi ještě nikdy nesetkali. Jednoduše se jedná o porovnávací operátory „je menší, je větší,“ případně jejich další kombinace s operátorem rovná se.

2.2.2.9 Ovládej diodu

Na předchozí kartě se žák seznámil se základním fungováním podmínek. Nyní je potřeba vložit vše dohromady a sestavit výsledný program.

Žák je veden postupně k tomu, aby si uvědomil, jaké hodnoty procesor registruje, pokud je tlačítko stisknuté a pokud ne. Postupně si žák osvojuje překlad reálných zásahů člověk do binární logiky Arduina.

K hlubšímu ukotvení učiva je zde ještě druhý úkol a tím je přepsání programu tak, aby fungoval opačně, tedy pokud je tlačítko zmáčknuté, dioda je zhasnutá. Výhodou tohoto programování je, že si žák může svůj program ihned odzkoušet a získat tak zpětnou vazbu.

2.2.2.10 Světlo

Tato karta seznamuje žáka s novou součástkou a tou je fotorezistor. Ve skutečném zapojení se však později bude používat fototranzistor, ten bohužel v TinkerCadu není a fotorezistor je fototranzistoru svým chováním nejpodobnější.

Žák je nejprve veden k uvědomění si, že název fotorezistor nějakým způsobem pracuje se světlem. Dále se již podruhé žák setkává se zapojením pull-down rezistoru. Toto zapojení již blíže není vysvětlování, neboť se předpokládá, že žák již problematiku pochopil. Stejným způsobem jako u tlačítka žák postupuje i v případě připojení fotorezistoru.

2.2.2.11 Kolik je tu světla

Zde vstupuje do výuky nová věc, se kterou se žák dosud nesetkal a tím je použití sériového monitoru. Vyučující má za úkol vysvětlit velmi zjednodušeně žákům jakým způsobem převádí digitální procesor analogovou hodnotu do jedniček a nul.

Pomocí sériového monitoru žáci zjišťují, jaká hodnota je v proměnné uložena v případě, kdy je fotorezistor zatemněn a kdy je osvětlen.

Následně tyto hodnoty aplikují v kódu a za pomocí fotorezistoru ovládají výstupní LED diodu.

2.2.2.12 Stmívá se

Dosud žáci pracovali pouze s digitálním výstupem. Nyní se seznámí s pulzní šířkovou modulací jako možnost, jak digitálně simulovat analogové hodnoty.

Žák si musí nejprve uvědomit, že ne všechny piny Arduina disponují funkcí pulzní šířkové modulace. Následně do jednoho z těchto pinů připojí LED diodu.

Pro práci s PWM v TinkerCadu existují přímo vytvořené bloky, do nichž je možné vložit hodnotu od 0 do 255. Tento pracovní list nyní využije problémového úkolu, kdy žák řeší jak dostat hodnotu ze vstupu, která se pohybuje v rozmezí 0 až 1023 do hodnoty na výstup v rozmezí 0 až 255. Drobnou nápovědou je žák veden k tomu, aby našel číslo, kterým je potřeba hodnotu vydělit a tím zajistit nepřekročení hodnoty 255.

2.2.2.13 Intenzita světla

Předchozí nalezený výpočet je potřeba naprogramovat v TinkerCadu. Nyní žák zjišťuje, že programování není vždy pouze o tahání a práci s bloky, ale také o přemýšlení a matematice. Vypočtenou hodnotu žák nastaví na výstupní pin a může vyzkoušet pomocí simulace, že dioda postupně mění intenzitu svého svícení v závislosti na rezistoru.

Závěr pracovního listu je věnován krátkému zamyšlení, jakým způsobem můžeme využít dvojici součástek fotorezistor a led diodu k rozpoznání černé čáry, po které má robot ve výsledku jezdit. Tato část opět slouží jako motivační prvek pro žáky, aby neztratili ze zřetele celkový cíl projektu.

2.2.2.14 Virtuální robot (hardware)

Nyní již žák zná všechny potřebné součástky pro virtuální simulaci robota. Na úvod je potřeba správně zapojit veškeré hardwarové součásti. Zapojení je rozděleno na samostatné zapojení dvou čidel a samostatné zapojení dvou motorů.

2.2.2.15 Virtuální robot (řízení)

Tato karta vede žáka k tomu, aby si uvědomil, jakým způsobem funguje diferenční řízení. Žák má za úkol zapsat od předpřipraveného obrázku které čidlo co snímá a jakým způsobem na tuto informaci mají zareagovat kola robota. Pro výuku je vhodné tento úkol demonstrovat také s reálným robotem. Žáci si mohou vyzkoušet opatrně otočit koly a zjistit tak, kterým směrem se robot otočí pokud pohnou pouze pravým nebo pouze levým kolem.

2.2.2.16 Virtuální robot (diagram)

Žák se poprvé setkává s vývojovým diagramem. Vyučující provede postupně žáky doplněním chybějících dat ve vývojovém diagramu. Žák se zamýšlí nad tím, jaké má program

části a co vše je potřeba při pohybu robota vyřešit. Dále je zde zřejmé, že program pracuje v nekonečných cyklech.

2.2.2.17 Virtuální robot (software)

Po pochopení předchozího pracovního listu a zpracování vývojového diagramu je naní potřeba celý program přepsat do bloků TinkerCadu. Žák dostává nápovědu v podobě části kódu, který má v TinkerCadu doplnit.

Na závěr tohoto pracovního listu si žák vyzkouší simulaci svého robota. Musí být zřejmé, že při snímání čáry levým senzorem se musí otáčet vpřed pravé kolo, zatímco levé stojí, čímž dojde k zatočení robota. Pokud jsou obě čidla nad světlým povrchem pak robot může jet na plný výkon v před a tedy se točí obě kola dopředu.

2.2.3 Skutečné zapojení

Virtuální prostředí TinkerCadu nám neumožňuje další simulace nutné pro řízení robota. Navíc žáci se již seznámili s principy kontaktního pole, a proto se pracovní listy překlápí do další velké části a tou je zapojování reálných součástek a postupné sestavování výsledného robota.

V této části je třeba, aby vyučující dbal na bezpečnost práce, protože se zde pracuje s horkým tekutým cínem a velmi ostrým zalamovacím nožem. V metodických listekch je na tuto skutečnost vyučující upozorněn.

Častým úkazem v této a v následujících sekcích jsou testovací programy. Jde o krátké jednoduché programy, které jsou přílohou výukového materiálu, které otestují zda je vše zapojeno správně.

2.2.3.1 Základy

Prvním kamenem úrazu může být pro žáky skutečnost, že Arduino Nano vypadá jinak než Arduino, se kterým se setkali v prostředí TinkerCadu. Důvodem je skutečnost, že TinkerCad neposkytuje k použití součástku Arduino Nano. Avšak při bližším pohledu na tyto dvě součástky můžeme zjistit, že vývody jsou spolu vzájemně kompatibilní. Jinými slovy vývody procesoru Arduino Nano a Arduino Uno jsou naprosto identické, a proto můžeme program naprogramovaný pro Arduino Uno jednoduše přenést i do Arduina Nano.

2.2.3.2 Motory

Tato část práce je riziková, co se týče její bezpečnosti. V metodických listech je na to vyučující náležitě upozorněn. Pro připojení obou motorů je potřeba vodiče odizolovat a připájet

na dva měděné plíšky, které se na motoru nacházejí. Žákova práce je vedena pomocí návodných kreseb.

2.2.3.3 H-můstek

Nejprve je potřeba žákům vysvětlit, proč není možné motory připojit napřímo k Arduinu. Žák jako vždy, když se seznamuje s novou součástkou, je veden k prohlédnutí součástky a pojmenování jednotlivých vývodů. Vyučující vysvětlí, co jednotlivé vývody znamenají. Následně žák zapojí nejdříve motory a napájení obvodu podle nákresu.

2.2.3.4 Připojení a test

Pracovní list odkazuje žáka na práci, kterou již vytvořil v TinkerCadu. Je potřeba připojit ovládací piny H-můstku k Arduinu a pro jednoduchost budeme používat stejné piny, jaké již byly použity při práci ve virtuálním prostředí.

Žák přichází ve své práci k prvnímu testu. Test spočívá v nahrání programu do Arduina a zjištění, zda se děje správně to, co popisuje pracovní list. Pokud ne, žák je tímto způsobem obeznámen se zpětnou vazbou a veden k nápravě svého zapojení.

Před prvním testem je třeba, aby vyučující, který vede výuku vysvětlil, jakým způsobem je program do Arduina nahráván a s tímto postupem žáky seznámil.

2.2.3.5 Senzor čáry I.

Pracovní list seznamuje žáka se součástkou QRD1114, která je kombinací infračervené LED diody a fototranzistoru. Aby si žák tuto skutečnost uvědomil, jeho prvním úkolem je součástku rozebrat na části. Následně se pokusí tyto části pojmenovat. Pracovní list také opakuje pojmy, které již žák zná a těmi jsou anoda a katoda. Důležité je, aby si žák uvědomil, kam se který vývod připojuje.

Jednodušší na zapojení je LED dioda. S tou se již žák setkal a měl by tedy v tuto chvíli vědět, že je potřeba před LED diodu zapojit předřadný odpor. Opět je mu předložen nákres a požadavek na výpočet předřadného odporu.

2.2.3.6 Senzor čáry II.

Nyní žák zasadí oba senzory čáry do podvozku a zapojí nejprve LED diodu a následné také fototranzistory. Důležité je, aby žák zapojil levý a pravý fototranzistor do správných analogových pinů na Arduinu.

2.2.3.7 Test senzorů

Tento pracovní list se skládá ze dvou testů. Prvním je kontrola, zda jsou zapojeny správné infračervené LED diody. Infračervené světlo je sice pouhým okem neviditelné, ale senzory mobilních telefonů toto světlo často zachytávají a mohou jej zobrazit na displeji telefonu. Proto žáci mají za úkol vzít svůj mobilní telefon a podívat se přes fotoaparát na spodní část LED diod. Ve většině případů by mělo být vidět fialovo-bílé světlo.

Druhým testem je samotný test čidel. Opět se do Arduina nahraje testovací program a žák střídavě zakryje levé a pravé čidlo. Při zakrytí levého čidla by se měl otáčet levý motor a obráceně.

2.2.3.8 Prahová hodnota

S prahovou hodnotou se již žáci setkali při virtuálním zapojení robota, nyní ji musí zjistit v reálném zapojení. Pro tento úkol je opět přílohou materiálu předpřipravený program, který jednoduše zjišťuje hodnoty na čidlech a odesílá je pomocí sériové linky do počítače. Žáci střídavě pokládají robota čidly na bílý a černý povrch a sledují, jak se hodnoty v závislosti na povrchu mění. Zaznamenají si nejvyšší a nejnižší hodnoty čidel a následně z nich určí přibližně středovou hodnotu mezi těmito čísly.

Zjištěnou prahovou hodnotu žáci upraví v již existujícím programu, který vytvořili v TinkerCadu.

2.2.3.9 Program Kačenka

Nic nového již není potřeba programovat. Vystačíme si s programem, který jsme použili při virtuální simulaci robota. Tento pracovní list žáky provede pouze jeho stažením z TinkerCadu a nahráním do reálného Arduina. Výsledkem by měla být radost na tvářích žáků z prvního funkčního prototypu robota, který je schopný sledovat černou čáru.

2.2.3.10 Robot nejede

Může se stát, a pravděpodobně se to stane, že se některé roboty nerozjedou, nebo se začnou chovat podivně. Pro tyto případy je vložen pracovní list, který žáky navede směrem, kterým by se mohl nacházet jejich problém.

2.2.4 Skutečné zapojení – senzor překážek

Další části práce jsou jakousi nadstavbou, protože pro samotný pohyb robota po čáře není nezbytně nutné další části připojovat. Další pracovní listy je možné využít například pro zvídavější žáky, nebo v rámci volnočasových kroužků.

2.2.4.1 Ultrazvukový senzor

Žáky opět seznamujeme s novou součástkou a již tradičně začínáme popisem jednotlivých vývodů. Vzhledem k tomu, že ultrazvukový senzor je velmi jednoduchý zapojit přecházíme přímo k nákresu zapojení. Opět se jedná o vstupní zařízení, a tak žák vytváří proměnnou, do které se bude hodnota vstupu zapisovat. Následně je veden k samostatnému prohledání a prostudování funkce určené pro ultrazvukový senzor.

Na učitele je přenesen požadavek o vysvětlení principu ultrazvukového senzoru a uvedení dalších příkladů například radar pro měření rychlosti nebo užití echolokace u netopýrů.

2.2.4.2 Pozor překážka

Po připojení senzoru je potřeba lehce upravit program pro řízení robota. Úkolem studentů je přidat podmínku, která v případě, že je překážka dál než 10cm od robota zajistí, pokračování v pohybu po čáře. Žáci tedy musí veškerý kód, který se týká řízení robota přesunout dovnitř podmínky.

Na konec programu je přidána podmínka, že v opačném případě má robot zastavit. V rámci metodických listů je učitel upozorněn na možnost využití podmínky „když jinak“, která odpovídá programátorskému „if else“. Tato podmínka je však z hlediska didaktiky složitější na pochopení, a proto se pracuje pouze s jednoduchými podmínkami.

2.2.5 Skutečné zapojení - 4 čidla

Přidání dalších dvou čidel zajistí přesnější a plynulejší řízení robota při pohybu po čáře. Čidla se zapojují stejným způsobem jako první dva kusy, hlavní část této sekce se tak věnuje naprogramování nové logiky řízení. Do budoucna je možné rozšířit práci o řízení pomocí PID-regulace.

2.2.6 Skutečné zapojení – enkodér

Enkodér umožňuje sledovat úhel natočení kola, případně měření rychlosti otáčení. Princip fungování enkodéru jsme popsali v kapitole 1.4.6. Karty pro programování enkodéru jsou označeny červeným rámečkem, což znamená, že jsou jednou z nejtěžších úloh k sestavení robota. Nepočítá se s tím, že by enkodér sestavovali všichni žáci, je zde pouze pro umožnění nadanějším a zvídavějším žákům další práce.

2.3 Možné rozšíření práce

Edukační robot by měl seznámit žáky s absolutními základy robotiky. Nelze očekávat, že s robotem, který je sestavený podle vytvořeného návodu by bylo možné se účastnit

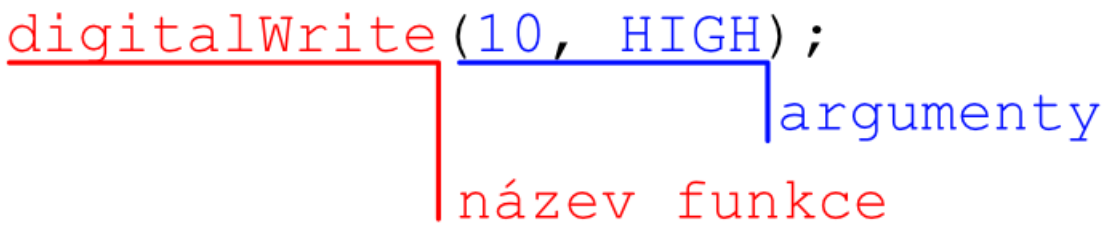
robotických soutěží s nějakým větším úspěchem. Proto zde v této kapitole uvádíme možnosti dalšího rozšíření výukového materiálu, které umožní pokročilejší práci s robotem a samozřejmě dosažení lepších výsledků.

2.3.1 Programovací jazyk

V práci jsme všechny algoritmy tvořili pouze pomocí bloků v programu TinkerCad, avšak pro složitější projekty a efektivnější práci s Arduinem je lepší psát kód přímo v jazyce Wiring. V této kapitole krátce popíšeme jakým způsobem by bylo možné rozšířit výuku o psaní kódu přímo v Arduino IDE.

Dobrým základem, na kterém by mohla stavět konstruktivisticky pojatá výuka je právě znalost blokového programování v TinkerCadu. Výhodou je, že žák si v TinkerCadu může zapnout přímo zobrazení kódu a porovnávat, jak se kód mění při používání určitých bloků. Touto metodou pokus omyl se snaživý žák jistě naučí programovat, ale možná neporozumí, proč se přidávají různé řádky kódu. Výuku samotného kódu by bylo vhodné rozdělit do větších ucelenějších částí.

Základem pro programování je porozumění struktuře jazyka. Konkrétně porozumění



Obrázek 14 - části funkcí

funkcím setup a loop, viz kapitola 1.2.4. Další částí je používání funkcí, neboli jejich volání. Funkce již žák zná a používá, proto můžeme výuku uvést na již známé funkci digitalWrite(); Tato funkce je funkcí beznávratové hodnoty, protože nám nevrací žádný výsledek. Pouze zapíše požadovanou hodnotu na požadovaný pin. Funkce s návratovou hodnotou jsou typicky funkce

Obrázek 13 - funkce s návratovou hodnotou

pro čtení vstupů. Jejich použití vyžaduje nějakou proměnnou, do které se navrácená hodnota zapíše. Čímž se plynule dostáváme k proměnným.

Proměnné jsou různých datový typů. Můžeme si je představit jako šuplík, do kterého můžeme uložit nějaké věci. Ale jsou zde šuplíky pouze na různé konkrétní typy věcí, které spolu nesmíme míchat. Tak by mohlo znít zjednodušené vysvětlení pro žáky. Základní typy proměnných, které by žáci měli znát jsou int, char a string. Ty odpovídají již známým datům a to číslo, znak a text.

Po proměnných je důležité žáky naučit podmínkám. Základní podmínku if již používali v TinkerCadu. Stačí jim pouze objasnit syntaxi v jazyce Wiring. Dále je vhodné seznámit žáky s dalšími typy podmínek jako if else a switch.

Novou věcí, kterou žáci v TinkerCadu nepoužívali jsou cykly. Ty můžeme začít vyučovat přímo v TinkerCadu, kde žáci pochopí funkci cyklů, následně je možné učit syntaxi přímo v jazyce Wiring.

Tento programovací aparát by měl stačit jako základ pro většinu práce s Arduinem. Dalším krokem rozvíjení programovacích schopností jsou tvorba vlastních funkcí, a to jak návratových, tak beznávratových. Dalšími rozšiřujícími možnostmi je používání define konstrukcí, které umožňují softwarové přepojování součástek, bez nutnosti zásahu do fyzického zapojení.

Doporučujeme v případě vyučovaní programovacího jazyka Wiring poskytnout žákům základní přehled syntaxe ideálně na jedné A4 natištěné na tvrdém papíře. Dále je vhodné žáky nasměrovat na referenční manuál jazyka Wiring, který je bohužel pouze v Angličtině, ale obsahuje kompletní dokumentaci celého jazyka. Mnoho problémů s programování, a v případě Arduina to platí dvojnásob, se dá vyřešit hledáním na internetu, protože Arduino má velmi silnou uživatelskou základnu. Vedení žáků k samostatnému hledání a řešení problémů rozvíjí jejich schopnost řešení problémů i v praktickém životě.

2.3.2 PID-regulace

PID regulace je pokročilý systém řízení, který je možné aplikovat na edukačního robota. Obecný princip spočívá ve zjištění skutečné hodnoty a jejím porovnání s hodnotou žádanou. V kontextu robota to znamená, že žádaný stav je, aby obě čidla robota byla na čáře, v okamžiku kdy jedno nebo druhé čidlo se od čáry vzdaluje, pak je to považováno za chybu. Velikost chyby se určí ze vzorce:

$$Err = Č - H_p$$

Rovnice 4 - výpočet chyby

Kde Č znamená právě naměřená hodnota čidla a H_p představuje požadovanou hodnotu.

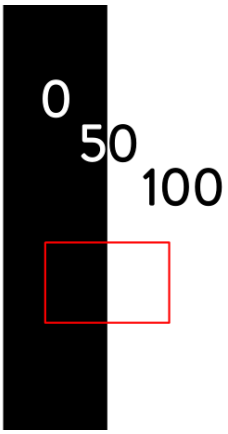
Je zřejmé že chyba může nabývat kladné i záporné hodnoty. To nám značí, zda se čidlo vychýlilo vpravo do bílé oblasti nebo vlevo do černé oblasti čáry. Pokud tuto hodnotu vynásobíme konstantou k_p dostáváme první z parametrů PID regulátoru P.

$$P = k_p . Err$$

Rovnice 5 - parametr p

2.3.2.1 Příklad

Představme si robota, který je řízen jedním čidlem sledujícím čáru na její pravé straně. Pokud je senzor přímo nad čárou neodráží se zpátky do senzoru ideálně žádné světlo. Pokud je senzor naopek mimo čáru, pak snímá 100% veškerého světla. Naší požadovanou hodnotou je tedy 50. Chceme, aby se robot pohyboval tak, aby jeho čidlo bylo neustále přesně na hraně černé a bílé barvy.



Obrázek 15 - senzor nad čárou

Ve vyváženém stavu vychází velikost chyby nulová. Není tak potřeba žádné regulace systému a i P parametr zůstane tedy nulový.

$$Err = 50 - 50 = 0$$

Rovnice 6 - nulová chyba

Pokud se čidlo vychýlí doprava od čáry začne snímat hodnoty vyšší než 50 a chyba tak poroste do kladných čísel. Potom musíme pravému motoru přidat výkon a levému naopak ubrat. Velikost tohoto výkonu bude závislá právě na velikosti chyby. Modelově čidlo snímá hodnotu 80 výpočet pak bude následující:

$$Err = 80 - 50 = 30$$

$$P = k_p . Err$$

Poprvé se nám zde objevuje konstanta k_p, tuto konstantu volíme a určujeme, jak velký vliv bude mít chyba na regulaci. Výkon pravého a levého motoru pak upravíme podle těchto vzorců:

$$M_P = v_0 + P$$

$$M_L = v_0 - P$$

Rovnice 7 - úprava rychlosti motorů parametrem P

Dalším parametrem je integrační člen. Integrační člen v sobě neustále kumuluje veškeré informace o chybách a tyto informace spolu sčítá. Vzorec pro integrační člen je tedy:

$$I_{SUM} = I_{SUM} + Err$$

$$I = k_i . I_{SUM}$$

Rovnice 8 - integrační člen

Posledním členem je derivační člen. Ten počítá s chybou předchozí.

$$D = k_D . (Err - Eold)$$

$$Eold = Err$$

Rovnice 9 - derivační člen

Z těchto tří parametrů se pak sečte P+I+D a tímto parametrem se upraví výkony motorů stejně jako v Rovnice 7.

3 Výzkumná část

Vzhledem ke Covidové situaci v posledních dvou letech, nebylo možné v dostatečné míře odzkoušet vytvořený výukový materiál. Proto jsme oslovili několik učitelů i žáků, aby se na předložený materiál podívali a poskytli svůj názor, zda by si uměli představit výuku podle těchto materiálů. Žáci jsou studenti prvního ročníku střední školy netechnického oboru.

3.1 Vyjádření žáků

Na celém dokumentu se mi líbí ilustrace – všude jde vidět, jak se má akce vykonat a ten malý robot s bublinami textu mi přijde roztomilý. Myslím si, že bych podle tohoto dokumentu byla schopna vytvořit alespoň nějaký kus práce. Mám již menší zkušenosti s programováním a bavilo by mě podle tohoto dokumentu práci vytvářet. Celý popis mi přijde vcelku jednoduchý a stručně popsaný. Když celkově shrnu svůj názor, práce se mi líbí.

Milada (16 let)

První, co mě zaujme je jednoduchá příjemně působící animace. Líbí se mi shrnutí na kartách. Těmto věcem moc nerozumím, ale přijde mi, že je to srozumitelně a stručně sepsané. Myslím, že takto hezky působící stručný návod usnadní práci. Líbí se mi vložení obrázků pro představivost. Asi bych s nějakou pomocí byla schopna úkol dokončit.

Petra (15 let)

Líbí se mi, jak je to zpracované a přehledné. Stránky mají hezký design. Myslím si, že by práci podle tohoto návodu zvládli všichni.

Tereza (15let)

Ačkoliv nejsem zrovna technicky zdatná, je mi velmi příjemná grafická úprava a vcelku jednoduché a neodborné pojmy. Krátké a stručné odstavce mi také napomohly k lepší orientaci a chápání tématu. Líbí se mi všude jasné nadpisy s příkazem jako: zapamatuj si, nezapomeň, pozor, zamysli se apod. Myslím, že s pomocí učitele a s touto prací bych byla schopná postavit robota, nebo se k tomu aspoň přiblížit.

Laura (17 let)

Velmi se mi líbí krátký a stručný popis. K pochopení mi velmi pomáhají hezké a dobře pochopitelné ilustrace a také to že je ukázaný správný a špatný postup. Myslím si, že bych zvládla robota vytvořit.

Natálie (16 let)

Sice nejsem úplně typ člověka, který se zajímá o technologie atd. Ale je to podle mě přehledné. Kartičky se mi moc líbí. Líbí se mi také ten robot, který tam přidává ty komentáře. A také se mi líbí že tam jsou přidané i řešení.

Eliška (15 let)

3.2 Vyjádření učitelů

Pracovní listy se mi zdají dobře srozumitelné a logicky uspořádané. S jejich využitím si určitě dokážu představit při hodinách fungovat a s žáky je i případně rozebrat dle potřeby. Grafické zpracování by ještě chtělo doladit. Líbí se mi jejich stručnost a zároveň výstižnost. V případě metodických listů, když jsem je poprvé otevřel, měl jsem problém se trochu mezi všemi nadpisy orientovat. Moc dobře jsou popsané jednotlivé kroky a případné aktivity, které mi dávají v celém kontextu smysl a jsou výstižné. Osobně bych na konci kapitol uvítal dodatečné obrázky, či pouze shrnutí všech obrázků u dané kapitoly pro lepší orientaci. Pro aprobovaného učitele věřím, že jak je metodika postavená, tak nebude problém s pochopením celého smyslu látky a zapojení jednotlivých dílů. Nicméně častý případ na ZŠ je, že ICT učí kde kdo a pak bych věřil, že dotyčný bude muset chvilku přemýšlet co a jak.

Abych to celé shrnul, materiály se mi líbí a přijdou mi smysluplné. Co bych ještě upravil je grafická stránka a doplnil bych obrázky k jednotlivým kapitolám. Rozhodně mně to přijde jako dobrá práce.

Bc. Ondřej Falta (učitel SŠ)

Po zběžném zhlédnutí práce jsem si vzpomněl na svá studia na průmyslové škole, což je již 30 let zpět. Bohužel od té doby jsem v elektrice a programování nic nedělal. Obávám se, že nejsem ten správný člověk, který by měl tyto věci učit. Vedu sice dva technické kluby, ale pracujeme hlavně se dřevem, kovem a plexisklem, případně nějaké údržbářské práce.

Petr Appl (vedoucí SVČ)

3.3 Shrnutí

Z odpovědí učitelů je zřejmé, že mladší učitelé, kteří sami prošli technickým oborem, nebo mají blízko k elektřině by si s vytvořeným materiálem poradili velmi snadno. Naopak starší učitelé s větší praxí se s programováním a mikročipy setkávají poprvé a je pro ně, vzhledem k jejich věku, již mnohem obtížnější se učit novým věcem.

Vyjádření žáků je ve směs pozitivní a odráží se v něm standardní rozdělení třídy na několik snaživých jedinců, velkou skupinu, která nezastává ani pozitivní ani negativní stanovisko a několik dětí, které nebaví vůbec nic a každá další nová věc je pro ně na obtíž.

4 Závěr

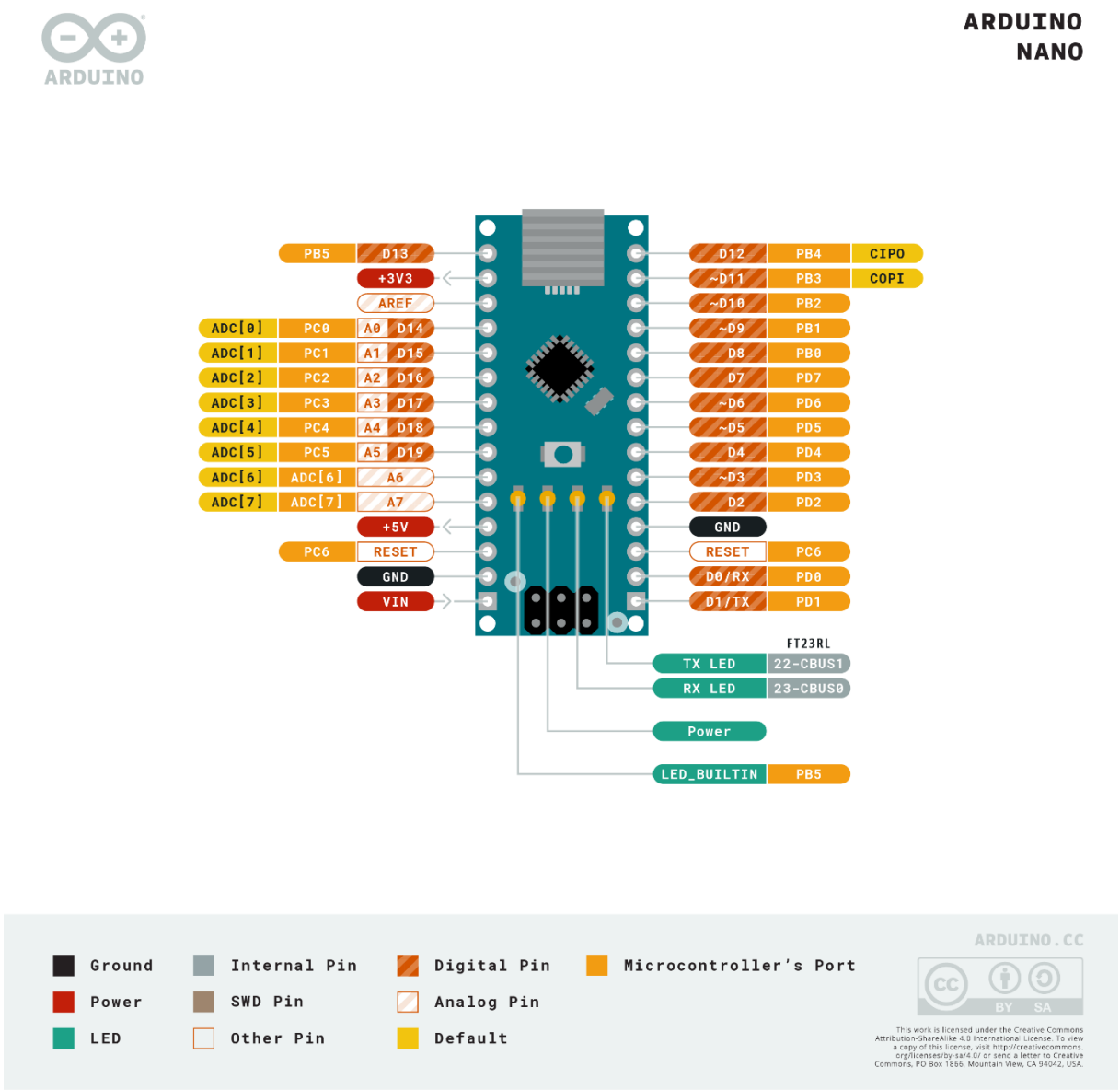
Cílem práce bylo vytvoření edukačního robota s enkodérem spolu s pracovními a metodickými listy tak, aby bylo možné zopakovat výuku ve školním prostředí. Cíle bylo dosaženo v podobě dvou prototypů, obou plně funkčních. Z vyjádření vyučujících i žáků plyne, že si umí dobře představit použití poskytnutých materiálů ve výuce. V praxi se stále setkáváme s množstvím učitelů informačních technologií, kteří nemají rádi novou informatiku a její robotickou část a je pro ně velmi obtížné tyto očekávané výstupy dle RVP plnit. Avšak jak bylo v práci několikrát zmíněno, je vidět, že výuka programování a algoritmizace má přesah i do skutečného života v podobě schopností dělit velký problém na snáze řešitelné části, hledat řešení tam, kde intuice říká, že žádné řešení neexistuje a využít práci výpočetní techniky na plnění repetitivních úkolů. Tyto kompetence jsou dnes velmi žádoucí a jejich rozvíjení u budoucí generace taktéž.

Jak je vidět, robot samotný není cílem výuky, ten pouze odvádí žákovu pozornost od nutnosti dalšího studia a výuky (v kladném slova smyslu). Funguje zde jako motivační prvek. Tento motivační prvek je zde velmi důležitý, protože konstrukce takového zařízení vyžaduje mnoho trpělivost a vynaloženého úsilí. Výsledný robot je naštěstí tak silným hnacím motorem, že žáci jsou schopni pro takový zisk prakticky všeho.

Během práce se žáci naučí mnoho technických dovedností, například práci s 3D modely, základům elektroniky, programování a algoritmizaci. Nabytí těchto dovedností může některé žáky přesvědčit o jejich směřování technickým směrem, které je v dnešní společnosti velmi žádoucí, neboť mnoho středních technických škol zeje prázdnotou. Splněním vytvořených pracovních listů si vyučující může odškrtnout mnoho požadovaných výstupů ze svého RVP. Práce tak splňuje svůj účel.

5 Přílohy

Příloha č. 1 – Popis pinů Arduino Nano [16]



6 Seznam obrázků, rovnic a tabulek

8 Seznam použitých zkratek

3D	označení pro třírozměrný předmět
FN	fakultní nemocnice
PID	proporcionální, integrační, derivační (regulace)
RVP	rámcový vzdělávací program
RVP ZV	rámcový vzdělávací program pro základní vzdělávání
SŠ	střední škola
SVČ	středisko volného času

9 Literatura

1. PRŮCHA, Tomáš. *Podpora výuky technických předmětů na středních školách za využití otevřené robotické platformy* [online]. České Budějovice, 2015 [cit. 2022-02-13]. Dostupné z: https://theses.cz/id/05dasj/. Diplomová práce. Jihočeská univerzita v Českých Budějovicích, Přírodovědecká fakulta. Vedoucí práce Mgr. Jiří Pech, Ph.D.
2. MALÝ, Martin. *Hradla, volty, jednočipy: úvod do bastlení*. Praha: CZ.NIC, z.s.p.o., 2017. CZ.NIC. ISBN 978-80-88168-23-2.
3. VODA, Zbyšek. *Průvodce světem Arduina*. Vydání druhé. Bučovice: Martin Stříž, 2017. ISBN 978-80-87106-93-8.
4. STŘÍTESKÝ, Ondřej, Josef PRŮŠA a Martin BACH. *Základy 3D tisku s Josefem Průšou* [online]. Praha: Prusa Research, 2019 [cit. 2022-02-14]. Dostupné z: https://www.prusa3d.com/cs/stranka/zaklady-3d-tisku-s-josefem-prusou_490/
5. ASIMOV, Isaac. *Já, robot*. Přeložil Oldřich ČERNÝ, přeložil Alexandr KRAMER, přeložil Zuzana MEYEROVÁ. Praha: Triton, c2012. Trifid (Triton). ISBN 978-80-7387-491-9.
6. *Rámcový vzdělávací program pro základní vzdělávání* [online]. Praha: MŠMT, 2021 [cit. 2022-02-15]. Dostupné z: http://www.nuv.cz/file/4982_1_1/
7. MAŇÁK, Josef a Vlastimil ŠVEC. *Výukové metody*. Brno: Paido, 2003. ISBN 80-7315-039-5.
8. KOMENSKÝ, Jan Amos. Velká didaktika. Druhé vydanie. Bratislava: Slovenské pedagogické nakladatelstvo, 1991. ISBN 80-08-01022-3.
9. ČERNÝ, Michal. Výukoví roboti: nástroj pro rozvoj algoritmického myšlení. Metodický portál: Články [online]. 25. 08. 2015, [cit. 2022-03-22]. Dostupný z WWW: <https://clanky.rvp.cz/clanek/19905/VYUKOVI-ROBOTI:-NASTROJ-PRO-ROZVOJ-ALGORITMICKEHO-MYSLENI.html>. ISSN 1802-4785.
10. BRDIČKA, Bořivoj. Informatické myšlení jako výukový cíl. Metodický portál: Spomocník [online]. 22. 04. 2014, [cit. 2022-03-22]. Dostupný z WWW: <https://spomocnik.rvp.cz/clanek/18689/INFORMATICKE-MYSLENI-JAKO-VYUKOVY-CIL.html>. ISSN 1802-4785.
11. SVOBODA, Aleš. H-můstek modul L9110S. ECLIPSERA s.r.o. Distributor pro ČR., 2016.
12. ČERNÝ, Michal. 3D tisk ve školním prostředí. Metodický portál: Články [online]. 08. 06. 2015, [cit. 2022-03-24]. Dostupný z WWW: <https://clanky.rvp.cz/clanek/19903/3D-TISK-VE-SKOLNIM-PROSTREDI.html>. ISSN 1802-4785.
13. KOZÁK, Petr. Nástroje pro výuku programování. Metodický portál: Spomocník [online]. 25. 05. 2015, [cit. 2022-03-24]. Dostupný z WWW: <https://spomocnik.rvp.cz/clanek/20019/NASTROJE-PRO-VYUKU-PROGRAMOVANI.html>. ISSN 1802-4785.

14. HROMÁDKA, Zdeněk. Učitel jako tvůrce výukových materiálů. Metodický portál: Články [online]. 23. 04. 2020, [cit. 2022-03-24]. Dostupný z WWW: <https://clanky.rvp.cz/clanek/22362/UCITEL-JAKO-TVURCE-VYUKOVYCH-MATERIALU.html>. ISSN 1802-4785.
15. How Rotary Encoder Works and How To Use It with Arduino. *How to mechatronics* [online]. Amazon, c2022 [cit. 2022-04-01]. Dostupné z: https://howtomechatronics.com/tutorials/arduino/rotary-encoder-works-use-arduino/
16. Arduino Nano. *Arduino.cc* [online]. c2022 [cit. 2022-04-01]. Dostupné z: https://docs.arduino.cc/hardware/nano
17. *Robotika pro střední školy: Programujeme Arduino*. České Budějovice: Jihočeská univerzita v Českých budějovicích, 2020. ISBN 978-80-7394-786-6.
18. „Želvy“ v Motole vozí léky nebo jídlo. Nemocnice zvažuje, že pro ně vybuduje další dvě trasy. *ČT 24* [online]. Praha: Česká televize, 2017 [cit. 2022-04-01]. Dostupné z: https://ct24.ceskatelevize.cz/regiony/2200726-zelvy-v-motole-vozi-leky-nebo-jidlo-nemocnice-zvazuje-ze-pro-ne-vybuduje-dalsi-dve
19. ČAPEK, Karel. *R.U.R.: rossum's universal robots*. Praha: Artur, 2004. Edice D. ISBN 80-86216-46-2.
20. Vývojové diagramy. *Základy informatiky pro střední školy* [online]. České Budějovice: Jihočeská univerzita v Českých Budějovicích, 2020 [cit. 2022-04-01]. Dostupné z: https://popelka.ms.mff.cuni.cz/~lessner/mw/index.php/U%C4%8Debnice/Algoritmus/V%C3%BDvojov%C3%A9_diagramy