STREDNÁ PRIEMYSELNÁ ŠKOLA ELEKTROTECHNICKÁ

HÁLOVA 16, 851 01 BRATISLAVA

Arduino radar

Komplexná odborná maturitná práca

Č. odboru: <číslo a názov súťažného odboru>

Alex Szabó

Bratislava

2026

Ročník štúdia: IV.D

STREDNÁ PRIEMYSELNÁ ŠKOLA ELEKTROTECHNICKÁ

HÁLOVA 16, 851 01 BRATISLAVA

Arduino radar

Komplexná odborná maturitná práca

Č. odboru: <číslo a názov súťažného odboru>

Alex Szabó

Bratislava

2026

Ročník štúdia: IV.D

Ing. Dominik Zatkalík, PhD.

Čestné vyhlásenie

Vyhlasujem, že prácu stredoškolskej odbornej činnosti na tému Arduino radar som vypracoval samostatne, s použitím uvedených literárnych zdrojov. Prácu som neprihlásil a ani neprezentoval v žiadnej inej súťaži, ktorá je pod gestorstvom MŠVVaM SR. Som si vedomý dôsledkov, ak uvedené údaje nie sú pravdivé.

.....

V Bratislave, <dd. mm. rrrr> Alex Szabó

Poďakovanie

Rád by som sa touto cestou poďakoval svojmu konzultantovi Ing. Dominikovi Zatkalíkovi, PhD. za prístup a odborné rady.

Obsah

Zoznam skratiek, značiek a symbolov

Obr. - Obrázok

GUI – Graphical User Interface

UART - Universal Asynchronous Receiver/Transmitter

IDE – Integrated Developer Environment

ADC – Analog-to-digital Converter

PWM – Pulse-width Modulation

Zoznam tabuliek, grafov a ilustrácií

<Zoznam skratiek, značiek a symbolov>

0 ÚVOD

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet commodo magna eros quis urna.

Nunc viverra imperdiet enim. Fusce est. Vivamus a tellus.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Proin pharetra nonummy pede. Mauris et orci.

Aenean nec lorem. In porttitor. Donec laoreet nonummy augue.

Suspendisse dui purus, scelerisque at, vulputate vitae, pretium mattis, nunc. Mauris eget neque at sem venenatis eleifend. Ut nonummy.

1 ZÁKLADNÉ PRINCÍPY RADAROVÝCH SYSTÉMOV

Radar (z angl. **ra**dio **d**etection **a**nd **r**anging), alebo rádiolokátor je systém, ktorý využíva rádiové vlny na určenie vzdialenosti, smeru (azimutálneho a elevačného uhla) a radiálnej rýchlosti objektov vzhľadom na miesto položenia konkrétneho radaru. [1], [2], [3]

1.1 FYZIKÁLNE ZÁKLADY RADAROV

Princíp fungovania radarového systému je jednoduchý na pochopenie, aj keď jeho teoretický základ je pomerne zložitý. Avšak pochopenie teórie je základom pre efektívne používanie a obsluhu radarového zariadenia. Opiera sa o tri základné fyzikálne zákony, ktoré sa týkajú šírenia a interakcie elektromagnetických vĺn (zvyčajne v mikrovlnnej oblasti spektra). [4]

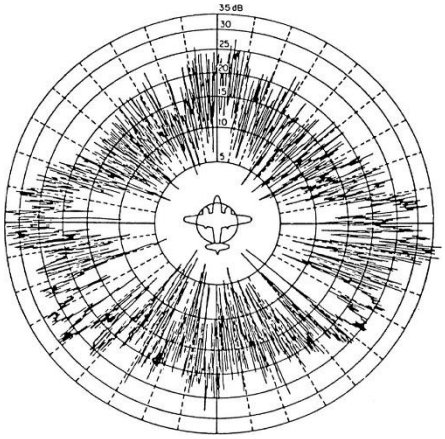
1.1.1 ODRAZ ELEKTROMAGNETICKÝCH VĹN

Ak elektromagnetická vlna narazí na elektricky vodivé teleso (napríklad lietadlo, loď alebo dokonca kvapky dažďa), jej energia sa odrazí. Radar je konštruovaný tak, aby zaznamenal túto odrazenú vlnu, nazývanú aj **ozvena** (echo). Prítomnosť ozveny signalizuje prekážku v smere šírenia vlnenia. [4]

Intenzita odrazeného signálu sa vyjadruje veličinou nazývanou **efektívna odrazová plocha (RCS – Radar Cross Section)**. Jej hodnota je ovplyvnená mnohými faktormi, z ktorých hlavné sú:

- **Veľkosť a geometrický tvar cieľa**,
- **Uhol pohľadu na cieľ**, ktorý určuje, ktorá časť povrchu cieľa je ožarovaná elektromagnetickou vlnou,
- **Pracovná frekvencia radaru**, presnejšie pomer medzi vlnovou dĺžkou lokátora a charakteristickými rozmermi cieľa,
- Elektrické vlastnosti materiálu, z ktorého je cieľová štruktúra vyrobená.

Vplyv týchto faktorov je komplexný, a preto je potrebné ich posudzovať spoločne. [5]



Obr. 1 – Efektívna odrazová plocha lietadla typu B-26
(Zdroj: www.radartutorial.eu, rok: 2024)

Existuje mnoho vzorcov na výpočet efektívnej odrazovej plochy jednoduchých telies (napr. guľa, valec), ale keďže v praxi má väčšina radarových cieľov zložitý geometrický povrch, na určenie ich efektívnej odrazovej plochy sa používa vzorec:

$$\sigma = \frac{4\pi.r^2.S_r}{S_t} \quad kde \quad r \text{ — polomer ekvivalentného odrazníka,}$$

S_r — hustota výkonového toku dopadajúcej vlny v polohe cieľa,

S_t — hustota výkonového toku rozptýlenej vlny na radarovej anténe.

[5]

1.1.2 KONŠTANTNÁ RÝCHLOSŤ ŠÍRENIA

Elektromagnetické vlny sa vo vákuu šíria približne rýchlosťou svetla. V reálnom prostredí je ale jedno, či sa tu počíta s rýchlosťou 3·108 m/s alebo s 300 000 km/s, alebo či je rýchlosť svetla udávaná veľmi presne 299 792 458 m/s. Pri výpočte ale musíme vždy používať rovnakú veľkosť. Vďaka tejto konštantnej rýchlosti šírenia je možné presne určiť **vzdialenosť** cieľa (lietadiel, lodí, vozidiel) meraním času *t* medzi vyslaním signálu a prijatím jeho ozveny. Vzťah pre výpočet vzdialenosti *R* je daný:

$$R = \frac{c \cdot t}{2} \quad kde \quad c = \text{rýchlosť svetla}$$

t = nameraná doba prejazdu

Faktor 2 v menovateli zohľadňuje skutočnosť, že signál musí prejsť vzdialenosť *R* dvakrát (cesta k cieľu a späť). [4], [6]

1.1.3 PRIAMOČIARE ŠÍRENIE A SMEROVANIE

Predpokladá sa, že šírenie elektromagnetických vĺn vo frekvenčnom rozsahu radarov je priamočiare. Pomocou špeciálnych smerových antén je možné elektromagnetické vlny zväzovať v určitom smere. Sledovaním smeru, z ktorého prichádza ozvena, je možné určiť **uhlové súradnice** cieľa, t. j. azimutu (uhol v horizontálnej rovine) a elevačného uhla (uhol vo vertikálnej rovine). [4]

2 RADAROVÉ KOMPONENTY A PRINCÍP ICH ČINNOSTI

Typický impulzný radarový systém je aktívne elektronické zariadenie, ktoré je navrhnuté na presné a rýchle meranie parametrov cieľa v priestore. Skladá sa z viacerých kľúčových funkčných blokov a modulov, ktoré pracujú v synchronizovanom cykle.

2.1 VYSIELAČ

Radarový vysielač generuje krátky vysokofrekvenčný impulz energie s vysokým výkonom. Tieto impulzy majú veľmi krátke trvanie (typicky v mikrosekundách), aby sa dosiahla dobrá rozlišovacia schopnosť v diaľke. Vysielač musí mať nasledujúce technické a prevádzkové vlastnosti:

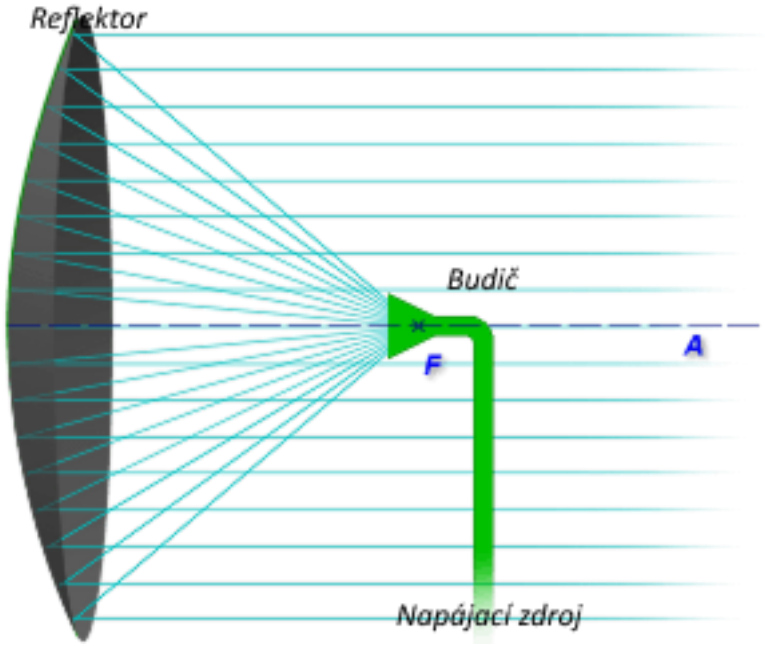
- Musí byť schopný generovať požadovaný stredný rádiofrekvenčný výkon a požadovaný špičkový výkon,
- Musí mať vhodnú rádiofrekvenčnú šírku pásma,
- Musí mať vysokú rádiofrekvenčnú stabilitu, aby spĺňal požiadavky na spracovanie signálu,
- Musí byť ľahko modulovateľný, aby spĺňal požiadavky na návrh tvaru vlny,
- Musí byť efektívny, spoľahlivý a ľahko udržiavateľný. Životnosť a náklady na výstupné zariadenie musia byť prijateľné.

Kľúčovými komponentmi sú vysokovýkonné elektrónky ako **magnetrón** (používaný v starších/jednoduchších radaroch, lacný, ale s obmedzenou frekvenčnou stabilitou) alebo **klystrón** a **TWT** (Travelling Wave Tube – elektrónka s putujúcou vlnou), ktoré ponúkajú vyšší výkon a lepšiu koherenciu (frekvenčnú stabilitu), čo je kľúčové pre meranie rýchlosti. [7]

2.2 ANTÉNA

Anténa plní duálnu úlohu. Je to konštrukcia, ktorá zaisťuje premenu voľne sa šíriaceho elektromagnetického vlnenia na kmitajúci prúd v obvode, ku ktorému je pripojená. Taktiež môže prijímať energiu elektromagnetického poľa a vysielať elektromagnetické vlny, ktoré sú vytvárané oscilátorom. Anténa sa zvyčajne otáča alebo skenuje priestor, čím zabezpečuje pokrytie celého sledovaného objemu. [8]

Najbežnejším typom antény je **parabolická anténa**. Je to anténa so špeciálne ohnutým reflektorom, ktorého tvar ohybu je určený parabolou.

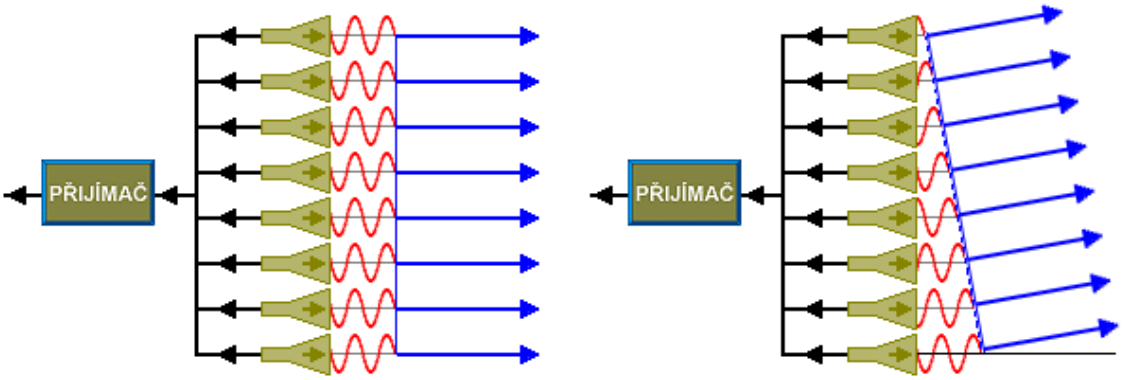


Obr. 2 - Princíp antény s parabolickým reflektorom

(Zdroj: www.radartutorial.eu, rok: 2024)

Obrázok ukazuje štruktúru „normálnej“ (symetrickej) parabolickej antény. Bodový zdroj osvetľuje symetrický reflektor. [9]

Druhý najbežnejší typ antén je **fázovaná anténa** (ang. Phased array), ktorá sa nachádza v modernejších radarových systémoch. Umožňuje elektronické a veľmi rýchle riadenie smeru lúča bez fyzického pohybu antény.



Obr. 3 – Princíp fázovanej antény

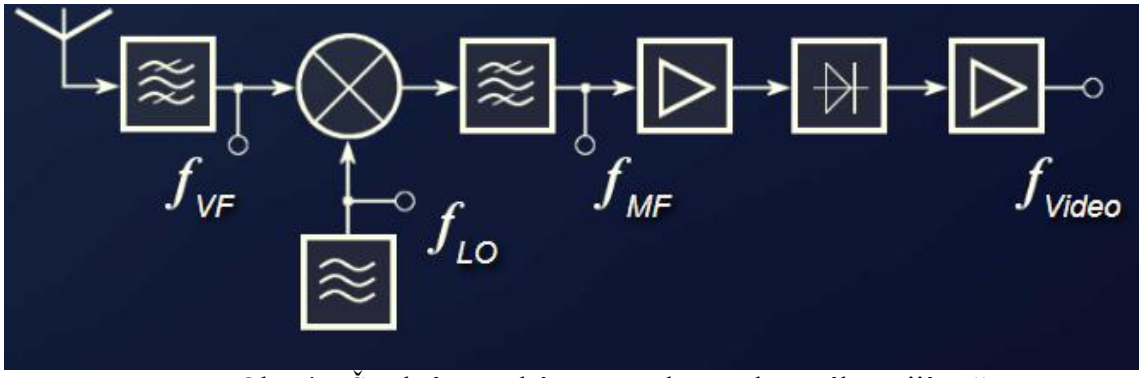
(Zdroj: www.mo.gov.cz, rok: 2008)

Takáto anténa má plochý tvar a je tvorená veľkým počtom (niekedy aj desiatok tisíc) malých modulov. Každý modul obsahuje miniatúrny polovodičový vysokofrekvenčný

generátor, tzv. T/R modul (z ang. transmitter/receiver) s malým výkonom (iba niekoľko Wattov). Jeho vysokofrekvenčná energia je vyžarovaná do určitého smeru. [9], [10]

2.3 PRIJÍMAČ

Funkciou prijímača je prijímať slabé ozveny z anténneho systému, dostatočne ich zosilniť, detekovať impulzy, zosilniť ich a priviesť ich do indikátora. Prijímače používané v radaroch sú schopné prijímať slabé ozveny a zvyšovať ich amplitúdy faktorom 20 alebo 30 miliónov. Prijaté rádiové signály sa najprv musia zosilniť a následne transformovať na videosignál, aby sa z ozvien získali požadované informácie. Túto transformáciu vykonáva **superheterodynový prijímač** (prijímač s nepriamym zosilnením, skrátene **superhet**). Superhet mení vysokofrekvenčný signál (RF) na ľahšie spracovateľnú nižšiu medzifrekvenciu (MF). Táto MF frekvencia sa zosilní a demoduluje, aby sa získal videosignál. Hlavné komponenty typického superheterodynového prijímača sú znázornené na nasledujúcom obrázku:



Obr. 4 – Štruktúrna schéma superheterodynového prijímača
(Zdroj: www.radartutorial.eu, rok: 2007)

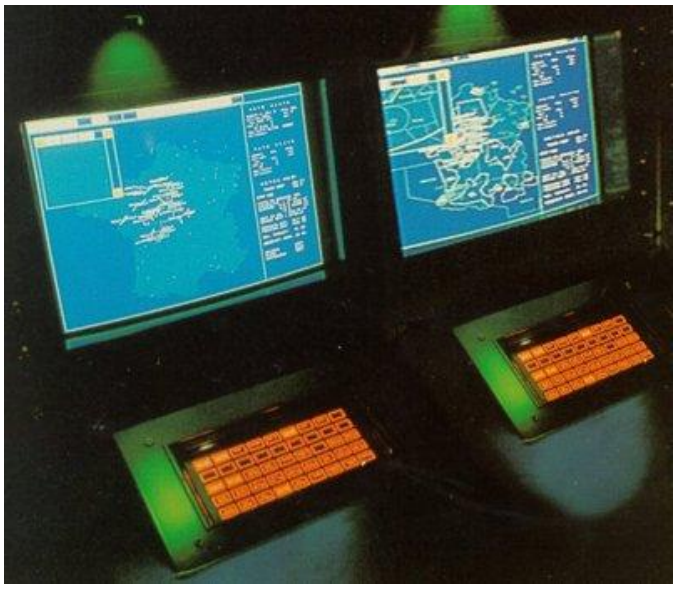
Obrázok znázorňuje blokovú schému typického superheterodynového prijímača. RF nosná frekvencia prichádza z antény a je privádzaná do filtra (f_{RF}). Výstupom filtra sú iba frekvencie požadovaného frekvenčného pásma. Tieto frekvencie sa privádzajú do zmiešavacieho stupňa. Zmiešavač tiež prijíma vstup z lokálneho oscilátora (f_{LO}). Tieto dva signály sú spoločne bité, aby sa získala MF prostredníctvom procesu heterodynovania. Ladením lokálneho oscilátora je vždy fixný rozdiel frekvencií medzi lokálnym oscilátorom a RF signálom. Tento rozdiel frekvencií sa nazýva MF. Tento fixný rozdiel a skupinové ladenie zabezpečuje konštantnú MF v celom frekvenčnom rozsahu prijímača. MF nosná frekvencia sa privádza do MF zosilňovača (f_{MF}). Zosilnená MF

frekvencia sa potom posiela do detektora. Výstupom detektora je video zložka vstupného signálu (f_{video}). [11], [12]

2.4 INDIKÁTOR

Existuje mnoho spôsobov vizuálneho zobrazenia radarových dát. Moderné radarové jednotky na tento účel obvykle používajú ploché obrazovky osobného počítača. Informácie z radarového prijímača môžu obsahovať až niekoľko miliónov samostatných dátových bitov za sekundu. Z týchto a ďalších údajov, ako je orientácia antény, by mal indikátor pozorovateľovi poskytnúť súvislý, ľahko zrozumiteľný grafický obraz relatívnej polohy radarových cieľov. Mal by poskytovať informácie o veľkosti, tvare a pokiaľ je to možné, aj o type cieľov. Základnými geometrickými veličinami v radarových zobrazeniach sú vzdialenosť, azimutálny uhol, elevačný uhol a iné (napr. kurz cieľa). Tieto zobrazenia vzťahujú polohu radarového cieľa k východiskovej polohe antény. [13]

Keďže existuje mnoho typov radarových indikátorov a pokrytie všetkých by bolo v tejto práci zbytočné, budeme sa venovať tomu najrelevantnejšiemu typu, ktorý sa v dnešnej dobe používa najviac. Tento typ sa nazýva **indikátor rastrového snímania**, ktorý môžeme vidieť na obrázku nižšie.



Obr. 5 – Indikátor rastrového snímania vzdušného priestoru Paríža
(Zdroj: radartutorial.eu, rok: 2007)

Väčšina zastaralých radarových indikátorov dokáže zobraziť iba dve z geometrických veličín spomínaných v predchádzajúcom odseku. Indikátor rastrového snímania ale dokáže zobraziť všetky informácie, čo v praxi znamená, že tento indikátor

poskytuje najviac priehľadný obraz s mnohými dodatočnými informáciami. Súradnice cieľových znakov sú prevedené na obrazové čiary a pixely a zobrazené na obrazovke počítača alebo televízora. Celá obrazovka je rozdelená do sústavy riadkov a bodov. Tieto indikátory majú najvyššie možné rozlíšenie. Značku cieľa je možné označiť kurzorom myši a potom sa zobrazia ďalšie informácie. Je možné zvoliť aj priebežné zobrazovanie doplnkových informácií. V prípade radaru používaného pri riadení letovej prevádzky môžu symboly vyzerať ako na obrázku 6. Štvorec je symbolom pre polohu lietadla. Bodky sú polohy lietadiel v predchádzajúcich otáčkach, ktoré symbolizujú kurz a rýchlosť. Horný reťazec znakov je identifikácia lietadla, dolný reťazec znakov udáva nadmorskú výšku buď ako letovú hladinu, alebo v malých výškach ako barometrickú

výšku. [14]

Obr. 6 – Zväčšené zobrazenie na indikátore radaru riadenia letovej prevádzky (Zdroj: radartutorial.eu, rok: 2017)

3 HISTÓRIA RADAROVÝCH SYSTÉMOV

Radarová technika, ako ju poznáme dnes, prešla mnohými desiatkami rokov vývoja. Objav a vývoj radarovej technológie nebol výsledkom práce jedného vynálezcu ani národa. Poznatky o radare treba vnímať ako súhrn mnohých objavov a vývojov v oblasti elektromagnetizmu a rádiovej techniky, na ktorých sa paralelne podieľali vedci z viacerých krajín.

3.1 TEORETICKÉ ZÁKLADY A RANÉ EXPERIMENTY

Základná myšlienka radaru mala svoj pôvod v klasických experimentoch s elektromagnetickým žiarením, ktoré vykonal nemecký fyzik Heinrich Hertz koncom 80. rokov 19. storočia. Hertz sa rozhodol experimentálne overiť skoršiu teoretickú prácu škótskeho fyzika Jamesa Clerka Maxwella. Maxwell sformuloval všeobecné rovnice elektromagnetického poľa a určil, že svetelné aj rádiové vlny sú príkladmi elektromagnetických vĺn, ktoré sa riadia rovnakými základnými zákonmi, ale majú veľmi odlišné frekvencie. Maxwellova práca viedla k záveru, že rádiové vlny sa môžu odrážať od kovových predmetov a lámať dielektrickým médiom, rovnako ako svetelné vlny. Hertz tieto vlastnosti demonštroval v roku 1888 pomocou rádiových vĺn s vlnovou dĺžkou 66 cm (čo zodpovedá frekvencii približne 455 MHz). [15]

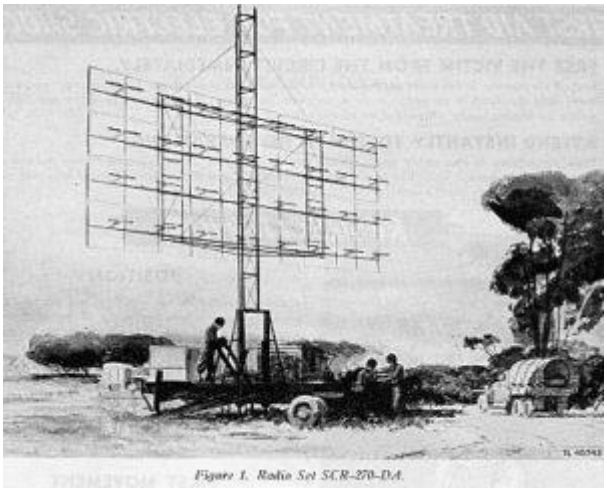
Hertzova práca a jej potenciál ako základ pre detekciu cieľov praktického záujmu v tom čase nezostala bez povšimnutia. V roku 1904 bol nemeckému inžinierovi Christianovi Hülsmeyerovi v niekoľkých krajinách udelený patent na „detektor prekážok a navigačné zariadenie pre lode“, založený na princípoch, ktoré Hertz demonštroval. Hülsmeyer svoj vynález zostrojil a predviedol nemeckému námorníctvu, ktoré ale nemalo žiadny vážny záujem. Až do začiatku 30. rokov 20. storočia jednoducho neexistovala žiadna ekonomická, spoločenská ani vojenská potreba radaru, až kým neboli vyvinuté vojenské bombardéry s dlhým doletom. Toto podnietilo hlavné krajiny sveta hľadať prostriedky na detekciu priblíženia nepriateľských lietadiel. [15]

3.2 MEDZIVOJNOVÉ OBDOBIE A PRELOM V 30. ROKOCH

Počas 30. rokov 20. storočia sa v ôsmich krajinách (USA, Veľká Británia, Nemecko, Francúzsko, Sovietsky zväz, Taliansko, Holandsko a Japonsko

), ktoré sa obávali prevládajúcej vojenskej situácie a už mali praktické skúsenosti s rádiovou technológiou, nezávisle a takmer súčasne začali snahy o využitie rádiových ozvien na detekciu lietadiel. [15]

Prvé pozorovanie radarového efektu v **Americkom** námornom výskumnom laboratóriu (NRL) vo Washingtone, D.C., sa uskutočnilo v roku 1922. Výskumníci z NRL umiestnili rádiový vysielač na jeden breh rieky Potomac a prijímač na druhý. Loď plaviaca sa po rieke nečakane spôsobila kolísanie intenzity prijímaných signálov, keď prechádzala medzi vysielačom a prijímačom. Princíp radaru bol „znovuobjavený“ v roku 1930, keď L.A. Hyland pozoroval, že lietadlo letiace cez lúč vysielacej antény spôsobuje kolísanie prijímaného signálu. Hoci Hyland a jeho spolupracovníci v NRL boli nadšení z možnosti detekcie cieľov rádiovými prostriedkami a dychtivo sa venovali jej vývoju, vyššie orgány v námorníctve prejavili malý záujem. Až keď sa zistilo, ako používať jednu anténu na vysielanie aj príjem (dnes nazývané monostatický radar), hodnota radaru na detekciu a sledovanie lietadiel a lodí bola plne uznaná. Takýto systém bol demonštrovaný na mori na bojovej lodi USS New York začiatkom roku 1939. Prvými radarmi vyvinutými americkou armádou boli SCR-268 na riadenie protilietadlovej paľby a SCR-270 na detekciu lietadiel. Práve SCR-270, ktorý 7. decembra 1941 detekoval približovanie japonských vojnových lietadiel k Pearl Harboru. [15]



Obr. 7. – Radarová jednotka SCR-270 v Pearl Harbor
(Zdroj: infoage.org, rok: 2023)

Nemecko na začiatku druhej svetovej vojny pokročilo vo vývoji radaru ďalej ako ktorákoľvek iná krajina. Nemci používali radar na zemi aj vo vzduchu na obranu proti spojeneckým bombardérom. Radar bol nainštalovaný na nemeckej bojovej lodi už v roku

1936. Vývoj radaru Nemci zastavili koncom roka 1940, pretože verili, že vojna sa takmer skončila. Spojené štáty a Británia však svoje úsilie zrýchlili. Keď si Nemci uvedomili svoju chybu, bolo už neskoro na to, aby dobehli zameškané. Skoro všetky úspešné radarové systémy vyvinuté pred začiatkom druhej svetovej vojny pracovali v pásme VHF. Používanie VHF predstavovalo niekoľko problémov. Šírka lúča VHF je široká, a tá má menšiu presnosť, horšie rozlíšenie a viac nežiadúcich ozvien od zeme alebo iného rušenia ako úzka šírka lúča. Ďalej, časť elektromagnetického spektra VHF neumožňuje široké pásma potrebné pre krátke impulzy, ktoré umožňujú väčšiu presnosť pri určovaní vzdialenosti. VHF je taktiež náchylné na atmosférický šum, ktorý obmedzuje citlivosť prijímača. Napriek týmto nevýhodám bolo v tej dobe VHF skutočným priekopníckym úspechom. Prví vývojári radarov si dobre uvedomovali, že prevádzka na ešte vyšších frekvenciách je nutná, najmä preto, že úzka šírka lúča sa dá dosiahnuť bez nadmerne veľkých antén. [15]

Spojené kráľovstvo začalo s výskumom radarov na detekciu lietadiel v roku 1935. Britská vláda povzbudzovala inžinierov, aby postupovali rýchlo, pretože sa dosť obávala rastúcej možnosti vojny. Do septembra 1938 bol prvý britský radarový systém, Chain Home, uvedený do 24-hodinovej prevádzky a zostal v prevádzke počas celej vojny. Radary Chain Home umožnili Británii úspešne nasadiť svoju obmedzenú protivzdušnú obranu proti silným nemeckým leteckým útokom, ktoré sa uskutočnili počas prvej fázy vojny. [15]

Aj **Sovietsky zväz** začal pracovať na radare v 30. rokoch 20. storočia. V čase nemeckého útoku na ich krajinu v júni 1941 Sovieti vyvinuli niekoľko rôznych typov radarov a vyrábali radar na detekciu lietadiel, ktorý pracoval na frekvencii 75 MHz (v pásme VHF). Ich vývoj a výrobu radarových zariadení narušila nemecká invázia a práce sa museli premiestniť. [15]

3.3 POVOJNOVÝ VÝVOJ A CIVILNÉ VYUŽITIE

Po druhej svetovej vojne sa výskum značne spomalil a zameral sa na vyššiu presnosť a nové metódy spracovania. Počas 50. rokov boli publikované dôležité teoretické koncepty, ktoré pomohli postaviť návrh radarov na kvantitatívnejší základ. Medzi ne patrili napríklad základné metódy Dopplerovho filtrovania v radaroch, ktoré sa neskôr stali dôležitými, keď digitálna technológia umožnila, aby sa teoretické koncepty stali praktickou realitou. V 60. rokoch boli do prevádzky uvedené prvé veľké radary s elektronicky riadeným fázovaným poľom. Tiež bol vyvinutý systém „Airborne MTI“ na

detekciu lietadiel z paluby iných strojov. 70. roky priniesli obrovský pokrok v digitálnych technológiách, ktoré umožnili praktické spracovanie signálov a dát, čo je nevyhnutné pre moderné systémy. [15]

Radarová technológia sa tiež začala používať v civilnom sektore. Radary sa stali základným pilierom bezpečnosti v civilnom letectve a riadení letovej prevádzky. Keďže lietadiel v dnešnej dobe lieta naozaj mnoho, pre zachovanie bezpečnosti a plynulosti letovej prevádzky je potrebné mať neustály prehľad o aktuálnej situácií vo vzdušnom priestore. Taktiež sa začali používať aj pri meteorologických pozorovaniach. Podľa použitej frekvencie je totiž možné detekovať ľubovoľne veľké objekty vo vzdušnom priestore, a toto zahŕňa aj zrážky. Takéto radary sú umiestňované nielen na zemskom povrchu, ale aj na palubách moderných lietadiel. [16]

Obr. 8: Radar na letisku M. R. Štefánika v Bratislave
(Zdroj: facebook.com/letiskobratislava, rok: 2018)

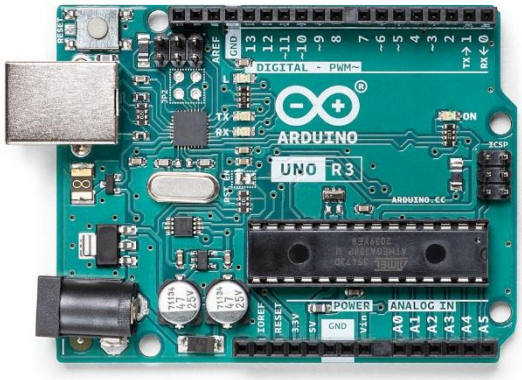
4 ANALÝZA TECHNOLÓGIÍ NA TVORBU PRODUKTU

V tejto časti sa venujeme technickému opodstatneniu výberu hardvérovej a softvérovej platformy. Pri návrhu interaktívneho systému musíme zvoliť technológie, ktoré sú navzájom kompatibilné a umožňujú efektívny prenos dát v reálnom čase.

4.1 HARDVÉROVÁ PLATFORMA: ARDUINO UNO R3

Platforma Arduino je open-source ekosystém založený na mikrokontroléroch. Jeho výber pre môj produkt je podmienený niekoľkými faktormi:

- **Abstrakcia hardvéru** - Arduino poskytuje knižnice, ktoré zjednodušujú prácu s perifériami (I/O piny, ADC prevodníky, časovače...).
- **Spracovanie v reálnom čase** - Mikrokontrolér pracuje na nízkej úrovni bez réžie operačného systému, čo zaručuje predurčené správanie pri čítaní hodnôt zo senzorov (napr. vzorkovanie analógového signálu).
- **Sériová komunikácia** - Arduino disponuje integrovaným USB-to-Serial prevodníkom, ktorý umožňuje okamžitú komunikáciu s PC bez potreby dodatočného hardvéru.



Obr. 9: Vývojová doska Arduino Uno R3

(Zdroj: store.arduino.cc, rok: 2026)

4.2 SOFTVÉROVÉ PLATFORMY: ARDUINO IDE A PROCESSING

Arduino IDE je oficiálne softvérové prostredie určené na písanie, kompiláciu a nahrávanie kódu do mikrokontrolérov Arduino. V rámci analýzy technológií ho posudzujeme ako primárny nástroj na správu firmvéru zariadenia. Je priamo založené na vývojovom prostredí Processing, čo znamená, že bol vytvorený s použitím zdrojového kódu Processingu a zdedil jeho vzhľad, dojem a štruktúru vývojového prostredia, aby

bolo programovanie prístupné, najmä pre vizuálne a interaktívne projekty. Používa programovací jazyk Wiring, ktorý je založený na C/C++, avšak s výrazným zjednodušením.[17]

Processing je grafické programovacie prostredie postavené na jazyku Java. Bol navrhnutý pre vizuálnych umelcov a dizajnérov, čo ho robí ideálnym pre tvorbu GUI (Graphical User Interface) pre Arduino projekty. Processing a Arduino IDE zdieľajú rovnakú funkciu „skicovania" (sketches). Ich syntax je veľmi podobná, čo uľahčuje paralelný vývoj na oboch stranách. Natívne podporuje 2D aj 3D grafiku (cez OpenGL/P3D), prácu s obrazom a videom, čo by bolo v čistom C++ alebo Pythone náročnejšie na implementáciu. Kľúčovým prvkom analýzy je knižnica *processing.serial*, ktorá umožňuje aplikácii pristupovať k systémovým COM portom a asynchrónne čítať dáta z Arduina.

4.3 SIMULÁCIA OBVODU - CIRKIT DESIGNER

Cirkit Designer je moderné „all-in-one" vývojové prostredie určené na návrh, simuláciu a dokumentáciu elektronických obvodov, najmä tých s mikrokontrolérmi ako Arduino. Medzi hlavné funkcie Cirkit-u patrí:

1. **Vizuálny návrh obvodov** - Obsahuje knižnicu reálnych komponentov (senzory, LED, displeje, servomotory, relé...), ktoré môžeme jednoducho prepájať na virtuálnom nepájivom poli (breadboarde).
2. **Interaktívna simulácia** - Môžeme v reálnom čase simulovať obvod a vidieť, ako funguje. Podporuje aj simuláciu kódu pre Arduino, čo pomáha pri vizualizácii a pochopení, ako program ovláda hardvér
3. **Editor kódu (IDE)** - Umožňuje priamo v prehliadači písať kód, kompilovať ho a spúšťať v simulácii.

4.4 KOMUNIKAČNÝ PROTOKOL: SÉRIOVÁ LINKA

Prepojenie týchto dvoch svetov prebieha prostredníctvom protokolu UART. V analýze musíme zvážiť spôsob kódovania dát:

1. **Binárny prenos** - Efektívny na prenosovú rýchlosť, ale náročnejší na synchronizáciu (potreba definovať začiatok a koniec paketu).
2. **ASCII (textový) prenos** - Hodnoty sa posielajú ako čitateľné znaky. Výhodou je jednoduché ladenie a automatická synchronizácia pomocou

oddeľovača nového riadku (Line Feed). Pre našu aplikáciu volíme ASCII prenos kvôli spoľahlivosti.

4.5 POROVNANIE S ALTERNATÍVAMI

V rámci analýzy je potrebné zdôvodniť, prečo neboli použité iné technológie:

- **Python (PySerial + Pygame/Tkinter)** - Python je silný konkurent, ale Processing ponúka priamočiarejšiu cestu k rýchlej vizualizácii bez nutnosti riešiť správu okien a externé závislosti.
- **C# (.NET/Unity)** - Ponúka profesionálne rozhranie, ale je výrazne komplexnejší na konfiguráciu sériového portu a náročnejší na systémové zdroje.
- **ESP32 namiesto Arduina** - Ak by aplikácia vyžadovala bezdrôtový prenos (Wi-Fi/Bluetooth), ESP32 by bol vhodnejšou voľbou. Pre káblové (USB) riešenie však Arduino ostáva stabilnejším štandardom.
- **Tinkercad** - Je ideálny na rýchle vzdelávacie simulácie a blokové programovanie, avšak v porovnaní s Cirkit Designerom poskytuje obmedzenú knižnicu reálnych komponentov a menej profesionálne technické výstupy.

5 ELEKTRICKÝ OBVOD PROTOTYPU

V tejto kapitole si podrobne popíšeme obvod môjho radarového prototypu, použité komponenty a funkcie.

5.1 POUŽITÉ KOMPONENTY

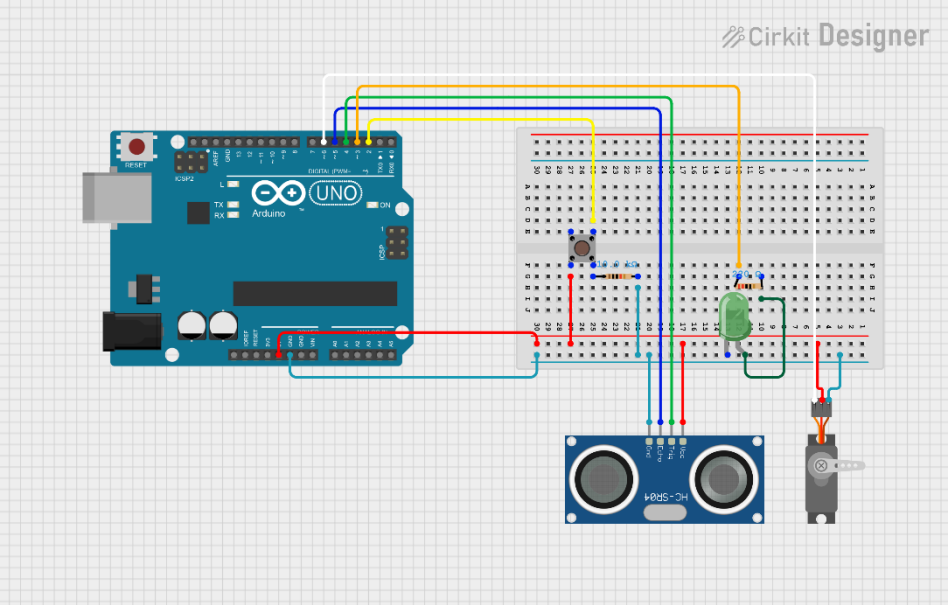
- **Arduino Uno R3** - Ako bolo spomenuté v predošlej kapitole, je to mikrokontrolér ideálny pre tento obvod. Je založený na 8-bitovom procesore ATmega328P. Verzia R3 je treťou revíziou tejto vývojovej dosky a stala sa priemyselným štandardom pre výučbu mechatroniky. Poskytuje 14 digitálnych „pinov“, ktoré je možné použiť ako na vstupy, tak aj na výstupy. Z nich je 6 PWM, ktoré pomocou digitálneho signálu (striedanie zapnutia/vypnutia) efektívne riadia zariadenia ako LED diódy alebo servomotory.
- **Ultrazvukový senzor HC-SR04** - Cenovo dostupný a ľahko použiteľný senzor merania vzdialenosti s rozsahom od 2 cm do 400 cm. Skladá sa z dvoch ultrazvukových prevodníkov. Jeden je vysielač, ktorý vysiela ultrazvukové zvukové impulzy, a druhý je prijímač, ktorý počúva odrazené vlny. V podstate ide o SONAR, ktorý sa používa v ponorkách a v princípe sa veľmi podobá radarovým systémom. Senzor má 4 piny:
    - VCC - týmto pinom senzor napájame z 5V pinu
    - GND - pripojený k GND pinu na uzemnenie systému
    - TRIG - vysiela ultrazvukovú vlnu z vysielača
    - ECHO - pomocou neho prijímame odrazený signál [18]
- **Servo motor SG90** - Malé a ľahké servo s vysokým výstupným výkonom. Môže sa otáčať o 180 stupňov (90 v každom smere). Riadime ho pomocou PWM signálov. Má 3 piny:
    - VCC – napájanie z 5V
    - GND – uzemnenie
    - PWM – vstupný pin, ktorý prijíma impulzy z mikrokontroléra. Šírka týchto impulzov určuje presný uhol natočenia hriadeľa motora (napr. 1 ms = 0°, 2 ms = 180°). [19]
- **Tlačidlo** - Po stlačení vodivo prepojí dva body v obvode. Má 4 piny (nožičky), ktoré sú interne spojené v pároch. Pri stlačení sa spoja všetky štyri piny, čím sa

uzavrie elektrický obvod. Piny sú univerzálne, na Arduino vývojovej doske môžu byť pripojené na piny GND, 5V na napájanie a niektorý zo 14 digitálnych pinov.

- **LED dióda** - Má dve nožičky. Kratšia z nich (katóda) musí byť pripojená na uzemňovací pin GND. Dlhšia nožička (anóda) môže byť napojená na ktorýkoľvek digitálny alebo analógový pin.
- **Rezistory –** Funguje na základe premeny energie získanej z prúdiacich elektrónov na teplo. Rezistory je možné použiť na znižovanie napätia alebo riadenie prúdu. Ich odpor sa meria v ohmoch. Odpor 1 ohm vzniká vtedy, keď cez rezistor preteká prúd 1A a na jeho vývodoch je napätie 1V.

5.2 POPIS OBVODU

Obvod prototypu je znázornený na nasledujúcom obrázku.



Obr. 10 – Elektrický obvod prototypu Arduino radar

(Zdroj: Vlastné spracovanie, rok: 2026)

1. **Riadiaca jednotka (Arduino Uno R3)** - slúži ako “mozog” celého obvodu. Je napájané cez USB kábel, ktorý vedie do PC. Napätie 5V a uzemnenie GND sú vyvedené na napájacie lišty nepájivého poľa.
2. **Ultrazvukový senzor** - bude merať a vracať vzdialenosť pomocou zvukových vĺn. Piny VCC a GND sú pripojené na nepájivé pole v príslušných riadkoch. Pin Echo (tmavomodrý vodič), ktorý bude odosielať impulz je pripojený k digitálnemu pinu 5. Pin Trig (zelený vodič), ktorý bude prijímať odrazy impulzov je pripojený k digitálnemu pinu 4.

3. **Servomotor** - malý motorček na obrázku vpravo dole, ktorý sa bude otáčať podľa zaslaného signálu. Piny VCC a GND sú opäť napojené na nepájivé pole pre maximálnu prehľadnosť. Pin PWM (biely vodič), ktorý prijíma signál je priamo napojený na digitálny pin 6. Na Arduine je vedľa tohto pinu znak „~", čo znázorňuje, že to je PWM pin. Signál pre servomotor je kľúčové zapojiť do takéhoto pinu, pretože nefunguje na princípe zmeny napätia (napr. 2V pre jeden uhol a 4V pre druhý). Namiesto toho očakáva digitálny signál v podobe pravouhlých kmitov.
4. **LED dióda (zelená)** - slúži ako vizuálna signalizácia, že obvod je aktívny. Ak je Arduino zapojené pomocou USB kábla, dióda bude staticky svietiť. Ak je radar aktívny, bude blikať. Katóda diódy je pripojená GND lištu nepájivého poľa. Anóda je pripojená oranžovým vodičom k digitálnemu PWM pinu 3 a chránená 220 Ω rezistorom, aby nevyhorela.
5. **Tlačidlo** - funguje ako vstupný prvok pre používateľa. Pri jeho stlačení sa zapne servomotor a ultrasonický radar začne snímať. Taktiež začne blikať LED dióda. Ak tlačidlo stlačíme znova, servomotor zastaví, radar prestane snímať a LED dióda prestane blikať. Tlačidlo je zapojené pomocou tzv. pull-down rezistora (s hodnotou 10 kΩ), ktorý je kritický pre stabilitu. Ak je tlačidlo rozpojené, rezistor „sťahuje" napätie na pine k nule, takže Arduino číta čistú logickú 0. Ak je tlačidlo stlačené, vytvorí sa priama cesta medzi 5 V a pinom 4. Keďže odpor rezistora je oveľa vyšší než odpor zopnutého tlačidla, na pine prevládne napätie 5 V a Arduino číta logickú 1. Ak by sme rezistor vynechali a pin 4 by nebol nikam pripojený (stav „v lufte" alebo floating), pin by fungoval ako malá anténa. Zachytával by elektromagnetický šum z okolia, čo by spôsobovalo náhodné spínanie serva alebo LED diódy, aj keď sa nikto tlačidla nedotkol.

Komponent	Pin na Arduine	Farba vodiča
Tlačidlo	D2	Žltá
LED dióda (anóda)	D3	Oranžová
HC-SR04 (Trig)	D4	Zelená
HC-SR04 (Echo)	D5	Tmavomodrá
Servomotor (signál)	D6	Biela

Tabuľka 1. - Zhrnutie pinov

6 IMPLEMENTÁCIA RIADIACEHO SOFTVÉRU NA ARDUINO

V tejto kapitole si prejdeme zdrojový kód na riadenie hardvéru, jeho hlavné funkcie a algoritmy.

6.1 KONFIGURÁCIA PINOV, PREMENNÉ A VOID SETUP()

Na úplnom začiatku, aby sme dokázali posielať a prijímať dáta z komponentov, musíme ich definovať na rovnaké piny, do ktorých sú hardvérovo zapojené.

Pomocou *#include* importujeme štandardnú knižnicu na ovládanie servomotorov.

Komponentom *#define* priraďujeme piny pre hardvér rovnako, ako bolo spomínané v predošlej kapitole.

```
#include <Servo.h>

#define trigPin 4
#define echoPin 5
#define buttonPin 2
#define ledPin 3

long duration;
int distance;
int opState = 0;
int servoPos = 90;
int stepDir = 1;

Servo servo;

unsigned long lastDebounceTime = 0;
const unsigned long debounceMs = 40;
int lastButtonState = LOW;

unsigned long lastBlinkMs = 0;
const unsigned long blinkPeriodMs = 200;
bool ledBlinkState = false;

unsigned long lastServoMoveMs = 0;
const int servoInterval = 15;

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(buttonPin, INPUT);
  pinMode(ledPin, OUTPUT);
  servo.attach(6);
  servo.write(servoPos);
  Serial.begin(9600);
}
```

Obr. 11 – Základy kódu

(Zdroj: Vlastné spracovanie, rok: 2026)

Inicializujeme premenné, ktoré uchovávajú informácie ako čas návratu zvukovej vlny, vzdialenosť, stav radaru, poloha serva a smer pohybu (1 pre vpravo, -1 pre vľavo).

Využívame premenné typu unsigned long na sledovanie času, čo umožňuje Arduinu vykonávať viac úloh súčasne (multitasking).

Funkcia *setup()* sa spustí iba raz pri štarte a pripraví hardvér na prevádzku . V nej inicializujeme jednotlivé piny na OUTPUT (vysielanie signálu) alebo INPUT (prijímanie signálu). Pomocou *servo.attach()* priraďujeme servo k pinu 6 a *servo.write()* ho nastaví do východiskovej polohy hneď po zapnutí. *Serial.begin(9600)* otvára sériovú komunikáciu s rýchlosťou 9600 baudov pre prenos dát do PC.

6.2 OBSLUHA TLAČIDLA A KLÁVESNICE

Tieto funkcie umožňujú používateľovi interagovať so zariadením.

Funkcia *handleButton()* slúži na zapínanie a vypínanie zariadenia fyzickým tlačidlom.

Pomocou millis() a debounceMs kód ignoruje falošné signály vznikajúce pri mechanickom stlačení tlačidla. Ak je detegované reálne stlačenie, hodnota *opState* sa zneguje (*!opState*), čím sa prepne režim ovládanie medzi manuálnym a automatickým.

```
48  void handleButton() {
49    int reading = digitalRead(buttonPin);
50    if (reading != lastButtonState) lastDebounceTime = millis();
51
52    if ((millis() - lastDebounceTime) > debounceMs) {
53      static int stableState = LOW;
54      if (reading != stableState) {
55        stableState = reading;
56        if (stableState == HIGH) opState = !opState;
57      }
58    }
59    lastButtonState = reading;
60  }
61
62  void handleKeyboard() {
63    if (Serial.available() > 0) {
64      char key = Serial.read();
65      if (key == 'd') {
66        opState = 0;
67        servoPos = constrain(servoPos - 1, 15, 165);
68        servo.write(servoPos);
69        calculateDistance();
70        sendData();
71      }
72      else if (key == 'a') {
73        opState = 0;
74        servoPos = constrain(servoPos + 1, 15, 165);
75        servo.write(servoPos);
76        calculateDistance();
77        sendData();
78      }
79      else if (key == 'm') {
80        opState = !opState;
81      }
82    }
83  }
```

Funkcia *handleKeyboard()* číta znaky poslané z klávesnice počítača cez sériový monitor. Pomocou podmienky kontrolujeme, aký znak bol poslaný:

- **‘a’ / ‘d’**: Tieto klávesy posunú servo o 1 stupeň doľava (a) alebo doprava (d). Funkcia *constrain*() zabezpečuje, aby servo neprekročilo bezpečný rozsah 15° až 165°.
- **‘m’**: Funguje rovnako ako fyzické tlačidlo – prepína medzi režimami.

Obr. 12 – Ovládanie zariadenia

(Zdroj: Vlastné spracovanie, rok: 2026)

6.3 AUTOMATICKÉ SKENOVANIE A MANIPULÁCIA S LED DIÓDOU

Funkcia *autoScan()* zabezpečuje plynulý pohyb radaru bez zastavenia zvyšku programu. Funguje nasledovne:

- **Časovač** - pohyb sa vykoná len vtedy, ak uplynulo aspoň 15 ms (*servoInterval*) od posledného pohybu.

- **Zmena smeru** - keď servo dosiahne hranicu 165° alebo 15°, hodnota premennej *stepDir* sa vynásobí -1, čím sa pohyb otočí.
- **Sled úloh** - po každom malom pohybe sa zavolá meranie vzdialenosti (*calculateDistance)* a okamžité odoslanie dát (*sendData*)

Funkcia *updateLed()* slúži na vizuálnu signalizáciu aktuálneho stavu systému pomocou LED diódy. Jej logika je postavená dvoch vetvách podľa hodnoty premennej *opState*. Má 2 režimy:

1. **Manuálny režim** (*opstate == 0*) - v tomto stave je servo zastavené a riadenie je možné iba cez klávesnicu. Vykoná sa príkaz *digitalWrite(ledPin, HIGH)*, čo rozsvieti LED diódu.
2. **Automatický režim** (*opstate == 1*) - radar sa sám otáča a skenuje okolie. Namiesto funkcie *delay()*, ktorá by zastavila chod celého programu (aj serva a radaru) sa používa porovnávanie aktuálneho času *millis()* s posledným uloženým časom bliknutia *lastBlinkMs*. Ak od poslednej zmeny stavu uplynulo viac ako 200 ms (*blinkPeriodMs*), podmienka sa splní a invertuje stav *ledBlinkState*. Pomocou ternárneho operátora tento stav kontrolujeme. Ak je stav *true*, LED sa zapne, ak je *false*, LED sa vypne. Toto zabezpečuje blikanie LED diódy každých 200 ms.

```
85  void autoScan() {
86    if (millis() - lastServoMoveMs >= servoInterval) {
87      lastServoMoveMs = millis();
88      servoPos += stepDir;
89      if (servoPos >= 165 || servoPos <= 15) stepDir *= -1;
90
91      servo.write(servoPos);
92      calculateDistance();
93      sendData();
94    }
95  }
96
97  void updateLed() {
98    if (opState == 0) {
99      digitalWrite(ledPin, HIGH);
100   } else {
101
102     if (millis() - lastBlinkMs >= blinkPeriodMs) {
103       lastBlinkMs = millis();
104       ledBlinkState = !ledBlinkState;
105       digitalWrite(ledPin, ledBlinkState ? HIGH : LOW);
106     }
107   }
108 }
```

Obr. 13 - AutoScan a manipulácia s LED diódou

(Zdroj: Vlastné spracovanie, rok: 2026)

6.4 VÝPOČET VZDIALENOSTI A ODOSIELANIE DÁT

Vo funkcii *calculateDistance()* prebieha výpočet vzdialenosti na základe rýchlosti zvuku. Na začiatku nastavíme vysielací pin senzoru na nízku úroveň, čo zabezpečí čistý začiatok bez elektrického šumu. Po dvoch mikrosekundách sa nastaví pin na vysokú úroveň presne na 10 mikrosekúnd, čo vyšle presne 8 ultrazvukových vĺn. Pomocou funkcie *pulseIn()* meriame, ako dlho trvá, kým sa zvukový signál vráti. Parameter 30000 je tzv. **timeout** - ak sa zvuk nevráti do 30 ms (čo odpovedá vzdialenosti cca. 5 metrov), meranie sa ukončí, aby program nezamrzol. Túto hodnotu potom vynásobíme rýchlosťou zvuku vo vzduchu, približne 340 m/s, čo je 0,034 $cm/\mu s$. Následne ju vydelíme polovicou, pretože zvuk musí prejsť cestu k objektu dvakrát (tam aj naspäť). Funkcia vráti hodnotu vzdialenosti v centimetroch.

Funkcia *sendData()* posiela do sériového monitoru polohu servomotora (*servoPos*) a vzdialenosť objektu (*distance*) oddelené čiarkou. Táto funkcia je kľúčová k tomu, aby sme mohli dáta zobraziť v radarovom zobrazení.

```
110  int calculateDistance() {
111    digitalWrite(trigPin, LOW);
112    delayMicroseconds(2);
113    digitalWrite(trigPin, HIGH);
114    delayMicroseconds(10);
115    digitalWrite(trigPin, LOW);
116    duration = pulseIn(echoPin, HIGH, 30000);
117    distance = duration * 0.034 / 2;
118    return distance;
119  }
120
121  void sendData() {
122    Serial.print(servoPos);
123    Serial.print(",");
124    Serial.print(distance);
125    Serial.print(".");
126  }
```

Obr. 14 - Výpočet a odosielanie dát

(Zdroj: Vlastné spracovanie, rok: 2026)

7 IMPLEMENTÁCIA VIZUALIZAČNÉHO SOFTVÉRU

V tejto kapitole si podrobne rozoberieme kód v prostredí Processing, ktorý slúži na spracovanie a zobrazenie dát prijatých z ultrazvukového senzora.

7.1 KONFIGURÁCIA PROSTREDIA A PREMENNÝCH

Na začiatku programu importujeme knižnicu *processing.serial.**, ktorú potrebujeme na komunikáciu s Arduinom. Deklarujeme port pomocou knižnice Serial a ďalšie potrebné premenné. Zadefinujeme základné farby radaru a detegovaných objektov. Booleany *showMenu* a *useMetric* nastavíme na *true*, aby sa po zapnutí aplikácie ukázalo menu a aby boli použité metrické jednotky. Vo funkcii *setup()* nastavíme rozlíšenie okna na 1920x1080 pixelov a inicializujeme sériový port COM3 s rýchlosťou 9600 baudov. Príkaz *port.bufferUntil('.')* zabezpečuje, že program spracuje dáta až po prijatí ukončovacieho znaku (bodky).

```
import processing.serial.*;

Serial port;
String angle="", distance="", data="";
float pixsDistance;
int iAngle, iDistance, index1=0;


color radarColor = color(98, 245, 31);
color objectColor = color(255, 10, 10);
boolean showMenu = true;
boolean useMetric = true;
float fadeAmount = 20;

void setup() {
  size(1920, 1080);
  smooth();
  port = new Serial(this, "COM3", 9600);
  port.bufferUntil('.');
}
```

Obr. 15 - Setup

(Zdroj: Vlastné spracovanie, rok: 2026)

Aby sme umožnili ľahké prepínanie medzi viacerými jazykmi, inicializujeme dvojrozmerné pole *translations* a do neho vložíme jednotlivé jazyky a preklady ako na obrázku 16.

```
String[] langNames = {"EN", "SK", "FR", "ES", "RU"};
String[][] translations = {
  // EN
  {"Angle", "Distance", "Out of Range", "Units", "Radar Theme",
  "Target Color", "Language", "Trail", "Green", "Blue", "White",
  "Pink", "Orange", "Red", "Yellow", "Metric", "Imperial", "Short",
  "Long", "Press 'M' to Toggle Menu"},
  // SK
  {"Uhol", "Vzdialenosť", "Mimo dosahu", "Jednotky", "Farba radaru",
  "Farba cieľa", "Jazyk", "Stopa", "Zelená", "Modrá", "Biela", "Ružová",
  "Oranžová", "Červená", "Žltá", "Metrické", "Imperiálne", "Krátka", "Dlhá",
  "Stlačte 'M' pre Menu"},
  // FR
  {"Angle", "Distance", "Hors de portée", "Unités", "Thème Radar", "Couleur Cible",
  "Langue", "Trace", "Vert", "Bleu", "Blanc", "Rose", "Orange", "Rouge", "Jaune",
  "Métrique", "Impérial", "Courte", "Longue", "Appuyez sur 'M' pour le Menu"},
  // ES
  {"Ángulo", "Distancia", "Fuera de rango", "Unidades", "Color del Radar",
  "Color de Objetivo", "Idioma", "Rastro", "Verde", "Azul", "Blanco", "Rosa",
  "Naranja", "Rojo", "Amarillo", "Métrico", "Imperial", "Corto", "Largo",
  "Presione 'M' para el Menú"},
  // RU
  {"Угол", "Расстояние", "Вне зоны", "Единицы", "Тема радара", "Цвет цели", "Язык",
  "След", "Зеленый", "Синий", "Белый", "Розовый", "Оранжевый", "Красный", "Желтый",
  "Метрическая", "Имперская", "Короткий", "Длинный", "Нажмите 'M' для меню"}
};
```

Obr. 16 - Dvojrozmerné pole s prekladmi

7.2 SPRACOVANIE A VYKRESĽOVANIE DÁT

Funkcia *serialEvent()* beží na pozadí a rozkladá prijatý reťazec (napr. “90,25.”) na dve samostatné hodnoty: **uhol** (*iAngle*) a **vzdialenosť** (*iDistance*).

Hlavná funkcia *draw()* sa vykonáva neustále a zabezpečuje prekresľovanie radaru. Používa techniku „fading“ pomocou polopriehľadného obdĺžnika (*fill(0, fadeAmount)*), čo vytvára efekt miznúcej stopy za pohyblivou líniou radaru. V každom snímku sa volajú funkcie na vykresľovanie jednotlivých komponentov:

- **drawRadar()**: vykresľuje statické pozadie, polkruhy a uhlové čiary,
- **drawLine()**: znázorňuje aktuálny smer natočenia servomotora,
- **drawObject()**: ak senzor deteguje objekt v dosahu 40 cm, vykreslí červenú čiaru na príslušnom uhle a vzdialenosti,
- **drawText()**: zobrazuje číselné údaje o uhle a vzdialenosti v dolnej časti obrazovky,
- **drawMenu()**: zobrazuje interaktívne menu s používateľskými nastaveniami.

```
51 void serialEvent (Serial port) {
52   data = port.readStringUntil('.');
53   if (data != null) {
54     data = data.substring(0, data.length()-1);
55     index1 = data.indexOf(",");
56     if (index1 > -1) {
57       angle= data.substring(0, index1);
58       distance= data.substring(index1+1, data.length());
59       iAngle = int(angle);
60       iDistance = int(distance);
61     }
62   }
63 }
64
65 void draw() {
66   noStroke();
67   fill(0, fadeAmount);
68   rect(0, 0, width, height-height*0.065);
69
70   drawRadar();
71   drawLine();
72   drawObject();
73   drawText();
74   drawMenu();
75 }
```

Obr. 17 - Práca s dátami v aplikácii

(Zdroj: Vlastné spracovanie, rok: 2026)

7.3 OVLÁDANIE A INTERAKTÍVNE MENU

Program umožňuje používateľovi prispôsobiť si prostredie pomocou interaktívneho menu, ktoré sa aktivuje/deaktivuje stlačením klávesy ‘M‘.

Funkcia *keyPressed()* reaguje pomocou podmienky na stlačenie klávesy ‘M‘ a invertuje stav premennej *showMenu*. Pri skrytí menu sa vykreslí čierny obdĺžnik, ktorý okamžite vymaže zvyšky grafiky menu z obrazovky.

Ďalej zachytávame stlačenie klávesov na manuálne ovládanie serva. Pomocou podmienky *CODED* zachytávame špeciálne klávesy, ktoré nemajú priradený štandardný ASCII znak. Ak používateľ stlačí šípku vľavo, program odošle cez sériový port znak 'a'. V Arduine tento príkaz spôsobí posun serva doľava. Naopak pri stlačení šípky vpravo sa odošle znak 'd', čo spôsobí posun serva doprava. Vetva *else* obsahuje aj klasické písmená, pričom berieme do úvahy aj malú a veľkú verziu znakov. Klávesy ‘A’ a ‘D’ fungujú ako alternatíva k šípkam. Klávesa ‘S’ funguje rovnako ako fyzické tlačidlo, slúži na prepínanie medzi automatickým a manuálnym režimom.

```
250 void keyPressed() {
251   if (key == 'm' || key == 'M') {
252     showMenu = !showMenu;
253     if (!showMenu) {
254       fill(0);
255       noStroke();
256       rect(0, 0, 300, 700);
257     }
258   }
259 
260   if (key == CODED) {
261     if (keyCode == LEFT) {
262       port.write('a');
263     } else if (keyCode == RIGHT) {
264       port.write('d');
265     }
266   } else {
267     if (key == 'a' || key == 'A') {
268       port.write('a');
269     } else if (key == 'd' || key == 'D') {
270       port.write('d');
271     } else if (key == 's' || key == 'S') {
272       port.write('m');
273     }
274   }
275 }
```

Obr. 18 - Ovládanie

(Zdroj: Vlastné spracovanie, rok: 2026)

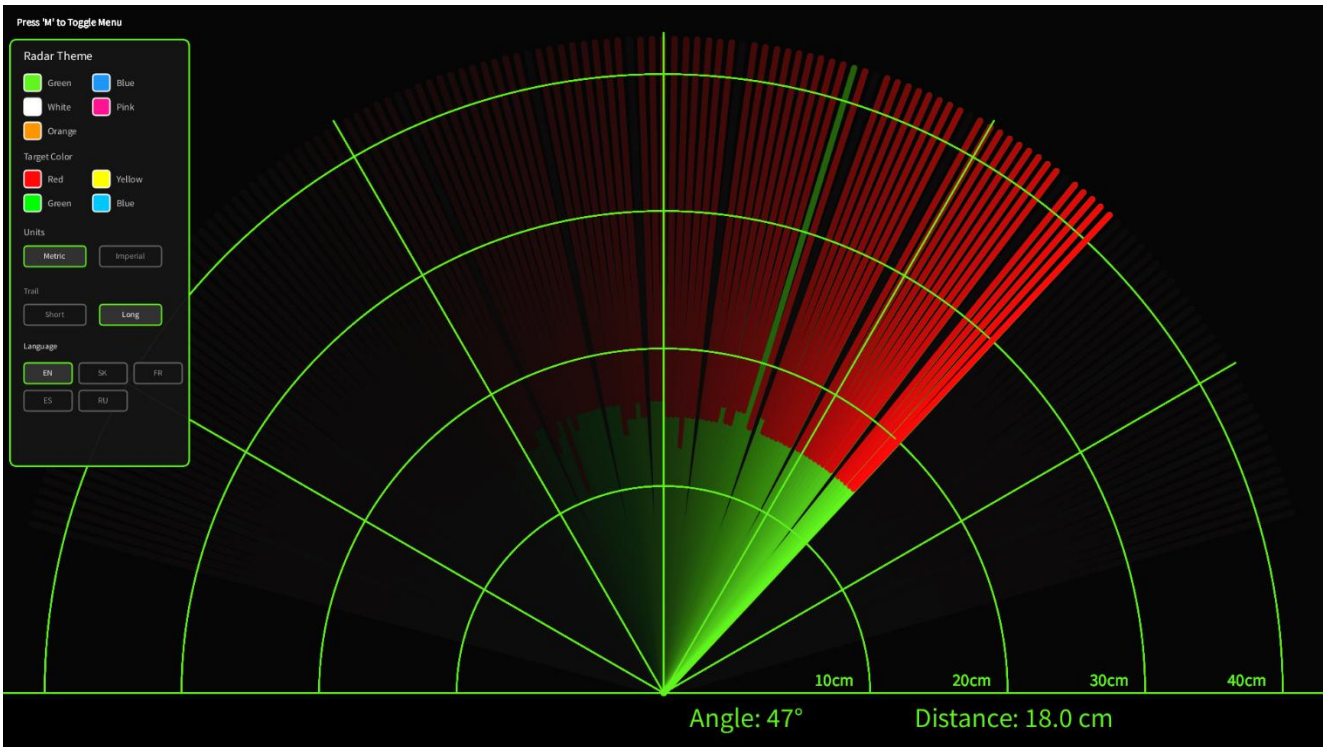
Pomocou funkcie *keyPressed()* detegujeme súradnice myši a umožňujeme:

- Zmenu farieb radarov a cieľov
- Prepínanie jednotiek medzi metrickými a imperiálnymi

- Nastavenie dĺžky stopy (krátka/dlhá) úpravou premennej *fadeAmount*
- Výber jazyka rozhrania prechádzaním poľa *langNames*

Celý zdrojový kód aplikácie sa nachádza v prílohe B.

Finálna podoba aplikácie je zobrazená na obrázku 19.



Obr. 19 - Finálna podoba aplikácie

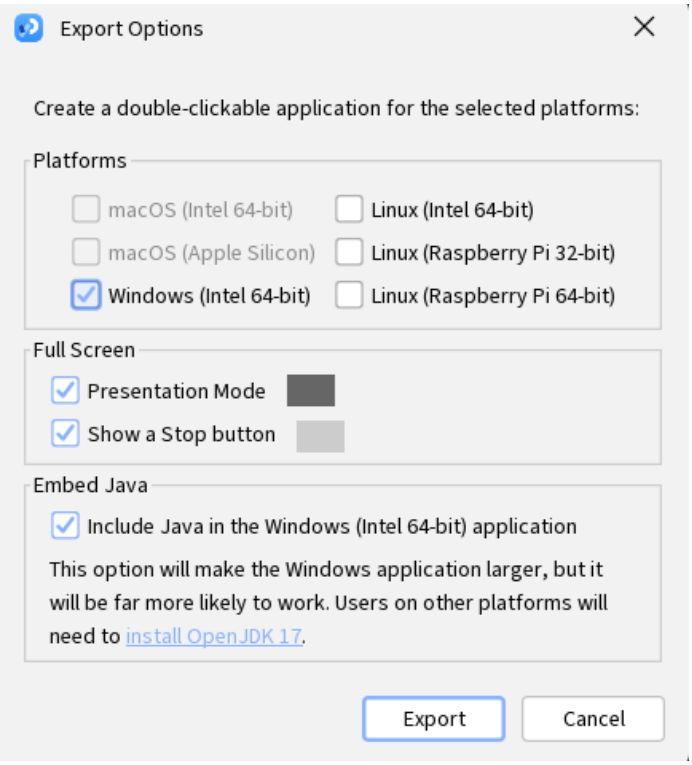
(Zdroj: Vlastné spracovanie, rok: 2026)

7.4 EXPORT APLIKÁCIE

Proces exportu premení zdrojový kód s príponou .pde na plnohodnotnú aplikáciu .exe. V prostredí Processing toto docielime cez ponuku File -> Export Application (alebo skratkou Ctrl + Shift + E). Pri exporte musíme zohľadniť nasledujúce nastavenia:

- **Voľba platformy**: cieľový operačný systém (Windows, Linux alebo MacOS). Pre našu aplikáciu zvolíme možnosť Windows (Intel 64-bit).
- **Full Screen**: ak chceme, aby sa naša aplikácia spustila v režime celej obrazovky, potrebujeme zaškrtnúť možnosť Presentation Mode.

- **Embed Java**: táto možnosť sa odporúča zaškrtnúť, aby sme zabezpečili, že aplikácia bude fungovať aj na počítačoch, ktoré nemajú nainštalované prostredie Java, čím sa zvyšuje kompatibilita.



Obr. 20 - Nastavenia exportu aplikácie

(Zdroj: Vlastné spracovanie, rok: 2026)

8 ZOZNAM POUŽITEJ LITERATÚRY

[1] RADAR. Wikipedia, The Free Encyclopedia. [online]. San Francisco (CA): Wikimedia Foundation, 2025. [cit. 2025-12-12]. Dostupné na internete: https://en.wikipedia.org/wiki/Radar

[2] RADAR. Wikipédia, slobodná encyklopédia. [online]. San Francisco (CA): Wikimedia Foundation, 2023. [cit. 2025-12-12]. Dostupné na internete: https://sk.wikipedia.org/wiki/Radar

[3] LUCAS-NÜLLE GmbH. Vzdelávací systém pre modernú radarovú techniku. Ref.-Nr.: P3160. [Online]. Lucas-Nülle GmbH, [s. a.]. [cit. 2025-12-12]. Dostupné na internete:https://admin2771.webygroup.sk/Data/2771/UserFiles/2018/ln/radarova_technika_sk.pdf

[4] BÖTTCHER, R. Základy radiolokace. Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2025. [cit. 2025-12-12]. Dostupné na internete: https://www.radartutorial.eu/01.basics/rb04.cz.html

[5] BÖTTCHER, R. Radar Cross Section. Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2025. [cit. 2025-12-12]. Dostupné na internete: https://www.radartutorial.eu/01.basics/Radar%20Cross%20Section.en.html

[6] BÖTTCHER, R. Přesnost měření a rozlišovací schopnost. Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2025. [cit. 2025-12-12]. Dostupné na internete: https://www.radartutorial.eu/01.basics/rb07.cz.html

[7] BÖTTCHER, R. Radar Transmitter. Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2025. [cit. 2025-12-12]. Dostupné na internete: https://www.radartutorial.eu/08.transmitters/Radar%20Transmitter.en.html

[8] BÖTTCHER, R. Typy antén. Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2025. [cit. 2025-12-12]. Dostupné na internete: https://www.radartutorial.eu/06.antennas/an05.cz.html

[9] BÖTTCHER, R. Radarová soustava pro řízení letového provozu (Air Traffic Control). Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2025 [cit. 2025-12-12]. Dostupné na internete: https://www.radartutorial.eu/06.antennas/an21.cz.html

[10] Kvalita zobrazenia – faktory, ktoré ovplyvňujú kvalitu zobrazenia. [online]. Praha: Ministerstvo obrany ČR, 2008. [cit. 2025-12-12]. Dostupné na internete: https://www.mo.gov.cz/images/id_8001_9000/8753/radar/k25.html

[11] BÖTTCHER, R. Přijímač radarového systému. Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2021. [cit. 2026-01-02]. Dostupné na internete: https://www.radartutorial.eu/09.receivers/rx05.cz.html

[12] BÖTTCHER, R. Radar Receiver. Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2007. [cit. 2026-01-02]. Dostupné na internete: https://www.radartutorial.eu/09.receivers/rx05.en.html

[13] BÖTTCHER, R. A-Scope. Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2007. [cit. 2026-01-02]. Dostupné na internete: https://www.radartutorial.eu/12.scopes/sc04.en.html

[14] BÖTTCHER, R. Plan Position Indicator (PPI). Radar Tutorial. [online]. Nemecko: Christian Hülsmeyer, 2007. [cit. 2026-01-02]. Dostupné na internete: https://www.radartutorial.eu/12.scopes/sc13.en.html

[15] HISTORY OF RADAR. Encyclopedia Britannica. [online]. Chicago (IL): Encyclopædia Britannica, Inc., 1998, 2015. [cit. 2026-01-05]. Dostupné na internete: https://www.britannica.com/technology/radar/History-of-radar

[16] TECHNIKY PREHĽADOVEJ RADIOLOKÁCIE LIETADIEL VO VZDUŠNOM PRIESTORE I. Airliners.sk. [online]. Bratislava: Airliners.sk, 2017. [cit. 2026-01-05]. Dostupné na internete: https://www.airliners.sk/techniky-prehladovej-radiolokacie-lietadiel-vo-vzdusnom-priestore-i/

[17] ARDUINO, PROCESSING, WIRING - HOW DO THEY RELATE? Arduino Forum. [online]. Arduino Cloud, 2008. [cit. 2026-01-09]. Dostupné na internete: https://forum.arduino.cc/t/arduino-processing-wiring-how-do-they-relate/2466

[18] NEDELKOVSKI, Dejan. Ultrasonic Sensor HC-SR04 and Arduino Tutorial. HowToMechatronics. [online]. 2022 [cit. 2026-02-06]. Dostupné na internete: https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/

[19] SG90 Micro Servo Datasheet. [online]. London: Imperial College London, 2026 [cit. 2026-02-06]. Dostupné na internete: http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf

[20] Čo treba vedieť o rezistoroch – typy rezistorov, ich zapojenie a výpočet odporu. TME: Transfer Multisort Elektronik. [online]. Łódź: TME, 2022 [cit. 2026-02-06]. Dostupné na internete: https://www.tme.eu/sk/news/library-articles/page/45818/co-treba-vediet-o-rezistoroch-typy-rezistorov-ich-zapojenie-a-vypocet-odporu/

9 PRÍLOHY

PRÍLOHA A – ZDROJOVÝ KÓD PRE ARDUINO

PRÍLOHA B - ZDROJOVÝ KÓD

PRÍLOHA C - USB KĽÚČ

PRE APLIKÁCIU