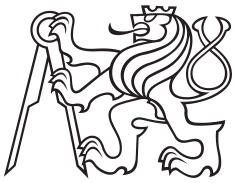
Diplomová práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra kybernetiky

Platforma pro snadné testování reakcí, postřehu a dalších kognitivních schopností

Bc. Matěj Štula

Vedoucí: Ing. Petr Novák, Ph.D.
Obor: Kybernetika a robotika
Leden 2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Štula** Jméno: **Matěj** Osobní číslo: **474601**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra kybernetiky**

Studijní program: **Kybernetika a robotika**

Studijní obor: **Kybernetika a robotika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Platforma pro snadné testování reakcí, postřehu a dalších kognitivních schopností

Název diplomové práce anglicky:

A Platform for Easy Testing of Reactions, Perceptions and Other Cognitive Abilities

Pokyny pro vypracování:

V mnoha případech, a i celkem opakovaně, jsou potřeba různé (převážně rychlé) testy pro vyšetření reakce, postřehu a mnoha dalších typů kognitivních schopností. Testy se většinou skládají ze souboru podnětů, odpovědí / reakcí a poté jejich vyhodnocení. Vzhledem k podobnosti těchto testů lze vytvořit vhodnou platformu pro jejich snadné sestavení podle aktuální potřeby.
1) Prostudujte různé testy využívané pro vyšetření zejména reakce, postřehu a dalších kognitivních schopností člověka (paměť, orientace, ...). Dále najděte jejich společné vlastnosti jako jsou: požadavky, použitá zařízení, konfigurace, průběh, hodnocení atd.
2) Navrhněte a sestavte platformu obsahující k tomuto účelu vhodná zařízení (světla, tvorba zvuků, zobrazení obrázků, tlačítka pro odpovědi, ...) a jejich vhodné propojení (USB, RS485, LAN, ...) pro snadné vytváření těchto testů přímo na míru.
3) Rovněž navrhněte a vytvořte aplikaci pro snadnou konfiguraci (tvorbu) testu podle aktuálních požadavků a připojených typů zařízení. Součástí by měla být i vhodná prezentace získaných výsledků, a to formou tabulek či grafů.
4) Demonstrujte použitelnost navržené platformy formou vytvoření několika jednoduchých testů ze zmíněného cílového oboru.

Seznam doporučené literatury:

[1] WWW stránky nalezených / obdobných projektů, manuály a další relevantní informace
[2] Price Mark, C# 8.0 and .NET Core 3.0 - Modern Cross-Platform Development, Packt, 2019
[3] MacDonald Matthew, Pro WPF 4.5 in C#, APress, 2012
[4] Noviello Carmine, Mastering STM32, LeanPub, 2017
[5] (další potřebné materiály poskytne vedoucí práce)

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Petr Novák, Ph.D. oddělení kognitivních systémů a neurovědy CIIRC

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **10.01.2022** Termín odevzdání diplomové práce: **10.01.2023**

Platnost zadání diplomové práce: **30.09.2023**

Ing. Petr Novák, Ph.D.	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. _____
Datum převzetí zadání

_____
Podpis studenta

Poděkování

Chtěl bych poděkovat všem, kteří mne během studia podporovali. Tedy zejména rodině a kamarádům.

Také bych rád poděkoval svému vedoucímu práce Ing. Petru Novákovi, Ph.D. za vedení této práce. Též děkuji za jeho pomoc a nesčetné rady během tvorby zde popsané platformy.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Liberec, 8. ledna 2023

. . . . . .

Abstrakt

Pro testování a procvičování kognitivních schopností existuje mnoho testů. Tyto však jsou často neměnné a neobsahují ani žádnou složku náhody. U takovýchto testů hrozí, že si je člověk při opakování zapamatuje, což není žádoucí. Takové testy nelze ani upravovat pro speciální potřeby testovaných nebo pro experimentální a vědecké využití.

Tyto testy obsahují podobné sady úloh, a proto byla navržena univerzálnější testovací platforma pro testování a také procvičování kognitivních schopností. Tato platforma se skládá ze tří hlavních částí - aplikace, hardwarových zařízení a koncových prvků. Aplikace slouží pro vytváření a správu testů. Zařízení umožňují připojit k aplikaci libovolný počet koncových prvků poskytujících testovanému podněty a snímajících jeho reakce. Platforma je připravena pro připojení mnoha typů výstupů i vstupů. Další je možné snadno nadefinovat dle aktuálních potřeb.

Samotný test lze sestavit přímo v aplikaci. K tvorbě testu je připraveno mnoho kroků testu pro ovládání výstupů/vstupů, opakování částí testu, podmínky apod. Do testů lze také vložit kroky s náhodnými prvky, jež proměňují každý průchod.

Výstupem z testu je časový záznam všech událostí v testu. Tato práce se nijak nezabývá samotným vyhodnocením získaných dat.

Klíčová slova: kognitivní schopnosti, podněty, testování, trénování, reakce, postřeh, zařízení, USB

Abstract

There are many tests for the testing and training of cognitive functions. However, these are often immutable and do not contain any randomness. There is a risk that a testee will remember such tests when repeating them, which is not desirable. Nor can such tests be modified for special needs of testees or for experimental and scientific use.

These tests contain similar sets of tasks, and therefore a more universal test platform was designed for testing and practicing cognitive functions. This platform consists of three main parts - application, hardware devices and end elements. The application is used for creating and managing tests. The devices make it possible to connect any number of end elements to the application providing stimuli to the testees and sensing their reactions. The platform is ready to connect many types of outputs and inputs. Others can be easily defined according to the current needs.

The test itself can be created directly in the application. To create a test, many test steps are prepared for controlling outputs/inputs, repeating parts of the test, conditions, etc. Steps can be also inserted into tests with random elements that changes on each pass.

The output of the test is a time record of all events in the test. This work does not deal with the actual evaluation of the obtained data.

Keywords: cognitive function, stimuli, testing, training, reaction, observation, device, USB

Title translation: A Platform for Easy Testing of Reactions, Perceptions and Other Cognitive Abilities

Obsah

Obrázky

Kapitola 1
Úvod

Kognitivní schopnosti slouží člověku nejen k poznávání okolního světa, ale rovněž ke správné interakci s ním. Při oslabení těchto schopností se člověku stěžuje činnost v běžném životě. Kognitivní schopnosti se vyvíjejí už u malých dětí, ve středním věku jsou na svém vrcholu a poté oslabují během procesu stárnutí. Omezeny mohou být také i z důvodu nemoci, úrazu či intoxikace. Mezi běžné příčiny postižení patří rovněž úrazy hlavy či stále častěji tzv. degenerační nemoci jako jsou Alzheimerova nebo Parkinsonova nemoc.

Člověk může samozřejmě při dostatečném tréninku svého mozku zpomalit postupné oslabování svých kognitivních schopností. V současné době tedy existují již testy nejen pro detekci úbytku těchto schopností, ale i testy pomáhající člověku s jejich trénováním/udržováním. Tyto testy se stále vyvíjejí, avšak ve většině případů se jedná bohužel o opakování stejného testu stále dokola. V takovém případě se člověk může postupem času naučit správná řešení v podstatě zpaměti, a poté již test začíná postrádat svůj původní smysl.

Testy existují v několika provedeních od neměnných papírových formulářů a dotazníků, přes webové či mobilní aplikace vytvořené jako soubor úloh určených zejména pro cvičení. Až po skutečně specializované testy probíhající rovněž často i na specifickém hardwaru. Nejčastěji používané testy buďto nelze upravit vůbec, případně pouze za součinnosti autora, a tedy je rozsah jejich modifikace poněkud omezen.

Nabízí se myšlenka vytvořit systém umožňující poněkud univerzálnější nejen testování, ale současně i procvičování kognitivních schopností. Systém, jenž by bylo současně možné, do jisté míry, snadno upravovat dle požadavků aktuálně testovaných/trénovaných osob. Jelikož jednotlivé testy využívají různé prvky jako výstupy (podněty) či vstupy (snímání), tak by takováto testovací platforma měla být do budoucna snadno rozšiřitelná i o další prvky, které nebudou její součástí od samého začátku. Hlavní rozdíl zamýšleného systému od stávajících souborů testů by měla být možnost testy nejen upravovat, ale dokonce vytvářet zcela nové s použitím stejného souboru hardwaru (výstupů/vstupů). Dále platforma musí umožnit vytvářet určitou náhodnost při jejich průběhu. Tímto se zabrání, aby se testovaný naučil, jak má test správně probíhat, čímž by v podstatě postupně test ztratil svůj význam.

Kapitola 2
Kognitivní funkce a jejich testování

V této kapitole jsou popsány kognitivní schopnosti člověka a různé možnosti pro jejich testování či procvičování. Na konci je shrnut aktuální stav těchto testů.

2.1 Kognitivní funkce

Kognitivní funkce patří mezi nejzákladnější smyslové schopnosti a umožňují nejen lidem, ale i zvířatům, vnímat okolní svět a vhodně s ním interagovat. Mezi tyto schopnosti patří orientace, paměť a poznání, ale také pozornost, řeč a využití získaných informací. V dalším textu je rozebrán zejména význam těchto schopností pro člověka a možnosti jejich testování a trénování.

Orientovat se lze v prostoru a čase, jedná se tedy o schopnost vnímat a chápat, kde se osoba nyní nachází, odkud a kam se přesouvá nebo v jaké je pozici, a také kde je nahoře či dole. U vnímání prostoru a vzdáleností lze rozlišit tři pohledy. Osoba může vnímat směr a vzdálenost od sebe k cílovému předmětu. Dále může vnímat vzdálenosti mezi různými předměty v okolí, nezávisle na tom, kde se osoba sama nachází. Třetí pohled je o vnímání vlastního pohybu, například pokud má osoba zavřené oči.[1] Také se lze orientovat v čase, tedy jaký je nyní čas, jaký je den, rok či roční období. Rovněž jak dlouho trvá nějaká činnost, a která ze dvou různých pozorovaných činností trvá déle.[2]

Orientovat se lze také v osobní rovině, tedy kdo „já“ jsem, dokázat poznat sám sebe a schopnost uvědomění si sama sebe, tedy i svého chování. K tomu patří vědomí toho, co umím a co znám. Díky orientaci osoba ví, co má v daných situacích dělat, například uhnout před padajícím stromem či si uprostřed dne dát oběd.[2]

Pozornost je širší téma pokrývající rychlost zpracování informací a schopnost soustředit se na plnění vybraného úkolu. Dovoluje osobě zaměřit se na jednu činnost a tu se snažit dokončit. Pozornost ovlivňuje také reakční dobu, tedy interval mezi stimulací a patřičnou reakcí. Problém u pozornosti může být jak neschopnost ji udržet, tak rovněž neschopnost přepnout na jinou činnost. Úroveň pozornosti může být ovlivněna různými stavy osoby.[2, 3]

Paměť a učení slouží k uchování získaných informací a následně k jejich opětovnému vybavení. Důležitá je i schopnost dokázat pracovat s vybavenými

vzpomínkami. Paměť se dělí na krátkodobou a dlouhodobou. Osoba si pamatuje, co se kdy stalo. Také si může pamatovat obecné vědomosti a do třetice slouží paměť k naučení se často používaných postupů a akcí.

Řeč a zpracování informací umožňují vytvářet a chápat spojitosti mezi kódy a objekty. Za kódy mohou být považována slova, obrázky či jakékoli jiné symboly. Dále k tomuto patří schopnost porozumění pokynům. Řeč, respektive jazyk, pokrývá několik funkcí, například slovník, schopnost pojmenovávat objekty a události, schopnost popsat a pochopit objekt, ale také schopnost plynule tvořit jazykový obsah a též dovednost se srozumitelně vyjadřovat.[2]

S omezenými kognitivními schopnostmi má jedinec ztížené vnímání okolního světa. Zároveň může takový deficit být příznakem různých chorob, jejichž včasná detekce zvyšuje šanci na jejich lehčí průběh. Mezi takové nemoci patří v současné době zejména Alzheimerova nemoc, stařecká demence a Parkinsonova choroba. Neblahý vliv na aktuální stav kognitivních schopností může mít též únava či intoxikace organismu.

2.2 Testy pro kognitivní funkce

Kognitivní schopnosti jsou často vcelku komplexní, a proto byly v průběhu mnoha let vytvořeny různé sady testů na detekci jejich stavu a/nebo poruch. Prvotní testy byly v podstatě pouze papírové a poskytovaly velmi malou variabilitu, při jejich častém opakování se je pacienti postupem času dokázali naučit z paměti, a tím takový test pozbyl svého významu. S nástupem počítačů začaly vznikat testy jako již variabilnější a rovněž poněkud komplexnější aplikace.

Papírové i elektronické testy se mohou zaměřovat na více schopností testované osoby najednou, což slouží k posuzování celkového stavu jedince. Nejen pro procvičování existují ovšem testy zaměřené i pouze na jednu z těchto schopností, případně pouze na jeden aspekt vybrané kognitivní schopnosti.

Dále je uvedeno, jak se v současné době nejčastěji testují jednotlivé kognitivní schopnosti. První zde popsané testy jsou určené pro orientaci. Testovaný má za úkol odpovědět, kde se nalézá, tedy například v jakém městě či v jaké budově. Podobné otázky jsou i pro časovou orientaci, aneb zda testovaný ví, jaká je roční doba či den. Další testy se zaměřují na posloupnosti událostí a testovaný je má správně seřadit. Testy tyto otázky obsahují často čistě ve formě dotazníku. Existují i elektronické verze, kde se testovanému zobrazí zadání a testovaný jej interaktivně řeší. Viz obrázek 2.1, na němž má testovaný seřadit jednotlivé části dle toho, jak obrázek vznikal.

Také je potřeba zmínit test, při kterém se sleduje, jak se testovaný pohybuje v prostoru. Pro tyto účely se používá například tzv. Blue Velvet Arena, tedy uzavřený stan, v němž se testovaný orientuje pouze pomocí dočasně rozsvěcovaných značek na stěnách a podlaze.[1]

Obrázek 2.1: Ukázka internetového testu „Orientace v čase a prostoru - Časová posloupnost 2“.[4]

Další důležitou schopností je paměť. Testovat lze krátkodobou i dlouhodobou paměť. K zapamatování lze použít různé prvky, například:

- psaný text přečtený testovaným,
- zvuky či slova přehrávaná testovanému,
- symboly či sekvence symbolů zobrazené testovanému,
- úkoly vykonané testovaným.

Testovaný si má po určité době tyto podněty vybavit.

Ve většině testů testovaný opakuje či vybírá symboly, jež si předtím zapamatoval. Také může určovat, zda nové symboly odpovídají těm původním. U testů krátkodobé paměti se opakuje několik testů po sobě. Při testu dlouhodobé paměti má testovaný pro kontrolu zadání zopakovat, aby se předešlo chybným výsledkům, kdy si testovaný nezapamatuje zadané symboly. Po nějaké době je testovaný tázán, jaké byly původně zadané symboly. Tento dotaz bývá položen opakovaně s časovými odstupy.

Poslední zde rozebírané možnosti testování patří k pozornosti a reakcím testovaného. Pozornost se většinou testuje jako časová prodleva mezi podnětem pro testovaného a jeho reakcí, případně se sleduje, na jaké podněty či s jakým zpožděním testovaný reaguje. Podněty mohou být například zvukové či vizuální. Podněty lze samozřejmě kombinovat, tedy testovaný má reagovat pouze tehdy, nastane-li více podnětů najednou. Požadovaná reakce testovaného je většinou stisk nějakého tlačítka.

Jako příklad lze uvést test nazvaný Batak sloužící k procvičování pozornosti a v některých konfiguracích slouží i k fyzickému procvičení. Myšlenka testu je jednoduchá. Jeho platforma sestává z několika podsvícených tlačítek, kde se v náhodných intervalech rozsvěcí příslušná světla. Úkol je co nejrychleji stisknout právě rozsvícené tlačítko.[5]

Pozornost se též testuje pomocí hledání rozdílných či naopak stejných obrazců, symbolů či jiných podnětů. Jako příklad lze uvést zkoušku vizuální

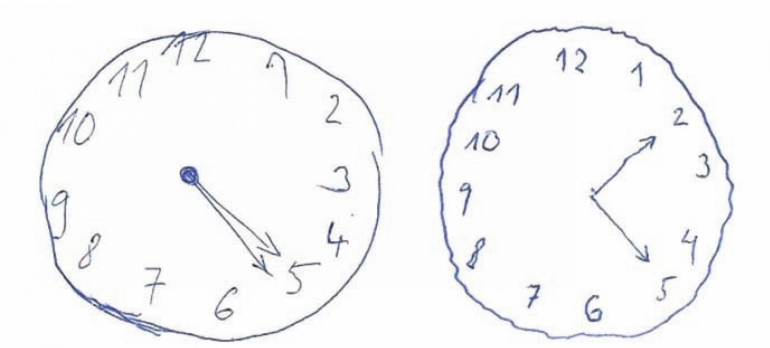
pozornosti z fakultní nemocnice Brno. Testovanému je nejprve vysvětlen jeho úkol v testu spočívající v zakroužkování všech dvojic symbolů shodujících se s úvodní dvojicí symbolů. Následuje pokus, který je následně před testovaným opraven, a třetí pokus je již hodnocen.[6]

2.3 Příklady používaných testů

V této podkapitole je popis několika vybraných konkrétních testů používaných pro detekci snížení kognitivních schopností nebo k jejich procvičování. Pro záchyt demence a orientační kontrolu stavu kognitivních schopností se používá test nazvaný MMSE (Mini Mental State Exam). Jedná se o dotazníkový test obsahující 30 otázek či úkolů, v nichž může pacient získat až 30 bodů. Úkoly jsou zaměřeny na orientaci v místě i čase, zapamatování si údajů i jejich opětovné vybavení, řeč, příkazy a také psaní a kreslení. Zdravý jedinec by měl dosáhnout výsledku v rozmezí 27-30 bodů.[7]

Další test s podobným zaměřením je MoCA (Montrealský kognitivní test). Podobně jako MMSE se jedná o sadu úkolů hodnocených až 30 body. Tento test je oproti úkolům z MMSE více zaměřený na vizuální vnímání. Mezi úkoly je mimo jiné pojmenovávání zvířat na obrázku, kreslení hodin a také hledání specifického symbolu mezi jinými a opakování pokynů zadávajícího.[8]

Jako rychlý orientační test pro detekci demence se používá i samostatný test na kreslení hodin. Pacient dostane za úkol nakreslit ciferník, jehož ručičky mají ukazovat zadaný čas. Poté se hodnotí jednotlivé grafické prvky nakreslených hodin. Tedy označení všech dvanácti hodin po obvodu ciferníku, vedle existence čísel se hodnotí i jejich správné umístění. Dále se hodnotí hodinové ručičky, zda jsou právě dvě a zda ukazují požadovaný čas. Další body může pacient získat, pokud ručičky mají zřetelný rozdíl ve své délce a jsou správně nastaveny.[9] Na obrázku 2.2 je ukázka dvou chybně nakreslených ciferníků se zadaným časem 5 h 10 min. Oba ciferníky mají chybně stejně dlouhé ručičky a varianta vlevo zobrazuje dokonce nesprávný čas.



Obrázek 2.2: Příklady chybně nakreslených ciferníků se zadaným časem 5:10.[9]

Nedostatky kognitivních funkcí se mohou projevit i při orientaci a pohybu osoby. Zkoumat pohyb v reálném prostředí je náročné, a proto nejen v Laboratoři prostorové kognice na Neurologické klinice 2. lékařské fakulty UK je

k dispozici tzv. Blue Velvet Arena. Jedná se o kruhovou arénu o průměru asi 3 metry bez stálých orientačních bodů, v níž je sledován pohyb pacienta.[10]

Na závěr je potřeba zmínit i různé internetové testy na procvičování kognitivních schopností. Jejich výhoda spočívá v možnosti testování přímo z pohodlí domova. Jelikož se jedná o testy na počítači, tak se nabízí možnost, aby jejich obsah byl při generování nového zadání vždy aspoň částečně modifikován. Jako příklad lze uvést online test COGIT. Tento test hodnotí pozornost a práci s rušivými vlivy. Dále také hledání vzoru mezi zobrazenými symboly a jednoduché výpočty. Test COGIT ale poskytuje pouze dva předem definované průběhy.[11] K procvičování lze použít i testy primárně určené pro malé děti, například cvičení dostupná na serveru Školákov. Zde lze najít cvičení na pozornost, hledání či opakování vzorů a mnoho dalších.[12]

2.4 Zhodnocení současného stavu

V současné době se při podezření na indispozici kognitivních schopností používají v lékařství převážně standardizované testy. Mezi takové testy patří například již zmíněné MoCA a MMSE testy. Oba tyto testy jsou hodnoceny do 30 bodů a dokáží dostatečně rozlišit poruchu kognitivních schopností od normálního stavu. Dále se často k detekci demence používá test kreslení hodin.

Výše zmíněné testy jsou standardizované a mají normované výsledky. Standardizace usnadňuje vyhodnocení stavu. Toto je vhodné pro detekci úbytku schopností. Při častém opakování testu může ale dojít k ovlivnění výsledků, jelikož si testovaný může průběh testu postupem času zapamatovat. Vzhledem k normovaným zadáním nelze bohužel testy pro pacienty nijak znatelněji upravovat.

Kognitivní testy lze využívat samozřejmě i k procvičování kognitivních schopností. Pro tento účel se existující testy často nejen upravují, ale rovněž stále vyvíjejí nové. Takto vznikla myšlenka této práce, vytvořit poněkud univerzálnější systém, kde si může lékař snadno „naklikat“ příslušný typ testu podle vlastní potřeby, a kdykoli si jej i podle aktuálního stavu testovaného dále modifikovat. Tímto by se testy staly nejen ještě více variabilnější, ale současně i snadno přizpůsobitelné stavu aktuálně testované osoby.

Jelikož existuje mnoho typů testů je potřeba pro tvorbu zamýšlené univerzálnější testovací platformy vybrat vhodné skupiny těchto testů, jež by mohla zahrnovat. Tedy testy, které lze velmi dobře převést do počítačové podoby, snadno modifikovat a rovněž do nich vložit určitou složku náhody proti zapamatování si jejich stále se opakujícího průběhu. Jako vzorové testy pro vývoj univerzální testovací platformy byly vybrány následující základní typy/oblasti testů:

- reakce na světelné a zvukové podněty,
- reakce na posloupnost podnětů,
- zapamatování si posloupnosti podnětů,

- získání informace ze zarušeného textu,
- určení času na hodinách.

Kapitola 3

Cíle práce

Tato práce si klade za cíl vytvořit experimentální platformu sloužící k testování postřehu a dalších běžných kognitivních schopností člověka. V první části byly popsány existující testy používané pro testování již zmíněných schopností. Jednotlivé testy se liší, ale přesto mají často i společné znaky/postupy, na nichž je založena myšlenka dále vytvářené testovací platformy.

V další části práce je navržena základní struktura platformy. Budou vybrány hlavní části představující základní stavební kameny pro platformu a budou vytvořena jejich vzájemná propojení. Dále budou definovány prvky připojitelné do platformy za účelem vytváření požadovaných testů.

Poté bude navržena a vytvořena počítačová aplikace umožňující snadnou konfiguraci jednotlivých testů stejně jako jejich spouštění a zobrazení získaných výsledků. V této části je též popis variability vytvářených testů ukazující skutečné možnosti a využití vytvořené testovací platformy.

V poslední části práce jsou prezentovány dosažené výsledky. Navržená platforma bude experimentálně (v základní sestavě) realizována a bude pro ni vytvořeno několik vzorových testů. Ty lze upravovat a rovněž mohou mít pokaždé poněkud odlišný průběh. Dále bude prezentováno vzorové použití některých navržených komponent v prostorách Národního ústavu duševního zdraví pro testy orientace.

Tato práce se však nikterak nevěnuje těmto oblastem:

- lékařskému a psychiatrickému oboru,
- vyhodnocení získaných dat,
- uchování a zabezpečení získaných dat,
- spolehlivosti a bezpečnosti celého systému/platformy (jedná se o experimentální řešení).

Hlavním úkolem této práce tedy je vytvořit experimentální testovací platformu a demonstrovat možnost jejího reálného využití (nikoli přímo odborné využití).

Kapitola 4

Použité pojmy

Pro celkovou přehlednost této práce byly zavedeny některé pojmy velmi často využívané v dalším textu. Zde jsou tyto základní pojmy podrobněji vysvětleny:

Testovací platforma (platforma).
Celkový vytvořený systém pro testování nejen kognitivních funkcí. Platforma tvoří libovolný počet externích zařízení s napojenými koncovými prvky (výstupy = aktuátory a vstupy = senzory) a z řídicí aplikace (na PC). Platforma může samozřejmě sloužit rovněž i k procvičování kognitivních funkcí.

Test.
Posloupnost událostí jako série vytvářených podnětů, na něž testovaný reaguje. Podněty jsou výstupy z platformy a reakce testovaného jsou vstupy do platformy. Průběh testu představuje vykonávání posloupnosti jednotlivých kroků testu. Test může také reagovat i přímo na chování testovaného, čímž lze průběh testu upravovat podle aktuálních reakcí/chování testovaného.

Krok testu.
Dílčí položka v testu. Může se jednat například o nastavení hodnoty na výstupu, zjištění vstupu nebo zobrazení poznámky o části testu. Existují i kontrolní kroky testu řídící jeho průběh.

Testovaná osoba (testovaný).
Osoba podstupující test. Testovaný reaguje na podněty vytvářené pomocí výstupů platformy (aktuátorů) a jeho reakce jsou snímány pomocí k tomu určených vstupů platformy (senzorů).

Koncový prvek (element).
Představuje reálný výstup z platformy či vstup do ní. Jedná se o aktuátory či senzory, s jejichž pomocí test interaguje s testovaným. Jeden koncový prvek může slučovat více reálných výstupů či vstupů.

Zařízení.
Externí HW modul umožňující připojení předem daných koncových prvků (elementů poskytujících výstupy a/nebo vstupy). Zařízení komunikuje s řídicí

aplikací. Přijímá z aplikace pokyny pro nastavení výstupů (podnětů pro testovaného) a současně aplikaci zpět zasílá informace o změnách na připojených vstupech (reakce od testovaného).

Komunikace.
Způsob předávání instrukcí a událostí mezi řídicí aplikací a jednotlivými zařízeními. Tento pojem zahrnuje nejen použitý protokol pro přenos informací, ale současně i vybrané způsoby propojení aplikace se zařízeními podle požadavků platformy.

Aplikace.
V podstatě hlavní řídicí část celé platformy. Umožňuje sestavit test a poté jej vykonávat/řídit. Rovněž zobrazuje aktuální průběh testu a poskytuje jeho výsledky, tedy záznam podnětů a reakcí. Neobsahuje však žádné vyhodnocení získaných dat. Zmíněná aplikace se skládá ze dvou částí. První část se stará o komunikaci se zařízeními a o předávání informací mezi nimi a druhou částí aplikace. Druhá část zahrnuje grafické rozhraní a dovoluje tedy správu testů, jejich úpravu a též samotné vykonávání.

Tester.
Osoba připravující a spouštějící test. Tato osoba zodpovídá za přípravu testu, připojení potřebných zařízení s elementy (výstupy/vstupy) a rovněž za průběh/vykonání testu. Tester by měl test dobře znát a testovanému dostatečně srozumitelně vysvětlit jeho úkoly. Po spuštění testu tester dohlíží na průběh testu a může jej kdykoli pozastavit nebo i zcela ukončit. Po skončení testu se testerovi v aplikaci zobrazí výsledky vykonaného testu, tedy chronologický seznam akcí platformy a reakcí testovaného. Tyto výsledky může tester sám vyhodnotit či je předat k vyhodnocení komukoli jinému.

V běžném, zejména počítačovém popisu, se používá označení přenosu dat nebo informací formou vstup/výstup. V popisu zde uvedené platformy je však použit formát výstup/vstup. Tento formát lépe vystihuje činnost celé platformy, a tedy vykonávaného testu. Platforma nejprve generuje výstup (podnět pro testovaného), na něj poté reaguje testovaná osoba a teprve nakonec platforma snímá vstup (reakci testovaného).

Kapitola 5

Struktura testovací platformy

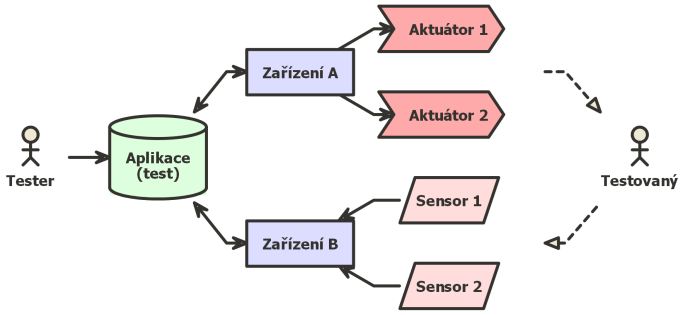
V této kapitole je popsán zamýšlený účel platformy. Tedy jaké úkoly bude vytvářená platforma plnit a z jakých částí bude sestavena. Dále je popsáno, jak jsou jednotlivé součásti vzájemně propojeny. Jak bylo popsáno v úvodu, v současnosti používané testy jsou ve většině případů neměnné. Pokud testovaný podstoupí jeden test opakovaně, tak má test pokaždé v podstatě stejný průběh. Ještě výraznější je tato skutečnost v případě, kdy se test opakuje s krátkými časovými odstupy. Jako příklad lze uvést nejčastěji používané papírové testy. Výjimkou jsou testy pozornosti typu Batak, mající pokaždé jiný průběh, ale vždy stejnou podstatu. Testy v elektronické podobě již mohou mírně modifikovat svůj průběh, ale nelze jednoduše změnit jejich podstatu ani k nim přidávat nové prvky/kroky.

Hlavním nedostatkem, jenž se zde navrhovaná platforma pokusí odstranit, jsou obtížné změny stávajících testů a tvorba zcela nových testů. Platforma musí nejen dovolit vytvářet testy, ale rovněž je i později modifikovat. Kromě vytváření podnětů testovanému je důležité i snímání jeho reakcí. Součástí testů jsou tedy nejen výstupy, na něž může testovaný reagovat, ale i vstupy snímající jeho reakce. Aby bylo pokryto široké spektrum požadavků a testů a současně byla platforma dostatečně jednoduchá, tak by jednotlivé výstupy a vstupy měly být připojitelné do platformy vždy dle požadavku daného testu. Platforma může podporovat mnoho různých typů výstupů a vstupů, ale není nutné, aby všechny podporované byly vždy připojeny.

Platforma je schematicky znázorněna na obrázku 5.1. Celá platforma se skládá v podstatě z těchto částí:

- Tester – Tvůrce testu zodpovídající také za jeho průběh.
- Aplikace – Řídicí aplikace umožňující tvorbu, editaci a spouštění testu.
- Komunikace – Přenos dat mezi aplikací a zařízeními.
- Zařízení – Externí HW zařízení pro připojení elementů.
- Elementy – Reálné výstupní (aktuátory) a vstupní (senzory) prvky.
- Testovaný – Osoba podstupující test svých kognitivní schopností.

Podrobněji jsou jednotlivé zde uvedené součásti popsané v následujících podkapitolách.



Obrázek 5.1: Základní blokové schéma univerzální testovací platformy.

5.1 Testovaný

Testovaný je osoba podstupující test svých kognitivních schopností. Platforma bude nejčastěji využívána právě k posouzení/detekci, v jakém stavu jsou kognitivní schopnosti testovaného. Před testem musí být test testovanému řádně vysvětlen. Tedy testovaný musí vědět, jak má reagovat na jednotlivé podněty. Za tímto účelem je vhodné, aby test vždy obsahoval nějakou úvodní část pro ověření, zda testovaný zadání testu správně porozuměl.

Po spuštění testu testovaný sleduje podněty a reaguje na ně podle zadání. Podněty jsou testovanému poskytovány prostřednictvím výstupů z platformy. Své reakce testovaný předává testovací platformě pomocí vstupů do platformy jako vstupní data. Po dokončení testu testovaný může získat svůj výsledek. Případně výsledky tester předá k další interpretaci.

5.2 Koncové elementy

Koncové elementy jsou jednotlivé reálné výstupy a vstupy použitelné v testu. Tyto elementy jsou do platformy připojeny přes hardwarová zařízení. Jednotlivé elementy tedy mohou být velmi jednoduché, a tudíž nepotřebují žádnou vlastní logiku. Jejich chování je řízeno zcela skrze zařízení.

Testovanému jsou poskytovány jednotlivé podněty pomocí reálných výstupů, tedy koncových elementů. Jeho reakce jsou následně snímány pomocí obdobně připojených vstupů. Výstupy a vstupy jsou jediná část platformy, s níž testovaný interaguje během testu. Vstupy lze kromě snímání reakcí testovaného použít i ke sběru dalších informací o průběhu testu, například k měření okolní teploty či úrovně osvětlení v testovací místnosti.

Každý výstup i vstup obsahuje definici, o jaký se jedná typ. Tyto typy sdružují více prvků s dostatečně podobným chováním. Platforma bude připravena pro budoucí použití s opravdu různými typy výstupů a vstupů. Nejčastěji používané typy budou samozřejmě předdefinovány, ale pro případného technicky znalého uživatele by nemělo být obtížné později přidat i další typy podle vlastní potřeby. Kromě typu výstupu/vstupu je také definován datový typ

každého prvku.

Základní typy výstupů (aktuátorů) předdefinované v platformě jsou:

- LED – Jednoduchá ovládaná světla. Může se jednat o světla jedno a více barevná, například RGB LED.
- Buzzer – Bzučák slouží k vytváření tónů o dané frekvenci a trvání.
- Číselné displeje – Displeje určené pro zobrazování číslic případně jednoduchých znaků.
- LED Matrix – Maticový displej složený z LED ve čtvercové či obdélníkové síti. Tento typ displeje disponuje často konfigurací 8x8 nebo 16x16 bodů. Mohou být jedno i více barevné. Na těchto typech displejů lze zobrazovat barevné obrázky či písmena a čísla.

Základní typy vstupů (senzorů) předdefinované v platformě jsou:

- Button – Typ pokrývající nejen tlačítka, ale i snímače tlaku, pozice, případně přítomnosti. Celkově tedy všechny typy senzorů poskytující binární či proporcionální informaci o svém stavu.
- Keyboard – Typ určený pro zadávání i rozsáhlejšího textu testovaným.
- NFC čtečka – Typ na principu NFC čtečky využívaný například pro identifikaci přiloženého předmětu/karty/štítku s nalepeným NFC tagem.

V případě výstupních elementů aplikace používá zapisovaná data k nastavování podnětů pro testovaného. Tyto podněty mohou být přímé, jako je rozsvícení světla či přehrání zvuku, ale i nepřímé, jako jsou třeba různé pohyby motorů (například pro okolní úmyslné rušení).

5.3 Zařízení

Zařízení v této platformě představuje externí HW komponentu umožňující připojení elementů (reálných výstupů a vstupů) a předávající data mezi nimi a aplikací. Běžné počítače neobsahují dostatek vstupně-výstupních portů, aby mohl být každý jednotlivý element připojen přímo do počítače s řídicí aplikací. Na každé externí zařízení může být připojeno i více různých elementů současně. Data a signály elementů jsou unifikovány do jednotných datových formátů.

Jelikož k aplikaci může být připojeno více zařízení současně, tak je nutné jednotlivá zařízení vhodně rozlišit. Pro rozlišení zařízení se používají jedinečná sériová čísla. Pomocí těchto sériových čísel je možné rozlišit jednotlivá zařízení a budou rovněž použita v propojování testu a reálných výstupů/vstupů.

Jak již bylo napsáno výše, zařízení bude obsahovat porty pro připojení jednotlivých elementů. Některé porty mohou být odpojitelné. To umožní rozšiřovat možnosti zařízení dle požadavků. Všechny porty v zařízení budou očíslované a toto očíslování bude neměnné. Podobně, jako se budou rozlišovat

zařízení mezi sebou sériovými čísly, tak uvnitř zařízení budou rozlišovány jednotlivé porty svými pořadovými čísly. Konkrétní element bude pro řídicí aplikaci určen kombinací sériového čísla zařízení a pořadového čísla portu.

5.4 Komunikace

Platforma potřebuje pro svou činnost spolehlivou a jednoduchou komunikaci pro přenos pokynů z aplikace do zařízení a odpovědí/stavů v opačném směru. Komunikační rozhraní by mělo být přímo podporované v běžném počítači s řídicí aplikací. Stejně tak by mělo být podporované na úrovni mikroprocesoru obsaženého v zařízení.

Platforma může být použita ve třech zamýšlených scénářích z pohledu délky komunikace (vzdálenosti zařízení od počítače). Pro každé z těchto použití je vhodný poněkud jiný typ komunikace.

- Základní využití předpokládá umístění všech zařízení v blízkosti počítače, například přímo na ploše pracovního stolu. Tato oblast představuje velmi malou vzdálenost od počítače. Takovéto použití bude zřejmě nejčastější. V této konfiguraci jsou vyšší požadavky na měření doby reakce testovaného na podněty.

- Střední varianta předpokládá rozložení testovacích elementů například v rámci jedné místnosti. V tomto scénáři se testovaný může i přemísťovat tak, aby mohl interagovat s jednotlivými elementy umístěnými třeba na stěnách. Vzhledem k možnému přemisťování testovaného budou reakční doby delší než v první variantě. Tedy požadavky na měření reakční doby jsou méně přísné. Zároveň tento scénář využívá komunikaci na poněkud delší vzdálenost.

- Nejvyšší varianta umožňuje rozmístit zařízení s elementy po opravdu rozsáhlé ploše, jako je celá budova či i menší areál. Tohoto lze využít například pro procvičování a testování orientace a paměti. Tomuto scénáři již nevadí vyšší latence v komunikaci. Než údaj o přesném čase je důležitější informace, zda k dané akci od testovaného vůbec došlo. Důležitější parametr v takto rozsáhlé instalaci je možnost komunikovat na relativně velkou vzdálenost a zároveň, zda je potřeba vytvořit novou infrastrukturu či lze využít nějakou stávající.

Ve všech scénářích bude samozřejmě pro jednoduchost použita stejná struktura přenášených dat. U komunikace umožňující rychlý datový přenos bude použit textově orientovaný protokol. U komunikace s velmi omezenou přenosovou kapacitou bude naopak použit binárně orientovaný protokol.

5.5 Aplikace

Aplikace bude hlavní řídicí částí testovací platformy. Tato aplikace umožní vytváření nových testů, jejich úpravu a samozřejmě i spouštění. Dále aplikace

bude nastavovat všechny výstupy a číst vstupy dle jednotlivých zadaných kroků testu. Po spuštění testu aplikace také bude zaznamenávat jeho průběh a veškeré údaje uloží pro pozdější hodnocení.

Před spuštěním testu tester skrze aplikaci propojí výstupně-vstupní kroky testu s reálnými výstupy a vstupy, tedy s elementy na externích zařízeních. Aplikace zkontroluje, zda zařízení odpovídající jednotlivým výstupům a vstupům jsou skutečně připojena. Pokud je vše v pořádku, tak bude možné test spustit.

Součástí aplikace bude i servisní část poskytující informace o všech připojených zařízeních a jejich elementech. Pomocí této servisní části bude možné nastavit požadované stavy na výstupech i mimo test a obdobně vyčítat aktuální stavy vstupů. Toto je vhodné zejména pro servisní a vývojové účely.

5.6 Tester

Osoba zodpovídající za tvorbu jednotlivých testů a jejich spouštění testovaným osobám. Tester celou platformu ovládá pomocí k tomu určené řídicí aplikace. Také může připojovat do počítače potřebná zařízení či přidávat a ubírat koncové prvky (elementy) ze zařízení. Po vykonání testu tester obdrží z aplikace výsledek/záznam testu. Ten může sám interpretovat nebo jej může předat jiné osobě.

Kapitola 6

Výběr použitých technologií

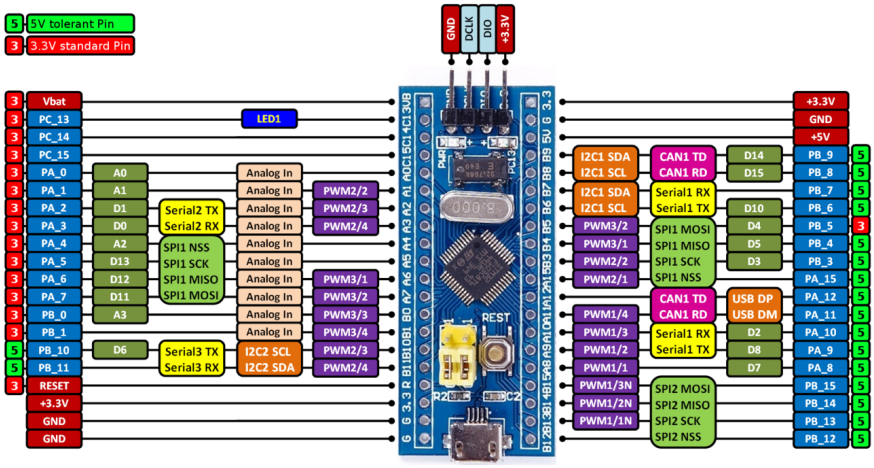
V předchozí kapitole je obecně popsána činnost platformy a úkoly jejích jednotlivých části. Tato kapitola obsahuje postup výběru technologií použitých v navrhované platformě. U vybraných technologií jsou rovněž uvedeny jejich hlavní výhody a nevýhody zejména pro zde zamýšlené použití. Jelikož se předpokládá modifikovatelnost/rozšiřitelnost platformy budoucími uživateli, tak při pozdějších úpravách nebude nutné použít pouze zde vybrané technologie, ale lze použít i v podstatě libovolné jiné. Konkrétní výběr technologií se řídí požadavkem na celkovou jednoduchost platformy, její snadné použití a možnost budoucího rozšíření. Dále se řídí také možnostmi předpokládaných uživatelů, což jsou ve většině případů lékaři, případně jejich asistenti.

6.1 Zařízení a jeho mikrokontroler

Zařízení představuje externí jednotku pro obsluhu připojených elementů (výstupů/vstupů). Elementy mohou být k zařízením připojovány/odpojovány podle předem stanovených pravidel, tedy podle typu elementu a vlastností zařízení. Zařízení dále obstarává přenos dat do/z řídicí aplikace běžící na počítači. Zařízení je vybaveno řídicím mikrokontrolerem. Z na trhu dostupných mikrokontrolerů byly vybrány čipy s architekturou ARM, a to od výrobce ST-Microelectronics, známého též pod zkratkou STM. Výhody architektury ARM jsou následující:

- dostupné i bezplatné vývojové nástroje,
- nízká spotřeba energie při dostatečném výkonu,
- velmi rozšířená architektura procesorů,
- velké množství dokumentace a příkladů.

Konkrétně jsou pro tuto platformu vybrány mikrořadiče z řady STM32F využívající jádro architektury ARM. Tyto řadiče disponují velkým počtem vstupně-výstupních pinů a přímo podporují mnoho komunikačních protokolů jako USB, CAN, I^2C či běžnou sériovou linku. Rovněž poskytují časovače, analogově digitální převodníky a další běžně používané periferie.



Obrázek 6.1: Velmi levná a dostupná vývojová deska BluePill obsahující ARM typu STM32F103.[13]

Podstatnou výhodou mikrokontrolerů z této rodiny je dostupnost velkého počtu vývojových desek obsahujících nejen tyto mikrořadiče, ale často i programátory umístěné přímo na vývojové desce. Jako příklad vývojových desek lze uvést verze Nucleo a Discovery, případně velmi levné destičky nazvané BluePill/BlackPill. Lze tedy zakoupit vývojovou desku a v podstatě s ní ihned pracovat, není potřeba vyvíjet vlastní desku a poté na ni pájet mikrokontroler nebo používat externí programátory. Toto vše nejen velmi zjednodušuje, ale zejména zrychluje celkový vývoj. V zařízeních platformy jsou použity vývojové desky BluePill, viz obrázek 6.1. Tato deska obsahuje mikroprocesor STM32F103C8 s 64 kB FLASH a 20 kB SRAM.[14]

Nicméně použití vývojové desky BluePill a vybraných mikrokontrolerů je jen doporučené. Platforma je navržena v podstatě pro libovolný řadič umožňující ovládat připojené elementy a komunikovat některým z navržených protokolů s řídicí aplikací běžící na počítači.

6.2 Komunikace zařízení s aplikací

Přenos dat mezi aplikací a zařízením bude probíhat v různých režimech. Jak již bylo uvedeno, dle typu použití se rozlišuje blízká komunikace s malou latencí, komunikace na střední vzdálenost a rovněž komunikace na velkou vzdálenost, kde latence již může být větší. Krátkou vzdáleností se rozumí maximálně jednotky metrů (plocha stolu), střední vzdálenost jsou až desítky metrů (místnost) a poslední varianta je pro ještě rozsáhlejší instalace (budova). Pro komunikaci je zřejmě nejdůležitější její snadná implementace v mikrořadiči použitém v zařízení a také dostatečná podpora v počítači s řídicí aplikací.

Vybraný typ mikrokontroleru podporuje následující typy komunikace:

- CAN – Controller Area Network je rychlá sběrnice využívaná často v automobilovém průmyslu.

- U(S)ART – Universal (Synchronous) Asynchronous Receiver/Transmitter je základní rozhraní pro sériovou komunikaci.
- USB – Universal Serial Bus slouží pro komunikaci počítače s běžnými periferiemi a dalšími například malými počítači. Poskytuje různé možnosti dle požadavků připojeného zařízení.
- SPI/I^2C – Lokální sběrnice používané pro komunikaci mezi mikroprocesorem a dalšími periferiemi na velmi krátké vzdálenosti.

Dále existují rozšiřující moduly doplňující mikrořadič o podporu LAN či bezdrátových technologií jako jsou WiFi, Bluetooth, LoRa atd.

Výhoda bezdrátových komunikací je snadné rozmístění a pak i případné přemisťování jednotlivých zařízení, jelikož není potřeba, aby k nim vedly kabely. Bezdrátové komunikace mají také několik nevýhod, a proto nebyly ve zde vytvářené platformě využity. První nevýhoda je omezení možností pohybu bezdrátových zařízení pouze na oblast dosahu signálu. Dále potřeba bateriového napájení. Naopak u drátových komunikací lze zařízení napájet po komunikačních vodičích, případně lze vedle nich vést i napájecí vodiče. Další podstatná nevýhoda je elektromagnetické záření, jež bezdrátová komunikace z podstaty vyzařuje. V nemocnicích i na dalších místech je žádoucí, aby elektromagnetické rušení bylo co nejmenší.

Jako nejčastěji používané drátové technologie jsou tedy k dispozici LAN, USB, USART a CAN. Poslední zmíněná (CAN) je průmyslová sběrnice určená pro rychlou komunikaci zejména mezi několika mikrořadiči. Mezi její výhody patří detekce kolizí a jejich automatické řešení. Dále využívá různých priorit mezi připojenými zařízeními, důležité zprávy mají přednost před zprávami s menší prioritou. Hlavní nevýhoda CANu je absence ovladačů a konektorů/portů v běžných počítačích, a tedy potřeba dodatečných externích převodníků.

6.2.1 USB

Další je komunikace pomocí standardu USB. Jedná se o velmi rozšířený standard plně podporovaný jak běžnými počítači, tak i skutečně velkou řadou mikrokontrolerů. Tento standard využívá pro komunikaci model Master/Slave, kdy se Master označuje jako hostitel a jednotlivá připojená zařízení jsou Slave. Dle typu mikrokontroleru STM32 může být podporováno USB Full-speed i High-speed. Z pohledu kompatibility i s levnějšími mikrokontrolery je uvažována pouze verze Full-speed. V této verzi je maximální délka připojení 5 metrů a po stejném kabelu může být zařízení i napájeno, což je v případě této platformy velká výhoda. Maximální parametry napájení po kabelu USB2.0 je 500 mA při 5 V, tedy maximálně 2,5 W.

Standard USB definuje různé třídy pro připojované zařízení, podle toho, jakým způsobem bude dané zařízení data přijímat či vysílat. Mezi tyto třídy patří například „Media“ pro přenos multi-medií, „Mass storage“ pro datová úložiště, „Communcation Device Class“ (CDC) pro obecný přenos dat a „Human interface device“ (HID) pro komunikaci se zařízeními poskytujícími

rozhraní mezi počítačem a člověkem. Třída HID je vhodná i pro tuto platformu, jelikož se jedná o rozhraní mezi testem a testovaným, tedy počítačem a člověkem. Tyto třídy kromě dat definují i to, jak s nimi má hostitel nakládat, tedy jaký má použít ovladač a případně, jakému programu mají být data předávána. V případě použití obecné/generické HID třídy není potřeba instalovat na počítač žádný dodatečný ovladač a lze využít univerzální předinstalovaný ovladač v operačním systému, což usnadňuje budoucí instalaci aplikace na cílové počítače.

Další výhoda komunikace po USB je v jeho protokolu. Zajišťuje kontrolu přenášených dat a v případě poškození dat, nepřijetí druhou stranou nebo z jiného chybového důvodu je zpráva automaticky odeslána znovu. V aplikaci tedy není potřeba řešit úplnost dat. Jednotlivá USB zařízení mohou mít nastaven název (například dříve zmíněné unikátní sériové číslo) a dle tohoto názvu lze s nimi rovněž komunikovat. Není tedy potřeba řešit žádné složité adresování. Komunikace při použití třídy HID probíhá v rámci tzv. Interrupt transfer s omezenou maximální latencí. Zařízení může hostiteli sdělit, jak často bude generovat data, a hostitel poté s odpovídající frekvencí vyvolává komunikaci. Tento přenos má maximální délku jedné zprávy 64 B (ve verzi Full-Speed).

Každé zařízení připojené k hostiteli po USB potřebuje vlastní USB kabel. Počítače nemají mnoho USB portů, ale lze velmi snadno využít libovolný USB hub dovolující rozšířit jeden USB port do více portů. Celkově lze do USB sítě připojit až nejvýše 127 zařízení.

6.2.2 RS-485

Pro komunikaci na větší vzdálenost (nad 5 m) již není USB vhodné, ale mikrokontroler poskytuje podporu pro běžnou sériovou linku, konkrétně UART. Jedná se o standard, který definuje datový rámec pro zprávy, kdy je vyslán Start bit, poté 5 až 9 datových bitů. Po datech následuje volitelný bit pro kontrolu parity, a poslední je konec rámce obsahující 1 až 2 Stop bity. Komunikační rychlost může být velmi různá. Konkrétní napěťové úrovně a další vlastnosti definují další přídavné standardy pro sériovou komunikaci, například RS-232 nebo RS-485.

První zmíněný standard, RS-232 používá různé úrovně napájení až do rozsahu -15 V až 15 V. Kladné napětí znamená logickou nulu a záporné napětí logickou jedničku. Tato specifikace se primárně používá pro modemy či datové terminály. Obsahuje i speciální signál například pro vyzvánění. Maximální délka kabelu v této verzi je 15 metrů.

Druhý uvedený standard RS-485 definuje pomocí UARTu síť pro připojení až 32 jednotkových zátěží/zařízení. Tato verze se používá v řídicích a podobných systémech. Data jsou přenášena po dvouvodičovém krouceném kabelu a jsou interpretována pomocí rozdílového napětí. Maximální délka vedení je 1200 metrů. Pro využití RS-485 na straně mikrokontroleru stačí doplnit pouze potřebný koncový budič sběrnice. Jelikož se v tomto případě jedná o sběrnici a vysílaná data mohou přijímat všechna připojená zařízení současně, tak je

potřeba vyřešit adresaci k určení cílového zařízení.

6.2.3 LAN

Dalším uvažovaným typem komunikace je LAN, tedy místní síť. Různé způsoby připojení k internetu mají prakticky všechny počítače. Komunikace po místní síti dovoluje přenášet větší objem dat a dle stavu sítě i s větší přenosovou rychlostí než u USB a RS-485. Tato komunikace může probíhat i na opravdu velkou, teoreticky neomezenou vzdálenost (po cestě jsou umístěny různé aktivní prvky). Přenosy však mohou mít různě dlouhé prodlevy.

Dále každé připojené zařízení potřebuje vlastní kabel a pro větší množství připojených zařízení by bylo potřebné použít přepínače/rozbočovače. Ty jsou na rozdíl od hardwaru potřebného pro předchozí komunikace znatelně dražší. V případě rozmístění platformy po větší ploše, jako je areál nemocnice, by bylo možné využít již stávající LAN infrastrukturu pro propojení i jednotlivých zařízení navrhované platformy. Nevýhoda tohoto řešení je sdílení kapacity i s cizí komunikací a případně otázky bezpečnosti. Na straně jednoduchého mikrokontroleru je často nutné připojit modul zajišťující komunikaci po LAN.

Při komunikaci po LAN je potřeba každému zařízení přidělit tzv. IP adresu (unikátní adresu zařízení v síti). Existují dvě možnosti, jak zařízení získá svoji IP adresu. Prvně je možné pro každé zařízení definovat jeho adresu v jeho konfiguraci před připojením. V tomto případě má každé zařízení v platformě určenou jedinečnou IP adresu ještě před prvním připojením do sítě. Druhá možnost spočívá v použití tzv. DHCP serveru. Ten automaticky přiděluje IP adresy zařízením, která si o ni požádají. Využití DHCP serveru je výhodné, protože není potřeba předem definovat pevné adresy v zařízeních. Nevýhoda je, že poté není zřejmé, jaké adresy získala zařízení patřící k platformě. Stejně tak zařízení nemusejí dokázat zjistit, jakou adresou disponuje počítač s řídicí aplikací.

6.2.4 Topologie propojení zařízení

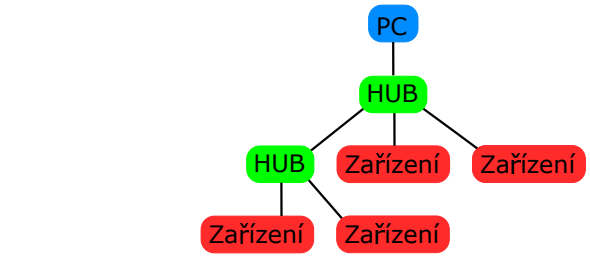
Pro platformu jsou podle vzdálenosti, na jakou je potřeba komunikovat, vybrány tyto typy komunikace:

- USB ve třídě HID pro blízká zařízení, tedy rozmístěná na testovacím stole,
- RS-485 pro zařízení rozmístěná v rozsahu testovací místnosti,
- a LAN pro vzdálenější komunikaci, tedy pro případ rozmístění platformy přes několik místností až celý areál (možno použít i pro menší rozsah, například místnost).

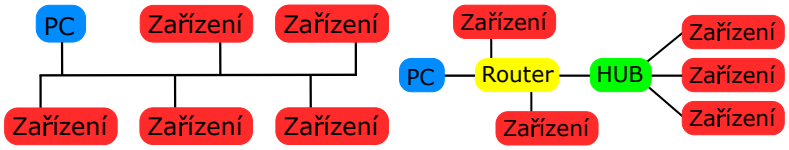
Každý typ využívá jinou topologii propojení. Topologii stromu využívá USB (viz Obr 6.2a). Jako kořen stromu je počítač s aplikací, ten řídí veškerou komunikaci. Data proudí v podstatě přímo mezi počítačem a jednotlivými zařízeními.

Sérová linka v podání standardu RS-485 využívá topologii sběrnice (viz Obr 6.2b). Všechny připojené prvky si jsou na síťové úrovni rovnocenné. Data mohou vznikat na libovolném zařízení a všechny připojené prvky mohou přijímat veškeré zprávy.

Komunikace po LAN má zcela obecnou topologii (viz Obr 6.2c). Konkrétní topologie bude záviset na síti, do níž bude platforma umístěna. Data jsou v LAN přeposílána přes routery či switche/huby.



(a) : USB - topologie stromu.



(b) : RS-485 - topologie sběrnice. **(c) :** LAN - obecná topologie.

Obrázek 6.2: Topologie propojení dle použití různých typů komunikace.

6.3 Aplikace a OS

Řídicí aplikace je určena pro běžné počítače. Měla by být vytvořena pro operační systém nejčastěji používaný cílovými uživateli. Také by měla disponovat snadnou instalací. V dnešní době se používají webové a desktopové aplikace. Výhoda webových aplikací je v nezávislosti na operačním systému. Nevýhoda je obtížná práce s periferiemi počítače při používání těchto webových aplikací. Desktopová aplikace je naopak plně závislá na operačním systému, ale může získat plný přístup k potřebným periferiím. Jelikož je platforma postavena zejména na využívání periferií, tak byl zvolen způsob formou desktopové aplikace. Celosvětově nejrozšířenějším operačním systémem pro osobní počítače je Microsoft Windows. Tento operační systém používají nejčastěji i lékaři a výzkumníci.

Pro snadný vývoj aplikací firma Microsoft poskytuje framework nazvaný .Net Core distribuovaný v podstatě přímo jako součást operačního systému. Pomocí tohoto frameworku lze snadno přistupovat k jednotlivým součástem operačního systému i hardwaru počítače. Framework .Net Core dále obsahuje předpřipravené knihovny pro tvorbu grafických rozhraní. Pro vývoj aplikací pod .Net Core se nejčastěji využívá programovací jazyk C# a ten je použitý i při vývoji řídicí aplikace do této platformy. Další výhoda volby tohoto frameworku je existence vývojového prostředí Visual Studio ve verzi Com-

munity. V této verzi je prostředí bezplatné a umožňuje vývoj ve zvoleném programovacím jazyku C#.[15]

Kapitola 7
Řešení

Tato kapitola je zaměřena na konkrétní řešení jednotlivých částí platformy. V platformě jsou nadefinovány základní stavební prvky tak, aby bylo snadné přidat později další dle potřeby.

7.1 Elementy

V popisu struktury v kapitole 5.2 byly navrženy koncové elementy jako samotné reálné výstupy či vstupy. Později se ukázala nutnost využít v platformě i prvky mající současně výstup i vstup, například podsvícené tlačítko. Dále mohou být v platformě výhodné i prvky mající dokonce více stejných výstupů či vstupů. V realizaci tedy došlo ke změně a elementy označují prvek logicky sdružující i více reálných výstupů či vstupů. Výstup a vstup platformy je vždy označován z pohledu řídicí aplikace.

Elementy jsou hlavními akčními i snímacími prvky celé platformy. Elementy mohou být jednoduché koncové prvky, jako jsou běžná tlačítka, světelné prvky či bzučáky, tak i složitější prvky jako třeba pohyblivé laserové ukazovátko. Jednotlivé elementy dokáže řídicí aplikace rozlišit podle definovaných typů elementů a dále podle datových typů pro jednotlivé výstupy, respektive vstupy.

Každý element z pohledu řídicí aplikace a přenosu dat obsahuje určitý počet výstupů a/nebo vstupů. Výstupy slouží k nastavování stavu elementu, jako je jas LED či tón bzučáku a vytváří tedy podněty. Vstupy naopak slouží jako vstup informací o dění v reálném světě během testu, například stav tlačítka či poloha přepínače, a tedy pracují jako senzory. Každý výstup i vstup má dále specifikováno, jakého je datového typu, a popis, co daný výstup/vstup znamená, tedy například jas či frekvence.

Možnost přidělení více výstupů a vstupů, či jejich kombinací jednomu elementu umožňuje vytvářet současně poněkud komplexnější elementy a zároveň neodporuje použití i velmi jednoduchých elementů, jako jsou samostatné LED či tlačítka. Takto složený element může být třeba tlačítko s integrovanou LED či laserové ukazovátko obsahující v jednom elementu obsluhu zamíření paprsku, respektive pozici cíle, a současně údaj o tom, zda laser svítí či nikoli.

7.1.1 Datové typy

Pro přenos dat z aplikace do zařízení i zpět byly definovány potřebné datové typy. Tyto typy jsou zvoleny zejména z důvodu komprese, tedy minimálního množství přenášených dat. Předem definované datové typy jsou:

- Bool – hodnoty True/False.
- Int/Uint8/16/32/64 – celočíselná hodnota s či bez znaménka reprezentovaná v 8-64 bitech.
- Int/Uint16p2 – desetinné číslo s pevnou desetinnou čárkou se znaménkem či bez něj. Toto číslo je reprezentováno 16 bity a poskytuje 2 pevná desetinná místa.
- Int/Uint32p4 – jako u předchozí varianty, ale na reprezentaci je použito 32 bitů a číslo poskytuje 4 pevná desetinná místa.
- String – pro přenos textových řetězců.
- Bytes – datový typ pro přenos obecných dat, jako pole bytů.

7.1.2 Popisy dat

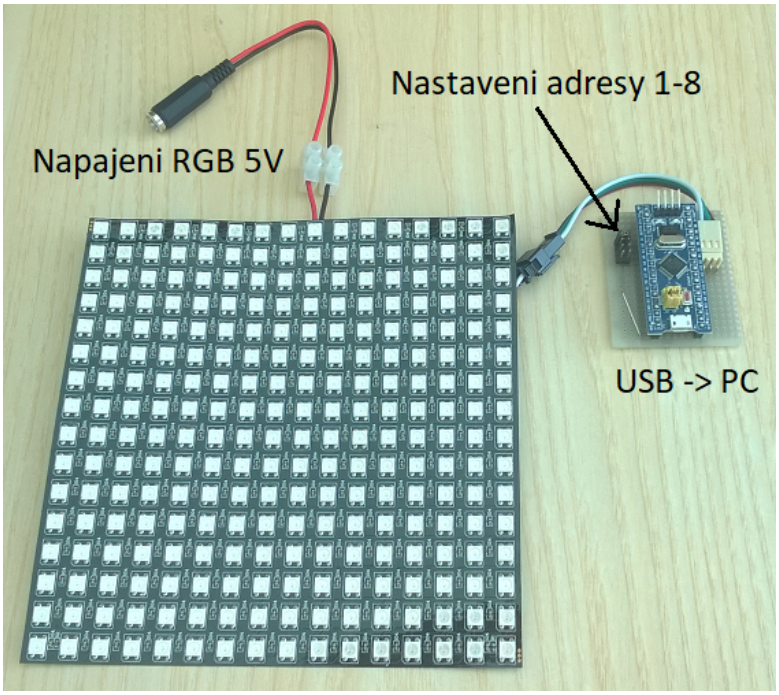
Kombinací popisu a datového typu lze rozlišit různé schopnosti podobných výstupů či vstupů. Definované popisy výstupů a vstupů jsou:

- Jas – úroveň svitu výstupu, případně hodnota jasu vstupního senzoru.
- Frekvence – možnost nastavení frekvence výstupu či snímání periodického děje.
- Trvání – časový údaj.
- Stisk – pro použití s tlačítky.
- Poloha – údaj pro pohyblivé výstupy či snímače polohy.
- Image – data obsahující obrázek.

7.1.3 Typy elementů

Element může slučovat několik spolu souvisejících výstupů i vstupů. Platforma nerozlišuje jen typy výstupů a vstupů, ale i typy celého elementu. Typy elementů vycházejí z typů výstupů a vstupů v kapitole 5.2 a jsou následující:

- Tlačítko – pro tlačítka či spínače.
- LED – pro světelné výstupy.
- Bzučák – pro generování tónů.



Obrázek 7.1: Ukázka zařízení s elementem RGB maticový zobrazovač.

- RGB maticový zobrazovač – pro zobrazování obrázků či symbolů na maticovém displeji.
- NFC čtečka – pro čtení hodnot přiložených NFC tagů.
- Obrázek – pro zobrazení obrázků na displeji.

Uvedené typy elementu jej neomezují pouze na použití odpovídajících výstupů či vstupů, ale měly by označovat hlavní určení daného elementu. Například element typu LED by měl obsahovat převážně svítivé diody, ale může obsahovat i podsvícená tlačítka či jiné podobné typy výstupů. Součástí jednoho elementu mohou být tedy současně výstupy i vstupy. Dle potřeby lze definovat i další typy elementů.

Element nepotřebuje vlastní mikrokontroler. Může se jednat například o samostatné mechanické tlačítko. Veškerou logiku elementu zpracovává zařízení, do něhož je element připojen. Ukázka zařízení s připojitelným elementem RGB maticového zobrazovače je na obrázku 7.1.

7.1.4 Ukázka práce s daty vybraných prvků

Zde je ukázka, jak s daty a datovými typy pracují vybrané výstupy a vstupy.

LED jako nejjednodušší element

Typ LED pokrývá nejjednodušší použití v platformě, ale zároveň dovoluje složitější strukturu dat. Zmíněná základní varianta využívá datový typ Bool, další možný datový typ je Uint8. Možná využití dle datového typu jsou:

- Bool – pouze rozsvítit a zhasnout.
- Uint8 – možnost plynulého nastavení jasu.

Obě varianty by měly být použitelné s víceméně libovolnou LED. Finální možnosti závisí na firmwaru daného zařízení. V případě použití RGB LED lze každou barvu uvažovat jako vlastní výstup nebo lze celé trojbarevné LED přiřadit datový typ Bytes. V takovém případě jsou v datové části přenášeny za sebou tři čísla, každé pro jednu barevnou složku. Ve většině případů se bude jednat o tři hodnoty typu Uint8.

Bzučák

Výstup typu bzučák je určený ke generování různých tónů. V této ukázce požaduje datový typ Bytes. Tento výstup tedy využívá vlastní datovou strukturu, v níž se přenášejí dva číselné údaje:

- Frekvence tónu v Hz jako UInt16.
- Doba trvání tónu v milisekundách jako Uint16.

Bzučák je možné do platformy připojit i s datovým typem Bool. V takovém případě jej lze pouze zapnout, respektive vypnout a není tedy možné nastavit trvání nebo měnit výšku tónu.

Maticový zobrazovač

Maticové zobrazovače dokáží zobrazit jednoduchou grafiku, vlastně obrázek o malém rozlišení. Většina zobrazovačů má rozlišení a vlastně tedy i počet LED 8 x 8 nebo 16 x 16. Každá dioda dokáže svítit 255 různými úrovněmi jasu. V případě použití RGB LED toto platí pro každou barvu, a tedy lze na jednom RGB bodu zobrazit až 16 miliónů různých barev. Lidské oko tolik barev od sebe nerozliší, a proto byl rozsah barev omezen.

Obrazová data lze v současné době přenášet ve dvou formátech. Další formáty mohou být přidány dle potřeby. Data pro maticový zobrazovač jsou typu Bytes. Dále je popsána struktura těchto dat.

- Typ I – Formát typu I je jednobarevný (bod svítí/nesvítí). Tři byty v hlavičce určují RGB barvu společnou pro všechny svítící (aktivní) body. Jelikož se jedná o jednobarevný obraz, tak pak pro každý bod postačuje použití jednoho bitu (svítí/nesvítí).
- Typ II – Formát typu II umožňuje přenos obrázku obsahujícího více barev. Pro jeden přenášený obrázek bude použita paleta 15 (platných) barev a šestnáctá barva indikuje průhledný bod, jenž se nezobrazí.

Jednotlivé zobrazované obrázky lze skládat z více dílčích obrázků. Tímto skládáním lze například ve výsledném obrázku zobrazit více než 15 barev definovaných v paletě, neboť každý dílčí obrázek může mít vlastní paletu

barev. Případně lze již zobrazované obrázky pouze částečně aktualizovat o nová data, tedy změnit pouze jeho část například na jinou požadovanou barvou. K tomu slouží další parametry v hlavičce:

- Bit NoShow určuje, zda se má obrázek zobrazit (false) nebo nikoli (true). Kompletní obrázek se zobrazí na zobrazovači. Pokud obrázek zatím kompletní není, tak se čeká na jeho další části.

U každého obrázku je informace, jak se s ním má naložit v kontextu předchozího zobrazovaného obrázku. Existují tři hodnoty určující, co se s obrázkem má stát:

- Show – Nastaví nově přijatý obrázek k zobrazení. Průhledné pixely se nahradí černou barvou. Původní (zobrazovaný) obrázek je zcela vymazán.
- Add – Přijatý obrázek je přidán k již existujícímu/zobrazovanému obrázku. Z původního obrázku zůstanou pouze ty pixely, které nový obraz překryje průhlednou barvou.
- Remove – Umožní z obrázku odstranit pouze vybrané pixely. Z původního obrázku zůstanou ty pixely, které nový obrázek překryje průhlednou barvou.

Pomocí skládání lze tedy obrázek průběžně upravovat i bez pamatování si jeho aktuálního stavu v řídicí aplikaci.

7.2 Zařízení

Zařízení v této platformě představuje externí jednotku řízenou mikroprocesorem z rodiny ARM. Na toto zařízení může být připojeno několik elementů. Zařízení komunikuje s řídicí aplikací v počítači. Zařízení jsou jednoduchá a data zpracovávají jen minimálně.

Hlavní úkol zařízení je nastavovat elementy s výstupy podle povelů z aplikace a informovat aplikaci o změnách na elementech se vstupy. Z tohoto vyplývá nutnost pro zařízení, umět komunikovat s aplikací a zároveň potřeba ovládat jednotlivé elementy. Preferovaný je velmi rozšířený řadič STM32F103, pro nějž je napsaný vzorový firmware. Tento řadič je osazený na velmi dostupné vývojové desce BluePill. Dále zařízení potřebuje možnost připojení k alespoň jednomu typu komunikace s počítačem, a tedy s řídicí aplikací.

7.2.1 Práce s elementy

Do zařízení jsou připojeny jednotlivé elementy představující v podstatě reálné výstupy a vstupy. Tyto elementy mohou být pevnou součástí zařízení nebo mohou být odpojitelné. Odpojitelné elementy mají na zařízení určený port/konektor. Každý port má definováno, jakého typu může být připojitelný element. Port by kromě samotného připojení elementu měl dokázat detekovat,

zda je element skutečně připojen či nikoli. Na jedno zařízení může být připojeno i více elementů libovolných typů, tyto typy však musejí být v zařízení definovány.

Jednotlivé elementy připojené na zařízení jsou očíslovány. Tímto číslem jsou označeny všechny údaje související s daným elementem. Očíslovány jsou i výstupy a vstupy v rámci každého elementu. Výstupy i vstupy mají v každém elementu společný seznam a u každé položky je uvedeno, zda se jedná o výstup či vstup. Čísla elementu a výstupu/vstupu jsou použita v aplikaci pro určení konkrétního elementu a konkrétního výstupu/vstupu, s nímž aplikace pracuje.

Firmware v zařízení musí obsahovat podporu na obsluhu všech typů připojitelných elementů. Uživatel platformy může samozřejmě využít i vlastní elementy za dodržení podmínky, kdy se jeho nové elementy budou chovat stejně jako některé z již existujících, případně do firmwaru v zařízení bude doplněn kód na jejich správnou obsluhu. Součástí firmwaru je i detekce, v jakém stavu se elementy nacházejí. U odpojitelných elementů má být součástí stavu i informace o tom, zda jsou připojeny či nikoli.

Zařízení má tedy na starost předávání informací mezi elementy a aplikací a kompletní obsluhu elementů a jejich reálných výstupů a vstupů. Když zařízení obdrží z aplikace pokyn k nastavení nové hodnoty na výstupu, tak na daném elementu nastaví požadovaný výstup.

Se vstupy pracuje zařízení podobně. Aplikace si může kdykoli vyžádat stav vybraného vstupu. V takovém případě zařízení zjistí stav vstupu a odešle do aplikace zprávu s aktuálním stavem požadovaného vstupu. Kromě modelu dotaz-odpověď zařízení rovněž automaticky odesílá do aplikace informace o změnách na všech vstupech. Tyto údaje se ukládají do záznamu testu. Je tedy vhodné, aby údaj o změně byl odeslán co nejdříve, a nejen až na dotaz řídicí aplikace.

V případě, kdy elementy nejsou využívány, mohou být blokovány. Blokované elementy nereagují na požadavky na změnu svého stavu a v případě vstupů zařízení neodesílá údaje o jejich změnách. V platformě tedy může být během testu připojeno více elementů, než je k testu potřeba. Jejich blokováním lze omezit množství údajů v záznamu testu nesouvisejících s aktuálním testem.

Po zapnutí zařízení jsou všechny elementy blokovány, dokud nejsou aplikací povoleny. Kromě blokování může aplikace vyžádat i reset elementu, či jeho opětovné povolení. Zařízení tedy potřebuje obslužné povely nejen na nastavování a vyčítání stavu elementů, ale také pro jejich blokování/povolení a případné resetování.

7.2.2 Činnost zařízení

Zařízení slouží jako externí rozšíření aplikace pro nastavování/čtení reálného světa. Po připojení se musí každé zařízení řídicí aplikaci představit pomocí série zpráv a tím sdělit, o jaké zařízení se jedná, a jakými aktuálně připojenými elementy disponuje. Dále zařízení plní požadavky aplikace jako je blokování/povolení elementů a nastavování nových výstupů a odesílá informací o změnách na vstupech.

Během představování zařízení aplikaci se o každém elementu odešle seznam následujících údajů:

- Index elementu v zařízení.
- Jeho stav, tedy zda je připojený, nehlásí chybu a zda je blokovaný či povolený.
- Typ elementu.
- Seznam reálných výstupů a vstupů elementu.

V seznamu výstupů/vstupů jsou pro každou položku následující informace:

- Index položky v daném elementu.
- Směr položky, tedy zda se jedná o výstup či vstup.
- Popis položky.
- Datový typ.

Když proběhne představení, tak lze začít s testem či nastavováním elementů i mimo test. Aby mohlo zařízení jakkoli pracovat s elementem, tak ten musí být povolen. S blokovaným elementem nelze nikterak pracovat.

U aktivních elementů aplikace může nastavit výstupy na požadovanou hodnotu. Rovněž si může vyžádat informaci, v jakém stavu jsou jednotlivé elementy a jaké hodnoty aktuálně poskytují vstupy. Současně zařízení při jakékoli změně aktivovaných vstupů rovněž odesílá automaticky zprávu s novými hodnotami změněných vstupů.

7.2.3 Sériová čísla

Jelikož v platformě může být připojeno více zařízení současně, tak je pro aplikaci důležité je správně rozlišit. Z tohoto důvodu každé zařízení disponuje unikátním sériovým číslem. Pod tímto číslem se zařízení hlásí aplikaci a aplikace pomocí něj rozlišuje dostupná externí zařízení, a tedy i dále k němu připojené elementy.

Jelikož je toto jediný údaj, podle kterého může aplikace a uživatel zařízení rozlišit, tak by mělo být sériové číslo na zařízení viditelně uvedené. Stejné číslo se ukáže v aplikaci při tvorbě a spouštění testu. Dle typu použité komunikace se toto sériové číslo zobrazí již při nalezení zařízení (například pro USB) nebo jej bude potřeba z připojených zařízení dodatečně aplikací zjistit až po jejich detekci (RS-485 a LAN).

Jedinečnost sériového čísla umožňuje přemisťování jednotlivých zařízení mezi instalacemi platformy. Pokud by různá zařízení měla stejná sériová čísla, tak v případě přesunu zařízení mezi instalacemi by došlo k chybě během načítání zařízení. Různí uživatelé mohou vyrábět stále další zařízení, a tak nelze snadno dohlížet na udělování zcela unikátních běžných sériových čísel. Proto se zde využívané sériové číslo skládá ze dvou částí. První je identifikace výrobce a druhá část je jedinečné číslo mezi výrobky tohoto výrobce.

7.3 Komunikace a protokol

V této kapitole jsou popsány realizované druhy komunikace externích zařízení s řídicí aplikací a použitý datový protokol. Pro přenos dat mezi aplikací a zařízeními platforma používá tří druhů komunikace. Na krátkou vzdálenost (metry) je to USB ve třídě HID zařízení. Na střední vzdálenost (do zhruba desítek metrů) používá sběrnici RS-485. A na větší vzdálenost komunikuje po LAN, případně internetu.

7.3.1 Komunikační rozhraní

Podle návrhu byly vybrány tři komunikační rozhraní pro komunikaci na různé vzdálenosti. Jedná se o USB, RS-485 a LAN. Na nejkratší vzdálenost platforma využívá standard USB ve třídě HID s tzv. Interrupt trasferem. Toto nastavení dovoluje vyslat paket o délce až 64 bytů a lze nastavit maximální prodlevu mezi jednotlivými zprávami na 100 ms. Jelikož je omezena velikost paketu, tak se používá binární verze protokolu. Jednotlivá zařízení jsou rozlišena pomocí názvu USB zařízení, jehož součástí je i sériové číslo zařízení.

Na střední vzdálenost platforma využívá sériovou komunikaci dle standardu RS-485. Každé zařízení je k datové lince připojeno potřebným budičem, tedy vysílačem a přijímačem. Avšak v jeden okamžik smí na sběrnici vysílat pouze jedna připojená jednotka. Počítač s aplikací je k této sběrnici připojen pomocí převodníku RS-485 na virtuální sériový port (VCP/COM). Data jsou přenášena rychlostí 115200 baud. Na tomto rozhraní se data přenáší v textové formě. Struktura zprávy je stejná jako v dále popsané binární formě, ale všechny položky jsou uvedeny svým názvem a hodnotou. V počítači jsou všechna připojená zařízení dostupná na jednom sériovém portu. Každá zpráva ještě před hlavičkou obsahuje řetězec sériového čísla požadovaného cílového zařízení.

Poslední varianta je komunikace po LAN. Tato varianta potřebuje úpravy dle konkrétní sítě, do níž bude platforma umístěna. Navržené řešení vyžaduje, aby počítač s aplikací měl pevně nastavenou IP adresu a tato adresa byla zapsána v jednotlivých zařízeních. Tato zařízení získají IP adresu od DHCP serveru. Zařízení se poté připojí k aplikaci na známé IP adrese. Pro přenos bude použit TCP protokol a zprávy budou přenášeny v textové podobě.

7.3.2 Datový protokol

Pro komunikaci aplikace se zařízeními je využit vlastní datový protokol. Tento protokol je primárně vytvořen v binární formě, ale je možné jej využít i v textové formě. V textové podobě se jednotlivé položky oddělují středníkem a v binární formě má každý údaj definovanou délku. Položky s proměnnou délkou, například datové typy String a Bytes, mají vlastní délku uvedenou vždy v prvním bytu těchto dat.

Binární forma je vytvořena tak, aby odpovídala možnostem USB Full-Speed HID zařízení, jelikož tato verze bude nejčastěji používaná v platformě a je

široce rozšířená mezi mikrořadiči. Pakety mají v tomto případě maximální délku 64 B. Delší zpráva musí být rozdělena do více paketů. Každý paket je dále rozdělený na dvě části, hlavičku a data. Hlavička zabírá 4 B a data v jednom paketu mohou využít tedy až 60 B.

V hlavičce je přenášeno šest údajů:

- Command – označení významu dat v paketu.
- Dir – označení směru přenosu.
- Sub – označení jaké části zařízení se daný příkaz týká.
- Status – stav zpracování zprávy.
- Split – zda je přenášená zpráva rozdělena mezi více paketů.
- RemCount – počet zbývajících paketů, aby přenášená zpráva byla kompletní.

A struktura datové části je závislá na daném příkazu (Command) v hlavičce. Příkaz určuje, jak mají být data interpretována, a tedy co se s nimi má stát. Možné hodnoty příkazu jsou:

- Info – Jedná se o informativní zprávu. V případě zprávy z aplikace se jedná o požadavek k vypsání informací o zařízení či elementech. V případě zprávy ze zařízení se jedná o seznam s požadovanými informacemi.
- Command – Speciální typ zprávy, pomocí níž lze z aplikace ovládat zařízení, resetovat jej jako celek případně jednotlivé elementy. Touto zprávou jsou také povolovány/blokovány jednotlivé elementy.
- Write – Tato zpráva slouží k zápisu nových hodnot na výstupy.
- Read – Tato zpráva vyčítá hodnoty ze zařízení či elementů. Respektive vyvolává jejich vyčtení.
- Changed – Typ zprávy informující o změnách na vstupech.
- Data – Zpráva určená pro přenos libovolných dat.
- Debug – Zpráva určená pro ladění a vývoj zařízení.

Směr přenosu určuje, zda daná zpráva pochází z aplikace či ze zařízení. Zařízení by mělo na všechny zprávy od aplikace odpovědět zprávou se stejnou hlavičkou, v níž se změní směr přenosu a případně je upraven status zprávy. Pokud si aplikace danou zprávou vyžádala nějaká data, tak je zařízení připojí v datové části paketu.

Další položka s názvem Sub (Subcomand) upřesňuje, jaké části zařízení se daná zpráva týká. Aktuálně se rozlišují zprávy pro zařízení jako celek a zprávy pro jednotlivé elementy. Dle této položky se liší obsah zprávy, například příkaz typu Info může zjišťovat informace jak o jednotlivých elementech, tak o celém zařízení.

Pokud zařízení odpovídá aplikaci, tak využívá položku Status v hlavičce zprávy. Tato položka zařízení umožňuje sdělit aplikaci, zda pochopilo dotaz a zda dokázalo zpracovat zadaný požadavek. Možné stavy jsou.

- Ok – Vše je v pořádku.
- Unknown command – Zařízení nepodporuje použitý příkaz.
- Unknown element – Zařízení nedisponuje požadovaným elementem.
- Uknown – Zařízení nerozeznalo nějaký údaj ve zprávě.
- Err – Došlo k chybě ve zpracování požadavku.
- Buffer overflow – Zpráva byla příliš dlouhá.

Poslední dva údaje v hlavičce přenášené zprávy slouží k obsluze zpráv rozdělených do více paketů. Pokud se jedná o krátké zprávy v jediném paketu, tak jsou obě položky nulové. V případě rozdělení zprávy na více částí mají všechny dílčí zprávy nastavenou položku Split na hodnotu True. Položka RemCount poté obsahuje, kolik dalších paketů je potřeba přijmout, než bude zpráva kompletní. Poslední zpráva má tuto položku nulovou. V případě rozdělených zpráv je důležité, aby jednotlivé pakety nebyly narušeny jiným přenosem do cílového zařízení. Pokud by během přenosu rozdělené zprávy přišla zpráva s jinou hlavičkou, tak budou data zahozena.

Po hlavičce následují samotná data. Konkrétní struktura datové části je závislá na daném příkazu. Jednotlivé datové položky jsou rozlišeny svým identifikátorem. Takto lze za sebou odeslat i více údajů v jedné zprávě. Konec zprávy je identifikován nulovým identifikátorem dalších dat.

Příkaz Info

Zprávy s příkazem Info slouží ke zjišťování informací o zařízení či elementech. Zpráva má následující strukturu:

- ID – Jedinečný číselný identifikátor elementu v zařízení.
- Status – Určuje aktuální stav elementu, například je-li připojen či vykazuje chybu nebo ideálně je-li v pořádku, a tedy připravený k činnosti.
- ElementType – Definovaný typ elementu.

Za označením každého elementu následuje popis jeho výstupů a vstupů s následujícími informacemi:

- DataID – Číselný identifikátor výstupu/vstupu v rámci daného elementu.
- DataDesc – Popis určující, o jaká data se jedná.
- DataDir – Rozlišení, zda se jedná o výstup či vstup.
- DataType – Datový typ dané položky.

Každý element je tedy následován svými výstupy a vstupy. Element často bude obsahovat pouze jeden výstup či vstup, ale protokol podporuje elementy i s více výstupy či vstupy. Tento výčet je ukončen s položkou DataID rovno 0. Ihned poté následuje ID dalšího elementu. Celá zpráva je ukončena ve chvíli, kdy ID elementu je nulové.

Příkaz Command

Další je příkaz Command. V datové části obsahuje ID elementu a poté některý z následujících příkazů:

- Enable – Povolení daného elementu.
- Disable – Blokování daného elementu.
- Reset – Reset daného elementu.

Pokud Subcommand obsahuje hodnotu Device, tak bude daný příkaz proveden na úrovni celého zařízení. V takovém případě se použije ID 1. Zbylá ID jsou ponechána pro budoucí využití.

Příkazy Write a Read

Další jsou příkazy Write a Read. Oba mají stejnou strukturu. Položka ID představuje identifikátor elementu a poté následují data pro jednotlivé výstupy, respektive vstupy:

- DataID – Číselný identifikátor výstupu/vstupu v rámci daného elementu.
- Data – Samotná data přenášená dle odpovídajícího datového typu.

Opět platí, že 0 v položce DataID ukončuje jeden element a následuje další element. Zpráva je ukončena s ID elementu 0. Příkaz Write je použit pro nastavení výstupů a příkaz Read pro zjištění stavu vstupů.

U příkazu Read ve zprávě z aplikace je vyslán požadavek na zjištění stavů, tedy jsou vynechány položky Data. Aplikace si může samozřejmě zvolit, jaké vstupy chce zjistit a pouze na tyto se zeptá. Tedy pro každý relevantní element jen vypíše jednotlivá zájmová DataID. Pokud chce znát všechny vstupy daného elementu, tak stačí odeslat ID daného elementu a na místě DataID zapsat 0.

Příkaz Changed

Příkaz Changed slouží pro zařízení k odesílání informací o změnách na vstupech. Struktura je stejná jako v případě příkazu Read. Odlišení od příkazu Read je z důvodu, že tyto zprávy nejsou odpovědí na dotaz aplikace, ale zařízení je vysílá zcela bez vyzvání řídicí aplikace.

Příkazy Data a Debug

Příkazy Data a Debug se zatím v protokolu nevyužívají. Jsou ale rezervované pro budoucí použití. Příkaz Data je určen pro přenos obecných dat. Tohoto lze využít například při updatu firmwaru v zařízení nebo pro přenos objemnější konfigurace. Pod objemnou konfigurací si lze představit například obrázky ve velkém rozlišení. Příkaz Debug bude v budoucnu možné používat pro kontrolu/ladění činnosti celé platformy.

7.4 Test

Všechny soubory potřebné pro jeden test jsou uloženy v jednom adresáři. V daném adresáři je přítomno několik XML souborů obsahující test a jeho konfiguraci. Adresář dále může obsahovat další podpůrné soubory, jako jsou například obrázky použité v testu.

Pro úspěšné spuštění testu je potřeba definovat potřebné elementy, tedy použité výstupy (podněty) a vstupy (senzory). Dále je potřeba pro každý element použitý v testu určit, jakému připojenému zařízení a jeho připojenému elementu odpovídá. A také je potřeba popsat samotnou sekvenci výstupů, na něž testovaný reaguje, a seznam vstupů snímajících tyto reakce. Tedy vytvořit posloupnost kroků testu.

7.4.1 Identifikace testu

Test je identifikován dle parametrů popsaných v souboru s příponou TestInfo.xml. Zde je uložen název testu a jeho stručný popis.

Název (Name) – Výstižný krátký název testu.

Popis testu (Desc) – Stručný popis testu pro jeho rozlišení od jiných testů při jeho výběru ze seznamu dostupných testů.

7.4.2 Použité elementy ToDo

Každý test vyžaduje seznam použitých elementů, respektive výstupů a vstupů. V této části se nejedná o reálné výstupy a vstupy ani o elementy jakožto koncové prvky. Jedná se o virtuální elementy použité v testu, které budou později napojeny na reálné výstupy/vstupy. Tento seznam je uložen v souboru s příponou TestElems.xml. V tomto souboru je uložen popis požadovaných typů elementů a jejich datových typů pro vykonání testu.

Název elementu (TestName) - Položka TestName obsahuje jedinečný název elementu v testu. V celém testu se pro práci s daným elementem používá tento název. TestName je generován automaticky aplikací při vytvoření daného elementu v testu. Název se skládá z typu elementu a pořadového čísla v rámci testu.

Směr elementu (Dir) - V položce Dir se uchovává informace, zda se jedná o výstup či vstup. Podporované směry jsou Out pro výstupy poskytující podněty a In pro vstupy, tedy senzory.

Typ elementu (ElmType) - Typ elementu je použit při vytváření a zobrazování testu. Dle typu jsou testerovi zobrazeny odpovídající položky v nastavení.

Datový typ (DataType) - Tato položka určuje, jaká data a případně v jakém formátu daný element data přijímá či poskytuje.

Uživatelský název (UsrName) - Jelikož použití platformou generovaného názvu pro elementy nemusí být pro testera přehledné, tak lze touto položkou nastavit uživatelsky přívětivější název. Pokud tento uživatelský název bude nastaven, tak se bude v testu zobrazovat místo automaticky generovaného názvu. V opačném případě bude zobrazen původní vygenerovaný název. Uživatelský název se bude zobrazovat jak v nastavení testu, tak i ve výsledcích.

7.4.3 Kroky testu

Jak již bylo řečeno, tak se test skládá z posloupnosti kroků. V průběhu testu se tyto kroky postupně vykonávají, a tím se vytváří například výstupní podněty či se očekávají reakce testovaného. Celý test, tedy vytvořený seznam kroků, je uložen v souboru s názvem TestSteps.xml. Kroky mohou být několika typů z různých kategorií a dle těchto kategorií jsou dále popsány.

Kroky řízení testu

Řízení testu (Control).
Jedná se o kontrolu běhu testu. Tato položka může mít tyto hodnoty:

- Start – Zde začíná celý test (inicializace testu/platformy).
- Stop – Na tomto kroku se pozastaví vykonávání testu. Tester poté nechá test pokračovat.
- End – Na tomto kroku končí celý test (ukončení vykonávání testu).

Informační kroky

Slouží pro přehlednost testu zejména pro testera. Tyto kroky jsou i součástí výstupu testu, takže je lze využít i během vyhodnocování.

Oddělovač (Separator).
Oddělovač slouží k oddělení různých částí testu. Jeho součástí může být textový řetězec. Slouží pro lepší vizualizaci struktury testu při jeho vytváření či prohlížení. Separátor představuje vodorovnou linku obsahující ve svém středu zadaný text.

Poznámka (Note).
Poznámka slouží k vložení libovolné textové poznámky do testu. Položka je zobrazena jako prostý text. Například, co se v dané části testu uskutečňuje či co je úkolem testovaného. Při průchodu/vykonávání testu je tato položka zcela ignorována. Parametr je zobrazovaný text.

Kroky pro časování

Tyto kroky slouží k vytvoření časové prodlevy při vykonávání testu.

Konstantní časová prodleva (DelayFixed).
Konstantní časová prodleva vytvoří při každém průchodu ve vykonávání testu pauzu s konstantní délkou. Parametr je počet vteřin jako desetinné číslo.

Náhodná časová prodleva (DelayRand).
Náhodná časová prodleva umožňuje vytvořit v testu prodlevy s proměnným trváním. V každém průchodu tímto krokem bude test pozdržen o dobu náhodně vybranou ze zadaného intervalu. Tento krok má dva parametry, a to minimální a maximální prodlevu. Prodlevy jsou ve vteřinách jako desetinné číslo.

Skoky

Troky tohoto typu slouží pro řízení běhu testu.

Návěstí (Label).
Návěstí slouží k označení konkrétního místa v testu. Na jednotlivá návěstidla je možné během testu přeskakovat a tím v podstatě upravovat průběh testu.

Nepodmíněný skok (Goto).
Nepodmíněný skok má pouze jeden parametr, a to Label. Vykonávání testu dále pokračuje od zvoleného návěstí.

Opakování (While).
Opakování dané části testu. Krok While má dva parametry, první udává počet opakování a druhý určuje místo skoku, pokud není zadaný počet opakování ještě splněn.

Vynechání kroku (SkipFixed a SkipRand).
Pro přeskočení několika následujících kroků jsou v testu dvě možnosti. Obě přeskočí předem zadaný počet následujících kroků. Liší se však tím, kdy k vynechání těchto kroků dojde.

- SkipFixed je nepodmíněný a vždy přeskočí daný počet následujících kroků. Parametr je počet následujících kroků pro přeskočení.
- SkipRand je podmíněný zadanou pravděpodobností. Tento krok má dva parametry, a to počet následujících kroků k přeskočení a pravděpodobnost tohoto přeskoku.

Krok pro práci s proměnnými

Práce s proměnnou (VarWork).
Nejen pro zjednodušení zápisu, ale i pro rozšíření možností testů, mohou být

v testu použity proměnné. V proměnných mohou být uloženy názvy výstupů, vstupů a také další hodnoty nastavované jako parametry v určitých krocích.

Proměnné se od ostatních řetězců v testu odlišují prefixem $. Do proměnných lze ukládat výstupy některých kroků, a pokud proměnná obsahuje odpovídající data, tak může být použita jako libovolný odpovídající parametr.

Proměnné mohou obsahovat buď jednu hodnotu, nebo pole hodnot. V případě pole lze snadno přistupovat k jeho první či poslední hodnotě, případně lze využít indexu. Možnosti použití proměnné nazvané $Var jsou následující:

- $Var.Last – Vrátí poslední hodnotu a ta v seznamu stále zůstane.
- $Var.LastOut – Vrátí poslední hodnotu a ta je ze seznamu odstraněna.
- $Var.First – Vrátí první hodnotu a ta v seznamu stále zůstane.
- $Var.FirstOut – Vrátí první hodnotu a ta je ze seznamu odstraněna.
- $Var[index] – Vrátí hodnotu na zadané pozici a ta v seznamu stále zůstane.

Krok testu určený pro práci s proměnnou/proměnnými obsahuje pouze výraz pro vyhodnocení. Proměnné lze využít ve většině z dále popsaných typů kroků testu.

Krok pro náhodný výběr

Náhodný výběr (Select).
Náhodný výběr je určen pro vytvoření vhodné náhodnosti v průběhu testu. Ze seznamu zadaných možností se vybírají jejich variace, tedy z „n“ možností se vybere „k“ hodnot. Jako možnosti mohou být proměnné, hodnoty, elementy a další. Možnost náhodného výběru lze nastavit volbou dvou vlastností. První volba určuje, zda se mohou opakovat stejné hodnoty. Druhá volba je upřesnění předchozí a upravuje, zda mohou být dvě hodnoty vybrané po sobě stejné. Náhodný výběr v případě jedné hodnoty vrací hodnotu uloženou do proměnné a v případě výběru více hodnot vrací proměnnou obsahující seznam hodnot.

Podmíněný krok

Podmínka (IfThenElse).
Podmínka dovoluje měnit průběh testu podle parametrů či dle předchozího průběhu testu nebo reakcí testovaného. Může být definováno současně více podmínek a podle nastavení musejí být splněny všechny nebo stačí jedna libovolná z nich. Pokud je podmínka splněna, tak test pokračuje dalším krokem za krokem s touto podmínkou, pokud ale podmínka splněna není, tak test skočí na zadané návěstí.

Kroky pro práci s výstupy a vstupy

Tyto kroky umožňují nastavit hodnoty na výstupech. Také umožňují zjistit, v jakém stavu jsou jednotlivé vstupy.

Výstup (OutOne).
Nastaví vybraný výstup na požadovanou hodnotu. Nová hodnota musí odpovídat datovému typu vybraného výstupního elementu. Aplikace napoví, jaká data element požaduje. Parametry jsou název výstupu a nová výstupní hodnota. Také je možné u některých elementů nastavit všechny jejich výstupy jedním krokem OutOne.

Vstup (InOne).
Tento krok zjistí aktuální stav jednoho vstupu. Hodnota se objeví v záznamu testu a může být i uložena do proměnné. Parametr je název vstupu a případně název proměnné pro uložení hodnoty.

Kontrola vstupu (InMany).
Tento krok může kontrolovat více vstupů současně. U každého vstupu je definován očekávaný stav. Tento krok čeká, dokud se zadané vstupy nedostanou do požadovaných stavů. V nastavení tohoto typu kroku lze zvolit, zda mají být splněny všechny podmínky nebo stačí splnění libovolné z podmínek. Tento krok má parametry timeout pro nastavení maximální doby čekání a poté pro každý testovaný vstup má zadán požadovaný/očekávaný stav.

Jednotlivé varianty chování jsou:

- WaitForAll – Krok čeká na splnění všech vypsaných podmínek (všech zadaných stavů vstupů).
- WaitforAny – Krok čeká na první splněnou podmínku (libovolný ze zadaných vstupů).

Krok pro kontrolu vstupu je mimo jiné vhodný, pokud má pokračování testu počkat na reakci testovaného. Také umožňuje zpracovat stav, kdy testovaný může reagovat více tlačítky.

Vstup z klávesnice (InKeyboard).
Klávesnice je speciální vstup nevytvořený jako externí zařízení. Pomocí tohoto kroku lze snímat stisky kláves z klávesnice připojené přímo k počítači s testovací aplikací. Jsou k dispozici tři různé způsoby čtení vstupu z klávesnice:

- Jeden znak – Test získá informaci o první stisknuté klávese.
- Zadaný počet znaků – Test získá zadaný počet stisknutých kláves.
- Libovolný text – Poslední varianta je určená pro vkládání obecného textu. Vstup se ukončí stiskem klávesy Enter.

Klávesy klávesnice mohou rovněž simulovat běžná externí tlačítka, jako by byla připojena pomocí reálného externího zařízení. V takovém případě jsou čtena kroky InOne či InMany.

Kroky pro záznam

LogText.
Všechny kroky se sice ukládají do záznamu testu, ale podoba jejich záznamu je pevně daná. Pro lepší a upravitelné záznamy vznikl krok LogText. Tento krok v záznamu vytvoří vlastní položku, do níž lze vložit vlastní formátovaný popis včetně stavu proměnných či výstupů a vstupů.

LogEvent.
Aby mohly být do záznamu testu ukládány i (v podstatě asynchronní) změny vstupů nezávisle na průběhu/stavu testu, tak je přítomen krok LogEvent. Jako parametr požaduje název vstupu a formátovaný řetězec, pomocí něhož se do výstupu testu uloží aktuální hodnota tohoto vstupu (jako text) při jeho každé změně. Některé vstupy tedy mohou být zpracovány pouze testem, například čekání na reakci testovaného a některé vstupy mohou být pouze zaznamenávány do výstupu, protože nemají vliv na průběh testu (jsou pouze detekovány při hodnocení testu).

7.4.4 Propojení testu s reálným světem

Poslední konfigurační soubor TestHW.xml popisuje finální propojení elementů definovaných v testu s reálnými elementy připojenými na externí zařízení. Pro každý element v testu je určeno, jakému zařízení a koncovému prvku odpovídá. Pokud je element v testu definovaný jako samostatný výstup/vstup, a ne celý element, tak je rovněž uvedeno, jakému konkrétnímu reálnému výstupu/vstupu na daném koncovém prvku odpovídá.

7.5 Aplikace

Aplikace je napsána v jazyce C# pro operační systém Windows 10. Pro běh využívá framework .NET Core a pro správnou činnost je potřeba na počítač nainstalovat .NET 7.0 Desktop Runtime.

Aplikace slouží testerovi k řízení celé platformy. Musí tedy umožnit vytvořit, upravit a spustit test, komunikovat s připojenými zařízeními a jejich elementy a aplikace také zobrazuje výsledky testu. Pro přehlednost je vytvořená aplikace rozdělena na dvě části, hardwarovou část (HW) starající se o komunikaci aplikace se zařízeními a testovou část (GUI) pro správu testů. Propojení testových elementů s reálnými výstupy/vstupy probíhá na úrovni hardwarové aplikace poskytující vrstvu abstrakce pro test. Testová aplikace pracuje pouze s testovými elementy označenými vygenerovanými názvy.

7.5.1 HW část aplikace

Tato aplikace představuje rozhraní mezi testem a zařízeními s reálnými výstupy a vstupy. Umožňuje vyhledat všechna zařízení připojená přes podporované komunikace. Po nalezení zařízení umožňuje HW část aplikace číst údaje ze zařízení a také do něj zapisovat nové hodnoty a stavy výstupů.

Pro každé připojené zařízení jsou zjištěny jeho elementy a výstupy/vstupy. Pro každý výstup a vstup se vytvoří datová položka s odpovídajícím datovým typem. Po přijetí zprávy ze zařízení s novými stavy vstupů se tyto nové stavy uloží do takto vytvořených datových položek. Obdobně zápis nových hodnot do těchto datových položek z testu způsobí odeslání zprávy do zařízení s pokyny ke změně stavu výstupů.

Testová část aplikace má přístup pouze k datovým položkám v HW části aplikace. Se zařízeními tedy nemůže komunikovat přímo. Všechny požadavky na změny stavů výstupů testovací aplikace zapisuje do datových položek HW části aplikace, která následně sestaví zprávu a odešle ji do odpovídajícího zařízení.

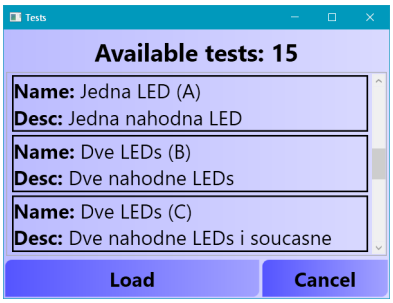
Obrázek 7.2: Servisní dialog HW části aplikace.

Součástí HW části aplikace je i servisní dialog zobrazující aktuálně nalezená zařízení a jejich připojené elementy. Tento dialog je zobrazen na obrázku 7.2. V případě problémů se zařízením či během vývoje nového zařízení lze pomocí tohoto dialogu dále nastavit požadované stavy na výstupních elementech a rovněž zjistit aktuální stav vstupních elementů.

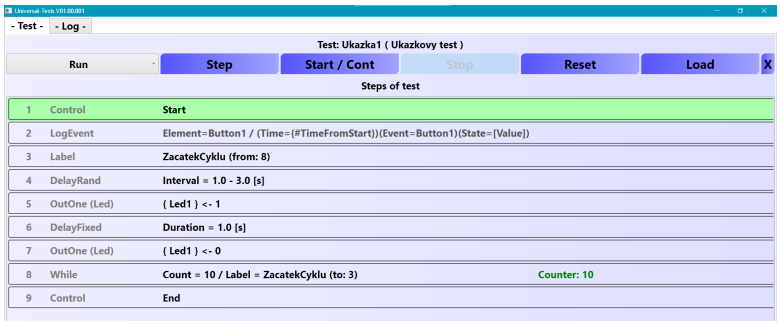
7.5.2 Aplikace pro správu testu

V této aplikaci probíhá tvorba testů, jejich úpravy, spouštění a také se zde zobrazí záznam testu. Po svém spuštění aplikace neobsahuje žádný test. Uživatel tedy musí načíst z disku dříve vytvořené testy nebo založit nový test. Požadovaný test vybírá z nabídky dle názvu testu a jeho popisu (viz obrázek 7.3). Po načtení testu jej lze rovnou spustit a případně upravit. Načtený ukázkový test je na obrázku 7.4. Spuštěný test ukládá jednotlivé vykonané kroky do záznamu testu. Při práci s výstupy/vstupy aplikace komunikuje s HW aplikací a ta následně komunikuje se zařízeními. Záznam testu je zobrazený na obrázku 7.5.

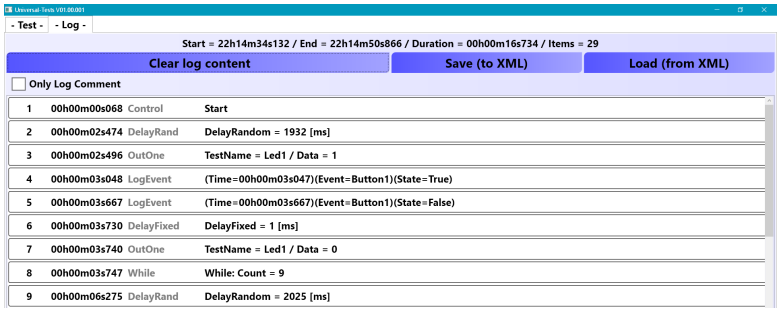
Prostředí editace testu je zobrazeno na obrázku 7.6. Upravovat lze test načtený z disku. Případně lze vytvořit zcela nový test. Při úpravách lze přidávat libovolné kroky testu, u každého typu kroku je i počítadlo použití daného typu kroku daného typu v celém testu. Po vložení kroku do testu lze nastavit jeho parametry a případně jej přesunout na jiné místo v rámci testu. Levá část obsahuje dostupné kroky testu a v pravé části je samotný upravovaný test.



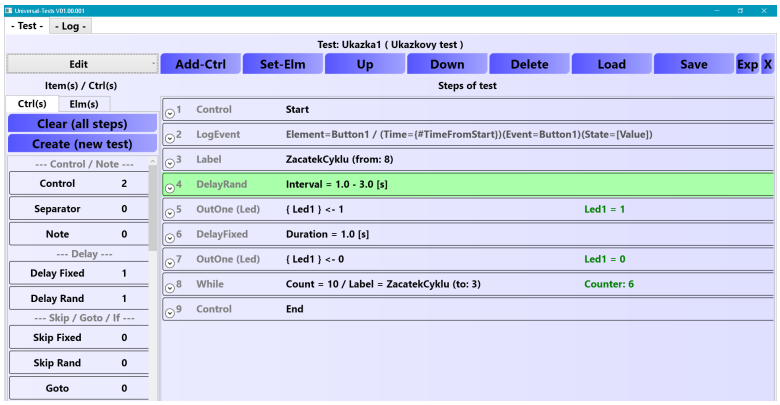
Obrázek 7.3: Dialogu pro výběr testu ze seznamu dostupných testů.



Obrázek 7.4: Aplikace s načteným testem.



Obrázek 7.5: Ukázka záznamu testu v aplikaci.



Obrázek 7.6: Aplikace v prostředí pro editaci testu.

Kapitola 8
Příklady testu

V této kapitole jsou popsány ukázkové testy představující možnosti platformy. Hlavní testované kognitivní schopnosti v těchto testech jsou pozornost, paměť a rozpoznávání symbolů. Prvně jsou popsány tři varianty jednoho testu na pozornost. Jednotlivé varianty vždy přidávají další komplexitu do testu. Další ukázkový test je zaměřen na paměť. Tyto uvedené testy lze všechny spustit na stejném zařízení se stejnými elementy, pokud obsahuje alespoň tři výstupy (LED) a alespoň tři tlačítka (Button). Tyto testy lze výměnou výstupů upravit i na jiný typ podnětů, například různé symboly pomocí displejů či rozlišování zvuků a tónů pomocí generátorů zvuků.

Na konci kapitoly je ukázka testu na rozpoznávání symbolů. Tento test ukazuje možnost platformy pracovat nejen s externím HW, ale také pouze s testovacím počítačem.

8.1 Testy na pozornost

První test vyžaduje jednu LED a jedno tlačítko, tedy po jednom výstupu a vstupu. Zbylé dvě varianty vyžadují dvě LED a jedno tlačítko. Úkol testovaného je ve všech třech testech stejný, a to stisknout tlačítko ve chvíli, kdy svítí jen a pouze Led1. Pokud svítí jiná LED či svítí obě dvě LED, tak se tlačítko nemá stisknout.

Test na pozornost I

Kroky první varianty testu jsou na obrázku 8.1. Tento obrázek obsahuje celý průběh testu. Jednotlivé kroky jsou pro přehlednost očíslovány. Krok Control s parametrem Start značí začátek testu. Krok LogEvent nastavuje parametry záznamu, podle nichž se budou ukládat reakce testovaného. Krok Separator odděluje začátek testu a zároveň uvádí jeho popis. Samotný výkonný test, tedy nastavování výstupů (zde Led1), je mezi kroky 5 a 11.

Jedná se o cyklus probíhající desetkrát. V každém průchodu se Led1 rozsvítí s pravděpodobností 50 %. Po náhodně dlouhé pauze (krok 8) LED zhasne (krok 9). Další kolo následuje po pauze pevné délky 1 vteřiny (krok 10). Návěstí Cyklus-Start (krok 5) označuje začátek každého kola testu. Stav

Steps of test

1	Control	Start	
2	LogEvent	Element=Button1 / (Time={#TimeFromStart})(Event=Button1)(State=[Value])	
3	Separator	--- Demo Test A - Rozsveci se pouze LED 1 ---	
4	DelayFixed	Duration = 2.0 [s]	
5	Label	Cyklus-Start (from: 11)	
6	SkipRand	Items = 1 / Prob = 50%	
7	OutOne (Led)	{ Led1 } <- 1	
8	DelayRand	Interval = 2.0 - 4.0 [s]	
9	OutOne (Led)	{ Led1 } <- 0	
10	DelayFixed	Duration = 1.0 [s]	
11	While	Count = 10 / Label = Cyklus-Start (to: 5)	Counter: 10
12	Control	End	

Obrázek 8.1: Kroky testu na pozornost I.

(stisk) tlačítka Button1 je automaticky (na pozadí) ukládán do záznamu testu. Krok Control s parametrem End ukončuje celý test.

Test na pozornost II

1	Control	Start
2	LogEvent	Element=Button1 / (Time={#TimeFromStart})(Event=Button1)(State=[Value])
3	Separator	--- Demo Test B - Dve LED, sviti prave jedna z nich ---
4	DelayFixed	Duration = 2.0 [s]

Obrázek 8.2: Kroky testu na pozornost II - úvodní část testu.

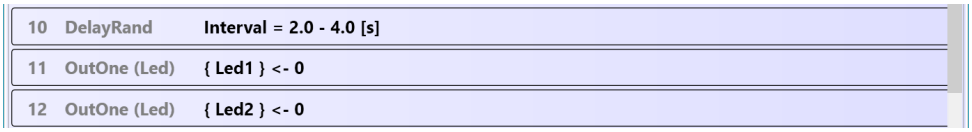
Druhý test má navíc druhou LED (Led2). V každém okamžiku svítí nejvýše jedna z nich. Úkol testovaného je stále stejný, pouze přibyl rušivý element. Na obrázku 8.2 je úvodní sekvence testu. Tento úvod je stejný jako v předchozí variantě testu. Jediný rozdíl je v popisu Separatoru (krok 3).

5	Label	Cyklus-Start (from: 14)
6	SkipRand	Items = 2 / Prob = 50%
7	OutOne (Led)	{ Led1 } <- 1
8	SkipFixed	Items = 1
9	OutOne (Led)	{ Led2 } <- 1

Obrázek 8.3: Kroky testu na pozornost II - rozsvěcení LED.

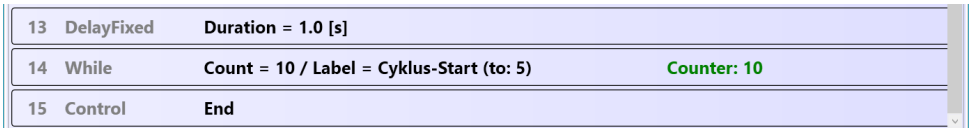
Další část testu je na obrázku 8.3. Návěstí Cyklus-Start (krok 5) značí začátek kola testu. Následující 4 kroky vytvářejí náhodné rozhodnutí, která z LED se rozsvítí. S pravděpodobností 50 % (krok 6) se buď přeskočí kroky 7 a 8 a rozsvítí se Led2 (krok 9), nebo bude vykonávání testu pokračovat

následujícím krokem a rozsvítí se Led1 (krok 7). Následně se v kroku 8 přeskočí rozsvěcení Led2 (krok 9). Na konci této části svítí právě jedna LED a v obou případech je další vyhodnocovaný krok číslo 10.



Obrázek 8.4: Kroky testu na pozornost II - zhasnutí LED.

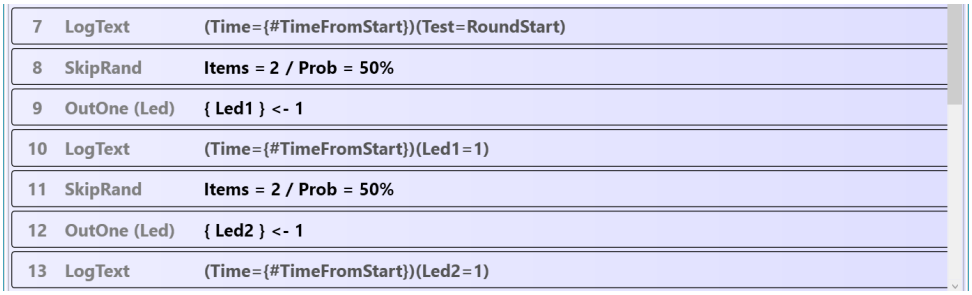
Následující část testu na obrázku 8.4 vytváří interval v rozmezí 2 až 4 vteřin (krok 10), kdy LED svítí. Vložení náhodného intervalu místo pevné pauzy nevytvoří konstantní rytmus testu. Kroky 11 a 12 zhasnou obě LED, jelikož test si pro svou jednoduchost nepamatuje, která z nich je rozsvícena.



Obrázek 8.5: Kroky testu na pozornost II - konec testu.

Poslední část tohoto testu je na obrázku 8.5. Po zhasnutí LED v předchozí části je pevná pauza 1 vteřina, kdy nesvítí žádná LED (krok 13). Krok 14 uzavírá kolo testu. Tento krok přesune vykonávání testu na krok 5, tedy na návěstí Cyklus-Start, pokud neproběhlo všech deset kol testu. Po desátém průchodu se již test nevrátí na návěstí Cyklus-Start, ale pokračuje dalším krokem. V tomto testu je další krok Control s parametrem End (krok 15), který celý test ukončuje.

Test na pozornost III



Obrázek 8.6: Kroky testu na pozornost III - hlavní část testu.

Ve třetí variantě tohoto testu mohou nastat situace z předchozích variant a navíc mohou obě LED svítit současně. Začátek a konec testu jsou v podstatě stejné jako v případě předchozích testů. Hlavní část třetího testu je na obrázku 8.6. První krok v ukázce je typu LogText. Tento krok umožní do záznamu testu vložit libovolný řetězec, případně i se stavovými informacemi. V tomto

případě je do záznamu uložen čas od začátku testu s indikací začátku kola testu.

Náhodný skok (krok 8) opět buď pokračuje v dalším vykonávání, nebo přeskočí dva následující kroky. Tato přeskakovaná část je rozsvícení Led1 (krok 9) a zápis času této události do záznamu (krok 10). Bez ohledu na stav skoku v kroku 8 následuje stejná sekvence kroků pro Led2 (kroky 11-13). Každá LED se tedy rozsvěcí s pravděpodobností 50 % a jsou na sobě navzájem nezávislé. Po této ukázce následuje stejné zakončení jako v případě druhého testu. Tedy pauza, zhasnutí obou LED, opakování dalšího kola a na konec ukončení celého testu.

8.2 Test na paměť

V ukázkovém testu na paměť se rozsvítí sekvenci tří LED a testovaný má následně ve stejném pořadí stisknout odpovídající tři tlačítka. Tento test ihned kontroluje správnost odpovědí. Pokud testovaný udělá chybu, pokus je označen za chybný a zobrazí se nová sekvence, tedy přejde se na další kolo testu.

Tento test kromě možnosti testovat paměť ukazuje také pokročilejší možnosti tvorby testu poskytované platformou. Začátek testu je na obrázku 8.7 kromě začátku podobného jako v testech na pozornost jsou navíc zhasnuty všechny používané LED (kroky 3-5).

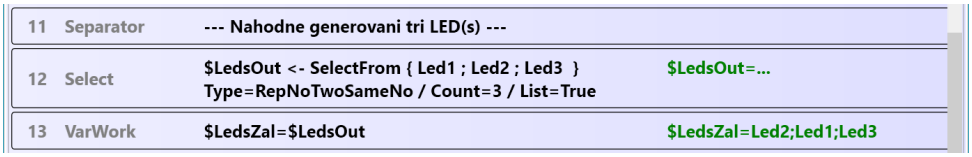
Steps of test

1	Control	Start
2	Separator	--- Demo Test D - Test na paměť (opakování posloupnosti, ktera je rovnou kontrolovana) ·
3	OutOne (Led)	{ Led1 } <- 0
4	OutOne (Led)	{ Led2 } <- 0
5	OutOne (Led)	{ Led3 } <- 0
6	Note	Test na zapamatování si posloupnosti barevných světel
7	DelayFixed	Duration = 1.0 [s]
8	LogText	(Time={#TimeFromStart})(Test=Start)
9	Label	Cyklus-Start (from: 40)
10	LogText	(Time={#TimeFromStart})(Test=RoundStart)

Obrázek 8.7: Kroky testu na paměť - úvodní část testu.

Další část testu se zabývá generováním náhodné posloupnosti LED (viz Obr 8.8). Posloupnost LED je potřeba vytvořit náhodně, k tomu slouží krok Select (krok 12). Do proměnné $LedsOut se uloží všechny tři LED, ale pokaždé v jiném, náhodném pořadí pro příslušné kolo testu. Pro další práci s touto sekvencí je toto pořadí zkopírováno i do proměnné $LedZal pro zálohu (krok 13).

Celá část testu starající se o rozsvícení sekvence LED je na obrázku 8.9. Jedná se o cyklus se třemi opakováními mezi kroky 15 a 22. V každém průběhu je do záznamu uloženo, jaká LED se rozsvítí (krok 16). LED určená pomocí



11	Separator	--- Nahodne generovani tri LED(s) ---	
12	Select	$LedsOut <- SelectFrom { Led1 ; Led2 ; Led3 } Type=RepNoTwoSameNo / Count=3 / List=True	$LedsOut=...
13	VarWork	$LedsZal=$LedsOut	$LedsZal=Led2;Led1;Led3

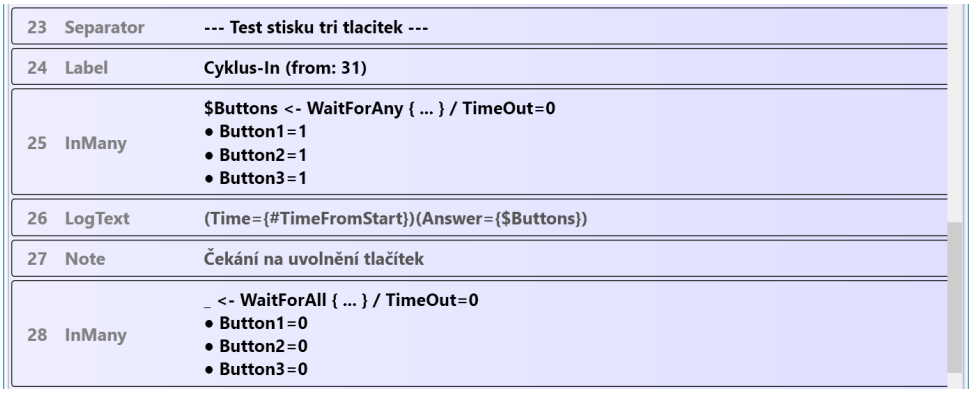
Obrázek 8.8: Kroky testu na paměť - generování náhodné posloupnosti.

první položky v proměnné $LedOut se rozsvítí (krok 18) a po pauze (krok 19) opět zhasne a je z proměnné odstraněna (krok 20). Po krátké pauze (krok 21) test pokračuje (krok 22) dalším rozsvěcením LED z náhodně generované posloupnosti. Pokud již byly rozsvíceny všechny LED z posloupnosti, tak se pokračuje dalším krokem.

14	Separator	--- Postupne rozsviceni tri LED(s) ---	
15	Label	Cyklus-Out (from: 22)	
16	LogText	(Time={#TimeFromStart})(Input={$LedsOut.First})	
17	DelayFixed	Duration = 1.0 [s]	
18	OutOne (None)	{ $LedsOut.First } <- 1	
19	DelayFixed	Duration = 1.0 [s]	
20	OutOne (None)	{ $LedsOut.FirstOut } <- 0	
21	DelayFixed	Duration = 1.0 [s]	
22	While	Count = 3 / Label = Cyklus-Out (to: 15)	Counter: 3

Obrázek 8.9: Kroky testu na paměť - rozsvícení LED.

Testovanému byla zobrazena sekvence tří LED a nyní má stisknout tři tlačítka ve stejné posloupnosti. Na obrázku 8.10 je začátek cyklu čtení tlačítek. Test čeká na stisk libovolného tlačítka (krok 25), stisknutá tlačítka se jako seznam uloží do proměnné $Buttons. Tento stisk se zapíše do záznamu (krok 26) a poté test čeká, dokud nebudou uvolněna všechna tlačítka (krok 28).

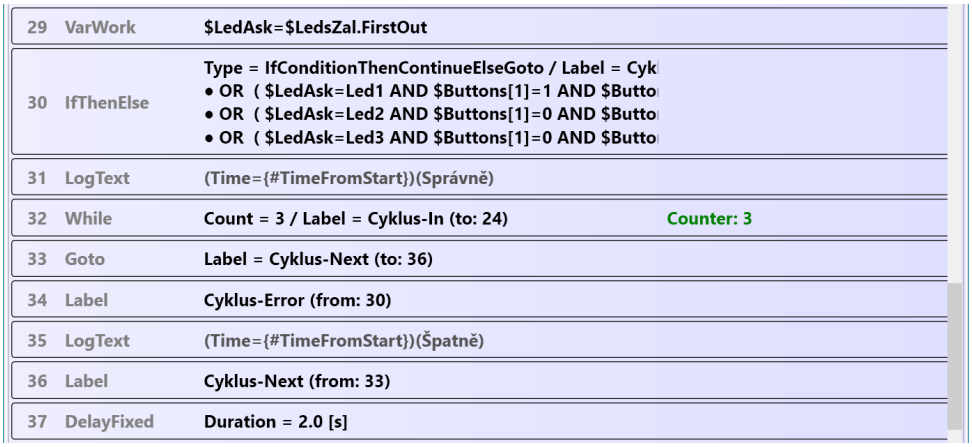


23	Separator	--- Test stisku tri tlacitek ---
24	Label	Cyklus-In (from: 31)
25	InMany	$Buttons <- WaitForAny { ... } / TimeOut=0 • Button1=1 • Button2=1 • Button3=1
26	LogText	(Time={#TimeFromStart})(Answer={$Buttons})
27	Note	Čekání na uvolnění tlačítek
28	InMany	_ <- WaitForAll { ... } / TimeOut=0 • Button1=0 • Button2=0 • Button3=0

Obrázek 8.10: Kroky testu na paměť - čtení tlačítek.

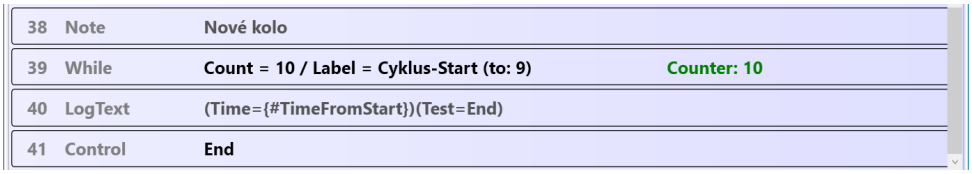
Každý stisk testovaného je ihned kontrolován, zda je správný či špatný. Tato kontrola je na obrázku 8.11. Prvně se ze záložní proměnné získá odpovídající LED (krok 29). Následuje kontrola, zda tato LED odpovídá právě stisknutému

tlačítku (krok 30). Při této kontrole se využije proměnná $Butons. Pokud bylo stisknuto správné tlačítko, tak test zapíše do záznamu, že stisk byl správný (krok 31), a pokračuje čekáním na stisk dalšího tlačítka (krok 32). Chybný stisk přesune vykonávání testu z kroku 30 na návěstí Cyklus-Error (krok 34). Zde se do záznamu zapíše chybný pokus (krok 35) a test pokračuje další sekvencí LED (krok 36).



Obrázek 8.11: Kroky testu na paměť - kontrola reakce.

Jednotlivá kola jsou od sebe oddělena pauzou (krok 37). Na obrázku 8.12 je skok na další kolo testu (krok 39) a ukončení testu (kroky 40 a 41).



Obrázek 8.12: Kroky testu na paměť - konec testu.

Záznam testu je stejný jako u předchozích testů. Navíc obsahuje položky LogText, jež lze zobrazit samostatně. Jelikož v testu všechny důležité části byly uloženy do záznamu pomocí kroku LogText, tak je v záznamu možné zobrazit seznam správných a špatných reakcí testovaného.

8.3 Test na rozpoznávání symbolů

Platforma je primárně určena pro použití s externím hardwarem, ale je možné využít i monitor testovacího počítače a jeho klávesnici. Tento test je pro ukázku zobrazen ve variantě na testovacím počítači, ale s připojeným elementem typu displej lze test spustit i na externím hardwaru.

Během testu se testovanému postupně budou zobrazovat obrázky hodin s nabídkami časů. Testovaný má rozhodnout, který z uvedených časů je opravdu na hodinách zobrazen, a podle toho stisknout odpovídající tlačítko.

Steps of test

1	Control	Start
2	Separator	--- Rozpoznání času na hodinách ---
3	DelayFixed	Duration = 1.0 [s]
4	Dialog (Define)	Action = Define • Picture / Pos: 60%;40% / Size: 60%;60%
5	Dialog (Show)	Action = Show • Picture
6	LogText	(Time={#TimeFromStart})(Test=Start)
7	Label	Cyklus-Start (from: 25)

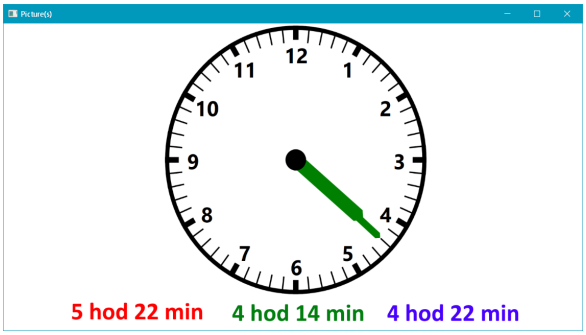
Obrázek 8.13: Kroky testu na rozpoznávání symbolů - úvodní část.

Na obrázku 8.13 je inicializace testu. Oproti předchozím testům je nová inicializace dialogu na počítači, nastavuje se jeho velikost a umístění na monitoru (kroky 4 a 5). Zbytek začátku testu je běžný.

8	LogText	(Time={#TimeFromStart})(Test=RoundStart)
9	Note	Nahodny výběr jednoho obrázku
10	Select	$PicOut <- SelectFrom { Hodiny:* } $PicOut=... Type=RepNoTwoSameNo / Count=0 / List=False
11	OutOne (Picture	{ Picture } <- Hodiny:$PicOut
12	LogText	(Time={#TimeFromStart})(Input={$PicOut[1]};{$PicOut[2]};{$PicOut[3]})(Correct={$PicOu

Obrázek 8.14: Kroky testu na rozpoznávání symbolů - zobrazení obrázku.

Podobně, jako se v minulých testech náhodně vybíraly LED, i v tomto testu je náhodný výběr (viz obrázek 8.14). Náhodně se vybírá obrázek hodin k zobrazení ze zadaného adresáře (krok 10). Vybraný obrázek se zobrazí na displeji (krok 11) a vybraný obrázek se uloží rovněž do záznamu (krok 12). Ukázka zobrazených hodin je na obrázku 8.15.



Obrázek 8.15: Obrázek hodin určený k rozpoznávání času. Na obrázku jsou tři možné odpovědi.

Další část testu je zjištění reakce testovaného a její porovnání se správnou hodnotou. Toto vyhodnocení je na obrázku 8.16. Tato část je velmi podobná ukázce na obrázku 8.11. Hlavní rozdíl je ve zkoumaném vstupu, v tomto

případě se jedná o vstup z klávesnice připojené k počítači (krok 13). Správná odpověď je v tomto případě součástí názvu vybraného obrázku, z něhož je získána v kroku 15. V kroku 16 jsou obě hodnoty porovnávány.

13	InKeyboard	$KeyIn <- Char { One Char } / TimeOut = 0
14	LogText	(Time={#TimeFromStart})(Answer={$KeyIn})
15	VarWork	$KeyWait=$PicOut[4]
16	IfThenElse	Type = IfConditionThenContinueElseGoto / Label = Cyk • OR ($KeyWait=$KeyIn)
17	LogText	(Time={#TimeFromStart})(Správně)
18	Goto	Label = Cyklus-Next (to: 21)
19	Label	Cyklus-Error (from: 16)
20	LogText	(Time={#TimeFromStart})(Špatně)
21	Label	Cyklus-Next (from: 18)

Obrázek 8.16: Kroky testu na rozpoznávání symbolů - čtení klávesnice.

Poslední část testu je opět jeho ukončení (viz Obr 8.17). Nový je krok Dialog s parametrem CloseAll (krok 27). Tento krok slouží ke správnému zavření všech dialogů testu.

22	Dialog (ClearAll)	Action = ClearAll	
23	DelayFixed	Duration = 2.0 [s]	
24	Note	Další obrázek	
25	While	Count = 10 / Label = Cyklus-Start (to: 7)	Counter: 10
26	LogText	(Time={#TimeFromStart})(Test=End)	
27	Dialog (CloseAll	Action = CloseAll	
28	Control	End	

Obrázek 8.17: Kroky testu na rozpoznávání symbolů - konec testu.

Pomocí tohoto či podobného testu lze testovat schopnost rozpoznávání různých symbolů či barev. Další příklad použití testu je při čtení symbolů z rušivého textu. Možnost takového vstupního obrázku je na obrázku 8.18.

Obrázek 8.18: Ukázka obrázku na poznávání čísel v šumu. Zde je číslo 69173.

Kapitola 9

Ukázka realizace

V této kapitole je popsáno využití některých prvků této platformy v Národním ústavu duševního zdraví (NÚDZ). V tomto ústavu probíhá projekt na testování schopností orientace různě postižených pacientů. K tomuto účelu se používá dříve zmíněná tzv. Blue-Velvet Aréna (BVA). Jde o kruhový stan, podobný malému cirkusovému šapito neobsahující pro pacienta v podstatě žádné stálé orientační body.

Stan má kruhovou podlahu s možností motorového otáčení, černé zdi (přesněji řečeno tmavé závěsy) a strop, takže skutečně neobsahuje žádné znatelné orientační body. Za zmíněnými tmavými, avšak vhodně průsvitnými, závěsy se po obvodu stanu na stěnách nachází osm rozsvěcujících se grafických displejů (obrázků). Na stropě je umístěna soustava dvou serv pohybujících laserem vytvářejícím světelnou značku na podlaze.

Při testu se postupně rozsvítí několik bočních obrázků na různých pozicích a laserem se zobrazí koncový cíl trasy na podlaze. Úkolem testovaného je označit na stěnách místa, kde se původně postupně rozsvítily obrázky a poté dojít na cílovou značku zobrazenou laserem.

Tento projekt se zdá velmi vhodný pro využití některých komponent zde vytvářené testovací platformy. Pro BVA byly tedy zhotoveny tři zařízení a jejich elementy:

- Zařízení pro otáčení podlahou a detekci aktuálního natočení.
  - Element1 (Out) – Spínání směru motoru pro otáčení podlahou.
  - Element2 (In) – Snímání úhlu natočení podlahy inkrementálním rotačním snímačem.
- Zařízení pro ovládání RGB Matrix 16x16 bodových displejů.
  - Element1-8 (Out) – 16x16 RGB matrix umístěné rovnoměrně po stěnách arény a umožňující zobrazit barevné obrázky podle vlastní předlohy.
- Zařízení pro ovládání laseru a soustavy dvou serv pro pohyb v osách XY. Tato serva umožňují směrovat laserový paprsek na libovolné místo podlahy arény.
  - Element1 (Out) – Pozice pro serva v osách XY.

- Element2 (Out) – Zapnutí/vypnutí svitu laseru.

Průměr BVA je zhruba 3 m a je umístěna v podstatě těsně vedle pracovního stolu. Původně byl návrh využít na propojení vytvořených zařízení sběrnici typu RS-485, protože se jedná v podstatě o rozsah jedné místnosti. Avšak po zhodnocení stavu (rozměrů arény, jejího umístění, umístění zařízení, délky propojovacích kabelů atd.) bylo přistoupeno k využití klasického USB rozhraní ve verzi HID. Na strop BVA byl umístěn mnoha-portový USB hub s vlastním napájením. Ten byl propojen pomocí 5 m kabelu s řídícím PC. Z tohoto USB hubu jsou poté rozvedeny datová USB do všech zařízení arény, tedy pro obrázky na stěnách, do zařízení pro otáčení podlahy i do zařízení pro pohyb laserem. Všechny procesory v zařízeních jsou napájeny přímo z USB.

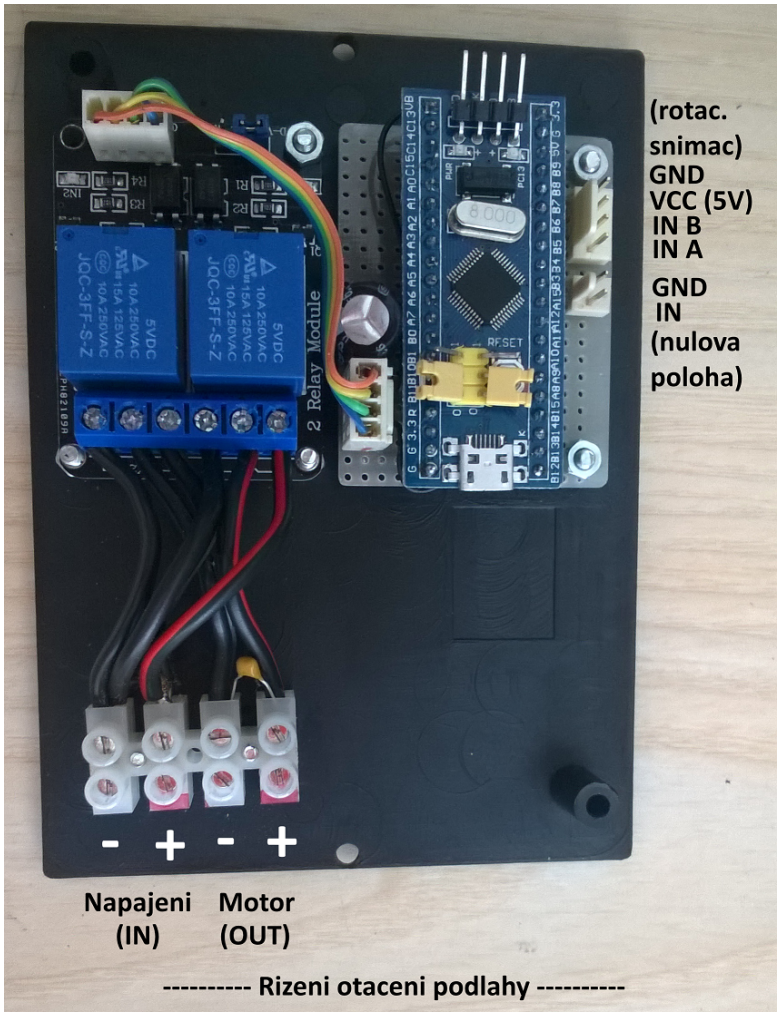
Silové napájení přes USB využívá pouze zařízení pro otáčení podlahou, a to pouze pro spínání dvou relé pro jízdu motoru. Motor je však napájen ze speciálního dostatečně silného zdroje 24 V. Pro napájení osmi obrázků (RGB maticovek) na stěnách a stropního zařízení (dvě serva a laser) je využit rovněž samostatný zdroj 5 V/20 A. Jedna plně rozsvícená RGB maticovka může vyžadovat až 3 A, ale v této instalaci by nikdy neměly svítit všechny připojené displeje současně.

Z navrhované platformy byly tedy využity tyto koncepty/komponenty/části:

- Princip a topologie platformy, komunikace mezi zařízením a aplikací, protokoly pro přenos dat atd.
- Základní SW pro činnost zařízení a k nim připojených výstupů/vstupů.
- Knihovna pro PC poskytující základní obsluhu zařízení platformy (HW část aplikace).

Řídící aplikace testu pro BVA není nikterak součástí této práce, a proto zde není ani nijak více zmíněna.

HW pro NÚDZ zhotovil vedoucí práce. Softwarová část byla vytvořena spolu se softwarem celé platformy, avšak upravena pro konkrétní potřeby dané instalace. Z této experimentální realizace plyne vhodnost navržené platformy pro zamýšlená použití.



Obrázek 9.1: Zařízení pro ovládání otáčení podlahy BVA.

Kapitola 10

Závěr

V této práci byla navržena a realizována univerzálnější platforma pro snadné testování a trénování nejen kognitivních schopností. Na začátku bylo potřeba zjistit, jak lze tyto kognitivní schopnosti vůbec testovat a jaké testy se v současnosti nejčastěji používají. Ukázalo se, že jednotlivé testy si jsou v podstatě velmi podobné, ale ve většině případů není možné hotové testy upravovat.

Po zjištění aktuálního stavu ve zmíněné oblasti byla navržena struktura nové dostatečně univerzální a modifikovatelné platformy. Vše řídí aplikace v počítači, do nějž jsou připojena externí zařízení (moduly s vlastními mikroprocesory). Do těchto zařízení lze připojit požadované koncové prvky, tedy reálné výstupy pro tvorbu podnětů testovanému a reálné vstupy pro snímání jeho reakcí.

Dále byly zvoleny vhodné způsoby propojení řídicí aplikace a externích zařízení pro malé, střední a případně i velké vzdálenosti. Dle různého rozsahu použití platformy byla ověřena komunikace pomocí USB, RS-485 a LAN. Součástí tohoto byl návrh vlastního datového protokolu co nejméně závislého na použité komunikaci. Navržený protokol umožňuje rozšířit platformu nejen o nová zařízení, ale také o úplně nové typy koncových prvků. Zároveň je možné daný protokol použít v budoucnu i se zcela novým typem komunikace.

Následně byla vytvořena vzorová externí zařízení řízená mikroprocesorem STM32F103. Tato zařízení komunikují navrženým protokolem s aplikací, nastavují požadované stavy na výstupy a odesílají zpět do aplikace informace o změnách vstupů. Na těchto vzorových zařízeních byl ověřen nejen navržený komunikační protokol, ale také navržené vzorové testy.

Poslední realizovaná část byla řídicí aplikace. Ta byla vytvořena pro operační systém MS Windows. Aplikace sestává ze dvou částí. První, tzv. HW část se stará o komunikaci mezi počítačem a zařízeními a vytváří v počítači datové struktury pro jednotlivá zařízení a jejich koncové prvky. Druhá část slouží ke správě, tvorbě a spouštění testů. Tato část přistupuje pouze k HW části aplikace, jež následně předává instrukce zařízením. V aplikaci lze nejen načíst dříve uložené testy, ale i vytvářet zcela nové. Všechny testy je zároveň možné kdykoli libovolně upravovat.

Součástí návrhu platformy je i způsob tvorby testů. Každý test se skládá z posloupnosti kroků různých typů, což jsou kromě práce s výstupy a vstupy

také možnosti přeskoků, podmínek, opakování a vkládání náhodných prvků do testu. Testy se také mohou podmíněně větvit. V testu lze používat i proměnné, čímž se tvorba testu přibližuje jednoduchému programovacímu/skriptovacímu jazyku. Tvorba testu ale zároveň zůstává stále jednoduchá.

Vykonávaný test vytváří záznam o svém průběhu. Do tohoto záznamu se ukládají všechny události testu a navíc lze vkládat vlastní zprávy definované při tvorbě testu. Tento záznam lze následně přímo vyhodnotit či předat k dalšímu podrobnějšímu vyhodnocení.

Pro demonstraci možností platformy a skutečně velkou variabilitu testů bylo vytvořeno několik vzorových demonstračních testů. Tyto testy ukazují možnosti od úplně jednoduchých testů přes pokročilejší až po komplexní, které jsou schopny za běhu samy vyhodnocovat reakce testovaného. Některé navržené testy jsou si velmi podobné a liší se pouze použitými typy koncových prvků.

Během tvorby platformy byly navržené principy využity rovněž pro podporu testování v Národním ústavu duševního zdraví. Pro toto byl upraven firmware zařízení na konkrétní požadavky daného použití. Tato část pomohla ověřit skutečnou realizovatelnost navržené platformy.

Závěrem lze tedy konstatovat, že navržená platforma je skutečně vhodná pro zamýšlené použití. Lze ji využít pro mnoho typů nejen kognitivních testů a cvičení. I když se v této fázi jedná spíše o prototyp tak jde o velmi dobrý nástroj v tomto oboru. V průběhu práce a zejména základního testování platformy vyvstaly návrhy na její další velmi užitečná rozšíření a prověření vlastností, například:

- Bylo by vhodné poněkud hlouběji ověřit, jaká je reálná latence mezi reakcí testovaného a záznamem v aplikaci.
- Vytvořit speciální typ testu pro měření extra rychlých reakcí.

Literatura

[1] GAŽOVÁ, Ivana, K. VLČEK, Z. NEDELSKÁ, I. MOKRIŠOVÁ, E. HYNČOVÁ, J. LACZÓ a J. HORT. Prostorová orientace při fyziologickém a patologickém stárnutí. *Česká a slovenská neurologie a neurochirurgie* [online]. 2012, **75/108**(4), 411-414 [cit. 2022-05-01]. ISSN 1802-4041. Dostupné z: https://www.csnn.eu/casopisy/ceska-slovenska-neurologie/2012-4/prostorova-orientace-pri-fyziologickem-a-patologickem-starnuti-38417

[2] Cognitive Functions. *NeuronUP. Web platform of cognitive rehabilitation* [online]. La Rioja, c2022 [cit. 2022-05-13]. Dostupné z: https://neuronup.us/areas-of-intervention/cognitive-functions/

[3] BENEŠOVÁ, Daniela. *Kognitivní funkce a pohybový výkon.* Plzeň: Západočeská univerzita v Plzni, 2020. ISBN 978-80-261-0998-3. Dostupné také z: https://www.researchgate.net/publication/351284423_kognitivni_funkce_a_pohybovy_vykon

[4] KUBÁSKOVÁ, Jana. Jak jdou obrázky po sobě?. *Náprava SPU* [online]. Plzeň [cit. 2022-09-12]. Dostupné z: http://naprava.spu.sweb.cz/cas/cas02.htm

[5] *Batak: A Total Fitness Training Solution* [online]. United Kingdom, c2020 [cit. 2022-09-02]. Dostupné z: https://www.batak.com/index.htm

[6] KOŠŤÁLOVÁ, Milena, Markéta BEDNAŘÍKOVÁ a Radka MICHALČÍKOVÁ. Zkouška vizuální pozornosti. *Fakultní nemocnice Brno* [online]. Brno [cit. 2022-11-23]. Dostupné z: https://www.fnbrno.cz/areal-bohunice/neurologicka-klinika/zkouska-vizualni-pozornosti/t4083

[7] Mini-Mental State Examination (MMSE). *Trusted Health Advice* [online]. 2022 [cit. 2022-10-03]. Dostupné z: https://www.healthdirect.gov.au/mini-mental-state-examination-mmse

[8] REKTOROVÁ, Irena. Screeningové škály používané v neurologii. *Neurologie pro praxi* [online]. 2011, 1.12.2011,

2011(Suppl.G), 37-45 [cit. 2022-12-04]. ISSN 1803-5280. Dostupné z: http://www.neurologiepropraxi.cz/pdfs/neu/2011/92/11.pdf

[9] BARTOŠ, Aleš, M. JANOUŠEK, R. PETROUŠOVÁ a M. HOHINOVÁ. Tři časy Testu kreslení hodin hodnocené BaJa skórováním u časné Alzheimerovy nemoci. *Česká a slovenská neurologie a neurochirurgie* [online]. 2016, **79/112**(4), 406-415 [cit. 2022-05-01]. ISSN 1802-4041. Dostupné z: https://www.csnn.eu/casopisy/ceska-slovenska-neurologie/2016-4-3/tri-casy-testu-kresleni-hodin-hodnocene-baja-skorovanim-u-casne-alzheimerovy-nemoci-58742/download?hl=cs

[10] GAZOVA, Ivana, Jan LACZÓ, Eva RUBINOVA, et al. Spatial navigation in young versus older adults. *Frontiers in Aging Neuroscience* [online]. 2013, **5**(94) [cit. 2022-12-04]. ISSN 1663-4365. Dostupné z: doi:10.3389/fnagi.2013.00094

[11] COGIT - Test mentální výkonnosti. *PSYCHOTESTY online* [online]. Jaroměř, c2008-2016 [cit. 2022-10-15]. Dostupné z: https://www.psychotesty.psyx.cz/vykonove-testy.php

[12] *Vítejte na palubě* [online]. Plumlov, 2020 [cit. 2022-10-14]. Dostupné z: https://skolakov.eu/

[13] HUDAK, Zoltan. Bluepill board support for Mbed OS 6. In: *Free open source IoT OS and development tools from Arm / Mbed* [online]. c2023 [cit. 2023-01-02]. Dostupné z: https://os.mbed.com/users/hudakz/code/mbed-os-bluepill/

[14] Medium-density performance line Arm®-based 32-bit MCU with 64 or 128 KB Flash, USB, CAN, 7 timers, 2 ADCs, 9 com. interfaces. In: *STMicroelectronics: Our technology starts with you - STMicroelectronics* [online]. c2022 [cit. 2022-12-21]. Dostupné z: https://www.st.com/resource/en/datasheet/stm32f103c8.pdf

[15] Visual Studio 2022 Community Edition — stáhnout nejnovější verzi zdarma. *Visual Studio: Integrované vývojové prostředí (IDE) a editor kódu pro vývojáře softwaru a týmy* [online]. Microsoft, c2022 [cit. 2022-08-12]. Dostupné z: https://visualstudio.microsoft.com/cs/vs/community/

Příloha A

Obsah přiloženého CD

- Diplomová práce ve formátu PDF.
- Spustitelná verze řídicí aplikace s ukázkovými testy, bez připojeného HW lze spustit pouze testy určené na počítač (F, G, H).

Aplikace stále prochází dalším vývojem. V případě zájmu o více informací je potřeba se obrátit na vedoucího práce.