



Fakulta Elektrotechnická
Katedra radioelektroniky

Bakalářská práce

Interaktivní hra využívající IoT prostředky

Interactive Game Based on IoT Technology

Jan Závorka

Vedoucí práce: **Ing. Stanislav Vítek, Ph.D.**

Studijní program: **Elektronika a komunikace**

Praha, Květen 2019

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Závorka** Jméno: **Jan** Osobní číslo: **466133**
Fakulta/ústav: **Fakulta elektrotechnická**
Zadávající katedra/ústav: **Katedra radioelektroniky**
Studijní program: **Elektronika a komunikace**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Interaktivní hra využívající IoT prostředky

Název bakalářské práce anglicky:

Interactive Game Based on IoT Technology

Pokyny pro vypracování:

Cílem práce je návrh a implementace platformy, která bude demonstrovat možnosti využití levných a energeticky nenáročných zařízení v IoT sítích. Aplikace bude mít podobu interaktivní hry pro více hráčů. Koncová zařízení budou založena na kytu Arduino (nebo podobném) s potřebnými periferiemi (dotykový displej) a budou pomocí protokolu navrženém studentem komunikovat s centrální jednotkou.

Seznam doporučené literatury:

[1] WAHER, Peter. Learning internet of things. Packt Publishing Ltd, 2015.
[2] IEEE INTERNET INITIATIVE, et al. Towards a definition of the Internet of Things (IoT). Revision-1, on-line: http://iot.ieee.org/images/files/pdf/IEEE_IoT_Towards_Definition_Internet_of_Things_Revision1_27MAY15.pdf

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Stanislav Vítek, Ph.D., katedra radioelektroniky FEL

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

Datum zadání bakalářské práce: **14.02.2019**

Termín odevzdání bakalářské práce: **24.05.2019**

Platnost zadání bakalářské práce: **20.09.2020**

Ing. Stanislav Vítek, Ph.D.
podpis vedoucí(ho) práce

prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) ústavu/katedry

prof. Ing. Pavel Ripka, CSc.
podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

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

Datum převzetí zadání

Podpis studenta

„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.“

V Praze dne 20. května 2019

.....

Podpis

Poděkování

Děkuji Ing. Stanislavu Vítkovi, Ph.D. za pomoc při vedení bakalářské práce.
Mé poděkování patří též Ing. Petru Skalovi za cenné rady a pomoc s 3D tiskem.

Abstrakt

Tato bakalářská práce se zabývá vývojem a výrobou jednoduchých zařízení, která umožňují demonstrovat využití energeticky nenáročných zařízení v IoT sítích. Propojená zařízení mají podobu jednoduché hry pro více hráčů - piškvorek a je postaveno na platformě Arduino. Vzniklá síť je tvořena až pěti koncovými zařízeními (klienty) s dotykovými displeji pro interakci s uživatelem a jednoho centrálního řídicího prvku (serveru). V práci je popsán konkrétní použitý hardware, včetně návrhu krabiček. Dále je zde podrobně rozepsán vytvořený software, včetně možnosti úprav pro použití s jinými moduly. Nakonec je uveden i návod na oživení a obsluhu.

Klíčová slova: Arduino Ethernet, IoT demonstrátor, Arduino hra, Arduino IoT

Abstract

This bachelor thesis deals with the development and production of a simple devices which can demonstrate usage of low power devices in IoT networks. This network of devices have a form of the multiplayer game - Noughts and crosses, and it is based on Arduino platform. The network have up to five end nodes (clients) with touch screen for interaction with the user and one control device (server). There is described used hardware including design of cases for all devices in this thesis. There is also a description of software including a list of possible changes which could be made for the purpose to use this product with different modules. In the end, there is a manual for starting and operating this devices.

Key words: Arduino Ethernet, IoT demonstration device, Game based on Arduino, Arduino IoT

Obsah

1	Úvod	1
2	Hardware	3
2.1	Zařízení typu server	3
2.2	Úprava Ethernet shieldu	4
2.3	Zařízení typu klient	5
3	Popis softwaru	7
3.1	Knihovny	7
3.2	Konfigurační hodnoty	7
3.2.1	Konfigurace serveru	8
3.2.2	Konfigurace klienta	9
3.3	Komunikace server - klient	13
3.4	Komunikace klient - server	13
3.5	Sestavení spojení	15
3.5.1	Příprava serveru	15
3.5.2	Příprava klienta	15
3.5.3	Sestavení pojení	17
3.6	Řízení průběhu hry	17
4	Oživení a obsluha	21
4.1	Ovládání klienta	21
4.2	Ovládání serveru	24
5	Závěr	27
	Reference	29
	Seznam použitého softwaru	31
A	Soubor s daty	32

Seznam obrázků

1	Schéma zapojení jednotlivých zařízení do sítě, zdroj [5, 6, 7, 8]	3
2	Schéma zapojení periferií u serveru	4
3	Návrh krabičky pro server	5
4	Zkompletovaná krabička pro server	5
5	Návrh krabičky pro klienta	6
6	Zkompletovaná krabička pro klienta	6
7	Diagram procesu přípravy a zapnutí serveru	16
8	Diagram procesu příjmu nového klienta na straně serveru	18
9	Diagram procesu spuštění hry	19
10	Diagram procesu řízení a ukončení hry	20
11	Obrazovka dvou klientů po úspěšném spuštění	21
12	Obrazovka dvou klientů při připojování k serveru	22
13	Obrazovky dvou různých klientů po úspěšném připojení k piškvorko- vému serveru	22
14	Obrazovka klienta po započaté hře (a) čekající hráč, (b) hrající hráč . .	23
15	Rozehraná hra dvou hráčů	23
16	Zobrazená hláška na displeji (a) výherce a (b) ostatních hráčů, barva textu se shoduje s barvou vítězného hráče	24

Seznam tabulek

1	Rozložení pole <i>board</i> pro přenos dat a řízení hry mezi serverem a klientem	14
2	Rozložení packetu pro komunikaci klient - server	14
3	Seznam příkazů dostupných pro server	25
4	Význam stavů svítivé diody na serveru	25

1 Úvod

Cílem práce bylo vytvořit zařízení, které by demonstrovalo využití jednodeskových počítačů v IoT sítích. Pro realizaci byla zvolena platforma Arduino. Důvodem pro zvolení této platformy bylo především její rozšíření mezi uživateli a dostupnost doplňkových periférií. Propojení jednotlivých zařízení by mělo mít funkci hry pro více hráčů. Koncová zařízení budou vybavena dotykovým displejem pro interakci s uživatelem. Celá hra pak bude řízena jednou centrální jednotkou taktéž založenou na platformě Arduino. Komunikace mezi jednotlivými zařízeními pak bude probíhat po Ethernetové síti. Pro dobrou názornost a nenáročnost (co se týče složitosti implementace, tak i potřebného hardwarového výkonu) byla zvolena hra piškvorky.

Práce je rozdělena do tří částí. V první části je popsán použitý hardware, včetně jeho případných úprav. Nechybí zde ani popis dalších komponent, které jsou potřeba pro komplekaci celého zařízení. Druhá část je zaměřena na popis softwaru, především na metody komunikace koncových zařízení. Pro názornost jsou některé důležité úkony doplněny diagramy. Poslední část je věnována už samotnému funkčnímu zařízení. Lze zde nalézt návod na zprovoznění, včetně fotografií funkčního prototypu.

Samotný pojem IoT („Internet of things“ - Internet věcí) označuje propojení fyzických zařízení prostřednictvím internetu. Jako koncová fyzická zařízení jsou většinou používány různé vestavěné systémy vybavené samotným řídicím mikrokontrolérem a řadou senzorů. Tato zařízení dokáží komunikovat jak mezi sebou, tak se zařízením uživatele (mobilním telefonem) nebo dokáží předávat naměřená data na cloud.

IoT samozřejmě není jen o koncových zařízeních, dalším důležitým prvkem jsou brány (*gateway*), které zajišťují spojení mezi koncovými zařízeními a například cloudem. IoT síť lze rozdělit do několika kategorií [1] podle fyzického rozmístění:

- *nanonetwork* ... spojení několika malých (velikost řádově mikrometrů) zařízení, která mají za úkol plnit jednoduché úkony.
- *NFC (Near-Field Communication)* ... spojení zařízení na vzdálenost řádově jednotek centimetrů.
- *BAN (Body Area Network)* ... spojení zařízení v oblasti lidského těla, zejména různá nositelná zařízení případně senzory uvnitř těla.
- *PAN (Personal Area Network)* ... síť v oblasti jedné místnosti.
- *LAN (Local Area Network)* ... síť v oblasti jedné budovy.
- *CAN (Corporate Area Network)* ... síť v oblasti jednoho kampusu/společnosti, spojuje několik lokálních sítí.
- *MAN (Metropolitan Area Network)* ... síť v oblasti jednoho města.
- *WAN (Wide Area Network)* ... síť pokrývající větší geografickou oblast, spojuje menší sítě.

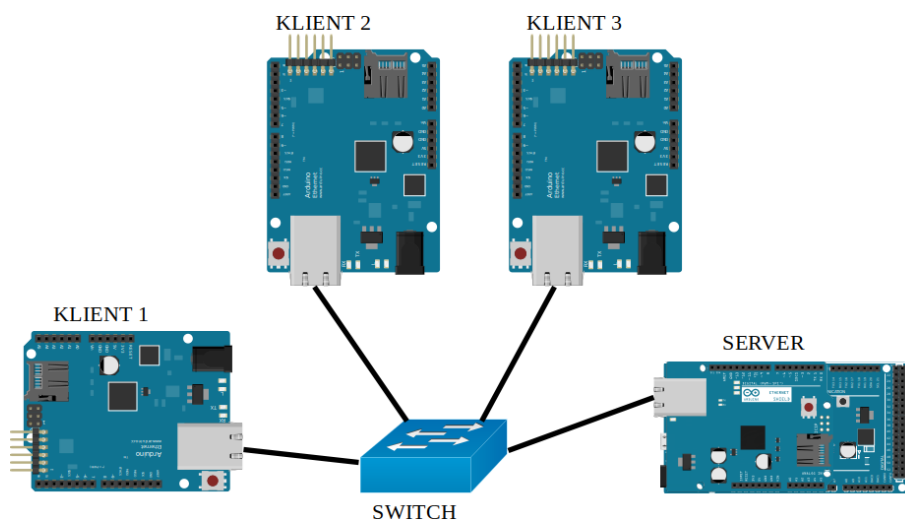
Z tohoto pohledu spadá vytvořená platforma do sítí typu *LAN*. Teoreticky by bylo možné ho připojit například do internetu, avšak problém je, že zařízení na to není stavěné - nemá implementovány žádné ochranné/šifrovací mechanismy ani žádné autentizační procesy. Díky hardwarové implementaci TCP/IP v kontrolérech WIZnet [2] by nemělo docházet k výraznému ovlivnění útoky typu DDoS [3].

Další důležitou součástí IoT je i implementace nových protokolů a standardů pro komunikaci. Mezi nejznámější lze například zařadit úspornou verzi Bluetooth - *BLE: Bluetooth low energy*, která se používá například u nositelných zařízeních. Dalším, spíše na průmysl zaměřeným, protokolem je *ZigBee*, který je známý díky svému dobremu zabezpečení. Pro použití v rozlehlých sítích s velkým počtem zařízení je vhodný LoRaWAN [4]. I když je na výběr v velké palety protokolů (lišící se například přenosovými rychlostmi, dosahem), byl pro komunikaci mezi zařízeními vybrán Ethernet. Důvodem je jednoduchost implementace, spolehlivost a navíc dostupnost Arduino desek s vestavěným kontrolérem.

2 Hardware

Celá platforma se skládá z jednoho řídicího prvku (viz. kapitola 2.1) a třech klientských zařízení (viz. kapitola 2.3), přičemž maximální počet je softwarově omezen na pět. Při použití s WIZnet W5100 kontrolérem na straně serveru lze maximálně připojit čtyři klienty. Datové spojení je realizováno hvězdicovou topologií (schéma na obrázku 1). Jako centrální prvek byl použit switch D-Link DGS-105. Celá demonstrační sestava se pak skládá z:

- 1x switch D-Link DGS-105
- 1x server s Arduino DUE
- 3x klient s Arduino Ethernet a dotykovým displejem
- 4x propojovací UTP kabel
- 1x napájecí adaptér pro switch (5 V/1 A součástí balení)
- 4x napájecí adaptér 12 V/1500 mA



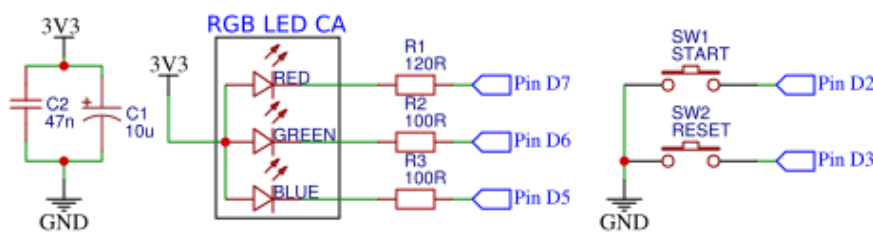
Obrázek 1: Schéma zapojení jednotlivých zařízení do sítě, zdroj [5, 6, 7, 8]

2.1 Zařízení typu server

Zařízení je založeno na desce *Arduino Due*. Tato deska obsahuje mikrokontrolér Atmel SAM3X8E ARM Cortex-M3 s 512 kB flash paměti a nabízí dostatečný výkon pro správné fungování serveru. Původní varianta totiž počítala s nasazením desky *Arduino Ethernet* i jako serveru. To se ovšem vzhledem k omezeným prostředkům ukázalo jako problematické, proto byla zvolena právě deska *Arduino Due*.

Pro připojení do sítě je použit Ethernetový shield s čipem Wiznet W5100, který je přímo napojen na Arduino. Tímto je dána ona limitace maximálně na 4 hráče, protože dle datasheetu výrobce čipu [2] je maximální počet spojení právě čtyři. Ethernetový shield komunikuje s Arduinem pomocí SPI sběrnice. V některých verzích Arduino Ethernet shieldu (především neoficiálních klonů) se objevují problémy se startem samotného shieldu. Stejným problémem trpěl i shield použitý v této práci, podrobný popis problému a použité funkční řešení je popsáno v kapitole 2.2.

Pro pohodlné ovládání jsou k serveru připojena dvě tlačítka a jedna barevná svítivá dioda. Význam jednotlivých stavů svítivé diody a funkce tlačítek je popsána v kapitole 4. Propojení těchto periférií s Arduinem je realizováno pomocí jednostranné DPS, schéma zapojení je pak na obrázku 2.



Obrázek 2: Schéma zapojení periférií u serveru

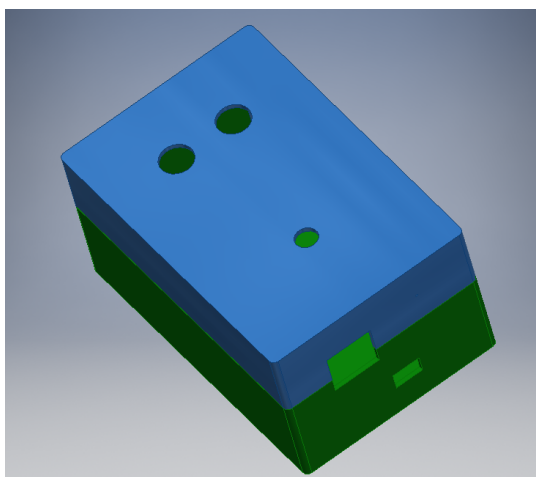
Napájení je řešeno externím adaptérem. Dle stránek výrobce [9] je možné použít napětí 6 - 16 V (využívá se interní stabilizátor), přičemž odběr je kolem 140 mA při napájení 12 V. V případě, že je pro ovládání použita sériová linka (server je připojen USB kabelem k počítači, ovládání tímto způsobem je popsáno v kapitole 4), postačuje napájení dodané přes USB kabel a není potřeba připojovat externí napájecí zdroj.

Celé zařízení je pak umístěno v krabičce, jejíž návrh je na obrázku 3 a realizace na obrázku 4. Krabička byla navržena v programu Autodesk Inventor Professional 2019 Student Edition a realizovaná 3D tiskem na tiskárně Original Prusa i3 MK3S. Krabička je osazena červeným a zeleným tlačítkem, barevnou svítivou 5 mm diodou (se společnou anodou) a souosým napájecím konektorem 5,5x2,1 mm. Pro upevnění Arduina jsou použity šrouby M2,5x10 a závitové vložky M2,5x6 vtavené do připravených otvorů v krabičce. Arduino deska sice nabízí montážní otvory o průměru 3 mm, ale vzhledem k rozložení součástí zde není dostatek místa pro hlavu šroubu M3.

2.2 Úprava Ethernet shieldu

Některé Arduino shieldy založené na kontroléru WIZnet (především neoficiální klony) mají problém se správným startem. Problém se projevuje tak, že při zapnutí napájení nedojde ke správnému připojení do sítě (indikační svítivé diody na shieldu sice blikají, ale Ethernet shield není k síti připojen). Tento problém se projevil pouze v případě, kdy bylo Arduino napájeno externím zdrojem (při připojení přes USB k počítači vše fungovalo normálně). Správné připojení k síti pak nastalo pouze pokud bylo stisknuto tlačítko *RESET* na Ethernet shieldu.

Dle webu [18] je problém způsoben krátkým resetovacím časem Arduina. Existuje několik způsobů, jak tento problém vyřešit. Pro Ethernet shield použitý v této práci byla aplikována metoda popsaná v [19]. Nejdříve bylo potřeba izolovat reset shieldu od resetu Arduina. To bylo provedeno odstříhnutím pinu *Reset* na Ethernet shieldu a odvrtáním prokovky, která vede resetovací signál z ICSP konektoru. Dále byl připojen rezistor 10 k Ω mezi pin *Reset* a napájecí pin 3,3 V. Nakonec byl připojen kondenzátor o kapacitě 1 μ F mezi pin *Reset* a zem (pin *Gnd*). Tímto způsobem byl daný problém vyřešen.



Obrázek 3: Návrh krabičky pro server



Obrázek 4: Zkompletovaná krabička pro server

2.3 Zařízení typu klient

Zařízení je založeno na desce Arduino Ethernet, která je vybavena mikrokontrolérem ATmega328 s 32 kB flash pamětí. Tato deska byla zvolena především kvůli tomu, že má vestavěný ethernetový kontrolér, který tak nezabírá piny pro připojení shieldu s displejem. Malá flash paměť se však během vývoje ukázala jako značně limitující, protože při nahrání všech potřebných knihoven (popsáno v kapitole 3.1) zůstalo k dispozici 30 % programové paměti. I z tohoto důvodu byla zvolena jako hra piškvorky, která není programově příliš složitá a také bylo nutné vynechat složitější menu například s nastavením barvy nebo změny IP adresy serveru (to se nyní musí provádět změnou v kódu a přeprogramováním Arduina, více v kapitole 3.2.2).

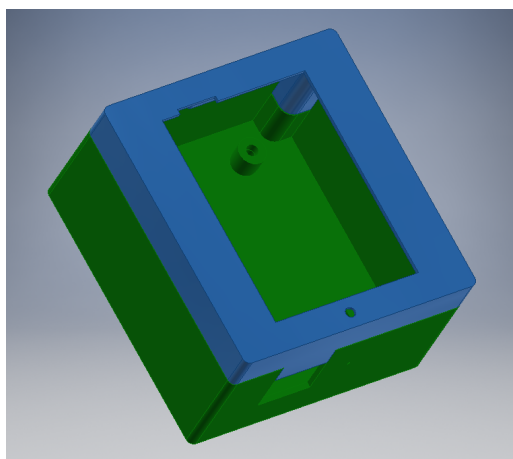
Jak už bylo zmíněno výše, tato deska má vestavěný ethernet kontrolér WIZnet, konkrétně typ W5100. U klienta není maximální počet spojení limitující, klient drží pouze jedno spojení se serverem. Jedinou nevýhodou kontroléru W5100 tak zůstává, že neobsahuje registr, ve kterém je uložena informace o fyzickém připojení ethernetového kabelu k desce [2]. Tím je zkomplikována detekce připojení a odpojení kabelu

a zůstává tak pouze možnost vizuální kontroly pomocí svítivých diod na konektoru RJ-45.

Pro interakci s uživatelem je klient vybaven 2,4" barevným TFT LCD displej s rozlišením 320x240 pixelů s rezistivní dotykovou plochou. Displej je vybaven řadičem SUM74HC245T. Vzhledem k rozměrům (výšce) RJ-45 konektoru, který je umístěn na desce, je nutné pro správné připojení použít lištu s oboustrannými kolíky o délce kolíku minimálně 15 mm. Protože u displejů použitých v tomto projektu byly kolíky připájeny už od výrobce, byla dodatečně vyrobena patice z dutinkové lišty a lišty s oboustrannými kolíky.

Pro napájení byl zvolen externí napájecí adaptér. Připojení na integrovaný stabilizátor není možné, protože Arduino s displejem při 5 V odebírá přibližně 400 mA, což integrovaný stabilizátor nedokáže poskytnout (vlivem velkého ztrátového výkonu dochází k jeho značnému zahřívání). Napájení přímo napětím 5 V není také příliš vhodné, neboť vlivem například ztrát přívodních vodičů může dojít ke kolísání napětí, čímž dojde i k pohybu reference AD převodníku připojenému k dotykové vrstvě displeje a tak může docházet k nesprávnému vyhodnocení stisku (může se lišit reálné místo stisku od toho, které vyhodnotil mikrokontrolér). Jako nejlepší varianta se ukázalo použití modulu se snižujícím DC-DC měničem. Modul obsahuje spínací regulátor MP1584 a dle dodavatele je schopen pracovat s napětím 6 - 25 V (při výstupním napětí 5 V) a dodat proud až 1,5 A, což je pro tuto aplikaci dostačující.

Stejně jako v případě serveru je celé zařízení umístěno ve vytištěné krabičce, návrh a realizovaná krabička jsou na obrázcích 5, 6. Princip uchycení Arduina je stejný jako v případě serveru, pro napájení je opět osazen souosý napájecí konektor 5,5x2,1 mm. Dále je z boku výřez pro konektor RJ-45 pro připojení do ethernetové sítě a na vrchu se nachází výřez pro displej, vedle kterého je umístěn otvor pro přístup k resetovacímu tlačítku.



Obrázek 5: Návrh krabičky pro klienta



Obrázek 6: Zkompletovaná krabička pro klienta

3 Popis softwaru

3.1 Knihovny

Pro realizaci byly použity následující knihovny:

- *Ethernet library*: knihovna slouží pro připojení Arduina k internetu. Jak je uvedeno na webových stránkách [10] jsou podporovány desky a shieldy založené na kontrolérech W5100, W5200, W5500. Tato knihovna je použita jak u serveru, tak i u klientů. U serveru připojený ethernet shield obsadí piny 10, 11, 12, 13. Komunikace Arduina se shieldem (ethernet kontrolérem) probíhá po SPI sběrnici.
- *MCUFRIEND_kbv library*: knihovna slouží k ovládání displeje u klienta. Pro správnou funkci je nutné, jak se zmiňuje autor na domovské stránce knihovny [11], mít k dispozici také knihovnu *Adafruit_GFX* [12]. Pro použití displeje se nejdříve vytvoří instance třídy *UTFTGLUE*, kde je nutné správně definovat piny pro daný shield, v tomto případě *UTFTGLUE LCD(0x0154, A2, A1, A3, A4, A0)*; . Pro řízení displeje jsou pak volány příslušné metody formou *LCD.metoda(parametry)*; , například pro vyplnění celého displeje černou *LCD.clrScr()*; . Pomocí ukázkového zdrojového kódu z této knihovny (*examples/TouchScreen_calibr_kbv*) byly získány kalibrační hodnoty pro dotykovou plochu displeje.
- *Adafruit_TouchScreen*: slouží k zaznamenání hodnot ze čtyřvodičové dotykové plochy. Stejně jako u knihovny pro displej je nejdříve nutné vytvořit instanci dané třídy *TouchScreen* v tomto případě tak *TouchScreen Touch(XP, YP, XM, YM, 300)*; , kde *XP, YP, XM, YM* jsou piny, na které je vyvedená dotyková plocha (pro použitý shield jsou to: *XP = 6, YP = A1, XM = A2, YM = 7*). Hodnota 300 by dle popisu knihovny [13] označuje elektrický odpor dotykové plochy X (měřeno multimetrem mezi piny *XP* a *XM*). Pro použité shieldy je hodnota přibližně 300 Ω. Dále je třeba určit minimální a maximální tlak pro vyhodnocení dotyku pomocí *#define MINPRESSURE* a *#define MAXPRESSURE*. Vyhovující jsou hodnoty uvedené v ukázkových kódech pro knihovnu a to konkrétně 10 pro *MINPRESSURE* a 1000 pro *MAXPRESSURE*.
- *SimpleTimer Library for Arduino*: jednoduchá knihovna, která slouží k řízení určitých časových událostí (kde není třeba velká přesnost), například pro zobrazení a skrytí chybových hlášek, blikání indikační svítivé diody u serveru. Dle webových stránek [14] je knihovna založená na funkci *millis()*; (vrací počet milisekund od začátku běhu programu) a nevyužívá přerušování ani hardwarový timer.

3.2 Konfigurační hodnoty

V této kapitole jsou popsány jednotlivé konfigurační hodnoty, by mohly být upraveny pro správnou funkci celého zařízení. Změny se vždy provádí přímo ve zdrojovém kódu, vždy pouze v hlavním souboru: *piskvorky_MP_client.ino* pro klienta a *piskvorky_MP_server.ino* pro server. Jak už bylo zmíněno v kapitole 2.3, vzhledem k malé paměti pro program nebylo možné implementovat menu, proto je nutné změny

provádět přímo ve zdrojovém kódu a dané zařízení pak přeprogramovat. V případě serveru je k dispozici microUSB konektor, který je dostupný skrze obdélníkový otvor na boku zařízení. V případě klienta je nutné demontovat víko, odpojit napájecí modul (v modré bužírce) a připojit Arduino k počítači pomocí převodníku USB - UART.

Jednotlivé konfigurační bloky jsou vždy ohraničeny komentáři, mezi nimiž se nacházejí jednotlivé proměnné, popis jejich funkce a omezení hodnot. Níže je uveden příklad ohraničení bloku pro nastavení sítě (IP adresa a port):

```
/* ----- KONFIGURACE - nastavení sítě -----*/

/* ----- KONEC - nastavení sítě -----*/
```

3.2.1 Konfigurace serveru

Nejprve je nutné vybrat síťový režim, v případě *ETHMODE_STATIC* je použita IP adresa, která je uložena v proměnné *serverAddress*. V případě volby *ETHMODE_DHCP* je adresa přiřazena DHCP serverem a je nutné dodržet aby se přiřazená adresa neměnila a byla zároveň nastavená u jednotlivých klientů.

Dále je třeba nastavit příslušnou MAC adresu, použitému Ethernet shieldu nebyla přiřazena výrobcem, takže je nutné nějakou zvolit. Vzhledem k tomu, že zařízení je provozováno na samostatné lokální síti, byla MAC adresa vybrána náhodně. V případě, že v síti jsou další zařízení, je možné vybrat MAC adresu například podle postupu popsaneho v [16]. Vezme se MAC adresa nějaké zařízení v síti a poslední bajt se zvětší o jedna a takto vzniklá MAC adresa se přiřadí shieldu. Vybraná MAC adresa je v poli *mac*.

Jako poslední je potřeba přiřadit port, na kterém budou zařízení komunikovat. Dle normy [15] je možné zvolit jakýkoliv dynamický port (49152-65535), protože tyto porty nebudou nikdy přiřazeny žádné službě. Port je uložen v proměnné *localPort* a stejný port musí být nastaven i u klientů.

```
/* !!! ----- KONFIGURACE - nastaveni ethernetu ----- !!! */
#define ETHMODE_STATIC //Varianty: ETHMODE_DHCP; ETHMODE_STATIC
byte mac[] = {
  0x54, 0xAB, 0x3A, 0x5B, 0x6F, 0x47
};
IPAddress serverAddress(10,0,0,8);
unsigned int localPort = 55555;
/* !!! ----- KONEC nastaveni ethernetu ----- !!! */
```

Blok konfigurace ethernet shieldu pro server

Jak je popsáno v kapitole 2.1, jsou k serveru připojena dvě tlačítka a jedna barevná svítivá dioda. V případě, že je nutné tyto periferie připojit jinak, je možné změnit čísla pinů v konfiguračním bloku *nastavení pinů a LED*. Tlačítka využívají interní pull-up rezistory a jejich zapojení je uvedeno na obrázku 2.

```

/* !!! ----- KONFIGURACE - nastaveni pinu a LED ----- !!! */
//Piny tlacitka start a reset
#define startPIN 2
#define resetPIN 3
//
//Piny na kterych jsou pripojeny jednotlivy barvy LED, musi podporovat
  PWM
#define LED_red 7
#define LED_green 5
#define LED_blue 6
//
//Nastaveni maximalniho jasv vsech LED, interval <0; 100>
const byte LED_br = 40;
/* !!! ----- KONEC nastaveni pinu a LED ----- !!! */

```

Blok konfigurace pinů pro svítivou diodu a tlačítka

Jako poslední je možné nastavit jakou barvu budou mít jednotliví klienti. V ukázkovém případě jsou barvy přiřazeny z výběru, který je uveden v kódu, ale je možné také definovat vlastní ve formátu RGB565 (16-bit barva).

```

/* !!! ----- KONFIGURACE - nastaveni barev hracu ----- !!!
  */
#define PLAYER1COLOR RED //0xF800
#define PLAYER2COLOR GREEN //0x07E0
#define PLAYER3COLOR PURPLE //0x780F
#define PLAYER4COLOR GREENYELLOW //0xAFE5
#define PLAYER5COLOR OLIVE //0x7BE0
/* !!! ----- KONEC nastaveni barev hracu ----- !!! */

```

Blok konfigurace jednotlivých barev pro klienty/hráče

3.2.2 Konfigurace klienta

Klientů se v celé sestavě nachází několik (v tomto případě tři). Ti se liší pouze určitým nastavením (IP adresa, MAC adresa, kalibrační hodnoty displeje), proto jsou vytvořeny profily pro jednotlivé klienty (softwarově omezeno na pět). Každému profilu se nastaví požadované parametry a při nahrávání programu do Arduina se pak profily mění pomocí `#define CLIENTx`, kde `x` je číslo jednotlivých klientů/profilů. Pokud není zvolen žádný profil, použity jsou defaultní hodnoty a uživatel je o tom informován při překladu pomocí direktivy `#warning` (zobrazí se hlášení, ale překlad neukončí).

```

/* ----- KONFIGURACE - nastaveni schemat-----*/
#define CLIENT1 //Vyber profilu pro klienta
/* ----- KONEC - nastaveni schemat-----*/

```

Blok výběru profilu pro klienta

V této části se přiřazují jednotlivé MAC adresy daným profilům. Výhodou oficiálních Arduino Ethernet desek je, že mají MAC adresu přidělenou výrobcem a je možné ji nalézt na bílém štítku ze spodní strany.

```
#ifndef CLIENT1
//Klient 1
byte mac[] = {
  0x90, 0xA2, 0xDA, 0x11, 0x08, 0x18
};
#elif defined(CLIENT2)
//Klient 2
byte mac[] = {
  0x90, 0xA2, 0xDA, 0x11, 0x09, 0x78
};
#elif defined(CLIENT3)
//Klient 3
byte mac[] = {
  0x90, 0xA2, 0xDA, 0x11, 0x08, 0xA0
};
#elif defined(CLIENT4)
//Klient 4
byte mac[] = {
  0xDE, 0xAD, 0xBE, 0xEE, 0xFE, 0xBD
};
#elif defined(CLIENT5)
//Klient 5
byte mac[] = {
  0xDE, 0xAD, 0xBE, 0xEE, 0xFE, 0xAD
};
#else
//Defaultni hodnoty
#warning "Je pouzita defaultni MAC adresa"
byte mac [] = {
  0xDE, 0xAD, 0xBE, 0xEE, 0xFE, 0xAB
}
#endif
/* ----- KONEC - nastaveni mac adres-----*/
```

Blok přiřazení MAC adres jednotlivým klientským profilům

Stejně jako v případě serveru i zde je možné vybrat ze dvou režimů sítě, a to *ETHMODE_STATIC* a *ETHMODE_DHCP*. V případě volby *ETHMODE_DHCP* není na adresu přiřazenou DHCP serverem kladen žádný zvláštní nárok. V případě použití režimu *ETHMODE_STATIC* je ještě nutné dodefinovat IP adresy pro jednotlivé klienty. Opět lze s výhodou využít profilů. Pokud se použije defaultní, je o tom uživatel opět při překladu informován. Kromě toho je ještě nutné doplnit IP adresu serveru, ke kterému se budou klienti připojovat. Ta byla nastavena při konfiguraci serveru (kapitola 3.2.1) stejně tak jako port, který je také nutno zvolit stejný.

```
/* ----- KONFIGURACE - nastaveni site -----*/
#define ETHMODE_STATIC //Varianty: ETHMODE_DHCP; ETHMODE_STATIC

#ifdef ETHMODE_STATIC
#ifdef CLIENT1
IPAddress clientAddress(10,0,0,138);

#elif defined(CLIENT2)
IPAddress clientAddress(10,0,0,139);

#elif defined(CLIENT3)
IPAddress clientAddress(10,0,0,140);

#elif defined(CLIENT4)
IPAddress clientAddress(10,0,0,141);

#elif defined(CLIENT5)
IPAddress clientAddress(10,0,0,142);
#else
//Defaultni nastaveni
IPAddress clientAddress(10,0,0,100);
#warning "Pozor - je pouzito defaultni nastaveni IP, pro kazdeho
        klienta nutno zmenit"
#endif
#endif
IPAddress serverAddress(10,0,0,8); //Nastaveni IP adresy serveru
unsigned int localPort = 55555; //Port
/* ----- KONEC - nastaveni site -----*/
```

Blok nastavení síťového režimu a IP adres

Jako poslední je nutné nastavit kalibrační hodnoty dotykové plochy displeje. Nejjednodušší způsob jejich získání je použít ukázkový program v knihovně *MCUFRIEND_kbv library*, který lze nalézt v (*examples/TouchScreen_calibr_kbv*). V ukázkovém programu je třeba upravit podle použitého shledu nastavení pinů. Poté stačí postupovat podle pokynů na displeji a výsledek se vypíše na sériovou linku (lze použít integrovaný v Arduino IDE - *Tools->Serial Monitor*). Krom kalibračních hodnot je také nutné doplnit orientaci displeje (`#define TOUCH_LANDSCAPE` nebo `#define TOUCH_PORTRAIT`). V případě, že nebude zvolena ani jedna možnost, překlad bude ukončen a bude vypsána chybová hláška (direktiva `#error`). I v tomto případě lze s výhodou využít profilů.

```

/* ----- KONFIGURACE - kalibrace displeje -----*/
#ifdef CLIENT1
//Klient 1
#define TOUCH_XMIN 221
#define TOUCH_XMAX 950
#define TOUCH_YMIN 200
#define TOUCH_YMAX 950
#define TOUCH_LANDSCAPE
#elif defined(CLIENT2)
//Klient 2
#define TOUCH_XMIN 233
#define TOUCH_XMAX 937
#define TOUCH_YMIN 210
#define TOUCH_YMAX 910
#define TOUCH_LANDSCAPE
#elif defined(CLIENT3)
//Klient 3
#define TOUCH_XMIN 230
#define TOUCH_XMAX 960
#define TOUCH_YMIN 220
#define TOUCH_YMAX 920
#define TOUCH_LANDSCAPE
#elif defined(CLIENT4)
//Klient 4
#define TOUCH_XMIN 0
#define TOUCH_XMAX 100
#define TOUCH_YMIN 0
#define TOUCH_YMAX 100
#elif defined(CLIENT5)
//Klient 5
#define TOUCH_XMIN 0
#define TOUCH_XMAX 100
#define TOUCH_YMIN 0
#define TOUCH_YMAX 100
#else
//Defaultni nastaveni, je nutne upravit
#define TOUCH_XMIN 0
#define TOUCH_XMAX 100
#define TOUCH_YMIN 0
#define TOUCH_YMAX 100
#endif
/* ----- KONEC - kalibrace displeje -----*/

```

Blok nastavení kalibračních hodnot dotykové plochy

3.3 Komunikace server - klient

Server komunikuje s klienty pomocí pole *board*. Jedná se o jednorozměrné pole typu *byte*¹ o velikosti 136 (velikost musí být dělitelná osmi). V tomto poli jsou vyplněné všechny důležité informace o stavu hry i jednotlivých klientech. Význam jednotlivých hodnot v poli *board* je popsán v tabulce 1.

Při odesílání je toto pole rozděleno na části po osmi bajtech. Každá tato část je vybavena pořadovým číslem a dvoubajtovým kontrolním součtem (v něm je zahrnuto i pořadové číslo) a následně odeslána připojeným klientům.

Klienti postupně přijímají všechny části a v případě bezchybného příjmu data vyhodnotí. V případě, že klient na základě kontrolního součtu identifikoval chybnou část, odešle požadavek podle pravidel komunikace klient - server (popsáno v kapitole 3.4). Server pak v případě přijetí požadavku odešle danému klientovi vyžadovanou část pole.

3.4 Komunikace klient - server

Klient komunikuje se serverem, až na výjimku při sestavení spojení, prostřednictvím dvoubajtového pole. Popis jednotlivých částí pole je v tabulce 2. Pro kontrolu přenosu se každá zpráva odešle třikrát. Na straně serveru se pak porovnají dvě ze tří přijatých zpráv a ostatní přijatá data se zahodí. Klient může poslat data maximálně jednou za sekundu (posláním dat se rozumí odeslání tří stejných zpráv).

Pokud při porovnání server zjistí, že přijaté zprávy nejsou stejné, odešle danému klientovi znovu herní desku. V případě, že byl daný hráč na tahu a přenos požadavku na vyplnění pole se nezdařil, je mu prostřednictvím znovuodeslání pole znovu aktivován tah.

V případě, že klient přijal od serveru chybná data, odešle požadavek ve tvaru $\{20, x\}$, kde x je číslo chybně přijaté části pole. Pokud je toto serverem správně přijato, odešle požadovanou část znovu (dle pravidel komunikace server - klient, kapitola 3.3).

¹Dle webu Arduina [17] se jedná o stejný typ jako *unsigned char*. Kvůli konzistentnosti stylu programování Arduina je však doporučeno používat datový typ *byte*.

Tabulka 1: Rozložení pole *board* pro přenos dat a řízení hry mezi serverem a klientem

Index	Hodnoty	Popis
0-89		Každý index odpovídá jednomu čtverečku na herní desce piškvorek
	0	Pole je prázdné (neobsazené)
	1-5	Obsazeno některým hráčem/klientem
90		Přenos řídicí informace pomocí kódu
	0	nic nedělat
	1	Překreslit obrazovku, nějaký hráč je na tahu (herní mříž)
	1	Pouze překreslit obrazovku (herní mříž)
	3	Příprava nové hry, zobrazit úvodní obrazovku
	9	Informace pro hráče, že bude odpojen
	100	Hra skončila remízou
	10x	Hodnota podle hráče, který vyhrál: 101-105 ('x' je číslo hráče)
	20x	Problémy/odpojení daného hráče: 201-205 ('x' je číslo hráče)
91	1-5	Číslo hráče, který je na tahu, pokud je 0, nikdo nehraje
93	0-89	Počet odehraných kol (vyplňuje server)
95-96	Kód barvy	Barva hráče 1
97-98		Barva hráče 2
99-100		Barva hráče 3
101-102		Barva hráče 4
103-104		Barva hráče 5
105-108	IPv4 adresa	IP adresa hráče 1
109-112		IP adresa hráče 2
113-116		IP adresa hráče 3
117-120		IP adresa hráče 4
121-124		IP adresa hráče 5

Tabulka 2: Rozložení packetu pro komunikaci klient - server

Index	Hodnoty	Popis
0	10	Jedná se o informaci, že další přenesený bajt bude číslo vyplněné herní pozice v poli <i>board</i>
	20	Žádost klienta o poslání dané části herního pole (<i>board</i>), v další bajtu je číslo dané části <i>boardu</i>
1	0-89, 0-16	Podle hodnoty předchozího bajtu: index vyplněného pole nebo pořadí packetu, který má být poslán znovu

3.5 Sestavení spojení

3.5.1 Příprava serveru

Při spuštění serveru dojde nejprve k identifikaci ethernet kontroléru. V případě, že je identifikován kontrolér WIZnet W5200 nebo WIZnet W5500, je k dispozici funkce `Ethernet.linkstatus()`, která umožňuje získat informaci o fyzickém připojení kabelu. Pokud není kabel připojen, je toto signalizováno blikáním signalizační svítivé diody červenou barvou (viz. tabulka 4) a program nepokračuje dále, dokud není kabel připojen.

V případě, že server získává adresu z DHCP serveru, je možné ještě ověřit správné připojení pomocí kontroly získané adresy. V knihovně Arduino Ethernet [10] je toto řešeno návratovou hodnotou funkce `Ethernet.begin(mac)` (v režimu DHCP má pouze jeden parametr a to lokální MAC adresu, jinak funkce `Ethernet.begin()` nemá návratovou hodnotu). Proces přípravy serveru je znázorněn na obrázku 7.

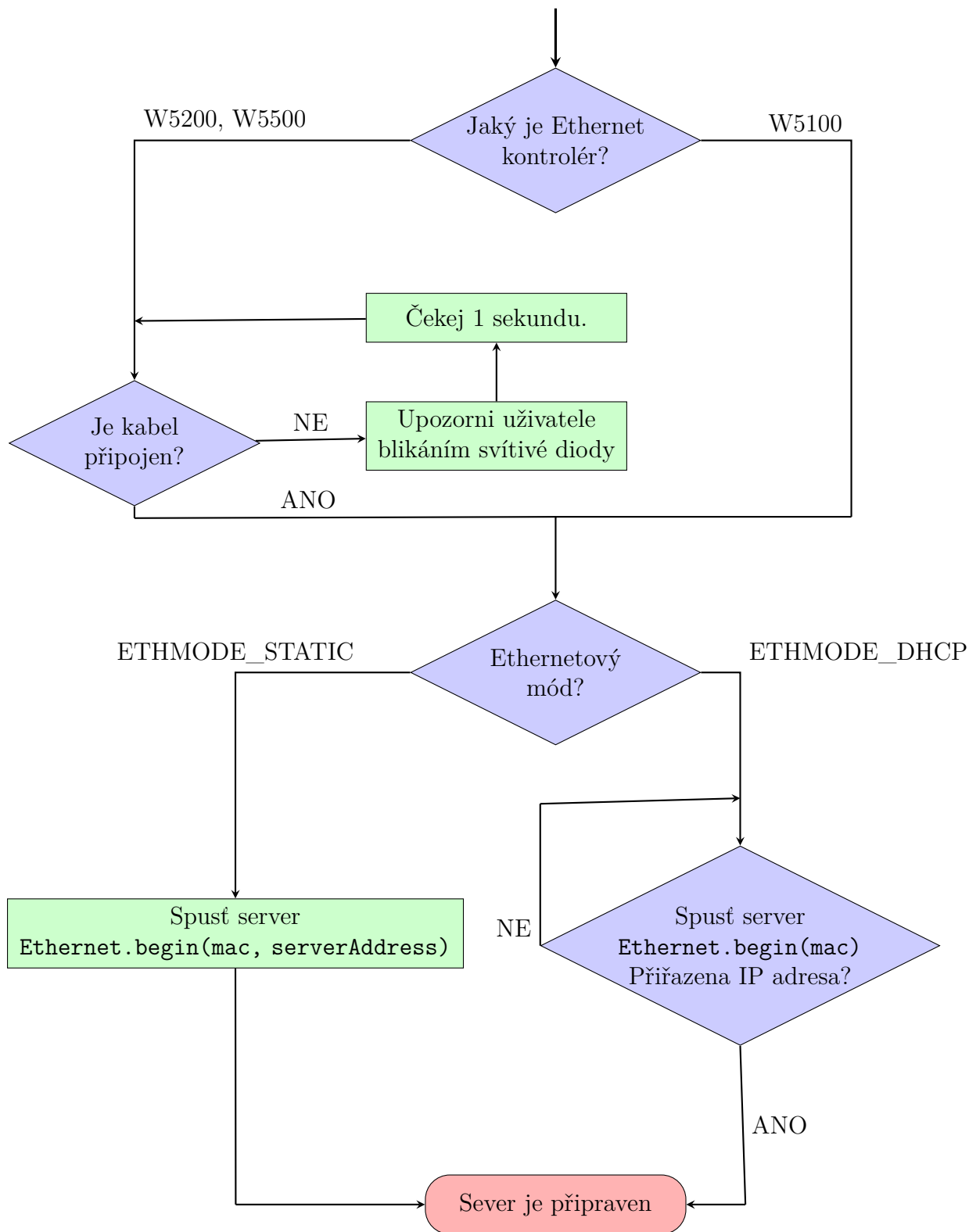
Pokud je však nastaven mód se statickou IP adresou a je použit shield s ethernet kontrolérem například WIZnet W5100 (jako je tomu v této práci), není zde žádná vhodná možnost kontroly správného připojení. Jedna z možností by byla, že by se server pokusil poslat zprávu *Echo Request* nějakému známému zařízení v síti a očekávat odpověď *Echo Reply*. Ale vzhledem k tomu, že jako centrální prvek je v tomto zapojení zvolen switch a žádné další zařízení v síti není, je i tato možnost nepoužitelná.

V této fázi je příprava serveru dokončena, postup navázání spojení s klientem je popsán v následující kapitole 3.5.3.

3.5.2 Příprava klienta

Příprava probíhá obdobně jako u serveru, ale vzhledem k tomu, že použité Arduino Ethernet desky mají kontrolér WIZnet W5100, je vynechána část s kontrolou připojení kabelu. Jediná možná detekce správného připojení kabelu je tak pouze v režimu, kdy je IP adresa získávána z DHCP serveru. V takovém případě se na displeji vypíše zpráva *Zkontrolujte připojení kabelu* a klient čeká dokud IP adresu neobdrží. V opačném případě je přepnuto na úvodní obrazovku.

V této fázi je klient připraven a čeká na příkaz uživatele připojit se k serveru.



Obrázek 7: Diagram procesu přípravy a zapnutí serveru

3.5.3 Sestavení pojení

Pokud nastavení a spuštění serveru proběhlo správně, server v nekonečné smyčce (`void loop() { }`) pravidelně ověřuje, zda nějaké zařízení nevyžaduje připojení. To je realizováno metodou z knihovny [10] - `server.accept()`. V případě příchozího spojení vrátí tato metoda objekt typu `client`, v opačném případě `NULL`. To je uloženo do proměnné `newClient`. Metoda `server.accept()` vrací data o daném příchozího spojení pouze jednou a proto je nutné je uložit do extra proměnné.

Připojení klienta spustí uživatel stisknutím příslušného tlačítka na displeji. Klient se v pravidelných intervalech pokouší o navázání spojení funkcí `client.connect(serverAddress, localPort)`. V případě, že je připojení k serveru úspěšné, pošle klient serveru zprávu „100“. Tento kód informuje server, že se jedná o klienta pro piškvorky, pokud by server přijal jiná data, spojení zruší.

Pokud server zaznamená nové příchozí spojení, spustí kontrolu nového hráče. Celý proces připojení nového hráče je naznačen na obrázku 8.

V této fázi se čeká na připojení dalších klientů. Pravidelně je také kontrolováno, zda se nějaký klient neodpojil. V případě, že ano, jsou jeho data smazána jak z pole `clients[]`, tak z herního pole `board` a o této skutečnosti jsou ostatní klienti informováni zprávou (odeslání pole `board`) s kódem 20x, kde `x` je číslo odpojeného klienta. Tato zpráva je zobrazena na displejích klientů po dobu 4000 ms (nastaveno v proměnné `clientErrorMessageLast`).

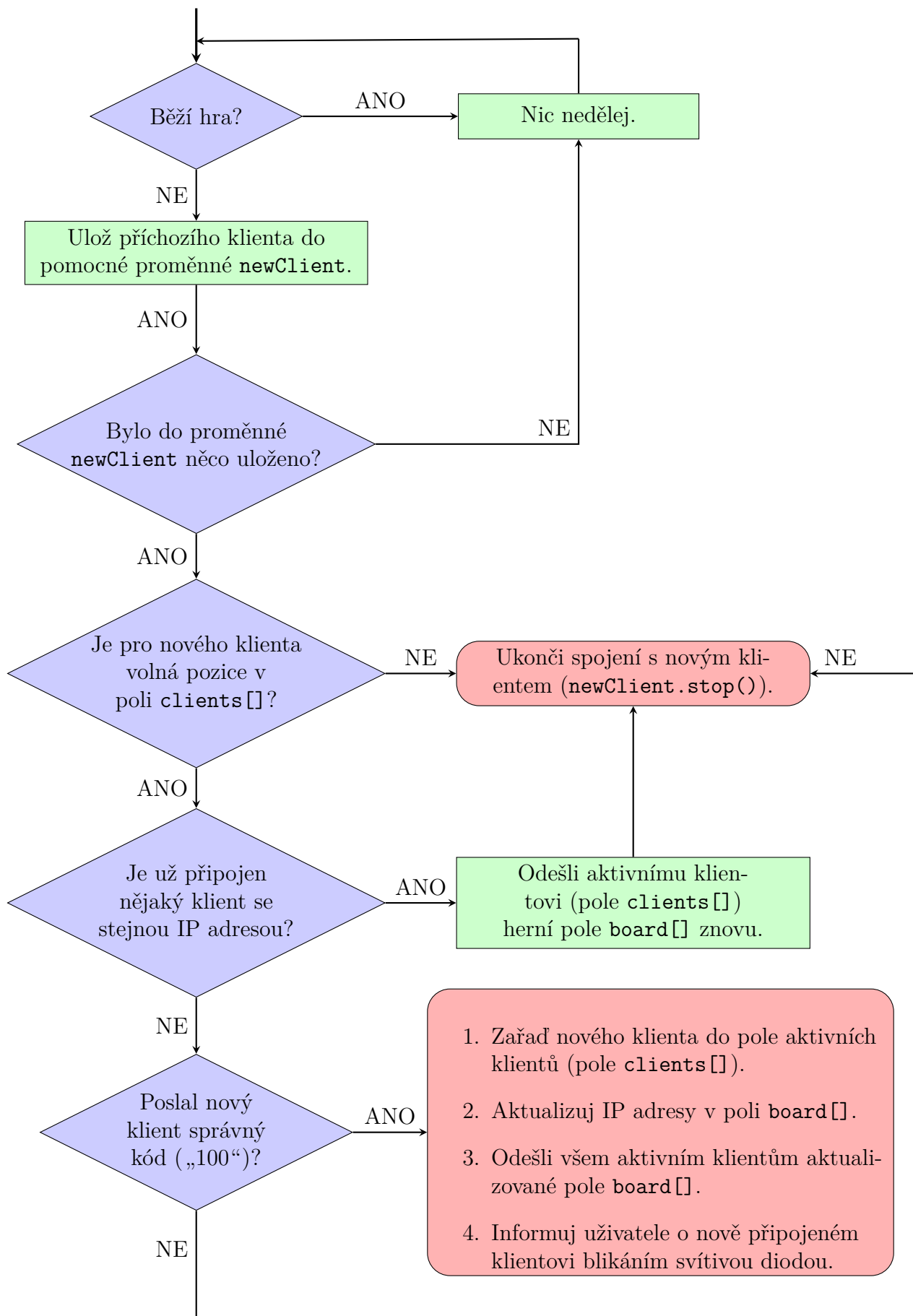
Nyní se čeká na zahájení hry ze strany uživatele (stisk zeleného tlačítka na serveru nebo příkazem přes sériovou linku, viz. kapitola 4).

3.6 Řízení průběhu hry

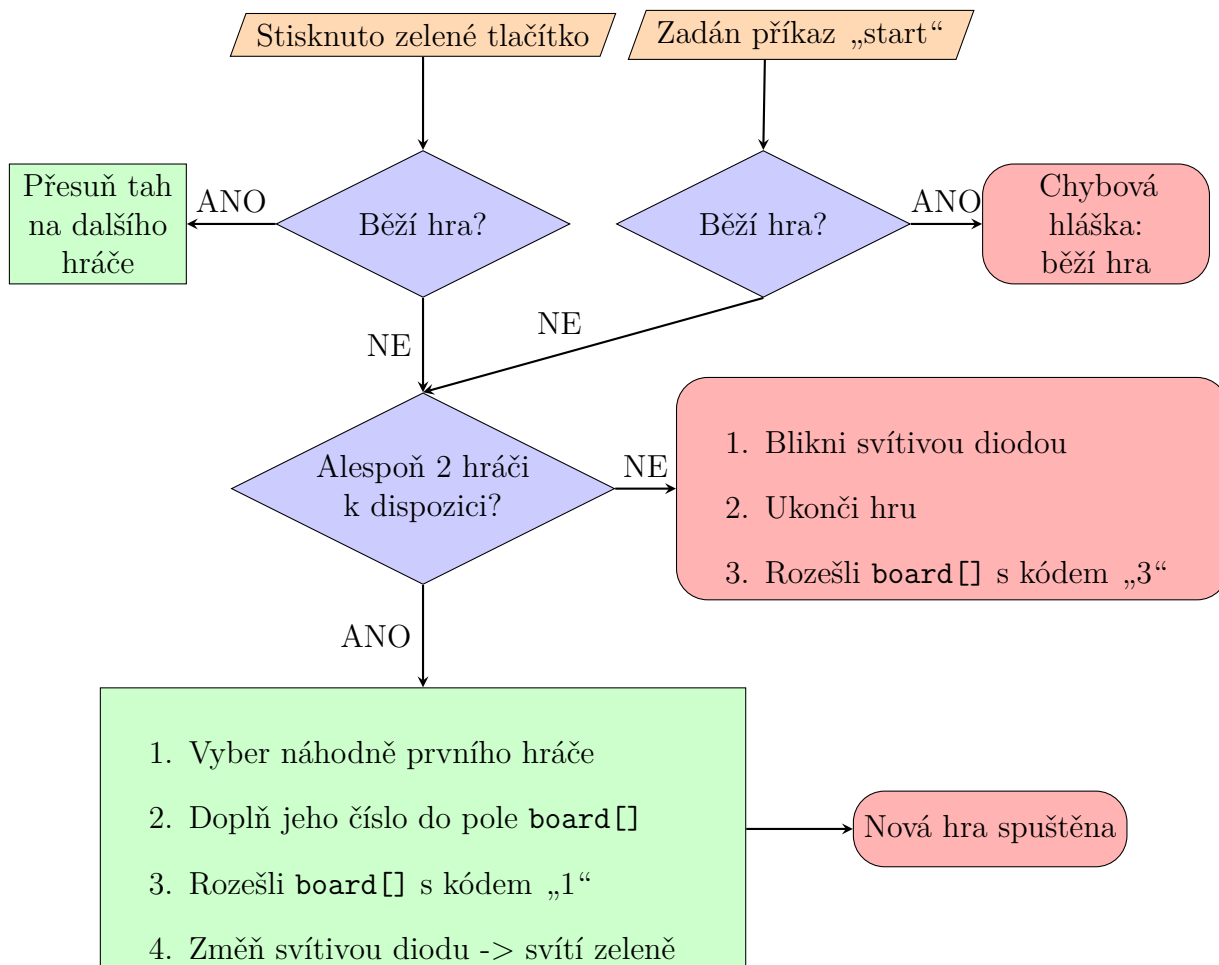
Po příkazu ke spuštění hry se nejdříve ověří, zda jsou k dispozici alespoň dva hráči. Pokud tomu tak není, hra se nezahájí a uživatel je o tom informován indikační svítivou diodou (viz. tabulka 4).

V případě, že je k dispozici dostatečný počet hráčů, je náhodně vybrán (funkce `random()`) první hráč a všem hráčům se rozešle pole `board[]` s vyplněným herní kódem (1) a číslem hráče (metoda odeslání popsána v kapitole 3.3). Jednotliví hráči porovnají pořadí svých IP adres v poli `board[]` a zjištěný index porovnají s číslem hráče vyplněným na pozici 91 v poli `board[]`, hráč, který nalezne shodu, je na tahu, ostatní vyčkávají. Proces startu hry je na obrázku 9.

Server nyní čeká dokud mu hrající klient neodešle pozici, kterou si přeje vyplnit. Komunikace klienta se serverem se řídí podle pravidel popsanych v kapitole 3.4. Přeskočit daného hráče lze pomocí stisku zeleného tlačítka na serveru. V případě, že si hrající uživatel přeje vyplnit danou pozici, stiskne patřičné místo na displeji. Klient zkontroluje, zda je daná pozice volná a pokud ano okamžitě vyplní žeton a následně odešle informaci o vyplnění serveru. Okamžité vykreslení žetonu před potvrzení serverem je implementováno proto, že při delší odezvě serveru může docházet ke zpoždění vykreslení (v poměru s dotykem), což je uživatelsky nekomfortní.



Obrázek 8: Diagram procesu příjmu nového klienta na straně serveru

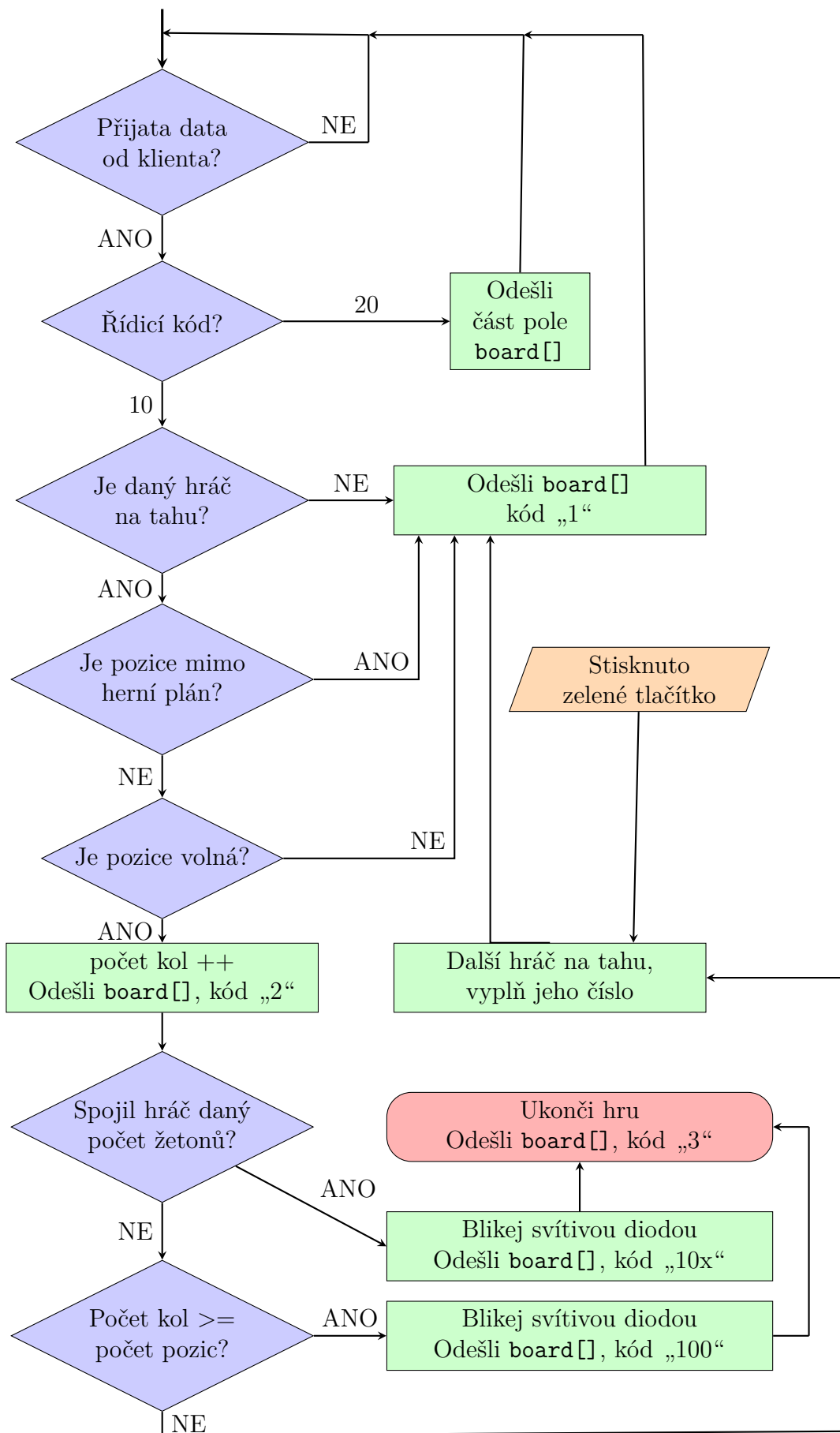


Obrázek 9: Diagram procesu spuštění hry

Pokud by server vyplnění dané pozice zamítl, je při dalším překreslení displeje daný žeton překreslen barvou pozadí. Celý proces zpracování a vyhodnocení tahu je znázorněn na obrázku 10.

Během hry se také ověřuje, zda se nějaký hráč neodpojil, pokud ano, jeho IP adresa je smazána z pole `board[]` a na straně serveru je smazán z pole aktivních klientů (`clients[]`). Všem připojeným klientům je odesláním pole `board[]` s kódem „20x“ (hlášení o odpojení klienta). Během hry se však žádný klient nemůže připojit ani v případě, že ve hře při jejím započítí byl. Žetony odpojených klientů zůstávají ve hře. Pokud by došlo k odpojení všech klientů, server hru ukončí.

V případě, že je hra ukončena, dojde k resetu pole `board[]`: vymaže se obsazenost herní desky, vynuluje se aktuální hráč a počet kol, znovu se vyplní IP adresy aktivních klientů a znovu se přiřadí barvy. Klientům je poslán `board[]` s kódem „3“, který u nich vyvolá překreslení na úvodní obrazovku (viz. obrázek 13).



Obrázek 10: Diagram procesu řízení a ukončení hry

4 Oživení a obsluha

Příprava ke spuštění probíhá v několika krocích:

1. Připraví se centrální jednotka (v tomto případě switch) a k ní se pomocí síťového kabelu připojí všechna zařízení (klienti i server). Zapojení je znázorněno na obrázku 1.
2. Připojit napájení k serveru i klientům, pro server napájení 6 - 16 V DC schopné dodat alespoň 250 mA, pro klienty 6 - 25 V DC alespoň 500 mA.
3. Pokud je vše v pořádku, na serveru svítí svítivá dioda modrou barvou a na displejích klientů je úvodní obrazovka s tlačítkem „*PRIPOJIT*“ (viz. obrázek 11).

4.1 Ovládání klienta

1. Po úspěšném zapnutí se vykreslí úvodní obrazovka (obrázek 11). Zobrazeny jsou informace o nastavené IP adrese serveru a o IP adrese klienta (podle režimu přiřazena DHCP serverem nebo ručně).



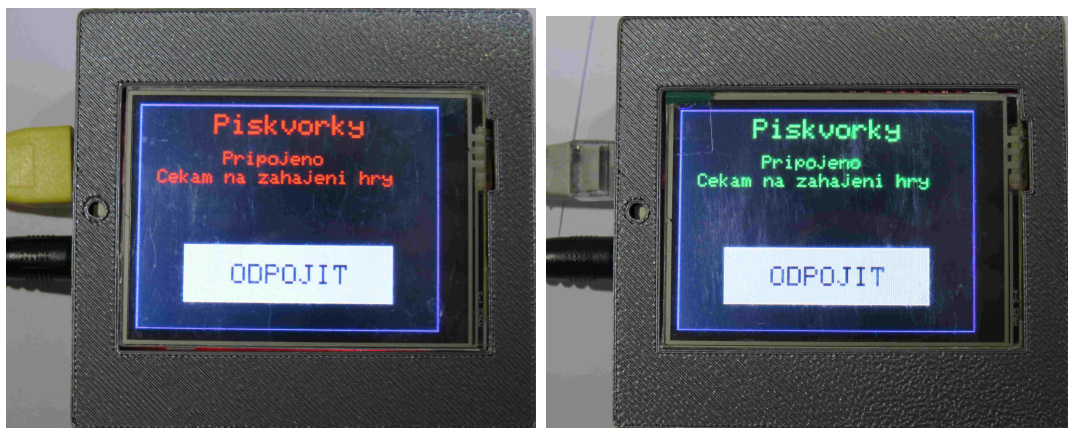
Obrázek 11: Obrazovka dvou klientů po úspěšném spuštění

2. Připojení k piškvorkovému serveru lze spustit stisknutím tlačítka „*PRIPOJIT*“. Připojování je indikováno pomocí displeje (obrázek 12). Připojování může být kdykoliv přerušeno stiskem tlačítka „*PRERUSIT*“. Připojování bude dokončeno pokud na serveru není rozehraná hra.



Obrázek 12: Obrazovka dvou klientů při připojování k serveru

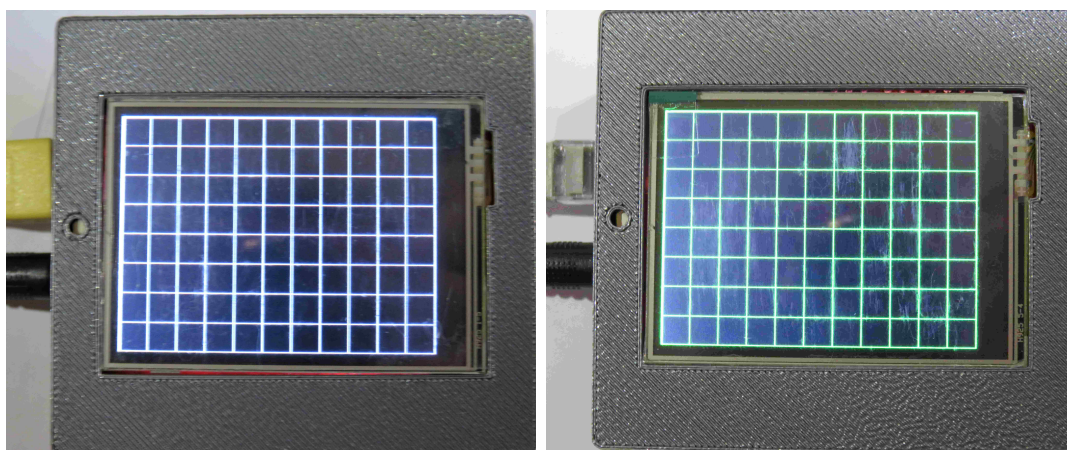
3. Pokud připojení proběhlo úspěšně, klient má od serveru přiřazené číslo a barvu. To je indikováno pomocí displeje (obrázek 13), kde barva textu odpovídá barvě hráče. Než začne hra je možné se ze serveru odpojit stiskem tlačítka „ODPOJIT“. Jinak se čeká na zahájení hry.



Obrázek 13: Obrazovky dvou různých klientů po úspěšném připojení k piškvor-
kovému serveru

4. V případě, že ze strany serveru byla započata hra (příkazem nebo stiskem zeleného tlačítka na serveru), vykreslí se na obrazovce herní pole (obrázek 14).

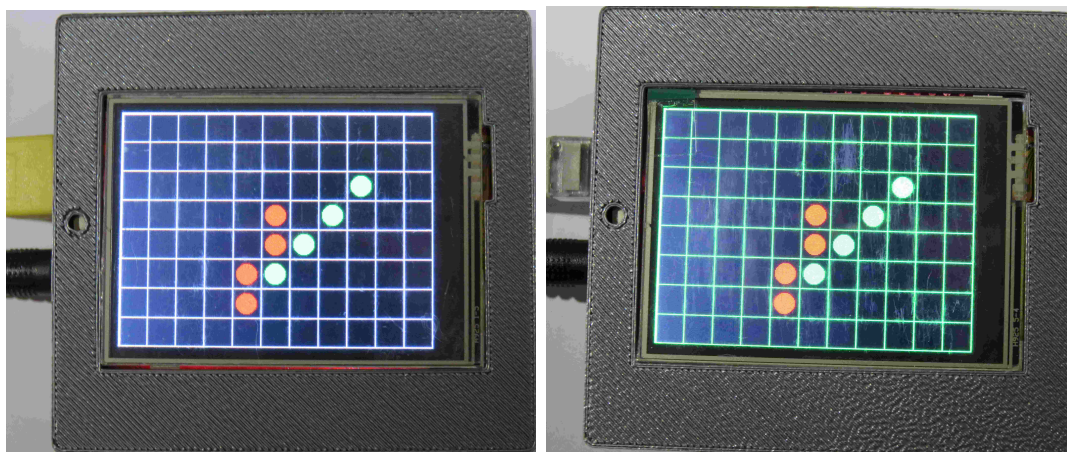
Jednotlivé čtverce mají pro tento typ displeje fyzický rozměr přibližně 0,5x0,5 mm. Tato velikost byla zvolena proto, že je možné na takto malý displej umístit dostatečně velké pole (11x8 čtverců) a zároveň je možné jej bez problémů ovládat stylusem a případně i prstem.



Obrázek 14: Obrazovka klienta po započaté hře (a) čekající hráč, (b) hrající hráč

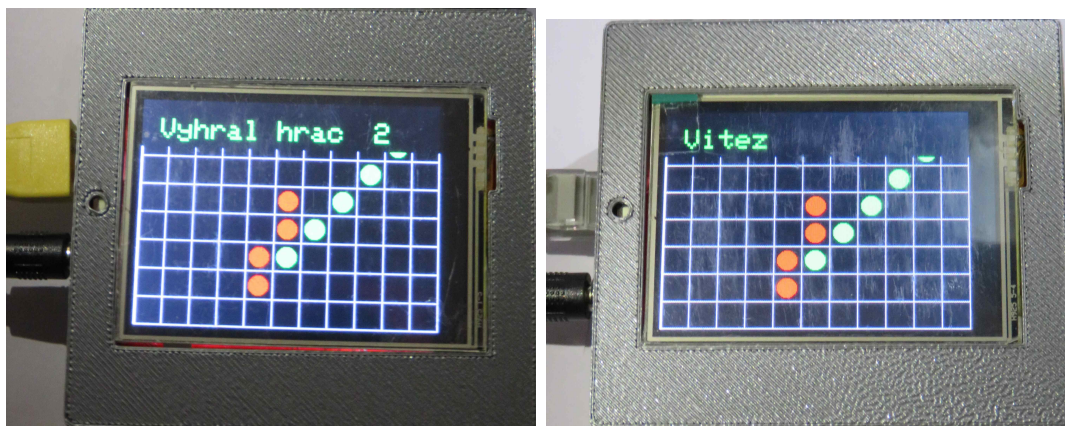
5. Během hry: pokud je mřížka šedá, hraje jiný hráč; pokud má mřížka nějakou barvu (barva se shoduje s barvou hráče), je na tahu daný hráč. Ten má za úkol stiskem libovolného volného čtverečku umístit svůj žeton.

Když uživatel stiskne čtverec, dojde k jeho vyplnění příslušným žetonem (barevným kruhem), mříž zezedne a je na řadě další hráč. Rozehraná hra je vidět na obrázku 15.



Obrázek 15: Rozehraná hra dvou hráčů

6. Hráči se střídají do té doby než: jsou všechna pole vyplněna (ukončeno remízou) nebo některý hráč spojil požadovaný počet žetonů (v základním nastavení pět žetonů). O výsledku hry jsou hráči informováni hláškou na displeji (obrázek 16). Tato hláška po deseti sekundách zmizí (nastaveno v proměnné `clientMessageLast` v milisekundách) a hra přechází do fáze 3.



Obrázek 16: Zobrazená hláška na displeji (a) výherce a (b) ostatních hráčů, barva textu se shoduje s barvou vítězného hráče



4.2 Ovládání serveru

Server lze ovládat dvěma způsoby. Prvním z nich je ovládání pomocí dvou tlačítek.

Pokud je stisknuto červené tlačítko, dojde k přerušení aktuálně běžící hry (dosavadní stav hry je resetován).

Pokud je stisknuto zelené tlačítko a neběží hra (indikační LED svítí modře) - stiskem tlačítka dojde ke spuštění hry (s ověřením zda je k dispozici dostatek hráčů). Pokud hra již běží (indikační LED svítí zeleně) stiskem zeleného tlačítka dojde k posunutí tahu na dalšího hráče.

Druhým způsobem ovládání je posílání příkazů přes sériovou linku. V tomto případě je nutné serverem připojit k počítači pomocí microUSB kabelu (konektor z přední strany serveru). K zobrazení dat lze použít nástroj *Serial monitor* přímo v Arduino IDE nebo například sériový terminál *RealTerm*². Nastavení je následující: rychlost = 9600 baudů, Data bits = 8, Stop bits = 1, Flow control = none. Každý příkaz musí být zakončen novým řádkem (LF). Seznam příkazů je v tabulce 3.









Informování uživatele o stavu serveru je realizováno pomocí barevné svítivé diody. Význam jednotlivých stavů je v tabulce 4. Symboly:  svítivá dioda svítí,  svítivá dioda bliká.

²Domovská stránka: <https://realterm.sourceforge.io/>

Tabulka 3: Seznam příkazů dostupných pro server

Příkaz	Význam
help	Vypíše nápovědu (dostupné příkazy)
info	Vypíše informace o serveru (HW, verze SW, apod.)
players	Zobrazí čísla a IP adresy připojených hráčů
kick 'x'	Odpojí hráče číslo x
nextP	Přepne na dalšího hráče
start	Spustí hru (ekvivalent zeleného tlačítka)
reset	Přeruší a resetuje hru (ekvivalent červeného tlačítka)

Tabulka 4: Význam stavů svítivé diody na serveru

Stav svítivé diody	Význam
	server je vypnutý/nemá napájení
	server je připraven
 3x	nový klient připojen
 3x	klient se odpojil/byl odpojen
	aktuálně běží hra
	hra ukončena (výhra/remíza)
 3x	chyba (nedostatek hráčů pro hru)
 stále	chyba sítě (připojení kabelu)

5 Závěr

V rámci této bakalářské práce se podařilo sestrojít zařízení, která zábavnou formou demonstruje možnosti komunikace nízkoeenergetický jednočipových počítačů (Arduin) po síti a tím i prezentovat možnosti komunikace v IoT sítí, kdy mezi jednotlivými zařízeními není potřeba přenášet velké množství dat.

Během realizace projektu se vyskytlo několik problémů, některé z nich bylo možné úplně eliminovat, jiné vedly k určitým kompromisům. Všechny tyto problémy jsou pak zmíněny v této práci včetně zvoleného řešení. I když zařízení bylo během vývoje pravidelně testováno, nevylučuje se, že by mohlo obsahovat chyby. Opravené kódy pak budou zveřejňovány na autorově [GitHubu](#)³, kde má celý projekt od počátku vytvořenou stránku. Kromě všech zdrojových kódů, na ní lze nalézt i tuto práci a další materiály, například soubory krabiček (stl soubory i soubory pro program Autodesk Inventor). Krom oprav chyb zde budou umístěna i případná vylepšení.

³Adresa: https://github.com/janzavorka/BP_PROJ

Reference

- [1] Internet of Things (IoT) Protocols and Connectivity Options: An Overview. *SaM Solutions* [online]. SaM Solutions, c1993-2019, 22.8.2019 [cit. 2019-05-17]. Dostupné z: <https://www.sam-solutions.com/blog/internet-of-things-iot-protocols-and-connectivity-options-an-overview/>
- [2] W5100 Datasheet. *WIZnet* [online]. c2009-2011, 8.1.2016 [cit. 2019-05-05]. Dostupné z: https://www.wiznet.io/wp-content/uploads/wiznethome/Chip/W5100/Document/W5100_Datasheet_v1.2.7.pdf
- [3] IAM_MAKER_LEO. How to Make Unattackable Secure Arduino IoT Device. *Instructables circuits* [online]. Autodesk, c2019 [cit. 2019-05-17]. Dostupné z: <https://www.instructables.com/id/How-to-make-unattackable-secure-arduino-IoT-device/>
- [4] MEHEDI, Hasan. Top 15 Standard IoT Protocols That You Must Know About. *UbuntuPIT* [online]. c2017-2019 [cit. 2019-05-17]. Dostupné z: <https://www.ubuntupit.com/top-15-standard-iot-protocols-that-you-must-know-about/>
- [5] Fritzing Parts: Arduino_Ethernet. In: *Paulvollmer* [online]. c2018, 2013 [cit. 2019-05-05]. Dostupné z: https://paulvollmer.net/FritzingParts/parts/Arduino_Ethernet.html
- [6] Using the SD library to create and remove files on a SD card. In: *Arduino* [online]. c2019, 2015/08/18 [cit. 2019-05-05]. Dostupné z: <https://www.arduino.cc/en/tutorial/files>
- [7] Lan Switch Icon #83279. In: *Free Icons Library* [online]. c2018-2019 [cit. 2019-05-05]. Dostupné z: <http://chittagongit.com/icon/lan-switch-icon-25.html>
- [8] Fritzing Parts: Arduino_DUE_V02b. In: *Paulvollmer* [online]. c2018, 2013 [cit. 2019-05-05]. Dostupné z: https://paulvollmer.net/FritzingParts/parts/Arduino_DUE_V02b.html
- [9] Arduino store: ARDUINO DUE. *Arduino* [online]. c2019 [cit. 2019-05-05]. Dostupné z: <https://store.arduino.cc/duo>
- [10] Arduino: Ethernet library. *Arduino* [online]. c2019 [cit. 2018-12-18]. Dostupné z: <https://www.arduino.cc/en/Reference/Ethernet>
- [11] PRENTICEDAVID. MCFRIEND_kbv library. In: *Github* [online]. c2019 [cit. 2019-02-05]. Dostupné z: https://github.com/prenticedavid/MCFRIEND_kbv
- [12] BURGESS, Phillip. Adafruit GFX Graphics Library: Overview. *Adafruit* [online]. 29.6.2012 [cit. 2019-05-08]. Dostupné z: <https://learn.adafruit.com/adafruit-gfx-graphics-library/overview>
- [13] ADAFRUIT. Adafruit_TouchScreen library. *Github* [online]. c2019 [cit. 2019-02-05]. Dostupné z: https://github.com/adafruit/Adafruit_TouchScreen

-
- [14] ROMANI, Marcello. SimpleTimer Library for Arduino. *Arduino* [online]. c2019 [cit. 2019-05-08]. Dostupné z: <https://playground.arduino.cc/Code/SimpleTimer/>
- [15] IETF [INTERNET ENGINEERING TASK FORCE], IANA [INTERNET ASSIGNED NUMBERS AUTHORITY]. *Service Name and Port Number Procedures: rfc6335, BCP165*. 2011, 33 s. ISSN: 2070-1721. Dostupné také z: <https://tools.ietf.org/html/rfc6335>
- [16] MALÝ, Martin. Arduino: webový server i klient do ruky. *Root.cz* [online]. 27. 7. 2010 [cit. 2019-02-06]. Dostupné z: <https://www.root.cz/clanky/arduino-webovy-server-i-klient-do-ruky/>
- [17] Arduino: unsigned char. *Arduino* [online]. c2019 [cit. 2019-05-16]. Dostupné z: <https://www.arduino.cc/reference/en/language/variables/data-types/unsignedchar/>
- [18] Wiznet W5100 Ethernet Shield. *HOBBYIST.CO.NZ* [online]. [cit. 2019-05-16]. Dostupné z: <https://www.hobbyist.co.nz/?q=ethernet-shield-w5100>
- [19] MARCO. *Arduino Wiznet ethernet shield proper reset* [online]. 22.12.2010 [cit. 2019-05-16]. Dostupné z: <https://marco.guardigli.it/2010/11/arduino-wiznet-ethernet-shield-proper.html>

Seznam použitého softwaru

1. [T_EXmaker](#), [T_EXLive](#)
2. [Tables Generator](#)
3. [Citace PRO](#)
4. [Photopea](#)
5. [IrfanView](#)
6. [Autodesk Inventor Professional 2019 Student Edition](#)
7. [PrusaControl](#)
8. [Arduino IDE](#)
9. [Atom IDE](#)
10. [RealTerm](#)
11. [EasyEDA](#)
12. Linux Mint 19.1 Cinnamon 64-bit
13. Windows 10 Home 64-bit

A Soubor s daty

Součástí této práce je i soubor „zavorja4_BP_priloha.zip“ s zdrojovými kódy a dalšími daty. Soubor má následující strukturu:

