GYMNASIUM JANA KEPLERA

Parléřova 2/118, 169 00 Praha 6

Univerzální sériový programátor

Maturitní práce

Autor: Petr Šícho

Třída: R8.A

Školní rok: 2021/2022

Předmět: Informatika

Vedoucí práce: Bc. Emil Miler

Praha, 25.3.2022

GYMNASIUM JANA KEPLERA
Kabinet informatiky

ZADÁNÍ MATURITNÍ PRÁCE

Student:	**Petr Šícho**
Třída:	**R8.A**
Školní rok:	**2021/2022**
Platnost zadání:	**30. 9. 2022**
Vedoucí práce:	**Emil Miller**
Název práce:	**Univerzální sériový programátor**

Pokyny pro vypracování:

Cílem práce je vymyslet a vytvořit univerzální programátor minimálně pro nejznámnější čipy AVR, tedy řady Attiny a Atmega. Univerzální programátor má ulehčit nahrávání do zmíněných čípů, bez nutnosti zapojovat cokoliv na nepájivém poli. Programátor by měl detekovat jaký je do něj vložený čip a umožnit do něj nahrávat z počítače. Za dodatečné, vedlejší, funkce by se dalo například považovat možnost nahrávat do čipů přes pin header, tedy bez vyndavání čipu z postaveného obvodu. Výsledkem projektu by minimálně měl být funkční prototyp programátoru, který je použitelný pro specifikovaný účel.

Doporučená literatura:

Zde bude *vedoucím práce* doplněna literatura, ze které je doporučeno při práci na tématu vycházet.

URL repozitáře:

`https://github.com/Xmen097/USP`

student *vedoucí práce*

V Praze dne 19. 10. 2021

Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů. Nemám žádné námitky proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Praze dne 19. března 2022 Petr Šícho

Poděkování

Chtěl bych poděkovat především open hardware komunitě, specificky pak Arduino komunitě, na jejíž fórem jsem našel značné množství informací. Dále děkuji Bc. Emilovi Milerovi za pomoc při vedení této maturitní práce.

Abstrakt

Tato maturitní práce si klade za cíl zjednodušit proces programování a nahrávání do mikrokontrolérů v pozdře DIP. Univerzální sériový programátor (USP), který v rámci této práce vznikl, je zařízení, které je schopné automaticky detekovat rozložení vývodů čipu vloženého do ZIF patice. Na základě toho rozpozná o který čip se jedná a uzpůsobí pro něj nahrávací obvod. Teoreticky by měl USP umět programovat všechny čipy pracující na 5 voltech a podporovat dostatečně pomalé bit-banging programovací protokoly, včetně těch, které vyžadují vysoké napětí (12V) pro resetování. Na softwarové úrovni aktuálně USP pracuje na protokolu Arduino ISP.

Klíčová slova

programátor, čip, AVR, nahrávač, USP

Abstract

Abstract.

Keywords

programmer, chip, AVR, uploader, USP

Obsah

1. Teoretická část

Existuje nepřeberné množství různých čipů a s nimi podobné množství různých rozložení vývodů, které nejsou jednotné ani u jedné velikosti a jednoho výrobce. Pro každé rozložení je nutné mít externí nahrávací obvod. Cílem práce je sjednotit tyto nahrávací obvody do jednoho univerzálního, který se přizpůsobí vloženému čipu. Fungování programátoru lze rozdělit na dvě části - nejdřív je nutné detekovat čip a poté je potřeba do něj nahrávat.

Pro univerzální programátor není možné použít žádný dedikovaný hardware, který by zajišťoval nahrávání. Naopak se nám hodí použít univerzálních vstupních / výstupních pinů neboli GPIO[1] pinů. I přes snahu o maximální univerzálnost není reálné docílit podpory úplně všech existujících čipů. Jako minimální požadovanou funkcionalitu jsem tedy bral podporu čipů AVR od výrobce Atmel, konkrétně pak řady ATmegy a ATtiny, které zahrnují čipy, jež pohání známé desky Arduino. Přesto doufám, že vlastnosti těchto čipů, na kterých je detekce a nahrávání postaveno, budou do velké míry přenositelné i na jiné rodiny či výrobce čipů.

1.1 Detekce

Základem detekce je poznat jak velký čip je - tedy kolik má vývodů. Jen díky tomu se nám radikálně sníží počet možných kandidátů. Při zahrnutí jen několika modelů, jednoho pro každou velikost, bychom tak měli prakticky hotovo. Bylo by však nutné arbitrárně stanovit tyto podporované čipy, přičemž při vložení jiného, ať třeba ze stejné rodiny, který má prohozené piny GND a VCC, by došlo pravděpodobně k jeho zničení. Naší ambicí tedy bude zjistit o čipu co nejvíce informací, díky nimž budeme schopni rozeznávat mezi co největším počtem různých čipů.

Při detekci se musíme dávat pozor, abychom náhodou čip nepoškodili. Ideálně bychom se chtěli držet v hodnotách, které povoluje datasheet. Problém je, že to, o co se pokoušíme, není standardní zacházení s čipy a v datasheetu tak nenalezneme konkrétní povolené hodnoty a situace, kterým můžeme čip bezpečně vystavit. Můžeme pomocí něj alespoň odhadnout některé základní principy, kterých když se budeme držet, tak minimalizovali poškození čipu. Konkrétně se jedná o:

- Nevystavovat žádný pin napětí nižšímu než $-0.5V$
- Nevystavovat žádný pin, kromě RESETu, napětí vyššímu než $V_{cc} + 0.5V$
- Nevystavovat RESET napětí vyššímu než $13V$
- Nenechat mezi žádný dvěma piny nezapojeného čipu téct nezanedbatelně velký proud po nezanedbatelně dlouhou dobu [2]

Při splnění těchto podmínek by měl být čip dostatečně chráněn obvody, které běžně slouží k ochraně před elektrostatickým výbojem (ESD).

[1] z anglického General-purpose input/output, tedy pin, který se umí chovat buď jako digitální vstup, nebo výstup, přičemž mezi těmito módy je možné přepínat.

[2] Pro naše účely budeme považovat za zanedbatelný proud řádu mikroampér a čas mikrosekund.

1.1.1 Detekce velikosti a polohy

Pro detekci velikosti čipu můžeme využít toho, že každá reálná součástka, kromě svých primárních vlastností, vykazuje také elektrickou kapacitu. To bude jistě platit i pro spoje a součástky uvnitř integrovaných obvodů. Elektrická kapacita pinů jednoduchých CMOS[3] čipů[5, 6, 11] se pohybuje v jednotách pikofaradů, což není mnoho. Není mi znám žádný způsob přímého měření elektrické kapacity, budeme tedy muset provést měření nepřímé a dostatečně přesné. Jednou z možností je kapacitor pomalu nabíjet nebo vybíjet a měřit čas, který k tomu budeme potřebovat. Zkusíme odhadnout jak přesné takové měření bude s pomocí rovnice definující energii elektrického kondenzátoru

$$E = \frac{1}{2} C U^2 \tag{1.1}$$

Budeme-li kapacitor nabíjet napětím U a proudem I po dobu t, můžeme psát

$$W_{max} = U \cdot I \cdot t \tag{1.2}$$

Jedná se o horní odhad vykonané práce, která počítá s nulovým odporem a nulovým elektrickým potenciálem kapacitoru. Reálná energie předaná kapacitoru bude nižší, což nám však nevadí. Úpravou těchto dvou vztahů můžeme vyvodit úměrnost

$$C \simeq t/R \tag{1.3}$$

Zjistili jsme, že kapacita je přímo úměrná času, po který jsme kapacitor nabíjeli, a nepřímo úměrná odporu, přes který jsme nabíjeli.[4] Čas budeme schopni měřit velice přesně, řádově v mikrosekundnách (10^{-6}s). Řekněme, že pin čipu má kapacitu $10pF = 10^{-11}F$. Dosazením do vztahu (1.3) zjistíme, že k jejímu změření budeme potřebovat nabíjecí odpor o velikosti 100kΩ. To nebude problém, měření vypadá proveditelně.

Díky schopnosti změřit kapacitu na každém z pinů socketu zjistíme nejenom jak velký čip je, ale zároveň i kde v socketu je vložen. Není tedy nutné nijak specifikovat na jakou pozici v socketu má být čip pro nahrávání vložen.

1.1.2 Detekce speciálních pinů

Každý mikrokontrolér má kromě vstupních / výstupních pinů i speciální piny, které plní nějakou zvláštní funkci. Například je to RESET pin, napájecí piny (GND a VCC) nebo piny hodin (XTAL). Tyto piny by mohly vykazovat rozdílné elektrické vlastnosti a možná i rozdílnou kapacitu. Největší naději vkládám do napájecích pinů, které jsou vnitřně připojené k všemožným součástem mikrokontroléru a očekávám u nich vyšší kapacitu. Při dostatečně přesném měření kapacity bychom ji mohli detekovat a určit tak napájecí piny.

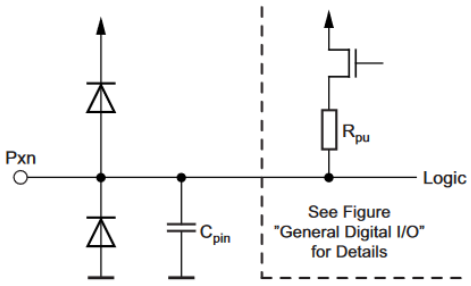
1.1.3 Rozpoznání VCC od GND

Při špatném rozpoznání vloženého mikrokontroléru může největší poškození způsobit převrácení pinů GND a VCC. Abychom se tohoto rizika vyvarovali, bylo by dobré rozpoznávat s jistotou VCC od GND. Tím zároveň obecně snížíme šanci na špatné rozpoznání čipu.

[3] Complementary metal-oxide semiconductor; technologie výroby integrovaných obvodů

[4] Ke stejnému výsledku bychom došli i použitím rozměrové analýzy.

Využijeme ochranných diod, které jsou připojeny ke každému IO pinu mikrokontroléru, jak ukazuje obrázek 1.1. Povšimněme si, že dioda spojující pin s GND směřuje opačně, než dioda spojující pin s VCC. Díky tomu bychom měli být schopni rozpoznat, zda je určitý pin GND či VCC.



Obrázek 1.1: Schéma vnitřního zapojení IO pinu mikrokontroléru.[3, str. 58]

Pin GND můžeme detekovat následovně. Budeme znát úbytek napětí na diodě mezi pinem a GND U_d. Přivedeme-li na pin GND napětí U převyšující U_d, měli bychom na všech IO pinech detekovat napětí $U - U_d$. Z bezpečnostních důvodů by napětí U by mělo být co nejmenší možné, ideálně $U < 0.5V$. Napětí přivedeme přes dostatečně velký odpor tak, aby mikrokontrolérem a diodou tekl co nejmenší proud. S malým procházejícím proudem budeme mít také menší úbytek napětí na diodě (menší, než běžně udávaných $\sim 0.7V$ a snad i menší než $0.5V$)

Obdobným způsobem je možné detekovat pin VCC. Stačí na libovolný (nebo lépe postupně na všechny) IO pin přivést napětí U a na pinu VCC by se nám vždy mělo ukázat napětí $U - U_d$.

1.1.4 Závěr

S touto výbavou bychom měli být schopni dostatečně detailně rozpoznat čipy ATmega a ATtiny. Díky rozpoznáním GND a VCC budeme schopni čip bezpečně napájet. Nebude pak problém, kdybychom chtěli rozšířit podporované čipy a vyskytly by se nám dva čipy, které mají na stejných místech GND a VCC, ale na různých místech piny potřebné pro nahrávání. V této situaci bychom prostě postupně vyzkoušeli různá rozložení nahrávacích pinů, až bychom našli to správné.

1.2 Nahrávání

Po detekování čipu nám už zbývá pouze do něj nahrát. Vzhledem k tomu, že přesně nevíme jaké rozložení pinů bude čip mít, musíme použít tzv. bit-banging[5]. Díky tomu se nemusíme při stavbě hardwaru nahrávače omezovat na určité čipy, ale budeme schopni teoreticky obsluhovat všechny čipy, které nevyžadují příliš velké komunikační rychlosti a existuje pro ně bit-banging implementace nahrávání. Nebude tak ani obtížné obměňovat nahrávací protokoly pro různé čipy nebo situace.

Komplikací může být, že některé nahrávací protokoly vyžadují použití vyššího napětí. U AVR čipů jde konkrétně o protokoly HVSP (high voltage serial programming) a HVPP (high voltage parallel programming), které se hodí při určitém nastavení mikrokontroléru, kdy není možné standardní

[5]Softwarová implementace určitého protokolu, která přímo generuje protokolem specifikovaný signál, bez použití specializovaného hardwaru

programování.[6] Vysokým napětím je v obou případech myšleno 12 voltů.[9] Vzhledem k tomu, že ve valné většině případů lze programovat mikrokontrolér bez použití vysokého napětí, beru jeho dostupnost spíše jako bonus než nutný požadavek.

1.3 Shrnutí požadovaných funkcí

V této sekci si shrneme funkce, které budeme potřebovat pro úspěšnou detekci a nahrávání do čipů. Těmito funkcemi by měl disponovat každý z pinů socketu, neboť nevíme kde se zrovna čip ocitne. Některé funkce budou komplexní (přečti kapacitu připojenou k pinu), u jiných se bude jednat o nějaký elementární stav (připojení k zemi/5V).

1. **5V s proudem >20 mA** je potřeba pro napájení čipu. Proud 20 mA by měl stačit pro napájení malých AVR čipů.[4] Větší čipy mají sice větší spotřebu než 20 mA, ale mají také několik separátních napájecích pinů, mezi které se proud rozloží.[3]
2. **0V (GND) s proudem >20 mA** je potřeba též pro napájení čipu.
3. **5V s libovolně malým proudem** reprezentuje logický stav 1 při komunikaci, odebíraný proud bude zanedbatelný.
4. **0V (GND) s libovolně malým proudem** reprezentuje logický stav 0 při komunikaci.
5. **Čtení digitální hodnoty** pro čtení odpovědi čipu při nahrávání; standardní funkce GPIO pinů.
6. **Přečtení kapacity** pro detekci čipu.
7. **Napětí přibližně 0.4-0.5V** je potřeba pro rozpoznání GND a VCC, viz sekce 1.1.3.
8. **Přečtení napětí na pinu** je také potřeba pro rozpoznání GND a VCC.
9. **12V** bude potřeba pokud budeme chtít high-voltage programování.
10. **Nepřipojeno** k ničemu. Piny, které nevyužíváme k jiné funkci, by se měly chovat tak, jako kdyby nebyly k ničemu připojené.

[6] Jde např. o situace kdy je vypnuté resetování pomocí RESET pinu, protože ho chceme použít na jinou funkci.

2. Implementace

Implementace projektu měla dvě fáze. Nejprve bylo potřeba navrhnout hardwarovou část - vymyslet a vyzkoušet jednotlivé části obvodu tak, aby plnily požadované funkce. Poté bylo potřeba všechny spojit do jednoho celku, který se vyrobí jako plošný spoj. Po osazení plošného spoje součástkami a otestování základní funkčnosti přišel čas na software. Zde jsem začal implementací low-level komunikace s jednotlivými součástkami, pak přišly na řadu high-level funkce detekce.

Pojďme si nyní jednotlivé fáze rozebrat detailněji.

2.1 Hardware

Převážná část práce na hardwaru probíhala s pomocí nepájivého pole, na kterém jsem postupně testoval jednotlivé ideje uvedené v teoretické části práce. V následujících sekcích si rozebereme jaké součástky byly použity a jak se pomocí nich implementují požadované funkce ze sekce 1.3.

2.1.1 Vkládání čipu

Jako patici, do které se bude vkládat čip k programování, jsem zvolil ZIF (zero insertion force) socket. Jak již jméno napovídá, pro vložení čipu do patice není nutné užití síly. Stačí čip položit na patici a pomocí páčky na straně patice čip zajistit, čímž vznikne spolehlivý kontakt s piny čipu. Nehrozí tak ohnutí nožiček čipu a jeho vyndání a zandání je velmi rychlé. Nevýhodou ZIF socketu je o trochu vyšší pořizovací cena a větší velikost, ale i přesto se v našem případě určitě vyplatí. Konkrétně jsem zvolil patici se 40 piny (dvaceti na každé straně), která by měla postačovat pro naprostou většinu DIP[1] pouzder. Patice podporuje vložení jak úzké (0.3”), tak široké (0.6”) varianty DIP.

2.1.2 Řídící mikroprocesor

Je potřeba zvolit mikroprocesor, který bude ovládat celý nahrávač. Volil jsem mezi mikrokontroléry AVR, s kterými jsem nejvíce obeznámen a které také umí jednoduše komunikovat s Arduino IDE. Zásadní otázkou je kolik budeme potřebovat GPIO pinů k ovládání USP. Určitě budeme potřebovat jeden pin pro každý pin ZIF patice, tedy minimálně 40 pinů. Další desítky pinů budou potřeba pro ovládaní komponent a součástek, o kterých se budu detailněji zmiňovat v dalších sekcích. Teoreticky by bylo možné přidat další piny pomocí shift registrerů či IO expanderů. Tím by však utrpěla rychlost komunikace a vzrostla by komplexita desky a tím i prostor pro chyby. Rozhodl jsem se tedy pro jeden co největší čip, mikrokontrolér ATmega2560 v pozdře TQFP100, který má 84 GPIO pinů, což by mělo stačit.[2]

Nejprve jsem plánoval, že bych využil dostupných schémat Arduino desek a v upravené podobě je přímo zakomponoval do desky nahrávače. To by umožnilo využít řídící čip bez omezení. Na druhou stranu by tím extrémně vzrostla komplexita desky a také celková cena. Nakonec jsem se

[1] Dual in-line package; pouzdro s THT piny s roztečí 2.54mm

tedy rozhodl programátor vyvynout jako shield (nasazovací desku) na Arduino Mega s mikroprocesorem ATmega2560. Nevýhodou tohoto postupu je snad jen to, že Arduino mega nám dává k dispozici pouze 70 z 84 GPIO pinů, ale i to by mělo pro projekt stačit.[1]

2.1.3 Základní funkcionalita pinů

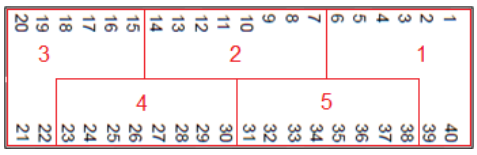
Nyní přichází na řadu implementace požadovaných funkcí popsaných v sekci 1.3. Jenom tím, že ke každému pinu ZIF socketu bude připojený GPIO pin, zajistíme funkce 3, 4, 5, které GPIO piny zvládají nativně. Funkce 1 a 2, které vyžadují minimální proud 20 mA, také s trochou opatrnosti zvládneme. Digitální piny ATmegy2560 dokáží dodávat maximální proud 40 mA [2, str. 355]. Musíme tedy jen dát pozor na to, aby další součástky, které budou zapojeny mezi GPIO pinem a pinem ZIF socketu neměly příliš velký odpor nebo nevyžadovaly více omezený protékající proud.

Desátý požadavek vyžaduje trochu zamyšlení. Pin určitě nebude možné fyzicky odpojit, to bychom museli provést manuálně. Budeme tedy alespoň chtít minimalizovat proud, který může skrz tento 'odpojený'pin téct. Když nastavíme GPIO pin jako vstupní, bude ve stavu s tzv. vysokou impedancí, tedy budou jím téct velmi malý proud. Datasheet specifikuje maximální proud jako $1\mu A$, typický proud lze očekávám výrazně menší. Odpojení si navíc můžeme pojistit použitím analogového přepínače, u kterého je udáván maximální proud $100nA$ a typický $0.05nA$.[11, str. 3]

2.1.4 Zapojení multiplexerů

Implementaci funkcí 6, 7, 8 a 9 si zjednodušíme pozorováním, že v jednu chvíli bude daná funkce potřebná pouze na jednom pinu. Nemusíme ji tedy implementovat 40x, ale můžeme použít multiplexery k spojení určitého pinu s něčím, co nám funkci zajistí. Napětí na pinu budeme číst pomocí ADC[2] převodníku. Atmega2560 disponuje jedním ADC převodníkem, který je ale vnitřně multiplexován na 16 GPIO pinů. My pomocí dalších multiplexerů rozšíříme tuto funkcionalitu na všech 40 pinů.

Multiplexery se standardně vyrábí s 2^n kanály, které se ovládají pomocí n pinů. Zvolil jsem 8 kanálové multiplexery[5], tedy s 3 ovládacími piny. Pro obsluhu 40 pinů jich bude potřeba celkem 5. Multiplexery jsou připojeny k pinům ZIF socketu způsobem znázorněným na obrázku 2.1.



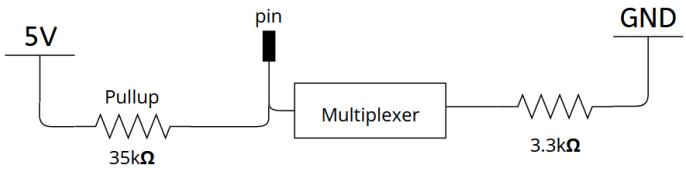
Obrázek 2.1: Schéma zapojení mulitplexerů k pinům ZIF patice.

Díky tomuto způsobu zapojení budeme schopni odhalit závadu na multiplexeru při rozpoznávání čipu, neboť i nejmenší mikrokontrolér, který má pouzdro DIP8 se 4 piny na každé straně, bude mít vždycky alespoň dva piny v různých multiplexerech. Nestane se tedy, že by kvůli závadě na jednom multiplexeru (ta může být způsobená i třeba dotekem uživatele, kterým se zvýší kapacita) byl čip chybně rozpoznán.

[2]Analogově digitální převodník; zařízení, které převede analogový signál (napětí) v určitém rozsahu na digitální tvar s určitým počtem bitů, v našem případě se jedná o převodník 10-bitový

2.1.5 Napěťový dělič

Napětí 0.4-0.5V, jak si žádá 7. požadavek, vytvoříme pomocí napěťového děliče. Zde se nám budou hodit již výše zmíněné multiplexery. Dělič napětí bude totiž právě mezi GPIO pinem a multiplexovaným pinem. Na každém GPIO pinu je možné zapnout tzv. pullup rezistor, který pin připojí přes odpor s přibližnou velikostí $30 - 40k\Omega$[3] k 5 voltům. Kýžené napětí přibližně půl voltu vytvoříme dalším rezistorem s velikostí $3.3k\Omega$[4] k zemi. Zapojení děliče napětí na pinu je vidět na obrázku 2.2



Obrázek 2.2: Schéma děliče napětí.

2.1.6 Měření kapacity

Kapacita lze měřit, jak jsme vyvodili v sekci 1.1.1, pomocí pomalého nabíjení - tedy nabíjení přes dostatečně velký rezistor. Vypočítali jsme, že by se nám hodil rezistor s odporem $100k\Omega$. Připojovat takový odpor k pinům by si vyžádalo dalších multiplexerů nebo switchů. Můžeme ale využít stejných pullup rezistorů, které využíváme pro napěťový dělič. Ty jsou o něco menší a mohlo by se stát, že kapacitor nabijeme příliš rychle. Můžeme ale chytře využít ADC převodníku, který máme multiplexovaný k pinům. Původně jsme chtěli měřit čas, za který nabijeme plně kapacitor, což bychom zjišťovali pomocí digitálního vstupu GPIO pinů. Můžeme to ale provést i opačně. Kapacitor budeme nabíjet nějaký konstantní čas, a poté pomocí ADC převodníku zjistíme napětí na kapacitoru. Čím vyšší napětí bude, tím více jsme kapacitor nabili a tím je tedy kapacitor menší. Opačně by šlo i kapacitor plně nabít a poté ho nechat samovolně vybíjet. Počkáme konstantní čas a opět pomocí ADC převodníku zjistit jak moc se vybil.

Při testování se ukázalo, že pro měření kapacity se nám velmi hodí i dělič napětí, více o tom však až v sekci 2.2.3

2.1.7 Napětí 12 voltů

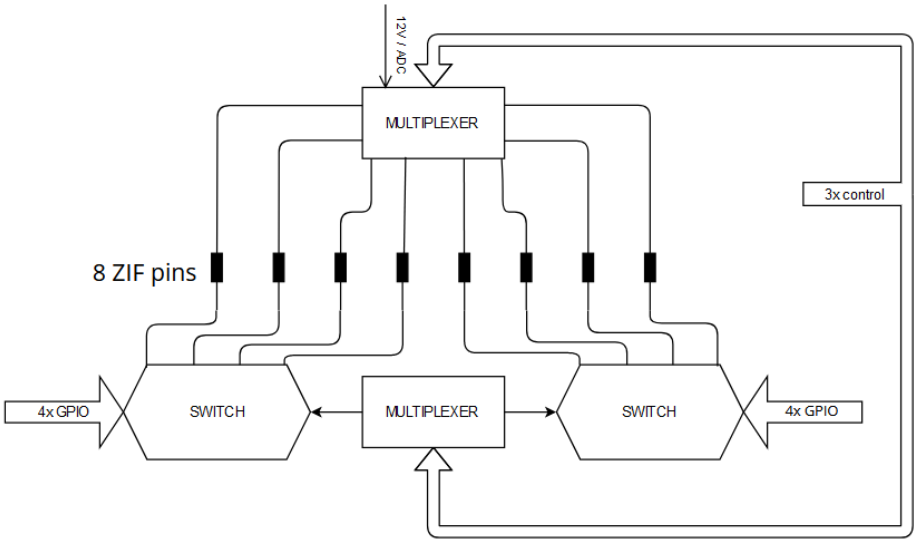
Poslední funkcionalitou, kterou nám zbývá implementovat je dostupnost napětí 12 voltů. To můžeme na pin pouštět pomocí stejných multiplexerů, které využíváme k připojení ADC převodníku. Stačí pomocí switche CD4066B [6] přepínat zdroj multiplexeru mezi 12 volty a ADC převodníkem. Zapojení tohoto obvodu pro multiplexer 1 a 2 je vyobrazeno v příloze A.2. Od použitého switche nepožadujeme žádné zvýšené nároky ohledně proudu, pro HVSP i HVPP nám stačí libovolně malý

[3] Datasheet udává, že odpor bude vždy větší než $20k\Omega$ a menší než $50k\Omega$. Vzhledem k tomu, že tyto hodnoty jsou pro teplotní rozmezí od -55°C do 125°C můžeme konstatovat, že při pokojové teplotě bude odpor přibližně mezi $30 - 40k\Omega$.

[4] Chceme dělič napětí přibližně 1:10, hodnotu odporu jsem zvolil o něco nižší, protože i samotný multiplexer bude mít určitý odpor (asi 100 ohmů).

proud. Problém je, že náš řídící mikrokontrolér, ale ani většina dobře dostupných multiplexerů, nepodporuje vyšší napětí než 5 voltů. S trochou snahy se dají najít i multiplexery podporující 12 voltů. Řídíci čip, jehož piny odolají napětí 12 voltů, ale prakticky není možné najít. Budeme proto muset piny řídíciho mikrokontroléru odpojit předtím, než připojíme 12 voltů. K tomu využijeme analogové switche MC14066B.[11] Vzhledem k tomu, že přes tyto switche budeme mimo jiné napájet čip, který budeme chtít programovat, musí podporovat průtok proudu alespoň 20mA.[5]

Každý switch má 4 kanály, které dokáže přepínat pomocí 4 ovládacích pinů. Není však možné ani potřebné ovládat každý z těchto pinů individuálně, na to bychom totiž spotřebovali dalších 40 pinů. Šlo by pomocí tranzistorů a odporů detekovat zvýšené napětí a přepnout ovládací pin switche. To by však vyžadovalo značné množství externích komponent - na každý pin jeden tranzistor a několik odporů. Lepší řešení je založené na faktu, že 12 voltů bude vždy připojené na pinu, který je multiplexován. Toho využijeme a přidáme další multiplexer, který bude mít stejně zapojené ovládací piny jako první multiplexer. Jeho funkce bude, že vypne kanál, který obsluhuje pin, ke kterému je multiplexováno. Tímto zajistíme přepínání s použitím pouze jediného pinu, který bude sloužit k potlačení této funkcionality, která by nám rozbila napětový dělič a další funkce. Zjednodušené schéma tohoto systému pro 8 pinů ZIF patice je znázorněno na obrázku 2.3. Tento obvod je na USP celkem 5 krát - pro každých 8 pinů, a to podle stejné schématu jako zapojení původních multiplexerů (obrázek 2.1). Kompletní zapojení toho obvodu je dostupné jako příloha A.1.



Obrázek 2.3: Zjednodušené schéma obvodu připojeného k 8 pinům ZIF patice

Další kompilkací spojenou s vyšším napětím je, že multiplexery a switche potřebují být ovládané tímto napětím. Nestačí tedy již připojit na ovládací pin multiplexeru/switche GPIO pin řídícího mikrokontroléru, ale je nutné použít tranzistor a pullup/pulldown[6] rezistor.

Poslední věcí, kterou musíme vyřešit, je kde získat napětí 12 voltů. Jednou možností by bylo zapojit do USP přímo toto napětí z externího zdroje, to by ale výrazně ztížilo jeho použití. Rozhodl jsem se proto přímo do USP zabudouvat step-up převodník, který nám vytvoří napětí 12V. Obvod step-up převodníku byl navržen podle datasheetu regulátoru MC34063A.[10] Celé zapojení obvodu převodníku je zobrazeno v příloze A.3.

[5] Z tohoto důvodu se použitý switch liší od switchů [6], které používáme u zdroje multiplexeru.

[6] V závislosti na tom v jakém výchozím stavu chceme, aby byl ovládací pin; ve většině případu se bude jednat o pulldown rezistor

2.1.8 Dodatečný hardware

Krom výše uvedených součástek a obvodů, které slouží k zajištění detekce nebo nahrávání, je USP vybaven také 0.91" OLED displayem a čtyřmi tlačítky. To umožní tvorbu uživatelského rozhraní pro jednodušší ovládání programátoru. OLED display byl také skvělým nástrojem pro debugování. Dále jsou pak na programátoru tři LED - modrá, zelená a červená, které slouží pro grafickou signalizaci stavu programátoru.

2.1.9 Využití pinů

Nakonec se ukázalo, že 70 pinů není až zase tolik a bylo nutné s nimy trochu šetřit. Následovat bude krátký přehled toho, k čemu bylo použito kolik pinů a případně jakým způsobem byly piny uspořeny.

- **40 pinů** je nutných k obsloužení všech pinů na ZIF patici.
- **5 pinů s podporou ADC** je připojeno k pěti multiplexerům.
- **3 piny** jsou použity pro sdílenou kontrolu multiplexerů na právé straně ZIF patice (1, 2 a 3). [7]
- **3 piny** jsou použity pro sdílenou kontrolu multiplexerů na levé straně ZIF patice (4 a 5). [7]
- **5 pinů** ovládá jestli je každý z pěti multiplexerů připojen k 12 voltům nebo ADC převodníku.
- **1 pin** ovládá zda switche odpojují piny, na které je právě multiplexováno, jak je popsáno v sekci 2.1.7.
- **2 piny** jsou použity pro display.
- **4 piny** jsou využívány tlačítky.
- **3 piny** jsou ovládají LEDky.
- **2 piny** jsou potřeba pro sériovou komunikaci s počítačem.
- **zbylé 2 piny** jsou vyvedeny do headeru.

Celkem je tedy využito všech 70 dostupných pinů.

2.1.10 Plošný spoj

Po otestování částí některých výše zmíněných obvodů na nepájivém poli bylo potřeba vytvořit návrh desky plošného spoje. Rozhodl jsem se rovnou pro 4-vrstvý plošný spoj, vzhledem k tomu, že jsem plánoval umístit co nejvíce součástek na co nejmenší plochu a vrchní vrstva je tak prakticky celá pokryta ploškami pro kontakty součástek. Prostřední dvě vrstvy jsou udělány jako tzv. plane, kde většina vrstvy zabírá jeden signál, zde konkrétně země a napájení, a slouží pro snížení rušení na desce. Hotového zapojení plošného spoje je k nahlédnutí v příloze A.4.

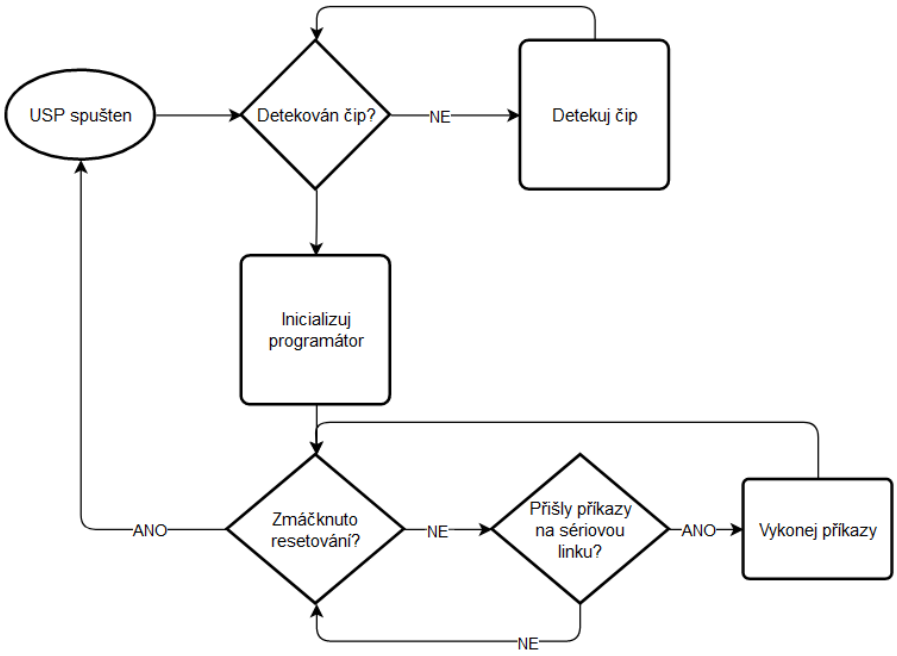
[7] Pro všechny funkce krom rozpoznání VCC a GND (sekce 1.1.3) nám stačí vždy jen jeden pin připojený k multiplexeru, ostatní multiplexery můžeme ignorovat. Pro rozpoznání VCC a GND budeme potřebovat dva multiplexery, jeden pro vytvoření malého napětí a druhý pro čtení ostatních pinů. K tomu nám ale postačí nezávisle ovládat multiplexery na jiných stranách, neboť díky rozložení multiplexerů podle obrázku 2.1 bude mít i nejmenší možný čip alespoň dva piny na různých stranách.

2.2 Software

Přestože bylo cílem projektu vytvořit co nejlepší hardwarový výrobek, je jeho nedílnou součástí i software nutný k zprovoznění tohoto hardwaru. Nicméně ne všechny funkce implementované na hardwarové úrovni jsou využity ze strany softwaru.

2.2.1 Princip fungování

Po zapnutí USP detekuje vložení čipu, který následně identifikuje a data o něm předá nahrávači (více o něm v sekci 2.2.5). Nahrávač pak vykonává příkazy, které dostane po sériové lince a programuje vložený čip, dokud uživatel nezmáčkne resetovací tlačítko. Princip fungování konkrétně zachycuje obrázek 2.4.



Obrázek 2.4: Diagram USP

2.2.2 Nízkoúrovňové funkce

Jako první bylo potřeba implementovat funkce pro ovládání hardwaru - konkrétně multiplexerů, OLED displeje a tlačítek. U tlačítek stačilo implementovat tzv. debouncing, tedy brát tlačítko jako stisknuté/puštěné až tehdy, když se jeho stav po nějakou dobu nemění.

Pro práci s displayem jsem použil knihovnu od Adafruitu pro ovladač SSD1306, kterým je display vybaven. Pro převod obrázků a grafik na display jsem využil aplikaci image2cpp.[8] Na displaji jsou implementovány převážně debugovací funkce, které se však mohou hodit i během provozu zařízení. Není totiž vyloučené, že měření, která bylo potřeba nakalibrovat, se budou v určitých situacích chovat trochu jinak a bude nutné je kalibrovat znovu. Z tohoto důvodu jsou kalibrační data uložena v paměti EEPROM, aby je bylo možné měnit i bez flashování.

Pro zprovoznění multiplexerů jsem použil návrhový vzor Composite, kdy každý z pěti multiplexerů je instancí třídy, který obstarává vnitřní mapování jeho ovládacích pinů na výstup a další nastavení. Všechny multiplexery jsou pak společně ovládány pomocí jiné třídy, která pozná, který multiplexer má použít.

2.2.3 Interface

Interface funguje jako prostředník mezi nízkoúrovňovými hardwarovými funkcemi a Detektorem, kterým se bude zabývat následující sekce. Poskytuje funkce změření kapacity či napětí na pinu, připojení napětí 0.5 voltu a další.

Měření kapacity se povedlo implementovat dostatečně přesné na rozpoznání speciálních pinů. Prakticky měření probíhá tak, že pin nabíjíme pomocí napětí 0.5 voltů po dobu 5 mikrosekund[8]. Pak nabíjet přestaneme a změříme napětí, které na pinu je. Samotné změření napětí trvá nějaký čas, během kterého se pin stihne částečně vybít. Změřené napětí je tedy zčásti dáno rychlostí nabíjení i rychlostí vybíjení. S použitím tohoto způsobu měření se jeví, že piny GND a VCC mají prakticky dvojnásobnou měrnou kapacitu než normální piny. Pin RESET je také zajímavý. U některých čipů se z pohledu kapacity chová jako kdyby tam vůbec nebyl. U jiných čipů má naopak kapacitu výrazně vyšší než normální piny. V obou situacích jsme schopni pin RESET rozpoznat jako speciální pin.

2.2.4 Detektor

Detektor implementuje vysokoúrovňovou detekci čipů. Jeho fungování je znázorněno na obrázku 2.5.

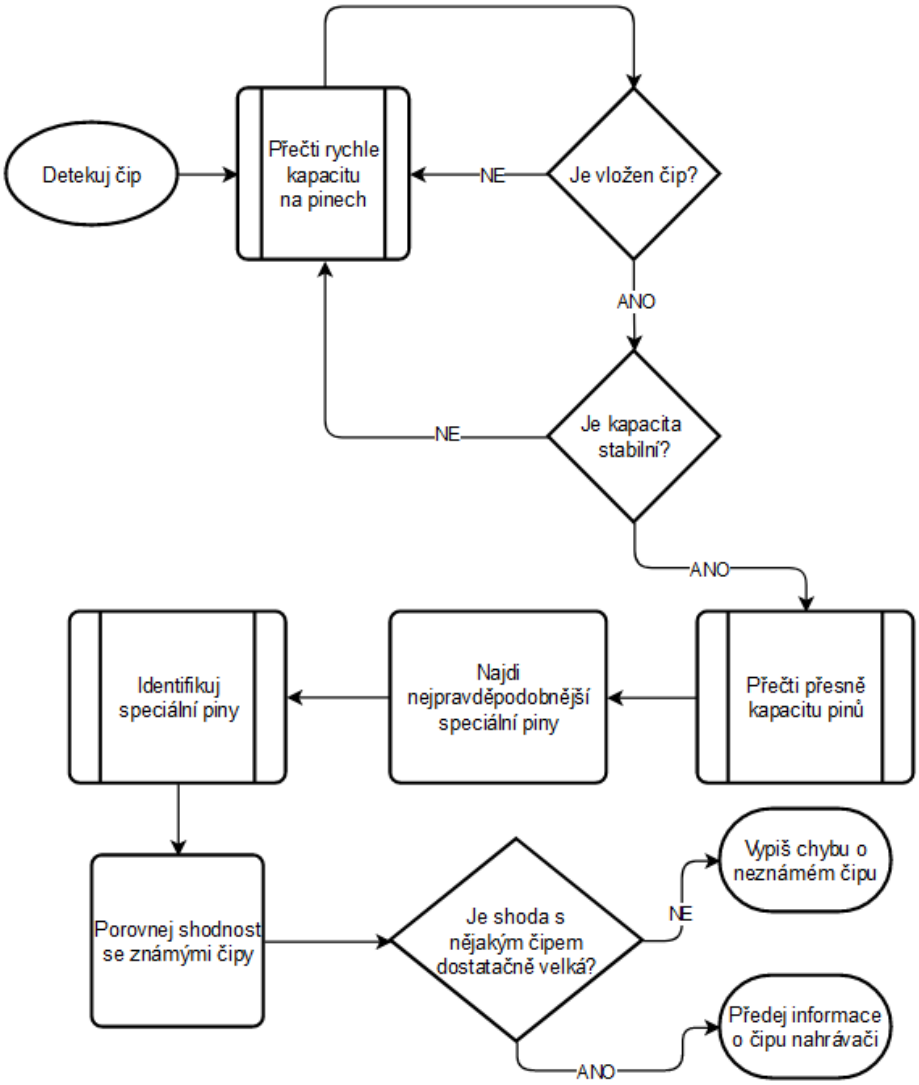
Za okomentování stojí detekce speciálních pinů. Vzhledem k zjištěním při měření kapacity popsaných v sekci 2.2.3 jsem schopni vyčíst jaké piny jsou speciální - tedy buď GND, VCC nebo RESET. Pro každý pin vyčíslíme jak moc se relativně odlišuje od průměrné kapacity pinů na daném čipu. Toto číslo je budeme přeneseně brát jako pravděpodobnost, že pin je speciálním. Pět pinů[9] s největší pravděpodobností vybereme a budeme je dále identifikovat, jak zachycuje obrázek 2.6.

Identifikace je omezena na rozpoznávání GND pinů, neboť rozpoznání VCC se ukázalo jako nespolehlivé.

Nakonec budeme iterovat přes všechny podporované čipy dané velikosti a pro obě možná otočení čipu zjistíme, jak moc jeho speciální piny sedí s piny, které jsme identifikovaly. Výsledná shoda čipu je součet pravděpodobností těchto speciálních pinů. Nejpravděpodobnější čip, který překonal stanovenou minimální shodu, je prohlášen za identifikovaný a informace o něm jsou předány dále.

[8] Tato doba při testování dávala nejlepší výsledky.

[9] Pět je kolik nejvíce může mít čip o maximální podporované velikosti speciálních pinů včetně RESETu, ale teoreticky ani čip s více speciálními piny nebude problém detekovat, neboť i pomocí pěti pinů bychom ho měli s jistotou určit



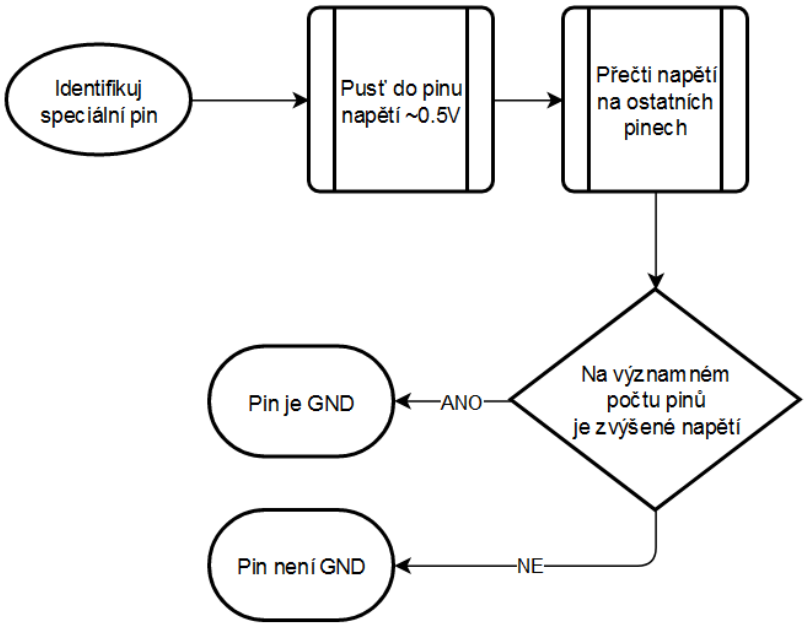
Obrázek 2.5: Diagram fungování detektoru. Dvojité obdélníky značí funkce, které interagují s hardwarem

2.2.5 Nahrávač

Díky detektoru teď přesně víme jaký čip je vložen do patice, na jaké je pozici a jak je orientovaný. Víme tedy i kde jsou umísteny piny, které potřebujeme pro nahrávání. Tyto informace již tedy stačí předat nahrávači. Protože nemá smysl znovu objevovat kolo, využil jsem již implementovaný nahrávač, Arduino ISP, který je distribuovaný s Arduino IDE. Tento nahrávač jsem zvolil mimo jiné proto, že ho lze snadno ovládat právě z Arduino IDE a pro použití USP tedy nebude nutné nic instalovat ani upravovat.

Do Arduino ISP jsem doimplementoval softwarový oscilátor pomocí hardwarového časovače zabudovaného v čipu ATmega2560.[2] Díky tomu je možné programovat i čipy, které vyžadují externí oscilátor.

Implementace nahrávače je udělaná tak, aby ho v případě potřeby bylo možné snadno vyměnit za jiný. To bude nutné třeba pro přidání podpory HVSP či HVPP, neboť tyto protokoly Arduino ISP



Obrázek 2.6: Diagram identifikace speciálních pinů

nepodporuje.

3. Technická dokumentace

3.1 Použití

Univerzální sériový programátor nevyžaduje instalaci žádného dalšího programu pro použití. Z vnějšího pohledu se chová stejně jako Arduino ISP, které je široce podporované. Po zapojení do počítače může chvíli (několik sekund) trvat, než se USP inicializuje a zahájí komunikaci s počítačem přes sériový port[1]. Poté, co se na obrazovce USP objeví nápis 'Ready'je USP připraven k vložení čipu. Ten může být vložen s libovolnou orientací a na libovolné místo. Aktuálně by měly být podporovány prakticky všechny čipy řad Atmega a ATtiny v provedení DIP.[2]

Ihned po vložení čipu a jeho zajištění páčkou probíhá detekování. Během tohoto procesu se programátoru nedotýkejte, mohlo by to způsobit špatnou detekci čipu. Po detekování se na displeji ukáže jméno detekovaného čipu.[3] V této chvíli je již možné do čipu nahrávat stejným způsobem jako do normálního zařízení Arduino ISP.

Po vyndání čipu je možné stiskem libovolného tlačítka resetovat USP do původního stavu a připravit ho na další čip, který je možné vložit poté, co se na displeji objeví 'Ready'.

3.1.1 Použití s Arduino IDE

Použití s Arduino IDE je pravděpodobně nejjednodušší. Jako příklad řekněme, že chceme nahrát program do čipu ATtiny45. Prvním krokem bude zvolit Arduino ISP jako programátor pod Nástroje>Programátor>Arduino as ISP a zvolit Port do kterého je USP připojen. V nabídce portů se bude USP ukazovat pod jménem Arduino Mega. Dále je potřeba zadat informace o čipu pro kompilaci programu v Nástroje>Vývojová deska a navolit další nastavení jako frekvenci oscilátoru. Je možné, že bude nutné stáhnout knihovnu pro kompilaci na čip, pro ATtiny45 např. ATTinyCore.[7] Možné nastavení je vidět na obrázku 3.1.

V případě, že je toto první nahrávání do konkrétního čipu bude potřeba vypálit bootloader a nastavit čip. To lze v Arduino IDE udělat pod Nástroje>Vypálit zavaděč. Poté je již možné nahrávat do čipu pod Projekt>Nahrát pomocí programátoru nebo pomocí zkratky Ctrl+Shift+U.

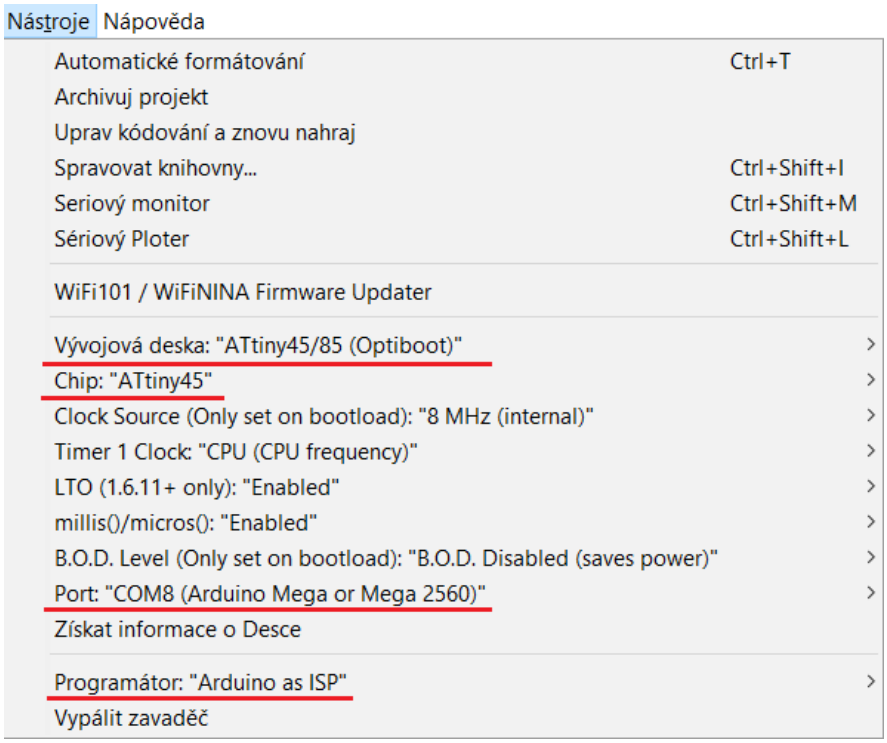
3.1.2 Použití s AVRDUDE

Podobně jako s Arduino IDE, lze nahrávat i programem avrdude, případně s použitím UI verze zvané AVRDUDESS. Opět je nutné pouze zvolit port, programátor Arduino, cílový čip a tentokrát i rychlost komunikace 19200 baudů. Nastavení pro ATtiny45 je na obrázku 3.2. Hlavní výhodou

[1] Ke komunikaci je nutné mít ovladače, ty se ale nainstalují samy při instalaci Arduina IDE, AVRDUDE nebo podobných programů

[2] Fyzicky jsem však testoval jen dva různé čipy ATtiny a jednu ATmegu, takže je možné, že pro určité specifické čipy bude potřeba upravit parametry v kódu, více o tom v sekci 3.2.

[3] Jméno na displeji se může lišit od reálného použitého čipu, neboť existuje mnoho variant čipů se stejným rozložením. Na displeji je zobrazeno jméno nejznámnějšího čipu s daným rozložením nebo čipu s největší pamětí.

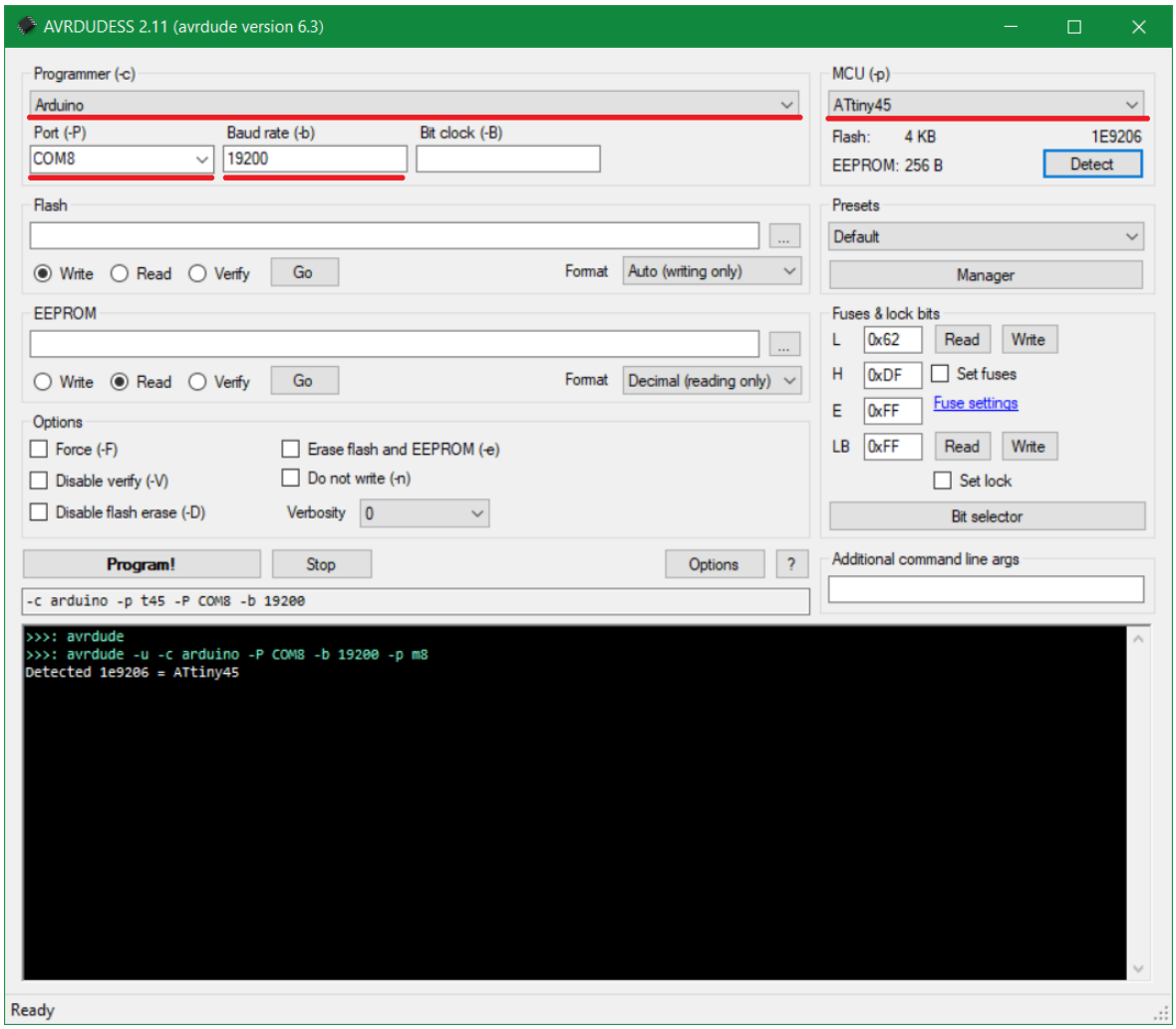


Obrázek 3.1: Nastavení Arduino IDE pro nahrávání do čipu ATtiny45; zvýrazněny jsou nejdůležitější parametry.

avrdude je, že pomocí něj jsme schopni nastavovat a číst fuse bits, číst a zapisovat do EEPROM či flash paměti či nastavovat lock bits. Lze pomocí něj do určité míry oživovat i zdánlivě mrtvé čipy.

3.2 Nahrávání od USP

Deska USP chrání řídící mikroprocesor na Arduino Mega proti nechtěnému flashování. V případě, že bude potřeba do něj nahrávat, je nutné nejdříve odstranit žlutý jumper s označením 'self prog', který se nachází na druhé straně desky než display a tlačítka. Poté bude již možné nahrávat do Arduino Mega standardním způsobem.



Obrázek 3.2: Nastavení AVRDUDESS pro nahrávání do čipu Attiny45, zvýrazněny jsou nejdůležitější parametry.

Závěr

Během práce na projektu jsem toho mnoho naučil nejenom o AVR čipech a CMOS obvodech, které jsem využíval, ale také o elektrotechnice obecně. Při zpětném pohledu bych některé věci implementoval na hardwarové úrovni jinak, čímž bych se vyhnul nutnosti dělat hotfixy na vyrobené desce a napadají mě i dodatečné funkce, které se současným hardwarem budou obtížně proveditelné.

Musím říct, že jsem měl při implementaci a návrhu i dostatek štěstí. Přestože jsem se snažil většinu základních věcí testovat dopředu, u pár věcí jsem spíše doufal a tipnul si, že budou fungovat, než že bych je měl detailně prověřené. A nakonec snad všechny z těchto přání fungovali tak, jak měly.

Projektu by v této fázi prospělo důkladné testování na všech různých čipech a doladění parametrů detekce. Já jsem ho testoval pouze na několika čipech, které jsem měl po ruce. Přesto si myslím, že by měl být schopen s minimálním zásahem podporovat programování daleko většího počtu různých čipů a možná i od jiných výrobců než Atmel.

Hlavním cílem projektu bylo vytvořit hardwarový výrobek, který bude reálně užitečný a tento cíl projekt určitě splnil a dovolím si tvrdit, že splnil zadání práce. Na hardwarové úrovni je implementování i několik bonusových funkcí, jako například podpora vysokonapěťového programování, které pouze čekají na implementaci v softwaru.

Seznam použité literatury

[1] Arduino. *Arduino Mega*. URL: https://www.arduino.cc/en/pmwiki.php?n=Main/ArduinoBoardMega (cit. 15. 03. 2022).

[2] Atmel. *8-bit Atmel Microcontroller with 16/32/64KB In-System Programmable Flash*. Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V. Datasheet 2549Q–AVR–02/2014.

[3] Atmel. *8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash*. ATmega328P. Datasheet 7810D-AVR-01/15.

[4] Atmel. *Atmel 8-bit AVR Microcontroller with 2/4/8K Bytes In-System Programmable Flash*. ATtiny25/V / ATtiny45/V / ATtiny85/V. Datasheet 2586Q-AVR-08/2013.

[5] Texas Instruments. *CD405xB CMOS Single 8-Channel Analog Multiplexer/Demultiplexer with Logic-Level Conversion*. CD4051B. Datasheet SCHS047I, Rev. 2017.

[6] Texas Instruments. *CD4066B CMOS Quad Bilateral Switch*. CD4066B. Datasheet SCHS051H, Rev. 2020.

[7] Spence Konde. *ATTinyCore*. URL: https://github.com/SpenceKonde/ATTinyCore (cit. 17. 03. 2022).

[8] Jasper van Loenen. *image2cpp*. URL: https://javl.github.io/image2cpp/ (cit. 17. 03. 2022).

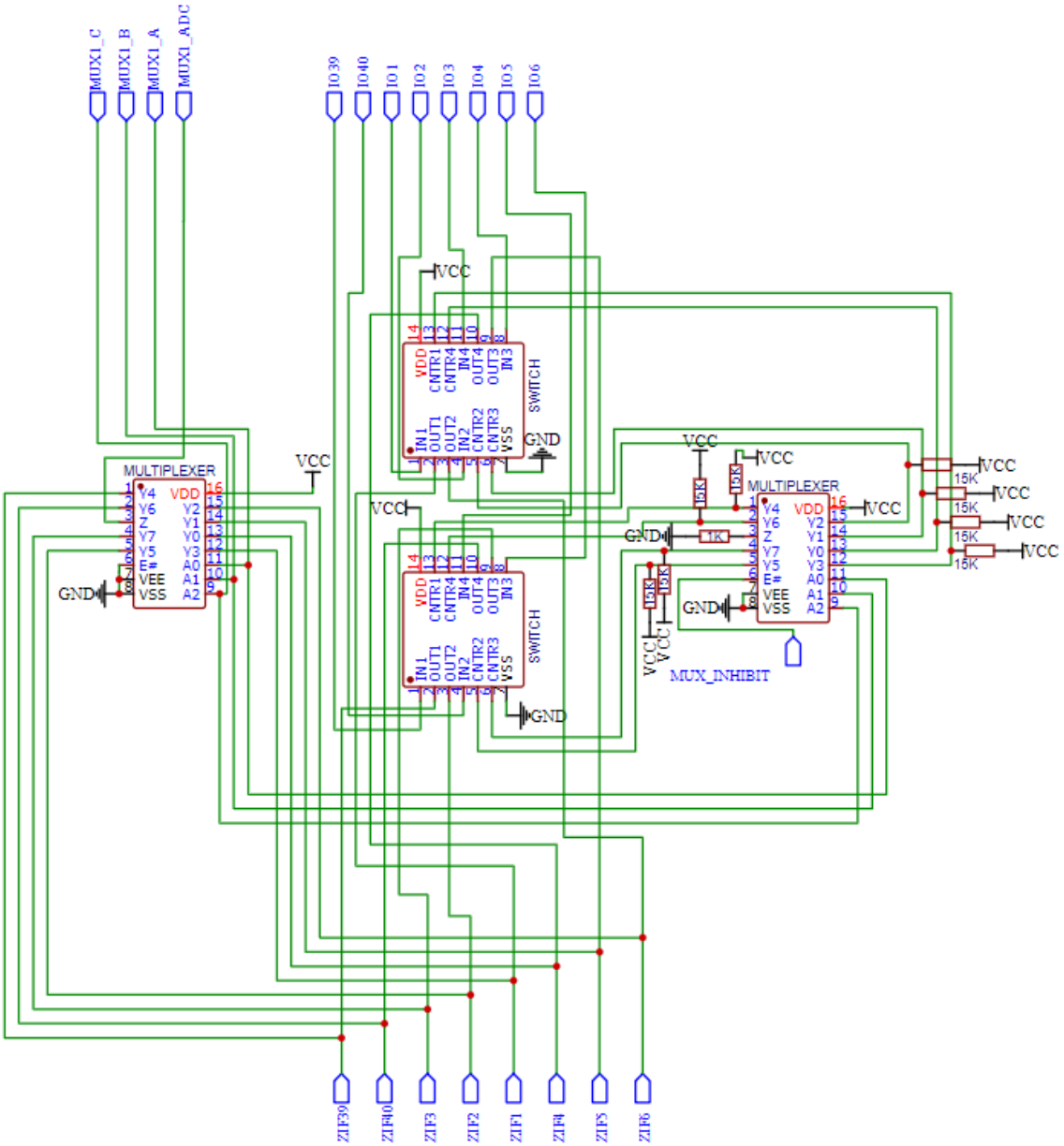
[9] Matthias Neeracher. *ScratchMonkey - Overview*. 2016. URL: https://microtherion.github.io/ScratchMonkey/Overview.html (cit. 14. 03. 2022).

[10] ON Semiconductor. *1.5 A, Step-Up/Down/ Inverting Switching Regulators*. MC34063A. Datasheet MC34063A/D, Rev. 2010.

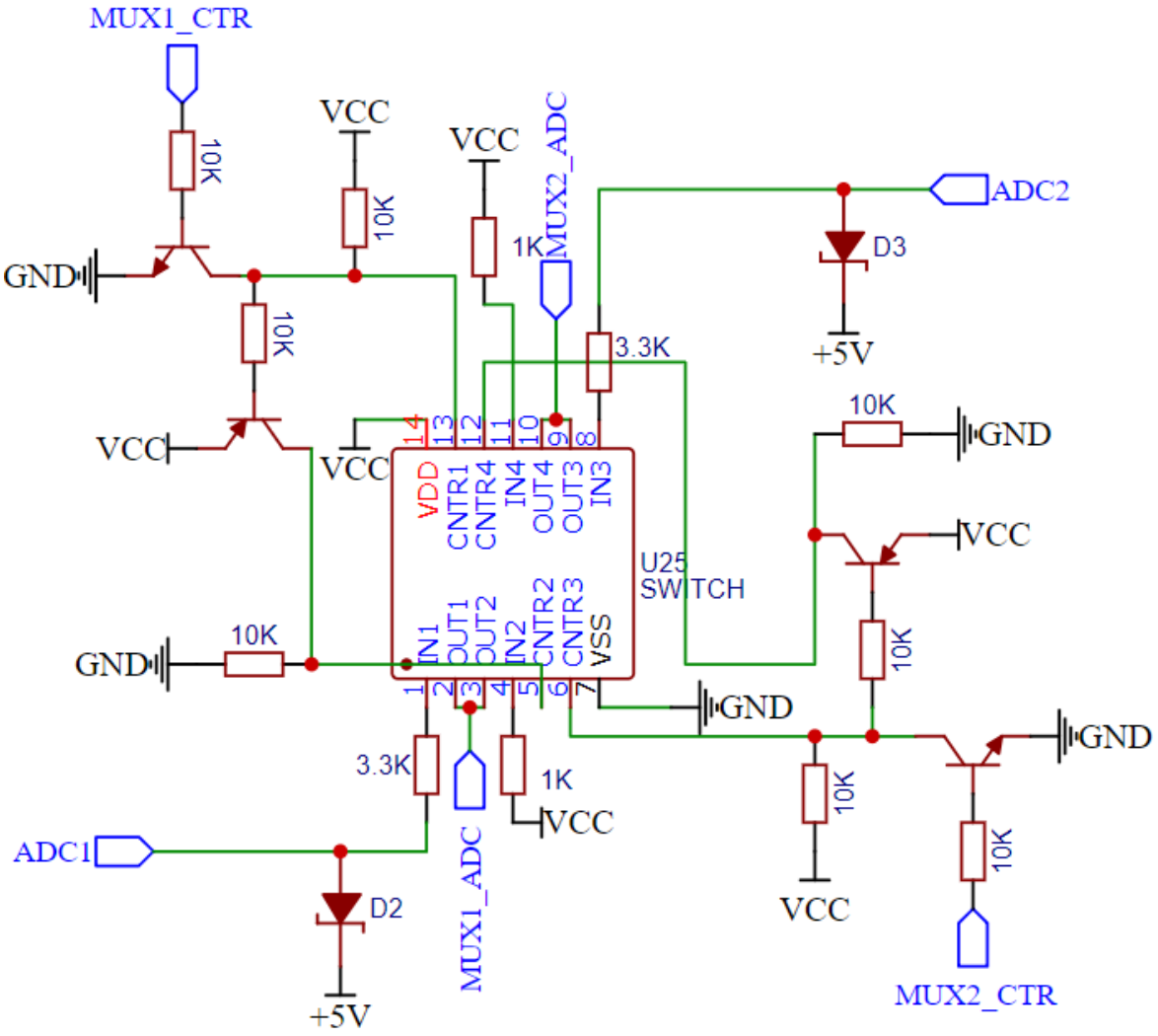
[11] ON Semiconductor. *Quad Analog Switch/Quad Multiplexer*. MC14066B. Datasheet SCHS047I, Rev. 2017.

Seznam obrázků

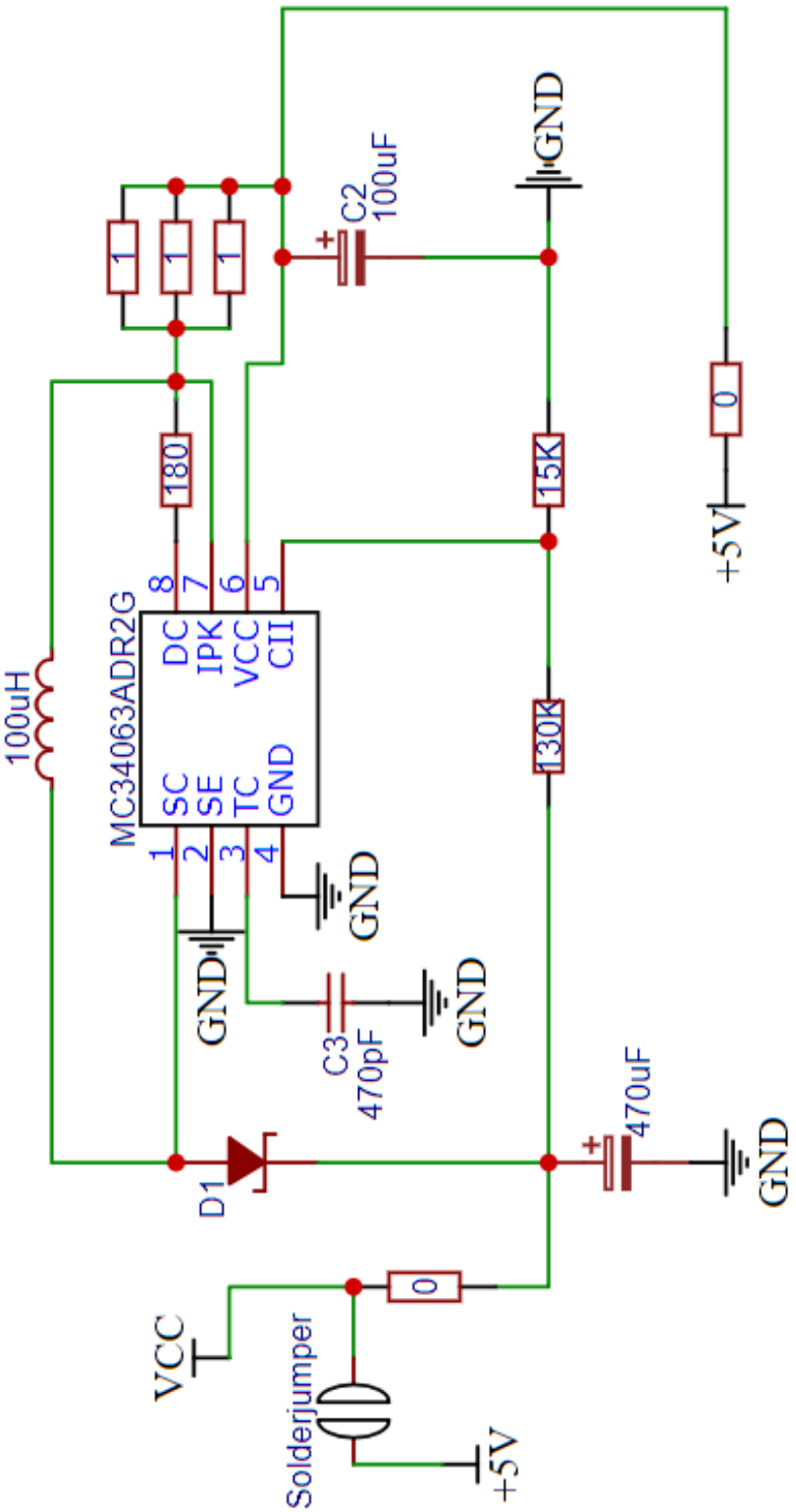
Přílohy



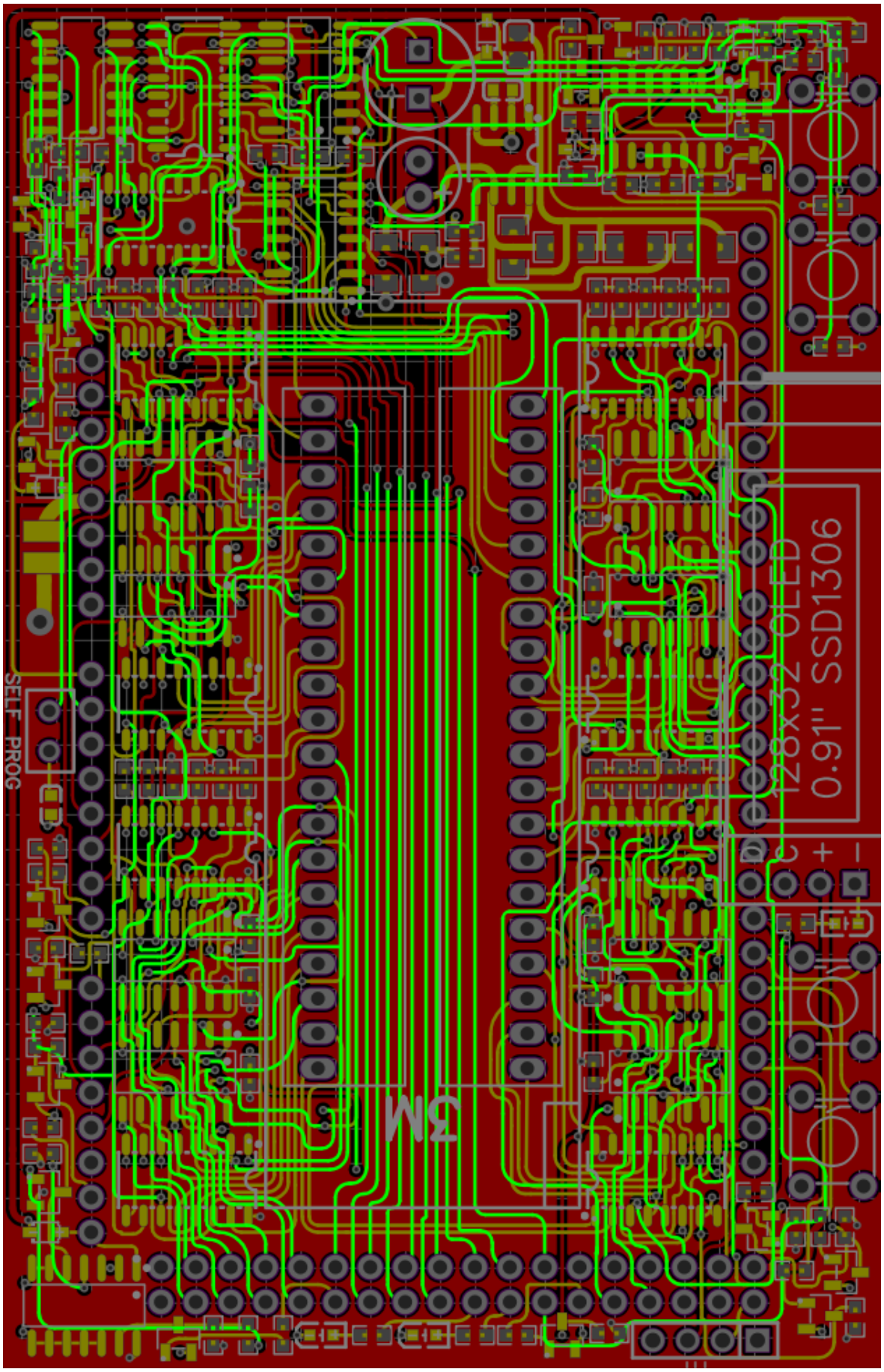
Obrázek A.1: Úplné schéma obvodu připojeného k 8 pinům ZIF patice.



Obrázek A.2: Úplné schéma zapojení switche, který přepíná vstup do multiplexerů 1 a 2 mezi ADC pinem a 12V.



Obrázek A.3: Úplné schéma obvodu step-up převodníku z 5 na 12 voltů. Obvod zahrnuje o ohmové odpory a solder jumper, které umožňují jednoduché odpojení převodníku.



Obrázek A.4: Zapojení plošného spoje. Žlutou barvou jsou vyznačeny spoje ve vrchní vrstvě, zelenou spoje spodní vrstvy a červenou spoje jedné z vnitřních vrstev. Druhá vnitřní vrstva není na obrázku zachycena.